Automating Supply Chains

Michael N. Huhns • *University of South Carolina* • *huhns@sc.edu*
Larry M. Stephens • *University of South Carolina* • *stephens@sc.edu*

A recent study found that supply-chain problems cost companies between 9 and 20 percent of their value over a six-month period.[1] The problems range from part shortages to poorly utilized plant capacity. When you place this in the context of the overall business-to-business (B2B) market expected to reach US$7 trillion by 2004 (37 percent of which is projected to be e-commerce sales),[2] it's easy to see that effective supply-chain management (SCM) tools could save companies billions of dollars.

Attempts to automate solutions to these problems are complicated by the need for the different companies in a supply chain to maintain the integrity and confidentiality of their information systems and operations. The modeling technologies currently used within the manufacturing business-to-business standards communities — such as the Open Applications Group (http://www.openapplications.org) and RosettaNet (http://www.rosettanet.org) — do a good job of capturing user requirements. Unfortunately, current technologies do not explicitly link the requirements to formal process models. This missing link is crucial to efficient SCM implementations.

Automation Requirements

One way to automate supply chains is to gather companies into e-marketplaces (such as ChemConnect, http://www.chemconnect.com, for chemicals and Covisint, http://www.covisint.com, for automotive supplies), where they can negotiate for goods and services. However, because companies must participate independently, such centralization does not foster the kinds of alliances or long-term relationships that can significantly improve supply-chain efficiency.

A distributed architecture with point-to-point connections is thus preferable, but computer applications that automate supply chains require several properties beyond traditional software approaches:

- *Disintermediation* (the direct association between users and their software).[3] Seamless access to and interaction with remote information, application, and human resources requires a distributed active-object architecture.[4]
- *Dynamic composability and execution.* A system should execute as a set of distributed parts, but the resources required will be mostly unknown until runtime. Thus the infrastructure must enable resource discovery and composition as needed.
- *Interaction.* Interaction among participants might include subtle and critical patterns, but the specific interactions might be variable and unknown until runtime. The patterns must therefore be explicitly represented and reasoned with. Recent work describes the power of interactions.[5]
- *Error tolerance and exploitation.* As the deployed systems gain complexity, they should anticipate and compensate for errors in their components and interaction protocols.

Recent advances in software agent architecture and languages can address these requirements.

Agent-Based Automation

We have been collaborating with a team from the Manufacturing Systems Integration Division at the U.S. National Institute of Standards and Technology (http://www.mel.nist.gov/msid/) to identify and test methods for automating SCM. In this process, we investigated a coordination methodology reported in previous development work on autonomous agents.

The methodology has its origins in the work of linguist Robert A. Dooley. In 1976, Dooley invented a graphical notation to show the structure of conversations among people.[6] Van Parunak adopted this work in 1996 for his work on agent inter-

 1089-7801/01/$10.00 ©2001 IEEE

actions, which he realized were a lot like human conversations.[7] Then in 2000, Munindar Singh recognized that Parunak's Dooley graphs looked like the structures of database transactions he was trying to implement, and applied them to the automated construction of agents for managing those transactions.[8]

We extended Singh's application to supply-chain management and B2B interactions.[9] Our methodology begins with a supply-chain or B2B scenario. Such a scenario, and its associated Unified Modeling Language (UML) interaction diagram, which is exemplified in Figure 1, represent the starting point for automating the business interactions among a number of independent organizations.

The interactions in Figure 1 consist of the exchange of structured documents, which the OAG calls business object documents (BODs). For B2B interactions, a `ProcessPO` BOD is a *directive* that carries the composite semantics of *request* and *inform*; that is, the sender requests that the recipient evaluate the PO and inform the sender of the results. The informal semantics is that `ProcessPO` will be followed by a response from the recipient and that the response will be either an `AckPO` or a `DeclinePO`. The semantics of each document is being formalized.

