Law-Governed Peer-to-Peer Auctions

Marcus Fontoura
IBM Almaden Research Center
650 Harry Road, San Jose, CA, 95120
fontoura@almaden.ibm.com

Mihail Ionescu, Naftaly Minsky
Computer Science Department
Rutgers University, Piscataway, NJ 08854
ionescu,minsky@cs.rutgers.edu

ABSTRACT

This paper proposes a flexible architecture for the creation of Internet auctions. It allows the custom definition of the auction parameters, and provides a decentralized control of the auction process. Auction policies are defined as *laws* in the Law Governed Interaction (LGI) paradigm. Each of these laws specifies not only the auction algorithm itself (e.g. open-cry, dutch, etc.) but also how to handle the other parameters usually involved in the online auctions, such as certification, auditioning, and treatment of complaints. LGI is used to enforce the rules established in the auction policy within the agents involved in the process. After the agents find out about the actions, they interact in a peer-to-peer communication protocol, reducing the role of the centralized auction room to an advertising registry, and taking profit of the distributed nature of the Internet to conduct the auction. The paper presents an example of an auction law, illustrating the use of the proposed architecture.

Categories and Subject Descriptors

D.2.11 [**Software Engineering**]: Software Architectures; D.2.2 [**Software Engineering**]: Design Tools and Techniques; K.4.4 [**Electronic Commerce**]: Security; C.2.4 [**Computer Communication Networks**]: Distributed Systems

General Terms

Design, Performance, Security

Keywords

Online Auctions, Distributed Systems, Distributed Enforcement, Law Governed Interaction

1. INTRODUCTION

The Internet has made possible the creation of virtual auctions rooms, in which buyers and sellers scattered across the globe interact to close deals. Internet auctions allow faster and less expensive transactions with no geographical barrier [5]. Although Internet auctions have been successfully used [4], current auction applications still do not fully exploit the distributed nature of the Internet. They are based on centralized systems that necessarily make decisions about the auction process, taking away important choices from the auction participants. These choices include:

WWW2002, May 7–11, 2002, Honolulu, Hawaii, USA.
ACM 1-58113-449-5/02/0005.

- The auction algorithm itself: several types of such algorithms can be used (like open-cry, dutch, sealed, etc.), and each of them may have many variations [2, 3];
- Certification: how to compute reputation and trust information about the auction participants;
- Auditing: what needs to be audited, and by whom;
- The treatment of complaints: how to handle complaints about inappropriate behavior of auction participants or about unsuccessful transactions.

Although some centralized auction systems provide flexibility in the choice of the auction algorithm (e.g. through parameters, as in the AuctionBot system [9]), the ability to make the other auction parameters flexible, such as certification, auditing, and treatment of complaints, is much more difficult, if at all possible, to be accomplished via a centralized system.

Let us consider two auction scenarios: in the first the seller wants to sell modern art items while in the second he or she wants to sell audio CDs. In the modern art case, it is very important for the seller to present some certificate that can be used in order to determine the authenticity of the items being sold. If this auction is being conducted by a centralized auction server, the buyer has to trust either the server or the certifying authorities appointed by the server. This might not be desirable for the buyer, since different people trust different entities, which is especially true for transactions involving valuable items. Therefore, a buyer should be able to decide to participate in such an auction only if some particular certifying authorities (CAs) are involved. On the contrary, in the case of audio CDs, a sophisticated certification mechanism is not necessary since buyers can easily check the accuracy of the product description and complain afterwards, if not satisfied.

This simple example illustrates that the level of certification required in an auctioning system is product dependent, and cannot be fully implemented by a centralized server. Instead, auction participants should be able to indicate which certifying authorities they trust, customizing the auction process. The same argument is true for other auction parameters. For example, the management of user reputation in eBay (http://www.ebay.com) allows people to build a good reputation by doing false transactions. Buyers would be more willing to participate in some auctions if someone they trust knew the real identity of the seller. If such an

entity exists, complaints can more easily be solved and the possibility of cheating decreases.

Having a centralized auction server to perform all the tasks of an Internet auction (advertising, enforcing the auction policy, controlling user reputation, and so on) is not suitable for online auctioning. Since different people trust different entities for performing each of the individual tasks in the auction process, we need a different model, in which all the entities involved can be dynamically specified in an *auction policy*.

In this paper we propose a mechanism that dispenses with a centralized auction server, replacing it with (1) an *auction registry*, which maintains information about active auctions, each with the policy which is to govern it; and (2) a highly distributed mechanism, called *law-governed interaction* (LGI) [7], that conducts each auction according to the policy specified for it. More specifically, under the proposed mechanism, an auction is registered by specifying the item to be auctioned along with the policy that is to govern this particular auction. The policy is specified as a *law* under LGI. It can be chosen from some library of such laws, or be written specifically for this particular auction, using a language for writing such laws under LGI. Once an auction is registered, the seller and the buyers can participate in it via peer-to-peer (P2P) communication, subject to the strictly enforced law of this particular auction, and not involving the central auction registry.

