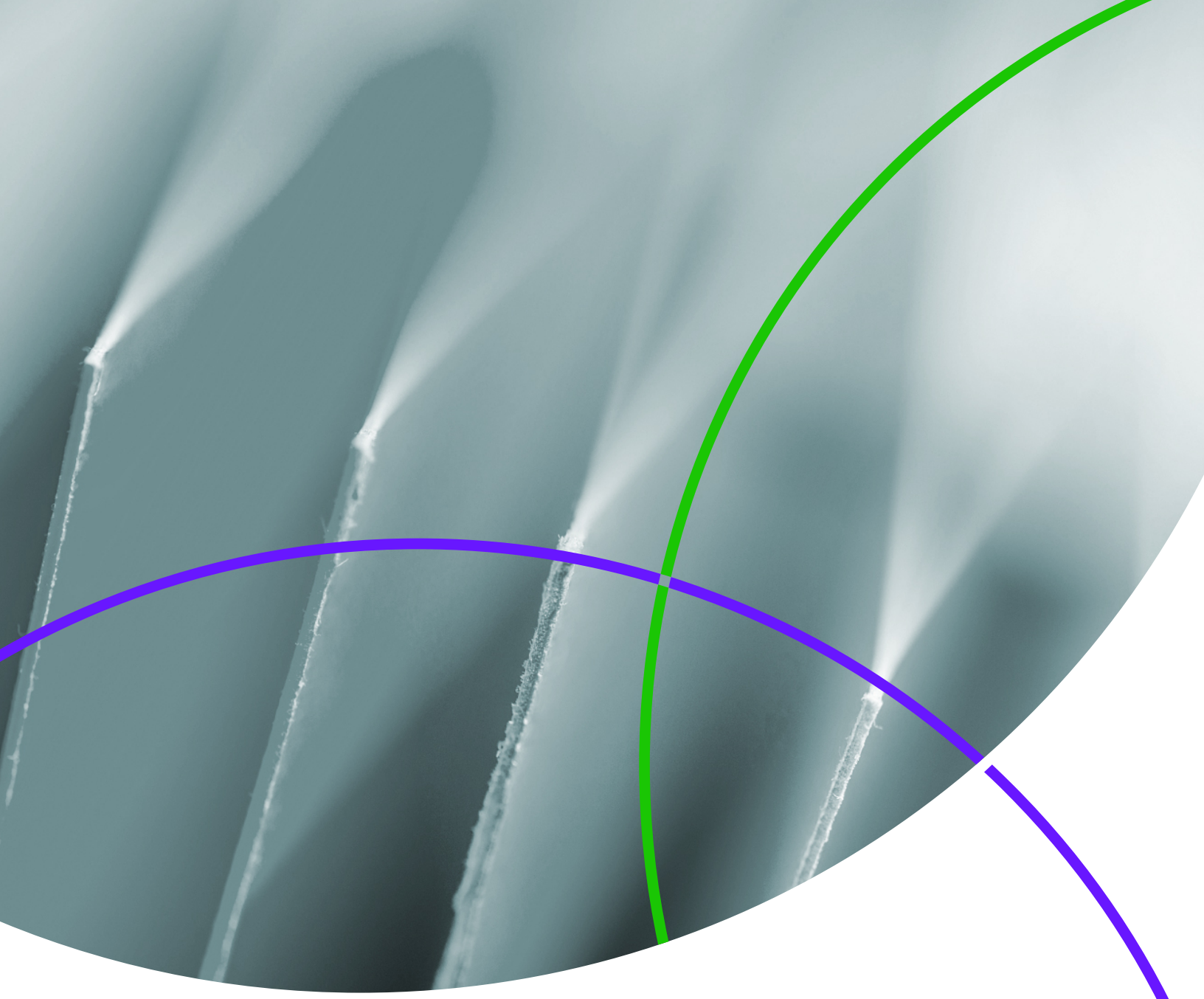

ScholarOne Manuscripts ™

Web Services

Sample Client Guide

27-September-2017

Clarivate Analytics

Table of Contents

INTRODUCTION

Unlike any other system out there, *ScholarOne Manuscripts* includes game-changing, enterprise-level technology designed to enable publishing processes to be lifted to the next level of market competitiveness. *ScholarOne Web Services* is state-of-the-art technology designed to advance your business by helping you get the most out of your own, native systems. Through APIs, it enables easy access to your detailed, real-time manuscript status and author information as often as you need it throughout your business day. Transactional information can now be automatically retrieved and inserted into the flow of your current production process, instantly making your peer review system part of the larger publishing network.

ScholarOne Web Services are built on the same reliable infrastructure our customers have come to trust. You can be sure that the technology standards and systems we use will keep your information safe, secure, and functioning properly. ScholarOne consistently delivers incredible scheduled uptime rates, has built-in safeguards enforced at both the user and infrastructure levels, and secures your data to ensure the privacy of your peer review process.

Why A Web Services Sample Client?

The first version of *ScholarOne Web Services* is an exciting step forward in making it easier for publishers to gain value-added, real-time support of the production process. *ScholarOne Web Services* provides instant, seamless, ready-when-you-are access to *ScholarOne Manuscripts* data. The Web Services are atomic in nature, allowing for integration singly or in combination. As a result, your native support systems and processes can perform straightforward or more complex data integration and process improvement tasks.

ScholarOne has created a **Web Services Sample Client** to showcase the services themselves, and more importantly, to demonstrate how easy it is to access and consume our Web Services. The **Sample Client** and the **Sample Client Guide**, used in conjunction with the **ScholarOne Web Service API Reference Guide**, will help you understand how to access and tailor your Web Service interaction to take full advantage of the services. When used together, these guides provide a comprehensive reference to the messaging details and standard conventions that are relied upon to ensure accurate provision of data between your site and the *ScholarOne Manuscripts* premier journal and peer review tool.

ScholarOne Web Services are RESTful and transfer resources in XML and JSON format. These standards are commonly used throughout the internet and are language-

and platform-agnostic, allowing you to focus on the information provided within the message rather than how the service was implemented.

Tip: The Web Services can and should be tailored to conveniently and efficiently support tasks specific to your own goals.[1]

Key Concepts

All of the following concepts are relevant to interacting with our Web Services and will be reviewed in detail in this guide.

1. The Sample Client has been developed to invoke *ScholarOne Web Services* manually via the Sample Client's command line interface. See Table 4 – Sample Client – Options & Call Parameters for all of the Sample Client's command line options and call parameters. The command line options offer flexibility and extensibility to the message request code samples provided in this Guide.

2. The Web Services can be scripted to run one after another in any order and integrated into your existing business logic.

3. All of the Web Services can be used to retrieve a single object (manuscript revision, submission, etc.) at a time.

4. Most of the Version 1.0 Web Services work in "list mode" to support the retrieval of more than one object (manuscript revision, submission, etc.) at a time. Later when they are made available, Web Services for create, update and delete actions will operate on individual records at a time.

[1] The Sample Client provided is a Java version of the client; however, the users of this service are not required to use the provided client. Your client can be coded in any language and communicates with the service using the HTTP protocol.

5. The consumer of these Web Services must first integrate necessary logic to establish an authenticated, secure connection, and provide required information to open up authorized channels to your requested information.

6. The Sample Client permits access to 3 of the four Version 1.0 GET Web Services; however, in production you will only be able to access the Web Services applicable to your organization's provisioned service: lite (complimentary) or premium (fee). Lite provisioning provides access to the "basic detail" service calls whereas premium provisioning provides access to the "full detail" Web Service calls in addition to the "basic detail" calls. The Sample Client provides an example of being blocked due to provisioning.

7. Since Web Services provide an abstraction layer on top of your site's data in the ScholarOne Manuscripts system, direct access to your data is not permitted for you or for anyone else attempting to retrieve sensitive, site-specific information. This means that the sub-domain URL and product designation used when calling our Web Services, coupled with versioning information, your site's short name, your username and password API key, the method you are accessing, and audited callIDs associated with each individual request provide multiple layers of data security.

8. As with the requirement for proper authentication, success of Web Service integration will rely on handling all possible success and failure scenarios so that risks to your local data stores and business processes and to ScholarOne Web Services are mitigated.

Sample Client Audience and Goals

This guide is primarily intended for the technical audience: the system architect or developer. It is a go-to technical resource, assisting in understanding the process and

technical details involved in leveraging the available Services in an efficient, flexible and extensible manner.

- This guide will walk you through the process of successfully unpackaging and installing the Sample Client.
- It will first focus on single-item request call processing and associated basic SUCCESS and FAILURE response messaging.
- It will assist in demonstrating how to modify a subset of methods to retrieve lists of items.
- It will build on learned Sample Client concepts to examine the complex function of exception call response processing.
- It will review errors and error response handling in detail.

