Developing Collaborative Commerce System Based on Roles and Components

Hwagyoo Park[ac], Woojong Suh[b] and Heeseok Lee[a]

a. Graduate School of Management, Korea Advanced Institute of Science and Technology, 207-43, Chongryangri-dong, Dongdaemun-gu, Seoul 130-012 Korea
Tel: +82-33-639-0344, E-mail hkpark@kl.ac.kr
b. Division of Business Administration, College of Business Administration, Inha University,
c. Division of Management Information System, College of Management, Kyungdong University,

Abstract

Due to the growing competitive and more global pressures, firms are compelled to adopt collaborative commerce philosophy to create and sustain a competitive edge. The c-commerce demands extensive interactions among multiple-stakeholders with different core competences and roles, aiming at a common goal. From this motivation, This paper proposes a role-driven component-oriented design methodology (RCOM) for developing c-commerce systems. The methodology consists of four phases: collaboration analysis, component analysis, component design, and implementation.

Keywords: Collaboration, Role, Component

1. Introduction

Many companies have sustained their success by continually interacting with their business partners. Collaborative commerce (c-commerce) is a paradigm in which a variety of business stakeholders collaborate interactively via the Internet and the related integration technologies. When we look at our business through this stakeholder lens, we can often see what we are blind to: untapped opportunities to serve the business in ways that fundamentally change our economics. This collaboration results in an agile and highly integrated 'virtual' enterprise [9]. The stakeholders may collaborate on product designs, procurement plans, demand forecasts, manufacturing schedules, distribution activities and transportation movements. A primary objective is to harness product information and application assets accessible to the stakeholders in a commerce community including manufacturers, distributors, dealers, service groups and end consumers [3, 15].

The requirements for implementing c-commerce are usually complex and ever-changing [15]. For more successful c-commerce, the stakeholders need to share valuable intellectual assets and develop community-oriented strategies to collaborate with each other more effectively. Accordingly, the concept of c-commerce may be understood in terms of (i) interactive activities, (ii) strategy focus and (iii) intellectual assets, as depicted in Figure 1.

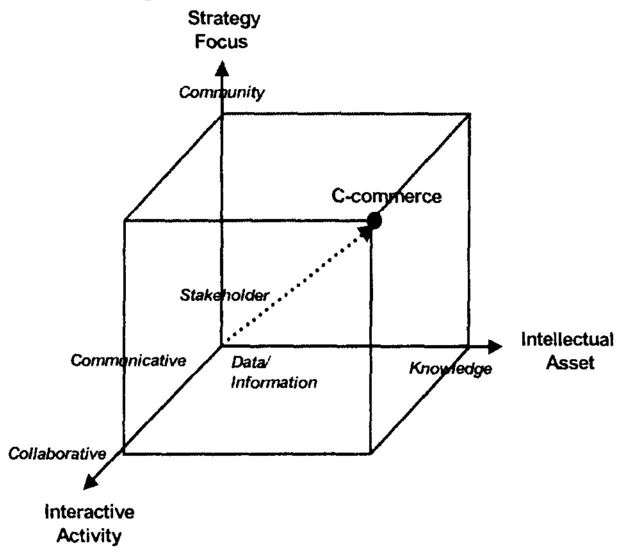

Figure 1. A perspective on c-commerce

C-commerce emphasizes collaborative interactions beyond the communications or transactions for e-commerce. Rather than producing and then selling, c-commerce focuses on collaboration among the participating stakeholders for c-commerce own value proposition enhancement [15], [32]. The collaboration mechanisms should be based on well-defined roles for each stakeholder; the roles should be determined by focusing the core competencies of stakeholders on maximizing the competitive advantage of the c-commerce community, along with conforming to the strategy focus in the best way. Another feature of c-commerce is that although stakeholders have their own localized strategies that may be employed in e-commerce they need a community-oriented strategy—a consensus is essential. In c-commerce, stakeholders should adhere to the rules of engagement on the basis of community strategies.

To produce the synergy of collaboration, stakeholders need to understand and share with each other more of their intellectual capital such as knowledge and expertise. Consequently, implementing a collaborative system to exchange and share knowledge as well as information on products and services is important, especially a system based on a variety of mechanisms that support the specific tasks of the stakeholders.

