# Multi-context argumentative agents

Simon Parsons, Carles Sierra[*] and Nick R. Jennings
Department of Electronic Engineering,
Queen Mary and Westfield College,
University of London,
London E1 4NS, United Kingdom.

{S.D.Parsons,C.A.Sierra,N.R.Jennings}@qmw.ac.uk.

December 14, 1997

**Abstract**

We propose a new approach to designing agents based upon multi-context systems and argumentation. This approach allows the development of agent architectures which have a formal model in logic and a direct link between that model and its implementation. To exemplify our approach, we describe a case study of this relationship for a particular class of agent model (namely the strong realist Belief-Desire-Intention model).

## 1 Introduction

There are many ways of designing and building agent systems. However, the most common means is probably through an agent architecture. The role of such architectures is to define a separation of concerns—they identify the main functions which ultimately give rise to the agent's behaviour and they define the interdependencies between them. This approach to system design affords all the traditional advantages of modularisation in software engineering [28] and enables complex artifacts to be designed out of simpler components. However, one problem with much of the work on agent architectures is that it is somewhat ad hoc in nature. There is often little connection between the specification of the architecture and its implementation. This situation is clearly undesirable.

For this reason, our work seeks to provide a means of developing agent architectures which have a clear link between their specification and their implementation.

[*]On sabbatical leave from Artificial Intelligence Research Institute—IIIA, CSIC, Campus UAB, 08193 Bellaterra, Barcelona, Spain— thanks to a Spanish MEC grant PR95-313. Research partly supported by the Spanish CICYT project SMASH, TIC96-1038-C04001.

To do this, we make use of *multi-context systems* [12], a framework which allows distinct theoretical components to be defined and interrelated. We use different contexts to represent different components of an agent architecture, and specify the interactions between the components by means of the *bridge rules* between contexts. This makes it possible to move directly from the specification of the architecture to a formal description in terms of multi-context systems. Then, since each context contains a set of statements in a logic along with the axioms of that logic, it is possible to move directly to an implementation in which the various contexts are concurrent theorem provers which exchange information. In such an implementation each theorem prover component corresponds directly to one of the components of the original architecture. This approach enforces a modular structure with well-defined interfaces, and thus accords well with good software engineering practice.

This paper puts forward the idea of using multi-context systems as the link between architecture and implementation by developing a multi-context description of a Belief-Desire-Intention (BDI) agent. It also shows how this might work in practice with reference to a running example of two agents negotiating. Finally, the paper also discusses a system of argumentation which allows the agents to deal with conflicting information and makes it possible for two or more agents to engage in dialogues to resolve conflicts between them.

The remainder of the paper is structured in the following manner. Section 2 shows how multi-context systems can be used to specify agent architectures in general and BDI architectures [23] in particular. Section 3 presents the argumentation model and shows how it can be used to handle interactions between agents. Section 4 places our work in the context of previous work in the fields of multi-agent systems, argumentation and multi-context systems. Finally, section 5 outlines a number of issues which require further investigation.

## 2 Multi-context agents

An agent can be viewed as a multi-context system [12] in which each of the architecture's blocks is represented as a separate *unit*, an encapsulated set of axioms and an associated deductive mechanism, whose interrelationships are precisely defined via *bridge rules*, inference rules connecting units. To this end, section 2.1 indicates the general method of using multi-context systems to specify agent architectures. Then section 2.2 makes the discussion more concrete by indicating how a particular class of agent architecture—namely a BDI agent—can be modelled with this approach. Section 2.2 also provides an example of the BDI agent in action.

### 2.1 Generic multi-context agents

Using the multi-context approach, an agent architecture consists of the following four components (see [19] for a formal definition):

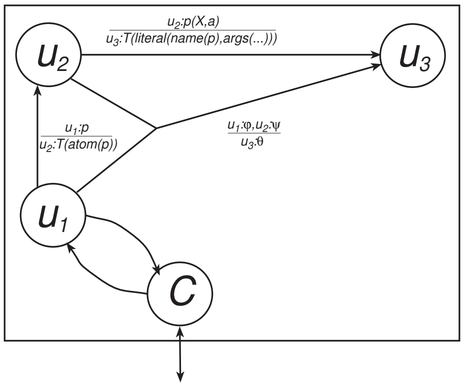


Figure 1: An example multi-context agent.

- *Units*: Structural entities representing the main components of the architecture.
- *Logics*: Declarative languages, each with a set of axioms and a number of rules of inference. Each unit has a single logic associated with it.
- *Theories*: Sets of formulae written in the logic associated with a unit.
- *Bridge rules*: Rules of inference which relate formulae in different units.

Figure 1 shows an example architecture in which the units are $u_1$, $u_2$, $u_3$ and $c$. $u_1$ contains a propositional logic, $u_2$, $u_3$ and $c$ contain a first order logic. No specific theories are given. The bridge rules are shown as arcs connecting the units.

Using the notation of [13], an agent is defined as a group of interconnected units represented by a pair $\langle \{u_i\}_{i \in I}, \Delta \rangle$ where $I$ is the set of unit indices, $u_i$ is the unit name given to the triplet $\langle L_i, A_i, \Delta_i \rangle$, where $L_i$, $A_i$ and $\Delta_i$ respectively are the language, axioms and rules of inference defining the logic. The axioms, the rules of inference, and the initial formulae in the theory generate the theory of the unit. $\Delta$ is the set of all bridge rules between the units. Bridge rules can be understood as rules of inference with premises and conclusions in (possibly different) units. For instance:

$$\frac{u_1 : \varphi, u_2 : \psi}{u_3 : \theta}$$

means that formula $\theta$ may be deduced in unit $u_3$ if formulae $\varphi$ and $\psi$ are deduced in units $u_1$ and $u_2$ respectively (see Figure 1). We will also write such rules as:

$$u_1 : \varphi, u_2 : \psi \Rightarrow u_3 : \theta$$

where more convenient.

In our approach, bridge rules are used to enforce relations between the various components of the agent architecture. For example in a BDI agent, a bridge rule between the intention unit and the belief unit might be:

$$I : I(\alpha) \Rightarrow B : B(\lceil I(\alpha) \rceil)$$

meaning that if the agent has an intention $\alpha$ then it is aware of this fact. Note that we take $B$, $D$ and $I$ to be predicates rather than modal operators. Therefore when one predicate comes into the scope of another, for instance because of the action of a bridge rule, it needs to be quoted using $\lceil \cdot \rceil$. However, because it is clear where quotation is necessary, we will omit its use for the remainder of the paper.

