Agent Communication

José M. Vidal

Tue Sep 23 10:15:32 EDT 2003

We cover agent communications, KIF, KQML, and FIPA-ACL. This talk summarizes:

- Michael Wooldridge. Introduction to MultiAgent Systems.[1] John Wiley and Sons;. 2002. Chapter 8
- FIPA ACL Message Structure Specification.[2] 2002.
- FIPA Communicative Act Library Specification.[3] 2002.
- FIPA Contract Net Interaction Protocol Specification.[4] 2002.
- FIPA Iterated Contract Net Interaction Protocol Specification.[5] 2002.
- FIPA English Auction Interaction Protocol Specification.[6] 2001.
- James Odell, H. Van Dyke Parunak, and Bernhard Bauer. Representing Agent Interaction Protocols in UML.[7] Paolo Ciancarini and Michael Wooldridge ed.In *Agent-Oriented Software Engineering,* p. 121–140, Springer-Verlag. 2001.

1 Communication

- Processes need to synchronize. e.g., when updating a shared variable.
- In OOP messaging is done by invoking member functions–`object.msg()`
- Autonomous agents can say no.
- So, all one can do is try to influence the agent to doing something.

[1] `http://www.amazon.com/exec/obidos/ASIN/047149691X/multiagentcom/`
[2] `http://www.fipa.org/specs/fipa00061/`
[3] `http://www.fipa.org/specs/fipa00037/`
[4] `http://www.fipa.org/specs/fipa00029/`
[5] `http://www.fipa.org/specs/fipa00030/`
[6] `http://www.fipa.org/specs/fipa00031/`
[7] `http://jmvidal.cse.sc.edu/library/odell01a.pdf`

2 Speech Act Theory

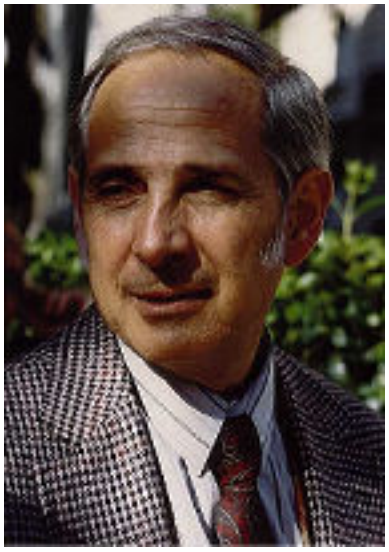
- **Speech Acts** were introduced by John Austin[8].
- They have the characteristic of actions in that their utterance changes the world in an analogous way to physical actions.
- He identified a number of **performative verbs** which correspond to different types of speech acts.
  - request
  - inform
  - promise
- He also distinguished three aspecs of speech acts

1. **locutionary act** - the act of making an utterance.
2. **illocutionary act** - the action performed in saying something.
3. **perlocution** - the effect of the act

- If I say: "I will teach you the beauty of multiagent systems"
- locutionary- is the actual speaking I did.
- illocutionary- is the promise I made to you
- perlocution- is the effects it had on you. Hopefully, to convince you that I am here with a precise purpose.

[8] `http://www.philosophypages.com/ph/aust.htm`

2.1 John Searle



- The work on speech acts was extend by John Searle[9] on his "Speech Acts" (1969) book.
- He attempted to classify speech acts into five classes.

1. **Representatives** - commit the speaker to the truth of an expressed proposition. (informing)
2. **Directives** - attempts on the part of the speaker to get the hearer to do something (request).
3. **Commissives** - commit the speaker to a course of action (promise).
4. **Expressives** - express some psychological state (thanking).
5. **Declarations** - effect some changes in an institutional state of affairs. (declare marriage).

3 Knowledge Interchange Format

- KIF[10] grew out of a the DARPA-funded Knowledge Sharing Effort[11]
- It is basically first-order logic in lisp format.
- It is meant to be a content language.

