[MS-ASUR-Diff]:

Analysis Services Usage Reporting

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards ~~as well as overviews of the interaction among each of these technologies~~support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights**. This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you ~~may~~can make copies of it in order to develop implementations of the technologies that are described in ~~the Open Specifications~~ this documentation and ~~may~~can distribute portions of it in your implementations ~~using~~that use these technologies or in your documentation as necessary to properly document the implementation. You ~~may~~can also distribute in your implementation, with or without modification, any ~~schema, IDL's~~schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications~~.~~ documentation.
- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
- **Patents**. Microsoft has patents that ~~may~~might cover your implementations of the technologies described in the Open Specifications~~.~~ documentation. Neither this notice nor Microsoft's delivery of ~~the~~this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open ~~Specification may~~Specifications document might be covered by the Microsoft Open Specifications Promise or the Microsoft Community Promise. If you would prefer a written license, or if the technologies described in ~~the Open Specifications~~this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs**. To see all of the protocols in scope under a specific license program and the associated patents, visit the Patent Map.
- **Trademarks**. The names of companies and products contained in this documentation ~~may~~might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit ~~.~~www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, ~~e-mail~~email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than as specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications ~~do~~documentation does not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments, you are free to take advantage of them. Certain Open Specifications documents are intended for use in conjunction with publicly available ~~standard~~standards specifications and network programming art~~,~~ and ~~assumes~~, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

Support. For questions and support, please contact dochelp@microsoft.com.

Revision Summary

Date	Revision History	Revision Class	Comments
5/10/2016	1.0	New	Initial Availability
8/16/2017	1.0	None	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

The Analysis Services Usage Reporting protocol specifies a method by which a client application, that gathers Analysis Services models from a host server and then loads them onto other servers that are running Analysis Services, can report back to that host server with the details about how those models are being used and the resources those models consume on the other servers.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

connection handle: A GUID that represents a unique connection that is made to a previously loaded and reported Analysis Services model. The Usage Reporting Service generates a unique handle for each connection and returns that GUID to the model's client application.

model handle: A GUID that represents an Analysis Services model that is loaded on a server that is running an Analysis Services instance. The Usage Reporting Service generates a unique handle for each such model and returns that handle to each model's client application.

PowerPivot mode: A server deployment mode of Microsoft SQL Server Analysis Services that supports loading models that are streamed from a client application.

WSDL message: An abstract, typed definition of the data that is communicated during a WSDL operation [WSDL]. Also, an element that describes the data being exchanged between web service providers and clients.

WSDL operation: A single action or function of a web service. The execution of a WSDL operation typically requires the exchange of messages between the service requestor and the service provider.

XML namespace: A collection of names that is used to identify elements, types, and attributes in XML documents identified in a URI reference [RFC3986]. A combination of XML namespace and local name allows XML documents to use elements, types, and attributes that have the same names but come from different sources. For more information, see [XMLNS-2ED].

XML Schema (XSD): A language that defines the elements, attributes, namespaces, and data types for XML documents as defined by [XMLSCHEMA1/2] and [W3C-XSD] standards. An XML schema uses XML syntax for its language.

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

[SOAP1.2~~/~~-1/2007] Gudgin, M., Hadley, M., Mendelsohn, N., ~~Moreau, J., and Nielsen, H.F~~et al., "SOAP Version 1.2 Part 1: Messaging Framework~~",~~ (Second Edition)", W3C Recommendation, ~~June 2003~~April 2007, http://www.w3.org/TR/~~2003~~2007/REC-soap12-part1-~~20030624~~20070427/

[WSA1.0] ~~World Wide Web Consortium,~~Gudgin, M., Hadley, M., Rogers, T., et al., Eds., "Web Services Addressing 1.0 - WSDL Binding", W3C Candidate Recommendation, May 2006, http://www.w3.org/TR/2006/CR-ws-addr-wsdl-20060529/

[WSDL] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S., "Web Services Description Language (WSDL) 1.1", W3C Note, March 2001, http://www.w3.org/TR/2001/NOTE-wsdl-20010315

[XMLNS] Bray, T., Hollander, D., Layman, A., et al., Eds., "Namespaces in XML 1.0 (Third Edition)", W3C Recommendation, December 2009, http://www.w3.org/TR/2009/REC-xml-names-20091208/

[XMLSCHEMA1/2] Thompson, H., Beech, D., Maloney, M., and Mendelsohn, N., Eds., "XML Schema Part 1: Structures Second Edition", W3C Recommendation, October 2004, http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/

[XMLSCHEMA2/2] Biron, P., and Malhotra, A., Eds., "XML Schema Part 2: Datatypes Second Edition", W3C Recommendation, October 2004, http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/

1.2.2 Informative References

~~None.~~

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000, http://www.rfc-editor.org/rfc/rfc2818.txt

[RFC7230] Fielding, R., and Reschke, J., Eds., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, June 2014, http://www.rfc-editor.org/rfc/rfc7230.txt

[RFC793] Postel, J., Ed., "Transmission Control Protocol: DARPA Internet Program Protocol Specification", RFC 793, September 1981, http://www.rfc-editor.org/rfc/rfc793.txt

[SOAP1.1] Box, D., Ehnebuske, D., Kakivaya, G., et al., "Simple Object Access Protocol (SOAP) 1.1", W3C Note, May 2000, http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[SOAP1.2-2/2007] Gudgin, M., Hadley, M., Mendelsohn, N., et al., "SOAP Version 1.2 Part 2: Adjuncts (Second Edition)", W3C Recommendation, April 2007, http://www.w3.org/TR/2007/REC-soap12-part2-20070427

1.3 Overview

The Analysis Services Usage Reporting protocol provides a method for a server that hosts an analysis services tabular model to track client application usage of that model. When a client application, which itself can be a server, uses any document that contains a SQL Server Analysis Services Tabular Model that is hosted by a server in any file format, the server that hosts the model can track client usage of that model.

The conceptual flow of this protocol is that the client application reports when it loads a model and when it connects to a loaded model, and the client refers back to those load and connect operations at a later time when any queries against the model are completed.

Additionally, this protocol defines a mechanism for the client application to check the known Service URI to validate that a service implementing this protocol exists.

The following ought to be considered before this protocol is used with any client application:

- The reporting of usage data is not inherently required and, therefore, cannot be assumed to be supported by either server or client.
- The server that hosts the model defines a static location for the service that is implementing this protocol to exist.

1.4 Relationship to Other Protocols

Analysis Services uses the SOAP messaging protocol for formatting requests and responses as specified either in [SOAP1.1] or in [SOAP1.2-1/2007] and [SOAP1.2-2/2007]. It transmits these messages using HTTP [~~RFC2616~~RFC7230], HTTPS [RFC2818], or TCP [RFC793].

