XYZ Data System

Content Delivery Service

Web Services API Guide

Issue: 1.0

Date of Issue: 5 May 2006

Author: Jim Foley

Contents

This is a sample API guide for a mock web service that provides, for instance, weather data.

1 ItemService

1.1 Overview

ItemService retrieves current snapshots of content from the XYZ cache. It supports only one operation, RetrieveItem.

1.2 RetrieveItem operation

The RetrieveItem operation retrieves the data specified in an item request message. It may specify the requested quality of service (e.g., realtime or delayed), and may specify a list of desired fields.

1.2.1 Request

The RetrieveItem_Request consists entirely of one or more item requests, each represented by a separate ItemRequest. An ItemRequest uses one or more "request keys" to uniquely identify the desired data. Other features of the request, as shown in the diagram below, are optional.

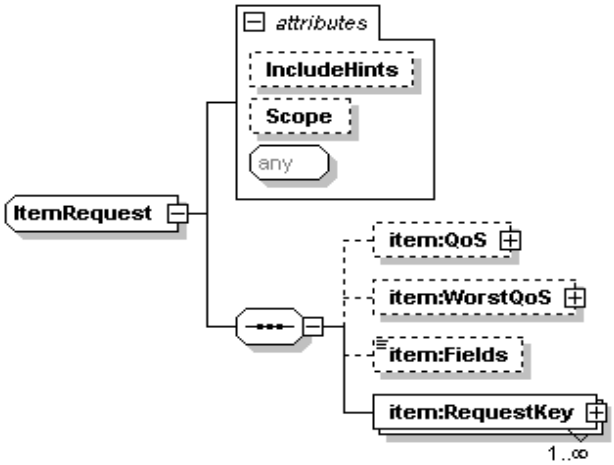

Figure 1: ItemRequest structure. Dotted lines indicate optional elements.

Table 1: ItemRequest elements and attributes

Element	Description
RequestKey	Specifies a particular cache item to retrieve. RequestKey is of type ItemKey, described in Figure 2.
Scope	If set to "List", then only the fields listed in Fields are returned. If set to "All", then all fields are returned and Fields is ignored. The value "AllExcept" is not supported in the current version. Optional; the default value is "List".

Fields	A string containing the names of the desired fields, separated by a colon (no spaces). This set of fields will be returned for each RequestKey in the ItemRequest. The service permits the specification of an additional namespace and user-defined data type in this string.
QOS	The desired Quality Of Service, i.e., realtime or delayed. See Figure 3 for more information. Optional.
WorstQOS	Specifies the minimal acceptable Quality Of Service. See Figure 3 for more information. Optional.
IncludeHints	Set this attribute to True to receive hints on how to display the data. Hints are explained under the Field structure later in this chapter. Optional.

The RequestKey (an element of type ItemKey) specifies the desired cache item to retrieve. As shown in Figure 2, it must provide an item name (e.g., "us_cn_9reg" for current midwest regional weather), may specify a data source other than the default feed, and may specify the name type (e.g., "NOAA" or "ISO"). A RequestKey is also used in the response message, to identify the actual data item returned.

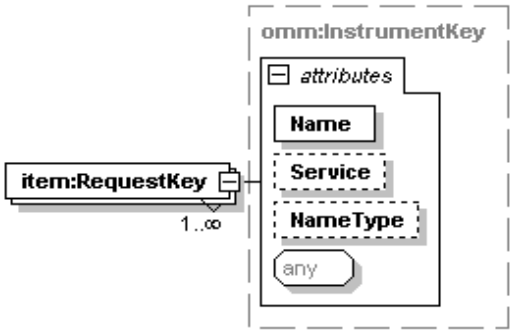

Figure 2: ItemKey type (used by RequestKey)

The QOS and WorstQOS are of type QOS, shown in Figure 3. Note that the same structure is used to identify the QOS in the response message. The two primary features of QOS are TimelinessInfo and RateInfo.

