CMR End-To-End Services Study (Task 25)

EED2-TP-025

Technical Paper
November 2016

Prepared Under Contract NNG15HZ39C

RESPONSIBLE OFFICE

11/14/16

Dana Shum, Task 25 Lead — Date
EOSDIS Evolution and Development 2 (EED-2) Contract

RESPONSIBLE AUTHOR

S. D. Cantell — 11/14/2016

Simon Cantrell, CMR Systems Engineer — Date
EOSDIS Evolution and Development 2 (EED-2) Contract

Raytheon Company
Riverdale, Maryland

Contents

1. Introduction 1
2. Statement of the Problem 2
3. Vision for the Future 2
4. Workflow Analysis 4
5. High-Level Architecture 19
6. Identify Candidate Services/Tools for Re-use 26
7. Define a minimum set of services to recommend to the DAACs. 30
8. Future Work 31
References 32
Abbreviations 33
Appendix A Tools, Services Candidates for Reuse 34
Appendix B Service Definitions 35

1. Introduction

The purpose of this document is to analyze the current capabilities of EOSDIS' subsetting services and propose an improved approach to data services which will enable a simplified end-to-end user experience for discovering and subsetting data. To do so, we will examine the current end-to-end user experience, analyze the shortcomings of our offerings today, investigate how our current architecture and service offerings could change to improve the user experience and recommend near-term next steps to move towards implementation of this solution.

There are then two high level goals for this exercise:

1) Minimize the number of clicks, or required inputs in order to execute a data search and to subsequently access data.

2) Maximize the ability to expose the subsetting capabilities which are available in the EOSDIS system.

Combined, we can then state the requirement as: "provide a specification of the end-to-end user experience needed to minimize the required input specification, while maximizing the capabilities, provided by services, which are required to perform the desired data transformations, prior to downloading a file containing subsetted data".

For the purposes of this document, the following definitions will be used:

<u>Simple Subsetting</u>:

- Variable subsetting - we can subset the data set by variable, where only the variable field within the data set is returned.
- Spatial subsetting – data is resampled across a geographic spatial region, and this results in a subsetted data set/variable (which may include the vertical dimension in spatial subsetting).
- Temporal subsetting – data is resampled across a time-range. Temporal subsetting can include taking samples at discrete times, daily over a specified period, or computing daily averages over a specified period.

<u>Complex Subsetting:</u>

- Regridding – A data set may be resampled on a different gridding scheme, i.e. by moving from a native scheme to a new grid scheme. To achieve this an interpolation method often needs to be employed.
- Reprojection – A data set may be represented on a projection scheme, e.g. Polar Stereographic, Mercator, etc.
- Resampling – A data set may be interpolated using a resampling scheme, e.g. Bilinear Interpolation, Bicubic Interpolation, Distance-weighted average resampling, Nearest neighbor resampling.

- Mosaicing – A data set may be joined to another data set using a mosaicing/stitching scheme – used where spatial area selected is covered by data layers contained in separate data sets.

2. Statement of the Problem

Many tools and services exist today across EOSDIS and the DAACs. Some are stand-alone clients, some are tools which rely on services, some are services which require specialized UIs, often specific to particular collections. Since these have all evolved independently, there is very little consistency of the tools available to perform subsetting, and there is very little consistency with services across the DAACs. Because of this, end users still spend vast amounts of time attempting to find the data they want and transform it into their desired format, configuration or projection. For example, we learned of one case where graduate students using NSIDC DAAC science granules spent so much time downloading and post-processing the data, that their summer programs ended before they were truly able to perform "actual science" with the retrieved data granules. This has been re-validated recently by the "NASA DAAC User Research" study conducted by Blink in September of 2016 which interviewed over 90 users and found that regardless of their experience level, they all had difficulty in finding the right data set and preparing the data to get it in the desired format.

Some key contributors to this problem are: that there is disconnect between subsetting requirements and the desired end-to-end user experience, lack of adherence to a service standard(s) and an inability for users and clients to easily determine which subsetting capabilities are available for which data products.

3. Vision for the Future

This work will begin with the following end-to-end-user experience vision statement, which has been kindly provided by ESDIS (Courtesy of Chris Lynnes). This will be referred to as the Minimalist Vision throughout this paper. This is described in the following way:

1) User is required to provide a minimum of input for service invocation[1].
    a. Provide defaults for everything where possible.
    b. Include defaults for spatial, temporal – if a user has previously entered subsetting specification.
2) Optimization for the majority of the user population, not the totality.
    a. Make the most popular selection in an option the default.
3) Use the user's jargon, i.e. options not services; files not granules.

The emphasis, therefore, is on minimizing the number of user selections, clicks or keystrokes needed to completely define the search, refine the search and subsequently download the subsetted data. The user input captured in a client should be rich enough to enable the full end-to-end

[1] *"Perfection is achieved, not when there is nothing left to add, but when there is nothing left to take away."* - ***A. de Saint-Exupery***

processing of the data, including any simple or complex subsetting requests, resulting in a file download which more closely meets the user specification.

A layout of this "idealized" workflow for the Consistent Subsetting Interface, is given below:

I) User is required to Search for (or select) Measurement/Variables

Sub Option 1 - User can search for a Measurement/Variable by name using a search box.
Sub Option 2 - Alternatively, can select a Measurement/Variable from a facet.

II) User is required to Identify Constraints

Sub Option 1 - Variable Constraints.
Sub Option 2 - Spatial Constraints.
Sub Option 3 - Temporal Constraints.

An example of a measurement is "methane", and the corresponding variables are "methane total column (daytime/ascending)", or "methane mole fraction in air (nighttime/descending)".

Another example of a measurement is "aerosols", and the corresponding variables are "aerosol optical depth 388nm", "aerosol optical depth 500nm", "aerosol single scattering albedo 442nm", or "aerosol angstrom exponent 510/670nm (ocean only)".

III) User is required to Identify Options (Attributes)

Variant 1 - Use Default Options.
Variant 2- Specify Options.

Sub Option 1 – File Format.
Sub Option 2 – Units.
Sub Option 3 – Resolution.
Sub Option 4 – Resampling.
Sub Option 5 – Reprojection.
Sub Option 6 – Regridding.
Sub Option 7 – Mosaicing.
Sub Option 8 - Time Options.
Sub Option 9 - File Restructure.

An example of identifying options is regridding of a SMAP L3 data set from 36 km EASE-Grid to a 10km grid, using a Polar Stereographic projection, and the choice to receive the data in a geoTIFF file format.

IV) Preview and refine selection

V) User requests download using Get Data button

Variant 1 - Synchronous, i.e. data is immediately available for download. (Preferred)
Variant 2 - Asynchronous, i.e. data becomes available for download following a processing delay.

We understand that to implement this vision, services will play no small part and are destined to become part of the solution towards moving the data into the final form desired, whether this is to perform spatial, temporal subsetting, or by optionally executing the data transformations on the server-side, hence saving the user time and effort following download of the data.

4. Workflow Analysis

Next, we took a small sample of search/subsetting tools and determine the workflows for these tools. The first tool we analyzed was the Simple Subsetting Wizard (SSW). The next candidate was the Earthdata Search Client (EDSC). Finally, we considered one of the custom subsetting tools which is provided by ASDC, the Calipso Search and Subsetting Client Web Application.

Simple Subset Wizard Workflow

Intro to Simple Subset Wizard

Simple Subset Wizard is a straightforward web-based application UI, with the ability to allow users to enter search terms, and constrain the results sets by temporal and spatial constraints. It has a basic map interface, which allows the specification of a bounding box to define the region of interest, or via North, West, East, South latitude and longitude values. The temporal constraints allow a time range to be entered directly, or selected from a calendar. A desirable feature of this subsetting tool, it that it has information on multiple subsetters and related tools for over 10 DAACs. Some of these subsetting requests are handled by services, and the tool has two key XML configuration files that contain all the information needed: the data set (collection) name, the parameters (measurement) names and the variable field names, and the service URL from which DAAC the subsetting service is available.

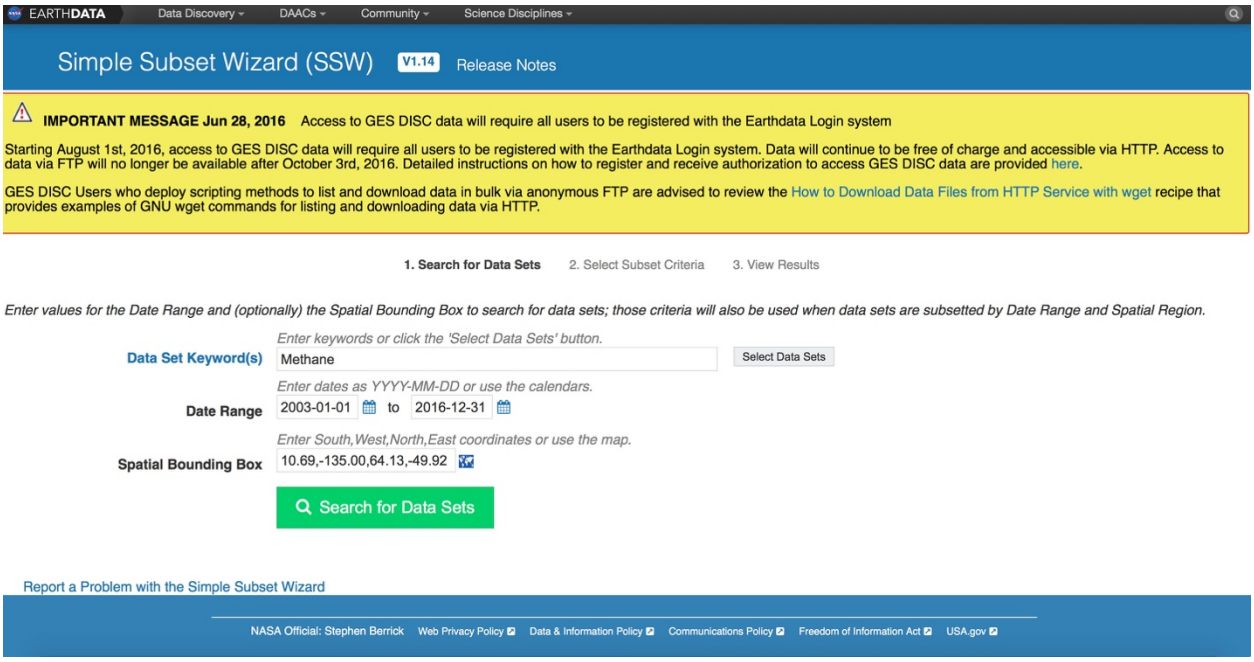
The main configuration file is the subsettableDatasets.xml file. For each dataset, it lists the subsetter for that dataset, along with some extra information. There is also a file called agents_configuration.xml with information on the subsetting agents, i.e. the service providers, one per distinct subsetter.

Another interesting feature of the SSW is that it has the ability to distinguish between synchronous and asynchronous subsetting services. If the type="synchronous", it means you can issue a URL request and get the result in the response directly. If the type = "asynchonous", it means you supply an email, the subsetter subsets and stages the data, then sends an email to the user to pick it up later, once the request has been serviced.

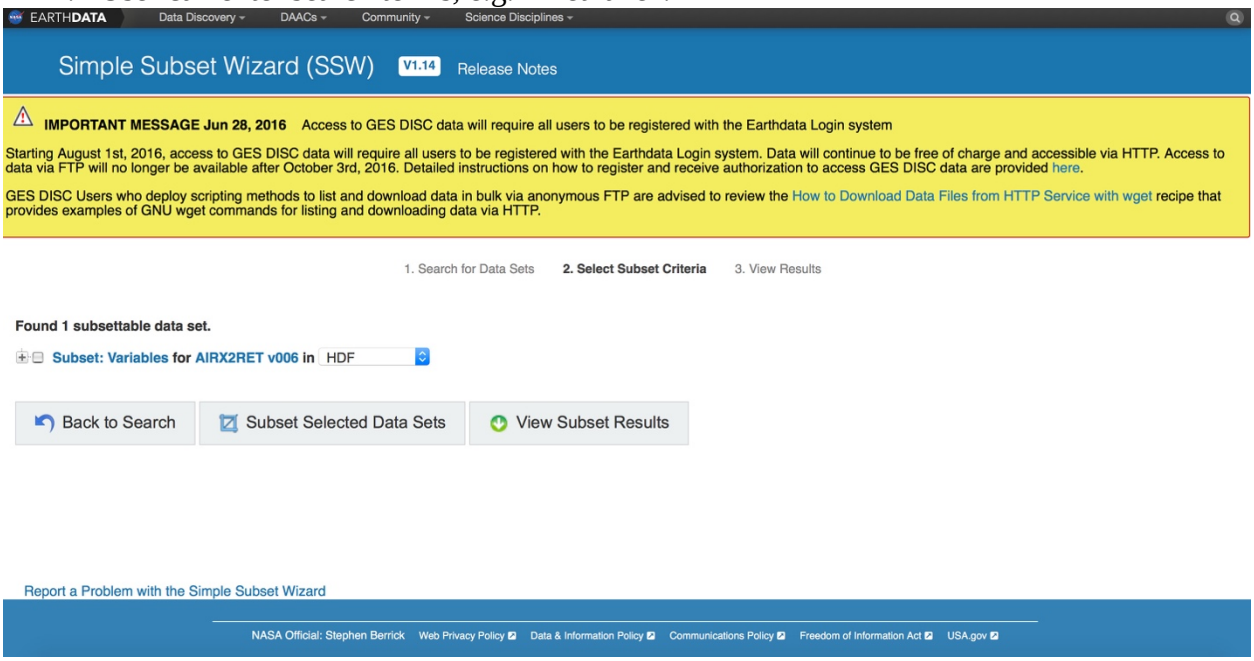
One example of this is MODIS Reprojection Tool (MRTWeb), whereby the extra, more complex subsetting options needed (projection, grid, bands) were combined to the basic subsetting options allowed by the SSW, by enumerating the possibilities into "Recipes".

The SSW subsetter is available to view at: http://ssw.gsfc.nasa.gov/SSW.

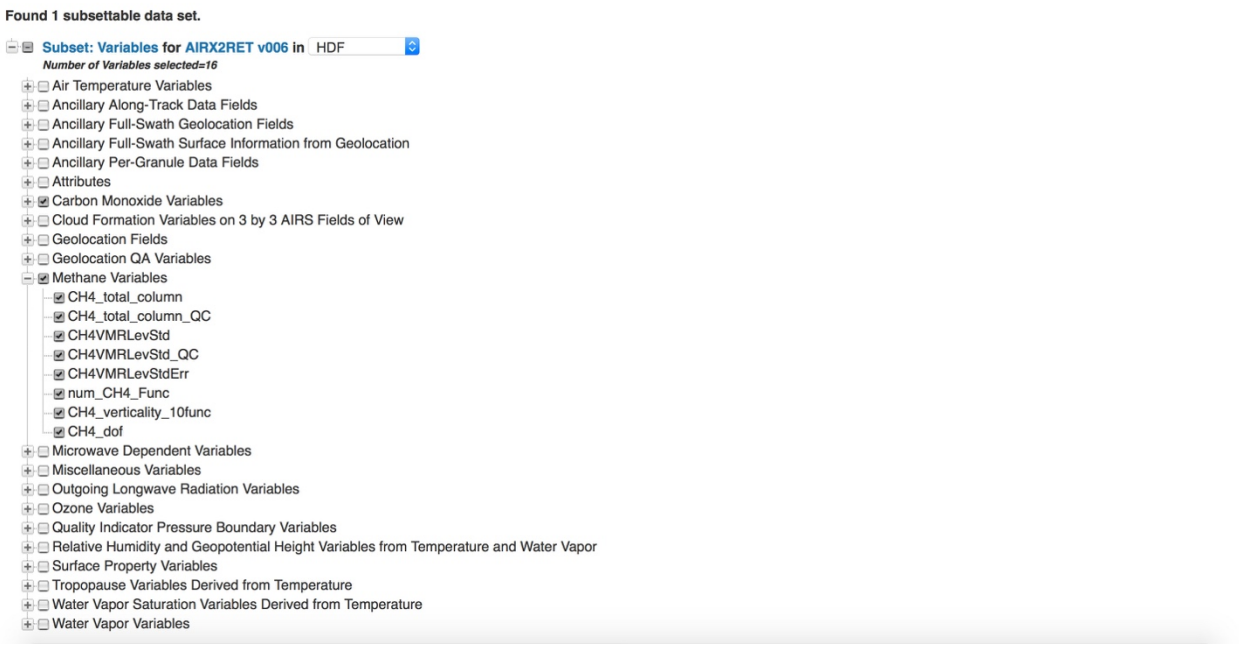
Taking an example, e.g. “Methane over Colorado between 2003 and 2016”, we entered the terms into the UI. See the series of screen grabs to follow the workflow.



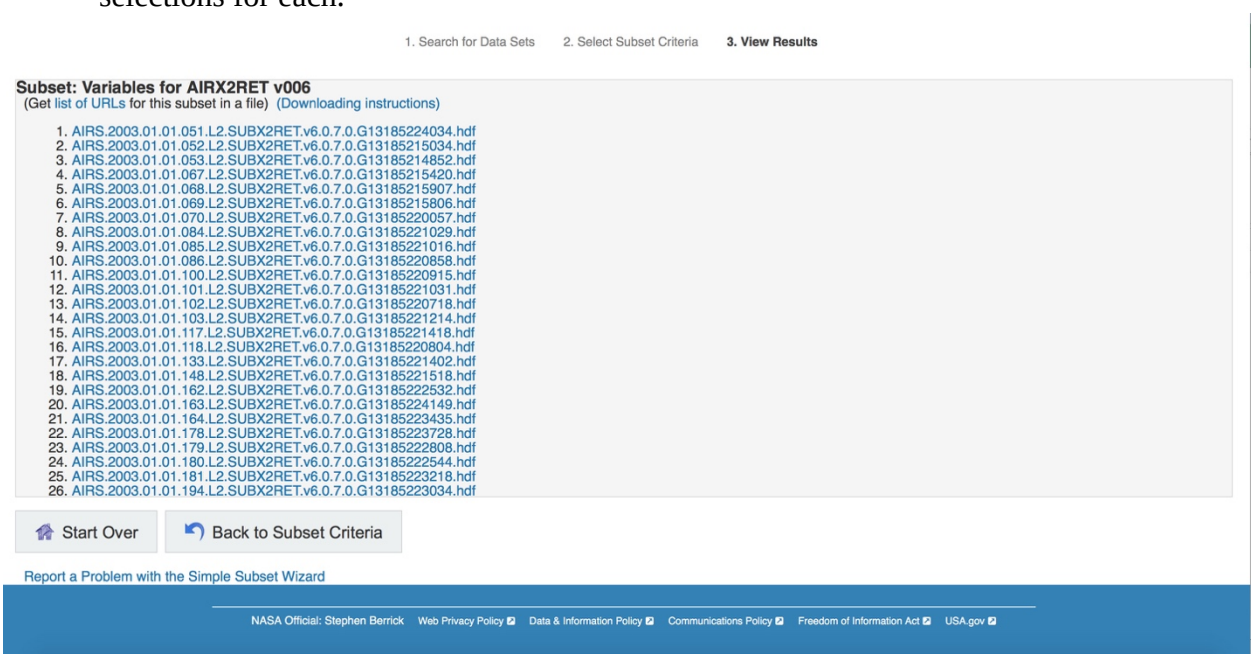
1. User can enter search terms, e.g. “Methane”.