[9] `http://ist-socrates.berkeley.edu/~jsearle/`
[10] `http://www.cs.umbc.edu/kse/kif/`
[11] `http://www-ksl.stanford.edu/knowledge-sharing/papers/kse-overview.html`

- Agents can define properties of things in a domain as well as relationships between things and general properties of the domain.
- For example, the sentences shown below encode 3 tuples in a personnel database

```
(salary 015-46-3946 widgets 72000)
(salary 026-40-9152 grommets 36000)
(salary 415-32-4707 fidgets 42000)
```

- The following sentence states that one chip is larger than another:

```
(> (* (width chip1) (length chip1))
(* (width chip2) (length chip2)))
```

- This one asserts that the number obtained by raising any real–number ?x to an even power ?n is positive:

```
(=> (and (real-number ?x)
(even-number ?n))
(> (expt ?x ?n) 0))
```

- This one says that on object is a bacher if it is a man and is not married

```
(defrelation bachelor (?x) :=
  (and (man ?x)
    (not (married ?x))))
```

- This one says that anyone who is a person is also a mammal

```
(defrelation person (?x) :=> (mammal ?x))
```

4 Knowledge Query and Manipulation Language

- KQML[12] is a language for communications. It is also part of the KSE.
- Each message starts with a **performative** and has a number of parameters. It looks like:

```
(ask-one
  :content (PRICE IBM ?price)
  :receiver stock-server
  :language LPROLOG
  :ontology NYSE-TICKS)
```

[12] http://www.cs.umbc.edu/kqml/

Parameter	Meaning
`:content`	content of the message
`:force`	wether the sender of the message will ever deny the content of the message
`:reply-with`	whether the sender expects a reply and, if so, an identifier for the reply
`:in-reply-to`	reference to the `:reply-with` parameter
`:sender`	sender of the message
`:receiver`	intended recipient of the message
`:language`	language of the content
`:ontology`	ontology of the content

4.1 KQML Performatives

Performative	Meaning
achieve	S wants R to do make something true of their environment
advertise	S is particulary suited to processing a performative
ask-about	S wants all relevant sentences in R s VKB
ask-all	S wants all of R s answers to a question
ask-if	S wants to know if the sentence is in R s VKB
ask-one	S wants one of R s answers to a question
break	S wants R to break an established pipe
broadcast	S wants R to send a performative over all connections
broker-all	S wants R to collect all responses to a performative
broker-one	S wants R to get help in responding to a performative
deny	the embedded performative does not apply to S anymore
delete	S wants R to remove a ground sentence from its VKB
delete-all	S wants R to remove all matching sentences from its VKB
delete-one	S wants R to remove om matching sentence from its VKB
discard	S will not want R s remaining responses to a previous performative
eos	end of a stream of responses to an earlier query
error	S considers R s ealier message to be mal formed
evaluate	S wants R to simplify the sentence
forward	S wants R to route a performative
generator	same as a standby of a stream all
insert	S asks R to add content to its VKB
monitor	S wants updates to R s response to a stream all
next	S wants R s next response to a previously mentioned performative
pipe	S wants R to route all further performatives to a another agent
ready	S is ready to respond to R s previously mentioned performative
recommend-all	S wants all names of agents who can respond to a performative
recommend-one	S wants the name of an agent who can respond to a performative
recruit-all	S wants R to get all suitable agents to respond to a performative
recruit-one	S wants R to get another agent to respond to a performative
register	S can deliver performatives to some named agent
reply	communicates an expected reply
rest	S wants R s remaining responses to a previously mentioned performative
sorry	S cannot provide a more informative reply
standby	S wants R to be ready to respond to a performative
stream-about	multiple response version of ask about
stream-all	multiple response version of ask all
subscribe	S wants updates to R s response to a performative
tell	the sentence in S s VKB
transport-address	S associates symbolic name with transport address
unregister	a deny of a register
untell	the sentence is not in S s VKB

4.2 KQML Example

- An example dialogue:
```
(evaluate
  :sender A :receiver B
  :language KIF :ontology motors
  :reply-with q1 :content (val (torque m1)))
(reply
  :sender B :receiver A
  :language KIF :ontology motors
  :in-reply-to q1 :content (= (torque m1) (scalar 12 kgf)))
```
-
```
(stream-about
  :sender A :receiver B
  :language KIF :ontology motors
  :reply-with q1 :content m1)
(tell
  :sender B :receiver A
  :in-reply-to q1 :content (= (torque m1) (scalar 12 kgf)))
(tell
  :sender B :receiver A
  :in-reply-to q1 :content (= (status m1) normal))
(eos
  :sender B :receiver A
  :in-reply-to q1)
```

5 FIPA ACL

- The Foundation for Intelligent Physical Agents[13] was founded on 1995.
- They developed new Agent Communication Language, based on what was learned on KQML (they are similar).