Included in this guide are diagrams that illustrate key processing that occurs each time a Web Service is called. This is essential information to aid in identifying and responding to the programmatic scenarios that may occur while interacting with the Web Service API methods:

- **Figure 1 - Web Service Call Flow Processing** presents the basic Web Services call flow from connecting to the server, through authentication, and on to retrieving data. At a high level it also indicates the types of responses you can expect for various success and failure conditions.
- **Figure 2 - Response Processing Logical View** provides a detailed view of response processing and handling techniques. This diagram provides valuable insight into why server communication may be failing as well as what might be preventing authentication or authorized access to the Web Services.

- **Table 2 – ScholarOne Response Code Explanations** provides substantive detail on the call responses that will require client-side methods be developed for proper handling and subsequent client-server interaction.
- **Table 3 - Suggested Strategies for Client-Side Response Handling** expands upon the brief response strategy suggestions of Figure 2. Table 3 provides guidance on how to handle both success and failure responses.

In order to take full advantage of the *ScholarOne Web Services*, it will be necessary to implement appropriate processes for gracefully handling all of the possible error conditions likely to occur during interaction with the server.

Tip:	The Response Processing Strategy section of **Figure 2** guides you through the methods that must be developed by you to properly handle ScholarOne Web Service error codes.

After you have read through this guide and exercised the examples in the Sample Client, you will be familiar with:

- The *ScholarOne Web Services* Call Process Flow
- How to discover, understand and access *ScholarOne Web Services*
- Important concepts for interacting with *ScholarOne Web Services*
- Key best practices for calling *ScholarOne Web Services*
- Handling successful and partially successful Web Service responses
- Handling failure responses based on returned error codes
- Implementing suggested error response processing strategies

THE SCHOLARONE WEB SERVICES SAMPLE CLIENT

The *ScholarOne Web Services* client is a Java-based command line client. The Sample Client will support executing Web Service requests one at the time via the command line. To run the Client, you must have access to the Java Runtime Environment with the system path variable to the java executable defined correctly. On Windows, PATH is the system variable that your operating system uses to locate the needed executable from the command line.

The command line prompt can accept an input file and will respect other command line parameters and options to further refine the request or desired response. See **Table 4 – Sample Client – Options & Call Parameters** for a complete listing.

Installation Requirements

- Installed JRE or Java JDK (preferably 1.6 or higher)

To verify your Java version on Windows 7 do one of the following:

- Click Start Button >> Type “cmd” in the search box >> at the command prompt, type “java –version”
- Click Start Button >> go to the Control Panel >> find the “Programs” icon and click on it >> find the “Java” icon and click on it
- Development IDE (preferably Eclipse)[2]

This document assumes familiarity with the Java development environment and Java general product setup.

[2] The Sample Client can be installed and exercised without an IDE installed.

Sample Client Package Contents

The Sample Client for *ScholarOne Web Services* is a zip file package, S1-WebServices-SampleClient-V1.0.0.zip that includes:

- SampleClient.jar – command-line executable JAR file used to run samples
- SampleClient.zip – zipped Java project including source code for Sample Client
- config.xml – configuration file for Sample Client
- A set of additional XML files containing Sample Input parameters for use with the Sample Client. These will be used in the Web Services Call Examples section of this document.

Note: The source code files for the sample client can be used for implementation examples of Web Service features. They show the correct sequence of calls and parameter data types to provide a generalized method for API use that developers can modify for their specific needs.

Sample Client Features

- Java based Sample Client
- Ability to securely access the ScholarOne Sample Client environment
- Ability to call ScholarOne Web Service API Methods from a Command-Line and review the response
- Sample XML datasets demonstrating API request and response structures
- Working Java project which can be imported and used in an Integrated Development Environment (IDE) that supports Java development

Terminology

1. **Document Id** – this is a unique identifier for each Manuscript Revision in the *ScholarOne Manuscripts* system. Internally to ScholarOne, this is a numeric value.

2. **Submission Id** – this is also known as the Manuscript Number or Document Number. This is the identifier that is seen in the ScholarOne UI for each Manuscript.

3. **Document No/Document Number** – this is synonymous to Submission Id

For each manuscript, there may be one or more Document IDs associated to it. Each Document ID represents a specific version of the Manuscript.

The Submission ID is equivalent to the displayed Manuscript ID in the *ScholarOne Manuscripts* application. Manuscript ID is a number given to a manuscript upon submission. It is generated based on a configuration-defined formula that may produce duplicates within a particular site. This number may also have been changed manually in the *ScholarOne Manuscripts* system. The Document ID -- a system-generated unique identifier given to each version of a manuscript -- is unique and is not editable.

WEB SERVICE ENVIRONMENTS

Each application environment provides access to Web Services appropriate to the environment. For example, our Implementation (IMPL) environment will provide the Web Services specifically intended for use with the Sample Client. Further, this environment will host Web Services that have been modified to enable end-to-end Request / Response analysis that cannot readily be performed against the Production environment. For example, the IMPL environment will expose API methods that will assist your development effort by simulating various server-side exceptions such as scheduled and unscheduled outages.

So that you can confidently validate the data returned in a successful response, the IMPL environment will host a recent, static, generalized copy of data that will be accessible in the Production environment. The IMPL environment and data is separate and distinct from your Production (PROD) environment. A handful of sample documents

are available in the Sample Client environment. Full Manuscript and full Author data is available for all of these documents.

Each Environment will have a new sub-domain where *ScholarOne Web Services* can be accessed.

To access Web Services for your Journal(s)/Site(s), you simply need to add "-api" to the sub-domain you normally use to access the ScholarOne Web UI.

Note: Instead of including your site's short name on the end of the URL as is done when accessing the ScholarOne Web UI, when accessing the Web Services your Journal/Site short name will be passed as a parameter. Examples included with the Sample Client will clarify this convention.

SERVICE ENDPOINT ADDRESSES

The public endpoint root addresses, the API entry point, for *ScholarOne Web Services* are:

- Production (PROD)
  - Web UI: http://mc.manuscriptcentral.com/<yoursiteshortname(s)>
  - Web Services: https://mc-**api**.manuscriptcentral.com/ (bolded for emphasis only)
- Implementation (IMPL)
  - Web UI: http://mc-impl.manuscriptcentral.com/<yoursiteshortname(s)>
  - Web Services: https://mc-impl-**api**.manuscriptcentral.com/

The resource location or path segment is specified relative to the API entry point as a combination of:

- <context> api/s1m
- <version> /v1
- <resource> /submissions/full/metadata/documentids

Examples of a fully-qualified destination URL in a sample Web Service call:

https://mc-api.manuscriptcentral.com/api/s1m/v1/submissions/full/metadata/documentids?ids='88083'&site_name=<siteshortname>&locale_id=1

or

https://mc-api.manuscriptcentral.com/ api/s1m/v1/submissions/basic/contributors/authors/submissionids?ids='WEB-2013-006','WEB-2013-002'&site_name=<siteshortname>

WADL File Locations

ScholarOne web application descriptions (WADL) can be accessed on the Production environment that corresponds to where your Journals (Sites) exist. WADL specifications define a communication contract between complex enterprise systems, defining the rules clearly. The WADL specification documents the RESTful interface.

The WADL specific to your site can be found at:

https://<your site's subdomain>-api.manuscriptcentral.com/api/s1m/v1?_wadl

An example of the ScholarOne Web Services WADL may be provided to you by contacting ScholarOne Support.

SCHOLARONE VERSION 1.0 WEB SERVICES OVERVIEW

The first release of *ScholarOne Web Services* will contain read-only services. They will provide immediate status of a specific manuscript or information about a manuscript's author(s) based on the type and format of the API request.

For the purpose of integrating real-time data into the transactional processing flow for your publishing process and/or system, ScholarOne has exposed four key data retrieval methods:

- getSubmissionInfoBasic
- getSubmissionInfoFull
- getAuthorInfoBasic
- getAuthorInfoFull

Each method can be called using either a Document ID or a Submission ID (Manuscript ID), or a list of Document IDs or Submission IDs as input and provide data about Manuscripts in response.

- All methods require **either** Submission ID **or** Document ID
- All methods can accept list input of up to 25 comma-separated Submission IDs or Document IDs
- All methods can output data in JSON format
- Calls using Submission ID (Document Number) are case sensitive

If your organization has Premium Web Services provisioned, you will have access to the Basic and to the Full Web Services in production. Otherwise, access is authorized for the two Basic methods only.

