Paper No. 483

Automating Supply-Chain Management

Michael N. Huhns
University of South Carolina
Center for Information Technology
Columbia, SC 29208 USA
+1-803-777-5921
huhns@sc.edu

Larry M. Stephens
University of South Carolina
Center for Information Technology
Columbia, SC 29208 USA
+1-803-777-2895
stephens@sc.edu

Nenad Ivezic
Nat'l. Inst. of Standards & Technology
Manufacturing System Integration Div.
Gaithersburg, MD 20899 USA
+1-301-975-3536
nivezic@nist.gov

ABSTRACT

This paper explores a linguistic approach to coordination modeling as a formal basis for supply-chain management (SCM) in manufacturing. The approach promotes the interchange of standard documents: enterprises need only describe their supply processes using OAG business object documents and UML interaction diagrams. Our methodology and tools analyze the documents and interactions in terms of four linguistic primitives and convert the diagrams into specifications and implementations of software agents. The agents then cooperate in automating the resultant supply chain. We evaluate our methodology in the context of several industrial scenarios. We conclude that supply-chain automation using software-agent technology is feasible.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence – *multiagent systems.*

General Terms

Management, Economics, Standardization

Keywords

Supply-chain automation, agent generation, agent-based process control.

1. INTRODUCTION

A recent study [1] has found that companies lose between 9% and 20% of their value over a six-month period due to supply chain problems. The problems range from part shortages, excessive finished good inventories, underutilized plant capacity, unnecessary warehousing costs, and inefficient transportation of supplies and finished goods. Because supply chains involve independent participants—suppliers and manufacturers—who must maintain the integrity and confidentiality of their information systems and operations for business advantages, the problems are exacerbated.

One approach to automating supply chains is to gather companies into e-marketplaces, where they can negotiate for goods and services [11]. However, such centralization does not foster collaborations, alliances, and long-term relationships, which are the more significant drivers of improved efficiency in supply chains. A distributed architecture is thus preferable, but computer applications that can automate supply chains require a number of important properties beyond traditional software approaches.

- **Disintermediation** (the direct association between users and their software). Providing a participant with seamless access to and interaction with remote information, application, and human resources requires a distributed, active-object architecture.
- **Dynamic composability and execution.** A system should execute as a set of distributed parts, but the resources required will be mostly unknown until run-time: this requires an infrastructure to enable their discovery and composition as needed.
- **Interaction.** There might be subtle and critical patterns of interaction among supply-chain participants, but the specific interactions may be unknown until run-time, and may vary: this requires that the patterns of interaction be explicitly represented and reasoned with. There is recent work on the power of interactions [16].
- **Error tolerance and exploitation.** As deployed systems become increasingly complex, they should anticipate and compensate for errors in their components and interaction protocols.

Recent advances in software agent architecture and languages can address the above requirements. This reported effort has made use of these advances in investigating and developing tools and methodologies for supply-chain management [10].

The Open Applications Group (OAG) [12], ebXML, and RosettaNet are standardizing the syntax and semantics of B2B transactions. We are developing a basis for standardizing (and automating) the *behaviors* that are expected of the participants in a B2B transaction (also, how to handle misbehaviors). For example, current specifications of a purchase order (PO) do not say what to do in the following case: if a company does not receive a response to a PO, should it assume the recipient was not interested or the PO was lost?

We have found that a B2B transaction (such as in supply chains) is a formal conversation [5], in the linguistic sense, among several participants (buyers and sellers or consumers and suppliers) that follows certain rules and conventions. By making the rules and conventions explicit, we can guarantee

- The correctness of the transactions
- There are no misunderstandings among the participants
- Exceptions are handled.

AAMAS'02, July 15-19, 2002, Bologna, Italy.

Our approach is an agent-based coordination methodology [8,9,13], utilizing linguistic models with formal logic, process semantics, and accommodations for exceptions.

2. METHODOLOGY

We have been working to identify and test methods for automating supply-chain management. The general approaches that have been suggested for automation can be categorized as centralized, distributed, and agent-based. As discussed in the above section, the requirements for robustness and efficiency favor an agent-based approach. Previous coordination methodologies developed in the area of autonomous agents are relevant to SCM [10,14,15]. We performed preliminary evaluations of these methodologies and designed a prototype software system that could automate the construction of industrial supply chains and B2B processes. For a given B2B scenario, the prototype software system:

1. Captures the scenario as a UML interaction diagram
2. Converts the UML description into a DAML-OIL (DARPA Agent Markup Language-Ontology Interchange Language) [2] description
3. Processes the DAML-OIL description of the interaction diagram to extract B2B conversations
4. Creates state machines for agent behavior in B2B transactions
5. Augments the state machines to include exception-handling
6. Enacts agents to represent the B2B participants and their software systems.

