Market Data Markup Language™ (MDML™) Specification
Version 0.12

Working Draft 03 May, 2000

Bridge Information Systems, Inc.

MDML SPECIFICATION NOTICE

By obtaining, using, and/or reproducing the MDML Specification, you (the licensee) agree that you have read, understood, and will comply with the following terms and conditions:

1. You are granted (a) a world-wide, royalty-free, non-exclusive, perpetual license to use and reproduce the MDML Specification and (b) the right to use new properties with the Market Data Master Property Set (Section 27) in the MDML Specification. All other licensable rights, including but not limited to the right to prepare derivative works, are excluded from this license.

2. All copies of the MDML Specification or portions thereof shall include the full text of this NOTICE.

3. Any written or printed use of the Market Data Markup Language trademark or the MDML trademark shall include the following notice: "Market Data Markup Language and MDML are trademarks of Bridge Information Systems, Inc."

4. THIS MDML SPECIFICATION IS PROVIDED "AS IS". BRIDGE INFORMATION SYSTEMS, INC. MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, (A) WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE AND (B) THAT THE USE OR REPRODUCTION OF THE MDML SPECIFICATION WILL NOT INFRINGE ANY THIRD PARTY PATENT, COPYRIGHT, OR OTHER RIGHTS. BRIDGE INFORMATION SYSTEMS, INC. SHALL NOT BE RESPONSIBLE OR LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OR LOSS ARISING OUT OF ANY USE OR REPRODUCTION OF THE MDML SPECIFICATION. ALL USE AND REPRODUCTION OF THE MDML SPECIFICATION SHALL BE AT THE RISK AND PERIL OF THE LICENSEE.

5. No revision, modification, or alteration of these terms and conditions shall be valid unless made in a writing that specifically purports to do so, that is signed by an authorized representative of Bridge Information Systems, Inc., and that is signed by an authorized representative of the licensee.

6. These terms and conditions represent the complete agreement regarding the MDML Specification. If any of these terms or conditions are held to be invalid, illegal, or unenforceable, the validity, legality, and enforceability of the remaining terms and conditions shall not in any way be affected or impaired thereby.

7. The performance, construction, and interpretation of these terms and conditions shall be governed by the laws of the State of Missouri in the United States of America, without regard to the application of its conflict of laws. Bridge Information Systems, Inc. and the licensee agree to exclusive jurisdiction in a state or federal court in St. Louis, Missouri or in the Eastern District of Missouri.

1 Introduction

This document describes an implementation of XML to be used for distributing financial information. A primary goal of this implementation is to be flexible enough to carry data from any financial market data vendor, including, but not limited to, exchanges, brokerage houses, banks, and information vendors like Bridge. While all market data vendors carry largely overlapping sets of data, they have all developed different data models, protocols and symbologies. Each vendor supplies a highly specialized application set for making the best use of their data. Extracting information from a vendor's data feed into common desktop applications can only be done with custom software. A class of products called integration platforms has been developed to ease this problem. Integration platforms convert vendor data into their own proprietary models so their specialized application set can make use of data from many vendors. While this solves the immediate problem of a trader who wants access to multiple sources of data through a single desktop, it doesn't solve the larger problem of distributing information to disparate systems in an organization. The solution to this problem lies in the use of widely accepted standard protocols and data models. XML and its related standards from the W3C have experienced unprecedented support from all major software vendors, which makes it an ideal candidate for developing the interfaces required.

Most if not all market data vendors have solved the problem of data modeling by developing simple models which allow them to carry an ever increasing variety of information without making changes to their infrastructure or simple display applications. There are two broad classes of representation, which here will be referred to as *pages* and *records*. Pages are unstructured collections of data formatted for display in a fixed sized window. A record is a set of *properties* each containing a piece of information related to that record. Records representing different types of securities (i.e. Japanese Government Bonds and Dollar Yen Swaps) will contain different sets of properties. Even when vendors' wire protocol represents information in fixed sized structures, it is still conceptually sets of properties. Pages themselves can be, and often are, distributed as sets of properties - containing the page's width, height, text, color attributes in separate properties. Other types of information can be represented as sets of properties, but tend to have more complex request parameters, and usually contain repeating rows of data. Examples are news headlines, historical trades, and options and futures chains.

In order to represent this information in XML, MDML defines a property and its attributes, as well as conventions for assembling properties into the objects that are returned from queries on a vendor's data feed. MDML also defines a convention for representing repeating rows of data. In order to improve access to the information, MDML defines a default set of object types with default properties, as well as request structures for them. The set of object types, and their properties, is extensible so a market data vendor can include its own value-added information and differentiate its products.

This specification includes examples of MDML, but not schemas or DTDs. Examples are much better at conveying the structure and use of an XML. Once MDML is finalized, schemas and/or DTDs will be published, separate from this document, to allow parsers to validate MDML.

2 Scope

MDML is attempting to solve a narrow range of problems specific to the delivery and representation of real-time market data. There are a few services a deployment of MDML should take advantage of in order

to build a robust and secure market data delivery service. These services have been left out of MDML itself because there are industry standard implementations available.

Image and Update Synchronization: Images and updates are often handled differently in market data delivery systems. Images are generally destined for a unique client, and are always less time sensitive than updates. Images can be cached and kept up to date at different layers of a network infrastructure. Updates are generally delivered to a large number of clients simultaneously. Updates are extremely time sensitive. The delivery of updates can be decoupled from that of images by bypassing image caches, choosing a different transport (e.g. multicast vs. point to point), prioritizing updates over images, or all three. The result of these strategies is that images and updates become unsynchronized during transport, and must be re-synchronized at their destination. MDML assumes that images and updates are always delivered in the proper sequence.

Authentication: In any market data delivery system, a user must establish his or her identity so that the system can determine what information the user is allowed to see. MDML assumes that this is done outside its domain.

Permissioning: A market data delivery system must ensure that only properly permissioned information is presented to the user. MDML facilitates this by delivering permissioning information, but it does not do the permissioning itself. This subject is covered in more detail in Section 11.

Compression: Most delivery of updating market data across wide area networks faces bandwidth limitations.
The use of XML for market data delivery exacerbates the problem due to its inherent verbosity. Systems using MDML would benefit by using standard data compression technologies or XML tokenization to reduce its bandwidth usage.

Encryption: Market data is generally time sensitive proprietary information. When this type of information is transmitted over public networks, it should be encrypted to reduce the chance that it is read or forged by unauthorized people. A system sending MDML over a public network must encrypt the data itself.

Internationalization: MDML will be used to deliver content in many languages. It relies on XML's use of UTF-8 encoding, coupled with the XML `xml:lang` attribute to carry this content.

3 Property

A property is the basic element of information in a response. A response is built up from individual properties and sets of properties. A property has several attributes that can be used to interpret and display its data. Roughly speaking, these are broken into value, display, and label type attributes. The value and display attributes are needed because of the nature of financial price data. Many securities are traded not in decimal values, but in fractional values such as eighths and 32nds. Over time, conventions have evolved in different markets for displaying fractions. A property carries enough information to allow for calculations, simple fractional display, and market targeted price display.

Each vendor defines the universe of properties that can be delivered with a vehicle. The vendors regard the names, types and contents of the properties as proprietary. For this reason, it is important to ensure that the set of properties is completely extensible. At the same time, application developers can benefit greatly if they know about specific properties. This specification includes a basic master set of properties (Master Property Set, see Appendix 1). Vendors should provide mappings from their proprietary properties into the

Master Property Set whenever possible. If these are readily available, it is more likely that application developers will not create one-off inconsistent mappings.

There may be more than one value for a property, for example, when a vendor delivers a list of market maker identifiers. In these cases, the property is repeated in the response object. To be updated, repeating properties must be uniquely identifiable. This is accomplished with one of two attributes. In cases where properties are not ordered or re-ordered by the server, the properties are identified by an "id" attribute. The "id" attribute is a natural identifier, such as market maker id in a list of market maker quotes, or some other arbitrary string value. In cases where the server orders properties, the properties include a "row" attribute. The "row" attribute contains a numerical value specifying the position of the property in the list. If properties are inserted or deleted from the list, the "row" attributes of other properties in the list may change.

3.1 Property Types

Properties in MDML can be categorized into a small set of market data types. These types are used to assist in decoding and interpretation of properties. The following tables describe the data types used to encode MDML property values, and the market data types that can be assigned to properties.

