Commitment-Based Enhancement of E-Commerce Protocols

Pınar Yolum and Munindar P. Singh[*]
Department of Computer Science
North Carolina State University
Raleigh, NC 27695-7534, USA

{pyolum, mpsingh}@eos.ncsu.edu

Abstract

Protocols represent the allowed interactions among communicating components. Protocols are essential in electronic commerce to constrain the behaviors of autonomous entities. Traditional representations of protocols include the actions, but not their content, which limits their applicability in settings where autonomous entities must flexibly interact to handle exceptions and exploit opportunities. We develop a commitment-based representation, which provides a content to the protocols, enabling flexible execution. We show how an existing protocol can be systematically enhanced to yield a protocol that allows the given actions as well as other legal moves.

1 Introduction

Although many e-commerce protocols have been developed in recent years, limitations caused by their rigid specification have remained. Traditionally, protocols have been specified directly in terms of legal sequences of actions without considering the fundamental meaning behind the actions.

However, e-commerce protocols should not only constrain the actions of the participating components, but also accommodate the open, dynamic nature of e-commerce interactions:

- *Autonomy*. Components must retain their autonomy and be minimally constrained in their interactions, i.e., only to the extent necessary.
- *Exceptions*. Components must be able to modify their interactions to handle any unexpected conditions.
- *Opportunities*. Components should be able to take advantage of available opportunities to improve their choices or to simplify their interactions.

Traditional approaches are based on the study of network protocols, and lack the key abstractions to handle the above dynamic aspects of e-commerce. By contrast, we develop an agent-based approach that incorporates the key abstraction of *commitments*. *Agents* are persistent computations that can perceive, reason, act, and communicate. Agents can be autonomous and heterogeneous, and can represent different interacting components. The agents' communications affect and are affected by their commitments. The agents' commitments reflect the protocols they are following and the communications they have made.

By stepping through a running example, we first show the different interactions that can take place among the parties and then show how these interactions can be added to the original protocol to yield an enhanced protocol.

Section 2 reviews the keys concepts and challenges dealing with communication. Section 3 describes our proposed approach. Section 4 describes our contributions in relation to the most relevant literature.

2 Semantic Analysis

We now analyze the concepts and challenges underlying communication, especially with regard to the execution of activities. As a running example, we consider the NetBill protocol that has been developed to handle buying and selling of electronic goods, such as software and electronic documents over the Internet [10].

Example 1 As shown in Figure 1, the protocol starts with a customer requesting a quote for a particular good, followed by the merchant sending the quote. If the customer accepts the quote, then the merchant delivers the good and waits for an Electronic Payment Order (EPO). The good delivered at this point is encrypted, i.e., not usable. After receiving the

[*] This research was supported by the National Science Foundation under grants IIS-9529179 and IIS-9624425 (Career Award).

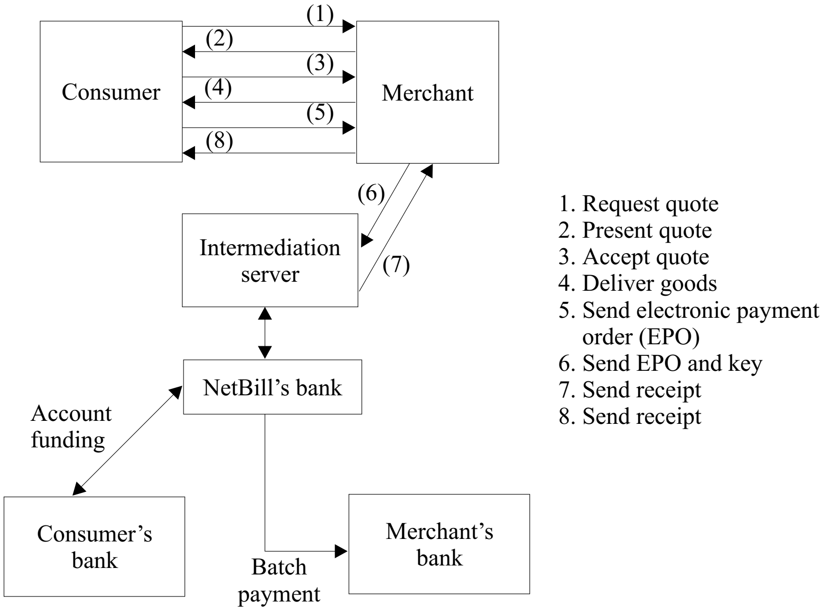

Figure 1. The NetBill payment protocol [10]

EPO, the merchant forwards the EPO and the key to the Intermediation Server, which then contacts the bank to take care of the banking process. Once the debit-credit operations are handled, the intermediation server sends a receipt back to the merchant, which contains the decryption key for the sold good. As the last step, the merchant forwards the receipt to the customer, who can successfully decrypt and use the good. ▮

For our present purposes, we are concerned neither about the details of the actual transactions that take place among the banks nor about the underlying security or encryption mechanisms. Therefore, we simplify the protocol to the point where we assume that once the merchant gets an EPO, he can take care of the banking services successfully. We use this simplified version of the protocol as our main example throughout this paper.

