

## **EPA's Watersheds-based Monitoring and Interoperable Data Platforms Project**

### **Lessons Learned**

December 2016

#### **Introduction**

Water quality agencies at many levels (local, state and national) deploy water quality sensors and collect continuous data. Some agencies already make their data available publicly via web services. However, unless these data providers and available datasets are known to the individual seeking data, this information can be difficult to discover and access. The purpose of EPA's Watersheds-based Monitoring and Interoperable Data Platforms project was to pilot the development of a centralized catalog for discovering sensor data. This could provide a single portal, where participants could register their sensors, and be included in data searches based on parameters collected or geographic attributes.

The project centered around three main components:

- Developing a metadata catalog and associated web services.
- Aiding identified partner agencies in making data from their deployed sensors available through web services.
- Building a web-based data discovery tool that illustrates the capabilities of the catalog and services.

The pilot sensor network was set up and tested for agencies in two selected watersheds: the Little Miami River watershed in Ohio and the Hackensack-Passaic watershed in New Jersey. Corresponding workgroups were set up in each watershed and provided insight and feedback throughout the catalog and web services design process. Each workgroup began by developing two use cases for evaluation at the end of the project, and partners were identified in each watershed to receive assistance in setup of a "data appliance" for serving up their data and registering with the catalog. In Ohio, the data partners were Clermont County and EPA Office of Research and Development (ORD). In New Jersey, the data partners were NJ Department of Environmental Protection (DEP) and the Meadowlands Environmental Research Institute (MERI).

In addition to these partners, EPA Region 7 made their KC Waters sensors available for registration in the metadata catalog and data discoverability through web services. These sensors were a nice addition to the pilot because the data appliance is provided by a commercially available product that already employs Open Geospatial Consortium (OGC) data sharing standards (Kisters). The ease with which they were able to participate in this system makes it likely that other agencies using the Kisters software solution will be able to share data with good result as well. Additionally, EPA Region 1, EPA Region 10, and USGS Instantaneous Values sites were also registered in the catalog.

This project was initiated as part of EPA's E-Enterprise initiative which is a transformative 21st century strategy to modernize how government agencies deliver environmental protection. The states, EPA, and tribes are collaboratively streamlining business processes across agencies and programs. These changes will improve environmental results and enhance services to the regulated community and the public by making government more efficient and effective. This project is a key component of the E-Enterprise strategy in that it is a step towards standardizing data collected using advanced monitoring techniques

which will improve the efficiency in which EPA, states, and tribes are able to conduct business and provide the public with a more transparent view of water quality and water quantity.

### **Metadata Catalog and Services**

The goal of the metadata catalog is to provide a centralized query-able repository that can be used to discover sensor data. Many of the early meetings with the watershed teams focused on identifying the types of data that were important for query and access. Many of the data elements the participants were interested in populating to the catalog were easy to derive directly from the data, such as parameters sensed and timeframe of the data observations. Others were more difficult to derive using existing standardized OGC Sensor Observation Service (SOS) attributes, such as the measurement interval for the data or the QC status. Feedback from these meetings was used to develop a conceptual model (Figure 1) for the metadata catalog, and also to build a metadata harvester that could be used to collect and update the information in the catalog. Early plans for the project were to extend the capabilities and web services of the existing 52° North (52N) platform as the foundation framework for the metadata catalog. However, while 52N is a powerful system for storing and providing service access to sensor monitoring data, it is not designed to support a system that is exclusively metadata. One of the most significant limitations, related to query functionality, is that 52N is not designed to store information for multiple organizations and the web services do not expose organization as a query-able parameter.

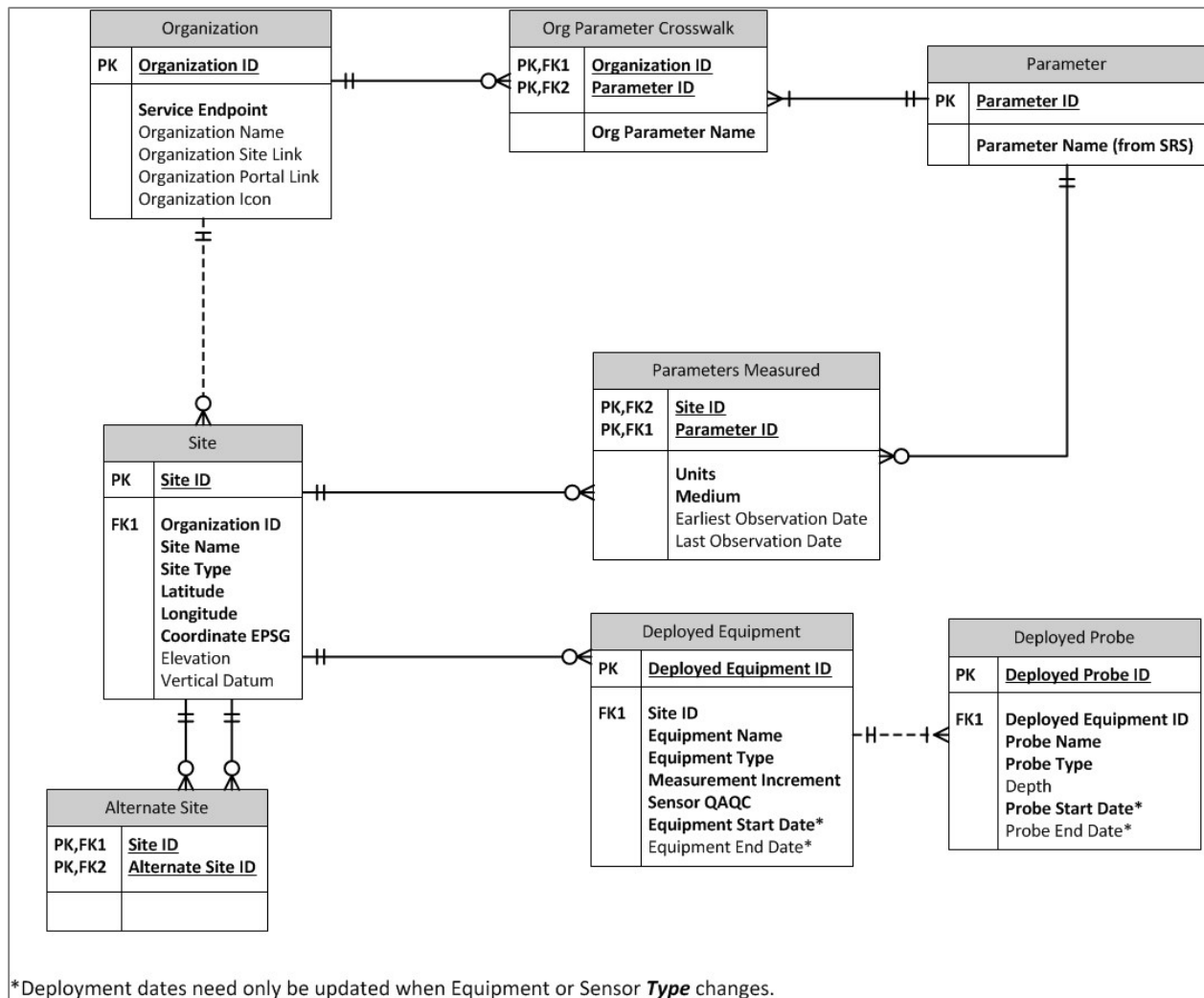


Figure 1. Metadata Catalog Conceptual Model

Early pilot implementations also suggested that better performance could be achieved parsing JSON results than XML results when retrieving all registered sensors for presentation on a web map. This led to the decision to develop a custom metadata catalog in PostgreSQL with accompanying REST services implemented using the Java Spring Framework. These web services provide GeoJSON responses for sensor/site based results, and JSON responses for attribute results. Appendix A contains the detailed web service API documentation.

Another important task for the metadata catalog was to reconcile the parameters reported. Maintaining consistent parameter names across different agencies and sensors was a key goal of this system as well as crucial to success. An important feature of the metadata catalog services is to allow the users to query for a single parameter value, like “Dissolved Oxygen”, and provide the intelligence to search across values reported as “DO” or “Oxygen, Dissolved”. Different organizations use their own identification schemes for parameters, but maintaining consistent names and format of the data across all data providers is necessary for data usability. To address this issue, parameters reported by the data providers are mapped to EPA’s Substance Registry Service (SRS). This allows for consistent parameter data query across organizations.

Another important data attribute that was discussed, was the QC status of the data. While these status values may vary by data provider, they can typically be characterized by:

Raw – the raw value collected by the sensor.

Provisional – data that has received some type of automated QC (such as range validation).

Final – data that has completed all QC checks as defined by the organizations standard operating procedures.

Based on the information from the participating data providers, it is clear that this information is not provided in a standard way. USGS provides this information via data qualifiers. The KC Waters provider separates draft and final results into different “procedures”. Due to the inconsistencies with the way this information is reported, it cannot be collected or queried in the metadata catalog in a meaningful way. Great care has been taken to ensure the information is communicated in the data results by the Currents Discovery Tool discussed later in this document.

#### Station and Sensor Metadata and QAQC

Discussions with the watershed teams revealed that there is a desire for very detailed access to sensor metadata. However, there is a corresponding constraint that having to enter and update detailed metadata for sensors would be a burden and possible deterrent for data provider’s participation in the system. Metadata is crucial to understanding and utilizing the sensor data, and without these data the sensor data can be rendered completely unusable. For the purposes of this pilot project, this issue was resolved by adding a “Sensor QAQC” field to the conceptual model. The intent of this field is for the data provider to store a link to detailed metadata that could possibly include station metadata, Quality Assurance Protection Plans, Standard Operating Procedures, sensor maintenance records, and data QAQC procedures. Ideally, this will allow providers to register this attribute one time, and then the information provided through the link will be updated as part of their normal operating procedures. At the conclusion of this pilot, none of the data participants have the data to use this link as intended. This method of documenting the metadata may need to be revisited if this system is pushed to production at a larger scale.

#### Successes

As of December, 2016 15,674 sites/sensors were registered in the metadata catalog. The distribution of sites by data provider is shown in Table 1.

Data Provider	Data Partner	Number of Sites
Clermont County	✓	4
EPA ORD	✓	3
EPA Region 1		2
EPA Region 10		1
KC Waters		18
MERI	✓	3
NJ DEP	✓	106
USGS		15,537

<b>Total</b>		<b>15,674</b>
--------------	--	---------------

Table 1. Number of sensor sites registered in the metadata catalog. Data partners are the organizations identified as part of this E-Enterprise project. Additional providers are workgroup participants that were also willing to share data.

The data harvester was developed to harvest information from SOS 2 compliant data appliances, as well as the USGS Instantaneous Values Web Service (<http://waterservices.usgs.gov/rest/IV-Service.html>).

The harvester runs every night at midnight and takes less than two minutes to run. The harvester updates the data available for the sites, along with any new parameters that may be available. The catalog also pings the data providers every five minutes and maintains a “lastPinged” date and “status” flag value that is returned when calling the getOrganizations service.

Thirteen REST services were developed for interacting with the metadata catalog. Most of these support sensor discovery via geography or other attributes. The others describe the data available from the catalog, with services such as getOrganizations which returns a list of data providers registered in the catalog, and availableParameters which returns all of the parameters that are mapped in the catalog. These web services support the Currents web-based discovery tool that was developed as part of this pilot, and are also being used to develop a Mobile App in another one of EPA’s E-Enterprise projects.