Note: While the Sample Client will return data from the getAuthorInfoFull API call, an example is provided that simulates a production authorization failure by requesting data using getSubmissionInfoFull.

See Table 1 – Version 1.0 Web Services Summary for the information provided for each API call. Please see the companion *ScholarOne Web Services API Reference Guide* for detailed API information specifically geared toward the developer community.

API REQUEST

Request Message Contents

The following information is **required** when calling *ScholarOne Web Services*:

Field	Description
User Name	Profile User Name – not a *ScholarOne Manuscripts* user. Also called the "system" or "application" user name.
Password	API Key[3]
URL	The Web Service End Point Address, which identifies the Service, plus the API entry point and path to the desired resource or resource collection.
Site Short Name	The configured alias for your *ScholarOne Manuscripts* web UI destination.

The following information is **optional** when calling *ScholarOne Web Services*:

Field	Description
Locale ID	The LOCALE_ID is the unique identifier for a specific language. Options are

[3] API Key and UserName are part of the HTTP Header (HTTP Digest authentication handshake); The API Key is your password and is obtained through your ScholarOne Relationship Manager.

Field	Description
	• 1 (United States English) • 2 (Simplified Chinese, Pinyin ordering) • 3 (French)
External ID	An ID value that can be set by the client for call tracking and audit purposes.
_Type[4]	The Data type requested from the Service, XML or JSON. Default is XML.

Basic Request Message Structure

```
<inputData>
  <ids>'xxx',’xxx’,…</ids>
  <site_name>shortname</site_name>
  <locale_id>1,2, or 3</locale_id>
  <url><api entry point><context><version><resource></url>
  <external_id>short_string</external_id>
  <username>UserName</username>
  <password>XXXX</password>
</inputData>
```

Sample Request

```
<inputData>
  <ids>'80831',’80832’</ids>
  <site_name>web_svs</site_name>
```

[4] The _type query parameter is represented by the <format> command line option of the Sample Client

```
  <locale_id>1</locale_id>
  <url>v1/submissions/full/metadata/submissionids</url>[5]
  <external_id>23412</external_id>
  <username>sample_user</username>
  <password>SRU4DQ5WOJ2PX8CA</password>
</inputData>
```

API RESPONSE

Response Message Contents

The following information will be returned as a result of calling *ScholarOne Web Services*:

Field	Description
Profile Call ID	If provided in the request, this is the value of the caller-provided external_id.
Status	The request outcome from the ScholarOne Web Services perspective[6].
Call ID	This is the unique identifier for a given Web Service call.

Provisionally, the following information will be provided in an error response:

Field	Description

[5] The Sample Client's config.xml file points to the base URL, e.g. <baseURL>http://IMPL-api.manuscriptcentral.com/api/s1m/</baseURL> which gets coupled with the appropriate method's endpoint address which in the case of the Sample Client is referenced in example XML files. An example is shown in the Sample Request above.

[6] Version 1.0 *ScholarOne Web Services* response status can be "SUCCESS", "FAILURE", or "MAINTENANCE" and are explained in detail in **Table 2 – ScholarOne Response Code Explanations**.

Field	Description
Error Code	*ScholarOne Web Services* code returned to the requesting client in the case of an exception response or fault.
More Info	Human-readable, *ScholarOne Web Services* request call-specific error description.
User Message	Human-readable, *ScholarOne Web Services* error-specific error description.
Call Back Time	In UTC (Coordinated Universal Time)[7], the date/time to try calling again. Will present only in server maintenance situations.
Status	In a failure response message, Status will be "FAILURE" or "MAINTENANCE"

A successful GET request will retrieve information of interest per the details of the request message itself. See the **ScholarOne Web Services API Reference Guide** for successful response details.

Particularities of successful response messages:

- If there is a null value for an XML field in the output, the XML element will not be listed.
- If there was a value for an XML field and the value was removed effectively leaving it 'blank', an empty XML element will appear.
- To correlate each ID in the request data set to an associated response element, an inputIndex will be provided. inputIndex will be discussed below.

[7] Coordinated Universal Time is based on the 24-hour UTC time scale with a special UTC designator "Z", ex. format YYYY-MM-DDThh:mm:ssZ e.g. 2013-11-08T17:57:25Z

- All Web Services calls which use Submission ID will return a "success" message without details if a manuscript is in following states:
    - Draft (never submitted)
    - Un-submitted (submitted then un-submitted)

inputIndex

List data is not necessarily returned in the order requested. Duplicate but valid input IDs will yield a successful result for the second, duplicated ID only. The inputIndex value provides a way for you to consume the response payload in a way that reassembles call request IDs against those of the response despite nuances or imperfections in the input or output data. It will be necessary for the Web Services consuming endpoint to correlate successfully retrieved data to the list of data requested.

Note: The **inputIndex** is an important concept to understand when implementing API methods that request lists of data.

A simplified inputIndex example is provided to aid in general understanding:

Input Document IDs: 1, 14, 5, 157 (this is not a valid Document Id), 30

	Input Index 1	**Input Index 2**	**Input Index 3**	**Input Index 4**	**Input Index 5**
Document Id	1	14	5	157	30
Appears In Response Data	Yes	Yes	Yes	No	Yes
inputIndex Value	0	1	2		4

Basic Error Response Message Structure

```
<Response>
  <errorDetails>
     <callBackTime>UTC      Time      in      YYYY-MM-DDThh:mm:ssZ
format</callBackTime>
     <errorCode>Server, not HTTP, response code</errorCode>
     <moreInfo>string</moreInfo>
     <userMessage>string</userMessage>
  </errorDetails>
  <status>FAILURE, or MAINTENANCE</status>
  <callID>string</callId>
</Response>
```

Sample Error Response

```
<Response>
  <errorDetails>
     <callBackTime>2013-11-18T16:45:24Z</callBackTime>
     <errorCode>500</errorCode>
     <moreInfo/>
     <userMessage>Throttle limit exceeded</userMessage>
  </errorDetails>
  <status>MAINTENANCE</status>
  <callID>8e49900d-0c69-4a7b-8e9c-7356bf0c7d93</callId>
</Response>
```

Sample Success Response

```
<Response>
     <profileCallId>22222</profileCallId>
  <result                                    xsi:type="submission"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```
     <authorFullName>Collins, Mahally Q.</authorFullName>
     <authorPersonId>686739</authorPersonId>
     <documentId>88026</documentId>
     <inputIndex>0</inputIndex>
     <submissionDate>2013-10-18T20:56:15Z</submissionDate>
     <submissionId>WEB-2013-0002</submissionId>
     <submissionStatus>
          <task>
               <statusName>Complete Checklist</statusName>
               <taskId>682726</taskId>
               <taskName>Admin Checklist</taskName>
               <taskStatus>PENDING</taskStatus
          </task>
     </submissionStatus>
     <submissionTitle>The Use of Computers in Elementary
Classrooms</submissionTitle>
     <submissionType>Original Article</submissionType>
  </result>
  <status>SUCCESS</status>
  <callId>744ea98c-7d16-411c-8007-0689b60d0fd5</callId>
</Response>
```

JSON Output

Here is the same success response in JSON formatted output:

```
{"Response":{"profileCallId":22222,"result":{"authorFullName":"C
ollins,                                                    Mahally
Q.","authorPersonId":686739,"documentId":88026,"inputIndex":0,"s
ubmissionDate":"2013-10-18T20:56:15Z","submissionId":"WEB-2013-
0002","submissionStatus":{"documentStatusId":3,"documentStatusNa
me":"Submitted","inDraftFlag":false,"task":{"taskId":682726,"tas
kName":"Admin
Checklist","taskStatus":"PENDING","taskStatusName":"Complete
Checklist"}},"submissionTitle":"The   Use   of   Computers   in
Elementary                Classrooms","submissionType":"Original
```

```
Article"},"status":"SUCCESS","callId":"4ff861da-e784-4424-910e-
f620c66ec8d1"}}
```


SAMPLE CLIENT WEB SERVICES BASICS

This section of the Sample Client Guide will introduce you to the basic API methods and best practices to using the Web Service API. This section of the guide is intended to prepare the caller technically and intellectually to take full advantage of the Sample Client Guide code examples. The series of examples provided will convey key concepts about the ScholarOne API methods and how they function. They highlight the recommended way to consume *ScholarOne Web Services* and provide a starting point in the form of sample code for those developing their own client.

Some concepts will apply to all calls made to *ScholarOne Web Services* while others will demonstrate specific capabilities which apply to specific types of calls.

