eMarketplace Model: An Architecture for Collaborative Supply Chain Management and Integration

Hamada Ghenniwa[1], Jiangbo Dang[2], Michael Huhns[2], Weiming Shen[3]

[1] Dept. of Electrical and Computer Engineering
University of Western Ontario, London, Ontario Canada
hghenniwa@eng.uwo.ca
[2] Dept. of Computer Science and Engineering
University of South Carolina, Columbia, SC, USA
dangj@sc.edu, huhns@sc.edu
[3] Integrated Manufacturing Technologies Institute
National Research Council, London, Ontario, Canada
weiming.shen@nrc.gc.ca

Abstract. The current economic climate forces businesses to collaborate more frequently and build efficient organizations and supply chains that reduce time-to-market and costs. This chapter argues that an electronic marketplace (*eMarketplace*) is a promising architectural model to develop collaborative supply chain management and integration platform. It supports coordination mechanisms and integration at the business and systems levels of the enterprise and the supply chain. In this architecture, the *eMarketplace* exists as a collection of economically motivated software agents of service-oriented cooperative distributed systems. It enables and supports common integration and economic services between market participants. This chapter presents an agent-oriented dynamic trading mechanism that produces an integrated supply chain for the *eMarketplace*. Future *eMarketplaces* need to support both market-based and relationship-based supply chains. To this end, this chapter discusses coordination approaches based on market mechanisms such as auctions and multi-issue negotiation. The objective is to enable business entities to obtain efficient resource allocation while preserving long-term relationships.

1 Introduction

Information technology has enabled, and in some cases has forced, companies and organizations to redefine their business models and to reorient their internal capabilities to exploit electronic business (*eBusiness*) techniques. They are finding it necessary to collaborate to build more efficient operations and supply chains that reduce times-to-market and costs. *eBusiness* is the use of the Internet along with other electronic means and technologies to conduct such collaborations within businesses, from businesses to consumers, among businesses, and from businesses to government.

Traditional models of *eBusiness* integration, such as those based on EDI (Electronic Data Interchange) and enterprise-centric views, are useful for businesses with

well-defined trading relationships, but not enough for the rapidly growing and changing global marketplace. In these models, point-to-point interfaces are created to support transactions involving replenishment orders for the goods of a previously negotiated contract. In the sell-side model, either a single distributor is responsible for aggregating all the suppliers, or the customer is responsible for comparison-shopping between suppliers. This makes it inefficient and expensive for both customers and suppliers. In the buy-side model, the buying organizations are responsible for setting-up and maintaining catalogs of their suppliers, and hence it is costly and technically demanding.

An electronic marketplace (*eMarketplace*), however, appears to be a promising integration model for *eBusiness*. *eMarketplaces* provide an integrated and efficient environment for consumers, who depend on a variety of products and services that can spread across several suppliers or marketplaces. Likewise, they provide suppliers with the ability to reach, discover, and develop new customers within a single *eMarketplace* or across various *eMarketplaces* quickly with low cost. In general, *eMarketplaces* offer businesses the chance to develop and enhance their most important relationships—those with customers and suppliers. It enables the creation and leveraging of services and supply operations in a way that seamlessly integrates business entities (customers, suppliers, partners, and competitors) in a dynamic trading community. In this work, we view an *eMarketplace* as a cooperative distributed system that integrates participating business entities, including consumers, suppliers, and other intermediaries. This architecture enables and facilitates common economic services and commerce transactions between the consumers and suppliers, such as brokering, pricing, and negotiation, as well as cross-enterprise integration and cooperation in an electronic supply chain. In this architecture, the *eMarketplace* exists as a collection of economically motivated software agents.

The rest of the chapter is organized as follows. First, it reviews some of the business models related to *eBusiness* applications with a brief analysis of the main architectural design issues for *eMarketplaces*. After that, it briefly describes an architecture for a cooperative distributed system, Business-Centric Knowledge-Oriented architecture (BCKOA), for *eMarketplace* integration. This is followed by a description of a layered BCKOA implementation for an *eMarketplace*. Then the main components of an agent-oriented BCKOA for an *eMarketplace* are presented, including a supply chain automation system for integration and management using a group of cooperating software agents. Unlike traditional buyer-centric approaches, the proposed architecture emphasizes on the supplier perspective as well through the enablement of relationship-based supply chain management. A short description of an ongoing implementation of the proposed model for virtual enterprise *eMarketplace* is described next. It next describes a multi-issue negotiation model and protocol that considers both functional and quality attributes. Then it discusses some of the related work in both the academic and industrial communities. Finally, it summarizes the main contributions of this chapter.

2 *eBusiness* Models

One of the most frequently mentioned barriers to successful *eBusiness* applications is the lack of an appropriate business model. In simple words, it is "an architecture for the product, services, and information flows, including a description of the various business actors and their roles; and a description of the potential benefits for the various business actors; and a description of the sources of revenues" [47]. Possible architectures for business models can be constructed by combining interaction patterns of its components with value-chain integration [4][15]. These can be fully open, with an arbitrary number of business participants (customers and suppliers), or semi-open, with one customer and multiple suppliers or vice-versa. In principle, several architectures can be conceived for *eBusiness* applications; in practice, however, only a limited number can be realized [47]. The following are the most widely realized models [47].

A basic model is *eShop*. It is based on providing a self-service storefront with the company's catalogs and product offerings on the Web. The business objective is to lower the sales cost. A major concern with this model is making customers responsible for surfing a large number of *eShop* sites for comparisons among the products from different suppliers. An *eProcurement* model, however, focuses on the buying aspect of the business. A typical architecture for *eProcurement* consists of a browser-based self-service interface to the corporate purchasing system or its ERP. The supplier catalogs are presented to end-users through a single unified catalog, thereby facilitating a corporate-wide standard procurement process. In addition, *eProcurement* might support calls for tender through the Web, which might be accompanied by an electronic submission of bids. Nonetheless, an *eProcurement* model does not support dynamic trading. The business objective of this model is cost savings on purchasing operations. Online auction models have also received much attention for automating dynamic trading. The primary business objective is to increase efficiency, reduce waste, and minimize overall cost. Other models are based on creating value-chain businesses. One model describes service provisioning of specific functions, such as electronic payments or logistics. Other approaches are also emerging in production and stock management, where new intermediary service providers are formed to provide specialized expertise to analyze and fine-tune production. The business objective of this model is to generate revenue based on fee or revenue percentage.

Although each of the above models attempts to provide an *eBusiness* solution, none addresses the creation and leveraging of services and supply operations in a way that seamlessly integrates business entities (customers, suppliers, partners, and competitors) in a dynamic trading community. A very promising business model that can effectively deal with this challenge is *eMarketplace*. This model supports value-chain integration and provisioning in its structure and services. It combines the advantages of the sell-side, the buy-side, and the value-chain models. The business objective of the *eMarketplace* model can be based on a combination of subscription fees, transaction fees, and service fees.

The next section lays the engineering foundation for developing an architectural framework for *eBusiness,* with special attention on an *eMarketplace* model. The specification of an *eMarketplace* as a cooperative distributed system describes the

architecture of an ontology driven *eBusiness* environment that deals with technological and business issues.

3 *eMarketplaces*: Requirements Analysis and Design Issues

Early attention to *eMarketplaces* focused on lowering the business operation costs. Garciano and Kaplan [18] suggested that the transaction cost savings alone from *eBusiness* exchanges could be a significant portion of the total cost of production and order fulfillment. However, as *eBusiness* grows and becomes viable in the real world, its corresponding *eMarketplaces* must expand to support a broader base of services ranging from baseline interaction and directory services to specialty market services, such as dynamic trading, supply chain integration and management. By automating and lowering the cost of searching and matchmaking between consumers and suppliers, *eMarketplace* becomes an appropriate solution for businesses to conduct large volumes of transactions using dynamic trading approaches such as auctions. Also, through the facilitation of collaboration and information-sharing services, *eMarketplaces* enable and support sharing of supply chain information such as forecasts and inventory levels. *eMarketplaces* can also improve the efficiency of the supply chain by automating business processes such as procurement, order management, and fulfillment. In addition, an *eMarketplace* should enable and strengthening the relationship between business participants and their supporting systems.

To this end, a fundamental aspect that our proposed *eMarketplace* architecture supports is to maintain various relationships between customers and suppliers. This enables both customers and suppliers to leverage economies of scale in their trading relationships and provide them with an access to various liquid marketplaces. This in turn allows the use of dynamic pricing models[1], such as exchanges and auctions. The following subsections provide a detailed analysis of the aspects that sets the foundation for the proposed architecture.

3.1 Market Structure and Economy Model

The market structure governs the trading process and defines the formal rules for market access, traders' interactions, price determination, and trade generations. Its behavior restricts the set of message sequences that traders may exchange and determines the trading outcome. Therefore, a market institution [32] is the specification of the set of admissible messages (i.e., traders' actions, usually price and/or quantity offers), and the final goods/services allocation given any combination of messages chosen by the participants and any initial allocation. In classical economic theory, there are several market models for specific trading situations and structural behaviors. In the commodity market model, various suppliers and consumers participate to

[1] We use the term dynamic pricing broadly to refer to short-term flexibility of prices to respond to changing supply and demand conditions.

trade goods/services (commodity) of the same type. The market price is publicly agreed upon for each commodity independent of a particular supplier. The challenge in this market structure is to deploy a pricing methodology that produces price adjustments that bring about market equilibrium (i.e., equalize supply and demand).

In an auction-based market, each participant (consumer and supplier) acts independently and contracts to buy or sell at a price agreed upon privately. An auction-based *eMarketplace* is a form of centralized facility, or clearinghouse, by which costumers and suppliers execute trades in an open and competitive bidding process. In open auctions, bidders can know the bid value of the others and will iteratively have an opportunity to offer competitive bids. However, in open distributed environments where an auction can be distributed over space and/or time, an iterative mechanism might not be feasible. A standard form of one-shot auctions is the first-price sealed-bid auction. It avoids iterations but introduces another computational problem of counter-speculation of the other agents' valuations and might not achieve the highest price. The Vickrey auction eliminates the computational cost of both the iterative valuations and counter-speculations overhead.

The two market structures above are not appropriate for bargaining situations where few participants try to reach an agreement that will leave them at least as well off as they could be if they reached no agreement. Most of these situations cannot be entirely determined by the market forces. In bargaining, both customers and suppliers have their own objective functions and they negotiate with each other as long as their objectives are met. The participants can engage in direct negotiations with each other using their respective bargaining strategies to arrive at a "fair" price for a particular item. This market structure does not support a specific negotiation protocol; rather the participants will use an unrestricted bidding protocol. A major challenge in this structure is how to enable any participant to determine the "fair" price.

3.2 Supply Chain Management and Integration

To provide smooth and effective integration at the business level, the *eMarketplace* architecture accommodates and supports interfaces to the existing business models of the participant entities through cooperative supply chain integration and management. An *eMarketplace* can be treated as a physically and logically distributed system of interacting autonomous business entities. Yet, there is a need for well-accepted interoperability standards, which must be meshed for supply chain integration to meet business demands. Conceptually, a supply chain manages coordinated information and material flows, production operations, and logistics of the *eMarketplace*. It provides the *eMarketplace* with flexibility and agility in responding to customer demand shifts without conflicts in resource utilization. The fundamental objective is to improve coordination within and between various participant business entities in the supply chain. Effective coordination can lead to a reduction in lead times and costs, alignment of interdependent decision-making processes, improvement in the overall performance of each participant in the chain, as well as the supply chain itself. In an *eMarketplace* setting, supply chain management can be viewed as a cooperative distributed problem-solving activity among a society or group formed by autonomous

business entities that work together to solve a common problem [45]. The decision-making process is centralized for the group, but decentralized for the local decisions of each member. Therefore, the problem of supply chain design in an *eMarketplace*, as discussed later, can be solved by the design of a structure and mechanism for coordination and integration in a distributed system.

