# A microservice-based portal for X-ray transient and variable sources

**Daniele D'Agostino** [Corresp., 1], **Luca Roverelli** [1], **Gabriele Zereik** [1], **Andrea De Luca** [2], **Ruben Salvaterra** [2], **Andrea Belfiore** [2], **Gianni Lisini** [3], **Giovanni Novara** [4], **Andrea Tiengo** [2, 4, 5]

[1] Institute for Applied Mathematics and Information Technologies "E. Magenes", National Research Council of Italy, Genoa, Italy

[2] Istituto di Astrofisica Spaziale e Fisica Cosmica, National Institute for Astrophysics, Milan, Italy

[3] Institute for Advanced Study, Pavia, Italy

[4] Scuola Universitaria Superiore IUSS Pavia, Pavia, Italy

[5] Sezione di Pavia, National Institute for Nuclear Physics, Pavia, Italy

Corresponding Author: Daniele D'Agostino
Email address: dagostino@ge.imati.cnr.it

Modern soft X-ray observatories can yield unique insights into time domain astrophysics, and a huge amount of information is stored - and largely unexploited - in data archives. Like a treasure-hunt, the EXTraS project is harvesting the hitherto unexplored temporal domain information buried in the serendipitous data collected by the European Photon Imaging Camera (EPIC) instrument onboard the ESA XMM-Newton, in 16 years of observations. Part of this analysis is performed through a dedicated science gateway, the EXTraS portal, whose initial release is the subject of this paper. In particular the focus is on its light software architecture, based on the use of microservices, providing a better resilience and more decoupled development lifecycle with respect to the approaches followed by the most used science gateway toolkits.

# A Microservice-Based Portal for X-ray Transient and Variable Sources

Daniele D'Agostino*, Luca Roverelli*, Gabriele Zereik*, Andrea De Luca†, Ruben Salvaterra†, Andrea Belfiore†, Gianni Lisini‡, Giovanni Novara‡ and Andrea Tiengo‡†§

* CNR-Istituto di Matematica Applicata e Tecnologie Informatiche "Enrico Magenes", via Dei Marini 6, 16149 Genova, Italy

† INAF-Istituto di Astrofisica Spaziale e Fisica Cosmica Milano, via E. Bassini 15, 20133 Milano, Italy

‡ Scuola Universitaria Superiore IUSS Pavia, piazza della Vittoria 15, 27100 Pavia, Italy

§ Istituto Nazionale di Fisica Nucleare, Sezione di Pavia, via A. Bassi 6, I-27100 Pavia, Italy

**Abstract**

Modern soft X-ray observatories can yield unique insights into time domain astrophysics, and a huge amount of information is stored - and largely unexploited - in data archives. Like a treasure-hunt, the EXTraS project is harvesting the hitherto unexplored temporal domain information buried in the serendipitous data collected by the European Photon Imaging Camera (EPIC) instrument onboard the ESA XMM-Newton, in 16 years of observations. Part of this analysis is performed through a dedicated science gateway, the EXTraS portal, whose initial release is the subject of this paper. In particular the focus is on its light software architecture, based on the use of microservices, providing a better resilience and more decoupled development lifecycle with respect to the approaches followed by the most used science gateway toolkits.

**Keywords**

Science Gateways; Microservices; Astrophysics

## I. Introduction

A wide diversity of astrophysical phenomena - from stellar flares in the solar neighborhood to accretion in galactic nuclei at cosmological distances - are characterized by flux and spectral changes on time scales, ranging from a fraction of a second to several years. Current observing facilities produce "time-resolved" images of the sky, with a time resolution of the order of 1 sec. or better [1]. Thus, a huge amount of potentially interesting information is collected each day, but it remains mostly unused, stored in data archives. This is especially true in the high energy range of the electromagnetic spectrum, where source variability is very common but the time dimension is seldom systematically exploited.

