Universitat
Autònoma
de Barcelona

(3858: Integración del Archivo de Imágenes de un Telescopio Robótico al Observatorio Virtual.)

Memoria del Proyecto de Fin de Carrera
Ingenieria en Informática
realizado por
Guillem Pérez Delgado
y dirigido por
Xavier Otazu i Porter y *Octavi Fors*
Bellaterra, 9 de septiembre de 2011

CERTIFICACIÓ DE DIRECCIÓ

El sotasignat, *Xavier Otazu i Porter*

Professor/a de l'Escola Tècnica Superior d'Enginyeria de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en *Guillem Pérez Delgado*

I per tal que consti firma la present.

Signat:

Bellaterra, 9 de septiembre de 2011

CERTIFICACIÓ DE DIRECCIÓ EN EMPRESA

El sotasignat, *Dr. Octavi Fors*

del **Departament d'Astronomia i Meteorologia** de la UB (Universitat de Barcelona)

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat en l'empresa sota la seva supervisió mitjançant conveni.

firmat amb la Universitat Autònoma de Barcelona.

Així mateix, l'empresa en té coneixement i dóna el vist-i-plau al contingut que es detalla en aquesta memòria.

Signat:

Cerdanyola, 9 de septiembre de 2011

Índice

1. Introducción

Este proyecto nace de la necesidad de mostrar y distribuir todas las imágenes tomadas por el Telescopi Fabra ROA Montsec (TFRM) a la comunidad Astronómica.

El TFRM es una colaboración entre el Departament d'Astronomia i Meteorología de la UB, la Reial Academia de Ciències i Arts de Barcelona y el Real Instituto y Observatorio de la Armada en San Fernando, con la que se pretende dotar de la tecnología necesaria a un telescopio para que pueda operar robóticamente y remotamente, es decir, sin personal científico en las instalaciones. Dicho telescopio se encuentra ubicado en el Montsec.

La parte que concierne a este proyecto será la de implementar un archivo de datos astronómicos VO (Virtual Observatory), para compartir el conjunto de imágenes utilizando los estandares definidos por la organización IVOA (Internacional Virtual Observatories Alliance).

1.1. Objetivos

El objetivo principal es la implementación de un servicio VO (Virtual Observatory). Este servicio esta compuesto por dos grandes bloques, Ingestor y Comunicador SIA. Ambos con su aplicación correspondiente.

Para dicha implementación se han establecido los siguientes objetivos.

1. Diseño tanto de las aplicaciones como de la base de datos.
   - Dar soporte al diseño de la base de datos, para una correcta implementación.
   - Diseño modular y estructurado de la aplicación Ingestor. Ésta debe ser configurable para un posible tratamiento posterior y adaptable a los distintos medios con los que trabaja.
   - Diseño estructurado del Comunicador SIA (Simple Image Access).
2. Implementación de la solución final.
3. Integración del servicio VO a las instalaciones del TFRM.

4. Realización de un exhaustivo testeo para verificar que todo funciona correctamente.

Para alcanzar estos objectivos es necesario realizar ciertas labores de formación, como son:

- Adquisición de conocimientos necesarios para la implementación de las aplicaciones.
  - Conocimientos teóricos básicos en astronomía, como coordenadas celestes, algoritmos de transformación e información básica relacionada con las imágenes tomadas con telescopio.
  - Estudio del protocolo SIAP (Simple Image Access Protocol) usado para la comunicación entre el Archivo e IVOA.
  - Conocimientos informáticos, como tecnología «XML» y «web services», tecnología de base de datos y lenguajes de programación adecuados.
- Estudio de las tecnologías y librerías necesarias para la implementación, tanto de la base de datos, como de las propias aplicaciones.

1.2. Motivaciones del proyecto

Uno de los principales motivos que me llevaron a realizar este proyecto, es el simple hecho de implementar una aplicación relacionada con el mundo científico, y especialmente con el mundo astronómico, que desde siempre me ha atraído.

Quitando la parte emotiva del proyecto, éste requiere de una serie de tecnologías, de las cuales no estaba muy familiarizado y que por su contra, suponía un reto a superar, brindándome además la posibilidad de aumentar mis conocimientos como programador. Otra motivación ha sido la de implementar el Ingestor como un tipo de aplicación modularizada y estructurada, capaz de ser modificable con muy pocos cambios en el propio código. Esto implicaba un diseño inicial meticulosamente estudiado para evitar fallos tanto de código como de ejecución.

El encanto que tiene el mundo celeste junto con la complejidad de la implementación han hecho de este proyecto, el ideal para un programador con ganas de usar el Proyecto Final de Carrera como un modo más de aprender nuevas tecnologías y ámbitos.

1.3. Descripción de la solución

La solución propuesta permitirá distribuir las imágenes tomadas por el telescopio TFRM a través del VO de acuerdo con los estándares descritos en IVOA. VO ya dispone de una serie de aplicaciones ya implementadas que permiten conectar con los distintos observatorios registrados y que disponen de este servicio, descargándose las imágenes que contengan la porción de cielo que se desee.

Este proyecto, implica todas aquellas aplicaciones que se encuentran en el observatorio y que son necesarias para comunicarse con VO. Ello conlleva varias partes que se deberán realizar. A continuación se procederá a describir brevemente, cada una de las funcionalidades que realiza la solución propuesta.

- Base de datos que almacena toda la información de las imágenes tomadas con el telescopio. Dicha información, esta relacionada tanto por los propios parámetros de las imágenes, como por los valores que facilitan la búsqueda de las mismas.
- Aplicación encargada de ingerir las imágenes tomadas por el telescopio. Esta aplicación realiza diversas tareas:
  - capturar la meta-información necesaria de cada una de las imágenes e introducirla en la base de datos.
  - realizar un testeo de integridad consiguiendo un archivo de imágenes coherente.
  - reubicar las imágenes según convenga al administrador.
- Aplicación encargada de comunicarse con el servicio proporcionado por IVOA. Pasos a seguir:
  - recibe una petición con la zona de interés que el cliente desea observar.
  - búsqueda de dicha zona de interés en la base de datos.
  - retorno de XML con la información encontrada.
- Aplicación encargada de enviar, vía Internet, las imágenes seleccionadas por el cliente.

VO dispone de distintos tipos de archivo. Estos tipos son dados dependiendo del tipo de servicio e imágenes se desean proporcionar. El archivo TFRM entra en la categoria de “Atlas Image Archive”, lo que implica que TFRM proporcionará imágenes calibradas que contienen una porción grande del cielo.

1.4. Entorno de la solución

El TFRM se ubica en Àger, y dispone de un telescopio con una cámara Baker-Nunn (BNC) de relación focal f/1 y apertura 50cm, controlada robóticamente con fines astronómicos. La instalación se ha creado gracias a una colaboración entre Reial Acadèmia de Ciències i Arts de Barcelona (RACAB) - Observatori Fabra, el Real Instituto y Observatorio de la Armada (ROA) y el Departament d'Astronomia i Meteorologia de la Universitat de Barcelona (DAM). [1]

Figura 1: Instalaciones del TFRM.

Por otro lado, IVOA se creó para facilitar la coordinación internacional y la colaboración necesaria para el desarrollo y la implementación de las herramientas, sistemas y estructuras organizativas necesarias para permitir la utilización internacional de los archivos astronómicos como un observatorio virtual integrado e interoperable. Principalmente, IVOA se centra en el desarrollo de estándares y promueve la creación de aplicaciones en beneficio de la comunidad astronómica mundial. [2]

Figura 2: Ubicación geográfica de IVOA.

Uno de estos estándares/aplicaciones es el VO (Virtual Observatory). Un proyecto innovador y vanguardista, que pretende comunicar todos los observatorios con todos y cada uno de los científicos que requieran de imágenes tomadas por telescopio. Esto se consigue a través de la cooperación de los distintos archivos y aplicaciones ubicados en las instalaciones astronómicas que proporcionan el servicio [3].

SIA (Simple Image Access) representa la negociación entre el cliente y el servicio de imágenes. El cliente describe la imagen ideal y el servicio de imágenes retorna un listado con las imágenes que cumplen los requisitos [4].

Aladin Sky Atlas, es una aplicación ampliamente usada por astrónomos y profesionales que permite visualizar las imágenes astronómicas digitalizadas, superponer a estas las entradas de los catálogos o bases de datos astronómicas y acceder de forma interactiva a la base de datos y a otros archivos de todas las fuentes conocidas en este campo, como VO. [5]

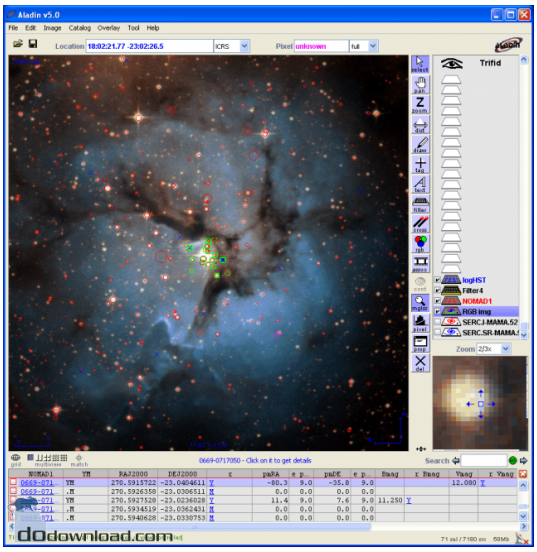


Figura 3: Interfaz de la aplicación Aladin Sky Atlas.

