New Trends in Software Methodologies, Tools and Techniques (SoMeT 2008)
H. Fujita (Ed.)
IOS Press, 2008.

Designing and Implementing B2B Applications Using Argumentative Agents

Jamal BENTAHAR [a], Nanjangud NARENDRA [b], Zakaria MAAMAR [c], Rafiul ALAM [a], and Philippe THIRAN [d]

[a] *Concordia University, Canada*
[b] *IBM India Research Lab, India*
[c] *Zayed University, U.A.E*
[d] *Namur University, Belgium*

Abstract. This paper presents a framework for modeling and deploying $\mathcal{B}$usiness-to-$\mathcal{B}$usiness ($\mathcal{B}2\mathcal{B}$) applications, with autonomous agents exposing the individual components that implement these applications. This framework consists of three levels identified by strategic, application, and resource, with focus here on the first two levels. The strategic level is about the common vision that independent businesses define as part of their decision of partnership. The application level is about the business processes that get virtually integrated as result of this common vision. Since conflicts are bound to arise among the independent applications/agents, the framework uses a formal model based upon computational argumentation theory through a persuasion protocol to detect and resolve these conflicts. In this protocol, agents can reason about partial information using *partial arguments*, *partial attack* and *partial acceptability*. Agents can then jointly find arguments supporting a new solution for their conflict, which is not known by any of them individually. Termination, soundness, and completeness properties of this protocol are presented. Distributed and centralized coordination strategies are also supported in this framework, which is illustrated with a simple online purchasing case study.

Keywords. $\mathcal{B}2\mathcal{B}$, argumentation theory, agent communication, conflict, persuasion.

Introduction

Performance and competitivity challenges are nowadays putting businesses under constant pressure to meet changing requirements. This fuels the need for continuous merge and sometimes re-engineering of business processes, resulting in $\mathcal{B}$usiness-to-$\mathcal{B}$usiness ($\mathcal{B}2\mathcal{B}$) applications development. Briefly, a $\mathcal{B}2\mathcal{B}$ application is a set of business processes that make disparate autonomous entities (e.g., departments, businesses) collaborate to achieve a common set of goals. Despite the multiple initiatives on $\mathcal{B}2\mathcal{B}$ applications [13,17,22,24], not much exists in terms of modeling and deploying such applications from intelligent and argumentative-agent perspective. By modeling, we mean identifying all the necessary components that connect assets of independent entities engaged in a $\mathcal{B}2\mathcal{B}$ scenario. By deployment, we mean identifying all the necessary technologies that make the connection of these assets happen effectively. Finally, by argumentation we mean making agents comply with a dialectical process to affirm or disavow the conclusions that these agents wish to reciprocally convey. In a $\mathcal{B}2\mathcal{B}$ scenario, argumentation would broadly mean assisting businesses, through representative agents, engage in intense negotiation and persuasion sessions prior to making any joint decisions. The argumentation capability of an agent representing a business, can assist this business in negotiating with its peers during a conflict situation and in collaborating with them to achieve agreements about their strategies. This paper addresses the challenge of using argumentation theory for multi-agent systems to develop $\mathcal{B}2\mathcal{B}$ applications, and a case study is used to illustrate the proposed framework. This framework is an initiative within the emerging field of developing intelligent software [11,9]. The technique we are using in this paper is different from other techniques proposed in this field such as the Lyee methodology [12].

This framework suggests three levels, *resource*, *application*, and *strategic*[1] that are connected through *rely-on* and *run-on-top-of* relations (Fig. 1). These levels represent the way businesses generally function: the strategic level, associated with a set of $\mathcal{S}$trategic Argumentative Agents ($\mathcal{S}$-AAs), sets the goals to reach (e.g., 10% revenue increase), the application level, associated with a set of $\mathcal{A}$pplication Argumentative Agents ($\mathcal{A}$-AAs), sets the automatic and manual processes (e.g., new auditing system) that permit fulfilling these objectives, and the resource level, associated with a set of $\mathcal{R}$esource Argumentative Agents ($\mathcal{R}$-AAs), sets the means that achieve the performance of these processes. The framework couples components (that reside in one of the three levels) with agents equipped with argumentation capabilities to assist a specific component (i) persuade peers of collaborating, (ii) interact with peers during business process implementation, (iii) resolve conflicts that could impede collaboration, and (iv) track conflict resolution. Still in Fig. 1, rely-on relation means mapping the business objectives onto concrete system applications, and run-on-top-of relation means performing these system applications' business processes subject to resource availabilities. In addition, both relations make issues at lower levels influence goals at higher levels. For example, lack of resources could result in reviewing goals.

In Fig. 1, horizontal relations permit linking similar levels of separate businesses. We refer to these relations by *interconnectivity*, *composition*, and *collaboration*. Underneath each horizontal relation's name, an example of conflict to fix in a $\mathcal{B}2\mathcal{B}$ scenario is shown for illustration purposes. Collaboration relation bridges the strategic levels and focusses on how businesses adapt their goals and plans so that these businesses can now reach the goals that result out of their decision of partnership. Composition relation bridges the application levels and focusses on how new business processes are developed, either from scratch or after re-engineering existing processes. Finally, interconnectivity relation bridges the resource levels and focusses on the means that make the performance of business processes happen despite distribution and heterogeneity constraints.

Section 1 introduces our agent-based framework for $\mathcal{B}2\mathcal{B}$ applications. Section 2 presents the argumentation model upon which this framework operates. Section 3 discusses the argumentative protocol for $\mathcal{B}2\mathcal{B}$ conflict resolution and analyzes its formal and computational properties. A case study illustrating this model through a running example is provided in Section 4. Prior to concluding and presenting future work in Section 6, related work is discussed in Section 5.

[1] Decisions affecting a business growth are made at this level.

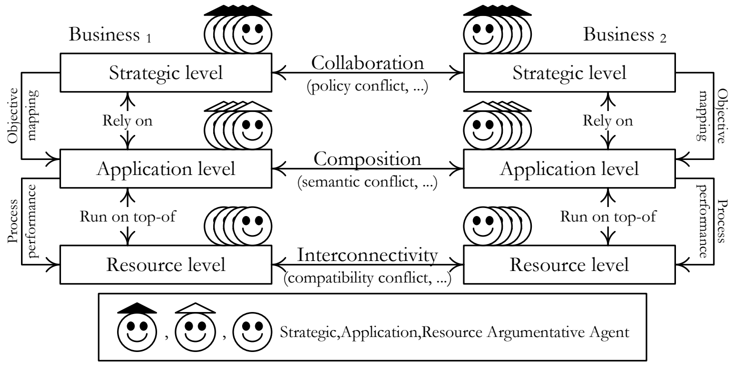

Figure 1. The argumentative agent framework for $\mathcal{B}2\mathcal{B}$ applications.

1. The Proposed Framework for $\mathcal{B}2\mathcal{B}$ Applications

1.1. Brief Description of Levels & Relations

The resource level includes data and software resources (e.g., DBMS) that a business owns or manages, and the hardware resources upon which these software resources run.

The application level is about the software applications that businesses operate such as payroll. From a $\mathcal{B}2\mathcal{B}$ perspective, the application level hosts a number of $\mathcal{A}$-AAs according to the number of these applications. The role of $\mathcal{A}$-AAs is (i) to monitor the external business processes that will make use of software applications and (ii) to initiate interaction sessions with other $\mathcal{A}$-AAs. These sessions frame application compositions according to the guidelines that $\mathcal{S}$-AAs set and resolve possible conflicts during these compositions as depicted by *composition* relation in Fig. 1. For illustration purposes we assume that software applications are implemented as Web services [16], although other technologies could be used.

The strategic level is about the planning and decision-making mechanisms that underpin a business growth. Like the application level, the strategic level hosts a number of $\mathcal{S}$-AAs according to the number of active collaborations that a business initiates with its partners. The role of $\mathcal{S}$-AAs is (i) to reason over the business plans and (ii) initiate interaction sessions with other $\mathcal{S}$-AAs as depicted by *collaboration* relation in Fig. 1. These sessions aim at persuading peers to participate in collaborations, reviewing policies in case of conflicts, optimizing some parameters such as distribution network, etc. $\mathcal{A}$-AAs feed $\mathcal{S}$-AAs with details related to the execution progress of business processes. Particularly, if a conflict during the composition process cannot be resolved, $\mathcal{A}$-AAs inform their respective $\mathcal{S}$-AAs.

From an argumentation perspective, $\mathcal{S}$-AAs and $\mathcal{A}$-AAs are equipped with the same reasoning capabilities. However they differ in terms of the knowledge they manage and the responsibilities they are in charge of. For example, to resolve conflicts at the application or strategic levels, $\mathcal{A}$-AAs or $\mathcal{S}$-AAs use the same persuasion and negotiation protocols but execute them differently. Protocols publicly describe the allowed moves, but how to select a certain move would dependent on the knowledge that feed agents' private strategies.

Fig. 1 shows vertical and horizontal relations. In a $\mathcal{B}2\mathcal{B}$ context, the focus is on horizontal relations. Interconnectivity relation targets the resource level and allows (i) data to freely and securely flow between businesses without any restriction related to format, location, or semantics and (ii) disparate resources to trigger each other without any restriction related to access rights, time-slot availabilities, or compatibilities. Communication-protocol incompatibility (e.g., different vendors) is an example of conflict that falls under interconnectivity relation.

Composition relation targets the application level and exhibits how business processes associated with $\mathcal{A}$-AAs get "virtually" integrated without being subject to any functional or structural changes. Lack of common semantics (e.g., different measurement units) is an example of conflict that falls under composition relation. When it comes to Web services-based applications, composition targets users' requests that cannot be satisfied by any single, available Web service, whereas a composite Web service obtained by combining available Web services may be used.

Collaboration relation targets the strategic level and emphasizes the mechanisms that $\mathcal{S}$-AAs set-up for coordinating the new $\mathcal{B}2\mathcal{B}$ processes using $\mathcal{A}$-AAs. These processes result out of composing applications, stretch beyond businesses' boundaries, and have to consider the requirements/limitations of the resource and application levels per business. Policy incompatibility (e.g., various tax rates) is an example of conflict that falls under collaboration relation. Policies of businesses can be in contradiction, and some core business policies cannot be easily re-engineered. By using argumentative agents, we aim at handling these issues. Through their argumentative reasoning, and interaction, negotiation, and persuasion abilities, these agents could reason about these policies, identify possible conflicts and update their policies to resolve these conflicts. They can also persuade each other for the benefit of collaborating and sharing their resources and determining alternative agents to work with, in case current conflicts cannot be resolved.

