User interface of diagnostic systems

Wojciech Cholewa[1], Krzysztof Psiuk[1]

[1] Institute of Fundamentals of Machinery Design - Silesian University of Technology
Konarskiego 18a, 44-100 Gliwice, Poland
{Wojciech.Cholewa, Krzysztof.Psiuk}@polsl.pl

Abstract

The paper presents a user interface displaying the operation results of a diagnostic system based on multilayer statement networks. The design of the interface facilitating the monitoring of the reasoning process and presenting the obtained results of the diagnostic system was proposed. The functionality of the proposed user interface was described. The operation of the diagnostic system which is based on multilayer statement networks was presented. The method of knowledge representation in such a system was described together with the method of reasoning.

1 Introduction

One of the main objectives of technical diagnostics is the assessment of technical condition of devices and machines. Owing to the development of methods involving both the possibility to carry out measurements and also the processing of the existing data, the assessment of technical state can be carried out basing on a great amount of data. Such a large number of input data leads to a bigger number of investigated diagnostic symptoms. Furthermore, the collected data is very frequently burdened with uncertainty and inaccuracy resulting from the carried out measurements and from changing conditions in which they were carried out. In such cases, it is recommended to apply expert systems which are supposed to support the work of both diagnosticians and maintenance staff. Two categories of expert systems are considered [1][2]: static systems and dynamic systems. Static systems are characterized by the fact that they most frequently work in a so called “off-line” mode and they support the search for solutions in fixed environment. And dynamic systems most frequently work in the “on-line” mode and in the changing environment.

Dynamic expert systems are applied for the realization of tasks, in limited time and with limited resources. A special category of dynamic systems involves the systems for continuous (cyclic) analysis (interpretation) of data these systems are fed with. When adapting such systems for the needs of technical diagnostics, we can allow for simplifications resulting from the limited dynamics involving the changes of technical state of real objects. One of the methods consists in freezing out the external impact acting on the system (description of the surroundings) for the duration time of the basic reasoning cycle. In such an approach, we can treat the dynamic system as a quasi-static system (for the duration of one cycle). By freezing out the surroundings, we can reduce many inconveniences which occur in dynamic systems.

A different option was applied in dynamic systems with a guaranteed reasoning time. Such systems provide solutions which according to the knowledge accumulated in them are the best solutions which can be provided in the preset, limited, available time.

Typical diagnostic systems consist of some specific modules. Among others they include the modules of data collection, data storage, knowledge base or user interface. The task of the data collection module from the monitored machine is to accumulate such information as diagnostic data, processing data, etc. Another module is responsible for storing the collected data in a database, for further analyses with various methods. The knowledge base is a module of the diagnostic system responsible for gathering field knowledge and its application in the reasoning process involving the technical state of the monitored object. Another interesting option involves the knowledge base interpreted as a network of statements. The diagnostic system investigated in this paper is a dynamic system based on multilayer statement networks. Since the tasks realized by particular modules of this system are carried out in an asynchronous way, therefore the exchange of information between particular modules must be ensured. Such an exchange of information was accomplished in the form of a blackboard. The structure and tasks realized by these modules have been widely discussed in literature, e.g. [7][8][9]. We can find information on blackboards, data collection modules, their storage and methods of reasoning process. Rarely, however, can we find a description involving the form or functionality of a user interface for such systems.

2 Diagnostic system

2.1 Multilayer statement networks

One of the methods of knowledge representation involves belief networks. The concept of belief networks is based on the Bayes theory of conditional probability. Belief networks are made up by such elements as nodes and connecting them directed edges. In such networks, each node of the network is ascribed a set of mutually excluding states and vectors of probability value. Particular probability values can be interpreted in these networks as the belief level about the fact that a given node is in a definite state. For each node, an array of conditional probability values for parent nodes should be also described. Such an array allows for a combination of all possible states of parent nodes. Such an array is treated as a description of relations which connect a given node with other nodes of the network.

The concept of belief networks has been extended into statement networks [1]. In statement networks, each node of the network is described by means of a statement, and the connections between the nodes are described by the relations defining the influence of a given statement on other statements. A statement is understood as an utterance stating observed facts or representing a given opinion. A statement can be written down as a pair:

$$s = < c, v > \tag{1}$$

where *c* is the content of the statement and *v* is its value.

The content of a statement is the information that a given object has been ascribed an attribute of a definite value. And the value of a statement is defined as the truth level of the utterance being the content of the statement. The statements can be of two types. We are talking about precise statements, having the values like <*true*, *false*> and about approximate statements whereof values belong to the range [0;1].

