An Approach to the Knowledge-based Data Handling in Complex Process Control Systems

R. Gamzayev, D. Kuklenko, M. Tkachuk
Computed-Aided Management Systems Department,
National Technical University "Kharkiv Polytechnic Institute",
Frunze str., 21,
Kharkiv, Ukraine
gamzaev@mail.ru, dmytrokuklenko@yandex.ru, tka@kpi.kharkov.ua

Abstract. This paper presents an approach for knowledge-based data handling in complex Process Control Systems (PCS). Special attention is paid to the specific situations taking place in the technical processes, which run under control of such PCSs. To describe those specific situations we use the concept of Active Rule (AcR). The considered PCSs operate in the real-time mode. Because of this, handling a big number of active rules may cause violations of some real-time constraints. We propose an approach of Association Rules (AsR) in order to eliminate those problems. The collection of AsR is mined from the data describing the specific situations of processes. Those data are collected in PCS as a result of executing appropriate AcR. We elaborate a collection of modification patterns for simplifying the structure of existing AcR, and the general scheme for the interaction between AcR and AsR. The results presented in the paper are based on real-life projects that we performed in the domain of Web-based PCSs for Ukrainian gas-and-oil production enterprises.

1. Introduction

The quick rise of computer and information technologies has set the conditions for such progress in the field of PCS. These systems are now supposed to solve not just traditional tasks of Supervisory Process Control and Data Acquisition (SCADA), but rather the tasks in the decision support domain. Therefore, the problem of intelligent data processing may take place in such systems.
Our current research is closely connected with the development of PCS for the gas-and-oil production enterprises situated in the region of Kharkiv (Eastern Ukraine). We presented our practical experience in several scientific articles. In particular, in [Tk02] we presented the reference architecture of multilevel distributed PCS. [Tk01] introduced our knowledge-based maintenance environment for such systems. We discussed an approach to data visualization in Web-based SCADA systems in [Ku04]. Given paper presents an approach to advanced data processing in multi-level PCS. We call this approach the Intelligent Data Engineering (IDE).
This paper is structured as follows: Section 2 presents the approach of *Intelligent Data Engineering* used for advanced data processing in multi-level PCS. In Section 3 we describe a framework for 1) extracting associations from technical process data, and 2) using these associations for simplifying the structure of specific situations' declarations. The formal models of modification patterns that specify possible Active Rules simplifications are

presented in Section 4. We evaluate our approach in Section 5 and provide some experimental results. A brief overview of related works is presented in Section 6, and finally we conclude the paper and discuss future works in Section 7.