### Limitations

A current limitation of the metadata currently stored in the catalog is that USGS is not an SOS compliant data provider. While special functions are included in the data harvester to include USGS, USGS does not expose services equivalent to the SOS getDataAvailability or DescribeSensor services that allow the metadata catalog to derive the parameters reported and the time frame. Some customized efforts were made to populate this information by scraping existing USGS HTML services and other available data, but often the data availability for the USGS sites is not known, and as a result these sites cannot be included in metadata catalog query results that include dates.

The SOS services generally provide access to data based on three data categories:

- FeatureOfInterest – which is equivalent to the sites/sensors
- ObservedProperty – which is equivalent to the parameters measured
- Procedure – which can be interpreted and used multiple ways

For the purposes of the catalog, ObservedProperties were mapped to the EPA’s SRS parameters so that this information would be query-able. A similar effort for “Procedure” was considered, but it is not clear that providers organize their data into procedures in a consistent way, or that there is an obvious way to interpret what these procedures mean using a common standard in the metadata catalog. For example, the KC Waters provider has procedures for raw data, final data, data reported as daily means, as well as others. Discussions with Woods Hole Oceanographic Institute, a leading researcher into SOS implementation though not a participant in the current pilot, suggest that procedure can additionally be used to track equipment changes. OGC’s description is

*A procedure is defined as a measurement process, analysis, or processing algorithm that is used to obtain an observation result.*

There is a great deal of flexibility for how this may be interpreted, so it seems unlikely that procedure is something that can be mapped in the metadata catalog in a universal way.

One other requirement that was identified later in the project was that it is possible for a single data appliance to contain sensor information for multiple agencies or sub-groups. One use case for this is EPA's 52N cloud server that currently hosts Region 1 and Region 10. Similarly, NJ DEP would like to be able to distinguish between their fresh water and marine programs. For data providers that were set up as part of the pilot project, this information is made available in the DescribeSensor results as part of the "ContactList" data block. This level of detail has yet to be added to the metadata catalog and harvester.

### **Partners/Data Providers**

One of the key aspects of this project was to identify data providers that could be registered in the metadata catalog. Discussions with the workgroups revealed a great deal of interest in the ability to share data via this mechanism. The four data partners received direct contractor support for setting up data appliances, and all four of these partners successfully registered sensors and provide data that is accessible via the Currents Discovery Tool. These partners represented agencies at multiple geographic scales, with NJ DEP providing statewide data, Ohio's Clermont County representing a more local scale agency and MERI and EPA ORD representing very specific initiatives. The workgroup discussions, which included participants that were not data partners, resulted in participation from EPA Regions 1 and 10, as well as the EPA Region 7 KC Waters project. When the metadata catalog is made available to a broader audience, evidence suggests that agencies at multiple levels, large scale and small scale, public and private, will want to participate.

Supporting the four data partners meant helping them set up a data appliance that could receive SOS2 data requests and return data formatted as WaterML2. In lay terms a data appliance is basically a web server with publicly available web services. 52N was selected as the software for managing the sensor data because it is open source and can be installed on multiple platforms. One of the lessons learned was that the ability to support a data appliance is a high bar for some data providers due to limitations in accessing IT resources and the ability of the data provider to influence policies around network security and accessibility at an agency level. In the last month of this pilot, the details of making Clermont County's data publicly available are still being worked out. MERI is unable to host their own site at this time and their data will remain on a contractor website for a year. A new data partner interested in registering their sensors in our system was the Oregon Department of Environment Quality. They manage their water quality sensors with the Kisters software, which as previously described is relatively easy to ingest in our system, but they were unable to overcome Agency IT policy limitations that prevent the data services from being made public.

Another lesson learned through the workgroups was that some agencies may not be able to provide real time/raw data via their web services. This is the case with NJ DEP, which has policies that prevent download of both provisional and real-time data. As a result, only finalized data from NJ DEP is available via their data appliance. While not available for download, real-time continuous water quality data can be viewed through NJ DEP's continuous monitoring website.

Access to the Region 7 KC Waters sites that use the Kisters' software proved to be a real boon to the project because much was learned from the way they format their data, and Kisters also shared the SOS2 Hydrology Profile, which is considered a best practice (<http://www.opengeospatial.org/pressroom/pressreleases/2119>). This configuration setting on the data appliance can significantly improve performance by limiting the size of the data responses, and provides only the most recent sensor observation for the getObservations service by default. This configuration option was used for setting up the data partner appliances, and partners that do not follow this recommendation will not perform well in the Currents Discovery Tool.

### Successes

During the process of setting up the partner data appliances, an instruction set was developed for configuring a data appliance using Apache Tomcat and 52N (Appendix B). This will be a tremendous resource for sensor owners that want to configure a data appliance. In addition to the technical details of configuring the server, this includes guidelines for setting up Uniform Resource Name (URN) based identifiers, data templates, and python import scripts. While it is important that the metadata catalog and Currents Discovery Tool accept any SOS2 compliant provider data, these data guidelines provide an opportunity for data providers to take advantage of some value added functionality.

The URN scheme recommended by the guidelines currently provide two main advantages. The first is that the parameter URNs are standardized, and recognized by the metadata catalog. So data providers following this convention do not need to map their parameters as part of the sensor registration process. The other advantage is that by using a standard URN format, additional information can be embedded in the identifier. So in the case of "Procedure", the second to last element in the URN string indicates the QC status of the data. When this is recognized by the Currents Discovery Tool, it can be used to label the data.

The other benefit of using the templates and guidelines is that information of interest is stored in a standardized way. For example, it was discussed in the previous section, that standard SOS2 services do not really support the concept of "sub-organization". Following these data import instructions store this information in the "ContactList" data block. Other additional attributes of interest have also been managed in a standardized way as a result of the templates, such as Site Type (river, facility, etc.) and Medium (air, water, etc.). Both workgroups indicated that Site Type and Medium for the sites are data elements of interest, but the OGC standards do not dictate a standard way to provide this information.

### Limitations

Discussions with the workgroup revealed a great deal of interest in data quality and data qualifiers. The URN guidelines provide a way to flag data sets as raw or final, however 52N does not currently support the use of data qualifiers for individual data points. This will be available in a future version of 52N, at which time the import scripts should be updated to take advantage of this feature. The Region 7 KC Waters sites provided by the Kisters software provide data qualifier codes as URLs. Unfortunately, these URLs are not publicly accessible.

## **Web-based Discovery Tool**

At the onset of this project, the vision for the web-based discovery tool was very rudimentary and intended only as a simple proof on concept. An early lesson learned in the workgroup meetings was that having a visualization – no matter how simple – greatly fostered discussion with the workgroup teams. Abstract conversations about web service architecture made it difficult to engage water quality professionals, but they were able to supply a lot of great feedback on this functionality when described as a web tool they could interact with.

Each work group developed two real life use cases that the functionality of the pilot project could provide. These were formatted as user stories, and acceptance criteria for each use were identified by the workgroups. These use cases were used as rubrics and revisited often during system development to evaluate if these goals were being accomplished (See Appendix C for a complete list of acceptance criteria).

Each work group identified a use case that was largely focused on data discovery and data visualization.

<b>Hackensack-Passaic Watershed User Story # 1</b>
<b>Drinking Water/Source Water Protection Early Warning</b>
As a water manager, I want to view trends in selected parameters so that I can predict and remediate a water quality issue before it occurs.
<b>Little Miami Watershed User Story #1</b>
<b>Water Safety (Drinking Water and Recreation) optimization: Maximizing output while minimizing cost with a Harmful Algal Bloom focus</b>
As a water manager, I need to detect water quality issues, such as a harmful algal bloom, so that I can alert the public.

The Currents Discovery Tool does provide data visualization for the continuous monitoring data. Users have the ability to identify sites by the parameters measured, and can click on them to view the most recent data available (Figure 2). Early prototypes of Currents also had the ability to bookmark queries, so that it was easy to return to the same sites, or so that the results could be sent to other interested parties via an e-mail link. This is a feature that could easily be restored to Currents in the future.



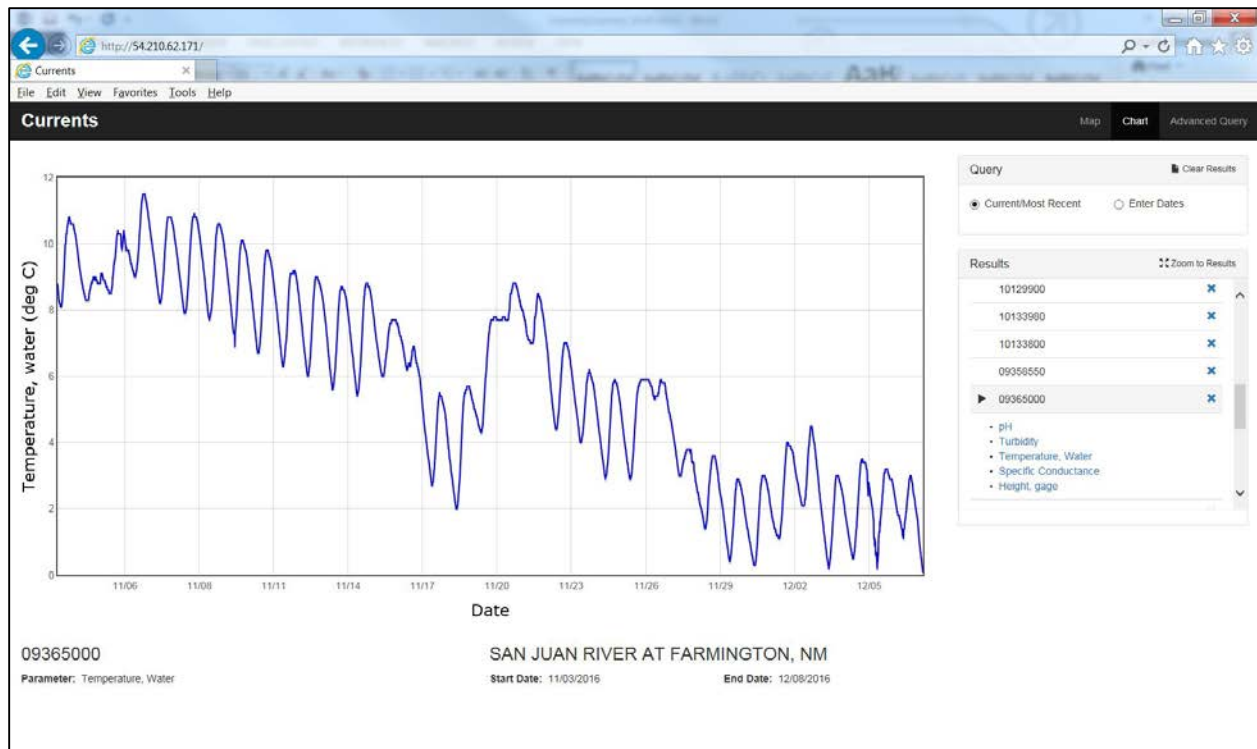


Figure 2. Real time Water Temperature data from a USGS Monitoring Data

The other user stories are related to Clean Water Act reporting and TMDL implementation. What these use cases have in common is a need to download large datasets for multiple sites. The volume of data needed to support these use cases is more than can reasonably be provided in real time via a web browser download. These kinds of requests need to be placed into a queue, processed on a server, and then the results can be provided to the user via e-mail as an attachment or data download link. The simple, open source architecture selected for the Currents tool does not support this model. This download requirement is something that should be addressed in a future version of the tool.

