Intelligent Environment Architecture For Heterogeneous Applications

Magdy Aboul-Ela
Sadat Academy for Management Sciences
Computer and Information Systems Department
P.O. Box 2222, Kournich-ElNile, El-Maadi, Cairo, Egypt

ABSTRACT: - The Intelligent Environment consists of rich environments that allow access from multiple applications with varying capabilities. Context information is drawn from what the Intelligent Environment can sense about its internal resources and external environments. This context information relates to aspects of the environment that include, who is in the environment, what they are doing, what they have done, and what their actions are. This paper presents a proposed architecture of an Intelligent Environment for integrating heterogeneous applications based on methods for capturing such information, interpreting, aggregating and integrating it into increasingly more abstract structure. The functional and spatial distribution of tasks naturally lends itself to the existence of shared resources creates interactions over which the resources must coordinate.

KEYWORDS: - Intelligent Environment Architecture, Knowledge Representation, Multi-Agents systems, and Knowledge base.

1. Introduction

Multiagent System composed of multiple, interacting agents. An agent may have explicitly represented knowledge and a mechanism for operating on drawing inferences from its knowledge. An agent may also have the ability to communicate, which is the ability of perception (receiving of message) and action (sending of message). Communication is defined at several levels and there are basic message types such as: assertion and queries. Communication protocols are specified at several levels, method of interconnection, format, syntax, and meaning, [6], [7]:

Knowledge Query and Manipulating Language (KQML)
A fundamental decision for the interaction of agents is to separate the semantic of the communication protocol (which must be domain independent) from the semantics of enclosed message (which may depend on domain). All agents must universally share the communication protocol. KQML is a protocol for exchanging information and knowledge [6].

Knowledge Interchange Format (KIF)
Agents need descriptions of real-world things. The descriptions could be expressed in natural languages that are capable of describing a wide variety of things and situations. Symbolic Logic is a general mathematical tool for describing things. KIF, a particular logic language, has been proposed as a standard to use to describe things within expert systems, databases, intelligent agents, … etc. It is readable by both computer systems and people. It was specifically designed to serve as an "inter-lingua", or mediator in the translation of other languages. KIF is a prefix version of first order predicate calculus with extensions to support non-monotonic reasoning and definitions. KIF provides for encoding of knowledge about knowledge. KIF can also be used to describe procedures, i.e., to write programs or scripts for agents [6].

Ontology
Ontology is a specification of the objects, concepts, and relationships in an area of interest. For a given area, the ontology may be explicitly represented or implicitly encoded in an agent. More specifically, to support the sharing and reuse of formally represented knowledge among AI systems, it is useful to define the common vocabulary in which shared knowledge is represented; a specification of such a common vocabulary for a shared domain is called ontology [9].

Blackboard
An information processing structure composed of: several cooperating knowledge sources (each containing any kind of algorithm, rules, data, and so forth), a separate control element

(determining the order in which the knowledge sources are executed), and the blackboard itself (the locus of communication and global memory) [7].

Negotiation
A frequent form of interaction that occurs among agents with different goals is termed negotiation. Negotiation is a process by which a joint decision is reached by two or more agents, each trying to reach an individual goal or objective. The agents first communicate their positions, which might conflict, and then try to move towards agreement by making concessions or searching for alternatives. The major features of negotiations are: the language used by the participating agents, the protocol followed by the agents as they negotiate, and the decision process that each agent uses to determine its positions, concessions, and criteria for agreement [9].

Open System
A system composed of a variable number of parts that interact although they are developed independently, that act concurrently and asynchronously, that have a decentralized control, that possess limited knowledge, and that have limited and potentially inconsistent views of the overall system [6].

Integrating the different components of required in the multiagent system [8] and intelligent systems in an open environment, which can interact with heterogeneous applications and independent on the format of their input and output, gives a flexible, and easy use environment, and it can also be incorporated in the low level of a multiagent system.

2. Proposed Intelligent Environment Architecture (IEA):

The proposed architecture for context layers for the sensed Intelligent Environment is shown in the following figure.

