WAS: the data archive for the WEAVE spectrograph

Guerra, Jose, Molinari, Emilio, Lodi, Marcello, Martin, Adrian, Dalton, Gavin, et al.

Jose Guerra Sr., Emilio Molinari, Marcello Lodi, Adrian Martin, Gavin B. Dalton, Scott C. Trager, Shoko Jin, Don Carlos Abrams, Piercarlo Bonifacio, Jose Alfonso López Aguerri, Antonella Vallenari, Esperanza E. Carrasco Licea, Kevin F. Middleton, "WAS: the data archive for the WEAVE spectrograph," Proc. SPIE 9913, Software and Cyberinfrastructure for Astronomy IV, 99131X (8 August 2016); doi: 10.1117/12.2231300

Event: SPIE Astronomical Telescopes + Instrumentation, 2016, Edinburgh, United Kingdom

WAS: The Data Archive for the WEAVE Spectrograph

Jose Guerra*, Emilio Molinari, Marcello Lodi, Adrian Martin, Telescopio Nazionale Galileo (Spain); Gavin B. Dalton, Univ. of Oxford (United Kingdom); Scott C. Trager, Univ. of Groningen (Netherlands); Shoko Jin, Leiden University (Netherlands); Don Carlos Abrams, Isaac Newton Group of Telescopes (Spain); Piercarlo Bonifacio, Observatoire de Paris à Meudon (France); Jose Alfonso López Aguerri, Instituto de Astrofísica de Canarias (Spain); Antonella Vallenari, INAF - Osservatorio Astronomico di Padova (Italy); Esperanza E. Carrasco Licea, Instituto Nacional de Astrofísica, Óptica y Electrónica (Mexico); Kevin F. Middleton, STFC Rutherford Appleton Lab. (United Kingdom)

FGG-INAF, Telescopio Nazionale Galileo - Rambla J.A. Fernández Pérez, 7 38712 Breña Baja, TF - Spain.

ABSTRACT

The WAS[1](WEAVE Archive System) is a software architecture for archiving and delivering the data releases for the WEAVE[7] instrument at WHT (William Herschel Telescope).

The WEAVE spectrograph will be mounted at the 4.2-m WHT telescope and will provide millions of spectra in a 5-year program, starting early 2018. The access and retrieval of information will be through its dedicated archive, the WEAVE Archive System (WAS). This will be developed and maintained at the TNG[2] premises on the same island as the WHT. Its structure foresees the main axes of scalability, virtualization, and high availability. We present here the first performances on a simulated data set of 20M spectra, using different architectures and hardware choices.

Keywords: TNG, Telescopio Nazionale Galileo, WEAVE, WAS.

1. INTRODUCTION

WEAVE is a new wide-field spectroscopic facility that requires a new high-performance database structure in order to respond to the increasing number of science use cases in astronomy. WAS is meant to be able to cope with the new challenges to enable rapid access with more data to search for than any other database for the same category. WAS sits atop a new structure that will provide a distributed and replicated access with a very high capacity to adapt to the increasing amount of stored data during the project's life-span, giving the kind of flexibility required for the project.

The main database of WAS is based on Apache Cassandra[3] and Apache Solr.[4] Apache Cassandra is based on Amazon's Dynamo[5] and Google's Bigtable[6] and it is currently one of the most successful NoSQL databases being used in clusters of 1,000+ nodes in many corporations all over the world. Cassandra provides a simple data-model instead with dynamic control over the data layout and its format. Another important feature of this database is that it was designed to be able to run on cheap hardware with a high write throughput without sacrificing the read cycles.

Cassandra's data-model is basically a multi-dimensional *map* structure indexed by a *key*. The *value* is a highly structured object. Every operation on a row *key* is atomic per replica no matter how many columns are being read or written. Columns are grouped into what is called *Column Families* (tables) similar to Google's Bigtable structure.

Another Cassandra feature is the ability to scale up linearly. This feature needs a sophisticated partition system able to dynamically adapt to a new cluster structure. Cassandra partitions use a *hashing* algorithm to keep consistent data across a fixed ring space across the cluster.

*jose.guerra@tng.iac.es; phone +34 922 433-651; fax +34 922 420-508; www.tng.iac.es

Software and Cyberinfrastructure for Astronomy IV, edited by Gianluca Chiozzi, Juan C. Guzman, Proc. of SPIE Vol. 9913, 99131X · © 2016 SPIE
CCC code: 0277-786X/16/$18 · doi: 10.1117/12.2231300

Replication is also present in the database to guarantee high availability and reliability. Data is replicated depending on a replication factor N defined on the **keyspace**[†]. Apart of the data for an existing node, the database also replicates the node data into other N-1 nodes in the ring. This architecture concept allows very straight forward replication of the the WAS database on either the Cloud or in a private cloud from another WEAVE associated member. Cassandra's world-wide replication capabilities are proven in many top companies across the globe.

Apache Solr is a high-performance search engine with advanced features such as indexing, faceting[‡] and real-time analytics. The engine is based on Apache Lucene sub-project and it is an enterprise grade scalable searching solution able to work across multiple datacenters or even on The Cloud.

The solution described for the WEAVE archive is one of the most scalable database solutions, allowing future upgrades for new areas such as batch analytics and graphs allowing the adoption of new novel methods of searching and data analysis. This support of big-data technologies is provided by DataStax Enterprise[§] (DSE) packages.

This article describes the storage and search basis defined for the WEAVE Archive (WAS) and its three level of pipelines; OCS (Observatory Control System) at La Palma, Spain, CPS (Core Processing System) at CASU in Cambridge, UK and APS (Advanced Processing System) at the IAC in La Laguna, Spain. The architecture is also open to CDP (Contributed Data Product) for the results of data reduction and analyses performed by WEAVE Science Teams, external to the pipelines of the OCS, CPS and APS..

2. SYSTEM ARCHITECTURE

The System Architecture chosen for the WEAVE archive is based on a well-known three-tier web architecture; front-end, middleware and the backend. However, the front-end was partially removed (proxies and firewall) from the test environment in order to simplify the architecture as is shown in Figure 1 below.

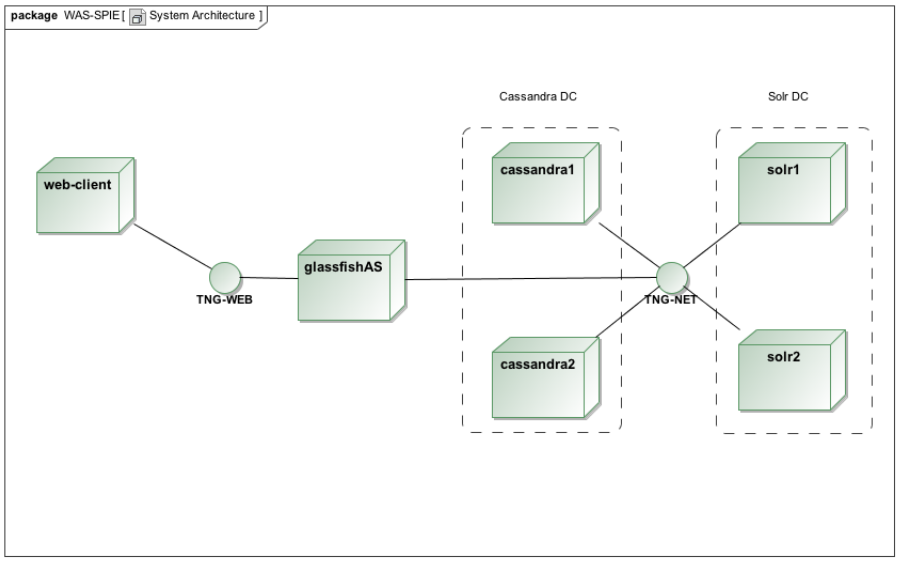

Figure 1. Test System Architecture environment

[†]keyspace is a Cassandra database structure to keep all the tables in the same space.
[‡]Arrange search results in columns with numerical count of key terms.
[§]Visit http://www.datastax.com for further details about the framework.

