Building Software eCommerce Site

Term Paper

Date : 2001. 6. 14

Department : eCommerce

Member : Chung Sukwon, Lee Ilnam

CONTENTS

I. Introduction

1. Background

The increased popularity and use of the Internet has to be one of the outstanding trends in buying and selling software today. Instead of hosting traditional sales calls or ordering printed brochures, we as customers can now locate and evaluate products using the web.

Not only can we search the web for products, we can evaluate software for a trial period by downloading it with FTP or by ordering an evaluation CD-ROM. We haven't left the office and we haven't had a meeting with anyone about the software, and technical support is still available by phone. All transactions have been at our convenience.

Still without leaving the office, it is possible to purchase the product by issuing a purchase order, and receive via e-mail, an updated electronic license to allow us to continue running our software while the PO is processed. We can even request "floating licenses" to allow all users at a site to share the software.

After a permanent license is issued, we can download updated versions of the software, up to a specific version level, as controlled by the electronic license. A year later, we renew our support contract and get an updated electronic license via e-mail allowing us to download updates for another year.

While this electronic process is efficient for us as customers, it is a major challenge to software companies selling us their products. For starters, how do they keep us from using the software beyond the trial period? How do they know that we have a support contract and can download updates? While electronic commerce is convenient for us, it has to be integrated into the software company's business. Why do they do it?

The reason is simple. Software customers and vendors are busy; they need to get work done. The easier it is to acquire new software and solutions tailored to them, the more inclined they are to buy.

2. Purpose

In this project our team aim to implement a prototype of eCommerce website for software with SSL support and understand issues regarding software ecommerce such as licensing, delivery, payment, and security.

II. Ecommerce for software

1.Introduction

Over the past seven years there has been a quiet technology revolution in selling software electronic licensing. Electronic licensing helps software vendors and customers monitor or control software use at a customer's site. Through this technology, customers receive better and more flexible licensing terms from software vendors. Electronic licensing allows software vendors to market, sell and distribute their products more effectively, while significantly reducing the problem of honest customers unknowingly using software beyond what they licensed or of dishonest users pirating software.

Many innovative software vendors combine electronic licensing with electronic software distribution to create a better way to run a software business (electronic commerce for software). Not surprisingly, these companies link the granting of "electronic licenses" to the customer issuing a purchase order, providing a credit card number or some other means of verifying that purchase of a license has taken place. Electronic commerce for software is not limited to doing business over the world-wide-web. It is also used for CD-ROMs holding entire product lines, where electronic licensing limits access to only authorized users.

2. What can be done with Electronic Commerce for Software?

Electronic commerce for software has parallel technologies and business practices for each of the essential parts of software commerce - licensing, software delivery, and payment. Well executed electronic commerce strategies integrate technology and company specific business practices.

3. Electronic licensing

Electronic licensing precisely defines the product sold. By embedding electronic licensing into a product, software vendors define a customer's license rights in a human readable "license file." Compliance with these license rights is then automatically monitored. The software vendor may choose to block users from running a product if doing so violates the license rights, or simply provide notification to the user or system administrator when license use has exceeded the customer's license rights. This allows customers and vendors to buy and sell software licenses using much more flexible licensing terms than traditional one-computer-one-license or site license approaches.

Some of these flexible licensing models include floating licenses where a specific number of licenses are shared over a network, product suites where several product licenses are combined to be licensed as a single product, and demo licenses where a prospective customer has full functional use of a product, but the right to use expires on a specific date. Electronic licensing technology is based on the following six concepts:

1. Policy in the License - Part I. Developers integrate electronic licensing into a product in a few lines of code without "hard-coding" license policies into their products. This avoids the need to change a product's source code and to support multiple product releases when licensing terms change for a product, a targeted market or an important customer.
2. Policy in the License - Part II. License terms are described in a human readable text file as an "electronic license certificate." This is where marketing, sales, support or order administration define licensing policies, without the need for software engineers to make changes in software.
3. Security by Authentication not Encryption. As the license file contains licensing policies in human readable text, preventing dishonest users from changing licensing terms by modifying the license certificate is critical. Keeping license certificates human readable reduces administrative and support costs as vendors, resellers and customers understand which products have been licensed, without the need for finding the means of decrypting licensing data. Finally, US export control laws severely limit the export of strong encryption technology, making strong encryption inappropriate for products intended for US export. However, very strong authentication may be used and exported in electronic licensing without violation of export laws.
4. Intellectual Property Notices. Software vendors can include information identifying, customer and other purchasing information as part of the license certificate. This is done so that if the software and license were diverted to another company, an "audit trail" is left making discovery of improper use of the license far more likely.
5. Authenticated Report log. Electronic licensing records the use of software licensing into a transaction log called the "Report Log." The information in this log is authenticated and compressed so software vendors and customers can use this information as a basis for a pay-per- use or some other usage based pricing/licensing business model. In addition, customers may use this information to better manage their software assets and to "bill-back" software related costs to different departments or projects in their company.
6. Full Internet Support. The Internet and TCP/IP have become the basis for organizations to build wide area networks. Electronic licensing must be well behaved to work in this

new world. This means the electronic licensing technology cannot use network broadcasts; it must work through bridges and routers without the need for re-configuration of network hardware or software. Network administrators must be allowed to configure firewalls (both filtering router and proxy server approaches) to either permit or prohibit electronic licensing passing through.

Electronic licensing requires no additional hardware, but may work in cooperation with hardware keys.

4. Electronic delivery

This is largely based on well known technologies such as Internet FTP, CD-ROMs and magnetic media. All computer software is delivered by electronic means, but electronic commerce for software adds a control mechanism that limits access to licensed users.

In many instances electronic licensing provides this access control - the user gains access by delivery of an electronic license. Processing a credit card transaction before an FTP download, or before a CD-ROM is sent in the mail is another access control mechanism - payment must be made in advance for a user to gain access.

In other cases CD-ROMs of software products lack electronic licensing but need to be licensed individually. Each product may be in its own encrypted file, so that the customer needs a specific "password" to gain access to a given product.

5. Electronic payment with Web & FTP sites

For today's business-to-business software purchases, making electronic commerce for software work with traditional purchase orders, invoices, accounts payable, receivables and company checks is critical. Essentially all software customers and vendors support these traditional purchasing methods.

The most common way to accommodate these traditional purchasing methods is to provide customers with the complete product with a date limited electronic license during the evaluation phase. The software vendor does not issue a permanent electronic license (by email, FAX or telephone) until after a purchase order (or even payment) has been issued by the customer. Software vendors must continue to support this mode of payment for the foreseeable future, as many software customers will be slow to adopt electronic payment methods.

By using CGI (Computer Gateway Interface) programming, access control and the credit card processing capability available with Netscape's or OpenMarket's web commerce servers, software vendors can build electronic "software stores." In this case, securing payment by credit card in advance of allowing a customer to FTP the software is an effective and simple access control approach. This approach, however, does not monitor or control software use after it is dispensed.

Other methods vendors use include "authorization codes" indicating that "payment" has been made through traditional payment systems or purchasing a shrink wrapped product in a box containing an authorization certificate.

6. CD-ROM and FTP based distribution

A software vendor may place its entire product line on a CD-ROM (or set of CD-ROMs) controlled by electronic licensing. The vendor distributes the same CD to all customers regardless of product purchased. For a customer to use a given product, he must first acquire an electronic license authorizing the use of a particular software product on the CD.

This authorization might be delivered in the form of a paper certificate with authorization codes the user types during software installation. Alternatively, the vendor might e-mail the customer an electronic license certificate authorizing the software to be used.

Significant cost economies result from manufacturing, stocking and distributing a single CD instead on one CD per product. In addition, if an existing customer would like to purchase or evaluate some other product, there is no need to make an additional physical shipment of a CD. The customer could receive an immediate trial or fully paid-up license by fax, phone or e-mail. As existing customers are the people most likely to be future customers, CD-ROMs containing entire product lines are powerful marketing and sales tools.

Making software available from an FTP site is logically quite similar to CD-ROM based distribution. While there are a few instances of major software distribution strategies using the Internet for FTP (File Transfer Protocol), this approach has been successful mainly for products that are smaller than 5 MB, and for the distribution of product updates.

CD-ROM distribution of software continues to be more important than Internet based distribution for software products requiring more than 10 MB for a software installation kit. However, even for very large software installation kits, Internet FTP provides a convenient means to distribute updates, patches and add-on software products. For most software vendors, the most effective

software distribution strategy is to use a mix of CD- ROMs and Internet FTP, with electronic licensing to control access.

7. Electronic Pay-Per-Use

This is an emerging technology for selling software licenses. It is a fundamentally different way to sell software that mixes Electronic Licensing and Electronic Payment. The Internet and Electronic Licensing make it possible to sell access to software based on actual software use, by authenticating the validity of the usage data, automating the collection of data at a given customer site, generating detailed usage reports for the customer and transmitting the billing information via the Internet to the software vendor.

Ironically, while pay-per-use is an emerging license/payment model on the Internet, pay-per-use was widely used in the 1970s with time-sharing mainframe services.

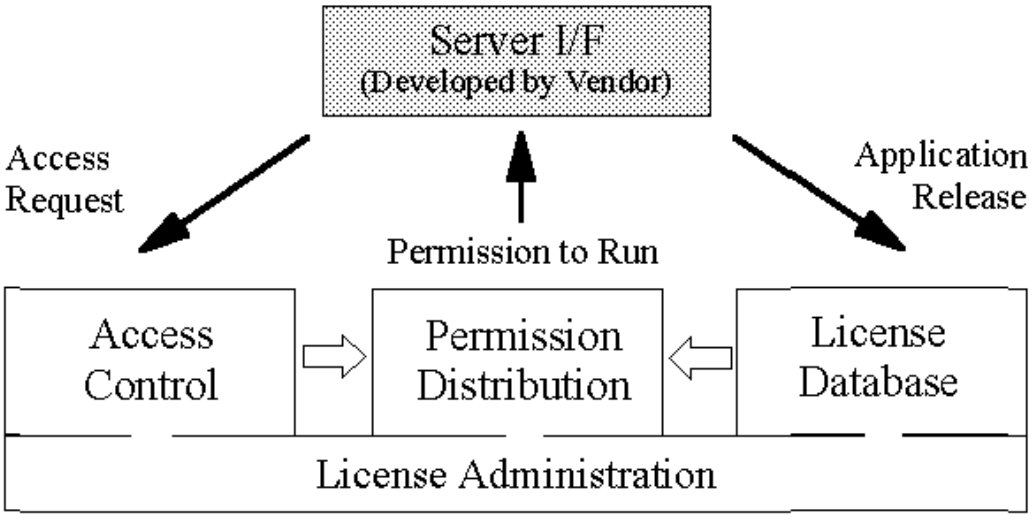
III. Software licensing

1. Conceptual description of a generic license manager

In general terms, a license manager is a system utility-like application that controls or monitors the use of another end-user application. Since the license manager controls the number of application *users*, there is no need to control the number of application *copies*. Therefore, users can move applications quickly and easily between different machines without violating the terms of the license agreement.