In general, the nature of the units will vary between architectures. For example, a BDI agent may have units which represent theories of belief, desire and intention, whereas an architecture based on a functional separation of concerns may have units for cooperation, situation assessment and plan execution [15, 16]. However for the purposes of this work, we assume that all agents have a dedicated *communication unit* ($C$ in Figure 1) which is responsible for enacting the agent's communication needs. We assume the existence of this unit because: (i) we want to encapsulate the agent's internal structure by having a unique and well defined interface with the environment; and (ii) we wish to have a cognitive interpretation of the architecture—the communication unit acts metaphorically as the agent's sensors and actuators (eyes, mouth and ears) by means of which the agent's 'mind' is effectively situated in the environment.

Since the communication unit deals with both incoming and outgoing messages, we could split it into two units; one for incoming messages, and one for outgoing messages. However we do not feel that this is necessary at the moment (though we do not rule out the possibility in the future). The reason for this is that we would like to keep the model relatively simple and so only introduce new units when either (i) they are necessary to make different cognitive components (which is why we have different units for desires and intentions) or (ii) they are necessary to capture different logics (which is why we have different units for beliefs and desires). At the moment we don't feel that either of these conditions apply to the different parts of the communication unit.

The formulae the agent can utter are determined by the language $L_C$ used by the communication units. In turn, $L_C$ is the result of the nested embeddings that the different bridge rules make between the languages of the various units. In this sense, the bridge rules play a key role in the design of an architecture. As we will show in section 2.2, important differences in behaviour can be attained simply by changing the pattern of "combination" of the units. Moreover, interaction between agents is carried out exclusively by the interchange of illocutions. Listening to an illocution is a form of sensing and speaking is a form of action. Hence the communication unit is responsible for making effective the actions—illocutions—selected to execute in the negotiation with the other agents.

The set of formulae that a given unit may contain depends on the unit's initial theory, axioms, inference rules and the incoming bridge rules. The formulae introduced by a bridge rule depend on the formulae present at the unit(s) of the premise(s) of the bridge rule. These may, in turn, depend on the bridge rules leading to that unit, and so on. The communication unit will receive formulae from other agents that will extend its theory [27]. In order to allow for the definition of flexible multi-agent communication—i.e. not tied to a fixed and predefined content language—the language $L_c$ must be defined only partially. As one of the units is partially defined, the propagation of formulae by bridge rules means the languages of all the units must also be partial. The evolution of the reasoning process by the application of bridge rules and the communication between agents extends these languages incrementally. For example, we can fix the set of predicates to be used in a certain language $L_{FOL}$ but leave the definition of $L_{FOL}$ parametric with respect to the terms the predicates may be applied over. By doing this, we underspecify the signature of $L_{FOL}$. For instance, we can declare a metapredicate ($T$) and then by means of bridge rules define which terms the predicate will apply over. The following:

$$\frac{u_1 : p}{u_2 : T(atom(p))}$$

is a bridge rule which embeds atoms of the theory of unit $u_1$ into the propositional metatheory of unit $u_2$, and:

$$\frac{u_2 : p(X, a)}{u_3 : T(literal(name(p), args(variable(X), constant(a))))}$$

does a similar job in the case of a first order language defined as a metalanguage for $u_2$ in $u_3$ (in a similar way to that in which it is done in OMEGA [1]). The partial nature of the language is essential if the agents are to negotiate and argue for these processes often involve the introduction of new concepts [27]. By definition, therefore, the agent's languages must be extensible.

An agent's deductive mechanism, $\vdash_i$, can be thought of as the relation between the utterances heard by the agent, the current theories of the agent's units and the utterances generated by the agent. This mechanism is realised by the use of an execution model based on the following assumptions:

1. *Concurrency.* The execution of each unit is a non-terminating deductive process (which may be formulated using dynamic logic [19]). All units execute concurrently. Moreover, the bridge rules are also concurrent processes. They examine the theories of the units in their premises for sets of formulae that match them, whenever a new match is found the concluding formula is asynchronously added to the theory of its associated unit.

2. *Reactivity.* The communication unit immediately processes (and thus adds to its theory) all messages it receives from other agents. This enables the agent to respond in an appropriate manner to important events which occur in the environment in which it is situated [2, 11].

## 2.2 Multi-context BDI Agents

To provide a specific exemplar of the method of approach advocated in the previous sub-section, we examine how a particular class of agent architecture—BDI agents—can be modelled. This seems an appropriate choice because BDI agents are currently of wide interest within the multi-agent system community. The particular theory of BDI on which the architecture is based is that of Rao and Georgeff. This model has evolved over time (as can be seen by comparing [24] and [25]) and in this section we account for the most recent approach [25] where three modalities are distinguished: $B$ for beliefs—used to represent the state of the environment, $D$ for desires—used to represent the motivations of the agent, and $I$ for intentions—used to represent the ends (or goals) of the agent. In order to fit this kind of model into our multi-context framework, we associate a separate unit for each of the modalities[1] (see Figure 2).

We could then give each of these units exactly the same interpretation as they are given in the Rao and Georgeff model—what we will refer to as the *direct interpretation*. This involves giving each modality a semantics in terms of possible worlds and the relation between modalities as relations between the associated possible worlds. This relation is often semantically modelled as inclusions between accessible worlds and syntactically modelled as axioms in the form of implications between the modalities. For instance, the fact that any intention-accessible world is also a belief-accessible world—the agent believes what it intends—is syntactically represented as $I(\alpha) \rightarrow B(\alpha)$. These implications have different deductive readings from each side of the connective (modus ponens or modus tollens) which is why some of the architectures we propose associate two bridge rules (in opposite directions) with each implication (see for instance Figure 2). In the direct interpretation the logics in the $B$, $D$ and $I$ units embody the temporal logic $CTL$ [8] (exactly as they do in Rao and Georgeff's model). In addition to the axioms of $CTL$ which are common to all the units, each unit has its own axioms encoding the behaviour of the modality. In the examples of figure 2, for instance, the axioms are the set S5 for $B$, and $K$ for $D$ and $I$. Each unit also contains the *generalisation* and *modus ponens* inference rules.

This completes the discussion of the logics within each unit, and so we turn to considering the bridge rules. As stated above, the set of bridge rules determine the relationship between the modalities and hence the behaviour of the agent. Three well established sets of relationships for BDI agents have been identified [25] (Figure 2):

- *Strong realism.* The set of intentions is a subset of the set of desires which in turn is a subset of the beliefs. That is, if an agent does not believe something, it will neither desire nor intend it [24].

