Component-based E-Commerce: Assessment of Current Practices and Future Directions

Martin Bichler
Department of MIS
Vienna University of Economics and Business Administration
Augasse 2-6, A-1090 Wien, Austria

Arie Segev
Haas School of Business, University of California &
Lawrence Berkeley National Laboratory, Berkeley, CA 94720

J. Leon Zhao
School of Business and Management
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong

Abstract

Component-based e-commerce technology is a recent trend towards resolving the e-commerce challenge at both system and application levels. Instead of delivering a system as a prepackaged monolith system containing any conceivable feature, component-based systems consist of a lightweight kernel to which new features can be added in the form of components. In order to identify the central problems in component-based e-commerce and ways to deal with them, we investigate prototypes, technologies, and frameworks that will transcend the current state of the practice in Internet commerce. In this paper, we first discuss the current practices and trends in component-based electronic commerce based on the International Workshop on Component-based Electronic Commerce. Then, we investigate a number of research issues and future directions in component-based development for electronic commerce.

1. Introduction

E-commerce has been embraced by many as the new frontier of information revolution, and millions of dollars are being invested in Web-based electronic commerce systems (Dogac, 1998; Shaw, Gardner, and Thomas, 1997). However, the development of electronic commerce applications today is considered expensive and risky because of many technological limitations, such as the absence of application-level interoperability, industrial standards for electronic trading and large-scale reusability of electronic commerce applications.

A current trend is to move towards object frameworks or component-oriented programming. Instead of delivering a system as a prepackaged monolithic software system containing any conceivable feature, component-based systems consist of a lightweight kernel to which new features can be added in the form of components. Distributed object standards as well as document-centric electronic commerce standards enable easier inter-operability of electronic commerce applications in a heterogeneous environment.

In the software industry, components are referred to as a specific piece of functionality that can be accessed by other software through a contractually specified interface. They are self-contained, clearly identifiable artifacts that describe and/or perform specific functions (Sametinger, 1997). The basic capabilities of component-based systems should include the plug and play features at various granularities, interoperability across networks, portability on different hardware and software platforms, coexistence with legacy applications, mobility in various networking environments such as Internet, Intranet, and Extranet, and ability of self-managing data resources. The electronic market place requires higher level components that deliver particular business functionality. Inter-operability at this level is also required.

In order to identify the central problems in component-based e-commerce and ways to deal with them, we investigate prototypes, technologies, and frameworks that will transcend the current state of the practice in Internet commerce. In this paper, we first investigate the current practices and trends based on the International Workshop on Component-based Electronic Commerce. Then, we develop a framework of concepts and issues for research in component-based e-commerce. Finally, we indicate our view on future directions in component-based development in the context of electronic commerce.

2. International Workshop on Component-Based Electronic Commerce

An International Workshop on Component-based Electronic Commerce (CEC'98) was held on July 25, 1998 at the Fisher Center for Management and Information Technology, Haas School of Business,

University of California, Berkeley. The main objective of the workshop was to identify the central issues of component-based e-commerce and ways to deal with them by gathering researchers and developers from industry, government, and academia to report their experiences with setting up, running, and maintaining electronic commerce systems. In the workshop, researchers presented prototypes, technologies, and frameworks that reflect the current state of the practice in Internet commerce. The workshop included four sessions and a closing panel, the focal problems and topics of which are listed below (The workshop can be accessed at http://haas.berkeley.edu/~cmit/cec/).

Session 1: E-Commerce Components and Frameworks

Problem:

A large proportion of today's electronic commerce applications is custom developed. Custom development is risky and expensive. There is little or no reuse even for very generic functionality like order entry modules or payment servers as many software modules are developed to run with a certain Web server or database server and cannot simply be "plugged" into another environment.

Topics:

- Examples for reusable software components in e-commerce
- Experiences with commercial component models like JavaBeans, Java Servlets, DCOM, Oracle Cartridges, etc.
- What are the requirements for plug-and-play component solutions?
- Architectural properties: What are "ilities", i.e., some property added to the system that is independent of the functionality of that system (reliability, availability, scalability, evolvability, ...)?
- Are today's fast moving component standards enabling or invalidating the idea of components for e-commerce?

