



HAL
open science

ASON: An OWL-S based ontology for astrophysical services

Thierry Louge, Mohamed Hedi Karray, Bernard Archimède, Jürgen Knödlseder

► **To cite this version:**

Thierry Louge, Mohamed Hedi Karray, Bernard Archimède, Jürgen Knödlseder. ASON: An OWL-S based ontology for astrophysical services. *Astronomy and Computing*, 2018, 24, pp.1-16. 10.1016/j.ascom.2018.05.001 . hal-01945190

HAL Id: hal-01945190

<https://hal.science/hal-01945190>

Submitted on 5 Dec 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Open Archive Toulouse Archive Ouverte

OATAO is an open access repository that collects the work of Toulouse researchers and makes it freely available over the web where possible

This is an author's version published in: <http://oatao.univ-toulouse.fr/20078>

To cite this version:

Louge, Thierry  and Karray, Mohamed Hedi  and Archimède, Bernard  and Knödlesder, Jurgen *ASON: An OWL-S based ontology for astrophysical services*. (2018) *Astronomy and Computing*, 24. 1-16. ISSN 2213-1337

Any correspondence concerning this service should be sent to the repository administrator: tech-oatao@listes-diff.inp-toulouse.fr

ASON: An OWL-S based ontology for astrophysical services

T. Louge^{a,*}, M.H. Karray^b, B. Archimède^b, J. Knödlseeder^c

^a Université de Toulouse - UPS, Institut de Recherche en Astrophysique et Planétologie, 57 Avenue d'Azereix, 65000 Tarbes, France

^b Université de Toulouse, Ecole Nationale d'Ingénieurs de Tarbes ENIT, 47 avenue d'Azereix, BP1629, 65016 Tarbes Cedex, France

^c Université de Toulouse - UPS, Institut de Recherche en Astrophysique et Planétologie, 9, avenue du Colonel Roche BP 44346, 31028 Toulouse Cedex 4, France

A B S T R A C T

Modern astrophysics heavily relies on Web services to expose most of the data coming from many different instruments and researches worldwide. The virtual observatory (VO) has been designed to allow scientists to locate, retrieve and analyze useful information among those heterogeneous data. The use of ontologies has been studied in the VO context for astrophysical concerns like object types or astrophysical services subjects. On the operative point of view, ontological description of astrophysical services for interoperability and querying still has to be considered. In this paper, we design a global ontology (Astrophysical Services ONtology, ASON) based on web Ontology Language for Services (OWL-S) to enhance existing astrophysical services description. By expressing together VO specific and non-VO specific services design, it will improve the automation of services queries and allow automatic composition of heterogeneous astrophysical services.

1. Introduction

The virtual observatory for astrophysics is a coordinated effort involving worldwide institutes, astronomers and engineers. It has been created in the early 2000s in front of the growing amount of data generated by modern science, in order to share and understand heterogeneous astrophysical data.

VO may be defined as an “ecosystem of mutually compatible datasets, resources, services, and software tools which use a common set of technologies and a common set of standards”.¹ Different types of VOs co-exist with their own data descriptions, protocols and dedicated tools (International Virtual Observatory Alliance IVOA,² HELIO (Bentley et al., 2013), Virtual Atomic and Molecular Data Center VAMDC (Dubernet et al., 2016) among others). Each one of them offers solutions to maximize the availability of data in a specific astrophysical context. Today, thousands of services are registered within those ecosystems.

Hence, seeking services for one's specific needs and sorting out the multiple services available remains a very complex task. The existing methods for locating services either involve registries query or built-in software interfaces (e.g. offered by Aladin³ or

Topcat⁴). In order to efficiently manage his data, a user has to learn the use of the various VO tools available. Many of those software are listed on a dedicated webpage,⁵ but an automatic selection of the most appropriate tool suitable for handling or retrieving some specific data, or to conduct a specific analysis is not included in the VO technical specifications. Besides, functionalities included in the various tools are not exhaustively listed. This is why the IVOA organizes meetings during which scientists can learn to use VO tools for handling their data. This mandatory step of learning new tools with unclear functionalities is one of the most frequently encountered reasons concerning the abandonment of the use of VO tools.

Moreover, technical details for developers outside of the core of VO community are not easily identified. As a consequence, using VO services and tools in a non-VO framework, software or process is tricky. Choosing which VO protocol to use, how to switch from one protocol to another and how to parse the results are not easy tasks for a non-VO related developer.

As a consequence, this is a key challenge for VO services to be able to describe the data they can provide and the functionalities they can achieve as precisely as possible. Another key challenge is to describe how VO services may interoperate (i.e. how they may be called and their results interpreted) together but also with non-VO services, inside and also outside of VO-dedicated applications.

* Corresponding author.

E-mail address: tlouge@irap.omp.eu (T. Louge).

¹ http://www.ivoa.net/deployers/intro_to_vo_concepts.html.

² <http://www.ivoa.net/>.

³ <http://aladin.u-strasbg.fr/>.

⁴ <http://www.star.bris.ac.uk/~mbt/topcat/>.

⁵ <http://www.euro-vo.org/?q=science/software>.

This work contributes in addressing those two challenges by exploring the possibilities offered by semantic representation of knowledge through the use of ontologies. Ontologies form the key-stone of semantic interoperability, such as in the federated interoperability approach from INTEROP network of excellence (Chen, 2016). In computer science, ontologies can be defined as “explicit specification of a shared conceptualization” (Gruber, 1993). In this work, we study the available ontologies for describing services. Two main specifications, namely Web Ontology Language for Services (OWL-S⁶) and Web Services Modeling Ontology (WSMO⁷) are widely discussed (Lara et al., 2004) in ontological representation of Web services. We will show how OWL-S may be oriented to fit the needs for astrophysical services, VO and non-VO alike.

The aim of our work is to use the resulting ontology (Astrophysical Services ONtology, ASON) as the reasoning base inside automatic workflows composition software based on description of scientific use cases. Therefore ASON should not only describe the services, but integrate them into a more global environment in which the descriptions of scientific requirements can be understood.

It must allow retrieving a set of services based on their capacities and give a fine description of data and general observational context (instrument characteristics...). Its task will be to help the discovery and composition of relevant astrophysical services for answering a user’s request.

ASON is at the core of “Composing Automatically and Semantically Astrophysical Services” (CASAS) (Louge et al., 2017) application available online,⁸ and may be reused and enriched in other ontology-based applications by adapting QuerySoftware (discussed in Section 3.2 - Documentation) to suit specific design.

In Section 2 of this paper, we discuss the state-of-the-art on ontological description of Web services as well as an illustration of IVOA architecture. We identify the issues that need improvements in available Web services ontological descriptions in order to be able to efficiently describe astrophysical services.

In Section 3, we expose the methodology we used to develop a formal ontology for astrophysical services. We explore the construction of this ontology and the definition of its structure. Conclusion and future works are exposed in Section 4.

2. State of the art

The definition of ontologies in computer science cited above references a shared conceptualization. That implies that different sources of knowledge have to agree on a common conceptualization of their domain. The “explicit specification” part of the definition states that this conceptualization has to be expressed in an unambiguous manner. A concrete way of describing ontologies in computer science is given by Ehrig and Staab (2004), an ontology being a set of concepts linked through a subsumption hierarchy (classes and subclasses). Individuals are instances of concepts. Relations that also come with a hierarchy exist between concepts and/or individuals.

Ontologies can be seen as an oriented graph with nodes (the concepts, or classes) and edges being the relations between concepts. Individuals are instances of concepts, and concepts may be annotated to enrich their definition. Tools like Pellet (Sirin et al., 2007), working on this structure may enrich ontologies with unexpressed relationships and subsumptions logically deriving from the knowledge that they contain. The most widely used language to define and use ontologies is OWL (Ontology Web Language) and

its OWL-2 version. Additional languages, like Semantic Web Rules Language (SWRL) extend OWL and allow the use of logical rules (Horn-like rules) (Horn, 1951) with OWL.

The use of ontologies as a way of representing knowledge has become usual in the information sciences, particularly because of the role that ontologies play within the Semantic Web (Berners-Lee et al., 2001). The Semantic Web is an ongoing evolution of the existing Web towards the use of reasoning algorithms for automatic information retrieval. In this evolution, regular technologies like Web Services Description Language (WSDL), used to describe the functionalities and technical details of Web services are replaced by the use of ontologies.

Since ontologies have encountered a great success in representing knowledge in different fields, different categories of ontologies have appeared that mainly differ in their targeted use and genericity. Examples of ontologies categories include:

- Domain ontologies, frequently used in “intelligent” systems (Rashmi and Krishnan, 2017). Domain ontologies contain a representation concerning a specific domain of use (e.g. medicine, industrial maintenance).
- Task ontologies also describe a domain-dependent knowledge, but from a “task” point of view. Those ontologies focus on how a specific task is achieved (e.g. the sub-tasks it includes...) (Martins and De Almeida Falbo, 2008).
- Upper ontologies intent do describe generic concepts that may be encountered in various domain of knowledge (Mascardi et al., 2007). Upper ontologies are frequently used to find mappings between different ontologies, through an upper level of abstraction. Services ontologies are a specific case of upper ontologies that describe Web services behavior and architecture.

Several ontologies can be found in literature to describe Web services, among which OWL-S and WSMO are the most widely discussed. OWL-S and WSMO are upper-level ontologies that do not describe any knowledge but a generic description of Web services. As a consequence, describing a specific set of services using OWL-S or WSMO needs an important work of specialization. This work needs to be done both for services management (ensuring that the service’s operation is correctly described) and domain description (ensuring that the service’s input and output semantic is correctly described).

When dealing with services ontologies, several definitions and concepts are frequently encountered, which include:

- Preconditions, which are some conditions to be met before a service may be used (e.g. when dealing with financial operations a payment may be made only if a bank account is sufficiently stocked).
- Effects, which are the consequences of a service call (e.g. the command is validated)
- Inputs, which are information that must be provided to a service when this service is called (e.g. dealing with astrophysics, if a service is called for observations for a given object, then the coordinates of this object may be inputs for the service)
- Outputs, which are information provided by the service.

Schematically, an output (e.g. a measurement) has an effect (e.g. the measurement is known) and inputs (i.e. information needed for a service to run) are used by a service only when the preconditions are met (e.g. every input information needed is known).

⁶ <http://www.w3.org/Submission/OWL-S/>.

⁷ <http://www.w3.org/Submission/WSMO/>.

⁸ <http://cta1.bagn.obs-mip.fr>.