The EU-funded FP7 EXTraS (Exploring the X-ray Transient and variable Sky, http://www.extras-fp7.eu) project aims to search and characterize variable sources in the soft X-ray energy range, by exploiting the whole database collected by the European Photon Imaging Camera (EPIC) cameras onboard the ESA's X-ray space observatory XMM-Newton [2]. Sixteen years after its launch, EPIC is still fully operational and its immensely rich archive of data keeps growing. The catalog of serendipitous sources extracted from EPIC observations, dubbed 3XMM, is the largest and most sensitive compilation of X-ray sources ever produced, listing more than 500,000 detections over about 800 square degrees of the sky [3].

EXTraS is going to systematically extend 3XMM by designing and implementing four main lines of analysis. At first, the characterization of the 3XMM sources on all possible time scales (from the duration of an observation to the instrument time resolution, typically 73 ms and 2.6 sec.) will result in a systematic study of aperiodic, short-term variability. The other three analyses have never been performed before on the EPIC database. They are i) a systematic search for short, weak transient sources that are above the detection threshold just for a small interval of time; ii) the variability on longer time scales, thanks to the large number of overlapping observations performed in 16 years; iii) a systematic search for periodicities due to candidate pulsators.

As the most sensitive search for variability ever performed, EXTraS may raise new questions in high-energy astrophysics and serve as a pathfinder for future missions. Therefore EXTraS results, together with new software tools related to the four lines of analysis, will be released to the community at the end of the project. The software is of particular importance for enhancing the potential of discovery of the XMM-Newton mission [4], because it is ongoing and therefore produces daily new data.

There are four possibilities to provide the software to the scientific community. The first, basic solution is to make available an installer or an archive containing all the files required to build the analysis tool. This approach has been adopted for some important tools of the Astrophysics community as the Science Analysis System (SAS), a collection of scripts and libraries specifically designed for the XMM-Newton observations (http://www.cosmos.esa.int/web/xmm-newton/sas-download). A second solution can be to provide the software by exporting the corresponding workflows, that can be thus executed using a Workflow Management System. Also this solution has been adopted by the Astrophysics community [5], [6]. The third solution

is to provide a virtual machine with all the software installed on it. In general software tools may exhibit different levels of maturity: they may be written in a multitude of programming languages, they may depend on specific libraries (sometimes even specific library versions), and on specific execution environments (e.g. Linux or Windows). This solution is probably the most effective for non-expert astronomers, who wants to run a few experiments, and for dissemination purposes, for example for educational programs or citizen scientists. It is worth noting that the SAS are available also as a Linux virtual machine. The fourth solution is to provide the software as a set of services through a Web portal designed following the science gateway paradigm [7].

Science gateways are gaining an increasing interest also in the astrophysics community [8]. In this paper we describe the architecture and the first release of the science gateway (http://portal.extras-fp7.eu) that is under development in the project. Presently it allows the analysis for transient X-ray sources on the whole XMM-Newton Science Archive, containing the data from the XMM-Newton mission. The portal is the result of the joint effort of the two communities of the project, astrophysics and ICT, and the main contribution is represented by the description of the microservice-based approach we are following, that is based on a previous experience in deploying a science gateway for the Hydrometeorological scientific community [9].

The paper is organized as follows. Section II discusses related works, while Section III presents the architecture of the Web portal. Section IV describes the service for the analysis of transient X-ray sources, while Section V concludes.

## II. Related Works

Science gateways can be defined as a set of software, data collections, instrumentations and computational capabilities that are integrated via a Web portal (or a desktop application) in a user friendly and effective environment supporting the scientific research and education activities of a specific community. Each community presents different requirements, due to the software and/or data it shares and the goals it aims to achieve, but normally there is the need to setup services for user management, the deployment of user interfaces (UI) for experiment definition and configuration, with workflow engine and job submission services for orchestrating the experiments execution.

