Requirements Analysis Document

Michael Buchanon, Bianka Iliev, Jack Daoud, James Dennig

Realtime Commercial Bidding System
Team1

March 15, 2002
Contents

1 Introduction .................................................. 3
  1.1 Problem Description ......................................... 3
  1.2 Motivation .................................................. 3

2 Overview .................................................... 3

3 Requirements ............................................... 4
  3.1 Sign up With Auction Site .................................. 4
  3.2 Join Auction ............................................... 4
  3.3 Withdraw .................................................. 4
  3.4 Notification ............................................... 4
    3.4.1 End of Auction ....................................... 4
    3.4.2 New best bid ......................................... 4
  3.5 Bid Placement ............................................. 5
  3.6 Reserve/Non-Reserve ..................................... 5
  3.7 Bid History ............................................... 5
  3.8 Auctions start Automatically ............................ 5
  3.9 Create Auction .......................................... 5
  3.10 Auctioneer ............................................... 6
  3.11 Auctions with no winner ................................. 6

4 UML Analysis ............................................... 7
  4.1 Use Cases ................................................. 7
  4.2 Object Model .............................................. 12
    4.2.1 Data Dictionary .................................... 12
    4.2.2 Class Diagram ..................................... 14
  4.3 Dynamic Model .......................................... 15
    4.3.1 State Diagrams .................................... 15
    4.3.2 Sequence Diagrams ................................. 17

List of Figures

1 UML Use Case Diagram ....................................... 7
2 UML Class Diagram .......................................... 14
3 Extended State Machine for Auction Site ................... 15
4 Extended State Machine for Auctioneer ..................... 16
5 Sequence Diagram for Failed Login ........................ 17
6 Sequence Diagram for Joining an Auction ................... 19
7 Sequence Diagram for Creating an Auction .................. 21
1 Introduction

With the advent of widespread Internet use among the general public, many traditional commercial practices have found a new medium to make its products and services more widely available. One of the more popular services to come about are distributed online brokering systems for commercial products.

The following document is a requirements analysis for a Realtime Commercial Bidding System. The purpose of this document is to reveal the functionality, components, and rules by which our Realtime Commercial Bidding System will be run. The requirements analysis will be divided into four distinct sections: introduction to the domain, overview, specific requirements, and UML analysis of the component and user interaction within the domain of the Realtime Commercial Bidding System. The document includes individual Use-Case scenarios based on use case diagrams, an Object Model consisting of a Class Diagram, Dynamic Models, State Diagrams to thoroughly illustrate the system functionality, and Sequence Diagrams showing the flow of events for particular scenarios.

1.1 Problem Description

We will be designing the user interface and system functionality for our distributed online brokering system. The system will consist of five main components needed to represent a distributed online brokering system. The needed components are the Auction Site, Auction, Auctioneer, User, and Items. Auction refers to only one auction of goods or services. The auctioneer is the automaton that runs the auction and is the primary facilitator between the Auction creator and the bidders. The auction creator is the individual that initiates the brokering of a good or service and determines what characteristics the auction will be run by. The bidder is the individual that bids on the good or the service.

1.2 Motivation

Many distributed online brokering systems have already found great success and established that there is a market for this type of commercial service on the Internet. However, none are run in real time or offer the user the amount of up to date information that simulates a real time auction. Our Realtime Commercial Bidding System will offer the Auction creator and the user current information as to the state of the Auction as it is actually occurring. Our system also offers the auction creator a greater variety of options as to how the auction will be conducted. Specifically, our system allows the auction creator to not only sell a good or service that they are currently in possession of, but also to solicit bids for a good or service that they are interested in obtaining. This concept is referred to as a Reverse Auction.

2 Overview

The Realtime Commercial Bidding System is structured around the concept of an Auctioneer. The Auctioneer acts as the middleman between the auction creator and the bidders of the current auction and as a referee for the Auction Site, lowering the need for human staffing. When an auction is created the Auctioneer conducts the auction based on the characteristics defined by the Auction Creator. When the Bidder makes a bid the Auctioneer receives the bid and determines whether the bid is acceptable. If the bid is acceptable the Auctioneer informs the Auction Creator and the Current Bidders that there is a new best bid and what that bid was. The Bidder is also
informed when they currently have the best bid. If the bid is unacceptable then only the Bidder is informed that their bid was not accepted and the state of the Auction does not change. The Auctioneer mediates these exchanges throughout the duration of the auction. At the end of the Auction, the Auctioneer contacts the winner of the Auction, if any, and the Auction Creator upon the completion of the auction with contact information and transaction specifics.