2. User can view select from subsettable data sets found. Optionally, user can select format option.



3. User can expand collection found to display measurement and variables, and make selections for each.



4. User can click on individual subsetted data sets

Analysis of SSW workflow

Once a user has provided the constraints and entered search terms, the results are returned as a list of collections. Most collections can be subsetted immediately, and those which cannot immediately be subsetted are indicated with an *. The main problem with this workflow is that it is limited in scope in terms of selection of options, and does not allow users to preset options for subsequent subsetting. It is quite common to need to make several attempts at entering the spatial

constraints, or temporal constraints in order to bring the granule count (and hence the size of the download) down to something reasonable. Should a user arrive at the "Get Data" step and find they need to refine the search, then they are redirected to return to the first UI form where they must re-enter the majority of the information for the constraints. The UI form is actually spread across three different UI forms, which are displayed as the user proceeds through the workflow.

If the subsetted granules are available in a choice of file formats, e.g. HDF, GeoTIFF, etc., then these are presented to the user in a drop-down list for each granule file. In many cases, the user may prefer GeoTIFF for all granules but is required to go to each granule in turn and click the desired option. This is time consuming.

Some users will need to enable a repeat of all the entries on a subsequent visit to the Simple Subset Wizard form, and will need to keep on repeating their user input each time they want to perform a similar search, perhaps at a different time. This effort could be reduced, if the UI was able to recall the user preferences, in terms of spatial region of interest, time frame, measurement names used as search terms, and other options. Sometimes, only one of these constraints may change, between subset requests. It may be only the measurement name, or the time range that changes between searches.

SSW Shortfalls

The SSW UI falls short of the mark in the following ways: while it does provide a simple search term entry letter box form, it cannot interpret a complex search string, e.g. "Methane over Colorado between 2003 and 2015" and it does not allow users to preset options for subsequent subsetting. Also, the CMR is not employed in the search for data and services. This means that even though a cross-DAAC search is possible using SSW, the search results often diverge from those that would be returned via a CMR search, and the availability of data and services is not as perfect as it could be, with all the richness of services available across the DAACs as they exist today, and which exist under the auspices of disparate systems, perhaps requiring a separate authentication process, and limited to those data sets which exist in any given data pool. Other shortcomings of SSW are listed below:

- Missing metadata surfaced to the UI (not a metadata standard for services defined).
- UI does not have the full set of subsetting capabilities. It can only perform subsetting capabilities, e.g. spatial, temporal and variable subsetting. It cannot perform complex subsetting e.g. reprojection, resampling, regridding, reformatting, although saving to a alternate file format is possible, where the option is allowed.
- UI is limited to data sets in a XML config file (not dynamically updated).
- Not all DAACs offer services, hence not a universal standard of services available at the DAACs, i.e. limited to 10 DAACs, but is highly dependent on what the local services policy is at a DAAC.
- Not a faceted search.
- Synchronous vs. asynchronous, i.e. asynchronous is not fully implemented or supported.

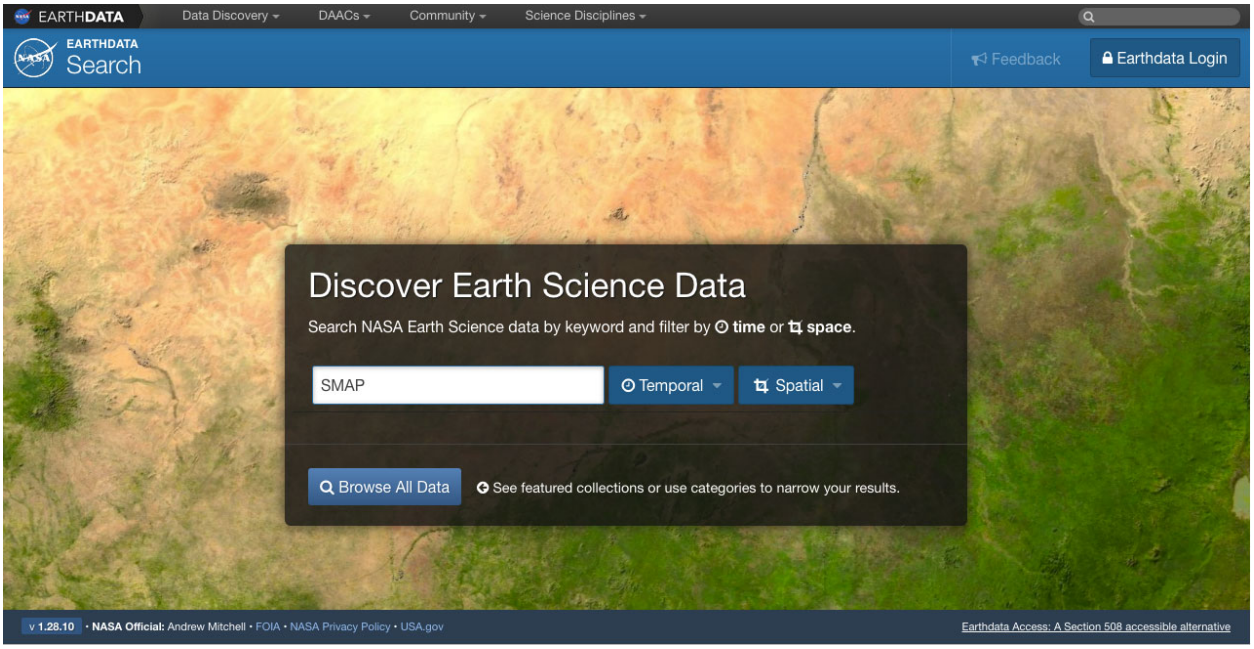
Ultimately, we need to find a way to make the user experience more seamless and less onerous for continuous long-term usage of the tool.

Earthdata Search Client Workflow

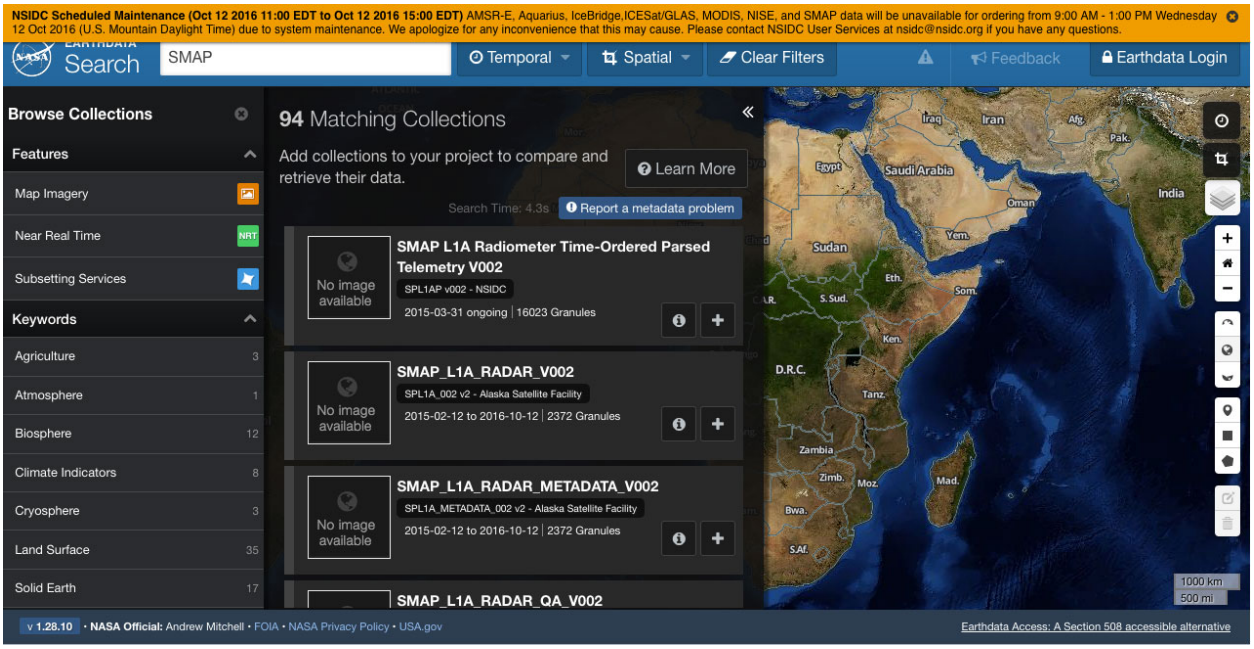
Introduction to Earthdata Search Client

Next we considered the Earthdata Search Client (EDSC), and performed a similar search example. The main difference between the SSW and the EDSC is that the EDSC is the latest service offering for discovering, searching, visualizing and retrieving earth science data. It has a more sophisticated UI, with a mapping display timeline and offers a variety of ways to search: via search terms, via science (GCMD) keywords, faceted search, and/or by spatial, temporal constraints, instrument/platform etc. Another key difference between SSW and EDSC is that the EDSC uses the Common Metadata Repository (CMR) for discovery and search, providing access to a larger set of collections and granules. Once a collection has been found, the UI offers the user an option to further select granules within a collection, and once the results set is determined, it offers a variety of ways for a user to obtain the data, via direct download (via HTTP) or via a service, (e.g. OPeNDAP).

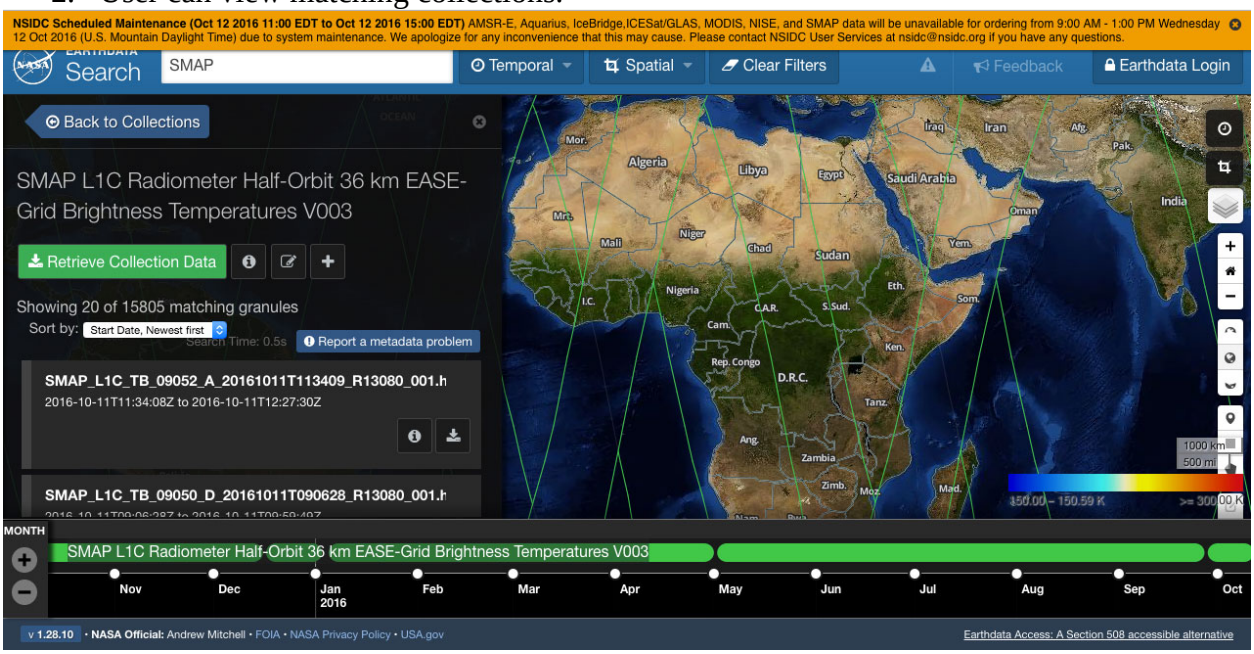
To see what a given dataset looks like through the search client itself, go to: https://search.earthdata.nasa.gov



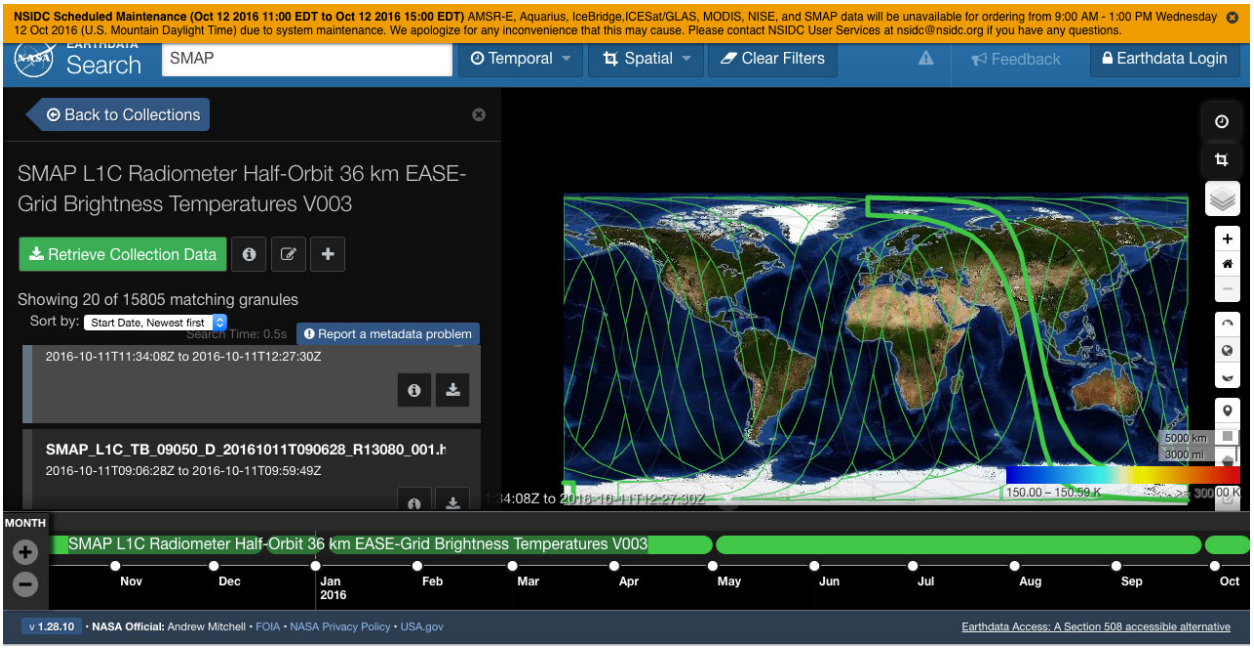
1. User can enter search terms, e.g. “SMAP”. Optionally, user can define temporal and spatial constrains, or perform this at the next screen.



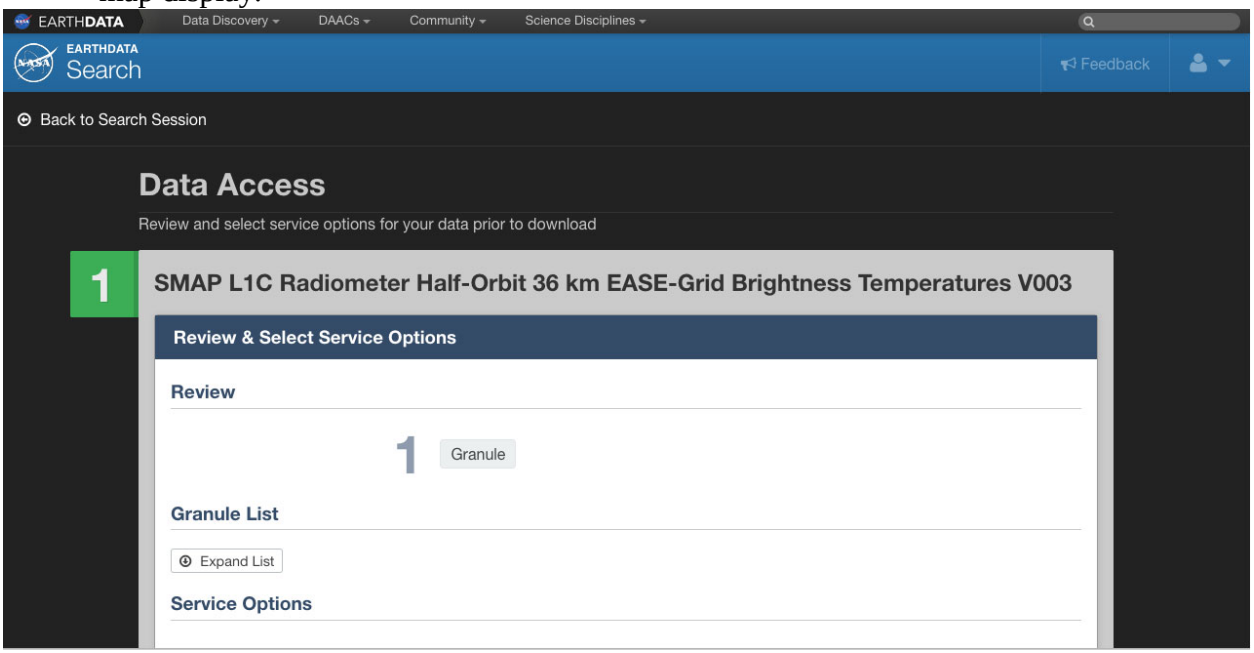
2. User can view matching collections.