1.5. Herramientas usadas

Para el desarrollo de este proyecto se han usado distintas herramientas de licencia libre y que se pueden encontrar en el mercado. Estas herramientas que se describirán a continuación, son tanto lenguajes de programación, como librerías y aplicaciones que se han usado.

- OpenSUSE
- Servidor apache
- PostGreSQL
- pgSphere
- nom.tam.fits
- JAVA
- PHP
- XML

1.5.1. Opensuse

Se escogió Linux por su fiabilidad en los servidores y máquinas permanentemente conectadas, por su fácil configuración y adaptación a los distintos medios y dispositivos en el que se encuentra el proyecto y por su accesibilidad a los distintos módulos que forman el sistema operativo. Si a ello se le añade que es una distribución gratuita y soportada por una amplia comunidad de programadores, obtenemos el sistema operativo ideal para el proyecto.

Dentro de Linux, la distribución OpenSUSE. Como el resto de las distribuciones, tiene sus ventajas y desventajas, que no procederemos a describir.

1.5.2. Servidor Apache

Es el servidor más usado en la red desde el 1996 donde su ventajas son que es modular, multi-plataforma, extensible y Open Source. Este ultimo punto y el conocimiento popular de este servidor es el que ha decantado su uso en este proyecto. Cabe añadir que también ha sido usado dada la gran cantidad de información que hay en la red al respecto de dicho servidor.

1.5.3. PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional (ORDBMS) basado en el proyecto POSTGRES, de la universidad de Berkeley. Es una derivación libre (OpenSource) de este proyecto, y utiliza el lenguaje SQL92/SQL99, que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. Dicho sistema de gestión fue elegido por su gran capacidad de procesamiento y porque es uno de los sistemas de gestión de base de datos mas usados en los distintos entornos en que se encuentra este proyecto.

PostgreSQL dispone de una ventaja sobre los demás sistemas y su gran librería de módulos extras de que dispone. Uno de ellos es el pgSphere que también ha sido usado.

1.5.4. pgSphere

pgSphere es un modulo extra de PostgreSQL el cual añade a la base de datos una serie de tipos de datos y una serie de funciones de búsqueda relacionados con la trigonometría esférica.

Dicho módulo ayuda a realizar búsquedas rápidas y análisis de objetos con coordenadas esfèricas como es el caso de las imágenes tomadas por el telescopio[6].

1.5.5. JAVA

Lenguaje de programación orientado a objetos e interpretado, es conocido por su fácil uso y modularidad de las aplicaciones.

Uno de los objetivos del proyecto era la modularización de las aplicaciones para su uso y modificación posterior, lo que hace de este lenguaje de programación el ideal para la creación del mismo.

1.5.6. nom.tam.fits

Librería JAVA usada para tratar con las imágenes tomadas por el telescopio. Estas imágenes son en formato FITS y contienen una cabecera que almacena información relacionada con las coordenadas celestes, variables astronómicas y cualquier dato necesario para su interpretación.

Esta librería ha sido usada principalmente para la captura de la cabecera y su posterior tratamiento [12].

1.5.7. PHP

PHP (Hypertext Pre-processor) es un lenguaje interpretado e integrado en las paginas HTML, que se ejecuta en el servidor, permitiendo desarrollar paginas dinámicas. PHP ha sido usado para la creación de las distintas páginas que se comunican con VO, recibiendo peticiones, realizando búsquedas a la base de datos y devolviendo la información necesaria en formato XML.

1.5.8. XML

Metalenguaje extensible de etiquetas que permite definir la gramática de lenguajes específicos, no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Es usado por requisito de IVOA, ya que el protocolo SIAP define el modo en que se debe transportar la información entre los distintos agentes de la comunicación.

2. Especificaciones

Este apartado describe los requisitos iniciales y necesarios para la implementación del servicio VO en el TFRM.

2.1. Requerimientos

Requisitos Ingestor :

1. Deberá ser capaz de comunicarse con un sistema de Base de Datos instalado en el mismo ordenador, donde los datos del administrador de la base de datos se almacenen en un fichero de configuración.
2. Ubicaciones físicas de las imágenes configurables. La aplicación deberá obtener las imágenes y moverlas una vez procesadas, según un fichero de configuración que almacenará dichas ubicaciones físicas.
3. Integridad del archivo. Antes de ingerir una imagen, se deberá realizar un testeo para verificar que ésta es ingerida correctamente.
4. Tratamiento con formato FITS. Las imágenes tomadas por el telescopio se encuentran en formato FITS.
5. Control de Errores y Log. La aplicación deberá almacenar en un fichero y en la base de datos, cualquier movimiento que realicen las imágenes y cualquier error producido durante la ejecución.
6. Captura de datos especificados por el protocolo SIAP. Esto implica que las imágenes deberán disponer de la información necesaria y requerida para una correcta comunicación con VO.
7. Ejecución automática y diaria del proceso de ingestión de imágenes. El proceso de ingestión de imágenes se deberá realizar una vez al día, principalmente durante las horas diurnas.
8. Modularización. La aplicación deberá ser fácilmente modificable para poder adaptarse a las posibles modificaciones realizadas en las imágenes FITS.

Requisitos comunicador SIA :

- El servicio deberá contemplar todo aquello que concierne a la comunicación entre VO y un archivo de tipo «Atlas Image Archive». Especificado en la documentación SIAP [4]. Esto implica tanto requisitos de entrada de datos como formatos de salida de los mismos.

3. Diseño

3.0.1. Introducción general

A continuación se mostrará un esquema general, con los distintos puntos que forman el servicio VO y su relación. (figura 4)

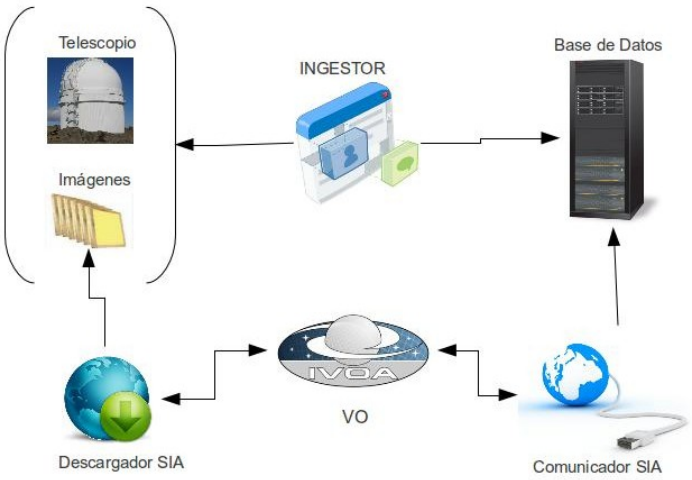


Figura 4: Esquema General.

Como se puede ver en el esquema, el Ingestor captura las imágenes tomadas por el telescopio y las inserta en la base de datos, manteniendo el archivo al día. El servicio SIA busca en la base de datos toda la información que VO necesita y el Descargador envía la imagen que el mismo VO ha solicitado descargar.

3.1. Esquema de las base de datos

La figura figura5 muestra el esquema físico que tiene la base de datos usada en el servicio.