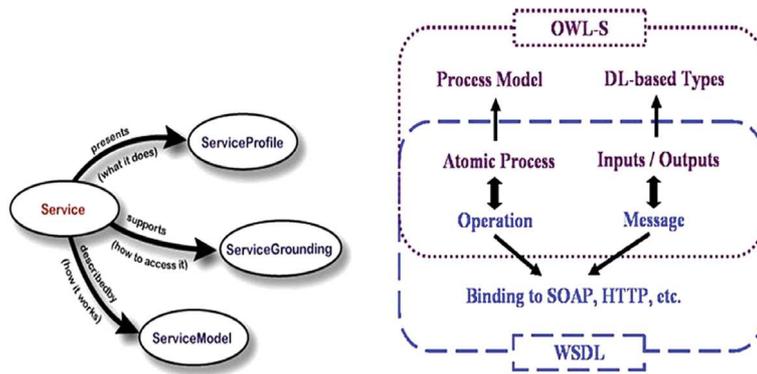


Fig. 1. OWL-S general architecture and relations with WSDL.

2.1. OWL-S

OWL-S (Martin et al., 2007) stands for OWL-Services. It defines an ontology for Web services descriptions, dividing the aspects of a Web service into different sub-ontologies. Details concerning services are divided into three sub-ontologies, each focusing on a specific aspect among:

- what a service does (service profile, or generic description)
- how to access it (service grounding)
- how it works (service process, also sometimes called “service model”).

OWL-S often references Web Services Definition Language (WSDL⁹) as the language for services definition. Fig. 1 exposes the general architecture of this ontology as found in OWL-S submission.¹⁰

OWL-S models services as processes, divided into two categories (a third category, composite processes exist in the submission that represents a composition of services):

- Simple processes describe simple services corresponding to a single query followed by an answer
- Complex processes describe services implying a multi-message dialog between the client and the service.

The text of the submission states that many elements of the ontology have been designed to take those complex processes into account. It is important to note that OWL-S separates outputs from effects, and inputs from preconditions.

OWL-S describes services as processes, using logical expressions to describe preconditions, effects and results. Despite the fact that OWL-S, especially in the grounding concepts for a service often refers to WSDL/SOAP architecture, the concepts themselves stay relevant no matter the technical implementation and description of services.

Nevertheless, it appears difficult to match OWL-S grounding concepts for non-WSDL services as stated in a 2015 paper (Roman et al., 2015).

From the submission and provided theoretical examples given, a concrete way of using OWL-S needs the creation of three ontologies: Profile, Grounding and Process for every service and registering services in the top-level Service ontology.

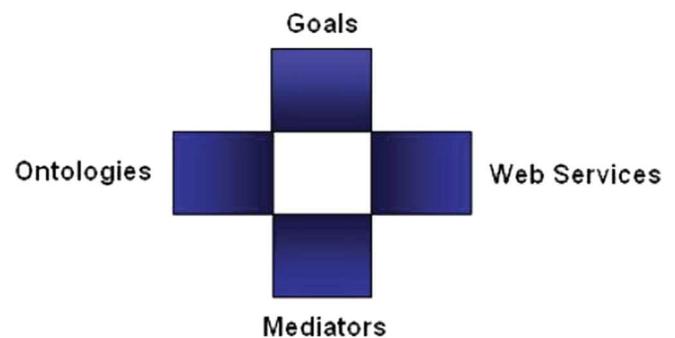


Fig. 2. Top-level elements of WSMO.

2.2. WSMO

WSMO¹¹ pursues the same goal as OWL-S, but with a very different organization. Fig. 2 shows the top-level elements of WSMO.

Every top-level element in WSMO describes a specific aspect of Web services. Like OWL-S, WSMO is an upper-level ontology. This means that Web services specific domain of knowledge needs to be described, on a case by case basis. Common understanding of this domain knowledge must be ensured for every service described. A top-level elements overview for WSMO consists in:

- “Ontologies” describe the domain knowledge that registered services deal with.
- “Goals” are very clearly defined in the submission itself: “Goals are representations of an objective for which fulfillment is sought through the execution of a Web service”.¹²
- “Mediators” are useful when different ontologies describe concepts from the same domain knowledge. They ensure the matchings and mappings between those different ontologies.
- “Web Services” contain the core descriptions of Web services in WSMO. As for Goals, we can refer to the description given in the submission: “WSMO Web service descriptions consist of functional, non-functional and the behavioral aspects of a Web service” (see footnote 12).

⁹ <https://www.w3.org/TR/wsdl20-primer/>.

¹⁰ <http://www.w3.org/Submission/OWL-S/>.

¹¹ <http://www.w3.org/Submission/WSMO/>.

¹² <http://www.w3.org/Submission/WSMO/>.

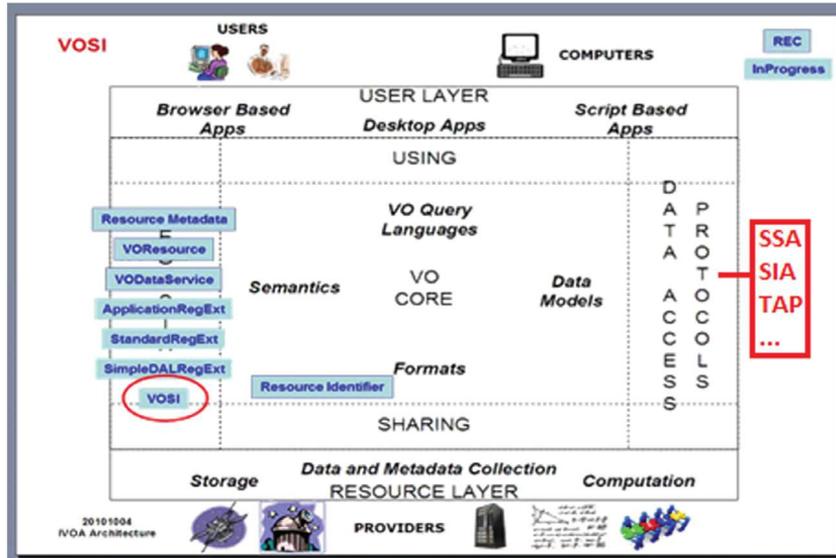


Fig. 3. Virtual observatory supported interfaces (VOSI) and general IVOA architecture.

WSMO focuses on choreography (the dialog between the user and the service) just like OWL-S does, but adds orchestration in an explicit way, stating “how the Web service makes use of other Web services in order to achieve the required functionality” (quoted from the submission).

WSMO is a complete, but rather complex way of describing services. Its exhaustivity implies a division into several ontologies layers and the use of multiple logical expressions inside the different levels of descriptions to ensure their consistency. These logical expressions must be built upon a specific set of vocabulary and terms forming an embedded language inside WSMO, called Web Service Modelling Language (WSML).¹³

Roman and co-authors (Roman et al., 2015) focus on a minimal service model, guided by the need to unify RESTful and WSDL-compliant services around common concepts in a single service modeling ontology.

This model called WSMO-Lite has been a W3C submission since 2010, and derives from WSMO from which it extracts and reorganizes a subset in order to support real-world challenges in intelligent service integration.¹⁴ Goals and mediators are not present in this model that aims at providing lightweight service ontology.

Since WSMO-Lite is built from WSMO base that is heavily related to WSDL, authors expose in the same paper hRESTS mechanism that allows adding semantic annotations directly in HTML pages for RESTful services.

2.3. Overview of astrophysical VO architecture

Concerning astrophysics, interoperability which is the core concern of VO is reached through definite descriptive fields and software tools, able to understand the VO formats, data models and protocols.

2.3.1. Virtual observatory support interfaces (VOSI)

IVOA architecture shown in Fig. 3 (Dowler et al., 2014) is built upon several layers. The diagram in Fig. 3 is organized in blocks. The upper block comprising users and computers (which represent manual or automatic use of the VO) interacts with “USER LAYER”.

This user layer comprises the various means of access to the VO, whether by Web services, dedicated software or scripts. The lower block presents the Providers who interact with the “RESOURCE LAYER” to make their data available.

The service registries are shown on the left block of the diagram, the service access protocols (how to use the services) in the block on the right. Simple Spectrum Access (SSA), Simple Image Access (SIA) and Table Access Protocol (TAP) are examples of protocols available in IVOA specifications.

The central block concerns the elements supporting this architecture, such as data models and semantics.

IVOA data models are schemas for describing metadata associated with observational or theoretical data. These patterns may change over time depending on the feedback from service providers and their specific use cases. The distinction between data model and access protocol is sometimes blurred. Thus, Simple Spectrum Access protocol (SSA) is both a protocol and a data model.

Two elements defined by the IVOA identify the role and the semantics of the elements in a model and the astrophysical quantities present in the data: These are UTypes¹⁵ and Unified Content Descriptors (UCDs).¹⁶

IVOA-compliant software¹⁷ designates any software able to query some of those protocols and understand some of those data models.

2.3.2. The role of UCDs

UCDs express the semantics of the elements (i.e. the measured astrophysical quantities) returned by a service, or queried through a database. A UCD is a string which contains textual tokens called ‘words’, separated by semicolons (;). A word is composed of ‘atoms’, separated by periods (.). The hierarchy is as follows: atoms > words > composed words. As an example, the following UCD word: “pos.eq.ra” refers to the following semantic: “Right ascension (ra) part of the position (pos) of a target, in equatorial coordinates (eq)”.

Every valid word is listed on IVOA documents, though composed words are not, as they may be composed by every data producer at their will.

An example of UCDs definition is given in Table 1.

¹³ <http://www.wsmo.org/2004/d4/d4.1/v0.1/20050106/>.

¹⁴ <http://www.w3.org/Submission/WSMO-Lite/>.

¹⁵ <http://wiki.ivoa.net/internal/IVOA/Utypes/WD-Utypes-0.7-20120523.pdf>.

¹⁶ <http://www.ivoa.net/documents/UCD1+/20170831/index.html>.

¹⁷ <http://www.ivoa.net/astronomers/applications.html>.

Table 1
UCDs examples.

UCD	Signification
phot.flux	Photon flux
phot.flux.bol	Bolometric flux
phot.flux.density	Flux density (per wl/freq/energy interval)
phot.flux.density.sb	Flux density surface brightness
phot.flux.sb	Flux surface brightness
meta.bib.bibcode	Bibcode

Table 2
UTYPES definition examples.

UTYPE	Description
Char.FluxAxis.Ucd	UCD for flux
Char.SpectralAxis.Ucd	UCD for spectral coord
Target.Name	Target name
Dataset.DataModel	Data model name and version
Curation.Reference	URL or Bibcode for documentation

2.3.3. The role of UTYPES

UTYPES identify elements in a data model. Their role is to make it possible to bring together on common elements two documents instantiated from the same data model. They are also used for tailoring existing data models to the specific needs of services. Proprietary metadata of a service can be mapped to existing UTYPES to generate a data model based on the mapping results. When needed, UTYPES can also be customized in order to express some specificities of a given metadata. An example from the IVOA Spectral data model is the following: “sed:Data.FluxAxis.value”, identifying the flux value. When a VO service is accessible using TAP, the columns of the returning table are described using a name, a UCD, a UYTPE and a description.