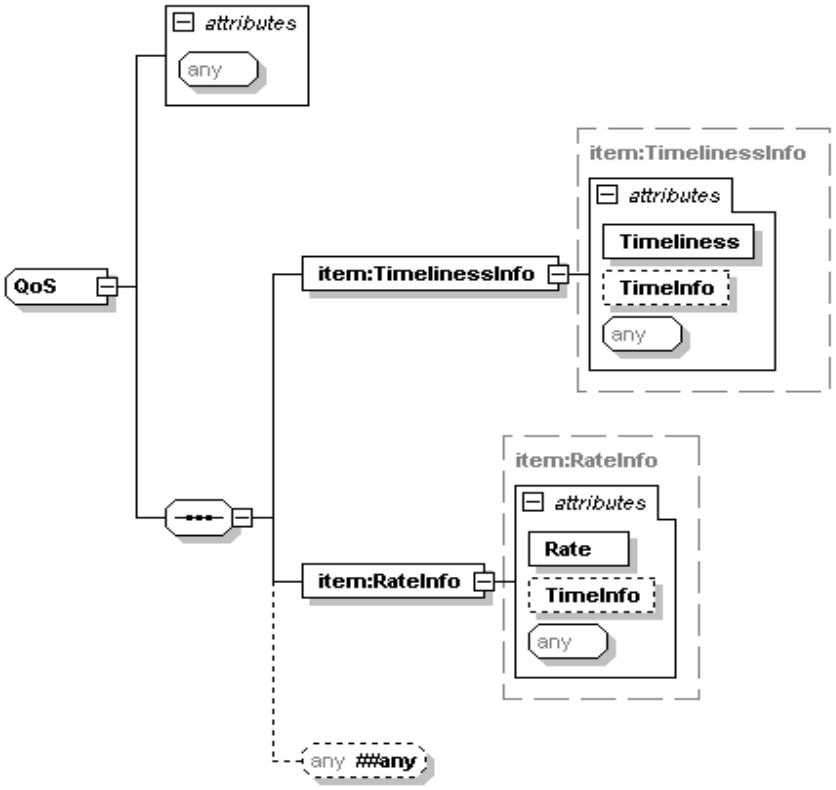

Figure 3: QOS

TimelinessInfo addresses whether a delay is applied to the data. It contains the Timeliness setting, and optionally the TimeInfo (the length of the delay).

Table 2: Values for Timeliness

Value	Description
"REALTIME"	No delay is applied to the data.
"DELAYED"	The data is delayed by the number of whole seconds specified in the TimeInfo attribute.
"DELAYED_UNKNOWN"	A fixed delay (such as 60 seconds) is applied to the data. Valid only in responses, not requests.
"UNSPECIFIED"	In QoS, use this value to request the most timely data allowed by the client's permissions. In WorstQoS, use this value to indicate that any timeliness is acceptable.

Rate addresses the rate at which the data is being updated on the server.

Table 3: Values for Rate

Value	Description
"TICK_BY_TICK"	The data is updated on the server with every tick.

"JIT_CONFLATED"	"Just in time" conflation is applied to the data before it reached the server's cache.
"TIME_CONFLATED"	Time-based conflation is applied to the data before it reaches the server's cache, in cycles of a duration specified in milliseconds in TimeInfo.
"PERIODIC_REFRESH"	The data is updated on the server by a periodic refresh mechanism (e.g., a "pull"), at periods specified in milliseconds in TimeInfo.
"UNSPECIFIED"	In QoS, this value corresponds with data being updated as often as possible. In WorstQoS, this value indicates that the update method is not critical.

1.2.2 Response

The response message contains one ItemResponse for every ItemRequest that was contained in the request message. Each ItemResponse contains one or more Item elements. Each Item element includes the requested set of fields for a single cache item.

The following diagram shows the general structure of the ItemResponse, including the Item structure. Within the Item structure, the RequestKey and QOS elements should be familiar; they are of the same type as their counterparts in the ItemRequest. Other Item components are the Status, Fields, and ChildItem.

