Rule-based tool in attributive logic for system simulation

Andrzej Macioł[1]

Abstract: This paper presents rule-based tool for system modeling and simulation. Our idea utilizes Attributive Logic applied to rule-based systems and very similar to the ontology used in relational databases. Thanks to them it is possible to handle relational databases tools (SQL) to store and edit knowledge rules and also to solve the basic selection problems in the rule-based system. Moreover it is possible to integrate an expert system like inference engine with data included in the information system model. The present application, necessary for the deployment of the method presented is being tested in MS SQL Server environment. The results of provisional tests show that using the Inference with Queries (IwQ) idea to construct simulation models enables to construct universal and flexible tools.

Keywords: Simulation, Rule-based systems, Attributive logic, Structured query language.

1. Introduction

Computer simulation is widely used in analyzing and mastering complex systems. It refers both to manufacturing systems and business process management or CASE tools. Earlier, specific methods of modeling and simulation were created in each of these areas. At present more and more universal solutions, which enable mutual solving complex problems are being sought. Examples of these problems are manufacturing plan optimization, job shop scheduling and others, which are optimization problems characteristic for operational rsearch, but, at the same time they must be tested in connection with mastering workflows, business process management and IT.

Methods which are currently used in simulation vary in modelling real systems and possibilities of carrying out experiments. Among the BPM tools two important standardization efforts include the development of the Business Process Modeling Notation (BPMN) and the Unified Modeling Language (UML) version 2.0 (Hall & Harmon, 2006). The BPMN was created by the Business Process Management Initiative (BPMI) organization (White, 2004). BPMN is designed to facilitate the graphical representation of business processes. It is important because it provides a standard way of describing business processes and it can be used to generate BPEL, an XML-based business process execution language. BPMN comes in two versions, a simple version that business managers can use and a more sophisticated version that provides all the details needed to generate software code. Initial versions of

[1] Wydział Zarządzania, Akademia Górniczo-Hutnicza, Gramatyka 10, 30-067 Kraków
e-mail: amaciol@zarz.agh.edu.pl

the Object Management Group's (OMG) -UML (UML 1.0, 1.1) standard were primarily used by software developers for process automation efforts. However, UML was never very popular with business users. The latest release of UML, version 2.0, represents a major redesign of the language and includes a much improved Activity Diagram notation which has generated a lot of interest among organizations that use UML for software development. In essence, as with the BPMN, if business modelers use UML Activity Diagrams they can be passed to IT developers who can then use those diagrams as the starting point for software development. UML Activity Diagrams and basic BPMN diagrams are very similar, and it is possible they will be combined or linked in the near future. No matter what happens, the similarity between BPMN and UML Activity Diagrams suggests that in the near future business modeling tools will be shifting away from their various proprietary notations to one of the standard, public notations.

Extended BPMN models and UML Activity Diagrams models enable to record sufficient amount of information necessary for simulating system behavior. Most modeling tools give the possibilities of Discrete Event simulation. Discrete Event allows users to introduce a higher level of precision into the simulation process because it provides the ability to simulate the model of a business process as it changes with time, with the passage of time tracked as a series of discrete events rather than as a continuous transformation. Unfortunately, in most cases the type of modeling does not allow precise depicting of complex inference rules, only describes general statistics concerning past or possible future events (e.g. number of actual events, flows in the time unit or percentage distribution of their attributive value).

Apart from the methods mentioned above, which are currently the most widespread, still the following business process modeling conventions are used: data flow diagrams, system flowcharts, resource-event-agent diagrams, IDEF0/IDEF3, event process chains (Carnaghan, 2006). These conventions allow the construction of simulation models as well.

Simulation tools for manufacturing systems modeling were created using a slightly different source (Eldabi & Paul, 2001). The first works on that subject used high-level programming languages, such as FORTRAN and Pascal, or general-purpose simulation languages, such as GASP, GPSS, SIMSCRIPT, SLAM, and SIMULA. However, many simulation software tools have become commercially available, and these require little or no programming effort and experience to use. Examples of these tools include SIMFACTORY II.5, ProModel, AutoMod II, WITNESS, SIMPROCESS, Simul8 and others. Contrary to business processes modeling methods it is difficult to speak about model notation standards in simulation software tools for manufacturing systems. Each tool gives its user specific GUI, which enable to define processes, activities, dataflows, events easily. For example, SIMPROCESS utilizes an activity-based modeling paradigm, in which real world behavior of activities such as copying, assembling, transformation, batching, and branching are built into the tool. SIMPROCESS features a suite of pre-built "Activities blocks" which are used for assembling logic-based business models and simulations. These activities can be connected or embedded into processes by

using simple flowcharting techniques, thus making process documentation fairly straightforward. Users can also customize pre-built activity blocks to represent the operational characteristics of their own business processes. Thanks to these, tools such as SIMPROCESS give much more possibilities of quick modeling complex processes and activities, not to mention complex system logic, than BPM tools.

