An Infrastructure for Meta-Auctions

C. Bornhövd, M. Cilia*, C. Liebig, A. Buchmann
Dept. of CS, Darmstadt University of Technology
Darmstadt, Germany, D-64283
{*bornhoev, cilia, chris, buchmann*}*@informatik.tu-darmstadt.de*

Abstract

Auctions have been a popular trading paradigm for centuries but have gained new interest through world-wide trading on the Internet. Many B2C sites are embracing reverse auctions as an additional service to registered customers. In all these cases the efficient notification of the participants is essential. In this paper we develop the notion of a meta-auction that allows a potential buyer to roam automatically across auction sites and we identify critical communication and notification requirements of the next generation of Internet-scale trading systems: First, today's information systems are limited in their growth and interaction potential because the typical client-server and n-tier system architectures are solely based on a request/response interaction; second, the user-initiated query metaphor from the database domain is the primary means for information acquisition; and third, many assumptions about the meaning of data and notifications provided and exchanged through the Internet are left implicit.

We argue that Internet-scale business applications require publish/subscribe as an additional interaction paradigm, should leverage proactive information dissemination and caching mechanisms, and that there is a compelling need for metadata-based infrastructures providing common vocabularies for semantically meaningful exchange of data and notifications. We illustrate these points through examples from the auction domain and the development of the meta-auction concept.

1. Introduction

Auctions are a popular trading mechanism when multiple buyers compete for scarce resources. Famous auction houses, such as Sotheby's or Christie's for art or high-priced collectibles come immediately to mind. The advent of auction sites on the Internet, such as eBay, Yahoo or ricardo.de has popularized the auction paradigm and has made it accessible to a broad public that can trade in a consumer to consumer interaction anything from beeney-babies to electronics and from comics to vintage fountain pens. The mechanism has become so popular that many e-businesses are now offering a reverse auction mechanism as an additional service to their registered customers.

Serious art collectors have used similar services for centuries. Agents or gallery owners notify a potential buyer whenever an article that might interest a customer becomes available. In the more mundane world of Internet-auctions collectors would like to enjoy a similar service that would alert them whenever a certain object comes on the market. In addition, a collector might prefer to deal with one common auction portal instead of registering her interests with multiple auction sites. Therefore, we introduce the notion of a meta-auction. A meta-auction allows a potential buyer to roam automatically and seamlessly across auction sites for auctions and items of interest.

To realize the meta-auction several problems must be solved. We argue that today's systems that are based primarily on user-initiated communication are not adequate and will not scale properly. The large number of interconnected users and systems, as well as their wide-area distribution imposes particular restrictions with respect to response times and network bandwidth. Internet-scale information systems therefore must leverage proactive information dissemination and caching techniques. However, typical client/server and n-tier system architectures are merely based on a request/response interaction and do not take into account the asymmetric nature of such systems [1, 16], where the significant data flow is from a (database) backend-tier which stores operational data – such as items to be sold – to the application-tier which then provides access for end-users through a Web gateway.

Furthermore, the query metaphor from the database domain is currently the primary means for information acquisition, which results in the user polling for changes and happenings of interest. We argue, that notifications about events, such as the placement of a highest bid, and their timely delivery to the user represent valuable information. Therefore, publish-subscribe as an additional interaction paradigm is needed to make the efficient dissemination of process related information possible.

* Also ISISTAN, Faculty of Sciences, UNICEN, Tandil, Argentina.

Each site participating in the meta-auction system provides information about items and the auction process but does not share a global data schema nor may we assume a global schema for notifications. Still, all participants come from the same application domain and at least conceptually, share a common – domain-specific but participant-independent – vocabulary. While in most of today's systems the vocabulary is left implicit, we propose an ontology-based infrastructure for explicit metadata-management on top of which the meta-auction service can be realized. The suggested ontology-based infrastructure provides common vocabularies for semantically meaningful exchange of data and notifications, and supports incremental integration of participating information systems as needed.

In Section 2 we describe the mechanics of a typical auction, and introduce the meta-auction service. Section 3 describes the architecture of a meta-auction site. In Section 4 we address the semantic issues and present an ontology-based framework for creating and managing the needed common vocabulary. Section 5 puts it all together. Section 6 addresses related work while Section 7 summarizes and provides an outlook.

2. Meta-Auctions

Person-to-Person online-auctions, as provided by services like eBay, ricardo.de, and Yahoo look all very similar. Typical entities involved are Seller, Item, Bidder and Bid. *Items* represent those objects that are on sale, each of them is described using a unique id, a title, a description and a picture. Items are classified using a hierarchical *category* organization, like those depicted in Figure 1.

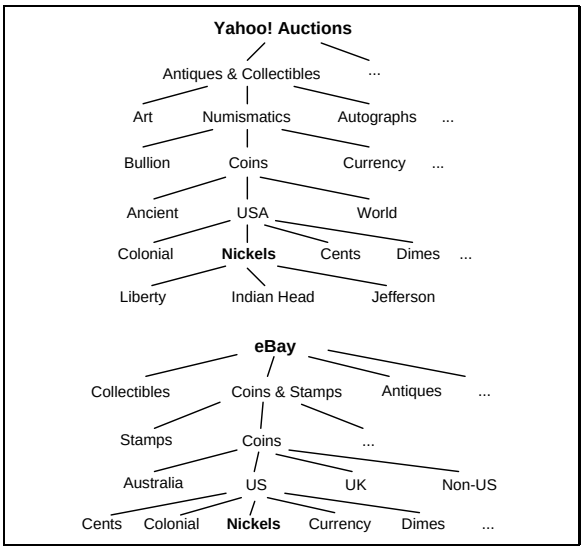

Figure 1. Item categories.

