

Deliverable No.: 2.6.4

**STorage and RETrieval System (STORET) –
Water Quality Exchange (WQX)**

STORET Web Services Consumption User Guide

Version 2.0

August 3, 2007

**ITS-ESE Task Order No.: 50
Contract No.: 68-W-04-005**

Developed for:



**United States Environmental Protection Agency
Office of Environmental Information
1200 Pennsylvania Avenue, NW
Washington, DC 20460**

Developed by:

LOCKHEED MARTIN



**Systems Engineering Center (SEC)
1010 N. Glebe Road
Arlington, VA 22201**

Document No.: 50-WQX-MAN-0064

Approvals

This document has been reviewed and approved for use in the project development process. Minor changes and corrections may be made to this document without re-approval.

Date: 08/03/2007

Ernestine Bryant, Task Order (TO) 50 Contract Task Manager (CTM), Lockheed Martin, ITS-ESE Program

Date: 08/03/2007

Ganesh Thadkamalla, Project Tech Lead, Lockheed Martin, ITS-ESE Program

Date: 08/03/2007

Steve Siegel, Quality Assurance (QA), Lockheed Martin, ITS-ESE Program

Document Change History

Version #	Date	Change Author	Change Summary
1.0	06/28/2007	Sree Rayankula	Initial version
1.0	06/29/2007	Reneé Rufo	Technical editing
1.1	07/23/2007	Sree Rayankula	Reformatted and updated the content
1.1	07/26/2007	Reneé Rufo	Technical editing
2.0	08/03/2007	Sree Rayankula	Reformatted and updated the content. Added Section 2 and Appendix D.

TABLE OF CONTENTS

1.0	INTRODUCTION.....	4
1.1	Purpose.....	4
1.2	Scope.....	4
1.3	Acronyms and Definitions.....	4
1.4	Document Overview.....	5
2.0	STORET WEB SERVICES.....	6
2.1	Project Catalog Web Service.....	6
2.1.1	getProjectInfoByParameters.....	7
2.1.2	getProjectInfoByLatLong.....	8
2.2	Station Catalog Web Service.....	9
2.2.1	getWatershedOrgStationCharTypeSummary.....	10
2.2.2	getStationCharTypeSummaryUsingXMLInput.....	10
2.2.3	getStationCharacteristicSummary.....	11
2.2.4	getStationCharTypeSummaryUsingStringInput.....	12
2.2.5	getWebServiceCatalog.....	13
2.2.6	getWatershedStationCharTypeSummary.....	13
2.3	Station Web Service.....	14
2.3.1	getStations.....	14
2.3.2	getWebServiceCatalog.....	15
2.4	STORET Result Web Service.....	16
2.4.1	getActivityCount.....	16
2.4.2	getResults.....	18
2.5	Watershed Summary Web Service.....	21
2.5.1	getOrganizationSummary.....	22
2.5.2	getCharTypeCharacteristicSummary.....	23
2.5.3	getCharacteristicSummary.....	24
2.5.4	getOrganizationCharTypeSummary.....	25
2.5.5	getOrganizationCharTypeCharacteristicSummary.....	25
2.5.6	getCharTypeSummary.....	26
2.5.7	getWebServiceCatalog.....	27
3.0	STORET WEB SERVICES HOME PAGES.....	28
3.1	View or Download WSDL Document.....	29
3.2	Test Web Service Operations.....	30
3.3	Download OC4J Proxy Client jar Files.....	31
4.0	CONSUMING STORET WEB SERVICES.....	32
5.0	REFERENCES.....	34
APPENDICES.....		35
	APPENDIX A: Jar Files Included in the CLASSPATH of the Consumer Application.....	35
	APPENDIX B: Description of Configuration File Parameters Used With WebServicesAssembler.....	36
	APPENDIX C: Sample Station Web Service Client Program, Using Client-side Proxy.....	37
	APPENDIX D: Pictorial View of Web Service Output Schema.....	39

1.0 INTRODUCTION

This document provides a description of the STOrage and RETrieval (STORET) Web Services and an introduction to consuming STORET Web Services from other applications. STORET Web Services were developed using OracleWebServicesAssembler tool, and they are hosted on an Oracle Application Server (OAS). The STORET Web Services are based on Simple Object Access Protocol (SOAP) Remote Procedure Call (RPC). They consist of Java classes written to implement the core functionality.

1.1 Purpose

This document aims at providing guidelines to the software developer community intending to consume STORET Web Services. A description of the operations supported by each of the STORET Web Services is also provided.

1.2 Scope

The currently available STORET Web Services are described in detail in later sections of the document:

- Project Catalog Web Service
- Station Catalog Web Service
- Station Web Service
- STORET Result Web Service
- Watershed Summary Web Service.

Methodology used to consume all of the STORET Web Services is similar. For example, this document outlines the steps involved in consuming the Station Web Service. The same steps should be followed to access the remaining STORET Web Services while using the Uniform Resource Locators (URLs) and client proxy classes relevant to the Web Service(s) being consumed. Operations supported by each of the Web Services are listed in this document, along with a list of input parameters and other details required for requesting information using the individual operations.

The document will provide the steps to help consume the Web Service. However, it does not provide the details of parsing the Extensible Markup Language (XML) output returned by the Web Service.

1.3 Acronyms and Definitions

The following table provides a brief description for terms and acronyms used throughout this document.

Acronym	Definition
CTM	Contract Task Manager
EPA	U.S. Environmental Protection Agency
ITS-ESE	Information Technology Solutions – Environmental Systems Engineering
Jar	Java archive
JRE	Java Runtime Environment
OAS	Oracle Application Server
OC4J	Oracle Containers for Java
OW	Office of Water
QA	Quality Assurance

Acronym	Definition
RPC	Remote Procedure Call
SEC	Systems Engineering Center
SOAP	Simple Object Access Protocol
STORET	STORage and RETrieval
TBD	To Be Determined
TO	Task Order
UDDI	Universal Description Discovery and Integration
URL	Uniform Resource Locator
WSDL	Web Service Description Language
WQX	Water Quality Exchange
XML	Extensible Markup Language

1.4 Document Overview

The document provides a brief description of STORET Web Services and the procedure involved in consuming STORET Web Services, using a static Web Service client. In Section 2.0, the individual services are described in detail in addition to the methods including data type, wild card conventions and exceptions. A context diagram is also provided to give a high level view of the consumption process. Section 3.0 gives an overview of the Web Service home pages provided by the Oracle Web Services. Section 4.0 describes the procedure involved in consuming a STORET Web Service.

2.0 STORET WEB SERVICES

The STORET Data Warehouse consolidates the surface and ground water quality data received from states, tribes and agencies across the United States. Thus, the STORET data warehouse is a comprehensive source of water quality monitoring data for the entire nation. STORET Web Services are based on STORET Data Warehouse, and they serve as a means of retrieving nationwide water quality information based on a few search criteria.

Station Catalog Web Service, Watershed Summary Web Service and Project Catalog Web Service provide summary information of the water quality monitoring result data available for stations, watershed and projects. Station Web Service provides monitoring location information of all the stations within a geographic bounding box and STORET Result Web Service provides result data.

STORET Web Services are registered to the U.S. Environmental Protection Agency (EPA) Universal Description Discovery and Integration (UDDI) registry. The URL to the EPA UDDI registry and the provider name for STORET Web Services is shown in Table 1.

Table 1. UDDI Registry

URL of the EPA UDDI Registry	https://uddi.epacdxnode.net/uddi/bsc/web
Provider Name	Environmental Protection Agency (EPA) Office of Water STORET Web Services

The following sections provide the list of Web methods supported by each of the Web Services.

2.1 Project Catalog Web Service

Web Service End Point: <http://iaspub.epa.gov/webservices/ProjectCatalogService/>

Web Service Description Language (WSDL) Location:
<http://iaspub.epa.gov/webservices/ProjectCatalogService/index.html?WSDL>

Description: Project Catalog Web Service provides project wide summary of the number of results for each characteristic that was monitored. The Web Service provides two Web methods. Both of the Web methods, provided by the Web Service, return information of the same format and type. The information returned by the Web methods of Project Catalog Web Service is as follows:

- Identifier and name of each organization and project summary.
- Attributes pertaining to projects that meet the specified criteria, including the project identifier, project name, activity start date, activity end date, characteristic count, result count and result information for each characteristic within the characteristic types found for the specified criteria.
- Names and identifiers of monitoring locations where the project related activities were conducted.

Project Catalog Web Service provides the following Web methods:

- getProjectInfoByParameters
- getProjectInfoByLatLong.

These methods are described, in detail, in the following subsections with the parameters and information required for consumption.

2.1.1 getProjectInfoByParameters

Signature: getProjectInfoByParameters(String OrganizationIdentifier, String ProjectIdentifier, String StationIdentifier, String HydrologicUnitcode)

Description: Returns project information that satisfies the specified criteria. Information returned by the Web method is listed under “Description” in section 2.1.

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty value is required for all the required parameters. Input parameters of the Web method are listed in Table 2.

Table 2. *getProjectInfoBy*

Serial #	Input Parameter	Data Type	Mandatory/ Optional	Description
1	Organization Identifier	String	Conditionally optional	Organization Identifier is required whenever a non-empty value is provided for Project Identifier or Station Identifier.
2	Project Identifier	String	Optional	When a non-empty value is specified for Project Identifier, Organization Identifier must also be provided.
3	Station Identifier	String	Optional	When a non-empty value is specified for Station Identifier, Organization Identifier must also be provided.
4	Hydrologic Unit Code	String	Optional	Identifier of the watershed, for which information is being requested.

Output Format: An XML document that conforms to the Project Catalog Web Service output schema. A pictorial view of the Project Catalog Web Service output XML schema can be seen in Figure 6, Appendix D.

Output XML Schema Location: TBD

Exceptions that are likely to be thrown, while consuming the Web method are listed in Table 3.