Next, using the formal semantics, a tool under development can convert the messages in the interaction diagram into a bipartite conversation graph (not shown here), which delineates each participant's conversations. A bipartite conversation graph also helps identify the roles of the participants in B2B transactions. This graph is the basis for constructing Dooley graphs, shown in Figure 2 (next page) in their equivalent form as collaboration diagrams.[5] Note that collaboration participants can fill different roles at different times, and thus can be involved in many conversations simultaneously.

A software agent can fill each of the roles identified in the collaboration

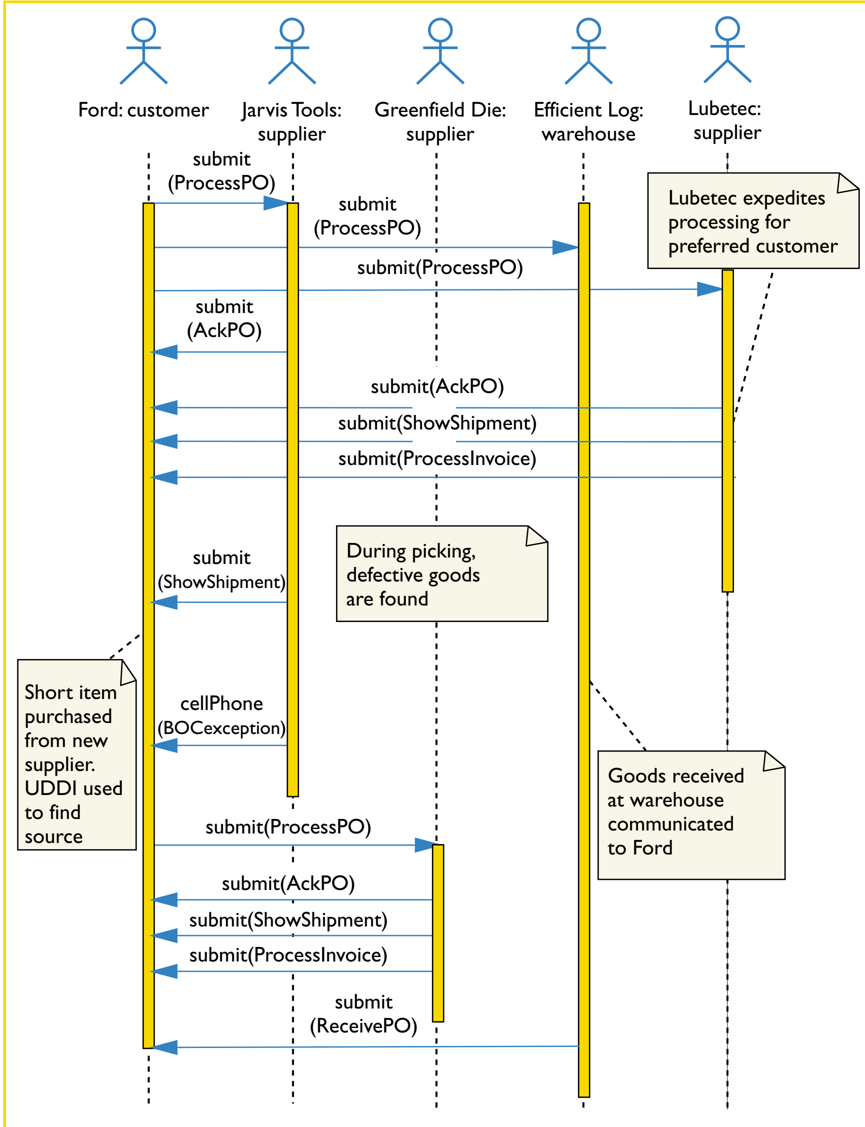

Figure 1. A UML interaction diagram. This diagram, along with the scenario it represents, is the first step in automating business interactions among independent organizations.

diagram. Moreover, the diagram for each role can be converted directly into a state-machine description for the agent's behavior, enabling automatic agent generation. After being installed at each company, the agents manage the B2B supply-chain process. Figure 3 (next page) shows several of the state-machine behavioral descriptions.