A typical use case for an auction site consists of the following steps: The user searches or browses the categories looking for an item of interest. Once the item is found she can track the auction process by continuously polling the auction site's server. Notice that reaching the deadline of the auction could take some days. In an ascending-price or English auction bidders increase the price during the course of the auction until its deadline. When the auction deadline is reached, the bidder whose bid is the highest "wins". If the minimum price has not been reached, the item might not be sold and the bidder must be notified.

Now consider the case of a collector. With the current auction sites, she has to manually search for the item of interest, possibly visiting more than one auction site. If successful, she might end up being engaged in different auctions at multiple auction sites. There are two obvious shortcomings to this approach: First, the user must poll for new information and might miss the window of opportunity, and second, the user must handle different auction sites with different category setups and different handlings. This motivates the need for the meta-auction broker.

The meta-auction broker provides a unified view of different auction sites and services for category browsing, item search, auction participation and auction tracking, as depicted in Figure 2.

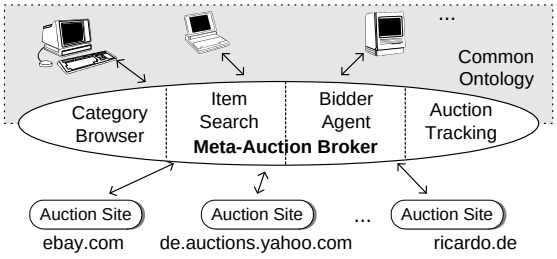

Figure 2. Meta-Auction Broker.

The realization of such a broker service is a major challenge, due to the large number of users, the expected wide-area distribution of the cooperating information systems and the dynamic nature of interconnected systems. Web caching techniques alone are not adequate, as most of the operational data is kept in backend-data stores. Web server content is generated by the application gateways and is thus frequently changing. Therefore, site replication using proactive information dissemination is a better approach.

While user interactions such as placing an offer, placing a bid, or browsing categories are well supported by the request/reply nature of the Web. The above scenario reveals a fundamental weakness of today's WWW infrastructure, namely, the missing support for a publish/subscribe paradigm and timely notifications. In fact, events that arise in the context of an auction process should be treated as first class information and propagated as notifications to the users who subscribed to the event. Propagation of events leads to a useful and efficient non-polling realization of an auction tracking service.

Finally, we identify the need to cope with heterogeneity and the goal to provide a unified view as well as unified

access to different participating auction sites. Today, the exact meaning of terms, entities and notifications used by different auction sites is still left implicit. To enable the brokering between different participating auction sites the precise understanding of the terms used by each site is needed and should be made explicit through a domain-specific common vocabulary.

3. Meta-Auction Broker Architecture

The meta-auction system should be seen as a cooperative information system where autonomous sites participate [10].

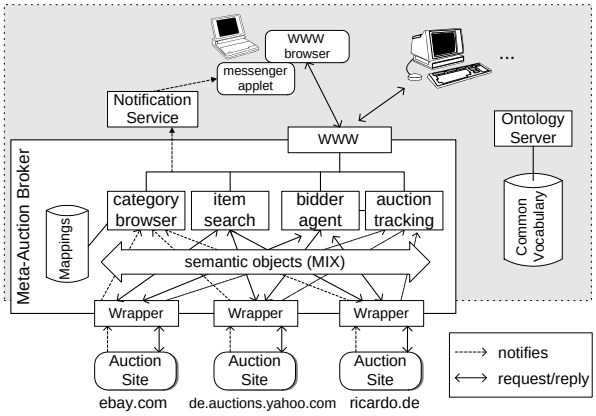

Figure 3. Infrastructure for Meta-Auctions.

One of the main challenges of cooperative information systems is to provide communication protocols that allow efficient and transparent cooperation in distributed heterogeneous environments. In the following we present core technology for proactive information dissemination, publish/subscribe based event notification and ontology-based meta-data management that should be in place to support a scalable and feasible solution for a meta-auction system shown in Figure 3.

3.1. Information Dissemination

The scalability of a meta-auction cooperative information system depends on its component systems. Given that the operational data, like item data, auction data, bids, etc. is typically stored in database backends, WWW caching techniques alone are not sufficient to provide response times and availability with adequate quality of service. In such systems the significant data flow is from a (database) backend-tier to the application-tier which then provides access for a large number of end-users through a WWW gateway. The resulting information-dissemination based architecture consists of a multi-tier multi-sited auction service, where multiple (quasi replicated) application servers together with the corresponding WWW gateways access operational data through a push-based CORBA Persistent State Service (PSS) [23] (see Figure 4).

The CORBA PSS prototype architecture makes use of a commercial multicast-enabled publish-subscribe MOM (Message Oriented Middleware) [30]. Instead of addressing by location (i.e., IP number, DB listener socket address), publish-subscribe interaction uses the generic communication paradigm of *subject based addressing* [27]. Publishers send messages under specific subjects. Every listener that has subscribed to that subject will receive the messages published under the respective subject.