To construct such systems effectively, a systematic approach is required to support the major features of our c-commerce concept. Nevertheless, other methodologies that focus on inter-organizational collaboration, such as the allied concurrent engineering methodology (ACEM) [4], the specifications of coordinated and cooperative activity (SOCCA) [30], the inter-organizational workflow (IWOF) [31] and the inter-enterprise electronic commerce (IEEC) [27], lack analysis and design methods that fully cover the major features of c-commerce. These methodologies do not include the series of developmental phases that were derived from systematically and seamlessly transforming the stakeholders' roles into technical specifications.

For these reasons, we propose a role-driven component-oriented methodology (RCOM) for developing a c-commerce system that can successfully acclimate to the evolving e-business environment. In the role-driven approach, which is a useful way of identifying and analyzing the ever-changing c-commerce requirements [19], [34], we need to understand and specify the complex behavior of c-commerce with multiple stakeholders while addressing the dynamic interactions [1], [2], [5], [6], [8], [11]. As a result, the role-driven approach helps identify and control the roles of multiple stakeholders so that the stakeholders cooperate with each other effectively [2], [5].

The component-oriented approach also includes a variety of benefits: reusability, rapid development, cost effectiveness, and better and more dynamic service [20]. Accordingly, these advantages can provide a flexible environment that caters to dynamic change [12]. For consistent and seamless transition mechanisms in our methodology, we used a component approach in which the conceptual artifacts of analysis were adapted to the technical artifacts. The analysis results can be transformed effectively and consistently into reusable units in implementation via design works. Though the object-oriented approach brought about a major revolution from traditional software development, the promise of being able to reuse large-scale codes has not become a reality [14].

2. Architecture

In this section, we define the terminology employed in our methodology, and then we introduce the architecture of the methodology.

2.1 Terminology Definitions

The terminology used in our methodology has three levels, as depicted in Figure 2: conceptual, logical and physical. In the conceptual level, a role domain is identified by the core competencies of the stakeholders in the c-commerce community toward their strategy. Core competency has already been addressed as a critical factor in defining the roles of stakeholders [13], [24]. Role domains, which are valuable capabilities that are unique among stakeholders' characteristics, are strategically flexible in developing the potential and scope of c-commerce.

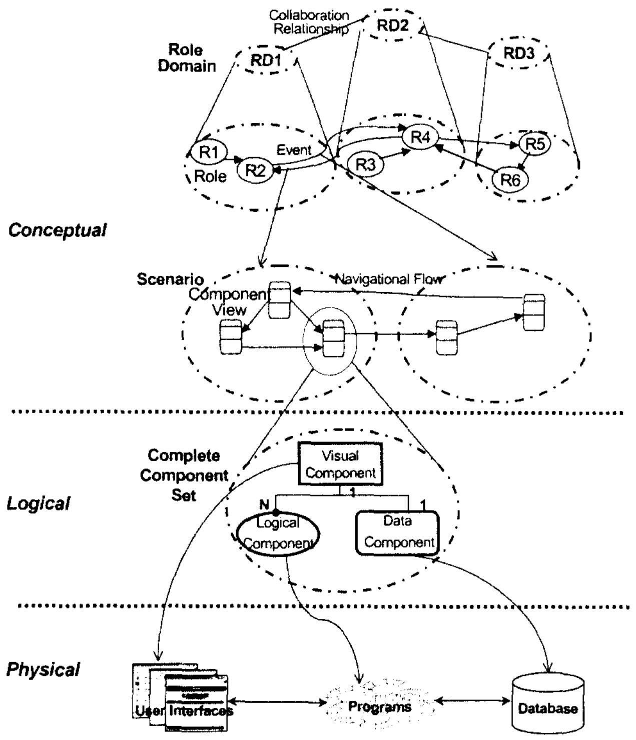

Figure 2. Concepts employed in RCOM

To specify a role domain, understanding the concept of a role is required. The concept of a role is useful in identifying, analyzing and constructing the ever-changing c-commerce behavior and responsibilities [19], [34]. Roles are more specified units for performing a role domain based on the strategy focus of c-commerce. A role domain may have more than one role. Determining the roles is a basic part of the analysis for effectively maintaining and helping the complex systems to evolve [18], [34].

