Annals of "Dunarea de Jos" University of Galati
Fascicle I. Economics and Applied Informatics
Years XXI – nº1/2015
ISSN-L 1584-0409 ISSN-Online 2344-441X
www.eia.feaa.ugal.ro

A Formal Definition for Expert Systems used in Real-time Applications

Vasile MAZILESCU [*]

A R T I C L E I N F O

Article history:
Accepted April 2015
Available online May 2015

JEL Classification
C63, M15

Keywords:
Intelligent Knowledge Management System, Formal definition of an Expert System

A B S T R A C T

The present paper is situated on the grounds of research in the field of symbolic Artificial Intelligence systems, applied to the new Knowledge Management Systems. The basic feature of these systems is represented by the processing of the fuzzy knowledge involved in the synthesis of some decisions. The work reported in this paper serves to promote an Intelligent System that can operate in dynamic and uncertain environments based on a formal definition of an expert system. We can develop and justify thus a series of modelling and design techniques for Intelligent Knowledge Management Systems (IKMS), as well as methods for the analysis of expert systems performance, and, of a fuzzy expert system in particular, between which there are strong similarities. We will also outline a number of differences between conventional problem solving systems and IKMS, the links between expert systems and those of structural and functional planning, the analogy between the model of the problem or business process and the problem domain.

1. Introduction

There are highlighted the ratios between conventional and intelligent management, by the use of planning and multi-agent systems. Knowledge-based systems cu real-time functioning bear features that the majority of classic system do not have: reasoning is evolutionary and non-monotonous due to the dynamic nature of the application, and events can change the status of the expert management system. The management architectures based on symbolic techniques acquire characteristics specific to the domain of the problem and to the type of expert system within the management structure. Planning is a difficult problem and may become much simpler if certain restrictions are applied. Different planning problems may be defined by restricting the type of operators, imposing a series of limitations to the number of preconditions and post conditions. Knowledge-based knowledge management (KBKM) focuses on applications of knowledge-based systems (KBS) tailored to knowledge management (KM) problems. The terms express the use of KBS to enable knowledge management (Bhatt, 2000). KM practitioners and research scientists have been implementing various frameworks to address pragmatic KM problems reusing decades of technology developed for KBS. Therefore, when we talk about knowledge-based knowledge management, we talk about the overlap of knowledge-based systems and knowledge management. It is easy to understand the scope of KM by focusing on a knowledge process that collects, stores and reuses knowledge leveraging it and making organizational the knowledge that once was individual. This knowledge process supports organizational goals by controlling the collection, storage, and use of knowledge (Bhatt, 2001). Accordingly, KM applications can be envisioned along the dimensions of a knowledge process that fundamentally performs knowledge tasks to support, steer and control organizational goals. Expert systems (ES), case-based reasoning (CBR), and ontologies are examples of relevant knowledge-based methodologies that have much to contribute to KM systems because they manipulate knowledge to implement various tasks. Although KM practitioners frequently comment that KM is not a technology problem, it is also the case that most KM solutions include an element of technology. It would seem that using technologies for collecting, storing, and reusing knowledge would be critical to consider in any KM effort. Therefore, it is natural that the understanding of a knowledge process differs in different organizations; e.g., a knowledge process in one organization has a different set of tasks than in another. Sometimes these knowledge processes differ on the surface, though their conceptual model is the same.

It is difficult to distinguish a knowledge process from KM and that may be the reason why there are so many definitions on KM as a discipline. On the one hand we represent a knowledge process as a cycle – one in which KM manifests itself but can we also say that each of the tasks that are part of a knowledge cycle represent a kind of KM. We can describe what role they play in the knowledge process and we can also describe how they support the tasks of the knowledge process (another form of KM) (Mazilescu, 2012).

[*] Faculty of Economics and Business Administration, "Dunarea de Jos" University of Galati, Romania. E-mail address: vasile.mazilescu@ugal.ro (V. Mazilescu)

Expert systems represent one way that expertise can be captured, coded, and reused. They are embedded in IKMS and are based on Artificial Intelligence techniques. Fundamentally, an expert system consists of some representation of expertise, some representation of a problem to be solved, and some mechanism to apply the expertise to a problem. Although expertise may be represented in various forms, one common representation for expertise is in the form of rules. The representation of an expert's knowledge would be a collection of rules that were derived from the expert. A rule consists of two parts: an antecedent and a consequent. The rule's antecedent consists of one or more conditions that specify when and where to apply the rule. If the conditions of the rule are met, then the second part of a rule – the consequent – specifies the actions to take when the conditions of the rule are met. The mechanism that chooses rules to see if they can be used in a problem is called an inference engine. The inference engine checks antecedents of rules, and based on their values performs the actions specified in the consequents of the rules. To maintain the state of a problem being solved, the inference engine uses a special structure to store the state of problem solution. The structure is called short-term memory. Rules are scanned from the knowledge base to determine which apply to the current problem state in short-term memory. When the inference engine identifies a rule, the actions of that rule are carried out, which may result in a change to the problem state in short-term memory. The process repeats itself until it solves the problem, or no condition can be fulfilled or the expert system is explicitly stopped. There are in IKMS conception three large groups of problems that should approach in terms of decision-based applications: human-environment interface, qualitative knowledge modeling and time management. Such applications obviously require dated event operations the life-time of which should be managed by the system, which often works asynchronously with the acquisition and control system.