We performed our analysis within the context of several industrially specified scenarios formulated by the OAG. The scenarios are typified by the UML interaction diagram in Figure 1. Such scenarios, and their associated interaction diagrams, would represent the starting points for setting up automated supply chains among a number of independent organizations. Figure 2 shows the resultant use cases (for a scenario involving Ford and four of its suppliers) that can be identified from the interaction diagram. Our precise methodology is

1. Participants construct an interaction (i.e., sequence) diagram in UML. The messages in the sequence diagram must be standard business documents, such as the OAG Business Object Documents (BODs).
2. Participants provide values for the parameters of the BODs. To automate this, a tool is being constructed that parses the interaction diagram, identifies the BODs, and queries the participants for the values. For example, the parameters needed in the subsequent phase to identify the threads of a purchase-order conversation are as follows:
   a. `ProcessPO (id, sender, receiver)`
   b. `AckPO (id, sender, receiver, {partial/final}, {accept/reject/modify})`
   c. `ShowShipment (id, sender, receiver)`
   d. `ProcessInvoice (id, sender, receiver)`

   We represent this information in DAML-OIL and store it in a .daml file. The file is then validated against the DAML-OIL representation for standard BODs that we created.
3. The Conversation Table Generator software takes as input the DAML-OIL file representing a particular scenario and produces a conversation table. Entries in this table are produced by applying the following general rules:
   - **Rule 1:**
     ```
     If first message in a scenario
     Then all entries := 0.
     Else If messageID(BOD-instance) ==
       messageID(earliest BOD-instance)
     Then respondTo(BOD-instance) :=
       messageNumber(earliest BOD-instance)
     Else If messageID(BOD-instance) has no
       earlier match AND
       sender(BOD-instance) ==
         receiver(previous BOD-instance)
     Then respondTo(BOD-instance) :=
       messageNumber(previous BOD-instance)
     Else respondTo(BOD-instance) := 0
     ```
   - **Rule 2:**
     ```
     If type(BOD-instance) == AckPO
       AND NOT decision(BOD-instance) == Refuse
       OR type(BOD-instance) == ShowShipment
       OR type(BOD-instance) == ReceivePO
       OR type(BOD-instance) == ANSI_X12_855
     Then
       replyTo(BOD-instance) :=
         respondTo(BOD-instance) AND
       resolve(BOD-instance) :=
         respondTo(BOD-instance) AND
       complete(BOD-instance) := 0
     Else replyTo(BOD-instance) := 0 AND
       resolve(BOD-instance) := 0
     ```
   - **Rule 3:**
     ```
     If type(BOD-instance) == ProcessPO OR
       type(BOD-instance) == ANSI_X12_850
     Then
       complete(BOD-instance) := 0
     Else If [type(BOD-instance) ==
                 ProcessInvoice OR
       type(BOD-instance) == AckPO AND
       decision(BOD-instance) == Refuse] AND
       messageID(BOD-instance) ==
         messageID(earliest AckPO-instance) AND
       status(earliest AckPO-instance) == Final
     Then
       complete(BOD-instance) :=
         messageNumber(earliest AckPO-instance)
     ```
4. From the conversation table, the Dooley Graph Generator software generates a collaboration diagram (Dooley graph).
5. From the Dooley graph, the Agent Generator software generates the state machines for agent behaviors representing each role that the business entities are assuming.

The interactions in Figure 1 consist of the exchange of structured documents termed by the OAG Business Object Documents (BODs). For B2B interactions, a `ProcessPO` BOD is a *directive* that carries the composite semantics of *request* and *inform*, i.e., the sender requests the recipient to 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`.

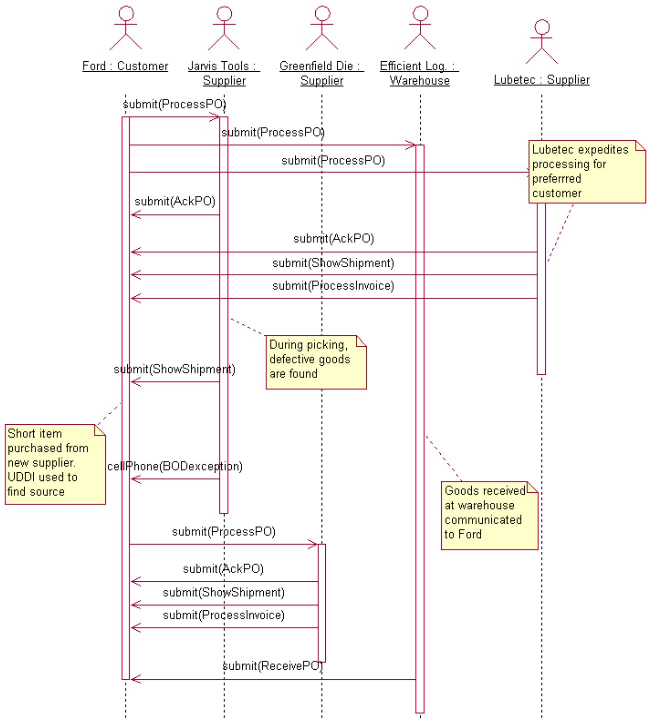

Figure 1: Interaction diagram for the OAG scenario involving Ford and its suppliers.

A more formal semantics can be specified using DAML-OIL. This is shown for the Ford scenario in Appendix A. Alternatively, the semantics for the BODs can be represented using the PSL formalism [6].

