An Agent-Mediated E-Commerce Environment for the Mobile Shopper

Steven Fonseca, Martin Griss, Reed Letsinger
Software Technology Laboratory
HP Laboratories Palo Alto
HPL-2001-157
June 21st, 2001*

E-mail: steven_fonseca@hp.com, martin_griss@hp.com, reed_letsinger@hp.com

E-commerce, agents, appliances, Zeus

Agents and mobile appliances offer the promise to change the way people purchase products by connecting the physical presence of stores with their Internet representation and delegating consumer tasks to intelligent pieces of autonomous software. The archetypical example is the shopping mall of the future; shoppers use personal digital assistants (PDA) to interact with store services while in the mall, on the way to the store, or in the store itself. Mall-wide services, such as directories and locators are also available through the PDA. Intelligent agents represent both shoppers and the store to negotiate for desired products based on shopper preferences. A lightweight version of this scenario was implemented with a multi-agent system framework (Zeus), Jornadas, CoolTown infrared beacons, and additional support software. Experience suggests that while the scenario is compelling, current agent building tools are insufficient for rapidly prototyping multi-agent societies and a more sophisticated means of staying connected to the virtual world is needed.

* Internal Accession Date Only
Approved for External Publication

An Agent-Mediated E-Commerce Environment for the Mobile Shopper

Steven Fonseca, Martin Griss, Reed Letsinger
Software Technology Laboratory
HP Laboratories Palo Alto

E-mail: steven_fonseca@hp.com
martin_griss@hp.com
reed_letsinger@hp.com

Keywords: E-commerce, agents, appliances, Zeus

Abstract

Agents and mobile appliances offer the promise to change the way people purchase products by connecting the physical presence of stores with their Internet representation and delegating consumer tasks to intelligent pieces of autonomous software. The archetypical example is the shopping mall of the future; shoppers use personal digital assistants (PDA) to interact with store services while in the mall, on the way to the store, or in the store itself. Mall-wide services, such as directories and locators are also available through the PDA. Intelligent agents represent both shoppers and the store to negotiate for desired products based on shopper preferences. A lightweight version of this scenario was implemented with a multi-agent system framework (Zeus), Jornadas, CoolTown infrared beacons, and additional support software. Experience suggests that while the scenario is compelling, current agent building tools are insufficient for rapidly prototyping multi-agent societies and a more sophisticated means of staying connected to the virtual world is needed.

1 Introduction

A vision of the not so distant future serves as the impetus for research activities collectively known as the Agora project. The fundamental assumptions are that:

1) The distinction between the physical world and the virtual world (Internet) will be blurred as electronic presence for people, places, and things [1] becomes commonplace.
2) Accessing the virtual world from a desktop or laptop computer will partially give way to access via a variety of mobile appliances such as the PDA's of today.
3) User context and preferences are vital considerations for delivery of electronic content, services, and support.
4) The distributed nature of the virtual world requires software to operate in a heterogeneous and dynamic environment composed of many components that must communicate with one another [3].
5) Software agents promise to offer the programming paradigm necessary to enable intelligent, dynamic, and heterogeneous component interaction.

6) Future generations of e-commerce should employ agent technology [7] to realize and benefit from dynamic pricing, access to worldwide markets, and flexible partnership formation [4].

Given this vision, our research group at HP laboratories investigated and integrated three broad research areas during the summer 2000: Mobile appliance infrastructure; user context and preferences; and software agents for electronic commerce.

An e-commerce, mobile appliance testbed was created to study these topics individually while also looking at their integration. The work began by developing general scenarios outlining a vision of how users could benefit from emerging mobile appliance and software agent technology. Two scenarios are presented in this introduction after which follows a description of the relevant technologies, specific research questions considered, and the value of this work to Hewlett-Packard. In section two the Agora project vision is shared. The original project motivations and objectives are discussed. Included in this section is a high-level description of the implemented shopping mall scenario. Section three includes a step-by-step description of all user and agent interaction. The technical details of this interaction are discussed further in Section four. Section five concludes with a lessons learned summary.

1.1 Motivating Visions

- People, Places, and Things

The Cooltown project [5] provides one of two visions motivating the Agora project. It is believed that given the maturation of Internet protocols, wireless networking, and mobile electronics, future generations of people will want and can benefit from remaining regularly connected to the World Wide Web. Further, the relationship between the physical world and virtual world will be melded together by these enabling technologies. First, all people, places, and things will have a virtual presence (an online representation). Because people have simultaneous access to both the physical entity and its virtual presence, new opportunities for interaction become possible and desirable. Consider a bookstore with customers that are physically present also having access to the store's virtual representation. "If a book is out of stock, it can be ordered and shipped immediately without waiting in line for a store clerk to help. It can give suggestions of related books depending on the section of the bookstore where the user is physically" [5]. In this scenario, the shopping experience of a customer is improved by using a web-enabled appliance, wireless network, Internet protocols, and by utilizing contextual information provided by physical location.