Data Type	**Description**
bool	1=true, 0=false
date	Date in IS0 8601 subset. Precision can be reduced by omitting smaller values. Examples: 1999-12-25, 2000-6
int	A number, with optional sign, no fractions, no exponent.
r8	A number, with no limit on digits, optional sign, optional fractional digits, optional exponent. Examples: 31, 2.2, -22.89E+5
string	A UTF-8 text string
time	Time in ISO 8601 subset. All times are GMT. Precision can be reduced by omitting smaller values. Example: 15:30:26

Table 3.1: Data Types used in MDML

Property Type	**Data Type**	**Description**
bool	bool	1=true, 0=false
change	r8	Contains a change from a previous value, such as the change from yesterday's close to the last price
date	date	Date in IS0 8601 subset. Precision can be reduced by omitting smaller values. Examples: 1999-12-25, 2000-6
int	int	Signed integer value
percentage	r8	Contains a percent value
price	r8	A currency or monetary value

real	r8	A real value that isn't a monetary value
size	int	Contains a size, such as a bid size
time	time	Time in ISO 8601 subset. All times are GMT. Precision can be reduced by omitting smaller values. Example: 15:30:26
text	string	Text
volume	int	Contains a volume, such as trade volume

Table 3.2: MDML Property Types

3.2 Simple Display

A property is identified by a 'name' attribute. For simple display, the value of a property is its display string.

```
<property name="last">42 3/8</property>
```

3.3 Calculation

A property contains an attribute called 'value'; the value of this attribute is the decimal value of the property. A companion 'type' property contains the property type, such as price, change, or date.

```
<property name="last" type="price" value="42.375">42 3/8</property>
```

3.4 Display

The 'displayHint' attribute categorizes a price value so an application can better display the price. With the value and displayHint attributes, an application has enough information to construct a specialized display value, selecting fractional fonts if necessary.

```
<property name="last" type="price" value="42.375" displayHint="reducible64">42 3/8</property>
```

3.5 Label

Property names may not be appropriate display labels. Therefore properties include attributes that contain displayable property labels to be used to identify the data. A long and short label is included so an application can better tailor its display.

```
<property name="last" longDisplay="Last" shortDisplay="ls" type="price" value="42.375"
                                        displayHint="reducible64">42 3/8</property>
```

4 Property Sets

In some responses properties are grouped into repeating sets. In these cases, a convention of including these into an element is used. As with repeating properties, repeating property sets should have an id or row attribute to uniquely identify them.

```
<setName id="id1">
        <property name="name">value</property>
        <property name="name1">value1</property>
```

```
...
</setName>
<setName id="id2">
        <property name="name">value'</property>
        <property name="name1">value1'</property>
 ...
</setName>
...
```

5 Markup Namespace

MDML tags are defined to be in the MDML namespace, which is uri:xml.com.bridge/mdml-1.0. Appendix 2, *Market Data Control Set* lists the elements and attributes that are used by MDML.

6 Request

Market data vendors provide varying methods for specifying a request for data. Some vendors encode all parameters in a string, while other vendors define protocol structures to be filled. Each vendor has different sets of request modifiers available to reflect their varying capabilities. Vendors must be free to add new requests without requiring infrastructure changes in the software modules responsible for transporting the requests. MDML defines request elements to allow structured requests, but it also recognizes that a vendor's request syntax may have all the information packed into a vehicle symbol.

Query elements contain the elements and attributes specifying the data to be returned in a request. A request contains one or more query elements. Request elements and the query elements they contain are specific to the type of information being requested. They are not generic collections.

All request elements share some attributes and elements that serve to modify the style of the request and response, but not the information query. These are

- updating,
- useMasterProperties,
- Properties.

The updating attribute is set to 'T' when a client wants to receive updates. The default is no updates (snapshot).

The useMasterProperties attribute is set to 'T' when a client would like to have source specific properties mapped into a set of universally understood properties. This allows a client to look for specific fields without knowledge of the different names assigned by the market data vendors. The default is 'F', meaning do not map source specific properties into Master Field Set properties.

The Properties element is a collection of properties a client wishes to receive in a response. When the Properties element is empty, a default list of properties is returned. This is the most reliable way to discover what properties are available from a system for a given query. The Properties element contains boolean attributes indicating which property attributes should be included with a response's properties. The default behavior is to include no property attributes.

Query elements share some or all of the following attributes

- symbol,
- symbology,

- source,
- userAttribute.

The symbol attribute is the vendor's identifier or symbol for the vehicle. There is no default value.

The symbology attribute is used if the vendor allows the use of alternate symbologies in a request. The default value is "native".

Source is used if information is coming through an integration platform, which may have multiple vendor feeds connected to it. There is no default value.

UserAttribute can contain any legal attribute string. User attribute can be used as a tag to match requests with responses; this removes the need to look up by multiple query attributes. There is no default value.

7 Response Types

An image response contains a full set of properties for the requested object. These properties supersede any existing properties, which must be discarded. An image will be sent as the initial data response to a request. If the request is updated, an image may be sent at any time to bring all properties up to date. An update response contains a partial set of properties, along with an operator describing how the properties should be interpreted. Updates are described in detail in the next session. An error response contains information about the status of the request. Errors are described in a later section.

8 Update

To enable a market data client to keep its version of a data object up to date, a server can send responses containing new data along with instructions on how to apply the new data to the existing cache. The response data includes an "op" attribute indicating how the data should be interpreted. Update types are "replace", "add", "insert", and "delete". The default operation is "replace". A simple example is shown here; more detailed examples are shown in each of the request type sections. This example shows how the bid and ask properties of a vehicle called "IBM" are updated.

```
<VehicleUpdate op="replace" symbol="IBM" …>
        <property name="bid">42</property>
        <property name="ask">42 1/4</property>
</VehicleUpdate>
```

Replace means, "Replace matching properties in the existing cache with the properties in the update." If no matching properties are found, the properties in the update should be added to the existing cache. This is the default update operation, to be used if the "op" attribute is not present.

Add is used to add properties to the existing cache, without first checking for matching properties. It is used in cases such as news headlines, where updates continually add to the cache.

The "insert" instruction is supported in request types where the upstream system maintains ordering. In these cases, the repeating property or property set will have an attribute named "row", with the zero-based row number. Row is set in initial and refresh images, and must be maintained by the client. When a client receives an update with <Insert>, it should increment by one the "row" attribute of each member of the repeating property or property set where "row" is greater or equal to the update's "row". The update should then be inserted into the cache in front of the first node that was adjusted.

The “delete” instruction means “delete matching properties in the existing cache.” In request types where “insert” is supported, the “row” attribute of the nodes following the deleted node should be decremented by one.

For example, to insert a new order between the first and second:

The current cache contains:

```
<OrderBook>
        <Order row="0">
                <property name="bid">42</property>
        </Order>
        <Order row="1">
                <property name="bid">42 1/8</property>
        </Order>
</OrderBook>
```

Update with:

```
<OrderBookUpdate>
        <Order op="insert" row="1">
                <property name="bid">42 1/16</property>
        </Order>
</OrderBookUpdate>
```

Resulting cache:

```
<OrderBook>
        <Order row="0">
                <property name="bid">42</property>
        </Order>
        <Order row="1">
                <property name="bid">42 1/16</property>
        </Order>
        <Order row="2">
                <property name="bid">42 1/8</property>
        </Order>
</OrderBook>
```

9 Status

In MDML, status information is delivered in attributes in the image and update messages. Any element in a response can contain status attributes. The status of an element applies to all elements below it in the hierarchy where no status is specified. This allows an MDML server to deliver status information on queries, parts of responses, or individual properties.

The following attributes can be contained in any MDML response element:

- status
- message

The status attribute can contain a value of “ok”, “permanent”, or “temporary”. When an element has no status element, it takes on the status of its parent, up to a query response element, such as Vehicle or Page. The default status for query response elements is “ok”.

Status	Description
ok	Information in this element is current.
permanent	Information in this element is stale and will not automatically recover. No further information will be delivered for this element.
temporary	Information in this element is stale and will automatically recover.

The message attribute contains a string that should be displayed to the user. It can be used to deliver an explanation for a fault status. It can also be used to deliver the informative messages that are often sent by data feeds.

If a query response contains any status information anywhere in its hierarchy, it must include the hasStatus attribute with a value of “true”. If a query response contains messages anywhere in its hierarchy, it must include the hasMessage attribute with a value of “true”.

10 Cancel

A client can send a cancel message at any time to stop the flow of updates on a query, or to inform the server it is no longer interested in receiving information it requested. The server should not send any more responses to a cancelled query. The client must be prepared to receive more responses on a query, as there could be messages in transit when it sends a cancel.

A cancel message contains the queries a client is no longer interested in.

11 Permissioning

Market data vendors require that access to their data is controllable. Users who have not paid for access to classes of data must not be able to view or use it. Data objects are assigned to one or more groups. Users who have been given access to a group can see any data assigned to that group. Vendors combine groups into packages or products, which are what they generally sell to users. Systems which provide user access to such data are required to compare each user’s access rights to an object’s group membership list, and deny access when a user does not have permission to see that object.

