

Dynamic intention structures I: a theory of intention representation

Luke Hunsberger · Charles L. Ortiz, Jr.

Springer Science+Business Media, LLC 2008

Abstract This article introduces a new theory of intention representation which is based on a structure called a Dynamic Intention Structure (DIS). The theory of DISs was motivated by the problem of how to properly represent incompletely specified intentions and their evolution. Since the plans and intentions of collaborating agents are most often elaborated incrementally and jointly, elaboration processes naturally involve agreements among agents on the identity of appropriate agents, objects and properties that figure into their joint plans. The paper builds on ideas from dynamic logic to present a solution to the representation and evolution of agent intentions involving reference to incompletely specified and, possibly, mutually dependent intentions, as well as the objects referenced within those intentions. It provides a first order semantics for the resulting logic. A companion paper extends further the logical form of DISs and explores the problem of logical consequence and intention revision.

Keywords Intentions · Representation · Collaborative planning

1 Introduction

Numerous theories of intention and collaboration have been developed over the past 25 years [4,5,7,14,15,21,22,30,32,33,36]. However, the representation of the content of agent intentions has not addressed the following important issues. First, agents frequently need to communicate with others about parameters in their plans (e.g., when negotiating over constraints on parameters). Thus, the representation of such parameters must enable agents to unambiguously refer to them when communicating with other agents, even if those parameters have not yet received values. Second, agents frequently delegate parameter-binding decisions in

L. Hunsberger (✉)
Vassar College, Poughkeepsie, NY, USA
e-mail: hunsberg@cs.vassar.edu

C. L. Ortiz
SRI International, Menlo Park, CA, USA
e-mail: ortiz@ai.sri.com

21 collaborative activity. Thus, the representation of parameters in the content of an agent's
 22 intention must be able to distinguish cases where the agent is free to select the value of a
 23 parameter from cases where the agent's intention involves some parameter whose value shall
 24 be selected by some other agent. Third, a group of collaborating agents frequently delegate
 25 subsidiary tasks or goals to one or more group members. Thus, the representation of the
 26 content of intentions must be able to properly reflect the hand-off of responsibility for the
 27 subsidiary tasks or goals while still maintaining the higher-level intention "that the selected
 28 agent do the subsidiary task or accomplish the subsidiary goal." Fourth, the representation
 29 of the content of agent intentions needs to be able to address the partiality of agent intentions
 30 and plans. For example, an agent might intend to rent a car without having a particular car
 31 in mind. Fifth, the representation of the content of agent intentions needs to accommodate
 32 the evolution of agent intentions and plans over time. Thus, a basic suite of intention-update
 33 operators must be provided. Sixth, intentions in any kind of complex task are frequently
 34 related to one another. For example, if I intend to get a car by renting it, but then find out that
 35 renting a car is impossible, then I would typically revert to my original intention to get a car,
 36 which would lead me to find other ways of getting a car (e.g., by borrowing one). Thus, the
 37 representation of intentions must address the relationships between intentions in a complex
 38 plan.

39 This article presents a theory of intention representation that provides solutions to these
 40 representational problems. Insodoing, it fills an important gap in existing theories of agents,
 41 planning and collaborative planning. In a companion paper [27] we present a theory of
 42 intention revision based on this representation.

43 *Note:* Throughout this article, we use the term *intention update* to refer to the incremental
 44 modification of a single intention, as opposed to *intention revision* which, in the companion
 45 paper, refers to the process of modifying the contents of an agent's entire database of intentions
 46 to accommodate the updating of a single intention or the insertion of a new intention.

47 2 Overview of our approach

48 Suppose Alice and Bob plan to travel to California together by car. Suppose further that, as
 49 part of their plan, Alice intends to rent a car and Bob intends to drive *that car* (with both of
 50 them in it) to California. How can the content of their intentions be formally represented?
 51 A straightforward approach to representing Alice's intention might employ a formula such
 52 as

$$53 \quad \text{IntTh}(A, (\exists x)\text{Rent}(A, x) \wedge \text{Car}(x))$$

54 where *IntTh* is a modal intention operator, *A* is a constant denoting Alice, and the existentially
 55 quantified variable *x* represents an as-yet-unspecified car. The problem with such a
 56 representation of Alice's intention is that the scope of the existentially quantified variable
 57 is closed—and embedded within an intention operator—and, thus, unavailable for use when
 58 representing Bob's intention. However, the representation of Bob's intention must contain a
 59 reference to that same car.

60 Similar representational problems have been encountered when attempting to analyze
 61 sequences of sentences in a natural language. For example, consider the two sentences given
 62 below:

- 63 (1) A man walked into my room. (2) He was tall.

64 A straightforward translation of the first sentence into first-order logic (FOL) might yield
 65 a formula such as: $(\exists x)Man(x) \wedge WalkedInto(x, MyRoom)$. However, the scope of the
 66 existentially quantified variable, x , being closed complicates matters for the analysis of the
 67 second sentence. One would like to use a formula such as $Tall(x)$ for the second sentence,
 68 but x is free in that formula.

69 Kamp [20] developed Discourse Representation Theory (DRT) to address these kinds of
 70 quantifier scoping and reference problems in linguistics, which had been resistant to solution
 71 using first-order logic.¹ In DRT, certain data structures are used to represent the current state
 72 of a discourse (i.e., the information that can be derived from the current point in a discourse).
 73 The semantics of the data structures is given in terms of so-called *verifying embeddings*;
 74 alternatively a translation can be provided directly from discourse representation structures
 75 to formulas in first-order logic. Our approach to the representation of intentions is similar
 76 to that of DRT in that we use a certain kind of data structure—called a Dynamic Intention
 77 Structure (DIS)—to represent the content of an agent’s intention. As in DRT, we define
 78 the semantics of our DIS structures by providing a translation function that maps DISs to
 79 logical formulas—in our case, a version of first-order modal logic developed by Fitting and
 80 Mendelsohn [8].

81 Although our approach draws from DRT, there are several significant differences. First,
 82 the content of agent intentions is structured into a variety of different fields, each of which
 83 plays a different role in the translation of that structure into logic. Second, the treatment of
 84 parameters is richer in that it clearly distinguishes parameters for which the intending agent
 85 will select values from those for which the values will be selected by some other agent or
 86 group. Third, the content distinguishes actions that the agent intends to do from propositions
 87 that the agent intends should hold. Fourth, the content includes subsidiary boxes, representing
 88 portions of a group activity that will be the responsibility of member agents or subgroups.
 89 Many of these differences are in response to the different needs to which these structures are
 90 being put in our work. Naturally, the intentions constituting a collaboration among a group
 91 of agents are different from the sentences constituting a discourse.

92 2.1 Overview of the syntax of dynamic intention structures

93 A Dynamic Intention Structure (DIS) comes in two varieties, containing the fields shown in
 94 the boxes in Fig. 1. The box on the right, called a DIS* structure, contains an extra field, *Grp*,
 95 that specifies the group in a collaborative activity. The *Agt* field specifies the agent holding
 96 the intention. The *ID* field specifies an identifier for the intention. The *ExVars* and *DefVars*
 97 fields specify two kinds of parameters: those the intending agent is free to find values for
 98 and those whose values will be determined by some other agent(s). The *ActType* field spec-
 99 ifies the kind of action the agent intends to be done. The *Conds* field specifies conditions
 100 (or constraints) that the agent intends shall hold.² The *SubBoxes* field contains subsidiary
 101 DIS structures corresponding to subsidiary tasks or goals. The syntax of DISs will be given
 102 formally in Sect. 3, after presenting a series of motivating examples; however, we must first
 103 say a few words about semantics.

¹ Heim [18] presented a similar approach based on what she calls *file change semantics*. Groenendijk and Stokhof [12] presented still another approach based on Dynamic Predicate Logic.

² The *Conds* field contains a set of *intended conditions* (i.e., propositions that the agent intends shall hold). In contrast, this paper does not address *conditional intentions* (i.e., intentions conditioned on some proposition). Conditional intentions are treated in the companion paper [27].

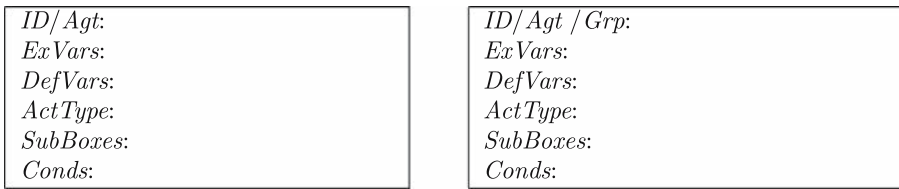


Fig. 1 Dynamic Intentions Structures: DIS (left) and DIS* (right)

104 2.2 Overview of the semantics of dynamic intention structures

105 The semantics of the intention structures defined in this article is based on their translation
 106 into a version of first-order modal logic developed by Fitting and Mendelsohn [8] having the
 107 following features.

- 108 • *The iota operator:* Terms of the form, $(\iota s. \phi(s))$, can be glossed as “the unique object s
 109 that satisfies $\phi(s)$.” More precisely, if there is a unique object in the semantic domain
 110 such that the interpretation of ϕ holds for that object, then the iota expression designates
 111 that object; otherwise, it fails to designate. The iota operator is useful for constructing
 112 *definite descriptions* such as “the car that Zoe selects”.
- 113 • *Predicate abstractions:* Predicate abstractions, which have the form, $\langle \lambda s. \phi(s) \rangle$, are
 114 similar to *lambda expressions* in the lambda calculus. The application of a predicate
 115 abstraction to the term t is notated: $\langle \lambda s. \phi(s) \rangle (t)$. If ϕ has no occurrences of modal oper-
 116 ators, then this expression is equivalent to what would, in most logics, be written as $\phi(t)$.
 117 However, things get more interesting when ϕ involves a modal operator. For example,
 118 the term t is evaluated outside the scope of the modal \Box operator in $\langle \lambda s. \Box \psi(s) \rangle (t)$,
 119 whereas it is evaluated inside the scope of the \Box operator in $\Box(\langle \lambda s. \phi(s) \rangle (t))$.

120 *Intention operators; and the Achieved and Done predicates*

121 This article presumes the existence of a suitable intention operator and uses it in the logical
 122 formulas generated by the translation of intention structures.³ The paper restricts attention to
 123 intentions having propositional content, such as “I intend that we go to the movies tonight.”
 124 We employ expressions of the form, $IntTh(g, \phi)$, where $IntTh$ is the intention operator, g is
 125 the agent holding the intention, and ϕ is the intended proposition.

126 *Intention/plan identifiers.* When agents coordinate their activities, it is useful to have iden-
 127 tifiers for their plans. For example, if Alice and Bob have a plan to drive to Boston using
 128 some as yet unspecified car, then they might arbitrarily assign an identifier such as `Plan39`
 129 to their plan, thereby enabling them to refer to “the car in `Plan39`” instead of “the car that
 130 we plan to rent for our trip to Boston”. Using identifiers in this way is especially convenient
 131 when the activities the agents are involved in might otherwise require lengthy descriptions
 132 to uniquely identify them. Similarly, in this article, we allow intentions to have identifiers
 133 associated with them. Thus, we presume that the intention operator can be augmented to
 134 include an additional *identifier* argument. For example, agent g ’s intention that ϕ , identified
 135 by id , would be represented by the formula, $IntTh(g, id, \phi)$.

³ In this article, we do not provide a semantics for the intention operator. Instead, we focus on the use of DISs to provide a fine-grained structure to the content of an intention, with an eye toward its subsequent manipulation. In Sect. 7, we present an overview of a companion paper [27], in which we address the semantics of intention in detail.

136 Although one could always distinguish intentions by adding more descriptive content
 137 such as time or place, we include identifiers for convenience, to distinguish otherwise iden-
 138 tical intentions and to simplify reference to them. However, not all intentions need have
 139 identifiers—for example, intentions that are inferred from others. To handle such cases one
 140 can simply add an axiom of the form, $IntTh(g, id, \phi) \Rightarrow IntTh(G, \phi)$, where the converse
 141 need not hold.

142 *Act types.* We presume that actions fall into types, called *act types*. Act types can be either
 143 *basic* or *complex*. A basic act type represents a class of atomic actions that, under the right
 144 circumstances, a capable agent can directly perform. A complex act type represents a class
 145 of actions that can be performed by executing a set of subsidiary tasks, commonly called a
 146 *recipe*. Thus, an agent might rent a car by walking to the rental car agency, filling out a form,
 147 paying lots of money, and so forth. Similarly, a group of agents might build a house by laying
 148 a foundation, buying some wood, nailing boards together, and so forth.

149 Following Ortiz [28], we allow act types to be partially specified using the @ constructor.
 150 An act type expression has the form, $A@arg_1(val_1) \dots @arg_n(val_n)$, where A is an act
 151 type, each arg_i is the name of an argument, and each val_i is the value of the i th argu-
 152 ment. For example, a `drive` act type, by itself, does not specify any arguments; however,
 153 `drive@obj(Car39)@to(Boston)` specifies a car and a destination.

