# SITOOLS:
# ENHANCED USE OF LABORATORY SERVICES AND DATA
# - ARCHITECTURE AND FIRST FEEDBACK -

Thierry Levoir[1]

[1] CNES, 18, Avenue E.Belin, 31401 Toulouse Cedex, France
Thierry.Levoir@cnes.fr

**Abstract.** SITools, is an « open-source » Framework, to be installed in scientific laboratories and managed by them. It allows access to services and data that were previously inaccessible or accessible through specific system hard to maintain. SITools aims to reduce the number of such specific systems, and federate the effort of the laboratories.

## 1 Introduction

SITools is a technical layer, based on a concept of services interconnected via a Web Services virtual bus and accessed by one or more client applications (SITOOLS initially includes one client application – a Web interface -).

Five types of basic services have been implemented (On-line catalog, Off-line catalog, Repository, Command and User Space). But the most important, SITools is able to ingest and to use very easily added-value services (AVSs) as plug-in. AVSs are independent software programs, interfacing with the system to provide new features (graphs, data-mining, 3D navigation, etc.).

All the services may use the already installed facilities (database, services, …).

The Web Services bus gives a native interoperability. It may run locally, on a single machine, or be distributed over several machines or even several laboratories. It enables the creation of a distributed data centre.

All the services have to be configured (to access data, to specialize the GUI, …) to create what we call “a SITools Instance” i.e. a data access system.

## 2 SITools components

SITOOLS is broken down into services and into client applications accessing these services. SITOOLS initially includes one client application, but any project using the tools can develop another client application using the SITOOLS Services.

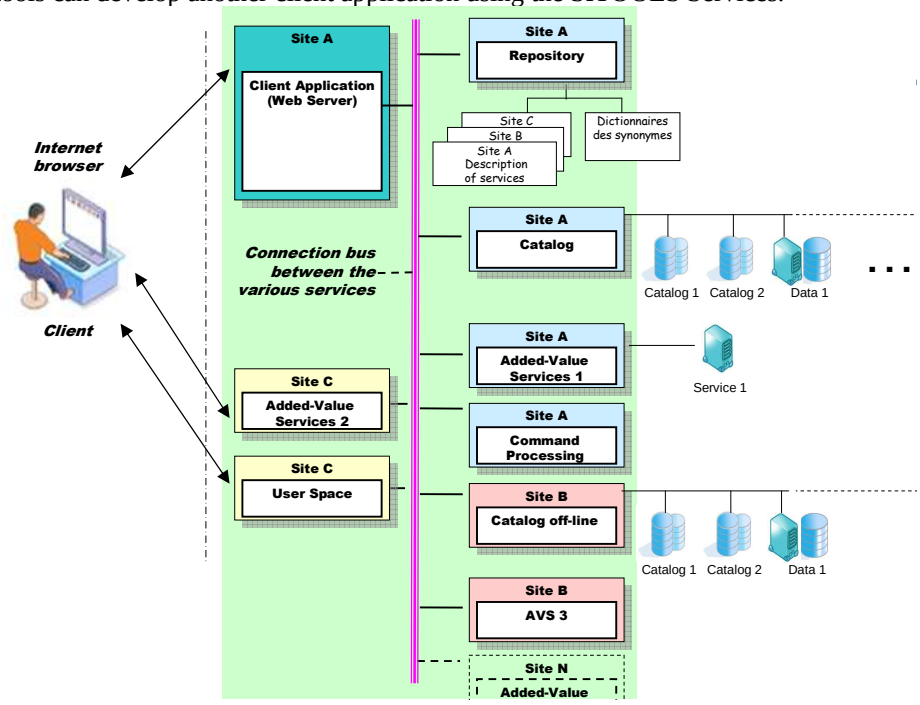


**Fig. 1.** General architecture: Diagram of the services software bus

### 2.1 Basic services

SITOOLS defines 5 types of basic service which can be "plugged in" to the SITOOLS virtual bus :

- On-line catalog : provides access to metadata and data accessible on-line.
- Off-line catalog : identical to the On-Line Catalog Service with batch access to data.
- Repository : links all accessible services to form a global system; it is the nerve center of SITOOLS, managing its global configuration :
  - Catalog management and synonym association (providing the link between metadata with the same meaning, but not the same syntax for different data sets),
  - Data set grouping management (for a given subject, for example).