Hackensack-Passaic Watershed User Story # 2
<b>Water Quality Assessment (for Clean Water Act Integrated Reporting)</b>
As a water quality manager, I want to download continuous monitoring parameter data so that I can compare it to numeric criteria and evaluate if the water is meeting state standards.
Little Miami Watershed User Story #2
<b>TMDL Implementation</b>
As a water quality manager, I need to be able to download parameter data for use in running TMDL models.

## Successes

The Currents tool is able to display the more than 15,000 sites registered in the metadata catalog on an interactive map, and users can query them by parameters sensed, as well as geographical attributes such as HUC watershed and County boundaries or a user defined polygon. Users can view real time data on a time series chart display with the ability to select different date ranges and toggle between multiple parameters. Users can download the observation data for a single parameter in a comma delimited text format for all SOS compliant providers for the period of record. The Currents tool is also designed so that much of this functionality is also available on mobile devices. There is a complete list of Currents features and potential future enhancements in Appendix D.

## Limitations

The two main limiting factors for the functionality of the Currents tool are development time and architecture. There are several more features that could be added to the tool using the existing catalog services which simply did not fit within the scope of the pilot. For example, the web services provide the ability to select sites using a specified point and buffer, as well as using upstream/downstream navigation, but these features are not yet accessible via the tool.

It was clear early on in the discussions with the workgroups that there is a desire to be able to download large datasets over multiple sites. This has been implemented in many other data discovery systems, such as the Water Quality Portal and the access points for the National Estuarine Research Reserve data. Future iterations of the Currents tool will need to expand the software architecture to add the ability to queue and deliver these large data jobs.

The existing Currents tool has a simple data access model, where it requests data directly from the provider, and then manipulates the results on the client side. A limitation of the existing SOS services is that there is not an easy way to evaluate the data response that is returned for a request. Some providers limit the amount of data that is returned to a single service call. For example, USGS limits operational data that is not quality assured (which includes temperature and precipitation) to 120 days or less. There is not currently a way to communicate these provider data limitations via the metadata catalog and services. There is also the possibility that more data can be requested than the browser can process. An issue here is that there are no services in place that allow users to query how much data will be returned, prior to making the request. So users can make a request that will take a very long time to process, or hang their web browser, but there are currently no mechanisms for the application to anticipate this situation and prevent it.

## **Taking this Pilot to Production**

For the purposes of the pilot, an open source architecture was selected, and development resources favored functionality over detailed error handling and reporting. The following are some areas that should be addressed prior to moving this system to production.

## Scalability

The current metadata harvester takes less than two minutes to run against the 7 registered SOS providers and the USGS instantaneous values service. USGS does not provide SOS compliant services for the data harvest. However, the services they do provide allow for the harvester to divide up the data

requests and run concurrent harvests to improve performance. The SOS services do not provide an obvious mechanism for dividing up the requests, so the metadata are processed in a serial fashion. This is not currently an issue since NJ DEP as the largest registered SOS provider only has 106 sites, but if USGS transitions to SOS, or a large scale SOS provider is identified, the SOS section of the metadata harvester should be re-evaluated and possibly refactored.

The Currents Discovery Tool has been developed and tested against three general types of data providers: USGS, Kisters Commercial Software (KC Waters) and the data partners that have been configured using the guidelines and templates developed for this project. Even among this small sample size of providers, it is clear that the standards can have slight variations in how they are implemented. Figure 3 shows the DefaultPointMetadata element returned from the WaterML2 response from the USGS instantaneous values service, and the Clermont County getObservation service. The difference in the way the units is reported is small, but important, because the application needs to access this value for display on the charts and also to report it in the data download.

```
<!--USGS WaterML 2-->
<wml2:defaultPointMetadata>
  <wml2:DefaultTVPMeasurementMetadata>
    <wml2:qualifier>
      <swe:Category definition="http://waterdata.usgs.gov/nwisweb/rdf?agingCd=P">
        <swe:value>P</swe:value>
      </swe:Category>
    </wml2:qualifier>
    <wml2:uom xlink:title="ft3/s"/>
    <wml2:interpolationType
      xlink:href="http://www.opengis.net/def/waterml/2.0/interpolationType/Continuous" xlink:title="Continuous"/>
    </wml2:DefaultTVPMeasurementMetadata>
  </wml2:defaultPointMetadata>

<!--Clermont County WaterML 2-->
<wml2:defaultPointMetadata>
  <wml2:DefaultTVPMeasurementMetadata>
    <wml2:uom code="ft"/>
    <wml2:interpolationType
      xlink:href="http://www.opengis.net/def/timeseriesType/WaterML/2.0/continuous" xlink:title="Instantaneous"/>
    </wml2:DefaultTVPMeasurementMetadata>
  </wml2:defaultPointMetadata>
```

Figure 3. WaterML 2 Default Point Metadata from USGS and an SOS project data partner.

There are also some slight differences between the way data is reported between the 52N data partners and the Kisters Commercial software. For example, in the getObservation response, the Kisters response reports procedure as a single element. The 52N data partners report procedure as a data block (Figure 4).

```

<!--Kisters Response-->
<om:procedure xlink:href="http://kc-geos-wq01.kc.umkc.edu:8080/tstypes/Cmd.P" xlink:title="02.FINAL"/>

<!-- Clermont County -->
<om:procedure>
  <wml2:ObservationProcess gml:id="process.6F8DAA65C1D4E31D93A0A364D7C3A99CFCD6D44D">
    <gml:identifier codeSpace="http://www.opengis.net/def/nil/OGC/0/unknown">urn:x-
    epaiwpp:sensor:oh-cty-cle:wrld:steflwr:raw:rainfall</gml:identifier>
    <wml2:processType
    xlink:href="http://www.opengis.net/def/waterml/2.0/processType/Algorithm"/>
    <wml2:parameter>
      <om:NamedValue>
        <om:name xlink:href="urn:ogc:def:identifier:OGC:uniqueID"
        xlink:title="uniqueID"/>
        <om:value xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xsi:type="xs:string">urn:x-epaiwpp:sensor:oh-cty-cle:wrld:steflwr:raw:rainfall</om:value>
      </om:NamedValue>
    </wml2:parameter>
    <wml2:parameter>
      <om:NamedValue>
        <om:name xlink:href="urn:ogc:def:identifier:OGC:1.0:shortname"
        xlink:title="shortName"/>
        <om:value xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xsi:type="xs:string">urn:x-epaiwpp:sensor:oh-cty-cle:wrld:steflwr:raw:rainfall</om:value>
      </om:NamedValue>
    </wml2:parameter>
    <wml2:parameter>
      <om:NamedValue>
        <om:name xlink:href="urn:ogc:def:identifier:OGC:1.0:longname"
        xlink:title="longName"/>
        <om:value xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xsi:type="xs:string">urn:x-epaiwpp:sensor:oh-cty-cle:wrld:steflwr:raw:rainfall</om:value>
      </om:NamedValue>
    </wml2:parameter>
  </wml2:ObservationProcess>
</om:procedure>

```

Figure 4. WaterML 2 Procedure Values from Kisters Commercial Software and Clermont County 52N. Highlighted values illustrate the source of the procedure information provided in the data downloads from the Currents Discovery Tool.

New partners registered in the metadata catalog that use additional variations on how this information is reported, may not display correctly or have missing information in the Currents Discovery Tool charts and downloads. Prior to releasing this site to production, it should be evaluated against a wider variety of data providers.

As the number of sites registered in the catalog increases, there may also be some design considerations in the Currents Discovery Tool that should be re-evaluated. The pilot application uses marker clustering to display all 15,000 sites on the map. The marker clustering plugin documentation suggests that it can support up to 50,000 points. If the number of sites grows very large, it may not be feasible to display all sites on the map at once. The Currents tool should also be evaluated for dealing with large query result datasets to make sure the users browser does not become bogged down.

## Implementation

To prepare the overall system for production deployment, there are several implementation upgrades that should be made to enhance stability and the user experience. For the metadata catalog database, this would include updating the architecture to provide system backup, replication, failover and load balancing features. For the harvester, in addition to the scalability issues previously discussed, this could include developing a REST API for interacting with the system rather than relying Tomcat cron jobs. This API could include the ability to stop and restart the system, provide performance metrics, and even allow an administrator to approve/disapprove registered organizations. A component should also be developed that allows data providers to register their sensors in the catalog. This is currently managed via a manual process.

The data discovery REST service API should be updated to provide more detailed error handling results that communicate to the user why a request is invalid. Currently, all services respond to a failed request with a default error. The catalog functionality that is used to ping the data providers to determine their status should also be upgraded. Currently, whether or not a provider is up is derived solely on whether or not the server responds. This means that a data provider that responds with an error is still considered “up” in the current implementation.

The pilot Currents Discovery Tool is entirely client based, using freely available open source technologies. Feedback from the pilot suggests that there is a lot of interest in enhancing the charting functionality to allow more interactive features, along with the ability to view multiple parameters. There may be an advantage to considering a fee based charting package such as Highcharts or amCharts to enhance the charting capability. There is also a lot of interest in adding additional descriptive capability, to display information about the provider organizations, their SOPs for data collection, as well as site level metadata. The Currents Discovery Tool would benefit from the addition of a server side language to provide application state. This would allow for these different features to be handled independently on dedicated web pages which should improve performance. The current architecture manages all of the data concurrently and presents the data to users by showing and hiding different panels, which is likely to become limiting as the feature list continues to grow.

## Architecture

It is clear based on the requirements developed by the workgroups for the use cases, as well as their feedback on the outcome of this project, that large scale data download is an absolute requirement for success. Adding this functionality will require the ability in the Currents Discovery Tool for users to define a multi-site download request and then submit that request to a data harvester. This data harvester will need to compile the SOS service calls needed to request the data from each of the data providers, loop over these calls to make the requests, compile the data from the responses, and provide a delivery mechanism for the data back to the user. The system will additionally need to provide the user with a log detailing the success of these calls, so that users can be alerted if one of the data providers was not available at the time of the data request, or if the provider did not provide the extent of data requested.

## OGC Service Wish List

The workgroup discussions conducted as part of this pilot revealed some requirements that are not easily resolved with the existing OGC SOS2 services. One issue is that there is a need to store data from multiple agencies, or sub-agencies, on a single data appliance. NJ DEP would like to differentiate between sites that belong to their coastal and freshwater programs. At the conclusion of this pilot, EPA's cloud based data appliance hosts data from EPA Regions 1 and 10, and the EPA ORD data. Even though these data are managed on a single appliance, there is a desire to query these data separately. A useful addition to the standards would be to provide a way to identify data by organization, with a corresponding query parameter added to the services that could be used to filter the data results by organization.

Another capability that would be useful is the ability to request information about a data result prior to actually requesting the result. The ability to obtain information about the number of results returned by a request would be useful managing the data in applications, and this type of service could also indicate if the data response would be limited by provider imposed server constraints, such as a restriction that would only return data for a single year, even when multiple years are requested.

The most valuable enhancements that could be made, based on workgroup user feedback, is improved standards and access to metadata information. One of the lessons learned discussed in this document is that QC information is not stored in a standard way among providers, and as a result cannot be harvested into the metadata catalog and made query-able. Even though this was discussed at length with the workgroups, it was still identified in their final feedback as a major shortcoming of the system. There is a strong desire by the user community to be able to filter the data by QC status. In order for this to be possible, there needs to be standards for how this information is stored and can be accessed.

### Setting Up Data Partners with 52N

The 52 North implementation of OGC SOS was used to build distributed data appliances for the pilot watershed partners and it met the project needs well. The following observations were made during implementation.

