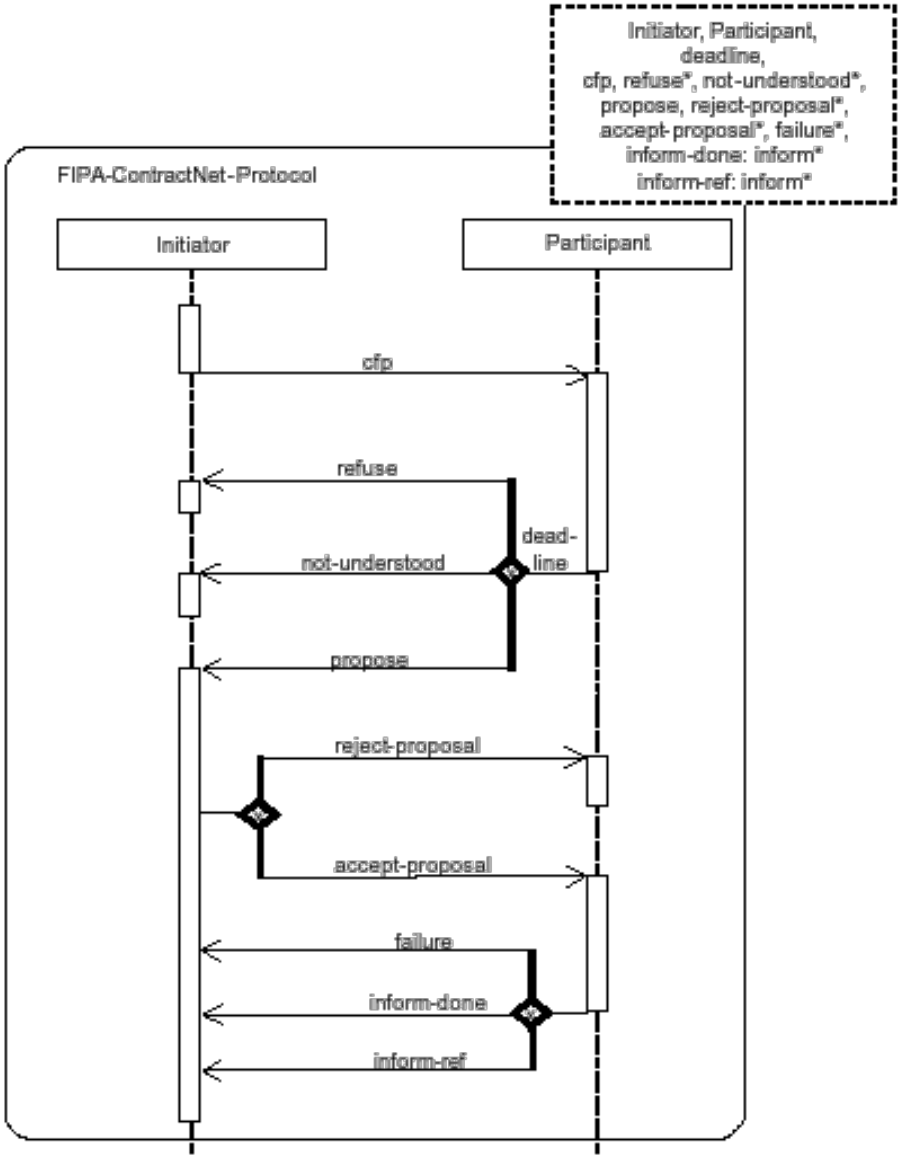
Contract Net Protocol

Protocolos FIPA: FIPA-contract-net

- un agente desea que se realice una acción
- hay varios candidatos
- se desea minimizar una función que caracteriza la tarea (precio)



Sistema Multiagente

1 iniciador y 5 participantes

```
MAS cnp {
        agents:
                c;      // the CNP initiator
                p #3;   // three participants able to send proposals
                pr;     // a participant that always refuses
                pn;     // a participant that does not answer

}
```

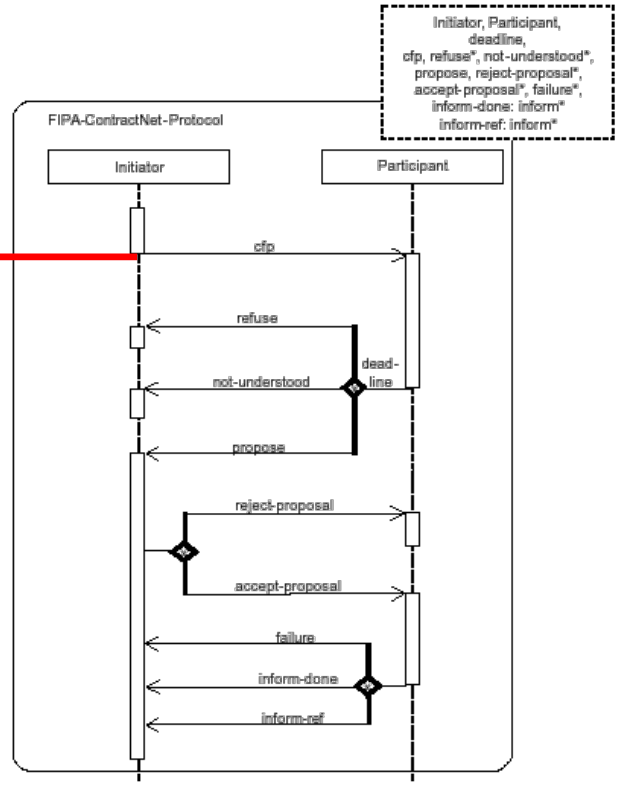
Cuando se inicializan los participantes enviarán un mensaje al iniciador (c) introduciendose como participantes.