- Agent Based Electronic Commerce

There are currently only a small number of distinct examples of agent technology being applied to the electronic commerce domain, primarily price comparisons and aggressive buying agents (pricebots and shopbots). What is more common is the tedious and time-consuming process that consumers go through when shopping over the Internet. The process might start with a search for a particular product from which several links to stores are returned. The consumer typically must visit each website to check pricing and additional terms. This could involve

considering alternate products from an online catalog, product availability, delivery options, return policy, or payment methods. After all relevant information is gathered the consumer then purchases the product using a credit card.

In the future, it is envisioned that software agents can be made responsible for autonomously mediating purchases for both businesses and consumers. An agent can handle all of the information gathering, decision-making, and payment processes. More importantly, "Companies will have instant access to unbounded, world-wide markets; prices and product packaging can be determined dynamically through negotiation on a per transaction basis; many short-lived, task-specific collaborations between companies will replace the more expensive, long-lasting partnerships and contracts common today" [4].

1.2 Project Elements

The Agora (a Grecian marketplace) project was rich with opportunities to learn and evaluate existing technologies and standards, gain experience with agents and appliances in the electronic commerce domain, and study the concept and implications of context-aware computing. Six project elements summarizing the main topics of research are shared and their scope for the Agora project is defined.

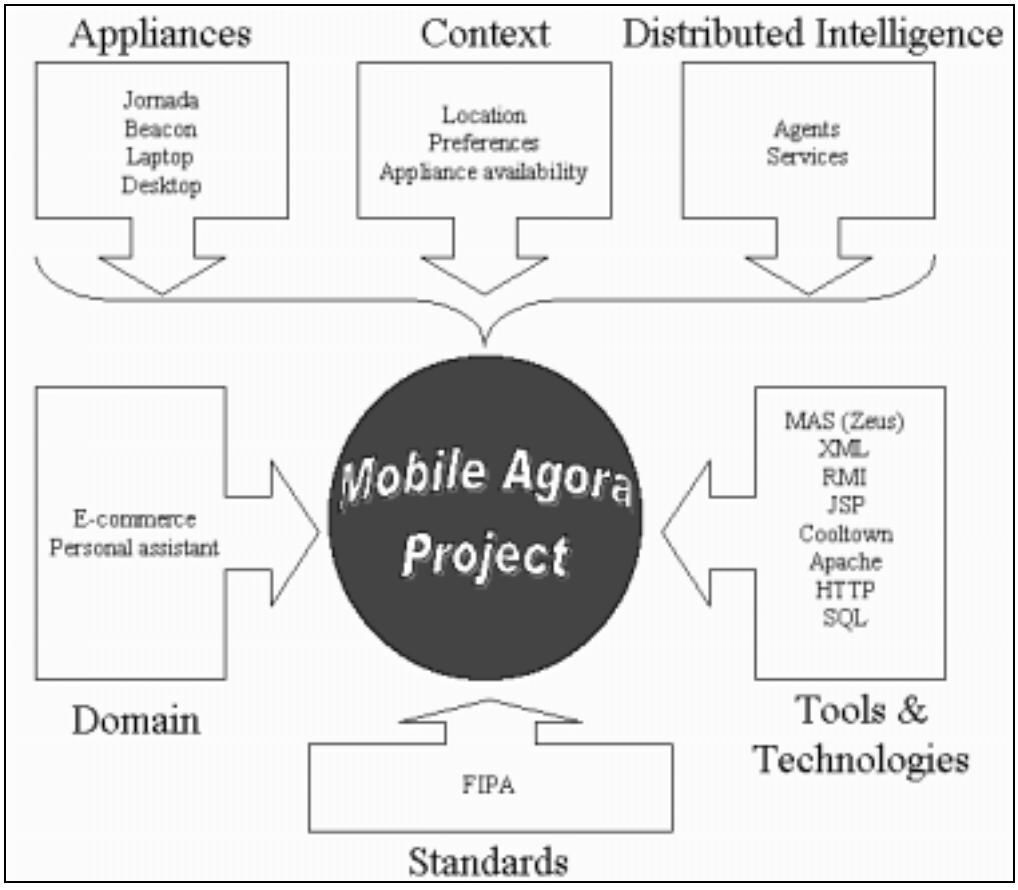

Figure 1: Agora Project Elements

- Domain

The application of software agent technology to the electronic commerce and personal assistant domains serve as the backdrop for the Agora project. Characteristic of agents in the electronic commerce domain are their understanding of commerce related concepts including buying and selling, ability to negotiate, and responsibility for handling payment and goods exchange. The focus of agents in the personal assistant domain is to help their owner achieve a task by collaboratively interfacing directly with their owner and considering their owner's personal preferences. A more sophisticated personal assistants also has the ability to learn from interactions with their owner thereby offering better support over time. The Agora project operated at the intersection of these two domains. An electronic commerce agent was developed that also took user preferences into account while providing a web-based interface with which to control the agent.

