Constitutive Interoperability

Amit K. Chopra
Department of Computer Science
North Carolina State University
Raleigh, NC 27695-8206, USA
akchopra@ncsu.edu

Munindar P. Singh
Department of Computer Science
North Carolina State University
Raleigh, NC 27695-8206, USA
singh@ncsu.edu

ABSTRACT

Commitments have recently emerged as a valuable abstraction for characterizing interactions among autonomous agents at the level of their business relationships. Traditionally, interoperation is approached from the standpoint of data exchange or of messaging. We use commitments to characterize interoperability in high-level terms: at the level of the communications among agents. Specifically, two agents are interoperable if their commitments align. Drawing upon Kant's famous distinction, we distinguish between two kinds of interoperability, constitutive and regulative. Constitutive interoperability takes into account solely the meaning of messages whereas regulative interoperability also takes into consideration message order, occurrence, and data flow. We present a language for specifying agents constitutively and a decision procedure for determining their interoperability.

Categories and Subject Descriptors

D.2.12 [**Software Engineering**]: Interoperability; I.2.11 [**Artificial Intelligence**]: Distributed artificial intelligence—*Multiagent systems*

General Terms

Verification

Keywords

Commitments, Business-level interoperability, Alignment

1. INTRODUCTION

Interoperability is a matter of *manifest agreement*. In other words, the interoperability of two or more parties means not only that there is an agreement among the parties but also that they can act according to the agreement. An *agent* is a computational representation of a "real" business principal. Agents interact with each other and their environment. We restrict attention to arms-length interactions in the form of *communications* among agents. These may be naturally realized in the computational infrastructure through messaging. For concreteness, we refer to the elements of communication as messages.

The scope of the agreement among the agents determines the scope of their interoperability. Communicating agents may thus interoperate at the level of text encoding (as in ASCII), syntax (as in XML InfoSet), grammar (as in UBL, the universal business language or more simply the specification of a purchase order), messaging (as in SOAP), terminology (as in the Dublin Core vocabulary), and so on. Effective interoperation among two or more agents presumes that they are interoperable at all relevant levels.

As agents communicate, they enter into *commitments* with one another. The commitments reflect the organizational or social relationships among the agents, and thus characterize their interactions at a high level. We propose a commitment-based theory of interoperability. This approach reflects the intuition that the most relevant—and least implementation dependent—kind of agreement is based on the commitments that the agents have toward one another. Thus, agents are deemed interoperable if they can enter into and maintain well-aligned commitments to each other. Commitments represent an essential level at which to assess and establish interoperability because they yield a notion of compliance eminently suitable for open settings: the principals may act as they please provided it is in accordance with their commitments. Ensuring or verifying that agents act according to their commitments is a different challenge [15].

Early in the study of software architecture, Parnas proposed that connectors between components should be treated not as control or data flow constructs but as *assumptions* made by each component about the others [12]. Arguably, much of the subsequent work on software architecture and interoperability regressed from Parnas' insight: it has primarily considered connectors and concomitant assumptions at the level of flow, e.g., dealing with message order and occurrence [11]. Such low-level criteria are largely orthogonal to considerations of business meaning. It is generally irrelevant whether the parties communicate via a procedure call or a message, and whether they follow a specific message order (unless the message order has a bearing on the meaning). Specifically, just because two agents are able to interact according to a specified *choreography* (i.e., a description of message ordering and occurrence) doesn't mean that their principals agree on the business meaning of the messages they exchange. Thus existing and emerging standards such as the Web Services Choreography Description Language (WS-CDL) [17] apply at too low a level of abstraction.

By contrast, commitments enable naturally expressing the assumptions that agents make of other agents regarding the business meanings of their interactions. What matters at the business level is what commitments exist, not what low-level means are used to create or manipulate a commitment. Of course, checking commitment-level interoperability does not obviate the need for checking the other kinds of interoperability, such as those alluded to above. But checking other kinds of interoperability is rarely adequate, and we need ways to define and check commitment-level interoperability, which is what this paper seeks to do.

Cite as: Constitutive Interoperability, Amit K. Chopra, and Munindar P. Singh, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. 797-804.

1.1 Commitments

Commitments help us address business level interoperability. For the purposes of this paper, a commitment can be thought of as a reified directed obligation. Commitments are directed from one agent (the *debtor*) to another (the *creditor*), and arise within a particular organizational context [14, 7]. When the condition of the commitment is met, the commitment is said to be discharged. In addition, a commitment may be operated upon, for example, by being delegated to a new debtor or assigned to a new creditor.

For us, interoperability is concerned with whether the agents involved can enter into and maintain well-aligned commitments with each other. Stated informally, this means that if an agent's state models a commitment of which the agent is the creditor, then the debtor's state must also model the same commitment. In other words, the debtor *covers* the creditor's assumption about the commitment. For example, let's say a customer takes a quote message to mean that the merchant commits to sending goods if the customer pays first, whereas the merchant takes it to mean no such commitment. This problem can arise in foreign exchange transactions as well [6]. The above illustrates commitment misalignment: on receiving the message, the customer's state models a commitment in which it is the creditor and the merchant the debtor, but the merchant's state does not reflect this commitment. They are thus noninteroperable.

The reverse condition—if a debtor's state models a commitment, then the creditor's state must also model the commitment—is of no relevance. An agent may adopt commitments towards other agents; however, if other agents do not expect it, those commitments are just harmless.

Our proposed definition of interoperability gives primacy to observations of each agent, i.e., the messages each sends and receives. We model communication between agents as being asynchronous and make only fundamental assumptions about it.

1.2 Commitment-Based Interoperability

We base our study of interoperability on Kant's distinction between constitutive and regulative rules, as developed by Searle [13]. In simple terms, a constitutive rule specifies what action counts as what. For example, raising your hand may count as bidding in an outcry auction, or offering to give an answer if you are a student in a class. In this case, bidding or offering to answer are *institutional* actions. Similarly, a judge's specific actions in the right context may count as creating a married couple. By contrast, a regulative rule constrains the performance of an action, e.g., that you cannot bid in an auction after a winner has been declared. In our approach, commitments are the key institutional facts, and the loci of institutional actions. Messages perform such actions by creating and manipulating commitments.