A Web Service client is required to send and receive messages programmatically. As illustrated in the examples, records can also be retrieved one at a time or in list form from the command line prompt. Please refer to **Table 4 – Sample Client – Options & Call Parameters** for a complete listing of command line options and associated parameters supported in the *ScholarOne Web Services* client.

Learning by Example

Each of the Sample Client examples can be executed by opening a console window, navigating to the directory where you unzipped the Sample Client and executing the supplied Command Lines.

On Windows:

- Click Start Button
- Click “Run…” Menu Item
- In the Run Dialog Box, type in “cmd” and press the <ENTER> key or Click “OK”
- Change Directory to the directory where you unzipped the Sample Client by typing “cd <directory pathname>

- Copy and Paste the Command Line for the example and press <ENTER>

or

- Hold Shift + Mouse right click on the Sample Client directory on your machine
- Select “Open Command window here”

Sample Client Command Line Help

The Sample Client can accept several command line parameters. To review these parameters, use the following command line from the directory where you unzipped the Sample Client:

```
java -jar SampleClient.jar -h
```

KEY EXAMPLES

The Sample Client will present examples of all possible scenarios encountered in the Web Services call flow depicted in **Figure 1 - Web Service Call Flow Processing**. In addition, the Sample Client includes coding samples to demonstrate some key techniques to be used when interacting with the Services, including the management of valid output data as well as the error scenarios depicted in **Figure 2 - Response Processing Logical View**.

When interacting with *ScholarOne Web Services*, server responses will fall into 3 main HTTP response categories:

HTTP Code Category	Description
200 Successful	This class of status code indicates that the client's request was successfully received, understood, and accepted.
400 Client Error	This class of status code is intended for cases in which the client seems to have erred.
500 Server	Response status codes beginning with the digit "5" indicate cases in

HTTP Code Category	Description
Error	which the server is aware that it has erred or is incapable of performing the request.

Within each of these categories, the service will return a specific application service-level error to aid the caller in proper response handling. The Guide will often refer to these types of error codes as S1 (ScholarOne) Error Codes, or S1 Code.

Figure 2 - Response Processing Logical View presents ScholarOne Error Codes with user messages that fall into the 3 main HTTP response categories. More information regarding each error condition is provided in Table 2 – ScholarOne Response Code Explanations, cross-referenced by the letter shown on the Logical View Diagram in the "ScholarOne Error Code" section. As we review Sample Client examples, response handling techniques are provided to serve as a sample implementation that can be used or modified to match your needs. These techniques are shown on the Logical View Diagram using the numbering convention "R1" – "R7" in the "Response Processing Strategy" section of the diagram and are cross-referenced by this number to the suggested response handling techniques in **Table 3 - Suggested Strategies for Client-Side Response Handling**.

To assist you with response handler development, we have provided examples that demonstrate key techniques to be used when interacting with the S1 Web Services. Use your IDE (preferably Eclipse which is an open source community) to view the sample code.

HTTP 200 – Level: Successful Response

Successful responses fall into two subcategories: Success and Partial Success. Successful responses return all of the information requested. Partially successful responses – not included in the Version 1.0 API – will return some of the information requested, but not all of it. In the future, a Partial Success response can be expected when the server rate limits output to manage server load and distribute resource bandwidth fairly to all callers while ensuring a robust response.

Sample Basic Request: getSubmissionInfoBasicDocumentId

DESCRIPTION	This sample shows a successful call to our Basic Manuscript Web Service using one Document ID.
Notes	The response objects include a special output value called “inputIndex” which allows the caller to correlate the response objects to input parameters. See inputIndex.
Request Command Line	java -jar SampleClient.jar --input oneDocumentIdForBasic.xml --output oneDocumentIdForBasic_output.xml
Handler Command Line	java -jar SampleClient.jar -ex consumeResponseSingleItem
Table 2 - ScholarOne Error Code XREF	none
Table 3 – Suggested response handling strategies XREF	R1

Sample Full Request: getSubmissionInfoFullDocumentId

DESCRIPTION	This sample shows a successful call to our Full Manuscript Web Service using multiple Document IDs and receiving a list of Manuscript Revisions matching the Document IDs.
Notes	The response objects include a special output value called “inputIndex” which allows the caller to correlate the response objects to input parameters. See inputIndex.
Request Command Line	java -jar SampleClient.jar --input multipleDocumentIdsAllValidForFull.xml --output multipleDocumentIdsAllValidForFull _output.xml
Handler Command Line	java -jar SampleClient.jar -ex consumeResponseOnePerInput
Table 2 - ScholarOne Error Code XREF	none
Table 3 – Suggested response handling strategies XREF	R1

Sample Basic Request: getAuthorBasicSubmissionId

DESCRIPTION	This sample shows a successful call to our Basic Manuscript Author Web Service using one Document Number (aka Submission Id).
Notes	You may receive multiple Authors for the Document Number requested. Pay careful attention to the formatted XML output. As with requests for Submission metadata, requests for Author information also return an inputIndex value.
Request Command Line	java -jar SampleClient.jar --input oneSubmissionIdForAuthorFull.xml --output oneSubmissionIdForAuthorFull_output.xml
Handler Command Line	none
Table 2 - ScholarOne Error Code XREF	none
Table 3 – Suggested response handling strategies XREF	R1

Sample Full Request: getAuthorFullSubmissionId

DESCRIPTION	This sample shows a successful call to our Full Manuscript Author Web Service using multiple Document Numbers (aka Submission IDs) and receiving a list of Manuscript Revisions matching the Document Numbers (aka Submission IDs).
Notes	The response objects include a special output value called “inputIndex” which allows the caller to correlate the response objects to input parameters. See inputIndex.
Very important note	Examine the _output file carefully after executing this example. In some valid cases, multiple results are returned for the *same* Submission ID. For example, Submission ID='WEB-2013-0005' in the Request results in three Authors returned in the Response. Since 'WEB-2013-0005' is the fourth Submission ID listed in the Request, all Authors' data associated to this Submission ID will have an inputIndex of 3 (inputIndex is zero-based); the XML Response will have three authors whose inputIndex is 3.
Request Command Line	java -jar SampleClient.jar --input multipleSubmissionIdsAllValidForAuthorFull.xml --output multipleSubmissionIdsAllValidForAuthorFull_output.xml
Handler Command Line	-ex consumeResponseMultiplePerInput
Table 2 - ScholarOne Error Code XREF	none
Table 3 – Suggested response handling strategies XREF	R1

Sample Basic Request: getAuthorBasicDocumentId (with valid, duplicate input IDs)

DESCRIPTION	This sample shows a call to our Basic Manuscript Author Web Service using multiple Document IDs with some duplicate input Document IDs. This demonstrates how our Web Services will respond to duplicated input Document Id values.
Notes	The response objects include a special output value called "inputIndex" which allows the caller to correlate the response objects to input parameters. See inputIndex.
Request Command Line	java -jar SampleClient.jar --input multipleDocumentIdsDuplicateInputsForBasic.xml --output multipleDocumentIdsDuplicateInputsForBasic_output.xml
Handler Command Line	java -jar SampleClient.jar -ex consumeResponseDuplicateInputs
Table 2 - ScholarOne Error Code XREF	none
Table 3 – Suggested response handling strategies XREF	R1

Sample Basic Request: getAuthorBasicDocumentId (with valid and invalid input IDs)

DESCRIPTION	This sample shows a successful call to our Basic Manuscript Author Web Service using multiple Document IDs with some valid and some invalid input Document IDs.
Notes	All valid input Document IDs will get a response. Invalid input Document IDs will not get a response. The "inputIndex" can be used to correlate which input values received a response. See inputIndex.
Request Command Line	java -jar SampleClient.jar --input multipleDocumentIdsSomeInvalidForBasic.xml --output multipleDocumentIdsSomeInvalidForBasic_output.xml
Handler Command Line	java -jar SampleClient.jar -ex consumeResponseInvalidInputs
Table 2 - ScholarOne Error Code XREF	None
Table 3 – Suggested response handling strategies XREF	R1

Sample Basic Request: getSubmissionBasicDocumentId (valid site but wrong document ID)

DESCRIPTION	This sample shows a special case of calling ScholarOne Web Services. The caller has provided an invalid Document Id for a Journal/Site. The Document Id may (or may not) be valid on another Journal/Site. It is not valid for the Journal/Site that was called.
Notes	The caller must provide the correct Journal/Site (parameter – site_name) when calling ScholarOne Web Services.
Request Command Line	java -jar SampleClient.jar --input validSiteWrongDocumentId.xml --output validSiteWrongDocumentId_output.xml
Handler Command Line	java -jar SampleClient.jar -ex consumeResponseZeroItem
Table 2 - ScholarOne Error Code XREF	none
Table 3 – Suggested response handling strategies XREF	R2

Error Handling

A major element of Web Services is planning for when things go wrong. Clients connecting to *ScholarOne Web Services* must be ready to handle a variety of API-level errors. Moreover, errors can originate on either the server side or on the client side. ScholarOne Web Service error codes are numeric and are included in the response message in a section called “errorDetails”. To facilitate error human-readability, we include a “userMessage” and a “moreInfo” section in the error response. While each error may have a “userMessage” and “moreInfo” description (optional), be aware that Web Service clients which interact with *ScholarOne Web Services* **must** use only returned error codes for error identification and handling. Error descriptions can change at any time, they are provided only for debugging and logging purposes.

