A Review and Development of Agent Communication Language

Moamin Ahmed, Mohd Sharifuddin Ahmad, Mohd Zaliman Mohd Yusoff
College of Information Technology
Universiti Tenaga Nasional
Kajang, Malaysia
sharif@uniten.edu.my

Abstract—In this paper, we review the research findings in agent communication language (ACL). We then propose our model to develop an agent communication language based on the Foundation for Intelligent Physical Agents (FIPA) Standards. The use of communicative acts enables agents to recognize the intent of the requesting agents for a specific service. The ACL handles the semantics of information exchanged between agents by the use of ontologies defined for the domain as specified by FIPA. We test the communication language in a collaborative process involving three agents.

Keywords—software agents; agent communication language; collaborative work; ontology.

I. INTRODUCTION

Agent communication has been an area of intense research along with the development of multiagent systems [3, 8, 14]. The role of communication in a multiagent system is to provide a means of exchanging information based on an agreeable set of rules or protocol of sending and receiving messages. Upon receiving a message, an agent must be able to decipher the meaning of the message and respond accordingly to produce a coherent structure of coordination between agents to achieve a shared goal. An agent needs to communicate with other agents to commit to some performance of actions for those agents. In distributed systems of agents, communication can be achieved in five ways [19]:

No communication: This is possible if agents can infer each other's plan, or when a structured community of agents is designed obviating the need for communication, or when agents reside in pre-established harmony, where every agent act on a specific set of pre-defined actions.

Signaling: The activities of agents are synchronized via semaphore-like signals.

Message passing: Many distributed systems (e.g. object-oriented systems) operate on message passing.

Plan passing: This is a specific technique for coordinating multiagent activity, used in partial global planning paradigm.

Speech acts: The contents of a communication are meaningful statements about the environment, knowledge or actions.

In this paper, we review the research findings in agent communication language (ACL). In particular, we look at two major development in ACL, i.e. the Knowledge Query and Manipulation Language (KQML) and the Foundation for Intelligent Physical Agents (FIPA) ACL. We then propose a model for the development of an agent communication language and develop a prototype based on the FIPA ACL standards for agent communication involving three agents in a collaborative process.

Section II of this paper dwells on the related work in agent communication. In Section III, we present our approach to the development of agent communication language. Section IV discusses the development and testing of the communication language and Section V concludes the paper.

II. RELATED WORK IN AGENT COMMUNICATION

In the following subsections, we review and analyze some important aspects on the development of an agent communication language. We use the outcome of our analysis to conceive a model to develop an agent communication language for a collaborative process using the FIPA Standards.

A. The Speech Act Theory

Philosophers and linguist developed the speech act theory to model human communication but Computational Linguistics and Artificial Intelligence researchers exploited such theory to model communication between software agents. Austin suggests that the role of languages in communication is to impart actions [14]. Speakers do not simply utter sentences that are true or false, but rather perform speech actions such as requests and suggestions. Consequently, all utterances are speech acts, i.e., they are actions of some sort.

The speech act theory [14, 18] considers three aspects of utterances. Location refers to the act of utterance itself. Illocution refers to the 'type' of utterance as in a request to turn on the heater. It conveys the speaker's intentions to the listener. Perlocution refers to the effect of an utterance, i.e., how it influences the recipient. For example, the utterance "Check the document" is a locution. Its utterance is the illocution of a request to check a document. Perlocution is when the listener actually checks the document.

Most ACLs in used today are based on illocutionary speech acts. The illocutionary verbs (e.g. request, tell) in a natural language typically correspond to performatives in ACL. The theory is consistent with the mentalistic notion of agents in that the message is intended to communicate attitudes about information such as beliefs, goals, etc.
B. The Knowledge Query and Manipulation Language (KQML)

The Knowledge Query and Manipulation Language (KQML) is another important milestone of ACL, developed as part of the DARPA Knowledge Sharing Effort (KSE) [3, 13]. It is a high-level communication language and protocol for exchanging information and knowledge and support run-time knowledge sharing among agents.

