Systemics and Informatics World Network
Volume 13 ▪ December 2012 ▪ pp. 37-52
sai: siwn.0013.2012.064

ISSN 2044-7272 (print)
ISSN 2051-5634 (online)

Semantic Enhanced Argumentation Based Group Decision Making for Complex Problems

Haibo Jia

Chapter 4 Agent-Based Support System Architecture for the Semantic Enhanced Argumentation Based Group Decision Making

Abstract: In this chapter, different roles in the group argumentation and interaction among them are identified. An agent based approach is employed to analyse and model the function and behaviour of those roles in the different stages of collaborative decision making. The knowledge representation for individual agent and communication protocols for group agents are discussed. As result, an agent-based support system architecture for the semantic enhanced argumentation based group decision making is proposed. Based on it, an open, distributed and automatic group argumentation based decision making support software system can be developed.

Keywords: group argumentation based decision making data model, semantic representation of argumentation ontology, group argumentation based decision making system architecture, multi-agent based interaction modelling.

4.1 System Overview

In group decision making, the decision task must be first defined. Around the task, group members propose their solution and reason or justification based on their knowledge and at the same time they also argue or question others' idea. During the group discussion, various solutions will be proposed and argumentation towards them will provide the evidence for evaluating their justification. Questions constantly raised by group members can decompose the initial task into small sub-tasks. Answers to these questions tend to construct the solution space from initial task to the sub-tasks. Thereafter, the group members will evaluate each solution or sub-solution based on the enumerated criteria and draw a conclusion.

The Conceptual framework for group decision making proposed in the last chapter has shown the basic blueprint of the function the system can offer. We intend to design a system which should provide an argumentation interface for each of the decision makers to allow them to make claims and debates following a certain argumentation model in order to exchange the information with other group members and stimulate each other to externalize their knowledge. This argumentation model should provide a logic and reasoning mechanism to support an automatic decision making and evaluation process. In order to better utilize the argumentation information, a knowledge representation model within a problem solving domain needs to be devised. Our proposed solution empowers the system via development of semantics and ontology, where parameters or criteria of decision making could be identified from the semantically annotated utterances and the argumentation based decision process can be reused in the similar context of the problem solving process. In addition, group decision support system should receive and process different inputs from different experts with distinct ontologies. Under such a semantic framework the software agent should better understand decision makers' utterances so that it can analyse decision makers' externalized knowledge and actively provide relevant information which may satisfy their information demand. The software agent can dynamically update decision makers' profile such as their credibility towards some topic based on their performance and others' response. The profiles of decision makers have already been identified as an important factor to evaluate their claim during group decision making in literatures (Indiamma, 2008; Xiong, 2008). We assume the semantic support group argumentation can structurally record decision makers' discussion contents around a certain decision task which can be utilized by a decision model to support group decision making, but also it can construct a group memory to capture the process of problem solving from problem domain to solution domain. The well-defined argumentation model can offer explanation and justification information towards a solution or opinion rather than only solution information. So the semantically enhanced group memory, namely formal knowledge representation of group argumentation process for decision making, can be intelligently searched by the software agent and reused in different granularity and aspects in a similar context. The argumentation process is a process of defeasible reasoning. The new evidence and argument will maintain or update a conclusion. The research has shown the evaluation of argumentation quality can indirectly reflect the justification of conclusion (Amgoud, 2009). If we assume the conclusion of the group decision making system is the final decision made by the group, the evaluation of argumentation should be the crucial part of decision function. In our system, we propose an argumentation evaluation based decision model and design an approach to retrieve the necessary parameters required for this model in the group interaction.

The system offers an open intelligent group decision making environment which is not restricted by the limitation of time and location. Simply, the role in the system can be identified as facilitator or expert. The facilitator is responsible to authenticate user, publish decision task, keep and disseminate discussion information, manage the process of group decision making. The experts are responsible to participate in the group argumentation, propose their opinions based on a well-defined argumentation model, actively or passively collect relevant information, and maintain their own profile. Facilitator and Experts can reside in the same location or distribute in the different places. The processes of decision making can be divided into discussion stage; criteria identified stage and decision making stage. Moreover, two implicated roles in the system are defined as knowledge agent and decision agent. The knowledge agent can exploit the domain knowledge model. Its duty is to annotate the experts' utterances with the formal concept in the domain model, analyse the experts' mental space and provide relevant information to the expert. The Knowledge agent is equipped with ontology based semantic processing functions such as ontology query, ontology reasoning and ontology authoring, so that it can deliver intelligent services for experts' information demands. The Decision agent is designed to analyse and evaluate experts' argumentation to output group decision according to the information provided by facilitator. The decision output should include all the solution and sub-solution with its weight accepted by a group of expert. The decision function can be reconfigured to adapt to different requirement.

To conclude, the main objectives of the system are:

(i) To aid the group to structure the decision problem
(ii) To support group communication in the decision process
(iii) To aid group to generate criteria for decision evaluation
(iv) To support group to evaluate the solution and generate structured resolution of group discussion.
(v) To support the record of contribution generated during the discussion process and reuse of other contribution in previous discussion.

4.2 System Architecture Design

Considering the openness, distribution and complexity of the system, we adopt the agent oriented paradigm to design the system. Agent-Oriented Programming (AOP) essentially models an application as a collection of components called agents that are characterized by autonomy, proactivity and the ability to communication. Agents can independently carry out complex and often long-term task and take the initiative to perform a given task without stimulus from an external user. They can also interact with other agents or service entities to achieve their own or others' goals. In the system overview described above, we have identified 4 agents in our system. They are expert agent, facilitator agent, knowledge agent and decision agent. The system architecture is shown as

Figure 4.1. Agents are located in their own agent containers which can be distributed in different physical places and can achieve some certain functionality by calling the underlying basic service block in the Domain Service Layer. Agents also can communicate with each other by sending messages to complete a task collaboratively. For example, the expert agent can externalize the experts' mental space from their utterance and expose it to knowledge agent, and knowledge agent will search knowledge bases including current and past discussion repository to return the relevant knowledge the expert may be interested in.