154 *Intending to do an action.* Since all intentions in this article have propositional content, an
 155 agent g 's intention to do an action α is represented as an intention that α is done by g .
 156 For this purpose, we employ the *Done* predicate, where $Done(g, \alpha)$ is true if g has done an
 157 action of type α . In many cases, the act type expression will include an `agt` argument
 158 that specifies the agent of the action. For example, `drive@agt(Bob)@obj(Car39)`
 159 specifies the act type of Bob driving a particular car. In such a case, the agent argument
 160 of the *Done* predicate is redundant and is therefore omitted. For example, the formula,
 161 $Done(drive@agt(Bob)@obj(Car39))$, represents that Bob has done a `drive` action
 162 using `Car39`. Furthermore, Bob's intention to do the indicated `drive` action is represented
 163 as follows.

$$164 \quad intTh(Bob, Done(drive@agt(Bob)@obj(Car39)))$$

165 *Intentions in the context of collaborative activity.* The SharedPlans formalization [14, 15]
 166 specifies the mental state (i.e., intentions and beliefs) of a group of collaborating agents. In
 167 particular, when a group of agents are collaborating on a group activity α , then each agent
 168 in the group holds (among other things) an intention that can be glossed as “I intend that
 169 we do α .” Grosz and Hunsberger [13] have identified a constraint they call the Coordinated
 170 Cultivation Requirement (CCR) that constrains a collaborating agent from certain forms
 171 of unilateral decision making. In their model of collaborative activity, called the Coordi-
 172 nated Cultivation of SharedPlans (CCSP) model, intentions subject to the CCR are called
 173 Group-Activity-Related (GAR) intentions. For the purposes of this article, the effect of the
 174 CCR on an agent's subsequent cultivation of its GAR intention is not important. However, a
 175 GAR intention does require an extra argument that specifies the group to which the collabor-
 176 ating agent belongs. In this article, we represent the GAR intention held by an agent g , in a
 177 group GR , toward a proposition ϕ , with plan identifier id as follows:

$$178 \quad IntTh^*(g, id, GR, \phi)$$

179 In the case that the GAR intention concerns a group's doing of α , then it would have the
 180 following form:

$$181 \quad IntTh^*(g, id, GR, Done(\alpha@agt(GR)))$$

182 *The achievement operator.* In the context of collaborative activity, a group of agents might
 183 decide that some agent (or subgroup) g should achieve some subsidiary goal ϕ . In that case,
 184 the content of each group member's GAR intention would include a clause that can be glossed
 185 as "that g achieves ϕ ." In this article, such clauses are represented using the *Achieved* opera-
 186 tor, where an expression of the form, $Achieved(g, \phi)$, represents that the agent (or subgroup)
 187 g achieved the proposition ϕ .⁴ Thus, an agent g_1 in a collaboration might hold the following
 188 GAR intention:

$$IntTh^*(g_1, id, GR, Achieved(g_2, \phi)).$$

190 2.3 Preliminary examples of intention structures

191 This section provides some preliminary examples of Dynamic Intention Structures (DISs).
 192 The examples illustrate the syntax of DISs and their translation into first-order logic. The
 193 syntax and semantics of DISs are formally defined in Sects. 3 and 4, respectively.

194 *Example 1* Alice intends to rent a car

195 This example is interesting because Alice may intend to rent a car without having any par-
 196 ticular car in mind yet. An important requirement is that Alice should have some way of
 197 referring to the car that she intends to rent (e.g., she may later declare that the car that she
 198 intends to rent will be blue) even if she has not yet selected any such car. Here's the DIS, D_1 ,
 199 for Alice's intention:⁵

$$D_1 = \begin{array}{l} ID/Agt: \quad id1 / A \\ ExVars: \quad v_1 \\ ActType: \quad rent@obj(v_1) \\ Conds: \quad Car(v_1) \end{array}$$

201 The constant $id1$ serves as an identifier for Alice's intention. Alice is represented by the
 202 constant A . The constant v_1 is the parameter representing the car. The act type expression,
 203 $rent@obj(v_1)$, represents the rental action. The constraint, $Car(v_1)$, stipulates that the
 204 object to be rented must be a car. The translation of D_1 into first-order logic, which we notate
 205 as $\|D_1\|$, is given below.

$$\|D_1\| = IntTh(A, id1, (\exists x_1)(Done(rent@agt(A)@obj(x_1)) \wedge Car(x_1)))$$

207 Notice that each occurrence of the parameter v_1 in D_1 has been replaced by an occurrence
 208 of the existentially quantified variable x_1 in the generated formula, $\|D_1\|$. The existential
 209 quantification indicates that Alice intends that she rent some car, even if she does not yet
 210 know which car she will eventually rent. In addition, the act type expression from D_1 has
 211 been augmented to stipulate that Alice is the agent of the action. The augmented act type
 212 expression appears within the *Done* predicate in the generated formula.

213 In this example, Alice's selection of a value for the car parameter is not explicitly
 214 represented. If it is desired to do so, there are at least two alternatives: (1) represent the act
 215 of selecting a car; or (2) represent the goal of having selected a car. In this article, we focus
 216 on the latter alternative, employing a predicate, *Sel*, to represent an agent having selected an
 217 item to be the value of a parameter for a given plan.

⁴ Saying that g achieved ϕ is not a statement about g 's mental state. g might have achieved ϕ accidentally or intentionally.

⁵ To save space, only the non-empty fields in a DIS are shown in the box notation.

218 For example, consider D'_1 , given below, which is identical to D_1 , except that it has
 219 been augmented to include the proposition, $Sel(A, v_1, \text{"v}_1", id1)$, in the *Conds* field. The
 220 semantics of this extra entry is revealed by the translation of D'_1 into first-order logic.

221
$$D'_1 = \begin{array}{l} ID/Agt: \quad id1 / A \\ ExVars: \quad v_1 \\ ActType: \quad rent@obj(v_1) \\ Conds: \quad Car(v_1), Sel(A, v_1, \text{"v}_1", id1) \end{array}$$

222
$$\|D'_1\| = IntTh(A, id1, (\exists x_1)(Done(rent@agt(A)@obj(x_1)) \wedge Car(x_1) \\ 223 \wedge Sel(A, x_1, v_1, id1)))$$

224 Notice that each occurrence of the parameter v_1 in D'_1 —except for the quoted one—is trans-
 225 lated into an occurrence of the existentially quantified variable x_1 in the generated formula.
 226 In contrast, the quoted term, “ v_1 ”, is translated into v_1 . Thus, v_1 is translated into a term
 227 designating the object that A selects, whereas “ v_1 ” is translated into a constant designat-
 228 ing the name of the parameter for which that object is selected. In the generated formula,
 229 $Sel(A, x_1, v_1, id1)$, holds if the item denoted by x_1 has been selected by A to be the value of
 230 the parameter named v_1 in the plan identified by $id1$. For example, $Sel(A, Car39, v_1, id1)$
 231 holds if Alice has selected the car denoted by $Car39$ to be the value of the parameter v_1 in
 232 that plan.

233 The quoting of variables discussed above is related to the quoting of expressions in
 234 programming languages such as Lisp [34]. For example, in Lisp, an expression such as,
 235 $(let ((v 'x)) (cons v 'v))$, is syntactically valid. It evaluates to the list, $(x v)$.
 236 The reason is that v evaluates to x , and $'v$ evaluates to v . In the same sense, the structure
 237 D'_1 contains the terms v_1 and “ v_1 ”. The “evaluation” (or translation) of D'_1 yields a formula
 238 containing the corresponding terms, x_1 and v_1 . In this case, v_1 translates into x_1 , which is a
 239 logical variable denoting the car selected by the agent A; and “ v_1 ” translates into v_1 , which
 240 is a logical constant denoting a *name* that A can use to *refer* to that car—even before such a
 241 car has been selected.

242 *Example 2* Alice intends to rent whatever car Zoe selects

243 This example is similar to the first example, except that Zoe will be selecting the car that Alice
 244 rents. Thus, in addition to Alice’s intention to rent a car, we also consider Zoe’s intention to
 245 select a car for Alice to rent.

246 *Alice’s intention.* Alice’s intention is given by a DIS—call it D_2 —in which the car to be
 247 rented is represented by a *DefVar* specification—that is, a variable whose value is given by
 248 a *definite description*.

249
$$D_2 = \begin{array}{l} ID/Agt: \quad id2 / A \\ DefVars: \quad (v_2, Sel(Z, _, \text{"v}_2", id2)) \\ ActType: \quad rent@obj(v_2) \\ Conds: \quad Car(v_2) \end{array}$$

250 The *DefVars* field normally contains a list of specifications. In this case, there is only one.
 251 It stipulates that the value of the parameter named v_2 is to be whatever object satisfies the
 252 predication, $Sel(Z, _, \text{"v}_2", id2)$, where the underscore character is a place holder for the
 253 object in question. In other words, the value of the parameter v_2 is to be whatever car Zoe
 254 selects. Notice that although Zoe will be making the selection, the parameter, v_2 , and the ID,
 255 $id2$, refer to elements of Alice’s DIS, D_2 .

256 The translation of D_2 into first-order logic is given by:

$$257 \quad \|D_2\| = \text{IntTh}(\mathbb{A}, \text{id}2, (\lambda x_2. \Phi(x_2)) \Upsilon),$$

258 where:

$$259 \quad \begin{aligned} \Phi(x_2) &\equiv \text{Done}(\text{rent}@\text{agt}(\mathbb{A})@\text{obj}(x_2)) \wedge \text{Car}(x_2) \\ \Upsilon &\equiv (\iota s. \text{Sel}(\mathbb{Z}, s, v_2, \text{id}2)). \end{aligned}$$

260 Notice that the *DefVar* parameter, v_2 , is translated into a variable, x_2 , that is bound within a
 261 lambda expression in the generated formula. That lambda expression is applied to the value,
 262 $(\iota s. \text{Sel}(\mathbb{Z}, s, v_2, \text{id}2))$, which is a definite description involving the iota operator derived
 263 from the second part of the *DefVar* specification.⁶ The iota expression can be glossed as
 264 “whatever object, s , that Zoe selects to be the value of the parameter, v_2 , in Alice’s plan iden-
 265 tified by $\text{id}2$.” Hence we see an important property of DIS’s: their incrementality. Whereas
 266 in ordinary logic the elements in the scope of a quantifier cannot be accessed outside the
 267 sentence, in DIS—as in DRT—they can be immediately accessed for further modification.
 268 *Zoe’s intention*. Zoe’s intention that she select some car for Alice can be represented by
 269 the following DIS—call it \hat{D}_2 —where the parameter named \hat{v}_2 in \hat{D}_2 is distinct from the
 270 parameter named v_2 in D_2 .

$$271 \quad \hat{D}_2 = \begin{array}{|l} \text{ID/Ag}t: \quad \text{id}2b / \mathbb{Z} \\ \text{ExVars}: \quad \hat{v}_2 \\ \text{Conds}: \quad \text{Car}(\hat{v}_2) \wedge \text{Sel}(\mathbb{Z}, \hat{v}_2, “v_2”, \text{id}2) \end{array}$$

$$272 \quad \|\hat{D}_2\| = \text{IntTh}(\mathbb{Z}, \text{id}2b, (\exists \hat{x}_2)(\text{Car}(\hat{x}_2) \wedge \text{Sel}(\mathbb{Z}, \hat{x}_2, v_2, \text{id}2)))$$

273 Notice that Zoe intends to select some car \hat{x}_2 to be the value of Alice’s parameter v_2 in her
 274 plan identified by $\text{id}2$. In addition, the semantics of Zoe’s intention ensures that \hat{x}_2 will also
 275 be the value of Zoe’s parameter \hat{v}_2 , since \hat{v}_2 gets translated into \hat{x}_2 .

276 2.3.1 Intentions in the context of collaborative group activity

277 Example 2 explored the related intentions of Alice and Zoe; however, Alice and Zoe did not
 278 have a SharedPlan. Thus, there were no GAR intentions [13]. The next examples consider
 279 GAR intentions in the context of collaborative activity. In our model, a GAR intention is rep-
 280 resented by a DIS*, which has an extra field denoting the group to which the intending agent
 281 belongs. The examples also explore the *SubBoxes* field of a DIS. (Although the *SubBoxes*
 282 field can be used in the context of single-agent activity; its most interesting features arise in
 283 the context of collaborative group activity.)

284 *Example 3* Alice and Bob plan to travel to boston together

285 In this example, Alice and Bob each hold a GAR intention concerning their plan. The con-
 286 tents of their GAR intentions are identical, except for the *Ag*t field. Let GR denote the group
 287 consisting of Alice and Bob; and let g denote either Alice (\mathbb{A}) or Bob (\mathbb{B}). Then D_3 , below,
 288 is a DIS* that represents g ’s GAR intention.⁷

$$289 \quad D_3 = \begin{array}{|l} \text{ID/Ag}t / \text{Gr}p: \quad \text{id}3 / g / \text{GR} \\ \text{ActType}: \quad \text{travel}@\text{to}(\text{Boston}) \end{array}$$

⁶ The choice of s in the iota expression is arbitrary. It replaces the underscore from the *DefVar* entry.

⁷ Recall that only non-empty fields are shown in the box notation.

290 $\|D_3\| = \text{IntTh}^*(g, \text{id}_3, \text{GR}, \text{Done}(\text{travel}@_{\text{agt}}(\text{GR})@_{\text{to}}(\text{Boston})))$

291 Next, we consider a slight modification of this example in which the destination is
 292 represented by a variable instead of a constant. The idea here is to facilitate replanning
 293 should it turn out that the group will be unable to go to Boston. In that case, they might
 294 decide to change the value of their destination variable to New York. The modified DIS*,
 295 called D'_3 , is defined below. In this case, a *DefVar* is used to represent the destination of the
 296 trip. The value of this *DefVar* is represented by the constant Boston.

297
$$D'_3 = \begin{array}{l} \text{ID/Agt/Grp: } \text{id}_3 / g / \text{GR} \\ \text{DefVars: } (\text{v}_3, \text{Boston}) \\ \text{ActType: } \text{travel}@_{\text{to}}(\text{v}_3) \end{array}$$

298 $\|D'_3\| = \text{IntTh}^*(g, \text{id}_3, \text{GR}, \langle \lambda x_3. \text{Done}(\text{travel}@_{\text{agt}}(\text{GR})@_{\text{to}}(x_3)) \rangle \text{Boston})$

299 *Example 4* Alice and Bob intend that Alice should do β_1 subject to the constraint ϕ_1 and
 300 Bob should do β_2 subject to the constraint ϕ_2

301 In this example, we suppose that Alice and Bob begin by making a group decision (or agree-
 302 ment) that Alice shall do β_1 subject to the constraint ϕ_1 and Bob shall do β_2 subject to
 303 the constraint ϕ_2 . According to the CCSP model [13], such an agreement entails certain
 304 obligations on the participants.⁸ First, it obliges Alice and Bob to each adopt a GAR inten-
 305 tion concerning their group activity. Second, it obliges Alice to adopt a subsidiary intention
 306 concerning her doing of β_1 , and Bob to adopt a subsidiary intention concerning his doing
 307 of β_2 . The following discussion concentrates on the representation of these intentions. For
 308 expository convenience, the subsidiary intentions are addressed first.