- *Realism.* The set of beliefs is a subset of the set of desires which in turn is a

[1]In fact the general approach allows more than one unit for beliefs (as in [3]), desires or intentions if deemed appropriate. In the examples presented, however, this is not necessary.

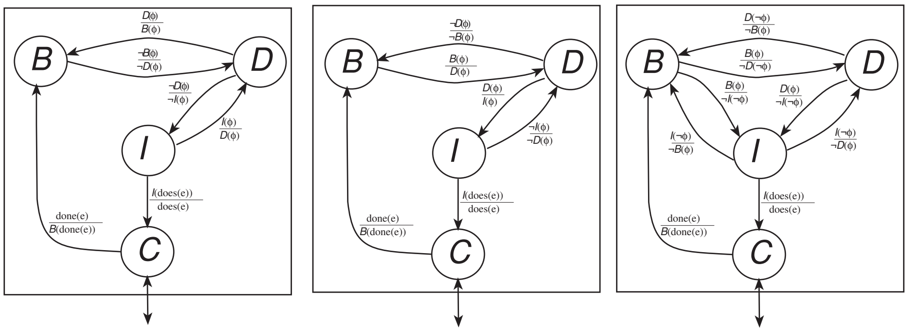


Figure 2: Different types of BDI agent. From left to right, the relations between modalities correspond to strong realism, realism and weak realism.

subset of the set of intentions. That is, if an agent believes something, it both desires and intends it [5].

- *Weak realism.* A case in between strong realism and realism. Agents do not desire properties the negation of which are believed, do not intend propositions the negations of which are desired, and do not intend propositions the negations of which are believed [22].

This completes the direct interpretation of the BDI model in terms of multi-context systems. However, this is not the interpretation we use in our work. Instead we prefer to build multi-context systems using an *indirect interpretation* in which the $B$, $D$ and $I$ are taken as predicates, as hinted at in Section 2.1. Such systems again have separate $B$, $D$ and $I$ units (along with a communication unit), and use the same sets of bridge rules as discussed above (exactly which set depends upon the kind of realism we want for our agents). To show exactly what we mean, we provide the following example which will be referred to throughout the paper. It should be noted that this example is intended to be illustrative rather than persuasive in that it shows how simple BDI agents may be set up using our multi-context approach, rather than showing that the agents can only be set up using our approach. Clearly it is possible to set up the agents without making them BDI agents. Furthermore, we acknowledge that the knowledge the agents use to reason about their world is rather simplistic. We could, of course, use more realistic theories of actions and planning but we feel that this would rather cloud the issue since:

(i) such a theory would make the example more complicated; and

(ii) the usefullness of both BDI models in general and our approach in particular hinges more on the fact that they make it possible to clearly specify agents in general than on the specifics of those agents' approaches to representing actions.

**Example** Consider the case of two home improvement agents which are strong realists in the sense introduced above. All languages are first order, the operators $B$, $D$, and $I$ are represented as predicates with a subindex representing the agent, and the formulae are prefixed by the name of the unit to which they belong.

Throughout the example there is some scope for confusion. The reason is that there are two languages in operation here. The first is the language of the $B$, $D$ and $I$ predicates in which the connectives are those of first order logic. The second is the language quoted within the scope of the $B$, $D$ and $I$ predicates in which all the connectives are just terms of the relevant predicates. In this language, conjunction is as usual, but $\rightarrow$ does not represent material implication, Instead it represents the relationship, admittedly a rather naive one, between the goals of the agent and the means the agent has to achieve them.

Assume agent $a$ has the intention of hanging a picture, and that it has various beliefs about resources and how they can be used to hang mirrors and pictures:

$$\begin{aligned}
I &: I_a(Can(a, hang_picture)) && (1)\\
B &: B_a(Have(a, picture)) && (2)\\
B &: B_a(Have(a, screw)) && (3)\\
B &: B_a(Have(a, hammer)) && (4)\\
B &: B_a(Have(a, screwdriver)) && (5)\\
B &: B_a(Have(b, nail)) && (6)\\
B &: B_a(Have(X, hammer) \wedge Have(X, nail) \wedge Have(X, picture) \rightarrow \\
&\qquad Can(X, hang_picture)) && (7)\\
B &: B_a(Have(X, screw) \wedge Have(X, screwdriver) \wedge Have(X, mirror) \rightarrow \\
&\qquad Can(X, hang_mirror)) && (8)
\end{aligned}$$

Assume agent $b$ wants to hang a mirror (and has this as an intention) and has various beliefs about its resources and the action of hanging mirrors:

$$\begin{aligned}
I &: I_b(Can(b, hang_mirror)) && (9)\\
B &: B_b(Have(b, mirror)) && (10)\\
B &: B_b(Have(b, nail)) && (11)\\
B &: B_b(Have(X, hammer) \wedge Have(X, nail) \wedge Have(X, mirror) \rightarrow \\
&\qquad Can(X, hang_mirror)) && (12)
\end{aligned}$$

Both agents need a simple theory of action that integrates a model of the available resources with their planning mechanism. This theory needs to model the following ideas (with $i \in \{a, b\}$):

**Ownership.** When an agent (X) is the owner of an artifact (Z) and it gives Z to another agent (Y), Y becomes its new owner:

$$B : B_i(Have(X, Z) \wedge Give(X, Y, Z) \rightarrow Have(Y, Z)) \qquad (13)$$

**Unicity.** When an agent (X) gives an artifact (Z) away, it no longer owns it [2]:

$$B : B_i(Have(X, Z) \wedge Give(X, Y, Z) \rightarrow \neg Have(X, Z)) \tag{14}$$

**Benevolence.** When an agent $i$ has something (Z) that it does not intend to use and is asked to give it to another agent (X), $i$ adopts the intention of giving Z to X. Naturally more complex cooperative strategies can be defined if desired:

$$\begin{aligned} B : B_i(Have(i, Z) \wedge \neg I_i(Have(i, Z)) \wedge Ask(X, i, Give(i, X, Z)) \rightarrow \\ I_i(Give(i, X, Z))) \end{aligned} \tag{15}$$

The following axioms represent a similarly simplistic theory of planning (but again one which suffices for our example). In crude terms, when an agent believes that it has the intention of doing something and has a rule for achieving that intention then the pre-conditions of the rule become new intentions. Recall that the $\rightarrow$ between the $P_i$ and $Q$ is not material implication.