Interactions of the roles can be induced as events, representing collaboration mechanisms for implementing c-commerce. In other research, the event concept has been addressed in terms of the role concept [1], [2], [5], [6], [8]. To specifically analyze events in our methodology, we adopted a scenario method in which a scenario corresponds to an event and is described by the use of natural language. Although a scenario is similar to a use case or scripts [17], [33], it is different from them in a usable context. Our scenario approach conceptually focused on business for the purpose of determining the collaborative relationships among stakeholders while the use case or script was mainly used technical purposes. Our scenarios can be used effectively in determining specific tasks for achieving the collaborative mechanisms of c-commerce in a natural fashion. In our methodology, a unit of such a specific task is referred to as a "component view".

The concept of a component view is employed at a conceptual level. The component view, which is a navigational primitive of a c-commerce system that depends on the events between the roles of participating stakeholders, is represented as a piece of information and the logic (or function) for handling the tasks of the events. Component views comprise three categories: cockpit domain, task domain and supplement domain. The component view of the cockpit domain plays a controlling work in the component views of the task and supplement domains; it serves as a starting point of the system or as a menu for accessing the other component views. Task component views are identified as a unit representing a business task. Supplementary component views provide additional information that supports task component views, which are determined by behavioral descriptions in the scenarios. Once the task view components are determined, other component views are then derived by brainstorming on the additional information required for performing task component views and on ways to organize and access the task component views.

At the logical level, the encapsulated data attribute information of the view component corresponds to the data component (DCO) and the encapsulated responsibility information corresponds to the logical component (LCO). Furthermore, at the logical level, the visual component (VCO) is also considered for user interface. The application of the component concept begins at the logical level. Consequently, at the logical level, the component view is specified as a set of components that includes the DCO, the LCO and the VCO.

The logical level provides the specific design results of these components for their implementation. From the structural perspective, the VCO may be separated into a screen type and a dialogue type. The screen type is part of the VCO of a system's complete display. The complete display of a component view may consist of more than one screen component by the use of frame tags. The dialogue type typically implements a dialogue window for confirmation of a user's action on a system's screen. Since the dialogue component is likely to be standardized as in the case of confirming a user's action, it is often shared by more than one screen.

The LCO is subdivided into program type and method type. A program type is an encapsulated component that will be implemented according to the component view responsibilities, while the method type will be implemented for simple program logic such as creating, retrieving, updating, deleting and calling other component views. The DCO is a component that refers to a data attribute set or relational table. In summary, the LCO controls the DCOs by working the functions of the VCOs. Finally, a complete component set means a set of components that are mapped to a component view. The definitions of key terminology used in RCOM are summarized in Table 1.

Table 1. Key definitions employed in RCOM

Terminology			Definitions
Role Domain			A core competence unit of stakeholder that will be participated in c-commerce.
Role			A specified unit for a role domain.
Scenario			A natural language form of an event description.
Component view			A navigational primitive of c-commerce requirements. This is determined at a conceptual level.
Compo-nent View Domains	Cockpit		Guide users to other units in the task and supplement domain.
	Task		Perform business tasks with data, information and knowledge.
	Supplement		Provide additional information for task component view.
Compo-nent Types	Visual	Screen	A navigational primitive that will be implemented as a screen interface. The unit may become a whole or partial interface.
		Dialogue	A primitive of multimedia data that may be accessed from a screen unit.
	Logical	Program	An encapsulated component that will be implemented according to the component view responsibilities. The component is invoked from a screen or dialogue unit. The component may become a whole or partial view responsibility.
		Method	An encapsulated component that will be implemented for simple logistics such as creating, retrieving, updating, deleting and caller functions from a component view. The component is also invoked from a screen or dialogue unit.
	Data		A component referring to relational table components or data attribute set.
Complete Component Set			A set of components mapped to a component view; it consists of three types of components.

As already described, the component-oriented approach is a major part of our methodology. Recently, the reuse of components has been growing explosively in many organizations because of the large number of attempts to develop systems by integrating existing components [12], [14], [22]. This approach can potentially be used to reduce the cost of software development, to enable rapid reassembly of the systems and to reduce the spiraling maintenance burden associated with the evolving c-commerce system. Accordingly, it can enhance the flexibility of development and the maintainability of results.

