



International
Virtual
Observatory
Alliance

Simple Line Access Protocol

Version 1.0

Proposed Recommendation 14 July 2009

This version:

PR-SLAP-1.0-20090714

Latest version:

<http://www.ivoa.net/Documents/SLAP>

Previous version(s):

Editor(s):

Pedro Osuna
Jesus Salgado

Author(s):

Jesus Salgado
Pedro Osuna
Matteo Guainazzi
Isa Barbarisi
Marie-Lise Dubernet
Doug Tody

Abstract

This specification defines a protocol for retrieving spectral lines from different spectral line databases through a uniform interface.

The interface is meant to be reasonably simple to implement by service providers. A basic query will be done in a wavelength range for the different services. The service returns a list of spectral lines formatted as a VOTable.

This type of query, based in a spectral range search, could be too simple to obtain useful results for some services. For instance, in the case of theoretical spectral line databases, the number of lines returned, even for small wavelength ranges, could be huge.

However, some services could implement some other extra search parameters to restrict the search or to score the line as per its probability within a certain physical conditions. How these extra search parameters can be handled by the protocol will be studied in this document.

Status of This Document

This is an IVOA Proposed Recommendation made available for public review. It is appropriate to reference this document only as a recommended standard that is under review and which may be changed before it is accepted as a full recommendation.

Acknowledgements

The authors acknowledge the comments from the DAL forum members and from the IVOA members in general

Contents

1	Overview	3
2	Requirements for Compliance	3
2.1	Compliance	4
3	Spectral Line Service Types	4
4	Spectral Line Query	5
4.1	Input format	5
4.2	Input Parameters	6
4.3	Compulsory input	7
4.4	Non-compulsory Parameters	8
4.5	Free query parameters	10
5	Service Metadata	10
5.1	Metadata Query	10
5.2	Registering a Compliant Service	12
6	Successful Output	12
6.1	Standard output fields	13
6.2	Non-Standard output fields	18

Appendix A: Range definition for input parameters	18
Appendix B: SLAP services for uncorrected and unidentified lines	18
Appendix C: SLAP valid response example	20
Appendix D: SLAP data model summary	21
UTYPE	21
UCD	21
Description	21
DataType	21
Array	21
Size	21
References	23

1 Overview

This Simple Line Access Protocol (SLAP henceforth) specification makes use of the work done in the Simple Spectral Lines Data Model (SSLDM henceforth) definition, as the source of the abstract representation of a spectral line. All the information collected in this document will be used to homogenize the access to the different already existing spectral line databases.

The approach followed by this protocol is the same one as the successful SIAP protocol. This is why this document is intended to be a translation of the SIAP protocol specification for spectral lines access, diverging from it when the physics of the problem force to do it. At the same time, some ideas of the access to Theoretical spectra have been developed and used for Theoretical Spectral Line databases.

The SIAP and SSAP protocols are a two steps process. In a first step, the VO client application request for metadata to the server. These metadata include links to images or spectra respectively. In a second step, the VO client application requests the data from the server. In the particular case of Simple Line Access Services, this kind of services diverges from the SIAP and SSAP ones, as the SLAP service will be, in essence, a one step process, i.e., only the first request for metadata is needed.

Even when no link to astronomical products is expected because of the nature of the service, the SLAP metadata could contain access reference links to html pages, spectra files, spectral line profiles,... etc.

2 Requirements for Compliance

The spectral line query web method **MUST** be supported as in section 4 below. Through this web method, clients search for spectral lines that match certain

client-specified criteria. The response is a VOTable that describes the output spectral lines.

2.1 Compliance

The keywords "MUST", "REQUIRED", "SHOULD", and "MAY" as used in this document are to be interpreted as described in RFC 2119 [34].

An implementation is compliant if it satisfies all the MUST or REQUIRED level requirements for the protocols it implements. An implementation that satisfies all the MUST or REQUIRED level and all the SHOULD level requirements for its protocols is said to be "unconditionally compliant"; one that satisfies all the MUST level requirements but not all the SHOULD level requirements for its protocols is said to be "conditionally compliant".

3 Spectral Line Service Types

It is assumed that compliant spectral line services fall into one of two categories.

1. Observational line databases. Lines observed and identified in real spectra collected by different instrument/projects.
2. Theoretical line databases. Servers containing theoretical spectral lines will be included in this group.

In both cases, the line description and the identification might be already present in a scientific publication, which could be used as a curation mechanism.

This document describes the most common query parameters that these services **must/should/may** support, classified in two main categories: compulsories and non-compulsories.

However, the theoretical line databases services could make use of extra parameters not cited in this document to filter out lines not expected to be identified in an observed spectrum or to score the output lines due to the application of physical models.

