A System for Teaching MIS and MBA Students to Deploy a Scalable Database-driven Web Architecture for B2C E-Commerce

Alexander Y. Yap
Campus Box 2075
Martha and Spencer Love School of Business, Elon University
Elon, North Carolina 27244, USA
Tel. 336 278 5590, Fax 336 278 6000
ayap@elon.edu

Claudia Loebbecke
Department of Media Management
University of Cologne
Pohligstr. 1, 50969 Koeln, Germany
Tel. +49-221-470 5364, Fax +49-221-470 5300
claudia.loebbecke@uni-koeln.de

Abstract

The growing need for real-time information and interactive online feedback has shifted the thrust of web development from static websites to dynamic database-driven web applications. 'E-Commerce capable' or 'transaction-capable' websites are naturally database-driven due to the simple fact that transaction-related information (customer and order information) needs to be captured or entered into a database. Although database-driven web applications are seen as solutions for automating online transaction processing, optimizing business processing, and improving online customer relations management, Internet statistics reveal that a substantial majority of websites on the Internet are not E-Commerce capable or transaction-capable, and nor are they dynamically scalable in terms of content. To understand how technology is an extension of corporate strategy in the 21st century, it is vital that MIS and MBA students have a certain level of knowledge about how e-commerce systems actually automate web transactions, help optimize business processes, and how web application systems are designed to handle content, data, or information. This paper discusses how a systems blueprint has been developed over time for educating students to build and deploy a database driven e-commerce website. The system has been successfully used for both MIS and MBA students over a period of 4 years. It has been successful in the sense that students who did not have any background in database and web programming were still able to deploy and design a working system in one semester by themselves.

Keywords: e-Commerce, web architecture, system design, database integration, scalability

1 Research Context and Objective

While many companies have jumped onto the E-Commerce bandwagon, E-Commerce statistics (Netfactual.com 2003, Friscia 1999) indicate that more than 80% of websites have relatively more 'static' rather than 'dynamic and real-time interactive' system architectures. In 1999, an AMR research report, only 12% of companies (approximately) in retail, finance, and the utilities sectors have 'transactions-capable' websites and order tracking capabilities. Four years later, Netfactual.com statistics (2003) claimed that only 19% of all websites are E-Commerce capable, with

5.8 million e-commerce capable websites out of 31 million websites. E-Commerce-enabled or transaction-capable websites are defined as sites that can capture orders, record or track customer transactions, and process payments via an application server and a database. Contrary to the belief of some, not all large corporations have E-Commerce capable sites. Levi.com, for example, has deliberately created a website without transaction capabilities as they felt the need to focus on their bricks and mortars distribution outlets.

It may also come as a surprise to long time users and surfers of the Internet, but the US Census Bureau (2003) reported that 'less than 2%' of total retail sales is transacted online (excluding online auction transactions). Eight years after the start of the Internet Revolution, there are still several factors impeding enterprises from implementing a fully scalable and interoperable B2C E-Commerce web architecture system. Obvious reasons include lack of technical skills and competence, organizational issues, technological bottlenecks (i.e. software application incompatibilities and interoperability problems), system development time, or simply a vague understanding why such system is needed to improve business operations (also refer to Porter, 2001). The trend is that most companies, large, medium, and small, will be putting a lot of effort in choosing and developing their e-commerce infrastructure for the next ten to fifteen years and the new workforce and entrepreneur of tomorrow, and the students of today, needs to understand the importance of e-commerce technology from the 'strategic information systems' perspective (Galliers 1998, Willcocks et al. 1997).

Although IS strategists constantly talk about scalability and interoperability as IS objectives (Treese and Stewart 1998; Turban et al 2002), several e-commerce solutions do not seem to address these goals. Without scalability and interoperability, websites choke at peak web traffic or are difficult to administer when new content or data needs to be added dynamically or linked to another application or server. Therefore, research that develops and evaluates systems architecture and development environments for such systems is in demand especially from a strategic (SIS) perspective. In the classroom, it is worthwhile for students to learn the importance of 1) systems scalability – to address e-commerce growth, 2) web content management administered by a back-end database system – addressing the complexity of managing information, and 3) interoperability between different web applications and systems to see how information is exchanged between different applications and servers.

In this context, the objectives of this paper are

(1) to illustrate a scalable and interoperable e-commerce system that has been successfully taught in 4 years among MIS majors and MBA students
(2) to illustrate and evaluate the process of deploying a highly scalable 'multi-tiered' architecture system.

The paper is structured in the following manner. In Section 2, it starts with a comparative framework illustrating four different E-Commerce solutions and architectures that have been reviewed before developing a teaching course on e-commerce. Then, in Section 3, the work provides a case study discussing the development of an end-to-end E-Commerce solution. The final section then summarizes and concludes the experience in setting up the system.

2 System Development: Requirements and Alternative Architecture Approaches

The paper covers the experience of developing a system that is fully 'E-Commerce capable', i.e. 'transaction-capable' via the Web platform. The objective of setting up the system was to generate a prototype B2C E-Commerce system that was highly scalable, so that it could be used to build either a sparse or extensive online catalog, with complete transaction processing capabilities, web administrative tools, real-time inventory checking, and reporting. We based this architecture on the earlier work of E-Commerce transaction processing systems (Treese and Stewart 2000) and modular multi-tier systems architecture (Mathiassen et al. 2000). We envisioned an E-Commerce system where more than 90% of its content (e.g. product descriptions, prices) is extracted from a database and where the content could be updated by any non-technical person who knows how to fill up a web form and click a 'submit' button. MBA students who had no

previous knowledge of how a system was designed and developed were successful in deploying this architecture within one semester.

2.1 System Requirements

As we planned the system architecture for teaching, we pinpointed six essential components of a B2C E-Commerce system that students should be aware of:

(1) an advertising component (i.e. banners, keywords for web spiders, attractive front-end design) that attracts consumers to surf the online catalog,
(2) an online catalogue that allows consumers to navigate and choose items they want to buy,
(3) an electronic shopping cart that captures the items chosen by consumers from the online catalog (but not mandating consumers to buy the items),
(4) a trigger that captures and transfers order information from the shopping cart into a database (when the consumer finally decides to buy the items),
(5) a payment system that captures and accepts credit card payments along with personal customer information (i.e. billing address, phone), and
(6) a customer relations component that provides satisfaction to customers before and after order fulfillment (i.e. order/item tracking, complain/feedback forms).

2.2 Architecture Approaches

To develop the system, we first compared different solutions for developing transaction-capable web systems. Although there are several variations, we have categorized four 'typical architectural approaches' for purposes of illustration, which we call

Model '1' - Complete Outsourcing of Development and Deployment to a Web-Hosting Firm,
Model '2' In-House Content Generation and Partial Outsourcing of Website Deployment to a Web-Hosting Firm
Model '3' - Stand-Alone E-Commerce Systems Fully Developed In-House, and
Model '4' - E-Commerce Systems Built for Scalability and E-Business Connectivity.

