Nonlinear navigation and context-sensitive help in the Sloan Electronic Commerce Project

By

Rong Fong Tian

Submitted to the Department of Electrical Engineering and Computer Science in Partial Fulfillment of the Requirements for the Degrees of Bachelor of Science in Electrical Engineering and Computer Science and Master of Engineering in Electrical Engineering and Computer Science at the Massachusetts Institute of Technology May 22, 1998

© 1998 Rong Fong Tian. All rights reserved.

The author hereby grants to M.I.T. permission to reproduce and distribute publicly paper and electronic copies of this thesis and to grant others the right to do so.

Author

Department of Electrical Engineering and Computer Science

May 22, 1998

Certified by

Glen L. Urban

Dean

Thesis Supervisor

Accepted by

Arthur C. Smith

Chairman, Department Committee on Graduate Theses

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

JUL 1 1998

LIBRARIES
Nonlinear navigation and context-sensitive help in the Sloan Electronic Commerce Project

By

Rong Fong Tian

Submitted to the Department of Electrical Engineering and Computer Science on May 22, 1998, in partial fulfillment of the requirements for the Degrees of Bachelor of Science in Electrical Engineering and Computer Science and Master of Engineering in Electrical Engineering and Computer Science

Abstract

This thesis describes the construction of a Web-based virtual environment called “Trucktown.” Trucktown is a part of the Sloan Electronic Commerce project, which aims to study Web-based marketing technologies and strategies. Trucktown provides an interactive shopping experience for pick up truck consumers. This experience focuses on building a trustworthy relationship with the consumer, through the employment of various trust-building features, such as accurate and abundant information and needs-based product recommendation. Trucktown is implemented as a Java applet running in the Netscape browser, connecting to a back-end Microsoft Access database. I was responsible for the design and implementation of the user interface and the underlying environment of Trucktown.

Thesis Supervisor: Glen L. Urban

Title: Dean
## Contents

1. Introduction  
   1.1 Web and Electronic Commerce  
2. Thesis Overview  
   2.1 Objective  
   2.2 Methodology  
   2.3 Summary of Results  
3. Technology  
   3.1 Java and Object-oriented programming  
   3.1.1 Java Overview  
   3.1.2 Object-Oriented programming  
   3.2 Development environment  
   3.3 Access Database  
4. Features  
   4.1 User Interface  
   4.1.1 Main Map  
   4.1.2 Guide  
   4.2 Assistance in selecting a product  
   4.2.1 Expert Advisors  
   4.2.2 Auto Mile  
   4.2.3 Consumer segmentation based on amount of knowledge  
   4.3 Information Delivery  
   4.3.1 Town Hall  
   4.3.2 Showroom  
   4.3.3 Newsstand  
   4.4 Consumer community interaction  
5. Architecture  
   5.1 User Interface Implementation  
   5.1.1 Environment: TrucktownApplet and TrucktownFrame  
   5.1.2 Main Map and Guide  
   5.2 Client/Server  
   5.3 Data structures  
6. Project management  
   6.1.1 Programming methodology  
   6.1.2 Milestones  
7. Research Summary  
8. Next Steps
1. Introduction

The Sloan Electronic Commerce project aims to study the concept of trust-based marketing. Trust-based marketing focuses the marketing effort on promoting trust between the consumer and the retailer, thereby increasing the likelihood of purchase.

There are eight sections in this paper. Section 1 will introduce web-based marketing and how it affects electronic commerce. I will also give a brief background on the project, which was originally known as Information Acceleration. Section 2 discusses the scope, methodology, and summarizes the project.

Section 3 discusses technology choices made in the construction of Trucktown. Trucktown was an Object Oriented and client/server application. It was written as a Java applet in the Symantec Visual Café development environment, connecting to a back end Microsoft Access Database.

Section 4 describes the features in Trucktown: assistance in selecting a product, information delivery, community interaction, and intuitive user interface.

Section 5 describes the technical architecture behind these features. I will highlight the design of the user interface and the client/server structure.

Section 6 discusses project management issues encountered in Trucktown. Finally, section 8 and 9 discusses future research directions and concludes the paper.
1.1 Web and Electronic Commerce

In the last few years, the World Wide Web has become an important part of the commerce space. The web started out as a vast repository of easily accessible information. Initially, the information is stored as static pages, which everyone with a capable browser can view. More recently, with the advent of dynamic content capabilities, such as CGI scripts and Java applets, the Web has grown to become a medium for electronic commerce.

A large part of a successful electronic commerce effort is marketing, and marketing on the Web is especially intriguing. Traditional marketing targets specific market segments with a set of predefined messages through broadcasting media such as television, newspaper, and radio. A message is designed to generate the most impact for the most people in a particular segment. The Web is also a broadcasting medium, but its real value is its interactivity: a person browsing the Web can interact with the content on a particular web site. A Web-based marketing effort can be directed on an individual rather than a large population segment. By gathering information about the user on the web site, a marketing message can be personalized for that user. Thus, the Web allows the construction of interactive environments in which the marketing results are greatly enhanced through personalized messages for an individual consumer.
With a personalized marketing message, an electronic commerce web site can proceed to sell its product to informed and receptive consumers.

1.2. Previous Work

General Motors funded the Sloan Electronic Commerce project in 1996 to study web-based marketing for automobiles. Initially named Information Acceleration (Urban et al, 1995), the project aimed to devise a methodology for predicting sales of innovative products that have not arrived on the market. The first version of the project (Shoemaker, 1996) was developed in spring of 1996. It consisted of an interactive dialogue with an elimination-based expert engine. The consumer is led through a series of questions about their needs for a pickup truck, including "Are you going to use the truck for off road driving?" and "how many passengers do you need to carry in the front seat?" The expert then recommends several trucks based on the consumer's answers. The consumer is then led to a showroom displaying the recommended trucks. The use can then browse through information for these trucks. From the showroom the consumer can also visit a bulletin board and interact with other consumers using the site. He may browse through the archived questions and answers, and post his own questions.

