[MS-TPXS]:

Telemetry Protocol XML Schema

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **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 make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
- **Patents**. Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks**. The names of companies and products contained in this documentation may 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 addresses, logos, people, places, and events 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 specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do 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 are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Revision Summary

Date	Revision History	Revision Class	Comments
12/16/2011	1.0	New	Released new document.
3/30/2012	1.0	None	No changes to the meaning, language, or formatting of the technical content.
7/12/2012	1.0	None	No changes to the meaning, language, or formatting of the technical content.
10/25/2012	1.0	None	No changes to the meaning, language, or formatting of the technical content.
1/31/2013	1.0	None	No changes to the meaning, language, or formatting of the technical content.
8/8/2013	2.0	Major	Significantly changed the technical content.
11/14/2013	2.0	None	No changes to the meaning, language, or formatting of the technical content.
2/13/2014	2.0	None	No changes to the meaning, language, or formatting of the technical content.
5/15/2014	2.0	None	No changes to the meaning, language, or formatting of the technical content.
6/30/2015	2.0	No Change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

The Telemetry Protocol XML Schema defines the message structure used by the Software Quality Metrics (SQM) Client to Service Protocol, specified in [MS-SQMCS2]. The schema is used to send software instrumentation metrics from a client to the SQM service and for the client to download client-specific control data.

Sections 1.7 and 2 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in [RFC2119]. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are specific to this document:

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS): An extension of **HTTP** that securely encrypts and decrypts webpage requests.

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data-a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication (2) using X.509 certificates (2). For more information, see [X509]. The SSL protocol is precursor to Transport Layer Security (TLS). The TLS version 1.0 specification is based on SSL version 3.0.

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

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, http://www.rfc-editor.org/rfc/rfc2616.txt

1.2.2 Informative References

[MS-SQMCS2] Microsoft Corporation, "Software Quality Metrics (SQM) Client-to-Service Version 2 Protocol".

1.3 Overview

This XML data structure specifies a client-to-server request message and a server-to-client response message. The client request message contains one or more commands specifying the work the client is requesting of the server. The server response message contains one response to each request (command) from the client.

1.4 Relationship to Protocols and Other Structures

The Telemetry Protocol XML Schema is used by the SQM Client-to-Service Protocol, as specified in [MS-SQMCS2], to transmit request and response messages between a client and the SQM service.

The request and response data is transmitted over **Hypertext Transfer Protocol (HTTP)** and **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)** as specified in [RFC2616].

1.5 Applicability Statement

This structure is defined to support clients that are enabled to request and send telemetry data to a telemetry server. The structure describes the client sender and the data being sent so the server can evaluate each request and either accept or reject the request based on the data contained within the message.

1.6 Versioning and Localization

This specification documents Version 2 of the Telemetry Protocol XML Schema. The XML schema does not support localization.

1.7 Vendor-Extensible Fields

There are no vendor-extensible fields. It is possible for vendors to add key-value arguments to **arg** elements (see the schemas in section 5). However, these key-value pairs are not interpreted unless they are used by the vendor or identified by specific contract between the client and server.

2 Structures

The telemetry data structure describes a telemetry client request and server response.

2.1 Telemetry Request Message

The telemetry request message describes the client request from the server. Each element is described below. The complete schema is specified in the section 5.1.

2.1.1 req

The **req** (request) element is the topmost element of a client-to-server request message. It contains an associated version number that is used to differentiate schema changes in the telemetry request message format. The version number of this schema is 2.

```
<xs:element name="req">
 <xs:complexType>
  <xs:attribute name="ver" type="xs:unsignedInt" use="required" />
 </xs:complexType>
</xs:element>
```

2.1.1.1 tlm

The **tlm** (telemetry) element is the namespace of the requested service. The **tlm** element is a child element of req.

```
<xs:element name="tlm">
 <xs:complexType>
 </xs:complexType>
</xs:element>
```

2.1.1.1.1 src

The **src** (source) element is a child element of tlm**.** The **src** element and the child elements **desc** (description) and **mach** (machine) describe the client that is making the request.

```
<xs:element name="src" minOccurs="1" maxOccurs="1">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="desc" minOccurs="1" maxOccurs="1">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="mach" minOccurs="1" maxOccurs="1">
       <xs:complexType>
        <xs:sequence>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

2.1.1.1.1.1 os

The **os** is a child element of the **mach** element. It describes the client operating system with a set of key-value pair descriptive attributes.

