Data Integration Using SOAP in the Virtual Solar Observatory

Richard S. Bogart[1], Alisdair Davey[2], George Dimitoglou[3], Joseph B. Gurman[3], Frank Hill[4], Piet Martens[2], Igor Suarez-Sola[4], Karen Q. Tian[1], Stephen Wampler[4]

1: Stanford University 2: Montana State University 3: NASA/GSFC 4: National Solar Observatory

The aim of the Virtual Solar Observatory (VSO) is the integration of diverse data archives relevant to the study of Solar Physics into a virtual collection providing common search and delivery services.

Technology

- XML (eXtensible Markup Language)
  - ➢ A mechanism to identify structure in a document
    - ➪ "keyword=value" + structure
    - ➪ user defined *arbitrary* tags, hence no semantics
    - ➪ text-based and platform-independent
  - ➢ Application
    - ➪ format for data exchange — widely accepted
    - ➪ format for data storage — ???, native XML databases exist
    - ➪ mid-ground — relational DB that provides XML view
      - ✔ XML query
      - ✔ Mapping between XML view and relational DB
- Web Services
  - ➢ What is it?
    - ➪ Service available over the network
    - ➪ Standardized XML messaging
    - ➪ Independent of platform and programming language
  - ➢ Protocol stack

Discovery	UDDI
Description	WSDL
XML messaging	XML-RPC,SOAP
Transport	HTTP,SMTP,FTP

  - ➢ Application-centric replacing human-centric (POST/GET)
  - ➢ Automation of the Web: service description, service registry
- SOAP (Simple Object Access Protocol)
  - ➢ What is it?
    - ➪ RPC (Remote Procedure Call) mechanism
    - ➪ HTTP as transport
    - ➪ Client-server messaging encoded in XML documents.
      - ✔ Independent of platform and programming language
  - ➢ Three major components
    - ➪ Data encapsulation specs: XML envelope
    - ➪ Data encoding rules: agreed-upon data types
    - ➪ RPC conventions: one- or two-way messaging
  - ➢ Implementation available for Java, Perl, Python, etc.
- Perl SOAP::Lite Module
  - ➢ Written by Paul Kulchenko
  - ➢ Interface

Client

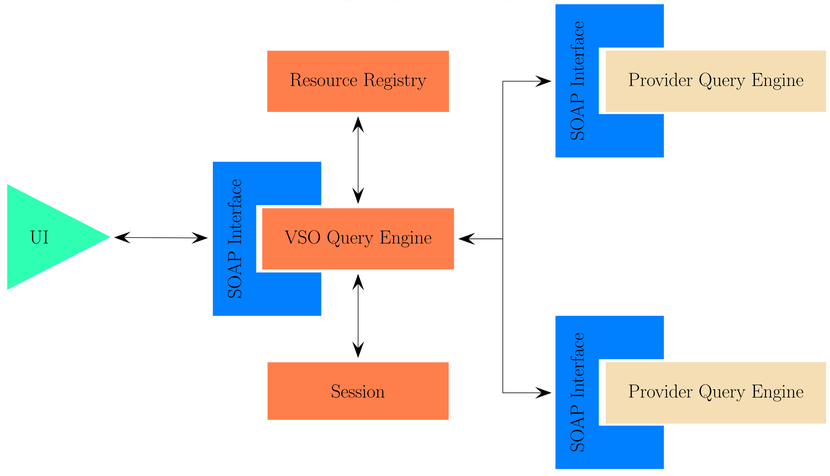
```
use SOAP::Lite;
$soap = SOAP::Lite
-> uri('http://vso.stanford.edu/MDI')
-> proxy('http://vso.stanford.edu/mdi.cgi');

$result = $soap->Query();
```

Server

```
use SOAP::Transport::HTTP;
SOAP::Transport::HTTP::CGI
-> dispatch_to('MDI')
-> handle;
package MDI;
sub Query { ... }
```

  - ➢ Error handling mechanism
    - ➪ Timeout
    - ➪ Reason of failure: standard and custom-defined
- gSOAP
  - ➢ Written by Robert van Engelen et al.
  - ➢ C/C++ SOAP API: compiler and library
  - ➢ Great potential for grid computing

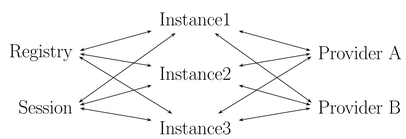
VSO Architecture



VSO Data Model (DM)