- Context

Context is an important consideration that should serve as an input to the decision making process of an agent. Contextual information can be extracted from the environment of the agent (virtual context) or the user's immediate physical surroundings (physical context). Virtual contextual elements might include the availability of services or infrastructure agents while user location or hardware availability are examples of physical contextual elements. Additionally, user preferences also help establish what might be collectively called the operating context of the agent and user. A simplified operating context was established for the Agora project that included virtual and physical contextual elements. To a more limited extent, user preferences were also considered.

- Distributed Intelligence

A main objective of the Agora project was to gain experience creating software agents for electronic commerce. Agents were created for this domain and a personal assistant agent was also created. These agents and services were deployed on multiple hosts within the Hewlett Packard Intranet. A question discussed was the difference between an agent and a service. In our mall game the distinction was easy. If an object descended from one of the Zeus agent classes then it was an agent. A direction lookup capability was implemented both as an agent and as a service. The service had an object-oriented interface; the object's methods were invoked to query for and receive directions. More generally, agents are autonomous software components that are personalized to a particular user – while services are web accessible computing resources.

- Appliances

The term appliance is loosely meant to describe user-oriented devices capable of communicating with the surrounding environment. Being mobile and having a built-in ability to connect to a centralized/established communication channel often further characterize appliances. These devices are usually good at providing a few specialized capabilities to the user but often do not offer a comprehensive suite of features. During the Agora project, the limitations of appliances were considered, as were best methods for choosing from multiple appliances to communicate most effectively. The appliances used for the Agora project include a Jornada (palmtop), desktop, laptop, and infrared beacons.

- Standards

The Zeus agent building toolkit, developed by British Telecommunications, was used to implement all of the agents for the Agora project. At the time Zeus was written, the Foundation for Intelligent Physical Agents (FIPA) was beginning to develop specifications for multi-agent system development, agent-to-agent communication, and domain specific application of agents. The Zeus communication subsystem was written to conform to what are presently the ACL Message Structure and Communicate Act Library specifications. The infrastructure agents provided by Zeus speak the FIPA agent communication language (ACL) and parsing classes are available to decode messages following the FIPA performative syntax.

Additional infrastructure and personal agents developed for the Agora project were required to speak the FIPA ACL because they had to interface with the nameserver agent (ANS) and the directory facilitator (DF) provided by the Zeus toolkit. Efforts were also made to conform to the FIPA ACL for conversations between agents written solely for the Agora project.

Further specifications and additions to the FIPA standard were introduced after Zeus was created. Consequently, British Telecommunications could not implement standards defining such things as an abstract agent platform architecture or agent lifecycle.

Zeus was chosen because it was available in Java, opensource, and integrated a complete set of intelligent multi-agent capabilities into a single, uniform package: FIPA agents, rules, facts, protocols, facilitation, etc. Each Zeus agentruns in a single Java virtual machine. Agents communicate via sockets. Agents do not use HTTP or XML and Zeus does not provide a web interface.

- Tools and Technology

The Agora project was rich with opportunities to evaluate current technology and tools for building agents, dynamic and customized display of information over the Internet (web servers and content API's), databases, distributed programming API's, information representation, and communication protocols. One of our goals was to have agents provide web based interfaces, rather than use only local GUI's. The specific tools and technologies used include the Zeus Agent Toolkit [8], XML, RMI, JSP, E-squirt, Apache server, HTTP and SQL. Integrating these technologies for the Agora project proved to be a challenge.

- Software Engineering

Managing the Agora project required incrementally evolving our agent infrastructure by organizing contributions from 13 researchers over the course of a summer. To facilitate development, Visual SourceSafe was used for version control and file sharing, a standard environment was established, and local build and execution scripts were used. A nightly build process supported our goal for incremental development. Using Visual SourceSafe required researchers to communicate such things as code changes, current work in progress, and source

code organization. Missing but needed was an established agent development process and techniques for communicating agent and multi-agent system design.

1.3 Research Questions

Application of agent technology to the personal assistant and e-commerce domains is in its infancy. Learning and research occurred throughout the Agora project at many levels including understanding the application domain, ramping-up on tools and technology, and integrating diverse and distributed software subsystems with corresponding hardware. The questions we initially posed, as well as the questions we encountered during the Agora project, span a variety of topics that are loosely organized below.

- Domain Analysis and Requirements

What capabilities or intelligence is needed to engage in e-commerce or be an effective personal assistant? What are the reusable components for the e-commerce and personal agent domain? What do the e-commerce and personal assistant domains have in common? What are the requirements of a MAS platform that supports building agent societies for multiple domains?

- Agent and Multi-Agent System Architecture

What architectural patterns can be extracted from the open source MAS toolkits? What are the main subsystems that compose a generic agent? What domain specific (e-commerce, personal assistant) subsystems are required? Can agent subsystems be reused through clean and well-defined interfaces? Is the FIPA abstract architecture useful when building "real" agent systems?