2.2.1 Model '1' – Complete Outsourcing of Development and Deployment to a Web-Hosting Firm

In this solution, the entire E-Commerce system (catalog, shopping cart, order capture, payment system) is fully outsourced. Merchants can customize and manage the system using only a web browser and an Internet connection. In addition, the web hosting company provides the online merchant with an email so that consumers can interact with the merchant for customer relations (see Figure 1).

The basic advantage of this solution is low cost. There are no resource and technology acquisition costs, and it frees the company from problems of technology maintenance.

This approach is sufficient for companies that do not need to update catalog content frequently. It is more appropriate for micro/small enterprises with scarce resources. The disadvantage is that the company's online business is at the mercy of the Web hosting company as they have total control of the technology and its setup.

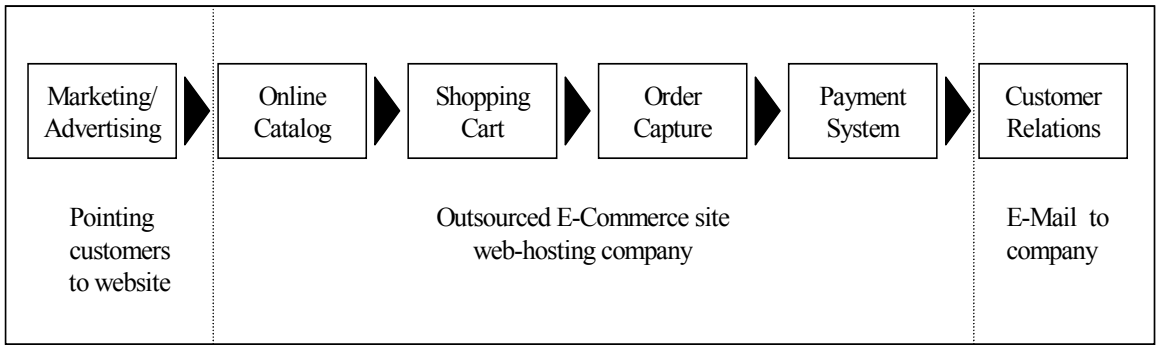

Figure 1. Model '1' - Complete Outsourcing of Development and Deployment to a Web-Hosting Firm

2.2.2 Model '2' - In-House Content Generation and Partial Outsourcing of Website Deployment to a Web-Hosting Firm

The second solution is to partially outsource the development of the E-Commerce system (see Figure 2). In this case, the company has an in-house webmaster that develops/designs online advertising and catalog layouts, and is also an expert in front-end design (HTML, Flash, and JavaScript). However, the company outsources electronic transaction components like the shopping cart and the payment system to a Web hosting company, because these components need a web application server to manage clients and client sessions (e.g. remembering what clients put in their shopping cart during a browsing session) which cannot be fulfilled using static websites using only HTML.

The advantage of this solution is that companies have more freedom to be creative in designing and presenting the 'content' or the 'website design layout' that it wants conveyed to the electronic market. In terms of technical constraint, deploying an online transaction system goes beyond HTML technology, and most companies lack such technology or implementation competence. The weakness of this model is that customer and order information are still transparent to the web-hosting company, because such information are captured on the web-hosting companies' database server. Although there is the apprehension that customer data is held by a third-party, the positive side is that excellent web-hosting companies perform regular data back-ups, system checks, and keep clients' privacy in sacred trust (as they claim).

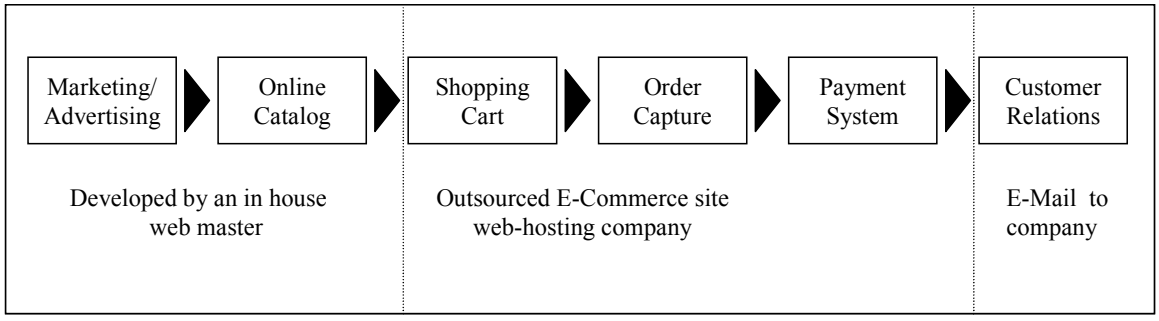

Figure 2. Model '2' - In-house Content Generation and Partial Outsourcing of Website Deployment to Web-Hosting Firm

2.2.3 Model '3' – Stand-Alone E-Commerce Systems Fully Developed In-House

When a company has a large product selection and sizeable customer base, it may fully develop and deploy a complete system in-house (see Figure 3). However, in this model, the system is accessible to few technical people (webmaster, graphic designer, network technician, database administrator).

The personnel maintaining the system are fully responsible for updating the website's content and the maintenance/ improvement of the system. Non-technical personnel are not involved in the evolution of the E-Commerce system. The system uses a stand-alone file database (e.g. Access), which is not directly accessible across the entire enterprise. It may come as a surprise that some large enterprises, whose E-Commerce systems are separate from their other systems, are running them on a rudimentary database such as MS Access. However, we have encountered some E-commerce websites that throw out errors saying 'OBDC connection not found for MS Access database'. This clearly indicates that some E-Commerce outfit still uses MS Access as their database.

The weakness of this system is that file database systems, such as MS Access, are frail in terms of simultaneous data access or multi-thread database connections. Therefore, this system is not very scalable and robust. However, it can handle the volume of online transactions up to a certain level. It works for companies that are not keen in going into systems integration. These are companies that run their E-Commerce system separate from other corporate information system. Companies use this particular model/ solution because they hold the conservative notion that E-Commerce should be a separate division from other parts of business operations. The problem with this kind of E-Commerce system is the lack of scalability and connectivity to information systems used in other business operations. Problems will arise when: (1) the time comes for the system to accommodate more web traffic; (2) there is need to improve customer relationship management by linking the E-Commerce transaction processing system to other systems (i.e. procurement system, customer support) or to the physical logistics (transport and warehousing systems); or (3) when the consumer expects other processes (e.g. order fulfillment) to be almost as fast as the electronic transaction process (hence the need to connect the online transaction processing system to other enterprise systems).

