# Intelligent Agents in Support System for Discrete Processes Control Design

**Budinská I., Frankovič B., Dang. T.T.**

Inrtitute of Informatics SAS

Dúbravská cesta 9

845 07 Bratislava, Slovakia

utrrfran, utrrbudi, utrrtung@savba.sk

*Abstract: The paper describes a multi-agent support system (MASS) for discrete processes control design. The system is intended for discrete production processes. However some functions may work for continuous production processes as well. The proposed support system consists of several intelligent agents. The core of the systems is created by a database of modeling, control and simulation tools (algorithms) for discrete manufacturing processes and an intelligent decision system. The goal is to utilize information from the system's database and support a process control design on the basis of designer's requirements. The system consists of three modules: modeling, simulation and control. The intelligent decision system provides relevant reasoning about suitable algorithms and tools in respect of user's requirements and description of a process.*

*Keywords: multi-agent systems, intelligent control, discrete processes, knowledge management*

## 1. Introduction

A multi agent support system (MASS) for producing systems is presented in this paper. A production system control design includes three phases. First a model of production system has to be designed, then a control algorithm has to be suggested, and finally the proposed solution should be simulated and verified. This procedure

determains the basic architecture of the MASS. There are many problems connected to the system development. For start, knowledge has to be represented in such a form that is either human or computer readable. For second, all data and knowledge about specific domains has to be stored, archivated and organized for future reuse. Next, there is importance to create new knowledge on the base of stored and archivated information. The scope of the paper is to present a multi agent architecture to handle knowledge and provide decision support for discrete processes modeling, control and simulation. Description of the MASS is in Section 2. Section 3 describes agents' behavior, Section 4 deals with knowledge management, particularly with konwledge formalization, capturing and reusing. It is also adressed to reasoning, especialy to Case-Based Reasoning (CBR). Section 5 provides some conclusions and future work outline. At the end of the paper relevant references and resources for further information are listed.

## 1.1. Related works

Case-based reasoning systems have a wide domain of applications. There are some known applications that exploit CBR. [4] describes CBR in a Pellucid system, which is an IST project aiming to support organizationally mobile employees in their work. The purpose of CBR in Pellucid is to exploit historical experience and knowledge of former employees to assist new employees to adapt in a new working environment. In this system, case retrieval is based on a fuzzy classification as well, but calculation of the similarity degree requires a comparison of each attribute of the first situation with all attributes of the second one. The next system uses CBR for the same purpose is GMCR (Graph Model for Conflict Resolution methodology) presented by [9]. CBR is used to assist in identifying conflict situations and proposing their solutions, but case retrieval is built in a graph model. [6] deals with the problem of recommendation engineering that requires CBR, too. The similarity degree is calculated by a ratio between the numbers of the same attributes and the different ones. [1] present a fuzzy clustering method, which exploits a K-nearest neighbor search algorithm to extract a similar situation. In this paper the feedback is used to improve extraction accuracy. Further papers, which apply fuzzy sets to CBR, should be mentioned, namely [5], [10]. However, their work focuses more on introducing new operators to express more complicated relationships among situations.

A similarity is usually evaluated by the distance between two nodes in *n*-dimensional space. A number of CBR systems and their applications can be also found in [2].

## 2. Architecture of the system

The MASS consists of four modules. Each of them is created by one ore more agents. Fig 1 depicts a general architecture of the system modules.

