Perspectives in Engineering Web-Enabled Electronic Commerce Systems

Xue Li

University of New South Wales

Follow this and additional works at: http://aisel.aisnet.org/pacis2000

Recommended Citation
http://aisel.aisnet.org/pacis2000/35

This material is brought to you by the Pacific Asia Conference on Information Systems (PACIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in PACIS 2000 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.
Perspectives in Engineering Web-Enabled Electronic Commerce Systems

Xue Li
School of Information Systems, Technology and Management
The University of New South Wales
Sydney, Australia

Abstract
Engineering the web-enabled electronic commerce system is to build web-based enterprise information systems for carrying out business transactions over the Internet. This engineering task is related to three aspects: the requirement specification, the Internet technology, and the development methodology. In the requirement specification, the business analysis and design is conducted to create a semantic business model that will reflect both the business and the system requirement. With the Internet technology, the modern information technology infrastructure is investigated in order to transform a business model into an implementation model. The system analysis and design will be performed and the architecture issues should be discussed. With respect to the development methodology, an efficient way to build enterprise information systems is addressed. This paper is to provide an overview of the problems, concerns, and the background in an effort to rationalize the engineering web-enabled electronic commerce systems.

Keywords: E-Commerce System Development, Benchmarking Business-Critical Applications, Web-Based Technology

1. Introduction
The engineering of web-enabled Electronic Commerce (EC) systems is considered in three aspects: the requirement specification, the Internet technology, and the development methodology. We will discuss the problems, concerns, and background related to these three aspects in an effort to rationalize the EC Engineering. Figure 1 illustrates this idea.

The requirement specification is a process that generates business requirement specifications and other necessary documents such as the explanatory files. The outcome is regarded as a business model. The process is for the business analysis and design that maps business information needs to the Internet technology.

The Internet technology is characterized by the Object-oriented technology, Internetworking, and the Client-server architecture (Umar, 1997). The system analysis and design techniques are used to derive an implementation model that is a mapping between the business model and the technological details with regard to the system components and the architecture.

The development methodology is applied to give a roadmap that shows the path of the system development. It provides answers on how to apply what technology on which business applications. There is a phenomenon that both business and the Internet may experience rapid changes. This requires the growth management to be built into the system development. Currently many development methodologies rely on the underlying software tools supplied by the major software market players.
A web-enabled **EC system** is an information system that provides services including:

- the operational business transactions carried out over the web,
- the ability to maintain system security and data integrity in a web environment, and
- the strategic business planning and decision support in advanced web-applications such as data warehousing, on-line analytical processing, data mining, and enterprise knowledge management.

![Diagram of the Engineering Process of Web-Enabled Electric Commerce Systems](image)

**Figure 1. The Engineering Process of Web-Enabled Electric Commerce Systems**

Many businesses are now engineering their information systems on to the Internet. The **EC Engineering** is about how to build a web-enabled enterprise information system for businesses. The engineering may also need to deal with the legacy systems that were developed without regards to the Internet. The trend has already begun that many businesses are now starting using web sites for their information needs. More and more software engineers are now employed to bring legacy systems on to the Internet.

The understanding of web-based businesses involves the understanding of the applicability of the technology. It is like a marriage. It requires the mutual understanding between the business requirement and the technology applicability. It may require a recursive refinement process during the system development. It is evident that the new innovative technology may impact on the business and change the business process. Therefore, the requirement specification in the EC Engineering can be affected by the Internet technology. If the requirement specification is subject to a recursive refinement process, the development methodology should also play a role in the requirement specification. In fact, these three aspects of EC Engineering are overlapped. For example, a prototyping methodology may be applicable to the system development to accommodate the recursive refinement of the system requirement specification. In this case, the prototyping is a reflective process that incorporates the system requirement dynamically. The EC Engineering should integrate the requirement specification, the Internet technology and the development methodology into a progressive and reflective process.

Comparing with the traditional information system analysis and design, we will give the viewpoints that reflect their current evolution in the EC Engineering.

