XML Message Application Programming Interface For Merkato 2.3

October 2002

Document Number: API-2.3-1.00

Table of Contents

Introduction

Merkato is a software platform that supports a unique way to obtain bandwidth, on demand, at fair and optimal prices.

There are two ways to purchase bandwidth in a Merkato system.

- You can get a fixed amount of bandwidth, at a fixed price, for a fixed term
- You can create a bidding strategy and contend for bandwidth with other buyers at frequent intervals, establishing a *fair market price* based on supply and demand

The first method is called the Reservation market. The second method is called the Spot market.

This document describes the Application Programming Interface that can be used to get and set configuration information from a buyer or seller agent in a Merkato garage.

For a description of the theory and use of Merkato, see *Merkato Buyer's Reference Manual* and *Merkato Seller's Reference Manual*.

Merkato Agents and Markets

Merkato uses intellegent agents that interact on behalf of buyers and sellers. Buyers can acquire bandwidth by configuring their agents with the price they are willing to pay for a range of available quantity, or with the reservation they want. Sellers configure their agents with a quantity of bandwidth for sale and a minimum price for that quantity. Buyer agents then express what they are willing to pay for bandwidth in the form of bids, consisting of a unit price and a quantity.

The Merkato market responds to bids with proposed allocations and pricing. The buyer and seller agents can either accept the proposed allocation (by not bidding further), or reject the proposed allocation (by submitting another bid). When there is no more bidding, the auction closes, and Merkato makes bandwidth allocations.

Merkato agents consist of Java modules that use XML, running on the Merkato server. The Merkato interface, either HTML or Java, includes a system that manipulates the XML modules.

This document demonstrates how you should send HTML requests, containing XML elements that can be handled by a Merkato server. Merkato interprets the XML elements as function calls.

This description can be considered complete; Merkato supports no other XML elements.

Terminology in the XML API does not correspond to terminology in the current Merkato GUI. The Glossary (on page 41) lists API words with current terminology.

Throughout this document, terms shown in **bold** are to be replaced by values that you select.

Syntax

The Merkato server accepts XML input as a POST request to a Merkato “access servlet.”

- The request parameter’s name will be “call.”
- The request parameter’s value is a <methodCall> element with some child elements.

XML code should not contain white space between elements.

Following is an example of an HTTP request to Merkato.

```
POST /merkato/access HTTP/1.1
HOST: MERKATO_HOST:HTTP_PORT (hostname and port of Merkato
server)
ACCEPT: text/html, text/xml
CONTENT-LENGTH: varies


call=<methodCall>
  <action name="x-apply | x-view"
    garageURL="http://HOSTNAME:HTTP_PORT/merkato/
    garage-name/"/>
  <Subelements-with-type-of-action/>
  <BuyerIdentity name="name" password="password"/>
</methodCall>
```

As shown above, always address requests to `access` under `merkato`:
`HTTP://`***`MERKATO_HOST:HTTP_PORT`***`/merkato/access`

Among the child elements of <methodCall>, the <action> and <BuyerIdentity> elements are mandatory. (See “Authentication and Security” on page 32.). Other elements are the XML that the Merkato garage could dispatch into a player (that is, a buyer or seller) object’s methods. (See “Recognized Elements” on page 4.)

Usage

There are two types of actions you can perform:

- see your player's current attributes via XML code (x-view)
- change your player's attributes (x-apply)

For the first type of action, the user does not need to specify anything about what the player should do; there are no method calls and any non-mandatory element is ignored.

For the second type of action, the user tells the player how it should change, using method calls.

The user's choice is encoded in the `<action>` element's name attribute. The value of this can should be either `x-view` or `x-apply`.

Other mandatory elements are encoded in the `<BuyerIdentity>` element.

To change the configuration of a player inside the garage, encode the method calls per "Recognized Elements" on page 4.

Example:

To see the player's state use the following syntax:

```
call=<methodCall>
     <action name="x-view"/>
     <BuyerIdentity name="Uname" password="Upasswd"/>
     <!-- password in PLAIN TEXT -->
     </methodCall>
```

This method returns one of the following:

- the player's XML
- the player's XML and an <*exception*> element
- an <*exception*> element

Example:

This example changes the player's state:

```
call=<methodCall>
     <action name="x-apply"/>
     <BuyerIdentity name="Uname" password="Upasswd"/>
     <!-- password in PLAIN TEXT -->
     <USER's ELEMENT ON METHOD CALL>
     </methodCall>
```

This method returns either an <*Exit/*> element upon success or an <*exception*> element.

Recognized Elements

This section lists all the currently supported method calls a player-object can take and the values that the HTTP request parameters call.

The method calls are shown in the format that the garage understands. You can use other formats and let XSL transform it into this format. InvisibleHand does not provide XSL code.

A <*param*> element's "name" attribute is a fixed value. Indents are shown inside elements for ease of reading.

Summary

Player's Strategy

Player's Strategy

Use the following element to change the player strategy to `StrategyName`.

```
<currentStrategySelected>
     <param name="string" value="StrategyName"/>
</currentStrategySelected>
```

For Buyers, ***`StrategyName`*** can be one of the following:

```
Manual Strategy | Auto Strategy | Reservation Strategy
```

For Sellers, ***`StrategyName`*** can be one of the following:

```
Dynamic Seller Strategy | Static Seller Strategy
```

Note: These selections are case sensitive and space sensitive (additional spaces cannot be placed between words).

Player's Type of Valuation

Use the following element to change the player valuation to `ValuationName`.

```
<currentValuationSelected>
     <param name="string" value="ValuationName"/>
</currentValuationSelected>
```

`ValuationName` can be one of the following:

```
Budget Valuation | Budget Valuation with Limits |
Square Root Valuation | Logarithmic Valuation |
Linear Valuation
```

For standard Merkato equivalent terms, see the Glossary on page 41.

