# A MULTI-AGENT SYSTEM FOR MOBILE ENVIRONMENTS

Jianwen Chen[1,2] and Yan Zhang[2]
[1] *IBM Australia, 20 Berry Street, North Sydney, NSW 2060, Australia, jchen@au1.ibm.com*

[2] *School of Computing & IT, University of Western Sydney, Penrith South DC NSW 1797, Australia, yan@cit.uws.edu.au*

Abstract: In this paper, we present a framework/model for a logic programming multi-agent system in mobile environments. Such a system consists of a number of agents connected via wire or wireless communication channels, and we model the interactions between agents in our formalization. Our formalization is knowledge oriented with declarative semantics. Our model can be used to study the details of knowledge transaction in mobile environments.

Key words: multi-agent system, mobile environments, extended logic programming.

## 1. INTRODUCTION

The advent of widespread portable computers has led to a wide variety of interesting hardware and software issues, and presented new challenges for researchers. Comparing to stationary environments, mobile environments have introduced a few specific features such as disconnection due to wireless network and mobility due to cell migration. In mobile environments, the communication channels can be wire or wireless. We believe that research on multi-agent system and knowledge transaction in mobile environments is important because this will significantly improve current development on both multi-agent systems and mobile systems. But so far no framework/model has been presented for multi-agent system in mobile environments and no study has been conducted for knowledge transaction in

mobile multi-agent system. There seems to be a separation between multi-agent systems and the intelligent agents community on one side, and the mobile system community on the other side [13, 10, 17]. On mobile system community side, work in paper [4, 5, 12] has introduced calculus to describe the movement of processes and devices in mobile ambient, and the work in [3, 11, 6] has presented a Java based mobile agent to implement functionalities for mobile systems. The approaches above are not suitable for knowledge and have no declarative semantics. They are low level algorithms for "how to do" and have no high level "what to do" intelligent functionality. The details of transaction can't be specified in these approaches. On multi-agent and intelligent agent community side, a lot of frameworks/models have been developed for problem solving, knowledge representation and reasoning such as stable model/answer set, SMODEL, DLV and XSB model in paper [7, 15, 16]. But these models are only discussed and limited in classic non-mobile environments, and haven't be extended to mobile environments. In this paper we present a formalism and definition for a mobile logic programming multi-agent system (MLPMAS). With respect to previous work, our model has three advantages: 1) Our model is knowledge oriented and has declarative semantics inherited from logic programming; 2) It can specify details of knowledge transaction; 3) Our model can be used to study knowledge transaction in mobile environments.

The rest of this paper is organized as follows. In section 2, we give an overview of extended logic programming. In section 3, we introduce our knowledge study environmental model. In section 4, we formalize our mobile logic programming multi-agent system (MLPMAS). In section 5, we give an example to demonstrate how to specify a MLPMAS system in a particular problem domain. Finally, in section 6, we conclude our work.

## 2. EXTENDED LOGIC PROGRAMS

Logic programming has been proved to be one of the most promising logic based formulations for problem solving, knowledge representation and reasoning. In non-mobile environments, traditional logic programming is used as a knowledge representation tool. An important limitation of this method is that logic programming does not allow us to deal directly with incomplete information, and therefore we only can get either *yes* or *no* answer from a query. When we study knowledge transaction in mobile environments, we should clearly understand that there is a major different between the scenario that the transaction fails and the transaction hangs on due to mobile user's sleep. The first scenario is transaction fails in the sense

of its negation succeeds, it is a *no* answer for a query. The second scenario is transaction doesn't succeed because of incomplete information, the answer is *unknown* for a query transaction, but may become a definite answer *yes* or *no* after sometime. Therefore, in mobile environments, we need a method which can deal with incomplete information explicitly. The extended logic program [2, 8, 1] can overcome such a limitation, it contains classical negation $\neg$ in addition to negation-as-failure *not,* and includes explicit negative information. In the language of extended programs, we can distinguish between a query which fails in the sense that it does not succeed and a query which fails in the stronger sense that its negation succeeds.