Your code must be resilient and be able to handle our maintenance windows, system-wide and Profile-level throttling, validation errors, issues with authentication and authorization, etc.

- HTTP 400-Level Error Codes indicate that there is an issue with the calling message that originates at the client.

- HTTP 500-Level Error Codes let the client know there is a problem outside of their control; when possible we try to include information about whether the client should retry and when. Having a defined protocol for retries helps avoid the situation where a system comes back up only to fall over again with all the traffic from people retrying every minute (or other interval) - this is a real concern for systems that are under heavy load.

Examples provided in the following two sections are intended to reinforce the need to support proper handling of FAILURE response messages. Suggested strategies for properly handling or otherwise reattempting failed requests are presented in **Figure 2 - Response Processing Logical Vie**w and are expanded upon in **Table 3 - Suggested Strategies for Client-Side Response Handling**.

HTTP 400 – Level: Client Failure Response

Sample Basic Request: getSubmissionInfoBasicDocumentId (valid credentials, wrong site)

DESCRIPTION	This sample shows a problematic call to ScholarOne Web Services. The caller has tried to access a Journal/Site which they have not been authorized to access.
Notes	HTTP Response Code: 401
Request Command Line	java -jar SampleClient.jar --input siteAccessAuthorizationFailure.xml --output siteAccessAuthorizationFailure_output.xml
Handler Command Line	none
Table 2 - ScholarOne Error Code XREF	405
Table 3 – Suggested response handling strategies XREF	R5

Sample Full Request: getSubmissionInfoFullSubmissionId (profile not provisioned for Web Service)

DESCRIPTION	This sample shows a problematic call to *ScholarOne Web Services*. The caller has tried to access a Web Service which they have not been authorized to call.
Notes	HTTP Response Code: 400
Request Command Line	java -jar SampleClient.jar --input apiMethodAuthorizationFailure.xml --output apiMethodAuthorizationFailure_output.xml
Handler Command Line	None
Table 2 - ScholarOne Error Code XREF	407
Table 3 – Suggested response handling strategies XREF	R5

Sample Basic Request: getSubmissionInfoBasicDocumentId (throttled)

DESCRIPTION	This sample shows a problematic call to *ScholarOne Web Services*. The system is temporarily unavailable to the client because the throttle limit has been exceeded. Throttling ensures consistent and reliable performance for all users of the service.
Notes	HTTP Response Code: 400. For the Version 1.0 API, throttling can be caused by too many client requests per unit time at the individual profile level and at the global system level. If your client repeatedly is causing throttling issues, you will be contacted by the ScholarOne Support Team for assistance in resolving the issue. The server response for this example is simulated.
Request Command Line	java -jar SampleClient.jar --input throttled.xml --output throttled_output.xml
Handler Command Line	-ex handleThrottled
Table 2 - ScholarOne Error Code XREF	500
Table 3 – Suggested response handling strategies XREF	R4, R5

Sample Basic Request: getSubmissionInfoBasicDocumentId (invalid ID format)

DESCRIPTION	This sample shows a problematic call to ScholarOne Web Services. The caller has provided improperly formatted input parameters. This sample is calling the Basic Manuscript Web Service by Document Id which expects numeric Document Id values.
Notes	HTTP Response Code: 400
Request Command Line	java -jar SampleClient.jar --input invalidInputFormatForIds.xml --output invalidInputFormatForIds _output.xml
Handler Command Line	none
Table 2 - ScholarOne Error Code XREF	700
Table 3 – Suggested response handling strategies XREF	R7

Sample Basic Request: getSubmissionInfoBasicDocumentId (too many input IDs)

DESCRIPTION	This sample shows a problematic call to ScholarOne Web Services. The caller has provided too many document IDs. The limit is 25. If the caller includes more than 25, no data is returned.
Notes	HTTP Response Code: 400
Request Command Line	java -jar SampleClient.jar --input moreThan25Inputs.xml --output moreThan25Inputs _output.xml
Handler Command Line	java -jar SampleClient.jar -ex handleValidationError
Table 2 - ScholarOne Error Code XREF	700
Table 3 – Suggested response handling strategies XREF	R7

Sample Basic Request: getSubmissioninfoBasicSubmissionId (draft submission info requested)

DESCRIPTION	This sample shows a problematic call to ScholarOne Web Services. The caller has tried to access a draft manuscript by Document Number (aka Submisison Id).
Notes	HTTP Response Code: 400
Request Command Line	java -jar SampleClient.jar --input submissionIdOfDraft.xml --output submissionIdOfDraft_output.xml
Handler Command Line	none
Table 2 - ScholarOne Error Code XREF	700
Table 3 – Suggested response handling strategies XREF	R7

Sample Basic Request: getSubmissionInfoBasicSubmissionId (request includes no IDs)

DESCRIPTION	This sample shows an unsuccessful call to ScholarOne Web Services because the caller has not provided all the appropriate input values.
Notes	HTTP Response Code: 400
Request Command Line	java -jar SampleClient.jar --input zeroInputValues.xml --output zeroInputValues_output.xml
Handler Command Line	none
Table 2 - ScholarOne Error Code XREF	700
Table 3 – Suggested response handling strategies XREF	R7

Sample Basic Request: getSubmissionInfoBasicDocumentId (valid document ID but wrong site)

DESCRIPTION	This sample shows a problematic call to ScholarOne Web Services. The caller has provided a valid Document Id for another Journal/Site.
Notes	HTTP Response Code: 400
Request Command Line	java -jar SampleClient.jar --input validDocumentIdWrongSite.xml --output validDocumentIdWrongSite_output.xml
Handler Command Line	none
Table 2 - ScholarOne Error Code XREF	701
Table 3 – Suggested response handling strategies XREF	R7

Sample Basic Request: getSubmissionInfoBasicDocumentId (invalid user password)

DESCRIPTION	This sample shows an unsuccessful call to ScholarOne Web Services because the caller has not provided valid credentials.
Notes	HTTP Response Code: 401 – there is no S1 Error Code provided. The response file is empty.
Request Command Line	java -jar SampleClient.jar --input invalidUserPassword.xml --output invalidUserPassword _output.xml
Handler Command Line	java -jar SampleClient.jar -ex handleAuthenticationError
Table 2 - ScholarOne Error Code XREF	none
Table 3 – Suggested response handling strategies XREF	R5

HTTP 500 – Level: Server Failure Response

Sample Basic Request: getSubmissionInfoBasicDocumentId (unexpected error)

DESCRIPTION	This sample shows an unsuccessful call to ScholarOne Web Services. The error cannot be identified or the server is not available. The server response is simulated for this example.
Notes	HTTP Response Code: 500
Request Command Line	java -jar SampleClient.jar --input unexpectedError.xml --output unexpectedError _output.xml
Handler Command Line	java -jar SampleClient.jar -ex handleUnscheduledOutage
Table 2 - ScholarOne Error Code XREF	100
Table 3 – Suggested response handling strategies XREF	R6

Sample Basic Request: getSubmissionInfoBasicDocumentId (server maintenance - API)

DESCRIPTION	This sample shows an unsuccessful call to ScholarOne Web Services. The API is undergoing maintenance. The server response is simulated for this example.
Notes	HTTP Response Code: 500. The response will include a suggested call back time.
Request Command Line	java -jar SampleClient.jar --input maintenanceApi.xml --output maintenanceApi _output.xml
Handler Command Line	none
Table 2 - ScholarOne Error Code XREF	600
Table 3 – Suggested response handling strategies XREF	R6 => R5

Sample Basic Request: getSubmissionInfoBasicDocumentId (server maintenance - Stack)

DESCRIPTION	This sample shows an unsuccessful call to ScholarOne Web Services. The server application is undergoing maintenance. The response will include a suggested call back time. The server response is simulated for this example.
Notes	HTTP Response Code: 500
Request Command Line	java -jar SampleClient.jar --input maintenanceStack.xml --output maintenanceStack _output.xml
Handler Command Line	java -jar SampleClient.jar -ex handleScheduledoutage
Table 2 - ScholarOne Error Code XREF	601
Table 3 – Suggested response handling strategies XREF	R4 => R5