The requirement specifications will be discussed in section 2. The benchmarking is discussed as the business-critical specifications. We then give some emphasise on the current EC technologies in section 3. In this section, we will compare and contrast some different EC system architectures. The methodology issues are discussed in section 4. The implementation of a system is to apply the Internet technology on the system design. We give an overview on the implementation techniques in section 5. The final conclusions are presented in section 6.
2. Requirement Specification

One of the challenges in the Information Age is that consumers are now accessing same information via web. This gives consumers a larger verity of options and thus businesses are facing a bigger and more competitive market.

The EC requirement specification is a process of the business analysis and design that is to create logistic models, process models, or semantic business models for the business-trading environment. This section will not be discussing on how to create those models but rather on the explanation of the problems and concerns in doing so.

In terms of business transactions, there are mainly two types of EC systems on the Internet, business-to-business and business-to-customer. In a business-to-business system, most transactions are automated by using EDI (Electronic Data Exchange) or EDI/XML (Plaplanete, 1998). The network traffic is predictable and stable. The transactions are mostly in batch mode. In a business-to-customer system, most transactions are manual. The network traffic is unpredictable and dynamic. Also the most transactions are in on-line mode.

Three major concerns in the EC requirement specification are that, the business vision, the assessment criteria, and the system viability. A successful business must have a vision. The perceptions that developers may have in their minds may be business-centric or customer-centric. The assessment criteria will be discussed in this section for benchmarking the web-enabled enterprise information systems. Four factors for the business-critical applications will be discussed. They are availability, reliability, security, and performance of the systems. Finally, the system viability in terms of the growth management will be mentioned as the ability of coping with changes.

2.1 Having a Vision: Business-Centric verses Customer-Centric Applications

A business vision is about the perception that we posses towards the success of a business. To have a vision is to identify the success factors and foresee the changes. It will fundamentally affect the design and the implementation of a system. We may view a system in many different ways. For example, it can be viewed on what and how a system does, or on changes a system can bring. In this subsection, we compare and contrast two opposite views on a business information system, from within the business and from its outside. The concepts are the business-centric systems and the customer-centric systems.

Business-centric systems are those systems specialized in business activities. They may require prerequisite knowledge and training of users. Traditionally, non-web business information systems are almost all business-centric because consumers do not need to directly interact with the computer systems. Operators of business systems are trained and designated. There is no need to create interfaces especially for the targeted consumers who want to visit the business via computer systems. Business is conducted normally through facsimile, telephone, mails, or face to face. In business-to-business type of systems, business-centric application still has its raison d’être because there are mostly still machine-to-machine batch transactions and the specialized interfaces. However when the Internet comes into play, the business-centric style increasingly becomes a problem to the general users. Customer-centric systems are applications that are designed in a way that is to put the customer needs first. The system will try to satisfy customer in all possible ways. Therefore a customer can use the system without much knowledge of computing. The investigations on
general web pull technology in the shopping activities or the web push technology in business sales may also improve the satisfaction of the customers.

2.2 Benchmarking: Business-Critical Applications

Successful EC Engineering requires an assessment benchmark that brings the confidence and better understanding to the EC. The building of business-critical applications requires the quality assurance. Under these assumptions, we identify availability, reliability, security, and the performance of a system as the critical factors for business applications. One needs to start from the analysis of business functions and specify the requirements for the system. A top-down design methodology can then be applied to bring a perfect marriage between business functions and Internet technology.

2.2.1 Availability

A system is available may imply that it is accessible, scalable, and attractive. Hence the availability of a system is a measurement of its accessibility, scalability, and attractiveness. The accessibility of a system harnesses the Internet technology for its business solutions. On the one hand, a system should be accessible for supplying a range of business services. On the other hand, a system should be accessible for providing a variety of business transactions. There should be no limit on supporting the businesses whether they are selling tangible or intangible goods, whether businesses are located in a shopping mall or as a warehouse-based distributor, whether they are the business partners in a suppliers chain or in a virtual enterprise. Moreover, different types of users of a system may have different purposes, such as for front-end business operations, for executive information needs, for decision supports, or for enterprise knowledge management. The accessibility can be achieved by a unified "on-the-web" standard user interface. It may provide users a great help on accessing information but may introduce security problems and other concerns. The Intranet and Extranet technologies can be applied as a leverage of business functions on the web.