Example 2 Figure 2 shows the simplified version of the NetBill payment protocol as a finite state machine (FSM) labeled with the actions of merchant agent M and customer agent C. ▮

The participating parties in an e-commerce protocol are self-interested and eager to practice a variety of interactions to increase their personal benefit. Thus, in an e-commerce setting, parties should have several choices of actions and be able to practice the action that benefits them the most.

Example 3 The rigid specification in Figure 2 cannot handle some of the natural situations that arise in e-commerce:

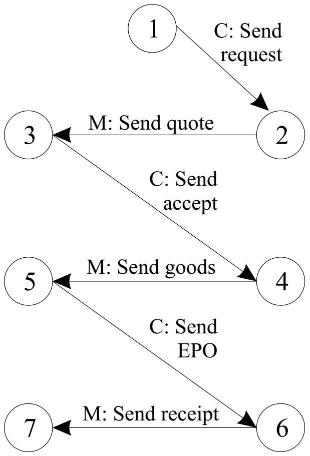

Figure 2. FSM representation of NetBill

- Instead of waiting for the customers to ask for quotes, the merchant may start the protocol by sending a quote, mimicking the idea of advertising.

- The customer may send an accept message without first exchanging explicit messages about the quote. This could very well reflect the level of trust between the parties. That is, the customer who trusts the merchant to give him the best quote may accept the quote without a prior announcement. Alternatively, this action could result from the customer's lack of interest in the quote, emergency of the transaction, insignificance of money, and so on.

- The merchant may send the goods without an explicit price quote. This could represent the trial versions in the software industry where after a certain period of time the customer is expected to pay to continue using the software.

■

3 Proposed Approach

Our proposed approach is as follows. We begin with the concept of *social commitments* from multiagent systems [2] and reasoning and dialogue [12]. Informally, a commitment is like an obligation from one party to another. A metacommitment is a commitment that refers to other commitments in a conditional form. Additional properties of commitments are discussed in [9, 11]; we lack the space to review those details here.

We propose a new formalism, *commitment-based finite state machine (CFSM)*, to be used in specification of protocols. Instead of specifying protocols merely in terms of legal sequences of actions, we define them with valid meanings based on commitments. This has the effect of enhancing a protocol to allow a broader range of interaction. As long as each state has a well-defined meaning, a variety of interactions can take place without violating the protocol. Thus, using this formalism, we show how the protocols can be enhanced to allow flexible execution.

3.1 CFSM

We define a CFSM conceptually as an FSM whose states and alphabet are given a semantic content, defined in terms of commitments. A CFSM can be specified by a three-tuple $M = \langle \mathbf{M}, \mathbf{\Sigma}, \mathbf{F} \rangle$ where M is a set of meanings, Σ is a set of actions defined in terms of commitments, and $\mathbf{F} \subseteq \mathbf{M}$ is the set of valid final meanings.

Example 4 We define the semantic content of each state in Figure 2 based on the participants' commitments.

- By sending a quote to a customer in state 3, the merchant commits to delivering goods and sending a receipt afterwards, if the customer promises to pay.

- By sending an accept to a merchant in state 4, the customer agrees to pay, only if the merchant promises to send a receipt afterwards.

- In state 5, the merchant fulfills first part of his promise by sending the goods.

- In state 6, the customer discharges his commitment of sending the goods.

- In state 7, the merchant discharges his commitment of sending the receipt.

These commitments arise from communications in conjunction with the metacommitments that are in force among the participants. ■

These informally defined meanings constitute the core meaning set, **M**, of the protocol. The action set, Σ, includes the actions used in Figure 2, defined in terms of commitments. The set of final meanings, **F**, contains all the states where no commitments are in force. In our example, this set contains two states. The first one is in effect when both parties make mutual commitments and fulfill their commitments. The second one arises, when one of the parties make a commitment that is not acknowledged by the other party. For instance, if the merchant makes an offer but the customer does not accept the offer, the protocol can end. The customer can either explicitly reject the offer, or remain silent. The latter case can be interpreted as rejecting after a certain time period. The formal definitions for **M**, Σ, and **F** are given in Section 3.2.