[13] http://jmvidal.cse.sc.edu/talks/agentcommunication/../fipaintro/index.xml

5.1 Message Structure

Element	I
performative	Denotes the type of the co
sender	Denotes the identity of the sender of the mess
receiver	Denotes the identity of th
reply-to	This element indicates that subsequent messages in this conversation thread are to be dire
content	Denotes the content of the message
language	Denotes the language in w
encoding	Denotes the specific encodi
ontology	Denotes the ontology(s) used to give a
protocol	Denotes the interaction protocol that the
conversation-id	Introduces an expression (a conversation identifier) which is used to iden
reply-with	Introduces an expression that will be us
in-reply-to	Denotes an expression that references
reply-by	Denotes a time and/or date expression which indicates the la

- The only element that is mandatory is the **performative** , but most will also contain **sender** , **receiver** , and **content** elements.
- If an agent does not recognize or is unable to process one or more of the elements or element values, it can reply with the appropriate **not-understood**

5.2 FIPA Performatives

- They are listed in the FIPA Communicative Act Library Specification[14].
- `accept-proposal`- The action of accepting a previously submitted `propose` to perform an action.
- `agree`- The action of agreeing to perform a `requestd` action made by another agent. Agent will carry it out.
- `cancel`- Agent wants to cancel a previous `request`.
- `cfp`- Agent issues a call for proposals. It contains the actions to be carried out and any other terms of the agreement.
- `confirm`- The sender confirms to the receiver the truth of the content. The sender initially believed that the receiver was unsure about it.
- `disconfirm`- The sender confirms to the receiver the falsity of the content.
- `failure`- Tell the other agent that a previously `requested` action failed.

[14] `http://jmvidal.cse.sc.edu/talks/agentcommunication/../../library/XC00037H.pdf.xml`

- `inform`- Tell another agent something. The sender must believe in the truth of the statement. *Most used performative* .
- `inform-if`- Used as content of `request` to ask another agent to tell us is a statement is true or false.
- `inform-ref`- Like `inform-if` but asks for the value of the expression.
- `not-understood`- Sent when the agent did not understand the message.
- `propagate`- Asks another agent so forward this same `propagate` message to others.
- `propose`- Used as a response to a `cfp`. Agent proposes a deal.
- `proxy`- The sender wants the receiver to select target agents denoted by a given description and to send an embedded message to them.
- `query-if`- The action of asking another agent whether or not a given proposition is true.
- `query-ref`- The action of asking another agent for the object referred to by an referential expression.
- `refuse`- The action of refusing to perform a given action, and explaining the reason for the refusal.
- `reject-proposal`- The action of rejecting a proposal to perform some action during a negotiation.
- `request`- The sender requests the receiver to perform some action. Usually to request the receiver to perform another communicative act.
- `request-when`- The sender wants the receiver to perform some action when some given proposition becomes true.
- `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.
- `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.

5.3 Inform and Request

- The semantics are given using a formal (logical) language called SL, which uses beliefs, desires, and uncertain beliefs.
- FP: is the **feasability precondition** which the sender of the message must satisfy in order to send this message.

- RE: is the **rational effect** of the action–the purpose of the message. It cannot be guaranteed since agents are autonoumous and rational.
- All FIPA-ACL performative semantics are defined in terms of `inform` and `request`. <i, inform(j,ϕ)>

  FP: $B_i\phi \quad \neg B_i(Bif_j\phi \vee Uij_j\phi)$

  RE: $B_j\phi$

  <i,request(j,α)>

  FP: B_iAgent(α,j) $\neg B_i I_j$Done(α)

  RE: Done(α)
- Bifϕ means that agent i has a definite opinion about the truth or falsity of ϕ
- Uif$_i\phi$ means that i is uncertain about ϕ
- Agent(α,j) means that the agent of action α is j.
- *Aside:* There is currently some controversy about the feasability of verifying whether a system obeys the ACL semantics; since it uses mental states the only way to verify is by examining the agents' mental states.