In [17], the nodes in a general distributed information system are classified into: i) *data sources* which provide the base data that is to be disseminated, ii) *clients* which are net consumers of information and iii) *information brokers* (mediators) that acquire information from data sources and provide the information to the clients. Data delivery mechanisms are distinguished along three main dimensions: push vs. pull, periodic vs. aperiodic and 1:1 vs. 1:n. Based on these classification, the CORBA PSS that we implemented provides proactive information dissemination as follows:

1. The implementation of the PSS uses a hierarchy of subject names to address objects. Several data sources may be federated in a single data source domain.
2. The PSS on the CORBA side interacts with the data sources at the backend in *aperiodic pull* combined with 1:n delivery. A persistent state object lookup request is initiated on application demand. The response is *published* by the DB Connector under an associated *subject* and all PSS instances that have *subscribed* to that kind of object will snoop the resulting messages and possibly refresh their object cache.
3. Updates to persistent state objects result in *publishing* update notifications under an associated subject including the new state of the object, i.e., *aperiodic push* combined with *1:n* delivery. Again, the PSS instances snoop the update notifications to refresh the object cache and notify the application of the update.
4. In addition to update notifications, creation and deletion events can be signaled to the application by letting the PSS snoop the respective messages. The application is thus relieved from polling and, equally important, may extend the chain of notification to the client-tier in order to facilitate timely information delivery.

Given the potential distribution of auction sites we expect to benefit from reference locality not only in the scope of a single PSS instance but because of the snooping of load replies and update notifications we benefit from reference locality throughout the datastore domain across all sites of the auction service provider. Thus the operational data is proactively disseminated to the auction sites.

The proposed architecture is integrated with a notification service. Notifications are reifications of events

and event context. An event is a happening of interest and represents process related information. Event context provides supplementary information to understand the circumstances under which the event was raised and to support decision making for appropriate reactions by the consumer. The notification service leverages the publish-subscribe paradigm for loosely coupling event producers and event consumers. Notifications are published under a subject and propagated to subscribed consumers who are automatically notified. Notification context may carry data objects by value or reference. References may be implemented by a backchannel address to the data source, for example a URL or an IOR to a CORBA object.

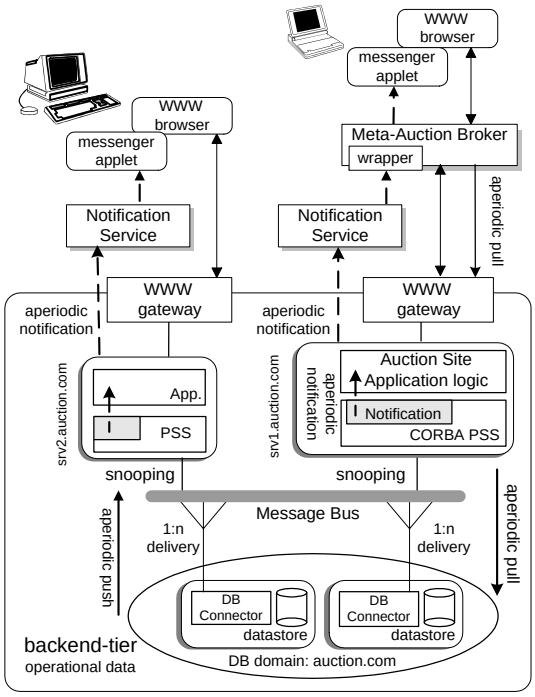

Figure 4. Information dissemination.

The use of a notification service for publishing auction related events reduces the load of auction sites that otherwise would be swamped by continuously polling clients. For example, subscription to creation of new bids and insertions of new items of a specific category are possible and when the event occurs will result in a notification to the subscribing user or application (i.e., the meta-auction broker). Thereby, timely notifications can be provided as first class information to the user of an auction site, who in turn may react appropriately.

4. Using Ontologies for Data Exchange

To represent notifications, message content, or data in general, in an unambiguous way, we use domain-specific ontologies. An ontology provides an agreement about a shared set of terms, or concepts, of a given subject domain [20]. The concepts given in the ontology provide a common vocabulary for which no further negotiation concerning their meaning is necessary.

In this way, an ontology can serve as a common basis for the interpretation of data and notifications. This provides a prerequisite for semantically meaningful information exchange between a frequently changing set of independent participants in a large-scale business scenario, like the auction scenario we described here. In particular, common vocabularies are needed on three levels in our scenario:

• **User level.** For communication between independent participants, i.e., participants that may have never met before, we have to establish some kind of common vocabulary. Auction sites approach this by providing a categorization of possible auction items. However, the categories vary among auction sites as shown in Figure 1. For a meta-auction we must provide for proper mappings among category hierarchies. The same is valid for the representation of all the descriptive information, such as price, deadline, etc.

• **Infrastructure level.** Our infrastructure is based on the exchange of notifications and data created from different components. For example, in our framework we may have to deal with messages from different auction sites being totally different in physical representation and terms used, however all "saying" that a user was outbid by someone else. To make the messages compatible, the underlying assumptions regarding structure and organization must be made explicit as additional metadata.

This metadata has to be based on a common vocabulary, or ontology, to support its semantically correct interpretation. Otherwise we will face the diversity problem on the metadata level. Such a common description basis has to be managed independently of participating data providers or consumers and should be extensible.

• **Heterogeneity at the data store level.** Relevant data in the auction system comes from different auction sites which may provide heterogeneities with respect to structure and semantics. The available data has to be integrated to be usable by a broker. Different auction sites can be integrated into the system by mapping them to a common representation model based on a shared ontology.

This mapping resolves heterogeneities with regard to organization and structure of the data, and the use of different terms referring to the same real-world aspects. In addition, metadata is added to the available data to make implicit modeling assumptions concerning organization and meaning explicit. Based on this representation heterogeneities in the semantics of the data, e.g., use of different units of measure, scale factors, derivation formulas, coding, or naming schema, can be resolved by the system at query time. The meta-auction broker then sees the available data and notification based on a common representation (with additional metadata), and based on a common vocabulary. The use of semantic metadata for the integration of data

from heterogeneous data sources is discussed in more detail in [7].