In most cases the data object is the thing returned from a query, such as the Dollar-Yen cross rate. And in most cases, the object will belong to a single permissioning group. A user with access rights for that group is entitled to see the object and all its properties. However, there are some cases where an object can belong to multiple permissioning groups. MDML allows for two mechanisms for including objects in multiple permissioning groups. When these are not sufficient for an object, that object can contain one or more properties to help define its permissioning parameters. These would most likely be vendor specific.

The two models for including data objects in multiple groups can be called ‘flat’ and ‘hierarchical’. In a flat model, a requested object is simply assigned to more than one group. If a user has access rights to any of those groups, he or she should be allowed to see all properties in that object. In a hierarchical model,

permissions group membership is assigned not only to a requested object, but also to properties and structures within that object. A user must have access rights to each node of the tree that was traversed to find a property in order to use or view that property.

News headlines permissioning is a common example of the flat model. In the following example, the headline belongs to permission groups 42, 86, 212, and 85. If the user can access any of these groups, he or she can see the headline.

```
<Headline perm="42 86 212 85">
        <property name="newsTime"/>
        <property name="date"/>
        <property name="text"/>
        …
</Headline>
```

The hierarchical model is required for property level permissioning, where a user has access rights to some, but not all properties in a data object. In the following example, some properties belong to an additional permissioning group. A user must have access rights to group 64 *and* 361 to see the secondBid and </property> properties. Some market data may have levels of permissioning between requested object and property. In this case, the principle is the same. The user must be granted access to each level of the hierarchy before he or she can drill down to the next level.

```
<Vehicle perm="64">
        <property name="bid">33</property>
        <property name="ask">33 1/4</property>
        <property name="secondBid" perm="361">32 7/8</property>
        <property name="secondAsk" perm="361">33 1/8</</property>>
</Vehicle>
```

12 Vehicle

The vast majority of requestable objects in a market data vendor's database are simple vehicles. While these are generally populated by a set of properties in a flat hierarchy, at least one vendor delivers its data in a structured format with repeating structures.

The contents of a response to a vehicle request are determined by the properties contained in the request. Multiple instances of a property may be returned. Properties may be grouped into repeating sets, to make it easier to display groups in a tabular display. Properties will not appear in different sets because the property name space is flat, not hierarchical.

There are three pre-defined sets of repeating properties. ExchangeBlock is used when a Vehicle can contain price data from multiple exchanges. MarketMakerBlock is used when a Vehicle can contain price quotes from multiple market makers. DividendHistoryBlock is used when a Vehicle can contain a series of historical dividends.

12.1 Request Format

```
<VehicleRequest updating="T F" useMasterProperties"T F">
        <Vehicle source="" symbology="native SEDOL CUSIP ISIN"
                        symbol="" userAttribute""/>
```

```
        <Vehicle source="" symbology="native SEDOL CUSIP ISIN"
                        symbol="" userAttribute""/>
        <Properties longDisplay="T F" shortDisplay="T F" displayHint="T F">
                <property name="property1"/>
                <property name="property2"/>
        </Properties>
</VehicleRequest>
```

12.2 Cancel Format

```
<VehicleCancel>
        <Vehicle source="" symbology="native SEDOL CUSIP ISIN"
                        symbol="" userAttribute""/>
        <Vehicle source="" symbology="native SEDOL CUSIP ISIN"
                        symbol="" userAttribute""/>
</VehicleCancel>
```

12.3 Image Format

The first form shows a vehicle with no repeating property sets.

```
<Vehicle source="" symbology="native" symbol="IBM"
                                userAttribute"" perm="">
        <property name="symbol"/>
        <property name="description"/>
        <property name="bid"/>
        <property name="ask"/>
        <property name="last"/>
        …
</Vehicle>
```

The second form shows a vehicle with a repeating property set.

```
<Vehicle source="" symbology="" symbol="" userAttribute"" perm="">
        <property name="symbol"/>
        …
        <ExchangeBlock perm="">
                <ExchangePrice id="">
                        <property name="exchange"/>
                        <property name="netChange"/>
                        <property name="bid"/>
                        …
                </ExchangePrice>
                <ExchangePrice id="" perm="">
                        <property name="exchange"/>
                        <property name="netChange"/>
                        <property name="bid"/>
                        …
                </ExchangePrice>
                <ExchangePrice id="" perm="">
                        <property name="exchange"/>
```

```
                    <property name="netChange"/>
                    <property name="bid"/>
                    …
                </ExchangePrice>
        </ExchangeBlock>
</Vehicle>
```

12.4 Update Format

The first form shows updating simple properties.

```
<VehicleUpdate source="" symbology="" symbol="" userAttribute="">
<property name="ask"/>
        <property name="askSize"/>
        <property name="bid"/>
        <property name="bidSize"/>
</VehicleUpdate>
```

The next example shows an update of an object with repeating sets.

```
<VehicleUpdate source="" symbology="" symbol="" userAttribute="">
        <ExchangeBlock>
                <ExchangePrice id="">
                        <property name="exchange"/>
                        <property name="bid"/>
                        <property name="ask"/>
                        <property name="last"/>
                </ExchangePrice>
                <ExchangePrice id="">
                        <property name="exchange"/>
                        <property name="bid"/>
                        <property name="ask"/>
                        <property name="last"/>
                </ExchangePrice>
        </ExchangeBlock>
<VehicleUpdate>
```

13 Page

A page is a fixed size, formatted display of text, along with optional graphics commands. Any character on a page can have foreground & background color, and video attributes. Pages are often chained together, so pointers to the next and previous pages are included. Page graphics are included as a series of simple drawing commands included with the page. Graphics should be scaled per the pageGraphic xres and yres scale factors.

13.1 Request Format

```
 <PageRequest updating="" useMasterProperties"">
        <Page source="" symbol="" userAttribute"">
        <Page source="" symbol="" userAttribute"">
        <Properties longDisplay="T F" shortDisplay="T F" displayHint="T F">
```

```
                <property name="prop1"/>
                <property name="prop2"/>
        </Properties>
</PageRequest>
```

13.2 Cancel Format

```
<PageCancel>
        <Page source="" symbol="" userAttribute"">
        <Page source="" symbol="" userAttribute"">
</PageCancel>
```

13.3 Page Image and Update

A Page Image contains a information about a page, a list of text segments, and a list of page graphic commands. Unlike other images, a page image is not a cacheable version of a page. Instead, it is a transmission format. It contains a sequence of text and drawing commands which, when applied to a screen, will duplicate the display of the page source. With this method, it is straightforward for an end-user application to update a screen display. However, someone who wants to implement an intermediate cache for MDML pages will have to write their own caching functionality.

Since the MDML for a page is a transmission format, images and updates are largely the same, and can be processed the same way.

A TextSegment contains a segment of text starting at a 0-based row and column, with the given foreground and background colors, and the given attributes. Colors are given in RGB values. The mdp:attributes property contains a space separated list of attributes that should be applied to each character.

A PageGraphicCommand is an instruction to draw a single object. PageGraphicCommands contain x and y pixel coordinates. The origin (0,0) is in the upper left of the page; x increases to the right and y increases towards the bottom of the page. If the command is DRAW_RECT or FILL_RECT, then (x1, y1) represents the top left corner and (x2, y2) represents the bottom right corner of the rectangle. The output resolution of these coordinates is given in the xres and yres properties. Applications must scale the coordinates to match the resolution of their output device.

Graphics related properties will only be delivered in pages with graphic content. The remaining properties in the following definition form the suggested minimum set of properties that should be delivered in a page.

```
<Page source="" symbol="" userSymbol="" userAttribute"">
        <property name="symbol"/>
        <property name="rows"/>
        <property name="columns"/>
        <property name="previousPage"/>
        <property name="nextPage"/>
        <property name="xres"/>
        <property name="yres"/>
        <TextSegment>
                <property name="row"/>
                <property name="column"/>
                <property name="text"/>
                <property name="attributes">
```

```
                    BOLD MEDIUM REVERSE BLINK UNDERLINE DOUBLE_WIDE
                </property>
                <property name="fg"/>
                <property name="bg"/>
        </TextSegment>
        <PageGraphicCommand>
                <property name="command">DRAW_RECT DRAW_LINE FILL_RECT</property>
                <property name="color"/>
                <property name="x1"/>
                <property name="y1"/>
                <property name="x2"/>
                <property name="y2"/>
        </PageGraphicCommand>
</Page>
```

14 News Headline

A headline query returns a set of headlines matching the request criteria. Headlines are most often delivered as part of a single headline stream; in this case symbol may not be required.