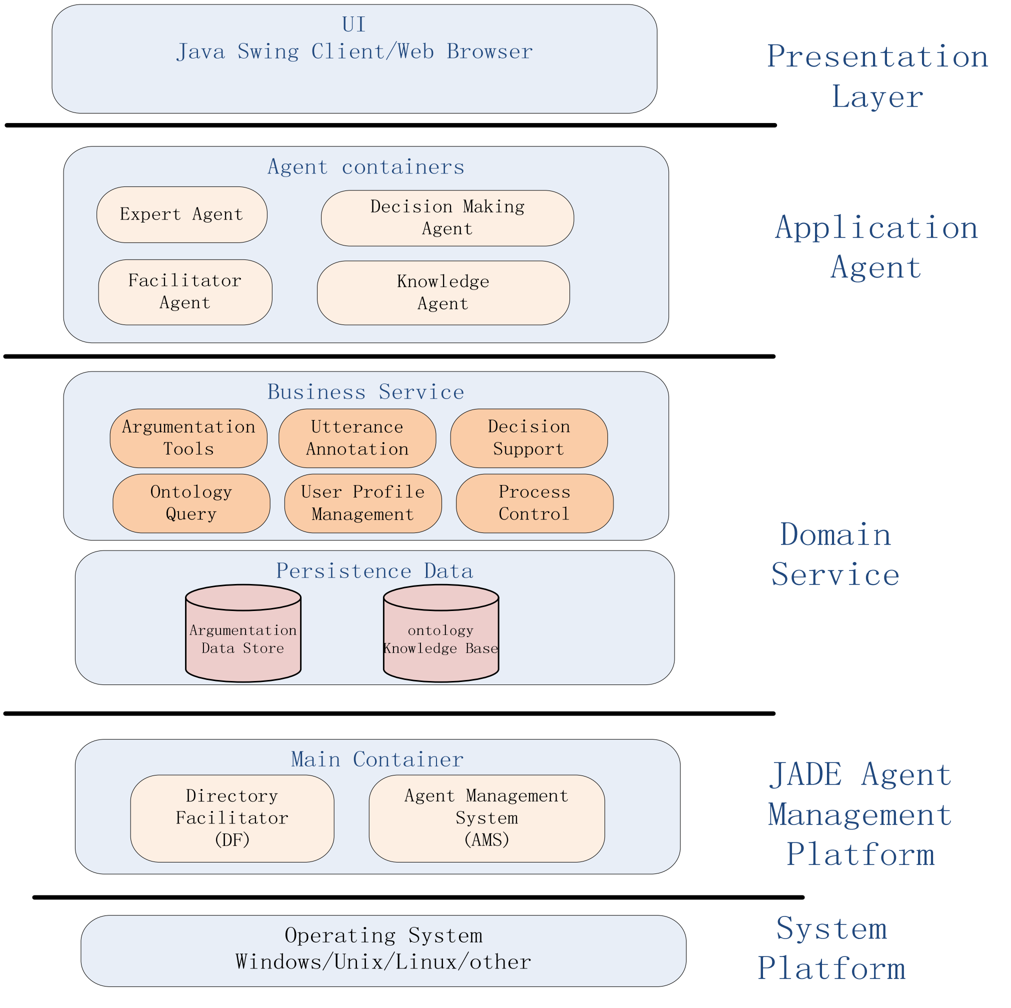

Figure 4.1 Group Argumentation Based Decision Making System Architecture

In the Domain Service Layer, the basic service function blocks are provided for application agents to fulfil their requirement. Different services can be mapped to different agents. For example, argumentation tools and user profile management can be used by an expert agent to participate in group argumentation and maintain its profile according to its performance in the group. Utterance annotation and ontology query can be utilised by the knowledge agent to provide knowledge based semantic service such as knowledge creation and knowledge reuse etc. Facilitator agent can use process control service to control the different stages of the group decision making such as group argumentation, voting, and decision. Decision support service which includes the evaluation of utterance and decision function can also be used by decision agent.

In our system, we employ JADE (Java Agent Development Framework) (Bellifemine, 2007) as the agent management platform, which provides an infrastructure to solve domain-independent issues such as agent communication, agent management and agent discovery etc. JADE is a software platform which is fully compliant with the FIPA specification and implements software agent abstraction over a well-known object-oriented language, Java, to provide a simple and friendly API. The main container of JADE resides on the top of operating system, which represents the bootstrap point of a platform. The main container includes 2 agents which respectively are Agent Management System Agent (AMS) and Directory Facilitator Agent (DF). AMS manages the life cycle of other agents and implements the white pages service which is used by any agent wishing to register its service and search for other available services. Application agents can also subscribe other agents via DF agent so that they can be notified whenever a service registration or modification is made that matches some specified criteria. This is called the yellow pages service. In the group argumentation system, the expert can join or leave the discussion at any time. The agent platform can assure the expert agent aware of other experts' existence and properly send the utterance message to all online experts.

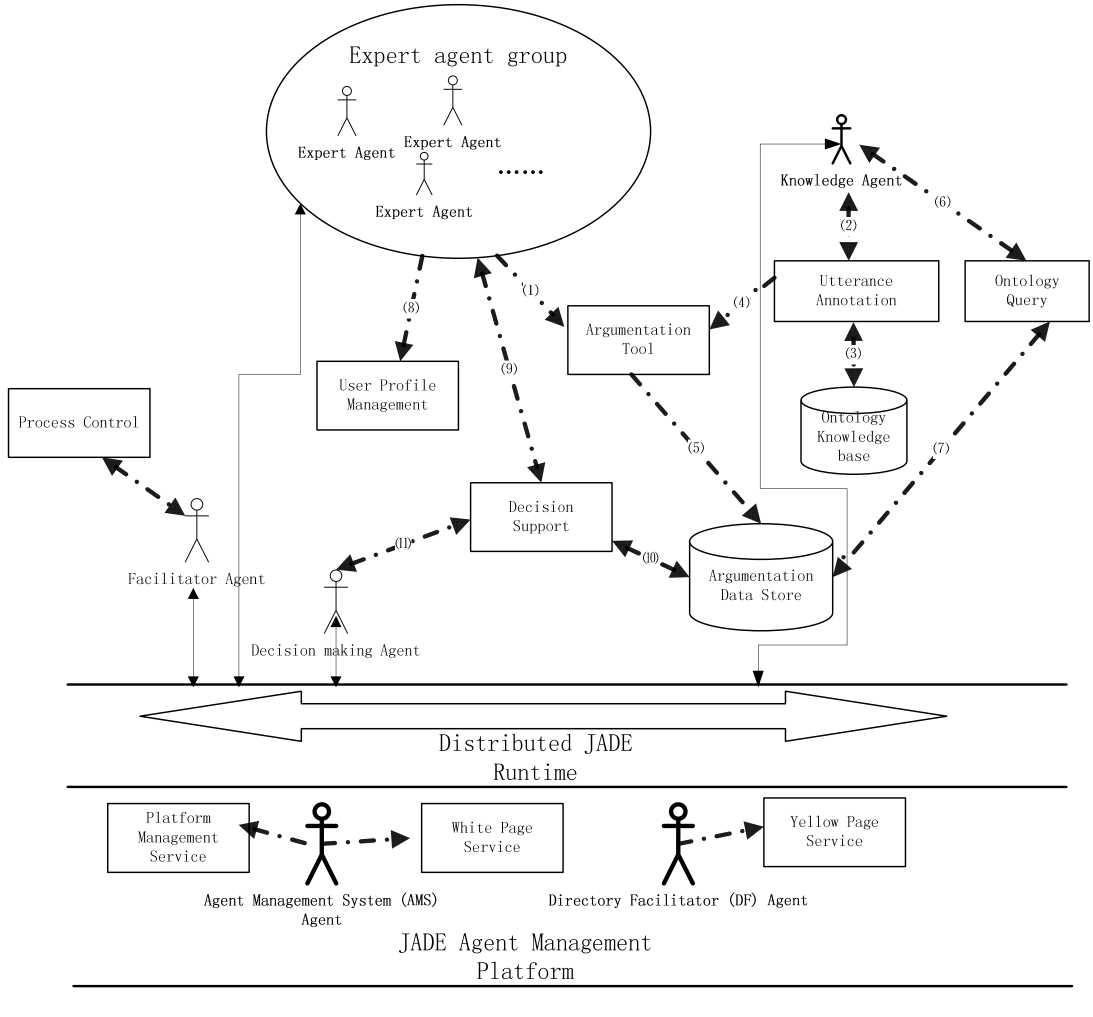

Figure 4.2 The relationship diagram among the subsystems/toolkits

In Figure 4.2, the working relationship among the agent, service and platform are depicted, which shows how the agents achieve the group decision making task via collaboration in the above designed system architecture. As the illustration in Figure 4.2, application agents reside above the distributed JADE runtime environments and exchange message via them. And also they use the different services or tools to complete their tasks. The whole process of the agents' collaboration can be divided into different 3 different stages --- discussion stage; criteria identified stage and decision making stage. The number in Figure 4.2 shows the order of agents' action using related service, which can be described as the following:

- The Expert agents use argumentation tool to submit their utterance (Step 1);
- The Knowledge agent uses the annotation service and ontology knowledge base to annotate expert agent's utterance with ontological concepts; (Steps 2, 3 & 4)
- Ontological concept enhanced argumentative utterances are stored in the argumentation data store; (Step 5)
- The knowledge agent uses ontology query service to answer the expert agent's query or actively analyse the expert agents' interest via the expert agent's argumentation and its context; (Steps 6, 7)
- The expert agent uses user profile management service to update their credibility based on others' response to its argumentation; (Step 8)
- Expert agents provide the preference of the identified criteria, their credibility which are aggregated by the decision support service; (Step 9)
- The Decision support service evaluates the structured argumentation contents annotated by semantic ontology to make recommendation towards the final decision and identify the further problem to be considered; (Steps 10 & 11)
- The facilitator agent uses process control service to schedule the different stages of the whole decision making process. (throughout all steps)

Two special system level agents (AMS agent & DF agent) sit in the JADE management platform, which provide system level support for agent's registration and subscription and also for agent runtime environment to manage the application agents' life cycle (start, kill, etc).

4.3 Data Model Design

4.3.1 Data Model Overview

In chapter 3, a conceputal argumentation schema has been proposed. Following this schema, the decision oriented group argumentative information can be structurally recorded . Based on this schema, a simplified data model for the group argumentation based decision making is outlined in Figure 4.3. This model shows the information which will be captured during the discussion and decision making process in the group. In this data model, the discussion element is an abstraction of group members' utterance which comprises the question utterance, the argumentation utterance and solution/idea utterance. The design of classification of utterance is inspired by the IBIS information model. It has been regarded as a useful model to capture the rationale of complex and unstructured problem solving (Conklin, 1998).

The group memory organised by this model will disclose the group rationale for certain decision task and also can be utilized by the agent software to automatically analyse and evaluate group members' contribution and draw a conclusion in a certain structure. Argument can be further decomposed into the practical argument and the epistemic argument. The practical argument aims to justify the solution and is built from both beliefs and preferences/goals; however the epistemic argument aims to justify beliefs and is based only on beliefs. In the context of group argumentation, one proposed solution may have multiple challenge/support practical arguments. The aggregation of practical argument evaluation will directly affect the judgement about the solution. Likewise, the aggregation of epistemic argument will indirectly affect the judgement of relevant solution and directly affect the quality of practical argument. For different type of arguments, there are different approaches to evaluate them and it has been discussed in Chapter 3. The question utterance can raise a question towards any discussion elements. However, the solution/opinion utterance only can be made towards decision task or question utterance.

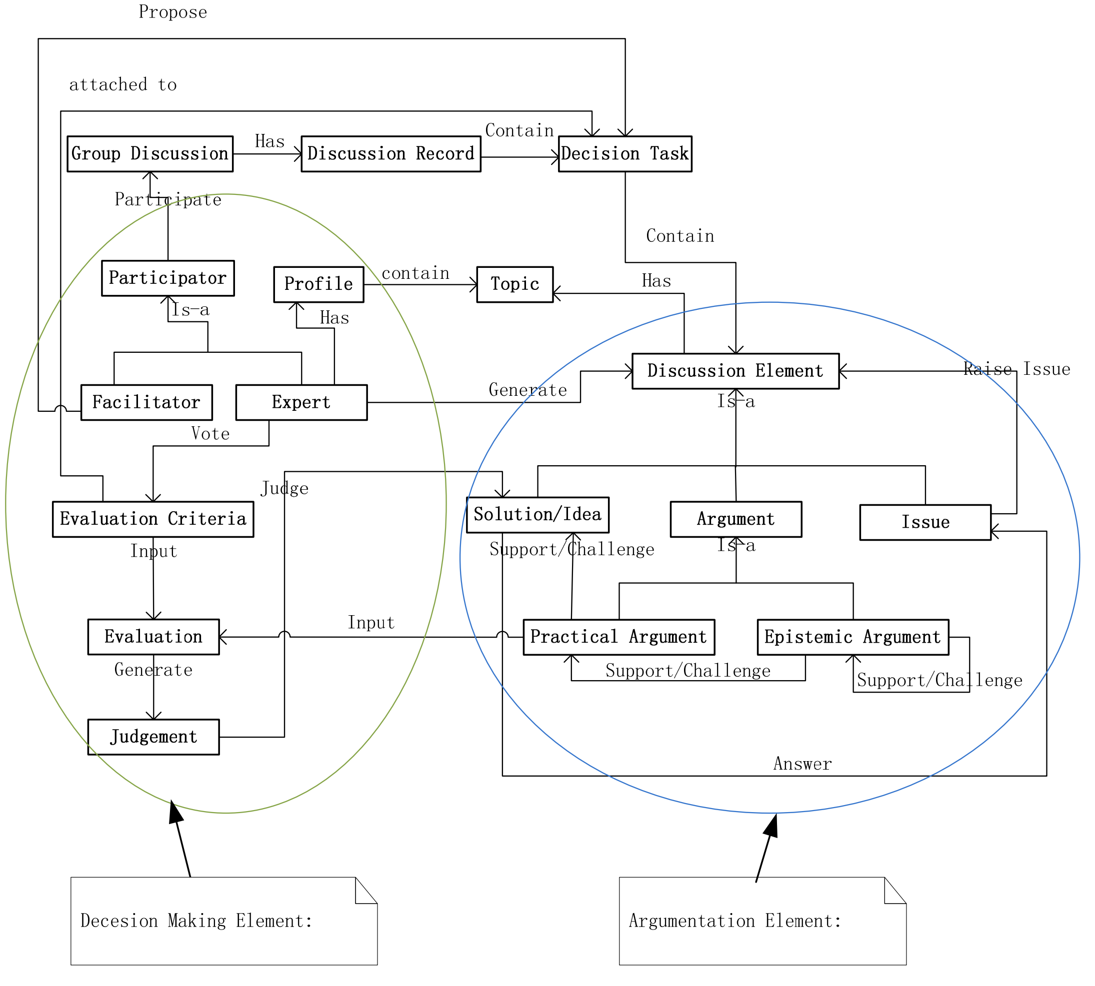

Figure 4.3 Group Argumentation Based Decision Making Data Model

In this data model the participants can be defined as facilitator and experts, who collaboratively carry out discussion tasks. A group of experts not only contribute the discussion element based on their knowledge and desire but also collectively vote to generate criteria for evaluating the decision. Based on evaluation of the argument, the system can automatically produce the judgement for each solution or sub-solution. The judgement will reflect the ordering of a solution among the whole solution sets. So this data model records both the structured group discussion contents and the judgement of the decision and justification of the judgement process.