The scalability of a system means that it is capable of being tailored to suit different environment or is flexible to satisfy different business needs. When considering the environment, the system may be scalable for deployment on mainframe machines, on personal computers, or on portable devices such as mobile phones. In this case the software should be platform independent and capable of dealing with different types of network traffics, and capable of processing different volume of data. When considering the business needs, the system may be scalable for different software configurations so that the system is installed to suit for particular business needs.

Availability may also refer to the attractiveness of the web site. The metrics of attractiveness consist of two factors: the publicity and the satisfaction. Both of these two factors are important for the system viability. Without the publicity, the system will have no users. Without the satisfaction, the system will be useless. The analysis of the web site attractiveness may start with the statistics of the user numbers. It may give the numbers of the first-time visitors, second-time visitors, or the number of customers who have done the business with the web site (and come back again). On the other hand, there may be many attributes that affect the attractiveness of the web site. The web data mining technique may be used to discover the significance of those attributes (see section 3.2).
2.2.2 Reliability

Reliability is about the robustness and soundness of a system. It can be operational related: so the system can be recovered from either software or the hardware crashes. It may also require the backups in order to recover from environmental disasters. On the other hand, it can be application related. It is very hard to have a system bug-free. In many cases the error is not easy to be classified as whether a logical error or a system error. A logical error could be a design error or a data error. In many cases, it may give no error message. A system error could be an operating system run-time error, network traffic problem, or a software-interfacing problem. A system error message may not indicate the real problem but just a report on the failure of a system operation. When an error is discerned, it is sometimes necessary to repeat an error in order to fix it. Theoretically, any error should be repeatable. However, the ability of repeating an error is very much dependent on the understanding of the error.

There are generally two ways to provide application reliability, the verification and the software test. These two ways may compensate each other. The verification is a process that uses the theorem proof technique to check the system specification against the system requirement, while the software test is a process that checks the system implementation against the system specification.

Other issues may also affect the system availability, such as the virus protection, the network security, and the system administration including recovery, security control, performance tuning, etc. In dealing with a changing world, the availability can also be affected by the system viability in coping with the changes. Over all, the availability is the ability that keeps the system constantly alive on the Internet.

2.2.3 Performance

The performance is a dynamic factor of the system. It is related to three aspects of a system. Firstly there is a cost-effective decision made in the system design phase that is to bring a best integration of the software and the hardware at a cost within the system development budget. This decision is meant to give the system performance a base line. Secondly, there is a system architecture problem. In a client-server paradigm, the network traffic and the network dynamics are both considered in providing a balance in the principles of the distributed computing. In considering the network traffic, the performance is a problem of the distributed database management in order to minimise the network traffic by using techniques such as RPCs (remote procedure calls) or data fragmentation and data replica cache techniques. In considering the network dynamics, the performance is problem of the network management that is associated to the network topology and the configuration to maintain the network reachability and the high-level interoperability. Thirdly, the performance is a system administration matter in the performance tuning process that is to find a balance of the trade-offs between the system storage space and the system responding time in a dynamic business environment. Within a changing environment, the performance problem may involve above three aspects recursively.

2.2.4 Security

The security problem of an EC system is about the authenticity, confidentiality, and data integrity for the web-enabled business transactions. It is very important and crucial to an EC
system. This problem is inherent in the Internet security and deserves a detailed discussion elsewhere.

2.3 A Viability Study: Growth Management

The growth management is about the strategies on how to deal with changes. The business reconceptualization is discussed in many research papers on how to re-build business in facing the opportunities and challenges in the Information Age.

Here is a to-do list:
- Anticipate and monitor changes in the business environment.
- Assess the changes and find out what the impact could be in both aspects of loses or gains in business.
- Understand the reason of changes and analyse the triggers or conditions of changes.
- Build the business model that is able to incorporate changes.

If the business model cannot reflect and represent the changes, the business model then needs to be changed. There are two ways to do so. (1) Change or redesign the business model. This involves the reverse engineering process. (2) Use a trial-and-error approach. This is to modify the system in order to adopt a change.

The growth management reflects the viability of a system. The strategies used to deal with changes can either be exploitative or defensive.

3. EC Technology

The EC technology is the application of the Internet technology on the building of EC systems. In this section we discuss three prominent issues: the web architecture, the web data databases and data mining, and the web intelligence.