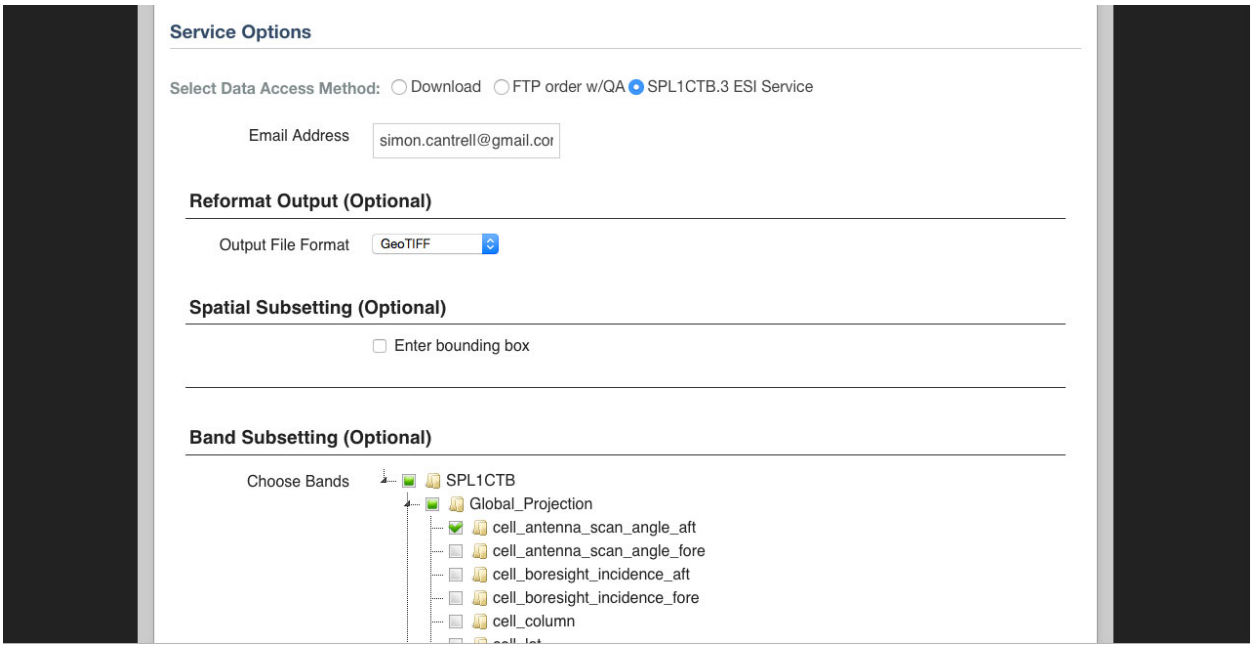
3. User can optionally enter temporal and spatial constraints via a top hat bar, to reduce number of matching collections.



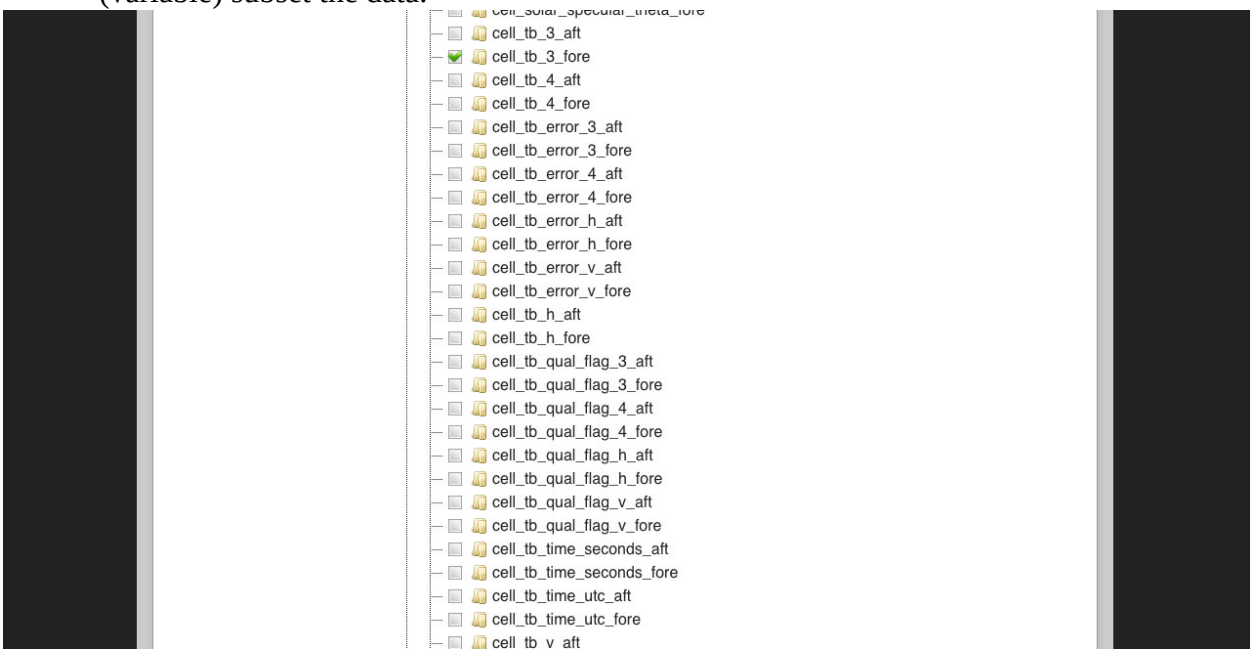
4. User can scroll through the list of matching collections. The orbit is highlighted on the map display.



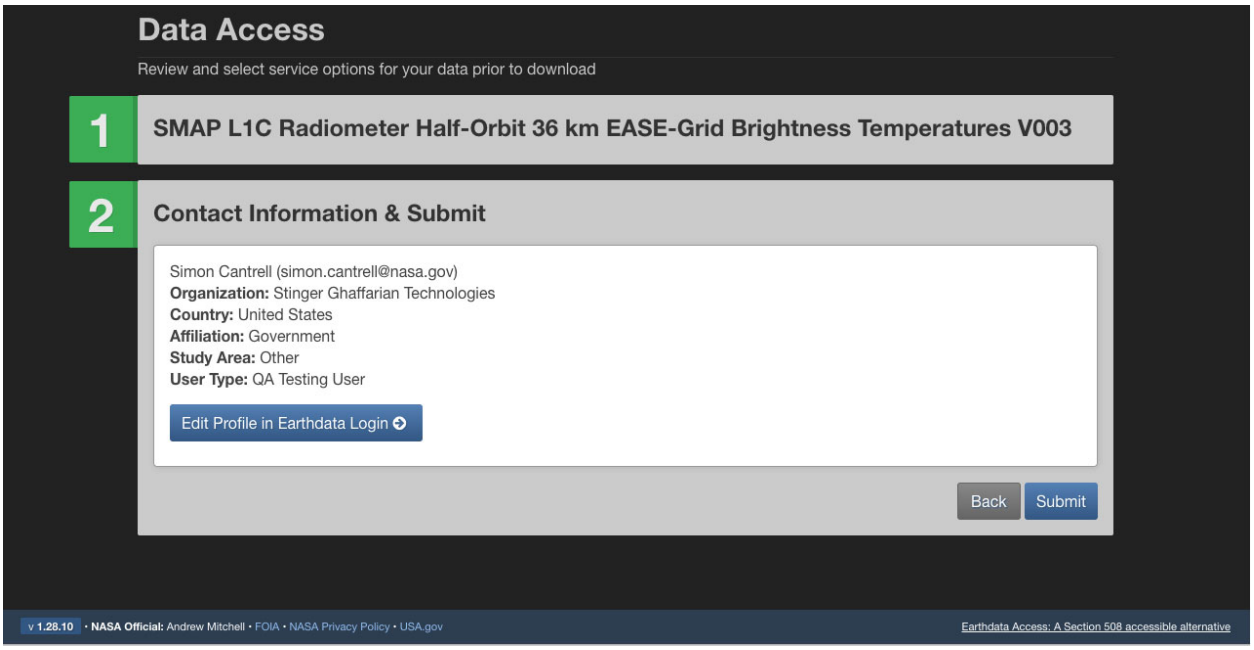
5. User can select a collection of interest, e.g. “SMAP L1C Radiometer Half-Orbit 36 km EASE-Grid Brightness Temperatures V003.



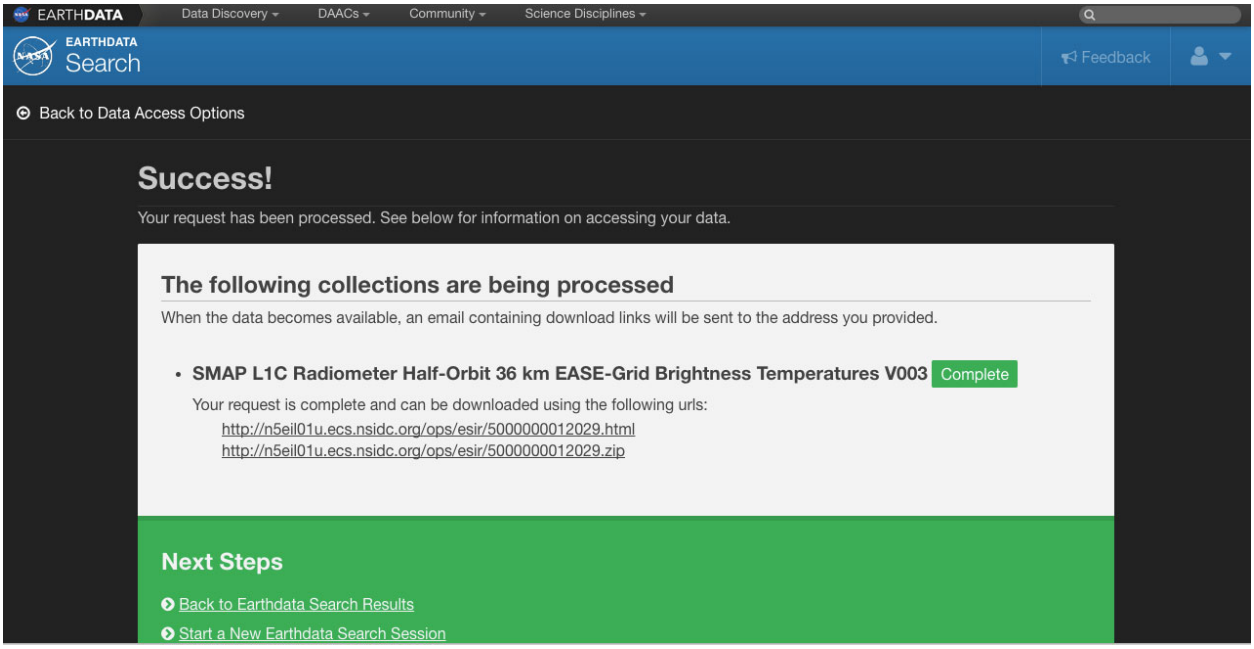
6. User can review Service Options. Note: data is available via direct download via http, via FTP Order w/QA or via ESI Service. In some cases, data is available via OPeNDAP service. Optionally, user may select to reformat the output, or spatially subset or band (variable) subset the data.



7. User can select a single band (variable) from the data.



8. User can click submit to confirm contact information.



9. User receives confirmation that the collection(s) are being processed. When the data becomes available, an email containing download links are sent to the email address provide by the user.

Output files for request id: 5000000012029

Click on the following link for a Request Summary:

requestSummary.txt

Retrieve list of files as a text listing (no html):

5000000012029.txt

Download all files in a single Zip file:

5000000012029.zip

Click on the following links for generated output files:

For Input Granule: 86687981

processed_SMAP_L1C_TB_09052_A_20161011T113409_R13080_001.h5_North_Polar_Projection_cell_tb_3_fore.tif (<1 MB, SCIENCE, image/tiff)

Note: Data transfer tools like wget or cURL can be used to retrieve all the data with one command using the above text listing as input. Or, simply paste any of the above output file URLs into your browser's location bar to download a file.

In a UNIX-like environment, use the following commands to download data to the current

10. User clicks on link(s) provided. In this case, the http link provides a) the Request Summary, b) the list of files as a test file, c) a link to a zip file, d) a link to the generated output files, in this case a tiff formatted output file was requested.

Analysis of EDSC workflow

One of the features of the EDSC workflow is that the user is immediately given the opportunity to specify and enter the search terms of interest. Once this has been completed, the next UI form provides confirmation of collections which contain data of interest. The user is able to enter the spatial and temporal constraints, or alternatively to select science keywords, on the left hand panel, to further constrain the data which will be returned. Optionally, these can be saved into a user's Project to reduce re-entry of these next time around. An attractive aspect of the UI is the map display. If the browse imagery is available, via GIBS or via browse metadata, then that granule can be displayed immediately on the map. This has the added advantage that the user gets some confirmation of what will be included in the downloaded data. If not, the user can get a general idea of the number of collections included in the results set, but often has to scroll down to find the collection of interest. Many collections are included in the results set which are not relevant to the user, or not favored by the user, so a substantial amount of scrolling is required in many cases, in order to arrive at the preferred choice of collection. If the user remembers to enter the spatial and temporal constraints before scrolling through all the collections listed, then the number of choices is further limited, and this reduces the effort on behalf of the user to arrive at the preferred collection, or collections.

Once the user has determined the specific collections which are of interest, then pressing the "Retrieve Collection" button takes the user to a Review and Select Service Options form. The good thing about this form is the user is alerted immediately as to how many granules is to be included in the download and what the total download size is.

EDSC Shortfalls

The user is not always given the option to select a desired file format, and in this case, would be required to convert the files individually following the download. In many cases, users may be given an option with some collection types, but not with others. In either case, then the final screen is usually a list of URLs pointing to the data pool. The challenge for a user with a large request is to perform a bulk download of the granules. Alternatively, if the files are small in size, but numerous then the user is not given a choice to bundle these into one file for the purposes of download. Again, if the number of files exceed 2000, and these are small in size, it is arbitrary that the limit is still in place, to restrict what would be a download of a relatively small-sized file, or files.

If the order option is chosen, then the user is required go over to their email to retrieve the order and download instructions, which are additional steps, with the necessary authentication and download steps included in that workflow.

In addition, there are the following deficiencies:

- No variable level in search or in facets.
- Inconsistent use of services, and choice of file formats from one collection to another.
- Limited subsetting capability. Simple Subsetting only. Complex subsetting is not there yet. Currently only works with collections configured for ESI requests, or OPeNDAP requests. Not all DAACs offer services, hence not a universal standard of services available at the DAACs, i.e. limited to a handful of DAACs but highly dependent on the local DAAC service configuration, which varies from DAAC to DAAC. Note, the earlier "Methane" search example did not yield a subsettable collection. SMAP L1/L3 collections are configured for ESI requests, so we used this as a subsetting example.
- No storage of preferences.

The EDSC is constantly evolving and the current interface is being improved.

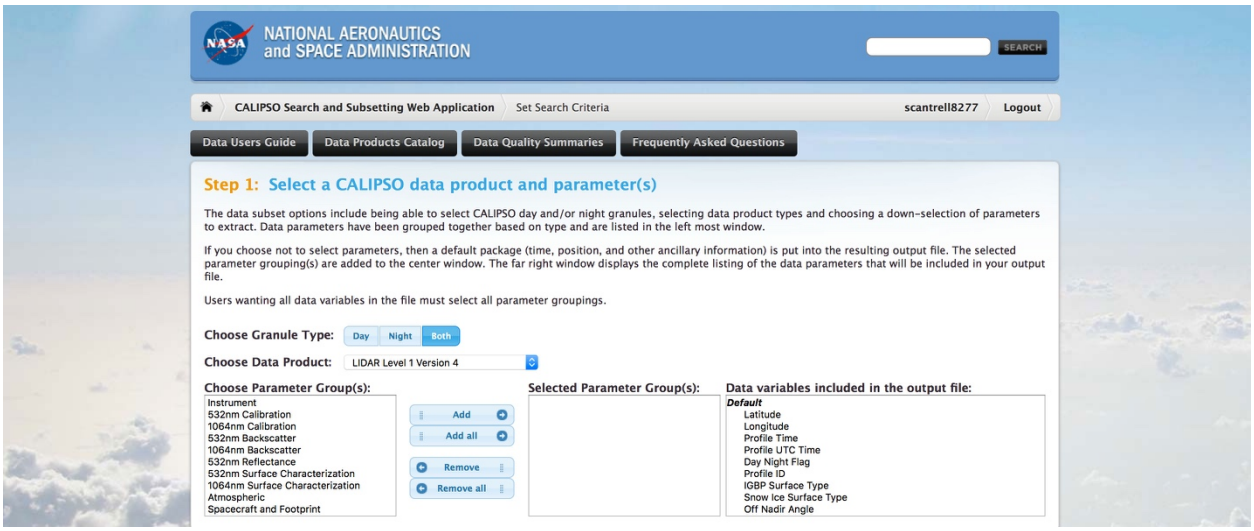
CALIPSO Search and Subsetting Client Web Application Workflow

The last candidate we selected for comparison purposes is the CALIPSO Search and Subsetting Client Web Application. This client is dedicated to a single data type and one of the most desirable features is its ability to perform measurement / variable subsetting on Calipso data products.

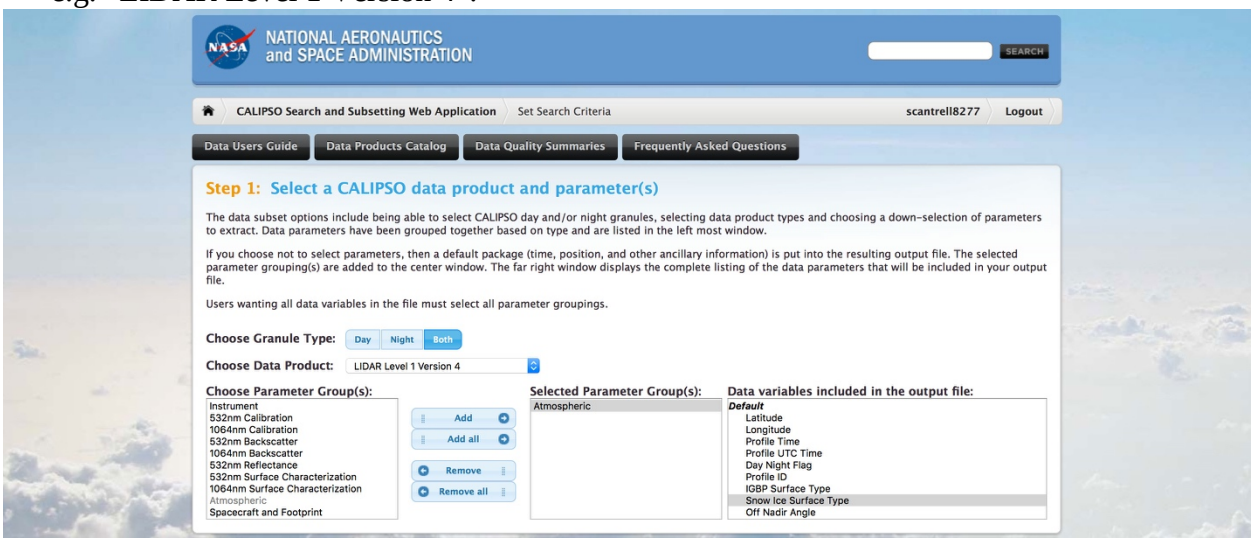
Intro to CALIPSO Search and Subsetting Client Web Application

The CALIPSO Search and Subsetting web application enables a more sophisticated approach to selecting and ordering CALIPSO lidar data by date, time and geolocation.

