Mem. S.A.It. Vol. 85, 808

Memorie della

Web applications meet astronomical archives

Because astronomers are experienced internet users too

L. Nicastro[1], D. Ricci[2], and P. G. Sprimont[1]

[1] INAF, Istituto di Astrofisica Spaziale e Fisica Cosmica di Bologna, via P. Gobetti, 101, 40129 Bologna, Italia.

[2] IA-UNAM, Instituto de Astronomía – Universidad Nacional Autónoma de México, Km 103 Carretera Tijuana Ensenada, 422860 Ensenada, Baja California, Mexico.

Abstract. We present here the most modern internet and web technologies which we have tested and verified to be very suitable for the development of innovative, web-based tools for observational astronomy and archival data exploitation. Nothing but an updated browser is required to the user. This approach can boost the exploitation of huge data archives such as those that will be produced by projects like GAIA and TAOS-II, and will allow to easily manage heterogeneous data, like those of the IPERCOOL project. In spite these technologies are still not fully mature, they are already accepted standards in the browsers of our laptops, tablets and cellphones. We discuss the impact of these new technologies in astronomy and present test examples to show their capabilities.

Key words. Astronomical databases – Astronomical software

1. Introduction

The work of an astronomer is strictly connected with a computer, and with the internet as well, used everyday to get access to a wide range of web sites strictly related to their work: from bibliography / papers publication related ones (ADS[1], astro-ph[2]), to astronomical catalogues and object parameters archives (Vizier, Simbad – see `cds.u-strasbg.fr/`), image databases, and so on. Browsing these sites, it becomes natural to compare them to other sites we come across surfing the web. Doing that, it becomes clear that the usage of modern web technologies in astronomy related sites is very limited. A lot of textual sections, interaction limited to hyperlinks or "mapped" images, cross-browser incompatibilities, lack of vector graphics, an overall poor look and feel. We must admit that in some cases this does not affect too much usability and effectiveness of these sites, but in general it significantly reduces the quality of the user's experience and quantity of accesses, e.g. by non experts or users that make use of portable devices. Not to mention the pointless (nowadays) usage of heavy Java applets (e.g. Aladin[3]). Astronomers also use custom application, personally programmed to solve very specific tasks or, more often, developed

[1] `www.adsabs.harvard.edu`

[2] `http://arxiv.org/archive/astro-ph`

[3] `http://aladin.u-strasbg.fr/java/nph-aladin.pl`

within the astronomical community (e.g ds9[4], IRAF[5]). These suite of programs are being constantly maintained and adapted, sometimes with poor results, to meet new requirements. For example a web-based version of ds9 exists, but it has very limited capabilities (`http://js9.si.edu`). Overall these tools remain conceptually outdated. But astronomers also use other, general purpose graphics packages like paw[6] and smongo[7], that were instead partially or completely abandoned by their authors. This in spite more flexible and still free packages exist (e.g. GDL[8]).

Moreover, researchers deal with large amounts of data, daily produced by space- and ground-based telescopes, e.g. GAIA and TAOS-II[9]. Projects like IPERCOOL[10] have to deal with a variety of data, from images to spectra, to time series, and eventually get access to large astronomical catalogues, simultaneously. These are example of projects, we are involved in, that in modern astronomy represent typical cases, rather than the exception. However access to huge astronomical archives of images, spectra, light curves, etc., is performed through outdated web interfaces that require a significant amount of time (and skill) to give you exactly what you need. A modern interface should offer the means to apply in a simple and intuitive way filters, on any parameter, that would allow him/her to select what required. Displaying and allowing selections on graphically presented data is one way to reach this aim. Things that are implemented in web applications we use every day (to book a hotel, a plane ticket, browse a map, etc.) and that are particularly appreciated when we have in our hands a tablet or a smartphone. In fact they are being implemented by Silicon Valley giants and start up companies, in the new software approach that we learned to know with the abbreviation *"Apps"*. Web-tools have the extra advantage that there is nothing to install or maintain. Being connected to internet is the drawback to pay. But that's not completely true, thanks to the new `html5` caching system.

Just to mention another project we are involved in, the GAIA Coordination Unit 9 (CU9) aims at giving unrestricted access to the whole one billion-source final catalogue. Browsing the, likely, cloud-based archive, being able to easily select what is needed, and finally get the result in an immediately usable format is the sort of challenges that new web technologies can certainly help to tackle.

That's great, but still astronomy remains almost untouched from this technological leap. We believe that programmers and users must take advantage of these web-based technologies. In particular the user only needs an updated web browser to exploit these capabilities. An enormous saving of resources, especially in terms of human time.