This protocol uses SOAP over HTTP or HTTPS, as shown in the following layering diagram:

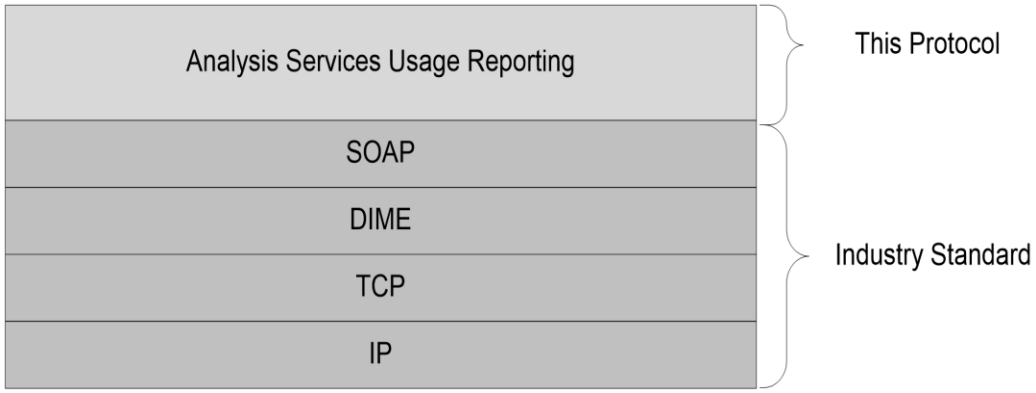

Figure 1: SOAP over HTTP or HTTPS

1.5 Prerequisites/Preconditions

This protocol assumes that the following preconditions exist in the server environment:

- Security authentication/authorization has already taken place at a lower layer of the protocol stack.
- A client application is managing an Analysis Services tabular model that needs to be loaded into an Analysis Services instance that is running in PowerPivot mode.
- The client application is capable of determining the URI of the implementing service by itself. This protocol provides for no discovery mechanism other than a simple ping request to validate that an implementing service exists.
- Any usage action that is reported refers specifically to the Analysis Services instance in which the action takes place. Action that is taken on a model that is not specific to a specific Analysis Services instance is not relevant to this protocol and cannot be reported because the protocol does not support such action.

1.6 Applicability Statement

This protocol applies whenever an Analysis Services tabular model that is hosted on a server needs to be handled and loaded onto an Analysis Services instance in PowerPivot mode. A likely candidate for using this protocol is indicated when the server that originally hosts the model and the application that loads the model into the instance are not the same application.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

This protocol relies on SOAP Version 1.2 [SOAP1.2/-1/2007].

Any security that is required has to occur at the HTTP/HTTPS layer and is assumed to be defined at a per-implementation level. Every implementing service can require different security, as it deems necessary.

2.2 Common Message Syntax

This section contains common definitions that are used by this protocol. The syntax of the definitions uses XML Schema as defined in [XMLSCHEMA1/2] and [XMLSCHEMA2/2] and the definitions of Web Services Description Language as defined in [WSDL].

2.2.1 Namespaces

This specification defines and references various XML namespaces by using the mechanisms specified in [XMLNS]. Although this specification associates a specific XML namespace prefix for each XML namespace that is used, the choice of any particular XML namespace prefix is implementation-specific and not significant for interoperability.

Prefix	Namespace URI	Reference
	http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting	
tns	http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting	
serialization	http://schemas.microsoft.com/2003/10/Serialization/	
wsaw	http://www.w3.org/2006/05/addressing/wsdl	[WSA1.0]
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]
xs	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1/2] [XMLSCHEMA2/2]

2.2.2 Messages

This specification does not define any common XML schema message definitions.

2.2.3 Elements

This specification does not define any common XML schema element definitions.

2.2.4 Complex Types

This specification does not define any common XML schema complex type definitions.

2.2.5 Simple Types

The following table summarizes the set of common XML Schema simple type definitions defined by this specification. XML Schema simple type definitions that are specific to a particular operation are described with the operation.

Simple type	Description
xs:boolean	A simple Boolean value.
xs:string	Any string value.
xs:long	A 64-bit integer value.
xs:int	A 32-bit integer value.
serialization:guid	A UUID string that represents a GUID.

2.2.5.1 xs:boolean

An **xs:Boolean** simple type specifies a value that indicates true or false.

The following is the XML Schema definition for the **xs:boolean** simple type.

```
<xs:element name="boolean" nillable="true" type="xs:boolean"/>
```

2.2.5.2 xs:string

An **xs:string** simple type is any string value.

The following is the XML Schema definition for the **xs:string** simple type.

```
<xs:element name="string" nillable="true" type="xs:string"/>
```

2.2.5.3 xs:long

An **xs:long** simple type is a 64-bit integer.

The following is the XML Schema definition for the **xs:long** simple type.

```
<xs:element name="long" nillable="true" type="xs:long"/>
```

2.2.5.4 xs:int

An **xs:int** simple type is a 32-bit integer.

The following is the XML Schema definition for the **xs:int** simple type.

```
<xs:element name="int" nillable="true" type="xs:int"/>
```

2.2.5.5 serialization:guid

A **serialization:guid** simple type provides a unique identifier (GUID).

The following is the XML Schema definition for the **serialization:guid** simple type.

```
<xs:element name="guid" nillable="true" type="serialization:guid" />
```

```
<xs:simpleType name="guid">
    <xs:restriction base="xs:string">
        <xs:pattern value="[\da-fA-F]{8}-[\da-fA-F]{4}-[\da-fA-F]{4}-[\da-fA-F]{4}-[\da-fA-
F]{12}" />
    </xs:restriction>
</xs:simpleType>
```

2.2.6 Attributes

This specification does not define any common XML schema attribute definitions.

2.2.7 Groups

This specification does not define any common XML schema group definitions.

2.2.8 Attribute Groups

This specification does not define any common XML schema attribute group definitions.

2.2.9 Common Data Structures

This specification does not define any common XML schema data structures.

3 Protocol Details

3.1 IPowerPivotUsageReportingService Server Details

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

A sample server needs to maintain two sets of objects in this protocol:

- Model handles—This set of objects contains a key identifier for each Analysis Services model. A model handle is generated by the server for each model when the client application indicates it has loaded that model into an Analysis Services instance. This way, both the client and the server can refer to this particular instance of the loaded model by using a common key. Abstractly, this GUID needs to be keyed into a dictionary where the rest of the load data, such as the image identifier, server name, and so on, resides.
- Connection handles—This set of objects contains a key identifier for each new connection to an Analysis Services model. A connection handle is generated by the server for each new connection to a model, that was previously loaded and reported when the client application indicates that it has created such a connection.

By using these two sets of objects, it is possible for the server to recall enough state to record information about the interactions that the client application is having with the Analysis Services instance.

For consistency, all messages are represented in a wrapped message type that is specific to each operation.

3.1.2 Timers

From time to time, it is necessary for the server to remove stale data to keep the lists of handles clean of orphans, that is, loads that were reported but that no longer have a client working with them. This action is implementation specific, but a reasonable time is 60 minutes, which is enough time for most operations on the model to complete. Note that this amount of time is the amount of time from last access of the handle and not the amount of time since the handle was created.

3.1.3 Initialization

The server MUST start, begin listening for requests, and initialize its handle collections to empty lists, meaning that no handles have been assigned or created yet. All valid connections begin stateless, so no further initialization is required.