The first version was well liked by the sponsors and pickup truck drivers in the marketing research. But it had many limitations. One limitation was the sequential user interaction. The user must go through all the questions in the same order in order to get recommended trucks and view them in the showroom. If the user wanted to view other trucks, there
was no other way than to go back and answer the questions differently. Thus, the user had little control over the interactive dialogue. The second limitation is the elimination-based recommendation algorithm. When a truck did not fit the answers in a question, it is immediately discarded. The next question would eliminate some more trucks, and so on, until the last question. The problem was: given the large number of questions (about 15), all the trucks would be eliminated halfway into the questions. This problem was temporarily patched by counting the number of times a truck was eliminated and recommending the trucks that had the lowest elimination count. This was, of course, not a particularly great solution.

All in all, the first version of the project provided a solid starting point for an electronic commerce site. It demonstrated the concept of product recommendation and personalized marketing.
2. Thesis Overview

2.1 Objective

This thesis focuses on the development of the second version of the project. The goal is to expand the linear dialogue in the first version into a truly interactive shopping environment named "Trucktown." The motivation for this research direction was the impact of control over the shopping experience on the trustworthiness of the site. It was found that the more control a consumer had when browsing the site, the more trustworthy the site seemed. The first version gave the consumer little control with its sequential expert dialogue. Therefore, Trucktown was made with as much control over navigation and functionality as possible for the consumer.

Similar to the first version, Trucktown contained an expert vehicle recommendation system, a virtual showroom, consumer community interaction. However, these modules are no longer connected sequentially. The consumer can choose to visit them in any order. There are also several new modules in Trucktown. There is an intelligent guide that introduces the consumer to Trucktown, provides assistance when needed, and guides the consumer around the town. Other modules include a chat room, a town hall that dispenses important information on consumer rights in Trucktown, and a newsstand that contain popular literature such as “Car and Driver.” In order to support these modules, Trucktown has an underlying environment that facilitates many of the common interactions.
2.2 Methodology

Building Trucktown was an iterative and exploratory process. Given this was a research project, we were pushing the boundary of Internet-based electronic commerce product. We did not start with a clear plan of attack and precise requirements. Instead, we sought to experiment and explore new and better ways to doing things, keeping in mind the goal of the project, which was to make electronic commerce an accessible and powerful tool. Some features did not even occur to us at the beginning of the research, while many others emerged as a result of countless iterations.

There were four programmers working on Trucktown, two UROP students working 10 hours a week; and two MEng students, Ken Lynch and myself. Ken Lynch focused on the algorithms and technical architecture of the expert advisors, while I was responsible for overall design of Trucktown environment, as well as the modules other than the advisors. The UROP students worked on portions of various modules, as the project demanded.

2.3 Summary of Results

Trucktown version 2.0 was completed in April. A marketing study was conducted comparing Trucktown against Microsoft Carpoint, the leading web-based electronic
commerce site for trucks. Around 300 truck drivers spent 20 to 30 minutes on the each site. The results are being analyzed at the time of writing this thesis, but the preliminary conclusions are promising. It was found that people in general liked Trucktown, especially the expert advisors that provided recommendations. Although the overall preference was given to Carpoint, Trucktown compared favorably against Carpoint in many criteria, including trust in the site and willingness to purchase a vehicle from the site.

Java proved to be an excellent tool for the client/server portion and the expert algorithms. However, the relative immaturity and rapid evolution of the technology created problems. For future research, other technologies such as dynamic HTML may be used to complement Java.
3. Technology

Technology selection is crucial to a software project. There is a plethora of technologies available for use with the Web, such as CGI, Dynamic HTML, Java, etc. Using one technology produces an entirely different product than another one, since different technology enables different things to be done. In the end, Trucktown was an Object Oriented and client/server application. It was written as a Java applet in the Symantec Visual Café development environment, connecting to a back end Microsoft Access Database.

3.1 Java and Object-oriented programming

3.1.1 Java Overview

Released in 1995, Java has become the hottest technology on the Internet. A large part of its appeal comes from its ability to run the application on different platforms. The Internet is heterogeneous, that is, you can find every kind of platform out there, from Macintosh to UNIX stations to PC’s running Windows. The myriad of platforms is a headache for anyone wishing to development an Internet application, since this application will need to be ported to several different platforms, each requiring special expertise as well as significant development cost.
Java promises to change all this by running its applications on a virtual machine. In the case of C++, which is used to write a large number of business applications, the compiler directly compiles the source code into platform-dependent machine code, which can only be run on that particular platform. In contrast, the Java compiler compiles the source code into an intermediate byte-code format, which is defined independently of any platform. In order to execute the application, a piece of software called the virtual machine is used to translate the byte-code into platform dependent machine code on the fly. Thus, only the virtual machine needs to be rewritten for each platform. An application in the byte-code format can be executed on any computer with a Java virtual machine. This frees the programmer to focus on developing applications that take advantage of the Internet and the Web, with assurance that their applications would behave the same whether they are running Windows or UNIX. The obvious disadvantage of using a virtual machine is the degradation in performance. However, with the emergence of just-in-time compilers, Java’s performance has increased by leaps and bounds. Sun Microsystems’ upcoming HotSpot compiler promises to raise Java’s performance even beyond C++.

Java’s platform-independence is a major reason for choosing Java for Trucktown, since Trucktown needs to be available to many customers over the Internet. In addition, from a programming point of view, Java is a great tool because it is a pure object-oriented language. Object-oriented languages offer great advantages in programming, and they are discussed in the next section.
3.1.2 Object-Oriented programming

The concept of object-oriented programming has been around for a long time, but it only came into mainstream in the last few years. In essence, object-oriented programming is a conceptual approach to programming. It is a way for a programmer to make sense of the world around him and the program he is writing. Thus, it is more than a collection of techniques, but a mental model that applies to all processes of programming.

In the old style of procedural programming, the application is composed of a collection of data types and the functions that manipulate the data. In contrast, an object-oriented application is made up of a collection of “objects.” These objects contain data and functions, called methods that act on the data. The key here is that the data is hidden inside the objects, so that the internal workings of an object are hidden from an outside viewer. One interacts with the object by calling its methods, which specify the behavior of the objects. The functionality of the application is performed by its objects, or through the interactions between these objects.

