A NEW HIERARCHICAL NEGOTIATION SCHEME ON TOP OF A MULTI-AGENT ARCHITECTURE

AHMED SAMEH, NEHAL HAMMOUDA
Department of Computer Science
The American University in Cairo
P.O.Box 2511, Cairo,
EGYPT.

Abstract: The increasing demand for distributed entities to interact in complex domains necessitates the need for cooperation, coordination and negotiation to reach agreement. Examples of such systems are: the flow of work and information through cooperating companies (virtual enterprises), international air traffic controllers, the coordination of logistic processes in shipping companies, the use of flexible transport systems in industrial manufacturing and assembly, and the operation of multi-guidance and control systems. This paper describes the design of a model that uses a hierarchy of negotiation protocols to solve conflicts in any of the INTERRAP layers. Two testbeds have been developed to test the proposed hierarchical model: a Desktop Configuration and a Travel Planning Domains. The testbeds proved the effectivness of the model in solving conflicts among several agents in all the INTERRAP layers. These two domains have been chosen because they introduce various types of conflicts - which could be solved using various negotiation protocols in the model.

Key-Words: Multi-Agent, Negotiation Protocols, Local Planning, Cooperative Planning.

1-Introduction

The development of INTERRAP [Muller96] has been influenced by the following design decisions:

(1) Layered Control: three layers of control describe an agent, each layer represents a certain level of complexity and sophistication.
(2) Layered Knowledge: The beliefs of an agent are stored in a hierarchical Knowledge base, each level of the hierarchy represents a certain information level.
(3) Bottom-up activation: Control is shifted bottom-up; layer i gains control only if layer i-1 is not competent to deal with the situation.
(4) Top-down execution: Each layer uses operational primitives defined at the next lower layer to achieve its goals.
(5) The agent consists of three modules: a World Interface (WIF), Knowledge Base (KB), and a Control Unit (CU): *The WIF:* provides the agent's sensoric, communicative, and actoric links to its environment. *The KB:* stores the agent's beliefs. *The CU:* guides the agent's flow of control among different layers.

Both the KB and the CU modules are structured in three vertical layers. The CU layers are the *Behavior-Based Layer (BBL)*, the *Local Planning Layer (LPL)*, and the *Cooperative Planning Layer (CPL)*. Each control layer consists of two processes called SG (Situation recognition and Goal activation) and PS (Planning, Scheduling, and execution). The KB is partitioned accordingly into a World Model (WM), a Mental Model (MM), and a Social Model (SM): *The WM:* contains object level beliefs about the agent's environment. *The MM:* holds representations of the agent's local goals. These representations are used in the Local Planning Layer. *The SM:* holds representations of other agents' goals. These representations are used in the Cooperative Planning Layer.

2-Negotiation

Research in distributed artificial intelligence (DAI) is concerned with how automated agents can be designed to interact effectively. One important capability that could aid inter-agent cooperation would be that of negotiation. Negotiation is a mechanism allowing autonomous agents to find mutual agreement on a matter. Negotiation is used to resolve conflicts, or to allocate tasks among agents. There are three main issues that determines negotiation mechanisms among agents:

Let D be a negotiation domain; a negotiation is a tuple NEG=(A,R,p,N,U,P,S), where:

-A= $\{a_1,\ldots,\ldots a_k\}$,.. K>=2, is a set of agents a_i with mental states B_i; B_i consists of informational, motivational, and deliberative state of a_i.
-R = $\{r_1,\ldots,r_k\}$, l<=k is a set of roles;
-p is a function that assigns roles to agents: p(a)=r for a $\in$ A, r $\in$ R.
-N is the Negotiation Set
-U= $\{u_1,\ldots,u_k\}$, where u_i: N$\rightarrow$ Real Numbers, is the utility function for agent a_i.
-P= (K,{ π:R X K $\rightarrow 2^K$|r $\in$ R}) Here, R is the set of roles, K=$\{k_1,\ldots k_m\}$ is a finite set of communication primitives. There are two distinguished primitives {start, done}. They facilitate internal control and are not communicated. π Maps communication primitives into a subset of admissible reactions with respect to a specific role r within the protocol.
-S=$\{\sigma_i : P \times R \times K \times 2^D \times U \rightarrow K \times N \mid 1 <= i <= k\}$ is a set of negotiation strategies, one for each agent. The input of σ_i is the current protocol and the role of the agent in the protocol, a received (possibly parameterized) message, the current negotiation set, the agent's utility function, and its current internal state. The output of σ_i is a reaction (i.e., the reply to the received message), the modified negotiation set and internal state.

2-1 Joint Plan Negotiation Protocol

J*oint plans* are plans that represent both actions of multiple agents and coordination relationships among these actions. A joint plan describes a set of single-agent plans whose coordinated execution leads to a world state in which the goals of the agents involved are no longer blocked (not necessarily satisfied, but simply not blocked, in other words the involved agents would be able to resume their local goals instead of being blocked by a conflict for example).

Assume two agents: a_1 and a_2 have run into a conflict and have to agree on a single joint plan. There are two roles in the protocol: R={leader, follower}. Let the role assignment function p be a random function with Pr (p (a_1) = leader) = Pr (p (a_2) = follower)=0.5. Let the protocol be P=(K, π_{JPN}), where K={PROPOSE, ACCEPT, MODIFY, CONFIRM}. In this and all the following negotiation protocols, the comma means OR, and the semi colon means AND. Let p, p', p" be plans, and let π_{JPN} define the Joint Plan Negotiation Protocol as:

Negotiation Protocol

π_{JPN} (start)={PROPOSE (p)}
π_{JPN} (ACCEPT (p)) = {CONFIRM (p)}
π_{JPN}(MODIFY(p, p')) ={ACCEPT(p'),MODIFY(p', p")}
π_{JPN}(PROPOSE(p))={ACCEPT(p),MODIFY(p, p')}
π_{JPN}(CONFIRM(p))={done}.

Beginning with a state start, the leader proposes a solution from the negotiation set. The follower either accepts the proposed solution or it makes a counterproposal; the latter is indicated by MODIFY message. This process continues until either agent accepts the other's proposal. In that case, the proposing agent confirms the deal, and the protocol is finished.

Negotiation Strategy

σ_1(start,N,u_1)={(PROPOSE(p),N) with p$\in$ N with u_1(p)= max u_1(p') where p'$\in$ N}
σ_f(PROPOSE(p), N, u_f)=
{if u_f(p)>=max u_f(p') $\in$ N then (ACCEPT(p),N)
else (MODIFY(p,p'),N – {p}) with p' $\in$ N, u_f(p')= max (p') where p' $\in$ N}
σ_i (ACCEPT (p),N,u_i)= (CONFIRM(p),{p})
σ_i (MODIFY (p,p'),N,u_i)=
{N'=N- {p};
If u_i (p')>=max u_i(p') $\in$ N' then (ACCEPT (p'), {p'})
Else (MODIFY (p', p"), N' – {p'}) where p" $\in$ N' with u_i (p")= max u_i (p") where p"
$\in$ N}
σ_i (CONFIRM(p), N, u_i)=(done, N)

Each agent starts with the offer that maximizes its utility. If the other agent rejects an offer, it proposes a counter offer. In case an offer is rejected, the agent who made this offer deletes the corresponding element from the negotiation set. Similarly, if an agent rejects an offer, it deletes this element from the negotiation set.

2-2 Contract Net Negotiation Protocol

The contract net protocol is the classical protocol for task allocation in MAS [Sandholm 89]. Its original purpose has been to allocat tasks among a group of distributed problem solvers. In the contract net, a manager agent who has a task to be performed is looking for the most suitable agent from the list of bidder agents.