Resource Agent Selection

The selection of the Resource Agent determines the marketplace from which the buyer agent will bid. (Seller agents are dedicated to a single market.) Use the following element to change the Resource agent selection to ***RA-Name***.

Note that this is a text string, not a URL.

```
<currentResourceAgentURLSelected>
     <param name="string" value="RA-Name"/>
</currentResourceAgentURLSelected>
```

The supported value for both the param name and the value is a text string.

Possible values for the ***RA-Name*** value are the names of the available resource agents for the buyer. They can be obtained by viewing the player's XML and reading them from the ResourceAgentURL element.

Valuation Quantities

In the sample below, the element `<param name="AType" value="AValue">` is interpreted as "a parameter of type `AType` with the value `AValue`."

To assign quantity values to types of valuations, use the following syntax.

```
<valuationChanged>
     <param name="string" value="Valuation Name"/>
     <ValuationElement>
          <Quantity name="qmax"
                displayValue="quantity"
                displayUnit="quantity-units"/>
          <!-- qmin, as below, if necessary for the
                valuation type -->
          <Quantity name="qmin"
                displayValue="quantity"
                displayUnit="quantity-units"/>
          <!-- "vmax", as below, is replaced by "budget",
                depending on the valuation type -->
          <Value name="vmax"
                displayValue="value"
                displayUnit="value-units"/>
     </ValuationElement>
</valuationChanged>
```

- ***Valuation Name*** can be one of the following:
  Budget Valuation | Linear Valuation | Square Root Valuation | Budget Valuation with Limits | Logarithmic Valuation | Parabolic Valuation
- ***ValuationElement*** is the programmatic label for the valuation. It must match the ***Valuation Name.*** The following table shows the ***ValuationElement*** labels for the supported valuations

Valuation Name	***ValuationElement***
Budget Valuation	BudgetValuation
Linear Valuation	LinearValuation
Square Root Valuation	RobertsValuation
Budget Valuation with Limits	BoundedBudgetValuation
Logarithmic Valuation	LogarithmicValuation
Parabolic Valuation	ParabolicValuation

- ***`quantity`*** is the maximum quantity the buyer agent requests in the valuation. Each type of valuation uses this value differently. Enter it as a positive real number
- ***Quantity-units*** may be: kbps | Mbps | Gbps
- Each type of valuation uses ***`value`*** differently. In valuations without "budget" in its name, it is the positive real number corresponding to the maximum amount that the user is willing to pay for the maximum quantity requested. It is also used to refer to the budgeted amount, for valuations with "budget" in their name.
- ***`value-units`*** are expressed in terms of currency units per unit time.
  - ***`Currency`*** units can be: c | $
  - ***`Time`*** units can be: min | h | day | month

```
For example "$/month" would be a valid entry for
     value-units
```

Example 1 (Square Root Valuation):

```
<valuationChanged>
     <param name="string" value="Square Root Valuation"/>
     <RobertsValuation>
          <Quantity name="qmax"
               displayValue="50" displayUnit="kbps"/>
          <Value name="vmax"
               displayValue="900" displayUnit="$/month"/>
     </RobertsValuation>
</valuationChanged>
```

Example 2 (Logarithmic Valuation):

```
<valuationChanged>
     <param name="string" value="Logarithmic Valuation"/>
     <LogarithmicValuation>
          <Quantity name="qmax" displayValue="100"
               displayUnit="Mbps"/>
          <Value name="vmax" displayValue="700"
               displayUnit="c/day"/>
     </LogarithmicValuation>
</valuationChanged>
```

Example 3 (Linear Valuation):

```
<valuationChanged>
     <param name="string" value="Linear Valuation"/>
     <LinearValuation>
          <Quantity name="qmax" displayValue="150"
               displayUnit="Gbps"/>
          <Value name="vmax"
               displayValue="500"
               displayUnit="$/h"/>
     </LinearValuation>
</valuationChanged>
```

Example 4 (Budget Valuation):

```
<valuationChanged>
     <param name="string" value="Budget Valuation"/>
          <BudgetValuation>
               <Value name="budget"
                    displayValue="17" displayUnit="$/min"/>
          </BudgetValuation>
</valuationChanged>
```

Example 5 (Budget Valuation with Limits):

```
<valuationChanged>
     <param name="string"
          value="Budget Valuation with Limits"/>
     <BoundedBudgetValuation>
          <Quantity name="qmax" displayValue="200"
               displayUnit="Mbps"/>
          <Quantity name="qmin" displayValue="170"
               displayUnit="Mbps"/>
          <Value name="budget" displayValue="2000"
               displayUnit="c/min"/>
     </BoundedBudgetValuation>
</valuationChanged>
```

Traffic-Based and Quick-Response Parameters

Changing parameters for Traffic-Based and Quick-Response valuation works the same way as changing parameters for valuation quantities. The elements are part of the call to change the valuation.

Note: You cannot change the type of traffic feedback by using the API. You can alter the parameters for the current traffic feedback feature assigned by the administrator.

For a complete description of Traffic-base Valuation and how to use it, see *Merkato Buyer's Reference Manual*.

