Computer Science

Based on:
FIPA. FIPA Communicative Act Library Specification. Foundation for Intelligent Physical Agents, Document number SC00037J, Document source FIPA TC Communication. 2002.

Agent Communications
FIPA Performative Semantics

CPSC 662/CPSC 568
Rob Kremer
Department of Computer Science
University of Calgary

Inform

Summary	The sender informs the receiver that a given proposition is true.
Content	A proposition.
Description	The sender intends that the receiver believe the proposition, the the receiver might not.
Formal Model	$<i, inform\ (j, \varphi)>$ FP: $B_i\varphi \wedge \neg B_i(Bif_j\varphi \vee Uif_j\varphi)$ RE: $B_j\varphi$
Examples	Agent i informs agent j that (it is true that) it is raining today.

```
(inform
        :sender (agent-identifier :name i)
        :receiver (set (agent-identifier :name j))
        :content  "weather (today, raining)"
        :language Prolog)
```

Now that we've tried to convince the receiver of something, we might wonder if he really believes it or not….

Query-if

Summary	The action of asking another agent whether or not a given proposition is true.
Content	A proposition.
Description	Might get back a refuse.
Formal Model	$<i, query\text{-}if\ (j, \varphi)> \equiv$ $<i, request\ (j, <j, inform\text{-}if\ (i, \varphi)>)>$ FP: $\neg Bif_i\varphi \wedge \neg Uif_i\varphi \wedge \neg B_i\ I_j\ Done(<j, inform\text{-}if\ (i, \varphi)>)$ RE: $Done\ (<j, inform(i, \varphi)> \mid <j, inform\ (i, \neg\varphi)>)$
Examples	Agent i asks agent j if j is registered with domain server d1. Agent j replies that it is not.

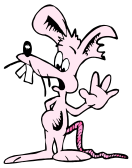
$<j, inform\text{-}if\ (i, \varphi)> \equiv$
$<j, inform\ (i, \varphi)> \mid <j, inform\ (i, \neg\varphi)>$

```
(query-if
 :sender (agent-identifier :name i)
 :receiver (set(agent-identititfier :name j))
 :content
   "((registered (server d1) (agent j)))"
 :reply-with r09
…)
```

```
(inform
 :sender (agent-identifier :name j)
 :receiver (set (agent-identifier :name i))
 :content
   "((not (registered (server d1) (agent j))))"
 :in-reply-to r09)
```

But if the agent doesn't know about the proposition, it can't do an inform…

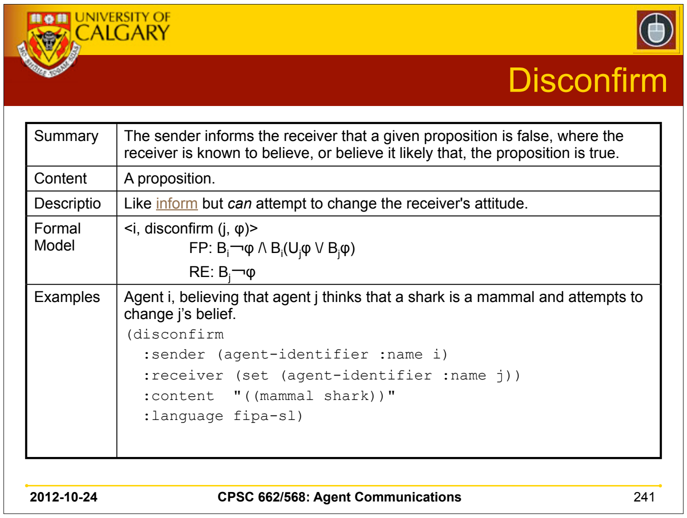
So it refuses.