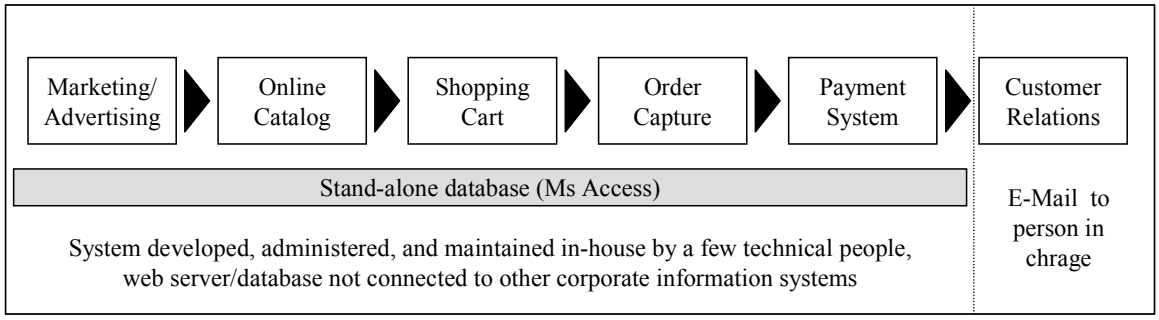

Figure 3. Model '3' - Stand-Alone –E-Commerce Systems Fully Developed In-House

2.2.4 Model '4' - E-Commerce Systems Built for Scalability and E-Business Connectivity

The architecture for this E-Commerce system is built on the principles of scalability, interoperability, and rapid integration with a larger or more extensive E-Business information system. This type of architecture can be scaled down for small or medium-scale businesses that needs some basic system integration, or scaled up for larger enterprises for a more elaborate integration effort with several enterprise-level information systems.

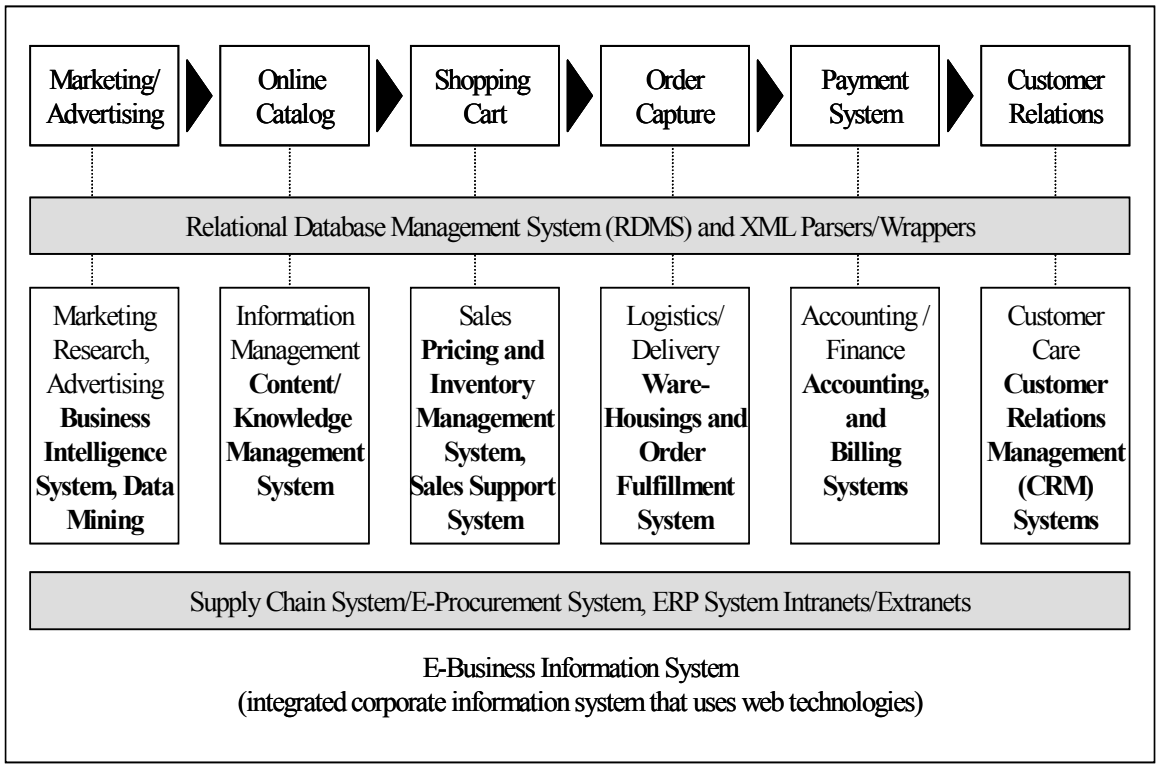

Figure 4. Model '4' - E-Commerce System for Scalability and E-Business Connectivity

Unlike Model '3', this system enables non-technical personnel to update and manage web content without the help of technical people. In this architecture, web content management functions are intentionally separated from the design interface and application logic. In this respect, scalability concerns should not only be limited to the ability of a system to accommodate web traffic, because scalability can also be seen as the ability of the system to support rapid content expansion and richness. In addition, the system is intended to improve content structure and real-time content update. Content updates are not necessarily limited to content needed by consumers, but content updates can include real-time reporting of daily sales or inventory levels needed by managers. As products are bought from the online store, events in this transaction can be immediately reflected in an online sales report or a stock inventory update.

Figure 4 illustrates how the transaction process components of this system can be connected to other E-Business systems either through an enterprise-level relational database such as Oracle using special database drivers or standard ODBC connections (Merant website, 2002) or through XML parsers/wrappers via the API (Application Programming Interface) of the Middleware that is uses. (See next section for a detailed discussion of the business process).

Figure 5 illustrates such a multi-tiered E-Business architecture with a three-tier E-Commerce architecture acting as the core system for linking online customers to the rest of the enterprise. With three-tier architecture, the E-Commerce system can be connected to other systems (i.e. ERP), because the relational database is an independent tier and runs with other non-E-commerce systems as well.

