AN APPROACH TO REDUCING E-COMMERCE COORDINATION COSTS IN EVOLVING MARKETS USING A SEMANTIC ROUTING PROTOCOL

Vijay K. Vaishnavi
Department of Computer Information Systems
J. Mack Robinson College of Business
Georgia State University
vvaishna@gsu.edu

William L. Kuechler, Jr.
Accounting and Information Systems
College of Business
University of Nevada at Reno
kuechler@unr.edu

ABSTRACT

Attending the rapid growth of B2B e-commerce is an equally rapid fragmentation of what was once—and still is, in many quarters—conceived of as a global marketplace. Indeed, the most successful examples of B2B e-commerce have taken place in niche markets with specialized vocabularies and processes. Relative to the ideal of a universally accessible global marketplace, specialized markets represent significant coordination inefficiencies. We propose that electronic markets are open systems in the general systems theory sense of the term, and that any electronic commerce architecture must deal with open system semantics to avoid progressive segmentation into isolated sub-markets. Reducing buyer search costs represents a special but important case of reducing coordination costs to improve market efficiency and decrease the pressure for fragmentation. We analyze the components of buyer search cost to identify core issues that must be handled by any electronic commerce architecture intended for evolving markets. We propose XML record types arranged in an autonomously defined type hierarchy and a semantic routing protocol as potential mechanisms for reducing message processing and buyer search costs in this open system environment. We suggest the method is potentially generalizable for coping with other problems introduced by Web marketplaces with evolving semantics.

Keywords: E-Commerce, Coordination Costs, Evolving Markets, Semantic Routing Protocol

1. Introduction

The rapid growth of the Internet and its ability to support commercial transactions has sustained intense interest in electronic commerce. Malone, Yates, and Benjamin (1987) have provided a conceptual argument that, in part, explains electronic commerce's growth: information technology tends to reduce coordination costs and thus increases the use of markets in preference to hierarchies. While Web-based marketplaces have the potential to reduce coordination costs in many areas (Malone and Crowston, 1994), experience with EDI shows that unless the issues of semantic heterogeneity and evolving semantics are directly addressed in transaction protocols, market dis-coordination through fragmentation and specialization is likely. Indeed, much of the actual benefit from B2B trading to emerge from the early promise has been in highly specialized markets such as RosettaNet (http://www.rosettanet.com/) for the semiconductor industry (Krazit, 2002). Even within specialized industry e-markets, known as *B2B verticals* (Chang, 2000), fragmentation is common. However, increasingly such fragmentation is recognized as inefficient. Currently the trade press for industries as diverse as food, industrial chemicals and securities describe initiatives for e-market consolidation (Kelly, 2001; Chang, 2000; Tully, 2000). In the chemical industry B4B (business-for-business) consortiums are consolidating multiple B2B marketplaces in search of full supply chain integration.

Reducing buyer search costs represents an important special case of the general problem of reducing market fragmentation. Searching for products (suppliers) represents the first in the series of transactions between buyer and seller, and the ability to locate a wide range of potential suppliers for a good greatly reduces the frustration with e-markets that inhibits their formation and results in their fragmentation (Spence, 2000). Bakos (1997, 2001) gives an economic analysis of this issue and the efficiencies and cost savings accruing to improved buyer search explains the

drive in many areas to consolidate e-markets. Bichler and Segev (1998) likewise indicate that reducing search costs and providing more complete information about a product or service is a key issue in overcoming current electronic commerce market limitations.

1.1 Prior Related Research

1.1.1 Agent Based E-trading Systems

A number of general architectures for supporting more efficient electronic commerce have been proposed by researchers. Ciancarini, Tolksdorf, Vitali, Rossi, and Knoche (1998) developed PageSpace, a reference architecture that enables the development of multi-agent applications while minimizing space and time coupling among the various agents comprising the application. Andreoli, Pacull, and Perschi (1997) created XPect, a framework for conducting electronic commerce based on Linda-like concepts of dynamically placing and retrieving tuples in a shared space. Tsvetovatyy, Gini, Mobasher, and Wieckowski (1997) have developed MAGMA, an agent-based virtual market designed to enable simulations of actual markets. Each of these research architectures sought to alleviate a different aspect of coordination inefficiency by providing more effective communication between market agents, by allowing inexpensive multiple agent interactions, and so on.

As research in agent-based trading systems progressed, the stress shifted from computing and communication efficiencies toward more capable agent interactions. Karacapilidis and Moraitis (2001) describe an agent-based e-commerce system featuring buyer and seller agents with sophisticated decision and negotiation features. Further, the agents are persistent and so are capable of ongoing learning of both features of the marketplace and of their clients' preferences. However, though the agent's decision algorithm is potentially capable of responding to new product attributes, the system presumes a marketplace with a known architecture and low-level communications protocol and discovery mechanisms. Methods for spanning marketplaces or for dealing with unique new products are not fully developed.

Each of the research architectures has highlighted problems that must be addressed in the evolution of electronic commerce, however, none of these architectures has been implemented on a broad scale.

In contrast to the research architectures, industry proposals for evolving the WWW for e-commerce have been more pragmatic and based on existing Internet protocols and infrastructure. Some of the multiple industry initiatives proposed in pursuit of ever more frictionless e-commerce are currently in wide usage and others are quickly gathering support. One of the most advanced of these is Universal Description, Discovery and Integration (UDDI), an Internet-based service discovery technology (OASIS, 2003). UDDI allows e-market participants to globally publish their offerings and discover what others have offered. However, a discovery mechanism alone is insufficient for e-commerce. Also required are a widely understood service description language, such as ebXML (http://www.oasis.org; Kotok and Webber, 2002) or xCBL (http://www.xcbl.org/) and an ontology or classification scheme, such as the *Universal Standard Products and Services Classification* (UNSPSC: http://www.eccma.org/unspsc/).

However, all of the technologies and initiatives described above assume a semantically homogeneous environment where the mutual understanding of agent-to-agent communication is not an issue. In the next section of the paper we present in more detail the position that electronic markets represent *open systems* in the *general systems theory* sense of the term (von Bertalanffy, 1968): they are embedded in the larger environment of global trade from which they continuously take information and to which they must continuously adapt. Thus the message formats and content of electronic marketplaces will evolve continuously, and robustly handling this change is essential to creating a real-world electronic commerce architecture that minimizes market fragmentation. A relatively smaller number of researchers have proposed architectures that directly address some of the issues inherent in semantic heterogeneity.

Eco System (Tenenbaum, Chowdhry, and Hughes, 1997) explicitly recognizes the need for several independent systems to interoperate (i.e., heterogeneous semantics). However, Eco System's approach to the problem is, in effect, to have a central hub that can translate between systems, a technique that rationally links systems in which the semantics differ at the outset of intersystem communication. The issue of systems that drift apart semantically as they continue to participate in the marketplace is not directly addressed. Our approach differs from Eco System in that it does not require a centrally defined translator, though it is capable of supporting one, yet does enable semantically drifting systems to be more robustly linked.

Even UDDI, the most ambitious service discovery protocol to date implicitly assumes homogenous semantics: the architectural section of UDDI known as the "Yellow Pages" contains the classification of companies and products, described using various industry syntaxes such as xCBL, according to cross-industry ontologies (product classifications) such as UNSPSC. However those ontologies evolve far more slowly than the markets they describe, and are incompatible (Fensel et al., 2001). In fact, an active stream of e-commerce research focuses on how to integrate these various ontologies (Fensel et al., 2001). Much of this research is conducted under the general heading

of *Web semantics* in the wake of Tim Berners-Lee's popularization of a future WWW in which all accessible resources are semantically annotated and linked (Hendler, Berners-Lee, and Lassia, 2001). In the next subsection we position the semantic routing protocol contribution within this broad area.