As shown in Fig. 4.3, a rectangle block represents the abstract concept of the domain and a link represents the relationship between concepts. The relationship can further be divided into semantic relationship such as “raise question”, “challenge/support” etc. and subsumption relationship such as “Is-a”. Although the populated data are finally stored in a central database, during the group discussion the agents maintain part of data such as their own utterances and updated profile so that they can analyse experts’ intention and provide more semantic service. In order to enable agent better interpret and query the argumentative information, a formal representation of the data model should be developed. In the proposed system, we use an ontology language -- OWL[1] -- to represent the ontology and encode the data by which the concept and relationship between them can be formally defined.

4.3.2 Semantic Representation of Argumentation Ontology

Preliminaries. Ontology is the formal specification of concepts and relationship among the concepts (Gruber, 1993). An ontology is a set of classes C, properties P and instances i. Generally, concepts have three types of relation among them: subClassOf, disjointWith, and equivalentClass. The semantic scope (SC) of a concept (class) C_i is represented as $(SC(C_i))$. The definitions of these three types of relations are:

- subClassOf: $SC(C_1) \subseteq SC(C_2)$, /* the semantic scope of C_1 is narrower than that of C_2
- disjointWith: $SC(C_1) \subseteq \neg SC(C_2)$, /*the semantic scope of C_2 is disjoint with that of C_1.
- equivalentClass: $SC(C_1) \equiv SC(C_2)$, /*the semantic scope of C_1 is equivalent with that of C_2.
- In addition, we also define semantic relation as:
- $(\forall i_1,i_2)P(i_1, i_2) \wedge i_1 \in C_1 \wedge i_2 \in C_2$, /* state that i_1 relate with i_2 through the property P and domain (P) $\subseteq C_1$ and Range (P) $\subseteq C_2$.

Define concepts through the class. In the ontology, the key concepts of the argumentation domain are defined through the class.

- $\{Argument, Idea, Issue, Task\} \subseteq Utterance$
- $\{\mathrm{Pr}\,atical\ \ \mathrm{arg}\,ument, \mathrm{Epistemic\ argument}\} \subseteq Argument$

All the classes disjoint with others.

- Argument $\subseteq \neg$Issue, Idea $\subseteq \neg$Issue, etc.

Define concepts through owl construct. The concept defined by owl construct can be used for consistency check or class inference.

- $\mathrm{Pr}\,actical\ \ \mathrm{arg}\,ument \equiv \mathrm{sup}\,port\ \ \exists\ \ (Idea\ \ )\ \ \vee challenge\ \ \exists\ \ (Idea)$

where ($\exists$ = hasValue). It can be read as the practical argument can be defined as a concept which either support idea or challenge idea.

- $Epistemic\ \ \mathrm{arg}\,ument \equiv \mathrm{sup}\,port\ \ \exists\ \ (Argument) \vee challenge\ \ \exists\ \ (Argument)$

It can be read as the epistemic argument can be defined as a concept which support other argument or challenge other argument.

- $Idea \equiv\ \ resolve\ \ \exists\ \ (Issue\ \ \vee\ \ Task)$

It can be read as the idea can be defined as a concept which can resolve the issue or task.

- $Issue \equiv raise\_issue\ \ \exists\ \ (Idea\ \ \vee\ \ Argument \vee\ \ Issue)$

It can be read the issue can be defined as a concept which can raise issue to the idea or argument or other issue.

Consistency check based on axioms can be exploited by the agent to identify the class of the utterance. If the user doesn't specify the category of the utterance, the reasoner can infer the right class which this utterance belongs to. So the semantic definition of the concept can assure the utterance information can be formally structured and well annotated by predefined concept, which will be ready for the agent to do the further analysis and make decision based on the defined decision function.

Define property through domain and range. In the ontology, a property often relates two instances of the classes. Properties have a domain and range. Syntactically, domain links a property to a class and range links a property to either a class or a data range (Smithe, 2004). Table 4.1 only lists object properties with the class range used in the ontology. Other data properties with the simple data range can be referred from data structure of utterance in Section 3.2.3.

[1] http://www.w3.org/TR/2004/REC-owl-features-20040210/

Table 4.1 Properties list of argumentation ontology

Property name	Domain	Range
Resolve	Idea	Issue $\vee$ Task
Support	Argumentation	Idea $\vee$ Argument
Challenge	Argumentation	Idea $\vee$ Argument
raise_issue	Issue	Idea $\vee$ Argument $\vee$ Issue

Realizing concepts through instances. The real utterance contents of a practical group argumentation are defined through the instances which belong to the classes. Figure 4.4 demonstrates a simple snippet of realizing concepts about traffic congestion control discussion. There are 4 instances which respectively belong to issue, idea and practical argument class. Each instance is annotated by a formal topic term predefined in the domain ontology. The annotation approach will be discussed in detail later.

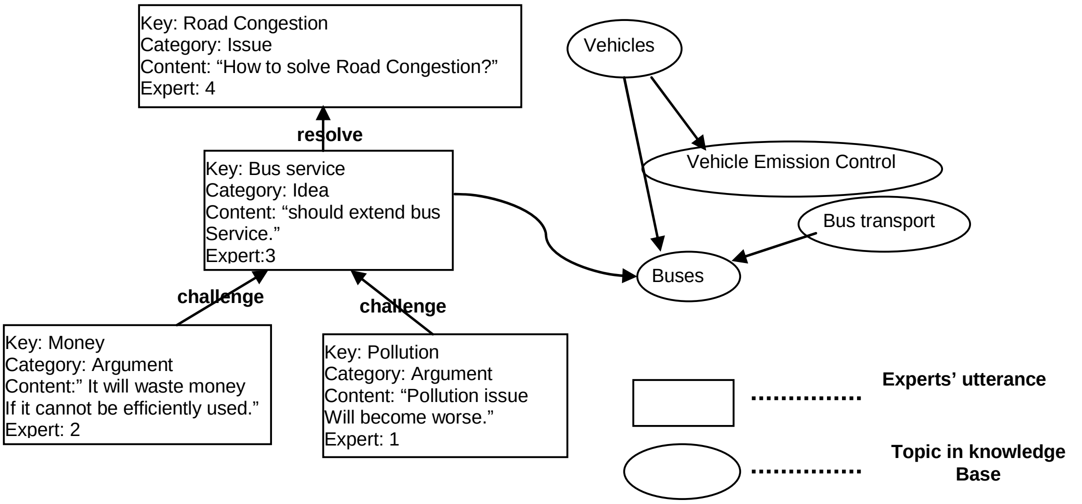

Figure 4.4 Exemplified instances of Group Argumentation model

In this discussion snippet example, expert 2 proposes the argument about 'economic' issue (money) to challenge expert 3's idea about 'bus service' aspect. This piece of information can be encoded in OWL as in Figure 4.5.

