[MS-CEPM-Diff]:

Microsoft Complex Event Processing Engine Manageability Protocol

Revision Summary

Date	Revision History	Revision Class	Comments
8/7/2009	0.1	Major	First release.
11/6/2009	1.0	Major	Updated and revised the technical content.
3/5/2010	2.0	Major	Updated and revised the technical content.
4/21/2010	3.0	Major	Updated and revised the technical content.
6/4/2010	4.0	Major	Updated and revised the technical content.
9/3/2010	4.0.1	Editorial	Changed language and formatting in the technical content.
2/9/2011	4.0.1	None	No changes to the meaning, language, or formatting of the technical content.
7/7/2011	5.0	Major	Updated and revised the technical content.
11/3/2011	5.0	None	No changes to the meaning, language, or formatting of the technical content.
1/19/2012	6.0	Major	Updated and revised the technical content.
2/23/2012	7.0	Major	Updated and revised the technical content.
3/27/2012	7.0	None	No changes to the meaning, language, or formatting of the technical content.
5/24/2012	8.0	Major	Updated and revised the technical content.
6/29/2012	8.0	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	8.0	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	8.0	None	No changes to the meaning, language, or formatting of the technical content.
10/23/2012	8.0	None	No changes to the meaning, language, or formatting of the technical content.
3/26/2013	8.0	None	No changes to the meaning, language, or formatting of the technical content.
6/11/2013	8.0	None	No changes to the meaning, language, or formatting of the technical content.
8/8/2013	8.0	None	No changes to the meaning, language, or formatting of the technical content.
12/5/2013	8.0	None	No changes to the meaning, language, or formatting of the technical content.
2/11/2014	9.0	Major	Updated and revised the technical content.
5/20/2014	9.0	None	No changes to the meaning, language, or formatting of the technical content.
5/10/2016	10.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
8/16/2017	11.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

Complex Event Processing Engine Management (CEPM) is a web service protocol that defines the communication between a client application and a complex event processing (CEP) server. By using this protocol, a client application can create metadata objects on a CEP server, start and stop queries, and query about the CEP system state.

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:

Application object: A CEP metadata object that defines a containing namespace for all child objects, which can be any of the following: Entity, EventType, InputAdapter, OutputAdapter, Query, or QueryTemplate.

CEP metadata object: An object that the CEP server allows an implementer to name and define. A metadata object can be any of the following types: Application, Entity, EventType, InputAdapter, OutputAdapter, Query, or QueryTemplate.

complex event processing (CEP): The continuous and incremental processing of event streams from multiple sources, based on declarative query and pattern specifications with near-zero latency.

CountByStartTime window: A segmentation of the timeline based on the count of distinct event start times. If every event has a unique timestamp, the window spans the specified number of events, starting at the first event's start time and spanning up to the last event's start time plus one tick. If multiple events carry the same timestamp, they count as one unit with respect to the specified window count.

current time increment (CTI): A "heartbeat" event type that does not carry any payload, only a single timestamp. CTIs advance application time in the CEP engine.

Entity object: A CEP metadata object that is represented by an expression that is deployed to the server.

event sink: A destination for an event stream within the CEP platform. In the current version, only output adapters can be event sinks.

EventType object: A CEP metadata object that is used to define the structure of the payload of an event, including the associated fields.

hopping window: A segmentation of the timeline according to a specific fixed window size and a specific fixed hopsize. The hopsize specifies the offset of one window to the next. If the hopsize equals the window size, the windows are non-overlapping and without gaps.

InputAdapter object: A CEP metadata object that is the registration of the binary file compiled from user-written code, which makes the input adapter available to the CEP system so that it can be used in query definitions. This object represents an input stream source and converts proprietary event data into CEP event format.

insert: An event type that declares that a payload is valid for the actual observed duration of the specified event's lifetime (start time–end time).

OutputAdapter object: A CEP metadata object that is the registration of the binary file compiled from user-written code, which makes the output adapter available to the CEP system so that it can be used in query definitions. This object represents an output stream source and receives events that are produced by the CEP engine for further processing.

Query object: A CEP metadata object that represents the binding of input and output adapters and a QueryTemplate object within an application.

QueryTemplate object: A CEP metadata object that defines how to compute an output stream from one or more input streams.

retract: An event type that shortens the lifetime of an event. To be associated with an event, the retract must match the specified event's start time, end time, and entire set of payload field values.

snapshot window: A division of the timeline that is created when the timeline is divided into segments along every event start and every event end. A snapshot window, by definition, does not contain any start or end of an event except at its boundaries.

SOAP body: A container for the payload data being delivered by a SOAP message to its recipient. See [SOAP1.2-1/2007] section 5.3 for more information.

SOAP fault: A container for error and status information within a SOAP message. See [SOAP1.2-1/2007] section 5.4 for more information.

SOAP header: A mechanism for implementing extensions to a SOAP message in a decentralized manner without prior agreement between the communicating parties. See [SOAP1.2-1/2007] section 5.2 for more information.

Web Services Description Language (WSDL): An XML format for describing network services as a set of endpoints that operate on messages that contain either document-oriented or procedure-oriented information. The operations and messages are described abstractly and are bound to a concrete network protocol and message format in order to define an endpoint. Related concrete endpoints are combined into abstract endpoints, which describe a network service. WSDL is extensible, which allows the description of endpoints and their messages regardless of the message formats or network protocols that are used.

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.

[ISO3166-1:2006] ISO, "Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes", ISO 3166-1:2006, http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=39719

Note There is a charge to download the specification.

[ISO639-2] ISO, "Codes for the representation of names of languages -- Part 2: Alpha-3 code", ISO 639-2:1998, http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=4767

Note There is a charge to download the specification.

[MS-LETSF] Microsoft Corporation, "LINQ Expression Tree Serialization Format".

[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~~RFC7230] Fielding, R., ~~Gettys~~and Reschke, J., ~~Mogul, J., et al~~Eds., "Hypertext Transfer Protocol ~~--~~(HTTP/1.1): Message Syntax and Routing", RFC ~~2616~~7230, June ~~1999~~2014, http://www.rfc-editor.org/rfc/~~rfc2616~~rfc7230.txt

[SOAP1.2-1/2003] Gudgin, M., Hadley, M., Mendelsohn, N., et al., "SOAP Version 1.2 Part 1: Messaging Framework", W3C Recommendation, June 2003, http://www.w3.org/TR/2003/REC-soap12-part1-20030624

[WSA] Gudgin, M., Hadley, M., and Rogers, T., "Web Services Addressing 1.0 - Core", W3C Recommendation, May 2006, http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/

[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

[XML10/5] Bray, T., Paoli, J., Sperberg-McQueen, C.M., et al., Eds., "Extensible Markup Language (XML) 1.0 (Fifth Edition)", W3C Recommendation, November 2008, http://www.w3.org/TR/2008/REC-xml-20081126/

[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

[MSDN-AsmblyNames] Microsoft Corporation, "Assembly Names", http://msdn.microsoft.com/en-us/library/k8xx4k69.aspx

[MSDN-CIPN] Microsoft Corporation, "CultureInfo.Name Property", http://msdn.microsoft.com/en-us/library/system.globalization.cultureinfo.name.aspx

[MSDN-CompareOptions] Microsoft Corporation, "CompareOptions Enumeration", ~~.NET Framework Class Library,~~ http://msdn.microsoft.com/en-us/library/system.globalization.compareoptions.aspx

[MSDN-IDPTETW] Microsoft Corporation, "Improve Debugging and Performance Tuning With ETW", April 2007, http://msdn.microsoft.com/en-us/magazine/cc163437.aspx

[MSDN-MPCEP] Microsoft Corporation, "Introducing Microsoft StreamInsight", http://download.microsoft.com/download/F/D/5/FD5E855C-D895-45A8-9F3E-110AFADBE51A/Microsoft%20CEP%20Overview.docx

[MSDN-StringComparison] Microsoft Corporation, "StringComparison Enumeration", http://msdn.microsoft.com/en-us/library/system.stringcomparison.aspx

[MSDN-SysName] Microsoft Corporation, "System Namespace", http://msdn.microsoft.com/en-us/library/system(VS.71).aspx

[MSDN-TAQNP] Microsoft Corporation, "Type.AssemblyQualifiedName Property", http://msdn.microsoft.com/en-us/library/system.type.assemblyqualifiedname.aspx

1.3 Overview

Complex event processing (CEP) is the continuous and incremental processing of event (data) streams from multiple sources based on declarative query and pattern specifications with near-zero latency. The goal is to identify meaningful patterns, relationships, and data abstractions from among seemingly unrelated events and to trigger immediate response actions. For more information, see [MSDN-MPCEP].

Typical event stream sources include data from manufacturing applications, financial trading applications, web analytics, and operational analytics.

The CEP engine provides a dedicated web service to handle requests from client applications for managing the system. Using the protocol described in this document, applications issue instructions to the CEP engine to create, start, and stop queries, and to inquire about query status and other parameters that describe the health of a running CEP engine. The protocol also supports messages that are used to enable and disable specific performance counters and event tracing.

The CEPM protocol is used to communicate with the web service that is provided by the CEP engine to define and manage all of the CEP ~~system's~~system’s objects. As soon as all of the objects are defined and in place in the CEP engine, a protocol message to start the query causes the CEP engine to tap into the streaming data and to calculate and send output data. Another such message will stop the engine from recording and computing data. The CEPM protocol is used to create and manage the following objects:

- Application object
- Entity object
- EventType object
- InputAdapter object
- OutputAdapter object
- Query object
- QueryTemplate object

The CEPM protocol is stateless. All communication is initiated by the client. The server only sends responses in response to messages received. The following figure shows the methods available in this protocol.

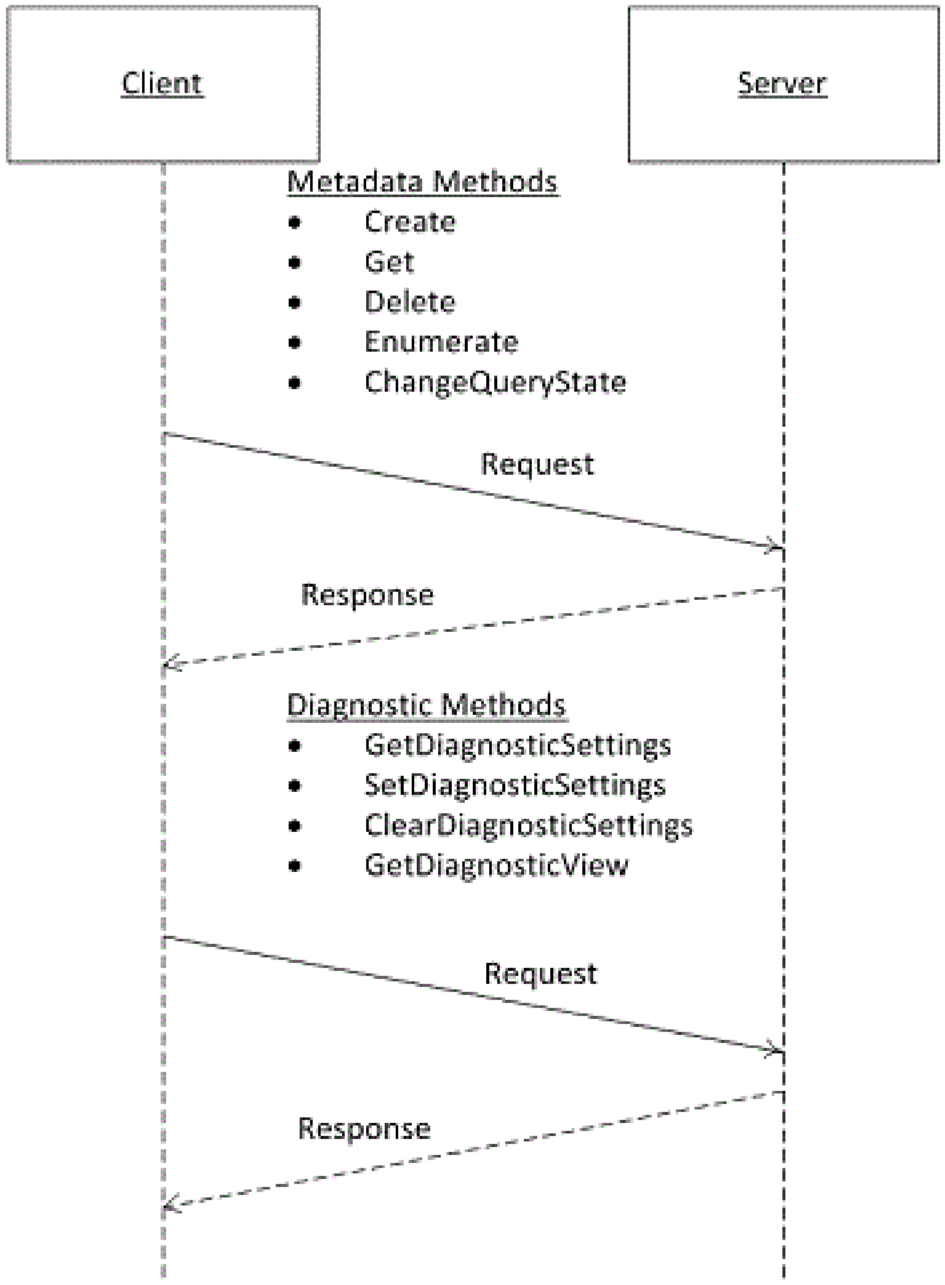

Figure 1: CEPM web service protocol showing the available methods

1.4 Relationship to Other Protocols

The CEPM protocol uses SOAP over HTTP, as shown in the following layering diagram.

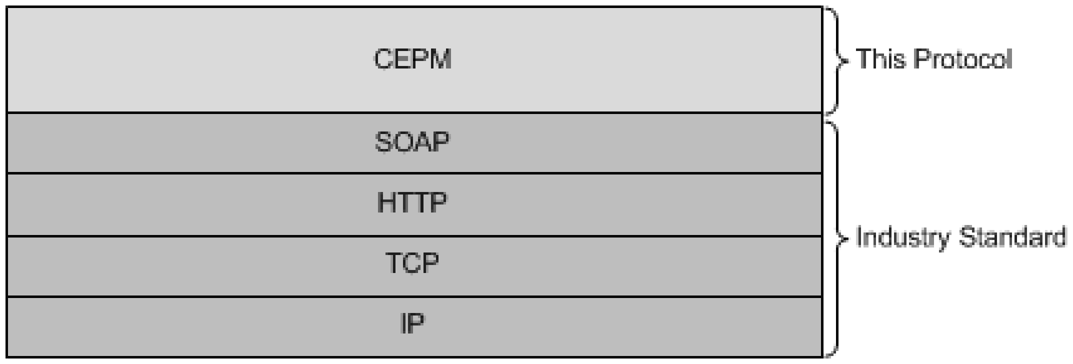

Figure 2: SOAP over HTTP

The CEPM protocol relies on the LINQ expression tree serialization format protocol [MS-LETSF].

1.5 Prerequisites/Preconditions

To implement the CEPM protocol successfully, a running instance of the complex event processing (CEP) engine in a stand-alone or embedded configuration is the only prerequisite.

1.6 Applicability Statement

None.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

- **Supported Transports:** This protocol uses transports with SOAP, as specified in Transport (section 2.1) later in this document.
- **Localization:** This protocol allows text characters in any language, but it does not support localization of text strings into multiple languages. The protocol supports creating expressions that can be properly compared in different cultures, as described in ComparisonExpression (section 2.2.3.2.3.7.3) later in this document.
- **Capability Negotiation:** This is the first released version of this protocol. No protocol capability negotiation is supported.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

Protocol messages MUST be formatted as a SOAP envelope as specified in [SOAP1.2-1/2003].

Protocol servers MUST support SOAP [SOAP1.2-1/2003] over Hypertext Transfer Protocol (HTTP) [~~RFC2616~~RFC7230].

The message format is clear-text XML [XML10/5].

No authentication is supported by this protocol at this time.

2.2 Messages

This section defines messages used by this protocol. The syntax of the definitions uses XML schema (XSD) as defined in [XMLSCHEMA1/2] and [XMLSCHEMA2/2] and Web Services Description Language (WSDL) as defined in [WSDL].

2.2.1 Namespaces

This specification defines and references various XML namespaces that use 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
metadata	http://schemas.microsoft.com/ComplexEventProcessing/2010/01/Metadata	http://schemas.microsoft.com/ComplexEventProcessing/2010/01/Metadata
msc	http://schemas.microsoft.com/ws/2005/12/wsdl/contract	http://schemas.microsoft.com/ws/2005/12/wsdl/contract
s	http://www.w3.org/2003/05/soap-envelope	http://www.w3.org/2003/05/soap-envelope
soap	http://schemas.xmlsoap.org/wsdl/soap/	http://schemas.xmlsoap.org/wsdl/soap/
soapenc	http://schemas.xmlsoap.org/soap/encoding/	http://schemas.xmlsoap.org/soap/encoding/
soap12	http://schemas.xmlsoap.org/wsdl/soap12/	http://schemas.xmlsoap.org/wsdl/soap12/
tns	http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management	http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management
wsa	http://schemas.xmlsoap.org/ws/2004/08/addressing	http://schemas.xmlsoap.org/ws/2004/08/addressing
wsam	http://www.w3.org/2007/05/addressing/metadata	http://www.w3.org/2007/05/addressing/metadata
wsap	http://schemas.xmlsoap.org/ws/2004/08/addressing/policy	http://schemas.xmlsoap.org/ws/2004/08/addressing/policy
wsa	http://www.w3.org/2006/05/addressing/wsdl	http://www.w3.org/2006/05/addressing/wsdl

Prefix	Namespace URI	Reference
w		
wsa10	http://www.w3.org/2005/08/addressing	http://www.w3.org/2005/08/addressing/
wsdl	http://schemas.xmlsoap.org/wsdl/	http://schemas.xmlsoap.org/wsdl/
wsp	http://schemas.xmlsoap.org/ws/2004/09/polic	http://schemas.xmlsoap.org/ws/2004/09/policy/
wsu	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd
wsx	http://schemas.xmlsoap.org/ws/2004/09/mex	http://schemas.xmlsoap.org/ws/2004/09/mex
xs	http://www.w3.org/2001/XMLSchema	http://www.w3.org/2001/XMLSchema
ser	http://schemas.microsoft.com/2003/10/Serialization/	http://schemas.microsoft.com/2003/10/Serialization/
sera	http://schemas.microsoft.com/2003/10/Serialization/Arrays	http://schemas.microsoft.com/2003/10/Serialization/Arrays
linqs	http://schemas.microsoft.com/ComplexEventProcessing/2011/10/LinqExpression	http://schemas.microsoft.com/ComplexEventProcessing/2011/10/LinqExpression
mgmt	http://schemas.datacontract.org/2004/07/Microsoft.ComplexEventProcessing.ManagementService	http://schemas.datacontract.org/2004/07/Microsoft.ComplexEventProcessing.ManagementService
desn	http://schemas.microsoft.com/ComplexEventProcessing/2011/10/Design	http://schemas.microsoft.com/ComplexEventProcessing/2011/10/Design

2.2.2 Methods

The following table summarizes the set of method definitions defined by this specification.

Method	Description
Create	Used to create CEP metadata objects that include a complex event processing (CEP) application instance, such as event types, input and output adapters, query templates, and queries.
Get	Retrieves the definition of an object that has already been created.
Delete	Used to delete an object that has been created.
Enumerate	Used to return the names in a collection of like objects that have already been created, such as a collection of names of event types.
ChangeQueryState	Used to change the state of a query from stopped to started, or vice versa.
GetDiagnosticSettings	Retrieves the diagnostic settings that are currently in effect on a specific object in the system.
SetDiagnosticSettings	Sets diagnostic settings for monitoring a specific object in the CEP system.
ClearDiagnosticsSettings	Clears any previously set diagnostic settings on a specific object in the CEP system.
GetDiagnosticView	Used to request the observed values for a set of **DiagnosticView** properties that

Method	Description
	have been previously defined. The list of properties that are returned is variable; it depends on the settings that are set by using the **SetDiagnosticSettings** message and on the type of object for which the diagnostic view is being retrieved.

2.2.2.1 Metadata Methods

These methods are used to create, remove, and manage metadata objects on the complex event processing (CEP) server.

2.2.2.1.1 Create Message

A **Create** message is used to create objects on a complex event processing (CEP) server and to receive the response to the message.

2.2.2.1.1.1 CreateRequest Message

The **CreateRequest** message is used to create all CEP metadata objects within an application. The **Application** object is the top-level object that scopes a CEP application. Each **Application** object includes zero or more of the following metadata objects:

- Entity objects
- EventType objects each for input and output, each of which ~~may~~can contain multiple **Field** objects.
- InputAdapter objects
- OutputAdapter objects
- Query objects
- QueryTemplate objects

A CreateRequest message MUST set elements in both the SOAP header and the SOAP body, as described in the following sections.

2.2.2.1.1.1.1 CreateRequest SOAP Header

A CreateRequest message MUST set the following elements in the SOAP header.

Element	Type	Description
wsa10:Action	**xs:string**	The ~~value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management/Create**~~.~~
tns:Name	**xs:anyURI**	This URI MUST be set to the name of the object being created with "cep:/Server" as the highest-level object.<1>

2.2.2.1.1.1.2 CreateRequest SOAP Body

The following elements MUST be present in the SOAP body of a CreateRequest message.

Element	Type	Description
CreateRequest	**CreateRequest**	The **CreateRequest** element contains the definition of the object being created as a child of the object referenced in the **tns:Name** element of the SOAP header. For the definition of the **CreateRequest** type, see section 2.2.3.1.1.

2.2.2.1.1.2 CreateResponse Message

The **CreateResponse** message MUST be sent by the server in response to a received CreateRequest message, unless there is a fault or an exception.

2.2.2.1.1.2.1 CreateResponse SOAP Header

The following elements MUST be set in the SOAP header of a CreateResponse message.

Element	Type	Description
wsa10:Action	**xs:string**	The **wsa10:Action** element MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/CreateResponse**
tns:ResourceAddress	**wsa:ResourceAddress**	A standard SOAP **ResourceAddress** element. For more details, see [WSA].

2.2.2.1.1.2.2 CreateResponse SOAP Body

The SOAP body for a CreateResponse message MUST be empty.

2.2.2.1.1.2.3 Faults

The response to the CreateRequest message ~~may~~can be one of the following faults:

- InvalidNameFault
- InvalidDefinitionFault
- ManagementFault

For a description of the content of the fault return result, see section 2.2.2.3.

2.2.2.1.1.3 Create Examples

The following examples show a ~~client's~~client’s CreateRequest message and the complex event processing (CEP) ~~server's~~server’s CreateResponse message that is sent in response to the received **CreateRequest** message.

2.2.2.1.1.3.1 CreateRequest

The following example **CreateRequest** message is an instruction from the client to create an EventType object with a name of EventType1, which contains one field (named Field1) of type System.Int32.

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
            xmlns:s="http://www.w3.org/2003/05/soap-envelope">
            <s:Header>
              <a:Action s:mustUnderstand="1">
              http://schemas.microsoft.com/ComplexEventProcessing
              /2009/10/Management/Create
              </a:Action>
        <h:Name xmlns:h=
        "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
                cep:/Server/Application/app1</h:Name>
        <a:MessageID>urn:uuid:364ba4c5-3cbb-42a6-b094-2611663168cc</a:MessageID>
              <ActivityId CorrelationId="8af6ef4d-2ec5-45a8-b485-14d603158907"
        xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
                00000000-0000-0000-0000-000000000000</ActivityId>
              <a:ReplyTo>
                <a:Address>http://www.w3.org/2005/08/addressing/anonymous
                </a:Address>
              </a:ReplyTo>
            </s:Header>
            <s:Body>
              <CreateRequest xmlns=
    "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
                <EventType Name="EventType1"
    xmlns="http://schemas.microsoft.com/ComplexEventProcessing/2010/01/Metadata">
                  <Field Name="Field1" Type="System.Int32" Nullable="false" />
                </EventType>
              </CreateRequest>
            </s:Body>
          </s:Envelope>
```

2.2.2.1.1.3.2 CreateResponse

The following example shows the **CreateResponse** message that is sent by the server in response to the preceding CreateRequest message.

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">
"http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetResponse"
    </a:Action>
    <h:ResourceAddress xmlns:h=
      "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
      <a:Address>http://localhost:8090/</a:Address>
      <a:ReferenceParameters>
        <h:Name>cep:/Server/Application/app1/EventType/EventType1</h:Name>
      </a:ReferenceParameters>
    </h:ResourceAddress>
    <a:RelatesTo>urn:uuid:364ba4c5-3cbb-42a6-b094-2611663168cc</a:RelatesTo>
    <ActivityId CorrelationId="de658eac-6f7c-4b75-99db-64d4e917ab4f"
     xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
      00000000-0000-0000-0000-000000000000</ActivityId>
  </s:Header>
  <s:Body></s:Body>
</s:Envelope>
```

2.2.2.1.2 Get Message

A **Get** message is used to request and receive the definition of a CEP metadata object that has already been created.

2.2.2.1.2.1 GetRequest Message

A **GetRequest** message is used to fetch the definition of a CEP metadata object that has already been created.

2.2.2.1.2.1.1 GetRequest SOAP Header

The SOAP header for a GetRequest message MUST contain the following elements.

Element	Type	Description
wsa10:Action	**xs:string**	The ~~value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management/Get**~~.~~
tns:Name	**xs:anyURI**	This URI MUST be set to the fully qualified URI of the object that is being requested.<2>

2.2.2.1.2.1.2 GetRequest SOAP Body

The SOAP body for a GetRequest message MUST be empty.

2.2.2.1.2.2 GetResponse Message

The **GetResponse** message MUST be sent by the server in response to a received GetRequest message, unless there is a fault or an exception.

2.2.2.1.2.2.1 GetResponse SOAP Header

The SOAP header for a GetResponse message MUST contain the following elements.

Element	Type	Description
wsa10:Action	**xs:string**	The **wsa10:Action** element MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetResponse**

2.2.2.1.2.2.2 GetResponse SOAP Body

The SOAP body for a GetResponse message MUST set the following element.

Element	Type	Description
GetResponse	**GetResponse**	The **GetResponse** element contains the definition of the CEP metadata object that was requested in the **tns:Name** element of the SOAP header of the GetRequest message. For more information, see section 2.2.3.1.2 for the definition of the **GetResponse** type.

2.2.2.1.2.2.3 Faults

The response to the GetRequest message ~~may~~can be one of the following ~~fault~~faults:

- InvalidNameFault

- ManagementFault

For a description of the content of the fault return result, see section 2.2.2.3.

2.2.2.1.2.3 Get Examples

The following examples show a ~~client's~~client’s GetRequest message and the complex event processing (CEP) ~~server's~~server’s GetResponse message that is sent in response to the received **GetRequest** message.

2.2.2.1.2.3.1 GetRequest

The following example **GetRequest** message is an instruction from the client to get the definition of the URI given in the **h:Name** element of the SOAP header, "cep:/Server/Application/app1/EventType/EventType1".

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">
      http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management/Get
    </a:Action>
    <h:Name xmlns:h=
      "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
      cep:/Server/Application/app1/EventType/EventType1</h:Name>
    <a:MessageID>urn:uuid:5b7cba99-8c7a-4045-b4ea-921749f8b417</a:MessageID>
    <ActivityId CorrelationId="a9c26108-46eb-4378-89e2-a87a49e18aa9"
      xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
      00000000-0000-0000-0000-000000000000</ActivityId>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://localhost:8090/</a:To>
  </s:Header>
  <s:Body></s:Body>
</s:Envelope>
```

2.2.2.1.2.3.2 GetResponse

The following example shows the **GetResponse** message that is sent by the server in response to the preceding GetRequest message.

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">
"http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetResponse"
    </a:Action>
    <a:RelatesTo>urn:uuid:5b7cba99-8c7a-4045-b4ea-921749f8b417</a:RelatesTo>
    <ActivityId CorrelationId="7e9927d1-b818-4efa-98b1-f7d5909e9833"
      xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
      00000000-0000-0000-0000-000000000000</ActivityId>
  </s:Header>
  <s:Body>
    <GetResponse xmlns=
       "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
      <EventType Name="cep:/Server/Application/app1/EventType/EventType1"
       xmlns="http://schemas.microsoft.com/ComplexEventProcessing/2010/01/Metadata">
        <Field Name="CountSegmentHitLogicId" Type="System.Int32" Nullable="true"
               MaxSize="4"></Field>
        <Field Name="SegmentHitLogicId" Type="System.Int32" Nullable="true"
               MaxSize="4"></Field>
        <Field Name="UserId" Type="System.Int32" Nullable="true"
               MaxSize="4"></Field>
      </EventType>
    </GetResponse>
  </s:Body>
</s:Envelope>
```

2.2.2.1.3 Delete Message

The **Delete** message is used to delete a CEP metadata object that was previously created in an instance of the complex event processing (CEP) server, and to receive the response to the message.

2.2.2.1.3.1 DeleteRequest Message

A **DeleteRequest** message is used to request the deletion of a CEP metadata object that was previously created, and which currently exists on the server.

2.2.2.1.3.1.1 DeleteRequest SOAP Header

The following elements MUST be set in the SOAP header of a DeleteRequest message.

Element	Type	Description
wsa10:Action	**xs:string**	The ~~value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Delete**~~.~~
tns:Name	**xs:anyURI**	This URI MUST be set to the fully qualified name of the CEP metadata object to be deleted.

2.2.2.1.3.1.2 DeleteRequest SOAP Body

The SOAP body for a DeleteRequest message MUST be empty.

2.2.2.1.3.2 DeleteResponse Message

A **DeleteResponse** message MUST be sent in response to a received DeleteRequest message, unless there is a fault or an exception.

2.2.2.1.3.2.1 DeleteResponse SOAP Header

The following elements MUST be set in the SOAP header of a DeleteResponse message.

Element	Type	Description
wsa10:Action	**xs:string**	The ~~value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/DeleteResponse**~~.~~
tns:Name	**xs:anyURI**	This URI MUST be set to the fully qualified name of the object that has been deleted.

2.2.2.1.3.2.2 DeleteResponse SOAP Body

The SOAP body for a DeleteResponse message MUST be empty.

2.2.2.1.3.2.3 Faults

The response to the DeleteRequest message ~~may~~can be one of the following faults:

- InvalidNameFault
- ManagementFault

For a description of the content of the fault return result, see section 2.2.2.3.

2.2.2.1.3.3 Delete Examples

The following examples show a ~~client's~~client's DeleteRequest message and the complex event processing (CEP) ~~server's~~server's DeleteResponse message that is sent in response to the received **DeleteRequest** message.

2.2.2.1.3.3.1 DeleteRequest

The following example **DeleteRequest** message is an instruction from the client to delete the URI contained in the **h:Name** element "cep:/Server/Application/app1/EventType/EventType1".

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">
    http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management/Delete
    </a:Action>
    <h:Name xmlns:h=
      "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
      cep:/Server/Application/app1/EventType/EventType1</h:Name>
    <a:MessageID>urn:uuid:22ed0175-f845-464f-aec0-d641c3f1ef7b</a:MessageID>
    <ActivityId CorrelationId="00f03903-98c8-41be-b128-d9b9759714ff"
      xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
      00000000-0000-0000-0000-000000000000</ActivityId>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://localhost:8090/</a:To>
  </s:Header>
  <s:Body></s:Body>
</s:Envelope>
```

2.2.2.1.3.3.2 DeleteResponse

The following example shows the **DeleteResponse** message that is sent by the server in response to the preceding received DeleteRequest message.

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">
http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/DeleteResponse</a:Acti
on>
```

```
      <h:Name xmlns:h=
      "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
        cep:/Server/Application/app1/EventType/EventType1</h:Name>
      <a:RelatesTo>urn:uuid:22ed0175-f845-464f-aec0-d641c3f1ef7b</a:RelatesTo>
      <ActivityId CorrelationId="f0c88453-217e-4d8a-b0de-c00328943ac2"
        xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
        00000000-0000-0000-0000-000000000000</ActivityId>
    </s:Header>
    <s:Body></s:Body>
  </s:Envelope>
```

2.2.2.1.4 Enumerate Message

An **Enumerate** message is used to request and receive the names of a collection of CEP metadata objects that has already been created.

2.2.2.1.4.1 EnumerateRequest Message

The **EnumerateRequest** message is used to request the enumeration of definitions for a collection of CEP metadata objects with a common parent (for example, a collection of EventType objects with a common application object parent).

2.2.2.1.4.1.1 EnumerateRequest SOAP Header

The following elements MUST be set in the SOAP header of the EnumerateRequest message.

Element	Type	Description
wsa10:Action	**xs:string**	The~~ value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Enumerate**~~.~~
tns:Name	**xs:anyURI**	This URI MUST be set to a URI that represents the collection to be retrieved. This consists of the fully qualified URI of the parent CEP metadata object that contains the collection of objects to be enumerated, plus one of the following strings that represents the collection to be enumerated: ▪ "Application" ▪ "EventType" ▪ "InputAdapter" ▪ "OutputAdapter" ▪ "QueryTemplate" ▪ "Query" ▪ "Operator" ▪ "Branch" ▪ "Stream" ▪ "OutputStream"

2.2.2.1.4.1.2 EnumerateRequest SOAP Body

The SOAP body for an EnumerateRequest message MUST be empty.

2.2.2.1.4.2 EnumerateResponse Message

An **EnumerateResponse** message MUST be sent in response to a received EnumerateRequest message, unless there is an exception or a fault.

2.2.2.1.4.2.1 EnumerateResponse SOAP Header

The following elements MUST be set in the SOAP header of an EnumerateResponse message.

Element	Type	Description
wsa10:Action	**xs:string**	The~~ value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/EnumerateResponse**~~.~~

2.2.2.1.4.2.2 EnumerateResponse SOAP Body

The following elements MUST be set in the SOAP body of an EnumerateResponse message.

Element	Type	Description
ResourceNames	**sera:ArrayOfanyURI**	A collection of CEP metadata object names, expressed as URIs, that are contained in the collection specified in the **tns:Name** element of the SOAP header of the EnumerateRequest message.

The following code is the XML schema (XSD) for the **ArrayOfanyURI** type.

```
<xs:element name="ResourceNames" nillable="true"
xmlns:q2="http://schemas.microsoft.com/2003/10/Serialization/Arrays" type="q2:ArrayOfanyURI"
/>
```

2.2.2.1.4.2.3 Faults

The response to the EnumerateRequest message ~~may~~can contain one of the following ~~fault~~faults:

- InvalidNameFault
- ManagementFault

For a description of the content of the fault return result, see section 2.2.2.3.

2.2.2.1.4.3 Enumerate Examples

The following examples show a ~~client's~~client’s EnumerateRequest message and the complex event processing (CEP) ~~server's~~server’s EnumerateResponse message that is sent in response to the received **EnumerateRequest** message.

2.2.2.1.4.3.1 EnumerateRequest

The following example **EnumerateRequest** message is request from the client to retrieve the collection of EventType objects with parent "cep:/Server/Application/app1/Query/Select1/Stream" as specified in the **h:Name** element of the SOAP header.

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">
http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Enumerate
    </a:Action>
    <h:Name
 xmlns:h="http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
      cep:/Server/Application/app1/Query/Select1/Stream</h:Name>
    <a:MessageID>urn:uuid:70451518-1da1-45c8-9ebb-a2fd144d19ae
    </a:MessageID>
    <ActivityId CorrelationId="48b23d28-78aa-4bf2-b776-e3d57a116adb"
      xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
      00000000-0000-0000-0000-000000000000</ActivityId>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous
      </a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://localhost:8090/</a:To>
  </s:Header>
  <s:Body></s:Body>
</s:Envelope>
```

2.2.2.1.4.3.2 EnumerateResponse

The following example shows the **EnumerateResponse** message that is sent by the server in response to the preceding received EnumerateRequest message.

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">
      http://schemas.microsoft.com/ComplexEventProcessing/2009/05/
      Management/EnumerateResponse</a:Action>
    <a:RelatesTo>urn:uuid:70451518-1da1-45c8-9ebb-a2fd144d19ae</a:RelatesTo>
    <ActivityId CorrelationId="463b7847-c8bd-40db-9b18-e0a1155cb845"
      xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
      00000000-0000-0000-0000-000000000000</ActivityId>
  </s:Header>
  <s:Body>
    <ResourceNames
      xmlns="http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management"
      xmlns:b="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
      xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
      <b:anyURI>cep:/Server/Application/app1/Query/Select1/Stream/select1</b:anyURI>
      <b:anyURI>cep:/Server/Application/app1/Query/Select1/Stream/import1</b:anyURI>
    </ResourceNames>
  </s:Body>
</s:Envelope>
```

2.2.2.1.5 ChangeQueryState Message

The **ChangeQueryState** message is used to start and stop a Query object that has been created on a complex event processing (CEP) server and to receive the response to the message.

2.2.2.1.5.1 ChangeQueryStateRequest Message

The **ChangeQueryStateRequest** message is used to start a Query object running or stop it while it is running.

2.2.2.1.5.1.1 ChangeQueryStateRequest SOAP Header

The following elements MUST be set in the SOAP header of a ChangeQueryStateRequest message.

Element	Type	Description
wsa10:Action	**xs:string**	The ~~value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management/ChangeQueryState**~~.~~
tns:Name	**xs:anyURI**	The URI of the Query object for which it is desired to change the query state.

2.2.2.1.5.1.2 ChangeQueryStateRequest SOAP Body

The following elements MUST be set in the SOAP body of a ChangeQueryStateRequest message.

Element	Type	Description
QueryState	QueryState	The **QueryState** element is an enumeration that sets the query state. For more details, see section 2.2.3.1.3.

2.2.2.1.5.2 ChangeQueryStateResponse Message

The **ChangeQueryStateResponse** message MUST be sent by the server in response to a received ChangeQueryStateRequest message, unless there is an exception or a fault.

2.2.2.1.5.2.1 ChangeQueryStateResponse SOAP Header

The following elements MUST be set in the SOAP header of a ChangeQueryStateResponse message.

Element	Type	Description
wsa10:Action	**xs:string**	The ~~value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management/ChangeQueryStateResponse**~~.~~
h:Name	**xs:anyURI**	This URI MUST be set to the URI of the complex event processing (CEP) object whose query state has been changed as a result of the sent ChangeQueryStateRequest message.

2.2.2.1.5.2.2 ChangeQueryStateResponse SOAP Body

The following elements MUST be set in the SOAP body of a ChangeQueryStateResponse message.

Element	Type	Description
QueryState	QueryState	The **QueryState** element is an enumeration that sets the query state. For more details, see section 2.2.3.1.3.

2.2.2.1.5.2.3 Faults

The response to the ChangeQueryStateRequest message ~~may~~can be one of the following faults:

- InvalidNameFault
- RuntimeFault
- ManagementFault

For a description of the content of the fault return result, see section 2.2.2.3.

2.2.2.1.5.3 ChangeQueryState Examples

The following examples show a ~~client's~~client’s ChangeQueryStateRequest message and the complex event processing (CEP) ~~server's~~server’s ChangeQueryStateResponse message that is sent in response to the received **ChangeQueryStateRequest** message.

2.2.2.1.5.3.1 ChangeQueryStateRequest

The following example **ChangeQueryStateRequest** message is an instruction from the client to set the query state to **QueryStateStarted** for the query with the URI in the **h:Name** element, "cep:/Server/Application/app1/Query/Select1".

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">
      http://schemas.microsoft.com/ComplexEventProcessing/2009/10/
      Management/ChangeQueryState</a:Action>
    <h:Name
  xmlns:h="http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
      cep:/Server/Application/app1/Query/Select1</h:Name>
    <a:MessageID>urn:uuid:55d3ffb6-15a5-47a3-93e4-81057e05b57e</a:MessageID>
    <ActivityId CorrelationId="38d5660b-6cbf-4865-9db5-ab5c1c64aeb7"
      xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
      00000000-0000-0000-0000-000000000000</ActivityId>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://localhost:8090/</a:To>
  </s:Header>
  <s:Body>
    <QueryState
    xmlns="http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
      http://schemas.microsoft.com/ComplexEventProcessing/2009/10/
      Management/QueryStateStarted</QueryState>
  </s:Body>
</s:Envelope>
```

2.2.2.1.5.3.2 ChangeQueryStateResponse

The following example shows the **ChangeQueryStateResponse** message that is sent by the server in response to the preceding received ChangeQueryStateRequest message.

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">
      http://schemas.microsoft.com/ComplexEventProcessing/2009/10/
      Management/ChangeQueryStateResponse</a:Action>
    <h:Name xmlns:h=
      "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
      cep:/Server/Application/app1/Query/Select1</h:Name>
    <a:RelatesTo>urn:uuid:55d3ffb6-15a5-47a3-93e4-81057e05b57e</a:RelatesTo>
    <ActivityId CorrelationId="22ccf7e3-3095-4073-94e6-fff007153b40"
      xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
      00000000-0000-0000-0000-000000000000</ActivityId>
  </s:Header>
  <s:Body>
    <QueryState
    xmlns="http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
    http://schemas.microsoft.com/ComplexEventProcessing/2009/10/
    Management/QueryStateStarted</QueryState>
  </s:Body>
</s:Envelope>
```

2.2.2.1.6 ResumeProcess Message

The **ResumeProcess** message is used to restart a process that was previously running on a complex event processing (CEP) server and to receive the response to the message.

2.2.2.1.6.1 ResumeProcessRequest Message

The **ResumeProcessRequest** message is used to restart a process that was previously running on a complex event processing (CEP) server.

2.2.2.1.6.1.1 ResumeProcessRequest SOAP Header

The following elements MUST be set in the SOAP header of a ResumeProcessRequest message.

Element	Type	Description
tns:Name	**xs:anyURI**	The URI of the **Entity** to be restarted.

2.2.2.1.6.1.2 ResumeProcessRequest SOAP Body

The SOAP body for a ResumeProcessRequest message MUST be empty.

2.2.2.1.6.2 ResumeProcessResponse Message

The **ResumeProcessResponse** message MUST be sent by the server in response to a received ResumeProcessRequest message, unless there is an exception or a fault.

2.2.2.1.6.2.1 ResumeProcessResponse SOAP Header

The following elements MUST be set in the SOAP header of a ResumeProcessResponse message.

Element	Type	Description
h:Name	**xs:anyURI**	This URI MUST be set to the URI of the Entity that has been restarted as a result of a ResumeProcessRequest message.

2.2.2.1.6.2.2 ResumeProcessResponse SOAP Body

The SOAP body for a ResumeProcessResponse message MUST be empty.

2.2.2.1.6.2.3 Faults

The response to the ResumeProcessResponse message can be one of the following faults:

- InvalidNameFault
- ManagementFault
- RuntimeFault

For a description of the content of the fault return result, see Faults (section 2.2.2.3).

2.2.2.1.7 GetEntityExpressionType Message

The **GetEntityExpressionType** message is used to request the description of an **Entity** that is deployed to a CEP server and to receive the response to the message.

2.2.2.1.7.1 GetEntityTypeRequest Message

The **GetEntityTypeRequest** message is used to request the description of an **Entity** that is deployed to a CEP server.

2.2.2.1.7.1.1 GetEntityTypeRequest SOAP Header

The following elements MUST be set in the SOAP header of a GetEntityTypeRequest message.

Element	Type	Description
tns:Name	**xs:anyURI**	The URI of the Entity object that the description is being requested for.