There are many advantages of object-oriented programming. I will highlight two of them here, modularization and reusability.

Object-oriented programming enables us to modularize our design so that multiple modules can be developed simultaneously, provided the interconnections between these modules are well defined. The programmer who is working on one module does not need
the knowledge of the other modules in order to do his work, he just needs to understand how his module interacts with the other modules. This enables specialization of a programmer, so that he can focus on the things that he does well. In Trucktown, we were able to divide the features into modules that can be programmed in parallel, significantly reducing production time.

Another significant advantage of object-oriented programming is reusable objects. When the interfaces of objects are well defined, one can reuse objects in different applications. For example, a text-editing object can be reused in a word processor as well as a programming development environment. This enables tremendous productivity gain, since programmers no longer have to rewrite common functionalities from scratch. Applications can be built by plugging in different reusable components like Lego blocks. Reusing components lead to lower cost as well as shortened development time.

Finally, object-oriented programming enables the application components to resemble real-world objects. For example, in Trucktown the component Main Map does exactly what the real-world counterpart does: it provides a navigational hub for Trucktown, and the user can go to any location in Trucktown from the Main Map. This was very helpful in the design of Trucktown, where there was a natural mapping from real-world objects to programming objects.

3.1.3 Java JDK issues
Since Java is still at its infant stage, there are many risks and potential problems that come with its power and usability. One of the biggest frustrations comes from the rapid evolution of the Java Development Kit (JDK). The JDK contains the Java language and core classes that enable a programmer to write Java programs. It is the all-in-one toolbox for the programming world. Since Java’s introduction in 1995, the JDK has been evolving in a break-neck pace. Most of the core classes have undergone vast changes and improvements. This creates a problem when coupled with the cross-platform abilities of Java: you need to be careful with which version of the JDK you are developing, since different platforms may support different versions of the JDK. Java programs that contain new features may not run at all in an older version. This is a problem because it takes significant effort to port a new JDK to a particular platform.

Trucktown is an applet meant to be deployed in Netscape. If this were commercial product, then we would have to support other browsers including Microsoft Internet Explorer. Since this was a research project and thus controlled, we could limit the deployment to Netscape, in order to minimize potential problems. After choosing Netscape, we had to be very careful which version of JDK was supported in Netscape. When we started developing Trucktown, we had to choose between JDK 1.0 or 1.1. JDK 1.1 was released in summer of 97, and it contained an improved AWT package over 1.0. AWT stands for Abstract Windows Toolkit, and it contains the essential GUI building blocks such as panels, buttons, and labels. Trucktown is a GUI intensive program, so the decision between JDK 1.0 or 1.1 was a very important one.
JDK 1.1’s AWT package was superior to 1.0, but the latest version of Netscape supported only 1.0. All the latest development environments were running JDK 1.1, so Netscape was lagging behind the crowd. Fortunately, Netscape recognized the demand for 1.1 support, and it made mad a patch available that enabled a particular browser to run JDK 1.1 applets. The superior GUI capabilities of JDK 1.1 meant that we could put more functionality into the program. Thus, we decided to go ahead and use JDK 1.1 and use the patch from Netscape to run the applet. Later on, this proved to pose more problems that we originally anticipated. These problems will be highlighted in the Implement section.

3.2 Development environment

Trucktown version 1.0 was developed by Jon Shoemaker in Metroworks CodeWarrior, an integrated development environment (IDE) (Shoemaker, 1997). CodeWarrior was extremely unwieldy, since it was primarily a Macintosh development environment, and not written for Windows NT. Among the problems we encountered was its instability. The software would crash erratically while compiling, sometimes even erase the changes made to the code. In addition, CodeWarrior had almost nonexistent debugging capability, which is crucial to the development of any software.

In selecting a development environment, we listed the following qualities as desirable:

1. Debugging – the IDE must have adequate debugging abilities, such as setting break points and line-by-line stepping.
2. Ease of use – the IDE must have an easy-to-use user interface, with a good set of editing features.

3. Code management – the IDE must have a good code management system that gives structure to the large project.

The IDE’s that we looked at were:

- Microsoft Visual J++
- IBM Visual Age
- Borland JBuilder
- Visual Café Professional

We chose Visual Café because of its balance of features and freedom of coding. Visual Age and JBuilder had great set of tools, but they both restricted the programmer to a by-method coding approach, where for each class, you could only see and edit one method at a time. This approach was more in tune with a pure object-oriented thinking, but we felt it was too unwieldy. In contrast, Visual Café enabled both by-method coding and by-class coding, where you are free to roam around the code of the entire class.

In addition, Visual Café had a powerful debugging environment that was simple to use yet very reliable. One could learn the interface in a short amount of time, greatly enhancing productivity.
3.3 Access Database

Trucktown contained information on 86 pickup trucks. For each truck, there were over one hundred data points, including price and horsepower. In version 1.0 of Trucktown, the data was stored in a tab-delimited text file, which was parsed by the program. This was not acceptable for version 2.0, since most changes made to the data file would need changes in the code as well, and if there was an error in the database, it was extremely difficult to debug. Since version 2.0 is a much larger program, the database itself needed to accommodate continual addition and modification of data.

Thus, we decided to use an Access database for data storage. Other databases were considered, including Oracle and Microsoft SQL Server, but the size of the project did not warrant the added complexity and cost.
4. Features

The goal of Trucktown is to assist consumer in purchasing a pick up truck. To this end, the following key features were incorporated into Trucktown:

- intuitive user interface
- consumer community interaction
- information delivery
- assistance in selection a product

This section discusses each of these key features and the components that illustrate them:

- Main Map
- Guide
- Advisors: neighbor, mechanic, and editor
- Auto Mile
- Town Hall
- Showroom
- Newsstand
- Coffee Shop

4.1 User Interface
As a consumer product, Trucktown must be intuitive to navigate and use. Each location in Trucktown is modeled after a real world environment, and there is a person in each location to greet the consumer and help her make use of the functionality. To give authenticity to Trucktown and to make the interactions as natural as possible, photographs of real people and real locations are used and most of the interaction is in a conversational style. For example, when the consumer goes to the garage, there is a mechanic that introduces himself and tells the consumer what services he can perform. The dialogues are written to be friendly yet concise, so that the consumer gets enough information about the mechanic to visualize real personality behind the image, but she does not get tired of wading through a ton of dialogue screens.

