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

Frederick T. Sheldon and Kshamta Jerath
Software Engrng. for Dependable Systems Lab
Sch. of Electrical Engrng. & Computer Science
Washington State University, Pullman, USA.
sheldon@acm.org | kjerath@eecs.wsu.edu

Young-Jik Kwon and Young-Wook Baik
School of Computer and Information Engineering
Daegu University, Korea.
yjkwon@eecs.wsu.edu | ywbaik@daegu.ac.kr

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 RegistrationGood (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.

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]. 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 be obtained conveniently via online auction systems. Online auctions are a major component of the electronic marketplace that makes use of electronic commerce mechanisms.

In this case study, we undertake the task of designing and developing a Web-based auction system that satisfies the requirements of ease of use, adaptability to changing requirements, maintainable and has scope for reusability. 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. Component-based programming also 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.

In this paper, 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.

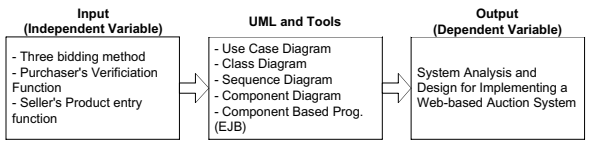

Figure 1a. Process for analysis and design

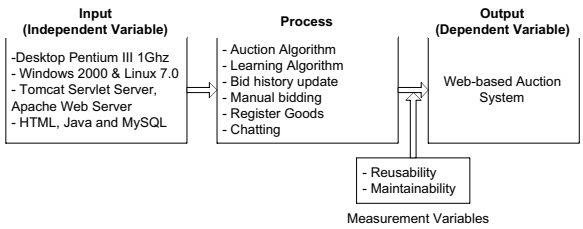

Figure 1b. Implementation and test of the web-based auction system

2. Background and related research

Developing an online auction system requires making decisions and selecting technologies to support those decisions. Some background information and related research on the technologies that we employed are presented here.

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. 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]. UML has been used successfully in numerous projects to model varying architectures and requirements. [8-13].

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.

Components are highly reusable units of functionality and they let developers conceptualize software as inter-connectable blocks [14]. Research topics in this area include design of component integration frameworks to prescribe an architecture that permits flexible composition of third-party components into applications [15], web design reuse vis-à-vis code reuse [16], and reusability of components for quality software [17].

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.

Previous studies on auction systems include those by Jane Hillston and Leila Kloul, Martin Bichler, Kao et. al. Sandholm, Baik and Kim, Wooksun Shin [18]~[23].

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 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.

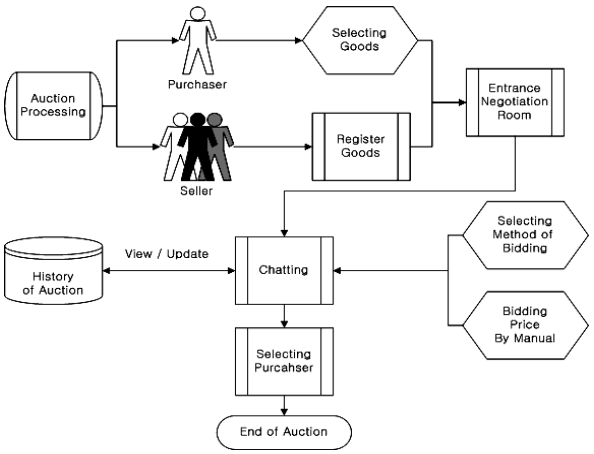

Figure 2. Scenario of auction process

3.1. Scenario-based specification

Scenario-based specifications allow an intuitive way of visualizing, understanding and analyzing the system design requirements. The scenario was devised (depicted in Figure 2) and used to analyze the requirements.