The rest of the paper is organized as follows: Section 2 briefly describes the LGI mechanism. Section 3 describes the architecture for LGI-based auctions, and demonstrates it by means of a detailed example; Section 4 presents an overview of related work. Section 5 concludes the paper and presents our future research directions.

2. LAW-GOVERNED INTERACTION

This section provides an overview of LGI, introducing the concepts necessary for understanding the LGI-based auctioning system. Section 3 builds upon this section and it describes how LGI can be applied as the underlying technology for supporting online auctions.

LGI is a message-exchange mechanism, first introduced in [6], that allows an *open group* of distributed agents to engage in a mode of interaction *governed* by an explicitly specified policy, called the *law* of the group. The messages thus exchanged under a given law $\mathcal{L}$ are called $\mathcal{L}$-messages, and the group of agents interacting via $\mathcal{L}$-messages is called a *community* $\mathcal{C}$, or, more specifically, an $\mathcal{L}$-community $\mathcal{C}_\mathcal{L}$.

By the phrase "open group" we mean (a) that the membership of this group (or, community) can change dynamically, and can be very large; and (b) that the members of a given community can be heterogeneous. In fact, we make here no assumptions about the structure and behavior of the agents[1] that are members of a given community $\mathcal{C}_\mathcal{L}$, which might be software processes, written in an arbitrary languages, or human beings. All such members are treated as black boxes by LGI, which deals only with the interaction between them via $\mathcal{L}$-messages, making sure it conforms to the law of the community. (Note that members of a community are not prohibited from non-LGI communication across the Internet, or from participation in other LGI-communities.)

For each agent `x` in a given community $\mathcal{C}_\mathcal{L}$, LGI maintains, what is called, the *control-state* $\mathcal{CS}_x$ of this agent. These control-states, which can change dynamically, subject to law $\mathcal{L}$, enable the law to make distinctions between agents, and to be sensitive to dynamic changes in their state. The semantics of control-states for a given community is defined by its law, and could represent such things as the role of an agent in this community, and privileges and tokens it carries.

We continue this section with a brief discussion of the concept of law, emphasizing its local nature, and with a description of the decentralized LGI mechanism for law enforcement. We do not discuss here several important aspects of LGI, including its concepts of *obligations* and of *exceptions*, its treatment of certificates, the deployment of $\mathcal{L}$-communities, the expressive power of LGI, and its efficiency. For these issues, and for implementation details, the reader is referred to [7, 1].

2.1 Laws, and Their Enforcement

Generally speaking, the law of a community $\mathcal{C}$ is defined over a certain types of events occuring at members of $\mathcal{C}$, mandating the effect that any such event should have—this mandate is called the *ruling* of the law for a given event. The events subject to laws, called *regulated events*, include (among others): the *sending* and the *arrival* of an $\mathcal{L}$-message; the *coming due of an obligation* previously imposed on a given object; and the *submission of a digital certificate* (more about the latter two kinds of events, later). The operations that can be included in the ruling of the law for a given regulated event are called *primitive operations*. They include, operations on the control-state of the agent where the event occured (called, the "home agent"); operations on messages, such as `forward` and `deliver`; and the imposition of an obligation on the home agent.

Thus, a law $\mathcal{L}$ can regulate the exchange of messages between members of an $\mathcal{L}$-community, based on the control-state of the participants; and it can mandate various side effects of the message-exchange, such as modification of the control states of the sender and/or receiver of a message, and the emission of extra messages, for monitoring purposes, say.

2.1.1 On The Local Nature of Laws:

Although the law $\mathcal{L}$ of a community $\mathcal{C}$ is *global* in that it governs the interaction between all members of $\mathcal{C}$, it is enforceable *locally* at each member of $\mathcal{C}$. This is due to the following properties of LGI laws:

- $\mathcal{L}$ only regulates local events at individual agents,
- the ruling of $\mathcal{L}$ for an event `e` at agent `x` depends only on `e` and the local control-state $\mathcal{CS}_x$ of `x`.
- The ruling of $\mathcal{L}$ at `x` can mandate only local operations to be carried out at `x`, such as an update of $\mathcal{CS}_x$, the forwarding of a message from `x` to some other agent, and the imposition of an obligation on `x`.

The fact that the same law is enforced at all agents of a community gives LGI its necessary global scope, establishing a *common* set of ground rules for all members of $\mathcal{C}$ and providing them with the ability to trust each other, in spite

[1]Given the popular usages of the term "agent," it is important to point out that we do not imply by it either "intelligence" nor mobility, although neither of these is being ruled out by this model.

of the heterogeneity of the community. And the locality of law enforcement enables LGI to scale with community size.

