Whatever You Say

Luke Hunsberger

Vassar College
Poughkeepsie, NY 12604-0444, USA
hunsberg@cs.vassar.edu

Abstract. This paper addresses an important problem in multi-agent coordination: the formal representation of parameters in the content of agent intentions that are only partially specified (e.g., when the intended action has not yet been executed and values for the parameters have not yet been chosen or the authority for choosing such values has been delegated to others). For example, Abe might intend to rent "whatever car Zoe tells him to", in which case the problem is how to formally represent the quoted clause (i.e., the "whatever" content). The paper presents a two-pronged approach. First, it uses the event calculus to model declarative speech-acts which agents use to establish facts about parameters in a social context. Second, it partitions the content of agent intentions into (1) a condition that the agent should refrain from determining and (2) a goal that the agent should strive to achieve. The satisfaction conditions of such intentions treat these types of content differently; however they can share variables and, thus, are linked in a restricted sense.

1 Introduction

Since people have limited computational resources, they cannot, at each moment, instantaneously compute their optimal action for that moment; instead, they must plan ahead [3]. Thus, they adopt plans and intentions concerning their future activity which are only partially specified and which they subsequently elaborate over time [4, 10]. One common way for plans to be only partially specified is that their parameters may not be fully determined. For example, while having no particular car in mind, Abe might intend to rent *a car*. Later on, Abe might select a car—say, `Car39`—to rent. However, before he makes such a selection, there is no car about which we can say Abe intends to rent *that car.*

In addition to frequently being only partially specified, the plans and activities of different people are frequently interdependent, thus motivating people to coordinate their future-directed planning activity [9]. As a result, they must frequently negotiate about objects, such as the car mentioned above, that may be only partially specified. For example, suppose that Abe decides to let the rental agent Zoe select the car that he is going to rent. Abe must update his intention to reflect this delegation of parameter-binding authority; he now intends to rent *whatever* car Zoe selects for him. In this paper, intentions concerning this kind of partially specified content are called intentions with "whatever" content. For

M. Fisher et al. (Eds.): JELIA 2006, LNAI 4160, pp. 229–241, 2006.

computer agents to participate effectively in these kinds of commonplace, multi-agent planning and coordination scenarios, they must be able to represent and reason about intentions with "whatever" content.

An intention is *satisfied* [17] if it successfully motivates the intending agent to eventually do the action or achieve the state of affairs stipulated in its content. Whereas Abe's original intention "to rent a car" might be satisfied by his renting any of perhaps a hundred different cars, his updated intention "to rent whatever car Zoe selects" can only be satisfied by his renting whatever car Zoe happens to eventually select for him. Thus, the satisfaction of Abe's updated intention depends on how its "whatever" content is eventually determined—by Zoe.

This paper presents an approach to modeling intentions with "whatever" content that is based on public names (or identifiers) that agents mutually agree to use. For example, Abe and Zoe might agree to use the name `C1` to refer to the *context* of their rental-car interaction.[1] Similarly, they might agree to use the name `Car` to represent whatever car Zoe eventually selects for him. In this paper, `C1` is called a *social context* and `CAR` is called a parameter within a social context (or, a *social parameter*). Linking parameters to social contexts helps to disambiguate scenarios in which different contexts might have identically named parameters. For example, Zoe might be selecting rental cars for several people in different contexts.

The first part of this paper formally models the processes whereby agents assign names to social contexts and parameters associated with those contexts. The assignment of names is established by group declarations—that is, declarations attributable to groups. Agents also use group declarations to bind parameters to values and to delegate to other agents the authority for binding parameters. In this paper, a group declaration is modeled as an abstract event that "happens" when a group of agents make a decision, as a group, to make such a declaration. The formal definitions are given in terms of the existing GDMM framework for formally specifying group decision-making mechanisms [11, 12] that is recast, in this paper, in terms of the event calculus [14]. The event calculus facilitates the expression of axioms governing propositions (e.g., those established by declarations) that hold only over certain temporal intervals.

The second part of the paper addresses the syntax and satisfaction conditions for intentions with "whatever" content. It is formulated in terms of the *stit theory* of Belnap et al. [1]. The content of intentions is augmented to include not just a goal that the intending agent is committed to achieving, but also a proposition that the agent refrains from determining (e.g., Abe might intend to rent whatever car Zoe selects, while refraining from determining which car she selects).

The rest of this paper is organized as follows. Section 2 recasts the pre-existing GDMM framework in terms of the event calculus. Section 3 models group declarations that agents can use to manage social contexts and social parameters. Section 4 presents the syntax and satisfaction conditions for intentions with "whatever" content. Section 5 discusses related work and presents conclusions.

[1] Grosz [8] highlights the importance of context in collaborative, multi-agent scenarios.