An example of UTYPES is given in [Table 2](#).

The first two UTYPES refer to UCDs. They indicate, in the vocabulary set by the IVOA for UCDs, which words are used to describe the flux and spectral coordinates that appear in the data. The third indicates the name of the target object of the observation concerned. In some cases, both UCDs and UTYPES can designate the same information, but at a different scope (at a general scope for UTYPES identifying data model items, and at the scope of the column content from UCDs).

An example is the UTYPE Target.Redshift and the UCD src.redshift, both of which denote the redshift of the light emitted by the observed target. Although the roles of UTYPES and UCDs are different, both carry some semantic content that can be re-used in ASON.

2.3.4. A concrete example

Elements composing IVOA architecture such as services or databases rely on the joint use of UCDs and UTYPES to expose their metadata in an interoperable way.

The XML schemas proposed by the IVOA for the definition of services according to the protocols used (SSA, ConeSearch for example) are flexible, as shown in [Table 3](#).

The schema structure is respected, but each service implements this schema according to its own profile. We see that the service 1 specifies the UCD for the quantities described, where the service 2 does not. The name of the quantities is described according to different habits (Ks-band or simply Ks) and different degrees of accuracy (service 1 specifies a wavelength, service 2 does not).

The VO provides answers to the problems raised by the heterogeneity of the data and proposes common interrogation protocols for data provider services.

However, limitations still exist:

- The flexibility of the service description schemes used by the IVOA comes from the limited number of mandatory

Table 3
Extracts from IVOA services definition.

Service 1
<pre> <column> <name>Hmag</name> <description>? 2MASS H magnitude (1.6um)</description> <unit>mag</unit> <ucd>phot.mag;em.IR.H</ucd> </column> <column> <name>e_Hmag</name> <description>? Mean error on H magnitude</description> <unit>mag</unit> <ucd>stat.error;phot.mag;em.IR.H</ucd> </column> <column> <name>Kmag</name> <description>? 2MASS Ks magnitude (2.2um)</description> <unit>mag</unit> <ucd>phot.mag;em.IR.K</ucd> </column> </pre>
Service 2
<pre> <column> <name>Kmag</name> <description>? DENIS Ks-band magnitude (5)</description> <unit>mag</unit> </column> <column> <name>Kcorr</name> <description>DENIS Ks-band correlation factor</description> </column> <column> <name>Kxpos</name> <description>X-position in DENIS K-band image</description> <unit>pix</unit> </column> </pre>

keywords. In return, this flexibility generates inaccuracy and a form of heterogeneity since each service can enrich its own description depending on what format it offers. Neighboring services can therefore provide common information with a different level of detail.

- Existing data models are sometimes insufficient in describing certain types of particular data. For example, the definition of a mandatory target name for a spectral data model is meaningless for theoretical spectra that do not correspond to the observation of a real source. For some fields of research (gamma ray astrophysics), the possibilities for describing the observations made are limited. This is why initiatives such as Helio for heliophysics are emerging, and today high-energy gamma observations are at a crossroads: either they choose to integrate the IVOA by defining a suitable data model, or they ‘organize differently but for the moment there is no’ high energy VO’.
- Services implementing IVOA standards and protocols are not expressed in the WSDL language. The discovery of their capabilities, and the automation of their use, must therefore take into account IVOA internal standards and protocols that are not covered by computer research work on the composition (automatic or semi-automatic) of Web services.
- The division of services within the IVOA is another limit: Automatically switching from one data to another (e.g. from an imagery data to a spectroscopic data, or from a radio observation to an infra-red data) is not covered by IVOA specifications. When of two services offering data of the same type (spectra, for example) and in the same wavelength, nothing allows to put them in relation to each other, and a user accessing one of the services will not be informed of the existence of the second.

“DataLink”¹⁸ protocol is the IVOA answer concerning the latter limit of services partitioning.

This protocol allows to define for a service which other VO services can be complementary. This new protocol is likely to improve the situation of the fragmentation of services in the VO but knows two main limitations:

- It assumes the development of a new service or extension of an existing service to provide that link.
- Linked services are based on the knowledge of the protocol programmer. Although this knowledge ensures the adequacy of data between them to a very high level of detail, this level of detail must be defined and controlled by an expert. In addition, there may be other useful services that will not be linked, because of the difficulty of taking into account the large number of services available (more than 12 000).

2.3.5. IVOA registries

“Registries” are directories in which VO services are registered. Those registries play the same role as Universal Description Discovery and Integration (UDDI) registries in WSDL/ Simple Object Access Protocol (SOAP) approach and are WSDL-compliant, while services themselves are not.

Each entity offering VO services can create its own directory, therefore different registries coexist. The Euro-VO, NASA Astronomical Virtual Observatories (NAVO¹⁹) are among the best known. The various existing directories are sometimes limited to services developed by a single entity. Sometimes they include all service providers who contact them to register and whose technical description is validated. To enable users around the world to contact their services once they are operational and usable, an entity may either:

- Create its own directory and save it in the “registry of registries” (RofR) which compiles them all.
- Directly register its services in a more general directory (Euro-VO registries, NAVO...), whose content is referenced in the RofR.

To access all available VO services, a user must first know the existence of this general directory and know how to interrogate and interpret its results. In order to simplify this interrogation, VO-Paris has developed a tool to directly query the RofR according to keywords referring to both the technical description of the services seen in the directories and their general description to the attention of human users. This initiative has many advantages, such as providing a single point of entry to users and allowing more accurate service selection requests than can be found in a single directory, expressed in a single method. However, the number of services that meet the criteria can be significant (several hundred), and identifying correspondences or complementarities between the data of different services remains the sole responsibility of the user, such as the interrogation of each of these services.

Annotating the services based on their outputs and the estimated quality for each output based on the characteristics of the instrument from which the data is derived is also impossible.

In the same way that IVOA services are not expressed in WSDL, IVOA directories do not use UDDI. The clients available to query the UDDI directories are therefore not usable to find these particular services, and this task of service discovery is left to the charge of the VO-compatible software developers. The RegTAP protocol, intended to provide a structured interface for queries to IVOA directories, was finalized recently (end of 2014) and its adoption will depend on the responsiveness of software developers.

2.3.6. Ontologies and services composition using VO services

The need for efficient mapping between services content, IVOA specifications and user vocabulary has led to consider the use of ontologies inside the IVOA, specifically by the IVOA semantics group.²⁰ Such use of ontologies have for example aimed to describe the astronomical object types (Derriere et al., 2010) or the subject of VO registries (Thomas, 2015).

Similar considerations of semantic description for astrophysics guided the creation of a Unified Astronomy Thesaurus (Accomazzi et al., 2014), evolving following user’s suggestions and aiming to provide common vocabulary for every astrophysics or astronomy-related fields.

Nevertheless, there is no current ontology structure or specifications available covering a wide range of astrophysical domains. Besides, automatic composition of VO services remain out of IVOA specifications. As mentioned above, IVOA services can be discovered either using VO-compliant software or through the use of registries.

In our knowledge, the best example of the use of ontologies for services discovery and composition in a branch of astrophysics (namely, heliophysics) is given by Helio (Bentley et al., 2013) through the development of AstroTaverna (Ruiz et al., 2014), which is based on an existing services composition software named “Taverna”. It offers various services to the user and widely uses the ontology described in a 2013 paper (Bentley et al., 2013) for interoperability purposes, providing the dedicated services composition application (Taverna) a smooth access to Helio descriptions of concepts of heliophysics. Nevertheless, the underlying ontology describes concepts and relations specifically for heliophysics but is not a services ontology. Aspects such as services profiles, grounding and process (in the OWL-S vocabulary) are accessible through the HELIO registry service. This registry service is not part of the ontology and is queried through a dedicated API, lowering the reusability of the whole in other application contexts.

2.4. Motivations

We have seen that IVOA specifications proposes some solutions for astrophysical services interoperability. Nevertheless, the diversity of astrophysical observations leads to many, slightly different implementations of the data models. Besides, even if the VO is nowadays a reality and a success, its everyday use requires an important effort of learning. VO-compliant software offers multiple ways to deal with many different use cases, but understanding those tools and their possibilities do require some involvement from the user. This limitation rebukes many researchers, and the VO is mostly used by expert public with knowledge of its inner details.

To hide this complexity and to provide simple interfaces, the VO architecture is often used through dedicated software to complete specific use cases such as:

- The Web interface for extracting photometry measurements (Allen et al., 2014).
- The Web interface for comparing theoretical and observation spectra (Lèbre et al., 2012).

The works presented above offer solutions for services discovery (HELIO), technical interoperability of data (IVOA) and data retrieval (IVOA, HELIO). The solutions for web services description discussed previously (OWL-S, WSMO) offer ontological description for regular Web services. Some specificities encountered when dealing with IVOA services, and with astrophysical services in general, need to be taken into account for an efficient description in this application field.

¹⁸ <http://www.ivoa.net/Documents/WD/DataLink-20120419.html>.

¹⁹ <http://vao.stsci.edu/keyword-search/>.

²⁰ <http://wiki.ivoa.net/twiki/bin/view/IVOA/IvoaSemantics>.

