# MAS DEVELOPMENT AND ITS ANALYSIS BASED ON FCA

**Petr Gajdoš and Michal Radecký**

Department of Computer Science, FEECS, VŠB – Technical University of Ostrava,
17. listopadu 15, 708 33 Ostrava-Poruba, Czech Republic

*petr.gajdos@vsb.cz, michal.radecky@vsb.cz* (Petr Gajdoš, Michal Radecký)

## Abstract

MAS development process has similar features to standard Information Systems. However, there are special features that have to be taken into account, in particular: agent based architecture, agents autonomy and communication, etc. According to mentioned points, MAS development process should be extended or changed. The process development can be handled and documented by the standard UML tool. The output of such development process is MAS with automatic or semi-automatic generated agents that behave according to defined MAS model. The first part of this paper describes MAS model and its development. Whole model should be analyzed as well as particular agents behavior. This paper also describes a possible usage of Triadic Formal Concept Analysis (FCA) which helps us to find some hidden information on MAS and agents interaction. The process of FCA integration within MAS is mentioned in separated part.

**Keywords:** MAS, Multi-Agent System, Process modeling, UML, Meta-model, Formal Concept Analysis.

## 1 Introduction

Process modeling has become more an more important during last decade. The main reason for this increased interest is the need to provide computer aided system integration which is also very important in Multi-Agent System (MAS) technology. The primary focus such modeling is to describe the way how activities are ordered in time. Such ordering affects behavior of all agent and whole MAS, respectively. An integration of process and behavior modeling is a part of MAS development process which will be described in following text. Automatic or semi-automatic generated agents (program source codes) represent the main goal of our research. These agents can be executed within different MAS frameworks. JADE framework was selected in our case. The real behavior of agents can be analyzed during system runtime. Formal Concept Analysis (FCA) represents a power tool for such kind of analysis. Its output is a list of triadic formal concepts which give us the information on agents' behavior in dependence on conditions resulting from state of the MAS world. Thanks to this approach, it is able to find agents that can substitute each other according to their capabilities or knowledge. This makes whole system more flexible and stable.

## 2 Agent within the MAS

Several types of agents can be found in one Multi Agent System as well as in the "real world" which is realized by this MAS. These "types" are able to denote as *Agent Classes*, according to the goals, internal architecture and behavior of particular agents. Combination of two perspectives is concerned with agent's classification (from our research point of view) - outside behavioral view (proactive or reactive agents) and the view of internal specification (deliberative or process specifications) [9].

Each Agent is determined by its own objectives and the way to meet these objectives is founded on internal behavior of a given agent. Internal behavior of such agent is specified by algorithms. The agent lives, behaves and reacts to stimulus and environment, according to requirements of these algorithms.

It is necessary to take into account the fact that each agent is an absolutely autonomous element of MAS and thus all internal behavior have to be based only on processes, activities, knowledge and facilities that belong to a given agent. Then, a goal-seeking behavior of whole MAS is formed by communication of separated agents and by interconnection of several internal agent behaviors. This interaction is realized through the use of message passing adapted to the MAS demands. Generally, this idea of agents autonomy is ground for modeling and specification of whole MAS as well as the particular agents.

## 3 Modeling of the MAS

The initial phases of development process are concerned with MAS modeling as a standard information

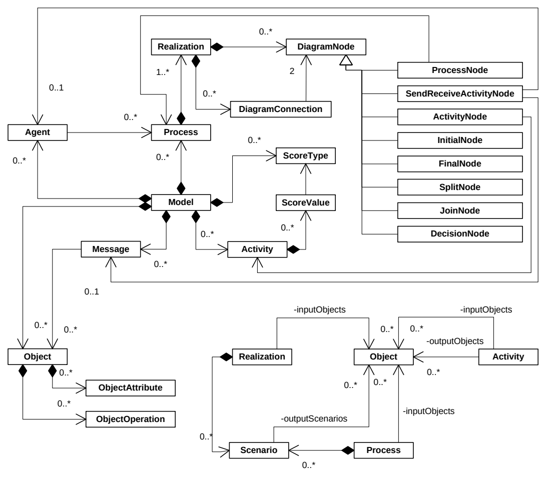


Fig. 1 The class diagram of MAS Model meta-model.

system. It means, that it is necessary to specify some requirements, input or output objects, roles and last but not least, the business processes or conceptual models. It is possible to use standard UML techniques (Class Diagrams, Activity Diagrams, Use Case Diagrams) for these purposes. Detection of the several elements as agents or objects is the main goal of the initial phases of the MAS development process.