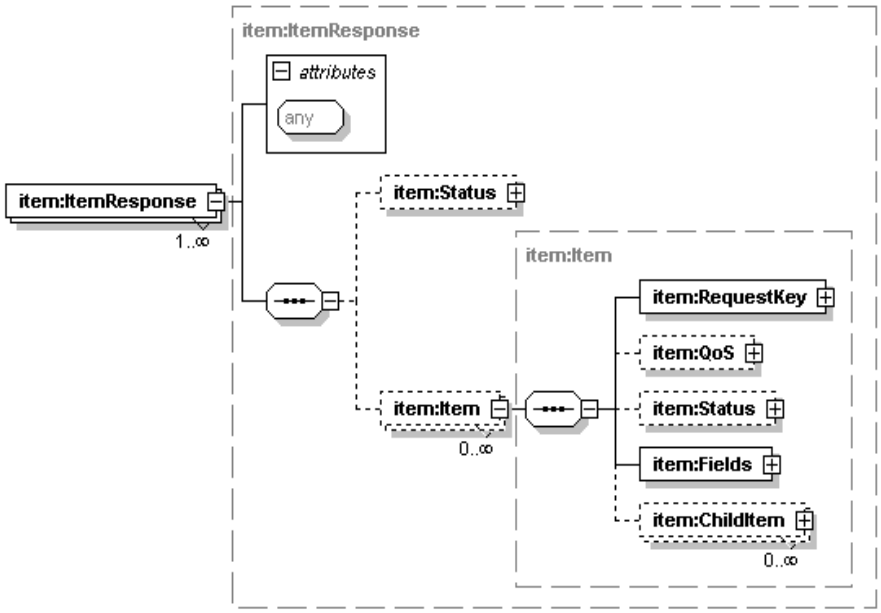

Figure 4: ItemResponse

An Item always contains a RequestKey, which identifies the cache item (as seen in Figure 2), and a Fields structure, which contains the requested data. Here are the optional elements of the Item:

- QOS, if present, indicates the actual quality of service of the Item data. See Figure 3 for more information on QOS.
- Status, if present, contains a status message or a message from the data source. Status is discussed later in this section.
- ChildItem contains a constituent item, and is present only if the requested item is a chain (such as the Dow Jones Industrials or a list of the day's top gainers). ChildItem is discussed later in this section.

The Fields structure (not the same as Fields in the data request) is a collection of Field elements, each representing a data field for an item from the XYZ cache. The Field structure is described in Table 4.

Table 4: Field structure

Value	Description
Name	The name of the field, e.g., "IMAGE_TIME", "IMAGE_SIZE", or "IMAGE".
DataType	A string indicating the data type. Supported types are: "Int8", "UInt8", "Int16", "UInt16", "Int32", "UInt32", "Int64", "UInt64", "Float", "Double", "Date", "Time", "DateTime", "Utf8String", "RmtesString", "AsciiString", "Binary".
Hint	A hint is an integer that specifies a suggested display format – often the industry-standard format for the given data. Valid hints include: **Display Format** / **Hint Value** Integer: 0 String: 64 Fraction: 1 through 8, indicating one of the following base fractions: 1, 1/2, 1/4, 1/16, 1/32, 1/128, 1/256 Decimal: 16 through 25, indicating the number of decimal places where 16 indicates zero decimal places, 17 indicates one decimal place, and so on up to nine decimal places. Date as MM DDD YY: 258 Time as 00:00: 259 Time as 00:00:00: 260 For instance, given "1.119" as a data value, the hint "16" should result in a display of "1", the hint "17" should result in a display of "1.1", and the hint "18" should result in a display of "1.12".
(value)	The field value is formatted as one of the above-mentioned data types so that it can be extracted in a strongly typed manner.

Child Items

Figure 5 shows the ChildItem structure within the Item. It can be seen that ChildItem structure is analogous to that of Item. The differences are that ChildItem does not include a QOS, and instead of using a RequestKey to identify an item, it uses a Name attribute.

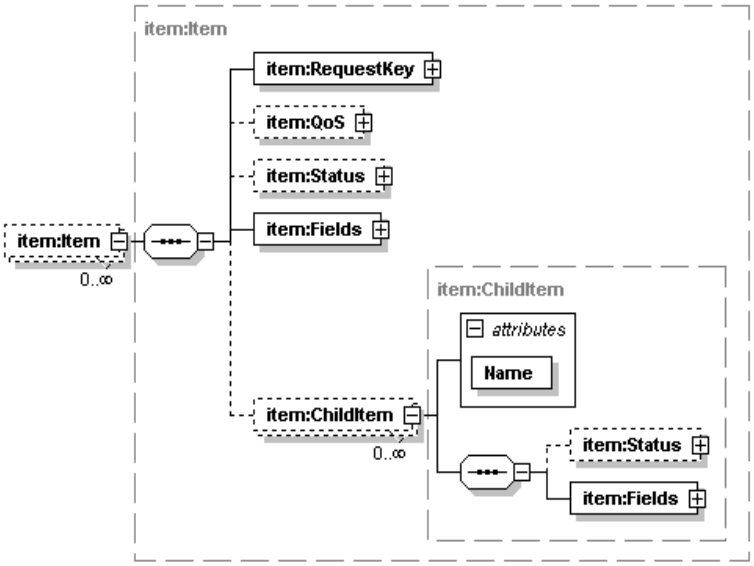

Figure 5: Item structure including ChildItem

If the client requests a symbol that represents a chain of related records, then the returned Item contains zero or more ChildItem elements – one for each constituent of the specified chain. The sequence of a chain is often important, so the chain data in the Fields structure preserves this sequence. Note that chains only nest to one level of depth. If it happens that one of the chain's constituents is itself a chain, the returned data will not include that second-level chain's constituents – they must be retrieved in a subsequent request.