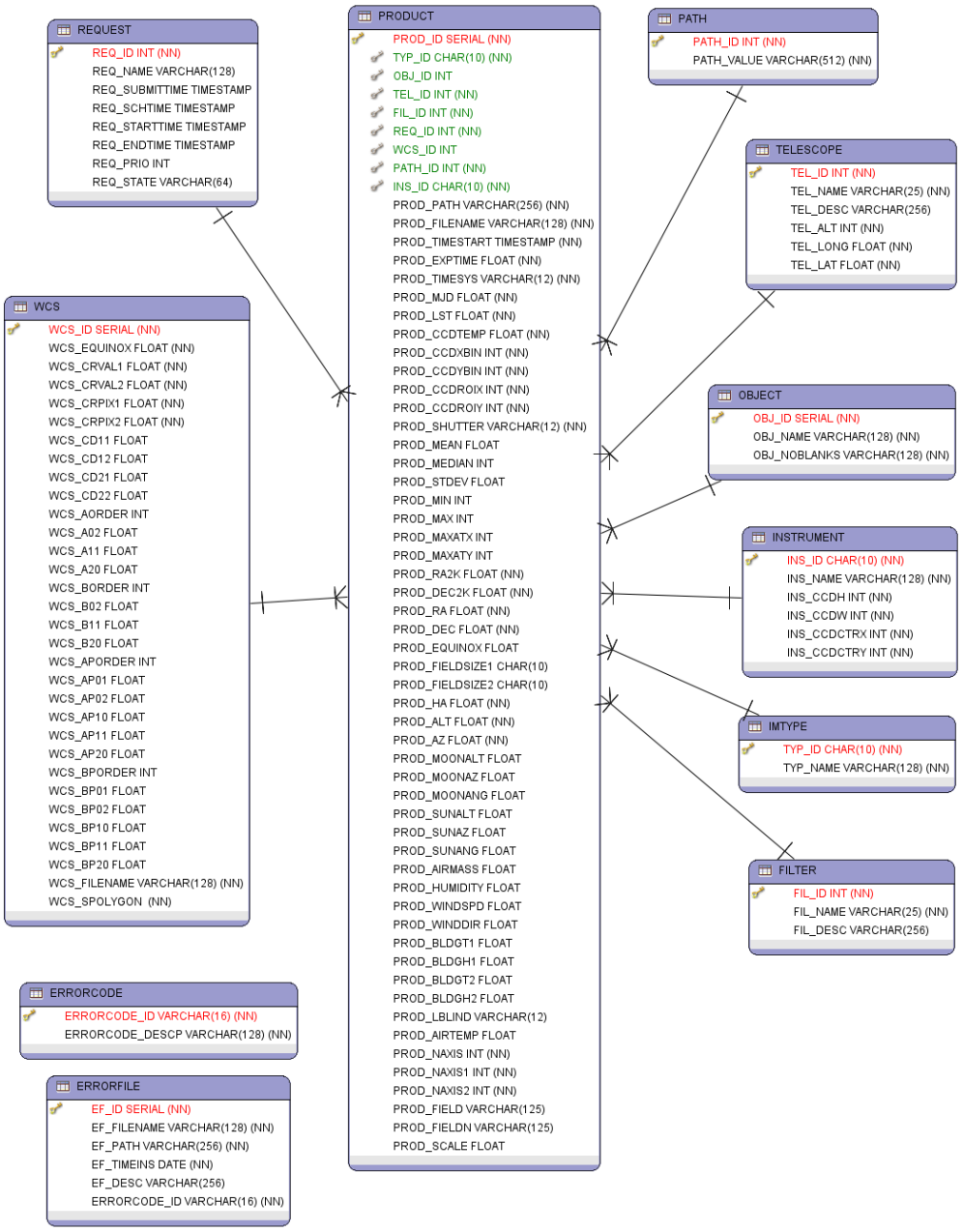


Figura 5: Diagrama de Tablas de la Base de Datos.

El diseño del diagrama de tablas, ha sido realizado entre el tutor del TFRM Dr. Octavi Fors (DAM, RACAB) y los colaboradores Sr. Raúl Gutiérrez y Dr. Enrique Solano, del Centro de Astrobiología - Laboratorio de Astrofísica Espacial y Física Fundamental (CAB-LAEFF). En lo que concierne al proyecto, se ha integrado dicho diseño a la base de datos PostgreSQL.

3.1.1. Descripción de las tablas

Errorcode Almacena los posibles errores que se puedan realizar en el sistema de ingestión. Tabla estática.

Column name	Type	Length	PK	NOT NULL	FK
errorcode_id	varchar(n)	16	*	*	
errorcode_descp	varchar(n)	128		*	

El campo *errocode_id* contiene el identificador del error, mientras *errocode_descp* almacena la descripción del error sucedido.

Errorfile Registro de todos los errores que se han producido durante la ejecución. Relación fichero con el tipo de error. Tabla dinámica.

Column name	Type	Length	PK	NOT NULL	FK
ef_id	serial		*	*	
ef_filename	varchar(n)	128		*	
ef_path	varchar(n)	256		*	
ef_timeins	date			*	
ef_desc	varchar(n)	256			
errorcode_id	varchar(n)	16		*	

El campo *ef_filename* y *ef_ path* contiene el nombre del fichero y su ubicación, *ef_timeins* el momento en que se ha producido el error y *errorcode_id* el código del error que se ha producido.

Filter Almacena los distintos tipos de filtros que dispone el telescopio. Tabla estática.

Column name	Type	Length	PK	NOT NULL	FK
fil_id	int		*	*	
fil_name	varchar(n)	25		*	
fil_desc	varchar(n)	256			

El campo *fil_id* contiene el identificador del nombre, *fil_name* el nombre y *fil_desc* almacena una breve descripción del filtro.

Imtype Almacena los distintos tipos de imágenes que puede gestionar la aplicación, Darks, Flats, Bias, Science, etc... Tabla estática.

Column name	Type	Length	PK	NOT NULL	FK
typ_id	char(n)	10	*	*	
typ_name	varchar(n)	128		*	

El campo *typ_id* guarda el identificador del tipo de imagen y *typ_name* contiene el nombre del tipo de la imagen, es decir, bias, flat, dark, science o wcs.

Instrument Guarda los distintos instrumentos con los que se toman las imágenes. Tabla estática.

Column name	Type	Length	PK	NOT NULL	FK
ins_id	char(n)	10	*	*	
ins_name	varchar(n)	128		*	
ins_ccdh	int			*	
ins_ccdw	int			*	
ins_ccdctrx	int			*	
ins_ccdctry	int			*	

El campo *ins_id* guarda el identificador del instrumento y *ins_name* el nombre. El resto de parámetros son usados para almacenar información relacionada con las medidas del instrumento, como altura y anchura del ccd, entre otros.

Object Registro de los distintos objetos celestes que toma el telescopio. Tabla dinámica.

Column name	Type	Length	PK	NOT NULL	FK
obj_id	serial		*	*	
obj_name	varchar(n)	128		*	
obj_noblanks	varchar(n)	128		*	

El campo *obj_id* contiene el identificador del objeto observado, *obj_name* el nombre del mismo y *obj_noblanks* almacena el nombre del objeto sin espacios en blanco.

Path Registro de las ubicaciones físicas en las que se pueden encontrar las imágenes. Tabla dinámica.

Column name	Type	Length	PK	NOT NULL	FK
path_id	int		*	*	
path_value	varchar(n)	512		*	

El campo *path_id* contiene el identificador del path y *path_value* la ruta física a la que representa el identificador.

Telescope Nombre del telescopio. Dicha tabla permite la utilización de varios telescopios para un mismo archivo. Tabla estática.

Column name	Type	Length	PK	NOT NULL	FK
tel_id	int		*	*	
tel_name	varchar(n)	25		*	
tel_desc	varchar(n)	256			
tel_alt	int			*	
tel_long	float			*	
tel_lat	float			*	

Los campos *tel_id* y *tel_name* contiene el identificador y el nombre del telescopio. Mientras que *tel_alt*, *tel_long* y *tel_lat* almacenan la ubicación geográfica en la que se encuentra el telescopio.

Request Registro de las peticiones que agrupan un conjunto de imágenes. Principalmente sirve para controlar peticiones que se realizan al telescopio. Tabla dinámica

Column name	Type	Length	PK	NOT NULL	FK
req_id	int		*	*	
req_name	varchar(n)	128			
req_submittime	timestamp				
req_schtime	timestamp				
req_starttime	timestamp				
req_endtime	timestamp				
req_prio	int				
req_state	varchar(n)	64			

Los campos *req_id*, *req_name* y *req_submittime* contienen el identificador, el nombre y el momento en que se realizó la petición. El resto de campos hacen referencia a información relacionada con la petición.

Product Tabla general que almacena los datos relacionados con las imágenes ingestadas. Guarda un registro de todos los parámetros de las cabeceras FITS. Tabla dinámica.

Column name	Type	Length	PK	NOT NULL	FK
prod_id	serial		*	*	
typ_id	char(n)	10		*	imtype.typ_id
obj_id	int				object.obj_id
tel_id	int			*	telescope.tel_id
fil_id	int			*	filter.fil_id
req_id	int			*	request.req_id
wcs_id	int				wcs.wcs_id
path_id	int			*	path.path_id
ins_id	char(n)	10		*	instrument.ins_id
prod_path	varchar(n)	256		*	
prod_filename	varchar(n)	128		*	
prod_timestart	timestamp			*	
prod_exptime	float			*	
prod_timesys	varchar(n)	12		*	

Column name	Type	Length	PK	NOT NULL	FK
prod_mjd	float			*	
prod_lst	float			*	
prod_ccdtemp	float			*	
prod_ccdxbin	int			*	
prod_ccdybin	int			*	
prod_ccdroix	int			*	
prod_ccdroiy	int			*	
prod_shutter	varchar(n)	12		*	
prod_mean	float				
prod_median	int				
prod_stdev	float				
prod_min	int				
prod_max	int				
prod_maxatx	int				
prod_maxaty	int				
prod_ra2k	float			*	
prod_dec2k	float			*	
prod_ra	float			*	
prod_dec	float			*	
prod_equinox	float				
prod_fieldsize1	char(n)	10			
prod_fieldsize2	char(n)	10			
prod_ha	float			*	
prod_alt	float			*	
prod_az	float			*	
prod_moonalt	float				
prod_moonaz	float				
prod_moonang	float				
prod_sunalt	float				
prod_sunaz	float				
prod_sunang	float				
prod_airmass	float				
prod_humidity	float				
prod_windspd	float				
prod_winddir	float				

Column name	Type	Length	PK	NOT NULL	FK
prod_bldgt1	float				
prod_bldgh1	float				
prod_bldgt2	float				
prod_bldgh2	float				
prod_lblind	varchar(n)	12			
prod_airtemp	float				
prod_naxis	int			*	
prod_naxis1	int			*	
prod_naxis2	int			*	
prod_field	varchar(n)	125			
prod_fieldn	varchar(n)	125			
prod_scale	float				

El campo *prod_id* contiene el identificador del producto.

Los campos *obj_id*, *tel_id*, *fil_id*, *ins_id*, *path_id* y *req_id* hacen referencia al objeto al que apunta el telescopio, el telescopio, filtro e instrumento usados para obtener el producto, la ubicación física donde se encuentra y la petición a la que esta relacionada.

wcs_id almacenará el identificador únicamente en el caso que el producto haga referencia a una imagen calibrada. Este atributo relaciona el producto con la información de calibración almacenada en la tabla «wcs».

Los campos *prod_path* y *prod_path* guardan la ubicación interna dentro del path genérico y el nombre del fichero al que esta vinculado este producto.

