Case Study: Implementing a Web Based Auction System using UML and Component-Based Programming*

Frederick T. Sheldon
Software Engineering for Dependable Systems Laboratory
School of Electrical Engineering and Computer Science
Washington State University
email: sheldon@acm.org

Young-Jik Kwon
SEDS Laboratory Visiting Professor
School of Computer and Information Engineering
Taegu University
email: yjkwon@eeecs.wsu.edu

Young-Wook Baik
School of Computer and Information Engineering
Taegu University
email: ywbaik@taegu.ac.kr

Kshamta Jerath
Software Engineering for Dependable Systems Laboratory
School of Electrical Engineering and Computer Science
Washington State University
email: kjerath@eeecs.wsu.edu

Abstract

This paper presents a case study highlighting the best practices for designing and building a web-based auction system using UML (Unified Model Language) and component-based programming. We use the Use Case, Class, Sequence, and Component Diagrams offered by UML for designing the system. This enables new functions to be added and updated easily. Our implementation, with its basis in component-based programming, enabled us to develop a highly maintainable system with a number of reusable components: the MethodofBidding (the bidder can bid at three different frequencies - fast, medium or leisurely), the Certification (Identity verification function), and the RegisterGood (Product entry function) Components. Further, the system uses intelligent agents that permit fair help to bidders participating in auctions and at the same time achieve maximum profit for the seller. The design and implementation environment, along with the tools used, provide excellent support for the successful development of the system.

Keywords - Electronic commerce, UML, Component based programming, Online auction

* Upon translation from Word document to pdf the figures have become undecipherable. Given the last minute occurrence of this situation we are submitting the paper as is. The problem, however, will be fixed for the final version.

1 Contact Author: Frederick T. Sheldon. Stettener Strasse 50/2, 73732 Esslingen, Germany. (+49 711 174 1339 Office | +49 179 675 9316 Handy | +49 711 1747054 Fax) sheldon@acm.org
1. **Introduction**

In the past few years, the electronic marketplace has witnessed an exponential growth in worth, size, and usability. Projections indicate that this trend will intensify in the coming years [1]. Online auctions are a major component of the electronic marketplace that makes use of electronic commerce mechanisms.

With the help of the fast-growing Internet environment, the products that were previously purchased in a traditional market (i.e., Brick-and-Mortar) can now be obtained conveniently via online auction systems. An online auction system works by allowing users to register at the site to sell and/or buy products. A participant must be registered at the site to partake. When a Seller decides to sell a product, he logs-in and sets up an auction by specifying the product details, the minimum bid amount, the duration of the auction, etc. A Purchaser logs-in and participates in an auction by placing bids for a chosen product and has the chance to bid again during the time the auction is active. The bidding ends when the auction time expires and the highest bidder is the one who has successfully made a purchase.

Purchasers participate in existing auction systems knowing only the product description/price while the Seller provides the detailed product description. Since the Purchaser participates in an auction without accurate information about the products, it is difficult for him to cope with rapid changes to the auction system’s environment. At the same time, it is necessary for one to implement a Web based auction system with the capability to cope with rapid bidding and changes in the price of products, the system processing speed, etc., while, at the same time being maintainable and reusable.

In this case study, we undertake the task of designing and developing a Web-based auction system that satisfies the above stated requirements. To do this, we need an effective programming methodology. Object Oriented Programming (OOP) method has been used in the past because it was thought to conquer the problems mentioned, just as structure and encapsulation improve reusability in a semiconductor chip or component [2]. However, OOP doesn’t guarantee reusability and is unsuitable for a large-scale project because of the complexity of relations among the objects. Moreover, it is difficult to support perfect encapsulation due to inheritance among classes.