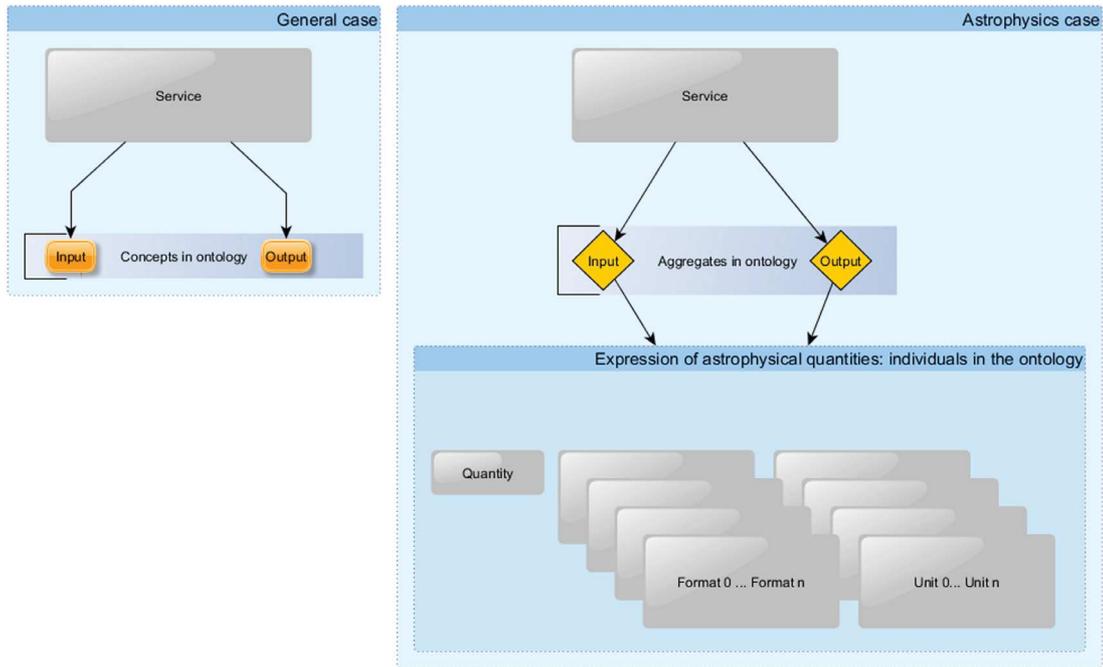


Fig. 4. Expressing different formats and units for the same concept.

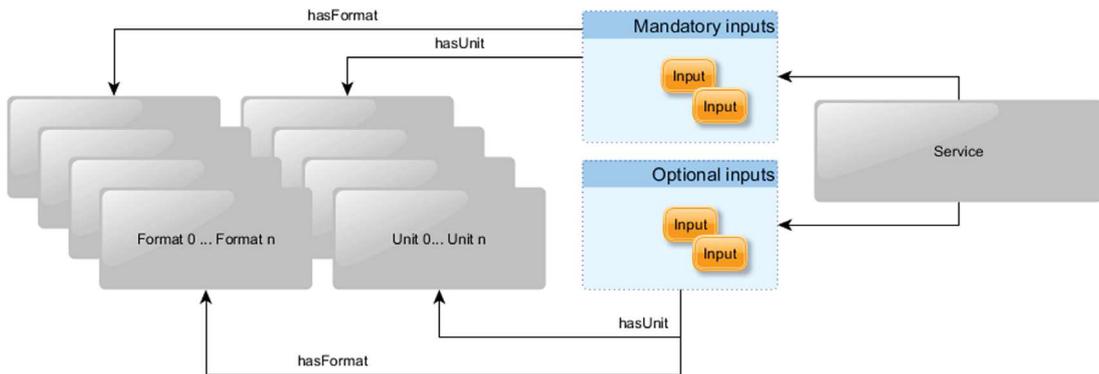


Fig. 5. Optional and mandatory inputs.

2.4.1. What makes IVOA services different from regular Web services?

- IVOA architecture defines protocols for using services that do not embrace regular technologies such as Simple Object Access Protocol (SOAP) or WSDL. Those protocols are specific to IVOA, and IVOA services are not WSDL-compliant. As a consequence, VO protocol-based grounding architecture for atomic services poorly fits into OWL-S grounding or WSMO choreography.
- Outputs and inputs of services are not single concepts, individuals or variables. The result of the combination of three individuals, as an astrophysical quantity goes with its own semantic (i.e. what is the measured quantity?) associated with a unit and a format, as shown in Fig. 4.
- Inputs have different status that can be either mandatory or optional. Only mandatory inputs for a service are to be matched, whereas optional inputs improve the quality of the outputs. This is illustrated in Fig. 5.
- Some inputs are linked to each other in a way that they must be provided by outputs of the same service. We call those inputs correlated inputs. Correlated inputs indicate measurements that only make sense when coming from the same source, because they need the exact same observing

conditions, for example. A basic example is when a service needs a measurement value and the error bar related to the said measurement. The measurement itself and error bar become meaningless if they do not come from the same source. Fig. 6 shows the need to separate correlated inputs from independent inputs, red dashed lines being impossible combination of inputs for the service “Service”.

Therefore, despite their usefulness neither OWL-S nor WSMO fully cover the requirements of our specific field of application in a concrete, applicable way. Our aim is to build an ontology for astrophysical services, more expressive than OWL-S concerning grounding mechanisms and input/output parameters description. We also aim to avoid the multi-layer approach of WSMO and the use of WSML language, specific to WSMO.

The resulting design will help the selection of services by reasoning on a common description fitting astrophysical specificities, and ease the joint use of otherwise non-related services.

OWL-S and WSMO are upper-level ontologies. The domain knowledge corresponding to the services they describe is out of the scope of their specifications. This means that descriptions of the domain knowledge of Web services need to be converted into an

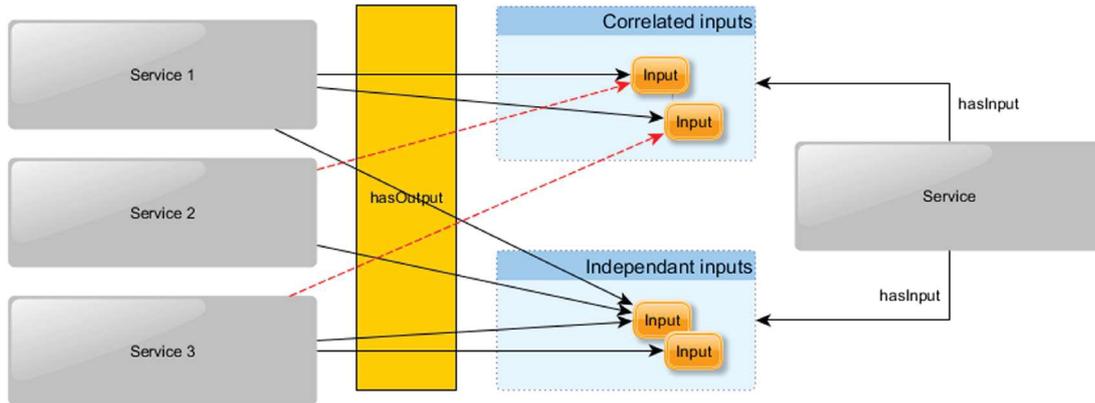


Fig. 6. Correlated inputs.

ontological design usable through the specifications. Such a work is non-trivial and needs to be done on a case-by-case basis (Bensaber and Malki, 2012).

Due to the fact that WSMO relies on different layers of ontologies to describe services and covers some aspects (e.g. mediators) that OWL-S does not, it appears a more complicated task to describe a service in WSMO than it is in OWL-S.

A theoretical example of a service described in WSMO with explanations can be found in WSMO primer.²¹ Apart from this primer WSMO lacks concrete implementations, and only a few basic examples are presented in the WSMO website.²²

Nevertheless, when the great majority of services are atomic services, the different layers of ontologies used in WSMO through states accompanied by guarded transitions are far too expensive for the needs. WSMO-Lite simplifies the services' annotations into a lightweight ontology.²³

Even if WSMO-Lite provides hRests (Roman et al., 2015) to annotate non-WSDL services based on their HTML structure, IVOA services descriptions are not defined in native HTML. So their descriptions, expressed through a specific XML schema called VOTable cannot benefit from this approach. These reasons made OWL-S a more suitable basis for astrophysical services than WSMO and WSMO-Lite.

Description of Web services through use of ontologies is not a recent topic, and some reviews of research in this domain are available (Tosi and Morasca, 2015). Those reviews highlight that real-world use cases are not easy to find, even in the recent literature. Regardless of the ontology description discussed above, most of this field remains theoretical, but some directions are pointed for future research in this area: "Focus on the definition and adoption of a de facto standard formalism and language to creating ontologies for different domains. Several methodologies and languages already exist but their compatibility and their interoperability are not proven and far from the reality." (Tosi and Morasca, 2015).

In this paper, we will address some of those directions by proposing a formalism describing astrophysical services. Our conviction is that enhancing services selection and interoperability in astrophysics goes through selecting services from a description based on their domain knowledge.

To be able to obtain a functional ontology-based selection and use of astrophysical web services, we provide a more expressive description than OWL-S while avoiding complexity of WSMO. This description allows expressing scientific web services, and more generally any domain-related software independently from its

implementation. This will help to set a bridge between different existing infrastructures like IVOA, HELIO and independent Web services.

We will expose the needs that we encountered that were not easily fulfilled by existing submissions and present the solutions that we propose. Then, we will expose how we managed to get a usable ontology that we tested using IVOA services.

In OWL-S, IVOA services should be considered as atomic processes, meaning that a single call waits for a single answer, without further dialog.

Grounding of services is our first concern, as we want both Web services and local scientific libraries to be grounded in order to be able to actually use every source of information. IVOA services descriptions are provided in XML documents, with no WSDL specifications available, and we saw in Section 2 that both OWL-S and WSMO are best used when WSDL-like descriptions are provided.

Besides, IVOA services are related to different IVOA protocols, and their grounding therefore is best represented through a generic mechanism taking such protocols into account than described separately for each service.

Scientific libraries are local software, and their grounding has nothing in common with Web services. We will discuss in Section 3 how we managed to provide a simple grounding mechanism allowing protocols, independent Web services and local libraries to be properly described.

Nevertheless, connecting the inputs in the message sent to the process and the outputs into the result message from the process may be tricky due to the nature of the domain knowledge itself.

First of all, every input corresponds in our use case to a measurement coming from a scientific experiment that may be expressed in multiple ways. Two similar spectrums for example, may be expressed in various combinations of units and formats by different services. They may also be accepted in various combinations of units and formats as inputs for different services. So, the "hasInput" and "hasOutput" relations defined in OWL-S needs to be related to a mechanism generalizing these requirements that we will discuss in Section 3, stating which units and formats are inputs or outputs of services for a same measurement. We will also discuss in Section 3 why putting such a requirement is not straightforward.

Furthermore, some of those parameters may be correlated, in the sense that they must come from the outputs of a single service. This happens for example when one needs a measurement and its error bars. Error bars and the measurement must exist as parameters in the ontology, and should be investigated together during information retrieval. More generally, this will happen whenever two or more given measurements may influence each other or have to be taken in the exact same observation conditions. As a consequence, they should not be treated independently from each other during services' selection.

²¹ <http://www.w3.org/Submission/WSMO-primer/>.

²² <http://www.wsmo.org/>.

²³ <http://www.w3.org/Submission/WSMO-Lite/>.

Lastly, it is important for us that this work comes up with a usable application in the real world, easing the use of the VO out of dedicated software. This means that populating the ontology with available services descriptions in big repositories like IVOA registries is mandatory.