In expert system, knowledge is very frequently written down in the form of rules. In the case of statement networks, the elements of rules i.e. premises and conclusions are represented by means of statements. In the inference process in such a system the rules are replaced by necessary conditions and sufficient conditions which accept the truth of a given statement. In the case of precise statements, a necessary condition is understood as a condition in which, in order to accept the truth of statement *x*, the acceptance of statement *y* is required, but not necessarily the other way round. Then we say that *y* is treated as a necessary condition for *x*, and *x* as a sufficient condition for *y*. In the case of approximate statements, necessary or sufficient conditions are examined in a slightly different way. The values of dynamic statements are written in the following form:

$$b(x) \in [0;1], \quad b(y) \in [0;1] \tag{2}$$

In such a case the necessary and sufficient conditions can be written in the form of the following inequality:

$$b(y) \geq b(x) \tag{3}$$

The conditions described in this way are defined in a precise way. If we want to allow for a certain imprecision of these conditions, i.e. to determine necessary and sufficient conditions in an approximate way, then they assume the following form:

$$b(y) \geq b(x) - \delta; \quad \delta \geq 0 \tag{4}$$

The number δ describes the permissible approximation level of the condition, and it is a non-negative number.

The acceptance of necessary and sufficient conditions (3) and (4) enables to build a relation between the statements. For the construction of statement network, we make use of a closed set of statements from which a network is built. For a particular statement, necessary and/or sufficient conditions are determined, defining in this way the relations binding the individual statements. If there are no contradictions in such a network, their solution will not necessitate the application of complicated calculation algorithms. When contradictions do occur in such a network, then its solution will come down to finding a minimum of a certain criterion function, considered e.g. in the following form:

$$e = \sum_{x,y} k_{x,y}^2 \delta^2 \tag{4}$$

where $k_{x,y}$ is a coefficient defining the significance level of the condition defining the influence of statement x on the statement y.

When we investigate complex technical objects, the number of statements needed to define the state of such an object can be very big. It may lead to the situation where the constructed statement network is very big and complicated, and hence longer time can be needed to calculate it. Therefore, when there are many statements, multilayer networks are applied [3]. A multilayer statement network is a set of smaller statement networks organized in layers. For the construction of statement network for particular layers, we make use of a subset of statements from the group of all statements worked out for a given object. Each of the available statements can occur on any layer of such a network. The connections between particular layers are realized by means of statements occurring simultaneously on several layers. The transformation of such a multilayer network into one network can be done in two ways. In the first approach, the connection of networks is taking place in the aggregation process of node values in which particular nodes act as independent bodies in subsequent layers of the network, where their values are defined independently. In the second case the merging of networks is taking place in the equalization process, where each node occurs as a single copy in all layers of the network. In such an approach, it is necessary to work out appropriate procedures to equalize node values in particular layers of the network.

The application of multilayer networks has many advantages. In this approach, we can build particular layers of the statement network in an independent way, by various experts. We can also build networks in which particular layers of the network can be built as models of different types.