Figure 4 (page 93) summarizes the steps in our methodology. The methodology uses — and begins to formalize — the BODs that OAG and RosettaNet are standardizing. It provides a basis for the convergence of multiple standards for supply-chain management, which could become ready-to-use technology for software vendors.

Standardization Efforts

Consortiums of industrial component manufacturers, distributors, system integrators, and resellers, such as OAG, RosettaNet, Oasis (http://www.oasis-open.org), and the United Nations Center for Trade Facilitation and Electronic Business (UN/Cefact, http://www.unece.org/cefact), are working to define and standardize electronic business processes. For example, Rosetta-

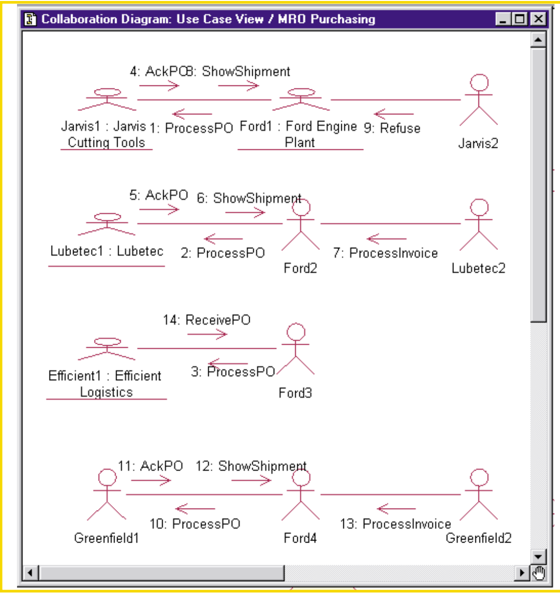

Figure 2. A collaboration diagram (Dooley graph), built from a bipartite conversation graph. Software agents can fill each of the roles identified in the diagram.

Net's members have made significant progress in implementing and adopting *partner interface processes* (PIPs), which are XML-based definitions of business processes, terminology, and BODs. So far, they have developed more than 120 PIPs to aid circuit-board manufacturers and assemblers in distributing product information and providing shipping notices.

UN/Cefact and Oasis are sponsoring ebXML, a set of specifications that enables enterprises of different sizes and in different locations to conduct business over the Internet. Using ebXML, companies can exchange business messages, conduct trading relationships, communicate data in common terms, and define and register business processes.

Similarly, the OAG has specified more than 55 business scenarios that describe interactions for purchasing, order management, billing, shipping, receiving, and financials.

Several open issues remain. For example,

- no formal process exists for developing scenarios or PIPs,
- reuse approaches have not been specified,
- validation techniques have not been defined, and
- there are mismatches among the process levels.

NIST is working to resolve these issues, while providing a principled basis for the documents that are being standardized.