In our framework, message meanings are expressed as constitutive rules. The meaning of a message is specified in terms of its effects. The meaning may directly refer to commitments or indirectly affect commitments, as when the message counts as bringing about a condition of the commitment. For example, a price quote may constitute an offer to sell, treated as a (conditional) commitment. Each agent is described via its *constitutive specification*, which serves as an interface describing its assumptions. In essence, the constitutive specification of an agent tells us the meanings of the messages that the agent (presumably) respects. The intuition behind constitutive interoperability is that, in order to interoperate, the agents ought to agree about the institutional reality in which they exist. In other words, the agents agree on what their communications count as.

Constitutive interoperability is determined from constitutive specifications, that is, specifications that consist only of constitutive rules. If the interacting agents happen to apply mutually inconsistent constitutive rules, they would fail to interoperate. The above example where quote means different things to the customer and the merchant shows a violation of constitutive interoperability.

Message occurrence (when a particular message must be sent), ordering between the sends and receives of messages, and data flow among the messages are all regulative rules. A regulative specification may be viewed as encoding an agent's policies. For example, the merchant may have a regulatory rule that the customer must pay first in order for shipment to proceed. Regulative interoperability is determined from regulative specifications, that is, specifications that consist of regulative rules.

This paper concentrates on constitutive interoperability. In earlier work, we used $C+$ [10], an action description language, to specify and reason about protocols [5]. Here we employ a simpler language that is adequate for expressing constitutive rules and for reasoning about interesting cases of constitutive interoperability.

1.3 Contributions and organization

Our contributions in this paper are:

- A high-level definition of constitutive interoperability that takes into account the business meaning of communication, and that supports asynchronous communication.
- A language for constitutive specifications and a decision procedure for determining the constitutive interoperability of pairs of agents. A benefit of this approach is that it operates by program analysis rather than by building potentially large transition systems.

Section 2 presents our technical framework. Section 3 formalizes constitutive interoperability and provides a decision procedure for the same. The correctness proof is also provided. Section 4 places this work in the context of the literature.

2. TECHNICAL FRAMEWORK

The framework consists of a language for constitutive specifications, an operational model of asynchronously communicating agents, and the formal semantics of the language in terms of the model.

2.1 Constitutive Specifications

Below, m_i range over messages; x, y, ... range over agents; p, q, ... range over propositions or Boolean formulas over them; $\top$ is the constant for truth; α is a propositional literal or its negation: identify α with $\neg\neg\alpha$. A commitment is a propositional letter. A commitment $\mathsf{C}(x, y, p, q)$ means that x is committed to y to bring about *condition* q if *precondition* p comes about.

Let's define our formal language via the following Backus-Naur Form productions. L is the starting symbol of our formal language. Below, Φ is a set of atomic propositions, $\mathcal{X}$ is a set of agent names, and *Message* names a message. We simplify the syntax by eliding parameters to concentrate on the points of interest here.

- $L \longrightarrow \{$*Message* means *Clause* $\}$
- *Clause* $\longrightarrow$ *Conjunction* $\mid$ *Commitment*
- *Commitment* $\longrightarrow \mathsf{C}(\mathcal{X}, \mathcal{X},$ *Conjunction*, *Disjunction*$)$
- *Conjunction* $\longrightarrow \Phi \mid \Phi \wedge$ *Conjunction*
- *Disjunction* $\longrightarrow \Phi \mid \Phi \vee$ *Disjunction*

As described in the above grammar, we restrict the precondition and condition of a commitment to be a conjunction and disjunction of propositional literals, respectively. This simplifies the presentation of the decision procedure for interoperability without a loss of expressiveness. For example (omitting agent names in commitments), m means $\mathsf{C}(p \vee q, r \wedge s)$ may be expressed as the four rules m means $\mathsf{C}(p, r)$, m means $\mathsf{C}(p, s)$, m means $\mathsf{C}(q, r)$, and m means $\mathsf{C}(q, s)$. Our grammar places an additional restriction that commitments may not be nested.

From a technical standpoint, an agent x's constitutive specification, $\mathbb{C}_x$, is a finite set of rules, each of the form of Schema 1.

SCHEMA 1. m means p

The idea behind Schema 1 is to capture the *counts as* relationships that describe the institutional meanings of messages. In Schema 1, the head p is a conjunction of propositional letters, and the body m is an action corresponding to a single message. When p is a commitment, the constitutive rule describes the creation of a commitment. Table 1 shows the constitutive specifications of a customer and merchant.

Table 1: Constitutive specifications of a customer and merchant

customer (c)
$Offer(m, c)$ means $\mathsf{C}(m, c, pay, goods)$
$Pay(c, m)$ means pay
$Goods(m, c)$ means $goods$
merchant (m)
$Offer(m, c)$ means $\mathsf{C}(m, c, pay, goods)$
$Pay(c, m)$ means pay
$Goods(m, c)$ means $goods$

2.2 Modeling Communicating Agents

This paper goes beyond existing formalizations of commitment protocols. It applies the language introduced above from the perspective of each agent. That is, each agent maintains its own theory of the world.

We write the actions of sending and receiving a message m as $!m$ and $?m$, respectively. Each message is uniquely identified, and has exactly one sender and one receiver. Where the sender and receiver are relevant, m is expanded to $m(x, y)$ to indicate a message m from x to y. Let $\mathcal{A}$ be a multiagent system. We restrict the formulation of constitutive interoperability to systems that have only *two agents*. We abuse notation in places in letting $\mathcal{A}$ denote a set of agents.

An agent's observations are limited to the messages it sends or receives. Thus agent x makes observations of the form $!m(x, y)$ or $?m(y, x)$, for a message m sent or received from some agent y. We assume that each agent has a single input queue and thus its observations are a sequence of messages sent or received (one at a time). O_x, a finite sequence of observations of agent x is given by a list $\langle o_0, o_1, \ldots, o_n \rangle$.