Some of these items can be addressed by general-purpose, ready-to-use solutions (e.g Grid certificates, workflow management systems), some others instead rely on ad-hoc solutions (i.e. the UI) that can be developed using general-purpose technologies. In general, science gateway toolkits as gUSE [10], Apache Airavata [11], Agave [12] and HUBzero [13] are powerful solutions that allow non-ICT users to quickly deploy Web portals by providing a set of enabling technologies, front-end and back-end gateway services. For example gUSE allows submitting jobs through the portlet technology on almost all the distributed computing infrastructures in Europe and US. Airavata can be used as a middleware layer and it provides APIs to submit and monitor jobs from gateways front-end written in several languages (e.g. Java, PHP, Python and C++). Agave focuses on providing a "Science-as-a-Service" solution for hybrid cloud computing, i.e. an enhancement of the "Platform-as-a-Service" paradigm with computational, data, and collaborative services as well. The defining characteristics of HUBzero are the delivery of visual simulation tools, that look like simple Java applets embedded within the browser, and the strong focus on the collaborative aspects for research and educational activities.

However, most of these toolkits are based on time-proved technologies, while it is possible to identify new technologies and different architectural approaches that are well received by the ICT community. Moreover, in designing and maintaining a science gateway there is the need to deal with different aspects related to the continuous improvement of the software used by the reference scientific community, characterized by heterogeneous functional and non-functional requirements, new data repositories, distributed computational resources and other software components. The experience gained in deploying a science gateway for the Hydrometeorological scientific community, the DRIHM portal, [14] led us to investigate alternative solutions. Interested readers can refer to [9] for more details.

## III. The EXTraS Portal

Our approach for developing a science gateway is based on a three-layer architecture made up by a presentation layer, containing the UI, an application layer, exposing a set of microservices, and a foundation layer responsible for interaction with computational infrastructure. In this paper we focus on the UI management and the microservice-based approach, with the goal to have a user-friendly way to define, retrieve and share experiments, plus an effective approach to create customized UI for astrophysical software tools easy to update and maintain.

The architecture of the EXTraS Portal and its main components are shown in Figure 1.

### *PortalTS*

The EXTraS portal has been based on PortalTS, an original Web Portal under development as an independent project at CNR for the refactoring of the DRIHM portal.

PortalTS has been developed in Typescript using the NodeJS and Express frameworks. It is composed by reusable modules and implements standard features available for a Web site (e.g. user management and registration) along with other features (such as a simple API for data persistence) that enables fast development of custom modules.



PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.2519v1 | CC BY 4.0 Open Access | rec: 11 Oct 2016, publ: 11 Oct 2016