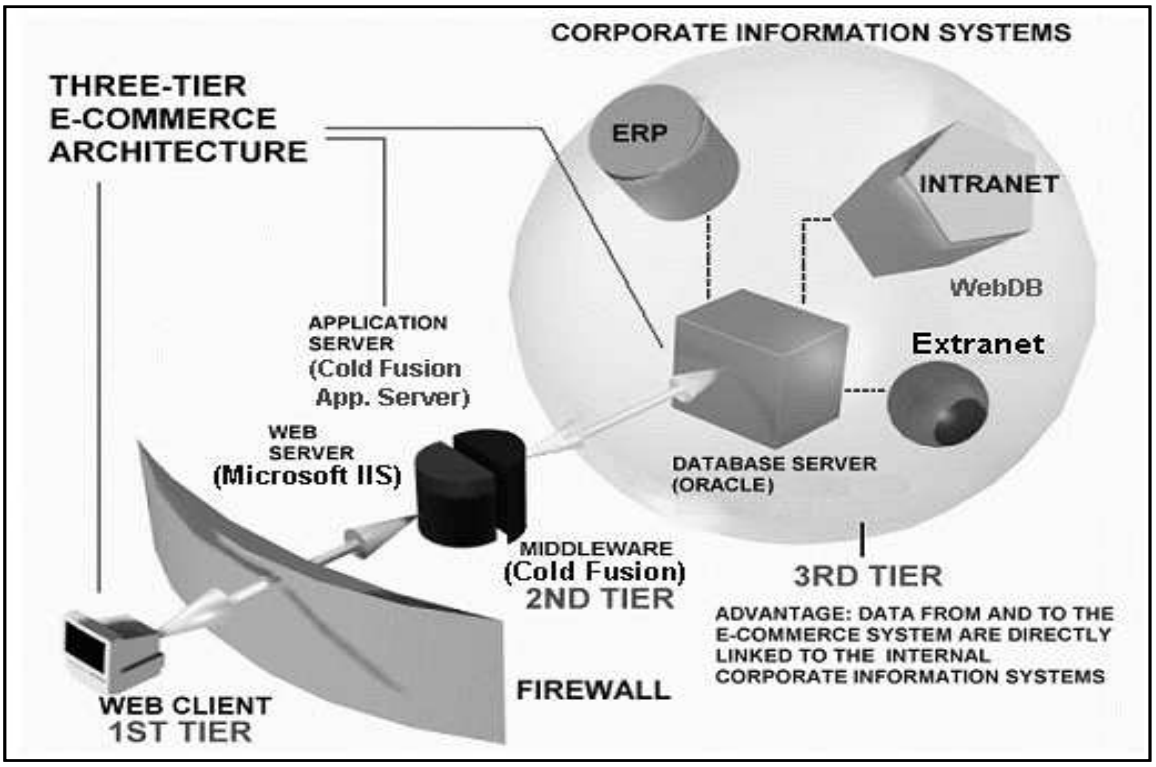

Figure 5. Resulting to Multi-Tiered System - Three-Tier E-Commerce Architecture with Interconnectivity to other Corporate Information Systems

With this type of architecture, the E-Commerce system can be integrated or interfaced with other types of information systems. A database-driven E-commerce system could be interfaced with a larger system such as an ERP or a department-specific information system via the relational database or other application interfaces (i.e. XML). This includes systems such as business intelligence, data mining, content/knowledge management, inventory management; order fulfillment, accounting/billing, customer relationship management, and warehousing/logistics systems. The marketing or advertising component of the E-Commerce system can be linked to a Business Intelligence system that helps direct or redirect advertising content to specific customer profiles collected from the transactional data (i.e. previous orders and favorite items bought). Data Mining systems can also help search valuable information to support form personalized advertising or one-to-one marketing. The online catalog can be linked to a content or knowledge management system that will help customers in their decision making process (i.e. product specification comparisons; the right type of product to buy base on customer requirements). The catalog, shopping cart, and order capture system can be linked to Inventory Management systems so that whatever is bought online is immediately reflected in the stock inventory levels. The E-Commerce system can also be linked to a sales support system, so that whatever is bought online is properly documented with the sales team in case the customer needs to phone sales support for questions on the order or problems with the order. Once orders are captured in a relational database, warehousing systems, logistics and order fulfillment system can process the transactional data for delivery to customers. The payment process of the E-Commerce system can be connected to the Accounting or Billing system, so that orders are processed faster and at the same time properly recorded in the system. Lastly, the E-Commerce system can be connected to a separate customer relationship management system (CRM).

In such a system, information content displayed on the web pages is not contained or encoded in HTML web-pages, but in data variables queried

from a relational database and automatically translated into HTML format by a web application server. Moreover, data captured by the E-Commerce system (i.e. order data and customer data) become immediately accessible and transparent across other E-Business systems in real-time via an enterprise-level relational database (sometimes with the help of XML to preserve complex data structures, such as data arrays). In brief, the E-Commerce system is created with E-Business systems connectivity in mind, and the enterprise-wide relational database plays a pivotal role in how it inter-connects to other information systems.

Such a system is better implemented using a three-tier architecture where the database runs on a separate platform (a database server) - a 'tier' independent from the web application server. With such a setup, the database server could be located in any geographic location and is Internet-accessible via TCP/IP and HTTP protocols/ connections and services. The web server can fetch data from remote database servers using the middleware's ability to pass data between the web client and the database.

The E-Commerce system is fully developed in-house and can be maintained by both technical and non-technical personnel. Technical personnel are in charge of systems maintenance or upgrades, while non-technical personnel (i.e. marketing/sales people) take care of content management (i.e. online catalog, other information needed by consumers). What a non-technical person needs to know, for him/her to update web content, is the simple ability to 'key-in' data in a web form (like filling up online surveys).

3. Towards a Scalable Architecture E-Commerce System

When setting up our E-Commerce system, we chose to develop a system tailored according to the Model '4' architecture for MIS students and Model '3' for MBA students (as elaborated in Section 2.2). MIS students already had a background of how Oracle database systems work and could therefore do Model 4, but MBA students could only deal with a stand alone database such as MS Access. At any rate, both systems showed students how web content could be scaled automatically by adding more content in the database and having the middleware translate database information to web content.

We also envisioned a system that was modular in terms of its ability to be ubiquitously connected to other systems even outside a LAN environment of the university, so students could work at home as long as they had Internet connection. Lastly, this E-Commerce application should be suitable for small enterprises, but also easy to be scaled-up for use with larger corporations. Below is a description of the process that we undertook when developing the E-Commerce system for teaching.

3.1 Step 1: Choosing the Appropriate Technology

When we initiated plans to build the E-Commerce system, we had a variety of top-level E-Business software in our inventory - SAP/R3, JD Edwards' One World ERP software, Oracle JDeveloper, Oracle Data Warehouse, SAS Enterprise Miner, Oracle 8i, and IBM WebSphere to name a few. There were some discussions on acquiring Oracle's E-Commerce suite or using IBM's WebSphere, since we were historically committed to using Oracle and IBM products.

Therefore, the initial dilemma was the choice of technology. Looking closely at the software available, we knew that most options for SMEs were expensive solutions and affordable only by larger corporations. We had to consider that some students would like to set websites for small businesses, and that some MBA students mentioned that they want to set up their own small businesses online and was excited to learn about designing their website.