Session 2: Business-to-Business E-Commerce

Problem:

Business-to-business e-commerce is an area where e-commerce will have the most major impact in the short term. Currently, business-to-business e-commerce is dominated by EDI type applications and long standing relationships. IT is expected to enable ad-hoc collaboration of companies, virtual enterprises and streamlined supply chains.

Topics:

- What are special requirements for B2B e-commerce?
- Which business models are likely to evolve?
- Experiences in supply chain management, procurement, catalog management?
- How can e-commerce components contribute to this domain?
- What are good scenarios for integration of components with legacy systems?
- What is the role of agents, market mechanisms and automated negotiations and their potential for facilitating business-to-business commerce?

Session 3: E-Commerce Interoperability and Standards

Problem:

One of the most obvious problems of Web-based electronic commerce systems is the lack of high-level inter-operability. Web sites format their CGI requests and HTML outputs in vastly different and often changing ways, each of which must be processed differently. There are no widely accepted high-level standards for electronic trading.

Topics:

- How can we communicate needs (e.g. buyers) and capabilities (e.g. sellers) in electronic marketplaces?
- New electronic commerce protocols XML/EDI, OBI, OMG's IDL, OTP, X.12, EDIFACT ...
- Approaches to protocol negotiation and translation
- Product taxonomies: How should product taxonomies deal with product innovations and complex products?

Session 4: Business & Software Components

Problem:

The development of an inter-operable component-based E-commerce infrastructure will impact the design and use of high-level business objects, processes and architecture within companies, as well as the software development process in general.

Topics:

- Impact of component-based technology and market place infrastructure on business objects development
- Changes in the development methodologies of Network Centric Component-based Applications
- Examples of component use in higher lever objects (e.g., decision support, and brokering systems)

Closing Panel "A Marketplace for E-commerce Components - What is missing?"

Topics:

- Do we face a market for e-commerce components of third party vendors or will packaged software of a few vendors become the predominant form?
- Can e-commerce interoperability standards be flexible/extensible enough for the fast changing requirements in e-commerce?
- How can we standardize e-commerce protocols / Who should standardize protocols?
- Which functionality is inadequate or missing for future electronic market places?
- Do we have a clear picture of the business requirements?

3. Fundamental Concepts of Component-based E-Commerce

Component-based E-commerce is a combination of several major information technologies. In this section, we analyze a number of fundamental concepts that are useful for understanding the issues in component-based e-commerce that are discussed subsequently. This summarizes also many discussions during the Workshop on Component-based E-commerce and reflects some of the core issues in this emerging field.

3.1. Object-Oriented Programming and Component-Based Programming

Newer software engineering technologies are based on the principle of making the expression of ideas simpler and more compact. Especially the renewed popularity of object-oriented programming concepts like encapsulation, information hiding and polymorphism in the early 80s raised the level of abstraction in problem formulation. Object-oriented programming constructs business systems in terms of objects that represent things in the real world naturally and effectively, thus making the software system easier to understand for designers, programmers, and users. Object-oriented programming has led to a revolution in software development and is making software reuse an easier task compared to older programming concepts. Object frameworks are one step ahead as they also reuse designs for specific problems. An object framework is a collection of cooperating objects that provide an integrated solution (customizable by the developer) within an application or technology. The components of an object framework are not intended to work alone. Both industries and universities around the world have been moving towards object-oriented programming.

Component-based programming focuses on building and packaging robust, extendible, and flexible components to allow reusers to build their systems more quickly, more effectively, and less expensively, which is another step forward towards more and better software reuse and towards higher software productivity. A component framework defines rules for independently developed and dynamically loadable components, rather than for classes that are linked together. In contrast to most object frameworks, component frameworks are black-box frameworks, i.e., frameworks that can be used without access to their source code and are extended through composition. Component-based programming requires a whole new set of methodologies for software modeling, design, development, and implementation, and will change the ways software are developed and reused (Jacobson, Griss, and Jonsson, 1997).

