G. PIERANTONI
B. COGHLAN
E. KENNY

THE USE OF STANDARDS IN HELIO

Abstract

HELIO [8] is a project funded under the FP7 program for the discovery and analysis of data for heliophysics.

During its development, standards and common frameworks were adopted in three main areas of the project: query services, processing services, and the security infrastructure.

After a first, proprietary implementation of the security service, it was suggested moving it to a standard security framework to simplify the enforcement of security on the different sites. As the HELIO front end is built with Spring and the TAVERNA server (HELIO workflow engine) has a security framework compatible with Spring, it has been decided to move the CIS in Spring security [2].

HELIO has two different processing services: one is a generic processing service called HELIO Processing Services (HPS), the other is called Context Service (CTX) and it runs specific IDL procedures. The CTX implements the UWS [4] interface from the IVOA [5], a standard interface for job submission used in the helio and astro-physics community. In its final release, the HPS will expose an UWS compliant interface.

Finally, some of the HELIO services perform queries, to simplify the implementation and usage of this services a single query interface (the HELIO Query Interface) has been designed for all these services. The use of these solutions for security, execution, and query allows for easier implementation of the original HELIO architecture and for a simpler deployment of the services.

1. HELIO: An infrastructure for Heliophysics

To be an effective infrastructure for research, HELIO must offer support across all the different steps that a scientist performs in his inquiry. Furthermore, it should offer this support in the modalities that the scientific community deems most useful. A common, albeit not entirely comprehensive, approach to scientific inquiry in Heliophysics can be decomposed into four main steps:

- The first step (*metadata search*) is devoted to identifying interesting phenomena. This phase uses only metadata which has been extracted from observations.
- The second step (*instrument search*) consists in the search of instruments that have observed the events and features of step 1. This can be quite a complicated task as the search must be undertaken in 4 dimensions across several domains, from the sun itself to the end of the solar system following the phenomena as they propagate.
- The third step (*observations search*) uses the list of instruments obtained in the second step to review the availability of suitable observations by querying lists with the position and type of observations of every instrument.
- The fourth step (*data search*) consists in locating, selecting, and eventually retrieving the data on the Internet related to the observations selected in step 3.

  HELIO offers services to each of these steps:
- For the first step services that query catalogues of events and features,
- For the second step HELIO offers a service that queries the positions of instruments scattered across the solar system, models to track the propagation of phenomena from the sun to the outer regions of the solar system, and a Coordinate Transformation Service (CTS).
- For the third step HELIO offers services that query catalogues of instruments by capabilities, position, and status at any given time.
- For the fourth step, HELIO offers a service, called DPAS (Data Provider Access Services) that knows which types of data are held at what place and how to access this data.

As said at the beginning of this paragraph, this four-step model is widespread but does not cover entirely all the possible approaches in the Heliophysics community; some scientists may want to start from the analysis of data as they extracted the information of steps 1 to 3 from scientific literature or an exchange of views with colleagues and only then look for metadata information, others may choose other approaches altogether.

To cater for all this variety, HELIO offers various ways in which to access and orchestrate its services. All these access modalities are comprised in a layer of the architecture called Access Layer. Most HELIO services expose their own standalone graphical user interface but they are also all connected to a centralized graphical user interface named HELIO Front End (HFE), a browser-based user interface intended to solve the most common tasks. Figure 1 shows as an example three services two

of which (Service B and C) expose standalone graphical user interfaces, while one (Service A) is only connected to the HFE.

The HFE is also the best choice for the less experienced users for its user-friendliness and for its capability to allow users to connect services to each other to explore new orchestration patterns. For more advanced users HELIO exposes an IDL API that connects the system directly to custom-made IDL programs, and for expert users HELIO offers programmatic access through a Java library (HELIO Client API) that simplifies this access by providing client stubs for services. Furthermore, HELIO services can also be handled through the Taverna Desktop workflow tool to orchestrate several services into more complex workflows.

Finally, HELIO not only accesses existing metadata information sources or observational data but also allows us to process data to create new metadata catalogues or derived results. Specific services offer computation, storage, and an IDL environment to execute metadata extraction algorithms.