The choice of coordination mechanism depends on the setting of the interaction between consumers and suppliers. There are four possible settings between them (i.e., consumer-to-supplier): many-to-many, one-to-one, one-to-many, and many-to-one. Market transactions are appropriate for a many-to-many setting. Since adequate liquidity is critical to the success of an *eMarketplace*, only commodities, near commodities, or other highly standardized products are likely to attract adequate trading volumes to support many-to-many interactions. At the other side of the spectrum is the one-to-one setting of negotiation and partnerships, where prices often vary by customer in relation to different non-price attributes, such as purchasing volumes and service requirements. Coordination is based on one-to-one negotiations that are influenced by the long-term relationship between the consumer and the supplier. In one-to-many and many-to-one settings, participants can have more flexibility to select the most beneficial coordination mechanism. In this context *eMarketplaces* can provide flexible structure for mechanisms that improve supply chain coordination. When a single consumer is interacting with multiple suppliers, the consumer can use either a portfolio of long-term contracts, or market-based approaches such as reverse auctions. When a single supplier is interacting with multiple consumers, the supplier has a number of choices, including revenue management, "forward" auctions, dynamic pricing, and long-term contracting.

Using market-based mechanisms to match supply and demand prevents consumers with low valuations from receiving limited goods and prevents suppliers with high production costs from supplying limited demand. Nevertheless, there are a number of challenges that need to be addressed for *eMarketplaces* to be successful, including adequate market liquidity and establishing approaches for realizing the benefits of market mechanisms without undermining existing supply chain relationships [22]. In relationship-based supply chains, long-term relationships have higher value than that of the efficient resource allocation. In these settings, prices are usually negotiated rather than determined by the market, and efficient allocation is sacrificed in favor of the other benefits of relationship-based negotiations. Usually supply and demand are balanced by non-price mechanisms. In this context, allocation is treated as a function of negotiation and relationship rather than as allocation efficiency in the economic sense. For example, to deal with an oversupply, suppliers may negotiate special deals on forward buys or inventory buys to "borrow" demand from the future, or even build-up excess inventory. However, in the absence of market-clearing prices, supply and demand usually are not in equilibrium. For example, supply shocks or unanticipated demand increases can lead to shortages due to inability of the prices to change in a sufficient rate to dampen demand or stimulate production. The same for weak demand or excess production which can result in overstocked inventory while prices are not able to fall in a rate to equilibrate supply and demand. Furthermore, contract prices that lag true market-clearing prices can cause poor capacity investment decisions, leading to an undesirable cycle of oversupply and undersupply. *eMarketplaces*

can address these challenges by providing a platform that combines both relationship-based and market-based coordination mechanisms. Therefore, relationship-based supply chain participants can use the market-based mechanism as a spot market to buffer supply and demand shocks [9][10]. For example, suppliers can use it to offload excess inventory. Also, consumers can use it to deal with periodic shortages. In addition, through spot markets contract prices can be adjusted in response to shifts in supply and demand by providing benchmarks for contract negotiations.

Dynamic pricing also provides an important allocation mechanism for highly differentiated goods and services. Transactions can be complex and often require evaluation and negotiation along multiple attributes and different factors. They can affect the purchasing volumes between a given set of supply chain partners. Transactions may also involve bundles or combinations of possibly complementary goods and services. Auctions and multi-issue negotiations can provide supply chain participants with the adequate decision support tools to efficiently carry out complex multi-dimensional transactions.

3.3 Foundation Architecture for Integration

It is also important that the architecture of an *eMarketplace* supports and leverages the participants' legacy environments with minimum overhead. The support can take place, as will be described later, over technology-independent cooperative distributed system architecture. Another key factor for the foundation of an *eMarketplace* is the ability to operate in an open environment. This is driven by the fact that in many cases a customer's needs may go beyond the specialist capabilities of any single *eMarketplace*. The architecture of the *eMarketplace* provides the foundation to integrate and leverage the participants' resources, such as applications and databases. Traditionally, the foundation technology that enables enterprises to connect resources together is known as *middleware*. Mainstream middleware solutions focus on integration at the data-level, such as those based on OMG CORBA™ (Object Management Group, Inc. 1995) and J2EE™ (Java™ 2 Platform, Enterprise Edition). Enterprise application integration (EAI) has emerged as middleware technology with an objective to ease the burden and lower the costs of application integration. However, different EAI tools are developed to accommodate different levels of integration requirements, including Object-level, business process-level, and cross-enterprise process-level. However, there are currently very few EAI solutions specifically designed for cross-enterprise integration. While EAI tools focus on technology-centered integration, other complementary approaches focus on integration as an architectural aspect. One approach is a mediator-based architecture [52], which comprises a layer of "intelligent" middleware services to link data resources and applications. Another approach is the facilitator [18], in which integration is based on the principle that any system (software or hardware) can interoperate with any other system without the intervention of human users or their developers. This level of automation depends on

supporting ontologies to describe the resources. Facilitators use meta-level information in converting, translating, or routing data and information.

In the proposed *eMarketplace* environment there are significant interactions between the systems deployed by the participating business units, their customers, and other businesses. Therefore, designing *eMarketplaces* requires embodying greater levels of business knowledge within the *eMarketplace* transactions, activities, and service definitions. Additionally, it requires a greater degree of communication, coordination, and cooperation within and among the business entities and their systems in the *eMarketplace*. In other words, the *eMarketplace* architecture represents an integrated body of people, systems, information, processes, services, and products. Several attempts in business-process reengineering addressed structural integration only. The focus was on reorganizing enterprise units along critical business processes, such as supply chain and the product life cycle [22]. However, in this chapter, the focus will be on structural, behavioral, and informational integration of the participant business entities. The following sections address these aspects in more details in order to set the foundation for the proposed architecture.

4 Business-Centric Knowledge-Oriented Architecture

The *eMarketplace* architecture must be semantically rich and describe the organization and the interconnection among the software components, business services, and business ontologies of the *eMarketplace*. In this work, we deal with both the *fundamental* and the *practical* issues of integration.

Fundamentally, we view integration as an abstraction level at which a distributed system environment can be described as collective coherent universe of cooperating entities. Here we describe a business-centric knowledge-oriented architecture (BCKOA) for cooperative distributed systems. BCKOA specifications provide the abstraction to support the domain entities and applications independent of any specific technology. The main elements of BCKOA include domain services, integration services, and domain ontology. A key to BCKOA is a service-oriented model in which the overall connectivity of the system supports a “virtual” point-to-point integration mechanism. BCKOA recognizes the separations among functionalities supported by its services. Yet, they can be ubiquitously integrated in an *ad hoc* structure to fulfill a complex business service or a market structure.

To support heterogeneity and technology-independent properties at the system level, the boundaries between the layers correspond to standardized interfaces. Additionally, BCKOA includes domain ontology to capture and implement the conceptualization of an application domain at the knowledge level. BCKOA provides three families of integration services. (1) Ontology and semantic integration services support the semantic manipulations needed when integrating and transforming information or knowledge to satisfy a BCKOA task; also when capabilities require re-using components. (2) Coordination and cooperation services support ad hoc and automated BCKOA configurations. This includes locating and discovering domain and BCKOA services that are potentially relevant to a domain or BCKOA service. (3) Wrapping services make different applications, components, objects, or modules comply with

internal or external standards. Such standards may involve the interface to the software system or its behavior.

The specifications of BCKOA services are independent of any component framework, but their implementation can be based on the services provided by the target framework. Here, the concept of service is viewed as a computational model that enables a designer to capture and represent complex applications in open environments, such as *eMarketplaces*, as a software artifact independent of the target framework. To this end, we proposed a meta-model for services description based on DAML-S constructs [9] and BCKOA services. The meta-model is treated at two levels: (i) UML-based model for service capabilities and processes, and (ii) agent-oriented model for service interactions (cooperative or competitive) [30].

A key challenge in putting BCKOA into a practical context is the transformation or the mapping of its abstract description into the specification of the target component framework. To deal with this issue, BCKOA requires that business-object implementations be obligated to conform to the domain ontology. The business-object specification itself in the domain ontology becomes the reusable component that can be configured and assembled into multiple solutions (business-objects), independent of technology implementation. Therefore, the domain ontology in BCKOA governs the structural and the behavioral semantics of the business-objects in a way that is consistent across all implementations, and is accessible from any implementation. The BCKOA framework, shown in Fig. 1, provides an integrated execution environment for integrated business object implementations. Mapping a BCKOA description to an implementation framework is driven by three specifications: domain ontology description, maps, and a profile. Technology mapping specifications include a map to specify a transformation from the BCKOA domain ontology and services to the implementation components and service extensions for the target component framework. The mapping of each business concept representation to its implementation is managed by a *profile*, as a set of properties that defines the environment for a mapping. This mechanism enables an automated transformation from a relatively stable domain ontology and service description to different component technologies. This framework has been supported by an integration development environment (SOAStudio) [31]. It also includes development workspaces and mechanisms as well as basic application programming interfaces. It enables designers to utilize the proposed framework effectively and transparently to develop agent-based services and further build business systems like *eMarketplace*. They are based on the proposed meta-model, and have agent constructs in software entity and DAML-S description to build ontologies. The extracted WSDL interfaces are published through UDDI-compliant registrar, which is also an agent-based service.