Generally speaking, an extended logic program is a finite set of rules:

$L_0 \leftarrow L_1, \ldots, L_m, not\ L_{m+1}, \ldots, not\ L_n,$

where $n \geq m \geq 0,$ and each $L_i$ is a literal. A literal is a formula of the form $A$ or $\neg A,$ where A is an atom. We say logic program $\Pi$ entails a literal $L$ if $L$ is always true in all answer sets of $\Pi,$ this is denoted by $\Pi \models L.$

## 3. ENVIRONMENTAL MODEL

When we study the transaction processing in mobile environments, we use the three level mobile environment model in the paper [9, 14] to represent the salient features of mobile environments. There is a Home Server (HS) acting as permanent storage of Mobile hosts' (MH) Files. There are Mobile Support Stations (MSS) providing services to a MH when it is within its cell. The MSS is connected to the HS via hardwires. The MH is continuously connected to a MSS via a wireless link while accessing data. It may become disconnected either voluntarily or involuntarily. In classical environments, an intelligent agent is an active object with the ability to perceive, reason and act. We assume that an agent has explicitly represented knowledge and a mechanism for operating on or drawing inferences from its knowledge. We also assume that an agent has the ability to communicate. In a distributed computing system, intelligent agent has been introduced to communicate with each other in order to achieve their goals.

Here we propose a new environment model to study knowledge base in mobile environments. This model integrates the features of both mobile environment [13, 10] and intelligent agents [17, 2] as shown in Figure 1.

In this environment model, we assume that every Mobile Host (MH) has its own knowledge base (KB) and intelligent agent (A11, A12, A21, A22), every MSS has knowledge base and agent residing on it as well, MSS1 and MSS2 represent different MSS in different geographic areas. Home Server (HS) level has a knowledge base and an agent that represents a set of rules of knowledge base. Every intelligent agent on MH will work on behalf of MH

that resides on all the agents in the same geographic area will negotiate, communicate, and cooperate with each other to achieve the goal for themselves and their systems.

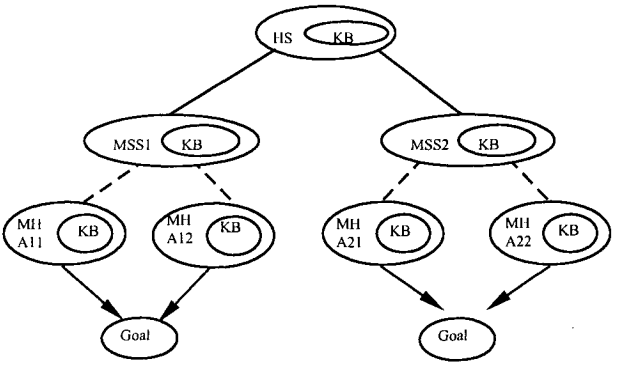


*Figure 2.* Knowledge Study Environment Model

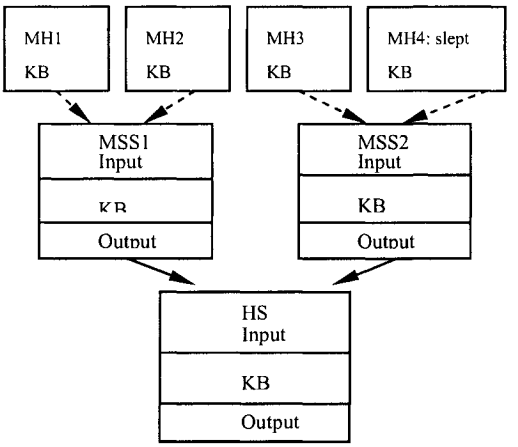


*Figure 1. A MLPMAS*

Mobile Logic Programming Multi-Agent System Formalization

In this section we formalize and define a Mobile Logic Programming Multi-Agent System (MLPMAS) in mobile environments, where each agent is represented by an extended logic program that contains its knowledge about itself and other agents. Agents communicate via communication channels.