2. The architecture of HELIO

The overall architecture of HELIO is described in Figure 1; it is a multi-layered service oriented architecture.
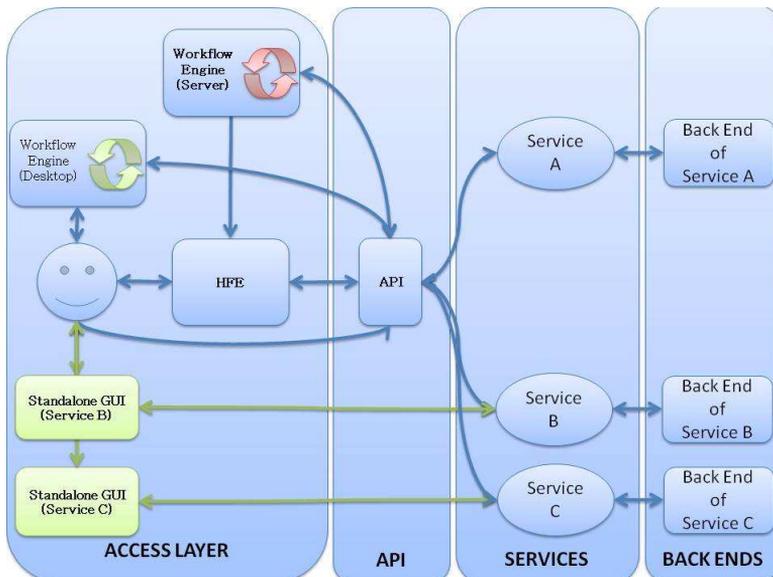
Figure 1. The HELIO architecture

The *Access Layer* comprises three different elements: the HELIO Front End, the Taverna Desktop [6] that the scientists can use to experiment and evaluate different orchestrations of the HELIO services, and a Taverna server instance used to execute

procedures that are well-established by the scientific community and will be available as pre-defined workflows on the HELIO Web Site.

The *HELIO API* comprehends the stubs of the various services and offers an entry point that contains all the logic to hide all the information flows that are not necessary to the user.

There are of four main kinds of services in HELIO that mirror the four basic steps, as outlined below.

2.1. Metadata services

These are services that offer query capabilities to metadata catalogues. They are used to find information on events and features that relate to solar phenomena.

- the *Heliophysics Event Catalogue (HEC)*: a catalogue that queries several dozens of solar and heliophysics event lists,
- the *Heliophysics Feature Catalogue (HFC)*: a service that employs image recognition algorithms to extract features of the sun. The HFC outsources the execution of computationally intense algorithms to the *HELIO Processing Services (HPS)*, one of the *Enabling Services* of Section 2.4 below.
- the *Data Evaluation Service (DES)*, and
- the *Context Service (CTX)*.

2.2. Discovery services

These are services that provide information on which instrument has observed a specific phenomenon. To do this they take into account the position of the instrument in the Heliosphere as well as its operating periods.

- the *Instrument Capabilities Service (ICS)*,
- the *Instrument Location Service (ILS)*, and
- the *Observation Coverage Service (OCS)*.

2.3. Data access service

The *Data Provider Access Service (DPAS)* offers a single, unified access point to the observation data.

2.4. Enabling services

These services offer access to computation, storage, security, coordinate transformation, and semantic mapping services.

- the *HELIO Processing Service (HPS)* and the *HELIO Storage Service (HSS)* that offers access to computation and storage facilities.
- the *Community Interaction Service (CIS)* that offers authentication and a centralized repository to store user's preferences.

- the *Coordinate Transformation Service* that offers conversion between the different coordinate systems used in the solar and heliophysics domain, and,
- the *Semantic Mapping Service (SMS)* that compares data with a different structure by using an ontology for mapping it to a global mode.

3. The use of standards and common services in HELIO

Adoption of standards and common platforms are beneficial for two main reasons: firstly in distributed computing infrastructures like HELIO they aid development by encouraging homogeneity of design solutions and the implementation of the best practices across all the partners of the project. Secondly, they allow the system to be more open to external projects and users.