For quality of presentation and cohesiveness among all the modules, we hired a graphic artist to draw the buttons, icons, and main map. A photographer took the photographs in each module. The professional appearance of Trucktown added to its trustworthiness as a legitimate commerce tool.

In addition to the natural user interface in each module, Trucktown features two components that assist the consumer in navigation: Main Map and the Guide.

4.1.1 Main Map
The Main Map of Trucktown, as show above, is the starting point and central hub for Trucktown. The consumer is given a bird’s-eye view of Trucktown, and can access any location from the map. When the mouse moves over a location that can be accessed, such as the Town Hall, the icon darkens, signaling that the user can visit this location. After the consumer exits a particular module, she is returned to the Main Map.

4.1.2 Guide

The owl on the map is the Guide for Trucktown. The guide’s main purpose is to provide information on Trucktown’s modules and assist the consumer in navigating around Trucktown. An owl was chosen over other characters because it had the connotations of being wise and helpful, essential characteristics of a guide.
When a consumer enters Trucktown, the guide introduces her to the purpose of Trucktown. The guide also offers to lead the consumer around the town, explaining what is available at each location, as well as suggesting the next logical place for the consumer to visit given where she has already. There is an optimal order in which the locations should be visited, and the Guide makes his suggestions based on this order. For example, the first place the Guide suggests is the Town Hall, where the consumer can get information on the policies of Trucktown. After each location is visited, the Guide will no longer suggest that location as the next place to visit.

If the consumer wants to explore Trucktown on her own without the lead of the guide, she can choose to click on the button labeled “Let me look around Trucktown on my own” to turn off the guide. After the guide is turned off, it shrinks down to a help button at the top left-hand corner of the map, and the consumer is free to wander around the town. When the consumer needs further help, she can click on the help button to bring up the guide.

In addition, the Guide also enforces restrictions on the visiting order of some modules. For example, since the Showroom displays the trucks that are recommended by the expert advisors, if the consumer has not visited any advisor, she should not be allowed to enter the Showroom. When the consumer attempts to go into the Showroom in this scenario, the Guide will appear on the screen informing the consumer that she has not visited any advisor, and offers to take her there.
4.2 Assistance in selecting a product

The pick up truck is a complex product with a large amount of specifications and accessories. With over eighty trucks available on the market, the consumer is presented with a bewildering array of choices. Selecting the right truck requires a high degree of knowledge about trucks, and it takes time for a person to gather enough knowledge to make an informed choice. Trucktown seeks to alleviate the consumer’s information overload by offering assistance in selecting a truck. Product assistance is offered through two different methods in Trucktown: need-based configuration, used in the expert advisors, and elimination engine, used in Auto Mile.

4.2.1 Expert Advisors

The expert advisors employ need-based configuration (Lynch, 1998) to make recommendations for the consumer. There are three advisors residing in Trucktown: a mechanic, a Consumer Reports Editor, and a contractor who’s a neighbor.

All three advisors have the same interface. The advisor first introduces him or herself to the consumer via a dialogue. Then consumer is asked a series of questions to find out what her preferences are, such as:

- How much are you willing to spend?
- How are you going to use the truck?
Some of these preferences include:

- Price
- Truck style
- Truck bed length
- Number of passengers
- Engine type

After enough data points on the consumer’s preferences have been collected, the advisor then proceeds to make recommendations of four trucks. These recommendations are made through two distinct methods: Bayesian and segment recommendations.

Bayesian recommendations are based on probabilistic analysis of truck. For each answer in a question, each truck has an a priori probability of being bought, based on past purchases. Let’s call this probability the “purchase” probability. For example, for the question “How are you going to use the truck?” An answer of “Off-road driving” would imply the need of a 4-wheel drive. Therefore, a truck without 4-wheel drive option has a low a priori purchase probability.

When the consumer answers a question, the overall probability of a truck being bought is updated with the a priori purchase probability for that answer. This is done for all trucks. Thus, the answer of “Off-road driving” would lower the overall probability of trucks without 4-wheel drive. As the consumer goes through the series of questions, trucks’
purchase probabilities rise and fall based her answers. When the questions are finished, or when the consumer decides to quit, the top trucks are recommended.

Segment recommendations are made based on market segments. Truck manufactures segment the market into 8 distinct segments. The trucks in each segment have similar characteristics, such as “big and powerful,” or “compact and fuel-efficient.” These segments define the customers’ needs, and require a specific set of preferences. Thus, segment recommendations are made based on matching the consumer’s preferences with a market segment. The trucks in that segment will then be recommended to this consumer. Given that there are a small amount of segments, the recommendations may not fit some of the consumer’s preferences, but in general they will be adequate.

The advisor makes the final recommendation of four trucks by combining two Bayesian trucks, and two segment trucks.

4.2.2 Auto Mile

A second method of product recommendation is available in the Auto Mile. This is an elimination engine that enables a consumer to pick trucks with specifications. The consumer access the engine through a tabbed panel, with the following five feature panels:

- Price
- Brand
Each feature panel contains a set of options for that feature. For example, the Engine contains options for “V6,” “V8”, “Diesel”, etc. When no options are checked, the engine assumes all choices are allowed, thus no trucks are filtered out. When one or more options are checked, only the trucks that include those options will be left on the list. The engine updates the list of selected trucks in real time after each option is selected or deselected.

By clicking on the name of a truck in the truck list, the consumer can access information about the truck in a similar fashion as in the Showroom, which will be discussed in the Information Delivery section.

