Agents for Sales Automation

Sermet Yucel

Certusoft
5117 Duggan Plaza
Edina, MN 55439
sermety@certusoft.com

From: AAAI Technical Report WS-99-02. Compilation copyright © 1999, AAAI (www.aaai.org). All rights reserved.

Abstract

Sales Automation systems face the complex task of integrating knowledge based systems with the traditional transactional systems running over a number of occasionally connected parties and software systems. The sharing and collaboration between occasionally connected parties and systems participating in the selling/buying of complex products have been difficult to realize within the common system architectures. We identify the fundamental challenges facing common Client/Server, Web, and Distributed Object architectures, and their shortcomings in supporting such collaboration and sharing. We show that agent based architectures can realize the wide area collaboration and workflow, as required by Sales Automation systems, over the Internet.

Sales Automation

Eighty percent of those surveyed in the 1998 Sales and Field Force Automation Reader Survey responded to the question of "Why Implement" a Sales Automation (SA) system with "To Increase Sales" (Goldenberg 1999). Respondents identified the E-Commerce and the Sales Configuration/Proposal Generation as top two imperatives over the next three years. The survey showed that providing mobile/portable hardware to sales force is the top priority. The connectivity, mobility and autonomy of the sales person, and the product knowledge are the most critical to increasing sales. The criterion for technology selection is its impact on the bottom line.

The emphasis on configuration is the evidence for that the product knowledge and the ability to use knowledge without being an expert is critical to a successful sale process. The proposal is one of the tools with which the customer participates in the sales process. A good proposal is a portable knowledge base about the products, services, specifications, features, and benefits that are of interest to the customer. The proposal also captures and presents the current state of the overall sale process. The knowledge and the ability to share it are the critical enablers for sellers, buyers, and their agents.

Copyright © 1999, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

The Internet has enabled the consumers to select and purchase off-the-shelf products as well as configurable products such as desktop computers. Armed with the on-line product knowledge, ability to configure, and ability to enter and track orders on-line, consumers are increasingly willing to buy without the help of a human "Sales Force". The sale of complex products on the Internet, however, proved more challenging. Specifying a complex product without the assistance of an industrial strength configurator may not be practical. Final decision on the configuration and sale of a complex product may take an extended period of time and involve a number of disconnected parties.

SA systems for complex products must manage and utilize large amounts of dynamic product knowledge and provide workflow support for processes spanning multiple heterogeneous hardware and software platforms. The SA of complex products is a special case of Wide Area Workflow Management (WAWM) (Riempp 1998). Empirical studies discovered that manual storage and movement of data between different systems can account for 80 percent of the time needed to complete an office process (Riempp 1998). WAWM targets this process overhead to minimize the overall processing time. The portable and electronic representation of the knowledge and the capability to share the knowledge within and/or between organizations can not only minimize the process overhead but it also is the foundation for creating global partnerships and collaborations.

Trucks, Construction and Manufacturing Equipment, HVAC Systems, Fire Pumps, Turbines, and Compressors are typical examples of complex products. The number of component types involved in configuring these products may be in the hundreds. Laws may mandate a minimal performance and specification on these products. Customers may impose performance and cost criteria. The Marketing Department may promote certain configurations by giving price discounts. The eventual specification of the product that meets engineering, marketing, legal and customer requirements may involve tens of thousands of rules to be checked interactively during the product configuration. A fast interactive

configuration and pricing engine is most critical to the success of the SA systems for the complex products (Yu, and Skovgaard 1998, Haag 1998).

The process of selling and purchasing a complex product may extend over many months due to changing customer requirements, sales and purchasing strategies, and product knowledge. It may involve geographically separate team members and vendors. This picture is in sharp contrast to the purchasing of simple products on the Internet where a single person can complete the process on a single vendor's web site within minutes.