2 The GDMM Framework in the Event Calculus

The GDMM framework [11, 12] is a framework for formally specifying group decision-making mechanisms (GDMMs). In that framework, a GDMM (or interaction protocol) is defined in terms of declarative speech-acts and the *incremental accumulation of authority.* In a typical GDMM, each agent might be authorized (by the group) to initiate a GDMM instance (i.e., a run of a protocol) by making an appropriate declaration. Such a declaration authorizes other agents to make further declarations, thereby establishing facts that in certain combinations authorize other agents to make still more declarations, and so on, until, in successful instances, some agent is eventually authorized to declare, on behalf of the group, that they have made a decision. That final, authorized declaration establishes the group decision as a mutually believed fact among the group members—an example of what Searle calls *institutional facts* [18].

The GDMM framework was originally presented using a dynamic, deontic, temporal logic that enabled various properties to be formally proven. However, that logic can be somewhat cumbersome when dealing with propositions—like those established or terminated by declarative speech-acts—that hold only over certain temporal intervals. Thus, this section recasts the original GDMM framework in terms of the event calculus [14]. In the recast framework, declarative speech-acts are represented by events; authorization conditions by fluents (i.e., reified propositions); and an authorized speech-act establishing the truth of its propositional content is represented by a speech-act event initiating an appropriate fluent. In addition, protocol-specific axioms specify a protocol's method of incrementally accumulating authority. For example, an axiom might stipulate that certain combinations of fluents authorize certain speech-acts; or that certain speech-acts initiate or terminate certain fluents. The recast GDMM framework is demonstrated on a sample *propose-accept-reject* (PAR) protocol [11, 12]; the formal definitions were tested using Shanahan's abductive event-calculus planner [19].

2.1 A Quick Summary of the Event Calculus

The event calculus [14] is based on *events, fluents* and *time-points.* Events include actions such as buying a book or making a declarative speech-act. Fluents are reified propositional terms that are *initiated* or *terminated* by events. Time-points mark the occurrence of events and the initiation or termination of fluents. The most important event-calculus predicates are listed in Fig. 1. In the figure, $\mathtt{E}$ represents an arbitrary event, $\mathtt{F}$ an arbitrary fluent, and $\mathtt{T}$, $\mathtt{T}_1$ and $\mathtt{T}_2$ arbitrary time-points. The predicates are governed by axioms such as SC1 and SC2 listed below. SC1 states that if $\mathtt{F}$ holds at time $\mathtt{0}$ and is not subsequently clipped, then it continues to hold. SC2 states that if $\mathtt{F}$ is initiated by $\mathtt{E}$ at time $\mathtt{T}_1$, and is not clipped between $\mathtt{T}_1$ and some later time $\mathtt{T}_2$, then $\mathtt{F}$ continues to hold at $\mathtt{T}_2$.

(SC1) $\mathtt{[initially(F)} \wedge \neg\mathtt{clip(0,F,T)]} \;\Rightarrow\; \mathtt{holds(F,T)}$

(SC2) $\mathtt{[happens(E,T_1)} \wedge \mathtt{inits(E,F,T_1)} \wedge \mathtt{(T_1{<}T_2)} \wedge \neg\mathtt{clip(T_1,F,T_2)]}$
$\quad\Rightarrow\; \mathtt{holds(F,T_2)}$

`happens(E,T)`	–	event `E` happens at time `T`.
`holds(F,T)`	–	fluent `F` holds at time `T`.
`initially(F)`	–	fluent `F` holds from time `0` onward (until clipped).
`inits(E,F,T)`	–	if `E` happens at time `T`, then `F` is initiated at time `T`.
`terms(E,F,T)`	–	if `E` happens at time `T`, then `F` is terminated at time `T`.
`clip(`T_1`,F,`T_2`)`	–	`F` is terminated sometime between times T_1 and T_2.
`declip(`T_1`,F,`T_2`)`	–	`F` is initiated sometime between times T_1 and T_2.

Fig. 1. Predicates used in the event calculus

2.2 Declarative Speech-Acts and Authorization Conditions

A declarative speech-act is represented by an event term of the form

$$\texttt{decl(G,Hs,Content)},\ \text{abbreviated as}\ \ \delta$$

where `G` is an agent (the speaker), `Hs` is a group of agents (the hearers), and `Content` is a fluent representing the propositional content of the declaration. Authorization for such a speech-act is represented by a fluent of the form

$$\texttt{auth(Gr, decl(G,Hs,Content))},\ \text{abbreviated as}\ \ \texttt{auth(Gr,}\delta\texttt{)}$$

That is, agent `G` is authorized by the group `Gr` to make a declaration with content `Content` to a set of hearers `Hs` $\subseteq$ `Gr`. Axiom A1, below, is the main axiom governing declarative speech-acts. It stipulates that any suitably authorized declaration establishes the truth of its propositional content.[2]

$$\texttt{holds(auth(Gr,}\delta\texttt{),T)} \Rightarrow \texttt{inits(}\delta\texttt{,Content,T)} \qquad \text{(A1)}$$

2.3 The PAR Protocol in the Recast Framework

In the sample PAR protocol [11, 12], agents use declarative speech-acts to make proposals, vote on proposals, and announce group decisions. Such speech-acts are represented by the event terms listed in Fig. 2. Axioms governing the incremental accumulation of authority in the PAR protocol are listed in Fig. 3.