**Parsimony.** If an agent believes that it does not intend something, it does not believe that it will intend the means to achieve it.

$$B : B_i(\neg I_i(Q)) \wedge B_i(P_1 \wedge \ldots \wedge P_j \wedge \ldots \wedge P_n \rightarrow Q) \rightarrow \neg B_i(I_i(P_j)) \tag{16}$$

**Reduction.** If there is only one way of achieving an intention, an agent adopts the intention of achieving its preconditions.

$$\begin{aligned} B : B_i(I_i(Q)) \wedge B_i(P_1 \wedge \ldots \wedge P_j \wedge \ldots \wedge P_n \rightarrow Q) \\ \wedge \neg B_i(R_1 \wedge \ldots \wedge R_m \rightarrow Q) \rightarrow B_i(I_i(P_j)) \end{aligned} \tag{17}$$

where $R_1 \wedge \ldots \wedge R_m$ is not a permutation of $P_1 \wedge \ldots \wedge P_n$.

**Unique Choice.** If there are two or more ways of achieving an intention, only one is intended. Note that we use $\bigtriangledown$ to denote exclusive or.

$$\begin{aligned} B : B_i(I_i(Q)) \wedge B_i(P_1 \wedge \ldots \wedge P_j \wedge \ldots \wedge P_n \rightarrow Q) \wedge B_i(R_1 \wedge \ldots \wedge R_m \rightarrow Q) \rightarrow \\ B_i(I_i(P_1 \wedge \ldots \wedge P_n)) \bigtriangledown B_i(I_i(R_1 \wedge \ldots \wedge R_m)) \end{aligned} \tag{18}$$

where $R_1 \wedge \ldots \wedge R_m$ is not a permutation of $P_1 \wedge \ldots \wedge P_n$. As mentioned above, we acknowledge that both the theory of action and the theory of planning are rather naive. The interested reader is encouraged to substitute their own such theories if desired.

So far, we have specified the initial states of the two agents (1–12) and provided limited theories of action (13–15) and planning (16–18) to enable the agent to operate. Finally, we require some domain dependent bridge rules to link inter-agent communication and the agent's internal states:

[2] As it stands this formula appears contradictory. This is because we have, for simplicity, ignored the treatment of time. Of course, the complete specification of this example (which is not our main focus) would need time to be handled. We could do this by including time as an additional argument to each predicate, in which case the unicity formula would read $B : B_i(Have(X, Z, t) \wedge Give(X, Y, Z, t) \rightarrow \neg Have(X, Z, t + 1))$. Doing this would involve making the base logic for each unit "time capable", for instance by using the system introduced by Vila [30].

**Request.** When an agent (i) needs something (Z) from another agent (X), it asks for it:

$$I : I_i(Give(X, i, Z)) \Rightarrow C : Ask(i, X, Give(X, i, Z)) \tag{19}$$

**Offer.** When an agent (i) has the intention of offering something (Z) to another agent (X), it informs the recipient of this fact:

$$I : I_i(Give(i, X, Z)) \Rightarrow C : Tell(i, X, Give(i, X, Z)) \tag{20}$$

**Trust.** When an agent (i) is told of a belief of another agent (X), it accepts that belief:

$$C : Tell(X, i, B_X(\varphi)) \Rightarrow B : B_i(\varphi) \tag{21}$$

**Awareness of intentions.** Agents are aware of their intentions.

$$I : I_i(\alpha) \Rightarrow B : B_i(I_i(\alpha)) \tag{22}$$

$$I : \neg I_i(\alpha) \Rightarrow B : B_i(\neg I_i(\alpha)) \tag{23}$$

**Awareness of illocutions.** Agents are aware of the requirements received by the communication unit.

$$C : \alpha \Rightarrow B : B_i(\alpha) \tag{24}$$

**Impulsiveness.** When an agent believes it has an intention, it adopts that intention.

$$B : B_i(I_i(\alpha)) \Rightarrow I : I_i(\alpha) \tag{25}$$

□

We have now demonstrated how the multi-context approach can be used to specify BDI agents. In particular, we have defined two home improvement agents which we will return to after we have discussed the use we make of argumentation.

## 3 Agents and argumentation

The system of argumentation which we use is based upon that proposed by Fox and colleagues [9, 17]. As with many systems of argumentation, it works by constructing a series of logical steps (arguments) for and against propositions of interest and as such may be seen as an extension of classical logic. In classical logic, an argument is a sequence of inferences which prove that a conclusion is true. In the system of argumentation adopted here, an argument is a sequence of inferences which suggest that a conclusion may be true. The strength of such a suggestion is ascertained by examining the propositions used in the relevant arguments. This form of argumentation may be seen as a formalisation of work on informal logic and argumentation in philosophy [29], though it should be stressed that it was developed independently. It is summarised by the following schema:

$$\Gamma \vdash (\varphi, G)$$

where $\Gamma$ is the set of formulae available for building arguments, $\vdash$ is a suitable consequence relation, $\varphi$ is the proposition for which the argument is made and $G$ indicates the set of formulae used to infer $\varphi$, $G \subseteq \Gamma$. The pair $(\varphi, G)$ may also be extended to the triple $(\varphi, G, \alpha)$ to take account of the fact that $\varphi$ may not be known to be true by giving it a degree of belief $\alpha$ [17]. The kind of reasoning afforded by argumentation is similar to that provided by labelled deductive systems [10], but it differs in its use of the labels. Whilst most labelled deductive systems use their labels to control inference, this system of argumentation uses the labels to determine which of its conclusions are most valid.

The remainder of this section extends this system of argumentation to the multi-agent case and demonstrates how it can be used within the agent architecture introduced in section 2. Again this is described first in a general setting in section 3.1 and then for BDI agents in section 3.2.

