Building a Virtual Solar Observatory: Lessons Learned

R. S. Bogart[1], K. Q. Tian[1], A. Davey[2], G. Dimitoglou[3], J. B. Gurman[3], F. Hill[4], J. A. Hourclé[3], P. C. H. Martens[5], I. Suárez-Solá[4], S. Wampler[4], K. Yoshimura[5]

1: Stanford University 2: Southwest Research Institute 3: NASA/GSFC 4: National Solar Observatory 5: Montana State University

Distributed vs Centralized Service

The design of VSO is intrinsically a distributed problem, since the VSO serves as a broker between users and diverse, distributed data sources. VSO strives to remove the distinctions among the data providers from the user's viewpoint, however, and to serve as a centralized clearing house for browsing solar data archives and selecting data records for inspection, download, and/or analysis.

- Unified views through the creation of the VSO data model

  The VSO data model is the first step towards unifying the view of data from various sources. At each data provider, a service must exist to translate to and from the VSO data model. The translated metadata can then be treated in a unified manner by the VSO server as long as they conform to the VSO data model. As a consequence,

  - ➢ the VSO server provides a uniform interface for querying data from a wide range of data providers – single user experience
  - ➢ changes at the data provider will be transparent to the user through the VSO server – source abstraction
  - ➢ any changes at the provider end only require changes in the provider translation service, leaving VSO servers untouched

- Incorporating providers who do not have a database

  Besides providing translation assistance for archives that maintain relational databases of their data holdings, we also provide a service for those providers whose datasets are simply organized in ftp/http directory trees. We found it is easiest to incorporate such providers by offering a proxy server for that provider. Not all such directory trees are well-designed for automated searching. To optimize performance, we periodically crawl the directory trees and cache gathered information locally. It is important to note that we only cache meta-data, e.g., the data file path names, not the actual data. Also, the crawling and caching are performed by only the designated proxy server, not the VSO instances.

- Sharing the load

  To avoid bottlenecks and single-point failures, and to distribute the potential load, VSO utilizes multiple VSO servers, or instances.

  - ➢ From a user's perspective, there is only a single point of entry. The user's queries are directed to a (randomly) selected VSO instance.
  - ➢ Since VSO only acts as a broker to the data providers, each instance does not store any significant amount of information. The instances therefore occupy a small footprint and consume a small amount of resources – instances are lightweight by design.
  - ➢ VSO instances run identical code; they can and do run on different platforms (Solaris, Linux, MacOSX).
  - ➢ The price for the lack of centralized control is the loss of some functionality, such as monitoring and logging, or at least greater complexity.

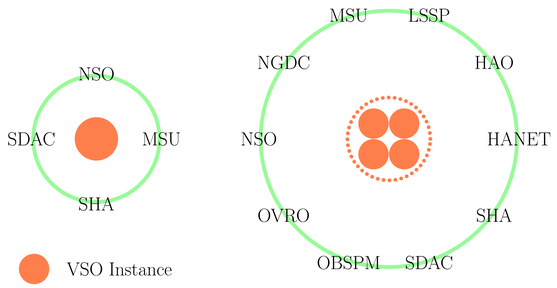
- Catalog services

  Catalog information (e.g. lists of flares, CME's, and other events) is collected from various sources and managed by a MySQL database.

  - ➢ This is a supplementary service; the data are presented in a format similar to the metadata results of data archive record searches – integration with other VSO services remains to be done.

What a "Small-Box" Can't Do

In our "small-box" design, the actual delivery of data bypasses a VSO server. In doing so, we not only reduce network traffic but also make the VSO servers lightweight: no temporary storage is needed at a VSO server. The drawback of this approach is that we miss opportunities to package the data. To give users better information about the datasets, we will provide information about the availability of the datasets, e.g., on-line, near-line, or off-line.



VSO version 0.5β — Current VSO version 1.2

Application Programming Interface (API)

The VSO web services are described by a WSDL (Web Service Description Language) document. From our experience with SOAP::Lite, gSOAP, Axis,and wscompile, different language code generators have varying degree of tolerance for an API WSDL specification. To reduce the overhead of SOAP encoding, we are considering moving from RPC/encoded to document/literal in WSDL. Support for document/literal is not yet widespread, however.