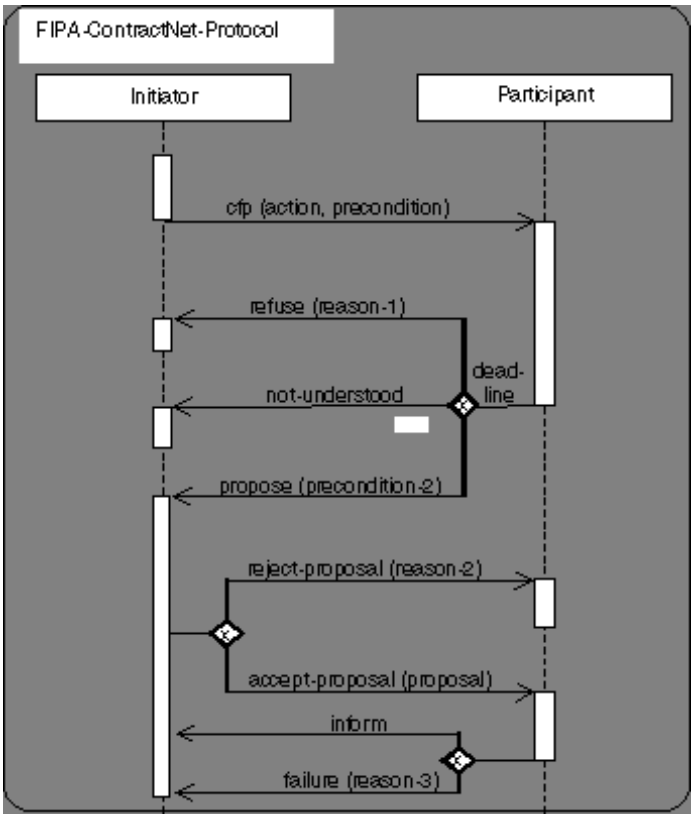
5.4 Performative: cfp

Summary	
Message Content	
Description	*CFP* is a general-purpose action to initiate a negotiation process by making a call for pr
Formal Model	
Example	

5.5 Interaction Protocols

- They are typical patters of message exchange.
- A designer of agent systems has the choice to make the agents sufficiently aware of the meanings of the messages and the goals, beliefs and other mental attitudes the agent possesses, and that the agent's planning process causes such IPs to arise spontaneously from the agents' choices. This is hard to do!
- IPs are given pre-defined names.
- A FIPA ACL-compliant agent need not implement any of the standard IPs, nor is it restricted from using other IP names. However, if one of the standard IP names is used, the agent must behave consistently with the IP specification given here.

5.6 Agent UML



- Needed because UML lacks support for agents since they run on their own thread.
- FIPA calls IP representation in AUML, Protocol Diagrams (PD).
- The vertical dimension is time, proceeding down the page.

5.6.1 Agent Roles

- Agents can perform various roles within on IP. e.g., in contract-net the buyer and seller roles could be played by one agent, at different times.
- A protocol can be defined between different concrete agent instances or a set of agents satisfying a distinguished role and/or class. An agent satisfying a distinguished AgentRole and class is called agent of a given AgentRole and class, respectively.
- An AgentRole is shown as a rectangle that is filled with some string of one of the following forms:

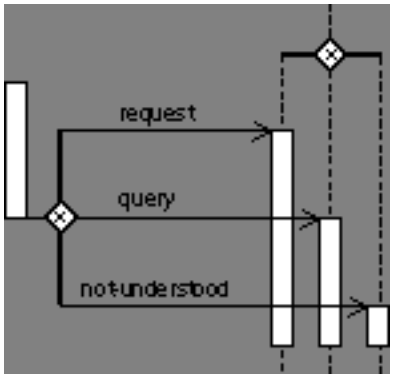
- Seller **role** : This denotes arbitrary agents satisfying the distinguished AgentRole.
- Seller-1/Seller, Buyer **instance / role-1 ... role-n** : This denotes a distinguished agent instance that satisfies the AgentRoles 1-n where n>0.
- Seller-1/Seller, Buyer:ComercialAgent
  instance / role-1 ... role-n : class : This denotes a distinguished agent instance that satisfied the AgentRoles 1-n where n>0 and class it belongs to.