El resto de atributos hacen referencia a información de la imagen, como tiempo de exposición, momento en que se inició la observación, valores atmosféricos, Ascensión y Declinación Recta del punto central de la imagen, etc. Estos parámetros son usados principalmente por los astrónomos para la comprensión de la imagen tomada.

Wcs Tabla general que almacena los datos relacionados con las imágenes WCS ingestadas. En puntos posteriores se especificará mas al detalle dicho tipo de imágenes. Tabla dinámica.

Column name	Type	Length	PK	NOT NULL	FK
wcs_id	serial		*	*	
wcs_equinox	float			*	
wcs_crval1	float			*	
wcs_crval2	float			*	
wcs_crpix1	float			*	
wcs_crpix2	float			*	
wcs_cd11	float				
wcs_cd12	float				
wcs_cd21	float				
wcs_cd22	float				
wcs_aorder	int				
wcs_a02	float				
wcs_a11	float				
wcs_a20	float				
wcs_border	int				
wcs_b02	float				
wcs_b11	float				
wcs_b20	float				
wcs_aporder	int				
wcs_ap01	float				
wcs_ap02	float				
wcs_ap10	float				
wcs_ap11	float				
wcs_ap20	float				
wcs_bporder	int				
wcs_bp01	float				
wcs_bp02	float				
wcs_bp10	float				
wcs_bp11	float				
wcs_bp20	float				
wcs_filename	varchar(n)	128		*	
wcs_spolygon				*	

El campo *wcs_id* contiene el identificador de la información específica de una imagen calibrada. El resto de atributos almacenan distintos datos astronómicos relacionados únicamente con imágenes calibradas.

3.2. Diseño del Ingestor

A continuación se muestra el diagrama de clases de la aplicación Ingestor. Dicho esquema muestra cada una de las clases que dispone la aplicación. (figura 6)

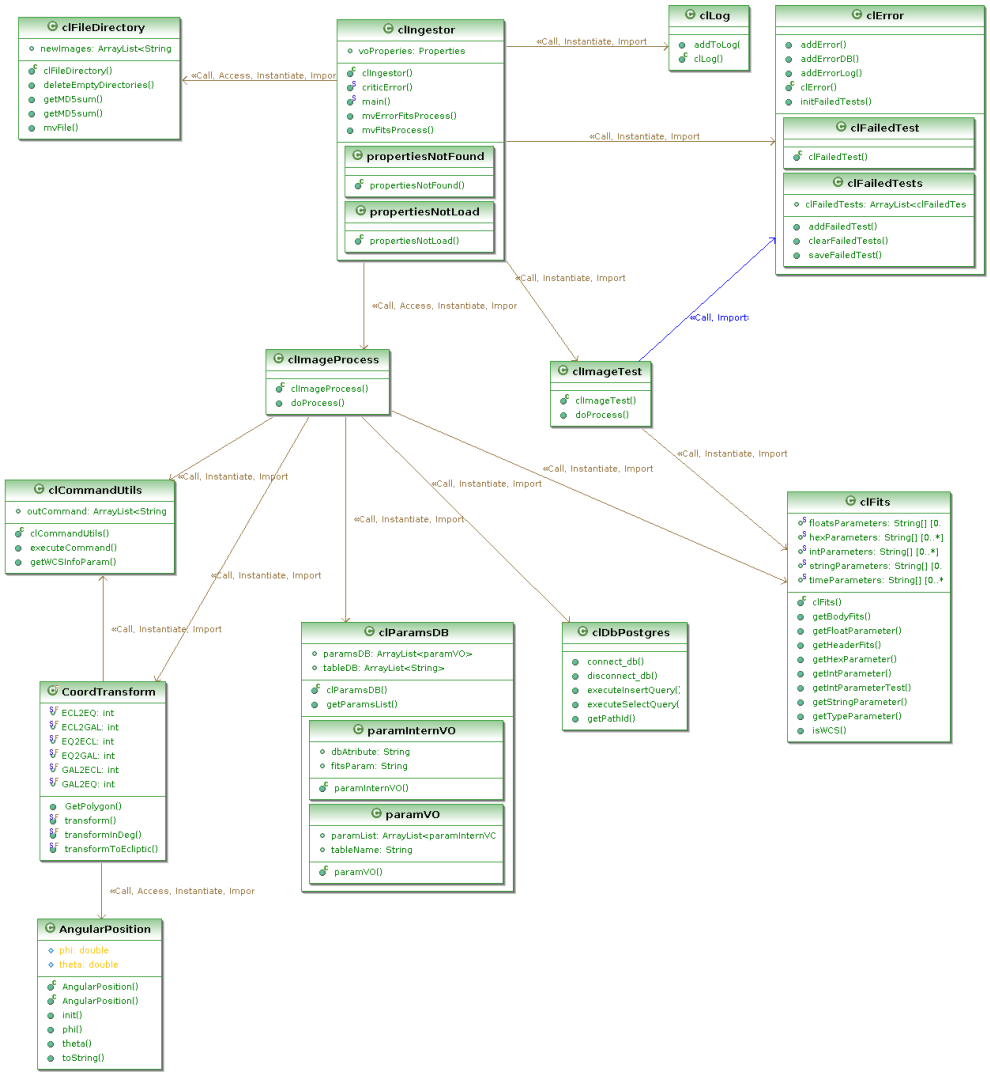


Figura 6: Diagrama de Clases de la aplicación Ingestor.

En el siguiente punto, se procederá a describir la funcionalidad de cada una de las clases que forman la aplicación.

3.2.1. Descripción de las clases

clIngestor clase principal, que realiza las llamadas según convenga el guión del proceso. También gestiona todos los posibles errores que se puedan producir.

clImageProcess encargada de procesar cada una de las imágenes que recibe. Captura de valores de la cabecera, adaptación de dichos valores según convenga e inserción en la base de datos.

clImageTest encargada de realizar el proceso de integridad de las imágenes. Contiene todos los testeos que se hacen en las imágenes.

clDbPostgres contiene la funcionalidad necesaria para acceder e insertar datos en la base de datos.

clFileDirectory contiene las funciones necesarias para obtener y mover las imágenes tomadas por el telescopio.

clFits contiene funciones para el tratamiento de imágenes FITS.

clParamsDB gestiona toda la parte de configuración del fichero de propiedades del Ingestor. Dicha clase sirve para conseguir que la aplicación sea adaptable a los posibles cambios producidos en las imágenes FITS.

clCommandUtils ha sido creada por motivos excepcionales, ya que realiza tareas de obtención de datos que a priori deberían estar almacenados en las cabeceras FITS.

clError Inserta los errores producidos en las ubicaciones necesarias, Base de datos y ficheros de error.

clLog Inserta los mensajes de Log que se producen durante la ejecución del Ingestor.

CoordTransform Clase que almacena la funcionalidad necesaria para el cambio de coordenadas celestes que se producen durante la ejecución.

AngularPosition Contiene el tipo de dato usado durante las transformaciones.

4. Implementación

4.1. Base de la aplicación

A continuación, se describirá ciertos puntos teóricos necesarios para la comprensión del entorno y temática del proyecto.

4.1.1. Base teórica de la aplicación

Una coordenada celeste es un parámetro que define una posición en la esfera celeste.

Se han usado dos tipos de coordenadas celestes:

- *Coordenadas Ecuatoriales*: donde el plano de referencia es el ecuador celeste y se denominan como Ascensión recta y Declinación.
- *Coordenadas Eclípticas*: donde el plano de referencia es la eclíptica o curva por donde orbita la Tierra alrededor del Sol y sus coordenadas se denominan Longitud y Latitud.

El ángulo origen de las coordenadas Ascensión recta y la Longitud Eclíptica viene definido por la posición del Punto Aries en la esfera celeste en una determinada fecha. En el caso del TFRM, esa fecha se corresponde con el equinoccio del año 2000 (EQUINOX = 2000). Las otras dos coordenadas se pueden encontrar definidas con distintos parámetros o tipos de coordenadas. [7]

Durante la implementación del servicio, se han encontrado distintos entornos con los que trabajar. Cada uno de ellos, al estar ya implementados, usan un tipo de coordenada celeste concreta.

- Imágenes FITS y el servicio VO de IVOA, trabajan con coordenadas Ecuatoriales.
- Base de datos PostgresSQL y su complemento PgSphere, trabajan con coordenadas Eclípticas.

Ello nos lleva a que la aplicación usa ciertos algoritmos de conversión de coordenadas, para mantener la integridad de los datos.

- Alg. Conversión de coordenadas ecuatoriales a coordenadas eclípticas. [8]
- Alg. Conversión de coordenadas eclípticas a coordenadas ecuatoriales. [9]

Dichos algoritmos son libres y se pueden encontrar en Internet. Cabe decir que se ha tenido que adaptar el algoritmo al proyecto para obtener los resultados correctos. Esta parte de adaptación del algoritmo ha sido fruto de la comprensión y soporte que ha proporcionado el tutor del TFRM, Dr. Octavi Fors (DAM, RACAB).

4.1.2. Imágenes FITS

Las imágenes FITS (Flexible Image Transfer System) son el tipo de formato en que se almacenan las imágenes tomadas por el telescopio.