Sample Basic Request: getSubmissionInfoBasicDocumentId (server maintenance - Site)

DESCRIPTION	This sample shows an unsuccessful call to ScholarOne Web Services. The site is undergoing maintenance. The response will include a suggested call back time. The server response is simulated for this example.
Notes	HTTP Response Code: 500
Request Command Line	java -jar SampleClient.jar --input maintenanceSite.xml --output maintenanceSite _output.xml
Handler Command Line	none
Table 2 - ScholarOne Error Code XREF	602
Table 3 – Suggested response handling strategies XREF	R4 => R5

Sample Basic Request: getSubmissionInfoBasicDocumentId (Gateway timeout)

DESCRIPTION	This sample shows an unsuccessful call to ScholarOne Web Services. The server application is undergoing maintenance. The server response is simulated for this example.
Notes	HTTP Response Code: 504
Request Command Line	java -jar SampleClient.jar --input callFailure.xml --output callFailure _output.xml
Handler Command Line	java -jar SampleClient.jar -ex handleGatewayTimeout
Table 2 - ScholarOne Error Code XREF	none
Table 3 – Suggested response handling strategies XREF	R6 => R5

EXHIBITS

Table 1 – Version 1.0 Web Services Summary

METHOD NAME	OUTPUT[8]	COMMENTS
getSubmissionInfoBasic	Author Full Name Author ORCID ID Author Person ID Author Researcher ID Author Membership ID Submission Date Submission Type Submission ID[9] Submission Title Submission Status Decision Name Document Status Name Document Status ID In Draft Flag Task Task Name Task ID Task Status Name Task Status Document ID	• This method returns basic summary data about the submission. • getSubmissionInfoBasicBySubmissionId • getSubmissionInfoBasicByDocumentId
getSubmissionInfoFull	Author Full Name Author ORCID ID Author Researcher ID Author Membership ID Author Person ID Submission Date Due Date of Next Revision/Resubmission Decision Date Submission Date - Original Submission Type Submission ID Submission Title Abstract Submission Status Decision Name Document Status Name Document Status ID In Draft Flag Task Task Name Task ID Task Status Name Task Status	• This method returns detailed summary data about the submission. • Author info relates to submitting Author or submitting Agent • This method will not return drafts and will provide an empty response when the provided documentId is the last and in draft. • getSubmissionInfoFullBySubmissionId • getSubmissionInfoFullByDocumentId

[8] Fields are fully defined in the *ScholarOne Web Services API Reference Guide*
[9] Note that submission ID is case-sensitive.

METHOD NAME	OUTPUT[8]	COMMENTS

	Journal Name
	Decision Type
	Submission ID - Latest
	Submission ID - Original
	Submission Custom Question Information
	  Answer Type
	  Custom Question ID
	  Question Name
	  Question Status
	  Question Text
	Submission Custom Answer Information
	  Abbreviated Response
	  Answer Name
	  Answer ID
	  Answer Status
	  Answer Text
	DOI
	Revision Number
	Submission Files Information
	  Customer File Name
	  File Designation
	  System File Name
	  File ID
	Transmission Date
	Submission Flags
	  Flag Configured Text
	  Flag Name
	CrossCheck: Overall Similarity Index
	Archive Date
	Archive Status ID
	Archive Status
	Withdrawn Date
	Reference Submission Information
	  Reference Manuscript Document ID
	  Reference Manuscript Submission ID
	  Reference Manuscript Title
	  Reference Manuscript Submission Type
	Document ID
	Document ID - Latest
	Document ID - Original	

getAuthorInfoBasic	Author Full Name Author Type: Co-Author Author Type: Contact Author Author Type: Corresponding Author Author Type: Submitting Author Author Type: Submitting Agent Author ORCID ID Author Person ID Author Researcher ID Author Membership ID Author Order Number Submission ID Document ID	• This method returns basic author list data for a submission. • getAuthorBasicBySubmissionId • getAuthorBasicByDocumentId
getAuthorInfoFull	Submission ID Document ID Author Person ID	• This method returns full author list data for a submission.

METHOD NAME	OUTPUT[8]	COMMENTS
	Author Order Number Author Full Name Author Salutation Author First Name Author Middle Name Author Last Name Author Suffix Author Type: Co-Author Author Type: Contact Author Author Type: Corresponding Author Author Type: Submitting Author Author Type: Submitting Agent Author Type: Invited Author Invited Author: Date Selected Invited Author: Date Invited Invited Author: Date Responded Invited Author: Date Assigned Invited Author: Invitation Response Invited Author: Invitation Response ID Author Primary E-Mail Address Author ORCID ID Author Researcher ID Author Membership ID Author Full Address Affiliations Order Number Department1 Institution1 Address1 Address2 Address3 Country Country Code State / Province City Postal Code Phone1 Phone2 Fax Title Room / Suite	• getAuthorFullByDocumentId • getAuthorFullBySubmissionId • Author Order Number is blank for Submitting Agent

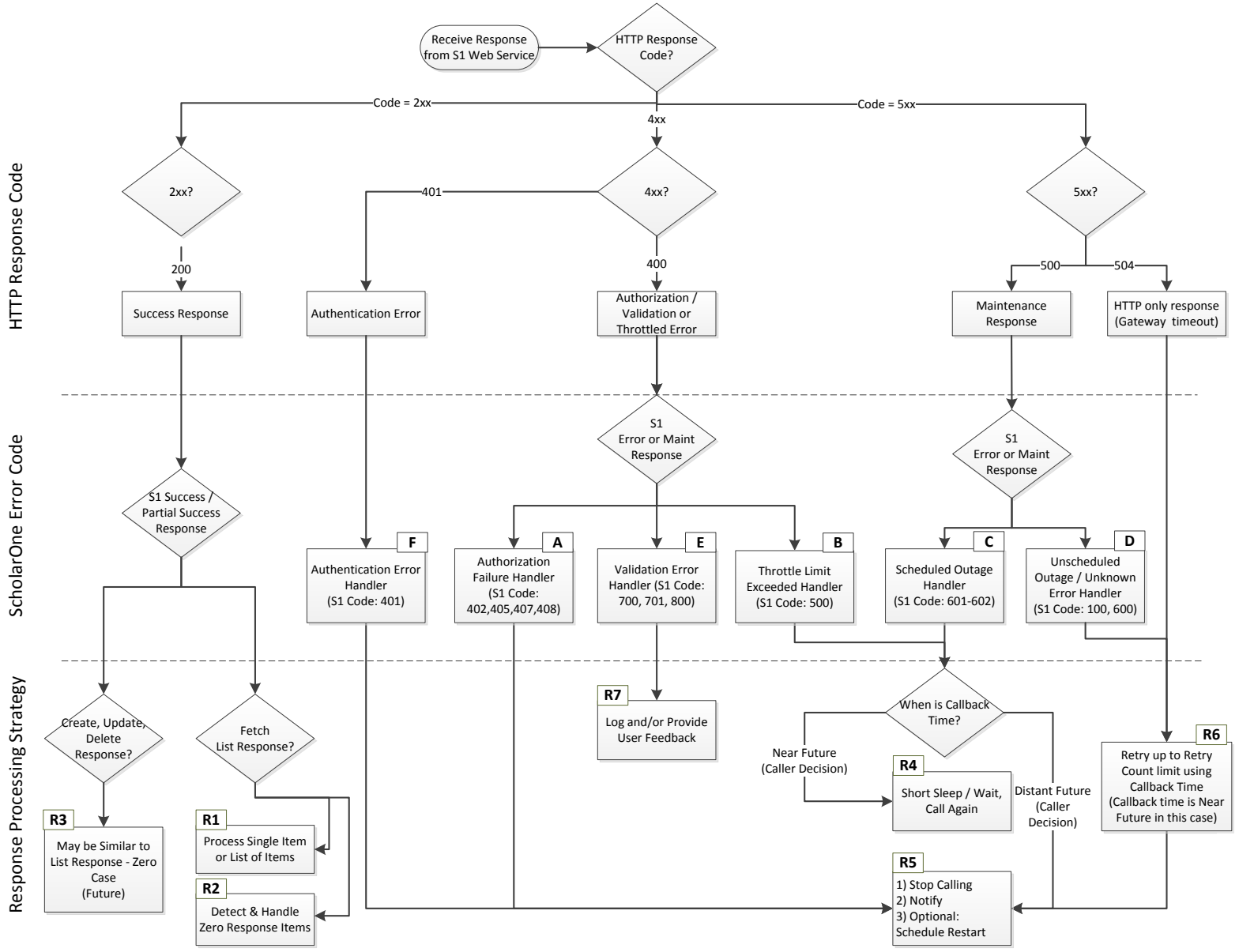
HTTP Response Code
ScholarOne Error Code
Response Processing Strategy
Receive Response from S1 Web Service
HTTP Response Code?
Code = 2xx
4xx
Code = 5xx
2xx?
4xx?
5xx?
200
401
400
500
504
Success Response
Authentication Error
Authorization / Validation or Throttled Error
Maintenance Response
HTTP only response (Gateway timeout)
S1 Success / Partial Success Response
S1 Error or Maint Response
S1 Error or Maint Response
F
Authentication Error Handler (S1 Code: 401)
A
Authorization Failure Handler (S1 Code: 402,405,407,408)
E
Validation Error Handler (S1 Code: 700, 701, 800)
B
Throttle Limit Exceeded Handler (S1 Code: 500)
C
Scheduled Outage Handler (S1 Code: 601-602)
D
Unscheduled Outage / Unknown Error Handler (S1 Code: 100, 600)
Create, Update, Delete Response?
Fetch List Response?
R7
Log and/or Provide User Feedback
When is Callback Time?
Near Future (Caller Decision)
Distant Future (Caller Decision)
R6
Retry up to Retry Count limit using Callback Time (Callback time is Near Future in this case)
R3
May be Similar to List Response - Zero Case (Future)
R1
Process Single Item or List of Items
R2
Detect & Handle Zero Response Items
R4
Short Sleep / Wait, Call Again
R5
1) Stop Calling
2) Notify
3) Optional: Schedule Restart

