

February 2005

An Auction Reference Model for Describing and Running Auctions

Daniel Rolli

University of Karlsruhe, Germany

Andreas Eberhart

University of Karlsruhe, Germany

Follow this and additional works at: <http://aisel.aisnet.org/wi2005>

Recommended Citation

Rolli, Daniel and Eberhart, Andreas, "An Auction Reference Model for Describing and Running Auctions" (2005).

Wirtschaftsinformatik Proceedings 2005. 16.

<http://aisel.aisnet.org/wi2005/16>

This material is brought to you by the Wirtschaftsinformatik at AIS Electronic Library (AISeL). It has been accepted for inclusion in Wirtschaftsinformatik Proceedings 2005 by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.

In: Ferstl, Otto K, u.a. (Hg) 2005. *Wirtschaftsinformatik 2005: eEconomy, eGovernment, eSociety*;
7. Internationale Tagung Wirtschaftsinformatik 2005. Heidelberg: Physica-Verlag

ISBN: 3-7908-1574-8

© Physica-Verlag Heidelberg 2005

An Auction Reference Model for Describing and Running Auctions

Daniel Rolli, Andreas Eberhart

University of Karlsruhe, Germany

Abstract: With electronic auctions gaining importance, economists have increasingly gotten interested in researching the impact of variations in auction protocols. However, the infinite number of possible auction configurations and the significance of details make this a hard and tedious task. Therefore we address the issue of comprehensive description both from a conceptual and an implementation perspective. Central components that we identify serve as building blocks for an Auction Reference Model. We show how the components of the Auction Reference Model can be described in order to achieve an overall auction specification and how the descriptions are operationalized by a generic auction runtime engine¹.

Keywords: Auction Mechanisms, Auction Reference Model, Markets, Business Processes

1 Introduction

With the profound prosperity of business-to-business (B2B) procurement platforms [Bapn⁺01; Prem03], the publicity of government spectrum auctions² [VrVo01], and eBay³ as the most prominent of consumer-oriented marketplaces uniting both prosperity and publicity, auctions have gained significant importance. They have created a lasting impact on the conduct of business in practice as well as the disciplines of Economics and Information Systems. A multitude of different auction implementations is in practical use and infinite variations exist in theory. One major criterion for distinction in the employed auctions is, for example, the bidding side of market participants. In B2B procurement a buyer typically initiates an auction and potential sellers bid for a decreasing price. On consumer-oriented platforms as well as in spectrum auctions it is mostly one seller having several aspiring buyers bid. Especially on consumer marketplaces rather intuitive auction

¹ An alpha version of the software is available for download at <http://www.iw.uni-karlsruhe.de/auctionrm/>

² <http://wireless.fcc.gov/auctions/>

³ <http://www.ebay.com/>

types are encountered that often resemble classical types like, for example, the English auction⁴ or Dutch auction⁵. But particular implementations resembling the same traditional auction type may still differ in various aspects. Varying specifics sometimes appear as details but can have major impacts on the outcome of the auctions. [Roth⁺02] As market users can be quite explorative and inventive, they adapt their behavior to even the smallest features, when provided sufficient incentives. Economists want to systematically analyze the effects of particular rules on the market outcome as they have realized the significance of detail. [Neu04]

However, for economists as well as participants from the business, customer or government side, it is difficult to grasp the specification of an auction in its entirety. Not to speak of auction analysis, auction design or appropriate strategy derivation. The lack of a coherent method for representing auctions hampers progress in all of these fields. Systematically tying together all aspects of an auction's structure would enable a comprehensive description of all details and mark the stepping stone to tackling many challenges in the auction domain.

When writing about auctions and the topics related to them we follow the terminology of McAfee and McMillan. According to them "an auction is a market institution with an explicit set of rules determining resource allocation and prices on the basis of bids from the market participants." [McMc87] As bid is the specific term for a message exchanged in auctions, offer is a more general equivalent for a market message in trade negotiations [StWe03] that we will also use.

Autonomous software agents for bidding in auctions exemplify the need and major requirements for coherent auction specification. Anthony and Jennings motivate and present an "autonomous agent that can participate in multiple heterogeneous auctions" [AnJe03]. They run simulations in an electronic marketplace that implements three auction types accessible for the agent. Each type is one particular embodiment of either the English, Dutch, or Vickrey auction. Multiple instances of the auctions can run concurrently, so there are several auctions available to the agent at a time. Based on the "intelligent bidding strategy" [AnJe03] the agent decides what bid to place in which auction considering its principal's preferences and the characteristics of each auction. Anthony and Jennings state that their model and method, which work well on the simulated marketplace, can be readily extended to other auction settings. However, they point out: "In terms of the practical application of our method, one of the key problems that needs to be addressed is how well an agent can assess the type of environment⁶ it is situ-

⁴ Besides eBay also found on <http://auctions.amazon.com>, <http://auctions.yahoo.com/>, etc.

