VAMDC XSAMS Processor Service Specification

Document Information

Editors:	G. Rixon, M. Doronin
Authors:	G. Rixon
Contributors:	VAMDC WP6 working group, T. Marquart
Type of document:	standards documentation
Status:	draft
Distribution:	public
Work package:	WP6
Version:	12.07
Date:	31/07/2012
Document code:	
Document URL:	`http://www.vamdc.org/documents/xsams-processor_v12.07.pdf`

Abstract: Applications to process data in XSAMS format may be made available as web sites, which makes them accessible for interactive use, or as web services, making them accessible to scripts and other software. This standard prescribes a form for these web applications that includes both the interactive web-site and scriptable web-service.

Version History

Version	Date	Modified By	Description of Change
V11.12	21/12/2011	G. Rixon	first version
V12.07	31/07/2012	M. Doronin	More strict specification

Disclaimer

License

All rights reserved

Acknowledgements

VAMDC is funded under the "Combination of Collaborative Projects and Coordination and Support Actions" Funding Scheme of The Seventh Framework Program. Call topic: INFRA-2008-1.2.2 Scientific Data Infrastructure. Grant Agreement number: 239108.

CONTENTS

CHAPTER

ONE

UNIFORM PROTOCOL FOR A WEB APPLICATION PROCESSING XSAMS

Applications to process data in XSAMS format may be made available as web sites, which makes them accessible for interactive use, or as web services, making them accessible to scripts and other software. This standard prescribes a form for these web applications that includes both the interactive web-site and scriptable web-service.

Web applications conforming to this standard can be registered in the VAMDC registry. Registration makes the applications available to generic UIs such as the VAMDC portal.

Conforming web applications can read data either from a URL (e.g. the portal passes a data-extract URL leading to a VAMDC database) or from an uploaded file (e.g. a user loads data from a file on his computer).

1.1 General nature of the web application

The web application consists in a set of web resources (where a "web resource" is anything that has its own URL) arranged in a tree structure. This standard specifies the web resources that must be provided, including their names, semantics and positions in the tree. A web application may also provide other web resources.

All the required web-resources must be available via HTTP v1.1.

None of the required web-resources use the Simple Object Access Protocol (SOAP). All these resources follow the paradigm Representational State Transfer (REST).

The core of the application is a web resource called, in this standard, its primary result. This resource represents the result of the application's processing of one or more XSAMS input documents. The application must have exactly one primary result, implying that all the inputs are combined.

The primary result may be machine-readable (e.g. a transformation of the XSAMS input into HITRAN's legacy format) or human readable (i.e. a web page). If human readable, it may well link to further pages showing related results (e.g. the primary result might be a web page showing a table of atomic states, with links to the details of each state). A machine-readable result may also contain links, but this standard does not define how the links are accessed. Hence, a generic client, such as the VAMDC portal, would not know how to follow the links.

An HTTP request to the primary result specifies the input data, either by passing URLs for those or by including them in the request. Therefore, a client application or script can call the primary result directly as a web service.

To make the application accessible from web browsers, the application must also supply a web page that leads the user to the primary result, allowing him to specify the input data. This resource is typically an HTML form and is referred to as "the form" in the rest of this standard.

Applications following this standard are expected to be registered, so they must provide a "VOSI capabilities" resource to convey registration details to the VAMDC registry.

Some of the registration details are repeated, in human-readable format, on the form.

The application should also provide a "VOSI availability" resource to allow checking of the system.

1.2 Details of the web resources

1.2.1 The root resource

The root resource of the application must be available via HTTP GET and must contain a browser-accessible html form and a short verbose description of the XSAMS processor service.

The form must allow the user to specify the input data either by giving URLs (which the user might copy and paste from some other UI) or by uploading files from the desktop. Both modes must be available.

The appearance and behaviour of the form should be fixed. It should not vary according to parameters in the URL that displays it, or in response to information cached in the user's browser session. If the form is made adaptable via parameters or responsive to session history, then it must be useable in the absence of this information.

The form must give a general description of the processing that will be applied to incoming XSAMS documents. This description may be given either in form of a link to another page or in form of a paragraph on the root page.

The author of the application may choose freely the content of the form provided that it meets the requirements above.

1.2.2 The primary result

The primary result is at the URL /service relative to the root URL.

In a given application, the primary result may have any MIME-type, but it must always have the same MIME-type in that application.

Submitting XSAMS documents

The primary result must accept all of the following ways to specify the input data.