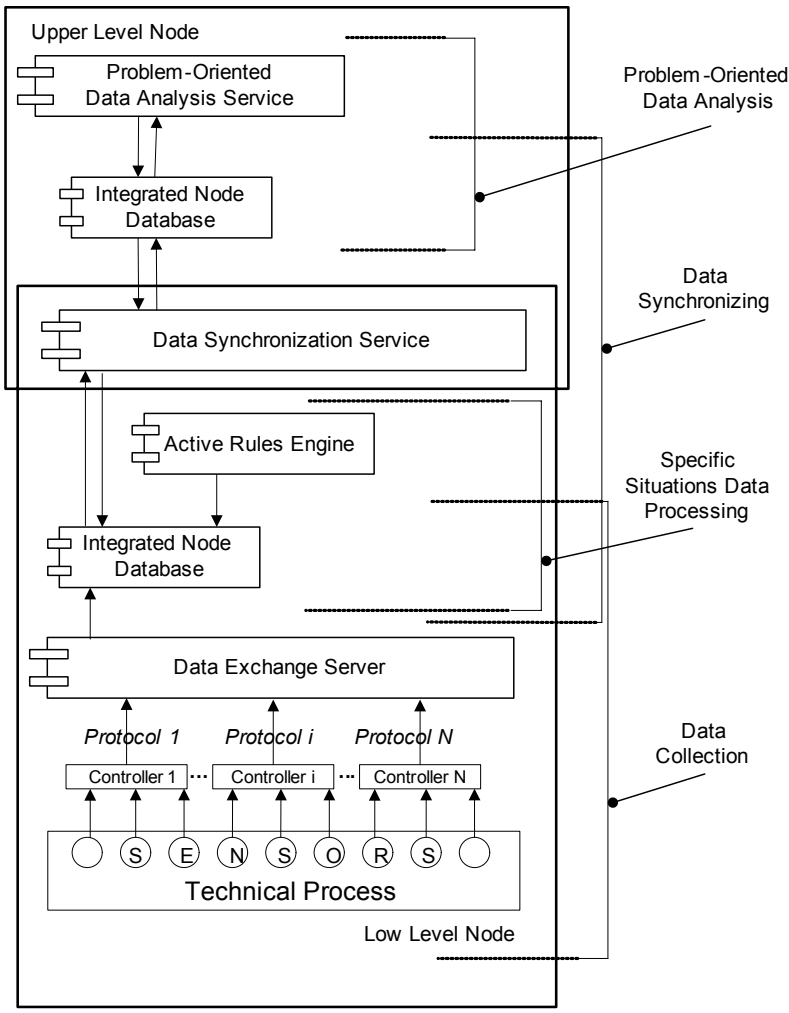

Fig. 1 – Architecture of IDE processes and components

2. Intelligent Data Engineering as an Approach to Advanced Data Processing in the Distributed Multi-level PCS

We define *Intelligent Data Engineering* (IDE) as a knowledge-based information framework that provides the information support for PCS staff. We propose the following architecture of the IDE processes and components (see Fig. 1).
The IDE framework consists of the following processes:

1) *Data Collection*. The main goal of this process is to solve the data storage problem. By 'data storage problem' we understand saving the parameters values in database as well as performing their management (calculating the average parameter values, etc.). The Data Exchange Server (DES) provides parameter values to Integrated Node Database (INDB) with accordance to data exchanging protocols stored in its protocol scheme. These data are stored in the retrospective scheme and processed with respect to time granularities (seconds, minutes, hours, days).
2) *Specific Situations Data Processing*. The goal of this process is to detect the specific situations. The concept of Active Rule (AcR)[Pa94] is used in this process. It allows specifying the structure of and the reaction to specific situations. We realize this idea with the help of Active Rules Engine (ARE).
3) *Data Synchronization*. This process performs the data exchange between different nodes, and Data Synchronization Service (DSS) is the component that realizes this function.
4) *Problem-Oriented Data Analysis* (PODA). This process consists in Association Rules (AsR) mining. Basing on the mined AsR, we can perform the simplification of AcR structure. The Problem-Oriented Data Analysis (PODA) implements these tasks in a way shown in the Section 3.

In our previous works we discussed models, functionality and appropriated information technologies that realize several IDE processes. In particular, we described Data Collection process in [Ku03b], Specific Situation Data Processing - in [Ku03a], and Data Synchronization process - in [Ku04] respectively. The given paper finalizes the presentation of IDE processes by describing the models and procedures that are used in Problem-Oriented Data Analysis process.

3. Models and Framework Proposals for Problem-Oriented Data Analysis Process

In this section we briefly consider the enhanced model of AcR that is elaborated in the IDE framework. Then we introduce the structures of AsR we are seeking when analyzing the stored technical process data. And finally we provide the common scheme of PODA process.

As it is well-known from the theory of active databases (see, e.g. [Pa94]), the classical AcR has the following structure:

$$<Rule>=<Event><Condition><Action> \qquad (1)$$

It can be interpreted in the following way: database system checks when some <Event> (calendar or external with respect to database) occurs, and then if the <Condition> indicated in (1) holds true, the Rule <Action> is being executed. The <Condition> represents usually some predicate on the data inside the database. The <Action> of an AcR can be specified as a SQL-script or some another executable pieces of code belonging to an PCS business logic

The model of classical AcR R can be also presented in the following way: $R \subseteq E \times C \times A$ (here E is a set of possible events, C is a set of possible conditions, and A is a set of possible actions). The enhanced AcR declaring the specifics situations and system reaction has the following model:

$$R' \subset E \times C' \times A' \times DA_1 \times DA_2 \qquad (2)$$

As an event $e \in E$ we consider receiving a new data package by the INDB from DES. The specific situation is defined by the values of technical process parameters, therefore a condition $c' \in C'$ contains the predicate that operates these values. In the normal mode the values of technical parameters are stored in the system for a limited time. But when specific situation occurs, it is important to guarantee the values to be retained. Those values can help analyzing the state of technical process during the progress of specific situation and the pre-history and the post-history thereof. We call those ranges zones of specific situation (see Fig. 2). Thus, an action $a' \in A'$ contains the identifiers of parameters, which values should be stored as a description of the specific situation. DA_1 and DA_2 are correspondingly the sets of pre-history and post-history possible length respectively (so, $\underline{t} \in DA_1, \overline{t} \in DA_2$).