The benefits of the component-oriented approach have often been discussed from a technical perspective with a focus on software, rather than from a more abstract perspective. Handling the software component is known as component-based systems engineering, a field which is heavily dependent on integration technologies such as

Common Object Request Broker Architecture (CORBA) [21], the Component Object Model (COM) [26] and Enterprise Java Beans (EJB) [23].

However, to benefit more effectively from the advantages of the component-oriented approach, we need to use it in a generic way independently of any specific technical platforms or component integration technologies; at the same time, the implementation of this approach depends on incorporating a preferred technology architecture [14], [25], [29]. This perspective is our approach. Our component concept aims to respond to the dynamically and opportunistically evolving c-commerce system through quick modifications, as well as plug-and-play compositions and reusability.

2.2 Methodology Architecture

Our methodology, RCOM, was devised for developing c-commerce systems: it transforms conceptual roles and events for collaborative interactions in a community of stakeholders into logical-level specifications of adaptable components. The role-driven approach used in the methodology emphasizes the need to establish a collaborative structure in a c-commerce community. Furthermore, this approach emphasizes the need to determine logical-level components; this will enhance its methodological capabilities for effective development and economical maintenance of the c-commerce requirements or dynamism of the environment. Thus, as shown in Figure 3, RCOM consists of four phases: collaboration analysis, component analysis, component design and the implementation phase. Although collaboration analysis, component analysis and component design are independent of any specific technical platforms and details, the implementation phase depends on such platforms and details. Each phase was conducted iteratively, though feedback is not presented for the simplicity of the presentation.

The purpose of the first phase of the RCOM, collaboration analysis, is to establish the structure of collaborative relationships among the stakeholders in a c-commerce community by identifying the generic roles to be allocated to the stakeholders. This phase comprises two sub-phases: collaboration domain analysis and collaboration specification analysis. The aim of collaboration domain analysis is to grasp the abstracted generic roles, called role domains, and to define their relationships according to the core competence required for maximizing the collaborations in a c-commerce community. This sub-phase results in a collaboration context diagram (CCD), which shows the collaborative structure among the required role domains, and a collaboration context table (CCT), which provides detailed descriptions for the role domains in the CCD. On the basis of the CCD and the CCT, we then modeled further specifications such as roles and their relationships, which are called events. From this work we produced a collaboration specification diagram (CSD) and a collaboration specifications table (CST). The CST provides descriptions of the roles.

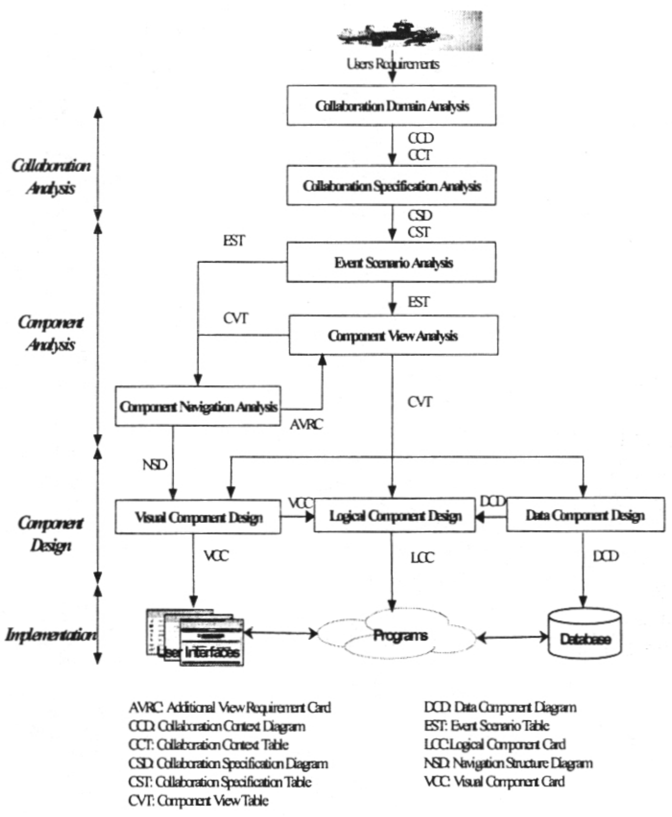
Users Requirements
Collaboration Domain Analysis
CCD
CCT
Collaboration Specification Analysis
CSD
CST
EST
Event Scenario Analysis
EST
CVT
Component View Analysis
Component Navigation Analysis
AVRC
CVT
NSD
VCC
DCD
Visual Component Design
Logical Component Design
Data Component Design
VCC
LCC
DCD
User Interfaces
Programs
Database
Collaboration Analysis
Component Analysis
Component Design
Implementation
AVRC: Additional View Requirement Card
CCD: Collaboration Context Diagram
CCT: Collaboration Context Table
CSD: Collaboration Specification Diagram
CST: Collaboration Specification Table
CVT: Component View Table
DCD: Data Component Diagram
EST: Event Scenario Table
LCC: Logical Component Card
NSD: Navigation Structure Diagram
VCC: Visual Component Card