2.1 Middleware

The Business logic tier or the Middleware is formed by an Oracle Glassfish 4.1.1 Application Server which runs a Java EE container. This Applications server is running on a virtual machine with 2 CPUs, 3GB RAM and CentOS 7.1. The virtual environment is supported by OpenNebula.

2.2 Backend

The database architecture is based on a four nodes cluster that is split into two Datacenters; Cassandra Datacenter (*cassandra1* and *cassandra2*) and Solr Datacenter (*solr1* and *solr2*). This Datacenter architecture allows workload segregation in the DSE cluster. Each of these datacenter is composed by two virtual machines running with 12GB RAM, 4 CPU on CentOS 7.1. As the previous Section 2.1, the virtual environment is supported by OpenNebula.

3. SOFTWARE ARCHITECTURE

3.1 Web GUI

The WAS test Web GUI is based on HTML5 and JSF 2.2+ as the base-line framework. The Oracle Glassfish container behaves as the middleware layer, and it supports the implemented MVC pattern. Additionally, Twitter's Bootstrap 3.3 framework was included to add the extra functionality needed for the support of widgets, CSS styles and basic javascript controllers.

For the representation of the data, the GUI has three ways to visualize data to their users; using paginated tables, one-dimensional charts such as histograms and two-dimensional charts such as heatmaps and contours. In order to represent these views, the GUI is using standard third-party libraries such as Highcharts for the charts and Tables to display the summary of the query results as a paginated table.

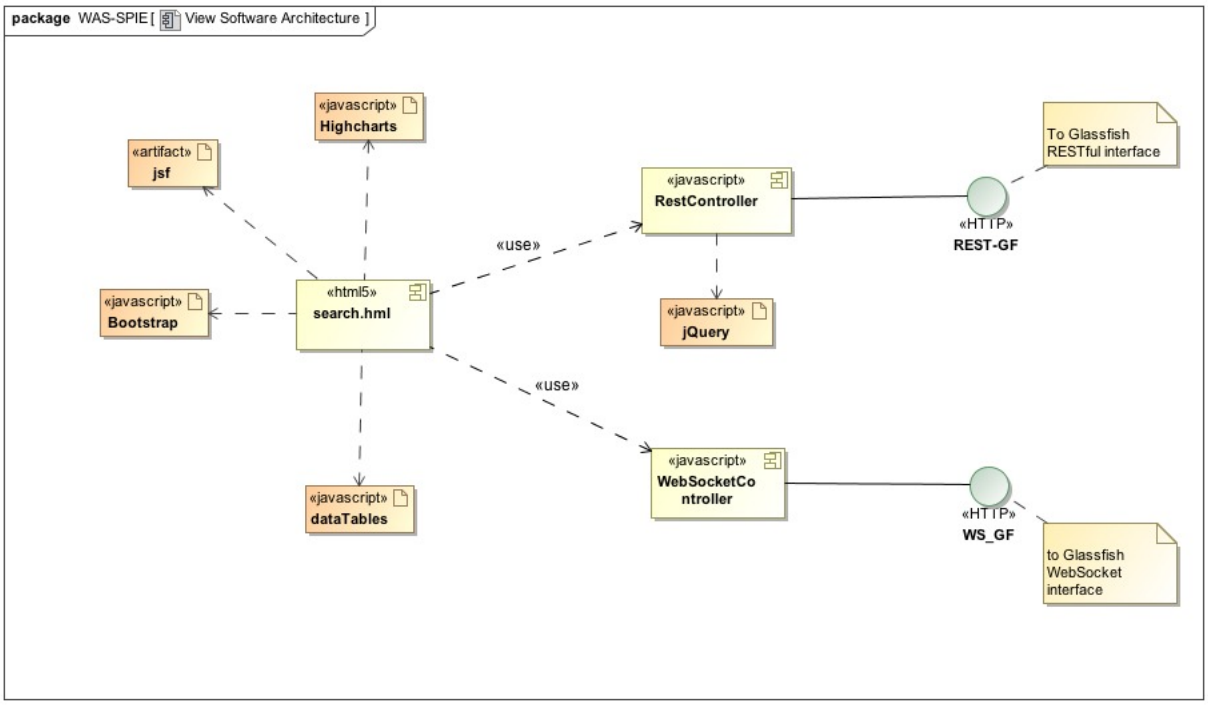

Figure 2. General design of the view

Communications wise, the GUI controller is a fundamental piece to communicate remotely with the business logic. Two mechanisms are used to communicate the view; RESTful WebServices providing a synchronous

channel and theWebSockets to enable a fully asynchronous communication channel for long processing queries. RESTful services are provided by jQuery while WebSockets are embedded within a standard Javascript library. The way to encode the messages among the view-controllers and the business logic controllers is based on the JSON format.

The proposed design is meant to make for an easier framework as a whole to be flexible enough for adding new requirements very quickly. An overview of this architecture is described in the next two sections and Figure 2 shows its Software Architecture.

3.1.1 Design of the View

The View is a web page and is the main entry-point for the most of the WAS use cases. This GUI drives all the iterations from the WEAVE community with the search engine. Its final design is still under discussion and it may change in future, however its current approach can be considered as a first attempt of design and it is shown in Figure 3.

The initial design of the main web page has been driven by the product of the pipelines from OCS, CPS and APS with a Query Pad to allow the users to query the database in a native language. The web page also allows one to navigate to different links from its main menu features such as Display, Download, Preferences and Help as well as an online help-desk web page. The next five sub-sections briefly describes some features of the main menu.