1.2. Forms of Coordination

We split coordination in the argumentative agent-based framework into two forms: *vertical* between strategic and application levels via rely-on relation, and *horizontal* between strategic or application levels via collaboration or composition relations, respectively. We discuss hereafter how argumentation is woven with coordination using Figs. 2 and 3 where plain lines and dotted lines denote interactions and conflict detection/resolution respectively.

Vertical Coordination occurs within the boundaries of the same business. Here a $\mathcal{S}$-AA has the authority to execute a set of actions over an $\mathcal{A}$-AA (Fig. 2): "select", "ping", "trigger", "audit", and "retract & select". Due to lack of space, only few of them are presented:

1. "*select*" action makes the $\mathcal{S}$-AA identify the $\mathcal{A}$-AA of an application that will pursue the interactions with other $\mathcal{A}$-AAs as part of the partnership decision;

2. "*trigger*" action makes the $\mathcal{S}$-AA forward the execution requests to the $\mathcal{A}$-AA of an application; these requests arrive from others $\mathcal{A}$-AAs;

3. "*audit*" action makes the $\mathcal{S}$-AA monitor the performance of an application through its $\mathcal{A}$-AA; this is needed if the $\mathcal{S}$-AA has to guarantee a certain QoS to other $\mathcal{S}$-AAs.

Argumentation in vertical coordination is illustrated with two cases: *Application-to-Strategic* (this paper focus) and *Strategic-to-Application*.

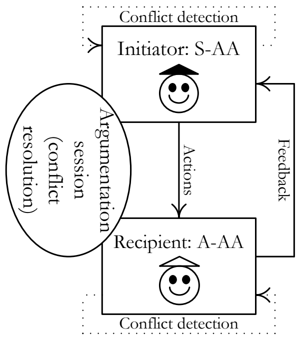

Figure 2. Argumentation in vertical coordination

Application-to-Strategic case highlights an $\mathcal{A}$-AA that faces difficulties in resolving conflicts and completing its operations. For example, this $\mathcal{A}$-AA was put on hold for a long period of time due to occupied resources or did not receive information in the right format from other businesses' $\mathcal{A}$-AAs. As a result, the $\mathcal{A}$-AA notifies its $\mathcal{S}$-AA so that both set up an argumentation session for the sake of discussing the current difficulties and the potential solutions to put forward. This notification is represented with "feedback" in Fig. 2. Briefly, we report on the way conflict resolution progresses in this argumentation session.

Case 1. The $\mathcal{S}$-AA has an argument supporting the fact that the conflict facing the $\mathcal{A}$-AA could be resolved based on similar past situations for example. Thus, the $\mathcal{S}$-AA argues with the $\mathcal{A}$-AA about the feasibility of this solution using persuasion (Section 3.2). If the $\mathcal{A}$-AA is not convinced (i.e., persuasion fails), the $\mathcal{S}$-AA will decide to select another $\mathcal{A}$-AA to continue the uncompleted composition work of the withdrawn $\mathcal{A}$-AA.

Case 2. The $\mathcal{S}$-AA does not have any argument for or against the possibility of resolving the conflict facing the $\mathcal{A}$-AA. Thus, $\mathcal{S}$-AA and $\mathcal{A}$-AA collaborate to find a solution through an inquiry dialogue game like the one proposed in [5]. As defined by Walton and Krabbe [23], inquiry dialogues rise from an initial situation of general ignorance and the purpose is to achieve the growth of knowledge and agreement.

If neither **case** (**1**) or (**2**) succeeds, the respective $\mathcal{S}$-AAs of the collaborative businesses try to work out a solution via horizontal coordination. When a solution is found, the $\mathcal{S}$-AAs invite the same $\mathcal{A}$-AAs if they are still available, or new ones to take part in the composition to deploy at the application level.

Strategic-to-Application case highlights a $\mathcal{S}$-AA that expects the occurrence of conflicts if appropriate actions are not taken on time. Examples of actions include reprimanding an $\mathcal{A}$-AA that released private details to peers. Expecting conflicts is based on the different feedbacks that the $\mathcal{S}$-AA receives from their $\mathcal{A}$-AAs. This shows a preventive strategy to conflict occurrence. However, preventive strategies are beyond the scope of this paper.

Horizontal Coordination spreads over the boundaries of businesses and thus, reflects $\mathcal{B}2\mathcal{B}$ applications in a better way. We identify two scenarios where each scenario involves either $\mathcal{S}$-AAs or $\mathcal{A}$-AAs. For the sake of simplicity, our description is restricted to $\mathcal{A}$-AAs. Here an $\mathcal{A}$-AA has the authority to carry out a set of actions over another peer engaged in the same composition (Fig. 3): "ping" and "trigger".

1. "ping" action makes the $\mathcal{A}$-AA check the liveness of a remote application through its $\mathcal{A}$-AA; this is needed before the former $\mathcal{A}$-AA submits requests;

2. "trigger" action makes the $\mathcal{A}$-AA submit its requests to a remote application through its $\mathcal{A}$-AA.

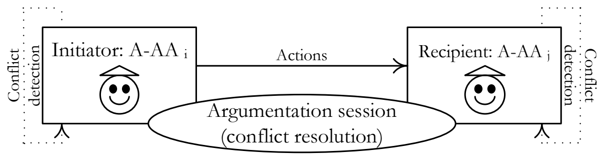

Figure 3. Argumentation in horizontal coordination

Argumentation in horizontal coordination is illustrated with two cases: *Application-to-Application* and *Strategic-to-Strategic*.

Application-to-Application case stresses an $\mathcal{A}$-AA that identifies a conflict after interacting with peers. Conflicts could be of many types like different security policies, different semantics (e.g. different measurement units, different ontologies, etc.), conflicting quality of service, different cost associated with the application, etc.

$\mathcal{A}$-AAs try to resolve these conflicts via argumentation using a combination of *persuasion* and *inquiry* (Section 3.2). $\mathcal{A}$-AA agents engage in pure persuasion if one of them has already a solution that could be accepted by the other with respect to the beliefs it has. However, merging persuasion with inquiry allows these agents to build up a joint agreed argument.

Strategic-to-Strategic case highlights a $\mathcal{S}$-AA that identifies a conflict and tries to resolve it with its $\mathcal{S}$-AA partner. Some conflicts at this level concern penalty policies (e.g., collaboration's contract terms and conditions not respected) and payment policies. This case also stresses the situation where two $\mathcal{S}$-AAs of the collaborative businesses try to work out a solution of a conflict reported by the respective $\mathcal{A}$-AAs which cannot be resolved by vertical coordination. To resolve these conflicts, $\mathcal{S}$-AAs engage in persuasions and inquiries (Section 3.2). Before presenting this protocol, let us discuss its formal framework based on computational argumentation theory.

2. Formal Argumentation System

2.1. Generic Background

This section discusses the formal argumentation system that frames the internal operations in our $\mathcal{B}2\mathcal{B}$ framework. This discussion includes the configuration featuring argumentative agents as well. We use an abstract formal language $\mathcal{L}$ to express agents' beliefs. Here abstract means that beliefs could be propositional formulas like in [18], Horn clauses like in [3], or a set of facts and rules like in [5,10]. The use of an abstract language would make our framework generic and able to capture properties that other frameworks concentrate on. The set of well-formed formulas (wff) built from $\mathcal{L}$ is denoted by $\mathcal{WF}$. Agents build arguments using their beliefs. The set $Arg(\mathcal{L})$ contains all those arguments. Similarly to [1,6,7], we abstractly define argumentation as a dialectical process that underpins the exchange of for/against arguments that lead to some conclusion. Because we are using an abstract language, we are not interested in the internal form of an argument. Formally, we define our argumentation framework as follows:

Definition 1 (Argumentation Framework) *An abstract argumentation framework is a pair* $\langle A, \mathcal{AT} \rangle$*, where* $A \subseteq Arg(\mathcal{L})$*, and* $\mathcal{AT} \subseteq A \times A$ *is a binary relation over* A *that is not necessarily symmetric. For two arguments* a *and* b*, we use* $\mathcal{AT}(a, b)$ *instead of* $\mathcal{AT} \in (a, b)$ *to indicate that* a *is an attack against* b*.*

For example, an argument may be defined as a deduction of a conclusion from a given set of rules, or as a pair (H, h) where h is a sentence in $\mathcal{WF}$ and H a subset of a given knowledge base such that (i) $H \vdash h$, (ii) H is consistent, and (iii) there is no subset of H with properties (i) and (ii).

As conflicts between arguments might occur, we need to define what an *acceptable argument* is. To this end we define first the notions of "defense" and "admissible set of arguments" (from [7,8]):

Definition 2 (Defense) *Let* $A \subseteq Arg(\mathcal{L})$ *be a set of arguments over the argumentation framework, and let* $S \subseteq A$*. An argument* a *is defended by* S *iff* $\forall b \in A$ *if* $\mathcal{AT}(b, a)$*, then* $\exists c \in S : \mathcal{AT}(c, b)$*.*

Definition 3 (Admissible Set) *Let* $A \subseteq Arg(\mathcal{L})$ *be a set of arguments over the argumentation framework. A set* $S \subseteq A$ *of arguments is admissible iff:*
1) $\nexists\, a, b \in S$ *such that* $\mathcal{AT}(a, b)$ *and*
2) $\forall a \in S$ a *is defended by* S*.*

In other words, a *set of arguments* is admissible iff it is conflict-free and can counter-attack every attack.