```
<xs:element name="os" minOccurs="1" maxOccurs="1">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="arg" minOccurs="0" maxOccurs="unbounded" >
    <xs:complexType>
     <xs:attribute name="nm" type="xs:string" use="required" />
     <xs:attribute name="val" type="xs:string" use="required" />
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

The **os** element has a set of child **arg** elements with name-value (**nm**,**val**) attribute pairs describing the operating system.

nm: Unique key attribute describing a dimension of the client operating system.

val: Value field of the key-value pair describing a dimension of the operating system.

2.1.1.1.1.2 hw

The **hw** (hardware) element is a child element of the **mach** element. It describes the client machine hardware with a set of key-value pair descriptive attributes.

```
<xs:element name="hw" minOccurs="1" maxOccurs="1">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="arg" minOccurs="0" maxOccurs="unbounded" >
    <xs:complexType>
     <xs:attribute name="nm" type="xs:string" use="required" />
     <xs:attribute name="val" type="xs:string" use="required" />
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

The **hw** element has a set of child **arg** elements with name-value (**nm**,**val**) attribute pairs describing the hardware platform.

nm: Unique key attribute describing a dimension of the client hardware platform.

val: Value field of the key-value pair describing a dimension of the client hardware platform.

2.1.1.1.1.3 ctrl

The **ctrl** (control) element is a child element of the **mach** element. It describes a set of client control values with a set of key-value pair descriptive attributes.

```
<xs:element name="ctrl" minOccurs="1" maxOccurs="1">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="arg" minOccurs="0" maxOccurs="unbounded" >
    <xs:complexType>
     <xs:attribute name="nm" type="xs:string" use="required" />
     <xs:attribute name="val" type="xs:string" use="required" />
```

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

The **ctrl** element has a set of child **arg** elements with name-value (**nm**,**val**) attribute pairs describing the client control values.

nm: Unique key attribute describing a dimension of the client control values.

val: Value field of the key-value pair describing a dimension of the client control values.

2.1.1.1.2 reqs

The **reqs** (requests) element is the client request section of the message. **reqs** is a child element of tlm.

```
<xs:element name="reqs" minOccurs="1" maxOccurs="1">
 <xs:complexType>
  <xs:sequence>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

2.1.1.1.2.1 payload

The **payload** element is a child element of **reqs**. It describes the binary data (if any).

```
<xs:element name="payload" minOccurs="0" maxOccurs="1">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
     <xs:attribute name="nm" type="xs:string" use="required" />
     <xs:attribute name="val" type="xs:string" use="required" />
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

The **payload** element has a set of child **arg** elements with name-value (**nm**,**val**) attribute pairs describing the type and size of the binary data payload.

nm: Unique key attribute describing a dimension of the payload.

val: Value field of the key-value pair describing a dimension of the payload.

2.1.1.1.2.2 req

The **req** (request) element is a child element of reqs. It describes a client request of the server. The attribute name is **key** and represents a message-wide unique key associated with the **req** element.

```
<xs:element nm="req" minOccurs="1" maxOccurs="unbounded">
 <xs:complexType>
  <xs:attribute nm="key" type="xs:string" use="required"/>
 </xs:complexType>
```

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

2.1.1.1.2.2.1 namespace

The **namespace** element is a child element of **req** and describes the specific client request environment in terms of a set of hierarchical namespaces. The namespace is order-defined as the 5tuple **root.svc.ptr.gp.app**. **tlm** is the implied value of the **root** component of the tuple.