Refuse

Summary	The action of refusing to perform a given action, and explaining the reason for the refusal.
Content	A tuple, consisting of an action expression and a proposition giving the reason for the refusal.
Description	
Formal Model	$<i, refuse\ (j, <i, act>, \varphi)> \equiv$ $<i, disconfirm\ (j, Feasible(<i,act>))>;<i, inform\ (j, \varphi \wedge \neg Done\ (<i,act>) \wedge \neg I_i\ Done\ (<i,act>))>$ FP: $B_i\ \neg Feasible\ (<i, act>) \wedge B_i\ (B_j\ Feasible\ (<i, act>) \vee U_j\ Feasible\ (<i, act>)) \wedge B_i\ \alpha \wedge \neg B_i\ (Bif_j\ \alpha \vee Uif_j\ \alpha)$ RE: $B_j\ \neg Feasible\ (<i, act>) \wedge B_j\ \alpha$ Where: $\alpha = \varphi \wedge \neg Done\ (<i, act>) \wedge \neg I_i\ Done\ (<i, act>)$ Agent i informs j that action act is not feasible, and further that, because of proposition φ, act has not been done and i has no intention to do act.
Examples	Agent j refuses to i reserve a ticket for i, since there are insufficient funds in i's account.

```
(refuse
 :sender (agent-identifier :name j)
 :receiver (set (agent-identifier :name i))
 :content
  "((action (agent-identifier :name j)
     (reserve-ticket LHR MUC 27-sept-97))
    (insufficient-funds ac12345))"
 :language fipa-sl)
```

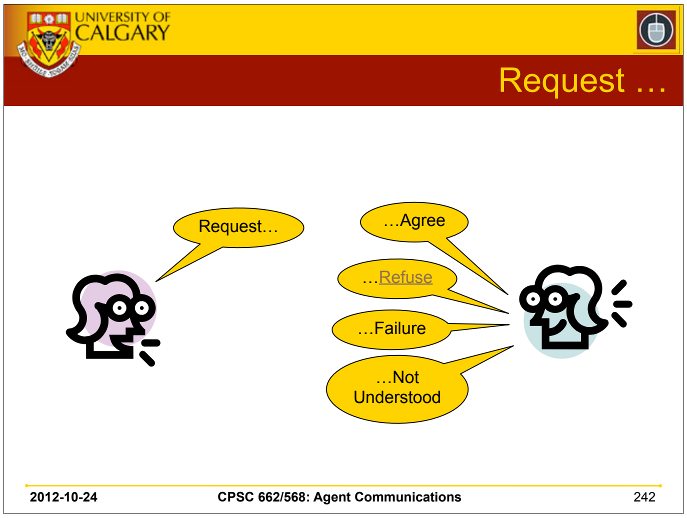


UNIVERSITY OF CALGARY

Disconfirm

Summary	The sender informs the receiver that a given proposition is false, where the receiver is known to believe, or believe it likely that, the proposition is true.
Content	A proposition.
Descriptio	Like inform but *can* attempt to change the receiver's attitude.
Formal Model	$\langle i, disconfirm\ (j, \varphi)\rangle$ FP: $B_i \neg\varphi \land B_i(U_j\varphi \lor B_j\varphi)$ RE: $B_j \neg\varphi$
Examples	Agent i, believing that agent j thinks that a shark is a mammal and attempts to change j's belief.

```
(disconfirm
  :sender (agent-identifier :name i)
  :receiver (set (agent-identifier :name j))
  :content  "((mammal shark))"
  :language fipa-sl)
```