4.1. MIX – An Ontology-based Integration Model

Data and messages from different auction sites are mapped to an ontology-based representation model called MIX where terms from the shared vocabulary are used to make their intended meaning explicit to provide a common interpretation basis. MIX (*Metadata based Integration model for data X-change*) is a self-describing data model in the sense that structure and semantics of the data is given as part of the available data itself. This makes transferred data self-contained and allows a flexible association of additional metadata [6, 7].

The model is based on the concept of a *semantic object* which represents data together with its underlying *semantic context*, which is a variable set of meta-attributes that explicitly describe implicit assumptions about the organization and semantics of the represented data object. For example, the use of different currencies like US Dollar or Euro for item prices, or different time zones and formats for time and date values can be described by the context of a semantic object. A semantic object may represent any object that is relevant to an auction such as a given notification, or information content of a message to be transferred.

In addition, each semantic object has a *concept label* associated with it that provides additional information about the intended meaning of the data object. These concept labels are taken from an ontology. Thus, the concept label and the semantic context of a semantic object help to describe its supposed meaning.

In MIX we distinguish between simple and complex semantic objects. *Simple* semantic objects represent atomic data items, such as simple number values or text strings. In contrast, *complex* semantic objects can be understood as heterogeneous collections of semantic objects, each of which describes exactly one attribute of the represented real-world object. These subobjects are grouped under a corresponding ontology concept. The attributes given for a complex semantic object are divided into mandatory, which may be used to identify an object belonging to a given concept, and additional (optional) attributes. Thus semantic objects belonging to the same ontology concept may have different sets of attributes.

For example an auction item in our system can be represented as a complex semantic object of concept *AuctionItem* shown by the semantic object of Figure 7. Mandatory attributes, i.e., *AuctionSite-Identifier* and *ItemIdentifier*, have been underlined. Optional attributes like *ItemCategory*, or *ItemPicture* may not be given for each *AuctionItem* object (depending on the information made available by the respective site). Notice that *ItemCategory* is specified according to the eBay taxonomy.

Semantic objects can be converted among different semantic contexts by using conversion functions. These functions are specified in the underlying ontology, as far as they are concept-specific but application-independent, or may be stored in an application-specific conversion library. Based on these mapping functions, semantic objects from different sources can be compared and used in combination by converting them to a common semantic context. For example, currencies can be converted or category classifications can be mapped. Thus, the MIX model is capable of representing notifications and data from different sources in a uniform way, and, by referring to a common vocabulary, on a common interpretation basis. For a more detailed and formal presentation of our representation model see [5, 6].

4.2. The Role of the Ontology Server

We use an ontology server to store and manage the common vocabulary used in our framework. This vocabulary provides the extensible description basis for the meta-auction system to which wrappers and interactive users refer.

In an ideal situation, all participants should adhere to the corresponding ontology. In an imperfect real world, the vocabulary must be extensible to adapt it to changing needs on data provider or consumer side. Ontologies, as we use them, should follow existing standards if possible to enhance the chances that they will be accepted by other participants.

Aspects of the auction scenario for which no such standards or conventions exist require new concepts to be specified and registered with the server. To ensure consistency of the ontology, new concepts have to be introduced by extending or specializing existing concepts in a predefined way to avoid ambiguous specifications or homonyms. By providing a way to extend the ontology, we believe that we can claim a reasonable combination of rigor and flexibility that makes our infrastructure applicable in real-life situations. The approach has been successfully tested with real application data from the travel domain.

We use Java classes to represent ontology concepts. This provides two main advantages: First, concepts can be made available as pre-compiled Java classes via ontology servers. Using Java as the concept specification language allows their shipping, as well as that of the corresponding MIX objects, between different platforms without any further transformations. Second, by mapping local data and notifications of an auction site to the corresponding concepts we specify how local data is mapped to Java objects which can be used, according to the semantics of the concept associated, without any additional translation. This avoids any impedance mismatch between programming language and ontology specification language. Wrapper and clients can load the respective concept specifications from the ontology server when needed.

4.3. Provider-Specific Wrappers

In our infrastructure wrappers are used to wrap data and notifications of the respective auction site in semantically corresponding MIX objects. Therefore, heterogeneities in the organization and the terms used by different auction sites are resolved, and differences in the underlying semantics, e.g., different units of measure, scaling factors, derivation formulas, or coding and naming schemas, are made explicit. These semantic heterogeneities can be resolved by the broker by using appropriate conversion functions.

Because the meaning of data and notifications is usually known locally, it is preferable that the mappings to MIX objects are specified by the institution running the auction site. However, if they are not willing to invest in the effort in order to participate in the meta-auction, wrappers can be built by the institution providing the meta-auction service by evaluating local interfaces. The wrappers do not affect either the auction site nor local users.

Wrappers are registered with the meta-auction broker and are used by it to interoperate with the available auction sites. Because they are instantiated on the same machine as the broker, the local site remains unaffected by the use of a wrapper, which appears simply as another user to it. In addition, metadata is added on the meta-auction machine and does not need to be transferred from the machine on which the respective auction site operates.

Wrappers are supposed to provide three services in our architecture. First, they map global requests for auction-related information concerning *AuctionItem*s, etc., to corresponding queries for the local provider, and in return present answers in form of MIX objects back to the broker. Second, they provide a unified interface for registering a user with an auction site. Third, they wrap/filter/compose provider-specific notifications to the corresponding MIX objects, and provide ontology-based subscription. Each of these functions is discussed in the following subsections.

Mapping MIX Objects to Local Queries, and Local Data to MIX Objects. Wrappers have to map global information requests, e.g., requests concerning an item, the scheduled start and end times of an auction process, or a seller. Such requests have to be expressed based on the common vocabulary as queries processable by the local auction site. For this mapping, the wrapper first has to translate the global vocabulary, that is the terms from the common ontology to the corresponding local terms. To access the actual data it has to use the specific interface of the underlying auction site. The underlying sites in most cases only allow the use of forms.