1.1.2 Semantic-Web Developments

Bussler, Fensel, and Maedche (2002) propose a general, very high level architecture for Semantic-Web-enabled Web services which enacts the services *after* buyers and sellers have linked; our semantic routing protocol enables buyer and seller to discover each other at minimal cost and is compatible with and orthogonal to this research.

Schlosser, Sintek, Decker, and Nejdl (2002), and Verma et al. (in press) both propose peer-to-peer (P2P) hardware architectures for semantic Web services delivery. Both suggest P2P is highly desirable for its scalability and both integrate semantics directly into their architectures, however they differ radically in their worldview. Schlosser et al. actually construct their P2P network (physically link it) according to ontological constraints, assuming a universal ontology. This arrangement greatly reduces network traffic, if a universal ontology can be found and if it never changes. The value of our protocol is reduced under the conditions Schlosser et al. envision. However, Verma et al. take a position more compatible with ours: that multiple, constantly evolving ontologies will need be linked under the semantic Web, and they accomplish this by proposing an evolvable hierarchy of ontology servers. Our semantic protocol is conceptually and practically compatible with this approach.

Leukel, Schmitz, and Dorloff (2002) present preliminary research on a master translation mechanism between vertical market electronic product catalogs using XML, the lingua franca of the semantic Web. This research focuses on integrating semantic elements from different organizations that differ initially, but are constant over time. Our research complements this work by proposing a mechanism for handling semantics that vary over time, whether or not they were initially coherent.

Guha, McCool, and Miller (2003) propose a more general approach in which future search engines operating on the semantic Web can make sophisticated inferences about *anything* on the semantic Web, and specifically for our consideration, product and service descriptions. However a global implementation of the semantic Web is many years away while our proposed system works with existing technologies. Further, though the search mechanisms described in semantic Web research are quite general, they are manually initiated, and not intended to eliminate market friction through the automatic routing to interested parties of service requests, as our system is.

1.1.3 Goal-Directed E-Trading Systems

Several other proposed technologies approach the linking of evolving systems by the general technique of using the intentionality (the goals, and intentions) of trading partners to serve as a higher level invariant that guides B2B transactions even as transaction details evolve. In the Spheres of Commitment approach (Jain, Aparicio, and Singh, 1999), intelligent trading agents use the goals of a successful business transaction (to enable a mutually profitable trade, to have product delivered on time, etc.) to direct transactions and correct for variant semantics between trading partners. In the workflow derived HOPI (hierarchical overlay of process intention; Kuechler, Vaishnavi, and Kuechler, 2001) the steps of a trading process are linked with the goal hierarchy motivating the process at the start of a transaction. If the details of the process change, the goals are assumed stable, and used to correct for process misalignment.

1.2 Research Scope

The architecture presented in this paper is in many ways an intermediate approach to the research and commercial systems discussed above. It is less susceptible to market fragmentation than architectures assuming fixed semantics (closed systems) but sacrifices some of the advanced processing capability of the intentional approaches in return for a pragmatic base in existing technologies. It is slightly less flexible than the most sophisticated "semantic Web enabled" approaches, but it can be implemented now, with current technologies. Finally, though all of the architectures, protocols or initiatives described above focus on the transactions (services) that occur *after* buyer and seller have discovered each other our approach specifically targets an area of considerable friction in e-commerce, buyer search costs, and presents a pro-active protocol for linking buyer and seller.

In summary then, the objective of this research is to determine a B2B message exchange protocol that substantially reduces search costs and maximizes vendor response, while operating flexibly and robustly in an open system environment, and to explore its implementation feasibility by utilizing predominantly proven, existing techniques.

1.3 Structure of the Paper

The remainder of the paper is organized as follows: Section 2 first defines the open systems environment in which B2B transactions take place and the constraints inhering in such an environment. Then, using an example, the specific problem of search cost in a semantically evolving environment is analyzed. Section 3 develops the key concepts of our approach, structuring messages as a concept hierarchy, and presents an overview of a specific architecture that incorporates the approach. Section 4 traces a transaction path through the architecture as it responds

to the problematic example of Section 3, develops a proof-of-concept performance analysis of the architecture, and summarizes the limitations and strengths of the general approach. Section 5 suggests directions for future research.

2. Problem Definition

2.1 Internet Markets as Open Systems: General Constraints

Less than 10 years ago the term *open systems* was universally understood as a *general systems theory* concept (von Bertalanffy, 1968) which meant an informationally open system, a system embedded in a larger environment that took in information from and continually adapted to that environment. Today the term has been largely co-opted by computer science communities and is more widely understood as a system designed to *public* standards or constructed from *public* source code, or both. In this paper, we use the term in its original sense. There is no logical relation between open-source systems and informationally open systems. A system based on public source code can assume fixed semantics and thus be a closed system in the general systems theory sense. The developing marketplaces of the WWW are open systems of the informationally open type; they are continuously evolving, and are driven by forces in the larger, global economy in which they are imbedded. An intuitive sense of the strong drive toward semantic drift can be gained from recalling that the basis of modern marketing is to *create distinctions* between product offerings. Complementing this is the universal goal of management to lead organizations to positions of strategic advantage—to become meaningfully different from the competition.

Theoretical and empirical support for e-markets-as-open-systems is based on the fact that offices have been empirically demonstrated to be open systems for which it is impossible to anticipate all contingencies a priori (Hewitt, 1986); instead, issues such as preferred modes of information exchange have been observed to be the result of ongoing, decentralized negotiations (Gerson and Star, 1986). Since business-to-business (B2B) marketplaces are essentially a forum for transactions between offices—open, evolving systems—a marketplace is itself open and evolving (Gasser, 1991).

To introduce our understanding of the information systems implications of *open systems*, we choose a prescient quotation from one of the longest standing proponents of such systems, Carl Hewitt (1991; 1986). In this passage, Hewitt describes the types of computer systems that must be created to deal with an open environment:

> Computer applications will be based on communication between subsystems that will have been developed separately and independently. Some of the reasons for independent development are: competition, economics, geographical distribution, and diverse goals and responsibilities. We must deal with all the problems that arise from *conceptual* disparities. Subsystems will be open-ended and incremental—undergoing continual evolution. There are no global objects … (Hewitt, 1986).

The change in business information process thinking that accompanies the understanding of organizations as open systems is well illustrated by a quotation from Tom Davenport writing in the *Harvard Business Review*:

> Many planners still assume that organizations have a core of invariant pieces of information—such as customers, products, and business transactions—around which key systems can be developed … Information evolves in many directions, taking on multiple meanings. While IT specialists are drawn to common definitions of terms like *customer* or *product*, most information doesn't conform to such strict boundaries. Forcing employees to come to one common definition, as some technologies require, only truncates the very conversations and sharing of perspectives that the technology is supposed to ensure (Davenport, 1994).

In the same paper, Davenport suggests contra-traditional managerial heuristics for business information systems; two of these are especially salient in our context: (1) assume transience of solutions, (2) assume multiple meanings of terms. While the concept of and even the term *semantic drift* originated in empirical observations of workgroups in organizations (Hirschheim, 1986), recently the computer science community has begun to recognize that an increasing number of software systems are informationally open and to seek techniques for designing and linking systems that have no universal standards (Open Systems CFP, 2003).