Modern simulation tools are characterized by better integration possibilities. The possibility of converting models to different notation conventions has become a standard. Also, on-line access to other tolls, e.g. SQL databases is possible. Modern simulation tools give vast possibilities of business process decision modeling, but differ from specialized tools for business rules modeling (BRM) in this area. Gensym's G2 software is one of the exceptions (Barnett, 2003). Its concept is based on simultaneous application of BPM and BRM tools to construct simulation models. Also, new concepts combining the advantages of business and simulation modeling into a single framework have appeared (Gregoriades & Karakostasb, 2004). Integrating business objects with simulation objects enable the elimination of duplication of business process modeling for simulation and enterprise operation. As a result, information consumed or produced during enterprise operation can be directly communicated to the simulation model. Unfortunately, there is no information about the practical implementation of this concept.

Irrespective of modeling rules simulation tools for manufacturing system and business process modeling must fulfill the following requirements:

- Their integration with other modeling and information systems tools is necessary (eg. BPEL, WSDL etc.),
- They must enable to record procedural and declarative knowledge (characteristic for expert systems)
- They must faithfully reflect the real environment where manufacturing systems an others business systems operate, eg. via on-line cooperation with SQL databases.

Rule-based tool in attributive logic for system modeling and simulation, which we have worked out, can be such a solution.

If we want to construct a simulation model describing deal and complex system decisions, then we must find knowledge representation method. The problem solved with simulation methods belongs usually to the class of unstructured problems. The knowledge about such problems cannot be represented in procedural manner. As a result, only those methods for knowledge representation can by used, which allow building declarative model of decisions. One of these methods is a frame-based approach to knowledge formulation (Minsky, 1977). Our solution is very near to this idea but instead of specific relations between atomic data implemented by frames and hierarchy facets ("A-kind-of"), as well procedural facets (e.g. "If-needed", "If-created",...) we use mechanisms of relational database (relationships, triggers, stored procedures etc). At first, this tool was made for business rules modeling, however, the specifics of inference rules modeling enables to adjust the tool to system simulation needs easily.

2. Modeling of business rules in expert system like inference engine for simulation model

Many researchers have suggested approaches or ideas to integrate AI and databases. From the one side there are works concerned on intelligent databases as Datalog (Bertino et al., 2001). From the other hand, there are investigations to couple expert systems with relational or object databases (Sonar, 1999). Our solution is based on using of SQL mechanisms with data stored in relational database as a knowledge base. Our solution joins the concept of frame-based knowledge representation, replaced partially with relational database model, with inference possibilities given by procedural languages (Transact-SQL in current version of the solution) and is near to rule-based languages idea (Liu, 1998).

Our idea utilizes Attributive Logic applied to rule-based systems and very similar to the ontology used in relational databases (Macioł, 2007). Thanks to them it is possible to handle relational databases tools (SQL) to store and edit knowledge rules and also to solve the basic selection problems in the rule-based system. Combining possibilities of SQL with a concept of the rule-based system allows to build a solution according to Variable Atomic Attributive Logic (VAAL), i.e. attributive logic with atomic values of attributes incorporating variables (Ligęza, 2006). Moreover it is possible to integrate an expert system like inference engine with data included in the information system model.

Let us take the following set of symbols:

- O - a set of object name symbols,
- A - set of attribute names,
- V - set of variables.

The atomic formula of VAAL can be specified as follows:

$$A_i(o)\ r\ X \tag{1}$$

where $X \in V$, $o \in O$ and r is a relational symbol, i.e. $=$, $>$, $<$, etc. An example of such a formula is the following statement:

Repair team 1 is occupied
this can be understood as follows: attribute: $ObjectIsOccupied$ of $RepairTeam1$ equals $true$.

where $ObjectIsOccupied$ is an attribute A_i, $RepairTeam1$ is an object o and $true$ is unknown by specific value (variable).