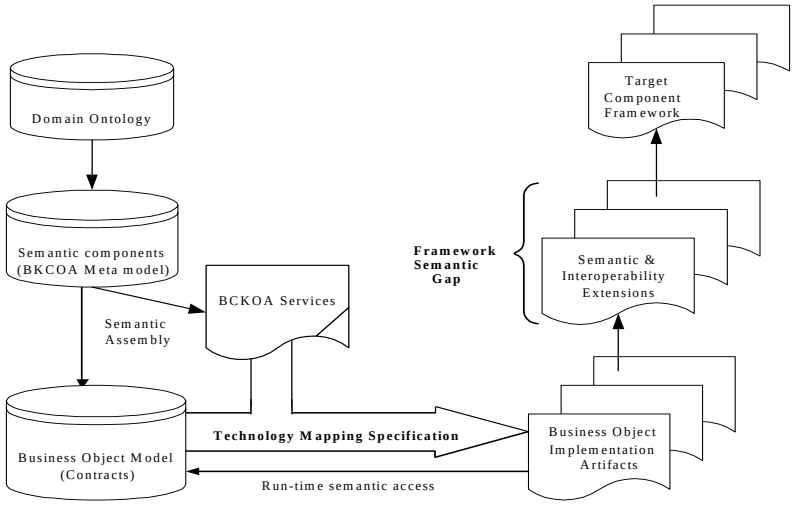


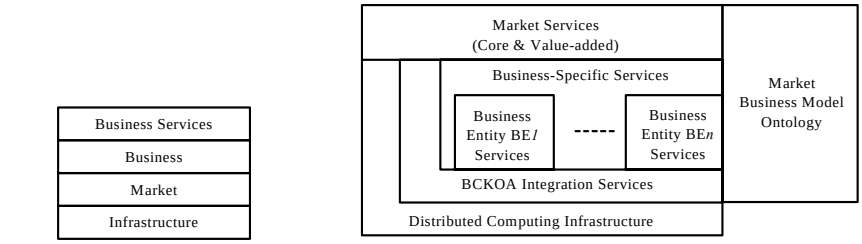
Fig. 1: BCKOA Framework

5 BCKOA-based *eMarketplace*

This section describes the application of BCKOA to develop an *eMarketplace* as a cooperative distributed system. The objective is to provide an automated framework that enable businesses (suppliers, customers, and intermediaries) to effectively engage in complex and diverse collaborative activities. The proposed BCKOA-based *eMarketplace* is shown in Fig. 2(b), which builds upon the abstraction architecture of the *eMarketplace* in Fig. 2(a) [20]. The lower layer of the *eMarketplace* architecture in Fig. 2(a) is the infrastructure that represents one or more physical network-based environments in which *eBusiness* systems can exist. The BCKOA model, in Fig. 2(b), supports the *eMarketplace* infrastructure using two layers: the distributed-computing layer and the integration-services layer. The assumption is that this infrastructure can support various markets for providing or obtaining specific goods and services. Yet, each *eMarketplace* may be independent and may support its own rules, procedures, and protocols as described by the market layer.

The market layer may support several business domains as described by the business layer. BCKOA, in Fig. 2(b), provides the integration between the business context of the market, and the services provided by the participant entities. A business-entity may participate in multiple *eMarketplace*s. A bank, for example, could participate with different roles in investment management market, mutual fund management market, and financial advisory market.

In BCKOA, the business-service layer is supported by three types of services, depicted in Fig. 2(b), (1) Business-specific services, (2) Business-entity services, which represent the implementation of the business services by specific business entities, and (3) Market services, which are categorized further into core, such as dynamic trading and supply chain services, and value-added, such as procurement process and workflow services. Here we focus on the core services. Ideally, the market services should be able to offer a wide variety of coordination and trade mechanisms to fit with multiple business models.



(a) Abstraction layers for *eMarketplace* **(b) BCKOA-based *eMarketplace***

Fig. 2: Use of BCKOA for the Architecture of an *eMarketplace*

Based on the success of applying economic theories in the real world as a sustainable model for exchanging and regulating resources, goods and services, we propose to apply a flexible computational economy framework for market services. Therefore, a BCKOA-based *eMarketplace* incorporates mechanisms for different types of market structures, such as auctions and bilateral negotiation. Each of which is viewed as a separate market session. In this chapter we will focus on multi-issue negotiation and auction based market sessions. Each participant (consumer or supplier) acts independently and contracts to buy or sell at a price agreed upon privately. Here we focus on coalition-based model for multi-issues negotiation [13] and private-value auctions, such as the Vickrey mechanism [51]. As it will be discussed later that coalition-based model for multi-issue negotiation is more effective and computational efficient than traditional issue-by-issue and package deal approaches. Also, Vickrey auction provides a market mechanism that is simpler, but more efficient and more stable than open auction mechanisms and classical sealed bid auctions [49]. While it is a simple yet powerful mechanism, it is important to mention that the Vickrey mechanism may not be appropriate in all domains. For example, truthful bidding is not necessarily the dominant strategy for domains where an agent's marginal costs (and thus its reservation price) are determined by other agents' valuations, such as the case with *public-value auctions* in the stock market [43].

In BCKOA-*eMarketplace*, supply chain management is treated as a coordination methodology that manages information and material flows, production operations, and logistics. The objective is to provide an automated coordination mechanism for the participants in a supply chain. The proposed solution combines both the multi-issue negotiation as coordination mechanism for relationship-based supply chain and Vickrey based auctions as coordination market-based supply chain. The adopted

integration framework for the supply chain makes use of the methodologies reported in [41][46]. In this work, we particularly extended Singh's application to supply chain integration [24]. The methodology, as described later, promotes the interchange of standard business documents and compensate for exceptions that might occur during execution. This methodology requires that the participant business entities in a cooperative supply chain only describe their supply processes using Open Applications Group (OAG) standard business documents and UML interaction diagrams. These are converted automatically into roles and specifications of the software agents for the corresponding business entities.

A combination of the market services and the business-entity services can be used to generate different business models of an *eMarketplace* as desired by the participating business entities. This structure enables a business-entity to integrate and describe the types of business services offered and the information needed to use a particular service offering within the *eMarketplace*. The details of each service type and the required information might vary among business entities, although the description of the service type is based on some common conventions described for an *eMarketplace* using service meta-model [30].

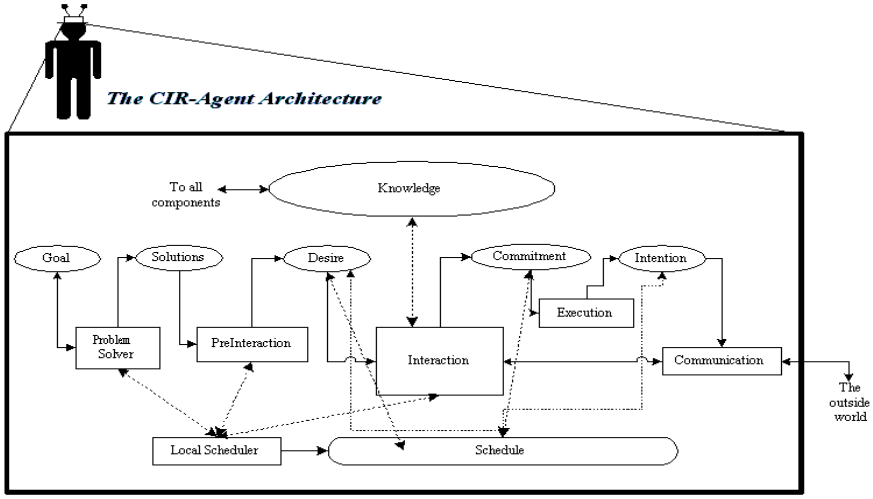
BCKOA recognizes the integration services as separate functionalities. Yet, they can be ubiquitously integrated in an *ad hoc* structure to fulfill a complex business service or a market structure. The interaction mechanisms supported by the integration layer describe both the pattern and protocol of exchanging messages between the services, such as brokering, resource discovery and ontology mapping

6 Agent-Oriented *eMarketplace* Model

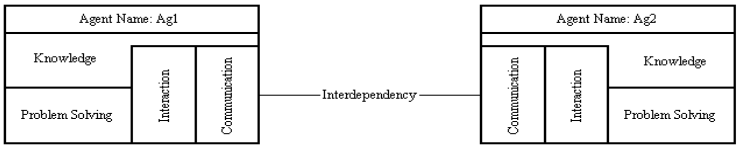
All services (business, market, and integration) in a BCKOA-based *eMarketplace* usually involve complex and nondeterministic interactions, often producing results that are ambiguous and incomplete. Auctions and *ad hoc* service integrations are some examples. In addition, the dynamic nature of the environment requires that the components of the system be able to change their configuration to participate in different, often simultaneous roles in *eMarketplaces*. These requirements could not be accomplished using traditional ways of manually configuring software. Agent-orientation is a very promising design paradigm for integration. In fact, such a paradigm is essential to model an open environment, such as an *eMarketplace*, especially considering the multiple dynamic and simultaneous roles a single business-entity may need to participate in given *eMarketplace* sessions (a financial services organization may have representatives acting on its behalf simultaneously within the context of brokering, service provisioning, and marketing).

Software agent technology provides the next generation in the evolution of computational modeling, programming methodologies, and software engineering paradigms. Here, we view "agent" as a metaphorical conceptualization tool at a high level of abstraction (knowledge level) that captures, supports, and implements features that are useful for distributed computation in open environments. The first feature of an agent is that it should be able to operate as a part of a community of cooperative distributed systems, including human users. In our view, an agent can be described as a

collection of primitive components that provide a focused and cohesive set of capabilities. Fig. 3 depicts the Coordinated Intelligent and Rational, Agent (CIR-Agent) model [21]. The basic components include a problem solver, interactions, and communication, as shown in Fig. 3(b). A particular arrangement or interconnection of the agent's components is required to constitute an agent, as shown in Fig. 3(a). This arrangement reflects the pattern of an agent's mental state as related to its reasoning to achieve a goal.



(a) Detailed Architecture of CIR-Agent



(b) Logical Architecture of CIR-Agent

Fig. 3: The CIR-Agent Architecture

However, no specific assumption is made on the detailed design of the agent's components. Therefore, the internal structure of the components can be designed and implemented using object-oriented or any other technology, provided that a developer conceptualizes the specified architecture of the agent as described in Fig. 3(b). A CIR-Agent model provides software engineers with features at a higher level of abstraction that are useful for cooperative environments. It supports flexibility at different levels of the design: system architecture, agent architecture, and agent component architecture. These degrees of flexibility allow information systems to adapt to changes with minimum requirements for redesign. An agent within the context of a BCKOA-based *eMarketplace* might play several roles and should be able to coordinate, cooperatively or competitively, with the other agents, including humans. There-

fore, as shown in Fig. 4, an agent's role can be categorized as user-interface, business-specific service, business-entity service, market service, or integration service.

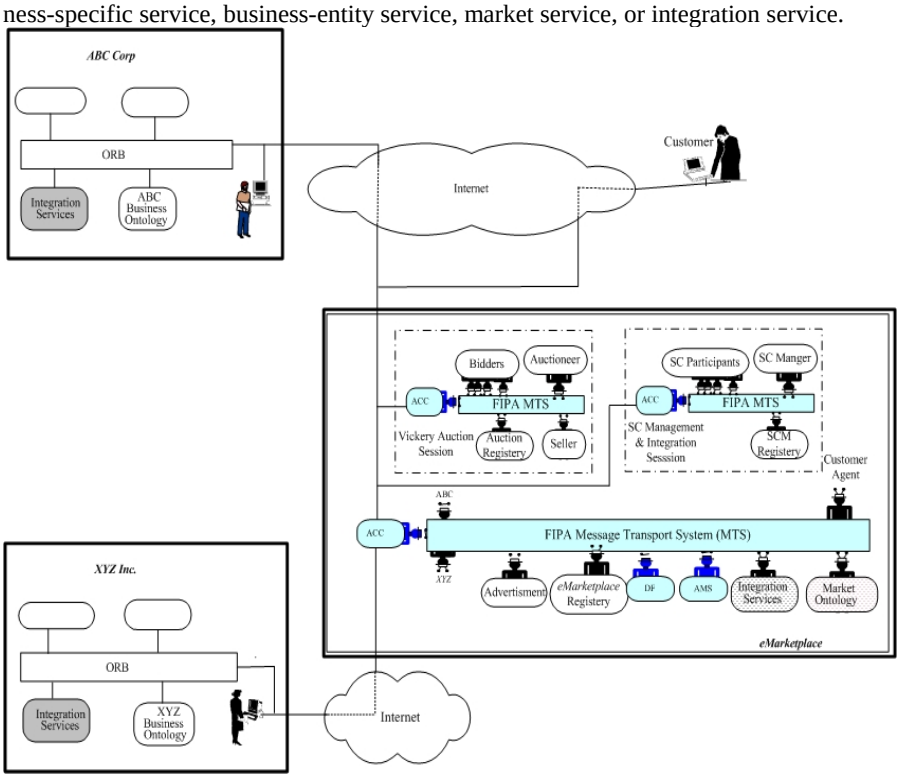


Fig. 4. The architecture of the *eAuction* Market

User interface agents play an important and interesting role in many applications. The main functionality of user interface agents is to support and collaborate with users in the same work environment to achieve the users' goals.