309 *Alice and Bob's subsidiary intentions.* Alice's subsidiary intention concerning her doing of β_1
 310 can be represented by the DIS, D_{4a} , defined below (on the left). Similarly, Bob's subsidiary
 311 intention concerning his doing of β_2 can be represented by the DIS, D_{4b} , defined below (on
 312 the right).

313
$$D_{4a} = \begin{array}{l} \text{ID/Agt: } \text{id}_{4a} / \text{A} \\ \text{ActType: } \beta_1 \\ \text{Conds: } \phi_1 \end{array} \quad D_{4b} = \begin{array}{l} \text{ID/Agt: } \text{id}_{4b} / \text{B} \\ \text{ActType: } \beta_2 \\ \text{Conds: } \phi_1 \end{array}$$

$\|D_{4a}\| = \text{IntTh}(\text{A}, \text{id}_{4a}, \phi_1 \wedge \text{Done}(\beta_1@_{\text{agt}}(\text{A})))$ $\|D_{4b}\| = \text{IntTh}(\text{B}, \text{id}_{4b}, \phi_2 \wedge \text{Done}(\beta_2@_{\text{agt}}(\text{B})))$

314 *Alice and Bob's GAR intentions.* The content of each agent's GAR intention refers *both* to
 315 Alice's doing of β_1 and Bob's doing of β_2 . Their GAR intentions, represented by D_4 below,
 316 can be glossed as "I intend that Alice do β_1 subject to ϕ_1 and Bob do β_2 subject to ϕ_2 ." Once
 317 again, g represents Alice or Bob.

318
$$D_4 = \begin{array}{l} \text{ID/Agt/Grp: } \text{id}_4 / g / \text{GR} \\ \text{SubBoxes: } \begin{array}{l} \text{ID/Agt: } \text{id}_{4a} / \text{A} \\ \text{ActType: } \beta_1 \\ \text{Conds: } \phi_1 \end{array}, \begin{array}{l} \text{ID/Agt: } \text{id}_{4b} / \text{B} \\ \text{ActType: } \beta_2 \\ \text{Conds: } \phi_2 \end{array} \end{array}$$

319 Notice that D_4 contains two subsidiary boxes that are identical to D_{4a} and D_{4b} given above.
 320 Based on the description of the translation function given so far, the translation of D_4 into

⁸ As discussed by Grosz and Hunsberger [13], the obligations entailed by such agreements have been pointed out by many philosophers, including Bratman [2], Gilbert [10] and Tuomela [35].

321 first-order logic would generate a formula containing nested intentions. Such a formula might
 322 be glossed as, “I intend that Alice intend that she does $\beta_1 \dots$ ”. However, this does not capture
 323 the desired relationship between the GAR intention and the subsidiary activities: “I intend
 324 that Alice *do* $\beta_1 \dots$ ”.⁹ To distinguish the “stand-alone” translation of a DIS—such as the
 325 stand-alone translations of D_{4a} and D_{4b} given earlier—and the “in-context” translation of a
 326 DIS as a subsidiary box within a parent DIS, we employ two different translation functions.
 327 The *stand-alone* translation function, which is the only one that has been seen so far, gen-
 328 erates formulas involving the intention operator. The *in-context* translation function, which
 329 has not yet been seen, generates formulas involving the *Achieved* operator. For a DIS such as
 330 D_4 , which contains subsidiary boxes, the translation of everything but its subsidiary boxes
 331 is performed by the stand-alone translation function, which generates the top-level intention
 332 formula. In contrast, the translation of its subsidiary boxes is performed by the in-context
 333 translation function, which generates subsidiary formulas, Φ_1 and Φ_2 , involving the *Achieved*
 334 operator.

$$335 \quad \|D_4\| = \text{IntTh}^*(g, \text{id}4, \text{GR}, \Phi_1 \wedge \Phi_2),$$

336 where:

$$337 \quad \begin{aligned} \Phi_1 &\equiv \text{Achieved}(A, \phi_1 \wedge \text{Done}(\beta_1 @ \text{agt}(A))) \\ \Phi_2 &\equiv \text{Achieved}(B, \phi_2 \wedge \text{Done}(\beta_2 @ \text{agt}(B))) \end{aligned}$$

338 The stand-alone and in-context translation functions are formally defined in Sect. 4.

339 *Example 5* Alice and Bob plan to drive whatever car Alice selects

340 In this example, Alice and Bob have a SharedPlan to drive whatever car Alice selects. (Here,
 341 the *drive* act type represents a multi-agent action that Alice and Bob will do together.)
 342 Thus, Alice and Bob each have a GAR intention to that effect. In addition, Alice has an
 343 intention concerning her selection of a car.

344 *The GAR intentions.* The DIS*, D_5 , represents the GAR intention held by g (either Alice or
 345 Bob).

$$346 \quad D_5 = \begin{array}{l} \text{ID/Agnt/Grp: } \text{id}5 / g / \text{GR} \\ \text{DefVars: } (\nu_5, \{\text{Sel}(A, _, \text{“}\nu_5\text{”}, \text{id}5), \text{Car}(_)\}) \\ \text{ActType: } \text{drive@obj}(\nu_5) \\ \text{SubBoxes: } \begin{array}{l} \text{ID/Agnt: } \text{id}5a / A \\ \text{ExVars: } \nu_6 \\ \text{Conds: } \text{Sel}(A, \nu_6, \text{“}\nu_5\text{”}, \text{id}5) \wedge \text{Car}(\nu_6) \end{array} \end{array}$$

$$347 \quad \|D_5\| = \text{IntTh}^*(g, \text{id}5, \text{GR}, \langle \lambda x_5. \Psi_1 \wedge \Psi_2 \rangle (\text{is.Sel}(A, s, \nu_5, \text{id}5) \wedge \text{Car}(s))),$$

348 where: $\Psi_1 \equiv \text{Done}(\text{drive@agt}(\text{GR}) @ \text{obj}(x_5))$

$$\Psi_2 \equiv \text{Achieved}(A, (\exists x_6)(\text{Sel}(A, x_6, \nu_5, \text{id}5) \wedge \text{Car}(x_6)))$$

349 Notice that the *DefVar* specification for ν_5 is a set of predicates containing the underscore
 350 placeholder. These predicates are conjoined within the corresponding iota expression in the
 351 translation, $\|D_5\|$. In addition, notice that the “in context” translation of a subsidiary box
 352 yields an *Achieved*(...) clause.

353 *Alice’s intention concerning her selection of a car.* Alice’s intention concerning her selec-
 354 tion of a car is represented by a DIS—call it D_6 —that is identical to the subsidiary box in

⁹ Grosz and Hunsberger [13] discuss the relationship between GAR intentions and subsidiary activities in more detail.

355 D_5 above. However, the stand-alone translation of D_6 yields an intention formula, not an
 356 *Achieved* formula. In particular, A intends that there is a car x_6 that she selects as the value
 357 of v_5 in the DIS with id id_5 .

$ID/Agt:$	id_5a / A
$ExVars:$	v_6
$Conds:$	$Sel(A, v_6, "v_5", id_5) \wedge Car(v_6)$

358

$$\|D_6\| = IntTh(A, id_5a, (\exists x_6)(Sel(A, x_6, v_5, id_5) \wedge Car(x_6)))$$

359

360 It is important to point out that the parameters, v_5 (in D_5) and v_6 (in D_6), refer to the
 361 *same car* in the following sense. First, the only way that Alice can satisfy her intention D_6
 362 is by ensuring that there is some car—call it X —for which the predicate $Sel(A, X, v_5, id_5)$
 363 holds. (Alice would normally establish such a predication by declaring that she has made
 364 such a selection.) In such a case, the value of her parameter v_6 , which corresponds to the
 365 existentially quantified variable x_6 in the translation of D_6 , would be X . Moreover, the iota
 366 expression, $(\iota s.Sel(A, s, v_5, id_5) \wedge Car(s))$, in the formula representing the translation of
 367 D_5 would also denote X . Thus, the only way the GAR intention represented by D_5 could
 368 be satisfied would be if the value of v_5 , which corresponds to the lambda variable x_6 in the
 369 translation of D_5 , were also X .

370 3 The syntax of dynamic intention structures

371 This section presents the syntax of DIS structures, which will be slightly extended in Sect. 6.2.

372 First, we presume the following sets of symbols:

- 373 • *IdNames*, *AgtNames* and *GrpNames*—sets of constant symbols used for plan/intention
 374 identifiers (e.g., id_1, id_2, id_3 , etc.), agent names (e.g., A, B, C , etc.) and names of
 375 groups of agents (e.g., GR)
- 376 • *Constants*—a set of constant symbols that includes the above sets as subsets, but may
 377 include other symbols as well (e.g., $Boston, Car_{39}$ and $Chair_{61}$)
- 378 • *VarNames*—a set of symbols used for *ExVar* and *DefVar* parameter names (e.g., $v, v_1,$
 379 w , etc.)
- 380 • *ActTypeNames*—a set of symbols used as act-type names (e.g., $rent, travel$ and
 381 $drive$)
- 382 • *ActTypeArgs*—a set of symbols used as names of arguments within act type expressions
 383 (e.g., $agt, from, to$ and obj)
- 384 • *PredNames*—a set of predicate symbols (e.g., *Blue* or *Econ*) that does *not* include the
 385 symbol *Sel*

386 A DIS contains the following fields: *ID*, *Agt*, *ExVars*, *DefVars*, *ActType*, *SubBoxes* and *Conds*.
 387 A DIS* contains all of these fields together with a field named *Grp*. The contents of these
 388 fields are constrained as follows. Each item is numbered to facilitate future reference.

- 389 (1) The *ID* field must contain a constant, $id \in IdNames$.
- 390 (2) The *Agt* field must contain a constant, $g \in AgtNames$.
- 391 (3) The *Grp* field must contain a constant, $GR \in GrpNames$.
- 392 (4) The *ExVars* field must contain a list of zero or more constants belonging to the set
 393 *VarNames*.
- 394 (5) The *DefVars* field must contain a list of zero or more *DefVar* specifications, each having
 395 one of the following two forms:

- 396 (a) (v, X) , where:
- 397 o $v \in VarNames$, but $v \notin ExVars$; and
- 398 o X can be a constant, $c \in Constants$; or a set of zero or more propositions, each
- 399 having the form, $\phi(_, t_1, \dots, t_n)$, where $\phi \in PredNames$, $n \geq 0$, and each
- 400 $t_i \in Constants \cup VarNames$.
- 401 (b) (v, Y, g_2, id_2, v_2) , where:
- 402 o $v \in VarNames$, but $v \notin ExVars$;
- 403 o Y is a set of propositions, like X above, except that exactly one of the propo-
- 404 sitions in Y must have the form, $Sel(g_2, _, "v", id)$, where g_2 is as described
- 405 below and id is the value of the ID field for this DIS.¹⁰
- 406 o $g_2 \in AgtNames \cup GrpNames$;
- 407 o $id_2 \in IdNames$; and
- 408 o $v_2 \in VarNames$.
- 409 Note: The variables appearing as first arguments of *DefVar* specifications must be
- 410 distinct.
- 411 (6) The *ActType* field must be empty or contain a single act-type expression of the form
- 412 $type@arg_1(val_1) \dots @arg_n(val_n)$, where $type \in ActTypeNames$, $\{arg_1, \dots, arg_n\}$ is an
- 413 n -element subset of *ActTypeArgs*, and each $val_i \in Constants \cup VarNames$;
- 414 (7) The *SubBoxes* field in a DIS structure must contain a set of zero or more DIS structures;
- 415 the *SubBoxes* field in a DIS* structure can contain either DIS or DIS* structures (or
- 416 both).
- 417 (8) The *Conds* field must contain a set of zero or more conditions, each having one of the
- 418 following forms:
- 419 o $\phi(t_1, \dots, t_n)$, where $\phi \in PredNames$ and each $t_i \in VarNames \cup Constants$; or
- 420 o $Sel(g, v, "w", id)$, where g is the value of the *Agt* field for this DIS, $v \in VarNames$,
- 421 $w \in VarNames$, and $id \in IdNames$.

422 4 The semantics of dynamic intention structures

423 As has already been seen, there are two ways that a DIS can be translated into a formula

424 in first-order logic: either as a stand-alone box, which yields an intention formula, or as a

425 subsidiary box (i.e., in the context of a parent structure), which yields an *Achieved*(...) for-

426 mula. Each translation function takes a DIS or DIS* structure, D , as its input and generates

427 as its output a sentence in first-order modal logic. $\|D\|$ is the translation of D as a stand-alone

428 box, while $\|D\|^c$ is the "in context" translation of D as a subsidiary box.

429 *A note on logical combinations of intentions.* In this article we restrict attention to DISs

430 whose content involves conjunctions of propositions within the scope of various quantifiers.

431 In the companion paper [27] on intention revision, we address arbitrary logical combinations

432 of intentions, including conditional intentions, and their consequences.

433 *A note about free variables.* The syntax rules presented in Sect. 3 permit DISs to have free

434 variables. For example, a DIS might have the proposition, $Car(v)$, in its *Conds* field, but not

435 have any corresponding entry for v in its *ExVar* or *DefVar* fields. However, the translation

436 functions defined in this section, like all of the examples seen so far in this article, restrict

437 attention to DISs that do not have any free variables. Later on, in Sect. 6.2, examples involv-

¹⁰ The last three arguments, g_2 , id_2 and v_2 , are optional. They are allowed only for bookkeeping convenience when some other agent or group, g_2 , shall determine the value for v . Example 3 in Sect. 6.1 illustrates their use.

438 ing free variables are considered. At that time, the definitions of the translation functions are
439 extended to accommodate free variables.

440 *A note about the optional arguments in a DefVar specification.* The optional arguments in a
441 *DefVar* specification (cf. item 5b in Sect. 3) are for convenience only. They play no role in the
442 translation of a DIS or DIS* into a formula of first-order logic. Thus, they are not addressed
443 in this section.