UNIVERSITY OF CALGARY

Request …

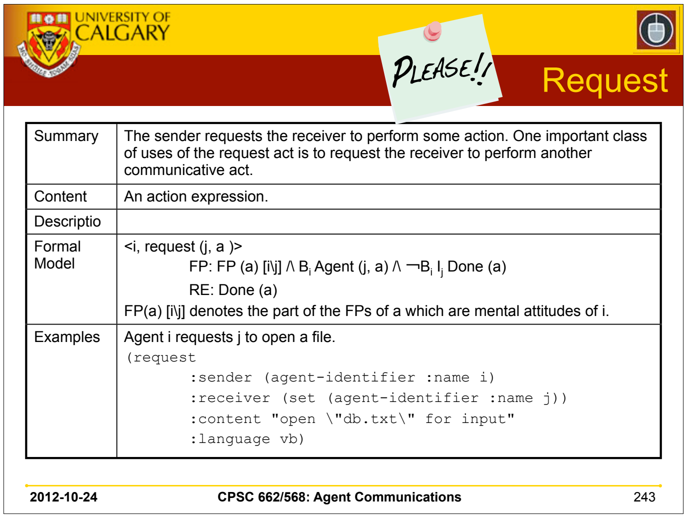
Request…

…Agree

…Refuse

…Failure

…Not Understood



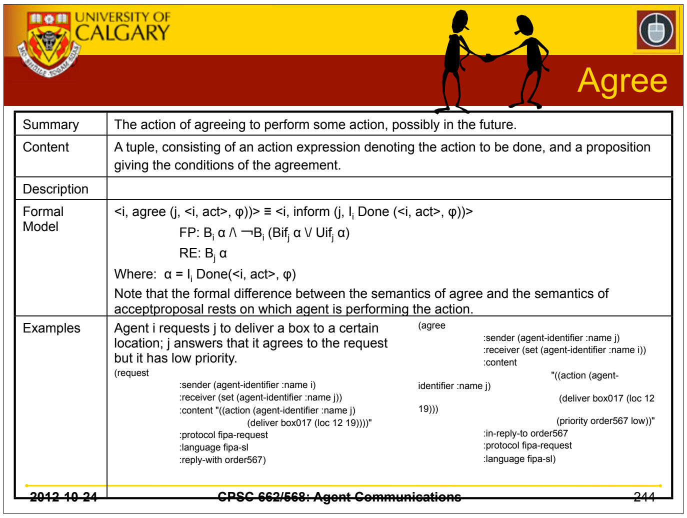
UNIVERSITY OF CALGARY

PLEASE!!

Request

Summary	The sender requests the receiver to perform some action. One important class of uses of the request act is to request the receiver to perform another communicative act.
Content	An action expression.
Descriptio	
Formal Model	$\langle i, request\ (j, a)\rangle$ FP: FP (a) [i\j] $\land B_i$ Agent (j, a) $\land \neg B_i I_j$ Done (a) RE: Done (a) FP(a) [i\j] denotes the part of the FPs of a which are mental attitudes of i.
Examples	Agent i requests j to open a file.

```
(request
    :sender (agent-identifier :name i)
    :receiver (set (agent-identifier :name j))
    :content "open \"db.txt\" for input"
    :language vb)
```



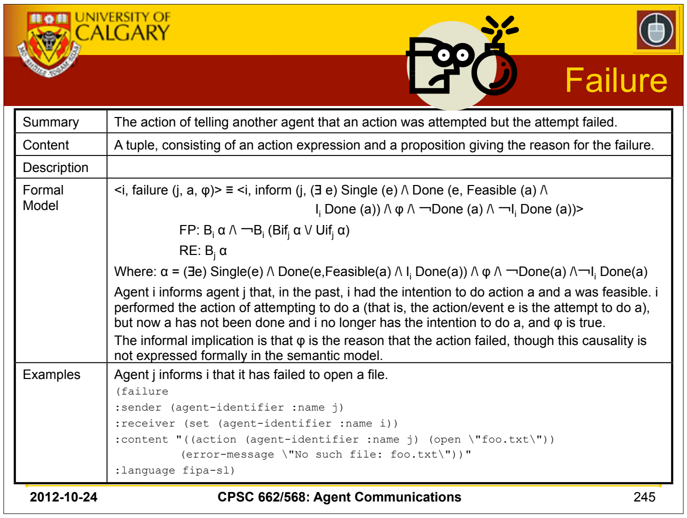
UNIVERSITY OF CALGARY

Agree

Summary	The action of agreeing to perform some action, possibly in the future.
Content	A tuple, consisting of an action expression denoting the action to be done, and a proposition giving the conditions of the agreement.
Description	
Formal Model	$\langle i, agree\ (j, \langle i, act\rangle, \varphi))\rangle \equiv \langle i, inform\ (j, I_i\ Done\ (\langle i, act\rangle, \varphi))\rangle$ FP: $B_i\ \alpha \land \neg B_i\ (Bif_j\ \alpha \lor Uif_j\ \alpha)$ RE: $B_j\ \alpha$ Where: $\alpha = I_i\ Done(\langle i, act\rangle, \varphi)$ Note that the formal difference between the semantics of agree and the semantics of acceptproposal rests on which agent is performing the action.
Examples	Agent i requests j to deliver a box to a certain location; j answers that it agrees to the request but it has low priority.

```
(request
        :sender (agent-identifier :name i)
        :receiver (set (agent-identifier :name j))
        :content "((action (agent-identifier :name j)
                (deliver box017 (loc 12 19))))"
        :protocol fipa-request
        :language fipa-sl
        :reply-with order567)
```

```
(agree
        :sender (agent-identifier :name j)
        :receiver (set (agent-identifier :name i))
        :content
                "((action (agent-
identifier :name j)
                (deliver box017 (loc 12
19)))
                (priority order567 low))"
        :in-reply-to order567
        :protocol fipa-request
        :language fipa-sl)
```