- Modeling Agents

How should agent conversations be represented? How can protocols be specified? At what level must individual messages be specified and how should this be done? How are agent society conventions communicated? What other agent or MAS characteristics (ontology, agent relationships, etc.) require modeling and how can they be represented? Is AUML expressive enough to capture key agent characteristics? What design diagrams are needed to facilitate the implementation and documentation of a MAS?

- MAS API Design

Can the programming abstraction level be raised from object-oriented to agent-oriented? What does an agent-oriented API look like? Can an API be created to wrap a rule-based execution engine such that "ordinary" programmers can quickly develop agent behavior? What built-in capabilities should a MAS API provide to support protocols and conversation management? Can a multi-level API be written that enables programmers to program high-level agent behavior that may be less flexible while also providing an API at a lower level of abstraction when agent behavior requires more customization?

- Agent Engineering (agent building tools)

How can a toolkit be built that actually allows agents to be rapidly prototyped? Is it possible to develop a code generation tool that takes a conversation model and

outputs MAS platform specific code? What tools are necessary to provide adequate debugging of a MAS platform and visualization of agent interactions?

- Agent and MAS Development Processes

Are current OO-centric development processes applicable to the construction of agents and agent societies?

- Agents and Appliances

How can agent technology be integrated with mobile appliance technology? How easy is it for agents to "fit into" mobile appliances with limited processing power?

- Agent Standards

Do current MAS platforms built following FIPA standards provide developers with a means of rapidly prototyping multi-agent societies? Given that FIPA standards are not primarily derived by analyzing implemented MAS platforms, in what ways will real experience influence future versions of the standards? Does being FIPA compliant overly burden programmers for domains where agent interaction is not sophisticated? Similarly, how can a FIPA compliant message be efficiently constructed? Should other competing standards bodies exist to provide a checks and balances mechanism that allows for more critical evaluation of agent standards development?

- Agent Roles (infrastructure and participatory agents)

What infrastructure agents must be present for any agent society? What infrastructure agents are required for the electronic commerce domain? What are the advantages and disadvantages of having a single agent with a comprehensive set of behaviors and conversely, having a single agent serve as a top-level coordinator of a group of primitive (with only a few behaviors) agents? What common roles are participatory agents likely to play? How can agent roles be dynamically loaded at runtime?

- Agent Communication

What agent communication languages (ACL) will facilitate efficient communication? How successful are current agent communication languages such as FIPA, KQML, or FLBC? What common concepts do these languages share and would refactoring this commonality be useful? How can communication protocols be reused? What are the best methods for managing conversations? How should ontologies be represented, managed, and translated?

- MAS Deployment

How well do MAS societies scale? What technologies must be in place for agents to live in a heterogeneous network of computers and other appliances? What security infrastructure is needed to guarantee a high-level of trust? In what other ways can trust be established? What standards need to be in place for multiple multi-agent societies to interact?

1.4 Value

There was a strong belief in the group that agent technology, with its promise of agents as flexible, autonomous, loosely coupled components would be ideal for

building complex e-commerce systems [6]. This in turn would allow much more customized application systems to "grow" dynamically.

2 Agora Shopping Mall

2.1 Implementation Objectives

In a single sentence, the objective of the Agora project was to build a testbed for studying agents, appliances, and their relationship in the electronic commerce and personal assistant domains. From an implementation standpoint, the main objective was to provide the hardware and software support necessary to integrate appliances and agents into a test platform. This platform, built incrementally, would enable experimenting with agent technology (negotiation, roles, decision making, etc.), agent-based electronic commerce economics (market stability, fairness, efficiency, etc.), mobile appliances, and user context.

Many more requirements for the experimental test platform were pursued including ease of agent configuration, support for agent plug-and-play behavior, a web-based agent interface, dynamic and customized web page display, initialization software, support for visualization of agent interaction, and additional debugging tools necessary for development.

2.2 High-Level Scenario Description

To build the Agora experimental test platform, a shopping experience of the future was developed. The implemented scenario finds a person named Josh arriving at a shopping mall. Here, the stores are physically present, like conventional malls, but they also have a virtual web presence that can facilitate interaction. Josh brought his HP Jornada; a personal digital assistant running Windows CE. It also has a wireless LAN card to connect to the Internet and an infrared port to communicate with other IrDA capable devices that Josh might encounter while shopping, in accord with the CoolTown vission. Josh uses his Jornada to interact with stores in the mall and also to gain access to the services provided by his personal assistant agent. The personal assistant is able to communicate with mall infrastructure agents and other software agents representing the stores.