⁵ See for example <http://www.azubo.de/>

⁶ Anthony and Jennings use the term *environment* not in the microeconomic sense as, for example, Smith [Smit03] refers to the economic environment. With their *environment* they rather embrace both the institution and economic environment of Smith: "There are three types of auctions running in the environment: English, Dutch

ated in.” [AnJe03] They state that in contrast to their experiment it will be significantly more difficult for an agent to “quickly and accurately determine its environment type” in real world scenarios.

A comprehensive specification of each accessible auction mechanism would enable autonomous agents to assess the structure of an auction quickly and accurately as desired. In the software agent setting the three following major requirements for description from the auction participant perspective exist. Firstly, for software agents to adapt their strategies properly in a setting of arbitrary auctions, they require a complete specification comprising every detail. Secondly, as the description methodology should be the same for every auction an agent potentially encounters, it should allow for the specification of every possible auction. Thirdly, the description should be machine-readable for direct agent access.

We address these three issues with an Auction Reference Model (ARM) that contains and connects all details of an auction mechanism. Furthermore, for the benefit of the auction operator the model will support systematic auction implementation.

After this introduction we present related work in section 2. Section 3 outlines an example auction for subsequent illustration. The presentation and explanation of the Auction Reference Model follow in section 4. Section 5 then proceeds from the conception to concrete implementation and gradually exemplifies solutions with the introduced example auction. We conclude our work with a summary of our results in section 6 and give an outlook to future work.

2 Related Work

Some research attempts have been made to tackle the shortcomings of auction description in practice, but none of the following approaches fulfills all of our requirements as we will explain in this section.

Dividing markets into **market phases** in order to describe market processes has been quite common in the recent years of market research. [Schm⁺98; Maes⁺99; StWe03] Information, Matching, Allocation and Settlement are, for example, the phases Ströbel and Weinhardt [StWe03] identify. The different phases describe what a single offer passes through when closing a deal. However, these phases can apply independently to different offers of one market at different times. Two participants may, for example, be settling a deal, while others are just informing themselves about the market. So, in general, these phases are not applicable – from a functional perspective – to the state of a whole market itself.

and Vickrey. [...] At the start of each auction (irrespective of the type), a group of random bidders are generated to simulate other auction participants.” [AnJe03]

Communication protocols describe the message flow between actors in auctions. The Foundation for Intelligent Physical Agents (FIPA)⁷ presents an agent Interaction Protocol (IP) for the English auction⁸ and the Dutch auction⁹. Neumann [Neu04] elaborates on this aspect but brings out that IPs remain on the design level neglecting many details.

Parameterization approaches to market description list enumerations of parameters that typically all have to be specified. They are well studied by Wurman et al. [Wur⁺01] and Neumann [Neu04] for auctions and Ströbel and Weinhardt [StWe03] for the broader notion of trade negotiations. Since parameterization approaches remain on a conceptual level [Neu04] and describe a market with a plain list of parameter values, the understanding of each parameter's meaning, function and impact, presupposes fundamental knowledge about markets. The reader inexperienced with auctions will rely on a lot of external information, since the underlying structure of auctions is not contained within the listing. The parameters are "one-dimensional" for example the revelation of information – no languages are applied. Due to this fact, the number of auctions configurable with parameterization on a detailed level is limited, although quite impressive.

Flow chart descriptions provide a rudimentary functional description of what is going on "inside" the auction mechanism.[Rod⁺97] However, they also remain on the design level and only regard one aspect of market description. They roughly sketch the outline of mechanisms, but are neither appropriate for thorough theoretical analysis nor a sufficient basis for implementation.

3 Example Auction

In this section we present an example auction for subsequent illustration. We will now outline it in common terminology and amplitude to give a first impression of the auction. As it is symptomatic for verbal description not all aspects are considered in detail. A comprehensive description according to our requirements will step by step be introduced throughout the paper.

The example auction comprises one item to be sold and one negotiable attribute, namely the price. It goes a step beyond traditional auction types to gain additional

⁷ <http://www.fipa.org/>

⁸ <http://www.fipa.org/specs/fipa00031/XC00031F.pdf>, Document number: XC00031F, Document source: FIPA TC C, Document status: Experimental, Date of this status:2001/08/10

⁹ <http://www.fipa.org/specs/fipa00032/XC00032E.pdf>, Document number: XC00032E, Document source: FIPA TC C, Document status: Experimental, Date of this status: 2001/01/29

characteristics for illustrating the capabilities of our approach. Therefore, the provided example combines notions of the English and the Vickrey auction.

The “English auction part” starts at a fixed point in time. The seller offers a mobile phone of the type “MP1” in the color “ice blue”. She has specified a reserve price of 10 Euros. Potential buyers can bid throughout this stage and are continually informed about the offered price in the highest bid. This part ends, when there is no new bid arriving for 5 seconds.

The “Vickrey auction part” starts immediately after the first part ends. In this stage only the three bidders with the highest bids of the first stage are considered. For exactly one minute they can submit sealed bids.

Directly afterwards, winner and price are determined and all participants of the auction are informed about the price. Therefore all bids of the first stage and only the last one submitted by each of the remaining three participants in the second stage are considered. The participant of the bid with the overall highest price of both stages wins. She pays the second highest price of all considered bids.