***Example* 1** *Let* $A = \{a, b, c, d\}$ *and* $\mathcal{AT}$ *defined as follows:* $\mathcal{AT}(b, a)$*,* $\mathcal{AT}(c, a)$*,* $\mathcal{AT}(d, b)$*,* $\mathcal{AT}(d, c)$*. The sets:* $\emptyset, \{d\}$ *and* $\{a, d\}$ *are all admissible. However, the sets* $\{b\}$ *and* $\{d, c\}$ *are not admissible.*

Definition 4 (Characteristic Function) *Let* $A \subseteq Arg(\mathcal{L})$ *be a set of arguments and let* S *be an admissible set of arguments over the argumentation framework. The characteristic function of the argumentation framework is:*

$$F : 2^A \rightarrow 2^A$$
$$F(S) = \{a \mid a\ is\ defended\ by\ S\}$$

Definition 5 (Extensions) *Let* S *be an admissible set of arguments, and let* F *be the characteristic function of the argumentation framework.*

- S *is a complete extension* (S_{co}) *iff* $S = F(S)$*.*
- S *is the grounded extension* (S_{gr}) *iff* $S = F(S)$ *and* S *is minimal* $($*w.r.t. set-inclusion*$)$ $($*grounded extension corresponds to the least fixed point of* $F$$)$*.*
- S *is a preferred extension* (S_{pr}) *iff* $S = F(S)$ *and* S *is maximal* $($*w.r.t. set-inclusion*$)$*.*

***Example* 2** *Let us consider the same argumentation framework as in Example 1.*

We have:

- $F(\emptyset) = \{d\}$*, so the admissible set* $\emptyset$ *is not a complete extension.*
- $F(\{d\}) = \{a, d\}$*, so the admissible set* $\{d\}$ *is not a complete extension.*
- $F(\{a, d\}) = \{a, d\}$*, so the admissible set* $\{a, d\}$ *is a complete extension.*

In this example, the only complete extension $\{a, d\}$ *is the grounded extension and is also the only preferred extension.*

***Example* 3** *Let A=*$\{$*a,b,c*$\}$ *and* $\mathcal{AT}$ *defined as follows:* $\mathcal{AT}(a, b)$*,* $\mathcal{AT}(b, a)$*. The sets:* $\{c\}, \{a, c\}$*, and* $\{b, c\}$ *are the complete extensions of the argumentation framework. The minimal complete extension* $\{c\}$ *is the grounded extension, and the maximal complete extensions* $\{a, c\}$ *and* $\{b, c\}$ *are the preferred extensions.*

According to Definition 5, an admissible set S is a complete extension if and only if S is a fixed point of the function F, which means that all arguments defended by S are also in S. Also, the grounded extension is the *least fixed point* of F. Consequently, the grounded extension contains all the arguments that are not attacked (the arguments that are defended by the empty set: $F(\emptyset)$), all the arguments that are defended by these non-attacked arguments $F(F(\emptyset)) = F^2(\emptyset)$, all the arguments that are defended by the defended arguments ($F^3(\emptyset)$), and so on until a fixed point is achieved. The grounded extension corresponds to the intersection of all the complete extensions. Finally, a preferred extension is a maximal complete extension that cannot be augmented by adding another arguments while staying complete.

We have the following direct proposition:

Proposition 1 *Let* $\langle A, \mathcal{AT} \rangle$ *be an argumentation framework.* $\exists! S_{gr}$ *in* $\langle A, \mathcal{AT} \rangle$*.*

In other words, there exists a single grounded extension for the abstract argumentation framework. Now we can define what are the acceptable arguments in our system.

Definition 6 (Acceptable Arguments) *Let* $A \subseteq Arg(\mathcal{L})$ *be a set of arguments, and let* $G = S_{gr}$*. An argument* a *over* A *is acceptable iff* $a \in G$*.*

According to this acceptability semantics, which is based on the grounded extension, if we have two arguments a and b such that $\mathcal{AT}(a, b)$ and $\mathcal{AT}(b, a)$, then a and b are both non-acceptable. In a $\mathcal{B}2\mathcal{B}$ scenario, this can happen when two $\mathcal{A}$-AAs present two conflicting arguments about the type of security policies to use for the current transaction: a weak policy which is simple to implement and less expensive or a strong policy which is hard to implement and more expensive. This notion is important in $\mathcal{B}2\mathcal{B}$ applications since agents should agree on an acceptable opinion, which is supported by an acceptable argument when a conflict arises. However, during the argumentative conversation, agents could use non-acceptable arguments as an attempt to change the status of some arguments previously uttered by the addressee, from acceptable to non-acceptable. This idea of using non-acceptable arguments in the dispute does not exist in the persuasion and inquiry protocols in the literature. For this reason, we introduce two new types of arguments based on the preferred extensions to capture this notion. We call these arguments *semi-acceptable and preferred semi-acceptable arguments*.

Definition 7 ((Preferred) Semi-Acceptable Arguments) *Let G be the grounded extension in the argumentation framework, and let $E_1, \ldots, E_n$ be the preferred extensions in the same framework. An argument a is:*

- *Semi-acceptable iff $a \notin G$ and $\exists\ E_i, E_j$ with $(1 \leq i, j \leq n)$ such that $a \in E_i \wedge a \notin E_j$.*
- *Preferred semi-acceptable iff $a \notin G$ and $\forall\ E_i\ (1 \leq i \leq n)\ a \in E_i$.*

In other words, an argument is *semi-acceptable* iff it is not acceptable and belongs to some preferred extensions, but not to all of them. An argument is *preferred semi-acceptable* iff it is not acceptable and belongs to all the preferred extensions. Preferred semi-acceptable arguments are stronger than semi-acceptable and grounded arguments are the strongest arguments in this classification.

***Example* 4** *Let $A = \{a, b, c, d\}$ and $\mathcal{AT}$ is defined as follows: $\mathcal{AT}(a, b)$, $\mathcal{AT}(b, a)$, $\mathcal{AT}(a, c)$, $\mathcal{AT}(b, c)$, $\mathcal{AT}(c, d)$.*

- *$\emptyset$ is the grounded extension in this argumentation framework.*
- *The argumentation framework has two preferred extensions: $\{a, d\}$ and $\{b, d\}$. The arguments a and b are then semi-acceptable, and the argument d is preferred semi-acceptable.*

A concrete scenario of this example in a $\mathcal{B}2\mathcal{B}$ setting would be as follows: Suppose we have a transaction Tr and four possible security policies for it: s_1, s_2 and s_3. The four arguments a, b, c and d are as follows:

- *a: s_1 is the most suitable policy for the transaction Tr.*
- *b: Alone, s_2 is not sufficient to secure the transaction Tr, but by combining it with s_3 it becomes the most suitable.*
- *c: s_2 is less expensive than s_1.*
- *d: s_1 is not expensive to implement, and is sufficient to secure the transaction Tr.*

In some extent, the argument d is stronger than a and b because it is defended by these two arguments against the only attacker c, and a and b attacks each other. From a chronological point of view, we can imagine the following scenario leading to build these four arguments at the application level of two businesses represented respectively by $\mathcal{A}$-AA_1 and $\mathcal{A}$-AA_2. First, $\mathcal{A}$-AA_1 presents the argument d, then $\mathcal{A}$-AA_2 attacks by moving forward the argument c. $\mathcal{A}$-AA_1 replies by attacking c using the argument a. At that stage, arguments a and d are grounded. $\mathcal{A}$-AA_2 tries then to degrade one of these two arguments by attacking a using d. $\mathcal{A}$-AA_2 is aware that by using b to attack a, b is at the same time attacked by a. The idea here is just to change the status of the argument presented by $\mathcal{A}$-AA_1 from acceptable to semi acceptable.

Proposition 2 *Let $A \subseteq Arg(\mathcal{L})$ be a set of arguments, and let $SD = \{a \in A | \forall\ b \in A\ \mathcal{AT}(b, a) \Rightarrow \mathcal{AT}(a, b)\ \&\ \nexists\ c \in A : \mathcal{AT}(c, b)\}$. $\forall a \in SD, a$ is semi acceptable.*

In other words, the arguments defending themselves by only themselves against all the attackers are semi-acceptable.
Proof see Appendix.

Proposition 3 *Complete extensions are not closed under intersection.*

Proof see Appendix.

Definition 8 (Eliminated Arguments) *Let $A \subseteq Arg(\mathcal{L})$ be a set of arguments, $a \in A$ be an argument, and EL be the set of eliminated arguments over the argumentation framework. Also, let $E_1, \ldots, E_n$ be the preferred extensions in the same framework. $a \in EL$ iff $a \notin E_i, \forall\, i \in [1, n]$.*

In other words, an argument is eliminated iff it does not belong to any preferred extension in the argumentation framework. We have the following proposition:

Proposition 4 *Let a be an argument in A, and AC, PS, SA be respectively the sets of acceptable, preferred semi-acceptable, and semi-acceptable arguments over the argumentation framework. $a \in EL$ iff $a \notin AC \cup PS \cup SA$.*

In other words, an argument is *eliminated* iff it is not acceptable, not preferred semi-acceptable, and also not semi-acceptable.
Proof see Appendix.

Consequently, arguments take four exclusive statuses namely acceptable, preferred semi-acceptable, semi-acceptable, and eliminated. The dynamic nature of agent interactions is reflected by the changes in the statuses of *uttered arguments*.

2.2. Partial Arguments and Conflicts for $\mathcal{B}2\mathcal{B}$ Aplications

In a $\mathcal{B}2\mathcal{B}$ scenario, it happens that argumentative agents $\mathcal{S}$-AAs and $\mathcal{A}$-AAs do not have complete information on some facts. In similar situation, they can build *partial arguments* for some conclusions out of their beliefs. We define a partial argument as follows:

Definition 9 (Partial Arguments) *Let x be a wff in $\mathcal{WF}$. A partial argument denoted by a^p_x is part of an argument $a \in A$, which misses an argument (or a proof) for x. In other words, by adding a proof supporting x to a^p_x an argument is obtained.*

For example, if arguments are defined as deductions from a set of rules, x will represent some missing rules, and if arguments are defined as pairs (H, h), x will represent a subset of H.

***Example* 5** *let us suppose that arguments are defined as pairs (H, h) using propositional logic. $a = ((m, m \to n), n)$ is an argument for n and $a^p_x = ((m \to n), n)$ is a partial argument for n missing the support for $x = m$.*
$a = ((m, m \to n, n \to l, l \to r), r)$ is an argument for r and $a^p_x = ((n \to l, l \to r), r)$ is a partial argument for r missing the support for $x = n$. In this case a possible support is $((m, m \to n), n)$.

In a $\mathcal{B}2\mathcal{B}$ scenario, an example where partial arguments are needed is when $\mathcal{A}$-AA$_1$ of business B_1 knows that security policy s_2 that another business B_2 uses can be substituted by policy s_1 that B_1 uses if some conditions are met when deploying s_2. Thus, $\mathcal{A}$-AA$_1$ can build a partial argument supporting the fact that B_2 can use s_1. To be an argument, this partial argument needs a support that implementing s_2 in B_2 mets these conditions.

The idea behind building partial arguments by an agent is to check if the other agent can provide the missing part or a part of this missing part so that the complete argument could be jointly built (progressively). This idea which is a part of the inquiry dialogue will be made clear in the persuasion protocol defined in Section 3.2.

As for arguments, we need to define what an acceptable partial argument is. This acceptability is defined in the same way as for arguments. We use the notation $a_x^p.x$ to denote the resulting argument of combining the partial argument a_x^p and an argument supporting x supposing that this latter exists.