At this point, it is fine to clarify that term *agent* expresses only the "type of agents" in the context of MAS modeling, e.g. Car is a type of agents, and car_A, car_B are the particular agents. Real particular agents represent the instances of this type. It is analogous with the terms class and object from the Object Oriented Technology. The real agents are not issues of MAS design and modeling phases, but they will appear only during the implementation phase, simulation and operation of a given MAS.

The output of this modeling phases represents the first version of MAS Model. This model is built on the defined meta-model that covers all elements and their relations within the rising MAS. The figure 1 contains a UML Class Diagram that depicts the basic structure and elements of our MAS Model on the mentioned metamodeling level. This abstraction is sufficient for better readability and clarity of MAS description from the structural and semi-formal point of view.

## 4 Modeling of the Agents and their Behaviors

Just the first version of the MAS Model is mentioned in the previous section. Why just the first version? Modeling on the highest level is not able to capture all specificity and features of the individual elements. So it stands to reason that this version of MAS Model is incomplete and describes only the several elements (agents, processes, objects, etc.) with their fundamental attributes but without their internal specifications. This internal specification is a task for the following phases of MAS development process and some result from previous phases are usable here, e.g. business processes.

### 4.1 Agent Behavior Diagram

An agent internal behavior is specified by algorithms expressed in the processes. Each such algorithm is modeled as just one *realization*. The *Agent Behavior Diagram* tool is used for purposes of these realizations modeling. The *Agent Behavior Diagram* is based on UML Activity Diagram so it is a diagrammatic technique which describes the series of standard UML Activity Diagram elements such activities, processes and other control elements as well as some new extensions and elements. All of them express the algorithm of agent's internal behavior together. These new elements are concerned with message passing among agents or with other specific facilities of MAS.

These extensions are mainly supported by the implementation of special “Send/Receive Activities” which includes an additional information on message content and message receiver/sender identification. Decision elements from standard UML Activity Diagrams are improved too. The modified “decision elements” and their output edges can hold some extra information. This information is used within a message-based determination of agent's behavior which leads to agent's objectives. Brand-new new modeling or specification elements are *scenarios* of processes. [8, 7] Also, a couple of rules are bound together with creation of these diagrams. First, each process, as well as diagram or algorithm, have to have exactly one “initial node” and exactly one “final node”. This prerequisite is necessary for further connection of processes together with model overall agent behavior.

### 4.2 Internal structure of particular agents

Internal structure of agents is also important attribute of their modeling. One new notion is important to clarify – *realization*. The realization is a modeling element that represents one of possible algorithms of process firing. It means that each process (except primary process which has just one realization) can have one or more realizations. Each of such algorithms is expressed by one Activity Behavior Diagram, so then process can have more realizations and each realization has just one ABD. Each agent must have one *primary process* that covers whole behavior (life).

The activities and processes, except primary processes, can be specified globally within the MAS. These globally specified elements are possible to subsume into the behaviors of a set of agents (a set of types of agents). Some advanced features are concerned with the distribution of particular process specifications too. The process with identical name can be defined within an agent, as well as a member of set of processes on the global level. In this case, some possible restrictions can be set up for each process. The agent's local process can extend a set of realizations of this process which is defined within the global repository. Of course, it can narrow this set only to the local realizations owned by a given agent. Actually, the realizations can be deliberated in real-time too. This approach is required and used for reconfiguration principle that is designed for intelligence and decision making within the Intelligent Agent. [4] The choice of a suitable realization depends on a current situation. Agents behavior can evolve in time; the agents have to be able to reconfigure their state and behavior in accordance to the situation.

