A Web-Based Intelligent Monitoring Agent for Real-Time Data Processing

Shaban Laban[†*] **and Ali I. El-Desouky** [††],

[†] Consultant, Preparatory Commission for the Comprehensive Nuclear-Test-Ban Treaty (CTBT) Organization, Vienna International Centre, P.O. Box 1200, A-1400, Vienna, Austria.
[††]Computer and Systems Department, Faculty of Engineering, Mansoura University, Mansoura, Egypt.

Summary
Real-time processing contains generally chains of complex, heterogeneous, and critical processes. Different monitoring programs are used for detecting anomalies, failures and tracing workflow progress of the running processes. However, the majorities of the traditional monitoring functions of the workflow systems are usually platform dependent, task specific, lacking alerts capabilities, inflexible, and consuming many of the organization resources. This paper introduces the design and implementation of a generic knowledge-based agent model that overcomes these limitations and restrictions. The proposed intelligent agent is using customized rules for workflow monitoring and generating alerts as well as exception reports to the operators. The suggested rule-based agent is fast, autonomous, configurable, reactive, and platform-independent.
Key words:
Monitoring, Agent, Real-Time, web, and Intelligent Systems.

1. Introduction

The workflow of real-time data processing systems consist generally of series or sequence of complex stages/processes [4]. In every stage, the states of the different objects or elements, as well as their internal attributes, are dynamically changed during the different steps. Generally, The real-time systems use monitoring tools to increase their productivity and efficiency by detecting anomalies, potential workflow failures and tracing workflow progress for the different processes [5,6,7]. The operators of the real-time systems use several instances of the monitoring tools to supervise, control, monitor workflow progress and trace the states of the different resources during the life cycle of monitored systems.

The majorities of the traditional monitoring techniques have many limitations. Such monitoring approaches are usually platform dependent, task specific, lacking reactivity, inflexible, and having limited resources management. Also, they consume most of the organization resources, need longer time, more human resources to maintain or configure. Moreover, usually they lack intelligence decision support. In addition, the monitoring tools are not available to remote users and are not portable.

Furthermore, the monitored data have no standard format or unified structures which make the monitoring tools more complex.

During the last decade intelligent agents have been widely used in complex industrial and business applications [1,2]. Applying agent technology and knowledge-based concepts to workflow monitoring allow flexible interaction and interoperation among heterogeneous systems. Recently agents have been greatly used and integrated with web technology and services [3].

The scope of this paper is to present an approach for implementing a platform-independent and task-unspecific knowledge-based agent model that overcomes the previous limitations and restrictions. The agent uses a simple and flexible format for representing the time-based data intervals as well as its different attributes. Also, the agent is using customized rules for workflow monitoring and generating different exception reports and alerts when necessary.

The overall structure of the proposed approach is explained in Section 2. In Section 3, detailed architecture and design of the rule-based agent are presented. Real-time practical implementation of the proposed system is illustrated in Section 4. Section 5 evaluates the performance of the suggested agent and presents the results of the comparison of the agent with the organization available tools. Section 6 concludes this paper by summarizing the contributions of the proposed agent and discussing future directions.

2. The Proposed Framework

In order to overcome extensive resources utilization, the web-enabled monitoring approach is structured into three main modules as shown in Fig. 1. The arrows in the graph

Disclaimer: The views expressed on this paper are those of the authors and do not necessarily reflect the view of the CTBTO.

indicate the direction of data flow between the different modules.

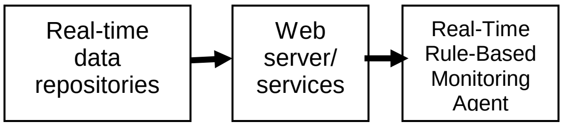


Fig. 1 Overall framework.

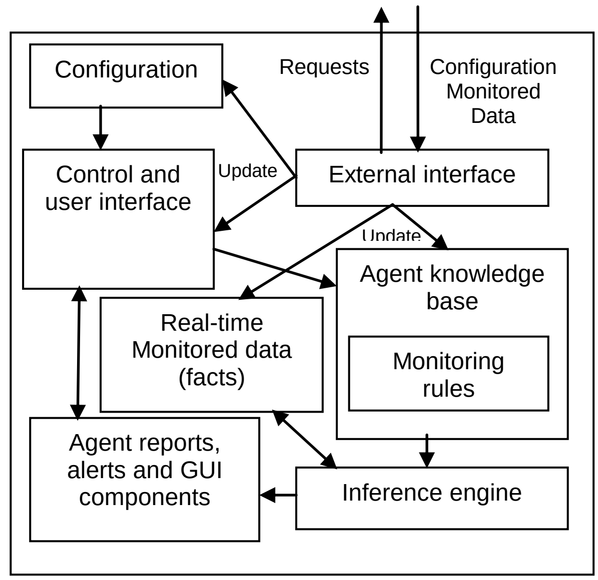
The first module comprises a set of intelligent scheduled or continuous running programs. These programs/services act as data collectors and simple knowledge-based services. They regularly collect status of objects from the different data sources of the monitored system. Then, they infer the monitored data and classify the status of the different objects as well as their necessary attributes/properties from the different stages of the monitored system. Finally, these programs store the data into dedicated data repositories either in an Extensible Markup Language (XML) format or in a simple standard data format [8,9]. Data interval preparation and classification is not limited to the automatic programs. In addition, users can also write their own data repositories that could be used later by the other monitoring agents as well.