Definition 10 (Partial Attack) *Let a_x^p be a partial argument over the argumentation framework. $\mathcal{AT}(a_x^p, b)$ iff $\mathcal{AT}(a_x^p.x, b)$ and $\mathcal{AT}(b, a_x^p)$ iff $\mathcal{AT}(b, a_x^p.x)$.*

Definition 11 (Acceptable Partial Arguments) *A partial argument a_x^p is acceptable $($preferred semi-acceptable, semi-acceptable$)$ iff $a_x^p.x$ is acceptable $($preferred semi-acceptable, semi-acceptable$)$.*

***Example* 6** *Let $\Sigma = \{n \rightarrow m, r \rightarrow l, l \rightarrow t, \neg l\}$ be a propositional knowledge base. The partial argument $((n \rightarrow m), m)$ is acceptable, however the partial argument $((r \rightarrow l, l \rightarrow t), t)$ is not acceptable since the argument $((r, r \rightarrow l, l \rightarrow t), t)$ is attacked by the argument $(\neg l, \neg l)$.*

Proposition 5 *Let a be an argument in A. If a is acceptable, then $\forall x \in \mathcal{WF}$ a_x^p is acceptable.*

Proof see Appendix.

After having specified the argumentation model, We define the notions of conflict and conflict resolution in our $\mathcal{B}2\mathcal{B}$ framework as follows:

Definition 12 (Conflict) *Let p and q be two wffs in $\mathcal{WF}$. There is a conflict between two argumentative agents α and β about p and q in the $\mathcal{B}2\mathcal{B}$ framework iff one of them $($e.g., $\alpha)$ has an acceptable argument a for p $($denoted $a \uparrow p)$ and the other $($i.e., $\beta)$ has an acceptable argument b for q $(b \uparrow q)$ such that $\mathcal{AT}(a, b)$ or $\mathcal{AT}(b, a)$. We denote this conflict by $\alpha_p \ncong \beta_q$.*

For example, if p and q represent each a security policy s_1 and s_2 such that s_1 and s_2 cannot be used together, then there is a conflict if one agent has an acceptable argument for using s_1 while the other agent has an acceptable argument for using s_2 (the two arguments are conflicting). This conflict arises when both agents need to agree on which security policy to use.

Before defining the notion of conflict resolution, we need to define the notions of interaction and interaction's outcome. An utterance u made by an agent α in a given interaction is denoted $u \rightsquigarrow \alpha$.

Definition 13 (Interaction) *Let α and β be two argumentative agents. An interaction $($denoted by $I_{\alpha,\beta})$ between α and β in the $\mathcal{B}2\mathcal{B}$ framework is an ordering sequence of utterances $u_1, u_2, \ldots, u_n$ such that $u_i \rightsquigarrow \alpha \Rightarrow u_{i+1} \rightsquigarrow \beta$ and $u_i \rightsquigarrow \beta \Rightarrow u_{i+1} \rightsquigarrow \alpha$. CS_α $($resp. $CS_\beta)$ is the set $($called commitment store$)$ containing the arguments used by α $($resp. $\beta)$ during the interaction.*

Definition 14 (Conflict Resolution) *Let p and q be two wffs in $\mathcal{WF}$ and α and β be two argumentative agents in the $\mathcal{B}2\mathcal{B}$ framework such that $\alpha_p \ncong \beta_q$. Also let $I_{\alpha,\beta}$ be an* *interaction in this framework. The conflict $\alpha_p \ncong \beta_q$ is resolved by the interaction $I_{\alpha,\beta}$ iff the outcome of $I_{\alpha,\beta}$ is a formula $r \in \mathcal{WF}$ such that $\exists\, a \in CS_\alpha, b \in CS_\beta : a \uparrow r, b \uparrow r$ and a and b are both acceptable.*

In the aforementioned security example, the conflict is resolved iff (i) after interaction, one of the agents can build an acceptable argument from its knowledge base and the arguments exchanged during this interaction, supporting the use of the other policy, or (ii) when both agents agree on the use of a new policy such that each agent can build an acceptable argument, from its knowledge base and the exchanged arguments, supporting the use of this policy. The idea here is that by exchanging arguments, new solutions (and arguments supporting these solutions) can emerge. In this case, agents should update their beliefs by withdrawing attacked (i.e. eliminated) assumptions. However, there is still a possibility that each agent keeps its viewpoint at the end of the conversation.

3. Argumentative Persuasion for $\mathcal{B}2\mathcal{B}$

3.1. Notations

The outcome of an interaction aiming to resolve a conflict in a $\mathcal{B}2\mathcal{B}$ setting depends on the status of the formula representing the conflict topic. As for arguments, a wff has four statuses depending on the statuses of the arguments supporting it (an argument supports a formula if this formula is the conclusion of that argument). A wff is *acceptable* if there exists an acceptable argument supporting it. If not, and if there exists a preferred semi-acceptable argument supporting it, then the formula is preferred semi-acceptable. Otherwise, the formula is semi-acceptable if a semi-acceptable argument supporting it exists, or eliminated if such an argument does not exist. Let St be the set of these statuses. We define the following function that returns the status of a wff with respect to a set of arguments:

$$\Delta : \mathcal{WF} \times 2^{A} \rightarrow St$$

Generally, the interactions we need in a $\mathcal{B}2\mathcal{B}$ scenario involve two argumentative agents. For simplicity, we will not refer in the remainder of the paper to agent types (strategic or application), but denote participating agents by α and β. Each agent has a possibly inconsistent belief base Σ_α and Σ_β respectively containing, for example, all the policies on which these agents should reason when they manage businesses as explained in previous sections.

Agents use their argumentation systems to decide about the next move to play (e.g., accept or attack the arguments advanced during their interactions). When an agent accepts an argument that an addressee suggests, this agent updates its knowledge base by adding the elements of this argument and removing all the elements that attack this argument. Each agent α has also a commitment store (CS_α) publicity accessible for reading but only updated by the owner agent. The commitment stores are empty when interaction starts, and updated by adding arguments and partial arguments that the agents exchange. CS_α refers to the commitment store of agent α *at the current moment*.

The possibility for an agent α to build an acceptable argument a (respectively an acceptable partial argument a^p_x) from its knowledge base and the commitment store of the addressee β is denoted by $\mathcal{AR}(\Sigma_\alpha \cup CS_\beta) \rhd a$ (respectively $\mathcal{AR}(\Sigma_\alpha \cup CS_\beta) \rhd a_x^p$). Building a partial argument a_x^p from a knowledge base means that no argument for or against x can be built. $\mathcal{AR}(\Sigma_\alpha \cup CS_\beta) \not\rhd a$ (respectively $\mathcal{AR}(\Sigma_\alpha \cup CS_\beta) \not\rhd a_x^p$) means that agent α cannot build an acceptable argument a (respectively an acceptable partial argument a_x^p) from $\Sigma_\alpha \cup CS_\beta$. The symbols $\unrhd$ and $\ntrianglerighteq$ associated with semi-acceptable (partial) arguments are defined in the same way.

3.2. Protocol Specification

In our $\mathcal{B}2\mathcal{B}$ framework, agents engage in persuasion and inquiry dialogues to resolve conflicts. Atkinson et al. [2], Pasquier et al. [19], and Prakken [20] propose persuasion protocols for multi-agent systems. However, these protocols consider only pure persuasion without inquiry stages and do not address completeness (or pre-determinism) property [18]. We propose a persuasion protocol including inquiry stages for our $\mathcal{B}2\mathcal{B}$ framework, in which pre-determinism is considered. The protocol is modeled with dialogue games [14,15]. Dialogue games are interactions between players (agents), in which each player moves by performing utterances according to a pre-defined set of rules. Let us define the notions of protocol and dialogue games.

Definition 15 (Protocol) *A protocol is a pair $\langle \mathcal{C}, \mathcal{D} \rangle$ with $\mathcal{C}$ a finite set of allowed moves and $\mathcal{D}$ a set of dialogue games.*

The moves in $\mathcal{C}$ are of c different types ($c > 0$). We denote by $M_i(\alpha, \beta, a, t)$ a move of type i played by agent α and addressed to agent β at time t regarding a content a. We consider four types of moves in our protocol: $Assert$, $Accept$, $Attack$, and $Question$. Generally, in the persuasion protocol agents exchange arguments. Except the $Question$ move whose content is not an argument, the content of other moves is an argument a ($a \in Arg(\mathcal{L})$). When replying to a $Question$ move, the content of $Assert$ move can also be a partial argument or "?" when the agent does not know the answer. We use another particular move $Stop$ with no content. It could be played by an agent to stop the interaction.

Intuitively, a dialogue game in $\mathcal{D}$ is a rule indicating the possible moves that an agent could play following a move done by an addressee. This is specified formally as follows:

Definition 16 (Dialogue Game) *A dialogue game Dg is either of the form:*

$$M_i(\alpha, \beta, a_i, t) \Rightarrow \bigvee_{0<j\leq n_i} M_j(\beta, \alpha, a_j, t')$$

where M_i, M_j are in $\mathcal{C}$, $t < t'$ and n_i is the number of allowed moves that β could perform after receiving a move of type i from α;
or of the form:

$$\Rightarrow \bigvee_{0<j\leq n} M_j(\alpha, \beta, a_j, t_0)$$

where M_j are in $\mathcal{C}$, t_0 is some initial time, and n is the number of allowed moves that α could perform initially.

According to this definition, a dialogue game is in general non-deterministic, in that, for example, given an incoming move of type i, the receiving agent needs to choose amongst n_i possible replies. As proposed in [3,4,5], we combine public dialogue games with private strategies so that agents become deterministic. To this end we introduce the conditions within dialogue games, each associated with a single reply.

Definition 17 (Strategic Dialogue Game) *A strategic dialogue game SDg is a conjunction of rules, specified either as follows:*

$$\bigwedge_{0<j\leq n_i} \big(M_i(\alpha,\beta,a,t) \wedge C_j \Rightarrow M_j(\beta,\alpha,a_j,t')\big)$$

where $t < t'$ and n_i is the number of allowed communicative acts that β could perform after receiving a move of type i from α;
or as follows:

$$\bigwedge_{0<j\leq n} \big(C_j \Rightarrow M_j(\alpha,\beta,a_j,t_0)\big)$$

where t_0 is the initial time and n is the number of allowed moves that α could play initially.

In order to guarantee determinism, conditions C_j need to be mutually exclusive [21]. Agents use their argumentation systems to evaluate, in a private manner, conditions C_j. These argumentation systems are based on the private agents' beliefs and the public commitments recorded in the commitment stores.