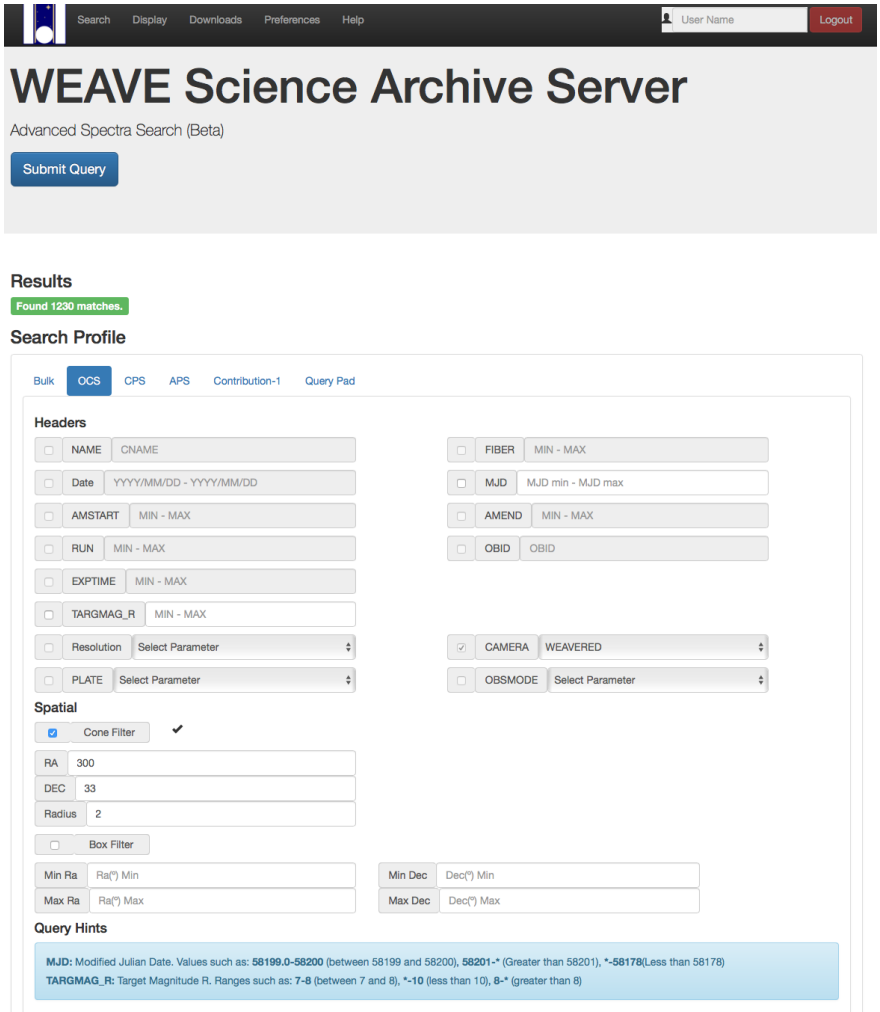

Figure 3. WAS main search page

Search This page is the main entry page to the web site. On the page, users can select on several tabs, different parameters available for each of the pipelines and also parameters relative to the catalog, telescope coordinates, airmass, run numbers etc. The search page also allows the users to look for areas on the sky in either cone or box regions. The page can also help the users out with hints on their queries with small banners at the very bottom of the page.

For this very first prototype, the required scope of these test scenarios was to enable the application to run only a few search features as a first proof of the technology concepts. The structure of the web page was to split it up in several tabs matching the different pipelines.

The OCS tab panel has information related to the raw files and also full support for spatial searches, cone search and box search. The CPS tab panel is meant to be the selection criteria panel for the CPS pipeline. Currently the GUI supports three flux criteria in three bands; TARGETFLUX_G, TARGFLUX_I and TARGFLUX_R. The APS tab supports the selection of parameters from the APS pipeline. Currently the GUI has support for filtering using object classification (CLASS) and sub-class (SUBCLASS).

Given the resources available from the Solr indexer cluster, the search tool is able to pin down complicated filter queries at blazing speed. Section 5 will talk about some search performance figures.

Display This menu is meant to be a page for display the results from the queries done from the Search page. The Display menu gives the user the possibility to display data in tables and several kind of plots such as histograms, heatmaps and sky plots using equatorial and galactic coordinate systems. Displaying targets in tables has also the possibility to select a particular target and visualize its spectra too.

Download This section will allow the users for downloading the result of the searches in several ways and formats. There are currently three use cases for downloading data from WAS; the first is the most straight forward and is downloading a selected target FITS file from the display spectra page described in previous section. The file is made on the fly to be downloaded. The second use case, likely the most common case, is downloading a list of targets. This method works as a job submitted to WAS. Once the user selects their list of targets, a submission mechanism will enable WAS to process the job sending back a response by email with instructions for downloading the list of files using HTTP protocol and Unix tools such as *wget* or *curl*. If the volume of data is bigger than a normal HTTP process could handle, the recommended third method will be a *rsync* as the third possible scenario. This command will provide a bigger level of synchronization and data transfer. If after some years, the volume of data is too large or the speed is not high to cope with the operation, an extra fourth option will be enabled to do a bulk copy of the archive by just mirroring a disk.

This feature has not been written for the release that is presented in this article.

Preferences This section features the user preferences. Users can have several levels of search depending on their profile. The less restricted profile will be granted access to more than 180+ searchable features in the database including searches from several external catalogs and be able to query the database in its native language.

This section is also a place to check out queries previously saved from other session searches as a kind of "query bookmark".

Help This section will instruct the users on how to use the search tool in depth. More examples and a full instruction manual of the query language is also planed.

3.1.2 Javascript Embedded Controller

This controller is embedded within the web pages as is shown in Figure 2 and provides a communication path among the view controller (in the web page) and Business Logic Controllers running on the Glassfish server. The application server then feeds the JSON structures to inflate the charts and tables with the requested datum by the user through the GUI.

3.2 Business Logic

This layer is the middleware between the View and and database and it has been fully developed on Java EE Web Services. As it is shown in Figure 4, the architecture provides two kinds of services to the View; RESTful and WebSockets. Real-time services are provided by the RESTful interfaces and services that need to be deferred in time are supported by WebSockets.

The services provided by the RESTful interface for OCS, CPS, APS and for any other contributed data products from the WEAVE's members are:

- Search view
- Spectra plot
- Paginated table view
- Histogram plot
- Heatmap Plot

The service provided by the WebSocket interface for the same sources is:

- Scatter Plot

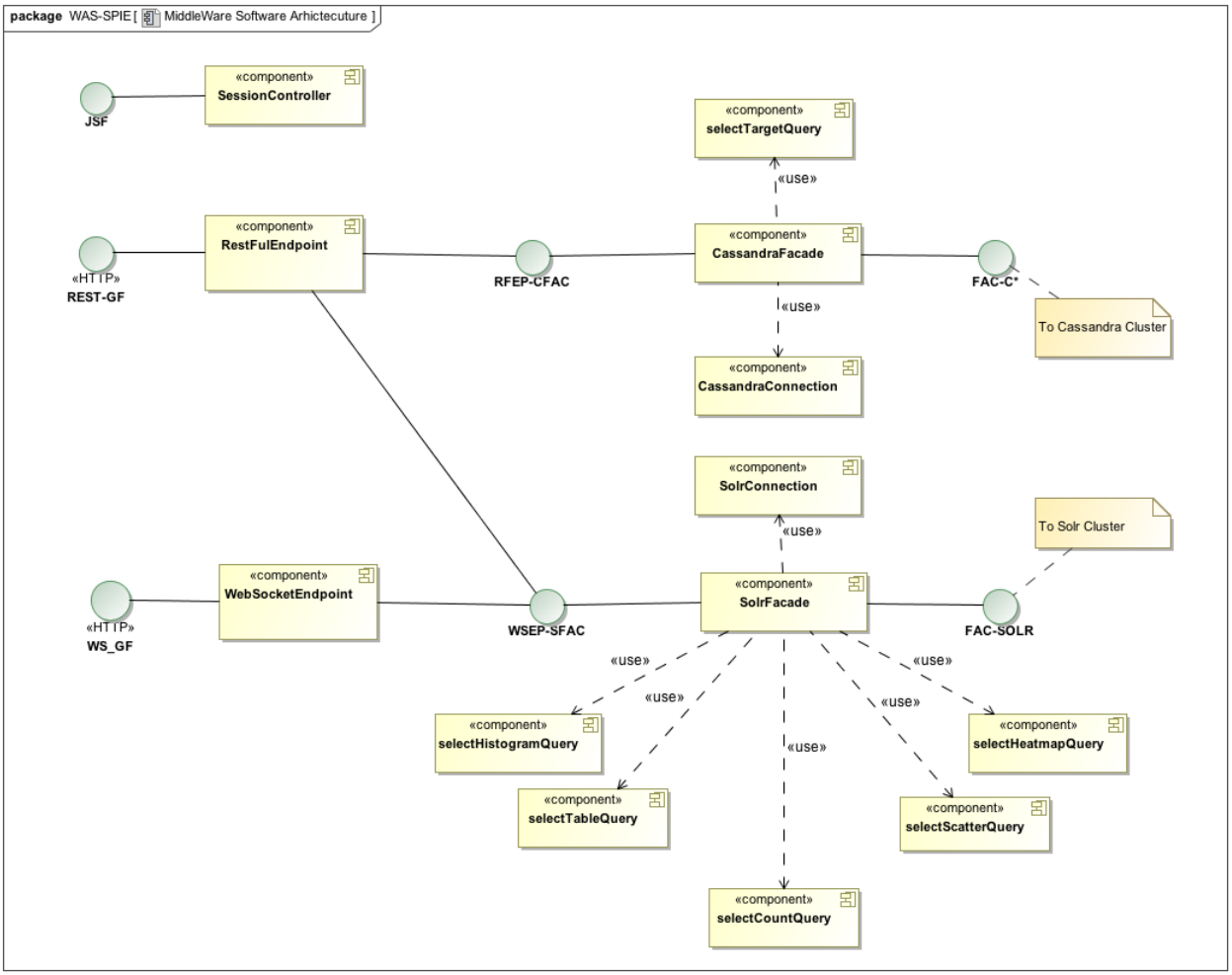

Figure 4. Core WAS business logic Software Architecture

3.3 Data Model

The data model is currently not yet frozen, and there are ongoing discussions regarding WEAVE's data products. However, there is already a first release of the data model, which forms the baseline for our current test environment. This release of the data model is composed of a list of the released FITS header from OCS, CPS and APS pipelines.

Following the described architecture for the database, a cluster was created with four nodes as described in Section 2 and having two datacenters; Cassandra Datacenter (**DC_Cassandra**) and Solr Datacenter (**DC_Solr**). The schema of the test database is based on a Cassandra keyspace and two tables. One of the tables keeps the header information and the other table keeps the spectra information. Currently WAS tests include four simulated spectra belonging to the two WEAVE spectrograph arms (Red and Blue) and the other two for the synthetic spectra.

These headers were combined in a single table on the Cassandra database which is called the ***mos*** table. Spectra are kept in another table, ***mos_flux*** with four 6k randomly generated spectra simulating both of WEAVE's arms.

3.3.1 Header table

With the current WEAVE documents, it was possible to define a first data model for this table. This model includes as described before, a single table keeping all of WEAVE's OCS, CPS and APS FITS headers parameters. So far, there are 281 columns in the table. The general structure of this table is conceptually described below. Contributed data products will also be additional columns of the same table.

	ID	CAMERA	OCS	CPS	APS
			...	...	...
			...	...	...
			...	...	...

Table 1. MOS Table

In Cassandra's terminology, the table has a composed primary key formed by the ID as a partition key and CAMERA as the clustering key. However this configuration may change in the future. The OCS section has 91 columns, the CPS has 31 columns and the APS has 159 columns.

3.3.2 Spectra table

The spectra table has 5 columns as shown below. Every spectra has been coded as 6K long CSV values. In principle, this table is not meant to be part of the final WAS data model design; its inclusion is for performing other tests related to analytics with Spark in a similar fashion to those ones performed for HARPS-N as described in[8] (E. Molinari et al)

	ID	RED	BLUE	SYNTHETIC_RED	SYNTHETIC_BLUE
		...			
		...			

Table 2. Flux table

4. SIMULATION OF DATA

This section shows the numerical process of simulating target data from GSC2 (Guide Star Catalog V2) in order to run the search tests. For the requirement to create a catalog of 20 million spectra, we use the actual GCS2 catalog at the TNG as the input. In the catalog we use the parameters: *RA*, *DEC*, *V*, *J*, *class* (0=star, 3= nonstar, which is assumed as galaxy). After the selection in the two magnitudes, $5<Vz<16$, $0<J<100$, the ingestion of a subset of almost 6 million of objects was started. This correspond to roughly a year and a half of WEAVE observations.

4.1 Stars

The relations between physical parameters are extracted from the database of Boyajian et al (2013). Their Table 7 (as per published erratum) allows us to determine the SDSS *g, r* and *i* magnitudes.

Assuming linear relationships between color *g-V*, *r-V*, *i-V* and *B-V,* the formulas thus obtained and used in the simulations are:

$$g - V = 0.2744 * (V - J) - 0.0199$$

$$r - V = -0.2604 * (V - J) + 0.1473$$

$$i - V = -0.5097 * (V - J) + 0.1473$$

The spectral type (from A0 to M5.5) is derived from their Table 6, and we code the type as a float number. We choose the *V-K* color as the input variable to determine spectral type and $T_{\rm eff}$. The *V-K* color is derived from the same Table 7 of Boyajian et al (2013).

We use the inverse relation (*V-J* vs *V-K*) to check the sensibility of the spectral type to the values of *V-J* we have in our GSC2 catalog.

$$V - J = 0.7354 * (V - K) + 0.0049$$

As the curve is not simple, we define two regions for the fit of the spectral type to the *V-J* color, being aware that for late types (>M2) the relation will not be physically meaningful.

$$iType = 20.3047 * (V - J) - 0.4289 \qquad (V - J) < 1.64$$

$$iType = 5.832 * (V - J) + 23.2647 \qquad (V - J) > 1.64$$

For the computation of $T_{\rm eff}$, the data in Table 6 of Boyajian et al (2013) were fitted as well as their regression formula. Finally the equation becomes:

$$T_{\rm eff} = 8817 - 3416 * (V - J) + 707 * (J - V)^2 - 50.5 * (V - J)^3$$

5. SEARCH TESTS

The tests performed on the Cassandra/Solr clusters have been split into four categories or batches of tests during the ingestion phase, searches, deep paging and finally displaying information.

The simulated data for all these tests were based on the general GSC2 catalog as described in the previous section.

A single instance of the Postgres database was also set up also to compare certain queries, especially deep paging, using the same structure of table and data.

5.1 Ingestion Phase

The ingestion phase was tested using two scenarios; importing GSC2 data targets from the Oracle database and importing the targets from a Cassandra/Solr TNG cluster. They were performed on a virtual machine with 8GB RAM and 4 CPUs and running CentOS 7.1. In both cases, the query was filtered by $0 < V < 16$ and $0 < J < 100$ and the ingestion process dumped around six million of simulated WEAVE targets into the 281 columns of the ***mos*** table as described in section 3.3.1.