We will expose the challenges we encountered in gathering information from those sources and relating them with relevant ontology concepts. Finally, we will conclude by exposing how the proposed approach may serve the VO architecture through its use for services discovery and query application.

3. The proposed ontology: ASON

3.1. Ontology development methodology

Several methodologies for ontologies building and maintenance have been discussed in [Karray et al. \(2012\)](#). We choose to use Methontology ([Fernández-López et al., 1997](#)) approach. Technical activities described in Methontology go from specification to maintenance, through conceptualization, formalization and implementation. Support activities underlay technical activities, to ensure that none aspect of the development is neglected.

Specification

The purpose of our work is to use ASON as a reasoning basis within an automatic workflow composition software based on the description of scientific use cases. As a result, ASON must not only describe services, but also integrate them in a more global environment in which the descriptions coming from the domain knowledge (i.e. astrophysics) are understood. All the input and output parameters of the services must be semantically reified.

Automation of the use of services must be ensured. This implies being able to represent, in addition to the available knowledge and the general profiles of the services, the technical details of their calls. ASON specifications, according to METHONTOLOGY criteria are exposed in [Table 4](#).

Conceptualization

ASON conceptualization benefited from both existing ontological representations of services and domain knowledge such as IVOA and HELIO representations, which are our main source of knowledge acquisition.

Finding relevant alignments between those sources of knowledge was the first step in conceiving ASON structure. The core structure for the description of astrophysical knowledge in the ontology directly comes from the analysis of International Astronomical Union (IAU) thesaurus, UCDs descriptions and HELIO ontology content.

As new observation methods and results appear in the future in astrophysics, the definition of new individuals or new classes in the ontology may be necessary. Nevertheless, the core structure as it has been defined based on domain knowledge and the available services ontologies is unlikely to change.

Formalization and implementation

In order to obtain a description of the application domain coupled with a functional description of the services, we propose a modular ontology. The modular structure of this ontology aims at separating the description of the technical aspects of services from the representation of their field of application.

An ontology module may be defined as such: “An ontology module is a reusable component of a larger or more complex ontology, which is self-contained but bears a definite relationship to other ontology module” ([Doran, 2005](#)).

ASON is therefore composed of two modules, the first entitled “GENERIC Ontology for Services” (GEOS) and the second is a thematic module which, for the astrophysical case, is entitled “ASTRO-THEM”. The role of ASTRO-THEM is to provide a representation of

the domain-dependent knowledge that is knowledge related to the field of application of the services. As a consequence, ASTRO-THEM only contains astrophysics-dependent concepts, and it contains every astrophysics-dependent concept available in ASON.

ASON is a global ontology ([Wache, 2001](#)), meaning that it contains all the services it describes accompanied by a representation of knowledge related to their field of application. This feature allows ASON to assume the role of a service directory, along with a service ontology and a domain ontology. In contrast, existing approaches for service composition require the use of a services registry in addition to semantic reconciliation between service capabilities and the description of the application domain ([Bansal et al., 2014](#)).

The role of the GEOS module is to describe the technical aspects of services, regardless of their area of application. The definition of this module responds to the need to take into account certain specificities of services that make it difficult to use existing service ontologies. Its task is to express the functional requirements of the services, so that these requirements can be met during the composition of services.

The elements used for ASON construction and an overview of ASON are presented in [Fig. 7](#).

One of the goals is that GEOS can be reused in different contexts than this application case. This means that GEOS has to be considered like a Content Ontology Design Patterns (CP). To propose a CP that can effectively play its role of building block for other uses, there are some requirements to be met ([Presutti and Gangemi, 2008](#)):

- Representation of the CP in OWL must be available
- The CP must be reduced (ideally, from two to ten classes with relations between them)
- The inferences inside a CP must be possible
- Visualization of the CP must be compact and intuitive
- The components must have a linguistic meaning
- The definition of the CP must come from a real case of application.

3.2. Ontology development: support activities

Knowledge acquisition

Knowledge acquisition describes the choice of sources from which knowledge is extracted, and how the concepts are chosen to be part of the ontology. From the IVOA descriptions of services, we get generic information (general scope of the observations contained in the service, name of the observatory, etc.) and data-specific information (wavelengths observed, units expressed, etc.). ASON domain knowledge structure has been conceived using HELIO ontology and IVOA descriptions for main sources.

UCDs definitions are expressed in classes and subclasses and individuals annotated to guide the attachment of new services either from their given UCDs or from their literal description compared to the ontology annotations.

Those annotations describing the concepts come from the UCD IVOA specifications,²⁴ services descriptions and HELIO ontology comments and descriptions.

VO-Paris registry of registries²⁵ is used to get IVOA services description.

IVOA-based and HELIO-based knowledge is expressed in ASTRO-THEM to express services environment, while GEOS module contains concepts and relations for domain-independent services definition. Services themselves are described in ASON, with

²⁴ <http://www.ivoa.net/documents/latest/UCDlist.html>.

²⁵ <http://api.voparis-tmp.obspm.fr/registry>.

Table 4
ASON specifications.

Domain	Astrophysics
Name	ASON : Astrophysical Services ONtology
Conception	Thierry Louge, Mohamed-Hedi Karray, Bernard Archimède
Development	Thierry Louge
Objective	The ontology aims at the description of services in the field of astrophysics. These services will not only be Web services, but also scriptable software and local libraries for example. The functional requirements of the services will have to be expressed. To ensure the fulfillment of functional requirements, the technical description of these services should be related to concepts from their field of application.
Formalism	Semi-formal, annotations and concepts inherited from astrophysics.
Context	ASON will be used as part of an automatic and semantic composition of astrophysical services.
Knowledge sources	HELIO ontology, International Astronomical Union (IAU) thesaurus, UCDs semantic elements from IVOA specifications.

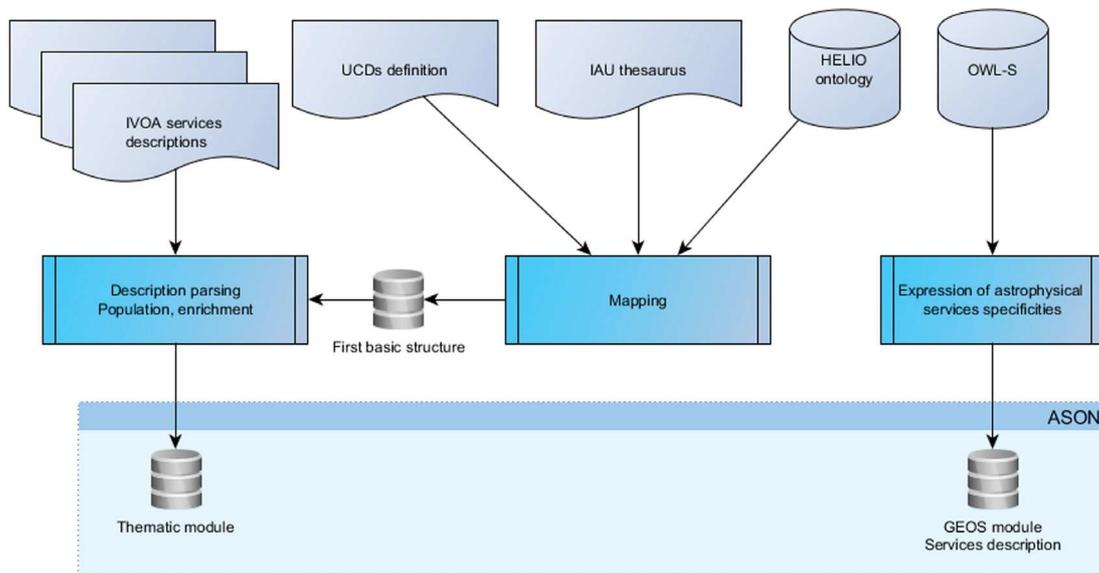


Fig. 7. Elements used for ASON construction.

descriptions related to concepts present in the domain knowledge expressed in ASTRO-THEM and to concepts described in GEOS. The knowledge acquisition itself is a complicated task, and going into the details would need a paper on its own. The most important part of this knowledge acquisition is to gather information concerning the information available through a call to a service, and the service concrete execution and information retrieval. The process of mapping services capacities and technical execution details with ontology content may be summarized as follows:

- Gathering service description (information that the service returns, input data, execution call...)
- Separating technically, domain-independent information from domain knowledge (i.e astrophysical quantities and their expression)
- Expressing both technical and domain-related description through semantics (concepts and relations) provided by the ontology.

Documentation

Documentation in Methontology provides the end user with explanations on what roles are taken by important or representative concepts and relations in the ontology. It exposes the reasons and

the general philosophy of the ontology structure. That is what we do in this subsection.

Grounding services through protocols

Services are the center part in ASON. IVOA services grounding correspond to a WSDL request–response operation, even if IVOA services do not provide WSDL descriptions. OWL-S specification is very clear on WSDL/SOAP groundings, but lacks precision on how non-WSDL/non-SOAP services grounding should be described.

Furthermore, without help from SOAP bindings OWL-S does not specify how service request is actually sent and its results received. Still, the general context is very well known, a service is reachable via an URL, and using a specific protocol to the domain eScience organization like Simple Spectrum Access Protocol (SSAP) or Conesearch protocol for IVOA.

In our approach, a protocol federates the grounding mechanism in a simple and expressive way.

Protocols have some input parameters and a specific way to form the URL to query services, discussed hereafter. For every result the service can give, the way to express query parameters and the input parameters are common to every service using a given protocol independent from the service URL.