### 3.1 Multi-context multi-agent argumentation

We fit argumentation into our multi-context agents by building arguments using the rules of inference of the various units and the bridge rules between units. However, there is an important difference between the system of argumentation we employ and that used by other authors [6, 7, 18, 21]. This is as follows. Often the grounds of an argument are just the formulae from which the argument is built; it is taken for granted that the agent in question can build the necessary proof from the grounds when desired. However, this assumption does not necessarily hold in multi-agent systems. In particular, different agents may have different rules of inference within their units and different bridge rules between them. This means that there is no guarantee that other agents are able to reconstruct the proof for a formula from the formulae on which it is based. Hence, the grounds must contain complete proofs, including the rules of inference and the bridge rules employed and we need to augment the notation for arguments to identify which rules of inference and which bridge rules are employed. We do this by exploiting the fact that rules of inference and bridge rules have a similar deductive behaviour and can be noted in an identical way. We also need to identify the agent making the argument. We use:

$$\Gamma \vdash_d \varphi$$

with $d = a_{\{r_1,\ldots,r_n\}}$, to mean that the formula $\varphi$ is deduced by agent $a$ from the set of formulae $\Gamma$ by using the set of inference rules or bridge rules $\{r_1, \ldots, r_n\}$. When there is no ambiguity the name of the agent will be omitted. The following are examples of the use of the notation to define deductive steps in agent $a$. In the first the agent uses the "request" bridge rule (19) to create a request from an intention, and in the second it applies an inference rule (mp stands for modus ponens) to two formulae in unit $I$:

$$\begin{array}{rcl} \{I : I_a(Give(b, a, nail))\} & \vdash_{a_{\{19\}}} & C : Ask(a, b, Give(b, a, nail)) \\ \{I : p, I : p \rightarrow q\} & \vdash_{a_{\{mp\}}} & I : q \end{array}$$

Making the rules of inference and bridge rules explicit means that they become part of the argument. This then makes it possible to build arguments about the applicability of such rules. As a result, agents which use different logics and which therefore use different rules of inference and bridge rules are in principle able to engage in argumentation about which rules are valid. However, to do this in practice is complex since we need to find ways of representing the reasoning mechanism of other agents within individual agents so that each agent has a model of the ways in which its aquaintances reason. While it is one of our main lines of continuing research, we will say little more about it in this paper.

At this point we should also say a few words about the relationship between our description of arguments and the meta-theory of our agents. When we describe an argument we are making a statement in the meta-theory of the agent concerned since we are talking about what the agent may prove. Thus we could talk about arguments in general purely in terms of statements in the meta-theories of agents. However, we choose not to since we don't think that it adds anything to the explanation, and possibly even makes things less clear.

In the remainder of the paper we drop the '$B$ :', '$D$ :' and '$I$ :', once again to simplify the notation. With this in mind, we can follow [7] in defining arguments and associated notions:

**Definition 1** *Given an agent $a$, an* argument *for a formula $\varphi$ is a pair $(\varphi, P)$ where $P = \{s_1, \ldots, s_n\}$ and either $s_i$ is a formula in the theories of agent $a$ or $s_i = \Gamma_i \vdash_{d_i} \psi$ and $p_j \in \Gamma_i$ is either a formula in the theories of $a$ or the conclusion of a previous step in $P$, and $\psi$ is a formula in the language of $a$, and $s_n = \Gamma_n \vdash_{d_n} \varphi$.*

For the sake of readability, we will often refer to the conclusion of a deductive step with the identifier given to the step. It is helpful to distinguish consistent arguments (since we allow inconsistent ones even though we don't make use of them):

**Definition 2** *We say that an argument $(\varphi, P)$ is consistent if there are no $s_i, s_j \in P$ such that $s_i = \Gamma_i \vdash_{d_i} \psi$ and $s_j = \Gamma_j \vdash_{d_j} \neg\psi$*

We can now identify two useful classes of arguments, trivial and tautological:

**Definition 3** *An argument $(\varphi, P)$ is* non-trivial *if it is consistent.*

**Definition 4** *An argument $(\varphi, P)$ is* tautological *if all deductive steps in $P$ are built using only rules of inference, bridge rules and axioms of the logics of the agent's units.*

Clearly the notion of a tautological argument will vary between agents when agents use different rules of inference and different bridge rules. Thus agents which use such different rules will differ in the way in which they classify arguments. The effects of this are, once again, out of the scope of this paper.

Now, because in argumentation a proof for a formula only suggests that the formula may be true (rather than indicating that it is true), we can have arguments for and against the same formula. In particular, given an argument for a formula, there are two interesting types of argument against it; arguments which rebut it and arguments which undercut it:

**Definition 5** *An argument* $(\varphi_i, P_i)$ rebuts *an argument* $(\varphi_j, P_j)$ *if* $\varphi_i$ *attacks* $\varphi_j$*.*

Note that the notion of "attack" is defined in Section 3.2; for the moment it is considered primitive, but can be thought of as meaning that the arguments disagree over the truth of the proposition $\varphi$.

**Definition 6** *An argument* $(\varphi_i, P_i)$ undercuts *an argument* $(\varphi_j, P_j)$ *if there exists* $s_k \in P_j$ *such that (1)* $s_k$ *is a formula and* $\varphi_i$ *attacks* $s_k$*, or (2)* $s_k = \Gamma_k \vdash_{d_k} \psi$ *and* $\varphi_k$ *attacks* $\psi$

Relationships between arguments such as rebutting and undercutting have been widely studied, for instance by [6, 18, 21, 31]. The notions that we use here are broadly in line with the consensus on the issue. However, there is another form of conflict between arguments which stems from the inclusion of rules of inference and bridge rules in the argument. This is, as mentioned above, that one argument might attack the use of a rule used to build another argument and, as also mentioned above, we intend to discuss this matter in later papers.

It should be noted that, unlike the authors mentioned above, we do not present a universal definition of what it means for one argument to attack another. We firmly believe that the form of attack depends upon the underlying language, and so, in our terms, will depend upon which units arguments are built in and what the units represent. We discuss notions of attack relevant to BDI agents in Section 3.2. Our motivation for classifying arguments in this way is that it allows us to split arguments into classes of acceptability, again following [7] and our previous work on argumentation in multi-agent systems [20]. We have, in order of increasing acceptability:

A1 The class of all arguments that may be made from $\Gamma$.

A2 The class of all non-trivial arguments that may be made from $\Gamma$.

A3 The class of all arguments that may be made from $\Gamma$ for propositions for which there are no rebutting arguments that may be made from $\Gamma$.

A4 The class of all arguments that may be made from $\Gamma$ for propositions for which there are no undercutting arguments that may be made from $\Gamma$.

A5 The class of all tautological arguments that may be made from $\Gamma$.

Informally, the idea is that arguments in higher numbered classes are more acceptable because they are less questionable. Thus, if we have an argument for a proposition $\varphi$ which is in class A4, and an argument for $\psi$ which is in A2, then the better argument is that for $\varphi$. Since any argument from any class is included in all classes of lower acceptability, there is an order over the acceptability classes defined by set inclusion:

$$A_5(\Gamma) \subseteq A_4(\Gamma) \subseteq A_3(\Gamma) \subseteq A_2(\Gamma) \subseteq A_1(\Gamma)$$

Thus arguments in smaller classes are more acceptable than arguments in larger classes. Acceptability is important because it gives agents a way of deciding how to revise what they know (see below). Clearly the acceptability class of an argument is local to an agent since it is depends upon the database from which the argument is built.