2.1.2 On the structure and formulation of laws:

Abstractly speaking, the law of a community is a function that returns a *ruling* for any possible regulated event that might occur at any one of its members. The ruling returned by the law is a possibly empty sequence of primitive operations, which is to be carried out locally at the location of the event from which the ruling was derived (called the *home* of the event). (By default, an empty ruling implies that the event in question has no consequences—such an event is effectively ignored.)

Concretely, the law is defined by means of a Prolog-like program[2] `L` which, when presented with a goal `e`, representing a regulated-event at a given agent `x`, is evaluated in the context of the control-state of this agent, producing the list of primitive-operations representing the ruling of the law for this event. In addition to the standard types of Prolog goals, the body of a rule may contain two distinguished types of goals that have special roles to play in the interpretation of the law. These are the *sensor-goals*, which allow the law to "sense" the control-state of the home agent, and the *do-goals* that contribute to the ruling of the law. A *sensor-goal* has the form `t@CS`, where `t` is any Prolog term. It attempts to unify `t` with each term in the control-state of the home agent. A *do-goal* has the form `do(p)`, where `p` is one of the above mentioned primitive-operations. It appends the term `p` to the ruling of the law.

2.1.3 Distributed Law-Enforcement:

Broadly speaking, the law $\mathcal{L}$ of community $\mathcal{C}$ is enforced by a set of trusted agents called *controllers*, that mediate the exchange of $\mathcal{L}$-messages between members of $\mathcal{C}$. Every member `x` of $\mathcal{C}$ has a controller $\mathcal{T}_x$ assigned to it ($\mathcal{T}$ here stands for "trusted agent") which maintains the control-state $\mathcal{CS}_x$ of its client `x`. And all these controllers, which are logically placed between the members of $\mathcal{C}$ and the communications medium (as illustrated in Figure 1) carry the *same law* $\mathcal{L}$. Every exchange between a pair of agents `x` and `y` is thus mediated by *their* controllers $\mathcal{T}_x$ and $\mathcal{T}_y$, so that this enforcement is inherently decentralized. However, several agents can share a single controller, if such sharing is desired. (The efficiency of this mechanism, and its scalability, are discussed in [7].)

Controllers are *generic*, and can interpret and enforce any well formed law. A controller operates as an independent process, and it may be placed on any machine, anywhere in the network. We have implemented a *controller-service*, which maintains a set of active controllers. To be effective in a widely distributed enterprise, this set of controllers need to be well dispersed geographically, so that it would be possible to find controllers that are reasonably close to their prospective clients.

2.1.4 On the basis for trust between members of a community:

For a members of an $\mathcal{L}$-community to trust its interlocutors to observe the same law, one needs the following assurances: (a) messages are securely transmitted over the network; (b) the exchange of $\mathcal{L}$-messages is mediated by controllers interpreting the *same law* $\mathcal{L}$; and (c) all these controllers are *correctly implemented*. If these conditions are satisfied, then it follows that if `y` receives an $\mathcal{L}$-message from some `x`, this message must have been sent as an $\mathcal{L}$-message; in other words, that $\mathcal{L}$-messages cannot be forged.

Secure transmission is carried out via traditional cryptographic techniques. To ensure that a message forwarded by a controller $\mathcal{T}_x$ under law $\mathcal{L}$ would be handled by another controller $\mathcal{T}_y$ operating under the *same* law, $\mathcal{T}_x$ appends a one-way hash [8] `H` of law $\mathcal{L}$ to the message it forwards to $\mathcal{T}_y$. $\mathcal{T}_y$ would accept this as a valid $\mathcal{L}$-message under $\mathcal{L}$ if and only if `H` is identical to the hash of its own law.

As to the correctness of controllers, we assume here that every $\mathcal{L}$-community is willing to trust the controllers certified by a given certification authority (CA), which is specified by law $\mathcal{L}$. And, every pair of interacting controllers must first authenticate each other by means of certificates signed by this CA.

2.2 Adopting a Law Under LGI

For an agent to be able to send and receive $\mathcal{L}$-messages, it must: (a) find an LGI controller, and (b) notify this controller that he or she wants to use it, adopting law $\mathcal{L}$. We will discuss these two steps below.

2.2.1 Locating an LGI Controller:

The Moses toolkit includes a controller-naming server, which can be used to maintain a set of active controllers. This server provides the address (host and port) of the available controllers to any agent that wishes to engage in LGI. One may have any number of such servers so that controllers can be distributed in different regions of the Internet. Efficiency-wise, an agent would do best by selecting a controller closest to it (to minimize the overhead of forwarding $\mathcal{L}$-messages through the controller). But functionally-wise, one is free to choose a controller anywhere on the Internet, and several agents may share a single controller without knowing of each other.

