An Ontology-based Architecture for Event Detection and Prediction on Monitoring Systems of Wireless Sensor Networks

■ Ricardo González
email: rgonzalez@ldc.usb.ve

■ María-Esther Vidal
email: mvidal@ldc.usb.ve

■ Claudia J. Barenco Abbas,
email: barenco@ldc.usb.ve

Departamento de Computación
Universidad Simón Bolívar, Valle de Sartenejas, Baruta.
Caracas, Venezuela, 89000

Fecha de Recepción: 8 de octubre de 2008
Fecha de Aceptación: 21 de enero de 2009

Abstract

In large-scale monitoring systems, integration of detailed data is required to detect and predict the occurrence of events that impact the performance of the whole environment. In this paper, we present an ontology-based multi-level architecture called HAEDEP, for event detection and event prediction on large-scale monitoring systems based on Wireless Sensors Networks. In HAEDEP, an ontology is used to represent domain knowledge that can be used at different levels, and enable the system to detect and predict the events that will affect the system behavior. We illustrate the benefits of HAEDEP by using a case study in the oil refinery field. However, HAEDEP could be used in other domains where the checking, recording and controlling activities are required.

Keywords: WSN, monitoring system, event oriented, domain knowledge, ontologies.

Resumen

En sistemas de monitoreo de gran escala, la integración y el análisis de datos sencillos es un requerimiento importante, ya que permite detectar y predecir la ocurrencia de eventos relevantes en el sistema. En este artículo se propone la arquitectura HAEDEP, que consta de varios niveles y que emplea una ontología para representar el conocimiento del dominio. Este conocimiento es usado a su vez en la detección y predicción de estos eventos relevantes, específicamente en un sistema que emplea redes inalámbricas

de sensores para capturar información del mundo físico. Este trabajo ilustra los beneficios de usar una arquitectura como la de HAEDEP mediante un caso de estudio asociado al campo de la refinación de petróleo. HAEDEP también puede ser usado en otros dominios donde actividades de verificación, almacenamiento y control se requieran.

Palabras Clave: Redes Inalámbricas de Sensores, monitorización de sistemas, sistemas orientados a eventos, conocimiento del dominio, ontologías.

1 Introduction

There have been developed a great number of applications using Wireless Sensors Networks (WSNs) on different fields [1] [2] [3]. However, it is unlikely the usage of monitoring systems that combine domain-knowledge bases and information collected by WSNs, to derive, predict, and control the monitored systems future behavior.

In large-scale monitoring systems, the integration of individual measurements that describe the behavior of its components can play an important role in the detection and prediction of changes in the system behavior. In industrial environments and ambient systems, a small variance in individual measurements of a parameter can be the preamble for further significant changes in the whole system performance.

A great variety of strategies can be used to detect events that may affect a system performance. Starting from detecting when a value is out of normal range to more sophisticated reasoning-based strategies where domain knowledge is used, a variety of techniques can be used to analyze and interpret raw collected data and discovery the relevant events and their side effects.

We propose a distributed architecture called HAEDEP (Hierarchical Architecture for Event Detection and Event Prediction), that combines WSNs gathering systems, with some tools to manage domain knowledge. In HAEDEP, an ontology is used to represent each WSNs system properties and their reactive behavior. Thus, effects of relevant events on the performance of the monitored system(s) can be modeled; also, knowledge encoded in the ontology can be used to infer simple facts or symptoms to develop a detection of relevant events in the system(s).

The remainder of this paper is organized as follows. In section 2, we describe a scenario that requires the usage of domain knowledge in order to identify some relevant events generated from data collected by WSNs. In section 3, we present our proposal for a monitoring architecture and a detailed overview of each of its components. Section 4 presents some strategies for event detection, and for generating new knowledge based on the analysis of the data that describes the system behavior. In section 5, we describe how the architecture components can interoperate through current trends on distributed system integration. Finally, we point our conclusions and future work in section 6.