Such BOD semantics are used to construct conversation tables (as shown in Table 1) automatically and then check the consistency of the messages in the tables. Each message (document) that is exchanged during a B2B transaction is analyzed in terms of four

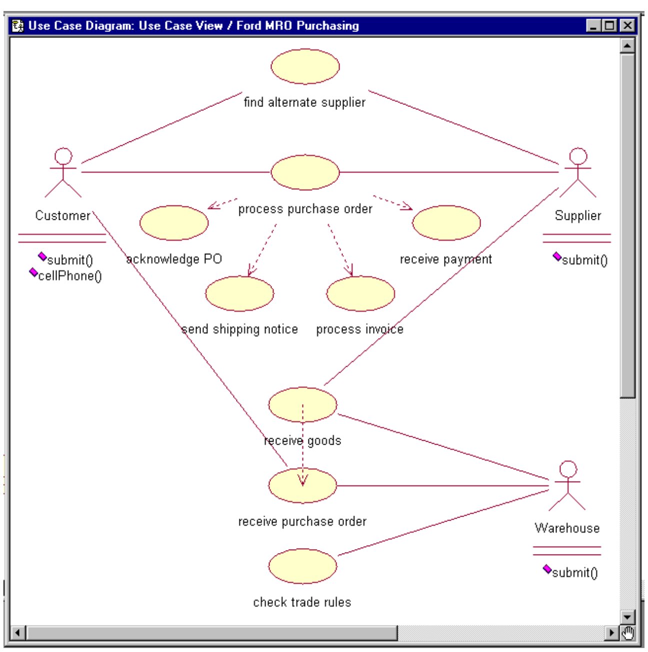

Figure 2: Use-case model for Ford supply-chain interoperability scenario.

(antisymmetric and irreflexive) binary relations: (1) *respondsTo*, (2) *repliesTo*, (3) *resolves*, and (4) *completes*. Please see [14,15] for a precise specification of the semantics of these relations.

They enable different B2B protocols to be compared and analyzed for correctness and completion. They also enable a large protocol to be decomposed into a number of smaller, standardized subprotocols, where the participants have simple predefined roles.

The next steps in the methodology are to convert the messages in the conversation table into a bipartite conversation graph, as shown in Figure 3, and then into a collaboration diagram (Figure 4) that delineates the specific conversations in which each participant is engaged. The arcs in Figure 3 help identify the roles of the participants in the B2B transactions. The notation Ford1 indicates that message 1 (to Jarvis) is being sent by Ford; the notation 4Ford indicates that message 4 (sent by Jarvis) is being received by Ford. More specifically, the arc from Ford1 to 4Ford indicates that Jarvis in message 4 is *replying* to Ford's message 1 (ProcessPO). The arc from Lubetec6 to 2Lubetec indicates that message 6 (ShowShipment) *resolves* Ford's message 2 (ProcessPO). The arc from Ford2 to 7Ford indicates that message 7 *completes* intermediate messages 5, which, in turn, *resolves* message 2. Rules for constructing these arcs are found in [14].

This graph is the basis for constructing Dooley graphs [14], shown in Figure 4 in their equivalent form as collaboration diagrams. Dooley graphs help identify the roles of participants in a B2B transaction. Note that participants in collaborations can fill different roles at different times, and thus can be involved in many simultaneous conversations. The role changes that occur over time for each participant in a B2B transaction can be shown as histories or as partitioned character timelines. We show these for the Ford scenario in Table 2.

Both Singh [14] and Parunak [15] encountered the problem that parts of a conversation that *should* be connected are not. This is because some messages require multiple responses, and their methodology could not capture this. For example, a ProcessPO message is two requests: "Will you supply the item?" and "Will you inform me when you ship?" We have solved this by allowing partial replies to earlier messages. The result is that parts of an overall conversation are reconnected meaningfully, thereby preventing the proliferation of participant roles.

A software agent can fill each of the roles that can be identified in the collaboration diagram. Moreover, the diagram for each role can be converted directly into a state-machine description for the behavior of the agent. This leads to a capability for automatically generating the agents, who then operate as managers of the B2B supply-chain process. Several of the state-machine behavioral descriptions are shown in Figure 5, with a textual description for one of the Ford roles as follows:

```
Agent State Machine Behavior for Ford2
  States:  Start, State1, Stop, State2, State3
  Arcs are
    send:ProcessPO         from Start   to State1
    receive:Refuse         from State1  to Stop
    receive:AckPO          from State1  to State1
    receive:Timeout        from State1  to Start
    receive:AckPO          from State1  to State2
    receive:ShowShipment   from State2  to State2
    receive:ProcessInvoice from State2  to State3
```

Table 1. Conversation Table for Ford Interoperability Scenario

ID	Sender	Receiver	Message	Respond To	Reply To	Resolve	Complete
1	Ford	Jarvis	ProcessPO				
2	Ford	Lubetec	ProcessPO				
3	Ford	E Logistics	ProcessPO				
4	Jarvis	Ford	AckPO	1	1	1	
5	Lubetec	Ford	AckPO	2	2	2	
6	Lubetec	Ford	ShowShipment	2	2	2	
7	Lubetec	Ford	ProcessInvoice	2			5
8	Jarvis	Ford	ShowShipment	1	1	1	
9	Jarvis	Ford	Refuse	1			4
10	Ford	Greenfield	ProcessPO	9			
11	Greenfield	Ford	AckPO	10	10	10	
12	Greenfield	Ford	ShowShipment	10	10	10	
13	Greenfield	Ford	ProcessInvoice	10			11
14	E. Logistics	Ford	ReceivePO	3	3	3	