The KQML message structure consists of three layers: content, message and communication layers. The content layer contains the actual content of the message specified in any language. The message layer consists of the set of performative parameters provided by the language. Performatives in KQML are grouped into nine categories, each define the allowable speech acts that agents may use. They specify whether the content is a query, an assertion or any of those allowable speech acts that agents may use. They specify permitted action for request(s, h, φ), the preconditions are:

- s believe h can do φ
  - you do not ask someone to do something unless you believe they can do it
- s believe h believe h can do φ
  - you do not ask someone to do something unless you believe they can do it
- s believe s want φ
  - you do not ask someone unless you want it

Similarly, the post-condition for such request is that:

- h believe s believe s want φ
  - the effect is to make them aware of your desire

A completion condition indicates the final state, after possibly a conversation has taken place and the intention associated with the performative that started the conversation is fulfilled.

C. The Foundation for Intelligent Physical Agents (FIPA)

FIPA is an IEEE Computer Society standards organization that promotes agent-based technology and the interoperability of its standards with other technologies. FIPA proposes a standard for an ACL based on the speech act theory, which is quite similar to KQML [4].

A FIPA ACL message contains a set of one or more message parameters. The use of these parameters needed for effective agent communication varies according to the situation. The mandatory parameter in all ACL messages is the performative, although most ACL messages also contain the sender, receiver, and content parameters. Users have the flexibility to include user-defined message parameters other than that specified by FIPA. If an agent does not recognize or is unable to process one or more of the parameters or parameter values, it replies with the not-understood message. The following is an example of a FIPA ACL message.

```
(disconfirm
  :sender (agent-identifier :name i)
  :receiver (set (agent-identifier :name j))
  :content "((mammal shark))"
  :language fipa-sl)
```

FIPA uses the Semantic Language (SL) to formally define the semantics of its ACL [6], otherwise KQML and FIPA ACL are almost identical with respect to their basic concepts and the principles they observe.

D. Multiagent Systems Architecture

Genesereth and Ketchpel [8] approach the development of a multiagent systems architecture based on the communication aspect of agent coordination. Two different approaches have been explored to facilitate agent coordination: direct communication, in which agents handle their own coordination, and assisted coordination, in which agents rely on special system programs to achieve coordination.

In the direct communication approach, two techniques have been implemented: contract net and specification sharing. Due to the disadvantages of cost and implementational complexity entailed by the direct communication approach, a popular alternative is to opt for a federated system (see Figure 1).

In a federated system, agents do not communicate directly with each other but via a service agent called a facilitator. Agents use an ACL to register their needs and abilities with their local facilitators. This information constitutes the agent’s meta-level information.
E. Other Agent Communication Techniques

While KQML and FIPA ACL epitomize agent communication, many researchers have developed other techniques of agent communication. Payne et al. [16] propose a shallow parsing mechanism that provides message templates for use in message construction. This approach alleviates the constraint for a common ACL between agents and support communication between open multiagent systems.

Chen and Su [12] develop Agent Gateway which translates agent communication messages from one multiagent system to an XML-based intermediate message. This message is then translated to messages for other multiagent systems. Pasquier and Chaib-draa [15] offer the cognitive coherence theory to agent communication pragmatic. The theory is proposed as a new layer above classical cognitive agent architecture and supplies theoretical and practical elements for automating agent communication. The incoherence and utility measures defined within the cognitive coherence framework provide the necessary mechanisms to resolve the issues plaguing agent communication.

F. Conversation Policy

Greaves et al. [9] discuss the role of conversation policies in agent communication, and suggest several subtypes of conversation policy. They suggest that the use of language by an agent is no different from any other action that an agent might take. An agent’s production of a message is always the result of a plan to bring about some identified state in the world.

Their reasoning suggests that conversation policies are best modeled as sets of fine-grained constraints on ACL usage - the individual constraints are presumed to only address a single feature of a particular conversation. These constraints then define the computational process models that are implemented in agents.

G. Ontology

The term ontology was first used to describe the philosophical study of the nature and organization of reality [11, 12]. In AI it is simply defined as “an explicit specification of a conceptualization” [10]. This definition provokes many controversies within the AI community especially with regard to the meaning of conceptualization. An ontology associates vocabulary terms with entities identified in the conceptualization and provides definitions to constrain the interpretations of these terms.

Most researchers concede that an ontology must include a vocabulary and corresponding definitions, but there is no consensus on a more detailed characterization [12]. Typically, the vocabulary includes terms for classes and relations, while the definitions of these terms may be informal text, or may be specified using a formal language like predicate logic as implemented in [7].

FIPA ontology uses a specification of a representational vocabulary for a shared domain of discourse involving definitions of classes, relations, functions, and other objects [5].