2.2 Learning from EDI: Why Rigidly Defined Systems Fragment

Fully automated supply chain management is a vision that first seemed achievable with the advent of EDI, a decade prior to Web-based commerce. However, the failure of EDI to fully live up to its promise is well documented (Riggins and Mukhopadhyay, 1999; Krcmar, Bjørn-Anderson, and O'Callaghan, 1995) and the problems that we foresee for non-evolving Web-based B2B marketplaces are identical to those that occurred with EDI. EDI attempted to define business transaction types in a way that was syntactically flexible, but assumed fixed semantics. This approach had two notable results: (1) the hoped for *universal* trading standard quickly fragmented into multiple highly specialized standards for specific industries; and (2) system costs deterred small and medium enterprises (SMEs) from the use of EDI unless coerced by large, dominant trading partners. Damsgaard and Truex (2000) use insights from linguistics to explain EDI's problems: EDI is a *language* between human activity systems, and as such

is inevitably contextually interpreted (has variant semantics). The attempt to impose standards on a human behavior is thus seen to be fundamentally different from imposing a technical standard for a computer circuit board, which *always* has a rigidly defined environment. Although multiple specific markets have already been proposed for e-commerce in implicit recognition of variant terminology across industries, this solution is less satisfactory and efficient than a truly global market; an organization must implement varying systems to process transactions for each market, and this will discourage SME participation, just as it has for EDI. Moreover, as Damsgaard and Truex illustrate with the case of the Hong Kong shipping industry, even within a niche market there is significant variation in the *interpretation* of "standard" documents by different organizations.

Thus, to pursue a truly global e-marketplace, or even to keep a specialty market from further fragmenting, a system must incorporate the means to accommodate terminological drift, and it should ideally be normalized within the trading architecture. That is, even though *fixed universal* definitions are oxymoronic, a standard universal mechanism *can* be set in place for coping with evolving definitions.

2.3 Analysis of Buyer Search Cost Within an Open Marketplace

In the analysis that follows, a specific area of friction in e-commerce, buyer search cost, is analyzed with consideration of general open-system principles and lessons learned from a historical examination of EDI. From the perspective of a buyer participating in electronic commerce, three key factors comprise the cost of any transaction: (1) search cost (to locate the product/vendor), (2) the price of the product itself, and (3) delivery/acquisition cost (Gupta, Stahl and Whinston, 1997). An ideal system would eliminate search costs and any comprehensive electronic commerce architecture must deal with these issues.

Obviously, for an electronic commerce transaction to take place, messages must be exchanged between the buyer and potential vendors (or their representatives). The process of searching for a product can be subdivided into six steps: (1) formulating the request, (2) transmitting the request, (3) handling the request (done by all potential vendors), (4) formulating the reply, (5) transmitting the reply, and (6) considering the vendors' replies. Note that steps 2 and 5 are "message transmission costs" and that steps 1, 3, 4 and 6 can be considered "message processing costs."

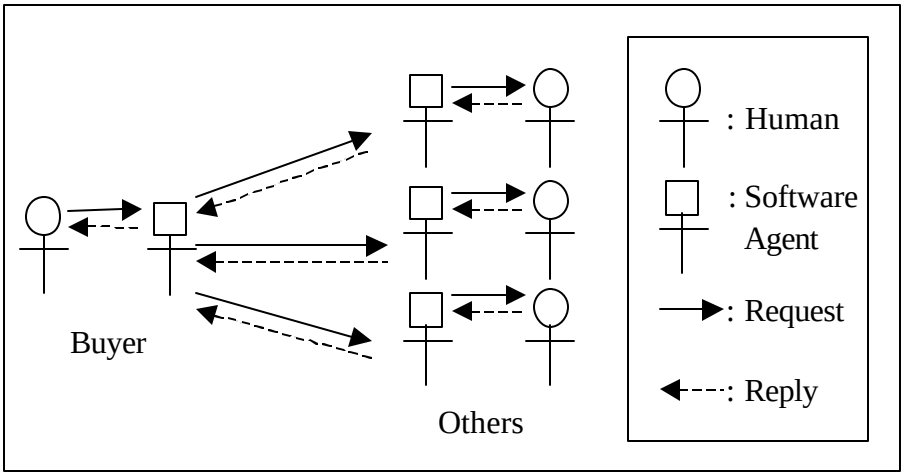

Figure 1. Mediated Transactions

Figure 1 portrays a hypothetical situation in which a buyer has transmitted a request to all "others"; each individual is interacting with his or her respective agent. The term "others" is used instead of "vendors", since the buyer will not, a priori, know the potentially qualified vendors; in effect, this diagram represents a broadcast of the buyer's request to every other entity with whom the buyer can communicate. If the message transmission costs are zero and the message processing costs are zero, then implementing an optimal market becomes trivial: every message is sent to every potential recipient who considers it and replies as appropriate. The Internet has reduced the cost of sending messages to near zero but the cost of a person processing a message is *not* zero. However, in Figure 1, it is the (software) *agents* that exchange messages. Thus, minimizing message processing costs is primarily an issue of minimizing the processing costs of agents, not humans, and this significantly increases the degrees of freedom available for addressing the problem.

In searching for a product, the buyer must describe the desired product to his or her agent. The agent communicates this request to other agents and this request must be in a form that other agents can interpret to make a decision on handling. The vendor agents then (potentially) interact with their human counterparts and formulate a

reply. This reply is sent back to the buyer's agent, and must be in a form that the buyer's agent can interpret. This overall process can thus be separated into two broad issues: (1) human-to-agent communication (the interface) and (2) agent-to-agent communication. The focus of the research reported in this paper is on the latter.

Each agent's message processing can be subdivided into two major components: filtering and handling. Filtering refers to the determination of whether a message is relevant or not, i.e., determining if it is worth handling. Handling refers to the process of appropriately processing a relevant message. For example, consider a request for a particular book. Vendors who sell books would want to receive that message and handle it, but vendors that do not sell books would want to filter that message, i.e., they would not even wish to consider it. The precise handling of any message will be vendor-specific; separate databases will be searched for inventory and cost quotes, the handling of out-of-stock items will be different, etc. While message filtering could be handled by having every agent receive every message, each agent would then have to process a high volume of messages, most of which would be discarded. Ideally, a system could be constructed that would route messages to an agent only if that agent will consider them relevant.

The preceding discussion gives rise to two related but distinct research questions that must be addressed by any general electronic commerce architecture: How can an agent be certain that it can correctly process a message it receives while operating in an open system environment? How can the message filtering process be handled efficiently, also under conditions of constant market evolution? The next section provides an overview of solutions to both of these issues.

3. Proposed Approach Using a Semantic Routing Protocol

3.1 An Overview of Current Technologies

Our solution to some of the open-systems issues we have raised for e-business is purposely grounded in widely supported e-business technology initiatives, which utilize predominantly proven techniques. We next describe these initiatives at a high level and indicate their shortcomings, as currently proposed, in supporting markets with evolving semantics.

Universal Description, Discovery and Integration (UDDI) is a vendor-driven service discovery technology, which provides, in essence, a globally accessible database of goods and services available in the marketplace: *what* goods are available from *whom*, and *how* purchase, payment, delivery, etc., can be transacted. At a block-diagram level, UDDI uses: XML service descriptions, a repository for storing these descriptions, and standard methods for publishing services on the repository and discovering services that have been published on the repository. It has been likened to a phone directory, with white pages for discovering *who* (vendors), yellow pages for discovering *what* (products and services) and green pages for describing *how* vendor and buyer can link and conduct business. To maximize flexibility, UDDI deliberately does not specify types of business transactions (bid, purchase, ship, etc.) or formats for such service descriptions. This is the function of business language initiatives such as ebXML, cXML or the semiconductor industry-specific RosettaNet (http://www.rosettaNet.org). Each of these competing service description languages parses the overall business transaction process into different sub-transactions, and UDDI is designed to support any or all of them. A still lower level of description, vital to conducting any sort of business, is a product/service classification scheme, or ontology, such as UNSPSC, which describes the product or service sought and purchased on the e-market. Most of the business language initiatives support multiple ontologies.