To simplify the technical development, we assume that messages are not created or lost by the infrastructure. Thus if the sender observes $!m$, the recipient will eventually observe $?m$, and if a recipient observes $?m$ the sender must already have observed $!m$. Further, any two messages sent to the recipient by the same sender arrive in the order in which they are sent. We term these the *fundamental constraints* on messaging. Even so, in general, agents would make distinct observations because they send and receive different messages. More importantly, even in a two-party system and even if we neglect message direction, the agents may observe the same messages in different orders.

An *observation vector* for a multiagent system is a vector each of whose elements is an observation sequence, one for each agent in the system. Observation vectors satisfy the fundamental constraints on messaging: no messages are lost and messages arrive in order. But the observation sequences could be incomplete: thus not every message that has been sent would have been received. However, if a message from x is received by y, then that message must have been sent by x, and all messages previously sent by x to y would have been previously received by y. In other words, an observation vector describes a (partial) execution of the system.

DEFINITION 1. $\mathbb{O} = [O_x, O_y]$ *is an* observation vector *over the agents x and y provided O_x and O_y are observation sequences of x and y, respectively.*

- *If $?m(x, y)$ occurs in O_y, then $!m(x, y)$ occurs in O_x*
- *If $?m_1(x, y)$ occurs in O_y and $!m_0(x, y)$ precedes $!m_1(x, y)$ in O_x, then $?m_0(x, y)$ precedes $?m_1(x, y)$ in O_y*

A system is said to be in observational *quiescence* for a partial execution where no messages are in transit—all sent messages have been received. (Any or all agents could be computing even if no messages are in transit, but this is not relevant here.) Definition 2 formalizes this intuition.

DEFINITION 2. *An observation vector $\mathbb{O}_Q = [O_x, O_y]$ is* quiescent *provided if $!m(x, y)$ occurs in O_x, then $?m(x, y)$ occurs in O_y, and $!m(y, x)$ occurs in O_y, then $?m(y, x)$ occurs in O_x.*

Below, we apply the subscript Q to an observation vector (as in $\mathbb{O}_Q$) to say it is a quiescent vector. $\mathcal{O}_{\mathcal{A}}$ is the set of all possible observation vectors for system $\mathcal{A}$.

An observation sequence O_x may be a prefix of another (i.e., intuitively later) observation sequence O'_x. This is written $O_x \preceq O'_x$. The definition of prefix expands to apply to vectors in the obvious manner.

DEFINITION 3. $\mathbb{O} = [O_x, O_y] \preceq \mathbb{O}' = [O'_x, O'_y]$ *iff* $(\forall z \in \{x, y\} : O_z \preceq O'_z)$.

A simple consequence of the assumption of no messages being lost is that if $!m(i, j)$ occurs in $\mathbb{O}_i$, then there exists $\mathbb{O}'$ such that $\mathbb{O} \preceq \mathbb{O}'$ and $?m(i, j)$ occurs in $\mathbb{O}'_j$.

We require that communicating agents have standard names and that their vocabularies are aligned. Thus we can talk coherently of the commitments in which each agent features. Specifically, if x and y are agents, and x refers to $\mathsf{C}(y, x, p, q)$, then we can compare this to $\mathsf{C}(y, x, p, q)$ as referred to by y. This assumption is not fundamental but simplifies our exposition.

2.3 Operational Semantics

Let x be an agent and $\mathbb{C}_x$ its constitutive specification. The formula $\langle m_0, \ldots, m_n \rangle \Vdash_x p$ means that the state of x after having observed $\langle m_0, \ldots, m_n \rangle$ models the proposition p. (Below, $p \vdash q$ means that we can derive q from p. When p and q are propositions, $\vdash$ is Boolean consequence.) Thus, $\Vdash_x$ is closed under the following rules of inference.

UNIT says that a message (by itself) always brings about the head of its defining constitutive rule.

$$\frac{m \text{ means } p \in \mathbb{C}_x}{\langle m \rangle \Vdash_x p} \qquad (\text{UNIT})$$

PROP states that a proposition (that does not derive any commitments) holds if brought about by a message.

$$\frac{\langle m_n \rangle \Vdash_x p \qquad p \nvdash \mathsf{C}(r, s)}{\langle m_0, \ldots, m_n \rangle \Vdash_x p} \qquad (\text{PROP})$$

HOLD states that a message that means a commitment brings it about unless the condition of the commitment holds simultaneously. The condition would cause the commitment to discharge. Thus a commitment may result only if it has not already and is not concurrently discharged (see below). A special case of this rule is when the precondition is $\top$.

$$\frac{\langle m_0,\ldots,m_n\rangle \Vdash_x \neg q \qquad \langle m_n\rangle \Vdash_x \mathsf{C}(p,q)}{\langle m_0,\ldots,m_n\rangle \Vdash_x \mathsf{C}(p,q)} \qquad (\text{HOLD})$$

DETACH explains the consequences of a commitment and its precondition holding simultaneously. A stronger commitment, namely, with a precondition of $\top$ comes to hold.

$$\frac{\langle m_0,\ldots,m_n\rangle \Vdash_x p \wedge \neg q \wedge \mathsf{C}(p,q) \qquad p \not\equiv \top}{\langle m_0,\ldots,m_n\rangle \Vdash_x \mathsf{C}(\top,q)} \qquad (\text{DETACH})$$

SAT explains the satisfaction or discharge of a commitment. When the condition of a commitment holds, the commitment is discharged and is thus active no more.

$$\frac{\langle m_n\rangle \Vdash_x q}{\langle m_0,\ldots,m_n\rangle \Vdash_x \neg\mathsf{C}(p,q)} \qquad (\text{SAT})$$

WEAKEN states that $\Vdash_x$ is closed under propositional derivation given by $\vdash$, as mentioned above.

$$\frac{\langle m_0,\ldots,m_n\rangle \Vdash_x p \qquad p \vdash q}{\langle m_0,\ldots,m_n\rangle \Vdash_x q} \qquad (\text{WEAKEN})$$

NEG states that $\Vdash_x$ deals with binary logic.

$$\frac{\langle m_0,\ldots,m_n\rangle \nVdash_x p}{\langle m_0,\ldots,m_n\rangle \Vdash_x \neg p} \qquad (\text{NEG})$$