Axiom E1 stipulates that each agent `G` in a group `Gr` is initially authorized to make proposals to `Gr`. In this axiom, the authorizing group and the set of hearers are the same (`Gr`); the content of the declarative speech-act, $\delta_{\texttt{MP}}$, is the fluent, `madeProp(G,Gr,Prop)`; and the predicate `proposable(Prop)` is used to restrict the range of allowable content for proposals. Axioms SC1, A1 and E1 together entail that any agent `G` is authorized to make a PAR proposal to any group `Gr` as long as: (1) `G` is a member of `Gr`; (2) the content of the proposal is "proposable"; and (3) the agent's authorization to make such proposals has not been "clipped" by an intervening event (e.g., a group decision to revoke it). To make a proposal, `G` simply declares that it has done so, whereupon (by Axiom A1) a fluent of the form, `madeProp(G,Gr,Prop)`, is initiated. Axiom E2

[2] In all axioms in this paper, all free variables are implicitly universally quantified.

- $\texttt{decl(G,Gr,made(G,Gr,Prop))}$, abbreviated $\delta_{\texttt{MP}}$:

  "G declares to the group Gr that it has made a proposal Prop."
- $\texttt{decl(G2,\{G,G2\},voted(G2,G,Gr,Prop,Vote))}$, abbreviated $\delta_{\texttt{V}}$:

  "G2 declares to G that it has made a vote concerning the proposal Prop, where Vote $\in$ {accept,reject}."
- $\texttt{decl(G2,\{G,G2\},voted(G2,G,Gr,Prop,accept))}$, abbreviated $\delta_{\texttt{VA}}$:

  "G2 declares to G that it has voted to accept the proposal Prop."
- $\texttt{decl(G,Gr,grAcc(Gr,Prop))}$, abbreviated $\delta_{\texttt{GA}}$:

  "G declares to the group Gr that they have accepted Prop."

Fig. 2. Event terms representing declarations in the PAR protocol

$$[(\texttt{G} \in \texttt{Gr}) \wedge \texttt{proposable(Prop)}] \Rightarrow \texttt{initially(auth(Gr,}\delta_{\texttt{MP}}\texttt{))} \tag{E1}$$

$$[(\texttt{G2} \in \texttt{Gr}) \wedge (\texttt{G2} \neq \texttt{G}) \wedge \texttt{holds(auth(Gr,}\delta_{\texttt{MP}}\texttt{),T)}] \Rightarrow \texttt{inits(}\delta_{\texttt{MP}}\texttt{,auth(Gr,}\delta_{\texttt{V}}\texttt{),T)} \tag{E2}$$

$$\texttt{holds(auth(Gr,}\delta_{\texttt{MP}}\texttt{),T)} \Rightarrow \texttt{inits(}\delta_{\texttt{MP}}\texttt{,accepters(G,Gr,Prop,}\emptyset\texttt{),T)} \tag{E3}$$

$$[\texttt{holds(accepters(G,Gr,Prop,Others),T)} \wedge \texttt{holds(auth(Gr,}\delta_{\texttt{VA}}\texttt{),T)}] \Rightarrow \texttt{inits(}\delta_{\texttt{VA}}\texttt{,accepters(G,Gr,Prop,\{G2\}} \cup \texttt{Others),T)} \tag{E4}$$

$$[(\texttt{Gr = \{G\}} \cup \texttt{Others}) \wedge \texttt{holds(accepters(G,Gr,Prop,Others),T)}] \Rightarrow \texttt{inits(}\delta_{\texttt{GA}}\texttt{,grAcc(Gr,Prop),T)} \tag{E5}$$

Fig. 3. Axioms governing incremental accumulation of authority in the PAR protocol

stipulates that the making of a proposal authorizes each of the other agents in the group to vote on it—either to accept or reject it.