#### General Comment:

- The DeleteObservation request added by 52 North as an extension to the SOS standard is of high value. Data partners sometimes identified errors in their data after posting, and DeleteObservation supports the replacement of erroneous data.

#### Possible Bug:

- The DeleteObservation request threw a NullPointerException for an example test case that was created with a single observation in the associated observation series, so there may be an edge case not fully addressed in the implementation.

#### Feature Requests:

- Use of InsertResult and DeleteObservation are not currently compatible in 52N SOS, as the observation.identifier field is not set in an InsertResult operation. It is simple to work around this by assigning values directly in the database, but it would be useful to have the identifier assigned automatically by InsertResult requests.

- <gda:observedproperty xlink:title> and <om:observedproperty xlink:title> do not return the expected same value from equivalent GetDataAvailability and GetObservation requests in 52N SOS 4.3.6 or 4.3.7; this is corrected in the current development branch (unreleased as of 12/28/2016) (*Addressed*)
- The field observedproperty.name must be set manually within the database in 52N SOS 4.3.6 or 4.3.7 instead of being read from the InsertSensor <sml:output name> field. This is already planned for correction in 52N SOS version 4.4 (no release date set) (*Addressed*)
- Applying the OGC Sensor Observation Service 2.0 Hydrology Profile requires sever access to manually edit the profile files. This is already planned for correction in 52N SOS version 4.4 (no release date set). (*Addressed*).
- Observation-specific data qualifiers would be useful for the IWN, but data qualifiers as defined in the WaterML2 standard (e.g. <wml2:qualifier xlink>) are not yet supported in the 52N SOS database model. Observation-specific qualifiers can be included with InsertObservation requests using the <om:parameter> tag in the current development branch, but cannot be entered in InsertResult requests. Implementation of wml2 qualifiers and/or om parameters for InsertResult could be useful.
- Downloading of results from large GetObservation requests can be time-consuming, and it would be useful to provide the user with feedback on the progress of their request. One way 52N SOS might help is to allow a temporal filter to be placed on the GetDataAvailabilityRequest to allow the querying individual/software to assess roughly how large a given retrieval might be to set expectations and perhaps strategy, such as breaking down the retrieval into smaller subretrievals.

The 52N SOS development team and community have provided consistently supportive and positive experiences with questions and requests.

## Conclusions and Future Directions

The most significant lesson learned from this pilot is that this type of centralized portal would not be possible if it wasn't for the OGC Standards: SOS2 and WaterML 2. The features of these services and consistently formatted results allow the metadata catalog to be populated via the harvester and data presented in a common format across all providers via the Currents Discovery Tool. The current system focuses on SOS 2 and WaterML 2, but future iterations could be expanded to include SOS 1 and WaterML 1. This may allow additional data providers to participate in this initiative.

The conceptual design for the catalog included metadata elements related to deployed equipment and deployed probes, along with the Sensor QAQC field that was meant to provide the URL link to additional information. None of the data partners in the pilot project currently provide this information. It would be helpful to identify a partner that has this level of information, or to work with a partner to develop this level of equipment detail, so that these features can be added to the metadata catalog and tested to see if this provides the level of detail data consumers are looking for in order to use the data with confidence.

Some of the partners in the project have noticed anomalies in their data that they did not see prior to visualizing it via the services, for example when data points in their time series suddenly drop to 0. This has encouraged these partners to look at their data more closely, and illustrates the need to include

data qualifiers or nil indicators as part of the dataset. When 52N supports data qualifiers, adding data qualifiers to the import scripts and templates will be a valuable next step.

The intent of the Currents Discovery Tool was to provide basic proof of concept features for the catalog. This was accomplished, and future endeavors could choose to continue to develop this tool, or use the catalog services to develop customized tools. Conversations with the workgroups revealed a large suite of potential use cases that were beyond the scope of this project. Some potential applications could be to build customized dashboards to support specific use cases. There was a lot of interest in evaluating data for specific purposes – early detection of harmful algal blooms and drinking water/source water protection. EPA is current using the catalog services to build a mobile app that will allow users to identify parameters at specific sites, and receive notifications if the parameters exceed certain values.

The key to long term success for this project will be the ability for agencies with sensor data to participate. For some agencies, this will be as simple as registering with the metadata catalog when SOS compliant services are available from their commercial software. For others, EPA may need to provide access to a data appliance that can host their data. Currently the tools are only interacting with three categories of data providers; USGS, data hosted by Kisters, and data appliances configured using the standard methodology developed for this project. Expanding this sample size, and interacting with additional partners that are also providing SOS 2 compliant services will go a long way to furthering the lessons learned.



## Appendix A. Service API Documentation for the Metadata Catalog Services.

### Service Index

GetOrganizations .....	1
AvailableParameters .....	2
GetSensors .....	3
GetSensorParameters .....	6
GetOrganizationParameters .....	7
GetStates .....	8
GetCounties .....	9
Interacting with the SOS 2.0 Services .....	10

### **GetOrganizations**

This service can be used to retrieve the list of organizations that are currently registered as data providers in the metadata catalog. The service will accept an optional organization id (org\_id) parameter and limit the results to the requested organization. The attributes for each organization include the service end point, the date of the most recent data harvest, when the server was last pinged, and an indication of whether the endpoint is available.

### **Url Signature**

<http://54.210.62.171/catalog/api/getOrganizations/>

### **Parameters**

Name	Value	Required
org_id	Unique id of an organization. (Corresponds to the organizationID value returned by this service.)	Optional

### **Result Key Values**

Key	Value
organizationID	Identifier used by the catalog for the organization.
serviceEndpoint	Service endpoint for the data provider.
organizationName	Name of the organization providing the information
organizationSiteLink	A link to the organizations website if provided.

organizationPortalLink	A link to the organizations data portal website if provided.
qaqclink	A link to the organizations sensor metadata, such as sensor maintenance and standard operating procedures, if provided.
serviceType	"SOS2" indicates that this is a provider that accepts standard SOS2 requests. "NON" indicates that this is not an SOS2 compliant provider.*
mostRecentHarvest	Date of most recent harvest into the metadata catalog,
lastPinged	Date that the organization was last pinged by our service.
defaultProcedure	Default procedure shown in the Currents Discovery Tool GUI.**
status	Indicator whether the services are active ("up" or "down").
* USGS is currently the only provider that is not compliant with SOS2	
** This is a temporary element to support the Discovery Tool that will likely be removed.	

## Example Query

[http://54.210.62.171/catalog/api/getOrganizations/?org\\_id=USGS](http://54.210.62.171/catalog/api/getOrganizations/?org_id=USGS)

## Example Results

```
[{"organizationID":"KCWaters","serviceEndpoint":"http://kc-geos-wq01.kc.umkc.edu:8080/KiWIS/KiWIS?service=SOS&version=2.0","organizationName":"KC Waters","organizationSiteLink":null,"organizationPortalLink":null,"qaqclink":null,"serviceType":"SOS2","mostRecentHarvest":"2016-09-21","lastPinged":"2016-09-22","defaultProcedure":"http://kc-geos-wq01.kc.umkc.edu:8080/tstypes/Cmd.P","status":"up"}]
```

## AvailableParameters

This service returns the list of parameters that are available for query via the metadata catalog.

## Url Signature

<http://54.210.62.171/catalog/api/availableParameters/>

## Parameters

None

## Result Key Values

Key	Value
parameterName	Descriptive name of the parameter.
id	Unique ID for the parameter in the metadata catalog.

## Example Query

<http://54.210.62.171/catalog/api/availableParameters/>

## Example Results

```
[{"parameterName": "Air Temperature", "id": 79246}, {"parameterName": "Ammonia-Nitrogen (NH3)", "id": 38644}, {"parameterName": "Ammonium-Nitrogen (NH4+)", "id": 59836}, {"parameterName": "Barometric Pressure", "id": 79443}, {"parameterName": "BOD", "id": 79331}, {"parameterName": "Chloride", "id": 59791}, {"parameterName": "COD", "id": 79521}, {"parameterName": "Conductivity", "id": 79261}, {"parameterName": "Depth", "id": 79607}, {"parameterName": "Depth to Water Level", "id": 79605}, {"parameterName": "Dewpoint Temperature", "id": 78485}, {"parameterName": "Discharge", "id": 79588}, {"parameterName": "DO", "id": 101627}, {"parameterName": "Elevation above NGVD", "id": 79616}, {"parameterName": "Flow", "id": 1643212}, {"parameterName": "Gage Height", "id": 98309}, {"parameterName": "Nitrate-Nitrogen (NO3-)", "id": 97628}, {"parameterName": "ORP", "id": 79933}, {"parameterName": "pH", "id": 78426}, {"parameterName": "Precipitation", "id": 79912}, {"parameterName": "Relative Humidity", "id": 79436}, {"parameterName": "Rhodamine WT", "id": 98304}, {"parameterName": "Salinity", "id": 79345}, {"parameterName": "Specific Conductivity", "id": 79373}, {"parameterName": "TDS", "id": 79555}, {"parameterName": "Turbidity", "id": 79196}, {"parameterName": "Velocity", "id": 79287}, {"parameterName": "Water Temperature", "id": 79281}, {"parameterName": "Wind Direction", "id": 79648}, {"parameterName": "Wind Gust", "id": 78471}, {"parameterName": "Wind Speed", "id": 79645}]
```

## GetSensors

These services return a feature collection which specifies the sitelid, siteName, orgId and geometry (type and coordinates) of a sensor. There is a different “getSensor” service for each spatial geometry input. All of these services can accept an organization id (can derive from the getOrganizations service), parameter id (from availableParameters), as well as a minimum and maximum observation date.

- “getSensors/” will return all sensors registered in the catalog if no input parameters are specified.
- “getSensorsByCounty/” returns all sensors that are filtered by a specific user defined county value.
- “getSensorsByHuc/” returns all sensors within a single 8 digit HUC.
- “getSensorsByBuffer/” returns all sensors within a buffer generated from a user specified point and radius.
- “getSensorsByBBBox/” returns all sensors within a bounding box.

- “getSensorsUpstream/” returns all sensors upstream from a specified point based on the NHDPlus hydrography. Results are limited to the HUC 8 watershed boundary.
- “getSensorsDownstream/” returns all sensors downstream from a specified point based on the NHDPlus hydrography. Results for the downstream service are not limited.