To select a subset of the total headline database, a client may include a NewsMatch element, which is a news search tree, with the request. A NewsMatch contains basic boolean operators, and operands allowing a client to search for or filter on specific news parameters.

News stories can be delivered in a variety of formats. A requester can send a space separated list with each request enumerating the types of story formats it can accept. This list should be in order of preference, so that the first format is the most desired. The upstream system should return the first format on the list it can match.

If the storyBody property is included in Properties, each headline will be returned with its story. The storyBody property will contain the body of the story in a CTEXT block, formatted according to the storyBodyFormat property. The content can be extracted from the storyBody property and passed on to an appropriate decoder.

14.1 Request Format

```
<HeadlineListRequest updating="T F" useMasterProperties"T F"
                        storyBodyFormats="TEXT EXACTTEXT SGML XML HTML PDF">
        <HeadlineList source="" symbol="" userAttribute"">
                <NewsMatch>
                        (see below)
                </NewsMatch>
</HeadlineList>
        <Properties longDisplay="T F" shortDisplay="T F" displayHint="T F">
                        <property name="property1"/>
                        <property name="property2"/>
</HeadlineListRequest>
```

14.2 Cancel Format

```
<HeadlineListCancel>
```

```
        <HeadlineList source="" symbol="" userAttribute"">
                <NewsMatch>
                </NewsMatch>
        </HeadlineList>
</HeadlineListCancel>
```

14.3 Image Format

The properties listed in the headline element below form the suggested set of properties that should be delivered with each headline.

```
<HeadlineList source="" symbol="" userAttribute"">
        <NewsMatch>
                (see below)
        </NewsMatch>
        <headline>
                <property name="newsDate"/>
                <property name="newsTime"/>
                <property name="sourceDate"/>
                <property name="sourceTime"/>
                <property name="issueDate"/>
                <property name="issueTime"/>
                <property name="releaseDate"/>
                <property name="releaseTime"/>
                <property name="expireDate"/>
                <property name="expireTime"/>
                <property name="categories"/>
                <property name="sourceCategories"/>
                <property name="newsSource"/>
                <property name="relatedSymbols"/>
                <property name="headlineBody"/>
                <property name="editorialMessage"/>
                <property name="editorialMessageAuthor"/>
                <property name="storyBodyFormat">
                                TEXT EXACTTEXT SGML XML HTML PDF</property>
                <property name="storyBody"/>
                <property name="reviewFlag"/>
                <property name="revision"/>
                <property name="storyId"/>
                <property name="storyLength"/>
                <property name="storyAuthor"/>
        </headline>
</HeadlineList>
```

14.4 Update Format

A client's headline cache will grow over time, as almost all updates will add to the cache.

```
<HeadlineListUpdate source="" symbol="" userAttribute"">
        <NewsMatch>
                (see below)
```

```
        </NewsMatch>
        <headline op="add">
                <property1/>      (same set as for HeadlineList)
                <property2/>
                …
        </headline>
</HeadlineListUpdate>
```

14.5 NewsMatch

An NewsMatch element is used to transport a boolean search or filter expression. This is used in filtering news headlines and submitting news headline searches. NewsMatch allows a client system to build an expression containing equality tests, ANDs, ORs, and NOTs, as well as time ranges. These can be arranged in a tree to allow a user to specify the precedence and grouping of the operations.

There are four operators: isOp, timeRangeOp, andOp, orOp, and notOp. The isOp operator performs a match, of a type specified in the match attribute, of the value in the value attribute. Valid types of matches are "source", "category", "keyword", "text" and "symbol". The timeRangeOp operator constrains a search to a time range. The remaining operators represent logical operations on the results of their child operators.

<isOp match="source category keyword text symbol" value=""> returns a headline or story if the story has source, category, keyword, text, or related symbol equal to the contents of the value attribute.

<timeRangeOp timeType="issue expire release story delete" startDate="" startTime="" endDate="" endTime=""> returns a headline or story if the story has an issue date, expire date, release date, story date or deleted date between the dates specified in the start and end attributes. The timeTypes correspond to the master properties with the same names.

Example: Return headlines where category is HITECH or BIOTECH, but not INTERNET, with keywords pikachu and charmeleon.

```
<andOp>
        <orOp>
                <isOp match="category" value="HITECH"/>
                <isOp match="category" value="BIOTECH"/>
        </orOp>
        <notOp>
                <isOp match="category" value="INTERNET"/>
        </notOp>
        <isOp match="keyword" value="pikachu"/>
        <isOp match="keyword" value="charmeleon"/>
</andOp>
```

Example: Return the same headlines above, released between Feb 16, 1998 and June 23, 1998.

```
<timeRangeOp releaseDate" start="1998-02-16" end="1998-06-23">
        <andOp>
                <orOp>
                        <isOp match="category" value="HITECH"/>
```

```
                    <isOp match="category" value="BIOTECH"/>
                </orOp>
                <notOp>
                    <isOp match="category" value="INTERNET"/>
                </notOp>
                <isOp match="keyword" value="pikachu"/>
                <isOp match="keyword" value="charmeleon"/>
        </andOp>
</timeRangeOp>
```

15 News Story

The News Story request allows a user to make a request for a specific story and its related information. The set of properties that can be requested is the same as for the News Headline Request.

15.1 Request Format

```
<StoryRequest useMasterProperties"T F"
                storyBodyFormats="TEXT EXACTTEXT SGML XML HTML PDF">
        <story source="" storyId="" userAttribute”"/>
        <Properties longDisplay="T F" shortDisplay="T F" displayHint="T F">
                <property name="property1"/>
                <property name="property2"/>
</StoryRequest>
```

15.2 Cancel Format

```
<StoryCancel>
        <story source="" storyId="" userAttribute”"/>
</StoryCancel>
```

15.3 Image Format

A story is delivered as a single headline element, which is described in the News Headline section.

```
<story source="" storyId="" userAttribute”">
        <headline/> (as defined in News Headline)
</story>
```

15.4 Update Format

News stories don’t have an update format, since no vendor supplies updating news stories.

16 History

This data is a series of data points going back in time. A time series of data points may contain each trade on a security, or it may contain summaries of time intervals from minutes to months. It is the most complicated of the types to use, because different intervals have different rules for interpretation. Requests are the same no matter what the interval, but different intervals return slightly different sets of properties.

History queries do not update.

The properties listed in the history property sets form the suggested sets of properties that should be delivered with each relevant history response.

16.1 Request Format

```
<HistoryRequest useMasterProperties"T F">
        <History source="" symbol="" symbology="native CUSIP SEDOL ISIN
                startDate="" startTime="" endDate="" endTime=""
                span="" interval="TICK MINUTE HOUR DAY WEEK MONTH YEAR"
                intervalSize=""/>
        <Properties longDisplay="T F" shortDisplay="T F" displayHint="T F">
                <property name="property1"/>
                <property name="property2"/>
        </Properties>
</HistoryRequest>
```

16.2 Cancel Format

```
<HistoryCancel>
        <History source="" symbol="" symbology="native CUSIP SEDOL ISIN
                startDate="" startTime="" endDate="" endTime=""
                span="" interval="TICK MINUTE HOUR DAY WEEK MONTH YEAR"
                intervalSize=""/>
</HistoryCancel>
```

16.3 History Image

The history image contains a repeating property set holding each of the points in the time series. The name of the set depends on what was requested, since each type of interval can contain a different set of properties. The image will also contain a set of properties relevant to the whole query.

```
<History source="" symbol="" symbology="native CUSIP SEDOL ISIN"
                        startDate="" startTime="" endDate="" endTime=""
                        span="" interval="TICK MINUTE HOUR DAY WEEK MONTH YEAR"
                        intervalSize"">
        <property name="symbol"/>
        <property name="exchange"/>
        <property name="timeZone"/>
        <Bar/>
        <DailyBar/>
        <QuoteBar/>
</History>
```

16.4 Bar

```
<Bar>
        <property name="open"/>
        <property name="high"/>
        <property name="low"/>
        <property name="close"/>
        <property name="barDate"/>
        <property name="barTime"/>
```

```
        <property name="volume"/>
        <property name="tickCount"/>
</Bar>
```

16.5 Daily Bar

```
<DailyBar>
        <property name="date"/>
        <property name="open"/>
        <property name="high"/>
        <property name="low"/>
        <property name="close"/>
        <property name="settle"/>
        <property name="openTime"/>
        <property name="highTime"/>
        <property name="lowTime"/>
        <property name="closeTime"/>
        <property name="settlementTime"/>
        <property name="openInterest"/>
        <property name="contractVolume"/>
</DailyBar>
```

16.6 Quote Bar

```
<QuoteBar>
        <property name="barDate"/>
        <property name="barTime"/>
        <property name="bid"/>
        <property name="ask"/>
        <property name="bidSize"/>
        <property name="askSize"/>
        <property name="conditions"/>
</QuoteBar>
```

17 Time & Sales

Time & Sales data is similar to History data. It is a view of all exchange events on a security.