A sales person working for an independent dealership may initiate the complex sales process. The sales person may request a nonstandard component from the manufacturer to meet the customer requirements. The manufacturer may accept or decline the request. The customer may decide to order if the manufacturer agrees to a certain delivery date. The manufacturer may decide to limit the warranty in turn. The sales manager may withdraw the discount associated with the standard configuration. The financing department may increase the interest rate. The customer may route the specifications to another vendor to get a quote for a customization before the final decision to order. Another sales person may take over the account and eventually enter the order. And so on.

What type of architecture is needed to enable multiple sales persons, customer contacts, and vendors to participate in selling and buying of a complex product? How do the participants collaborate? Who controls and coordinates the flow of data and processes that may cross many boundaries in geography, hardware, software infrastructure, and processes? In the this paper we review a possible three-tier client server architecture, a Web based architecture, and an agent based architecture as answers to these questions. Our goal is to identify the requirements and functionality of an agent-based approach and to compare an agent based architecture to these more common architectures. We identify roles of relevant AI techniques and reasons why an agent-based architecture is preferable.

Client/Server Approach

The Client/Server has been the dominant architecture for applications that serve a large number of permanently connected and concurrent users. The Client/Server architectures divide the application and its infrastructure into layers. The three-tier version consists of a data layer, a business or application layer, and a presentation layer (Keller and Teufel 1998).

From a mobility and autonomy perspective, the fundamental issue faced by the Client/Server architecture is that a sales person may not have a connection to the server all the time when he/she works with the system. Extending the Client/Server architecture to include mobile users means simply replicating a subset of all three layers and the necessary infrastructure to the disconnected computer. The master and replica databases, and the application and presentation layers may get out of synch as soon the mobile user disconnects. A robust and fault tolerant synchronization mechanism for all three layers and their infrastructure is necessary to keep these systems up and running. In principle, a three-tier architecture and a replication/synchronization mechanism can meet the demands of a mobile sales force selling complex products. The ability to access the same data and the functionality can potentially provide a seamless workflow and a high level of functionality like complex configuration.

As recently stated by the CEO of Oracle (Purdue 1999), the replication and synchronization of the data layer is too complicated. The implementation-level coupling between the application layer, the presentation layer, the data layer, and the infrastructure demands not only the ability to synchronize each layer but also the ability to synchronize the synchronization of each layer. In general, these systems may not recover completely from failures. For example, a rollback of the master database may require rebuilding of all the mobile databases and may result in loss of data. A new user must install all the necessary and possibly proprietary software including communication software, databases, components, and applications. These systems are scalable within enterprises but not to the masses served by the Internet. The functionality, workflow, and collaboration are limited to the licensed users of the software within a single enterprise and, if any, to the users of external systems that are interfaced.

The fundamental barrier to coordination and collaboration of diverse and disconnected users is the implementation level homogeneity and synchronization necessary for the operation and maintenance of these systems. The cost and the complexity of entry into these systems are prohibitive for large numbers of outside users who might occasionally need access. Most of the SA vendors provide mobile user support. Many provide integrated application and data synchronization tools. But momentum is building in favor of WEB based architectures. After spending $7 million, Oracle killed its internal Client/Server based SA initiative and declared that Web-based SA is the future (Purdue 1999).

WEB Approach

A Web-based approach delivers the benefits of the client/server at much larger scale. The Internet has standardized the client and the communication protocol. Enterprises may still have the same client/server data layer and the same business and application logic. However, the data and the applications do not need to be replicated to the user's computer. They are accessed on demand while the user is on-line. The complexity and the cost of entry to

the Internet are minimal. Unlike the Client/Server approach, in the Internet world the server makes no assumption about the client beyond the Internet standards. A server can easily talk to any client that could be any browser. The data and the functionality access on the Internet is a clear alternative to the Client/Server based SA.

The unit of communication between the Web server and the browser is an HTML document. The functionality of the document is limited by the contents of the document. From the client's perspective the HTML document is the application. The scripting and dynamic HTML add interactivity. However, creating an HTML document that can configure and price a complex product within a reasonable time is beyond the current capabilities of the HTML documents and the scripting languages.