## Url Signatures

<http://54.210.62.171/catalog/api/getSensors/>

<http://54.210.62.171/catalog/api/getSensorsByCounty/>

<http://54.210.62.171/catalog/api/getSensorsByHuc/>

<http://54.210.62.171/catalog/api/getSensorsByBuffer/>

<http://54.210.62.171/catalog/api/getSensorsByBBox/>

<http://54.210.62.171/catalog/api/getSensorsUpstream/>

<http://54.210.62.171/catalog/api/getSensorsDownstream/>

## Parameters

Name	Value	Required
<b><i>Input Parameters accepted by all getSensor services.</i></b>		
org_id	Catalog unique ID for an organization	Optional
param	Catalog unique for a sensor parameter.	Optional
min_obs_date	When specified, only sensors with observations after this date will be returned.	Optional
max_obs_date	When specified, only sensors with observations before this date will be returned.	Optional
<b><i>Example</i></b>		
<a href="http://54.210.62.171/catalog/api/getSensors/?org_id=KCWaters">http://54.210.62.171/catalog/api/getSensors/?org_id=KCWaters</a>		
<b><i>getSensorsByCounty</i></b>		
countyfips	The 5 digit combined state and county FIPS value.	Required
<b><i>Example</i></b>		
<a href="http://54.210.62.171/catalog/api/getSensorsByCounty/?countyfips=37001">http://54.210.62.171/catalog/api/getSensorsByCounty/?countyfips=37001</a>		
<b><i>getSensorsByHuc</i></b>		
huc8	An 8 digit HUC value.	Required
<b><i>Example</i></b>		
<a href="http://54.210.62.171/catalog/api/getSensorsByHuc/?huc8=11040001">http://54.210.62.171/catalog/api/getSensorsByHuc/?huc8=11040001</a>		

<b>getSensorsByBuffer</b>		
point	Coordinate value of point to buffer.	Required
buffer	Length in meters of the radius used to generate the buffer.	Required
<b>Example</b>		
<a href="http://54.210.62.171/catalog/api/getSensorsByBuffer/?point={" type":"point","coordinates":[-79.0,35.0]}&amp;buffer='50000"'>http://54.210.62.171/catalog/api/getSensorsByBuffer/?point={"type":"Point","coordinates":[-79.0,35.0]}&amp;buffer=50000</a>		
<b>getSensorsByBBox</b>		
bbox	Bounding box coordinates for a polygon used to select the sensors.	Required
<b>Example</b>		
<a href="http://54.210.62.171/catalog/api/getSensorsByBBox/?bbox={" type":"polygon","coordinates":[[[-95.5810546875,39.6733703917656],[-95.51513671875,37.84015683604134],[-93.09814453125,37.87485339352928],[-93.14208984375,39.7240885773337],[-93.14208984375,39.7240885773337],[-95.5810546875,39.6733703917656]]]}"="">http://54.210.62.171/catalog/api/getSensorsByBBox/?bbox={"type":"Polygon","coordinates":[[[-95.5810546875,39.6733703917656],[-95.51513671875,37.84015683604134],[-93.09814453125,37.87485339352928],[-93.14208984375,39.7240885773337],[-93.14208984375,39.7240885773337],[-95.5810546875,39.6733703917656]]]}</a>		
<b>getSensorsUpstream</b>		
point	GeoJSON representing the input point.	Required
<b>Example</b>		
<a href="http://54.210.62.171/catalog/api/getSensorsUpstream/?point={" type":"point","coordinates":[-78.355,35.508]}"="">http://54.210.62.171/catalog/api/getSensorsUpstream/?point={"type":"Point","coordinates":[-78.355,35.508]}</a>		
<b>getSensorsDownstream</b>		
Point	GeoJSON representing the input point.	Required
<b>Example</b>		
<a href="http://54.210.62.171/catalog/api/getSensorsDownstream/?point={" type":"point","coordinates":[-78.355,35.508]}"="">http://54.210.62.171/catalog/api/getSensorsDownstream/?point={"type":"Point","coordinates":[-78.355,35.508]}</a>		

## Result Key Values

Key	Value
siteld	Unique ID for the site used by the data provider.
siteName	Descriptive name for the site.
id	Unique ID for the site used by the metadata catalog.
orgId	Unique ID for the organization used by the metadata catalog.
type	Type of geometry for the site (currently always "Point").
coordinates	Geographic coordinates of the site.

## Example Results

```
{
  "type": "FeatureCollection",
  "features": [{
    "properties": {
      "siteId": "02096500",
      "siteName": "HAW RIVER AT HAW RIVER, NC",
      "id": "e6475444-de18-434d-ad15-3461b841a495",
      "orgId": "USGS"
    },
    "type": "Feature",
    "geometry": {
      "type": "Point",
      "coordinates": [-79.3661111,
        36.0872222]
    }
  }
]}
```

## GetSensorParameters

This service returns the list of parameters that are registered in the catalog for the input sensor ID. The results include the organization parameter ID's that can be used to query data by parameter from the service endpoint.

## Url Signature

<http://54.210.62.171/catalog/api/getSensorParameters/>

## Parameters

Name	Value	Required
uuid	Unique value of the sensor from the metadata catalog	Required

## Result Key Values

Key	Value
parameterOrgId	Unique ID used by the provider for a parameter.
minObservationDate	The earliest observation date for the input site for the parameter.
maxObservationDate	The most recent observation date for the input site for the parameter.
parameterName	The descriptive name used by the metadata catalog for the parameter.

## Example Query

<http://54.210.62.171/catalog/api/getSensorParameters/?uuid=1cba34b9-55f6-4821-bca3-481f5955366b>

## Example Results

```
[{"parameterOrgId":"http://kc-geos-wq01.kc.umkc.edu:8080/parameters/Turbidity","minObservationDate":"2010-05-26","maxObservationDate":"2011-09-13","parameterName":"Turbidity","id":79196},{ "parameterOrgId":"http://kc-geos-wq01.kc.umkc.edu:8080/parameters/EC","minObservationDate":"1996-01-01","maxObservationDate":"2013-05-31","parameterName":"Conductivity","id":79261},{ "parameterOrgId":"http://kc-geos-wq01.kc.umkc.edu:8080/parameters/WT","minObservationDate":"1996-12-31","maxObservationDate":"2013-05-31","parameterName":"Water Temperature","id":79281},{ "parameterOrgId":"http://kc-geos-wq01.kc.umkc.edu:8080/parameters/Depth","minObservationDate":"1995-01-01","maxObservationDate":"2013-05-31","parameterName":"Depth","id":79607}]
```

## GetOrganizationParameters

This service returns the list of parameters that are registered in the catalog for the input organization ID. The results include the organization parameter ID's that can be used to query data by parameter from the service endpoint.

## Url Signature

<http://54.210.62.171/catalog/api/getOrganizationParameters/>

## Parameters

Name	Value	Required
org_id	The unique id for the organization from the metadata catalog.	Required

## Result Key Values

Key	Value
parameterOrgId	Unique ID used by the provider for a parameter.
minObservationDate	The earliest observation date for the input site for the parameter.
maxObservationDate	The most recent observation date for the input site for the parameter.
parameterName	The descriptive name used by the metadata catalog for the parameter.
id	Unique id for the parameter used by the metadata catalog.

## Example Query

[http://54.210.62.171/catalog/api/getOrganizationParameters/?org\\_id=KCWaters](http://54.210.62.171/catalog/api/getOrganizationParameters/?org_id=KCWaters)

## Example Results

```
[{"maxObservationDate":"2016-08-18","parameterOrgId":"http://kc-geos-wq01.kc.umkc.edu:8080/parameters/WT","minObservationDate":"1995-12-31","parameterName":"Water Temperature","id":79281}, {"maxObservationDate":"2016-08-18","parameterOrgId":"http://kc-geos-wq01.kc.umkc.edu:8080/parameters/EC","minObservationDate":"1995-01-01","parameterName":"Conductivity","id":79261}, {"maxObservationDate":"2016-08-18","parameterOrgId":"http://kc-geos-wq01.kc.umkc.edu:8080/parameters/Depth","minObservationDate":"1994-01-01","parameterName":"Depth","id":79607}, {"maxObservationDate":"2016-08-18","parameterOrgId":"http://kc-geos-wq01.kc.umkc.edu:8080/parameters/Turbidity","minObservationDate":"1994-01-01","parameterName":"Turbidity","id":79196}, {"maxObservationDate":"2016-08-19","parameterOrgId":"http://kc-geos-wq01.kc.umkc.edu:8080/parameters/Precip","minObservationDate":"2011-06-10","parameterName":"Precipitation","id":79912}]
```

## GetStates

This service returns a list of states. This service was created to support the discovery tool so that it can provide a county drop down lists controlled by a selected state.

## Url Signature

<http://54.210.62.171/catalog/api/getStates/>

## Parameters

None.

## Result Key Values

Key	Value
name	Name of the state
id	State FIPS code

## Example Query

<http://54.210.62.171/catalog/api/getStates/>



## Example Results

```
[{"name": "Alabama", "id": "01"}, {"name": "Alaska", "id": "02"}, {"name": "American Samoa", "id": "60"}, {"name": "Arizona", "id": "04"}, {"name": "Arkansas", "id": "05"}, {"name": "California", "id": "06"}, {"name": "Colorado", "id": "08"}, {"name": "Commonwealth of the Northern Mariana Islands", "id": "69"}, {"name": "Connecticut", "id": "09"}, {"name": "Delaware", "id": "10"}, {"name": "District of Columbia", "id": "11"}, {"name": "Florida", "id": "12"}, {"name": "Georgia", "id": "13"}, {"name": "Guam", "id": "66"}, {"name": "Hawaii", "id": "15"}, {"name": "Idaho", "id": "16"}, {"name": "Illinois", "id": "17"}, {"name": "Indiana", "id": "18"}, {"name": "Iowa", "id": "19"}, {"name": "Kansas", "id": "20"}, {"name": "Kentucky", "id": "21"}, {"name": "Louisiana", "id": "22"}, {"name": "Maine", "id": "23"}, {"name": "Maryland", "id": "24"}, {"name": "Massachusetts", "id": "25"}, {"name": "Michigan", "id": "26"}, {"name": "Minnesota", "id": "27"}, {"name": "Mississippi", "id": "28"}, {"name": "Missouri", "id": "29"}, {"name": "Montana", "id": "30"}, {"name": "Nebraska", "id": "31"}, {"name": "Nevada", "id": "32"}, {"name": "New Hampshire", "id": "33"}, {"name": "New Jersey", "id": "34"}, {"name": "New Mexico", "id": "35"}, {"name": "New York", "id": "36"}, {"name": "North Carolina", "id": "37"}, {"name": "North Dakota", "id": "38"}, {"name": "Ohio", "id": "39"}, {"name": "Oklahoma", "id": "40"}, {"name": "Oregon", "id": "41"}, {"name": "Pennsylvania", "id": "42"}, {"name": "Puerto Rico", "id": "72"}, {"name": "Rhode Island", "id": "44"}, {"name": "South Carolina", "id": "45"}, {"name": "South Dakota", "id": "46"}, {"name": "Tennessee", "id": "47"}, {"name": "Texas", "id": "48"}, {"name": "United States Virgin Islands", "id": "78"}, {"name": "Utah", "id": "49"}, {"name": "Vermont", "id": "50"}, {"name": "Virginia", "id": "51"}, {"name": "Washington", "id": "53"}, {"name": "West Virginia", "id": "54"}, {"name": "Wisconsin", "id": "55"}, {"name": "Wyoming", "id": "56"}]
```

## GetCounties

This method accepts a state FIPS code as input and returned a list of counties in that state, sorted alphabetically.

## Url Signature

<http://54.210.62.171/catalog/api/getCounties/>

## Parameters

Name	Value	Required
state_id	Unique state FIPS identifier.	Required

## Result Key Values

Key	Value
name	County Name
id	Five digit combined state and county FIPS code.

## Example Query

[http://54.210.62.171/catalog/api/getCounties/?state\\_id=15](http://54.210.62.171/catalog/api/getCounties/?state_id=15)

## Example Result

```
[{"name": "Hawaii", "id": "15001"}, {"name": "Honolulu", "id": "15003"}, {"name": "Kalaupapa", "id": "15005"}, {"name": "Kauai", "id": "15007"}, {"name": "Maui", "id": "15009"}]
```

## Interacting with the SOS 2.0 Services

The following examples describe how to use the information derived from the metadata catalog to interact with the SOS 2.0 services available through the data appliances. These examples are limited to the services and query parameters used by the Currents Discovery Tool. More detailed information about available SOS 2.0 services and query parameters is provided in this 52° North Resource: <https://wiki.52north.org/SensorWeb/SosOperations>. More complete information about OGC's SOS 2.0 Hydrology Profiles best practices is available from <http://docs.openeospatial.org/bp/14-004r1/14-004r1.html>. All of these examples can only be used with organizations where the serviceType returned by metadata catalog getOrganizations service is equal to "SOS2".