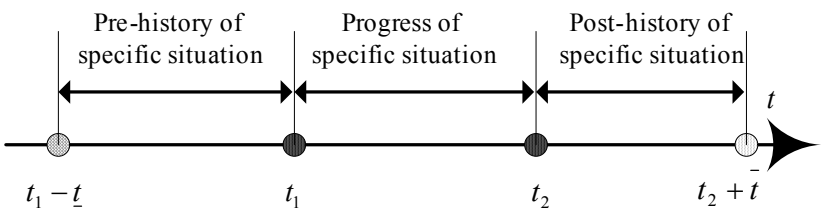

Fig. 2 – The specific situation zones

Further we will use the term *AcR* only for enhanced AcR used in our IDE framework.

The condition of such an AcR is composed of the following three types of elementary conditions. These types are:

1) Numerical parameter value p_i is compared with the constant c, or it is checked if this value belongs to the range $[\underline{c}, \overline{c}]$.
2) The values of two numerical parameters p_1, p_2 are compared.
3) The value of Boolean parameter is checked.

We define S as a set of all possible elementary conditions. Then, the condition of an AcR can be presented as $\widetilde{\Phi}(S, \{\wedge, \vee, \neg\})$. Here $\widetilde{\Phi}$ is generalized **superposition** symbol. The structure of an AcR can be presented as follows:

$$\widetilde{\Phi}(S, \{\wedge, \vee, \neg\}) \to NP' \quad (NP' \subseteq NP) \qquad (3)$$

Here the set *NP'* contains the identifiers of numeric parameters which values are important with respect to the specific situation described by the AcR

condition, and NP is a set of all numeric technical process parameters defined in the PCS.

The concept of Association Rules (AsR) is used for representing the regularities mined from the cumulative data. The classical structure of Associative Rule (see [Sa94]) is given as: $A \Rightarrow B$, where A and B are predicates defined on a set Q of database transactions (records). Each an AsR is considered with its numerical characteristics: support level *s* and confidence level *c*. Support level shows (in percentage) the number of transactions that meet the condition of AsR, and confidence level shows (in percentage) the ratio of number of transactions that meet AsR conditions to number of transactions that meet its left-hand condition..

The following typical structures of AsR are used in our IDE framework for the representation of regularities in technical process parameters:

- $\bigwedge_{j \in U} \left(p_j \in \left\lfloor \underline{c'_j}, \overline{c'_j} \right\rfloor\right) \rightarrow \bigwedge_{k \in V} \left(p_k \in \left\lfloor \underline{c'_k}, \overline{c'_k} \right\rfloor\right)$, $p_j, p_k \in NP$, $U \cap V = \varnothing$,

$U \subseteq NP, V \subseteq NP$ - for regularities in numeric parameters.

- $E_L \rightarrow E_R$, $E_L \subset BP, E_R \subset BP$, $E_L \cap E_R = \varnothing$ - for regularities in boolean parameters.

When AsR R_i is received after the analysis of functioning history of some existing AcR A_j, we can find the confidence and support level separately for the pre-history, progress, and post-history zones of this rule ($s_i^1, s_i^2, s_i^3, c_i^1, c_i^2, c_i^3$).

Definition. AsR A_j is valid for some of the zones of AcR R_i if it has the confidence and support levels for this zone greater that pre-defined minimum values $s_{\min}$ and c_{min}. It is valid for the whole AcR, if $s > s_{\min}, c > c_{\min}$.

Statement. If AsR A_j is valid with respect to some AcR R_i, then it is valid for at least one of its zones.

Let us introduce the following predicates:

- $Eff(A, R, n)$ is a predicate claiming that AsR *A* is valid for a zone *n* of an AcR *R*
- $Eff(A, R)$ is a predicate claiming that AsR *A* is valid for all zones of an AcR *R*.