2.2.2 Adopting a Law:

Upon selecting a controller agents would send the message

```
certify(law,name,certificate)
```

where `law` is the law that it wants to adopt and `name` is the name that it wants to be known by; the use of `certificate`, which may be empty, is explained below. The argument `law` can take the form of either the text of the law to be adopted or the name of such a law, given to it by a specified *law-repository* service, which is another tool provided by Moses. We will not discuss here the details of this service but rather assume that the text of the entire law is always passed to the controller.

When the controller receives the `certify` message, it checks the supplied law for syntactic validity, and the chosen name for uniqueness among the names of all agents it currently handles. If these two conditions are satisfied[3], it uses a certifying authority to verify the certificate, as exemplified in Section 3.2.

For more details about the implementation of LGI, the basis for trust between members of a community, or how an

[2]Note, however, that Prolog is incidental to this model, and can, in principle, be replaced by a different, possibly weaker, language; a restricted version of Prolog is being used here.

[3]If any one of these conditions is not satisfied, the agent would receive an appropriate diagnostic, and it would be able to try again.

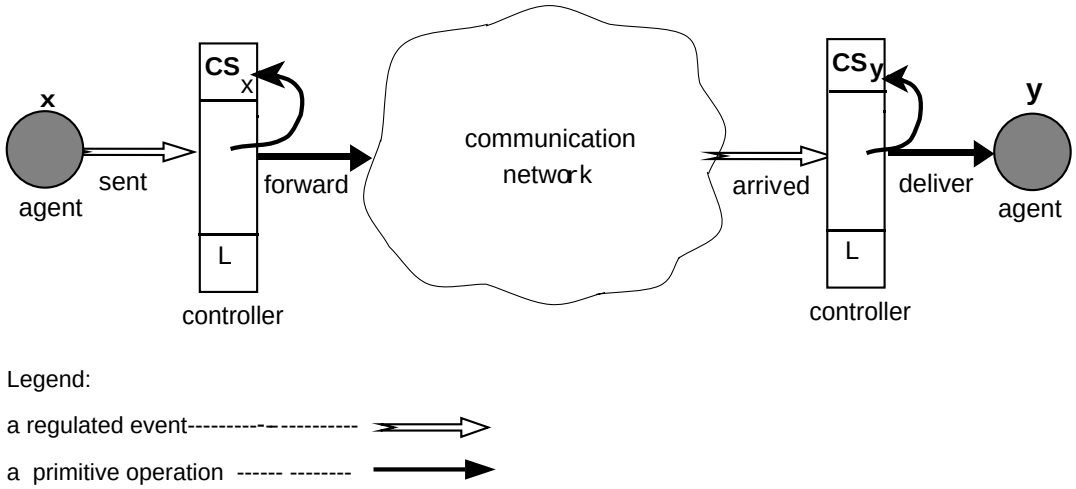

Figure 1: Enforcement of the law.

agent can engage in an $\mathcal{L}$-community, the reader is referred to [7].

3. LAW-BASED AUCTIONS AND SCENARIOS OF USE

In this section we present the LGI-based auctioning system. We start with the general architecture and we then describe an example law for an open-cry auction policy [3]. We also propose solutions for two different issues related to the auctioning in the Internet: auditing and treatment of complaints. We show how these issues, which can be viewed as auction parameters, are specified as part of the auction policy using LGI. The same approach is valid for other auction parameters.

3.1 LGI-based Auctioning System

The three main entities in the LGI-based auctioning architecture are: auction registry, sellers, and buyers. The following paragraphs detail each of them.

- **Auction registry**. The auction registry is a separate agent that holds the selling offers as a tuple {ProductName, Description, SellerAddress, AuctionLaw, Timeout}. Sellers can insert or delete tuples from the registry, while buyers can just query the registry about current auctions [4]. If there is no offer until the timeout for the auction expires, the registry withdraws the auction tuple.
- **Sellers and Buyers**. All the interaction between sellers and buyers is governed by LGI according to the auction policies (laws) specified in the registry tuples. We assume, for simplicity, that the actual exchange of product and money between the buyer that wins the auction and the seller is handled offline, by the two parties involved, like in other auction systems, such as eBay (http://www.ebay.com). However, LGI laws can also incorporate electronic methods of payment like electronic money or PayPall (http://www.paypal.com).

[4]In the case of reverse auctions, such as Priceline (http://www.priceline.com), the opposite situation is possible: the buyers post the auctions and the sellers query the registry for new auctions. For the sake of simplicity, in the rest of the paper our discussion does not consider reverse auctions.

The following messages summarize the interaction between the different entities, as illustrated in Figure 2:

1 Sellers send messages to the auction registry to insert or delete auction tuples. Before a tuple is inserted in the auction registry, or after it is deleted, it is not possible for buyers to bid on the specified product.

2 Buyers make requests for offers that meet some conditions and the registry returns the list of such tuples (if any) back to the buyer. When a buyer discovers about an interesting auction, it can adopt the auction law and join the community that is conducting the auction.