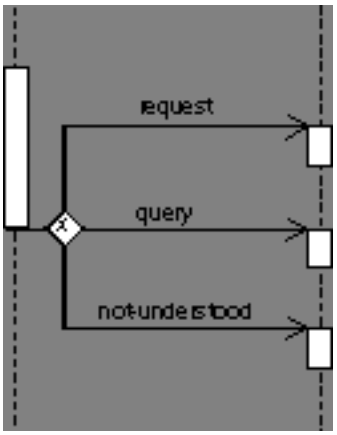
5.6.2 Agent Lifeline

- The agent lifeline defines the time period when an agent exists.
- An agent lifeline is shown as a vertical dashed line.
- Lifelines can be split to show AND/OR parallelism:
  -  AND parallelism starts at a horizontal heavy bar.
  -  OR parallelism (inclusive-or) starts at a horizontal heavy bar with a non-filled diamond, and,
  -  decision (exclusive-or) starts at a horizontal heavy bar with a non-filled diamond with "x" inside the diamond and is continued with a set of parallel vertical lifelines connected to the heavy bar.

5.6.3 Threads of Interaction

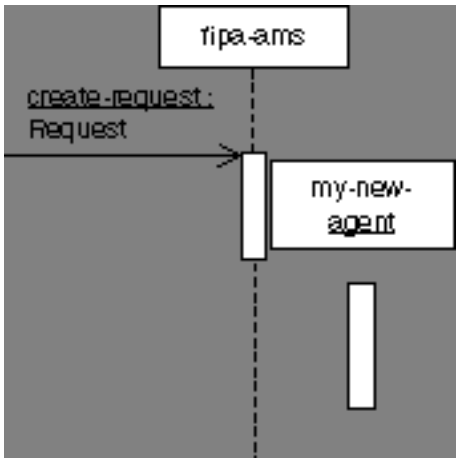
- Since the behaviour of an AgentRole depends on the incoming message and different communicative acts are allowed as an answer to a communicative act, the thread of interaction, that is, the processing of incoming messages, has to be split up into different threads of interaction. The lifeline of an AgentRole is split and the thread of interaction defines the reaction to received messages.
- A thread of interaction is shown as a tall thin rectangle whose top is aligned with its initiation time and whose bottom is aligned with its completion time. It is drawn over the lifeline of an AgentRole.
- The following two representations are equivalent.