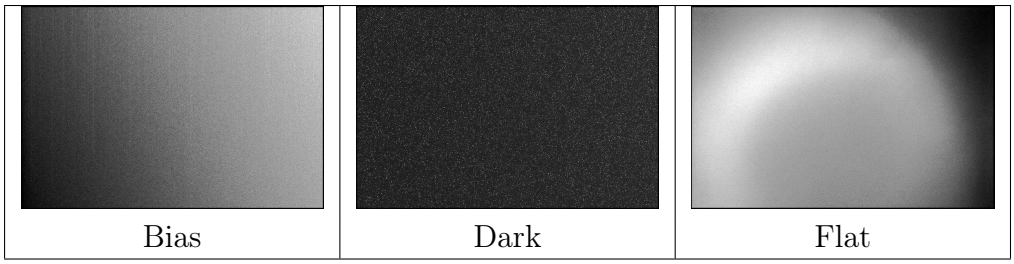
La peculiaridad de dicho formato es la flexibilidad de su cabecera que permite insertar datos de cualquier tipo. Estas han sido usadas por los astronomos para almacenar datos relacionados con la observación tomada con la imagen. Estos datos hacen referencia a coordenadas celestes, parámetros atmosféricos, datos del objeto observado y distintos parámetros astronómicos necesarios para la comprensión de la imagen.

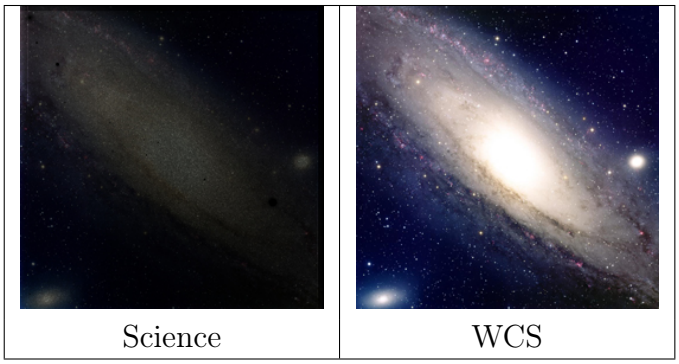
Existe una librería JAVA, anteriormente comentada, nom.tam.fits [12] que está especialmente diseñada para el tratamiento de ficheros FITS.

4.1.3. Tipos de imágenes

Durante el proceso de observación de un objeto celeste, se realizan distintos tipos de imágenes que posteriormente servirán para obtener un resultado válido para su estudio científico.

El proceso es, primero la obtención de imágenes CCD de calibración (Darks, Flats y Bias) para después poder realizar una correcta calibración de la imagen Science obteniendo una imagen de tipo WCS, que es con la que se realizan los estudios científicos.





A continuación se procede a describir cada uno de estos tipos de imágenes.

Bias imagen usada para calibrar el ruido fotónico que genera la circuiteria de la cámara. Son imágenes tomadas con el obturador cerrado y tiempo de exposición nulo.

Darks Este tipo de imagen es usado para eliminar el ruido producido por la generación espontánea de fotoelectrones debido a la temperatura de la cámara. Estas imágenes se toman con el obturador cerrado y con un tiempo de exposición igual al de la imagen Science.

Flats Ésta es usada para calibrar la eficiencia cuántica diferencial que existe entre los píxeles del chip CCD. Es tomada a la luz del día y durante un tiempo de exposición corto.

Science Imagen del objeto celeste deseado. Dicha imagen esta sin calibrar.

WCS Imagen Science calibrada y reducida astrométricamente. Dicha imagen son las usadas para el estudio Astronómico.

El archivo almacena todos los tipos de imágenes, es decir, que el Ingestor trata con todas las imágenes que captura el telescopio, ya sean de calibración como de observación. Mientras que el Comunicador SIA, únicamente trabaja con las imágenes ya calibradas, llamadas WCS.

A continuación se muestran imágenes obtenidas del mismo telescopio TFRM.

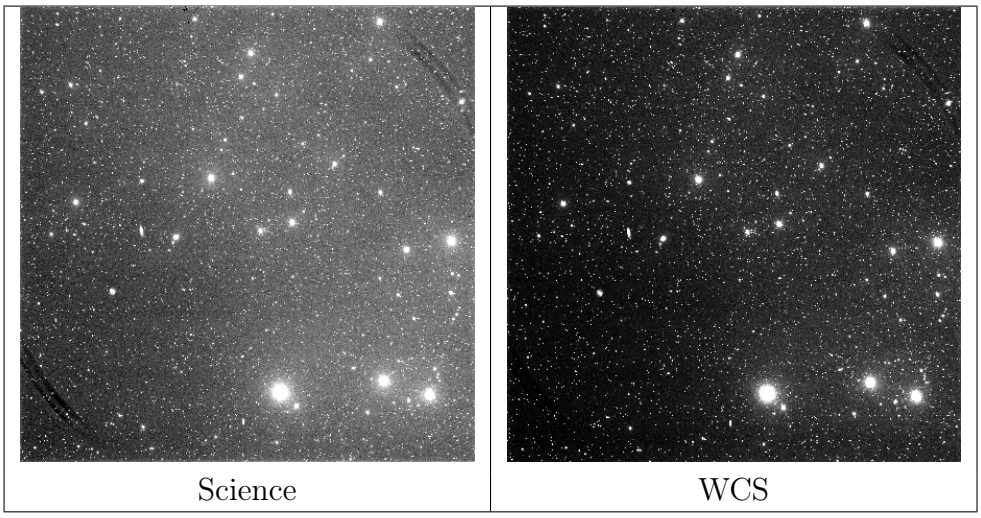


Figura 7: Campo de «blazars» obtenidos con el telescopio TFRM.

4.2. Ingestor

El Ingestor es la aplicación encargada de obtener toda la información necesaria de las imágenes tomadas con el telescopio e introducirlas en la base de datos.

A continuación se procede a describir más detalladamente los puntos de la aplicación.

4.2.1. Propiedades y configuraciones

Dispone de un fichero de propiedades con el que se permite configurar la aplicación según convenga, ya sea parámetros físicos como la ubicación de las imágenes, o parámetros que permiten adaptar cambios en las imágenes FITS.

- *DirIn, DirErrorFits, DirOut, DirLog, DirError y DirTest*: definen la ubicación física que deberán tener las imágenes antes y después del tratamiento y la ubicación de los diferentes ficheros de log y error.
- *dbName, dbUser y dbPassword*: efinen los parámetros necesarios para poder acceder a la base de datos.
- *params, dbTables, stringParameters, floatsParameters, intParameters, hexParameters y timeParameters*: contiene un listado con la relación entre los parámetros que contienen las imágenes FITS y la tabla-atributo en que se deben colocar dicha información. También, almacenan la información necesaria para saber el tipo de cada uno de los KEYWORDS de la imagen FITS, junto con las tablas contempladas dentro de dicha configurabilidad. Estos parámetros son muy útiles para adaptar el Ingestor a modificaciones de los KEYWORDS de las imágenes FITS, evitando así, tener que modificar el código en caso de futuros cambios. Es muy importante mantener correctamente actualizados todos los parámetros para un correcto funcionamiento.
- *testToDo*: lista que contiene el nombre de los test que se van a realizar en el proceso de integridad de las imágenes.
- *useCommandPolygon, useCommandScale, commandPolygon*: contienen información al respecto del modo en que se obtienen los valores SCALE y parámetros necesarios para la creación del polígono que forma el área de la imagen.

Con algunos de los parámetros del fichero de propiedades se puede realizar modificaciones en el funcionamiento de la aplicación.

Hay 4 puntos a destacar:

- El caso en que el administrador del archivo TFRM desee añadir un nuevo parámetro de las imágenes a la base de datos, únicamente deberá seguir los siguientes pasos:
  1. Crear el parámetro como nuevo atributo en la tabla de la base de datos.
  2. Añadir el nombre del parámetro, el atributo al que está relacionado en la base de datos (tabla/atributo) y el tipo de parámetro en el fichero de propiedades del Ingestor.

Con ello, la aplicación ya interpreta que debe insertar un nuevo parámetro de cada una de las imágenes en el archivo TFRM.

- El caso en que el administrador del archivo TFRM considere que el nombre de uno de los KEYWORDS de la cabecera FITS debe ser modificado. Éste simplemente deberá modificar el nombre que hace referencia al KEYWORD modificado en el fichero de propiedades del Ingestor, para que la propia aplicación capture el parámetro correcto de la cabecera FITS.
- Para configurar los distintos testeos que se realizarán a las imágenes.
- Para la obtención de valores necesarios de las imágenes FITS a través de un comando específico del sistema o a través de la misma cabecera FITS. Estos son el parámetro SCALE de la imagen FITS, necesaria para el Comunicador SIA, y las esquinas del polígono que formara el área de la imagen.

Como se puede observar, gracias a dicho fichero de propiedades, se consigue alcanzar uno de los objetivos mencionados con anterioridad, el de configurabilidad y adaptación a posibles cambios.

4.2.2. Funcionamiento interno

La ingestión de las imágenes conlleva múltiples subtareas necesarias para que el proceso obtenga el resultado esperado.

- Revisa que los distintos parámetros del fichero properties sean correctos. Lo que implica: controlar que disponga de permisos de lectura/escritura sobre las imágenes y ubicaciones físicas en las que se encuentran las imágenes y ficheros Log.
- Comprobar que dispone de un correcto acceso a la base de datos.
- Verificar que los valores insertados en la propiedad "params"(comentado anteriormente) sean correctos.
- Y revisar que las direcciones físicas insertadas en la tabla "path", sean correctos y con sus correspondientes permisos de lectura/escritura.