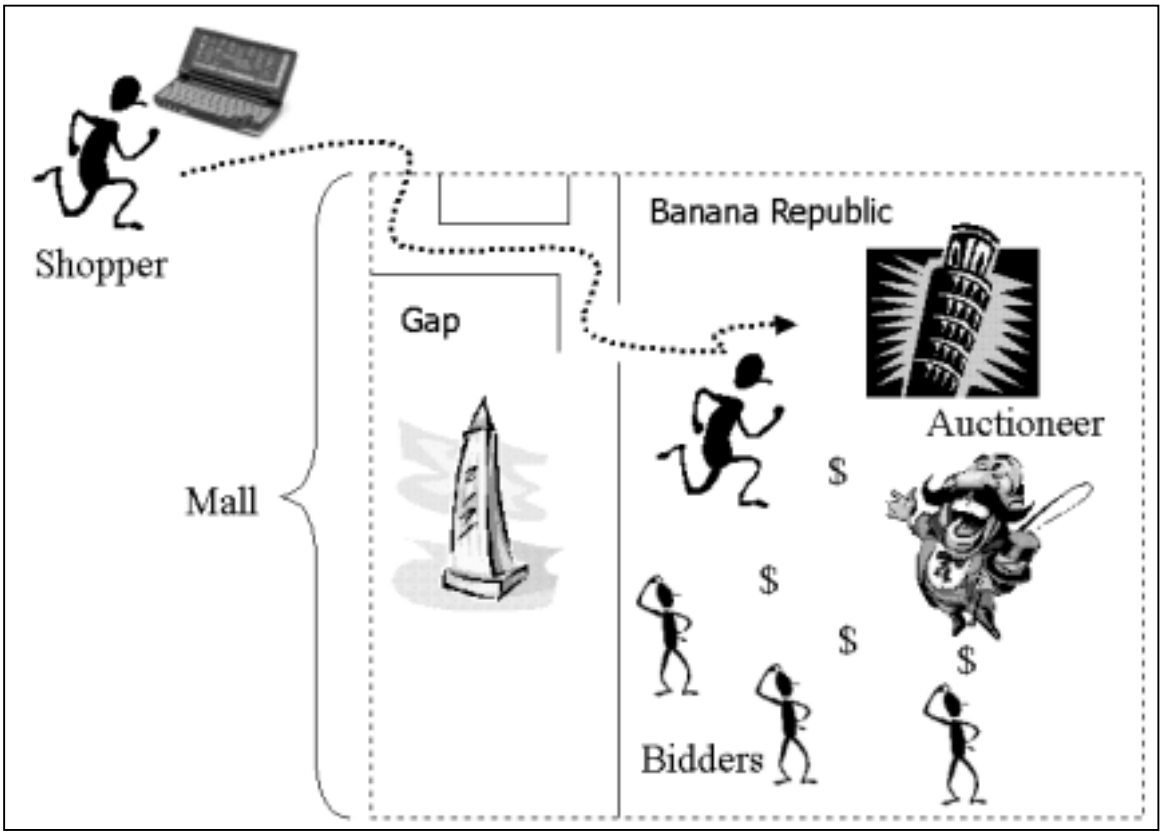

Figure 2: The shopping mall scenario

Josh's objective is to purchase a list of products with the help of his personal assistant.

Josh physically arrives at the mall entrance. The mall entrance, like every other physical location, has a beacon that connects physical locations with their virtual presence. Beacons are infrared devices programmed to broadcast a URL for a site. The mall entrance beacon transmits a URL that is received by Josh's Jornada. The Jornada browser requests the broadcasted URL from the mall web server and a login page is displayed. Josh registers with the mall and then requests a list of stores selling the product he would to purchase. From this list, he then asks the mall direction service to provide directions to the store of his choice. A mall infrastructure agent provides step-by-step instructions to the mall that are displayed for Josh. He consults the textual directions displayed on his Jornada as he walks until reaching the store entrance.

Upon reaching the store, Josh's Jornada receives a new URL broadcast by the store's beacon. The store begins displaying product information. Josh decides to continue shopping using a desktop computer provided by the store to take advantage of the larger display. He peruses the list of products and eventually chooses the products he would like to purchase. The web-based interface allows him to choose the type, quantity, and maximum price he is will to pay for each product. This information is sent to his personal assistant who consults with the mall yellow pages (facilitator) agent to locate store agents.

Instead of being paired with a store agent directly, Josh's personal assistant is given contact information for an English auction agent. This agent holds auctions on behalf of

store agents for the requested products. It allows any number of bidding agents to participate in an English auction of fixed length. Josh's personal assistant assumes the role of a bidder and attempts to win the auction.

If Josh's personal assistant wins the auction, it engages in a fulfillment phase where product and payment are transferred. Once finished, Josh is ready to continue shopping at the same store or may decide to move to another store of his choosing.

3 Agora, The First Iteration

To complement the high-level description of the implemented shopping scenario that was given in the previous section, the step-by-step details of purchasing a product are provided below. It is important to note that initialization of the Agora infrastructure is not described. This is deferred until the next section. The agents that are active when Josh arrives at the mall include a name server (ANS), directory facilitator (DF), direction agent (DA), personal agent (PA), store agent (SA), English auction agent (EAA), and bidder agents (BA). The role and interactions of each agent are described below.