3 Requirements

3.1 Sign up With Auction Site

The user shall provide contact information such as a phone number, address, and payment information. The user will also provide a username and password, which will be used to login to the Auction Site. All of this information shall be stored on a central server maintained by the Client. This information will be used by the Auction Site for validating logins, contacting users, and billing users.

3.2 Join Auction

In order to bid, a User must join an auction. From conversations with the Client, this can be before or after an auction has started. A list of current and future auctions will be provided after the user logs into the Auction Site. The user can select any auction and choose to join that auction.

3.3 Withdraw

A user can choose to withdraw from an auction. However, if their bid is the current best bid their withdrawal will not be accepted. When a user is able to withdraw, the Auctioneer will no longer accept bids from this user.

3.4 Notification

3.4.1 End of Auction

The auctioneer keeps a record of all the Users and the two Best Bidders in order to determine the winner of the auction. If someone wins the auction, then all the active bidders including the winner and the Auction Creator will be notified. The Auction Site will be notified of the end of the auction and the winning price. Notification will display on the client screen to inform the bidders that the auction has ended and who has won. If the auction ends with no winner, all parties will be notified of this as well.

3.4.2 New best bid

From the client specification, we shall keep track of the highest bid in a straight auction and the lowest bid in a reverse auction. This is so that when the auction ends, either by timeout or by reaching the designated end time, we can determine a winner. We shall also keep track of the second best bid in any given auction so that the Auctioneer can declare that user the winner in the event that the current best bidder is disqualified. From conversations with the customer, we know that we should notify all users when the Best Bid has changed. This shall be done via a notification window within the Auction client. The user's identity shall remain anonymous too all but the creator of the auction until the end when a winner is declare.
3.5 Bid Placement

All registered users must have access to the current list of bids and must be able to join a certain auction. Bids are received by the auctioneer from bidders in real time. The Auction Creator shall not be able to bid in the Auction. All currently active bidders will be notified about the best bids every time a new best bid comes out. If the users bid is not the Best Bid, it is disregarded by the Auctioneer. In either case, the User's bid history is updated to reflect the submitted bid.

3.6 Reserve/Non-Reserve

The auction could be in two different modes, the "Reserve" and the "Non-Reserve" mode. In a reserved auction, a minimum or maximum price must be met to complete a sale, depending on the type of auction. The seller will not sell his/her item below/above that price. If the auction is a "Non-reserve" auction, then the best bidder wins regardless of the price.

3.7 Bid History

From our conversations with the client, it is obvious that the user will keep a list of bids for the auction in which they are currently active. Also from our talks with the client it has become apparent that the client would like to keep a running history for each user. Consisting of all the bids and auctions participated in by any given user. These past bid histories are to be stored on a server along with the rest of the clients contact information. They will be retrieved when the user logs into the Auction Site.

3.8 Auctions start Automatically

From the specification, auctions are allowed to start in the future. When an Auction Creator creates an auction, they specify the time and date that the auction will start. The defaults are the current time and the current date. In this case, once created the auction would be live. If the Creator specifies a date or time in the future, then this auction would start at that time. Bidders for this auction will not be able to place bids until after the auction starts. If a bidder attempts to bid before the auction has started they will receive a notification that the auction has not yet started. When the time has arrived for the auction to start all bidders currently waiting will be notified of the start.

3.9 Create Auction

From the specifications requested by the client, the user must have the option of creating an auction so that they may sell items or request goods and or services. The auction site should allow two types of auctions, the traditional auction in which a client sells an item to a group of bidders for the highest possible price. Or a Reverse Auction in which a client will be buying a certain items or services from a group of bidders. In both cases the automated Auctioneer will mediate these auctions. As show in the sequence diagrams, after the user logs in and is authorized, they should have at least two choices, join a current auction or create an auction of their own. If they decide to create an auction, then they should supply a name, description, start and end times, start and end dates, initial price(if any), the reverse price(if any) and if it is a normal or reverse auction. The client has asked that the user be able to configure the amount of dead-time before the auction ends. By default, this time will be ten minutes. If this dead-time elapses and there is now activity in the auction, it will end automatically. Otherwise, the auction will end at the specified end time.
3.10 Auctioneer