UDDI, together with a business service language and an ontology are frequently considered to have solved the problem of universal B2B communication on the Web. Unfortunately, they do not provide the mechanisms to handle ongoing, changing business requirements. The problem is readily seen to be with the rigidly defined service description languages and classification ontologies. These function only as long as it takes for new products, services or vendor-buyer relationships to emerge, at which time market efficiency begins to degrade. Without an inherent evolution mechanism, we suggest these fixed standards will eventually encounter the same market fragmentation problems as the fixed EDI standard. Some business process description languages do explicitly provide the means to define new transaction types. These are useful for one-to-one buyer-seller linkages, but do not provide a means for the entire market to share in the new definitions, a necessity for allowing e-marketplaces to follow evolving market trends.

3.2 Solution Approach: Extending Current Technology for Open Systems

Having discussed the technological infrastructure for our proposed architecture, we return now to a discussion of our example problem: minimizing search costs in evolving Web marketplaces. To enable an agent to be certain that it can correctly process a message it receives, the messages can be encoded using XML (Gartner Group, 2001, 1998; Bosak, 1997), and each message identified by a unique type. The message types would be uniquely specified using a globally unique identifier (GUID). A GUID is a 16-byte identifier that is guaranteed to be unique *without* requiring a central authority to issue such an identifier (Rogerson, 1997). This is the same technology commercially

employed by Microsoft's Component Object Model to uniquely identify components and interfaces in their ActiveX architecture, and now part of the ebXML/UDDI standards. If the agent recognizes the message GUID (i.e., type), then the agent can be certain that it can process the message correctly.

However, given the open system assumption, new products and services and market perspectives (interpretations) will be continuously introduced. Unique type identifiers insure that each agent can unambiguously determine if it can process a message or not, but do not handle new message types. The basic concept used by our architecture for handling partially understood messages, which ensures graceful degradation of the system, dates from the dawn of AI—the use of a *concept hierarchy* (Sacerdoti, 1974; Tenenberg, 1986). From its inception, object oriented programming has promoted the use of abstract classes—base concepts—which are progressively specialized while each specialization retains the attributes of the higher level concept (Kamin and Reddy, 1994). The approach was later developed specifically for communication between autonomous office environments, where semantic drift is a known problem, by Lee and Malone (1990). That technique, *Partially Shared Views*, was demonstrated to enable automated message processing *without* arbitrarily constraining the system to fixed or pre-specified message meanings. This approach dictates that each message has a type, and that the message types form a type hierarchy. New messages are introduced within that type hierarchy, but no central authority is required to manage the type hierarchy. Each message need only identify its supertype(s). From the viewpoint of an agent any message will fall into one of three classes: fully processible, partially processible, or not processible. If the message is of a known type, then it is fully processible. If the message is not of a known type, then the agent would look at the supertypes and determine if any of those are known message types; if so, the message would be partially processible (as the known supertype). Finally, it is possible that the message and all of its supertypes are unknown; in this case, the message would not be processible. However, this hierarchical arrangement enables new message types to be introduced *and processed*, even though they are not fully understood; thus new message types can be introduced and the system will degrade gracefully.

The second major issue in minimizing discovery costs is efficiently handling the message filtering process. As noted above, every message could be sent to every agent, but such an approach is clearly unworkable. Suppose, however, that "somehow" the Internet knew every entity that was interested in receiving a message of a particular type; in this case, the system would operate at optimal efficiency since the message would only be routed to agents interested in receiving it. A plausible approach to this ideal lies in viewing this as a message routing problem, analogous to routing problems already solved in TCP/IP.

To understand this, consider a simplified view of the process of sending an email message—all that is required is the recipient's email address. This is a "human understandable" address. This address (or, more accurately, the domain name) is converted to a TCP/IP address. This TCP/IP address is a logical address. This logical address is then converted to a physical address (typically the address of a network card), which specifies the exact computer that queues the email message. The conversion from logical to physical address is accomplished by an Internet service known as DNS (domain name service) that is implemented by a set of Internet server hosts. The actual handling of the mail messages is directed by a mail protocol. Thus, one conceptualization of a solution to the filtering problem, analogous to existing Internet services, is a *semantic routing protocol* layer, located above the standard layers, that routes messages based on content rather than specific addresses. Servers that understand and interpret this semantic protocol will provide the routing services. A possible implementation approach is the publish/subscribe protocol defined by UDDI. The UDDI implementation proposes globally accessible servers on which vendors can *publish* ebXML-formatted definitions of their services. Potential buyers can scan the servers for these services and then *subscribe* to be automatically notified of any changes to the service. In the remainder of this paper we define the basic structure of the semantic routing protocol and outline an implementation architecture.

3.3 Overview of Protocol Implementation

This section develops the book request exampled mentioned in Section 2.3 to illustrate the protocol in operation: Internet enabled buying and selling of books. The example provides concrete illustrations of concepts presented in the previous subsection and is used throughout the remainder of the paper.

Figure 2 presents a high level diagram of the proposed implementation of the semantic routing protocol. Continuing with the book request example, the buyer's agent submits the book request to the appropriate XML record server. The XML record server is aware of the agents that have outstanding queries for that type—in this case vendors A and C. Note that the message is *not* routed to vendor B since this vendor has not registered a query for messages of this type. Theoretically the transfer of the request to all interested vendors can be accomplished with a total of three message transfers: one message transfer from the buyer to the appropriate XML record server, and two message transfers from the XML record server to the two interested vendors.

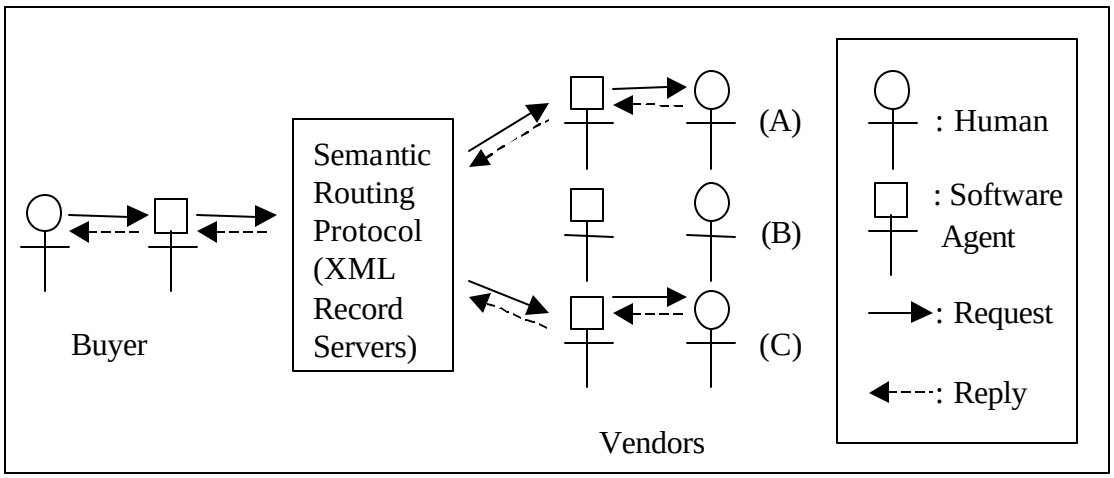

Figure 2. Semantic Routing Protocol Implementation

Figure 3 presents an example of an autonomously defined type hierarchy. Each XML message type is represented as a three-section box. The first section contains a natural language description of the XML record type. The next section contains the GUID; a GUID is actually a meaningless but unique 16-byte identifier, but for ease of description it has been presented as a leading letter that identifies the vendor that created the message type and a month-day-year field indicating when it was created. The third section contains the actual field names. Lines connecting boxes indicate inheritance. For example, BookRequest2 contains all of the information in BookRequest plus the unique field Keywords. The right panel shows the XML for BookRequest2, which consists of the semantic routing protocol header (SRPHeader with GUID and supertype information), and various message-specific fields.