Examples:

For the Atomic spectral line database from CD-ROM 23 of R. L. Kurucz
The absorption oscillator strength, $\log(gf)$, can be selected

For the NIST Atomic Spectra Database Lines

As it is extracted from the Saha-LTE model, "The level populations are calculated according to the Boltzmann distribution within each ion and Saha distribution between the ion stages. Thus, to calculate the spectrum from a single ion, e.g., C I, only T_e is required, while for the spectrum from

several ions of the same element (e.g., C I-V), N_e must be defined as well."

At the same time, for observational spectral line databases, some project specific search parameters may be used.

Example:

ISO Astronomical Spectroscopy Database (IASD)

The observation number parameter can be used to select only the lines observed during this ISO satellite observation.

As this kind of extra parameters are not easy to be taken in account in a simple implementation or compiled in the document, a general mechanism is described. As it will be explained later, to allow the discovery of these extra parameters by the VO client applications or by the registry, an implementation of the format=metadata input parameter will be needed. The exact expected response when this parameter appears in the query will be discussed in point 5.

4 Spectral Line Query

The purpose of the spectral line query is to allow users/clients to search in a wavelength range for spectral lines. The most basic query parameters will be the minimum and maximum value for the wavelength range. Additional parameters may be used to refine the search or to model physical scenarios.

4.1 Input format

A URL with two parts transmits the spectral Line query input as an HTTP GET request represented:

* A base URL of the form:
http://<server-address>/<path>? [<extra GET arg>&[...]]

Example:

```
http://esavo02.esac.esa.int:8080/slap/slap.jsp?
```

Note that when it contains extra GET arguments, the base URL ends in an ampersand, &; if there are no extra arguments, then it ends in a question mark, ?.

Every query to a given spectral line query service uses the same base URL.
Constraints expressed as a list of ampersand-delimited GET arguments, each of the form:

```
<name>=<value>
```

Examples

```
WAVELENGTH=5.1E-6/5.6E-6
```

The constraints represent the query parameters that can vary for each successive query.

As it is showed in the example, ranges can be defined following a simple syntax described in appendix A.

This range syntax will be used for the wavelength range and for other input parameters.

For example:

```
INITIAL_LEVEL_ENERGY =8.01E-18/1.6E-17
```

is equivalent to

```
INITIAL_LEVEL_ENERGY>=8.01E-18 AND INITIAL_LEVEL_ENERY<=1.6E-17
```

4.2 Input Parameters

Parameters may appear in any order. If the same parameter appears multiple times in a request the operation is undefined (if alternate values for a parameter are desired the range-list syntax may be used instead). Parameter names are case-insensitive. Parameter values are case-sensitive unless defined otherwise in the description of an individual parameter.

All operations define the following standard parameters:

- **REQUEST** A service **MUST** implement a REQUEST parameter that indicates which service operation is being invoked. The value shall be the name of one of the operations offered by the server. It is an error to reference an unknown service operation. Valid values for SLAP: queryData, getCapabilities(reserved), getAvailability(reserved)

The response to a REQUEST=queryData operation would be a normal query response (see next sections). To have backwards compatibility with older SLAP implementations, if the REQUEST parameter does no appear in the query, the default value is queryData (see example of use in next section)

The response to a REQUEST=getCapabilities operation would be a description of the service capabilities as described later in the Metadata Query section. For

the time being, getCapabilities and getAvailability are only reserved but not fully defined in the present protocol version.

- **VERSION** A service **SHOULD** implement a VERSION parameter that indicates to the service the expected SLAP version to be used in the client-server communication.

Valid values in the form:

```
<major_version>.<minor_version>[.<patch_version>]
```

Examples:

```
1.0  
1.2.1
```

If a SLAP client does not specify the version number in a request, the server assumes the highest standard version supported by the service, and no explicit version checking takes place. If the client specifies an explicit version number, and this does not match a version available from the service at level two, the service returns a version number mismatch error. The client can determine what versions of the protocol the service supports by a prior call to getCapabilities or via a registry query. Please refer to SSAP for version negotiation.

The values of both the REQUEST and VERSION parameters are case-insensitive. The use of REQUEST is mandatory to provide upwards compatibility with future versions.

A given service instance may support multiple versions of the SLAP interface, which includes both the input parameters and the query response with all of its complex metadata, and by default the service assumes the highest standard version which is implemented (access to any experimental versions supported by a service requires explicit specification of the version by the client). Explicit specification of the interface version assumed by the client is necessary to ensure against a runtime version mismatch, e.g., if the client caches the service endpoint but a newer version of the service is subsequently deployed. If desired the client can omit the VERSION parameter to disable runtime version checking, and default to the highest version standard interface implemented by the service. All other request parameters are defined separately for each operation.