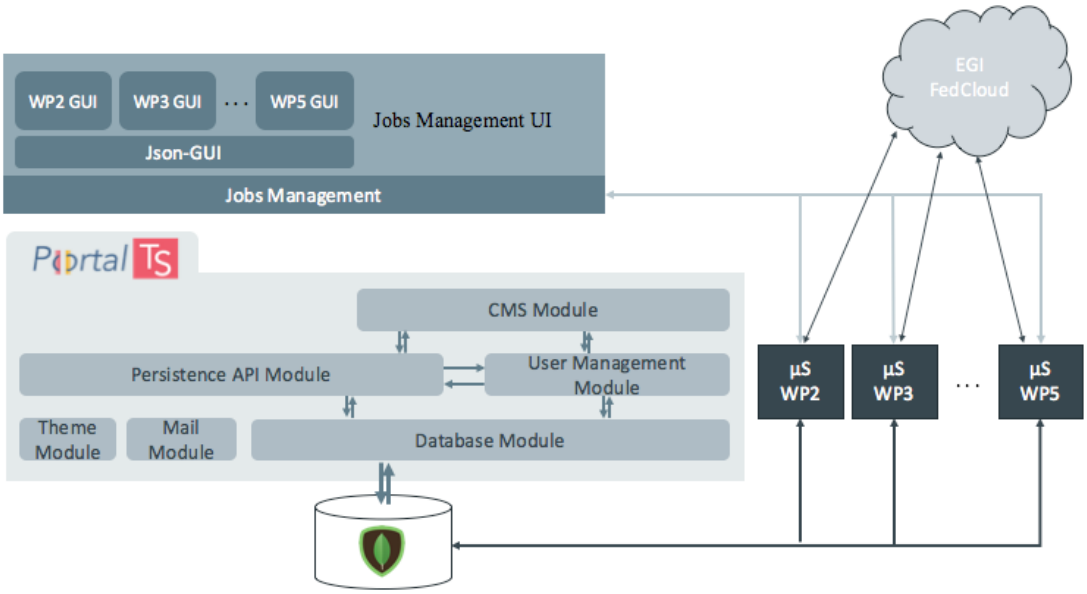


Fig. 1. The architecture of the EXTraS science gateway

A module is a component that implements and exposes a feature, but can also use features exposed by other modules. It is a very general component representing, for example, a set of web pages, a web service, a web app (aka Single Page Application), a set of static files like css files, images, or something different. According to the NodeJS philosophy, each module should be as simple as possible and implement a single functionality.

PortalTS loads modules on the bootstrap phase, using a configuration file to ensure modules loading order. Since PortalTS and any module are developed in Typescript, module error loading caused by typos and other typical Javascript errors (e.g. undefined functions or undefined function arguments) is dramatically reduced, resulting in an improved software reliability and stability.

PortalTS includes some ready-to-use modules, as shown in Figure 1. The Database module defines and manages the connection with the MongoDB database. Despite its simplicity, this module is fundamental and it is used by higher-level modules for the communication with the database. MongoDB has been chosen because it is extremely well integrated with NodeJS. Moreover, there are some high level libraries, e.g. mongoose (http://mongoosejs.com/), that are stable and maintained, making them usable in a production environment.

The User Management module defines an API for a complete authentication system, including user registration, login, and administration pages. It implements also role and group concepts, at the basis of the authorization mechanism for pages, modules and entities.

The Persistence API Module defines the interface to store, retrieve and manage heterogeneous data on the MongoDB database. It exposes both a RESTful and an internal API, that can be directly used by other modules, as the CMS described below. The RESTful API is very important since it allows to store data directly from a web app, that can be built upon this modules. One example is the Jobs Management, described below. The Persistence API defines entities and collections. A collection is a set of entities, while an entity represents possibly heterogeneous data stored with some additional metadata, like creation, update time or the owner. An entity can only belong to one collection.

The Persistence API relies on the User Management Module to ensure security and user authentication on the data. By default, an entity is only accessible by the owner, but its read and write access policy can be changed, using a group-based policy. The Persistence API Module is fundamental since it allows to store and retrieve persistence data without any effort, enabling a ready-to-use persistence layer. Moreover, this layer is integrated with an AngularJS library that implements all the methods necessary to give a quicker and simpler access to the persistent data.

The CMS (Content Management System) module defines some web pages for user login and registration, and it allows the creation of user-defined web pages and a menu. Each web page or menu element can be publicly available or accessible by a particular group, since the CMS module uses the Persistence API module. There are also some further basic modules, like the Theme module, that defines the web pages header and the footer. This module can be exploited by all portal modules to define a standard look and feel of a portal instance.



PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.2519v1 | CC BY 4.0 Open Access | rec: 11 Oct 2016, publ: 11 Oct 2016

### *Json-GUI*

Json-GUI is a front-end library, developed as a set of reusable AngularJS directive, that allows the dynamic generation of full-featured form-based web interfaces including validation and constraints. It can considered a companion tool with respect to PortalTS, but it can be exploited in any AngularJS/Javascript web application.

The choice of building such a module from scratch, against using already existing ones, is derived from the need to address some specific requirements. In particular, the way the module is built lets any non-IT scientist to easily design an advanced configuration interface, freeing the IT developer from editing the source code any time a scientist decides to update the parameter list or any other interface element. In fact, Json-Gui comes with some high-level features well suited for the scientific context, that gives scientists the possibility to easily define high level validation and constraints between configuration parameters.

### *Jobs Management*

The Jobs Management Module is a custom module specifically developed for the EXTraS portal. It provides user the possibility to create, submit and manage the different analysis experiments defined in the project, as described in the Introduction.

This module is based on AngularJS and it is a complete web app, without any server side code. It uses the Persistence API to store and retrieve experiments data, and it directly communicates with microservices, which are responsible to execute the different analysis software. Each analysis tool is based on a set of specific configuration parameters: the corresponding UI has been created using the Json-GUI library. Jobs executions are performed remotely on available computing Infrastructure as EGI FedCloud (https://www.egi.eu/infrastructure/cloud/) or High Performance Computing facilities.