4 Auction Reference Model

4.1 Markets and Auctions

We draw on the notion that “a market is the impartial structured condensation of participants’ intentions into exchange agreements.” [Rol⁺04] What participants therefore do is utter their intentions in the market which are the basis for agreements. “An intention represents the smallest closed entity of purpose within a market.” [Rol⁺04] It is characterized by its associated participant, products and attributes. An agreement comprises two intentions that concur with each other and indicates that the involved participants have committed themselves to exchanging the products of the respective intentions. It can be seen as the economic equivalent to a fully specified legal contract complying with all formalities.

A market model then intends to capture the structure that the condensation from intentions to agreements follows. The structure is governed by rules – the institution [Smit03] – that are contained in a market model. Auctions provide a particularly comprehensive set of explicit rules. We will call a market model that contains such a set of auction rules an auction model. Accordingly, the challenge of providing the desired comprehensive auction specification is to represent the respective auction model in a coherent and formalizable way. Therefore, we will follow the notion of a Minimal Market Model [Rol⁺04] and refine its structure to meet an auction model’s requirements.

The Minimal Market Model (MMM) describes a formalizable minimal set of conditions as a market core. [Rol⁺04] It claims to contain every condition common to all markets in the most compact way and divides markets into two main parts:

- The *statics* of the MMM provide the means for fully describing all fixed market instance data at any single point in time for any market.
- The *dynamics* describe the patterns that occur in the evolution of the static market instance data over time. No significant common core was identified for the dynamic structures in the MMM, as they are what is individual for each market model and disjunctive for the whole of all.

For all MMM details constitutive for auction description the following subsections will subsequently provide thorough explanations in the place where they become relevant for the auction model.

In the context of markets and auctions, there are two kinds of market data. Firstly, there is data gathered in the process of an actually running market, a market instance. All of this information constitutes the data basis of a market. As this data is represented in instances of concepts from the market model level, we call it *market instance data*. Secondly, there is data describing the market model. We will call this *market model data*.

This applies analogously to the term of market information.

For building the Auction Reference Model, we will start with an auction definition by Wellman and Wurman: “Technically, the auction mechanism is that part of the protocol governing the behavior of the auctioneer, essentially defining the rules of interaction, and determination of the result. The entire auction protocol comprises the mechanism, plus the behaviors of participating bidders.” [WeWu98] This auctioneer for us is a theoretical notion that represents the mandatory control over the rule execution and compliance. What merely counts is that all participants obey the rules that are prescribed. This can be ensured by software implementation of rules, the traditional auctioneer in person or even mutual control of participants. The auction mechanism is what we want to describe in order to seize the character of an auction.

4.2 Architecture of the Auction Reference Model

To explain and structure auction mechanisms we put them in the context of participating bidders and a data basis with auction instance data. Conceptually, the data basis is simply a collection of intentions and agreements in whatever way they may be represented. A container for collecting and storing this data may be as traditional as a proficient human auctioneer’s mind. The requirements for the data basis are allowing intention access via the validation component, accepting agreements exclusively from the auction flow and storing all this market instance data permanently together with timing information. If these basic, common data

storage requirements are provided, the individual character of an auction mechanism is fully described by the auction flow.

These considerations entail the architecture of the Auction Reference Model as illustrated in the following Figure 1.

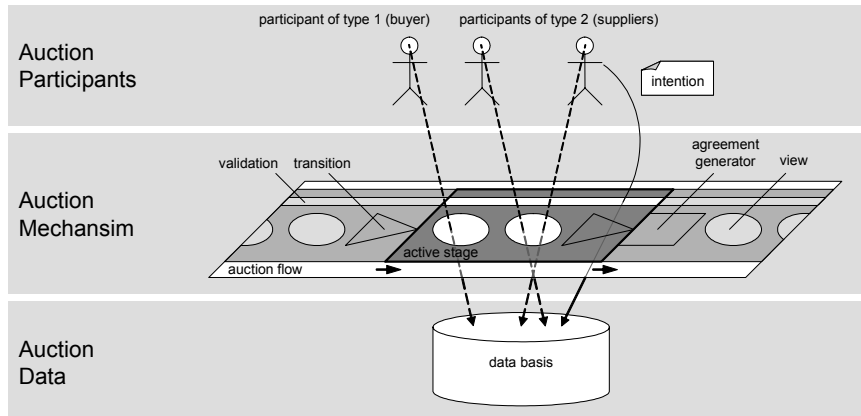


Figure 1: Architecture of the Auction Reference Model

On the bottom of Figure 1 is the data basis that contains the market instance data, namely the intentions and agreements of the respective auction. The parallelogram in the middle of the picture represents an excerpt of an auction mechanism that always consists of stages. There is only one stage active at a time and the sequence of stages is controlled by the auction flow. Each stage contains components of the four types view, validation, agreement generator and transition that will be explained in detail in chapter 4.3. At its top, Figure 1 shows participants. Naturally, they do not belong to the auction mechanism itself. They are primarily displayed for illustrating how views take effect. The middle and right participants are of the same type, therefore share the same view and consequently can see the same market instance information in the active stage. The participant to the left is of another type and therefore can only access another view which implies different information visibility for her. The right participant submits an intention that is checked positive and forwarded to the data basis by the validation component.