3.2. Designing with UML notations

The auction system was designed using the Use Case, Sequence, Class and the Component Diagrams 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, i.e., the interaction between the auction system and the users. Figure 3 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. Figure 4 is the Use Case

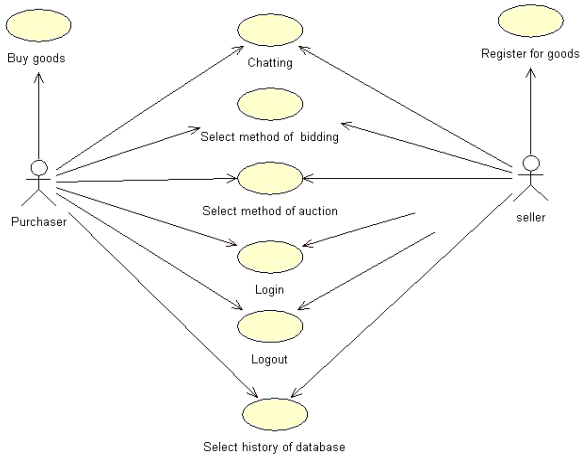

Figure 3. Use case diagram of auction system

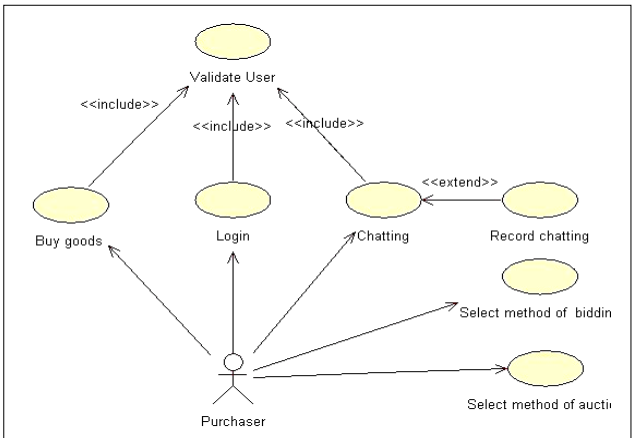

Figure 4. Purchaser's use case diagram

Diagram that represents the Purchaser's behavior. It defines the behavior of the purchaser while participating in an auction after login.

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. 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.

Figure 5 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.

3.2.3. Sequence diagram. The Sequence Diagram displays the overall flow of control in an object-oriented program. Typically, it captures the behavior of a single use-case. Figure 6 shows the Sequence Diagram for the

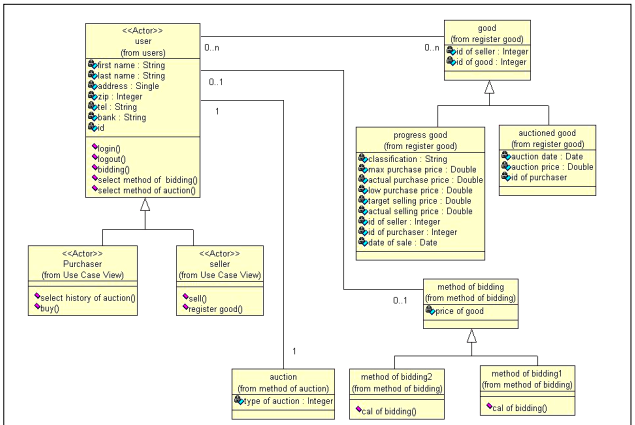

Figure 5. Class diagram of system

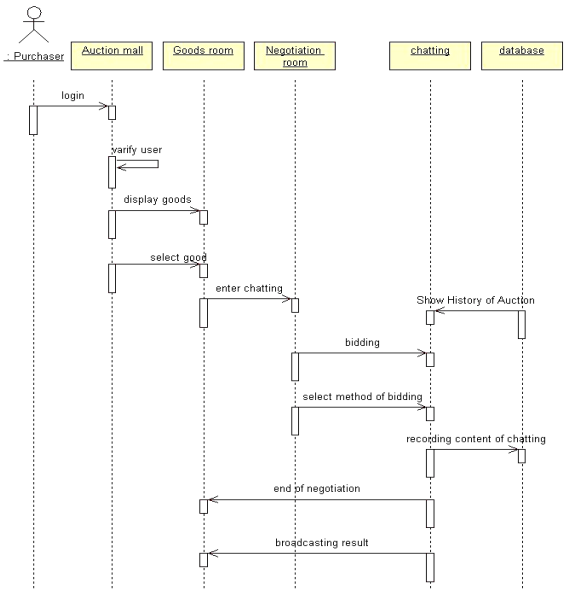

Figure 6. Sequence diagram for purchaser.