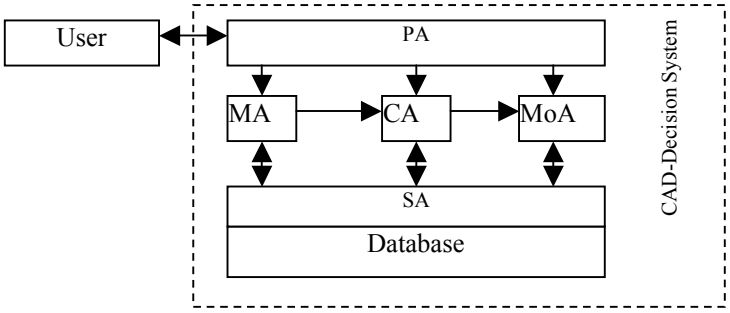


**Figure 1: General architecture of the agent decision support system**

The following agents in the system are considered:

**Personal Assistant agent (PAA)** – provides an intelligent interface between a designer (a user) and the other agents in the system. PAA receives designer's requirements and description of the process to be controlled as inputs for reasoning and decision making and returns suggested solution(s) for the user.

**Modelling Agent (MA)** – after receiving information about the process, search for apropriate modelling tools and algorithms on the basis of previous cases. It returns suggested algorithms and ask for more precise information accordig to a chosen algoritm. The final decision on which algorith and/or tool has to be chosen is up to the user. Finaly, Modelling agent returns a model of described production process.

**Control Agent (CA)**- receives a finaly chosen model with all necessary atributes defined. On the basis of the model, CA searches for an appropriate control algoritms in the database. Through PAA it negotiates with the user and finaly chooses appropriate control algoritms. The user has to choose an algorith from the suggested ones and specify all needed values for that. CA returns control algoritm for the process according user's specifications.

**Simulation Agent (SA)** – is responsible for simulation of control for the chosen model and control algorithm with the aim to help the designer to assess the proposed solution.

**Monitoring Agent (MoA)** - follows the system behavior after applying the recommended method for designing the control. If all the requirements are satisfied, then MoA updates the database by newly achieved results. That means the system stores all solutions for the next reuse and application. Otherwise, the MA and CA have to repeat their calculations.

All agents access the database through **Search Agent (SeA)**.

MA, CA, and SA utilize case based reasoning in order to find the best solution for the user.

Agents' communication is clarified in a sequence diagram, Fig. 2.

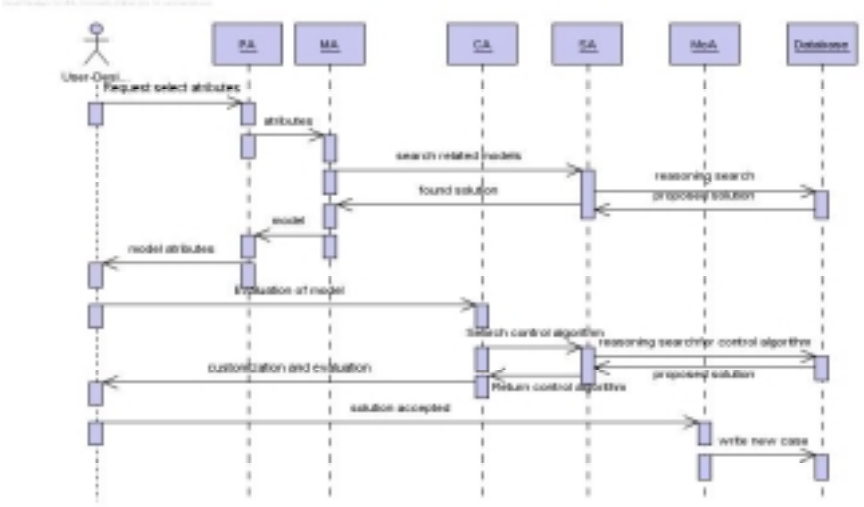


**Figure 2 A sequence diagram of the proposed support system.**

# 3. Agents' behavior in the MASS

Agents in a proposed system cooperate to find a solution to satisfy user's requirements. Solving a current problem means finding a similar case in case based library, finding solutions assigned to the found case and suggest that solution(s) to the user. The found solution has to be adapted to a current state, when the current state is not identical with the one in the database. After proposed solution is adapted to a new situation, it should be tested and when the proposed adapted solution shows to be good, it is entered to a database as a new case related to a new solution.