To simplify the notations, we omit the time parameter form the moves and use the notation $\cup CS$ as an abbreviation of $CS_\alpha \cup CS_\beta$. In our $\mathcal{B}$usiness-to-$\mathcal{B}$usiness $\mathcal{P}$ersuasive $\mathcal{P}$rotocol ($\mathcal{B}2\mathcal{B}$-$\mathcal{PP}$), agents are not allowed to play the same move (with the same content) more than one time. The strategic dialogue games we consider in this protocol are:

1- Initial game

$$C_{in1} \Rightarrow Assert(\alpha,\beta,a)$$

where:

$$C_{in1} = \exists\, p,q \in \mathcal{WF} : \alpha_p \ncong \beta_q \wedge \mathcal{AR}(\Sigma_\alpha) \rhd a \wedge a \uparrow p$$

The persuasion starts when a conflict is detected and one of the two agents asserts an acceptable argument supporting its position. In the remainder of this section, we suppose that the persuasion topic is represented by the wff p.

2- Assertion game

$$\begin{array}{ll} Assert(\alpha,\beta,\mu) \wedge C_{as1} \Rightarrow Attack(\beta,\alpha,b) & \wedge \\ Assert(\alpha,\beta,\nu) \wedge C_{as2} \Rightarrow Question(\beta,\alpha,x) & \wedge \\ Assert(\alpha,\beta,\nu) \wedge C_{as3} \Rightarrow Accept(\beta,\alpha,a) & \wedge \\ Assert(\alpha,\beta,\nu) \wedge C_{as4} \Rightarrow Stop(\beta,\alpha) & \end{array}$$

where μ is an argument or partial argument, ν is an argument, partial argument, or "?" and:

$$
\begin{array}{ll}
C_{as1} & = Op^{at_1}_{as_1} \vee (\neg Op^{at_1}_{as_1} \wedge Op^{at_2}_{as_1}) \\
Op^{at_1}_{as_1} & = \exists\, b \in A : \mathcal{AR}(\Sigma_\beta \cup CS_\alpha) \rhd b \\
& \quad \wedge\ \Delta(p, \cup CS) \neq \Delta(p, \cup CS \cup \{b\}) \\
Op^{at_2}_{as_1} & = \exists\, b \in A : \mathcal{AR}(\Sigma_\beta \cup CS_\alpha) \trianglerighteq b \\
& \quad \wedge\ \Delta(p, \cup CS) \neq \Delta(p, \cup CS \cup \{b\}) \\
C_{as2} & = \neg C_{as1} \wedge (Op^{qu_1}_{as_2} \vee (\neg Op^{qu_1}_{as_2} \wedge Op^{qu_2}_{as_2})) \\
Op^{qu_1}_{as_2} & = \exists\, b^p_x,\ b^p_x.x \in A : \mathcal{AR}(\Sigma_\beta \cup CS_\alpha) \rhd b^p_x \\
& \quad \wedge\ \Delta(p, \cup CS) \neq \Delta(p, \cup CS \cup \{b^p_x.x\}) \\
Op^{qu_2}_{as_2} & = \exists\, b^p_x,\ b^p_x.x \in A : \mathcal{AR}(\Sigma_\beta \cup CS_\alpha) \trianglerighteq b^p_x \\
& \quad \wedge\ \Delta(p, \cup CS) \neq \Delta(p, \cup CS \cup \{b^p_x.x\}) \\
C_{as3} & = \exists\, a \in A : \mathcal{AR}(\Sigma_\beta \cup CS_\alpha) \rhd a \wedge a \uparrow p \\
& \quad \wedge \neg Op^{qu_1}_{as_2} \wedge \neg Op^{qu_2}_{as_2} \\
C_{as4} & = \neg Op^{at_1}_{as_1} \wedge \neg Op^{qu_1}_{as_2} \wedge \neg Op^{qu_2}_{as_2} \wedge \neg C_{as3} \\
& \quad \wedge \forall\, b \in A,\ \mathcal{AR}(\Sigma_\beta \cup CS_\alpha) \trianglerighteq b \Rightarrow \\
& \qquad\qquad \Delta(p, \cup CS) = \Delta(p, \cup CS \cup \{b\})
\end{array}
$$

In this game, the content of $Assert$ could be an argument, partial argument, or "?". Indeed agents can use this move to assert new arguments in the initial game or to reply to a question in the question game, which is a part of *inquiry* in our protocol. The move that agent β can play as a reply to the $Assert$ move depends on the content of this assertion. When α asserts an argument or a partial argument, CS_α gets changed by adding the advanced (partial) argument. Agent β can attack agent α if β can generate an acceptable argument from its knowledge base and the α's commitment store so that this argument will change the status of the persuasion topic. Consequently, in this protocol agents do not attack only the last advanced argument, but any advanced argument during the interaction, which is still acceptable or (preferred) semi-acceptable ($Op^{at_1}_{as_1}$). This makes the protocol more flexible and efficient (for example agents can try different arguments to attack a given argument). If such an acceptable argument cannot be generated, β will try to generate a (preferred) semi-acceptable argument changing the status of p ($Op^{at_2}_{as_1}$). The idea here is that if β cannot make α's arguments eliminated, it will try to make them (preferred) semi-acceptable. This is due to the following proposition whose proof is straightforward from the definition of semi-acceptable arguments and the fact that only four statuses are possible.

Proposition 6 *If β plays the $Attack$ move with a semi-acceptable argument, then the α's attacked argument changes the status from acceptable to semi-acceptable, and the persuasion topic changes the status from acceptable to semi-acceptable or preferred semi-acceptable.*

We notice that in Assertion game changing the status of p is a result of an attack relation:

Proposition 7 *In Assertion game we have:* $\forall\, b \in A$,
$\Delta(p, \cup CS) \neq \Delta(p, \cup CS \cup \{b\}) \Rightarrow \exists\, a \in \cup CS : \mathcal{AT}(b, a)$.

If β cannot play the $Attack$ move, then before checking the acceptance of an α's argument, it checks if no acceptable and then no (preferred) semi-acceptable argument in the union of the knowledge bases can attack this argument (inquiry part). For that, if β can generate a partial argument changing the status of p, then it will question α about the missing assumptions ($Op^{qu_1}_{as_2}$ and $Op^{qu_2}_{as_2}$). This new feature provides a solution to the "pre-determinism" problem identified in [18]. If such a partial argument does not exist, and if β can generate an acceptable argument supporting p, then it plays the $Accept$ move (C_{as3}).

Proposition 8 *An agent plays the Accept move only if it cannot play the Attack move and cannot play the Question move.*

Proof see Appendix.
Agent β plays the $Stop$ move when it cannot accept an α's argument and cannot attack it. This happens when an agent has a semi-acceptable argument for p and the other a semi-acceptable argument against p, so the status of p in the union of the commitment stores will not change by advancing the β's argument (C_{as4}). Finally, we notice that if the content of $Assert$ move is "?", β cannot play the $Attack$ move. The reason is that such an $Assert$ is played after a question in the Question game, and agents play $Question$ moves only if an attack is not possible. By simple logical calculus, we can prove the following proposition:

Proposition 9 *An agent plays the $Stop$ move iff it cannot play another move.*

3- Attack game

$$
\begin{array}{ll}
Attack(\alpha,\beta,a) \wedge C_{at1} \Rightarrow Attack(\beta,\alpha,b) & \wedge \\
Attack(\alpha,\beta,a) \wedge C_{at2} \Rightarrow Question(\beta,\alpha,x) & \wedge \\
Attack(\alpha,\beta,a) \wedge C_{at3} \Rightarrow Accept(\beta,\alpha,a) & \wedge \\
Attack(\alpha,\beta,a) \wedge C_{at4} \Rightarrow Stop(\beta,\alpha) &
\end{array}
$$

where:

$$
\begin{array}{ll}
C_{at1} & = Op^{at_1}_{at_1} \vee (\neg Op^{at_1}_{at_1} \wedge Op^{at_2}_{at_1}) \\
Op^{at_1}_{at_1} & = Op^{at_1}_{as_1} \\
Op^{at_2}_{at_1} & = Op^{at_2}_{as_1} \\
C_{at2} & = \neg C_{at1} \wedge (Op^{qu_1}_{at_2} \vee (\neg Op^{qu_1}_{at_2} \wedge Op^{qu_2}_{at_2})) \\
Op^{qu_1}_{at_2} & = Op^{qu_1}_{as_2} \\
Op^{qu_2}_{at_2} & = Op^{qu_2}_{as_2} \\
C_{at3} & = \mathcal{AR}(\Sigma_\beta \cup CS_\alpha) \rhd a \wedge \neg Op^{qu_1}_{at_2} \wedge \neg Op^{qu_2}_{at_2} \\
C_{at4} & = \neg Op^{at_1}_{at_1} \wedge \neg Op^{qu_1}_{at_2} \wedge \neg Op^{qu_2}_{at_2} \wedge \neg C_{at3} \\
& \quad \wedge \forall\, b \in A,\ \mathcal{AR}(\Sigma_\beta \cup CS_\alpha) \trianglerighteq b \Rightarrow \\
& \qquad\qquad \Delta(p, \cup CS) = \Delta(p, \cup CS \cup \{b\})
\end{array}
$$

The conditions associated with the Attack game are similar to the ones defining the Assert game. The $Attack$ move also includes the case where the agent that initiates the persuasion puts forward a new argument, which is not attacking any existing argument but changing the status of the persuasion topic. This is useful when the advanced arguments cannot be attacked/defended, so that the agent tries another way to convince the addressee.

4- Question game

$$\begin{array}{ll} Question(\alpha,\beta,x) \wedge C_{qu1} \Rightarrow Assert(\beta,\alpha,a) & \wedge \\ Question(\alpha,\beta,x) \wedge C_{qu2} \Rightarrow Assert(\beta,\alpha,y^p_{x'}) & \wedge \\ Question(\alpha,\beta,x) \wedge C_{qu3} \Rightarrow Assert(\beta,\alpha,?) & \end{array}$$

where:

$$\begin{array}{ll} C_{qu1} & = \exists\, a \in A : \mathcal{AR}(\Sigma_\beta \cup CS_\alpha) \rhd a \wedge (a \uparrow x \vee a \uparrow \bar{x}) \\ C_{qu2} & = \exists\, y^p_{x'}, y^p_{x'}.x' \in A : \mathcal{AR}(\Sigma_\beta \cup CS_\alpha) \rhd y^p_{x'} \\ & \quad \wedge (y^p_{x'} \uparrow x \vee y^p_{x'} \uparrow \bar{x}) \\ C_{qu23} & = \neg C_{qu1} \wedge \neg C_{qu2} \end{array}$$

Agent β can answer the α's question about the content x by asserting an argument for or against x. If not, it answers by a partial argument if it can generate it. Otherwise, it answers by "?" which means that it does not know if x holds or not. We recall that this game is played when an agent has a partial argument and asks the addressee about the missing part, so that the answer could be the complete missing part, a part of it, or nothing.