III. A Proposed Development Model

Our approach to develop the agent communication language follows a sequential process. The following subsections discuss these sequential steps and identify the required information exchanges used in the process.

A. Select a suitable Domain

In selecting a suitable domain for the development of the ACL, we consider the following issues:

- The processes in the domain must exhibit clear interactions between at least two agents.
- The number of interactions between any two agents must not be too excessive.
- The agents perform specific tasks based on the interactions.

Based on the above considerations, we select a collaborative process involving at least three agents. We use the Examination Paper Preparation and Moderation Process (EPMP) of an academic faculty to develop an agent communication language.

EPMP has three agents - Examination Committee (C), Moderator (M) and Lecturer (L). The goal of this collaborative process is to complete the moderation of examination papers.

B. Analyze the Domain Processes

The process starts when the Examination Committee sends out an instruction to start prepare examination papers. The Lecturer then prepares the examination paper, together with the solutions and the marking scheme (Set A). He then submits the set to be checked by an appointed Moderator. The Moderator checks the set and returns them back with a moderation report (Set B) to the Lecturer. If there are no corrections, the Lecturer submits the set to the Examination Committee for further actions. The Lecturer and Moderator
are given deadlines to complete the process, as Table 1 below shows:

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Deadlines</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set A should be submitted to the respective moderators</td>
<td>Week 10</td>
</tr>
<tr>
<td>1st moderation</td>
<td>Week 10 &amp; 11</td>
</tr>
<tr>
<td>2nd moderation (Set B)</td>
<td>Week 12 &amp; 13</td>
</tr>
<tr>
<td>Set B should be submitted to EC</td>
<td>Week 14</td>
</tr>
</tbody>
</table>

C. Identify the Tasks and Communication Exchanges

From the analysis of the process, we draw a diagram of communication exchanges as shown in Figure 2, to represent the sequence of interactions involved in the EPMP. We then identify all the tasks including the communication exchanges each agent has to perform.

We represent the tasks and the communication exchanges for each agent as $T_X^#$ and $E_X^#$ respectively, where $#$ is the task or communication exchange number and $X$ refers to the agents C, M, or L. A message from an agent is represented by $\mu_{SR}^#$, where $#$ is the message number, $S$ is the sender of the message $\mu$, and $R$ is the receiver. $S$ and $R$ refer to the agents C, M, or L. We reserve the need to label the tasks, communication exchanges and messages for our future use in one-to-many and many-to-many message exchanges [4].

The following interactions show the tasks and message exchanges between the three agents to complete the moderation process. The interactions do not include those activities for which human interventions are required. Due to space limitation, we omit the sequences for paper corrections and remoderation.

(a) Agent C:
$E_{1C}^1$: C sends a message $\mu_{1CL}^1$ to L – PREPARE examination paper.

(b) Agent L:
$E_{1L}^1$: L replies with a message $\mu_{1LC}^1$ to C – ACKNOWLEDGE.

(c) Agent M:
$E_{1M}^1$: M replies with a message $\mu_{1ML}^1$ to L – ACKNOWLEDGE.

(d) Agent L:
$E_{3L}^1$: L replies with a message $\mu_{3LM}^1$ to M – ACKNOWLEDGE.

(e) Agent C:
$E_{2C}^1$: C replies with a message $\mu_{2CL}^1$ to L – ACKNOWLEDGE.

D. Study and Analyze FIPA Specifications

FIPA specifications deal with ACL messages, message exchange interaction protocols, speech act theory-based communicative acts and content language representations. The references we used for the development are as follows [4, 5]:

![Figure 2. The EPMP Communication Exchanges](image-url)
SC00061G - FIPA ACL Message Structure Specification
XC00086D - FIPA Ontology Service Specification

The FIPA ACL Message Structure Specification defines a set of message parameters for our domain [4]. The FIPA Ontology Service Specification assumes that two agents, who wish to converse, share a common ontology for the domain of discourse, i.e. the agents ascribe the same meaning to the symbols used in the message [5]. We describe further analysis and compliance to these FIPA specifications in Section IV.

E. Select suitable Software and Agent Platforms

To implement the EPMP, we use Win-Prolog and its extended module Chimera, which has the ability to handle multiagent systems. We use Prolog for two reasons: Firstly, Prolog is well suited for expressing complex ideas because it focuses on the computation’s logic rather than its mechanics where the drudgery of memory allocation, stack pointers, etc., is left to the computational engine. Reduced drudgery and compact expression means that one can concentrate on what should be represented and how. Secondly, since Prolog incorporates logical inferencing mechanism, this powerful property can be exploited to develop inference engines specific to a particular domain.