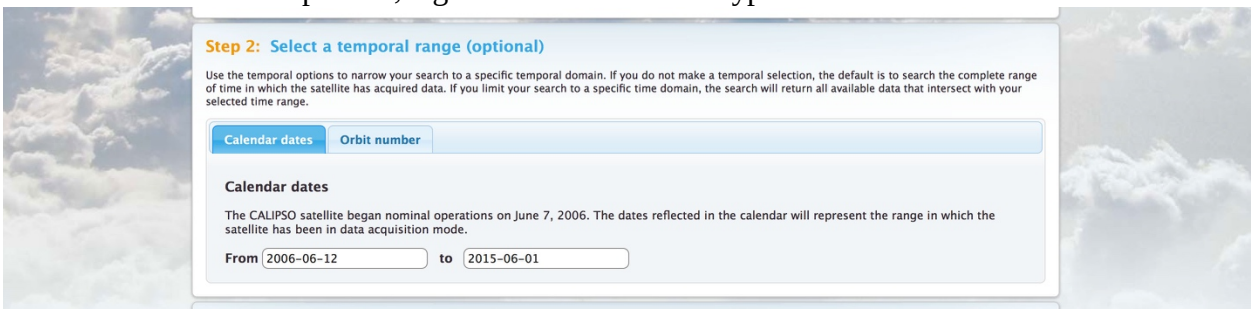
The CALIPSO Search and Subsetting Client Web Application (CSSC) subsetter is available to view at: https://www-calipso.larc.nasa.gov/search/login.php. It requires a URS login to enable access.



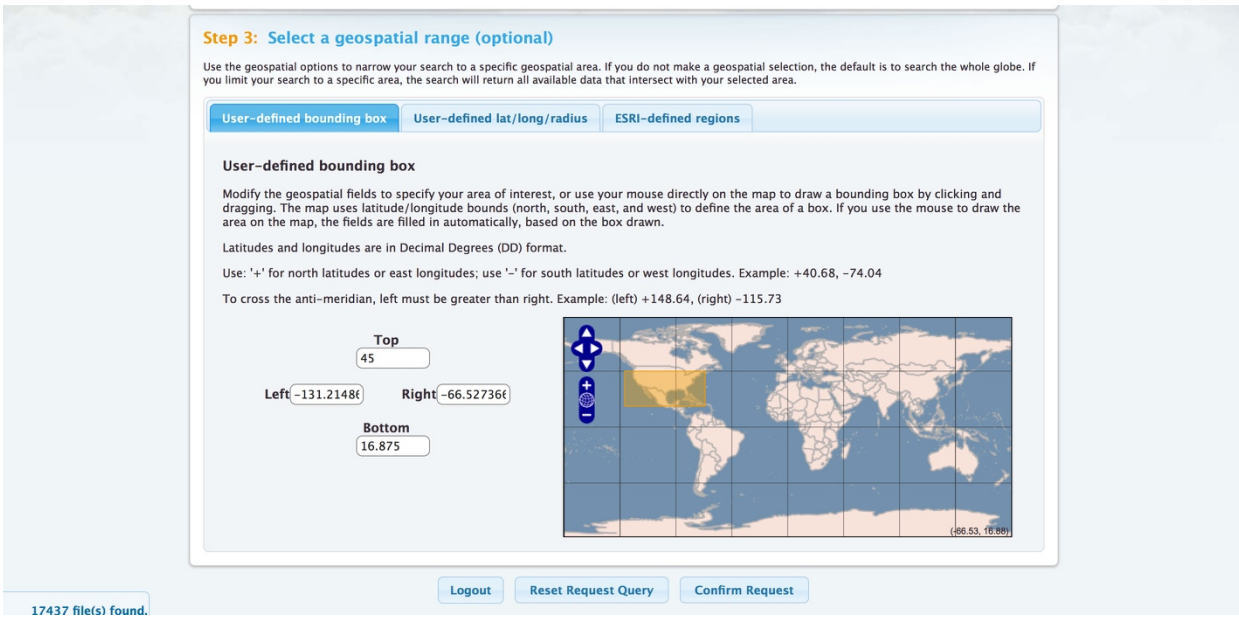
1. User can choose granule type, e.g. Day, Night, or Both. User can also choose a Data Product, e.g. “LIDAR Level 1 Version 4”.



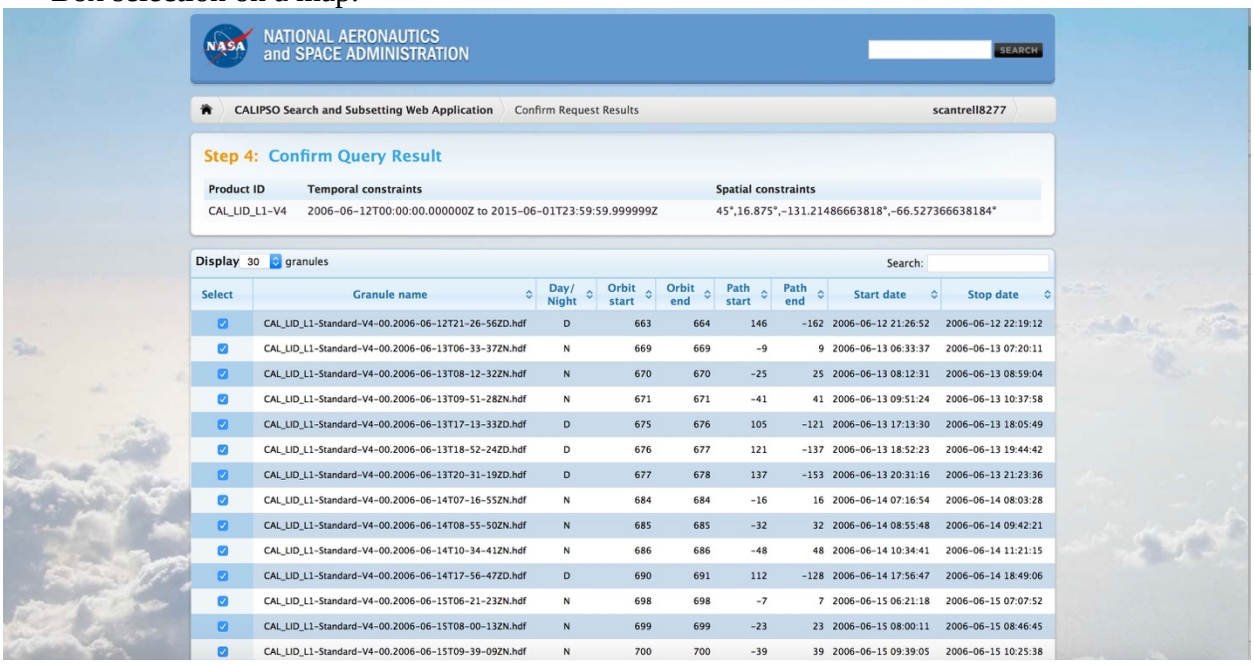
2. User can choose Parameter (Measurement) Groups, e.g. “Atmosphere”, and include Data Variables in the output file, e.g. “Snow Ice Surface Type”.



3. User can identify temporal constraints, e.g. a time range from 2006-08-12 through 2015-06-01.

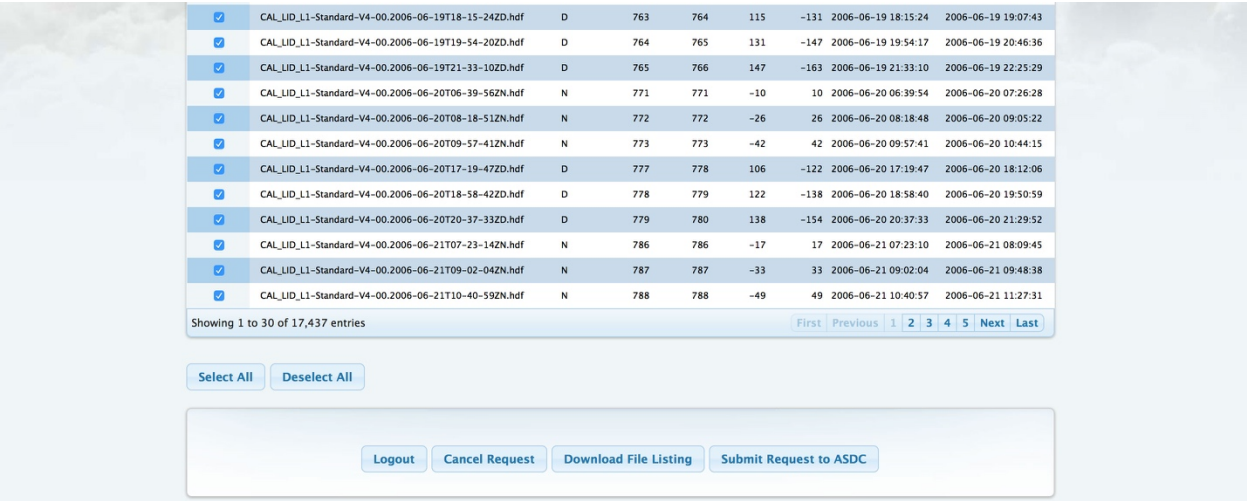


4. User can identify spatial constraints, e.g. over North American continent, via a Bounding Box selection on a map.



Select	Granule name	Day/ Night	Orbit start	Orbit end	Path start	Path end	Start date	Stop date
☑	CAL_LID_L1-Standard-V4-00.2006-06-12T21-26-56ZD.hdf	D	663	664	146	-162	2006-06-12 21:26:52	2006-06-12 22:19:12
☑	CAL_LID_L1-Standard-V4-00.2006-06-13T06-33-37ZN.hdf	N	669	669	-9	9	2006-06-13 06:33:37	2006-06-13 07:20:11
☑	CAL_LID_L1-Standard-V4-00.2006-06-13T08-12-32ZN.hdf	N	670	670	-25	25	2006-06-13 08:12:31	2006-06-13 08:59:04
☑	CAL_LID_L1-Standard-V4-00.2006-06-13T09-51-28ZN.hdf	N	671	671	-41	41	2006-06-13 09:51:24	2006-06-13 10:37:58
☑	CAL_LID_L1-Standard-V4-00.2006-06-13T17-13-33ZD.hdf	D	675	676	105	-121	2006-06-13 17:13:30	2006-06-13 18:05:49
☑	CAL_LID_L1-Standard-V4-00.2006-06-13T18-52-24ZD.hdf	D	676	677	121	-137	2006-06-13 18:52:23	2006-06-13 19:44:42
☑	CAL_LID_L1-Standard-V4-00.2006-06-13T20-31-19ZD.hdf	D	677	678	137	-153	2006-06-13 20:31:16	2006-06-13 21:23:36
☑	CAL_LID_L1-Standard-V4-00.2006-06-14T07-16-55ZN.hdf	N	684	684	-16	16	2006-06-14 07:16:54	2006-06-14 08:03:28
☑	CAL_LID_L1-Standard-V4-00.2006-06-14T08-55-50ZN.hdf	N	685	685	-32	32	2006-06-14 08:55:48	2006-06-14 09:42:21
☑	CAL_LID_L1-Standard-V4-00.2006-06-14T10-34-41ZN.hdf	N	686	686	-48	48	2006-06-14 10:34:41	2006-06-14 11:21:15
☑	CAL_LID_L1-Standard-V4-00.2006-06-14T17-56-47ZD.hdf	D	690	691	112	-128	2006-06-14 17:56:47	2006-06-14 18:49:06
☑	CAL_LID_L1-Standard-V4-00.2006-06-15T06-21-23ZN.hdf	N	698	698	-7	7	2006-06-15 06:21:18	2006-06-15 07:07:52
☑	CAL_LID_L1-Standard-V4-00.2006-06-15T08-00-13ZN.hdf	N	699	699	-23	23	2006-06-15 08:00:11	2006-06-15 08:46:45
☑	CAL_LID_L1-Standard-V4-00.2006-06-15T09-39-09ZN.hdf	N	700	700	-39	39	2006-06-15 09:39:05	2006-06-15 10:25:38

5. User can select from granules returned in a query result list, and select individual granules with a check mark.



☑	CAL_LID_L1-Standard-V4-00.2006-06-19T18-15-24ZD.hdf	D	763	764	115	-131	2006-06-19 18:15:24	2006-06-19 19:07:43
☑	CAL_LID_L1-Standard-V4-00.2006-06-19T19-54-20ZD.hdf	D	764	765	131	-147	2006-06-19 19:54:17	2006-06-19 20:46:36
☑	CAL_LID_L1-Standard-V4-00.2006-06-19T21-33-10ZD.hdf	D	765	766	147	-163	2006-06-19 21:33:10	2006-06-19 22:25:29
☑	CAL_LID_L1-Standard-V4-00.2006-06-20T06-39-56ZN.hdf	N	771	771	-10	10	2006-06-20 06:39:54	2006-06-20 07:26:28
☑	CAL_LID_L1-Standard-V4-00.2006-06-20T08-18-51ZN.hdf	N	772	772	-26	26	2006-06-20 08:18:48	2006-06-20 09:05:22
☑	CAL_LID_L1-Standard-V4-00.2006-06-20T09-57-41ZN.hdf	N	773	773	-42	42	2006-06-20 09:57:41	2006-06-20 10:44:15
☑	CAL_LID_L1-Standard-V4-00.2006-06-20T17-19-47ZD.hdf	D	777	778	106	-122	2006-06-20 17:19:47	2006-06-20 18:12:06
☑	CAL_LID_L1-Standard-V4-00.2006-06-20T18-58-42ZD.hdf	D	778	779	122	-138	2006-06-20 18:58:40	2006-06-20 19:50:59
☑	CAL_LID_L1-Standard-V4-00.2006-06-20T20-37-33ZD.hdf	D	779	780	138	-154	2006-06-20 20:37:33	2006-06-20 21:29:52
☑	CAL_LID_L1-Standard-V4-00.2006-06-21T07-23-14ZN.hdf	N	786	786	-17	17	2006-06-21 07:23:10	2006-06-21 08:09:45
☑	CAL_LID_L1-Standard-V4-00.2006-06-21T09-02-04ZN.hdf	N	787	787	-33	33	2006-06-21 09:02:04	2006-06-21 09:48:38
☑	CAL_LID_L1-Standard-V4-00.2006-06-21T10-40-59ZN.hdf	N	788	788	-49	49	2006-06-21 10:40:57	2006-06-21 11:27:31

6. User can select download file listing and download granules (via a synchronous request). Optionally, user can submit a request to ADSC (via an asynchronous request).

Analysis of CALIPSO Search and Subsetting Web Application workflow

CALIPSO Search and Subsetting Web Application workflow is highly conformant to CALIPSO data products and qualities/options related to these data products. The user is immediately able to select CALIPSO day and/or night granules when using the tool. The UX is relatively straightforward, and requires some knowledge of the data products. The choice of data product immediately presents the measurement groups and the corresponding variables included in the output file. Selection of temporal range and geographic range is optional, and can be done by for temporal via selection of calendar dates or orbit number, and via a bounding box for geographic range.

Once the user has made the data product selections, and has provided the temporal and geographic constraints, then the request may be confirmed. If the user has made a large request, in terms of the number of granules, this in indicated as a pop-up notification prior to confirmation of the request. The user may return to the constraint definition or data product selections in the form to further refine the search selections/qualities, or simply reset the request query.

The CALIPSO Search and Subsetting Web Application is one of a number of pure search and subsetting tools available from ASDC. Other are dedicated to enabling search and subsetting of other data products, i.e. CERES, MISR, MOPITT and TES.

CALIPSO Search and Subsetting Web Application Shortfalls

CSS Web Application has the following deficiencies:

- The tool is limited to search and discovery of CALIPSO data products only and uses services provided by the ASDC.

- The subsetting request does not store the options, for continued use, the next time the user uses the tool. The user does not get confirmation of the subsetted data being available until the data becomes available.

- The user cannot choose between data formats to receive the data. Also, the projection scheme and gridding schemes in which the data is returned is native to the product, if available.

- There is no universal standard of services available at the DAACs, i.e. this tool is limited to using services available at the ASDC DAAC, therefore, it is highly dependent on what the local DAAC services policy is.

- The user cannot combine data products into one file, or opt to mosaic several data sets together.

A network diagram which summarizes the workflows for each UI in relation to the idealized workflow is shown in Figure 1 below:

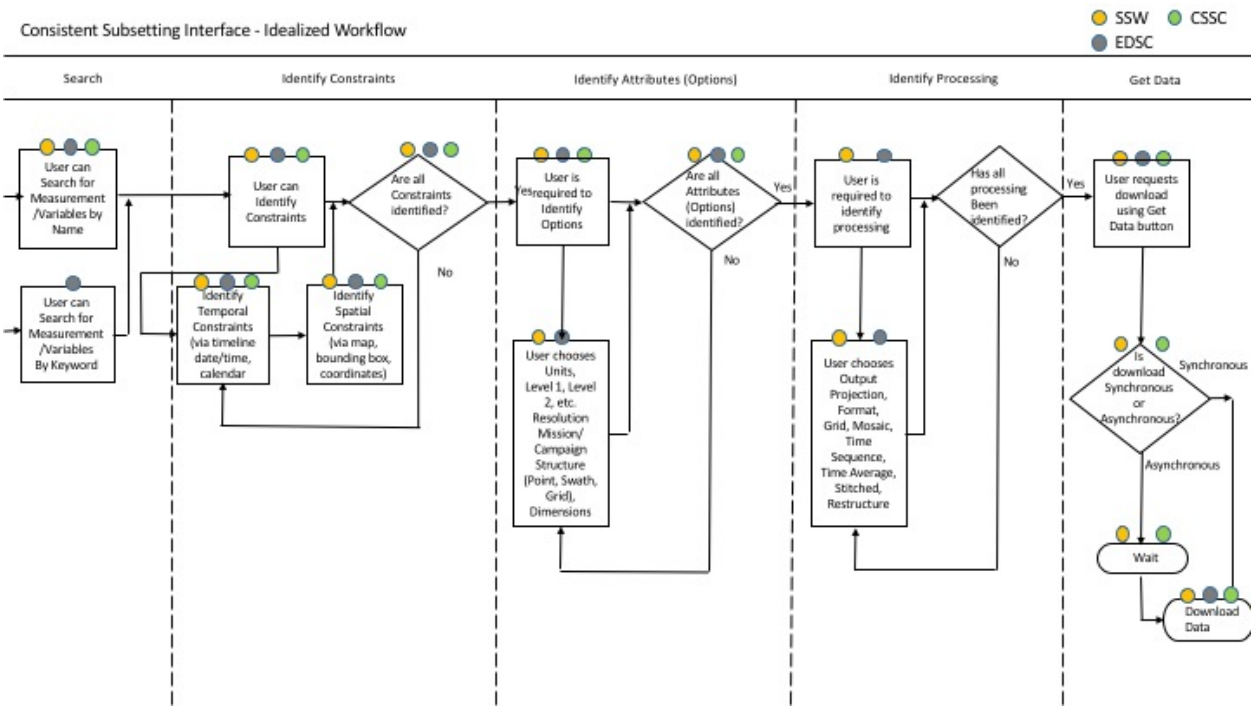


Figure 1. Consistent Subsetting Interface - Idealized Workflow

The above steps describe the idealized end-to-end user experience (UX). The complete workflow enables search (via name or keyword) followed by identification of constraints, followed by identification of desired attributes (options)/processing, and acquiring the data (either synchronously or asynchronously).