2 Monitoring to Detect the Occurrence of System´s Relevant Events

The early detection of some out of range values in monitoring activities can be enough to identify undesired behaviors and establish some correctives. For example, a system can measure some of its performance variables and gather its associated values into a supervisor station, to identify simple relevant events and to solve some of the situations that may be fired.

If we could find a tool that allows us to read and write some variables that show the state and the performance of the system elements, it could seem to be enough to achieve the monitoring objectives. In the computer network domain, there is one protocol for this purpose, the Simple Network Management Protocol [4] (SNMP), which is essentially a specialized request/reply protocol that supports some request messages: GET, TRAP and SET [5]. These commands are used to retrieve the value of some metrics, that show the behavior of a network node, and that are defined in a Management Information Base (MIB). The SNMP uses agents that can capture and sent data from network nodes to a Manager Station, which gathers information about the whole system, and that allows this information to be available for users through a Human Machine Interface.

However, capabilities of SNMP are not always enough because, sometimes the detection of some relevant events requires the usage of domain knowledge about the system under examination, and reasoning processes to infer conditions that can fire new events. This is the reason that motivates us to propose a new monitoring architecture instead of using available SNMP implementations.

In order to illustrate the benefits of the proposed architecture, we describe a case study in the context

of a system that detects emissions on a refinery or a petrochemical plant.

A refinery is a big industrial plant that performs different processes to transform crude oil in diverse valuable oil derivates. Due to the chemical characteristics of the input commodities and the nature of the involved chemical process, theses processes generate air pollution emissions of undesired products components, such as: sulphur dioxide (SO_2), nitrogen oxides (NO_x), carbon monoxide (CO), particulate matter, volatile organic compounds, benzene, etc [6]. Due to these potential dangerous emissions, refineries normally are located at countryside and far from large cities.

There exist some rules that regulate the maximum amount of emissions that could be tolerated in a refinery facility. Because of this, there are a lot of commercial sensors to measure these emissions levels. Although air emissions can be measured by sensors on chimneys, emissions of a refinery may be produced by a variety of chimneys associated with different processes, and managed by different companies. In consequence, even though one sensor on a chimney can detect that a particular process exceeds some threshold value, there could be some cases where no regulation is violated because these out-of-range values are compensated by the decrease of emissions from other closed process. Therefore, total emissions in a locality depend on the sum of every emission source in the same place.

On the one hand, when a regulation is violated or some emissions exceed its maximum values, some actions must be taken. However, these actions depend on the values assigned to some of the properties that characterize the system. For example, if a violation is periodical or constant, meteorological data and particularly the wind speed could cause that a plume of emissions could reach a large population. In this case some questions have to be answered to establish the correct actions to be taken: Which values of the emissions concentration could reach when the emissions pass a particular place? What happen if in that place there is a population?, should this population be evacuated? Answers to these questions require the representation and control of the reactive behavior of the concepts and parameters that characterize the universe of discourse, as well as, the events that affect each concept behavior.

The problem of representing and manipulating reactive behavior, have been considered in databases systems during the last decades. First, active databases were proposed as a rule-based framework to model relationships between concepts and the events for which they react [7]. Similarly, in the context of the Semantic Web, XML and OWL[8] have been enhanced with rules or operators to represent reactivity. In this paper, we will illustrate the effects of representing the reactive behavior on the quality of the tasks of monitoring. We will use the ontology-based framework ACTION [9]. In this framework, events are categorized as concepts of an ontology and, in conjunction with classes, properties and instances, are considered during the query answering and reasoning tasks. We have chosen ACTION, because this formalism provides a more expressive solution to the problem of representing and querying active knowledge than existing rule-based approaches.

3 HAEDEP: A Monitoring Architecture

Inspired on the study of monitoring oil production activities [10], and following the MVC Model View Controller Paradigm [11], we propose a four-level architecture presented in Figure 1.