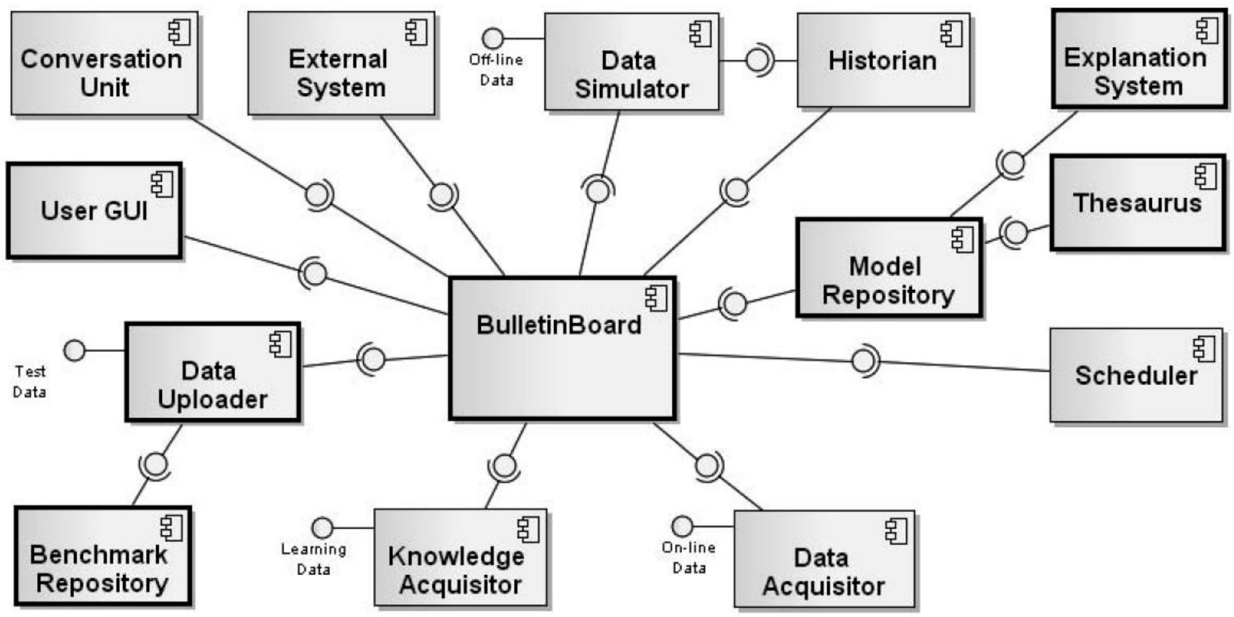

Figure 1: Components of REx system [2]. The existing version of REx platform (http://ipkm.polsl.pl in section Projects) contains elements marked with thicker frame

2.2 REx system

The REx system [2] is a tool enabling the creation of hybrid expert systems based on multilayer statement networks. It enables the creation of knowledge bases for complex objects on the basis of many sources of knowledge. It offers an easy and transparent construction way of knowledge bases. Thanks to the application of multilayer statement networks, it enables the construction of knowledge bases by a team of independent specialists from many fields and on the basis of the carried out diagnostic experiments. The software enables proper control and management of the accumulated knowledge. The REx system has a number of built-in inference algorithms which facilitate the testing and tuning process of the constructed expert system. Since the REx system was constructed in the R language, which is an open environment, it is possible, depending on the needs involving the planned application of the worked out expert system, to develop it by adding additional tools allowing to transform process variables. The structure of this system is presented in Fig.1. The figure presents a complete structure of REx system. The modules which have been already worked out are marked with thicker frames..

The operation of the REx system is based on the application of blackboard. The concept of blackboard has been applied in many types of expert systems. The application of the blackboard concept in the case of diagnostic systems seems to be particularly grounded. A blackboard is a place where various information is placed on currently valid values of statements. The elements of the blackboard are active elements, i.e. the change of the value of one statement can entail the value change of other statements. The values of statements on the blackboard can be changed by different users of such a system. In general, they can be changed by different modules which are responsible for the determination of the value of a given statement. Other modules can read the values of statements placed on the blackboard, and basing on the read information they can update their own values. The blackboard itself is very often realized in the form of a single table in the relational database or a set of tables in such a base.

3 User interface of the diagnostic system

3.1 Requirements for the blackboard interface

In the case of expanded expert systems based on multilayer statement networks, looking through the contents of the blackboard may become quite a challenge. Big statement networks can contain several hundred or even several thousand statements whose values are changing dynamically in time. What is more, the determination process of statement value can be carried out in many steps, since the values of some statements can have influence on the values of other statements. Therefore, the values of all statements are written down on the blackboard. In the REx system, which enables to define such multilayer statement networks, we can define two types of statements. They are referred to as primary statements and secondary or auxiliary statements. Furthermore, the statements can belong to different layers of statement networks, defined within the frames of one expert system. The same statements can be also the elements of different networks. Therefore, it is necessary to employ an appropriate presentation method of both the content of the statements and their value, which can be found on the blackboard, and to work out an appropriate tool which would facilitate the browsing process of the blackboard contents.

To facilitate the browsing process of the contents, an explorer had to be worked out to look through the contents of the blackboard. The worked out interface of the blackboard should satisfy not only some definite requirements involving the construction of a user interface, but also requirements involving how the information about the statements on the blackboard is presented. It resulted not only from a great number of statements placed on the blackboard but also from the way how they occurred in statement networks and how they were described. The description of a statement contains two types of descriptions, full descriptions and shortened ones. The problem was to decide which descriptions were to be presented to the user.

The designed blackboard explorer was to satisfy the functionalities covering among others the following:

- Presentation of the content of statements,
- Presentation of currently valid values of statements,
- Presentation of historical values of statements
- Grouping process of statements,
- Filtration possibility of statements,
- Possibility to distinguish relationships between particular statements,
- Possibility to work in the on-line/off-line modes.

The basic information that was to be presented to the user was the content of the statement and its values (both current and historical). In the REx system [2], the content of the statement is written down in a full way (describing actual content of the statement) and shortened way (short content of the statement). The values of statements which are the elements of multilayer statement networks can be written down in different ways. Some statements can have only two values yes/no. Other statement can be multi-variant statements whose values are changing in the discrete or continuous way. Therefore, it was necessary to work out a uniform presentation method of statement value.

From the user's viewpoint, in terms of the functionality of the expert system, it is important to be able to observe the currently valid statements made available by the browser. Furthermore, in many applications it is also important to be able to analyze the changes of statements in the past. Therefore, another requirement which a blackboard browser was supposed to satisfy was the ability to observe what happened earlier in time.