Table 2. Messages and histories with partial replies

Role	History
Ford	(F1, 1, J1); (F2, 2, L1); (F3, 3, E1); (J1, 4, F5); (L1, 5, F6); (L1, 6, F2);(L2, 7, F2); (J1, 8, F1); (J2, 9, F1); (F4, 10, G1); (G1, 11 F7); (G1, 12, F4); (G2, 13, F4); (E1, 14, F3)
Jarvis	(F1, 1, J1); (J1, 4, F5); (J1, 8, F1); (J2, 9, F1);
Lubetec	(F2, 2, L1); (L1, 5, F6); (L1, 6, F2); (L2, 7, F2);
Greenfield	(F4, 10, G1);(G1, 11 F7);(G1, 12, F4);(G2, 13, F4);
Efficient Logistics	(F3, 3, E1); (E1, 14, F3)

3. Automated Exception Handling

3.1 Types of Exceptions

Exceptions can occur at a variety of places in a B2B process and in a variety of forms. In order to accommodate exceptions, they must all be anticipated. One way of doing this exhaustively is to consider the inverse of all goals, as done in [7], which helps in identifying the failure mechanisms for all goals. We have applied this to our specific B2B scenario, with the following result:

- *Goal*: Unachieved state
- *Deadline*: Missed; Achieved late
- *Product goal*: Product violates constraints; Wrong quantity
- *Payment*: Missed; Sent late; Wrong amount
- *Order*: Product violates constraints; Wrong quantity; Missed deadline; Wrong customer
- *Storage*: Wrong product; Wrong quantity; Wrong location
- *Delivery*: Wrong product; Wrong quantity; Missed deadline; Wrong customer; Wrong location

An alternative approach is an exception-type taxonomy [3,4].

3.2 Incorporating Exceptions in Agents

The result of applying this exception-handling analysis to B2B transactions consisting of OAG BODs is that for messages

refuse ➔ refuse | commit | timeout

commit ➔ refuse | commit | timeout

and for actions on state transitions

getAckPO ➔ getAckPO | getRefuse

getAckPO ➔ getAckPO | getRefuse

where the symbol ➔ means "is replaced by." The result is an agent description that handles exceptions automatically, leading to a robust implementation of B2B transactions. Figure 6 shows how an agent description is augmented to consider exceptions.

Figure 7 summarizes all of the steps in the methodology that we have formulated above.

4. DISCUSSION

The methodology investigated in this paper provides a basis for the convergence of multiple standards for supply-chain management that could potentially become ready-to-use technology for software vendors. The methodology makes use of—and begins to formalize—the standard business documents that OAG and RosettaNet are developing. We have also produced a prototype that demonstrates how the methodology can be automated. Although promising, it has not yet been deployed.

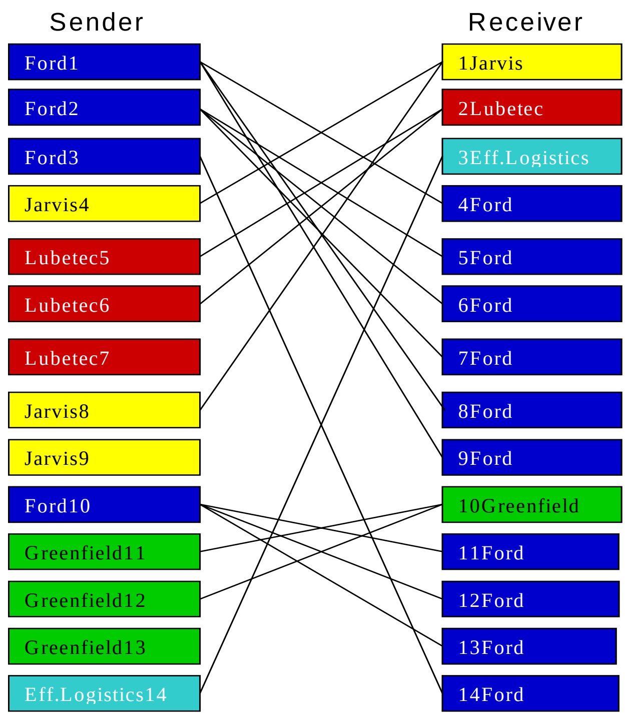

Figure 3: Bipartite conversation graph derived from the conversation table for the Ford scenario.

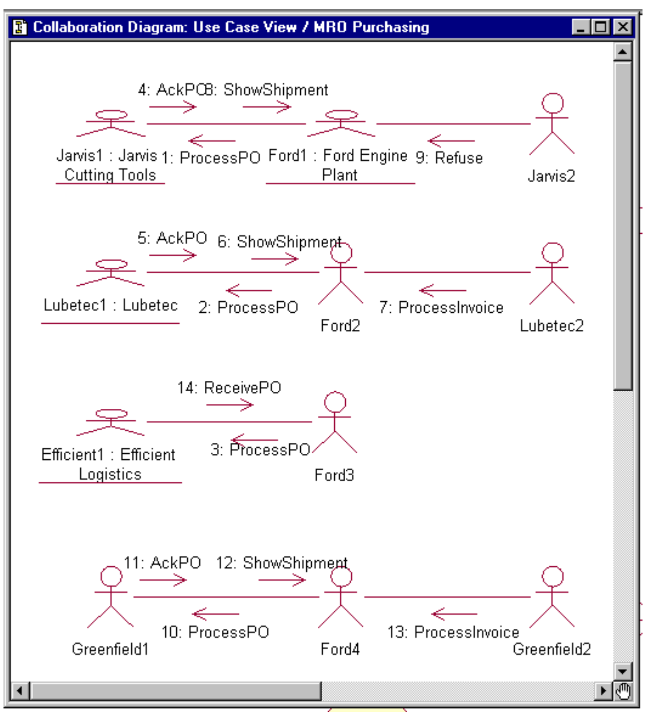

Figure 4: Collaboration diagram (Dooley graph) for Ford interoperability scenario, constructed from bipartite conversation graph and showing participant roles.