Conclusion

The methodologies we've described here promote the interchange of stan-

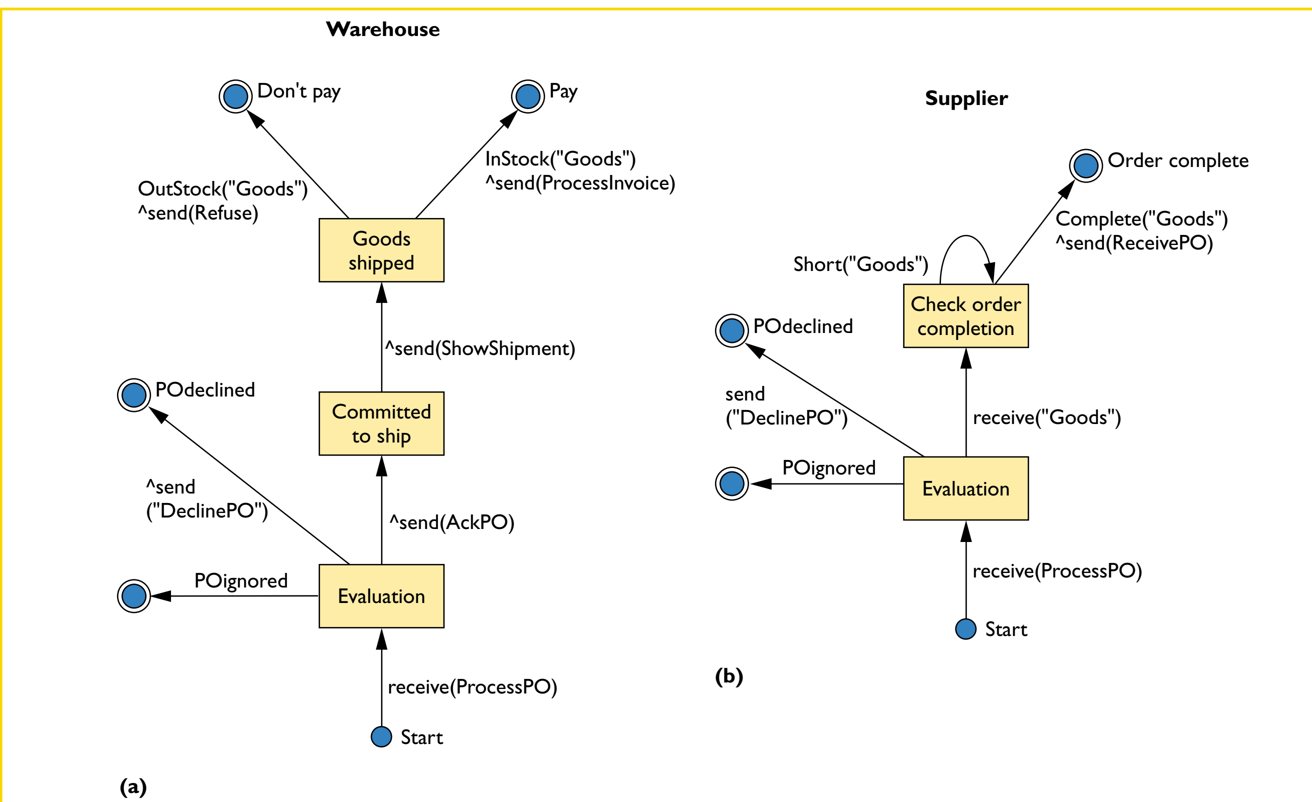

Figure 3. State-machine behavioral skeletons for enacting (a) a warehouse agent and (b) a supplier agent. Both agents implement B2B supply-chain processes.