An example for an item request given in a SQL-like notation is shown below where additional information, in particular *ItemDescription*, *ItemLongDescription*, and *Picture*, about a certain *AuctionItem* is requested.

```
select  ItemDescription, ItemLongDescription, ItemPicture
from    AuctionItem
where   AuctionSiteIdentifier = "eBay"  and
        ItemIdentifier = "217316338"
```

The wrapper has to wrap the answer to a request into a MIX object which is then returned to the broker. This comprises the mapping of local terms and structures to corresponding ontology concepts and representation constructs of MIX. In addition, metadata is added that explicitly describes the semantic context, e.g., *TimeZone* and *TimeFormat*, or *Currency*. Since, this metadata is represented as an integral part of a MIX object it is shipped with the object and directly accessible at client side. For example, a given *AuctionItem* may look like the objects shown in Figure 7.

For both directions, i.e., from global requests to local queries and local data to MIX objects, the underlying mapping rules are specific for the auction site and coded in the programming logic of the respective wrapper component.

Wrapping Notifications. As far as possible, the event-driven logic of the meta-auction broker should be implemented in an auction site independent way. Therefore, the notification service wrapper realizes the abstraction of *concept-based addressing*. By subscribing to a notification concept defined in the common vocabulary, the auction tracking service layer is able to receive notifications like for example *NewItemOfInterest*, *AuctionBegin*, *NiceTry*, *OutBid*. The wrapper maps a subscription to a concept into subscriptions to notifications. On incoming notifications, the wrapper transforms the auction-site dependent notifications to provider-independent semantic objects. In our ontology, we classify the notifications as shown in Figure 5.

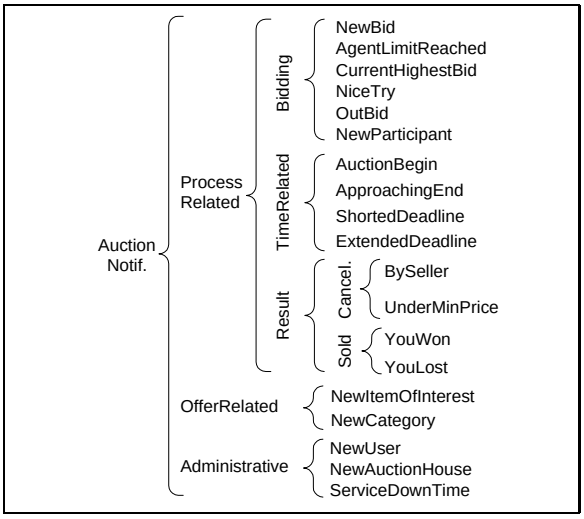

Figure 5. Classification of Notifications.

We assume that a basic Internet notification service will provide publish-subscribe subject-based addressing functionality, as quasi standardized by the JMS specification [21]. Typically, the subject namespace is hierarchically struc-

tured and subject patterns are supported, as for example in [30, 22]. We suggest to add more advanced services such as filtering of notifications by content and to compute notifications which represent composite events (sequence, time-relative, and, or, not) [25, 23]. The wrapper then uses an event-driven rule engine, similar to the concept of Event-Condition-Action rules in active databases [11], to transform notifications into semantic (notification) objects.

With respect to subscription mapping we distinguish two major cases, direct mapping and complex mapping. *Direct mapping* is possible if the auction site announces a notification which directly supports the addressed concept of the notification ontology. The notification might be more general than the addressed concept, in which case the wrapper must install a filter. For example, the auction site might raise "end of auction" events when an auction is closed and add more details about the outcome of the auction process in the notification context. If the user wants to subscribe to the *Sold::YouWon* concept, a filter on the context data of the notification must be used to discriminate between more specific concepts.

Complex mappings are needed, if the addressed notification concept is not directly supported by the auction site, but the concept may be composed out of different notifications. For example, the auction site might announce specific notifications for new items per category or distinguish between commercial offers and private offers. As the notification ontology only provides a rather general concept *NewItemOfInterest* the wrapper must subscribe to all of the more specific notifications and signal the event if any of the notifications arises using an OR-composition-operator. Complex mappings can also be constructed using locally generated events. This is especially useful in conjunction with time-related notification concepts, which might not be supported by the auction-site but can be generated locally, for example to provide subscription to *ApproachingEnd* or subscription to *AuctionBegin*.

In all of the above cases, the wrapper must compose a subject-name for subscription with the auction-site. In the simplest case, the auction site uses a flat, unstructured subject namespace. It is to be expected, however, that the subject-namespace is structured. "end of auction" notifications might include the auction-site, category-id, item-id, notification type and bidder-id in the subject name as shown in Table 1. As you can see, some context information is coded into the subject name. This allows the notification service to efficiently route and filter notifications for delivery to the potential large number of subscribers and provides the consumer with a simple yet powerful subject-pattern based subscription mechanism.

Note, that the subscription mappings might not be complete, in the case that a notification concept is not supported by the auction site in any way. The meta-auction broker should inform the user about incomplete mappings.

The second service of the wrapper with respect to notifications is to transform the notification context into a MIX object. The values to fill the attributes of the MIX object may be extracted from the subject name, be shipped as payload data in the notification context or may be hardcoded in the wrapper. The payload data may represent data objects by value or by reference. In the first case, the context data is wrapped into semantically corresponding MIX objects. In the second case, the needed data to fill the attributes of the MIX object is not shipped with the notification but must be acquired by a request through a given backchannel.

ebay.com.4101.217316338.EndofAuction.6603	
Ts	1999/12/17 16:07:35
Item-id	217316338
URL	264.71.201.176/auction/217316338
Seller-id	3213
Price	215.50
Comment	Congratulations ! Pay quickly !
...	...