Then we introduce the *condition of range intersection*. Let us consider AsR A with the structure $\tilde{r}_1(p_1) \rightarrow \tilde{r}_2(p_2)$ and AcR R with the structure:

$r_1(p_1) \wedge r_2(p_2) \ldots \wedge r_n(p_n)\ \alpha\ \{p_{n+1}, p_{n+2}, \ldots, p_{n+m}\}$

Here *r(p)* is a range conditions on the values of numeric parameter *p* that has the form $p \in \left\lfloor \underline{c}, \overline{c} \right\rfloor$. The range intersection condition has the following structure:

$$\hat{r}_1(p_1) = r_1(p_1) \cap \tilde{r}_1(p_1),\ \hat{r}_1(p_1) \neq \varnothing;$$
$$\hat{r}_2(p_2) = r_2(p_2) \cap \tilde{r}_2(p_2),\ \hat{r}_2(p_2) \neq \varnothing$$

It is also shown in the Fig. 3:

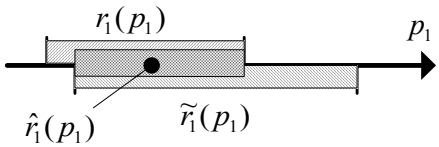

Fig. 3 – Graphical interpretation of range intersection condition

Rng (R,A,p_i) is the predicate claiming that for an AsR *A* and an AcR *R* the range intersection condition holds true with respect to a numeric parameter p_i. These predicates a very important when using the *modification patterns* which are included in our IDE framework. They play a significant part in the common scheme of PODA shown in the Fig.4.

In order to begin PODA we need to specify the initial set of AcRs that determines the specific situations in the controlled technical process. Since these AcR are defined in low-level node by technical process operator, those definitions can be redundant, i.e. their conditions can be confluent or never hold false. As it was shown in the paper [Ku04], the number of AcRs and complexity thereof (which can be indirectly determined by the number of elementary conditions in AcR) are the factors that can critically influence on the system performance with respect to real-time limitations that are important for low-level node. Therefore, the idea of PODA is not just to figure out the regularities in technical process parameters but rather use them for the structure simplifying of some complex AcRs.

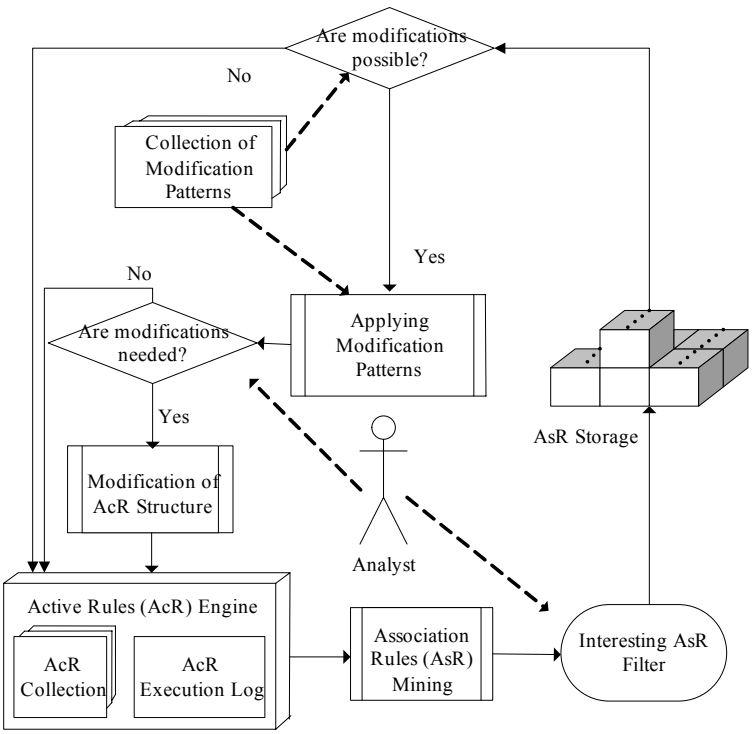

Fig. 4 – PODA cycle scheme

After the initial AcR set is complete the following stages are performed in PODA:

1) During the technical process operating the AcR are executed as a reaction on specific situations occurred. The appropriate PCS-parameter values are accumulated in the AcR Execution Log.
2) The AsR are mined using accumulated data.
3) An analyst (any person, who is supposed to be an expert in the appropriate PCS-domain) filters out the primitive or unimportant regularities found after AsR mining.
4) The accepted AsR are saved into the AsR storage
5) The PODA service checks out the possibility to simplify the structure of existing AcR. The collection of so-called modification patterns is used for this purpose (see below for more details).
6) If appropriate modifications are found, the analyst evaluates their possible results, and approves acceptable ones.
7) The AcR set at the low-level node is modified.