The following subsection takes a closer look at each component and motivates its function and position within the architecture.

4.3 Structure and Components of the Auction Reference Model

As the Auction Reference Model elaborates on a particular subset of markets, namely those with auction mechanisms, it provides far more (dynamic) conditions

than the MMM. Therefore the idea of a core that is common to all markets is succeeded by a more detailed model common to all auctions in its nature. Minimality, however, is preserved in the sense of striving for an essential description as compact as possible and consequently eliminating information superfluous to the theoretical concept. In this section we take a closer look at the compact ARM outlined by the architecture in Figure 1 and provide further motivation and a more detailed explanation. At first, we address the general issue of a suitable data structure that (potentially) remains unchanged for every auction mechanism. Then we examine the structure of stages that combine the auction-specific components view, validation, agreement generator as well as transition. A closer look at the auction-specific components follows. Combinations of their various embodiments over the different stages form the unique character of each mechanism.

4.3.1 Data

The common data basis requirements of intention access from participants via the validation, exclusively accepting agreements from an auctioneer and storage of relevant market instance data with timing information have already been introduced. What remains to do for the data component is examining the structure of intentions and agreements that constitute the data to be stored.

Since the MMM is applicable to every type of market, which includes every auction, and fully specifies all respective static aspects, we can fully adopt its statics for the auction setting. According to [Rol⁺04] a market is characterized by a set of intention and agreement instances that change over time. What the statics do is capture the nature of both concepts intention and agreement with the following basic data structure.

According to Figure 2 an intention is defined by an associated participant and two groups of products, namely incoming and outgoing ones. Incoming products are what the participant of the respective intention wants to receive while outgoing products are what she is willing to give away in exchange. An intention can be set to be binding. This means that it – respectively its participant – is committed to the exact fulfillment of the expressed purpose. If it is not binding, it merely reveals market information but does not in any way represent commitment nor the will for commitment. The `forMatching` flag of attributes declares that a respective attribute is considered when matching the underlying intention with others. An attribute that is not `forMatching` gives information about the respective product, yet it is not regarded for the matching. For the context of this paper, it is sufficient to now simply state that an agreement connects and comprises two intentions. For further explanation on the data structure see [Rol⁺04].

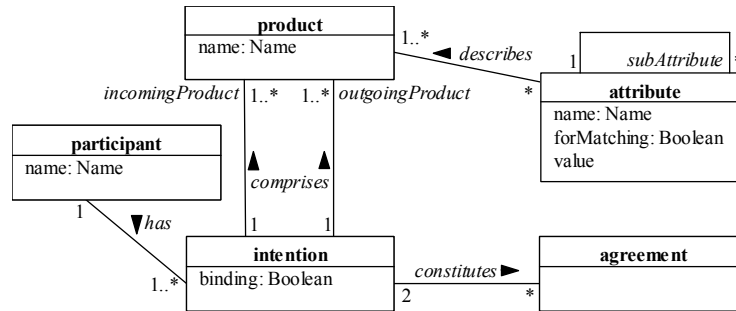


Figure 2: UML class diagram for data structure of intention and agreement

So the data basis must be capable of storing information in the sense of this data structure. Participants as well as the auctioneer take part in the market referring to the same structure for intentions and agreements. For specific purposes, the structure can be refined, but it is conceptually sufficient for every market and therefore also every auction.

Intentions are a theoretical construct and can practically be communicated in a multitude of different ways, for example by raising one's hand in a real-life English auction. In a real-life setting the intention per se is, of course, not visible, but still precisely defined in the context of the market model and the history of preceding intentions.

For the determination of auction results as well as for views it must be recognizable when intentions and agreements appear in the market. This can be provided for by keeping timestamps for their market entry. Time aspects are also crucial for the start and end of auction stages, especially when referring to them with the views. Therefore, the information of when a stage was active must also be kept in the data basis.

4.3.2 Stage

An auction always consists of several stages. Every stage contains embodiments of the four components view, validation, agreement generator and transition, which are defined as follows. There is exactly one stage active at any time of the auction and as it is active each stage merely activates the components it contains. The order in which the stages are executed is determined by the auction flow.

4.3.3 View

The MMM does not limit the visibility of market instance data by any means, as there are no respective restrictions common to all markets. However, we already mentioned that it is crucial in particular markets, especially auctions, to express

that not every market participant will be granted the rights to see all market instance data at any time. To account for the restricted subjective perspective of participants, we provide the concept of views. In each stage there can be any number of views greater or equal to zero. A simple example for systematic implementation of data subjectivity is only providing the number of all participants who have submitted bids to an auction up to a certain point in time instead of revealing their names to a buyer.

Typically, a view is not unique for only one participant but rather a group. Therefore views are usually assigned to types of participants. With the exception of external events, views solely operate on the market instance data including the timing information for intentions, agreements and stages. In addition to the visibility for participants, the three remaining component types validation, agreement generator and transition rely on views.