    - Management of added-value services, access, dynamic parameters.
    - Management of user rights : rights to catalogs, services.
- Command Processing : end-to-end management of user commands,
- User Space : manages the command and processing result space, where the user will find command results.

A service type is characterized only by the interface to this service. SITOOLS will supply some "instances", but the laboratories can create others using different technologies.

## 2.2 Added-Value Services

The added-value services (AVSs) are independent software programs, interfacing with the system to provide new features (graphs, data-mining, 3D navigation, etc.). An AVS has no specific boundary. It may be an independent application, a SITOOLS services client application, a web or Windows/Linux application, etc.

Management of the AVSs in SITOOLS consists in

- defining the various types of operation of the AVSs and therefore knowing how to call them (call method, type, call parameters, etc.),
- connecting them with the information model elements (collections, data or browse sets), and therefore proposing them whenever useful to the user,
- managing access rights and user quotas,
- limiting the need of adding service types (a new service type involves significant modifications - repository, client applications, other used services, etc.).

## 2.3 Client Application

SITOOLS proposes a client application. It is a Web application based on Struts for implementation of the MVC2 model. The MMI layer is in HTML 4 and CSS 2 generated in JSP.

This webapp, when running, creates what we call “super catalog” (see Fig. 2) :

- It handles interaction with the catalog services authorized for a user.
- It dynamically prepares the display of the list of search criteria, integrating any synonym dictionaries.

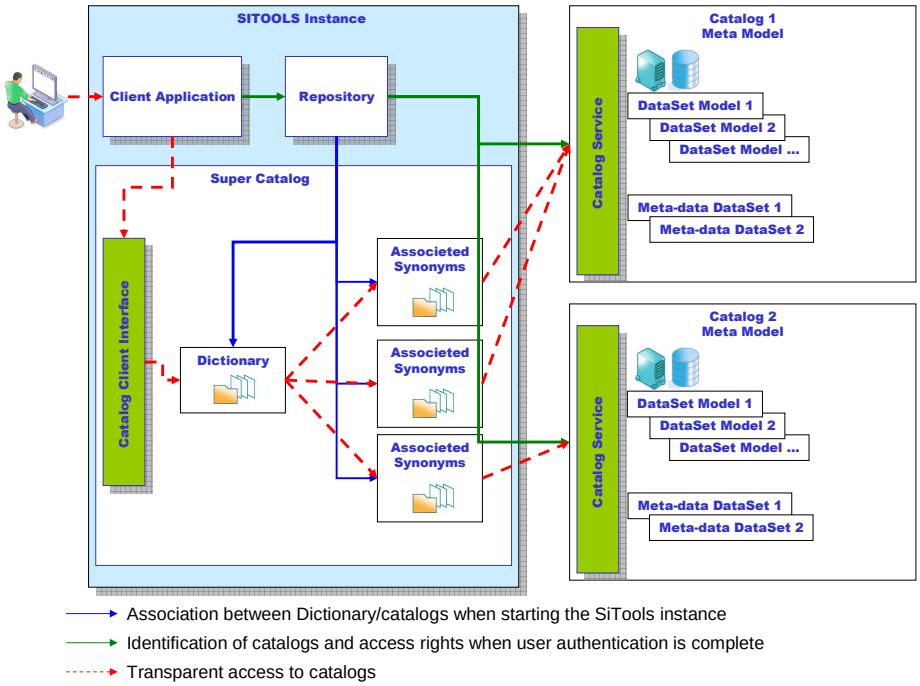


**Fig. 2.** Super catalog, dictionary of synonyms

## 3 Users

SITOOLS is designed for 2 types of user :

- The end user, the scientist who consults the final system set up to look for and download useful data. He/she will be referred to as the "user" in the remainder of this document.
- The manager, the entity (scientific laboratory, organization, etc.) which will set up and run the system to provide access to its data.