Una vez realizado el chequeo inicial, la aplicación procede a capturar todas las imágenes ubicadas en "DirIn"(, del fichero de propiedades).

A continuación se describe los pasos que se realizan con cada imagen capturada.

1. Testeo de integridad de la imagen. Se procede a realizar los distintos testeos implementados para revisar que las imágenes sean correctas para ser incluidas en el archivo.
2. Obtención de la cabecera FITS a través de las funciones necesarias.
3. Captura de los distintos identificadores estáticos que están relacionados con la imagen, como el identificador del objeto observado, el tipo de imagen que se está procesando y el filtro, instrumento y telescopio usado.
4. En caso de imagen WCS, captura de los valores de la cabecera FITS necesarios.
5. Inserción en la base de datos de un nuevo registro con toda la información obtenida, introduciendo así la imagen en el archivo.
6. Mover la imagen a su ubicación de salida correspondiente.
7. Escribir en el fichero de Log, que la imagen ha sido ingerida correctamente.

Si ha sucedido algún problema o se ha producido algún error, la aplicación registra el mismo en la base de datos y en el fichero de error, moviendo la imagen a la ubicación física especificada.

4.2.3. Sistema de Testeos de las imágenes

El sistema de Testeo, realiza el proceso de control de integridad de las imágenes para evitar que el archivo contenga imágenes erróneas o inválidas.

Dicho sistema realiza los siguientes testeos:

testUbicacion Controla que la imagen a ingestar sea capturada la posición correcta.

testShutter Controla que el obturador (KEYWORD SHUTTER) de la cámara estuviese en la posición correcta en el momento de la obtención de la imagen.

testExptime Revisa que el tiempo de exposición de la cámara sea el correcto.

testCamera Revisa que la temperatura de la cámara sea la correcta, es decir que la temperatura esté comprendida entre unos valores previamente definidos.

testMedian Controla que la intensidad o luminosidad capturada por cada uno de los píxeles sea la correcta.

testLblind Controla que la tapa de la cámara se encuentre en correcta posición durante el momento de la obtención de la imagen.

testBitpix Revisa que los bits por pixeles sean los correctos.

testTime Revisa que el tiempo inicial de exposición de la cámara sea concordante con el resto de parámetros relacionados.

testParameters Revisa que cada tipo de imagen contenga los parámetros adecuados en sus cabeceras correspondientes.

A través del fichero de propiedades, se puede habilitar o deshabilitar cada uno de los testeos mencionados.

4.2.4. Sistema de Error

Existe la posibilidad de que durante la ejecución del Ingestor se produzcan errores de varios tipos.

Hay dos grandes bloques:

- los errores que bloquean la aplicación al completo.
- los errores que bloquean la ingestión de una imagen en particular.

También, se puede dividir los errores en distintos bloques, según el momento en que se produce.

Los errores de Configuración bloquean la aplicación, mientras que los errores de Testeo y de Ingestión simplemente bloquean la imagen que se estaba procesando.

A continuación se procede a describir más en detalle los tipos de errores:

- **Error de Configuración:** Estos son aquéllos que se producen al inicio de la aplicación, durante el chequeo inicial del fichero de propiedades.
- **Error de Testeo:** Cada testeo está relacionado con un error. En el caso de fallo de testeo, se produce el error al que se encuentra vinculado. A diferencia del resto de errores, estos no terminan la ejecución hasta que se hayan realizado todos los testeos posibles. De este modo, se conocen todos los motivos del porqué la imagen no ha sido ingestada correctamente.
- **Error de Ingestión:** Este tipo de errores hacen referencia a cada una de las imágenes ingeridas. Existen muchos errores que entrarían dentro de este bloque, como fallo al insertar en la base de datos, fallo del tipo o valores de la cabecera FITS, etc.

La siguiente tabla muestra todos los errores que se pueden producir durante la ejecución.

Codigo Error	Descripción
ER_PSQL_000	Specific error of class PSQL.
ER_SQL_000	Specific error of class SQL.
ER_SQL_001	Don't find path_id in DB.(path table)
ER_SQL_002	Don't find typ_id in DB.(imtype table)
ER_SQL_003	Don't find ins_id in DB.(instrument table)
ER_SQL_004	Don't find tel_id in DB.(telescope table)
ER_SQL_005	Don't find fil_id in DB.(filter table)
ER_PROP_000	Specific error of properties file.
ER_PROP_001	Incorrect DirIn path.
ER_PROP_002	No privileges in DirIn.
ER_PROP_003	Incorrect DirOut path.
ER_PROP_004	No privileges in DirOut.
ER_PROP_005	Incorrect fileName when try move it.
ER_PROP_006	Error on moving file.
ER_PROP_007	Unknown type of parameter. Not specified in properties.
ER_FITS_000	Specific error of class nom.tam.fits.
ER_FITS_001	Don't find param in header FITS.
ER_FITS_002	Problems on insert wcs parameters.

Codigo Error	Descripción
ER_TEST_001	The file not is in correct location.
ER_TEST_002	Bad position of SHUTTER.
ER_TEST_003	Bad value of EXPTIME.
ER_TEST_004	CCD temperature too high.
ER_TEST_005	MEDIAN value exceeds reasonable value (25.000) for BIAS and DARK frames.
ER_TEST_006	Bad value for LBLIND.
ER_TEST_007	Incorrect BITPIX, is diference of 16.
ER_TEST_008	Incorrect TIMESTART relation.
ER_TEST_009	Miss any of compulsory parameter in header FITS.

Estos errores son almacenados en varios sitios, para poder tener un control de todos aquellos fallos producidos durante la ejecución del Ingestor.

Primeramente se almacena en un fichero local, llamado voError.log, donde se guarda el detalle del error producido, es decir, instante en que se ha producido, clase que lo ha provocado y descripción del error. Este fichero es muy útil para ver más detalladamente el error que se ha producido. Por otra parte, cada error es almacenado en la base de datos, relacionado con un código de error (como hemos visto en la anterior tabla). Dicha tabla también almacena información importante y descriptiva sobre el error producido.

Las imágenes que han sido bloqueadas por el hecho de haberse producido alguno de los errores comentados, serán trasladadas a la ubicación física adecuada, especificada en el fichero properties.

4.2.5. Sistema de Log

Este sistema almacena todos los movimientos que realizan las imágenes. Hay dos tipos de registros, los que hacen referencia a imagen correctamente procesada y los que hacen referencia a las imágenes que han provocado un error y han sido movidas a una nueva ubicación.

Estos ficheros de Log son creados cada vez que es ejecutado el Ingestor y es llamado voLog.log.

4.3. Comunicador SIA

A continuación, se describe más detalladamente todo lo que hace referencia al servicio SIA que se comunica con VO, a través del web-service implementado en el TFRM.

4.3.1. Funcionamiento interno

Para que VO pueda acceder al archivo TFRM, se ha implementado un servicio llamado SIA (Simple Image Acces), ya estandarizado según IVOA.

Dicho servicio esta formado por dos aplicaciones PHP, cuyas funciones son:

1. La encargada de recibir las peticiones y volcar la información necesaria.
2. La encargada de enviar la imagen seleccionada al cliente.

La primera recibe peticiones definidas por el protocolo SIAP de VO [4]. Estas peticiones son recibidas vía HTTP desde el mismo VO. La petición del cliente contiene la información necesaria de la imagen que desea obtener y vienen definidos por el SIAP. Son los siguientes:

POS posición central de la imagen que se desea obtener. Está formado por Ascensión Recta y Declinación.

SIZE tamaño de la imagen.

FORMAT formato en que se desea obtener la imagen. Suele ser en formato FITS.

Otros Existen otros parámetros especificados por el protocolo que no son obligatorios y que en TFRM no se encuentran habilitados.

Ejemplo de una petición enviada por VO:

```
http://localhost/tfrmVO/tfrmVO.php?pos=220,0.6&size=0.6
```

La aplicación, una vez recibida la petición, analiza los datos adjuntos con la misma para poder construir una *query* (consulta) que será enviada a la base de datos; esta

query se encuentra descrita más adelante. Con esta *query*, la base de datos devuelve los registros de las imágenes que cumplen con los requisitos establecidos por el cliente.

Estos registros, serán devueltos al VO en formato XML con una estructura previamente definida en el protocolo SIAP. Las imágenes obtenidas con la *query*, son devueltas junto a una serie de parámetros; algunos obligatorios y otros optativos. Todo estipulado por el mismo protocolo.