Our model of the rule-based system use an extended form of the rules including both control statement and dynamic operations. Generic form of a rule can be presented as follows:

$$rule(i) : (A_1\ r\ d_1) \wedge (A_2\ r\ d_2) \wedge ... \wedge (A_n\ r\ d_n)$$
$$\rightarrow$$
$$set(B_1 = b_1, B_2 = b_2, ..., B_b = b_b)$$

$H_1 = h_1, H_2 = h_2, ..., H_h = h_h$
$next(j)$
$else$
$set(C_1 = c_1, C_2 = c_2, ..., C_b = c_b)$
$G_1 = g_1, G_2 = g2, ..., G_h = g_h$
$else(k)$

where $(A_1\ r\ d_1) \wedge (A_2\ r\ d_2) \wedge ... \wedge (A_n\ r\ d_n)$ is the regular precondition formula, $B_1 = b_1, B_2 = b_2, ..., B_b = b_b$ is the specification of the facts to be changed in knowledge base after successful execution of the rule, $H_1 = h_1, H_2 = h_2, ..., H_h = h_h$ is the specification of conclusions forming a direct output of the rule (e.g. decisions or queries to be displayed on the terminal or control actions to be executed) in case when it is successful executed, $C_1 = c_1, C_2 = c_2, ..., C_b = c_b$ is the specification of the facts to be changed in knowledge base in case of failure, $G_1 = g_1, G_2 = g2, ..., G_h = g_h$ is the specification of conclusions in case of failure and $next(j)$, $else(k)$ are the specifications of control; the next(j) part specifies which rule should be examined immediately after successful execution of rule i and $else(k)$ part specifies which rule should be tried in case of failure.

In our model both objects, attribute Ai(o) and variable X, can be selected from any relational structure of RDBS (relation or join of relations). Our manner of relational data specifying is an extension of selection operation in SQL (in SQL it is not possible to specify the index of row in results collection). In our model we use one object class to store all the necessary data. This object is called facet - F. The "facet" concept in our system corresponds to "value facet" concept in frame-based knowledge representation. The facet can have a value taken from a relational structure and can be used as object attribute value or variable value. The facets can be used as arguments in operations on the right hand of the rule. Thanks to the utilization of extended selection formula, knowledge definition process gets simplified. SQL queries realize a significant part of inference. The combination of rule-based system in attributive logic with reasoning by queries is called Inference with Queries (IwQ).

The example of modeling rules has been taken from our research on applying simulation to optimization of maintaining process system in the complex production structure. Below we present knowledge record of the possibility of doing a repairing task by a repair team. Suppose we have two teams groups: the repair team, which can remove the fault and the protecting team, which can diminish breakdown results and make the work of other aggregates in the process line, but which cannot repair the fault. Suppose also that when the repair team is free within an hour, we will not call the protecting team. The inference diagram about the possibilities of repairing or protecting the breakdown is shown in Fig. 1.

To solve this problem we have to build following relation in the database:

$RTeams = \{TeamId, TeamType, Engaged, ReadyTime\}$

with following example rows:

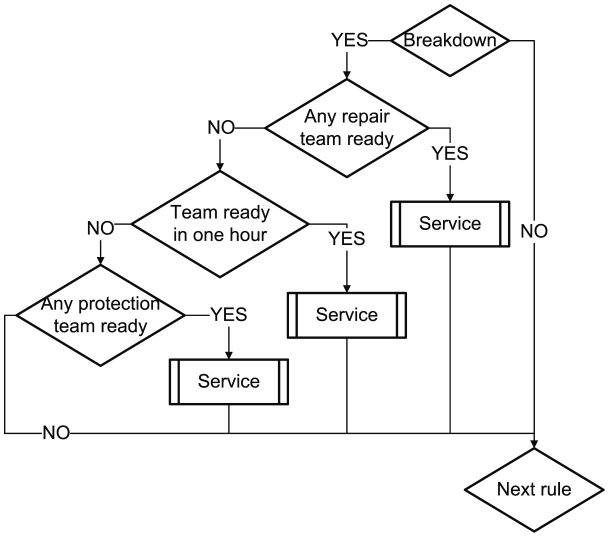

Figure 1. Inference schema

TeamId	TeamType	Engaged	ReadyTime
1	repair	No	0
2	repair	True	98
3	protective	True	127