One crucial difference between the known parameterization approaches and the ARM becomes apparent at this point. The extensive parameter set of the Montreal Taxonomy [StWe03] contains, for example, parameter 9.f) information content. It can have one of two values:

- *unrestricted* means “all elements of the winning offer, such as price quotes or constraints, are revealed” [StWe03]
- *selected* means “only selected offer or status information (e.g. only quality but no price information) is provided by the electronic negotiation medium” [StWe03]

While this parameter does add to the description of a trade negotiation, in particular an auction, it is not comprehensive in the sense of our requirements. The potential revelation of related information such as the number of bids submitted is neglected but could easily be solved by adding more parameters. What is, however, beyond the scope of parameterization is enabling the precise specification of what *selected* information exactly is provided to which participant.

4.3.4 Validation

One validation component exists in every stage. Throughout the whole duration of the stage it belongs to, it checks if incoming participant intentions are valid for the respective auction (stage) or not. In case they are, the intentions are stored in the data basis; otherwise they are discarded. The validation typically relies on views. It may, for example, retrieve the current highest price from a view and compare it to the price indicated in a new intention in order to store the intention only if its price is higher than the currently leading one.

4.3.5 Agreement Generator

Each stage contains zero or more agreement generators. An agreement generator is the only entity that can produce an agreement which is done at the very beginning of the stage the agreement generator belongs to. In order to construct every agreement in compliance with the introduced data structure, a template according to the agreement structure is taken and the intention parts filled out basing on views. One such view might deliver the first seller intention of the example auction, which the agreement generator would then add to the agreement structure.

4.3.6 Transition

Exactly one transition is part of every stage. Its purpose is to end the stage it belongs to according to a certain condition. Such a condition can either base on views, a timer or a combination of both. A very simple example is a transition that incorporates a timer and waits for precisely one minute to then end its stage.

4.3.7 Auction Flow

There is exactly one auction flow for every auction. With elements from business process theory, respectively workflow theory, it controls the order of an auction's stages. Therefore we need the concepts of coordinating stages in sequence and in loops with break conditions. Once a stage is finished by its transition the auction flow determines the next stage to be activated. Every stage is started immediately after the proceeding one ended. In case of our example auction all stages are executed in sequence.

4.4 Description of the Example Auction

In order to illustrate the use of the ARM for description, we apply its ideas to the example auction introduced in section 3.

The example auction is built on a data basis with the presented common requirements, namely validated intention access for participants, only accepting agreements from an auctioneer and storage of relevant market instance data including timing information.

We do not consider the specification of the product that is sold in the example auction as part of the mechanism description. The product is rather contained in the first intention the auctioneer considers, namely the seller intention. The auction flow coordinates four stages in sequence.

The **first stage** does not provide any views to participants. It is started and ends at fixed points in time. The validation of stage one accepts intentions of the following kind: they must be declared binding, specify one incomingProduct *money* with

one attribute *currency* that is forMatching and another attribute *amount* that is an interval ranging from a defined positive number – including null – to infinity. Furthermore one outgoingProduct must be specified.

In our example the first intention of this kind defines the *currency* of the *money* as “Euros” and the *amount* as “10...*”. The outgoingProduct is a mobile phone with the forMatching attributes *type* set to “MP1” and *color* to “ice blue”. Such an intention is often specified on the webpage of an auction platform provider via input fields. For our example, however, we will regard it as a message complying with the intention structure.

The validation of the **second stage** only accepts buying intentions that match the initial seller intention. All these intentions must be binding, have the incomingProduct equaling the outgoingProduct of the seller intention, specify as their outgoingProduct *money* with the same *currency* of the incomingProduct of the seller as well as offer an *amount* higher than the lower limit of the seller interval and the highest submitted bid *amount* so far. For determining the latter the validation makes use of a view that provides this leading price. An identical view is provided to the potential seller, all bidders and anyone interested in the auction. This view selects the buying intention with the highest *amount* and returns this *amount* – the current leading price. The transition also keeps track of this view and ends the first stage when there has been no change in the delivered value for 5 seconds.

The **third stage** also provides one view. This view returns the participants with the three highest bids in the second stage. It is accessible for everyone who has submitted intentions in the first and second stage to let them know whose intentions are regarded in stage three and whose not. The validation accepts buying intentions only from these participants – three or less. Neither seller nor any buyers see what intentions are submitted during this stage as they have no view assigned concerning current market instance data. The respective transition ends stage three exactly one minute after it started.

The **fourth stage** is the stage where an agreement is generated. The agreement generator of stage four selects the last intention submitted by each of the three or less allowed participants in stage three. From the latter intentions plus all the ones from the second stage, the agreement generator selects the participant of the one with the highest *amount* as the winner. This participant can access the view that delivers the winner result and is there informed that she wins. The second-highest amount of the latter intentions is selected and provided by a view accessible for everyone as the price to be paid. Based on views, the agreement generator generates an agreement that connects two intentions: one is the original seller intention with its *amount* concretized to the price to be paid in the auction; the second is the winner intention with its *amount* adapted also to this price. A transition ends stage four at the time when the information provided by the views visible to participants is not meant to be offered anymore.

In the following section we introduce methods on how to declaratively specify all this auction information.