Table 3. *Possible Exceptions*

Exception Type	Potential Cause for the Exception
javax.xml.rpc.soap.SOAPFaultException	<p>One of the following input conditions will cause SOAPFaultException:</p> <ol style="list-style-type: none"> 1. A non-empty value was provided for Project Identifier, but a null value or an empty string was provided for organization identifier. 2. A non-empty value was provided for Station Identifier, but a null value or an empty string was provided for organization identifier.

Example usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/ProjectCatalogService/";
ProjectCatalogServiceProxy wsStub = new ProjectCatalogServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element projectCatalogXML = wsStub.getProjectInfoByParameters(
        "emap-cs","","","","" );
    //process projectCatalogXML to extract the required information
}catch(...){
}
```

2.1.2 getProjectInfoByLatLong

Signature: getProjectInfoByLatLong(float MinimumLatitude, float MaximumLatitude, float MinimumLongitude, float MaximumLongitude)

Description: Returns project information from all the monitoring locations that fall under the specified geographic bounding box. Information returned by the Web method is listed under "Description" in section 2.1.

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty string value is required for all the required parameters. Input parameters of the Web method are listed in Table 4.

Table 4. *getProjectInfoByLatLong*

Serial #	Input Parameter	Data Type	Mandatory/ Optional	Description
1	Minimum Latitude	float	Mandatory	A non-numeric value will result in java.lang.NumberFormatException.
2	Maximum Latitude	float	Mandatory	A non-numeric value will result in java.lang.NumberFormatException.
3	Minimum Longitude	float	Mandatory	A non-numeric value will result in java.lang.NumberFormatException.
4	Maximum Longitude	float	Mandatory	A non-numeric value will result in java.lang.NumberFormatException.

Output Format: An XML document that conforms to the Project Catalog Web Service output schema. A pictorial view of the Project Catalog Web Service output XML schema can be seen in Figure 6, Appendix D.

Output XML Schema Location: TBD

Exceptions that are likely to be thrown, while consuming the Web method are listed in Table 5.

Table 5. *Possible Exceptions*

Exception Type	Potential Cause for the Exception
Java.lang.NumberFormatException	A non-numeric value was provided for one more of the input parameters.

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/ProjectCatalogService/";
ProjectCatalogServiceProxy wsStub = new ProjectCatalogServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element projectCatalogXML = wsStub.getProjectInfoByLatLong(
        New Float(30), new Float(40), new Float(-90), new Float(-70));
    //process projectCatalogXML to extract the required information
}catch(...){
}
```

2.2 Station Catalog Web Service

Web Service End Point: <http://iaspub.epa.gov/webservices/StationCatalogService/>

WSDL Location: <http://iaspub.epa.gov/webservices/StationCatalogService/index.html?WSDL>

Description: The Station Catalog Web Service provides a station-wide summary of results available for each characteristic. The Web Service supports Web methods that accept a different set of parameters, and ultimately returns the result summary information for monitoring locations. The station summary information returned by all the Web methods except the 'getWebServiceCatalog' contains the following information:

- Identifier and name of each organization
- Station identifier, station name and the hydrologic unit code of the watershed to which the station belongs
- Activity start date, activity stop date, characteristic count and result count for each station
- Characteristic-based result counts for each characteristic type within each of the stations being returned.

In addition, the Station Catalog Web Service provides the 'getWebServiceCatalog' Web method that allows the clients to request information about all the other Web methods provided by the Web Service.

The following is a list of Web methods supported by the Station Catalog Web Service:

- getWatershedOrgStationCharTypeSummary
- getStationCharTypeSummaryUsingXMLInput
- getStationCharacteristicSummary
- getStationCharTypeSummaryUsingStringInput
- getWebServiceCatalog
- getWatershedStationCharTypeSummary.

The following subsections describe the above methods, list the parameters, and provide the information for consumption.

2.2.1 getWatershedOrgStationCharTypeSummary

Signature: getWatershedOrgstationCharTypeSummary(String HydrologicUnitCode, String OrganizationIdentifier)

Description: Provides station catalog information described in section 2.2 for the specified criteria.

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty value is required, for all the required parameters. Input parameters of the Web method are listed in Table 6.

Table 6. *getWatershedOrgStationCharTypeSummary*

Serial #	Input Parameter	Data Type	Mandatory/ Optional	Description
1	Hydrologic Unit Code	String	Mandatory	Unique identifier of the watershed.
2	Organization Identifier	String	Mandatory	Identifier of the organization that has monitoring locations within the watershed.

Output Format: An XML document that conforms to the Station Catalog Web Service output schema. A pictorial view of the Station Catalog Web Service output XML schema can be seen in Figure 7, Appendix D.

Output XML Schema Location: TBD

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/StationCatalogService/";
StationCatalogServiceProxy wsStub = new StationCatalogServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element stationCatalogXML = wsStub.getWatershedOrgStationCharTypeSummary
        ("02060007", "MDEDAT04");
    //process stationCatalogXML to extract the required information
}catch(...){
}
```

2.2.2 getStationCharTypeSummaryUsingXMLInput

Signature: getstationCharTypeSummaryUsingXMLInput(Element xmlStationIdsDocument)

Description: Provides station catalog information described in section 2.2 for the specified criteria.

Input Parameters: Arguments provided as input must be non-null values. Input parameters of the Web method are listed in Table 7.

Table 7. *getStationCharTypeSummaryUsingXMLInput*

Serial #	Input Parameter	Data Type	Mandatory/ Optional	Description
1	Station XML	Element	Mandatory	An XML document containing a list of organization and station identifiers, for which information is being requested.

Input XML Schema Location: TBD

Output Format: An XML document that conforms to the Station Catalog Web Service output schema. A pictorial view of the Station Catalog Web Service output XML schema can be seen in Figure 7, Appendix D.

Output XML Schema Location: TBD

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/StationCatalogService/";
StationCatalogServiceProxy wsStub = new StationCatalogServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element stationIdentifierXML = getStationIdentifiersXML();
    Element stationCatalogXML = sStub.getStationCharTypeSummaryUsingXMLInput
        (stationIdentifierXML);
    //process stationCatalogXML to extract the required information
}catch(...){
}
```

2.2.3 getStationCharacteristicSummary

Signature: getstationCharacteristicSummary(String CharacteristicType, String StationIdentifier, String OrganizationIdentifier)

Description: Provides station catalog information described in section 2.2 for the specified criteria.

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty value is required for all parameters marked mandatory. Input parameters of the Web method are listed in Table 8.

Table 8. *getStationCharacteristicSummary*

Serial #	Input Parameter	Data Type	Mandatory/ Optional	Description
1	Char Type	String	Mandatory	Characteristic type name.
2	Station Identifier	String	Mandatory	Identifier of the monitoring location.
3	Organization Identifier	String	Mandatory	Identifier of the organization.

Output Format: An XML document that conforms to the Station Catalog Web Service output schema. A pictorial view of the Station Catalog Web Service output XML schema can be seen in Figure 7, Appendix D.

Output XML Schema Location: TBD

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/StationCatalogService/";
StationCatalogServiceProxy wsStub = new StationCatalogServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element stationCatalogXML = wsStub.getStationCharacteristicSummary
        ("Microbiological", "WISCHU", "MDEDAT08");
    //process stationCatalogXML to extract the required information
}catch(...){
}
```

}

2.2.4 getStationCharTypeSummaryUsingStringInput

Signature: getStationCharTypeSummaryUsingStringInput(String StationIdentifierInputstring)

Description: Provides station catalog information described in section 2.2 for the specified criteria.

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty value is required for all the parameters marked mandatory. Input parameters of the Web method are listed in Table 9.

Table 9. *getStationCharTypeSummaryUsingStringInput*

Serial #	Input Parameter	Data Type	Mandatory/ Optional	Description
1	Station Identifier Input String	String	Mandatory	Organization Identifier and Station identifier are separated by a comma, and enclosed inside parenthesis. Information for multiple stations can be requested by specifying a number of organization identifier and station identifier pairs, each separated by a semi-colon. Example 1: (EMAP-CS, VA93-640) Example 2: (EMAP-CS, VA93-640);(MDEDAT04, ADW0001)

Output Format: An XML document that conforms to the Station Catalog Web Service output schema. A pictorial view of the Station Catalog Web Service output XML schema can be seen in Figure 7, Appendix D.

Output XML Schema Location: TBD

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/StationCatalogService/";
StationCatalogServiceProxy wsStub = new StationCatalogServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element stationCatalogXML
        wsStub.getStationCharTypeSummaryUsingStringInput
            ("(EMAP-CS, VA93-640);(MDEDAT04, ADW0001)");
    //process stationCatalogXML to extract the required information
}catch(...){
}
```

2.2.5 getWebServiceCatalog

Signature: getWebServiceCatalog().

Description: This Web method is intended to help the client application obtain the details of Web methods supported by the parent Web Service. It returns the following information for each of the other Web methods supported by the Web Service:

- Names of Web methods.
- End point of the Web Service, WSDL document location, service protocol and transfer method for each Web method.
- Name, label and type of input parameters for each Web method.
- Format of the output returned by each Web method.

Input Parameters: None

Output Format: An XML document that conforms to the Web Services Catalog output schema. A pictorial view of the Web Services Catalog output XML schema can be seen in Figure 11, Appendix D.