444 The most general forms of a DIS D (on the left) and a DIS* D^* (on the right) are shown
445 below.

$ID/Agt:$	id / g	$ID/Agt / Grp:$	$id / g / GR$
$ExVars:$	$\forall_1, \dots, \forall_m$	$ExVars:$	$\forall_1, \dots, \forall_m$
$DefVars:$	$(w_1, X_1), \dots, (w_n, X_n)$	$DefVars:$	$(w_1, X_1), \dots, (w_n, X_n)$
$ActType:$	α	$ActType:$	α
$SubBoxes:$	S_1, \dots, S_k	$SubBoxes:$	S_1, \dots, S_k
$Conds:$	ϕ_1, \dots, ϕ_p	$Conds:$	ϕ_1, \dots, ϕ_p

447 “Stand-Alone” translation. The “stand-alone” translations of D and D^* are defined as
448 follows.

$$449 \quad \|D\| \equiv_{def} IntTh(g, id, \langle \lambda y_1. \langle \lambda y_2. \dots \langle \lambda y_n. \Omega(g) \Upsilon_n \dots \rangle \Upsilon_2 \rangle \Upsilon_1)$$

$$450 \quad \|D^*\| \equiv_{def} IntTh^*(g, id, GR, \langle \lambda y_1. \langle \lambda y_2. \dots \langle \lambda y_n. \Omega(GR) \Upsilon_n \dots \rangle \Upsilon_2 \rangle \Upsilon_1)$$

451 where:

- 452 • $\Omega(A) \equiv (\exists x_1, \dots, x_m) (\|\phi_1\| \wedge \dots \wedge \|\phi_p\| \wedge Done(\|\alpha @ agt(A)\|) \wedge \|S_1\|^c \wedge \dots \wedge \|S_k\|^c)$
- 453 • for each $j \in \{1, \dots, p\}$, $\|\phi_j\|$ is the same as ϕ_j , except that all occurrences of \forall_i , w_i and
454 “ \forall_i ” have been replaced by x_i , y_i and \forall_i , respectively.
- 455 • $\|\alpha @ agt(A)\|$ is the same as $\alpha @ agt(A)$, except that all occurrences of \forall_i and w_i have
456 been replaced by x_i and y_i , respectively;
- 457 • for each $e \in \{1, \dots, n\}$, X_e can be either a constant symbol or a set of propositions.
 - 458 ○ If X_e is a constant symbol, c , then Υ_e is simply c .
 - 459 ○ If X_e is a set of propositions, $\{\Phi_1, \dots, \Phi_{q_e}\}$, then Υ_e is the iota expression, $(\iota s_e. \bigwedge_{j=1}^{q_e} \|\Phi_j\|)$, where each $\|\Phi_j\|$ depends on the form of Φ_j , as follows.
 - 460 – If Φ_j has the form, $\phi(_, t_1, \dots, t_r)$, then $\|\Phi_j\|$ is the same as $\phi(s_e, t_1, \dots, t_r)$,
461 except that all occurrences of \forall_i and w_i have been replaced by x_i and y_i , respec-
462 tively.
 - 463 – If Φ_j has the form $Sel(g, _, “v”, id')$, then $\|\Phi_j\| \equiv Sel(g, e, v', id')$.

465 Notice that each subsidiary box, S_i , is translated using the “in context” translation function,
466 $\|\cdot\|^c$.

467 “In Context” translation. The “in context” translations of D and D^* are defined as follows.

$$468 \quad \|D\|^c \equiv_{def} \langle \lambda y_1. \langle \lambda y_2. \dots \langle \lambda y_n. Achieved(g, \Omega(g)) \Upsilon_n \dots \rangle \Upsilon_2 \rangle \Upsilon_1$$

$$469 \quad \|D^*\|^c \equiv_{def} \langle \lambda y_1. \langle \lambda y_2. \dots \langle \lambda y_n. Achieved(GR, \Omega(GR)) \Upsilon_n \dots \rangle \Upsilon_2 \rangle \Upsilon_1$$

470 where Ω and $\Upsilon_1, \dots, \Upsilon_n$ are as given above. Notice that when treating D (resp. D^*) as
471 a subsidiary box in the context of some parent box, its sub-boxes are also translated “in
472 context”, yielding *Achieved*(...) clauses.

473 **5 DIS-creation and DIS-update operations**

474 This section presents the primitive DIS-creation and DIS-update operations that are used to
 475 create new DIS structures and incrementally update existing ones. The DIS-update operations
 476 discussed in this section are restricted to cases where new data is *added* to an existing DIS;
 477 they do not address cases where existing content is *deleted* from a DIS.¹¹ Furthermore, the
 478 DIS resulting from an update operation is not guaranteed to be self-consistent. For example,
 479 it is possible, although irrational, for an agent to add the condition “not red” to its intention
 480 to rent a red car. However, such cases are easily discovered by examining the translation of
 481 the resulting DIS into FOL. Finally, even if a new or updated DIS is self-consistent, it may
 482 not be consistent with the rest of the intentions in an agent’s database of intentions. Such
 483 problems are resolved through a process of intention *revision*, which is treated in a companion
 484 paper [27].

485 In view of these restrictions, the rest of this article presumes that DIS-update operations
 486 are applied to consistent DISs, and that the resulting structures are also consistent.

487 Since the syntax of these operations is closely related to the syntax of the DIS structures
 488 presented in Sect. 3, the descriptions of most of the operations avoid needless repetition by
 489 referring to the corresponding numbered items from Sect. 3.

490 **5.1 DIS-creation operations**

491 There are two DIS-creation operations.

- 492 • $NewDIS(id, g)$ —creates a new DIS with its *ID* field set to $id \in IdNames$ and its *Agt* field
 493 set to $g \in AgtNames$ (cf. items 1 and 2 in Sect. 3).
- 494 • $NewDIS^*(id, g, GR)$ —creates a new DIS* with *ID* set to $id \in IdNames$, *Agt* set to
 495 $g \in AgtNames$, and *Grp* set to $GR \in GrpNames$ (cf. items 1, 2 and 3 in Sect. 3).

496 **5.2 DIS-update operations**

497 Each of the following DIS-update operations takes an existing DIS (or DIS*) structure as its
 498 first argument. In the descriptions below, D stands for an existing DIS (or DIS*) structure.
 499 Each DIS-update operation generates an updated version of D , which is guaranteed to be
 500 another DIS (or DIS*). Alternatively, each update operation can be viewed as destructively
 501 modifying the contents of its input DIS.

- 502 • $AddExVar(D, v)$ —Adds the parameter $v \in VarNames$ to the *ExVar* field in D (cf. item 4
 503 in Sect. 3). Only applicable if v does not already appear as an *ExVar* or *DefVar* parameter
 504 in D .
- 505 • $AddDefVar(D, v, X)$ —Adds the entry (v, X) to the *DefVars* field in D , where (v, X) is
 506 as described in item 5a of Sect. 3. Not applicable if v already appears as an *ExVar* or
 507 *DefVar* parameter in D .
- 508 • $AddDefVar(D, v, Y, g_2, id_2, v_2)$ —Adds the entry (v, Y, g_2, id_2, v_2) to the *DefVars* field
 509 in D , where (v, Y, g_2, id_2, v_2) is as described in item 5b of Sect. 3. Not applicable if v
 510 already appears as an *ExVar* or *DefVar* parameter in D .

¹¹ That is not to say that deleting information from a DIS is not important. For example, I might originally intend to rent a red car, only to find out later that red cars are way too expensive for my budget. In response, I would normally delete the “red” condition from my intention. However, the deletion of content from a DIS is beyond the scope of this article.

- 511 • *AddDefVarCond*(D, v, Φ)—Adds the condition, Φ , to the *DefVar* entry for v in D , where
- 512 Φ has the form, $\phi(_, t_1, \dots, t_n)$, as described in item 5a of Sect. 3. Only applicable if D
- 513 already contains a *DefVar* entry for v , as described in item 5b of Sect. 3.
- 514 • *AddActType*(D, α)—Adds the act type expression α to the *ActType* field of D (cf. item 6
- 515 in Sect. 3). Only applicable if the *ActType* field of D is empty.
- 516 • *AddActArg*($D, @arg(val)$)—Appends the act type argument, $@arg(val)$, to the *ActType*
- 517 entry in D . Only applicable if the *ActType* field of D is non-empty and does not already
- 518 contain an act-type argument named *arg* (cf. item 6 in Sect. 3).
- 519 • *AddSubBox*(D, D_s)—Adds D_s to the *SubBoxes* field of D . If D is a DIS, then D_s must
- 520 be a DIS; if D is a DIS*, D_s can be either a DIS or a DIS* (cf. item 7 in Sect. 3).
- 521 • *AddConds*($D, \{\Phi_1, \dots, \Phi_p\}$)—Adds the conditions, Φ_1, \dots, Φ_p , to the *Conds* field in
- 522 D , where each Φ_i must have one of the forms, $\phi(t_1, \dots, t_n)$ or *Sel*($g, v, "w", id$), as
- 523 described in item 8 of Sect. 3.
- 524 • *ShiftVar*(D, v, X)—Removes the parameter v from the *ExVar* field of D , and adds the
- 525 entry, (v, X) , to the *DefVar* field of D , where (v, X) is as described in item 5a of Sect. 3.
- 526 Only applicable if D already has v as an *ExVar* entry.
- 527 • *ShiftVar*(D, v, Y, g_2, id_2, v_2)—Removes the parameter v from the *ExVar* field of D , and
- 528 adds the entry, (v, Y, g_2, id_2, v_2) , to the *DefVar* field of D , where (v, Y, g_2, id_2, v_2) is as
- 529 described in item 5b of Sect. 3. Only applicable if D already has v as an *ExVar* entry.

530 6 Sample scenario illustrating dynamic intention structures

531 This section presents a dynamic scenario involving four agents—Alice, Bob, Chris and
 532 Zoe—represented as A, B, C and Z, respectively. Alice, Bob and Chris constitute a group,
 533 GR, planning to travel to Boston together. The scenario is dynamic because, we presume, the
 534 GAR intentions held by the agents motivate them to participate in group decision-making
 535 processes aimed at elaborating their partially specified plan. When they successfully arrive at
 536 a group decision (e.g., that Alice should be the one to get the car), it obliges them to update
 537 their GAR intentions and, in some cases, to adopt new intentions. Although the examples
 538 in this section may make reference to the decisions that prompted the agents to adopt new
 539 intentions or update existing intentions, the focus here is on representing the intentions them-
 540 selves using DIS structures and on the operations that create new DISs or update existing
 541 DISs.¹²

542 In the process of elaborating their plan, the group decides that Alice should be the one
 543 to get the car. In response, Alice subsequently adopts a separate intention to rent a car from
 544 Zoe. Since this side scenario is simpler, we begin with it.

545 6.1 Side scenario: Alice rents a car

546 For the sake of expositional simplicity, we assume that Alice and Zoe do not have all of
 547 the intentions required of a SharedPlan. In particular, they have no GAR intentions. Instead,
 548 each simply has an intention concerning certain aspects of the rental action. The main sce-
 549 nario, treated afterward, examines intentions (including GAR intentions) in the context of a
 550 SharedPlan.

¹² Grosz and Hunsberger [13] discuss in detail the relationships between GAR intentions, group decision-making processes, group decisions, obligations and intention updates.

551 *Example 1* Alice intends to rent a car

552 This example has already been encountered (cf. Example 1 in Sect. 2). However, here we
 553 provide some additional details. Alice's intention to rent a car can be represented by a DIS.
 554 The creation of that DIS can be decomposed into several primitive operations—namely, to
 555 create a new DIS and then update it. The DIS—call it D_A —representing Alice's intention to
 556 rent a car is created by the following operations.

- 557 $[\sigma_{1.1}] D_A = \text{NewDIS}(\text{id}_A, A)$ —Create a new DIS with id id_A for the agent A .
 558 $[\sigma_{1.2}] \text{AddExVar}(D_A, w_c)$ —Add a new *ExVar* parameter named w_c to D_A .
 559 $[\sigma_{1.3}] \text{AddConds}(D_A, \{\text{Car}(w_c)\})$ —Add the condition, $\text{Car}(w_c)$, to D_A .
 560 $[\sigma_{1.4}] \text{AddActType}(D_A, \text{rent@obj}(w_c))$ —Add the act-type, $\text{rent@obj}(w_c)$, to D_A .

561 Here is the resulting DIS and its translation into first-order logic:

$$562 \quad D_A = \begin{array}{l} \text{ID/Ag}t: \quad \text{id}_A / A \\ \text{ExVars}: \quad w_c \\ \text{ActType}: \quad \text{rent@obj}(w_c) \\ \text{Conds}: \quad \text{Car}(w_c) \end{array}$$

$$563 \quad \|D_A\| = \text{IntTh}(A, \text{id}_A, (\exists y_c)(\text{Done}(\text{rent@agt}(A)\text{@obj}(y_c)) \wedge \text{Car}(y_c)))$$

564 *Example 2* Alice decides that the car she rents should be Blue

565 Alice's decision in this case is, in effect, a commitment to perform a single primitive update
 566 operation, $\sigma_{2.1}$.

- 567 $[\sigma_{2.1}] \text{AddConds}(D_A, \{\text{Blue}(w_c)\})$ —Add the condition, $\text{Blue}(w_c)$, to D_A .

568 Here is the updated D_A and its translation into first-order logic:

$$569 \quad D_A = \begin{array}{l} \text{ID/Ag}t: \quad \text{id}_A / A \\ \text{ExVars}: \quad w_c \\ \text{ActType}: \quad \text{rent@obj}(w_c) \\ \text{Conds}: \quad \text{Car}(w_c), \text{Blue}(w_c) \end{array}$$

$$570 \quad \|D_A\| = \text{IntTh}(A, \text{id}_A, (\exists y_c)(\text{Done}(\text{rent@agt}(A)\text{@obj}(y_c)) \wedge \text{Car}(y_c) \wedge \text{Blue}(y_c)))$$

571 *Example 3* Alice decides to rent the car from Zoe

572 This situation might arise, for example, if Alice (the customer) and Zoe (the rental car agent)
 573 made an agreement stipulating that Zoe would select a car for Alice, and Alice would rent
 574 that car. (Some features of this example are similar to those seen earlier in Example 5 of
 575 Sect. 2.) Such an agreement would entail obligations on Alice and Zoe. In particular, Alice
 576 would be obliged to rent whatever car Zoe selects, and Zoe would be obliged to select a car
 577 for Alice. In response, Alice updates her intention to reflect that Zoe will be responsible for
 578 selecting the car, and Zoe adopts an intention to select a car for Alice. (These responses could
 579 occur simultaneously or in either order.)