Status

A Status may appear at three levels in the ItemResponse:

- in the ItemResponse node for global status, e.g., service down
- in the Item node for the status of a requested item, e.g., invalid symbol
- in the ChildItem node for the status of a constituent, e.g., temporarily unavailable

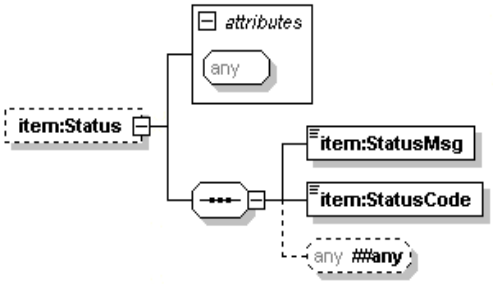

Figure 6: Status

The Status element, if present, contains a StatusMsg and a corresponding integer StatusCode. It may also contain other attributes or elements , which may convey additional

messages from the data source – such as a detailed error message, back-end error ID, or both. Most of the status messages are self-explanatory, as can be seen in this table.

Table 5: `Status` messages

Value	Description
"ACCESS_DENIED"	Access denied, possibly due to insufficient permissions.
"ITEM_TEMP_UNAVAILABLE"	Item temporarily unavailable.
"ITEM_UNKNOWN"	Unknown item or invalid symbol.
"NO_RESPONSE"	No response from upstream data source.
"OK"	Success.
"SERVICE_DOWN"	Service down.
"SERVICE_UNKNOWN"	Unknown service.
"STALE"	Data may be stale.

2 TimeSeriesService

2.1 Overview

The TimeSeriesService web service provides access to time series data and supporting data, returned as a static dataset. TimeSeriesService supports three operations, each structured as a web-service request and response using SOAP over HTTP. The three operations correspond to the three types of data provided by the service:

- RetrieveTimeSeriesData
- RetrieveResourceData
- RetrieveResourcelistData

The term "time series" refers to a series of data points for a given period of time, such as a series of temperature readings during the course of a day, or average temperatures for a year. This data might be used, for instance, for performing analysis or for generating a graphic or tabular display.

Requests for time series data can specify:

- a standard region ID
- desired data fields
- date and time range
- maximum number of data points
- interval between data points

Additionally, the API supports a "realtime mode" that provides every data point on record within the specified region and time period, subject to specified filters. Note that this API does not provide live updates. For streaming updates, the separate WeatherStream service must be used. Some applications use TimeSeriesService to create an initial display, which is then updated with live data using WeatherStream.

"Resource" data includes a variety of system data and background data that does not change on a daily basis. System data includes information about the time series service, such as codes for preconfigured filters. Background data includes information on specific fields, such as style.

Requests for "resource lists" return a variety of lists of resource-related data. For instance, where RetrieveResourceData allows an application to look up the data associated with a particular region, RetrieveResourcelistData might be used to request a list of all regions, with associated details such as short and long name, local timezone code(s), etc. Similarly, RetrieveResourcelistData can be used to look up details related to the value of a particular item found in the resource data.

2.2 RetrieveTimeSeriesData

This operation retrieves time series weather data. Requests can specify a region, data source, desired data fields, date and time range, maximum number of data points, and the desired interval between data points.

2.2.1 Request

A TimeSeries_Request message contains one TimeSeriesRequest, which must specify the region, a summarization interval for the data points, and one or more flags for basic parameters. Other request elements are optional.