The expert agent maintains a local copy of their own utterances, through which agent can analyse the expert's mental space and intelligently provide more relevant information via the knowledge agent. It can complement experts' mental space and stimulate experts to contribute more utterances to the group discussion which may cover more aspects of the decision task and better for the final decision making. In above example, expert2 agent can represent knowledge as:

- Practical_Argument (utterance1)
- Has_topic(utterance1, Topic:Economic)
- Challenge(utterance1,utterance2)
- Idea(utterance2)
- Has_topic(utterance2, Topic:Buses)

"Topic:" means the namespace of domain ontology in which the hierarchical concept terms are defined. As Figure 4.4 shown, concept "Buses" has 3 super concepts. So the expert agent will remind the expert if there are any other utterances about concept "Buses" or "Economic" and their sub or super concepts coming. Particularly if any others propose an utterance with the similar pattern as

```
<Practical_Argument rdf:ID="utterance_1">
    <has_topic rdf:resource="#economic_1"/>
    <challenge rdf:resource="#utterance_2"/>
   <has_creator rdf:resource=”#expert_2” />
   <has_content rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
       It will waste money if it cannot be efficiently used.
   </has_content>
</Practical_Argument>

<Idea rdf:ID="utterance_2">
   <has_topic rdf:resource="#buses _1" />
   <resolve rdf:resource=”issue_1” />
  <has_creator rdf:resource=”#expert_3” />
   <has_content rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
         find a solution to deal with congestion growth?
  </has_content>
</Idea>

<Topic:Economic rdf:id="economic_1" />
 <Topic:Buses rdf:id=”buses_1” />

<owl:Class rdf:ID="Idea">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Utterance"/>
    </rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:ID="Practical_Argument"/>
   <rdfs:subClassOf>
      <owl:Class rdf:ID=”Argument” />
          <rdfs:subClassOf rdfs:resource="#Utterance />
       </owl:Class>
     </rdfs:subClassOf>
     <owl:disjointWith  rdf:resource="#Idea" />
  </owl:Class>
```

Figure 4.5 OWL encoding snippet

“Challenge (Topic:Economic, Topic:Buses)” or opposite pattern as “Support (Topic:Economic, Topic:Buses)”, the expert3 agent will suggest expert3 to highly concern about them. Because those utterances may reflect the same or different views regarding some aspect of decision task which the expert3 is interested in or have sufficient knowledge on that. In addition, via a knowledge agent the expert agent can query previous discussion information base by the constraint of topic and pattern to reuse the problem solving process of similar task. These can be represented as the following:

- find all the utterances related with the exact topic T:

$$hasTopic(?X, T)$$

- find all the utterance related with the topic T or its subtopic:

$$hasTopic(?X,T) \vee (hasTopic(?X,?O) \wedge subClassOf(?O,T))$$

- find all the argumentation whose pattern satisfy Topic1 challenge Topic2

$$Challenge(?X,?Y) \wedge hasTopic(?X,T1) \wedge hasTopic(?Y,T2)$$

- find all the argumentation whose pattern satisfy Topic1 or its subtopic challenge Topic2 or its subtopic

$$\mathrm{Challenge}(?X,?Y) \wedge \mathrm{hasTopic\_all}(?X,T1) \wedge \mathrm{hasTopic\_all}(?Y,T2)$$
$$\mathrm{hasTopic\_all}(?X\ ,T) \rightarrow \mathrm{hasTopic}(?X,T) \vee (\mathrm{hasTopic}(?X,?O) \wedge \mathrm{subClassOf}(?O,T))$$

- find all the argumentations which act on Topic1 or its subtopic and Topic2 or its subtopic

$$\mathrm{Rel}\ (?X,?Y) \wedge \mathrm{hasTopic\_all}(?X,T1) \wedge \mathrm{hasTopic\_all}(?Y,T2)$$
$$\mathrm{hasTopic\_all}(?X\ ,T) \rightarrow \mathrm{hasTopic}(?X,T) \vee (\mathrm{hasTopic}(?X,?O) \wedge \mathrm{subClassOf}(?O,T))$$
$$\{\mathrm{Challenge, Support, Raise\_issue, Resolve}\} \subseteq \mathrm{Rel}$$

This pattern based semantic reminding service and knowledge reuse service can stimulate the expert to further think about more related aspects of the task.

4.4 System Interaction Design

In the system architecture design, multi-agent approach has been proposed to deal with the distributed and openness issue. System agents and application agents have been described as two kinds of agents. Generally, the system agents are responsible for maintaining and managing the application agent and functions of them have been implemented by agent platform. So in our system interaction design, we are mainly concerned about modelling interaction between application agents, and investigating the internal state transition of each agent.

Based on the analysis about the process of group argumentation based collaborative decision making, different roles of agents have been identified. Among them expert agent, facilitator agent, knowledge agent, and decision agent are the most crucial one. Intelligence, design and choice have been commonly regarded as the 3 steps of decision making problem. Our agent driven approach is also guided by this convention. In the role definition of the designed agents, the knowledge agent is responsible for related knowledge and information provision; this is more like intelligence activity. The expert agent is responsible for constructing an argumentation tree, identifying the weight of their own utterance and voting for criteria of solution; from certain sense it is a design process of decision making. And the decision agent utilizes different strategies to aggregate experts' utterance and evaluates the solution based on the strength and credibility of argumentation and experts' profile; thereafter give choice recommendations. The facilitator agent's duty is to manage the workflow of the group argumentation based decision making which includes publishing a decision task, managing the process of the group discussion, formulating the criteria by group voting, forming the group decision. The facilitator agent represents the meeting facilitator in a real group meeting and it sends messages to the other agents to indicate in which different stages they are during the decision making process.

4.4.1 Agent Based System Interaction

In a multi-agent system, agents cooperate with others to achieve a task. It is crucial to model the functionality of the individual agent and interaction between them. As shown in Figure 4.6, an agent interaction sequence is depicted. For simplicity, the agents' interactions in the different stages are described in one diagram. They include registration of agent, agent based group argumentation, agent based criteria generation and agent based decision making.

In the stage of registration of agents, all the agents register in the Directory Facilitator Agent with its id, address and capability description in order that other agent can find them when required. After the facilitator agent registers, it publishes the decision task and subscribes the notification of expert agents' registration from the Agent Manager. Once an expert agent registers, the Agent Manager notifies the facilitator agent with the newly registered agent's address. The facilitator agent will send the decision task and current utterance list to the new expert agent. After the expert agent receives the information from facilitator, it will notify the UI agent to update UI with received information. The Subscription service can offer the agents ability to dynamically update their acquaintance table as soon as other agents register or deregister in case that they have to repeatedly discover other agents from Directory Facilitator. Similarly, expert agents also subscribe the registration of other expert