The main advantages of the idealized workflow are that all the information could be entered via a single form, and options can be selected once and stored along with the user's preferences, such

that the number of keystrokes may be limited to either entering a search string into a single search box, or make a selection from a list.

The consistent subsetting UI could be designed to interpret the constraints parsed from part of the text string entered in the search box, or alternatively, the user may also select the spatial and temporal constraints through a map and timeline, or via a form, and, optionally, further refine the qualities. These user selections will enable the system to identify the services needed perform the needed processing.

Use Cases

The next logical step is to develop the "idealized" workflow and realize this into a use case analysis for a fully-fledged UI. We built up the tool functionality by constructing a series of use cases, for every step in the workflow.

As part of this study we took this analysis into a collaborative environment. A more detailed analysis of the "idealized" workflow for the Consistent Subsetting Interface, is available for viewing at: https://docs.google.com/document/d/11dwu3PKieAE875IFV9OsSYWf-6AfIp0f7M0Tc_bI4es/edit

5. High-Level Architecture

Next we set out to identify the High-Level Architecture which would enable implementation of the workflow defined earlier, and which can be realized within a reasonable timeframe.

Current Architecture

Today, we have a layered architecture with a stack which looks like this:

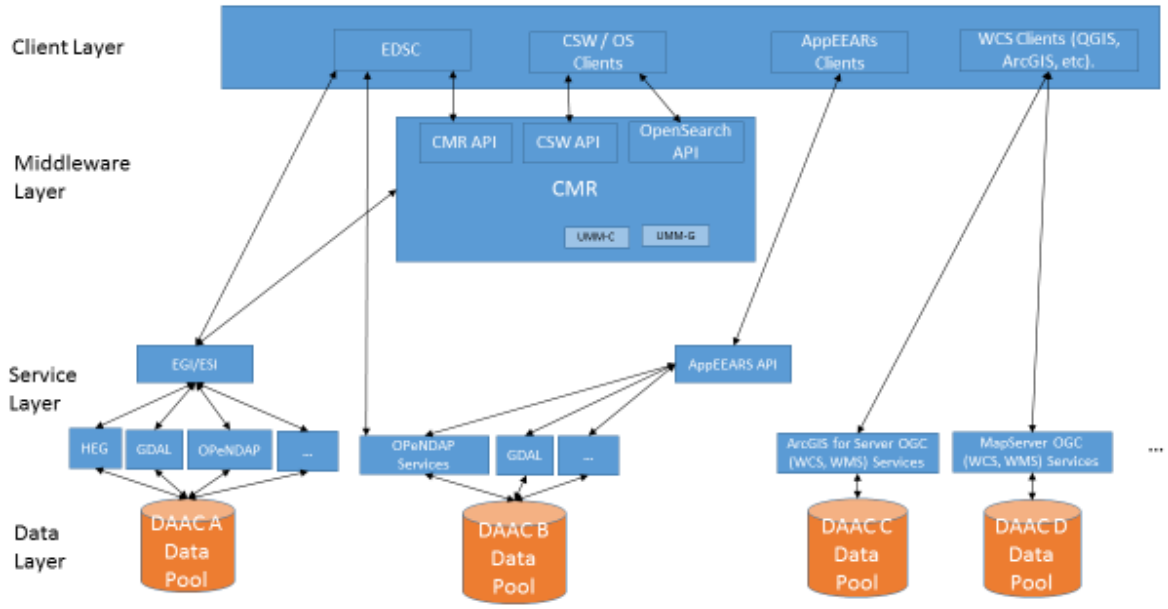


Figure 2. Current High-Level Architecture

While this robust architecture has greatly enabled discovery of collection and granules and is working well for users who already know exactly what service they want to use and how to access that service, it is currently unable to assist users in discovering variables, services and service capabilities in a consistent manner early in a user workflow. This inability to discover services and service capabilities is ultimately what prevents clients from being able to consistently allowing users to select subsetting options early in the workflow. For example, if a client wanted allow users to specify options such as "available in geoTiff" or "available in polar stereographic" projection, the only way to do so today is for the client to talk directly to the service to get those capabilities. If a client attempts to allow a user to discover a handful of collections and then talks to 1 or more services for each collection in order to determine if the services capabilities meet the user's request, it can quickly become non-performant.

Additionally, in order for clients to get service capabilities, they have to speak the language of each of the service protocols offered. Today, that could be via WCS, ESI, OPeNDAP, AppEEARS, and more. The protocol needed varies based on collection requested. For example, the capability to make an ESI service request from Earthdata Search Client exists, but only ESI-enabled collections can service these requests. The format of these requests is ESI-specific, and any other client cannot easily issue a compliant request. For standards based clients, such as ArcGIS, a large portion of ESDIS' data is unreachable because in the current architecture there are very few WCS front-ends.

These problems lead us to envision tweaks to the existing architecture which would allow for standards-based variable and service discovery and access.

Proposed High-Level Architecture

In order to arrive at a high-level architecture, we need to make several considerations:

1) No one application programming interface (API) may entirely solve this problem.
   - Standards have been evolving at different rates. APIs have been developed to address these new standards. For example, OPeNDAP started out using DAP2, DAP4 protocols, and recently added OGC compliance. It was not until recently that OPeNDAP has been extended to meet the simple and complex subsetting needs.
2) We value accepted standards first, then extend those standards when necessary.
   - With a standards-based approach, we can consolidate development activities to focus on the technologies, whether new or re-used, to enable the standards-based APIs to evolve, hence aligning these efforts to achieve a nearer term realization of the appropriate architecture.
3) Any approach taken should ultimately enable migration to the cloud in the future.
   - The cloud must be a factor in the choice of a future architecture. A simple incremental approach can dictate a path to extend service functionality into the cloud.
4) Centralized architecture vs. Decentralized Architecture.
   - With the current CMR, we have a system which has evolved what is essentially a centralized architecture to ensure consistent and high performance search results. A more decentralized architecture for services has desirable characteristics, i.e. distributed functionality on discrete computing nodes, accessible from any client, etc. Agents

capable of running simple and complex subsetting jobs can be staged as needed, and co-located with data pools to reduce latency.

5) What is the role for a possible UMM-S and can or should it fit into new service model?
   - UMM-C already has some limited service metadata in the RelatedURL class. However, if we wanted to allow clients to really discover services in an efficient manner, we would need more metadata than the UMM-C currently holds and thus need to create an UMM-S. The most important thing here is that we stay as close to standards as possible, so that UMM-S can become mostly a vehicle by which to cache and search for service capabilities.

There are also several further considerations which need to play a part in any solution, given the amount of investment made by ESDIS in technologies which could be re-used:

a) With an OPeNDAP approach, how you subset the data is highly dependent on the structure and complexity of the data. When can we expect the functionality needed to achieve simple subsetting (spatial, temporal, band (variable) subsetting, and complex subsetting (reprojection, resampling, regridding, mosaicing, repackaging)? With which services code bases can we acquire this functionality?

b) With an ESI approach, how far do you go down the path of customization of tool adapters, in order to handle the complexities of each data set? Which DAACs will participate in an ESI-driven solution?

c) With a custom client, e.g. GDAL approach, where some users opt to create their own custom client to address a specific scientific need, there is the overhead of seeking out the documentation for the GDAL library, writing the tool, and invoking the tool. The client developers will need to know collection metadata, variable metadata and service metadata to access the services successfully. The architecture also needs to support these users.

With all of these considerations in mind, we believe the following key changes are necessary in order to enable clients to implement the Idealized Workflow while re-using as much of the existing architecture as possible.

- Addition of variable metadata into the CMR by implementing UMM-Var. This enables variable level search as a first-class means of discovery
- Addition of WCS compliance throughout the architecture. A standard protocol for the request is needed at the front-end of the architecture, and a standards-based service configuration is needed at the back-end of the architecture (Service layer). See sequence diagram in Figure 4 for additional details.
- Addition of service metadata by implementing UMM-S to make subsetting options a first-class means of discovery also. Our goal would be to auto-generate service level metadata to populate UMM-S as much as possible. We can augment this with the ability to do manual updates to allow data providers to enhance their service metadata.

This architecture can be seen below:

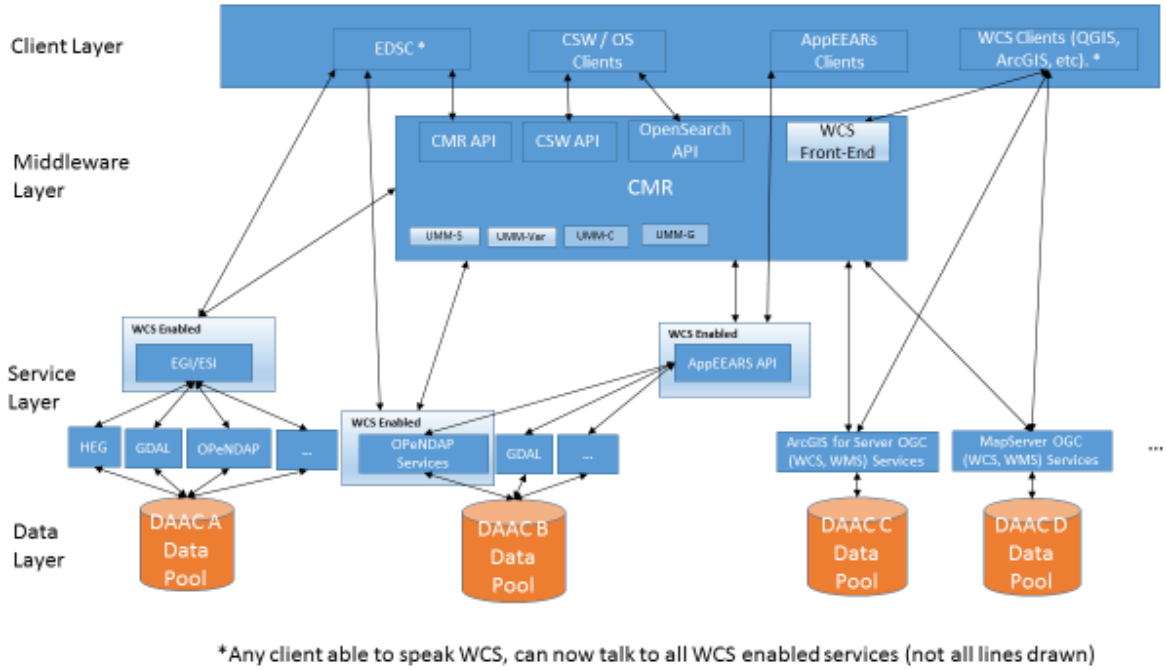


Figure 3. Proposed High-Level Architecture

In this service-oriented architecture, we are able to expose any service which supports WCS to standard WCS clients. Clients could interact with WCS services in two ways. First, for clients that already know how to connect to a WCS service at a DAAC, the client may connect directly to that service as they would today. As more and more services become WCS enabled, the ESDIS offering to WCS clients grows. At the service layer, the WCS protocol specification can be specified as a front-end to a backend service, i.e. OPeNDAP or ESI. Within this service model, the WCS request to be sent to the service is based on usable (non-opaque) arguments, i.e., spatial and temporal ROI. At the back-end, WCS formatted OPeNDAP URLs, or ESI URLs, with time and/or spatial aggregation, would return subsetted data (coverages) from the appropriate service. Some WCS backend services could query the CMR in order to find the granules that satisfy the spatio-temporal constraints if the underlying service was unable to do so efficiently. The benefit of this would be that the client would only need to know only WCS.

Second, for clients that do not know which DAAC WCS services exist, the CMR can add value by providing WCS getCapabilities across the enterprise and allowing clients to be relatively blind to the underlying services or DAACs. The CMR could do this by regularly polling all WCS services specified in UMM-C in order to cache the services' getCapabilities in UMM-S and UMM-Var. While the CMR would then have enough information to aid a client in developing a getCoverage request, ideally, the CMR would not be in the middle of all getCoverage requests. We would need to prototype how the CMR could provide the right information to a WCS client in the getCapabilities response to allow them to then connect directly to the correct backend service for the getCoverage request. This would stay true to CMR's goal in life of enabling search and discovery, while not becoming a bottleneck for accessing of data.

For clients who wanted to take advantage of non-WCS features, the CMR could now expose even greater advanced searching and faceting functionality based on UMM-S and UMM-Var. This is discussed in more detail in the next section.

Sequence Diagram of the Idealized Workflow

In Figure 4 below we walk through how this architecture would be used to enable the idealized workflow in more detail.

> Scenario [a]: As CMR, I can (periodically) get service capabilities via a getCapabilities request so that I can cache this information and enable complex searching on it..
> Scenario [b]: As a user of the UI, I can search for a given measurement/variable, spatial range and temporal range to view the available service capabilities.
> Scenario [c]: As CMR, I can receive detailed coverage metadata for a given collection and measurement/variable, via a describeCoverage request.
> Scenario [d]: As a user of the UI, I can receive subsetted data for my requested criteria via a getCoverage request.

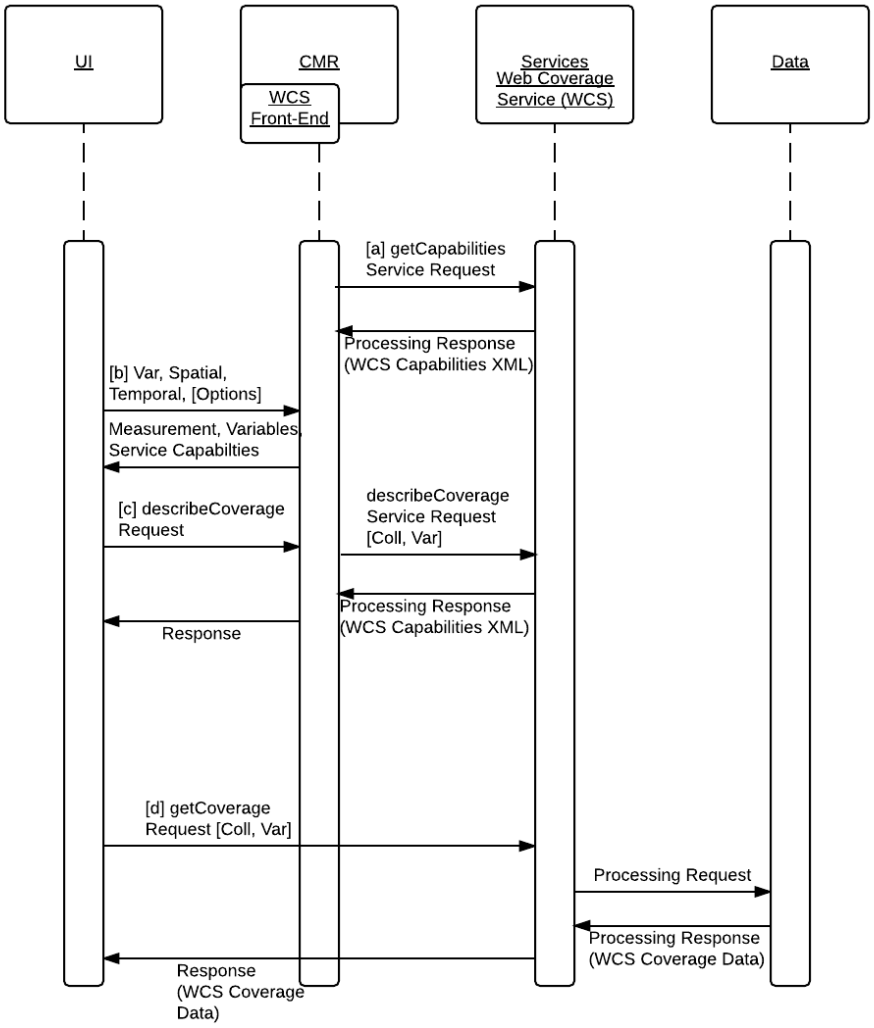


Figure 4. Proposed Services Sequence Diagram

Scenario for search and discovery

So, with the architecture implemented as described above, a search for "CH4_Total_Column over Colorado in 2016 in geoTiff" could now return enough information for a client to automatically request the actual data. This scenario is illustrated in Figure 5. "CH4_Total_Column" is represented by Variable B in Collection X and Granule Y.