3 Buyers and sellers exchange messages according to the law specified in the auction tuple. They interact directly, in a peer-to-peer communication model, meaning that there is no centralized auction room. The enforcement of the auction law is distributed, using LGI.

It is up to the agents involved to agree to participate in the auction or not, after examining the auction law. Please note that the auction registry does not necessarily belong to the community since interaction with the registry does not need to be governed by LGI. In fact, several implementations of the auction registry can be used, as long as they provide methods for registering and consulting auctions.

Other kinds of agents may also participate in the auctioning process, such as auditors and complaints agents. Figure 3 shows a snapshot of the system with two buyers and two sellers: buyer 1 interacts with seller 1 under law K and with seller 2 under law L. Buyer 2 only interacts with seller 2 under law L. Moreover, in the case of law K buyers and sellers interact with a given auditor, while in law L they interact with an another auditor and with a complaints agent. All interaction among buyers, sellers, auditors, and complaints agents, is peer-to-peer and enforced using LGI.

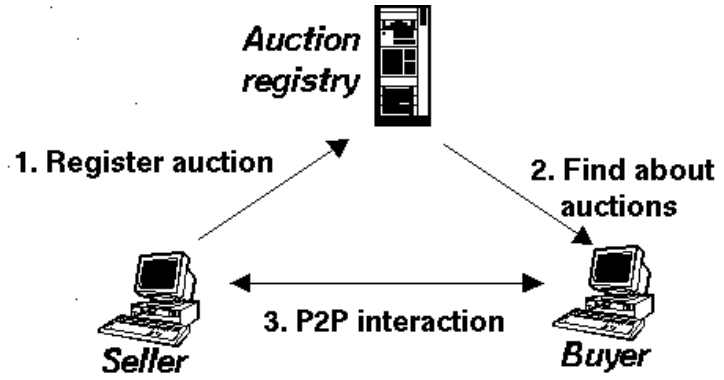

Figure 2: Interaction among sellers, buyers, and the auction registry

Section *3.2.1* details auditing, while Section *3.2.2* describes how complaints can be handled in the proposed architecture.

3.2 Auction Law Example

In this section we present an example of auction law (called open-cry auction [3]) written using LGI. The main idea is that buyers compete with each other to purchase the specified product. At each moment, a buyer can either make a bid or ask for the highest bid so far. If the bid value is greater than the current maximum value, this value is stored at the seller as the maximum current bid. If not, the bid is rejected and the current maximum value is sent back to the buyer. When a higher bid is accepted, the buyer that had the previous highest bid is notified that it was out-bided. At the end of the auction period the seller notifies the winning buyer that he or she actually won the auction. In this way, all of the involved buyers will be notified if they won the auction (by receiving a succeeded message) or if they were out-bided. The law is presented in Figure 4. The rules of the law are followed by comments (in italic) that, together with the following discussion, should provide the reader with some understanding of the nature of LGI laws. Using the techniques described in this paper, one can easily develop laws for different types of auctions.

The first three rules deal with the initialization of the auction. By Rule $\mathcal{R}1$ the law specifies the auditor for this auction, the certifying authority (CA) is specified in Rule $\mathcal{R}2$, and the initial state is set to empty in Rule $\mathcal{R}3$.

Rule $\mathcal{R}4$ deals with the adoption of this law, allowing an agent to start operating either as a seller or as a buyer by getting the term `seller` or `buyer` in its initial control-state. The adoption of the law is conditioned on presenting a valid certificate, previously obtained from a certifying authority. Any agent that presents a valid certificate gets the term `certified` in its control state, allowing him to participate (as a buyer or as a seller) in any auction. The certificates expire after a specified period of time (100 seconds in our example). When that happens, Rule $\mathcal{R}13$ is trigerred and the term `certified` is removed from the control state of the agent. The agent is then not able to participate in any auction until he or she presents a valid certificate to its controller. We will now discuss how a buyer can make an offer and what are the possible answers from the seller.

First, by Rule $\mathcal{R}5$, the seller actually starts the auction for the product with the name P. The maximum price is initially set to 0. An obligation is imposed in the sense that the auction should be finished after a specified period of time. Buyers can now send offers using Rule $\mathcal{R}6$. According to the rule $\mathcal{R}7$, the bid is checked if it is the best offer. If it is, the maximum price is set to this value and the buyer that had previously the best offer is informed that his offer was out-bided triggering the execution of Rule $\mathcal{R}12$. If it not the best offer, according to Rule $\mathcal{R}8$, the buyer is informed that his or hers bid was rejected and the execution of Rule $\mathcal{R}11$ is triggered. When the auction is finished (rule $\mathcal{R}9$) the seller informs the winning buyer through Rule $\mathcal{R}10$.

We also try to minimize the possibility of a seller to act as a buyer in the same auction to artificially increase the price. When an offer is received by the seller (in rules $\mathcal{R}7$ and $\mathcal{R}8$), the law checks whether the seller is also a buyer or not.