Negotiation Protocol

In the negotiation model presented so far the contract net is formalized as follows: the set of agents is $A=\{a_1,\ldots, a_k\}$. The roles is R={manager, bidder}. The role assignment function is given by $p(a_1)$= manager, $p(a_2)=\ldots=p(a_k)$=bidder, The protocol is $P=(K, \pi_{CNP})$, where K={ANNOUNCE, BID, AWARD, TAKEAWARD, REJECT} and π_{CNP} describes the Contract Net Protocol for manager and bidder. In the following, t is a task, and v denotes a value.

π_{CNP} (start)={ANNOUNCE (t)}
π_{CNP} (ANNOUNCE (t))={BID (t, v), REJECT (t)}
π_{CNP} (BID (t, v)) = {AWARD (t, v), REJECT (t)}
π_{CNP} (REJECT (t))={done}
π_{CNP} (AWARD (t, v))={TAKEAWARD (t, v)}
π_{CNP} (TAKEAWARD (t, v))={done}

Negotiation Strategy

The negotiation strategy for this protocol for agents a_1 ,a_2,….and a_k is as follows:

σ_m(start,u_m)={(ANNOUNCE (t),u_m) with t∈ manager agent Task list, and u_m the manager's task cost }
σ_b (ANNOUNCE (t),u_b)= {if u_b (t) < u_m then BID(t, v)}
σ_m(BID(t),v, u_m)= {If Time >= Fixed Time
{if v<min_all_bids then {(AWARD (t), v); min_all_bids = v}
else {(REJECT (t), v); {(AWARD (t), v) reject the coming bidder and award the one with the min_all_bids value }}
else
{if v<min_all_bids then min_all_bids=v}}
σ_b (AWARD(t), v)=(TAKEAWARD())
σ_m(TAKEAWARD())={done}
σ_b (REJECT(t),v)=(done)

2-3 Negotiated Search Negotiation Protocol

Negotiated search [Lander 94] is a flexible and widely applicable distributed search strategy that specifically addresses issues that arise in MAS comprising reusable and heterogeneous agents. Negotiated search acknowledges the inevitability of conflict among the agents, and exploits that conflict to drive agent interaction and guide local search. It treats conflict as an integral part of problem solving and as a source of control information for agent communication.

Negotiation Protocol

The set of agents in $A=\{a_1\ldots,a_k\}$, the set of roles is R={ Problem Initiator (PI),Solution Initiator (I), Solution Extender (E),Solution Critic(C)}, the protocol $P=(K,\pi_{NSP})$,where K={ANNOUNCE, MERGE, NEW, ANNOUNCE DESIGN, SEND CONFLICT, REGISTER, REJECT DESIGN, SAVE DESIGN , CRITIC, RELAX CONSTRAINT, NULL,INFORM}, and π_{NSP} define the Negotiated Search Protocol. Where Merge is an action performed by the (PI) agent to merge a solution with a design, NEW DESIGN is an action performed by the (PI) agent to come up with a new design in case the solution can't be merged with any previous design. REJECT DESIGN is an action performed by the (PI) agent to stop receiving solutions from the other agents if it received an infeasible solution. ANNOUNCE DESIGNS is an action performed by the (PI) agent when a whole cycle is finished, that is when all involved agents with appropriate roles have sent their solutions to the (PI) agent and their solutions have been processed, this action announces all the resulting designs. SEND CONFLICT is an action performed by the (PI) agent to send a notification for all agents about a certain constraint that should be respected in all coming designs. CRITIC is an action performed by the (C) agent to criticize a complete design. REGISTER is an action performed by all (I) and (E) agents, to register the new constraint that has to be respected in all their coming designs. RELAX CONSTRAINT is performed by the (PI) agent after the maximum number of cycles between agents has been exhausted, the (PI) agent then revises all the previous designs and relaxes their constraints, thus coming up with many announcements to be revisited. NULL is performed by all agents except the (PI), to indicate that there are no more suggestions available. INFORM is performed by all agents except the (PI) to suggest a solution to the (PI) agent.