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/StationCatalogService/";
StationCatalogServiceProxy wsStub = new StationCatalogServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element catalogXML = wsStub.getWebServiceCatalog();
    //process catalogXML to extract the required information
} catch(...){
}
```

2.2.6 getWatershedStationCharTypeSummary

Signature: getWatershedStationCharTypeSummary(String HydrologicUnitCode).

Description: Provides station catalog information described in section 2.2 for the specified criteria.

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty value is required for all the required parameters. Input parameters of the Web method are listed in Table 10.

Table 10. *getWatershedCharTypeSummary*

Serial #	Input Parameter	Data Type	Mandatory/ Optional	Description
1	Hydrologic Unit Code	String	Mandatory	Unique identifier of the watershed.

Output Format: An XML document that conforms to the Station Catalog Web Service output schema. A pictorial view of the Station Catalog Web Service can be seen in Figure 7, Appendix D.

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/StationCatalogService";
StationCatalogServiceProxy wsStub = new StationCatalogServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element stationCatalogXML = wsStub.getWatershedStationCharTypeSummary
        ("02060007");
    //process stationCatalogXML to extract the required information
} catch(...){
}
```

}

2.3 Station Web Service

Web Service End Point: <http://iaspub.epa.gov/webservices/StationService/>

WSDL Location: <http://iaspub.epa.gov/webservices/StationService/index.html?WSDL>

Description: The Station Web Service provides information pertaining to all the monitoring locations within a specified geographic bounding box. The service shall also provide an organization identifier and formal name of parent organization to which the monitoring location(s) belong. A geographic bounding box is specified by using minimum latitude, maximum latitude, minimum longitude and maximum longitude. In addition, the Web Service provides the 'getWebServiceCatalog()' Web method, which lists the Web methods and details of input parameters for each method. The Web method is provided to help applications such as EnviroMapper consume the Station Web Service.

The following is a list of Web methods supported by the Station Web Service:

- getStations
- getWebServiceCatalog.

The following subsections describe the above methods, list the parameters, and provide the information for consumption.

2.3.1 getStations

Description: Returns the following information for the monitoring locations located within the specified geographic bounding box:

- Identifier and name of all the parent organizations that have monitoring locations within the specified geographic bounding box.
- Attributes related to Monitoring Location Identity, including Monitoring Location Identifier, Monitoring Location Name, Monitoring Location Type Name, Eight Digit Hydrologic Unit Code, and Hydrologic Unit Name for all the Monitoring Locations within the specified geographic bounding box.
- Attributes that help establish Geospatial Identity of the monitoring locations, such as Latitude Measure, Longitude Measure, Horizontal Collection Method Name, County Name, State Name and Country Name.

For a complete list of Monitoring Location Identity and Geospatial Identity attributes returned by the Web Service, please refer to the Station Web Service Output Schema.

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty value is required for all the required parameters. Input parameters of the Web method are listed in Table 11.

Table 11. *getStations*

Serial #	Input Parameter	Data Type	Mandatory/ Optional	Description
1	Minimum Latitude	float	Mandatory	Must be a numeric value. Non-numeric values cause a <code>java.lang.NumberFormatException</code> .
2	Maximum Latitude	float	Mandatory	Must be a numeric value. Non-numeric values cause a

Serial #	Input Parameter	Data Type	Mandatory/Optional	Description
				java.lang.NumberFormatException.
3	Minimum Longitude	float	Mandatory	Must be a numeric value. Non-numeric values cause a java.lang.NumberFormatException.
4	Maximum Longitude	float	Mandatory	Must be a numeric value. Non-numeric values cause a java.lang.NumberFormatException.

Output Format: An XML document that conforms to the STORET Result Web Service output schema. A pictorial view of the Station Web Service output XML schema can be seen in Figure 8, Appendix D.

Exceptions that are likely to be thrown, while consuming the Web method are listed in Table 12.

Table 12. Possible Exceptions

Exception Type	Potential Cause for the Exception
java.lang.NumberFormatException	A non-numeric value was provided for one more of the input parameters.

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/StationService/";
StationServiceProxy wsStub = new StationServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element stationXML = wsStub.getStations(new Float(30), new Float(40),
        New Float(-90), new Float(-70));
    //process stationXML to extract the required information
}catch(...){
}
```

2.3.2 getWebServiceCatalog

Description: This Web method is intended to help the client application obtain the details of Web methods supported by the parent Web Service. It returns the following information for each of the other Web methods supported by the same Web Service:

- Names of Web methods
- End point of the Web Service, location of WSDL document, service protocol and transfer method for each Web method.
- Name, label and type of input parameters for each Web method.
- Format of the output returned by each Web method.

Input Parameters: None.

Output Format: An XML document that conforms to the Web Services Catalog output schema. A pictorial view of the Web Services Catalog output XML schema can be seen in Figure 11, Appendix D.

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/StationService/";
StationServiceProxy wsStub = new StationServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element catalogXML = wsStub.getWebServiceCatalog();
    //process catalogXML to extract the required information
}catch(...){
}
```

2.4 STORET Result Web Service

Web Service End Point: <http://iaspub.epa.gov/webservices/StoretResultService/>

WSDL Location: <http://iaspub.epa.gov/webservices/StoretResultService/index.html?WSDL>

Description: The STORET Result Web Service returns result information of organizations based on the criteria specified. The result information is returned as part of the parent activity for each organization. The results XML output returned by the Web Services conforms to the XML output schema defined for the Web service.

The following is a list of Web methods supported by STORET Result Web Service:

- getActivityCount
- getResults.

Special Considerations: At times, the criteria specified for obtaining the result information can lead to a huge amount of result data being returned by the Web service. This condition can potentially result in unavailability and scalability issues for the Web Service. To ensure good scalability and availability, the result output returned by the Web Service is restricted using a start activity number and an end activity number. A maximum limit is also enforced on the number of activities returned. The current limit is set to **100** activities per request. If the specified range of start activity number and end activity number exceed the maximum number of activities allowed, then the end activity number provided by the client application will be ignored. The service will return the maximum number of activities allowed beginning with the start activity number.

The following subsections describe the Web methods of the Web Service, list the parameters, and provide the information for consuming the Web methods.

2.4.1 getActivityCount

Description: Returns the number of activities available for the specified criteria. The total number of activities available can be used to request the result information using multiple calls to the 'getResults' Web method. Knowledge of the number of activities available for the specified criteria can help breakdown the request for results into a number of pages, where the start activity number and end activity number for each request is obtained based on the total number of activities available. The idea of pagination is explained in section 2.4.2.

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty value is required for all the required parameters. Input parameters of the Web method are listed in Table 13.

Table 13. *getActivityCount*

Serial #	Input Parameter	Data Type	Mandatory/ Optional	Description
1	Organization Identifier	String	Conditionally optional	A value must be provided for the Organization Identifier whenever a value is specified for one of the following: Activity Identifier, Monitoring Location Identifier, Project Identifier.
2	Activity Identifier	String	Optional	The Organization Identifier must also be specified whenever an Activity Identifier is specified. Otherwise, the call to the Web method will result in a SOAPFaultException. <i>Exact match to a valid value is not required. Partial values are recognized.</i>
3	Activity Type	String	Optional	Monitoring activity type. <i>Exact match to a valid value is not required. Partial values are recognized.</i>
4	Monitoring Location Identifier	String	Optional	The Organization Identifier must also be specified whenever a Monitoring Location Identifier is specified. Otherwise, the call to the Web method will result in a SOAPFaultException.
5	Monitoring Location Type	String	Optional	Type of monitoring location. <i>Exact match to a valid value is not required. Partial values are recognized.</i>
6	Project Identifier	String	Optional	Organization Identifier must also be specified whenever a Project Identifier is specified. Otherwise the call to the web method will result in a SOAPFaultException.
7	Minimum Activity Start Date	String	Optional	Minimum Activity Start Date of the format MM/DD/YYYY.
8	Maximum Activity Start Date	String	Optional	Maximum Activity Start Date of the format MM/DD/YYYY.
9	Characteristic Type	String	Optional	Characteristic type name. <i>Exact match to a valid value is not required. Partial values are recognized.</i>
10	Characteristic Name	String	Optional	Display or search name of the characteristic. <i>Exact match to a valid value is not required. Partial values are recognized. For instance, the following two values for characteristic name are considered valid by the Web Service:</i> Example 1: Nitrogen, ammonium (NH4) as NH4 Example 2: NH4

Output Format: An integer that represents the number of activities embedded in a SOAP envelope.

Exceptions that are likely to be thrown, while consuming the Web method are listed in Table 14.

Table 14. Possible Exceptions

Exception Type	Potential Cause for the Exception
javax.xml.rpc.soap.SOAPFaultException	One of the following conditions will cause a SOAPFaultException: <ol style="list-style-type: none">1. All of the values passed as arguments to the Web method are empty.2. A non-empty value was specified for one of the following input parameters: Activity Identifier, Monitoring Location Identifier, and Project Identifier, but the value provided for the Organization Identifier was empty.3. Minimum or Maximum Activity Start Date was provided in an incorrect format.
java.lang.NullPointerException	A null value was provided for one of the input parameters.

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/StoretResultService/";
StoretResultServiceProxy wsStub = new StoretResultServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Integer objActivityCount = wsStub.getActivityCount("EMAP-CS", "", "",
        "", "", "", "01/01/1980", "01/01/1998", "", "");
    int activityCount = objActivityCount.intValue();
} catch(...){
}
```

2.4.2 getResults

Description: Returns result information based on specified criteria. The result information returned for each organization is attached to the parent activity. The volume of output returned by the Web method is restricted by requiring the Web method consumer to specify a start activity number and an end activity number. In addition, the maximum number of activities returned by the Web method is set to a pre-determined value. The current value of maximum number of activities is set to **100**. It is re-configurable; hence, it can be changed at any time. The number of activities requested is determined by the start activity number and the end activity number. If this number exceeds the limit set on the maximum number of activities, then the Web method will return only up to the maximum number of allowed activities beginning at the start activity number.

Pagination: Using the start activity number and the end activity number, results for a given set of criteria can be obtained in multiple requests. The following description is provided as an example for pagination:

Assume that the maximum number of activities available for the given criteria using the Web method 'getActivityCount("org1", "", "", "", "", "", "", "", "", "")' is 100. The results may be requested in multiple requests of chosen page size. If the client decides to obtain 20 at a time, then it will take five requests for the client to obtain all the results. In this case, the five requests used to obtain the results would be:

```
getResults("org1", "", "", "", "", "", "", "", "", "", new Integer(1), new Integer(20));
getResults("org1", "", "", "", "", "", "", "", "", "", new Integer(21), new Integer(40));
getResults("org1", "", "", "", "", "", "", "", "", "", new Integer(41), new Integer(60));
getResults("org1", "", "", "", "", "", "", "", "", "", new Integer( 61), new Integer(80));
getResults("org1", "", "", "", "", "", "", "", "", "", new Integer(81), new Integer(100));
```

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty value is required for all the required parameters. Input parameters of the Web method are listed in Table 15.

Table 15. *getResults*

Serial #	Input Parameter	Data Type	Mandatory/ Optional	Description
1	Organization Identifier	String	Conditionally optional	A value must be provided for the organization identifier whenever a value is specified for one of the following: Activity Identifier, Monitoring Location Identifier, Project Identifier.
2	Activity Identifier	String	Optional	The Organization Identifier must also be specified whenever an Activity Identifier is specified. Otherwise, the call to the Web method will result in a SOAPFaultException. <i>Exact match to a valid value is not required. Partial values are recognized.</i>
3	Activity Type	String	Optional	Monitoring activity type. <i>Exact match to a valid value is not required. Partial values are recognized.</i>
4	Monitoring Location Identifier	String	Optional	The Organization Identifier must also be specified whenever a Monitoring Location Identifier is specified. Otherwise, the call to the Web method will result in a SOAPFaultException.
5	Monitoring Location Type	String	Optional	Monitoring location type. <i>Exact match to a valid value is not required. Partial values are recognized.</i>

Serial #	Input Parameter	Data Type	Mandatory/Optional	Description
6	Project Identifier	String	Optional	The Organization Identifier must also be specified whenever a Project Identifier is specified. Otherwise, the call to the Web method will result in a SOAPFaultException.
7	Minimum Activity Start Date	String	Optional	Minimum Activity Start Date of the format MM/DD/YYYY.
8	Maximum Activity Start Date	String	Optional	Maximum Activity Start Date of the format MM/DD/YYYY.
9	Characteristic Type	String	Optional	Characteristic type name. <i>Exact match to a valid value is not required. Partial values are recognized.</i>
10	Characteristic Name	String	Optional	Display or search name of the characteristic. <i>Exact match to a valid value is not required. Partial values are recognized.</i>
11	Starting Activity Number	int	Mandatory	Lower bound of the range of activities for which the results are being requested.
12	Ending Activity Number	int	Mandatory	Upper bound of the range of activities for which the results are being requested.

Output Format: An XML document that conforms to the STORET Result Web Service output schema. A pictorial view of the STORET Result Web Service output XML schema can be seen in Figure 9, Appendix D.

Exceptions that are likely to be thrown, while consuming the Web method are listed in Table 16.

Table 16. Possible Exceptions

Exception Type	Potential Cause for the Exception
javax.xml.rpc.soap.SOAPFaultException	<p>One of the following conditions will cause a SOAPFaultException:</p> <ol style="list-style-type: none"> 1. All of the values passed as arguments to the web method are empty. 2. A non-empty value was specified for one of the following input parameters: Activity Identifier, Monitoring Location Identifier, and Project Identifier, and the organization Identifier is provided an empty value. 3. Minimum or Maximum Activity Start Date was not provided in correct format.

Exception Type	Potential Cause for the Exception
	<p>4. Values supplied for Start Activity Number and End Activity Number are not valid. Following are two examples of invalid input for Start/ End Activity Number: one or both of the activity numbers are not positive integers, or Start Activity Number is greater than the End Activity Number.</p>
java.lang.NullPointerException	A null value was provided for one of the input parameters.

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/StoretResultService/";
StoretResultServiceProxy wsStub = new StoretResultServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element resultXML = wsStub.getResults("EMAP-CS", "", "",
        "", "", "", "01/01/1980", "01/01/2001", "",
        "", new Integer(1), new Integer(2));
    //process resultXML to extract the required information
} catch(...){
}
```

2.5 Watershed Summary Web Service

Web Service End Point: <http://iaspub.epa.gov/webservices/WatershedSummaryService/>

WSDL Location: <http://iaspub.epa.gov/webservices/WatershedSummaryService/index.html?WSDL>

Description: The Watershed Summary Web Service provides organization and characteristic summary information aggregated at the watershed level or aggregated for a given organization within the watershed. The operations provided by the Web Service are described in the sub-sections of this section. The XML output returned by the Web methods conform to the XML schema given below.

The following is a list of Web methods supported by the Watershed Summary Web Service:

- getOrganizationSummary
- getCharTypeCharacteristicSummary
- getCharacteristicSummary
- getOrganizationCharTypeSummary
- getOrganizationCharTypeCharacteristicSummary
- getCharTypeSummary.

The subsections below describe the above methods, list the parameters, and provide the information for consuming the Web methods.

2.5.1 getOrganizationSummary

Description: The Web method returns a summary of result information for each organization within the watershed identified by the given hydrologic unit code. The organization summary information returned by the Web method includes characteristic types, number and names of characteristics studied and the number of results logged for each characteristic. The types of information returned by the Web method are listed below:

- Number of organizations within the watershed
- Number of stations within the watershed
- Total number of characteristics monitored within the watershed
- Total number of results logged throughout the watershed
- Summary information about each organization within the watershed, including organization identifier, organization name, activity start date, activity end date and number of stations
- Characteristic types monitored for each organization within the watershed
- Characteristics of each characteristic type that were monitored, and the number of results available for each characteristic.

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty value is required for all the required parameters. Input parameters of the Web method are listed in Table 17.

Table 17. *getOrganizationSummary*

Serial #	Input Parameter	Data Type	Mandatory/ Optional	Description
1	Hydrologic Unit Code	String	Mandatory	Unique watershed identifier.

Output Format: An XML document that conforms to Watershed Summary Web Service output schema. A pictorial view of the Watershed Summary Web Service output XML schema can be seen in Figure 10, Appendix D.

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/WatershedSummaryService/";
WatershedSummaryServiceProxy wsStub = new WatershedSummaryServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element orgInfo = wsStub.getOrganizationSummary("02060007");
    //process the result XML
}catch(...){
}
```

2.5.2 getCharTypeCharacteristicSummary

Description: Provides result summary information for the specified characteristic type within a specified watershed, identified by the Hydrologic Unit Code. The following information can be extracted from the XML returned by the Web method:

- Total number of characteristics of all characteristic types monitored within the watershed
- Total number of results logged for all characteristic types within the watershed
- Number of characteristics of the specified characteristic type that were monitored within the given watershed
- Number of results available for each characteristic of the specified characteristic type within the watershed.

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty value is required for all the required parameters. Input parameters of the Web method are listed in Table 18.

Table 18. *getCharTypeCharacteristicSummary*

Serial #	Input Parameter	Data Type	Mandatory/Optional	Description
1	Characteristic Type	String	Mandatory	Characteristic type name.
2	Hydrologic Unit Code	String	Mandatory	Unique identifier of the watershed.

Output Format: An XML document that conforms to Watershed Summary Web Service output schema. A pictorial view of the Watershed Summary Web Service output XML schema can be seen in Figure 10, Appendix D.

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/WatershedSummaryService/";  
WatershedSummaryServiceProxy wsStub = new WatershedSummaryServiceProxy();  
wsStub._setSoapURL(_soapURL);  
try{  
    Element charTypeCharInfo =  
        wsStub.getCharTypeCharacteristicSummary("Metal", "02060007");  
    //process the result XML  
}catch(...){  
}
```

2.5.3 getCharacteristicSummary

Description: Given a hydrologic unit code, the Web method returns a count of characteristics that were used for monitoring and the total number of results logged for the watershed identified by the hydrologic unit code. The method also lists the characteristic types and number of results for each characteristic for the characteristic types listed. The method provides the following types of information for the specified watershed:

- Total number of characteristics monitored in the watershed
- Total number of results logged for the watershed
- Characteristic types that were monitored in the watershed
- Individual characteristics of each characteristic type that were studied
- Total number of results logged for each characteristic within the watershed.

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty value is required for all the required parameters. Input parameters of the Web method are listed in Table 19.

Table 19. *getCharacteristicSummary*

Serial #	Input Parameter	Data Type	Mandatory/ Optional	Description
1	Hydrologic Unit Code	String	Mandatory	Unique watershed identifier.

Output Format: An XML document that conforms to Watershed Summary Web Service output schema. A pictorial view of the Watershed Summary Web Service output XML schema can be seen in Figure 10, Appendix D.

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/WatershedSummaryService/";
WatershedSummaryServiceProxy wsStub = new WatershedSummaryServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element charInfo = wsStub.getCharacteristicSummary("02060007");
    //process the result XML
}catch(...){
}
```

2.5.4 getOrganizationCharTypeSummary

Description: Given a hydrologic unit code, the Web method returns the organizations within the watershed and the number of results available for each organization across each characteristic type. The output XML returned by the Web method can be parsed to obtain the following types of information:

- Number of organizations within the watershed
- Number of stations within the watershed
- Total number of characteristics monitored within the watershed
- Total number of results logged throughout the watershed
- High level information about each organization within the watershed, including organization identifier, organization formal name, activity start date, activity stop date, number of stations, total results
- Characteristic types monitored for each organization
- Characteristic count and result count for each characteristic of the characteristic types returned.

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty value is required for all the required parameters. Input parameters of the Web method are listed in Table 20.

Table 20. *getOrganizationCharTypeSummary*

Serial #	Input Parameter	Data Type	Mandatory/Optional	Description
1	Hydrologic Unit Code	String	Mandatory	Unique watershed identifier.

Output Format: An XML document that conforms to Watershed Summary Web Service output schema. A pictorial view of the Watershed Summary Web Service output XML schema can be seen in Figure 10, Appendix D.

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/WatershedSummaryService/";
WatershedSummaryServiceProxy wsStub = new WatershedSummaryServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element orgCharTypeSummary = wsStub.getOrganizationCharTypeSummary(
        "02060007");
    //process the result XML
} catch(...){
}
```

2.5.5 getOrganizationCharTypeCharacteristicSummary

Description: Given the characteristic type, organization identifier and hydrologic unit code, the method returns the number of results available for each characteristic within the characteristic type for the specified organization and watershed.

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty value is required, for all the required parameters. Input parameters of the Web method are listed in Table 21.