4.3 Compulsory input

Wavelength spectral range: The service **MUST** support the following parameter that is used to specify the spectral wavelength range to be used in the search

- **WAVELENGTH**

A service **MUST** implement a parameter to specify the wavelength spectral range. To be specified in meters. This wavelength range will be interpreted as the wavelength in the vacuum of the transition originating the line (ucd="em.wl" ;utype="ssldm:Line.wavelength.value").
For range definition, please refer to appendix A.

As the units in the spectral line database could be different than meters, the service will need to translate from the selected units (meter) to the internal ones. The election of one type of units (in this case the SI unit meters) will help to unify access to different spectral line databases, even when in some cases, the units selected (meter) could not be the best one for the range on interest.

Example:

<http://esavo02.esac.esa.es:8080/slap/jsp/slap.jsp?REQUEST=queryData&WAVELENGTH=5.1E-6/5.6E-6>

to query for spectral lines in the wavelength range between 5.1 and 5.6 micrometers.

4.4 Non-compulsory Parameters

The next list of non-compulsory parameters may be implemented at server side:

- **CHEMICAL_ELEMENT**

A service **MAY** have a search parameter called CHEMICAL_ELEMENT. This parameter would constraint the search to the chemical element selected. A list of different chemical elements could be queried using the comma separator as described in appendix A
(ucd="phys.atmol.element";
utype="ssldm:Line.initialElement.name")

Energy Level range: A service **MAY** accept constraints in the energy for the starting and final levels of the transition. For these parameters, only a range constraint compulsory with appendix A is allowed.

- **INITIAL_LEVEL_ENERGY**

A service **MAY** implement a parameter to specify the minimum and maximum energy for the INITIAL level of the transition. To be expressed in Joules
(ucd=" phys.energy; phys.atmol.initial;phys.atmol.level";
utype="ssldm:Line.initialLevel.energy.value")

- **FINAL_LEVEL_ENERGY**

A service **MAY** implement a parameter to specify the minimum and maximum energy for the FINAL level of the transition. To be expressed in Joules (ucd="phys.energy; phys.atmol.final;phys.atmol.level"; utype="ssldm:Line.finalLevel.energy.value")

The implementation of these search parameters may be extremely useful to model in the client physical situations.

- **TEMPERATURE**

A service **MAY** implement a parameter to specify the expected temperature of the object. To be specified in Kelvin. This parameter would be used (in particular for theoretical spectral line databases) to score the lines in the output using physical models. (ucd="phys.temperature"; utype="ssldm:Line.environment.temperature.value")

Emission probability: A service **MAY** accept constraints in the transition probability through the Einstein A coefficient, defined as the probability per unit time s^{-1} for spontaneous emission in a bound-bound transition.

- **EINSTEIN_A**

A service **MAY** implement a parameter to specify the minimum and maximum Einstein A coefficient as described before. To be specified in s^{-1} . (ucd="phys.at.transProb"; utype="ssldm:Line.einsteinA.value"). For range definition, please refer to appendix A.

It is possible to discriminate the result lines per physical process that originates or modify the line. To do that, and in line with SSLDM, the following input parameters could be added:

- **PROCESS_TYPE**

A service **MAY** implement a parameter to specify the physical process type responsible for the generation of the line or for the modification of its physical properties (utype="ssldm:Process.type"). Valid values are: "Matter-radiation interaction", "Matter-matter interaction", "Energy shift", "Broadening".

- **PROCESS_NAME**

A service **MAY** implement a parameter to specify the physical process exact description responsible for the generation of the line or for the modification of its physical properties (utype="ssldm:Process.name"). There is a great variety of possible values, so this input parameter would need to make use of the FORMAT=METADATA discovery query, as described in next point. Some possible values are: "Photoionization", "Collisional excitation",

“Gravitational redshift”, “Stark broadening”, “Resonance broadening”, “Van der Waals broadening”, etc

4.5 Free query parameters

As we saw in point 3, there is a need to give a general mechanism for the use of some free query parameters to filter out or to score the table result.

Both for the non-compulsory parameters and/or for the free ones, the way to notify to a VO client the implementation or not of these parameters will be through the `getCapabilities` operation or through the `FORMAT=METADATA` response until the `getCapabilities` operation is fully defined. The exact format of this response will be studied in the next section.

Using this information, a VO client would be able to construct on the fly a form, where this information could be entered.

In the future, the SLAP services will be registered and the registry could be able to give the exact description of the SLAP service to the clients, so this process of discovery of special parameters could be done at this point. The exact mechanism for this discovery process will be defined in the registry evolution.

5 Service Metadata

Compliant simple spectral line services describe themselves in two ways. They advertise their availability by registering with a registry service. If possible, depending of the registry service type and the registry definition evolution, all the metadata needed to characterize the service should be included, in particular the **non-compulsory** and **free parameters** implemented by the service. The way this information will be supplied to VO clients will be defining in the registry definition documents.