The steps 2-7 form the life-cycle model of PODA. Its scheme is shown in Fig.4.

4. Modification Pattern (MP) for AcR: Definition and Examples

The structure of a modification pattern *P* is following:

$$R' = P(R, A, U) \qquad (4)$$

Here *R'* is a modified AcR, *R* is a legacy AcR, *A* is extracted AsR, and *U* is a set of conditions that let the pattern be applied.

Let us consider some examples of modification patterns. The 1st pattern is used for excluding a confluent elementary condition from an AcR Condition. Confluent elementary condition the one that is overlapped by another elementary condition presented in same AcR. According to (4) we specify the following elements of this pattern:

R: $r_1(p_1) \wedge r_2(p_2) \wedge r_3(p_3) ... \wedge r_n(p_n) \text{ ☋ } \{p_{n+1}, ..., p_{n+m}\}$. Thus, the legacy AcR contains a set of range conditions on numeric parameters.

A: $\tilde{r}_1(p_1) \to \tilde{r}_2(p_2)$. The AsR represents dependency between two of the parameters that are present in AcR.

U: $(Eff(A, R, 2)) \wedge (Rng(A, R, p_2)) \wedge (Eff(\hat{r}_1(p_1) \to \hat{r}_2(p_2), R, 2)) = 1$. The AsR A must be effective with respect to progress zone of AcR R. The condition of range intersection with respect to parameter p_2 holds true. Also, the AsR of the form $\hat{r}_1(p_1) \to \hat{r}_2(p_2)$ is effective with respect to the progress zone of *R*.

R': $r_1(p_1) \wedge r_3(p_3) \wedge ... \wedge r_n(p_n) \text{ ☋ } \{p_{n+1}, p_{n+2}, ..., p_{n+m}\}$. The elementary condition on parameter p_2 is excluded from R.

Another example shows the possibility to merge two AcRs into one (*Rule Merging* pattern). In this pattern the legacy AcRs have the following structure:

$R_1 : r_1(p_1) \wedge r_3(p_3) \wedge ... \wedge r_n(p_n) \text{ ☋ } \{p_{m1}, p_{m2}, ..., p_{mn}\}$

$R_2 : r_2(p_2) \wedge r_3(p_3) \wedge ... \wedge r_n(p_n) \text{ ☋ } \{p_{k1}, p_{k2}, ..., p_{kn}\}$.

The structure of mined AsR is the same as in previous pattern: $\tilde{r}_1(p_1) \to \tilde{r}_2(p_2)$. The conditions *U* have the structure like:

- $C(R_1) \cap C(R_2) = C(R_1) - \{p_1\} = C(R_2) - \{p_2\}$. Here $C(R_i)$ means the condition of AcR R_i.
- $(Eff(A,R,2)) \wedge (Rng(A,R,p_1)) \wedge (Rng(A,R,p_2)) = 1$.

The new AcR *R'* received by merging R_1 and R_2 has the following structure: $r_1(p_1) \wedge r_3(p_3) \wedge ... \wedge r_n(p_n)$ ☋ $\{p_{m1},...,p_{mn},p_{k1},...,p_{kn}\}$.

We also introduce the modification pattern for Boolean parameters (*Boolean* Pattern). In this pattern the boolean parameters from the right side of mined AsR can be excluded from legacy AcR:

R: E ☋ $\{p_1,...,p_n\}, E \subset BP$

A: $E_L \to E_R \;\; E_L \subset BP, E_R \subset BP, \; E_L \cap E_R = \varnothing$

U: $E_L \subset E, E_R \subset E$

R': $E - E_R$ ☋ $\{p_1,...,p_n\}$.