Figure 3. Methodology architecture

The next phase, component analysis, comprises three sub-phases: event scenario analysis, component view analysis and component navigation analysis. The first sub-phase, event scenario analysis, generates scenarios for events determined from the CSD. As a result, event scenario tables (ESTs) are produced, which describe the event scenarios in the form of natural language. Component view analysis then determines the task-oriented component views and models them into a component view table (CVT) by referring to the descriptions in the event scenario table (EST). The CVT shows the data attributes and responsibilities for each component view, and presents the views' component indexes. The component views are handled at the conceptual level.

In component navigation analysis, we analyze component views in terms of their navigations in a target system. Firstly, the views included in the supplement and cockpit domains are defined, whereas the views in the task domain were already determined in component view analysis. The result of defining the views produces an additional view requirement card. Secondly, the navigational structure of all the views, including the views in the task domains, and the navigational paths of the views are defined in a navigational structure diagram. The navigational structure diagram is based on the EST and the CVT that shows the view structures.

The component design phase deals with implementation issues in terms of database, interface and program by using component artifacts at the logical level

[10], [28]. Firstly, the data component design sub-phase designs DCOs by considering technical attributes required for a target system's operations. This sub-phase is performed on the basis of the CVT, which includes conceptual data attributes of the views. As a result, the data component diagram (DCD) is produced.

Secondly, the visual component design sub-phase designs VCOs for implementing the interfaces of the system. This sub-phase is carried out on the basis of the CVT, which shows the data attribute set of the views and the responsibilities for each component view. As a result of this work, visual component cards (VCCs) are generated.

Finally, the logical design sub-phase designs LCOs to provide program logic for implementation. This sub-phase is performed by considering the responsibilities of the views in the CVT, as well as specification of the DCD and VCOs. This work produces logical component cards (LCCs), which include program logic in the form of pseudo codes. The implementation phase transforms the DCD into a database, the VCCs into user interfaces and LCCs into programs to construct a running c-commerce system.

3. Construction

Each phase of the RCOM is described in more specifically by the use of the real-life case for an escalator industry company in South Korea. For the purpose of confidentiality, the company is referred as "H Company". Currently, H Company produces elevator and escalator systems for commercial sector. The case of H Company can help understand the proposed methodology effectively, although three phases such as collaboration analysis, component analysis, and component design are not represented here due to the limitation of paper length. The implementation phase transforms the DCD, VCDs, and LCCs into a database, user interfaces, and programs, respectively. In H company project, the c-commerce system is implemented in the Web-based environment. As implementation results in H company, 150 VCDs, 195 LCCs, 1 DCD were implemented. To support interoperability in a heterogeneous environment, the system was based on the concept of DCOM [7]. Each component in the system was defined as a distributed component and packaged as an independent piece of code that could be accessed by remote clients via method invocations. The broker is the component that establishes the client–server relationships between the components. Using the broker, a . component can transparently invoke a method on a server that may be on the same machine or on the Internet.

Figure 4, which shows the processes for the selection of optimal parts, depicts the screen for the component named "Customer's Part Design Choice" where design candidates are generated by typing in the escalator type and the design part selection. First, a design engineer for the customer logs on to the H Company c-commerce system; then the c-commerce system requires information on the type of escalator and the part to be designed before being connected to a design stakeholder at O Company. Then, using the case base management system for O Company, the engineer automatically generates a possible design along with complete technical specifications in a real-time. Even more help can be given with on-line visual communication, if required. The engineer then refines the designs until one appears to satisfy the requirements.