Fig. 5 shows an example of grounding for an IVOA service using OWL-S description.

```

<grounding:Grounding rdf:ID="Grounding_ix_30">
  <service:supportedBy rdf:resource="#ix_30"/>
  <!-- Groundings specifications -->
  <grounding:hasAtomicProcessGrounding rdf:resource="#Grounding_RetrieveData_ix30"/>
</grounding:Grounding>

<grounding:AtomicProcessGrounding rdf:ID="Grounding_RetrieveData_ix30">
  <!-- Reference to the corresponding operation -->
  <grounding:Operation rdf:resource="#RetrieveData_ix30_operation"/>
  <grounding:owlsProcess rdf:resource="#RetrieveData_ix30"/>
  <!-- Reference to the input message -->
  <grounding:InputMessage rdf:datatype="&xsd;#anyURI">#RetrieveData_ix30_Input</grounding:InputMessage>
  <!-- Definition of parts of input message -->
  <grounding:Input>
    <grounding:InputMessageMap>
      Specifying how the non-WSDL/SOAP mechanism is used to form the service query
    </grounding:InputMessageMap>
  </grounding:Input>
  <!-- Mapping of outputs to message parts -->
  <grounding:Output>
    <grounding:OutputMessageMap>
      Specifying how the non-WSDL/SOAP mechanism is used to understand the service results
    </grounding:OutputMessageMap>
  </grounding:Output>
</grounding:AtomicProcessGrounding>

<grounding:OperationRef rdf:ID="RetrieveData_ix30_operation">
  <rdfs:comment>
    A pointer to the operation used for retrieving data
  </rdfs:comment>
  <!-- locate port type to be used -->
  <grounding:portType rdf:datatype="&xsd;#anyURI">
    ASON.owl#http://vizier.u-strasbg.fr/viz-bin/votable/-A?-out.all+-source=IX%2F30%2Fseq+
  </grounding:portType>
  <!-- locate operation to be used -->
  <grounding:operation rdf:datatype="&xsd;#anyURI">
    ASON.owl#http://vizier.u-strasbg.fr/viz-bin/votable/-A?-out.all+-source=IX%2F30%2Fseq+
  </grounding:operation>
  For Non-WSDL/SOAP atomic services description in OWL-S,
  we would have to figure out if port/operation distinction is still relevant.
</grounding:WsdlOperationRef>

<process:AtomicProcess rdf:ID="RetrieveData_ix30">
  <rdfs:label>Retrieve some data (ATOMIC)</rdfs:label>
  <rdfs:comment>Retrieves data from ix_30 service</rdfs:comment>
  <process:hasInput>
    <process:Input rdf:ID="RetrieveData_ix30_AvailableCoords">
      <process:parameterType rdf:datatype="&xsd;#anyURI">&concepts;pos.eq.ra</process:parameterType>
      <process:parameterType rdf:datatype="&xsd;#anyURI">&concepts;pos.eq.dec</process:parameterType>
      ... concepts / units combination has still to be expressed
    </process:Input>
  </process:hasInput>
  <process:hasOutput>
    <process:Output rdf:ID="RetrieveData_ix30_AvailableData">
      ... Insert every output concept/unit combination for service ix_30
    </process:Output>
  </process:hasOutput>
</process:AtomicProcess>

```

Fig. 8. Sample of grounding for service ix30 using OWL-S native concepts.

In ASON, we propose a specific grounding for each protocol and a link from the service to the protocol. This makes easier both the grounding definition of services related to protocols and the expressivity of those groundings in the ontology by avoiding large duplications of parameters.

To link a service and its IVOA-specific protocol, we express the URL of the service inside a class (accessUrl concept). Expressing groundings through protocols and queries software allows separating protocols from services, thus generalizing the groundings. It also makes easier to express groundings where http is not the transport mechanism, like local scientific libraries, data repositories (e.g. databases access) that may be described in the ontology like any other service independently from the transport layer. Software components are needed in order to be able to actually use resources described in the ontology. Their role is to call the services, understand their return and deliver the output information. Those pieces of software may be as generic as possible (e.g. for calling a specific protocol as IVOA SSAP), as specialized as needed (e.g. for calling a specific library described in ASON) or somewhere in-between (e.g. calling a generic REST service with its specific input parameters).

QuerySoftware is the class introducing those pieces of software inside the ontology. Fig. 6 shows a concrete example of grounding in ASON (see Fig. 9).

To be able to express such a grounding using a protocol, we have to define related protocol grounding as shown in Fig. 10. Using a protocol is not mandatory, and a service may specify query software and input parameters on his own.

Protocols used by services lead the actual use of said services. To be able to query a service using a protocol, it is necessary to provide every information needed by the protocol to the service. In ASON, we use Semantic Web Rule Language (SWRL²⁶) rules to express that, if a service can be queried through a given protocol, then the services needs to match the requirements of the given protocol.

As an example, the following SWRL rule expresses that the parameter "aggregate" is mandatory for the service "s":

```

UsesProtocol(?s, ?p), HasMandatoryParam(?p, ?aggregate),
  Service.owlpresents(?s, ?profile)
  → HasMandatoryParam(?profile, ?aggregate).

```

²⁶ <http://www.w3.org/Submission/SWRL/>.

```

<ObjectPropertyAssertion>
  <ObjectProperty IRI="#UsesProtocol"/>
    <NamedIndividual IRI="#ix_30"/>
      <NamedIndividual IRI="#cs:ConeSearch"/>
    </ObjectPropertyAssertion>
  <ObjectPropertyAssertion>
    <ObjectProperty IRI="#isAccessedThrough"/>
      <NamedIndividual IRI="#ix_30"/>
        <NamedIndividual IRI="#http://vizier.u-strasbg.fr/viz-bin/votable/-A?-out.all+-source=IX%2F30%2Fseqp+"/>
      </ObjectPropertyAssertion>
</ObjectPropertyAssertion>

```

Fig. 9. Grounding of ix30, a protocol-related service in ASON.

```

<ObjectPropertyAssertion>
  <ObjectProperty IRI="#HasMandatoryParam"/>
    <NamedIndividual IRI="#cs:ConeSearch"/>
    <NamedIndividual IRI="#deg_pos.eq.dec"/>
  </ObjectPropertyAssertion>
<ObjectPropertyAssertion>
  <ObjectProperty IRI="#HasMandatoryParam"/>
    <NamedIndividual IRI="#cs:ConeSearch"/>
    <NamedIndividual IRI="#deg_pos.eq.ra"/>
  </ObjectPropertyAssertion>
<ObjectPropertyAssertion>
  <ObjectProperty IRI="#HasMandatoryParam"/>
    <NamedIndividual IRI="#cs:ConeSearch"/>
    <NamedIndividual IRI="#deg_radius"/>
  </ObjectPropertyAssertion>
<ObjectPropertyAssertion>
  <ObjectProperty IRI="#HasQuerySoftware"/>
    <NamedIndividual IRI="#cs:ConeSearch"/>
    <NamedIndividual IRI="#ConeSearchQuery.py"/>
  </ObjectPropertyAssertion>

```

Fig. 10. Grounding of IVOA cs:ConeSearch protocol in ASON.

Exposing the domain specificities

Building ontology for scientific services does not naturally fit inside a description made of states related to preconditions and post conditions, as expressed by OWL-S and WSMO. Our astrophysical example is based on services both simpler (atomic processes) and more complex than industrial use cases. This complexity comes from the domain knowledge, where many ways of expressing data exists, and where scientific analysis is sometimes very dependent from “nonfunctional parameters” that are in computer science related to “QoS” concepts. A service can give a result with a core of mandatory information, and provide a better result with complementary measurements (a multi-wavelength analysis, for example may result from joint analysis of two to many different wavelengths measurements of the same astrophysical object). These measurements may be individually expressed in various units and formats. This does not mean that a state has to be reached where the service can be used, but rather that a service depends on a core set of information (that may be considered as a starter state) that may be completed.

Expressing which inputs are mandatory or not is expressing which combinations of information, units and formats are available as inputs or outputs for a service and which one is correlated to each other. Every service has inputs and outputs. Those inputs and outputs are part of the domain knowledge, information with their own units and formats. For example, a spectrum may be given as an ascii file, or a FITS (Flexible Image Transport System) file or other formats. It may be expressed in erg/cm², Jy among others.

Every service accepts its inputs and expresses its outputs in a set of formats and units. From OWL-S point of view, this means that a service may have a great number of pre-conditions and be considered usable if a given subset of those conditions is true. In addition, some of those preconditions may be absolutely necessary

for a service, where others may be optional to obtain enhanced results. This would lead to express multiple conditions for every process in OWL-S.

We choose to provide a way of defining such conditions that is more easily understandable, more expressive and less verbose by integrating aggregates (Severi et al., 2010) in ASON. Aggregates link services to information, formats and units in a simple and efficient way. We define relations in the ontology expressing in which units and formats a service expresses or accepts its inputs and outputs. Identifying every combination of parameters and ensuring that they can be used in conjunction plays a key role for services selection and the vastness of services profiles in the domain makes this task non-trivial.

Expressing the complexity of the domain using aggregates

To illustrate how we overcome this difficulty, we will give an example of a service profile **P1**, instance of **Service:Profile**, information needed **pos.eq.ra** instance of **PosEq** and unit **deg**, instance of **Unit**.

Let us consider the relation:

HasRelevantUnit(information, unit).

It says that the given information may be expressed in the given unit.

The relation

HasInput(profile, information)

states that the given information is an input parameter for the given profile and

HasRequestUnit(profile, unit)

states that the given unit is understood by the given profile for its input parameters. Supposing that the logical expression

$$HasRelevantUnit(pos.eq.ra, deg) \bigwedge HasInput(P1, pos.eq.ra) \bigwedge HasRequestUnit(S1, deg)$$

is true would imply that it is possible to give **P1** the information **pos.eq.ra** expressed in **deg** for an input parameter. But this may not be correct, as it is possible to encounter

HasRequestUnit(P1, deg)

because P1 accepts **deg** as a unit for another information than **pos.eq.ra** and accepts **pos.eq.ra** in another unit, but not in **deg**. Stating that

HasAggregation(profile, information, unit)

expresses that a service profile accepts given information in a given unit as an input parameter we can say that:

$$HasRelevantUnit(pos.eq.ra, deg) \bigwedge HasInput(P1, pos.eq.ra) \bigwedge HasRequestUnit(P1, deg) \rightarrow HasAggregation(P1, pos.eq.ra, deg).$$

But this may not necessarily be true and it is impossible in OWL to define relations of arbitrary arity such as **HasAggregation** that we could use to state what combinations of parameters by unit are defined for a given service. To overcome this issue, ASON uses the aggregations and reifications mechanisms (Severi et al., 2010) to express those cases when they appear.

Individuals in ASON **Aggregate** class ties up the couple profile/information with relevant units for a specific information and a specific profile.

Relation **HasInput** or **HasMandatoryParam** coupled with **IsCombinedToUnit** define which aggregates are really usable by the service presented by the profile, taking the units into account.

Therefore individuals of class **Aggregate** have three relations:

- one to the profile (either **HasOutput**, **HasInput** or **HasMandatoryParam**)
- one to the information (**IsCombinedToParam**)
- one to the unit (**IsCombinedToUnit**).

Relations **IsCombinedToUnit** and **IsCombinedToParam** are part of the domain knowledge. An aggregate **IsCombinedToUnit** some unit and **IsCombinedToParam** some information means that this information is expressed in the given unit.

Data formats use the same mechanism that is used for data units.

Expressing services specific needs

Some parameters only make sense if they come from a single source (e.g. a value and its corresponding error bars). Some services may so enjoin that a given set of inputs is set to be coming from a single source. We propose the **hasCorrelatedInput** relation to express this, linking a subset of inputs of a given service together. This is again a 3-ary predicate that we can express via an aggregation. Fig. 11 shows ASON services definition with aggregates, and Fig. 12 shows an internal view of ASON actual content.