Agents have to exchange messages using the format required by the law. The current implementation of LGI provides an XML interface, allowing agents to communicate using XML as the main language for describing the data. In this case, the law plays also the role of the schema for the XML documents that are exchanged between the agents.

3.2.1 Auditing

The LGI paradigm naturally supports the concept of auditing. An auditor is basically an agent that is not involved in the auction but that receives copies of the messages that were exchanged. If an auditor exists, an agent can request copies of the messages exchanged during the auction and use this information to find out about the behavior of some of the involved agents. An auction can have more than one auditor and the auction law specifies the messages that are sent to each of them. An agent can choose not to participate in an auction if it does not trust its auditors. Although the law specifies what messages are sent to the auditors, it imposes no restrictions in the way they handle the messages they receive.

3.2.2 Treatment of Complaints

An agent can complain about another agent (A) if he or she thinks that A did not have a correct behavior. Examples of incorrect behavior include: not sending the item once the auction is over, sending a defect product, not sending the payment, and so on. To handle these situations an auction

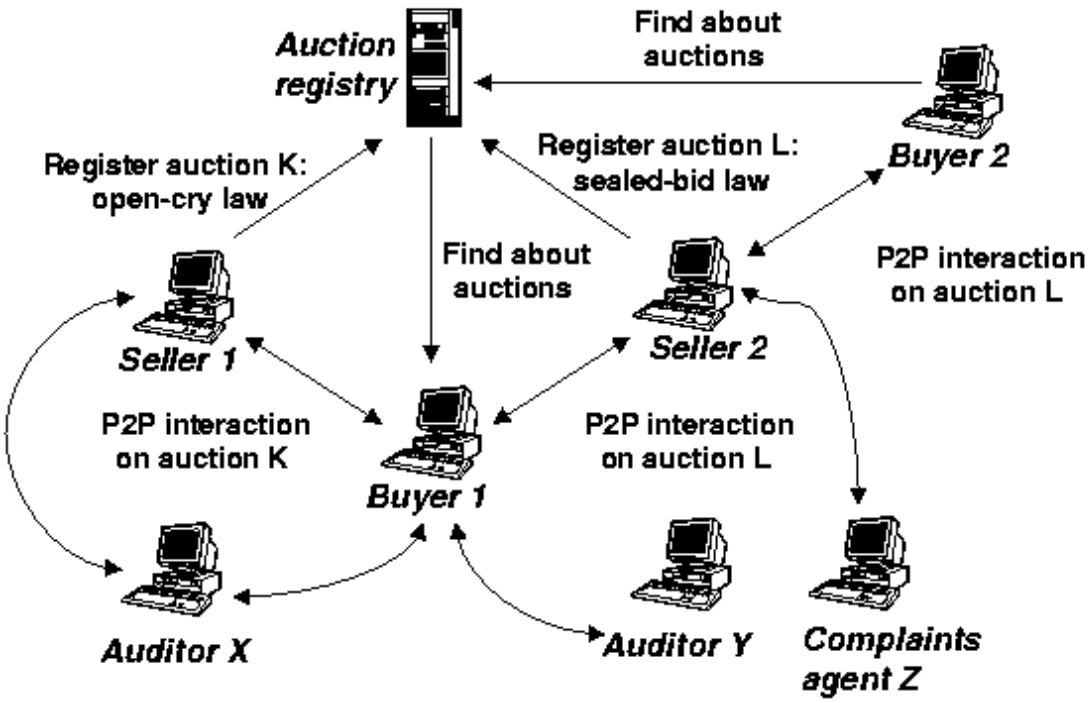

Figure 3: A snapshot of the system with 2 buyers and 2 sellers

law can specify a complaints agent, which is an analogue of the Better Business Bureau from the real life. A complaints agent interacts with the auditors and with the certifying authorities to get copies of the messages exchanged and the IDs of the involved agents. By processing this information, it can check if a given complaint is accurate or not and it can inform the certifying authorities about the improper behavior of agents.

An example of how a complaints agent can be used to increase the trust of agents in the correctness of the auction is the prevention of the artificial increase of the price by the seller. As we detailed in the previous section, the law can check if the seller is registered at the same controller as a seller and as a buyer. However, the law cannot check whether the seller registered himself as a buyer on another controller and participates in the auction as a regular buyer. Moreover, it might happen that "friends" of the seller participate in the auction to artificially increase the price.

Once a complaint is filed, the complaints agent can talk to the auditor to retrieve copies of all the exchanged messages and the real IDs (as are written in the certificates) of the agents. With this information at hand, it can find out about the malicious behavior of such a seller. When the seller certificate expires and he or she requests a new one from the certifying authority (CA), the CA can ask the complaints agent about the seller behavior and it can refuse to issue the new certificate.