A large number of statements present on the blackboard impeded their clear and transparent presentation in the browser. Therefore, it was extremely important to group and aggregate them appropriately to make them clear and transparent for the user. Filtration methods of statements were proposed, which enabled to distinguish between primary and secondary statements, active and inactive, statements ascribed to the selected elements of the object and statements ascribed to the network of statements. These methods allowed the user to move around the blackboard more freely. One can consider the application of other filters, which would enable e.g. to separate out statements of some definite parameters (informative, warning, alarming statements), or to separate statements describing elements of the object for which the system was constructed. Particular statements belonging to a given expert system are connected with relations defined in the model of this system.

3.2 Assumptions to the systems

Having completed the analysis involving the requirements which a blackboard explorer should meet, a concept of appropriate software could be proposed. It was assumed that the blackboard would be realized in the form of two tables written down in the relational database. The first table was to contain information about statements. It was to be used to write down the content of statements, their shortened descriptions and explanations. The second blackboard was to be used to write down the changing in time values of statements together with the time marker. The blackboard explorer was to collect statement values from the relational database (Fig.2). The values of statements on the blackboard were to be modified by external data sources. The access to the statements from the blackboard was to be realized by a set of functions defined as API for the prepared blackboard. By the application of an option it was possible to access the blackboard from different external equipment.

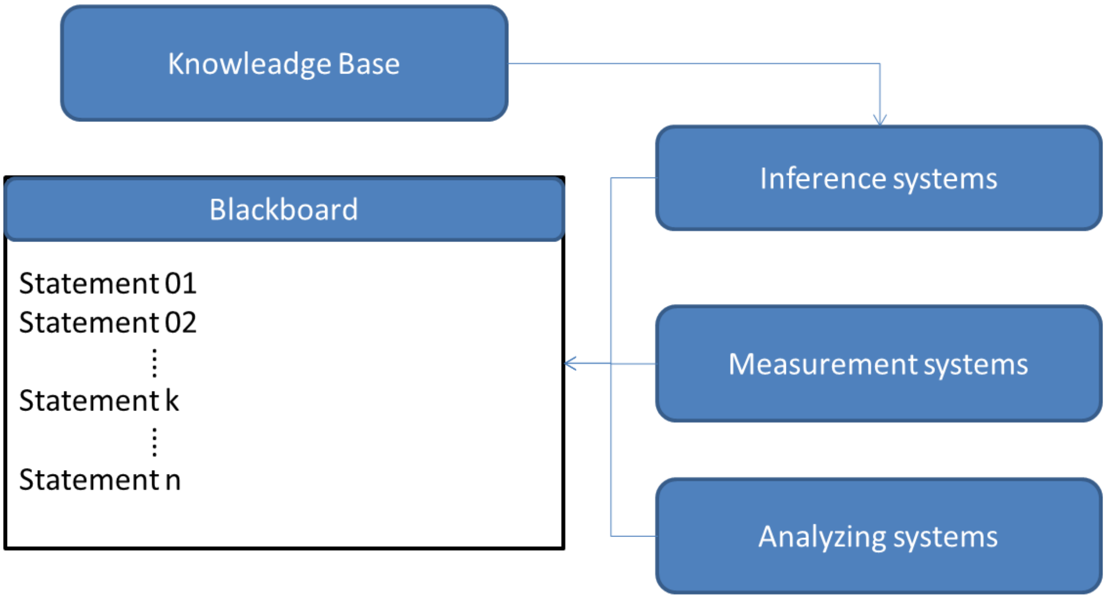

Figure 2: Blackboard structure

The concept of user interface of blackboard explorer was based on the solution proposed in [4][5]. An example of such an interface is presented in Fig.3. The left-hand side of the screen presents the content of statements and their current values, and on the right we can see the values of statements from the previous moments of time.