On the other hand, the services advertise their capabilities through support of the special metadata query signaled through the `FORMAT=METADATA` parameter. The service metadata will allow the registry to discover the service. At the same time, this service will allow the VO clients to make use of the service capabilities, either through a direct call to the service or through the registry in the future.

5.1 Metadata Query

A compliant simple spectral line service **MUST** support spectral line queries with `FORMAT=METADATA` or `REQUEST=getCapabilities`, used to query the service metadata; only metadata is returned by the service in this case. The response to this query advertises of the input parameters the service supports.

Please note that, until a proper definition of the getCapabilities would be provided to supersede current FORMAT=METADATA operation, the service could use the described FORMAT=METADATA response as a valid getCapabilities response.

The structure of the FORMAT=METADATA result will be a VOTable with the parameters supported by the service, using the VOTable <PARAM> tag. Every input parameter supported by the service should be listed as a PARAM element of the RESOURCE that normally contains the spectral line list table. Each param should have a name attribute of the form "INPUT:param_name", where param_name is the parameter name as it should appear in the query URL. For example, name="INPUT:WAVELENGTH" refers to the input parameter "WAVELENGTH".

All input parameters meant to be available to clients of the service must be listed as PARAM elements, including required parameter WAVELENGTH), non-compulsory parameters and free parameters specific to the service. The PARAM may contain a value attribute that should contain the default value that will be assumed if the parameter is not set in the query input URL. Implementors are encouraged to include, as children of the PARAMs, DESCRIPTION elements to describe the parameter and (where appropriate) VALUES elements to given allowed ranges or values.

At the same time, implementers are encouraged to include UCD elements in the PARAM tags to describe the physical meaning of one parameter, not always obvious in particular for physical models.

Example: The input parameter listing below from Observed Spectral Lines database shows that in addition to supporting the required parameter (WAVELENGTH), it also supports the free parameter OBSNO.

```
<?xml version="1.0" encoding="UTF-8"?>
<VOTABLE xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="xmlns:http://www.ivoa.net/xml/VOTable/VOTable-1.1.xsd"
xmlns:ssldm="http://www.ivoa.net/xml/DmSSLDM/v1.0" version="1.0">

<RESOURCE type="results">
  <DESCRIPTION>IASD Simple Line Access Service</DESCRIPTION>
  <INFO name="QUERY_STATUS" value="OK"/>
  <PARAM name="INPUT:WAVELENGTH"
          ucd="em.wl" utype=" ssldm:Line.wavelength.value" value="30">
    <DESCRIPTION> Specify the wavelength spectral range. To be specified in meters. This
wavelength will be interpreted as the wavelength in the vacuum of the transition originating the
line
    </DESCRIPTION>
  </PARAM>

  <PARAM name="INPUT:OBSNO" ucd="obs.id">
```

```
<DESCRIPTION> Specify the ISO observation number where this line has been identified
.</DESCRIPTION>
</PARAM>
```

5.2 Registering a Compliant Service

To be updated in coordination with the Registry Working Group

6 Successful Output

The output returned by a SLAP service is a VOTable, an XML table format, returned with a MIME-type of "text/xml;content=x-votable". The table lists all the Spectral lines found in the server database that match the query constraints. The following requirements are placed on the contents of the table when the query successfully returns a list of spectral lines:

1. The VOTable **MUST** contain a RESOURCE element identified with the tag type="results" containing a single TABLE element which contains the results of the query. The VOTable is permitted to contain addition RESOURCE elements, but the usage of any such elements is not defined here. If multiple resources are present it is recommend that the query results be returned in the first resource element.
2. The RESOURCE element **MUST** contain an INFO with name="QUERY_STATUS". Its value attribute should set to "OK" if the query executed successfully, regardless of whether any matching spectral lines were found.

Examples:

```
<INFO name="QUERY_STATUS" value="OK">
<INFO name="QUERY_STATUS" value="OK"> Successful Search </INFO>
```

3. Each table row represents a different spectral line available to the client.
4. Each row of the output VOTable **MUST** contain FIELDS where the following UCDs have been set.
5. Every FIELD **SHOULD** contain a utype reference to the Simple Spectral Line Data Model whenever possible.
6. The VOTable **MUST** contain a reference to the SSLDM namespace

```
xmlns:ssldm="http://www.ivoa.net/xml/DmSSLDM/v1.0"
```

7. The VOTable **MAY** contain references to other name spaces, like SLAP, Characterization, etc

6.1 Standard output fields

- One field **MUST** have utype="ssldm:Line.wavelength.value", with datatype="double" and ucd="em.wl", containing the wavelength in the vacuum of the transition originating the line in meters.
- Note: It is allowed to have more than one wavelength field in different units in order to preserve the precision of the original value prior to unit conversion in the client. If this is the case and to get backwards compatibility with already existing services, there **MUST** be one field with utype="ssldm:Line.wavelength.value" and unit="m" in the VOTable response. Other fields with the same utype should have a different value in the unit field descriptor.
- Exactly one field **MUST** have utype="ssldm:Line.title", with datatype="char" arraysize="*" and ucd="em.line", containing a small description identifying the line.