When encountering a service in IVOA registries (e.g. the service “vii_231”), the first thing to determine is the protocol that serves this service. For vii_231, this protocol is “ConeSearch”. Therefore, the service is linked with its specific URL (which is an accessUrl individual) through the relation “isAccessedThrough” and with the generic protocol “ConeSearch” through the relation “UsesProtocol”. In GEOS, the ConeSearch protocol is linked to a specific “QuerySoftware” individual (ConeSearchQuery.py). The elements of the service that are domain-dependent can then be described through the relevant relations (“Profile.owlhasOutput”, “hasMandatoryParam” etc.). Those relations link the services inputs and outputs with elements from ASTRO-THEM describing what is mandatory, correlated, and the units/formats combination of input and outputs.

The description of the service is complete when all those elements are described in the ontology. As stated before, protocols are of great help but are in no way mandatory. A service may have its own specific “driver” defined as a QuerySoftware individual. When this is the case, the corresponding software must be provided. It must be callable with the inputs defined for the service in the ontology and also provide the outputs defined for the service in the ontology, so that the call to this service can be fully automated. Fig. 13 exposes the main elements in GEOS used to fulfill astrophysical services requirements, and Table 5 summarizes the concepts described in the previous section.

3.3. Ontology evaluation

ASON is available for online browsing (with tools such as Protégé) and download,²⁷ and a concrete use of this ontology is given

²⁷ <http://cta1.bagn.obs-mip.fr/ASONv1.0.owl>.

in the Composing Automatically and Semantically Astrophysical Services (CASAS) application.²⁸ Metrics for evaluating ontologies exist, such as OntoQA (Tartir et al., 2005) indicators, which it is one of the few methodologies that give evaluation directed both to the end user and to ontology developers.

Those indicators, while relevant for domain or task ontologies, are not well suited for ASON. Its double nature of a modular services ontology embedding domain-specific content makes it difficult to find a relevant set of metrics.

Nevertheless several criteria, non-metric based may be indicated in Table 6.

As a part of its evaluation, we can examine how ASON reaches its goals. An important goal for GEOS is that it can be considered as a design pattern, for reusability concerns in other application domains. We stated in the formalization and implementation section, which are the conditions to be met. We now indicate how those criteria are met in GEOS.

- Representation of the CP in OWL must be available, which is the case for GEOS

- The CP must be reduced (ideally, from two to ten classes with relations between them). GEOS only partly meets this criterion, since it imports OWL-S components that increase this number of concepts up to 96 concepts and 91 relationships.

- The inferences inside a CP must be possible, which is the case here.

- Visualization of the CP must be compact and intuitive, which is the case as expressed in Fig. 13.

- The components must have a linguistic meaning, in order to respect this requirement we gave explicit names to the relations and the concepts used.

- The definition of the CP must come from a real case of application, which is the case.

We see that, although not all the criteria are completely fulfilled, GEOS can be considered very close to the definition of a CP. This module makes it possible to represent Web services, and more generally, all the software sharing a common application domain regardless of their technical aspects. This helps to bridge different existing infrastructures in the same representation, such as VO-compliant services, independent Web services and software not accessible via the Internet (e.g. local libraries).

ASON, in its current form describes 11 136 services. Most of them have been gathered through the automatic query of VO-Paris registry.²⁹ Some additional services (automation of Aladin sky atlas³⁰ through scripts, hyperleda service³¹ ...) that do not come from IVOA architecture have been integrated do demonstrate the feasibility of mixing heterogeneous services in the same information retrieval. This can be tested using the examples presented in the dedicated Web application³² using ASON.

4. Conclusion and future work

This paper focuses on the specification and the conceptualization of ASON, a global ontology for astrophysical services.

ASON knowledge mainly comes from existing thesaurus, other existing ontologies such as HELIO and experts' advices. It is composed of two main modules, which are GEOS for services description and ASTRO-THEM for thematic knowledge.

ASON is available for download and is also at the core of an application aiming at discovering, query and run astrophysical Web services and tools. This application offers automatic service

²⁸ <http://cta1.bagn.obs-mip.fr>.

²⁹ <http://voparis-srv.obspm.fr/portal/vo.php>.

³⁰ <http://aladin.u-strasbg.fr/#information>.

³¹ <http://leda.univ-lyon1.fr/>.

³² <http://cta1.bagn.obs-mip.fr/>.

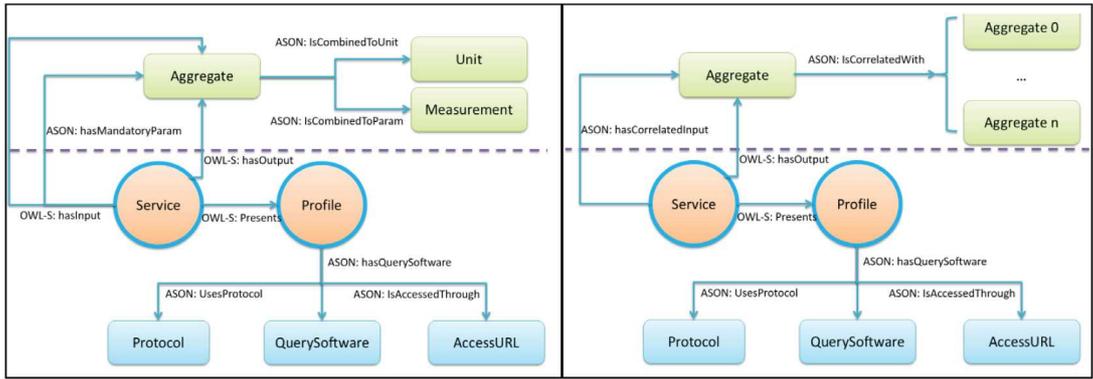


Fig. 11. ASON services structure.

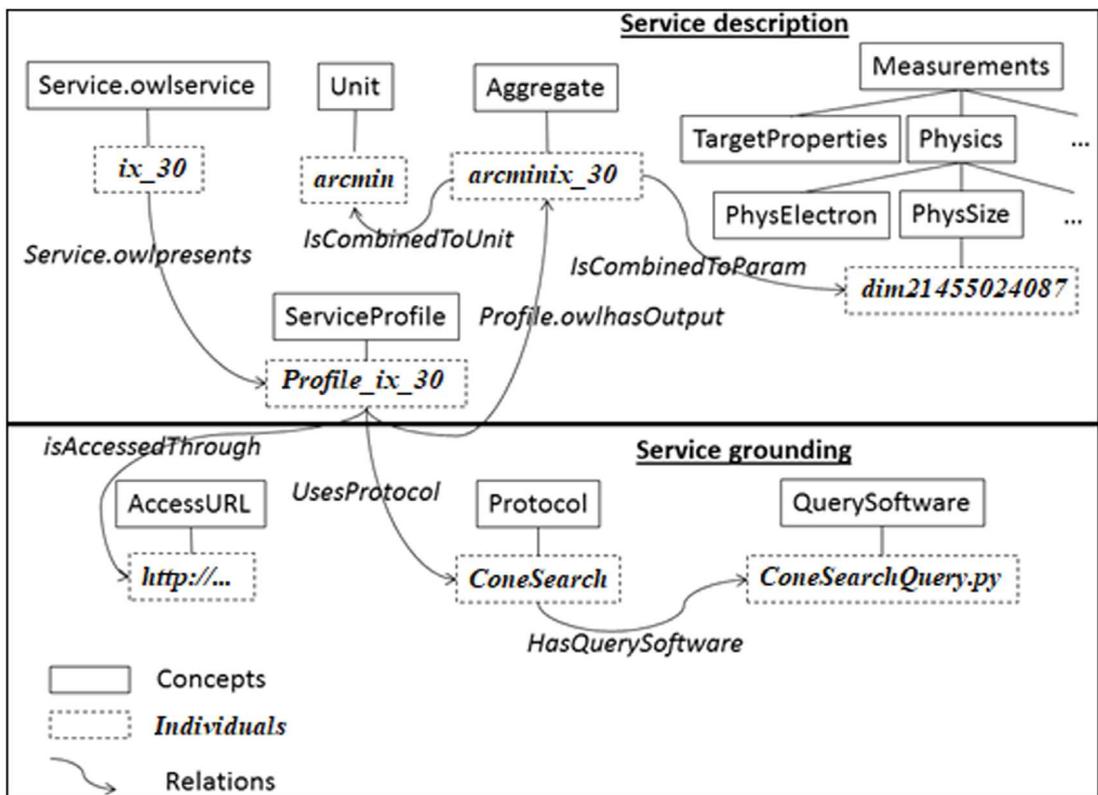


Fig. 12. Extract from the description of ix_30 in ASON (cf. Fig. 8).

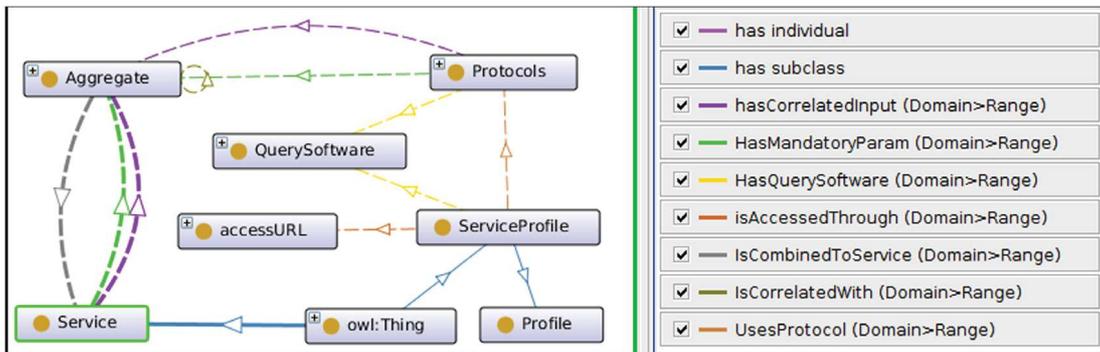


Fig. 13. GEOS main elements.

Table 5
ASON services structure concepts summary.

ASON individual	Documentation
Unit	A unit (Jy, km/s...)
Measurement	Any astrophysical information (e.g. magnetic field measurement, name of a target...) that a service can give (output) or require to be used (input).
Aggregate	Assembles a Unit with a Measurement and is linked to a Service. It expresses that the Unit/Measurement combination is available as an input or an output for the service.
Protocol	A protocol used to query a service. Not mandatory.
QuerySoftware	A local piece of software allowing to query a service or to use a protocol.
AccessURL	The URL of a service.