Table 21. *getOrganizationCharTypeCharacteristicSummary*

Serial #	Input Parameter	Data Type	Mandatory/ Optional	Description
1	Char Type	String	Mandatory	Characteristic type name.
2	Organization Identifier	String	Mandatory	Organization identifier.
3	Hydrologic Unit Code	String	Mandatory	Unique watershed identifier.

Output Format: An XML document that conforms to the Watershed Summary Web Service output schema. A pictorial view of the Watershed Summary Web Service output XML schema can be seen in Figure 10, Appendix D.

Output XML Schema Location: TBD

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/WatershedSummaryService/";
WatershedSummaryServiceProxy wsStub = new WatershedSummaryServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element orgCharTypeCharInfo =
        wsStub.getOrganizationCharTypeCharacteristicSummary(
            "metal", "emap-cs", "02060007");
    //process the result XML
} catch(...){
}
```

2.5.6 getCharTypeSummary

Description: Returns the number of results and characteristics logged for a given watershed. A breakdown of the number of results by characteristic type is also included in the result XML.

Input Parameters: Arguments provided as input must be non-null values. An empty string is allowed to be passed for all the string type parameters that are marked optional. A non-empty value is required for all the required parameters. Input parameters of the Web method are listed in Table 22.

Table 22. *getCharTypeSummary*

Serial #	Input Parameter	Data Type	Mandatory/ Optional	Description
1	Hydrologic Unit Code	String	Mandatory	Hydrologic unit code used to identify the watershed.

Output Format: An XML document that conforms to the Watershed Summary Web Service output schema. A pictorial view of the Watershed Summary Web Service output XML schema can be seen in Figure 10, Appendix D.

Output XML Schema Location: TBD

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/WatershedSummaryService/";
WatershedSummaryServiceProxy wsStub = new WatershedSummaryServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element charTypeInfo = wsStub.getCharTypeSummary("02060007");
    //process the result XML
}catch(...){
}
```

2.5.7 getWebServiceCatalog

Description: This Web method is intended to help the client application obtain the details of the Web methods supported by the parent Web Service. It returns the following information for each of the other Web methods supported by the same Web Service:

- Names of Web methods
- End point of the Web service, WSDL location, service protocol and transfer method for each Web method
- Name, label and type of input parameters for each Web method.
- Format of the output returned by each Web method.

Input Parameters: None.

Output Format: An XML document that conforms to the Web Services Catalog output schema. A pictorial view of the Web Services Catalog output XML schema can be seen in Figure 11, Appendix D.

Example Usage:

```
String _soapURL = "http://iaspub.epa.gov/webservices/WatershedSummaryService/";
WatershedSummaryServiceProxy wsStub = new WatershedSummaryServiceProxy();
wsStub._setSoapURL(_soapURL);
try{
    Element catalogXML = wsStub.getWebServiceCatalog();
    //process catalogXML to extract the required information
}catch(...){
}
```

3.0 STORET WEB SERVICES HOME PAGES

Each of the Web Services described in Section 2.0 have a home page. The home page can be accessed using the end point of the Web Service. The end points to the STORET Web Services are listed in Table 23.

Table 23. Web Services

Name of the Web Service	End Point of the Web Service
Project Catalog Web Service	http://iaspub.epa.gov/webservices/ProjectCatalogService/
Station Catalog Web Service	http://iaspub.epa.gov/webservices/StationCatalogService/
Station Web Service	http://iaspub.epa.gov/webservices/StationService/
STORET Result Web Service	http://iaspub.epa.gov/webservices/StoretResultService/
Watershed Summary Web Service	http://iaspub.epa.gov/webservices/WatershedSummaryService/

Figure 1 depicts a screenshot of the Station Web Service home page.

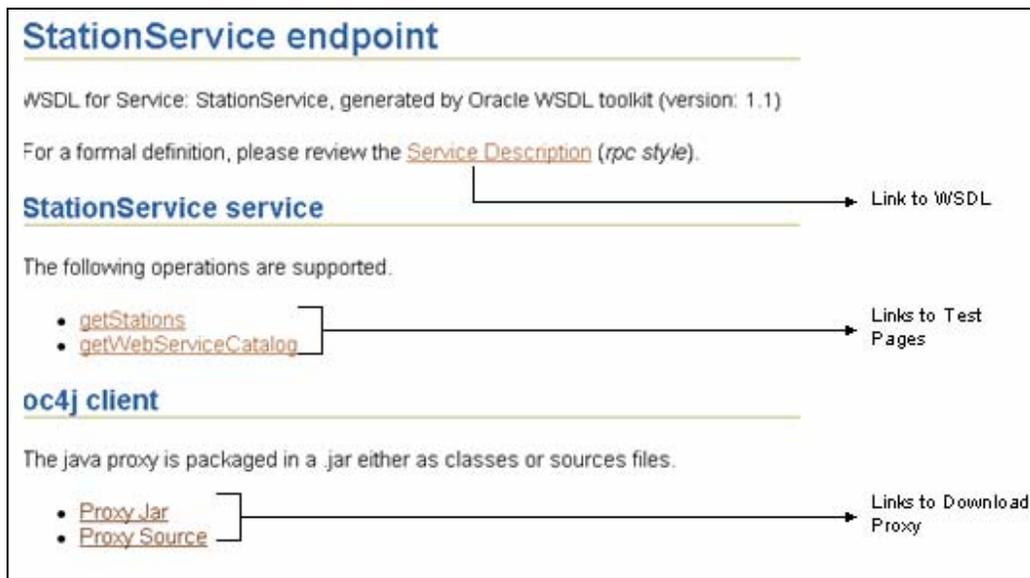


Figure 1. —Station Web Service Home Page.

As shown in Figure 1 above, the Web Service home page can be used to

- View or download the WSDL document.
- Test the operations of the Web Service.
- Download the Oracle Containers for Java (OC4J) proxy client files for the Web Service.

3.1 View or Download WSDL Document

As shown in Figure 2, follow the hyperlink 'Service Description' provided on the Web Service home page to view the WSDL document of the Web Service.

To download the WSDL document for the Web Service, right click the hyperlink 'Service Description' and save the target as an XML file (Figure 2).

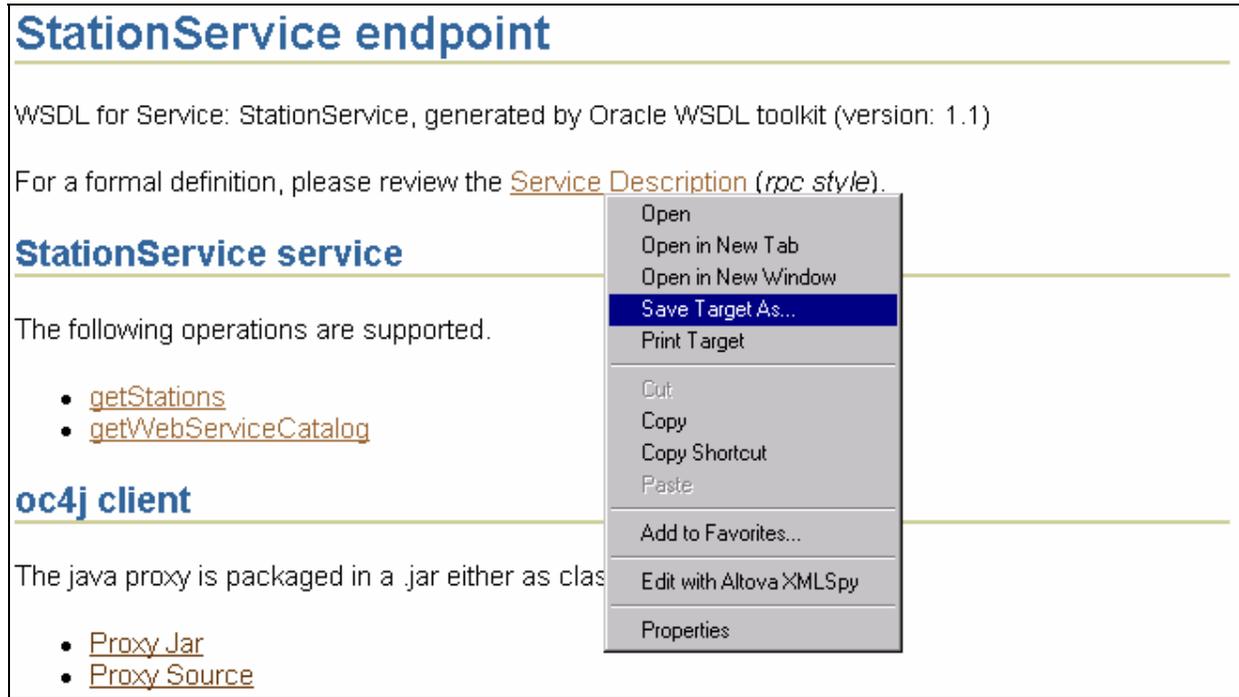


Figure 2. —Downloading the WSDL Document.

The WSDL document for the Web Service can also be directly accessed using the URLs provided in Table 24.

Table 24. WSDL URLs

Name of the Web Service	URL to the WSDL Document
Project Catalog Web Service	http://iaspub.epa.gov/webservices/ProjectCatalogService/index.html?WSDL
Station Catalog Web Service	http://iaspub.epa.gov/webservices/StationCatalogService/index.html?WSDL
Station Web Service	http://iaspub.epa.gov/webservices/StationService/index.html?WSDL
STORET Result Web Service	http://iaspub.epa.gov/webservices/StoretResultService/index.html?WSDL
Watershed Summary Web Service	http://iaspub.epa.gov/webservices/WatershedSummaryService/index.html?WSDL

3.2 Test Web Service Operations

Hyperlinks to test pages for the Web Service operations are provided on the Web Service home page. Each of the Web Service operations can be tested using the test pages. However, the following are two exceptional cases when the operations cannot be tested using test pages provided on the STORET Web Service home pages:

1. Complex input parameters for Remote Procedure Call (RPC) style Web Services are not supported; the invoke button is not provided when there are complex input parameters.
2. Document Style Web Services are not supported; the invoke button is not provided for document style Web Services.

getStationCharTypeSummaryUsingXMLInput (Element) of Station Catalog Web Service cannot be tested using test pages because of the first bullet listed above.