INERTIA says that if an atomic proposition α holds and is not overturned by the next messaging action, then α continues to hold.

$$\frac{\langle m_0,\ldots,m_{n-1}\rangle \Vdash_x \alpha \qquad \langle m_n\rangle \nVdash_x \neg\alpha}{\langle m_0,\ldots,m_n\rangle \Vdash_x \alpha} \qquad (\text{INERTIA})$$

CMT describes the consequence relation between commitments. Of two commitments, the stronger commitment is the one whose precondition is weaker or condition is stronger.

$$\frac{\mathsf{C}(x,y,p_0,q_0) \qquad p_1 \vdash p_0 \qquad q_0 \vdash q_1}{\mathsf{C}(x,y,p_1,q_1)} \qquad (\text{CMT})$$

Thus CMT captures our intuition about covering the assumptions of agents. Let c_0 and c_1 be commitments. We say $c_0 \vdash c_1$—read as c_0 *covers* the assumptions of c_1—if and only if they have the same debtor and creditor, c_1's precondition is stronger than c_0's, and c_1's condition is weaker than c_0's. For an example where the precondition is stronger, consider a customer c's commitment $c_0 = \mathsf{C}(c, m, goods, pay)$ to a merchant m that c will pay if m sends the goods. Let's say the merchant assumes the commitment $c_1 = \mathsf{C}(c, m, goods \wedge receipt, pay)$ from the customer instead. $c_0 \vdash c_1$ (c_0 covers c_1) because the customer's precondition is weaker than the merchant's. For an example where the condition is weaker, consider the customer's commitment $c_0 = \mathsf{C}(c, m, goods, pay)$. Let the merchant's assumption be $c_1 = \mathsf{C}(c, m, goods, pay \vee return)$. Clearly, $c_0 \vdash c_1$ because now the merchant's condition is weaker than the customer's.

LEMMA 1. *$\Vdash_x$ terminates.*

PROOF. (*Sketch*) Each inference rule for $\Vdash_x$ either reduces the sequence length or the depth of the formula being considered. □

3. CONSTITUTIVE INTEROPERABILITY

First we present the definition of constitutive interoperability and then a decision procedure for determining it from agents' specifications. We present numerous examples along the way to illuminate the concepts involved.

3.1 Definition

Interoperability considers the prospect of interoperation. For this reason, it considers all possible enactments in which the agents may participate. Constitutive interoperability means that the agents would agree about whatever commitments as might result from any messages they might exchange. Thus it considers all potential observations of all agents, and fails if even one set of potential observations would cause failure of interoperation. Definition 4 considers only observations of creditors and debtors from the same quiescent vectors.

DEFINITION 4. *$\mathcal{A}$ is C-interoperable (written $[\![\mathcal{A}]\!]$) iff*

$$\forall \mathbb{O}_Q \in \mathcal{O}_{\mathcal{A}} : (\forall x, y : [O_x, O_y] = \mathbb{O}_Q :$$

$$(O_y \Vdash_y \mathsf{C}(x,y,p,q)) \rightarrow (O_x \Vdash_x \mathsf{C}(x,y,p,q)))$$

The idea is that if y's observations model, under its constitutive specification, that y is the creditor of commitment c, then any observations made by the debtor x must model, under its constitutive specification, that x is the debtor of c.

The above definition considers only quiescent vectors. The motivation for doing so is to provide an opportunity for the agents to "sync" up. In a distributed systems, the agents would in general observe different messages. Requiring that they agree upon their commitments without observing the same messages would in general lead to such a strong definition that would fail even when our intuition would be that the agents interoperate.

For example, say a merchant sends an offer to a customer that states: if you send me the payment, I will send you the goods. Say the customer receives this offer and sends the payment to the merchant. At this point, the customer has no knowledge of when the payment will arrive at the merchant; the merchant has no knowledge that the payment has been sent. The customer's observations legitimize the unconditional commitment on part of the merchant to send it the goods. The merchant's observations do not. The above definition is not affected by this apparent discrepancy because it is only a transient discrepancy. When the payment arrives at the merchant, the commitment as expected by the customer will be covered by the merchant.

3.2 Decision Procedure

We introduce unique labels for the rules in a constitutive specification for easy reference in the text. For example, in $A = m \,\mathsf{means}\, p$, the label of the rule $m \,\mathsf{means}\, p$ is A, and we say that A *is* the rule $m \,\mathsf{means}\, p$). Even though two rules in different constitutive specifications may have the same label, we use different labels throughout to avoid confusion.

Let A be a constitutive rule. $\hat{A}$ and $\breve{A}$ denote the head and body of A, respectively. Given a set of rules $\mathbb{A}$, $\widehat{\mathbb{A}} = \bigwedge_{A \in \mathbb{A}} \hat{A}$. To simplify the presentation, we introduce the empty rule ϵ as a member of every constitutive specification and its head as $\top$.

The intuition behind our decision procedure is to verify that each rule in the constitutive specification of an agent that would cause an agent to have a credit (a commitment in which the agent is the creditor) should be covered by a rule in the debtor's constitutive specification.

EXAMPLE 1. *In Table 2, rule Cus_1 which encodes the customer's assumption about an offer is supported by Mer_1 in the merchant's specification, that is, $Mer_1 \vdash Cus_1$. In fact, the merchant's commitment to send a receipt is not an assumption of the customer.* ▮

EXAMPLE 2. *With reference to Table 2, if the merchant's specification had $Mer_2 = Offer$ means $\mathsf{C}(m, c, pay, receipt)$ instead of Mer_1, then $Mer_2 \nvdash Cus_1$ and when the $Offer$ message is exchanged, it will cause a commitment misalignment.* ▮

Table 2: Offer

customer (c)
$Cus_1 = \mathit{Offer}$ means $\mathsf{C}(m, c, pay, goods)$
merchant (m)
$Mer_1 = \mathit{Offer}$ means $\mathsf{C}(m, c, pay, goods \wedge receipt)$

There is an additional caveat though: the agents must also agree on the messages that affect a commitment. This means that the decision procedure must check the agents' specifications for the respective compatibility of the rules that bring about the precondition and condition of a commitment. Specifically, the debtor should cover the ways in which the creditor may bring about the precondition, and the creditor should cover all the ways in which the debtor may bring about the condition. In Table 2 there are no such rules, hence the agents vacuously agree.