- Mall Arrival, Product Search, Direction Query

The login process that Josh goes through connects him with his PA. The PA holds a shopping list for Josh. The products that need to be purchased are displayed on the screen of Josh's Jornada and he selects one from the list. His choice is passed back to his PA which contacts the DF to determine what stores are selling the selected product. The DF is a mall infrastructure agent that maintains a list of all the products that are for sale in the mall (SA's register products with the DF). The DF returns the list of stores currently selling the product to the PA and a short time later they are viewed on the Jornada. Josh now requests directions to one of the stores selling the product he wants. Two versions of this process were coded. In the demonstration version of the shopping mall scenario, a look-up object returns directions. In the experimental version, the PA requests directions from the DA. The DA returns textual instructions for navigating from any two points in the mall. These directions are displayed on Josh's Jornada and are available while walking to the store. For demonstration purposes, cubicles were assigned store names thus introducing the notion of physical location coupled with virtual representation.

- Store Login, Select Product to Purchase, Register for an English Auction

When Josh reaches the store, he logins to establish contact with the corresponding SA. This is done using a desktop computer provided by the store instead of his Jornada to take advantage of a bigger display. Josh views product information and eventually chooses his terms of purchase. Previous to Josh's arrival, the SA registered to sell his product of choice with the EAA. The EAA's responsibility is to hold English auctions for stores in the mall. When a request to sell a product comes in, the EAA contacts the DF to inform all agents in the mall that a product is being auctioned. Then when a PA makes a purchase request, the DF is queried to see if auctions are currently open. The DF returns the name of the auction to the PA who can then register with the EAA to compete.

- Compete in English Auction, Complete Fulfillment, Continue Shopping

Once an auction is found, the PA registers with the EAA. In the Agora demonstration version, BA's serve as dummy agents that are not owned by anyone but compete with the PA during the auction. The PA is loaded with a bidding English auction protocol and accompanying strategy. The strategy is not interesting as the PA simply bids a single increment higher than the maximum bid until it wins or its maximum bid is reached. The BA's also use the same protocol and strategy. At the conclusion of the auction, which is time bounded, the EAA informs the participants of who has won and lost. If Josh's PA wins the auction, then it proceeds to pay the EAA. The EAA informs the SA of the sell and the SA transfers the product to the EAA. Finally, the EAA gives the SA the money and the PA the product. Josh is then free to continue shopping.

During the English auction, a monitoring interface that displays the current state of the auction for each bidder and a textual listing of the bids that have been submitted. The monitoring interface is provided by the Zeus toolkit and is not web-based.

4 Additional Agora Implementation Details

Before the shopping mall scenario can be executed, several software infrastructure components must be in place. The Agora infrastructure is comprised of four subsystems including an Apache web server, RMI server, SQL database (mySQL), and Zeus utility agents. The order in which these subsystems are brought-up does not matter, however, all must be running before store or personal agents can be instantiated. Interaction between the subsystems appears in Figure 3. Notice that multiple languages (RMI, HTTP, SQL, and ACL) are needed to facilitate communication.