A real-time expert system shell must also represent imprecise, time and temporal data, encode temporal knowledge, and manage temporal/fuzzy reasoning for allocating temporally interdependent tasks to homogeneous or heterogeneous cooperative agents in dynamic large-scale networks. Planning systems based on Artificial Intelligence use specific models for the problem field, called problem representations and logic argumentation models. Current expert systems have many characteristics in common with planning systems (representation of knowledge, heuristic inference strategies), conceived specifically for communicating with the exterior, while conventional expert systems are strongly encapsulated. Planning systems execute actions in a dynamic manner to produce modification in the state of the problem field. The planner monitors the problem field to progressively obtain information useful for decision synthesis. An explicit loop is being crossed between the actions done by the planner, the problem field, the measured outputs and the planner that uses the outputs to decide the control actions, with a view of reaching the goal. Within expert systems, a similar loop exists: the knowledge base represents the problem field, while the inference engine represents the planner. We have developed an IKMS as an agent that works like a planner. It is realized as a fuzzy real-time expert system shell to meet the challenges of the dynamic environment, like Virtual Organizations (VOs) or Hierarchical Coalitions (HCs). VOs provide an effective instrument to integrate a company's operations with those of other enterprises, to work with customers and create a better product or service, to achieve a faster time to market, and to acquire a higher degree of product customization (Burden, 2009; Lin et al., 2007; Bergamaschi et al., 2006).

Section 2 is a comparative analysis of planning capabilities for IKMS. At higher levels or at the much more abstract levels, every step of the plan has more effects than a step of the plan at a lower level. Section 3 is dedicated for a presentation of our IKMS based on logical events. Section 4 presents our conclusions and future developments.

2. Basic capabilities for IKMS

A Multi-agent planning architecture (MPA) can be an option in the support systems for coalitions because it is a framework for the integration of various technologies into a system able to solve problems which cannot be solved by the independent systems. MPA is widely used in planning the applications, such as coalitions' operations. MPA agents are organized based on the concept of cells planning. Each cell has a managing member that performs the planning of the cell from a group of available agents and distributes the tasks among selected planning agents. Sharing the information generated during the planning process will be held by a central plan, and each agent of the cell can access it. MPA is important for the integration of the different planning solutions. This integration has limits for the development of support systems coalitions, like any other approach that tries to integrate existing instruments and possibilities in a simple framework. The main reason is collaboration between problem solving on different components and the need to be embedded from the beginning into systems. The problem is how to incorporate collaborative requirements into a distribution of processes planning, so that planning the final connection will not be a sum of independent plans (Barachini, 1990; David et al., 2006; Jonsson et al., 2007). Considering that the most important function of the knowledge-based instruments is to support the human user, we must also understand how they interact in the planning the collaboration process. It should be noted that we do not take into account aspects of the interface which could improve the man-agent interaction. The main idea is that we can join up the groups, associated with the development of hierarchical coalitions as support for the agents, by means of a framework in line with an ontology based on constraints and on the appropriate functions. We can argument a MAP framework brings several advantages such as: a well-known environment to represent and build plans; a transparent manner to integrate collaborative concepts that complement the planning skills; opportunities to develop the human-agent mechanisms; support for customizing the intermediaries. Based on this statement, we can have specific problems (Jonsson and Mattsson, 2006; Kim et al., 2003; Stoop and Wiers, 1996).