The following examples will use EPA's cloud-based data appliance. Information about available sensor data from this provider can be derived from the metadata catalog using the getOrganizations and getSensors service calls.

**Figure 1: Metadata Catalog getOrganizations Service Call**

[http://54.210.62.171/catalog/api/getOrganizations/?org\\_id=USEPA](http://54.210.62.171/catalog/api/getOrganizations/?org_id=USEPA)

### Result:

```
[{
  "organizationID": "USEPA",
  "lastPinged": "2016-12-27",
  "organizationSiteLink": null,
  "organizationPortalLink": null,
  "mostRecentHarvest": "2016-12-27",
  "defaultProcedure": null,
  "serviceEndpoint": "http://52.6.7.23/52n-sos-webapp/service?service=SOS&version=2.0.0",
  "serviceType": "SOS2",
  "organizationName": "U.S. EPA",
  "status": "up"
}]
```

**Figure 2: Metadata Catalog getSensors Service Call:**

[http://54.210.62.171/catalog/api/getSensors/?org\\_id=USEPA](http://54.210.62.171/catalog/api/getSensors/?org_id=USEPA)

## Result

```
{
  "type": "FeatureCollection",
  "features": [{
    "geometry": {
      "type": "Point",
      "coordinates": [-84.254731, 39.147167]
    },
    "properties": {
      "siteId": "urn:x-epaiwpp:feature:epa:ord:edmr-outfall",
      "siteName": "Outfall from EDMR",
      "id": "e33e4891-2482-4367-bab4-f2ca4c8279e5",
      "orgId": "USEPA"
    },
    "type": "Feature"
  },
  {
    "geometry": {
      "type": "Point",
      "coordinates": [-71.090129, 42.399288]
    },
    "properties": {
      "siteId": "urn:x-epaiwpp:feature:epa:epar1:mrbuoy",
      "siteName": "Mystic River",
      "id": "2fc892c7-5634-4872-8194-6c402616817a",
      "orgId": "USEPA"
    },
    "type": "Feature"
  },
  {
    "geometry": {
      "type": "Point",
      "coordinates": [-71.071376, 42.367187]
    },
    "properties": {
      "siteId": "urn:x-epaiwpp:feature:epa:epar1:crbuoy",
      "siteName": "Charles River",
      "id": "fe8a3118-6959-44ae-8f41-49e8e438ee4e",
      "orgId": "USEPA"
    },
    "type": "Feature"
  },
  {
    "geometry": {
      "type": "Point",
      "coordinates": [-122.592769, 48.838672]
    },
    "properties": {
      "siteId": "urn:x-epaiwpp:feature:epa:epar10:zap1",
      "siteName": "Ferndale, WA",
      "id": "fb4bb538-fbed-4d86-b963-2bffa278bb8c",
      "orgId": "USEPA"
    },
    "type": "Feature"
  }
  ]
}
```

Additional information about an individual sensor in the catalog can be derived using the getSensorParameters service call (note that the uuid parameter is derived from the results of the getSensors service call in Figure 2).

**Figure 3. Metadata Catalog getSensorParameters Service Call**

<http://54.210.62.171/catalog/api/getSensorParameters/?uuid=fe8a3118-6959-44ae-8f41-49e8e438ee4e>

**Result:**

```
{
  {
    "minObservationDate": "2015-06-10",
    "parameterOrgId": "urn:x-epaiwpp:observable:ph",
    "maxObservationDate": "2016-11-02",
    "parameterName": "pH",
    "id": 78426
  },
  {
    "minObservationDate": "2015-07-21",
    "parameterOrgId": "urn:x-epaiwpp:observable:turbidity",
    "maxObservationDate": "2016-11-02",
    "parameterName": "Turbidity",
    "id": 79196
  },
  {
    "minObservationDate": "2015-05-23",
    "parameterOrgId": "urn:x-epaiwpp:observable:temperature",
    "maxObservationDate": "2016-11-02",
    "parameterName": "Temperature, Water",
    "id": 79281
  },
  {
    "minObservationDate": "2015-06-17",
    "parameterOrgId": "urn:x-epaiwpp:observable:specific_conductance",
    "maxObservationDate": "2016-11-02",
    "parameterName": "Specific Conductance",
    "id": 79373
  },
  {
    "minObservationDate": "2015-05-26",
    "parameterOrgId": "urn:x-epaiwpp:observable:do_pct",
    "maxObservationDate": "2016-11-02",
    "parameterName": "Dissolved Oxygen Saturation",
    "id": 79586
  },
  {
    "minObservationDate": "2015-08-05",
    "parameterOrgId": "urn:x-epaiwpp:observable:dissolved_oxygen",
    "maxObservationDate": "2016-11-02",
    "parameterName": "Dissolved Oxygen",
    "id": 101627
  }
}
```

## GetObservation

The SOS 2.0 service GetObservation can be used to get the most recent observation for a sensor, or to retrieve the collection of observations for a defined time period. Compare the highlighted sections below to Figures 1, 2 and 3 to see the source of the information from the metadata catalog.

### Example Query to Retrieve the Most Recent Result for all Parameters Measured

<http://52.6.7.23/52n-sos-webapp/service?service=SOS&version=2.0.0&request=GetObservation&FeatureOfInterest=urn:x-epaiwpp:feature:epa:epar1:crbuoy>

### Example Query to Retrieve Results for a User Defined Time Period and Parameter

<http://52.6.7.23/52n-sos-webapp/service?service=SOS&version=2.0.0&request=GetObservation&FeatureOfInterest=urn:x-epaiwpp:feature:epa:epar1:crbuoy&ObservedProperty=urn:x-epaiwpp:observable:temperature&temporalFilter=om%3AphenomenonTime,2016-9-28T0:0:0.000Z/2016-11-2T0:0:0.000Z>

## GetDataAvailability

The SOS 2.0 service GetDataAvailability returns all of the parameters that are measured for a sensor along with the minimum and maximum observation dates. This is similar to the getSensorParameters service that is provided by the metadata catalog, but provides additional detail, such as the available procedures and any parameters that are not mapped in the metadata catalog.

<http://52.6.7.23/52n-sos-webapp/service?service=SOS&version=2.0.0&request=GetDataAvailability&FeatureOfInterest=urn:x-epaiwpp:feature:epa:epar1:crbuoy&ObservedProperty=urn:x-epaiwpp:observable:temperature>

## DescribeSensor

The SOS 2.0 service DescribeSensor provides descriptive information about the sensor, including descriptive name sensor metadata, and contact information when available. The procedure value for this service call must be derived from one of the other SOS 2.0 service calls such as GetDataAvailability.

<http://52.6.7.23/52n-sos-webapp/service?service=SOS&version=2.0.0&request=DescribeSensor&procedureDescriptionFormat=http%3A%2F%2Fwww.opengis.net%2FsensorML%2F1.0.1&procedure=urn:x-epaiwpp:sensor:epa:epar1:crbuoy:raw:temperature>

## Appendix B. 52° North SOS Installation, Ingestion and Notes

### Contents

Installing 52°North SOS.....	1
IWN SOS Ingester .....	3
Ingest Installation (Linux).....	3
Miscellaneous Notes.....	4
Setting Observable Property Names.....	4
Deleting Observations Added With InsertResult .....	6
IWN Uniform Resource Name (URN) Scheme .....	6
Configuration Files for SOS Data Ingestion .....	7

This section provides basic recipes and other information for installing and configuring 52°North SOS and the pilot IWN ingestion scripts on a Centos-based Linux data appliance.

### Installing 52°North SOS

- Install Java 1.8 SDK - <http://www.oracle.com/technetwork/java/javase/downloads/index.html>
- Install Tomcat 7 - <https://tomcat.apache.org/download-70.cgi>
  - [http://www.davidghedini.com/pg/entry/install\\_tomcat\\_7\\_on\\_centos](http://www.davidghedini.com/pg/entry/install_tomcat_7_on_centos) for useful hints
  - Tomcat 8 is also acceptable
  - Open port 8080 for access
- Install PostgreSQL 9.x - <https://www.postgresql.org/download/>
- Install PostGIS 2.2 - <http://www.postgis.net/>
  - [http://postgis.net/docs/postgis\\_installation.html](http://postgis.net/docs/postgis_installation.html)
  - StackBuilder can be used on Windows systems to install PostGIS
- Create SOS database and add PostGIS extensions:
  - Run psql and issue the following commands:

```
CREATE DATABASE sos;  
\c sos
```

```
CREATE EXTENSION postgis;  
CREATE EXTENSION postgis_topology;  
\q
```

- Install 52°North SOS:
  - Download and extract 4.3.x bundle from <http://52north.org/downloads/category/3-sos>
    - 4.3.7 used for IWN data appliances to date
    - 4.3.8 required if using PostgreSQL 9.6.x
  - Copy .war to [TOMCAT\_HOME]\webapps directory
- Configure 52°North SOS:
  - Connect to SOS client at <http://yourip:8080/52n-sos-webapp>
  - Configure data source, then enter provider information
  - Under Admin|Settings, confirm settings on Transactional Security tab and press the Save button
  - Under Admin|Settings, uncheck the checkbox “Should the SOS return overall extrema?” on the Miscellaneous tab and press the Save button.
    - This causes the SOS to return only the latest observation for each series (combination or procedure, observedProperty and featureOfInterest)."
  - Under Admin|Settings|Operations, make DeleteObservation, InsertObservation, InsertResult, InsertResultTemplate, and InsertSensor active.
  - Apply the OGC Sensor Observation Service 2.0 Hydrology Profile (requires direct access to computer hosting SOS):
    - Change the value of <profile:activeprofile> to **true** in [TOMCAT\_HOME]/webapps/52n-sos-webapp##4.3.x/WEB-INF/classes/profiles/hydrology-profile.xml
    - Change the value of <profile:activeprofile> to **false** in [TOMCAT\_HOME]/webapps/52n-sos-webapp##4.3.x/WEB-INF/classes/profiles/default-profile.xml
    - Reload the SOS instance

*Profile changes will be moved to the Admin web pages in version 4.4.x*

At this point, 52°North SOS should be configured and ready to accept data. Please see the Miscellaneous Notes section for an additional manual post-ingestion step necessary for proper data access related to observable property names.

## **IWN SOS Ingestor**

The IWN SOS Ingestor is a simple ingestion script written in Python and distributed for use with the Anaconda platform. It is designed to take observation data in CSV format and combine it with station and parameter metadata in CSV and JSON format to generate and submit appropriate InsertSensor, InsertResultTemplate, and InsertResult service requests to the SOS server. The ingest process assumes that the observational data is placed and updated regularly in a known location.