The second module consists of any ordinary web server and a set of CGI programs (act as simple data producers) in charge to read the data generated from the first module, prepare the browser configuration data and a user specific views and to send all the information to the monitoring web browsers in Hypertext Markup Language (HTML) using Hypertext Transfer Protocol (HTTP) [10,11].

The third module is our proposed monitoring agent. This agent consists of a set of Java classes or programs that can be run either from the web browsers for all known platforms as a Java applet or in standalone mode [12,13,14]. The agent is reactive, autonomous and communicative.

3. The Monitoring Model

In this paper, the rule-based programming and agent technologies are applied to workflow monitoring in order to provide more flexibility and intelligence to the monitoring process. The agent reads regularly the necessary information from the web server using the HTTP protocol. Then, the agent aggregate, sort, infer, and finally display the monitored data accordingly in the web browser. The agent is using a simple customized rule engine for monitoring purposes as well as generating exceptional reports and early warning alerts to the operators. The agent is able to communicate with its user/operator. It reads the user updated configuration parameters and reacts with user requests and needs. The internal structure including the main functions of the monitoring agent is shown in Fig. 2. The details of our approach are described as follows.



3.1 Agent data representation

The monitoring agent is completely modeled with object-oriented paradigm. In the monitoring agent, the data monitored are represented and modeled with novel interval-based objects or elements [15]. Each object, in our case *IntervalInfo* class, is composed of a set of well defined fields or slots and array of optional attributes (*AttributeInfo* class). The UML class diagram and composition relationship are shown in Fig. 3.

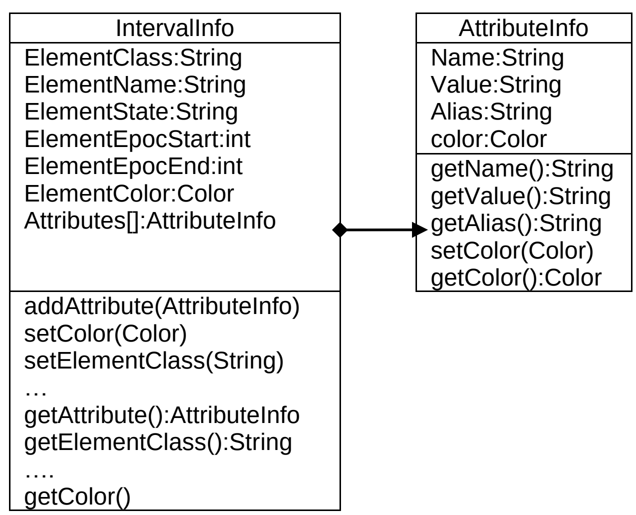


Fig. 3 Class diagram and composition relationship

3.2 Data exchange

In the proposed approach, the states of the interval data can be stored in the data repositories either in dedicated file system or cached from organization databases. The data format of can be any type of format including XML format or simple one-line record per interval data. Although, using XML is the most preferable choice as it provides a foundation for creating documents and document systems.

In addition, the implementation of simple text format is supported and understood by the monitoring agent model in order to support new generations of portable digital assistance devices such as iPhone, embedded systems, as well as other similar device. So, a set of parameters has to be passed to the web-based agent during initialization process as part of the HTML file. These parameters will be sent to the browser from the HTTP server. The parameters will allow the agent to read, parse, and display data from the server.

An example of the parameters is shown below.
<PARAM name="ClassIndex" value="1">
<PARAM name="NameIndex" value="2">
<PARAM name="StartTimeIndex" value="3">
<PARAM name="EndTimeIndex" value="4">
<PARAM name="IdIndex" value="5">
<PARAM name="StateIndex" value="6">
<PARAM name="DataStartIndex" value="7">
<PARAM name="DataSeparator" value=",">
<PARAM name="DataTitles"
value="Maximum Memory Utilization;
Maximum Swap Utilization;">

Each record or interval is sent to the agent in a separate section. The first line and the last line are special keywords to identify the beginning and ending of data transfer.