Unlike an FSM, the representation of a CFSM does not specify a starting state. The participants may start the protocol from a state where there are no commitments made or by accepting the commitments that are in force in that state. Again, unlike FSMs, the transitions between the states are not specified. The meanings of states are logically represented. Based on the intrinsic meaning of actions, the new state that is reached by performing an action at a particular state can be logically deduced. Thus, instead of specifying the sequences of actions that can be performed, we specify meanings that can be reached.

By specifying a protocol using a CFSM, we emphasize that the aim of executing a protocol is not to perform sequences of actions but to reach a state that represents the meaning of performing these sequences of actions. Once this fact is captured then we can come up with different paths that result in accomplishing the same goal as the original path. Thus, a protocol can be enhanced by finding alternative paths between states.

3.2 Formalization

Our formal language, $\mathcal{L}_c$ is based on the language $\mathcal{L}$ of propositional logic with the addition of a commitment operator to represent the commitments. Propositional logic is obviously too weak. However, the extension to temporal logic is straightforward [11].

The following Backus-Naur Form (BNF) grammar with a distinguished start symbol L gives the syntax of $\mathcal{L}$. $\mathcal{L}$ is based on a set of atomic propositions. Below, *slant* typeface indicates nonterminals; $\longrightarrow$ is a metasymbol of BNF specification; $\ll$ and $\gg$ delimit comments; and the remaining symbols are terminals.

- $L \longrightarrow$ *Prop* $\ll$atomic propositions$\gg$
- $L \longrightarrow \neg L$ $\ll$negation$\gg$
- $L \longrightarrow L \wedge L$ $\ll$conjunction$\gg$

We now define the syntax of the specification language, $\mathcal{L}_c$, through the following grammar whose start symbol is *Protocol*. The braces { and } indicate that the enclosed item is repeated 0 or more times.

- *Protocol* $\longrightarrow$ $\{Action\}$
- *Action* $\longrightarrow$ *Token: L*
- *Commitment* $\longrightarrow$ C_x (L)
- $L \longrightarrow$ *Commitment*

Example 5 Following Example 4, the messages can be given a content based on the following definitions:

- *request*: an atomic proposition meaning that the customer has requested a quote.
- *goods*: an atomic proposition meaning that the merchant has delivered the goods.
- *pay*: an atomic proposition meaning that the customer has paid the agreed amount.
- *receipt*: an atomic proposition meaning that the merchant has delivered the receipt.
- *promiseGoods*: an abbreviation for C_m[accept → goods] meaning that the merchant is willing to send the goods if the customer promises to pay.
- *accept*: an abbreviation for C_c[goods → pay] meaning that the customer is willing to pay if he gets the goods.
- *promiseReceipt*: an abbreviation for C_m[pay → receipt] meaning that the merchant is willing to send the receipt if the customer pays.
- *offer*: an abbreviation for (promiseGoods $\wedge$ promiseReceipt)

Based on these definitions, we can now formally define the protocol as a CFSM. For simplicity we place the content of an action in the state that results from it and since each action can be performed by only one party, we do not specify the performers explicitly.

- The set of meanings, **M**, contains the following:
  - State 1: `true`
  - State 2: request
  - State 3: offer
  - State 4: accept $\wedge$ offer
  - State 5: goods $\wedge$ $\mathsf{C}_c$pay $\wedge$ promiseReceipt
  - State 6: goods $\wedge$ pay $\wedge$ $\mathsf{C}_m$receipt
  - State 7: goods $\wedge$ pay $\wedge$ receipt
- The set of actions, Σ, contains the following:
  - t: request $\ll$Sending a request for quote$\gg$
  - q: offer $\ll$Sending a quote$\gg$
  - a: accept $\ll$Sending an accept$\gg$
  - g: goods $\wedge$ promiseReceipt $\ll$Delivering the good$\gg$
  - e: pay $\ll$Sending an EPO$\gg$
  - r: receipt $\ll$Sending the receipt$\gg$
- The set of final meanings, **F**, contains the states 1, 2, 3, and 7.

■

3.3 Application of CFSMs

The CFSM specification of a protocol can be used to execute the protocol at run time. Alternatively, using an automated tool, the CFSM specification of a protocol can be converted into a FSM specification at compile time. An agent can then use this FSM representation to execute the protocol at run time.

Run time In a CFSM specification, the states and the effects of performing actions are specified. Given a CFSM description of a protocol, an agent that can process logical formulae can compute the transitions between states. In this respect, the choice of actions is a planning problem for each agent. That is, from the possible end states, the agent first decides on the desired end state, and then logically infers a path that will take him from the current state to the desired end state.