580 *Alice's intention update.* Alice updates her intention by applying the update operations $\sigma_{3.1}$
 581 through $\sigma_{3.6}$, listed below, to her pre-existing DIS. Notice that part of Alice's updated intention
 582 stipulates that Zoe shall select a blue car for her. This part of Alice's intention is represented
 583 by a subsidiary box, D_s , generated by the update operations $\sigma_{3.3}$ through $\sigma_{3.6}$.

- 584 $[\sigma_{3.1}] \text{AddActArg}(D_A, \text{@from}(Z))$ —Add the argument, $\text{@from}(Z)$, to the act type in D_A .

- 585 [$\sigma_{3.2}$] *ShiftVar*($D_A, w_c, \{Sel(Z, _, "w_c", idA), Car(_), Blue(_)\}, Z, idZ, u_c$)
 586 — Shift the variable associated with w_c in D_A from the *ExVar* category to the *DefVar*
 587 category. The value of this variable is to be determined by the given set of conditions.
 588 The arguments, Z, idZ and u_c , indicate that Zoe shall be selecting the value of w_c ,
 589 that the ID of Zoe's DIS is idZ , and the parameter name for the car within Zoe's DIS
 590 is u_c .¹³
- 591 [$\sigma_{3.3}$] $D_s = NewDIS(idZ, Z)$ —Create the new box, D_s .
- 592 [$\sigma_{3.4}$] *AddExVar*(D_s, u_c)—Add a new *ExVar* parameter named u_c to D_s .
- 593 [$\sigma_{3.5}$] *AddConds*($D_s, \{Sel(Z, u_c, "w_c", idA), Car(u_c), Blue(u_c)\}$)
 594 — Add the conditions, $Sel(Z, u_c, "w_c", idA), Car(u_c)$ and $Blue(u_c)$, to the box,
 595 D_s .
- 596 [$\sigma_{3.6}$] *AddSubBox*(D_A, D_s)—Add D_s as a subsidiary box to D_A .
- 597 Here is Alice's updated DIS and its translation into first-order logic.

	$ID/Agt:$ idA / A $DefVars:$ $(w_c, \{Sel(Z, _, "w_c", idA), Car(_), Blue(_)\}, Z, idZ, u_c)$ $ActType:$ $rent@obj(w_c)@from(Z)$		
$D_A =$	<table border="1" style="border-collapse: collapse; width: 100%;"> <tr> <td style="width: 15%;"></td> <td style="padding: 5px;"> $ID/Agt:$ idZ / Z $ExVars:$ u_c $Conds:$ $Sel(Z, u_c, "w_c", idA), Car(u_c), Blue(u_c)$ </td> </tr> </table>		$ID/Agt:$ idZ / Z $ExVars:$ u_c $Conds:$ $Sel(Z, u_c, "w_c", idA), Car(u_c), Blue(u_c)$
	$ID/Agt:$ idZ / Z $ExVars:$ u_c $Conds:$ $Sel(Z, u_c, "w_c", idA), Car(u_c), Blue(u_c)$		
	$Conds:$ $Car(w_c), Blue(w_c)$		

599 $\|D_A\| = IntTh(A, idA, \langle \lambda y_c. \Phi_1 \wedge \Phi_2 \rangle \Upsilon)$

600 where:

601 $\Phi_1 \equiv Done(rent@agt(A)@obj(y_c)@from(Z)) \wedge Car(y_c) \wedge Blue(y_c)$
 602 $\Phi_2 \equiv Achieved(Z, (\exists z_c)(Sel(Z, z_c, w_c, idA) \wedge Car(z_c) \wedge Blue(z_c)))$
 603 $\Upsilon \equiv (\exists s.Sel(Z, s, w_c, idA) \wedge Car(s) \wedge Blue(s))$

604 *Zoe's new intention.* Zoe's new intention is represented by a DIS—call it D_Z —that, in this
 605 case, is identical to the subsidiary box, D_s , seen above. However, it is important to distinguish
 606 D_s and D_Z since updates that Zoe makes to her intention will be reflected in D_Z , but need
 607 not be reflected in D_s . For example, Zoe might not inform Alice that she plans to facilitate
 608 her selection of a car by putting on her glasses. The subsidiary box D_s represents what Alice
 609 intends Zoe to achieve for her; this information need not include a complete plan for how Zoe
 610 makes her selection. In contrast, D_Z represents Zoe's intention as it evolves over time. The
 611 operations $\sigma_{3.3}, \dots, \sigma_{3.6}$ above can be used to create D_Z , whose "stand-alone" translation
 612 into first-order logic is given by:

613 $\|D_Z\| = IntTh(Z, idZ, (\exists z_c)(Sel(Z, z_c, w_c, idA) \wedge Car(z_c) \wedge Blue(z_c)))$

614 *Example 4* Alice tells Zoe that the car should be an economy car

615 *Alice.* Below are the update operations that are applied to Alice's DIS.¹⁴

¹³ It would be perfectly fine for the parameter name in Zoe's DIS to be the same as the corresponding parameter name in Alice's DIS. They are shown here as different just to highlight the possibility.

¹⁴ The redundancy in these update operations derives from the following: (1) Alice intends that the car be an economy model; (2) Alice intends that Zoe select an economy model; and (3) Alice intends to rent whatever economy model, blue car Zoe selects. It might be possible to remove some of this redundancy by, for example, making the subsidiary box D_s one of the arguments of the *DefVar* specification. However, this would complicate both the syntax and semantics; thus, we do not explore it here.

- 616 [$\sigma_{4.1}$] *AddConds*($D_A, \{Econ(w_c)\}$)—Add the constraint, $Econ(w_c)$, to D_A .
 617 [$\sigma_{4.2}$] *AddDefVarCond*($D_A, w_c, Econ(_)$)—Add the constraint, $Econ(_)$, to the *DefVar*
 618 entry for w_c .
 619 [$\sigma_{4.3}$] *AddConds*($D_s, \{Econ(u_c)\}$)—Add the constraint, $Econ(u_c)$, to the subsidiary box,
 620 D_s .

621 Below are the resulting DIS and its translation into first-order logic.

622 $D_A =$

<i>ID/Agnt</i> :	idA / A						
<i>DefVars</i> :	$(w_c, \{Sel(Z, _, "w_c", idA), Car(_), Blue(_), Econ(_)\}, Z, idZ, u_c)$						
<i>ActType</i> :	$rent@obj(w_c)@from(Z)$						
<i>SubBoxes</i> :	<table border="1" style="border-collapse: collapse; width: 100%;"> <tr> <td style="padding: 5px;"><i>ID/Agnt</i>:</td> <td style="padding: 5px;">idZ / Z</td> </tr> <tr> <td style="padding: 5px;"><i>ExVars</i>:</td> <td style="padding: 5px;">u_c</td> </tr> <tr> <td style="padding: 5px;"><i>Conds</i>:</td> <td style="padding: 5px;">$Sel(Z, u_c, "w_c", idA), Car(u_c), Blue(u_c), Econ(u_c)$</td> </tr> </table>	<i>ID/Agnt</i> :	idZ / Z	<i>ExVars</i> :	u_c	<i>Conds</i> :	$Sel(Z, u_c, "w_c", idA), Car(u_c), Blue(u_c), Econ(u_c)$
<i>ID/Agnt</i> :	idZ / Z						
<i>ExVars</i> :	u_c						
<i>Conds</i> :	$Sel(Z, u_c, "w_c", idA), Car(u_c), Blue(u_c), Econ(u_c)$						
<i>Conds</i> :	$Car(w_c), Blue(w_c), Econ(w_c)$						

623 $\|D_A\| = IntTh(A, idA, \langle \lambda y_c. \Phi'(y_c) \Upsilon' \rangle)$

624 where:

625 $\Phi'(y_c) \equiv Done(rent@agt(A)@obj(y_c)@from(Z)) \wedge Car(y_c) \wedge Blue(y_c) \wedge Econ(y_c)$

626 $\Upsilon' \equiv (\lambda s. Sel(Z, s, w_c, idA) \wedge Car(s) \wedge Blue(s) \wedge Econ(s))$

627 *Zoe*. We assume that *Zoe* accepts Alice's new constraint that the car be an economy model.¹⁵

628 Thus, *Zoe* decides to update her intention accordingly. In this case, the intention update is
 629 modeled by the following operation (which is nearly identical to $\sigma_{4.3}$ above).

- 630 [$\sigma_{4.4}$] *AddConds*($D_Z, \{Econ(u_c)\}$)—Add the constraint, $Econ(u_c)$, to D_Z .

631 Below are the updated version of D_Z and its "stand-alone" translation.

632 $D_Z =$

<i>ID/Agnt</i> :	idZ / Z
<i>ExVars</i> :	u_c
<i>Conds</i> :	$Sel(Z, u_c, "w_c", idA), Car(u_c), Blue(u_c), Econ(u_c)$

633 $\|D_Z\| = IntTh(Z, idZ, (\exists z_c)(Sel(Z, z_c, w_c, idA) \wedge Car(z_c) \wedge Blue(z_c) \wedge Econ(z_c)))$

634 Notice that constraints inserted by the intention-update operations $\sigma_{4.1}, \dots, \sigma_{4.4}$ are only
 635 superficially different. The different versions could be obtained by applying the single pred-
 636 icate abstraction, $\langle \lambda s. Econ(s) \rangle$, to the respective terms, $w_c, _, u_c$, and u_c .

637 6.2 Main scenario: group travels to Boston

638 *Note*. Now that several DIS-creation and DIS-update operations have been demonstrated, the
 639 operations for the examples below will only be given English glosses when they are needed
 640 for clarification.

641 *Example 1* Alice, Bob and Chris decide to travel to Boston together

642 Following Grosz and Hunsberger's Coordinated Cultivation of SharedPlans (CCSP) model
 643 [13], we assume that in response to such a decision, each of the agents adopts a GAR intention

¹⁵ Again, the focus here is not on the communication between Alice and *Zoe*, or on any subsequent decisions. Instead, it is on intention-update operations and the resulting intentions, which are represented by DISs.

644 that the group does the travel action. The GAR intentions held by the different individuals in
 645 the group all have the same content, except for the *Agt* entry. Let g be A, B or C (i.e., Alice,
 646 Bob or Chris). Let D^* be the DIS* representing agent g 's GAR intention. Then D^* can be
 647 generated by the following operations.

648 $[\tau_{1.1}] D^* = \text{NewDIS}^*(\text{idGR}, g, \text{GR})$
 649 $[\tau_{1.2}] \text{AddDefVar}(D^*, v_{dn}, \text{Boston})$
 650 $[\tau_{1.3}] \text{AddActType}(D^*, \text{travel@to}(v_{dn}))$

651 Below are the resulting D^* and its translation into first-order logic.

652 $D^* = \begin{array}{l} \text{ID/Agt /Grp: } \text{idGR} / g / \text{GR} \\ \text{DefVars: } (v_{dn}, \text{Boston}) \\ \text{ActType: } \text{travel@to}(v_{dn}) \end{array}$

653 $\|D^*\| = \text{IntTh}^*(g, \text{idGR}, \text{GR}, \langle \lambda x_{dn}. \text{Done}(\text{travel@agt}(\text{GR})@\text{to}(x_{dn})) \rangle \text{Boston})$

654 *Example 2* The group decides that they will travel to Boston by getting and driving a
 655 car

656 The group's decision is to travel to Boston by doing two subsidiary actions involving a
 657 single car: one of them will get the car (*get*); and one of them will drive it (*drive*). This
 658 group decision obliges each agent to update its corresponding GAR intention. The composite
 659 update can be broken down into the following primitive DIS-update operations:

660 $[\tau_{2.1}] \text{AddExVar}(D^*, v_c), \text{AddExVar}(D^*, v_g), \text{AddExVar}(D^*, v_d)$ —Variables represent-
 661 ing the car, the *get* agent, and the *drive* agent.
 662 $[\tau_{2.2}] \text{AddConds}(D^*, \{\text{Car}(v_c)\})$
 663 $[\tau_{2.3}] D_1 = \text{NewDIS}(\text{idSub1}, v_g)$ —Subsidiary box for the *get* action.
 664 $[\tau_{2.4}] \text{AddActType}(D_1, \text{get@obj}(v_c))$
 665 $[\tau_{2.5}] \text{AddSubBox}(D^*, D_1)$
 666 $[\tau_{2.6}] D_2 = \text{NewDIS}(\text{idSub2}, v_d)$ —Subsidiary box for the *drive* action.
 667 $[\tau_{2.7}] \text{AddActType}(D_2, \text{drive@obj}(v_c)@\text{to}(v_{dn}))$
 668 $[\tau_{2.8}] \text{AddSubBox}(D^*, D_2)$

669 The resulting DIS* and its translation into first-order logic are given below.

670 $D^* = \begin{array}{l} \text{ID/Agt /Grp: } \text{idGR} / g / \text{GR} \\ \text{ExVars: } v_g, v_d, v_c \\ \text{DefVars: } (v_{dn}, \text{Boston}) \\ \text{SubBoxes: } \begin{array}{l} \text{ID/Agt: } \text{idSub1} / v_g \\ \text{ActType: } \text{get@obj}(v_c) \end{array}, \\ \begin{array}{l} \text{ID/Agt: } \text{idSub2} / v_d \\ \text{ActType: } \text{drive@obj}(v_c)@\text{to}(v_{dn}) \end{array} \\ \text{Conds: } \text{Car}(v_c) \end{array}$

671 $\|D^*\| = \text{IntTh}^*(g, \text{idGR}, \text{GR}, \langle \lambda x_{dn}. (\exists x_g, x_d, x_c)(\Phi_1 \wedge \Phi_2 \wedge \text{Car}(x_c)) \rangle \text{Boston})$

672 where:

673 $\Phi_1 \equiv \text{Achieved}(x_g, \text{Done}(\text{get@agt}(x_g)@\text{obj}(x_c)))$

674 $\Phi_2 \equiv \text{Achieved}(x_d, \text{Done}(\text{drive@agt}(x_d)@\text{obj}(x_c)@\text{to}(x_{dn})))$

675 *Translating subsidiary boxes that contain free variables.* Notice that in the above example,
 676 D^* contains four parameters: v_g, v_d, v_c and v_{dn} . In the translation of D^* into a logical formula,
 677 each occurrence of these variables is translated into an occurrence of x_g, x_d, x_c or x_{dn} ,
 678 respectively. Notice further that, viewed as stand-alone boxes, the subsidiary boxes within
 679 D^* include free-variable occurrences of v_g, v_d, v_c and v_{dn} , including within the *Agt* and
 680 *Grp* fields. Thus, we first extend the syntax rules for these fields:

- 681 (2') The *Agt* field must contain either a single constant, $g \in \text{AgtNames}$, or a single variable,
 682 $v \in \text{VarNames}$.
 683 (3') The *Grp* field must contain a constant, $GR \in \text{GrpNames}$, or a variable, $v \in \text{VarNames}$.