An algorithm for assisting a user in modeling, designing the control system and simulation is as follows:

*Initialization*: filling database with default data. There are two types of data in the database: description of cases – basic attributes, set of default solutions. There are relations among default cases and default solutions.

*Input*: User's requirements are given through PA. Users are supervised to fill in a questionnaire, where basic attributes for the current situation (description of production process) are described. After user's requirements are recorded, MASS begins search for appropriate modeling tool.

1. MA asks SeA to find a similar situation from database of cases. Requirements: search and data mining algorithms.
2. After finding similar case, related solutions – a method for modeling, are sent to the user. The returned solution(s) may not be the right solution for the current situation. It depends on degree of similarity of current and historical cases. The user has to pick one of proposed solutions and then MA adapts it to the current case, then proposes a solution (model with all necessary parameters) to the user, simultaneously sends it to CA. MoA monitor all these activities. After proposed solution is adapted and the user is satisfied with it, a new case is entered to database with relation to a new solution. Otherwise a new search is executed as long as a solution is found.
3. CA receives a (mathematical) model identified by MA and numerical parameters entered by the user. CA works in the same manner like MA. It asks SeA to find a similar model stored in database. On the basis of numerical parameters of the model, CA adapts the past solution to the current model and designs the control system.
4. MoA follows the system behavior after applying the designed control system. If all requirements are satisfied, MoA updates the database by newly achieved results. Otherwise a process is repeated as long as all requirements are satisfied.
5. SA receives mathematical model and design of control system. Its goal is to simulate proposed control system and verify it.

One agent can serve to many users and can access information in database only through SeA. Such approach is used to synchronize an access the database to avoid conflict situations (e.g., one agent performs database updating and at the same time other one extracts information from it). Similarly, all communication between the user

and the CAS-Decision System is performed by PA, which provides an appropriate interface to simplify the user work.

## 3.1. Two methods of cooperation among agents in the CBR MASS

Cooperation among CBR agents involves exploiting the set all cases in common memory of all agents and to reason about these cases using similarity-based reasoning. A problem is solved on the base of knowledge that can be learned by another agent within CBR MASS. Let's suppose that one agent $A_i$ has to solve a problem. It can ask another agent $A_j$ to find a similar case in database using its own method for evaluation of similarity degree, or it can send together with the case description also a method that might be used to retrieve similar case from database. A solution is assigned to a case on the basis of database structure. One or more solution can be assigned to one case.

In [8] two methods of cooperation in CBR systems are introduced. There are Distributed Case Base Reasoning (DisCBR) and Collective case based reasoning (ColCBR) described. The first methods means that agent delegate responsibility to solve a problem to another agent(s) within a CBR MASS. The Collective CBR means, that agents use common database and the same method to find appropriate solution of a problem. While DistCBE requires capability of remote evaluation of cases and solutions, ColCBR requires mobility of code, because methods of CBR have to be transported from one agent to another.

CBR system building requires cooperation of domain and IT experts while default cases and solution are entered to the system. Cases and solutions do not need to be encoded in classical rules. Each case typically contains a description of the problem, plus a solution and/or the outcome. Reasoning process and knowledge of experts about how to solve a problem is not recorded but it is implicitly given in solutions related to cases.

Case-based reasoning is often used for problem solving where a large amount of historical data exists, or experts can give a large volume of examples, or where a lot of valuable experience is recorded. Also CBR is preferred when there cannot be formulated rules for reasoning, or when there are a lot of exceptions from formulated rules. RBR are used when it is difficult to collect historical cases data.

CBR process includes a problem description, a new case specification, finding a similar case in database, finding related solutions, adapting proposed solution, testing adapted solution, confirmation of new solution, and learning new case with solution.

All case-based reasoning methods have some common steps:

- retrieve the most similar case (or cases) to a current case in the case library;
- reuse the retrieved case to try to solve the current problem;
- revise and adapt the proposed solution to a current case;
- record a new case with a new solution

Retrieving the most similar case is based on one of the similarity based reasoning methods. The similarity based reasoning involves formalized description of case in a case library. Two methods of similarity based reasoning are described in Section 4.5 in this paper.