4. RELATED WORK

Currently several Web sites provide auction services. These sites can be classified into two categories: consumer-to-consumer (C2C) and business-to-consumer (B2C). In C2C auctions, such as Ebay (http://www.ebay.com), users can act as both sellers and buyers, posting new auctions and participating in existing auctions. In B2C auctions, such as Egghead (http://www.egghead.com), users act only as buyers, participating in the available auctions. B2C auctions are less "risky" for buyers, since the seller is usually a "brand name" company that can be trusted. In C2C auctions, although the action site acts as a mediator, there is no guarantee about the reputation of the seller. As discussed throughout the paper, centralized auction systems cannot solve this issue since they do not allow auction participants to engage the entities they trust as part of the auction process.

The AuctionBot system [9] allows users to customize auction policies. The system is centralized, having a very similar structure to eBay (http://www.ebay.com) or other commercial systems. The main advantage of AuctionBot is that the user can parameterize the pre-defined auction policies (like dutch, sealed) to choose, for example, between the first prize or the second prize as the auction winner. The marketplace framework described in [3] also allows the definition of several auction and negotiation protocols. However, both systems do not support distributed enforcement and rely on a centralized marketplace, limiting the customization options of the auction participants.

The auction registry component of the proposed LGI-based architecture can be view as a yellow pages or discovery service, where buyers search for auctions. UDDI (Universal Description, Discovery and Integration) can be used to standardize the auction registry interface. UDDI is an "open industry initiative enabling businesses to (i) discover

$\mathcal{R}1$. `Directory(auditor, auditor@enterprise.com)`

The law specifies the auditors for this auction. Any number of auditors can be involved in a given auction.The roles and responsibilities of the auditors are described in the next section.

$\mathcal{R}2$. `Authority(ca,URL(http://aramis.cs.rutgers.edu:9020))`

The list of the certifying authorities (CA's) that are to be used for certification of the involved agents is specified. Any number of CA's can be involved in a given auction.

$\mathcal{R}3$. `InitialCS([])`

Initially, the control state is empty

$\mathcal{R}4$.
```
certified(X,certificate(issuer(ca),subject(Y),attributes([seller(N)])))
        :- do(deliver(X,certificate(issuer(ca),subject(Y),attributes([seller(N)])),X)),
           do(+certified),do(+role(seller)),repealObligation(endCertified(X)),
           imposeObligation(endCertified(X),100), do(deliver(X,attributes([seller(N)],auditor).
```

An agent can participate in the auction as a seller —i.e., having a term `role(seller)` *in its control state or as a buyer (with a similar rule). In order to participate in the auction it has to present a valid certificate. An obligation is imposed to limit the duration of the validity of the certificate (in this case the certificate is valid for 100s). The content of the certificate, along with the agent name, is also sent to the auditor.*

$\mathcal{R}5$.
```
sent(X,start(P,T),X) :- certified@CS, role(seller)@CS, do(+P),do(+max(P,0)),do(+winner(P,X)),
           do(imposeObligation(timeout(P),T)),do(deliver(X,start(P,T),auditor).
```

Only a seller can start the auction for the product P. From this moment on, offers are received and an obligation is imposed that the auction should stop after the specified period of time. The tuple corresponding to this product should have been already placed in the auction registry, otherwise the buyers cannot find about this auction. A copy of the message is sent to the auditor.

$\mathcal{R}6$.
```
sent(X,offer(P,M),Y)
        :- certified@CS, role(buyer)@CS, do(forward(X,offer(P,M),Y)), do(deliver(X,offer(P,M,Y),auditor).
```

The offer can be made by the buyer only if he or she has been certified. A copy of the message is also sent to the auditor.

$\mathcal{R}7$.
```
arrived(X,offer(P,M),Y) :- role(seller)@CS, max(P,Q)@CS,winner(P,Z)@CS, M>Q, not role(buyer)@CS, do(-max(P,Q)),
           do(+max(P,M)), do(-winner(P,Z)), do(+winner(P,X)), do(forward(Y,accepted(P,M),X)), do(deliver(Y,
           accepted(P,T,X),auditor), do(forward(Y,outbid(P,M),Z)), do(deliver(Y,outbid(P,T,Z),auditor).
```

If the offer is the best offer, it is recorded as the max bid for the item and the buyer becomes the current winner of the auction. We do not allow for the seller to bid on the same item, in order to prevent artificial increase of the price. Copies of the accepted and out-bid messages are also sent to the auditor.

$\mathcal{R}8$.
```
arrived(X,offer(P,M),Y) :- role(seller)@CS, max(P,Q)@CS,winner(O,Z)@CS, not role(buyer)@CS, Q >= M,
           do(forward(Y,rejected(P,Q),X)), do(deliver(Y,rejected(P,M,X),auditor).
```

If the offer is not the best offer, a message is sent to the buyer indicating that his or hers bid was rejected and informing value of the current maximum bid. We do not allow for the seller to bid on the same item, in order to prevent artificial increase of the price. A copy of the message is also sent to the auditor.