4.2.3 Consumer segmentation based on amount of knowledge

Trucktown offers two different ways to assist the consumer in product selection in order to target consumers with different levels of knowledge about trucks. Trucks as products require a large amount of knowledge to make sense of, and the levels of knowledge in consumers vary greatly. The expert advisors are aimed at consumers who have a small amount of knowledge about trucks. The advisors hide the vast amount of technical
jargon and specifications from the consumer, and instead focus on the consumer’s needs. Thus, a consumer who do not know a V8 from a V6 can still find trucks that fit her need.

In contrast, there are also consumers that have a great deal of knowledge about trucks. They already know what kind of trucks they want in terms of technical specifications, such as a particular engine and horsepower. For these consumers, Auto Mile offers an alternative to the involved recommendation process of the advisors. In a few clicks, a consumer can find a list of trucks that meet her specifications. This way, knowledgeable consumers do not have to go through a dialogue with an advisor in order to make most of their shopping experience.

Of course, a consumer can elect to use both advisors and Auto Mile for assistance. They complement each other in terms of power and ease of use.

4.3 Information Delivery

Crucial to the success of any electronic commerce site is the quantity and quality of information offered on the product. After all, consumers want to make an informed decision, and they would want to be able to access as much quality information as possible. After a consumer has found the trucks she is looking for, either through Auto Mile or the expert advisors, she would now want to access all the information on these trucks that would help her in her decision. In addition to the quality and quantity of information, it should be easy for the consumer to access any information. Information
that is unorganized or hidden deep inside a site can easily inundate and frustrate the consumer. Trucktown makes its information available to the consumer in these logical and intuitive locations:

- Town Hall
- Showroom and Auto Mile
- Newsstand

4.3.1 Town Hall

The Town Hall contains consumer rights information, including product quality guarantees, disclaimers, and other policies of Trucktown. These policies give consumers an understanding of the service they are getting in Trucktown and what they should expect. In addition, they give credibility and legitimacy to Trucktown, increasing the trustworthiness of the buying process.

4.3.2 Showroom

After the consumer has visited an advisor, she can go to the Showroom to get a great deal of information about the trucks recommended by the advisor. When the consumer first enters the Showroom, she is shown the four recommended trucks. From there she can access the following types of information:
There are a total of 86 trucks presented in Trucktown, each with over one hundred pieces of information, such as price and accessories. These specifications are gathered from manufacturers and on-line sources.

There are two types of specification comparisons available. The consumer can compare all four of the trucks recommended by the advisor, or she can compare a particular truck against other trucks that are popular with other consumers in the same market segment.

The financial information panel offers the consumer information on the different financing options available, such as leasing, monthly payments, etc.

Customer comments about a particular truck are also featured. These comments give the consumer a perspective on how real consumers perceive the truck. They reinforce trustworthiness of Trucktown.
In addition to viewing the information about a particular truck, the consumer can also arrange for a test drive. The consumer fills out a form with their personal information, which is then given to the nearest dealer in the consumer’s area to schedule a test drive. This feature is currently here only for demonstration purposes, and for a commercial version, it will be linked with popular car buying services such as Auto-By-Tel.

4.3.3 Newsstand

Popular literature, including “Car and Driver,” “4 By 4,” and “Motor Trend,” are available in the Newsstand. The consumer can read about their favorite trucks from any one of these magazines. These magazines provide additional information resources for the consumer.

4.4 Consumer community interaction

Virtually all Internet sites nowadays offer chat as a way to promote community among the users visiting the site. The site provides the user with a shared experience around which to socialize. In order to promote interaction among the consumers, who are visiting Trucktown at the same time, Trucktown offers a chat room in the Coffee Shop. This allows consumers to talk about their experiences in Trucktown, and get more information about trucks of their choice from their fellow consumers.
5. Architecture

When designing Trucktown, we tried to keep the modules fairly independent of each other, with little or no overlap in functionality and data. They each manage their own resources and layout, so that they could be developed in parallel without worrying too much about impacting other modules. The only exception of the rule is the Auto Mile, which uses the information panels found in the Showroom.

This section discusses the design of Trucktown from several perspectives:

- User interface
- Client/server
- Data structure

5.1 User Interface Implementation

Most of Trucktown consists of GUI components. Here’s the overall module diagram of Trucktown:
5.1.1 Environment: TrucktownApplet and TrucktownFrame

TrucktownApplet is the applet loaded into Netscape. It creates TrucktownFrame on instantiation. TrucktownFrame is the root for all objects in Trucktown. Upon initialization, it instantiates all the modules in Trucktown. The MainMap is then displayed, awaiting user input.

In addition to containing all the modules, Trucktown provides a uniform way of navigating from one module to another, via

\[ \text{goToModule(String moduleName)} \].

Upon exiting a module, \text{goToModule()} is called to display either the Main Map, or the next appropriate module. It cleans after the old module and perform any special preparation needed to display the new module.
TrucktownFrame also includes two helper classes, SessionModel and SessionUtil. SessionModel allows the module to access all the data available, such as dialogue text and truck information. SessionUtil provides utility methods to obtain resources such as images and text.

In their constructors, all the modules are passed a reference to TrucktownFrame, so that they can access SessionModel, SessionUtil, and the `goToModule()` method.

5.1.2 Main Map and Guide

The Main Map and Guide facilitate navigation around Trucktown. For each location on the Main Map, there’s a corresponding HotArea that captures mouse clicks and invoke TrucktownFrame’s `goToModule()` method to display the desired module.

The Guide serves two functions. Its primary responsibility is to offer suggestions on which module the consumer should go next. It also enforces the order in which certain modules should be visited.

5.2 Client/Server
Trucktown is a client/server application. The client is the TrucktownApplet running in Netscape, and the server is a Java application residing on a back end Windows NT machine. The primary function of the server is to enable the client to get data from and put data into the Access database.

Two pieces of technology were used to implement client/server architecture in Trucktown:

- Remote Method Invocation (RMI)
- Java Database Connectivity (JDBC).