3.1 Web Architecture

The web architecture concerns how the information flows among the system components and how the system components are organised. We will firstly give an overview on a progressive and reflective procedure of building the web business applications. This idea shows how the web architecture is established. Then, we introduce the basic concepts of front-end and back-end system components. Based on those concepts, the client-server paradigm will be discussed in the context of the information flow and the system software deployment.

3.1.1 Progressive and Reflective EC Engineering

The globalisation and deregulation are two main drives to the changes of the business environment. The impact from the growth of the Internet has also accelerated these changes. The EC Engineering is challenging these changes by using a progressive and reflective process to build the EC systems.

Figure 2 illustrates the idea that an EC system can be developed progressively with reflection of changes. The initial EC system may be a few primitive web information pages. Then the interactive features may be added on to establish a basic trading environment to allow customers to search for or purchase the goods and services. The operational business
transactions are supported at this layer. Web-enabled database applications are deployed at this layer. When a trust has been built on the system, a full-fledged business information system can be built on the web using Corporate Portals (Finkelstein, 2000). Corporate Portals are the web applications that provide a unified information gateway for distributing, analysing, consolidating, and managing information across and outside an enterprise. Corporate Portals are also called Enterprise Information Portals (EIP). The trend has already started that the Corporate Portals are used to unify the effort of the enterprises to integrate their information access methods. For example, Corporate Portals can be configured with data warehouses to enable different users to access information for their needs, such as Executive Information Systems (EIS), Decision Support Systems (DSS), On-line Analytical Processing (OLAP), and Knowledge Management Systems (KMS). Above the Corporate Portals layer, the Intelligent Agents (Wooldridge, 1995) can be built to retrieve, discover, reason, or deliver the knowledge for various needs. They can be used for business purposes such as buying and selling products, or be used for strategic purposes. For example, Intelligent Agents can be proactive to mine data in data warehouses in order to discover the trends or changes in a business environment; or be reflective in order to solicit the expert advises to review business policies. Although the enterprise knowledge is available at the Corporate Portals layer, it is about its accessibility and maintenance. While at the Intelligent Agents layer, the enterprise knowledge is available with the interests of its manipulation and application.

3.1.2 Front-end and Back-end Concepts

All web-enabled applications have two ends, front-end for the user support and the back-end for the system support. The front-end has the responsibility to:

- provide user interfaces,
- implement the representation logic,
- implement the application logic, and
- specify the interfaces with back-end components.

![Figure 2. The Progressive and Reflective EC Engineering](image)

The back-end includes all those invisible system components that must be used to support the functions of front-end. The back-end may include functions of:

- database management,
- data communication,
- multimedia management,
- security control (e.g. firewall).

It is important to distinct the system front-end and the back-end components. Front-end is application-oriented and business specific. It should be user-friendly, flexible, and competent...
for business functions. The tools used to develop front-end should be easy to learn, easy to use, yet functionally sufficient. While back-end is implementation-oriented and business independent. It should be highly efficient, standard, and versatile for supporting different requirements. For example, the front-end of an EC system can be built based on XML, VRML, or SMIL (Synchronised Multimedia Integration Language, from W3C, www.w3c.org) tools, and the back-end can be built based on CORBA (from ODMG) or ADO (ActiveX Data Objects, from Microsoft®), for relational or object-oriented databases. By the distinction between the front-end and back-end components we can develop an EC system separately and we have a better chance to organise the system for different business environments.

3.1.3 Client and Server Paradigm

In a client-server application, a client can initiate the information request and expect the server to reply. Here are two concerns: (1) the information flow between client and server, (2) the software deployment between client and server. When a web application is to be deployed on the Internet, there are at least three ways to deploy the software components: a thin client, a medium client, or a fat client (Stallings, 1997). This is decided based on the principle that to minimise the information flow between client and server and to maximise the system performance.

The client-server paradigm can be incorporated into two kinds of EC frameworks, the site-based or the agent-based frameworks. In a site-based framework, the information flow is basically in a request-reply pattern: the client asks for information, then the server replies. This request-reply pattern is execute as either RPC (Remote Procedure Call) or RDA (Remote Data Access). In this pattern, the essence is the data and data processing. It is called site-based because the value of information is decided at the site: the one who is asking for information. In an agent-based framework, the information flow is basically in a goal-satisfying pattern: the agent travels to the server and interacts with the server for the goal to be achieved. The agent carries the goal and the knowledge. The server provides the inference mechanism and information for agent to reason out the goal. In this pattern, the essence is the knowledge and goal achieving. It is called agent-based because the agent, who has the autonomy to make decisions on behalf of its creator, decides the value of information. An agent is characterised by its mobility, intelligence, and autonomy.