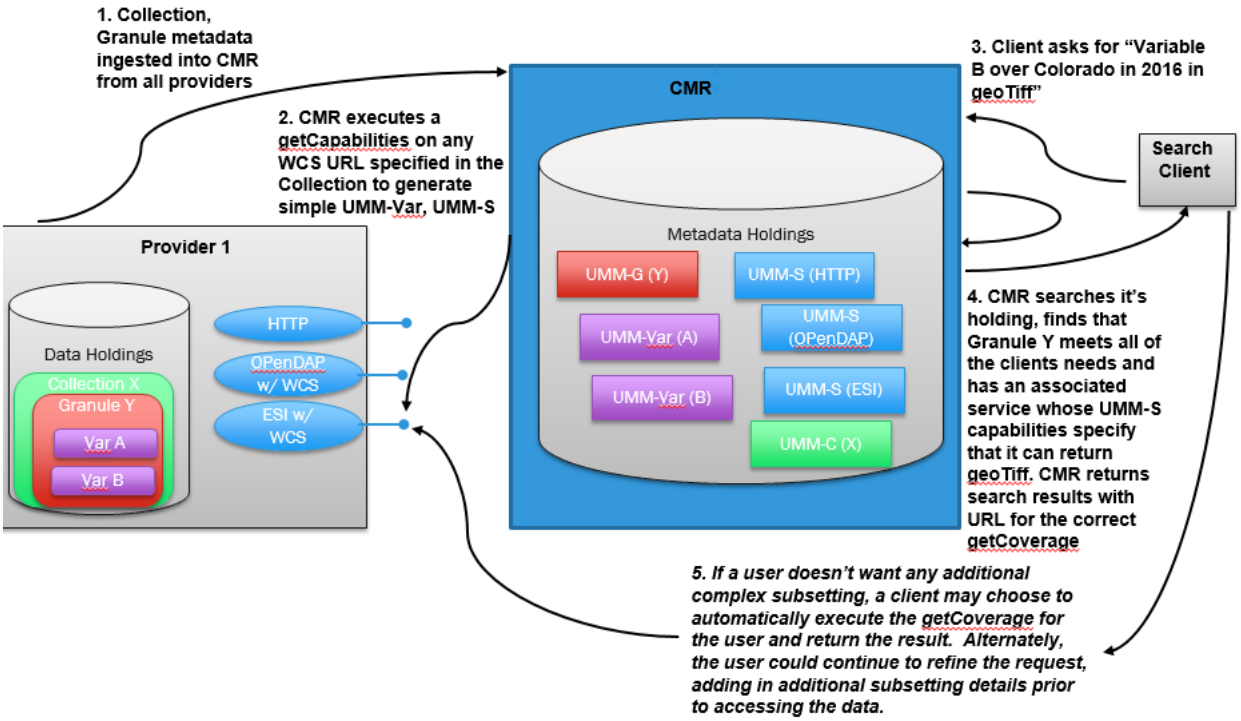


Figure 5. Possible query scenario – CMR (via UMM-C, G, Var, S) enables query request via a service adapter.

As a result of the search request, "Variable B over Colorado in 2016", Variable B, which is related to Collection X and Granule Y, is found to be available either for direct download via HTTP or via an OPeNDAP service at Provider 1. However, the power of services is to enable both the simple and complex subsetting functionality to be performed by the service, provided that those capabilities are available for Collection X. So now the request can be further extended to include any of the simple or complex subsetting options. In the example, only the OPeNDAP service is capable of providing the Variable B layer in GeoTIFF format.

By this method, more complex subsetting requests can be made via the UI/client (band, reprojection, regridding, etc.) depending on whether the service capabilities exist at the data provider for that data set (collection).

This flow works very well assuming that all services can return synchronous responses when providing the data. However, initial analysis of even just the ESI service shows that there are requests being made that take multiple hours to process. As such, this architecture needs to still account for asynchronous requests. Today, this is handled via "orders" in the CMR. However,

this makes the synchronous / asynchronous decision very static. Going forward, we would propose that services themselves determine, on a request by request basis, if data can be returned synchronously or not. According to the WCS standard, the getCoverage request is allowed to return a URL for accessing the data. We propose that if data needs to be returned asynchronously, the service provider would send a URL to the client indicating where the data will be staged once complete. The provider would then notify the user when the subsetted data is available. This is an area where more prototyping would need to be done to ensure this approach would work and still be fully WCS compliant. Until then, the CMR "order" approach would still exist and be able to be used.

A more detailed look at the layers of the architecture is given below:

Client Layer

The client layer features the UI. The function of the client layer is to enable the user to define the search, identify the constraints and qualities, and processing required. The UI represents the realization of the consistent subsetting interface which is characterized by a simple, minimalist workflow, which uses defaults wherever possible. This would consist of the ability to query the CMR to include UMM-Var and UMM-S targeted requests. It could then issue service requests to any WCS compliant service.

Middleware Layer

The CMR is a core component of the middleware layer. The function of the middleware layer is to drive the search via the CMR, and surface the results to the UI. Where services are available, these will be identified as part of the search. The Middleware layer will be extended to include a WCS API. In implementation, portions of the WCS-compliant ESI/Service Gateway which generate compliant WCS requests may be reused to reduce implementation time.

Service Layer

The service layer features the native services, or service APIs. These include: WCS Back-End, OGC (WCS, WMS) Services, etc. The function of the services layer is to handle the subsetting requests directly and pass these to the applicable service, (i.e. ESI, AppEEARS, OPeNDAP/Hyrax, ESRI ArcGIS Server, etc.) and return the coverages (data layers) or URIs to the requestor. Due to the complexities associated with some EOS data sets, i.e. MISR, which include the use of complex file formats, multi-dimensional data stacking and exotic projection schemes, we expect that the HEG or GDAL service will be required to handle some subsetting services on the Back-End. This configuration would require the use of the ESI tool adapters. For the mainstream simple subsetting cases, the WCS 2.0 EO-X compliant service configuration would handle the request. The variety of service pathways, and service capabilities and exceptions to these, could be stored in the CMR, within the UMM-S specifically (See Appendix C).

Data Layer

The data layer comprises all the available data, which typically reside in data pools, accessible directly only via HTTP or OPeNDAP protocols. FTP access in the in process of being phased out across the DAACs. The function of the data layer is to store the data products at each DAAC, or data provider. The data sets, or the subsetted versions of the data sets will be represented by URIs,

and these are surfaced in the UI. Ongoing efforts towards moving the data layer into the cloud, i.e. CUMULUS, would make data and services more accessible.

6. Identify Candidate Services/Tools for Re-use

In order to minimize the effort needed to fully realize the proposed End-to-End Architecture, we need to identify candidate services/tools for re-use. Much of this development effort will be on the UI layer and middleware front-end, with the need for a new WCS client, and the service adapter. We also need to focus on the back-end with the choice of service configurations, and the development of any server-side tool adapters. In the case of an ESI/Service Gateway front-end configuration to be used as a WCS client, we need to test with one or more of the current WCS service configurations, e.g. WCS 1.1.2, and continue this effort to test with a WCS 2.0.1 or greater service configuration.

Currently the candidate services/tools for re-use, by layer are:

UI Layer:

The UI layer candidates are: SSW, EDSC, WorldView, any WCS-clients, including GDAL, IDL, MatLab, QGIS (note that these clients need to be explicit as to which version of the service the request is made to). Note: currently, a QGIS client can issue a WCS 2.0 request, but does not understand the response from a WCS 2.0 service. Each of these would need to be extended to meet the full needs. The goal would really be to lower the barrier to entry for UIs to discover and access EOSDIS data by choosing the WCS standard, and by surfacing measurements/variables and their subsetting capabilities, enabled by services.

Middleware Layer:

Any candidates for re-use which do not provide programmatic access, or exist in a modular component form are off the table for consideration as candidates for reuse in this layer. That leaves us with quite a short list of candidates.

CMR –The CMR is intended to enable broader use of NASA's EOS data by providing a more uniform view of NASA's substantial and diverse data holdings. Its two primary purposes within this architecture are to: 1) enable discovery and search of data and services; and 2) enable the data provider to manage service and variable level metadata. UMM-S and UMM-Var are important as part of the CMR service-oriented architecture, since these will enable the discovery and search of variable level data via services. As much as CMR will strive to autogenerate simple forms of this metadata, the Metadata Management Tool will ultimately need to be augmented to allow for manual curation of this metadata.

Services Layer:

EGI/ESI - EGI/ESI is primarily a framework for services and can provide whatever services are available in the registered tool adapters. That said, it is also a protocol for requesting services, and that protocol is currently focused on Subsetting (Parameter, Spatial, Temporal), Formatting (extensible set of formats) and Reprojection/Resampling of gridded data (extensible set of projection type and parameters).

The locally implemented HegService is a principal tool adapter for MODIS, AMSR, ASTR, GLAS and SMAP collections and provides for subsetting, formatting and reprojection. GdalService provides access to gdal_translate. SMAP-L1/L2 provides similar services for the SMAP L1/L2 collections.

The WCS Front-End component could be developed from portions of the the EGI/ESI WCS compliance work. The advantages of doing so will expand the limited collection-based ESI services model, and eventually remove the "order' vs. "services" paradigm by moving from the PUMP model to a simpler UMM-S model, using MMT as the services metadata curation client.

MRT Service API

This API provides access to the Moderate Resolution Imaging Spectroradiometer (MODIS) Reprojection Tool, allowing users to mosaic, subset, resample, and reproject tiled MODIS data available on the LP DAAC Data Pool or on the Level 1 and Atmosphere Archive and Distribution System (LAADS) FTP site.

WELD Service API

This API provides access to the mosaicing, subsetting, resampling, and reprojecting tools provided through the Web-Enabled Landsat Data (WELD) Interface, via an http service.

AppEEARS API

The Application for Extracting and Exploring Analysis Ready Samples (AppEEARS) API enables users to input specific geographic coordinates, such as field study sites or flux towers, and receive analysis-ready data in return from the Moderate Resolution Imaging Spectroradiometer (MODIS) sensor and Web-enabled Landsat Data (WELD) products.

LP DAAC has built an API which provides access to the AppEEARS web services (Product Service, Quality Service, and Sample Extraction Service) that allows users to extract pixel values from select MODIS data and CONUS and Alaska WELD (Web Enabled Landsat Data).

The functions of the main components of the AppEEARS service architecture are:

- Tilemap - Web service to convert Latitude and Longitude to tile, line, and sample. For MODIS, WELD, SRTM and ASTGTM datasets.
- Product Service - Returns the list of products and layers available.
- Quality Service - Returns the list product layers that have quality layers associated with them.
- Inventory - Service that queries LP DAAC product inventory. This service utilizes CMR.
- Sample Extraction Service - Returns the layer data values for the specified parameters.

AppEEARS currently has the ability to construct OPeNDAP style URL syntax, which closely resembles the style needed to support the consistent subsetting interface. In its current state of development AppEEARS calls the CMR API to discover the Collection level metadata (UMM-C). When Variable level metadata (UMM-Var) becomes available, it will also discover this from CMR.

AppEEARS has the potential to build the syntax necessary to send both simple subsetting requests and complex subsetting requests in an OPeNDAP style syntax. OPeNDAP URLs are complex. The service will need to know:

- Syntax – i.e. which OPeNDAP protocols (versions, formats, etc.)
- Product name - product service (at the CMR collection level)
- Layer name - product service (at the CMR variable level)
- Time - time services including time conversion (e.g. fromjulian, tojulian - but not actually julian dates)
- Geolocation – tilemap service for both tile and pixel values
  - Tile locations (H & V) – Data is aggregated over time by tile
  - Pixel location (Ydim & Xdim)

An example of an AppEEARS OPeNDPAP simple subsetting request is given below:

http://opendap.cr.usgs.gov/opendap/hyrax/MOD09Q1.005/h11v04.ncml.ascii?sur_refl_b01[683:1:683][2000:1:2010][2000:1:2010]

Recall that currently AppEEARS can only operate across the LP DAAC data pool, and can provide subsetting services for a limited range of data sets, e.g. MODIS, WELD and ASTER DEM. The development effort needed to bring it to the level of functionality needed to act as an enterprise-level service adapter would be to extend it to operate across other DAAC data pools, and to have the ability to recognize additional data sets.

ESDIS and HDF Tools

Within the services layer there is a need to perform various complex subsetting functions on the data sets, including: band subsetting, reprojection, resampling, regridding and mosaicing.

We anticipate the need to perform various types of file reformatting, including the conversion of HDF 4 and HDF 5 files to netCDF-4 or to convert HDF 4 and HDF 5 to CF-compliant versions of these formats. ESRI ArcGIS Server, for example, only works with CF-compliant NASA data products. So we expect that in order to capitalize on ESRI's configuration of WCS services, there would need to be a provision for this. Additionally, the options available in the consistent subsetting interface afford the user choices as to what format the data sets are to returned with, i.e. HDF4, HDF5, netCDF, GeoTIFF, KML, etc., so the DAAC services may need to perform this step using ESDIS or HDF tools, if not available within the service configuration.

Next, we surveyed each DAAC to determine which services were available today to understand the work that would be required by the providers in order to expose their current services as WCS compliant services.

DAAC Services

Table 1 shows the services available across the DAACs by service type.

DAAC	ASDC	ASF	CDDIS	GES DISC	GHRC	LAADS	LP DAAC	NSIDC	OB DAAC	ORNL	PO DAAC	SEDAC
FTP	X	x	x			x				x	X	X
HTTP/HTTPS		x		x	X	x	X	X	x	x	X	X
OGC (WCS, WFS, WMS)	X	x		x		x	X	x	x	x	X	X
OPeNDAP/Hyrax	X			x		x	X					
Unidata/ THREDDS										x	x	
ESI/Service Gateway	X						X	X				
AppEEARS							X					

Table 1. DAAC Services by Service Type

Note: FTP, HTTP/HTTPS and OGC (WMS, WFS, WCS) are protocol standards.

OPeNDAP/Hyrax and Unidata/THREDDS are service configurations. ESI/Service Gateway and AppEEARS are ESDIS or DAAC developed services.

DAACs which currently offer OGC (WMS, WFS, WCS) Services in various configurations:

ASDC, ASF, GES DISC, LAADS, LP DAAC, NSIDC, OB DAAC, ORNL, PO DAAC, SEDAC.

Note: Most of the DAACs which provide OGC (WCS) services are at WMS 1.x, WFS 1.x, WCS 1.x, but we expect this will change in the next 6-12 months as upgrades and patches are rolled in. It is recommended that this activity be coordinated by ESDIS.

Service Configurations comparison

With all the possible variations on service configurations and service standards, a simple analysis is required here to show what can be achieved in the current state and what will be realizable in the near future (6 – 9 months). For example, what does ESI offer compared to OPeNDAP compared to vendor provided configurations of OGC (WFS, WCS, etc.) services? Table 2 shows a feature comparison of services.

Feature	AppEEARS #	ESI/ Service Gate-Way #	ArcGIS Server OGC (WCS 2.0 w/EO-exts)	EOx-Server (open-source project for WCS 2.0 w/EO-exts)	MapServer	OPeNDAP (WCS 1.1.2)	OPeNDAP (WCS 2.0 w/EO-exts)	Rasdaman (WMS, WCS, WCS-T, WCPS, WPS) *
Spatial Subsetting	x	x	x	X	X	x(v1.13.0)	X	x
Temporal Subsetting	X	x	x	X	X	x(v1.13.0)	X	x
Variable Subsetting	X	x	x(v10.4 Templates)	X	X	x(v1.13.0)	X	x
Band Subsetting		x(HEG,GDAL)	x(v10.4 Templates)	X	X		x(v1.14.0)	x
Reprojection		x(HEG,GDAL)	x(v10.4 Templates)	X	X		x(v1.14.0)	x
Resampling		x(HEG,GDAL)	x(v10.4 Templates)	x(v0.x)	x(v7.0.x)		x(v1.14).0	x
Regridding		x(HEG, GDAL)	x(v10.4 Templates)	x(v0.x)	x(v7.0.x)		x(v1.14.0)	x
Mosaicing		x(HEG, GDAL)	x(v10.4 Templates)	x(v0.x)	x(v7.0.x)		x(v1.14.0)	X

Table 2. Feature Comparison of Services

 # - note that both AppEEARS and ESI/Service Gateway services are limited to specific data sets, and the subsetting capabilities are highly dependent on the characteristics of each data set.
**** - note that Rasdaman is limited to subsetting capabilities to data sets configured to that service, and none of these are EOS data sets currently.***

Data Layer:

The Data Layer is primarily composed of the Data Pools, containing archived data sets. Data set availability varies by DAAC.

7. Define a minimum set of services to recommend to the DAACs.

In order to identify a path forward we need to identify the minimum set of services to recommend to the DAACs to support the consistent subsetting interface. Based on the analysis performed so far, and a careful read of the OGC (WMS, WCS) service definitions (see Appendix B), we recommend the following:

The minimum set of services to recommend to the DAACs is: WMS 1.3.0 for mapping (visualization), WCS 2.0.1 (either implemented via OPeNDAP/Hyrax OGC, ESRI OGC ArcGIS Server 10.4.x, MapServer, or GeoServer) for coverages (data layers).

Currently, we can use the following service configurations for basic subsetting:

WCS 1.1.2 servers: available with the OPeNDAP/Hyrax v1.13.0 (release 1, or release 2).

WCS 2.0.1 requires the following service configurations:

- OPeNDAP/Hyrax v1.14.0 (next release). ArcGIS Server v10.4 (current release) or v10.5 (next release), MapServer 7.0.x, EO-Server 0.x, RASTERMAN server.

- ArcGIS Server v10.4 is WCS 2.0.1 compliant using templates, however relies on the data products to be in HDF 5 format and CF-compliant.

- ArcGIS Server v10.5 will need a customer request to have EO extension added to the server capabilities. Therefore, we recommend that ESDIS participates in the Beta Testing of ArcGIS Server v10.5.x. and request the EO extensions. (NASA has a site-wide license with ESRI).

We can use any of the following servers for complex subsetting:

- Currently, the only service configuration that is EO-X capable is EO-Server, which is based on MapServer 7.0.x, explicitly EO-Server 0.x.
- In the future, once the ESDIS task related to the development of the OPeNDAP in the cloud capability, has been completed, then the next release of Hyrax, v1.14.0 will include the EO extensions. ESRI does not foresee adding EO extensions to ArcGIS Server until v10.5.1. This will enable a richer client/UI experience towards the end of next year.

We discussed the utility of the WPS service with the OPeNDAP developers, and we agreed that this has no merit at this time, not until we have become confident with using the minimum WMS, WCS services recommended here.

We learned also that there is a separate ESDIS sponsored task with GDAL development team, which will eventually lead to the capability to perform simple and complex subsetting requests to be issued from a GDAL WCS client. We recommend that progress on this is tracked, and the same minimum standard is worked towards.

With the minimum service configuration on the WCS back-end recommended to be WCS 2.0.x, we would recommend that current efforts in EGI/ESI development towards WCS and programmatic access be extended to meet this standard. This would require the development of a WCS client on the front-end and extensive testing with new tool adapters on the back-end. The timing and deployment of service configurations at the DAACs would need careful coordination as capabilities become available.

The most important consideration is to ensure that each new capability meeting a distinct OGC standard, i.e. WCS 1.1.2, WCS 2.0.x, WCS 2.0.x with EO extensions, should be rolled out at the same time, until testing and evaluation results in certification of the new capabilities with each of the new subsetting components, and metadata models.

8. Future Work

We recommend the following next steps:

1. Extend CMR to implement the core UMM-Var design. This will enable the search and discovery of variables and provides a foundation for variable subsetting. Make the correspond changes in the UI layer, initially EDSC, to expose variables.