Let t be a task, ac = Agent Contribution to the solution, d = Design Solution, c = Constraint Violation, v = critic value, s = solution state = {acceptable, unacceptable (a conflict with a flexibility >0), infeasible (a conflict with a flexibility =0) according to presence of conflicts and their flexibility level}.

π_{NSP} (start)={ANNOUNCE (t)}
π_{NSP} (ANNOUNCE (t)) = {INFORM (t, ac, s), NULL ()}
π_{NSP} (NULL ())= { RELAX CONSTRAINTS () }
π_{NSP} (INFORM (t, ac, s))= { (MERGE (t, ac) , NEW DESIGN(t, ac) , REJECT DESIGN(t, ac))

; ANNOUNCE DESIGNS() ; SEND CONFLICT (t, c) ; CRITIC (t)}

π_{NSP} (ANNOUNCE DESIGNS ())={ANNOUNCE (t) , SAVE DESIGN (t) if the design is unacceptable }

π_{NSP} (SEND CONFLICT (c))={REGISTER (c)}

π_{NSP} (RELAX CONSTRAINT (t))={ANNOUNCE (t) , CRITIC (t) }

π_{NSP} (CRITIC (t))= {done}

Negotiation Strategy

The negotiation strategy for this protocol for the agents is as follows:

σ_{PI}(start,N)={(ANNOUNCE(t)) to all agents}

$\sigma_{I,E}$ (ANNOUNCE(t))= {(INFORM (t, ac, s)) If t parameters satisfy the agent-input parameters ,
(NULL ()) if no more suggestions available }

σ_{PI} (INFORM (t, ac, s))=
{(((MERGE (t, ac)) If s != infeasible and ac can be merged with the already formed proposed solutions , (NEW DESIGN (t, ac)) If s != infeasible and ac can not be merged with the already
formed proposed solutions , (REJECT DESIGN ()); (done)) if s=infeasible)
; (ANNOUNCE DESIGNS()) if all agents announcing solutions have sent their proposals to the PI agent that is the cycle is finished ; (SEND CONFLICT (t, c)) to all agents involved in violating the constraints if s = unacceptable; (CRITIC (t)) if design has complete set of components;
(RELAX CONSTRAINTS ()))
if max # of cycles exhausted or there are unacceptable designs already saved then relax the constraints}

σ_{PI} (NULL ())= { (RELAX CONSTRAINTS ()) if max # of cycles exhausted or there are unacceptable designs already saved then relax the constraints}

σ_{PI} (RELAX CONSTRAINTS ())=
{ ((ANNOUNCE(t))) announce all the saved unacceptable designs t to be further processed after there constraints have been relaxed in case they are not complete,
(CRITIC (t)) if available unacceptable designs saved are complete}

$\sigma_{I,E}$ (SEND CONFLICT (c))={(REGISTER (c))}

σ_{PI} (ANNOIUNCE DESIGNS()) = {(ANNOUNCE(t)) announce designs if acceptable ones
exist or announce a new request for a design, (SAVE DESIGN (t)) if the design is unacceptable}

σ_{C} (CRITIC (t))={(REPORT (t, v))}

σ_{C} (REPORT (t))={(done)}

3-The Proposed System

A model can be devised to use a hierarchy of several negotiation protocols to solve conflicts in any of the INTERRAP three layers. Conflicts in INTERRAP occur when there are more than one direction to resolve a certain situation, and such an occurrence can take place in any of the three layers: Behavior Based Layer (BBL), Local Planning Layer (LPL), and Cooperative Planning Layer (CPL).