Here we briefly present a modern approach to astronomical data managing, visualization and analysis based on the most innovative web tools available in the Web 2.0 field. To trace the path, we have implemented several test cases and discussed them in Ricci et al. (2013); Ricci & Nicastro (2013) and (Sprimont et al., 2014; Ricci et al., 2014b, submitted).

2. Why use a web browsers

The main advantage for switching from standalone to web-based applications is that a web page is an already accepted universal standard to interact with a computer, and it hides to the end user all the low-level components that can evolve independently. In particular browsers provide:
easy access: every device (pc, laptop, tablet, smartphone, smart tv) has a browser and it works more or less the same way everywhere;
user experience: to build a "website" one can use a large set of components, like hyperlinks, checkboxes, buttons, etc., that make web pages easy to navigate; and everyone knows how to browse and work on it even without any kind of training. Moreover, a skilled user is ready from the beginning to use shortcuts, tabbed navigation and additional features to speed the work;

[4] `http://hea-www.harvard.edu/RD/ds9/`
[5] `iraf.noao.edu`
[6] `paw.web.cern.ch/paw`
[7] `www.astro.princeton.edu/~rhl/sm`
[8] `gnudatalanguage.sourceforge.net`
[9] `http://taos2.asiaa.sinica.edu.tw/`
[10] `http://ipercool.oato.inaf.it/`

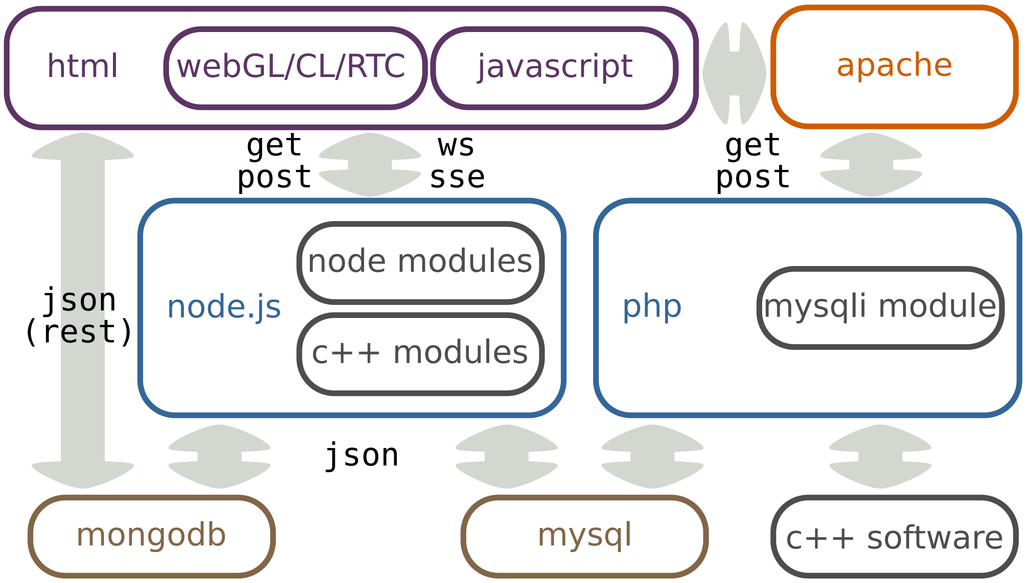

Fig. 1. A modern web application in `html5` using `javascript` and its libraries (`jquery`, `d3.js`), `webGL`, `webCL` and/or `webRTC` solutions, can interact with astronomical database archives (the schemaless `mongodb` or `MySQL`) in different ways. A solution for quick and very basic queries is to use the `mongodb` REST interface, communicating without introducing server-side components. Another modern LAMP way is to use GET/POST requests through an Apache web server to a `php` class extended with the `mysqli` module, which can call, for the more complex tasks, external `C++` software. A cutting-edge implementation for the client-server communication is via WebSockets (ws) or server-sent-events (sse) through a `node.js` server, eventually boosted with custom `C++` modules. `json` (or binary `json`) can be used for data exchange.

modularity: behind the web layer, any kind of technology could in principle be used to manage the various tasks;
customization: it can be easily adapted to specific requirements: from meeting the needs of a color-blind user to the possibility to be configured as a node of a cluster of machines.

Fig. 1 summarizes the possible solutions presented in this work and their components, treated in detail in Sects. 3 and 4.