### 3.2 Argumentation in BDI Agents

To instantiate our argumentation model within the context of a particular agent architecture, like the one proposed in section 2.2, we need to say exactly when two formulae attack one another. This is a rather more complex issue than is the case in single agent argumentation when two propositions attack one another if one is the negation of the other. In our BDI agents, the complication comes largely from the "modalities" since there is no conflict between an agent which believes $\varphi$, that is $B_i(\varphi)$, and one which believes $\neg\varphi$, that is $B_j(\neg\varphi)$. Conflicts only occur when:

1. agents have opposite intentions (since then they actively intend to bring about incompatible results);

2. one agent intends to change a particular mental state in another agent; in other words intends to persuade another agent to believe (or desire or intend) the negation of one of its current beliefs (respectively desires or intentions).

That is:

1. $I_i(\varphi)$ attacks $I_j(\neg\varphi)$. For example, the fact "Carles intends to be Prime Minister", $I_{Carles}(Prime(Carles))$, attacks the fact "Simon intends that Carles is not Prime Minister", $I_{Simon}(\neg Prime(Carles))$.

2. $I_i(M_j(\varphi))$ attacks $M_j(\neg\varphi)$. For example, the fact "Kate intends that Simon believes that God exists", $I_{Kate}(B_{Simon}(God))$, attacks the fact "Simon believes that God does not exist", $B_{Simon}(\neg God)$.

In the first case Simon and Carles are in conflict about who should be Prime Minister. In the second case there is a conflict because Kate wants to change Simon's beliefs to a view that is the opposite of what he already believes. The second case can be generalised so that $I_i(M_{j_1}(M_{j_2}(\varphi)))$ attacks $M_{j_1}(\neg M_{j_2}(\varphi))$ and also attacks $M_{j_1}(M_{j_2}(\neg\varphi))$ where $j_k$ is an agent identifier. Thus we get the following definition:

**Definition 7** *Given agents $i$ and $j$, we say that a formula $\varphi_i$ of the language of agent $i$* attacks *a formula $\varphi_j$ of the language of agent $j$ if one of following cases hold:*

1. $\varphi_i = I_i(\varphi)$ *and* $\varphi_j = I_j(\neg\varphi)$
2. $\varphi_i = I_i(M_{j_1}(M_{j_2}(\ldots(M_{j_n}(\varphi))\ldots)))$ *and* $\varphi_j = M_{j_1}(\ldots(\neg M_{j_k}(\ldots(M_{j_n}(\varphi))\ldots)\,)\ldots)$ *with* $1 \leq k \leq n$ *or* $\varphi_j = M_{j_1}(M_{j_2}(\ldots(M_{j_n}(\neg\varphi))\ldots))$.

With this notion of attack, our use of rebut, undercut and the acceptability classes is a natural extension of the use proposed by Elvang-Gøranssen *et al.* [7] to the multi-agent case. The difference is as follows. The notion of attack proposed by Elvang-Gøranssen *et al.* would recognise the conflict between $I_a(\varphi)$ and $\neg I_a(\varphi)$ (which in our approach would be inconsistency), but would not identify the conflict between $I_a(\varphi)$ and $I_b(\neg\varphi)$. Our extension, by virtue of the fact that it looks inside the modalities, is able to detect this latter type of attack. This is important because it is the latter form of attack that figures most prominently in interactions between agents. Because it does not figure greatly in the interaction between agents, at the moment we have nothing much to say about the handling of inconsistency within our multi-context agents. However, it might well be the case that an agent will have contradictory beliefs $B_a(\varphi)$ and $\neg B_a(\varphi)$, and if it becomes necessary to handle such situations, it seems likely that we can make use of the argument-based approaches to dealing with inconsistency which already exist (see for example [6, 7, 18, 21]).

We should also point out that, even when handling contradictory arguments, the process of building arguments is monotonic. If we can build an argument for $\varphi$, then we can always build an argument for it, even if we are able to build an argument for $\neg\varphi$ later. However, the process of coming to conclusions using arguments is non-monotonic. If we have an argument for $\varphi$ and no argument for $\neg\varphi$ then we conclude $\varphi$. If later we can build an argument for $\neg\varphi$ which is more acceptable than the argument for $\varphi$ then we change our conclusion to $\neg\varphi$.

**Example** (continued) Using the home improvement agents specified earlier, we illustrate how argumentation is used by agents both to build up justified plans of actions and to resolve conflicts that arise between different agents' plans. For clarity we assume that agents can simply pass each other arguments, ignoring issues such as what language they are passed in and what the protocol for passing them is—such issues are discussed at length by the authors in [27].

**Step 1**: Agent $a$ tries to find a proof for $Can(a, hang\_picture)$ because of its intention $I_a(Can(a, hang\_picture))$. The only proof it can build is based on $B_a(Have(a, nail))$, which in turn, by the theory of planning, makes $B_a(I_a(Give(b, a, nail)))$ true. This is transformed, by means of bridge rule 25, into $I_a(Give(b, a, nail))$. More formally, agent $a$ builds an argument

$$(I_a(Give(b, a, nail)), P_a)$$

where $P_a$ is[3]:

$$\begin{array}{rll}
\{I_a(Can(a, hang_picture))\} & \vdash_{22} & B_a(I_a(Can(a, hang_picture))) \quad (26)\\
\{(26),(17),(7)\} & \vdash_{mp} & B_a(I_a(Have(a, nail))) \quad (27)\\
\{(6),(13)\} & \vdash_{mp} & B_a(Give(b, Y, nail) \to Have(Y, nail)) \quad (28)\\
\{(28),(27),(17)\} & \vdash_{mp} & B_a(I_a(Give(b, a, nail))) \quad (29)\\
\{(29)\} & \vdash_{25} & I_a(Give(b, a, nail)) \quad (30)
\end{array}$$

This is then converted into an action using bridge rule 19

$$\{(30)\} \vdash_{19} Ask(a, b, Give(b, a, nail))$$

When agent $a$ generates the argument $(I_a(Give(b, a, nail)), P_a)$ it is placed in acceptability class $A_4$ since $a$ cannot build any undercutting arguments against it and so $a$ deems it to be a suitable suggestion to be passed to $b$.