3.3 Agent rule engine

The agent uses a time-dependent rule engine for states identification and classification. The engine is supported with specific arithmetic and string operators. Default rules definition is loaded at start up time from the HTTP server. The rules are passed to the agent as name value pairs.
Simplified rule syntax for the agent engine can be illustrated as follows:

RULE-n [*description*]
[SALIENCE *salience*]
IF
Interval.Property Operator State1 [*State2*]
[*Interval.Property Operator state1* [*State2*]]
THEN
set State Identifier <color code>
[*THEN generate alert for Operator*]
[*THEN DO Task x]*

Due to the nature of the monitored data and the agent, currently one rule will be fired per matched interval. The rule with higher salience will have high priority over the rule with lower salience. The agent is using by default depth strategy for conflict resolution. This means that newly activated rules are placed above all rules of the same salience.

Additionally, an exceptional conflict report entry will be available to the operator for conflicting rules.

4. Implementation

The International Data Centre (IDC) of the Preparatory Commission for the Comprehensive Nuclear-Test-Ban Treaty Organization (CTBTO) is receiving and processing, in near real time, data from all the International Monitoring System (IMS) stations or facilities via a dedicated Global Communications Infrastructure (GCI) [17]. Monitoring IMS data includes receiving, forwarding, automatic processing, interactive review, generation and distribution of IDC scientific bulletins and reports, and archiving of the data.

The proposed prototype agent has been implemented in monitoring the real-time processing of the data received continuously from the IMS stations, including meteorological data, or similar interval-based data. The monitored data are from different technologies and different characteristics IDC seismic data processing and pipeline, radionuclide data processing and atmospherics data processing. A fresh real implementation of the proposed system is shown in Fig. 4.

5. Comparison and Results

More than ten performance attributes have been identified to evaluate the proposed system. Also, a comparison between the proposed system and the current available monitoring tools has been made. Selected datasets from different records, extracted from IDC databases and different critical processes, have been used to evaluate the performance of the proposed system with the existing similar monitoring tools.

The results of the comparison between the proposed system and the other available monitoring tools are summarized in Table I. The response time of proposed knowledge-based monitoring agent is faster time than the current tools. Moreover, it is dynamic and simpler in configuration, integration and implementation.

