What is Contract Net Interaction Protocol?

Zafeer Alibhai, B.A.Sc.

IRMS Laboratory, SFU

July 25, 2003

In the next 20 years, computers will become so cheap that they will be everywhere – ultimately every product will have at least one computer chip in it. More complex items like cars will have hundreds, if not thousands or more, computer chips each controlling a different aspect of the vehicle. Computers will essentially be a necessity of life, like the oxygen we breathe. How will these computers interact and communicate with one another? How will they work together, solving common goals, to fulfill human needs and desires? A distributed system using Contract Net Interaction Protocol could solve these problems, revolutionizing not only the way we think of computers but our entire world. A distributed system could literally connect every person to every other entity (person, business, object, etc.) creating an intelligent global net. The possibilities of this distributed network compared to the current internet would be like comparing the modern computer to the abacus used in ancient times.

Contract Net specifies the interaction between agents for fully automated competitive negotiation through the use of contracts. In essence, Contract Net allows tasks to be distributed among a group of agents. Originally perceived by Smith in 1980, it was first applied to a simulated distributed acoustic sensor network. One of the most promising uses for Contract Net is to create an electronic marketplace for buying and selling goods. An important idea to note is that each agent is self-interested, meaning that the final solution maybe be the best for the agents involved, but not for the group as a whole.

A simple analogy is to think of a system like eBay. However, Contract Net is an improvement on this online marketplace. Under a Contract Net system, a user could specify the good he wanted as well as a price maximum price he was willing to pay. The agent program would then find other user(s) willing to sell the good within the desired price range. The user with the lowest price would then be selected to fulfill the contract. Other details such as units (quantity) of the good and the delivery time could also be specified.

This example can be expanded to demonstrate wide-scale industrial and commercial uses of Contract Net. In any manufacturing process specific inputs are required at certain

times, whether they be raw materials, finished goods or resources (such as electricity). Using Contract Net, a manufacturing plant could use agents to negotiate delivery of these inputs at the lowest cost. In any system the cost could be dollars, time units, units of pollution or any other units as long as all agents calculated cost using the same unit. Moreover, the delivery of inputs could be scheduled at a certain frequency, allowing the negotiations to be divided into time slices. The plant could vary the amount of any input required each time slice. In the case of electricity, large loads could be requested during peak periods and visa versa. In allowing varying levels of inputs, the manufacturing plant could be more profitable by reducing storage space for excess stock and waste. Depending on the input, these time slices could be anywhere from a few minutes to days or even months. The time slices could even be eliminated and negotiations made in real-time. This scheduled delivery concept could also be expanded to residential customers for items such as groceries and utilities (water, gas, electricity, etc.). Using other technologies an agent could even automatically restock items, such as food products, as they were used. Agents could also schedule the delivery of specific media at designated times.

For Contract Net, there are two different types of agents, an **Initiator** and a **Participant**. At any time, any one agent can be an Initiator, a Participant or both. Contract Net creates a means for contracting as well as subcontracting tasks (or jobs), in this sense Initiators are managers and Participants are contractors. An Initiator could be an agent willing to buy some good or wanting to sell the right to supply some good. Participants, in each case, would be agents wanting to sell the good or willing to buy the right to supply the good.

Contract Net works best in environments where the following criteria are met:

1. The task is precise and hierarchical in nature
2. The task can be roughly broken down into subtasks
3. The subtasks are mutually independent

The Interaction Protocol is composed of a sequence of four main steps. The agents must go through the following loop of steps to negotiate each contract.

1. The Initiator sends out a Call for Proposals (CFP).
2. Each Participant reviews CFP's and bids on the feasible ones accordingly.
3. The Initiator chooses the best bid and awards the Contract to the respective Participant.
4. The Initiator rejects the other bids.

These steps will be explained further in a step-by-step example of the interactions between agents.

The Foundation for Intelligent Physical Agents (FIPA) has specification for Contract Net Interaction Protocol as well as for communication between agents. The FIPA representation of Contract Net is shown below in Figure 1.