Business-specific service agents are specialists that provide a collection of business-services available in the *eMarketplace*. Performing the functionality of a business service is typically the cooperative integration of several agents including business-specific service agents and market service agents. A business-specific service agent may be a representative in the *eMarketplace* for some functionality that is based on legacy applications or libraries, such as a product catalogue Web site.

Market service agents are specialists that provide a collection of functions for generic *eBusinesses* in *eMarketplace* environments in which a single entity (usually an agent) can perform its tasks in the *eMarketplace*. Market services (value-added and core services) are horizontal, i.e., services that are used in several business domains by several business entities. Here the focus is on core services, particularly dynamic trading services using coalition-based formation of multiple issue negotiation Vickrey

auctions and supply chain integration, which will be discussed further in the following sections.

Integration service agents are specialists that provide a collection of integration functions for a cooperative distributed system in which a single entity (agent, component, object, etc.) can perform its tasks. Integration services are used by several distributed entities. For example, a brokering service provides a capability-based integration in the *eMarketplace*. The brokering agent allows agents (for integration, market, or business services) to describe the properties of a requested service. Then, on behalf of the requester, it establishes interactions with service providers to fulfill the requests. The brokering agent is responsible for identifying and interacting with other integration services, such as resource discovery services and ontology manager services to accomplish its tasks. Another type of integration agent provides view-integration, which is a service to merge and map the description of business-objects (e.g., source schemas) in the *eMarketplace* supported by the business ontology into an integrated view or schema. For instance, a catalogue service might require information provided by several business entities supporting different product schemas. A view integration service provides the integration into a common definition language (e.g., XML-based), which is in turn mapped into a target representation language by a specialized language mapping service. View integration is responsible for identifying and interacting with several services to fulfill its functionality, including brokering, source-schema, ontology, and language-mapping services.

7 Multi-Attribute Negotiation Service: Coalition Deal Negotiation Model

Many researchers have investigated multiple issue negotiation [16][28][36]. One approach [16] developed an optimal agenda and procedure for two-issue negotiation. It introduced two negotiation procedures: *issue-by-issue negotiation* and *package deal.* However, for *n*-issue negotiation, over *n>2 issues*, which is a common setting in many *eMarketplace* applications, the computational cost to reach a package deal might exceed the benefits obtained by optimizing the participants' utilities and becomes impractical. Furthermore, these models often assume that private information of the business entities (software agents herein) is common knowledge while neglecting the associated computational cost. Therefore, these models do not fit typical dynamic and competitive environments. To deal with these challenges, namely utility optimization and computational efficiency, we propose an approach based on the *coalition deal* for multiple issue negotiation [13].

To this end, our approach of a coalition deal negotiation is to extend the properties of issue-by-issue negotiation and the package deal procedure with the flexibility to balance between time and utility.

Definition: *For a coalition deal, all negotiation issues are partitioned into disjoint partitions and each partition is negotiated independently of other partitions. Like the package deal, issues inside the same partition are negotiated as a whole package and an offer includes a value for each issue in this partition.*

Clearly, from the above definition, issue-by-issue negotiation can be treated as a specific case of a coalition deal with one issue per partition. The package deal can be viewed as a coalition deal with one partition for all issues. However, coalition-deal negotiation provides (a) better utility than issue-by-issue negotiation, (b) less computational cost than package deal negotiation, (c) more flexible negotiation, and (d) better management for negotiation. To discuss this formally, consider multiple-issue negotiation with set I of k issues, where $I=\{I_1,I_2,...,I_k\}$. Let IP be the set of all partitions of size s over I, where $IP=\{IP_j | 1\leq j\leq s\}$, *where* IP satisfies the constraint: $\forall 1\leq m\leq s,\ 1\leq n\leq s,\ m\neq n$, such that $IP_m\cap IP_n=\phi$ and $\cup_{j\in IP}\cup_{i\in j} i=I$. For two agents a and b, respectively they can be defined as a 4-tuple of a parameters:

$$S_a=\left\langle \mathrm{P}_a^{IP}, U_a^{IP}, T_a, \delta_a \right\rangle$$

$$S_b=\left\langle \mathrm{P}_b^{IP}, U_b^{IP}, T_b, \delta_b \right\rangle$$

where $\mathrm{P}_a^{IP}=\left\{P_a^i \mid i\in j, j\in IP\right\}$ denotes agent a's set of reserve prices over I and P_a^i denotes a's reserve price over issue i, which belongs to partition j, $U_a^{IP}=\left\{U_a^i \mid i\in IP\right\}$ denotes agent a's utility functions over IP where U_a^i denotes agent a's utility function over one partition I from IP, T_a and δ_a denote agent a's bargaining deadline and discount factor. Agent b's negotiation parameters are defined analogously. An agent's utility from IP of I is the sum of its utilities from all partitions. For a coalition deal, each partition is negotiated separately and independently of other partitions. An agreement can take place either on some of the partitions or all of them. For each partition, an offer includes a value for each issue inside this partition that would be the same as the package deal for this partition. This allows trade-offs to be made between issues inside this partition. An agreement has to take place either on all the issues inside the partition or none of them. For each partition, we assume that the agents use the same protocol as for the package deal, but instead of making a set of offers over I, an agent offers a set of offers over issues from this partition. An agent can make trade-offs only across issues in the same partition, resulting in a set of offer sets, all of which give it equal utility.

At time t, Agent b generates a set of offers over a partition IP_i of k_i issues that give itself the equal utility. We define $P_{b,t}^{IP_i}=\left\langle P_{b,t}^{IP_i(1)}, P_{b,t}^{IP_i(2)}, ...P_{b,t}^{IP_i(k_i)} \right\rangle$ as agent b's current utility optimal offer over partition IP_i for agent a if it gives a the maximum utility. For a coalition deal, each partition is considered using the package deal negotiation protocol. In this context, agent a's action $Ac_{a,t}$ for the coalition deal procedure is defined as follows:

$$Ac_{a,t}=\begin{cases} Quit & if \quad t\geq T_a \\ Accept\ Package\ deal \quad for \quad IP_i & if \quad U_a^{IP_i}(P_{b,t}^{IP_i})\geq U_{a,t+1}^{IP_i}{}' \\ Offer \quad \mathrm{P}_{t+1}(U_{a,t+1}^{IP_i}{}') & for \quad IP_i\in IP \quad otherwise \end{cases}$$

Where $U_{a,t+1'}^{I}$ is the utility value for an agent to generate its count-offer at time $t+1$ over partition IP_i. Similarly, we define agent a as playing its equilibrium strategy for the package deal over a partition if $U_{a,t+1}^{IP_i}{}'=(1-y_{a,t+1}^{IP_i})U_{\max,a}^{IP_i}$, where $U_{\max,a}^{IP_i}$ is the maxi-

mum possible cumulative utility agent *a* can get from partition IP_i. The equilibrium strategy for agent *a* and agent *b* over other partitions is defined analogously.

Coalition Deal Utility and Efficiency

For a given set of issues, $I=\{I_1,I_2,\ldots,I_n\}$, and a partition $IP=\{IP_1,IP_2,\ldots,IP_k\}$, we assume generating a price value for any issue require the same unit of computational cost. Furthermore, we assume issue-by-issue negotiation can be performed in parallel for each issue and the same for every partition in a coalition deal. Therefore, to compare the computational efficiency, we need to compare the computational cost of generating an offer in each round only. If each negotiation type approach requires the same number of rounds to reach an agreement, then we can compare their computational costs by comparing the cost of generating an offer in each round.

An *n*-issue negotiation can be viewed as a distributed search through an *n*-dimensional space. In issue-by-issue negotiation, each issue is negotiated separately. Therefore, this lead to a computational complexity of $O(m^n)$, where *n* is the size of the issue set and each issue may have *m* possible values. The complexity gets worst when we have to solve this problem each round using the package deal negotiation procedure. In coalition deal negotiation, however, issues are partitioned into *k* disjoint partitions and each partition is settled independently of the other partitions. Like the package deal, issues inside a partition are negotiated as a whole and an offer includes a value for each issue in the partition. Therefore, the computation problem is reduced to the sum of *k* independent search problems where the *i*-th search is in an n_i-dimensional space, where $n_i<<n$ and $\sum_{i=1}^{k} n_i = n$. For sequential implementation the corresponding computational complexity is $\mathrm{O}(km^{n_s})$, which can be reduced to (m^{ns}) using parallel implementation, where $n_s = \arg\max n_i$. Moreover, we can limit the maximum size of a partition to a constant *C* without losing value due to interdependency as discussed below. In this case, the computational cost of a coalition deal reduces to $O(nm^C)$. The computational complexity of the parallel implementation will be $O(m^C)$ to generate a coalition deal. Now we discuss some basic ideas for partitioning the set of issues:

- If agent's utility from issues is additive, the issue set could be partitioned arbitrarily and independently, agent can choose different partition sizes corresponding its computation capacity and time pressure.
- In the situation where some issues could be dependent and their utilities are in form of one multi-variable function, one variable for each issue. We can put the dependent issues into the same partitions and assume a more general form of additive utility in which each addend is a multi-variable function for dependent issues.
- It is always possible to form a better utility frontier for one partition by putting issues with distinct comparative interests into the same partition. In real world, people usually put their most interested issues with their opponents' most in-

terested issues into one basket. If the opponent's preference is unknown, people can put their most interested issues together with the issues that they most likely to compromise.

Therefore, the proposed coalition deal provides more flexibility to balance between computation costs and utility gains:

- A business entity can adjust the partition size based on its current time pressure and computation resource.
- Most distributed automated negotiation systems encounter a message congestion problem and the coalition deal can mitigate it. A message congestion problem occurs when an agent cannot process its received messages as fast as they arrive.
- If new issues are introduced by adding new partitions the coalition deal provides better scalability without affecting the existing issues.

Using negotiation mechanism to match supply and demand prevents consumers with low valuations from receiving limited goods and prevents suppliers with high production costs from supplying the limited demand.

8 *eAuction* Market Service

Auctions are dynamic and often efficient mechanisms for selling items in complex *eMarketplaces*. The proposed auction market here incorporates a Vickrey auction mechanism. In private-value Vickrey auctions, an agent's valuation is determined locally and independent from other agents' valuations. It provides incentives to promote truthful bidding among self-interested agents and avoid the computational cost of counter-speculations. Hence, there is no need for iterative negotiation strategies, dynamic strategic behavior to reveal other bidders beliefs, or high degree of security. Furthermore, it has been shown that this mechanism can achieve the same utility for the participants as other less direct mechanisms where bids might not be the true valuation [49].