The Jobs Management Module provides two further key features: the ability to share a job and the support for the interaction and discussion (in terms of comments) among the scientists sharing it. Sharing a job means not only that the experiment results are visible to other users, but also the configuration is shared and can be used as a starting point for re-submit the experiment on a new set of data. Thus, a job execution can be replicated by other users that can, for example, validate the experiment results or explore the behavior by changing one or a few parameter values.

EXTraS deems outreach and involvement with schools and other institutions a fundamental component of its scientific mandate. These features can be relevant to provide students with the possibility to play with some scenarios. The sharing implementation is extremely simple thanks to the Persistence API Module: it is sufficient, in fact, to modify the access policy of a job to share it with one or more groups.

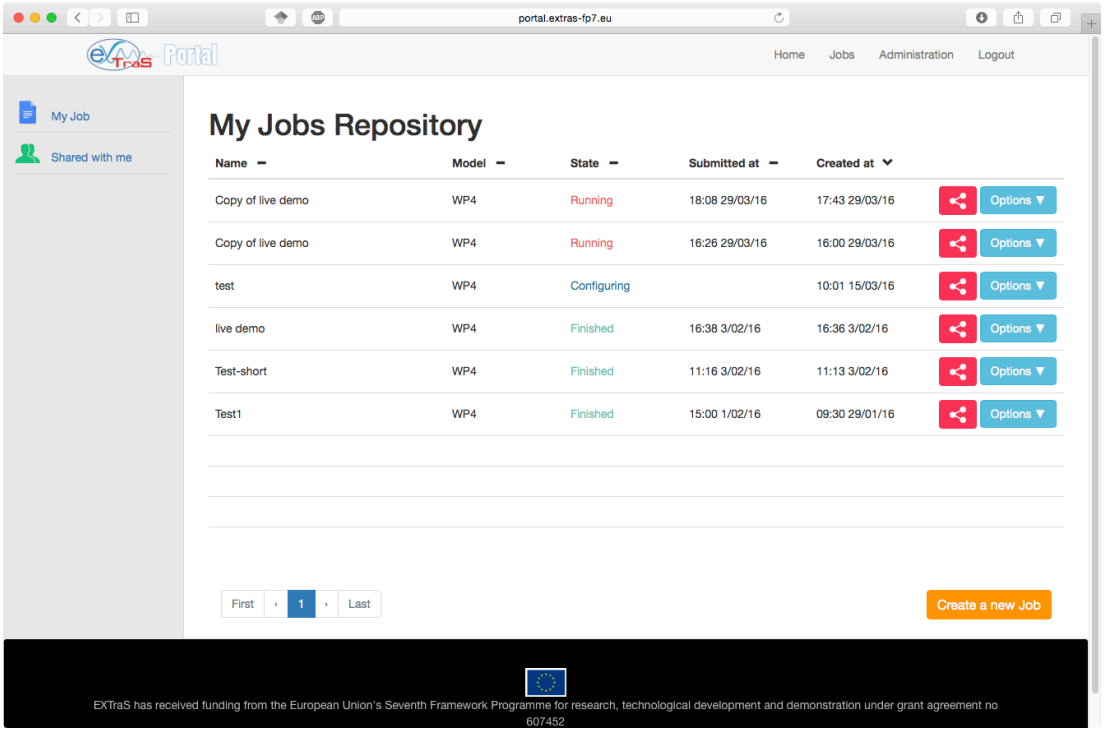


Fig. 2. The Jobs Management UI



PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.2519v1 | CC BY 4.0 Open Access | rec: 11 Oct 2016, publ: 11 Oct 2016

*Microservices*

A microservice is a self-contained reusable component that fulfills a specific task. They are loosely coupled components that can be independently updated and provide better scalability with respect to other solutions [15]. For example, if a single microservice of a portal is accessed by many more users than the others, thus it needs to manage many more connections and activities, it is possible to instantiate it multiple times, performing load balancing without wasting resources, thus scaling at very low granularity. On the contrary, with a traditional single Web Service approach, the whole Web Service requires to be instantiated multiple times, increasing the consumption of partially exploited resources.