3.2. Containers, Wrappers, and Mediators

Several techniques can be used to deal with incompatibility between components and systems, including container, wrapper, and mediator. A container refers to a piece of application software that understands certain types of components and can provide an application context for these components. Examples of a container includes the Oracle Application Server that can provide services to application components called cartridges, which are tied to the server using industry standard protocols including CORBA, Java, and HTTP.

Wrappers are used to bridge mismatches between various components of a software system. A wrapper refers to a small piece of software that hides certain portions of the component interface that may have undesired effects on the software system. An example of wrapper application is to integrate a Java application into a DCOM framework by a wrapper around the Java program so that it becomes DCOM-aware.

Interfaces between incompatible components can also be provided by a mediator such as an intelligent agent through a proper coordination among different components. The main difference among mediator, wrapper, and container lies in the way communication and coordination are done; a container can understand the components directly, a wrapper hides the heterogeneity between

components, and a mediator passes messages and helps with negotiation between two components.

3.3. Black-Box Versus White-Box

Off-the-shelf components are often referred to as black-box components because in general they do not allow for customization and are used usually as is (Pour, 1998). One successful example of black-box framework is based on Microsoft's Visual Basic that offers a large set of prebuilt components called Visual Basic Controls (VBXs) that are developed by both Microsoft and third-party vendors. Black-box frameworks can also be a code reuse strategy within an enterprise that encourages or requires the reuse without modification of carefully built components (Jacobson, Griss, and Jonsson, 1997). On the other hand, software components can also be built based on object frameworks that rely heavily on implementation inheritance that can only be achieved through detailed understanding of the source code of the objects. The difficulty with these object frameworks is that they are abstract "designs" whose reuse requires customization by expert programmers (Thompson et al., 1997). This type of object frameworks is referred to as white-box framework (Segev and Bichler, 1998).

3.4. Document-Centric Versus API--Based Interoperability

Interoperability between software components within an application is achieved through component models like CORBA, DCOM and JavaBeans. Here interoperability is achieved through the standardization of component interfaces or application programming interfaces (API's). The classical example is OMG's Object Management Architecture. Interfaces of CORBA components are defined with the Interface Definition Language (IDL) and interfaces of generic components are standardized by the OMG. Several approaches try to achieve interoperability of electronic commerce components. OMG's Electronic Commerce Domain Task Force (ECDTF) tries to standardize interfaces of catalogs, brokers, agencies and other components for electronic markets. This should lead to a third party market of vendors for these electronic commerce components and to interoperability also between systems of different market participants. A good example is the Workflow framework, which specifies the API protocols between various components in a typical workflow system as defined by Workflow Management Coalition (WFMC, 1994).

Another approach tries to achieve interoperability between electronic commerce applications through the standardization of documents and document exchanges. Some of the first approaches in this direction come from an Internet Engineering Task Force (IETF) workgroup covering EDI (EDIINT), which has recommended standards for secure, interoperable electronic data interchange over the Internet. Many new approaches use XML, the eXtensible Markup Language created and developed by the W3C XML Working Group, as an underlying basis. The language combines the simplicity of HTML with the computability of EDI Standards and is meant to make transaction sets easier to define and use across companies. XML-centric interoperability enables computer systems to exchange documents – invoices, loan applications, contracts, insurance claims, and so on, as Electronic Data Interchange has tried to achieve for many years.

XML is intended for use on the World Wide Web and retains features such as vendor independence, user extensibility, complex structure, validation, and human readability. Microsoft and Netscape have promised to provide XML parsers in future generations of Web browsers, and other companies such as Veo Systems (Meltzer and Glushko, 1998), a spin-off of CommerceNet, are rushing to develop common business libraries built on XML.

Compared to document-centric interoperability, API-oriented interoperability is difficult to achieve across various different application domains although it may work well in some application domains. However, realizing document-centric interoperability may require many years to mature as it takes time to unify the domain specific vocabulary and common business libraries.

4. Research Issues of Component-based Electronic Commerce

In this section we try to identify some of the research issues in the field of component-based electronic commerce. We focus on three specific aspects, namely e-commerce frameworks and standards, development methodologies and business components.