8.1 The Auction Market System Architecture

Our proposed auction market is a collection of auction sessions along with some generic services required by all markets such as the market registration, advertisement and search engines. Each auction session has an initiator agent represented by the auctioneer-agent to sell or buy items or services under particular auction rules. The term "session" emphasizes the temporally extended nature of these auction events or the market and enables the formation of spot markets. A forward auction session (regular or a spot market) extends from the initial call-for-bids through the negotiation, awards, and final closing.

Generically, each session mechanism should ensure the continuity of competitive bidding and deliver dynamic tracking of the negotiation status. A multilateral market can also be applied for supply chain management, in which both supply and demand curves are constructed to determine a market-clearing price. In fact, several market

mechanisms can be applied to model the multilateral market. For example, Wilson [53] initiated the study of double auction as a means to model multilateral trading. In such a double auction, all trades are made at a single market clearing price. McAfee [33] proposed a double auction model that explicitly considers the role of an auctioneer who intervenes in the trade and keeps track of supply and demand at asked and bid prices. Babaioff and Nisan [4] proposed protocols for exchange of information between multilateral markets along a single supply chain. Each market form a link in the supply chain operates as a double auction, where the bids on one side of the double auction come from bidders in the corresponding segment of the industry, and the bids on the other side are synthetically generated by the protocol to express the combined information from all other links in the chain.

One of the common services provided by an auction market structure is the "Registry", through which business-entities, suppliers and consumers, register with the *eMarketplace* and corresponding representative agents are created. The entries in the agent's registry database would include the identity of business-entity agent, its role as a supplier or consumer, its list of preferences, such as product, price range, and specific supplier brand. Potential suppliers advertise their goods or services with the *advertise agent*. The *search agent* represents the first interaction with the trading agents after they join the *eMarketplace*. It is deployed to expand the supplier and buyer agents' awareness by recommending a certain auction session based upon their reported preferences and/or product or service specifications.

Auction sessions can be seller centric "forward" auction or buyer (consumer) centric "reverse" auction. A seller-centric auction helps a supplier-agent in maximizing its revenue at equilibrium, whereas a buyer-centric auction helps a buyer-agent in identifying the lowest price at equilibrium. In bilateral markets a combination of double auction (both forward and reverse auctions) and supply-demand profile integration help a market-based supply chain determine efficient allocation.

Any potential supplier or consumer (buyer) agent that wishes to join an auction session is required to register once again with the auction's *registry service* a priori to be able to participate in the bidding process. Finally, the *auction session agent* can be viewed as a repository of auction protocols and auction initialization parameters. Once an auction event is decided upon, the auctioneer-agent is able to extract the auction parameters such as start bid, ask price, and increment bid from the *auction session agent*. In addition, the *auction session agent* will provide the auctioneer-agent with the specifications of interaction rules such as the way it will be performed either open or sealed, the direction of bidding progression either ascending or descending and the stopping criteria.

8.2 Market Session: Vickrey Auction

The basis of work presented here is limited to the Generalized Vickrey Auction (GVA). We restrict our discussion on the forward auction protocol. In a single item forward auction, each agent, A_i, has a "maximum willingness to pay" or "value" for the item that is assumed to be private information known only by the agent A_i, de-

noted by v_i. Then, the objective is to award the item to A^*_i with the highest bid value. Although the Vickrey auction ensures that the item is awarded to the highest bidder-agent, but it does not maximize the supplier's revenue unless "reserve prices" are imposed [25]. Given a set of combinatorial bids, the supplier-agent (through auctioneer-agent) then decides how best to allocate individual goods to those bundles for which bids were placed, with the objective to maximize revenue. Groves' mechanism in *combinatorial auctions* is a combinatorial allocation problem (CAP). It involves allocating bundles of items that may overlap to *n* bidders. CAP is an optimization problem equivalent to the weighted set packing problem [40]. The proposed auction market is comprised mainly of three agents: the *Auctioneer-agent*, *the Bidder-agent* and the *Supplier-agent.* Under the general Vickrey mechanism, it is in the interest (the dominant strategy) of the bidder to report its true valuation function. Then, the auctioneer-agent

- calculates the allocation (x_i^*) that maximizes the sum of the bids subject to the items constraint;
- calculates the allocation $(x_{\sim i}^*)$ that maximizes the sum of the bids other than that of bidder agent *i* such that it excludes all items allocated to agent *I*;
- announces the winners and their payment given by

$$p_i = \sum_{j \neq i} v_j(x_{\sim i}^*) - \sum_{j \neq i} v_j(x^*)$$

Under the assumption of quasi-linear preferences, each bidder-agent calculates its utility. For bidder-agent *i* the utility will be

$$u_i(x^*) = v_i(x^*) - p_i = v_i(x^*) - \sum_{j \neq i} v_j(x_{\sim i}^*) - \sum_{j \neq i} v_j(x^*)$$

From the computational perspective, the above GVA introduces several challenges at two distinct levels. At the auctioneer-agent level, the winner determination is an *NP-hard* optimization problem in which the auctioneer needs to solve the problem twice: first with all participating agents reporting their preferences and then with each agent removed from the system to compute payments. At the bidder-agent level, agents must compute and communicate their complete valuations for an exponential number of bundles of items for different outcomes, each of which might involve solving a *hard local optimization problem*. These computational issues need to be resolved without losing the main game-theoretic properties, especially allocative-efficiency and strategy-proofness. Firstly, the GVA requires complete information revelation from each agent. Secondly, the valuation problem for a single bundle can be hard [40], and in combinatorial domains there are an exponential number of bundles to consider. Thirdly, agents must communicate that information to the auctioneer, which might be costly in network resources in addition to security problems

Fortunately, in sealed-bid auctions, as with our case, the bidding rules amount to no more than a bidding language, i.e., the syntax and semantics of bids. Agents would require a representation language, such as OR* [37], with expressive and compact methods to accurately express their preferences, make the bid structure more explicit and reduce the intractability of the valuation complexity. In other words, rather than requiring the agent to solve valuation problem locally, it simply sends the local prob-

lem specification directly to the auctioneer. In this manner, strategy-proofness is not reserved, given that the auctioneer can interpret the bidding language faithfully.

The second challenge concerning the computational complexity of the auctioneer's *winner determination* problem can be handled by several optimal and approximations algorithms [17]. In our case, once all agents have placed their bids in the OR* bidding language for any arbitrary set of bids, and if these bids conform linear-order bids, nested-hierarchical bids or a combination of both structures, then the linear program guarantees an exact optimal solution. This linear formulation can utilize standard algorithms and hence can be run directly on standard commercially available optimization software, such as CPLEX [26]. In other cases, the linear programming relaxation may not yield non-integral-solution. Thereby, heuristics can be used to somehow find an approximation to the optimal solution. A *greedy* algorithm that provably runs in polynomial time, however does not guarantee to always produce an optimal solution. Alternatively, a recursive *branch-and-bound* algorithm based on an upper bound LP solution can be used. This technique trims the search space in exponential searches and can provably produce an optimal solution, however, does not guarantee to always run in polynomial time.

8.2.1 Auction Session Module

In this section, we focus on the Vickrey auction market components and functionalities as an integral *eBusiness* model in a BCKOA-based *eMarketplace*. The auction market mainly recognizes three types of agents representing the auctioneer, suppliers, and consumers. These agents represent specialist market service agents that provide a collection of functions to achieve the intended business objective of auctions. However, the trading process mainly involves the auctioneer and consumers (or bidders) agents. Initially, both supplier- and consumer-agents register with the auction session. The supplier-agent then formulates a sell-order and assigns it to an auctioneer-agent that has the capability to carry out the selling task on its behalf in an auction session. The auctioneer-agent receives a request from the supplier-agent to process the sell order. Consequently, the auctioneer-agent formulates an "announcement" including all the auction parameters supplied by the supplier-agent such as the item details, quantity, minimum price and desired selling deadline. This "announcement" is encapsulated inside a "call-for-bid" message that is sent out to all potential bidder-agents registered with the current auction session.

At a certain time, the auctioneer-agent sends "propose" messages to all bidder-agents indicating the start of the bidding process. Bidder-agents express their preferences in OR* bid format and sends "bid" messages back to the auctioneer-agent. When the selling deadline reaches, the auctioneer-agent ceases to accept any messages and send a "reject" message for late arriving messages. Immediately, the bid evaluation process commences by comparing the structure of aggregated bids against certain classes of bids that indicate an optimal allocation of winners and calculation of payments is ultimately guaranteed. Otherwise, the auctioneer-agent runs a heuristic algorithm in order to arrive at a computationally feasible approximate winner allocation. Finally, the auctioneer-agent sends "inform" messages to all the bidder-agents as well as the supplier-agent to notify them of the winner and his payment. Each bidder-agent is responsible for notifying the consumer of the auction outcome. The supplier-

agent must also inform the human supplier (or the supplier's and consumer's representative-agents in this case) of the auction outcome at the end of the auction session.

8.2.2 Agent Architecture

Each agent's architecture is based on the CIR-agent model [21]. The primitive components of the CIR-agent are arranged in a particular order to represent each agent's role and capability. Basically, each agent consists of *knowledge* and *capability* packages, each of which is tailored according to the agent's specific role. An auction session mainly recognizes three types of agents, namely, supplier-agent, auctioneer-agent and bidder-agent. The architecture of each agent type is described in detail below.

An auctioneer-agent is composed of two packages: *knowledge* and *capability*. The *knowledge* package contains the information in the agent's memory about the environment and the expected world. This includes the agent self-model, other agents' model, goals that need to be satisfied, possible solutions generated to satisfy each goal, and the local history of the world that consists of all possible local views for an agent at any given time. The agent's knowledge also includes the agent's desires, commitments and intentions toward achieving each goal.

The *capability* package includes the *reasoning* component, the *domain actions* component which contains the possible set of domain actions that when executed the state of the world will be changed, and the *communication* component where the agent sends and receives messages to and from other agents and the outside world. The *reasoning* component is further decomposed into the *problem-solver* component and *coordination* component. The *problem-solving* component maps the agent's goal into a solution. For example, the auctioneer's goal is to determine a market price for an item of unknown value. The problem-solver involves aggregating bids and evaluating them against a certain criteria. The *coordination* component includes several interaction devices that deal with various interdependency problems, such as capability dependency and conflict of interest [21].

Fig. 5 depicts the logical structure of the *reasoning* component representing the transformation process of the mental state of the auctioneer-agent, once a goal pops-up. Therefore, the arrival of OR* bids via agents communication messages coming from the bidder-agents through the *communication* component assigns a goal-state. The problem-solver component is responsible for the collection of bids and the winner determination process to allocate the item or bundles and their respective prices. Through the *interaction* component the auctioneer-agent sends out an "accept" message at the beginning of auction session informing the supplier-agent that the auction will be carried on its behalf. Once the auction ends, the outcome must be reported to all participants. This information is encapsulated in outgoing messages and sent to the outside world through the *communication* component.

Similarly, as shown in Fig. 6, the bidder-agent is designed using the same CIR-Agent architectural principles. The exception is the classes that constitute the *problem solver* component, which are tailored according to its specific role and functionality as described in Vickrey auction session module. The bidder-agent interacts with the consumer via a user-interface designed for the OR* language to express the atomic bids as well as constraints that signify which item/bundle is mutually exclusive. The

formulated bid represents the body of the agent's communication message that will be sent to the auctioneer-agent via the communication module.