```
<xs:element name="namespace" minOccurs="1" maxOccurs="1">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
     <xs:attribute name="nm" type="xs:string" use="required" />
     <xs:attribute name="val" type="xs:string" use="required" />
    </xs:complexType>
   </xs:element>
  </xs:sequence>
  <xs:attribute name="svc" type="xs:string" use="required" />
  <xs:attribute name="ptr" type="xs:string" use="required" />
  <xs:attribute name="gp" type="xs:string" use="required" />
  <xs:attribute name="app" type="xs:string" use="required" />
 </xs:complexType>
</xs:element>
```

The **namespace** element has a set of child **arg** elements with name-value (**nm**,**val**) attribute pairs that serve as additional namespace descriptors.

nm: Unique key attribute describing a dimension of the namespace.

val: Value field of the key-value pair describing a dimension of the namespace.

The **namespace** element has the following attributes:

svc: An attribute that describes the client service sending the request.

ptr: An attribute that describes the client service partner sending the request. **ptr** differentiates the service namespace (**root.svc.ptr**).

gp: An attribute that describes the client service partner group sending the request. **gp** differentiates the service namespace (**root.svc.ptr.gp**).

app: An attribute that describes the client service partner group application sending the request. **app** further differentiates the group namespace (**root.svc.ptr.gp.app**).

2.1.1.1.2.2.2 ctrl

The **ctrl** (control) element is a child element of **req** and describes the control data values for the request with a set of name-value pairs. **ctrl** arguments are specific to the **req** element whereas the machine-level **ctrl** (see section 2.1.1.1.1.3) arguments are client machine-wide.

```
<xs:element name="ctrl" minOccurs="0" maxOccurs="1">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
     <xs:attribute name="nm" type="xs:string" use="required" />
     <xs:attribute name="val" type="xs:string" use="required" />
    </xs:complexType>
   </xs:element>
```

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

The **ctrl** element has a set of child **arg** elements with name-value (**nm**,**val**) attribute pairs describing the client request control values.

nm: Unique key attribute describing a dimension of the client request control values.

val: Value field of the key-value pair describing a dimension of the client request control values.

2.1.1.1.2.2.3 contents

The **contents** element is a child element of **req** and describes the payload content (if any).

```
<xs:element name="contents" minOccurs="0" maxOccurs="1">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
     <xs:attribute name="nm" type="xs:string" use="required" />
     <xs:attribute name="val" type="xs:string" use="required" />
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

The **contents** element has a set of child **arg** elements with name-value (**nm**,**val**) attribute pairs describing the contents of the client request payload.

nm: Name attribute describing a dimension of the payload contents. The name is not required to be unique.

val: Value field of the name-value pair describing a dimension of the payload contents.

2.1.1.1.2.2.4 cmd

The **cmd** (command) element is a child element of the **req** element. It describes the client request work from the server. The **cmd** element has one attribute **nm**, a name value describing the command. The server uses the name to determine what work is requested.

```
<xs:element name="cmd" minOccurs="1" maxOccurs="1">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
     <xs:attribute name="nm" type="xs:string" use="required" />
     <xs:attribute name="val" type="xs:string" use="required" />
    </xs:complexType>
   </xs:element>
  </xs:sequence>
  <xs:attribute name="nm" type="xs:string" use="required" />
 </xs:complexType>
</xs:element>
```

The **cmd** element has a set of child **arg** elements with name-value (**nm**,**val**) attribute pairs describing the command arguments.

nm: Unique key attribute describing an argument of the command.

val: Value field of the key-value pair describing an argument of the command.

2.2 Telemetry Response Message

The telemetry response message describes the server response to a client telemetry request message. Each element is described in the following sections. The complete schema is specified in section 5.2.

2.2.1 resp

The **resp** (response) element is the topmost element of a server-to-client response message. There is an associated version number. The version number is used to differentiate schema changes in the telemetry response message format. The version number of this schema is 2.

```
<xs:element name="resp">
 <xs:complexType>
  <xs:attribute name="ver" type="xs:unsignedInt" use="required" />
 </xs:complexType>
</xs:element>
```

2.2.1.1 tlm

The **tlm** (telemetry) element is a child element of the **resp** element. It is the namespace of the service.

```
<xs:element name="tlm">
 <xs:complexType>
 </xs:complexType>
</xs:element>
```

2.2.1.1.1 resps

The **resps** (responses) element is a child element of the **tlm** element. It contains the server responses to the client requests.

```
<xs:element name="resps" minOccurs="1" maxOccurs="1">
 <xs:complexType>
  <xs:sequence>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

2.2.1.1.1.1 resp

The **resp** (response) element describes a server response to a client request. **resp** is a child element of reps. **key** is an attribute and maps to the client **req** key in the client telemetry request message. There is a one-to-one mapping of the telemetry response message **resp** element to the telemetry request message **req** element (see section 2.1.1.1.2.2).