## 4 Utilisation

To create a SITools Instance, there are 2 main steps to do, catalogs configuration and repository configuration.

## 4.1 Catalog services

### 4.1.1 Data

The guiding principle of SITOOLS is to give access to data. SITools uses a data base to refer and possibly to store the data.

In the first case, the data are in files described by metadata. It is those metadata which are in database. They will make it possible to the end-user to make a research by criterion.

In the second case the data are directly in database. They then, will also be used as metadata for research.

### 4.1.2 One dataset, one table

SITools considers a dataset as a unit of homogeneous data, i.e. data generally coming from the same source with the same level of treatment, with identical attributes and criteria.

To a dataset, a table in the catalogue will correspond. For each new dataset inserted in a SITools instance, a new table will be created.

So that SITools can reach it and offer the access to them automatically, this table is described in SITools. The dataset is initially declared in a technical table of SITools named DataSets. Then each field of the dataset (column of the table of the set) is described using the table attribute:

**Table 1.** attribut

| Colonne | Type | Description |
|---|---|---|
| dataset_name | Varchar | Data set name |
| name | Varchar | Attribute name |
| label | Varchar | Displayed name |
| tooltip | Varchar | More information to be displayed |
| type | Varchar | Attribute Type (see below) |
| class | Varchar | Attribute class (see below) |
| size | Int | size (for display information) |
| keyindex | Int | Indicates if it is a key |
| criterion | Int | Indicates if this attribute is to be used as a criteria for the end user. |
| display | Int | Indicates if this attribute is to be displayed to the end user. |
| advanced | Int | Indicates if this attribute may be displayed if the end user asks for it. |
| mandatory | Int | Indicates if the attribute is mandatory |
| updatable | Int | Indicates if the attribute is updatable |
| default_value | Varchar | |
| min_value | Varchar | |
| max_value | Varchar | |

| | |
|---|---|
| comment | Varchar |

The main information to give to the catalog is the type and the class. It is what will make possible for SITools to know how to manage this information, how to present it like criteria and how to present it as results.

### 4.1.3 Type

The concept of the type addresses the traditional basic types, but also a certain number of others types, which make possible for SITools to manage the data of a dataset in a single table, while offering the power of a relational database.

SITools thus will automatically create additional tables according to the type of the data.

Example: When one declares fields of the type "multi _..." that corresponds in the table of the dataset to a field "string" which contains several values (numerical, string, ...) separated by the separator ";". It is obvious that with such a string, it is impossible to make an elaborate request (like Greater than). SITools thus will automatically create the tables necessary to answer these requests efficiently.

Basic types: boolean, Float, Geometry, string, int, length, timestamp

Complex types:

- ids_child Idents of the child data
- ids_parent Idents of the parent data
- multi_string Multiple values string
- multi_int Multiple values integer
- multi_long Multiple values big integer
- multi_float Multiple values decimal
- multi_date Multiple values date
- multi_timestamp Multiple values date time
- multi_inter_int Multiple values of interval integer
- multi_inter_long Multiple values of interval big integer
- multi_inter_float Multiple values of interval decimal
- multi_inter_date Multiple values of interval date
- multi_inter_timestamp Multiple values of interval date time
- serial Auto-increment id type

### 4.1.4 Class

The concept of class makes it possible to specialize a type to give it a specific behavior. It adds semantics to the type which is purely syntactic. For example for a type "string", we can use the class "URL" which allows the customer application to present it in the form of URL.