Although there was no explicit planning from the beginning, the use of some standards and common platforms naturally arose during the design and development phase in three main areas of the HELIO architecture.

- Many of the HELIO services perform queries in catalogues and lists of data and metadata; not only do these services perform a similar action but they also use a common data format known as *VOTable* [7, 1], hence there is potential for harmonization. The VOTable format is an XML standard for the interchange of data where data is represented as a set of tables.
- Two of the HELIO services (the *HELIO Context Service (CTX)* and the *HELIO Processing Service (HPS)* offer computation resources. The same consortium that defined the VOTable also defined an interface for computation services, the *Unified Worker Service (UWS)* [4]. HELIO has adopted UWS for its Context Service and will adopt it for the HELIO Processing Service in its final release.
- Finally, the security infrastructure of HELIO, catered for by the *Community Interaction Service (CIS)*, offers an authentication service and a centralized repository for user's preferences. As the HELIO system is distributed and needs authentication and authorization services across components of different types such as code and web interfaces, it has been decided to adopt Spring Security [2], a widely used standard for security in distributed applications.

HELIO stands at the crossroads of three main areas: Web Services [18], Grid Computing [17], and Heliophysics [19].

In fact, HELIO connects different components (wrapped as *Web Services*) to help the *Heliophysics community* and some of these components use *Grid computing* based back ends.

Obviously, in each of these areas standards and best practices have been investigated and among them, many deal with security, data and meta-data exchange formats and remote job execution.

In Heliophysics the main efforts towards standardization are those of *International Virtual Observatory Alliance*, or *IVOA*, a worldwide scientific organisation that

aims at facilitating international coordination and collaboration to enable global and integrated access to data produced by astronomical observatories.

IVOA efforts cover many different fields; among these, the most significant to HELIO are the *VO-Table* format [7] and the *Universal Worker Service* specification [4] that define a common data format for all services that deal with data produced by astronomical observatories and attempts at defining an asynchronous protocol for job execution in distributed systems.

Security in Web Services [9] and more broadly web applications witnessed an increasing use of Spring Security [2] while the Grid Computing [10][11] community relies on the use of X509 certificates for Authentication and Authorization and has developed services such as *MyProxy* [12] and *VOMS* [13] for the secured storage and management of these certificates.

The Grid Community has developed well estabilished services and standards for the submission and execution of jobs such as those used in *gLite* [14], Globus [16] and *Condor* [15] while the Heliophysics community has defined its own standard [4].

HELIO has to carefully balance the benefits and constraints of all the three types of standards in order to be extensible to other similar and overlapping platforms (expecially other Heliophysics and Astrophysics software) while being compliant with the requirements of the web service layer and the grid-based back-ends. These constraints are particularly demanding regarding security.

For what regards security, HELIO uses Grid back ends (MyProxy and VOMS) in addition to standard web service security only for users that require a high level of security to access grid resources while the other users can use only standard login and password based security offered by the web service layer.

For what regards job submission, HELIO employs a combination of light-weight and Grid-backed computational resources to offer optimized execution for long and short running jobs. For this a specific processing service called HELIO Processing Service, or HPS finds the brokers for the optimal resource. Although it is not completed yet, the HPS will translate from the IVOA standards on the user's side to the grid-compliant standards on the back-end side.

3.1. The HELIO query interface

Some of the HELIO services (DPAS, ICS, ILS, HEC and HFC) perform queries; accordingly a single query interface (the HELIO Query Interface) has been designed for all these services where a parameter query can be used to query a set of tabular data.

The HELIO Query Interface supports both HTTP GET and SOAP. One of its variants, the *HELIO Long-running Query Interface (HLQI)*, supports results streaming and is ideal for query services that return large amounts of data and/or take a long time to complete the query.