RMI enables the programmer to create distributed Java-to-Java applications, in which the methods of remote Java objects can be invoked from other Java virtual machines, possibly on different machines. A Java program can make a call on a remote object once it obtains a reference to the remote object, either by looking up the remote object in the bootstrap naming service provided by RMI or by receiving the reference as an argument or a return value. In our case, the remote object is the server serving data from the database. RMI hides the underlying network operations so that the programmer can concentrate on the functionality of the client/server application.

JDBC enables a Java application to access any database supporting Open Database Connectivity, or ODBC, in this case, Access database.
There were also other distributed computing solutions such as CORBA, but we chose RMI for its ease of use and integration with Java applications.

![Client/Server Interaction Diagram]

Figure 3: Trucktown client/server interaction diagram

### 5.3 Data structures

A major contributor to the overall quality of an electronic commerce site is the quality and quantity of information. This is especially important in Trucktown since the pick up truck is a product that has a large amount of specifications. These specifications and other product-related information are stored in the Access database. There are also other
static data fetched over HTTP: text for the interactive dialogues, and images produced by our photographer and graphic artist.

There are five different sets of truck-related data in Trucktown:

- Manufacturer’s truck data
- Articles for Newsstand and Showroom
- Bayesian probabilities for the expert advisors
- Market segment information for each truck
- Recommendation reasons for the expert advisors

Each set of data is stored in a separate table in the Access database, keyed by the truck names.

The first set of data contains all the manufactures’ information on the trucks. This includes specifications, prices, and options available. For each truck, this entailed to over one hundred data points. Since there are 86 trucks on display in Trucktown, this is a huge table. Originally the market segment information and Bayesian probabilities were also stored in the truck data table, but it became too large for Access to handle, so we chose to take them out and give them their own tables.

After we have retrieved data from Access, we need some ways of representing the data so it can be used effectively in Trucktown. We created a class called Truck that
encompasses all the data from the Access database. Since the manufacture’s data points were in the format of decimals and strings, we used a decimal array and a string array to contain these data points. For each data point, its name was used in Truck class as a static integer variable set to the index of this particular data point in the data arrays. The rest of the data, such as the Bayesian probabilities, were stored as other arrays and Hashtables in the Truck class.
6. Project management

Good project management was essential to the success of Trucktown, since Trucktown was a fairly large project, with four programmers working a total of 80 hours a week for 7 months. The complexity of project called for structured project management, as well as uniform coding practices. This section describes the project management approach we took, as well as the milestones we established.

6.1.1 Programming methodology

There were several programming methodologies that we considered for this project. Waterfall is one of the most widely used and also oldest. It consists of a series of "waterfalls," where each waterfall is a predefined procedure leading to the next procedure. The procedures are commonly: requirement analysis, design, implementation, test, and maintenance.

There are several problems with this approach:

- Real projects rarely flow in a sequential process.
- It is difficult to define all requirements at the beginning of a project.
- This model has problems adapting to change.
- A working version of the system is not seen until late in the project's life.
Since Trucktown was a research and therefore exploratory project, the initial requirements were not precise. We had some ideas as to what the applet would look like and what its rudimentary functionalities were, but there was not enough clear and concrete requirements for us to make a detailed working plan. During our research, if we encountered an innovative solution to our problem, or if we had thought of a great new feature, we would need the flexibility to be able to incorporate them into the final product.

Therefore, the inability of the waterfall model to adapt to change would be a costly shortcoming. If there is any change in the requirements and the project is already in the coding phase, it will be costly to accommodate the additional changes. In addition, Trucktown was a GUI intensive project, and there were many usability issues involved in the design. Not being able to see the working version of the system until the last few weeks would have guaranteed usability problems and would be also costly. Being able to see Trucktown as soon as possible was also important since we had to demo it to our sponsors, GM, in November of last fall.

The other methodology we considered was a more modern rapid development methodology, the spiral model. The name “spiral” comes from iterating the same steps over and over again in greater complexity.

There are four phases associated with each major cycle of the spiral.
Phase 1: The baseline approach and appropriate alternatives are developed to meet program objectives.

Phase 2: The approaches are evaluated against the objectives and alternatives, and the risks associated with these approaches are evaluated.

Phase 3: The prototype is evaluated, and the next level of the product is developed. This phase results in a prototype of the design.

Phase 4: The product is reviewed, and plans for the next development stage are established.

The entire process is then repeated to the next level of detail.

In this model, the project is mapped out in a series of milestones based on functionalities. At each milestone, there would be a working version of the project with the functionalities specified for that milestone. Each subsequent milestone would add more functionality, but the overriding principle is that the project must be working and able to be demoed at each milestone.

We chose the spiral model, since it adequately addressed the uncertain requirements and the exploratory nature of the project.
6.1.2 Milestones

Three milestones were used in the project lifecycle.

1st milestone:
Due date: December
Deliverables:
- Trucktown basic environment:
  - TrucktownFrame, SessionModel, SessionModel
- Main Map
  - Hot Areas
- Expert advisor:
  - Basic GUI
- Showroom
  - Basic Truck specifications

2nd milestone:
Due date: March
Deliverables:
- Expert advisor:
  - Improved logic
- Guide
- Showroom
- Articles
- All truck specifications
- Coffee shop
- Server

3rd milestone:

Due date: March

Deliverables:
- Newsstand
- Auto mile
  - Truck search
  - Truck information panels from Showroom
- Expert advisor:
  - Reasons for recommending this truck
- Showroom
  - Financial information

We largely adhered to these milestones. Some functionalities could not be completed as well as others, but the ones we did were prioritized as the most important ones.
7. Research Summary

Trucktown aimed to explore the new frontier of web-based marketing through an interactive environment. It combined features such as needs-based product recommendation and community interaction into a virtual town that was appealing to consumers. This thesis focused on the development of the user interface and the general environment, such as the client/server architecture. The user interface emphasized freedom of navigation and intelligent guidance for the user.

The use of Java and related technologies proved to be excellent for the recommendation engines, but the incompatibility issues in different versions of Java made GUI development less than efficient. The concept of web-based marketing was well demonstrated by Trucktown.
8. Next Steps

Future research can be divided up into two main directions: product recommendation technologies and UI. The needs-based product recommendation engine is only in the first stage; there are many more extensions possible. The second direction is in the UI. Good UI is crucial to the success of an electronic commerce site, and it was not a particular focus for this version of Trucktown.