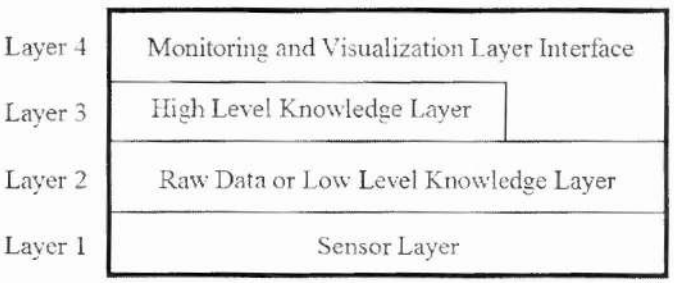

Fig 1. Hierarchical View of the HAEDEP Monitoring Layered Architecture

In the HAEDEP architecture, the Sensor layer is responsible for collecting system information from the real world; the Low Level Knowledge Layer stores and manages data and metadata collected from the Sensor Layer; the High Level Knowledge Layer uses domain knowledge to identify and store relevant events through different techniques; and finally, the Visualization Layer offers an interface that allows system operators to access information that describes relevant events of the monitored system. The HAEDEP architecture and the systems that compose it can be published as Web Services. Thus, HAEDEP and its components can be described by using the Semantic Web formalisms, and

their functionalities can be enhanced by combing them with other published Web services.

A general description of the elements that comprise each HAEDEP layer is illustrated in Figure 2 and detailed in the next subsection.

The usage of an ACTION ontology makes a deployed WSN capable to dynamically adapt itself to environmental conditions and changes. Also, this ontology is able to represent the properties of the network application that recognizes patterns in order to understand the properties of the collected raw data. This information is gathered from low levels layers, and it is used to distinguish the behavior of the parameters that indicate if a system event can be fired [12].

3.1 The Sensor Layer

The Sensor Layer includes any physical system or field devices that need to be sensed or monitored. These devices collect and transfer raw data to a central or distributed data storage.

3.1.1 The Physical Systems

The physical systems refer to any system components that are been monitored, e.g., industrial equipments, warehouse, pipe, valves, chimney emissions, etc. Each system is sensed with specific sensors that measure operational parameters, adapted to its inherent process, which may reflect the system reactive behavior.

3.1.2 The Sensor Networks

To collect data of physical system activities, we propose the usage of wireless sensor networks (WSNs). In contrast to traditional wired sensor information collection platforms. These networks have the following advantages:

- Low cost of components.
- Flexibility in placing sensor nodes.
- Self powering using traditional batteries.
- Reduction on deployment time.
- Nodes with capacity of local processing near of measured process.