With each class is associated a template in the system (in the form of jsp) defining the presentation of the attribute as well in its criterion representation (for input in a user search) as in its result representation.

| | |
|---|---|
| applet_carto | Toponym fields (point & zone) with Map Selector |
| checkbox | Display True/False/All choice |

| | |
|---|---|
| date | Full date (dd-mm-yyyy) |
| date_short | Short date (dd-mm-yy) |
| date_time | full date (dd-mm-yyyy) with time (hh:mm:ss) |
| fichier_dav | HTTP link to files stored into DAV |
| histogram | Link which open a popup that display the histogram |
| ids_asso | Idents of the parent and child data |
| image | Image to be displayed |
| liste_choix | Single Selection list |
| liste_choix_multi | Multiple Selection list |
| liste_choix_dyn | Single Selection list filled dynamically |
| liste_choix_multi_dyn | Multiple Selection list filled dynamically |
| multiple_criterion | Multiple values for criterion field only |
| multiple_values | Multiple values field |
| multiple_intervalle | Multiple values field |
| number_float | Float number values field |
| number_integer | Long or integer values field |
| password | Password field |
| quicklook | Display quicklook with a link to data |
| text | Text field (default class) |
| url | HTTP link |
| url_filedata | HTTP link to the external data (used by zip) |
| zone_texte | Text field with vertical scrollbar |

## 4.2 Le service repository

Through configuration XML files, the manager will connect all the elements of the SITools instance which he sets up.

- He will declare all the catalogs connected. SITools then will determine alone the common criteria of research of the various datasets.
- He will declare the dictionaries of synonyms making it possible to establish the link between attributes of different datasets, having the same meaning and to even make little more with convertor features (for instance, conversion of °C to °F).

```
  <catalogs>
    <catalogue name="catalog">
      <adress>

http://<server>:<port>/<catalog_name>/services/ServiceCatalogue
      </adress>
      <synonym>
          http://<server>:<port>/synonymes/synonyme.xml
```

```
      </synonym>
    </catalog>
     ...
  </catalogs>
```

File repository.xml : catalog declaration

```
<synonym word="startDate">
  <synonym>beginDate</synonym>
    <convertisseur
classe="com.cnes.sitools.webapp.supercat.StandardConverter"/>
  <operator>GE</operator>
</synonym>
```

File synonyme.xml : example date synonym associated with a converter to create a special Greater or Equal criteria

- he will declare groups of datasets, making it possible to then offer to the end-user, graphs of navigation, for example by mission/experiment.

```
<graphs>
    <!-- graph simple -->
    <graph id="<NAME_GRAPH>" search="<GRAPH_SEARCH>">
        <label><LABEL_GRAPH></label>
        <tooltip><TOOLTIP_GRAPH></tooltip>
        <dataset catalog="<NAME_CATALOG>">
          <NAME_DATASET>
        </dataset>
    </graph>
    <!-- Noeud de graph complexe -->
    <graph id="<NAME_GRAPH>" search="<GRAPH_SEARCH>">
      <label><LABEL_GRAPH></label>
      <tooltip><TOOLTIP_GRAPH></tooltip>
      <graph id="<NAME_GRAPH>" search="<GRAPH_SEARCH>">
         <label><LABEL_GRAPH></label>
         <tooltip><TOOLTIP_GRAPH></tooltip>
         <dataset catalog="<NAME_CATALOG>">
             <NAME_DATASET>
         </dataset>
     </graph>
     <dataset catalog="<NAME_CATALOG>">
          <NAME_DATASET>
     </dataset>
    </graph>
  </graphs>
```

File repository.xml : graph declaration

- He will declare all the user rights
- He will finally declare all the Added Values Services.

```
<service nom="zip" internal="false" batch="false" syn-
chrone="true" precopieEU="false">
        <affichage>Extraction (zip)</affichage>
        <adress>http://localhost:8080/zip/servlet.zip</adress>
```

```
        <picto>/images/picto_zip.gif</picto>
</service>
```

File repository.xml : very sample (zip) AVS declaration

## 5 Experimentation

SITools V2 (full version) was released in september 2005. As it was created as an R&D product, it has to be fully tested and qualified by some laboratories before being available.