EXAMPLE 3. *Let's consider the agents in Table 3. Clearly, $Mer_3 \vdash Cus_2$. In addition, the merchant covers all the ways in which a customer expects to make a payment (Cus_3 is covered by Mer_4) therefore ensuring that when the customer pays, the merchant understands that. Similarly, the customer understands all the ways the merchant can cause the $goods$ condition to hold (Mer_6 is covered by Cus_4).* ▮

Table 3: Offer with precondition and condition rules

customer (c)
$Cus_2 = \mathit{Offer}$ means $\mathsf{C}(m, c, pay, goods)$
$Cus_3 = \mathit{PayCash}$ means pay
$Cus_4 = \mathit{GoodsShip}$ means $goods$
$Cus_5 = \mathit{GoodsExpedited}$ means $goods$
merchant (m)
$Mer_3 = \mathit{Offer}$ means $\mathsf{C}(m, c, pay, goods)$
$Mer_4 = \mathit{PayCash}$ means pay
$Mer_5 = \mathit{PayCredit}$ means pay
$Mer_6 = \mathit{GoodsShip}$ means $goods$

EXAMPLE 4. *Referring to Table 3, suppose that the merchant did not have the rule Mer_4 meaning she only accepts credit cards. Then upon doing $PayCash$, the customer's state would model the commitment $\mathsf{C}(m, c, \top, goods)$ (because of* DETACH*); however the merchant's state would not model $\mathsf{C}(m, c, \top, goods)$ upon receiving $PayCash$. Hence, interoperability fails.* ▮

Definitions 5 and 6 introduce the machinery necessary to formalize this caveat. Definition 5 introduces the notion of a *precondition predecessor*. Let $A = m$ means $\mathsf{C}(x, y, p, q) \in \mathbb{C}$. A precondition predecessor of A is a subset of $\mathbb{C}$ such that the rules in the predecessor explain the causation of each propositional letter in p. The set of all precondition predecessors of an rule A is denoted by $\mathbb{P}_A$.

EXAMPLE 5. *In Table 3, the commitment in Cus_2 has only one propositional letter pay. A precondition predecessor of Cus_2 is $\{Cus_3\}$. Mer_3 has two precondition predecessors: $\{Mer_4\}$ and $\{Mer_5\}$. Thus, $\mathbb{P}_{Cus_2} = \{\{Cus_3\}\}$ and $\mathbb{P}_{Mer_3} = \{\{Mer_4\}, \{Mer_5\}\}$.* ▮

EXAMPLE 6. *In Table 2, $\mathbb{P}_{Cus_1} = \{\}$, $\mathbb{P}_{Mer_1} = \{\}$.* ▮

No subset of a precondition predecessor of a rule A should itself be a precondition predecessor of A because that means the former contains rules not relevant to the precondition. What is not relevant to a commitment will necessarily have no effect on commitment-level interoperability.

DEFINITION 5. *Let $A = m$ means $\mathsf{C}(x, y, p, q)$ be a rule in $\mathbb{C}$. Then the* precondition predecessors *of A in $\mathbb{C}$ denoted by $\mathbb{P}_A$ is defined as*

$$\{\Delta | \Delta \subseteq \mathbb{C} : (\widehat{\Delta} \equiv p \text{ and } \neg(\exists \Delta' : \widehat{\Delta}' \equiv p \text{ and } \Delta' \subset \Delta))\}$$

Recall that the condition of a commitment is a disjunction of propositional literals. Bringing any one of those about satisfies the commitment. Unlike the precondition predecessor which is a set of rules, a condition predecessor is a single rule whose head derives at least one propositional literal in the condition.

DEFINITION 6. *Let $A = m$ means $\mathsf{C}(x, y, p, q)$ be a rule in $\mathbb{C}$ where q is the disjunction $q_0 \vee \ldots \vee q_n$ of propositional letters. The* condition predecessors *of A in $\mathbb{C}$ denoted by $\mathbb{C}_A$ is defined as*

$$\{R | R \in \mathbb{C} \text{ and } \exists q_i\ (0 \leq i \leq n)\ : \hat{R} \vdash q_i\}$$

EXAMPLE 7. *In Table 2, $\mathbb{C}_{Cus_1} = \{\}$, $\mathbb{C}_{Mer_1} = \{\}$.* ▮

EXAMPLE 8. *Referring to Table 3, $\mathbb{C}_{Cus_2} = \{Cus_4, Cus_5\}$, $\mathbb{C}_{Mer_3} = \{Mer_6\}$.* ▮

Definition 7 finally puts together all the elements discussed above in defining the *complete coverage* of a rule that causes a commitment credit. For such a rule to be completely covered, the following conditions must be satisfied:

1. *Rule coverage*: The debtor must cover the rule: a credit represents an assumption of the creditor.
2. *Precondition coverage*: The debtor must cover all the ways in which the creditor may bring about the precondition of the commitment—these represent the assumptions of the creditor. Additionally, it means that if the creditor expects to deal in n distinct messages to bring about the precondition of a commitment, then the debtor's cover cannot involve more than those n messages.
3. *Condition coverage*: The creditor must cover all the ways in which the debtor may bring about the condition of the commitment—these represent the assumptions of the debtor. This ensures that any message that can discharge a commitment on the debtor's side will also discharge the commitment on the creditor's side.