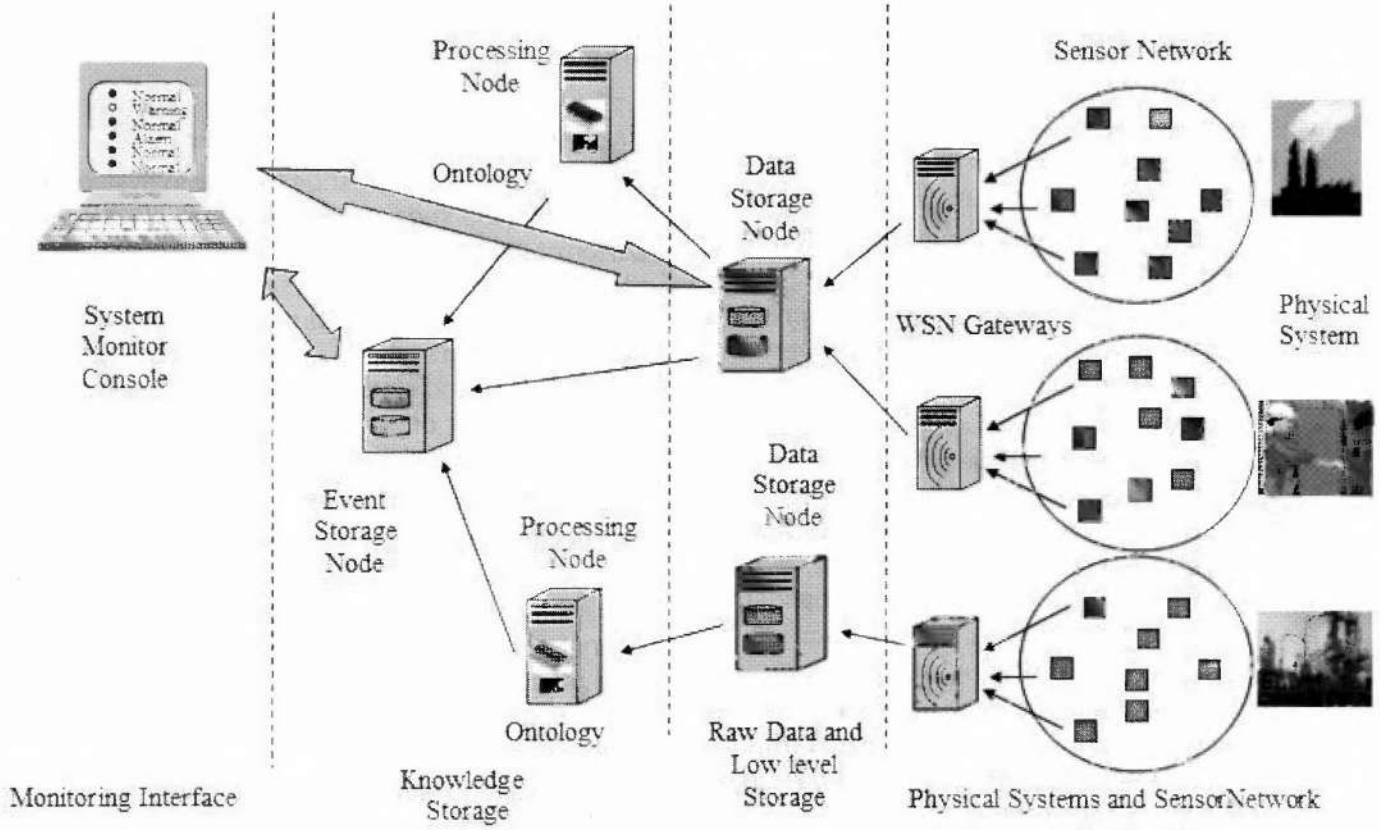

Fig 2. The HAEDEP Architecture and its components

A wireless sensor network is a set of tiny sensor nodes, called motes [13], and one or more base stations that gather data from sensor nodes. Each sensor node or mote is a tiny computer composed of a processing unit, a memory, sensors, a wireless networking device, and a power supply [14]. In Sensor Networks, a base station is a component with higher processing capabilities, less limitation of consuming power, and more complex communication devices. A Base Station, or Sink, plays the role of a gateway between sensor nodes and others communication architecture components outside the WSN.

Some additional characteristics of a WSN are as follows [15]:

- A very large range of nodes can be used in one network; this number ranges from a couple of motes to thousands of them.
- There are generally asymmetric flows of information, from the sensor nodes to a base station.
- Communications could be triggered by queries or events.
- At each node there is a limited amount of energy which in many applications is difficult to replace or recharge.
- Motes are characterized by its low cost, small size, and little weight.
- Motes could get profit of using broadcast instead of point-to-point communications.
- Nodes do not normally have a global ID such as an IP address.
- Security, both physical and at the communication level, is more limited than in conventional wireless networks.
- In a WSN node, if certain event is fired, the Sensor layer can send an especial warning message to the Raw Data or Low Level Knowledge layer. In extreme situations, some control actions could be fired or executed by motes, in order to get a faster response and to avoid dangerous and undesired situations.
- Information collected by the Sensor layer could be used on other layers to infer the causes of unusual behaviors, or to build prediction models able to identify the conditions that trigger these unexpected behaviors.