The HTTP is designed for document and form processing, not for direct support of a distributed computing environment. A client application running in a browser can locate and execute methods on a server using a method invocation mechanism layered on top of HTTP. The network latency is the minimal performance overhead for a remote method invocation layered on top of HTTP. The latency of the HTTP, ~0.5 seconds, puts severe limits on the usability of the Internet as a distributed or a standard Client/Server computing environment. The next generation Internet protocol HTTP-NG will improve this situation but the latency will remain (Frystyk, et. al. 1998) as a fundamental bottleneck. With the anticipated proliferation of wireless communications that has 1-10 seconds latency, it will be impractical to make remote method calls. The imperative for the next generation systems is to minimize the calls to the server. In other words, computations, such as product configuration, involving more than a few method calls per second must be performed on the client.

Code-on-demand (Lange, and Oshima 1999) is another approach to overcome the limitations of the HTTP and HTML. Java Applets are the practical examples. The idea is to download the code to where and when you need it. The applet may run autonomously or it may still communicate with a server. Applets do not have the look, feel, and ease of use of the HTML. Unless they are really simple, download times may be intolerable.

Web does not offer a solution to a Sales Person on the road if he/she does not have a connection to the Internet. Sharing processes and collaborating globally are difficult within the client/server architectures, including the Internet, because business processing is confined to the application server and its connected clients. Business processes and their execution environments (and hosts) are inseparable. If inter- or intra-organizational business servers and/or their clients are not interoperable, standard client/server or the current Web model cannot provide the Global or Wide Area Workflow. The Internet is neither directly concerned with nor provider of such interoperability.

Mobile Agents

The phenomenal success of the Internet can be attributed to its design principles: ease of use and deployment, support for heterogeneity, scalability to millions of sites, cost effectiveness, simple and global connectivity as the overall goal (Carpenter 1996). We assume that a computing model for masses should build upon these proven principles. It should be upward compatible with the Internet. The browser concept and the look and feel of HTML are essential for ease of use and deployment.

We propose a Portable Knowledge Base (PKB) and a Visual Knowledge Base Browser (VKBB) as the successors to the HTML and the HTML browsers. In addition to the capabilities of the HTML documents, users can add, edit, query, route, and customize look and feel of a PKB. When users complete their work with PKBs, they can e-mail them to friends, or to colleagues, or they can submit the PKB to a server for processing. Users can post it in a read-only format on a Web site. One might refer to such a knowledge base as an Active Document (Stefik 1995) that embeds data, business rules, configuration rules, goals, and whatever else necessary to make it self-contained and visually interactive.

PKBs and VKBBs need to merge AI techniques, such as those in product configurators, with object-oriented techniques and languages, such as those in a Customer Relationship Management System. The product configurators are typically rule and constraint based and they include an inference engine. The configurators are closer to AI than to the object-oriented programming. This difference is the reason why configurators typically come with their own data modeling and maintenance tools. The internal representations of configuration objects are significantly different than common business objects like customer, order, or company. A rule that excludes a component combination is represented and used differently than a business rule that states that an "order cannot be entered if sales person and customer are in different states". A PKB for SA must represent the configuration and business entities (rules and objects) in a uniform way. The configuration objects, rules and constraints are typically considered data while the usual business rules are implemented as compiled or scripted code. PKB design should eliminate the distinction between rules as data and rules as compiled or scripted code as well as the distinction between the configuration and business objects. In summary, a PKB must merge the AI and object-oriented paradigms. The overall goal is to merge the business logic layer, data layer, presentation logic, and configuration logic and data into a single PKB.