2.2.2.1.7.1.2 GetEntityTypeRequest SOAP Body

The SOAP body for a GetEntityTypeRequest message MUST be empty.

2.2.2.1.7.2 GetResponse

The GetResponse message (section 2.2.2.1.2.2) is used to respond to a GetEntityTypeRequest message. This result MUST contain an **Entity** element.

2.2.2.1.8 GetTypeDefinition Message

The **GetTypeDefinition** message is used to request the description of the type of an **Entity** that is deployed to a CEP server and to receive the response to the message.

2.2.2.1.8.1 GetTypeDefinitionRequest Message

The **GetTypeDefinitionRequest** message is used to request the description of the type of an **Entity** that is deployed to a CEP server.

2.2.2.1.8.1.1 GetTypeDefinitionRequest SOAP Header

The SOAP header for a GetTypeDefinitionRequest message MUST be empty.

2.2.2.1.8.1.2 GetTypeDefinitionRequest SOAP Body

The following elements can be set in the SOAP body of a GetTypeDefinitionRequest message.

Element	Type	Description
TypeIdentifier	TypeIdentifier	The type whose definition is being requested. The **TypeIdentifier** type MUST be specified.
GenericArguments	ArrayOfTypeIdentifier	An array of type identifiers that specify the generic type parameters of the requested type. These type identifiers MUST be specified if the type requested has type parameters. Otherwise, type identifiers MUST NOT be specified.

2.2.2.1.8.2 GetResponse

The GetResponse Message (section 2.2.2.1.2.2) is used to respond to a GetEntityTypeRequest message. The **GetResponse** message MUST contain a TypeRoot element.

2.2.2.2 Diagnostic Methods

The following methods are used for diagnosing system health or system performance by monitoring the individual objects and their resource usage from the complex event processing (CEP) engine.

2.2.2.2.1 GetDiagnosticSettings Message

The **GetDiagnosticSettings** message is used to request the retrieval and to receive the response of the current diagnostic settings that are in effect.

2.2.2.2.1.1 GetDiagnosticSettingsRequest

The **GetDiagnosticSettingsRequest** message is used to request the retrieval of the current diagnostic settings that are in effect for a specific named CEP metadata object.

2.2.2.2.1.1.1 GetDiagnosticSettingsRequest SOAP Header

The following elements MUST be set in the SOAP header of a GetDiagnosticSettingsRequest message.

Element	Type	Description
wsa10:Action	**xs:string**	The ~~value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/GetDiagnosticSettings**~~.~~
tns:Name	**xs:anyURI**	This URI MUST be set to the name of the CEP metadata object for which the diagnostic settings will be retrieved.

2.2.2.2.1.1.2 GetDiagnosticSettingsRequest SOAP Body

The SOAP body for a GetDiagnosticSettingsRequest message MUST be empty.

2.2.2.2.1.2 GetDiagnosticSettingsResponse

The **GetDiagnosticSettingsResponse** message MUST be sent by the complex event processing (CEP) server in response to a received GetDiagnosticSettingsRequest message.

2.2.2.2.1.2.1 GetDiagnosticSettingsResponse SOAP Header

The following elements MUST be set in the SOAP header of the GetDiagnosticSettingsResponse message.

Element	Type	Description
wsa10:Action	**xs:string**	The~~ value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/GetDiagnosticSettingsResponse**~~.~~

2.2.2.2.1.2.2 GetDiagnosticSettingsResponse SOAP Body

The following elements MUST be contained in the SOAP body of the GetDiagnosticSettingsResponse message.

Element	Type	Description
GetDiagnosticSettingsResponse	**GetDiagnosticSettingsResponse**	This element contains the diagnostic settings currently in effect. For more information, see section 2.2.3.3.2.

2.2.2.2.1.2.3 Faults

The response to the GetDiagnosticSettingsRequest message ~~may~~can be one of the following faults:

- InvalidNameFault
- GetDiagnosticSettingsNotSupportedFault
- ManagementFault

For a description of the content of the fault return result, see section 2.2.2.3.

2.2.2.2.1.3 GetDiagnosticSettings Examples

The following examples show a ~~client's~~client’s GetDiagnosticSettingsRequest message and the complex event processing (CEP) ~~server's~~server’s GetDiagnosticSettingsResponse message that is sent in response to the received **GetDiagnosticSettingsRequest** message.

2.2.2.2.1.3.1 GetDiagnosticSettingsRequest

The following example **GetDiagnosticSettingsRequest** message is a request to retrieve the Diagnostic settings for the URI specified in the **h:Name** element, "cep:/Server/Application/app1/Query/Select1".

```
      <s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
                  xmlns:s="http://www.w3.org/2003/05/soap-envelope">
        <s:Header>
          <a:Action s:mustUnderstand="1">

    http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/GetDiagnosticSettings
          </a:Action>
          <h:Name s:mustUnderstand="1"

    xmlns:h="http://schemas.microsoft.com/ComplexEventProcessing/2009/10/management">
            cep:/Server/Application/app1/Query/Select1</h:Name>
          <a:MessageID>urn:uuid:2fb6989f-7078-4f84-89da-23c6135142e1</a:MessageID>
          <ActivityId CorrelationId="a4afc40d-4927-45a4-84c1-fd2295137fbd"
            xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
            00000000-0000-0000-0000-000000000000</ActivityId>
          <a:ReplyTo>
            <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
          </a:ReplyTo>
        </s:Header>
        <s:Body></s:Body>
      </s:Envelope>
```

2.2.2.2.1.3.2 GetDiagnosticSettingsResponse

The following example shows the **GetDiagnosticSettingsResponse** message that is sent by the server in response to the preceding received GetDiagnosticSettingsRequest message.

```
      <s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
                  xmlns:s="http://www.w3.org/2003/05/soap-envelope">
        <s:Header>
          <a:Action s:mustUnderstand="1">

    http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/GetDiagnosticSettingsR
    esponse
          </a:Action>
        </s:Header>
        <s:Body>
          <GetDiagnosticSettingsResponse
            xmlns="http://schemas.microsoft.com/ComplexEventProcessing/2009/10/management">
            <DiagnosticAspects>Memory</DiagnosticAspects>
            <DiagnosticLevel>Critical</DiagnosticLevel>
          </GetDiagnosticSettingsResponse>
        </s:Body>
      </s:Envelope>
```

2.2.2.2.2 SetDiagnosticSettings

A **SetDiagnosticSettings** message is used to set the diagnostic settings on a specified CEP metadata object and to receive the response.

2.2.2.2.2.1 SetDiagnosticSettingsRequest

The **SetDiagnosticSettingsRequest** message is used to set diagnostic settings on a CEP metadata object.

2.2.2.2.2.1.1 SetDiagnosticSettingsRequest SOAP Header

The following elements MUST be set in the SOAP header of the SetDiagnosticSettingsRequest message.

Element	Type	Description
wsa10:Action	**xs:string**	The~~ value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/SetDiagnosticSettings**~~.~~
tns:Name	**xs:anyURI**	This URI MUST be set to the name of the CEP metadata object for which the diagnostic settings are being set. Valid objects are the server object or a query object.

2.2.2.2.2.1.2 SetDiagnosticSettingsRequest SOAP Body

The following element MUST be contained in the SOAP body of the SetDiagnosticSettingsRequest message.

Element	Type	Description
SetDiagnosticSettings	**SetDiagnosticSettings**	This element contains the settings that will be instantiated on the complex event processing (CEP) server. For more information, see section 2.2.3.3.1.

2.2.2.2.2.2 SetDiagnosticSettingsResponse

The **SetDiagnosticSettingsResponse** message MUST be sent by the complex event processing (CEP) server in response to a received SetDiagnosticSettingsRequest message.

2.2.2.2.2.2.1 SetDiagnosticSettingsResponse SOAP Header

The following element MUST be set in the SOAP header of the SetDiagnosticSettingsResponse message.

Element	Type	Description
wsa10:Action	**xs:string**	The~~ value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/SetDiagnosticSettingsResponse**~~.~~

2.2.2.2.2.2.2 SetDiagnosticSettingsResponse SOAP Body

The SOAP body for the SetDiagnosticSettingsResponse message MUST be empty.

2.2.2.2.2.2.3 Faults

The response to the SetDiagnosticSettings message ~~may~~can be one of the following faults:

- InvalidNameFault
- SetDiagnosticSettingsNotSupportedFault

- ManagementFault

For a description of the content of the fault return result, see section 2.2.2.3.

2.2.2.2.2.3 SetDiagnosticSettings Examples

The following examples show a ~~client's~~client’s SetDiagnosticSettingsRequest message and the complex event processing (CEP) ~~server's~~server’s SetDiagnosticSettingsResponse message that is sent in response to the received **SetDiagnosticSettingsRequest** message.

2.2.2.2.2.3.1 SetDiagnosticSettingsRequest

The following example **SetDiagnosticSettingsRequest** message is an instruction from the client to set the **DiagnosticSettings** values for the URI specified in the **h:Name** element of the SOAP header, "cep:/Server/Application/app1/Query/MulticastUnionQuery".

```
  <s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
              xmlns:s="http://www.w3.org/2003/05/soap-envelope">
    <s:Header>
      <a:Action s:mustUnderstand="1">

http://schemas.microsoft.com/ComplexEventProcessing/2009/10/management/SetDiagnosticSettings
      </a:Action>
      <h:Name s:mustUnderstand="1" xmlns:h=
              "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/management">
        cep:/Server/Application/app1/Query/MulticastUnionQuery</h:Name>
      <a:MessageID>urn:uuid:3447fce4-6a9e-477b-9a03-ba1b5781937d</a:MessageID>
      <ActivityId CorrelationId="8ab2b897-04dd-443d-98f8-7bcef440995a"
         xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
        00000000-0000-0000-0000-000000000000</ActivityId>
      <a:ReplyTo>
        <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
      </a:ReplyTo>
    </s:Header>
    <s:Body>
      <SetDiagnosticSettings
xmlns="http://schemas.microsoft.com/ComplexEventProcessing/2009/10/management">
        <DiagnosticAspects>Memory</DiagnosticAspects>
        <DiagnosticLevel>Critical</DiagnosticLevel>
      </SetDiagnosticSettings>
    </s:Body>
  </s:Envelope>
```

2.2.2.2.2.3.2 SetDiagnosticSettingsResponse

The following example shows the **SetDiagnosticSettingsResponse** message that is sent by the server in response to the preceding received SetDiagnosticSettingsRequest message.

```
  <s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
              xmlns:s="http://www.w3.org/2003/05/soap-envelope">
    <s:Header>
      <a:Action s:mustUnderstand="1">

http://schemas.microsoft.com/ComplexEventProcessing/2009/10/management/SetDiagnosticSettingsR
esponse
      </a:Action>
    </s:Header>
    <s:Body></s:Body>
  </s:Envelope>
```

2.2.2.2.3 ClearDiagnosticSettings

A **ClearDiagnosticSettings** message is used to request and receive the response for the clearing of the diagnostic settings from an object for which **DiagnosticSettings** values had previously been set.

2.2.2.2.3.1 ClearDiagnosticSettingsRequest

The **ClearDiagnosticSettingsRequest** message is used to clear the diagnostic settings that were previously instantiated for a Query object. The new settings in effect are inherited from the parent object; otherwise, the default settings are used.

2.2.2.2.3.1.1 ClearDiagnosticSettingsRequest SOAP Header

The following elements MUST be set in the SOAP header of a ClearDiagnosticSettingsRequest message.

Element	Type	Description
wsa10:Action	**xs:string**	The~~ value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/ClearDiagnosticSettings**~~.~~
tns:Name	**xs:anyURI**	This URI MUST be set to the name of the object for which the diagnostic settings will be cleared. Valid objects are the server object or a query object.

2.2.2.2.3.1.2 ClearDiagnosticSettingsRequest SOAP Body

The SOAP body for a ClearDiagnosticSettingsRequest message MUST be empty.

2.2.2.2.3.2 ClearDiagnosticSettingsResponse

The **ClearDiagnosticSettingsResponse** message MUST be sent by the CEP server in response to a received ClearDiagnosticSettingsRequest message.

2.2.2.2.3.2.1 ClearDiagnosticSettingsResponse SOAP Header

The following element MUST be set in the SOAP header of a ClearDiagnosticSettingsResponse message.

Element	Type	Description
wsa10:Action	**xs:string**	The~~ value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/ClearDiagnosticSettingsResponse**~~.~~

2.2.2.2.3.2.2 ClearDiagnosticSettingsResponse SOAP Body

The SOAP body for a ClearDiagnosticSettingsResponse message MUST be empty.

2.2.2.2.3.2.3 Faults

The response to the ClearDiagnosticSettings message ~~may~~can be one of the following faults:

- InvalidNameFault
- ClearDiagnosticSettingsNotSupportedFault
- ManagementFault

For a description of the content of the **Fault** return result, see section 2.2.2.3.

2.2.2.2.3.3 ClearDiagnosticSettings Examples

The following examples show a ~~client's~~client’s ClearDiagnosticSettingsRequest message and the complex event processing (CEP) ~~server's~~server’s ClearDiagnosticSettingsResponse message that is sent in response to the received **ClearDiagnosticSettingsRequest** message.

2.2.2.2.3.3.1 ClearDiagnosticSettingsRequest

The following example **ClearDiagnosticSettingsRequest** message is an instruction from the client to clear the diagnostic settings from the URI specified in **h:Name**, "cep:/Server/Application/app1/Query/MulticastUnionQuery," which had previously been set.

```
  <s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
              xmlns:s="http://www.w3.org/2003/05/soap-envelope">
    <s:Header>
      <a:Action s:mustUnderstand="1">

http://schemas.microsoft.com/ComplexEventProcessing/2009/10/management/ClearDiagnosticSetting
s
      </a:Action>
      <h:Name s:mustUnderstand="1"

xmlns:h="http://schemas.microsoft.com/ComplexEventProcessing/2009/10/management">
        cep:/Server/Application/app1/Query/MulticastUnionQuery</h:Name>
      <a:MessageID>urn:uuid:f609118e-0ed4-46ca-b955-61c3028cbb7a</a:MessageID>
      <ActivityId CorrelationId="77a69c8d-fa71-41dc-ae72-c5b14635b192"
        xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
        00000000-0000-0000-0000-000000000000</ActivityId>
      <a:ReplyTo>
        <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
      </a:ReplyTo>
    </s:Header>
    </s:Envelope>
<s:Body></s:Body>
```

2.2.2.2.3.3.2 ClearDiagnosticSettingsResponse

The following example shows the **ClearDiagnosticSettingsResponse** message that is sent by the server in response to the preceding received ClearDiagnosticSettingsRequest message.

```
  <s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
              xmlns:s="http://www.w3.org/2003/05/soap-envelope">
    <s:Header>
      <a:Action s:mustUnderstand="1">

http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/ClearDiagnosticSetting
sResponse
      </a:Action>
    </s:Header>
    <s:Body></s:Body>
  </s:Envelope>
```

2.2.2.2.4 GetDiagnosticView

A **GetDiagnosticView** message is used to request and receive the content of a diagnostic view that had gathered diagnostic statistics.

2.2.2.2.4.1 GetDiagnosticViewRequest

The **GetDiagnosticViewRequest** message is used to request the observed values for a set of **DiagnosticView** properties that have been previously defined. The list of properties that are returned is variable and depends on the settings that are set by using the SetDiagnosticSettings message and on the type of object for which the diagnostic view is being retrieved.

2.2.2.2.4.1.1 GetDiagnosticViewRequest SOAP Header

The following elements MUST be set in the SOAP header of a GetDiagnosticViewRequest message.

Element	Type	Description
wsa10:Action	**xs:string**	The ~~value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/GetDiagnosticView**~~.~~
tns:Name	**xs:anyURI**	This URI MUST be set to the name of the object for which the diagnostic view statistics are being requested.

2.2.2.2.4.1.2 GetDiagnosticViewRequest SOAP Body

The SOAP body for a GetDiagnosticViewRequest message MUST be empty.

2.2.2.2.4.2 GetDiagnosticViewResponse

The **GetDiagnosticViewResponse** message MUST be sent by the complex event processing (CEP) server in response to a received GetDiagnosticViewRequest message.

2.2.2.2.4.2.1 GetDiagnosticViewResponse SOAP Header

The following element MUST be set in the SOAP header of a GetDiagnosticViewResponse message.

Element	Type	Description
wsa10:Action	**xs:string**	The ~~value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/GetDiagnosticViewResponse**~~.~~

2.2.2.2.4.2.2 GetDiagnosticViewResponse SOAP Body

The following element MUST be contained in the SOAP body of a GetDiagnosticViewResponse message.

Element	Type	Description
GetDiagnosticViewResponse	**GetDiagnosticViewResponse**	This element contains the diagnostic view that shows the properties of a specific object for which the diagnostics are being retrieved. Each property is a name/value pair that provides information about some performance or statistical property of the object. For more information, see section 2.2.3.3.3.

2.2.2.2.4.2.3 Faults

The response to the GetDiagnosticViewRequest message ~~may~~can be one of the following faults:

- InvalidNameFault
- GetDiagnosticViewNotSupportedFault
- ManagementFault

For a description of the content of the fault return result, see section 2.2.2.3.

2.2.2.2.4.3 GetDiagnosticView Examples

The following examples show a ~~client's~~client's GetDiagnosticViewRequest message and the complex event processing (CEP) ~~server's~~server's GetDiagnosticViewResponse message that is sent in response to the received **GetDiagnosticViewRequest** message.

2.2.2.2.4.3.1 GetDiagnosticViewRequest

The following example **GetDiagnosticViewRequest** message is an instruction from the client to retrieve the diagnostic view for the URI in the **h:Name** element of the SOAP header, "cep:/Server/Query".

```
      <s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
                  xmlns:s="http://www.w3.org/2003/05/soap-envelope">
        <s:Header>
          <a:Action s:mustUnderstand="1">

    http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/GetDiagnosticView
          </a:Action>
          <h:Name s:mustUnderstand="1"

    xmlns:h="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/GetDiagnostic
    View">
            cep:/Server/Query</h:Name>
          <a:MessageID>urn:uuid:80128549-da22-4cb5-b04d-f3236aeb12fe</a:MessageID>
          <ActivityId CorrelationId="042b3829-5b49-42f8-9f74-bfae8d517d9a"
            xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
            00000000-0000-0000-0000-000000000000</ActivityId>
          <a:ReplyTo>
            <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
          </a:ReplyTo>
        </s:Header>
        <s:Body></s:Body>
      </s:Envelope>
```

2.2.2.2.4.3.2 GetDiagnosticViewResponse

The following example shows the **GetDiagnosticViewResponse** message that is sent by the server in response to the preceding received GetDiagnosticViewRequest message.

```
  <s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
              xmlns:s="http://www.w3.org/2003/05/soap-envelope">
    <s:Header>
      <a:Action s:mustUnderstand="1">

http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/GetDiagnosticViewRespo
nse
      </a:Action>
    </s:Header>
    <s:Body>
      <GetDiagnosticViewResponse
        xmlns="http://schemas.microsoft.com/ComplexEventProcessing/2009/10/management">
        <View xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
          <Name>cep:/Server/Query</Name>
          <Properties>
            <Property>
              <Name>TotalOperatorCount</Name>
              <Value xmlns:d7p1="http://www.w3.org/2001/XMLSchema"
                     i:type="d7p1:long">0</Value>
            </Property>
            <Property>
              <Name>TotalStreamCount</Name>
              <Value xmlns:d7p1="http://www.w3.org/2001/XMLSchema"
                     i:type="d7p1:long">0</Value>
            </Property>
            <Property>
              <Name>CurrentEventCountInStream</Name>
              <Value xmlns:d7p1="http://www.w3.org/2001/XMLSchema"
                     i:type="d7p1:long">0</Value>
            </Property>
            <Property>
              <Name>TotalEventCountInStream</Name>
              <Value xmlns:d7p1="http://www.w3.org/2001/XMLSchema"
                     i:type="d7p1:long">0</Value>
            </Property>
            <Property>
              <Name>TotalStreamMemoryInKB</Name>
              <Value xmlns:d7p1="http://www.w3.org/2001/XMLSchema"
                     i:type="d7p1:long">0</Value>
            </Property>
            <Property>
              <Name>CurrentEventCountInOperatorSynopsis</Name>
              <Value xmlns:d7p1="http://www.w3.org/2001/XMLSchema"
                     i:type="d7p1:long">0</Value>
            </Property>
            <Property>
              <Name>TotalEventCountProcessedByOperator</Name>
              <Value xmlns:d7p1="http://www.w3.org/2001/XMLSchema"
                     i:type="d7p1:long">22</Value>
            </Property>
            <Property>
              <Name>TotalEventCountOutputedByOperator</Name>
              <Value xmlns:d7p1="http://www.w3.org/2001/XMLSchema"
                     i:type="d7p1:long">22</Value>
            </Property>
            <Property>
              <Name>TotalOperatorMemoryInKB</Name>
              <Value xmlns:d7p1="http://www.w3.org/2001/XMLSchema"
                     i:type="d7p1:long">0</Value>
            </Property>
            <Property>
              <Name>TotalOperatorCpuUsage</Name>
              <Value xmlns:d7p1="http://www.w3.org/2001/XMLSchema"
                     i:type="d7p1:long">0</Value>
            </Property>
          </Properties>
        </View>
      </GetDiagnosticViewResponse>
    </s:Body>
  </s:Envelope>
```

2.2.2.3 Faults

All faults in this protocol return the **s:Fault** element [SOAP1.2-1/2003] in the SOAP body.

2.2.2.3.1 InvalidNameFault Fault

An **InvalidNameFault** message is returned when the system tries to reference a CEP metadata object name or a CEP metadata object type through an invalid URL. A URL is invalid if it refers to an inexistent object type, inexistent collection, inexistent protocol, has filters, or some other elements that are not supported. Note that a valid reference to a non-existing object returns a response containing null.

2.2.2.3.1.1 InvalidNameFault SOAP Header

The following elements MUST be set in the SOAP header of an InvalidNameFault.

Element	Type	Description
wsa10:Action	**xs:string**	The **wsa10:Action** element MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/GetDiagnosticViewResponse**

2.2.2.3.1.2 InvalidNameFault SOAP Body

The InvalidNameFault SOAP body MUST contain an **s:Fault** element as defined in [SOAP1.2-1/2003]. The **s:Fault** element for this protocol MUST contain an **s:Code** element, an **s:Reason** element, and an **s:Detail** element. The following table provides additional information about the elements contained in the **s:Fault** element.

Element	Description
s:Code	The **s:Value** element of the **s:Subcode** element of the **s:Code** element MUST be set to the following value: **a:InvalidNameFault**~~.~~
s:Reason	The **s:Text** element of the **s:Reason** element is set to a human-readable description of the reason for the fault.
s:Detail	The **s:Detail** element MUST contain an **InvalidNameFault** element of type **InvalidNameFault**. For more information, see section 2.2.3.4.1.

2.2.2.3.1.3 InvalidNameFault Example

The following example shows an **InvalidNameFault** element.

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">
      http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/InvalidName
    </a:Action>
    <a:RelatesTo>urn:uuid:d097f723-0a5b-476e-8e55-39472ea6eefd
    </a:RelatesTo>
    <ActivityId CorrelationId="e96ffc8b-dd1c-4e31-b090-f39d2136bc50"
                xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
      00000000-0000-0000-0000-000000000000</ActivityId>
  </s:Header>
  <s:Body>
    <s:Fault>
      <s:Code>
        <s:Value>s:Sender</s:Value>
        <s:Subcode>
          <s:Value xmlns:a=

"http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management">
            a:InvalidNameFault</s:Value>
        </s:Subcode>
      </s:Code>
      <s:Reason>
        <s:Text xml:lang="en-US">The argument cannot be null.</s:Text>
      </s:Reason>
      <s:Detail>
        <InvalidNameFault xmlns=

"http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management"
                          xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
          <Message>The argument cannot be null.</Message>
        </InvalidNameFault>
      </s:Detail>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

2.2.2.3.2 InvalidDefinitionFault Fault

An **InvalidDefinitionFault** message is returned when an attempt is made to create a CEP metadata object, and the attempted definition is invalid.

2.2.2.3.2.1 InvalidDefinitionFault SOAP Header

The following elements MUST be set in the SOAP header of an InvalidDefinitionFault.

Element	Type	Description
wsa10:Action	**xs:string**	The **wsa10:Action** element MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/InvalidDefinition**

2.2.2.3.2.2 InvalidDefinitionFault SOAP Body

The InvalidDefinitionFault SOAP body MUST contain an **s:Fault** element as defined in [SOAP1.2-1/2003]. The **s:Fault** element for this protocol MUST contain an **s:Code** element, an **s:Reason** element, and an **s:Detail** element. The following table provides additional information about the elements contained in the **s:Fault** element.

Element	Description
s:Code	The **s:Value** element of the **s:Subcode** element of the **s:Code** element MUST be set to the following value: **a:InvalidDefinitionFault**~~.~~
s:Reason	The **s:Text** element of the **s:Reason** element is set to a human-readable description of the reason for the fault.
s:Detail	The **s:Detail** element MUST contain an **InvalidDefinitionFault** element of type **InvalidDefinitionFault**. For more information, see section 2.2.3.4.2.

2.2.2.3.2.3 InvalidDefinitionFault Example

The following example shows an **InvalidDefinitionFault** element.

```
  <s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
              xmlns:a="http://www.w3.org/2005/08/addressing">
    <s:Header>
      <a:Action s:mustUnderstand="1">

http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/InvalidDefinition
      </a:Action>
      <a:RelatesTo>urn:uuid:4e484014-b7c7-42b0-9a60-c335441bd1e7</a:RelatesTo>
      <ActivityId CorrelationId="519e87a1-edb0-4f8a-9825-be09882a2934"
               xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
        00000000-0000-0000-0000-000000000000</ActivityId>
    </s:Header>
    <s:Body>
      <s:Fault>
        <s:Code>
          <s:Value>s:Sender</s:Value>
          <s:Subcode>
            <s:Value xmlns:a=

"http://schemas.microsoft.com/ComplexEventProcessing/2009/10/management">
              a:InvalidDefinitionFault</s:Value>
          </s:Subcode>
        </s:Code>
        <s:Reason>
          <s:Text xml:lang="en-US">The definition is not valid: The
   'http://schemas.microsoft.com/ComplexEventProcessing/2010/01/Metadata:Applicationss'
          element is not declared.-->The
   'http://schemas.microsoft.com/ComplexEventProcessing/2010/01/Metadata:Applicationss'
          element is not declared.</s:Text>
        </s:Reason>
        <s:Detail>
          <InvalidDefinitionFault
            xmlns="http://schemas.microsoft.com/ComplexEventProcessing/2009/10/management"
            xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
            <Message>The
   'http://schemas.microsoft.com/ComplexEventProcessing/2010/01/Metadata:Applicationss'
            element is not declared.</Message>
          </InvalidDefinitionFault>
        </s:Detail>
      </s:Fault>
    </s:Body>
  </s:Envelope>
```

2.2.2.3.3 ManagementFault Fault

A **ManagementFault** message is returned whenever a generic error happens in any of the manageability operations on objects in the complex event processing (CEP) system.

2.2.2.3.3.1 ManagementFault SOAP Header

The following elements MUST be set in the SOAP header of a ManagementFault message.

Element	Type	Description
wsa10:Action	**xs:string**	The **wsa10:Action** element MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/Fault**

2.2.2.3.3.2 ManagementFault SOAP Body

The ManagementFault SOAP body MUST contain an **s:Fault** element as defined in [SOAP1.2-1/2003]. The **s:Fault** element for this protocol MUST contain an **s:Code** element, an **s:Reason** element, and an **s:Detail** element. The following table provides additional information about the elements contained in the **s:Fault** element.

Element	Description
s:Code	The **s:Value** element of the **s:Subcode** element of the **s:Code** element MUST be set to the following value: **a:ManagementFault**~~.~~
s:Reason	The **s:Text** element of the **s:Reason** element is set to a human-readable description of the reason for the fault.
s:Detail	The **s:Detail** element MUST contain a **ManagementFault** element of type **ManagementFault**. For more information, see section 2.2.3.4.3.

2.2.2.3.3.3 ManagementFault Example

The following example shows a ManagementFault message.

```
      <s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
                  xmlns:a="http://www.w3.org/2005/08/addressing">
        <s:Header>
          <a:Action s:mustUnderstand="1">

    http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/Fault</a:Action>
          <a:RelatesTo>urn:uuid:dbcfedb6-4198-437a-a086-541e07860aba</a:RelatesTo>
          <ActivityId CorrelationId="49acac6d-423c-4f2a-8747-d0b7dedb5056"
                  xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
            00000000-0000-0000-0000-000000000000</ActivityId>
        </s:Header>
        <s:Body>
          <s:Fault>
            <s:Code>
              <s:Value>s:Sender</s:Value>
              <s:Subcode>
                <s:Value xmlns:a=

    "http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management">
                  a:ManagementFault</s:Value>
              </s:Subcode>
            </s:Code>
            <s:Reason>
              <s:Text xml:lang="en-US">The management service encountered an error:
              Delete operation failed.-->The address 'bogus:/app/address' is not valid.
              Additional information: Invalid scheme 'bogus'.</s:Text>
            </s:Reason>
            <s:Detail>
              <ManagementFault
    xmlns="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management"
                               xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
                <Message>Delete operation failed.</Message>
              </ManagementFault>
            </s:Detail>
          </s:Fault>
        </s:Body>
      </s:Envelope>
```

2.2.2.3.4 RuntimeFault Fault

A **RuntimeFault** message is returned whenever a generic error happens during the runtime operation of the complex event processing (CEP) system.

2.2.2.3.4.1 RuntimeFault SOAP Header

The following element MUST be set in the SOAP header of a RuntimeFault.

Element	Type	Description
wsa10:Action	**xs:string**	The **wsa10:Action** element MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/RuntimeFailure**

2.2.2.3.4.2 RuntimeFault SOAP Body

The RuntimeFault SOAP body MUST contain an **s:Fault** element as defined in [SOAP1.2-1/2003]. The **s:Fault** element for this protocol MUST contain an **s:Code** element, an **s:Reason** element, and an **s:Detail** element. The following table provides additional information about the elements contained in the **s:Fault** element.

Element	Description
s:Code	The **s:Value** element of the **s:Subcode** element of the **s:Code** element MUST be set to the following value: **a:RuntimeFault**~~.~~
s:Reason	The **s:Text** element of the **s:Reason** element is set to a human-readable description of the reason for the fault.
s:Detail	The **s:Detail** element MUST contain a **RuntimeFault** element of type **RuntimeFault**. For more information, see section 2.2.3.4.4.

2.2.2.3.4.3 RuntimeFault Example

The following example shows a RuntimeFault element.

```
  <s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
              xmlns:a="http://www.w3.org/2005/08/addressing">
    <s:Header>
      <a:Action s:mustUnderstand="1">

http://schemas.microsoft.com/ComplexEventProcessing/2009/05/management/RuntimeFailure</a:Acti
on>
      <a:RelatesTo>urn:uuid:b65a1aab-4673-4e51-92a7-97f46c59031e</a:RelatesTo>
      <ActivityId CorrelationId="f1f8f11f-3b9f-4501-98a7-7e2ea485f412"
           xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
        00000000-0000-0000-0000-000000000000</ActivityId>
    </s:Header>
    <s:Body>
      <s:Fault>
        <s:Code>
          <s:Value>s:Sender</s:Value>
          <s:Subcode>
            <s:Value
xmlns:a="http://schemas.microsoft.com/ComplexEventProcessing/2009/10/management">
              a:RuntimeFault</s:Value>
          </s:Subcode>
        </s:Code>
        <s:Reason>
          <s:Text xml:lang="en-US">There was an error in the runtime:
          Get operation failed.-->The address 'bogus:/app/address' is not valid.
          Additional information: Invalid scheme 'bogus'.</s:Text>
        </s:Reason>
        <s:Detail>
          <RuntimeFault
xmlns="http://schemas.microsoft.com/ComplexEventProcessing/2009/10/management"
                        xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
            <Message>Get operation failed.</Message>
          </RuntimeFault>
        </s:Detail>
      </s:Fault>
    </s:Body>
  </s:Envelope>
```

2.2.2.3.5 GetDiagnosticSettingsNotSupported Fault

A **GetDiagnosticSettingsNotSupported** message is returned when an attempt is made to get diagnostic settings from objects for which this operation is not supported.

2.2.2.3.5.1 GetDiagnosticSettingsNotSupported SOAP Header

The following element MUST be set in the SOAP header of a GetDiagnosticSettingsNotSupported message.

Element	Type	Description
wsa10:Action	**xs:string**	The **wsa10:Action** element MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetDiagnosticSettingsNotSupported**

2.2.2.3.5.2 GetDiagnosticSettingsNotSupported SOAP Body

The GetDiagnosticSettingsNotSupported SOAP body MUST contain an **s:Fault** element as defined in [SOAP1.2-1/2003]. The **s:Fault** element for this protocol MUST contain an **s:Code** element, an **s:Reason** element, and an **s:Detail** element. The following table provides additional information about the elements contained in the **s:Fault** element.

Element	Description
s:Code	The **s:Value** element of the **s:Subcode** element of the **s:Code** element MUST be set to the following value: **a:GetDiagnosticSettingsNotSupported**~~.~~
s:Reason	The **s:Text** element of the **s:Reason** element is set to a human-readable description of the reason for the fault.
s:Detail	The **s:Detail** element MUST contain a **GetDiagnosticSettingsNotSupported** element of type **GetDiagnosticSettingsNotSupported**. For more information, see section 2.2.3.4.5.

2.2.2.3.5.3 GetDiagnosticSettingsNotSupported Example

The following example shows a GetDiagnosticSettingsNotSupported element.

```
    <s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
                xmlns:a="http://www.w3.org/2005/08/addressing">
      <s:Header>
        <a:Action s:mustUnderstand="1">
          http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/
          GetDiagnosticSettingsNotSupported</a:Action>
        <a:RelatesTo>urn:uuid:9f6b03b5-a6c1-475b-b900-819bf34b8bf8</a:RelatesTo>
        <ActivityId CorrelationId="de4aa3e7-da65-4661-ab49-f9aabef3ed4a"
                 xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
          00000000-0000-0000-0000-000000000000</ActivityId>
      </s:Header>
      <s:Body>
        <s:Fault>
          <s:Code>
            <s:Value>s:Sender</s:Value>
            <s:Subcode>
              <s:Value xmlns:a=
              "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
                a:GetDiagnosticSettingsNotSupported</s:Value>
            </s:Subcode>
          </s:Code>
          <s:Reason>
            <s:Text xml:lang="en-US">Getting the diagnostic settings for
  'cep:/Server/Application/app/EventType'
            is not supported.</s:Text>
          </s:Reason>
          <s:Detail>
            <GetDiagnosticSettingsNotSupported xmlns=
              "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management"
               xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
              <Message>Getting the diagnostic settings for
  'cep:/Server/Application/app/EventType'
              is not supported.</Message>
              <Name>cep:/Server/Application/app/EventType</Name>
            </GetDiagnosticSettingsNotSupported>
          </s:Detail>
        </s:Fault>
      </s:Body>
    </s:Envelope>
```

2.2.2.3.6 SetDiagnosticSettingsNotSupported Fault

A **SetDiagnosticSettingsNotSupported** message is returned when an attempt is made to set diagnostic settings from objects for which this operation is not supported.

2.2.2.3.6.1 SetDiagnosticSettingsNotSupported SOAP Header

The following element MUST be set in the SOAP header of a SetDiagnosticSettingsNotSupported message.

Element	Type	Description
wsa10:Action	**xs:string**	The **wsa10:Action** element MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/SetDiagnosticSettingsNotSupported**

2.2.2.3.6.2 SetDiagnosticSettingsNotSupported SOAP Body

The SetDiagnosticSettingsNotSupported SOAP body MUST contain an **s:Fault** element as defined in [SOAP1.2-1/2003]. The **s:Fault** element for this protocol MUST contain an **s:Code** element, an **s:Reason** element, and an **s:Detail** element. The following table provides additional information about the elements contained in the **s:Fault** element.

Element	Description
s:Code	The **s:Value** element of the **s:Subcode** element of the **s:Code** element MUST be set to the following value: **a:SetDiagnosticSettingsNotSupported**~~.~~
s:Reason	The **s:Text** element of the **s:Reason** element is set to a human-readable description of the reason for the fault.
s:Detail	The **s:Detail** element MUST contain a **SetDiagnosticSettingsNotSupported** element of type **GetDiagnosticSettingsNotSupported**. For more information, see section 2.2.3.4.5.

2.2.2.3.6.3 SetDiagnosticSettingsNotSupported Example

The following example shows a SetDiagnosticSettingsNotSupported element.

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">
      http://schemas.microsoft.com/ComplexEventProcessing/2009/05/
      Management/SetDiagnosticSettingsNotSupported</a:Action>
    <a:RelatesTo>urn:uuid:51b46784-947e-49da-89d8-2d69bee43e6d</a:RelatesTo>
    <ActivityId CorrelationId="3aa972ea-0ba3-4e67-82e0-b0726d883412"
            xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
      00000000-0000-0000-0000-000000000000</ActivityId>
  </s:Header>
  <s:Body>
    <s:Fault>
      <s:Code>
        <s:Value>s:Sender</s:Value>
        <s:Subcode>
          <s:Value xmlns:a=
            "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
            a:SetDiagnosticSettingsNotSupported</s:Value>
        </s:Subcode>
      </s:Code>
      <s:Reason>
        <s:Text xml:lang="en-US">Setting the diagnostic settings for
```

```
            'cep:/Server/Application/app1/EventType' is not supported.</s:Text>
          </s:Reason>
          <s:Detail>
            <GetDiagnosticSettingsNotSupported xmlns=
                "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management"
                 xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
              <Message>Setting the diagnostic settings for
              'cep:/Server/Application/app1/EventType' is not supported.</Message>
              <Name>cep:/Server/Application/app1/EventType</Name>
            </GetDiagnosticSettingsNotSupported>
          </s:Detail>
        </s:Fault>
      </s:Body>
    </s:Envelope>
```

2.2.2.3.7 ClearDiagnosticSettingsNotSupported Fault

A **ClearDiagnosticSettingsNotSupported** message is returned when an attempt is made to clear diagnostic settings from objects for which this operation is not supported.

2.2.2.3.7.1 ClearDiagnosticSettingsNotSupported SOAP Header

The following element MUST be set in the SOAP header of a ClearDiagnosticSettingsNotSupported message.

Element	Type	Description
wsa10:Action	**xs:string**	The **wsa10:Action** element MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/ClearDiagnosticSettingsNotSupported**

2.2.2.3.7.2 ClearDiagnosticSettingsNotSupported SOAP Body

The ClearDiagnosticSettingsNotSupported SOAP body MUST contain an **s:Fault** element as defined in [SOAP1.2-1/2003]. The **s:Fault** element for this protocol MUST contain an **s:Code** element, an **s:Reason** element, and an **s:Detail** element. The following table provides additional information about the elements contained in the **s:Fault** element.

Element	Description
s:Code	The **s:Value** element of the **s:Subcode** element of the **s:Code** element MUST be set to the following value: **a:ClearDiagnosticSettingsNotSupported**~~.~~
s:Reason	The **s:Text** element of the **s:Reason** element is set to a human-readable description of the reason for the fault.
s:Detail	The **s:Detail** element MUST contain a **ClearDiagnosticSettingsNotSupported** element of type **ClearDiagnosticSettingsNotSupported**. For more information, see section 2.2.3.4.6.

2.2.2.3.7.3 ClearDiagnosticSettingsNotSupported Example

The following example shows a ClearDiagnosticSettingsNotSupported element.

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">
      http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management
      /ClearDiagnosticSettingsNotSupported</a:Action>
    <a:RelatesTo>urn:uuid:056eaafc-40f3-4757-94db-deea457220a9</a:RelatesTo>
    <ActivityId CorrelationId="ef5b4a0f-8102-41e9-bafe-f8cdf5f80b6e"
          xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
      00000000-0000-0000-0000-000000000000</ActivityId>
  </s:Header>
  <s:Body>
    <s:Fault>
      <s:Code>
        <s:Value>s:Sender</s:Value>
        <s:Subcode>
          <s:Value xmlns:a=
           "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
            a:ClearDiagnosticSettingsNotSupported</s:Value>
        </s:Subcode>
      </s:Code>
      <s:Reason>
        <s:Text xml:lang="en-US">Clearing the diagnostic settings for
        'cep:/Server/Application/app1/EventType' is not supported.</s:Text>
      </s:Reason>
      <s:Detail>
        <ClearDiagnosticSettingsNotSupported xmlns=
          "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management"
           xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
          <Message>Clearing the diagnostic settings for
          'cep:/Server/Application/app1/EventType' is not supported.</Message>
          <Name>cep:/Server/Application/app1/EventType</Name>
        </ClearDiagnosticSettingsNotSupported>
      </s:Detail>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

2.2.2.3.8 GetDiagnosticViewNotSupported Fault

A **GetDiagnosticViewNotSupported** message is returned when an attempt is made to get a diagnostic view from objects for which this operation is not supported.

2.2.2.3.8.1 GetDiagnosticViewNotSupported SOAP Header

The following element MUST be set in the SOAP header of a GetDiagnosticViewNotSupported message.

Element	Type	Description
wsa10:Action	**xs:string**	The **wsa10:Action** element MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetDiagnosticViewNotSupported**

2.2.2.3.8.2 GetDiagnosticViewNotSupportedFault SOAP Body

The GetDiagnosticViewNotSupported SOAP body MUST contain an **s:Fault** element as defined in [SOAP1.2-1/2003]. The **s:Fault** element for this protocol MUST contain an **s:Code** element, an **s:Reason** element, and an **s:Detail** element. The following table provides additional information about the elements contained in the **s:Fault** element.

Element	Description
s:Code	The **s:Value** element of the **s:Subcode** element of the **s:Code** element MUST be set to the following value: **a:GetDiagnosticViewNotSupported**~~.~~
s:Reason	The **s:Text** element of the **s:Reason** element is set to a human-readable description of the reason for the fault.
s:Detail	The **s:Detail** element MUST contain a **GetDiagnosticViewNotSupported** element of type **GetDiagnosticViewNotSupported**. For more information, see section 2.2.3.4.7.

2.2.2.3.8.3 GetDiagnosticViewNotSupported Example

The following example shows a GetDiagnosticViewNotSupported fault message.

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">
      http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management
      /GetDiagnosticViewNotSupported
    </a:Action>
    <a:RelatesTo>urn:uuid:056eaafc-40f3-4757-94db-deea457220a9</a:RelatesTo>
    <ActivityId CorrelationId="ef5b4a0f-8102-41e9-bafe-f8cdf5f80b6e"
      xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">
      00000000-0000-0000-0000-000000000000
    </ActivityId>
  </s:Header>
  <s:Body>
    <s:Fault>
      <s:Code>
        <s:Value>s:Sender</s:Value>
        <s:Subcode>
          <s:Value xmlns:a=
    "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management">
            a:GetDiagnosticViewNotSupported
          </s:Value>
        </s:Subcode>
      </s:Code>
      <s:Reason>
        <s:Text xml:lang="en-US">
          Request the diagnostic view for
          'cep:/Server/Application/app1/EventType' is not supported.
        </s:Text>
      </s:Reason>
      <s:Detail>
        <GetDiagnosticViewNotSupported xmlns=
    "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management"
           xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
          <Message>
            Request the diagnostic view for
            'cep:/Server/Application/app1/EventType' is not supported.
          </Message>
          <Name>cep:/Server/Application/app1/EventType</Name>
        </GetDiagnosticViewNotSupported>
      </s:Detail>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

2.2.2.4 Administrative Methods

2.2.2.4.1 Checkpoint Message

The **Checkpoint** message is used to take a checkpoint of a Query object that is running on a complex event processing (CEP) server and to receive the response to this message.

2.2.2.4.1.1 CheckpointRequest Message

The **CheckpointRequest** message is used to request that a checkpoint of a query be taken.

2.2.2.4.1.1.1 CheckpointRequest SOAP Header

The following elements MUST be set in the SOAP header of the CheckpointRequest message.

Element	Type	Description
wsa10:Action	**xs:string**	The~~ value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Checkpoint**~~.~~
<tns:Name>	**xs:anyURI**	The URI of the Query object for the query to take a checkpoint of.

2.2.2.4.1.1.2 CheckpointRequest SOAP Body

The SOAP body for a **CheckpointRequest** message MUST be empty.

2.2.2.4.1.2 CheckpointResponse Message

The **CheckpointResponse** message MUST be sent by the server in response to a received CheckpointRequest message, unless there is an exception or a fault.

2.2.2.4.1.2.1 CheckpointResponse SOAP Header

Element	Type	Description
wsa10:Action	**xs:string**	The~~ value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/CheckpointResponse**~~.~~

2.2.2.4.1.2.2 CheckpointResponse SOAP Body

Element	Type	Description
CheckpointResponse	<xs:sequence> <xs:element minOccurs="0" name="Result" type="xs:boolean" /> </xs:sequence>	The **CheckpointResponse** element specifies whether the checkpoint was or was not successful. See section 2.2.2.4.1.4.2 for an example.

2.2.2.4.1.3 CheckpointRequest Faults

The response to the CheckpointRequest message can be one of the following faults:

- InvalidNameFault
- ManagementFault

For a description of the content of the fault return result, see Faults (section 2.2.2.3).

2.2.2.4.1.4 Checkpoint Examples

The following examples show a ~~client's~~client’s CheckpointRequest message and the CEP ~~server's~~server’s CheckpointResponse message that is sent in response.

2.2.2.4.1.4.1 CheckpointRequest Message Example

The following example CheckpointRequest message is an instruction from the client to begin a checkpoint of the query with the URI in the **h:Name** element, "cep:/Server/Application/app/Query/query."

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
            xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action s:mustUnderstand="1">
      http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Checkpoint
    </a:Action>
    <h:Name xmlns:h=
            "http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management">
      cep:/Server/Application/app/Query/query</h:Name>
    <a:MessageID>urn:uuid:0b03cb8c-6f42-425a-93e7-679bce853a40</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
  </s:Header>
  <s:Body></s:Body>
</s:Envelope>
```

2.2.2.4.1.4.2 CheckpointResponse Message Example

The following CheckpointResponse message is an example response from the server to the preceding **CheckpointRequest** message.