5 Description Methods and Implementation Example

In this section, we outline our approach to the individual description of the components and explain the respective choice of technologies for the prototype developed. Furthermore, implementation details as well as a discussion of the design and architecture applied are presented to illustrate the first proof-of-concept implementation of the auction runtime engine we developed.

Our vision is to provide a market engineer with the freedom to specify all aspects of a market without being hampered by technological constraints and without being dependent on costly and time-consuming implementation efforts. The paradigm of Model Driven Architecture (MDA) seems to be applicable to achieve this goal. Rather than simply coding a solution, MDA suggest to generate code from abstract models. Since the components introduced in the previous section allow specifying auctions in their entirety, we are able to avoid manual coding. Instead, the user can specify various aspects of the auction at a higher level of abstraction.

5.1 System Overview

The implementation architecture closely follows the conceptual architecture of the Auction Reference Model depicted in Figure 1. The concrete realization is shown in Figure 3.

The runtime hosts the auction and handles low level threading, communication, and timing issues. Inside, the central component handles the action flow, the stage transitions, and the agreement generation. The respective logic is captured in a Java class, which is generated from a BPEL script. The idea behind this setup is that the designer only needs to use a restricted set of directives. Therefore, BPEL is expressive enough for representing the respective information. A key advantage of BPEL is the availability of graphical design tools that allows to visually model the process. The script accesses the data and auction stage information via the views. The participants are shown on the right. We developed a simple GUI (for a screenshot refer to the website), which is mostly used for testing. The communication is carried out via RMI and flows both ways. The market offers the following services:

```
public interface Server {
    public void placeBid(Intention i);
    public Object query(String attribute);
}
```

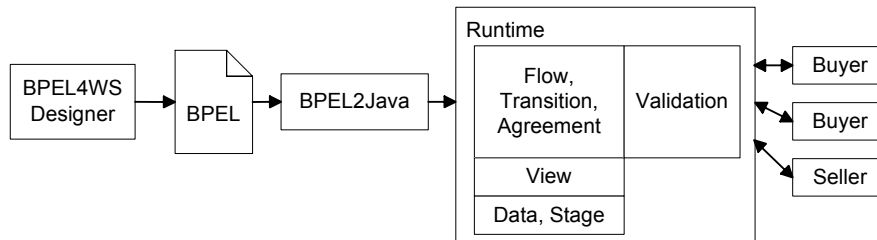


Figure 3: System Architecture

The first method allows transmitting an intention. The method `query` is used to obtain information about the state of the auction. Typical examples are the current highest bid, the auction's stage, or the time remaining until the next stage is reached.

At the moment, the participants can only receive simple text messages sent from the market:

```
public interface Participant {
    public void receive(String msg);
}
```

5.2 Structure and Components

This section picks up the structure and components introduced in section 4 and explains their technical realization.

5.2.1 Data

The core data structures are directly derived from the UML Model shown in Figure 2. The auction history stored in the data repository is a list of intentions. Note that the arbitrary attribute structure is not supported at the moment. We currently use a single product attribute. The data repository is a simple Java Class, which persists its information in a relational database.

A central design principle of the data basis is that it behaves like a recording of the auction. Therefore, no value gets changed or deleted in retrospect. Intention structures arriving from participants are inserted into the repository. The possibility of intention deletion is not required, since the withdrawal of offers is not accounted for in our example auction. Update or modification of intentions – as it can happen in both the second and third stage of the example auction – is possible by sending a new valid intention. All necessary calculations are performed via the view mechanism outlined in section 5.2.3.

5.2.2 Stage

In addition to the list of intentions, we keep the stage information by remembering the points in time, when the auction finished one stage and moved to the next. In general, every timestamp is created by the data repository upon insertion of the respective entry. This mechanism allows the system to determine which stage an intention or agreement belongs to.

5.2.3 Views

The previous sections mentioned that no intermediate information is needed for auction execution in the model. Views are the perfect vehicle for achieving this goal and are created to satisfy the information need of the auction. Consider the following example from our auction scenario. Recall from subsection 4.4 that participants considered in the third stage are those having submitted one of the three highest bids in the second stage. This information is needed to inform every participant from the second stage about these three considered participants and it is reused for incorporation into the overall winner and price determination in stage 4. This computation is performed as follows:

```
history.getIntentionsOfStage(2).  
    sortByProdInAmountDesc().getFirstI(3)
```

Starting from the entire history, i.e. all intentions received, we get the intentions of the first stage. `getIntentionsOfStage` is a view which transforms a list of intentions into another list of intentions by filtering intentions from other stages. The resulting set is sorted in descending order by price. Finally, the top three intentions are selected. The views themselves are simple Java methods like the following:

```
public IntentionList getIntentionsOfStage(int stage) {  
    IntentionList res = new IntentionList();  
    for (int i=0; i<this.size(); i++)  
        if (this.getIntention(i).stage == stage)  
            res.add(this.getIntention(i));  
    return res;  
}
```

5.2.4 Validation

Validation is performed for both auction services. When the query service is invoked, validation needs to make sure that no confidential information is issued to the caller. The following example shows the validation rule stating that the highest price is not accessible in stage 3:

```
if (history.getStage() == 3)
    if (attribute.equals("HighestBid"))
        throw new AccessControlException(
            "current price is hidden");
```

Validation rules for submitting intentions make sure that incoming and outgoing products match, that the bids are high enough, and so on.