3.2 Web Database and Data Mining

This subsection is to consider the data and their meanings in a web environment. The web database is discussed in terms of the interconnectivity and interoperability. The web database mining is considered in terms of the knowledge discovering of rule-base patterns and topology-based patterns for the publicity problems and the attractiveness problems of web sites.

3.2.1 Web database

EC systems need database support as the back-end. This challenges a DBMS to have its data web-accessible. Web database is a database that has its data accessible for web applications. So the data can be inserted, retrieved, or updated from within a web addressable unit (e.g., an HTML document). A web addressable (i.e., hypertext-linked) unit is given as a URL (Uniform Resource Locator) that can be a document, a program of a business transaction, or
an interactive user action (e.g., sending email or downloading software). There are generally
two different ways to make a database web-enabled: a connection-based approach or a
content-based approach. In the connection-based approach, a standard protocol is proposed to
specify an interface between data and programs. Examples can be the CORBA architecture
using a Java-based programming environment to manipulate data over the Internet; or the
ADO architecture (Gutierrez, 2000) using Microsoft® proprietary ActiveX technique for the
data interfacing in the network environment. In the content-based approach, a standard
protocol is proposed to specify the meaning (content) of the data for different applications.
Examples are the ICE (Information and Content Exchange, www.w3c.org) and XML/EDI
(www.xmledi.com). In this approach, the data need to be extracted from the database
according to the content-based protocols before they are exchanged. So the interaction is
performed in terms of the document exchange instead of the interfacing.

The main difference between the connection-based and content-based approaches is that the
former is computing-centric, while the latter is business-centric. The connection-based
approaches are considered in terms of the independence between platforms and applications,
so those application programs can share and exchange data over different database
architectures. The essential point here is the interconnectivity of the databases. The content-
based approaches are considered in terms of the independence between business data and
business applications, so those different business applications can share and exchange
topology information for the suppliers chain, purchasing, manufacturing, shipping, auditing,
and other business transactions. The central point here is the interoperability that makes the
data understandable to each other applications.

In a top-down design viewpoint, the content-based approaches should be considered and
evaluated before we consider the web database implementations. However the engineering
concerns can always be constrained by the availability of the web database techniques for
their ability to support the content-based approaches.

3.2.2 Web Data mining

The web data mining (WDM) activity is to automatically discover rules, patterns, or
associations from the web-collected data (Mobasher, 1997). The WDM can be applied on two
categories of data recorded: the business transactions and the web-user patterns. The purpose
of the WDM can be either for the improvement of the business functions or for the
enhancement of the web site attractiveness. The data mining on the business transaction data
is the traditional data mining while the data mining on the user-web patterns is an emerging
research topic.

It may be useful to understand on not only how but also why user visits the web site. This
problem is the satisfaction problem of web sites. The topology-based web data mining
approach (Chen, 1998) can be applied to find out how and why. By viewing the frequently
visited topology pattern of the hypertext-linked web business components, we may find out
what business components are visited frequently and how they are visited.

3.3 Web Intelligence

Web intelligence is to apply artificial intelligence in the WWW (World Wide Web)
environment. To this end, the intelligence refers to the knowledge and the application of
knowledge in problem solving. Knowledge can bring change and knowledge itself may
change. If we view the Internet as a network of servers and clients, we can then see that the collection of servers and clients is a representation of a virtual world that reflects the image of our perceptual world to the business activities. In the virtual world, knowledge can be discovered, stored, transported, and applied.

4. Development Methodology

The EC development methodology is a procedural work that can be followed step-by-step to create the EC systems. In this section, we are not going to give the detailed procedures but to discuss some issues that can affect the application of proper procedures. Firstly we discuss the modeling methodologies that provide the formal method for catching semantics of EC systems. Then we discuss the methodologies that are used to make legacy systems web accessible. Finally, we give an overview on the system development methodologies that are applicable to different engineering environments.