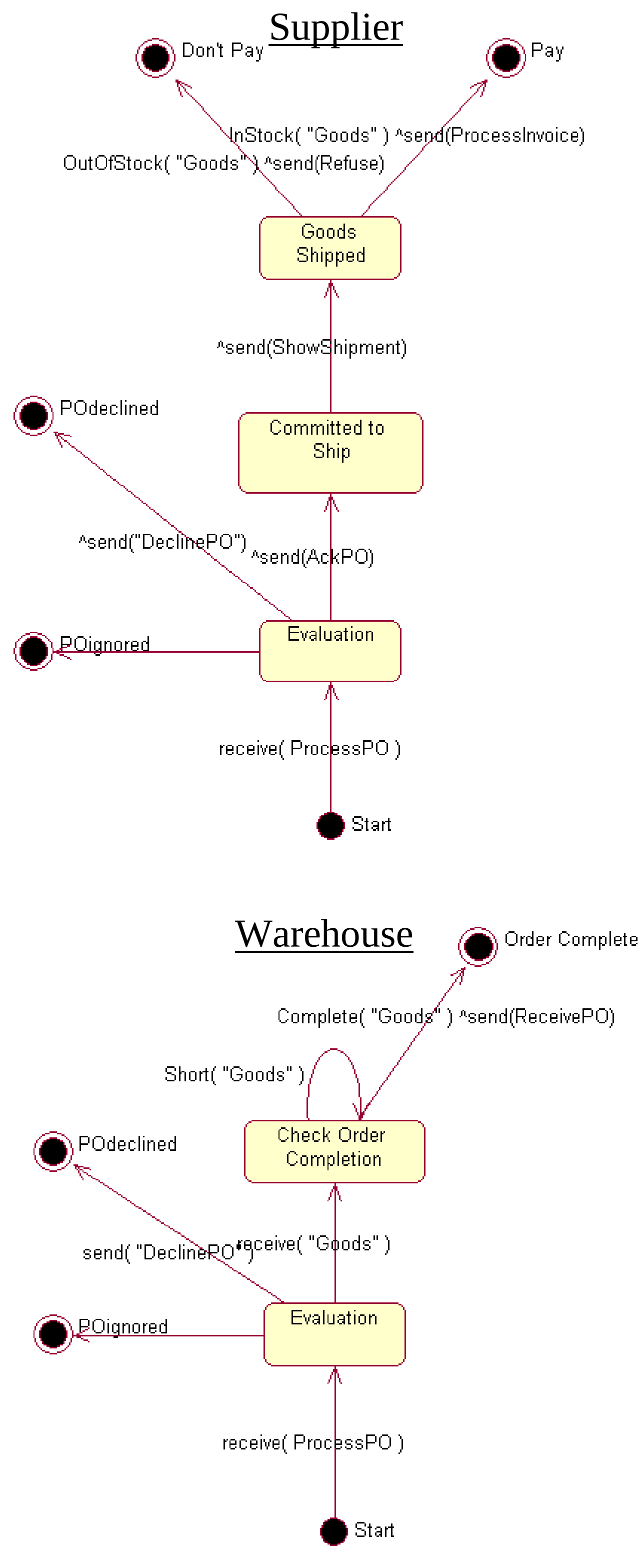

Figure 5: State-machine behavioral descriptions for enacting agents that implement B2B supply-chain processes.