SITools is currently tested by 3 CNRS laboratories and by one project at CNES. Data accessed through those SITools instances are very heterogeneous.

So far, one of the main issues is to install all the needed production tools on a linux platform. All the Framework was developed (or integrated) with Open Source tools exclusively (Apache – Tomcat - MySQL / PostGres - Struts - Axis - Log4J - …) and the way to configure them may change from on linux distribution to another.

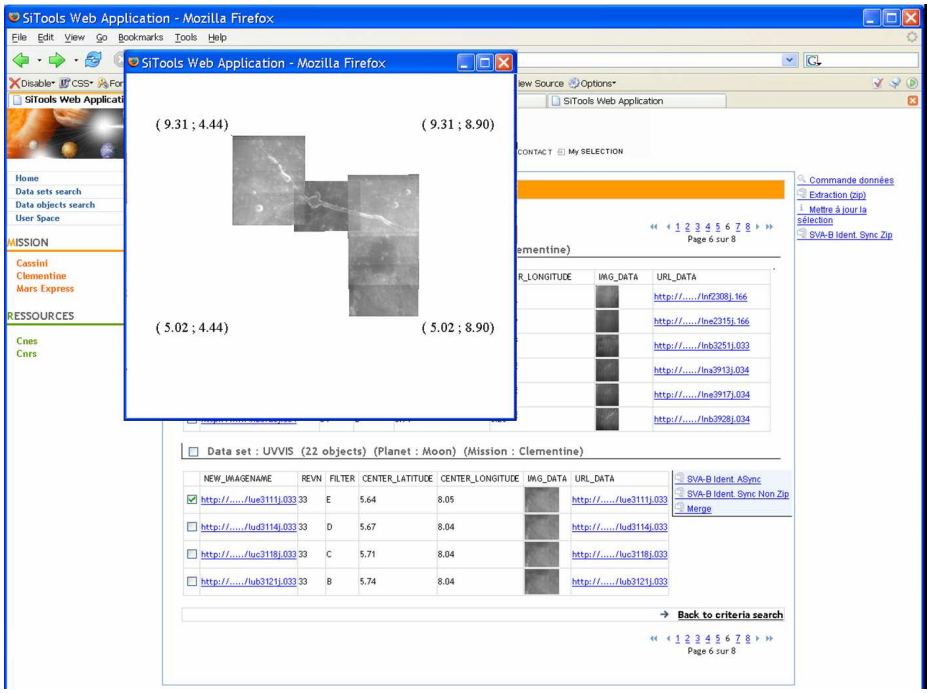


**Fig. 3.** Planets SItools Instance with the use of “Merge” AVS

### 5.1 SITools & Virtual Observatory[1]

During the experimentation we have understood that SITools must be compliant with the VO requirements.

As SITools design was done in VO logical concepts (from the interoperability point of view), it's not a big challenge to integrate VO capabilities. All the services are Web-services, catalogs are linked through synonyms, which could be UCDs, SITools has also the capability to use an added value service to handle VO protocol queries (SIAP, SSAP, VOQL, …) and to transform results in VOtable format.

## 6 Conclusion

SITools is born. The framework is available. Let's wait the end of the experimentation to have a reliable product, but it's almost ready.

It's not the end of the project; maybe we have only done the easiest part.

We have now to help people to use it, to create new functions, new Added Values Services, new skins, … and to share everything.

[1] VO : «an enabling and coordinating entity to foster the development of tools, protocols, and collaborations necessary to realize the full scientific potential of astronomical data bases in the coming decade», NVO White Paper, juin 2000. more information : http://www.ivoa.net