The refining designs process can be improved by a concurrent collaboration between the customer's engineer and the designer from the O Company by on-line communication and a shared white board in a screen called "Searching Part by Customer", as shown in Figure 5.In the next step, the engineer can run real-time simulations of the design using a parametric programming software application provided on the site by the c-commerce system.

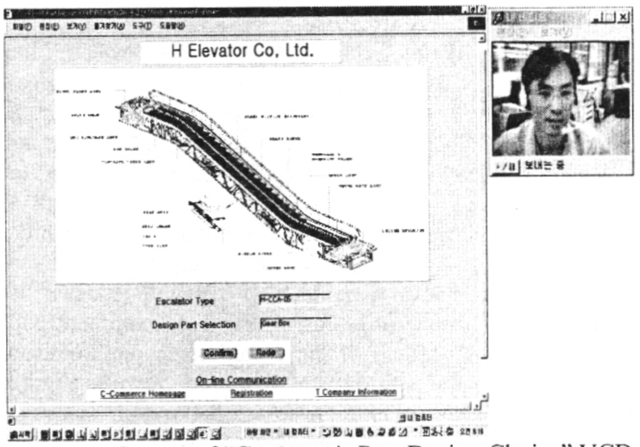

Figure 4. Screen of "Customer's Part Design Choice" VCD

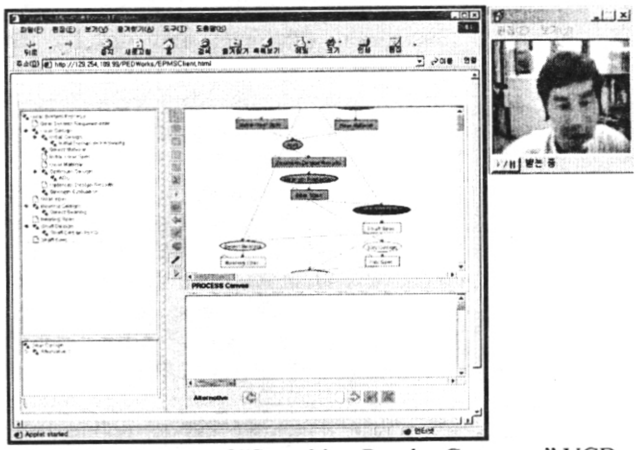

Figure 5. Screen of "Searching Part by Customer" VCD

4. Conclusion

Our methodology, RCOM, adopts two modeling perspectives: role and component. The role-driven perspective focuses on capturing business requirements for a c-commerce community by defining the roles of stakeholders and their collaborative relationships. The roles are determined in the light of core competences and then, on the basis of the roles, the business requirements for the collaborative interactions can be analyzed in a systematic fashion. On the other hand, the component-oriented perspective focuses on the consistent and systematic development of the component artifacts required in implementation; this consistency can reduce the development cycle time and thus enhance marketability.

References

[1] Albano, A., Bergamini, R., Ghelli, G., & Orsini, R. (1993). An object data model with roles. in: Agrawal, R., Baker, S., Bell, D. (Eds.) (1993). *Proceedings of the 19th International Conference on Very Large Databases*, Morgan Kaufmann, Dublin, 39-51.

[2] Bock, C. & Odell, J.J. (1998). A more complete model of relations and their implementation: Roles. *Journal of Object-Oriented Programming, 11(2),* 51-54.

[3] Burdick, D. (1999). What people are saying about Windchill. *Parametric Technology Corporation.* [Online] Available: http://www.ptc.com/products/quotes.html.

[4] Chen, Y.M. & Liang, M.W. (2000). Design and implementation of a collaborative engineering information system for allied concurrent engineering. *International Journal of Comput. Integrated Manufacturing, 16,* 9-27.

[5] Chu, W.W. & Zhang, G. (1997). Associations and roles in object-oriented modeling, in: Embley, D.W., Goldstein, R.C. (Eds.), *Proceedings of the 16th International Conference on Conceptual Modeling: ER'97*, 257-270.

[6] D'Souza, D.F. & Wills, A.C. (1998). *Objects, Components and Frameworks with UML*, Addison-Wesley, Reading, MA.

[7] Eddon, G. & Eddon, H. (1998). *Inside Distributed COM*, Microsoft Press.

[8] Elmasri, R., Weeldreyer, J., & Hevner, A. (1985). The category concept: an extension to the entity relationship model, *Data & Knowledge Engineering,* 1(1), 75-116.