Table 6
ASON criteria evaluation.

Criteria	Criteria evaluation in ASON
Adaptability	Adaptability in ASON is ensured by the separation between ASON-ASTRO-THEM and GEOS module. Reusing ASON outside of astrophysics will be done by redefining ASON-ASTRO-THEM content.
Completeness	ASON concepts cover all domains specified through available IVOA services descriptions, plus HELIO concepts for heliophysics.
Coupling	HELIO concepts have been embedded into ASON, so there is no coupling between the two. Outside references to OWL-S (mainly Process, Profile and Service) represent the inheritance from OWL-S in ASON.
Accuracy	ASON knowledge comes from standardized, widely used definitions. Defaults in accuracy, if they happen to appear may mainly come from the definition of subsumption relations between IVOA concepts.

discovery and composition on demand, expanding existing possibilities found in VO software and in registries interfaces. The problem addressed by using ASON in this application is to automatically compose Web and non-Web astrophysical services (by using Aladin sky atlas scripts) for information retrieval.

Bringing state-of-the-art semantic capacities in the VO is important as this active field of research will be at the center of the semantic Web to appear in the coming years. Benefiting of interoperability and new possibilities coming with the semantic Web requires semantic-compliant architecture adapted to the domain. That is why we propose ASON as a basis for future semantic-related investigations in the field of astrophysics Web services capacities.

Improving astrophysical services capacities within the semantic Web will also go through the development of dedicated ontologies describing the domain, its terms and their relations. This could provide a structured thesaurus similar in a sense to a "WordNet for astrophysics" that could greatly improve ASON and related ontologies performance, reliability and capacities.

ASON may be reused in different scientific fields outside of astrophysics (geophysics, astrochemistry...) that may encounter the same concerns with existing services ontologies and services descriptions. This could be done by changing ASTRO-THEM content with the corresponding domain knowledge. GEOS module may be improved to fully satisfy the needs for being considered as a design pattern. The number of classes should be lowered. The concepts that GEOS does not inherit from OWL-S are only 6 with 7 relations. Therefore, GEOS should be decoupled from service and profile concepts from OWL-S to fully satisfy the conditions for being a content ontology design pattern. The feasibility is under study.

Appendix

List of acronyms:

ASON: Astrophysical Services ONtology
CASAS: Composing Automatically and Semantically Astro-physical Services

CP: Content Ontology Design Patterns
FITS: Flexible Image Transport System
GEOS: GEneric Ontology for Services
IAU: International Astronomical Union
IVOA: International Virtual Observatory Alliance
NAVO: NASA Astronomical Virtual Observatories
OWL: Web Ontology Language
OWL-S: Web Ontology Language for Services
REST: REpresentational State Transfer
RofR: Registry of registries
SSA: Simple Spectrum Access
SIA: Simple Image Access
SOAP: Simple Object Access Protocol
SWRL: Semantic Web Rule Language
TAP: Table Access Protocol
UCD: Unified Content Descriptors
UDDI: Universal Description Discovery and Integration
VAMDC: Virtual Atomic and Molecular Data Center
VO: Virtual Observatory
VOSI: Virtual Observatory Support Interfaces
WSDL: Web Services Description Language
WSML: Web Service Modeling Language
WSMO: Web Services Modeling Ontology
XML: eXtensible Markup Language.

References

- Accomazzi, A., Gray, N., Erdmann, C., Biemesderfer, C., Frey, K., Soles, J., 2014. The Unified Astronomy Thesaurus. arXiv preprint <http://arxiv.org/abs/arXiv:1403.6656>.
- Allen, M.G., Ochsenbein, F., Derriere, S., Boch, T., Fernique, P., Landais, G., 2014. Extracting photometry measurements from Vizier catalogs. In: *Astronomical Data Analysis Software and Systems XXIII*, Vol. 485. pp. 219–222.
- Bansal, S., Bansal, A., Gupta, G., Blake, M.B., 2014. Generalized semantic Web service composition. *Serv. Oriented Comput. Appl.* 10 (2), 111–133.
- Bensaber, D.A., Malki, M., 2012. Model driven approach for specifying WSMO ontology. In: *CEUR Workshop Proceedings*. pp. 203–213.

- Bentley, R., et al., 2013. HELIO: Discovery and analysis of data in heliophysics. *Future Gener. Comput. Syst.* 29 (8), 2157–2168. Available at: [doi:10.1016/j.future.2013.04.006](https://doi.org/10.1016/j.future.2013.04.006).
- Berneers-Lee, T., Hendler, J., Lassila, O., 2001. The semantic web. *Sci. Am.* 284 (5), 34–43.
- Chen, D., 2016. Enterprise Interoperability Framework (January 2006).
- Derriere, S., Martinez, A.P., Richard, A., 2010. Ontology of astronomical object types. *Int. Virtual Obs. Alliance* Available at: <http://www.ivoa.net/Documents/Notes/AstrObjectOntology/20100117/NOTE-AstrObjectOntology-1.3-20100117.html>.
- Doran, P., 2005. Ontology reuse via ontology modularisation. *KnowledgeWeb Ph.D. Symposium* (Vol. 2006).
- Dowler, P., et al., 2014. IVOA Recommendation: DALI: Data Access Layer Interface Version 1.0. CoRR, [abs/1402.4](https://arxiv.org/abs/1402.4750) Available at: <http://arxiv.org/abs/1402.4750>.
- Dubernet, M.L., et al., 2016. The virtual atomic and molecular data centre (VAMDC) consortium. *J. Phys. B: At. Mol. Opt. Phys.* 49 (49), 74003–74018. Available at: <http://iopscience.iop.org/0953-4075/49/7/074003%5Cnhttp://www.vamdc.eu>.
- Ehrig, M., Staab, S., 2004. QOM - Quick Ontology Mapping. *The Semantic Web—ISWC 2004*, 3298, pp. 1–28. Available at: <http://www.springerlink.com/index/fj4075bm4231x35w.pdf>.
- Fernández-López, M., Gómez-Pérez, A., Juristo, N., 1997. METHONTOLOGY: From ontological art towards ontological engineering. In: *AAAI-97 Spring Symposium Series*, SS-97-06, pp. 33–40. Available at: <http://oa.upm.es/5484/>.
- Gruber, T.R., 1993. A translation approach to portable ontology specifications. *Knowl. Acquis.* 5 (2), 199–220. Available at: <http://www.sciencedirect.com/science/article/pii/S1042814383710083>.
- Horn, A., 1951. On sentences which are true of direct unions of algebras. *J. Symbolic Comput.* 16 (1), 14–21.
- Karray, M.H., Chebel-Morello, B., Zerhouni, N., 2012. A formal ontology for industrial maintenance. *Appl. Ontology* 7 (3), 269–310.
- Lara, R., Roman, D., Polleres, A., Fensel, D., 2004. A conceptual comparison of WSMO and OWL-S. In: *Web Services* (pp. 254–269). Springer, Berlin, Heidelberg, Available at: http://link.springer.com/10.1007/978-3-540-30209-4_19.
- Lèbre, A., Palacios, A., Sanguillon, M., Maeght, P., 2012. Automatic comparison between observed and computed stellar spectra with tools and protocols from the virtual observatory. In: *SF2A-2012: Proceedings of the Annual Meeting of the French Society of Astronomy and Astrophysics*. pp. 365–368.
- Louge, T., Karray, M.H., Archimède, B., Knödseder, J., 2017. CASAS: A tool for composing automatically and semantically astrophysical services. *Astron. Comput.* 20, 34–43. Available at: <http://linkinghub.elsevier.com/retrieve/pii/S2213133717300148>.
- Martin, D., Burstein, M., Mcdermott, D., Mcilraith, S., Paolucci, M., Sycara, K., ... Srinivasan, N., 2007. Bringing semantics to web services with OWL-S. *World Wide Web*, 10, 3Martin, D. et 2007, pp. 243–277.
- Martins, A.F., De Almeida Falbo, R., 2008. Models for representing task ontologies. In: *CEUR Workshop Proceedings*. p. 427.
- Mascardi, V., Cordì, V., Rosso, P., 2007. A comparison of upper ontologies. *Woa* 55–64.
- Presutti, V., Gangemi, A., 2008. Content ontology design patterns as practical building blocks for Web ontologies. *Conceptual Modelling - ER 2008*, Vol. 5231. pp. 128–141. Available at: <http://dl.acm.org/citation.cfm?id=1478324.1478339>.
- Rashmi, S.R., Krishnan, R., 2017. Domain ontologies and their use in building intelligent systems: A comprehensive survey. In: *2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*, (Icimia), pp. 611–616. Available at: <http://ieeexplore.ieee.org/document/7975534/>.
- Roman, D., Kopecký, J., Vitvar, T., Domingue, J., Fensel, D., 2015. WSMO-Lite and hRESTS: Lightweight semantic annotations for Web services and RESTful APIs. *J. Web Semant.* 31, 39–58. Available at: [doi:10.1016/j.websem.2014.11.006](https://doi.org/10.1016/j.websem.2014.11.006).
- Ruiz, J.E., Garrido, J., Santander-Vela, J.D., Sánchez-Expósito, S., Verdes-Montenegro, L., 2014. {AstroTaverna}: Building workflows with {Virtual Observatory} services. *Astron. Comput.* 7–8, 3–11 Available at: <http://www.sciencedirect.com/science/article/pii/S2213133714000419>.
- Severi, P., Fiadeiro, J., Ekserdjian, D., 2010. Guiding reification in OWL through aggregation. In: *CEUR Workshop Proceedings*, 573(January 2010), pp. 408–419.
- Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y., 2007. Pellet: A practical OWL-DL reasoner. *Web Semant. Sci. Serv. Agents World Wide Web* 5 (2), 51–53. Available at: <http://www.sciencedirect.com/science/article/pii/S1570826807000169> [Accessed 02.01.18].
- Tartir, S., Arpinar, I.B., Moore, M., Sheth, A.P., Aleman-Meza, B., 2005. OntoQA: Metric-based ontology quality analysis. *IEEE Workshop on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources*. pp. 45–53.
- Thomas, B., 2015. Development of a VO Registry Subject Ontology using Automated Methods, pp. 1–4. Available at: <http://arxiv.org/abs/1502.05974>.
- Tosi, D., Morasca, S., 2015. Supporting the semi-automatic semantic annotation of web services: A systematic literature review. *Inf. Softw. Technol.* 61, 16–32.
- Wache, H., 2001. Ontology-based information integration: A survey of existing approaches. In: *International Joint Conference on Artificial Intelligence; Workshop: Ontologies and Information Sharing*. pp. 108–117. Available at: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.12.7857>.