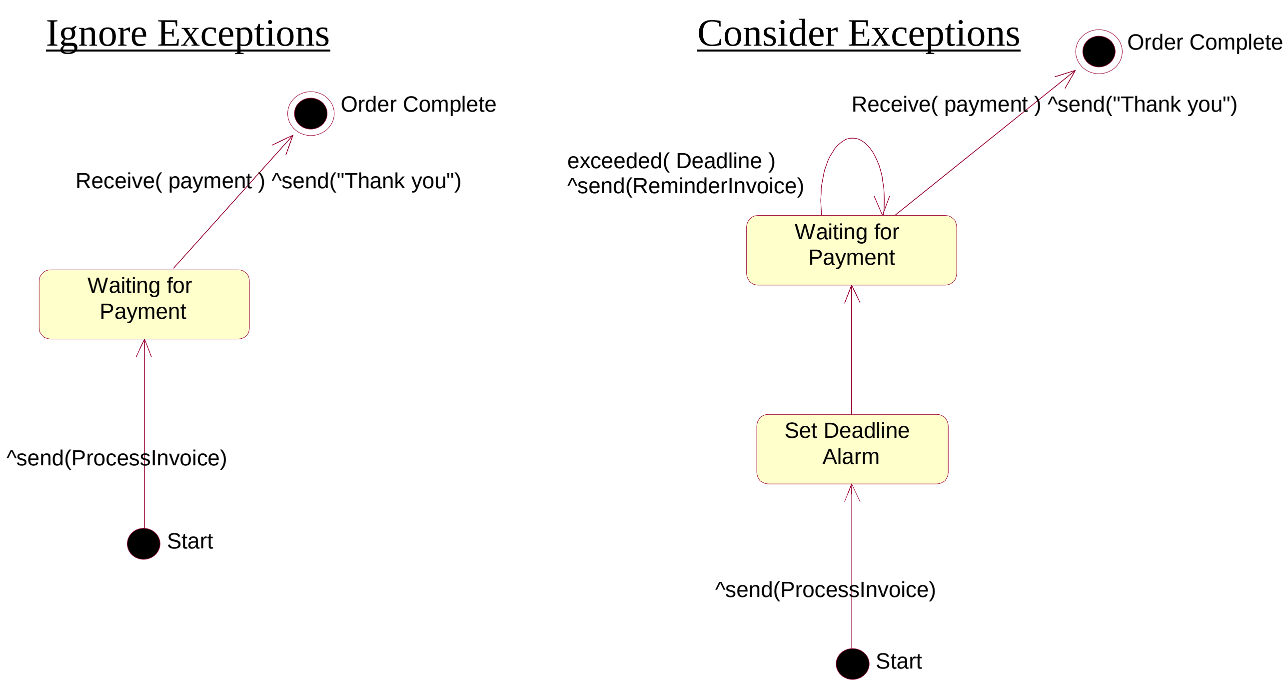

Figure 6: Agent descriptions can be automatically augmented to handle exceptions.

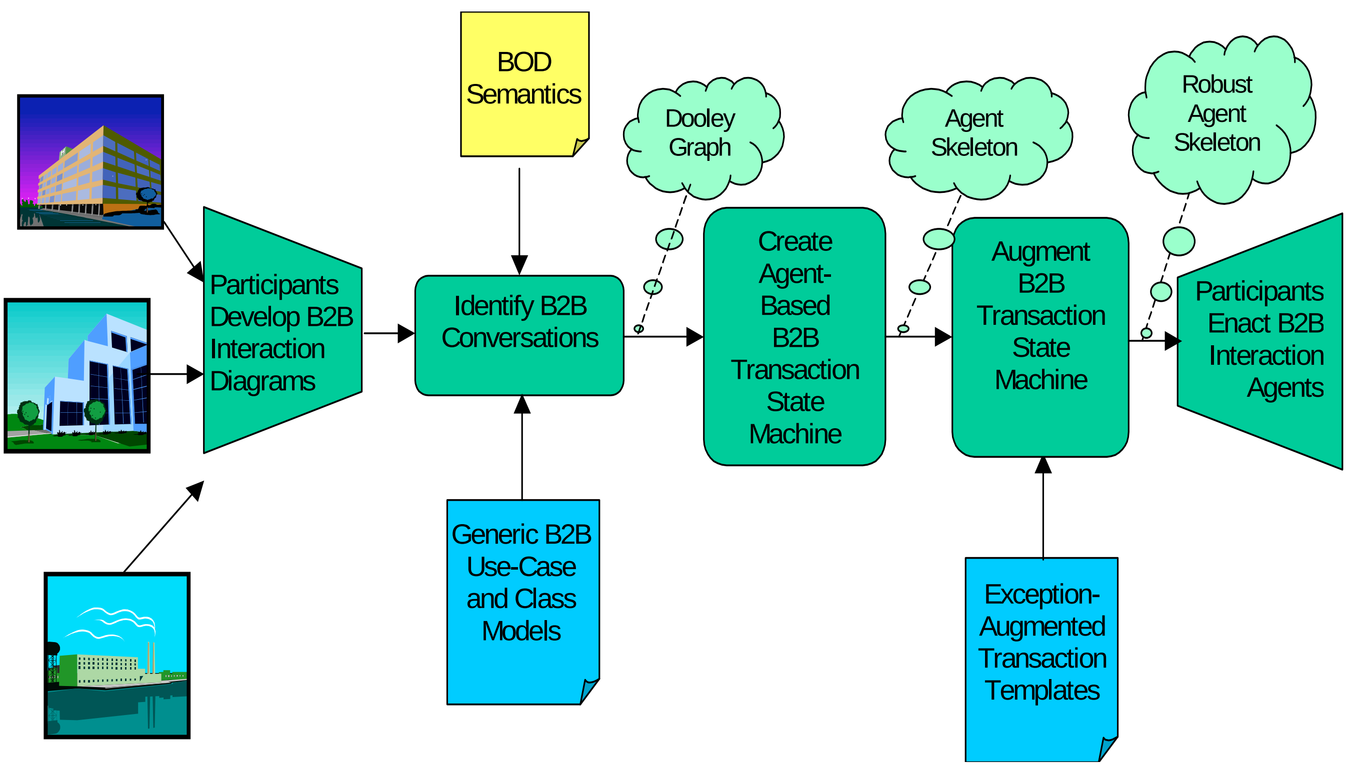

Figure 7: Agent-based coordination methodology for B2B automation.