In the PAR protocol, if every agent votes to accept a proposal, then the originator of that proposal—here, G—becomes authorized to declare, on behalf of the group, that they have made a decision—namely, to accept the proposal. G's authorization to make such a declaration is accumulated incrementally, over time, as each agent declares its own acceptance of the proposal, as governed by axioms E3, E4 and E5. Axiom E3 stipulates that the making of a proposal initiates a fluent of the form, $\texttt{accepters(G,Gr,Prop,}\emptyset\texttt{)}$, representing that no one in the group has (yet) voted to accept G's proposal. Axiom E4 stipulates that an agent G2's authorized vote to accept a proposal incrementally updates the list of "accepters" (by adding G2). Axiom E5 stipulates that if all of the other agents (Others) have voted to accept G's proposal, then G becomes authorized to declare on behalf of the group that they have accepted the proposal. If so authorized, then, by Axiom A1, `G`'s declaration establishes the fluent, `grAcc(Gr,Prop)`.

These axioms were encoded in Prolog and fed as input to Shanahan's abductive event-calculus planner [19] which was able to come up with valid sequences of speech-acts to yield various group decisions under the PAR protocol. For example, the following sequence was generated in response to a query about how a group of agents {`g,h,i`} might decide to accept a proposal `prop`:

```
happens(decl(g,[g,h,i],madeProp(g,[g,h,i],prop)),t51)
happens(decl(i,[g,i],voted(i,g,[g,h,i],prop,accept)),t52)
happens(decl(h,[g,h],voted(h,g,[g,h,i],prop,accept)),t50)
happens(decl(g,[g,h,i],grAcc([g,h,i],prop)),t48)
```

where the time-points were subject to the constraints: `t51 < t52 < t50 < t48`. Although the PAR protocol is quite simple, the same approach can be used to specify arbitrarily complex protocols based on declarative speech-acts and the incremental accumulation of authority in the GDMM framework.

3 Group Declarations for Contexts and Parameters

For this paper, the most important types of group decisions are those that establish names for social contexts or parameters within those contexts, and those that bind parameters to values or delegate authority for binding parameters. Such decisions can be made using any GDMM; thus, it is convenient to abstract away the GDMM used to generate the group decision and focus instead on the proposition established by that decision. Toward that end, this section uses the GDMM framework to model *group declarations*—that is, declarations attributable to groups of agents. It then addresses the use of group declarations to manage social contexts and parameters within those contexts.

In the single-agent case, an agent might establish the binding of a parameter thusly: "I hereby declare that the parameter `P` in the context `C` shall be bound to the value `67`." By analogy, a group can establish such facts by making *group declarations.* In particular, a group declaration, if suitably authorized, has the power to establish the truth of its propositional content. However, a group declaration is not uttered; instead it "happens", by convention, when, at the successful culmination of a GDMM instance, one of the agents announces, on behalf of the group, that they have decided to make a declaration. For example, at the end of a complex group decision-making procedure, a member of Congress might announce that the Congress has decided, as a group, to declare war against some other country. In such a case, the declaration of war is attributed to the Congress as a whole, not to the individual making the announcement.

It is important to distinguish two kinds of authorization associated with group declarations: internal and external. Internal authorization is that which is incrementally accumulated during a run of whatever GDMM is being used to generate the group declaration. For example, the member of Congress announcing their decision to declare war must be suitably authorized by the Congress; otherwise, no group declaration takes place. In contrast, the external authorization for a group declaration is independent of the GDMM used to generate it. Instead, external authorization, which frequently comes from outside the group making the declaration, is that which gives the group's declaration the power to establish the truth of its propositional content. In other words, the external authorization for group declarations is analogous to the authorization conditions for single-agent declarations. For example, a declaration of war by the Congress has the power to establish a state of war only because the people, via the Constitution, have authorized Congress to make such declarations.

A group declaration is represented by an event term of the form

$$\texttt{grDecl(Gr,Hs,Content)},\ \text{abbreviated as}\ \varDelta$$

where `Gr` represents the group making the declaration, `Hs` represents the set of hearers, and `Content` is a fluent representing the content of the declaration.[3] A group declaration is not an action that is directly "executable" by the group. Instead, a group declaration "happens", by convention, when a group makes a group decision whose content has the form

$$\texttt{done(grDecl(Gr,Hs,Content))},\ \text{abbreviated as}\ \texttt{done(}\varDelta\texttt{)}.$$

In such a case, the group decision initiates (e.g., by Axiom E5) a fluent

$$\texttt{grAcc(Gr,done(}\varDelta\texttt{))}$$

which can be glossed as "`Gr` has decided to make a group declaration to `Hs` that `Content` holds." Axiom A2, below, stipulates that such a fluent "counts as" [18] a group having made the indicated group declaration.

$$\texttt{holds(grAcc(Gr,done(}\varDelta\texttt{)),T)} \Rightarrow \texttt{happens(}\varDelta\texttt{,T)} \qquad \text{(A2)}$$

Then, in direct analogy with Axiom A1, Axiom A3 below stipulates that an authorized group declaration establishes the truth of its propositional content.

$$\texttt{holds(auth(AuthGr,}\varDelta\texttt{),T)} \Rightarrow \texttt{inits(}\varDelta\texttt{,Content,T)} \qquad \text{(A3)}$$

In this axiom, `AuthGr` represents the (external) authorizing group.