In the discussion below, our decision to acquire Allaire's Cold Fusion (the middleware application), the enterprise-level relational database system 'Oracle 8i' and macromedia's front end design suite (Dreamweaver, Fireworks, and Flash) will be outlined. Allaire's Cold Fusion and Macromedia's suite contained cheaper solutions compared to Oracle or IBM web development products. (Allaire in 2002 had a large business merger with Macromedia and Cold

Fusion products are now packaged with Macromedia products).

3.1.1 Choosing the Middleware Package (Cold Fusion)

When choosing the middleware package, we assessed the middleware to be the most critical component for providing real-time functionality and interactivity to the E-Commerce application, because middleware creates the application logic that addresses the automation of business processes and tasks. When we compared Cold Fusion to Oracle's E-Commerce suite, there was hesitation to use the newly released Oracle suite for two reasons. Firstly, the Oracle suite was new to the market, and secondly we did not want to 'lock-in' our web application development environment to a big vendor such as Oracle or IBM.

There were several reasons why we decided to use Cold Fusion (CF) over the cheaper Active Server Pages (ASP). This reflects our bias, but also shows why we decide to use one technology over another. Table 1 summarizes our arguments.

(1) Although ASP comes 'basically' free with Windows, ASP is constrained to the Windows platform. Some MIS students wanted to run the server on a Linux box. The basic ASP package also has limited functions and third-party software (an additional cost) is needed to seriously run ASP for robust commercial E-Commerce applications. On the other hand, CF is ready to run commercial sites and is a fully interoperable application server as it runs on Windows NT/2000 server, Linux, Solaris, and HP UNIX. Therefore, we had the flexibility to set CF up in most major operating systems (aside from Windows).

(2) CF has native drivers for enterprise RDBMS such as Oracle, IBM DB2, Sybase, Informix, and SQL Server, as well as OBDC connection for MS Excel, MS Access, FoxPro, Oracle and any OBDC-equipped database. Thus, CF was seen as more versatile in terms of multi-vendor database scalability, modularity, interconnectivity, and compatibility. In contrast, out-of-the-box ASP only comes with OBDC connectivity. We confirmed that CF native database drivers were faster as they did not pass through the extra OBDC layer when the CF application server communicated with the database.

(3) CF can parse XML codes using CF tags; therefore, CF can work with E-Business systems that employ XML or with mobile commerce systems that use WAP (wireless application protocol), which is based on WML (wireless mark-up language). WML is a derivative of XML. Cold Fusion also addresses the need of B2B applications for data translation using XML due to its inherent XML support. Only one CF tag, <CFWDDX>, is needed to parse/serialize/de-serialize XML codes.

(4) Unlike ASP, Cold Fusion comes with software load-balancing for web server clustering and distributed web traffic handling for web farms (multiple web servers connected to each other). Cold Fusion includes a software service module called 'Cluster Cats' that performs this function. This offsets expensive hardware load-balancing solutions (such as CISCO Catalyst). However, Cluster Cats is limited to eight web servers. Beyond eight servers, hardware-based load balancing is the answer.

(5) Cold Fusion has a feature called 'security sandboxes' that allow several developers to work independently and concurrently on a project in their own secured workspace/directory. This feature supports rapid deployment as several developers can work simultaneously on one central deployment server.

(6) Cold Fusion has support for CORBA, COM and JAVA objects used for object-oriented components/applets needed for certain web application functions.

(7) Macromedia Homesite Plus (which used to be Cold Fusion Studio), the script editor that generates CF templates (*.cfm) and bundled with Macromedia Dreamweaver, is also geared for rapid application deployment. It has features like 'SQL Builder' that automatically generates SQL scripts, while 'Template Wizards' could generate 3-5 web-pages

(with data-handling capabilities) almost instantaneously, saving a lot of coding time. Homesite Plus also allows the creation of custom tags and code snippets, making coding very modular and re-usable. Custom tags and code snippets are similar to database 'stored procedures'. These are modular codes/scripts that can be used over and over again, and ASP does not have such feature.

A very short comparison table between CF and ASP (Table 1) summarizes why we chose Cold Fusion at that time. As of 2002-03, ASP has improved its functionality and ASP is an excellent language for web development. However, it remains a fact that ASP is native to the Windows/Microsoft operating environment, while Cold Fusion has always been multi-platform running on Unix and Linux in addition to Windows. As can be seen in Table 1, our primary concern was scalability, interoperability, and rapid deployment rather than cost.

Features	Cold Fusion (CF)	Active Server Pages (ASP)
Multi-Platform (Interoperability)	Windows, HP Unix, Solaris, Linux	Windows only
Native RDBMS Drivers (i.e. Oracle drivers)	Yes	No
Simplified One Tag XML Parser	Yes	No
Clustering and Software Load Balancing	Yes	No
Security Sandboxes	Yes	No
CORBA / COM / JAVA Support	Native support for all three	Only COM is native to ASP
Rapid Deployment	Drag and Drop coding, custom tags, automated scripting with SQL Builder, Template Wizards	No

Table 1. Comparing Cold Fusion and ASP for Scalability, Interoperability and Rapid Deployment

3.1.2 Choosing the Relational Database System

For MIS students who were trained in enterprise-level relational database, it was necessary to use an enterprise-level relational database, such as Oracle, to achieve the level of scalability and robustness if these students needed to work with larger Fortune 500 companies that use Oracle. For this reason, MIS students needed to dovetail and link their knowledge of Oracle with that of an E-commerce system. With Cold Fusion, we knew that we could easily downscale the database by using Microsoft Access (as a cheaper and easier setup) for MBA students who did not have previous knowledge of complex relational database programming, such as Oracle. Both Oracle and MS Access connected well with Cold Fusion, as a middleware for delivering database-driven e-commerce system.

3.1.3 Choosing the Front-End System: Macromedia

Finally, we needed to decide on the tools to be used for front-end design. Considering that Macromedia was the perceived leader in the front-end design market, we choose to use Macromedia Flash, Dreamweaver, and Fireworks. (Macromedia claims that more than 90% of web browsers have a Flash plug-in that runs Macromedia animations.) We used Macromedia Flash and Fireworks more than Dreamweaver, because Cold Fusion Studio (now renamed as Macromedia Homesite Plus) could generate HTML coding and formatting as well as other web languages like Cold Fusion, ASP, JSP and Java. Unlike Cold Fusion, Macromedia Dreamweaver was more WYSIWYG-oriented and the objective was to expose students to some coding and programming, so we choose Homesite Plus as the coding software. In our experience, Macromedia Flash was the best tool for two-dimensional (2D) web

animation. Flash not only could create motion graphics, but it also provided sound effects and real-time interactivity to web presentations with its multimedia capability and scripting language (Flash Actionscripts). However, if we embed graphic files (*.bmp, *.jpg) in addition to the drawings that Flash generates with its native vector-based graphic tools, downloading a web page with Flash animation can take longer. Several surfers consider animations only as a browsing obstacle. On the other hand, the fast growth of broadband Internet connection (DSL, Cable, Optical Cable) is to address the issue of downloading multimedia-based or graphic-intensive web-pages.