### **Ingest Installation (Linux)**

- Check for bzip2, unzip utilities needed to extract distribution files:

```
sudo yum install bzip2
sudo yum install unzip
```

- Download and install Anaconda Python distribution:

```
cd ~
wget https://repo.continuum.io/archive/Anaconda3-4.2.0-Linux-x86_64.sh
bash Anaconda3-4.2.0-Linux-x86_64.sh
```

- Download and install current ingest from <https://github.com/LimnoTech/IWN-Ingest/raw/master/IWN-Ingest.zip>.
- Configure and test:
  - Check metadata\_parameter.csv and metadata\_station.csv files for completeness
  - Ensure that station configuration files describing type and column assignments are available
  - Edit/create a script (Linux .sh, Windows .bat or .vbs) to perform the ingest that:
    - Pulls the data file from a different server or in a different directory to the data ingest directory (if needed);
    - Optionally trims to n most recent results using tail or similar
    - Invokes the ingest script
  - Additional details are provided in the miscellaneous notes section
- Set up a repeating task to update observations on a regular schedule using cron:

```
crontab -e
I (insert in vi)
*/15 * * * * cd ~/xml52north && bash ingest.sh dataref station.json
```



[enter] [esc]

:wq

Please see the Miscellaneous Notes section below for an additional manual post-ingestion step necessary for proper data access related to observable property names.

## Miscellaneous Notes

This section documents two additional actions – setting observable property names and deleting observations – that may be of interest to the user, and also provides guidance on constructing identifiers for SOS entities and on configuring the ingester.

### Setting Observable Property Names

The current (4.3.x) version of 52° North SOS does not read the field observableproperty:name from the InsertSensor submittals. This field is used in GetObservation for detailed identification of the observed parameter, and is therefore it is important to populate by logging on the server:

```
sudo psql -U postgres    --- Start postgres command-line processor
\c sos                   --- open 52N SOS database
```

```
select observablepropertyid, identifier, name from observableproperty where name = '';
```

observablepropertyid	identifier	name
211	urn:x-epai:wpp:observable:undefined	
212	urn:x-epai:wpp:observable:battery_volt age	
213	urn:x-epai:wpp:observable:temperature	
214	urn:x-epai:wpp:observable:conduct i v i t y	
215	urn:x-epai:wpp:observable:sal i n i t y	
216	urn:x-epai:wpp:observable:ph	
217	urn:x-epai:wpp:observable:turbi di t y	
218	urn:x-epai:wpp:observable:do_pct	
219	urn:x-epai:wpp:observable:di ssol ved_oxygen	
220	urn:x-epai:wpp:observable:water_level	

```
update observableproperty set name='n/a' where observablepropertyid=211;
```

```
.
```

```
update observableproperty set name='Water Level' where observablepropertyid=220;
```

```
select observablepropertyid, identifier, name from observableproperty;
```

observablepropertyid	identifier	name
211	urn:x-epai:wpp:observable:undefined	n/a
212	urn:x-epai:wpp:observable:battery_volt age	Battery
213	urn:x-epai:wpp:observable:temperature	Temperature, Water
214	urn:x-epai:wpp:observable:conduct i v i t y	Conduct i v i t y
215	urn:x-epai:wpp:observable:sal i n i t y	Sal i n i t y
216	urn:x-epai:wpp:observable:ph	pH
217	urn:x-epai:wpp:observable:turbi di t y	Turbi di t y
218	urn:x-epai:wpp:observable:do_pct	D0 (%)
219	urn:x-epai:wpp:observable:di ssol ved_oxygen	Di ssol ved Oxygen
220	urn:x-epai:wpp:observable:water_level	Height, gage

```
\q
```

```
---- Exit psql
```

For consistency purposes, you should use the Substance Registry System ([www.epa.gov/srs](http://www.epa.gov/srs)) to identify a common name for each parameter. This will allow your data to be discovered, accessed and understood

in a consistent manner with other data sets. SRS parameter names used to date in pilot watershed partner datasets are shown in Table B.1

Version 4.4 of 52° North SOS is expected to address this issue and remove the need for manual updating.

**Table B-1. SRS Parameter Names, Synonyms and Units Found in Partner Datasets**

SRS Parameter Name	Synonyms found in Pilot	Units found in Pilot
<b><i>Water Quality</i></b>		
Blue Green Algae-Phaeocystin (proposed)		(cells/mL, RFU)
Chlorophyll	Chlorophyll RAW	(ug/L, RFU)
Conductivity		(mS/cm, uS/cm)
Depth		(m, ft)
Dissolved Oxygen	YSI DO, DO, ODO	(mg/L, ppm)
Dissolved Oxygen Saturation	ODOSat	(mg/L, ppm)
<i>Dissolved Oxygen Percent Saturation (no SRS)</i>	DO%	(%)
Inorganic Nitrogen (Nitrate and Nitrite)		(mg/L)
Oxidation Reduction Potential (ORP)	ORP	(mV)
pH	YSI pH	(none)
pHmv		(mV)
Salinity		(ppt, psu, (none))
Specific Conductance	YSI SP Cond, Sp Cond	(mS/cm, uS/cm)
Temperature, Water	Temperature, Temp, YSI Temperature	(°C, °F, °K)
Turbidity	Turb	(NTU)
<b><i>Hydraulic/Hydrological</i></b>		
Flow	Flow Rate	(m3/s, ft3/s, gpm)
Flow, Total	Total Flow	(m3)
Height, Gage	Levl, 720 Level	(ft, m)
Stream Velocity	Velocity	(m/s, ft/s)
<b><i>Other</i></b>		
Barometric Pressure	Rel. BP, Air Pressure	(mb, mmHg, psi, atm)
Dew Point	Wet Bulb Temp	(°C, °F)
Light, Photosynthetic Active Radiation (PAR)	Light	(mV, umol/s/m2, mmol/m2/15min)
Precipitation	Rainfall	(mm, in, count)
Relative Humidity	RH	(%)
Temperature, Air	Temperature, AirTemp, Air Temperature	(°C, °F)
Wind Direction	WindDir	(°)
Wind Gust, Peak	Max WindSp	(m/s, mph)
Wind Velocity	WindSP	(m/s, mph)

## Deleting Observations Added With InsertResult

During the development of the pilot IWN, stakeholders identified the need to delete observations from the database. This is supported in 52° North SOS through the DeleteObservation operation, but the IWN ingest scripts' use of the InsertResult operation to add observations does not populate the observation:identifier field used to select observations for deletion. Two solutions are available:

- To populate the observation:identifier field, connect to the SOS database and issue the command

```
UPDATE observation SET identifier = 'o' || observationid::text;
```

The gml:identifier shown in GetObservation requests can then be used in DeleteObservation requests.

- To delete all observations associated with a sensor connect to the SOS database and issue the command:

```
UPDATE observation
SET   deleted = 'T'
WHERE observation.seriesid IN (SELECT seriesid
                               FROM   (series
                                         INNER JOIN procedure
                                         ON   procedure.procedureid = series.procedureid
                                         AND
                                              identifier = 'urn:x-epaiwpp:sensor:njdep:bfbm:01367693:final:Dissolved Oxygen'));
```

In either case, the DeleteObservation or SQL shown above ONLY set the observation:deleted field and do not actually delete the observations. Removal of the observations requires logging in to the Datasource Panel (<http://localhost:8080/52n-sos-webapp/admin/datasource>) and pressing the "Delete deleted Observations" button.

## IWN Uniform Resource Name (URN) Scheme

Uniform Resource Names (URNs) are used extensively to provide unique machine-readable identifiers for different entities represented in 52N SOS-based data appliances deployed on behalf of Pilot partners. This document describes the basic elements of the URN scheme.

URNs consist of the term "urn:" followed by a namespace ID and a namespace-specific string. The namespace ID is currently "x-epaiwpp", so all URNs will begin with the text "urn:x-epaiwpp". The organization, suborganization, station, and parameter IDs are specified in metadata files for the data appliance.

## Organizations, suborganization and station IDs

Every organization in the Pilot must have a unique name identifier. The name identifier will begin with either a two-letter state postal abbreviation or "US" for national-scale organizations:

- usepa – United States Environmental Protection Agency
- njdep – New Jersey Department of Environmental Protection

- njmeri – Meadowlands Environmental Research Institute (located in New Jersey)
- ohclec – Clermont County, Ohio – OR OH39025 (FIPS-BASED)

Organizations are assumed to have suborganizations such as:

- usepa:ord – Office of Research and Development (EPA)
- ohclec:wr – Water Resources Division of Clermont County, Ohio
- njmeri:meri – no suborganization, organization acronym repeated

Station IDs are assigned by the organization, and consist of alphanumeric characters.

## Parameter IDs

Parameter IDs are used to uniquely identify observable properties, sensors, features, offerings and templates within 52N SOS. The parameter IDs must be consistent across all organizations.

## Observable property URNs

Observable property URNs are consistent across the entire network, and consist of the namespace ID, followed by the classifier “parameter” and the parameter ID:

*urn:x-epaiwpp:parameter:temperature*

## Station, offering, sensor, feature, and template URNs

URNs for stations, sensor, features, and templates are created by concatenation:

- Station URNs identify platforms deployed for sensors, and consist of the namespace ID followed by the classifier “station”, and the organization, suborganization, and station IDs:

*urn:x-epaiwpp:station:ohclec:wr:efrm34.8*

- Sensor URNs identify sensors deployed at a platform, and consist of the namespace ID followed by the classifier “sensor”, the organization, suborganization, and station IDs, a data quality status indicator (“raw”, “provisional” or “final”), and the sensor parameter:

*urn:x-epaiwpp:sensor:ohclec:wr:efrm34.8:raw:temperature*

Offering, feature, and template URNs are structured similarly to the sensor URNs but use a different classifier:

*urn:x-epaiwpp:offering:ohclec:wr:efrm34.8:raw:temperature*

*urn:x-epaiwpp:feature:ohclec:wr:efrm34.8:raw:temperature*

*urn:x-epaiwpp:template:ohclec:wr:efrm34.8:raw:temperature*

## Configuration Files for SOS Data Ingestion

This section describes the contents of important files for ingestion of observation data into an IWN 52N SOS-based data appliance.

### *IWN-ingest.py*

IWN-ingest.py contains the main body of the IWN ingestion script. When invoked, the user should specify the CSV-formatted file containing observation data to be ingested, and the accompanying JSON-formatted station configuration file:

```
IWN-ingest observations.csv station.json or
```

```
IWN-ingest observations.csv input/station.json
```

The ingestion code checks the SOS database to identify the most recent available observation for a given parameter and station, and only uploads observations that are more recent. Two typical use cases have been identified in the IWN pilot project: batch and near real-time.

### *Batch (manual) operation*

For batch operations – the occasional ingestions of long-term, typically historical and lengthy records – direct use of the ingestion program is suggested:

```
~/anaconda/python IWN-ingest historical.csv input/station.json (Linux example)
```

### *Near real-time*

For continuous near real-time operations, scheduled invocation of a .sh (Linux) or .bat/.vbs (Windows) script is suggested. The following Linux script pulls an observation file from another site, creates a truncated data file with only the most recent data points, and ingests it:

```
wget http://someplace/posted.csv
```

```
head posted.csv > nrt.csv
```

```
tail -10 posted.csv >> nrt.csv
```

```
~/anaconda/python IWN-ingest nrt.csv input/station.json
```

### *dataconvert.py*

dataconvert.py is called by IWN-ingest.py to parse the CSV-formatted file potentially containing observations for multiple parameters and stations into “chunks” containing observations for a single parameter and station.