The rest of this section focuses on how agents can use group declarations to establish names for social contexts and social parameters, and to bind such parameters or delegate the authority for binding them. In what follows, all contexts and parameters are presumed to be under the sole control of the group `Gr`—that is, `Gr` is its own "external" authorizing group. In addition, the set of hearers is presumed to be the entire group. Thus, `Gr` = `AuthGr` = `Hs`. In addition, for convenience, repeated arguments are omitted. Thus, a group declaration is represented by a term of the form, `grDecl(Gr,Content)`—abbreviated as $\varDelta$—and the corresponding authorization condition is represented by a fluent of the form,

[3] $\varDelta$ denotes a group declaration; δ denotes a single-agent declaration.

- grDecl(Gr,sContext(Gr,C)), abbreviated $\Delta_{\mathtt{C}}$:
  "Group Gr declares a new social context named C."
- grDecl(Gr,sParam(Gr,C,P)), abbreviated $\Delta_{\mathtt{P}}$:
  "Group Gr declares a new parameter named P associated with context C."
- grDecl(Gr,sBindParam(Gr,C,P,V)), abbreviated $\Delta_{\mathtt{BP}}$:
  "Group Gr declares that parameter P in context C is bound to value V."
- grDecl(Gr,sDelegParam(G,Gr,C,P)), abbreviated $\Delta_{\mathtt{DP}}$:
  "Group Gr declares that the authority for binding the parameter P in the context C is delegated to the agent G."
- decl(G,Gr,sBindParam(Gr,C,P,V)), abbreviated $\delta_{\mathtt{BP}}$:
  "Agent G declares that parameter P in context C is bound to value V."

Fig. 4. Event terms representing declarations for social contexts and parameters

initially(auth($\Delta_{\mathtt{C}}$))	(E6)
holds(auth($\Delta_{\mathtt{C}}$),T) $\Rightarrow$ inits($\Delta_{\mathtt{C}}$,auth($\Delta_{\mathtt{P}}$),T)	(E7)
holds(auth($\Delta_{\mathtt{P}}$),T) $\Rightarrow$ inits($\Delta_{\mathtt{P}}$,auth($\Delta_{\mathtt{BP}}$),T)	(E8)
holds(auth($\Delta_{\mathtt{P}}$),T) $\Rightarrow$ inits($\Delta_{\mathtt{P}}$,auth($\Delta_{\mathtt{DP}}$),T)	(E9)
holds(auth($\Delta_{\mathtt{DP}}$),T) $\Rightarrow$ inits($\Delta_{\mathtt{DP}}$,auth(Gr,$\delta_{\mathtt{BP}}$),T)	(E10)

Fig. 5. Axioms pertaining to the declarations in Fig. 4

auth(grDecl(Gr,Content))—abbreviated as auth(Δ). Fig. 4 lists the types of group declarations (and one single-agent declaration) to be discussed. Fig. 5 lists the axioms pertaining to the declarations in Fig. 4.

A group Gr creates a social context named C by making a declaration of the form $\Delta_{\mathtt{C}}$ in Fig. 4. By Axiom E6 in Fig. 5, any group is initially authorized to create arbitrary social contexts for itself. Thus, by Axiom A3, a group declaration, $\Delta_{\mathtt{C}}$, establishes a fluent of the form, sContext(Gr,C).

A group Gr creates a social parameter named P, linked to a social context C, by making a declaration of the form $\Delta_{\mathtt{P}}$ in Fig. 4. By Axiom E7, a group's creation of a social context (Gr,C) automatically authorizes that group to create social parameters within that context. Similarly, a group's creation of a social parameter (Gr,C,P) automatically authorizes that group to bind that parameter to some value (Axiom E8) or *delegate* the authority for binding that parameter to some other agent (Axiom E9).

A group Gr binds a parameter P in the context C to the value V by making a declaration of the form $\Delta_{\mathtt{BP}}$ in Fig. 4. If suitably authorized, then, by Axiom A3, such a declaration would initiate a fluent of the form `sBindParam(Gr,C,P,V)`.[4] Alternatively, a group might decide to delegate the authority for binding that parameter to some agent, say `G`, by making a declaration of the form $\Delta_{\mathtt{DP}}$ in Fig. 4. By Axiom E10, such a declaration authorizes `G` to bind `P` to *any* value `V` by making a declaration of the form $\delta_{\mathtt{BP}}$ in Fig. 4.[5] Should `G` make such a declaration, it would, by Axiom A1, initiate the fluent, `sBindParam(Gr,C,P,V)`. Thus, whether the group `Gr` binds `P` directly using a group decision or indirectly via the delegate `G`, the end result is the initiation of the same fluent: `sBindParam(Gr,C,P,V)`.