Time & Sales requests do not update.

17.1 Request Format

```
<TimeAndSalesRequest useMasterProperties"T F">
        <TimeAndSales source="" symbology="native SEDOL CUSIP ISIN"
                symbol="" userAttribute""
                date="" relativeDay="" startTime="" endTime="" maxResponseCount=""
                trades="T F" quotes="T F">
                <Filter>vendor specific</Filter>
        </TimeAndSales>
</TimeAndSalesRequest>
```

17.2 Cancel Format

```
<TimeAndSalesCancel>
        <TimeAndSales source="" symbology="native SEDOL CUSIP ISIN"
                symbol="" userAttribute""
                date="" relativeDay="" startTime="" endTime="" maxResponseCount=""
                trades="T F" quotes="T F">
                <Filter>vendor specific</Filter>
        </TimeAndSales>
</TimeAndSalesCancel>
```

17.3 Image Format

```
<TimeAndSales source="" symbology="native SEDOL CUSIP ISIN"
                symbol="" userAttribute""
                date="" relativeDay="" startTime="" endTime="" maxResponseCount=""
                trades="T F" quotes="T F">
        <TimeAndSalesSet>
                (properties)
        </TimeAndSalesSet>
</TimeAndSales>
```

18 Market Maker List

Market Makers is the entire list of Buy and Sell “indications” for all the brokers in the market, one record for each broker.

18.1 Request Format

```
<MarketMakerListRequest updating="T F" useMasterProperties"T F">
                <MarketMakerList source="" symbology="native SEDOL CUSIP ISIN" symbol=""
                userAttribute""
                order="ALPHA BID BID_TOUCH BID_COMPRESS ASK ASK_TOUCH
                ASK_COMPRESS"/>
        <Properties longDisplay="T F" shortDisplay="T F" displayHint="T F">
                <property name="property1"/>
                <property name="property2"/>
        </Properties>
</MarketMakerListRequest>
```

18.2 Cancel Format

```
<MarketMakerListCancel>
                <MarketMakerList source="" symbology="native SEDOL CUSIP ISIN" symbol=""
                userAttribute""
                order="ALPHA BID BID_TOUCH BID_COMPRESS ASK ASK_TOUCH
                ASK_COMPRESS"/>
</MarketMakerListCancel>
```

18.3 Image Format

```
<MarketMakerList source="" symbology="native SEDOL CUSIP ISIN"
```

```
                symbol="" userAttribute""
                order="ALPHA BID BID_TOUCH BID_COMPRESS ASK ASK_TOUCH
                ASK_COMPRESS"/>
        <MarketMaker id="MADF">
                <property name="name">MADF</property>
                <property name="bid">116</property>
                <property name="ask">116 5/16</property>
        </MarketMaker>
        <MarketMaker id="TRIM">
                <property name="name">TRIM</property>
                <property name="bid">115 13/16</property>
                <property name="ask">116 1/16</property>
        </MarketMaker>
</MarketMakerList>
```

18.4 Update Format

```
<MarketMakerListUpdate source="" symbology="native SEDOL CUSIP ISIN"
                symbol="" userAttribute""
                order="ALPHA BID BID_TOUCH BID_COMPRESS ASK ASK_TOUCH
                ASK_COMPRESS"/>
        <MarketMaker id="MADF">
                <property name="bid">116 1/16</property>
                <property name="ask">116 3/8</property>
        </MarketMaker>
</MarketMakerListUpdate>
```

19 Order Book

Order Book provides a list of all open orders on an exchange.

19.1 Request Format

```
<OrderBookRequest updating="T F" useMasterProperties"T F">
        <OrderBook source="" userAttribute""/>
        <Properties longDisplay="T F" shortDisplay="T F" displayHint="T F">
                <property name="price"/>
                <property name="size"/>
                <property name="orderDate"/>
                <property name="orderTime"/>
                <property name="orderSeq"/>
                <property name="priceModifier"/>
                <property name="id"/>
                <property name="expireDate"/>
                <property name="expireTime"/>
        </Properties>
<OrderBookRequest>
```

19.2 Cancel Format

```
<OrderBookCancel>
```

```
        <OrderBook source="" userAttribute""/>
<OrderBookCancel>
```

19.3 Image Format

```
<OrderBook source="" userAttribute"">
        <Order row="0">
                <property name="bid">42</property>
        </Order>
        <Order row="1">
                <property name="bid">42 1/8</property>
        </Order>
        …
</OrderBook>
```

19.4 Update Format

Order Book updates replace, insert or delete order elements. Orders are identified by their row attribute. When an insert or delete is applied to a cache, the row attributes of existing order elements must be updated.

```
<OrderBookUpdate source="" userAttribute"">
        <Order op="insert" row="1">
                <property name="ask">42</property>
        </Order>
</OrderBookUpdate>
```

20 Market Depth

Market Depth is the summarized volume and market maker list at each available price point, typically split by Buy/Sell indicator. The market maker list can either be a count of market makers, or an actual list of market maker identifiers, usually dictated by the exchange. Market Depth can show either a single exchange's depth information, or all known depth information for the vehicle.

20.1 Request Format

To request Market Depth information from a single exchange, set the exchange="" attribute to the exchange identifier. To receive information from all known sources, omit exchange="".

```
<MarketDepthRequest updating="T F" useMasterProperties"T F">
        <MarketDepth source="" symbology="native SEDOL CUSIP ISIN"
                                symbol="" exchange="" userAttribute""/>
        <Properties longDisplay="T F" shortDisplay="T F" displayHint="T F">
                <property name="bid"/>
                <property name="ask"/>
        </Properties>
<mdcsn:MarketDepthRequest>
```

20.2 Cancel Format

```
<MarketDepthCancel>
        <MarketDepth source="" symbology="native SEDOL CUSIP ISIN"
```

```
                                    symbol="" exchange="" userAttribute""/>
<mdcsn:MarketDepthCancel>
```

20.3 Image Format

```
<MarketDepth source="" symbology="native SEDOL CUSIP ISIN"
                                    symbol="" exchange="" userAttribute""/>
        <ExchangeInfo>
                <property name="exchange"/>
                …
        </ExchangeInfo>
        <ExchangeInfo>
                <property name="exchange"/>
                …
        </ExchangeInfo>
        <MdRow row="">
                <property name="bid"/>
                <property name="marketMakerID"/>
                <property name="marketMakerID"/>
                …
        </MdRow>
        <MdRow row="">
                <property name="bid"/>
                <property name="marketMakerID"/>
                <property name="marketMakerID"/>
                …
        </MdRow>
        …
</MarketDepth>
```

20.4 Update Format

Market Depth updates replace, insert or delete MdRow elements. The elements are identified by their row attribute. When an insert or delete is applied to a cache, the row attributes of existing elements must be updated.

```
<MarketDepthUpdate source="" symbology="native SEDOL CUSIP ISIN"
                                    symbol="" exchange="" userAttribute""/>
        <MdRow row="">
                <property name="marketMakerID"/>
                <property name="marketMakerID"/>
                …
        </MdRow>
</MarketDepthUpdate>
```

21 Option List

This request returns a list of options on a security. The caller can elect to receive just a list of vehicle names, or the contents of the vehicles, depending on the properties requested. The list can be ordered by an upstream system, in which case, each Option element will have a row attribute to maintain the order.

21.1 Request Format

```
<OptionListRequest updating="T F" useMasterProperties"T F">
        <OptionList source="" symbology="native SEDOL CUSIP ISIN"
                        symbol="" userAttribute""/>
        <Properties longDisplay="T F" shortDisplay="T F" displayHint="T F">
                <property name="symbol"/>
                <property name="description"/>
                <property name="currency"/>
                <property name="peRatio"/>
                <property name="strike"/>
                <property name="maturityDate"/>
                <property name="bid"/>
                <property name="ask"/>
        </Properties>
</OptionListRequest>
```

21.2 Cancel Format

```
<OptionListCancel>
        <OptionList source="" symbology="native SEDOL CUSIP ISIN"
                        symbol="" userAttribute""/>
</OptionListCancel>
```

21.3 Image Format

```
<OptionList source="" symbology="native SEDOL CUSIP ISIN"
                        symbol="" userAttribute"">
        <Underlying>
                <property name="symbol"/>
                <property name="description"/>
                <property name="currency"/>
                <property name="peRatio"/>
        </Underlying>
        <Option id="">
                <property name="symbol"/>
                <property name="strike"/>
                <property name="maturityDate"/>
                <property name="bid"/>
                <property name="ask"/>
        </Option>
        <Option id="">
                <property name="symbol"/>
                <property name="strike"/>
                <property name="maturityDate"/>
                <property name="bid"/>
                <property name="ask"/>
        </Option>
</OptionList>
```

21.4 Update Format

Option Lists can be updated with replace, delete, add and insert.

```
<OptionListUpdate source="" symbology="native SEDOL CUSIP ISIN"
                        symbol="" userAttribute="">
        <Option id="">
                <property name="bid"/>
                <property name="ask"/>
        </Option>
</OptionListUpdate>
```

22 Future List

This request returns a list of futures on a security. The caller can elect to receive just a list of vehicle names, or the contents of the vehicles, depending on the properties requested. The list can be ordered by an upstream system, in which case, each Future element will have a row attribute to maintain the order.