The Unified Modeling Language (UML) has emerged to provide a standardized notation for describing Object-oriented models. However, for the UML notation to be effectively applied, it must be used in conjunction with an Object-Oriented Analysis and Design method [3]. Object-Oriented Analysis and Design (OOAD) describes a community of methodologies for producing business component based software. The methodology outlines the system development lifecycle identifying the tasks and deliverables in an object-oriented project. Using a combination of UML notation and OOAD (Object-Oriented Analysis and Design) process, we can reduce the system development life cycle, easily maintain
the system, and improve the reusability of modules.

Complex software takes more time to develop and is difficult and costly to maintain. Reusable software components present a solution to these problems because components can be plugged into other components with very little effort. Component-based programming results in maintainable and reusable code.

Consequently, we design and implement the Web based auction system using UML and components.

The web-enabled auction system uses four UML Diagrams and three components that are reusable at the specification-level. The process models used for the development of the system are shown in Figure 1a and Figure 1b.

The paper is organized as follows: Section 2 discusses the background information and related research in the areas of UML, components and auction systems. Sections 3 and 4 describe our empirical study, and the results and analysis of the study respectively. Finally, the conclusion and the future research direction are presented in Section 5.

2. Background Information and Related Research

Developing an online auction system requires making decisions and selecting technologies to support those decisions. We developed the online auction system described in this paper using UML and components and therefore discuss components, online auction systems and UML related research.

2.1 Unified Modeling Language

Traditionally, requirements analysis consisted of identifying relevant data and functions that a software
system would support. The data to be handled by the system could be described in terms of entity-relationship diagrams, while the functions could be described in terms of data flows [4]. Object-oriented software development utilizes new design methodologies, and computer-aided software engineering tools such as Rational Rose can support these methodologies [5]. The UML (Unified Modeling Language) is a language used to specify, visually model [6], and document the artifacts of an Objected-Oriented system under development. It represents the unification of a number of ideas from different methodologists as shown in Figure 2. Using UML to design a system improves its maintainability and reusability. Object-oriented analysis techniques offer class, use case, state chart, sequence, and other diagrammatic notations for modeling [7].

Wolf et. al. [8] introduced UML briefly on the basis of the different diagrams and successfully adapted the activity flow model (on the basis of UML) to a professional project of a leading German organization in the banking sector. Kari Kivisto [9] analyzed features of a use case-driven, architecture centric, iterative and incremental development process model and suggested some extra features needed in developing Object-Oriented Client/Server applications (including the Internet).

Paul Allen [10] presented a pragmatic approach to using UML within the context of a service-based architecture and component-based process grounded in business requirements. Domain modeling was used, governed by the concept of service layering, separation of operational from core functionality, the application of patterns and design guidelines. Use cases were employed not to create the service-based architecture (appropriate for legacy systems) but to test it out and as a vehicle for solution delivery. Other techniques such as collaboration diagramming and sequence diagramming were also used.

Back et. al. [11] presented an analysis method based on the notion of a contract, understood as a formal specification of the behavior of the UML Use Cases, to enhance the (informal) Use Case Diagrams by providing the contracts as a formal counterpart. Hassan Gomaa [12] described how the UML Notation could be used to model families of systems including kernel, optional and variant classes, feature dependencies and object interdependencies.
We selected Use Case Diagrams, Sequence Diagrams and Component Diagrams for analyzing the user’s requirements, the ordering of messages and documenting relationship among components. We selected Class Diagrams for representing the static structure of classes.

2.2 Component-Based Programming

Component-Based programming enables fast deployment of maintainable software by reusing prefabricated components that are independent executable units. Individual components can be custom-made to meet new requirements and can be rearranged in different compositions. Reusability and Maintainability are the two main advantages of component-based programming.

Alexander Repenning et. al. [13] emphasized that component-based development is one solution that has a long tradition of advocates, is recommended by leading experts, and is quickly gaining support. Components are highly reusable units of software functionality and they let developers conceptualize software as interconnect-able building blocks. These software components support modular programming practices. The same organization assembling the components might produce and maintain them in complete applications, or acquire them from third-party developers producing so-called commercial-off-the-shelf software packages.