XML de respuesta a la petición VO:

```
<VOTABLE version="1.0">
 <RESOURCE type="results">
  <DESCRIPTION>TFRM Image Query Service</DESCRIPTION>
  <INFO name="QUERY_STATUS" value="OK"> Successful Search</INFO>
  <TABLE>
   <FIELD name="OBS_ID" ucd="OBS_ID" datatype="char" arraysize="*"/>
   <FIELD name="Reference" ucd="VOX:Image_AccessReference" datatype="char" arraysize="*"/>
   <FIELD name="Product_Name" ucd="VOX:Image_Title" datatype="char" arraysize="*"/>
   <FIELD name="Instrument" ucd="VOX:INST_ID" datatype="char" arraysize="*"/>
   <FIELD name="Julian_Date" ucd="VOX:Image_MJDateObs" datatype="char" arraysize="*"/>
   <FIELD name="RA" ucd="VOX:POS_EQ_RA_MAIN" datatype="double" arraysize="*"/>
   <FIELD name="DEC" ucd="VOX:POS_EQ_DEC_MAIN" datatype="double" arraysize="*"/>
   <FIELD name="NAXES" ucd="VOX:Image_Naxes" datatype="int" arraysize="*"/>
   <FIELD name="NAXIS" ucd="VOX:Image_Naxis" datatype="int" arraysize="*"/>
   <FIELD name="SCALE" ucd="VOX:Image_Scale" datatype="double" arraysize="*"/>
   <FIELD name="CRFRAME" ucd="VOX:STC_CoordRefFrame" datatype="char" arraysize="*"/>
   <FIELD name="EQUINOX" ucd="VOX:STC_CoordEquinox" datatype="double" arraysize="*"/>
   <FIELD name="FORMAT" ucd="VOX:Image_Format" datatype="char" arraysize="*"/>
   <DATA>
    <TABLEDATA>
    <TR>
     <TD>1281</TD>
     <TD>http://localhost/tfrmVO/tfrmVOdown.php?downIDPROD=1281</TD>
     <TD>2011-04-09T02_29_52_wcs.fits</TD>
     <TD/>
     <TD>55660.1</TD>
     <TD>221.92372</TD>
     <TD>0.6466214</TD>
     <TD>2</TD>
     <TD>4096 4096</TD>
     <TD>3.88990823145</TD>
     <TD>ICRS</TD>
     <TD>2000</TD>
     <TD>image/fits</TD>
    </TR>
    <TR>
     <TD>1301</TD>
     <TD>http://localhost/tfrmVO/tfrmVOdown.php?downIDPROD=1301</TD>
     <TD>2011-04-09T02_29_52_wcs.fits</TD>
     <TD/>
     <TD>55660.1</TD>
     <TD>221.92372</TD>
```

```
      <TD>0.6466214</TD>
      <TD>2</TD>
      <TD>4096 4096</TD>
      <TD>3.88990823145</TD>
      <TD>ICRS</TD>
      <TD>2000</TD>
      <TD>image/fits</TD>
     </TR>
     <TR>
      <TD>1282</TD>
      <TD>http://localhost/tfrmVO/tfrmVOdown.php?downIDPROD=1282</TD>
      <TD>2011-04-09T01_59_12_wcs.fits</TD>
      <TD/>
      <TD>55660.082</TD>
      <TD>221.91469</TD>
      <TD>0.64544207</TD>
      <TD>2</TD>
      <TD>4096 4096</TD>
      <TD>3.87903608734</TD>
      <TD>ICRS</TD>
      <TD>2000</TD>
      <TD>image/fits</TD>
     </TR>
     </TABLEDATA>
    </DATA>
   </TABLE>
  </RESOURCE>
</VOTABLE>
```

La aplicación contempla otro parámetro de entrada que va relacionado con el servicio que da el TFRM, el ya comentado parámetro “FORMAT”.

Ejemplo de petición VO:

http://localhost/tfrmVO/tfrmVO.php?format=METADATA

El parámetro de entrada es llamado "METADATA". Con este obtendremos toda la información que concierne a parámetros de entrada y salida del servicio SIA proporcionado por TFRM. Principalmente es usado para conocer qué tipo de parámetros son necesarios durante la consulta inicial y cuáles son los parámetros que serán devueltos.

XML de respuesta a la petición de METADATA de VO:

```
<VOTABLE version="1.0">
 <RESOURCE type="results">
  <DESCRIPTION>TFRM Metadata Query Service</DESCRIPTION>
  <INFO name="QUERY_STATUS" value="OK"> Successful Search</INFO>
  <PARAM name="INPUT:POS" value="0,0" datatype="double">
   <DESCRIPTION>
    Search Position in the form 'ra,dec' where ra and dec are given in decimal
    degrees in the ICRS coordinate system.
   </DESCRIPTION>
  </PARAM>
  <PARAM name="INPUT:SIZE" value="0.05" datatype="double">
   <DESCRIPTION>Size of search region in the RA and DEC directions</DESCRIPTION>
  </PARAM>
  <PARAM name="INPUT:INTERSECT" value="OVERLAPS" datatype="char" arraysize="*">
   <DESCRIPTION>Choice of overlap with requested region</DESCRIPTION>
   <VALUES>
    <OPTION>OVERLAPS</OPTION>
    <OPTION>COVERS</OPTION>
   </VALUES>
  </PARAM>
  <PARAM name="INPUT:EQUINOX" value="J2000" datatype="char" arraysize="*">
   <DESCRIPTION>Static value is always considered J2000.</DESCRIPTION>
  </PARAM>
  <PARAM name="INPUT:CRFRAME" value="ICRS" datatype="char" arraysize="*">
   <DESCRIPTION>Static value is always considered ICRS.</DESCRIPTION>
  </PARAM>
  <TABLE>
   <DATA><TABLEDATA/></DATA>
  </TABLE>
 </RESOURCE>
</VOTABLE>
```

La segunda aplicación PHP realiza la tarea de atender la petición de descargar una imagen. Donde el parámetro de entrada es simplemente el identificador de la imagen que se desea descargar.

Con dicha información, la aplicación obtiene la ubicación física en la que se encuentra la imagen y procede a descargarla.

Ejemplo de petición de descarga:

```
http://localhost/tfrmVO/tfrmVOdown.php?downIDPROD=1282
```

Hay que añadir que dicha petición se encuentra descrita en el XML de respuesta a la petición de imágenes de VO.

A continuación se muestra una ejemplo de imagen obtenia a través de la petición de descarga.

Figura 8: Campo exoplaneta WASP-37b, obtenido con el telescopio TFRM.

4.3.2. Consulta de imágenes

Antes de construir la *query* que se enviará a la base de datos, se deben controlar cuáles son los parámetros de entrada por varios motivos.

- Para evitar que dichos parámetros de entrada puedan provocar daños irreversibles en la base de datos, o los llamados "Injection Attacks".
- Para saber qué tipo de consulta se debe hacer a la base de datos.

Para evitar «Injection Attacks» o inyección de consultas dañinas, únicamente se permite la entrada de parámetros de tipo numérico y los caracteres punto y coma, evitando así cualquier tipo de ataque [10].

Por otro lado, SIAP, especifica que si el parámetro SIZE es igual a 0, la respuesta del SIA deberá ser todas aquellas imágenes que contengan la posición POS. Esto implica que dependiendo del valor de SIZE, la consulta hará referencia a buscar una posición celeste o buscar el polígono formado por POS y SIZE.

El polígono lo obtenemos a través de una serie de formulas de trigonometría esférica, que permiten encontrar las esquinas en coordenadas celestes a partir del

punto central y el tamaño deseado. Estas formulas han sido proporcionadas por el tutor del TFRM, Dr. Octavi Fors (DAM, RACAB).

$$\begin{aligned} RA_{min} &= RA - ((SIZE/2) * \cos(DEC)) \\ RA_{max} &= RA + ((SIZE/2) * \cos(DEC)) \\ DEC_{min} &= DEC - (SIZE/2) \\ DEC_{max} &= DEC + (SIZE/2) \end{aligned} \tag{1}$$

Con estos 4 valores, obtenemos las esquinas del polígono que representa la región de interés del cliente.

Una vez obtenido el polígono, simplemente falta transformar las coordenadas al tipo adecuado [8] para poder trabajar con el complemento PgSphere de PostgresSQL, el cual, a partir del polígono y una base de datos correctamente formada, realiza una búsqueda de los registros que contienen la zona de interés o polígono que se ha construido.

4.3.3. Comunicación TFRM - VO

La siguiente imagen muestra las distintas interacciones que se producen durante una consulta de VO al archivo TFRM. (figura 9)

Como se puede observar, se sigue la siguiente traza:

1. TFRM recibe petición de información de VO.
2. TFRM responde a VO mediante un XML.
3. En caso que se seleccione algunas de las imágenes retornadas, VO envia una petición de descarga a TFRM.
4. TFRM descarga a VO la imagen seleccionada.

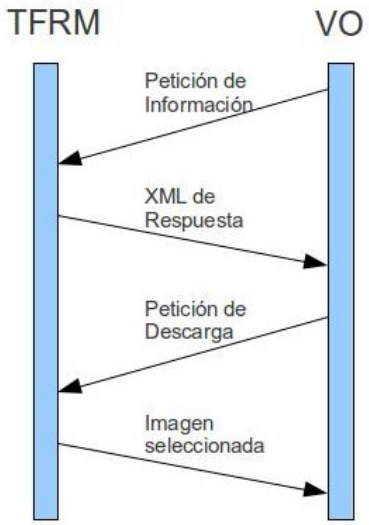


Figura 9: Traza de comunicación entre TFRM y VO.

4.3.4. Privacidad

Todo telescopio dispone de la discrecionalidad de proporcionar las imágenes tomadas según convenga.

En el caso que concierne a este proyecto, el servicio SIA únicamente proporcionará imágenes al VO una vez transcurridos un número de meses después de su observación aún por determinar. Esto es debido a la diversidad de diferentes programas observacionales que el TFRM llevará a cabo.

5. Instalación y Pruebas en el TFRM

La Instalación y Pruebas en el TFRM, es el unico objetivo especificado inicialmente, que no se han podidio realizar. Donde los motivos han sido la falta tiempo y disponibilidad de todas las partes implicadas en el proyecto.

Esta parte sera realizada una vez finalizado el proyecto y con carácter privado.