VSO Data Model

In its early versions, the data model looked much like a data dictionary; it was intended to cover as many dataset descriptions as possible. We moved away from the data dictionary approach, and adopted a much simpler data model. This data model defines a single world view out of many possible ways to describe a dataset. Instead of enumerating half a dozen formats for time, for example, we now define a single format for VSO time representation. Similarly, we have chosen three basic units (wavelength in Ångström; energy in keV, and frequency in GHz) to describe observational spectral range. Translation to and from the VSO data model is handled at the data provider. The current VSO data model is version 1.8 (`http://vso.stanford.edu/datamodel.html`). This data model still has significant limitations:

- It is primarily designed to describe Level 0-1 image and spectral data; it lacks descriptions appropriate to higher level or more complex data products, such as helioseismic data.
- It does not contain summary/overview type of information.
- Candidates for additional information, e.g. data format and data processing information, have not yet been incorporated – addition of fields requires evaluation of the impact on providers and the utility.
- The data model so far lacks useful mechanisms for dealing with dataset-specific information.

Performance

The performance of the VSO is primarily manifested in its query response time. This response time can be tuned at various level of operation:

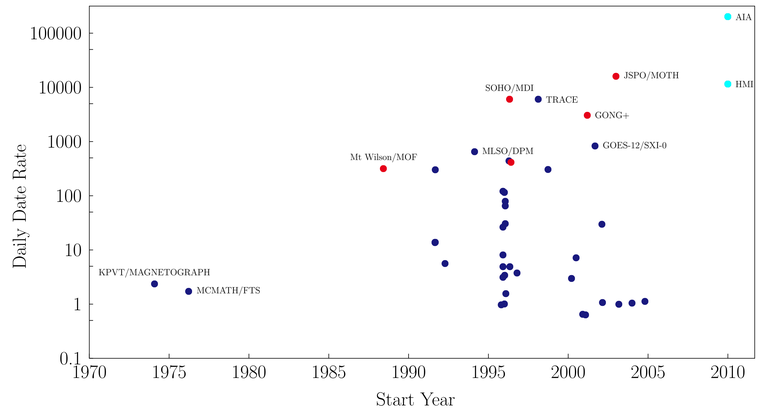
- The http servers can be enhanced with mod_perl or fastcgi.
- The VSO server dispatches requests to data providers in parallel.
- Data providers are encouraged to limit the number of query returns. This turns out to be a major factor in overall response time.
- The WSDL interface would benefit from a document/literal style.

.

Diverse Data Rates

The primary (and required) method for data search is time-based. The rate at which data records are accumulated in the various data sets supplied by the VSO data providers varies by about four orders of magnitude. Rates for new data will increase by another order of magnitude with the launch of the Solar Dynamics Observatory. Conversely, there is significant activity in incorporating long historical data records with very low data rates into digital archives. The disparity of data rates poses a unique challenge in the presentation of query options and results within a single framework.

We are considering solutions from the perspectives of the VSO data model, the VSO registry, and the data providers. One provider (the Stanford Helioseismology Archive) already addresses the problem by packaging its fundamental data "records" into collections (hourly or daily) which are actually returned as the provider's individual data records. (The data rates for those datasets in the accompanying chart are shown in orange; they reflect estimates of the actual rate of images, not packaged data records, which are typically two orders of magnitude lower.) Such packaging is not reflected in the VSO data model, so the meaning of the term data record is confused. As that scheme has not been adopted elsewhere, and SHA is planning to move away from it in future, it is likely that solutions to the problem must be sought in the VSO data model and registry design.



User Interface (UI) Design

- Design issues:
  - ➢ Should functionality be implemented at the UI or the backend?
    User interface design reflects a balanced distribution of functionality between the UI and backend services. For example, nickname translation is handled at the UI level.
  - ➢ How much information to display?
- Test issue:
  Browser compatibility test is time-consuming.
- Desirable features:
  - ➢ Cart ID for publication
    The idea is to assign a unique identifier to dataset(s) acquired through the VSO during a session. Upon receiving such IDs at a future time, the VSO should be able to return the *same* datasets.
  - ➢ "More" button
    If the number of returns is smaller than the number of records found, we need a way for the user to get the rest of the search results, subject to further selection. This would require that provider limits be "soft."