Traffic-Based Extension to Valuation

To configure Traffic-Based valuations (valuations whose attributes depend on measured traffic), use the following syntax, which is an expansion of the ValuationElement syntax.

```
<valuationChanged>

     <param name="string" value="Valuation Name"/>

     <ValuationElement>

           <Quantity name="qmax"
             displayValue="quantity"
             displayUnit="quantity-units"/>

           <Value name="vmax"
             displayValue="value"
             displayUnit="value-units"/>

           <TrafficValuationModulator
             dataCollector="TrafficDataCollector"
             active="[true|false]"/>

           <Duration name="measuredTimeFrame"
             displayValue="time-frame"
             displayUnit="time-units"/>

           <Quantity name="margin"
             displayValue="quantity-margin"
             displayUnit="quantity-units"/>

           </TrafficValuationModulator>

     </ValuationElement>

</valuationChanged>
```

- ***ValuationElement,*** ***quantity, quantity- units, value,*** and ***value-units*** are expressed as for the simple valuation.
- The ***active*** parameter activates and deactivates the Traffic-Based valuation. Possible values are the strings **true** and **false**.

- ***`time-frame`*** is the Time Window**.** The agent continuously measures the amount of traffic you generate. For determining the peak traffic, the agent examines past traffic as far back as the time-frame value. Possible values must be positive, with a minimum value of 6 minutes.
- ***`quantity-margin`*** is the amount of bandwidth the Buyer agent attempts to obtain above the amount of the measured peak. Positive or negative number are allowed for this value. Negative numbers are allowed to compensate for active reservations which contribute to the total bandwidth allocation received. In general, for the feature to take effect, the measured traffic peak plus the quantity margin should be lower than the qmax of the selected valuation.
- ***`Quantity-units`*** and ***`time units`*** are the unit type in which the quantity is expressed. Possible values are as follows:
  - ***Quantity-units*** can be: kbps | Mbps | Gbps
  - ***Time-units*** can be: min | h | day | month

Quick Response Extension to Traffic-based Valuation

To configure the Quick Response extension to traffic-based valuations, use the following syntax, which is an expansion of the ValuationElement syntax, shown above.

For Quick Response, the expression *TrafficValuationModulator* (in Traffic-Based Valuation code) is substituted with *QuickResponseTrafficValuationModulator*. The child element is the Quick Response parameters.

To configure Quick Response valuations, use the following syntax, which is an expansion of the Traffic-Based syntax.

```
<valuationChanged>
     <param name="string" value="Valuation Name"/>
     <ValuationElement>
          <Quantity name="qmax"
            displayValue="quantity"
            displayUnit="quantity-units"/>
          <!—add qmin as above if supported-->
          <Value name="vmax"
            displayValue="value"
            displayUnit="value-units"/>
          <QuickResponseTrafficValuationModulator
            dataCollector=”TrafficDataCollector”
            active=”[true|false]”/>
          <Duration name=”measuredTimeFrame”
            displayValue=”time-frame”
            displayUnit=”time-units”/>
          <Quantity name=”margin”
            displayValue=”quantity-margin”
            displayUnit=”quantity-units”/>
          <Quantity name="threshold"
            displayValue="threshold"
            displayUnit="quantity-units"/>
          <Quantity name="estimatedPeak"
            displayValue=“estimated-peak"
            displayUnit="quantity-units”/>
          </QuickResponseTrafficValuationModulator>
          </TrafficModulator>
     </ValuationElement>
</valuationChanged>
```

- ***`ValuationElement,`*** ***quantity, quantity- units, value,*** and ***value-units*** are expressed as for the simple valuation.
- The *active* parameter activates and deactivates the Traffic-Based valuation. Possible values are the strings **true** and **false**.
- ***`time-frame`*** is the Time Window**.** The agent continuously measures the amount of traffic you generate. For determining the peak traffic, the agent examines past traffic as far back as the time-frame value. Possible values must be positive, with a default minimum value of 6 minutes.
- ***`quantity-margin`*** is the amount of bandwidth the Buyer agent attempts to obtain above that of the measured peak. Any number is allowed for this value. In general, for the feature to take effect, the measured traffic peak plus the quantity margin should be lower that qmax of the selected valuation.
- ***`Quantity units`*** `and` ***`time units`*** are the unit type in which the quantity and time is expressed. Possible values are as follows:
  - **Quantity** units can be: kbps | Mbps | Gbps
  - **Time** units can be: min | h | day | month
- ***`threshold`*** is the traffic value that engages Quick Response. Possible values are positive numbers.
- **`estimated-peak`** is the value that is used in the calculation for desired quantity when Quick Response is engaged. Possible values are positive numbers.

Unit Values

Units selections change the screen display units for currency, time, or quantity.

Use the following syntax:

```
<unitSelected>
      <param name="string" value="U-Unit"/>
</unitSelected>
```

To change a value of a particular set of units to the value of ***`U-Unit`***, you do not need to specify whether the ***`U-Unit`*** is currency, time, or quantity.

The type ("name") of ***`U-Unit`*** is always `string`.

Permissible values for *`U-Unit`* are:

```
Quantity: kbps | Mbps | Gbps
Currency: c | $
Time: min | h | day | month
```

Starting the Player

To start the player inside garage, use the following syntax:

```
<startRequested/>
```

Stopping the Player

To stop the player inside garage, use the following syntax:

```
<stopRequested/>
```

Reservations

You must send two calls to place a reservation. The first call, the request call, returns a reservation price quotation. The second call, the confirmation call, accepts the quotation.

You can create a function that handles both the request call and the confirmation call in one procedure. The function must require the user to specify a maximum price to accept, and then evaluate the return from the request call to check that it does not exceed the user's maximum.

Request a Reservation

To submit a request for a reservation, use the following syntax,

```
<bidSubmitted>
  <ReservationBid>
    <ReservedQuantity name="quantity"
     displayUnit="quantity-units"
     displayValue="quantity">
       <Time name="start"
         year="year"
         month="month"
         day="day"
         hour="hour"
         min="min"/>
       <Duration name="duration"
         displayUnit="time-units"
         displayValue="duration"/>
    </ReservedQuantity>
  </ReservationBid>
</bidSubmitted>
```

- The `Time` element with `name="start"` represents the start date for the reservation. For the Time element, possible values are limited to a valid date in the future starting five minutes after confirmation. Alternatively, the year, month, day, hour, and min fields can be replaced by a single **time="now"** field, which begins the reservation immediately after it is confirmed.