We shall provide an automated Auctioneer in order to reduce staffing requirements and to make this system as simple to run as possible. The auctioneer will keep the users notified of all that happens in the auction by giving them status updates in their clients(code). From the Class Model, it is clear that there is one Auctioneer for every Auction. The Auctioneer will also implement any business logic that is needed for the Auction to proceed. Such as naming the lowest bid the best bid in a reverse auction or the highest bid the best bid in a normal auction. The Auctioneer will do very little actual computation. For resource management, most of the computation of the auction will be done on the clients computer. This will save us substantial amounts when the site grows to many users.

3.11 Auctions with no winner

From our conversations with the client, it has been determined that there is the possibility of an Auction ending with no winner. This can happen if the reserve is not met or if the auction ends with no bids being made, either through reaching the end time or through inactivity for longer than the length of the dead-time. The all users in the auction will be notified that there is no winner via their auction clients(code).
4 UML Analysis

4.1 Use Cases

The purpose of a Use-Case model in figure 1 is to demonstrate how the whole Bidding system in general interacts with the characters and sub-systems that are a part of it. The Use-Case diagram consists of a system boundary that is represented by the rectangle in the figure. The idea behind the system boundary is that separates the actors and services in the outside environment from the services provided by the system. Actors are the entities that interact with the system. In the Bidding system we have the User, Auctioneer, Auction Site and Security as actors. The Use-Case model is providing us with a model that gives us a visual idea of how the actors interact with the system. As shown in the figure 1, the lines between the actors and the use cases shows which goals the actors are involved with. Inside the bounded system we see lines connecting the different use cases, this describes how these use cases are related to each other, it describes the nature of the relationship between them within the system.

![Use-Case Diagram](image-url)

Figure 1: UML Use Case Diagram
Use case: Bid on Item  
Actors: User, Auctioneer  
Type: Primary  
Description: The user places a bid on the item up for auction. The Auctioneer notifies the user when he/she becomes the highest bidder and if they’ve won the auction when it’s over.  
Cross-reference:  
Includes: none  
Extended by: Withdraw, Create an Auction  
Extends: none

Use case: Get List of Auctions  
Actors: User  
Type: Primary  
Description: After the User logs in, A list of Auctions is available for the user to join.  
Cross-reference:  
Includes: none  
Extended by: none  
Extends: none

Use case: Get Bid History  
Actors: User  
Type: Primary  
Description: After the User joins an auction, A list of bids the user has made is available to them.  
Cross-reference:  
Includes: Log In, Join Auction  
Extended by: none  
Extends: none
<table>
<thead>
<tr>
<th>Use case:</th>
<th>Connect to Auction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actors:</td>
<td>User, Auctioneer</td>
</tr>
<tr>
<td>Type:</td>
<td>Primary</td>
</tr>
<tr>
<td>Description:</td>
<td>After the User logs in, the user chooses an Auction from the list available and chooses join.</td>
</tr>
<tr>
<td>Cross-reference:</td>
<td></td>
</tr>
<tr>
<td>Includes:</td>
<td>none</td>
</tr>
<tr>
<td>Extended by:</td>
<td>none</td>
</tr>
<tr>
<td>Extends:</td>
<td>Create an Auction</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Use case:</th>
<th>Authorization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actors:</td>
<td>User, Security</td>
</tr>
<tr>
<td>Type:</td>
<td>Primary</td>
</tr>
<tr>
<td>Description:</td>
<td>Is part of the Log In process. Checks to see if username and password are valid.</td>
</tr>
<tr>
<td>Cross-reference:</td>
<td></td>
</tr>
<tr>
<td>Includes:</td>
<td>none</td>
</tr>
<tr>
<td>Extended by:</td>
<td></td>
</tr>
<tr>
<td>Extends:</td>
<td>none</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Use case:</th>
<th>Log In</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actors:</td>
<td>User, Security</td>
</tr>
<tr>
<td>Type:</td>
<td>Primary</td>
</tr>
<tr>
<td>Description:</td>
<td>User enters username and password, which is authorized by Security</td>
</tr>
<tr>
<td>Cross-reference:</td>
<td></td>
</tr>
<tr>
<td>Includes:</td>
<td>Authorization</td>
</tr>
<tr>
<td>Extended by:</td>
<td>none</td>
</tr>
<tr>
<td>Extends:</td>
<td>none</td>
</tr>
</tbody>
</table>
Use case: Withdraw

**Actors:** User

**Type:** Primary

**Description:** User decides to withdraw from an auction.

**Cross-reference:**

**Includes:** none

**Extended by:** none

**Extends:** Bid on Item

---

Use case: Maintain list of Bidders

**Actors:** Auctioneer

**Type:** Primary

**Description:** Auctioneer maintains a list of current bidders. The list is updated when someone joins or quits an auction.