```
<xs:element nm="resp" minOccurs="1" maxOccurs="unbounded">
 <xs:complexType>
  <xs:attribute nm="key" type="xs:string" use="required" />
 </xs:complexType>
</xs:element>
```

2.2.1.1.1.1.1 namespace

The **namespace** element is echoed back from the telemetry request message **req** (section 2.1.1) unaltered. **namespace** is a child element of resp. See the telemetry request message **namespace** element as specified in section 2.1.1.1.2.2.1.

```
<xs:element name="namespace" minOccurs="1" maxOccurs="1">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
     <xs:attribute name="nm" type="xs:string" use="required" />
     <xs:attribute name="val" type="xs:string" use="required" />
    </xs:complexType>
   </xs:element>
  </xs:sequence>
  <xs:attribute name="svc" type="xs:string" use="required" />
  <xs:attribute name="ptr" type="xs:string" use="required" />
  <xs:attribute name="gp" type="xs:string" use="required" />
  <xs:attribute name="app" type="xs:string" use="required" />
 </xs:complexType>
</xs:element>
```

2.2.1.1.1.1.2 cmd

The **cmd** (command) element is a child element of the **resp** element and describes the server command response to the client. The **cmd** element has one attribute, an **nm** (name) attribute describing the command.

```
<xs:element name="cmd" minOccurs="1" maxOccurs="unbounded">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
     <xs:attribute name="nm" type="xs:string" use="required" />
     <xs:attribute name="val" type="xs:string" use="required" />
    </xs:complexType>
   </xs:element>
  </xs:sequence>
  <xs:attribute name="nm" type="xs:string" use="required" />
 </xs:complexType>
</xs:element>
```

The **cmd** element has a set of child **arg** elements with name-value (**nm**,**val**) attribute pairs describing the command arguments.

nm: Name attribute describing an argument of the command. The name is not required to be unique.

val: Value field of the key-value pair describing an argument of the command.

3 Structure Examples

3.1 Client-to-Server Request

This section contains an example of an SQM client-to-server request to upload SQM data as indicated by the **requpload** command. There are two requests (key 1,2), each with a separate **requpload** command.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<req ver="2">
 <tlm>
  <src>
   <desc>
    <mach>
     <os>
      <arg nm="vermaj" val="6" />
      <arg nm="vermin" val="2" />
      <arg nm="verbld" val="8044" />
      <arg nm="verqfe" val="0" />
      <arg nm="versp" val="0" />
      <arg nm="arch" val="0" />
      <arg nm="lcid" val="1033" />
      <arg nm="geoid" val="244" />
     </os>
     <hw>
      <arg nm="form" val="5" />
      <arg nm="arch" val="9" />
      <arg nm="sysmfg" val="Dell Inc." />
      <arg nm="syspro" val="Precision WorkStation 380" />
      <arg nm="bv" val="A07" />
      <arg nm="ram" val="2048" />
     </hw>
     <ctrl>
      <arg nm="tm" val="12955250909324806O" />
      <arg nm="mid" val="{1BC55FD8-3C15-4183-9E34-D8DCCE90535E}" />
     </ctrl>
    </mach>
   </desc>
  </src>
  <reqs>
   <req key="1">
    <namespace svc="sqm" ptr="windows" gp="winsqm8" app="13238528"></namespace>
    <ctrl>
     <arg nm="uid" val="{528EC640-26D0-48CC-9609-E2E44D9194C1}" />
    </ctrl>
    <cmd nm="requpload"></cmd>
   </req>
   <req key="2">
    <namespace svc="sqm" ptr="windows" gp="winsqm8" app="1"></namespace>
    <ctrl>
     <arg nm="uid" val="{528EC640-26D0-48CC-9609-E2E44D9194C1}" />
    </ctrl>
    <cmd nm="requpload"></cmd>
   </req>
  </reqs>
 </tlm>
</req>
```