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
            xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action s:mustUnderstand="1">
      http://schemas.microsoft.com/ComplexEventProcessing/
      2009/05/Management/CheckpointResponse
    </a:Action>
  </s:Header>
  <s:Body>
    <CheckpointResponse xmlns=
                        "http://schemas.microsoft.com/ComplexEventProcessing/
                        2009/05/Management">
      <Result>true</Result>
    </CheckpointResponse>
  </s:Body>
</s:Envelope>
```

2.2.2.4.2 CancelCheckpoint Message

The **CancelCheckpoint** message is used to request that a checkpoint in progress on a CEP server be canceled.

2.2.2.4.2.1 CancelCheckpoint SOAP Header

The following elements MUST be set in the SOAP header of the CancelCheckpoint message.

Element	Type	Description
wsa10:Action	**xs:string**	The~~ value of the~~ **wsa10:Action** element in the SOAP header MUST be set to the following value: **http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/CancelCheckpoint~~.~~**
tns:Name	**xs:anyURI**	The URI of the Query object for the query to take a checkpoint of.

2.2.2.4.2.2 CancelCheckpoint SOAP Body

The SOAP body for a CancelCheckpoint message MUST be empty.

2.2.2.4.2.3 CancelCheckpoint Faults

The response to the CancelCheckpoint message can be one of the following faults:

- InvalidNameFault
- ManagementFault

For a description of the content of the fault return result, see section 2.2.2.3.

2.2.2.4.2.4 CancelCheckpoint Examples

The following example **CancelCheckpoint** message is an instruction from the client to cancel a checkpoint of the query with the URI in the **h:Name** element, "cep:/Server/Application/app/Query/query".

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
            xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action s:mustUnderstand="1">
      http://schemas.microsoft.com/ComplexEventProcessing/2009/05/
      Management/CancelCheckpoint</a:Action>
    <h:Name xmlns:h=
            "http://schemas.microsoft.com/ComplexEventProcessing/
            2009/05/Management">cep:/Server/Application/app/Query/query</h:Name>
    <a:MessageID>urn:uuid:23c0f56a-9d5a-420d-9614-6c70aee9efbc</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
  </s:Header>
  <s:Body></s:Body>
</s:Envelope>
```

2.2.3 Types

The following table summarizes the set of type definitions that are defined by this specification.

Element	Description
CreateRequest	The **CreateRequest** type forms the SOAP body of the CreateRequest message. For more information, see section 2.2.3.1.1.
GetResponse	The **GetResponse** type forms the SOAP body of the GetResponse message. For more information, see section 2.2.3.1.2.
QueryState	The **QueryState** type forms the SOAP body of the ChangeQueryStateRequest message and the ChangeQueryStateResponse message. For more information, see section 2.2.3.1.3.
SetDiagnosticSettings	The **SetDiagnosticSettings** type forms the SOAP body of the SetDiagnosticSettings message. For more information, see section 2.2.3.3.1.
GetDiagnosticSettingsResponse	The **GetDiagnosticSettingsResponse** type forms the SOAP body of the GetDiagnosticSettingsResponse message. For more information, see section 2.2.3.3.2.
GetDiagnosticViewResponse	The **GetDiagnosticViewResponse** type forms the SOAP body of the GetDiagnosticViewResponse message. For more information, see section 2.2.3.3.3.

2.2.3.1 Metadata Method Types

2.2.3.1.1 CreateRequest

The following code is the XML schema (XSD) for the **CreateRequest** complex type.

```
<xs:complexType name="CreateRequest">
  <xs:choice>
    <xs:element minOccurs="1" maxOccurs="1" name="InputAdapter"
                type="metadata:InputAdapterType" />
    <xs:element minOccurs="1" maxOccurs="1" name="OutputAdapter"
                type="metadata:OutputAdapterType" />
    <xs:element minOccurs="1" maxOccurs="1" name="Application"
                type="metadata:ApplicationType" />
    <xs:element minOccurs="1" maxOccurs="1" name="EventType"
                type="metadata:EventType" />
    <xs:element minOccurs="1" maxOccurs="1" name="Query"
                type="metadata:QueryType" />
    <xs:element minOccurs="1" maxOccurs="1" name="QueryTemplate"
                type="metadata:QueryTemplateType" />
    <xs:element minOccurs="1" maxOccurs="1" name="Entity"
                type="metadata:EntityType" />
  </xs:choice>
</xs:complexType>
```

The following table describes the elements that are referenced in the XML schema (XSD).

Element	Type	Description
InputAdapter	InputAdapterType	The definition of an InputAdapter object.

Element	Type	Description
OutputAdapter	OutputAdapterType	The definition of an OutputAdapter object.
Application	ApplicationType	The definition of an Application object.
EventType	EventType	The definition of an EventType object.
Query	QueryType	The definition of a Query object.
QueryTemplate	QueryTemplateType	The definition of a QueryTemplate object.
Entity	EntityType	The definition of an Entity object.

2.2.3.1.2 GetResponse

The following code is the XML schema (XSD) for the **GetResponse** type.

```
<xs:complexType name="GetResponse">
  <xs:choice>
    <xs:element minOccurs="1" maxOccurs="1" name="InputAdapter"
                type="metadata:InputAdapterType" />
    <xs:element minOccurs="1" maxOccurs="1" name="OutputAdapter"
                type="metadata:OutputAdapterType" />
    <xs:element minOccurs="1" maxOccurs="1" name="Application"
                type="metadata:ApplicationType" />
    <xs:element minOccurs="1" maxOccurs="1" name="EventType"
                type="metadata:EventType" />
    <xs:element minOccurs="1" maxOccurs="1" name="Query"
                type="metadata:QueryType" />
    <xs:element minOccurs="1" maxOccurs="1" name="QueryTemplate"
                type="metadata:QueryTemplateType" />
    <xs:element minOccurs="1" maxOccurs="1" name="Entity" type="metadata:EntityType" />
    <xs:element minOccurs="1" maxOccurs="1" name="TypeRoot" type="design:TypeRoot" />
  </xs:choice>
</xs:complexType>
```

The types and descriptions for the **GetResponse** type are identical to those for the CreateRequest type with the addition of the TypeRoot element. For more on those elements that are shared with **GetResponse**, see **CreateRequest** (section 2.2.3.1.1).

Element	Type	Description
TypeRoot	**TypeRoot**	The definition of a **TypeRoot** object.

2.2.3.1.3 QueryState

The following code is the XML schema (XSD) for the **QueryState** type.

```
<xs:simpleType name="QueryState">
  <xs:restriction base="xs:string">
    <xs:enumeration value=
         "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/
         Management/QueryStateStarted" />
    <xs:enumeration value=
         "http://schemas.microsoft.com/ComplexEventProcessing/2009/10/
```

```
            Management/QueryStateStopped" />
    </xs:restriction>
  </xs:simpleType>
```

The following table describes the element that is referenced in the XSD.

Element	Type	Description
QueryState	**xs:string (restriction)**	This enumeration allows the starting and stopping of a query. The possible values are as follows: **http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management/QueryStateStarted**: The query is started and begins consuming and emitting events. **http://schemas.microsoft.com/ComplexEventProcessing/2009/10/Management/QueryStateStopped**: The query is stopped.

2.2.3.2 Metadata Definition Types

2.2.3.2.1 Metadata Object Types

2.2.3.2.1.1 QueryType

A query of type **QueryType** is used to bind together an input stream, an output stream, and a query template.

The following code is the XML schema (XSD) for the **QueryType** type.

```
        <xs:complexType name="QueryType">
            <xs:annotation>
                <xs:documentation>The schema of a CreateQuery command. It contains
                information to bind a query template's input and output streams to
                stream sources and sinks.</xs:documentation>
            </xs:annotation>
            <xs:sequence>
                <xs:element minOccurs="0" maxOccurs="unbounded"
                            name="OutputStreamBinding"
                            type="tns:OutputStreamBindingType" />
                <xs:element minOccurs="1" maxOccurs="unbounded"
                            name="InputStreamBinding"
                            type="tns:InputStreamBindingType" />
            </xs:sequence>
            <xs:attribute name="Name" type="xs:anyURI" use="required" />
            <xs:attribute name="QueryTemplate" type="xs:anyURI" use="required" />
            <xs:attribute name="Description" type="xs:string" use="optional" />
            <xs:attribute name="IsResilient" type="xs:boolean"use="optional" />
        </xs:complexType>
```

The following tables describe the elements and attributes for the **QueryType** type.

Element	Type	Description
OutputStreamBinding	OutputStreamBindingType	This element associates an event sink with a stream export operator in a query template.
InputStreamBinding	InputStreamBindingType	This element associates an event source (an input adapter or another query) with a stream import operator

Element	Type	Description
		in a query template.

Attribute	Type	Description
Name	**xs:anyURI**	The URI by which this query is referenced.
QueryTemplate	**xs:anyURI**	The URI of the QueryTemplate object to which this query is bound.
Description	**xs:string**	A human-readable description that is not processed by the complex event processing (CEP) server.
IsResilient	**xs:boolean**	A flag that determines whether the query ~~should be~~is run in a resilient mode capable of writing query state checkpoints.

2.2.3.2.1.1.1 OutputStreamBindingType

The following is the XML schema (XSD) for the **OutputStreamBindingType** type.

```
<xs:complexType name="OutputStreamBindingType">
    <xs:annotation>
      <xs:documentation>Output Stream Binding. Pairs a stream sink
      with a query template.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="AdapterConfiguration"
                  type="tns:AnySingleUserElementType">
        <xs:annotation>
          <xs:documentation>The contained XML element will be passed to the output
          adapter as initialization information. The child element is serialized
          from user-defined adapter configuration structure and has arity of one.
        </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="OutputStream" type="xs:anyURI" use="required">
      <xs:annotation>
        <xs:documentation>Reference to an export operator name.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="OutputStreamTarget" type="xs:anyURI" use="required">
      <xs:annotation>
        <xs:documentation>Reference to an output adapter.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="OutputStreamConsumerName" type="xs:anyURI" use="optional">
      <xs:annotation>
        <xs:documentation>The unique identifier to identify a given consumer
        of the query.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="EventShape" type="tns:EventShapeType" use="optional">
      <xs:annotation>
        <xs:documentation>Desired event shape in the output.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="StreamEventOrdering" type="tns:StreamEventOrderingType"
                  use="optional">
      <xs:annotation>
        <xs:documentation>Desired time ordering at the output.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="PayloadClassName" type="xs:string" use="optional"/>
  </xs:complexType>
```

The following tables describe the elements and attributes for the **OutputStreamBindingType** type.

Element	Type	Description
AdapterConfiguration	AnySingleUserElementType	This XML element is not interpreted by the complex event processing (CEP) server or by the CEPM protocol. This XML element is passed to the **OutputAdapter** component, which is pointed to by the **OutputStreamTarget** XML attribute on the **OutputStreamBinding** element. This XML element acts as a query startup parameter that can be used to initialize the adapter at run time. It is up to the adapter author as to whether and how to process this piece of XML.

Attribute	Type	Description
OutputStream	**xs:anyURI**	The name of the export operator in a QueryTemplate object.
OutputStreamTarget	**xs:anyURI**	The URI of the stream sink. This can be either another query or an output adapter.
OutputStreamConsumerName	**xs:anyURI**	The name that identifies a given consumer of the query.
StreamEventOrdering	**xs:string**	This enumeration specifies the desired temporal ordering of events in the ~~query's~~query's output stream. "ChainOrdered": An insert event and its associated chain of retraction events are in order in relation to one another, but different inserts can be out of order. "FullyOrdered" (default): All events are fully ordered.
EventShape	**EventShapeType**	The shape of events that an output stream contains.
PayloadClassName	**xs:string**	The class name that contains the output stream binding code.

2.2.3.2.1.1.2 InputStreamBindingType

The **InputStreamBindingType** type specifies the input stream that the query binds to.

The following is the XML schema (XSD) for the **InputStreamBindingType** type.

```
    <xs:complexType name="InputStreamBindingType">
      <xs:annotation>
        <xs:documentation>Input Stream Binding. Pairs a stream
        source with a query template.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element minOccurs="0" maxOccurs="1" name="AdvanceTime"
                    type="tns:AdvanceTimeType" />
        <xs:element minOccurs="0" maxOccurs="1" name="AdapterConfiguration"
                    type="tns:AnySingleUserElementType" />
      </xs:sequence>
      <xs:attribute name="InputStream" type="xs:anyURI" use="required">
        <xs:annotation>
          <xs:documentation>Reference to an import operator name.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="InputStreamSource" type="xs:anyURI" use="required">
        <xs:annotation>
          <xs:documentation>Reference to an input adapter.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="EventShape" type="tns:EventShapeType" use="required">
        <xs:annotation>
          <xs:documentation>Desired event shape in the input.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="PayloadClassName" type="xs:string" use="optional"/>
    </xs:complexType>
```

Element	Type	Description
AdvanceTime	AdvanceTimeType	The presence of this element indicates that the complex event processing (CEP) engine will inject current time increments (CTIs) in addition to those that come from the adapter code. The content of the element defines the injected events.
AdapterConfiguration	AnySingleUserElementType	The contents of this XML element are not interpreted by the CEP server or by the CEPM protocol. They are passed to an adapter for interpretation by the adapter code.

Attribute	Type	Description
InputStream	**xs:anyURI**	The **InputStream** name. The name of an import operator in the query template.
InputStreamSource	**xs:anyURI**	A reference to the input stream source URI. This can be either another query or an input adapter.
EventShape	EventShapeType	The shape of events that an input stream contains.
PayloadClassName	**xs:string**	A **dotNet** class name of the payload type for a typed adapter.

2.2.3.2.1.1.2.1 AdvanceTimeType

The **AdvanceTimeType** type is used to define current time increments (CTIs) that are injected into the input stream that comes from the adapter. CTIs can be generated based on generation settings, imported from another stream, or both.

The following code is the XML schema (XSD) for the **AdvanceTimeType** type.

```
<xs:complexType name="AdvanceTimeType">
  <xs:annotation>
    <xs:documentation>Specifies how to add CTIs as part of the binding. Can be
    either generated or imported from another stream or both.</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="Generate"
                type="tns:AdvanceTimeGenerateType" />
    <xs:element minOccurs="0" maxOccurs="1" name="Import"
                type="tns:AdvanceTimeImportType" />
  </xs:sequence>
  <xs:attribute name="Policy" type="tns:AdvanceTimePolicyType"
                use="required">
    <xs:annotation>
      <xs:documentation>Specifies how to treat incoming events that violate
      advance time CTIs.</xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:complexType>
```

The following tables describe the elements and attributes for the **AdvanceTimeType** type.

Element	Type	Description
Generate	AdvanceTimeGenerateType	Specifies how to generate CTIs to advance time.
Import	AdvanceTimeImportType	Specifies the source of imported CTIs to advance time.

Attribute	Type	Description
Policy	**base=xs:string**	Specifies the policy to be applied if events from the input adapter violate CTI semantics by having a timestamp earlier than the most recent CTI. The **AdvanceTimePolicyType** enumeration has the following values: ▪ **Adjust**–The timestamp on the event is adjusted to be equal to the most recent CTI. ▪ **Drop**–The violating event is dropped.

2.2.3.2.1.1.2.1.1 AdvanceTimeGenerateType

The **AdvanceTimeGenerateType** type is used to specify how to generate current time increments (CTIs) that are injected into the input stream that comes from the adapter to advance time. The generation is based on a frequency and a delay, which specifies the timestamp of the generated CTI with respect to the most recently seen event in the stream.

The following code is the XML schema (XSD) for the **AdvanceTimeGenerateType** type.

```
  <xs:complexType name="AdvanceTimeGenerateType">
    <xs:annotation>
      <xs:documentation>Specifies how to generate CTIs in order to advance time.
      The generation definition has two dimensions, as one child element each:
      (i) the frequency of advancing application time and (ii) the delay of the
      application time increments. The frequency can be given as a time period
      or as an event count. The delay has to be given as a time period.
    </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:choice>
        <xs:element name="EventCountFrequency"
```

```
                type="tns:AdvanceTimeEventCountFrequencyType" />
          <xs:element name="DurationFrequency"
                type="tns:AdvanceTimeDurationFrequencyType" />
        </xs:choice>
        <xs:element name="Delay" type="tns:AdvanceTimeDelayType" />
        <xs:element name="AdvanceToInfinityOnShutdown"
                type="tns:AdvanceToInfinityType"/>
      </xs:sequence>
    </xs:complexType>
```

The following table describes the elements for the **AdvanceTimeGenerateType** type.

Element	Type	Description
EventCountFrequency	AdvanceTimeEventCountFrequencyType	Specifies a frequency in terms of event count for injected events.
DurationFrequency	AdvanceTimeDurationFrequencyType	Specifies a frequency in terms of a temporal period for injected events.
Delay	AdvanceTimeDelayType	Specifies a delay between the current application time and the timestamp on injected events.
AdvanceToInfinityOnShutdown	AdvanceToInfinityType	If set to true, a CTI with timestamp +infinity will be injected at query shutdown to flush the entire query state.

AdvanceTimeEventCountFrequencyType

The **AdvanceTimeEventCountFrequencyType** type is used to specify a frequency for CTIs that are injected into the input stream. The frequency is specified in terms of a count of events.

The following code is the XSD for the **AdvanceTimeEventCountFrequencyType** type.

```
<xs:complexType name="AdvanceTimeEventCountFrequencyType">
  <xs:annotation>
    <xs:documentation>Specifies the frequency at which to advance application
    time in terms of event count.</xs:documentation>
  </xs:annotation>
  <xs:attribute name="Value" type="xs:unsignedInt" use="required" />
</xs:complexType>
```

The following table describes the attributes for the **AdvanceTimeEventCountFrequencyType** type.

Attribute	Type	Description
Value	**xs:unsignedInt**	Specifies the frequency count at which CTIs are injected in terms of a count of the number of events coming from the input adapter. Setting the **Value** attribute to N means that a CTI is inserted for every N events.

AdvanceTimeDurationFrequencyType

The **AdvanceTimeDurationFrequencyType** type is used to specify the duration between subsequent CTIs that are injected into the input stream. The frequency is specified in terms of a number of time units.

The following code is the XSD for the **AdvanceTimeDurationFrequencyType** type.

```
<xs:complexType name="AdvanceTimeDurationFrequencyType">
  <xs:annotation>
    <xs:documentation>Specifies the frequency at which to advance application
    time in terms of time duration.</xs:documentation>
  </xs:annotation>
  <xs:attribute name="Value" type="xs:duration" use="required" />
</xs:complexType>
```

The following table describes the attributes for the **AdvanceTimeDurationFrequencyType** type.

Attribute	Type	Description
Value	**xs:duration**	Specifies the frequency of CTIs to be injected in terms of a period timespan. Setting the **Value** attribute to T means that a CTI is inserted every T time units.

AdvanceTimeDelayType

The **AdvanceTimeDelayType** type is used to specify a delay between the current application time and the timestamp on injected CTIs.

The following code is the XSD for the **AdvanceTimeDelayType** type.

```
<xs:complexType name="AdvanceTimeDelayType">
  <xs:annotation>
    <xs:documentation>Specifies delay in terms of time duration. The application
    time is advanced to the start time of the most recent event minus the duration.
  </xs:documentation>
  </xs:annotation>
  <xs:attribute name="Value" type="xs:duration" use="required" />
</xs:complexType>
```

The following table describes the attributes for the **AdvanceTimeDelayType** type.

Attribute	Type	Description
Value	**xs:duration**	Specifies the time offset of injected CTIs. Zero indicates no offset from the most recent event's start time. A timespan greater than zero indicates that the CTIs will be timestamped with the corresponding delay, counted towards the past from the start time of the most recent event.

AdvanceToInfinityType

The **AdvanceToInfinityType** type is used to indicate whether an additional CTI with timestamp infinity ~~should be~~is generated at query shutdown.

```
<xs:complexType name="AdvanceToInfinityType">
  <xs:annotation>
    <xs:documentation>Specifies whether an additional CTI with timestamp
    infinity should be generated at query shutdown.</xs:documentation>
  </xs:annotation>
  <xs:attribute name="Value" type="xs:boolean" use="required" />
</xs:complexType>
```

The following table describes the attributes for the **AdvanceToInfinityType** type.

Attribute	Type	Description
Value	**xs:boolean**	True indicates that an additional CTI with timestamp infinity ~~should be~~is generated at query shutdown. Otherwise, false.

2.2.3.2.1.1.2.1.2 AdvanceTimeImportType

The **AdvanceTimeImportType** type is used to import current time increments (CTIs) from another stream to advance time.

The following code is the XML schema (XSD) for the **AdvanceTimeImportType** type.

```
<xs:complexType name="AdvanceTimeImportType">
  <xs:annotation>
    <xs:documentation>Specifies where to import CTIs from in order to
    advance time.</xs:documentation>
  </xs:annotation>
  <xs:attribute name="StreamName" type="xs:string" use="required" />
</xs:complexType>
```

The following table describes the attributes for the **AdvanceTimeImportType** type.

Attribute	Type	Description
StreamName	**xs:string**	Specifies the name of the stream from which CTIs are imported to advance time.

2.2.3.2.1.2 QueryTemplateType

The **QueryTemplateType** type defines how to compute one or more output streams from one or more input streams.

The following code is the XML schema (XSD) for the **QueryTemplateType** type.

```
    <xs:complexType name="QueryTemplateType">
        <xs:annotation>
            <xs:documentation>A Query template has m import and n export operators.
            </xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="unbounded" name="Import"
                        type="tns:ImportOperatorType" />
            <xs:element minOccurs="1" maxOccurs="unbounded" name="Export"
                        type="tns:ExportOperatorType" />
            <xs:group minOccurs="0" maxOccurs="unbounded" ref="tns:AnyOperator" />
        </xs:sequence>
        <xs:attribute name="Name" type="xs:anyURI" />
  <xs:attribute name="Description" type="xs:string" use="optional" />
    </xs:complexType>
```

The following tables describe the elements and attributes for the **QueryTemplateType** type.

Element	Type	Description
Import	ImportOperatorType	Defines a stream entry point of the query template.
Export	ExportOperatorType	Defines a stream exit point of the query template.
(group)	AnyOperator	A set of operators that defines the query graph and thus the query operation.

Attribute	Type	Description
Name	**xs:anyURI**	The name given to this **QueryTemplate**, by which it will be referenced.
Description	**xs:string**	A human-readable description that is not processed by the complex event processing (CEP) server.

2.2.3.2.1.2.1 ImportOperatorType

The following is the XML schema (XSD) for the **ImportOperatorType** type.

```
<xs:complexType name="ImportOperatorType">
  <xs:annotation>
    <xs:documentation>Import Operator. Denotes the query's import stream.
    The Name attribute identifies the stream. Refers to a single operator
    as its output. The attribute Type refers to the stream type using the
    type's name.</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="tns:TerminatorBaseType">
      <xs:sequence>
        <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
                    type="tns:StreamDefinitionType" />
      </xs:sequence>
      <xs:attribute name="Name" type="xs:anyURI" use="required" />
      <xs:attribute name="Type" type="xs:anyURI" use="required" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The following tables describe the elements and attributes for the **ImportOperatorType** type.

Element	Type	Description
OutputStream	StreamDefinitionType	Defines a stream entry point to be used by one or more operators in the specified query template.

Attribute	Type	Description
Name	**xs:anyURI**	The name by which this import operator will be referenced.
Type	**xs:anyURI**	The URI of the EventType object of the events that will be available on the stream.

2.2.3.2.1.2.2 ExportOperatorType

The following is the XML schema (XSD) for the **ExportOperatorType** type.

```
<xs:complexType name="ExportOperatorType">
    <xs:annotation>
        <xs:documentation>Export Operator. Makes the query's outgoing
        stream explicit. The Name attribute identifies the stream. Refers
        to a single operator as its input.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="tns:TerminatorBaseType">
            <xs:sequence>
                <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
                            type="tns:StreamReferenceType" />
            </xs:sequence>
            <xs:attribute name="Name" type="xs:anyURI" use="required" />
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The following tables describe the elements and attributes for the **ExportOperatorType** type.

Element	Type	Description
InputStream	StreamReferenceType	References a stream exit point that is defined by some other operator in the specified query template.

Attribute	Type	Description
Name	**xs:anyURI**	The name by which this export operator is referenced.

2.2.3.2.1.3 ApplicationType

The **ApplicationType** type defines an application object. The **Application** object is the top-level container of the system. A defined **Application** object acts as a namespace for other metadata entities that belong together.

The following code is the XML schema (XSD) for the **ApplicationType** type.

```
<xs:complexType name="ApplicationType">
    <xs:annotation>
        <xs:documentation>Application object.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="Name" type="xs:anyURI" use="required" />
</xs:complexType>
```

The following table describes the attributes for the **ApplicationType** type.

Attribute	Type	Description
Name	**xs:anyURI**	A URI that represents the application name. The application will be referenced by this name.

2.2.3.2.1.4 Adapter Types

Adapters are binary files compiled from user-written code, and they represent an input or output stream source. They convert proprietary event data into complex event processing (CEP) event format for input, or convert CEP event format into a proprietary format for output.

2.2.3.2.1.4.1 AdapterBaseType

In the CEPM protocol, the complex types InputAdapterType and OutputAdapterType are defined as extensions to the **AdapterBaseType** type.

The following code is the XML schema (XSD) for the **AdapterBaseType** type.

```
<xs:complexType name="AdapterBaseType">
  <xs:annotation>
    <xs:documentation>Adapter base type. The common attributes of input
    and output adapter.</xs:documentation>
  </xs:annotation>
  <xs:attribute name="Name" type="xs:anyURI" use="required" />
  <xs:attribute name="FactoryClassName" type="xs:string" use="required" />
  <xs:attribute name="IsTyped" type="xs:boolean"/>
  <xs:attribute name="Description" type="xs:string" use="optional" />
</xs:complexType>
```

The following table describes the attributes for the **AdapterBaseType** type.

Attribute	Type	Description
Name	**xs:anyURI**	The URI given to the adapter. The adapter will be referenced by this name.
FactoryClassName	**xs:string**	This string represents an assembly qualified name of a .NET Framework class implementing the adapter or the GUID that represents an adapter factory class in unmanaged code. For more information about how an assembly-qualified name is constructed, see [MSDN-TAQNP]. This assembly contains the code that represents the adapter at run time.
IsTyped	**xs:boolean**	True if the adapter was developed against a specific type; otherwise, false.
Description	**xs:string**	Adapter description.

2.2.3.2.1.4.2 InputAdapterType

Input adapters are binary files compiled from user-written code, and they represent an input stream source. They convert proprietary event data into complex event processing (CEP) event format. The **InputAdapterType** type is a reference to that user-written code.

The **InputAdapterType** type is an extension of the AdapterBaseType type. It adds no elements or attributes. The following code is the XML schema (XSD) for the **InputAdapterType** type.

```
<xs:complexType name="InputAdapterType">
    <xs:annotation>
        <xs:documentation>Input adapter.</xs:documentation>
    </xs:annotation>
```

```
        <xs:complexContent>
            <xs:extension base="tns:AdapterBaseType">
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

2.2.3.2.1.4.3 OutputAdapterType

Output adapters are binary files compiled from user-written code. They receive the events that are produced by the complex event processing (CEP) engine. The **OutputAdapterType** type references that user-written code. The **OutputAdapterType** type is an extension of the AdapterBaseType type. It adds no elements or attributes.

The following code is the XML schema (XSD) for the **OutputAdapterType** type.

```
    <xs:complexType name="OutputAdapterType">
        <xs:annotation>
            <xs:documentation>Output adapter.</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:extension base="tns:AdapterBaseType" />
        </xs:complexContent>
    </xs:complexType>
```

2.2.3.2.1.5 EventType

Objects of type **EventType** represent the transient data items in a complex event processing system. The **EventType** type is used to define the structure of an event, consisting of one or more fields.

The following is the XML schema (XSD) for the **EventType** type.

```
  <xs:complexType name="EventType">
    <xs:annotation>
      <xs:documentation>Specification of a CEP type.
      Contains zero or more fields.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="Field"
                  type="xs:anyURI" />
    </xs:sequence>
    <xs:attribute name="Name" type="xs:anyURI" use="required" />
  </xs:complexType>
```

The following tables describe the elements and attributes for the **EventType** type.

Element	Type	Description
Field	EventFieldType	A collection of objects of type **EventFieldType** that form the fields of the specified event.

Attribute	Type	Description
Name	**xs:anyURI**	The URI for the event name by which the specified event will be referenced.

2.2.3.2.1.5.1 EventFieldType

A field contains data values of a defined type. The values in each field are processed by the various operators in a query.

The following is the XML schema (XSD) for the **EventFieldType** type.

```
      <xs:complexType name="EventFieldType">
        <xs:annotation>
          <xs:documentation>Field of an Event Type. Can be of atomic or composite
    type.</xs:documentation>
        </xs:annotation>
        <xs:attribute name="Name" type="xs:anyURI" use="required" />
        <xs:attribute name="Type" type="tns:PrimitiveTypeIdentifier" use="required" />
        <xs:attributeGroup ref="tns:TypeFacetAttributes" />
      </xs:complexType>
```

The following table describes the attributes for the **EventFieldType** type.

Attribute	Type	Description
Name	**xs:anyURI**	The URI by which the specified field will be referenced.
Type	**PrimitiveTypeIdentifier**	An enumeration value that represents the type of the field.<3>
(group)	**TypeFacetAttributes**	Contains additional type- and domain-related information about a field beyond the type name.

2.2.3.2.1.6 EntityType

Objects of type **EntityType** represent persisted metadata objects in a complex event processing (CEP) system. The **EntityType** type is used to define the structure of an entity.

The following is the XML schema (XSD) for the **EntityType** type.

```
      <xs:complexType name="EntityType">
        <xs:annotation>
          <xs:documentation>The schema of a CreateEntity command. It contains the schema
    definition for an entity.</xs:documentation>
        </xs:annotation>
        <xs:sequence>
          <xs:element minOccurs="1" maxOccurs="1" ref="linqs:ExpressionRoot" />
        </xs:sequence>
        <xs:attribute name="Name" type="xs:anyURI" use="required" />
      </xs:complexType>
```

The following tables describe the elements and attributes for the **EntityType** type.

Element	Type	Description
ExpressionRoot	**ExpressionRoot**	An expression that defines the entity. For more details, see [MS-LETSF].

Attribute	Type	Description
Name	**xs:anyURI**	The URI by which the specified event is referenced.

2.2.3.2.2 AnyOperator Group

The **AnyOperator** group contains the top-level operator types that ~~may~~can be contained in a QueryTemplate object.

The following code is the XML schema (XSD) for the **AnyOperator** group.

```
<xs:group name="AnyOperator">
  <xs:annotation>
    <xs:documentation>Placeholder for exactly one operator element of any type.
    </xs:documentation>
  </xs:annotation>
  <xs:choice>
    <xs:element name="QueryTemplateReference"
                type="tns:QueryTemplateReferenceOperatorType" />
    <xs:element name="Multicast" type="tns:MulticastOperatorType" />
    <xs:element name="Project" type="tns:ProjectOperatorType" />
    <xs:element name="Select" type="tns:SelectOperatorType" />
    <xs:element name="Join" type="tns:JoinOperatorType" />
    <xs:element name="Union" type="tns:UnionOperatorType" />
    <xs:element name="Aggregate" type="tns:AggregationOperatorType" />
    <xs:element name="AlterLifetime" type="tns:AlterLifetimeOperatorType" />
    <xs:element name="GroupAndApply" type="tns:GroupAndApplyOperatorType" />
    <xs:element name="TopK" type="tns:TopKOperatorType" />
    <xs:element name="UserDefined" type="tns:UserDefinedOperatorType" />
    <xs:element name="UserDefinedStream" type="tns:UserDefinedStreamOperatorType" />
  </xs:choice>
</xs:group>
```

The following table describes the elements for the **AnyOperator** group.

Element	Type	Description
QueryTemplateReference	QueryTemplateReferenceOperatorType	This operator is used to embed another **QueryTemplate** object within a current **QueryTemplate** object.
Multicast	MultiCastOperatorType	The **Multicast** operator broadcasts an input stream to multiple output streams.
Project	ProjectOperatorType	The **Project** operator applies an arbitrary number of expressions to an input stream and produces a single output stream.
Select	SelectOperatorType	The **Select** operator is used to filter inputs and to select a subset of the inputs for output.
Join	JoinOperatorType	The **Join** operator is used to join two inputs based on an expression.
Union	UnionOperatorType	The **Union** operator provides the definition for combining multiple input streams and placing them on a single

Element	Type	Description
		output stream.
Aggregate	AggregationOperatorType	The **Aggregate** operator defines an operation that represents the arithmetic aggregation of inputs to produce an output stream.
AlterLifetime	AlterLifeTimeOperatorType	The **AlterLifetime** operator is used to define time windows for events, such that subsequent operations can be performed on the events within the defined windows.
GroupAndApply	GroupAndApplyOperatorType	The **GroupAndApply** operator is used to partition a stream into subsets and to perform operations on each subset.
TopK	TopKOperatorType	The **TopK** operator specifies that observed values are ranked and only the top K of them are placed on an output stream.
UserDefined	UserDefinedOperatorType	The **UserDefined** operator is a custom operator over event windows that is defined by the user.
UserDefinedStream	UserDefinedStreamOperatorType	The **UserDefinedStream** operator is a custom operator over event streams that is defined by the user.

2.2.3.2.2.1 QueryTemplateReferenceOperatorType

The **QueryTemplateReferenceOperatorType** type is used to embed another query template within a specified query template.

The following code is the XML schema (XSD) for the **QueryTemplateReferenceOperatorType** type.

```
<xs:complexType name="QueryTemplateReferenceOperatorType">
    <xs:annotation>
        <xs:documentation>Embeds another query template in the query.
        </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="tns:OperatorBaseType">
            <xs:sequence>
                <xs:element minOccurs="1" maxOccurs="unbounded"
                     name="InputStream" type="tns:QTrefInputStreamType" />
                <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
                            type="tns:QTrefOutputStreamType" />
            </xs:sequence>
            <xs:attribute name="QueryTemplateName" type="xs:anyURI"
                          use="required" />
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The following tables describe the elements and attributes for the **QueryTemplateReferenceOperatorType** type.

Element	Type	Description
InputStream	QTrefInputStreamType	The input stream for this query operator. This element refers to an import operator in the specified query template.
OutputStream	QTrefOutputStreamType	The output stream for this query operator. This element refers to the export operator in the specified query template.

Attribute	Type	Description
QueryTemplateName	**xs:anyURI**	This URI references an already existing query template.

2.2.3.2.2.1.1 QTrefInputStreamType

The **QTrefInputStreamType** type is used to reference an input operator in another QueryTemplate object.

This type is an extension to the StreamReferenceType type. Thus, it refers to a stream defined somewhere else in a specified query template and feeds it into the specified input operator in another query template. The following code is the XSD for the **QTrefInputStreamType** type.

The following code is the XML schema (XSD) for the **QTrefInputStreamType** type.

```
<xs:complexType name="QTrefInputStreamType">
    <xs:annotation>
        <xs:documentation>Type for the input stream in an QT reference operator.
        In addition to the local stream name, it also needs to refer to the
        respective endpoint in the other query template. This is done via
        the attribute "ExternalName". It refers to the stream name that is used
        in the Import in the embedded query template.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="tns:StreamReferenceType">
            <xs:attribute name="ExternalName" type="xs:anyURI" use="required" />
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The following table describes the attributes for the **QTrefInputStreamType** type.

Attribute	Type	Description
ExternalName	**xs:anyURI**	A reference to the endpoint in the **QueryTemplate** object that is referenced in this **QueryTemplate** object. This is the stream name in the **Import** operator of the referenced **QueryTemplate** object.

2.2.3.2.2.1.2 QTrefOutputStreamType

The **QTrefOutputStreamType** type is used to reference the embedded query ~~template's~~template’s output stream.

This type is an extension to the StreamDefinitionType. Thus, it receives the outgoing stream from another query template and makes it available in this query template.

The following code is the XML schema (XSD) for the **QTrefOutputStreamType** type.

```
<xs:complexType name="QTrefOutputStreamType">
    <xs:annotation>
        <xs:documentation>Type for the output stream in an QT reference
        operator. In addition to the local stream name, it also needs to refer
        to the respective endpoint in the other query template. This is done
        via the attribute "ExternalName". It refers to the stream name that is
        used in a Export in the embedded query template.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="tns:StreamDefinitionType">
            <xs:attribute name="ExternalName" type="xs:anyURI" use="required" />
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The following table describes the attributes for the **QTrefOutputStreamType** type.

Attribute	Type	Description
ExternalName	**xs:anyURI**	Reference to the endpoint in the QueryTemplate object that is referenced in this **QueryTemplate** object. This is the stream name in the export operator of the referenced **QueryTemplate** object.

2.2.3.2.2.1.3 Example

```
<QueryTemplateReference Name="QTReference1"
              QueryTemplateName="cep:/Server/Application/app1/QueryTemplate/Inner">
  <InputStream Name="import1" ExternalName="InputStreamSource1"/>
  <InputStream Name="import2" ExternalName="InputStreamSource2"/>
  <OutputStream Name="qtref1" ExternalName="OutputStreamSource1"/>
</QueryTemplateReference>
```

2.2.3.2.2.2 MultiCastOperatorType

The **MultiCastOperatorType** type defines an operator that replicates a single input stream to multiple output streams.

The following code is the XML schema (XSD) for the **MultiCastOperatorType** type.

```
<xs:complexType name="MultiCastOperatorType">
    <xs:annotation>
        <xs:documentation>A multicast creates multiple named streams out of
        a single input stream. The input events are simply replicated to all
        outputs.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="tns:OperatorBaseType">
            <xs:sequence>
                <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
                            type="tns:StreamReferenceType" />
                <xs:element minOccurs="2" maxOccurs="unbounded"
                        name="OutputStream" type="tns:StreamDefinitionType" />
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The following table describes the elements for the **MultiCastOperatorType** type.

Element	Type	Description
InputStream	StreamReferenceType	A reference to the input stream for this **MultiCast** element.
OutputStream	StreamDefinitionType	The definitions for the multiple output streams.

2.2.3.2.2.2.1 Example

```
<MultiCast Name="MulticastOperator">
  <InputStream Name="import1"></InputStream>
  <OutputStream Name="Multicast1"></OutputStream>
  <OutputStream Name="Multicast2"></OutputStream>
</MultiCast>
```

2.2.3.2.2.3 ProjectOperatorType

The **ProjectOperatorType** type is used as a container for defining an arbitrary number of project expressions on fields of an input stream to produce a single output.

The following code is the XML schema (XSD) for the **ProjectOperatorType** type.

```
    <xs:complexType name="ProjectOperatorType">
        <xs:annotation>
            <xs:documentation>A project operator applies an arbitrary number of
            project expressions to a single input stream and yields a single output
            stream.</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:extension base="tns:OperatorBaseType">
                <xs:sequence>
                    <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
                                type="tns:StreamReferenceType" />
                    <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
                                type="tns:StreamDefinitionType" />
                    <xs:element minOccurs="0" maxOccurs="unbounded"
                                name="ProjectExpression"
                                type="tns:ProjectExpressionContainerType" />
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

The following table describes the elements for the **ProjectOperatorType** type.

Element	Type	Description
InputStream	StreamReferenceType	The input stream for this **Project** element.
OutputStream	StreamDefinitionType	The output stream defined by this **Project** element.

Element	Type	Description
ProjectExpression	ProjectExpressionContainerType	An arbitrary number of project expressions ~~may~~can be specified. Each expression can contain multiple operations. Each project expression evaluates the value of one field in the resulting output event.

2.2.3.2.2.3.1 ProjectExpressionContainerType

The **ProjectExpressionContainerType** type contains a single expression that is used by the ProjectOperatorType type.