4.1. E-commerce Frameworks and Standards

New e-commerce frameworks are evolving rapidly, however, few have many clues on what and which will become the industrial standard(s). These

standards and frameworks are tackling different aspects of electronic commerce. Therefore, it is of great value to trace the various e-commerce frameworks and compare their characteristics, intended applications, advantages and limitations. A research framework for comparing existing and studying new e-commerce frameworks and standards is urgently needed to provide researchers and developers with evaluation metrics and R&D directions.

As we have seen, in the previous section, interoperability is a critical enabler for the development of next generation of electronic commerce applications. Several interoperable e-commerce frameworks are taking place, rooted in one or more business consortiums in the last couple of years. These frameworks also impact research in electronic commerce. In the next paragraphs we introduce some of the most advanced approaches.

- **Open Trading Protocol (OTP).** OTP is a protocol for interoperability of electronic purchases on the Internet that encapsulates payment protocols and offers/invoices/receipts for payment and delivery. OTP is focused on 3-way interchanges among consumers, merchants and support services. The Open Trading Protocol Consortium, a group of over 30 companies lead by Mondex, has released a draft standard (currently version 0.9 – might already be 1.0 at the time of publishing) aimed at retail trade on the Internet.

- **Open Financial Exchange (OFX).** Open Financial Exchange is a broad-based framework for exchanging financial data and instructions between customers and their financial institutions. It is an open specification that anyone can implement based on widely accepted open standards for data formatting (such as SGML and XML), connectivity (such as TCP/IP and HTTP), and security (such as SSL). OFX is backed up by the Banking Industry Technology Secretariat (BITS), consisting of CheckFree, Integrion, IBM, Intuit, and Microsoft.

- **Open Buying on the Internet (OBI).** The OBI consortium released the OBI standard version 1.0 in May 1997 as an open, flexible framework for business-to-business Internet commerce solutions. OBI is intended for high-volume, low-dollar amount transactions, which accounts for 80% of the purchasing activities in most organizations. These types of transactions are typically for indirect materials including commodity goods and services such as office supplies, scientific supplies, maintenance supplies, PCs, etc. The business model of the OBI framework assumes that buying organizations only have one or a few approved selling organizations for each category of commodity items.

4.2. Development Methodologies for E-Commerce Applications

The past few years have seen several major information technologies come together to produce viable electronic commerce systems. EDI, WWW, cryptography, database servers, Java applets and distributed object standards form a bewildering mix of techniques and standards for the development of electronic commerce applications. Development of such applications is considered expensive and risky.

Recently, component-based software development started to strongly influence this area. It is important to define new techniques for the design and implementation of electronic commerce applications. Component-based software engineering provides many useful concepts for the development of electronic commerce applications. Component-based e-commerce development is leading the change in component-based software development, and therefore, research on development methodologies for component-based e-commerce will have a far-reaching impact on the whole software industry. A component-based software system life cycle has been proposed that includes the following major activities (Pour, 1998):

- Requirements analysis
- Software architecture selection, creation, analysis, and evaluation
- Component evaluation, selection, and customization
- Integration
- Component-based software system testing
- Component maintenance and upgrade

Component-based e-commerce methodologies can be developed similarly, but with a more specific focus on electronic commerce.

4.3. Computational Business Process Components

Computational business processes have been considered as software components for Electronic Commerce (Scacchi and Noll, 1997). Business Process Components are on a higher level of abstraction and focus on the business needs of applications. These computational components can be configured into an organizational process architecture for developing e-commerce applications and can be reused in process-driven Intranets and

Extranets for distributed development and run-time support. The vision for reusable process components is to enable the design, integration, and enactment of virtual enterprises, eventually leading to a community of virtual enterprises and virtual markets.

These components can model, support and execute business process activities within an organization, and can also interconnect and coordinate business processes across organizations, such as to support inter-organizational workflow. As a framework, it is possible to compose appropriate business process components that provide computer-based support for common business processes such as purchasing, accounts payable, accounts, receivable, and other corporate financial operations. Such a framework of business process components may then be reused and specialized in different organizational settings, thus providing a foundation for reusable software components for e-commerce.