3.2 Step 2: Creating the Application Development Environment

Setting up the Cold Fusion web application server with an Oracle 8i database server as a separate tier was a trial-and-error configuration process due to the lack of public documents available to show how these two could work together. With Cold Fusion and Oracle 8i working together, we tested three scenarios (see Figure 6). We were successful on all three scenarios.

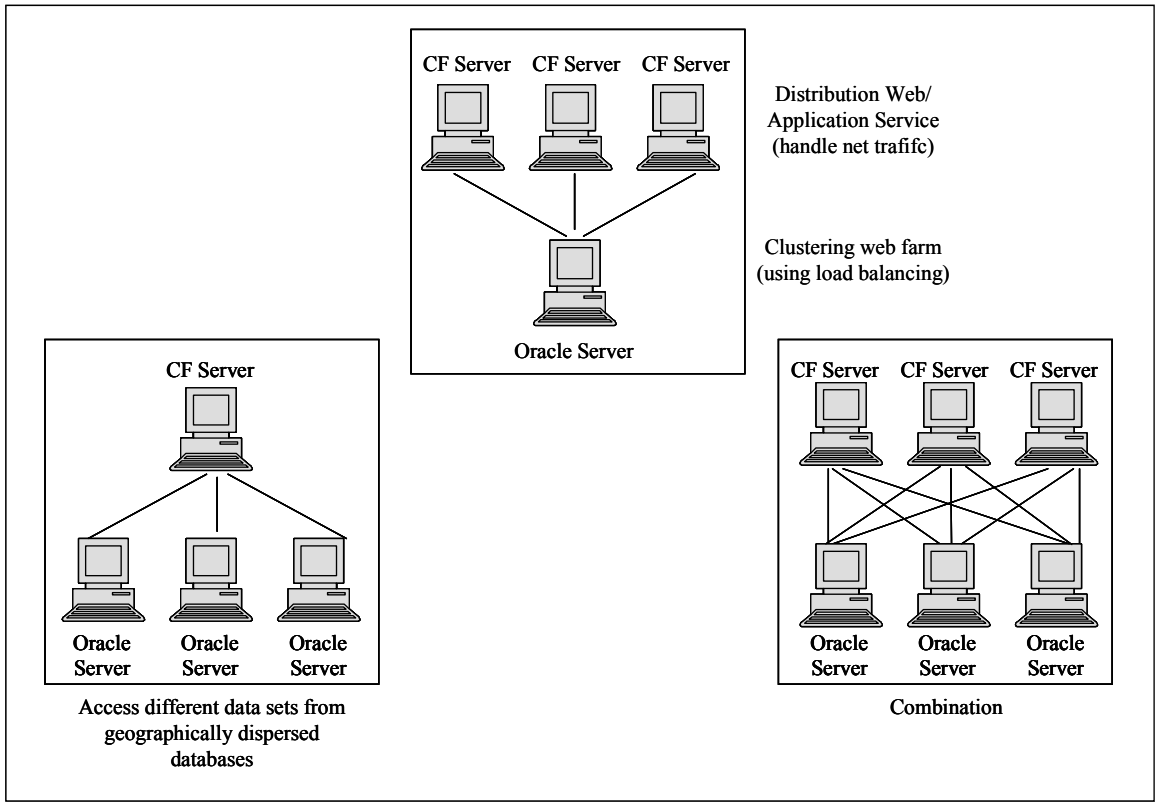

Figure 6. Set Up for Web Application and Database Scalability Using Oracle and Cold Fusion

(1) Application Scalability - the first scenario was for two or more Cold Fusion application servers to access one Oracle database server. The purpose of this testing was to see if the architecture is scalable at the web server level, where more than one web server can share the load burden of web traffic but keep one central database.

(2) Database Scalability - the second scenario was for one Cold Fusion server to access two or more Oracle database servers. The purpose was to test the Cold Fusion application server's ability to pass data to several remote databases.

(3) Multi-tier Scalability - the third test was to see two or more Cold Fusion servers accessing two or more Oracle

database servers. The purpose of this was to see scalability at both the web server and database server levels.

After testing the applications for scalability, the next step was to design a web development environment that allowed students to work at home with client software while the teaching faculty maintains the web and database server within the university premises (see Figure 7).

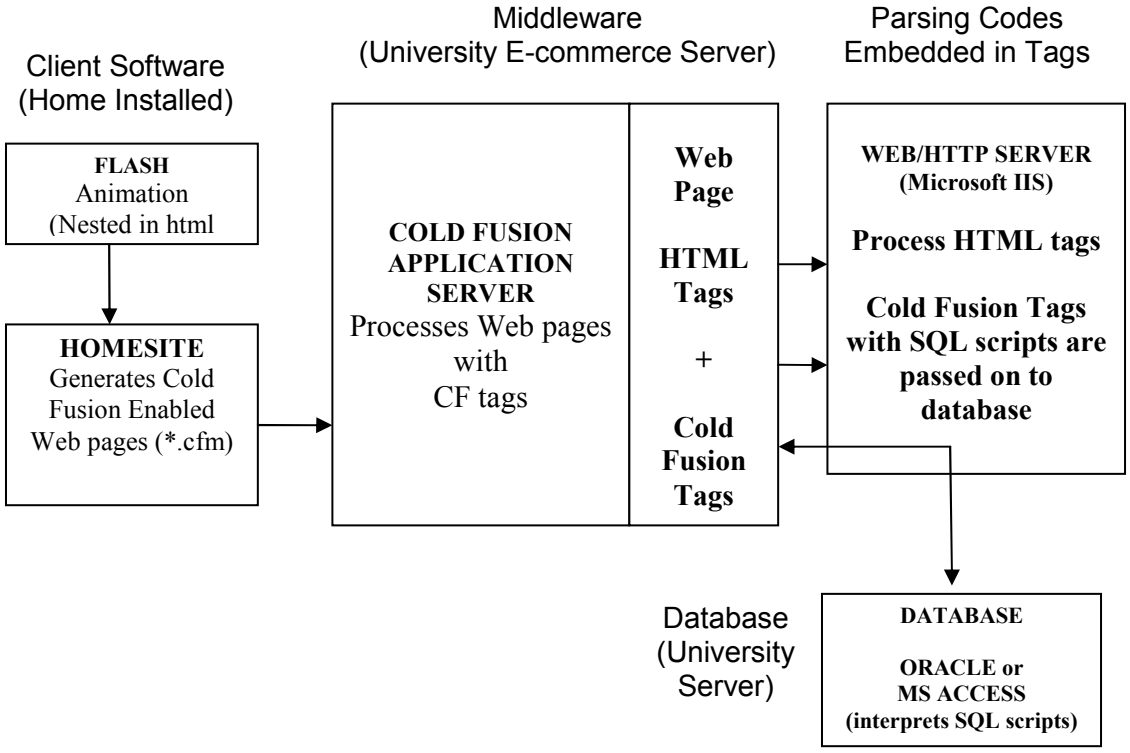

Figure 7. The Web Development Environment

The E-Commerce Server (mainly using the middleware application), as illustrated in Figure 7, is set up within the University premises and it is shared and accessed by more than 20 graduate students (MIS and MBA students) in a given semester. The students can access the server remotely by using a client development application such as Macromedia Homesite Plus, which is packaged with Dreamweaver MX. The students also use Macromedia Flash to create animated front-end design for their e-commerce storefront.