5.1. Instalación

Requisitos básicos para la instalación:

- Maquina capacitada para procesar centenares de peticiones diarias vía Internet. Disponer de una conexión de alta velocidad (está planificado un enlace de 100Mbps simétricos con l'Anella Científica para el otoño de 2011).
- Deberá disponer de un sistema de almacenamiento masivo de imágenes FITS. Cada imagen puede ocupar 32M.
- Sistema operativo Linux.

Pasos a seguir para la instalación del archivo en las instalaciones del TFRM.:

1. Instalar Apache, PHP
2. Instalar PostgreSQL
   a) Instalar complemento PgSphere
3. Instalar JAVA
4. Configurar Base de datos y permitir el acceso desde aplicaciones JAVA.
5. Instalar aplicación Ingestor
   a) Configurar *crontab*, para que este se ejecute cada día a una cierta hora.
6. Instalar aplicación SIA.
7. Configurar apache para permitir la recepción de peticiones a través de Internet.
8. Inscribirse en el registro de IVOA, como nueva entidad que ofrece un archivo VO.

5.2. Pruebas

Existen dos tipos de pruebas usadas para verificar el correcto funcionamiento del archivo.

- *Pruebas en entorno de desarrollo*, han sido realizadas con imágenes reales tomadas por el telescopio y grabadas en la maquina de forma manual, simulando el entorno de trabajo del servidor TFRM.
- *Pruebas en entorno real*, se realizan ya en el servidor TFRM con el archivo instalado correctamente.

Las únicas pruebas que se han podido realizar son las de entorno de desarrollo, obteniendo un buen rendimiento y efectividad en la ejecución del servicio y un correcto funcionamiento de todas las partes que implican al proyecto, tales como el Ingestor, el Comunicador SIA o la base de datos.

Las pruebas realizadas en el servidor TFRM son más útiles, ya que son las que pueden provocar los errores más reales y críticos, que a priori, son los errores/Bugs que deberían ser solucionados con máxima prioridad.

De igual modo que en el proceso de instalación, queda pendiente un plan de pruebas exhaustivo sobre todo el sistema, tanto en entorno de desarrollo como en el real, para poder certificar su correcto funcionamiento.

6. Conclusiones

El objetivo principal del proyecto, consistente en la implementación de un servicio VO, se ha llevado a cabo correctamente. Desde la investigación de la tecnología y diseño, hasta su implementación en un entorno de desarrollo. Únicamente ha quedado pendiente la parte de instalación, pruebas en entorno real, y puesta en marcha en la comunidad VO.

Mediante este proyecto y en colaboración del Spanish Virtual Observatory (SVO), el TFRM se convierte en el primer observatorio robótico Español en conseguir implementar un archivo VO y su pertinente servicio desde las bases de un Proyecto Final de Carrera, consiguiendo un valioso ingrediente de aprendizaje y formación del estudiante ponente, haciendo que el "know-how"del funcionamiento del archivo VO se mantenga en el seno del grupo TFRM.

También, gracias a eso, se ha obtenido un servicio totalmente a medida, autónomo y configurable, permitiendo tener un sistema que no requiere de personal cualificado supervisando las distintas tareas a realizar.

Principales méritos del proyecto:

- **Innovador**: Es una de las primeras veces que se implanta este tipo de servicio dentro de la comunidad de telescopios robóticos españoles, con lo que la realización de este proyecto ha implicado dar un pequeño paso más, en los avances tecnológicos en el mundo de la Astronomía. Este hecho, ha supuesto una dedicación y esfuerzo extra, que sin el apoyo de técnicos y especialistas en el campo del VO y de la Astronomía no se hubiera podido obtener una solución final óptima y eficiente.
- **Multidisciplinar**: El especial esfuerzo que ha implicado la unión entre las distintas tecnologías informáticas usadas y los múltiples conceptos científicos que el entorno del proyecto suponían. Dos ejemplos claros, seria la parte de algoritmos de conversión de coordenadas y la parte de protocolo SIAP, los cuales parten de la base de que el implementador ya es conocedor de ciertos aspectos teóricos astronómicos.
- **Modular**: La estructura diseñada de la aplicación Ingestor es clara y bien estructurada en capas que ha facilitando su posterior desarrollo.

En el ámbito más personal, las conclusiones que saco de este proyecto son muy positivas, ya que me ha aportado muchísimos conocimientos nuevos, que serán de gran ayuda de ahora en adelante.

Estos conocimientos hacen referencia a los distintos lenguajes de programación y librerías que se han usado para la implementación y también conocimientos teóricos relacionados con la Astronomía, que como ya he comentado, son para mí un extra muy positivo.

6.1. Posibles mejoras

El hecho de ser un servicio estandarizado por una organización y que ya ha sido implementado en otro centros astronómicos, limita las mejoras a hacer, ya que es evidente que IVOA dispone de mucha más experiencia para poder decir al respecto de las mejoras del servicio.

Posibles mejoras que pueden hacer que el servicio en el TFRM sea mas seguro, efectivo y automático.

- El uso de conexiones https y certificados digitales durante el intercambio de información entre VO y el TFRM, consiguiendo una mayor seguridad en las peticiones e información que se transmite.
- Implementar un sistema de triggering de envío de mail en caso de producirse un error. Esto facilitaría mucho la atención real que requiere el mismo sistema, evitando así, que el administrador del archivo tenga que estar revisando la base de datos a diario en busca de posibles errores.
- Independizar el Ingestor de aplicaciones externas. Durante el proceso de ingestión, se usan una serie de aplicaciones externas para obtener información de la imagen. La idea seria implementar el código necesario para evitar dichas llamadas a programas externos. (referente al parámetro «commandPolygon» del fichero de propiedades).
- Y quizás el más importante, implementar el comunicador SIA con la tecnología JSP, para así, formar parte del mismo Ingestor. Consiguiendo todo el servicio en una misma aplicación JAVA.

7. Agradecimientos

Este proyecto no se hubiera llevado a cabo sin la ayuda de las siguientes personas, las cuales me han estado aconsejando y guiando durante el transcurso de implementación.

- Equipo TFRM (DAM, RACAB).
  - Dr. Octavi Fors
- Equipo Spanish Virtual Observatory (CAB-LAEFF) [11].
  - Investigador principal Dr.Enrique Solano, y muy especialmente a Sr.Raúl Gutiérrez.

Agradecer también a amigos, familiares y concretamente a mi pareja por haberme soportado tanto en los buenos momentos como en los malos.

Referencias

[1] Pagina Oficial TFRM, http://www.am.ub.es/bnc/

[2] Pagina Oficial IVOA, http://www.ivoa.net/

[3] Web de VO, http://en.wikipedia.org/wiki/Virtual_Observatory

[4] Web de SIA, http://www.ivoa.net/Documents/SIA/

[5] Web Oficial Aladin Sky Atlas, http://aladin.u-strasbg.fr/

[6] Web de PgSphere, http://pgsphere.projects.postgresql.org/

[7] Web de coordenadas celestes, http://es.wikipedia.org/wiki/Coordenadas_celestes

[8] Web de algoritmo de conversion, http://es.wikipedia.org/wiki/Conversi\%C3\%B3n_de_coordenadas_ecuatoriales_a_coordenadas_ecl\%C3\%ADpticas

[9] Web de algoritmo de conversion, http://es.wikipedia.org/wiki/Conversi\%C3\%B3n_de_coordenadas_ecl\%C3\%ADpticas_a_coordenadas_ecuatoriales

[10] Web de Injection Attacks, http://en.wikipedia.org/wiki/SQL_injection

[11] Web Oficial Spanish VO, http://svo.cab.inta-csic.es/main/index.php

[12] Documentación específica y descriptiva relacionada con la libreria nom.tam.fits, http://www.google.es/url?sa=t&source=web&cd=1&ved=0CBwQFjAA&url=http%3A%2F%2Fheasarc.gsfc.nasa.gov%2Fdocs%2Fheasarc%2Ffits%2Fjava%2Fv0.9%2FJavaFits.doc&rct=j&q=nom.tam.fits%20tutorial&ei=7-9oTvuoFIrn-gbWyOjGCw&usg=AFQjCNGwZxEL5z6Eg3KQ2ZTNKPWDoM-emg&cad=rja

(Resum, resumen, summary)

S'ha implementat un servei VO (Virtual Observatori) a les instal·lacions del Telescopi TFRM, que permet distribuir les imatges preses amb el telescopi de manera remota i automàtica a qualsevol usuari del servei.

El servei està format per un arxiu d'imatges, una aplicació que integra les imatges al arxiu y una aplicació que es comunica amb els clients d'VO, rebent peticions i responen segons s'especifica al protocol SIAP (Simple Image Access Protocol).

Se ha implementado un servicio de Observatorio Virtual (VO) en las instalaciones del telescopio TFRM, que permite distribuir las imágenes tomadas por el telescopio de una forma remota y automática a cualquier usuario del servicio.

El servicio esta formado por un archivo de imágenes, una aplicación que integra las imágenes en el archivo y una aplicación que se comunica con los clientes de IVOA, recibiendo peticiones y respondiendo según se especifica en el protocolo SIAP (Simple Image Access Protocol).

We have implemented a Virtual Observatory (VO) service at Telescope Facility in TFRM, which allows distributing the images taken by the telescope in a remote and automatic way to any service user.

The service consists of an image archive, an application that integrate images on file and an application that communicates with VO clients, receiving and answering requests as specified in SIAP protocol (Simple Image Access Protocol).