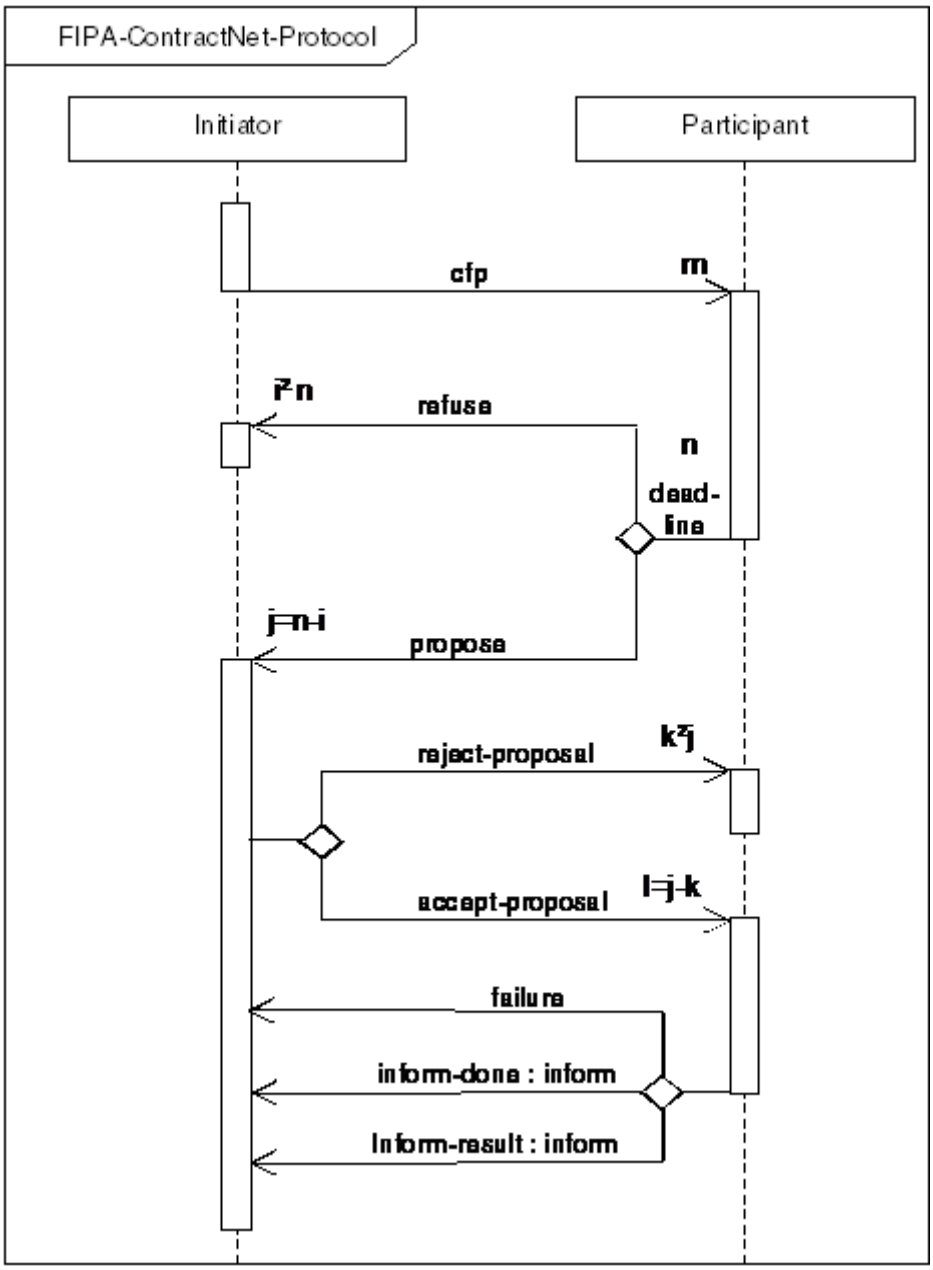

Figure 1 - FIPA Contract Net Interaction Protocol

By looking at Figure 1, we can see that the interaction is much the same as the four steps described above. First the Initiator sends CFP to m possible Participants. The CFP includes units, price and a deadline by which the Participants must respond. The Participants consider the request and by the deadline n have responded. Out of these responses j agents propose contracts while i^2n agents refuse to propose, where i = n – j. Some of the reasons that agents might refuse are because they are unable to provide the

goods needed or they can sell their goods for a higher price. The initiator considers the proposals (or bids) and chooses which l one(s) to accept. In this case the Initiators needs may be fulfilled by more than one Participant. The remaining k^2j bids are rejected, where $k = l - j$. The Initiator will likely reject bids for cost reasons. As a safeguard an additional step is included. Upon completion of the task, the Participant(s) will inform the Initiator that they have delivered the good (done) or the solution to the desired problem (result). In the case that a Participant is unable to complete the task a failure message will be sent. This message will usually include the reason for failure, such as insufficient goods or transportation accident.

In order to simplify communications each message will contain a certain set of parameters. These parameters will include the type of message (as shown in Figure 1), the Initiator ID, the Participant ID, and the Conversation ID as well as the details of the message. The structure of the message would be (<Conversation ID>;<Message Type>;<Sender>;<Receiver>;<Content 1>;<Content 2>;...), where the content items are the details of the message. In the case of a communications error, where part of a message is lost or the message doesn't make sense, a message can be sent back to the sender easily indicating which message was not understood.