J.P. Sousa and D. Garlan [14] studied the use of component integration frameworks to prescribe an architecture that permits flexible composition of third-party components into applications. A good example is Sun Microsystems’ Enterprise Java Beans (EJB) framework, which supports object-oriented distributed enterprise-level applications. The key idea in our approach is to take an architectural view of the problem, making explicit the protocols of interaction between the principle parts of the framework.

Daniel Schwabe [15] emphasized the importance of Web design reuse rather than code reuse. The study applied the Object-Oriented Hypermedia Design Model extension, OOHDM-frame, to determine the key architectural components and design structures that lend themselves to reuse. Forsell et. al. [16] emphasized that one approach to create quality software is to reuse components as building blocks of the new software. Recently there has been more interest in creating component-based software development methods to support this.

We implemented the auction system using component based programming for easy maintenance as well as convenient reuse of these components. Three reusable components were developed – MethodofBidding, Certification and RegistrationGood Components, each handling specific and well-defined functions in the auction system.
2.3 Auction Systems

Auction systems are a major component of the electronic marketplace that allow users at any site to sell and buy products. The sellers set up auctions for their products while the purchaser who bids the highest amount wins the right to purchase the product in an auction.

Jane Hillston and Leila Kloul [17] have investigated an on-line auction system using the process algebra formalism PEPA (Performance Evaluation Process Algebra) for the impact of introducing active nodes into the network on server performance. They were interested in the relation between adjacent links in terms of the cache and system throughput. For our intentions, we focus on how the Web based system can provide an effective and fair auction process, not by improving system throughput via different hardware topologies but by producing valuable suggestions on bidding price.

Martin Bichler [18] investigated multi-attribute auctions which automated negotiation on multiple attributes of a deal by combining decision analysis techniques and single-sided auction mechanisms. The results indicated that it is better to consider the factors for multi-attribute auctions like quality of products, price, payment method, delivery etc., and this requires a more dynamic bidding procedure.

Kao et. al. [19] studied basic problems in a multiple-object auction model using methodologies from theoretical computer science. In the two-bidder case, they derived an optimal randomized bidding algorithm, by which the disadvantaged bidder (where an adversary bidder knows his bidding algorithm) could procure at least half of the auction objects despite the adversary’s prior knowledge of his algorithm. Sandholm, [20] implemented the mathematical algorithm for winner determination in combinatorial auction systems.

Baik and Kim [21] implemented an auction system where the buyer purchases needed parts of a configuration using interactive methods on the Internet. In such an auction, the buyers must know the various features of the products and how to use them, while the Sellers only display their product configuration. Such a system reduces the consultation time between several manufacturers, as well as the design time and cost calculation time.

The study of Woosun Shin [22] presented an interesting algorithm that could suggest a new bidding price by statistically analyzing auction records. The auction agent takes the place of the auctioneer and suggests a proper bidding price using the knowledge of past experiences. The agent offers a good contract price to the buyer when the auction ends.

In considering the above studies, we used an agent-based approach for our implementation. We used three kinds of agents – PurchaserAgent, SellerAgent and FacilitatorAgent. The SellerAgent provides the
function of registering goods for an auction to the sellers. This design maximizes the probability that the product auctioned will sell. The second agent is the PurchaserAgent that requires bidding to buy and it suggests a proper bidding price by analyzing the bidding history of the bidding competitor. The third agent is the FacilitatorAgent that plays the role of an auctioneer and enables a bidder to look at the other person’s auction history while bidding for and buying a product.

3. System Analysis and Design

A detailed empirical study based on the above stated research and analysis is presented. The system was designed keeping in mind the requirements of the industrial partner. The system design and the algorithms employed are described in this section.

3.1 Scenario-based specification

Scenario-based specifications allow an intuitive way of visualizing, understanding and analyzing the system design requirements. The scenario we devised is depicted in Figure 3 and was used to analyze the requirements.