Our investigation has considered a number of issues in the evaluation of SCM and B2B automation:

1. Investigation of approaches to coordination modeling. We selected an approach based on a linguistic analysis of conversations. The investigation included a proof-of-concept for a real-world supply-chain scenario.
2. Identification of example SCM standards to test the selected approach. We have identified Open Application Group (OAG) B2B standards as the most relevant for supply-chain management in the automotive and aerospace manufacturing areas. Two example SCM scenarios developed by Ford and Lockheed Martin were chosen from the OAG Vendor Challenge event. Our choice was based on the realism of these scenarios, which were developed by manufacturers.
3. Evaluation. The linguistic approach was formalized to capture SCM semantics, based on the scenarios of SCM interactions. A multistep, formal procedure was defined, and then evaluated on the selected SCM scenarios.
4. Prototyping of a computational procedure that demonstrates use of the approach on the example SCM standards. The prototype includes
   - 4.1. Representation of an interaction diagram in DAML-OIL (an emerging Semantic Web standard).
   - 4.2. Conversation Table Generator software, along with the rules for producing conversation tables.
   - 4.3. Collaboration Diagram (Dooley graph) Generator, which takes the output of the Conversation Table Generator and constructs intermediate graphs for conversation analysis.
   - 4.4. Formalization and classification of exceptions to normal behavior. A key contribution of this work is the framework in which business exceptions to "normal" behavior can be included and represented within our method. With this capability, an end-user has a means to represent the entire behavior of SCM roles in a uniform, repeatable way.
   - 4.5. Agent state-machine generator software that defines the behavior of individual roles involved in the SCM scenarios. It takes as input the collaboration diagrams and definition of exceptions to normal behavior and produces descriptions of behavior (i.e., state machines) for each participating role.

During the application of coordination modeling to handle real SCM standards and scenarios we have identified two technical issues. The first issue has to do with the adopted, foundational, semantic categories (i.e., respond, reply, resolve, and complete) that have been proposed as a basis for reasoning about interaction and coordination. We have discovered that the semantics of these categories are insufficient to capture some of the SCM details. The other issue has to do with the algorithm for generation of the collaboration diagrams that is based on these four categories and creates indeterminate situations in some cases. One way of dealing with these issues is through identifying a heuristic rule to constrain a space of possible behavior diagrams to those that are more manageable.

Convergence is occurring among the three major standards efforts: OAG, RosettaNet, and ebXML. Our results offer a prototyped coordination modeling approach for aiding the convergence.

5. CONCLUSIONS

The methodologies described here promote the interchange of standard business documents and compensate for exceptions that might occur during execution. Enterprises need only describe their supply processes using 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.

This investigation has also identified additional work that needs to be done in order to refine the methodology, demonstrate its utility, and foster its adoption. The additional work includes

- Encoding the semantics of the complete set of BODs in DAML-OIL and/or PSL
- Resolving technical issues in a Dooley graph representation of conversations
- Formulating the agent-based coordination methodology for additional scenarios, enabling the automation of a larger set of interactions among B2B participants
- Collaborating with OAG, RosettaNet, OASIS, ebXML, and other B2B interest groups.

The final result will provide a major benefit to industrial and commercial efficiency, as well as competitiveness in global markets. 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.

6. ACKNOWLEDGMENTS

We have been collaborating with Michael Gruninger and James G. Nell at the U.S. National Institute for Standards and Technology, which supported the work reported here. Certain commercial products are identified in this paper. These products were used only for demonstration purposes. This use does not imply approval or endorsement by NIST, nor does it imply that these products are necessarily the best available for the purpose.

7. 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, December 12, 2000 http://www.gtresearchnews.gatech.edu/newsrelease/CHAINR.html.

[2] Tim Berners-Lee, James Hendler, and Ora Lassila, "The Semantic Web," *Scientific American*, May 2001.

[3] C. Dellarocas and M. Klein, "A Knowledge-Based Approach for Designing Robust Business Processes," in *Business Process Management: Models, Techniques, and Empirical Studies*, W.van der Aalst et al., eds., Springer-Verlag Lecture Notes in Computer Science 1806, pp. 50-65, 2000.

[4] Chrysanthos Dellarocas, Mark Klein, and Juan Antonio Rodriguez-Aguilar, "An Exception-Handling Architecture for Open Electronic Marketplaces of Contract Net Software Agents," *Proceedings International Conference on Electronic Commerce*, ACM Press, October 2000

[5] Mark Greaves, Heather Holmback, and Jeffrey Bradshaw, "What Is a Conversation Policy?" *Proceedings of Workshop on Specifying and Implementing Conversation Policies*, Autonomous Agents '99, Seattle, WA, May 1999.

[6] Michael Gruninger, "PSL Interoperability Example," Technical Report, Dept. of Industrial Engineering, University of Toronto, December 1997.

[7] Michael Gruninger, "Classes of Goals and Constraints," Technical Report, Dept. of Industrial Engineering, University of Toronto, September 1999.

[8] Michael N. Huhns and Munindar P. Singh, "Managing Heterogeneous Transaction Workflows with Cooperating Agents," in *Agent Technology: Foundations, Applications and Markets*, N. R. Jennings and M. J. Wooldridge, editors, Springer-Verlag, Berlin, 1998, pp. 219-240.

[9] Michael N. Huhns and Larry M. Stephens, "Automating Supply Chains," *IEEE Internet Computing*, vol. 5, no. 4, July-August 2001, pp. 90-93.

[10] Nenad Ivezic, Mario Barbacci, Don Libes, Tom Potok, and John Robert, "An Analysis of a Supply Chain Management Agent Architecture," in *Proceedings 4th International Conference on Multiagent Systems*, IEEE Computer Society Press, July 2000, pp 401-402.