All license managers, regardless of implementation, address one basic concept: application usage flexibility. Vendors can maintain control over their revenue stream without "locking" applications to a particular machine or a particular user. With license managers, application buyers have the option of buying less software and sharing it more or buying more software and sharing it less.

License managers come in many different forms. They may have many features other than usage flexibility that will add value to the applications they control. Like any other software product, each server is a unique implementation of the same concept. However, all servers contain the same basic building blocks as illustrated in the figure below.



- *Server Interface* - This module is part of the application and is responsible for communicating with the license manager. The license manager may be accessed by direct calls, by stubs embedded in the application code, or indirectly via another module (e.g., a subroutine).

- *Network communications* --These are messages passed across the network to communicate between the application and the license manager. At least three messages must pass between the application and its license manager:
- A license request to determine license availability;
- A "rights" message that grants or denies permission for a user to run an application (the license counter is decremented if the application is used) and;
- A license release message that increments the license counter when the application is no longer in use.
- *Access Control* - This module contains information on who can access the licenses. CPU-ID, user ID, time, etc., may control access, as determined by the application vendor or the end-user.
- *License Data Base* - This module contains information about the licenses under management. License information may include the software version or a list of the other modules that must be available to run a specific application. This module keeps track of how many copies of an application are in use and when the product was started.
- *Permission Distribution* - This module is responsible for sending permission to the correct application on the correct CPU. A user can start an application after receiving permission from the license manager. Some license manager implementations re-issue permission periodically for security reasons.
- *License Administration* - This module monitors license usage and related transactions as users access applications and check licenses in or out. This module may include utilities for adding more licenses, changing access information, or generating various reports.

2. How License Managers Work

License managers have a client-server architecture, with the client linked to the licensed application. The client provides the license server with information about the application (product name, vendor, version), user name, computer hostid, and the user's display.

If the server grants the client a license "token," the client informs the application whether the application has a valid license. It is up to the application to decide what behavior it should have when it does not have a token.

Besides notifying the user that license tokens are not available, many applications without valid license tokens go into a "demo" mode. One example of such a demo mode is an electronic publishing application that does not allow the saving of documents, and prints "DEMO" across

every printed page. A user could, therefore, "sample" the application, but this demo mode would not be useful for doing real work.

The server's major function is to respond to a client's checkout request for a license token if license tokens are still available. The server acts as an "accountant," and the accounting can become quite complex, because licenses may be reserved for certain groups of users, and an application may consider many invocations of an application by the same user on the same computer as one license (or several licenses). The server can also be used to provide reports on which licenses are being used and by which users.

While the application runs, the client and server periodically send messages to each other. The server or client can therefore know if the application or license server has terminated abnormally. If the application has terminated (due to a server or network failure), the license server records that the license token has been returned.

If the server has terminated, the client will try to reacquire a license from a back-up license server. If this fails, the client notifies the application that it no longer has a license token. Potential behavior changes for the application include:

1) continue running as normal,

2) give a warning that the application will terminate in ten minutes and then terminate, or

3) change the functionality or make the application run slowly.

The "license database," or "license file," holds the information about the license tokens on the server. This information generally uses a Message Authentication Code (MAC) to protect the licensing data from being changed by other than the software vendor or its resellers. The MAC is essentially a sophisticated checksum of the information. It is not encrypted data.

As additional licenses of products or new products are acquired, new entries are placed into the license file.

3. License Management: How developers control software licensing

When the United States of America ratified the U.S. Constitution in 1788, Congress received the power "To promote the Progress of Science and useful Arts, by securing for limited Times to Authors and Inventors the exclusive Right to their respective Writings and Discoveries." Upon

this legal foundation rest the rights of US companies to buy and sell software. What is generally known as "selling software" is really selling a limited license to use and copy the software. I apologize for this nationalistic point of view to my non-U.S. readers. Some software includes patented technology. However, most software has only copyright protection.

The first concepts of software licensing developed in the days of non-networked computing and million dollar "low-cost" computers. These concepts included:

1. licenses are fixed to a particular CPU,
2. prices of software licenses increase as the computer's performance increase, and
3. limiting the number of copies of software made to disk or tape is important.

When PCs became popular in the early 1980s, PC software vendors kept this earlier view of licensing software - with the exception of pricing based on CPU performance. The rapid evolution and market acceptance of networked computing in the late 1980s forced a rethinking of what software licensing means on a network of workstations, diskless workstations, servers, X-terminals and ASCII terminals.

Honest users want to live within the terms they purchased their licenses under. The old concept of software licenses allowed system administrators to keep track of software by limiting the copies of software to hard disks that were directly attached to licensed CPUs -- networked file systems made this view of licenses obsolete.

Workstation customers called for new concepts of software licensing:

1. software is a network resource -- licenses "float" on the network,
2. software costs are a function of how many users simultaneously run the software -- pricing is based on users, not CPU performance,
3. value exists in the use software, not in the number of copies on disk or tape.

The new concepts of licensing require dynamic tracking of users and software licenses. The technology that does this is called a "license manager." Without license management technology, it's impractical for a system administrator with a network of even ten workstations to keep track of how many floating licenses are used concurrently.

License management differs from software "copy protection" used in the PC marketplace by controlling execution instead of copying. PC users rebelled against copy protection because the practice inconveniences honest customers while giving them nothing in return.

In contrast, UNIX customers pressure software developers to provide them with license management because it makes software a network resource and makes software pricing more equitable.

Buying more licenses is easier under license management because it eliminates the need to send out another tape with the software. All that the user requires is a "license key" -- a brief string of data that describes the license plus some security codes. These license keys can be distributed by e-mail, FAX or verbally over the telephone.

Today, most major vendors of workstation software use license managers to control their products.

4. Licensing Policies

License managers embody the licensing terms of a developer into software and datafiles. To understand license managers, it is critical to understand the types of licensing policies used by software developers.

- *Node-locking,* once the most popular form of licensing, parallels the view that software is licensed to a particular computer. Typically, users may remotely login to the licensed computer. This licensing model is typically found in computationally intense applications and with software used on workstations dedicated to a particular application.
- *User-based licensing* assigns licenses to a specific user-id. This is useful for products that are user dependent, e.g., an e-mail product, or business transaction applications -- applications in which "lending" someone a user-id conflicts with the very nature of the application.
- *Site licensing* software to a geographical site is another approach, one that generally has a high entry price for the customer. Both the software vendor and customer frequently spend a great deal of time negotiating over price, as site licenses generally don't match pricing to actual usage very well. Defining a "site" can also be difficult. For example, a company with a large corporate campus may be so large that multiple sites are defined. Site licenses are most appropriate when companies standardize on a specific product.
- *"Network licensing" and "floating licenses"* are almost synonymous with license management. Floating licenses fitted very well with the concepts of networked computing. Floating licenses became the prevailing workstation licensing policy due to its efficient matching of usage to the number of licenses sold. Many customers at a site

can have access to a software product, but the cost of this access may be the price of one license. As the software becomes widely used, additional licenses are purchased.

Other licensing policies include specifying an "expiration date" or a "start date" for licenses, used in software evaluations or leasing programs.

There is no "right" licensing policy for all products. Developers decide policies based upon the way the software product is designed and used, and on competitive factors. Sometimes the same software may be licensed with two very different licensing models.

For example, an electronic publishing package may be available as either a floating license or a node-locked license, with the node locked license having a lower price.

License management helps users choose the right software, when software vendors implement extensive customer evaluation programs. This allows users to evaluate a software product for a few weeks before it is purchased.

Users at a site tend to "standardize" on a particular software product that is license managed software because the product is a networked resource that can be used by all, not just those who have a license for their workstation.

System administrators also have tools to control the use of licenses. For example, administrators may make licenses available only to specific groups, or specific users.

5. General Design Goals

The general design goals of a practical license manager may at first seem simple. However, because license management becomes an essential service on a network, such goals can become quite complex.

License managers must be very robust. Back-up license servers are essential. If the license server fails, users cannot run the software they purchased. That is not a graceful failure. High availability needs to be designed-in to accommodate network failures as well.

Finally, as servers go down or networks fail, the license manager must automatically configure itself without system administrators taking action -- her hands are already full getting the server or network working again.

The license manager must be flexible enough to allow the product to be licensed in a way that makes sense to customers and software vendors. The license manager must understand the issues presented by X-terminals, multiprocessor computers, products that can be licensed as floating or node locked licenses.

Although the concept of licensing by "user" is well accepted, there is not as much consistency about what is a license "use." For example, only some software licenses allow a person to run more than one "copy" of the software on the same display on the same workstation with the same user-id. The "right" answer varies with the product being licensed. The license manager must allow for different interpretations of what is a "use" of a license.

Most networks are heterogeneous, supporting workstations from SUN, HP, IBM, DEC, Silicon Graphics, etc. Moreover, many applications run on several different workstations. The license manager should allow licenses for SUN applications to be shared with the HP and IBM workstations - if the software developer so chooses.

Most workstations today are on large networks, so license managers may be called upon to serve hundreds or thousands of licenses at one time. The license manager's client-server processing must not place a high burden on the network by scaling linearly and allowing multiple license servers.

6. Buy or Develop -- a License Manager

Elsewhere in this report we try to simplify the conceptual description of a license manager or software metering program. In reality, these products are very tricky. They rely on low-level systems calls and include well thought out functions, reasonable documentation, and support (typically). They are also inexpensive.