2. Prototype asynchronous responses in WCS to determine if it is feasible to use a URL in the getCoverage response to handle asynchronous processing.
3. Extend CMR to implement the core UMM-S design. This will enable the search and discovery of services. Make the correspond changes in the UI layer, initially EDSC, to expose services in a more consistent manner. This would include auto-generation of a simple UMM-S from existing WCS backends.
4. Provide the minimum set of WCS services to recommend to the DAACs. This will enable each provider to work towards WCS Back-end service capabilities. Work with service provider to extend services to support additional UMM-S metadata needs.
5. 4. Continue EGI/ESI WCS and Programmatic access development efforts. Combine this WCS Front-end API to the CMR Standards-based APIs, so it can be accessible via compatible client/UIs. This extends the current CMR architecture to a WCS standards-based API.
6. Extend UMM-S to support WCS 2.0.x compliant capability descriptions to enable a full, complex and automatable WCS Front-end/WCS Back-end service model.
7. Continue development AppEEARS API to work with the minimum service standard so that it can be easily discovered by WCS clients.

References

NASA DAAC User Research – Study conducted by Blink, 2016.

CSW Best Practices – Presentation developed under the ESDIS contract.

Duma, C., White Paper, “ECS Support for OGC Services in Release 8.1 and future releases”, 2012.

OG_05_06 Ticket, Deploy OGC Web Services
OG_S5_07 Ticket, OGC Interface To ECHO
WD-S4_06 Ticket, Earth Science Gateway (ESG) Pilot

Abbreviations

AppEEARS – Application for Extracting and Exploring Analysis Ready Samples
ASDC – Atmospheric Sciences Data Center
CEOS – Committee on Earth Observation Satellites
CSW – Catalog Services for the WB
CMR – Common Metadata Repsoitory
CWIC – CEOS WGISS Integrated Catalog
DAAC – Data Active Archive Center
DAP – Data Access Protocol
ECS- EOSDIS Core System
EDSC – Earthdata Search Client
ESI – EOSDIS Service Interface
ESIP – Federation of Earth Science Information Partners
ESDIS – NASA's Earth Science
EOSDIS – Earth Observing System Distributed Information System
HEG – HDF-EOS-to-GeoTIFF Conversion Tool
LP DAAC – Land Processes Distributed Active Archive Center
OGC – Open Geospatial Consortium
HTTP – Hypertext Transfer Protocol
OPeNDAP – Open Source Project for a Network Data Access Protocol
TDS – THREDDS Data System
THREDDS – Thematic Real-time Environmental Distributed Data Services
UMM-C – Unified Metadata Model - Collection
UMM-G - Unified Metadata Model - Granule
UMM-S - Unified Metadata Model - Services
UMM-Var - Unified Metadata Model - Variable
UMM-Vis - Unified Metadata Model - Visualization
WSGI – Web Server Gateway Interface
WGISS – Working Group on Information Systems and Services
WCPS - Web Coverage Processing Service
WCS – Web Coverage Services
WCS-T - Web Coverage Service-Transactional
WMS – Web Mapping Services
WPS - Web Processing Service

Appendix A Tools, Services Candidates for Reuse

We provide a survey of the tools which are currently supported by the NASA ESDIS project and the HDF Group, to understand where key functions which may be relevant to the end-to-end services architecture may exist in their current state.

Tools supported by the NASA ESDIS Project

The tools currently supported by NASA ESDIS, via the EED-2 contract are:

- HEG – HDF-EOS-to-GeoTIFF Conversion Tool (HEG) allows users to reformat, re-project, and perform stitching/mosaicing and subsetting operations on HDF-EOS objects, e.g. converts HDF-EOS to GeoTIFF.
- HDF-EOS plugin for HDFView – entends HDFView functionality for browsing any HDF-EOS file.
- Heconvert – converts HDF-EOS Version 2.X files to HDF-EOS Version 5.x
- MTD_Toolkit – allows EOSDIS extended data providers to format their products in ECS standard formats.
- SDP_Toolkit – a set of tools to manage the metadata that are generated with each product.

Tools supported by the HDF Group

The tools supported by the HDF Group are:

- HDFView – a visual tool for browsing and editing HDF4 and HDF5 files.
- H4H5TOOLS – HDF4 to HDF5 Conversion Tools
- HDF-EOS5 - Augments HDF-EOS5 files for netCDF-4
- HDF4 OPeNDAP Handler – serves HDF4 via OPeNDAP
- HDF5 OPeNDAP Handler – serves HDF5 via OPeNDAP
- HDF-EOS2 Dumper – dumps information of HDF-EOS2 files
- H4CF Conversion Toolkit – provides CF-compliant access to HDF4 and HDF-EOS2 files
- HDF4 File Content Map Writer – a utility that creates an HDF4 file content map in XML.

Appendix B Service Definitions

Analysis of OGC Web Services

Current State:

WMS

Four versions of the WMS specification have been published so far. They are v1.0.0, v1.1.0, v1.1.1, and v1.3.0 (most recent). We'll focus on the two most recent here.

WMS services published to e.g., ArcGIS Server, support the following operations:

- Requesting metadata about the service (GetCapabilities)
- Requesting a map image (GetMap)
- Requesting information about features in the map (GetFeatureInfo [optional])
- Requesting user-defined styles (GetStyles)
- Requesting legend symbols (GetLegendGraphic)

WMS 1.1.1 provides GetCapabilities, and GetMap, GetFeatureInfo (optional), DescribeLayer (optional), GetLegendGraphic (optional)

- GetCapabilities provides all service-level metadata and a brief description of the data.
- GetMap retrieves a map image for a specified area and content
- GetFeatureInfo (optional) retrieves the underlying metadata, including geometry and attribute values, for a pixel location on a map
- DescribeLayer (optional) indicates the WFS or WCS to retrieve additional information about the layer
- GetLegendGraphic (optional) retrieves a generated legend for a map.

WMS 1.3.0

For a WMS service to support different versions of WMS protocol—for example, 1.0.0, 1.1.0, 1.1.1, and 1.3.0—the service must have one capabilities file for each version of WMS which is needed to support. This can be accomplished on the same server instance.

The major differences between versions 1.1.1 and 1.3.0 are:

- In 1.1.1 geographic coordinate systems specified with EPSE namespace are defined to have an axis ordering of longitude/latitude. In 1.3.0 the ordering is latitude/longitude.
- In the GetMap operations the srs parameter is called crs. (SRS is spatial reference system; CRS is coordinate reference system).
- In the GetFeatureInfo operation the x and y parameters are called i and j.

WCS

WCS offers the following capabilities:

- Service Discovery
- Data Discovery
- Data Services
- Data Ordering

What options do they have?

1. GetCapabilities: The *request* provides a brief description of the coverages (i.e. data layers or variables. The *response* contains a list of the server's data, as well as valid WCS operations and parameters
2. DescribeCoverage: The *request* provides a more detailed information about the selected service. The *response* contains an XML document that fully describes the request coverages.
3. GetCoverage: The *request* provides actual data. The *response* contains a coverage in a well-known format, i.e. GML, netCDF, KML etc. A WCS GetCoverage request is like a WMS GetMap request, but with several extensions to support the retrieval of coverages i.e. raster data sets.

A WCS service returns data in a format that can be used as input for analysis and modeling. This is in contrast with the OGC Web Map Service (WMS), which only returns a picture of the data. The raster datasets made available through WCS services are referred to as coverages.

Client applications work with a WCS service by appending parameters to the service's URL.

WCS 1.1.2 offers the following operations:

- Request service-level metadata and a brief description of the data (GetCapabilities)
- Request a full description of one or more coverages (DescribeCoverage)
- Request a coverage in a well-known coverage format (GetCoverage)

WCS 1.1.2 offers simple susbetting functions (spatial, temporal and variable subsetting). OPeNDAP/Hyrax Version 1.13 (r 1, r2) comply to the OGC WCS 1.1.2 standard.

WCS 2.0 offers significant improvement over WCS 1.1.2 to meet the needs of the Earth Observation community.

The OGC Working Group is currently performing Testbed evaluations of an Earth Observation (EO) Application Profile for WCS 2.0 (EO-WCS) (see public RFC on EO-WCS). For an implementation please refer to the Open Source project EOxServer which already implements this proposed EO-WCS based on MapServer. ESRI ArcGIS Server, versions 10.4 and higher, support WCS 2.0.

WCS 2.0.1 services support the following OGC extensions:

- Service extensions: Scaling, Interpolation, Range Subsetting, and CRS
- Protocol extensions: KVP/Get and XML/Post
- Format encoding extensions: GeoTIFF

This means that to a limited extent, some of the simple susbetting functions (variable subsetting) the complex subsetting functions (regridding, reprojection, mosaicing) are supported in WCS 2.0.1. It must be stressed that this is highly dependent on data set type, and requires thorough testing and evaluation.

If using a multidimensional data set in WCS 2.0.1, it's recommended to create a mosaic dataset using corresponding raster types (e.g. netCDF, HDF, GRIB).

Variables (e.g. temperature, salinity, speed, etc.) are mapped to rangeType in the coverage description and accessed through the range subsetting.

Dimensions (e.g. x, y, t and z) are mapped to domainSet and accessed through the dimension trimming and slicing.

Interpolation is supported only on the x and y dimensions.

Limitations and issues

What are the limitations and issues relating to OCG services (i.e. WMS, WCS)?

- Scaling to large archives. If any of the ECS DAACs were to return a list of all granules as a response to a GetCapabilities request, the resulting XML file would be very large and not appropriate for return as the result of a synchronous call. However, this performance issue can be alleviated by using an asynchronous call and adding nodes to include additional data pools.

- Lack of support towards standardized of the predefined WCS extensions. Currently many server configurations do not fully implement WCS 2.0.x. Most vendors are currently participating in Testbed 12 activities, and most code bases only exist in an experimental state.

- Not all vendors support WCS 2.0, although most support 1.x. Many of the DAACs have a capability which is at WCS 1.x. Likewise, most vendors support WMS 1.1.1 and 1.3.0, so it is more likely that each of the DAACs can stand up WMS 1.3.0 compliant services today. However, in order to meet the WCS 2.0 standards compliance, there would need to be additional investment in implementation of a WCS 2.0 compliant service configuration, e.g. OPeNDAP/Hyrax or ArcGIS for Server. Note: OPeNDAP/Hyrax version 1.14 will include support for WCS 2.0.x, but will not be available until around 9 – 12 months.

Extension of WCS 2.0 Earth Observation

WCS 2.0 EO offers a set of capabilities, designed specifically for EO coverages (data sets).

EO-AP provides the following operations that are applicable to EO data:

- GetCapabilities: is an extension of WCS 2.0.0 core. The *request* provides the option to retrieve only specific sections from the server capabilities such as OperationMetadata,

CoverageSummary or DatasetSeriesSummary. The *response* contains availability of EO-AP as an extension. It also provides additional coverage types and response sections such as OperationMetadata, CoverageSummary and DatasetSeriesSummary.

- DescribeCoverage: the *request* is unchanged from the WCS core specification. If called on a stitched mosaic it returns the description of the mosaic. The response contains additional coverage metadata under the <*wcseo:EOMetadata*> element.
- GetCoverage: the *request* is unchanged from the WCS core specification except that for EO Coverages slicing is disallowed as it would leave the <*wcseo:EOMetadata*> undefined.. The response is extended in two respects:
    - The coverage returned contains exactly one metadata element holding the <*wcseo:EOMetadata*> record
    - The lineage component of the <*wcseo:EOMetadata*> record returned consists of the pre-existing lineage sequence plus one element appended which describes the *GetCoverage* request on hand.
- DescribeEOCoverageSet: is a new EO-AP specific operation which provides spatio-temporal search. The *request* submits a Dataset Series, Stitched Mosaic, or Dataset identifier together with a spatio-temporal subsetting criterion. The spatial constraint is expressed in WGS84, the temporal constraint in ISO 8601. If called on a stitched mosaic it returns a set of descriptions of each Dataset used in the mosaic. The *response* to a successful request on a Dataset Series consists of a (possibly empty) set of descriptions of Datasets and Stitched Mosaics and a (possibly empty) set of descriptions of Dataset Series. The response to a successful request on a Stitched Mosaic consists of a (possibly empty) set of descriptions of Datasets. In any case, the result items are those which are (i) referred to by the object submitted and (ii) matched by the bounding box. The type of matching – contains or overlaps – is specified in the request.

In addition to the core WCS trimming and slicing services, the EO-AP supports WCS 2.0 Predefined Extensions such as band subsetting, scaling and interpolation, and Coordinate Reference System (CRS).

Service Configurations

OPeNDAP

OPeNDAP is the Open source Project for a Network Data Access Protocol, and is a protocol for accessing distributed scientific data (also known as DODS DAP).

Pydap is a pure Python library implementing the Data Access Protocol, also known as DODS or OPeNDAP. Pydap can be used as a client to access hundreds of scientific datasets in a transparent and efficient way, through the Internet, or as a server to easily distribute data from a variety of different formats. This simple server, which comes bundled with Pydap, is implemented as a WSGI application, and requires the installation of data handlers, e.g. to handle netCDF files. Pydap can also be configured to use Apache servers.

Hyrax is OPeNDAP's own software for server-side realization of the DAP2 and DAP4 protocols.

Hyrax has been updated to include OPeNDAP HDF 4 and HDF 5 handlers, i.e. SMAP Level 1, 3 and 4 products can now be handled.

Updates to the OPeNDAP services

In recent months some encouraging updates have been made to the OPeNDAP services capabilities, as discussed in the *OPeNDAP Workshop 2016 – Emphasizing Advances for EOSDIS, ESIP Summer Meeting 2016*[2]. These include the following features:

- Pushed Aggregations – aggregation of data files into virtual datasets.
- Pulled Aggregations – aggregation of data files – users may invoke a single query against user-specified lists of thousands of source granules simultaneously.

- Extended Web Services - The DAP2 and DAP4 specifications each define a general-purpose data model sufficient for building data services that offer selective access to meaningful data holdings. These models embrace both array and tabular data structures, along with means for assigning names, data types, array ranks and other attributes (i.e., metadata). Additionally, the DAP4 data model permits named hierarchical groups of variables and other entities.

- Recent EOSDIS-sponsored advances in Hyrax include support for data-access from clients that are HTTP-based, JSON-based or OGC-compliant (especially WCS and WMS), support for DAP4, use and extension of server-side computational capabilities, and several performance-affecting matters.

- OGC (WMS, WCS). Improvements in WMS and WCS standards compliance include WMS 1.1.1 to 1.3.0 WCS 1.0.1 to 2.0.1.

- Webification - Improvements to include addition of w10n improvements designed by Jet Propulsion Laboratory to deliver responses in JSON.

- Extension of server-side –embracing non-rectangular grids (UGIDS).

- Compatibility between OPeNDAP's Hyrax and Unidata's THREDDS Data Server (TDS) along with other performance enhancements.

Work under the NASA contract number NNG15HZ39C includes development of OPeNDAP capabilities by Hyrax developers, namely improvements to WCS 2.0 server side capabilities including addition of EO-extensions.

Output from Testbed 12 includes:

- Handling of 2D + time
- Handling of 3D + time

---

[2] *OPeNDAP Workshop 2016 available at* https://2016esipsummermeeting.sched.org/event/6xd4/opendap-workshop-2016-emphasizing-advances-for-eosdis

We expect that as a result of Testbed 12 activities there will be a subsequent Hyrax server release which comprises EO extensions, e.g. band subsetting, scaling and interpolation etc.

Limitations and issues
User Authentication – not able to take advantage of URS. This would need to be handled in the middleware client.

Examples of WMS, WCS requests for OPeNDAP (Hyrax)

GetMap

This request will return a TRMM 3B43_V6_MONTHLY WMS data layer for map display in PNG format.
Example[3]

- http://g0dup05u.ecs.nasa.gov/cgi-bin/wms_ogc?SERVICE=WMS&VERSION=1.1.1&REQUEST=Getmap&layers=buemarble,TRMM_3B43_V6_MONTHLY&BBOX=180,-50,180,50&TIME=2007-12-01&format=png&transparent=true&WIDTH=1500&HEIGHT=800

Example
This request will return a AIRX3STM_TOTO3_A WMS data layer for map display.

- http://g0hep12u.ecs.nasa.gov/mapserv-bin/wms_ogc?TARGET_SRS=ESPG:4326&Service=WMSVERSION=1.1.1&REQUEST=GetMap&SRS=EPSG:4326&WIDTH=768&HEIGHT=512&BBOX=-180,90&LAYERS=AIRX3STM_TOTO3_A.coastline

GetCapabilities
This request will return all service-level metadata and a brief description of the data in GML format.
Example

- http://acdisc.sci.gsfc.nasa.gov/daac-bin/wcsNO2?service=wcs&version=1.0.0&request=getCapabilities

DescribeCoverage
This request will return a full description of one or more coverages within the service in GML format.
Example

- http://acdisc.sci.gsfc.nasa.gov/daac-bin/wcsNO2?service=wcs&version=1.0.0&request=describeCoverage

GetCoverage
This request will return a coverage of NO2Total data in one of the supported formats: netCDF.

[3] GES DISC WMS Services: http://disc.sci.gsfc.nasa.gov/services/ogc_wms

Example

- http://acdisc.sci.gsfc.nasa.gov/daac-bin/wcsNO2?service=WCS&version=1.0.0&request=getCoverage&CRS=WGS84&resx=0.5&resy=0.5&coverage=NO2Total&bbox=-179.75,-89.75,179.75,89.75&TIME=2006-08-01/2006-08-04&format=netCDF

ESRI ArcGIS for Server

ESRI provides server-side configurations for (OGC) Web Coverage Service (WCS) and offers an open specification for sharing raster datasets on the web. ArcGIS for Server allows the publication of WCS services from imagery collections, maps, or geodatabases that contain raster data sets, including raster data sets sourced from satellite data products.

ArcGIS for Server is currently at v10.4

V10.4 comprises the following functionality:

WCS services published to ArcGIS Server support the following operations:

- Request service-level metadata and a brief description of the data (GetCapabilities)
- Request a full description of one or more coverages (DescribeCoverage)
- Request a coverage in a well-known coverage format (GetCoverage)

WCS 2.0.1 services also support the following OGC extensions:

- Service extensions: Scaling, Interpolation, Range Subsetting, and CRS
- Protocol extensions: KVP/Get and XML/Post
- Format encoding extensions: GeoTIFF

WCS 2.0.1 extensions can be enabled when publishing mosaic datasets as image services.

ArcGIS Server understands the WCS syntax necessary to receive and respond to simple subsetting requests and complex subsetting requests in an OPeNDAP style syntax. See examples of WCS requests below.