- ***`quantity`*** is the quantity requested in the reservation, possible values are positive numbers. Possible values for ***`quantity-units`*** are: `kbps | Mbps | Gbps`.
- ***`duration`*** is the duration of the reservation. This value may have to be calculated to achieve a desired end date and time. Possible values are positive numbers.
- Possible values for ***time-units*** are `min | h | day | month`.

The format for ***year, month, day, hour***, and ***min*** are all numeric.

The following is an example of a reservation request

```
<bidSubmitted>
     <ReservationBid>
          <ReservedQuantity
            displayUnit="Mbps"
            displayValue="1"
            name="quantity">
               <Duration displayUnit="min"
                displayValue="60"
                name="duration" />
          </ReservedQuantity>
     </ReservationBid>

</bidSubmitted>
```

Reservation Returned Quotation Values

Successful returned quotation values for requested reservations are the following. (Returned values shown in **bold.** Text has been reformatted to improve legibility) :

```
<ReservationBid sessionId="session-id"
 confirmed="false"
 percentCancellationFee="percent"
 type="bid">

    <Price name="price"
     displayUnit="price-unit"
     displayValue="price"
     value="raw-price"/>

    <ReservedQuantity name="quantity"
     displayUnit="quantity-unit"
     displayValue="quantity"
     value="raw-value">

         <Time name="start"
          year="year"
          month="month"
          day="day"
          hour="hour"
          min="min"/>

          <Duration name="duration"
          displayUnit="time-unit"
          displayValue="duration"
          value="raw-duration"/>

    </ReservedQuantity>

    <Value name="value"
     displayUnit="value-unit"
     displayValue="value"
     value="raw-value"/>

    <Cost name="cost"
     displayUnit="currency-unit"
     displayValue="cost"/>

    <Cost name="fee"
     displayUnit="currency-unit"
     displayValue="fee-cost"
     value="raw-fee"/>

</ReservationBid>
```

Many of the elements repeat the values that were sent in the request. This includes ReservedQuantity and Duration. The only difference in these elements is that an additional “raw” field is added, which always has time units of millisec, quantity units of kbps, and currency units of cents. For example, if the Price element shows displayUnit value of “$/month/Mbps”, then the ***raw-price*** value for price is represented in units of “c/millisec/Kbps”. It is useful programmatically if you wish to import the value without parsing the displayUnits text to determine the units employed. (Note that display values are often rounded to zero,whereas the raw values are always accurate).

- ***session-id*** is a unique numerical identifier for this quotation.
- ***percent*** is a number between 0 and 100. It represents the percent of remaining cost of the reservation which will be charged if a reservation is cancelled before it is expired – even if the reservation start time has not yet occured.
- The Price element represents the unit price quoted for the desired bandwidth**.** ***raw-price*** is always represented in units of "cents/millisec/kbps". ***price-unit*** in the Price element consists of units of currency per time per quantity, such as "$/month/Mbps". Valid values for each unit type are:

  ```
  Quantity: kbps | Mbps | Gbps
  Currency: c | $
  Time: min | h | day | month
  ```

price is a positive number represented in the displayUnit units

- The Value element represents the cost per unit time quoted for the desired bandwidth**.** ***Raw-value*** is always represented in units of "cents/millisec". ***value-unit*** in the Price element consists of units of currency per time, such as "$/month". Valid values for each unit type are:

  ```
  Currency: c | $
  Time: min | h | day | month
  ```

  value is a positive number represented in the displayUnit units
- The Cost element represents the total cost quoted for the desired bandwidth**.** ***raw-cost*** is always represented in units of "cents". ***cost-unit*** in the Cost element consists of units of currency, such as "$". Valid values for currency are:

  ```
  Currency: c | $
  ```

  cost is a positive number represented in the displayUnit units
- There is an additional `Cost` element where ***`fee-cost`*** represents the fee associated with the reservation and ***`currency-units`*** represents the currency unit in which the fee is expressed (`c` or `$`). ***raw-fee*** is always expressed in units of cents. **raw-fee** should always be "0.0" as it is not charged at this time.

The following is an example of a returned reservation quotation (requested reservation was for 1 Mbps for one hour). Text has been reformatted to improve legibility.:

```
<ReservationBid confirmed="false"
     percentCancellationFee="10.0" sessionId="56"
     type="bid">
     <Price displayUnit="$/month/Mbps" displayValue="300.0"
          name="price" value="1.15740738E-8"/>
     <ReservedQuantity displayUnit="Mbps"
          displayValue="1.0" name="quantity"
          value="1000.0">
          <Time name="start"  time="now"/>
          <Duration displayUnit="month" displayValue="0.0"
                name="duration" value="3600000.0"/>
          </ReservedQuantity>
     <Value displayUnit="$/month" displayValue="300.0"
          name="value" value="1.15740738E-5"/>
     <Cost displayUnit="$" displayValue="0.42" name="cost"
          value="42"/>
     <Cost displayUnit="$" displayValue="0.0" name="fee"
          value="0.0"/>
</ReservationBid>
```

Request Failure

If the request for a reservation fails, the returned element is the following:

`<Exception message="`***`explanation`***`"/>`

`explanation` is a text string describing the error that occurred.