**Cross-reference:**

**Includes:** none

**Extended by:** none

**Extends:** none

---

Use case: Notify Active Users

**Actors:** Auctioneer

**Type:** Primary

**Description:** Auctioneer notifies bidders if they have the new highest bid, when an auction starts, ends and if the user has won the auction.

**Cross-reference:**

**Includes:** none

**Extended by:** none

**Extends:** none
<table>
<thead>
<tr>
<th>Use case:</th>
<th>Set price/date/time of Auction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actors:</td>
<td>User, Auctioneer</td>
</tr>
<tr>
<td>Type:</td>
<td>Primary</td>
</tr>
<tr>
<td>Description:</td>
<td>User describes what is being bid on. Auctioneer stores this information.</td>
</tr>
<tr>
<td>Cross-reference:</td>
<td></td>
</tr>
<tr>
<td>Includes:</td>
<td>none</td>
</tr>
<tr>
<td>Extended by:</td>
<td>none</td>
</tr>
<tr>
<td>Extends:</td>
<td>none</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Use case:</th>
<th>Create Auction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actors:</td>
<td>User, Auctioneer</td>
</tr>
<tr>
<td>Type:</td>
<td>Primary</td>
</tr>
<tr>
<td>Description:</td>
<td>User creates an auction to sell an item.</td>
</tr>
<tr>
<td>Cross-reference:</td>
<td></td>
</tr>
<tr>
<td>Includes:</td>
<td>set price/date/time of Auction</td>
</tr>
<tr>
<td>Extended by:</td>
<td>Connect to Auction</td>
</tr>
<tr>
<td>Extends:</td>
<td>Bid on Item</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Use case:</th>
<th>Monitor Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actors:</td>
<td>User, Auctioneer</td>
</tr>
<tr>
<td>Type:</td>
<td>Primary</td>
</tr>
<tr>
<td>Description:</td>
<td>A User that creates an Auction can monitor the events of that auction. The Auctioneer will notify the User of the goings on of the Auction.</td>
</tr>
<tr>
<td>Cross-reference:</td>
<td></td>
</tr>
<tr>
<td>Includes:</td>
<td>none</td>
</tr>
<tr>
<td>Extended by:</td>
<td>none</td>
</tr>
<tr>
<td>Extends:</td>
<td>none</td>
</tr>
</tbody>
</table>
4.2 Object Model

4.2.1 Data Dictionary

Auction Site: A collection of auctions past, present and future.

- Properties
  1. auctionList: A list maintained by the Auction Site of the currently available auctions.
  2. knownUserList: List of users that are currently logged into the system.

- Methods
  1. Control(): A method used by site administrators to step in and take control of a particular auction through the auctioneer if the need ever arises. A failsafe to take control temporarily away from automation.
  2. CreateAuction(): A method in the Auction Site used to spawn an Auctioneer for a given auction based on information supplied by the user.
  3. GetAuctionHistory(user): A method in the Auction Site that returns the auction history for the given user.
  4. ListAuctions(): A method in the Auction Site that enumerates the current auctions.

Auction: An event run by the auctioneer to sell or buy goods and/or services.

- Properties
  1. auctionID: A unique string by which a specific auction is identified by the auction site.
  2. startTime: When the auction starts.
  3. startDate: The date on which the auction starts.
  4. endTime: When the auction ends.
  5. timeOut - If set, the Auctioneer will end the Auction after this many minutes of inactivity. The default is 10 minutes.
  6. reserve: The minimum price that a normal auction will sell for or the maximum price a reverse auction will sell for.
  7. initialPrice: The price at which the auction will start.
  8. isRegular: boolean variable that is true if the auction is a normal auction and false if it is a reverse auction.