Current approaches to mobile agents emphasize the ability to move from one computer to another. The goal is to give the impression that the agent is moving while executing. Java has been the choice of implementation language because it can serialize its objects into portable byte code (Lange, and Oshima 1999). What we are proposing here is the portability of the knowledge bases not their executable forms. Mobile agents typically require a distributed but homogenous host environment. The collaboration between the agents belonging to different agent systems is no likely to be easier than integrating different client server systems. For large-scale interoperability, an agent should execute, for example, in a Java environment first. Then its knowledge should be transferred to another agent environment, for example, in C/C++. The Portable knowledge base concept we are proposing here is very similar to HTML documents. An HTML document can be displayed without any assumption about its origin and implementation details of the target browser (We assume that both the HTML document and browser conform to same version of HTML). Documents can originate from a Web server written in C/C++ but can be displayed with a browser written in Java or Cobol by any third party vendor. What must be portable are the data, goals, processes, and history of the agent not its particular incarnation on a specific infrastructure.

Java offers many advantages for PKB and VKBB implementations. Java's unique capability to create and load classes dynamically is extremely advantageous for implementing knowledge bases that can be interactively updated while being used by end users. The ability to add methods and classes transparently to the end users are critical to realizing collaboration beyond simple information sharing. A PKB in a standard text based representation can be dynamically converted to internal Java byte code representation of the VKBB for execution. With the ability to create and add executable code to a running application, Java blurs the distinction between the development and usage of a system.

With the release of JDK 1.2, the Java user interface components can implement rich features expected by users. The VKBB should separate the implementation of presentation layer from its presentation logic. The VKBB must dynamically render the visual interface of the KKB from declarative statements in similar way a browser displays HTML documents using tags. The separation and standardization of the presentation layer implementation reduces the size of the distributed code, eliminates calls to the server for complex navigation and calculations, and improves performance.

Although our focus was on the portability and self-sufficiency, other characteristics of agents are not excluded. The planning, scheduling, problem solving and agent-to-agent communications can be added without compromising. To realize the full potential of the Internet, "collaboration at home and work" (Berners-Lee 1997), we must define mobile agents as portable knowledge bases, and we must replace the proprietary agent hosts with a generic knowledge base browser.

Conclusions

The collaboration and the coordination of a number of disconnected users and heterogeneous systems mandate the ability to share and communicate at the knowledge level. Client/Server architectures, mobile agent architectures that depend on proprietary execution environments, and distributed computing architectures (for example DCOM, and CORBA) require a permanent network connection to a server, and/or implementation level homogeneity at a global scale. Such homogeneity is neither practical nor desirable. A PKB embodies the data, processes, goals, plans, actions, and interaction details with human agents as well as software agents. The sharing and collaboration between disconnected users are supported by the physical portability and the self sufficiency of the PKBs that can travel across the Internet on its own, by actions of the current user, as a result of downloading by users, or by the mail. A KBB must dynamically render the visual interface from declarative statements stored within the PKB. Users can interact with the PKB using any implementation of the VKBB. When the PKB completes its mission, it can download or broadcast its knowledge to all the interested parties.

With XML, RDF, and RDFSchema, the World Wide Web Consortium has been already addressing the need for facilitation of knowledge sharing and exchange, interoperability of independently developed Web servers and clients, and metadata representation and transport (World Wide Web Consortium 1998a, 1999, 1999b). XML, RDF, and RDFSchema are designed to enable the machines to understand and exchange each other's data. VKBB transforms the machine understandable data (and business processes) to a visual format designed for human agents. The documents, PKBs, should be permanent and portable to support autonomy, and routing by a mobile user to a server or to another human agent.

Software interoperability by knowledge sharing, (Genesereth and Singh 1993), a general knowledge interchange format (Genesereth and Fikes 1992), and the representational portability (Gruber 1993) have already been proposed. The Knowledge Query and Manipulation Language (Labrou and Finin 1997) has been widely accepted as a standard for agent-to-agent communications. Technically, many pieces are in place for the PKB concept. Then, why are Agent or Knowledge Oriented software architectures not as commercially widespread as, for example, the object oriented architectures? The most likely reason is that the current standards do not adequately address the needs of the most important agent type: human agents. The VKBB may be considered as the

agent that bridges the gap between computer agents and human agents.