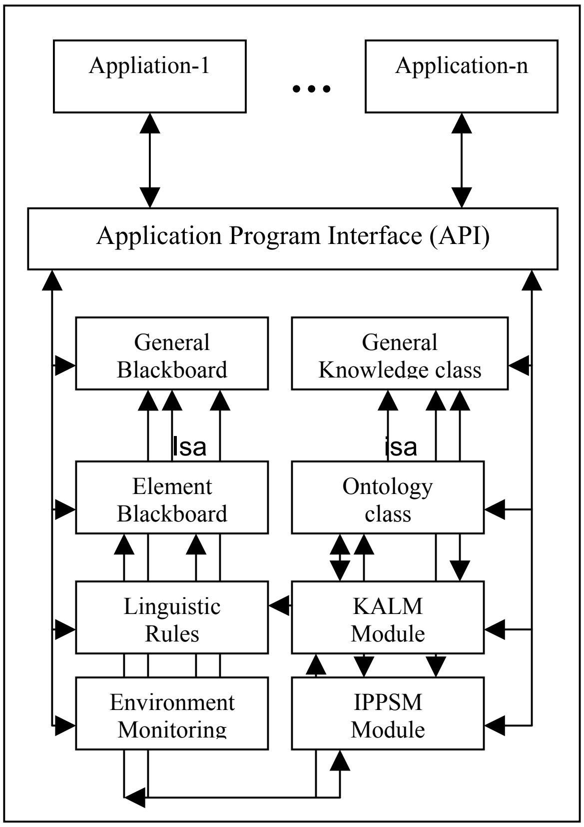
This architecture consists of:

Knowledge representation component: represents different types of knowledge in the *knowledge structure* and *knowledge index* [1], for the general knowledge level. This knowledge index or pattern can be used for knowledge-pattern similarity [3].

Ontology: represents specific domain knowledge in the same form of knowledge structure and knowledge index.

General blackboard: sharing communication area between all environment resources. It associates each element that can be application, object or concepts with a text area, the element type, and the module name of the application. This text area will be the mean of communication through understanding the text meaning by knowledge Acquisition module, which depends on the linguistic rules. The elements at this structure can be interacted by any other elements.

Element blackboard: is a subclass to general black board, but its member elements or text area needs a privilege.



Linguistic rules: represents the used natural language rules, syntactic, semantic, and pragmatic [1], with the semantic lexicon [5].

Knowledge acquisition and learning Module (KALM): this module [1] is responsible on interpreting the associated text with any element in the blackboard and converts it into the knowledge structure, using the linguistic rules. It can be extended to convert also any types of applications output to the knowledge structure. This module depends on paraphrasing the text into a specific syntax for declarative sentence of the used language to avoid the complexity in the free style text.

Inference process and problem solving module (IPPSM): this module [2] represents the state of the problem in the knowledge structure and apply the Meta knowledge, the inference process and the problem solving techniques, such as planning technique that can use the difference table, which defines the required operators to achieve the sub-goals.

Environment Monitoring: responsible on monitoring the status of the blackboard and announce the relevant elements.

Application Program Interface (API): is the Communication mean between the environment and external environment, or applications which can be any database application, business applications, expert systems, data mining, …etc.

The structure format of this environment is as follows:

Intelligent Environment Architecture (IEA):
{
- Knowledge Representation
{
Global Knowledge class
{
Knowledge Structure,
Knowledge Index
}
Ontology class
{
isa: Global Knowledge
Domain,
Domain Knowledge Structure,
Domain Knowledge Index
}
Linguistic Rules
{
Syntax, semantic, and pragmatic rules,
Semantic lexicon
}
}
- General Blackboard
{
General-Environment-element- List (element name, text area, type, module),
Difference-table (Event, operators-list),
Data Dictionary,

Add-element (element, type, module),
Remove-element (element, type, module)
Add-element-area (element),
Remove-element-area (element),
Get-element-area (element, text area)
Get- operators-list(event),
}
Element Blackboard
{
isa: General Blackboard
Environment-element-List (element name, text area),
}

- Knowledge Acquisition and Learning Module (KALM)
{
- Paraphrase the text into declarative sentences,
- Parse the sentences using the linguistic rules,
- Represent the sentences in the knowledge structure and knowledge index, and integrate it with the available knowledge.
}

- Inference Process and Problem Solving Module (IPPSM)
{
- Apply the difference table in the case of applying planning techniques.
- Apply the relevant operators, rules, or scripts from the knowledge index.