Table 1. Site-specific notification.

The example in Table 1 shows an "end of auction" notification for item 217316338 in category 4101 to the buyer 6603, which indicates that the user has "won" the auction. In order to receive this notification the user must have subscribed to the concept *YouWon* for the respective auction and the wrapper in turn has subscribed for subject `ebay.com.4101.217316338.EndofAuction.6603`. When receiving this notification, the wrapper has to go back to the given URL in order to retrieve the data needed to create the *AuctionItem* MIX object which is part of the *YouWon* notification presented to the user. The composed MIX object is shown in Figure 6.

5. Putting it All Together

To identify an item of interest the most convenient way for the user may be to use the category browsing service of the meta-auction broker. This service provides a handy way for an interactive user to find an item she wants to bid for by browsing through a tree of object categories. The auction broker provides different views to the set of available items according to the category taxonomies of the auction sites participating. Figure 1 shows a clipping of the category taxonomies of eBay and Yahoo, respectively. This allows a user to apply the category taxonomy of the local auction site she is familiar with.

In addition, the meta-auction broker provides a global categorization of items available via the sum of participating providers. This broker-specific classification should comprise local taxonomies to be a single entry point to all accessible categories.

When the user of the meta-auction service looks for a certain product, e.g., baseball cards, she may find her way through either a provider-specific, or the global taxonomy.

```
< YouWon, {
    < AuctionSiteIdentifier, "eBay",          {<AuctionSiteIdentifierCode, "FullSiteName">} >,
    < CreationTime, "17/12/99 16:07:35",      {<DateFormat, "DD/MM/YY HH:MM:SS">} >,
    < ReceptionTime, "17/12/99 16:07:36",     {<DateFormat, "DD/MM/YY HH:MM:SS">} >,
    < AuctionItem, {
        < AuctionSiteIdentifier, "eBay",      {<AuctionSiteIdentifierCode, "FullSiteName">} >,
        < ItemIdentifier, "217316338" >,
        < BuyerIdentifier, "6603" >,          {<IdentifierCode>, "eBayCode"} >,
        < SellerIdentifier, "3213" >,         {<IdentifierCode>, "eBayCode"} >,
        < Price, 215.50,                      {<Currency, "USD">, <Scale, 1>} >      }> }> }>
```

Figure 6. MIX representation for YouWon notification.

When she ends up at the most specific category she expects the item to be found in, she gets the sum of all items from the participating auction sites that provide items of this category. For example, Sports Memorabilia. Thereby the meta-auction broker can provide a unified interface to all available auction sites.

To do so, a mapping from a "leaf" category of a specific taxonomy to the corresponding categories of other participating auction sites is required. Within this mapping we have to distinguish between four cases.

1. In the simplest case, a given "leaf" category directly corresponds to a "leaf" category of another auction site. Items from this category have to be displayed to the user. However, in general we will not have such a simple one-to-one mapping.
2. More commonly, the selected category will correspond to a more abstract category, i.e., not a leaf category, of another auction site. In this case, all items of the sub-categories of this category have to be displayed. The more general category can be understood as a hypernym of the selected category.
3. The most difficult situation occurs when the selected "leaf" category corresponds to different categories of the classification of another site. Since we cannot further classify the items found in these categories we have to display them all. This in general leads to loss of precision, i.e., the resulting set of items may contain objects not corresponding to the selected category.
4. Finally, we may not find a corresponding category at all. That is, the other auction site provides no items of this category. In this case the respective auction site is not taken into consideration any further.

Starting from the taxonomies shown in Figure 1 a mapping table for the mapping from the meta-auction to the site-specific taxonomies can be defined as shown in Table 2. Mapping from the meta-auction Coins::Nickles to eBays Coins::Nickles provides an example of the one-to-one mapping of case 1. Case 2 is reflected by mapping Coins::Nickles from meta-auction to Coins::Nickles of Yahoo, since the latter is further subdivided. Coins::Australia from meta-auction to Yahoo, i.e., to Coins::World and Coins::Ancient, is an example for case 3. Finally, for Stamps::USA from the meta-broker to the Yahoo taxonomy is not possible, i.e., represents an example for case 4.

In addition to providing a mapping between different taxonomies, the category browser must keep track of changes in the category taxonomies of the local auction sites to reflect these changes to the user and adapt the mapping. Therefore, the browser component needs to be notified by a *NewCategory* notification from the corresponding wrapper. Since, in general, the correspondences of a new category and existing classifications cannot be derived automatically the category browser logs the notifications for later intervention by the meta-auction administrator.

Mapping Table for Meta-Auction Taxonomy			
Meta-Auction	**Yahoo!Auctions**	**eBay**	. . .
. . .	. . .	. . .	. . .
Coins::Australia	Coins::World, Coins::Acient	Coins::Australia	. . .
Coins::Nickles	Coins::Nickles	Coins::Nickles	. . .
. . .	. . .	. . .	. . .
Stamps::USA	–	Stamps	. . .
. . .	. . .	. . .	. . .

Table 2. Mapping table for taxonomies.

Another way of locating items of interest is in specifying a keyword or search text, e.g., "Baseball Cards Indians" via the item search service of the meta-auction. The broker then passes the given string to the available auction sites. These in general implement a simple pattern-based text search on available item headlines and return a (rather short) description of items matching the given search text. For example, Figure 7 shows two *AuctionItems* given by two different auction sites.