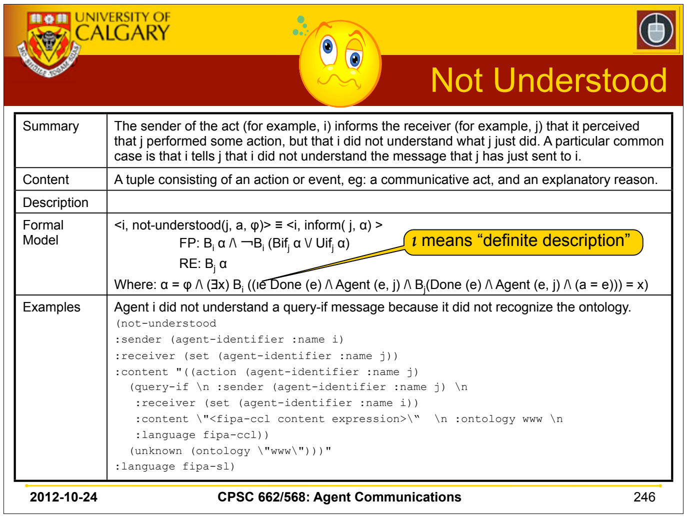


UNIVERSITY OF CALGARY

Failure

Summary	The action of telling another agent that an action was attempted but the attempt failed.
Content	A tuple, consisting of an action expression and a proposition giving the reason for the failure.
Description	
Formal Model	$\langle i, failure\ (j, a, \varphi)\rangle \equiv \langle i, inform\ (j, (\exists\ e)\ Single\ (e) \land Done\ (e, Feasible\ (a) \land I_i\ Done\ (a)) \land \varphi \land \neg Done\ (a) \land \neg I_i\ Done\ (a))\rangle$ FP: $B_i\ \alpha \land \neg B_i\ (Bif_j\ \alpha \lor Uif_j\ \alpha)$ RE: $B_j\ \alpha$ Where: $\alpha = (\exists e)\ Single(e) \land Done(e,Feasible(a) \land I_i\ Done(a)) \land \varphi \land \neg Done(a) \land \neg I_i\ Done(a)$ Agent i informs agent j that, in the past, i had the intention to do action a and a was feasible. i performed the action of attempting to do a (that is, the action/event e is the attempt to do a), but now a has not been done and i no longer has the intention to do a, and φ is true. The informal implication is that φ is the reason that the action failed, though this causality is not expressed formally in the semantic model.
Examples	Agent j informs i that it has failed to open a file.

```
(failure
:sender (agent-identifier :name j)
:receiver (set (agent-identifier :name i))
:content "((action (agent-identifier :name j) (open \"foo.txt\"))
         (error-message \"No such file: foo.txt\"))"
:language fipa-sl)
```



UNIVERSITY OF CALGARY

Not Understood

Summary	The sender of the act (for example, i) informs the receiver (for example, j) that it perceived that j performed some action, but that i did not understand what j just did. A particular common case is that i tells j that i did not understand the message that j has just sent to i.
Content	A tuple consisting of an action or event, eg: a communicative act, and an explanatory reason.
Description	
Formal Model	$\langle i, not\text{-}understood(j, a, \varphi)\rangle \equiv \langle i, inform(j, \alpha) \rangle$ FP: $B_i\ \alpha \land \neg B_i\ (Bif_j\ \alpha \lor Uif_j\ \alpha)$ RE: $B_j\ \alpha$ Where: $\alpha = \varphi \land (\exists x)\ B_i\ ((\iota e\ Done\ (e) \land Agent\ (e, j) \land B_j(Done\ (e) \land Agent\ (e, j) \land (a = e))) = x)$ ι means "definite description"
Examples	Agent i did not understand a query-if message because it did not recognize the ontology.

```
(not-understood
:sender (agent-identifier :name i)
:receiver (set (agent-identifier :name j))
:content "((action (agent-identifier :name j)
  (query-if \n :sender (agent-identifier :name j) \n
   :receiver (set (agent-identifier :name i))
   :content \"<fipa-ccl content expression>\"  \n :ontology www \n
   :language fipa-ccl))
  (unknown (ontology \"www\")))"
:language fipa-sl)
```

But what if an agent agrees to do something, then discovers it can't carry it out?