An expert system is a type of planner, since it emulates the way in which experts make decisions, in the presence of abstract, qualitative and approximate knowledge, in a limited expertise field. Currently, what is necessary is the formulation of a mathematical theory of intelligent planning systems, which operates in a dynamic environment, with real time properties. The *state* of a planner or an expert system describes the situation that defines the problem solving strategy at a given time. A *planning problem* for a system may be defined thus: given an initial state, a desired state and an aggregate of possible actions, let it an aggregate of actions (either partially or totally ordered) be determined, which applied to the initial state leads the system to the desired state. Planning is a difficult problem and may become much simpler if certain restrictions are applied. Different planning problems may be defined by restricting the type of operators, imposing a series of limitations to the number of preconditions and post conditions. For these types of restrictions, the planning problem is polynomial or NP-complete. The computational complexity of planning was investigated. Many authors have been analyzed the general problem of deciding the existence of a solution for the planning task in the context of the STRIPS system and it is demonstrated that this general problem is PSPACE-complete (Bylander, 1994). The planner's outputs are inputs for the problem domain, representing control actions (Lunardhi and Passino, 1995). The outputs of the problem field are inputs for the planner. They are measured by a planner and used to determine evolution in the problem solving process. Furthermore, there are non-measurable exogenous inputs for the problem field (disturbances), which represent the uncertainty associated with it. The measured exogenous input of the planner represents the goal. It is a task of the planner which examines the outputs of the problem field, it compares them to the goal function and determines what actions must be undertaken to reach it. Not all planners are completely autonomous. Some have a user interface, through which the goals may be generated, allowing for certain degrees of intervention of the human element in the planning process. A *solution for the problem* is a sequence of inputs and outputs (possible states), generated with a view to fulfil the purpose. A model for the field of the real problem is thus developed, called the representation of the problem. Planners based on Artificial Intelligence techniques are made from the following important components (McKay and Wiers, 2003): the plan generator, the plan simulator (uses plans in the shape of heuristic decision rules), the execution model for the selected plan, the situation evaluator (optional). The Artificial Intelligence techniques used in the design of planning or expert systems are diverse. They refer especially to representation problems (on must take care in selecting the degree of detail for the mathematical structure used or the allowed modeling power, since too much modeling power may prevent the development of certain components of the planner, the verification and validation of the system), the type of approach (dependant or independent of the field), the type of the planner (hierarchical or not, linear or non-linear, reactive, distributed, encompassing a meta-planner etc.), the type of interactions that may appear in the synthesis of decisions, the forms of searching and re-planning. Due to the strong similarities between the process and the problem field, an analogy may be outlined between the models used for the process and the problem field, on the one side, and between the fundamental systemic concepts on the other. The process is described through stochastic (possibly non-linear) or differential equations, called state and output equations, which describe the dynamics of the process, the structure and its inter-connections (Stadtler and Kilger, 2005). The problem field may be described through symbolic equations. There are still strong mathematical analogies between the two types of systems, studying certain proprieties of special significance in the classic control theory: controllability, observability, stability, system rate. There is a structural analogy between classic control systems and Artificial Intelligence systems (planning, expert system), both in an open loop as well as a closed loop. *Generating a plan* is the process of synthesis of an aggregate of candidate plans, to fulfil the g_i purpose at moment i (the goals can remain fixed or be modified). During the process of plan generation, the system *designs* (through a simulation based on a model of the problem field) the plausible future plan.

The system uses heuristic plans based on decision rules, resource management, success probabilities, in order to choose the plan to execute. *The plan execution module*, translates the chosen plan into actions over the problem field, using techniques of resource allocation. S*ituation evaluator* uses the inputs, outputs and the representation of the problem in order to determine the state of the field. The estimate state of the field is used for updated the model implicated in the design and generation process of the plan. The term of *situation* is preferred in this case to designate an abstract, general view of the state. *The monitoring of the execution* uses the estimated state of the problem field, in order to determine the viable nature of the elaborated plan, in cases of failure launching re-planning actions. In the structure of a planner based on techniques of Artificial Intelligence may also be integrated a *world modeler*, which does the permanent updating of the representation of the problem field. The evaluation of the situation and the monitoring of the execution in open loop systems cannot be assured, being no possibility of re-planning. In this case, the planner is sensible to the variations which appear in the problem field, open-loop planners not being able to reduce this sensibility.

These planners are of interest only for insignificant disturbances, which is not the case with real problems, which need complex and detailed problems. Seeing as the outputs of the system are not detected with inputs, if the problem field is unstable (input-output or internal), then it will never be able to be stabilized in open-loop planning. Open-loop planning absolutely imposes an exact amount of knowledge of the problem field. Since this cannot be obtained, any disturbance may be catastrophic. Open-loop planners present advantages due to their simplicity. If the problem field is stable and the disturbances are insignificant, then these planners may be useful. They also have the advantage of reduced costs, not being necessary measurements of states and outputs related to the problem field. Closed loop planning systems are analogous to closed loop conventional control systems and most of the time, do not use situation evaluation. The monitoring of the execution as well as the re-planning are thus permitted. The planning system examines the difference between the current output situation and the desired goal, with a view to executing certain actions. The error is not so easily obtained, compared to the classics systems of control, seeing as the distance between the fuzzy states is much harder to quantify. Ever more difficult is the evaluation of the similarities between states, based on a given criteria in the case of our fuzzy expert system. These aspects will clearly come into the design of the IKMS. Designing an IKMS as a planning system is a difficult problem, through the variety of the models used (satisfactory from a computational perspective), which must be a reflection of the complexity of the environment in which they operate (Berglund and Karlun, 2007; Chen, 2008).