3.1.4 Message Processing Events and Sequencing Rules

The following table summarizes the list of WSDL operations that are defined by this specification.

Operation	Description
IsAvailable	Allows a client application to function, regardless of whether there is a

Operation	Description
	service available on the hosting server.
Connect	Alerts the server to a loaded model on which the client application will perform one or more operations.
Load	Indicates that the client operation is now loading the Analysis Services model into an Analysis Services instance.
MachineHealthCalculated	Used whenever the client application determines a new health state for the server that is running an Analysis Services instance.
RequestComplete	Indicates that some given action on the Analysis Services instance has completed.
Unload	Indicates that the client application has finished with a model and that model is to be unloaded from the server.
UnloadAbandoned	Indicates that the client application is unable to finish unloading the model or has timed out.

3.1.4.1 IsAvailable

The **IsAvailable** operation is called when the client application handles, for the first time, a document that corresponds to the associated server. The client application uses this operation to determine whether usage events are to be reported for this server. The server returns a "true" response if usage reporting is to be implemented. The server returns a response of "false" if the service does not exist or throws an exception. Apart from logging, the client handles both events in the same way.

The client application then continues to call this operation intermittently to determine whether usage reporting has been implemented on the server. A call to this method occurs intermittently for the lifespan of the client application, as long as there are documents that are still being used from the server that hosts the usage reporting service.

The following is the WSDL definition of the **IsAvailable** operation.

```
<wsdl:operation name="IsAvailable">
  <wsdl:input
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/IsAvailable"
message="tns:IPowerPivotUsageReportingService_IsAvailable_InputMessage"/>
  <wsdl:output
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/IsAvailableResponse"
message="tns:IPowerPivotUsageReportingService_IsAvailable_OutputMessage"/>
</wsdl:operation>
```

3.1.4.1.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to the IsAvailable operation.

Message	Description
IPowerPivotUsageReportingService_IsAvailable_InputMessage	The message that the client application sends to the server to ask for availability.
IPowerPivotUsageReportingService_IsAvailable_OutputMessage	The message with which the server responds to the client application with availability status.

3.1.4.1.1.1 IPowerPivotUsageReportingService_IsAvailable_InputMessage

The **IPowerPivotUsageReportingService_IsAvailable_InputMessage** message is the request message for the **IsAvailable** operation.

The SOAP body contains an **IsAvailable** element

```
<wsdl:message name="IPowerPivotUsageReportingService_IsAvailable_InputMessage">
    <wsdl:part name="parameters" element="tns:IsAvailable"/>
</wsdl:message>
```

3.1.4.1.1.2 IPowerPivotUsageReportingService_IsAvailable_OutputMessage

The **IPowerPivotUsageReportingService_IsAvailable_OutputMessage** message is the response message for the **IsAvailable** operation.

The SOAP body contains an **IsAvailableResponse** element.

```
<wsdl:message name="IPowerPivotUsageReportingService_IsAvailable_OutputMessage">
    <wsdl:part name="parameters" element="tns:IsAvailableResponse"/>
</wsdl:message>
```

3.1.4.1.2 Elements

The following XML Schema element definitions are specific to the IsAvailable operation.

Element	Description
IsAvailable	Contains parameters for the **IsAvailable** operation.
IsAvailableResponse	Contains the response for the **IsAvailable** operation.

3.1.4.1.2.1 IsAvailable

The **IsAvailable** element is an empty wrapper element around an operation that accepts no parameters. This element can be omitted from calls to the IsAvailable operation.

The following is the XML Schema definition of the **IsAvailable** element.

```
<xs:element name="IsAvailable">
```

```
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
```

3.1.4.1.2.2 IsAvailableResponse

The **IsAvailableResponse** element holds the details of the response to the IsAvailable operation.

The following is the XML Schema definition of the **IsAvailableResponse** element.

```
<xs:element name="IsAvailableResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element minOccurs="0" name="IsAvailableResult" type="xs:boolean"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

IsAvailableResult: Specifies whether this server is available for usage reporting.

3.1.4.2 MachineHealthCalculated

The **MachineHealthCalculated** operation is called when the client application recalculates the health of one of the Analysis Services servers it is working with. This is an optional reporting operation and is not required for reporting usage of individual models, though it is recommended for truly useful reports.

The following is the WSDL definition of the **MachineHealthCalculated** operation.

```
<wsdl:operation name="MachineHealthCalculated">
    <wsdl:input
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/MachineHealthCalculated"
message="tns:IPowerPivotUsageReportingService_MachineHealthCalculated_InputMessage"/>
    <wsdl:output
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/MachineHealthCalculatedResponse"
message="tns:IPowerPivotUsageReportingService_MachineHealthCalculated_OutputMessage"/>
</wsdl:operation>
```

3.1.4.2.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to the MachineHealthCalculated operation.

Message	Description
IPowerPivotUsageReportingService_MachineHealthCalculated_InputMessage	The message that contains the input parameters.
IPowerPivotUsageReportingService_MachineHealthCalculated_OutputMessage	The message that contains the output parameters.

3.1.4.2.1.1 IPowerPivotUsageReportingService_MachineHealthCalculated_InputMessage

The **IPowerPivotUsageReportingService_MachineHealthCalculated_InputMessage** message is the request message for the **MachineHealthCalculated** operation.

The SOAP body contains a **MachineHealthCalculated** element.

```
<wsdl:message name="IPowerPivotUsageReportingService_MachineHealthCalculated_InputMessage">
  <wsdl:part name="parameters" element="tns:MachineHealthCalculated"/>
</wsdl:message>
```

3.1.4.2.1.2 IPowerPivotUsageReportingService_MachineHealthCalculated_OutputMessage

The **IPowerPivotUsageReportingService_MachineHealthCalculated_OutputMessage** message is the response message for the MachineHealthCalculated operation.

The SOAP body contains a **MachineHealthCalculatedResponse** element.

```
  <wsdl:message
name="IPowerPivotUsageReportingService_MachineHealthCalculated_OutputMessage">
    <wsdl:part name="parameters" element="tns:MachineHealthCalculatedResponse"/>
  </wsdl:message>
```

3.1.4.2.2 Elements

The following table summarizes the XML Schema element definitions that are specific to the MachineHealthCalculated operation.

Element	Description
MachineHealthCalculated	A wrapped element that contains the input parameters.
MachineHealthCalculatedResponse	A wrapped element that contains the output parameters.

3.1.4.2.2.1 MachineHealthCalculated

The **MachineHealthCalculated** element specifies the relevant information that is requested to assess the health of the server that is running Analysis Services with which the client application is connected.

The following is the XML Schema definition of the **MachineHealthCalculated** element.