For the spectra table, four 6k-long spectra were simultaneously generated according to the table described in Section 3.3.2 into the ***mos_flux*** table.

During this phase, in both ingestion scenarios, the loading speed of the ingested data came from the performance of the client, not from the database. It was clearly seen that the ingestion performance improves simple spawning more processes from one to four processes. This was possible simply by splitting up ranges of V magnitude or either RA or DEC.

5.1.1 Importing GSC2 from Oracle

The Java client was connected to a single instance Oracle database using a JDBC connector and performing 527,070 rows in 8 minutes with a single thread. The drawback of these method is that JDBC connector stores the resulting set in memory and runs out of memory in about 2.5-3 million targets and thus the ingestion process needs to be done in several stages to avoid this issue. The client process was running limited to 4GB of RAM and 4 CPUs.

5.1.2 Importing GSC2 from Cassandra/Solr

Another Java client was connected to a production TNG Solr server machine indexing a GSC2 table different to the WAS tests. This machine acts as the source for the GSC2 targets different to the Oracle one described before. This case uses a different strategy to the previous section starting from the fact there is no main-stream JDBC connector for Solr. The result was that reading GSC2 from the Solr server (single instance) is slower and thus the ingestion performance reached only 23,600 rows in approximately 8 minutes. In order to speed up the process, two instances of this client simultaneously ran two different ranges of V magnitude of the GSC2 catalog, pumping the database at a constant rate of around 31,000 targets in 8 minutes. The source Solr node was running with 6GB of RAM and 4 CPUs.

On the other hand, the client doesn't need to take special care with the returned dataset because the number of targets per page/fetch is preset at 250 and the client never runs out of memory due to Deep Paging feature of Solr. This preset can be changed (increased/decreased) depending on the cluster configuration and its architecture. This feature is described below in Section 5.3.

5.2 Searching Phase

Searching is one of the Solr's biggest strengths. With more than 180 indexed parameters, the database can be traversed with a large number of filter conditions giving the opportunity to run real-time searches.

Solr's pagination mechanism allows the production of a first set of results with 20-100 rows and 8 columns in no more than two seconds on average including spatial conditions in addition to two or three further filter conditions. Optionally, the size of the returned pages can be set up dynamically with up to 500 rows depending on the query. Searches can include not only spatial features but also ranges, strings with wildcards or literal values following Lucene Frameworksyntax.

Another important feature about Solr searches is that the queries from users do not monopolize the cluster computer power, giving only the resources necessary to bring up the requested page. This allows more user interactions against the database with fewer resources. Running on orphan queries issues is not the case for Cassandra/Solr clusters, as all of the queries are paginated and time-outed to avoid killer queries.

5.3 Deep Paging[¶]

What it is called Solr's Deep Paging is actually Large Queries in the SQL world, affecting thousands or even millions of rows. These searches normally involve large volumes of rows being transferred from the database to the client side in order to perform some bulk download. Under this scenario, Postgres clearly outperforms Solr clusters doing this in small datasets of 30,000 rows. Postgres is able to do a full-table scan operation on this dataset in one second or less with a single user. This same test was performed on Solr cluster and it was noticed that the Solr cluster displays a much slower performance under this scenario over Postgres database because of this pagination system which protects the cluster resources to be overwelmed with large dump operations instead.

On the TNG WAS test environment, the Solr cluster is able to pump out a sustained rate of around 200-400 targets per second in its first hit, which is far less than the rates that came out of Postgres that is able to pump out the entire 30K dataset in one second or less. After a second hit with the same query, the performance improves dramatically reaching up to 1,000 to 1,500 targets per second due to its caching mechanism. However it is still far away from the Postgres performance though, this operation is ran more safely on Solr specially when the number of users is increased.

Performing the same large queries over a much larger dataset, 5 million rows, Solr continues performing exactly with the same rates.

5.4 Displaying Searches

The WAS test GUI currently displays data in five different basic formats; paginated tables, histograms, scatter plots, heatmaps and sky plots.

5.4.1 Paginated Tables

Pagination the results is a very basic way to interact with the requested search. The WAS test GUI has a specific page where there are several columns representing a pre-selected list of key target parameters and in which the user can also use different embedded features such as row selection and spectral plotting. The size of the pages can be also settable from 20 to 500 targets.

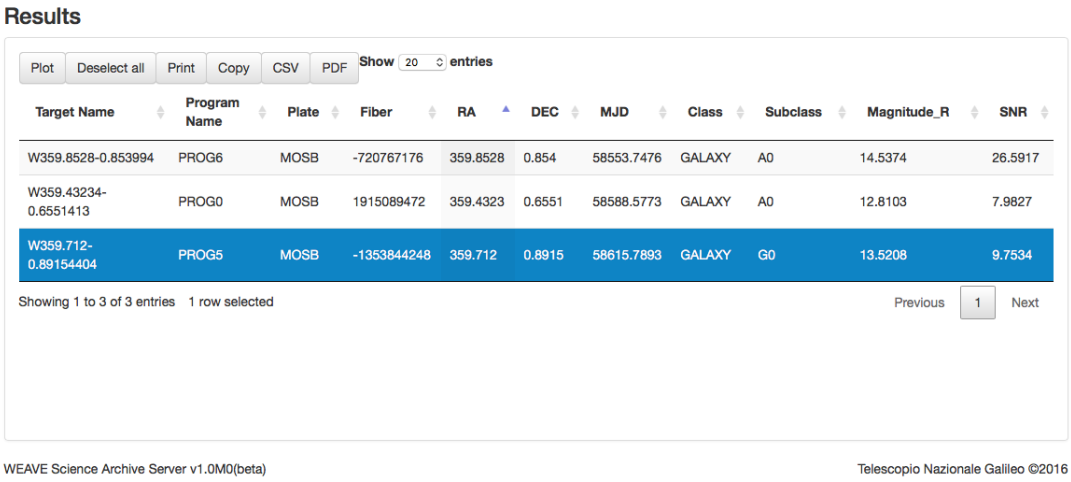

Figure 5. Paginated table view

The page can be reached from the main menu via Display -> Paginated Table. In Figure 5 gives a general view of the current feature.

[¶]Further information about Solr's deep paging feature can be found at: https://cwiki.apache.org/confluence/display/solr/Pagination+of+Results

5.4.2 Histograms

Histograms are one of the key ways of displaying real-time analytics and is a really strong asset for WAS analytics. The way to select a histogram over a previously selected dataset is to go onto the menu Display -> Histogram. This new web page has available all the possible parameters separated by pipeline (OCS, CPS, APS) and available to be displayed as histogram. Every displayed parameter has two sub-parameters that shall be defined before the query is launched; its range and binning. With these two sub-parameters, the user is available to select a specific view of the variable to display. All the tested histograms are performed in less than a second for the entire dataset.

Similar results were obtained with much larger pre-test datasets of over 20 millions of targets.