5- Stop game

$$\begin{array}{ll} Stop(\alpha,\beta) \wedge C_{st1} \Rightarrow Question(\beta,\alpha,x) & \wedge \\ Stop(\alpha,\beta) \wedge C_{st2} \Rightarrow Stop(\beta,\alpha) & \end{array}$$

where:

$$\begin{array}{ll} C_{st1} & = Op^{qu_1}_{st_1} \vee (\neg Op^{qu_1}_{st_1} \wedge Op^{qu_2}_{st_1})) \\ Op^{qu_1}_{st_1} & = Op^{qu_1}_{as_2} \\ Op^{qu_2}_{st_1} & = Op^{qu_2}_{as_2} \\ C_{st2} & = \neg C_{st1} \end{array}$$

Before answering the α's $Stop$ move by another $Stop$ to terminate the persuasion, β checks if no other partial arguments changing the status of p could be generated. Consequently, the $Stop$ move is played only if no such argument could be generated, which means that the conflict cannot be resolved.

3.3. Protocol Analysis

In this section, we prove the termination, soundness, and completeness of $\mathcal{B}2\mathcal{B}$-$\mathcal{PP}$.

Theorem 1 *$\mathcal{B}2\mathcal{B}$-$\mathcal{PP}$ always terminates either successfully by $Accept$ or unsuccessfully by $Stop$.*

Proof see Appendix.
When the protocol terminates, we define its soundness and completeness as follows:

Definition 18 (Soundness - Completeness) *A persuasion protocol about a wff p is sound and complete iff for some arguments a for or against p we have:*
$\mathcal{AR}(\Sigma_\alpha \cup \Sigma_\beta) \rhd a \Leftrightarrow \mathcal{AR}(\cup CS) \rhd a$.

Theorem 2 *The protocol* $(\mathcal{B}2\mathcal{B}\text{-}\mathcal{PP})$ *is sound and complete.*

Proof see Appendix.

4. Case Study

Our running example illustrates a purchase-order scenario (Fig. 4). A customer places an order for products via `Customer-WS` (WS for Web service). Based on this order, `Customer-WS` obtains details on the customer's purchase history from `CRM-WS` (Customer Relationship Management) of $\mathcal{B}$usiness $\mathcal{B}_1$. Afterward, `Customer-WS` forwards these details to $\mathcal{B}_1$'s `Billing-WS`, which calculates the customer's bill based on these details (e.g., considering if the customer is eligible for discounts) and sends the bill to `CRM-WS`. This latter prepares the detailed purchase order based on the bill and sends `Inv-Mgmt-WS` (Inventory Management) of $\mathcal{B}_1$ this order for fulfillment. For those products that are in stock, `Inv-Mgmt-WS` sends `Shipper-WS` of $\mathcal{B}_2$ a shipment request. `Shipper-WS` is now in charge of delivering the products to the customer. For those not in stock, `Inv-Mgmt-WS` sends `Supplier-WS` of $\mathcal{B}_3$ a supply message to the requisite, which provides the products to `Shipper-WS` for subsequent shipment to the customer.

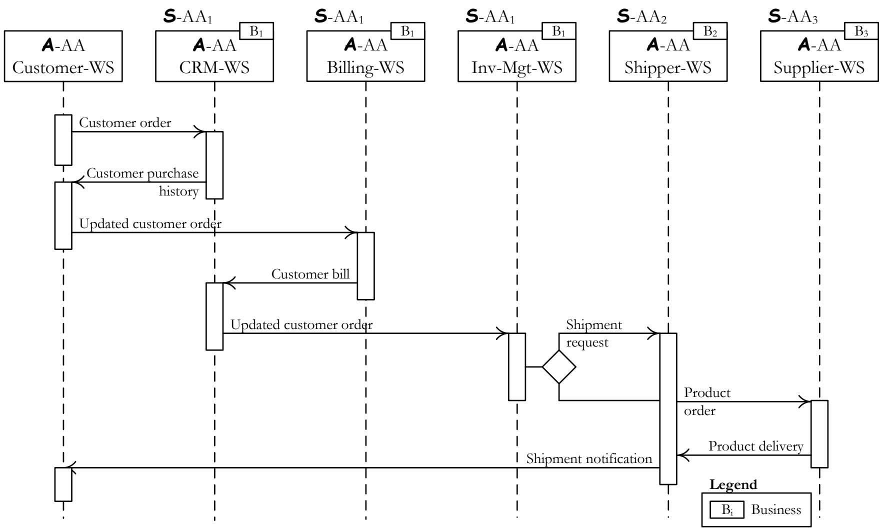

Figure 4. Specification of purchase-order scenario

The above scenario could be affected by several types of conflicts. For example, $\mathcal{B}_2$'s `Shipper-WS` may not deliver the products as agreed with $\mathcal{B}_1$'s `Inv-Mgmt-WS`, perhaps due to lack of trucks. This is an application-level conflict that needs to be resolved using our $\mathcal{B}2\mathcal{B}\text{-}\mathcal{PP}$ by which, `Shipper-WS` tries to persuade `Inv-Mgmt-WS` about the new shipment time and then inform `Customer-WS` of the new delivery time. If not, `Shipper-WS` may change its policies by canceling its partnership agreements without prior notice. This is a strategic-level conflict, that calls for either asking $\mathcal{B}_2$ to which `Shipper-WS` belongs to review its policies, or if that does not work, selecting an alternate shipper.

Let α_{B_1} be the $\mathcal{A}$-AA of `Inv-Mgmt-WS` and β_{B_2} be the $\mathcal{A}$-AA of `Shipper-WS`. The resolution of the application level conflict along with the use of dialogue games are hereafter provided:

1- β_{B_2} identifies the conflict and plays the Initial game by asserting an acceptable argument a about lack of trucks from its $\Sigma_{\beta_{B_2}}$ supporting its position: $Assert(\beta_{B_2}, \alpha_{B_1}, a)$.

2- α_{B_1} has an argument b attacking β_{B_2}'s argument which is about available trucks committed to others that could be used to ship the products. α_{B_1} plays then the Assertion game by advancing the $Attack$ move: $Attack(\alpha_{B_1}, \beta_{B_2}, b)$.

3- β_{B_2} replies by playing the Attack game. Because it does not have an argument to change the status of the persuasion topic, but has a partial argument for that, which is about the high price of these particular trucks that could be not accepted by α_{B_1}, it advances the move: $Question(\beta_{B_2}, \alpha_{B_1}, x)$ where x represents accepting or not the new prices. The idea here is that β_{B_2} can attack α_{B_1}, if it refuses the new prices that others have accepted.

4- α_{B_1} plays the Question game and answers the question by asserting an argument c in favor of the increased shipment charges: $Assert(\alpha_{B_1}, \beta_{B_2}, c)$.

5- β_{B_2} plays the Assertion game, and from $\Sigma_{\beta_{B_2}} \cup CS_{\alpha_{B_1}}$, it accepts the argument and agrees to deliver the products as per the agreed schedule with the new price, which is represented by d: $Accept(\beta_{B_2}, \alpha_{B_1}, d)$. Consequently, the persuasion terminates successfully by resolving the conflict.

Figs. 5 and 6 illustrate the scenario details with the exchanged arguments. We have implemented a proof-of-concept prototype of this scenario using the Jadex Agent System. Fig. 7 depicts a screenshot of the prototype illustrating the computation of the arguments in the scenario. Currently the prototype only demonstrates the case of horizontal coordination described above. Future extensions would include other scenarios as well as vertical coordination.

5. Related Work

Recent years have seen a continuing surge of interest in designing and deploying $\mathcal{B}2\mathcal{B}$ applications. Service-oriented architecture is the most widely methodology that have been used in this field [13,17,22,24]. In [13] the author proposes the exploitation of Web services and intelligent agent techniques for the design and development of a $\mathcal{B}2\mathcal{B}$ eCommerce application. A multi-party multi-issue negotiation mechanism is developed for this application. This negotiation is a Pareto optimal negotiation based on game theory. This proposal aims at achieving an agrement by computing concessions and generating offers in order to maximize the utility of the participating agents. However, unlike our argumentation-based framework, this mechanism cannot be used to resolve general conflicts as those discussed in Sections 1 and 2. In [17], the authors develop a methodology for $\mathcal{B}2\mathcal{B}$ design applications using Web-based data integration. The aim is the creation of adaptable semantics oriented meta-models to facilitate the design of mediators by considering several characteristics of interoperable information systems such as extensibility and composability. The methodology is used to build cooperative environments involv-

1- Conflict detection by A-AA of Shipper-WS after the request of the product P1 with normal delivery time from A-AA of Inv-Mgmt-WS. There is a conflict because A-AA of Inv-Mgmt-WS has an acceptable argument from its knowledge base for normal delivery time which is:

$$p, p \rightarrow q$$

where

p = "past agreement", and q = "normal delivery time of P1"

And A-AA of Shipper-WS has an acceptable argument $"a"$ for delayed delivery of P1 which is:

$$a = r, r \rightarrow s$$

Where r = "luck of trucks to ship product P1", and s = "delayed delivery of P1"

Here q and s are contradictory, hence the conflict. The formula s represents the conflict topic.

A-AA of Shipper-WS plays the Initial game by asserting his acceptable argument $"a"$ about lack of trucks

2- A-AA of Inv-Mgmt-WS has an argument $"b"$ in its knowledge base attacking A-AA of Shipper-WS's argument which is:

$$b = t_1, t_2, t_1 \wedge t_2 \rightarrow u$$

where t_1 = "some trucks *tr* committed to another businesses *BS*", t_2 = "trucks *tr* could be used to ship the product P1", and u = "available trucks to ship product P1"

Here u and r are contradictory. Inv-Mgmt Agent plays then the Assertion game by advancing the Attack move with the argument $"b"$.

3- At this stage, A-AA of Shipper-WS cannot change the state of the conflict topic by attacking the Inv-Mgmt Agent's argument. However, it has a partial argument for that, which is about the high price of these particular trucks that could be not accepted by Inv-Mgmt Agent. The partial argument is:

$$m, m \wedge x \rightarrow r$$

where m = "price of trucks *tr* is *pr*", x = "Inv-Mgmt Agent's not accept price *pr*", and r = "luck of trucks to ship product P1". This is a partial argument because it needs x to be an argument. For that, A-AA of Shipper-WS plays the Attack game with the *Question* move about x.