We define and formalize the MLPMAS systems based on three layer environmental model. The model of A MLPMAS system is shown in Fig 2.

A mobile logic programming multi-agent system includes MH, MSS and HS three levels, the local knowledge base is located on each level. The system consists of a set of agents, the agent resides on MH, MSS and HS levels respectively, connected through communication channels. The agent on each level contains its own logic program representing its local information and reasoning method. Agents use information received from their incoming channels as input for their reasoning, where the received information may be overridden by other concerns represented in their programs. Agents produce output to their outgoing communication channels.

**Definition 1:** *A mobile logic programming multi-agent system*, or *MLPMAS*, is a pair $F = < A, C >$ , where $A$ is a set of agents: $A = A_{MH} \cup A_{MSS} \cup A_{HS}$ , and $C \subseteq A \times A$ is a reflexive relation representing the communication channels between agents. For any $a_1, a_2 \in A$ , if $< a_1, a_2 > \in C$ , then we say agents $a_1$ and $a2$ have a *communication channel*. Each agent $a \in A$ , there is an associated extended logic programs *LocalKB(a)* which reprents agent *a*'s *local knowledge base*.

Now we explain the definition of MLPMAS system above through the following Example 1. In our example, investor agent resides on MH, group agent resides on MSS, and fund manager agent resides on HS. Investor agent manages the local knowledge base and provides output to group agent on behalf of MH. Group agent collects information from investor agents, manages local knowledge base on MSS and sends output to fund manager agent. Fund manager agent collects information from group agents, does the investment decision and manages the local knowledge base on HS. Investor agent, group agent and fund manager agent are represented by $a_{MH}$, $a_{MSS}$ and $a_{HS}$ respectively.

**Example 1**: We have a mobile logic programming multi-agent system $F = < A, C >$, in this MLPMAS system, we have four mobile hosts MH1, MH2, MH3 and MH4, the investor agent resides on each MH:

$A_{MH} = \{a_{MH1}, a_{MH2}, a_{MH3}, a_{MH4}\}$

We have two mobile support station MSS1 and MSS2, group agent resides on each MSS:

$A_{MSS} = \{a_{MSS1}, a_{MSS2}\}$

We have one home server HS, fund manager agent resides on HS:

$A_{HS} = \{a_{HS}\}$

MH1 and MH2 are in geographic location of MSS1, MH3 and MH4 are in geographic location of MSS2. We have wireless communication channel between MH and MSS:

$< a_{MH1}, a_{MSS1} > \in C$ , $< a_{MH2}, a_{MSS1} > \in C$ ,

$< a_{MH3}, a_{MSS2} > \in C$ , $< a_{MH4}, a_{MSS2} > \in C$

We have wire communication channel between MSS and HS:

$< a_{MSS1}, a_{HS} > \in C, < a_{MSS2}, a_{HS} > \in C$

As we mentioned earlier, each agent is associated with an extended logic program of its local knowledge base.

We define input and output of agents in MLPMAS systems as follows.

**Definition 2.** Let $F = < A, C >$ be a MLPMAS, where $A = A_{MH} \cup A_{MSS} \cup A_{HS}$. At MH, MSS or HS level, for $\forall a \in A$, we have two parts of inputs: *message input and knowledge input*, denoted by $MessageInput(a, X)$ and $KnowledgeInput(a, Y)$ respectively. That is,

$Input(a) =< MessageInput(a, X), KnowledgeInput(a, Y) >$

here $X \subseteq A, Y \subseteq A$, X, Y are subsets of A. Agent a collects message input from agents in X, and collects knowledge input from agents in Y, where

for $\forall b \in X$, we have $< a, b > \in C$, or $< b, a > \in C$ and

for $\forall b' \in Y$, we have $< a, b' > \in C$, or $< b', a > \in C$.

i.e. we know there is a communication channel between agent a and agent b, and agent a and agent $b'$ respectively.