3.2 Server-to-Client Response

This section contains an example of an SQM server-to-client response. There are two responses that map to the requests in section 3.1.

```
<?xml version="1.0" encoding="utf-8"?>
<resp ver="2">
 <tlm>
  <resps>
   <resp key="1">
    <namespace svc="sqm" ptr="windows" gp="winsqm8" app="13238528">
    </namespace>
    <cmd nm="throttle">
     <arg nm="period" val="30" />
     <arg nm="namespace" val=" app" />
    </cmd>
   </resp>
   <resp key="2">
    <namespace svc="sqm" ptr="windows" gp="winsqm8" app="1">
    </namespace>
    <cmd nm="approved">
     <arg nm="token" val="1.5c719a32ffe543c0.1cb128940d0e3a7" />
     <arg nm="tokenexp" val="129561149070780000" />
    </cmd>
   </resp>
  </resps>
 </tlm>
</resp>
```

4 Security

4.1 Security Considerations for Implementers

The Telemetry Protocol XML Schema relies on the implementor to secure the message over a network by using a secure transport such as **SSL**. Telemetry Protocol XML Schema does not have built-in support for security or authentication.

4.2 Index Of Security Fields

None.

5 Appendix A: Full XML Schema

5.1 Telemetry Request Message - Full Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="req">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="tlm">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="src" minOccurs="1" maxOccurs="1">
        <xs:complexType>
         <xs:sequence>
          <xs:element name="desc" minOccurs="1" maxOccurs="1">
           <xs:complexType>
            <xs:sequence>
             <xs:element name="mach" minOccurs="1" maxOccurs="1">
              <xs:complexType>
               <xs:sequence>
                <xs:element name="os" minOccurs="1" maxOccurs="1">
                 <xs:complexType>
                  <xs:sequence>
                   <xs:element name="arg" minOccurs="0" maxOccurs="unbounded" >
                    <xs:complexType>
                     <xs:attribute name="nm" type="xs:string" use="required" />
                     <xs:attribute name="val" type="xs:string" use="required" />
                    </xs:complexType>
                   </xs:element>
                  </xs:sequence>
                 </xs:complexType>
                </xs:element>
                <xs:element name="hw" minOccurs="1" maxOccurs="1">
                 <xs:complexType>
                  <xs:sequence>
                   <xs:element name="arg" minOccurs="0" maxOccurs="unbounded" >
                    <xs:complexType>
                     <xs:attribute name="nm" type="xs:string" use="required" />
                     <xs:attribute name="val" type="xs:string" use="required" />
                    </xs:complexType>
                   </xs:element>
                  </xs:sequence>
                 </xs:complexType>
                </xs:element>
                <xs:element name="ctrl" minOccurs="1" maxOccurs="1">
                 <xs:complexType>
                  <xs:sequence>
                   <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
                    <xs:complexType>
                     <xs:attribute name="nm" type="xs:string" use="required" />
                     <xs:attribute name="val" type="xs:string" use="required" />
                    </xs:complexType>
                   </xs:element>
                  </xs:sequence>
                 </xs:complexType>
                </xs:element>
               </xs:sequence>
              </xs:complexType>
             </xs:element>
            </xs:sequence>
           </xs:complexType>
          </xs:element>
         </xs:sequence>
        </xs:complexType>
```