22.1 Request Format

```
<FutureListRequest updating="T F" useMasterProperties"T F">
        <FutureList source="" symbology="native SEDOL CUSIP ISIN"
                        symbol="" userAttribute""/>
        <Properties longDisplay="T F" shortDisplay="T F" displayHint="T F">
                <property name="symbol"/>
                <property name="description"/>
                <property name="currency"/>
                <property name="property1"/>
                <property name="property2"/>
        </Properties>
</FutureListRequest>
```

22.2 Cancel Format

```
<FutureListCancel>
        <FutureList source="" symbology="native SEDOL CUSIP ISIN"
                        symbol="" userAttribute""/>
</FutureListCancel>
```

22.3 Image Format

```
<FutureList source="" symbology="native SEDOL CUSIP ISIN"
                        symbol="" userAttribute"">
        <Underlying>
                <property name="symbol"/>
                <property name="description"/>
                <property name="currency"/>
                <property name="peRatio"/>
        </Underlying>
        <Future id="">
                <property name="symbol"/>
                <property name="property1"/>
```

```
            <property name="property2"/>
        </Future>
        <Future id="">
            <property name="symbol"/>
            <property name="property1"/>
            <property name="property2"/>
        </future>
</FutureList>
```

22.4 Update Format

Future Lists can be updated with replace, delete, add and insert. If an insert or delete is applied to a cache when there is a row attribute on the Future elements, the row attributes of existing Future elements must be updated.

```
<FutureListUpdate source="" symbology="native SEDOL CUSIP ISIN"
            symbol="" userAttribute"">
        …
</FutureListUpdate>
```

23 Index List

This request returns a list of securities on an index such as the S&P500. The client can elect to receive just a list of vehicle names, or the contents of the vehicles, depending on the properties requested.

23.1 Request Format

```
<IndexListRequest updating="T F" useMasterProperties"T F">
        <IndexList source="" symbol="" userAttribute""/>
        <Properties longDisplay="T F" shortDisplay="T F" displayHint="T F">
            <property name="property1"/>
            <property name="property2"/>
        </Properties>
</IndexListRequest>
```

23.2 Cancel Format

```
<IndexListCancel>
        <IndexList source="" symbol="" userAttribute""/>
</IndexListCancel>
```

23.3 Image Format

```
<IndexList source="" symbol="" userAttribute"">
        <Index id="">
            <property name="symbol"/>
            <property name="property1"/>
            <property name="property2"/>
        </Index>
        <Index id="">
            <property name="symbol"/>
            <property name="property1"/>
```

```
            <property name="property2"/>
        </Index>
</IndexList>
```

23.4 Update Format

The constituents of an index list change infrequently. Most updates will be to index properties, but index elements can be added and deleted.

```
<IndexListUpdate source="" symbol="" userAttribute"">
        <Index id="">
            <property name="property1"/>
        </Index>
</IndexListUpdate>
```

24 Most Actives List

The Most Actives List shows the most active issues on an exchange, by number of trades or by volume. The client can elect to receive just a list of vehicle names, or the contents of the vehicles, depending on the properties requested.

24.1 Request Format

```
<MostActivesListRequest updating="T F" useMasterProperties"T F">
    <MostActivesList source="" exchange=""
                        mostActivesType="trades volume"  userAttribute""/>
    <Properties longDisplay="T F" shortDisplay="T F" displayHint="T F">
            <property name="property1"/>
            <property name="property2"/>
    </Properties>
</MostActivesListRequest>
```

24.2 Cancel Format

```
<MostActivesListCancel>
    <MostActivesList source="" exchange=""
                        mostActivesType="trades volume"  userAttribute""/>
</MostActivesListCancel>
```

24.3 Image Format

```
<MostActivesList source="" exchange=""
                        mostActivesType="trades volume" userAttribute""/>
        <mostActive id="">
            <property name="symbol"/>
            <property name="property1"/>
            <property name="property2"/>
        </mostActive>
        …
</MostActivesList>
```

24.4 Update Format

Most Actives can be replaced, deleted or added.

```
<MostActivesListUpdate source="" exchange=""
                mostActivesType="trades volume" userAttribute""/>
        <mostActive id="" op="delete"/>
        <mostActive id="" op="add">
                <property name="property1"/>
                <property name="property2"/>
        </mostActive>
</MostActivesListUpdate>
```

25 Symbol Search

A Symbol Search request allows a client to pass a set of search parameters to market data vendors with symbol searching capability. Symbol Search requests do not update.

Search parameters are contained in a set of isOp elements. Each isOp includes a match type and a value. Valid types of matches are “company”, “description”, “cusip”, “isin”, “sedol” and “native”. Matches are substring matches, which allows a client to one word of a description, or a partial native symbol, for example. When more than one isOp is included, returned symbols will match all of them – a logical AND operation.

25.1 Request Format

```
<SymbolSearchRequest useMasterProperties"T F">
        <SymbolSearch source="" userAttribute"">
                <isOp match="company description cusip isin sedol native" value=""/>
                …
        </SymbolSearch>
        <Properties longDisplay="T F" shortDisplay="T F" displayHint="T F">
                <property name="property1"/>
                <property name="property2"/>
        </Properties>
</SymbolSearchRequest>
```

25.2 Cancel Format

```
<SymbolSearchCancel>
        <SymbolSearch source="" userAttribute"">
                <Filter>vendor specific</Filter>
        </SymbolSearch>
</SymbolSearchCancel>
```

25.3 Image Format

```
<SymbolSearch source="" userAttribute"">
        <Filter>vendor specific</Filter>
        <symbolSet>
                <property name="property1"/>
                <property name="property2"/>
```

```
        </symbol>
        <symbolSet>
                <property name="property1"/>
                <property name="property2"/>
        </symbol>
</SymbolSearch>
```

26 Generic List

Not all symbol lists fall into the specific categories above. The Generic List is a way of providing access to symbol lists without changing MDML. The caller can elect to receive just a list of vehicle names, or the contents of the vehicles, depending on the properties requested. The list can be ordered by an upstream system, in which case, each Generic element will have a row attribute to maintain the order.

26.1 Request Format

```
< updating="T F" useMasterProperties"T F">
        <GenericList source="" symbology="native SEDOL CUSIP ISIN"
                        symbol="" userAttribute""/>
        <Properties longDisplay="T F" shortDisplay="T F" displayHint="T F">
                <property name="symbol"/>
                <property name="description"/>
                <property name="currency"/>
                <property name="peRatio"/>
                <property name="property1"/>
                <property name="property2"/>
        </Properties>
</GenericListRequest>
```

26.2 Cancel Format

```
<GenericListCancel>
        <GenericList source="" symbology="native SEDOL CUSIP ISIN"
                        symbol="" userAttribute""/>
</GenericListCancel>
```

26.3 Image Format

```
<GenericList source="" symbology="native SEDOL CUSIP ISIN"
                        symbol="" userAttribute"">
        <Generic id="">
                <property name="symbol"/>
                <property name="property1"/>
                <property name="property2"/>
        </Generic>
        <Generic id="">
                <property name="symbol"/>
                <property name="property1"/>
                <property name="property2"/>
        </Generic>
</GenericList>
```

26.4 Update Format

Generic Lists can be updated with replace, delete, add and insert. If an insert or delete is applied to a cache when there is a row attribute on the Generic elements, the row attributes of existing Generic elements must be updated.

```
<GenericListUpdate source="" symbology="native SEDOL CUSIP ISIN"
                   symbol="" userAttribute="">
        …
</GenericListUpdate>
```