Although most vendors don't want to place the family jewels (sales revenues, in this case) in the hands of an outsider, we encourage you to do so. Here are our reasons:

- Your development team should stick to your business: adding functions your customers want in your products.
- If you develop it, you own it -- and the maintenance, and the testing, and the bug fixing, and the documentation...
- These products are not "a neat hack" or trivial. Some developers of license managers were people familiar with highly secure environments and operating systems.
- License fees are inexpensive compared to the cost of your people involved in

development, support, and documentation.

- Thousands of users have tested most commercial license managers for functionality and reliability.
- Every vendor that has developed their own license manager has regretted doing so and has licensed a "real" product.

VI. General Guide of Web Site for Software Publisher

1. potential audiences

- Current customers
- Potential customers
- Partners
- Investors
- Employees
- Potential employees
- Your competition

2. Base Marketing Requirements

- Home page as positioning
- Product benefits as lead item
- Comprehensive technical information
- Integrated, useful search
- Synchronization with current marketing efforts
- Clear contact information

3. Moving Toward the Top Tier

- Evaluation or demo products
- Links to reviews
- Links to partners
- Comparison charts
- Promotions
- E-mail list sign-ups
- Awareness of International customers

V. Development of Prototype of eCommerce site for software

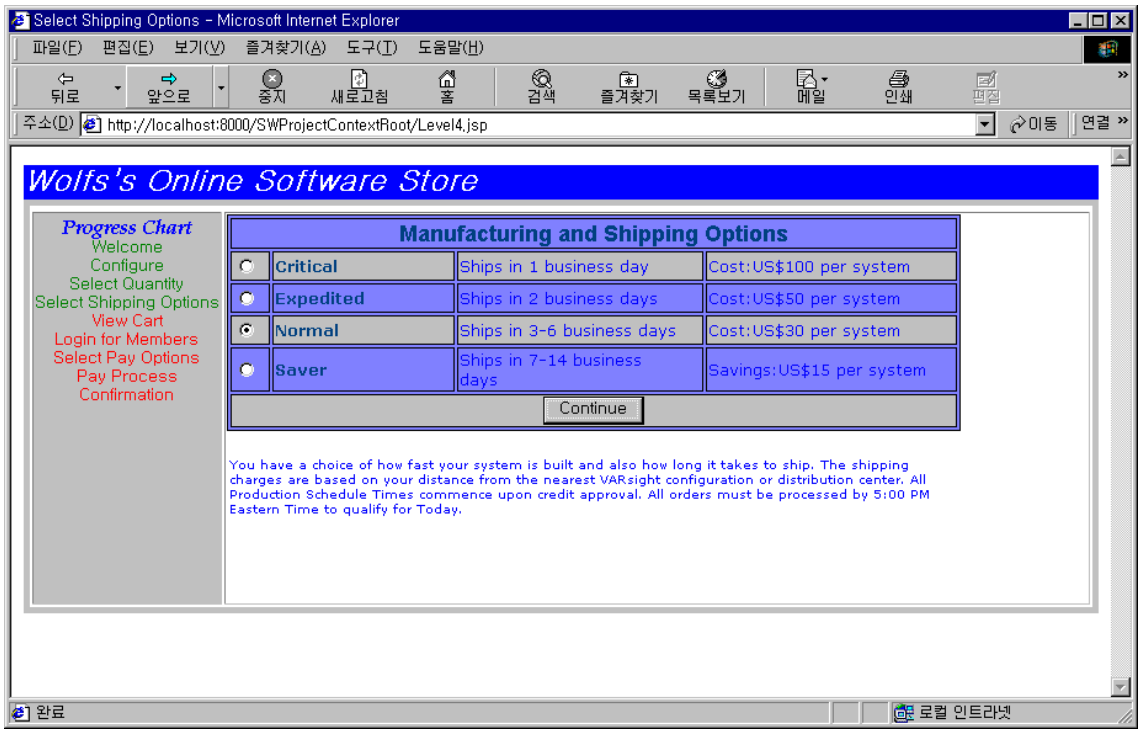
1. Tools

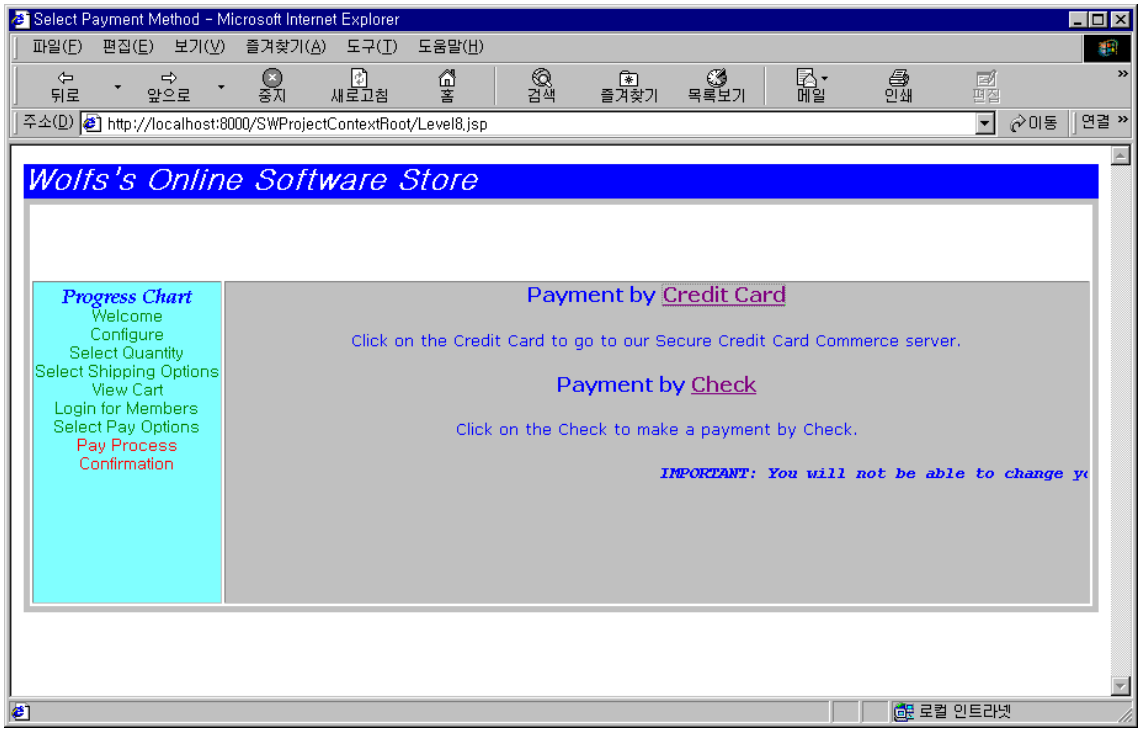
- JDK Enterprise Edition 1.2.1
- JSP
- JDBC
- Cloudscape DB