5. Experimental scheme and some results

In this section we present some experimental results of our data analysis approach as well as practical examples that describe some steps of AcR - AsR modification scheme. The scheme of the experiment is shown in Fig. 5.

For mining the AsR we use the fuzzy transaction data-mining algorithm (FTDA) proposed in [HK99]. This algorithm transforms quantitative values into linguistic terms and filters them to find AsR. We select this algorithm because it allows to obtain quantitative AsR and can be easily implemented.

Using the following example the experiment stages can be demonstrated. We consider the following AcR:

$$(P_4 > 6.9) \wedge (P_{10} > 7) \wedge (P_{13} > 63.01) \wedge (P_{14} > 90) \wedge (P_{18} > 66.1) \wedge (P_{19} > 20) \wedge (P_{20} > 2) \to$$
$$\to \{P_4, P_{10}, P_{13}, P_{14}, P_{18}, P_{19}, P_{20}\}$$

We determine the membership functions for each parameter in this AcR. In order to generate membership functions according to [HL96] we should determine the variables *C* and α. *C* is a parameter affecting the shape of the membership function [HL96]. The larger value of α will signify the smaller number of groups in membership function. In our experiment we took *C=4*, *α =0.5*. Parameter numbers are shown in the first column; the next three columns contain the parameters of triangle membership functions (MF). The value of such MF equals 0 in points A and C and equals 1 in point B. As an example the graphical chart for membership function of parameter 9.is shown in Fig. 6, and membership function values received on this step are shown in Fig. 7.

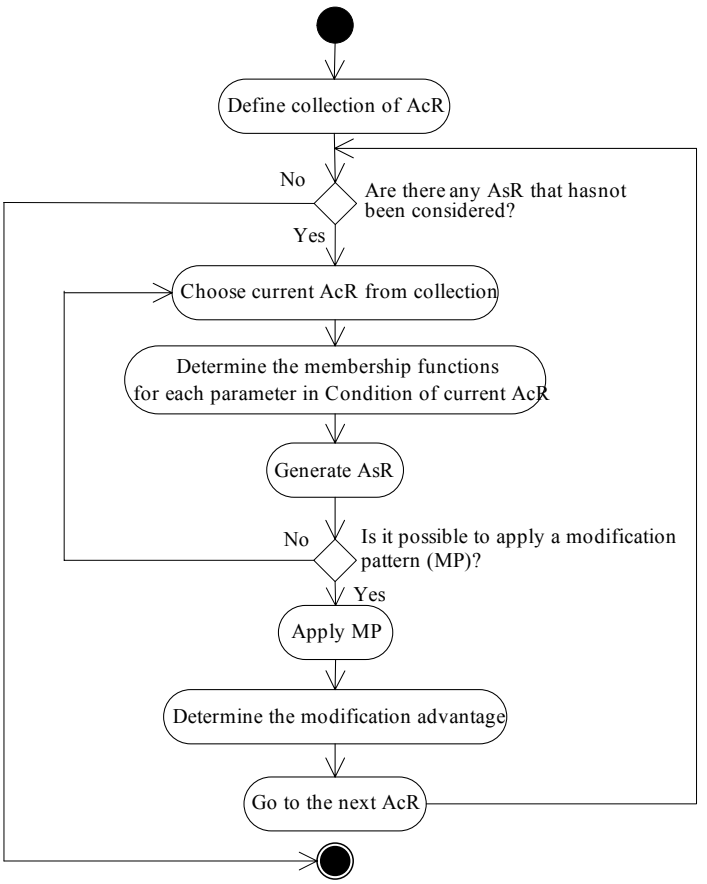

Fig. 5 - Scheme of experiment

After FTDA has been applied we receive the following AsR:

If $P_{14} \in [91.33,+\infty]$ and $P_{18} \in [76.79,+\infty]$ and $P_{19} \in [-\infty,62.01]$ then $P_4 \in [6.73,6.97]$.