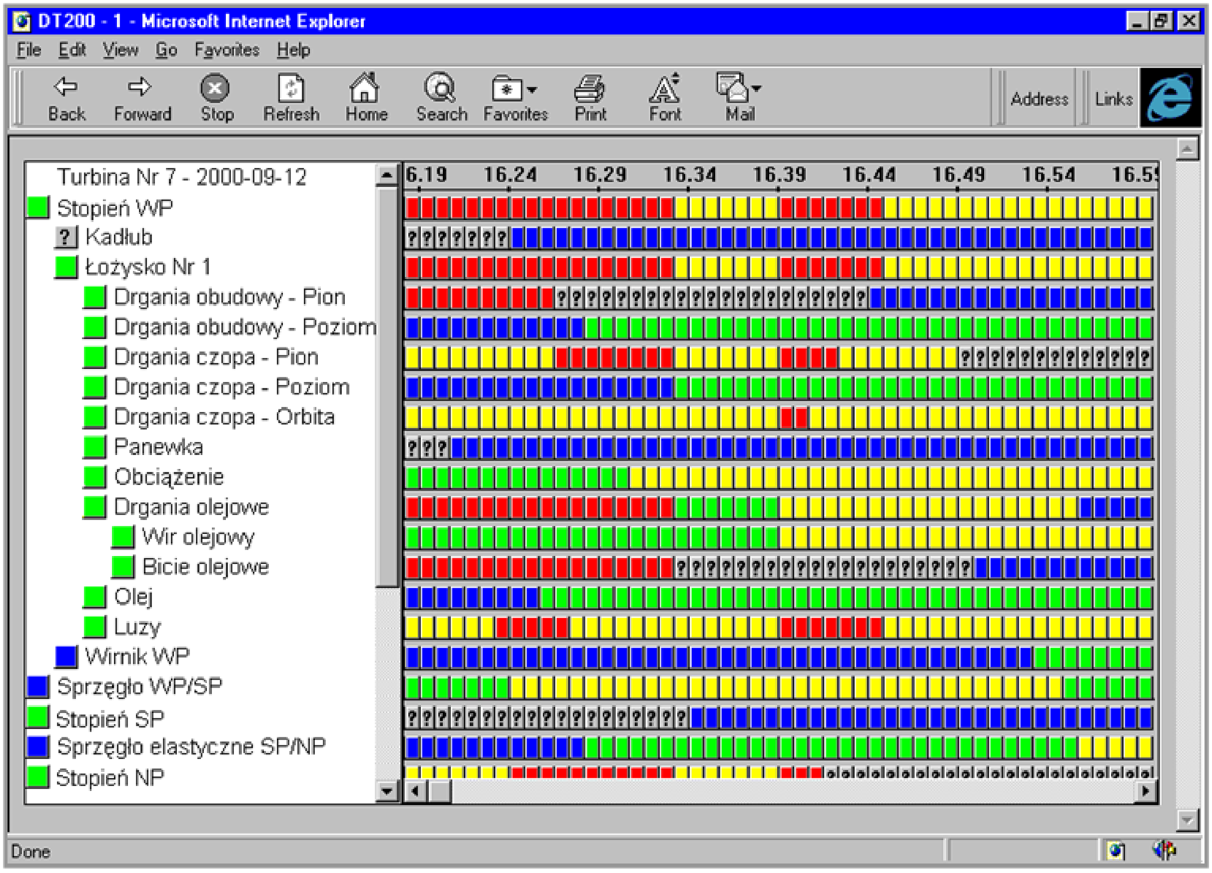

Figure 3: Prototype of the blackboard interface [4]

The proposed blackboard interface is based on the above example in Fig.3. The blackboard explorer can present different networks of statements available within one model. The elaborated software enables the following:

- Switching over between the available networks of statements,
- Preview of the current value of statements,
- Preview of full content of statements,
- Browsing through historical vales of statements (within a selected range in online mode),
- Browsing through historical values of statements (within full range in offline mode).

3.1 Blackboard browser

An early concept of blackboard browser presented in Fig.3 still requires a few modifications. The modified version of a window of such a browser is presented in Fig.4. Due to a multilayer character of the models which can be created in the REx system, it was necessary to write down the information about defined networks in the way ensuring that they could be used in the blackboard explorer. The user of the blackboard browser can switch over between particular statement networks available within a given model. It was assumed that the basic structure presented to the user would be a tree-shaped set of statement networks defined within one model. Each of the models could be written in the form of a dropdown tree. Particular elements of the tree are made up by statements available within one model and the descriptions which the user can enter by themselves. In the description of the tree structure, shortened descriptions of statements are applied, and full contents of statements are displayed after selecting of a given statement. The user can also reduce the number of presented statements by the application of various filters which reduce the number of statements displayed in the explorer. Since the explorer can operate in the online and offline mode, certain differences were introduced in each of the mode. In both cases the explorer enables the presentation of historical data. In the online mode, the scope of currently presented historical values of statements is limited to a selected number of values, starting from the present moment. Older values are available in offline mode, after collecting them from the blackboard and writing down in a separate data file.