Note that this line title is only a short string representation to be used in the clients for display. Syntax is free.

Examples:

```
H I  
N III 992.873 A
```

In case of corrected but unidentified lines, some examples could be:

Examples:

```
M31 1001.784 A  
011910191 800.2 A
```

- Exactly one field **SHOULD** have utype="ssldm:Line.identificationStatus" with datatype="char", arraysize="*", containing the identification status of the line. Possible values are: unidentified, identified and uncorrected. Note that this last one value is only valid for a SLAP-like service, but not for a standard SLAP service. See appendix B
- Exactly one field **SHOULD** have utype="ssldm:Line.species.name" with datatype="char", arraysize="*" and ucd="phys.atmol.element", containing a name of the chemical element source of this line.
- Exactly one field **SHOULD** have utype="ssldm:Line.initialLevel.name", with datatype="char", arraysize="*" and ucd="phys.atmol.initial;phys.atmol.level", containing a full description of the initial level of the transition originating the line.

- Exactly one field **SHOULD** have `utype="ssldm:Line.finalLevel.name"`, with `datatype="char"`, `arraysize="*"` and `ucd="phys.atmol.final;phys.atmol.level"`, containing a full description of the final level of the transition originating the line.
- Exactly one field **MAY** have `utype="ssldm:Line.observedWavelength.value"`, with `datatype="double"` and `ucd="em.wl"`, containing the observed wavelength in the vacuum of the transition originating the line in meters, as it was observed. This may be used by observational spectral line databases.
- Exactly one field **MAY** have `utype="slap:Query.Score"`, with `datatype="double"`. A line with a higher score more closely matches the query parameters. The query response table should normally be returned sorted in order of decreasing values of score, with the top-scoring items at the top of the list. The details of the heuristic used to compute score are left to the service. This is particularly useful for the theoretical spectral line databases examples explained in section 3. Please note that a reference to the relevant characterization namespace is needed in the VOTable response.

xmlns:slap="http://www.ivoa.net/xml/DalSlap/v1.0"

- Exactly one field **MAY** have `utype="ssldm:Line.initialLevel.energy.value"`, with `datatype="double"` and `ucd="phys.energy;phys.atmol.initial;phys.atmol.level"`, describing the energy for the starting level of the transition which originates this line. To unify results, the value must appear in Joules.
- Exactly one field **MAY** have `utype="ssldm:Line.finalLevel.energy.value"`, with `datatype="double"` and `ucd="phys.energy;phys.atmol.final;phys.atmol.level"`, describing the energy for the final level of the transition which originates this line. To unify results, the value must appear in Joules
- Note: It is allowed to have more than one **Line.initialLevel.energy.value** and **Line.finalLevel.energy.value** fields in different units in order to preserve the precision of the original value prior to unit conversion in the client. If this is the case and to get backwards compatibility with already existing services, there **MUST** be one field with `utype="ssldm:Line.initialLevel.energy.value"` and one with `utype="ssldm:Line.finalLevel.energy.value" unit="Joules"` in the VOTable response. Other fields with the same `utype` should have a different value in the unit field descriptor.

- o Exactly one field **MAY** have
`utype="ssldm:Line.initialLevel.configuration"` and one with
`utype="ssldm:Line.finalLevel.configuration"`, with `datatype="char"` and
`arraysize="**"` describing the electron configuration of the initial/final levels
of the line.

The format of the string representing the electron configuration is as follows.

For atomic levels:

Fe basic atomic level configuration is:

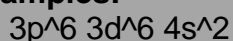
Examples:



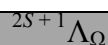
Where we have subtracted the closed shell configuration from the enumeration.

We have selected the LATEX convection to describe sub-indexes and super-indexes. That means the previous atomic configuration could be serialized as string in the SLAP response in the following way:

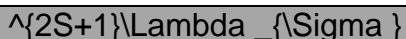
Examples:



In the case of molecular levels, we usually need to use Greek symbols, so, e.g.,



Would be serialized as



- o Exactly one field **MAY** have
`utype="ssldm:Line.initialLevel.quantumState"` and one with
`utype="ssldm:Line.finalLevel.quantumState"`, with `datatype="char"` and
`arraysize="**"` describing the quantumState of the initial and final levels in a
parseable string representation. The format would be the following:

`[label:type:numerator:denominator;label:type:numerator:denominator; ...][...]`

i.e., every quantum number is defined by:

- label is:
`ssldm:Line.initialLevel.quantumState.quantumNumber.label`
- type is:
`ssldm:Line.initialLevel.quantumState.quantumNumber.type`

- numerator is:
ssldm:Line.initialLevel.quantumState.quantumNumber.numeratorValue
- denominator is:
ssldm:Line.initialLevel.quantumState.quantumNumber.denominatorValue

(Please refer to the Simple Spectral Line Data Model for meaning and format)
Quantum Numbers are separated by the “;” character.