}

- Environment Monitoring Module
{
Check-Blackboard-Changes (element),
sAnnounce (element)
}

- Application Program Interface (API)
{
Convert (application output, text format),
Apply KALM (element),
Apply IPPSM *(element),*
Add-element (element, type, module),
Remove-element (element, type, module),
Get-element (element, text Area)
}

}

Scenario of Interaction

{

- Convert (application output, text format):
  The output of any application, or element, should be written in the text format associated with the element at the blackboard, using linguistic rules. Monitoring module will check the blackboard for any new state and announce the elements.

- Apply KALM (element):
  Apply Knowledge acquisition module that extract the information from the text into the knowledge structure and index, for both the general knowledge and ontology knowledge.

- Apply IPPSM (element):
  Apply inference process and problem solving module for solving the problem relevant to the available knowledge, using the difference table in the blackboard.

- Get-element (element, text Area):
  The result of processing any element or application in the text format is converted into the application format.

}

The following is the representation of the knowledge structure and knowledge index [1]:

Knowledge Structure
{
<u>*Entity/Action*</u> *{*
Entity-Name, Relation,
Script of action, Attribute [1..n]}.
<u>*Relation*</u> *{*
Relation-Type [1..n], Entity[1..n]).
<u>*Script of action*</u> *(*
Script-name, Roles: entity [1..n],
People: entity [1..n], Initial-State: state,
Goal-State: state, Events-Scenario:Action [1..n]}.
<u>*Attribute*</u> *{ isa: Attribute-Class,*
Attribute-Name, Attribute-Value [1..n]}.
<u>*Attribute-Class*</u> *{*
Domain: string/number/entity, Range: range-values, Range-Constraint: not range-values}.
<u>*Attribute-Value*</u> *{*
Operation: [arithmetic, logical],
Values: [entity,string], RelationTo}.
<u>*RelationTo*</u> *{*
Relation, Object}.
<u>*State/Object*</u> *{ isa: Entity, Id.}.*
}

The Knowledge index, pattern, of the above knowledge representation is [2]:

Knowledge Index
{
Entities(
Entity-1(
(Attributes –1,…,Attribute –n),
(Operator-1,…, Operator-n))
Entity-2(…),…,Entity-n(…))
Attributes(Attribute-1
(Entity-1,…, Entity-n),
Attribute-2(),…,Attribute-n(…))
Scripts(Script-1(
Goals(Goal-1, …, Goal-n)
Props(Prop-1…, Prop -n)
Roles(Role-1,…, Role –n)
Conditions(
Condition-1,…,Condition-n)
Actions(action-1,…,action-n)
Script-2(…), …Script-n(…))
Operators(Operator-1
(Action-1(Entity-1),
… …
Action-n(Entity-n)),
Operator-2(…),…,Operator-n.(…))
Actions(Action-1(attribute-1,…, attribute-n),
Action-2(…),…,Action-n(…))
Relations(Relation-1(Entity-i1,…, Entity-j1),
Relation-2(…),…,Relation-n(…))
Events(Event-i1(…),Event-i2(…),…,Event-in(…))
}.

3. Example

In the case of ebusiness system for an organization, where the system has the following components:

- Clients, Vendors, and competitors database.
- Financial, inventory, Production, and marketing Systems.
- Expert, Decision Support, and data mining Systems.

These applications are integrated through IEA as follows:

Consider the following scenario of interaction:

- Convert the output of the database applications into declarative sentences or facts and vice versa, using the linguistic rules, as example:

column1 of table-name has values row11,…and rown1.
columnn of table-name has values row1n,…and rownn.
e.g.
average-purchasing-price of client has values 250, 300, 200,…,and 350.
Quantity of stock has values 1000, 1200, 3000,… and 1500.

- Apply KALM module to convert this facts into knowledge structure, and knowledge index.

- The output facts of the business applications, like expert system or data mining, if it is applied on the above facts are converted into declarative sentences as follows:

  The competitors have small prices of products.
  The clients prefer small prices of products.

- Apply KALM module, on these facts.

- Apply IPPSM for inference or problem solving process. Consider as example, the system has to answer the following question: why does the clients prefer competitors?

  Consider the following knowledge structure and index, or pattern:

```
Entity: competitor
  { relation
  {  relation-name = "has",
     relation-type = "association"
     entity[1]: product
      {relation{
        {attribute:{attribute-name:
                  price
                 attribute-value:
                small }
        } }}}.

Action: prefer
{
 attribute: { attribute-name: agent
              attribute-value: {entity:
                  client},
 attribute: { attribute-name: object
              attribute-value:
               { entity: product
                { attribute-name: prices
                  attribute-value: small
                  }}} }  }.
```

Example of the Meta knowledge for inference process:

If the knowledge structure has the form:

If Entity1 (relation-type="association", entity2(attribute)) and action(attribute-value= entity2(attribute))
Then add the following facts:
action(attribute-value=entity1).