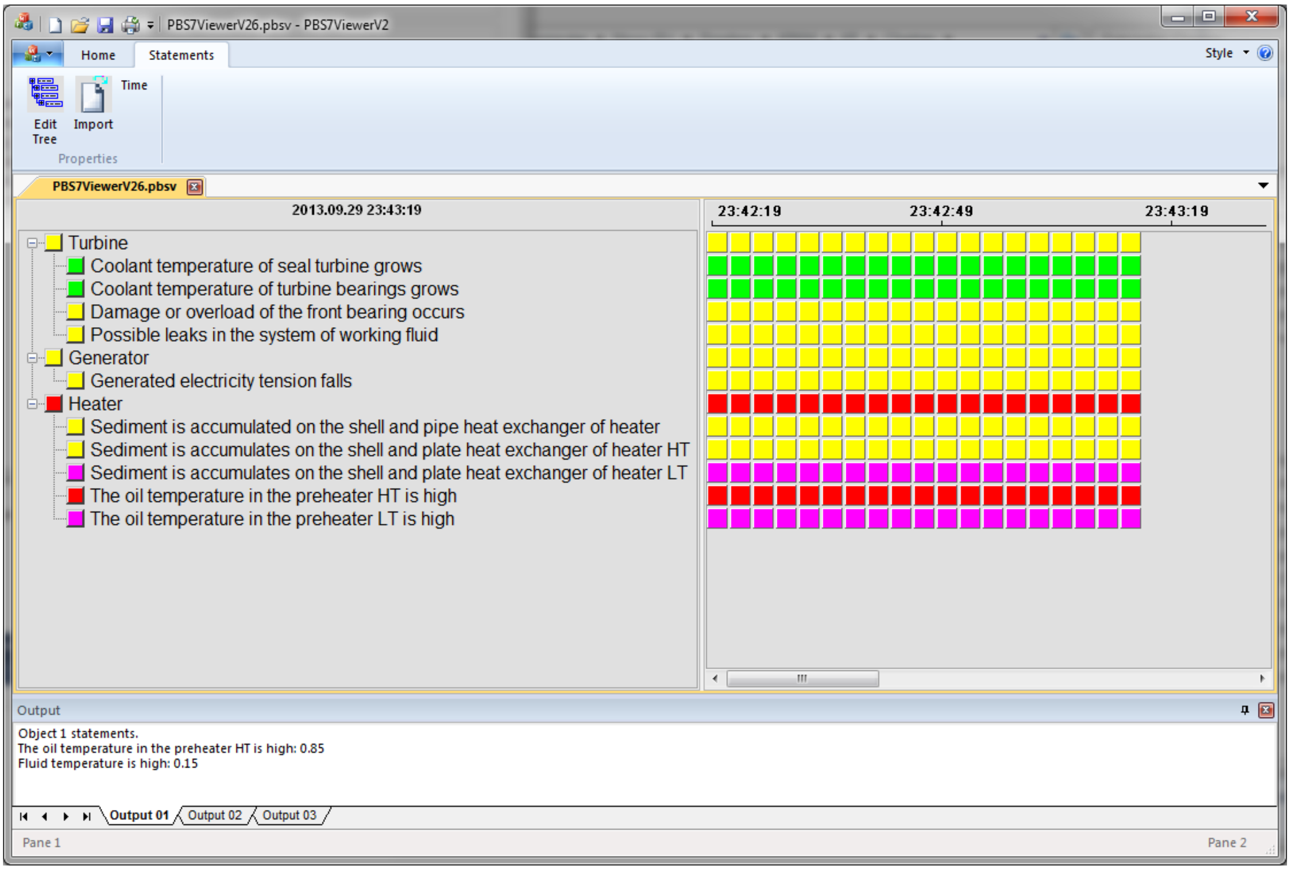

Figure 4: Blackboard structure

Another problem involves the presentation method of statement values. The values of statements are presented in two ways. In the first method, they are presented with colours. In the second method, the actual value of the statement written on the blackboard is displayed.

3.1.1 Statement tree

The statement tree, placed on the left-hand side of the blackboard's browser window (Fig.4), contains structured contents of statements which are made available for a specific object and a given user of the system. From among all available statements, the user of the blackboard browser can select the statements which are interesting to them and build different types of statement tree (Fig.5). For the construction of statement tree, the user can apply the existing statements, collected from the REx system. The user can also add organizing statements with their own descriptions which can also appear as items in the statement tree. The values for such statements are not determined by the REx system. They are determined by the blackboard browser, basing on the values of statements for which organizing statements are parents. The value of such a statement can be the conjunction or the alternative of child statements. If the information that something is not working is important to the user e.g. there are some faults in the functioning of some elements of the monitored object, the alternative of statements reporting malfunction should be applied. In other cases the conjunction of statements reporting normal functioning can be applied. The question whether the browser is to determine the value of descriptive statement as conjunction or as alternative is decided by the user during the arranging of statement tree. If a given organizing statement has no children, its value will remain undetermined.