Chimera provides the module to implement peer-to-peer communication via the use of TCP/IP. Each agent is identified by a port number and an IP address. Agents send and receive messages through such configurations.

IV. ACL DEVELOPMENT AND TESTING

We develop the agent communication language for the multiagent system of EPMP based on the above model and test the implementation in a laboratory environment on a Local Area Network. Each of the agents C, M and L runs on a PC connected to the network. In the test, communication is simplified and executed based on the tasks outlined in Section III.C.

For our message structure, we use the parameters shown in Table II. We include the performatives, the mandatory parameter, in all our ACL messages. We also define and use our own performatives in the message structure, which are Prepare, Check, Remind, Review, Complete, Pass, Modify, and Acknowledge.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Category of Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>performative</td>
<td>Type of communicative acts</td>
<td>The type of the communicative act of the ACL message</td>
</tr>
<tr>
<td>sender</td>
<td>Participant</td>
<td>The identity of the sender of the message, that is, the name of the agent of the communicative act.</td>
</tr>
<tr>
<td>receiver</td>
<td>Participant</td>
<td>The identity of the intended recipients of the message.</td>
</tr>
<tr>
<td>reply-to</td>
<td>Participant</td>
<td>Indicates that subsequent messages in this conversation thread are to be directed to the agent named in the reply-to parameter.</td>
</tr>
</tbody>
</table>

Table II. FIPA ACL Message Parameters Used

To complete the message structure, we include the message, content and conversational control parameters as shown in Table II. However, not all the FIPA parameters are used in our implementation [4].

As an illustration, we reproduce below the sample codes that implement a communicative act of requesting the Lecturer agent to Prepare an examination paper (see (a) in Section III.C):

```prolog
committee_dialog_handler((Agent,1003),msg_button,_,_):-
  committee_get(Agent,ontology,Ontology),
  committee_get(Agent,language,Language),
  committee_get(Agent,toagent,ToAgent),
  committee_get(Agent,performative,Performative),
  committee_get(Agent,inreplyto,InReplyTo),
  committee_get(Agent,replywith,ReplyWith),
  committee_get(Agent,goal,Goal),
  committee_get(Agent,ontology,Ontology),
  % Performative
  Prepare =
  prepare([% the performative: - Prepare
            [':sender',Agent,':receiver',Lecturer,
            ':reply-with',open_new_word_document,
            ':content',Performative([':sender',Agent,':reply-with',ReplyWith,
            ':in-reply-to',InReplyTo],Language),
            ':ontology',Ontology,':language',Language],
            % the message
            ':content',Goal),
  agent_post(Agent,Link,Prepare).
```

To implement the EPMP, we use Win-Prolog and its extended module Chimera, which has the ability to handle multiagent systems. We use Prolog for two reasons: Firstly, Prolog is well suited for expressing complex ideas because it focuses on the computation’s logic rather than its mechanics where the drudgery of memory allocation, stack pointers, etc., is left to the computational engine. Reduced drudgery and compact expression means that one can concentrate on what should be represented and how. Secondly, since Prolog incorporates logical inferencing mechanism, this powerful property can be exploited to develop inference engines specific to a particular domain.

Chimera provides the module to implement peer-to-peer communication via the use of TCP/IP. Each agent is identified by a port number and an IP address. Agents send and receive messages through such configurations.

IV. ACL DEVELOPMENT AND TESTING

We develop the agent communication language for the multiagent system of EPMP based on the above model and test the implementation in a laboratory environment on a Local Area Network. Each of the agents C, M and L runs on a PC connected to the network. In the test, communication is simplified and executed based on the tasks outlined in Section III.C.

For our message structure, we use the parameters shown in Table II. We include the performatives, the mandatory parameter, in all our ACL messages. We also define and use our own performatives in the message structure, which are Prepare, Check, Remind, Review, Complete, Pass, Modify, and Acknowledge.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Category of Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>performative</td>
<td>Type of communicative acts</td>
<td>The type of the communicative act of the ACL message</td>
</tr>
<tr>
<td>sender</td>
<td>Participant</td>
<td>The identity of the sender of the message, that is, the name of the agent of the communicative act.</td>
</tr>
<tr>
<td>receiver</td>
<td>Participant</td>
<td>The identity of the intended recipients of the message.</td>
</tr>
<tr>
<td>reply-to</td>
<td>Participant</td>
<td>Indicates that subsequent messages in this conversation thread are to be directed to the agent named in the reply-to parameter.</td>
</tr>
</tbody>
</table>

Table II. FIPA ACL Message Parameters Used

To complete the message structure, we include the message, content and conversational control parameters as shown in Table II. However, not all the FIPA parameters are used in our implementation [4].