Agente que no contesta (pn)

```
// the name of the agent playing initiator in the CNP

plays(initiator,c).

// send a message to the initiator introducing myself
// as a participant

+plays(initiator,In)
        : .my_name(Me)
        <- .send(In,tell,introduction(participant,Me)).
```

Agente que siempre rechaza (pr)

.send(LP, tell, cfp(Id,Object))




```
plays(initiator,c).

// send a message to the initiator introducing myself
// as a participant

+plays(initiator,In)
        : .my_name(Me)
        <- .send(In,tell,introduction(participant,Me)).

// plan to answer a CFP

+cfp(CNPId,Task)[source(A)]
        : plays(initiator,A)
        <- .send(A,tell,refuse(CNPId)).
```

Agente participante (p) [3 instancias]

```
// gets the price for the product,
// (a random value between 100 and 110).
price(Task,X) :- .random(R) & X = (10*R)+100.

plays(initiator,c).

/* Plans */

// send a message to initiator introducing myself
// as a participant
+plays(initiator,In)
        : .my_name(Me)
        <- .send(In,tell,introduction(participant,Me)).

// answer a Call For Proposal
@c1 +cfp(CNPId,Task)[source(A)]
        : plays(initiator,A) & price(Task,Offer)
        <- +proposal(CNPId,Task,Offer); // remember my proposal
        .send(A,tell,propose(CNPId,Offer)).

@r1 +accept_proposal(CNPId)
        : proposal(CNPId,Task,Offer)
        <- .print("My proposal '",Offer,"' won CNP ",CNPId,
        " for ",Task,"!").
// do the task and report to initiator


@r2 +reject_proposal(CNPId)
        <- .print("I lost CNP ",CNPId, ".");
        -proposal(CNPId,_,_). // clear memory
```

Agente CNP

```
/* Rules */

all_proposals_received(CNPId) :-

        .count(introduction(participant,_),NP) & // number of participants

        .count(propose(CNPId,_), NO) & // number of proposes received

        .count(refuse(CNPId), NR) & // number of refusals received

        NP = NO + NR.

/* Initial goals */

!startCNP(1,fix(computer_123)).
```

Agente CNP(2)

```
/* Plans */

// start the CNP

+!startCNP(Id,Object)
        <- .wait(2000); // wait participants introduction
        +cnp_state(Id,propose); // remember the state of the CNP
        .findall(Name,introduction(participant,Name),LP);
        .print("Sending CFP to ",LP);
        .send(LP,tell,cfp(Id,Object));
        .concat("+!contract(",Id,")",Event);

        // the deadline of the CNP is now + 4 seconds, so
        // the event +!contract(Id) is generated at that time

        .at("now +4 seconds", Event).
```

Agente CNP(3)

```
// receive proposal
// if all proposal have been received, don't wait for the deadline

@r1 +propose(CNPId,Offer)
        : cnp_state(CNPId,propose) & all_proposals_received(CNPId)
        <- !contract(CNPId).

// receive refusals

@r2 +refuse(CNPId)
        : cnp_state(CNPId,propose) & all_proposals_received(CNPId)
        <- !contract(CNPId).
```

Agente CNP(4)

```
// this plan needs to be atomic so as not to accept
// proposals or refusals while contracting
@lc1[atomic]
        +!contract(CNPId)
        : cnp_state(CNPId,propose)
        <- -+cnp_state(CNPId,contract);
                .findall(offer(O,A),propose(CNPId,O)[source(A)],L);
                .print("Offers are ",L);
                L \== []; // constraint the plan execution to at least one offer
                .min(L,offer(WOf,WAg)); // sort offers, the first is the best
                .print("Winner is ",WAg," with ",WOf);
                !announce_result(CNPId,L,WAg);
                -+cnp_state(Id,finished).
```

Agente CNP(4)

```
// nothing todo, the current phase is not 'propose'
@lc2 +!contract(CNPId).

-!contract(CNPId)
        <- .print("CNP ",CNPId," has failed!").

+!announce_result(_,[],_).

// announce to the winner
+!announce_result(CNPId,[offer(O,WAg)|T],WAg)
        <- .send(WAg,tell,accept_proposal(CNPId));
        !announce_result(CNPId,T,WAg).

// announce to others
+!announce_result(CNPId,[offer(O,LAg)|T],WAg)
        <- .send(LAg,tell,reject_proposal(CNPId));
        !announce_result(CNPId,T,WAg).
```