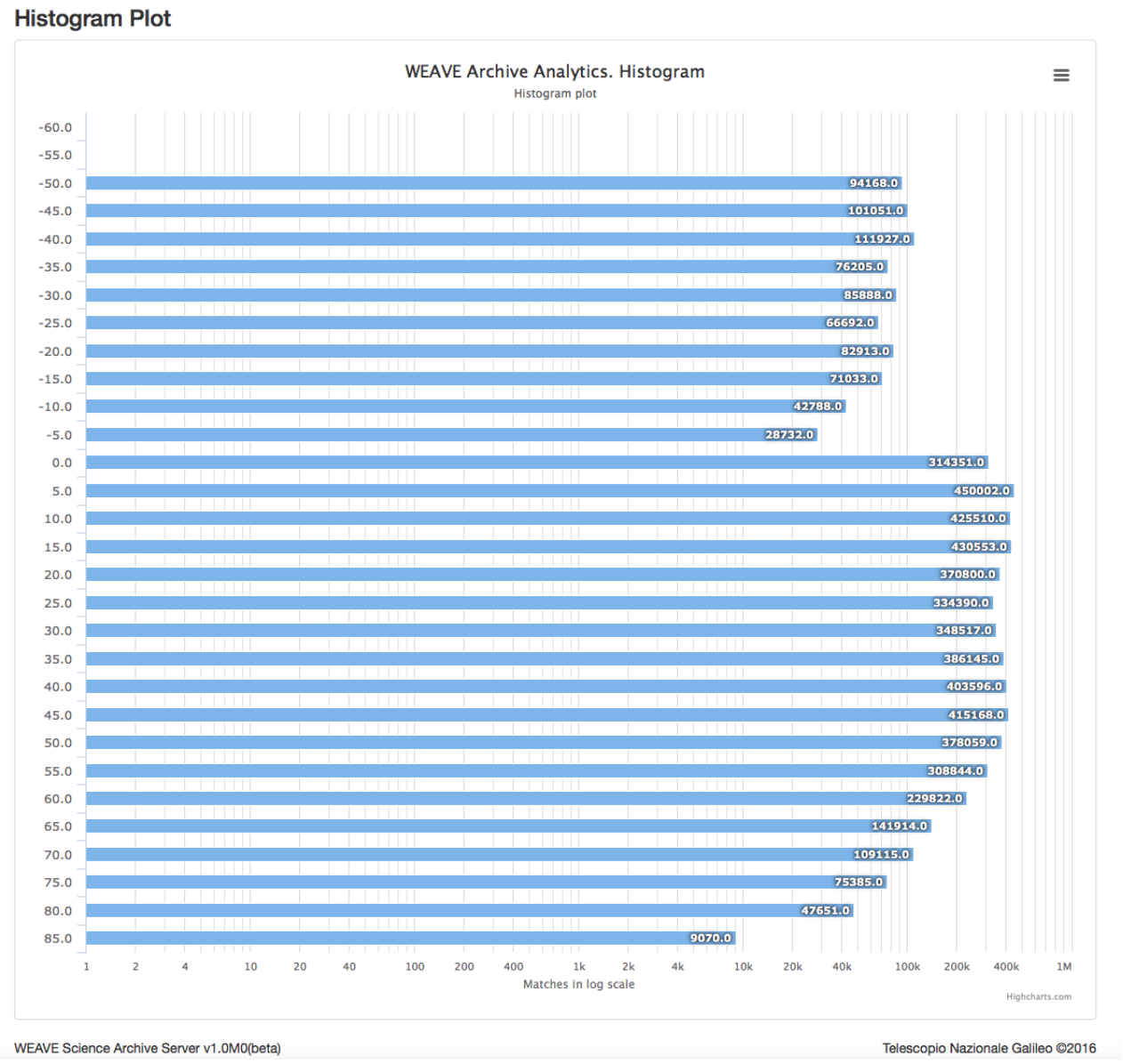

Figure 6. Histogram plot for *DEC* coordinate with binning of 5 degrees for 5 millions of targets.

5.4.3 Scatter Plots

These plots brings us to 2D analysis. The way to select this plot over a previously selected dataset is through menu Display -> Scatter Plot. This brings up a web page driven by the pipeline again as before and in which the user can select two parameters among all of the available parameter to be displayed for this case. This is actually the slowest diagrams to create due to poor dump capabilities explained above in Section 5.3.

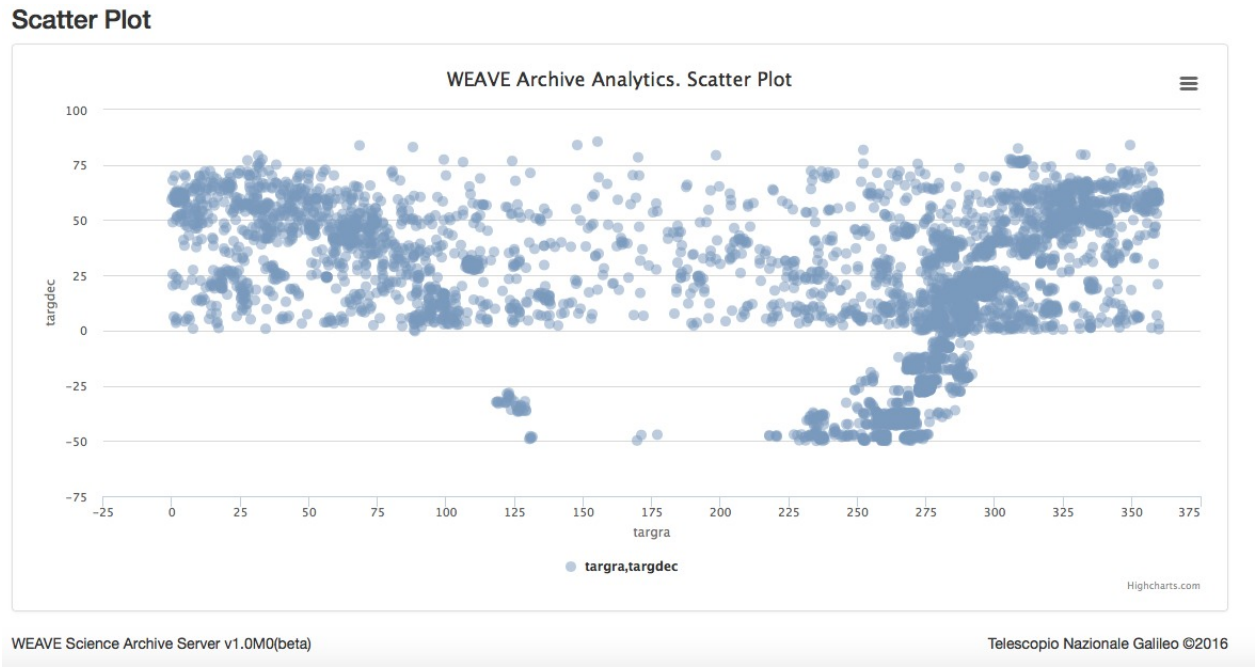

Figure 7. Scatter plot between *RA* and *DEC* from 5,402 GSC2 targets

5.4.4 Heatmaps

This is one of the most powerful diagrams available in WAS analytics. WAS analytics can perform 1,000 elements heatmap over any indexed pair of variables in less than 1 secs on average running on the described test environment.

This plot comes in two flavors; Low Density Heatmaps shown in Figure 8 and the High Density Heatmaps shown in Figure 9. These heatmaps can be resolved in as short as a few seconds over the same test dataset. They are reachable from the main menu; Display -> LD Heatmap/HD Heatmap.