## 3.2. Case library for CBR

A case library is a relational database, which involves the important features of each historical situation. Because the MASS has to work with different systems and kinds of information, the case library must be generic enough to represent all systems and their associated information. Determining which information is essential for storing is too difficult, since it is impossible to record all information and, on the other hand, because of practical reasons, the database should not be too large.

Other problems related to the database include the format used for data representation, e.g., number, text, graph, etc., and the method used for encoding/decoding these data. In the MASS, historical cases are represented by a number of basic attributes, which are indexed in a hierarchical scheme. These attributes are expressed by number or text (the example shown in Figure 3).

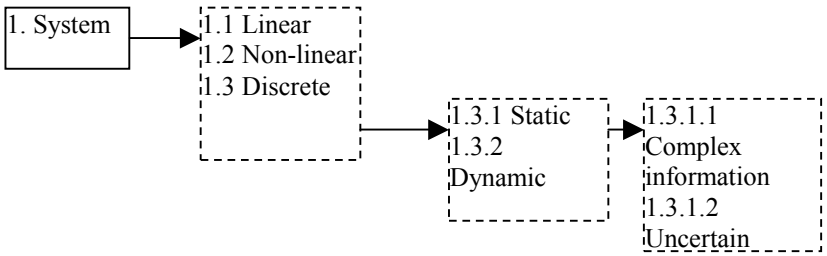


**Fig. 3: An example of data representation**

A case based library building for MASS is described in [3] The main database consists of two parts. The first part stores a description of all historical situations provided by the experts or added by users during customization of the MASS. A description of past situation consists of a number of basic attributes defined in advance. For example, situation ***si*** is a combination of a number of attributes as follows:

Let ***EL*** be a set of all possible attributes.

$$\boldsymbol{si} = \{ el_1 \cup el_2 \cup \ldots \cup el_n \} ; \forall i \neq j \in [1,n] ; el_i \neq el_j \in \boldsymbol{EL}.$$

In this representation, all the attributes are different each from other and they are sorted in a fixed order in order to facilitate search and retrieval.

The second part implies all solutions associated with the cases stored in the first part of the database.

# 4. Knowledge management for CBR MASS

## 4.1. Knowledge, information and data

The support system for control system designers is based on user requirements and general knowledge about process control design. Generally system handles data – a pure record of basic data about production processes (e.g. number of workstations); information – data connected with context of production process (discrete production process with number of workstation), and knowledge – interpretation of data (e.g. for discrete processes modeling use Petri Nets). The problem is how to organize data, information and knowledge that agents could easily access them, query and find appropriate answer, and interpret the answer to the user.

## 4.2. Knowledge formalization

Importance of knowledge formalization is obvious. Ontology presents the most useful tool for knowledge formalization.
Basic steps for knowledge representations are as follows:

- generate ontology
- define inferential operations on ontology
- identify query style
- add representations for operations
- add representations for basic claims

Ontology is an organized record of things and events among them.

## 4.3. Knowledge capture

There are several ways how to capture knowledge. First, the designers of the system in cooperation with experts from control system designers fill the system with initial knowledge about control system design. It is similar process to an expert system creation. For second, the system captures knowledge automatically, while users and agents provide any action. Such a way the system is evolving during its working process. Monitoring Agent from BG monitors each action in the system and records it. New knowledge is captured and stored in the system. The other way, how to capture knowledge is in cooperation with users. Users are asked to fill some notices and they are supervised to write as much details as possible about the notice. This is done through interactive user interface – Personal Agent. Also this knowledge is recorded and stored in the system.

## 4.4. Knowledge capitalization

The stored knowledge has to be organized and capitalized for future reuse. Support system provides advices on the basis of users requirements.