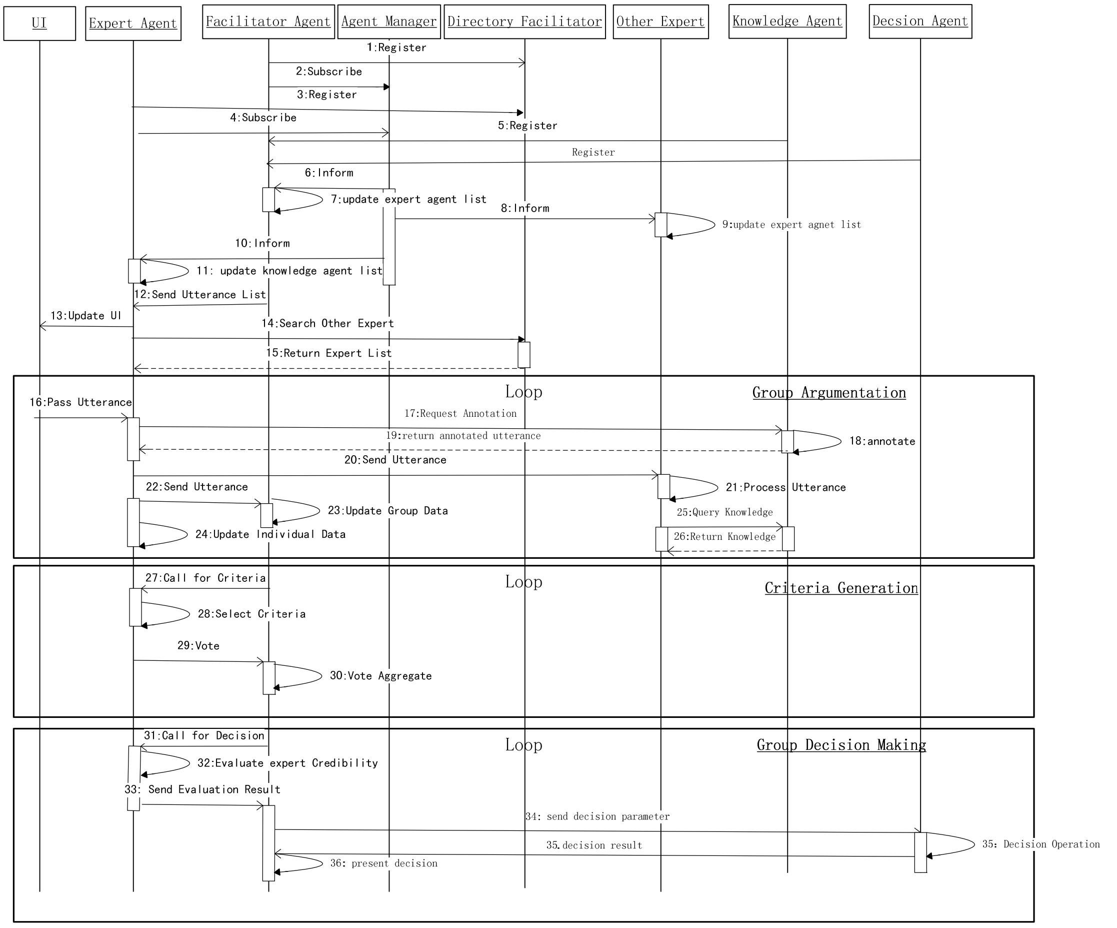

Figure 4.6 Agents interaction sequence for Group Argumentation

agents and knowledge agent. So the agent manager will notify all the existing expert agents when a new expert agent registers, at the same time the new agent request Directory Facilitator agent to retrieve the existing expert agents and facilitator agent. This mechanism can assure that the newly registered agent can obtain other agents' information before the subscription service is available. The expert agent cannot succeed in the registration before the facilitator agent has registered and published the decision task. It means that the facilitator must register and assign the decision task first and then group members can register for the group argumentation.

In the stage of group argumentation, group experts can exchange utterance by the discussion interface. The Expert agent receives the utterance from UI agent; the following interaction among multi-agents will be occurred:

(1) expert agent request annotation from knowledge agent;
(2) knowledge agent search external knowledge base and use a domain concept term to annotate the utterance and send back the annotation to the expert agent;
(3) The expert agent sends annotated utterance to other expert agents and facilitator agent;
(4) Other expert agents process incoming utterance and pass it to UI agent to present it; they also remind their user if incoming utterance is relevant to this user's utterances; in addition expert agent may update the expert's profile by the certain rule if received argumentation targets its own utterance;
(5) The Facilitator agent backup each incoming utterance to a group memory;

(6) The expert agent (utterance proponent) will update its own utterance list by which the expert agent can analyse the expert's mental space and proactively request knowledge agent for more semantic knowledge and even reuse similar utterance in other discussion.

In the stage of criteria generation, the facilitator agent calls for criteria by sending message to all expert agents. The message will include all of concept terms existed in the group memory. This concept term collection represents various aspects of the decision task produced in the group argumentation and will be the candidates of criteria for solution evaluation. The expert agents receive message and pass to the UI agent to visualize those information. The expert user can work on that to rank those concepts using different weights based on his preference. Finally, the criteria terms with the weight value from different expert agents will be sent back to the facilitator agent and aggregated to produce the group level criteria.

In the decision making stage, after the expert agent receives the decision making request from the facilitator agent, it starts to evaluate its own credibility towards the different domain of interest; this domain based credibility of the expert agent will reflect the trustworthiness of related utterances. Generally the expert agents' credibility is dynamically updated during the argumentation, and it is decided by agents' initial credibility and other agents' response to their utterance. The expert agents' credibility is sent back to Facilitator agent as one of decision making parameter for final decision operation. The facilitator agent collects decision parameters, in our case which include the weight of all criteria, all expert agents' domain based credibility and the strategy for decision making. Those parameters are sent to the decision making agent for decision operation. The decision function has been defined in Chapter 3. The decision making agent will work on the whole argumentation dataset; the result of decision is that each utterance will be assigned with one of the specified states such as "solved, unsolved, unverified" for the issue utterance and "accepted, rejected, unverified" for idea and the argumentative utterance with a weight value. The idea utterances with the higher weight value are the most possibly accepted solutions; and unsolved issue utterances are the possible new decision task to be considered in the next sessions. After the decision operation, the decision making agent sends the decision result back to the facilitator agent who will present it to the group members.

4.4.2 Agent Behaviour Modelling

In the above section, an agent based system interaction has been described. In this section we will describe the internal state and action of each agent. The agents have various states based on the different stage in the group activities. Under the different agent's state, they have different actions to respond to incoming or internal message. As Fig. 4.7 to Fig. 4.10, different agents' behaviour models are shown. Here we use a finite-state automata diagram to model agents' action and state transition. The agent can detect the change of environment by incoming message and decide in which states it goes. In the certain state, the agent takes corresponding action according to the message and the rule it has.

In Figure 4.7, the expert agent has four various states which are registered state, discussion state, voting state and decision making state. The state can be switched from registered state to discussion state once upon the decision task message is received. Expert agent will take different action when it is in various states. For example, in discussion state agent will judge different type of message and choose proper action such as update utterance list, analyse utterance to provide reminding service or update expert profile etc. This modelling approach can improve the automation of the agent.

In Figure 4.8, facilitator agent has similar set of states but having different corresponding actions. According to the role of the facilitator agent, it more concerns about the process management and information aggregation. In the criteria formation and decision making states, its action is to aggregate the local contribution from each individual expert agents.

In Figures 4.9 and 4.10, the knowledge agent and decision making agent are illustrated. The knowledge agent takes action in the discussion state, which can respond to different message type to annotate utterance and make ontology based query. However, the decision making agent only takes action in the decision making stage, which can receive the decision parameter and evaluate all utterances based on the predefined decision function.