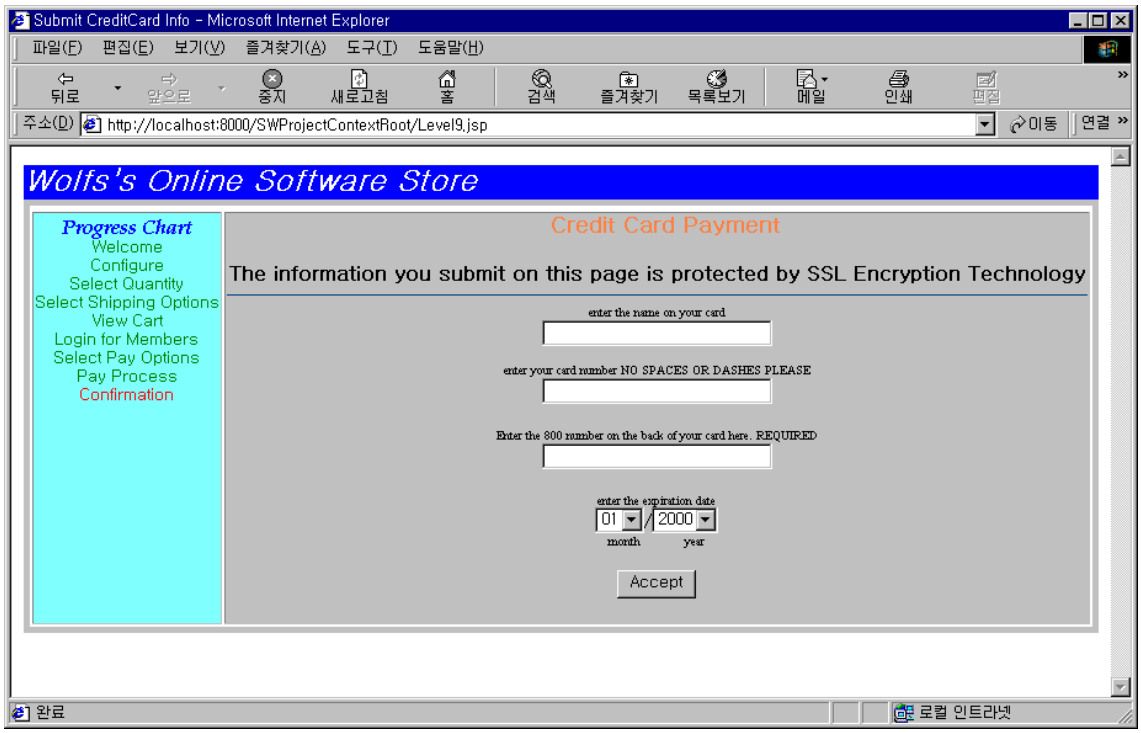
2. SSL Implementation

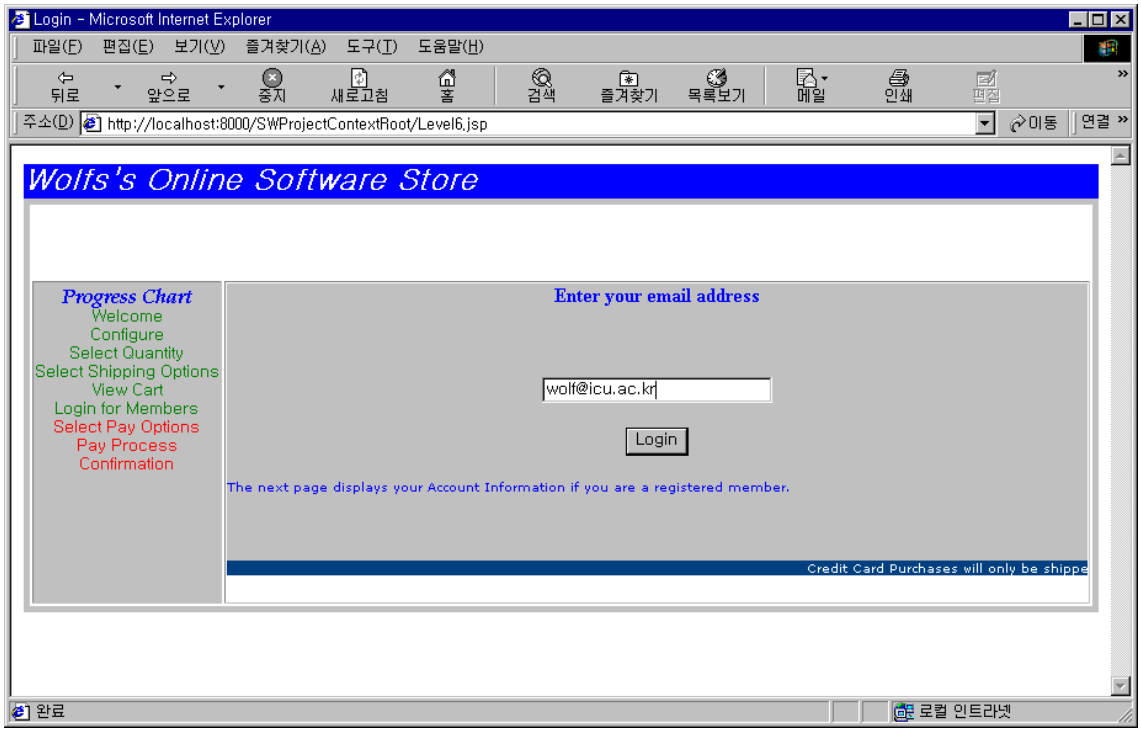
Verisign Test CA

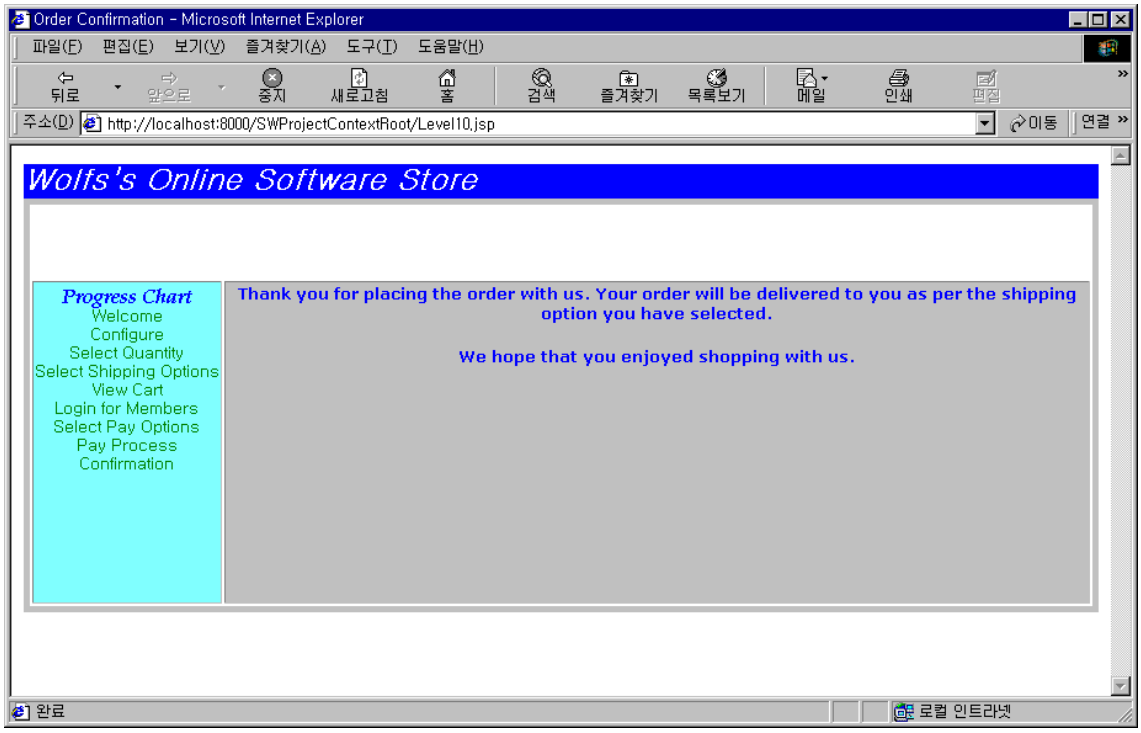
3. Screen Examples


Select Shipping Options - Microsoft Internet Explorer
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
뒤로 앞으로 중지 새로고침 홈 검색 즐겨찾기 목록보기 메일 인쇄 편집
주소(D) http://localhost:8000/SWProjectContextRoot/Level4.jsp
이동 연결
Wolfs's Online Software Store
Progress Chart
Welcome
Configure
Select Quantity
Select Shipping Options
View Cart
Login for Members
Select Pay Options
Pay Process
Confirmation
Manufacturing and Shipping Options
Critical
Ships in 1 business day
Cost:US$100 per system
Expedited
Ships in 2 business days
Cost:US$50 per system
Normal
Ships in 3-6 business days
Cost:US$30 per system
Saver
Ships in 7-14 business days
Savings:US$15 per system
Continue
You have a choice of how fast your system is built and also how long it takes to ship. The shipping charges are based on your distance from the nearest VARsight configuration or distribution center. All Production Schedule Times commence upon credit approval. All orders must be processed by 5:00 PM Eastern Time to qualify for Today.
완료
로컬 인트라넷


Select Payment Method - Microsoft Internet Explorer
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
뒤로 앞으로 중지 새로고침 홈 검색 즐겨찾기 목록보기 메일 인쇄 편집
주소(D) http://localhost:8000/SWProjectContextRoot/Level8.jsp
이동 연결
Wolfs's Online Software Store
Progress Chart
Welcome
Configure
Select Quantity
Select Shipping Options
View Cart
Login for Members
Select Pay Options
Pay Process
Confirmation
Payment by Credit Card
Click on the Credit Card to go to our Secure Credit Card Commerce server.
Payment by Check
Click on the Check to make a payment by Check.
IMPORTANT: You will not be able to change yo
로컬 인트라넷


Submit CreditCard Info - Microsoft Internet Explorer
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
뒤로 앞으로 중지 새로고침 홈 검색 즐겨찾기 목록보기 메일 인쇄 편집
주소(D) http://localhost:8000/SWProjectContextRoot/Level9.jsp
이동 연결
Wolfs's Online Software Store
Progress Chart
Welcome
Configure
Select Quantity
Select Shipping Options
View Cart
Login for Members
Select Pay Options
Pay Process
Confirmation
Credit Card Payment
The information you submit on this page is protected by SSL Encryption Technology
enter the name on your card
enter your card number NO SPACES OR DASHES PLEASE
Enter the 800 number on the back of your card here. REQUIRED
enter the expiration date
01 / 2000
month year
Accept
완료
로컬 인트라넷


Login - Microsoft Internet Explorer
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
뒤로 앞으로 중지 새로고침 홈 검색 즐겨찾기 목록보기 메일 인쇄 편집
주소(D) http://localhost:8000/SWProjectContextRoot/Level6.jsp
이동 연결
Wolfs's Online Software Store
Progress Chart
Welcome
Configure
Select Quantity
Select Shipping Options
View Cart
Login for Members
Select Pay Options
Pay Process
Confirmation
Enter your email address
wolf@icu.ac.kr
Login
The next page displays your Account Information if you are a registered member.
Credit Card Purchases will only be shippe
완료
로컬 인트라넷


Order Confirmation - Microsoft Internet Explorer
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
뒤로 앞으로 중지 새로고침 홈 검색 즐겨찾기 목록보기 메일 인쇄 편집
주소(D) http://localhost:8000/SWProjectContextRoot/Level10.jsp
이동 연결
Wolfs's Online Software Store
Progress Chart
Welcome
Configure
Select Quantity
Select Shipping Options
View Cart
Login for Members
Select Pay Options
Pay Process
Confirmation
Thank you for placing the order with us. Your order will be delivered to you as per the shipping option you have selected.
We hope that you enjoyed shopping with us.
완료
로컬 인트라넷

Appendix I (Patents of Software Licensing)

Patent # 5671412
License management system for software applications

Inventor: Cristiano; Matt, Saratoga, CA

Assignees: Globetrotter Software, Incorporated, San Jose, CA

Issued: Sep. 23, 1997

Filed: July 28, 1995

ABSTRACT: An improved software license management system in accordance with the present invention is disclosed. A license server initializes a license database by receiving a package license description that includes component license descriptions for component software products in a package. Licenses for software products are also received, and license records are created in the license database for components and suite packages, where each record includes a number of licenses available to be checked out. A client computer system can request a license for a component product in a package. A license is granted to the client when the client is allowed to receive the license according to a license policy. When a component license is checked out, a linked suite license is also automatically checked out. No other client thus may use a component license linked with the suite license record unless another suite license is checked out. The license management system also provides a number of modifiers to be included in license records, including an overdraft quantity, a fail safe indicator, a minimum license quantity, and a capacity indicator. A finder and a diagnostic process can be implemented at the client computer system to find the license server over a network and provide a tool to diagnose failures in the license management system.