4- Inv-Mgmt Agent's has an argument from its knowledge base against x. It plays the Question game and answers the *Question* move by asserting an argument $"c"$ in favor of the increased shipment charges. This argument is:

$$k, k \rightarrow l$$

Where k = "*pr* is less than Max", and l = "accept *pr*"

(l and x are contradictory)

5- From the Inv-Mgmt Agent's commitment store and the A-AA of Shipper-WS's knowledge base, this latter plays an *Accept* move in which it accepts to deliver the product P1 with normal delivery time and the new price *pr*.

Figure 5. Scenario description

ing the integration of Web data and services. Unlike our methodology, this proposal does not consider conflicts that can arise during the cooperation phase and only addresses the cooperation from technological point of view.

On the other side, and from an argumentation viewpoint, some interesting protocols for persuasion and inquiry have been proposed. [2] propose Persuasive Argument

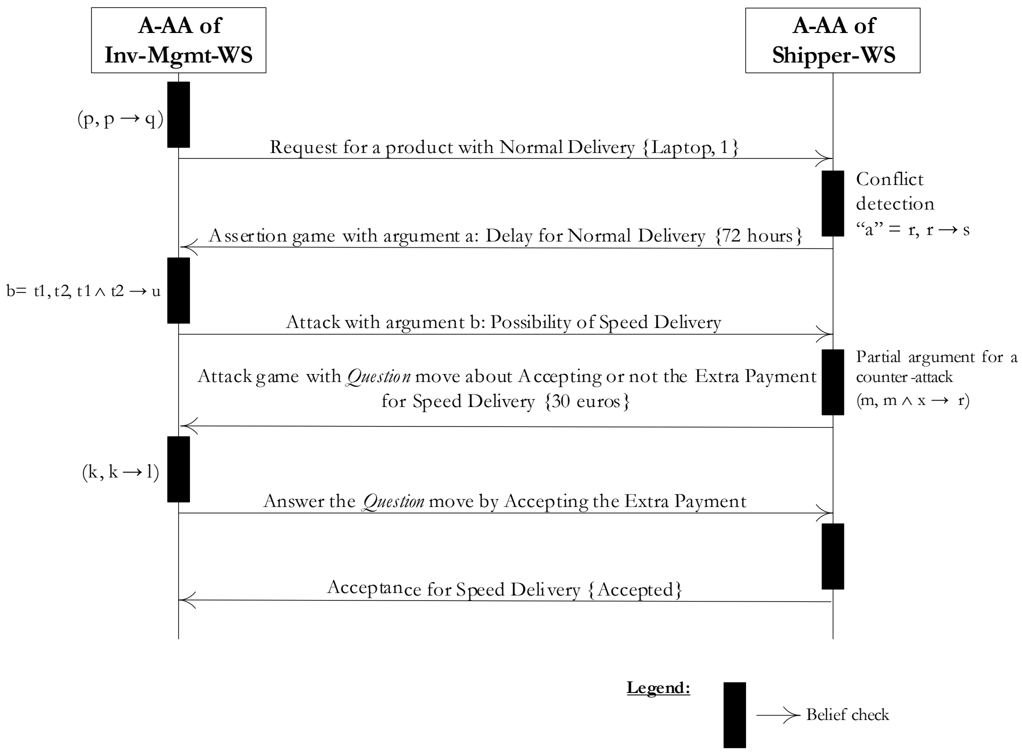

Figure 6. Sample of interaction between $\mathcal{A}$-AA of Inv-Mgmt-WS and $\mathcal{A}$-AA of Shipper-WS

for Multiple Agents (PARMA) Protocol, which enables participants to propose, attack, and defend an action or course of actions. This protocol is specified using logical consequence and denotational semantics. The focus of this work is more on the semantics of the protocol rather than the dynamics of interactions. [5] propose a dialogue-game inquiry protocol that allows two agents to share knowledge in order to construct an argument for a specific claim. There are many fundamental differences between this protocol and ours. Inquiry and persuasion settings are completely different since the objectives and dynamics of the two dialogues are different. In [5], argumentation is captured only by the notion of argument with no attack relation between arguments. This is because agents collaborate to establish joint proofs. However, in our system, agents can reason about conflicting assumptions, and they should compute different acceptability semantics, not only to win the dispute, but also to reason internally in order to remove inconsistencies from their assumptions. From the specification perspective, there are no similarities between the two protocols. Our protocol is specified as a set of rules about which agents can reason using argumentation, which captures the agents' choices and strategies. However, in [5] the protocol is specified in a declarative manner and the strategy is only defined as a function without specifying how the agents can use it. The adopted moves in the two proposals are also different. Another technical, but fundamental difference in the two protocols is the possibility in our protocol of considering not only the

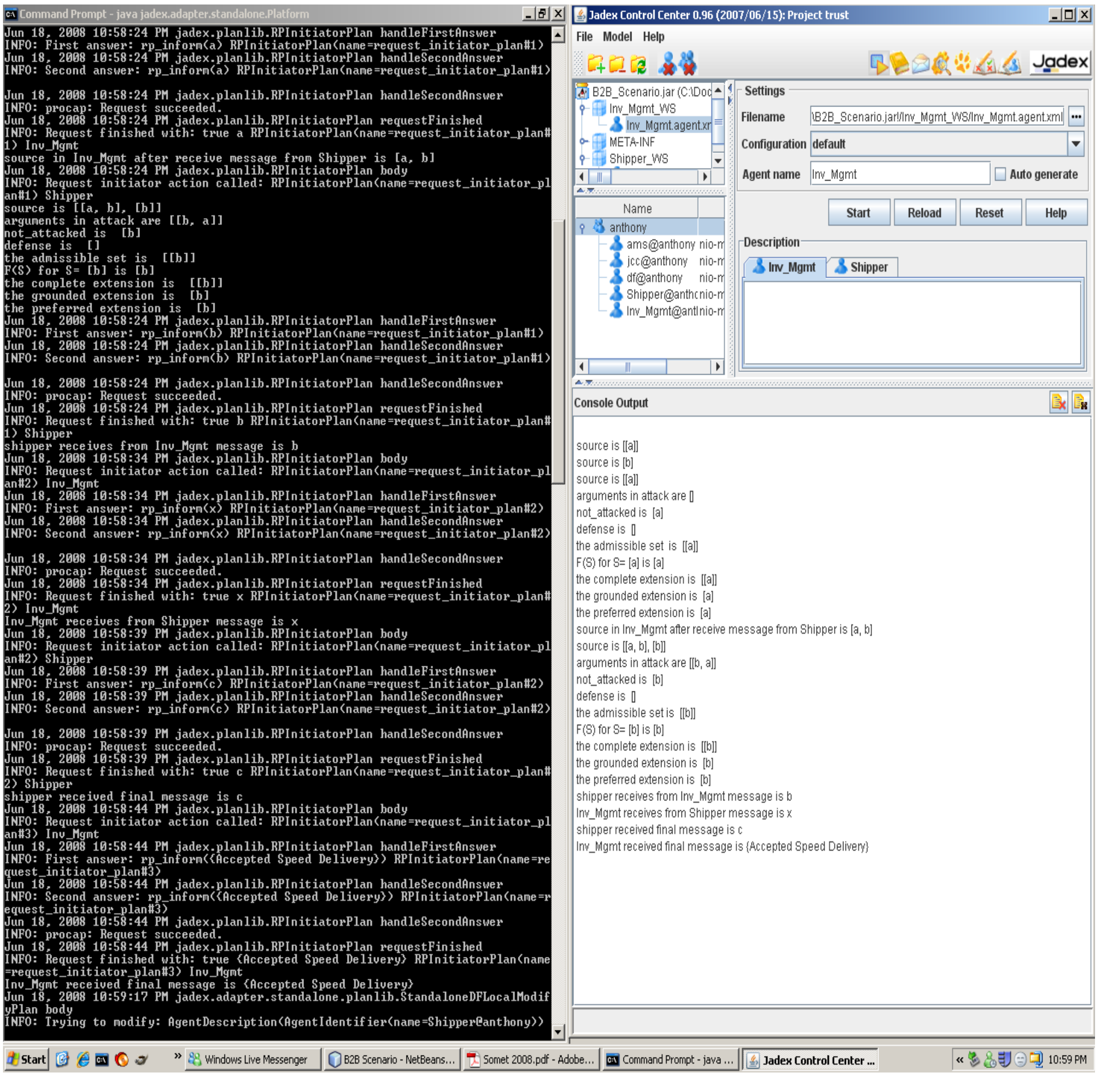

Figure 7. A screenshot from the prototype -computing arguments-

last uttered argument, but any previous argument which allows agents to consider and try different ways of attacking each other.

6. Conclusions and Future Work

In this paper, a 3-level framework for $\mathcal{B}2\mathcal{B}$ applications was presented. The three levels namely strategic, application and resource where populated with argumentative agents. We have shown how our framework can be used to set up collaborations among autonomous businesses (via strategic level), and execute and manage these collaborations (via application level). Inevitably, given the autonomous nature of businesses, conflicts are bound to arise. We have shown how our framework can detect and resolve conflicts. To this end an argumentation-based model was developed. This model was the basis of a persuasion protocol that includes inquiry stages for resolving conflicts between agents acting on behalf of applications of type Web services.

Future work would involve adding a negotiation protocol and scaling up and demonstrating our argumentation-based model on larger examples. Additionally, we will be enriching our model with contextual ontologies when modeling knowledge bases of individual agents.

References

[1] L. Amgoud, Y. Dimopoulos, and P. Moraitis. A Unified and Genaral Framwork for Argumentation-based Negotiation. In *The International Conference on Autonomous Agents and Multiagent Systems*. ACM Press, 2007.

[2] K. Atkinson, T. Bench-Capon, and P. McBurney. A Dialogue Game Protocol for Multi-Agent Argument over Proposals for Action. *Journal of Autonomous Agents and Multi-Agent Systems*, 11(2), 2005.

[3] J. Bentahar, Z. Maamar, D. Benslimane, and P. Thiran. An Argumentation Framework for Communities of Web Services. *IEEE Intelligent Systems*, 22(6), 2007.

[4] J. Bentahar, F. Toni, J.-J. Meyer, and J. Labban. A Security Framework for Agent-based Systems. *Journal of Web Information Systems*, 3(4):341–362, 2007.