Applying this AsR according to the Parameter Exclusion pattern we receive the following AcR:

$(P_{10} > 7) \wedge (P_{13} > 63.01) \wedge (P_{14} > 90) \wedge (P_{18} > 66.1) \wedge (P_{19} > 20) \wedge (P_{20} > 2) \rightarrow$

$\rightarrow \{P_{10}, P_{13}, P_{14}, P_{18}, P_{19}, P_{20}\}$

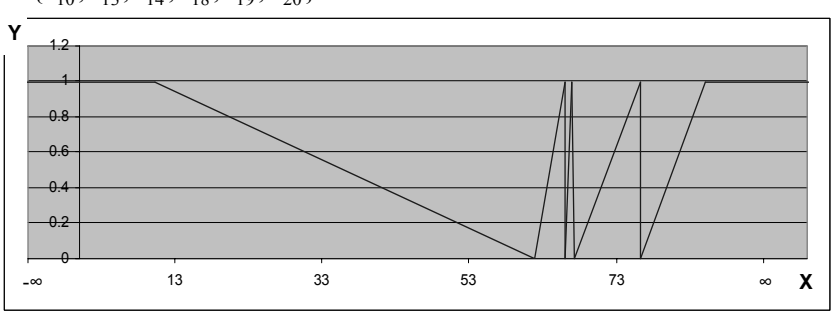

Fig. 6 - Membership function for the parameter 9.

Parameter No	A	B	C
4	6.73	6.83	6.97
4	6.96	7	7.09
4	7.05	7.14	7.37
10	-∞	6.15	+∞
13	-∞	61.69	+∞
14	-∞	89.79	91.1
14	91.09	91.32	91.34
14	91.33	91.93	+∞
18	76.83	78.17	+∞
18	76.78	76.82	76.84
18	-∞	19.75	76.79
19	62	66.07	66.09
19	-∞	10.18	62.01
19	67.36	76.25	76.27
19	66.08	66.93	67.37
19	76.26	85.17	+∞
20	44.99	82.18	+∞
20	33.58	44.98	45
20	1.99	32.10	33.59
20	-∞	1.37	1.76
20	1.75	1.98	2

Fig. 7 - Parameters of membership functions

We have carried out 10 experiments with 10 different AcR. We have manually created these AcR making their complexity different. For each AcR the appropriate AsR were created. After that we modified AcR based on AsR. The experimental results are shown in Fig. 8.

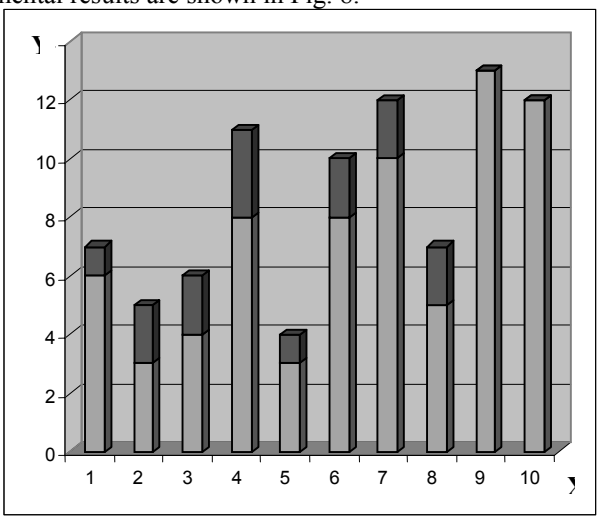

Fig. 8. Experimental results

Here the X-axis shows the number of AcR evaluated, and the Y-axis shows the complexity of AcR. The advantage of the modification shown in top of charts is marked with the different color.
Based on the experimental results we can say that having used our approach we get 19.96% decrease of total AcR complexity. This approach allows to improve the performance in real Web-based PCS, which were elaborated and actually are used in some Ukrainian gas-and-oil production enterprises [Ku02,Ku04,Tka03].

6. Related works

The research in the field of Active Databases was mostly popular in the 2nd half of 90s. Most of the papers published that time were dedicated to the problems of Active DBMS reference architecture and functionality thereof (e.g. [AC96]), consistence of a set of AcRs ([MT02, BW00]), extensions of traditional AcR model, esp. cooperating the temporal operators in Condition section of an AcR (see, e.g. [SW95]). [MT02] considered the problem of AcR optimization, but described only the static structures of AcRs and mentioned also the problem of DBMS consistence. But at the end of 90s, since the concept of AcR had been implemented in many DBMS (mostly in a form of trigger), the research on this topic slowed down
At the same time, the plenty of papers related to AsR (e.g. [HL96, HK99]) and methods of their mining was published. Most of the papers that presented methods of AsR extraction did not consider the implementation of knowledge that were obtained in the form of these rules. Especially we have not found any research that provided the feedback that based on the extracted knowledge to the lower-tier data collecting service.