684 Next, we must extend the “in context” translation function to accommodate the presence of
 685 free variables within the subsidiary boxes of D^* . The proper “in context” translation of these
 686 variables should be consistent with occurrences of those same variables in the parent box, D^* .
 687 Thus, occurrences of v_g inside the subsidiary boxes should be translated into occurrences
 688 of x_g , and so forth. To ensure that this happens, we include an *environment* as an optional
 689 argument to the “in context” translation function, $\| \cdot \|_c^e$. An environment is simply a list of
 690 pairs, where each pair has the form (v, x) , where v is a parameter (either *ExVar* or *DefVar*)
 691 and x is the logical variable that v is translated into. When performing the “in context” trans-
 692 lation of a subsidiary box, S , that resides within some DIS D , the translation of S is given
 693 by:

694 $\|S, E\|_c^e \equiv \|S\|_c^e$, except that all free occurrences of parameters in S must be translated
 695 according to the corresponding entry in the environment E —where E contains an entry
 696 for each *ExVar* and each *DefVar* parameter in D .

697 The example given above illustrates the “in context” translation of subsidiary boxes that
 698 contain free variables that are “captured” by the parent box.

699 *Example 3* The group decides Alice should be the one to get (and thereby select) the car
 700 *updating* D^* (i.e., the *GAR intentions held by each agent in the group*).

701 This decision obliges each agent to update its *GAR* intention to reflect (1) the selection of
 702 A as the value for the agent variable, v_g , in their plan; and (2) that Alice’s getting of the
 703 car should determine which car they use in their plan (i.e., that the value of the variable v_c
 704 should be whatever car Alice gets). The first update is accomplished by shifting the variable
 705 v_g in the box D^* from the *ExVar* to the *DefVar* category, and giving it the value A (cf.
 706 $\tau_{3.1}$ below). The second update is accomplished by similarly shifting the variable, v_c , from
 707 the *ExVar* to the *DefVar* category; however, in this case, the value of v_c is determined by
 708 a set of conditions (cf. $\tau_{3.2}$ below). Simultaneously, the subsidiary box for the *get* action
 709 needs to have a new *ExVar* variable representing the car that Alice gets (cf. $\tau_{3.3}$ and $\tau_{3.4}$
 710 below).

- 711 $[\tau_{3.1}] \text{ShiftVar}(D^*, v_g, A)$
 712 $[\tau_{3.2}] \text{ShiftVar}(D^*, v_c, \{Done(\text{get}@agt(v_g)\text{obj}(_)), Car(_)\}, A, \text{idSub1}, w_c)$
 713 $[\tau_{3.3}] \text{AddExVar}(D_1, w_c)$
 714 $[\tau_{3.4}] \text{AddConds}(D_1, \{Car(w_c)\})$

715 Below are the updated version of D^* and its translation into first-order logic.

716 $D^* =$

<i>ID/Agt/Grp:</i>	$idGR / g / GR$																
<i>ExVars:</i>	v_d																
<i>DefVars:</i>	$(v_{dn}, Boston), (v_g, A),$ $(v_c, \{Done(get@agt(v_g)@obj(_)), Car(_)\}, A, idSub1, w_c)$																
<i>SubBoxes:</i>	<table border="1" style="border-collapse: collapse; width: 80%; margin-bottom: 5px;"> <tr> <td style="padding: 2px;"><i>ID/Agt:</i></td> <td style="padding: 2px;">$idSub1 / v_g$</td> </tr> <tr> <td style="padding: 2px;"><i>ExVars:</i></td> <td style="padding: 2px;">w_c</td> </tr> <tr> <td style="padding: 2px;"><i>ActType:</i></td> <td style="padding: 2px;">$get@obj(w_c)$</td> </tr> <tr> <td style="padding: 2px;"><i>Conds:</i></td> <td style="padding: 2px;">$Car(w_c)$</td> </tr> </table> <table border="1" style="border-collapse: collapse; width: 80%;"> <tr> <td style="padding: 2px;"><i>ID/Agt:</i></td> <td style="padding: 2px;">$idSub2 / v_d$</td> </tr> <tr> <td style="padding: 2px;"><i>ExVars:</i></td> <td style="padding: 2px;"></td> </tr> <tr> <td style="padding: 2px;"><i>ActType:</i></td> <td style="padding: 2px;">$drive@obj(v_c)@to(v_{dn})$</td> </tr> <tr> <td style="padding: 2px;"><i>Conds:</i></td> <td style="padding: 2px;"></td> </tr> </table>	<i>ID/Agt:</i>	$idSub1 / v_g$	<i>ExVars:</i>	w_c	<i>ActType:</i>	$get@obj(w_c)$	<i>Conds:</i>	$Car(w_c)$	<i>ID/Agt:</i>	$idSub2 / v_d$	<i>ExVars:</i>		<i>ActType:</i>	$drive@obj(v_c)@to(v_{dn})$	<i>Conds:</i>	
<i>ID/Agt:</i>	$idSub1 / v_g$																
<i>ExVars:</i>	w_c																
<i>ActType:</i>	$get@obj(w_c)$																
<i>Conds:</i>	$Car(w_c)$																
<i>ID/Agt:</i>	$idSub2 / v_d$																
<i>ExVars:</i>																	
<i>ActType:</i>	$drive@obj(v_c)@to(v_{dn})$																
<i>Conds:</i>																	
<i>Conds:</i>	$Car(v_c)$																

717 $\|D^*\| = IntTh^*(g, idGR, GR, (\lambda x_{dn}. (\lambda x_g. (\lambda x_c. (\exists x_d)(\Phi'_1 \wedge \Phi_2 \wedge Car(x_c))) \Upsilon)A)Boston)$

718 where: $\Phi'_1 \equiv Achieved(x_g, (\exists y_c)Done(get@agt(x_g)@obj(y_c)) \wedge Car(y_c))$

$\Phi_2 \equiv Achieved(x_d, Done(drive@agt(x_d)@obj(x_c)@to(x_{dn})))$

$\Upsilon \equiv (\exists s.Done(get@agt(x_g)@obj(s)) \wedge Car(s))$

719 *Alice.* In response to the group decision that Alice should get the car, and insodoing select the
720 car for the group activity, Alice is obliged to get the car. Thus, she adopts a new, subsidiary
721 intention to get a car. Although each of the agents in the group holds the high-level GAR
722 intention, only Alice adopts the subsidiary intention aimed at getting the car.

723 Let D_1^A be the DIS representing Alice's intention that she get a car. D_1^A is nearly identical
724 to the subsidiary box, D_1 , discussed above.¹⁶ The only difference is that the *Agt* field contains
725 the constant A instead of the *DefVar*, v_g . (Later examples will illustrate cases where there
726 are greater differences between a subsidiary DIS and the corresponding DIS representing an
727 adopted intention.)

728 $D_1^A =$

<i>ID/Agt:</i>	$idSub1 / A$
<i>ExVars:</i>	w_c
<i>ActType:</i>	$get@obj(w_c)$
<i>Conds:</i>	$Car(w_c)$

729 $\|D_1^A\| = IntTh(A, idSub1, (\exists y_c)(Done(get@agt(A)@obj(y_c)) \wedge Car(y_c)))$

730 *Example 4* Alice decides to get a car by renting it

731 In response to her decision, not only must Alice update her original intention to reflect
732 that she will be getting a car by renting it, but also she must adopt a subsidiary intention to
733 rent a car that, if satisfied, will necessarily satisfy her original intention. In fact, Alice will
734 normally suspend processing of her intention to get a car, while she focuses on processing
735 her intention to rent a car. The group need not know anything about how Alice is getting a
736 car; thus, this decision of Alice's need not have any impact on the GAR intentions held by
737 each agent in the group.

738 *Updating Alice's intention to get a car.* Because Alice intends to get a car by renting it, the
739 variable representing the car in D_1^A must be changed from the *ExVar* category to the *Def-*
740 *Var* category. By doing this, Alice's intention is effectively changed from "I intend to get a

¹⁶ Since the DIS-creation and DIS-update operations are so similar to those used to create D_1 , they are not presented here.

741 car” to “I intend to get whatever car I rent.” Notice that this implicit kind of selection does
 742 not employ explicit selection, for example, as represented by the $Sel(\dots)$ predicate in prior
 743 examples. Below are the update, $\tau_{4.1}$, the resulting DIS, and its translation.

744 $[\tau_{4.1}] \text{ ShiftVar}(D_1^A, w_c, \{Done(\text{rent}@agt(A)@obj(_)), Car(_)\}, A, \text{idSub1r}, u_c)$

$$745 \quad D_1^A = \begin{array}{l} ID/Agt: \quad \text{idSub1} / A \\ DefVars: \quad (w_c, \{Done(\text{rent}@agt(A)@obj(_)), Car(_)\}, A, \text{idSub1r}, u_c) \\ ActType: \quad \text{get}@obj(w_c) \\ Conds: \quad Car(w_c) \end{array}$$

746 $\|D_1^A\| = \text{IntTh}(A, \text{idSub1}, \langle \lambda y_c. Done(\text{get}@agt(A)@obj(y_c)) \wedge Car(y_c) \rangle \Upsilon_r)$
 747 where: $\Upsilon_r = (\text{is}.Done(\text{rent}@agt(A)@obj(s)) \wedge Car(s))$.

748 *Alice’s new intention to rent a car:* Let D_{1r}^A be the new DIS representing Alice’s new intention
 749 to rent a car. Like the DIS for her previous intention to get a car, D_{1r}^A will contain an existential
 750 variable representing the car, a constraint that the item be a car, and an act type—in this case,
 751 representing her rental action. Below are D_{1r}^A and its translation into first-order logic.

$$752 \quad D_{1r}^A = \begin{array}{l} ID/Agt: \quad \text{idSub1r} / A \\ ExVars: \quad u_c \\ ActType: \quad \text{rent}@obj(u_c) \\ Conds: \quad Car(u_c) \end{array}$$

753 $\|D_{1r}^A\| = \text{IntTh}(A, \text{idSub1r}, (\exists z_c)(Done(\text{rent}@agt(A)@obj(z_c)) \wedge Car(z_c)))$

754 *Recalling the side scenario involving Alice and Zoe.* A quick glance back at Alice’s intention
 755 to rent a car that began the side scenario from Sect. 6.1 will reveal that the DIS in that section
 756 is essentially the same as D_{1r}^A , above—the only difference being the choice of names for
 757 the constants. Thus, the main scenario now continues, assuming that Alice has decided to
 758 rent a car by having Zoe select it (a blue car, of course); and that Zoe adopts an intention to
 759 select such a car for Alice; and that Alice later informs Zoe that she wants the car to be an
 760 economy car.

761 *Example 5* The group decides that Bob should drive the car

762 As with the group’s decision that Alice should get the car, their decision that Bob should
 763 drive the car obliges each agent to update its corresponding GAR intention. In addition, it
 764 obliges Bob to adopt a new, subsidiary intention aimed at the “drive” action. The updating
 765 of the GAR intention held by each agent is similar to the updating in the group’s selection
 766 of Alice to do the “get” action; however, the representation of Bob’s intention concerning
 767 the “drive” action is more complex, primarily because Bob is supposed to drive to whatever
 768 destination is chosen by the group, using whatever car is chosen by whatever agent is chosen
 769 by the group to do the “get” action. Notice that Alice, the car and the destination are all
 770 represented by free variables in the subsidiary box corresponding to the “drive” action (in
 771 the most recent version of D^*). As of this moment, the group has chosen Alice to do the
 772 “get” action, but, let us suppose, she has not yet chosen a car. Similarly, the group has chosen
 773 Boston as its destination. The representation of Bob’s intention should not simply hardwire
 774 the current choices for the values of the relevant free variables since the group might later
 775 decide to change those values. For example, the group might decide to select a different agent
 776 for the “get” action, thereby invalidating Alice’s current choice of a car; or the group might
 777 decide to go to New York, thereby invalidating the current choice of destination. Similarly,
 778 if Alice had already chosen a car, Bob’s intention must not simply hardwire that choice of

779 car since she might later change her mind. Thus, the DIS representing Bob’s intention must
780 be carefully constructed to be robust to these sorts of changes.

781 *Updating the GAR intentions.* The updating of the GAR intentions in this case is much sim-
782 pler than in the earlier case in which the group selected Alice to do the “get” action (cf.
783 Example 3, above) since the group is not requiring Bob to select values for any parameters.
784 The only required update is to shift the parameter, v_d , from the *ExVar* to the *DefVar* category.