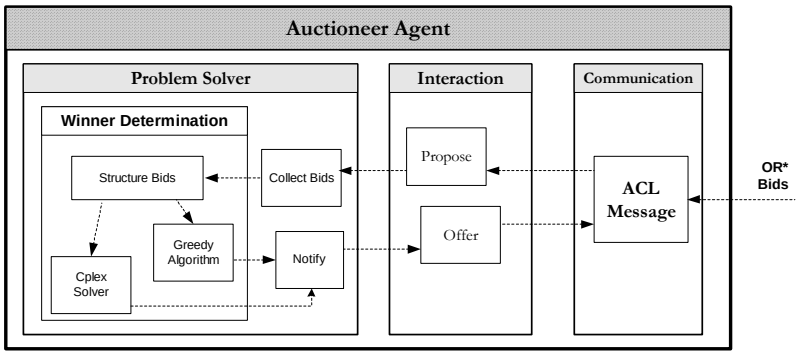


Fig. 5 Architecture of the auctioneer agent's reasoning component

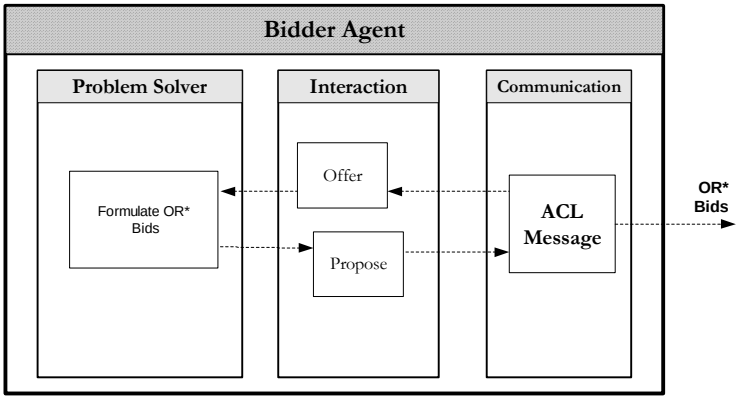


Fig. 6 Architecture of the bidder agent's reasoning component

9 Agent-Based Supply Chain Integration Service

An *eMarketplace* enables the creation and leveraging of services and supply operations in a way that seamlessly integrates business participants in a dynamic trading community. Negotiations and auction-based markets provide efficient coordination mechanisms for the supply chain. As described in Section 3, in relationship-based supply chains, long-term relationships usually have higher value than that of the efficient resource allocations. BCKOA *eMarketplace* addressed these challenges by providing a platform that combines both relationship-based and market-based coordi-

nation mechanisms. Utilizing multi-issue negotiation services the agents can create a spot market, in which the set of issues can include both price and non-price attributes. According to different context, agents may adopt different mechanisms to balance between long-term relationships and prices in a negotiation by assigning different weights to relationship issues and price issues in a utility function.

For supply chain integration, our proposed methodology requires the description of the supply processes involved between participating business entities using OAG standard business documents and UML interaction diagrams. The methodology begins with capturing a supply chain scenario and its associated UML interaction diagrams, exemplified in Fig. 7. The interactions, as shown in Fig. 7, consist of the exchange of structured documents, such as the OAG business-object documents (BODs). For B2B interactions, a ProcessPO BOD is a *directive* that carries the composite semantics of *request* and *inform*, in which the sender requests that the recipient evaluates a purchase order and informs the sender of the results. The informal semantics is that ProcessPO will be followed by a response from the recipient and that the response will be either an AckPO or a DeclinePO. Using the semantics of each document, the messages in the interaction diagram are converted into a bipartite conversation graph (not shown here), which delineates each participant's conversations. A bipartite conversation graph is used to identify the roles of the participants in B2B transactions. This graph is the basis for constructing Dooley graphs [14], shown in Fig. 8 as collaboration diagrams.

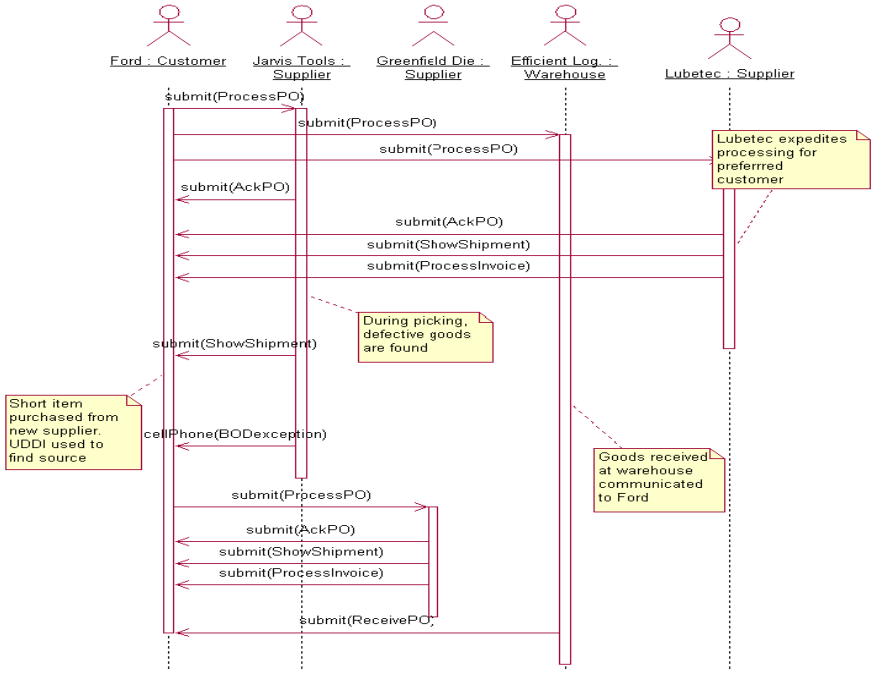


Fig. 7. Interaction Diagram for the OAG Scenario Involving Ford and its Suppliers

Note that collaboration participants can fill different roles at different times, and thus can be involved in many conversations simultaneously. Each of the roles identified in the collaboration diagram can be assigned to an agent in the supply chain. Moreover, the diagram for each role is mapped directly into a state-machine description for the agent's behavior. The corresponding agents then carry on the roles and for the affiliated business entities, and then collectively they manage the supply chain process.

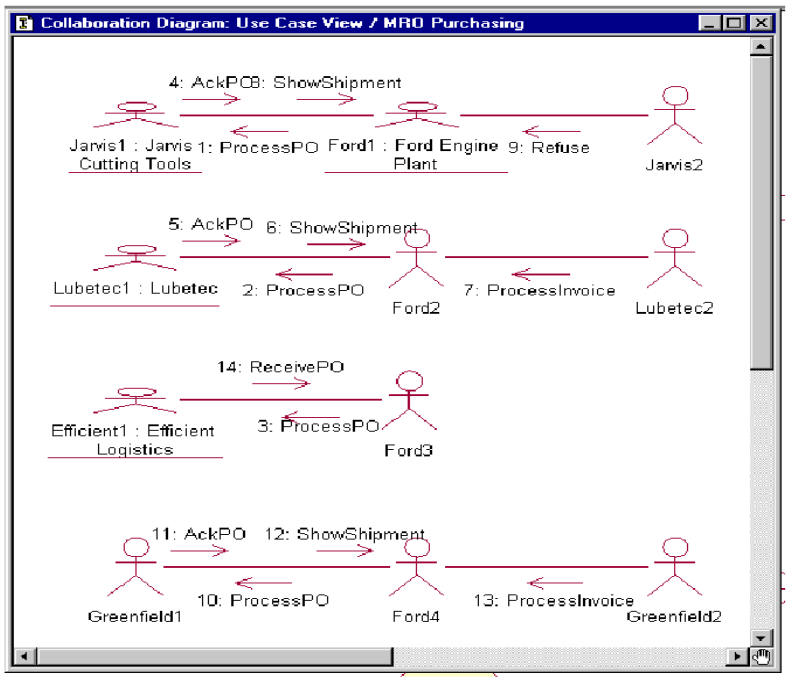


Fig. 8. Collaboration Diagram with Participant Roles for Ford Interoperability Scenario.

The methodology, summarized in Fig. 9, uses—and begins to formalize—the BODs that OAG and RosettaNet are standardizing. It provides a basis for the convergence of multiple standards for supply chain management, which could become ready-to-use technology for different participant business entities in the *eMarketplace.*

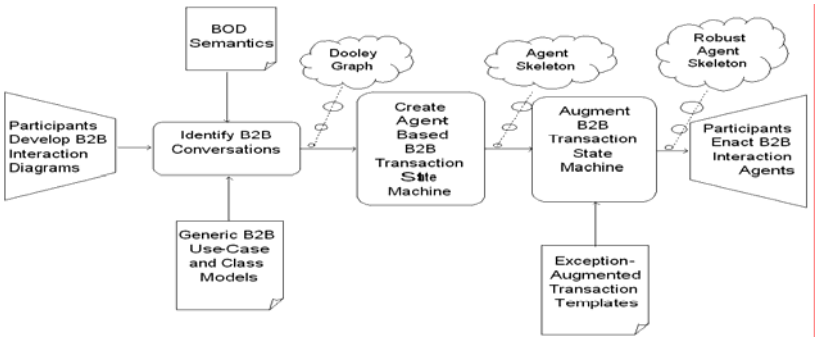


Fig. 9. Agent-Based Coordination Methodology for B2B Automation

10 Prototype Implementation

To validate and experiment with our analyses and foundations described in the previous sections, we have developed a prototype of an agent-oriented BCKOA for an *eMarketplace,* as shown in Fig. 4. *ABC* Corp and *XYZ* Inc. are virtual business entities registered with the *eMarketplace* for both purchase and sales services. Both organizations use a BCKOA-based computation environment. Individual customers or business-entity personnel in the *eMarketplace* can participate in the market through their user interface agents. Similarly, an agent in the eMarketplace represents each business-entity service. These agents provide thin, intelligent, highly autonomous interfaces for the business-entity services that might be based on legacy applications. For example, the *ABC* purchasing-service agent represents the implementation of the business-specific purchases by *ABC* in the *eMarketplace*. Each user interface and business-entity service agent is registered in the *eMarketplace*. Thus, a user interface agent can benefit from the market, business-specific, and business-entity services by interacting with their representative agents. Each business-entity service must also be registered with a registry agent for the corresponding business-specific service. Each layer, and its registry services, are intended to provide some aspect of information about the *eBusiness* environment and enable an interested party to obtain information to potentially use offered services, or to join the *eMarketplace* and either provide new services or interoperate as a trading partner with other business-entities in the *eMarketplace.*

In the current prototype, we experiment with auction market structures, as shown in Fig. 4, where customers and suppliers are brought together to trade with each other and prices are set by the selected market structure. An individual customer is able to participate in the market through a dedicated user interface agent possibly assigned by the *eMarketplace*. Similarly, each participating business entity is assigned to a team of CIR-Agents for the registered services and representative personnel who might have a direct contact with the market as well as with the customers. The market ontology provides a conceptualization of the domain at the knowledge level.