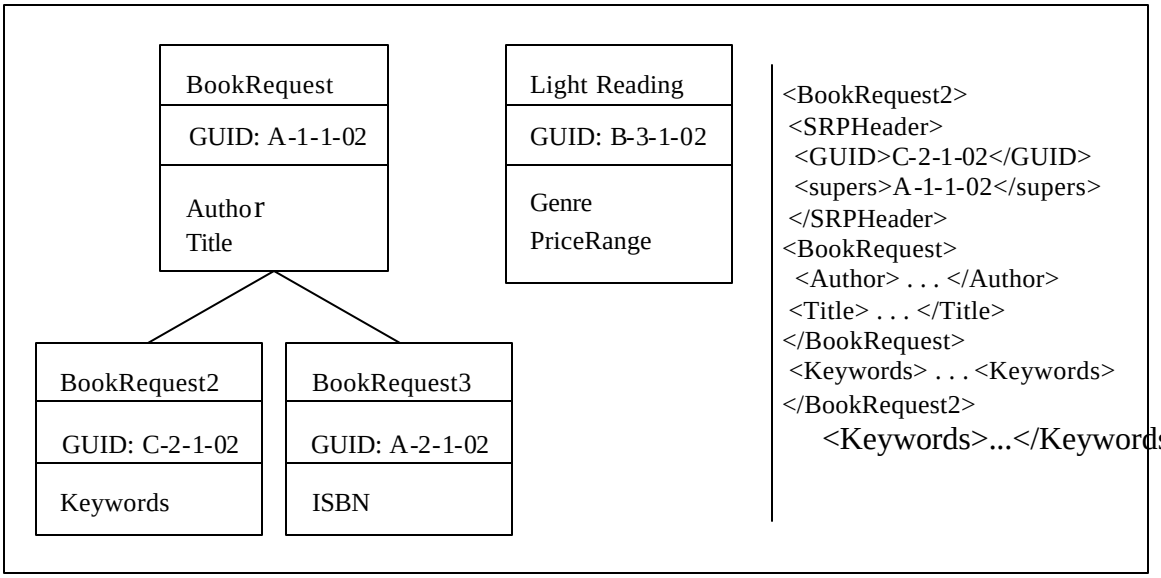

Figure 3. Autonomous Type Hierarchy Example (with XML for BookRequest2)

The following example illustrates the autonomous creation of this type hierarchy. The first type, BookRequest, was created by vendor A on 1/1/2002. Through environmental scanning, vendor C became aware of BookRequest, and also began querying for it (this is the situation shown in Figure 2, where the request is sent to vendors A and C). Both vendors A and C saw limitations of this format, but perceived *different* limitations and solutions. Vendor C determined that keywords would help to clarify ambiguous requests; vendor A determined that the ISBN would be helpful for customers who knew the exact book they wanted. Neither was aware of the other's developments, and they simultaneously introduced their own enhancements on 2/1/2002. The fragmentation of search terms for even the most common objects is an empirically demonstrated example of semantic drift in an open environment (Furnas,

Landauer, Gomez, and Dumais, 1987). Since both new types are subtypes of BookRequest, customers can use the new types for requests and still reach vendors that can process the original BookRequest. In contrast to vendors A and C, vendor B chose a differentiation strategy and introduced a completely new type, "Light Reading," on 3/1/2002, to address customers who enjoy reading and want certain types of books, but are not looking for a particular book. Note that vendor B can also register to receive the book requests created by vendors A and C (just as vendors A and C can register a query for "Light Reading"). If the "Light Reading" new message type becomes widely used, vendors A and/or C may introduce their own enhanced types that incorporate this extended specification capability. A base ontology (a base set of message types) such as the Common Business Language (CBL) proposed by Glushko, Tenenbaum, and Meltzer (1999), or the initiatives based on CBL such as Rosetta Net or ebXML, can be easily used in this architecture, but is not required.

3.4 The Protocol in Use: A Tale of Two Vendors

To enable a more concrete understanding of benefits and limitations of the semantic routing protocol, and its relationship to message discovery, the following discussion extends the bookseller example to give a walk-through of the use of the system by two different vendors. The vendors characterize the two extremes of innovation adoption, early and late adopters (Moore, 1991). First however, we point out that the system supports both new product introductions and unique selling proposition (USP) tactics, just as the non-electronic market does. In fact, the ability of any vendor to introduce new products or append attributes to existing products without authorization from a centralized authority while maintaining the ability to (at least partially) process the new introductions intelligently using existing infrastructure, is a unique feature of the system. Consider first vendor B, an innovator and early adopter. B scans the marketplace constantly, using the Message Discovery Service (MDS) of Figure 4 as simply another marketing tool. Let us assume the MDS is implemented with one of the "find me something similar to this" search engines such as the currently-in-use *Teoma* search engine (Teoma, 2003). Vendor B, as a seller of popular books, is constantly on the lookout for new ways of describing and differentiating books (different messages). When he finds them, he categorizes his existing inventory according to their specifications—using keywords for example—and uses the semantic routing protocol to distribute information to any potential buyer who has posted a request for books using the new keyword book specification. B is also an innovator. As described above, he has introduced the new book classification: *light reading*. Just as in the current marketplace, buyers hungry for innovations in personalized product delivery will be scanning the MDS with *their* search engines. If the new classification strikes a responsive element in the market then buyers will begin to post requests for books using this descriptor and B will gain a temporary market advantage by being the first to use that descriptor. Finally, B is willing to process messages he only partially understands and has programmed his electronic agents to receive such messages. This requires manual intervention on the part of B's purchasing staff, however B considers this an integral part of his marketplace scanning process—an automatic advisory of new product differentiation tactics!

Now consider vendor A, a conservative, late adopter of innovations. Let us assume A is an institutional textbook seller in a sub-market that moves far more slowly than B's recreational reading market. A goes to the textbook sellers' convention once a year, and there learns of any new messages used to electronically describe textbooks. He makes a judgment as to whether or not a significant portion of his market will use the new message type, and if so, he will have his inventory described with the new message type and post its availability on the MDS. Experience has shown that an insignificant portion of his business depends on partially understood messages, and so he has programmed his electronic agents to reject any requests not fully understood. Note that in general, the proposed protocol and associated services offer the benefits of e-commerce while constraining vendor behavior very little from current, manual market behaviors. It was precisely this lack of artificial constraint that was a primary design goal in order to keep e-markets from fragmenting. In the following section we describe in detail *how* the high level functions just described can be implemented.

4. Protocol Implementation Details

4.1 Service Events

In our initial approach, a single server provides services to all customers and all vendors. There are four externally generated events that can take place:

A vendor can register a query to receive specific message type(s)

A buyer can send a specific message type

A vendor can cancel a registered query

A buyer (or vendor) can inquire about the message types that are available

4.1.1 Registering a Query

The vendor sends a query to the server containing a specific GUID identifying the buyer messages that the vendor wants to receive. The server adds this vendor's IP address to a list of addresses associated with this GUID.

To help minimize Internet traffic, a group of vendors might establish a multicasting (Furht, Westwater, and Ice, 1998) address to receive messages. The vendor list associated with a message type will be stored in IP order so that duplicate addresses can be found, and only a single message will be sent to any particular address.