From this rule the following facts is added to the knowledge structure::
Clients prefer competitors

Thereafter, the search through the knowledge structure can answer the question, and add it to the blackboard.

The application converts this fact into its internal structure in the domain of business, using element blackboard, with the relevant ontology class. But if we ask the same questions for another domain like educational organization:

The competitors have good teachers.
The clients prefer the quality of teaching.

The knowledge structures of these facts are:

```
Entity: competitor
  { relation
  {  relation-name = "has",
     relation-type = "association"
     entity[1]: teachers
      {relation{
        {attribute:{attribute-name:
                  level
                 attribute-value:
                good }
        } }}}.

Action: prefer
{
 attribute: { attribute-name: agent
              attribute-value: {entity:
                  client},
 attribute: { attribute-name: object
              attribute-value:
               { entity: teaching
                { attribute-name: level
                  attribute-value:
                         quality
               }}} }  }.
```

There are two facts can be extracted:

Teaching (level, quality),
Teacher (level, good).

Applying similarity matching [3] on these facts, we can conclude the fact:

Clients prefer competitors

And the question will be answered, but may be at certain level of uncertainty.

4. Conclusions

In this paper we present a proposed architecture for an intelligent environment (IEA) for the integration of different heterogeneous applications or system components. IEA is based on representing different types of knowledge in the knowledge structure, and knowledge index to access the knowledge from any point of view. The environment resources or applications can be communicated through the black board structure, in which a text area is associated with each application and is used to present the information of this application. Using text area as a communication mean provides more flexibility of interacting the heterogeneous applications with more easily, and let the environment can adopt any application independent on the input-output format. IEA is independent on the domain since it represents the general knowledge and the specific domain knowledge or ontology, and can be incorporated in a multiagent system in the lower level to avoid destroying its architecture or framework. Application Program Interface (API) provides the environment with a high level of abstraction. Environment monitoring has a very important role in monitoring the behavior of the elements and the status of the environment. IEA can adopt and integrate different applications such as ebusiness, elearning, enterprise information portal, and information extraction.

References: -

[1] Aboul-Ela, Magdy; "Framework For Open Mind Learner", The Egyptian Computer Journal, Institute of Statistical Studies and Research, Cairo University, December 2004.

[2] Aboul-Ela, Magdy; " A Framework for an Intelligent Problem Solver", WSEAS (The World Scientific and Engineering Academy and Society) Transaction Journal on Computers, ISSN: 1109-2750, Issue 5, Volume 3, November 2004, Page 1563.

[3] Aboul-Ela, Magdy; "knowledge-pattern based information Extraction", WSEAS (The World Scientific and Engineering Academy and Society) WSEAS Conferences, Copacabana, Rio de Janeiro, Brazil, April 25-27 2005.

[4] Verspoor, M, C.; Papcun, J, G. and Sentz, K. "A Theoretical Motivation for Patterns in Information Extraction", Los Alamos Unclassified Report LAUR 02-1504, 2003.

[5] Gheith, Mervat; Aboul-Ela, Magdy; and Arafa, Waleed; "Lexical Acquisition for Information Extraction from Arabic Text Document", WSEAS (The World Scientific and Engineering Academy and Society) Transaction Journal on Systems, ISSN: 1109-2777, Issue 5, Volume 3, July 2004, Page 1964.

[6] Michael N. Huhns and Larry M. Stephenes. Multiagent Systems. Gerhard Weiss(Edt.), Massachusets Institute of Technology, 1999.

[7] Michael Wooldridge. Multiagent Systems. Gerhard Weiss(Edt.), Massachusets Institute of Technology, 1999.

[8] Bob Kummerfeld, Aaron Quigley, Chris Johnson, Rene Hexel; Merino:Towards an intelligent environment architecture for multi-granularity context description. Workshop on User Modelling for Ubiquitous Computing, 2003.

[9] M. Nowostawski, M. Purvis and S. Cranefield, "Modeling and Visualizating Agent Conversations", Proceedings of the fifth international conference on Autonomous agents, july 2001.