TimeSeriesRequest

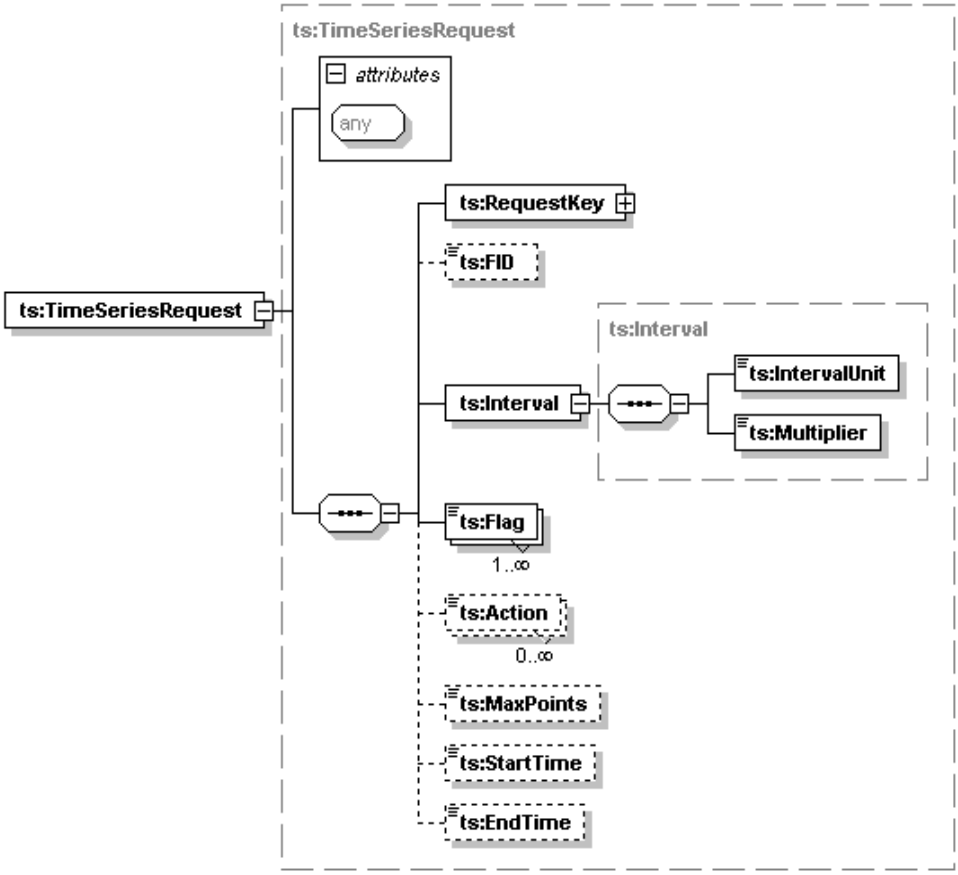

Figure 7: TimeSeriesRequest. Dotted lines indicate optional elements.

Table 6: Features of the TimeSeriesRequest

Element or Attribute	Description

RequestKey	Identifies the region for which data is requested. RequestKey is of type ItemKey, described in Figure 2. It must provide a name (e.g., "us_cn_9reg"), may specify a data source other than the default feed, and may specify the name type (e.g., "NOAA" or "ISO").
FID	Integer that identifies a FID (field ID). Optional; omitting this element or setting it to "0" activates the default FID. A "FID" here is a data view, an associated set of table columns called "actions", and a specific field to which the summarization interval applies. Sample FID names include: "GMT_DATE", "LOCAL_DATE", "TEMPERATURE", "BAR_PRESSURE", etc. To discover all the FID's available on a server, issue a Resourcelist request for "HISTORICAL_FID" or "REALTIME_FID". To discover all the FID's supported for a particular region and data source (recommended), issue a resource request with classification "RIC" and resource type "FIDLIST".
Interval	Specifies the data intervals, i.e., the length of time or number of ticks between successive data summarization points in the data set. Interval contains two elements: IntervalUnit, which specifies a unit such as a tick or a period of time, and Multiplier, which specifies how many time units should elapse between successive data points. The options are: **IntervalUnit** — **Supported values for Multiplier** "TICK" — 1 - 1000 "SECOND" — 1 - 3600 (up to one hour) "MINUTE" — 1 - 1440 (up to one day) "DAY" — 1 - 366 (up to one year) "WEEK" — 1 - 52 (up to one year) "MONTH" — 1 - 36 (up to three years) "SEASON" — 1 - 400 (up to 100 years) "YEAR" — 1 - 4000 (to earliest weather data on source)
Flag	One or more flags must be set to indicate certain retrieval options to the server. For more information, see Table 7.
Action	"Actions" are the columns in a table of time series data. The Action element is used to specify desired actions from within a standard set supported by the FID. For summarized data, the following actions are normally available: "DAY" (daytime average), "NIGHT", "DAILY" (24-hour average), "HIGH", "LOW", "RECORD". For tick data, the following actions are normally available: "AVERAGE", "HIGH", "LOW". Optional.
MaxPoints	Maximum number of data points to return. Optional.
StartTime	Starting GMT time value for the data set; all datapoints must occur at or after this moment. Used only if a Flag is set to "USE_START". This value is a standard xsd:dateTime. Optional. If not specified, the earliest data point is equal to the time of the request minus the amount of time specified in the Interval.
EndTime	Final GMT time value for the data set; all data points must occur at or before this moment. Used only if a Flag is set to "USE_END". This value is a standard xsd:dateTime. Optional. If not specified, the time of the request is used.