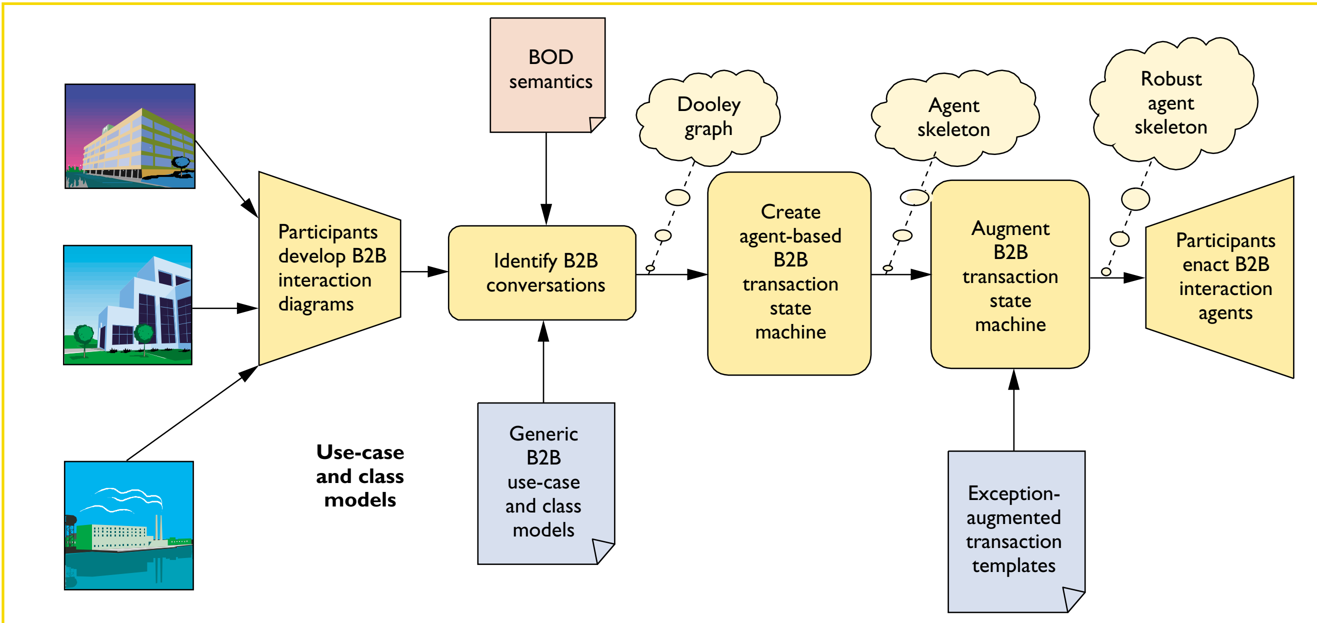

Figure 4. Agent-based coordination methodology for B2B automation.

dard business documents and compensate for exceptions that might occur during execution. Enterprises need only describe their supply processes using OAG standard business documents and UML interaction diagrams. The methodologies and tools convert the diagrams into specifications for software agents, which then cooperate in automating the resultant supply chain.

Although our work to date indicates that supply-chain automation using software-agent technology is feasible, its widespread adoption will require appropriate standards so that companies can confidently invest their efforts in techniques that will truly be interoperable.

Acknowledgments

We have been collaborating with Nenad Ivezic, Michael Gruninger, and James G. Nell at the U.S. National Institute for Standards and Technology, which supported the work reported here.

References

1. T.J. Becker, "Putting a Price on Supply Chain Problems: Study Links Supply Chain Glitches with Falling Stock Prices," *Georgia Tech Research News*, 12 Dec. 2000, available at http://www.gtresearchnews.gatech.edu/newsrelease/CHAINR.html.
2. C. Pettey, "Future of Online Marketplaces and Exchanges to Be Discussed at Gartner's B2B eMarketplaces Conference," Gartner, http://gartner11.gartnerweb.com/public/static/aboutgg/pressrel/pr20001004b.html
3. M. Shaw, "Outlook on Software System Design," *Computer*, vol. 31, no. 1, Jan. 1998, p. 32.
4. D. Woelk, M. Huhns, and C. Tomlinson, "Uncovering the Next Generation of Active Objects," *Object Magazine*, July-Aug. 1995, pp. 33-40.
5. P. Wegner, "Why Interaction Is More Powerful Than Algorithms," *Comm. ACM*, vol. 40, no. 5, May 1997, pp. 80-91.
6. R.A. Dooley, "Repartee as a Graph," in *An Anatomy of Speech Notions*, R.E. Longacre, ed., Peter de Ridder, Lisse, 1976, pp. 348-358.
7. H. Van Dyke Parunak, "Visualizing Agent Conversations: Using Enhanced Dooley Graphs for Agent Design and Analysis," *Proc. Second Int'l Conf. Multiagent Systems*, AAAI Press, Menlo Park, Calif., 1996, pp. 275-282.
8. M.P. Singh, "Synthesizing Coordination Requirements for Heterogeneous Autonomous Agents," *Autonomous Agents and Multi-Agent Systems*, vol. 3, no. 2, June 2000, pp. 107-132.
9. N. Ivezic et al., "An Analysis of a Supply-chain Management Agent Architecture," *Proc. Fourth Int'l Conf. Multiagent Systems*, IEEE Computer Soc. Press, Los Alamitos, Calif., July 2000, pp. 401-402.

Michael N. Huhns is a professor of computer science and engineering at the University of South Carolina, where he also directs the Center for Information Technology.

Larry M. Stephens is a professor of computer science and engineering at the University of South Carolina, where his primary research interests are agents and ontologies.