Assume several agents: a_1, a_2….., a_n have run into a conflict, and that all agents agree on one of them to have the role of the manager to perform the constraint satisfaction search procedure in case all other negotiation protocols have failed. There are two roles in this conflict resolution protocol: R={Manager, Follower}. Let the role assignment function p be a random function with Pr (p (a_1)=Manager) = Pr (p (a_2))= follower) = 0.5. Let the protocol be P=(K, π_{JPN}), where K={QUERY, SATISFY, ACCEPT, REJECT, CONFIRM}. Let p be the whole plan arrived at, p' the partial plan of the follower agent, s the status of the follower agent and let π_{CSP} define the protocol as:

Negotiation Protocol

π_{CSP} (start)={QUERY(s)}

π_{CSP} (QUERY(s)) = {CONFIRM (s)}

π_{CSP} (CONFIRM(s)) ={SATISFY (p), REJECT ()}

π_{CSP} (SATISFY (p))={ACCEPT (p'), REJECT ()}

π_{CSP} (ACCEPT (p'))= {done}.

π_{CSP} (REJECT (p'))= {done}.

Beginning with a state start, the Manager queries the status of all follower agents participating in a problem. Then the Manager performs a constraint satisfaction search procedure, which might come up with a solution or not. If a solution is reached then partial plans are to be sent to the follower agents.

Negotiation Strategy

In the example the negotiation strategy for this protocol for the agents is as follows:

σ_{M} (start)={(QUERY(s)) for all follower agents (i)}

σ_{i} (QUERY(s))={CONFIRM(s)}

σ_{M} (CONFIRM(s))= {SATISFY (P), REJECT () if one follower agent rejects or if all
CONFIRM but no solution found}

σ_{i} (SATISFY(p))={ACCEPT(p'),REJECT()}

σ_M (ACCEPT (p'))= {done}
σ_M (REJECT ())= {done}

4-SystemDesign and Implementation

4-1 Desktop Configuration Domain

The environment of the domain is the Internet, where several software agents exist and employ search techniques for contacting vendors for desktop components' offers. Those offers are written in XML format which agents can read and interrupt. These are considered managing agents who locate vendors, creates agents to represent them, and manage negotiation among them.

Communication between negotiating agents is through sending messages managed by the managing agents. The agents negotiations take place in the Behavior Based Layer (BBL), Local Planning Layer (LPL) and in the Cooperative Planning Layer (CPL) as shown in figure 1. The manufacturer agent has facts in its BBL about several components in its warehouse, and thus when an order arrives, it can have many matches in its warehouse, so it uses negotiation in this layer to come up with the best offer. Negotiation in this layer takes place by creating two agents each using a different criterion for choosing the best possible component. For example one agent orders the components that match the offer according to their prices, and the other agent orders them according to their performance, thus the outcome of the two agents' negotiation would be really the best component out of the many components that match the specification required.

The Desktop Component agent has local plans in its LPL about how to get information from the Internet about the different vendors that deal with specific desktop components. And it might have different plans on how to get such information, either from search engines, or from local repository of previous negotiation results. And so negotiation here can be employed to get the best plan according to criteria like time or space. CNP Negotiation can be employed to get the best results in the least time possible for example from search engines using bidding. The Desktop Component agent needs negotiation in its CPL to come up with the best offer from the many vendors competing for getting the deal, and they negotiate over criteria like the best price for example. And the Desktop main agent also needs negotiation in its CPL to compile the desktop components from the several Desktop Component agents in a way that fits all the constraints between the components. Also the Vendor agent needs negotiation in its CPL to come up with the best offer from the many manufacturers competing for getting the deal.

4-2 Travel Planning Domain

Travel agencies are companies that deal with hotels, airlines, and tour guides to come up with vacation packages that suite customers. Suppose a travel agency deals with many airlines, hotels and tour guides to make a trip from USA to Egypt, at a specific date. Then each agent of the three types will provide its own prices to the Travel agency, who combines these offers in several packages according to their preferences, and then chooses the best package out of the many compiled.