```
<xs:element name="MachineHealthCalculated">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" name="serverName" nillable="true" type="xs:string"/>
      <xs:element minOccurs="0" name="highMemoryLimitKB" type="xs:long"/>
      <xs:element minOccurs="0" name="lowMemoryLimitKB" type="xs:long"/>
      <xs:element minOccurs="0" name="memoryUsageKB" type="xs:long"/>
      <xs:element minOccurs="0" name="shrinkableMemoryKB" type="xs:long"/>
      <xs:element minOccurs="0" name="percentProcessorUtilization" type="xs:int"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

serverName: Specifies the name of the server that is running the Analysis Services instance to which the client application is connected.

highMemoryLimitKB: Specifies, in kilobytes, the high memory limit of the Analysis Services instance.

lowerMemoryLimitKB: Specifies, in kilobytes, the low memory limit of the Analysis Services instance.

memoryUsageKB: Specifies, in kilobytes, the amount of currently used memory.

shrinkableMemoryKB: Specifies, in kilobytes, the available shrink space of the Analysis Services instance.

percentProcessorUtilization: Specifies the percentage of CPU utilization of the server that is running the Analysis Services instance.

3.1.4.2.2.2 MachineHealthCalculatedResponse

The **MachineHealthCalculatedResponse** element holds the details of the response to the MachineHealthCalculated operation. This operation includes no response and this result can be safely ignored.

The following is the XML Schema definition of the **MachineHealthCalculatedResponse** element.

```
<xs:element name="MachineHealthCalculatedResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
```

3.1.4.3 Load

The **Load** operation is called when the client application begins to load an Analysis Services model onto a server that is running Analysis Services. This operation includes all information that is necessary to identify which model was loaded onto which server.

The following is the WSDL definition of the **Load** operation.

```
  <wsdl:operation name="Load">
    <wsdl:input
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/Load" message="tns:IPowerPivotUsageReportingService_Load_InputMessage"/>
    <wsdl:output
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/LoadResponse"
message="tns:IPowerPivotUsageReportingService_Load_OutputMessage"/>
  </wsdl:operation>
```

3.1.4.3.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to the Load operation.

Message	Description
IPowerPivotUsageReportingService_Load_InputMessage	The message from the client application to the server.
IPowerPivotUsageReportingService_Load_OutputMessage	The response message from the server to the client application after the load is logged.

3.1.4.3.1.1 IPowerPivotUsageReportingService_Load_InputMessage

The **IPowerPivotUsageReportingService_Load_InputMessage** message is the request message for the **Load** operation.

The SOAP body contains a **Load** element.

```
<wsdl:message name="IPowerPivotUsageReportingService_Load_InputMessage">
  <wsdl:part name="parameters" element="tns:Load"/>
</wsdl:message>
```

3.1.4.3.1.2 IPowerPivotUsageReportingService_Load_OutputMessage

The **IPowerPivotUsageReportingService_Load_OutputMessage** message is the response message for the Load operation.

The SOAP body contains a **LoadResponse** element.

```
<wsdl:message name="IPowerPivotUsageReportingService_Load_OutputMessage">
  <wsdl:part name="parameters" element="tns:LoadResponse"/>
</wsdl:message>
```

3.1.4.3.2 Elements

The following table summarizes the XML Schema element definitions that are specific to the Load operation.

Element	Description
Load	The wrapping element that contains the input parameters.
LoadResponse	The wrapping element that contains the output parameters.

3.1.4.3.2.1 Load

The **Load** element specifies the information that describes the model that has been loaded, along with its source and the server onto which it was loaded.

The following is the XML Schema definition of the **Load** element.

```
<xs:element name="Load">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" name="serverName" nillable="true" type="xs:string"/>
      <xs:element minOccurs="0" name="databaseName" nillable="true" type="xs:string"/>
      <xs:element minOccurs="0" name="imageUrl" nillable="true" type="xs:string"/>
      <xs:element minOccurs="0" name="imageID" nillable="true" type="xs:string"/>
      <xs:element minOccurs="0" name="fileVersion" type="xs:int"/>
      <xs:element minOccurs="0" name="userName" nillable="true" type="xs:string"/>
      <xs:element minOccurs="0" name="imageSize" type="xs:int"/>
      <xs:element minOccurs="0" name="dbSize" type="xs:int"/>
      <xs:element minOccurs="0" name="healthScore" type="xs:int"/>
      <xs:element minOccurs="0" name="elapsedTime" type="xs:int"/>
    </xs:sequence>
  </xs:complexType>
```

```
</xs:element>
```

serverName: Specifies the names of the server that is running Analysis Services and of the Analysis Services instance onto which the model has been loaded. These names MUST be in the format [*Server Name*]\[*Instance Name*].

databaseName: Specifies the name of the database on the server that is running the Analysis Services instance onto which the model is loaded.

imageUrl: Specifies the original URL of the document from which the model comes.

imageID: Specifies the identifier of the model.

fileVersion: Specifies the integer that represents the version of the model file.

userName: Specifies the user who requests that the model be loaded or whose action causes the loading.

imageSize: Specifies, in kilobytes, how big the model is prior to the model being restored.

dbSize: Specifies, in kilobytes, how big the database is after the database is restored.

healthScore: Specifies the health score of the server, after the model has completed loading onto the server.

elapsedTime: Specifies, in milliseconds, the length of time it took to load the model onto the server.

3.1.4.3.2.2 LoadResponse

The **LoadResponse** element holds the server's response to a Load operation.

The following is the XML Schema definition of the **LoadResponse** element.

```
<xs:element name="LoadResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" name="LoadResult" type="serialization:guid" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

LoadResult: Specifies the GUID that represents the model handle of the loaded model.

3.1.4.4 Connect

The **Connect** operation is called when the client application notifies the server that a connection is being established to a previously loaded model.

The following is the WSDL definition of the **Connect** operation.

```
  <wsdl:operation name="Connect">
    <wsdl:input
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/IPowerPivotUsageReportingService/Connect"
message="tns:IPowerPivotUsageReportingService_Connect_InputMessage"/>
    <wsdl:output
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/IPowerPivotUsageReportingService/ConnectResponse"
message="tns:IPowerPivotUsageReportingService_Connect_OutputMessage"/>
```

```
</wsdl:operation>
```

3.1.4.4.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to the Connect operation.

Message	Description
IPowerPivotUsageReportingService_Connect_InputMessage	The message from the client application that details the new connection that is to be made to a loaded model.
IPowerPivotUsageReportingService_Connect_OutputMessage	The server's response message that is sent to the client application after the connection is logged.

3.1.4.4.1.1 IPowerPivotUsageReportingService_Connect_InputMessage

The **IPowerPivotUsageReportingService_Connect_InputMessage** message is the request message for the **Connect** operation.

The SOAP body contains a **Connect** element.

```
<wsdl:message name="IPowerPivotUsageReportingService_Connect_InputMessage">
  <wsdl:part name="parameters" element="tns:Connect"/>
</wsdl:message>
```

3.1.4.4.1.2 IPowerPivotUsageReportingService_Connect_OutputMessage

The **IPowerPivotUsageReportingService_Connect_OutputMessage** message is the response message for the Connect operation.

The SOAP body contains a **ConnectResponse** element.