3. Client-side technologies

Our concept of web application is composed by several modules, or layers. In particular the user layer, that provides interactions in real-time via the interfaces, will use the state-of-the-art real time web technologies to develop these module. All major browsers (Chrome / Chromium derivatives and Firefox derivatives), along with minor brothers (Safari, Opera), are already fully or quasi-fully compatible with these technologies. Here we give a simplified overview:

`html5` is the last revision of the `html` language, offering several new features[11], that we tested for their potential in astronomy. Among them: *server-sent-events and WebSockets*, provide the possibility to obtain real-time events as emitted by the server without a specific request from the client (e.g. with a click, a mouse movement, a polling). In observational astronomy it is for example useful to continuously monitor atmospheric parameters, or to follow the evolution of the light curve during a transient event such as a planetary transit. In general it is useful to perform any kind of real-time control without specifically asking the browser to contact the server to receive these information. *Canvas and SVG* allow to draw graphics

[11] `www.w3.org/TR/html5/`

and display data, on the fly, on a web page. They can be used for displaying FITS images or inline vector graphics for interactive plots and data representation, for example for graphically browse a database, such as in the case of TAOS[12] (Ricci et al., 2014a, submitted). *WebGL, WebCL* allows the rendering of interactive 3D graphics and shaders without the use of browser plug-ins, overriding the limitations of `javascript` as scripting language[13] (for example in slow *for* loops). Parallel computations can take place within the browser, too. Furthermore, thanks to *WebRTC*, direct communications between computers, without a central server or additional software, as well as real web-based peer-to-peer applications are now possible. Check `http://easyrtc.com` or try `http://tawk.com`.

`javascript` : This scripting language (`javascript` is not `java`, which is a software platform based on a virtual machine which runs programs written in its specific object-oriented *programming* language, also named `java`) is very popular to perform client-server calls and manipulation of the web page; typically through general purpose `javascript`-based libraries, like `jquery`, easily expandable with custom plugins, and `d3.js` [14], providing easy to use tools to generate vector `svg` plots. Additionally, a ModelViewController (MVC) design would fit very well the collaborative development of www applications because they are split in components. For this reason we will also consider frameworks like `AngularJS` in addition to `d3.js` and as an alternative to `jquery`.

`json` : Several kind of techniques and formats can be used to exchange data and requests between the browser and a web server (protocols). Typical cases are: plain text, `xml`, `json` (JavaScript Object Notation). The latter is fully integrated, and therefore easily manageable, within `javascript` and `Python`. It is particularly suitable for data-interchange and is often used in AJAX[15] exchanges. Recent browsers also provide native `json` encoding/decoding. It is easily manipulated by `javascript` code and, with respect to `xml`, it is meant to reduce the complexity of the markup providing native support for Data Types such as boolean, strings, numbers, arrays and objects. Its bigger advantage with respect to `xml` is compactness, which can be pushed even further by adopting `BSON`[16], short for Binary `json`, i.e. a binary serialization of `json`-like documents.

Modern webtools for astronomy we aim to build (see e.g. our images archive browser test filter in Fig. 2) will definitely make use of AJAX technology, but they will also go beyond, by using the above mentioned *WebSockets and server-sent-events* (see section 4).

4. Server-side and DBs

If web standards are evolving rapidly, the same occurs for server-side technologies and database management systems (DBMS). Here we present two options: a modern LAMP structure (Linux operating system, Apache http server, a database software, `MySQL` in our case, and a server-language such as `php`) and a completely new approach using only web technologies.

`php` +`mysqli` : LAMP is widely used in astronomy. For some applications requiring specific relational DB capabilities, we suggest the use of `php` and `mysqli`. Such an object-oriented `MySQL` database connection can be, for example, achieved through the use of a library that we implemented: `decibel`[17]. It handles all the aspects related to the database interaction via a class of `mysqli`-based methods to easily communicate with the `MySQL` server. This solution is independent from the upper levels and can provide an interface with any kind of scripting language.

`node.js` + `mongodb` : a new era has just start for what concerns web servers, and the credit comes, again, from the development of web browsers. Thanks to the Free Software

[12] `http://is.gd/ikaqun`
[13] `http://is.gd/efipav`
[14] `http://d3js.org/`
[15] `http://is.gd/qafuwe`
[16] `http://bsonspec.org/`
[17] `http://is.gd/zaniqu`