3. An analysis of our IKMS based on logical events

Real time expert systems are important applications both in terms of Artificial Intelligence techniques, and especially in terms of the purpose for which they are designed (Li and Xiao, 2006). They must operate in the presence of real-time restrictions. These systems are often designed and implemented without a formal analysis of how it interfaces with the process and without a careful examination of how the inference engine exploits a domain's specific knowledge. Regularly, such systems are evaluated by: **i)** *comparison* to how experts solve the problems empirically; **ii)** *studying their reliability* and degree of user-friendliness of the interface; **iii)** *examining the results* of difficult simulations; **iv)** *using* software engineering techniques.

Our IKMS consists of an expert system (ES) and the process (P), as outlined in figure 2. Efforts are concentrated on the class of expert systems based on imprecise knowledge and on the corresponding inference engine, characterizing the process management strategies. We will reduce the structure of an expert system only to the knowledge base and inference engine, which has the role of decision-making control by using the reference inputs $e^{ES}_{n_k}$ and the outputs of the process $e^{ES}_{p_k}$. In this way the results of the expert system's inferential process will be summarized, helping to obtain either the outputs $e^{ES}_{o_k}$, in the form of hypotheses or conclusions, provided to the user, or the potential commands e_{uk}. It is important that the design of the IKMS to allow the synthesis of an expert system based on imprecise knowledge, to perform certain management functions properly. The first step in analyzing the behaviour of a management expert system is to develop a global mathematical model for it, based on partial models, which are used for the representation and qualitative analysis of the expert system and of the process P (problem domain). Must be also analyzed the reasoning process of the rule-based expert system which emphasizes that representation of knowledge involved in the management strategy and the reasoning are more difficult than for controllers, highlighting how to develop a fuzzy expert system and in particular how to choose the conflict resolution strategy. In relation to the integration of the expert system in the structure of management expert system, its results can be used by the human decision-maker in decision-making or can be applied directly in the process.

Logical discrete event systems are a class of discrete event systems with discontinuous and nonlinear equations of motion in relation to the occurrence of events. Although we use the expert systems' terminology established in Artificial Intelligence, there are large differences in the validity of standard models in this field in relation to human cognitive structures specific to management expert systems. Therefore, the objective of the present approach is the analysis and design of a fuzzy expert system embedded in a management structure to support effectively the decision-making process. In this respect, the expert system in this work is able to use the management experience integrated into the process management model, permanently adapting to the current situation through the inference engine. It must be able to plan a finite number of steps to support the appropriate management actions and must be able to reformulate what plan (actions sequence) must be synthesized, by monitoring the current evolution of the process. This type of planning is not the most general possible, but that is not a limitation of the management expert system, since practical considerations indicate that frequently can be planned only a relatively small number of steps (especially in the presence of imprecise knowledge). For a more rigorous planning, expert system state previously defined may be completed the component x^{bt}, representing a vector of state trajectories generated by simulating different plans. In fact, with the planning properties mentioned, the expert system performs at every step a significant amount of reasoning, before synthesizing a decision. These features will be highlighted for the management expert system for the flexible manufacturing system presented in the case study.

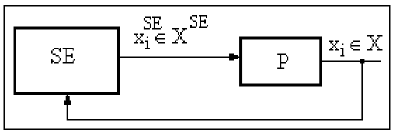

Figure. 1.

Consider a process P (a structure identical to systems interconnected following an imposed scheme) where data and knowledge related to the current state and management strategies are limited because they have a high degree of imprecision. Time is a discrete measure. If the process works as a nondeterministic finite automaton, and expert system will function as a deterministic finite automaton, the closed loop management expert system can be modelled as a nondeterministic automaton, whose output is the output of the process (fig. 1). Let X be the finite set of states of the process and X^{SE} is the finite set of states of the expert system. The model of the process may have, in this case, the following form: P = (X, X^{SE}, δ, X_0) where $\delta : X^{SE} \times X \to P(X) - \{\varnothing\}$ is the transition function of the process, and $X_0 \subset X$ is the set of initial states of P. Similarly, can be constructed the expert system model in the form SE = $(X^{SE}, X, \xi, x_0^{SE})$, where $\xi : X \times X^{SE} \to X^{SE}$ and $x_0^{SE} \subset X^{SE}$ is an initial state of the expert system. Due to the non-deterministic character of P, there are generally infinite strings of α formulas composed of states of P, which are generated by the management expert system. These strings are called ES-admissible strings (the reason for attaching ES symbol is that we may consider different structures of expert systems for a given process P in order to achieve a goal).