To record the fragment of the knowledge in IwQ model it is necessary to define the following facets:

constants
$F_1\{$"$Repair$", "$repair$"$\}$,
$F_2\{$"$Protection$", "$protection$"$\}$,
$F_3\{$"$Hour$", "60"$\}$,

facts
$F_4\{$"$Time$"$, v\}$,
$F_5\{$"$TimePlusHour$"$, v\}$,

select facets
$F_6\{$"$ReadyTeam$"$, TeamId, \phi_1, \phi_2\}$,
$F_7\{$"$ReadyTeamSoon$"$, TeamId, \phi_1, \phi_2, \phi_3\}$,
where $\phi_1 =' Engaged <> @True'$,
$\phi_2 =' TeamType = @TeamType'$,
$\phi_3 =' ReadyTime < @TimePlusHour'$.

The inference engine uses also such constants as False, Null, One, True, Zero.

The rest of the knowledge is recorded by the following rules:

$rule(10) : IsBreakDown.v = True.v$

$\rightarrow$
$set(TeamType.v = Repair.v)$
$next(15)$
$else$
$else(30)$

$rule(15) : ReadyTeam.c = Null.v$
$\rightarrow$
$set(TimePlusHour.v = Add(Time.v, Hour.v)$
$next(20)$
$else$
$H_1 = ServiceTimeGenerate(ReadyTeam.v)$
$else(30)$

$rule(20) : ReadyTeamSoon.c = Null.v$
$\rightarrow$
$set(TeamType.v = Protection.v)$
$next(25)$
$else$
$H_1 = ServiceTimeGenerate(ReadyTeamSoon.v)$
$else(30)$

$rule(25) : ReadyTeam.c = Null.v$
$\rightarrow$
$next(30)$
$else$
$H_1 = ServiceTimeGenerate(ReadyTeam.v)$
$else(30)$

Stored procedure *ServiceTimeGenerate* generates the time of service and updates *ReadyTeam* table.

Inference begins with firing rule 10. In case of a breakdown which is show by the fact that the IsBreakDown facet assumes true, the system goes to rule 15 and the facet is converted into SQL query. In our case there is the following query:

$SELECT\ TeamId\ AS\ ReadyTeam.v$
$FROM\ ReadyTeam$
$WHERE(Engaged <> @True)$
$AND\ (TeamType = @TeamType)$

After execution of the query collection index is set to 0 if one or more records meets the condition or to *null* when the output set is empty. The value of collection index is introducing into the rules left hand side together with "encoding" constant $Null$. If the facet is null, it means that there is no free repair team available at present

and rule 20 is fired. Otherwise records in facets are updated and this inference fragment is finished. Before verifying rule 20 the facet is converted into SQL query. In our case the query goes as follows:

$$
\begin{aligned}
&SELECT\ TeamId\ AS\ ReadyTeamSoon.v\\
&FROM\ ReadyTeam\\
&WHERE(Engaged <> @True)\\
&AND\ (TeamType = @TeamType)\\
&AND\ (ReadyTime < @TimePlusHour)
\end{aligned}
$$

If the collection number is 0, it means that none of the teams will finish their work within an hour and rule 25 is fired. Otherwise, the system ends this inference fragment assuming that the task will be served during the next simulation step. Next rule (25) checks the possibilities of using the protecting team and, if such a team is available, similar activities as for rule 15 are performed, but referring to the protecting team. Otherwise, no new tasks are undertaken, because the system is waiting until the teams will be released during next steps. The example described above is only a fragment of the knowledge describing the simulation model. However, it can easily be seen that using similar facets and rules not only firing and task serving rules but also system dynamics can be described.

3. Deployment of the method

The present application, necessary for the deployment of the method presented is being tested in MS SQL Server environment. Text interface to build models and do experiments have been elaborated. Results and input data are introduced via MS SQL Server to and from respective relations. Graphic interface, which will accelerate model construction and facilitate its verification, is being worked on. The Entity-Relationship diagram of relations depicting the Knowledge Base of the simulation tool is represented in Fig. 2. The full model consists of two parts describing facets and rules.

In the case of the facets representing SQL queries a set of conditions is specified. The screen for editing such facets is shown in Fig. 3. For each rule Left Hand Part and Right Hand Part are described. An example of editing a complex rule is shown in Fig. 4. On the right hand part the following tasks can be done:

- change of facet value (Set),
- opening the forms for input, selecting or displaying of specified by facets name variable, designed in procedural language,
- stored procedures designed by the user execution,
- starting user's methods recorded in dynamic link libraries.