- Methods
  1. IsReserve(): A method of the Auction that returns true if the ReservePrice is greater than $0.00 and false otherwise.
**Auctioneer** : Automation that controls the auctions, notifies the users of the new best bid, and notifies the winner of the auction if one exists.

- Properties
  1. auctionCreator: A reference to the user object that created the current auction.
  2. bestBid: Current Bid that is the highest in a normal auction and the lowest in a reverse auction.
  3. bestBidUser: current user object that has the best bid.
  4. nextBestBid: The second best bid in the current auction tracked in case some circumstance invalidates the current bestBidUser.
  5. nextBestBidUser: the user object that has the second best bid.
  6. listOfUsers: A list of all users currently in the auction.

- Methods
  1. Bid(): A function in the auctioneer that accepts a bid from a user.
  2. JoinAuction(user): Adds the user to the current auction.
  3. NotifyAuctionSite(): Used to notify the auction site of completion of the auction and also the details of the auction such as winning user, and price.
  4. NotifyUsers(): A function in the auctioneer that is used to inform all of the users of auction status. Such as closing and new Best Bids.

**Item** : the thing or services to be sold or bought in the Auction.

- Properties
  1. Description: a listing of the properties of the item up for auction.

- Methods
  1. None

**User** : The person who uses the Auction Site via the auction client.

- Properties
  1. userName: A unique identifier that is used to track all customers of the Auction Site.
  2. bidHistory: A history of all the bids that a particular user has placed.

- Methods
  1. None

**Security** : Security program installed to protect users privacy and auction site. Consists of encryption schemes and contains user names and passwords.
4.2.2 Class Diagram

The Class Diagram in figure 2 shows all the classes in the Bidding system including all the attributes and operations involved with each class. The diagram also shows the associations between the classes and what each is composed of. The class diagram is a key ingredient in the requirement analysis phase. The class diagram consists of six classes: Auction, Auction Site, Auctioneer, Item, Security and User are all classes. The User class has several attributes and operations associated with it. The User interacts with the Auction site, Security and Item. The User’s initial registration and login have to go through Security in addition to the notification and making a bid are also associations between the User and Security classes. The User owns an Item that has one attribute (its description) and interacts with the Auction Site by demanding a list of auctions. Security is an aggregation of the Auction site and it gives authorization to registered users and notifies both the user and auctioneer of bids and other conditions. The Auction site has several functions, it receives registration information from security and it provides the user with a list of auctions available. The Auctioneer is the backbone of the system that interacts with Security, Auction Site and Auction. Bids and notifications are sent between the Auctioneer and the Security, if the user decides to join or create and auction, then notification is sent to the Auction Site. The Auctioneer also informs the user of information such as best bid, start of the Auction, and the end of the Auction. The Auction class consists of information pertaining to the auction such as the initial price, start time, regular or reserve and ending time. It interacts with auctioneer and the Item class.

Figure 2: UML Class Diagram
4.3 Dynamic Model

4.3.1 State Diagrams

The following figure (fig. 3) represents the State diagram of our commercial bidding system. It consists of states and transitions with each transition accompanied by a label of the event and action that occur when the transition occurs. The states are represented in the diagram by rectangles with curved corners while the transitions are the arrows that can be seen between the states. As can be seen on the diagram some brackets exist with information in them, this information is simply the conditions needed for a transition to happen.

The starting point of the state diagram is the "Idle" state. This state asks the user if to log in to the auction site, if they accept, then the transition takes them to the next state which is the "logging in" state and another window requests a username and password to be verified. The User also has the option of quitting and this will take him back to the "Idle" state. If the login event occurs and Security determines that the user is not a valid user, then that takes the User to the "Invalid Login" state and the Auction Site and the User are both notified and we transition to the "Idle" state. In the "Logging In" state the User submits his information and that goes to the next state, the "Verification" state. Here the User is being verified and if they are an invalid User then they are sent back to the "Logging In" state. Otherwise, if Security determines that the User is a valid user we reach the "Logged in" state. At the "Logged in" state, the User is faced with three options, after the auction site displays a list of the current auctions, the user can join an auction and this will lead him to the "Join Auction" state. If the User is in the "Join Auction" state then they can quit and go back to the "Logged in" state only if they do not have the best bid, the quit event will happen here if this condition is satisfied. If the User decides to create a new auction then the "doCreate" event will move him to the "Create Auction" state, if he decides to cancel then he will go back to the "Logged in" state. If the submit event occurs and the Auction is valid then the Auction will be added to the existing list of Auctions and then the User will be back in the "Join Auction" state. His final option allows him to quit the "Logged in" state taking him back to the "Idle" state.