**Step 2**: Unit $C$ of agent $b$ receives the formula $Ask(a, b, Give(b, a, nail))$, which, as specified, brings with it the argument:

$$(I_a(Give(b, a, nail)), \{(26), (27), (28), (29), (30)\})$$

Now, agent $b$ has its own goal, $I_b(Can(b, hang\_mirror))$, which as we will see forms the basis of its argument:

$$(I_b(\neg Give(b, a, nail)), P_b)$$

where $P_b$:

$$\begin{array}{rll}
\{I_b(Can(b, hang_mirror))\} & \vdash_{22} & B_b(I_b(Can(b, hang_mirror))) \quad (31)\\
\{(31),(12),(17)\} & \vdash_{mp} & B_b(I_b(Have(b, nail))) \quad (32)\\
\{(32),(14)\} & \vdash_{mt} & B_b(I_b(\neg Give(b, Y, nail))) \quad (33)\\
\{(33)\} & \vdash_{pt} & B_b(I_b(\neg Give(b, a, nail))) \quad (34)
\end{array}$$

This argument rebuts the argument for $I_a(Give(b, a, nail))$. This means that for agent $b$ both arguments are in class $A_2$ (since they mutually rebut one another but they are consistent). Assuming the agents are rational, and given that both arguments are in the same class, $b$ will probably prefer (by some utility analysis) the second argument since this enables it to satisfy one of its intentions (adherence to the argument proposed by $a$ would clobber its intention of hanging the mirror) and so will return the second argument to $a$.

**Step 3**: When agent $a$ receives the argument from $b$ it classifies both its original argument and the incoming argument as class $A_2$ since they are both rebutted (by

[3]In what follows, 'mp' stands for *modus ponens*, 'mt' stands for *modus tollens* and 'pt' stands for *particularisation*, and we omit the axioms of the unit in which the deduction is made.

each other). Thus its original argument moves from $A_4$ to $A_2$. In response, agent $a$ generates a new argument which provides an alternative way of hanging the mirror that will satisfy $b$'s goal without using the nail:

$$(B_a(\neg I_b(Have(b, nail))), P'_a)$$

where $P'_a$ is[4]:

$$\begin{array}{rll}
\{\neg I_a(Can(a, hang_mirror))\} & \vdash_{23} & B_a(\neg I_a(Can(a, hang_mirror))) \qquad (35) \\
\{(35), (16), (8)\} & \vdash_{mp} & \neg B_a(I_a(Have(a, screw))) \\
 & & \wedge \neg B_a(I_a(Have(a, screwdriver))) \qquad (36) \\
\{(36)\} & \vdash_{sr} & \neg I_a(I_a(Have(a, screwdriver))) \\
 & & \wedge \neg I_a(I_a(Have(a, screw))) \qquad (37) \\
\{(36), (3), (5), (15)\} & \vdash_{mp,pt} & B_a(Ask(b, a, Give(a, b, screw)) \rightarrow \\
 & & \quad I_a(Give(a, b, screw))) \\
 & & \wedge B_a(Ask(b, a, Give(a, b, screwdriver)) \rightarrow \\
 & & \quad I_a(Give(a, b, screwdriver))) \qquad (38) \\
\{(38), (8)\} & \vdash_{mp} & B_a(Ask(b, a, Give(a, b, screw)) \\
 & & \wedge Ask(b, a, Give(a, b, screwdriver)) \\
 & & \rightarrow Can(b, hang_mirror)) \qquad (39) \\
\{(18), (39), (12)\} & \vdash_{mp} & B_a(\neg I_b(Have(b, nail))) \qquad (40)
\end{array}$$

This argument is classified in $A_4$ since $a$ can neither rebut nor undercut it. Agent $a$ then sends this latest argument to $b$ as a counterproposal. Agent $b$ cannot build any arguments which attack this new argument and so it is classified as being in $A_4$. Given the strength of the new argument, $b$ accepts it. Here the crucial point is that $b$ cannot construct a rebuttal for the new argument as a subargument of its previous argument because it can no longer use the reduction planning rule (17). This is because $b$ has now acquired a new rule for hanging mirrors (as part of the new argument). Moreover, the second argument can no longer be maintained for the same reason, so $a$'s original argument is reclassified as being in $A_4$. Hence agent $a$ will receive the nail, agent $b$ will ask for the screw and the screwdriver and both will reach their goals. □

Note that step 37 is crucial in the construction of the undercutting argument. This step depends upon the fact that agent $a$ has the bridge rules associated with strong realism and so can go from $\neg B_a(I_a(Have(a, screw)))$ to $\neg D_a(I_a(Have(a, screw)))$ and hence to $\neg I_a(I_a(Have(a, screw)))$. If the agent did not have these bridge rules (for instance if it had those of realism or weak realism) $a$ would not have been able to come up with its final suggestion. This gives some hint of the flexibility of our approach and shows that changing some basic assumptions about the relations between the units makes a substantial difference to the behaviour of the agents.

[4]'sr' stands for the set of bridge rules associated with *strong realism*.

## 4 Related work

This paper has dealt with a number of topics from various research areas—including argumentation-based reasoning, formal models of agent architectures, and multi-context systems. In this section we take a brief look at related work in these three areas. Traditionally work on argumentation-based reasoning has concentrated on the operation of a single agent which argues with itself in order to establish its beliefs [6, 7, 18, 21]. As indicated and discussed in section 3, this basic approach and framework needed to be extended to account for the multi-agent case in which several traditional assumptions do not hold. Previous work which has produced formal models of agent architectures (eg dMARS [14], Agent0 [26] and GRATE* [15]) has failed to carry forward the clarity of the specification into the implementation—there is a leap of faith required between the two. Our work, on the other hand, maintains a clear link between specification and implementation as demonstrated by the ease of implementing the deduction mechanism used in the example. There are also differences between our work and previous work on using multi-context systems to model agents' beliefs. In the latter [13], different units, all containing a belief predicate, are used to represent the beliefs of the agent and the beliefs of all the acquaintances of the agent. The nested beliefs of agents may lead to tree-like structures of such units (called *belief contexts*). Such structures have then been used to solve problems like the three wise men [3]. In our case, however, any nested beliefs are included in a single unit and we provide a more comprehensive formalisation of an autonomous agent in that we additionally show how attitudes other than belief can be incorporated into the architecture.