Microsoft's HTTP server, IIS, is installed with the Cold Fusion Application Server (the middleware) running on top of the HTTP server. The HTTP server is responsible for translating HTML tags and all HTTP requests from the Web, while the Cold Fusion server is responsible for translating Cold Fusion tags and Structured Query Language (SQL) which the database understands. The Cold Fusion server translates and passes all SQL scripts to the database, thereby allowing data to be culled from or inputted into the database via a Web browser. For MIS students, Oracle 8i was installed in a separate database server that connected to the Cold Fusion server in order to illustrate a more robust true three-tiered architecture. For MBA students, the MS Access software was installed on the same server as the Cold Fusion Server for ease of operation and use.

This was a good way to show students that they could develop and design a complete web application without having to run and maintain a web and database server themselves. However a

handful of savvy students were permitted to run their own Cold Fusion server.

The E-Commerce application sitting on the web server was developed remotely using Homesite Plus' remote development service (RDS). On a cable modem, RDS was fast enough that students did not notice that writing codes at home were directly saved to a remote server.

3.3 Step 3: Designing Database and Interface for the E-Commerce System

The next challenge was to determine the requirements at the database level. What tables and columns were to be created? What kind of data should be provided and accessible to the clients (e-consumers)?

What data were to be collected from the clients (e-consumers)? Table 2 illustrates the basic data to be provided and collected on a real-time basis. MIS students created the Oracle database with Oracle's SQL script editor, while MBA students created the database via a spreadsheet-like format with MS Access. The challenge was to integrate business processes and to link those to Web functionality. Table 3 shows how we linked business processes to the features of the website, and which tools we used.

The online catalog displays product name, price, product description, and the number of stocks available (product 'in stock'). Figure 8 depicts the Oracle database showing the same variables, except for the number of stocks available. Due to relational database normalization the 'stocks available' data is in another database table.

Cold Fusion displays all these data together in the catalog by using the SQL language. Figure 9 represents a 'database-driven HTML template' as the content is not written in HTML, but is culled from an Oracle database (Figure 8). Only the presentation format (i.e. font, colors) is HTML. Changing or manipulating the data variables in the Oracle database table will automatically change the content of the web interface.

Data to be Provided	**Data Content**	**Data to be Collected**	**Data Content (Samples)**
Product Categories (Catalog)	Product Categories (i.e. cars, computers)	Client Information (upon ordering)	Name, Address, E-Mail, Phone
Products (Catalog)	Product Name, Product Description (short & long), Price, Images (small & large)	Payment Information (upon ordering)	Credit Card Number, Billing Address
Product Inventory (Catalog)	Stock Available (quantity), Back Orders, Date of Delivery	Order Information (upon ordering)	Products Ordered, Quantity, Order Date
Order Invoice (after Ordering)	Order No., Item No., Items Ordered, Total	Customer Services (after ordering)	Complaints and Comments, Item and Order Tracking

Table 2. Database for E-Commerce Application

Business Process (Business Logic/Objectives)	**Website Functionality (Application Logic)**	**Software Usage (Implementation Tools)**
Marketing Process (attract customers to surf site)	- Storefront presentation - Layout of catalogue (Enticing and professional front-end design)	- Cold Fusion Studio (cascading style sheets, HTML formatting) - Flash 5 and Fireworks (2D Animation, Interactive Graphics, Dynamic Buttons)
Decision Support Process (help customers choose products)	- Navigability of online catalogue - Information and content - Drill down search engine for product search - Real-time inventory to show product availability	- Cold Fusion and SQL for linking web templates and data variables across templates - SQL scripts to search database - Flash5 for easy navigation menu
Buying/Shopping Process (ease the shopping experience)	- Passing data variables from online catalogue to shopping cart - Track client's shopping activities	- Cold Fusion to manage and keep track of multiple client sessions - Cold Fusion to hold temporary session variables in server's RAM before committing it to database
Order Capture Process (receive/record orders)	- Transferring temporary variables from the shopping cart into physical database - Committing the final order	- Cold Fusion and SQL to automate data input for shopping cart and into Oracle database - Cold Fusion automatically generating order invoice and sales reports
Payment Process (credit card verification, knowing customer's identity and address)	- Credit Card verification - Capture credit card number - Customer Information (name, billing address, phone, e-mail)	Cold Fusion allowing connectivity to: - Credit card verification - Financial payment gateways - E.g. Redi-check, Cybercash
Customer Service Process (give customer satisfaction)	- Item and order tracking - Automatic e-mail feedback - Customer feedback and complain Form	- Cold Fusion passing data to SMTP server to send automatic e-mail and include data variables in e-mail message - Cold Fusion linking item and order number to other information

Table 3. Integrating and Automating Business Processes through Web Functionality and Tools

ESHOP1.PRODUCTS

PRODUCTNAME	DESCRIPTSHORT	UNITPRICE
Athlon 1.2 Gigahertz	for power users and beginners	300
Intel Computer	fast processing	349
Intel Motherboard	Fast Processing	299
Samsung Flatscreen Trinitron	Brings sharper images	699

Figure 8. Oracle Database Table as Reflected in the HTML Web Template of Figure 9.

[Home][View Basket] PRODUCT CATEGORY: Computers

PRODUCTS
Click Links Below for Detailed Specs

Athlon 1.2 Gigahertz	$300.00	for power users and beginners		20 in stock	Quantity 1 Buy
Intel Computer	$349.00	fast processing		25 in stock	Quantity 1 Buy
Intel Motherboard	$299.00	Fast Processing		25 in stock	Quantity 1 Buy
Samsung Flatscreen Trinitron	$699.00	Brings sharper images		22 in stock	Quantity 1 Buy

Figure 9. Online Catalog Using Cold Fusion to Retrieve Content from Oracle Database

3.4 Step 4: Linking Business Processes to Web Functionality and Software Tools

We experienced a technical challenge to integrate and automate business processes for E-Commerce. Each business process has its own logic, and the scripts to execute such logic are embodied in one or more templates. These business processes are only integrated and automated electronically due to the ability of the E-commerce application to seamlessly pass and preserve data from one electronic process to another.