To allow that a level needs more than one quantum states to be described, the quantum states would be enclosed by squares.

Example:

```
[J:totalAngularMomentumJ:1:1;F1:totalAngularMomentumF:0:1;
F:totalAngularMomentum:1:1]
```

- One field **MAY** have utype=“**ssldm:Process.type**” with datatype=“char” and arraysize=“*” identifying the types of physical processes responsible for the generation of for the modification of its physical properties. As more than one value is possible per line, the values would be separated by the “:” character, in the following way:

○

Example:

```
“Matter-radiation interaction”：“Broadening”
```

Valid values are: “Matter-radiation interaction”, “Matter-matter interaction”, “Energy shift”, “Broadening”.

- One field **MAY** have utype=“**ssldm:Process.name**” with datatype=“char” and arraysize=“*”, identifying a description of the physical processes responsible for the generation of for the modification of its physical properties. As more than one value is possible per line, the values would be separated by the “:” character, in the following way:

Example:

```
“Photoionization”：“Natural broadening”
```

This list should be in correspondence with the value of the possible process types described in previous point.

- One or more fields **MAY** have ucd="meta.bib", with datatype="char" and arraysize="*", to specify a http links that contains a scientific publication related to the spectral line. Since the URL will often contain metacharacters the URL is normally enclosed in an XML CDATA section (<![CDATA[...]]>) or otherwise encoded to escape any embedded metacharacters.
- One or more fields **MAY** have ucd="meta.ref.url" with datatype="char" and arraysize="*" and free names, to specify URLs that contains extra information related to the spectral line. Same criteria than before about the CDATA section.

In the case of observational line databases, some characterization information of the observation itself could be relevant. Next, we present some examples of possible observation-related output metadata (note that all the following fields are in line to the ones described in the SSAP specification)

- Exactly one field **MAY** have utype="Target.Name", with datatype="char", and arraysize="*" containing a short string identifying the observed astronomical object, suitable for input to a name resolver.
- Exactly one field **MAY** have utype="char:SpatialAxis.Coverage.Location.Value", with datatype="double", arraysize="*" and ucd="pos", containing the observation position of the observation in the format: ra dec, white space separated and both in deg. Please note that a reference to the relevant characterization namespace is needed in the VOTable response.

```
xmlns:char ="http://www.ivoa.net/xml/DmCHAR/v1.13"
```

- Exactly one field **MAY** have utype="char:TimeAxis.Coverage.Bounds.Start", with datatype="char", arraysize="*" and ucd=" time.start;obs.exposure", containing the start time for the observation in MJD with units of days. Please note that a reference to the relevant characterization namespace is needed in the VOTable response.
- Exactly one field **MAY** have utype="char:TimeAxis.Coverage.Bounds.Stop", with datatype="char", arraysize="*" and ucd=" time.stop;obs.exposure", containing the end time for the observation in MJD with units of days. Please note that a reference

to the relevant characterization namespace is needed in the VOTable response.

6.2 Non-Standard output fields

In many occasions, extra scientifically interesting parameters may be added to the output. Implementors are encouraged to add descriptions and UCDs to the return fields to clarify the meaning of this information and utypes to the Line data Model or other existing IVOA Data Model, whenever possible.

Appendix A: Range definition for input parameters

The Simple Image Access Protocol (SIAP), Simple Spectrum Access Protocol (SSAP) and the Simple Line Access Protocol (SLAP) are URL based protocols, i.e., the constraints in the queries for the input parameters need to be done in a valid URL.

However, the URL standard is not very specific defining special constraints as, e.g., ranges. We describe here some examples of accepted range syntax for URL DAL protocols:

param = x	Equality	Equivalent to param=x
param = x/y	Closed range	Equivalent to param>=X AND param<=y
param = x/	Open range	Equivalent to param>=x
param = x,y	List	Equivalent to param=x OR param=y
param = x/a,b/y	Range list	Equivalent to [param<=x AND param<=a] OR [param>=b and param<=y]

Appendix B: SLAP services for uncorrected and unidentified lines

One important constraint for observational line databases in the SLAP definition is to limit the lines in the SLAP output to corrected lines, i.e. only the lines that have been corrected from wavelength shifts (including, not only the astrophysical redshift, but any process that could produce a line peak displacement). (Note that unidentified lines can be distributed as long as these lines could be corrected in wavelength. See observational “**ssldm:Line.title**” proposed syntax.)

The reason for that is, in case a user wants to compare two different SLAP services outputs or to compare one SLAP service output to one SSAP service output, the data comparison requires previous correction. This correction is most of the times quite complex and it should be done by the data providers, the ones who have the knowledge and expertise to correct properly the data and not by a client application, as this process is prone to scientific errors.