5.2.5 Agreement Generator

The job of the auction system is to match intentions into agreements according to the rules of the market. Consequently, every time an agreement is reached, it needs to be recorded. In our case this happens by calling the `create-Agreement` functionality from the respective position in the flow.

5.2.6 Transition

Similar to the agreement generator, advancing the stages is performed as part of the flow logic by grouping the flow into so-called BPEL scopes, which correspond to the stages.

5.2.7 Flow

The history in the database and the views are already capable of representing several aspects of an auction. However, the logical flow of auctions is the main part that orchestrates the other components into one coherent auction. The following list illustrates the tasks that need to be done for this:

- **Advance stages** As was mentioned before, auction stages are an important mechanism for structuring auctions and need to be defined in the flow.
- **Define timeout** Auctions advance to the next stage if a given timeout expires. Timeouts can also be adjusted upon events such as the arrival of a new bid.
- **Conditional flows** The auction flow can contain conditional statements. These conditions regulate the transition from one stage to another. There is a choice of modeling behavior via policies or by explicitly checking the condition within the flow.
- **Query Information** Over the lifetime of an auction, participants can query information. This activity is for in our case provided by the views. However, the availability of the views in the different stages is controlled by the validation.
- **Receive Intentions** The auction can receive intentions at various stages.
- **Propagate information** Pushes messages to participants.
- **Generate agreements** The ultimate goal of any market.

We tie all these tasks together with BPEL. Since the possibility of designing flows graphically seemed very appealing, we decided to use the Collaxa BPEL designer. After the flow is stored in the BPEL XML format, our BPEL2Java tool parses the flow and generates Java code from it. Obviously not all constructs available in the tool are supported and we restrict ourselves to the tasks shown above. The following text illustrates, how these tasks can be represented in BPEL and how the resulting XML is converted to Java.

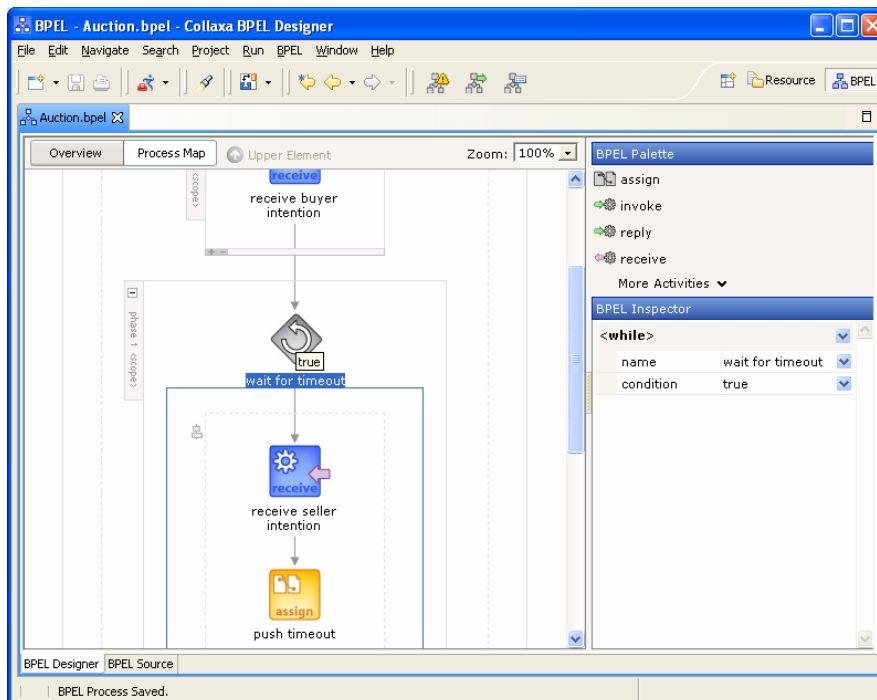


Figure 4: Sample auction in the Collaxa BPEL designer

Auction stages can be captured using BPEL scopes. A scope is very similar to a block in object-oriented languages. Consequently, a scope can have its own exception and event handling. Timeouts and deadlines are natively supported by BPEL and are treated as events. In our case, the timeout event handler does not do anything but simply advance to the next stage in the sequence of stage scopes. The wait deadline is captured in the internal variable `timeout`, which can be changed by an assignment. Figure 4 shows an example. During auction stage two, the expiration time is postponed every time another intention is received. In stage one a single receive activity is used to obtain the initial seller intention. Stage two uses the while construct in combination with a timeout.

On the XML level, part of the flow shown in figure BPEL looks as follows:

```

<sequence>
  <scope name="scope 1">
    <receive name="receive seller intention"/>
  </scope>
  <scope name="scope 1">
    <eventHandlers>
      <onAlarm until="timeout"/>
    </eventHandlers>
    <while condition="true" name="wait for timeout">
      ...