The most important aspects of historical situations are indexing and storing in a common database. New situations are described by using some basic attributes; afterwards, the similar, existing situations are identified and extracted from the knowledge base. Finally, the previous problem solutions are retrieved and the revised solutions are proposed for the current situation.

The first important requirement for the MASS is to build a database for storing information from numerous situations and their solutions. The second requirement is to define a set of rules for classification and extraction of data; and the last one is to design a method for decision making, so that the agents can identify what is best for the user in the current situation. CBR systems are learning from experience. New knowledge is created on the basis of historical solutions and their adaptations. In that sense knowledge is capitalized in the MASS.

## 4.5. Reasoning algorithms for extracting information

To find a similar case, the agents search by queries. The question is, how to achieve a case that is identical or very close to the current one described by the user. Two cases are identical if all their attributes are the same. Unfortunately the existence of the identical cases is rather rare. In many instances, the agents can find only a case that is "very close" to the target one. Exploiting the similarity degree and an inductive

reasoning method can help the agents distinguish situations and choose one of them that is closest to the new instance.

In this paper, two methods proposed to extract information from the database are presented. The first method is based on the similarity degree between two arbitrary situations; the second one is built on the inductive reasoning principle. These methods are also described in [3].

The first method **based on similarity degree,** works on the following principle: the agents calculate the similarity between two arbitrary situations by comparing all their attributes. Relationships among attributes are evaluated by any number (real or fuzzy) that reflects how much a solution of one situation is useful for the second one, with respect to these attributes. The similarity degree is defined by a combination of those partial relationships.

The second possible method is **an inductive reasoning** that works as follows: starting with the most important attribute, the agents sort and filter all cases that have the same or *to a certain degree* the same attribute like the target one. This cycle continues with less important attributes, until only one candidate remains. The last case remained is considered as the most similar to the target one.

The important requirement of the both methods is the identification of relationships among attributes. Due to a wide variety of attributes, classifying precisely the dependences among them is impossible. For that reason, the use of a fuzzy classification is preferred, because of its capabilities to express a number of difficult relations among attributes that other classification methods cannot do. For example, the following relation: "an algorithm for modeling system type A *could* be applied to system type B with *the same* or *very good* quality", could be easily expressed by exploiting fuzzy sets.

Let $re(el_1, el_2)$: $\{ EL \times EL \} \rightarrow [0,1]$ be a relation expressing the dependence between two attributes $(el_1, el_2) \in \boldsymbol{EL}$. There are some important properties of this relation.

- $re(el_1, el_2) \neq re(el_2, el_1)$ , resp. $\neq ( 1 - re(el_2, el_1))$, i.e. this relation is not symmetric or inverse.
- $re(el_1, el_2) = 1$; when the solution proposed for element $el_2$ could be applicable to element $el_1$ without changes. For example, $el_1$ is a linear system with complete information; $el_2$ is a linear system with parametrical uncertainty.
- $re(el_1, el_2) = 0$ ; when both the elements have disjoint domains of effects, i.e. the solution proposed for $el_1$ is useless for element $el_2$. For example, $el_1$ is a discrete event system and $el_2$ is a linear system.
- $\forall\, el \in \boldsymbol{EL}$; $re(\varnothing, el) = 1$ a $re( el, \varnothing ) \geq 0$

The similarity degree between two arbitrary situations *Sim(,)* is defined as follows:

Let us consider two situations $si_1$ and $si_2$ with the same number of attributes.

$si_1=\{el_{1,1}, el_{1,2},\ldots,el_{1,n}\}$ and $si_2=\{el_{2,1}, el_{2,2},\ldots, el_{2,n}\}$

If the numbers of attributes of each situation are not equal, we can add an empty attribute ∅ to them. Let us define matrix $W \in [0,1]^{nxn}$ as follows:

$$w_{i,j} = re(el_{1,i}, el_{2,j}) \quad (1)$$

Then,

$$Sim(si_1, si_2) = W^T Q W \quad (2)$$

where $Q=\{q_{i,j}\}^{nxn}$ is a symmetric and positively definite matrix of the same dimension ($n$ x $n$) as W. Each member $q_{i,j}$ of matrix Q expresses a weight of how much a relation $re(el_{1,i}, el_{2,j})$ can influence the dependence between both situations. Given a target case $si_{current}$ the case that maximizes a function $Sim(si, si_{current})$ is considered as the most similar one, and its solution could be applied or adapted to solve the current situation.