However, even when this kind of SLAP-like services cannot be registered in the IVOA as standard SLAPs, the creation of services for internal project consumption could be desirable.

If a service is able to provide both corrected and uncorrected lines, the service could be registered as soon as a default call to the service only return the corrected lines as output. The following optional parameter could be use to select other types of lines.

- **CORRECTION_STATUS**

A service **MAY** have a search parameter called `CORRECTION_STATUS`. This parameter would constraint the search to lines with a certain level of identification. As specified in [1] the possible values are:

UNCORRECTED
CORRECTED

And the default value **MUST** be “CORRECTED”.
(`ucd="em.line;meta.id.cross";utype="ssldm:Line.correctionStatus"`)

- **IDENTIFICATION_STATUS**

A service **MAY** have a search parameter called `IDENTIFICATION_STATUS`. This parameter would constraint the search to lines with a certain level of identification. As specified in [1] the possible values are:

UNIDENTIFIED
IDENTIFICATION_UNCERTAIN
IDENTIFICATION_PROVISIONAL
IDENTIFIED

(`ucd="em.line;meta.id.cross";utype="ssldm:Line.identificationStatus"`)

For theoretical line databases, identified will be synonymous to predicted.

For the output:

- o Exactly one field **MAY** have `utype="ssldm:Line.correctionStatus"`, with `datatype="char"` and `ucd="em.wl.central;meta.id.cross"`, describing if the "Idm:Line.wavelength" has been calculated.

Please note that for uncorrected lines the compulsory field with `utype="Idm:Line.wavelength"` (corrected wavelength in vacuum) will contain either the observed wavelength for "Uncorrected" or a provisional assignment value.

- o Exactly one field **MAY** have `utype="ssldm:Line.identificationStatus"`, with `datatype="char"` and `ucd="em.line;meta.id.cross"`, describing the identification status of the line.

Appendix C: SLAP valid response example

```
<?xml version="1.0" encoding="UTF-8"?>
<VOTABLE xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="xmlns:http://www.ivoa.net/xml/VOTable/VOTable-1.1.xsd"
xmlns:ssldm="http://www.ivoa.net/xml/DmSSLDM/v1.0" version="1.0">
<RESOURCE type="results">
<DESCRIPTION>European Space Astronomy Centre - Simple Line Access Protocol
(SLAP)</DESCRIPTION>
<INFO name="QUERY_STATUS" value="OK"/>
<INFO name="SERVICE_PROTOCOL" value="1.0">SLAP</INFO>
<INFO name="REQUEST" value="queryData"/>
<INFO name=" WAVELENGTH" value="7.6E-6/1.E-5"/>

<TABLE>
<FIELD ucd="obs.id" name="OBSNO" datatype="char" arraysize="*" />
<FIELD ucd="em.wl" name="WAVELENGTH" utype="ssldm:Line.wavelength.value"
datatype="double" />
<FIELD ucd="em.freq" name="FREQUENCY" utype=" ssldm:Line.frequency.value"
datatype="double" />
<FIELD ucd="em.wavenumber" name="WAVE_NUMBER"
utype="ssldm:Line.wavenumber.value" datatype="double" />
<FIELD ucd="spect.line.width" name="LINEWIDTH"
utype="ssldm:Line.observedBroadeningCoefficient.value" datatype="double" />
<FIELD ucd="spect.line.width;meta.unit" name="LINEWIDTH_UNIT" datatype="char"
arraysize="*" />
<FIELD ucd="meta.bib" name="ADS_CODE" datatype="char" arraysize="*" />
<FIELD ucd="em.line" name="IDENTIFICATION" utype="Idm:Line.title" datatype="char"
arraysize="*" />
<FIELD ucd="phys.atmol.transition" name="TRANSITION" datatype="char" arraysize="*" />
<FIELD name="LINE_TYPE" datatype="char" arraysize="*" />
<FIELD ucd="spect.line;phot.flux" name="FLUX" utype="ssldm:Line.observedFlux.value"
datatype="double" />
<FIELD ucd="spect.line.intensity" name="PEAK_INTENSITY"
utype="ssldm:Line.observedIntensity.value" datatype="double" />
<FIELD ucd="spect.line.intensity;stat.snr" name="SIGNAL_TO_NOISE" utype="
ssldm:Line.significanceOfDetection.value" datatype="double" />
```

```

<DATA>
<TABLEDATA>

<TR>
<TD>116003190</TD>
<TD>8.03E-6</TD>
<TD>37333.748443</TD>
<TD>1245.330012</TD>
<TD>0.008580</TD>
<TD>micron</TD>
<TD>2001ApJ...552..544F</TD>
<TD>H2</TD>
<TD>0-0 S(4) </TD>
<TD>L</TD>
<TD>6.800000088194428E-16</TD>
<TD> </TD>
<TD> </TD>
</TR>

<TR>
<TD>116003190</TD>
<TD>8.997E-6</TD>
<TD>33321.107035</TD>
<TD>1111.481604</TD>
<TD>0.010960</TD>
<TD>micron </TD>
<TD>2001ApJ...552..544F</TD>
<TD>[ArIII] </TD>
<TD>3P2-3P1</TD>
<TD>L</TD>
<TD>4.8899999785631705E-15</TD>
<TD> </TD>
<TD>0.000000</TD>
</TR>

..... more lines data.....

</TABLEDATA>
</DATA>
</TABLE>
</RESOURCE>
</VOTABLE>