[9] GartnerGroup. (1999). Gartner Group identifies c-commerce supply chain movement: an emerging trend in collaborative web communities. *Gartner McCarthy J.* [Online] Available: http://www.idg.net/idgns/1999/08/16/GartnerForetellsOfCollaborativeCommerce.shtml, 1999.

[10] Ginige, A. & Murugesan, S. (2001). Web engineering: an introduction. *IEEE Multimedia*, 8(1), 14-18.

[11] Gottlob, G., Schre, M., & Ock, B.R. (1996). Extending object-oriented systems with roles, *ACM Transactions on Information Systems, 14(3),* 268-296.

[12] Haines, C.G., Carney, D., & Foreman, J. (1997), Component-Based Software Development/COTS Integration, Software Technology Review, [Online]. Available: http://www.sei.cmu.edu/str/descriptions/CBD_body.html.

[13] Hamel, G. (1994). *The concept of core competence, competence-based competition.* New York, Wiley, 11-33.

[14] Henderson, P. & Walters, R. (2001). Behavioral analysis of component-based systems, *Information and Software Technology, 43*, 161-169.

[15] Holsapple CW & Singh M. (2000). Electronic commerce: definitional taxonomy, integration, and knowledge management. *Journal of Organizational Computing and Electronic Commerce* 10 (3).

[16] InSight. (1999). *Current Investments: e-commerce.* InSight Inc. [Online] Available: http://www.insightpartners.com/ecommerce.html.

[17] Jacobson, I. (1995). *Object-Oriented Software Engineering: A Use Case Driven Approach*, Addison-Wesley.

[18] Kendall, E.A. (1998). Agent roles and role models: new abstractions for multiagent system analysis and design, *International Workshop on Intelligent Agents in Information and Process Management-* September, Germany.

[19] Kendall, E.A. (1999). Role modelling for agent system analysis, design, and implementation, *International Conference on Agent Systems and Applications/ Mobile Agents- October (ASA/MA'99)*, Palm Springs.

[20] Kunda, D., Brooks, L. (2000). Assessing organisational obstacles to component-based development: a case study approach, Information and Software Technology 42, 715-725.

[21] Object Management Group, (2002). OMG CORBA, [Online] Available: http://www.omg.org.

[22] O'Brien, A., (1998). An intelligent component model for building design, *Second European Conference on Product and Process Modelling in the Building Industry, 19th-21st, Oct.*

[23] O'Neil, J. & Schildt, H. (1998). Java Beans Programming from the Ground Up, McGraw-Hill, New York.

[24] Prahalad, C. K. & Hamel, G. (1990). The core competence of the corporation, *Harv. Bus. Rev.*- May-June, 79-91.

[25] Rosenman, M.A. & Wang, F.J. (1999). CADOM: a Component Agent Model based Design-Oriented Model for Collaborative Design, *Research in Engineering Design II*, 193-205.

[26] Sessions, R. (1998). *COM and DCOM: Microsoft's Vision for Distributed Objects*, Wiley, New York.

[27] Shin, K. & Lim, C.S. (2002). A reference system for Internet based inter-enterprise electronic commerce, *The Journal of Systems and Software, 60(3),* 195-204.

[28] Smith, G., Gough, J., & Szyperski, C. (1998). Conciliation: the adaptation of independently developed components. *Second International Conference on Parallel and Distributed Computing and Networks (PDCN '98),* 31-38.

[29] Szyperski, C. (1998). *Component Software? Beyond Object-Oriented Programming*, Addison-Wesley, Reading, MA.

[30] Toussaint, P.J. (1998). *Integration of information systems: a study in requirements engineering*, Ph.D. Dissertation, Leiden University, The Netherlands.

[31] van der Aalst, W.M.P. (1999). Process-oriented architectures for electronic commerce and inter organizational workflow, *Information Systems, 24(9),* 639-671.

[32] Afuah, A & Tucci, C.L. (2000). *Internet Business Models and Strategies: Text and Cases*. McGraw-Hill.

[33] Rational Software. (2001). UML documentation: behavioral elements package: collaboration overview. [Online]Available: http://www.rational.com/uml/resources/docmentation/semantics/semanta.html.

[34] Zhao, L. (1999). Modeling roles with cascade, *IEEE Software, September/October,* 86-94.