Table 7: Flags

Flag	Description
"ALLOW_EMPTY"	If there is no data for a timeseries, the request will receive an empty table. This is useful if the client application needs to know a list of the actions available for the timeseries when no data is available.

"DO_NOT_CACHE "	Suggests that the server should not cache the data, typically because it is a very large or unusual data set that may bump other, more popular data from the cache.
"INCLUDE_IRREGULAR"	The server should include data points that may be flagged in the system as anomalous. The default is to filter those points out.
"REFRESH_CACHE"	Bypasses cached data in any intervening servers and goes to the authoritative back-end source. Depending on the configuration, this may give slower screen responses but more current data. Since this flag forces a refresh of the requested data in all caches, some applications may periodically issue a request with this flag simply to force a periodic cache refresh on the server.
"USE_END"	If USE_START and USE_END are present, the server will send all points between the StartTime and EndTime parameters, including the endpoints. Setting an EndTime that is earlier than the StartTime causes the dataset to be sequenced in reverse chronological order.
"USE_POINTS"	The server will obey the MaxPoints parameter and send no more than the specified number of datapoints. There are four variations: 1. If neither USE_START nor USE_END is specified: the server will return the most recent *n* points in the timeseries, in chronological order, where *n* is the MaxPoints value. 2. If USE_START is specified: the server will return the first *n* points after and including the given start time, in chronological order, where *n* is the MaxPoints value. 3. If USE_END is specified: the server will return the last *n* points before and including the given end time in date/time reverse chronological order, where *n* is the MaxPoints value. 4. If USE_START and USE_END are specified: the server will return *n* points bounded by the StartTime and EndTime, where *n* is the MaxPoints value. If StartTime is set to be later than EndTime, the data is arranged in reverse chronological order.
"USE_START"	If USE_START and USE_END are present, the server will send all points between the StartTime and EndTime parameters, including the endpoints. Setting an EndTime that is earlier than the StartTime causes the dataset to be sequenced in reverse chronological order.

2.2.2 Response

The TimeSeries_Response message contains a single TimeSeriesResponse, of type TSResponse as described in Section 0. The response always contains a status, and if data is successfully retrieved will contain a data table.

TSResponse

The TSResponse structure, shown in Figure 8, is used in the TimeSeriesResponse.

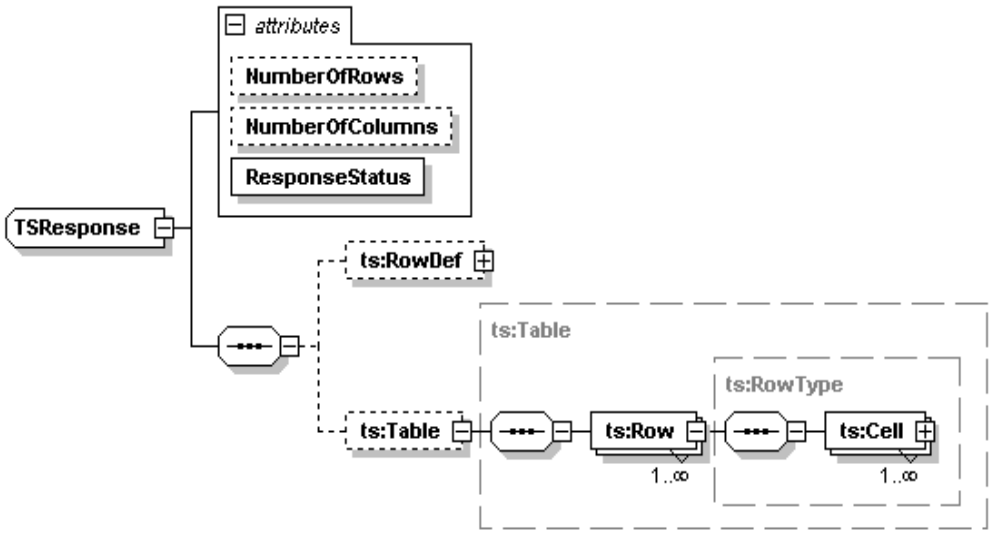

Figure 8: TSResponse (used by TimeSeriesResponse)

ResponseStatus is described in section 0. The table comprises one or more rows, each comprising one or more cells. The number of rows and the number of columns (i.e., the number of cells per row) are includes as TSResponse attributes. Rows consists of one or more cells, and are defined in the separate element RowDef as a sequence of cell definitions, as shown in Figure 9.