```

The corresponding Java code is structured very similarly. The method `nextStage` increments an internal counter and registers the time of the transition. The call to `receiveBid` can optionally contain the timeout point.

```

receiveBid();
nextStage();
try {
  while (true) {
    receiveBid(timeout);
    timeout = now() + timeout;
  }
} catch (Timeout t) {}
nextStage();

```

6 Conclusion and Outlook

After presenting the ARM architecture, we explained its components on a conceptual level. Means to declaratively represent each component's embodiment were discussed and chosen in a way that easily makes possible the implementation of a generic auction runtime engine. For demonstrating the feasibility we described and provide a prototypical example¹⁰.

The ARM structures and conceptually separates the aspects of an auction that are often intertwined in auction theory. It is consequently reduced to the minimal information required for comprehensive auction description and therefore very compact.

To demonstrate the ARMs versatility, our future work will provide specifications for other types of auctions like, for example, double-sided continuous and combinatorial ones. Several technical aspects of the system will also be improved. A

¹⁰ <http://www.iw.uni-karlsruhe.de/auctionrm/>

Web Service interface, supporting information push, hot deployment of auctions, and better tool support would be desirable. We are also investigating using Semantic Web technologies like RDF, Ontologies [Sta⁺04], and rule languages [Bol⁺01] in order to encode domain knowledge about auctions as such and also the goods being traded into the system in a declarative way. An overall migration to the Service Oriented Architecture (SOA) [Erl04] and XML-storage technology is considered in order to support these plans and take declarative representation to the next level.

The ultimate goal would be to leverage the declarative auction descriptions on the client side. It appears possible to use this information in order to create a more generic auction client that is capable of participating not only in the auction type it was designed for, but can also deal with variations thereof.

References

- [AnJe03] Anthony, P.; Jennings, N. R.: Developing a Bidding Agent for Multiple Heterogeneous Auctions. In: *ACM Transactions on Internet Technology (TOIT)* 3(3). 2003. S. 185-217.
- [Bapn⁺01] Bapna, R.; Goes, P.; Gupta, A.: Insights and analyses of online auctions. In: *Communications of the ACM* 44(11). 2001. S. 42-50.
- [Bol⁺01] Boley, H.; Tabet, S.; Wagner, G.: Design Rationale of RuleML: A Markup Language for Semantic Web Rules. *International Semantic Web Working Symposium (SWWS)*, Stanford, California, USA.
- [Erl04] Erl, T.: *Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services*. Prentice Hall PTR: 2004.
- [Maes⁺99] Maes, P.; Guttman, R. H.; Moukas, A. G.: Agents that buy and sell. In: *Communications of the ACM* 42(3). 1999. S. 81-ff.
- [McMc87] McAfee, R. P.; McMillan, J.: Auctions and Bidding. In: *Journal of Economic Literature* 25. 1987. S. 699-738.
- [Neu04] Neumann, D.: *Market Engineering - A Structured Design Process for Electronic Markets*. Dissertation, University of Karlsruhe (TH), 2004.
- [Prem03] Premkumar, G. P.: Perspectives of the E-Marketplace by Multiple Stakeholders. In: *Communications of the ACM* 46(12). 2003. S. 279-288.
- [Rod⁺97] Rodríguez, J. A.; Noriega, P.; Sierra, C.; Padget, J.: *FM96.5 A Java-based Electronic Auction House. Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM'97)*, London.
- [Rol⁺04] Rolli, D.; Neumann, D.; Weinhardt, C.: A Minimal Market Model in Ephemeral Markets. *Proceedings of the TheFormEMC 2004*, Toledo, Spain.

- [Roth⁺02] Roth, A. E.; Ockenfels, A.: Last-Minute Bidding and the Rules for Ending Second-Price Auctions: Evidence from eBay and Amazon Auctions on the Internet. In: *American Economic Review* 92(4). 2002. S. 1093-1103.
- [Schm⁺98] Schmid, B. F.; Lindemann, M. A.: Elements of a Reference Model for Electronic Markets. Thirty-First Annual Hawaii International Conference on System Sciences (HICSS-31), Kohala Coast, Hawaii.
- [Smit03] Smith, V. L.: Markets, Institutions, and Experiments. In: *Encyclopedia of Cognitive Science*. L. Nadel. Macmillan, 2003,
- [Sta⁺04] Staab, S.; Studer, R.: *Handbook on Ontologies*. Springer: Heidelberg, 2004.
- [StWe03] Ströbel, M.; Weinhardt, C.: The Montreal Taxonomy for Electronic Negotiations. In: *Journal of Group Decision and Negotiation* 12(2). 2003. S. 143-164.
- [VrVo01] Vries, S. d.; Vohra, R.: Auctions and the German UMTS-Auction. In: *Mitteilungen der DMV* 1/2001. 2001. S. 31-38.
- [WeWu98] Wellman, M. P.; Wurman, P. R.: Real Time Issues for Internet Auctions. First IEEE Workshop on Dependable and Real-Time E-Commerce Systems (DARE-98), Denver, Colorado, USA.
- [Wur⁺01] Wurman, P. R.; Wellman, M. P.; Walsh, W. E.: A Parametrization of the Auction Design Space. In: *Games and Economic Behavior* 35. 2001. S. 304-338.