As an illustration, we reproduce below the sample codes that implement a communicative act of requesting the Lecturer agent to Prepare an examination paper (see (a) in Section III.C):

```prolog
committee_dialog_handler((Agent,1003),msg_button,_,_):-
  committee_get(Agent,ontology,Ontology),
  committee_get(Agent,language,Language),
  committee_get(Agent,toagent,ToAgent),
  committee_get(Agent,performative,Performative),
  committee_get(Agent,inreplyto,InReplyTo),
  committee_get(Agent,replywith,ReplyWith),
  committee_get(Agent,goal,Goal),
  committee_get(Agent,ontology,Ontology),
  % Performative
  Prepare =
  prepare([% the performative: - Prepare
            [':sender',Agent,':receiver',Lecturer,
            ':reply-with',open_new_word_document,
            ':content',Performative([':sender',Agent,':reply-with',ReplyWith,
            ':in-reply-to',InReplyTo],Language),
            ':ontology',Ontology,':language',Language],
            % the message
            ':content',Goal),
  agent_post(Agent,Link,Prepare).
```

In the ontology development, we implicitly encode our ontologies with the actual software implementation of the agent themselves and thus are not formally published to an ontology service [5]. The sample codes below show the ontology implementation after the Lecturer agent (L), receives the Prepare message from the Committee agent (C) (see (b) in Section III.C):

```prolog
lecturer_handler(Name,Link,start([Args])):-
  fipa_member('sender',From,Args),
  fipa_member('reply-with',ReplyWith,Args),
  (!+lecturer_reply_remote_agent(From
    ; fipa_member('content',Content,Args),
```

In the ontology development, we implicitly encode our ontologies with the actual software implementation of the agent themselves and thus are not formally published to an ontology service [5]. The sample codes below show the ontology implementation after the Lecturer agent (L), receives the Prepare message from the Committee agent (C) (see (b) in Section III.C):

```prolog
lecturer_handler(Name,Link,start([Args])):-
  fipa_member('sender',From,Args),
  fipa_member('reply-with',ReplyWith,Args),
  (!+lecturer_reply_remote_agent(From
    ; fipa_member('content',Content,Args),
```
Content = Performative(|ContentArgs),

% Ontology calls
lecturer_message,
new_word_documents,
lecturer_form,
lecturer_reply(Name, From, ReplyWith, done, Reply),
agent_post(Name, Link, Reply)).

To test the ACL, we deploy human actors to perform the roles of Committee, Lecturer and Moderator. These people communicate with their corresponding agents to advance the workflow. An interface for each agent provides the communication between human actors and their agents (see Figure 3). Such interface and the subsequent communicative acts performed by agents and between agents contribute to the achievement of the shared goal.

In the test, agents performed the communication and the subsequent action to progress the workflow working together with the human actors. These cycle of communicative acts contribute to the completion of the moderation process.

![Figure 3. A Lecturer Agent Interface](image)

V. CONCLUSIONS AND FURTHER WORK

In this research, we developed and implemented an agent communication language, based on the FIPA Standards, to demonstrate the usefulness of the system to take over the timing and execution of communication from the human actors to achieve a shared goal. The important tasks, i.e. preparation and moderation tasks are still performed by the human actors. The agents perform communicative acts to other agents when the tasks are completed. Such acts could help reduce the cognitive stress of human actors in performing scheduled tasks.

The communication is based on one-to-one relationships, i.e. involving one Lecturer and one Moderator agents. In a real situation, there are many lecturer and moderator agents and each one of them communicates with many of the other. In our future work, we will study and analyze the technical issues involving one-to-many and many-to-many relationships between agents and implement the communication for such requirements. We will then develop a decision framework for agents to autonomously decide on initiating its own communication exchanges based on the states of the environment.

REFERENCES

[7] Fox M. S., Gruninger M., Ontologies and Enterprise Modelling, Enterprise Integration Laboratory, Dept. of Mechanical & Industrial Engineering, University of Toronto.