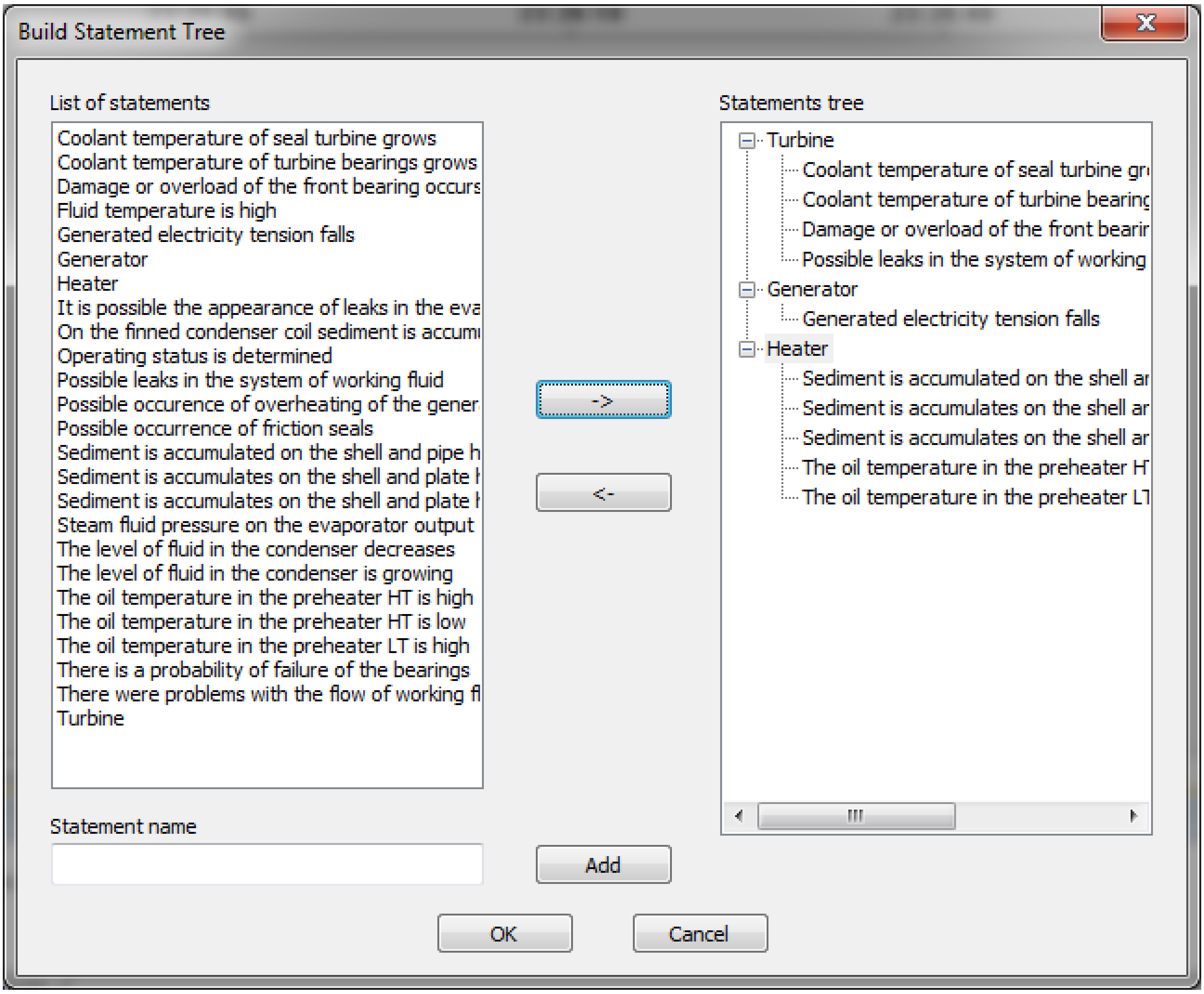

Figure 5: Build the statement tree

3.1.2 Values of statements

The values of statements defined in the REx system can be written down in different ways. The basic way involves the entry in the form of belief degree about the truth of a given statement. Such value is represented by the number from the interval <0:1>. The interval of statement values was divided into several subintervals. Each subinterval was ascribed a fixed colour. Additionally, for certain definite states which the statement value can assume, additional graphic symbols were defined. Basic colours accepted for the graphic presentation of statement values are presented below:

 used to present the value of statements interpreted as normal,

 used to present the value of statements interpreted as a warning,

 used to present the value of statements interpreted as an alarm,

 used to present the undefined value of statements,

? used to present the value of statements which do not occur on the blackboard.

When necessary, the user can define additional subintervals and corresponding to them colours.