Interpretation of the model. Let N be the set of natural numbers. Activation of the management expert system produces a sequence (infinite) of pairs $(x_k^{SE}, x_k)_{k\in N}$ where x_k^{SE} is the expert system's state, and x_k is the state of the process P, at the moment k. Operation of the management expert system is defined as a sequence of pairs (x_k^{SE}, x_k). For $(\forall)$ k, $x_{k+1}^{SE} = \xi(x_k, x_k^{SE})$, and $x_{k+1} = \delta(\xi(x_k, x_k^{SE}), x_k)$. The output of the process is $\{x_k\}_{k\in N}$. There is little asymmetry in the definition of management expert system's operation. We may write the transition function of the process so that x_{k+1} to be the value of $\delta(x_k^{SE}, x_k)$ and not of $\delta(x_{k+1}^{SE}, x_k)$. If the expert system's input is x_k and the result will be x_{k+1}^{SE}. If x_{k+1}^{SE} is the process' input, then the management expert system's output is x_{k+1}. We define, in terms of the selected model, the fulfilment of a temporal formula. We want a set of temporal formulas to describe the functioning of the management expert system. We answer the following questions for a fixed system model: When the expert system works? When we measure the size of the output of the process? There must be a device in the process P to measure the current state x_k based on which is calculated x_{k+1}^{SE}.

4. The Model of the Expert System Based on Fuzzy Knowledge

A real time expert system must react quickly to a series of events and respond with a certain delay. Expert fuzzy systems involved in decision-making for processes management have embedded in their structure hierarchically organized knowledge that allow associations and reasoning chains based on fuzzy logic inferences.

The Expert system has three types of inputs: reference input events $e_{r_k}^{SE} \in E_r^{SE}$, the outputs of the process $e_{p_k}^{SE} \in E_p^{SE}$ and user inputs IU. Based on these inputs and on its current state, the expert system can synthesize allowed input command events for the process $e_{o_k}^{SE} \in E_o^{SE}$, emulating in this way the way in which human decision-maker coordinates in loop the use of evidence knowledge from the process, the reference inputs and knowledge from its own knowledge base. Fuzzy expert system models the cognitive process used in decision making, and the interaction between the inference engine and the knowledge base forms the inference cycle.

Definition: A fuzzy expert system (SEF) can be modelled as follows:

SEF = $(\mathbf{X}^{SE}, E^{SE}, f_e^{SE}, \delta^{SEF}, g^{SEF}, x_0^{SE}, E_v^{SE})$

where: $\mathbf{X}^{SE} = X^b \times X^{int}$ is the *set of states* x^{SE} of the expert system, X^b is the set of imprecise states x^b from the facts base, and X^{int} is the set of internal states x^{int} of the inference engine. $E^{SE} = E_i^{SE} \cup R \cup E_o^{SE}$ is the set of events of the expert system, where takes place the following relationship $E_i^{SE} \subset P(E_p^{SE} \cup E_r^{SE} \cup IU) - \{\varnothing\}$. E_i^{SE} is the set of all input events of the expert system (reference inputs E_r^{SE}, the output events of the process E_p^{SE} that may arise and in relation to which the system must be able to respond in real time, and user inputs).

$R = \{R_1, ..., R_n\}$ is the set of fuzzy rules of the expert system.

$E_o^{SE} \subset P(E_u \cup IC) - \{\varnothing\}$ is the set of logical events out of the expert system. They are sets of command events allowed for the process or hypotheses / conclusions (IC) obtained from the inference process.

$g^{SE}: X^b \times X^{int} \to P(E_i^{SE} \cup R) - \{\varnothing\}$ is the activation function of fuzzy expert system.

$f_{e_k}^{SE}: X^b \times X^{int} \to X^b \times X^{int}$ for $e_k \in P(E_i^{SE} \cup R) - \{\varnothing\}$ is the fuzzy states transition function.

$\delta^{SE}: X^b \times X^{int} \to E_o^{SE}$ are the output functions of fuzzy expert system.

x_0^{SE} is the initial state of expert the system, with $x_0^{SE} = (x_0^b, x_0^{int}) \in X_0^{SE} \subset X^{SE}$.

$E_v^{SE} \subset E^{SE}$ is the set of allowed trajectories of the expert system in inference loop (physically realizable trajectories due to events).