4.4. Intelligent Agents as Part of Component Technologies

One of the new trends in component-based software development is to use intelligent agents as the gluing components within a component framework. Intelligent agents are thought of as a new candidate for providing interoperability in a volatile and dynamic environment where interactions among ad hoc market players are difficult to plan. According to Hedberg (1996), intelligent agents are autonomous software entities that can navigate heterogeneous computing environments and can, either alone or working with other agents, achieve some goal. Thus, they require on-board intelligence to achieve their task, such as planning, reasoning, and learning algorithms (Franklin and Graesser, 1997).

One important feature of intelligent agents is the ability to provide services via negotiation and to coordinate among multiple agents (Jennings et al., 1998). While research and development of intelligent agents have met with some remarkable success, notably in Internet search and electronic market simulation, the application of intelligent agent technology in component-based e-commerce remains an open research area.

4.5. Database Support for Component-based E-Commerce

As mentioned by Larry Ellison the CEO of Oracle, "The days of pure database market is out forever, and in is the integrated software market". Researchers of database community may have to look outwards to find new and exciting research problems. E-commerce might be one new business domain that can provide such opportunities.

4.5.1. New Roles of DB Management Systems

Databases provide a foundation for many types of information systems and can be thought of as the cornerstone of modern information technology. With the emergence of component-based software development, the role of database management systems may also change. The traditional view of information system development is to think of a database management system as the platform and the information systems developed as applications of databases. This view may prove to be limiting in component-based software engineering and new views might emerge that take the e-commerce applications as the principle system and databases as servant components.

New concepts in future databases are evolving towards support for electronic commerce. For instance, evolving databases are suggested to support negotiations in electronic commerce (Fordham, Abiteboul, and Yesha, 1997). The term "evolving" stresses the extremely dynamic nature of a negotiation, which requires to change the product descriptions, orders, and even protocols of negotiation on the fly. Another example is supporting EDI via distributed databases (Adam et al., 1998), which support the exchange of EDI messages through database transactions. As a result, the application-to-application communication is replaced by database-to-database communication.

4.5.2. Database Support for Cooperative Agents

New distributed database technologies may be needed for providing scalable, mobile, and inexpensive database systems as components for electronic commerce. For instance, mobile and autonomous intelligent agents need to carry their own data resources. Small, simple, yet efficient databases might be needed for intelligent agents or their agencies for cooperative problem solving. To support a set of cooperative computer agents, Berndtsson, Chakravathy, and Lings (1997) proposed the use of active databases as an enabling technology for cooperative information systems. The business model of database management system might change in this case since the database services may be provided by third parties that are not owners of the intelligent agents.

4.5.3. XML-based Document Databases

The document-centric interoperable e-commerce frameworks will rely heavily on database services to deposit and access the common business library and the domain specific information. These document databases will provide new challenges to database research since their requirements are different from the conventional business transaction databases. The data in XML-based documents are likely to be semi-structured and the transactions to the documents are mostly read-accesses. Although there are already several applications on the market that store XML documents in relational databases it is not clear if relational database is the best solution for XML documents.

4.5.4. Database Support for Large-scale Repositories of Business Process Components

The data for business process components discussed above need to be stored in large-scale repositories such as data warehouses. However, computational components require specialized supports that are not needed in document repositories. It would be interesting to study various forms of data related to computational components in both compilation time and runtime. This will amount to a computing environment where business programs are stored, developed, sold, and used. Exciting research topics are bound to emerge in this new area.

5. Concluding Remarks

Component-based programming with Microsofts ActiveX/COM or JavaSoft's JavaBeans has quickly become a popular programming paradigm of choice. These component models have a historical origin in the assembly of user interfaces, where they are used to route user interface events such as mouse clicks and keyboard entries. The most interesting future component markets will probably be for business components (often called business objects) on the server side, rather than on the client side where the margins are much lower and the requirements more generic. They are modeled after the real world to provide certain business functions like Customer, Order or Product.