Table 2 – ScholarOne Response Code Explanations

S1 Response Code	HTTP Response Code	Response Status/Type	Logical Flow Cross Reference [10]	Client or Server Error?	User Message	Notes
100	500	FAILURE/ Service Related	D	Server	An unexpected error has occurred or System is not available	This is a generic error that may occur in various unexpected exception scenarios
401[11]	401	FAILURE/ Authentication	F	Client	Not Authenticated: UserName/Password is incorrect	The user/API key combination is not valid
402	400	FAILURE/ Authorization	A	Client	Not Authorized	The user/API key combination is not authorized to access the Web Service operation
405	400	FAILURE/ Authorization	A	Client	Not Authorized: incorrect site	The user/API key combination is not authorized to access the Journal/Site specified in the site_name parameter
407	400	FAILURE/ Authorization	A	Client	Not Authorized: incorrect permissions	The user/API key combination is not authorized to access the Web Service operation
408	400	FAILURE/	A	Client	Not Authorized: operation	The user/API key combination is not authorized to access the

[10] Cross Reference to Error Codes depicted in Figure 2

[11] Detection of the server-generated nonce expiration use case should be handled by the challenge-response digest access authentication method.

S1 Response Code	HTTP Response Code	Response Status/Type	Logical Flow Cross Reference [10]	Client or Server Error?	User Message	Notes
		Authorization			unavailable	Web Service operation
500	400	FAILURE/ Throttle Related	B	Server	Throttle Limit Exceeded	Throttle limits will be enforced globally at the system level and also at the individual profile level. If either throttle threshold is exceeded, S1 will return a 500. It may become necessary to adjust your web client to limit the number of new session calls or the number of records requested per minute, hour, day, month, and year.
600	500	FAILURE/ Maintenance	D	Server	Maintenance	Web Service Platform Maintenance, All Web Services are off-line for all products
601	500	FAILURE/ Maintenance	C	Server	Maintenance	Web Service Platform Maintenance, All Web Services are off-line for all products
602	500	FAILURE/ Maintenance	C	Server	Maintenance: site in maintenance	Individual Application Maintenance, Web Services for specific products are off-line
700	400	FAILURE/ Validation Related	E	Client	Validation error: invalid input	API Validation Error. Could be an API-level validation error (invalid input), or an API exception such as a schema mismatch.

S1 Response Code	HTTP Response Code	Response Status/Type	Logical Flow Cross Reference [10]	Client or Server Error?	User Message	Notes
701	400	FAILURE/ Validation Related	E	Client	Validation Error: Site shortname	Validation Error. Could be an API-level validation error (invalid input), or an API exception such as a schema mismatch.
800	400	FAILURE/ Validation Related	E	Client	Resubmit request with valid input	Error from specific Web Service operation. The error message is custom from the Web Service operation implementation.

Table 3 - Suggested Strategies for Client-Side Response Handling

S1 Error Code	Figure 2 Cross Reference ID	Suggested Client Strategy	Notes
None (Success)	R1	Client should handle standard success response for retrieval of a single item	Service successfully returns a single record of data. This is a special case of list interactions. It is a list of size 1.
None (Success)	R1	Client should handle standard success response for retrieving a list of items	Service successfully returns more than one record of data.
None (Success)	R2	Client should detect and handle standard success response that retrieves no items	The submitted Document Id/Number doesn't exist on the Site/Journal but the service returns the status of SUCCESS because the call processed and gave the appropriate reply.
None (Success)	R3	Client should log successful response to a create, update or delete request	(future) The action requested by the client was received, understood, accepted and processed successfully.
500, 601-602	R4	Client should register response callBackTime and retry the request on or after suggested callBackTime	All maintenance responses will include a suggested callBackTime in UTC based on the 24-hour UTC time scale. In the message response UTC will be presented with the "Z" special designator indicating Coordinated Universal time, not local time.
100, 401, 402, 405, 407, 408, 600, 601, 602 (and HTTP 504)	R5	Client should exit gracefully or otherwise suspend calls to the server and initiate notification and restart procedures	Communication with the service was not successfully initiated or has been terminated. Retry limit has been exhausted or callBackTime (if provided) suggests scheduling a restart rather than continuing to retry.

S1 Error Code	Figure 2 Cross Reference ID	Suggested Client Strategy	Notes
100, 600	R6	Using the suggested callBackTime, client should retry the request until the client-configured retry count limit has been reached at which time the client should exit gracefully and initiate notification and restart procedures	The service is unavailable for an unknown reason.
700, 701, 800	R7	Client should log the issue and/or present feedback to the user.	A validation error has occurred. Invalid input was submitted, or the site shortname was incorrect. Re-submit request with valid input.

Table 4 – Sample Client – Options & Call Parameters

The Sample Client has been implemented to support examples of the following runtime options and call parameters. Please note:

- Parameters are presented in hierarchical order of processing precedence: each one is valid when the option(s) do not precede it on the command line.
- Unmarshalling is not supported for JSON.
- The –pretty and –prettyprint command line option can also be formatted as -pretty XML (and –prettyprint XML); -pretty json; and -pretty xml).
- The --example runtime option was implemented with the intention of providing a more robust view of how *ScholarOne Web Services* work. It demonstrates call response processing for predefined examples. While these examples can work with any input file, we use these particular examples to demonstrate certain types of responses.

Option/Call Parameter	Description	Through Command line	Input file	Other Required Options/Parameters	Other Supported Options/Parameters	Other Unsupported Options/Parameters
Help	Sample client runtime option to print the help message showing allowed options and inputs	-h or --help	NA		☐ Ignores all others	

Option/Call Parameter	Description	Through Command line	Input file	Other Required Options/Parameters	Other Supported Options/Parameters	Other Unsupported Options/Parameters
Inputfile	Sample client runtime option to provide a predefined input file	-in <arg> OR --input <arg>	NA	· -out, --output (used if not already set in the input file, otherwise ignored) · -f, --format (used if not already set in the input file, otherwise ignored) · -pretty, --prettyprint · -u, -- unmarshall · Ignores all others		
Example	Sample client runtime option to demonstrate call response processing for predefined examples	-ex <arg> OR --example <arg>		· -pretty, --prettyprint · -out, --output (used if not already set in the input file, otherwise ignored) · Ignores all others		· -f, --format (unsupported - Entering 'json' would cause exception)
URL	Web Service call input - The Web Service URL, which identifies the Service	-url <arg>	<url>	· -site, --site_name · -user, --username · -pwd, --password	· -ids · -out, --output · -pretty, --prettyprint · -u, -- unmarshall · -f, --format · -loc, --locale_id · -extid, --external_id	