The following code is the XML schema (XSD) for the **ProjectExpressionContainerType** type.

```
<xs:complexType name="ProjectExpressionContainerType">
    <xs:annotation>
        <xs:documentation>A project expression contains a single expression
        that determines the value of a new event field. It extends the base
        container type by adding an attribute to assign a name to that new
        field. This is also a base class for other operators' expressions that
        result in new event fields.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="tns:ExpressionContainerType">
            <xs:attribute name="OutputField" type="xs:anyURI" use="required" />
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The following table describes the attributes for the **ProjectExpressionContainerType** type.

Attribute	Type	Description
OutputField	**xs:anyURI**	The URI to be assigned to the new field in the event payload that contains the result of the application of the project expressions specified in the containing **Project** element.

2.2.3.2.2.3.2 Example

```
<Project Name="project2">
  <InputStream Name="project1"></InputStream>
  <OutputStream Name="project2"></OutputStream>
  <ProjectExpression OutputField="OutputField21">
    <MethodCall Nullable="0" Method="Substring" Class="System.String"
                MaxSize="10" SizeFixed="true">
      <Constant Nullable="0" Type="System.String"
                Value="11123456789000"></Constant>
      <Constant Nullable="0" Type="System.Int32" Value="2"></Constant>
      <Constant Nullable="0" Type="System.Int32" Value="10"></Constant>
    </MethodCall>
  </ProjectExpression>
  <ProjectExpression OutputField="OutputField22">
    <Condition>
      <Constant Nullable="0" Type="System.Boolean" Value="true"></Constant>
      <InputField Name="OutputField2"></InputField>
      <InputField Name="OutputField2"></InputField>
    </Condition>
  </ProjectExpression>
</Project>
```

2.2.3.2.2.4 SelectOperatorType

The **SelectOperatorType** type is used to filter inputs and to select a subset of the inputs for output.

The following code is the XML schema (XSD) for the **SelectOperatorType** type.

```
    <xs:complexType name="SelectOperatorType">
        <xs:annotation>
            <xs:documentation>A select expression contains exactly one filter
            expression.</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:extension base="tns:OperatorBaseType">
                <xs:sequence>
                    <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
                          type="tns:StreamReferenceType" />
                    <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
                          type="tns:StreamDefinitionType" />
                    <xs:element minOccurs="1" maxOccurs="1" name="FilterExpression"
                          type="tns:ExpressionContainerType" />
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

The following table describes the elements for the **SelectOperatorType** type.

Element	Type	Description
InputStream	StreamReferenceType	The input stream for this **Select** element.
OutputStream	StreamDefinitionType	The output stream defined by this **Select** element.
FilterExpression	ExpressionContainerType	A filter expression. This expression defines which of the inputs will be selected to be placed on the output stream. Only events that fulfill the filter expression will be output by the **Select** operator.

2.2.3.2.2.4.1 Example

```
<Select Name="SelectOperator1">
  <InputStream Name="import1"></InputStream>
  <OutputStream Name="select1"></OutputStream>
<FilterExpression>
    <Equal>
      <Modulo>
        <InputField Name="Field1"></InputField>
        <Constant Nullable="0" Value="3" Type="System.Int32"></Constant>
      </Modulo>
      <Constant Nullable="0" Value="0" Type="System.Int32"></Constant>
    </Equal>
  </FilterExpression>
</Select>
```

2.2.3.2.2.5 JoinOperatorType

The **JoinOperatorType** type is used to join two inputs based on an expression.

The following code is the XML schema (XSD) for the **JoinOperatorType** type.

```
        <xs:complexType name="JoinOperatorType">
            <xs:annotation>
                <xs:documentation>A Join element has two inputs and one output. The
                join predicate is specified as a child element. The join can include
                zero or more ProjectExpressions
    .
              </xs:documentation>
            </xs:annotation>
            <xs:complexContent>
                <xs:extension base="tns:OperatorBaseType">
                    <xs:sequence>
                        <xs:element minOccurs="2" maxOccurs="2" name="InputStream"
                                    type="tns:StreamReferenceType" />
                        <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
                                    type="tns:StreamDefinitionType" />
                        <xs:element minOccurs="1" maxOccurs="1" name="JoinPredicate"
                                    type="tns:ExpressionContainerType" />
                        <xs:element minOccurs="0" maxOccurs="unbounded"
                                    name="ProjectExpression"
                                    type="tns:ProjectExpressionContainerType" />
                    </xs:sequence>
                    <xs:attribute name="JoinType">
                        <xs:simpleType>
                            <xs:restriction base="xs:string">
                                <xs:enumeration value="LeftOuter" />
                                <xs:enumeration value="RightOuter" />
                                <xs:enumeration value="FullOuter" />
                                <xs:enumeration value="LeftAnti" />
                                <xs:enumeration value="RightAnti" />
                                <xs:enumeration value="LeftSemi" />
                                <xs:enumeration value="RightSemi" />
                                <xs:enumeration value="LeftAntiSemi" />
                                <xs:enumeration value="RightAntiSemi" />
                                <xs:enumeration value="Inner" />
                            </xs:restriction>
                        </xs:simpleType>
                    </xs:attribute>
                    <xs:attribute name="PointEvents" type="xs:boolean" use="optional"
                                  default="false" />
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
```

The following tables describe the elements and attributes for the **JoinOperatorType** type.

Element	Type	Description
InputStream	StreamReferenceType	The input stream for the specified **Join** element. A join has exactly two input streams.
OutputStream	StreamDefinitionType	The output stream defined by the specified **Join** element.
JoinPredicate	ExpressionContainerType	The element that contains the expression on which to base the join.

Element	Type	Description
ProjectExpression	ProjectExpressionContainerType	The element with which the join operator can optionally define project expressions on the join result. Each project expression computes the value for a single field in the output event from the values of the two input events. The result event of the join operator contains exactly the set of all project expressions.

Attribute	Type	Description
JoinType	**xs:string**(restriction)	This enumeration indicates the type of join that will be performed. ▪ "LeftOuter" ▪ "RightOuter" ▪ "FullOuter" ▪ "LeftAnti" ▪ "RightAnti" ▪ "LeftSemi" ▪ "RightSemi" ▪ "LeftAntiSemi" ▪ "RightAntiSemi" ▪ "Inner"
PointEvents	**xs:boolean**	This attribute can be set to true if both input streams of a join operator contain only point events. Setting it to true will result in a different **Join** implementation being used by the engine but will not affect the correctness of the results.

2.2.3.2.2.5.1 Example

```
<Join Name="join1" JoinType="Inner" PointEvents="true">
  <InputStream Name="alterlifetime1" />
  <InputStream Name="alterlifetime2" />
  <OutputStream Name="join1" />
  <JoinPredicate>
    <And>
      <Equal>
        <Compare>
          <InputField Name="UserId" StreamName="alterlifetime1" />
          <InputField Name="UserId" StreamName="alterlifetime2" />
        </Compare>
        <Constant Nullable="0" Type="System.Int32" Value="0" />
      </Equal>
      <Equal>
        <Compare>
          <InputField Name="SegmentHitLogicId" StreamName="alterlifetime1" />
          <InputField Name="SegmentHitLogicId" StreamName="alterlifetime2" />
        </Compare>
        <Constant Nullable="0" Type="System.Int32" Value="0" />
      </Equal>
    </And>
  </JoinPredicate>
  <ProjectExpression OutputField="UserId">
    <InputField Name="UserId" StreamName="alterlifetime2" />
  </ProjectExpression>
  <ProjectExpression OutputField="SegmentHitLogicId">
    <InputField Name="SegmentHitLogicId" StreamName="alterlifetime2" />
  </ProjectExpression>
  <ProjectExpression OutputField="Count">
    <InputField Name="Count" StreamName="alterlifetime2" />
  </ProjectExpression>
</Join>
```

2.2.3.2.2.6 UnionOperatorType

A **UnionOperatorType** type takes multiple input streams and places them on a single output stream.

The following code is the XML schema (XSD) for the **UnionOperatorType** type.

```
<xs:complexType name="UnionOperatorType">
    <xs:annotation>
        <xs:documentation>A union operator funnels multiple input stream into
        one output stream.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="tns:OperatorBaseType">
            <xs:sequence>
                <xs:element minOccurs="2" maxOccurs="unbounded"
                            name="InputStream"
                            type="tns:StreamReferenceType" />
                <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
                            type="tns:StreamDefinitionType" />
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The following table describes the elements for the **UnionOperatorType** type.

Element	Type	Description
InputStream	StreamReferenceType	The input streams for this **Union** element.
OutputStream	StreamDefinitionType	The output stream defined by this **Union** element.

2.2.3.2.2.6.1 Example

```
<Union Name="UnionOperator">
    <InputStream Name="Multicast1"></InputStream>
    <InputStream Name="Multicast2"></InputStream>
    <OutputStream Name="union1"></OutputStream>
  </Union>
```

2.2.3.2.2.7 AggregationOperatorType

The **AggregationOperatorType** type is used to define an arithmetic aggregation of inputs to produce an output stream.

The following code is the XML schema (XSD) for the **AggregationOperatorType** type.

```
<xs:complexType name="AggregationOperatorType">
  <xs:annotation>
    <xs:documentation>An aggregate element has one or more aggregate
    expressions, each yielding a new column that represents the aggregation
    result.</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="tns:WindowedOperatorBaseType">
      <xs:sequence>
        <xs:group minOccurs="1" maxOccurs="unbounded"
                  ref="tns:AnyAggregate" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The following table describes the element for the **AggregationOperatorType** type.

Element	Type	Description
(group)	AnyAggregate	The output stream defined by this **Aggregate** element.

2.2.3.2.2.7.1 AnyAggregate

The **AnyAggregate** type represents a single aggregation element.

The following code is the XML schema (XSD) for the **AnyAggregate** type.

```
<xs:group name="AnyAggregate">
  <xs:annotation>
    <xs:documentation>Set of all aggregation functions.</xs:documentation>
  </xs:annotation>
  <xs:choice>
    <xs:element name="Sum" type="tns:AggregateSumType" />
    <xs:element name="Count" type="tns:AggregateBaseType" />
    <xs:element name="Min" type="tns:AggregateMinType" />
    <xs:element name="Max" type="tns:AggregateMaxType" />
    <xs:element name="Avg" type="tns:AggregateAvgType" />
    <xs:element name="UserDefined" type="tns:AggregateUserDefinedType" />
  </xs:choice>
</xs:group>
```

Element	Type	Description
Sum	AggregateSumType	Contains information about an aggregation by summing.
Count	AggregateBaseType	Contains information about an aggregation by counting.
Min	AggregateMinType	Contains information about an aggregation by minimum.
Max	AggregateMaxType	Contains information about an aggregation by maximum.

Element	Type	Description
Avg	AggregateAvgType	Contains information about an aggregation by average.
UserDefined	AggregateUserDefinedType	Allows for definition of a user-defined aggregate.

2.2.3.2.2.7.1.1 AggregateBaseType

The **AggregateBaseType** is the base type from which aggregation types are specified by extension.

The following is the XML schema (XSD) for the **AggregateBaseType** type.

```
<xs:complexType name="AggregateBaseType">
  <xs:annotation>
    <xs:documentation>Base type for a single aggregation.
    The result is always assigned to an output field.</xs:documentation>
  </xs:annotation>
  <xs:attribute name="OutputField" type="xs:anyURI" use="required" />
</xs:complexType>
```

Attribute	Type	Description
OutputField	**xs:anyURI**	Specifies the URI for the output field to which the aggregation result is assigned.

2.2.3.2.2.7.1.2 AggregateSumType

The **AggregateSumType** type specifies an aggregation by the summing of an expression against a set of events.

The following is the XML schema (XSD) for the **AggregateSumType** type.

```
<xs:complexType name="AggregateSumType">
  <xs:annotation>
    <xs:documentation>Sum over an expression evaluated on all input events.
    </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="tns:AggregateBaseType">
      <xs:sequence>
        <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Element	Type	Description
(group)	**AnyExpression**	Specifies an expression for the aggregation.

2.2.3.2.2.7.1.3 AggregateMinType

The **AggregateMinType** specifies an aggregation that is the minimum of an expression evaluated against a set of events.

The following is the XML schema (XSD) for the **AggregateMinType** type.

```
<xs:complexType name="AggregateMinType">
  <xs:annotation>
    <xs:documentation>Numeric minimum of expressions evaluated on all input events.
    </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="tns:AggregateBaseType">
      <xs:sequence>
        <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Element	Type	Description
(group)	**AnyExpression**	Specifies an expression for the aggregation.

2.2.3.2.2.7.1.4 AggregateMaxType

The **AggregateMaxType** type specifies an aggregation that is the maximum of an expression evaluated against a set of events.

The following is the XML schema (XSD) for the **AggregateMaxType** type.

```
<xs:complexType name="AggregateMaxType">
  <xs:annotation>
    <xs:documentation>Numeric maximum of expressions evaluated on all input events.
    </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="tns:AggregateBaseType">
      <xs:sequence>
        <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Element	Type	Description
(group)	**AnyExpression**	Specifies an expression for the aggregation.

2.2.3.2.2.7.1.5 AggregateAvgType

The **AggregateAvgType** type specifies an aggregation that is the average of an expression evaluated against a set of events.

The following is the XML schema (XSD) for the **AggregateAvgType** type.

```
<xs:complexType name="AggregateAvgType">
  <xs:annotation>
    <xs:documentation>Numeric average of expressions evaluated on all input events.
    </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="tns:AggregateBaseType">
      <xs:sequence>
        <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Element	Type	Description
(group)	**AnyExpression**	Specifies an expression for the aggregation.

2.2.3.2.2.7.1.6 AggregateUserDefinedType

The **AggregateUserDefinedType** type defines a custom user-defined aggregation over zero or more expressions evaluated against a set of events.

The following is the XML schema (XSD) for the **AggregateUserDefinedType** type.

```
<xs:complexType name="AggregateUserDefinedType">
  <xs:annotation>
    <xs:documentation>A user-defined aggregate operates against a window
    of events and returns a single scalar value.</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="tns:AggregateBaseType">
      <xs:sequence>
        <xs:element minOccurs="1" maxOccurs="1" name="Implementation"
                    type="tns:ImplementationType" />
        <xs:element minOccurs="0" maxOccurs="1" name="Configuration"
                    type="tns:SerializedConfigurationType" />
        <xs:group minOccurs="0" maxOccurs="1" ref="tns:AnyExpression" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Element	Type	Description
Implementation	ImplementationType	Specifies the type of the implementation element. Contains information about the common language runtime (CLR) implementation of the element.
Configuration	SerializedConfigurationType	Specifies the XML for the fully serialized type definition of the implemented type. It is usually produced by the development tool.
(group)	AnyExpression	Expression for the user-defined aggregation.

2.2.3.2.2.8 Example

```
<Aggregate Name="aggregate1">
  <InputStream Name="applyinput2"></InputStream>
  <OutputStream Name="aggregate1"></OutputStream>
  <SnapshotWindow>
    <WindowDefinition></WindowDefinition>
    <InputPolicy>
      <Clip Left="true" Right="true"></Clip>
    </InputPolicy>
    <OutputPolicy>
      <Adjust Alignment="WindowStart" Lifetime="WindowSize"></Adjust>
    </OutputPolicy>
  </SnapshotWindow>
  <Count OutputField="Count"></Count>
</Aggregate>
```

2.2.3.2.2.9 AlterLifetimeOperatorType

The **AlterLifetimeOperatorType** type is used to create a time window of events that are passed to the output. It does that through the alteration of the events' start timestamp and lifetime period.

The following is the XML schema (XSD) for the **AlterLifetimeOperatorType** type.

```
<xs:complexType name="AlterLifetimeOperatorType">
    <xs:annotation>
        <xs:documentation>An AlterLifetime operator defines two expressions:
        One for the new start time and one for the new life time of the event.
        At least one of these must be specified.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="tns:OperatorBaseType">
            <xs:sequence>
                <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
                            type="tns:StreamReferenceType" />
                <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
                            type="tns:StreamDefinitionType" />
                <xs:element minOccurs="0" maxOccurs="1"
                            name="StartTimeExpression"
                            type="tns:ExpressionContainerType" />
                <xs:element minOccurs="0" maxOccurs="1"
                            name="LifetimeExpression"
                            type="tns:ExpressionContainerType" />
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The following table describes the elements for the **AlterLifetimeOperatorType** type.

Element	Type	Description
InputStream	StreamReferenceType	The input stream for this **AlterLifetime** element.
OutputStream	StreamDefinitionType	The output stream defined by this **AlterLifetime** element.
StartTimeExpression	ExpressionContainerType	The expression that is used to compute the new start time of the specified event.

Element	Type	Description
LifetimeExpression	**ExpressionContainerType**	The expression for the new lifetime of the specified event. At least one of **StartTimeExpression** and **LifetimeExpression** MUST be specified.

2.2.3.2.2.9.1 Example

```
<AlterLifetime Name="alt2">
  <InputStream Name="altin" />
  <OutputStream Name="onehour" />
  <StartTimeExpression>
    <ValidStartTime />
  </StartTimeExpression>
  <LifetimeExpression>
    <Constant Type="cep:/Server/Application/system/EventType/System.TimeSpan"
              Value="PT3600S" />
  </LifetimeExpression>
</AlterLifetime>
```

2.2.3.2.2.10 GroupAndApplyOperatorType

The **GroupAndApplyOperatorType** type is used to divide inputs into groups and then apply the same sub-query to each group.

The following code is the XML schema (XSD) for the **GroupAndApplyOperatorType** type.

```
    <xs:complexType name="GroupAndApplyOperatorType">
        <xs:annotation>
            <xs:documentation>
                Implements the Group and Apply operator. One or more grouping
                expressions determine the event partitions. The operator graph in
                the Apply element will be applied to each group separately. The
                grouping expression is of the same type as the project expression:
                it can contain any expression, but it must assign a field name to
                that expression result.
            </xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:extension base="tns:OperatorBaseType">
                <xs:sequence>
                    <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
                                type="tns:StreamReferenceType" />
                    <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
                                type="tns:StreamDefinitionType" />
                    <xs:element minOccurs="1" maxOccurs="unbounded"
                                name="GroupingExpression"
                                type="tns:ProjectExpressionContainerType" />
                    <xs:element minOccurs="1" maxOccurs="1" name="Apply"
                                type="tns:ApplyBranchType">
                        <xs:key name="ApplyStreamKey">
                            <xs:annotation>
                                <xs:documentation>Stream identifier to be used in the
                                operators of that apply element.</xs:documentation>
                            </xs:annotation>
                            <xs:selector xpath="./*/tns:OutputStream" />
                            <xs:field xpath="@Name" />
                        </xs:key>
                        <xs:keyref name="ApplyStreamKeyref"
                                   refer="tns:ApplyStreamKey">
                            <xs:annotation>
                                <xs:documentation>Stream reference for operators.
                                A stream reference has to match a stream identifier
                                in order to connect operators.</xs:documentation>
                            </xs:annotation>
                            <xs:selector xpath="./*/tns:InputStream" />
                            <xs:field xpath="@Name" />
                        </xs:keyref>
                    </xs:element>
                </xs:sequence>
                <xs:attribute name="AddGroupingFields" type="xs:boolean"
                              use="optional" default="false" />
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

The following tables describe the elements and attributes for the **GroupAndApplyOperatorType** type.

Element	Type	Description
InputStream	StreamReferenceType	The input stream for the specified operator.
OutputStream	StreamDefinitionType	The output stream defined by the specified operator.
GroupingExpression	ProjectExpressionContainerType	The expression that defines which events are partitioned into the same group. For each distinct result of the grouping expression, a separate group is created.
Apply	ApplyBranchType	The sub-query that will be applied to each group separately.

Attribute	Type	Description
AddGroupingFields	**xs:boolean**	If this flag is set, the result of the grouping expressions will be added to the ~~events'~~events' payloads in the ~~operator's~~operator's output stream.

2.2.3.2.2.10.1 ApplyBranchType

The **ApplyBranchType** type is used to specify the operations that are to be applied to each branch of the **GroupandApply** operator.

The following code is the XML schema (XSD) for the **ApplyBranchType** type.

```
<xs:complexType name="ApplyBranchType">
    <xs:annotation>
        <xs:documentation>The Apply element encapsulates the apply operator
        graph of the Group and Apply operator. It must have exactly one input
        and one output, which are terminated by elements of type ApplyInputType
        and ApplyOutputType. These elements are named ImportOperator and
        ExportOperator to be able to re-use exisitng query templates as apply
        branches. However, their type here is different from
        query-template-level imports and exports in that they do not require a
        type specification.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <xs:element minOccurs="1" maxOccurs="1" name="ApplyInput"
                    type="tns:ApplyInputType" />
```

```
            <xs:element minOccurs="1" maxOccurs="1" name="ApplyOutput"
                        type="tns:ApplyOutputType" />
            <xs:group minOccurs="0" maxOccurs="unbounded"
                      ref="tns:AnyOperator" />
        </xs:sequence>
    </xs:complexType>
```

The following table describes the elements for the **ApplyBranchType** type.

Element	Type	Description
ApplyInput	ApplyInputType	The input stream definition for the apply query graph.
ApplyOutput	ApplyOutputType	The output stream definition for the apply query graph.
(group)	AnyOperator	The set of operators that define the apply sub-query.

2.2.3.2.2.10.1.1 ApplyInputType

The **ApplyInputType** type is used to define the input operator for an apply branch.

The following code is the XML schema (XSD) for the **ApplyInputType** type.

```
    <xs:complexType name="ApplyInputType">
        <xs:annotation>
            <xs:documentation>Input terminator of the apply operator graph.
            </xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:extension base="tns:TerminatorBaseType">
                <xs:sequence>
                    <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
                                type="tns:StreamDefinitionType" />
                </xs:sequence>
                <xs:attribute name="Name" type="xs:anyURI" use="required" />
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

The following tables describe the elements and attributes for the **ApplyInputType** type.

Element	Type	Description
OutputStream	StreamDefinitionType	The output stream defined by this **ApplyInput** operator. Further operators in the apply branch can now refer to this stream with their input streams.

Attribute	Type	Description
Name	**xs:anyURI**	The assigned URI by which this **ApplyInput** operator will be referenced.

2.2.3.2.2.10.1.2 ApplyOutputType

The **ApplyOutputType** type is used to define the output operator for an apply branch.

The following code is the XML schema (XSD) for the **ApplyOutputType** type.

```
<xs:complexType name="ApplyOutputType">
    <xs:annotation>
        <xs:documentation>Output terminator of the apply operator graph.
        </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="tns:TerminatorBaseType">
            <xs:sequence>
                <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
                            type="tns:StreamReferenceType" />
            </xs:sequence>
            <xs:attribute name="Name" type="xs:anyURI" use="required" />
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The following tables describe the elements and attributes for the **ApplyOutputType** type.

Element	Type	Description
InputStream	StreamReferenceType	The input stream into this **ApplyOutput** operator. Refers to another ~~operator's~~operator’s **OutputStream** object in the apply branch.

Attribute	Type	Description
Name	**xs:anyURI**	The assigned URI by which this **ApplyOutput** operator will be referenced.

2.2.3.2.2.10.2 Example

```
<GroupAndApply Name="GroupAndApply1">
  <InputStream Name="import"></InputStream>
  <OutputStream Name="ga"></OutputStream>
  <GroupingExpression OutputField="GroupExpr1">
    <Modulo>
      <InputField Name="Field1"></InputField>
      <Constant Nullable="0" Value="3" Type="System.Int32"></Constant>
    </Modulo>
  </GroupingExpression>
  <Apply>
    <ApplyInput Name="appin">
      <OutputStream Name="applyin"></OutputStream>
    </ApplyInput>
    <ApplyOutput Name="appout">
      <InputStream Name="select"></InputStream>
    </ApplyOutput>
    <Select Name="SelectOperator">
      <InputStream Name="applyin"></InputStream>
      <OutputStream Name="select"></OutputStream>
      <FilterExpression>
        <Equal>
          <Modulo>
            <InputField Name="Field1"></InputField>
            <Constant Nullable="0" Value="4" Type="System.Int32"></Constant>
          </Modulo>
          <Constant Nullable="0" Value="0" Type="System.Int32"></Constant>
        </Equal>
      </FilterExpression>
    </Select>
  </Apply>
</GroupAndApply>
```

2.2.3.2.2.11 TopKOperatorType

The **TopKOperatorType** type performs a ranking based on observed or computed field values and returns only the top K in number, where K is user-specified in the definition. In the case of a tie, all events with the same rank are output so that the operation is always deterministic.

The following code is the XML schema (XSD) for the **TopKOperator** type.

```
<xs:complexType name="TopKOperatorType">
  <xs:annotation>
    <xs:documentation>TopK operator. The K is specified by the required
    RankDepth attribute. The calculated rank can be projected in the output
    of the operator by specifying a field name through the attribute
    RankOutputField. The rank is calculated according to the value of the rank
    expression, its datatype, and the specified ordering. If more than one rank
    expression is specified, they are evaluated subsequently, i.e., if one rank
    expression evaluates for a tie for any two events, the next expression in
    the sequence is evaluated, etc.</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="tns:WindowedOperatorBaseType">
      <xs:sequence>
        <xs:element minOccurs="1" maxOccurs="unbounded" name="RankExpression"
                    type="tns:RankExpressionContainerType" />
      </xs:sequence>
      <xs:attribute name="RankDepth" type="xs:int" use="required" />
      <xs:attribute name="RankOutputField" type="xs:anyURI" use="optional"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The following tables describe the elements and attributes for the **TopKOperator** type.

Element	Type	Description
RankExpression	RankExpressionContainerType	The expression whose result will be used to determine the rank of the input events.

Attribute	Type	Description
RankDepth	**xs:int**	An integer that specifies the maximum rank of events included in the output.
RankOutputField	**xs:anyURI**	The attribute that specifies the name of the optional output field in which to include the rank in the output ~~events'~~events’ payload.

2.2.3.2.2.11.1 RankExpressionContainerType

The **RankExpressionContainerType** type is used to specify one or more ranking expressions.

The following code is the XML schema (XSD) for the **RankExpressionContainerType** type.

```
<xs:complexType name="RankExpressionContainerType">
    <xs:annotation>
        <xs:documentation>A rank expression contains a single expression that
        is to be used to determine the rank in a TopK operator. It extends the
        base container type by adding an attribute to specify the ordering.
      </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="tns:ExpressionContainerType">
            <xs:attribute name="Order" type="tns:RankOrderType"
                          use="required" />
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The following table describes the attributes for the **RankExpressionContainerType** type.

Attribute	Type	Description
Order	RankOrderType	The attribute that specifies whether values are ordered in ascending or descending order.

2.2.3.2.2.11.1.1 RankOrderType

The **RankOrderType** type is an enumeration containing the values on which the ranking can be ordered.

The following code is the XML schema (XSD) for the **RankOrderType** type.

```
<xs:simpleType name="RankOrderType">
    <xs:annotation>
        <xs:documentation>The ordering of a rank expression can be ascending
        or descending.</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
        <xs:enumeration value="Ascending" />
        <xs:enumeration value="Descending" />
    </xs:restriction>
</xs:simpleType>
```

The enumeration values for the **RankOrderType** type are as follows.

Value	Description
Ascending	Ranked in ascending order.
Descending	Ranked in descending order.

2.2.3.2.2.11.2 Example

```
<TopK Name="TopK1" RankDepth="3" RankOutputField="Field3">
  <InputStream Name="import1"></InputStream>
  <OutputStream Name="TopKOutput1"></OutputStream>
  <SnapshotWindow>
    <WindowDefinition></WindowDefinition>
    <InputPolicy>
      <Clip Left="true" Right="true"></Clip>
    </InputPolicy>
    <OutputPolicy>
      <Adjust Alignment="WindowStart" Lifetime="WindowSize"></Adjust>
    </OutputPolicy>
  </SnapshotWindow>
  <RankExpression Order="Ascending">
    <InputField Name="Field1" StreamName="import1"></InputField>
  </RankExpression>
  <RankExpression Order="Ascending">
    <InputField Name="Field2" StreamName="import1"></InputField>
  </RankExpression>
</TopK>
```

2.2.3.2.2.11.3 UserDefinedOperatorType

The **UserDefinedOperatorType** type allows users to define their own operator over event windows, which can be used in addition to the built-in window-based operators. The **UserDefinedOperatorType** type operates on a set of events (as contained in the specified window) and returns a set of events.

The following is the XML schema (XSD) for the **UserDefinedOperatorType** type.

```
<xs:complexType name="UserDefinedOperatorType">
  <xs:annotation>
    <xs:documentation>A user-defined operator (UDO) is defined on top of a
    window of events and implements a custom function, returning a set of
    events.</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="tns:WindowedOperatorBaseType">
      <xs:sequence>
        <xs:element minOccurs="1" maxOccurs="1" name="Implementation"
                    type="tns:ImplementationType" />
        <xs:element minOccurs="0" maxOccurs="1" name="Configuration"
                    type="tns:SerializedConfigurationType" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Element	Type	Description
Implementation	ImplementationType Type	Specifies the type of the implemented element. Contains information about the CLR implementation of the element.
Configuration	SerializedConfigurationType Type	Specifies the XML for the fully serialized type definition of the implemented type. The XML is usually produced by the development tool.

2.2.3.2.2.11.3.1 Example

```
<UserDefined Name="UDO1">
  <InputStream Name="import1"></InputStream>
  <OutputStream Name="select1"></OutputStream>
  <SnapshotWindow>
    <WindowDefinition></WindowDefinition>
    <InputPolicy>
      <Clip Left="true" Right="true"></Clip>
    </InputPolicy>
    <OutputPolicy>
      <Adjust Alignment="WindowStart" Lifetime="WindowSize"></Adjust>
    </OutputPolicy>
  </SnapshotWindow>
  <Implementation Class="Microsoft.SqlServer.Test.TestShellTests.
                  ComplexEventProcessing.UserDefinedModuleSamples.
                  SampleUDO, Microsoft.SqlServer.Test.TestShellTests.
                  ComplexEventProcessing.UserDefinedModuleSamples,
                  Version=10.0.0.0, Culture=neutral"
                  InputClrType="Microsoft.SqlServer.Test.TestShellTests.
                  ComplexEventProcessing.UserDefinedModuleSamples.Input,
                  Microsoft.SqlServer.Test.TestShellTests.ComplexEventProcessing.
                  UserDefinedModuleSamples, Version=10.0.0.0, Culture=neutral"
                  ReturnClrType="Microsoft.SqlServer.Test.TestShellTests.
                  ComplexEventProcessing.UserDefinedModuleSamples.Output,
                  Microsoft.SqlServer.Test.TestShellTests.ComplexEventProcessing.
                  UserDefinedModuleSamples, Version=10.0.0.0, Culture=neutral">
    </Implementation>
  <Configuration Class="Microsoft.SqlServer.Test.TestShellTests.
                 ComplexEventProcessing.UserDefinedModuleSamples.UdoConfig,
                 Microsoft.SqlServer.Test.TestShellTests.ComplexEventProcessing.
                 UserDefinedModuleSamples, Version=10.0.0.0, Culture=neutral">
    <UdoConfig>
      <UdoConfigParameter>439</UdoConfigParameter>
    </UdoConfig>
  </Configuration>
</UserDefined>
```

2.2.3.2.2.11.4 UserDefinedStreamOperatorType

The **UserDefinedStreamOperatorType** type allows users to define their own operator over event streams, which can be used in addition to the built-in operators. The **UserDefinedStreamOperatorType** type operates on an ordered stream of events and returns one or more payloads for each incoming event. The operator definition includes the serialized initial state.

The following is the XML schema (XSD) for the **UserDefinedStreamOperatorType** type.

```
<xs:complexType name="UserDefinedStreamOperatorType">
  <xs:annotation>
    <xs:documentation>A user-defined stream operator is defined
    on top of a stream of events and implements a custom
    function, return a stream of events.</xs:documentation>
  </xs:annotation>
  <xs:complexContent mixed="false">
    <xs:extension base="tns:OperatorBaseType">
      <xs:sequence>
        <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
                    type="tns:StreamReferenceType" />
        <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
                    type="tns:StreamDefinitionType" />
        <xs:element minOccurs="1" maxOccurs="1" name="InitialState"
                    type="tns:SerializedConfigurationType" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The following table specifies the elements for the **UserDefinedStreamOperatorType** type.

Element	Type	Description
InputStream	StreamReferenceType	Specifies the input stream for this **UserDefinedStream** element.
OutputStream	StreamDefinitionType	Specifies the output stream for this **UserDefinedStream** element.
InitialState	SerializedConfigurationType Type	Specifies the XML for the fully serialized initial state of the user-defined stream operator. The XML is usually produced by the development tool.

2.2.3.2.2.11.4.1 Example

```
<UserDefinedStream Name="UserDefinedStream.3.1">
  <InputStream Name="Import.4.0" />
  <OutputStream Name="UserDefinedStream.3.1" />
  <InitialState Class="UserFunctions.SensorUdso, UserFunctions, Version=0.0.0.0,
                       Culture=neutral, PublicKeyToken=null">
    <SensorUdso xmlns:i=http://www.w3.org/2001/XMLSchema-instance
                xmlns="http://schemas.datacontract.org/2004/07/UserFunctions">
      <_horizon xmlns:d2p1="http://schemas.datacontract.org/2004/07/System">
        <d2p1:DateTime>9999-12-31T23:59:59.9999999Z</d2p1:DateTime>
        <d2p1:OffsetMinutes>0</d2p1:OffsetMinutes>
      </_horizon>
    </SensorUdso>
  </InitialState>
</UserDefinedStream>
```

2.2.3.2.3 Additional Types, Groups, and AttributeGroups

None.

2.2.3.2.3.1 BuiltinType

The **BuiltinType** type contains an enumeration of data types that are used for other elements within the system.

The following code is the XML schema (XSD) for the **BuiltinType** type.

```
    <xs:simpleType name="BuiltinType">
        <xs:annotation>
            <xs:documentation>List of all natively supported types, as relative URI.
            </xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:string">
            <xs:enumeration value="System.Boolean" />
            <xs:enumeration value="System.Char" />
            <xs:enumeration value="System.SByte" />
            <xs:enumeration value="System.Int16" />
            <xs:enumeration value="System.Int32" />
            <xs:enumeration value="System.Int64" />
            <xs:enumeration value="System.Byte" />
            <xs:enumeration value="System.UInt16" />
            <xs:enumeration value="System.UInt32" />
            <xs:enumeration value="System.UInt64" />
            <xs:enumeration value="System.Decimal" />
            <xs:enumeration value="System.Single" />
            <xs:enumeration value="System.Double" />
            <xs:enumeration value="System.Guid" />
            <xs:enumeration value="System.DateTime" />
            <xs:enumeration value="System.TimeSpan" />
            <xs:enumeration value="System.String" />
            <xs:enumeration value="System.Byte[]" />
        </xs:restriction>
    </xs:simpleType>
```

The values of the enumeration represent the types that are used in **Field** definitions of type EventFieldType.

2.2.3.2.3.2 OperatorBaseType

The **OperatorBaseType** type is a base type on which other operators are defined with extension or restriction.

The following code is the XML schema (XSD) for the **OperatorBaseType** type.

```
    <xs:complexType name="OperatorBaseType">
        <xs:annotation>
            <xs:documentation>Operator base type. Every operator has a name.
            </xs:documentation>
        </xs:annotation>
        <xs:attribute name="Name" type="xs:anyURI" use="required" />
    </xs:complexType>
```

The following table describes the attributes for the **OperatorBaseType** type.

Attribute	Type	Description
Name	**xs:anyURI**	The URI of the specified operator.

2.2.3.2.3.3 StreamReferenceType

The **StreamReferenceType** type is used to refer to a name that has been defined by a **StreamDefinition** element in another operator.

The following code is the XML schema (XSD) for the **StreamReferenceType** type.

```
    <xs:complexType name="StreamReferenceType">
        <xs:annotation>
            <xs:documentation>ID that refers to a stream.</xs:documentation>
        </xs:annotation>
        <xs:attribute name="Name" type="xs:anyURI" use="required" />
    </xs:complexType>
```

The following table describes the attributes for the **StreamReferenceType** type.

Attribute	Type	Description
Name	**xs:anyURI**	The URI of the stream that is referred to.

2.2.3.2.3.4 StreamDefinitionType

The **StreamDefinitionType** type is used to define a stream. It denotes an output stream from an operator in a QueryTemplate. It defines a name that can be referenced by the **StreamReference** element in another operator.

The following code is the XML schema (XSD) for the **StreamDefinitionType** type.

```
<xs:complexType name="StreamDefinitionType">
    <xs:annotation>
        <xs:documentation>ID that defines a stream. Stream here denotes the
        connection between operators.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="Name" type="xs:anyURI" use="required" />
</xs:complexType>
```

The following table describes the attributes for the **StreamDefinitionType** type.

Attribute	Type	Description
Name	**xs:anyURI**	The URI of the stream being defined.

2.2.3.2.3.5 ExpressionContainerType

The **ExpressionContainerType** type represents a base container for one of the expressions in the AnyExpression group.

The following code is the XML schema (XSD) for the **ExpressionContainerType** type.

```
<xs:complexType name="ExpressionContainerType">
    <xs:annotation>
        <xs:documentation>Expression container type. An element of this type
        must contain exactly one expression of any type.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
    </xs:sequence>
</xs:complexType>
```

The following table describes the element for the **ExpressionContainerType** type.

Element	Type	Description
(group)	**AnyExpression**	The expression contained in the specified container. This ~~must be~~is one of the expressions defined in the **AnyExpression** group.

2.2.3.2.3.6 TerminatorBaseType

The **TerminatorBaseType** type is used as a base type for stream termination elements, such as **Import**, **Export**, **ApplyInput**, and **ApplyOutput**.

The following code is the XML schema (XSD) for the **TerminatorBaseType** type.

```
<xs:complexType name="TerminatorBaseType">
    <xs:annotation>
        <xs:documentation>Base type for stream termination elements.
        </xs:documentation>
    </xs:annotation>
</xs:complexType>
```

The **TerminatorBaseType** type defines no elements or attributes.

2.2.3.2.3.7 AnyExpression Group

The **AnyExpression** group is the group that contains all of the expressions that are available for use in operations.

The following code is the XML schema (XSD) for the **AnyExpression** group.

```
<xs:group name="AnyExpression">
  <xs:annotation>
    <xs:documentation>Placeholder for exactly one expression element of any type.
    </xs:documentation>
  </xs:annotation>
  <xs:choice>
    <xs:element name="Abs" type="tns:UnaryArithmeticExpression" />
    <xs:element name="Add" type="tns:BinaryArithmeticExpression" />
    <xs:element name="And" type="tns:BinaryExpression" />
    <xs:element name="BitwiseAnd" type="tns:BinaryExpression" />
    <xs:element name="BitwiseOr" type="tns:BinaryExpression" />
    <xs:element name="BitwiseXor" type="tns:BinaryExpression" />
    <xs:element name="Compare" type="tns:ComparisonExpression" />
    <xs:element name="Condition" type="tns:ConditionExpression" />
    <xs:element name="Constant" type="tns:ConstantExpression" />
    <xs:element name="Convert" type="tns:ConvertExpression" />
    <xs:element name="Divide" type="tns:BinaryArithmeticExpression" />
    <xs:element name="Equal" type="tns:ComparisonExpression" />
    <xs:element name="EventKind" type="tns:SystemFieldExpression" />
    <xs:element name="GreaterThan" type="tns:ComparisonExpression" />
    <xs:element name="GreaterThanOrEqual" type="tns:ComparisonExpression" />
    <xs:element name="Hash" type="tns:HashExpression" />
    <xs:element name="InputField" type="tns:InputFieldExpression" />
    <xs:element name="LessThan" type="tns:ComparisonExpression" />
    <xs:element name="LessThanOrEqual" type="tns:ComparisonExpression" />
    <xs:element name="Max" type="tns:NaryArithmeticExpression" />
    <xs:element name="MethodCall" type="tns:MethodCallExpression" />
    <xs:element name="Min" type="tns:NaryArithmeticExpression" />
    <xs:element name="Modulo" type="tns:BinaryArithmeticExpression" />
    <xs:element name="Multiply" type="tns:BinaryArithmeticExpression" />
    <xs:element name="NewValidEndTime" type="tns:SystemFieldExpression" />
    <xs:element name="Negate" type="tns:UnaryArithmeticExpression" />
    <xs:element name="Not" type="tns:UnaryExpression" />
    <xs:element name="NotEqual" type="tns:ComparisonExpression" />
    <xs:element name="Or" type="tns:BinaryExpression" />
    <xs:element name="Subtract" type="tns:BinaryArithmeticExpression" />
    <xs:element name="ValidStartTime" type="tns:SystemFieldExpression" />
    <xs:element name="ValidEndTime" type="tns:SystemFieldExpression" />
  </xs:choice>
```

```
    </xs:group>
```

The following table describes the elements for the **AnyExpression** group.

Element	Type	Description
Abs	UnaryArithmeticExpression	The absolute value of a single operand.
Add	BinaryArithmeticExpression	An expression that performs an addition operation on two operands.
And	BinaryExpression	An expression that performs a logical And operation on two operands.
BitwiseAnd	**BinaryExpression**	Bitwise AND expression.
BitwiseOr	**BinaryExpression**	Bitwise OR expression.
BitwiseXor	**BinaryExpression**	Bitwise XOR expression.
Compare	ComparisonExpression	An expression that compares two operands. The result is negative if the first operand is less than the second operand, positive if the first operand is greater than the second, and zero if the two operands are equal.
Condition	**ConditionExpression**	An expression that performs a conditional evaluation. The first expression is evaluated to a Boolean value. If true, then the condition expression is assigned the second expression; otherwise, it is assigned the third expression.
Constant	ConstantExpression	An expression that specifies a constant.
Convert	ConvertExpression	An expression that converts a value to a new type.
Divide	**BinaryArithmeticExpression**	An expression that performs a division operation on two operands.
Equal	**ComparisonExpression**	An expression that compares two operands for equality.
EventKind	SystemFieldExpression	An expression that returns the event type as a number: 1 - Event is a current time increment (CTI) 2 - Event is an Insert 3 - Event is a Retract 4 - Event is an Expand
GreaterThan	**ComparisonExpression**	An expression that tests for Greater Than condition on two operands.
GreaterThanOrEqual	**ComparisonExpression**	An expression that tests for the Greater Than Or Equal condition on two operands.
Hash	HashExpression	An expression that specifies a hash expression. It computes a hash value on *n* child operands.
InputField	InputFieldExpression	An expression that returns the value of a field in an ~~event's~~event’s payload.
LessThan	**ComparisonExpression**	An expression that tests for the Less Than condition on two operands.

Element	Type	Description
LessThanOrEqual	**ComparisonExpression**	An expression that tests for the Less Than Or Equal condition on two operands.
Max	NaryArithmeticExpression	An expression that determines the maximum value of *n* operands.
MethodCall	MethodCallExpression	An expression that calls a method with *n* child operands as its arguments.
Min	**NaryArithmeticExpression**	An expression that determines the minimum value of *n* operands.
Modulo	**BinaryArithmeticExpression**	An expression that returns the remainder of integer division of one number by another number.
Multiply	**BinaryArithmeticExpression**	An expression that performs a multiplication operation on two operands.
NewValidEndTime	**SystemFieldExpression**	An expression that returns the new valid end time system field.
Negate	**UnaryArithmeticExpression**	An expression that negates a numeric expression.
Not	UnaryExpression	An expression that returns the logical Not operation on an operand.
NotEqual	**ComparisonExpression**	An expression that determines whether two operands are Not Equal.
Or	**BinaryExpression**	An expression that performs a logical Or operation on two operands.
Subtract	**BinaryArithmeticExpression**	An expression that performs subtraction on two operands.
ValidEndTime	**SystemFieldExpression**	An expression that refers to the end time of the specified event.
ValidStartTime	**SystemFieldExpression**	An expression that refers to the start time of the specified event.

2.2.3.2.3.7.1 UnaryArithmeticExpression

The **UnaryArithmeticExpression** type is used as a base type for unary arithmetic expressions.

The following code is the XML schema (XSD) for the **UnaryArithmeticExpression** type.