[5] E. Black and A. Hunter. A Generative Inquiry Dialogue System. In *The International Conference on Autonomous Agents and Multiagent Systems*. ACM Press, 2007.

[6] A. Bondarenko, P. Dung, R. Kowalski, and F. Toni. An Abstract, Argumentation-Theoretic Approach to Default Reasoning. *Artificial Intelligence*, 93(1–2):63–101, 1997.

[7] P. Dung. The Acceptability of Arguments and its Fundamental Role in Non-Monotonic Reasoning and Logic Programming and n-Person Game. *Artificial Intelligence*, 77:321–357, 1995.

[8] P. Dung, P. Mancarella, and F. Toni. Computing ideal sceptical argumentation. *Artificial Intelligence, Special Issue on Argumentation in Artificial Intelligence*, 171(10-15):642–674, 2007.

[9] H. Fujita. Special Issue on "Techniques to Produce Intelligent_Secure Software". *Knowl.-Based Syst.*, 20(7):614–616, 2007.

[10] A. Garcia and G. Simari. Defeasible Logic Programming: an Argumentative Approach. *Theory and Practice of Logic Programming*, 4(1):95–138, 2004.

[11] V. Gruhn and H. Fujita. Special Issue on "Intelligent Software Design". *Knowl.-Based Syst.*, 19(2):105–106, 2006.

[12] B. Ktari, H. Fujita, M. Mejri, and D. Godbout. Toward a New Software Development Environment. *Knowl.-Based Syst.*, 20(7):683–693, 2007.

[13] R. Lau. Towards a Web Services and Intelligent Agents-based Negotiation System for B2B eCommerce. *Electronic Commerce Research and Appl.*, 6(3), 2007.

[14] N. Maudet and B. Chaib-draa. Commitment-based and Dialogue Game-based Protocols, new trends in agent communication languages. *Knowledge Engineering Review*, 17(2):157–179, 2002.

[15] P. McBurney and S. Parsons. Games that Agents Play: A Formal Framework for Dialogues between Autonomous Agents. *Journal of Logic, Language, and Information*, 11(3):315–334, 2002.

[16] N. Milanovic and M. Malek. Current Solutions for Web Service Composition. *IEEE Internet Computing*, 8(6), November/December 2004.

[17] C. Nicolle, K. Yétongnon, and J. Simon. XML Integration and Toolkit for B2B Applications. *Journal of Database Management*, 14(4), 2003.

[18] S. Parsons, M. Wooldridge, and L. Amgoud. On the Outcomes of Formal Inter-Agent Dialogues. In *Proceedings of The International Conference on Autonomous Agents and Multiagent Systems*. ACM Press, 2003.

[19] P. Pasquier, I. Rahwan, F. Dignum, and L. Sonenberg. Argumentation and Persuasion in the Cognitive Coherence Theory. In *The 1st International Conference on Computational Models of Argument*. IOS Press, 2006.

[20] H. Prakken. Formal Systems for Persuasion Dialogue. *The Knowledge Engineering Review*, 24(2), 2005.
[21] F. Sadri, F. Toni, and P. Torroni. Dialogues for Negotiation: Agent Varieties and Dialogue Sequences. In *The International Workshop on Agents, Theories, Architectures and Languages*, 2001.
[22] Y. Udupi and M. Singh. Contract Enactment in Virtual Organizations: A Commitment-Based Approach. In *The 21st National Conference on Artificial Intelligence*, 2006.
[23] D. N. Walton and E. C. W. Krabbe. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning*. Suny Series in Logic and Language, 1995.
[24] A. Williams, A. Padmanabhan, and B. Blake. Local Consensus Ontologies for B2B-Oriented Service Composition. In *The International Joint Conference on Autonomous Agents and Multiagent Systems*, 2003.

Appendix

Proof of Proposition 2 Without loss of generality, let $a, a_1, \ldots, a_n$ be arguments in a given argumentation framework such that $a_1, \ldots, a_n$ are the only attackers of a and a is the only attacker of these arguments. According to Definition 5, the argument a is not acceptable since it is attacked and not defended, directly or indirectly by a non-attacked argument. Because it is defended, a belongs to some preferred extensions. However, a does not belong to all of them. For example, a does not belong to the preferred extension to which the arguments $a_1, \ldots, a_n$ belong since these arguments belong also to some preferred extensions because they are defended. a is then semi-acceptable. $\Box$

Proof of Proposition 3 We prove this proposition by a counter example using Example 4. In this example $\{a, d\}$ and $\{b, d\}$ are complete extensions (preferred extensions). However, $\{d\}$ is not a complete extension. $\Box$

Proof of Proposition 4 By Definition 5, the grounded extension is included in all preferred extensions. Consequently, using definition 4, an eliminated argument is not acceptable. Also, according to Definition 7, an eliminated argument is not semi-acceptable and not preferred semi-acceptable. $\Box$

Proof of Proposition 5 Suppose that $\exists x \in \mathcal{WF} : a_x^p$ is not acceptable. Therefore, a part of the non-missing part of a_x^p is not acceptable. Because this part is also a part of a, then a is not acceptable. Contradiction! $\Box$

Proof of Proposition 8 To prove this we should prove that $C_{as3} \Rightarrow \neg C_{as1} \wedge \neg C_{as2}$. Using the logical calculation, we can easily prove that $\neg C_{as1} \wedge \neg C_{as2} = \neg C_{as1} \wedge \neg Op_{as_2}^{qu_1} \wedge \neg Op_{as_2}^{qu_2}$. Also, if an agent β can build an acceptable argument a from $\mathcal{A}_\beta \cup CS_\alpha$, then it cannot build an acceptable or (preferred) semi-acceptable argument attacking a from the same set. Therefore, $\mathcal{AR}(\mathcal{A}_\beta \cup CS_\alpha) \rhd a \Rightarrow \neg C_{as1}$. Thus the result follows. $\Box$

Proof of Theorem 1 Agents' knowledge bases are finite and repeating moves with the same content is prohibited. Consequently, the number of $Attack$ and $Question$ moves that agents can play is finite. At a given moment, agents will have two possibilities only: $Accept$ if an acceptable argument can be built from $CS_\alpha \cup CS_\beta$, or $Stop$, otherwise. Therefore, the protocol terminates successfully by $Accept$, or unsuccessfully by $Stop$ when $Accept$ move cannot be played, which means that only semi-acceptable arguments are included in $CS_\alpha \cup CS_\beta$. $\Box$

Proof of Theorem 2 For simplicity and without loss of generality, we suppose that agent α starts the persuasion.
Let us first prove the $\Rightarrow$ direction: $\mathcal{AR}(\mathcal{A}_\alpha \cup \mathcal{A}_\beta) \rhd a \Rightarrow \mathcal{AR}(\cup CS) \rhd a$.
In the protocol, the persuasion starts when a conflict over p occurs. Consequently, the case where $\mathcal{A}_\alpha \rhd a$ and $\mathcal{A}_\beta \rhd a$ does not hold. The possible cases are limited to three:

1. $\mathcal{A}_\alpha \rhd a$ and $\mathcal{A}_\beta \not\rhd a$. In this case, agent α starts the persuasion over p by asserting a. Agent β can either play the $Attack$ move or the $Question$ move. Because $\mathcal{AR}(\mathcal{A}_\alpha \cup \mathcal{A}_\beta \rhd a)$ all the β's arguments will be counter-attacked. For the same reason, β cannot play the $Stop$ move. Consequently, at the end, β will play an $Accept$ move. It follows that $\mathcal{AR}(\cup CS \rhd a)$.
2. $\mathcal{A}_\alpha \not\rhd a$ and $\mathcal{A}_\beta \rhd a$. In this case, agent α starts the persuasion by asserting an acceptable argument b in its knowledge base against p ($\mathcal{A}_\alpha \rhd b$). This argument will be attacked by agent β, and the rest is identical to case 1 by substituting agent roles.
3. $\mathcal{A}_\alpha \not\rhd a$ and $\mathcal{A}_\beta \not\rhd a$. To construct argument a out of $\mathcal{A}_\alpha \cup \mathcal{A}_\beta$, two cases are possible. Either, (1) agent α has an acceptable partial argument a^Y_∂ for p and agent β has the missing assumptions (or some parts of the missing assumptions, and agent α has the other parts), or (2) the opposite (i.e., agent β has an acceptable partial argument a^Y_∂ for p and agent α has the missing assumptions (or some parts of the missing assumptions, and agent β has the other parts)). Only the second case is possible since the first one is excluded by hypothesis. For simplicity, we suppose that agent α has all the missing assumptions, otherwise the missing assumptions will be built by exchanging the different partial arguments. Agent α starts the persuasion by asserting an acceptable argument b in its knowledge base against p. Agent β can either play an $Attack$ or a $Question$ move. If attack is possible, then agent α can either counter-attack or play the $Stop$ move. The same scenario continues until agent α plays $Stop$, and then agent β plays a $Question$ Move. Agent α answers now the question by providing the missing assumptions, after which agent β attacks and agent α can only accept since $\mathcal{AR}(\mathcal{A}_\alpha \cup \mathcal{A}_\beta \rhd a)$. It follows that $\mathcal{AR}(\cup CS \rhd a)$.

Let us now prove the $\Leftarrow$ direction: $\mathcal{AR}(\cup CS) \rhd a \Rightarrow \mathcal{AR}(\mathcal{A}_\alpha \cup \mathcal{A}_\beta) \rhd a$.
In the protocol, to have $\mathcal{AR}(\cup CS) \rhd a$ one of the two agents, say agent α, puts forward the argument a and the other, agent β, accepts it. On the one hand, to advance an argument, agent α plays the $Assert$ move (in the initial or question rules) or $Attack$ move (in the assertion or attack rules). In all these cases, we have: $\mathcal{AR}(\mathcal{A}_\alpha \cup CS_\beta) \rhd a$ and there is no partial acceptable argument attacking a from $\mathcal{A}_\alpha \cup SC_\beta$. On the other hand, to accept an argument (in the assertion or attack rules), agent β should check that $\mathcal{AR}(\mathcal{A}_\beta \cup CS_\alpha) \rhd a$, there is no other arguments changing the status of the persuasion topic, and there is no partial acceptable argument attacking a from $\mathcal{A}_\beta \cup SC_\alpha$. Therefore we obtain: $\mathcal{AR}(\mathcal{A}_\alpha \cup CS_\beta \cup \mathcal{A}_\beta \cup CS_\alpha) \rhd a$. Because $CS_\alpha \subseteq \mathcal{A}_\alpha$ and $CS_\beta \subseteq \mathcal{A}_\beta$ we are done. $\Box$