3.2 Raw Data or Low Level Knowledge Layer

This layer stores unprocessed data collected from the physical system that could be used for the above layers to correlate individual parameter values, as an indication of a complex or unusual behavior. An external storage can be useful, due to limitations on WSN nodes design, with small memories that could store a very small amount of data, or it could lose collected data if its buffers get full or if a node runs out of battery.

3.2.1 Data Storage Node

The function of a Data Storage Node is to store and manage data gather by the Sensor Network, making them available to any other architecture's component or final users at any time. Data Storage node could also be used as a mediator in order to query current or past monitoring values of the system parameters. They have to store not only raw collected data but also Metadata that describe the real-system conditions, in order to establish when a data set could be compared with other similar data sets, and to help in analysis of system behavior trends. In relation to the system that detects emission on refinery, described in section 2. Some measurement of sensor on a chimney should be stored as a Single Event with some properties such as chemical components, value of concentration, time of the measurement (time stamp), sensor location, etc.

3.3 High Level Knowledge Layer

In this layer, domain knowledge and system behavior are represented and used to detect relevant system performance facts, tendencies and the events that fire them. These events could be stored and used, as high-level events, for operators during an analysis of what is happening in the system. This layer is also able to infer mathematical and statistical models that describe the regular behavior of the system, and that can be used in tests and forecast about future system trends.

3.3.1 Processing Nodes

These nodes can submit queries to the Data Storage Nodes in order to discover evidence of specific events occurred in the system. They can perform different kind of analysis, which varies from detecting a trespassing of a particular parameter threshold, to modeling and predicting activities based on historical data, which are saved on storage elements.

Complex models could be used together with ACTION [9] to represent the properties that characterize raw data. Knowledge encoded in the ontology and in the complex models can be used to derivate the occu-

rrence of relevant events, and to determine the actions or changes in the data that have to be performed when relevant events are detected.

In the case of system that detects emissions on a refinery, we can describe an ACTION ontology as a tuple Oa =< C, E, Ps, Pa, F, fr, I > [7], where:

- C: a set of classes or basic data types (Emission, Emission_SO2, Emission_NO2, Emission_CO, Regulation , Regulation_SO2, Regulation_NO2, Regulation_ CO, etc.)
- E: a set of events (Critical_Emision, Emission_ Alarm, and Evacuation_Alarm.)
- Ps: a set of static properties, where each property corresponds to a function from C U E to C U E. (Emission_level, Collection_Time, Min_Value, Max_Value, place, Dispersion_Factor)
- Pa: a set of active properties; each property corresponds to a function from C to C. (Qualitative_Event, Action).
- F: a set of predicates representing instances of the classes, properties and events. (type, individual).
- fr a function, s.t., fr : F ×Pa×E □ F; fr defines the reactive behavior in Oa.
- The following rules establish the conditions that need to be satisfied to fire an event, and also the changes that will be performed to the active property values:
- ((EmNO2235, emission_level,4) , (Cualitive_Level event(Local_Total_Emission, NO2_level ≥ 3), (Regulation_NO2, 3)

  ⟶ (Qualitative_level, Dangerous)
- (action, event(Qualitative_Level, Dangerous),

  (Wind_Direction=Population_Direction)

  (Emision_Level*(Population_Distance/Wind_ Speed)*Emission_Dispersion_Factor >= Regulation_Max_Nalue)

  (Regulation_Place, Population))

  ⟶ (EmNO2235,Action, Evacuation)