Inductive reasoning method extracts the desired situation by performing a search of a decision tree, which involves all possible historical cases satisfying certain conditions. A decision tree is generated as follows: starting with the most important attribute – propose it is $el_1$, all cases that satisfy the following condition are added to the decision tree.

$$re(el_1|_{(si)}, el_1|_{(si_{current})}) \geq \alpha \quad (3)$$

where coefficient α is a low bound that is used to restrict a set of candidate situations. The classification process continues with less important attributes, until only one candidate solution remains.

**Conclusions**

The paper introduces a multi-agent architecture of a support system for production systems control design. The detailed description of intelligent agents and their communication within the system is provided.

The MASS is related to a lot of problems; some of them have already been mentioned in this paper. In the future, the main interest is to focus on cooperation among agents, because there are many different kinds of agents situated in that system, developed by different teams, with different architecture, behavior, and knowledge. The problem that has to be solved in the nearest future is to find how suggested solutions can be adapt to user's requirements.

**Acknowledgement**
The system has been developing in cooperation with FEI STU Bratislava, FEI TU Košice, and MEF STU Bratislava, under project APVT-51 011602.

**References**

[1] Bhanu B and Dong A. (2002): Concepts learning with fuzzy clustering and relevant feedback. *Engineering Application of AI*, No. 15, 123-138.

[2] Bergmann R. (1999): Special issue on case-based reasoning. *Engineering Application of AI*, No. 12, 661-759.

[3] Dang T.T., Frankovič, Budinská: Case-based reasoning applied for CAS-decision system, In Proc. Of 2nd IFAC Conference: Control Systems Design, 2003, Bratislava, on CD

[4] Dang T.-Tung, Hluchý L., Budinská I., Nguyen T. G., Laclavík M. Balogh Z. (2003): Knowledge management and data classification in Pellucid. *In Intelligent Information Processing and Web Mining*, Advances in Soft Computing, Springer-Verlag, 563-568.

[5] Dubois D., Esteva F., Garcia P., Godo L., M`antaras R. L., and Prade H. (1998): Fuzzy set modelling in case-based reasoning, *International Journal of Intelligent Systems*, No. 13, 345–373.

[6] McSherry D. (2002): Recommendation Engineering. *ECAI-02*, 86-90.

[7] Mille: Proceedings of the Workshop: Adaptation in Case Based Resoning, A workshop at ECAI 1996, Budapest

[8] Prasad M.V.N, Plaza E.: Corporate Memories as Distributed Case libraries; http://ksi.cpsc.ucalgary.ca/KAW/KAW96/prasad/cm.html

[9] Ross S., Fang L. and Hipel K. (2002): A case-based reasoning system for conflict resolution: design and implementation. *Engineering Application of AI*, No. 15, 369-383.

[10] Rudas J. (1999): Evolutionary operators; new parametric type operator families. *Int. Journal of Fuzzy Systems,* **Vol. 23, No. 2**, 147-166

[11] Sycara K.: Using CBR for Plan Adaptation and Repair; In Proc. of the DARPA CBR workshop, 1988, http://online.loyno.edu/cisa494/papers/Sycara.html

[12] Zeng D., Sycara K.: Using Case Based Reasoning as a Reinforcement Learning Framework for Optimization with Changing Criteria; Proc. of. Seventh International Conference on Tools with Artificial Intelligence; Virginia USA, 1995; http://csdl.computer.org/comp/proceedings/tai/1995/7312/00/73120056.pdf