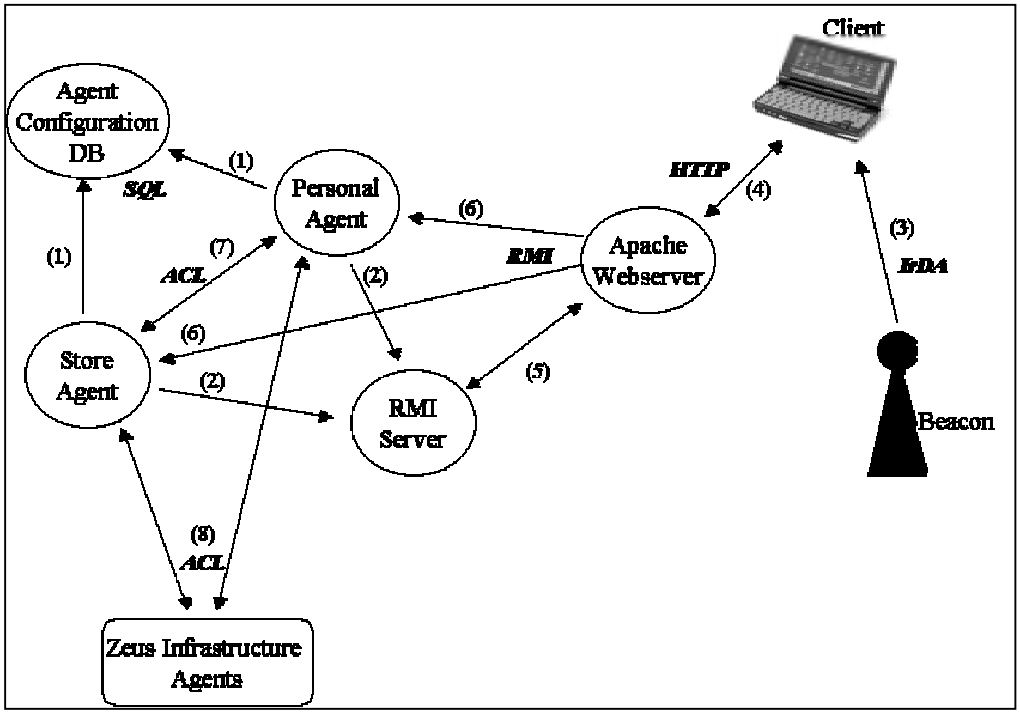

Figure 3: Agora Communication paths

When a store or personal agent is instantiated they query (1) a database that contains configuration and personal preference information. For the PA, the database contains the shopping list that determines what products should be purchased. The store agent retrieves the store inventory from the database. In the demonstration version of Agora, a placeholder for personal preference information exists in the database. The only user information stores was the agent's name for both types of agent. When a personal or store agent is brought to life, they must register (2) with the RMI Server. This allows the web server to communicate (6) with the agents by remotely executing their methods.

After all agents are running, the shopping mall scenario commences. Beacons broadcast (3) XML encoded URL's using an infrared communication. The Jornada runs e-squirt, an application that receives beacon transmissions and then creates a link for the URL. When the shopper clicks on this link, the web client running on the Jornada makes an HTTP request (4) to the web server using its PCMCIA wireless network card. Tomcat extends this server and allows dynamic web content to be displayed using Java Server Page technology.

Because web pages are the primary interface to the store and personal agents, the web server must consult (5) the RMI server to obtain references to them. After these references are obtained, the web server can execute (6) methods provided by the agents. Once the agents have been given a task to perform, they communicate (7) directly with each other using a FIPA compliant ACL message format. The personal and store agents use this same mode of communication to interact (8) with the Zeus Infrastructure (nameserver, facilitator, etc.) agents.

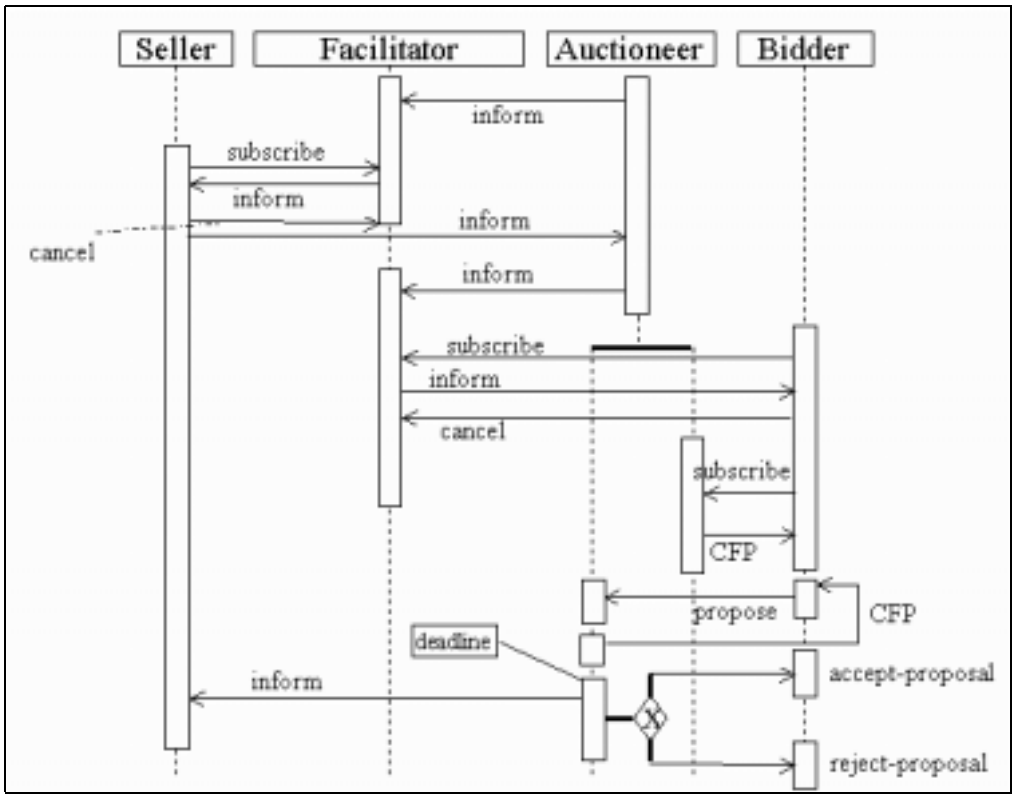

Figure 4: AUML sequence diagram for the Agora English auction[9]