```
</xs:element>
<xs:element name="reqs" minOccurs="1" maxOccurs="1">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="payload" minOccurs="0" maxOccurs="1">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
       <xs:complexType>
        <xs:attribute name="nm" type="xs:string" use="required" />
        <xs:attribute name="val" type="xs:string" use="required" />
       </xs:complexType>
      </xs:element>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="req" minOccurs="1" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="namespace" minOccurs="1" maxOccurs="1">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
           <xs:attribute name="nm" type="xs:string" use="required" />
           <xs:attribute name="val" type="xs:string" use="required" />
          </xs:complexType>
         </xs:element>
        </xs:sequence>
        <xs:attribute name="svc" type="xs:string" use="required" />
        <xs:attribute name="ptr" type="xs:string" use="required" />
        <xs:attribute name="gp" type="xs:string" use="required" />
        <xs:attribute name="app" type="xs:string" use="required" />
       </xs:complexType>
      </xs:element>
      <xs:element name="ctrl" minOccurs="0" maxOccurs="1">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
           <xs:attribute name="nm" type="xs:string" use="required" />
           <xs:attribute name="val" type="xs:string" use="required" />
          </xs:complexType>
         </xs:element>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
      <xs:element name="contents" minOccurs="0" maxOccurs="1">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
           <xs:attribute name="nm" type="xs:string" use="required" />
           <xs:attribute name="val" type="xs:string" use="required" />
          </xs:complexType>
         </xs:element>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
      <xs:element name="cmd" minOccurs="1" maxOccurs="1">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
           <xs:attribute name="nm" type="xs:string" use="required" />
           <xs:attribute name="val" type="xs:string" use="required" />
          </xs:complexType>
         </xs:element>
        </xs:sequence>
```

```
              <xs:attribute name="nm" type="xs:string" use="required" />
             </xs:complexType>
            </xs:element>
           </xs:sequence>
           <xs:attribute name="key" type="xs:string" use="required" />
          </xs:complexType>
         </xs:element>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
  <xs:attribute name="ver" type="xs:unsignedInt" use="required" />
 </xs:complexType>
</xs:element>
</xs:schema>
```

5.2 Telemetry Response Message - Full Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="resp">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="tlm">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="resps" minOccurs="1" maxOccurs="1">
        <xs:complexType>
         <xs:sequence>
          <xs:element name="resp" minOccurs="1" maxOccurs="unbounded">
           <xs:complexType>
            <xs:sequence>
             <xs:element name="namespace" minOccurs="1" maxOccurs="1">
              <xs:complexType>
               <xs:sequence>
                <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
                 <xs:complexType>
                  <xs:attribute name="nm" type="xs:string" use="required" />
                  <xs:attribute name="val" type="xs:string" use="required" />
                 </xs:complexType>
                </xs:element>
               </xs:sequence>
               <xs:attribute name="svc" type="xs:string" use="required" />
               <xs:attribute name="ptr" type="xs:string" use="required" />
               <xs:attribute name="gp" type="xs:string" use="required" />
               <xs:attribute name="app" type="xs:string" use="required" />
              </xs:complexType>
             </xs:element>
             <xs:element name="cmd" minOccurs="1" maxOccurs="unbounded">
              <xs:complexType>
               <xs:sequence>
                <xs:element name="arg" minOccurs="0" maxOccurs="unbounded">
                 <xs:complexType>
                  <xs:attribute name="nm" type="xs:string" use="required" />
                  <xs:attribute name="val" type="xs:string" use="required" />
                 </xs:complexType>
                </xs:element>
               </xs:sequence>
               <xs:attribute name="nm" type="xs:string" use="required" />
              </xs:complexType>
             </xs:element>
            </xs:sequence>
            <xs:attribute name="key" type="xs:string" use="required" />
```

```
            </xs:complexType>
           </xs:element>
          </xs:sequence>
         </xs:complexType>
        </xs:element>
       </xs:sequence>
      </xs:complexType>
     </xs:element>
    </xs:sequence>
    <xs:attribute name="ver" type="xs:unsignedInt" use="required" />
   </xs:complexType>
  </xs:element>
 </xs:schema>
```

6 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.

Note: Some of the information in this section is subject to change because it applies to an unreleased, preliminary version of the Windows Server operating system, and thus may differ from the final version of the server software when released. All behavior notes that pertain to the unreleased, preliminary version of the Windows Server operating system contain specific references to Windows Server 2016 Technical Preview as an aid to the reader.

- Windows 8 operating system
- Windows Server 2012 operating system
- Windows 8.1 operating system
- Windows Server 2012 R2 operating system
- Windows 10 operating system
- Windows Server 2016 Technical Preview operating system

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.

7 Change Tracking

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

8 Index