A vendor will post a request for the most general type of message that it wants to receive in the message type hierarchy. It will then automatically receive all messages of that type and all specializations.

4.1.2 Sending a Message

A buyer using the semantic routing protocol sends a message to the server containing an instantiated message specifying the goods or services that it wishes to purchase, i.e., a message with the specification fields filled with data. In the bookseller example illustrated in Figure 3, an instantiated message might contain the data: Genre=gothic; PriceRange = < 20. It is important to note that every message contains not only the GUID designating its own type but also the GUIDs of all supertypes. For each GUID in this list, the server determines the list of vendors interested in receiving messages of that type, and sends the message to those vendors.

4.1.3 Canceling a Registered Query

The vendor sends a request to the server with the GUID identifying the message type that it no longer wishes to receive. This could occur if a vendor no longer carries a particular product line, or if a vendor decides that a certain message type is too general (and decides to receive a more specialized message type instead). The server removes the vendor's address from the list of addresses associated with that GUID.

4.1.4 Inquiring about Available Message Types

The routing protocol does not require any knowledge about the message types beyond their globally unique identifiers (GUIDs), since a semantic protocol server simply stores a list of GUIDs for which registered queries have been posted together with the addresses of the vendors interested in receiving that message type. For this paper we assume that a separate system (one or more "message discovery servers") is used to store message types together with natural language descriptions; message descriptions and the corresponding XML record types can then be searched and retrieved through this system. Vendors will post their message types to this system to encourage buyers to use their enhanced message type(s). The exact mechanism by which GUIDs are assigned to messages should be decentralized to achieve the goal of spanning the entire market space, but is otherwise independent of the routing protocol. The decoupling of the message discovery mechanism from the routing protocol is beneficial since each can be considered separately. This separation of message discovery servers from the underlying protocol is analogous to the separation of HTTP (or HTML) as a mechanism to provide Web pages, using various search engines as a mechanism to discover new Web pages.

Message discovery (synonymous in the literature with "product and service discovery") is an active area of research; dozens of mechanisms exist and more are introduced at every conference dealing with Web-based services. UDDI as currently implemented incorporates the data structures (data and meta-data) for message discovery services (OASIS, 2003). Each UDDI user is expected to implement their own discovery routines to exploit the information provided. However UDDI presumes the use of fixed ontologies. Multiple schemes for augmenting UDDI service descriptions with enhanced semantics have been proposed. A recent and interesting UDDI extension maps the WSDL (Web service description language) service descriptions used by UDDI to domain ontologies and then uses AI algorithms to discover services on a "find something similar to" basis (Verma et al. in press). For the remainder of the paper we treat message discovery at a 'black box' level since the semantic routing protocol itself requires only that such a service exist in some form.

Figure 4 illustrates a single server architecture and several important refinements relative to the general architecture presented in Section 3. First, human actors have been eliminated from this figure to draw attention to this paper's focus on assisting agent-to-agent communication. Second, we have explicitly shown the vendor-to-server communication of registering a query so that the server can efficiently handle the routing of buyer messages to interested vendors. Third, while requests travel from buyers through the semantic routing protocol server to vendors, the server (and the protocol) does not directly handle replies from vendors. While further advantages and efficiencies may well be gained by handling return traffic through a more centralized mechanism, this is a separate issue from the current paper's focus on providing a mechanism for enabling more efficient buyer-initiated, agent-assisted communication. Finally, the message discovery server has been separated from the semantic routing protocol server.

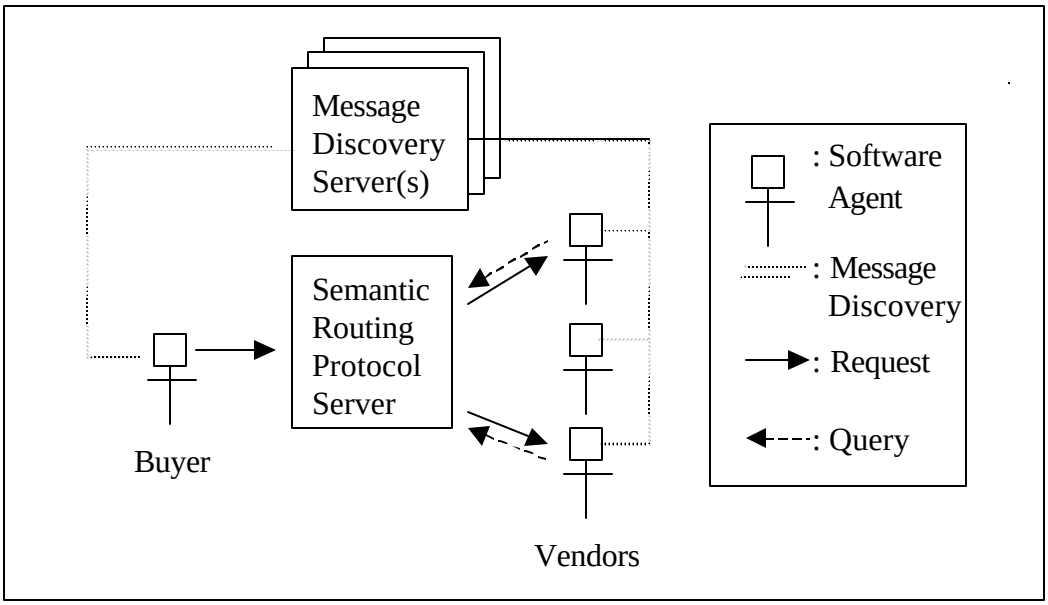

Figure 4. Single Server Semantic Routing Protocol Architecture

4.2 Analysis of the Architecture's Performance

The primary limitation of this approach is that a single machine is responsible for all processing and communication. To formalize the analysis, we use the following metrics to analyze the performance of the architecture:

# of incoming messages (per unit of time)

# of outgoing messages (per unit of time)

storage load (required to store the vendors interested in particular message types as specified by the corresponding message type GUIDs)

Given current technological capabilities, communication bandwidth resources are typically exhausted before computational resources. The algorithms used to store and determine the vendors associated with a particular message type are straightforward, and these algorithms will only be invoked upon receipt of a message from a buyer or a query registration by a vendor. We have therefore based our analysis of the protocol's scalability by analyzing the number of messages that a server is expected to send and receive, arguing that if a server can reasonably handle the communication load, it can also handle the processing load associated with that level of communication. We have also specifically included the "storage load" as a variable because the message type hierarchy and number of vendors asking to receive the message types may become quite large. An ideally scalable solution will be able to distribute the storage of this information across multiple servers.

Of the four events that can occur in the architecture, inquiries for message types can be handled by a separate system (the message discovery servers), and we place these inquiries outside the scope of our analysis. We also argue that a vendor canceling a registered query is going to be a relatively rare event—certainly more rare than a vendor registering a query. Therefore, in the interests of keeping our analysis simple, we ignore this event for analysis. This leaves the protocol system with two major events to handle: a vendor registering a query and a buyer sending a message.