The architecture of the HELIO Query Service, described in Figure 2, comprises a handler to manage all query requests, and a VOTable creator that creates the VOTables returned as results.
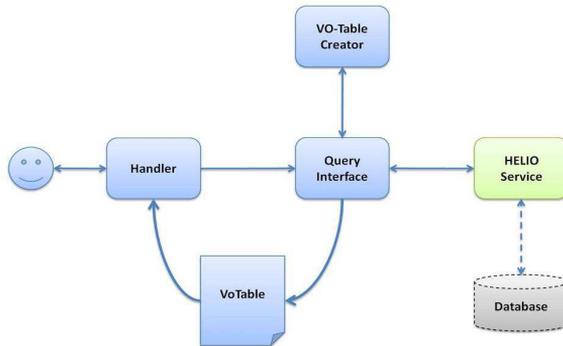

Figure 2. The HELIO Query Interface

3.2. The use of spring security

The architecture of the HELIO security framework is described in Figure 3.

The *CIS Server* holds a repository of users and their details. These details include not only information strictly pertinent to the accounts but also a set of preferences that each user can define to personalize the behaviour of the various HELIO services. The CIS Server acts as a repository for user details and as an authentication engine. Upon a successful authentication the CIS client will issue a valid, Spring compliant *Authentication Token* that the API can forward to the services that need to authorize users. Thus, the services can implement authorization following the Spring guidelines [3]. To support back-ends and services that require a higher level of security and the presence of a grid certificate, a proxy certificate can be added to the authentication token.

The API provides the Community Interaction Service with the information regarding the user and a unique identifier of the session. The CIS can issue tokens of the following kind:

- For users that do not want to login an "anonymous" identity token will be created.
- For those that login with a username and password, a token with their identity will be created.
- For users that identify themselves with a personal grid proxy certificate, the service will issue a high security token based on that certificate.

  Depending on the issued token, different privileges are given to the user:

- Anonymous users can only execute *canned jobs* with *canned data*; that is they can only select jobs and input data from pre-defined sets.
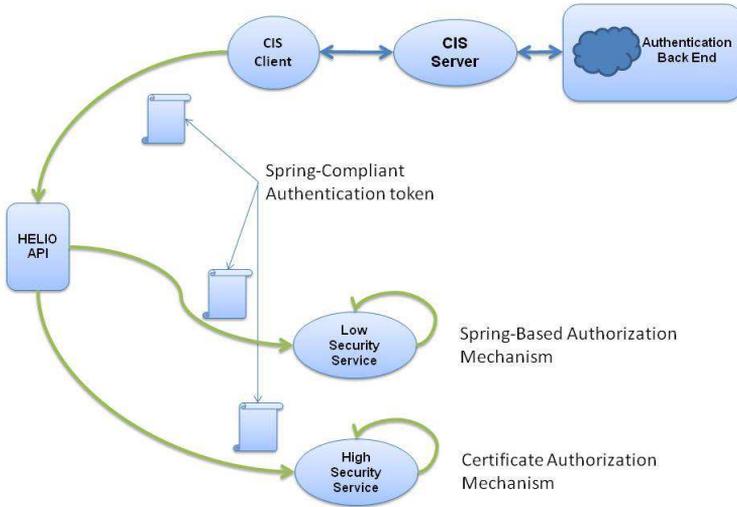
Figure 3. The HELIO Security framework

- Authenticated users withoout a grid certficate can only execute *canned jobs* with *canned data* and *user-defined data.*
- Authenticated users with a grid certficate can execute *canned jobs* and *user-defined jobs* with *canned data* and *user-defined data.*

3.3. The Universal Worker Service

HELIO employs two different services for computation, the HELIO Context Service (CTX) and the HELIO Processing Service (HPS). For the present the CTX exposes an interface called *Universal Worker Service (UWS).*

The UWS specification allows for the asynchronous submission, deletion, and status management of jobs on distributed resources. The adoption of UWS by the HPS too would bring two main advantages: a simplification of the API layer and a simpler connection to external helio and astro-physics related projects that adopt the UWS specification.

4. Conclusions and Future Work

The adoption of de-facto standards such as the IVOA VOtable plus UWS and the Spring security has greatly unified the variety of services in the HELIO distributed computing infrastructure. It is felt that this will aid its acceptance by its own community as well as provide an open basis for future exploitation and enhancement.