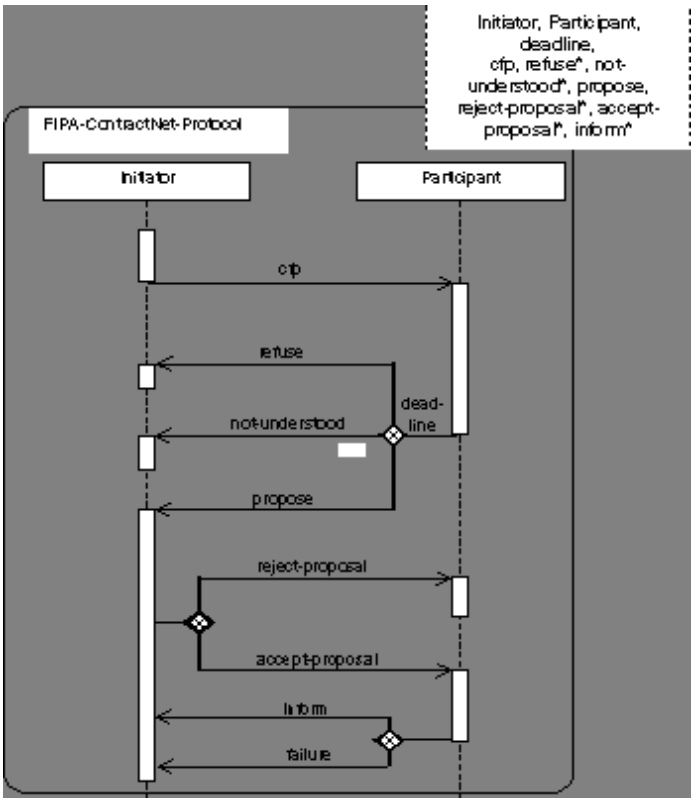


5.6.4 Messages

- A message or sending of a communicative act is a communication from one AgentRole to another that conveys information with the expectation that the receiving AgentRole would react according to the semantics of the communicative act. The specification of the protocol says nothing about the implementation of the processing of the communicative act.
- A message sending is shown as a horizontal solid arrow from a thread of interaction of an AgentRole to another thread of interaction of another AgentRole.
- Each arrow is labelled with a message label that has the following syntax:
  - **predecessor** This consists of at most one natural number followed by a slash (/) defining the sequencing of a parallel construct or the number of the input and output parameter.
  - **guard-condition** This is a usual UML guard condition, with the semantics, that the message is sent iff the guard is true.
  - **sequence-expression** This is a constraint, especially with n..m which denotes that the message is sent n up to m times.
  - **communicative-act** This is either the name, that is, a string representation with an underlined name, of a concrete communicative act instance, the name of a concrete communicative act instance and its associated communicative act, written as name:communicative-act or only the name of the communicative act, for example, inform.
  - **argument-list** This is a comma-separated list of arguments enclosed in parentheses.



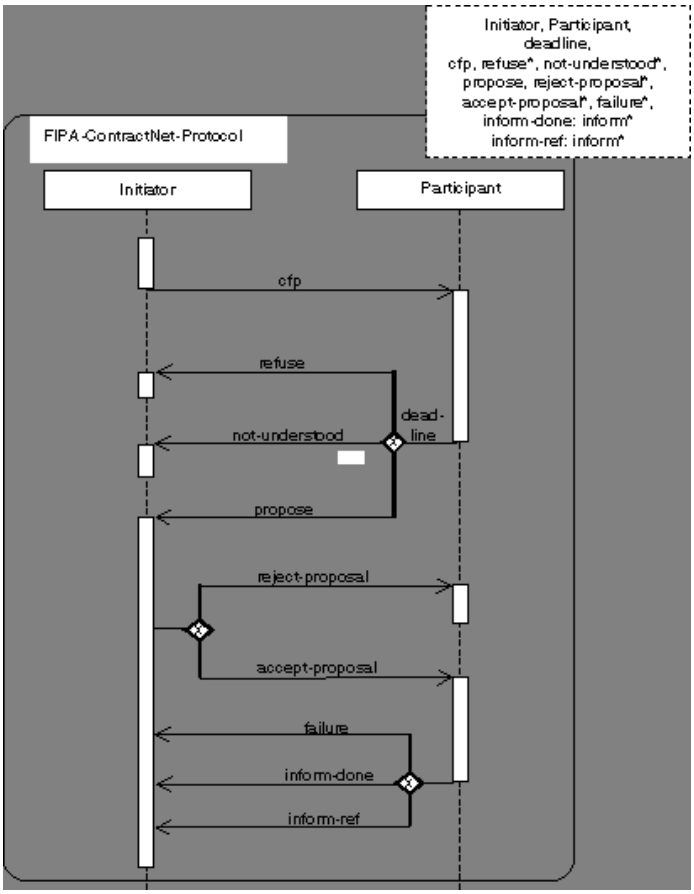
5.6.5 Parameterised Protocols



- A parameterised protocol is the description for an IP with one or more unbound formal parameters. It therefore defines a family of protocols, each protocol specified by binding the parameters to actual values.