### *Station configuration file (station.json)*

The station configuration file is used to guide the parsing of CSV-formatted observations into chunks for ingestion. The file contains four elements:

- File type – indicates general format of the CSV file:
  - “type”:1 – data for a single station, date and time are in separate columns
  - “type”:2 – data for a single station, date and time are in a single column
  - “type”:3 – data for multiple stations, date and time are in a single column
- Columns – array that identifies which parameter in the metadata\_parameter file (see below) corresponds to each column in the CSV file:
  - “columns":["date","time","pH"] interprets the first two columns as the datetime stamp and the third column as pH measurements
  - “columns":["","datetime","pH"] skips the first column, presents the second column as the datetime stamp, and uses the third column for pH

- Valid array values are "" (skip), "date", "time", "datetime", or any match for "parameter" values entered in metadata\_parameter.csv
- Station – identifies the station entry in the metadata\_station file:
  - "station": "stationid"
  - The station string must match a stationid value in the metadata\_station file.
- Header – specifies how many lines to ignore at the beginning of the file:
  - "header": 3

The following example is shown to demonstrate formatting:

```
{
  "type": 2,
  "columns": [
    "datetime",
    "water_level",
    "ph",
    "dissolved_oxygen",
    "specific_conductance",
    "temperature",
    "rainfall"],
  "station": "grssy0",
  "header": 7
}
```

#### *Station metadata file (station\_metadata.csv)*

The station metadata file station\_metadata.csv must be stored in the ./input subdirectory. It should contain the following elements in the indicated order for each station in CSV format, with the data starting on line 2 of the file:

- Station ID – a text field unique to the station containing no spaces. This field is used to (1) link an ingested .CSV file's contents to a station through the JSON station configuration file's "station" element, and (2) as part of the URNs used to uniquely identify stations, sensors, offerings, features, and result templates in IWN data appliances.
- Short Name – A text field containing a unique short descriptive name for the station.
- Long Name – A text field containing a unique long descriptive name for the station.
- Longitude – Longitude of station location in decimal degrees.
- Latitude – Latitude of station location in decimal degrees.
- Altitude – Altitude of station location in meters.
- Organization – A text field containing the name of the organization/office collecting the observations.
- Web URL – A URL linking to a web page specified by the organization, presumable relevant to the observations being collected.
- Contact – An e-mail address for a party responsible for the collected observations.
- Waterbody Type – A text field indicating the type of waterbody from which observations are collected. Examples include Estuary, Lake, River/Stream, Outfall
- Publisher – Organization publishing the data
- Data Status – Text field indicating QA/QC stats of the data. Examples include raw, provisional and final.

- URN org part – A text field with no spaces unique to the organization collecting the data, used in the SOS URNs specifying stations, sensors, offerings, features and result templates.
- URN suborg A text field with no spaces unique to the sub-organization (bureau, office, etc.) collecting the data, used in the SOS URNs specifying stations, sensors, offerings, features and result templates.

*Parameter metadata file (parameter\_metadata.csv)*

The station metadata file station\_metadata.csv must be stored in the ./input subdirectory. It should contain the following elements in the indicated order for each parameter in CSV format, with the data starting on line 2 of the file:

- Parameter – a text field unique to the station containing no spaces. This field is used to (1) link an ingested .CSV file’s contents to a parameter through the JSON station configuration file’s “column” element, and (2) as part of the URNs used to uniquely identify stations, sensors, offerings, features, and result templates in IWN data appliances.
- Parameter Name – Pretty name for parameter used in SOS templates; currently (52N SOS 4.3.x) must be entered manually in database.
- Parameter Units – Units for the parameter
- Parameter Field Name – A text field containing no spaces that is used for naming the observable property element in SOS.

Status – Text field indicating QA/QC stats of the data. Examples include raw, provisional and final.

## **Appendix C. User stories developed by the Watersheds-based Monitoring and Interoperable Data Platforms Watershed Workgroups.**

### **Hackensack-Passaic Watershed Workgroup**

#### **User Story 1. Drinking Water/Source Water Protection Early Warning**

As a water manager, I want to view trends in selected parameters so that I can predict and remediate a water quality issue before it occurs.

##### ***Acceptance Criteria***

- The user can find their location of interest on an interactive map.
- The user can specify a parameter, and observe which stations collect data on that parameter.
- The user can click on a station, and view a time series chart with the most recently available data. If data for a particular time period are available at multiple quality levels (for example, draft, provisional or final) the highest quality data will be displayed.
- The user can adjust the time frame on the time series chart so that they can see how values have been trending over time.
- The user will be able to select a different parameter for the selected sensor and view those results in the time series chart while the time frame is held constant.
- The user can download the parameter data from the sensor in a delimited text file if desired.

#### **User Story 2. Water Quality Assessment (for Clean Water Act Integrated Reporting)**

As a water quality manager, I want to download continuous monitoring parameter data so that I can compare it to numeric criteria and evaluate if the water is meeting state standards.

##### ***Acceptance Criteria***

- The user can find their location of interest on an interactive map.
- The user will be able to view stations that are registered in the metadata catalog on the map.
- The user must select stations within a geographic area of interest.
- The user must specify a start date and end date of interest.
- The user can choose to download all parameters, or specify one or more parameters of interest.
- The user can choose to download data from all organizations, or specify one or more specific organization.
- The user can choose to download data with all quality designations, or specify specific quality designations (for example, if data can be raw, provisional or final, users can specify to only download final data).
- The user will be able to download a delimited text file containing available parameter data in a consistent format from all data providers.



## **Little Miami Watershed Workgroup**

### **User Story 1. Water Safety (Drinking Water and Recreation) optimization: Maximizing output while minimizing cost with a Harmful Algal Bloom focus**

As a water manager, I need to detect water quality issues, such as a harmful algal bloom, so that I can alert the public.

- The user can find their location of interest on an interactive map.
- The user can specify a parameter, and observe which stations collect data on that parameter.
- The user can click on a station, and view a time series chart with the most recently available data. If data for a particular time period are available at multiple quality levels (for example, draft, provisional or final) the highest quality data will be displayed.
- The user can adjust the time frame on the time series chart so that they can see how values have been trending over time.
- The user will be able to select a different parameter for the selected sensor and view those results in the time series chart while the time frame is held constant.
- The user can download the parameter data from the sensor in a delimited text file if desired.

### **User Story 2. TMDL Implementation**

As a water quality manager, I need to be able to download parameter data for use in running TMDL models.

#### ***Acceptance Criteria***

- The user can find their location of interest on an interactive map.
- The user will be able to view stations that are registered in the metadata catalog on the map.
- The user must select stations within a geographic area of interest.
- The user must specify a start date and end date of interest.
- The user can choose to download all parameters, or specify one or more parameters of interest.
- The user can choose to download data from all organizations, or specify one or more specific organization.
- The user can choose to download data with all quality designations, or specify specific quality designations (for example, if data can be raw, provisional or final, users can specify to only download final data).
- The user will be able to download a delimited text file containing available parameter data in a consistent format from all data providers.

## Appendix D. Currents Discovery Tool Features and Enhancements

The watershed workgroups were asked to develop requirements list for the Currents Discovery Tool and prioritize them using the MoSCoW method. This process assigns a value to each item as follows.

**Must have** – highest priority requirements. Critical for project success.

**Should have** – second highest priority requirements. As critical as highest priority requirements, but with mitigating factors (i.e. there may be a workaround available if the requirement cannot be implemented, or the requirement may be implemented in a later release).

**Could have** – lower priority requirements. Not essential to project success but “nice to have.”

**Won’t have** – least critical requirements OR out of scope due to project resources at this time. Things that will not be on the schedule for the current development cycle.

The tables below list the features identified by the workgroups at the beginning of the project. Features indicated with a ✓ have been implemented in the tool. Features with a ⚡ are partially available. Features marked with an ✗ are not currently available.

Feature	Priority	Currents Discovery Tool
<b><i>Tools for Selecting By Attributes</i></b>		
Date Range	Must Have	✓
Parameter (1 or more)	Must Have	✓
Filter by organization	Must Have	✓
Filter by multiple criteria (dates, parameters, and org)	Must Have	✓
Filter by Highest Available QC Levels (raw, provisional, final)	Won’t Have	✗
Early versions of the metadata catalog conceptual model included flags to indicate the QC levels of data available from a provider. The workgroups decided to exclude this feature because of the intricacies of communicating which status levels were available for selected time periods. Eventually this was also dropped from the conceptual model due to variability in the way QC levels are reported.		
Search by hydrologic feature type (bay, estuary, river...)	Should Have	✗
There is currently not a good way to derive the type of hydrologic feature a site is located on. In the future, “site type” from the metadata catalog conceptual model may be a good surrogate for this.		
Select using waterbodies (enter a waterbody name and get results)	Won’t Have	✗
A service is being developed that provides a waterbody name search based on NHD/GNIS. The user must select a state to narrow their search results, and then the results are presented by County. When this service is available, it could be used by the Currents Discovery Tool.		
Select using equipment type	Should Have	✗
None of the current data partners make this data element available yet.		
<b><i>Tools for Spatial Selections</i></b>		
Select by User Defined Polygon	Must Have	✓
Select by Watershed	Must Have	✓

Select by County	Must Have	✓
Point with Buffer Selection	Should Have	0
This is not implemented in the Currents Discovery Tool, but this service is available via the metadata catalog.		
Stream Network Navigation from a point on a stream.	Must Have	0
This is not implemented in the Currents Discovery Tool, but upstream and downstream services are available via the metadata catalog.		
<b>Other Query Features</b>		
Provide toggle to limit query results to the current map window.	Must Have	✓
Provide the ability to add sites to the selected result set.	Must Have	✓
Provide the ability to remove sites from the selected result set.	Must Have	✓
<b>Map Features</b>		
Address/Location Finder	Must Have	✓
Zoom to selected set	Must Have	✓
Toggle Base Map (imagery, topographic map)	Must Have	✓
Zoom to current (physical) location	Should Have	0
This is implemented in the Currents Discovery Tool, however the Chrome browser places a restriction on providing user location information to sites that are not securely hosted ( <a href="https://developers.google.com/web/fundamentals/native-hardware/user-location/">https://developers.google.com/web/fundamentals/native-hardware/user-location/</a> ). As a result, the feature is hidden when the Chrome browser is detected.		
Zoom to Previous	Could Have	X
This feature was not identified as a priority.		
Reset/Start Over	Could Have	X
This feature was not identified as a priority – but can be accomplished functionally simply by refreshing the browser window.		
<b>Data Download Features</b>		
Download site/parameter data as a comma delimited text file	Must Have	✓
Specify data that are within a range.	Should Have	✓
Provide option to download data for a single site.	Must Have	0
Users can download data for a single site/parameter combination.		
Download everything from the query (sites, all params, in a single file)	Must Have	X
This large scale download functionality is not readily available using the simple architecture selected for the Currents Discovery Tool.		
Data results will include QA status	Must Have	0
This information will be included in the download when it is provided in a format that can be recognized by the Currents Discovery Tool. This currently includes data provided by USGS and the partner providers that follow the standardized URN scheme for identifying “procedure”.		
Specify data that are above or below a criteria	Should Have	X

This feature is not currently available. Additional discussions with the workgroup would be required to determine if adding this for the single site/parameter download would be useful.		
<b><i>Time Series Charts</i></b>		
Modify date range for data displayed on chart	Must Have	✓
Tooltip that shows data point details	Must Have	✓
Display Multiple Parameters	Must Have	X
This is possible, but has not yet been implemented.		

## Other Features

Other features that have been discussed for the Currents Discovery Tool include presenting a detailed metadata page for a selected organization. This page could provide summary counts or sites, types of available data and links to metadata documents. This would also be an opportunity to expose the organization website and organization level data portal when these things are populated in the metadata catalog. There has also been discussion about showing which data providers currently are not providing data in the organization list.