$\mathcal{R}9$.
```
obligationDue(timeout(P)) :- max(P,M)@CS,M>0,winner(P,X)@CS, do(-P), do(forward(Self,succeeded(P,M),X)),
           do(deliver(Self,winner(P,M,X),Self)), do(deliver(Self,succeeded(P,M,X),auditor).
```

When the auction is finished, the buyer with the biggest offer receives the succeeded message. This message also indicates the amount he or she has to pay. A copy of the message is also sent to the auditor.

$\mathcal{R}10$.
```
arrived(X,succeeded(P,M),Y) :- role(buyer)@CS, do(deliver).
```

The buyer that receives the succeeded message is the winner of the auction. The actual exchange of money and products is done by the two parties independently of LGI.

$\mathcal{R}11$.
```
arrived(X,rejected(P,M),Y) :- role(buyer)@CS, do(deliver).
```

The buyer is notified that his or hers bid was not accepted.

$\mathcal{R}12$.
```
arrived(X,outbid(P,M),Y) :- role(buyer)@CS, do(deliver).
```

The buyer is notified that he or she was out-bided.

$\mathcal{R}13$.
```
obligationDue(endCertified(X)) :- do(-certified).
```

The agent certificate expired without a replacement. When this happens, sellers are not able to start a new auctions (the current ones are not affected) and buyers are not able to send new offers.

Figure 4: Law $\mathcal{OPENCRY}$ for the open cry auction

each other, and (ii) define how they interact over the Internet and share information in a global registry architecture" (http://www.uddi.org).

5. CONCLUSIONS AND FUTURE WORK

Online auctions is an important application area - Forecast Research expects that in 2003 there will be a market of 14 million consumers and $19 billion in sales. In this paper we presented a novel system for supporting online auctions that takes full advantage of the distributed nature of the Internet.

In the proposed architecture, sellers can set up their own auction policies; and these policies are explicitly stated, readable by everybody, and strictly enforced by the LGI mechanism. Auctions are conducted in a totally distributed manner, through a peer-to-peer communication protocol among the several agents. There is no centralized authority that can act as a trusted mediator. However, we have shown how third parties, such as auditors and complaints agents, can participate on the auctioning process under a given law. Moreover, this architecture is not limited to auctions, but it can be applied to any online trading model.

We are currently working on the definition of laws for other types of negotiation. We are especially interested in studying the behavior of agents in the presence of several optional (and conflicting) laws. Another topic we plan to investigate is the integration of our system with the Web services paradigm, in which agents find out about each other using XML-based discovery mechanisms, such as UDDI (http://www.uddi.org), and exchange messages using XML and SOAP. We are particularly interested in investigating the relationship between laws and Web service description languages, such as WSDL (http://www.uddi.org). Another point of extension that we are currently working on is the development of a Web-based system for the definition of auction policies in a simple and straightforward manner. This interface allows users to define new laws visually, based on previously defined ones.

6. ACKNOWLEDGMENTS

The work reported here is supported in part by NSF grants No. CCR-98-03698, and by the NJ Commission on Science and Technology Excellence Award.

7. REFERENCES

[1] X. Ao, N. Minsky, T. Nguyen, and V. Ungureanu. Law-governed communities over the internet. In *Proc. of Fourth International Conference on Coordination Models and Languages; Limassol, Cyprus; LNCS 1906*, pages 133–147, September 2000.

[2] Ralph Cassady Jr. *Auctions and Auctioneering*. Univ. California Press, 1979.

[3] M. Kumar and S. Feldman. Internet auctions. In *Fifth USENIX Workshop on Electronic Commerce*, August 1998.

[4] H. G. Lee. Do electronic market marketplaces lower the price of goods. *Communications of the ACM*, 41(1):73–80, January 1998.

[5] H. G. Lee and T. H. Clark. Impact of the electronic marketplace on transaction cost and market structure. *International Journal of Electronic Commerce*, 1(1):127–149, Fall 1996.

[6] N. H. Minsky. The imposition of protocols over open distributed systems. *IEEE Transactions on Software Engineering*, February 1991.

[7] N. H. Minsky and V. Ungureanu. Law-governed interaction: a coordination and control mechanism for heterogeneous distributed systems. *TOSEM, ACM Transactions on Software Engineering and Methodology*, 9(3):273–305, July 2000.

[8] B. Schneier. *Applied Cryptography*. John Wiley and Sons, 1996.

[9] Peter R. Wurman, Michael P. Wellman, and William E. Walsh. The Michigan Internet AuctionBot: A configurable auction server for human and software agents. In Katia P. Sycara and Michael Wooldridge, editors, *Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98)*, pages 301–308, New York, 9–13, 1998. ACM Press.