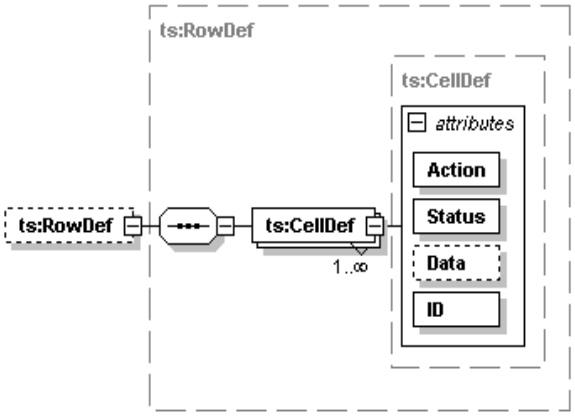

Figure 9: Row & cell definitions

One CellDef is returned for each Action that was specified in the data request. If none were specified, then one CellDef is returned for each action supported by the FID. A CellDef contains the name of the field (Action), a status, an optional data type (Data), and an ID that refers back to the cell definitions.

Table 8: Attributes of the CellDef

Attribute	Description
Action	Identifies the fieldname of the data.

Status	Contains either "Valid" or "Invalid". "Invalid" typically indicates that the requested field is not supported for the requested FID.
Data	Indicates the data type, enabling the application to extract data from the response in a strongly-typed manner. Values include: "Int8", "UInt8", "Int16", "UInt16", "Int32", "UInt32", "Int64", "UInt64", "Float", "Double", "Date", "Time", "DateTime", "Utf8String", "RmtesString", "AsciiString", "Binary".
ID	An integer that uniquely identifies the cell definition. The ID is referenced in the actual cell data to indicate the cell type. For example, a CellDef has ID=9 and Action =”AVERAGE”, then a Cell with ID=9 contains data for the action AVERAGE.

As seen in Figure 8, the data is presented as a table having one or more rows. Each instance of Row contains one or more instances of Cell. The number of cells per row is reflected in the NumberOfColumns attribute of TSResponse. This number is typically equal to the number of actions supported by the FID, or the number of actions specified in the request (assuming that all those fields are supported by the FID).

The structure of Cell is very simple. Cell extends the simple type DataBuffer, which contains one strongly-typed data item, by adding an ID as an attribute. The ID is an integer that refers to the CellDef that corresponds with the action that this cell represents.

For example, a request might specify five actions, of which it turns out that only the first four are actually supported by the FID. In that case, the RowDef would include five CellDef elements, with the integers 0 through 4 as ID's. The fifth CellDef in this example would have a Status of "Invalid". Since the last action is not supported by the specified FID, each row of the table data will include only four cells, with ID's ranging from 0 to 3.

ResponseStatus

In TimeSeriesService, responses always include a ResponseStatus containing one of these values:

Table 9: Status messages

Status	Description
OK	Success
ERR_INVALIDMESSAGE	Invalid request
ERR_MAXREGIONS	Maximum number of Regions exceeded
ERR_NORESOURCES	No system resources available to process the request
ERR_NOSUCHACTION	Invalid action (i.e., field) requested
ERR_NOSUCHCLIENT	Invalid client
ERR_NOSUCHFEED	Invalid server specified
ERR_NOSUCHFID	Invalid FID specified
ERR_NOSUCHINTERVAL	Invalid interval specified
ERR_NOSUCHREGION	Invalid Region specified

ERR_OUTOFRANGE	Value out of range

2.3 RetrieveResourceData

The RetrieveResourceData operation retrieves information about "resources" – relatively stable information entities within the system, such as the server configuration or background data on weather regions and data fields.

Etc.

2.3.1 Request

A Resource_Request contains a single ResourceRequest, which is required only to specify the resource classification, described in section XXX. All other elements of the request are optional, as described in section XXX and section XXX.

Etc.