The implementation utilizes the JADE platform [6], which is a software framework to develop agent applications in compliance with the FIPA specifications (The Foundation for Intelligent Physical Agents 1998) for multi-agent systems. JADE deals with all aspects that are external to agents and independent of their applications. These include message transport, encoding, parsing and agent lifecycles. JADE supports a distributed environment of agent containers, which provide a run-time environment optimized to allow several agents to execute concurrently. This feature has been utilized to create several concurrent market sessions, such as commodity and auction sessions. JADE provides support for standard FIPA ontologies and user-defined ontologies. Although our implementation takes advantage of the JADE platform and its supporting agents, such as a directory facilitator, the architecture of the application agents is based on the CIR-Agent model (shown in Fig. 3). Java features, such as portability, dynamic loading, multithreading, and synchronization support make it appropriate to implement the inherent complexity and concurrency in an *eMarketplace*. These features are also instrumental for executing the CIR-Agents concurrently. The design of each agent is described in terms of its knowledge and capabilities. The agent's knowledge includes the agent's self-model, goals, and local history of the world, as well as a model of its acquaintances. The agent's knowledge also includes its desires, commitments, and intentions as related to its goals.

The main capabilities of the CIR-Agent include communication, reasoning, and domain actions. Implementation of the communication component takes advantage of JADE messaging capabilities. It is equipped with an incoming message inbox, whereby message polling can be both blocking and non-blocking, and with an optional timeout mechanism. Messages between agents are based on the FIPA ACL. The agent's reasoning capabilities include problem solving and interaction devices. The problem solving of an agent is implemented through the use of complex behaviors. Behaviors can be considered as logical execution threads that can be suspended and spawned. The agent keeps a task list, containing active behaviors. The problem-solving component varies from one agent to another as will be the in the following subsections. The agent behaviors can be classified as follows: behaviors that are concerned with market services, such as a market-registry and auction services; and behaviors that are concerned with providing business-specific services, such as selling and purchasing.

10.1 Auctioneer-Agent

The role of the auctioneer agent is to formulate the auction proposal, sends it out to all registered bidder-agents and collects the bids. At the end of the bidding-time, it evaluates the bids, determines the winner bidder-agent along with the allocation and payment and notifies both the supplier-agent and bidder-agents of the outcome. As described before, the auctioneer-agent is a collection of *knowledge* and *capabilities* components. The knowledge component includes the agent's self-model, model of other agents, and the local history. The main capabilities of the CIR-Auctioneer agent include *communication, reasoning* and *domain actions*. The *communication* component consists of JADE *ACLMessage* class that is implemented by the problem-solver

or the coordination components to construct messages via several FIPA performatives such as REQUEST, INFORM, QUERY-IF. The auctioneer-agent's *reasoning* component consists of the *problem-solver* and *coordination* components, which are described below in detail.

The auctioneer-agent's *problem-solver* component contains a set of *Behavior* classes that represent the auctioneer-agent's specific tasks. These classes implement all the agent's platform tasks such as registration with the DF service. Also, they represent *simple behaviors* and *cyclic behaviors* to handle the incoming messages from both the supplier-agent and the bidder-agents. A basic set of methods that can be called to implement the application tasks of the auctioneer-agent such as the "*callForProposal*" method that formulates the "*announcement*" of an auction event, the "*updateBidList*" method that checks the auction's start time, and collects the bids and the "*informAuctionResults*" method for notification purposes. The winner determination class is described in detail below.

In case of combinatorial auctions, the auctioneer-agent has to solve the winner determination problem as an optimization problem expressed as an integer-program that is further relaxed into a linear-program. The "*structureBid*" method in BidEvaluation class determines whether a tractable or intractable solution is necessary according to certain restrictions on the size of OR*-bids. Interfacing to standard algorithms such as the CPLEX solver solves the tractable cases. AMPL (A Modeling Language for Mathematical Programming) was used to formulate a model of the optimization problem, as shown in Fig. 10, in order to be input to the CPLEX Solver. AMPL is a comprehensive and powerful algebraic modeling language for linear and nonlinear optimization problems, in discrete or continuous variables. AMPL's familiar algebraic notation makes it ideal for rapid prototyping and development of models as well as it can communicate with a wide variety of solvers. For intractable cases we use a greedy allocation algorithm [37] to find an approximate allocation.

The coordination component contains a class that extends a JADE behavior class namely the FIPA *ContractNetInitiatorBehavior* through which the auctioneer agent sends an INFORM message, which contains the auction specifications and the sell parameters to a set registered bidder-agents. The bidder agents can respond either by a PROPOSE-message (followed later by a BID), or a REFUSE message to reject participation in the auction and finally and NOT-UNDERSATND message in which case the auctioneer agent has to resend an INFORM message with the auction specifications and the sell parameters.

```
set n;                              # n is set of atomic bids
param p {i in n};          # p is bundle price
var x {i in n};                     # x is the winning bundle
maximize Total_Price: sum {i in n}x[i] * p[i];
# objective total of winning prices is maximum
subject to sum_Bundle: sum {i in n} x[i] <= 1;
#constraint: without allocating a bundle more than once
subject to Bundle_value {i in n}: 0 <= x[i]; # non-negative bundle
values
```

Fig. 10 Combinatorial Auction model in AMPL

10.2 Bidder-Agent

The role of the bidder-agent is to express its preferences in OR* bid format and sends "bid" messages out to the auctioneer-agent. The *problem-solver* component contains a Bid class that implements a *cyclic behaviour* in order to respond to incoming messages from the auctioneer-agent that requests bids. This class implements all the bidder-agent's tasks such as registration with the DF service as well as a single method that formulates preferences and bid valuations in OR* language. The *coordination* component contains a class that extends the JADE behavior class *ContractNetReponder Behavior* through which the bidder-agent prepares the PROPOSE message that is later followed by the formulated OR* Bids or the REFUSE message to refuse the proposal or the NOT-UNDERSTAND message to request for resending the information of the announcement again.

10.3 Supplier -Agent

The supplier-agent's role is to formulate the sell-order and assigns it to the auctioneer-agent. The supplier-agent's reasoning component consists of the following: The implementation of the supplier agent's *problem-solver* component, includes the "*SellOrder*" class that implements a method called *processSellOrder* that displays the user-interface through which the human supplier enters the item descriptions. The *SellOrder* class implements a *cyclic behaviour* that responds to incoming messages from the auctioneer-agent that either notifies the supplier of the auction outcome or requests the supplier-agent to leave the auction once the auction session has ended. This component contains the "*SellOffer*" class. It formulates a REQUEST message to ask the auctioneer-agent to carry out the selling task on the behalf of the supplier-agent. It also implements a simple behavior to respond to auctioneer-agent ACCEPT OR REJECT messages.

11 Related Work and Discussion

There have been several recent attempts to promote *eMarketplace* models by the academic and industrial communities. For example, the electronic market-place (EMP) [8] was an attempt to develop a business-to-business system architecture. It is viewed as a DBMS solution to support many-to-many relationships between customers and suppliers. The Global Electronic Market (GEM) [42] attempted to develop a logical market framework and infrastructure. A main objective was to separate system-related and market-related design issues. In GEM, the market provides trading mechanisms that include bids and offers. A more complex architecture for *eMarketplace* is MAGMA [48], with its special focus on the infrastructure required for conducting commerce on the Internet. In MAGMA, the *eMarketplace* has been viewed in terms of three main functionalities, namely, traders, advertising, and banking. Alternatively, OFFER [7] proposed a brokering-based marketplace. In OFFER the *eMarketplace* is viewed as a collection of suppliers, customers, and brokers. A customer

can search for a service either directly in the e-catalog of the supplier or use the e-broker to search all the e-catalogs of the suppliers that are registered with this broker. E-brokers employ a simple auction mechanism. In a different approach, MOPPET [3] proposed an *eMarketplace* system as agent-oriented workflows. MOPPET viewed the market as a workflow management system carried out by several types of agents: task, scheduling, facilitator, and recovery agents.

Another approach was driven by the bottom-up modeling of market processes with self-organizing capabilities [2]. The objective was to develop a computational study of economies modeled as evolving systems of autonomous interacting agents, and known as agent-based computational economics (ACE) [29][46]. The ACE researchers relied on computational laboratories [33] to study the evolution of decentralized market economies under controlled experimental conditions. The goal was to develop analysis tools that enable an economist to test economic theories developed using standard modeling approaches

Several companies have emerged to automate logistics and re-supply within specific industrial segments. For example, Ariba [1] developed a marketplace based on procurement portals and dynamic exchanges for horizontal marketplaces. Ariba Dynamic Trade, for instance, attempts to provide dynamic trade mechanisms, such as auctions, reverse auctions, and bid/ask exchanges and negotiations. SAP Service Marketplace (SAP Services Marketplace, SAP AG) is an Internet portal for the SAP community. It provides basic online services such as catalog browsing, matchmaking, and ordering from SAP and its partners. Other approaches were directed to support vertical marketplaces, such as PaperExchange [39], that enables customers and suppliers to negotiate pricing and transact directly with one another. PaperExchange also attempts to provide several supporting services, such as logistics and clearing services, industry-specific job listings, industry events, news headlines, and a resource directory. VerticalNet [50] also built a set of Web-based marketplaces for specific industrial segments, such as financial services, healthcare, and energy. Each Web site forms a community of vendors and customers in a specific area. Vertical trade communities are introduced in segments with a substantial number of customers and suppliers, fragmentation on both the supply and demand sides, and significant on-line access

Another direction adopted by many major software vendors is to develop Internet-based commerce platforms. Examples are IBM CommercePOINT [25], Microsoft Site Server Commerce Edition [35], Oracle Internet Commerce Server INTERSHOP [27], and Sun JavaSoft JECF (Java Electronic Commerce Framework) (Sun Microsystems). These proprietary attempts focus on providing infrastructure services such as security payment directories and catalogs to be integrated with existing systems and the Web.

The proposed agent-oriented BCKOA *eMarketplace*, however, provides a framework of enterprise integration that deals with several systems and business issues. Unlike the above-mentioned attempts, it is fundamentally based on business integration rather than systems integration. The objective is to develop an architecture that is semantically rich in describing an organization and the interconnection among all elements of the *eMarketplace*, including people, business services, software components, and business ontologies. Technologically, BCKOA is service-oriented in the

sense that it enables business entities and their supporting systems to join the market at the highest abstraction level (service level) with minimum overhead and independent of specific technology. Utilizing BCKOA, the business-object implementations of the participating services can be integrated in the execution environment. In addition, BCKOA provides an appropriate architecture for an *eMarketplace*. The form of the architecture supports *eMarketplace* functions that are inherited from real-world marketplaces. They are complex and nondeterministic, yet they characterize a real business environment. A BCKOA *eMarketplace* provides an integration environment for the broad-based services that are required for interaction and directory services, dynamic trading, cooperative supply chain integration and management. Therefore, we believe that the BCKOA-based *eMarketplace* is appropriate for integrating horizontal business services, vertical business services, specific business functionalities, and leveraging legacy systems in a way that supports end-to-end integration. Furthermore, a BCKOA *eMarketplace* provides a wide variety of coordinating and trade mechanisms to fit multiple business models. In our research we have applied a flexible computational economy model for the market services layer. Therefore, a BCKOA-based *eMarketplace* incorporates mechanisms for different types of market structures.