Reservation Confirmation

The call to confirm the reservation must use the *ReservationBid* element, which is returned by the bid submission call:

```
<bidConfirmed>
    <ReservationBid sessionId="session-id"
     confirmed="false"
     percentCancellationFee="percent"
     type="bid">
        <Price name="price"
         displayUnit="price-unit"
         displayValue="price"
         value="raw-price"/>
        <ReservedQuantity name="quantity"
         displayUnit="quantity-unit"
         displayValue="quantity"
         value="raw-value">
          <Time name="start"
           year="year"
           month="month"
           day="day"
           hour="hour"
           min="min"/>
          <Duration name="duration"
           displayUnit="time-unit"
           displayValue="duration"
           value="raw-duration"/>
        </ReservedQuantity>
        <Value name="value"
         displayUnit="value-unit"
         displayValue="value"
         value="raw-value"/>
        <Cost name="cost"
         displayUnit="currency-unit"
         displayValue="cost"/>
        <Cost name="fee"
         displayUnit="currency-unit"
         displayValue="fee-cost"
         value="raw-fee"/>
     </ReservationBid>
</bidConfirmed>
```

Note: All returned values must match those in the reservation request quotation. All fields in the returned quotation are not manditory, but it is harmless to include them. Note that the order of the elements within a category, or the order of fields within an element need not be identical to the returned quotation.

```
<bidConfirmed>
     <ReservationBid sessionId="56" confirmed="false"
          percentCancellationFee="10.0" type="bid">
          <Value displayUnit="$/month" displayValue="300.0"
               value="1.15740738E-5" name="value" />
          <Price displayUnit="$/month/Mbps"
               displayValue="300.0" value="1.15740738E-8"
               name="price" />
          <ReservedQuantity displayUnit="Mbps"
               displayValue="1.0" value="1000.0"
               name="quantity">
               <Duration displayUnit="month"
                     displayValue="0.0" value="3600000.0"
                     name="duration" />
               <Time time="now" name="start" />
          </ReservedQuantity>
          <Cost displayUnit="$" displayValue="0.42"
               value="42" name="cost" />
          <Cost displayUnit="$" displayValue="0.0" value="0.0" name="fee"/>
     </ReservationBid>
</bidConfirmed>
```

Note: The reservation is confirmed by the bidConfirmed element. The **confirmed="false"** field in the ReservationBid element must be left alone or omitted.

Reservation Confirmation Return Value

The return value will be `<Success/>` if the confirmation was sent successfully to the resource agent.

Reservation Confirmation Failure

If the reservation confirmation fails, the returned element is the following:

`<Exception message="`***`explanation`***`"/>`

`explanation` is a text string describing the error that occurred.

Result of x-view Action

Using a methodCall with the “x-view” action will result in the entire XML profile of the player being returned.

Spot Market Buyer Profile

Most of the information returned when the XML for a buyer is viewed is the same as that which can be set. Aside reading the current values of these parameters (which are explained above), other information can be gathered about a buyer. This includes:

- In the BuyerIdentity element, you can determine the IP address and subnet mask by which this buyer’s traffic is identified.
- In the Bid element, you can determine the current unit price and quantity of the current bid, as well as the extended value (price time quantity).
- In the Allocation section, you can learn the price, quantity and value (price time quantity) of the bandwidth allocated to this buyer in the last auction (the allocation is in effect during the auction in progress). Note that there is also a bid “fee” displayed whose purpose is to prevent frivolous bidding. It is not currently charged to buyers, but is still calculated for reference.

For Spot Market players, a typical XML profile will look like this:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>

<Buyer activationDate="null" active="true"
    class="net.invisiblehand.merkato.agents.player.
    PlayerImpl"
    context="http://63.111.138.7:8505/merkato/access"
    news="Received an allocation." revisionDate="$Date:
    2002/06/03 21:49:11 $" revisionNumber="$Revision:
    1.95 $">

  <BuyerIdentity name="buyer-A01" addresstype="ip"
      destinationipaddress="" destinationipmask=""
      password="YnV5ZXItQTAx"
      sourceipaddress="192.168.0.1"
      sourceipmask="255.255.255.255">

    <sessionID id="14"/>


  <TrafficDataCollector name="TrafficDataCollector"
      address="http://63.111.138.7:8505/merkato/
      graph?command=measure" direction="IN"
      refresh="100000"/>

  <TruthfulStrategy name="Auto Strategy"/>

  <ManualStrategy name="Manual Strategy"/>

  <BudgetValuation name="Budget Valuation">
```

```
    <Value name="budget" displayUnit="$/month"
       displayValue="150.0" value="5.7870369E-6"/>

    <QuickResponseTrafficValuationModulator
       active="false"
       dataCollector="TrafficDataCollector">

      <Quantity name="margin" displayUnit="Mbps"
         displayValue="2.0" value="2000.0"/>


      <Duration name="measuredTimeFrame"
         displayUnit="month" displayValue="0.0"
          value="1200000.0"/>

      <Duration name="minTimeFrame"
         displayUnit="month" displayValue="0.0"
         value="360000.0"/>

      <Quantity name="threshold" displayUnit="Mbps"
         displayValue="6.0" value="6000.0"/>

      <Quantity name="estimatedTrafficPeak"
         displayUnit="Mbps" displayValue="20.0"
         value="20000.0"/>

    </QuickResponseTrafficValuationModulator>

  </BudgetValuation>

<BoundedBudgetValuation name="Budget Valuation with
Limits">

    <Quantity name="qmin" displayUnit="Mbps"
       displayValue="1.0" value="1000.0"/>

    <Quantity name="qmax" displayUnit="Mbps"
       displayValue="10.0" value="10000.0"/>

    <Value name="budget" displayUnit="$/month"
       displayValue="222.0" value="8.564814612E-6"/>