The structure of each model is always equal, irrespective of the user's needs and consists of the elements mentioned above. It facilitates model conversion into any XML-based form.

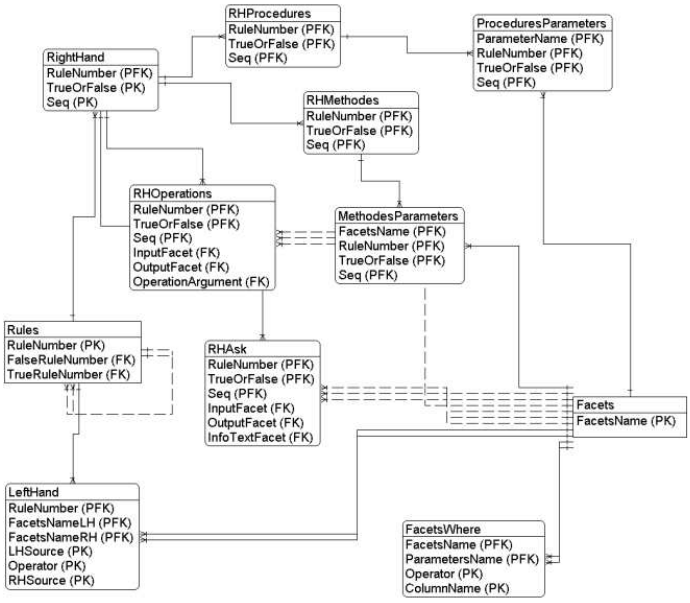

Figure 2. Entity-Relationship diagram of Knowledge Base

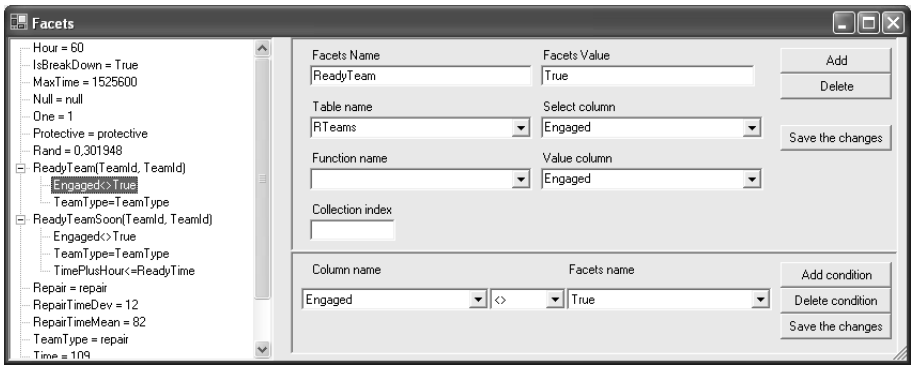

Figure 3. The screen for editing facets

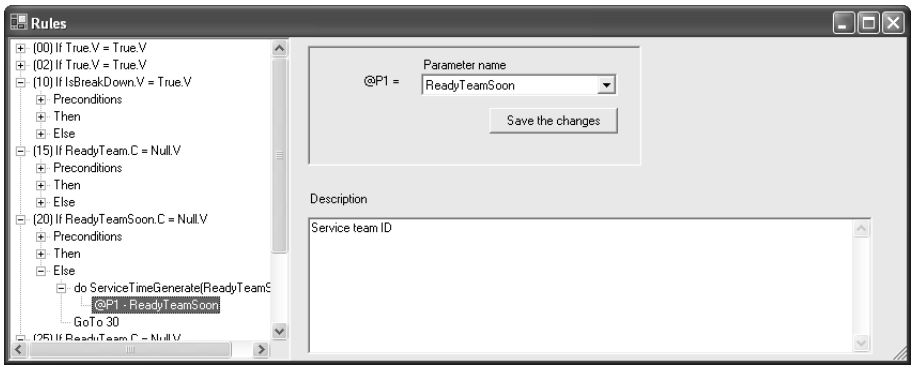

Figure 4. An example of editing a complex rule

4. Conclusions

The results of provisional tests show that using the Inference with Queries (IwQ) idea to construct simulation models enables to construct universal and flexible tools. The most important effect of implementing IwQ method in simulation is simplifying of the modelling process and full integration of the business processes model with information environment of the modelled enterprise. The advantages of this solutions are:

- combination of BPMN models simplicity, which was achieved thanks to maximum limiting the number of typical activities, with the possibilities of describing complex business rules in the manner characteristic for expert systems,
- simple logic structure of the model thanks to using facets as SQL queries,
- full representation of real environment, where most business processes take place, thanks to integration with SQL databases,
- easy on-line cooperation with ERP systems and user's own applications thanks to information exchange in SQL databases standard and easy use of user's own libraries,
- easy model conversion to any XML-based form, possible thanks to using relational schema for representation of the system and its behaviour.