Due to the length of describing the links between business process, web functionality, and software tools, we deemed it better to compress the

description of the business process flow presented in Table 3, where we show how business process integration and automation is enabled by web application technology (such as Cold Fusion). Business process integration is enabled with Cold Fusion technology, because Cold Fusion enables data variables to be passed/accepted/processed from one web-page template to another. Creating an application that could hold and keep track of temporary variables that are not committed to the database during a client session (e.g. data temporarily stored in a shopping cart culled from an online catalog when a client is in the process of buying/shopping) is where the strength of the Cold Fusion application server lies. Application servers can keep track of unique variables/data associated with a unique client session, a particular client, and the client's ongoing shopping activities. Cold Fusion uses only one HTML-like tag to perform such function - <CFAPPLICATION>. Cold Fusion tags are interpreted and processed by the CF application server. Companies that cannot develop this particular web functionality will have to resort to outsourcing of their online transaction process.

There are several unique Cold Fusion functionalities that help integrate/automate business processes electronically. To mention a few, Cold Fusion manages and keeps track of transaction data by using SQL scripts with the Cold Fusion <CFQUERY> tag. The CFQUERY tag enables Cold Fusion to transfer and manipulate data from HTML webpages to and from the database. This is how Cold Fusion captures customer and order data/information automatically. It is also a critical element that let users manage/administer the web application (e.g. inserting/updating new product data or price) via any web browser. Cold Fusion can also enable real-time credit card verification using <CFX_ICV> tag or automate credit card payments using its <CFX_CYBERCASH> tag. This tag connects the online store to financial payment gateway systems like 'www.icverify.com' or 'www.cybercash.com' that verify/authenticate credit cards and also accept payments for or in behalf of merchant bank accounts. In terms of the customer relationship and feedback process, the tag <CFMAIL> enables email automation (i.e. email order confirmation, email notification) through an SMTP server. Automated feedback is vital to customers who need to know that their order submissions are confirmed and processed. With Cold Fusion, an online form was created as part of the system. This is to enable customers to get support services and solutions by entering an Item Number (a unique identifier for each item sold). When customers enter such number, the web application traces the order invoice and customer information so that company personnel handling customer relations can see the complete transaction information. Although these illustrations only represent a small portion of the system's functionality, they show how Cold Fusion can effectively automate business process – from ordering, payment, credit card verification, to customer relations.

4 Conclusion and Outlook

There are several benefits from adopting a scalable and tiered B2C E-Commerce architecture as depicted and outlined in the Model '4' architecture. One benefit is to scale the online content (e.g. add new products, new product categories, expand product description) by just updating the Oracle database and without touching or recoding the web application or layout. The architecture also allowed separating 'content' and 'web design layout' from the 'application logic', which made the system more robust. Further we could scale the application both at the web-server and database levels as discussed earlier. Because of this, students saw opportunities for distributed computing and linking to several remote databases. It helped students understand how they could extend the system towards much broader E-Business applications and functionalities. By observing how Cold Fusion works with the SQL language, students also saw possibilities on how to improve the filtering of data into more meaningful information and generating real-time analytical reports (like sales reports, inventory reports). The disadvantage for deploying a three-tier system was that the learning curve was quite steep. Only MIS students with good knowledge of relational databases can cope with the Model 4 architecture. MBA students, using Model 3 (with MS Access), were content that they have learnt how to apply web and database technology on a more strategic level when planning on improving their work and company's business processes. Creating such an architecture is definitely not for beginners, so MBA students with no prior background needed more hands-on tutorial. Nonetheless, several MBA students found it useful knowing that they

had succeeded in working and deploying a systems architecture that allowed them to discover how database-driven web technologies can be used strategically and that the same technology could be applied for automating other business applications, other than e-commerce, in their line of work.

References

Abbey, M., Corey, M. & Abramson, I. (1999) Oracle 8i: A Beginner's Guide, Osborne McGraw-Hill (Oracle Press), California.

Allaire's, Cold Fusion (2004) www.allaire.com.

Brown, B. (2000) Oracle 8i Web Development, McGraw-Hill (Oracle Press).

CGI101 (2001) www.cgi101.com.

Commerce Blocks Website, Cold Fusion Developer (2001) www.commerceblocks.com.

Cybercash (2004) www.cybercash.com. (2004).

Danesh, A. & Motlagh, K.A. (1999) Mastering Cold Fusion 4, Sybex, California.

Francis, B., Kauffman, J., Llibre, J., Sussman, D. & Ullman, C. (1998) Beginning Active Server Pages 2.0, Wrox Press Ltd, UK.

Friscia, T. (2001: A B2B Space Odyssey, AMR Research Report (www.amrresearch.com), November 1999.

Forta, B. (1998) The Cold Fusion 4.0 Web Application Construction Kit, Que, Indiana.

Forta, B. (1999) Advanced Cold Fusion Application Development, Que, Indiana.

Galliers B. (1998) Reflections on BPR, IT and Organizational Change. In: Information Technology and Organizational Transformation: Innovation for the 21st Century Organization, Galliers, R. & Baets, W. pp.225-244. John Wiley & Sons, New York.

Greenwald, R. & Milbery J. (1999) Oracle WebDB Bible. IDG Books Worldwide Inc, New York.

ICVerify (2001) www.icverify.com.

Lang, P. (2001) Marketing for Long-term Success, SellitontheWeb.com.

Levis Corporation (2004) www.levi.com.

Macromedia (2004) www.macromedia.com.

Mathiassen, L., Munk-Madsen, A, Nielsen, P. & Stage, J. (2000) Object-Oriented Analysis and Design, Forlaget, Aalborg, Denmark.

Merant (2000) www.merant.com.

McGrath, S. (1998) XML by Example: Building E-Commerce Applications, Prentice Hall PTR, New Jersey.

NetFactual (2004) www.NetFactual.com., The E-Commerce Universe.

Oracle Website. (2003) www.oracle.com.

Pascarella, C. (2000) (representing www.virtualscape.com), CF Hosting Tips, CF Conference, Maryland, July.

Porter, M. E. (2001) Strategy and the Internet. Harvard Business Review, March, 63-78.

Redix (Cold Fusion-to-XML Patch) (2000) www.redix.com/granluarity.htm.

Treese, G. & Stewart, L. (1998) Designing Systems for Internet Commerce. Addison Wesley.

Turban, E., Lee, J., King, D. & Chung, H. (2000) Electronic Commerce: A Managerial Perspective, Prentice Hall.

US Census Bureau. Department of Commerce (2003) www.census.gov/mrts/www/current.html.

Willcocks L., Feeny D. & Islei G. (1997) Managing IT as a Strategic Resource, McGraw-Hill.