    <QuickResponseTrafficValuationModulator
       active="false"
       dataCollector="TrafficDataCollector" error=""
       status="off" thresholdTime="1.024072201E9">

      <Quantity name="margin" displayUnit="Mbps"
         displayValue="5.0" value="5000.0"/>

      <Duration name="measuredTimeFrame"
         displayUnit="month" displayValue="0.0"
         value="360000.0"/>

      <Duration name="minTimeFrame"
         displayUnit="month" displayValue="0.0"
         value="360000.0"/>
      <Quantity name="threshold" displayUnit="Mbps"
         displayValue="1.0" value="1000.0"/>
      <Quantity name="estimatedTrafficPeak"
         displayUnit="Mbps" displayValue="8.0"
         value="8000.0"/>
    </QuickResponseTrafficValuationModulator>
  </BoundedBudgetValuation>
  <RobertsValuation name="Square Root Valuation">
    <Quantity name="qmax" displayUnit="Mbps"
       displayValue="100.0" value="100000.0"/>
    <Value name="vmax" displayUnit="$/month"
       displayValue="10000.0"
       value="3.8580245999999993E-4"/>
    <QuickResponseTrafficValuationModulator
       active="true"
       dataCollector="TrafficDataCollector"
       error="">
      <Quantity name="margin" displayUnit="Mbps"
         displayValue="2.0" value="2000.0"/>
      <Duration name="measuredTimeFrame"
         displayUnit="month" displayValue="0.0"
         value="1200000.0"/>
      <Duration name="minTimeFrame"
         displayUnit="month" displayValue="0.0"
         value="360000.0"/>
      <Quantity name="threshold" displayUnit="Mbps"
         displayValue="6.0" value="6000.0"/>
      <Quantity name="estimatedTrafficPeak"
         displayUnit="Mbps" displayValue="20.0"
         value="20000.0"/>
    </QuickResponseTrafficValuationModulator>
  </RobertsValuation>
  <LogarithmicValuation name="Logarithmic Valuation">
    <Quantity name="qmax" displayUnit="Mbps"
       displayValue="50.0" value="50000.0"/>
    <Value name="vmax" displayUnit="$/month"
       displayValue="500.0" value="1.9290123E-5"/>
    <QuickResponseTrafficValuationModulator
       active="false"
       dataCollector="TrafficDataCollector">
      <Quantity name="margin" displayUnit="Mbps"
         displayValue="2.0" value="2000.0"/>
      <Duration name="measuredTimeFrame"
         displayUnit="month" displayValue="0.0"
         value="1200000.0"/>
      <Duration name="minTimeFrame"
         displayUnit="month" displayValue="0.0"
         value="360000.0"/>
      <Quantity name="threshold" displayUnit="Mbps"
         displayValue="6.0" value="6000.0"/>
      <Quantity name="estimatedTrafficPeak"
         displayUnit="Mbps" displayValue="20.0"
         value="20000.0"/>
    </QuickResponseTrafficValuationModulator>
  </LogarithmicValuation>
  <ParabolicValuation name="Parabolic Valuation">
    <Quantity name="qmax" displayUnit="Mbps"
       displayValue="50.0" value="50000.0"/>
    <Value name="vmax" displayUnit="$/month"
       displayValue="1000.0" value="3.8580246E-5"/>
    <QuickResponseTrafficValuationModulator
       active="false"
       dataCollector="TrafficDataCollector">
      <Quantity name="margin" displayUnit="Mbps"
         displayValue="2.0" value="2000.0"/>
      <Duration name="measuredTimeFrame"
         displayUnit="month" displayValue="0.0"
         value="1200000.0"/>
      <Duration name="minTimeFrame"
         displayUnit="month" displayValue="0.0"
         value="360000.0"/>
      <Quantity name="threshold" displayUnit="Mbps"
         displayValue="6.0" value="6000.0"/>
      <Quantity name="estimatedTrafficPeak"
         displayUnit="Mbps" displayValue="20.0"
         value="20000.0"/>
    </QuickResponseTrafficValuationModulator>
  </ParabolicValuation>
  <LinearValuation name="Linear Valuation">
```

```
  <Quantity name="qmax" displayUnit="Mbps"
     displayValue="45.0" value="45000.0"/>
  <Value name="vmax" displayUnit="$/month"
     displayValue="100.0" value="3.8580246E-6"/>
  <QuickResponseTrafficValuationModulator
     active="false"
     dataCollector="TrafficDataCollector">
    <Quantity name="margin" displayUnit="Mbps"
       displayValue="2.0" value="2000.0"/>
    <Duration name="measuredTimeFrame"
       displayUnit="month" displayValue="0.0"
       value="1200000.0"/>
    <Duration name="minTimeFrame"
       displayUnit="month" displayValue="0.0"
       value="360000.0"/>
    <Quantity name="threshold" displayUnit="Mbps"
       displayValue="6.0" value="6000.0"/>
    <Quantity name="estimatedTrafficPeak"
       displayUnit="Mbps" displayValue="20.0"
       value="20000.0"/>
  </QuickResponseTrafficValuationModulator>
</LinearValuation>
<ResourceAgentURL name="Resource-A-
   spot">http://63.111.138.7:8505/merkato/Resource-
   A-spot/</ResourceAgentURL>
<Selections resourceAgent="Resource-A-spot"
   strategy="Auto Strategy" valuation="Budget
   Valuation with Limits"/>
<GarageURL name="spot-
   garage">http://63.111.138.7:8505/merkato/
   spot-garage</GarageURL>
<Units currencyUnit="$" quantityUnit="Mbps"
   timeUnit="month"/>
<Bid sessionId="74" type="bid">
  <Price name="price" displayUnit="$/month/Mbps"
     displayValue="133.72"
     value="5.1590412721694115E-9"/>
  <Quantity name="quantity" displayUnit="Mbps"
     displayValue="1.66" value="1660.15625"/>
  <Value name="value" displayUnit="$/month"
     displayValue="222.0" value="8.564814612E-6"/>
  </Bid>
  <Allocation allocId="1029435927"
     reservationId="1029435927" sessionId="74">
    <BuyerIdentity name="buyer-A01" addresstype="ip"
       destinationipaddress="" destinationipmask=""
       sourceipaddress="192.168.0.1"
       sourceipmask="255.255.255.255">
      <sessionID id="14"/>
    </BuyerIdentity>
    <SellerIdentity name="seller-A"
       principalid="PRINCIPALID">
      <sessionID id="21"/>
    </SellerIdentity>
    <Price name="price" displayUnit="$/month/Mbps"
       displayValue="133.33" value="5.1440328E-9"/>
    <Quantity name="quantity" displayUnit="Mbps"
       displayValue="1.66" value="1660.15625"/>
    <Cost name="fee" displayUnit="$"
       displayValue="0.0"
       value="1.5000000000000001E-4"/>
    <Value name="value" displayUnit="$/month"
       displayValue="4109.35"
       value="1.5853989820312502E-4"/>
  </Allocation>
</Buyer>
```

Reservation Buyer Profile

For Reservation buyers, the XML profile will look similar to the following (when there is no pending or active reservations):