- Why? The DM defines a unified world view and is therefore free from any provider idiosyncrasies.
- Wait, must providers change their data storage? Not at all. The DM is almost completely virtual, i.e., there is no dataset stored according to the DM, only some metadata is organized according to the DM.
- Where is the DM used? Everywhere.
  - ➢ VSO user poses query in terms of the DM and obtains results also in terms of the DM.
  - ➢ Individual provider describes its data holding in terms of the DM in the resource registry.
  - ➢ Our implementation uses the DM as its internal data structure.
- Sounds good, but what's the cost? Translation.
  - ➢ Search requests: translate from the VSO DM to providers' specifications.
  - ➢ Search results: translate from providers' specifications to the VSO DM.

VSO Instance



- An instance is a clone of the VSO.
- It is lightweight and runs on local machines.
- VSO becomes a distributed system with no performance bottleneck.
- VSO requires centralized storage for the registry and session.

VSO UI and Data Export

- We provide sample UIs using web forms.
- But anyone can write his or her own UIs using the VSO API.
  - ➢ VSO API is currently for perl only. The root of the problem lies in the limited interoperability among various SOAP implementations. The problem aggravates when complex perl data structures are used. To alleviate this problem, we plan to simplify our data structure in future releases of the VSO API.
- Our data export taps into providers' existing export mechanisms, therefore bypassing the VSO.
  - ➢ Pros: save bandwidth
  - ➢ Cons: we can only export a single dataset at a time.
    Ideally one would like to bundle selected datasets together.

VSO Resource Registry

- The resource registry contains information on "What" and "How":
  - ➢ "What" datasets a provider has. The VSO query engine uses this information to decide to which provider to send query requests.
  - ➢ "How" to query and export a provider's datasets.
- An example of registry entry describing Stanford's MDI data holdings.
  Source: SOHO
  Instrument: MDI
  Observables: Dopplergram, Magnetogram LOS, Continuum Intensity, Line Depth
  Time Range: 1996.01.03 → *present*
- An example of registry entry describing how to access Stanford's data holdings.
  Query interface:
  Server: `l5-m5.stanford.edu`
  URI: `http://l5-m5.stanford.edu/SHAI`
  Proxy: `http://l5-m5.stanford.edu/cgi-bin/soap/shai.cgi`
  Export interface:
  Method: GET
  URL: `http://flap.stanford.edu/cgi-bin/export/expvrfy`

VSO Query Engine

- What does it do?
  - ➢ It queries the resource registry to decide to which providers to send query requests.
  - ➢ It dispatches queries to relevant providers.
  - ➢ It waits for responses from providers. In case of failure, a provider either times out or returns an error.
  - ➢ It assembles query results from providers.
- How? Query descriptions are given in terms of the VSO DM: $\{d_1, d_2, \cdots\}$
  - ➢ Relations among d_i's are "and".
  - ➢ We treat $d_i = null$ as don't-care.

  This defines a generic query interface. In contrast to specialized query interfaces, such as ObservableSearch and TimeSearch, our solution does not depend on complex algorithms for query construction.
- Finally something fancy: a "time join" query
  - ➢ Example: find (SHA SOHO) MDI Magnetogram that is within 1 hour of (SHA SOHO) MDI Dopplergram between 2001.10.30 00:00 and 2001.10.30 23:59
  - ➢ Even more interesting queries can be formulated (and answered) when combined with the Solar Event Catalog. We have downloaded these catalogs locally and created web services to support time queries on these catalogs. Currently our catalogs include:
    - ➪ Active Region Number
    - ➪ Yohkoh Flare, 1991/10/01-2001/12/14
    - ➪ RHESSI Flare, 2002-present
    - ➪ Lasco CME, 1996-2002

VSO Session

- The goal of logging a session is to collect usage statistics and to be able to repeat saved queries.
- We are faced with a multitude of design options.
  - ➢ How is a VSO session defined?
    A VSO session is defined outside the VSO query engine.
  - ➢ What are session parameters? Anything!
    Query inputs, intermediate results, and query results.
  - ➢ Can we really repeat a query? Hard because things change over time:
    - ➪ Provider's inventory: e.g., new data products.
    - ➪ VSO resource registry: reflecting changes in existing providers or addition of new providers joining the VSO.