- **Cancel**: no longer intends to carry out the action
- **Failure**: cancel + the action was partially done

Cancel

Summary	The action of one agent informing another agent that the first agent no longer has the intention that the second agent perform some action.
Content	An action expression denoting the action that is no longer intended.
Description	If the creditor wants to cancel, he should send a request to the debtor to stop the action.
Formal Model	<i, cancel (j, a)> ≡ <i, disconfirm (j, Ii Done (a))> // In fact, this doesn't always capture the intent… FP: $\neg I_i$ Done (a) $\wedge B_i (B_j I_i$ Done (a) $\vee U_j I_i$ Done (a)) RE: $B_j \neg I_i$ Done (a) cancel applies to any form of request action. Suppose an agent i has requested an agent j to perform some action a, possibly if some condition holds. This request has the effect of i informing j that i has an intention that j perform the action a. When i comes to drop its intention, it can inform j that it no longer has this intention with a disconfirm.
Examples	Agent j asks i to cancel a previous request-whenever act by quoting the action.

```
(cancel
 :sender (agent-identifier :name j)
 :receiver (set(agent-identifier :name i))
 :content
   "((action (agent-identifier :name j)
     (request-whenever
 :sender (agent-identifier :name j)
 :receiver (set(agent-identifier :name i))
```

```
:content
  \"((action (agent-identifier :name i)
     (inform-ref
      :sender (agent-identifier :name i)
      :receiver (set
         (agent-identifier  :name j))
      :content
       \"((iota ?x (=(price widget) ?x))\")
          (> (price widget) 50))"
       …)))"
:langage fipa-sl  …)
```

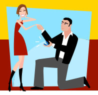
Failure

Summary	The action of telling another agent that an action was attempted but the attempt failed.
Content	A tuple, consisting of an action expression and a proposition giving the reason for the failure.
Description	
Formal Model	<i, failure (j, a, φ)> ≡ <i, inform (j,(∃e)Single(e) $\wedge$ Done(e,Feasible(a) $\wedge I_i$ Done(a)) $\wedge \varphi \wedge \neg$Done(a) $\wedge \neg I_i$ Done(a))> FP: $B_i \alpha \wedge \neg B_i (Bif_j \alpha \vee Uif_j \alpha)$ RE: $B_j \alpha$ Where: α = (∃e)Single(e) $\wedge$ Done(e,Feasible(a) $\wedge I_i$ Done(a)) $\wedge \varphi \wedge \neg$Done(a) $\wedge \neg I_i$ Done(a) Agent i informs agent j that, in the past, i had the intention to do action a and a was feasible. i performed the action of attempting to do a (that is, the action/event e is the attempt to do a), but now a has not been done and i no longer has the intention to do a, and φ is true. The informal implication is that φ is the reason that the action failed, though this causality is not expressed formally in the semantic model.
Examples	Agent j informs i that it has failed to open a file.

```
(failure
  :sender (agent-identifier :name j)
  :receiver (set (agent-identifier :name i))
  :content "((action (agent-identifier :name j) (open \"foo.txt\"))
(error-message \"No such file: foo.txt\"))"
  :language fipa-sl)
```

But what if the agent wants to do something for another agent? (As opposed to request)