Example. Abe (`a`) intends to rent whatever car Zoe (`z`) selects for him. In this case, they make group declarations that initiate the following fluents:

`sContext({a,z},c)`	– `c` is a social context for them.
`sParam({a,z},c,p)`	– `p` is a social parameter for them in that context.
`sDelegParam(z,{a,z},c,p)`	– they have delegated the binding of `p` to Zoe.

By Axiom E10, the last fluent in the above list initiates the following fluent, which represents that Zoe is authorized to bind `p` to *any* value `V`.

```
auth({a,z},decl(z,{a,z},sBindParam({a,z},c,p,V)))
```

4 Intentions with "Whatever" Content

This section presents a novel representation for intentions with "whatever" content. The satisfaction conditions for such intentions clearly distinguish conditions that the intending agent seeks to achieve and those that it actively refrains from determining. The representation is expressed in terms of the "sees to it that" (*stit*) operator defined by Belnap et al. [1], which is briefly summarized below. Afterward, intentions with "whatever" content and their satisfaction conditions are defined and the definitions are illustrated with examples.

4.1 Seeing to It That

Belnap et al. [1] present a theory of "agents and choices in branching time" within which they formally define a modal "sees to it that" (*stit*) operator, which they use to represent *agentive* expressions. They argue that "[a proposition] Q is agentive for [an agent] α just in case Q may be usefully paraphrased as [α *stit*: Q]." For example, the sentence, "Abe sees to it that a car is rented", is agentive for Abe since it has the form, [A *stit*: ϕ], where A denotes Abe and ϕ denotes the proposition, "a car is rented".

[4] The binding of a parameter should also terminate that group's authority to subsequently bind that same parameter or to delegate the binding of that parameter. For space reasons, providing such axioms is left to the reader.

[5] A decision to delegate parameter-binding authority to an agent `G` would also entail an obligation on `G` to eventually bind that parameter; however, this paper focuses on authorization conditions, not obligations. Grosz and Hunsberger [9] address some of the obligations entailed by various kinds of group decision.

The semantics of the *stit* operator stipulate that [α *stit*: Q] holds now if and only if: (1) Q holds now due to a prior choice (or sequence of choices) made by α; and (2) α's choice was real in the sense that some other choice open to α might have resulted in Q *not* holding. For example, I might see to it that my office gets cold by opening a window, where my alternative, leaving the window closed, might have resulted in my office staying warm.

In their "Restricted Complement Thesis", Belnap et al. argue that "a variety of constructions concerned with agents and agency—including deontic statements, imperatives, and statements of intention, among others—must take agentives as *their* complements." For example, the expression, *Int*: [α *stit*: Q], would represent that the agent α intends to see to it that the proposition Q holds.

Belnap et al. define *active refraining* in terms of the *stit* operator, as follows:

$$\mathit{refrain}(\alpha, \psi) \;\equiv\; [\alpha\ \mathit{stit}\colon \neg[\alpha\ \mathit{stit}\colon \psi]]$$

That is, an agent α actively refrains from bringing about ψ if α sees to it that α does *not* see to it that ψ holds. In other words, some choice made by α, perhaps even a choice to do nothing, must guarantee that $\neg\psi$ remains an option—at least insofar as α's choices are concerned. Of course, the choice(s) of some other agent(s) might nonetheless establish ψ, despite α's refraining.

The following abbreviation will be useful later in this section:

$$\mathit{refrain}(\alpha, \pm\psi) \;\equiv\; \mathit{refrain}(\alpha, \psi) \wedge \mathit{refrain}(\alpha, \neg\psi)$$

That is, α *both* refrains from ψ and refrains from $\neg\psi$. Such an expression holds if some prior choice(s) made by α guarantee that both ψ and $\neg\psi$ remain options.

4.2 Intentions with "Whatever" Content

Definition 1. *An* intention with "whatever" content *is an expression of the form:* $\mathit{Int}_w(G, x, \psi(x), \phi(x))$*, where* G *is a term,* x *is a variable, and* $\psi(x)$ *and* $\phi(x)$ *are arbitrary propositions that may contain free occurrences of* x*.*

The intended interpretation of such an expression is that the agent G intends to see to it that the proposition $\phi(x)$ holds for whatever (unique) value of x the proposition $\psi(x)$ holds, while refraining from determining the choice of x for which $\psi(x)$ holds. The formal interpretation is given in Definition 2.