We believe that agent-orientation is an adequate paradigm for producing the information architecture of next-generation *eBusiness* systems, especially *eMarketplaces*. Agent technology richly enables and supports the automation of complex tasks and yields systems that are reliable and able to assume the responsibilities of the *eMarketplace* in which they compete. The components of agent-based BCKOA, namely, *eMarketplaces*, business entities (products, suppliers, customers, etc.), and the foundation (integration) architecture and services that glue them together, are essential to building robust many-to-many value chains in emerging *eBusiness.*

12 Conclusions

This chapter presents our ongoing research on developing an agent-oriented architecture for an *eMarketplace* that provides intelligent integration of collaborative supply chains. The objective is to develop an engineering foundation for the *eMarketplace*. To this end, several *eMarketplace* design and integration issues have been addressed. The focus is on developing a foundation architecture that supports services ranging from baseline integration services to specialty business-related services, and to integrate business organizations in an open market environment.

This chapter presents an agent-based Business-Centric Knowledge-Oriented Architecture (BCKOA) for an *eMarketplace* model. BCKOA is a service-oriented architecture for cooperative distributed systems independent of any specific technology. In the BCKOA based *eMarketplace* several types of agent roles are identified: user-interface, business-specific services, market services, and integration services. This chapter proposed an agent-oriented dynamic trading mechanism (Vickrey auction) and multi-issue negotiation mechanisms for supply chain. Also it proposed a methodology based on OAG standard business documents and UML interaction diagrams for supply chain integration. A prototype of BCKOA using CIR-Agent model and a

FIPA-compliant platform, JADE, is developed with special focus on an auction market structure.

Currently, the objective is to demonstrate the feasibility and the effectiveness of the proposed architecture as a design and integration model for *eMarketplaces* platform for supply chain management and integration. In continuing our research, the computational effectiveness of the architecture will be the main concern. Also, we will expand the application and the implementation of our prototype *eMarketplace* to investigate the most appropriate techniques to support secure, reliable, and effective transactions.

Acknowledgements

This work has been partially supported by Mackenzie Financial Corporation and National Sciences and Engineering Research Council of Canada (NSERC). The authors would like to thank Mr. Hussam Dafaalla and Dr. Yinsheng Li for their help in the implementation of the prototype and the SOAStudio.

References

1. Ariba, B2B Marketplaces in the New Economy, (n.d). from http://www.commerce.net/other/research/ebusiness-strategies/2000/00_07_r.html, (October 19, 2000).
2. W.B. Arthur, J. Holland, B. LeBaron, R. Palmer, and P. Tayler. Asset pricing under endogenous expectations in an artificial stock market model. Proceedings on the Economy as an Evolving Complex System. W.B. Arthur, S.N. Durlauf, and D.A. Lane (Eds.). (1997).
3. S. Arpinar, S. A. Dogac, N. Tatbul, An Open Electronic Marketplace through Agent-based Workflows: MOPPET, International Journal on Digital Library, 3 (1) (July,2000).
4. M. Babaioff and N. Nisan, “Concurrent auctions across the supply chain”, Proceedings of the 3rd ACM conference on Electronic Commerce, 1-10. ACM Press, 2001
5. A. Bartelt, W. Lamersdorf, A Multi-Criteria Taxonomy of Business Models in Electronic Commerce, Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms, (2001).
6. F. Bellifemine, A. Poggi, and G. Rimassa, JADE – A FIPA-compliant agent framework, Proceedings of PAAM'99, London, (April 1999) 97-108.
7. M. Bichler, C. Beam, and A. Segev, OFFER: A Broker-centered Object Framework for Electronic Requisitioning, IFIP Conference Trends in Electronic Commerce, (1998).
8. S. Boll, A. Gruner, A. Haaf, and W. Klas, EMP-a database driven electronic marketplace for business-to-business commerce on the Internet, Journal of Distributed and Parallel Databases, Special Issue on Internet Electronic Commerce 7 (2) (1999, April).
9. G. Cachon and M. Lariviere, "An Equilibrium Analysis of Linear, Proportional, and Uniform Allocation of Scarce Capacity," IIE Transactions 31, No. 9, 835-850 (1999).
10. D. Carlton, "The Theory of Allocation and Its Implications for Marketing and Industrial Structure: Why Rationing is Efficient," Journal of Law & Economics 34, No. 2, 231-262 (1991).
11. DAML-S 0.7 Draft Release, (n.d). http://www.daml.org/services/daml-s/0.7

12. J. Dang, D. Shrotri, and M. N. Huhns, Distributed Coordination of an Agent Society Based on Obligations and Commitments to Negotiated Agreements," In Challenges in the Coordination of Large Scale Multiagent Systems, P. Scerri, Ed.: Springer Verlag, Lecture Notes in Computer Science, 2005.
13. J. Dang and M. N. Huhns, "Coalition Deal Negotiation for Services," presented at 1st International Workshop on Rational, Robust, and Secure Negotiations in Multi-Agent Systems, AAMAS 2005, Amsterdam, Netherland, 2005.
14. R.A. Dooley, Repartee as a Graph. In R.E, Longacre, (Ed.) An Anatomy of Speech Notions, (Lisse: Peter de Ridder Press, 1976) 348-358.
15. M. Dubosson, A. Osterwalder, and Y. Pigneur, eBusiness Model Design, Classification and Measurements. Thunderbird International Business Review, 44 (1) (2002).
16. S. S. Fatima, M. Wooldridge, and N. Jennings, "Optimal negotiation of multiple issues in incomplete information settings," presented at Third International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS'04), New York, USA, 2004.
17. Y. Fujisima, K. Leyton-Brown, and Y. Shoham “Taming the computational complexity of combinatorial auctions”, In Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, Stockholm, (1999) 548-553.
18. L. Garciano and L. Kaplan, "The Effects of Business-to-Business E-Commerce on Transaction Costs," Journal of Industry Economy 49, No. 4, 463-485 (2001).
19. M. Genesereth, An Agent-Based Approach to Software Interoperability, In Proceedings of the DARPA Software Technology Conference, (1992).
20. H. Ghenniwa, eMarketplace: Cooperative Distributed Systems Architecture, 4th International Conference on Electronic Commerce Research, Dallas, Texas, USA. (November 2001).
21. H. Ghenniwa, and M. Kamel, Interaction Devices for Coordinating Cooperative Distributed, Intelligent Automation and Soft Computing, 6 (2) (2000) 173-184.
22. W. Grey, T. Olavson, and D. Shi, “The role of e-marketplaces in relationship-based supply chains: A survey” IBM Systems Journal, Vol. 44, No 1, 109-123 (2005)
23. M. Hammer, and J. Champy, Reengineering the Corporation, (Harper Collins 1993).
24. N. Ivezic, M. Barbacci, D. Libes, T. Potok, and J. Robort, An Analysis of a Supply chain Management Agent Architecture, Proceedings of the Fourth International Conference on Multiagent Systems, (IEEE Computer Society Press, Los Alamitos, Calif., July 2000) 401-402.
25. IBM Corporation CommercePOINT Payment. (n.d). From http://www.internet.ibm.com.commercepoint.payment
26. ILOP CPLEX Suite, (n.d.), From www.cplex.com.
27. Intershop Communications, Inc. Intershop 3, from http://www.intershop.com. (1998).
28. C. M. Jonker and V. Robu, "Automated Multi-Attribute Negotiation with Efficient Use of Incomplete Preference Information," presented at Third International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS'04), New York,USA, 2004.
29. B. LeBaron, Agent-based computational finance: Suggested readings and early research, Journal of Economic Dynamics and Control, 24 (2000) 679-702.
30. Y. Li, H. Ghenniwa, and W. Shen, Integrated Description for Agent-Oriented Web Services in eMarketplaces, Proceedings of AI’2003 Workshop on Business Agents & the Semantic Web, Halifax, NS, (2003) 11-17.
31. Y. Li, H. Ghenniwa, and W. Shen, Web Service-Oriented Agents and Integration Description, to appear in Computational Intelligence a special Issue on Business Agents and the Semantic Web, (2004).
32. K. McCabe, S. Rassenti, and V. Smith, Institutional Design for Electronic Trading, Conference on Global Equity Markets, N.Y. University, Salomon Center, (1992).

33. R.P.A McAfee, “dominant strategy double auction”, Journal of Economic Theory, 56(2):434–450, (1992).
34. D. McFadzean, D. Stewart, and L. Tesfatsion, A computational laboratory for evolutionary trade networks, IEEE Transactions on Evolutionary Computation, 5 (2001) 546-560.
35. Microsoft Corporation, Internet Commerce, from http://www.microsoft.com. (1998).
36. T. D. Nguyen and N. Jennings, "Coordinating multiple concurrent negotiations," presented at Third International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS'04), New York,USA, 2004.
37. N. Nisan, Bidding and allocations in combinatorial auctions, In ACM Conference on Electronic Commerce, (2000).
38. M. J. Osborne and A. Rubinstein, A Course in Game Theory: The MIT Press, 1994.
39. Paperexchange Marketplace. (n.d). From http://www.paperexchange.com
40. D. Parkes, iBundle: An Efficient Ascending Price Bundle Auction, In Proceedings of the first International Conference on Electronic Commerce, (ACM Press 1999) 148-157.
41. H. Van Dyke Parunak, Visualizing Agent Conversations: Using Enhanced Dooley Graphs for Agent Design and Analysis, Proceedings of the Second International Conference on Multiagent Systems, AAAI Press, Menlo Park, Calif., (1996) 275-282.
42. B. Rachlevsky-Reich, and I. Ben-Shaul, et al., GEM: A Global Electronic Market System, Information Systems Journal, Special Issue on Electronic Commerce, 24 (6) (1999).
43. T. Sandholm and V. Lesser, "Issues in Automated Negotiation and Electronic Commerce: Extending the Contract Net Framework," presented at First International Conference on Multi-Agent Systems (ICMAS'95), San Francisco, CA, USA, 1995.
44. M.P. Singh, Synthesizing Coordination Requirements for Heterogeneous Autonomous Agents, Journal of Autonomous Agents and Multi-Agent Systems, 3 (2) (2000) 107-132.
45. A. Smirnov, and C. Chandra, Ontology-based Knowledge Management for Cooperative Supply Chain Configuration, Proceedings of AAAI Spring Symposium Bringing Knowledge to Business Processes, Stanford, California, (AAAI Press, March 20-22, 2000).
46. L. Tesfatsion, (Ed.) Special Double Issue on Agent-Based Computational Economics, Journal of Economic Dynamics and Control, 25 (3-4) (2001).
47. P. Timmers, Electronic Commerce: Strategies and Models for Business-to-Business Trading, ., (Wiley & Sons Ltd., 1999).
48. M. Tsvetovatyy, M. Gini, B. Mobasher, Z. Wieckowski, MAGMA: An Agent-Based Virtual Market for Electronic Commerce, Journal of Applied Artificial Intelligence, special issue on Intelligent Agents, 11 (6) (September1997) 501-523.
49. H. R. Varian, Mechanism Design For Computerized Agents, The First USENIX Workshop on Electronic Commerce, New York, 11 (19) (July 1995) 13-21.
50. VerticalNet® Marketplaces. (n.d). From http://www.VerticalNet.com
51. V. Vickrey, Counter Speculation, Auctions, and Competitive Sealed Tenders, Journal of Finance, 16 (1961) 8-37.
52. G. Wiederhold, Mediators in the Architecture of Future Information Systems, IEEE Computer, 25 (3) (March 1992) 38–49.
53. R.B. Wilson, “On equilibria of bid-ask markets”, in Arrow and the ascent of modern economic theory, New York Univ. Press, New York 375–414. (1987).