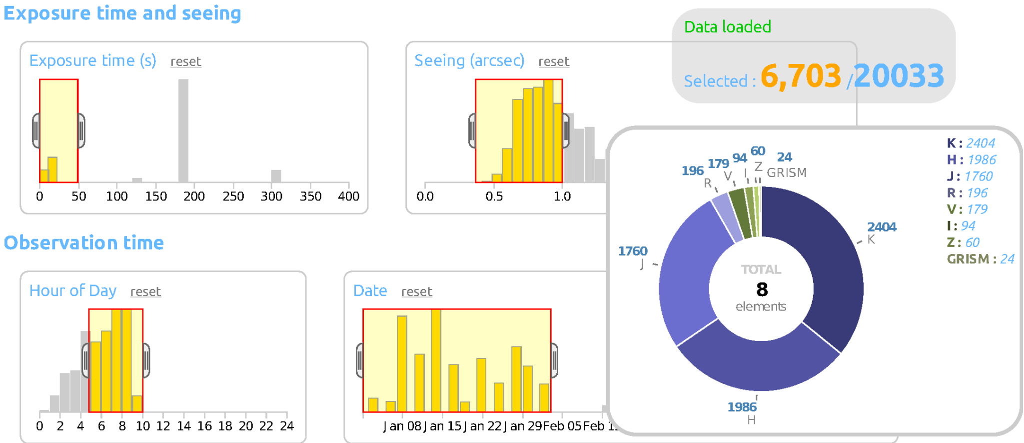

Fig. 2. An example of graphically filtering an astronomical database query with a web application. Here we use the `d3.js` *crossfilter* plugin and `php +mysqli`. See `http://ross.iasfbo.inaf.it/~rossusr/DBcross/` for a live test.

V8[18] engine, developed for the Chrome / Chromium browser, `javascript` execution became so fast to be used also as a powerful *browser-side programming language*, making use of *just in time compilation*[19]. V8 is nowadays used by other independent projects such as `node.js` [20], a fast and highly customizable web server, and `mongodb` [21], a non relational database management system.

4.1. node.js: javascript on server side

Thanks to V8 (`node.js` being written in `C++`), it is possible to extend the `javascript` engine built-in functions and objects with new "add-ons" written in `C++`, dynamically linked to the `node.js` interpreter. To test its capability, we developed the `sadira`[22] framework which makes use of these add-ons to provide `javascript` interfaces toward intensive computing tasks, such as CCD camera drivers and a real-time generator of png tiles[23]. Many other astronomy-oriented tools are being developed within this framework, in particular for to the EU funded project GLORIA[24].

[18] `code.google.com/p/v8`
[19] `http://is.gd/xeweri`
[20] `www.nodejs.org/`
[21] `www.mongodb.org/`
[22] `http://sadira.iasfbo.inaf.it:9999`
[23] `www.github.com/GLORIA-project/node-fits`

4.2. mongodb: a schemaless db in json

`mongodb`, a tree-structured DBMS which uses binary `json` to store data, gives us the flexibility necessary when dealing with content missing a pre-defined data structures. This can be, for example, the case of FITS files, whose content is producer dependent. It allows complex data structures to be created and managed on the fly; something that is impossible to be achieved with traditional relational databases. We propose to pair this technology with relational `MySQL` databases to handle the complex data archives researchers deal with. The GLORIA project archive system already adopts this approach. To notice that `mongodb` also provides a REST interface[25] for direct basic calls.

5. Conclusions

Access to Internet comes from a variety of portable and still powerful devices. Connection

[24] `http://gloria-project.eu/`
[25] `http://docs.mongolab.com/restapi/`

speed and reliability is constantly improving. This was not the case until a few years ago and opens completely new frontiers to the distributed, client-side, peer-to-peer computation, not just data exchange. Nowadays it does not make sense to concentrate the heavy tasks to dedicated servers only. Huge and growing amount of astronomical data are just waiting to be made easily and intuitively accessible. In fact astronomy would particularly benefit from an enlarged participation to data digging by the community at large. Citizen science projects like those developed within Zooniverse[26] show this very clearly. Web applications allow a fast, unlimited and "democratic" exploitation of astronomical databases. Regardless of the fact that you are a professional astronomer or a simple astronomy fan. We believe that web-tools are the key for a new phase in astronomy and science in general. The technologies described in this paper, summarized in Fig. 1 and shown at work in some use cases (e.g. Fig. 2), will help to reach this goal.

Acknowledgements. This work is supported by GLObal Robotic telescopes Intelligent Array for e-Science (GLORIA), a project funded by the European Union Seventh Framework Programme (FP7/2007-2012) under grant agreement number 283783.

References

Ricci, D. & Nicastro, L. 2013, in EAS Publications Series, 61, 263

Ricci, D., Nicastro, L., & Pio, M. 2013, in INTED2013 Proceedings, 7th International Technology, Education and Development Conference, L. Gómez Chova et al. eds. (IATED, Valencia), 2998

Ricci, D., Sprimont, P.-G., Ayala, C., et al. 2014a, in ASTROROB2013 Proceedings (Rev. Mexicana Astron. Astrofis.)

Ricci, D., Sprimont, P.-G., Ayala, C., et al. 2014b, in ASTROROB2013 Proceedings (Rev. Mexicana Astron. Astrofis.)

Sprimont, P.-G., Ricci, D., & Nicastro, L. 2014, in ASTROROB2013 Proceedings (Rev. Mexicana Astron. Astrofis.)

[26] http://zooniverse.org