Of all the communication between agents, the most complex series of message exchanges occurs during the English auction. As can be seen by the sequence diagram in Figure 4, message types (inform, subscribe, etc.) are taken from the FIPA Communicate Act Library Specification. The Agora English auction protocol defines four roles that include a seller, facilitator, auctioneer, and bidder. When the auctioneer agent is created it registers with the facilitator to inform the society that it can hold auctions. Then sellers can subscribe to the facilitator for a list of available auctioneers. The facilitator sends the names of available auctioneers to the seller who can then request that a product be auctioned. In response to this request, the auctioneer informs the facilitator that it is selling a product. This initiates the auction. Agents wishing to purchase this product consult the facilitator who informs them of auctions that are currently open. The auctioneer waits for bidder agents to register. Once registered, bids can be placed. Whenever a new high bid is received, the auctioneer informs all registered bidder agents. Bidding continues until a fixed time has passed. At the close of the auction, the auctioneer informs the agents who has won. Though the sequence diagram does not show the payment conversation, the winning bidder and seller engage in this message exchange.

5 Lessons Learned

A valuable lesson learned was that MAS framework development can benefit from the first round of abstractions elicited from the multi-agent system domain. British Telecom's main contributions are the identification of valuable agent concepts and component design for executing agent behavior using a protocol-based paradigm. Most of the Zeus MAS framework requires refactoring. The lesson is that multi-agent system design must follow object-oriented framework design principles if a development environment that offers significant design and code reuse is desired.

Further domain analysis is needed across both the domain of problems that MAS frameworks attempt to solve and the current MAS framework solutions.

The Zeus high-level architecture must be replaced with a flexible alternative that enables agents to be composed of subsystems from potentially different developers. Establishing interfaces among subsystems is a possible solution.

The success of a multi-agent system platform depends on the same factors that make any framework successful. At a minimum, a MAS should provide adequate documentation, a usable API, monitor and debugging tools, capture the essential concepts of the domain, and support points of variability. The struggle to achieve these design criteria will continue until the multi-agent system domain is well understood. Until such time, the iterative and incremental process of refining the domain model and architecture continues. Successful MAS implementations will follow.

Implementing an English auction protocol provided experience with agent communication languages, FIPA standards, conversation management, and modeling agent communication. Writing protocols in Zeus was not straightforward and much effort was spent studying uncommented source code. One of the biggest problems was the low programming abstraction level. This overly burdened programmers with writing code that should have been provided by default from classes in the toolkit. Also, support

for communication protocols and corresponding strategies (algorithmic flexibility points) was incompletely supported. FIPA verbs were used to implement the English auction and were beneficial because of the agreed upon semantics. A standardized method for describing agent conversations would have been beneficial. Though ad hoc notes and diagrams did help facilitate design, many details needed for implementation went unspecified. Finally, better conversation management was needed.

The Agora project served also to assess the current capabilities of the CoolTown technology. Beacon functionality should be dynamic to permit the runtime change of the URL that is broadcast. In some contexts, it seems appropriate to replace beacons with IrDA dongles. Instead of a single URL broadcast, a dongle could first receive information about the user that has just arrived and then transmit a user specific URL that might also depend on the state of the world at that instance in time. The embedded e-squirt software was another limiter of the Agora project. It was desirable, but not possible, to easily incorporate user preference information when selecting web page links for viewing. Ideally, all web content should be tailored to the user. In the case where only one URL is broadcast by a beacon, there is no opportunity to pass user information to the web server. This can be achieved by enabling the e-squirt software to append get or put parameters to the end of the URL that is received.

Acknowledgements

Special thanks to team members David Bell, Harry Chen, Ye Chen, Dick Cowen, Peter Finin, Ed Katz, Thomas Raffill, and Farrell Wymore.

References

[1] Tim Kindberg et al., *People, Places, Things: Web Presence for the Real World*, Hewlett Packard technical report, HPL -2000-16

[2] CoolTown home page, www.cooltown.hp.com

[3] Qiming Chen et al., *Multi-Agent Cooperation, Dynamic Workflow and XML for E-Commerce Automation* in Proc. Autonomous Agents 2000, June, Barcelona

[4] Martin Griss and Reed Letsinger, *Games at Work-Agent-Mediated E-Commerce Simulation* in Proc. Autonomous Agents 2000, June, Barcelona

[5] Deborah Caswell and Philippe Debaty, *Creating Web Representations for Places*, Hewlett Packard technical report, HPL -2000-32

[6] Martin Griss, Agent Components Speed Up Internet Application Development, IEEE Computer, May 2001, 37-43

[7] Patty Maes, R.H. Guttman, and A. Moukas, Agents that Buy and Sell, Commuincations of the ACM, March 1999, 81-91

[8] Nwana Hyacinth, Divine Ndumu, et al., ZEUS: A Tool-Kit for Building Distributed Mutli-Agent Systems, Applied Artificial Intelligence Journal, Vol 13(1), 1999, 129-186

[9] James Odell, H. Van Dyke Parunak, et al. AOIS Workshop at AAAI 2000, July 2000, 3-17