785 $[\tau_{5.1}] \text{ShiftVar}(D^*, v_d, B)$

786 Here’s the updated version of D^* , representing the GAR intention held by each agent in
787 the group, together with its translation into first-order logic.

$$788 \quad D^* = \begin{array}{l} \text{ID/Agnt/Grp:} \quad \text{idGR} / g / \text{GR} \\ \text{DefVars:} \quad (v_{dn}, \text{Boston}), (v_g, A), \\ \quad (v_c, \{\text{Done}(\text{get}@agt(v_g)@obj(_)), \text{Car}(_)\}, A, \text{idSub1}, w_c), \\ \quad (v_d, B) \\ \text{SubBoxes:} \quad \begin{array}{l} \text{ID/Agnt:} \quad \text{idSub1} / v_g \\ \text{ExVars:} \quad w_c \\ \text{ActType:} \quad \text{get}@obj(w_c) \\ \text{Conds:} \quad \text{Car}(w_c) \end{array}, \\ \quad \begin{array}{l} \text{ID/Agnt:} \quad \text{idSub2} / v_d \\ \text{ExVars:} \\ \text{ActType:} \quad \text{drive}@obj(v_c)@to(v_{dn}) \\ \text{Conds:} \\ \text{Car}(v_c) \end{array} \\ \text{Conds:} \quad \text{Car}(v_c) \end{array}$$

789 $\|D^*\| = \text{IntTh}^*(g, \text{idGR}, \text{GR}, \langle \lambda x_{dn}. \langle \lambda x_g. \langle \lambda x_c. \langle \lambda x_d. \Phi'_1 \wedge \Phi_2 \wedge \text{Car}(x_c) \rangle B \rangle \Upsilon \rangle A \rangle \text{Boston})$

790 where: $\Phi'_1 \equiv \text{Achieved}(x_g, (\exists y_c) \text{Done}(\text{get}@agt(x_g)@obj(y_c)) \wedge \text{Car}(y_c))$

$\Phi_2 \equiv \text{Achieved}(x_d, \text{Done}(\text{drive}@agt(x_d)@obj(x_c)@to(x_{dn})))$

$\Upsilon \equiv (1s. \text{Done}(\text{get}@agt(x_g)@obj(s)) \wedge \text{Car}(s))$

791 *Bob’s new intention concerning the “drive” action.* As it is currently configured, the box
792 D_2 , defined as the subsidiary box in D^* corresponding to the “drive” action, cannot serve
793 as the stand-alone DIS representing Bob’s intention to drive a car. The problem is that D_2
794 contains free variables—in particular, v_d , v_c and v_{dn} . Although the “in context” translation
795 of D_2 as a subsidiary box within D^* leads to the correct *Achieved*(...) clause within the
796 GAR intention represented by D^* , the generation of an intention clause requires using the
797 “stand-alone” translation. And the “stand-alone” translation function is not (and should not
798 be) defined for a DIS containing free variables. Thus, a new DIS must be generated that is
799 similar to D_2 , but in which the free variables have been converted to something else. After all,
800 Bob’s intention is to drive not any old car, but whatever car Alice has chosen for the group;
801 and Bob’s intention is to drive that car to the destination chosen by the group. Furthermore,
802 since the proper specification of the value of the car refers to the group’s choice for the agent
803 doing the “get” action, which is represented by the parameter, v_g , in D^* , that parameter too
804 must be properly defined in the DIS for Bob’s intention.

805 The robust solution is to start with a copy of the subsidiary box D_2 and convert the free
806 variables, v_{dn} , v_g and v_c , into *DefVar* parameters whose values are drawn from the corre-
807 sponding *DefVar* entries in D^* . (Since Bob is one of the agents holding the GAR intention
808 represented by D^* , it is proper to assume that the information contained in D^* is available to
809 Bob.) However, the free variable, v_d , which represents the agent of the “drive” action, can
810 simply be hardwired as B, since any change in the value of that parameter would lead Bob
811 to drop his intention anyway.

812 First, we introduce the following new syntax for a *DefVar* entry:

813 (5c) $(v, \text{defVarValue}(id, v))$

814 The intended interpretation of such an entry, which will be formalized in the extended trans-
815 lation function, is that the value of the parameter v shall be whatever value v has in the DIS
816 identified by id .

817 Next, let D_2^B be a copy of the subsidiary DIS D_2 in which the *Agt* field has been altered
818 to contain the constant, B (i.e., Bob). Here are the DIS-update operations that “capture” the
819 free variables, v_{dn} , v_g and v_c .

820 $[\tau_{6.1}] \text{AddDefVar}(D_2^B, v_{dn}, \text{defVarValue}(idGR, v_{dn}))$

821 —Add a *DefVar* entry for v_{dn} , thus capturing any of its formerly free occurrences in
822 D_2^B . Specify its value to be whatever value is being used in the parent DIS for the
823 variable of the same name.

824 $[\tau_{6.2}] \text{AddDefVar}(D_2^B, v_g, \text{defVarValue}(idGR, v_g))$ —Ditto for v_g .

825 $[\tau_{6.3}] \text{AddDefVar}(D_2^B, v_c, \text{defVarValue}(idGR, v_c))$ —Ditto for v_c .

826 Each of these operations lead to a *DefVar* entry in D_2^B containing the new *defVarValue* syntax.
827 The translation of this new kind of syntax will be addressed momentarily.

828 Here is the updated version of D_2^B together with its translation.

$$\begin{array}{l}
 829 \quad D_2^B = \begin{array}{l}
 \text{ID/Agt:} \quad idSub2 / B \\
 \text{DefVars:} \quad (v_{dn}, \text{defVarValue}(idGR, v_{dn})) \\
 \quad \quad \quad (v_g, \text{defVarValue}(idGR, v_g)) \\
 \quad \quad \quad (v_c, \text{defVarValue}(idGR, v_c)) \\
 \text{ActType:} \quad \text{drive@obj}(v_c)@to(v_{dn})
 \end{array} \\
 830 \quad \|D_2^B\| \equiv \text{IntTh}(B, idSub2, \langle \lambda y_{dn}. \langle \lambda y_g. \langle \lambda y_c. \Phi_2'' \Upsilon' \rangle A \rangle \text{Boston}) \\
 831 \quad \text{where:} \quad \Phi_2'' \equiv \text{Done}(\text{drive@agt}(B)@obj(y_c)@to(y_{dn})) \\
 \quad \quad \quad \Upsilon' \equiv (1s.\text{Done}(\text{get@agt}(y_g)@obj(s)) \wedge \text{Car}(s))
 \end{array}$$

832 *The translation of defVarValue(...) expressions.* The translation of *defVarValue(...)* expres-
833 sions is analogous to the evaluation (and *expansion*) of *macros* in the Lisp programming
834 language [34]. For example consider the expression, $\text{defVarValue}(idGR, v_g)$, in D_2^B . The
835 first step in the translation of this expression is to replace it by the corresponding expression
836 from the *DefVar* entry for the parameter, v_g , in the DIS, D^* , identified by $idGR$. That *DefVar*
837 entry is (v_g, A) . Thus, the “macro expansion” of $\text{defVarValue}(idGR, v_g)$ is simply A . That
838 expression is then translated as usual by the “stand-alone” translation function, yielding A .

839 Next, consider the expression, $\text{defVarValue}(idGR, v_c)$, in D_2^B . Ignoring the optional
840 arguments, the *DefVar* entry for the variable named v_c in D^* is: $(v_c, \{\text{Done}(\text{get@agt}$
841 $(v_g)@obj(_)), \text{Car}(_)\})$. Thus, the “macro expansion” of the expression, $\text{defVarValue}(idGR,$
842 $v_c)$, is simply, $\{\text{Done}(\text{get@agt}(v_g)@obj(_)), \text{Car}(_)\}$. This set of conditions then gets
843 translated by the “stand-alone” translation function, as usual, yielding:

$$844 \quad (1s.\text{Done}(\text{get@agt}(y_g)@obj(s)) \wedge \text{Car}(s))$$

845 Notice that the logical variable into which v_g is translated is determined by this application
846 of the translation function to D_2^B , which is independent of how v_g gets translated in D^* .

847 Notice that should the group subsequently decide to change any of their decisions (e.g.,
848 who should do the “get” action; or where they should travel to), these changes would auto-
849 matically be reflected in Bob’s intention, without requiring any changes to his DIS. That is,
850 the translation of Bob’s DIS would automatically reflect the changes because the *defVarValue*
851 expressions would generate different terms.

852 **7 Intentions in situ**

853 In this article we have focused primarily on the partial and dynamic nature of individual inten-
 854 tions without concern about how any particular intention might interact with other intentions.
 855 We believe that an adequate solution to the problems of partiality and dynamics is somewhat
 856 orthogonal to the solution of other important problems, such as the proper axiomatization
 857 of intention or the revision of intentions. In a companion paper [27], we explore these other
 858 issues more deeply, in the context of DIS theory. In this section we present a brief over-
 859 view of the axiomatization of intention and the process of intention revision covered in the
 860 companion paper.¹⁷

861 *Remark on how DISs are used by an agent.* We imagine that agents, during the elaboration
 862 and negotiation of shared plans, manipulate their respective DISs using the given update
 863 operations to modify the content of their intentions. At some point (e.g., as indicated by
 864 the particular agent's architecture) agents may wish to consider the consequences of new,
 865 individual intentions in the context of their other intentions or beliefs. To consider the logical
 866 consequences of a DIS, an agent need only translate the DIS, according to the semantics we
 867 have given, into first-order logic.¹⁸ To revise an existing collection of intentions, a process of
 868 intention revision, as summarized below and presented in our companion paper [27], is used
 869 to compute the maximal subset(s) of existing intentions consistent with the new intention.

870 **7.1 Axiomatization**

871 As we have indicated, intentions in our work have the general form, $IntTh(G, \phi)$, where G is
 872 an agent, ϕ is a formula, and $IntTh$ is a modal operator. In this section we dispense with refer-
 873 ence to the intention identifier; this is always possible, as discussed in Sect. 2.2. The truth of a
 874 formula, ϕ —where ϕ can contain modal operators—is then, as usual, expressed relative to a
 875 model, M , and a possible world, w , taken from a set of possible worlds, W : $M, w, \models \phi$. The
 876 semantics for intention formulas is expressed by way of an intention *accessibility* relation,
 877 $\mathcal{I}_G \subseteq W \times W$, such that (we suppress reference to a model) $w \models IntTh(G, \phi)$ just in case
 878 $w' \models \phi$ in all worlds w' such that $\mathcal{I}_G(w, w')$.

879 We adopt the weakest normal modal logic, known as System K, subscribing only to the
 880 following axiom of consequential closure [3]:

$$881 \quad (IntTh(G, \phi) \wedge IntTh(G, \phi \Rightarrow \psi)) \Rightarrow IntTh(G, \psi)$$

882 That is, if an agent intends that ϕ , and also intends that $\phi \Rightarrow \psi$, then the agent must also
 883 intend that ψ . An example might be the following, “If I intend that the house be clean by
 884 2:00 p.m., and I also intend that if the house is clean by 2:00 p.m. then I will go to the store at
 885 2:00 p.m., then I also intend that I will go to the store at 2:00 p.m.” A number of other typical
 886 axioms of modal logic all appear too strong and hence are not adopted here. For example,
 887 we do not adopt the axiom, $\models IntTh(G, \phi) \Rightarrow \phi$, since a rational agent will not adopt an
 888 intention that ϕ if ϕ is already true.

889 Rather than remaining within a modal logic, we instead adopt a reified approach to poss-
 890 sible worlds; in this way both the semantics of DISs and inference remains within first-

¹⁷ Our treatment in this article is restricted to DISs which do not involve combinations of logical operators. The companion paper [27] relaxes that restriction by augmenting the syntax to allow implication and negation over DISs. Thus, if Φ and Ψ are DISs, then so are $\neg\Phi$ and $\Phi \rightarrow \Psi$. Such DISs are then translated into FOL formulas involving the \neg and \Rightarrow operators.

¹⁸ There are, of course, alternatives to this approach. One would involve developing a proof theory for DISs analogous to that developed for DRTs [37].

891 order logic. We adopt a reified modal logic [24,26] in which ϕ becomes a function and
 892 $IntTh$ is introduced as a new predicate. The process of reification translates a statement,
 893 $w \models IntTh(G, \phi)$, into a FOL formula, $IntTh(G, \Phi, w)$, where Φ is a function which can
 894 be interpreted as a set of possible worlds—intuitively, the set of possible worlds where ϕ
 895 holds—and $IntTh(G, \Phi, w) \equiv (\forall w') acc(G, w, w')$, where $acc(G, w, w')$ is now a formula
 896 which is true just in case w' is accessible to G from w . For simplicity, we also adopt a common
 897 names assumption for each possible world [3]. This has the consequence that agents share
 898 cross-world identification of objects in the universe of discourse. Our approach differs signifi-
 899 cantly from Kamp’s mental structures approach for representing modalities in DRT [19]: our
 900 approach is closer in spirit to standard methods adopted in the AI knowledge representation
 901 literature; furthermore, we address intention revision, whereas Kamp does not.

902 Finally, note that time is not expressed explicitly in the logic, other than through the
 903 assumption that an agent’s set of intentions/DISs correspond to those that are true *now*; that
 904 set is assumed to persist until otherwise modified through an update or revision operation. It
 905 would be straightforward to extend DIS theory to represent time explicitly: see, for example,
 906 approaches within a reified logic [26].

907 7.2 Intention revision

908 We adopt a syntactic form of intention revision modeled on similar proposals from the belief
 909 revision literature [11,25,26]. Most approaches to belief revision are founded on the idea
 910 of *minimal change*: to revise a set of beliefs, S , with some new proposition, p , where p is
 911 inconsistent with S , one should make the minimal change necessary to S to accommodate p .
 912 In syntactic belief revision (also called base revision) the syntactic form of an agent’s beliefs
 913 (or intentions in our case) is important and is preserved. Briefly, the idea is that if a belief
 914 base contains $\{p, p \Rightarrow q\}$ and p is removed, then q will also, as it loses its “support” from
 915 p . In effect, one starts with a new set of beliefs containing just $\neg p$ and then iteratively adds
 916 contents from S , checking consistency along the way. In contrast, a model-based or belief set
 917 revision approach compares models in terms of the minimal changes that need to be made
 918 to accommodate the new belief. In this case, we have the initial model, $\{p, q\}$, which can be
 919 minimally modified to $\{\neg p, q\}$. In both approaches, the initial set of beliefs can be ordered
 920 according to some preference. For example, there might be certain causal and inviolable rules
 921 that an agent would never wish to disregard; those would be given highest priority.