Figure 3 is a screenshot of the test page provided for 'getStations(float, float, float, float) ' operation of the Station Web Service:

Click [here](#) for a complete list of operations.

getStations

Test

To test the operation using the HTTP GET protocol, click the 'Invoke' button.

Parameter	Type	Value
MinimumLatitude	float	<input type="text"/>
MaximumLatitude	float	<input type="text"/>
MinimumLongitude	float	<input type="text"/>
MaximumLongitude	float	<input type="text"/>

Figure 3.—Test Page for getStations (float, float, float, float) Operation.

The user may enter values in the input fields and click the 'Invoke' button to get the output from the Web Service operation.

3.3 Download OC4J Proxy Client jar Files

Download OC4J Proxy Client Class jar File

To download the Java Archive (jar) file containing compiled OC4J proxy client classes, follow the steps provided below:

1. Right click the hyperlink 'Proxy Jar' displayed on the Web Service home page.
2. A context menu is displayed. From the context menu, click the option 'Save Target As...'. A 'Save As' popup is displayed, as shown Figure 4. Then
 - navigate to the target location, where you wish to save the proxy source file
 - give a name to the file being downloaded and make sure the file extension is 'jar'
 - change the 'Save as Type' to 'All Files.'

Figure 4 is a screenshot of the download operation:

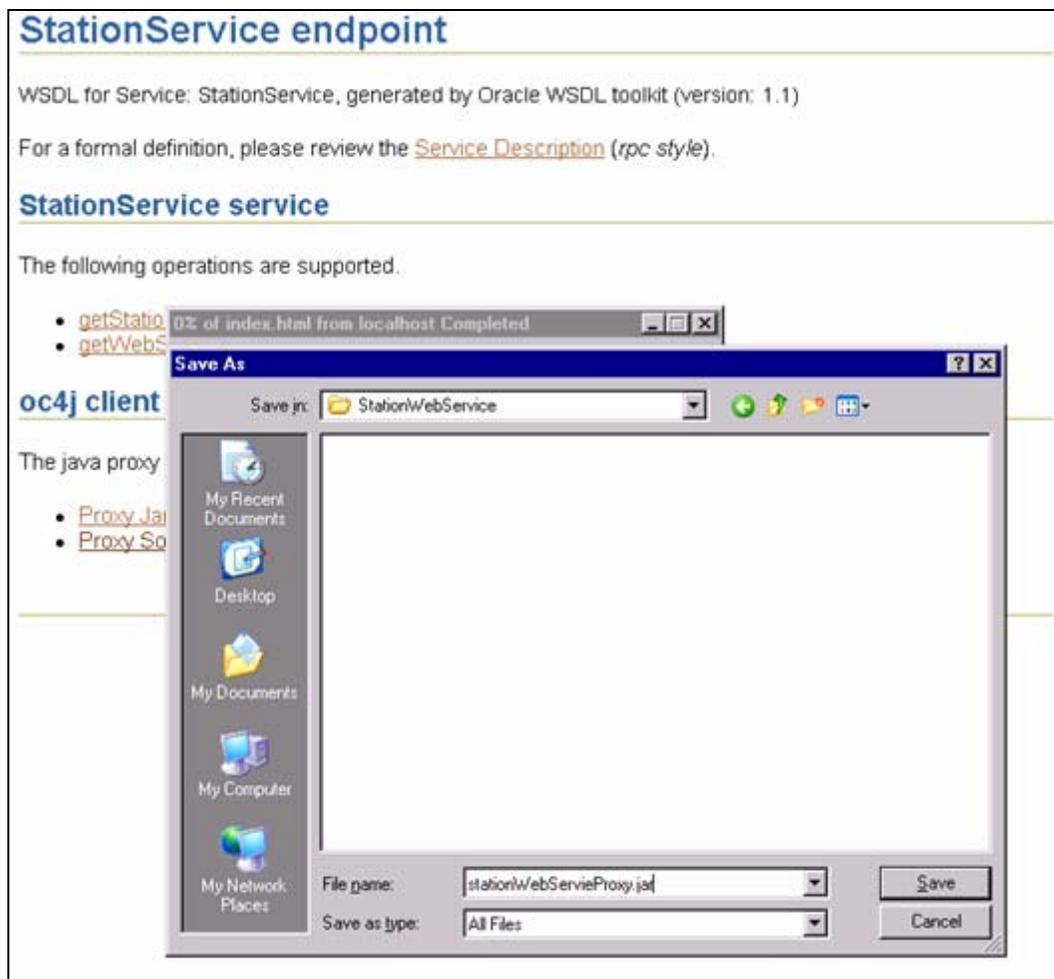


Figure 4.—Download OC4J Client Proxy from Web Service Home Page.

Download Java Source jar File

To download the OC4J client proxy source jar file, use the hyperlink 'Proxy Source' and follow the instructions listed above to download the jar file.

4.0 CONSUMING STORET WEB SERVICES

A static Web Service client knows where a Web Service is located, and a UDDI lookup will not be required at runtime. The location of the WSDL document of the Web Service can be used to obtain proxy client class for the Web Service. The proxy client class will be included in the client application. The client application sends all the requests to the proxy client class. The proxy client class formats the request into a SOAP message and forwards it to the Web Service. The proxy class receives the response from the Web Service and sends it to the requesting client application. The following diagram provides a high level view of Web Service consumption by a client application:

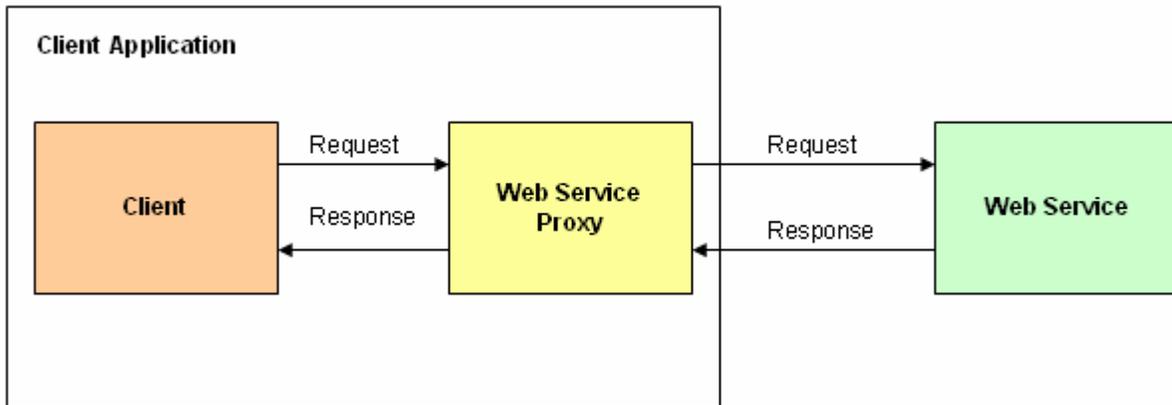


Figure 5.—Web Service Consumption - High Level Diagram.

A dynamic Web Service client performs a lookup to find the Web Service location in a UDDI registry before accessing the service.

Steps involved in consuming a Web Service using a static Web Service, are given below:

1. Get the URL to the Web Service end point. For a list of STORET Web Services end points, refer to the table listed in Section 3.0.
2. Get the proxy client class. The proxy client class can be downloaded from the Web Service home page, as described in Section 3.3. The proxy client can also be generated using the Oracle Web Services Assembler Toolkit and knowing the URL of the WSDL document of the Web service. Refer to Section 6.2 of the Appendix for more information about generating proxy client class using the Oracle Web Services Assembler Toolkit.
3. Add the proxy client class to the consumer/client application classes. Then add jar files required to support the Web Service consumption to the CLASSPATH of the client application. The list of jar files is provided in Section 6.1 of the Appendix.
4. Consume the Web Service using the proxy client class. Consuming the Web Service involves the following steps:
 - Create an instance of the proxy client class.
 - Set a value for the Web Service end point of the proxy client instance, using the method '`_setSoapURL(String)`' provided by the proxy client class.
 - Call the Web Service operation using the methods provided by the proxy client class to send a request to the target Web Service.
 - The proxy client receives the call and packages a SOAP request to be sent to the Web Service. The SOAP request contains the client request enclosed within a SOAP-Envelope.
 - The Web Service receives the request, processes and sends the response back to the proxy client; the proxy client parses the XML received and presents the value to the client application.
5. A portion of sample STORET Web Service consumption code is provided below using the Station Web Service. For a more complete listing of the code, refer to Section 6.3 of the Appendix.

```
String _soapURL = "http://iaspub.epa.gov/webservices/StationService/";

//create an instance of proxy client
StationServiceProxy wsStub = new StationServiceProxy();

//set the URL to the Web Service end point
wsStub._setSoapURL(_soapURL);

//call the Web Service method
try{
    Element stationXML = wsStub.getStations(new Float(30),
        new Float(40), new Float(-90), new Float(-70));
    //process stationXML to extract the required information
}catch(...){
}
```

All the STORET Web Services can be consumed in a similar manner by obtaining the proxy client of the target Web Service(s).

5.0 REFERENCES

- STORET Web Services System Requirements Specification - 50-0031(STORET WS REQ) 2007-0228.doc
- Oracle® Application Server Web Services Developer's Guide, 10g Release 2 (10.1.2), http://download.oracle.com/docs/cd/B14099_11/index.htm
- Oracle® Technology Network, Discussion Forums for OC4J/J2EE, <http://forums.oracle.com/forums/forum.jspa?forumID=46>
- Oracle® XML API Documentation <http://www.oracle.com/technology/tech/xml/xdk/doc/production/java/doc/java/javadoc/index.html>.