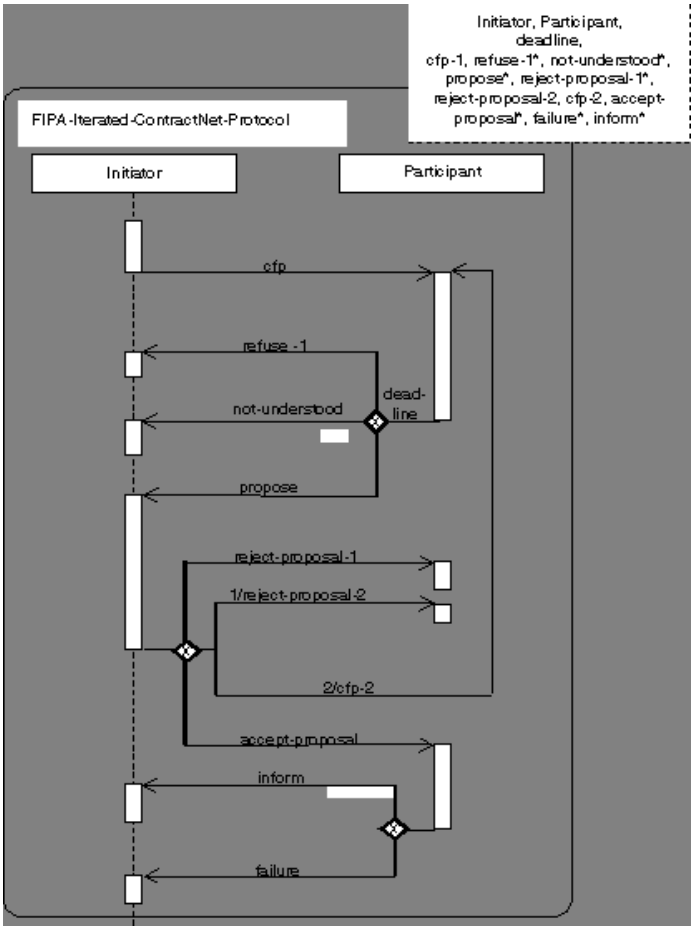
- A small dashed rectangle is superimposed on the upper right-hand corner of the rectangle with rounded corners as when defining a nested protocol. The dashed rectangle contains a parameter list of formal parameters for the protocol.
- Communicative act can be marked with an asterisk in the parameter specification, denoting different kinds of messages that can alternatively be sent in this context.

5.7 Contract Net IP



- Real world issues of cancelling actions, asynchrony, abnormal or unexpected IP termination, nested IPs, and the like, are explicitly not addressed here.

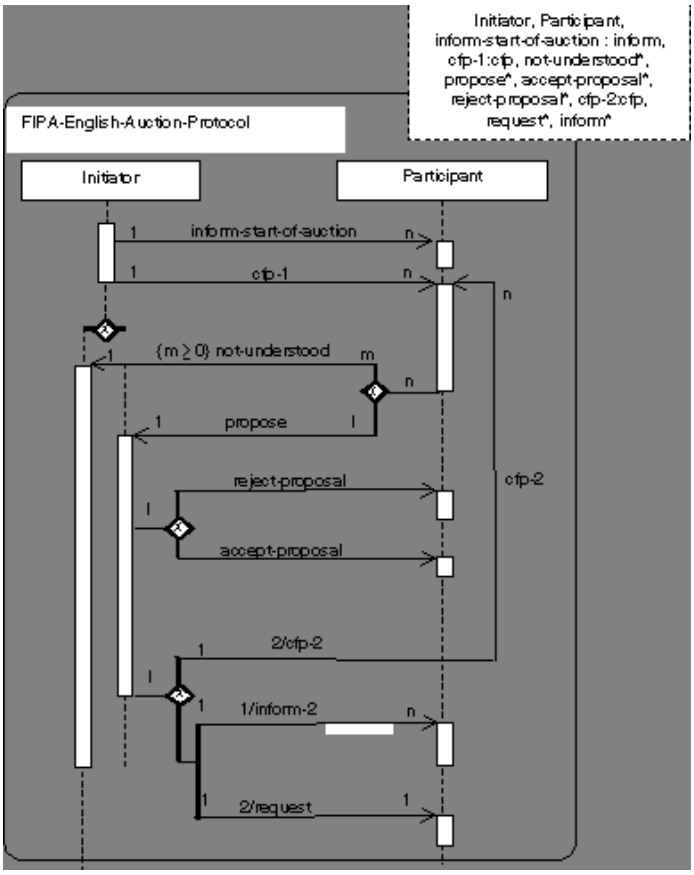
5.8 Iterated Contract Net IP



After receiving the bids, the Initiator can either (xor):

1. reject-proposal-1
2. accept-proposal
3. First refect-proposal-2, then cfp-2 (and we repeat...)

5.9 English Auction IP



- Even though the auction will continue for as long as there is at least one bidder, the agents will need to know whether their bid (represented by the propose act) has been accepted. Hence the appearance in the IP of the accept-proposal and reject-proposal acts, despite this being implicit in the English Auction process that is being modelled.
- At the end of the IP, the auctioneer will typically enter a request IP (see [FIPA00026]) with the winning bidder to complete the auction transaction