```
<wsdl:message name="IPowerPivotUsageReportingService_Connect_OutputMessage">
  <wsdl:part name="parameters" element="tns:ConnectResponse"/>
</wsdl:message>
```

3.1.4.4.2 Elements

The following table summarizes the XML Schema element definitions that are specific to the Connect operation.

Element	Description
Connect	The wrapping element that contains the connection properties.
ConnectResponse	The wrapping element that contains the ~~server's~~server's response and is a holder for the connection handle.

3.1.4.4.2.1 Connect

The **Connect** element specifies the information in the request that is required to describe the client application that connects to a model.

The following is the XML Schema definition of the **Connect** element.

```
<xs:element name="Connect">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" name="modelHandle" type="serialization:guid" />
      <xs:element minOccurs="0" name="username" nillable="true" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

modelHandle: Specifies the loaded model handle to which the client application connects to execute queries.

username: Specifies the user who requests to connect to the model. This user can differ from the user who requests to load the model.

3.1.4.4.2.2 ConnectResponse

The **ConnectResponse** element holds the server's response to a Connect operation.

The following is the XML Schema definition of the **ConnectResponse** element.

```
<xs:element name="ConnectResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" name="ConnectResult" type="serialization:guid" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

ConnectResult: Specifies the connection handle.

3.1.4.5 RequestComplete

The **RequestComplete** operation is called after the client application completes each unique query against the loaded model connection.

The following is the WSDL definition of the **RequestComplete** operation.

```
  <wsdl:operation name="RequestComplete">
    <wsdl:input
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/IPowerPivotUsageReportingService/RequestComplete"
message="tns:IPowerPivotUsageReportingService_RequestComplete_InputMessage"/>
    <wsdl:output
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/IPowerPivotUsageReportingService/RequestCompleteResponse"
message="tns:IPowerPivotUsageReportingService_RequestComplete_OutputMessage"/>
  </wsdl:operation>
```

3.1.4.5.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to the RequestComplete operation.

Message	Description
IPowerPivotUsageReportingService_RequestComplete_InputMessage	The message from the client application that details the request against a connection is completed.
IPowerPivotUsageReportingService_RequestComplete_OutputMessage	The message response from the server that is sent after the request completion is logged.

3.1.4.5.1.1 IPowerPivotUsageReportingService_RequestComplete_InputMessage

The **IPowerPivotUsageReportingService_RequestComplete_InputMessage** message is the request message for the **RequestComplete** operation.

The SOAP body contains a **RequestComplete** element.

```
<wsdl:message name="IPowerPivotUsageReportingService_RequestComplete_InputMessage">
  <wsdl:part name="parameters" element="tns:RequestComplete"/>
</wsdl:message>
```

3.1.4.5.1.2 IPowerPivotUsageReportingService_RequestComplete_OutputMessage

The **IPowerPivotUsageReportingService_RequestComplete_OutputMessage** message is the response message for the RequestComplete operation.

The SOAP body contains a **RequestCompleteResponse** element.

```
<wsdl:message name="IPowerPivotUsageReportingService_RequestComplete_OutputMessage">
  <wsdl:part name="parameters" element="tns:RequestCompleteResponse"/>
</wsdl:message>
```

3.1.4.5.2 Elements

The following table summarizes the XML Schema element definitions that are specific to the RequestComplete operation.

Element	Description
RequestComplete	The wrapping element for the parameters that provide the details about the request that the client application completed on the connection.
RequestCompleteResponse	The wrapping element that contains the ~~server's~~server's response that is sent after the response of the **RequestComplete** operation is logged.

3.1.4.5.2.1 RequestComplete

The **RequestComplete** element specifies the information that is required to describe the request that is completed against a model.

The following is the XML Schema definition of the **RequestComplete** element.

```
<xs:element name="RequestComplete">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" name="connectionHandle" type="serialization:guid" />
      <xs:element minOccurs="0" name="reqType" type="tns:RequestType" />
      <xs:element minOccurs="0" name="elapsedTime" type="xs:int"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

connectionHandle: Specifies the identifier of the connection handle against which the request is completed.

reqType: Specifies a **RequestType** object that represents the type of request that is to be executed.

elapsedTime: Specifies, in milliseconds, an integer that represents the length of the request.

3.1.4.5.2.2 RequestCompleteResponse

The **RequestCompleteResponse** element holds the server's response to a RequestComplete operation. This element is always an empty response and can be safely ignored.

The following is the XML Schema definition of the **RequestCompleteResponse** element.

```
<xs:element name="RequestCompleteResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
```

3.1.4.5.3 Simple Types

The following table summarizes the XML Schema simple definitions that are specific to the RequestComplete operation.

Simple type	Description
RequestType	An enumeration type that represents the type of request made against the connection.

3.1.4.5.3.1 RequestType

The **RequestType** simple type specifies a string with a predefined set of values that represents the type of request that is made against the connection.

The following is the XML Schema definition of the **RequestType** simple type.

```
<xs:simpleType name="RequestType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Execute" />
    <xs:enumeration value="Discover" />
    <xs:enumeration value="Process" />
  </xs:restriction>
</xs:simpleType>
```

Value	Meaning
Execute	Begin executing a request against the connection.
Discover	Find information about the connection.
Process	Process a request against the connection.

3.1.4.6 Unload

The **Unload** operation is called when the client application no longer needs a specified loaded model and removes it from the server that is running an Analysis Services instance.

The following is the WSDL definition of the **Unload** operation.

```
    <wsdl:input
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/IPowerPivotUsageReportingService/Unload"
message="tns:IPowerPivotUsageReportingService_Unload_InputMessage"/>
    <wsdl:output
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/IPowerPivotUsageReportingService/UnloadResponse"
message="tns:IPowerPivotUsageReportingService_Unload_OutputMessage"/>
  </wsdl:operation>
```

3.1.4.6.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to the Unload operation.

Message	Description
IPowerPivotUsageReportingService_Unload_InputMessage	The message from the client application that provides the details about the model that has been unloaded.
IPowerPivotUsageReportingService_Unload_OutputMessage	The message response from the server that is sent to the client application after the Unload operation is logged.

3.1.4.6.1.1 IPowerPivotUsageReportingService_Unload_InputMessage

The **IPowerPivotUsageReportingService_Unload_InputMessage** message is the request message for the **Unload** operation.

The SOAP body contains an **Unload** element.

```
<wsdl:message name="IPowerPivotUsageReportingService_Unload_InputMessage">
  <wsdl:part name="parameters" element="tns:Unload"/>
</wsdl:message>
```

3.1.4.6.1.2 IPowerPivotUsageReportingService_Unload_OutputMessage

The **IPowerPivotUsageReportingService_Unload_OutputMessage** message is the response message for the Unload operation.

The SOAP body contains an **UnloadResponse** element.

```
<wsdl:message name="IPowerPivotUsageReportingService_Unload_OutputMessage">
  <wsdl:part name="parameters" element="tns:UnloadResponse"/>
</wsdl:message>
```

3.1.4.6.2 Elements

The following table summarizes the XML Schema element definitions that are specific to the Unload operation.