- In HTTP GET, via the parameter called `url`, embedded in the URL used to call the application. The value of this parameter is the URL leading to the XSAMS data, and that URL must be in the HTTP scheme. The URL for the input data must be URL encoded before embedding in the URL for the primary result. (This necessary encoding makes it infeasible for a user to simply type the URL in a browser's address-bar, hence the need for the form to invoke the primary result.) The parameter may be repeated to specify more than one XSAMS document.
- In HTTP POST, via the parameter called `url` as above, except that the parameter is written in the body of the request instead of embedding in the URL of the primary result.
- In HTTP POST, via the parameter called `upload`, written in the body of the request. The value of the parameter is the XSAMS document itself, packaged according to internet RFC 1867. This parameter may be repeated to upload multiple XSAMS documents.

The application is only required to process XSAMS inputs and should typically reject other types.

The application may be written to process a single XSAMS document, a fixed number of documents, any number of documents up to a fixed limit, or any number without limit. If the request contains the wrong number of documents, then the application must reject the request.

Service response

To enable successful use of XSAMS Processor web service both as user-accessible and scriptable service, following response scenario should be taken, independently of the type of incoming request:

- When responding to an HTTP request (GET or POST) for the primary result, the application must immediately respond with a 302 result code, providing in **Location** header a temporary resource URL from which the cached processing result must be downloaded later. Cached result provided in the `Location` header must be a semi-permanent URL to the result of XSAMS inputs processing. Cached document must not depend on browser cookies, session headers and any other parameters that are not embedded within the URL passed to the client in the `Location` header.

- While result is not ready because XSAMS document either is not fully downloaded or not yet processed, service must respond with status code 202 "Accepted" to all requests (GET or HEAD) to the temporary result URL. Get requests should receive an HTML document indicating XSAMS document download and processing progress.
- When download and processing of submitted document is done, all requests (GET or HEAD) to the temporary result URL must be responded with status code 200 "OK", with appropriate MIME-type header and, in case of a GET request, result of processing as a response body.

Other status codes can be returned both for requests to primary result and cached result URLs:

- Status code 400, "Bad Request", with a web page explaining the the failure in the body of the response. This code implies that the application operated correctly but the request was inappropriate; e.g. a request containing the wrong number of inputs; or the wrong type of inputs or a URL for an input that cannot be read. Requests receiving this response should not be repeated by the client.
- Status code 500, "Internal Server Error", 502, "Bad Gateway", 503 "Unavailable" or 504, "Gateway Timeout", indicating a problem inside the application. This code indicates that the request was correct but the application failed to process it. Requests receiving this response might be processed correctly at some later date.

Caching policy

XSAMS Processor is naturally obliged to cache either XSAMS documents, intermediate or final transformation result, or both. If caching XSAMS documents, final processing must be re-applied on every request to the result URL. If result is static page, service may cache only the result of the processing itself, immediately destroying incoming XSAMS documents after the completion of processing.

If processing is done in a streaming manner, only the result of processing may be cached.

Cache lifetime is defined by the XSAMS Processor developer/maintainer, it should be reasonably high for users to be able to come from the portal using the link to processing result, but not eternally since the disk capacity of the server running XSAMS Processor service is always limited.

1.2.3 VOSI capabilities

The VOSI capabilities are a single XML-document at the URL /capabilities relative to the root resource. A "capability" is an XML fragment describing a particular aspect of an application. The general rules for VOSI capabilities are defined by IVOA's VOSI standard.

For applications conforming to the current standard, there must be a capability following the schema `http://www.vamdc.org/xml/XSAMS-consumer/v1.0`. Such a capability provides two access URLs, one for the form (of type `WebBrowser`) and one for the primary result (of type `ParamHTTP`).

Capabilities must contain at least the following information:

- Interface URL for the form
- Interface URL for the primary result
- Version of this standard supported.
- Number of XSAMS inputs required.

The following code shows a sample capabilities-document, with the namespaces and locations of schema filled in:

```
<?xml version="1.0" encoding="UTF-8"?>

<cap:capabilities
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:cap="http://www.ivoa.net/xml/VOSICapabilities/v1.0"
  xmlns:vs="http://www.ivoa.net/xml/VODataService/v1.0"
  xmlns:vr="http://www.ivoa.net/xml/VOResource/v1.0"
  xmlns:xc="http://www.vamdc.org/xml/XSAMS-consumer/v1.0"
  xsi:schemaLocation="
  http://www.ivoa.net/xml/VOSICapabilities/v1.0 http://www.vamdc.org/downloads/xml/VOSI-capabilit
  http://www.ivoa.net/xml/XSAMS-consumer/v1.0 http://www.ivoa.net/xml/XSAMS-consumer/v1.0
  http://www.ivoa.net/xml/VOResource/v1.0 http://www.ivoa.net/xml/VOResource/v1.0
  http://www.ivoa.net/xml/VODataService/v1.0 http://www.ivoa.net/xml/VODataService/v1.0">

  <capability standardID="ivo://vamdc/std/XSAMS-consumer" xsi:type="xc:XsamsConsumer">
    <interface xsi:type="vr:WebBrowser">
       <accessURL>http://some.server/some/app</accessURL>
    </interface>
    <interface xsi:type="vs:ParamHTTP">
      <accessURL>http://some.server/some/app/service</accessURL>
      <resultType>text/html</resultType>
    </interface>
    <versionOfStandards>12.07</versionOfStandards>
    <versionOfSoftware>whatever</versionOfSoftware>
    <numberOfInputs>1-100</numberOfInputs>
  </capability>

</cap:capabilities>
```