Definition 2. $\mathit{Int}_w(G, x, \psi(x), \phi(x))$ *is* satisfied *if:*

(1) $(\forall x)\mathit{refrain}(G, \pm\psi(x))$ *holds; and*
(2) **if** *there is a unique object* d *in the semantic domain for which the expression* $\psi(c_d)$ *holds, where* c_d *is a constant term denoting* d *and* $\psi(c_d)$ *is obtained from* $\psi(x)$ *by substituting* c_d *for each occurrence of the free variable* x *in* $\psi(x)$*,* **then** *the expression,* $\mathit{stit}(G, \phi(c_d))$*, also holds.*

Condition 1 stipulates that G should refrain from determining $\psi(x)$ or $\neg\psi(x)$ for any x—that is, choices made by G should guarantee that both $\psi(x)$ and $\neg\psi(x)$ remain options for any x. Condition 2 stipulates that *if* there is a unique value of x for which the expression $\psi(x)$ holds, then the agent G must see to it that the expression $\phi(x)$ holds *for that same value of* x.

Example. Recall Abe's intention to rent whatever car Zoe selects for him. Suppose that Abe (A) and Zoe (Z) have already established a name C for a social context and a parameter P for the car. Suppose further that they have delegated the binding of P to Zoe. Abe's intention can be represented by an intention with "whatever" content where:

$\psi(x) \equiv sBindParam(\{A, Z\}, C, P, x)$; and
$\phi(x) \equiv Rents(A, x)$.

According to Definition 2, for Abe's intention to be satisfied, he must refrain from both $\psi(x)$ and $\neg\psi(x)$ for all x. In other words, *his* choices must not constrain the possible values for the parameter P. In addition, if the condition $\psi(x)$ holds for some *unique* value of x (e.g., should Zoe declare P to have the value `Car39`), then Abe must see to it that $\phi(x)$ holds for that value of x (e.g., that he rents `Car39`). In short, if Abe refrains from determining which car is selected, and Zoe selects a unique car, then Abe must see to it that he rents *that* car; however, if no such car is selected, or more than one is selected, then Abe's intention is trivially satisfied.

Intentions with "whatever" content can be defined with multiple partially-determined objects by substituting $(x_1, x_2, \ldots, x_n)$ for x, $(\forall x_1, x_2, \ldots, x_n)$ for $(\forall x)$, and $\psi_1(x_1) \wedge \ldots \wedge \psi_n(x_n)$ for $\psi(x)$ in Definition 1; and, in addition, substituting $(d_1, \ldots, d_n)$ for d and $(c_{d_1}, \ldots, c_{d_n})$ for c in Definition 2. For example, Abe's intention to hammer in a nail using whatever hammer Zoe specifies and whatever nail Yao (Y) specifies could be represented by

$$Int_w(A, (x_1, x_2), \psi_1(x_1) \wedge \psi_2(x_2), \phi(x_1, x_2))$$

where
$$\begin{aligned} \psi_1(x_1) &\equiv sBindParam(Z, C, \texttt{hamr}, x_1); \\ \psi_2(x_2) &\equiv sBindParam(Y, C, \texttt{nail}, x_2); \text{ and} \\ \phi(x_1, x_2) &\equiv Pounds(A, x_1, x_2). \end{aligned}$$

5 Related Work and Conclusions

Several researchers are actively investigating the use of the event calculus to model interaction protocols in normative settings. For example, Yolum and Singh [20] use it to model protocols as *commitment machines.* In that work, agents use various kinds of speech-acts to adopt or modify (one-on-one) social commitments. Pitt et al. [16] use the event calculus to formalize a complex voting protocol for general-purpose decision-making in virtual organizations. In their work, events such as proposing, voting, and so forth initiate or terminate various powers (authorizations), permissions and obligations. These approaches are complementary to the approach taken in this paper where the GDMM framework is based solely on declarative speech-acts and the incremental accumulation of authority, and is used to model declarations attributable to groups.

Other researchers are investigating contracts for multi-agent systems in terms of normative concepts. For example, Farrell et al. [6] define a contract language in terms of obligation, power and permission and present an algorithm for tracking the normative state of a contract over its entire life-cycle. And Boella and van der Torre [2] view contracts as legal institutions based on Searle's *construction of social reality* [18]. In their work, mental states such as beliefs, desires and intentions are attributed not only to agents, but also to normative systems. In prior work, Grosz and Hunsberger [9] specify the obligations entailed by certain common types of group decision (e.g., binding a parameter, selecting a recipe for a complex task, or delegating a task) in the context of multi-agent coordination scenarios. Ongoing research is aimed at augmenting that work to include authorizations and permissions, as well as intentions with "whatever" content.