### 4.3 Reconfiguration approach

The Intelligent Agent is based on one main life process that specifies just the scope of its behavior. This agent can dynamically change pieces of its own behavior according to the situations. This principle is denoted as behavior reconfiguration approach. The process of reconfiguration is based on replacement of a given part of the agents processes by another one that is the most suitable for current situation and conditions. As mentioned above, a set of possible applicable algorithms (realizations) of each “reconfiguration point” (generally the process node of ABD) is defined for this purpose. An important and expected situation will appear whenever one reconfiguration point (process node or element) corresponds with two or more realizations. It means that such process is able to fire by different ways.

A brief summary of reconfiguration algorithm is illustrated here:

1. The Specification Phase - definition of all realizations related to the process that could be reconfigured.
2. The Selection Phase - checking of the applicable realizations of a given process and finding the most suitable one.
   (a) the selection of applicable realizations - based on realizations' input objects occurrence
   (b) the looking for the most suitable Realizations - based on realizations' input objects values and properties, scores, etc.
3. The Execution Phase - chosen realization firing.

Mentioned reconfiguration approach is not a main content of agents modeling but its requirements and principles have to affect the specification processes too. The tools and techniques used for the behavior specification should be adapted to the next reconfiguration algorithm employment.

## 5 Modeling Framework as basis of MAS development

The software process of MAS is very similar to standard information system software process however some dissimilarities and extensions are desirable. The MAS Model is headstone for MAS construction based on processional description of agents' internal behaviors. Correct and complete model specification can

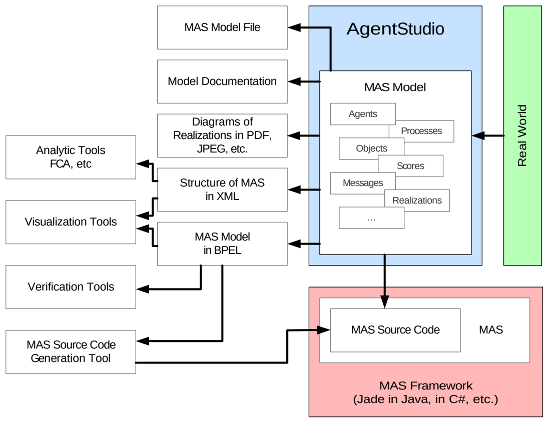


Fig. 2 The concept of AgentStudio architecture.

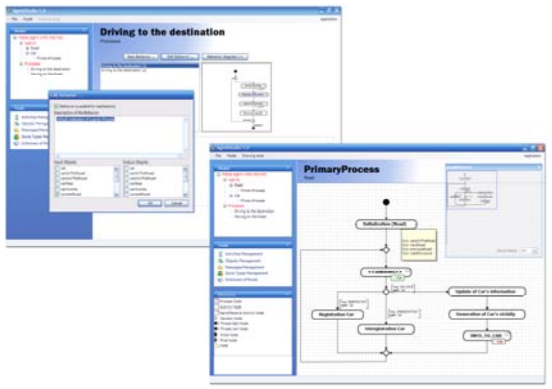


Fig. 3 Illustrative screen shots of AgentStudio.

be used for purposes of automatic or semi-automatic agents generation. It means that we can obtain all source codes (or skeletons of these source codes) from the MAS Model. From this point of view, the MAS Model become a core of whole MAS and its functionality. Also, the other facilities as behavior verification, formal model verification, simulation, etc. is possible to realize thanks to correct model specification.

The application called "**Agent**Studio" has been developed. Its task is to offer modeling framework (see the figures 2, 3). This framework is employed on the most lower level of pyramid (see the figure 6). Specification of MAS Model structure and its elements or attributes, production of ABD, mapping of communication among agents, reconfiguration support are the main functions of **Agent**Studio application. It could be also used for source code generation and programming specification in the view of afore mentioned approaches.

## 6 Triadic FCA introduction

Our analytical tool is based on a triadic approach of Formal Concept Analysis (FCA) which is well known in the area of data analysis. This section brings a shot introduction of triadic concepts and lattices.

### 6.1 Basic definitions

Till now, so called *biadic* (or *dyadic*) formal context and concepts were introduced. It means, that the context consists of two sets (objects and attributes) and there is one binary relation between them. The triadic approach of Formal Concept Analysis gave rise to a new class of algebraic structures; so-called trilattices [11, 3, 2] which are a triadic generalization of lattices. Since Boolean lattices are fundamental algebraic structures in Lattice Theory and Mathematical Logic, it is natural to ask for the triadic analogue of Boolean lattices, the Boolean trilattices, which play a similar role in the triadic case as Boolean lattices in the dyadic case.