4.1 Modelling of EC Systems

The modeling process has two general questions to be answered: the semantic completeness and the representation uniqueness. By the semantic completeness, two aspects are considered: (1) everything in the business-trading environment will be described; (2) anything expressed within the resulting model is true in the business-trading environment. By this two-way checking we can be sure that the business semantic model derived from the modeling process is a true representation.

By the representation uniqueness, two aspects are concerned: (1) the semantic non-ambiguity is maintained; (2) the minimum representation is achieved. The semantic non-ambiguity requires that the typing system enforce both the unique naming convention and the elementary domain values. Therefore complex values can be constructed from elementary values without semantic ambiguity. The minimum representation requires that all derivable artifacts cannot be directly stored but represented as formulas or the derivation rules, so that there is no redundancy in the system representation. The minimum representation will guarantee that the smallest number of syntactic units is used to produce the model. So this ensures a way of standardization for the modeling process. By checking on the representation uniqueness we can be sure that the business semantic model derived from the modeling process is a good representation.

There have been many modeling methodologies proposed in recent years. The current consensus is the UML (Unified Modeling Language) promoted by the OMG (www.omg.org). The UML can be used to visualize, specify, construct, and document artifacts of the systems ranging from enterprise information systems to distributed web-based applications (Booch, 1999).

4.2 From a Legacy System to a Web-Enabled System

Many methodologies available currently are:
- the traditional software design and implementation methodologies which have litter support for the system evolution towards the Internet information systems, or
- the software tool-based design and implementation methodologies that are applicable only on that tool. This leaves litter room for a software engineer to minimise his/her effort in transforming a legacy system into a web-enabled system.
The legacy systems are defined and viewed in many different ways (Alderson, 1999 and Umar, 1997). In our discussion we regard the legacy systems as the systems that are disadvantaged by not having their data and functions web accessible. So, the question now is to consider how we can convert the legacy systems into the web-enabled systems. There are two fundamental issues: (1) How do we make legacy data web accessible? (2) How do we make legacy systems run in the web environment? The former is a data conversion problem; the latter is a system integration problem.

4.3 Methodologies in System Development

Many different approaches to EC system development currently exist. These approaches are applicable to different situations in the system development. Four different system development methodologies according to the focus of the developers are commonly used: user-driven, data-driven, process-driven, and system-driven.

The user-driven methodology is to let the user to play the important role in the system development. It is an iterative process that is to build a prototype first and then the user will be working with the developer to improve the system functions. This approach is usually focused on small applications and the system can be quickly built and refined.

The data-driven methodology is used as a formal top-down refinement methodology. Following down a few successive stages, the system is developed. This approach starts with the formal specifications of the system requirement. Then the data model is derived. Based on the data model, the system transactions are defined and implemented. This methodology provides a chance to perform the verification for checking the system specification against the requirement. The data-driven methodology follows a formal software-engineering path and is good for the large systems that have clearly specified system requirements.

The process-driven methodology is used for the systems that consider the understanding of the system functions is more important than that of the data structure. In this kind of systems the fundamental processes are identified before the specification of a data model. This methodology is suitable for the systems that have complicated functions but have relatively simple data structure.

The system-driven methodology is applied on the existing system to reverse engineer it to a new one. By the reverse engineering, the new system interfaces and added functionality will be integrated with the old system. This will minimise the effort of creating a new system from the beginning. The new requirements can be compared with the old requirements to improve the system quality.

In the EC Engineering, all these four methodologies may be applicable. For example, a prototyping approach may be a good way to quickly develop a system for the demonstration of ideals. But it may be difficult for the further development regarding the change of the underlying system architecture. A reverse engineering approach may be helpful on re-developing a new system similar to an old one. But it is difficult to formally verify the new system because the reverse engineering may not be able to guarantee that the reversed system will be a true recover of the old one. Sometimes, the reverse engineering may have to “reinvent the wheel”, in which the problems were overcome before.
5. Implementation

In this section we consider how we can implement an EC system properly. Firstly, we discuss the independent relationships existing in the implementation environment. The understanding of this may help us on selecting tools and back-end systems for the implementation. Then, we discuss the component technology that is used to construct a system.