Moreover it is possible to update, improve or correct bugs of each microservice independently from the others. This enables a faster development cycle and simpler analysis, debug and deployment phases for each microservice, improving the maintenance of the whole system and enabling a fast prototyping on new functionalities.

In the EXTraS Portal, each analysis tool corresponds to a specific microservice implementation, which exposes two different APIs. The first one is used by the Jobs Management Module, to submit and monitor the experiments. The other one supports the communication with the computing resources used to run the experiments, in order to provide a nearly real-time feedback log to the user.

In the current release of the EXTraS portal, only one analysis operation has been made available: the software tools corresponding to the remaining ones are still beta versions, and they will be integrated as soon as they will be made available. This is the reason why only one microservice has been deployed. Since the present infrastructure is quite simple, we manage it with the the PM2 tool (http://pm2.keymetrics.io/), a production process manager for Node.js applications with a built-in load balancer that allows to monitor and auto-restart the instances of PortalTS and the microservice.

Of course, we know that this solution does not scale up with a number of microservices and more complex infrastructures. Moreover, it is important to mention that, beside the microservice and PortalTS, it is also necessary to manage and monitor a MongoDB instance, a Redis server (http://redis.io) necessary to manage user sessions on PortalTS, and an Apache HTTPD web server, used to forward the different requests to the microservice and PortalTS. For this reasons, we are considering several solutions that allow an easy management of the entire infrastructure, able to optimize the workload distribution of the coming different microservices, PortalTS, MongoDB and Redis as well. The first step will be to switch to a containerized environment, like Docker (https://www.docker.com/), that will improve the management and deploying of PortalTS and the different microservices, plus the MongoDB and Redis instances. However, Docker by itself does not provide an automatic scaling of the number of instances of the microservices and PortalTS, based on traffic loads or CPU usage, and a distribution of the instance among the nodes of a cluster. Therefore, it is necessary to adopt further systems like Kubernetes (http://kubernetes.io/) that support the management, deployment and execution of different containers on a cluster environment, also providing an automatic scaling and distribution of the containers.

## IV. The Analysis of Highly Variable and Transient Sources

The analysis of highly variable and transient sources aims at identifying burst-like variability during EPIC observations. It is based on standard source detection algorithm, that are applied to time-resolved images derived from the XMM-Newton observations. Images are analyzed to identify new point sources that might have brightened considering different energy bands.

Each observation can result in a single image or a set of images can be obtained by subdividing the observation into sub-exposures corresponding to different time intervals. The time intervals can either have fixed duration or they can be defined with a preliminary search for an excess of counts within a small region of the detector in limited time periods. This step of the analysis is performed using a Bayesian Blocks algorithm on the events detected in partially overlapping regions having a size comparable to the characteristic dimension of the telescope point spread function. The main parameters for the transient analysis on one or more observations, named hereafter the experiment, are the choice of instruments (i.e. one or both the EPIC instruments), the energy bands, the list of observation identifier(s) and the possible time intervals for source detection (single image, fixed bins, Bayesian blocks analysis, or all three).

The EXTraS portal provides users with the possibility to submit experiments for all the observation data contained in the ESA XSA Archive. After the login, a user interacts with the Jobs Management UI for managing experiments, as shown in Figure 2. One experiment consists in a (partially) configured analysis specification, a running job or the result achieved with a specific analysis configuration. An experiment can have been created by the user or shared with her/him by a member of her/his group. To create an experiment the user selects the transient analysis and a name for it. Then she/he can configure it and decide whether to submit it or to only save the defined parameter values. An experiment can also be created starting for an existing one: this feature, besides supporting the reproducibility of the results, has been designed to simplify and speed up the execution of an experiment by changing some parameter values or the considered observations, e.g. when new data are available. When an experiment correctly completes its execution, the user can download the output and the logs files. These data can be shared within the groups she/he belongs to. In this last case scientists can cooperate among them in evaluating the results by providing comments. The UI allows to apply the same analysis configuration to several observations. This feature



PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.2519v1 | CC BY 4.0 Open Access | rec: 11 Oct 2016, publ: 11 Oct 2016

results in the creation of multiple experiments, one for each observation. This choice allows to submit multiple analyses at a time but to maintain a one-to-one correspondency between one observation and its results.