DEFINITION 7. *Let $\mathbb{C}_x$ and $\mathbb{C}_y$ be the constitutive specifications of agents x and y, respectively. Let $E \in \mathbb{C}_y$ be m means $\mathsf{C}(x, y, p, q)$. E is* completely covered, *denoted by $\lfloor E \rfloor$ iff E is covered, that is, $\exists M \in \mathbb{C}_x : (M \vdash E)$ and the following hold:*

- *Precondition coverage:* $\forall \mathbb{S} \in \mathbb{P}_E : \exists \mathbb{V} \in \mathbb{P}_M : (\bigcup_{A \in \mathbb{V}} \breve{A}) \subseteq (\bigcup_{A \in \mathbb{S}} \breve{A})$
- *Condition coverage:* $\forall S \in \mathbb{C}_M : \exists V \in \mathbb{C}_E : \breve{S} \equiv \breve{V}$

In Definition 7 above, each $\mathbb{S}$ referred to in the *precondition coverage* clause represents a "way" (assumption) of the creditor that must be covered by the debtor. Similarly, each S in the *condition coverage* clause represents a "way" of the debtor that must be covered by the creditor.

DEFINITION 8. *Let $\mathcal{A}$ be a two agent system. $\mathcal{A}$ is* compatible, *denoted by $[[\mathcal{A}]]$, iff*

$$\forall y \in \mathcal{A} : \forall E = m \text{ means } \mathsf{C}(x, y, p, q) \in \mathbb{C}_y :\ \lfloor E \rfloor$$

Algorithm 1: Algorithm for determining $[[\mathcal{A}]]$

```
1  foreach (agent $y$ in a two agent system) do
2      foreach ($E$ in $y$'s specification which means a credit $\mathsf{C}(x, y, p, q)$ for $y$) do
3          if (there exists an $M$ in $x$'s specification such that $M$ covers $E$) then
4              foreach (way in which $y$ assumes $p$ can hold) do
5                  if ($x$ covers that way) then
6                      continue;
7                  else
8                      return false;
9              foreach (way in which $x$ assumes $q$ can hold ) do
10                 if ($y$ covers that way) then
11                     continue;
12                 else
13                     return false;
14         else
15             return false;
16 return true;
```

Algorithm 1 is pseudo code for Definition 8.

Figure 1 and 2 show the program analysis graphs for the agents in Table 2 and 3, respectively. A program analysis graph is constructed as follows. For each agent, for each rule E in which a credit is created, create a circle labeled with E. Denote each of its precondition predecessors by a dotted box connected to the circle by an arrow labeled P. Label the box with the rules in that precondition predecessor. Denote each of its condition predecessors by a dotted box connected to the circle by an arrow labeled with C. If a rule M in the other agent covers E ($M \vdash E$), then create a circle labeled M and connect M to E with an arrow directed towards E. Indicate M's precondition and condition predecessors as described for E. If there exists a precondition predecessor of M that covers one of E, draw a directed arrow from the former to the latter. Similarly, if there exists a condition predecessor of E that covers one of M, draw a directed arrow from the former to the latter. If at the end of this graph construction, E is not connected to some M, or if one of E's precondition predecessors is not connected to some of M's, or if one of M's condition predecessors is not connected to some of $E's$, then the agents are not compatible.

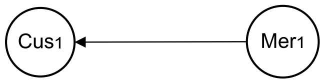

Figure 1: Program analysis graph for agents in Table 2

EXAMPLE 9. *Consider the agents in Table 4. Here the merchant ships goods to customers of legal age only. However, she accepts payment by credit card to be proof of legal age. The customer, however, provides her birth date as proof of legal age. First, $Mer_7 \vdash Cus_6$. $\mathsf{IP}_{Cus_6} = \{\{Cus_7, Cus_8\}\}$, $\mathsf{IP}_{Mer_7} = \{\{Mer_8\}\}$, and the set of messages involved in $\{Mer_8\}$ is a subset of the messages involved in $\{Cus_7, Cus_8\}$ ($\{PayCredit\} \subseteq \{PayCredit, ProvideBirthDate\}$). Therefore, $\lfloor Cus_6 \rfloor$.* ▮

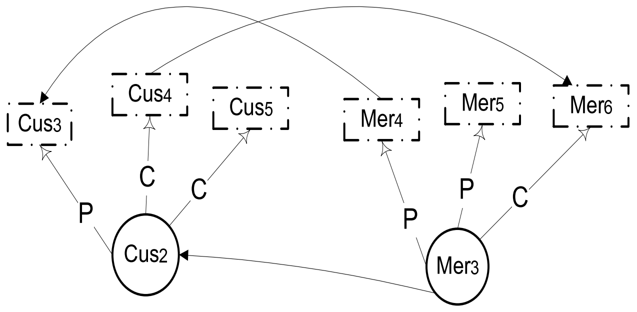

Figure 2: Program analysis graph for agents in Table 3

Table 4: Precondition coverage: merchant uses fewer messages

customer (c)
$Cus_6 = Offer$ means $\mathsf{C}(m, c, pay \wedge legalAge, goods)$
$Cus_7 = PayCredit$ means pay
$Cus_8 = ProvideBirthDate$ means $legalAge$
merchant (m)
$Mer_7 = Offer$ means $\mathsf{C}(m, c, pay \wedge legalAge, goods)$
$Mer_8 = PayCredit$ means $pay \wedge legalAge$

Figure 3 shows the program analysis graph for the agents in Table 4.

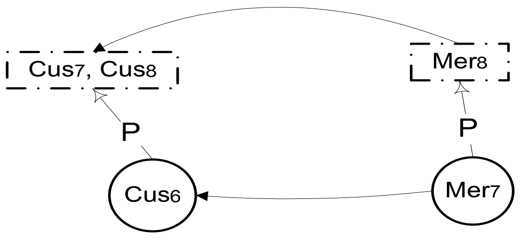

Figure 3: Program analysis graph for agents in Table 4

EXAMPLE 10. *Referring to Table 5, $\mathsf{IP}_{Cus_9} = \{\{Cus_{10}\}\}$ and $\mathsf{IP}_{Mer_9} = \{\{Mer_{10}, Mer_{11}\}\}$. The messages involved in $\{Mer_{10}, Mer_{11}\}$ are not a subset of $\{Cus_{10}\}$. Therefore, when customer does $PayCredit$ (after Offer), she assumes the commitment $\mathsf{C}(m, c, \top, goods)$ whereas the merchant does not because it does not see $PayCredit$ to mean $legalAge$: it also expects to observe $ProvideBirthDate$. Therefore, precondition coverage for $\{Cus_{10}\}$ does not hold. Therefore, $\lfloor Cus_9 \rfloor$ does not hold.* ▮

Figure 4 shows the program analysis graph for the agents in Table 5.