The message could also be specified in KQML (Knowledge Query and Manipulation Language). KQML messages contain the parameters from, to, reply-with, ontology, language and content. An example KQML message would be of the form "(monitor :from customer :to supplier :reply-with update-111 :ontology standard-units-and-dimensions :language KIF :content (= (q.magnitude (diameter shaft-a) inches) ?x))".

By using FIPA standards for the design of a local-area Contract Net we can help to ensure that our network of agents is compatible with most other Contract Nets. As a result, the integration process will be much easier, allowing for the construction of broad-spectrum Contract Net systems spanning the globe.

Using Contract Net can be advantageous when compared with other coordination strategies as outlined below.

1. Tasks are assigned (contracts awarded) dynamically, resulting in the better deals for the parties (agents) involved.
2. Agents can enter and leave the system at will.
3. The tasks will be naturally balanced among all the agents since agents that already have contract(s) don't have to bid on new ones. If an agent is already using all its resources, it will be unable to bid on new contracts until the current ones are completed.
4. A reliable strategy for distributed applications with agents that can recover from failures (to be discussed more in the following paragraphs).

Unfortunately, Contract Net also has a number of shortcomings. These issues can not only cause major inefficiencies but also cause the system as a whole to break down. These issues include systematic failures, which many computer (agent) systems suffer from.

Another inherent problem is that Contract Net assumes that all agents are friendly and benevolent. As a result, there is no mechanism to detect conflicts and more importantly solve them. Since each agent is self-interested, in reality it will likely be proactive and antagonistic.

The communications infrastructure isn't completely reliable. Nodes and/or links could fail at random creating pockets of isolating agents. The system must have mechanisms to try to reroute communications in the case of a failure. If communications can't be reestablished, the agents affected must be able recover. An agent could also go offline (die) for any number of reasons and the agents involved would have to act in the same manner as for a communications failure.

As the period of each time slice is decreased, the communications (messages) required increases dramatically. Since the communications infrastructure is already a limiting

factor, increased messages could cause further bottlenecks. A small period would also require agents to process messages and make decisions quickly. Also, a small period could mean that the Initiator misses out on lower cost bids because certain Participant(s) were not able to respond quickly enough.

Deliveries must occur at exactly the right time for certain inputs, such as electricity, otherwise problems, such as power outages, could occur. These problems could cost millions of dollars in lost production, downtime and information loss. In order to ensure delivery times, all the agents would have to run on a synchronized clock. However, synchronizing the internal clocks of all the agents could prove to be very difficult. Another solution would be for negotiating agents to know the difference between their internal clocks. This could be a simpler solution, but would still require a lot of work. Contract Net wasn't originally perceived with the use of time as one of the contract parameters; however, many contracts would require specific delivery times as mentioned above. Synchronized clocks would also be required for the time-bound framework, as discussed in the next paragraph.

Another time issue is the time-bound framework. A Participant could bid on one task and while waiting for a response could receive a CFP for another task. If the Participant has a limited amount of resources, it could lose out by not bidding. On the other hand, it may end up getting two contracts and be unable to fulfill either one. There are three alternative methods that can be used for the time-bound framework. These methods are the Nothing-Guaranteed Protocol (NGP), the Acceptance-Guaranteed Protocol (AGP) and the Finite-Time Guarantee Protocol (FGP). The names of the methods are pretty self-explanatory. Under NGP, there is no guarantee that the Participant will still be able to deliver the goods when the contract is awarded. As a result, extra messages would be needed to confirm that the contract could still be fulfilled and Initiators may need to negotiate contracts again. With AGP, the delivery is always guaranteed, but this means that Participants could lose out if their proposal is rejected because they can't make more bids (using the same resources) until the Initiator decides to award or reject the contract. The best solution for both parties is FGP, where delivery is guaranteed for a certain

period of time. If the Participant doesn't hear back from the Initiator within the given time, it can make other bids. In a more complex system a Participant could tell an Initiator to bid again a short time later if the time period for another proposal was almost expired. The time period under FGP would need to be based on the application as well as Participants circumstances and the time needed for Initiators to select a proposal.

Finally, in any wide-scale implementation of Contract Net, the agents will likely be developed by many different parties. As a result agents can't always be trusted to fulfill their end of a contract because of bugs and/or self-interest among other things. The system must have safeguards to deal with these problems and take punitive action if necessary.

Although Contract Net hasn't been used widely in any commercial applications yet, it has been used to simulate marketplaces in a number of research environments. A group at the University of Massachusetts has built a system called TRACONET (TRAnsportation COoperation NET), which allows the dispatch centers of different companies to automatically route trucks based on marginal cost. The experiment contained two companies (A with three dispatch centers and B with two for a total of five agents) making deliveries over a one-week period. The local solution for each company was found heuristically by using a parallel insertion algorithm. Using Contract Net, significant savings could be seen after a 15 minute simulation, where each agent would run through it's main control loop 50 – 100 times, as seen in Table 1 below.