The resulting data is then wrapped by the corresponding wrappers and given back to the broker as MIX objects of concept *AuctionItem*. Notice that different auction sites in general may describe different aspects of an item. Thus, *AuctionItem* objects from different sites may have different attributes. However, they are represented based on a common vocabulary and provide additional metadata added. The first object given in Figure 7, for example, provides information about the time left until the end of auction, and a picture of the item which is not given in the second. On the other hand, the second object provides information concerning the deadline of the auction. In addition, both objects are represented with regard to different contexts, i.e., refer to different category schemas, and currencies.

The item search service will then integrate the given MIX objects by converting them to a common context with

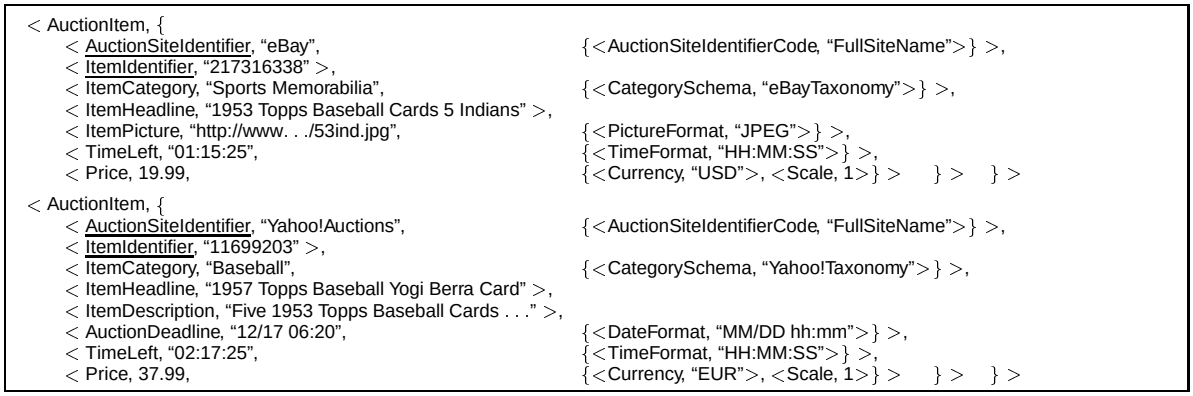

Figure 7. Two AuctionItem objects from different auction sites.

regard to currency, naming conventions, etc. The item search thus provides an integrated view on the set of items available through different auction sites.

After a user has identified an item she wants to bid for she has to join the corresponding auction process of the corresponding auction site. This means she first has to register with the performing auction site. For this the auction tracking service provides a unified interface to all participating auction sites. To register the user has to specify an *Actor-Name* and a provider-specific password.

The user can now get additional information concerning the available auction sites, ongoing auction processes, bidding histories, etc. by using the auction proxy service of the meta-auction broker. The auction proxy service again provides a unified view on available auction sites, auction processes, and other actors like sellers in the form of corresponding MIX objects.

In addition, to track an item during an auction process, for example to ascertain that another bidder has reached a highest bid, his agent was outbid, or that the deadline of an auction is approaching, the auction tracking service allows the user to subscribe for certain notifications. The user sees notifications, like process- and offer-related notifications as identified in Figure 5, based on the common ontology. Thus, we call this subscription service *concept based subscription*. For example, an item sold notification that says that the deadline of an auction process was reached and the bidder who receives the notification is the "winner" of this auction can be represented as shown in Figure 6.

6. Related Work

The research issues covered by this work span a variety of research domains and technology. A broad and in-depth discussion is out of the scope of this paper.

The significance of online auctions for e-commerce has been investigated in [12]. More insight on a particular successful auction-site can be found in [29].

To provide infrastructure for caching in the WWW has been the topic of many research efforts for some time [14, 18]. However, the case for efficiently accessing operational data is not appropriately covered by these efforts. Our work regarding the push-based PSS shares many ideas with the research on broadcast disks and information-dissemination systems [1, 16] and combines them with active database features [15, 11]. In contrast to the more powerful content-based subscription for information dissemination [17, 2] our approach makes use of the application domain knowledge about process related events to proactively disseminate data and therefore a notification service using subject-based addressing schemes [21, 27] is sufficient. Numerous notification services that provide advanced features such as filtering, content transformation and event composition have been proposed and prototypically developed, e.g., [25, 19, 28, 22, 24]. Commercial MOM products are available that provide means for message content filtering and transformation [30, 31]. However, none of them provide the paradigm of concept-based addressing and leave the semantic of notifications and messages implicit.

With regard to using ontologies as a common basis for data integration a number of projects have been carried out. Among them are Carnot [13] or the follow-on project of InfoSleuth [4], SIMS [3], OBSERVER [26], and COIN [9] to mention only a few. The most significant difference between these systems and our approach lies in representing and managing common vocabularies. They all use logic-based representation languages to represent concepts and relationships between them. In contrast, we use Java classes. Thus, the resulting data objects are represented as class instances in which semantics and representation are determined. Furthermore, because semantic metadata is part of the resulting objects, it is directly available to the application as additional object attributes.

7. Current Status and Outlook

The need for the kind of services described here becomes apparent from the fact that while we are writing this paper Yahoo! began offering its messenger service. However, Yahoo's service is limited to a single auction site and much

more limited in scope. We have detailed in this paper the infrastructure for a more ambitious approach and are convinced that such an approach can be successful.

This paper reported work in progress on the meta-auction. At present, the infrastructure described in the paper has been implemented. In particular, we have implemented a notification service based on TIBCO's messaging middleware and our own implementation of the CORBA PSS based on the publish-subscribe paradigm. As infrastructure for the integration of heterogeneous information from multiple sources we have developed the MIX model and implemented the MIBIA environment for representation of ontologies and integration of data obtained from the Internet.