Patent # 5390297

System for controlling the number of concurrent copies of a program in a network based on the number of available licenses

Inventors: Barber; Jon H., Santa Barbara, CA
Woodward; Ronald A., Boulder, CO
Burkley; Richard M., Boulder, CO
Rehme; Erwin L., Longmont, CO
Jackson; Matthew W., Boulder, CO
Young; Douglas M., Boulder, CO
Assignees: Auto-trol Technology Corporation, Denver, CO
Issued: Feb. 14, 1995
Filed: Nov. 10, 1987

ABSTRACT: License management systems and methods allow licenses for a computer program to be available for use at each of a plurality of nodes of a network. If a valid license file at a local node contains an unexpired, available license, a license manager at the local node permits the computer program to be executed at the requesting local node. If no such license is available in a valid license file at such local node, the license manager searches the other nodes for a valid license file containing an unexpired, available license. In one embodiment, if an unexpired available license is located in a valid license file at a second (or "remote") node, the license manager transfers such license to the local node, and assigns and encrypts a unique identification to such transferred license. The original record of the transferred license is modified by erasing it from the license file at the remote node so that the transferred license is no longer available there. In a second embodiment, the license manager modifies the license file to indicate use of the license at the local node without such transfer. The number of copies of the computer program that are authorized for execution simultaneously on the network is thus limited to the number of licenses that have been loaded into the license files on the network.

Appendix II(Source Codes)

```
/* AccountInfoBean.java */

import java.io.*;
import java.util.*;
import java.beans.*;
import java.sql.*;

/** AccountInfoBean class retrieves the customer information from the customer
table. The customer is selected by the e-mail ID */
public class AccountInfoBean
{
        private Connection con;
        private ConnectionManager conMan;
        private String sql;
        private Statement stmt;
        private ResultSet rs;
        private String Name, Company, Address, City, State, Pin, Phone, Fax;
        /** Obtain a connection to the database */
        public Connection connect() {
        try {
                        conMan = new ConnectionManager();
                con = conMan.logOn();
                }
        catch(Exception ex) {
                        ex.printStackTrace();
                }
                        return con;
        }

        /** Disconnect from the database */
        public void disconnect() {
        try {
                        conMan.logOff();
                }
                catch(Exception ex){
                        ex.printStackTrace();
```

```
                }
        }

        /** Retrieve the customer information from the CUSTOMER table */
        public void displayInfo(String Email) {
        try {

                String sql = "Select NAME, COMPANY, ADDRESS, CITY, STATE, PIN,
PHONE, FAX " +
                        "From CUSTOMER Where EMAIL = '" + Email + "'";
                System.out.println("sql = " + sql);
                stmt = con.createStatement();
                rs = stmt.executeQuery(sql);
        if(rs != null)
        {
                        rs.next();
                Name = rs.getString(1);
                Company = rs.getString(2);
                Address = rs.getString(3);
                City = rs.getString(4);
                State = rs.getString(5);
                        Pin = rs.getString(6);
                        Phone = rs.getString(7);
                        Fax = rs.getString(8);
        }
                        disconnect();
      }
        catch(Exception ex)
    {
                        ex.printStackTrace();
    }
 }

public String getName() {

                        System.out.println("name= " + Name);
```

```
                return Name;
 }

public String getAddress() {

                return Address;
 }

public String getCity() {

                return City;
 }

public String getState() {

                return State;
 }

public String getPin() {

                return Pin;
 }

public String getPhone() {

                return Phone;
 }

public String getFax() {

                return Fax;
 }

public String getCompany() {

                return Company;
```

```
  }
}
```

```
/* AssembleBean.java */
import java.io.*;
import java.util.*;
import java.beans.*;
import java.sql.*;

/** AssembleBean retrieves the Item description from INVENTORY table. The caller calls
getItemData method to retrieve the item description in a Vector */

public class AssembleBean {
        private Connection con;
        private ConnectionManager conMan = new ConnectionManager();
        private Statement stmt;
        private ResultSet rs;
        private Vector v;

   /** Connect to the database */
        public void connect() {
                try {
                        con = conMan.logOn();
                        }
                catch(Exception ex){
                        ex.printStackTrace();
                        }
        }

   /** Disconnect from the database */
        public void disconnect() {
                try {
                                conMan.logOff();
                        }
                catch(Exception ex){
                        ex.printStackTrace();
                        }
        }
```

```
   /** Retrieve description of the item from the inventory table and return to
   the caller in a vector */
          public Vector getItemData(String Item) {
                    String SQL;
                    try {
                              SQL = "SELECT DESCRIPTION FROM INVENTORY WHERE
NAME = '" + Item + "'";
                              v = new Vector();
                              stmt = con.createStatement();
                              rs = stmt.executeQuery(SQL);
                              if(rs != null) {
                                        while(rs.next()) {
                                        String str = rs.getString(1);
                                        v.addElement(str);
                                        }
                              }
                              }
                    catch(Exception ex){
          ex.printStackTrace();
                              }
                    return v;
   }
}
```

```
/* ConnectionManager.java */
import java.sql.*;

/** ConnectionManager class loads the jdbc driver and provides methods for
connecting and disconnecting to the database */
public class ConnectionManager {
      protected Connection con;
      protected String driver = "COM.cloudscape.core.RmiJdbcDriver";
      protected String url = "jdbc:cloudscape:rmi:OnlinePCDB";

/** Load the driver and make connection to the database */
      public Connection logOn(){
            try {
                                    Class.forName( driver ).newInstance();
                                    con = DriverManager.getConnection( url, "", "" );
                              }
            catch(Exception ex){
                                    ex.printStackTrace();
                              }
            return con;
      }

/** Close the database connection */
            public void logOff(){
                  try {
                                    con.close();
                  }
            catch(Exception ex){
                                    ex.printStackTrace();
                  }
      }
 }
```

```
/* GlobalVarBean.java */
import java.beans.*;
import java.util.*;

/**	The GlobalVarBean declares two variables
	ProgressLevel that tracks the level of shopping progress for the customer and
	cart Vector that is used for storing shopping contents */
public class GlobalVarBean {
   private int ProgressLevel;
   private Vector cart = new Vector();

   public void setLevel(int level) {
	ProgressLevel = level;
   }

   public int getLevel() {
      return ProgressLevel;
   }

   public Vector getCart() {
	return cart;
   }

   /** Add the configured PC item to the cart */
   public void addToCart(OrderItemBean OrderItem) {
	cart.addElement(OrderItem);
   }
}
```

```
/* OrderItemBean.java */
import java.util.*;
import java.beans.*;
import java.sql.*;
import java.io.*;
import java.util.Date;

/** OrderItemBean declares variables for storing order details for each
ordered PC. The class provides a utility method called calcPCPrice that
computes the item price based on the selections */

public class OrderItemBean{
   private String Speed, MotherBoard, Monitor, HD, CD, Video, RAM, Card ;
   private OrderUtil ordUtil;
   private double ItemPrice=0;
   private int Qty=1;

   /** Computes the price for the current configuration */
   public void calcPCPrice(){
            ordUtil = new OrderUtil();
            ordUtil.connect();
            ItemPrice = ordUtil.calcPrice(Speed);
            ItemPrice += ordUtil.calcPrice(MotherBoard);
      ItemPrice += ordUtil.calcPrice(HD);
      ItemPrice += ordUtil.calcPrice(RAM);
      ItemPrice += ordUtil.calcPrice(Monitor);
      ItemPrice += ordUtil.calcPrice(Video);
      ItemPrice += ordUtil.calcPrice(CD);
      ItemPrice += ordUtil.calcPrice(Card);
      ordUtil.disconnect();
   }

   public int getQty(){
            return Qty;
   }
```

```
public void setQty(int Qty){
        this.Qty = Qty;
}

public double getPrice(){
        return ItemPrice;
}

public String getSpeed(){
        return Speed;
}

public void setSpeed(String Speed){
        this.Speed = Speed;
}

public String getMother(){
   return MotherBoard;
}

public void setMother(String MotherBoard){
   this.MotherBoard = MotherBoard;
        }

public String getHD(){
   return HD;
}

public void setHD(String HD){
   this.HD = HD;
}

public String getRAM(){
        return RAM;
}
```

```
    public void setRAM(String RAM){
            this.RAM = RAM;
    }
       public String getMonitor(){
       return Monitor;
    }
    public void setMonitor(String Monitor){
       this.Monitor = Monitor;
    }

    public String getVideo(){
            return Video;
    }

    public void setVideo(String Video){
            this.Video = Video;
    }

    public String getCD(){
       return CD;
    }

    public void setCD(String CD){
       this.CD = CD;
    }

    public String getCard(){
            return Card;
    }

    public void setCard(String Card){
            this.Card = Card;
    }
}
```

```
/* OrderUtil.java */
import java.sql.*;

/** OrderUtil class is an utility class that provides a method called calcPrice which
returns the price for the selected Item */
public class OrderUtil{
        private Connection con;
        private ConnectionManager conMan;

  /** Connect to the database */
        public void connect() {
                try {
                                conMan = new ConnectionManager();
                                con = conMan.logOn();
    }
                catch(Exception ex){
                                ex.printStackTrace();
    }
  }

  /** Disconnect from the database */
  public void disconnect() {
        try {
                conMan.logOff();
        }
        catch(Exception ex){
                ex.printStackTrace();
        }
  }

  /** Determines the price of the item from the inventory database and returns
  to the caller */
  public double calcPrice(String Item){
        double UnitPrice=0;
        try {
                String sql = "Select UNIT_PRICE from INVENTORY where DESCRIPTION =
```

```
'" + Item + "'";
                        Statement stmt = con.createStatement();
                        ResultSet rs = stmt.executeQuery(sql);
                        rs.next();
            String Price = rs.getString(1);
                        UnitPrice = Double.parseDouble(Price);
        }
        catch(Exception e){
            System.out.println("Exception in totalPrice(): " +e.getMessage());
        }
        return UnitPrice;
    }
}
```