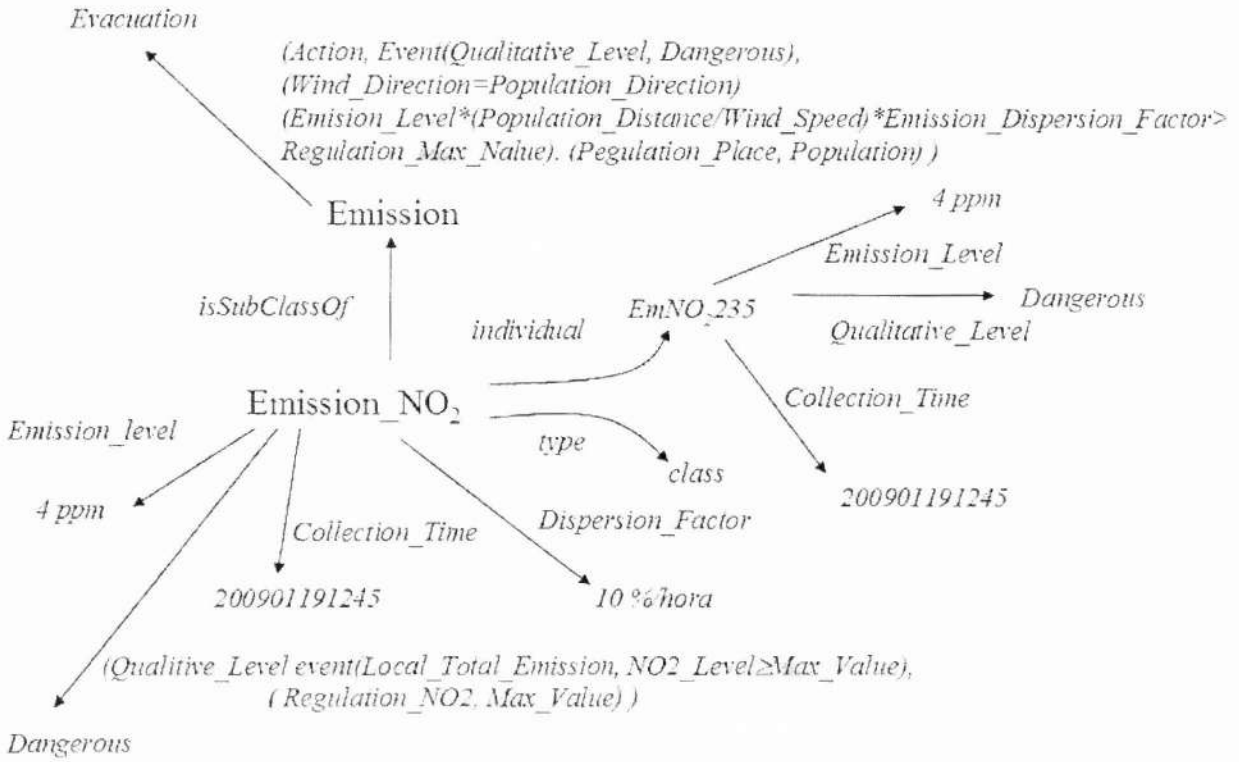

Fig 3. Representation of Knowledge associated with Emissions Domain

The set I is comprises of a set of axioms that describe the properties of the built-in properties provided in Ps and Pa. Particularly, the following axiom is used to infer and fire new events:

- if isSubEventOf(e2, e1) and isSubEventOf(e3, e2) then isSubEventOf(e3, e1)

Figures 3, 4, 5, 6 y 7 illustrate a graphical representation of the ACTION ontology, and static knowledge that characterizes a system which detects emissions on a refinery. This ontology is partially inspired in the SNS Environmental Vocabulary [16]