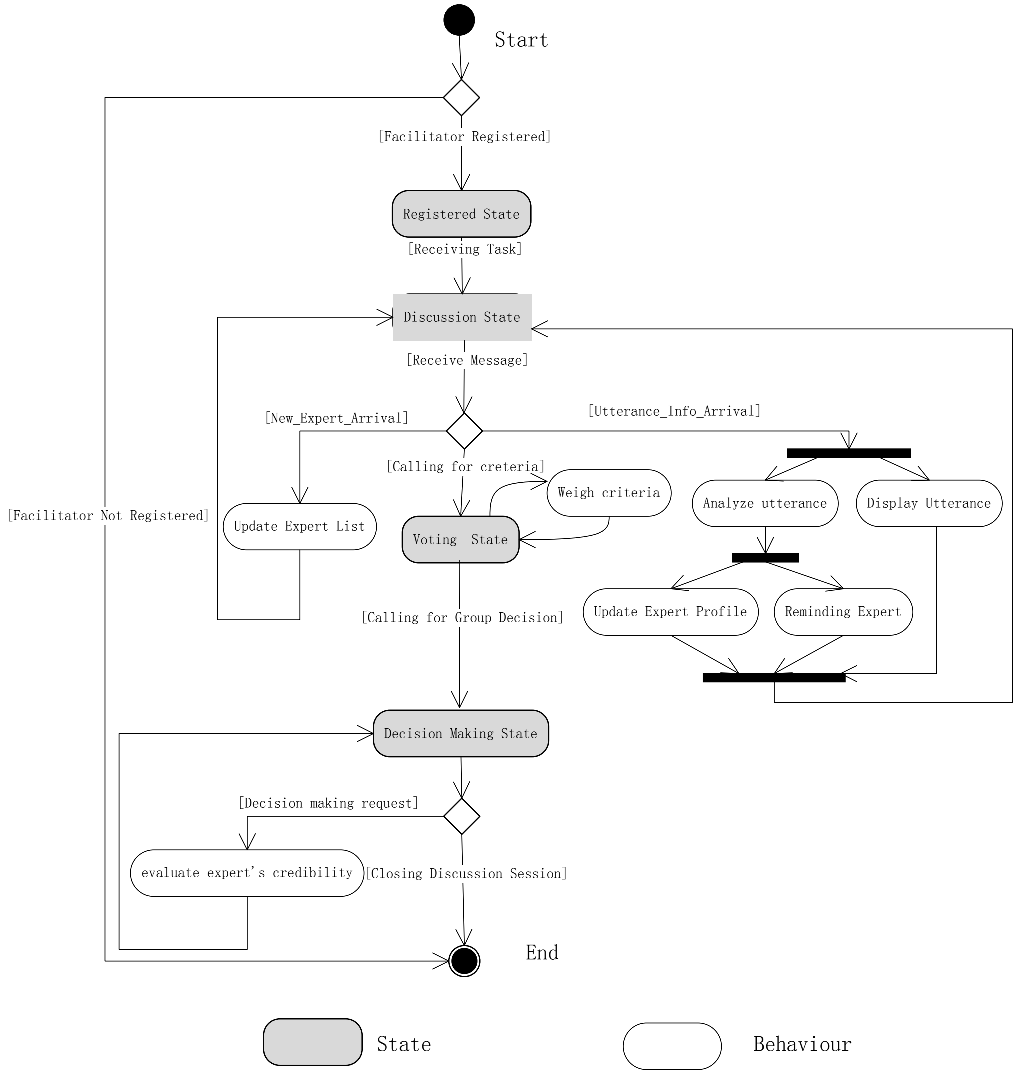

Figure 4.7 Expert agent activity diagram

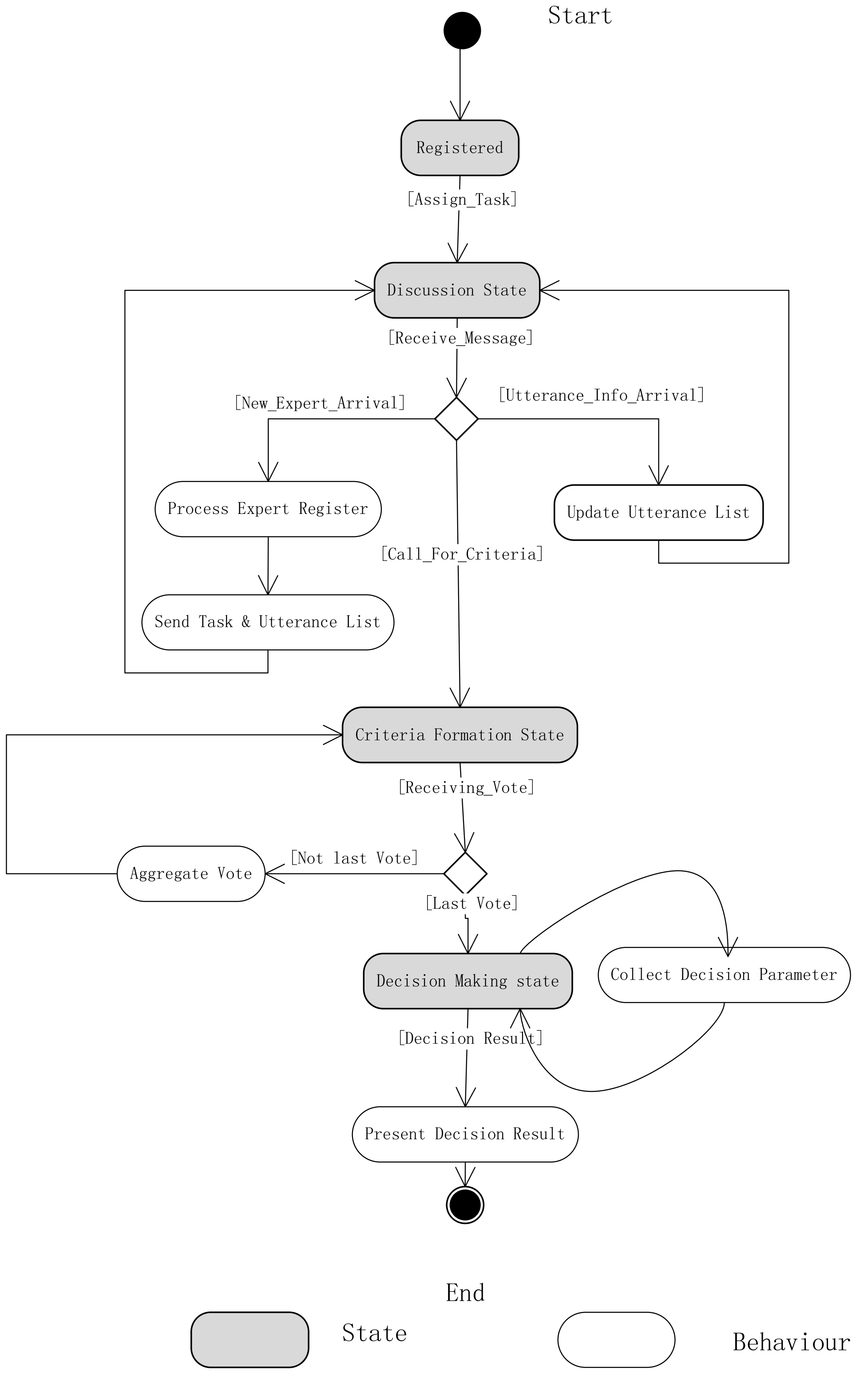

Figure 4.8 Facilitator agent activity diagram