In details, the EXTraS science gateway relies on the Federated Cloud infrastructure provided by EGI to run the experiments. In particular a virtual appliance (https://appdb.egi.eu/store/vappliance/extras.wp4) (i.e. the template for virtual machine - VM -instances), containing the transient analysis software an a contextualization mechanism based on cloud-init (https://appdb.egi.eu/store/software/fedcloud.cloud.init) has been registered to the EGI Applications Database (AppDB), in order to be able to instantiate it on the Federated Cloud sites supporting the project. EGI in fact relies on a single sign-on mechanism to access the federated services based on X.509 certificates and Virtual Organization (VO) membership. In our case the portal exploits a robot certificate. They were introduced to allow users, who cannot get or are not familiar with personal certificates, to exploit any distributed infrastructure relying on them in their research activities [16]. The robot certificate is usually associated with a specific application (or function) that the application developer/provider wants to share with all the VO members. This is exactly the scenario arisen in the EXTraS activities, because portal users are provided with the possibility to run only pre-defined software tools.

The experiment configuration is performed by interacting with the form created for the transient analysis by using Json-GUI. When the user saves an experiment the resulting parameter list is stored on the portal mongoDB database by interacting with the Persistence API module. The selection of the observation(s) and the actual submission is performed through the Jobs Management UI, that forward the request to the microservice associated with this kind of analysis. The microservice a) interacts with the Persistence API to retrieve the parameter list; b) selects (presently in a round-robin way) one of the Federated Cloud site supporting the *extras-fp7.eu* VO; c) creates the contextualization file for the analysis; d) activates the VM instance(s) via a rOCCIi client (https://github.com/EGI-FCTF/rOCCI-cli) installed on the server hosting the EXTraS portal; e) creates one record for the job on the database with the status "submitted".

The VM then starts its execution by downloading, via wget, the observation data from the XSA archive: the XSA Archive Inter Operability system (AIO) allows a direct access to the XSA data. Furthermore, it periodically notify via POST requests the microservices about its progress, in order to update its status (e.g. submitted_error, running, finished, error) in the associated record. It can also provides the log files in realtime to the user, The job execution can be monitored in fact via the Jobs Management UI, who retrieve the status from the database and the logs from the microservice. At the end of the analysis the VM notifies the microservice who a) transfer the resulting files on the server filesystem; b) update the record on the portal database; c) shut down the virtual machine using the rOCCI client. All the information related to this job (e.g. the configuration parameters, the logs, the results, the ownership/sharing information and possible comments) are stored on the portal database until it is deleted by the user who submitted it.

## V. Conclusions and Future Works

This paper presented the first release of the EXTraS portal, a science gateway for the astrophysics community devoted to the search and characterization of variable sources in the soft X-ray energy range by exploiting the XMM-Newton observations.

The portal relies on recent general-purpose technologies, architectural patterns and best practices adopted in the development of enterprise web application. In general, the decoupling of the presentation, application and foundation levels of a gateway and, whenever possible, the split of services in smaller, more manageable components give science gateway developers more control and a smoother software development lifecycle. The EXTraS portal has been validated and it is currently used for the project's activities related to the search for transient X-ray sources.

The integration of the other analysis tools currently under development in the project represents the future direction. The final, public release is expected at the end of the project in December 2016. We will consider also the possibility to provide a remote visualization service, for a better exploitation of experiment results. This feature will be of particular importance for outreach and students involvement activities of the project and, in perspective, to provide the portal to the large community of citizen scientists interested in the astrophysics research activities.

## Acknowledgment

EXTraS has received funding from the European Union's 7th Framework Programme for research, technological development and demonstration under grant agreement no. 607452.



PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.2519v1 | CC BY 4.0 Open Access | rec: 11 Oct 2016, publ: 11 Oct 2016

## REFERENCES


[1] C.R. Kitchin, Astrophysical techniques, CRC Press, 2013.

[2] A. De Luca, R. Salvaterra, A. Tiengo, D. D'Agostino, M.G. Watson, F. Haberl, J.Wilms, An overview of the EXTraS project: Exploring the X-ray Transient and Variable Sky, PoS(SWIFT 10)135, 2014.

[3] S. R. Rosen, N. A. Webb, M. G. Watson, J. Ballet, D. Barret, V. Braito, F. J. Carrera, M. T. Ceballos, M. Coriat, R. Della Ceca et al., The XMM-Newton serendipitous survey. VII. The third XMM-Newton serendipitous source catalogue, Astronomy & Astrophysics, in press. DOI: 10.1051/0004-6361/201526416

[4] Mushotzky R., What do Astronomers Really Want? Astronomical Data Analysis Software and Systems (ADASS XX) , ASP Conf. Proc., 442, p.235, 2011.

[5] J. Ruiz, J. Garrido, J. Santander-Vela, S. Sanchez-Exposito, L. Verdes-Montenegro Astrotavernabuilding workflows with virtual observatory services Astronomy and Computing, 78, pp. 3-11, 2014.

[6] Castelli, G., Taffoni, G., Sciacca, E., Becciani, U., Costa, A., Krokos, M., Pasian F, Vuerli, C., VO-compliant workflows and science gateways, Astronomy and Computing, 11, pp. 102-108, 2015.

[7] Kacsuk P (ed), Science Gateways for Distributed Computing Infrastructures, ISBN 978-3-319-11268-8, 2014.

[8] Becciani U., Sciacca E., Costa A., Massimino P., Pistagna C., Riggi S., Vitello F., Petta C., Bandieramonte M. and Krokos M. (2015), Science gateway technologies for the astrophysics community, Concurrency and Computation: Practice and Experience, 27, pp. 306-327, 2015.

[9] D. D'Agostino, E. Danovaro, A. Clematis, L. Roverelli, G. Zereik, A. Parodi, A. Galizia, Lessons learned implementing a science gateway for hydro-meteorological research. Concurrency and Computation: Practice and Experience, 8(7), pp. 2014-2023, 2016.

[10] Balasko Á, Farkas Z, Kacsuk P. Building Science Gateway by Utilizing the Generic WS-PGRADE/gUSE Workflow System. Computer Science, 14(2), pp. 307-325; 2013.

[11] Marlon E. Pierce, Suresh Marru, Lahiru Gunathilake, Don Kushan Wijeratne, Raminder Singh, Chathuri Wimalasena, Shameera Ratnayaka and Sudhakar Pamidighantam, Apache Airavata: design and directions of a science gateway framework. Concurrency Computat.: Pract. Exper, 27, pp. 4282-4291, 2015.

[12] Dooley, Rion, et al. Software-as-a-Service: The iPlant Foundation API, 5th IEEE Workshop on Many-Task Computing on Grids and Supercomputers (MTAGS), IEEE, 2012.

[13] M. McLennan, R. Kennell, HUBzero: A Platform for Dissemination and Collaboration in Computational Science and Engineering, Computing in Science and Engineering, 12(2), pp. 48-52, 2010.

[14] D'Agostino D, Clematis A, Galizia A et al. The DRIHM Project: A Flexible Approach to Integrate HPC, Grid and Cloud Resources for Hydro-Meteorological Research. Proceedings of the International Conference For High Performance Computing, Networking, Storage and Analysis 2014 (SC14), pp. 536-546, 2014.

[15] Newman, Sam, Building Microservices, O'Reilly Media, 2015.

[16] EUGridPMA, Guideline on IGTF Approved Robots, OID 1.2.840.113612.5.4.1.1.1.6, version 1.2 https://www.eugridpma.org/guidelines/robot/approved-robots-20140908.pdf




PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.2519v1 | CC BY 4.0 Open Access | rec: 11 Oct 2016, publ: 11 Oct 2016