A concise description of the scenario based specification follows: The first step is for the Sellers to register their products/goods with the auction system. The Purchaser obtains the necessary information by chatting. After the purchaser has chosen the products, he participates in a negotiation, and chooses the method of bidding (Speed, Medium, Leisure). The purchasers bid for products for sale and the auction agents assist by recommending a bid price. All actions in the auction are saved in a database. In addition, the purchasers can see the bidding history of other bidders. Finally, when the auction period is over the auction ends and the highest bidder buys the product.

3.2 Designing with UML notations

UML currently has as many as eleven different and partly overlapping notations which constitute different views of UML design. The auction system was designed using the Use Case Diagram, Sequence Diagram, Class Diagram, and Component Diagram offered by UML and the Rational Rose tool.
3.2.1 Use Case Diagram

The Use Case Diagram is a visualization of a use-case, that is, the interaction between the auction system and the users. A single Use Case represents a single aspect of the system.

Figure 4 is the Use Case Diagram for the actions that the Users (Seller, Purchaser) can perform in an auction. Users, after Login, can select the method of auction (auction, reverse auction) and the method of bidding (Speed, Medium, Leisure). They can also express opinions about products or exchange information about products. Also, Sellers can register their products while Purchasers can bid for a particular product. Users can also view the auction history from the database. Thus, this Use Case Diagram defines a series of actions, which Actors (Seller, Purchaser) can perform.

Figure 5 is the Use Case Diagram that represents the Purchaser’s behavior. It defines the behavior of the purchaser while participating in an auction after login. Before a Purchaser can buy goods or start chatting and also during login, the user needs to be validated, as depicted by the inclusion of “Validate User” in each of the aforementioned three cases. The Use Case of Chatting also functions to “record chatting,” and to store the contents of the auction in Data Base. The Purchaser can also select the method of bidding and the method of auction.

3.2.2 Class Diagram

The Class Diagram is the most important entity in Object-oriented analysis and design. It describes the types of objects that exist in the system and shows the static relationships among internal classes of the system. There are two principal kinds of static relationships: associations and subtypes. The Class Diagram can be used to show the attributes and the operations of a class and also the constraints that apply to the way the objects are connected. The Class Diagram can also show inheritance or multiplicity within a graph.
Figure 6 displays the Class Diagram for the Auction System. An Abstract Class (e.g. user class) abstracts common characteristics (Attribute, Operation; Name, Address, Telephone Number, and so on) about an Actor. We use the Concrete Classes (Purchaser, Seller) by inheriting attributes and operations from the Abstract Class user. The Class Diagram allows one to define such visual inheritance relationships between classes (i.e. super class, subclass), and shows the relationship with a graph. Due to these reasons, the graphical design method using UML can reduce implementation and maintenance hours of systems as compared to structured design methodology (function-oriented) and information engineering methodology (data-oriented).

3.2.3 Sequence Diagram

The Sequence Diagram depicts the overall flow of control in an object-oriented program. The top of the diagram shows the objects as boxes and the functions that define the behavior of the objects that belong to each class. The messages sent between objects for dynamic communication are represented as arrows from one object’s lifeline to another. Time progresses downward and the sequence of interaction between different objects by passing of messages is shown. Thus, the Sequence Diagram represents the interactive relationship among objects visually and sequentially.

Typically, the sequence diagram captures the behavior of a single use-case. Figure 7 shows the Sequence Diagram for the Purchaser in the Auction System. The Purchaser logs into the Auction Mall with ID and Password to purchase products. If the user
is successfully certified, the auction system displays the products on auction to the user. Next, if the Purchaser selects some products and enters the Negotiation room for buying, he can select the bidding method to purchase the products. Further, the Purchaser decides the bidding price by viewing the history of the other bidder’s previous auction records and the price that is suggested by the selected bidding method. Finally, the Purchaser that bids the highest price buys the products and the auction result is stored in the database. After that, the auction manager informs all bidders of the auction result and concludes the product’s auction.