Message input is the information that an agent sends to another agent for the communication purpose. Such as one agents informs another agent that it will move into another MSS geographic area. This information will not cause any influence to the other agent's local knowledge base. While knowledge input is the information produced by the other agent's local knowledge base, and will be taken into the agent's local knowledge base, i.e. the answer set of a logic program.

For $\forall a \in A$, we have two parts of output, message output and knowledge output, denoted by $MessageOutput(a, X)$ and $KnowledgeOutput(a, Y)$ respectively. That is,

$Output(a) =< MessageOutput(a, X), KnowledgeOutput(a, Y) >$

here $X \subseteq A, Y \subseteq A$. Agent a sends message output to agents in X, and sends knowledge output to agents in Y.

Message output is information output for communication purpose, this information will not cause any influence to the other agent's local knowledge base, while knowledge output is the information that produced by the agent's local knowledge base and will have impact for the other agent's knowledge base.

**Definition 3:** We define *knowledge input and output* in MLPMAS systems on MH level as follows.

There is no input for MHs at MH level because this is the first level in MLPMAS systems, i.e.

$KnowledgeInput(a_{MH}, Y) = \phi$ (1)

The knowledge output can be derived from the equation:

$KnowledgeOutput(a_{MH}, a_{MSS})$

=an answer set of $[LocalKB(a_{MH}) \cup KnowledgeInput(a_{MH}, Y)]$ (2)

i.e. knowledge output is an answer set of the program formed by the local logic program of agent $a_{MH}$ with extending of knowledge input from Y for agent $a_{MH}$. $LocalKB(a)$ is an extended logic program as we defined in Definition 1, $KnowledgeInput(a,Y)$ is a set of facts (beliefs). Note that $LocalKB(a_{MH}) \cup KnowledgeInput(a_{MH},Y)$ is viewed as a new logic program while fact $e \in KnowledgeInput(a,Y)$ is treated as a rule $e \leftarrow$.

**Definition 4:** We define *knowledge input and output* in MLPMAS systems on MSS level as follows.

The knowledge input can be derived from the equation:

$$KnowledgeInput(a_{MSS},Y) = cons(\bigcup_{a_{MH} \in Y} KnowledgeOutput(a_{MH},a_{MSS}),S_F) \quad (3)$$

where *cons(X)* represents the maximal consistent subnet. The knowledge input of $a_{MSS}$ is the maximal consistent subset of knowledge output from Y to agent $a_{MSS}$ with respect to the select function $S_F$. $S_F$ is the *selection function* of system *F*. For knowledge output, $\bigcup_{i \in I} knowledgeOutput(b,a)$ may be inconsistent, $S_F$ is introduced to solve such inconsistency by taking proper preference in the domain. Note that $S_F$ is domain dependent, it can be a special logic programming rule for specific problem domain.

The knowledge output can be derived from the equation:

$KnowledgeOutput(a_{MSS},a_{HS})$

=an answer set of $[LocalKB(a_{MSS}) \cup KnowledgeInput(a_{MSS},Y)]$ (4)

i.e. knowledge output is an answer set of the program formed by the local logic program of agent $a_{MSS}$ with extending of knowledge input of agent $a_{MSS}$.

**Definition 5:** *knowledge input and output* in MLPMAS systems on HS level as follows.

The knowledge input can be derived from the equation:

$$KnowledgeInput(a_{HS},Y) = cons(\bigcup_{a_{MSS} \in Y} KnowledgeOutput(a_{MSS},a_{HS}),S_F) \quad (5)$$

i.e. knowledge input of $a_{HS}$ is the maximal consistent subset of knowledge output from Y to agent $a_{HS}$ with respect to the select function $S_F$.