Occurrence of the input event $e_{i_k}^{SE} \in E_i^{SE}$ is always accompanied by the activation of a rule, of several rules or even of more instances of the same rule for imprecise knowledge ($R_i \in R$ or $\overline{\mathfrak{R}}_i \subset R_i$ with $\overline{\mathfrak{R}}_i$ the set of instances of rule R_i). Thus the inference engine will operate by updating the set of states $\mathbf{X}^{SE}$. A rule $R_i \in R$ cannot be activated alone, the inferential cycle being launched only if there is a change in the process (which is reflected by the changes in its outputs), at the level of the reference inputs or user inputs. In this case, the expert system's input events must contain (for the classical case) exactly one rule $R_i \in R$ and a single command input event $e_{i_k}^{SE} \in E_i^{SE}$. Each event $e_{i_k}^{SE}$ contains at most one event $e_{p_k}^{SE} \in E_p^{SE}$ and a reference input event $e_{r_k}^{SE} \in E_r^{SE}$. These aspects can be modelled using the notion of *allowed trajectory* of the expert system in inference loop. Operators $f_{e_k}^{SE}(x^{SE})$ corresponding to events $e_k \subset g^{SE}(x^{SE})$ describe the update of the knowledge base and of the internal states of the inference engine when the process' outputs, the reference, or both change, and certain rules $R_i \in R$ are activated. Function $\delta^{SE}(x^{SE})$ describes the expert system's outputs and the possible input commands allowed on the process P. It should be noted that events such as $e_{o_k}^{SE}$ are not allowed as long as the expert system is running (changes its state x^{SE}). This system can control the activation of the input command events of the process but not the set of events e_{dk}. Premises of the rules are functions $P_i: X^b \times E_i^{SE} \to [0,1]$; $P_i(x_k^b, E_{i_k}^{SE}) \in [0,1]$ indicates the degree of truth of premise P_i at moment k. Functions P_i will be used to measure the degree to which a rule (or an instance of it) may be enabled. Also, a consequent of rules is a function defined on $X^b \times E_i^{SE}$ with values in various sets (X^b, $X^b \times E_o^{SE}$, $X^b \times X^{int} \times E_o^{SE}$). The codomain of these functions is determined for each case against a series of properties of the management expert system: conflict resolution strategy, the type of knowledge. For a rule R_i, the event $e_k = \{R_i, e_{i_k}^{SE}\} \subset g^{SE}(x_k^{SE})$ can occur only if the filtering degree of the premise is satisfactory at the moment k, for the state x_k^b and input event $e_{i_k}^{SE}$. If event $e_k \subset g^{SE}(x_k^{SE})$ occurs, then the new state of the expert system $x_{k+1}^{SE} = f_{e_k}^{SE}(x_k^{SE})$ is obtained by applying the consequent functionto the state $x_k^b \in X^b$, in order to obtain the state x_{k+1}^b and by updating the inference engine's state $x_k^{int} \in X^{int}$.

Including input events E_i^{SE} in the rules base, allows the expert system to infer conclusions relative to the outputs and inputs of the process (reference, user commands), perceived directly as a part of the model. This aspect corresponds to the use of variables in *OPS5*-type conventional expert systems. Inference engine's modelling is limited to its basic functional stages: *restriction, filtering, conflict resolution, execution.* For the case of imprecise knowledge, the most important stage in terms of computation is the filtering stage, from which is obtained the set of conflicts:

$$MC_k = \{\overline{\mathfrak{R}}_i : \{\overline{\mathfrak{R}}_i, E_{i_k}^{SE}\} \subset g^{SE}(x_k^{SE})$$

so that the premise of rule (instance) $\overline{\mathfrak{R}}_i \in R$ has a satisfactory filtration degree for $e_{i_k}^{SE}$ at the moment k}. In the selection phase one rule is chosen from the set of conflicts MC_k, using different conflict resolution strategies. Below we summarize the basic operations of an expert system for processes management:

i) acquisition of events $e_{i_k}^{SE}$; **ii)** Synthesis of the set of conflicts MC_k in the filtering stage, using the set of rules R and the events $e_{i_k}^{SE}$, the facts from the facts base (blackboard) and the updated values of the variables $\mathbf{x}^{int}$; **iii)** Determining, based on conflict resolution strategies, the rule R_i (its instance) in MC_k, in order to be activated; **iv)** Performing the actions of consequent function of the selected rule (not before achieving unification and the spread of all parameters for the case of fuzzy knowledge). This stage involves updating the knowledge base, the inference engine's states and generating the inferred conclusions. Events occur in the expert system so that its operation is synchronous with the process (if an event occurs in the process will produce the activation of inference engine) and with reference input events, for which the system can generate events $e_{o_k}^{SE} \in E_{o_k}^{SE} \subset P(E_u \cup I/C) - \{\varnothing\}$.

The IKMS Inference Engine Algorithm

Initializations: k = 0, $\Pi_k[i] = 0$, $N_k[i] = 0$, $\Theta_k[i] = 0$, $K_k[i] = 0$, $\Lambda_k = 0$, $W_k = 0$, $ra_k^i = 0$, $a_k[i, j]$.

- **Step 1.** *(Identify the current state).* Pattern-matching is performed using the compiled fuzzy knowledge structure, specific to IKBSCP system, in relation to the occurrence of a certain type of event at expert system's input. At the end of this stage we get the set of conflicts MC_k.