The most related work on delegation is that of Norman and Reed [15]. In their work, agents use imperative speech-acts to delegate tasks to other agents and to command others to refrain from further delegating those same tasks. They use a propositional logic and thus do not address intentions with "whatever" content, but they employ two *stit* operators, one for propositions and one for actions.

In the field of linguistics, Dekker and van Rooy [5] formally analyze so-called *Hob-Nob sentences* in which "a number of people ... have attitudes with a common focus, whether or not there actually is something at that focus" which is a broad category that seems to include intentions with "whatever" content.[6] In addition, Kamp and Reyle [13] use Discourse Representation Theory (DRT) to analyze sentences (or sequences of sentences) that include partially specified content in the form of indefinite noun phrases and pronouns that subsequently refer to that content—as in: "Every farmer who owns a donkey beats it" or "John owns a Porsche. It fascinates him." An investigation into the potential application of these methods to intentions with "whatever" content (i.e., partially specified content to which agents need to refer as they coordinate with others), is the subject of ongoing research.

The research presented in this paper is part of a long-term project aimed at developing collaboration-capable computer agents [9]. Current work is focused on providing a comprehensive logical foundation for intentions with "whatever" content that can accommodate other types of partially specified content (e.g., Bea intends to drive whatever car Abe rents) as well as the obligations that are entailed by group decisions in multi-agent planning and coordination scenarios.

References

1. Nuel Belnap, Michael Perloff, and Ming Xu. *Facing the Future*. Oxford University Press, 2001.
2. Guido Boella and Leendert van der Torre. Contracts as legal institutions in organizations of autonomous agents. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*. ACM, 2004.
3. Michael E. Bratman. *Intentions, Plans, and Practical Reason*. Harvard University Press, Cambridge, MA, 1997.
4. Michael E. Bratman. *Faces of Intention: Selected Essays on Intention and Agency*. Cambridge University Press, 1999.

[6] The quotation is from Geach [7], cited in Dekker and Rooy [5].

5. Paul Dekker and Robert van Rooy. Intentional identity and information exchange. In R. Cooper and T. Gamkrelidze, editors, *Second Tbilisi Symposium on Language, Logic and Computation*, 1997.
6. Andrew D. H. Farrell, Marek Sergot, Mathias Salle, and Claudio Bartolini. Using the event calculus for tracking the normative state of contracts. *International Journal of Cooperative Information Systems*, 14(2–3), June–September 2005.
7. P. Geach. Intentional identity. *Journal of Philosophy*, 74:309–44, 1967.
8. Barbara J. Grosz. The contexts of collaboration. In E. Sosa K. Korta and X. Arrazola, editors, *Cognition, Agency and Rationality*, pages 175–188. Kluwer Press, Dordrecht, 1999.
9. Barbara J. Grosz and Luke Hunsberger. The dynamics of intention in collaborative activity. *Journal of Cognitive Systems Research*, 7:259–272, 2006.
10. Barbara J. Grosz and Sarit Kraus. Collaborative plans for complex group action. *Artificial Intelligence*, 86:269–357, 1996.
11. Luke Hunsberger. *Group Decision Making and Temporal Reasoning*. PhD thesis, Harvard University, 2002. Available as Harvard Technical Report TR-05-02.
12. Luke Hunsberger. A framework for specifying group decision-making mechanisms (poster). In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS-2005)*. ACM Press, 2005.
13. Hans Kamp and Uwe Reyle. *From Discourse to Logic*, volume 42 of *Studies in Linguistics and Philosophy*. Kluwer Academic Publishers, 1993.
14. R.A. Kowalski and M. Sergot. A logic-based calculus of events. *New Generation Computing*, 4:67–95, 1986.
15. Timothy J. Norman and Chris Reed. A model of delegation for multi-agent systems. In M. Fisher M. d'Inverno, M. M. Luck and C. Preist, editors, *Foundations and Applications of Multi-Agent Systems*, volume 2403 of *Lecture Notes in Artificial Intelligence*, pages 185–204. Springer-Verlag, 2002.
16. Jeremy Pitt, Lloyd Kamara, Marek Sergot, and Alexander Artikis. Formalization of a voting protocol for virtual organizations. In *Fourth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2005)*, pages 373–380. ACM, 2005.
17. J. R. Searle. *Speech acts: An essay in the philosophy of language*. Cambridge University Press, Cambridge, UK, 1969.
18. J.R. Searle. *The Construction of Social Reality*. Allen Lane, London, 1995.
19. Murray Shanahan. *The Event Calculus Explained*, volume 1600 of *Lecture Notes in Computer Science*. 1999.
20. Pinar Yolum and Munindar P. Singh. Reasoning about commitments in the event calculus: An approach for specifying and executing protocols. *Annals of Mathematics and Artificial Intelligence (AMAI)*, 2003.