The knowledge output can be derived from the equation:

$KnowledgeOutput(a_{HS})$

=an answer set of $[LocalKB(a_{HS}) \cup KnowledgeInput(a_{HS},Y)]$ (6)

i.e. knowledge output is an answer set of the program formed by the local logic program of agent $a_{HS}$ with extending of knowledge input of agent $a_{HS}$.

## 4. AN EXAMPLE FOR MLPMAS SYSTEM

We will go through a completed example in this section to specify a MLPMAS system according to the formalization in section 4. We still use MLPMAS system Fig 2 in this example.

**Example 2**: In this example, we study a case in a specific investment problem domain. As showed in Figure 2, at MH level, we have MH1, MH2, MH3 and MH4. MH1 and MH2 are in the cell of MSS1, MH3 and MH4 are in the cell of MSS2. MSS1 and MSS2 are connected to the same HS. At MH level, each MH has a local knowledge base that includes a set of investment rules, investor agent resides on it. At MSS level, MSS has own knowledge base, MSS accepts the input from MHs and produces the output based on the input and own belief. The HS accepts the input from MSS level, it has own local knowledge base, investment decision will be made on HS level.

For the initial status, we assume MH1 and MH2 are all alive when transaction is processed in MSS1 cell. In MSS2 cell, the MH3 is alive, while MH4 is slept at the moment HS is requesting the transaction information from all related MH agents. The HS will need information from MH4 when the time it does the decision making.

MH Level:

On MH level, there is no input for the agent on MH. According to equation (2), we have

$KnowledgeOutput(a_{MH}, a_{MSS})$

=an answer set of $[LocalKB(a_{MH}) \cup KnowledgeInput(a_{MH}, Y)]$

= an answer set of $[LocalKB(a_{MH})]$

i.e. on MH level, the knowledge output is an answer set of local knowledge base. Based the local knowledge base on MHs, the knowledge outputs are derived as below on MH1, MH2, MH3 and MH4.

$KnowledgeOutput(a_{MH1}, a_{MSS1}) = \{profit(share1), risk(share1), \neg cost(share1)\}$

i.e. it is high profit, high risk and low cost to invest share1 on MH1.

$KnowledgeOutput(a_{MH2}, a_{MSS1}) = \{profit(share1), \neg risk(share1), \neg cost(share1)\}$

i.e. it is high profit, low risk and low cost to invest share1 on MH2.

$KnowledgeOutput(a_{MH3}, a_{MSS2}) = \{profit(share1), \neg risk(share1), \neg cost(share1)\}$

i.e. it is high profit, low risk and low cost to invest share1 on MH3.

The MH4 is slept at the moment the information is retrieved from it.

MSS level:

On MSS level, according to equation (3), knowledge input on MSS1 is as below:

$$KnowledgeInput(a_{MSS\,1}, Y)$$

$$= cons(\bigcup_{a_{MH} \in Y} KnowledgeOutput(a_{MH}, a_{MSS\,1}), S_F)$$

$$= cons(KnowledgeOutput(a_{MH\,1}, a_{MSS1}) \cup KnowledgeOutput(a_{MH\,2}, a_{MSS1}), S_F)$$

For agent $a_{MSS1}$, $risk(share1)$ is a belief in output of $a_{MH1}$, while $\neg risk(share1)$ is a belief in output of $a_{MH2}$, they are inconsistent. Here we assume selection function $S_F$ takes positive atom as higher preference for investment risk, therefore $risk(share1)$ will become the input of $a_{MSS1}$.

We have knowledge input as below:

$$KnowledgeInput(a_{MSS1}, Y) = \{profit(share1), risk(share1), \neg cost(share1)\}$$

We can see that different knowledge input is derived with considering selection function in specific problem domain, therefore different answer set is derived for decision making due to selection function.