```
<xs:complexType name="UnaryArithmeticExpression">
    <xs:annotation>
        <xs:documentation>Unary arithmetic expression. Has 1 child
        expression and no attributes.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:restriction base="tns:UnaryExpression">
            <xs:sequence>
                <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
            </xs:sequence>
        </xs:restriction>
    </xs:complexContent>
```

```
        </xs:complexType>
```

The following table describes the element for the **UnaryArithmeticExpression** type.

Element	Type	Description
(group)	AnyExpression	The single child operand.

2.2.3.2.3.7.2 BinaryArithmeticExpression

The **BinaryArithmeticExpression** type is used as a base type for binary arithmetic expressions.

The following code is the XML schema (XSD) for the **BinaryArithmeticExpression** type.

```
        <xs:complexType name="BinaryArithmeticExpression">
            <xs:annotation>
                <xs:documentation>Binary arithmetic expression. Has 2 child
                expressions and no attributes.</xs:documentation>
            </xs:annotation>
            <xs:complexContent>
                <xs:restriction base="tns:BinaryExpression">
                    <xs:sequence>
                        <xs:group minOccurs="2" maxOccurs="2" ref="tns:AnyExpression" />
                    </xs:sequence>
                </xs:restriction>
            </xs:complexContent>
        </xs:complexType>
```

The following table describes the element for the **BinaryArithmeticExpression** type.

Element	Type	Description
(group)	AnyExpression	The two child operands.

2.2.3.2.3.7.3 ComparisonExpression

The **ComparisonExpression** type is used to compare two expressions. This type also ~~may~~can contain information about the culture to be used for the comparison.

The following code is the XML schema (XSD) for the **ComparisonExpression** type.

```
      <xs:complexType name="ComparisonExpression">
        <xs:annotation>
          <xs:documentation>Comparison expression. Compares two child expressions.
          The optional third child expression is the culture info. CompareOptions and
          StringComparison values are given as attributes here.</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
          <xs:extension base="tns:BinaryExpression">
            <xs:sequence>
              <xs:element name="CultureInfo" minOccurs="0" maxOccurs="1"
                          type="tns:CultureInfoExpression" />
            </xs:sequence>
            <xs:attribute name="CompareOptions"
                          type="tns:CompareOptionsParameterEnumType" use="optional" />
            <xs:attribute name="StringComparison"
                          type="tns:StringComparisonParameterEnum" use="optional" />
            <xs:attribute name="IgnoreCase" type="xs:boolean" use="optional" />
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
```

The following tables describe the elements and attributes for the **ComparisonExpression** type.

Element	Type	Description
CultureInfo	CultureInfoExpression	Contains a description of the culture information to uniquely identify a culture.

Attribute	Type	Description
CompareOptions	CompareOptionsParameterEnumType	The enumeration of types of comparisons that exist in .NET.
StringComparison	StringComparisonParameterEnum	The enumeration of types of string comparisons that exist in .NET.
IgnoreCase	**xs:boolean**	True indicates that character case ~~should be~~is ignored in comparisons. False indicates that case ~~should~~is not ~~be~~ ignored.

2.2.3.2.3.7.4 ConstantExpression

The **ConstantExpression** type is used to specify a constant as an input to another expression.

The following code is the XML schema (XSD) for the **ConstantExpression** type.

```
<xs:complexType name="ConstantExpression">
    <xs:annotation>
        <xs:documentation>Constant expression. Has no child expression.
        Contains type and value attributes.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:restriction base="tns:NullaryExpression">
            <xs:sequence />
            <xs:attributeGroup ref="tns:TypeIdentifier" />
            <xs:attribute name="Value" type="xs:string" use="required" />
            <xs:attribute name="NullValue" type="xs:boolean" use="optional"
                          default="false" />
        </xs:restriction>
    </xs:complexContent>
</xs:complexType>
```

The following table describes the attributes for the **ConstantExpression** type.

Attribute	Type	Description
(group)	TypeIdentifier	The type of the specified constant expression.

Attribute	Type	Description
Value	**xs:string**	The XML representation of the value according to its type.
NullValue	**xs:boolean**	If true, the value for the constant expression is null. Otherwise, the value for the constant expression is not null.

2.2.3.2.3.7.5 ConvertExpression

The **ConvertExpression** type is used to convert an expression to a different type.

The following code is the XML schema (XSD) for the **ConvertExpression** type.

```
<xs:complexType name="ConvertExpression">
    <xs:annotation>
        <xs:documentation>Conversion expression. Converts one child
        expression into a type.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:restriction base="tns:UnaryExpression">
            <xs:sequence>
                <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
            </xs:sequence>
            <xs:attributeGroup ref="tns:TypeIdentifier" />
            <xs:attribute name="DateTimeKind" type="tns:DateTimeType"
                          use="optional" />
            <xs:attributeGroup ref="tns:ExpressionReturnTypeFacets"/>
        </xs:restriction>
    </xs:complexContent>
</xs:complexType>
```

The following tables describe the elements and attributes for the **ConvertExpression** type.

Element	Type	Description
(group)	AnyExpression	The expression for the unary arithmetic.

Attribute	Type	Description
(group)	TypeIdentifier	The type that the expression is to be converted to.
DateTimeKind	DateTimeType	The type of date-time. For more information, see section 2.2.3.2.3.10.
(group)	**ExpressionReturnTypeFacets**	Additional facets about the return type of the expression.

2.2.3.2.3.7.6 HashExpression

The **HashExpression** type is used to perform a hashing operation on any number of operands.

The following code is the XML schema (XSD) for the **HashExpression** type.

```
<xs:complexType name="HashExpression">
    <xs:annotation>
        <xs:documentation>Hash expression. Represents a hash value based on
        1..n child expressions.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:restriction base="tns:ExpressionBase">
            <xs:sequence>
                <xs:group minOccurs="1" maxOccurs="unbounded"
                          ref="tns:AnyExpression" />
            </xs:sequence>
        </xs:restriction>
    </xs:complexContent>
</xs:complexType>
```

The following table describes the element for the **HashExpression** type.

Element	Type	Description
(group)	AnyExpression	The expressions that will be used to obtain the values over which the hash function will be computed.

2.2.3.2.3.7.7 InputFieldExpression

The **InputFieldExpression** type is used to identify an Input field from a specific stream.

The following code is the XML schema (XSD) for the **InputFieldExpression** type.

```
<xs:complexType name="InputFieldExpression">
    <xs:annotation>
        <xs:documentation>Input field expression. Has no child expression.
        Refers to a field in a stream by the field identifier.
      </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:restriction base="tns:NullaryExpression">
            <xs:sequence />
            <xs:attributeGroup ref="tns:FieldIdentifier" />
            <xs:attributeGroup ref="tns:StreamIdentifier" />
        </xs:restriction>
    </xs:complexContent>
</xs:complexType>
```

The following table describes the attributes for the **InputFieldExpression** type.

Attribute	Type	Description
(group)	FieldIdentifier	The input field to be used.
(group)	StreamIdentifier	The event stream to be used to look up the specified field.

2.2.3.2.3.7.8 NaryArithmeticExpression

The **NaryArithmeticExpression** type is used as a base type for arithmetic expressions with an arbitrary number of operands.

The following code is the XML schema (XSD) for the **NaryArithmeticExpression** type.

```
<xs:complexType name="NaryArithmeticExpression">
    <xs:annotation>
        <xs:documentation>N-ary arithmetic expression. Has 1..n child
        expressions and arbitrary attributes.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:restriction base="tns:ExpressionBase">
            <xs:sequence>
                <xs:group minOccurs="1" maxOccurs="unbounded"
                          ref="tns:AnyExpression" />
            </xs:sequence>
        </xs:restriction>
    </xs:complexContent>
</xs:complexType>
```

The following table describes the element for the **NaryArithmeticExpression** type.

Element	Type	Description
(group)	AnyExpression	The unbounded set of child operands.

2.2.3.2.3.7.9 MethodCallExpression

The **MethodCallExpression** type is used to call a .NET assembly to perform an operation on any number of operands. The possible methods include user-defined code as well as existing .NET methods.

The following code is the XML schema (XSD) for the **MethodCallExpression** type.

```
<xs:complexType name="MethodCallExpression">
  <xs:annotation>
    <xs:documentation>User-defined function. Its value is defined by a method
    of a class. 0..n input expressions can be passed to the method as parameters.
    In addition to CEP expressions, the input can also contain culture-related
    parameters as elements.</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="tns:ExpressionBase">
      <xs:sequence>
        <xs:group minOccurs="0" maxOccurs="unbounded"
                  ref="tns:AnyMethodCallSubExpression" />
      </xs:sequence>
      <xs:attribute name="Method" type="xs:string" use="required" />
      <xs:attribute name="Class" type="xs:string" use="required" />
      <xs:attribute default="false" name="Deterministic" type="xs:boolean"
                    use="optional" />
      <xs:attributeGroup ref="tns:TypeFacetAttributes" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The following tables describe the elements and attributes for the **MethodCallExpression** type.

Element	Type	Description
(group)	AnyMethodCallSubExpression	Contains a single element that is used as an argument to a method call.

Attribute	Type	Description
Method	**xs:string**	The name of the method to be called.
Class	**xs:string**	The name of the class that contains the method in a .NET assembly.
Deterministic	**xs:boolean**	If true, **MethodCallExpression** is deterministic. Otherwise, false.
(group)	TypeFacetAttributes	Attributes that specify additional information about the return type.

2.2.3.2.3.7.9.1 AnyMethodCallSubExpression Group

The **AnyMethodCallSubExpression** group contains a single element that is used as an argument for a method call. The **AnyMethodCallSubExpression** group extends the AnyExpression group by culture-related arguments, which the underlying method can use for operations that depend on culture information.

```
<xs:group name="AnyMethodCallSubExpression">
  <xs:annotation>
    <xs:documentation>Placeholder for exactly one element that can be used as
    arguments for method calls (CEP expressions plus culture parameters)
  </xs:documentation>
  </xs:annotation>
  <xs:choice>
    <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
    <xs:element name="CultureInfo" type="tns:CultureInfoExpression" />
    <xs:element name="CompareOptions" type="tns:CompareOptionsType" />
    <xs:element name="StringComparison" type="tns:StringComparisonType" />
  </xs:choice>
</xs:group>
```

The following table describes the elements for the **AnyMethodCallSubExpression** type.

Element	Type	Description
(group)	**AnyExpression**	One of the expressions contained in the **AnyExpression** group.
CultureInfo	CultureInfoExpression	Contains a description of culture information to uniquely identify a culture.
CompareOptions	CompareOptionsEnumType	Specifies the string comparison options to use.
StringComparison	StringComparisonEnum	Specifies the culture, case, and sort rules to be used for comparisons.

2.2.3.2.3.7.9.1.1 ComparisonOptionsType Type

The **CompareOptionsType** type represents a .NET **CompareOptions** object that specifies the string comparison options to use.

```
<xs:complexType name="CompareOptionsType">
  <xs:annotation>
    <xs:documentation>Represents a .NET CompareOptions object to use with
    CompareInfo as an element. Can be a parameter for a method call
    expression.</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:restriction base="tns:NullaryExpression">
      <xs:sequence />
      <xs:attribute name="Value" type="tns:CompareOptionsParameterEnumType"
                    use="required" />
    </xs:restriction>
  </xs:complexContent>
</xs:complexType>
```

The following table describes the attributes for the **CompareOptionsType** type.

Attribute	Type	Description
Value	CompareOptionsParameterEnumType	An enumeration that contains the compare options.

2.2.3.2.3.7.9.1.2 StringComparisonType Type

The **StringComparisonType** type represents a .NET **StringComparison** object that specifies the culture, case, and sort rules to be used for comparisons.

```
<xs:complexType name="StringComparisonType">
  <xs:annotation>
    <xs:documentation>Represents a .NET StringComparison object to use with
    .Net String.Compare and String.Equals as an element. Can be a parameter
    for a method call expression.</xs:documentation>
  </xs:annotation>
  <xs:sequence />
  <xs:attribute name="Value" type="tns:StringComparisonParameterEnum"
                use="required" />
</xs:complexType>
```

The following table describes the attributes for the **StringComparisonType** type.

Attribute	Type	Description
Value	StringComparisonParameterEnum	An enumeration that contains string comparison options.

2.2.3.2.3.7.10 UnaryExpression

The **UnaryExpression** type is used to specify a single operand on which a unary operation ~~may~~can be performed.

The following code is the XML schema (XSD) for the **UnaryExpression** type.

```
<xs:complexType name="UnaryExpression">
    <xs:annotation>
        <xs:documentation>Unary expression. Has 1 child expression.
        </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:restriction base="tns:ExpressionBase">
            <xs:sequence>
                <xs:group minOccurs="1" maxOccurs="1"
                          ref="tns:AnyExpression" />
            </xs:sequence>
            <xs:anyAttribute namespace="##any" />
        </xs:restriction>
    </xs:complexContent>
</xs:complexType>
```

The following tables describe the elements and attributes for the **UnaryExpression** type.

Element	Type	Description
(group)	AnyExpression	The expression for the unary expression.

Attribute	Type	Description
anyAttribute	**attributeGroup**	A placeholder attribute group that enables extensions to the specified class to define specific attributes.

2.2.3.2.3.7.11 BinaryExpression

The **BinaryExpression** type is used as a base type for binary expressions.

The following code is the XML schema (XSD) for the **BinaryExpression** type.

```
<xs:complexType name="BinaryExpression">
    <xs:annotation>
        <xs:documentation>Binary expression. Has 2 child expressions and
        arbitrary attributes.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:restriction base="tns:ExpressionBase">
            <xs:sequence>
                <xs:group minOccurs="2" maxOccurs="2" ref="tns:AnyExpression" />
            </xs:sequence>
            <xs:anyAttribute namespace="##any" />
        </xs:restriction>
    </xs:complexContent>
</xs:complexType>
```

The following tables describe the elements and attributes for the **BinaryExpression** type.

Element	Type	Description
(group)	AnyExpression	Exactly two expressions for a binary operation.

Attribute	Type	Description
anyAttribute	**attributeGroup**	A placeholder attribute group that enables extensions to the specified class to define specific attributes.

2.2.3.2.3.7.12 SystemFieldExpression

The **SystemFieldExpression** type is used to define the elements for the system field access expressions: **ValidStartTime**, **ValidEndTime**, **NewValidEndTime**, and **EventKind**.

The following code is the XML schema (XSD) for the **SystemFieldExpression** type.

```
<xs:complexType name="SystemFieldExpression">
    <xs:annotation>
        <xs:documentation>System field expression. Has no child expression.
        Refers to a system field in a stream.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:restriction base="tns:NullaryExpression">
            <xs:sequence />
            <xs:attributeGroup ref="tns:StreamIdentifier" />
        </xs:restriction>
    </xs:complexContent>
</xs:complexType>
```

The following table describes the attributes for the **SystemFieldExpression** type.

Attribute	Type	Description
(group)	StreamIdentifier	The stream that contains the system field.

2.2.3.2.3.8 NullaryExpression

The **NullaryExpression** type is the base type for expressions that take no operands.

The following code is the XML schema (XSD) for the **NullaryExpression** type.

```
<xs:complexType name="NullaryExpression">
    <xs:annotation>
        <xs:documentation>Nullary expression. Has no child expressions.
        </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:restriction base="tns:ExpressionBase">
            <xs:sequence />
            <xs:anyAttribute namespace="##any" />
        </xs:restriction>
    </xs:complexContent>
</xs:complexType>
```

The following table describes the attributes for the **NullaryExpression** type.

Attribute	Type	Description
anyAttribute	**attributeGroup**	A placeholder attribute group that enables extensions to the specified class to define specific attributes.

2.2.3.2.3.9 TypeIdentifier AttributeGroup

The **TypeIdentifier** attribute group is used to define an ~~expression's~~expression’s type. It contains a single attribute to specify the type name.

The following code is the XML schema (XSD) for the **TypeIdentifier** attribute group.

```
<xs:attributeGroup name="TypeIdentifier">
  <xs:annotation>
    <xs:documentation>Refers to a data type and facets in the
    StreamInsight type system.</xs:documentation>
  </xs:annotation>
  <xs:attribute name="Type" type="tns:PrimitiveTypeIdentifier"
               use="required" />
  <xs:attributeGroup ref="tns:TypeFacetAttributes" />
</xs:attributeGroup>
```

The following table describes the attributes for the **TypeIdentifier** attribute group.

Attribute	Type	Description
Type	**PrimitiveTypeIdentifier**	A **PrimitiveTypeIdentifier** that refers to a primitive type.
(group)	TypeFacetAttributes	Type facets (additional typing information) for this **TypeIdentifier**

2.2.3.2.3.10 DateTimeType

The **DateTimeType** type is an enumeration that indicates whether date and time values are based on local time or on UTC.

The following code is the XML schema (XSD) for the **DateTimeType** type.

```
<xs:simpleType name="DateTimeType">
    <xs:restriction base="xs:string">
        <xs:enumeration value="Utc" />
        <xs:enumeration value="Local" />
    </xs:restriction>
</xs:simpleType>
```

The following table describes the enumeration values for the **DateTimeType** type.

Value	Description
Utc	UTC time
Local	Local time

2.2.3.2.3.11 TypeFacetAttributes AttributeGroup

The **TypeFacetAttributes** attribute group is used to specify additional information about types for which the complex event processing (CEP) engine would not be able to infer that information on its own.

The following code is the XML schema (XSD) for the **TypeFacetAttributes** attribute group.

```
<xs:attributeGroup name="TypeFacetAttributes">
  <xs:annotation>
    <xs:documentation>Type identifier and facets.</xs:documentation>
  </xs:annotation>
  <xs:attribute name="Nullable" type="xs:boolean" use="required" />
  <xs:attribute name="Culture" type="xs:string" use="optional" />
  <xs:attribute name="MaxSize" type="xs:unsignedInt" use="optional">
    <xs:annotation>
      <xs:documentation>MaxSize is only applicable to string and byte
      array types. For string, this is the number of characters, for byte
      array this is the number of bytes.</xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="SizeFixed" type="xs:boolean" use="optional">
    <xs:annotation>
      <xs:documentation>SizeFixed is only applicable to string and byte
      array types. It denotes a field of a fixed size.</xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:attributeGroup>
```

The following table describes the attributes for the **TypeFacetAttributes** attribute group.

Attribute	Type	Description
MaxSize	**xs:unsignedInt**	The maximum size — in bytes or characters — for byte arrays or strings, respectively.
SizeFixed	**xs:boolean**	If true, specified type is a fixed-size type. Otherwise, false. This is applicable only to string and byte array types.
Nullable	**xs:boolean**	An attribute that indicates whether the specified value is nullable.
Culture	**xs:string**	The culture of the specified field. The format of this string is specified in [ISO639-2] and [ISO3166-1:2006]. The culture is used as the default culture for comparison operators. For more information, see [MSDN-CIPN].

2.2.3.2.3.12 StreamIdentifier AttributeGroup

The **StreamIdentifier** attribute group is used to identify a stream that has already been defined. This is necessary, for instance, in join predicate expressions, which can refer to event fields from multiple input streams.

The following code is the XML schema (XSD) for the **StreamIdentifier** attribute group.

```
<xs:attributeGroup name="StreamIdentifier">
    <xs:annotation>
```

```
            <xs:documentation>Refers to a stream by the stream name that was
            defined in the corresponding scope.</xs:documentation>
        </xs:annotation>
        <xs:attribute name="StreamName" type="xs:anyURI" use="optional" />
    </xs:attributeGroup>
```

The following table describes the attributes for the **StreamIdentifier** attribute group.

Attribute	Type	Description
StreamName	**xs:anyURI**	The URI of a stream that has already been defined.

2.2.3.2.3.13 ExpressionBase

The **ExpressionBase** type is the base type on which other expressions are defined as an extension or a restriction.

The following code is the XML schema (XSD) for the **ExpressionBase** type.

```
    <xs:complexType name="ExpressionBase">
        <xs:annotation>
            <xs:documentation>Expression base type. Can have 0..n child expressions.
            </xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:group minOccurs="0" maxOccurs="unbounded" ref="tns:AnyExpression" />
        </xs:sequence>
        <xs:anyAttribute namespace="##any" />
    </xs:complexType>
```

The following tables describe the elements and attributes for the **ExpressionBase** type.

Element	Type	Description
(group)	AnyExpression	An expression.

Attribute	Type	Description
anyAttribute	**attributeGroup**	A placeholder attribute group that enables extensions to the specified class to define specific attributes.

2.2.3.2.3.14 FieldIdentifier

The **FieldIdentifier** attribute group is used to identify a field by its name.

The following code is the XML schema (XSD) for the **FieldIdentifier** attribute group.

```
    <xs:attributeGroup name="FieldIdentifier">
        <xs:annotation>
            <xs:documentation>Refers to a field within a stream type by its name.
            </xs:documentation>
        </xs:annotation>
        <xs:attribute name="Name" type="xs:anyURI" use="required" />
    </xs:attributeGroup>
```

The following table describes the attributes for the **FieldIdentifier** attribute group.

Attribute	Type	Description
Name	**xs:anyURI**	The URI for the name of a field that has already been defined.

2.2.3.2.3.15 AnySingleUserElementType

The **AnySingleUserElementType** type is used for users to define arbitrary XML content for an element.

The following code is the XML schema (XSD) for the **AnySingleUserElementType** type.

```
<xs:complexType name="AnySingleUserElementType">
  <xs:annotation>
    <xs:documentation>Contains one user-defined XML element.
    The element has to define a separate namespace.</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:any minOccurs="1" maxOccurs="1" namespace="##any"
            processContents="skip" />
  </xs:sequence>
</xs:complexType>
```

The following table describes the element for the **AnySingleUserElementType** type.

Element	Type	Description
(any XML)	**xs:any**	The XML element contained in this element is user-defined and is not interpreted by the complex event processing (CEP) server or the CEPM protocol. This element and its contents is passed on to an external component for possible processing or use by that component.

2.2.3.2.3.16 EventShapeType

The **EventShapeType** type is an enumeration that indicates the shape of the event.

The following code is the XML schema (XSD) for the **EventShapeType** type.

```
<xs:simpleType name="EventShapeType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Point" />
    <xs:enumeration value="Interval" />
    <xs:enumeration value="Edge" />
  </xs:restriction>
</xs:simpleType>
```

The following table describes the enumeration values for the **EventShapeType** type.

Value	Description
Point	A **Point** event occurs at a single point in time. For the point event, only the start time is relevant. The lifetime is implicitly set to be equal to the smallest possible time span (one tick as defined in the CLR).
Interval	An event with a start time and an end time.
Edge	Edge events separately encode the start and end of a single interval event. An edge event can be of type "start" or "end".

2.2.3.2.3.17 ImplementationType

The **ImplementationType** type contains information about the CLR implementation of a user-defined element. Such an element (user-defined aggregate or user-defined operator) is always implemented in a class that derives from the proper base class provided by the .NET API.

The following code is the XML schema (XSD) for the **ImplementationType** type.

```
<xs:complexType name="ImplementationType">
  <xs:annotation>
    <xs:documentation>Specifies the signature of a user-defined
    operation/aggregation.</xs:documentation>
  </xs:annotation>
  <xs:attribute name="Class" type="xs:string" use="required">
    <xs:annotation>
      <xs:documentation>The .Net strong name of the implemented
      class.</xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="InputClrType" type="xs:string" use="required">
    <xs:annotation>
      <xs:documentation>The input type as a CLR strong name.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="ReturnClrType" type="xs:string" use="required">
    <xs:annotation>
      <xs:documentation>The output type as a CLR strong name.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:complexType>
```

The following table describes the attributes for the **ImplementationType** type.

Attribute	Type	Description
Class	**xs:string**	Fully qualified .NET Framework class name of the implementation.
InputClrType	**xs:string**	Fully qualified input type name of the implementation.
ReturnClrType	**xs:string**	Fully qualified output type name of the implementation.

2.2.3.2.3.18 SerializedConfigurationType

The **SerializedConfigurationType** type contains the fully serialized XML definition of a user-defined type.

The following code is the XML schema (XSD) for the **SerializedConfigurationType** type.

```
<xs:complexType name="SerializedConfigurationType">
  <xs:annotation>
    <xs:documentation>Runtime configuration structure for the UDO/UDA.
    </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="tns:AnySingleUserElementType">
      <xs:attribute name="Class" type="xs:string" use="required">
        <xs:annotation>
          <xs:documentation>Serialized class name of the configuration structure.
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The following table describes the attributes for the **SerializedConfigurationType** type.

Attribute	Type	Description
Class	**xs:string**	The .NET Framework strong class name.

2.2.3.2.3.19 CultureInfoExpression Type

The **CultureInfoExpression** type contains information that uniquely references a culture.

The following code is the XML schema (XSD) for the **CultureInfoExpression** type.

```
<xs:complexType name="CultureInfoExpression">
  <xs:annotation>
    <xs:documentation>Contains the description of a culture info to uniquely
    define a culture, either through a constant string or an event field
    reference. Can only be a parameter for a method call expression or a
    comparison expression.</xs:documentation>
  </xs:annotation>
  <xs:choice>
    <xs:element name="Constant" type="tns:ConstantExpression" />
    <xs:element name="InputField" type="tns:InputFieldExpression" />
  </xs:choice>
</xs:complexType>
```

The following table describes the elements for the **CultureInfoExpression** type.

Element	Type	Description
Constant	ConstantExpression	A **ConstantExpression** that contains the text for the culture information.
InputField	InputFieldExpression	An **InputFieldExpression** that contains the culture information.

2.2.3.2.3.20 CompareOptionsParameterEnumType Type

The **CompareOptionsParameterEnumType** type is an enumeration of the .NET Framework compare options. For more information, see [MSDN-CompareOptions].

The following code is the XML schema (XSD) for the **CompareOptionsParameterEnumType** type.

```
<xs:simpleType name="CompareOptionsParameterEnumType">
  <xs:annotation>
    <xs:documentation>List of all values for .Net CompareOptions.
    </xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:string">
    <xs:enumeration value="None" />
    <xs:enumeration value="IgnoreCase" />
    <xs:enumeration value="IgnoreNonSpace" />
    <xs:enumeration value="IgnoreSymbols" />
    <xs:enumeration value="IgnoreKanaType" />
    <xs:enumeration value="IgnoreWidth" />
    <xs:enumeration value="OrdinalIgnoreCase" />
    <xs:enumeration value="StringSort" />
    <xs:enumeration value="Ordinal" />
  </xs:restriction>
</xs:simpleType>
```

The following table contains the enumeration values and their descriptions for the **CompareOptionsParameterEnumType** type.

Value	Description
None	Indicates the default option settings for string comparisons.
IgnoreCase	Indicates that the string comparison ~~must ignore~~ignores case.
IgnoreNonSpace	Indicates that the string comparison ~~must ignore~~ignores nonspacing combining characters, such as diacritics. Nonspacing combining characters do not occupy a spacing position by themselves when rendered.
IgnoreSymbols	Indicates that the string comparison ~~must ignore~~ignores symbols, such as white-space characters, punctuation, currency symbols, the percent sign, mathematical symbols, and the ampersand.
IgnoreKanaType	Indicates that the string comparison ~~must ignore~~ignores kanatype. Kanatype refers to Japanese hiragana and katakana characters, which represent phonetic sounds in the Japanese language.
IgnoreWidth	Indicates that the string comparison ~~must ignore~~ignores the character width. For example, Japanese katakana characters can be written as full-width or half-width. If this value is selected, the katakana characters written as full-width are considered equal to the same characters written as half-width.
OrdinalIgnoreCase	Indicates that the string comparison ~~must ignore~~ignores case and then perform an ordinal comparison. This technique is equivalent to converting the string to uppercase using the invariant culture and then performing an ordinal comparison on the result.
StringSort	Indicates that the string comparison ~~must use~~uses the string sort algorithm. In a string sort, the hyphen and the apostrophe, as well as other non-alphanumeric symbols, come before alphanumeric characters.
Ordinal	Indicates that the string comparison ~~must use~~uses the Unicode values of each character. This technique leads to a fast comparison but one that is culture-insensitive.

2.2.3.2.3.21 StringComparisonParameterEnum Type

The **StringComparisonParameterEnum** type represents the values contained in the .NET Framework **StringComparison** object. For more information, see [MSDN-StringComparison].

The following code is the XML schema (XSD) for the **StringComparisonParameterEnum** type.

```
<xs:simpleType name="StringComparisonParameterEnum">
  <xs:annotation>
    <xs:documentation>List of all values for .Net StringComparison.
    </xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:string">
    <xs:enumeration value="CurrentCulture" />
    <xs:enumeration value="CurrentCultureIgnoreCase" />
    <xs:enumeration value="InvariantCulture" />
    <xs:enumeration value="InvariantCultureIgnoreCase" />
    <xs:enumeration value="Ordinal" />
    <xs:enumeration value="OrdinalIgnoreCase" />
  </xs:restriction>
</xs:simpleType>
```

The following table contains the enumeration values and their descriptions for the **StringComparisonParameterEnum** type.

Value	Description
CurrentCulture	Compares strings by using culture-sensitive sort rules and the current culture.
CurrentCultureIgnoreCase	Compares strings by using culture-sensitive sort rules and the current culture. The case of the strings being compared is ignored.
InvariantCulture	Compares strings by using culture-sensitive sort rules and the invariant culture.
InvariantCultureIgnoreCase	Compares strings by using culture-sensitive sort rules and the invariant culture. The case of the strings being compared is ignored.
Ordinal	Compares strings by using ordinal sort rules.
OrdinalIgnoreCase	Compares strings by using ordinal sort rules. The case of the strings being compared is ignored.

2.2.3.2.3.22 WindowedOperatorBaseType

The **WindowedOperatorBaseType** type is a base type that is used for definition of a set-based operator, that is, an operator that processes windows of events. The set of possible such windows are a snapshot window, a hopping window, and a CountByStartTime window.

The following code is the XML schema (XSD) for the **WindowedOperatorBaseType** type.

```
<xs:complexType name="WindowedOperatorBaseType">
  <xs:annotation>
    <xs:documentation>Windowed Operator base type. Includes the definition
    for windows.</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="tns:OperatorBaseType">
      <xs:sequence>
        <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
                    type="tns:StreamReferenceType" />
```

```
            <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
                        type="tns:StreamDefinitionType" />
            <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyWindow" />
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
```

The following table describes the elements for the **WindowedOperatorBaseType** type.

Element	Type	Description
InputStream	StreamReferenceType	A reference to the stream that will be the input to this operator.
OutputStream	StreamDefinitionType	The definition of the stream that will be the output from this operator.
(group)	AnyWindow	The definition of the window.

2.2.3.2.3.22.1 AnyWindow Group

The **AnyWindow** group contains the elements for defining windows.

The following code is the XML schema (XSD) for the **AnyWindow** group.

```
<xs:group name="AnyWindow">
  <xs:annotation>
    <xs:documentation>Placeholder for exactly one window element
    of any type.</xs:documentation>
  </xs:annotation>
  <xs:choice>
    <xs:element name="SnapshotWindow" type="tns:SnapshotWindowType" />
    <xs:element name="HoppingWindow" type="tns:HoppingWindowType" />
    <xs:element name="CountByStartTimeWindow"
                type="tns:CountByStartTimeWindowType" />
  </xs:choice>
</xs:group>
```

The following table describes the elements for the **AnyWindow** group.

Element	Type	Description
SnapshotWindow	SnapshotWindowType	If present, the window is a snapshot window and the **SnapshotWindow** element contains its definition.
HoppingWindow	HoppingWindowType	If present, the window is a hopping window and the **HoppingWindow** element contains its definition.
CountByStartTimeWindow	CountByStartTimeWindowType	If present, the window is a CountByStartTime window, and the **CountByStartTimeWindow** element contains its definition.

2.2.3.2.3.22.1.1 SnapshotWindowType Type

The **SnapshotWindowType** type contains the definition of a snapshot window.

The following code is the XML schema (XSD) for the **SnapshotWindowType** type.

```
<xs:complexType name="SnapshotWindowType">
  <xs:sequence>
    <xs:element name="WindowDefinition"
                type="tns:SnapshotWindowDefinitionType" />
    <xs:element name="InputPolicy"
                type="tns:WindowInputPolicyType" />
    <xs:element name="OutputPolicy"
                type="tns:SnapshotWindowOutputPolicyType" />
  </xs:sequence>
</xs:complexType>
```

The following table describes the elements for the **SnapshotWindowType** type.

Element	Type	Description
WindowDefinition	SnapshotWindowDefinitionType	Definition of a snapshot window.
InputPolicy	WindowInputPolicyType	The input policy for the snapshot window.
OutputPolicy	SnapshotWindowOutputPolicyType	The output policy for the snapshot window.

2.2.3.2.3.22.1.1.1 SnapshotWindowDefinitionType Type

The **SnapshotWindowDefinitionType** type contains the details of the definition of the snapshot window. The temporal characteristics of a snapshot window are defined dynamically by the events themselves. As a consequence, this is an empty element and simply serves to designate that the window being defined is a snapshot window.

The following code is the XML schema (XSD) for the **SnapshotWindowDefinitionType** type.

```
<xs:complexType name="SnapshotWindowDefinitionType">
  <xs:annotation>
    <xs:documentation>Snapshot window. Temporal window properties are
    defined by the stream of events.</xs:documentation>
  </xs:annotation>
  <xs:sequence />
</xs:complexType>
```

2.2.3.2.3.22.1.1.2 SnapshotWindowOutputPolicyType Type

The **SnapshotWindowOutputPolicyType** type specifies how the lifetime of the output on a snapshot window will be clipped or adjusted. This type is a specialization of the more generic **WindowOutputPolicyType**.

The following code is the XML schema (XSD) for the **SnapshotWindowOutputPolicyType** type.

```
<xs:complexType name="SnapshotWindowOutputPolicyType">
  <xs:choice>
    <xs:element name="Unaltered" />
    <xs:element name="Clip" type="tns:SnapshotOutputPolicyClipType" />
    <xs:element name="Adjust" type="tns:SnapshotOutputPolicyAdjustType" />
  </xs:choice>
```

```
</xs:complexType>
```

The following table describes the elements for the **SnapshotWindowOutputPolicyType** type.

Element	Type	Description
Unaltered	[empty element]	If present, this empty element signifies that there is no clipping or adjusting.
Clip	SnapshotOutputPolicyClipType	If present, this element signifies that events are clipped, and specifies the type of clipping.
Adjust	SnapshotOutputPolicyAdjustType	If present, this element signifies that events are adjusted, and specifies the type of adjustment.

2.2.3.2.3.22.1.1.3 SnapshotWindowOutputPolicyClipType Type

The **SnapshotWindowOutputPolicyClipType** type specifies the possible types of clipping of the output from an operation on a snapshot window.

The following code is the XML schema (XSD) for the **SnapshotWindowOutputPolicyClipType** type.

```
<xs:complexType name="SnapshotWindowOutputPolicyClipType">
  <xs:sequence />
  <xs:attribute name="Type" type="tns:SnapshotWindowOutputPolicyClipEnumType"
                use="required" />
</xs:complexType>
```

The following table describes the attribute for the **SnapshotWindowOutputPolicyClipType** type.

Attribute	Type	Description
Type	**base=xs:string**	Type is set to an enumeration value contained in the **SnapshotWindowOutputPolicyClipEnumType** type. The **SnapshotWindowOutputPolicyClipEnumType** enumeration indicates the type of clipping that can be applied to the ~~events'~~events' lifetimes produced by the set-based operation. The enumeration value is as follows: ▪ **WindowEnd**: The ~~events'~~events' lifetimes will be clipped at the window end.

2.2.3.2.3.22.1.1.4 SnapshotOutputPolicyAdjustType Type

The **SnapshotOutputPolicyAdjustType** type specifies how the output from a snapshot window is to be adjusted.

The following code is the XML schema (XSD) for the **SnapshotOutputPolicyAdjustType** type.

```
<xs:complexType name="SnapshotOutputPolicyAdjustType">
  <xs:sequence />
  <xs:attribute name="Lifetime" type="tns:SnapshotOutputAdjustLifetimeEnumType"
                use="required" />
```

```
    <xs:attribute name="Alignment" type="tns:SnapshotOutputAdjustAlignmentEnumType"
                  use="required" />
  </xs:complexType>
```

The following table describes the attributes for the **SnapshotOutputPolicyAdjustType** type.

Attribute	Type	Description
Lifetime	**base=xs:string**	An enumeration that indicates the manner of adjustment of the events' lifetimes produced by the set-based operation. The **SnapshotOutputAdjustLifetimeEnumType** enumeration values are as follows: ▪ **WindowSize**: Lifetime is set equal to the window size. ▪ **Point**: Lifetime is considered to be a single point in time.
Alignment	**base=xs:string**	An enumeration that indicates the manner of alignment of the events' lifetimes produced by the set-based operation.The **SnapshotOutputAdjustAlignmentEnumType** enumeration values are as follows: ▪ **WindowStart**: Events are aligned at window start time. ▪ **WindowEnd**: Events are aligned at window end time.

2.2.3.2.3.22.1.2 HoppingWindowType Type

The **HoppingWindowType** type contains the definition of a hopping window and the definition of, or reference to, the input and output streams for the hopping window.

The following code is the XML schema (XSD) for the **HoppingWindowType** type.

```
  <xs:complexType name="HoppingWindowType">
    <xs:sequence>
      <xs:element name="WindowDefinition" type="tns:HoppingWindowDefinitionType" />
      <xs:element name="InputPolicy" type="tns:WindowInputPolicyType" />
      <xs:element name="OutputPolicy" type="tns:WindowOutputPolicyType" />
    </xs:sequence>
  </xs:complexType>
```

The following table describes the elements for the **HoppingWindowType** type.

Element	Type	Description
WindowDefinition	HoppingWindowDefinitionType	The definition of the hopping window.
InputPolicy	WindowInputPolicyType	The input policy for the hopping window.
OutputPolicy	WindowOutputPolicyType	The output policy for the hopping window.

2.2.3.2.3.22.1.2.1 HoppingWindowDefinitionType Type

The **HoppingWindowDefinitionType** type specifies the definition of a hopping window.

The following code is the XML schema (XSD) for the **HoppingWindowDefinitionType** type.

```
<xs:complexType name="HoppingWindowDefinitionType">
  <xs:annotation>
    <xs:documentation>Fixed length window. Defined by a fixed window size,
    a hop size and an optional alignment.</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element minOccurs="1" maxOccurs="1" name="Size" type="xs:duration" />
    <xs:element minOccurs="1" maxOccurs="1" name="HopSize"
                type="xs:duration" />
    <xs:element minOccurs="1" maxOccurs="1" name="Alignment"
                type="xs:dateTime" />
  </xs:sequence>
</xs:complexType>
```

The following table describes the elements for the **HoppingWindowDefinitionType** type.

Element	Type	Description
Size	**xs:duration**	The duration of the hopping window.
HopSize	**xs:duration**	The size of the hop between windows.
Alignment	**xs:dateTime**	The alignment of the fixed-size hopping window along the timeline.

2.2.3.2.3.22.1.3 CountByStartTimeWindowType Type

The **CountByStartTimeWindowType** type contains the definition of a CountByStartTime window.

The following code is the XML schema (XSD) for the **CountByStartTimeWindowType** type.

```
<xs:complexType name="CountByStartTimeWindowType">
  <xs:sequence>
    <xs:element name="WindowDefinition"
                type="tns:CountByStartTimeWindowDefinitionType" />
    <xs:element name="InputPolicy" type="tns:WindowInputPolicyType" />
    <xs:element name="OutputPolicy" type="tns:WindowOutputPolicyType" />
  </xs:sequence>
</xs:complexType>
```

The following table describes the elements for the **CountByStartTimeWindowType** type.

Element	Type	Description
WindowDefinition	CountByStartTimeWindowDefinitionType	The definition of the **CountByStartTime** window.
InputPolicy	WindowInputPolicyType	The input policy for the **CountByStartTime** window.
OutputPolicy	WindowOutputPolicyType Type	The output policy for the **CountByStartTime**

Element	Type	Description
		window.

2.2.3.2.3.22.1.3.1 CountByStartTimeWindowDefinitionType Type

The **CountByStartTimeWindowDefinitionType** type specifies the definition of a CountByStartTime window.

The following code is the XML schema (XSD) for the **CountByStartTimeWindowDefinitionType** type.

```
<xs:complexType name="CountByStartTimeWindowDefinitionType">
  <xs:annotation>
    <xs:documentation>Count start times window. Defined by the count of
    member event start times.</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element minOccurs="1" maxOccurs="1" name="Size" type="xs:int" />
    <xs:element minOccurs="1" maxOccurs="1" name="HopSize" type="xs:int" />
  </xs:sequence>
</xs:complexType>
```

The following table describes the elements of the **CountByStartTimeWindowDefinitionType** type.

Element	Type	Description
Size	**xs:int**	The size of the window in terms of number of distinct event start times.
HopSize	**xs:int**	The number of distinct event start times to skip over to define the beginning of the next window.

2.2.3.2.3.22.2 WindowInputPolicyType Type

The **WindowInputPolicyType** type specifies how the ~~events'~~events' lifetimes are modified before being passed to a set-based operation.

The following code is the XML schema (XSD) for the **WindowInputPolicyType** type.

```
<xs:complexType name="WindowInputPolicyType">
  <xs:annotation>
    <xs:documentation>Specifies how to modify the temporal characteristics of
    events when they are passed to a time-sensitive user-defined
    operator/aggregate.</xs:documentation>
  </xs:annotation>
  <xs:choice>
    <xs:element name="Clip" type="tns:WindowInputPolicyClipType" />
  </xs:choice>
</xs:complexType>
```

The following table describes the element for the **WindowInputPolicyType** type.

Element	Type	Description
Clip	WindowInputPolicyClipType	Indicates the type of input clipping for the window.

2.2.3.2.3.22.2.1 WindowInputPolicyClipType Type

The **WindowInputPolicyClipType** type specifies how the events' lifetimes are clipped before they are input to the set-based operation.

The following code is the XML schema (XSD) for the **WindowInputPolicyClipType** type.

```
<xs:complexType name="WindowInputPolicyClipType">
  <xs:annotation>
    <xs:documentation>Specifies how to clip events that are input to a
    UDO/UDA with respect to the window boundaries. Events that are members
    of the window are not necessarily fully contained in the window. Hence,
    a clipping behavior on both window boundaries can be given.</xs:documentation>
  </xs:annotation>
  <xs:sequence />
  <xs:attribute name="Left" type="xs:boolean" use="required" />
  <xs:attribute name="Right" type="xs:boolean" use="required" />
</xs:complexType>
```

The following table describes the attributes for the **WindowInputPolicyClipType** type.

Attribute	Type	Description
Left	**xs:boolean**	True if the events' lifetimes ~~should~~are to be clipped to the window start time in case the event's start time falls outside of the window. False if the events' start times ~~should~~are not to be changed.
Right	**xs:boolean**	True if the events' lifetimes ~~should~~are to be clipped to the window end time in case the event's end time falls outside of the window. False if the events' end times ~~should~~are not to be changed.

2.2.3.2.3.22.3 WindowOutputPolicyType Type

The **WindowOutputPolicyType** type specifies how the ~~events'~~events' lifetimes are modified after the events are produced by a set-based operation and before they are being inserted back into the output event stream. The result events from a set-based operation can be left unaltered, clipped in certain ways with respect to the window, or completely adjusted with respect to the window.

The following code is the XML schema (XSD) for the **WindowOutputPolicyType** type.