Communication between negotiating agents is through sending messages managed by the managing agent. The agents' negotiation takes place in the Behavior Based Layer (BBL), Local Planning Layer (LPL) and in the Cooperative Planning Layer (CPL). The company agent has facts in its BBL about several offers it can offer, and thus when an order arrives it can have many matches to it, so it uses negotiation in this layer to come up with the best offer. Negotiation in this layer takes place by creating two agents each using a different criterion for choosing the best possible component. For example one agent orders the components that match the offer according to their prices, and the other agent orders them according to their performance, thus the outcome of the two agents' negotiation would be really the best component out of the many components that match the specification required.

4-3 The Design of the Simulation

The Simulation is designed using Object Oriented design. The implementations of the two examples are simulated by creating separate instances of different agent classes, such that there is a class for manufacturers, another for vendors and a third for desktop components in the Desktop Configuration Domain. And a class for the Company, another for the Company Criteria and a third for the travel components in the Travel Planning Domain. Objects communicate through function calls.

Negotiation is managed by a managing agent, which takes the negotiating agents' offers and directs negotiation among them by calling their functions. As to the negotiation levels, what concerns us here is the negotiation to get the best components either between

manufacturers for the same vendor or between different vendors, as well as the final negotiation to compile the several desktop components with maintaining their constraints. We are not concerned about accessing the Internet or the search engines to come up with a list of suitable vendors for each component. Implementation of agents is carried out using Borland Jbuilder [Hamouda00], which is an Integrated Development Environment for Java. The simulator written in Java reads information about components from text files. Information in text files should come from the web where information is written in Extended Markup Language (XML) file format. Sample XML files have been generated and validated using Document Type Definitions (DTDs) which is a file that can accompany a document, essentially defining the rules of the document, such as which elements are present and the structural relationship between the elements. Then displayed on the web using the Extensible Stylesheet Language (XSL) file format, which can be used to transform XML for display including converting XML to well-formed HTML. The tool used to write XML, DTD, and XSL files, is the Wattle software's XMLwriter [Hamouda00]. XMLwriter is a powerful XML Editor, designed to help web application programmers take advantage of the latest XML and XML-related technologies such as XSL and XQL. XMLwriter provides users with an extensive range of XML functionality such as: validation of XML documents against a DTD or XML Schema, and the ability to convert XML to HTML using XSL stylesheets. The XML files are read by an extractor program written in Microsoft Visual Basic, which can read XML files using the Document Object Model (DOM) in which is a recommendation providing a standard for programmatic access to structured data through scripting, so developers can consistently interact with and compute on XML-based data.

5-Conclusion

The main contribution in this paper is that using existing INTERRAP's model of negotiation is not sufficient in many domains, which require more diverse negotiation protocols than the ones supported in INTERRAP. There are many domains, which need more protocols and a more structured design of how to apply these protocols. Thus a negotiation hierarchy is introduced for diverse domains, those who requir one protocol only, and the ones that require a layering of several negotiation protocols to approach an agreement. The successful implementation of the Desktop Configuration domain and the Travel Planning domain simulation, evidence the appropriateness of the proposed approach for the design of a hierarchy of negotiation protocols in real world domains.

6. Bibliography

[1] Hammouda, Nehal (2000). "A New Hierarchical Negotiation Scheme ontop of a Multi-Agent Architecture", The American University in Cairo, M.Sc., 2000.

[2] Lander, Susan Ellen. (1994). "Distributed Search and Conflict Management among Reusable Heterogeneous Agents". PhD thesis, *University of Massachusetts Am*herst.