## 5 Conclusions and future work

This paper has demonstrated how multi-context agents which use argumentation-based reasoning might be developed. The examples show how agents capable of flexible and sophisticated argumentation can be specified both in general terms and in terms of a particular type of agent (namely a BDI agent). We have shown how agents can construct arguments to justify their proposals and how agents can exchange arguments to help guide their problem solving behaviour towards mutually acceptable solutions. Moreover, by basing our framework on multi-context systems we are able to show a clear link to potential implementations of agents which are built in the manner we have advocated. This link can be achieved by implementing the various units as concurrent theorem provers with connections between them as specified by the bridge rules. This gives a very natural connection between the formal model and implementation—one we claim is more natural than in many other cases and which is the major benefit of our approach. There are a number of other benefits in terms of the practical implementation of our agents which follow from using the multi-context approach [4]. First, the modular organisation of the architecture's components (in our case the BDI modalities) in different units reduces the complexity of the theorem proving mechanism. Second, it is easier to define proof strategies as combinations of the simple deductive elements in the system (local reasoning in the units and the application of bridge rules) than it is to have a monolithic, all encompassing approach.

A number of issues raised in this paper require further investigation. Most prominent amongst these is the need to produce an implementation which supports both the generic definition of agent architectures and the specific instantiations for particular types of agent. Secondly, the notion of attacking inference steps, as discussed in section 3.1, needs to be more fully elaborated to both ascertain whether it is useful and whether it can be achieved in a tractable manner. Thirdly, the means by which agents generate and rate arguments needs to be expanded. Acceptability classes provide a means of ordering arguments, but it is likely that we will require the ability to provide a more fine-grained ranking (see step 2 of the final example). Finally, agents need effective internal mechanisms for tracking and maintaining their arguments and propagating changes in their preferences as their knowledge changes over time (as illustrated in step 3 of the final example).

## References


[1] G. Attardi and M. Simi. A formalisation of viewpoints. *Fundamenta Informaticae*, 23(2,3,4):149–174, 1995.

[2] R. A. Brooks. Intelligence without reason. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, pages 569–595, 1991.

[3] A. Cimatti and L. Serafini. Multi-agent reasoning with belief contexts: The approach and a case study. In *Proceedings of the 3rd International Workshop on Agent Theories, Architectures and Languages*, 1994.

[4] A. Cimatti and L. Serafini. Multi-agent reasoning with belief contexts III: Towards the mechanization. In *Proceedings of the IJCAI Workshop on Modelling Context in Knowledge Representation and Reasoning in Artificial Intelligence*, pages 35–45, 1995.

[5] P. R. Cohen and H. J. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42:213–261, 1990.

[6] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and $n$-person games. *Artificial Intelligence*, 77:321–357, 1995.

[7] M. Elvang-Gøransson, P. Krause, and J. Fox. Dialectic reasoning with inconsistent information. In *Proceedings of the 9th Conference on Uncertainty in Artificial Intelligence*, pages 114–121, 1993.

[8] E. A. Emerson. Temporal and Modal Logic. In J van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 996–1071. Elsevier, 1990.

[9] J. Fox, P. Krause, and S. Ambler. Arguments, contradictions and practical reasoning. In *Proceedings of the 10th European Conference on Artificial Intelligence*, pages 623–627, 1992.

[10] D. Gabbay. *Labelled Deductive Systems.* Oxford University Press, Oxford, UK, 1996.

[11] M. P. Georgeff and A. L. Lansky. Reactive reasoning and planning. In *Proceedings of the 6th National Conference on Artificial Intelligence*, pages 677–682, 1987.

[12] F. Giunchiglia and L. Serafini. Multilanguage hierarchical logics (or: How we can do without modal logics). *Artificial Intelligence*, 65:29–70, 1994.

[13] F. Guinchiglia. Contextual reasoning. In *Proceedings of the IJCAI Workshop on Using Knowledge in Context*, 1993.

[14] F. F. Ingrand, M. P. Georgeff, and A. S. Rao. An architecture for real-time reasoning and system control. *IEEE Expert*, 7(6), 1992.

[15] N. R. Jennings. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. *Artificial Intelligence*, 75:195–240, 1995.

[16] N. R. Jennings, E. H. Mamdani, J. Corera, I. Laresgoiti, F. Perriolat, P. Skarek, and L. Z. Varga. Using ARCHON to develop real-word DAI applications Part 1. *IEEE Expert*, 11:64–70, 1996.

[17] P. Krause, S. Ambler, M. Elvang-Gøransson, and J. Fox. A logic of argumentation for reasoning under uncertainty. *Computational Intelligence*, 11:113–131, 1995.

[18] R. Loui. Defeat among arguments: a system of defeasible inference. *Computational Intelligence*, 3:100–106, 1987.

[19] P. Noriega and C. Sierra. Towards layered dialogical agents. In *Proceedings of the 3rd International Workshop on Agents Theories, Architectures and Languages*, pages 157–171, 1996.

[20] S. Parsons and N. R. Jennings. Negotiation through argumentation—a preliminary report. In *Proceedings of the International Conference on Multi Agent Systems*, pages 267–274, 1996.

[21] J. L. Pollock. Justification and defeat. *Artificial Intelligence*, 67:377–407, 1994.

[22] A. Rao and M. Georgeff. Asymmetry thesis and side-effect problems in linear time and branching time intention logics. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, 1991.

[23] A. Rao and M. Georgeff. BDI agents: From theory to practice. In *Proceedings of the 1st International Conference on Multi-Agent Systems*, pages 312–319, 1995.

[24] A. S. Rao and M. P. Georgeff. Modeling Rational Agents within a BDI-Architecture. In *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning*, pages 473–484, 1991.

[25] A. S. Rao and M. P. Georgeff. Formal Models and Decision Procedures for Multi-Agent Systems. Technical Note 61, Australian Artificial Intelligence Institute, 1995.

[26] Y. Shoham. Agent-oriented programming. *Artificial Intelligence*, 60:51–92, 1993.

[27] C. Sierra, N. R. Jennings, P. Noriega, and S. Parsons. A framework for argumentation-based negotiation. In *Proceedings of the 4th International Workshop on Agent Theories, Architectures and Languages*, 1997.

[28] I. Sommerville. *Software Engineering.* Addison Wesley, Wokingham, UK, 1992.

[29] F. H. van Eemeren, R. Grootendorst, F. S. Henkemans, J. A. Blair, R. H. Johnson, E. C. W. Krabbe, C. Plantin, D. N. Walton, C. A. Willard, J. Woods, and D. Zarefsky. *Fundamentals of Argumentation Theory: A Handbook of Historical Backgrounds and Contemporary Developments.* Lawrence Erlbaum Associates, Mahwah, NJ, 1996.

[30] L. Vila. *On temporal representation and reasoning in knowledge-based systems.* IIIA Monographies, Barcelona, Spain, 1994.

[31] G. Vreeswijk. The feasibility of defeat in defeasible reasoning. In *Proceedings of the 1st International Conference on Knowledge Representation and Reasoning*, pages 526–534, 1989.