APPENDIX I (Patents of Software Licensing)

```
<%-- WolfHome.jsp --%>
<%-- Displays the home page to the user, follows the template with Banner
at the top and Progress chart on the left --%>
<html>
<%@ page language="java"    import="java.util.*" %>

<jsp:useBean id="GlobalBean" scope="session" class="GlobalVarBean" />
<jsp:setProperty name="GlobalBean" property="level" value="1" />

<html>
<head>
        <title>Wolf's online Software shopping</title>
</head>

<body>

<table border="5" width="100%" style="border-style: solid" height="336">
        <tr bgcolor="#0000FF" text="#FFFFFF" colspan="2" valign="middle">
                        <%@ include file="Banner.html" %>
        </tr>
        <tr>
                <td width="18%" height="262" bgcolor="#66FFFF" valign="top">
        <jsp:include page="ProgressChart.jsp" flush="true" />
    </td>
    <td width="82%" height="262" bgcolor="#0000FF">
        <%@ include file="Main.html" %>
    </td>
        </tr>
</table>

</body>

</html>
```

```
<%-- AccountInfo.jsp --%>
<%-- Displays the account information of the user specified by the e-mail parameter --%>
<html>

<%@ page language="java" %>

<jsp:useBean id="AccountBean" scope="page" class="AccountInfoBean" />

<%-- Connect to the database, retrieve the account information for the
user selected by e-mail parameter and disconnect from the database --%>
        <%
                AccountBean.connect();
                String email = request.getParameter("email");
                AccountBean.displayInfo(email);
                AccountBean.disconnect();
        %>

<html>
<head>
        <title>Enter or Validate Customer Information</title>
</head>

<body>
<form method="POST" action = "Level8.jsp" target="_self">

<%-- Create an HTML table to display the account information --%>
<table border="0"    cellspacing="0" cellpadding="0" bgcolor="#C0C0C0" width="100%">
  <tr>
                <td valign="top" align="center" height="24">
                        <p      align="center"><font      size="4"      color="#0000FF">Account
Information</font>
                </td>
  </tr>
  <tr>
                <td valign="top" align="left" height="19">
                <p align="center"><font color="#0000FF" face="Arial"><small>
```

```
                    This Information remains confidential and your privacy is assurred.
                    </small> </font>
                </td>
  </tr>
</table>

<%-- In each table row, display the title and the corresponding field value by
                    calling the getXXX method on the AccountBean. --%>

<table border="0" cellpadding="0" cellspacing="0" bgcolor="#C0C0C0" width="100%">
  <tr>
                <td align="right" height="25" width="50%"><small><strong>
                <font color="#0000FF" face="Arial">Name: </font></strong></small>
                </td>
                <td height="25" align="left" width="50%"><font color="#0000FF">
                <%= AccountBean.getName() %>
                </td>
  </tr>
  <tr>
                <td align="right"    height="25" width="50%"><small><strong>
                <font                                                    color="#0000FF"
face="Arial">Company: </font></strong></small>
                </td>
                <td height="25" align="left" width="50%"><font color="#0000FF">
                <%= AccountBean.getCompany() %>
                </td>
   </tr>
   <tr>
                <td align="right" height="25" width="50%"><small><strong>
                <font color="#0000FF" face="Arial">Address: </font></strong></small>
                </td>
                <td height="25" align="left" width="50%"><font color="#0000FF">
                <%= AccountBean.getAddress() %>
                </td>
   </tr>
   <tr>
```

```
                    <td align="right"    height="25" width="50%"><strong><small>
                    <font color="#0000FF" face="Arial">City: </font></small></strong>
                    </td>
                    <td    height="25" align="left" width="50%"><font color="#0000FF">
                    <%= AccountBean.getCity() %>
                    </td>
    </tr>
    <tr>
                    <td align="right"    height="26" width="50%"><strong><small>
                    <font                                                color="#0000FF"
face="Arial">State/Province: </font></small></strong>
                    </td>
        <td height="26" align="left" width="50%"><font color="#0000FF">
                    <%= AccountBean.getState() %>
        </td>
    </tr>
    <tr>
                    <td align="right" height="25" width="50%"><strong><small>
                    <font                color="#0000FF"                face="Arial">Zip/Postal
Code: </font></small></strong>
                    </td>
                    <td height="25" align="left" width="50%"><font color="#0000FF">
                    <%= AccountBean.getPin() %>
                    </td>
    </tr>
    <tr>
                    <td align="right" height="16" width="50%"><strong><small>
                    <font color="#0000FF" face="Arial">Home/Office</font></small>
                    <font                                                color="#0000FF"
face="Arial"><small>Phone</small>: <br></font></strong>
                    </td>
                    <td height="16" align="left" width="50%"><font color="#0000FF">
                    <%= AccountBean.getPhone() %>
                    </td>
     </tr>
     <tr>
```

```
                    <td align="right" height="25" width="50%"><small><strong>
                    <font color="#0000FF" face="Arial">Fax: </font></strong></small>
                    </td>
                    <td height="25" align="left" width="50%"><font color="#0000FF">
                    <%= AccountBean.getFax() %>
                    </td>
    </tr>
</table>

<%-- Display a scrolling text message to the user --%>
        <p align="center"><strong>
        <font color="#004080"><small>
        <marquee bgcolor="#004080" style="color: #FFFFFF">Credit Card orders
        will only be shipped to the billing address of the credit card.</marquee>
        </small></font></strong>
        <p align="center"><input type="submit" value="Select Payment Options" name="B1">
</form>
</body>
</html>
```

```
<%-- Assemble.jsp --%>
<%-- Displays the various items from the inventory database to the user. Each
item is displayed in a drop-down list box --%>
<html>

<%@ page language="java" import="java.util.*,java.sql.*" %>
<jsp:useBean id="abean" scope="page" class="AssembleBean" />

<%-- Connect to the database and declare variables --%>
<%
   abean.connect();
   Vector v;
   Iterator it;
%>

<html>
<head>
        <title>Configure Your Desktop</title>
</head>

<body bgcolor="#FFFFFF">

        <p align="center">
                <img border="0" src="desktop.gif" width="198" height="168">
        </p>
        <p align="center">
                <b><i><font face="Garamond" size="6" color="#3333CC">Configure Your
Desktop</font></i></b>
        </p>

        <form method=POST action="Level3.jsp" target="_self">

<%-- Create an HTML table to display the various component options --%>
<table border="1" cellspacing="1" width="100%">
        <tr bgcolor="#9999FF">
                <td><font color="#3333CC">
```

```
                    <font face="Courier New"><b>CPU</b></font>
                    <%-- The getItemData method returns a vector containing the
different
                                        CPU types in our inventory --%>
                    <%
                              v = abean.getItemData("cpu");
                    %>

                    <%-- Create an Iterator on the vector and add the retrieved items to
                                        the drop down list box --%>
                    <p style="margin-top: 0; margin-bottom: 0">
                    <select name="speed" size="1" style="color: #000080">
                    <%
                              it = v.iterator();
                              while (it.hasNext()) {
                                        String itnew = (String)it.next(); %>
                              <option> <%= itnew %></option>
                    <% } %>
                    </select></font></p>
          </td>

          <td><font color="#3333CC">
                    <%-- Retrieve and add the mother board types to the list box --%>
                    <font face="Courier New"><b>MotherBoard</b></font>
                    <%
                              v = abean.getItemData("mother");
                    %>
                    <p style="margin-top: 0; margin-bottom: 0">
                    <select name="mother" size="1" style="color: #000080">
                    <%
                              it =v.iterator();
                              while(it.hasNext()){
                              String itnew = (String)it.next(); %>
                              <option> <%= itnew %></option>
          <% } %>
                              </select></font></p>
```

```
		</td>

		<td><font color="#3333CC">
<%-- Retrieve and add the hard drive types to the list box --%>
			<font face="Courier New"><b>HardDisk</b></font>
			<%
				v=abean.getItemData("hard");
			%>
			<p style="margin-top: 0; margin-bottom: 0">
			<select name="hard" size="1" style="color: #000080">
			<%
				it =v.iterator();
				while(it.hasNext()){
					String itnew = (String)it.next(); %>
				<option> <%= itnew %></option>
			<% } %>
				</select></font></p>
		</td>
</tr>
<tr bgcolor="#9999FF">
		<%-- Retrieve and add the RAM types to the list box --%>
		<td><font color="#3333CC">
			<font face="Courier New"><b>RAM</b></font>
			<%
				v=abean.getItemData("ram");
			%>
				<p style="margin-top: 0; margin-bottom: 0">
				<select size="1" name="ram" style="color: #000080">
			<%
				it =v.iterator();
				while(it.hasNext()){
					String itnew = (String)it.next(); %>
				<option> <%= itnew %></option>
			<% } %>
				</select></font></p>
```

```
		</td>

		<td><font color="#3333CC">
<%-- Retrieve and add the monitor types to the list box --%>
			<font face="Courier New"><b>Monitor</b></font>
			<%
				v=abean.getItemData("monitor");
			%>
			<p style="margin-top: 0; margin-bottom: 0">
			<select name="monitor" size="1" style="color: #000080">
			<%
				it =v.iterator();
				while(it.hasNext()){
					String itnew = (String)it.next(); %>
				<option> <%= itnew %></option>
			<% } %>
				</select></font></p>
		</td>

		<td><font color="#3333CC">
<%-- Retrieve and add the video board types to the list box --%>
			<font face="Courier New"><b>Video Card</b></font>
			<%
			  v=abean.getItemData("video");
			%>
			<p style="margin-top: 0; margin-bottom: 0">
			<select size="1" name="video" style="color: #000080">
			<%
				it =v.iterator();
				while(it.hasNext()){
					String itnew = (String)it.next(); %>
				<option> <%= itnew %></option>
			<% } %>
				</select></font></p>
		</td>
</tr>
```

```
<tr bgcolor="#9999FF">
        <%-- Retrieve and add the CD-ROM types to the list box --%>
        <td><font color="#3333CC">
                <font face="Courier New"><b>CD-ROM Drive</b></font>
                <%
                        v=abean.getItemData("cdrom");
                %>
                        <p style="margin-top: 0; margin-bottom: 0">
                        <select size="1" name="cd" style="color: #000080">
                <%
                        it =v.iterator();
                        while(it.hasNext()){
                                String itnew = (String)it.next(); %>
                        <option> <%= itnew %></option>
                <% } %>
                        </select></font></p>
        </td>

        <td><font color="#3333CC">
<%-- Retrieve and add the network card types to the list box --%>
                <font face="Courier New"><b>Network Card</b></font>
                <%
                        v=abean.getItemData("network");
                %>
                        <p style="margin-top: 0; margin-bottom: 0">
                        <select size="1" name="card" style="color: #000080">
                <%
                        it =v.iterator();
                        while(it.hasNext()){
                                String itnew = (String)it.next(); %>
                        <option> <%= itnew %></option>
                <% } %>
                        </select></font></p>
        </td>
</tr>
```

```
        </table>

<%-- Display a scrolling text message to the user --%>
                        <p align="center"> <font face="Courier" size="1"><b>
                                <marquee style="color: #FFFFFF" bgcolor="#0000FF"
width="500" height="13">Standard configuration
                                includes 1.44MB FDD, 104 Key Keyboard, Mouse, SMPS,
Cabinet. </marquee>
                        </b></font>
                        </p>

                        <p align="center"><font color="#0000FF"><input type="submit"
value="Accept" name="submit">
                        </font></p>

        </form>

        <%-- Close the database connection --%>
                <%
                        abean.disconnect();
                %>

</body>
</html>
```

```
<%-- CartItems.jsp --%>
<%-- Displays the current contents of the cart to the user and allows
the user to select the quantity for each configured PC --%>
<html>
<head>
        <title>Select Quantity</title>
</head>

<%@ page language="java" import="java.util.*" %>

<%-- OrderItemBean stores the configuration details of the PC --%>
<jsp:useBean id="ordItem" scope="page" class="OrderItemBean" />
<%-- GlobalVarBean maintains the shopping cart --%>
<jsp:useBean id="GlobalBean" scope="session" class="GlobalVarBean" />

<%-- Get the PC configuration details from the parameters and store
        them in the OrderItemBean class instance. Add the OrderItemBean
        object to the shopping cart maintained in GlobalVarBean object --%>
<%
        Vector cart = new Vector();

        String speed=request.getParameter("speed");
        String mother=request.getParameter("mother");
        String hard=request.getParameter("hard");
        String ram=request.getParameter("ram");
        String monitor=request.getParameter("monitor");
        String video=request.getParameter("video");
        String cd=request.getParameter("cd");
        String card=request.getParameter("card");

        ordItem.setSpeed(speed);
        ordItem.setMother(mother);
        ordItem.setHD(hard);
        ordItem.setRAM(ram);
        ordItem.setMonitor(monitor);
        ordItem.setVideo(video);
```

```
            ordItem.setCD(cd);
            ordItem.setCard(card);
            ordItem.setQty(1);
            ordItem.calcPCPrice();
            GlobalBean.addToCart(ordItem);
 %>

<body bgcolor="#FFFFFF">

<form method="post" action="Level4.jsp" target="_self">

<%-- Create an HTML table for displaying the list of selectd PC's to the user --%>
<table width="100%" border="1" bordercolor="#000000">
            <tr bgcolor="#9999FF">
                        <td><b><font face="Verdana, Arial, Helvetica, sans-serif" size="2">Item Description</font></b></td>
                        <td><b><font face="Verdana, Arial, Helvetica, sans-serif" size="2">Unit Price US$</font></b></td>
                        <td><b><font face="Verdana, Arial, Helvetica, sans-serif" size="2">Qty</font></b></td>
            </tr>

            <%-- Retrieve the cart and get each PC configuration from the cart --%>
            <%
                        cart = GlobalBean.getCart();
                        Iterator it = cart.iterator();
                        while (it.hasNext()) {
                                    ordItem = (OrderItemBean)it.next();
            %>

            <%-- For each PC configuration display the configuration in HTML table --%>
            <tr bgcolor="#9999FF"><font face="Verdana" size="2">
                        <td><b><u> Desktop PC </b></u><br>
                                    <b>CPU: </b><%= ordItem.getSpeed() %> <br>
                                    <b>MotherBoard: </b><%= ordItem.getMother() %> <br>
                                    <b>HardDisk: </b><%= ordItem.getHD() %> <br>
```

```
                    <b>RAM: </b><%= ordItem.getRAM() %> <br>
                    <b>Monitor: </b><%= ordItem.getMonitor() %><br>
                    <b>VideoCard: </b><%= ordItem.getVideo() %> <br>
                    <b>CD-ROM: </b><%= ordItem.getCD() %> <br>
                    <b>NetworkCard: </b><%= ordItem.getCard() %> <br>
                    <b>FDD: </b>1.44 MB<br>
                    <b>KeyBoard: </b>104 Keys<br>
                    <b>Mouse, SMPS, Cabinet<br></b></td>
            <td valign="top" align="right"><b> <%= ordItem.getPrice() %> </b></td>
            <td valign="top"><b>
                    <input type="text" name="Qty" value=<%= ordItem.getQty() %>
size="3"> </b></td>
                    </font>
        </tr>

<% } %>
</table>

<table border="0" width="100%" cellspacing="5" cellpadding="0">
        <tr>
            <td width="100%">
                    <p align="center"><a href="Level2.jsp" target="_self"><b><i>
                    <font face="Microsoft Sans Serif" size="3" color="#0000FF">Add
                    more to Cart</font></i></b></a></td>
        </tr>
        <tr>
            <td width="100%">
                    <p align="center">
                    <input type="submit" value="Select Shipping Options"
name="submit"></td>
        </tr>
        <tr>
            <td width="100%">
            <p align="center"><font face="Courier" size="1" color="#0000FF">
            Make sure that you have entered correct Quantity for each Item.
            </font></td>
```

```
        </tr>
</table>

</form>
</body>
</html>
```

```
<%-- Confirmation.jsp --%>
<%-- Displays an order confirmation note to the user --%>
<html>

<head>
	<title>Order Confirmation</title>
</head>

<body>

	<p align="center"><font face="Verdana" size="2" color="#0000FF"><b>Thank you for
	placing the order with us. Your order will be delivered to you as per the
	shipping option you have selected. </b></font></p>
	<p align="center"><font face="Verdana" size="2" color="#0000FF"><b>We hope that
	you enjoyed shopping with us.</b></font></p>

</body>

</html>
```

```
<%-- CreditCard.jsp --%>
<%-- Accept the credit card details from the user --%>
<HTML>
<HEAD>
	<TITLE>Submit CreditCard Info</TITLE>
</HEAD>

<BODY>

<FORM action="Level10.jsp" method=post target="_self">

<TABLE border=0 borderColorDark=#ffffff cellPadding=0 cellSpacing=0 width="100%">
  <TBODY>
	<TR>
		<TD align=center vAlign=top>
			<P><STRONG><FONT color=#004080 size=3></FONT><FONT color=#ff8000
			size=4>  Credit Card Payment</FONT></STRONG></P>
		<CENTER><B><STRONG>The information you submit on this page is protected by
			SSL Encryption Technology</B></STRONG></CENTER>
			<HR color=#004080 SIZE=1>
		</TD>
	</TR>
	<TR>
		<TD align=middle ><FONT face="Times New Roman" size=1>enter the
			name on your card </FONT><BR><FONT face="Times New Roman"
			size=1><INPUT name=CardName size=25><BR><BR>enter your card number NO
			SPACES OR DASHES PLEASE</FONT><BR><INPUT name=CardNumber
			size=25><BR><BR><FONT face="Times New Roman" size=1>Enter the 800 number
			on the back of your card here. REQUIRED</FONT><BR><INPUT name=Card800Num
```

```
                    size=25><BR><BR><FONT face="Times New Roman" size=1>enter
the expiration
                    date</FONT><BR><SELECT name=expmonth size=1> <OPTION
selected
                    value=01>01</OPTION>  <OPTION  value=02>02</OPTION>
<OPTION
                    value=03>03</OPTION>  <OPTION  value=04>04</OPTION>
<OPTION
                    value=05>05</OPTION>  <OPTION  value=06>06</OPTION>
<OPTION
                    value=07>07</OPTION>  <OPTION  value=08>08</OPTION>
<OPTION
                    value=09>09</OPTION>  <OPTION  value=10>10</OPTION>
<OPTION
                    value=11>11</OPTION>                          <OPTION
value=12>12</OPTION></SELECT>/<SELECT
                    name=expyear  size=1>  <OPTION  selected
value=00>2000</OPTION> <OPTION
                    value=01>2001</OPTION>  <OPTION  value=02>2002</OPTION>
<OPTION
                    value=03>2003</OPTION>  <OPTION  value=04>2004</OPTION>
<OPTION
                    value=05>2005</OPTION>  <OPTION  value=06>2006</OPTION>
<OPTION
                    value=07>2007</OPTION>                          <OPTION
value=08>2008</OPTION></SELECT><BR><FONT
                    face="Times New Roman" size=1>month

                    
                    year</FONT><BR><BR>
                    <INPUT type=submit name=submit value="Accept">
                </TD>
    </TR>
  </TBODY>
</TABLE>
</FORM>
```