```
<xs:complexType name="WindowOutputPolicyType">
  <xs:choice>
    <xs:element name="Unaltered" />
    <xs:element name="Clip" type="tns:WindowOutputPolicyClipType" />
    <xs:element name="Adjust" type="tns:WindowOutputPolicyAdjustType" />
  </xs:choice>
</xs:complexType>
```

The following table describes the elements for the **WindowOutputPolicyType** type.

Element	Type	Description
Unaltered	[empty element]	The set-based ~~operation's~~operation’s output will not be altered by either clipping or adjusting.
Clip	WindowOutputPolicyClipType	The set-based ~~operation's~~operation’s output will be clipped as specified.
Adjust	WindowOutputPolicyAdjustType	The set-based ~~operation's~~operation’s output will be adjusted as specified.

2.2.3.2.3.22.3.1 WindowOutputPolicyClipType

The **WindowOutputPolicyClipType** type specifies the possible types of clipping of the output from a set-based operation.

The following code is the XML schema (XSD) for the **WindowOutputPolicyClipType** type.

```
<xs:complexType name="WindowOutputPolicyClipType">
  <xs:sequence />
  <xs:attribute name="Type" type="tns:WindowOutputPolicyClipEnumType"
                use="required" />
</xs:complexType>
```

The following table describes the attribute for the **WindowOutputPolicyClipType** type.

Attribute	Type	Description
Type	**base=xs:string**	The **WindowOutputPolicyClipEnumType** enumeration indicates the type of clipping that can be applied to the ~~events'~~events’ lifetimes produced by the set-based operation. The enumeration values are as follows: ▪ **WindowEnd**: The ~~events'~~events’ lifetimes will be clipped at the window end. ▪ **Hop**: The ~~events'~~events’ lifetimes will be clipped at the hop size.

2.2.3.2.3.22.3.2 WindowOutputPolicyAdjustType

The **WindowOutputPolicyAdjustType** type specifies how the output from a set-based operation is to be adjusted with respect to the window.

The following code is the XML schema (XSD) for the **WindowOutputPolicyAdjustType** type.

```
<xs:complexType name="WindowOutputPolicyAdjustType">
  <xs:sequence />
  <xs:attribute name="Lifetime"
                type="tns:WindowOutputPolicyAdjustLifetimeEnumType"
                use="required" />
  <xs:attribute name="Alignment"
                type="tns:WindowOutputPolicyAdjustAlignmentEnumType"
                use="required" />
</xs:complexType>
```

The following table describes the attributes for the **WindowOutputPolicyAdjustType** type.

Attribute	Type	Description
Lifetime	**base=xs:string**	**Lifetime** is a value from an enumeration that indicates the manner of adjustment that can be applied to the ~~events'~~events' lifetimes produced by the set-based operation. The **WindowOutputPolicyAdjustLifetimeEnumType** enumeration values are as follows: ▪ **WindowSize**: **Lifetime** is set equal to the window size. ▪ **HopSize**: **Lifetime** is set equal to the hop size. ▪ **Point**: **Lifetime** is set to single point in time.
Alignment	**base=xs:string**	**Alignment** is a value from an enumeration that indicates the manner of alignment that can be applied to the ~~events'~~events' lifetimes produced by the windowed set-based operation with respect to the window.The **WindowOutputPolicyAdjustAlignmentEnumType** enumeration values are as follows: ▪ **WindowStart**: Events are aligned at window start time. ▪ **WindowEnd**: Events are aligned at window end time. ▪ **Hop**: Events are aligned at the hop size.

2.2.3.3 Diagnostic Method Types

This section contains the types that are used by the diagnostic methods.

2.2.3.3.1 SetDiagnosticSettings

The following code is the XML schema (XSD) for the **SetDiagnosticSettings** element, which contains the in-line complex type definition.

```
<xs:element name="SetDiagnosticSettings">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" name="DiagnosticAspects"
                  type="tns:DiagnosticAspects" />
      <xs:element minOccurs="0" name="DiagnosticLevel"
                  type="tns:DiagnosticLevel" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

The following table lists and describes the attributes for the **SetDiagnosticSettings** element.

Attribute	Type	Description
DiagnosticAspects	DiagnosticAspects	An enumeration value that indicates which diagnostic aspects will be included in a diagnostic view, performance counters, tracing, or event tracing.

Attribute	Type	Description
DiagnosticLevel	DiagnosticLevel	An enumeration value that indicates what level of verbosity of diagnostic events will be included in the diagnostic output.

2.2.3.3.1.1 DiagnosticAspects

The **DiagnosticAspects** type is an enumeration of the possible diagnostic aspects that can be enabled.

The following code is the XML schema (XSD) for the **DiagnosticAspects** type.

```
<xs:simpleType name="DiagnosticAspects">
  <xs:list>
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="None">
          <xs:annotation>
            <xs:appinfo>
              <EnumerationValue
                xmlns="http://schemas.microsoft.com/2003/10/Serialization/">
              0</EnumerationValue>
            </xs:appinfo>
          </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="DiagnosticViews">
          <xs:annotation>
            <xs:appinfo>
              <EnumerationValue
                xmlns="http://schemas.microsoft.com/2003/10/Serialization/">
              1</EnumerationValue>
            </xs:appinfo>
          </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="Debug" />
        <xs:enumeration value="StateChanges">
          <xs:annotation>
            <xs:appinfo>
              <EnumerationValue
                xmlns="http://schemas.microsoft.com/2003/10/Serialization/">
              16</EnumerationValue>
            </xs:appinfo>
          </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="CepEventTracing">
          <xs:annotation>
            <xs:appinfo>
              <EnumerationValue
                xmlns="http://schemas.microsoft.com/2003/10/Serialization/">
              64</EnumerationValue>
            </xs:appinfo>
          </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="GenerateErrorReports">
          <xs:annotation>
            <xs:appinfo>
              <EnumerationValue
                xmlns="http://schemas.microsoft.com/2003/10/Serialization/">
              128</EnumerationValue>
            </xs:appinfo>
          </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="PerformanceCounters">
          <xs:annotation>
```

```
              <xs:appinfo>
                <EnumerationValue
                  xmlns="http://schemas.microsoft.com/2003/10/Serialization/">
                256</EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Admin">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
                  xmlns="http://schemas.microsoft.com/2003/10/Serialization/">
                8</EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
        </xs:restriction>
      </xs:simpleType>
    </xs:list>
  </xs:simpleType  <xs:simpleType name="DiagnosticAspects">
    <xs:list>
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="None">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
                  xmlns="http://schemas.microsoft.com/2003/10/Serialization/">0
                </EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Debug">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
                  xmlns="http://schemas.microsoft.com/2003/10/Serialization/">1
                </EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="DiagnosticViews">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
                  xmlns="http://schemas.microsoft.com/2003/10/Serialization/">2
                </EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="PerformanceCounters">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
                  xmlns="http://schemas.microsoft.com/2003/10/Serialization/">4
                </EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="EndToEndTracing">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
                  xmlns="http://schemas.microsoft.com/2003/10/Serialization/">8
                </EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="CepEventTracing">
            <xs:annotation>
```

```
              <xs:appinfo>
                <EnumerationValue
                  xmlns="http://schemas.microsoft.com/2003/10/Serialization/">16
                </EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="StateChanges">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
                  xmlns="http://schemas.microsoft.com/2003/10/Serialization/">32
                </EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Memory">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
                  xmlns="http://schemas.microsoft.com/2003/10/Serialization/">64
                </EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="GenerateErrorReports">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
                  xmlns="http://schemas.microsoft.com/2003/10/Serialization/">128
                </EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
        </xs:restriction>
      </xs:simpleType>
    </xs:list>
  </xs:simpleType>
```

The following table describes the enumeration values for the **DiagnosticAspects** type.

Value	Description
None	A value specifying that no diagnostic aspects are included.
Debug	A value specifying that debug tracing is enabled.
DiagnosticViews	A value specifying that diagnostic view data collection is enabled.
PerformanceCounters	A value specifying that publishing performance counter data is enabled.
Admin	A value specifying that publishing administrative event data is enabled.
EndToEndTracing	A value specifying that event tracing for windows is enabled. For more information, see [MSDN-IDPTETW].
CepEventTracing	A value that enables tracing of complex event processing (CEP) events as they flow through a query. This tracing is used by the Event Flow Debugger.
StateChanges	A value that enables trace events that are emitted when CEP objects are created or destroyed, or when they change their state.
Memory	A value that enables tracing of the memory management subsystem.
GenerateErrorReports	A value specifying that error reports will be generated whenever a query crashes.

2.2.3.3.1.2 DiagnosticLevel

The **DiagnosticLevel** type is an enumeration of the different diagnostic level values. The level specifies the level of verbosity for the enabled diagnostic aspects.

The following code is the XML schema (XSD) for the **DiagnosticLevel** type.

```
<xs:simpleType name="DiagnosticLevel">
  <xs:annotation>
    <xs:appinfo>
      <ActualType Name="unsignedByte" Namespace="http://www.w3.org/2001/XMLSchema"
            xmlns="http://schemas.microsoft.com/2003/10/Serialization/" />
    </xs:appinfo>
  </xs:annotation>
  <xs:restriction base="xs:string">
    <xs:enumeration value="Always" />
    <xs:enumeration value="Critical" />
    <xs:enumeration value="Error" />
    <xs:enumeration value="Warning" />
    <xs:enumeration value="Informational" />
    <xs:enumeration value="Verbose" />
  </xs:restriction>
</xs:simpleType>
```

The following table describes the enumeration values for the **DiagnosticLevel** type. The values in this table are ordered from least verbose to most verbose.

Value	Description
Always	A value that denotes the least verbose level. Only events that are always emitted are included.
Critical	A value specifying that events with the Critical level are included.
Error	A value specifying that error events are included.
Warning	A value specifying that warning events are included
Informational	A value specifying that informational events are included.
Verbose	A value that denotes the most verbose level. All of the preceding values are included.

2.2.3.3.2 GetDiagnosticSettingsResponse

The **GetDiagnosticSettingsResponse** type contains the current values for diagnostic settings that are in effect.

The following code is the XML schema (XSD) for the **GetDiagnosticSettingsResponse** type.

```
<xs:element name="GetDiagnosticSettingsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" name="DiagnosticAspects"
            type="tns:DiagnosticAspects" />
      <xs:element minOccurs="0" name="DiagnosticLevel"
            type="tns:DiagnosticLevel" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

The following table describes the elements for the **GetDiagnosticSettingsResponse** type.

Element	Type	Description
DiagnosticAspects	DiagnosticAspects	An enumeration value that indicates which diagnostic aspects will be included in the DiagnosticView statistics.
DiagnosticLevel	DiagnosticLevel	An enumeration value that indicates what level of criticality of diagnostic events will be included in the Diagnostic View statistics.

2.2.3.3.3 GetDiagnosticViewResponse

The **GetDiagnosticViewResponse** type contains a report of the accumulated statistics for a Diagnostic View.

The following code is the XML schema (XSD) for the **GetDiagnosticViewResponse** type.

```
  <xs:element name="GetDiagnosticViewResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" name="View" nillable="true"
                    type="tns:DiagnosticView" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
```

The following table describes the element for the **GetDiagnosticViewResponse** type.

Element	Type	Description
View	DiagnosticView	This element contains the Diagnostic View results that were requested.

2.2.3.3.3.1 DiagnosticView

The **DiagnosticView** type contains the contents of the statistical results for a diagnostic view.

The following code is the XML schema (XSD) for the **DiagnosticView** type.

```
  <xs:complexType name="DiagnosticView">
    <xs:sequence>
      <xs:element name="Name" nillable="true" type="xs:anyURI" />
      <xs:element minOccurs="0" name="Properties" nillable="true"
                  type="tns:Properties" />
    </xs:sequence>
  </xs:complexType>
```

The following table describes the elements for the DiagnosticView type.

Element	Type	Description
Name	**xs:anyURI**	This URI represents the name of the object for which the diagnostic view is being returned.
Properties	Properties	The **Properties** element contains a collection of **Property** objects, each of which contains a single name-value pair from the view.

2.2.3.3.3.1.1 Properties

The **Properties** type contains a collection of **Property** elements, each of which is a name/value pair that contains a single value that is part of the diagnostic view.

The following code is the XML schema (XSD) for the **Properties** type.

```
<xs:complexType name="Properties">
  <xs:annotation>
    <xs:appinfo>
      <IsDictionary xmlns="http://schemas.microsoft.com/2003/10/Serialization/">
        true</IsDictionary>
    </xs:appinfo>
  </xs:annotation>
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="Property">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Name" nillable="true" type="xs:string" />
          <xs:element name="Value" nillable="true" type="xs:anyType" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

The following table describes the elements for the **Properties** type.

Element	Type	Description
Property	**complexType**	This XML element contains a single name/value pair, which together form a statistical result being returned in the Diagnostic View.
Name	**xs:string**	The name of the observed statistical value that is being returned.
Value	**xs:anyType**	The value for the statistical value that is being returned.

2.2.3.4 Fault Types

This section contains the definitions for the fault types.

2.2.3.4.1 InvalidNameFault

This complex type defines the type for the **s:Detail** child element of the **s:Fault** element in the SOAP fault body.

The following code is the XML schema (XSD) for the **InvalidNameFault** complex type.

```
<xs:complexType name="InvalidNameFault">
  <xs:sequence>
    <xs:element minOccurs="0" name="Message" nillable="true" type="xs:string" />
  </xs:sequence>
</xs:complexType>
```

The following table describes the element for the **InvalidNameFault** complex type.

Element	Type	Description
Message	**xs:string**	A descriptive message for the fault.

2.2.3.4.2 InvalidDefinitionFault

This complex type defines the type for the **s:Detail** child element of the **s:Fault** element in the SOAP fault body.

The following code is the XML schema (XSD) for the **InvalidDefinitionFault** complex type.

```
<xs:complexType name="InvalidDefinitionFault">
  <xs:sequence>
    <xs:element minOccurs="0" name="Message" nillable="true" type="xs:string" />
  </xs:sequence>
</xs:complexType>
```

The following table describes the element for the **InvalidDefinitionFault** complex type.

Element	Type	Description
Message	**xs:string**	A descriptive message for the fault.

2.2.3.4.3 ManagementFault

This complex type defines the type for the **s:Detail** child element of the **s:Fault** element in the SOAP fault body.

The following code is the XML schema (XSD) for the **ManagementFault** complex type.

```
<xs:complexType name="ManagementFault">
  <xs:sequence>
    <xs:element minOccurs="0" name="Message" nillable="true" type="xs:string" />
  </xs:sequence>
</xs:complexType>
```

The following table describes the element for the **ManagementFault** complex type.

Element	Type	Description
Message	**xs:string**	A descriptive message for the fault.

2.2.3.4.4 RuntimeFault

This complex type defines the type for the **s:Detail** child element of the **s:Fault** element in the SOAP fault body.

The following code is the XML schema (XSD) for the **RuntimeFault** complex type.

```
<xs:complexType name="RuntimeFault">
  <xs:sequence>
    <xs:element minOccurs="0" name="Message" nillable="true" type="xs:string" />
  </xs:sequence>
</xs:complexType>
```

The following table describes the element for the **RuntimeFault** complex type.

Element	Type	Description
Message	**xs:string**	A descriptive message for the fault.

2.2.3.4.5 GetDiagnosticSettingsNotSupported

This complex type defines the type for the **s:Detail** child element of the **s:Fault** element in the SOAP fault body.

The following code is the XML schema (XSD) for the **GetDiagnosticSettingsNotSupported** complex type.

```
<xs:complexType name="GetDiagnosticSettingsNotSupported">
  <xs:sequence>
    <xs:element minOccurs="0" name="Message" nillable="true" type="xs:string" />
    <xs:element minOccurs="0" name="Name" nillable="true" type="xs:anyURI" />
  </xs:sequence>
</xs:complexType>
```

The following table describes the elements for the **GetDiagnosticSettingsNotSupported** complex type.

Element	Type	Description
Message	**xs:string**	A descriptive message for a particular fault (error).
Name	**xs:anyURI**	The URI of the CEP metadata object that caused a particular fault.

2.2.3.4.6 ClearDiagnosticSettingsNotSupported

This complex type defines the type for the **s:Detail** child element of the **s:Fault** element in the SOAP fault body.

The following code is the XML schema (XSD) for the **ClearDiagnosticSettingsNotSupported** complex type.

```
<xs:complexType name="ClearDiagnosticSettingsNotSupported">
  <xs:sequence>
    <xs:element minOccurs="0" name="Message" nillable="true"
                type="xs:string" />
    <xs:element minOccurs="0" name="Name" nillable="true"
                type="xs:anyURI" />
  </xs:sequence>
</xs:complexType>
```

The following table describes the elements for the **ClearDiagnosticSettingsNotSupported** complex type.

Element	Type	Description
Message	**xs:string**	A descriptive message for a particular fault (error).
Name	**xs:anyURI**	The URI of the CEP metadata object that caused a particular fault.

2.2.3.4.7 GetDiagnosticViewNotSupported

This complex type defines the type for the **s:Detail** child element of the **s:Fault** element in the SOAP fault body.

The following code is the XML schema (XSD) for the **GetDiagnosticViewNotSupported** complex type.

```
<xs:complexType name="GetDiagnosticViewNotSupported">
  <xs:sequence>
    <xs:element minOccurs="0" name="Message" nillable="true" type="xs:string" />
    <xs:element minOccurs="0" name="Name" nillable="true" type="xs:anyURI" />
  </xs:sequence>
</xs:complexType>
```

The following table describes the elements for the **GetDiagnosticViewNotSupported** complex type.

Element	Type	Description
Message	**xs:string**	A descriptive message for a particular fault (error).
Name	**xs:anyURI**	The URI of the CEP metadata object that caused a particular fault.

2.2.3.5 Type Description Types

2.2.3.5.1 TypeRoot

The **TypeRoot** complex type defines the type of a root element for an XML document that describes a CLR type. The description includes public instance properties and fields of the CLR type. This recursively includes descriptions of those field and property types.

The following code is the XML schema (XSD) for the **TypeRoot** complex type.

```
<xs:complexType name="TypeRoot">
  <xs:sequence>
    <xs:element name="Types" type="linqexpression:ArrayOfType"/>
    <xs:element name="Assemblies" type="linqexpression:ArrayOfAssembly"/>
    <xs:element name="Definition" type="linqexpression:TypeReference"/>
  </xs:sequence>
</xs:complexType>
```

The following table describes the child elements for the **TypeRoot** complex type.

Element	Type	Description
Types	**ArrayOfType**	An array of CLR type descriptions that are referenced directly or indirectly by the type **Definition**. Any **AssemblyReference** that appears in a type description MUST have an ID that matches the ID of an **Assembly** element under the **Assemblies** element.
Assemblies	**ArrayOfAssembly**	Descriptions of the assemblies that are referenced in the **Types** array.
Definition	**TypeReference**	The definition for the type that is being described by this **TypeRoot** element.

2.2.3.5.2 Type

The **Type** complex provides the description of a CLR type.

The following code is the XML schema (XSD) for the **Type** complex type.

```
<xs:complexType name="Type">
  <xs:complexContent mixed="false">
    <xs:extension base="linqexpression:Type">
      <xs:sequence>
        <xs:element minOccurs="0" name="Properties" type="tns0:ArrayOfMember"/>
        <xs:element minOccurs="0" name="Fields" type="tns0:ArrayOfMember"/>
        <xs:element name="IsValueType" type="xs:boolean"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The following table describes the child elements for the **Type** complex type.

Element	Type	Description
Properties	ArrayOfMember	An array of public instance properties of the type that excludes indexers.
Fields	**ArrayOfMember**	An array of public instance fields of the type.
IsValueType	**boolean**	Indicates whether **Type** is a value type.

2.2.3.5.3 ArrayOfMember

The **ArrayOfMember** complex type defines a sequence of **ClrTypeMember** objects.

The following code is the XML schema (XSD) for the **ArrayOfMember** complex type.

```
<xs:complexType name="ArrayOfMember">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="Member" type="tns0:Member"/>
  </xs:sequence>
</xs:complexType>
```

Element	Type	Description
Member	Member	An element of the sequence.

2.2.3.5.4 Member

The **Member** complex type provides a description of a property or field of a CLR type.

The following code is the XML schema (XSD) for the **Member** complex type.

```
<xs:complexType name="Member">
  <xs:sequence>
    <xs:element name="Name" type="xs:string"/>
    <xs:element name="Type" type="linqexpression:TypeReference"/>
  </xs:sequence>
</xs:complexType>
```

The following table describes the child elements for the **Member** complex type.

Element	Type	Description
Name	**xs:string**	The name of the property or field.
Type	**TypeReference**	The type of the property or field.

2.2.3.6 Management Service Types

2.2.3.6.1 TypeIdentifier

The **TypeIdentifier** type identifies a CLR type by its name and assembly.

The following code is the XML schema (XSD) for the **TypeIdentifier** complex type.

```
  <xs:complexType name="TypeIdentifier">
    <xs:sequence>
      <xs:element minOccurs="0" name="Name" nillable="true" type="xs:string" />
      <xs:element minOccurs="0" name="AssemblyName" nillable="true" type="xs:string" />
    </xs:sequence>
  </xs:complexType>
```

The following table describes the child elements for the **TypeIdentifier** complex type.

Element	Type	Description
Name	**xs:string**	The name of the type.

Element	Type	Description
AssemblyName	**xs:string**	The full name of the assembly that contains the type (see [MSDN-AsmblyNames]).

2.2.3.6.2 ArrayOfTypeIdentifier

The **ArrayOfTypeIdentifier** complex type identifies a collection of TypeIdentifier elements.

The following code is the XML schema (XSD) for the **ArrayOfTypeIdentifier** complex type.

```
    <xs:complexType name="ArrayOfTypeIdentifier">
      <xs:sequence>
        <xs:element minOccurs="0" maxOccurs="unbounded" name="TypeIdentifier" nillable="true" 
  type="tns:TypeIdentifier" />
      </xs:sequence>
    </xs:complexType>
```

The following table describes the child elements for the **ArrayOfTypeIdentifier** complex type.

Element	Type	Description
Name	**TypeIdentifier**	A type in the collection

2.2.4 SOAP Headers

The following table summarizes the set of SOAP header definitions that are defined by this protocol specification.

Header	Description
CreateRequest_Headers	SOAP headers for the CreateRequest message. For more information, see section 2.2.2.1.1.1.1.
CreateResponse_Headers	SOAP headers for the CreateResponse message. For more information, see section 2.2.2.1.1.2.1.
GetRequest_Headers	SOAP headers for the GetRequest message. For more information, see section 2.2.2.1.2.1.1.
DeleteRequest_Headers	SOAP headers for the DeleteRequest message. For more information, see section 2.2.2.1.3.1.1.
DeleteResponse_Headers	SOAP headers for the DeleteResponse message. For more information, see section 2.2.2.1.3.2.1.
EnumerateRequest_Headers	SOAP headers for the EnumerateRequest message. For more information, see section 2.2.2.1.4.1.1.
ChangeQueryStateRequest_Headers	SOAP headers for the ChangeQueryStateRequest message. For more information, see section 2.2.2.1.5.1.1.
ChangeQueryStateResponse_Headers	SOAP headers for the ChangeQueryStateResponse message. For more information, see section 2.2.2.1.5.2.1.

Header	Description
GetDiagnosticSettingsRequest_Headers	SOAP headers for the GetDiagnosticSettingsRequest message. For more information, see section 2.2.2.2.1.1.1.
SetDiagnosticSettingsRequest_Headers	SOAP headers for the SetDiagnosticSettingsResponse message. For more information, see section 2.2.2.2.2.1.1.
ClearDiagnosticSettingsRequest_Headers	SOAP headers for the ClearDiagnosticSettingsRequest message. For more information, see section 2.2.2.2.3.1.1.
GetDiagnosticViewRequest_Headers	SOAP headers for the GetDiagnosticViewRequest message. For more information, see section 2.2.2.2.4.1.1.

3 Appendix A: Full WSDL

WSDL or schema name	Prefix	Section
Complex Event Processing (CEP) Management WSDL	wsdl:	3.1
Complex Event Processing Management Schema	xs:	3.2
Complex Event Processing Metadata Schema	xs:	3.3
W3C Addressing Schema	xs:	3.4
Serialization Schema	xs:	3.5
Serialization Arrays Schema	xs:	3.6
Type Design Schema	xs:	3.7
Management Service Schema	xs:	3.8

For ease of implementation, the full WSDLs and schemas are provided in the following sections.

3.1 Complex Event Processing Management WSDL

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-
1.0.xsd" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
xmlns:tns="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
xmlns:wsap="http://schemas.xmlsoap.org/ws/2004/08/addressing/policy"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:msc="http://schemas.microsoft.com/ws/2005/12/wsdl/contract"
xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:wsa10="http://www.w3.org/2005/08/addressing"
xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
targetNamespace="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <xsd:schema
targetNamespace="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Impor
ts">
      <xsd:import
namespace="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management" />
      <xsd:import namespace="http://schemas.microsoft.com/2003/10/Serialization/" />
      <xsd:import
namespace="http://schemas.microsoft.com/ComplexEventProcessing/2010/01/Metadata" />
      <xsd:import namespace="http://www.w3.org/2005/08/addressing" />
      <xsd:import namespace="http://schemas.microsoft.com/2003/10/Serialization/Arrays" />
      <xsd:import
namespace="http://schemas.microsoft.com/ComplexEventProcessing/2011/10/LinqExpression" />
      <xsd:import
namespace="http://schemas.microsoft.com/ComplexEventProcessing/2011/10/Design" />
      <xsd:import
namespace="http://schemas.datacontract.org/2004/07/Microsoft.ComplexEventProcessing.Managemen
tService" />
    </xsd:schema>
  </wsdl:types>
  <wsdl:message name="CreateRequest">
    <wsdl:part name="CreateRequest" element="tns:CreateRequest" />
  </wsdl:message>
  <wsdl:message name="CreateRequest_Headers">
    <wsdl:part name="Name" element="tns:Name" />
  </wsdl:message>
  <wsdl:message name="CreateResponse" />
  <wsdl:message name="CreateResponse_Headers">
    <wsdl:part name="ResourceAddress" element="tns:ResourceAddress" />
  </wsdl:message>
  <wsdl:message name="IManagementService_Create_InvalidNameFaultFault_FaultMessage">
    <wsdl:part name="detail" element="tns:InvalidNameFault" />
  </wsdl:message>
  <wsdl:message name="IManagementService_Create_InvalidDefinitionFaultFault_FaultMessage">
    <wsdl:part name="detail" element="tns:InvalidDefinitionFault" />
  </wsdl:message>
  <wsdl:message name="GetRequest" />
  <wsdl:message name="GetRequest_Headers">
    <wsdl:part name="Name" element="tns:Name" />
  </wsdl:message>
  <wsdl:message name="GetResponse">
    <wsdl:part name="GetResponse" element="tns:GetResponse" />
  </wsdl:message>
  <wsdl:message name="IManagementService_Get_InvalidNameFaultFault_FaultMessage">
    <wsdl:part name="detail" element="tns:InvalidNameFault" />
  </wsdl:message>
  <wsdl:message name="DeleteRequest" />
  <wsdl:message name="DeleteRequest_Headers">
    <wsdl:part name="Name" element="tns:Name" />
  </wsdl:message>
  <wsdl:message name="DeleteResponse" />
  <wsdl:message name="DeleteResponse_Headers">
    <wsdl:part name="Name" element="tns:Name" />
  </wsdl:message>
  <wsdl:message name="IManagementService_Delete_InvalidNameFaultFault_FaultMessage">
    <wsdl:part name="detail" element="tns:InvalidNameFault" />
  </wsdl:message>
  <wsdl:message name="IManagementService_Delete_ManagementFaultFault_FaultMessage">
    <wsdl:part name="detail" element="tns:ManagementFault" />
  </wsdl:message>
  <wsdl:message name="EnumerateRequest" />
  <wsdl:message name="EnumerateRequest_Headers">
    <wsdl:part name="Name" element="tns:Name" />
  </wsdl:message>
  <wsdl:message name="EnumerateResponse">
    <wsdl:part name="ResourceNames" element="tns:ResourceNames" />
  </wsdl:message>
  <wsdl:message name="IManagementService_Enumerate_InvalidNameFaultFault_FaultMessage">
    <wsdl:part name="detail" element="tns:InvalidNameFault" />
  </wsdl:message>
  <wsdl:message name="ChangeQueryStateRequest">
    <wsdl:part name="QueryState" element="tns:QueryState" />
  </wsdl:message>
  <wsdl:message name="ChangeQueryStateRequest_Headers">
    <wsdl:part name="Name" element="tns:Name" />
  </wsdl:message>
  <wsdl:message name="ChangeQueryStateResponse">
    <wsdl:part name="QueryState" element="tns:QueryState" />
  </wsdl:message>
  <wsdl:message name="ChangeQueryStateResponse_Headers">
    <wsdl:part name="Name" element="tns:Name" />
  </wsdl:message>
  <wsdl:message 
name="IManagementService_ChangeQueryState_InvalidNameFaultFault_FaultMessage">
    <wsdl:part name="detail" element="tns:InvalidNameFault" />
  </wsdl:message>
  <wsdl:message name="IManagementService_ChangeQueryState_RuntimeFaultFault_FaultMessage">
    <wsdl:part name="detail" element="tns:RuntimeFault" />
  </wsdl:message>
  <wsdl:message name="ResumeProcessRequest" />
  <wsdl:message name="ResumeProcessRequest_Headers">
    <wsdl:part name="Name" element="tns:Name" />
  </wsdl:message>
  <wsdl:message name="ResumeProcessResponse" />
  <wsdl:message name="ResumeProcessResponse_Headers">
```

```
      <wsdl:part name="Name" element="tns:Name" />
    </wsdl:message>
    <wsdl:message name="IManagementService_ResumeProcess_InvalidNameFaultFault_FaultMessage">
      <wsdl:part name="detail" element="tns:InvalidNameFault" />
    </wsdl:message>
    <wsdl:message name="IManagementService_ResumeProcess_ManagementFaultFault_FaultMessage">
      <wsdl:part name="detail" element="tns:ManagementFault" />
    </wsdl:message>
    <wsdl:message name="IManagementService_ResumeProcess_RuntimeFaultFault_FaultMessage">
      <wsdl:part name="detail" element="tns:RuntimeFault" />
    </wsdl:message>
    <wsdl:message name="GetDiagnosticSettingsRequest" />
    <wsdl:message name="GetDiagnosticSettingsRequest_Headers">
      <wsdl:part name="Name" element="tns:Name" />
    </wsdl:message>
    <wsdl:message name="GetDiagnosticSettingsResponse">
      <wsdl:part name="parameters" element="tns:GetDiagnosticSettingsResponse" />
    </wsdl:message>
    <wsdl:message 
name="IManagementService_GetDiagnosticSettings_InvalidNameFaultFault_FaultMessage">
      <wsdl:part name="detail" element="tns:InvalidNameFault" />
    </wsdl:message>
    <wsdl:message 
name="IManagementService_GetDiagnosticSettings_GetDiagnosticSettingsNotSupportedFaultFault_Fa
ultMessage">
      <wsdl:part name="detail" element="tns:GetDiagnosticSettingsNotSupported" />
    </wsdl:message>
    <wsdl:message name="SetDiagnosticSettingsRequest">
      <wsdl:part name="parameters" element="tns:SetDiagnosticSettings" />
    </wsdl:message>
    <wsdl:message name="SetDiagnosticSettingsRequest_Headers">
      <wsdl:part name="Name" element="tns:Name" />
    </wsdl:message>
    <wsdl:message name="IManagementService_SetDiagnosticSettings_OutputMessage" />
    <wsdl:message 
name="IManagementService_SetDiagnosticSettings_InvalidNameFaultFault_FaultMessage">
      <wsdl:part name="detail" element="tns:InvalidNameFault" />
    </wsdl:message>
    <wsdl:message 
name="IManagementService_SetDiagnosticSettings_SetDiagnosticSettingsNotSupportedFaultFault_Fa
ultMessage">
      <wsdl:part name="detail" element="tns:GetDiagnosticSettingsNotSupported" />
    </wsdl:message>
    <wsdl:message name="ClearDiagnosticSettingsRequest" />
    <wsdl:message name="ClearDiagnosticSettingsRequest_Headers">
      <wsdl:part name="Name" element="tns:Name" />
    </wsdl:message>
    <wsdl:message name="IManagementService_ClearDiagnosticSettings_OutputMessage" />
    <wsdl:message 
name="IManagementService_ClearDiagnosticSettings_InvalidNameFaultFault_FaultMessage">
      <wsdl:part name="detail" element="tns:InvalidNameFault" />
    </wsdl:message>
    <wsdl:message 
name="IManagementService_ClearDiagnosticSettings_ClearDiagnosticSettingsNotSupportedFaultFaul
t_FaultMessage">
      <wsdl:part name="detail" element="tns:ClearDiagnosticSettingsNotSupported" />
    </wsdl:message>
    <wsdl:message name="GetDiagnosticViewRequest" />
    <wsdl:message name="GetDiagnosticViewRequest_Headers">
      <wsdl:part name="Name" element="tns:Name" />
    </wsdl:message>
    <wsdl:message name="GetDiagnosticViewResponse">
      <wsdl:part name="parameters" element="tns:GetDiagnosticViewResponse" />
    </wsdl:message>
    <wsdl:message 
name="IManagementService_GetDiagnosticView_InvalidNameFaultFault_FaultMessage">
      <wsdl:part name="detail" element="tns:InvalidNameFault" />
    </wsdl:message>
```

```
  <wsdl:message
name="IManagementService_GetDiagnosticView_GetDiagnosticViewNotSupportedFaultFault_FaultMessa
ge">
    <wsdl:part name="detail" element="tns:GetDiagnosticViewNotSupported" />
  </wsdl:message>
  <wsdl:message name="CheckpointRequest" />
  <wsdl:message name="CheckpointRequest_Headers">
    <wsdl:part name="Name" element="tns:Name" />
  </wsdl:message>
  <wsdl:message name="CheckpointResponse">
    <wsdl:part name="parameters"
    element="tns:CheckpointResponse" />
  </wsdl:message>
  <wsdl:message name="IManagementService_Checkpoint_InvalidNameFaultFault_FaultMessage">
    <wsdl:part name="detail" element="tns:InvalidNameFault" />
  </wsdl:message>
  <wsdl:message name="IManagementService_Checkpoint_ManagementFaultFault_FaultMessage">
    <wsdl:part name="detail" element="tns:ManagementFault" />
  </wsdl:message>
  <wsdl:message name="CancelCheckpointRequest" />
  <wsdl:message name="CancelCheckpointRequest_Headers">
    <wsdl:part name="Name" element="tns:Name" />
  </wsdl:message>
  <wsdl:message name="IManagementService_CancelCheckpoint_OutputMessage" />
  <wsdl:message name="IManagementService_CancelCheckpoint_ManagementFaultFault_FaultMessage">
    <wsdl:part name="detail" element="tns:ManagementFault" />
  </wsdl:message>
  <wsdl:message
name="IManagementService_CancelCheckpoint_InvalidNameFaultFault_FaultMessage">
    <wsdl:part name="detail" element="tns:InvalidNameFault" />
  </wsdl:message>
  <wsdl:message name="GetEntityTypeRequest" />
  <wsdl:message name="GetEntityTypeRequest_Headers">
    <wsdl:part name="Name" element="tns:Name" />
  </wsdl:message>
  <wsdl:message
name="IManagementService_GetEntityExpressionType_InvalidNameFaultFault_FaultMessage">
    <wsdl:part name="detail" element="tns:InvalidNameFault" />
  </wsdl:message>
  <wsdl:message name="GetTypeDefinitionRequest">
    <wsdl:part name="TypeIdentifier" element="tns:TypeIdentifier" />
    <wsdl:part name="GenericArguments" element="tns:GenericArguments" />
  </wsdl:message>
  <wsdl:message
name="IManagementService_GetTypeDefinition_InvalidNameFaultFault_FaultMessage">
    <wsdl:part name="detail" element="tns:InvalidNameFault" />
  </wsdl:message>
  <wsdl:portType name="IManagementService">
    <wsdl:operation name="Create">
      <wsdl:input
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Create"
name="CreateRequest" message="tns:CreateRequest" />
      <wsdl:output
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/CreateRes
ponse" name="CreateResponse" message="tns:CreateResponse" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidNa
me" name="InvalidNameFaultFault"
message="tns:IManagementService_Create_InvalidNameFaultFault_FaultMessage" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidDe
finition" name="InvalidDefinitionFaultFault"
message="tns:IManagementService_Create_InvalidDefinitionFaultFault_FaultMessage" />
    </wsdl:operation>
    <wsdl:operation name="Get">
      <wsdl:input
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Get"
name="GetRequest" message="tns:GetRequest" />
```

```
      <wsdl:output
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetRespon
se" name="GetResponse" message="tns:GetResponse" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidNa
me" name="InvalidNameFaultFault"
message="tns:IManagementService_Get_InvalidNameFaultFault_FaultMessage" />
    </wsdl:operation>
    <wsdl:operation name="Delete">
      <wsdl:input
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Delete"
name="DeleteRequest" message="tns:DeleteRequest" />
      <wsdl:output
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/DeleteRes
ponse" name="DeleteResponse" message="tns:DeleteResponse" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidNa
me" name="InvalidNameFaultFault"
message="tns:IManagementService_Delete_InvalidNameFaultFault_FaultMessage" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Fault"
name="ManagementFaultFault"
message="tns:IManagementService_Delete_ManagementFaultFault_FaultMessage" />
    </wsdl:operation>
    <wsdl:operation name="Enumerate">
      <wsdl:input
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Enumerate
" name="EnumerateRequest" message="tns:EnumerateRequest" />
      <wsdl:output
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Enumerate
Response" name="EnumerateResponse" message="tns:EnumerateResponse" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidNa
me" name="InvalidNameFaultFault"
message="tns:IManagementService_Enumerate_InvalidNameFaultFault_FaultMessage" />
    </wsdl:operation>
    <wsdl:operation name="ChangeQueryState">
      <wsdl:input
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/ChangeQue
ryState" name="ChangeQueryStateRequest" message="tns:ChangeQueryStateRequest" />
      <wsdl:output
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/ChangeQue
ryStateResponse" name="ChangeQueryStateResponse" message="tns:ChangeQueryStateResponse" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidNa
me" name="InvalidNameFaultFault"
message="tns:IManagementService_ChangeQueryState_InvalidNameFaultFault_FaultMessage" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/RuntimeFa
ilure" name="RuntimeFaultFault"
message="tns:IManagementService_ChangeQueryState_RuntimeFaultFault_FaultMessage" />
    </wsdl:operation>
    <wsdl:operation name="ResumeProcess">
      <wsdl:input
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/ResumePro
cess" name="ResumeProcessRequest" message="tns:ResumeProcessRequest" />
      <wsdl:output
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/ResumePro
cessResponse" name="ResumeProcessResponse" message="tns:ResumeProcessResponse" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidNa
me" name="InvalidNameFaultFault"
message="tns:IManagementService_ResumeProcess_InvalidNameFaultFault_FaultMessage" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Fault"
name="ManagementFaultFault"
message="tns:IManagementService_ResumeProcess_ManagementFaultFault_FaultMessage" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/RuntimeFa
```

```
ilure" name="RuntimeFaultFault"
message="tns:IManagementService_ResumeProcess_RuntimeFaultFault_FaultMessage" />
    </wsdl:operation>
    <wsdl:operation name="GetDiagnosticSettings">
      <wsdl:input
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetDiagno
sticSettings" name="GetDiagnosticSettingsRequest" message="tns:GetDiagnosticSettingsRequest"
/>
      <wsdl:output
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetDiagno
sticSettingsResponse" name="GetDiagnosticSettingsResponse"
message="tns:GetDiagnosticSettingsResponse" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidNa
me" name="InvalidNameFaultFault"
message="tns:IManagementService_GetDiagnosticSettings_InvalidNameFaultFault_FaultMessage" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetDiagno
sticSettingsNotSupported" name="GetDiagnosticSettingsNotSupportedFaultFault"
message="tns:IManagementService_GetDiagnosticSettings_GetDiagnosticSettingsNotSupportedFaultF
ault_FaultMessage" />
    </wsdl:operation>
    <wsdl:operation name="SetDiagnosticSettings">
      <wsdl:input
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/SetDiagno
sticSettings" name="SetDiagnosticSettingsRequest" message="tns:SetDiagnosticSettingsRequest"
/>
      <wsdl:output
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/SetDiagno
sticSettingsResponse" message="tns:IManagementService_SetDiagnosticSettings_OutputMessage" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidNa
me" name="InvalidNameFaultFault"
message="tns:IManagementService_SetDiagnosticSettings_InvalidNameFaultFault_FaultMessage" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/SetDiagno
sticSettingsNotSupported" name="SetDiagnosticSettingsNotSupportedFaultFault"
message="tns:IManagementService_SetDiagnosticSettings_SetDiagnosticSettingsNotSupportedFaultF
ault_FaultMessage" />
    </wsdl:operation>
    <wsdl:operation name="ClearDiagnosticSettings">
      <wsdl:input
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/ClearDiag
nosticSettings" name="ClearDiagnosticSettingsRequest"
message="tns:ClearDiagnosticSettingsRequest" />
      <wsdl:output
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/ClearDiag
nosticSettingsResponse"
message="tns:IManagementService_ClearDiagnosticSettings_OutputMessage" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidNa
me" name="InvalidNameFaultFault"
message="tns:IManagementService_ClearDiagnosticSettings_InvalidNameFaultFault_FaultMessage"
/>
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/ClearDiag
nosticSettingsNotSupported" name="ClearDiagnosticSettingsNotSupportedFaultFault"
message="tns:IManagementService_ClearDiagnosticSettings_ClearDiagnosticSettingsNotSupportedFa
ultFault_FaultMessage" />
    </wsdl:operation>
    <wsdl:operation name="GetDiagnosticView">
      <wsdl:input
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetDiagno
sticView" name="GetDiagnosticViewRequest" message="tns:GetDiagnosticViewRequest" />
      <wsdl:output
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetDiagno
sticViewResponse" name="GetDiagnosticViewResponse" message="tns:GetDiagnosticViewResponse" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidNa
```

```
me" name="InvalidNameFaultFault"
message="tns:IManagementService_GetDiagnosticView_InvalidNameFaultFault_FaultMessage" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetDiagno
sticViewNotSupported" name="GetDiagnosticViewNotSupportedFaultFault"
message="tns:IManagementService_GetDiagnosticView_GetDiagnosticViewNotSupportedFaultFault_Fau
ltMessage" />
    </wsdl:operation>
    <wsdl:operation name="Checkpoint">
      <wsdl:input
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Checkpoin
t"
      name="CheckpointRequest" message="tns:CheckpointRequest" />
      <wsdl:output
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Checkpoin
tResponse"
      name="CheckpointResponse" message="tns:CheckpointResponse" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidNa
me"
      name="InvalidNameFaultFault"
      message="tns:IManagementService_Checkpoint_InvalidNameFaultFault_FaultMessage" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Fault"
      name="ManagementFaultFault"
      message="tns:IManagementService_Checkpoint_ManagementFaultFault_FaultMessage" />
    </wsdl:operation>
    <wsdl:operation name="CancelCheckpoint">
      <wsdl:input
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/CancelChe
ckpoint"
      name="CancelCheckpointRequest"
      message="tns:CancelCheckpointRequest" />
      <wsdl:output
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/CancelChe
ckpointResponse"
      message="tns:IManagementService_CancelCheckpoint_OutputMessage" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Fault"
      name="ManagementFaultFault"
      message="tns:IManagementService_CancelCheckpoint_ManagementFaultFault_FaultMessage" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidNa
me"
      name="InvalidNameFaultFault"
      message="tns:IManagementService_CancelCheckpoint_InvalidNameFaultFault_FaultMessage" />
    </wsdl:operation>
    <wsdl:operation name="GetEntityExpressionType">
      <wsdl:input
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetEntity
ExpressionType" name="GetEntityTypeRequest" message="tns:GetEntityTypeRequest" />
      <wsdl:output
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetEntity
ExpressionTypeResponse" name="GetResponse" message="tns:GetResponse" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidNa
me" name="InvalidNameFaultFault"
message="tns:IManagementService_GetEntityExpressionType_InvalidNameFaultFault_FaultMessage"
/>
    </wsdl:operation>
    <wsdl:operation name="GetTypeDefinition">
      <wsdl:input
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetTypeDe
finition" name="GetTypeDefinitionRequest" message="tns:GetTypeDefinitionRequest" />
      <wsdl:output
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetTypesR
esponse" name="GetResponse" message="tns:GetResponse" />
      <wsdl:fault
wsaw:Action="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/InvalidNa
```

```
me" name="InvalidNameFaultFault"
message="tns:IManagementService_GetTypeDefinition_InvalidNameFaultFault_FaultMessage" />
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="DefaultBinding_IManagementService" type="tns:IManagementService">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="Create">
      <soap:operation
soapAction="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Create"
style="document" />
      <wsdl:input name="CreateRequest">
        <soap:header message="tns:CreateRequest_Headers" part="Name" use="literal" />
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="CreateResponse">
        <soap:header message="tns:CreateResponse_Headers" part="ResourceAddress"
use="literal" />
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidNameFaultFault">
        <soap:fault name="InvalidNameFaultFault" use="literal" />
      </wsdl:fault>
      <wsdl:fault name="InvalidDefinitionFaultFault">
        <soap:fault name="InvalidDefinitionFaultFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="Get">
      <soap:operation
soapAction="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Get"
style="document" />
      <wsdl:input name="GetRequest">
        <soap:header message="tns:GetRequest_Headers" part="Name" use="literal" />
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="GetResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidNameFaultFault">
        <soap:fault name="InvalidNameFaultFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="Delete">
      <soap:operation
soapAction="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Delete"
style="document" />
      <wsdl:input name="DeleteRequest">
        <soap:header message="tns:DeleteRequest_Headers" part="Name" use="literal" />
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="DeleteResponse">
        <soap:header message="tns:DeleteResponse_Headers" part="Name" use="literal" />
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidNameFaultFault">
        <soap:fault name="InvalidNameFaultFault" use="literal" />
      </wsdl:fault>
      <wsdl:fault name="ManagementFaultFault">
        <soap:fault name="ManagementFaultFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="Enumerate">
      <soap:operation
soapAction="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Enumerate"
style="document" />
      <wsdl:input name="EnumerateRequest">
        <soap:header message="tns:EnumerateRequest_Headers" part="Name" use="literal" />
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="EnumerateResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidNameFaultFault">
        <soap:fault name="InvalidNameFaultFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="ChangeQueryState">
      <soap:operation 
soapAction="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/ChangeQuer
yState" style="document" />
      <wsdl:input name="ChangeQueryStateRequest">
        <soap:header message="tns:ChangeQueryStateRequest_Headers" part="Name" use="literal" 
/>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="ChangeQueryStateResponse">
        <soap:header message="tns:ChangeQueryStateResponse_Headers" part="Name" use="literal" 
/>
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidNameFaultFault">
        <soap:fault name="InvalidNameFaultFault" use="literal" />
      </wsdl:fault>
      <wsdl:fault name="RuntimeFaultFault">
        <soap:fault name="RuntimeFaultFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="ResumeProcess">
      <soap:operation 
soapAction="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/ResumeProc
ess" style="document" />
      <wsdl:input name="ResumeProcessRequest">
        <soap:header message="tns:ResumeProcessRequest_Headers" part="Name" use="literal" />
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="ResumeProcessResponse">
        <soap:header message="tns:ResumeProcessResponse_Headers" part="Name" use="literal" />
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidNameFaultFault">
        <soap:fault name="InvalidNameFaultFault" use="literal" />
      </wsdl:fault>
      <wsdl:fault name="ManagementFaultFault">
        <soap:fault name="ManagementFaultFault" use="literal" />
      </wsdl:fault>
      <wsdl:fault name="RuntimeFaultFault">
        <soap:fault name="RuntimeFaultFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="GetDiagnosticSettings">
      <soap:operation 
soapAction="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetDiagnos
ticSettings" style="document" />
      <wsdl:input name="GetDiagnosticSettingsRequest">
        <soap:header message="tns:GetDiagnosticSettingsRequest_Headers" part="Name" 
use="literal" />
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="GetDiagnosticSettingsResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidNameFaultFault">
        <soap:fault name="InvalidNameFaultFault" use="literal" />
      </wsdl:fault>
      <wsdl:fault name="GetDiagnosticSettingsNotSupportedFaultFault">
        <soap:fault name="GetDiagnosticSettingsNotSupportedFaultFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="SetDiagnosticSettings">
      <soap:operation
soapAction="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/SetDiagnos
ticSettings" style="document" />
      <wsdl:input name="SetDiagnosticSettingsRequest">
        <soap:header message="tns:SetDiagnosticSettingsRequest_Headers" part="Name"
use="literal" />
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidNameFaultFault">
        <soap:fault name="InvalidNameFaultFault" use="literal" />
      </wsdl:fault>
      <wsdl:fault name="SetDiagnosticSettingsNotSupportedFaultFault">
        <soap:fault name="SetDiagnosticSettingsNotSupportedFaultFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="ClearDiagnosticSettings">
      <soap:operation
soapAction="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/ClearDiagn
osticSettings" style="document" />
      <wsdl:input name="ClearDiagnosticSettingsRequest">
        <soap:header message="tns:ClearDiagnosticSettingsRequest_Headers" part="Name"
use="literal" />
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidNameFaultFault">
        <soap:fault name="InvalidNameFaultFault" use="literal" />
      </wsdl:fault>
      <wsdl:fault name="ClearDiagnosticSettingsNotSupportedFaultFault">
        <soap:fault name="ClearDiagnosticSettingsNotSupportedFaultFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="GetDiagnosticView">
      <soap:operation
soapAction="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetDiagnos
ticView" style="document" />
      <wsdl:input name="GetDiagnosticViewRequest">
        <soap:header message="tns:GetDiagnosticViewRequest_Headers" part="Name" use="literal"
/>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="GetDiagnosticViewResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidNameFaultFault">
        <soap:fault name="InvalidNameFaultFault" use="literal" />
      </wsdl:fault>
      <wsdl:fault name="GetDiagnosticViewNotSupportedFaultFault">
        <soap:fault name="GetDiagnosticViewNotSupportedFaultFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="Checkpoint">
      <soap:operation
soapAction="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/Checkpoint
"
      style="document" />
      <wsdl:input name="CheckpointRequest">
        <soap:header message="tns:CheckpointRequest_Headers"
        part="Name" use="literal" />
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="CheckpointResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidNameFaultFault">
```

```
        <soap:fault name="InvalidNameFaultFault" use="literal" />
      </wsdl:fault>
      <wsdl:fault name="ManagementFaultFault">
        <soap:fault name="ManagementFaultFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="CancelCheckpoint">
      <soap:operation
soapAction="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/CancelChec
kpoint"
      style="document" />
      <wsdl:input name="CancelCheckpointRequest">
        <soap:header message="tns:CancelCheckpointRequest_Headers"
        part="Name" use="literal" />
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="ManagementFaultFault">
        <soap:fault name="ManagementFaultFault" use="literal" />
      </wsdl:fault>
      <wsdl:fault name="InvalidNameFaultFault">
        <soap:fault name="InvalidNameFaultFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="GetEntityExpressionType">
      <soap:operation
soapAction="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetEntityE
xpressionType" style="document" />
      <wsdl:input name="GetEntityTypeRequest">
        <soap:header message="tns:GetEntityTypeRequest_Headers" part="Name" use="literal" />
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="GetResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidNameFaultFault">
        <soap:fault name="InvalidNameFaultFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="GetTypeDefinition">
      <soap:operation
soapAction="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/GetTypeDef
inition" style="document" />
      <wsdl:input name="GetTypeDefinitionRequest">
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="GetResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidNameFaultFault">
        <soap:fault name="InvalidNameFaultFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
  </wsdl:binding>
</wsdl:definitions>
```