Table 1 - TRACONET Cost Savings

Dispatch center	Deliveries	Vehicles	Average delivery length	Cost savings in 15 minutes	Cost savings in 30 minutes
A1	65	10	121 km	5%	6%
A2	200	13	169 km	12%	18%
A3	82	21	44 km	31%	34%
B1	124	18	145 km	11%	23%
B2	300	15	270 km	9%	15%
Total	771	77	187 km	11%	17%

For simplicity, the profit from each contract was divided between the two agents involved, meaning the contract price was set as the average of the maximum price in the announcement and the bid price. In the real world, the contract price would likely be the same as the bid price.

Honeywell is working on a system for coordinated aircraft missions and defense using multi-agent systems. In this system, Contract Net will be used for task assignment. Negotiations will occur both before and during the flight in order to assign goals and/or threats to the highest bidder in real-time.

A system called INTERRAP is also being used to model Contract Nets. It was first used to automate a loading dock. Miniature robots were used to load and unload by means of cooperative planning. This planning resolved goal conflicts, allowing robots to synchronize their actions. Contract Net has also been used for other AGV (Automated Guided Vehicle) problems.

There are also a number of agent frameworks which can be used to aide in the development of agents. Many of these frameworks are created by research labs at major corporations or universities, including MIT, Stanford, Carnegie Mellon, Toshiba, IBM, Boeing, Mitsubishi and NTT. One of these frameworks, called MadKit, has developed a simple Contract Net example to simulate a travel agency. Under this system Provider agents sell either plane or train tickets; Client agents are looking for either a plane or train ticket and Broker agents facilitate the interaction between the two.

Once a simple Contract Net is implemented it could easily be expanded. The code for the agents could be recycled, with changes to only the decision making (or optimization) processes and parameters such as marginal cost and inventory. In fact only one original Initiator and Participant would be needed and all the others could reuse most of the functions with minor changes. The Initiator and Participant could also share many

functions, allowing for the easier development of hybrid agents that could act as both an Initiator and a Participant.

Contract Net can also be improved by creating a hybrid Initiator/Participant agent. Since the Initiator sends out the CFP and selects the best bid, there is still a central arbiter for each contract. If all the agents are identical (hybrids), any agent can send out a CFP and the system will find the best solution for all the agents, especially if multiple contracts must be satisfied simultaneously. Under the best solution system, contracts can be fulfilled by multiple agents rather than just one. As a result, the lowest cost producers of the good would be used. The system would also be more redundant, because even if the agent that sent out the CFP went offline, a solution would still be found.

As computer systems and networks become more complex, truly distributed environments will be born. These systems will no longer be dependent on certain agents (central servers) as in the past. The networks will be dynamic, constantly evolving and expanding. Agents will be free to join or leave the system (or network) at their choosing. The importance will no longer be in the hardware but in the software, especially that facilitating communications between agents. Contract Net Interaction Protocol is one of these communication frameworks that are the life-force of distributed systems. In being independent of on another, small groups of agent will be self-sufficient and as long as at least a few of the original agents survive (remain online) the system will continue to function. Quoting from the movie Terminator 3, “there was no central core, it was all software. SkyNet [the distributed system] couldn’t be stopped.”

References

1. "FIPA Contract Net Interaction Protocol Specification." Foundation for Intelligent Physical Agents, December 3, 2002.
2. "FIPA Communicative Act Library Specification." Foundation for Intelligent Physical Agents, December 3, 2002.
3. M. D'Inverno and M. Luck, "Understanding Agent Systems," Springer, 2001.
4. W. Shen, D. Norrie and J.P. Barthes, "Multi-Agent Systems for Concurrent Intelligent Design and Manufacturing," Taylor & Francis, 2000.
5. C. Dellarocas and M. Klein, "Designing Robust, Open Electronic Marketplaces of Contract Net Agents," Massachusetts Institute of Technology, 1999.
6. J. Yang, R. Havaldar, V. Honavar, L. Miller and J. Wong, "Coordination of Distributed Knowledge Networks Using Contract Net Protocol," Iowa State University, 1998.
7. T. Sandholm, "An Implementation of the Contract Net Protocol Based on Marginal Cost Calculations," University of Massachusetts, 1993
8. K. J. Lee, "Time-Bound Framework for Automated Negotiation," School of Business, Korea University, 2000.
9. J.P Miller, "A Cooperation Model for Autonomous Agents," Mitsubishi Electric Digital Library Group, 1996.
10. G. Karsai, J. Doyle, G. Bloor, "Model Integrated Computing and Autonomous Negotiating Teams for Autonomic Logistics."
11. "Working Together: The Contract Net." http://www.csc.liv.ac.uk/~mjw/pubs/imas
12. MadKit. http://www.madkit.org
13. "Terminator 3," July 2, 2003.