```

Appendix D: SLAP data model summary

UTYPE	UCD	Description	Data Type	Array
ssldm:Line.title (COMPULSORY)	em.line	small description identifying the line	char	*

ssldm:Line.wavelength.value (COMPULSORY)	em.wl	wavelength in the vacuum of the transition originating the line in meters.	char	*
ssldm:Line.initialLevel.energy.value	phys.energy; phys.atmol.initial;phys.atmol.level	energy for the INITIAL level of the transition	double	
ssldm:Line.finalLevel.energy.value	phys.energy; phys.atmol.final;phys.atmol.level	energy for the FINAL level of the transition	double	
ssldm:Line.environment.temperature.value	phys.temperature	expected temperature of the object	double	
ssldm:Line.einsteinA.value	phys.at.transProb	Einstein A coefficient, probability per unit time s^{-1} for spontaneous emission in a bound-bound transition	double	
ssldm:Process.type	meta.title	physical process type responsible for the generation of the line or for the modification of its physical properties	char	*
ssldm:Process.name	meta.title	physical process exact description responsible for the generation of the line or for the modification of its physical properties	char	*
ssldm:Line.identificationStatus		identification status of the line	char	*
ssldm:Line.initialLevel.name	phys.atmol.initial;phys.atmol.level	description of the initial level of the transition originating the line	char	*
ssldm:Line.finalLevel.name	phys.atmol.final;phys.atmol.level	description of the final level of the transition originating the line	char	*
ssldm:Line.observedWavelength.value	em.wl	observed wavelength in the vacuum of the transition originating the line in meters	char	*
slap:Query.Score		Line ranking "more closely matches the query parameters"	double	
ssldm:Line.initialLevel.configuration	phys.atmol.configuration	Description of the electron configuration of the initial level of the line	char	*
ssldm:Line.finalLevel.configuration	phys.atmol.configuration	describing the electron configuration of the final level of the line	char	*

ssldm:Line.initialLevel.quantumState	meta.title	Description of the quantum state of the initial level in a parseable string representation	char	*
ssldm:Line.finalLevel.quantumState	meta.title	Description of the quantum state of the final level in a parseable string representation	char	*
Target.Name	meta.id;src	short string identifying the observed astronomical object, suitable for input to a name resolver	char	*
char:SpatialAxis.Coverage.Location.Value	Pos	observation position of the observation in the format: ra dec, white space separated and both in deg	char	*
char:TimeAxis.Coverage.Bounds.Start	time.start; obs.exposure	start time for the observation in MJD with units of days	char	*
char:TimeAxis.Coverage.Bounds.Stop	time.stop; obs.exposure	end time for the observation in MJD with units of days	char	*

Please note that only the two first utypes correspond to compulsory fields in the protocol.

References

[1] [Osuna/Guainazzi/Salgado/Dubernet/Roueff]

ssldm: Simple Spectral Lines Data Model v0.9

[2] [Tody/Plante]

Simple Image Access Specification,

<http://www.ivoa.net/Documents/latest/SIA.html>

[3] [Hanisch/Arviset/Genova/Rino]

IVOA Document Standards,

<http://www.ivoa.net/Documents/DocStd/20090302/PR-DocStd-1.2-20090302.html>

[4] [Smith/Heise/Esmond/Kurucz]

Atomic spectral line database from CD-ROM 23 of R. L. Kurucz.,

<http://cfa-www.harvard.edu/amdata/ampdata/kurucz23/sekur.html>

[5] [Various]

NIST National Institute of Standards and Technology,

NIST Atomic Spectra Database Lines,

http://physics.nist.gov/PhysRefData/ASD/lines_form.html

[6] [Ochsenbein et al]

VOTable Format Definition V. 1.1,

<http://www.ivoa.net/Documents/cover/VOT-20040811.html>

[7] [McDowell et al]

IVOA Spectral Data Model,

<http://www.ivoa.net/Documents/latest/SpectrumDM.html>

[8] [Tody et al]

IVOA Simple Spectral Access Protocol,

<http://www.ivoa.net/Documents/latest/SSA.html>