EXAMPLE 11. *Referring to Table 6, $\lfloor Cus_{11} \rfloor$ holds in spite of the fact that between the two agents the meanings of $PayCredit$ and $ProvideBirthDate$ are interchanged.* ▮

EXAMPLE 12. *A merchant may unconditionally commit to sending the goods. Referring to Table 7, $Mer_{15} \vdash Cus_{14}$, $\mathsf{IP}_{Cus_{14}} = \{\{Cus_{15}\}\}$, $\mathsf{IP}_{Mer_{15}} = \{\epsilon\}$ and since causing ϵ requires no messages, we obtain $\lfloor Cus_{14} \rfloor$.* ▮

EXAMPLE 13. *For the sake of completeness, let's consider an example that bring condition coverage into focus. Referring to*

Table 5: No precondition coverage: customer uses fewer messages

customer (c)
$Cus_9 = \mathit{Offer}$ means $\mathsf{C}(m, c, pay \wedge legalAge, goods)$
$Cus_{10} = \mathit{PayCredit}$ means $pay \wedge legalAge$
merchant (m)
$Mer_9 = \mathit{Offer}$ means $\mathsf{C}(m, c, pay \wedge legalAge, goods)$
$Mer_{10} = \mathit{PayCredit}$ means pay
$Mer_{11} = \mathit{ProvideBirthDate}$ means $legalAge$

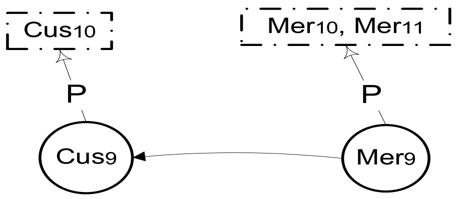

Figure 4: Program analysis graph for agents in Table 5

Table 8, $Mer_{16} \vdash Cus_{16}$, and $\mathbb{C}_{Cus_{16}} = \{Cus_{17}, Cus_{18}\}$ and $\mathbb{C}_{Mer_{16}} = \{Mer_{17}\}$. When the merchant sends GoodsShip and discharges her commitment, the customer also understands its receipt to mean discharge of the merchant's commitment. Thus, $\lfloor Cus_{16} \rfloor$ holds. ▮

Figure 5 shows the program analysis graph for the agents in Table 8.

EXAMPLE 14. *Also, it is worth considering the agents in Table 9. Even though GoodsShip means different things to the customer and merchant, when they observe the message, it discharges the merchant's commitment in both the customer and merchant's model. Here too, we have $\lfloor Cus_{19} \rfloor$.* ▮

Figure 6 shows the program analysis graph for the agents in Table 9.

THEOREM 1. *$⟅\{x, y\}⟆$ if and only if $[[\{x, y\}]]$.*

PROOF. (*Sketch*) The proof is by induction on the length of quiescent vectors. For quiescent observation vectors of length 1, any commitment that exists must be caused by rules (in the creditor's theory) pertaining to a single message. As a result, our decision procedure would find the rules (in the debtor's theory) that cover such creditor rules. Conversely, for any pair of covering rules the message mentioned in their bodies could be observed to be sent and received. Thus if the decision procedure finds a cover, there would be an observation vector where that is realized.

Now assume that the theorem holds for quiescent observation vectors of length up to k. Consider any quiescent observation vector of length $k + 1$. Any commitment that holds after a sequence of length $k + 1$ either held at the end of the first k observations in that sequence, or is caused by the $(k + 1)^{st}$ observation. In the former case, the inductive hypothesis applies. In the latter, the rules for the last message apply. The precondition and condition supports for these rules must have already have been accounted for in the first k observations. Hence, by induction, the result holds. □

Table 6: Offer with jumbled but adequate meanings

customer (c)
$Cus_{11} = \mathit{Offer}$ means $\mathsf{C}(m, c, pay \wedge legalAge, goods)$
$Cus_{12} = \mathit{PayCredit}$ means $legalAge$
$Cus_{13} = \mathit{ProvideBirthDate}$ means pay
merchant (m)
$Mer_{12} = \mathit{Offer}$ means $\mathsf{C}(m, c, pay \wedge legalAge, goods)$
$Mer_{13} = \mathit{PayCredit}$ means pay
$Mer_{14} = \mathit{ProvideBirthDate}$ means $legalAge$

Table 7: Making an unconditional commitment

customer (c)
$Cus_{14} = \mathit{Offer}$ means $\mathsf{C}(m, c, pay, goods)$
$Cus_{15} = \mathit{PayCash}$ means pay
merchant (m)
$Mer_{15} = \mathit{Offer}$ means $\mathsf{C}(m, c, \top, goods)$

Table 8: Condition coverage

customer (c)
$Cus_{16} = \mathit{Offer}$ means $\mathsf{C}(m, c, pay, goods \vee refund)$
$Cus_{17} = \mathit{GoodsShip}$ means $goods$
$Cus_{18} = \mathit{RefundMoney}$ means $refund$
merchant (m)
$Mer_{16} = \mathit{Offer}$ means $\mathsf{C}(m, c, pay, goods)$
$Mer_{17} = \mathit{GoodsShip}$ means $goods$

4. DISCUSSION

Researchers in software components have long addressed the problem of component interoperability. They have approached this problem from the point of view of coordination: the definitions of interoperability are couched in terms of process-algebraic notions of liveness, fairness, choice, and deadlock-freedom of the components [3, 9, 18, 8]. Such formalizations are no doubt relevant and essential; however, they do not capture the business meaning of business processes. Our commitment-based approach addresses this shortcoming. It abstracts away from the process-algebraic notions of interoperability, and makes commitment alignment the sole criterion. Our vision is that designers first specify agents in terms of commitments, check for commitment alignment, and then successively refine the specifications in a model-driven manner so as to obtain implementations that also meet the process-algebraic notions of interoperability.