</BODY>
</HTML>

```
<%-- LoginScreen.jsp --%>
<%-- Accepts the login information from the user --%>
<HTML>
<HEAD>
<TITLE>Login</TITLE>
</HEAD>

<BODY>

<FORM action="Level7.jsp" method=post name=theForm target=_self align="left">

<TABLE border=0 cellPadding=0 cellSpacing=0 bgcolor="#C0C0C0" width="100%">
        <TBODY>
                <TR>
                        <TD align=left vAlign=top >
                                <p align="center"><STRONG>
                                <font face="Times New Roman" size="3"
color="#0000FF">Enter
                                 your email address</font></STRONG></p>
                                <BR><BR>
                                <p align="center"><INPUT type="text" maxLength=50
name="email" size=25></p>
                                <p align="center"><INPUT type="submit" name="submit"
value="Login"></p>
                        <p style="margin: 0" align="left"><FONT color=#0000FF
                        face=Verdana size=1>The next page displays your Account
Information if you
                                are a registered member.</FONT></p>
                                <BR><BR><BR>
                                <font face="Verdana" size="1" color="#808080">
                                <marquee bgcolor="#004080" style="color: #FFFFFF"
height="12">Credit
                                Card Purchases will only be shipped to the Cardholder's
Address</marquee>
                                </font>
                        </TD>
```

```
                    </TR>
                </TBODY>
            </TABLE>
            </FORM>
            </body>
            </HTML>
```

```
<%-- ProgressChart.jsp --%>
<%-- Displays the current status of the shopping to the user --%>
<html>

<jsp:useBean id="GlobalBean" scope="session" class="GlobalVarBean" />

<body>

<p align="center" style="margin-top: 0; margin-bottom: 0">
        <b><i><font face="Book Antiqua" size="3" color="#0000FF">Progress Chart</font></i></b>
</p>

        <%-- The level variable in GlobalBean maintains the current shopping level. The program
                retrieves this variable and sets the colors for the shopping items according to the
                level reached so far --%>
        <%
                int level = GlobalBean.getLevel();
                String str1 = "<p align=\"center\" style=\"margin-top: 0; margin-bottom: 0\"><font face=Book Antiqua size=2 color=";
                String str2 = "</font></p>";
                String[] items = {
                        "Welcome",
                        "Configure",
                        "Select Quantity",
                        "Select Shipping Options",
                        "View Cart",
                        "Login for Members",
                        "Select Pay Options",
                        "Pay Process",
                        "Confirmation"
                };

                for (int i=0; i < items.length; i++) {
```

```
                    String colorStr = "#FF3300";
                    if (i < level){ colorStr = "#009933";}
                    out.write(str1 + colorStr + ">" + items[i] + str2);
                }
        %>

</body>

</html>
```

```
<%-- SelectPaymentMethod.jsp --%>
<%-- Requests the user to select the credit card or check payment options --%>
<html>

<head>
        <title>Select Payment Method</title>
</head>

<body>

<p align="center">
    <font size="3"><strong>
    <font face="Verdana" color="#0000FF">Payment by</font>
    <font face="Verdana" color="#004080">
    <a href="Level9.jsp" target="_self">Credit Card</a></font></strong></font></p>

<p align="center"><font face="Verdana" color="#0000FF" size="2">Click on the
    Credit Card to go to our Secure Credit Card Commerce server.</font></p>

<p align="center">
    <font size="3"><strong>
    <font face="Verdana" color="#0000FF">Payment by </font>
    <font face="Verdana" color="#004080">
    <a href="Level10.jsp" target="_self">Check</a></font></strong></font></p>

<p align="center"><font face="Verdana" color="#0000FF" size="2">Click on the
        Check to make a payment by Check.</font></p>

<p align="center"><font face="Courier" color="#0000FF" size="2">
        <marquee><i><strong><b>IMPORTANT: </b>You will not
        be able to change your order after you select a payment option</strong>
        </i></marquee></font></blink></p>
</body>
</html>
```

```
<%-- ShippingOptions.jsp --%>
<%-- Displays the various pre-defined shipping options to the user and
        requests user to select one for the current order --%>
<html>
<head>
        <title>Select Shipping Options</title>
</head>

<%@ page language="java" import="java.util.*" %>

<jsp:useBean id="GlobalBean" scope="session" class="GlobalVarBean" />

<body>

<%-- Sets the quantity for each configured PC in the shopping cart --%>
        <%
                String[] QtyValues = request.getParameterValues("Qty");
                Vector cart = GlobalBean.getCart();
                Iterator it = cart.iterator();
                for(int i=0; i<QtyValues.length; i++) {
                        OrderItemBean ordItem = (OrderItemBean)it.next();
                        int Qty = Integer.parseInt(QtyValues[i]);
                        ordItem.setQty(Qty);
                }
        %>

<form method="POST" action="Level5.jsp" align="left" target="_self">

<table border="1" bgcolor="#9999FF" bordercolor="#000000">
        <tr>
                <td width="590" align="center" colspan="4" valign="top">
                        <font face="Arial" size="4" color="#004080"><strong>
                        Manufacturing and Shipping Options</strong></font></td>
        </tr>
        <tr>
                <td width="25" align="left" bgcolor="#C0C0C0" valign="middle">
```

```
                    <input type="radio" value="100" name="Schedule"></td>
                <td width="130" align="left" valign="middle" bgcolor="#C0C0C0">
                    <font face="Verdana" size="2" color="#004080"><strong>Critical</strong></font></td>
                <td width="177" align="left" valign="middle" bgcolor="#C0C0C0">
                    <font face="Verdana" size="2" color="#0000FF">Ships in 1 business day</font></td>
                <td width="180" align="left" valign="middle" bgcolor="#C0C0C0">
                    <font face="Verdana" size="2" color="#0000FF">Cost:US$100 per system </font></td>
            </tr>
            <tr>
                <td width="25" align="left" valign="middle">
                    <input type="radio" value="50" name="Schedule"></td>
                <td width="130" align="left" valign="middle">
                    <font face="Verdana" size="2" color="#004080"><strong>Expedited</strong></font></td>
                <td width="177" align="left" valign="middle">
                    <font face="Verdana" size="2" color="#0000FF">Ships in 2 business days</font></td>
                <td width="180" align="left" valign="middle">
                    <font face="Verdana" size="2" color="#0000FF">Cost:US$50 per system</font></td>
            </tr>
            <tr>
                <td width="25" align="left" bgcolor="#C0C0C0" valign="middle">
                    <input type="radio" value="30" checked name="Schedule"></td>
                <td width="130" align="left" valign="middle" bgcolor="#C0C0C0">
                    <font face="Verdana" size="2" color="#004080"><strong>Normal</strong></font></td>
                <td width="177" align="left" valign="middle" bgcolor="#C0C0C0">
                    <font face="Verdana" size="2" color="#0000FF">Ships in 3-6 business days</font></td>
                <td width="180" align="left" valign="middle" bgcolor="#C0C0C0">
                    <font face="Verdana" size="2" color="#0000FF">Cost:US$30 per system</font></td>
```

```
		</tr>
		<tr>
			<td width="25" align="left" valign="middle">
				<input type="radio" value="15" name="Schedule"></td>
			<td width="130" align="left" valign="middle">
				<font face="Verdana" size="2" color="#004080"><strong>Saver</strong></font></td>
			<td width="177" align="left" valign="middle">
				<font face="Verdana" size="2" color="#0000FF">Ships in 7-14 business days</font></td>
			<td width="180" align="left" valign="middle">
				<font face="Verdana" size="2" color="#0000FF">Savings:US$15 per system</font></td>
		</tr>
		<tr>
			<td width="590" align="center" colspan="4" valign="middle" bgcolor="#C0C0C0">
				<font face="Arial" size="2" color="#004080"><strong>
				<input type="submit" name="submit" value="Continue"></strong></font></td>
			</td>
		</tr>
</table>
</form>

<table border="0" cellpadding="0" width="610">
		<tr>
			<td valign="top" align="left" width="589">
				<p align="left"><font face="Verdana" size="1" color="#0000FF">You have a choice
				of how fast your system is built and also how long it takes to ship. The shipping
				charges are based on your distance from the nearest VARsight configuration or
				distribution center.  All Production Schedule Times commence upon credit approval.
```