Purchaser in the Auction System.

3.2.4. Component diagram. Figure 7 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.

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 7 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.

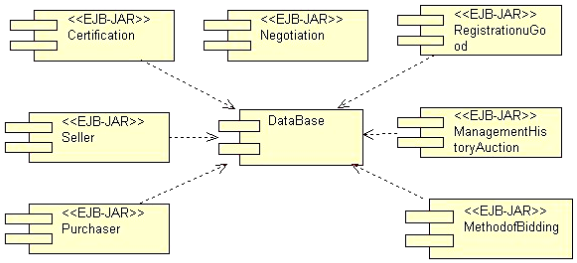

Figure 7. Component diagram of system

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 may choose to bid – Speed, medium or Leisure.

Figure 8 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

$$K = (HP - SP) \quad (1)$$
$$PABC = (PEP - PSP) / (PEP - PB) \quad (2)$$
$$Speed = (K/PABC)*1.2 \quad (3)$$
$$Medium = (K/PABC)*1 \quad (4)$$
$$Leisure = (K/PABC)*0.8 \quad (5)$$

Figure 8. Bidding method formulae

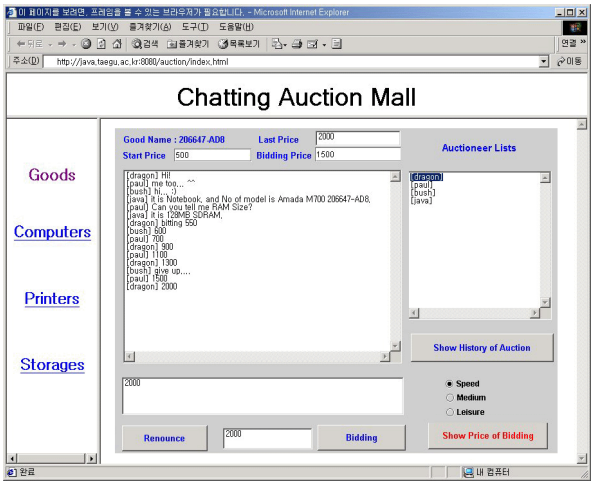

Figure 9. Chatting screen of auction system

between PEP and the average bidding price of products in the previous bidding (PB). Equations 3 through 5 are the prices that are suggested eventually depending on the respective method of bidding chosen.

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 details of system configuration and analysis of the result of the empirical study are presented.

4.1. Configuration / implementation environment

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/Servlet/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 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 on left frame after Login.
- If we select “Notebook” on the next screen, the Chatting screen is displayed as shown in Figure 9.
- If we select a part applicable in “Speed or Medium or Leisure” during the auction, and click on “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 auctioneer and click the button “Show History of Auction” and the auction history for that particular bidder is displayed.
- 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 [24] 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, (3) MethodofBidding component and, (4) Purchaser component – are all 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. The Purchaser component suggests an estimated amount to bidders by learning the bidding record which the other bidders have suggested before. Moreover, it can suggest better estimated bidder price by using experiments that are accumulated like this. 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 [25] for formal specification and mathematical algorithms like VCG (Vickrey-Clarke-Grove, Multidimension). 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.