The area where our tool can be used is wide. The first tests dealt with evaluating the possibilities of using the tool to solve rather simple problems connected with organizing business processes in manufacturing, such as queueing problems, repair teams organization, selecting logistic parameters (storage capacity, transport means, etc.). But the main goal of our tests is solving complex and multi-aspect problems of organizing business processes in manufacturing. We are mostly interested in verifying the correctness and effectiveness of different manufacturing scheduling techniques under uncertain and incomplete information conditions.

Our current tests show that the tool can be successfully used in solving complex problems connected with Business Process Reengineering and Improvement.

Further tests connected with the tool itself will now concentrate on improving and expanding its functionality with simultaneous preserving general assumptions about its operating rules (among other things, by introducing graphic interface). In the future, we think about its alternative use as the knowledge "container" of languages used for constructing ontology (e.g. RDF) with simultaneous substituting SQL mechanisms with other predicate inference engines (e.g. f-logic).

References

Barnett, M. W. (2003) Modeling and simulation in business process management. *Gensym Corporation, Available from* `www.gensym.com`. *Accessed April 20, 2007.*

Bertino, E., Catania, B., Zarri, G.P. (2001) *Intelligent Database Systems: A Synopsis.* Addison Wesley, New York.

Carnaghan, C. (2006) Business process modeling approaches in the context of process level audit risk assessment: An analysis and comparison. *International Journal of Accounting Information Systems* **7**, 2, 170-204.

Eldabi, T., Paul, R. J. (2001) Evaluation of Tools for Modeling Manufacturing Systems Design with Multiple Levels of Detail. *International Journal of Flexible Manufacturing Systems* **13**, 163-176.

Gregoriades, A., Karakostasb, B. (2004) Unifying business objects and system dynamics as a paradigm for developing decision support systems. *Decision Support Systems* **37**, 2, 307-311.

Hall, C., Harmon, P. (2006) The 2006 Enterprise Architecture, Process Modeling and Simulation Tools Report. *Available from* `http://www.bptrends.com` *Accessed April 20, 2007.*

Ligęza, A. (2006) *Logical Foundations for Rule-Based Systems.* Springer-Verlag, Berlin, Heidelberg.

Liu, M. (1998) An Overview of Rule-based Object Language. *Journal of Intelligent Information Systems* **10**, 5-29.

Macioł, A. (2007) An application of rule-based tool in attributive logic for business rules modeling. *Expert Systems with Applications*, doi:10.1016/j.eswa.2007.02.003.

Minsky, M. (1977) *A framework for representing knowledge, in: P. Winston (Ed.), The Psychology of Computer Vision.* McGrow-Hill, New York.

Sonar, R. M. (1999) Integrating Intelligent Systems using an SQL-database. *Expert Systems with Applications* **17**, 45-49.

White, S. A. (2004) Introduction to BPMN.
Available from `http://www.bpmn.org` *Accessed April 20, 2007.*

Regułowe narzędzie w logice atrybutywnej do symulacji systemów

W artykule przedstawiono regułowe narzędzie do symulacji systemów. Prezentowane rozwiązanie polega na wykorzystaniu logiki atrybutywnej zbliżonej do ontologii stosowanych w relacyjnych bazach danych. Dzięki temu możliwe jest jednoczesne wykorzystanie narzędzi relacyjnych baz danych (SQL) do zapisu reguł oraz rozwiązywania prostych problemów selekcji. Możliwa jest także integracja systemu wnioskującego z danymi opisującymi model systemu informacyjnego. Obecnie aplikacja niezbędna do realizacji przedstawionej metody znajduje się w fazie testowania w środowisku MS SQL Server. Wyniki wstępnych badań potwierdzają, że przyjęcie koncepcji Inference with Queries (IwQ) do budowy modeli symulacyjnych pozwala na zbudowanie uniwersalnych i elastycznych narzędzi.