A triadic context consists of sets of formal objects, formal attributes and formal conditions together with the formalization of the ternary relation saying when an object has an attribute under a certain condition. Triadic contexts provide a natural interpretation for modalities like *necessity* and *possibility*, in particular for the case of the dyadic relationships between formal objects and attributes are considered: a formal object $g$ has *necessarily* a formal attribute $m$ if $g$ has $m$ under all formal conditions of the context; $g$ has *possibly* $m$ if $g$ has $m$ under some formal condition. Such necessity and possibility relations give rise to dyadic contexts allowing a modal analysis of triadic data contexts. About ten years ago, triadic contexts were presented by Lehmann and Wille [5] as an extension of Formal Concept Analysis. However, they have rarely been used up to now, which may be due to a rather complex structure of a resulting diagrams.

R. Wille points out in [12] that the theory of multicontexts is closely connected with the triadic setting of Formal Concept Analysis. Three formal contexts $\mathbb{K}_1$, $\mathbb{K}_2$ and $\mathbb{K}_3$ can be regarded as a triadic context. Then a triadic concept lattice can be understood as a natural triadic extension of the three lattices of the concepts $\mathbb{K}_1$, $\mathbb{K}_2$ and $\mathbb{K}_3$. Now, the basic notions of Triadic Concept Analysis [11] are defined.

*Definition 1:* A **triadic context** $\mathbb{K} := (G, M, B, Y)$ consists of a set of objects $G$, a set of attributes $M$, a set of conditions $B$ and a ternary relation $Y$ between them, i.e., $Y \subseteq G \times M \times B$. A triple $(g, m, b) \in Y$ is read: The object $g \in G$ has the attribute $m \in M$ under the condition $b \in B$.

The triadic context can be represented by a three dimensional cross table. In our example (see the figure 4) there are three agents as objects, three behavior as attributes and three common conditions. The ternary relation $Y$ obviously means which of the agents have a behavior on specific condition.

We need some kinds of derivation operators to introduce the triadic concepts. For their definition, it is useful to write $K_1, K_2, K_3$ instead of $G, M, B$. We define for $Z \subseteq K_i \times K_j$, $A_i \subseteq K_i$ and $\{i, j, k\} = \{1, 2, 3\}$ with $i < j$:

- $Z^{(}k) := \{a_k \in K_k \mid a_k, a_i, a_j$ *are related by* $Y$ *for all* $(a_i, a_j) \in Z\}$
- $A_j^{(i,j,A_i)} := A_i^{(j,k,A_j)} := \{a_k \in K_k \mid a_k, a_i, a_j$ *are related by* $Y$ *for all* $a_i \in A_i$ *and* $a_j \in A_j\}$
- $A_k^{(k)} := \{(a_i, a_j) \in K_i \times K_j \mid a_i, a_j, a_k$ *are related by* $Y$ *for all* $a_k \in A_k\}$

Thus, triadic concepts can be introduced as a natural generalization of dyadic concepts:

*Definition 2:* A **triadic concept** of a triadic context $\mathbb{K}$ is defined as a triple $(A_1, A_2, A_3)$ of subsets $A_i \subseteq K_i$, where $i \in \{1,2,3\}$ with $A_k := (A_i \times A_j)^{(k)}$ for $\{i,j,k\} = \{1,2,3\}$ with $i < j$. The sets $A_1$, $A_2$ and $A_3$ are called *extent*, *intent* and *modus* of the concept $\mathfrak{c} := (A_1, A_2, A_3)$ and are denoted Ext($\mathfrak{c}$), Int($\mathfrak{c}$) and Mod($\mathfrak{c}$). The set of all triadic concepts of a triadic context $\mathbb{K}$ is denoted by $\mathfrak{T}(\mathbb{K})$.