Given these simplifications, the number of incoming messages will be the sum of the number of messages that buyers send plus the number of queries that vendors register. The number of outgoing messages will be a function of the number of messages that buyers send and the number of vendors registered for particular message types. For each message that a buyer sends, the server must receive that message and then relay it to all vendors that have registered a query for that message type; in addition, the server must also relay the message to all vendors that have registered a query for that message's supertypes. Clearly, the exact number of vendors associated with any message type (and all of its supertypes) will depend upon the particular message type. To reduce the complexity of this analysis, we introduce the concept of "average # of vendors per message received." Given this, we then have the following relationships for any given period of time:

<# of incoming messages> = <# of vendor queries> + <# of buyer messages>

<# of outgoing messages> = <# of buyer messages> × <average # of vendors per message received>

<storage load> = <# of message types>
+ <# of message types> × <average # of vendors per message type>

In general, there will be far more buyer messages than vendor queries. In addition, there will generally be many vendors who have registered to receive a particular message type. *Thus, the primary communication load on the server will be the outgoing communication load.*

To provide a concrete example of these various metrics, suppose that there are 5 vendors that have registered queries for the original BookRequest (GUID A-1-1-02 as shown in Figure 3), and that 10 vendors have registered queries on BookRequest2 (GUID C-2-1-02). The server will store these two GUIDs (A-1-1-02 and C-2-1-02) and the vendors interested in each of these message types. This storage load will be quite nominal. If 100 buyers send a BookRequest2 message, the server will have 100 incoming messages to process. The server will then need to send 100 × (10 + 5) = 1500 outgoing messages, routing the 100 incoming buyer messages to the interested vendors: the 10 vendors who explicitly queried for the BookRequest2 messages, and the 5 additional vendors who requested BookRequest (BookRequest2's supertype) messages.

The metrics described above are system-wide metrics, but to analyze the scalability of the solution, we need to introduce metrics that take into account the system (communication and storage) load on a per server basis. To do this, we simply take each of these three metrics and analyze them on a per server basis.

incoming messages/server
outgoing messages/server
storage load/server

For a single server system, it is evident that the total system load is equal to the per server load. Although a more complete development of the architecture is beyond the scope of this paper, an analysis that demonstrates the successful scaling of this architecture to multiple servers is given in an appendix available from the authors.

4.3 Limitations of the Implementation

Many significant implementation issues have not been addressed. It is possible that a single GUID and its derived types could form a very large structure with a very large number of vendors registered to receive queries for one or more GUIDs from this hierarchy. In this case neither the communication load nor the storage load would be distributed by the scaling procedures we have outlined in the appendix. We are exploring techniques to handle this situation using nested extensible hashing (Fagin, Nievergelt, Pippenger, and Strong, 1979), where the GUIDs of derived message types (rather than the root message types) would be used in these extreme cases. We note that these problems must also be addressed to make the UDDI/ebXML initiatives practical as currently proposed, and that significant resources are already being directed toward these issues.

In addition, the architecture presented has focused on buyer to vendor communication. A number of other issues such as optimal reply routes from vendors to buyers and the message type discovery mechanisms are also important to a fully deployable implementation of a semantic routing protocol, but have not been addressed by this research. Finally, we have not considered other mechanisms, contrasting with concept hierarchies, which have been proposed to attach semantics to electronic communications, notably the formalization of speech act theory (Verharen, 1997; Kimbrough and Thornburg, 1989).

5. Conclusions

The intent of this paper has been to propose a mechanism for dealing with semantic drift in electronic markets, defining messages as market-extensible concept hierarchies, and to demonstrate that such a mechanism can be practically implemented with existing infrastructure. We have identified a core set of requirements that must be handled by any e-commerce architecture in consideration of ongoing market evolution. In particular, we have argued that reducing buyer search costs is a special case of the more general problem of reducing coordination costs in evolving electronic markets, and that it can be decomposed into a problem involving computer-human interface issues and efficient message exchange between agents. XML record types identified by globally unique identifiers and arranged in an autonomously defined type hierarchy have been proposed as a means to provide extensible message exchange between agents in an open system environment.

A semantic routing protocol has been proposed as a conceptual solution to the message filtering problem. The architecture has been developed at a very high level as a proof of concept of the practical applicability of treating B2B transactions as conceptual inheritance hierarchies (Sacerdoti, 1974) to overcome some of the problems of evolving semantics in open markets. Note also that most of the techniques proposed for implementation of the architecture are currently in place and demonstrated as components of UDDI/ebXML. Further, the technology for

broadcasting a single message to multiple recipients efficiently has been the beneficiary of the significant interest over the past 20 years in webcasting (Furht et al. 1998).

The architecture presented in this paper has been meant only to provide an "existence proof" that scalable means for implementing the proposed semantic protocol exist, and not to argue that these are optimal. However, the existence proof underscores the two principal contributions of the paper. First, reasonable recognition and handling of new message types in an open system environment by mechanical agents can be straightforwardly achieved through the use of GUIDs, a universal XML syntax and a type hierarchy mechanism. This mechanism enables otherwise "dumb" agents to efficiently determine if they are capable of processing a message at all, and to gracefully degrade (without completely failing) given the introduction of new message types. Second, we believe that the concept of "intentional routing," i.e., routing messages to recipients based on what they want rather than who can be reached is important to aid the growth of electronic commerce.

Our current research is directed toward simulating the proposed architecture to determine approaches that further minimize network traffic, and in extending the underlying concepts of this architecture to address a broader array of electronic market coordination issues.

ACKNOWLEDGEMENTS

This work is partially supported by NSF Research Grants, IIS-9810901 and IIS-9811248, and a research grant to the first author from the Robinson College of Business, Georgia State University. David Kuechler, a doctoral student in the CIS program at Georgia State University, provided this analysis for an earlier version of this paper before his untimely death in November 2002.

REFERENCES

Andreoli, J.M., F. Pacull, and R. Perschi, "XPect: A Framework for Electronic Commerce," *IEEE Internet Computing*, Vol. 1, No. 4:40-48, July/August 1997.

Bakos, J.Y., "Reducing Buyer Search Costs: Implications for Electronic Marketplaces," *Management Science*, Vol 43, No. 12:1676-1692, December 1997.

Bakos, J. Y., "The Emerging Landscape for Retail E-Commerce," *Journal of Economic Perspectives*, Vol. 15, No. 1:69-80, Winter 2001.

Bichler, M. and A. Segev, "Brokerage in E-Commerce: State-of-the-Art and Open Issues," in *Proceedings of the 8th Annual Workshop on Information Technologies and Systems (WITS '98)*, Helsinki, Finland, pp. 53-63, 1998.

Bosak, J., "XML, Java, and the Future of the Web," Sun Microsystems, 1997, http://sunsite.unc.edu/pub/sun-info/standards/xml/why/xmlapps.htm, Last accessed July 10, 2003.

Bussler, C., D. Fensel, and A. Maedche, "Special Section on Semantic Web and Data Management: A Conceptual Architecture of Semantic Web Enabled Web Services," *ACM SIGMOD Record*, Vol. 31, No. 4:25-29, December 2002.

Chang, J., "Old Economy Industries Strive for Their Share of the Internet," *Chemical Market Reporter*, Vol. 257, No. 15:1-15, April 10, 2000.

Ciancarini, P., R. Tolksdorf, F. Vitali, D. Rossi, and A. Knoche, "Coordinating Multiagent Application on the WWW: A Reference Architecture," *IEEE Transactions on Software Engineering* Vol. 24, No. 5:362-375, May 1998.

Damsgaad, J. and D.Truex, "Binary Trading Relations and the Limits of EDI Standards: The Procrustean Bed of Standards," *European Journal Of Information Systems* Vol. 9, No. 3:173-188, September 2000.

Davenport, T., "Saving IT's Soul: Human-Centered Information Management," *Harvard Business Review*, Vol. 72, No. 2:119-131, March-April 1994.

Fagin, R., J. Nievergelt, N. Pippenger, and H.R. Strong, "Extendible Hashing—A Fast Access Method for Dynamic Files," *ACM Transactions on Database Systems*, Vol. 4, No. 3:315-344, September 1979.

Fensel, D., Y. Ding, B. Omelayenko, E. Schulten, G. Botquin, M. Brown, and A. Flett, "Product Data Integration in B2B E-Commerce," *IEEE Intelligent Systems*, Vol. 16, No. 4:54-59, July/August 2001.