Compile time To reduce the computations at run time, a CFSM can be converted into an FSM representation at compile time. This conversion can be done in several ways, one of which would be based on systematically producing paths and states to reach one of the final states. The meanings that have been already defined and each additional meaning captured in production of the path would map to a state in the FSM. The transitions between the states will follow from the valid paths. The set of final states would be all the states, including the newly produced states, that satisfy the

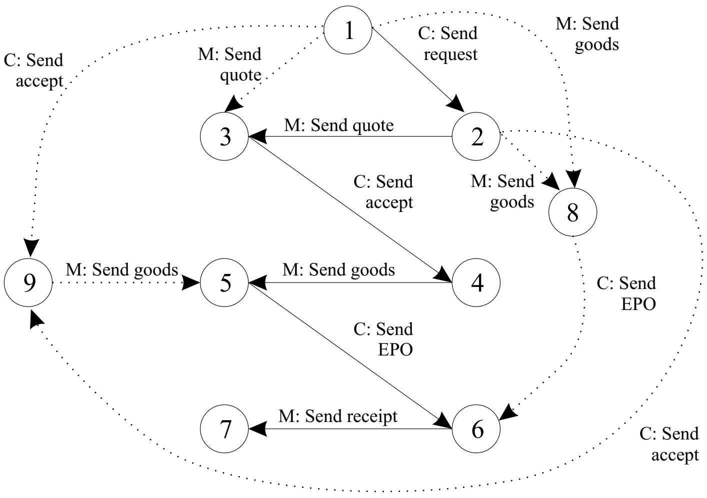

Figure 3. Enhanced FSM

ending constraints of CFSMs: any state where no commitments are in force, or any state where the metacommitment of a party has not been honored. The set of actions will remain the same. In general, we have the following development:

Let $X = \langle \mathbf{M}, \mathbf{\Sigma}, \mathbf{F} \rangle$ be a CFSM. Let $Y = \langle \mathbf{S}, \mathbf{\Sigma}, \mathbf{SInitial}, \mathbf{SFinal}, \delta \rangle$ be an FSM that is correct with respect to X.

Then Y may be constructed as follows:

- **S** = M
- $\mathbf{\Sigma}$ = Σ
- **SInitial** = $\{m_1 : (m_1 \in \text{M}) \wedge (m_1 = \mathsf{true})\}$
- **SFinal** = F
- $\delta = \{\langle m_i, s, m_j \rangle : m_i, m_j \in \text{M}, s \in \Sigma \wedge (m_i \not\models m_j \wedge m_i \models_s m_j \wedge (\forall\, m_k \in \text{M}\text{: } m_i \models_s m_k \wedge m_k \models m_j \rightarrow m_j{=}m_k))\}$

If FSM Y allows a computation, then CFSM X allows the same computation.

Example 6 Figure 3 shows the enhanced FSM that can be driven from the CFSM specification of our protocol. We now reconsider the shortcomings described in Example 3.

- The merchant can now start the protocol by sending a quote to a customer. As was the case, by performing this action he creates a metacommitment in state 3, namely that he is willing to send the goods and the receipt if the customer agrees to pay.

- By sending an accept message without prior conversation about the quote, the customer commits to paying if the merchant makes an offer. Thus, we move from a state where no commitments are made (state 1), to a state where the customer is making an offer (state 9). If the merchant does not make an offer, then the protocol ends at this point. On the other hand, if the merchant makes an offer by sending the goods, then both parties have to carry out their commitments; so the protocol moves to state 5.

- By sending the goods without an explicit accept message, the merchant makes an offer to send the receipt if the customer agrees to pay. Thus, we move from a state where no commitments are made (state 1), to a state where the merchant is making an offer (state 8). If the customer does not accept the offer, then the protocol ends at this point. Conversely, if the customer actually sends an accept message, then both parties need to fulfill their commitments: so the protocol moves to state 5.