Approaches based on verifying compliance at runtime [2, 15] are important in the context of open systems since agents may behave in unpredictable ways; also it is necessary to have independent arbiters in cases of dispute. Alberti *et al.* [1] present SCIFF, an abductive reasoning framework for reasoning about the policies of services with the purpose of verifying if a goal might be reached. In that sense, SCIFF is similar to specifying agents in $C+$ [10], and running queries in $CCalc$, which is the reasoning tool that implements $C+$.

Our work falls within the broader context of normative multiagent systems (for example, [4]). Our use of constitutive rules to means *counts as* is decidedly narrow in that a message only counts

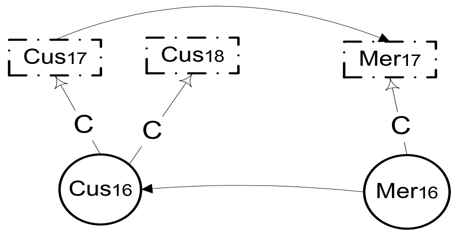

Figure 5: Program analysis graph for agents in Table 8

Table 9: Condition coverage: case of adequate meaning

customer (c)
$Cus_{19} = \mathit{Offer}$ means $\mathsf{C}(m, c, pay, goods \vee refund)$
$Cus_{20} = \mathit{GoodsShip}$ means $refund$
merchant (m)
$Mer_{18} = \mathit{Offer}$ means $\mathsf{C}(m, c, pay, goods)$
$Mer_{19} = \mathit{GoodsShip}$ means $goods$

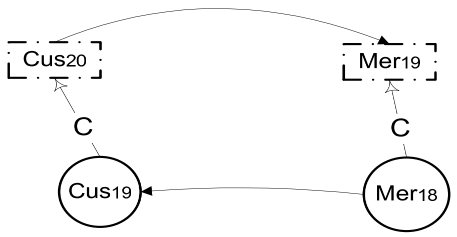

Figure 6: Program analysis graph for agents in Table 9

as meaning something for a particular agent, and not in the context of the institution the agent acts in. In this work, we assume that the agents act in an institution, but may have differing views on the creation of institutional facts. Such differences in views are the basis of failure of interoperability. In future work, we will broaden the formalization to include institutions.

Winikoff [16] studies the distributed enactment of a commitment protocol amongst agents. Commitments are mapped to BDI plans, and all possible plans are checked to see if they allow making progress towards desirable goal states. This enables designers to specify commitment protocols and not have to worry about low-level messaging details, which is highly desirable. Since commitments are already aligned in a commitment protocol, there is no need to check commitment-level interoperability between the distributed commitment machines, which is the question we address in this work.

We will extend our algorithm to handle additional commitment operations such as delegate, cancel, assign, and release. Doing so will enable modeling general multiparty interactions. Addressing regulative interoperability in more depth is also an important direction.

5. REFERENCES

[1] M. Alberti, F. Chesani, M. Gavanelli, E. Lamma, P. Mello, M. Montali, and P. Torroni. Web service contracting: Specification and reasoning with SCIFF. In *Proceedings of the 4th European Semantic Web Conference*, pages 68–83, 2007.

[2] M. Alberti, D. Daolio, P. Torroni, M. Gavanelli, E. Lamma, and P. Mello. Specification and verification of agent interaction protocols in a logic-based system. In *Proceedings of the 19th ACM Symposium on Applied Computing*, pages 72–78, 2004.

[3] M. Baldoni, C. Baroglio, A. Martelli, and V. Patti. Verification of protocol conformance and agent interoperability. In *6th International Workshop on Computational Logic in Multi-Agent Systems (CLIMA 2005)*, volume 3900 of *LNCS*, pages 265–283. Springer, 2006.

[4] G. Boella and L. W. N. van der Torre. Regulative and constitutive norms in normative multiagent systems. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Ninth International Conference (KR)*, pages 255–266. AAAI Press, 2004.

[5] A. K. Chopra and M. P. Singh. Contextualizing commitment protocols. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1345–1352, 2006.

[6] N. Desai, A. K. Chopra, M. Arrott, B. Specht, and M. P. Singh. Engineering foreign exchange processes via commitment protocols. In *Proceedings of the 4th IEEE International Conference on Services Computing (SCC)*, pages 514–521, Los Alamitos, 2007. IEEE Computer Society Press.

[7] N. Desai, A. K. Chopra, and M. P. Singh. Representing and reasoning about commitments in business processes. In *Proceedings of the 22nd Conference on Artificial Intelligence (AAAI)*, pages 1328–1333, Menlo Park, July 2007. AAAI Press.

[8] U. Endriss, N. Maudet, F. Sadri, and F. Toni. Protocol conformance for logic-based agents. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 679–684, 2003.

[9] C. Fournet, C. A. R. Hoare, S. K. Rajamani, and J. Rehof. Stuck-free conformance. In *Proceedings of the 16th International Conference on Computer Aided Verification (CAV)*, volume 3114 of *LNCS*, pages 242–254. Springer, 2004.

[10] E. Giunchiglia, J. Lee, V. Lifschitz, N. McCain, and H. Turner. Nonmonotonic causal theories. *Artificial Intelligence*, 153(1-2):49–104, 2004.

[11] G. Hohpe and B. Woolf. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. Signature Series. Addison-Wesley, Boston, 2004.

[12] D. L. Parnas. Information distribution aspects of design methodology. In *Proceedings of the International Federation for Information Processing Congress*, volume TA-3, pages 26–30, Amsterdam, 1971. North Holland.

[13] J. R. Searle. *The Construction of Social Reality*. Free Press, New York, 1995.

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

[15] 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.

[16] M. Winikoff. Implementing commitment-based interaction. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 868–875, Columbia, SC, May 2007. International Foundation for Autonomous Agents and MultiAgent Systems.

[17] WS-CDL. Web services choreography description language version 1.0, Nov. 2005. www.w3.org/TR/ws-cdl-10/.

[18] D. M. Yellin and R. E. Strom. Protocol specifications and component adaptors. *ACM Transactions on Programming Languages and Systems*, 19(2):292–333, 1997.