8.1 Product Recommendation

The product recommendation technologies were only explored on the surface in Trucktown. There were many issues that we have not tackled, such as detecting inconsistencies in consumer preferences.

Consumers often give conflicting responses to the advisor’s questions in an effort to satisfy all of their different needs. For example, they may ask for a small truck that can haul a boat, not realizing that hauling a boat requires a powerful, therefore large truck. The recommendation algorithm should be able to recognize these inconsistencies and point them out to the consumer. The consumer could then make a choice between their conflicting needs.
8.2 User Interface

8.2.1 UI environment

The Trucktown interface was designed to be as user friendly as possible, and as such, was regarded by some as not serious enough. In our market research, some truck drivers described Trucktown as “childish,” “too much like a game”, and “not professional.” These drivers had a negative view of Trucktown. In addition, since the GUI is pure Java, the performance is less than spectacular. Since this was a controlled experiment, the performance of high-powered computers. In order to make Trucktown closer to commercial deployment, the GUI could be re-implemented with dynamic HTML and server-push technologies. It would become much more lightweight, albeit entailing a complete redesign of the site.

8.2.2 Search

There is a lot of information in Trucktown, ranging from truck specifications to customer comments to magazine articles. However, much of this information was buried in the different locations around Trucktown. It takes a little while before the user can get access to the information that she wants. Since the information was not immediately available, people perceived the amount and quality of information as less than actual. Thus, it would be helpful to add search functionality to the site, so that visitors can quickly get access to information they want, without having to figure out where to visit in
Trucktown. This requires a careful analysis of all the available information in Trucktown. The information then needs to be organized and put into data structures so that they could be sorted and searched with keywords. This would require a completely new back end server as well as new database design. Ultimately, the product recommendation algorithms can be applied to this search to make it a powerful tool.
9. Conclusion

Internet and the Web have revolutionized the way we do business. The vast amount of information that previously only resided with the manufactures, dealers, and publications, is now available to any consumer with a computer and an Internet connection. Given this vast pool of information, the next step is to help the consumer to make sense of it all. In addition, the stereotypical car buying experience is a universal pain, bringing up mental images of sleazy salesmen and overpriced merchandises. Trucktown attempted to find out how we can inform and assist the consumer in making good purchase decisions, while at the same time make the process enjoyable.

This thesis focused on the user interface in Trucktown, as well as the general environment. Working on Trucktown was an interesting and challenging experience. The commercial motive of the project gave a real world dimension to the research. Being able to push the boundary of the electronic commerce ideas and technologies was both rewarding and exciting.
package Trucktown;

import Trucktown.News.*;
import Trucktown.Expert.*;
import Trucktown.ShowRoom.*;
import Trucktown.AutoMile.*;
import java.awt.*;
import java.applet.*;
import java.awt.event.*;
import symantec.itools.awt.ImagePanel;
import symantec.itools.awt.ImageButton;
import java.util.Vector;
import java.util.Hashtable;
import ice.htmlbrowser.Browser;

public class TrucktownFrame extends Frame {

    public String hostString = "http://vandelay.mit.edu/";
    public String baseURL = "http://vandelay.mit.edu/Trucktown/";
    public SessionModel data = null;
    public SessionUtil util = null;

    public boolean remote = true;
    boolean debug = false;
    boolean testGuide = true;
    int top = 20;
    Browser iceBrowser;

    TrucktownApplet applet = null;
    Button expertButton;
    //Trucktown
    //{{DECLARE_CONTROLS

    Hashtable moduleTable;

    // modules
    MainMapPanel mainMap;
    ShowRoomPanel showRoom;
    ExpertPanel mechanicExpert, neighborExpert, editorExpert;
    TownHallPanel townHall;
    public CafePanel cafe;
    DinerPanel diner;
    //xing's stuff: 2/2/98

}
NewsPanel newsstand;
AutoMilePanel autoMile;
TestDrivePanel testDrive;
MeetExpertsPanel meetExperts;
//}}
SpeechBubble speechBubble;

// reference to currently active module
Panel currentModule;
// Button expertButton;

public TrucktownFrame(TrucktownApplet parentApplet) {
    // added by Xing to add the password stuff.
    /*
     * PasswordDialog pwdd = new PasswordDialog(this, "Enter User ID and Password");
     * pwdd.show();
     */
    // end of modification by Xing 2/17/98
    iceBrowser = new Browser();

    applet = parentApplet;
    util = new SessionUtil(this);
    data = new SessionModel(this);
    moduleTable = new Hashtable();

    //{{ INIT_CONTROLS
    setLayout(null);
    reshape(insets().left, insets().right, insets().left + insets().right + 800, insets().top + insets().bottom + 630);
    setTitle("Trucktown");
    setResizable(true);

    speechBubble = new SpeechBubble(this, SpeechBubble.RIGHT);

    // the order that you add may end up being important!!
    moduleTable = new Hashtable();
    mainMap = new MainMapPanel(this);
    add(mainMap);
    mainMap.setVisible(true);

    moduleTable.put("MainMap", mainMap);
    //
    moduleTable.put("Guide", guide);

    /* three experts, **Ken** needs to modify these panels!!
    */
// expert 1
mechanicExpert = new ExpertPanel(this, ExpertPanel.MECHANIC);
add(mechanicExpert);
mechanicExpert.setVisible(false);
    mechanicExpert.reshape(0, top, 800,600);
moduleTable.put("MechanicExpert", mechanicExpert);

// expert 2
neighborExpert = new ExpertPanel(this, ExpertPanel.NEIGHBOR);
add(neighborExpert);
neighborExpert.setVisible(false);
    neighborExpert.reshape(0, top, 800,600);
moduleTable.put("NeighborExpert", neighborExpert);

// expert 3
editorExpert = new ExpertPanel(this, ExpertPanel.EDITOR);
add(editorExpert);
editorExpert.setVisible(false);
    editorExpert.reshape(0, top, 800,600);
moduleTable.put("EditorExpert", editorExpert);