1 Market Data Master Property Set

Name	Property Type	Short Display	Long Display	Description
bid	price	bid	Bid	The current best (highest) bid price
bidTime	time	btm	Bid Time	Time of bid
bidSize	size	bsz	Bid Size	Number of shares represented by bid
bidNetChange	change	bnc	Bid Net Change	Net change between yesterday's closing bid price and the current bid value
bidOpen	price	bop	Bid Open	Opening bid for the day
bidYield	real	by	Bid Yield	
ask	price	ask	Ask	The current best (lowest) ask price
askTime	time	atm	Ask Time	Time of the current ask
askSize	size	asz	Ask Size	Number of shares represented by ask
askOpen	price	aop	Ask Open	First ask of the day
askClose	price	acl	Ask Close	Closing ask price
askYield	real	ay	Ask Yield	
last	price	lp	Last	Last price or last trade, depending on the source
lastTime	time	ltm	Last Sale Time	Time of lastPrice
lastSize	size	lsz	Last Sale Size	Size of lastPrice

netChange	change	dac	Change for the day	Net change for the day from the previous trading day's closing price.
percentChange	percentage	pctc	Percent Change	Percentage change for the day from the previous trading day's closing price
lastDir	text	lsd	Last Sale Tick Direction	Last direction. Values are 0 for unchanged, u for up and d for down. The direction represents the change between this trade and the last price which altered the direction of trading for the security
open	price	opn	Open	Opening price.
openType	text	ot	Open Type	How open was set
high	price	hi	High	High for the day
low	price	lo	Low	Low for the day
close	price	cp	Closing	Closing price
yield	real	yld	Yield	Yield
mid	price	mid	Mid	Mid price
midClose	price	mdcl	Closing Mid Price	
kassaPrice	price	kp	Kassa Price	Kassa price. Valid only on German issues
openRange1	price	or	Opening Range 1	Opening Range price
openRange2	price	or2	Open Range 2	
closeRange1	price	cr	Close Range 1	Close Range
closeRange2	price	cr2	Close Range 2	
previousDayVolume	volume	pdvo	Previous Day Volume	Volume from the previous trading day.
volume	volume	vo	Volume for the Day	Trade Volume
session	number	sess	Session	Trading session. Values are 1 and 2

exchange	text	ex	Exchange	Indicates the exchange from which this data originates.
settle	price	se	Settlement	The settlement price
previousDaySettle	price	pdse	Previous Day Settlement	Settlement price from the previous trading day
settleDate	date	sedt	Settlement Date	Date of the price in settle
openInterest	number	tfoi	Total Futures Open Interest	Open interest for all contracts on this future
bondType	text	bndtype	Bond Type	Type of bond for corporate options
strike	price	spr	Strike Price	Option strike price
expirationDate	date	exdt	Expiration Date	
maturityDate	date	mat	Maturity Date	Maturity date for the vehicle
putCallIndicator	text	pc	Put/Call Indicator	
exDividendDate	date	PR.CD.Div	Dividend Exercise Date	Exercise date of most recent cash dividend
tradingBase	text	tbase	Trading Base	Represents the denominator in which this vehicle trades as a whole number. Examples are 2, 4, 8, 16, 32, 10, 100
currency	text	cur	Currency	The currency in which this security trades
beta	real	beta	Beta	Beta. A measure of volitity for a vehicle
annualHigh	price	yhi	Annual High	52 week high price
annualHighDate	date	yhid	Annual High Date	Date of the 52 week high
annualLow	price	ylo	Annual Low	52 Week low price
annualLowDate	date	ylod	Annual Low Date	Date of the 52 week low
rows	number	rs	Rows	The number of rows in this page

columns	number	cs	Columns	The number of columns in this page
previousPage	text	pp	Previous Page	The previous page link
nextPage	text	np	Next Page	The next page link
xres	number	xres	X Resolution	X resolution for page graphics coordinates. If your display has a different resolution, scale x1 and x2 to match your resolution.
yres	number	yres	Y Resolution	Y resolution for page graphics coordinates. If your display has a different resolution, scale y1 and y2 to match your display
row	number	row	Row	The page row this update applies to, base 0
column	number	col	Column	The page column this update applies to, base 0
text	text	txt	Text	The text of the page
attributes	text	attr	Attributes	The video attributes for the text on this page
fg	number	fg	Foreground Color	The default foreground color for the page
bg	number	bg	Background Color	The default background color of the page
command	text	cmd	Command	The graphic command
number	number	num	Number	
x1	number	x1	x1	The first x co-ordinate for a page graphic command

y1	number	y1	y1	The first y co-ordinate for a page graphic command
x2	number	x2	x2	The second x co-ordinate for a page graphic command
y2	number	y2	y2	The second y co-ordinate for a page graphic command
categories	text	cat	Categories	A space separated list of categories assigned to this news item
editorialMessage	text	em	Editorial Message	Used to send a note to the consumer about the contents of the story
editorialMessageAuthor	text	ema	Editorial Message Author	Who authored the contents of editorialMessage
expireDate	date	sedt	Story Expiration Date	Date story is no longer valid
storyBodyFormat	text	sfmt	Story Format	Format of contents of storyBody. Valid values are TEXT EXACTTEXT SGML XML HTML PDF
headlineBody	text	hdln	Headline	The actual headline
issueDate	date	sidt	Story Issue Date	Date and time story is issued. If not present, time is assumed to be on receipt
newsSource	text	nsrc	News Source	Source of news story
relatedSymbols	text	rsym	Related Symbols	Symbols of securities affected by this story
releaseDate	date	rdt	Story Release Date	When story may be released
reviewFlag	bool	rvflg	Story Review Flag	True if story has been reviewed

revision	number	rev	Story Revision	Revision number of story
sourceCategories	text	scat	Source Categories	Categories assigned by news source
sourceTime	time	stm	Source Time	Time of sourceDate
storyBody	text	story	Story	Story content
storyId	text	sid	Story Id	Identifier used to request associated story
storyLength	number	slen	Story Length	Story length, in characters (not bytes)
storyAuthor	text	sa	Story Author	Who wrote this story
tickCount	number	tikc	Tick Count	Number of ticks represented in this history bar
barTime	time	btm	Time	Time of barDate of this history bar
marketMakerID	text	mmid	Market Maker	Market Maker identifier
orderExpireDate	date	oexdt	Order Expiration Date	Date this order is no longer valid
orderExpireTime	time	oextm	Order Expiration Time	Time of orderDate when order is no longer valid
ordererId	text	oid	Broker ID	Broker ID
orderPriceModifier	text	opmc	Order Modifier	Modifiers on this order
orderDate	date	odt	Order Date	Date order was placed
orderTime	time	otm	Order Time	Time or orderDate order was placed
orderTimeSeq	number	otms	Order Time Sequence	Time sequence. When several orders are placed about the same time, this sequence will help tell them apart. Valu
orderPrice	price	opr	Order Price	Order price
orderSize	size	osz	Order Size	Number of shares of order

expireTime	time	setm	Story Expiration Time	Time of expireDate story is no longer valid
newsDate	date	ndt	News Date	Vendor feed date stamp
releaseTime	time	rtm	Story Release Time	Time of releaseDate when story may be released
sourceDate	date	sdt	Source Date	Story date assigned by news source
issueTime	time	sitm	Story Issue Time	Time of issueDate story is no longer valid
barDate	date	bdt	Date	Date of this history bar
newsTime	time	ntm	News Time	Time of newsDate for vendor feed date stamp

2 Market Data Control Set

These are the elements and attributes used by MDML.

property	An MDML property.
name	The name of a property.
type	In a property, the property's data type. In the "properties" element, a boolean indicating whether this attribute should be included in properties in the response.
value	In a property, the property's decimal value. In the "properties" element, a boolean indicating whether this attribute should be included in properties in the response.
displayHint	In a property, the property's formatting instruction. In the "properties" element, a boolean indicating whether this attribute should be included in properties in the response.
longDisplay	In a property, the long label for property. In the "properties" element, a boolean indicating whether this attribute should be included in properties in the

	response.
shortDisplay	In a property, the short label for the property. In the “properties” element, a boolean indicating whether this attribute should be included in properties in the response.
updating	T if request should update
useMasterProperties	T if server should send values using Master Property Set
Properties	List of properties to return in responses
symbol	Symbol for query
symbology	Symbology of value in id attribute. Legal values are “native”, “CUSIP”, “SEDOL”, “ISIN”
source	Source specifier if multiple sources are available
userAttribute	User supplied string returned with each query response
op	Attribute containing update operation. Legal values are “replace”, “add”, “insert” and “delete”.
replace	Update operator: Replace cached properties with new properties; if new properties aren’t in cache, create them
add	Update operator: add new properties to cache
insert	Update operator: used to insert a property set. Increment row attribute of matching property sets with row >= new property set row, then add new property set
delete	Update operator: delete property or property set. If row attribute is present, decrement row attributes of matching properties or sets where row > row of deleted element
id	Attribute used to identify properties and property sets that are not ordered.
row	Attribute used to identify properties and property sets that don’t have an id attribute, and to maintain server specified order
status	Attribute containing status of request. Legal values are “ok”, “permanent”, “temporary”
message	Attribute containing an error message
perm	Attribute containing an object’s permission group list