When presenting the changes of statement values, they can be regarded as discrete values limited to a few subintervals or as quasi-continuous values. A good presentation method of their value is effected by linearly variable set of colours, as presented in Fig.6. It allows to convey the changes of statement values fluently. In the REx system also multivariant statements can occur. Multivariant statements represent a specific case of discrete statements. Let us discuss it on the example. For a given object, we can define e.g. the temperature of one of its elements. Temperature can be described by means of certain states e.g. low, medium, high temperature. However, each of these states can be declared as true or false. In such a case, such a statement can be replaced with several variants of this statement, where for each variant it is possible to determine the degree of its truth. In the case of multivariant statements, we can speak of the truth degree of its particular variants. In the case of multivariant statements, we suggest a set of colours defined around beige colour, as presented in Fig.7. When there is a greater number of statement variants, it will be necessary to define a palette with a greater number of colours.

Figure 6: Linear representation of statement value

The values of statements presented in the tree are subject to propagation. The propagation of statement value is carried out in the following way. The value of parent statement in the tree of statements can depend on the value of its children statements. When the value of a given statement is entered on the blackboard, it is presented in the statement tree. When the value of statement has not been entered on the blackboard, then it is attributed to the value of one of its children whose value is the highest.

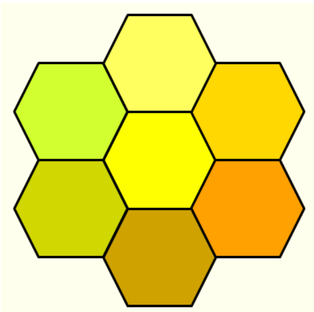

Figure 7: Stepwise representation of statement value

3.1.3 Time

One of major issues connected with the presentation of statement values is time. Each of the statements placed on the blackboard has its time marker. It is used as a basis to define a segment on time axis at which a given statement value will be considered. In the statement tree, we always present statement values at the time moment accepted as current moment. The user can change the current moment of time. And on the right-hand side of the window, the values of statements from a definite time interval are presented. Time axis runs from left to right. So on the left we have the oldest values and on the right the latest ones.

Statement values can be presented in time intervals of different length determined by the user. We can choose time covering a few minutes, a few hours, days or weeks. Since the values of statements are determined by the REx system in definite, small time intervals, therefore in longer time intervals the presented statement values had to be compressed. A similar option was applied in the case of the propagation of statement values in the statement tree. Time periods are divided into smaller subperiods from which the statement value respectively of the highest or the lowest value is selected. In effect, the user can see a colour band presenting a palette of colours. By magnifying such a palette and reducing the time interval displayed by the browser, the user will be able to look easily through the values of statements with different accuracy.

4 Summary

The paper presents the concept of interface which allows to look through the values of statements determined by the expert system REx. Such systems are used to assist diagnosticians in their routine tasks, especially when the investigated machines have complex structure. They should have enough persuasive power to convince the user that the realized inference process is right. The presented blackboard browser offers the reasoning involving the decisions proposed by the system.

Acknowledgments

Described herein are selected results of study, supported partly from the budget of Research Task No. 4 implemented under The National Centre for Research and Development (NCBiR) in Poland and ENERGA S.A. strategic program of scientific research and development entitled *Advanced technologies of generating energy*.

References

[1] J. Korbicz, J.M. Kościelny, Z. Kowalczuk, W. Cholewa (eds.), *Fault diagnosis: Models, Artificial Intelligence, Applications*. Springer, Heidelberg (2004)

[2] W. Cholewa, T. Rogala, P. Chrzanowski, M. Amarowicz, *Statement Networks Development Environment REx*, In Lecture Notes in Computer Science, 6923, Part II, Computational Collective – Intelligence Technologies and Applications. Proceedings of 3rd International Conference on Computational Collective Intelligence - Technologies and Applications. Gdynia, Poland, 2011 September 21-23, pp 30-39

[3] W. Cholewa, T. Rogala, P. Chrzanowski, M. Amarowicz, *Development environment for diagnostic multimodal statement networks*, 5th International Congress on Technical Diagnostics, Kraków, Poland, 3rd - 5th September 2012. Abstracts. pp. 30-31

[4] System DT200-1. Online: http://ipkm.polsl.pl/PROJEKTY/SystemDT200-1/default.htm. November 2013 (in Polish)

[5] K. Psiuk, W. Cholewa, *State visualization of complex technical objects* (in Polish),. IV Krajowa Konferencja Naukowo-Techniczna Diagnostyka Procesów Przemysłowych, 13-16 wrzesien, 1999, Kazimierz Dolny.

[6] F. Dawid, R. Cowell, S. Lauritzen, D. Spiegelhalter: *Probabilistic Networks and Expert Systems*. Springer, New York (2007)

[7] J.C. Giarratano, G.D. Riley, *Expert systems: principles and programming*. Thomson Course Technology, (2005).

[8] E. Castillo, J.M. Gutiâerrez, A.S. Hadi, *Expert systems and probabilistic network models*. Monographs in Computer Science Series. Springer Verlag, (1997).