3.2 Complex Event Processing Management Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:tns=http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management
xmlns:design="http://schemas.microsoft.com/ComplexEventProcessing/2011/10/Design"
xmlns:metadata="http://schemas.microsoft.com/ComplexEventProcessing/2010/01/Metadata"
elementFormDefault="qualified"
```

```
targetNamespace="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:annotation>
    <xs:documentation>(c) 2010 Microsoft Corporation.  All rights reserved. The following
schema for the management specification of the Microsoft Complex Event Processing (CEP)
platform is presented in XML format and is for informational purposes only. Microsoft
Corporation ("Microsoft") may have trademarks, copyrights, or other intellectual property
rights covering subject matter in the schema. Microsoft does not make any representation or
warranty regarding the schema or any product or item developed based on the schema. The
schema is provided to you on an AS IS basis. Microsoft disclaims all express, implied and
statutory warranties, including but not limited to the implied warranties of merchantability,
fitness for a particular purpose, and freedom from infringement. Without limiting the
generality of the foregoing, Microsoft does not make any warranty of any kind that any item
developed based on the schema, or any portion of the schema, will not infringe any copyright,
patent, trade secret, or other intellectual property right of any person or entity in any
country. It is your responsibility to seek licenses for such intellectual property rights
where appropriate. MICROSOFT SHALL NOT BE LIABLE FOR ANY DAMAGES OF ANY KIND ARISING OUT OF
OR IN CONNECTION WITH THE USE OF THE SCHEMA, INCLUDING WITHOUT LIMITATION, ANY DIRECT,
INDIRECT, INCIDENTAL, CONSEQUENTIAL (INCLUDING ANY LOST PROFITS), PUNITIVE OR SPECIAL
DAMAGES, WHETHER OR NOT MICROSOFT HAS BEEN ADVISED OF SUCH DAMAGES.</xs:documentation>
  </xs:annotation>
  <xs:import namespace="http://schemas.microsoft.com/ComplexEventProcessing/2010/01/Metadata"
/>
  <xs:import namespace="http://www.w3.org/2005/08/addressing" />
  <xs:import namespace="http://schemas.microsoft.com/2003/10/Serialization/Arrays" />
  <xs:import namespace="http://schemas.microsoft.com/2003/10/Serialization/" />
  <xs:complexType name="CreateRequest">
    <xs:choice>
      <xs:element minOccurs="1" maxOccurs="1" name="InputAdapter"
type="metadata:InputAdapterType" />
      <xs:element minOccurs="1" maxOccurs="1" name="OutputAdapter"
type="metadata:OutputAdapterType" />
      <xs:element minOccurs="1" maxOccurs="1" name="Application"
type="metadata:ApplicationType" />
      <xs:element minOccurs="1" maxOccurs="1" name="EventType" type="metadata:EventType" />
      <xs:element minOccurs="1" maxOccurs="1" name="Query" type="metadata:QueryType" />
      <xs:element minOccurs="1" maxOccurs="1" name="QueryTemplate"
type="metadata:QueryTemplateType"
      <xs:element minOccurs="1" maxOccurs="1" name="Entity" type="metadata:EntityType" />
 />
    </xs:choice>
  </xs:complexType>
  <xs:element name="CreateRequest" nillable="true" type="tns:CreateRequest" />
  <xs:element name="Name" nillable="true" type="xs:anyURI" />
  <xs:element name="ResourceAddress" nillable="true"
xmlns:q1="http://www.w3.org/2005/08/addressing" type="q1:EndpointReferenceType" />
  <xs:complexType name="InvalidNameFault">
    <xs:sequence>
      <xs:element minOccurs="0" name="Message" nillable="true" type="xs:string" />
    </xs:sequence>
  </xs:complexType>
  <xs:element name="InvalidNameFault" nillable="true" type="tns:InvalidNameFault" />
  <xs:complexType name="InvalidDefinitionFault">
    <xs:sequence>
      <xs:element minOccurs="0" name="Message" nillable="true" type="xs:string" />
    </xs:sequence>
  </xs:complexType>
  <xs:element name="InvalidDefinitionFault" nillable="true" type="tns:InvalidDefinitionFault"
/>
  <xs:complexType name="GetResponse">
    <xs:choice>
      <xs:element minOccurs="1" maxOccurs="1" name="InputAdapter"
type="metadata:InputAdapterType" />
      <xs:element minOccurs="1" maxOccurs="1" name="OutputAdapter"
type="metadata:OutputAdapterType" />
      <xs:element minOccurs="1" maxOccurs="1" name="Application"
type="metadata:ApplicationType" />
      <xs:element minOccurs="1" maxOccurs="1" name="EventType" type="metadata:EventType" />
      <xs:element minOccurs="1" maxOccurs="1" name="Query" type="metadata:QueryType" />
      <xs:element minOccurs="1" maxOccurs="1" name="QueryTemplate"
type="metadata:QueryTemplateType" />
      <xs:element minOccurs="1" maxOccurs="1" name="Entity" type="metadata:EntityType" />
      <xs:element minOccurs="1" maxOccurs="1" name="TypeRoot" type="design:TypeRoot" />
      <xs:element minOccurs="1" maxOccurs="1" name="NullObject">
        <xs:complexType>
          <xs:complexContent mixed="false">
            <xs:restriction base="xs:anyType" />
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
    </xs:choice>
  </xs:complexType>
  <xs:element name="GetResponse" nillable="true" type="tns:GetResponse" />
  <xs:complexType name="ManagementFault">
    <xs:sequence>
      <xs:element minOccurs="0" name="Message" nillable="true" type="xs:string" />
    </xs:sequence>
  </xs:complexType>
  <xs:element name="ManagementFault" nillable="true" type="tns:ManagementFault" />
  <xs:element name="ResourceNames" nillable="true"
xmlns:q2="http://schemas.microsoft.com/2003/10/Serialization/Arrays" type="q2:ArrayOfanyURI"
/>
  <xs:simpleType name="QueryState">
    <xs:restriction base="xs:string">
      <xs:enumeration
value="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/QueryStateStart
ed" />
      <xs:enumeration
value="http://schemas.microsoft.com/ComplexEventProcessing/2009/05/Management/QueryStateStopp
ed" />
    </xs:restriction>
  </xs:simpleType>
  <xs:element name="QueryState" nillable="true" type="tns:QueryState" />
  <xs:complexType name="RuntimeFault">
    <xs:sequence>
      <xs:element minOccurs="0" name="Message" nillable="true" type="xs:string" />
    </xs:sequence>
  </xs:complexType>
  <xs:element name="RuntimeFault" nillable="true" type="tns:RuntimeFault" />
  <xs:element name="GetDiagnosticSettingsResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" name="DiagnosticAspects" type="tns:DiagnosticAspects" />
        <xs:element minOccurs="0" name="DiagnosticLevel" type="tns:DiagnosticLevel" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:simpleType name="DiagnosticAspects">
    <xs:list>
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="None">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
xmlns="http://schemas.microsoft.com/2003/10/Serialization/">0</EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="DiagnosticViews">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
xmlns="http://schemas.microsoft.com/2003/10/Serialization/">1</EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Debug" />
          <xs:enumeration value="StateChanges">
```

```
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
xmlns="http://schemas.microsoft.com/2003/10/Serialization/">16</EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="CepEventTracing">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
xmlns="http://schemas.microsoft.com/2003/10/Serialization/">64</EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="GenerateErrorReports">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
xmlns="http://schemas.microsoft.com/2003/10/Serialization/">128</EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="PerformanceCounters">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
xmlns="http://schemas.microsoft.com/2003/10/Serialization/">
                256</EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Admin">
            <xs:annotation>
              <xs:appinfo>
                <EnumerationValue
xmlns="http://schemas.microsoft.com/2003/10/Serialization/">
                8</EnumerationValue>
              </xs:appinfo>
            </xs:annotation>
          </xs:enumeration>
        </xs:restriction>
      </xs:simpleType>
    </xs:list>
  </xs:simpleType>
  <xs:element name="DiagnosticAspects" nillable="true" type="tns:DiagnosticAspects" />
  <xs:simpleType name="DiagnosticLevel">
    <xs:annotation>
      <xs:appinfo>
        <ActualType Name="unsignedByte" Namespace="http://www.w3.org/2001/XMLSchema"
xmlns="http://schemas.microsoft.com/2003/10/Serialization/" />
      </xs:appinfo>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="Always" />
      <xs:enumeration value="Critical" />
      <xs:enumeration value="Error" />
      <xs:enumeration value="Warning" />
      <xs:enumeration value="Informational" />
      <xs:enumeration value="Verbose" />
    </xs:restriction>
  </xs:simpleType>
  <xs:element name="DiagnosticLevel" nillable="true" type="tns:DiagnosticLevel" />
  <xs:complexType name="GetDiagnosticSettingsNotSupported">
    <xs:sequence>
      <xs:element minOccurs="0" name="Message" nillable="true" type="xs:string" />
      <xs:element minOccurs="0" name="Name" nillable="true" type="xs:anyURI" />
    </xs:sequence>
  </xs:complexType>
  <xs:element name="GetDiagnosticSettingsNotSupported" nillable="true"
type="tns:GetDiagnosticSettingsNotSupported" />
  <xs:element name="SetDiagnosticSettings">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" name="DiagnosticAspects" type="tns:DiagnosticAspects" />
        <xs:element minOccurs="0" name="DiagnosticLevel" type="tns:DiagnosticLevel" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="ClearDiagnosticSettingsNotSupported">
    <xs:sequence>
      <xs:element minOccurs="0" name="Message" nillable="true" type="xs:string" />
      <xs:element minOccurs="0" name="Name" nillable="true" type="xs:anyURI" />
    </xs:sequence>
  </xs:complexType>
  <xs:element name="ClearDiagnosticSettingsNotSupported" nillable="true"
type="tns:ClearDiagnosticSettingsNotSupported" />
  <xs:element name="GetDiagnosticViewResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" name="View" nillable="true" type="tns:DiagnosticView" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="DiagnosticView">
    <xs:sequence>
      <xs:element name="Name" nillable="true" type="xs:anyURI" />
      <xs:element minOccurs="0" name="Properties" nillable="true" type="tns:Properties" />
    </xs:sequence>
  </xs:complexType>
  <xs:element name="DiagnosticView" nillable="true" type="tns:DiagnosticView" />
  <xs:complexType name="Properties">
    <xs:annotation>
      <xs:appinfo>
        <IsDictionary
xmlns="http://schemas.microsoft.com/2003/10/Serialization/">true</IsDictionary>
      </xs:appinfo>
    </xs:annotation>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="Property">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Name" nillable="true" type="xs:string" />
            <xs:element name="Value" nillable="true" type="xs:anyType" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="Properties" nillable="true" type="tns:Properties" />
  <xs:complexType name="GetDiagnosticViewNotSupported">
    <xs:sequence>
      <xs:element minOccurs="0" name="Message" nillable="true" type="xs:string" />
      <xs:element minOccurs="0" name="Name" nillable="true" type="xs:anyURI" />
    </xs:sequence>
  </xs:complexType>
  <xs:element name="GetDiagnosticViewNotSupported" nillable="true"
type="tns:GetDiagnosticViewNotSupported" />
  <xs:element name="CheckpointResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" name="Result"
        type="xs:boolean" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="TypeIdentifier" nillable="true"
xmlns:q3="http://schemas.datacontract.org/2004/07/Microsoft.ComplexEventProcessing.Management
Service" type="q3:TypeIdentifier" />
```