The concept of the meta-auction has been developed and integration of the major components is under way. A first proof of concept implementation will necessarily have limited functionality as no cooperation can be expected from established auction sites.

We described the infrastructure for integration of multiple auction sites. More work is needed in the analysis of specific sites, the development of rules for the integration of notifications from multiple sites, and all this while the current auction sites evolve at the typical Internet pace.

The problems discussed here in the context of auctions are definitely not limited to this domain. We argue that this infrastructure is beneficial to any large-scale e-business in which the components can act as cooperative information systems. Therefore, while motivated by a concrete application scenario, the underlying mechanism is applicable beyond auctions.

An extended version of this paper can be found in [8].

References

[1] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik. Broadcast Disks: Data Management for Asymetric Communications Environments. In *Proc. of SIGMOD*, 1995.

[2] M. Aguilera, R. Strom, D. Sturmann, M. Astley, and T. Chandra. Matching Events in a Content-based Subscription System. In *Proc. of ACM PODC*, 1999.

[3] Y. Arens, C. Knoblock, and W.-M. Shen. Query Reformulation for Dynamic Information Integration. *Journal of Intelligent Information Systems*, 6(2/3), 1996.

[4] R. Bayardo, W. Bohrer, R. Brice, and et al. The InfoSleuth Project. In *Proc. of ACM SIGMOD*, 1997.

[5] C. Bornhövd. MIX – A Representation Model for the Integration of Web-based Data. Technical report, Dep. CS, Darmstadt University of Technology, Germany, 1998.

[6] C. Bornhövd. Semantic Metadata for the Integration of Web-based Data for Electronic Commerce. In *WECWIS*, 1999.

[7] C. Bornhövd and A. Buchmann. A Prototype for Meteadata-based Integration of Internet Sources. In *Proc. CAISE*, 1999.

[8] C. Bornhövd, M. Cilia, C. Liebig, and A. Buchmann. An Infrastructure fo Meta-Auctions. Technical report, Dep. CS, Darmstadt University of Technology, Germany, 2000.

[9] S. Bressan, C. Goh, K. Fynn, and et al. The Context Interchange Mediator Prototype. In *Proc. of SIGMOD*, 1997.

[10] M. Brodie and S. Ceri. On Intelligent Information Systems: A Workshop Summary. *Journal of Intelligent and Cooperative Information Systems*, 1(3), 1992.

[11] A. Buchmann. Architecture of Active Database Systems. In *Active Rules in Database Systems*, chapter 2. Springer, 1998.

[12] K. Chui and R. Zwick. Auction on the Internet: A Preliminary Study. Technical report, Hong Kong University of Science and Technology, Department of Marketing, 1999.

[13] C. Collet, M. Huhns, and W. Shen. Resource Integration Using a Large Knowledge-Base in Carnot. *IEEE Computer*, 24(12), 1991.

[14] B. Davison. Web caching resources. Technical report, www.web-caching.com, 1999.

[15] U. Dayal, A. Buchmann, and D. McCarthy. Rules are Objects too: a Knowledge Model for an Active, Object-Oriented Database System. In *Proc. of OODBS*, 1988.

[16] M. Franklin and S. Zdonik. A Framework for Scalable Dissemination-Based Systems. In *Proc. of OOPSLA*, 1997.

[17] M. Franklin and S. Zdonik. "Data in your Face": Push Technology in Perspective. In *Proc. of SIGMOD*, 1998.

[18] J. Gettys, T. Lee, and H. Nielsen. Replication and Caching Position Statement. Technical report, W3C, 1997. www.w3.org/Propagation/Activity.html.

[19] R. Gruber, B. Krishnamurthy, and E. Panagos. High-Level Constructs in the READY Event Notification System. In *SIGOPS Euroean Workshop on Support for Composing Distributed Applications*, 1998.

[20] T. Gruber. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. *International Journal of Human-Computer Studies*, 43(5/6), 1995.

[21] M. Hapner, R. Burridge, and R. Sharma. Java Message Service, Version 1.0.2. Technical report, Java Soft, 1999.

[22] C. Liebig, B. Boesling, and A. Buchmann. A Notification Service for Next-Generation IT Systems in Air Traffic Control. In *GI-Workshop: Multicast – Protokolle und Anwendungen*, Braunschweig, Germany, 1999.

[23] C. Liebig, M. Cilia, M. Betz, and A. Buchmann. A publish-subscribe CORBA Persistent State Service Prototype. In *Proc. of Middleware*, 2000.

[24] C. Liebig, M. Cilia, and A. Buchmann. Event Composition in Time-dependent Distributed Systems. In *CoopIS*, 1999.

[25] C. Ma and J. Bacon. COBEA: A CORBA-based Event Architecture. In *Proc. of COOTS*, 1998.

[26] E. Mena, V. Kashyap, A. Sheth, and A. Illarramendi. OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies. In *Proc. of CoopIS*, 1996.

[27] B. Oki, M. Pfluegl, A. Siegel, and D. Skeen. The Information Bus – An Architecture for Extensible Distributed Systems. In *Proc. of SIGOPS*, 1993.

[28] O. M. G. (OMG). CORBA Notification Service. Technical Report OMG TC Document telecom/99-07-01, OMG, 1999.

[29] D. Prince. *Online Auctions at eBay*. Prima Publishing, 1999.

[30] TIBCO Software Inc. TIB/Active Enterprise. Technical report, TIBCO Software Inc., 1999. www.tibco.com/products/active_enterprise/index.html.

[31] TSI Software. Mercator. Technical report, TSI Software, 1999. www.tsisoft.com/enterprise/products/mercator.html.