Compared to the original version of the protocol in Figure 2, we have introduced two new states, state 8 and state 9. Using the definitions in Example 5, these states can be defined as follows:

- State 8: goods ∧ promiseReceipt
- State 9: accept

In addition to these states, we have added distinct links from state 1 to state 8, and 9; from state 2 to state 8 and 9; from state 8 to state 6 and from state 9 to state 5. The new transitions are shown with dashed lines. ▌

As we have demonstrated, specifying protocols using a CFSM enables the protocols to be executed flexibly, and thus improves the performance of the protocol drastically. At this point, it is important to restate what we mean by flexibility. Although we want the agents to practice a flexible protocol, we still want to preserve an ordering that will allow only meaningful conversations. For example, a merchant should not send a quote after sending the goods, or the customer should not start the conversation by sending an EPO. More importantly, flexibility can be introduced to the point where the intended meanings of the actions are preserved. When we allow more flexible interactions, we need to ensure that the original commitments are in force or they are altered by mutual agreement. Thus, if the applicable metacommitments are captured, the execution of a protocol is minimally constrained only to satisfy those metacommitments. This is a major advantage over low-level representations, which require specific execution sequences and provide no basis for deciding on the correct state independent of the execution sequence.

4 Discussion

We motivated a commitment-based treatment of protocol specification, analysis and execution. We developed a new way to specify protocols, CFSM, that is based on the meaning of actions rather than their sequences. The associated meanings are captured in terms of commitments. By walking through an example, we showed how we can define a CFSM specification, and how we can reason about the protocols and enhance them to achieve a flexible execution. We also pointed out how this specification can be realized at run time or compile time.

There is a substantial body of literature on ACLs and their semantics. The Foundation for Intelligent Physical Agents (FIPA) has been standardizing an ACL along with a formal semantics. FIPA also includes interaction protocols, which are characterized purely operationally. Labrou & Finin describe a grammar for constructing conversations or protocols [8].

Traditionally, communication protocols are specified by defining the allowed orders in which communicative acts may take place, but no more. This holds for protocol formalisms such as FSMs, push-down automata, formal grammars, Petri Nets, and temporal logic. Some of these formalisms can be quite powerful, but they are used only to specify allowed actions. The actions are just labels and the states, if explicit, do not capture the conceptual state of a protocol as we have attempted. By contrast, we use commitments to specify the communication protocols and thus analyze them through their intrinsic meaning. Commitments have been studied before [1, 3], but not used for protocol specification as here.

Most traditional deontic logics have had a single-agent focus and are therefore not suited for interoperation. Directed obligations, i.e., from one party to another, are more promising. Hohfeld's approach is an influential one in studies of law [6]. We previously showed that the sixteen main legal relations identified by Hohfeld can be captured using commitments [9]. Herrestad & Krogh propose a formalization of directed obligations that reduces them to preferences of the concerned parties [5]. By contrast, we treat commitments as first-class concepts and leave the preferences to additional inferences that can be made in some cases. Other formal research on interactions among agents includes Haddadi [4], who develops a formal semantics based on beliefs and intentions, but does not give an operational characterization as we do here.

References

[1] C. Castelfranchi. Commitments: From individual intentions to groups and organizations. In *Proceedings of the International Conference on Multiagent Systems*, pages 41–48, 1995.

[2] R. Conte and C. Castelfranchi. *Cognitive and Social Action*. UCL Press, London, 1995.

[3] L. Gasser. Social conceptions of knowledge and action: DAI foundations and open systems semantics. In *[7]*, pages 389–404. 1998. (Reprinted from *Artificial Intelligence, 1991*).

[4] A. Haddadi. Towards a pragmatic theory of interactions. In *[7]*, pages 443–449. 1998. (Reprinted from *Proceedings of the International Conference on Multiagent Systems, 1995*).

[5] H. Herrestad and C. Krogh. Obligations directed from bearers to counterparties. In *Proceedings of the 5th International Conference on Artificial Intelligence and Law*, pages 210–218, 1995.

[6] W. N. Hohfeld. *Fundamental Legal Conceptions as Applied in Judicial Reasoning and other Legal Essays*. Yale University Press, New Haven, CT, 1919. A 1919 printing of articles from 1913.

[7] M. N. Huhns and M. P. Singh, editors. *Readings in Agents*. Morgan Kaufmann, San Francisco, 1998.

[8] Y. Labrou and T. Finin. Semantics and conversations for an agent communication language. In *[7]*, pages 235–242. 1998. (Reprinted from *Proceedings of the International Joint Conference on Artificial Intelligence, 1997*).

[9] M. P. Singh. An ontology for commitments in multiagent systems: Toward a unification of normative concepts. *Artificial Intelligence and Law*, 7:97–113, 1999.

[10] M. A. Sirbu. Credits and debits on the internet. In *[7]*, pages 299–305. 1998. (Reprinted from *IEEE Spectrum, 1997*).

[11] M. Venkatraman and M. P. Singh. Verifying compliance with commitment protocols: Enabling open web-based multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 2(3):217–236, Sept. 1999.

[12] D. N. Walton and E. C. W. Krabbe. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning*. State University of New York Press, Albany, 1995.