Furht, B., R. Westwater, and J. Ice, "Multimedia Broadcasting Over the Internet: Part I," *IEEE Multimedia*, Vol 5, No. 4:78-82, October-December 1998.

Furnas, G. W., T.K. Landauer, L.M. Gomez, and S.T. Dumais, "The Vocabulary Problem in Human-System Communication," *Communications of the ACM*, Vol. 30, No. 11:964-971, November 1987.

GartnerGroup, "XML - The New EDI?" SPA-06-0528, November 1998.

GartnerGroup, "XML Builds the Collaborative Marketplace," DF-13-0461, March 2001.

Gasser, L., "Social Conceptions of Knowledge and Action: DAI Foundations and Open Systems Semantics," *Artificial Intelligence*, Vol. 47, No. 1-3:107-138, January 1991.

Gerson, E.M. and S.L. Star, "Analyzing Due Process in the Workplace," *ACM Transactions on Office Information Systems*, Vol. 4, No. 3:257-270, July 1986.

Glushko, R.J., J.M. Tenenbaum, and B. Meltzer, "An XML Framework for Agent-Based E-commerce," *Communications of the ACM*, Vol. 42, No. 3:106-114, March 1999.

Guha, R., R. McCool, and E. Miller, "Semantic Search," in *Proceedings of WWW2003*, May 20-24, Budapest, Hungary, pp. 700-709, 2003.

Gupta, A., D.O. Stahl, and A.B. Whinston, "Economic Issues in Electronic Commerce," in *Readings in Electronic Commerce*, R. Kalakota and A.B. Whinston (eds.), Addison-Wesley, Reading, MA, pp. 197-227, 1997.

Hendler, J., T. Berners-Lee, and O. Lassia, "The Semantic Web", *Scientific American*, Vol. 284, No. 5:35, May 2001.

Hewitt, C., "Offices Are Open Systems," *ACM Transactions on Office Information Systems*, Vol. 4, No. 3:271-287, July 1986.

Hewitt,C., "Open Information Systems Semantics for Distributed Artificial Intelligence," *Artificial Intelligence*, Vol. 47, No. 1-3:79-106, January 1991.

Hirschheim, R. "Understanding the Office: A Social-Analytic Perspective," *ACM Transactions on Information Systems*, Vol. 4, No. 4:331-344, December 1986.

Jain, A., M. Aparicio, and M.P. Singh, "Agents for Process Coherence in Virtual Enterprises," *Communications of the ACM*, Vol. 42, No. 3:62-69, March 1999.

Kamin, S.N. and U.S. Reddy, "Two Semantic Models of Object-Oriented Languages," in *Theoretical Aspects of Object-oriented Programming: Types, Semantics and Language Design*, C.A. Gunter and J.C. Mitchell (eds.), MIT Press, Cambridge, MA, pp. 463-495, 1994.

Karacapilidis, N. and P. Moraitis, "Building an Agent-Mediated Electronic Commerce System With Decision Analysis Features," *Decision Support Systems*, Vol. 32, No. 1:53-69, November 2001.

Kelly, R., "A Truly Global Market Place," *Wall Street & Technology*, Vol. 19, No. 3:62-64, March 2001.

Kimbrough, S. and M. Thornburg, "On Semantically Accessable Messaging in an Office Environment," in *Proceedings of HICSS* (22nd edn.), IEEE Press, pp. 10-18, 1989.

Kotok, A. and D. Webber, *ebXML: The New Global Standard for Doing Business over the Internet*, New Riders Publishing, 2002.

Krcmar, H., N. Bjørn-Andersen, and R. O'Callaghan (eds.), *EDI in Europe: How it Works in Practice*, John Wiley and Sons, New York, 1995.

Krazit, T., "Intel Does $5B in Transactions Through RosettaNet", *Computerworld*, December 2002.

Kuechler, W.L., V.K. Vaishnavi, and D. Kuechler, "Supporting Optimization of Business-to-Business E-Commerce Relationships," *Decision Support Systems*, Vol. 31, No. 3:363-377, August 2001.

Lee, J. and T.W. Malone, "Partially Shared Views: A Scheme for Communicating Among Groups that Use Different Type Hierarchies," *ACM Transactions on Office Information Systems*, Vol. 8, No. 1:1-26, January 1990.

Leukel, J., V. Schmitz, and F. Dorloff, "Modeling and Exchange of Product Classification Systems using XML," in *Proceedings of the 4th IEEE International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems*, 2002.

Malone, T.W., J. Yates, and R.I. Benjamin, "Electronic Markets and Electronic Hierarchies," *Communications of the ACM*, Vol. 30, No. 6, 484-497, June 1987.

Malone, T.W. and Crowston, K., "The Interdisciplinary Study of Coordination," *ACM Computing Surveys*, Vol 26, No. 1:87-119, March 1994.

Moore, G., *Crossing the Chasm: Marketing and Selling Technology Products to Mainstream Customers*, HarperBusiness, New York, NY, 1991.

OASIS (ebXML and UDDI specifications, white papers and resources), 2003, http://www.oasis-open.org/, Last accessed July 10, 2003.

Open Systems CFP, "First Workshop on Requirements Engineering in Open Systems," http://www.cs.uoregon.edu/~fickas/REOS, 2003, Last accessed July 10, 2003.

Riggins, F.J. and T. Mukhopadhyay, "Overcoming EDI Adoption and Implementation Risks," *International Journal of Electronic Commerce*, Vol 3, No. 4:103-123, Summer 1999.

Rogerson, D., *Inside COM*, Microsoft Press, Redmond, WA, 1997.

Sacerdoti, E., "Planning in a Hierarchy of Abstraction Spaces," *Artificial Intelligence*, Vol. 5, No. 2:115-135, Summer 1974.

Schlosser, M., M. Sintek, S. Decker, and W. Nejdl, "A Scalable and Ontology-Based P2P Infrastructure for Semantic Web Services," in *Proceedings of the 2nd IEEE Intl. Conference on P2P Computing (P2P2002)*, 2002, http://citeseer.nj.nec.com/schlosser02scalable.html, Last accessed July 10, 2003.

Spence, J., "Exchanges Bring Efficiency to Food Industry," *InfoWorld*, Vol. 22, No. 26:37-38, June 26, 2000.

Teoma, "Teoma™—Search with Authority", 2003, http://www.teoma.com, Last Accessed July 9, 2003.

Tenenberg, J., "Planning with Abstraction," in *Proceedings of AAAI '86*, pp. 76-80, 1986.

Tenenbaum, J.M., T.S. Chowdhry, and K. Hughes, "Eco System: An Internet Commerce Architecture," *IEEE Computer* Vol. 30, No. 5:48-55, May 1997.

Tsvetovatyy, M., M. Gini, B. Mobasher, and Z. Wieckowski, "MAGMA: An Agent-Based Virtual Market for Electronic Commerce," *Applied Artificial Intelligence*, Vol. 11, No. 6:501-524, 1997.

Tully, K., "Battle of the B2B Exchanges," *Corporate Finance*, pp. 12-15, November 2000.

Verharen, E.M., *A Language Action Perspective on the Design of Cooperative Information Agents*, Ph.D. Thesis, University of Brabant, Tilburg, Netherlands, 1997.

Verma, K., K. Sivashanmugam, A. Sheth, A. Patil, S. Oundhakr, and J. Miller, "METEOR-S WSDI: A Scalable Infrastructure of Registries for Semantic Publication and Discovery of Web Services," *Information Technology and Management*, in press.

von Bertalanffy, L., *General System Theory: Foundations, Development, Applications*, Braziller, New York, NY, 1968.