Element	Description
Unload	The wrapping element for the parameters that provide the details about the model that is no longer loaded.
UnloadResponse	The wrapping element that contains the ~~server's~~server's response that is sent after the **Unload** operation is logged.

3.1.4.6.2.1 Unload

The **Unload** element specifies the information that is required to describe the model that has been unloaded from the server that is running the Analysis Services instance.

The following is the XML Schema definition of the **Unload** element.

```
<xs:element name="Unload">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" name="modelHandle" type="serialization:guid" />
      <xs:element minOccurs="0" name="elapsedTime" type="xs:int"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

modelHandle: Specifies the identifier that is returned from the **Load** operation and that indicates the model that has been unloaded from the server that is running the Analysis Services instance.

elapsedTime: Specifies, in milliseconds, the amount of time that it took to unload the model.

3.1.4.6.2.2 UnloadResponse

The **UnloadResponse** element holds the server's response to the Unload operation. This element is always empty and can be safely ignored.

The following is the XML Schema definition of the **UnloadResponse** element.

```
<xs:element name="UnloadResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
```

3.1.4.7 UnloadAbandoned

This **UnloadAbandoned** operation is called by the client application when it is unable to unload a model from a server that is running an Analysis Services instance and is abandoning the attempt. This operation is optional.

The following is the WSDL definition of the **UnloadAbandoned** operation.

```
  <wsdl:operation name="UnloadAbandoned">
    <wsdl:input
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/IPowerPivotUsageReportingService/UnloadAbandoned"
message="tns:IPowerPivotUsageReportingService_UnloadAbandoned_InputMessage"/>
    <wsdl:output
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/IPowerPivotUsageReportingService/UnloadAbandonedResponse"
message="tns:IPowerPivotUsageReportingService_UnloadAbandoned_OutputMessage"/>
  </wsdl:operation>
```

3.1.4.7.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to the UnloadAbandoned operation.

Message	Description
IPowerPivotUsageReportingService_UnloadAbandoned_InputMessage	The message from the client application detailing that a model unload has been abandoned.
IPowerPivotUsageReportingService_UnloadAbandoned_OutputMessage	The message response from the server after the details of the abandoned unload are logged.

3.1.4.7.1.1 IPowerPivotUsageReportingService_UnloadAbandoned_InputMessage

The **IPowerPivotUsageReportingService_UnloadAbandoned_InputMessage** message is the request message for the **UnloadAbandoned** operation.

The SOAP body contains an **UnloadAbandoned** element.

```
<wsdl:message name="IPowerPivotUsageReportingService_UnloadAbandoned_InputMessage">
  <wsdl:part name="parameters" element="tns:UnloadAbandoned"/>
</wsdl:message>
```

3.1.4.7.1.2 IPowerPivotUsageReportingService_UnloadAbandoned_OutputMessage

The **IPowerPivotUsageReportingService_UnloadAbandoned_OutputMessage** message is the response message for the UnloadAbandoned operation.

The SOAP body contains an **UnloadAbandonedResponse** element.

```
<wsdl:message name="IPowerPivotUsageReportingService_UnloadAbandoned_OutputMessage">
  <wsdl:part name="parameters" element="tns:UnloadAbandonedResponse"/>
</wsdl:message>
```

3.1.4.7.2 Elements

The following table summarizes the XML Schema element definitions that are specific to the UnloadAbandoned operation.

Element	Description
UnloadAbandoned	The wrapping element for the parameters that provide the details about the server and model on which and **Unload** operation failed and is being abandoned.
UnloadAbandonedResponse	The wrapping element that contains the ~~server's~~server's response that is sent after the **UnloadAbandoned** operation is logged.

3.1.4.7.2.1 UnloadAbandoned

The **UnloadAbandoned** element specifies the information that is required to describe the **Unload** operation that is to be ignored.

The following is the XML Schema definition of the **UnloadAbandoned** element.

```
<xs:element name="UnloadAbandoned">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" name="serverName" nillable="true" type="xs:string"/>
      <xs:element minOccurs="0" name="databaseName" nillable="true" type="xs:string"/>
      <xs:element minOccurs="0" name="imageUrl" nillable="true" type="xs:string"/>
      <xs:element minOccurs="0" name="elapsedTime" type="xs:int"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

serverName: Specifies the name of the server on which an unload failed or timed out and is being abandoned.

databaseName: Specifies the name of the loaded database on which an unload was attempted.

imageUrl: Specifies the original URL of the file that contains the model.

elapsedTime: Specifies, in milliseconds, the amount of time that was spent unloading the database before it was abandoned.

3.1.4.7.2.2 UnloadAbandonedResponse

The **UnloadAbandonedResponse** element holds the server's response to the UnloadAbandoned operation. This element is always empty and can be safely ignored.

The following is the XML Schema definition of the **InloadAbandonedResponse** element.

```
<xs:element name="UnloadAbandonedResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
```

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

3.2 Client Details

The client side of this protocol is simply a pass-through. That is, no additional timers or other state is required on the client side of this protocol. Calls made by the higher-layer protocol or application are passed directly to the transport, and the results returned by the transport are passed directly back to the higher-layer protocol or application.

4 Protocol Examples

The following examples form a complete usage reporting conversation when performed in the order shown in this section.

4.1 Load

The client informs the server that it has loaded a model from a specified URL onto the specified server that is running Analysis Services.

```
<soap:Body>
  <Load xmlns="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting">
    <serverName>SSAS_SERVER\POWERPIVOT</serverName>
      <databaseName>WORKBOOK_9dd1615f48bb4e73b82a8d4045099671_614c55d236e74e959f7
               398743a12e8d0_SSPM</databaseName>
      <imageUrl>http://SERVER/BI Gallery/WORKBOOK.xlsx</imageUrl>
      <imageID>1bb035af-4ca2-4941-bd3d-9fe13b66843a</imageID>
      <fileVersion>0</fileVersion>
      <userName>i:0#.w|DOMAIN\USER</userName>
      <imageSize>6647808</imageSize>
      <dbSize>6647808</dbSize>
      <healthScore>0</healthScore>
      <elapsedTime>2</elapsedTime>
  </Load>
</soap:Body>
```

4.2 LoadResponse

The server responds with the model handle that it has assigned.

```
<s:Body>
  <LoadResponse xmlns="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting">
    <LoadResult>1436f40a-55a2-4292-be98-cbdda91ef4d1</LoadResult>
  </LoadResponse>
</s:Body>
```

4.3 Connect

The client informs the server that it is now establishing a connection to a loaded model.

```
<soap:Body>
  <Connect xmlns="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting">
    <modelHandle>1436f40a-55a2-4292-be98-cbdda91ef4d1</modelHandle>
    <username>i:0#.w|DOMAIN\USER</username>
  </Connect>
</soap:Body>
```

4.4 ConnectResponse

The server responds with the connection handle it has assigned to this connection.

```
<s:Body>
  <ConnectResponse
          xmlns="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting">
    <ConnectResult>cac11abb-fa08-41eb-89e1-f47ffe19d03a</ConnectResult>
  </ConnectResponse>
```

```
</s:Body>
```

4.5 RequestComplete

```
<soap:Body>
  <RequestComplete
          xmlns="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting">
    <connectionHandle>cac11abb-fa08-41eb-89e1-f47ffe19d03a</connectionHandle>
    <reqType>Execute</reqType>
    <elapsedTime>5</elapsedTime>
  </RequestComplete>
</soap:Body>
```

4.6 RequestCompleteResponse

The server responds with an empty acknowledgement.

```
<s:Body>
  <RequestCompleteResponse
          xmlns="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting"/>
</s:Body>
```

5 Security

5.1 Security Considerations for Implementers

This protocol does not depend on any specific authentication mechanisms. Security authentication/authorization is expected to take place at a lower layer of the protocol stack. Authentication can be Claims, Windows, SharePoint Service-to-Service (S2S), or any other supported protocol, but authentication is dependent on the server that hosts the implementing service.