The demand for large-scale, complex, and highly maintainable e-commerce systems is creating a fast growing market of third party components. From the buyers' perspective, this requires an application for component-based software systems life cycle that includes software architecture selection suitable for component-based system construction and component evaluation, selection, and customization. From the sellers' perspective, the components being built must conform to open standard, can be used under multiple frameworks, and can support a broad range of business applications. At the moment, two groups of component markets are evolving around the Java-based architecture led by Sun Microsystems, IBM, and others, and Visual Basic and ActiveX-based architecture supported by Microsoft.

In this paper, we studied the current practices in component-based e-commerce in industries and analyzed the future directions in its research and development. Our findings indicate that component-based e-commerce is a promising area for research, development, and application. All signs point to a trend of rapid change as numerous frameworks, standards, and systems in e-commerce mushroom. Many research issues are emerging in the general area of component-based software development and in electronic commerce such as e-commerce frameworks, standards, interoperability, etc. Furthermore, we also identified several research areas with a database focus.

References

1. Adam et al. "EDI through a distributed information systems approach", *Proceedings of the 31st Annual Hawaii International Conference on Systems Sciences,* Hawaii, USA, January 6-9, 1998.
2. Berndtsson, M.; Chakravathy, S.; and Lings, B. "Extending database support for coordination among agents", *International Journal of Cooperative Information Systems,* Sep.-Dec. 1997, Vol. 6, (no. 3-4):315-39.
3. Dogac, A., "A Survey of the Current State-of-the-Art in Electronic Commerce and Research Issues in Enabling Technologies", *Euro-Med Net 98 Conference, Electronic Commerce Track,* March 1998.
4. Franklin, S. and Graesser, A. "Is it an agent, or just a program? a taxonomy for autonomous agents". *Intelligent Agents III. Agent Theories, Architectures, and Languages. ECAI '96 Workshop (ATAL) Proceedings* (1997) p21-35.
5. Fordham, B.; Abiteboul, S.; and Yesha, Y. "Evolving databases: an application to eletronic commerce", *Proceedings of the International Database Engineering and Applications Symposium,* Montreal, Canada, August 25-27, 1997.

6. Hedberg, S. "Agents for sale: first wave of intelligent agents go commercial", *IEEE Expert* vol.11, no.6 (Dec. 1996), p16-19.

7. Jacobson, I., Griss, M., and Jonsson, P. *Software Reuse: Architecture, Process, and Organization for Business Success,* ACM Press & Addison Wesley Longman, 1997.

8. Jennings, N.R., Norman, T. J., and Faratin, P., "An agent-based business process management". In *this issue.*

9. Lewandowski, S. M., "Frameworks for component-based client/server computing," *ACM Computing Survey,* Vol. 30, No. 1, March 1998.

10. Meltzer, B., and Glushko, R., "XML and electronic commerce: enabling the network economy," In *this issue.*

11. Pour, G, "Component-based software development approach: new opportunities and challenges", *Proceedings of the 26th International Conference on Technology of Object-Oriented Languages and Systems (TOOLS USA), 1998.*

12. Sametinger, J. *Software Engineering With Reusable Components*, Springer-Verlag, Berlin Heidelberg, 1997.

13. Scacchi, W. and J. Noll, "Process-Driven Intranets: Life-Cycle Support for Process Reengineering," *IEEE Internet Computing*, 1(5):42-49, September 1997.

14. Schlueter, C.; Shaw, M.J. "A strategic framework for developing electronic commerce." *IEEE Internet Computing*, Nov.-Dec. 1997, vol.1, (no.6):20-8.

15. Segev, A. and Bichler, M. "Component-based electronic commerce", *Handbook of Electronic Commerce, 1998.*

16. Shaw, M.J.; Gardner, D.M.; Thomas, H. "Research opportunities in electronic commerce." *Decision Support Systems*, Nov. 1997, vol.21, (no.3):149-56.

17. Thompson, C., Linden, T., and Filman, B., "Thoughts on OMA-NG: the next generation object management architecture," (08/1997), http://www.objs.com/staging/OMG-OMA-NG.html.

18. WFMC, "Workflow reference model." Technical report, Workflow Management Coalition, Brussels, 1994.