Mathematical structure theory of $\mathfrak{T}(\mathbb{K})$ is elaborated in [11] and [3]. However, let me state some simple properties of triadic concepts:

Suppose $\mathbb{K} := (K_1, K_2, K_3)$ is a triadic context. Then for $X_i \subseteq K_i$ and $X_k \subseteq K_k$ with $\{i,j,k\} = \{1,2,3\}$ we set

$$A_j := X^{(i,j,X_k)}$$
$$A_i := A^{(i,j,X_k)}$$
$$A_k := A^{(j,k,A_j)}$$

Notice that a dyadic context $\mathbb{K} := (G, M, I)$ can be understood as a triadic context $\mathbb{K}_t := (G, M, \{b\}, Y)$ with only one condition $b$ and $Y := I \times \{b\}$. Then, the mapping $(A_1, A_2) \rightarrow (A_1, A_2, \{b\})$ is a bijection from $\mathfrak{B}(\mathbb{K}) \backslash \{(G,M)\}$ to $\mathfrak{T}(\mathbb{K}_t) \backslash \{\mathfrak{o}_3\}$. We refer to [10]. Thus, the triadic theory can be understood as a generalization of the dyadic theory.

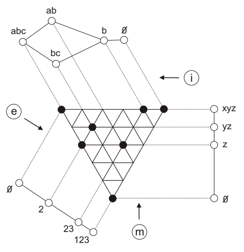
**The triadic diagram:** It is a symmetric structure, for the sets of objects, attributes, and conditions are all treated equally since none of them is preferred to the others. It could be drawn as a triangular graph. A simple example is shown in the figure 5. It consists of three orderings that represent all extents, intents and moduses of all triadic concepts. There is a possibility to create a complete lattice, *side lattice*, for a set of all extents as well as intents and moduses. These lattices are usually drawn along the sides of a triangular graph. Each parallel line from one vertex to the opposite side represents individual extent (intent and modus, respectively). In the vertices of the triangular graph there are bottom concepts of side lattices. The triples, triadic concepts, are represented by the circles in the triadic diagram.

The size of a triadic diagram is the only disadvantage of such visualization. Usually, the sets of objects, attributes or conditions could contain tens of elements. That is why the number of all triadic concepts grows rapidly. 3D visualization is more suitable in practical applications, programs for a larger data collection. Then X, Y and Z axes represent the three dimensions, the set of extents, intents and moduses.

*Example 1:* An illustrative example of a triadic diagram. First, there is a three dimensional cross table in the figure 4 that represents a triadic context $\mathbb{K} := (G, M, B, Y)$, where:
$G := \{1,2,3\}$ . . . the set of agents
$M := \{a,b,c\}$ . . . the set of behaviour
$B := \{x,y,z\}$ . . . the set of conditions
All triadic concepts and related diagram is shown in the figure 5.

| $x$ | a | b | c | $y$ | a | b | c | $z$ | a | b | c |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | | | 1 | | | | 1 | | | |
| 2 | | × | | 2 | | × | × | 2 | × | × | × |
| 3 | | × | | 3 | | × | | 3 | × | × | |

Fig. 4 A 3-dimensional cross table for the triadic graph of the figure 5.



| i | Ext($\mathfrak{c}_i$) | Int($\mathfrak{c}_i$) | Mod($\mathfrak{c}_i$) |
|---|---|---|---|
| $\mathfrak{c}_0$ | $\emptyset$ | abc | xyz |
| $\mathfrak{c}_1$ | 2 | abc | z |
| $\mathfrak{c}_2$ | 2 | bc | yz |
| $\mathfrak{c}_3$ | 23 | ab | z |
| $\mathfrak{c}_4$ | 23 | b | xyz |
| $\mathfrak{c}_5$ | 123 | abc | $\emptyset$ |
| $\mathfrak{c}_6$ | 123 | $\emptyset$ | xyz |

Fig. 5 The triadic graph and list of all triadic concepts.

## 7 Triadic FCA within MAS

Output of FCA is represented by a list of concepts and/or by a trilattice diagram. Each concept give us the information on common behavior of agents in dependence on some conditions. Such agents can substitute each other. Agent and its behavior is analysed in terms of whole system. FCA gives us the possibility to determine and predicate agent behavior. An application usage of FCA will be demonstrated on particular Multi Agent System. The following text describes possible tasks that could be solved by the MAS.