In the same way, we know the knowledge input of MSS2 agent equals:

$$KnowledgeInput(a_{MSS2}, Y)$$

$$= cons(\bigcup_{a_{MH} \in Y} KnowledgeOutput(a_{MH}, a_{MSS\,2}), S_F)$$

$$= cons(KnowledgeOutput(a_{MH\,3}, a_{MSS2}), S_F)$$

$$= \{profit(share1), \neg risk(share1), \neg cost(share1)\}$$

On MSS1, we have rule r1 related to this investment in its knowledge base

$$\{r1 : holds(\inf o - requested(HS, MHi)) \leftarrow holds(slept(MHi))\}$$

On MSS2, we have rule r2 related to this investment in its knowledge base

$$\{r2 : holds(\inf o - requested(HS, MHi)) \leftarrow holds(slept(MHi))\}$$

The r1 and r2 denote that if MHi is slept at the time the HS agent requests transaction information from MHs, HS will request information from MHi when HS does the decision making for the transaction.

According to the equation (4), the knowledge output can be derived:

$KnowledgeOutput(a_{MSS}, a_{HS})$

=an answer set of $[LocalKB(a_{MSS}) \cup KnowledgeInput(a_{MSS}, Y)]$

Thus, the knowledge output of MSS1is derived as below:

$KnowledgeOutput(a_{MSS1}, a_{HS}) = \{profit(share1), risk(share1), \neg cost(share1)\}$

The knowledge output of MSS2 is derived as below:

$$KnowledgeOutput(a_{MSS2}, a_{HS})$$
$$= \begin{Bmatrix} profit(share1), \neg risk(share1), \neg cost(share1), \\ \inf o - requested(HS, MH4) \end{Bmatrix}$$

i.e. new belief *info-requested(HS, MH4)* is added to the answer set on MSS2 because of rule r2 in its local knowledge base.

HS level:

On HS level, based on the equation (5), knowledge input of HS agent equals:

$$KnowledgeInput(a_{HS}, Y)$$
$$= cons(\bigcup_{a_{MSS} \in Y} KnowledgeOutput(a_{MSS}, a_{HS}), S_F)$$

$$= cons(KnowledgeOutput(a_{MSS1}, a_{HS}) \cup KnowledgeOutput(a_{MSS2}, a_{HS}), S_F)$$
$$= \{profit(share1), risk(share1), \neg cost(share1), \inf o - requested(HS, MH4)\}$$

$risk(share1)$ is a belief of input on HS with considering the selection function.

We have rules r3-r9 in local knowledge base of HS.

$$\left\{\begin{array}{l} r3: holds(invest(share1),s) \leftarrow holds(profit(share1),s), \neg holds(risk(share1),s), \\ \neg holds(\mathrm{cost}(share1),s), holds(\inf o - get(MHi), res(request - \inf o(MHi),s)) \\ r4: \neg holds(invest(share1),s) \leftarrow holds(risk(share1),s) \\ r5: \neg holds(invest(share1),s) \leftarrow holds(\mathrm{cost}(share1),s) \\ r6: \neg holds(risk(share1),s) \leftarrow notholds(risk(share1),s) \\ r7: \neg holds(\mathrm{cost}(share1),s) \leftarrow notholds(\mathrm{cost}(share1),s) \\ r8: \neg holds(invest(share1),s) \leftarrow holds(\inf o - requested(HS,MHi),s), \\ \neg holds(\inf o - get(MHi), res(request - \inf o(MHi),s)) \\ r9: \neg holds(\inf o - get(MHi),s) \leftarrow notholds(\inf o - get(MHi),s), \\ holds(\inf o - requested(MHi),s), holds(timeout(MHi),s) \end{array}\right\}$$

The r3 denotes if it is high profit, low risk, low cost to invest share1 and HS gets requested information from ever slept MHi, HS will do the decision to invest share1. The r4, r5 and r8 denote if share1 is high risk or high cost on any MHi, or can't get information from ever slept MHi, HS will make the decision that share1 won't be invested. The r6 and r7 denote that if share1 hasn't be specified to be high risk or high cost for any MHi, then it is considered to be low risk or low cost. The r9 denotes that if HS hasn't got requested information from slept MHi until time is out, then HS will assume no information is available from MHi.