Figure 3: Extended State Machine for Auction Site
In figure 4 we show the dynamic model for the auctioneer. The Auctioneer starts in the idle state. There are only two ways the Auctioneer moves out of the idle state. The first is when the Auctioneer receives a bid from a user. When a bid is received the Auctioneer moves to the check bid state. This is where the most comparisons are made within the Auctioneer to determine its next course of action. The first case is the bid is not a valid bid. An example of this case would be that the format of the bid is incorrect, such as a user entering a series of letters instead of a dollar amount. In this case the user is notified that it is an invalid bid and the Auctioneer returns to the idle state. The next case is when a valid bid is received from a user that is the auction creator. The auction creator is not allowed to bid on their own auction, so in this case the user is notified that their bid will not be accepted because bids from the creator are not allowed. The next case is when a valid bid is received but the bid does not beat the current best bid. This occurs when the auction is a standard auction and the bid received is lower than the current best bid. It also happens when the auction is a reverse auction and the bid is higher than the current best bid. When this occurs, the user is notified that there bid is not the best bid and the auctioneer returns to the idle state. The last scenario that can occur from the check bid state is when a valid bid is received and it is better than the current best bid. This occurs in a standard auction when the bid is higher than the current best bid, and it occurs in a reverse auction when the bid is lower than the current best bid. When this occurs the user is notified that they now have the best bid and the rest of the users are notified that there is a new best bid. The auctioneer then returns to the idle state. The second way that the auctioneer moves out of the idle state is when the timeout event has occurred. The timeout event occurs when there hasn’t been any activity within the auction and the auctioneer has been in the idle state for the amount of time set by the auction creator. When this happens the users are notified that the auction has ended and also who the user with the best bid is.

Figure 4: Extended State Machine for Auctioneer
4.3.2 Sequence Diagrams

Figure 5 on page 17 shows what happens when a User attempts to log in to the Auction Site and fails.

1. The Auction Site requests the User to Login
2. The User accepts
3. The Auction Site prompts the User for Username and Password
4. The User sends Username and Password to the Auction Site
5. The Auction Site requests Verification of Username and Password from Security
6. Security sends message Password Rejected to the Auction Site
7. Password not accepted message displayed to User

Figure 5: Sequence Diagram for Failed Login
Figure 6 on page 19 shows what happens when a User attempts to join an Auction and Bid on an item.

1. The Auction Site requests the User to Login
2. The User Accepts
3. The Auction Site prompts for the Users Username and Password
4. The User sends their Username and Password
5. The Auction Site requests verification of Username and Password from Security
6. Security sends to the Auction Site that Username and Password are accepted
7. The Auction Site displays the Join/Create window
8. User Selects an Auction to join, this is sent to Auction Site
9. Auction Site displays Bid Window to User
10. User Makes Bid and is sent to Security
11. Security forwards Bid amount to Auctioneer
12. Auctioneer notifies the user of the new best bid
13. User sends Request Quit to the Auctioneer
14. Auctioneer sends message(Can’t Quit if Highest Bidder) to User
15. Auctioneer sends message(Auction Over, You Win) to User
16. Auctioneer sends message(Auction Over) to Auction Site
Figure 6: Sequence Diagram for Joining an Auction
Figure 7 on page 21 shows what happens when a User attempts to Create an Auction.

1. The Auction Site requests the User to Login
2. The User Accepts
3. The Auction Site prompts for the Users Username and Password
4. The User sends their Username and Password to Security
5. Security sends to the Auction Site that Username and Password are accepted
6. The Auction Site displays the Join/Create window
7. The User selects Create and sends to the Auction Site
8. The Auction Site displays the Create Auction Window
9. The User sends the Auction information to the Auction Site
10. The Auction Site creates the Auction and the Auctioneer
11. The Auction Site displays the Auction Window
12. The User makes a bid and is sent to Security
13. Security forwards the bid amount to the Auctioneer
14. Auctioneer notifies the user that the Auction Creator cannot bid.
15. The Auctioneer sends Message(Auction Over, Who Won) to User
16. The Auctioneer sends Message(Auction Over) to Auction Site
Figure 7: Sequence Diagram for Creating an Auction