The distinction between these two flavors is due to the library used to display the results more than the design aspect itself.

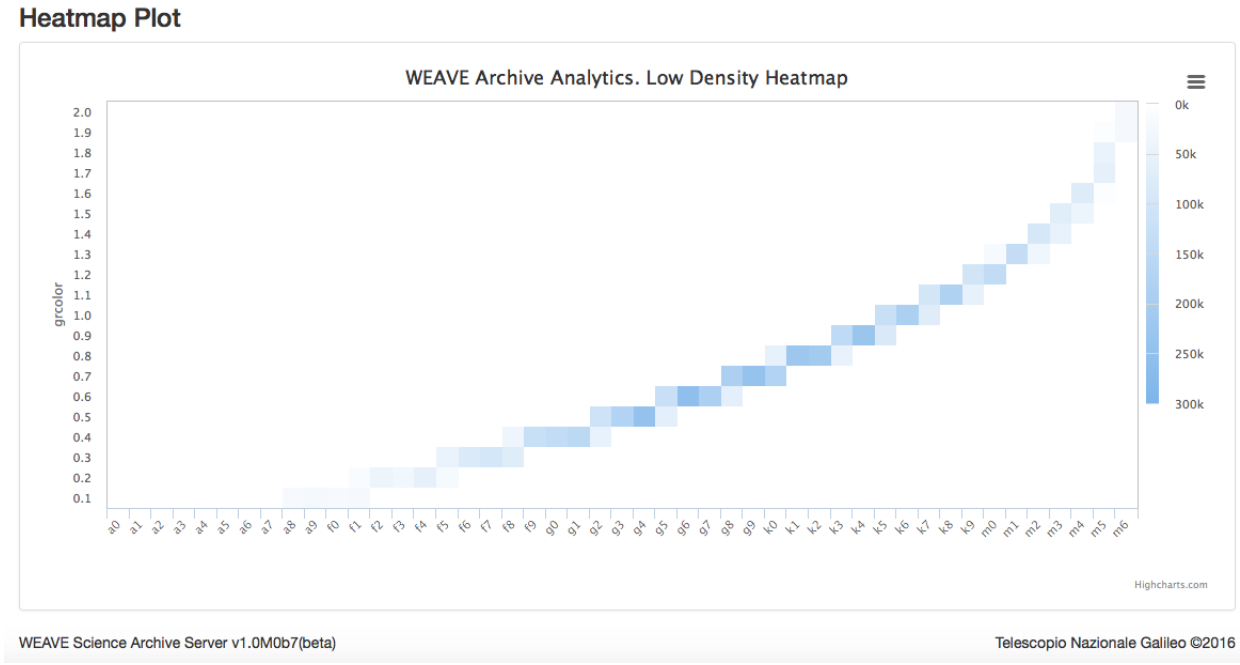

Figure 8. Low Density Heatmap. 940 values for *g-r color* vs star *subclass* for 5 million of targets.

The differences between these kind of plots is that the high density one is able to display data with a resolution of 65,000+ elements with smoothing algorithms as an extra option such as Delaunay triangulation. The low density one one the other hand, can only plot 1,000 elements max and there is no smoothing option. The web pages are also driven by the pipelines and the user could select two parameters of his/her choice and run a heatmap computation. The test GUI only allows to plot these heatmaps with *RA* vs *DEC*.

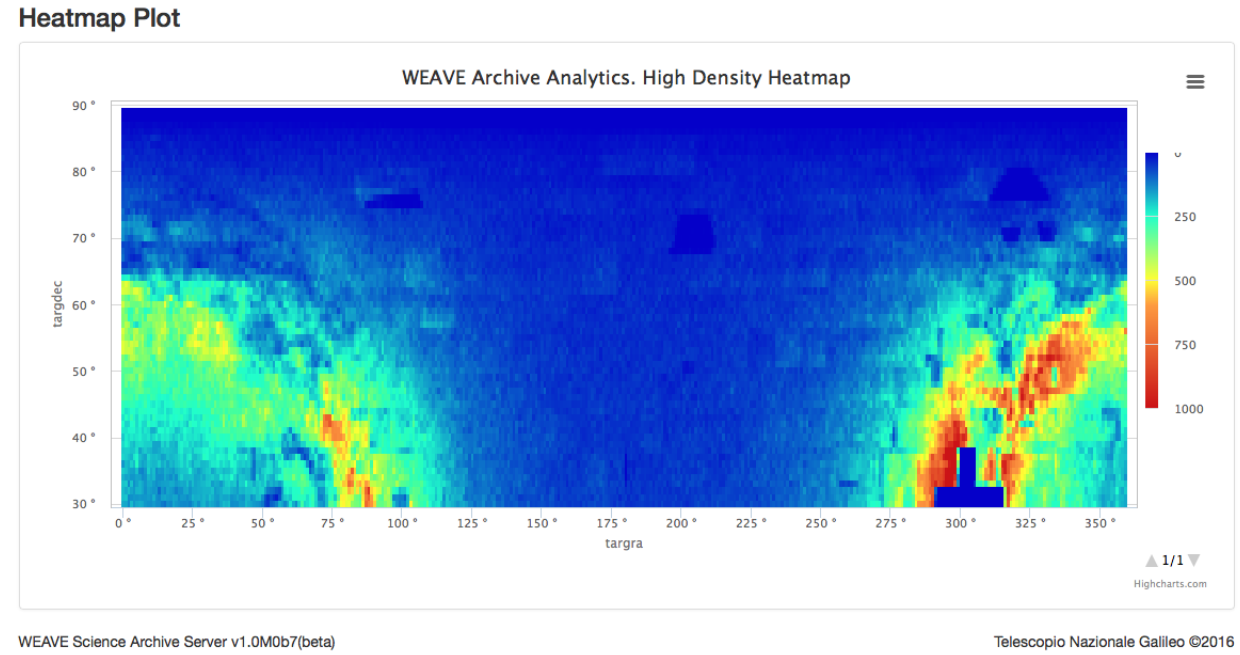

Figure 9. High Density Heatmap. 21,600 values for *RA* vs *DEC* evaluated for 5 million GSC2 targets.

5.4.5 Sky Plots

Among all of the plot possibilities, sky plots are important for locating the results of the analytics in spatial coordinates instead of a Cartesian system. Currently WAS is planned to support equatorial and galactic coordinate representations as displayed below. There is also the possibility to have healpix representation at several levels.

The diagram can not only pinpoint special features but can also plot large regions of the sky, making possible a nicer projection that can lead to an enhanced experience of the data.