2.3.2 Response

The Resource_Response contains a single ResourceResponse, which contains a status and collection of zero or more elements appropriate to the request. For instance, a request for a Region resource typically returns the region's local timezone, units, etc. If no data is returned, the ResponseStatus may identify the reason. (ResponseStatus is described in section 0 – "ResponseStatus".)

Etc.

2.4 RetrieveResourcelistData

RetrieveResourcelistData is used to request resource list information. Where the RetrieveResourceData operation retrieves a collection of data items connected to a single resource (e.g., information about a specific region or event type), RetrieveResourcelistData retrieves a list of a given type of resource (e.g., a list of regions or event types).

Etc.

2.4.1 Request

A Resourcelist_Request contains a single ResourcelistRequest, which must specify a resource classification and one or more masks. Supported classifications include: "TIMEZONE", "UNIT". etc. Masks, which specify a list of desired fields, are listed in table XXX. The request may also specify a server feed (e.g., "XYZ_SELECTFEED") if the default feed is not desired.

Etc.

2.4.2 Response

The Resourcelist_Response contains a single ResourcelistResponse, which contains a status and collection of zero or more elements appropriate to the request. If no data is returned, the ResponseStatus may identify the reason. (ResponseStatus is described in section 0 – "ResponseStatus".)

Etc.

Appendix 1 – AuthUserIdentity

The SOAP header of any request must include an AuthIdentity, which specifies to an XYZ web service the user identity that exists in the Auth system. AuthIdentity is of type AuthIdentityType, from AuthUserIdentity.xsd, as shown in Figure 10.

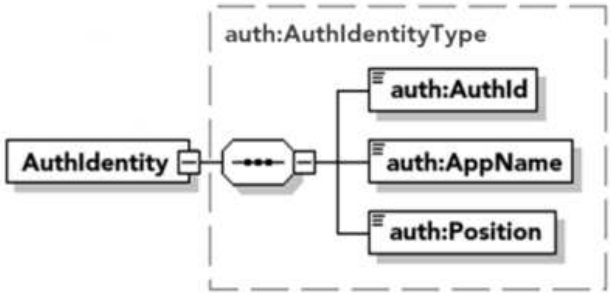

Figure 10:AuthIdentity

Table 10: Features of the AuthIdentity

Element	Description
AuthId	The Auth identity of the user requesting the data. The service confirms that the user exists in the Auth system, then determines whether the user is allowed access to the requested data.
AppName	The name of the application that the user is using to request the data. This value is provided for utilization logging.
Position	A string representing the unique location of this user. Commonly this is derived from a network address, but it can be any unique value representing the workstation. Multiple user requests that present the same Position string will be considered a single user. The Auth system can be configured to limit the number of unique positions that a user may utilize.

Appendix 2 – SOAP Faults

If a SOAP fault occurs, the SOAP response contains a Fault element with the following structure:

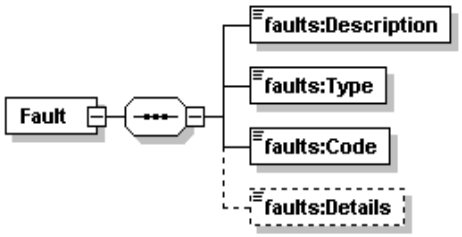

Figure 11: Fault

Details of the SOAP fault are as follows:

Table 11: Elements of the SOAP fault

Element	Description
Description	A brief description of the error, e.g., "Invalid Value for Element: Classification".
Type	Possible values for this element are described in Table 12.
Code	Contains one of two possible values: "SERVER" - Indicates the request could not be processed because of a processing error, not because of a problem with the message itself. "CLIENT" - Indicates the client's request was not formed correctly or did not contain the necessary information.
Details	Optional. Contains details about the problem, such as a stacktrace. Example: java.lang.NullPointerException Com.weathersource.itemservice.JaxProcessor (245) Com.weathersource.itemservice.JaxProcessor (451)

Table 12: Types of SOAP fault

Type	Description
"XML_PARSE_ERROR"	There was an error de-serializing a SOAP request, or the request syntax was rejected by the server.
"INIT_ERROR"	The server failed to start or has a configuration problem.
"BACKEND_UNAVAILABLE"	The backend server is not available. This is not a fatal error. The web service will continue to attempt connection until a session is established.

"BAD_CREDENTIALS"	Valid user credentials were not supplied with the request.
"NOT_AUTHORIZED"	The user account is not authorized for the requested content.
"UNSPECIFIED_ERROR"	The error is not one of the error types specified above.