It is recommended that a security mechanism appropriate for the situation is chosen, such as ensuring that HTTPS with a form of authentication or whatever authentication mechanism the service is configured with, is chosen.

5.2 Index of Security Parameters

None.

6 Appendix A: Full WSDL

For ease of implementation, the full WSDL and schema are provided in this appendix.

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:serialization="http://schemas.microsoft.com/2003/10/Serialization/"

xmlns:tns="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting"
                  xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl"
                  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
                  name="PowerPivotUsageReportingService"
targetNamespace="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting">
    <wsdl:types>
        <xs:schema elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting">
            <xs:import namespace="http://schemas.microsoft.com/2003/10/Serialization/" />
            <xs:element name="Connect">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element minOccurs="0" name="modelHandle"
type="serialization:guid" />
                        <xs:element minOccurs="0" name="username" nillable="true"
type="xs:string" />
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="ConnectResponse">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element minOccurs="0" name="ConnectResult"
type="serialization:guid" />
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="IsAvailable">
                <xs:complexType>
                    <xs:sequence />
                </xs:complexType>
            </xs:element>
            <xs:element name="IsAvailableResponse">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element minOccurs="0" name="IsAvailableResult" type="xs:boolean"
/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="Load">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element minOccurs="0" name="serverName" nillable="true"
type="xs:string" />
                        <xs:element minOccurs="0" name="databaseName" nillable="true"
type="xs:string" />
                        <xs:element minOccurs="0" name="imageUrl" nillable="true"
type="xs:string" />
                        <xs:element minOccurs="0" name="imageID" nillable="true"
type="xs:string" />
                        <xs:element minOccurs="0" name="fileVersion" type="xs:int" />
                        <xs:element minOccurs="0" name="userName" nillable="true"
type="xs:string" />
                        <xs:element minOccurs="0" name="imageSize" type="xs:int" />
                        <xs:element minOccurs="0" name="dbSize" type="xs:int" />
                        <xs:element minOccurs="0" name="healthScore" type="xs:int" />
                        <xs:element minOccurs="0" name="elapsedTime" type="xs:int" />
                    </xs:sequence>
```

```
                </xs:complexType>
            </xs:element>
            <xs:element name="LoadResponse">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element minOccurs="0" name="LoadResult" type="serialization:guid"
/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="MachineHealthCalculated">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element minOccurs="0" name="serverName" nillable="true"
type="xs:string" />
                        <xs:element minOccurs="0" name="highMemoryLimitKB" type="xs:long" />
                        <xs:element minOccurs="0" name="lowMemoryLimitKB" type="xs:long" />
                        <xs:element minOccurs="0" name="memoryUsageKB" type="xs:long" />
                        <xs:element minOccurs="0" name="shrinkableMemoryKB" type="xs:long" />
                        <xs:element minOccurs="0" name="percentProcessorUtilization"
type="xs:int" />
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="MachineHealthCalculatedResponse">
                <xs:complexType>
                    <xs:sequence />
                </xs:complexType>
            </xs:element>
            <xs:element name="RequestComplete">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element minOccurs="0" name="connectionHandle"
type="serialization:guid" />
                        <xs:element minOccurs="0" name="reqType" type="tns:RequestType" />
                        <xs:element minOccurs="0" name="elapsedTime" type="xs:int" />
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:simpleType name="RequestType">
                <xs:restriction base="xs:string">
                    <xs:enumeration value="Execute" />
                    <xs:enumeration value="Discover" />
                    <xs:enumeration value="Process" />
                </xs:restriction>
            </xs:simpleType>
            <xs:element name="RequestType" nillable="true" type="tns:RequestType" />
            <xs:element name="RequestCompleteResponse">
                <xs:complexType>
                    <xs:sequence />
                </xs:complexType>
            </xs:element>
            <xs:element name="Unload">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element minOccurs="0" name="modelHandle"
type="serialization:guid" />
                        <xs:element minOccurs="0" name="elapsedTime" type="xs:int" />
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="UnloadResponse">
                <xs:complexType>
                    <xs:sequence />
                </xs:complexType>
            </xs:element>
            <xs:element name="UnloadAbandoned">
                <xs:complexType>
                    <xs:sequence>
```

```
                        <xs:element minOccurs="0" name="serverName" nillable="true"
type="xs:string" />
                        <xs:element minOccurs="0" name="databaseName" nillable="true"
type="xs:string" />
                        <xs:element minOccurs="0" name="imageUrl" nillable="true"
type="xs:string" />
                        <xs:element minOccurs="0" name="elapsedTime" type="xs:int" />
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="UnloadAbandonedResponse">
                <xs:complexType>
                    <xs:sequence />
                </xs:complexType>
            </xs:element>
        </xs:schema>
        <xs:schema attributeFormDefault="qualified" elementFormDefault="qualified"
                targetNamespace="http://schemas.microsoft.com/2003/10/Serialization/">
            <xs:element name="anyType" nillable="true" type="xs:anyType" />
            <xs:element name="anyURI" nillable="true" type="xs:anyURI" />
            <xs:element name="base64Binary" nillable="true" type="xs:base64Binary" />
            <xs:element name="boolean" nillable="true" type="xs:boolean" />
            <xs:element name="byte" nillable="true" type="xs:byte" />
            <xs:element name="dateTime" nillable="true" type="xs:dateTime" />
            <xs:element name="decimal" nillable="true" type="xs:decimal" />
            <xs:element name="double" nillable="true" type="xs:double" />
            <xs:element name="float" nillable="true" type="xs:float" />
            <xs:element name="int" nillable="true" type="xs:int" />
            <xs:element name="long" nillable="true" type="xs:long" />
            <xs:element name="QName" nillable="true" type="xs:QName" />
            <xs:element name="short" nillable="true" type="xs:short" />
            <xs:element name="string" nillable="true" type="xs:string" />
            <xs:element name="unsignedByte" nillable="true" type="xs:unsignedByte" />
            <xs:element name="unsignedInt" nillable="true" type="xs:unsignedInt" />
            <xs:element name="unsignedLong" nillable="true" type="xs:unsignedLong" />
            <xs:element name="unsignedShort" nillable="true" type="xs:unsignedShort" />
            <xs:element name="char" nillable="true" type="tns:char" />
            <xs:simpleType name="char">
                <xs:restriction base="xs:int" />
            </xs:simpleType>
            <xs:element name="duration" nillable="true" type="tns:duration" />
            <xs:simpleType name="duration">
                <xs:restriction base="xs:duration">
                    <xs:pattern value="\-?P(\d*D)?(T(\d*H)?(\d*M)?(\d*(\.\d*)?S)?)?" />
                    <xs:minInclusive value="-P10675199DT2H48M5.4775808S" />
                    <xs:maxInclusive value="P10675199DT2H48M5.4775807S" />
                </xs:restriction>
            </xs:simpleType>
            <xs:element name="guid" nillable="true" type="tns:guid" />
            <xs:simpleType name="guid">
                <xs:restriction base="xs:string">
                    <xs:pattern value="[\da-fA-F]{8}-[\da-fA-F]{4}-[\da-fA-F]{4}-[\da-fA-
F]{4}-[\da-fA-F]{12}" />
                </xs:restriction>
            </xs:simpleType>
            <xs:attribute name="FactoryType" type="xs:QName" />
            <xs:attribute name="Id" type="xs:ID" />
            <xs:attribute name="Ref" type="xs:IDREF" />
        </xs:schema>
    </wsdl:types>
    <wsdl:message name="IPowerPivotUsageReportingService_Connect_InputMessage">
        <wsdl:part name="parameters" element="tns:Connect" />
    </wsdl:message>
    <wsdl:message name="IPowerPivotUsageReportingService_Connect_OutputMessage">
        <wsdl:part name="parameters" element="tns:ConnectResponse" />
    </wsdl:message>
    <wsdl:message name="IPowerPivotUsageReportingService_IsAvailable_InputMessage">
        <wsdl:part name="parameters" element="tns:IsAvailable" />
    </wsdl:message>
    <wsdl:message name="IPowerPivotUsageReportingService_IsAvailable_OutputMessage">
```