- Propose



Propose

Summary	The action of submitting a proposal to perform a certain action, given certain preconditions.
Content	A tuple containing an action description, representing the action the sender is proposing to perform, and a proposition representing preconditions on the performance of the action.
Description	
Formal Model	<i, propose (j, <i, act>, φ)> ≡ <i, inform (j, I_j Done (<i, act>, φ) → I_i Done (<i, act>, φ))> FP: $B_i \alpha \wedge \neg B_i (Bif_j \alpha \vee Uif_j \alpha)$ RE: $B_j \alpha$ Where: α = I_j Done (<i, act>, φ) → I_i Done (<i, act>, φ) Agent i informs j that, once j informs i that j has adopted the intention for i to perform action act, and the preconditions for i performing act have been established, i will adopt the intention to perform the communicative act.
Examples	Agent j proposes to i to sell 50 boxes of plums for $5 (this example continues the example of cfp).

```
(propose
 :sender (agent-identifier :name j)
 :receiver (set (agent-identifier :name i))
 :content "((action j (sell plum 50))
          (= (any ?x (and (= (price plum) ?x)
 (< ?x 10))) 5)"
 :ontology fruit-market
 :in-reply-to proposal2
 :language fipa-sl)
```

Accept Proposal

Summary	The action of accepting a previously submitted proposal to perform an action.
Content	A tuple consisting of an action expression denoting the action to be done, and a proposition giving the conditions of the agreement.
Description	
Formal Model	<i, accept-proposal (j, <j, act>, φ))> ≡ <i, inform (j, I_i Done (<j, act>, φ))> FP: $B_i \alpha \wedge \neg$Bi ($Bif_j \alpha \vee$ Uifj α) RE: $B_j \alpha$ Where: α = I_i Done (<i, act>, φ)
Examples	Agent i informs j that it accepts an offer from j to stream a given multimedia title to channel 19 when the customer is ready. Agent i will inform j of this fact when appropriate.

```
(accept-proposal
 :sender (agent-identifier :name i)
 :receiver (set (agent-identifier :name j))
 :in-reply-to bid089
 :content "((action (agent-identifier :name j)
              (stream-content movie1234 19))
             (B (agent-identifier :name j) (ready customer78)))"
 :language fipa-sl)
```

Reject Proposal

Summary	The action of rejecting a proposal to perform some action during a negotiation.
Content	A tuple consisting of an action description and a proposition which formed the original proposal being rejected, and a further proposition which denotes the reason for the rejection.
Description	
Formal Model	$\langle i, \text{reject-proposal}(j, \langle j, act\rangle, \varphi, \psi)\rangle \equiv \langle i, \text{inform}(j, \neg I_i\, Done(\langle j, act\rangle, \varphi) \land \psi)\rangle$ $FP : B_i\, \alpha \land \neg B_i\, (Bif_j\, \alpha \lor Uif_j\, \alpha)$ $RE : Bj\, \alpha$ Where: $\alpha = \neg I_i\, Done(\langle j, act\rangle, \varphi) \land \psi$ Agent i informs j that, because of proposition ψ, i does not have the intention for j to perform action act with precondition φ.
Examples	Agent i informs j that it rejects an offer from j to sell.

```
(reject-proposal
 :sender (agent-identifier :name i)
 :receiver (set (agent-identifier :name j))
 :content "((action (agent-identifier :name j)(sell plum 50))
            (cost 200)(price-too-high 50))"
 :in-reply-to proposal13)
```

Others

- **Confirm**: The sender informs the receiver that a given proposition is true, where the receiver is known to be uncertain about the proposition. (see also disconfirm)
- **Inform If**: A macro action for the agent of the action to inform the recipient whether or not a proposition is true. (see also inform)
- **Inform Ref**: A macro action for sender to inform the receiver the object which corresponds to a descriptor, for example, a name. (see inform)
- **Query Ref**: The action of asking another agent for the object referred to by a referential expression.

- **Propagate**: The sender intends that the receiver treat the embedded message as sent directly to the receiver, and wants the receiver to identify the agents denoted by the given descriptor and send the received propagate message to them.
- **Proxy**: The sender wants the receiver to select target agents denoted by a given description and to send an embedded message to them.

Others

- **Request When**: The sender wants the receiver to perform some action when some given proposition becomes true. (see also request)
- **Request Whenever**: The sender wants the receiver to perform some action as soon as some proposition becomes true and thereafter each time the proposition becomes true again. (see also request)
- **Subscribe**: The act of requesting a persistent intention to notify the sender of the value of a reference, and to notify again whenever the object identified by the reference changes. (see also request)

- **Call for Proposal**:The action of calling for proposals to perform a given action. (CfP protocol)