922 Intention revision takes place in two steps within our framework. Let S be the current set
 923 of an agent’s intentions, in DIS form. Suppose we wish to modify an existing DIS, $D \in S$,
 924 according to one of the update functions described in Sect. 5. Let D' correspond to the updated
 925 DIS. To revise S , we create a set of equivalence classes on S : $\{S_1, S_2, \dots, S_n\}$ such that S_1
 926 is meant to correspond to those elements of S that are most important and S_n those that are
 927 least important. We start with D' and augment it with the maximal subset of S_1 such that the
 928 result is consistent and where consistency is determined via the translation of DISs to FOL.
 929 We then repeat the process for each maximal subset of the next equivalence class and stop
 930 when no additional elements of S can be added without introducing an inconsistency. The
 931 resulting set corresponds to the output of the revision process, though it may not be unique.

932 The ordering relation on S which induces the equivalence classes is a preorder chosen in
 933 roughly the following way. The set S_1 consists of all implications or rules involving intentions
 934 and the next levels reflect the level of decomposition in any intention. For example, if I intend
 935 to travel to Boston by driving a car then my top level intention of traveling to Boston takes
 936 precedence over my intention to do so by driving. Only one modification of the standard DIS
 937 form is needed to provide a sufficiently fine-grain ordering. In particular, we rewrite DISs in

938 terms of binary relations so that each element of a DIS (e.g., the list of agents) is stored as
 939 a binary relation (e.g., $\text{agents}(\text{DIS25}, \text{Zoe})$). The resulting ordering satisfies the conditions
 940 described by Nebel [25] and implies that the AGM properties [9] of the resulting revision
 941 operation are respected.

942 Our motivation for adopting a syntactic form of intention revision is that we believe it
 943 provides an attractive approach to solving the well known intention “side-effects” problem.
 944 A typical example is given by Cohen and Levesque [4]: a person intends to go to the dentist
 945 and also knows that the visit will be painful. However, the person does not also intend the
 946 side-effect of receiving pain: if he did then if he should decide not to go to the dentist, he
 947 would be compelled to explore other ways to receive pain (since intentions persist)! In con-
 948 trast, in base revision the initial contents of the agent’s intentions and beliefs would include
 949 a rule of the form, “if I intend to α then I believe, *ceteris paribus*, that I will obtain the
 950 consequences of performing α (i.e., pain).” If the intention is subsequently retracted, then so
 951 too must the belief indicated in the consequent.¹⁹

952 8 Conclusions

953 A long line of research on the representation of intentions [4,7,22,30,32,33,36], starting
 954 with the seminal work of Cohen and Levesque [4], has centered on the important property
 955 of persistence of intention and also on the role of intentions in the deliberations of a rational
 956 agent.

957 More recently, de Boer et al. [6] presented an approach to modeling interactions in multi-
 958 agent systems based on process algebra and constraint programming. Their work focuses on
 959 synchronized communication and action execution in two distinct phases. The first phase is
 960 a negotiation phase in which agents independently propose sets of constraints to impose on
 961 the parameters of a single joint action. If the constraints imposed by all of the agents are
 962 consistent, the agents move to an execution phase. In the execution phase, each agent inde-
 963 pendently and simultaneously chooses a set of values for some or all of the action parameters.
 964 If the choices of all the agents in this phase are unifiable and satisfy the above-mentioned
 965 constraints, then the joint action succeeds; otherwise it fails. Certain elements of their work
 966 bear some similarity to the model presented in this article. For example, if an agent leaves
 967 a parameter free during the execution phase, then the value for that parameter might be
 968 determined by another agent. In addition, the constraints proposed by agents are similar to
 969 the propositions in the *Conds* field of a DIS. However, there are many more differences.
 970 For example, our work accommodates the interleaving of planning and execution, explicitly
 971 models the delegation of authority and responsibility for binding parameters, explicitly rep-
 972 resents intentions and intention-creation and intention-update operations, and accommodates
 973 hierarchical action decomposition involving tasks done by different agents.

974 Our work takes as its starting point the observation that people elaborate and revise their
 975 intentions in an *incremental* fashion: intentions will often be only partially specified, requiring
 976 the use of existential quantifiers within the scope of an intention; however, during intention
 977 elaboration, subsequent references to the same intention will require access to the elements
 978 within the scope of such quantifiers. This article examined various examples that illustrated
 979 these properties, beginning with such simple statements as, “Alice intends to drive the car that
 980 Bob picks”, in which the referenced object has not yet been identified. We also observed that

¹⁹ As we discuss in the companion paper [27], a proper treatment of this example requires a representation, along the lines of DISs, corresponding to belief.

981 such statements share some features of well known examples from the linguistics literature
 982 and the various dynamic logics, notably Kamp's DRT, developed to handle them [12, 18, 20].
 983 We developed DISs to provide a similar flexibility of representation and elaboration involving
 984 agent intentions.

985 A further focus of our work originates from the—reasonable we think—observation that
 986 agents working in a team will often delegate the choice of particular object or property of an
 987 object referred to by an intention to other agents; any adequate theory of intention must pro-
 988 vide mechanisms to support the consistent reference to such objects across groups during the
 989 multi-agent elaboration process. DISs provide a framework that enforces such dependencies
 990 during elaborations and execution.

991 In a companion paper [27], we show how these properties of DISs lend themselves nicely
 992 to a treatment of intention revision and intention side-effects that takes as its starting point the
 993 observation that the elements that constitute the content of an intention are “not all created
 994 equal.” In that paper we also extend DISs to support the representation of arbitrary logical
 995 combinations (within and outside the scope of the intention operator) of intentions and their
 996 logical consequences.

997 We believe a secondary contribution of this work is that it brings together, for the first time,
 998 several independent threads of research. The concept of intentional context plays a prominent
 999 role in certain theories of collaboration, notably the theory of SharedPlans [13–15, 17], as well
 1000 as in its application to discourse understanding and collaborative interface design [1, 29, 31].
 1001 In contrast, the notion of attentional state introduced in the work of Grosz and others [16, 23]
 1002 to reflect the salience of objects in a natural language discourse has never played a first-class
 1003 role in the theory of SharedPlans. In addition, the contributions of Kamp [20] and Heim [18]
 1004 in linguistics have both been developed independently of the above contributions from the
 1005 fields of multiagent systems and computational linguistics. We believe that the DIS theory
 1006 that we have presented represents a first attempt at bringing together, in a productive fashion,
 1007 these related but independently developed and motivated theories.

1008 References

- 1009 1. Babaian, T., Grosz, B. J., & Shieber, S. (2002). A writer's collaborative assistant. In *Proceedings of the*
 1010 *international conference on intelligent user interfaces, IUI'02*. ACM Press.
- 1011 2. Bratman, M. E. (1999). *Faces of intention: Selected essays on intention and agency*. Cambridge:
 1012 Cambridge University Press.
- 1013 3. Chellas, B. F. (1980). *Modal logic: An introduction*. Cambridge: Cambridge University Press.
- 1014 4. Cohen, P. R., & Levesque, H. J. (1990). Intention is choice with commitment. *Artificial Intelligence*, 42,
 1015 213–261.
- 1016 5. Cohen, P. R., & Levesque, H. J. (1991). Teamwork. *Nous*, 25, 487–512.
- 1017 6. de Boer, F. S., de Vries, W., Meyer, J.-J. Ch., van Eijk, R. M., & van der Hoek, W. (2005). Process algebra
 1018 and constraint programming for modeling interactions in mas. *Applicable Algebra in Engineering,*
 1019 *Communication and Computing*, 16(2–3), 113–150.
- 1020 7. Dignum, F., Meyer, J.-J. Ch., Wieringa, R. J., & Kuiper, R. (1996). A modal approach to intentions,
 1021 commitments and obligations: Intention plus commitment yields obligation. In M. A. Brown & J. Carmo
 1022 (Eds.), *Deontic logic, agency and normative systems* (pp. 80–97). Springer-Verlag.
- 1023 8. Fitting, M., & Mendelsohn, R. L. (1998). *First-order modal logic*, Vol. 277 of *Synthese Library: Studies*
 1024 *in epistemology, logic, methodology, and philosophy of science*. Dordrecht: Kluwer.
- 1025 9. Gärdenfors, P. (Ed.) (1992). *Belief revision*. Cambridge: Cambridge University Press.
- 1026 10. Gilbert, M. (2000). *Sociality and responsibility*. New York: Rowman & Littlefield Publishers, Inc.
- 1027 11. Ginsberg, M. L. (1986). Counterfactuals. *Artificial Intelligence*, 30, 35–79.
- 1028 12. Groenendijk, J., & Stokhof, M. (1991). Dynamic predicate logic. *Linguistics and Philosophy*, 14(1),
 1029 39–100.

- 1030 13. Grosz, B. J., & Hunsberger, L. (2006). The dynamics of intention in collaborative activity. *Cognitive*
 1031 *Systems Research*, 7(2–3), 259–272.
- 1032 14. Grosz, B. J., & Kraus, S. (1996). Collaborative plans for complex group action. *Artificial Intelligence*,
 1033 86, 269–357.
- 1034 15. Grosz, B. J., & Kraus, S. (1999). The evolution of Shared Plans. In M. Wooldridge & A. Rao (Eds.),
 1035 *Foundations of rational agency*, Number 14 in Applied Logic Series, (pp. 227–262). Dordrecht: Kluwer.
- 1036 16. Grosz, B. J., & Sidner, C. L. (1986). Attention, intentions, and the structure of discourse. *Computational*
 1037 *Linguistics*, 12, 175–204.
- 1038 17. Grosz, B. J., & Sidner, C. L. (1990). Plans for discourse. In P. R. Cohen, J. Morgan, & M. E. Pollack
 1039 (Eds.), *Intentions in Communication*. Cambridge, MA: The MIT Press.
- 1040 18. Heim, I. (1983). File change semantics and the familiarity theory of definiteness. In R. Baurle,
 1041 C. Schwarze, & A. von Stechow (Eds.), *Meaning, use and the interpretation of language* (pp. 164–189).
 1042 Berlin: de Gruyter.
- 1043 19. Kamp, H. (1990). In *Prolegomena to a structural account of belief and other attitudes* (Ch. 2, pp. 27–90).
 1044 Number 20 in CSLI Lecture Notes. Chicago: University of Chicago Press.
- 1045 20. Kamp, H., & Reyle, U. (1993). *From discourse to logic*, Vol. 42 of *Studies in Linguistics and Philosophy*.
 1046 Dordrecht, The Netherlands: Kluwer Academic Publishers.
- 1047 21. Kinny, D., Ljungberg, M., Rao, A. S., Sonenberg, E., Tidhar, G., & Werner, E. (1994). Planned team
 1048 activity. In C. Castelfranchi & E. Werner (Eds.), *Artificial social systems*, Vol. 830 of *Lecture Notes in*
 1049 *Artificial Intelligence*. Amsterdam: Springer Verlag.
- 1050 22. Konolige, K., & Pollack, M. E. (1993). A representationalist theory of intentions. In *Proceedings of*
 1051 *International Joint Conference on Artificial Intelligence (IJCAI-93)* (pp. 390–395). San Mateo: Morgan
 1052 Kaufmann.
- 1053 23. Lochbaum, K. E. (1998). A collaborative planning model of discourse structure. *Computational Linguis-*
 1054 *tics*, 24(4), 525–572.
- 1055 24. Moore, R. C. (1985). A formal theory of knowledge and action. In *Formal Theories of the Commonsense*
 1056 *World*. Norwood: Ablex Publishing Corporation.
- 1057 25. Nebel, B. (1992). In P. Gärdenfors (Ed.), *Syntax based approaches to belief revision* (pp. 52–88).
- 1058 26. Ortiz, C. L. (1999). Explanatory update theory: Applications of counterfactual reasoning to causation.
 1059 *Artificial Intelligence*, 108, 125–178.
- 1060 27. Ortiz, C. L., & Hunsberger, L. Dynamic intention structures II: A theory of intention revision. In prepara-
 1061 tion.
- 1062 28. Ortiz, C. L., Jr. (1999). Introspective and elaborative processes in rational agents. *Journal of the Annals*
 1063 *of Mathematics and Artificial Intelligence*.
- 1064 29. Ortiz, C. L., Jr., & Grosz, B. J. (2000). Interpreting information requests in context: A collaborative web
 1065 interface for distance learning. *Autonomous Agents and Multi-agent Systems*.
- 1066 30. Rao, A. S., & Georgeff, M. P. (1991). Modeling rational agents within a BDI-architecture. In J. F. Allen,
 1067 R. Fikes, & E. Sandewall, (Eds.), *Proceedings of the 2nd International Conference on Knowledge Rep-*
 1068 *resentation and Reasoning (KR-91)* (pp. 473–484). Los Altos: Morgan Kaufmann Publishers.
- 1069 31. Rich, C., Sidner, C. L., & Lesh, N. (2001). COLLAGEN: Applying collaborative discourse theory to
 1070 human-computer interaction. *AI Magazine*, 22(4), 15–26.
- 1071 32. Sadek, M. D. (1992). A study in the logic of intentions. In *Proceedings of the 3rd Conference on Principles*
 1072 *of Knowledge Representation and Reasoning (KR'92)*. Los Altos: Morgan Kaufman Publishers, Inc.
- 1073 33. Singh, M. P. (1994). *Multiagent systems: A theoretical framework for intentions, know-how, and commu-*
 1074 *nications*. Berlin: Springer-Verlag.
- 1075 34. Steele, G. L. (1990). *Common Lisp the Language*. Belford: Digital Press.
- 1076 35. Tuomela, R. (1995). *The importance of Us: A philosophical study of basic social notions*. Stanford:
 1077 Stanford University Press.
- 1078 36. van der Hoek, W., Jamroga, W., & Wooldridge, M. (2007). Towards a theory of intention revision. *Syn-*
 1079 *these*, 155(2), 265–290.
- 1080 37. van Eijck, J. (2005). Discourse representation theory. In K. Brown (Ed.), *Encyclopedia of Language and*
 1081 *Linguistics*. Amsterdam: Elsevier. (2 edn.).