3.2.4 Component Diagram

The Component Diagram is a simple high-level diagram that is used to show how the code is actually divided between the different modules. It shows the dependencies among the software components. Figure 8 illustrates the interactive relationship among the software components of the auction system. In our implementation, we made the entities that are used in the auction as components using EJB (Enterprise Java Beans). It is clear from the diagram that it is possible to modify relevant parts/components without affecting the entire system. The configuration of the whole system is easily understood looking at the Component Diagram because it graphically depicts the interrelationships and dependencies (if any) between the various software components.

3.3 Components and Algorithms

The different components and the algorithms used in the auction system are detailed in this section. The algorithm of primary concern here is the one that the PurchaserAgent uses to calculate the bidding price to be suggested to the purchaser.

3.3.1 Component Descriptions

The software components that are a part of the auction system are described. Figure 8 shows all the components mentioned herein.

- The Certification component is used to validate the user trying to log into the system.
- A seller enters products into the system by using the RegistrationGood component. At this time, the seller inputs an end date and time of auction, including the starting and end prices of products.
- Purchaser and Seller components manage information related to the auctions of the purchaser and the
seller, as well as their private information.

- The Negotiation component manages the auction. If a bidder arrives at the time of the auction close or a bidder who suggests the highest price exists, the auction will be closed. When an auction closes, the data record of the auction transfers to the ManagementHistoryAuction component.

- The ManagementHistoryAuction component shows the previous auction record of the auctioneer conducting the current auction.

- The DataBase component saves the relevant data pertaining to the current auction (e.g. the price of products and contents) separately in the database.

- According to the three kinds of bidding methods (Speed, Medium, Leisure), a purchaser decides the next bid after confirmation of the end price that has been suggested so far from the DataBase component using the MethodofBidding component.

Of the eight components developed, the Certification, RegistrationGood and MethodofBidding components are particularly useful and can be easily adapted for reuse in other systems.

### 3.3.2 Calculating Bidding Price

This subsection outlines the algorithm used by the PurchaserAgent to suggest a bid price to a purchaser that would maximize his chances of making a successful bid. There are three possible rates at which a purchaser might choose to bid – Speed, Medium or Leisure. If bidders want to bid at a higher frequency than the other bidders they use “Speed” Alternatively if the bidders choose to bid at the same bidding frequency as the other bidders, they use “Medium” while bidders who use “Leisure” are choosing to bid at a slower frequency.

Figure 9 illustrates the formulae used to arrive at the bid price. K is the amount of money that can be bid at the present auction’s starting price. Equation 1 calculates the difference between the highest forecast-price of products (HP) and the seller’s suggested starting price (SP). PABC is the average amount of the previous total bidding prices that the bidder has bid. Equation 2 calculates the ratio of the difference between the ending price of the product in the previous bidding (PEP) and the starting price of the product in the previous bidding (PSP) with the difference between the ending price of the product in the previous bidding (PEP) and the average bidding price of products in previous bidding (PB). Equations 3 through 5 are the prices that are suggested eventually depending on the respective method of bidding chosen.

<table>
<thead>
<tr>
<th>Equation</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>K = (HP – SP)</td>
<td>(1)</td>
</tr>
<tr>
<td>PABC = (PEP – PSP) / (PEP – PB)</td>
<td>(2)</td>
</tr>
<tr>
<td>Speed = (K/PABC)*1.2</td>
<td>(3)</td>
</tr>
<tr>
<td>Medium = (K/PABC)*1</td>
<td>(4)</td>
</tr>
<tr>
<td>Leisure = (K/PABC)*0.8</td>
<td>(5)</td>
</tr>
</tbody>
</table>

Figure 9. Bidding Method Formulae
4. System Implementation and Results

This section lays out the artifacts of our study as well as the implementation which followed the system analysis and design. The system configuration and implementation environment, screen shots and an analysis of the result of the empirical study are presented.

4.1 System Configuration and Implementation Environment

Figure 10 shows the configuration of the implemented system. The Seller and the Purchaser (who connect through their respective web browsers) are connected to the Auction Web Server. The Auction Web Server communicates with the Chatting Auction Daemon through JAVA/Servile/EJB and with the Database through JDBC. The developed auction system is interactive.