VehicleRequest	Vehicle request
VehicleCancel	Vehicle cancel
Vehicle	Vehicle query and image
VehicleUpdate	Vehicle update
ExchangeBlock	Container for sets of properties from individual exchanges. This is required if this information is to be permissioned by exchange.
MarketMakerBlock	Container for sets of properties from individual market makers. This is required if this information is to be permissioned by market maker.
DividendHistoryBlock	Container for sets of dividend history propertie. This is required if this information is to be permissioned.
ExchangePrice	Repeating property set for price information from individual exchanges. Used when a request can return information from multiple exchanges simultaneously.
DividendHistory	Repeating property set for dividend history information.
PageRequest	Page request
PageCancel	Page cancel
Page	Page query and image
TextSegment	Page segment with information for drawing text with with proper location and attributes. Page contains one or more TextSegments.
PageGraphicCommand	Drawing command for page with graphics
HeadlineListRequest	Headline list request
HeadlineListCancel	Headline list cancel
HeadlineList	Headline list query
StoryBodyFormats	Attribute containing a space separated list of formats the requester can accept. Supported values are: TEXT EXACTTEXT – linebreaks & character position are important SGML XML HTML PDF
NewsMatch	Contains a filter or search expression
isOp	Request to constrain returned data to that which matches this operator’s type and value
match	IsOp attribute containing the type of information contained in value. Values for News Headline requests:

	source category keyword text symbol Values for Symbol Search: company description cusip isin sedol native
value	isOp attribute containing the value of type indicated in match
timeRangeOp	Request to constrain the returned data to the specified time range.
timeType	timeRangeOp attribute specifying which time field to check. Valid values are: issue release expire story delete
notOp	Request to constrain returned data to that which does not match the child expression
andOp	Request to constrain returned data to that which matches ALL the child expressions
orOp	Request to constrain returned data to that which matches at least ONE of the child expression
HeadlineListUpdate	Update for headline list request
StoryRequest	News story request
StoryCancel	News story cancel
Story	News story query
HistoryRequest	History request
HistoryCancel	History cancel
History	History query and image
startDate	Far, or earliest date for which data is being requested. If not present, today is assumed.
startTime	Time of day for startDate, or today
endDate	Near, or latest date for which data is being requested. If not present, today is assumed.
endTime	Time of day for endDate, or today
span	Number of intervals
interval	History roll-up interval. Legal values: TICK, MINUTE, HOUR, DAY, WEEK, MONTH, YEAR
intervalSize	Number of periods in each interval (e.g. 7 minutes)

Bar	Interval property set for intervals larger than one day
DailyBar	Interval property set for DAY interval
QuoteBar	Interval property set for tick history
TimeAndSalesRequest	Time & Sales data request
TimeAndSalesCancel	Time & Sales cancel
TimeAndSales	Time & Sales query and image
date	Date for T&S query
relativeDay	Days in the past for T&S query
maxResponseCount	Maximum number of property sets in response
trades	T to return trades
quotes	T to return quotes
MarketMakerListRequest	Request all Market Makers
MarketMakerListCancel	Market Maker List cancel
MarketMakerList	Market Maker List query and image
MarketMaker	Market Maker property set.
MarketMakerListUpdate	Market Maker List update
order	Ordering for Market Maker List. ALPHA BID BID_TOUCH BID_COMPRESS ASK ASK_TOUCH ASK_COMPRESS
OrderBookRequest	Order Book request
OrderBookCancel	Order Book cancel
OrderBook	Order Book query and image.
Order	Property set containing a single order.
OrderBookUpdate	Order Book update
MarketDepthRequest	Market Depth request
MarketDepthCancel	Market Depth cancel
MarketDepth	Market Depth query
exchange	Attribute to MarketDepth specifying the exchange for which information is being requested
ExchangeInfo	Property set containing market depth information about an exchange.
MdRow	Property set containing data from one Market Data record.
OptionListRequest	Option List request
OptionListCancel	Option List cancel
OptionList	Option List query & image
Underlying	Option List or Future List underlying security
Option	Property set containing information about a single option.
OptionListUpdate	Option List update
FutureListRequest	Future List request

FutureListCancel	Future List cancel
FutureList	Future List query
Future	Property set containing information about one future.
FutureListUpdate	Future List update
IndexListRequest	Index List request
IndexListCancel	Index List cancel
IndexList	Index List query
Index	Property set containing information about one index entry.
IndexListUpdate	Index List update
MostActivesListRequest	Most Active issues on an exchange
MostActivesListCancel	Cancel Most Actives query
MostActivesList	Most Actives List query
MostActive	Property set containing information about one Most Actives entry.
MostActivesListUpdate	Most Actives List update
SymbolSearchRequest	Symbol Search request
SymbolSearchCancel	Symbol Search cancel
SymbolSearch	Symbol Search query & image
SymbolSet	Property set containing information about one symbol.
GenericListRequest	Generic symbol list request
GenericList	Generic symbol list query
Generic	Property set containing information about one element of the GenericList
GenericListCancel	Generic List cancel
GenericListUpdate	Generic List update

3 Display Hints

Hint	**Display Rules**
tic2	Display as non-reducible halves.
tic4	Display as non-reducible quarters.
tic8	Display as non-reducible eighths.
tic16	Display as non-reducible sixteenths.
tic32	Display as non-reducible $32^{nds.}$
tic64	Display as non-reducible $64^{ths.}$
tic128	Display as non-reducible $128^{ths.}$
tic32Plus	Take the integer portion and place it before a tick (‘) mark. Multiply the fractional portion by 256, then divide by 8, remembering the result and remainder. Place the result in the first two character positions folowing the tick mark, padding with a zero in the left position if necessary. If the remainder is a zero, display a space; if it is one half of an eighth (i.e., 4), display a “+”; if the

	remainder is anything other than a 0 or 4, display the digit.
reducible	Display smallest possible denominator of 2, 4, 8, 16, 32, 64, 128, 256.
dot0	Display as decimal.
dot1	Display with one digit to the right of the decimal point.
dot2	Display with two digits to the right of the decimal point.
dot3	Display with three digits to the right of the decimal point.
dot4	Display with four digits to the right of the decimal point.
dot5	Display with five digits to the right of the decimal point.
dot6	Display with six digits to the right of the decimal point.
dot7	Display with seven digits to the right of the decimal point.
dot8	Display with eight digits to the right of the decimal point.
dot9	Display with nine digits to the right of the decimal point.
dot100Plus	
eighthsOfCents	The integer portion is placed to the left of the decimal point. The fractional portion is multiplied by 800, then divided by 8. The result of this division is placed after the decimal point. If the remainder of the division by 8 is non-zero, it is considered to be the numerator of a fraction with a denominator of 8. This fraction should then be reduced. For example, the value 13.52375 would be displayed as 13.52 3/8.
half32	The integer portion is placed in the display as a whole number. The fractional portion is multiplied by 64, then divided by 2. The result of the division by 2 is placed as the numerator of a fraction with a denominator of 32. If the remainder of the division by 2 is non-zero, then it is placed as the numerator of a second fraction with a denominator of 2. For example, the value 168.984375 would appear on the display as 168 31/32 ½.
half32Plus	Differs from half32 in that if the second fraction is non-zero. A + is printed instead of the fraction.
quarter32	The integer portion is placed in the display as a whole number. The fractional portion is multiplied by 128, then divided by 4. The result of the division by 4 is placed as the numerator of a fraction with a denominator

	of 32. If the remainder of the division by 4 is non-zero, then it is placed as the numerator of a second fraction with a denominator of 4. For example, 84.4921875 would appear on the display as 84 15/32 ¾.
eighth32	The integer portion is placed in the display as a whole number. The fractional portion is multiplied by 256, then divided by 8. The result of the division by 8 is placed as the numerator of a fraction with a denominator of 32. If the remainder of the division by 8 is non-zero, then it is placed as the numerator of a second fraction with a denominator of 8. Here the value 42.24609375 would display as 42 7/32 7/8.
half64	The integer portion is placed in the display as a whole number. The fractional portion is multiplied by 128, then divided by 2. The result of the division by 2 is placed as the numerator of a fraction with a denominator of 64. If the remainder of the division by 2 is non-zero, it is placed as the numerator of a second fraction with a denominator of 2, e.g., 84.4921875 appears on the display as 84 31/64 ½.