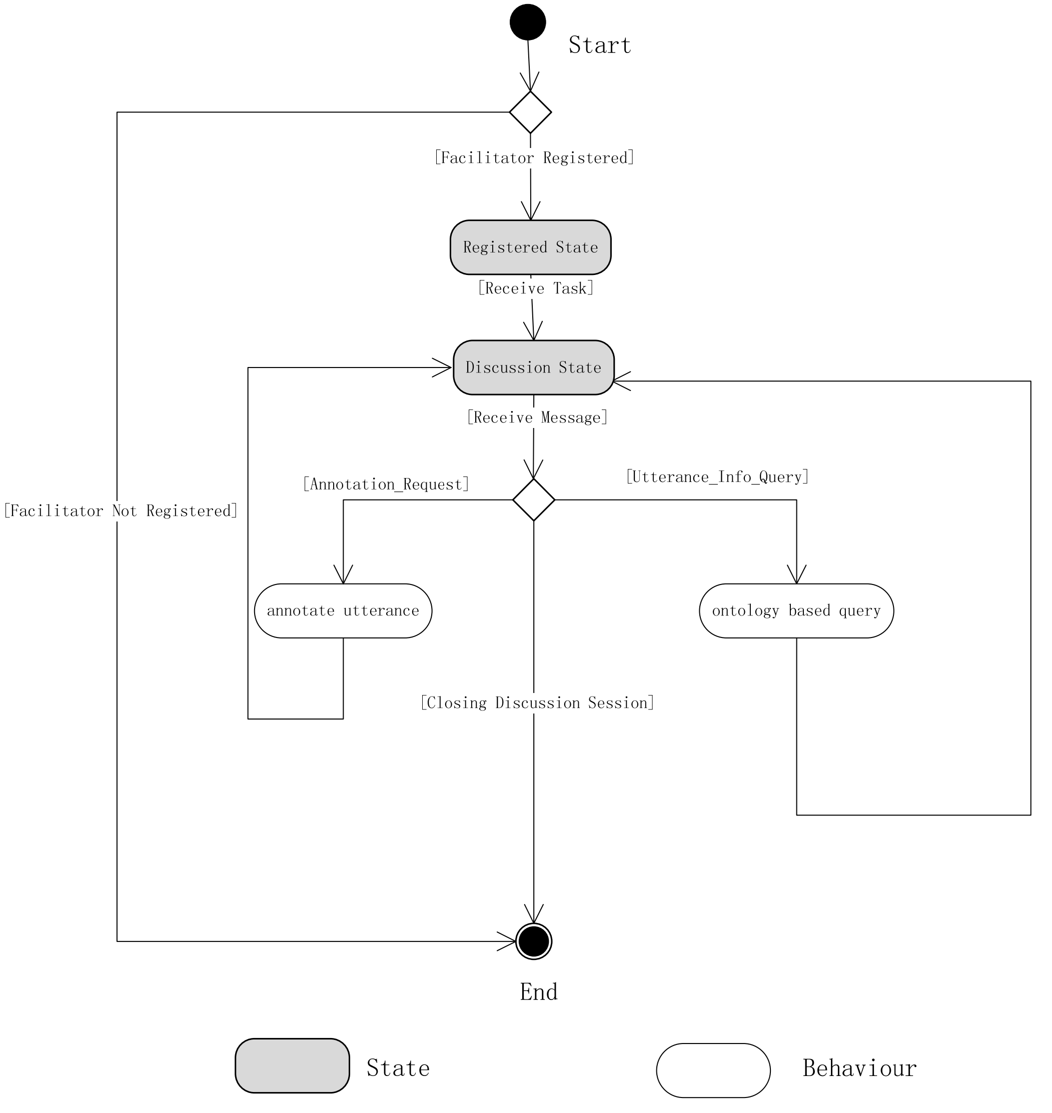

Figure 4.9 Knowledge agent activity diagram

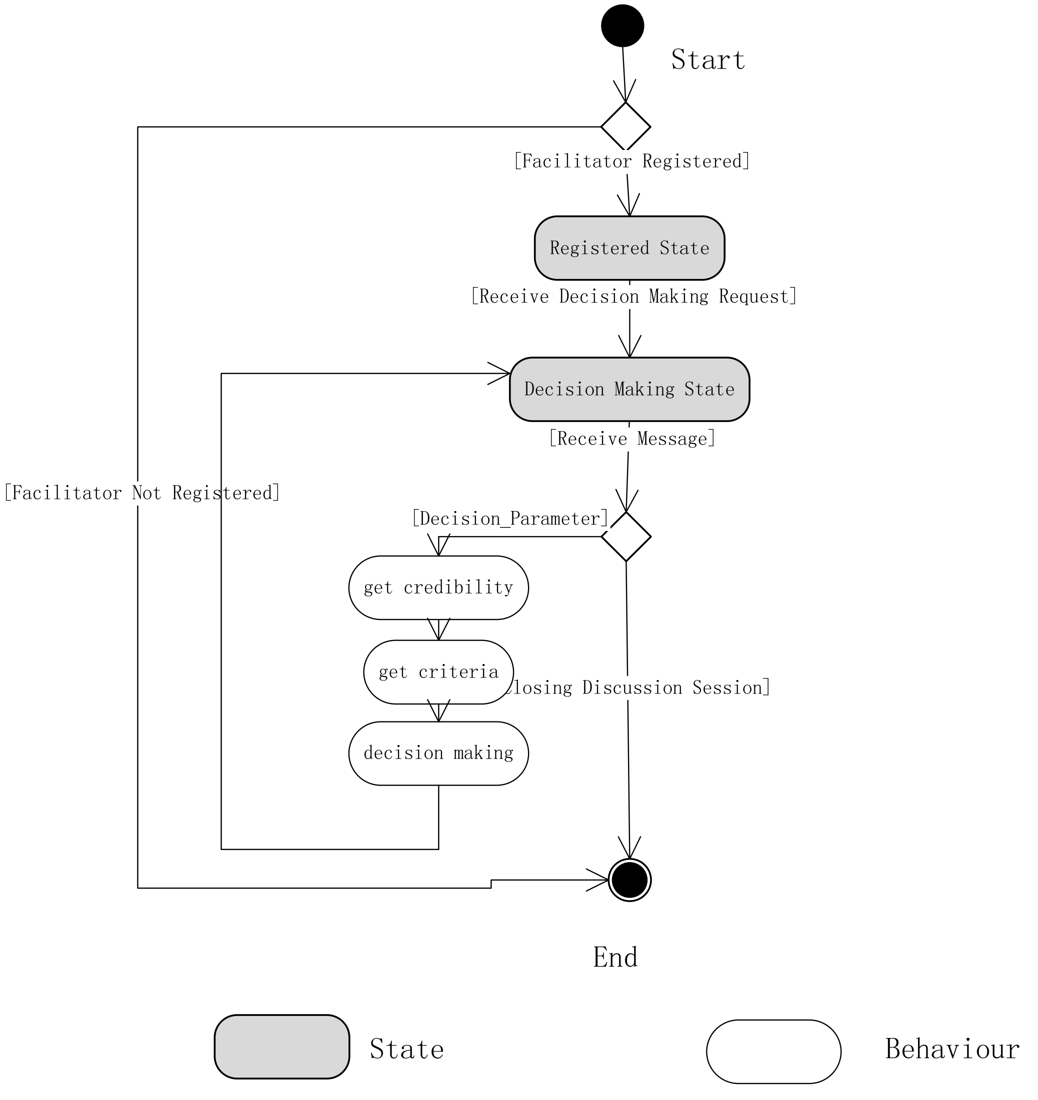

Figure 4.10 Decision making agent activity diagram