```
<?xml version="1.0" encoding="UTF-8" ?>
<Buyer
   context="http://63.111.138.7:8505/merkato/garage">
  <BuyerIdentity name="buyer-A10"
     password="YnV5ZXItQTEw"
     sourceipaddress="192.168.0.10"
     sourceipmask="255.255.255.255" addresstype="ip"/>
  <ReservationManualStrategy name="Reservation
     Strategy"/>
  <ResourceAgentURL name="Resource-A-
     reservation">http://63.111.138.7:8505/merkato/
     Resource-A-reservation/</ResourceAgentURL>
  <Selections strategy="Reservation Strategy"
     resourceAgent="Resource-A-reservation"/>
  <GarageURL name="reservation-
     garage">http://63.111.138.7:8505/merkato/
     reservation-garage</GarageURL>
  <Units currencyUnit="$" quantityUnit="Mbps"
     timeUnit="month"/>
</Buyer>
```

Viewing Active and Pending Reservations

The current active and pending reservations of the buyer can be extracted as child nodes of the Buyer element, under the Allocation element. Fields are the same as described for requesting and confirming reservations. Note that only the “quantity” and “value” of these reservations is provided. To determine the overall cost, the “value” must be multiplied by the “quantity”.

```
<Buyer>
.....other tags .............
<Allocation>
  <ReservedQuantity name="quantity"
    displayUnit="quantity-unit" displayValue="quantity">
  <Time name="start" year="year" month="month" day="day"
    hour="hour" min="min"/>
  <Duration name="duration" displayUnit="time-unit"
    displayValue="duration"/>
  </ReservedQuantity>
  <Value name="value" displayUnit="value-unit"
    displayValue="alloc-value"/>
  <Cost name="fee" displayUnit="cost-unit"
    displayValue="fee-cost"/>
<Allocation/>
…
<Buyer/>
```

Example of Reservation Buyer View

If there are multiple active reservations, the Allocation element will be repeated for each one, as shown in the example below. Note that information about the last requested or submitted bid is given in the ReservationBid element as well.