For r = 1 *to* n_r *do*

If ($r \in MC_k$) *then* $ra_k^r = 1$ {Determine the rules that can be activated}

If (there is exactly one rule r′ so that [illegible] = 1) *then* execute rule r′

If (there is not a rule r′ so that [illegible] = 1) *then stop* {I/C = ∅ or call the control module)

- **Step 2.** *(Conflict Resolution).* This step uses different strategies but which are not all active for a particular fuzzy knowledge model or for a given situation.
- **Step 3.** *(Execution)*

e′ = {r′, e_k^i} {we highlight the type of event for the IKMS}

(x_{k+1}^b, x_{k+1}^{int}) = f^{CES}(x_k^b, x_{k+1}^{int}) {as a result of the occurrence of event e′, the expert system's state is updated, i.e. the state component in the facts base and the component x_{k+1}^{int}, whose update we present below}

$\Lambda_{k+1}(r') = 1$ {Remove the rules from the set of conflicts, using refraction}

For r = 1 *to* n_r *do*

If ($r \in MC_k$) *then* $W_{k+1}^r = W_k^r + 1$ {Increment the age of each rule from MC_k}

For r = 1 *to* n_r *do*

If (A[r, r′] = 1) *then* $\Lambda_{k+1}(r) = 0$; $W_{k+1}^r = 0$ {Allows the rules affected by the activation of rule r′ to be considered within the set of conflicts and resets the age of these rules to 0}

End

In the phase of selection based on refraction, if the expert system inference engine, which exploits the management linguistic model developed in accordance with a specific strategy used by certain human decision-maker, cannot respond to the current situation, stop may be replaced by resetting the values of ra_k^r to the previous values, before applying the refraction and continuing the execution, so that the expert system to use the selection based on refraction in conflict resolution only if this strategy effectively reduces the set of conflicts MC_k. Note, that (x_{k+1}^b, x_{k+1}^{int}) = f^{SE}(x_k^b), as a result of the occurrence of an event of type e′= {r′, e_k^i}, is the application of the state modifying operator, equivalent to the application of the consequent formula of rule r′, selected for execution in the state x_k^b. In this way are updated also the components of the internal state vector x^{int} of the inference engine. There was no evidence, within the inference engine algorithm structure, of the mode of action of output function G_k^{SE}, as it has a character specific to the management model. This will be validated practically for the flexible production system, chosen for a case study to highlight the characteristics of IKMS system. The closed-loop control expert system can be modelled like a nondeterministic state machine, whose outputs are the process outputs.

5. Conclusions

The use of temporal aspects refers to the design of those tools to integrate time in control economic applications (e-commerce, Knowledge Management Systems, Virtual Organizations, Multi-agent Systems). These aspects are formally found on the inference engine algorithms, able to make full use of the specific knowledge to the process control. We assume that the process operates like finite nondeterministic state machine, while the expert system will operate like a finite deterministic state machine. The problem domain must be defined as a collection of problems that the expert system desires to solve. In conventional control, the plant is a dynamical system, described with linear or non-linear differential/ difference equations. An Intelligent Knowledge Management System based on knowledge control and planning models consists of the planner or the inference engine, the problem domain, the exogenous inputs, and their interconnections. An expert system can be modelled using predicate or temporal logic or other symbolic techniques such as finite state machine. Although the KMSs are frequently being used to perform complex control functions, most often it is the case that no formal analysis of the dynamics is conducted because mathematical analysis of such systems is often considered to be beyond the scope of conventional control theory. Mathematical Logics have been developed in three great directions: i) *The Theory of Models*. The model is considered the main concept of first-degree languages semantics. The main objective of the theory of models is to describe the models from certain axiomatic theories, in order to highlight their mathematical structures (algebraic, topological); ii) *The Demonstration Theory*. The notion of demonstration is important here, and one can study complexity-related problems of various demonstrations on a single theorem, according to different deductive systems or languages (regularity, equivalences, similarities); iii) *The Computability Theory*. This is the case in which one studies the intrinsic notion of computable function, with its various models: the Turing machine, recursive functions and combinatory logic with a set of properties. Starting from these notions we can study the decidability concept, together with a study of various models used to demonstrate the decidability or non-decidability of mathematical theories. The theory of abstract complexity that generally uses different variants of the Turing machine (as an elementary calculus model to evaluate the necessary calculus resources to solve certain given problems) allows us to establish the complexity classes for decidable problems. We must also highlight the unity of these three great aspects. Calculus and decidability notions are strongly inter-related, and we can thus establish precise correspondences between them. The theory of models can contribute as regards decidability problems and can also provide the semantic justification of the rules of inference and of automated demonstration systems. This unit is an essential characteristic of Logics which represents a fundamental theoretic aspect. From the symbolic Artificial Intelligence perspective, we can find tendencies specific to the three characteristics that were presented above: semantics and the theory of models, demonstration and reasoning, computability and complexity, but often with different methods compared to theories applied only to mathematical logics. As regards the contribution of Logics in terms of Artificial Intelligence systems, we can mention: i) on the theoretical level, logic contributes with a series of creation elements and methods such as the syntax/semantics/decision trilogy, coherence (verifying the coherence of the knowledge-base, maintaining the truth systems), decidability (for example, absence logic is not semi-decidable, contrary to first-degree logic, certain temporal logic are decidable, others are not), complexity of the decision methods. Logic is thus a reference model for the foundations of symbolic Artificial Intelligence systems. ii) The formalisation of different types of reasoning. Logic has a normative role even in the absence of some certain knowledge. Automated demonstration, representing knowledge languages, logical programming languages they all represent means that support and integrate reasoning methods.