```
  <xs:element name="GenericArguments" nillable="true"
xmlns:q4="http://schemas.datacontract.org/2004/07/Microsoft.ComplexEventProcessing.Management
Service" type="q4:ArrayOfTypeIdentifier" />
</xs:schema>
```

3.3 Complex Event Processing Metadata Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:tns="http://schemas.microsoft.com/ComplexEventProcessing/2010/01/Metadata"
elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/ComplexEventProcessing/2010/01/Metadata"
xmlns:xs=http://www.w3.org/2001/XMLSchema
xmlns:linqs="http://schemas.microsoft.com/ComplexEventProcessing/2011/10/LinqExpression"
>
  <xs:annotation>
    <xs:documentation>(c) 2010 Microsoft Corporation.  All rights reserved. The following
schema for the metadata specification of the Microsoft Complex Event Processing (CEP)
platform is presented in XML format and is for informational purposes only. Microsoft
Corporation ("Microsoft") may have trademarks, copyrights, or other intellectual property
rights covering subject matter in the schema. Microsoft does not make any representation or
warranty regarding the schema or any product or item developed based on the schema. The
schema is provided to you on an AS IS basis. Microsoft disclaims all express, implied and
statutory warranties, including but not limited to the implied warranties of merchantability,
fitness for a particular purpose, and freedom from infringement. Without limiting the
generality of the foregoing, Microsoft does not make any warranty of any kind that any item
developed based on the schema, or any portion of the schema, will not infringe any copyright,
patent, trade secret, or other intellectual property right of any person or entity in any
country. It is your responsibility to seek licenses for such intellectual property rights
where appropriate. MICROSOFT SHALL NOT BE LIABLE FOR ANY DAMAGES OF ANY KIND ARISING OUT OF
OR IN CONNECTION WITH THE USE OF THE SCHEMA, INCLUDING WITHOUT LIMITATION, ANY DIRECT,
INDIRECT, INCIDENTAL, CONSEQUENTIAL (INCLUDING ANY LOST PROFITS), PUNITIVE OR SPECIAL
DAMAGES, WHETHER OR NOT MICROSOFT HAS BEEN ADVISED OF SUCH DAMAGES.</xs:documentation>
  </xs:annotation>
  <xs:import
namespace="http://schemas.microsoft.com/ComplexEventProcessing/2011/10/LinqExpression"
></xs:import>
  <xs:complexType name="ApplicationType">
    <xs:annotation>
      <xs:documentation>Application object.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="Name" type="xs:anyURI" use="required" />
  </xs:complexType>
  <xs:complexType name="EventType">
    <xs:annotation>
      <xs:documentation>Specification of a CEP type. Contains zero or more
fields.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="Field" type="tns:EventFieldType"
/>
    </xs:sequence>
    <xs:attribute name="Name" type="xs:string" use="optional" />
  </xs:complexType>
  <xs:attributeGroup name="TypeFacetAttributes">
    <xs:annotation>
      <xs:documentation>Type identifier and facets.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="Nullable" type="xs:boolean" use="required" />
    <xs:attribute name="Culture" type="xs:string" use="optional" />
    <xs:attribute name="MaxSize" type="xs:unsignedInt" use="optional">
      <xs:annotation>
        <xs:documentation>MaxSize is only applicable to string and byte array types. For
string, this is the number of characters, for byte array this is the number of
bytes.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="SizeFixed" type="xs:boolean" use="optional">
```

```
      <xs:annotation>
        <xs:documentation>SizeFixed is only applicable to string and byte array types. It 
denotes a field of a fixed size.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:attributeGroup>
  <xs:simpleType name="PrimitiveTypeIdentifier">
    <xs:annotation>
      <xs:documentation>List of all natively supported primitive types.</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="System.Boolean" />
      <xs:enumeration value="System.Char" />
      <xs:enumeration value="System.SByte" />
      <xs:enumeration value="System.Int16" />
      <xs:enumeration value="System.Int32" />
      <xs:enumeration value="System.Int64" />
      <xs:enumeration value="System.Byte" />
      <xs:enumeration value="System.UInt16" />
      <xs:enumeration value="System.UInt32" />
      <xs:enumeration value="System.UInt64" />
      <xs:enumeration value="System.Decimal" />
      <xs:enumeration value="System.Single" />
      <xs:enumeration value="System.Double" />
      <xs:enumeration value="System.Guid" />
      <xs:enumeration value="System.DateTime" />
      <xs:enumeration value="System.TimeSpan" />
      <xs:enumeration value="System.String" />
      <xs:enumeration value="System.Byte[]" />
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="EventFieldType">
    <xs:annotation>
      <xs:documentation>Field of an Event Type. Can be of atomic or composite 
type.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="Name" type="xs:anyURI" use="required" />
    <xs:attribute name="Type" type="tns:PrimitiveTypeIdentifier" use="required" />
    <xs:attributeGroup ref="tns:TypeFacetAttributes" />
  </xs:complexType>
  <xs:complexType name="AdapterBaseType">
    <xs:annotation>
      <xs:documentation>Adapter base type. The common attributes of input and output 
adapter.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="Name" type="xs:anyURI" use="required" />
    <xs:attribute name="FactoryClassName" type="xs:string" use="required" />
    <xs:attribute name="IsTyped" type="xs:boolean" />
    <xs:attribute name="Description" type="xs:string" use="optional" />
  </xs:complexType>
  <xs:complexType name="InputAdapterType">
    <xs:annotation>
      <xs:documentation>Input adapter.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:extension base="tns:AdapterBaseType" />
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="OutputAdapterType">
    <xs:annotation>
      <xs:documentation>Output adapter.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:extension base="tns:AdapterBaseType" />
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="CompareOptionsType">
    <xs:annotation>
      <xs:documentation>Represents a .NET CompareOptions object to use with CompareInfo as an 
element. Can be a parameter for a method call expression.</xs:documentation>
```

```
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:restriction base="tns:NullaryExpression">
          <xs:sequence />
          <xs:attribute name="Value" type="tns:CompareOptionsParameterEnumType" use="required"
/>
        </xs:restriction>
      </xs:complexContent>
    </xs:complexType>
    <xs:simpleType name="CompareOptionsParameterEnumType">
      <xs:annotation>
        <xs:documentation>List of all values for .Net CompareOptions.</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:enumeration value="None" />
        <xs:enumeration value="IgnoreCase" />
        <xs:enumeration value="IgnoreNonSpace" />
        <xs:enumeration value="IgnoreSymbols" />
        <xs:enumeration value="IgnoreKanaType" />
        <xs:enumeration value="IgnoreWidth" />
        <xs:enumeration value="OrdinalIgnoreCase" />
        <xs:enumeration value="StringSort" />
        <xs:enumeration value="Ordinal" />
      </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="StringComparisonType">
      <xs:annotation>
        <xs:documentation>Represents a .NET StringComparison object to use with .Net
String.Compare and String.Equals as an element. Can be a parameter for a method call
expression.</xs:documentation>
      </xs:annotation>
      <xs:sequence />
      <xs:attribute name="Value" type="tns:StringComparisonParameterEnum" use="required" />
    </xs:complexType>
    <xs:simpleType name="StringComparisonParameterEnum">
      <xs:annotation>
        <xs:documentation>List of all values for .Net StringComparison.</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:enumeration value="CurrentCulture" />
        <xs:enumeration value="CurrentCultureIgnoreCase" />
        <xs:enumeration value="InvariantCulture" />
        <xs:enumeration value="InvariantCultureIgnoreCase" />
        <xs:enumeration value="Ordinal" />
        <xs:enumeration value="OrdinalIgnoreCase" />
      </xs:restriction>
    </xs:simpleType>
    <xs:group name="AnyExpression">
      <xs:annotation>
        <xs:documentation>Placeholder for exactly one expression element of any type within the
CEP expression system.</xs:documentation>
      </xs:annotation>
      <xs:choice>
        <xs:element name="Abs" type="tns:UnaryArithmeticExpression" />
        <xs:element name="Add" type="tns:BinaryArithmeticExpression" />
        <xs:element name="And" type="tns:BinaryExpression" />
        <xs:element name="BitwiseAnd" type="tns:BinaryExpression" />
        <xs:element name="BitwiseOr" type="tns:BinaryExpression" />
        <xs:element name="BitwiseXor" type="tns:BinaryExpression" />
        <xs:element name="Compare" type="tns:ComparisonExpression" />
        <xs:element name="Condition" type="tns:ConditionExpression" />
        <xs:element name="Constant" type="tns:ConstantExpression" />
        <xs:element name="Convert" type="tns:ConvertExpression" />
        <xs:element name="Divide" type="tns:BinaryArithmeticExpression" />
        <xs:element name="Equal" type="tns:ComparisonExpression" />
        <xs:element name="EventKind" type="tns:SystemFieldExpression" />
        <xs:element name="GreaterThan" type="tns:ComparisonExpression" />
        <xs:element name="GreaterThanOrEqual" type="tns:ComparisonExpression" />
        <xs:element name="Hash" type="tns:HashExpression" />
        <xs:element name="InputField" type="tns:InputFieldExpression" />
        <xs:element name="LessThan" type="tns:ComparisonExpression" />
        <xs:element name="LessThanOrEqual" type="tns:ComparisonExpression" />
        <xs:element name="Max" type="tns:NaryArithmeticExpression" />
        <xs:element name="MethodCall" type="tns:MethodCallExpression" />
        <xs:element name="Min" type="tns:NaryArithmeticExpression" />
        <xs:element name="Modulo" type="tns:BinaryArithmeticExpression" />
        <xs:element name="Multiply" type="tns:BinaryArithmeticExpression" />
        <xs:element name="NewValidEndTime" type="tns:SystemFieldExpression" />
        <xs:element name="Negate" type="tns:UnaryArithmeticExpression" />
        <xs:element name="Not" type="tns:UnaryExpression" />
        <xs:element name="NotEqual" type="tns:ComparisonExpression" />
        <xs:element name="Or" type="tns:BinaryExpression" />
        <xs:element name="Subtract" type="tns:BinaryArithmeticExpression" />
        <xs:element name="ValidStartTime" type="tns:SystemFieldExpression" />
        <xs:element name="ValidEndTime" type="tns:SystemFieldExpression" />
      </xs:choice>
    </xs:group>
    <xs:group name="AnyMethodCallSubExpression">
      <xs:annotation>
        <xs:documentation>Placeholder for exactly one element that can be used as arguments for
  method calls (CEP expressions plus culture parameters)</xs:documentation>
      </xs:annotation>
      <xs:choice>
        <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
        <xs:element name="CultureInfo" type="tns:CultureInfoExpression" />
        <xs:element name="CompareOptions" type="tns:CompareOptionsType" />
        <xs:element name="StringComparison" type="tns:StringComparisonType" />
      </xs:choice>
    </xs:group>
    <xs:complexType name="ExpressionContainerType">
      <xs:annotation>
        <xs:documentation>Expression container type. An element of this type must contain
  exactly one expression of any type.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
      </xs:sequence>
    </xs:complexType>
    <xs:complexType name="CultureInfoExpression">
      <xs:annotation>
        <xs:documentation>Contains the description of a culture info to uniquely define a
  culture, either through a constant string or an event field reference. Can only be a
  parameter for a method call expression or a comparison expression.</xs:documentation>
      </xs:annotation>
      <xs:choice>
        <xs:element name="Constant" type="tns:ConstantExpression" />
        <xs:element name="InputField" type="tns:InputFieldExpression" />
      </xs:choice>
    </xs:complexType>
    <xs:complexType name="ExpressionBase">
      <xs:annotation>
        <xs:documentation>Expression base type. Can have 0..n child
  expressions.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:group minOccurs="0" maxOccurs="unbounded" ref="tns:AnyExpression" />
      </xs:sequence>
      <xs:anyAttribute namespace="##any" />
    </xs:complexType>
    <xs:complexType name="NullaryExpression">
      <xs:annotation>
        <xs:documentation>Nullary expression. Has no child expressions.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:restriction base="tns:ExpressionBase">
          <xs:sequence />
          <xs:anyAttribute namespace="##any" />
        </xs:restriction>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="UnaryExpression">
      <xs:annotation>
        <xs:documentation>Unary expression. Has 1 child expression.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:restriction base="tns:ExpressionBase">
          <xs:sequence>
            <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
          </xs:sequence>
          <xs:anyAttribute namespace="##any" />
        </xs:restriction>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="BinaryExpression">
      <xs:annotation>
        <xs:documentation>Binary expression. Has 2 child expressions and arbitrary 
  attributes.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:restriction base="tns:ExpressionBase">
          <xs:sequence>
            <xs:group minOccurs="2" maxOccurs="2" ref="tns:AnyExpression" />
          </xs:sequence>
          <xs:anyAttribute namespace="##any" />
        </xs:restriction>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="UnaryArithmeticExpression">
      <xs:annotation>
        <xs:documentation>Unary arithmetic expression. Has 1 child expression and no 
  attributes.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:restriction base="tns:UnaryExpression">
          <xs:sequence>
            <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="BinaryArithmeticExpression">
      <xs:annotation>
        <xs:documentation>Binary arithmetic expression. Has 2 child expressions and no 
  attributes.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:restriction base="tns:BinaryExpression">
          <xs:sequence>
            <xs:group minOccurs="2" maxOccurs="2" ref="tns:AnyExpression" />
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="NaryArithmeticExpression">
      <xs:annotation>
        <xs:documentation>N-ary arithmetic expression. Has 1..n child expressions and arbitrary 
  attributes.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:restriction base="tns:ExpressionBase">
          <xs:sequence>
            <xs:group minOccurs="1" maxOccurs="unbounded" ref="tns:AnyExpression" />
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="InputFieldExpression">
      <xs:annotation>
        <xs:documentation>Input field expression. Has no child expression. Refers to a field in 
  a stream by the field identifier.</xs:documentation>
```

```
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:restriction base="tns:NullaryExpression">
        <xs:sequence />
        <xs:attributeGroup ref="tns:FieldIdentifier" />
        <xs:attributeGroup ref="tns:StreamIdentifier" />
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="SystemFieldExpression">
    <xs:annotation>
      <xs:documentation>System field expression. Has no child expression. Refers to a system
field in a stream.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:restriction base="tns:NullaryExpression">
        <xs:sequence />
        <xs:attributeGroup ref="tns:StreamIdentifier" />
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="ConstantExpression">
    <xs:annotation>
      <xs:documentation>Constant expression. Has no child expression. Contains type and value
attributes.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:restriction base="tns:NullaryExpression">
        <xs:sequence />
        <xs:attributeGroup ref="tns:TypeIdentifier" />
        <xs:attribute name="Value" type="xs:string" use="optional" />
        <xs:attribute default="false" name="NullValue" type="xs:boolean" use="optional" />
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="ComparisonExpression">
    <xs:annotation>
      <xs:documentation>Comparison expression. Compares two child expressions. The optional
third child expression is the culture info. CompareOptions and StringComparison values are
given as attributes here.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:extension base="tns:BinaryExpression">
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="1" name="CultureInfo"
type="tns:CultureInfoExpression" />
        </xs:sequence>
        <xs:attribute name="CompareOptions" type="tns:CompareOptionsParameterEnumType"
use="optional" />
        <xs:attribute name="StringComparison" type="tns:StringComparisonParameterEnum"
use="optional" />
        <xs:attribute name="IgnoreCase" type="xs:boolean" use="optional" />
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="MethodCallExpression">
    <xs:annotation>
      <xs:documentation>User-defined function. Its value is defined by a method of a class.
0..n input expressions can be passed to the method as parameters. In addition to CEP
expressions, the input can also contain culture-related parameters as
elements.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:extension base="tns:ExpressionBase">
        <xs:sequence>
          <xs:group minOccurs="0" maxOccurs="unbounded" ref="tns:AnyMethodCallSubExpression"
/>
        </xs:sequence>
        <xs:attribute name="Method" type="xs:string" use="required" />
        <xs:attribute name="Class" type="xs:string" use="required" />
        <xs:attribute default="false" name="Deterministic" type="xs:boolean" use="optional"
/>
        <xs:attributeGroup ref="tns:TypeFacetAttributes" />
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="ConvertExpression">
    <xs:annotation>
      <xs:documentation>Conversion expression. Converts one child expression into a
type.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:restriction base="tns:UnaryExpression">
        <xs:sequence>
          <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
        </xs:sequence>
        <xs:attributeGroup ref="tns:TypeIdentifier" />
        <xs:attribute name="DateTimeKind" type="tns:DateTimeType" use="optional" />
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="ConditionExpression">
    <xs:annotation>
      <xs:documentation>
        Condition expression. Has three child expressions:
        1. condition expression
        2. 'then' expression
        3. 'else' expression
      </xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:restriction base="tns:ExpressionBase">
        <xs:sequence>
          <xs:group minOccurs="3" maxOccurs="3" ref="tns:AnyExpression" />
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="HashExpression">
    <xs:annotation>
      <xs:documentation>Hash expression. Represents a hash value based on 1..n child
expressions.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:restriction base="tns:ExpressionBase">
        <xs:sequence>
          <xs:group minOccurs="1" maxOccurs="unbounded" ref="tns:AnyExpression" />
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
  <xs:attributeGroup name="TypeIdentifier">
    <xs:annotation>
      <xs:documentation>Refers to a data type and facets in the StreamInsight type
system.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="Type" type="tns:PrimitiveTypeIdentifier" use="required" />
    <xs:attributeGroup ref="tns:TypeFacetAttributes" />
  </xs:attributeGroup>
  <xs:attributeGroup name="FieldIdentifier">
    <xs:annotation>
      <xs:documentation>Refers to a field within a stream type by its
name.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="Name" type="xs:anyURI" use="required" />
  </xs:attributeGroup>
  <xs:attributeGroup name="StreamIdentifier">
    <xs:annotation>
      <xs:documentation>Refers to a stream by the stream name that was defined in the
corresponding scope.</xs:documentation>
```

```
      </xs:annotation>
      <xs:attribute name="StreamName" type="xs:anyURI" use="optional" />
    </xs:attributeGroup>
    <xs:simpleType name="DateTimeType">
      <xs:restriction base="xs:string">
        <xs:enumeration value="Utc" />
        <xs:enumeration value="Local" />
      </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="AnySingleUserElementType">
      <xs:annotation>
        <xs:documentation>Contains one user-defined XML element. The element has to define a 
separate namespace.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:any minOccurs="1" maxOccurs="1" namespace="##any" processContents="skip" />
      </xs:sequence>
    </xs:complexType>
    <xs:complexType name="SerializedConfigurationType">
      <xs:annotation>
        <xs:documentation>Runtime configuration structure for the UDO/UDA.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:AnySingleUserElementType">
          <xs:attribute name="Class" type="xs:string" use="required">
            <xs:annotation>
              <xs:documentation>Serialized class name of the configuration 
structure.</xs:documentation>
            </xs:annotation>
          </xs:attribute>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="StreamDefinitionType">
      <xs:annotation>
        <xs:documentation>ID that defines a stream. Stream here denotes the connection between 
operators.</xs:documentation>
      </xs:annotation>
      <xs:attribute name="Name" type="xs:anyURI" use="required" />
    </xs:complexType>
    <xs:complexType name="StreamReferenceType">
      <xs:annotation>
        <xs:documentation>ID that refers to a stream.</xs:documentation>
      </xs:annotation>
      <xs:attribute name="Name" type="xs:anyURI" use="required" />
    </xs:complexType>
    <xs:complexType name="TerminatorBaseType">
      <xs:annotation>
        <xs:documentation>Base type for stream termination elements.</xs:documentation>
      </xs:annotation>
    </xs:complexType>
    <xs:complexType name="OperatorBaseType">
      <xs:annotation>
        <xs:documentation>Operator base type. Every operator has a name.</xs:documentation>
      </xs:annotation>
      <xs:attribute name="Name" type="xs:anyURI" use="required" />
    </xs:complexType>
    <xs:group name="AnyWindow">
      <xs:annotation>
        <xs:documentation>Placeholder for exactly one window element of any 
type.</xs:documentation>
      </xs:annotation>
      <xs:choice>
        <xs:element name="SnapshotWindow" type="tns:SnapshotWindowType" />
        <xs:element name="HoppingWindow" type="tns:HoppingWindowType" />
        <xs:element name="CountByStartTimeWindow" type="tns:CountByStartTimeWindowType" />
      </xs:choice>
    </xs:group>
    <xs:complexType name="SnapshotWindowType">
      <xs:sequence>
```

```
        <xs:element name="WindowDefinition" type="tns:SnapshotWindowDefinitionType" />
        <xs:element name="InputPolicy" type="tns:WindowInputPolicyType" />
        <xs:element name="OutputPolicy" type="tns:SnapshotWindowOutputPolicyType" />
      </xs:sequence>
    </xs:complexType>
    <xs:complexType name="HoppingWindowType">
      <xs:sequence>
        <xs:element name="WindowDefinition" type="tns:HoppingWindowDefinitionType" />
        <xs:element name="InputPolicy" type="tns:WindowInputPolicyType" />
        <xs:element name="OutputPolicy" type="tns:WindowOutputPolicyType" />
      </xs:sequence>
    </xs:complexType>
    <xs:complexType name="CountByStartTimeWindowType">
      <xs:sequence>
        <xs:element name="WindowDefinition" type="tns:CountByStartTimeWindowDefinitionType" />
        <xs:element name="InputPolicy" type="tns:WindowInputPolicyType" />
        <xs:element name="OutputPolicy" type="tns:WindowOutputPolicyType" />
      </xs:sequence>
    </xs:complexType>
    <xs:complexType name="SnapshotWindowDefinitionType">
      <xs:annotation>
        <xs:documentation>Snapshot window. Temporal window properties are defined by the stream
  of events.</xs:documentation>
      </xs:annotation>
      <xs:sequence />
    </xs:complexType>
    <xs:complexType name="HoppingWindowDefinitionType">
      <xs:annotation>
        <xs:documentation>Fixed length window. Defined by a fixed window size, a hop size and
  an optional alignment.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element minOccurs="1" maxOccurs="1" name="Size" type="xs:duration" />
        <xs:element minOccurs="1" maxOccurs="1" name="HopSize" type="xs:duration" />
        <xs:element minOccurs="1" maxOccurs="1" name="Alignment" type="xs:dateTime" />
      </xs:sequence>
    </xs:complexType>
    <xs:complexType name="CountByStartTimeWindowDefinitionType">
      <xs:annotation>
        <xs:documentation>Count start times window. Defined by the count of member event start
  times.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element minOccurs="1" maxOccurs="1" name="Size" type="xs:int" />
        <xs:element minOccurs="1" maxOccurs="1" name="HopSize" type="xs:int" />
      </xs:sequence>
    </xs:complexType>
    <xs:complexType name="WindowInputPolicyType">
      <xs:annotation>
        <xs:documentation>Specifies how to modify the temporal characteristics of events when
  they are passed to a time-sensitive user-defined operator/aggregate.</xs:documentation>
      </xs:annotation>
      <xs:choice>
        <xs:element name="Clip" type="tns:WindowInputPolicyClipType" />
      </xs:choice>
    </xs:complexType>
    <xs:complexType name="WindowInputPolicyClipType">
      <xs:annotation>
        <xs:documentation>Specifies how to clip events that are input to a UDO/UDA with respect
  to the window boundaries. Events that are members of the window are not necessarily fully
  contained in the window. Hence, a clipping behavior on both window boundaries can be
  given.</xs:documentation>
      </xs:annotation>
      <xs:sequence />
      <xs:attribute name="Left" type="xs:boolean" use="required" />
      <xs:attribute name="Right" type="xs:boolean" use="required" />
    </xs:complexType>
    <xs:complexType name="WindowOutputPolicyType">
      <xs:choice>
        <xs:element name="Unaltered" />
        <xs:element name="Clip" type="tns:WindowOutputPolicyClipType" />
        <xs:element name="Adjust" type="tns:WindowOutputPolicyAdjustType" />
      </xs:choice>
    </xs:complexType>
    <xs:complexType name="SnapshotWindowOutputPolicyType">
      <xs:choice>
        <xs:element name="Unaltered" />
        <xs:element name="Clip" type="tns:SnapshotWindowOutputPolicyClipType" />
        <xs:element name="Adjust" type="tns:SnapshotWindowOutputPolicyAdjustType" />
      </xs:choice>
    </xs:complexType>
    <xs:complexType name="SnapshotWindowOutputPolicyClipType">
      <xs:sequence />
      <xs:attribute name="Type" type="tns:SnapshotWindowOutputPolicyClipEnumType"
  use="required" />
    </xs:complexType>
    <xs:complexType name="WindowOutputPolicyClipType">
      <xs:sequence />
      <xs:attribute name="Type" type="tns:WindowOutputPolicyClipEnumType" use="required" />
    </xs:complexType>
    <xs:simpleType name="SnapshotWindowOutputPolicyClipEnumType">
      <xs:annotation>
        <xs:documentation>Snapshot windows allow the clipping of the returned events to the
  window size.</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:enumeration value="WindowEnd" />
      </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="WindowOutputPolicyClipEnumType">
      <xs:annotation>
        <xs:documentation>Hopping windows allow the clipping of the returned events to the
  window size or the hop size.</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:enumeration value="Hop" />
        <xs:enumeration value="WindowEnd" />
      </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="SnapshotWindowOutputPolicyAdjustType">
      <xs:sequence />
      <xs:attribute name="Lifetime" type="tns:SnapshotWindowOutputPolicyAdjustLifetimeEnumType"
  use="required" />
      <xs:attribute name="Alignment" type="tns:SnapshotOutputPolicyAdjustAlignmentEnumType"
  use="required" />
    </xs:complexType>
    <xs:complexType name="WindowOutputPolicyAdjustType">
      <xs:sequence />
      <xs:attribute name="Lifetime" type="tns:WindowOutputPolicyAdjustLifetimeEnumType"
  use="required" />
      <xs:attribute name="Alignment" type="tns:WindowOutputPolicyAdjustAlignmentEnumType"
  use="required" />
    </xs:complexType>
    <xs:simpleType name="SnapshotWindowOutputPolicyAdjustLifetimeEnumType">
      <xs:annotation>
        <xs:documentation>Snapshot windows allow the adjustment of the returned events'
  lifetimes to the window size or to a point event.</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:enumeration value="WindowSize" />
        <xs:enumeration value="Point" />
      </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="SnapshotOutputPolicyAdjustAlignmentEnumType">
      <xs:annotation>
        <xs:documentation>Snapshot windows allow the alignment of the returned events'
  lifetimes to the window start or end.</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:enumeration value="WindowStart" />
        <xs:enumeration value="WindowEnd" />
      </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="WindowOutputPolicyAdjustLifetimeEnumType">
      <xs:annotation>
        <xs:documentation>Hopping windows allow the adjustment of the returned events'
lifetimes to the window size, the hop size or to a point event.</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:enumeration value="WindowSize" />
        <xs:enumeration value="HopSize" />
        <xs:enumeration value="Point" />
      </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="WindowOutputPolicyAdjustAlignmentEnumType">
      <xs:annotation>
        <xs:documentation>Snapshot windows allow the alignment of the returned events'
lifetimes to the window start, the window end or to the hop offset.</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:enumeration value="WindowStart" />
        <xs:enumeration value="WindowEnd" />
        <xs:enumeration value="Hop" />
      </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="WindowedOperatorBaseType">
      <xs:annotation>
        <xs:documentation>Windowed Operator base type. Includes the definition for
windows.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:OperatorBaseType">
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
type="tns:StreamReferenceType" />
            <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
type="tns:StreamDefinitionType" />
            <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyWindow" />
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="QueryTemplateType">
      <xs:annotation>
        <xs:documentation>A Query template has m import and n export
operators.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element minOccurs="1" maxOccurs="unbounded" name="Import"
type="tns:ImportOperatorType" />
        <xs:element minOccurs="1" maxOccurs="unbounded" name="Export"
type="tns:ExportOperatorType" />
        <xs:group minOccurs="0" maxOccurs="unbounded" ref="tns:AnyOperator" />
      </xs:sequence>
      <xs:attribute name="Name" type="xs:anyURI" />
      <xs:attribute name="Description" type="xs:string" use="optional" />
    </xs:complexType>
    <xs:group name="AnyOperator">
      <xs:annotation>
        <xs:documentation>Placeholder for exactly one operator element of any
type.</xs:documentation>
      </xs:annotation>
      <xs:choice>
        <xs:element name="QueryTemplateReference" type="tns:QueryTemplateReferenceOperatorType"
/>
        <xs:element name="Multicast" type="tns:MulticastOperatorType" />
        <xs:element name="Project" type="tns:ProjectOperatorType" />
        <xs:element name="Select" type="tns:SelectOperatorType" />
        <xs:element name="Join" type="tns:JoinOperatorType" />
        <xs:element name="Union" type="tns:UnionOperatorType" />
        <xs:element name="Aggregate" type="tns:AggregationOperatorType" />
        <xs:element name="AlterLifetime" type="tns:AlterLifetimeOperatorType" />
        <xs:element name="GroupAndApply" type="tns:GroupAndApplyOperatorType" />
        <xs:element name="TopK" type="tns:TopKOperatorType" />
        <xs:element name="UserDefined" type="tns:UserDefinedOperatorType" />
        <xs:element name="UserDefinedStream" type="tns:UserDefinedStreamOperatorType" />
      </xs:choice>
    </xs:group>
    <xs:complexType name="ExportOperatorType">
      <xs:annotation>
        <xs:documentation>Export Operator. Makes the query's outgoing stream explicit. The Name
attribute identifies the stream. Refers to a single operator as its input.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:TerminatorBaseType">
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
type="tns:StreamReferenceType" />
          </xs:sequence>
          <xs:attribute name="Name" type="xs:anyURI" use="required" />
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ImportOperatorType">
      <xs:annotation>
        <xs:documentation>Import Operator. Denotes the query's import stream. The Name
attribute identifies the stream. Refers to a single operator as its output. The attribute
Type refers to the stream type using the type's name.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:TerminatorBaseType">
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
type="tns:StreamDefinitionType" />
          </xs:sequence>
          <xs:attribute name="Name" type="xs:anyURI" use="required" />
          <xs:attribute name="Type" type="xs:anyURI" use="required" />
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="QTrefInputStreamType">
      <xs:annotation>
        <xs:documentation>Type for the input stream in an QT reference operator. In addition to
the local stream name, it also needs to refer to the respective endpoint in the other query
template. This is done via the attribute "ExternalName". It refers to the stream name that is
used in the Import in the embedded query template.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:StreamReferenceType">
          <xs:attribute name="ExternalName" type="xs:anyURI" use="required" />
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="QTrefOutputStreamType">
      <xs:annotation>
        <xs:documentation>Type for the output stream in an QT reference operator. In addition
to the local stream name, it also needs to refer to the respective endpoint in the other
query template. This is done via the attribute "ExternalName". It refers to the stream name
that is used in a Export in the embedded query template.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:StreamDefinitionType">
          <xs:attribute name="ExternalName" type="xs:anyURI" use="required" />
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="QueryTemplateReferenceOperatorType">
      <xs:annotation>
        <xs:documentation>Embeds another query template in the query.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:OperatorBaseType">
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="unbounded" name="InputStream"
type="tns:QTrefInputStreamType" />
            <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
type="tns:QTrefOutputStreamType" />
          </xs:sequence>
          <xs:attribute name="QueryTemplateName" type="xs:anyURI" use="required" />
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="MulticastOperatorType">
      <xs:annotation>
        <xs:documentation>A multicast creates multiple named streams out of a single input
stream. The input events are simply replicated to all outputs.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:OperatorBaseType">
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
type="tns:StreamReferenceType" />
            <xs:element minOccurs="2" maxOccurs="unbounded" name="OutputStream"
type="tns:StreamDefinitionType" />
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ProjectExpressionContainerType">
      <xs:annotation>
        <xs:documentation>A project expression contains a single expression that determines the
value of a new event field. It extends the base container type by adding an attribute to
assign a name to that new field. This is also a base class for other operators' expressions
that result in new event fields.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:ExpressionContainerType">
          <xs:attribute name="OutputField" type="xs:anyURI" use="required" />
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ProjectOperatorType">
      <xs:annotation>
        <xs:documentation>A project operator applies an arbitrary number of project expressions
to a single input stream and yields a single output stream.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:OperatorBaseType">
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
type="tns:StreamReferenceType" />
            <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
type="tns:StreamDefinitionType" />
            <xs:element minOccurs="0" maxOccurs="unbounded" name="ProjectExpression"
type="tns:ProjectExpressionContainerType" />
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="SelectOperatorType">
      <xs:annotation>
        <xs:documentation>A select expression contains exactly one filter
expression.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:OperatorBaseType">
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
type="tns:StreamReferenceType" />
```

```
          <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
type="tns:StreamDefinitionType" />
          <xs:element minOccurs="1" maxOccurs="1" name="FilterExpression"
type="tns:ExpressionContainerType" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="JoinOperatorType">
    <xs:annotation>
      <xs:documentation>A Join element has two inputs and one output. The join predicate is
specified as a child element. The join can include zero or more ProjectExpressions, which
define the output schema.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:extension base="tns:OperatorBaseType">
        <xs:sequence>
          <xs:element minOccurs="2" maxOccurs="2" name="InputStream"
type="tns:StreamReferenceType" />
          <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
type="tns:StreamDefinitionType" />
          <xs:element minOccurs="1" maxOccurs="1" name="JoinPredicate"
type="tns:ExpressionContainerType" />
          <xs:element minOccurs="0" maxOccurs="unbounded" name="ProjectExpression"
type="tns:ProjectExpressionContainerType" />
        </xs:sequence>
        <xs:attribute name="JoinType">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:enumeration value="LeftOuter" />
              <xs:enumeration value="RightOuter" />
              <xs:enumeration value="FullOuter" />
              <xs:enumeration value="LeftAnti" />
              <xs:enumeration value="RightAnti" />
              <xs:enumeration value="LeftSemi" />
              <xs:enumeration value="RightSemi" />
              <xs:enumeration value="LeftAntiSemi" />
              <xs:enumeration value="RightAntiSemi" />
              <xs:enumeration value="Inner" />
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute default="false" name="PointEvents" type="xs:boolean" use="optional" />
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="UnionOperatorType">
    <xs:annotation>
      <xs:documentation>A union operator funnels multiple input stream into one output
stream.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:extension base="tns:OperatorBaseType">
        <xs:sequence>
          <xs:element minOccurs="2" maxOccurs="unbounded" name="InputStream"
type="tns:StreamReferenceType" />
          <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
type="tns:StreamDefinitionType" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="AggregateBaseType">
    <xs:annotation>
      <xs:documentation>Base type for a single aggregation. The result is always assigned to
an output field.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="OutputField" type="xs:anyURI" use="required" />
  </xs:complexType>
  <xs:complexType name="AggregateSumType">
```

```
    <xs:annotation>
      <xs:documentation>Sum over an expression evaluated on all input
events.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:extension base="tns:AggregateBaseType">
        <xs:sequence>
          <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="AggregateMinType">
    <xs:annotation>
      <xs:documentation>Numeric minimum of expressions evaluated on all input
events.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:extension base="tns:AggregateBaseType">
        <xs:sequence>
          <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="AggregateMaxType">
    <xs:annotation>
      <xs:documentation>Numeric maximum of expressions evaluated on all input
events.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:extension base="tns:AggregateBaseType">
        <xs:sequence>
          <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="AggregateAvgType">
    <xs:annotation>
      <xs:documentation>Numeric average of expressions evaluated on all input
events.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:extension base="tns:AggregateBaseType">
        <xs:sequence>
          <xs:group minOccurs="1" maxOccurs="1" ref="tns:AnyExpression" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="AggregateUserDefinedType">
    <xs:annotation>
      <xs:documentation>A user-defined aggregate operates against a window of events and
returns a single scalar value.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:extension base="tns:AggregateBaseType">
        <xs:sequence>
          <xs:element minOccurs="1" maxOccurs="1" name="Implementation"
type="tns:ImplementationType" />
          <xs:element minOccurs="0" maxOccurs="1" name="Configuration"
type="tns:SerializedConfigurationType" />
          <xs:group minOccurs="0" maxOccurs="1" ref="tns:AnyExpression" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:group name="AnyAggregate">
    <xs:annotation>
```

```
        <xs:documentation>Set of all aggregation functions.</xs:documentation>
      </xs:annotation>
      <xs:choice>
        <xs:element name="Sum" type="tns:AggregateSumType" />
        <xs:element name="Count" type="tns:AggregateBaseType" />
        <xs:element name="Min" type="tns:AggregateMinType" />
        <xs:element name="Max" type="tns:AggregateMaxType" />
        <xs:element name="Avg" type="tns:AggregateAvgType" />
        <xs:element name="UserDefined" type="tns:AggregateUserDefinedType" />
      </xs:choice>
    </xs:group>
    <xs:complexType name="AggregationOperatorType">
      <xs:annotation>
        <xs:documentation>An aggregate element has one or more aggregate expressions, each 
yielding a new column that represents the aggregation result.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:WindowedOperatorBaseType">
          <xs:sequence>
            <xs:group minOccurs="1" maxOccurs="unbounded" ref="tns:AnyAggregate" />
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="UserDefinedStreamOperatorType">
      <xs:annotation>
        <xs:documentation>A user-defined stream operator is defined on top of a stream of 
events and implements a custom function, return a stream of events.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:OperatorBaseType">
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="InputStream" 
type="tns:StreamReferenceType" />
            <xs:element minOccurs="1" maxOccurs="1" name="OutputStream" 
type="tns:StreamDefinitionType" />
            <xs:element minOccurs="1" maxOccurs="1" name="InitialState" 
            type="tns:SerializedConfigurationType" />
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="UserDefinedOperatorType">
      <xs:annotation>
        <xs:documentation>A user-defined operator (UDO) is defined on top of a window of events 
and implements a custom function, returning a set of events.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:WindowedOperatorBaseType">
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="Implementation" 
type="tns:ImplementationType" />
            <xs:element minOccurs="0" maxOccurs="1" name="Configuration" 
type="tns:SerializedConfigurationType" />
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ImplementationType">
      <xs:annotation>
        <xs:documentation>Specifies the signature of a user-defined 
operation/aggregation.</xs:documentation>
      </xs:annotation>
      <xs:attribute name="Class" type="xs:string" use="required">
        <xs:annotation>
          <xs:documentation>The .Net strong name of the implemented class.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="InputClrType" type="xs:string" use="required">
        <xs:annotation>
          <xs:documentation>The input type as a CLR strong name.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="ReturnClrType" type="xs:string" use="required">
        <xs:annotation>
          <xs:documentation>The output type as a CLR strong name.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:complexType>
    <xs:complexType name="AlterLifetimeOperatorType">
      <xs:annotation>
        <xs:documentation>An AlterLifetime operator defines two expressions: One for the new 
  start time and one for the new life time of the event. At least one of these must be 
  specified.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:OperatorBaseType">
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="InputStream" 
  type="tns:StreamReferenceType" />
            <xs:element minOccurs="1" maxOccurs="1" name="OutputStream" 
  type="tns:StreamDefinitionType" />
            <xs:element minOccurs="0" maxOccurs="1" name="StartTimeExpression" 
  type="tns:ExpressionContainerType" />
            <xs:element minOccurs="0" maxOccurs="1" name="LifetimeExpression" 
  type="tns:ExpressionContainerType" />
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ApplyOutputType">
      <xs:annotation>
        <xs:documentation>Output terminator of the apply operator graph.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:TerminatorBaseType">
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="InputStream" 
  type="tns:StreamReferenceType" />
          </xs:sequence>
          <xs:attribute name="Name" type="xs:anyURI" use="required" />
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ApplyInputType">
      <xs:annotation>
        <xs:documentation>Input terminator of the apply operator graph.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:TerminatorBaseType">
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="OutputStream" 
  type="tns:StreamDefinitionType" />
          </xs:sequence>
          <xs:attribute name="Name" type="xs:anyURI" use="required" />
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ApplyBranchType">
      <xs:annotation>
        <xs:documentation>The Apply element encapsulates the apply operator graph of the Group 
  and Apply operator. It must have exactly one input and one output, which are terminated by 
  elements of type ApplyInputType and ApplyOutputType. These elements are named ImportOperator 
  and ExportOperator to be able to re-use exisitng query templates as apply branches. However, 
  their type here is different from query-template-level imports and exports in that they do 
  not require a type specification.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element minOccurs="1" maxOccurs="1" name="ApplyInput" type="tns:ApplyInputType" />
        <xs:element minOccurs="1" maxOccurs="1" name="ApplyOutput" type="tns:ApplyOutputType"
  />
        <xs:group minOccurs="0" maxOccurs="unbounded" ref="tns:AnyOperator" />
      </xs:sequence>
    </xs:complexType>
    <xs:complexType name="GroupAndApplyOperatorType">
      <xs:annotation>
        <xs:documentation>
          Implements the Group and Apply operator. One or more grouping expressions determine
  the event partitions. The operator graph in the Apply element will be applied to each group
  separately. The grouping expression is of the same type as the project expression: it can
  contain any expression, but it must assign a field name to that expression result.
        </xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:OperatorBaseType">
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="InputStream"
  type="tns:StreamReferenceType" />
            <xs:element minOccurs="1" maxOccurs="1" name="OutputStream"
  type="tns:StreamDefinitionType" />
            <xs:element minOccurs="1" maxOccurs="unbounded" name="GroupingExpression"
  type="tns:ProjectExpressionContainerType" />
            <xs:element minOccurs="1" maxOccurs="1" name="Apply" type="tns:ApplyBranchType">
              <xs:key name="ApplyStreamKey">
                <xs:annotation>
                  <xs:documentation>Stream identifier to be used in the operators of that apply
  element.</xs:documentation>
                </xs:annotation>
                <xs:selector xpath="./*/tns:OutputStream" />
                <xs:field xpath="@Name" />
              </xs:key>
              <xs:keyref name="ApplyStreamKeyref" refer="tns:ApplyStreamKey">
                <xs:annotation>
                  <xs:documentation>Stream reference for operators. A stream reference has to
  match a stream identifier in order to connect operators.</xs:documentation>
                </xs:annotation>
                <xs:selector xpath="./*/tns:InputStream" />
                <xs:field xpath="@Name" />
              </xs:keyref>
            </xs:element>
          </xs:sequence>
          <xs:attribute default="false" name="AddGroupingFields" type="xs:boolean"
  use="optional" />
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:simpleType name="RankOrderType">
      <xs:annotation>
        <xs:documentation>The ordering of a rank expression can be ascending or
  descending.</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:enumeration value="Ascending" />
        <xs:enumeration value="Descending" />
      </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="RankExpressionContainerType">
      <xs:annotation>
        <xs:documentation>A rank expression contains a single expression that is to be used to
  determine the rank in a TopK operator. It extends the base container type by adding an
  attribute to specify the ordering.</xs:documentation>
      </xs:annotation>
      <xs:complexContent mixed="false">
        <xs:extension base="tns:ExpressionContainerType">
          <xs:attribute name="Order" type="tns:RankOrderType" use="required" />
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="TopKOperatorType">
```

```
    <xs:annotation>
      <xs:documentation>TopK operator. The K is specified by the required RankDepth
attribute. The calculated rank can be projected in the output of the operator by specifying a
field name through the attribute RankOutputField. The rank is calculated according to the
value of the rank expression, its datatype, and the specified ordering. If more than one rank
expression is specified, they are evaluated subsequently, i.e., if one rank expression
evaluates for a tie for any two events, the next expression in the sequence is evaluated,
etc.</xs:documentation>
    </xs:annotation>
    <xs:complexContent mixed="false">
      <xs:extension base="tns:WindowedOperatorBaseType">
        <xs:sequence>
          <xs:element minOccurs="1" maxOccurs="unbounded" name="RankExpression"
type="tns:RankExpressionContainerType" />
        </xs:sequence>
        <xs:attribute name="RankDepth" type="xs:int" use="required" />
        <xs:attribute name="RankOutputField" type="xs:anyURI" use="optional" />
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:simpleType name="StreamEventOrderingType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="ChainOrdered" />
      <xs:enumeration value="FullyOrdered" />
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="EventShapeType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Point" />
      <xs:enumeration value="Interval" />
      <xs:enumeration value="Edge" />
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="OutputStreamBindingType">
    <xs:annotation>
      <xs:documentation>Output Stream Binding. Pairs a stream sink with a query
template.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="AdapterConfiguration"
type="tns:AnySingleUserElementType">
        <xs:annotation>
          <xs:documentation>The contained XML element will be passed to the output adapter as
initialization information. The child element is serialized from user-defined adapter
configuration structure and has arity of one.</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="OutputStream" type="xs:anyURI" use="required">
      <xs:annotation>
        <xs:documentation>Reference to an export operator name.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="OutputStreamTarget" type="xs:anyURI" use="required">
      <xs:annotation>
        <xs:documentation>Reference to an output adapter.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="OutputStreamConsumerName" type="xs:anyURI" use="optional">
      <xs:annotation>
        <xs:documentation>The unique identifier to identify a given consumer of the
query.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="EventShape" type="tns:EventShapeType" use="optional">
      <xs:annotation>
        <xs:documentation>Desired event shape in the output.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="StreamEventOrdering" type="tns:StreamEventOrderingType" 
use="optional">
      <xs:annotation>
        <xs:documentation>Desired time ordering at the output.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="PayloadClassName" type="xs:string" use="optional" />
  </xs:complexType>
  <xs:complexType name="EntityType">
    <xs:annotation>
      <xs:documentation>The schema of a CreateEntity command. It contains the schema 
definition for an entity.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="1" ref="linqs:ExpressionRoot" />
    </xs:sequence>
    <xs:attribute name="Name" type="xs:anyURI" use="required" />
  </xs:complexType>
  <xs:complexType name="AdvanceTimeEventCountFrequencyType">
    <xs:annotation>
      <xs:documentation>Specifies the frequency at which to advance application time in terms 
of event count.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="Value" type="xs:unsignedInt" use="required" />
  </xs:complexType>
  <xs:complexType name="AdvanceTimeDurationFrequencyType">
    <xs:annotation>
      <xs:documentation>Specifies the frequency at which to advance application time in terms 
of time duration.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="Value" type="xs:duration" use="required" />
  </xs:complexType>
  <xs:complexType name="AdvanceTimeDelayType">
    <xs:annotation>
      <xs:documentation>Specifies delay in terms of time duration. The application time is 
advanced to the start time of the most recent event minus the duration.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="Value" type="xs:duration" use="required" />
  </xs:complexType>
  <xs:complexType name="AdvanceToInfinityType">
    <xs:annotation>
      <xs:documentation>Specifies whether an additional CTI with timestamp infinity should be 
generated at query shutdown.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="Value" type="xs:boolean" use="required" />
  </xs:complexType>
  <xs:complexType name="AdvanceTimeGenerateType">
    <xs:annotation>
      <xs:documentation>Specifies how to generate CTIs in order to advance time. The 
generation definition has two dimensions, as one child element each: (i) the frequency of 
advancing application time and (ii) the delay of the application time increments. The 
frequency can be given as a time period or as an event count. The delay has to be given as a 
time period.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:choice>
        <xs:element name="EventCountFrequency" type="tns:AdvanceTimeEventCountFrequencyType" 
/>
        <xs:element name="DurationFrequency" type="tns:AdvanceTimeDurationFrequencyType" />
      </xs:choice>
      <xs:element name="Delay" type="tns:AdvanceTimeDelayType" />
      <xs:element name="AdvanceToInfinityOnShutdown" type="tns:AdvanceToInfinityType" />
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="AdvanceTimeImportType">
    <xs:annotation>
      <xs:documentation>Specifies where to import CTIs from in order to advance 
time.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="StreamName" type="xs:string" use="required" />
  </xs:complexType>
  <xs:simpleType name="AdvanceTimePolicyType">
    <xs:annotation>
      <xs:documentation>The policy type of advance time. Adjust will change the violating
event's start time to the earliest valid time. Drop will drop the violating
event.</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="Adjust" />
      <xs:enumeration value="Drop" />
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="AdvanceTimeType">
    <xs:annotation>
      <xs:documentation>Specifies how to add CTIs as part of the binding. Can be either
generated or imported from another stream or both.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="Generate"
type="tns:AdvanceTimeGenerateType" />
      <xs:element minOccurs="0" maxOccurs="1" name="Import" type="tns:AdvanceTimeImportType"
/>
    </xs:sequence>
    <xs:attribute name="Policy" type="tns:AdvanceTimePolicyType" use="required">
      <xs:annotation>
        <xs:documentation>Specifies how to treat incoming events that violate advance time
CTIs.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="InputStreamBindingType">
    <xs:annotation>
      <xs:documentation>Input Stream Binding. Pairs a stream source with a query
template.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="AdvanceTime" type="tns:AdvanceTimeType"
/>
      <xs:element minOccurs="0" maxOccurs="1" name="AdapterConfiguration"
type="tns:AnySingleUserElementType" />
    </xs:sequence>
    <xs:attribute name="InputStream" type="xs:anyURI" use="required">
      <xs:annotation>
        <xs:documentation>Reference to an import operator name.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="InputStreamSource" type="xs:anyURI" use="required">
      <xs:annotation>
        <xs:documentation>Reference to an input adapter.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="EventShape" type="tns:EventShapeType" use="required">
      <xs:annotation>
        <xs:documentation>Desired event shape in the input.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="PayloadClassName" type="xs:string" use="optional" />
  </xs:complexType>
  <xs:complexType name="QueryType">
    <xs:annotation>
      <xs:documentation>The schema of a CreateQuery command. It contains information to bind
a query template's input and output streams to stream sources and sinks.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="OutputStreamBinding"
type="tns:OutputStreamBindingType" />
      <xs:element minOccurs="1" maxOccurs="unbounded" name="InputStreamBinding"
type="tns:InputStreamBindingType" />
    </xs:sequence>
    <xs:attribute name="Name" type="xs:anyURI" use="required" />
```

```
      <xs:attribute name="QueryTemplate" type="xs:anyURI" use="required" />
      <xs:attribute name="Description" type="xs:string" use="optional" />
      <xs:attribute name="IsResilient" type="xs:boolean"use="optional" />
    </xs:complexType>
    <xs:element name="Application" type="tns:ApplicationType" />
    <xs:element name="EventType" type="tns:EventType" />
    <xs:element name="InputAdapter" type="tns:InputAdapterType" />
    <xs:element name="OutputAdapter" type="tns:OutputAdapterType" />
    <xs:element name="QueryTemplate" type="tns:QueryTemplateType">
      <xs:unique name="OperatorKey">
        <xs:annotation>
          <xs:documentation>Operator names are defined as unique.</xs:documentation>
        </xs:annotation>
        <xs:selector xpath="./*" />
        <xs:field xpath="@Name" />
      </xs:unique>
      <xs:key name="StreamKey">
        <xs:annotation>
          <xs:documentation>Stream identifier to be used in the operators of the query
template.</xs:documentation>
        </xs:annotation>
        <xs:selector xpath="./*/tns:OutputStream" />
        <xs:field xpath="@Name" />
      </xs:key>
      <xs:keyref name="StreamKeyref" refer="tns:StreamKey">
        <xs:annotation>
          <xs:documentation>Stream reference for operators. A stream reference has to match a
stream identifier in order to connect operators.</xs:documentation>
        </xs:annotation>
        <xs:selector xpath="./*/tns:InputStream" />
        <xs:field xpath="@Name" />
      </xs:keyref>
    </xs:element>
    <xs:element name="Query" type="tns:QueryType" />
    <xs:element name="Entity" type="tns:EntityType" />
</xs:schema>
```

3.4 W3C Addressing Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    W3C XML Schema defined in the Web Services Addressing 1.0 specification
    http://www.w3.org/TR/ws-addr-core

   Copyright Â© 2005 World Wide Web Consortium,

   (Massachusetts Institute of Technology, European Research Consortium for
   Informatics and Mathematics, Keio University). All Rights Reserved. This
   work is distributed under the W3CÂ® Software License [1] in the hope that
   it will be useful, but WITHOUT ANY WARRANTY; without even the implied
   warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

   [1] http://www.w3.org/Consortium/Legal/2002/copyright-software-20021231

   $Id: ws-addr.xsd,v 1.2 2008/07/23 13:38:16 plehegar Exp $
-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://www.w3.org/2005/08/addressing"
targetNamespace="http://www.w3.org/2005/08/addressing" blockDefault="#all"
elementFormDefault="qualified" finalDefault="" attributeFormDefault="unqualified">

   <!-- Constructs from the WS-Addressing Core -->

   <xs:element name="EndpointReference" type="tns:EndpointReferenceType"/>
   <xs:complexType name="EndpointReferenceType" mixed="false">
      <xs:sequence>
         <xs:element name="Address" type="tns:AttributedURIType"/>
```

```
          <xs:element ref="tns:ReferenceParameters" minOccurs="0"/>
          <xs:element ref="tns:Metadata" minOccurs="0"/>
          <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
       </xs:sequence>
       <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <xs:element name="ReferenceParameters" type="tns:ReferenceParametersType"/>
    <xs:complexType name="ReferenceParametersType" mixed="false">
       <xs:sequence>
          <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
       </xs:sequence>
       <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <xs:element name="Metadata" type="tns:MetadataType"/>
    <xs:complexType name="MetadataType" mixed="false">
       <xs:sequence>
          <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
       </xs:sequence>
       <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <xs:element name="MessageID" type="tns:AttributedURIType"/>
    <xs:element name="RelatesTo" type="tns:RelatesToType"/>
    <xs:complexType name="RelatesToType" mixed="false">
       <xs:simpleContent>
          <xs:extension base="xs:anyURI">
             <xs:attribute name="RelationshipType" type="tns:RelationshipTypeOpenEnum"
use="optional" default="http://www.w3.org/2005/08/addressing/reply"/>
             <xs:anyAttribute namespace="##other" processContents="lax"/>
          </xs:extension>
       </xs:simpleContent>
    </xs:complexType>

    <xs:simpleType name="RelationshipTypeOpenEnum">
       <xs:union memberTypes="tns:RelationshipType xs:anyURI"/>
    </xs:simpleType>

    <xs:simpleType name="RelationshipType">
       <xs:restriction base="xs:anyURI">
          <xs:enumeration value="http://www.w3.org/2005/08/addressing/reply"/>
       </xs:restriction>
    </xs:simpleType>

    <xs:element name="ReplyTo" type="tns:EndpointReferenceType"/>
    <xs:element name="From" type="tns:EndpointReferenceType"/>
    <xs:element name="FaultTo" type="tns:EndpointReferenceType"/>
    <xs:element name="To" type="tns:AttributedURIType"/>
    <xs:element name="Action" type="tns:AttributedURIType"/>

    <xs:complexType name="AttributedURIType" mixed="false">
       <xs:simpleContent>
          <xs:extension base="xs:anyURI">
             <xs:anyAttribute namespace="##other" processContents="lax"/>
          </xs:extension>
       </xs:simpleContent>
    </xs:complexType>

    <!-- Constructs from the WS-Addressing SOAP binding -->

    <xs:attribute name="IsReferenceParameter" type="xs:boolean"/>

    <xs:simpleType name="FaultCodesOpenEnumType">
       <xs:union memberTypes="tns:FaultCodesType xs:QName"/>
    </xs:simpleType>
```

```
   <xs:simpleType name="FaultCodesType">
      <xs:restriction base="xs:QName">
         <xs:enumeration value="tns:InvalidAddressingHeader"/>
         <xs:enumeration value="tns:InvalidAddress"/>
         <xs:enumeration value="tns:InvalidEPR"/>
         <xs:enumeration value="tns:InvalidCardinality"/>
         <xs:enumeration value="tns:MissingAddressInEPR"/>
         <xs:enumeration value="tns:DuplicateMessageID"/>
         <xs:enumeration value="tns:ActionMismatch"/>
         <xs:enumeration value="tns:MessageAddressingHeaderRequired"/>
         <xs:enumeration value="tns:DestinationUnreachable"/>
         <xs:enumeration value="tns:ActionNotSupported"/>
         <xs:enumeration value="tns:EndpointUnavailable"/>
      </xs:restriction>
   </xs:simpleType>

   <xs:element name="RetryAfter" type="tns:AttributedUnsignedLongType"/>
   <xs:complexType name="AttributedUnsignedLongType" mixed="false">
      <xs:simpleContent>
         <xs:extension base="xs:unsignedLong">
            <xs:anyAttribute namespace="##other" processContents="lax"/>
         </xs:extension>
      </xs:simpleContent>
   </xs:complexType>

   <xs:element name="ProblemHeaderQName" type="tns:AttributedQNameType"/>
   <xs:complexType name="AttributedQNameType" mixed="false">
      <xs:simpleContent>
         <xs:extension base="xs:QName">
            <xs:anyAttribute namespace="##other" processContents="lax"/>
         </xs:extension>
      </xs:simpleContent>
   </xs:complexType>

   <xs:element name="ProblemIRI" type="tns:AttributedURIType"/>

   <xs:element name="ProblemAction" type="tns:ProblemActionType"/>
   <xs:complexType name="ProblemActionType" mixed="false">
      <xs:sequence>
         <xs:element ref="tns:Action" minOccurs="0"/>
         <xs:element name="SoapAction" minOccurs="0" type="xs:anyURI"/>
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
   </xs:complexType>

</xs:schema>
```

3.5 Serialization Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:tns="http://schemas.microsoft.com/2003/10/Serialization/"
attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/2003/10/Serialization/"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:annotation>
    <xs:documentation>
  THE SCHEMA IS PROVIDED TO YOU ON AN "AS IS" BASIS, AND MICROSOFT
  DISCLAIMS ALL WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING,
  WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
  FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT, AS TO THE SCHEMA OR ANY
  PRODUCT OR OTHER ITEM THAT MAY BE DEVELOPED USING THE SCHEMA.

  Without limiting the generality of the foregoing, Microsoft makes no
  warranty that any product or other item that may be developed using the
  schema, or any portion of the schema, will not infringe any copyright,
  patent, trade secret or other intellectual property right of any
  individual or legal entity in any country. It is your responsibility to
  obtain licenses to use any such intellectual property rights as appropriate.

  MICROSOFT IS NOT LIABLE FOR ANY DAMAGES OF ANY KIND ARISING OUT OF OR IN
  CONNECTION WITH THE USE OF THE SCHEMA, INCLUDING, WITHOUT LIMITATION, ANY
  DIRECT, INDIRECT, INCIDENTAL, CONSEQUENTIAL (INCLUDING LOST REVENUES OR LOST
  PROFITS), PUNITIVE OR SPECIAL DAMAGES, WHETHER OR NOT MICROSOFT HAS BEEN
  ADVISED OF SUCH DAMAGES.

  (c) Microsoft Corporation.  All rights reserved.
</xs:documentation>
  </xs:annotation>
  <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:schema>
```

3.6 Serialization Arrays Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:tns="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:annotation>
    <xs:documentation>
  THE SCHEMA IS PROVIDED TO YOU ON AN "AS IS" BASIS, AND MICROSOFT
  DISCLAIMS ALL WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING,
  WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
  FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT, AS TO THE SCHEMA OR ANY
  PRODUCT OR OTHER ITEM THAT MAY BE DEVELOPED USING THE SCHEMA.
```

```
    Without limiting the generality of the foregoing, Microsoft makes no
    warranty that any product or other item that may be developed using the
    schema, or any portion of the schema, will not infringe any copyright,
    patent, trade secret or other intellectual property right of any
    individual or legal entity in any country. It is your responsibility to
    obtain licenses to use any such intellectual property rights as appropriate.

    MICROSOFT IS NOT LIABLE FOR ANY DAMAGES OF ANY KIND ARISING OUT OF OR IN
    CONNECTION WITH THE USE OF THE SCHEMA, INCLUDING, WITHOUT LIMITATION, ANY
    DIRECT, INDIRECT, INCIDENTAL, CONSEQUENTIAL (INCLUDING LOST REVENUES OR LOST
    PROFITS), PUNITIVE OR SPECIAL DAMAGES, WHETHER OR NOT MICROSOFT HAS BEEN
    ADVISED OF SUCH DAMAGES.

    (c) Microsoft Corporation.  All rights reserved.
  </xs:documentation>
    </xs:annotation>
    <xs:complexType name="ArrayOfanyURI">
      <xs:sequence>
        <xs:element minOccurs="0" maxOccurs="unbounded" name="anyURI" nillable="true"
type="xs:anyURI" />
      </xs:sequence>
    </xs:complexType>
    <xs:element name="ArrayOfanyURI" nillable="true" type="tns:ArrayOfanyURI" />
</xs:schema>
```

3.7 Type Design Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
xmlns:tns0="http://schemas.microsoft.com/ComplexEventProcessing/2011/10/Design"
elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/ComplexEventProcessing/2011/10/Design"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns="http://schemas.microsoft.com/ComplexEventProcessing/2011/10/Design"
xmlns:linqexpression="http://schemas.microsoft.com/ComplexEventProcessing/2011/10/LinqExpress
ion">
  <xs:import
namespace="http://schemas.microsoft.com/ComplexEventProcessing/2011/10/LinqExpression"
schemaLocation="schemas.microsoft.com.ComplexEventProcessing.2011.10.LinqExpression.xsd"/>
  <xs:complexType name="TypeRoot">
    <xs:sequence>
      <xs:element name="Types" type="linqexpression:ArrayOfType"/>
      <xs:element name="Assemblies" type="linqexpression:ArrayOfAssembly"/>
      <xs:element name="Definition" type="linqexpression:TypeReference"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="Type">
    <xs:complexContent mixed="false">
      <xs:extension base="linqexpression:Type">
        <xs:sequence>
          <xs:element minOccurs="0" name="Properties" type="tns0:ArrayOfMember"/>
          <xs:element minOccurs="0" name="Fields" type="tns0:ArrayOfMember"/>
          <xs:element name="IsValueType" type="xs:boolean"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="ArrayOfMember">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="Member" type="tns0:Member"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="Member">
    <xs:sequence>
      <xs:element name="Name" type="xs:string"/>
      <xs:element name="Type" type="linqexpression:TypeReference"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

3.8 Management Service Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
xmlns:tns="http://schemas.datacontract.org/2004/07/Microsoft.ComplexEventProcessing.Managemen
tService" elementFormDefault="qualified"
targetNamespace="http://schemas.datacontract.org/2004/07/Microsoft.ComplexEventProcessing.Man
agementService" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType name="TypeIdentifier">
    <xs:sequence>
      <xs:element minOccurs="0" name="Name" nillable="true" type="xs:string" />
      <xs:element minOccurs="0" name="AssemblyName" nillable="true" type="xs:string" />
    </xs:sequence>
  </xs:complexType>
  <xs:element name="TypeIdentifier" nillable="true" type="tns:TypeIdentifier" />
  <xs:complexType name="ArrayOfTypeIdentifier">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="TypeIdentifier" nillable="true"
type="tns:TypeIdentifier" />
    </xs:sequence>
  </xs:complexType>
  <xs:element name="ArrayOfTypeIdentifier" nillable="true" type="tns:ArrayOfTypeIdentifier"
/>
</xs:schema>
```

4 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.

- Microsoft SQL Server 2008
- Microsoft SQL Server 2008 R2
- Microsoft SQL Server 2012
- Microsoft SQL Server 2014
- Microsoft SQL Server 2016
- Microsoft SQL Server 2017

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.

<1> Section 2.2.2.1.1.1.1: In Microsoft SQL Server, except for SQL Server 2008, the server object is always set to the string "cep:/".

<2> Section 2.2.2.1.2.1.1: In SQL Server, except for SQL Server 2008, the separator character that is used to form the URI is always the forward slash (/) character. This character separates the parent object name and the child object name within the URI.

<3> Section 2.2.3.2.1.5.1: In Microsoft implementations, the enumeration values map to a subset of the **Microsoft.System** namespace for .NET Framework. For more information, see [MSDN-SysName].

5 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as ~~New,~~ Major, Minor, ~~Editorial,~~ or ~~No change~~None.

~~The revision class **New** means that a new document is being released.~~

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements~~ or functionality~~.
- ~~The removal of a~~A document ~~from the documentation set~~revision that captures changes to protocol functionality.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class ~~**Editorial** means that the formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.~~

~~The revision class **No change**~~**None** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the relevant technical content~~ of the document~~ is identical to the last released version.

~~Major and minor changes can be described further using the following change types:~~

- ~~New content added.~~
- ~~Content updated.~~
- ~~Content removed.~~
- ~~New product behavior note added.~~
- ~~Product behavior note updated.~~
- ~~Product behavior note removed.~~
- ~~New protocol syntax added.~~
- ~~Protocol syntax updated.~~
- ~~Protocol syntax removed.~~
- ~~New content added due to protocol revision.~~
- ~~Content updated due to protocol revision.~~
- ~~Content removed due to protocol revision.~~
- ~~New protocol syntax added due to protocol revision.~~
- ~~Protocol syntax updated due to protocol revision.~~
- ~~Protocol syntax removed due to protocol revision.~~
- ~~Obsolete document removed.~~

~~Editorial changes are always classified with the change type **Editorially updated**.~~

~~Some important terms used in the change type descriptions are defined as follows:~~

- ~~**Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.~~
- ~~**Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.~~

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

Section	~~**Tracking number (if applicable) and description**~~**Description**	~~**Major change (Y or N)**~~**Revision class**	~~**Change type**~~
~~2.2.3.2.1.1.2.1.1 AdvanceTimeGenerateType~~	~~Moved contents of the AdvanceTimeEventCountFrequencyType, AdvanceTimeDurationFrequencyType, AdvanceTimeDelayType, and AdvanceToInfinityType sections into section 2.2.3.2.1.1.2.1.1..~~	~~Y~~	~~Content update.~~
4 Appendix B: Product Behavior	Added SQL Server ~~2016~~2017 to the product applicability list~~ of applicable products~~.	~~Y~~Major	~~Content update.~~
~~5 Change Tracking~~	~~Removed section 2.2.3.2.1.1.2.1.1.1 and moved contents to section 2.2.3.2.1.1.2.1.1.~~	~~Y~~	~~Content removed.~~
~~5 Change Tracking~~	~~Removed section 2.2.3.2.1.1.2.1.1.2 and moved contents to section 2.2.3.2.1.1.2.1.1.~~	~~Y~~	~~Content removed.~~
~~5 Change Tracking~~	~~Removed section 2.2.3.2.1.1.2.1.1.3 and moved contents to section 2.2.3.2.1.1.2.1.1.~~	~~Y~~	~~Content removed.~~
~~5 Change Tracking~~	~~Removed section 2.2.3.2.1.1.2.1.1.4 and moved contents to section 2.2.3.2.1.1.2.1.1.~~	~~Y~~	~~Content removed.~~

6 Index