All orders must be processed by 5:00 PM Eastern Time to qualify for Today.</p>
</td>
</tr>
</table>
</body>
</html>

```
<%-- ShoppingCartContents.jsp --%>
<%-- Displays the complete order form to the user along with the quantity, price of
each configured PC and the total order value --%>
<html>
<head>
	<title>Shopping Cart Contents</title>
</head>

<%@ page language="java" import="java.util.*" %>

<jsp:useBean id="GlobalBean" scope="session" class="GlobalVarBean" />

	<%
		double TotalAmount = 0;
		   double TotalPrice = 0;
		   int TotalShipCharge = 0;
		   int TotalQty = 0;
		   String Charge = request.getParameter("Schedule");
		   int ShipCharge = Integer.parseInt(Charge);
	%>

<body bgcolor="#FFFFFF">

<form method="post" action="Level6.jsp" target="_self">

<table width="100%" border="1" bordercolor="#000000">
	<tr bgcolor="#9999FF">
		<td><b><font face="Verdana" size="2">Item Description</font></b></td>
		<td><b><font face="Verdana" size="2">Unit Price US$</font></b></td>
		<td><b><font face="Verdana" size="2">Qty</font></b></td>
		<td><b><font face="Verdana" size="2">Price US$</font></b></td>
	</tr>

	<%
		Vector cart = GlobalBean.getCart();
		Iterator it = cart.iterator();
```

```
                while (it.hasNext()) {
                        OrderItemBean ordItem = (OrderItemBean)it.next();
                        double Price = ordItem.getPrice();
                        int Qty = ordItem.getQty();
            TotalQty += Qty;
            double ItemPrice = Price * Qty;
            TotalPrice += ItemPrice;
            TotalShipCharge += ShipCharge * Qty;
        %>

                <tr bgcolor="#9999FF"><font face="Verdana" size="2">
                        <td><b><u> Desktop PC </b></u><br>
                                <b>CPU: </b><%= ordItem.getSpeed() %> <br>
                                <b>MotherBoard: </b><%=  ordItem.getMother()  %>
<br>
                                <b>HardDisk: </b><%= ordItem.getHD() %> <br>
                                <b>RAM: </b><%= ordItem.getRAM() %> <br>
                                <b>Monitor: </b><%= ordItem.getMonitor() %><br>
                                <b>VideoCard: </b><%= ordItem.getVideo() %> <br>
                                <b>CD-ROM: </b><%= ordItem.getCD() %> <br>
                                <b>NetworkCard: </b><%=    ordItem.getCard()    %>
<br>
                                <b>FDD: </b>1.44 MB<br>
                                <b>KeyBoard: </b>104 Keys<br>
                                <b>Mouse, SMPS, Cabinet<br></b></td>
                    <td valign="top" align="right"><b> <%= ordItem.getPrice() %> </b></td>
                    <td valign="top"><b> <%= ordItem.getQty() %> </b></td>
                    <td valign="top" align="right"><b> <%= ItemPrice %> </b></td>
                </tr>

        <%
                }
                TotalAmount = TotalPrice + TotalShipCharge;
        %>

        <tr bgcolor="#9999FF"><font face="Verdana" size="2">
```

```
                <td valign="top" align="right"><b>Shipping Charges:</b></td>
                <td valign="top">    </td>
                <td valign="top"><b> <%= TotalQty %> </b></td>
                <td valign="top" align="right"><b> <%= TotalShipCharge %> </b></td>
        </tr>

        <tr bgcolor="#9999FF"><font face="Verdana" size="2">
                <td valign="top" align="right"><b>Order Amount:</b></td>
                <td valign="top">    </td>
                <td valign="top">    </td>
                <td valign="top" align="right"><b> <%= TotalAmount %> </b></td>
        </tr>
</table>

<p align="center">
<input type="submit" value="Place an Order" name="submit">

</form>

</body>
</html>
```