References

1. *Barachini, F. (1990). Match-Time Predictability in Real-Time Production Systems, in Springer-Verlag, Lecture Notes in Artificial Intelligence, pp. 190-203. Retrived May 8, 2012, from http://link.springer.com/chapter/10.1007%2F3-540-53104-1_42.*
2. *Bergamaschi, S., Gelati, G., Guerra, F., Vincini, M. (2006). An intelligent data integration approach for collaborative project management in virtual enterprises, World Wide Web 9,35–61. doi 10.1007/s11280-005-2322-7.*
3. *Berglund, M. and Karlun, J. (2007). Human, technological and organizational aspects influencing the production scheduling process, International Journal of Production Economics, Vol. 110, No. 1/2, 160-174.*
4. *Bhatt, G. (2000). Organizing knowledge in the knowledge development Cycle, Journal of Knowledge Management, Vol. 4 No. 1, 15–26.*
5. *Bhatt, G. (2001). Knowledge management in organizations: examining the interaction between technologies, techniques, and people, Journal of Knowledge Management. Kempston: Vol. 5 No. 1, 68-75.*
6. *Burden, D.J. (2009). Deploying embodied AI into virtual worlds, Knowledge-Based Systems 22, pp. 540–544.*
7. *Bylander, T. (1994). The computational complexity of propositional STRIPS planning, Artificial Intelligence, nº 69, 165-204.*
8. *Chen, T.Y. (2008). Knowledge sharing in virtual enterprises via an ontology-based access control approach, Computers in Industry 59, pp. 502–519.*
9. *David, F., Pierreval, H., Caux, C. (2006). Advanced planning and scheduling systems in the aluminium conversion industry, International Journal of Computer Integrated Manufacturing, Vol. 19, No. 7, 705-715.*
10. *Jonsson, P. and Mattsson, S. (2006). A longitudinal study of material planning applications in manufacturing companies, International Journal of Operations and Production Management, Vol. 26, No. 9, pp. 971-995.*
11. *Jonsson, P., Kjellsdotter, L., Rudberg, M. (2007). Applying advanced planning systems for supply chain planning: three case studies, International Journal of Physical Distribution & Logistics Management, Vol. 37, No. 19, 816-834.*
12. *Kim, Y.G., Yu, S., Lee, H. (2003). Knowledge strategy planning: methodology and case, Expert Systems with Applications, Volume 24, Issue 3, pp. 295-307.*
13. *Li, H. and Xiao, R. (2006). A multi-agent virtual enterprise model and its simulation with Swarm, International Journal of Production Research 44, 1719–1737.*
14. *Lin, C.H., Hwang, S.L., Wang, M.Y. (2007). A reappraisal on advanced planning and scheduling systems, Industrial Management & Data Systems, Vol. 107, No. 8, 1212-1226.*
15. *Lunardhi, A.D. and Passino, K.M. (1995). Verification of qualitative properties of rule-based expert systems, in Applied Artificial Intelligence, pp. 587-621.*
16. *Mazilescu, V. (2012). A Knowledge Management System Embedded in the New Semantic Technologies, Chapter 1, in New Research on Knowledge Management Technologies, Intech, Rijeka, pp.1-22.*
17. *McKay, K.N. and Wiers, V.C. (2003). Planning, scheduling and dispatching tasks un production control, Cognition, Technology and Work, Vol.5, No. 2, 82-93.*
18. *Stadtler, H. and Kilger, C. (2005). Supply Chain Management and Advanced Planning-Concepts, Models, Software and Case Studies, 3rd ed., Springer, Berlin.*
19. *Stoop, P.M. and Wiers, C.S. (1996). The complexity of scheduling in practice, International Journal of Operations & Production Management, Vol. 16. No. 10, 37-53.*