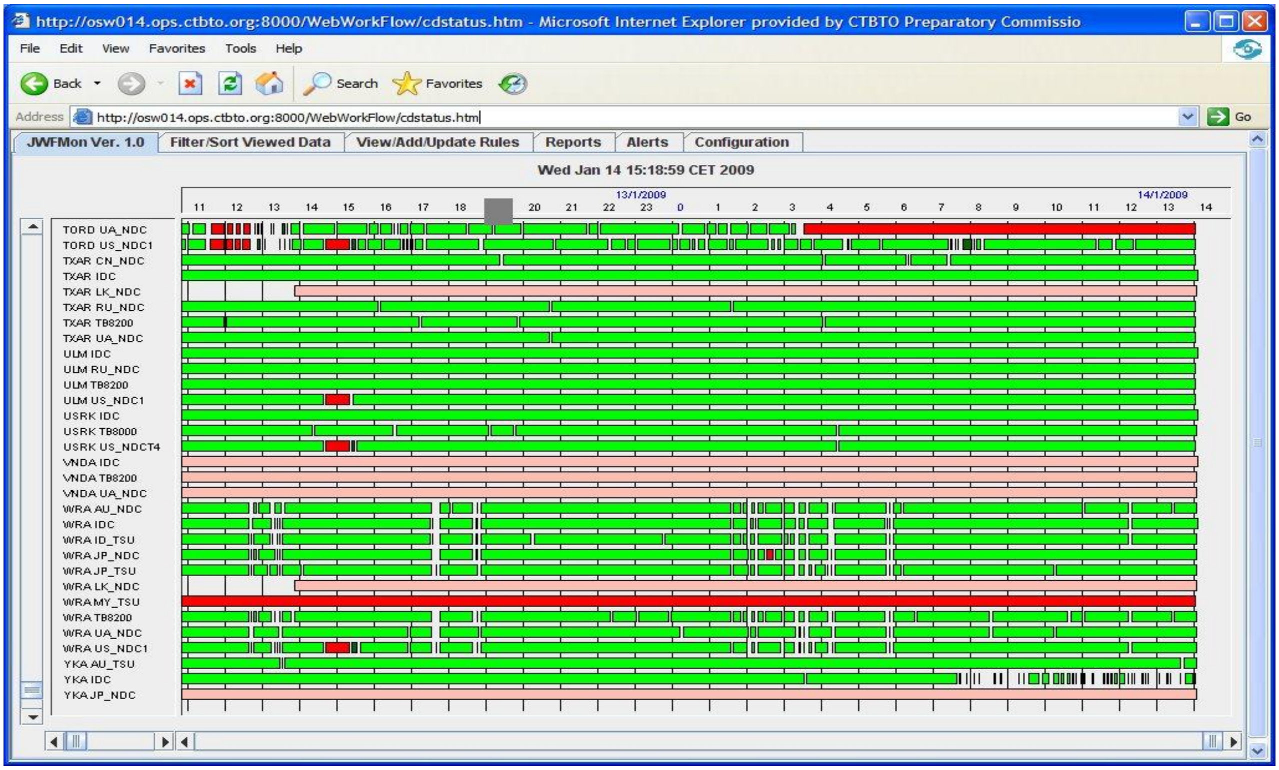


Fig. 4 Workflow of IDC continuous data distribution using proposed model

Table 1 Comparison results

No.	*Attribute*	Other Available Systems	Proposed Model
1	Response Time	Fast	Faster
2	Flexibility	Hard	Easy
3	Configurability	Simple	Simple
4	Portability	No	Yes
5	Dynamic Configuration	No	Yes
6	Web-Enabled	No	Yes
7	Task Specific	Yes	Generic
8	Scalability	Not Easy	Easy
9	Databases/ Resources Utilization (per monitored system)	N	1
10	Reports Capability	N/A	Yes
11	Alerts Capability	N/A	Yes

6. Conclusion and Future Work

The proposed lightweight knowledge-based agent is generic, portable, and flexible to handle any time interval, and platform independent. By implementing the proposed agent, status of the different objects, as well as their attributes or properties, can be easily monitored either via the organization Intranet, or remotely using Virtual Private Network (VPN) connections through the Internet. The agent allows timely generating exceptional reports and alerts. This will help the operators and decision making users to act faster on the anomalies that could be occurred during the workflow monitoring process.

Future work will be dedicated to create special monitoring agent, for simulation and analysis purposes, which fully use and implement the Java Rule Engine API (JSR 94) and its specifications. Also, it is planned to integrate or enable the use of the proposed monitoring agent in grid monitoring systems in order to share and distribute generated alerts and exceptional reports with the other intranet agents.

References

[1] A. M. Shaw, S. Yada, “A comprehensive agent-based architecture for intelligent information retrieval in a distributed heterogeneous environment”, Decision Support Systems, vol. 32, 2002, pp. 401-415.
[2] R. Jennings, “An agent-based approach for building complex software systems”, Communications of the ACM, Vol. 44, No. 4, 2001,pp. 35-41.
[3] M. Klusch, “Information agent technology for the internet: a Survey”, Data & Knowledge Engineering, vol. 36, 2001, 2001.
[4] J.-J. Jeng, J. Schiefer, and H. Chang, "An agent-based architecture for analyzing business processes of real-time enterprises," Proceedings of the IEEE 7th International Conference on Enterprise Distributed Object Computing, pp. 86, 2003.
[5] I. Seilonen, T. Pirttioja, A. Halme, K. Koskinen, and A. Pakonen, ,”Indirect process monitoring with constraint handling agents”, 4th International IEEE Conference on Industrial Informatics, Aug. 2006 , pp.1323 – 1328.
[6] F. J. Kurfess and D. P. Shah, "Monitoring distributed process with intelligent agents," In: Proceedings of IEEE Conference and Workshop on Engineering of Computer-Based Systems, 1998.
[7] L. Bunch, M. Breedy, J. M. Bradshaw, M. Carvalho, N. Suri, A. Uszok, J. Hansen, M. Pechoucek, V.Marik, “Software agents for process monitoring and notification”, ACM Symposium.
[8] Extensible Markup Language, http://www.xml.org/
[9] T. Bray et al., Extensible Markup Language (XML): W3C (World-Wide Web Consortium), http://www.w3c.org/.
[10] The Internet Society (ISOC), http://www.isoc.org/.
[11] Request For Comments (RFC) Editor, http://www.rfc-editor.org/.
[12] Java virtual machine, http://java.sun.com/
[13] J.E. Smith and R.Nair., “The architecture of virtual machines”, IEEE Computer, Volume 38(Issue 5):Pages 32–38, May 2005.
[14] S. Orlando, and S. Russo, “Java virtual machine for dependability benchmarking”, Proceedings of the Ninth IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC'06),2006,pp. 433-440.
[15] Bradshaw, J.M., Software agents. San Francisco, CA, USA: AAAI Press/MIT Press, 1997.
[16] S. Russell and P. Norvig, Artificial Intelligence, A Modern Approach, Prentice Hall, 1995.
[17] Preparatory Commission for the Comprehensive Nuclear-Test-Ban Treaty (CTBT) Organization, http://www.ctbto.org/.