7. Conclusions

During our research we have elaborated the knowledge-based framework called Intelligent Data Engineering. We have also estimated its effectiveness based on the data we had obtained in our real-life projects. In our experiment the total AcR complexity was decreased on approximately 20% during one PODA cycle.
In our experiment we have used FTDA algorithm only. This algorithm has been chosen just to demonstrate the ability of PODA functioning. We haven't compared the efficiency of FTDA and other AsR mining algorithms. We suppose that more efficient algorithm can improve the efficiency of PODA. Also we can try to achieve this by varying the pre-defined minimum confidence and support level used in AsR mining algorithms (during our experiment we used the values $s=0.1, c=0.7$).

References

[AC96] ACT-NET Consortium Joint Report. The active database management system manifesto: a rulebase of ADBMS features. ACM SIGMOD Record, ACM Press, New York, USA, v.25, i.5, 1996, pp. 40-49

[BW00] Baralis E., Widom J. Better Static Rule Analysis for Active Database Systems. ACM Transactions on Database Systems, 2000
[IG00] Information and Installation Guidelines for Advanced Control Systems for Isolated Power Networks. Technical report. The Symposium «Dissemination of the advanced control technologies and SCADA-systems for the isolated power network with increased use of Renewable Enegries», Ajaccio (France), 2000.
[HK99] Tzung-Pei Hong, Chan-Sheng-Kuo. Mining association rules from quantitative data. Intelligent Data Analysis #3, 1999.
[HL96]Tzung-Pei Hong,Chai-Ying Lee. Induction of fuzzy rules and membership functions from training examples. Fuzzy Sets and Systems, 1996.
[Ku02] D.V.Kuklenko, H.C. Mayr, et al. "Web-Based Information Systems For Technological Process Control: Architectural Framework And Software Solutions", "Problems of programming", the magazine of Science Academy of the Ukraine, Kyiv, #1-2, 2002 (special bulletin), pp.317-325.
[Ku03a] Kuklenko D. An Approach to the Intelligent Data Processing in SCADA systems based on the usage of Active Database Concept. Proceedings of Technical Science Seminar "Interpartner'2003", September, 2003, Crimea, Ukraine, pp.51-56 (in Russian with abstract in English).
[Ku03b] Kuklenko D. The concept and models of Integrated Node Database. NTU "KhPI" Scientific Bulletin, Kharkiv, #7, 2003, pp. 67-68 (in Russian with absctract in English).
[Ku04]Kuklenko D., Gamzayev R., et al. An Approach to Advanced Data Synchronization in Complex Process Control System. Lecture Notes in Computer Science, GI Edition, Bonn 2004, pp. 221-227.
[MT02]Montesi D., Torlone R. Analysis and optimization of active databases. Data & Knowledge Engineering 40 (2002) pp. 241–271.
[Pa94] Norman Paton, Active Rules in Database Systems. New York, Springer, 1998.
[SA94] R. Srikant, R. Agrawal. Fast Algorithms for Mining Association Rules. Proceedings of the 20th International Conference on Very Large Databases, 1994.
[SW95] A. Prasad Sisla, O. Wilfson, Temporal Triggers in Active Databases, Data and Knowledge Engineering, 7(3), 1995, pp. 471-486.
[Tk01] Tkachuk N.,Kuklenko D., et al. "Knowledge-based Maintenance Environment for Large Information Handling Systems". Lecture Notes in Computer Science, GI edition, P-2, Bonn 2001, pp. 139-154.
[Tk02] M. Tkachuk, H. Mayr, et al. Web-based Process Control Systems: Architectural patterns, Data Models, and Services. Lecture Notes in Computer Science, GI edition, Berlin 2002, pp. 721-729.
[Tk03] N. Tkachuk, V. Shekhovtsov, et al. An Approach to Efficient Data Handling in Web-based Process Control Systems. Proceedings of the IASTED International Conference on Intelligent Systems and control, ACTA Press 2003, Editor M.H.Hamza. pp.242-247.