6. References

1. Tsvetovat, M., et al. *Customer Coalitions in the Electronic Marketplace*. in *4th Int'l. Conference on Autonomous Agents*. 2000. Barcelona, Spain: ACM Press. p. 263-264.
2. Gottesdiener, E., *OO Methodologies: Process & Product Patterns*, in *Component Strategies*. 1998, SIGS Publications: New Albany, IN. p. 34-60.
3. Gomaa, H., *Designing Concurrent, Distributed and Real-Time Applications with UML*. 2000: Addison-Wesley. 816.
4. Mylopoulos, J., *Exploring Alternatives during Requirements Analysis.* IEEE Software, 2001. **18**(1): p. 92-96.
5. Post, G. and A. Kagan, *OO-CASE tools: an evaluation of Rose.* Information and Software Technology, 2000. **42**(6): p. 383-388.
6. Quatrani, T., *Visual Modeling with Rational Rose and UML*. 2 ed. 1998, Boston, MA: Addison Wesley. 288.
7. Breu, R., et al. *Systems, View and Models of UML*. in *The Unified Modeling Language -- Technical Aspects and Applications*. 1998: Physica Verlag. p. 101-102.
8. Wolf, M., R. Burkhardt, and I. Philippow. *Software Engineering Process with the UML*. in *The Unified Modeling Language -- Technical Aspects and Applications*. 1998: Physica-Verlag. p. 271-280.
9. Kivisto, K. *Considerations of and Suggestions for a UML-Specific Process Model*. in *UML'98: Beyond the Notation*. 1998. Mulhouse, France: Springer Verlag. p. 294-306.
10. Allen, P. *A Practical Framework for Applying UML*. in *UML'98: Beyond the Notation*. 1998. Mulhouse, France: Springer Verlag. p. 419-433.
11. Back, R.J., L. Petre, and I.P. Paltor. *Analysing UML Use Cases as Contracts*. in *UML'99: Beyond the Standard*. 1999. Fort Collins, CO: Springer Verlag. 1723. p. 518-533.
12. Gomaa, H. *Object Oriented Analysis and Modeling for Families of Systems with UML*. in *6th International Conference on Software Reuse*. 2000. Vienna, Austria: Springer Verlag. p. 89-99.
13. *Cardoso, J and C. Sibertin-Blanc. Ordering action in Sequence Diagrams of UML.* in *Proceedings of the 23rd International Conference on Information Technology Interfaces*, 2001. p. 3-14.
14. Repenning, A., et al., *Using Components for Rapid Distributed Software Development.* IEEE Software, 2001. **18**(2): p. 38-45.
15. Sousa, J.P. and D. Garlan. *Formal Modeling of the Enterprise JavaBeans Component Integration Framework*. in *Proceedings of FM'99*. 1999. Toulouse, France: Springer Verlag. 1709. p. 1281-1300.
16. Schwabe, D., *Engineering Web Applications for Reuse.* IEEE Multimedia, 2001. **8**(1): p. 20-31.
17. Forsell, M., V. Halttunen, and J.J. Ahonen. *Use and Identification of Components in Component-Based Software Development Methods*. in *6th International Conference on Software Reuse*. 2000. Vienna, Austria: Springer Verlag. 1844. p. 284-301.
18. Hillston, J. and L. Kloul, *Performance investigation of an on-line auction system.* Concurrency and Computation: Practice and Experience, 2001. **13**(1): p. 23-41.
19. Bichler, M., *An Experimental Analysis of Multi-Attribute Auctions.* Decision Support Systems, 2000. **29**(3): p. 249-268.
20. Kao, M.Y., J. Qi, and L. Tan, *Optimal Bidding Algorithms Against Cheating in Multiple-Object Auctions.* SIAM Journal on Computing, 1999. **28**(3): p. 955-969.
21. Sandholm, T., *Approaches to winner determination in combinatorial auctions.* Decision Support Systems, 2000. **28**(1-2): p. 165-176.
22. Baik, Y. and J. Kim. *Implementation of Interactive Online Auction System*. in *KFIS Fall Conference*. 2000. Korea. 10. p. 405-408.
23. Shin, W., *The Study on Bid Algorithm of Auction Agent using Statistical Method*, in *Dept. of Computer Information and Communication*. 2000, Konkuk University: Seoul, Korea.
24. Lim, W.C., *Effects of Reuse on Quality, Productivity and Economics.* IEEE Software, 1994. **11**(5): p. 23-20.
25. Duke, R., G. Rose, and G. Smith, *Object-Z: A Specification Language Advocated for the Description of Standards, Technical Report 94-95*. 1994, The University of Queensland: Australia.