### 7.1 Illustrative usage of FCA

For the sake of simplicity, let us consider *accessible*, *deterministic*, *static* and *discrete* environment of MAS. Let define the main part of such MAS:

- **O**: the set of objects, namely static elements of the environment, e.g. roads and crossroads.
- **A**: the set of agents. We consider *mobile* agents such as cars that can move along the environment.

More precisely they move along the objects, such as roads.

- **R**: the set of relations between agents and objects. It represents a possible movement of some agent along some object. We can find many restrictions of such movements. For example, agents can have features which impose limitations on their movement (a tractor cannot move along a motorway because of its speed, or the trucks cannot be on a village way from 16:00 PM to 6:00 AM next day).

Consider the 3-ary relation $R3 \subseteq WHERE \times WHAT \times WHO$ for the usage of triadic FCA in the above defined MAS. The concrete usage will be shown on a small instance of an imaginary city that will simulate the reality. The example is focused on the traffic simulation.

The whole geographical area is covered by several roads and crossroads. Each road type is connected with traffic restrictions, e.g. trucks can not move along the lanes. Such a system of roads represents a static structure of MAS. Hence in the above defined MAS, each road or crossroad is represented by an object $o \in O$. The inhabitants of the city make up the dynamic part of the system. Each inhabitant is an agent $a \in A$ of MAS. Then we have following context $\mathbb{K} := (G, M, B, Y)$, where $G$ is a set of places, $M$ is a set of traffic violations, $B$ is a set of agents and $Y$ is a ternary relations that represents the above defined kind of the relation $WHERE \times WHAT \times WHO$. $(g, m, b) \in Y$ if and only if an agent $b \in B$ has committed a traffic violation $m \in M$ on a place $g \in G$. Then the triadic concepts can be computed.

After having constructed the three dimensional incidence matrix, a concept list and a triadic lattice are computed. The process of such a computation as well as its integration into MAS will be described in the next section. However, the resulting concept list gives rise to several questions. Particular concepts can be analyzed. We look for common features of agents' behavior, that lead to committing traffic violations.

## 8 FCA integration within MAS

Particular implementation of FCA within MAS depends on many aspects, however, the main idea consists in a direct integration of FCA algorithms with MAS framework. An alternative approach consists in the representation of FCA as an agent of such a framework; this is the way we have chosen because it is in accordance with the FIPA specification [6].

### 8.1 Simplified scheme of FCA integration

The figure 6 represents a basic scheme of FCA integration. The following points summarize the basic description of each level:

1. A modeling framework is placed on the lowest level. The information on requirements specification as well as the real-world image make input data for modeling. On the other hand, outputs of this level are complete MAS Model and agents that behave according to defined MAS Model. Agents represent results of the afore mentioned processes and they are described in the form of source codes. The introduced **Agent**Studio application covers this level of pyramid.

2. The second level is represented by some MAS framework. In this case, JADE framework has been chosen. First, JADE platform is started, which initializes *agent containers* (see more details in [1]). Second, all agent are started. After then, the "FCA"-agent is created which collects all system information and stores them in incidence matrices. FIPA [6] defined basic elements of MAS as well as the structure of agents.

3. The top level consists of the FCA framework. It is a set of tools that comprises the 3D lattice visualization, FCA algorithms for biadic and triadic concept analysis including improved algorithms with a new storage system, analytical tools, etc. It allows the users to see the current situation of MAS in a tri-lattice.

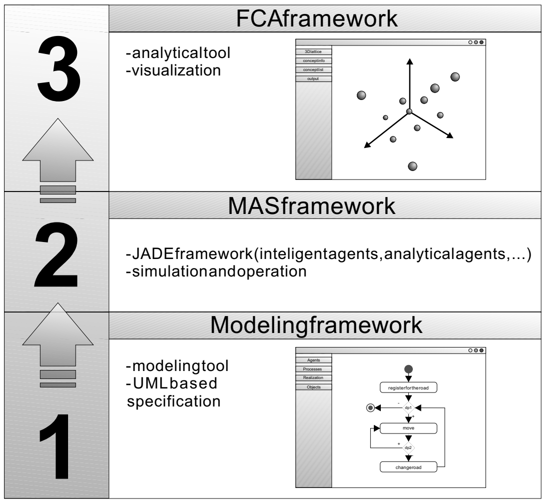


Fig. 6 The basic scheme of FCA integration.