APPENDICES

APPENDIX A: Jar Files Included in the CLASSPATH of the Consumer Application

Table A-1 provides a list of Jar files that need to be added to CLASSPATH of the client application, while consuming STORET Web Services:

Table A-1. Jar Files

Component Jar	Description
proxy.jar	The <i>proxy</i> jar file that provides access to the Web Service, if it is not already added to the classes folder of the client application.
\$ORACLE_HOME/lib/xmlparserv2.jar	The Oracle XML parser jar.
\$ORACLE_HOME/j2ee/home/lib/http_client.jar	The Oracle HTTP client jar.
\$ORACLE_HOME/soap/lib/soap.jar	The Oracle SOAP jar.
\$ORACLE_HOME/j2ee/home/lib/mail.jar	Generally, this is available in the Java Runtime Environment (JRE). If this is not available in the JRE, then include it in the CLASSPATH.
\$ORACLE_HOME/j2ee/home/lib/activation.jar	Generally, this is available in the JRE. If this is not available in the JRE, then include it in the CLASSPATH
\$ORACLE_HOME/webservices/lib/wsdl.jar	Required when the client is using a Dynamic Proxy.
\$ORACLE_HOME/webservices/lib/dsv2.jar	Required when the client is using a Dynamic Proxy.

APPENDIX B: Description of Configuration File Parameters Used With WebServicesAssembler

The WebServicesAssembler tool can be invoked to create Web Service client objects based on a WSDL document.

The WebServicesAssembler tool requires a configuration file to run commands. The following is an example configuration that helps generate the client-side proxy jar:

```
<?xml version="1.0"?>
<web-service>
  <proxy-gen>
    <proxy-dir>/TestArea/WatershedSummary</proxy-dir>
    <option name="include-source">true</option>
    <option name="wsdl-location" package-name="myPackage.proxy">
      http://iaspub.epa.gov/webservices/WatershedSummaryService/index.html?WSDL</option>
    </proxy-gen>
  </web-service>
```

The parameters used in the WebServicesAssembler configuration file to generate client-side proxy are listed in Table A-2.

Table A-2. WebServicesAssembler Parameters

Tag	Description
<pre><proxy-dir> directory</proxy-dir></pre>	<p>Specifies the directory for the generated client-side proxy stubs jar file that is included in the generated Web Service .ear file.</p> <p>This tag is required.</p>
<pre><option name="include- source"> value</option></pre>	<p>Setting <i>value</i> to true tells the WebServicesAssembler tool to include the classes and the source in the generated client-side proxy. When the value is false, the source is not included in the generated jar.</p> <p>This tag is optional.</p> <p>Valid values: true, false</p> <p>Default value: false</p>
<pre><option name="wsdl- location"> URL</option> or <option name="wsdl- location" package- name="package"> URL</option></pre>	<p>This tag sets the <i>URL</i> to use for the source WSDL to generate the client-side proxy.</p> <p>This option also supports the optional attribute <i>package-name</i>. The <i>package-name</i> can specify the name <i>package</i> for the generated client-side proxy.</p> <p>This tag is optional.</p> <p>Examples:</p> <pre><option name="wsdl-location"> http://system1:8888/webservice3/TestService?WSDL </option> <option name="wsdl-location" package-name="myPackage.proxy"> http://system1:8888/webservice3/TestService?WSDL </option></pre>

APPENDIX C: Sample Station Web Service Client Program, Using Client-side Proxy

```
import java.io.IOException;
import java.io.StringReader;
import java.io.StringWriter;
import java.util.ArrayList;
import java.util.List;
import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;
import oracle.xml.parser.v2.DOMParser;
import oracle.xml.parser.v2.XMLDocument;
import oracle.xml.parser.v2.XMLNode;
import oracle.xml.parser.v2.XMLParseException;
import oracle.xml.parser.v2.XSLException;
import stubs.WatershedSummaryServiceProxy;

public class StationClient {
    private Log _logger = LogFactory.getLog(this.getClass());
    private String _summaryInfo;
    private String _soapURL = "http://iaspub.epa.gov/webservices/StationService/";

    /**
     * custom constructor
     * @param soapURL
     */
    public StationClient(){
    }

    public List getStations(float minimumLatitude, float maximumLatitude,
                           float minimumLongitude, float maximumLongitude){
        List lstStationInfo = null;
        StationServiceProxy wsStub = new StationServiceProxy();
        wsStub._setSoapURL(_soapURL);

        try{
            Element stationXmlElement = wsStub.getStation(new Float(minimumLatitude),
                                                         new Float(maximumLatitude),
                                                         new Float(minimumLongitude),
                                                         new Float(maximumLongitude));
            XMLDocument xmlDataDoc = null;
            if (stationXmlElement!=null){
                xmlDataDoc = (XMLDocument)
                    stationXmlElement.getOwnerDocument();
            }
            if (xmlDataDoc!=null){
                /****validate xmlDataDoc using the Station Web Service Output
                 schema,
                 if necessary *****/
                lstStationInfo = getStationInfoList(xmlDataDoc);
            }
        }
    }
}
```

```
    }catch(SOAPException soapException){  
        //handle exception  
    }catch(Exception exception){  
        //handle exception  
    }  
    return IstStationInfo;  
}
```

APPENDIX D: Pictorial View of Web Service Output Schema

A pictorial representation of the output schema for **Project Catalog Web Service** is provided in Figure 6 below:

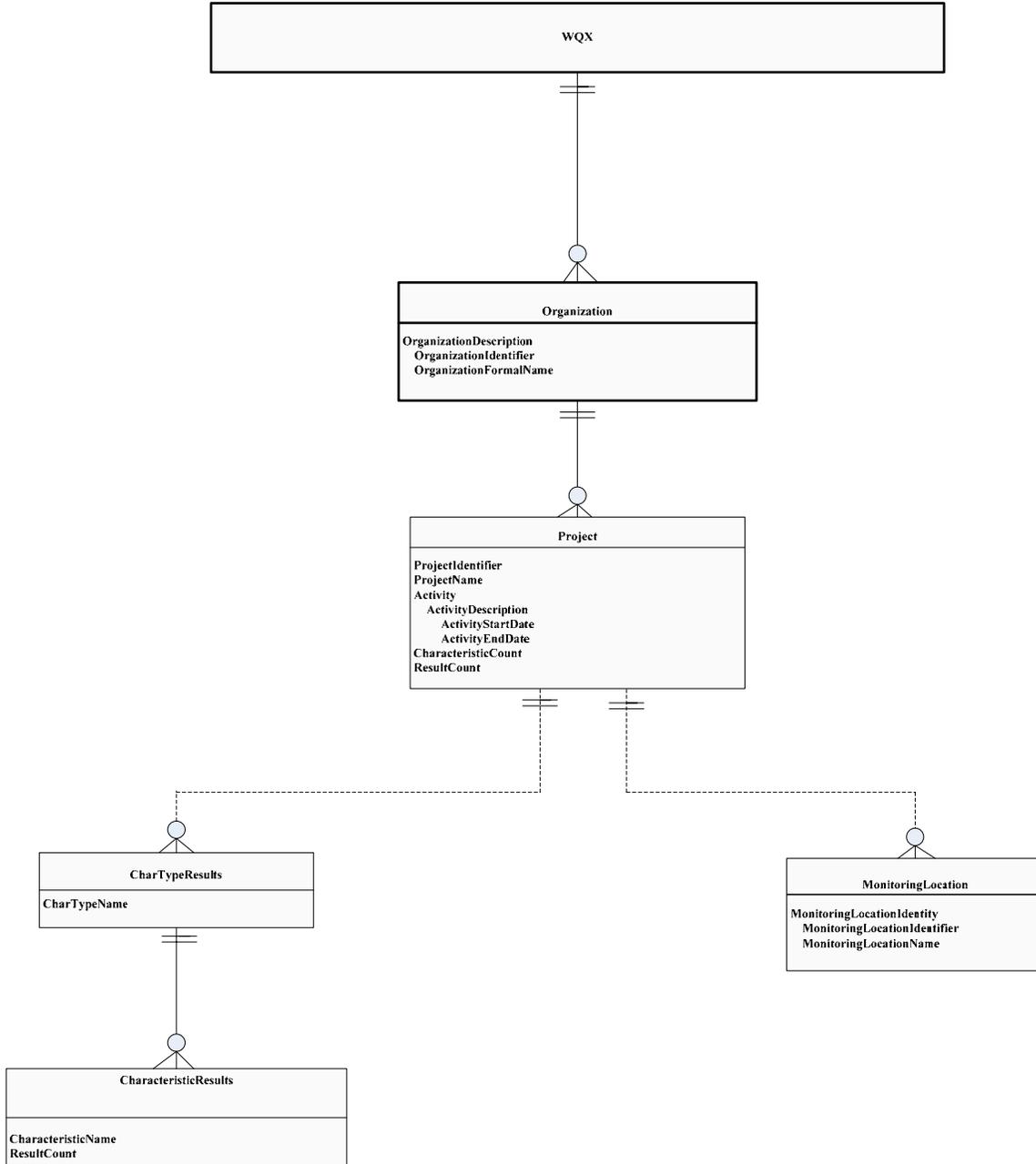


Figure 6. —Project Catalog Web Service Output Schema

A pictorial representation of the output schema for **Station Catalog Web Service** is provided in Figure 7 below:

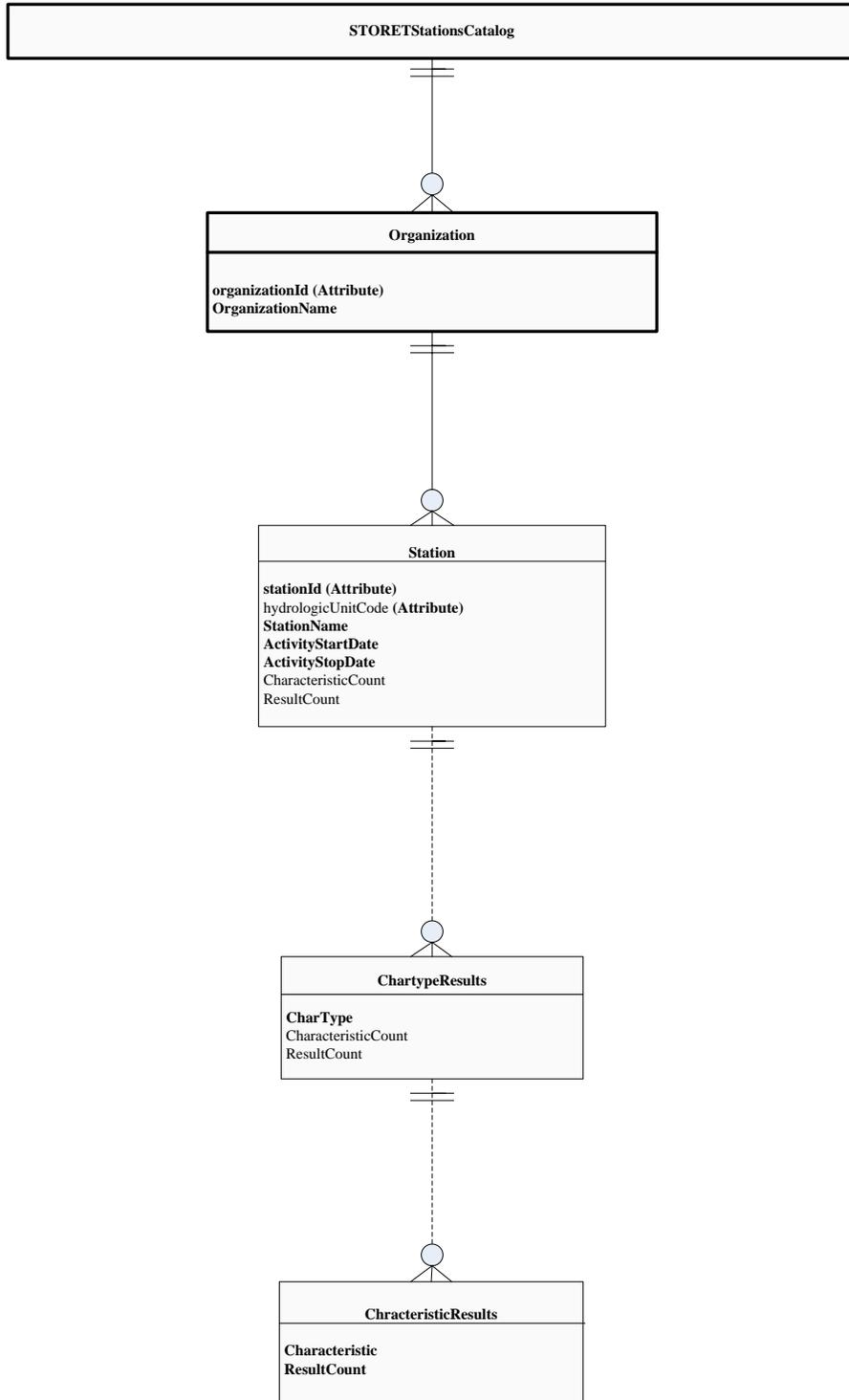


Figure 7. —Station Catalog Web Service Output Schema

A pictorial representation of the output schema for **Station Web Service** is provided in Figure 8 below:



Figure 8. —Station Web Service Output Schema

A pictorial representation of the output schema for STORET **Result Web Service** is provided in Figure 9 below:



Figure 9. —STORET Result Web Service Output Schema

A pictorial representation of the output schema for **Watershed Summary Web Service** is provided in Figure 10 below:

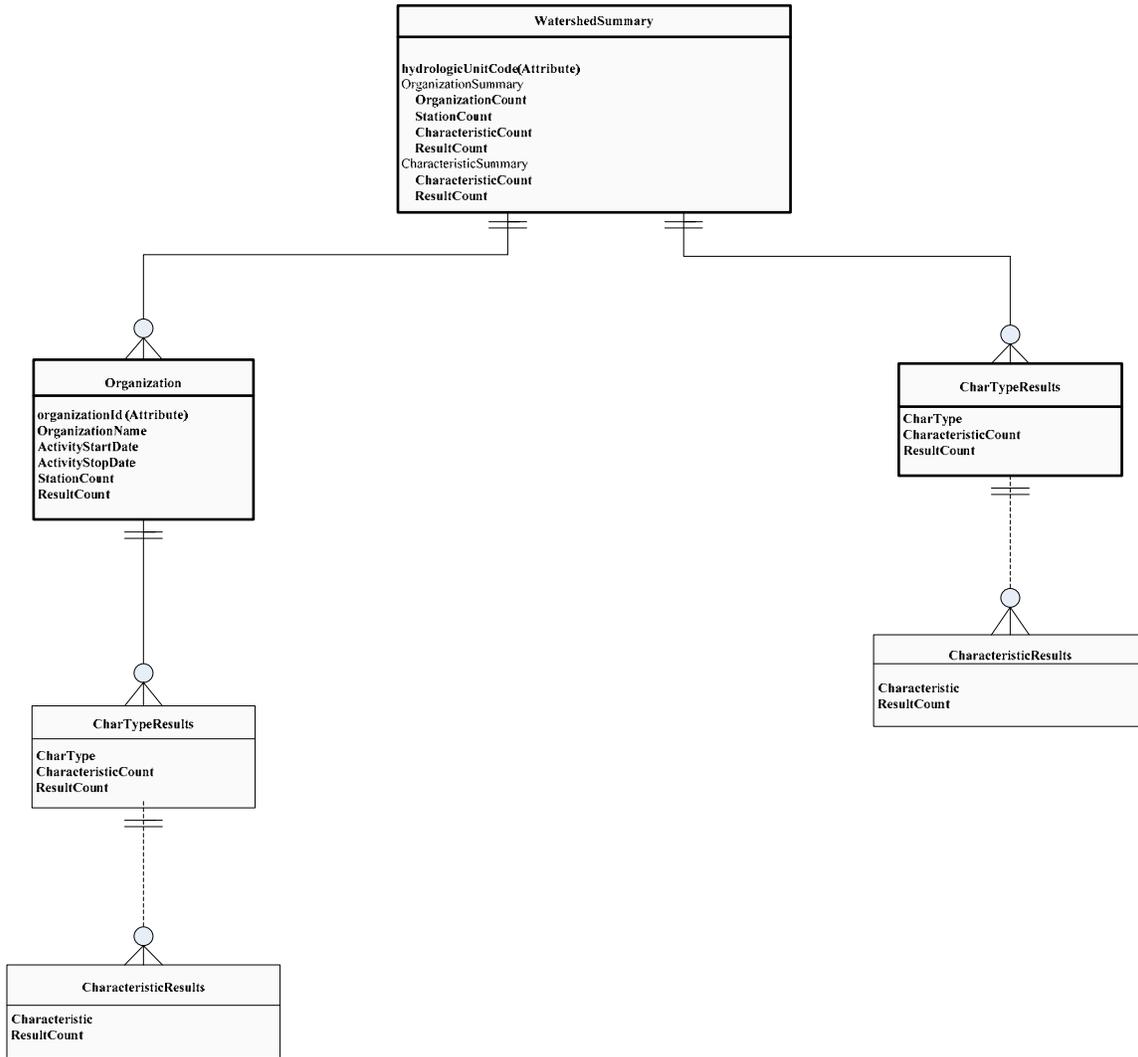


Figure 10. —Watershed Summary Web Service Output Schema

A pictorial representation of the output schema for **Web Service Catalog** method is provided in Figure 11 below:

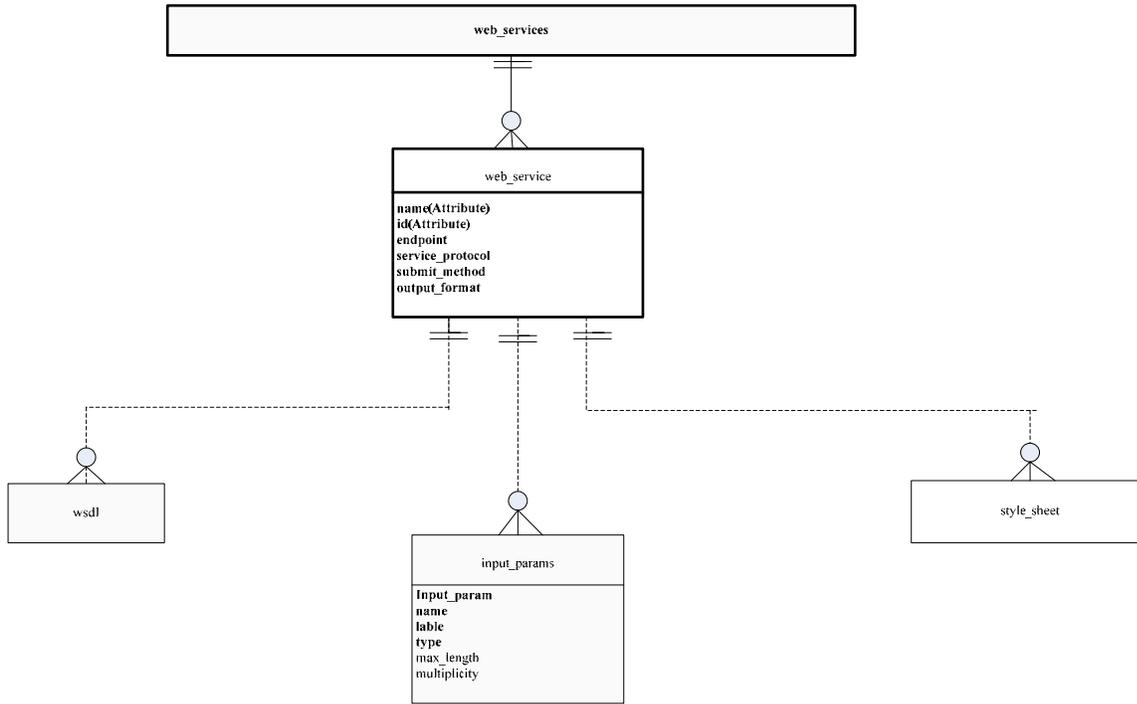


Figure 11. —Web Service Catalog Output Schema