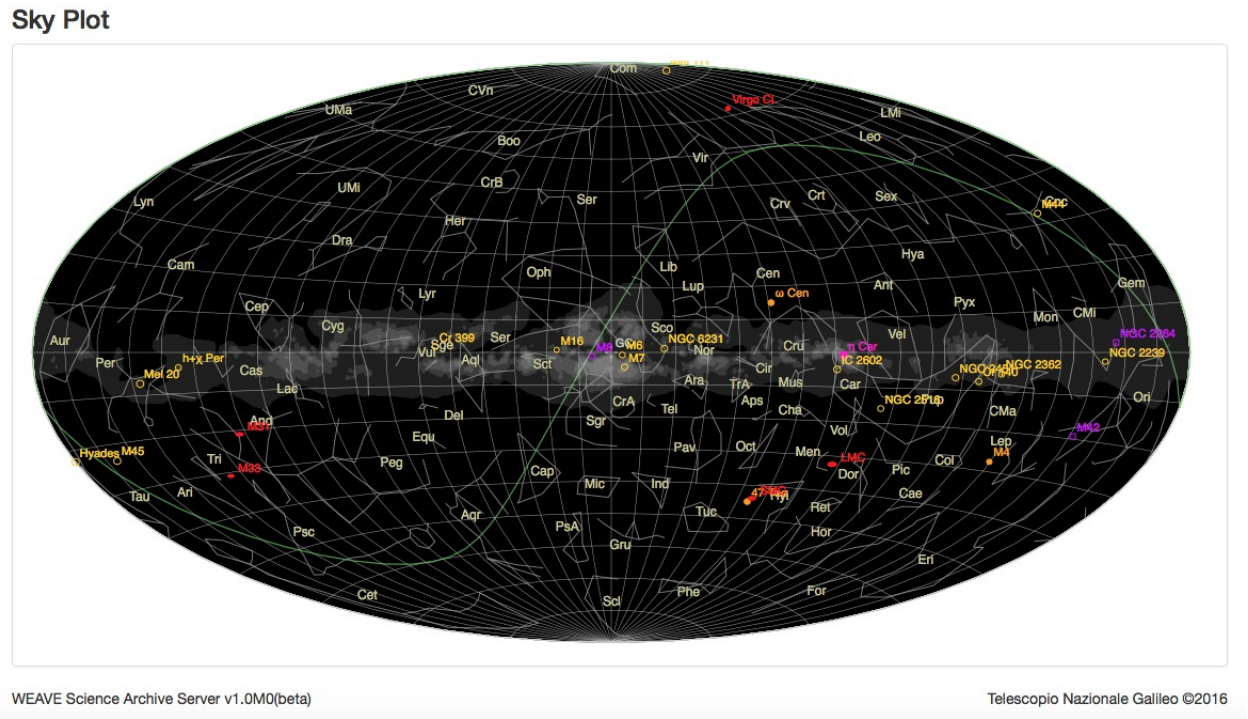

Figure 10. Main Milky Way galaxy features in galactic coordinates.

6. CONCLUSIONS

This article presents a new database concept for astronomical surveys, specifically in order to enable data from the spectrograph WEAVE to be treated using a big-data database kit, including real-time analytics for this scientific community. At a glance, this article presents some answers to different questions about the WAS data archive architecture at all levels. WAS shall provide a quick and reliable service to the scientific community in order to enable rapid progress to be made using the observational results and also share out the information in a reasonable way by different means; files, graphics and in tabulated form.

The WAS database design has been driven by four five aspects; storage capacity, searching capabilities, exporting data and analytics.

Given these aspects, two main database options were analyzed; NoSQL and SQL. As the NoSQL database, Cassandra and Solr were selected as a combined package from DataStax Enterprise. On the SQL side, PostgreSQL was the selected database for the tests.

6.1 Storage Capacity

Both options (Cassandra and PostgreSQL) are able to cope with ranges of 7-12 TB of data which is the range of data we foresee needing to be handled for the WEAVE surveys. Cassandra scales up extremely well in the sense that it is a distributed database and able to grow as the survey progress. On the other hand it requires more disk

space to keep the replicas across the cluster by approximately a factor of four for the test environment however increases the availability of the data making a more reliable system. Postgres has a single instance architecture, but with the right disk array storage, it can also scale up in storage capacity as well. The drawback is that it does not increase in availability and computer power unless a CPU or memory upgrade is scheduled too.

6.2 Searching Capabilities

Solr clusters are very specialized fo searching features in the database. Solr is able to index complex column structures and data formats. It currently indexes 180+ columns on the WEAVE test environment making possible to traverse the database for deep data mining operations. With these premises, Solr is able to resolve all tested search queries, including spatial features, in ranges of very few seconds from either the test GUI or its native console.

6.3 Exporting data

This is the main drawback found on Solr clusters. These clusters are not meant to be for doing massive full-scans or uncontrolled large dump operations. These operations are slower in comparison with Postgres in the tested range of small datasets with 30K rows. Solr features a query structure called *deep paging* explained above in Section 5.3, that it makes possible to improve sustained large dump ratios safely without hurting the cluster resources.

6.4 Analytics

On Cassandra/Solr clusters, analytics is performed by the Solr cluster. Solr supports faceted search[4] which is a high valuable feature for WAS real-time analytics. On Postgres you can do store procedures, third-part packages or just plain SQL to get similar results but with an extra penalty time to implement, produce and execute similar results.

REFERENCES

[1] E. Molinari, M. Lodi, J. Guerra, C. Benn, L. Dominguez. "WAS: Textures of the WEAVE Science Archive". Multi-Object Spectroscopy in the next decade. Big Questions, Large Surveys and Wide Fields. Santa Cruz de La Palma, Canary Islands, 2-6 March 2015.

[2] Barbieri C. "Galileo Italian National Telescope and its Instrumentation". Astronomical Observatory of Padova, Italy, Proc. SPIE 2871 (1997).

[3] Avinash Lakshman, Prashant Malik. "Cassandra - A Decentralized Structured Storage System". Facebook. 2009.

[4] Giovanni Simonini, Song Zhu. DIEF, Univ. of Modena & Reggio Emilia, Modena, Italy. "Big data exploration with faceted browsing". IEEE High Performance Computing & Simulation (HPCS), 2015 International Conference on. July 2015.

[5] Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall and Werner Vogels. "Dynamo: Amazon's Highly Available Key-Value Store". Amazon.com. Oct 14, 2007

[6] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber. "Bigtable: A Distributed Storage System for Structured Data". Google Inc. 2006.

[7] Gavin Dalton, Scott C. Trager, Don Carlos Abrams, David Carter, Piercarlo Bonifacio, J. Alfonso L. Aguerri, Mike MacIntosh, Chris Evans, Ian Lewis, Ramon Navarro, Tibor Agocs, Kevin Dee, Sophie Rousset, Ian Tosh, Kevin Middleton, Johannes Pragt, David Terrett, Matthew Brock, Chris Benn, Marc Verheijen, Diego Cano Infantes, Craige Bevil, Iain Steele, Chris Mottram, Stuart Bates, Francis J. Gribbin, Jürg Rey, Luis Fernando Rodriguez, Jose Miguel Delgado, Isabelle Guinouard, Nic Walton, Michael J. Irwin, Pascal Jagourel, Remko Stuik, Gerrit Gerlofsma, Ronald Roelfsma, Ian Skillen, Andy Ridings, Marc Balcells, Jean-Baptiste Daban, Carole Gouvret, Lars Venema, Paul Girard. "WEAVE: the next generation wide-field spectroscopy facility for the William Herschel Telescope". SPIE Proceedings. Volume 8446. Multi-Object Instruments II. 2012

[8] Emilio Molinari, Jose Guerra, Avet Harutyunyana, Marcello Lodi, Adrian Martin. "The HARPS-N archive through a Cassandra, noSQL database suite?". Telescopio Nazionale Galileo. SPIE-9971-125. June 2016