1.2.4 VOSI availability

The VOSI availability is a single XML-document at the URL /availability relative to the root resource.

The general rules for VOSI availability are defined by IVOA's VOSI standard.

1.3 Registration

The application should be registered in the VAMDC registry. This makes it visible to generic UIs such as the VAMDC portal.

If registered, the registration-document type must be `{http://www.ivoa.net/xml/VOResource/v1.0}Service` as defined in the IVOA standard for registration. The registration must include the capability data taken from the VOSI-capabilities resource of the application, as detailed above.

Generic UIs will typically present users with a list of XSAMS processor web services. The `title` element of the application's registration-document should be suitable to distinguish the application in such a list: it should state explicitly but tersely what the application does. More detailed description may be provided within the `Description` element of `Contents` block. This description may be presented to the end user before he submits XSAMS documents to the processor service.

1.4 Closely-related applications

There may arise sets of applications with closely related functions; e.g., format converters for different output-formats. There is a natural instinct to combine these in one application where the outputs are distinguished by an extra parameter on the primary result that is specific to that combined application. This approach fails because the generic clients do not understand the special parameter. An application must not rely on custom parameters on the primary result.

Two methods are allowed for combining applications: multiple registrations and onward links from web pages.

Multiple registrations means that the complete set of web resources specified above is replicated for each kind of primary result, but the resources are served by the same web application. Each set of resources is registered separately and appears to clients as a separate application. E.g., for the format-converters, we might provide these resources:

```
converter/csv/
converter/csv/service
converter/csv/capabilities
converter/lamda/
converter/lamda/service
converter/lamda/capabilities
...
```

These URLs differ only in one path component that can be treated by underlying web application as a parameter.

Onward links means that the primary result is a web page and contains links to multiple, related results. This approach works only when the application is used interactively.

1.5 Example command-line client script

To test the operation of processor service, or to integrate an existing service into some software as a data source, the following script may be used:

```
#!/bin/bash
#XSAMS Processor service URL, ending with /service
PROCESSOR=$1
#URL to XSAMS document, either a VAMDC node output or just saved anywhere on internet
XSAMSURL=$2

LOCATION=`curl -v --get --data-urlencode "url=${XSAMSURL}" $PROCESSOR 2>&1 \
| grep Location: \
| sed -e 's/<\ Location:\ //g' \
| sed -e 's/[\n\r]//g'`

while curl --head --silent ${LOCATION} | grep -q 202
do
        echo  "waiting for result from ${LOCATION}" 1>&2
        sleep 1
done

curl --get --silent ${LOCATION}
```

The script accepts two parameters:

- First is XSAMS Processor URL, ending with /service . May need to be quoted.
- Second is URL to XSAMS document, either a VAMDC node output or just saved anywhere. May need to be quoted.

The downloaded processing result is sent to the standard output. This script may be integrated as an input to some scientific tool, if there exists an on-line Processor service that converts XSAMS into the format of this tool.

CHAPTER

TWO

CHANGES BETWEEN 12.07 AND 11.12 VERSIONS

1. **Name change from XSAMS Consumers to XSAMS Processors.** XML schema registration name stays unchanged
2. Omit the requirement of /info resource
3. Precise request scenario that must be implemented by the service application:
   - Service must respond with 302 redirect to processing requests
   - Service must cache documents or results
   - service must employ 202 result code for the page indicating the progress of download and processing of incoming documents
4. Exclude mention of employing URL parameters to distinguish closely-related applications
5. Provide example client shell script

CHAPTER THREE

REFERENCES

- International Virtual Observatory Alliance (IVOA) : http://www.ivoa.net
- VOResource: an XML Encoding Schema for Resource Metadata : http://www.ivoa.net/Documents/REC/ReR/VOResource-20080222.html
- Virtual Observatory Support Interfaces (VOSI) : http://www.ivoa.net/Documents/VOSI/20100311
- XSAMS consumer service registration schema : http://www.vamdc.org/xml/XSAMS-consumer/v1.0