```
<Buyer activationDate="null" active="true"
        class="net.invisiblehand.merkato.agents.player.PlayerImpl"
        context="http://192.168.100.44:8505/merkato/access"
        news="Received an allocation." revisionDate="$Date: 2002/09/1921:36:56 $"
        revisionNumber="$Revision: 1.101 $">
        <BuyerIdentity addresstype="ip"
        destinationipaddress="" destinationipmask="" name="buyer-01"
        password="ZGthcGVsbA==" sourceipaddress="192.168.100.95"
        sourceipmask="255.255.255.255"/>
    <ReservationManualStrategy name="Reservation Strategy"/>
    <ResourceAgentURL name="qa-resource-resv01">
        http://192.168.100.44:8505/merkato/qa-resource-resv01/
    </ResourceAgentURL>
    <ResourceAgentURL name="qa-resource-resv02">
        http://192.168.100.44:8505/merkato/qa-resource-resv02/
    </ResourceAgentURL>
    <Selections resourceAgent="qa-resource-resv01"
        strategy="Reservation Strategy"/>
    <GarageURL name="reservation-garage">
        http://192.168.100.44:8505/merkato/reservation-garage
    </GarageURL>
    <Units currencyUnit="$" quantityUnit="Mbps" timeUnit="month"/>
    <ReservationBid confirmed="true" percentCancellationFee="10.0"
        sessionId="56" type="bid">
        <Price displayUnit="$/month/Mbps" displayValue="304.85"
            name="price" value="1.17611879931E-8"/>
        <ReservedQuantity displayUnit="Mbps" displayValue="5.0"
            name="quantity" value="5000.0">
        <Time name="start" time="now"/>
        <Duration displayUnit="month" displayValue="0.0" name="duration"
            value="900000.0"/>
        </ReservedQuantity>
        <Value displayUnit="$/month" displayValue="1524.25" name="value"
            value="5.8805939965500004E-5"/>
        <Cost displayUnit="$" displayValue="0.0" name="fee" value="0.0"/>
    </ReservationBid>
    <Allocation allocId="1033677704" percentCancellationFee="10.0"
        reservationId="26562491" sessionId="56">
        <BuyerIdentity addresstype="ip" destinationipaddress=""
            destinationipmask="" name="buyer-01"
            sourceipaddress="192.168.168.95"
            sourceipmask="255.255.255.255"/>
        <SellerIdentity name="qa-seller01" principalid="PRINCIPALID"/>
        <Price displayUnit="$/month/Mbps" displayValue="304.85"
              name="price" value="1.17611879931E-8"/>
        <ReservedQuantity displayUnit="Mbps" displayValue="5.0"
              name="quantity" value="5000.0">
              <Time day="3" hour="20" min="37" month="October"
                    msec="708" name="start" sec="4"
                    time="1033677424708" year="2002"/>
              <Time day="3" hour="20" min="52" month="October"
                    msec="708" name="end" sec="4" time="1033678324708"
                    year="2002"/>
        </ReservedQuantity>
        <Cost displayUnit="$" displayValue="0.53" name="cost" value="53"/>
<Cost displayUnit="$" displayValue="0.0" name="fee"
      value="0.0"/>
        <Value displayUnit="$/month" displayValue="1524.25" name="value"
              value="5.8805939965500004E-5"/>
</Allocation>
<Allocation allocId="1033677704" percentCancellationFee="10.0"
      reservationId="26562486" sessionId="56">
        <BuyerIdentity addresstype="ip" destinationipaddress=""
              destinationipmask="" name="buyer-01"
              sourceipaddress="192.168.168.95"
              sourceipmask="255.255.255.255"/>
        <SellerIdentity name="qa-seller01" principalid="PRINCIPALID"/>
        <Price displayUnit="$/month/Mbps" displayValue="290.53"
              name="price" value="1.120879994481E-8"/>
        <ReservedQuantity displayUnit="Mbps" displayValue="3.0"
              name="quantity" value="3000.0">
              <Time day="3" hour="20" min="36" month="October" msec="88"
                        name="start" sec="48" time="1033677408088"
                        year="2002"/>
              <Time day="3" hour="22" min="36" month="October" msec="88"
                        name="end" sec="48" time="1033684608088"
                        year="2002"/>
        </ReservedQuantity>
        <Cost displayUnit="$" displayValue="0.0" name="fee" value="0.0"/>
        <Value displayUnit="$/month" displayValue="871.6" name="value"
              value="3.362639983443E-5"/>
        </Allocation>
        <Allocation allocId="1033677704" percentCancellationFee="10.0"
                reservationId="26562481" sessionId="56">
                <BuyerIdentity addresstype="ip" destinationipaddress=""
                        destinationipmask="" name="buyer-01"
                        sourceipaddress="192.168.168.95"
                        sourceipmask="255.255.255.255"/>
                <SellerIdentity name="qa-seller01" principalid="PRINCIPALID"/>
                <Price displayUnit="$/month/Mbps" displayValue="300.0" name="price"
                value="1.15740738E-8"/>
                <ReservedQuantity displayUnit="Mbps" displayValue="1.0"
                        name="quantity" value="1000.0">
                        <Time day="3" hour="20" min="36" month="October"
                                msec="728" name="start" sec="26"
                                time="1033677386728" year="2002"/>
                        <Time day="3" hour="21" min="36" month="October"
                                msec="728" name="end" sec="26"
                                time="1033680986728" year="2002"/>
                </ReservedQuantity>
                <Cost displayUnit="$" displayValue="0.0" name="fee" value="0.0"/>
                <Value displayUnit="$/month" displayValue="300.0" name="value"
                        value="1.15740738E-5"/>
        </Allocation>
</Buyer>
```

Authentication and Security

Every XML call is authenticated. Each call contains the username and password of the player being controlled:

```
<Identity name=“name” password=“password”/>
```

The username and password are in plain ASCII text. They uniquely identify the player controlled.

To ensure security to the user the call can be sent using SSL (Secure Socket Layer). This ensures the privacy of the user information (password and player parameters). Merkato supports SSL calls if the Merkato server is configured to enable SSL support.

Protection From Misuse of APIs and Server Overload

To avoid excessive load on the server due to misuse of the APIs, the server processes only one request per session. All requests that reach the server while it is in the midst of processing a request are dropped.

For example, if an application sends requests to the servers without pausing, the server drops all requests while it is processing a request. It logs an error, which identifies the agent sending the excessive requests.

Troubleshooting

Internally, Merkato only takes the `<methodCall>` element—the value of the HTTP request parameter call. For the garage to respond, it has to know the following:

Are you authorized?	Include a <BuyerIdentity> element.
What should the agent in the garage do?	Include an <action> element.
Should player's internal state be changed?	Encode the player with the elements listed in “Recognized Elements.”

Error Handling

If the XML server encounters an error condition that prevents it from processing the current request, it emits a tag. The tag is an XML element:
`<Exception `***`name`***`=“`***`text_string`***`”/>`.

The ***`name`*** in the tag depends on the type of error call. The text string provides a description of the error encountered.

Client applications must be prepared to receive and handle the error tag at any time. The client application can include logic for deciding whether to discard or retain the information.

Software Development Kit

The software development kit contains:

- This manual in PDF format
- Perl Module for Controlling Merkato Buyer Agent via API, containing standard files

In addition, you will need to obtain the following libraries and Perl modules:

- libraries:
  - expat (may be obtained from http://www.jclark.com/xml/expat.html)

- perl modules: (may be obtained from http://www.cpan.org/)
  - XML::Parser >= 2.30
  - XML::Simple >= 1.08
  - libwww >= 5.65
  - URI >= 1.12

Use the following commands to install a Perl module:

```
tar -zxvf modulename.tar.gz
cd modulename
perl Makefile.PL
make
make install (as root)
```

Use the following command to install a CPAN Perl module

```
perl -MCPAN -e 'install XML::Parser'
```