Examples of WCS requests for ESRI ArcGIS for Server (10.4 Linux)

GetCapabilities
This request will return all service-level metadata and a brief description of the data in GML format.
Examples[4]

- WCS 1.0.0:
  `http://gisserver.domain.com:6080/arcgis/services/World/Temperature/ImageServer/WCSServer?SERVICE=WCS&VERSION=1.0.0&REQUEST=GETCAPABILITIES`
- WCS 2.0.1:
  `http://gisserver.domain.com:6080/arcgis/services/demo/OceanM`

[4] ArcGIS for Server documentation available at: WCS Services available at: http://server.arcgis.com/en/sever/latest/publish-services/linux/wcs-services.htm

odel_HYCOM/ImageServer/WCSServer?request=GetCapabilities&service=WCS&version=2.0.1

DescribeCoverage

This request will return a full description of one or more coverages within the service in GML format.

Examples

- WCS 1.0.0: http://gisserver.domain.com:6080/arcgis/services/World/Temperature/ImageServer/WCSServer?SERVICE=WCS&VERSION=1.0.0&REQUEST=DescribeCoverage&COVERAGE=1
- WCS 2.0.1: http://gisserver.domain.com:6080/arcgis/services/demo/OceanModel_HYCOM/ImageServer/WCSServer?request=DescribeCoverage&service=WCS&version=2.0.1&coverageid=Coverage1

GetCoverage

This request will return a coverage in one of the supported formats: GeoTIFF, NITF, HDF, JPEG, JPEG2000, and PNG.

Example

- http://gisserver.domain.com:6080/arcgis/services/Earthquakes/CaliforniaDEM/ImageServer/WCSServer?SERVICE=WCS&VERSION=1.1.1&REQUEST=GetCoverage&IDENTIFIER=1&FORMAT=image/GeoTIFF&BOUNDINGBOX=-13845885.5590586,3812317.53467798,-12705764.6522576,5131267.73075641,urn:ogc:def:crs:EPSG::54004&RangeSubset=Field_1:nearest[BAND[1]]&GridBaseCRS=urn:ogc:def:crs:EPSG::54004&GridCS=urn:ogc:def:crs:EPSG::54004&GridType=urn:ogc:def:method:WCS:1.1:2dGridIn2dCrs&GridOrigin=-13845885.5590586,5131267.73075641&GridOffsets=2879.09319899244,-2879.80392156863

Time and images examples

For WCS services originating from mosaic datasets, TIME and IMAGES parameters can be added.

Examples

Time

TIME—Supports time queries by specifying a time instance or time range. The format can be YYYYMMDDHHMMSS, YYYYMMDD, YYYYMM, or YYYY.

TIME=1999 (time at year 1999):

- http://gisserver.domain.com:6080/arcgis/services/folder/service/ImageServer/WCSServer?SERVICE=WCS&VERSION=1.0.0&REQUEST=GetCoverage&COVERAGE=1&FORMAT=GeoTIFF&BBOX=-180,-55.5,180,83.5&WIDTH=720&HEIGHT=278&CRS=EPSG:4326&INTERPOLATION=nearest%20neighbor&Band=1,2,3&TIME=1999

TIME=1980/2010 (time from year 1980 to 2010):

- http://gisserver.domain.com:6080/arcgis/services/folder/service/ImageServer/WCSServer?SERVICE=WCS&VERSION=1.0.0&REQUEST=

GetCoverage&COVERAGE=1&FORMAT=GeoTIFF&BBOX=-180,-55.5,180,83.5&WIDTH=720&HEIGHT=278&CRS=EPSG:4326&INTERPOLATION=nearest%20neighbor&Band=1,2,3&TIME=1980/2010

TIME=1999-12-31T14:30:30/2000-01-01T14:30:30:

- http://gisserver.domain.com:6080/arcgis/services/folder/service/ImageServer/WCSServer?SERVICE=WCS&VERSION=1.0.0&REQUEST=GetCoverage&COVERAGE=1&FORMAT=GeoTIFF&BBOX=-180,55.5,180,83.5&WIDTH=720&HEIGHT=278&CRS=EPSG:4326&INTERPOLATION=nearest%20neighbor&Band=1,2,3&TIME=1999-12-31T14:30:30/2000-01-01T14:30:30

This will request a slice of salinity data in GeoTIFF format; the data covers -20 ~ 20 latitude/longitude and at a particular time and z.

Example

- http://gisserver.domain.com:6080/arcgis/services/demo/OceanModel_HYCOM/ImageServer/WCSServer?request=GetCoverage&service=WCS&version=2.0.1&coverageId=coverage1&scalefactor=2&rangesubset=salinity&subset=y,http://www.opengis.net/def/crs/EPSG/0/4326(-20,20)&subset=x,http://www.opengis.net/def/crs/EPSG/0/4326(-20,20)&subset=StdTime,http://www.opengis.net(2014/04/07 00:00:00)&subset=StdZ,http://www.opengis.net(0)&format=image/tiff

V10.5 of ArcGIS for Server is about to be released which promises even better support for WCS 2.0.1. ESRI has confirmed that they are looking into implementing EO profile next year (10.5.1 or later). Note this version is available in an Enterprise version, which comprises each component needed for a service-oriented architecture: Geoportal (a CSW 3.0/OpenSearch compliant catalog service) and ArcGIS for Server.

ESRI is currently participating in OGC testbed 12.

This participation includes the testing and evaluation of the following functions:

- Web app builder based viewer for WMS, WCS, WMTS, KML, and WFS
- Supporting CSW/OpenSearch profile
- Supporting OGC Context document in the web app builder – being able to read/write OGC context documents – geoJSON encoding.

What additions are needed?

If ArcGIS Server web services does not offer the precise functionality or business logic needed, it can be expanded through server object extensions (SOEs). An SOE extends the base functionality of a web service using ArcObjects, the vast suite of components on which the ESRI family of products is built. SOEs are an advanced option requiring custom development, but once written they are easy to deploy to the web server. No special software other than ArcGIS for Server is required to run an SOE.

Templates

Image services support server-side processing through some default functions that are supported through REST, but also by publishing image services and associated raster function templates that can be accessed and used via REST and ArcGIS for Desktop applications, such as ArcMap.

To implement the raster function template (.rft.xml), it must be first created using the Raster Function Template Editor, then simply add the .rft.xml file to the image service when it is published.

Hardware configuration

ESRI recommended the minimum 2x4 core load balanced configuration for getting started, but that would need to scale to more cores depending on demand for the data from users, otherwise a performance degradation would be experienced.

Limitations and issues

User Authentication – not able to take advantage of URS. This would need to be handled in the middleware client. ESRI Subset Services require all data sets to be CF-compliant in order to function correctly.

MapServer

MapServer is an open source project whose purpose is to display dynamic spatial maps over the Internet. MapServer configuration can be found at: http://mapserver.org/ogc/wcs_server.html and requires PROJ.4, GDAL and linxml2. Mapfile configuration includes:

- Layer <> Coverage
- Mandatory Settings
- Metadata
- Output-format declaration.

MapServer offers the following capabilities:

- Support for display and querying of hundreds of raster, vector and database formats
- Ability to run on different systems (Windows, Linux, MacOSX, etc.)
- Support for popular scripting languages and development environments (PHP, Python, Perl, Ruby, Java, .NET).
- On-the-fly projections
- High quality rendering
- Fully customizable application output
- Many ready-to-use Open Source application environments

MapServer supports spatial subsetting in X and Y dimensions, band subsetting, multipart/mixed responses including the GML coverage description, resizing, reprojection, resampling, and protocol binding: KVP and XML/POST.

Examples[5]

GetCapabilities
http://www.yourserver.com/wcs?SERVICE=WCS&ACCEPTVERSIONS=1.0.0,2.0.1
&REQUEST=GetCapabilities
http://www.yourserver.com/wcs?SERVICE=WCS&REQUEST=GetCapabilities

DescribeCoverage 2.0
http://www.yourserver.com/wcs?SERVICE=WCS&VERSION=2.0.1
&REQUEST=DescribeCoverage&COVERAGEID=grey

GetCoverage 2.0 image/tiff full
http://www.yourserver.com/wcs?SERVICE=WCS&VERSION=2.0.1
&REQUEST=GetCoverage&COVERAGEID=grey&FORMAT=image/tiff

GetCoverage 2.0 multipart/related (GML header & image/tiff) full
http://www.yourserver.com/wcs?SERVICE=WCS&VERSION=2.0.1
&REQUEST=GetCoverage&COVERAGEID=grey&FORMAT=image/tiff
&MEDIATYPE=multipart/related

GetCoverage 2.0 image/tiff trim x y both in imageCRS
http://www.yourserver.com/wcs?SERVICE=WCS&VERSION=2.0.1
&REQUEST=GetCoverage&COVERAGEID=grey&FORMAT=image/tiff
&SUBSET=x(10,200)&SUBSET=y(10,200)
&SUBSETTINGCRS=imageCRS

GetCoverage 2.0 reproject to EPSG 4326
http://www.yourserver.com/wcs?SERVICE=WCS&VERSION=2.0.1
&REQUEST=GetCoverage&COVERAGEID=grey&FORMAT=image/tiff
&SUBSET=x(-121.488744,-121.485169)
&SUBSETTINGCRS=http://www.opengis.net/def/crs/EPSG/0/4326

Future work includes OGC CITE testing, implementation of future extensions e.g. WCS-T, further coverage types, e.g. ReferencableGridCoverage, WCS 2.0 clients e.g. in GDAL, and support for INSPIRE download service.

GeoServer

GeoServer is an open source server written in Java that allows users to share and edit geospatial data. Designed for interoperability, it publishes from any major spatial data source using open standards. GeoServer is available at http://geoserver.org

[5] MapServer 7.0.2 Documentation available at: http://mapserver.org/ogc/wcs_server.html

GeoServer offers the following capabilities:

- Support for shapefiles, Oracle databases, ArcSDE integration and other types of spatial data (vector and raster data)
- Uses GML (Geographic Markup Language)
- Support for WMS (Web Mapping Service)
- Support for WFS 2.0 (Web Feature Service)
- Support for WCS 2.0 Coverages, including structured coverages (multidimensional grids)
- Support for WPS (Web Processing Services)
- Support for CSW (Catalog Services for the Web)

Examples[6]

GetCapabilities:
http://localhost:8080/geoserver/ows?service=WCS&version=2.0.1&request=GetCapabilities

DescribeEOCoverageSet
http://localhost:8080/geoserver/ows?service=WCS&version=2.0.1&request=DescribeEOCoverageSet&eoId=nurc__watertemp_dss

GetCoverage
Any of the inner coverages can be then retrieved via a standard GetCoverage, even if it's not directly part of the capabilities document, for example, to retrieve the first granule in the watertemp layer the request would be:
http://localhost:8080/geoserver/ows?service=WCS&version=2.0.1&request=GetCoverage&coverageId=nurc__watertemp_granule_watertemp.1

GeoServer allows the display of spatial information and the provision for creating maps in a variety of formats. OpenLayers a free mapping library is integrated into GeoLayers, which enables map generation to be quick and easy. GeoServer is built on GeoTools, an open source Java GIS toolkit.

GeoServer is free software and can display data in any of the popular mapping applications, e.g. Google Maps, ArcGIS Earth, Microsoft Virtual Earth, In addition GeoServer can connect with other traditional GIS architectures, e.g. ESRI ArcGIS.

EO-Server w/EO-AP extensions

EOxServer is an open source software for registering, processing, and publishing Earth Observation (EO) data via Web Services. EOxServer is written in Python and relies on widely-used libraries for geospatial data manipulation. It can be found at: http://eoxserver.org

[6] GeoServer 2.6.2 documentation available at:
http://docs.geoserver.org/2.6.2/user/services/wcs/index.html

The EOxServer service model is designed to deliver (representations of) EO data using open standard web service interfaces as specified by the Open Geospatial Consortium (OGC).

EOxServer implements three versions of the WCS specification:

- Version 1.0.0
- Version 1.1.0
- Version 2.0.1 including the Earth Observation Application Profile (EO-WCS)

Each of these version supports three operations:

- GetCapabilities – returns an XML document describing the available coverages (and Dataset series)
- DescribeCoverage – returns an XML document describing a specific coverage and its metadata
- GetCoverage – returns (a subset of) the coverage data

The WCS 2.0 EO-AP (EO-WCS) adds an additional operation:

- DescribeEOCoverageSet – returns an XML document describing (a subset of) the datasets contained in a Rectified Stitched Mosaic of Dataset Series.

EOxServer now fully supports the following WCS 2.0.1 service extensions:

- Scaling Extension
- Interpolation Extension
- RangeSubsetting Extension
- CRS Extension
- GeoTIFF Encoding Extension

EOxServer also supports WCS-T and WPS service extensions.

Examples[7]

Domain sub-setting

```
http://someurl/wcs_demo? SERVICE=WCS& VERSION=1.0.0&
REQUEST=GetCoverage& COVERAGE=demo_coverage& FORMAT=image/tiff&
BBOX=300,400,400,700&
RESX=1&RESY=1&
CRS=ImageCRS
```

Domain sub-setting

```
http://someurl/wcs_demo? &SERVICE=WCS
&VERSION=1.1.0 &REQUEST=GetCoverage &IDENTIFIER=demo_coverage
&FORMAT=image/tiff &BOUNDINGBOX=300,400,400,700,
urn:ogc:def:crs:OGC::imageCRS
&GridCS=urn:ogc:def:crs:OGC::imageCRS
```

[7] EOx-Server documentation available at:
http://docs.eoxserver.org/en/stable/users/basics.html#eoxserver-documentation

```
&GridType=urn:ogc:def:method:WCS:1.1:2dGridIn2dCrs
&GridOrigin=300,400
&GridOffsets=1,-1
```

Domain sub-setting

```
http://someurl/wcs_demo? SERVICE=WCS& VERSION=2.0.0&
REQUEST=GetCoverage& COVERAGEID=demo_coverage& FORMAT=image/tiff&
SUBSET=x(300,400)& SUBSET=y(400,700)
SUBSET=long,http://www.opengis.net/def/crs/EPSG/0/4326(-40,-35)&
SUBSET=lat,http://www.opengis.net/def/crs/EPSG/0/4326(-10,0)
```

XML/POST binding

```
<wcs:GetCoverage xmlns:wcs="http://www.opengis.net/wcs/2.0"
service="WCS" version="2.0.0">
<wcs:Format>image/tiff</wcs:Format>
<wcs:CoverageID>grey</wcs:CoverageID> <wcs:DimensionTrim>
<wcs:Dimension>x</wcs:Dimension> <wcs:TrimLow>300</wcs:TrimLow>
<wcs:TrimHigh>400</wcs:TrimHigh>
</wcs:DimensionTrim>
</wcs:GetCoverage>
```

Band sub-setting

```
http://someurl/wcs_demo? SERVICE=WCS&
VERSION=2.0.0& REQUEST=GetCoverage& COVERAGEID=demo_coverage&
FORMAT=image/png& RANGESUBSET=Band4,Band15,Band6
RANGESUBSET=4,15,6
```

multipart/mixed Request

```
http://someurl/wcs_demo? SERVICE=WCS& VERSION=2.0.0&
REQUEST=GetCoverage& COVERAGEID=demo_coverage& FORMAT=image/tiff&
MEDIATYPE=multipart/mixed
```

multipart/mixed Responses

```
--wcs
Content-type: text/xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<gmlcov:RectifiedGridCoverage ... gml:id="demo_coverage">
<gml:boundedBy>...</gml:boundedBy>
<gml:domainSet>...</gml:domainSet> <gml:rangeSet>
<gml:File>
<gml:rangeParameters xlink:href="coverage/out.tif"
xlink:role="image/tiff" xlink:arcrole="fileReference"/>
<gml:fileReference>coverage/out.tif</gml:fileReference>
<gml:fileStructure/> <gml:mimeType>image/tiff</gml:mimeType>
</gml:File>
</gml:rangeSet> <gmlcov:rangeType>...</gmlcov:rangeType>
</gmlcov:RectifiedGridCoverage>

Content-Type: multipart/mixed; boundary=wcs
--wcs
Content-type: text/xml
```

```
...
--wcs
Content-Type: image/tiff
Content-Description: coverage data Content-Transfer-Encoding:
binary
Content-ID: coverage/out.tif
Content-Disposition: attachment; filename=out.tif
... --wcs--
```

Reprojecting

```
http://someurl/wcs_demo?
SERVICE=WCS&
VERSION=2.0.0&
REQUEST=GetCoverage& COVERAGEID=demo_coverage& FORMAT=image/png&
RANGESUBSET=Band4,Band15,Band6&
OUTPUTCRS=http://www.opengis.net/def/crs/EPSG/0/3035

SUBSET=x,http://www.opengis.net/def/crs/EPSG/0/3035(-3000000,0)
```

Resizing

```
http://someurl/wcs_demo? SERVICE=WCS& VERSION=2.0.0&
REQUEST=GetCoverage& COVERAGEID=demo_coverage& FORMAT=image/tiff&
SIZE=x(64)&
SIZE=y(117)
RESOLUTION=x(0.279)& RESOLUTION=y(0.279)
```

Interpolation

```
http://someurl/wcs_demo? SERVICE=WCS& VERSION=2.0.0&
REQUEST=GetCoverage& COVERAGEID=demo_coverage& FORMAT=image/tiff&
SIZE=x(64)&
SIZE=y(117)& INTERPOLATION=NEAREST
INTERPOLATION=AVERAGE or BILINEAR
```

EOxServer supports an MIT-style license, it is written in Python, and based on MapServer, Django, GDAL, SpatialLite or PostGIS, and PROJ.4. GeoServer adopts MapServer policies and procedures.

Rasdaman

The rasdaman ("raster data manager") system offers flexible, scalable access to multi-dimensional sensor, image, and statistics data. Database-style query techniques enable navigation, extraction, aggregation and general ad-hoc analytics on unlimited data volumes.

Rasdaman supports all the current raster-relevant OGC standards: WMS, WCS, WCS-T, WCPS, and WPS.

Web Mapping Service [8](WMS) and *Web Coverage Service* (WCS) definitions have been discussed earlier.

Web Coverage Service-Transactional (WCS-T) establishes how to upload complete coverages to a server or modify existing coverages. It provides options to insert (Add action) and delete (Delete) coverages which represent EO datasets. WCS-T also supports asynchronous request processing. The *wsct:ResponseHandler* element shall contain a URL of the remote response handler where the response is sent once the asynchronous processing is finished. Currently WCS-T implementation supports HTTP and FTP URL schemas for the response handler.

Web Coverage Processing Service (WCPS) provides an XQuery-style filtering and processing language on multi-dimensional raster coverages.

Web Processing Service (WPS) allows providers to offer any kind of algorithmic processing, in this case: on coverages. It provides rules for standardizing how requests and responses for invoking geospatial processing services, such as polygon overlay, as a Web service.

[8] Rasadaman documentation available at: http://standards.rasdaman.com/