[3] Muller, Jorg P. (1996). "The Design of Intelligent Agents: A Layered Approach". Published by *Springer-Verlag Berlin Heidelberg New York.*

[4] Sandholm, Tuomas.(1989). "An Implementation of the Contract Net Protocol Based on Marginal Cost". Technical Report*, http://mas.cs.umass.edu/index.html.*

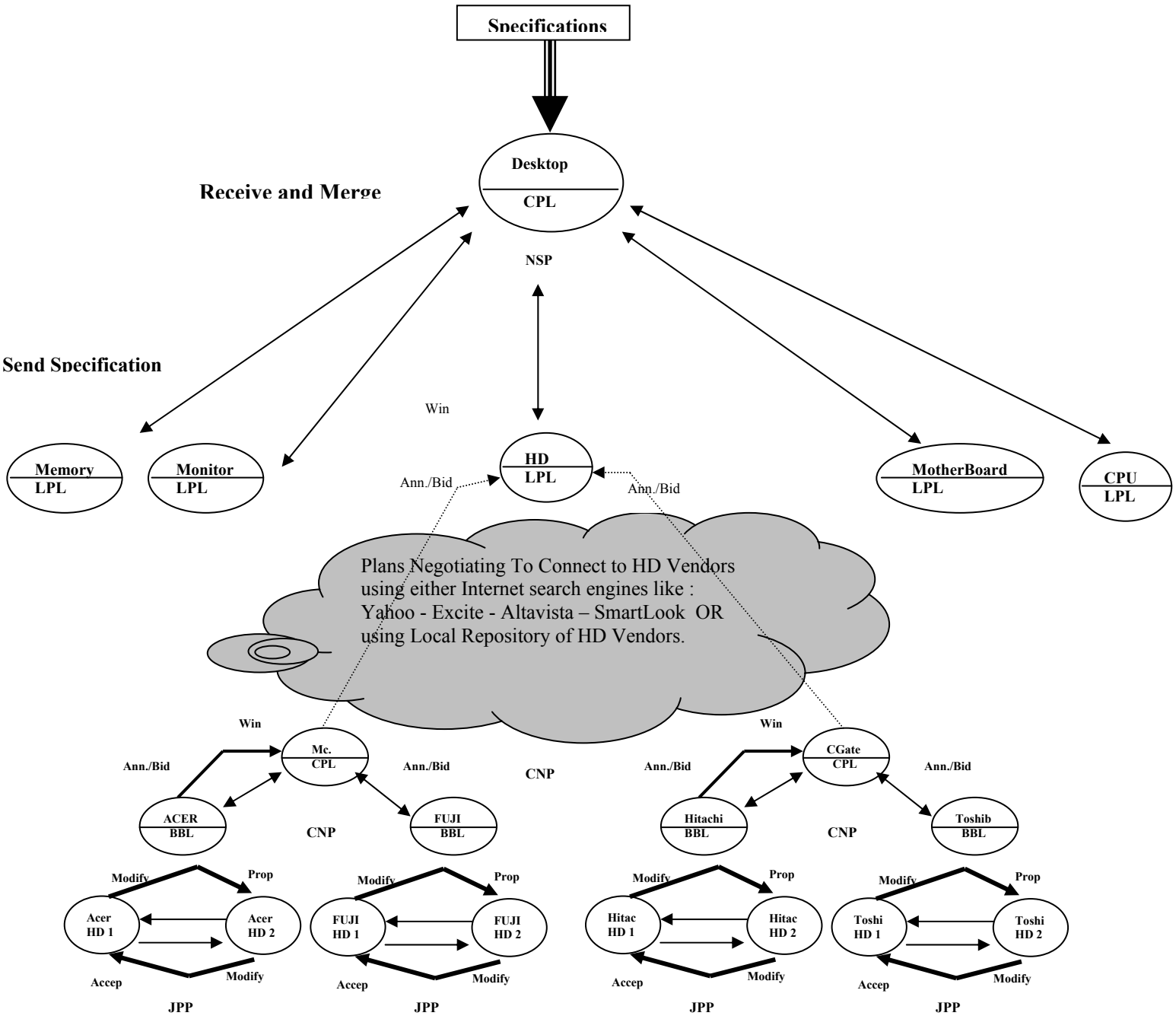

Figure 1: JPP, CNP, CSP & NSP Negotiations are Used to Come Up With a Desktop Offer