### 8.2 FCA visualization tool

After having constructed the three dimensional incidence matrix (agents, behavior, conditions), a concept list and a triadic lattice are computed. The process of such a computation as well as its integration into MAS have been described in the previous text. However, the resulting concept list gives rise to several questions. First of all, we can look for such of agents which behave in the same way depending on surrounding environment (conditions). All agents in one equivalent classes can substitute each other. Practically, this make whole system more flexible and stable. A visualization tool was created. It is connected to analytical methods and displays particular situation of MAS. It shows the

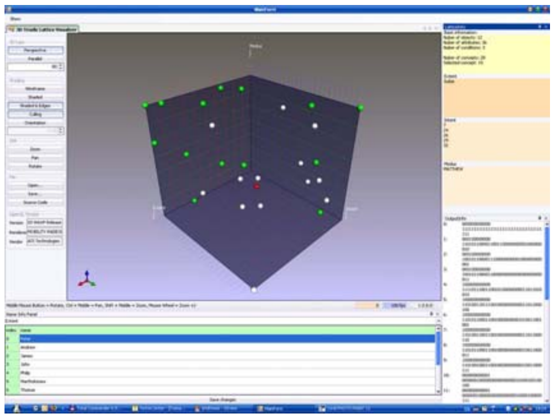


Fig. 7 The screen-shot of FCA visualization tool.

relations between agents, their behaviors and conditions of such relations.

## 9 Conclusion and Future Work

This paper is concerned with the MAS technology especially with specification of MAS Model. The specification tools must be designed with respect to skills of standard users that will be "modelers" of such MAS and that will determined all objectives, requirements and behavior of whole MAS from the real-world point of view. Till now, a meta-model of MAS, its elements and relationships have been specified. However, just theoretical conclusions are not sufficient themselves. Now, a new application called "**Agent**Studio" is in progress. It makes it possible to specify agent behavior with all above mentioned extensions. Next step consists in MAS analysis based on Formal Concept Analysis. The particular analytical tools have been integrated into MAS development process. The main goal of our future work is to create a complex software which will be based on the clear methodology and which will control whole MAS life cycle.

## 10 References

[1] F. Bellifemine. Java Agent Development Framework Documentation. http://jade.tilab.com/, 2005.

[2] K. Biedermann. How triadic diagrams represent conceptual structures. In *Proceedings of the 5th International Conference on Conceptual Structures*, volume 1257 of *LNAI*, pages 304–317, Berlin, 1997. Springer.

[3] K. Biedermann. Powerset trilattices. In *Proceedings of the 6th International Conference on Conceptual Structures: Theory, Tools and Applications (ICCS-98)*, volume 1453 of *LNAI*, pages 209–224, Berlin, 1998. Springer.

[4] J. Ferber. *Multi Agent Systems, An introduction to Distributed Artificial Intelligence*. 1993.

[5] F. Lehmann and R. Wille. A triadic approach to formal concept analysis. *Lecture Notes in Computer Science*, 954:32–45, 1995.

[6] J. Odell. The Foundation for Intelligent Physical Agents. http://www.fipa.org/, 2006.

[7] M. Radecký. Intelligent Selection of Realizations within the Agent Behavior. In *ECMS 2006*, 2006.

[8] M. Radecký and P. Gajdoš. Process and Logic Approaches in the Intelligent Agents Behaviours. In *EJC 2006*, pages 264–270, 2006.

[9] M. Radecký and I. Vondrák. Agents and Their Behavior's Reconfiguration. In *ECEC 2006*, pages 113–120, 2006.

[10] L. Schoolmann. Triadic concept graphs and their conceptual contents. In B. Ganter and R. Godin, editors, *ICFCA*, volume 3403 of *Lecture Notes in Computer Science*, pages 285–298. Springer, 2005.

[11] R. Wille. The basic theorem of triadic concept analysis. *Order 12*, pages 149–158, 1995.

[12] R. Wille. Conceptual structures of multicontexts. In *Conceptual Structures: Knowledge Representation as Interlingua. Proceedings of the 4th International Conference on Conceptual Structures*, volume 1115 of *LNAI*, pages 23–39, 1996.