```
        <wsdl:part name="parameters" element="tns:IsAvailableResponse" />
    </wsdl:message>
    <wsdl:message name="IPowerPivotUsageReportingService_Load_InputMessage">
        <wsdl:part name="parameters" element="tns:Load" />
    </wsdl:message>
    <wsdl:message name="IPowerPivotUsageReportingService_Load_OutputMessage">
        <wsdl:part name="parameters" element="tns:LoadResponse" />
    </wsdl:message>
    <wsdl:message
name="IPowerPivotUsageReportingService_MachineHealthCalculated_InputMessage">
        <wsdl:part name="parameters" element="tns:MachineHealthCalculated" />
    </wsdl:message>
    <wsdl:message
name="IPowerPivotUsageReportingService_MachineHealthCalculated_OutputMessage">
        <wsdl:part name="parameters" element="tns:MachineHealthCalculatedResponse" />
    </wsdl:message>
    <wsdl:message name="IPowerPivotUsageReportingService_RequestComplete_InputMessage">
        <wsdl:part name="parameters" element="tns:RequestComplete" />
    </wsdl:message>
    <wsdl:message name="IPowerPivotUsageReportingService_RequestComplete_OutputMessage">
        <wsdl:part name="parameters" element="tns:RequestCompleteResponse" />
    </wsdl:message>
    <wsdl:message name="IPowerPivotUsageReportingService_Unload_InputMessage">
        <wsdl:part name="parameters" element="tns:Unload" />
    </wsdl:message>
    <wsdl:message name="IPowerPivotUsageReportingService_Unload_OutputMessage">
        <wsdl:part name="parameters" element="tns:UnloadResponse" />
    </wsdl:message>
    <wsdl:message name="IPowerPivotUsageReportingService_UnloadAbandoned_InputMessage">
        <wsdl:part name="parameters" element="tns:UnloadAbandoned" />
    </wsdl:message>
    <wsdl:message name="IPowerPivotUsageReportingService_UnloadAbandoned_OutputMessage">
        <wsdl:part name="parameters" element="tns:UnloadAbandonedResponse" />
    </wsdl:message>
    <wsdl:portType name="IPowerPivotUsageReportingService">
        <wsdl:operation name="Connect">
            <wsdl:input
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/Connect"
message="tns:IPowerPivotUsageReportingService_Connect_InputMessage" />
            <wsdl:output
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/ConnectResponse"
message="tns:IPowerPivotUsageReportingService_Connect_OutputMessage" />
        </wsdl:operation>
        <wsdl:operation name="IsAvailable">
            <wsdl:input
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/IsAvailable"
message="tns:IPowerPivotUsageReportingService_IsAvailable_InputMessage" />
            <wsdl:output
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/IsAvailableResponse"
message="tns:IPowerPivotUsageReportingService_IsAvailable_OutputMessage" />
        </wsdl:operation>
        <wsdl:operation name="Load">
            <wsdl:input
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/Load" message="tns:IPowerPivotUsageReportingService_Load_InputMessage" />
            <wsdl:output
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/LoadResponse"
message="tns:IPowerPivotUsageReportingService_Load_OutputMessage" />
        </wsdl:operation>
        <wsdl:operation name="MachineHealthCalculated">
            <wsdl:input
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/MachineHealthCalculated"
message="tns:IPowerPivotUsageReportingService_MachineHealthCalculated_InputMessage" />
```

```
            <wsdl:output
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/MachineHealthCalculatedResponse"
message="tns:IPowerPivotUsageReportingService_MachineHealthCalculated_OutputMessage" />
        </wsdl:operation>
        <wsdl:operation name="RequestComplete">
            <wsdl:input
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/RequestComplete"
message="tns:IPowerPivotUsageReportingService_RequestComplete_InputMessage" />
            <wsdl:output
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/RequestCompleteResponse"
message="tns:IPowerPivotUsageReportingService_RequestComplete_OutputMessage" />
        </wsdl:operation>
        <wsdl:operation name="Unload">
            <wsdl:input
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/Unload" message="tns:IPowerPivotUsageReportingService_Unload_InputMessage"
/>
            <wsdl:output
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/UnloadResponse"
message="tns:IPowerPivotUsageReportingService_Unload_OutputMessage" />
        </wsdl:operation>
        <wsdl:operation name="UnloadAbandoned">
            <wsdl:input
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/UnloadAbandoned"
message="tns:IPowerPivotUsageReportingService_UnloadAbandoned_InputMessage" />
            <wsdl:output
wsaw:Action="http://schemas.microsoft.com/sqlserver/PowerPivot/UsageReporting/IPowerPivotUsag
eReportingService/UnloadAbandonedResponse"
message="tns:IPowerPivotUsageReportingService_UnloadAbandoned_OutputMessage" />
        </wsdl:operation>
    </wsdl:portType>
</wsdl:definitions>
```

7 Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs.

- Office Online Server
- Microsoft SQL Server 2016

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms "SHOULD" or "SHOULD NOT" implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term "MAY" implies that the product does not follow the prescription.

8 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

9 Index

N

O

P

R

S

T

V

W

X