The environment used for implementation was a desktop Pentium III 1GHz, with Windows 2000 and Linux 7.0 as Operating System. We also used Tomcat as Servlet Server and Apache as Web Server. Further, we also use HTML, Java, and MySQL software.

4.2 Processing Procedure of system

The Sequence Diagram (Fig. 7) represents several duties in a single class concretely, and these duties define the action of objects, which belong to the class. The implemented auction system can be accessed simultaneously as a dynamic collaboration by several objects on the Internet. The execution of the auction system proceeds as follows.
• Select “Computer” item of left frame after Login.

• If we select “Notebook” on the next screen, the Chatting screen is displayed as shown in Figure 11.

• If we select a part applicable in “Speed or Medium or Leisure” during the auction, and click the “Show Price of Bidding” button, it displays the next bid price.

• When we want to see the auction records of the other bidders during the auction, we can choose the auctioneers and click the button “Show History of Auction” and the auction history for that particular bidder is displayed (see Figure 12).

• Once the last successful bidder has been decided, the contract price is displayed on the screen and the auction is finished.

• The record of the conversation, which occurs between the bidder and seller or the system, is stored in the Database. Also, all records of the conversation, which occur during the course of the auction, are added in the record for the people who participated in the auction, stored in the Database.

4.3 Analysis of Results

We successfully implemented the web-based auction system using UML and components. The rigorous design and analysis phase and the robust component-based implementation enabled us to achieve a minimal defect rate in the final product. The defect rate of our reused code was 0.9 units per 1KLOC (0.9/1KLOC). In comparison, the defect rate of a new software system developed by Lim [23] was 4.1 per 1KLOC (4.1/1KLOC).

The scope of implementation and identification of entities that could be coded as reusable components was done with the help of UML. Further, because the system was designed using UML, any additions/modifications to the system design was easily facilitated. From the eight components we developed – (1) Certification component, (2) RegistrationGood component and, (3) MethodofBidding component – are easily reusable with little or no modification necessary. The Certification component that was used to certify if a user is a registered user will find wide-use applicability. The RegistrationGood and MethodofBidding components can be easily modified to include different attributes for registering a product or for different frequencies of bidding respectively, if required. Thus, the use of component-based
programming improved the maintainability and reusability of the system.

5. Conclusions and Future Work

This paper described a case study highlighting the best practices in designing and building a web-based auction system. We designed the auction system using UML. The Use Case Diagram, Sequence Diagram, Class Diagram and Component Diagram offered by UML were used successfully during the process. Rational Rose, used for the purpose, provided adequate support. Our implementation, with its basis in component-based programming enabled us to develop a highly maintainable system with a number of reusable components. Further, the system used intelligent agents that permitted fair help to bidders participating in auctions, and at the same time, achieved maximum profit for the seller. Again, the implementation environment and the tools used, provided excellent support for the successful development of the system.

The approach outlined here was more effective in implementing our auction system than the existing Information Engineering (data-oriented), Structured Development (function-oriented), or Object-oriented (data-oriented and function-oriented) methodology. Although we only made a few specific changes to the components, these changes indicate that subsequent changes to other system components will be straightforward. Consequently, the reusability of the system was facilitated and, as a direct result, we expect that the system will be easily able to suitably evolve in the fast changing Internet environment.

Our plans for future work include the (re)implementation of the auction system using Object-Z [24] for formal specification and mathematical algorithms. Object-Z is an object-oriented formal specification language that is based on Z, and has been developed at the University of Queensland, Australia. Extensions to the semantics of Z include implicit support for object identity, together with notions of inheritance and polymorphism. Further, we plan to develop additional algorithms that can be used for analyzing the other competitive bidder’s expectation price. We then plan to compare the results, in terms of defect rate and degree of maintainability and reusability achieved, from these two different approaches.

References