Although HELIO has reached a stable stage and will be completed in 2012, the adopted standards will allow it to further connect to other useful resources. More

specifically, the adopted standards will support HELIO in connecting to more computational services and more catalogue query services.

The adoption of UWS by HPS will allow the connection of HELIO to clusters that already host Heliophysics related codes, and there is ongoing discussions to extend the security infrastructure with more Spring-compliant Authentication Providers.

In the next years, HELIO's processing, storage, and security service will be used in the developement of a Heliophysics portal developed within the SCI-BUS project [20] and this will represent an additional test on the validity on the choices taken for HELIO's adoption of standards.

Acknowledgements

References

[1] VOTable 1.1 Specification,
`http://www.ivoa.net/internal/IVOA/IvoaVOTable/VOTable-1-1.pdf/`.

[2] Spring Security Project Page,
`http://static.springsource.org/spring-security/site/`.

[3] Spring Security Authorization Architecture,
`http://static.springsource.org/spring-security/site/docs/3.0.x/reference/authz-arch.html/`.

[4] Universal Worker Service Specification,
`http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices/uws.html/`.

[5] IVOA Project Page, `http://www.ivoa.net/`.

[6] TAVERNA Project Page, `http://www.taverna.org.uk/`.

[7] VOTable Project Page,
`http://www.ivoa.net/cgi-bin/twiki/bin/view/IVOA/IvoaVOTable/`.

[8] HELIO Project Page, `http://www.helio-vo.eu/science.php`.

[9] Web services and web service security standards, Christian Geuer-Pollmann and Joris Claessens, Information Security Technical Report, 2005.

[10] A security architecture for computational grids, Foster, Ian and Kesselman, Carl and Tsudik, Gene and Tuecke, Steven, Proceedings of the 5th ACM conference on Computer and communications security, 1998.

[11] A community authorization service for group collaboration, Pearlman, L. and Welch., V and Foster., I and Kesselman., C. and Tuecke S., Proceedings of the Third International Workshop on Policies for Distributed Systems and Networks, 2002.

[12] An online credential repository for the Grid: MyProxy, Novotny, J. and Tuecke, S. and Welch V., Proceedings of High Performance Distributed Computing, 2001.

[13] VOMS, an Authorization System for Virtual Organizations, Alfier, R. and Cecchini, R. and Ciaschini, V. and dell'Agnello, L. and Frohner, A. and Gianoli, A. and Lõrentey, K. and Spataro, F., Grid Computing, 2004.

[14] Programming the Grid with gLite, Laure, E. and Fisher, S. M. and Frohner, A. and Grandi, C. and Kunszt, Peter Z. and Krenek, A. and Mulmo, O. and Pacini, F. and Prelz, F. and White, J. and Barroso, M. and Buncic, P. and Hemmer, F. and Di Meglio, A. and Edlund, A., 2006.

[15] Condor and the Grid, Thain, D and Tannenbaum, T. and Livny, M., Grid Computing: Making the Global Infrastructure a Reality, 2003.

[16] GRAM5 Web page, `http://www.globus.org/toolkit/docs/latest-stable/gram5/gram5`.

[17] The Grid: Blueprint for a New Computing Infrastructure. Kesselman C. and Foster I. Morgan Kaufmann Publishers, November 1998.

[18] Introduction to Web Services Gunzer H. Borland, March 2002.

[19] The Sun, the Earth, and near-earth space Eddy J. A.: National Aereonautics and Space Administration.

[20] SCI-BUS Project Page, `http://www.sci-bus.eu/`.

Affiliations

G. Pierantoni
    Trinity College Dublin, Room 0.05, Lloyd Building, TCD, Dublin 2, Dublin, Ireland

B. Coghlan
    Trinity College Dublin, Room 0.05, Lloyd Building, TCD, Dublin 2, Dublin, Ireland

E. Kenny
    Trinity College Dublin, Room 0.05, Lloyd Building, TCD, Dublin 2, Dublin, Ireland