// end experts

showRoom = new ShowRoomPanel(this);
add(showRoom);
showRoom.setVisible(false);
showRoom.reshape(0, top, 800,600);
moduleTable.put("ShowRoom", showRoom);

townHall = new TownHallPanel(this);
add(townHall);
townHall.setVisible(false);
townHall.reshape(0, top, 800,600);
moduleTable.put("TownHall", townHall);

cafe = new CafePanel(this);
add(cafe);
cafe.setVisible(false);
cafe.reshape(0, top, 800,600);
moduleTable.put("Cafe", cafe);

diner = new DinerPanel(this);
add(diner);
diner.setVisible(false);
diner.reshape(0, top, 800,600);
moduleTable.put("Diner", diner);
autoMile = new AutoMilePanel(this);
    add(autoMile);
    autoMile.setVisible(false);
    autoMile.reshape(0, top, 800, 600);
    moduleTable.put("AutoMile", autoMile);

testDrive = new TestDrivePanel(this);
    add(testDrive);
    testDrive.setVisible(false);
    testDrive.reshape(0, top, 800, 600);
    moduleTable.put("TestDrive", testDrive);

meetExperts = new MeetExpertsPanel(this);
    add(meetExperts);
    meetExperts.setVisible(false);
    meetExperts.reshape(0, top, 800, 600);
    moduleTable.put("MeetExperts", meetExperts);

//xing's stuff: 2/2/98
newsstand = new NewsPanel(this);
    add(newsstand);
    newsstand.setVisible(false);
    newsstand.reshape(0, top, 800, 600);
    moduleTable.put("Newsstand", newsstand);

    // bottom most panel
    mainMap.setHotAreas();

    data.initStateTable(moduleTable);

    /*These window events will be passed to the processWindowEvent() method where we can handle the user closing the window:
    */
    enableEvents(AWTEvent.WINDOW_EVENT_MASK);

    // on startup, load introduction
    mainMap.guide.setDialogue("intro 1 ");
    mainMap.setEnabled(false);
    moduleTable.put("Guide", mainMap.guide);
    currentModule = mainMap;
}

protected void processWindowEvent(WindowEvent e) {
    super.processWindowEvent(e);
    if (e.getID() == WindowEvent.WINDOW_CLOSING)
setVisible (false);
}

public void hideModule(String moduleName) {
    pln("hideModule: " + moduleName);
    Component module = (Component) moduleTable.get(moduleName);
    if (module != null) {
        module.setVisible(false);
        mainMap.setEnabled(true);
        pln("done hiding module: " + moduleName);
    }
}

public SpeechBubble getSpeechBubble() {
    return speechBubble;
}

/* Ask guide to check module states */

public void goToModule(String moduleName) {
    pln("goToModule: " + moduleName);
    mainMap.setEnabled(false);
    mainMap.setVisible(false);
    // hides all other modules

    /* Component modules[] = getComponents();
    for (int i = 0; i < modules.length; i++)
        modules[i].setVisible(false);
    */
    if (moduleName == "Exit") {
        // load up the exit dialog
        mainMap.guide.setDialogue("Exit!");
        mainMap.setVisible(true);
        mainMap.setEnabled(false);
        return;
    }
    speechBubble.setVisible(false);
    speechBubble.dialogueView.setVisible(false);
    if (currentModule instanceof ModulePanel) {
        ((ModulePanel)currentModule).setVisible(false);
    }
else {
    currentModule.setVisible(false);
}

Panel module = (Panel) moduleTable.get(moduleName);

/* 1/15: asks the guide to check currentModule's states, if the mainMap is the
target
   module.
   Guide just suggests destinations. */

pln("asking the guide to check module state");
mainMap.guide.checkModuleState(data.getModuleName(currentModule), moduleName);

if (module instanceof ModulePanel) {
    ((ModulePanel)module).setDialogue(data.getWelcomeDialogue(moduleName));
}

if (mainMap.guide.isVisible()) {
    // if guide is visible, disable mainMap
    pln("guide is visible.");
    module.setVisible(false);
    mainMap.setVisible(true);
    mainMap.setEnabled(false);
} else if (module != null) {
    // guide is NOT visible, and module isn't null
    if (moduleName.equals("MainMap")) {
        // going to main map
        speechBubble.setVisible(false);
        speechBubble.dialogueView.setVisible(false);
        mainMap.setHelpButtonVisible(true);
    } else {
        mainMap.setVisible(false);
    }
    // guide is NOT visible, just show the module
    module.setEnabled(true);
    module.setVisible(true);
    currentModule = module;
}
pln("Done going to module: "+moduleName);
    if (debug) {
        speechBubble.list();
    }
}

/**
   * This is called by the experts and showroom
   */

public void finishedWithModule(Panel module, int state) {
    pln("finishedWithModule() called.");
    data.setModuleState(module, state);
    goToModule("MainMap");
}

public void shutDown() {
    pln("Shutting down...");

    Panel endSurveyPanel = new Panel();
    endSurveyPanel.setLayout(null);
    endSurveyPanel.setBounds(0,0,800,600);
    endSurveyPanel.setBackground(Color.white);
    Label helpLabel = new Label("Please raise your hand and ask for assistance.");
    helpLabel.setFont(new Font("Dialog", Font.BOLD, 24));
    helpLabel.setVisible(true);
    helpLabel.setBounds(100,200,600,50);
    endSurveyPanel.add(helpLabel);
    add(endSurveyPanel,0);
    endSurveyPanel.setVisible(true);
}

public void p(String s) {
    if (debug)
        System.out.print(s);
}

public void pln(String s) {
    if (debug)
        System.out.println("TrucktownFrame: "+s);
}

public void update(Graphics g) {
paint(g);
}

public TrucktownApplet getParentApplet()
{
    return applet;
}

public Browser getBrowserBean() {
    if (iceBrowser == null) {
        iceBrowser = new Browser();
    }
    iceBrowser.clearCache();
    return iceBrowser;
}
}
Bibliography