[11] Kuldeep Kumar, "Technology for Supporting Supply-Chain Management," *CACM*, vol. 44, no. 6, June 2001, pp. 58-61.

[12] Open Applications Group, "Plug and Play Business Software Integration," http://www.openapplications.org/downloads/whitepapers/whitepaperdocs/whitepaper.htm, 2000.

[13] Steve Osborn, "The Role of Agents in Business-to-Business (B2B) Electronic Commerce," www.agentlink.org, 2000.

[14] H. Van Dyke Parunak, "Visualizing Agent Conversations: Using Enhanced Dooley Graphs for Agent Design and Analysis," in *Proceedings 2nd International Conference on Multiagent Systems*, AAAI Press, 1996, pp. 275-282.

[15] Munindar P. Singh, "Synthesizing Coordination Requirements for Heterogeneous Autonomous Agents," *Autonomous Agents and Multi-Agent Systems*, Kluwer Academic Publishers, vol. 3, no. 2, June 2000, pp. 107-132.

[16] Peter Wegner, "Why Interaction is More Powerful Than Algorithms," *CACM*, Vol. 40, No. 5, May 1997, pp. 80-91.

Appendix A. B2B Scenario in DAML-OIL

```
<?xml version="1.0" ?>
<rdf:RDF
  xmlns:daml
="http://www.daml.org/2001/03/daml+oil#"
  xmlns:bod
="http://www.engr.sc.edu/research/cit/projects/
  DAML/BODs#">
<daml:Ontology rdf:about="">
  <rdfs:comment>
    Definition of OAG BODs for Ford scenario
  </rdfs:comment>
</daml:Ontology>
<bod:ProcessPO rdf:ID="BOD1">
    <bod:bodSender>Ford</bod:bodSender>
    <bod:bodReceiver>Jarvis
Tools</bod:bodReceiver>
</bod:ProcessPO>
<bod:ProcessPO rdf:ID="BOD2">
    <bod:bodSender>Ford</bod:bodSender>
    <bod:bodReceiver>Lubetec</bod:bodReceiver>
</bod:ProcessPO>
<bod:ProcessPO rdf:ID="BOD3">
    <bod:bodSender>Ford</bod:bodSender>
    <bod:bodReceiver>Efficient
Logistics</bod:bodReceiver>
</bod:ProcessPO>
<bod:AckPO rdf:ID="BOD1">
    <bod:bodSender>Jarvis Tools</bod:bodSender>
    <bod:bodReceiver>Ford</bod:bodReceiver>
    <bod:ackDecision>Accept</bod:ackDecision>
    <bod:ackStatus>Final</bod:ackStatus>
</bod:AckPO>
<bod:AckPO rdf:ID="BOD2">
    <bod:bodSender>Lubetec</bod:bodSender>
    <bod:bodReceiver>Ford</bod:bodReceiver>
    <bod:ackDecision>Accept</bod:ackDecision>
    <bod:ackStatus>Final</bod:ackStatus>
</bod:AckPO>
<bod:ShowShipment rdf:ID="BOD2">
    <bod:bodSender>Lubetec</bod:bodSender>
    <bod:bodReceiver>Ford</bod:bodReceiver>
</bod:ShowShipment>
<bod:ProcessInvoice rdf:ID="BOD2">
    <bod:bodSender>Lubetec</bod:bodSender>
    <bod:bodReceiver>Ford</bod:bodReceiver>
</bod:ProcessInvoice>
<bod:ShowShipment rdf:ID="BOD1">
    <bod:bodSender>Jarvis Tools</bod:bodSender>
    <bod:bodReceiver>Ford</bod:bodReceiver>
</bod:ShowShipment>
<bod:AckPO rdf:ID="BOD1">
    <bod:bodSender>Jarvis Tools</bod:bodSender>
    <bod:bodReceiver>Ford</bod:bodReceiver>
    <bod:ackDecision>Refuse</bod:ackDecision>
    <bod:ackStatus>Final</bod:ackStatus>
</bod:AckPO>
<bod:ProcessPO rdf:ID="BOD10">
    <bod:bodSender>Ford</bod:bodSender>
    <bod:bodReceiver>Greenfield</bod:bodReceiver>
</bod:ProcessPO>
<bod:AckPO rdf:ID="BOD10">
    <bod:bodSender>Greenfield</bod:bodSender>
    <bod:bodReceiver>Ford</bod:bodReceiver>
    <bod:ackDecision>Accept</bod:ackDecision>
    <bod:ackStatus>Final</bod:ackStatus>
</bod:AckPO>
<bod:ShowShipment rdf:ID="BOD10">
    <bod:bodSender>Greenfield</bod:bodSender>
    <bod:bodReceiver>Ford</bod:bodReceiver>
</bod:ShowShipment>
<bod:ProcessInvoice rdf:ID="BOD10">
    <bod:bodSender>Greenfield</bod:bodSender>
    <bod:bodReceiver>Ford</bod:bodReceiver>
</bod:ProcessInvoice>
<bod:ReceivePO rdf:ID="BOD3">
    <bod:bodSender>Efficient
Logistics</bod:bodSender>
    <bod:bodReceiver>Ford</bod:bodReceiver>
</bod:ReceivePO>
</rdf:RDF>
```