A sales process may involve a number of sales persons, sales engineers, sales managers, companies, contacts, products, quotes, orders, and departments. The PKB must hold the knowledge about all of these entities as well as the knowledge that are of interest to them. Its role is analogous to a file folder that contains all the paper work to be completed for the deal, the procedures to be followed, to do lists, and the product data sheets. The collaboration and workflow may be achieved by handing the folder to the next person who is responsible for the next step. The first person may simply walk down the floor to deliver the folder. The second person may air mail it. The PKB, on its own, may initiate a transfer by e-mail or by using Internet. Obviously, a real PKB implementation must overcome numerous other challenges that are beyond the scope of this paper. Our conclusion is that, regardless of the additional issues, physical and representational portability of knowledge, sharing at the knowledge level, self-contained knowledge bases, and knowledge base browsers are prerequisites to global collaboration and workflow.

References

Berners-Lee, T. 1997. Realising the Full Potential of the Web. *http://www.w3.org/1998/02/Potential.html*

Carpenter, B. 1996. Architectural Principles of the Internet, *Internet Architecture Board*, June 1996, RFC1958.

Frystyk, H., Spreitzer, M., Janssen, B., and Gettys, J. 1998. HTTP-NG Overview, Problem Statement, Requirements, and Solutions Outline. *Internet Draft, draft-frystyk-httpng-overview-00.txt,* November 1998. This work is in progress.

Genesereth, M. R., and Fikes, R. E. 1992. Knowledge Interchange Format, Version 3.0 Reference Manual. *Knowledge Systems Laboratory, KSL-92-86, June 1992. Computer Science Department, Stanford University.*

Genesereth, M. R., and Singh, N. 1993. A Knowledge Sharing Approach to Software Interoperation. *Logic Group, Computer Science Department, Stanford University.*

Goldenberg, B.; 1999. 1998 Sales & Field Force Automation Reader Survey. *Sales & Field Force Automation*, January 1999: 56-74.

Gruber, T. R. 1993. A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition.* 5(2):199-220.

Haag, A. 1998. Sales Configuration in Business Processes. *IEEE Intelligent Systems* 13(4): 78-85.

Keller, G., and Teufel, T. 1998. *SAP R/3 Process Oriented Implementation.* Reading, Mass.: Addison-Wesley.

Labrou, Y., and Finin, T. 1997. A Proposal for a new KQML Specification. *TR CS-97-03, February 1997, Computer Science and Electrical Engineering Department, University of Maryland Baltimore County*.

Lange, D. B., and Oshima, M. 1999. *Programming And Deploying Java Mobile Agents With Aglets.* Reading, Mass.: Addison-Wesley.

Purdue, M.; 1999. Lary Ellison's New World Order. *Sales & Field Force Automation*, March 1999: 14-15.

Stefik, M. 1995. *Introduction to Knowledge Systems.* San Fransisco, California: Morgan Kaufmann Publishers, Inc.

Riempp, G. 1998. *Wide Area Workflow Management, Creating Partnership for the 21st Century.* London: Springer-Verlag.

Yu, B., and Skovgaard, H. J. 1998. A Configuration Tool to Increase Product Competitiveness. *IEEE Intelligent Systems* 13(4): 34-41.

World Wide Web Consortium 1999. Resource Description Framework Model and Syntax Specification. *http://www.w3.org/TR/REC-rdf-syntax*

World Wide Web Consortium 1998a. Extensible Markup Language (XML) 1.0. *http://www.w3.org/TR/REC-xml*

World Wide Web Consortium 1998b. Resource Description Framework Schema Specification. *http://www.w3.org/TR/WD-rdf-schema*. This work is in progress.