5.1 Implementation Independence

When the system development comes to an implementation phase, it faces an intricate task: the decisions on building tools, on adherence of protocol standards, and on system construction. The complexity consists in the diversity of the current available technological products and the proprietary protocols that are all tangled. The following independent relationships are identified to help us to consolidate the decisions in an effort to unify the implementation process.

- Front-end and Back-end system independence,
- Platform independence, and
- Proprietary protocol independence.

The front-end is independent from the back-end so that the implementation work can have different focuses as mentioned in subsection 3.1. However, this means we have to design a standard interface for the data flow between front-end and back-end systems. For example, XML can be used to specify Corporate Portals (Finkelstein, 2000) as front-end and access to the back-end of standard relational or standard (ODMG) object-oriented databases. Another example is to use some scripting programs embedded in an HTML document (e.g., CGI, Java applets, or ActiveX components) in the front-end system and use CORBA (www.odmg.org) or ADO (Gutierrez, 2000) for the back-end support for accessing databases. It may be worthwhile to mention that some front-end supporting functions may have minor compatibility problems such as that between the Netscape and the IE (Internet Explorer). In this case, the front-end development should limit the functions to use only those compatible features.

Platform independence is implemented by choosing a development tool that is supported by all operating systems. It is not difficulty for implementing the platform independence as long as we use OMG/ODMG Standard and Java, XML, tools. Many off-shelf tools are now providing the developers a freedom to implement systems on different platforms.

Proprietary protocol independence is an effort to improve the interconnectivity or the interoperability of the system for the potential incremental development. Many software companies provide the tools that are not designed for an open system environment. Although they may provide some API (Application Program Interfaces) for standard protocols, the integration of these tools may require an extra effort on leaning and programming of the API, and an extra layer of the system software for the bugs.

5.2 Component Technology

The component technology (Szyperski, 1998) is developed based on the assumption of the open system environment. This technology is characterised by the reusability, portability, and the interoperability within an open system environment. A component is an independent and portable program that has its functions defined within the component interface. A component
interface consists of interface elements that are the ontological descriptive notions of the interface including the syntax, semantics, and the functions of the component. A component has ability to response to the events that trigger the execution of the component or the interaction with other components. Components can co-operate together in a pre-designed architecture to perform the tasks of data processing or the process control.

In building the EC systems, the component-based system development may start with the transformation from a business model into a component-based architecture. Then the system is assembled based on the transactions in the business trading. Currently the available tools are the EJB (Enterprise JavaBeans) from Sun Microsystems® and the ETS (Enterprise Transaction Server) from Microsoft®. Both of them are used for the component-based and transaction-oriented applications. In EJB, transactions are packaged as the JavaBean objects. In ETS, transactions are packaged as the COM (Component Object Model) objects.

The future however, is still considering using the component technology in a higher lever of the component packaging that is to group the components in terms of the business types instead of the transactions. Furthermore the packaged components should be web addressable so a business (e.g., a virtual enterprise) can be established based on the systems that are already functioning. We may call this business-oriented component technology as Web Addressable Business Objects (WABO). Since the WABO is web addressable, it inherits all properties from the component technology plus that it is interconnected and interoperable. In designing a WABO-based system, we need to concentrate on the business requirements and the business functions. After that, the WABO-based system should be generated automatically for the implementation of the business model.

6. Conclusions

The ultimate goal of the EC Engineering is to produce robust, competent, and viable web-enabled enterprise information systems. In this paper, we have tried to reason out some important factors in building such systems. A scenario of EC Engineering may start with the creating of a business model in the process of business analysis and design. Then transform it into a system model that has its main interests in the system architecture and components. Finally the system implementation will be carried out. Within this scenario, three aspects are considered: requirement specification, Internet technology, and development methodology.

To achieve the success in EC Engineering, this paper has given an overview on some current important issues. It aimed to provide a better understanding on the questions such as:

- What is the EC Engineering about?
- What are the general requirements for an EC system?
- How do we benchmark an EC system?
- How do we develop such a system?
- How do we implement such a system?

We consider EC Engineering as a progressive and reflective process. Within this process, we should not only understand what we are doing but also manage what we are doing. This paper has given some rationale on the issues of this process. Many EC tools provided by major software competitors are still weak in interoperability and interconnectivity mainly because of their proprietary protocols. It is expected that in a near future we may be able to build EC systems faster and easier by assembling systems with the plug-and-play WABO.
References