The knowledge output is derived as below according to the equation (6):

$KnowledgeOutput(a_{HS})$

=an answer set of $[LocalKB(a_{HS}) \cup KnowledgeInput(a_{HS}, Y)]$

$risk(share1)$ is a belief of knowledge input of HS, according to the rule r4 of knowledge base, $\{\neg invest(share1)\}$ will be in every answer set of $[LocalKB(a_{HS}) \cup KnowledgeInput(a_{HS}, Y)]$. Therefore we say $\{\neg invest(share1)\}$ is entailed, i.e. agent on HS makes the decision that share1 won't be invested. In this example, no matter what input from MH4, HS will do the decision that share1 can't be invested after considering the input from MH4. After HS has made decision that share1 will not be invested. The transaction decision will be sent to MSS, and all involved MHs will be noticed by broadcasting of MSS.

## 5. SUMMARY

In this paper, we have presented and formalized a logic programming multi-agent system for mobile environments. Our formalization is knowledge based and has declarative semantics inherited from logic programming. Based on our formalized MLPMAS system, the details of knowledge transaction can be studied in mobile environments.

# REFERENCES


1. Baral, C., Knowledge Representation, Reasoning and Declarative Problem Solving, Cambridge University Press, 2003.
2. Baral, C., and Gelfond, M., Logic Programming and Knowledge Representation, Logic Programming, 1994, pp. 73-148.
3. Bettini, L., et al, KLAVA: a Java package for distributed and mobile applications, Software Practice and Experience, 32 (2002) 1365-1394.
4. Cardelli, L., A. Gordon, D., Mobile Ambients, Theoretical Computer Science, 240 (2000), 177-213.
5. Cardelli, L., A. Gordon, D., Types for the Ambient Calculus, Information and Computation 177 (2002), 160-194.
6. Deugo D. Choosing a mobile agent messaging model, Proceedings of ISADS 2001. IEEE Press, 2001;278-286.
7. Eiter, T., and et al., A deductive system for nonmonotonic reasoning, in proceedings of the 4th International conference on Logic Programming and Nonmonotonic Reasoning (LPNMR97), pp 363-374. LNAI, Vol. 1265, 1997.
8. Gelfond, M., and Lifschitz, V., Classical Negation in Logic Programs and Disjunctive Databases, New Generation Computing, 1991, pp. 365-385.
9. Imielinski, T., and Korth, H.F., Mobile computing, Kluwer Academic Publishers, 1996.
10. Komiya, T., et al, Mobile Agent Model for Transaction Processing on Distributed Objects, Information Sciences, 2003, pp.1-16.
11. Lange D, Oshima M., Programming and Deploying Java Mobile Agents with Aglets, Addision-Wesley: Reading, MA, 1998.
12. Milner, R., et al,, A calculus of mobile processes, Parts 1-2, Information and Computation, 100 (1) (1992) 1-77.
13. Milojicic, D. Mobile Agent Applications, IEEE Concurrency, 1999, pp. 80-90.
14. Mirghafori, N., and Fontaine, A., A Design for File Access in a Mobile Environment, in proceedings of the IEEE - Conference on Mobile Computing, 1995, pp. 57-61.
15. Rao, P., et al, XSB: A system for efficiently computing well-founded semantics, in Proceedings of the 4th International Conference on Logic Programming and Nonmonotonic Reasoning, pp 2-17. LNAI, vol. 1265, 1997.
16. Vos, M. D., and Vermeir, D., Extending Answer Sets for Logic Programming Agents, in Proceedings of the Logic in Artificial Intelligence (Jelia2000) workshop, 2000.
17. Wooldridge , M., An Introduction to Multiagent Systems, John Wiley & Sons, LTD, 2002.