Option/Call Parameter	Description	Through Command line	Input file	Other Required Options/Parameters	Other Supported Options/Parameters	Other Unsupported Options/Parameters
IDs	Web Service call parameter - Document Id(s) or Submission Id(s) based on the Service	-ids <arg>	<ids>	· -site, --site_name · -user, --username · -pwd, --password · -url	· -out, --output · -pretty, --prettyprint · -u, -- unmarshall · -f, --format · -loc, --locale_id · -extid, --external_id	
Site Name	Web Service call parameter - the site short name	-site <arg> OR --site_name <arg>	<site_name>	· -user, --username · -pwd, --password · -url	· -ids · -out, --output · -pretty, --prettyprint · -u, -- unmarshall · -f, --format · -loc, --locale_id · -extid, --external_id	
Locale Id	Web Service call parameter - optional parameter to set the Locale	--locale_id <arg>	<locale_id>	· -site, --site_name · -user, --username · -pwd, --password · -url	· -ids · -out, --output · -pretty, --prettyprint · -u, -- unmarshall · -f, --format · -extid, --external_id	

Option/Call Parameter	Description	Through Command line	Input file	Other Required Options/Parameters	Other Supported Options/Parameters	Other Unsupported Options/Parameters
User Name	Web Service call parameter - The Profile User or Account Name	--username <arg>	<user_name>	· -site, --site_name · -pwd, --password · -url	· -ids · -out, --output · -pretty, --prettyprint · -u, -- unmarshall · -f, --format · -loc, --locale_id · -extid, --external_id	
Password	Web Service call parameter - The API_KEY for the sample client profile	· -pwd, --password	<password>	· -site, --site_name · -user, --username · -url	· -ids · -out, --output · -pretty, --prettyprint · -u, -- unmarshall · -f, --format · -loc, --locale_id · -extid, --external_id	
Format	Web Service call parameter - The Data type requested from the Service <xml> or <json>. This defaults to <xml> if Unmarshall (–u, --unmarshall ,	-f <arg> OR --format <arg>	<format>	· -site, --site_name · -user, --username · -pwd, --password · -url	· -ids · -out, --output · -pretty, --prettyprint · -loc, --locale_id · -extid, --external_id	* note that in production formatting output is accomplished via the *_type* option, see the API Reference Guide for

Option/Call Parameter	Description	Through Command line	Input file	Other Required Options/Parameters	Other Supported Options/Parameters	Other Unsupported Options/Parameters
	<unmarshall>) option is used					details
ExternalId	Web Service call parameter – An id value that can be set by the client for call tracking. Will be returned as <ProfileCallId> in the call response	-extid <arg>, --external_id <arg>	<external_id>	· -site, --site_name · -user, --username · -pwd, --password · -url	· -ids · -out, --output · -pretty, --prettyprint · -u, -- unmarshall · -f, --format · -loc, --locale_id	
Outputfile	Sample client runtime option to print the Web Service call response to a specified	-out <arg> OR -- output <arg>	<outputFile>	· -site, --site_name · -user, --username · -pwd, --password · -url	· -ids · -pretty, --prettyprint · -u, -- unmarshall · -f, --format · -loc, --locale_id · -extid, --	

Option/Call Parameter	Description	Through Command line	Input file	Other Required Options/Parameters	Other Supported Options/Parameters	Other Unsupported Options/Parameters
	file				external_id	
PrettyPrint	Sample client runtime option to format the output on the console or output file	-pretty OR -- prettyprint	<pretty>	· -site, --site_name · -user, --username · -pwd, --password · -url	· -ids · -out, --output · -u, -- unmarshall · -f, --format · -loc, --locale_id · -extid, --external_id	
Unmarshall	Sample client runtime option to unmarshall the output for xml format and handle Error Scenarios. Defaults to no unmarshalling (raw mode) if not set	-u OR --unmarshall	<unmarshall>	· -site, --site_name · -user, --username · -pwd, --password · -url	· -ids · -out, --output · -pretty, --prettyprint · -loc, --locale_id · -extid, --external_id · Ignores -f,--format	

Figures and Tables - Cross Reference Back Links

Clarivate Analytics

Table/Figure	Guide Section	Cross Reference Back Link (control-click to return to reference source)
Figure 1	Your Technical Guide	Figure 1 - Web Service Call Flow Processing presents the basic Web Services call flow from connecting to the server, through authentication, and on to retrieving data.
Figure 1	Key Examples Overview	The Sample Client will present examples of all possible scenarios encountered in the Web Services call flow depicted in Figure 1 - Web Service Call Flow Processing.
Figure 1	Request/Response Call Flow Processing	The Web Service call flow illustrated in Figure 1 - Web Service Call Flow Processing is very important because it shows the progression of request management through communication setup, caller authentication, authorization, and finally API method fulfillment.
Figure 2	Technical Guide to Integration	Figure 2 - Response Processing Logical View provides a detailed view of response processing and handling techniques.
Figure 2	Technical Guide to Integration	Table 3 - Suggested Strategies for Client-Side Response Handling expands upon the brief response strategy suggestions of Figure 2.
Figure 2	Technical Guide to Integration	The Response Processing Strategy section of Figure 2 guides you through the methods that must be developed by you to properly handle ScholarOne Web Service error codes.
Figure 2	Key Examples Overview	In addition, the Sample Client includes coding samples to demonstrate some key techniques to be used when interacting with the Services, including the management of valid output data as well as the error scenarios depicted in Figure 2 - Response Processing Logical View.

Figure 2	Key Examples Overview	Figure 2 - Response Processing Logical View presents ScholarOne Error Codes with user messages that fall into the 3 main HTTP response categories. More information regarding each error condition is provided in Table 2 – ScholarOne Response Code Explanations, cross-referenced by the letter shown on the Logical View Diagram in the “ScholarOne Error Code” section.
Figure 2	Error Handling	Suggested strategies for properly handling or otherwise reattempting failed requests are presented in Figure 2 - Response Processing Logical View and are expanded upon in Table 3 - Suggested Strategies for Client-Side Response Handling.
Table 1	Web Services Overview	See Table 1 – Version 1.0 Web Services Summary for the information provided for each API call.
Table 2	Your Technical Guide	Table 2 – ScholarOne Response Code Explanations provides substantive detail on the call responses that will require client-side methods be developed for proper handling and subsequent client-server interaction.
Table 2	Key Examples Overview	More information regarding each error condition is provided in Table 2 – ScholarOne Response Code Explanations, cross-referenced by the letter shown on the Logical View Diagram in the “ScholarOne Error Code” section.
Table 2	Individual Example	Table 2 - ScholarOne Error Code XREF
Table 3	Your Technical Guide	. Table 3 - Suggested Strategies for Client-Side Response Handling expands upon the brief response strategy suggestions of Figure 2.
Table 3	Key Examples	These techniques are shown on the Logical View Diagram using the numbering convention “R1” – “R7” in the “Response Processing Strategy” section of the diagram and are cross-referenced by this number to the

		suggested response handling techniques in Table 3 - Suggested Strategies for Client-Side Response Handling.
Table 3	Individual Example	Table 3 – Suggested response handling strategies XREF
Table 3	Error Handling	Suggested strategies for properly handling or otherwise reattempting failed requests are presented in Figure 2 - Response Processing Logical View and are expanded upon in Table 3 - Suggested Strategies for Client-Side Response Handling
Table 4	Key Concepts	See Table 4 – Sample Client – Options & Call Parameters for all of the Sample Client's command line options and call parameters.
Table 4	Sample Client Overview	See Table 4 – Sample Client – Options & Call Parameters for a complete listing.
Table 4	Sample Client Web Services Basics	Please refer to Table 4 – Sample Client – Options & Call Parameters for a complete listing of command line options and associated parameters supported in the ScholarOne Web Services client.

ScholarOne®

ScholarOne, a Clarivate Analytics Business, provides comprehensive workflow management systems for scholarly journals, books, and conferences. Its web-based applications enable publishers to manage the submission, peer review, production, and publication processes more efficiently, increasing their profile among authors, decreasing time-to-market for critical scientific data, and lowering infrastructure costs. ScholarOne offers workflow solutions for the submission and review of manuscripts, abstracts, proceedings, books, grants & awards, and production. Supporting over 365 societies and publishers, over 3,400 books and journals, and 13 million users, ScholarOne is the industry leader.

To learn more, visit:
Clarivate.com

CLARIVATE ANALYTICS MAIN OFFICES

North America:
+1 888 399 2917

Europe, Middle East & Africa:
+442038114093

Latin America:
+551183709845

Japan:
+81345893100

Asia Pacific:

Australia +61285877636
New Zealand +61285877636
China +861057601200
India +911130446419
Korea +82220768100
SE Asia & Pakistan +6567755088
Taiwan +886225033034

clarivate.com

Effective Date: 27-September-2017
Document Version: 1.1

Clarivate Analytics