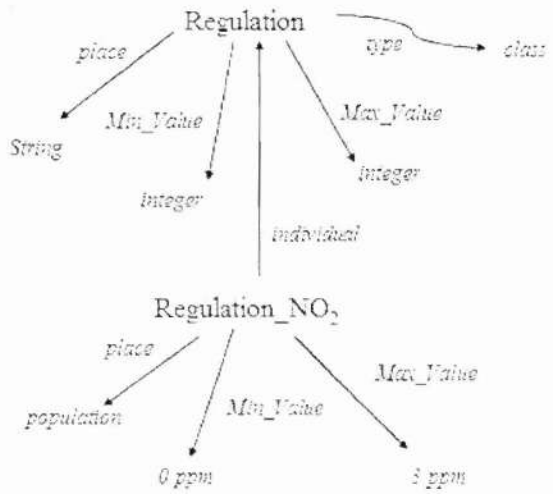

Fig 4. Representation of Knowledge associated with Regulation of Emissions Domain

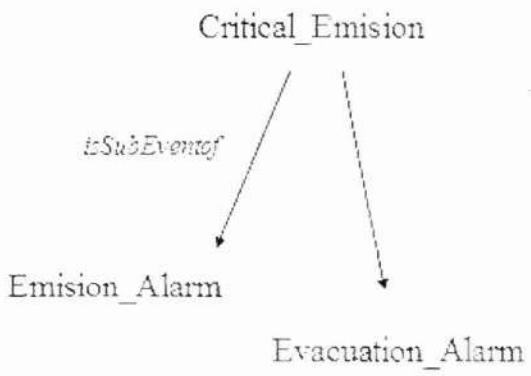

Fig 5. Representation of Critical Events on Emission Domain

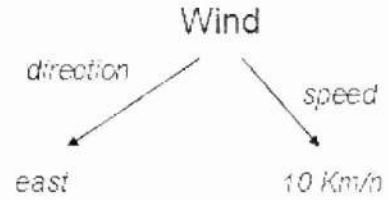

Fig 6. Representation of Population Features Associated with Emission Domain

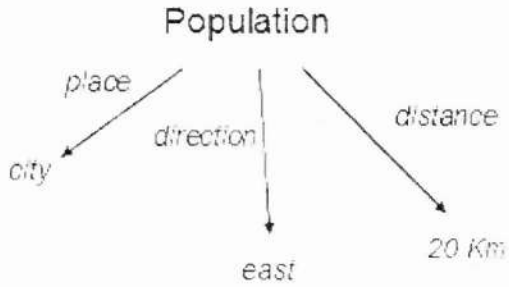

Fig 7. Representation of Population Features Associated with Emission Domain

3.3.2 Event Storage Node

This component stores and manages the whole family of new high-level events, which are generated by processing nodes that search clues and evidences of specific behaviors about system performance.

3.4 Monitoring and Visualization Layer

This layer is the window that shows the system behavior to operator, supervisors, manager and any other high level users.

3.4.1 System Monitor Console

A Monitoring Console is a visualization component, that will be used as a system interface to detected and analyze event occurrences. It also will assist users in explaining behavioral symptoms, establishing cause-effect relationships of system current parameter values.

4 Strategies for Relevant Event Detection on System Data

The main objective of a monitoring system is to extract information about relevant events that are fired in the monitored system. In this work we propose a combination of different event detection techniques that could be used by the processing nodes, in order to extract as much as possible relevant information from the collected data.

We can classify event detection techniques in the following four groups: simple detection, multiple detections, breaking its own trend, and breaking similar trends.

Simple detection and *multiple detection* strategies are common in system monitoring activities and can be applied by comparing online sensing information with predefined expected values on measured parameters. On the other hand, *breaking its own trend* and *breaking similar trends* are less common, and involve some sort of comparison of current and historical information.

4.1 Breaking Threshold Strategy on Event Detection

This kind of analysis implies that a collected parameter violates a value by crossing a boundary. The strategy can be simple or multiple depending on the type of the parameters detected.

4.1.1 Simple Detection

Some important events can be recognized once a simple parameter crosses a specific threshold. In this kind of analysis, original sensed information will be compared with predefined upper or lower bound value on measured parameters. This kind of detection analysis, also known as Threshold rules [17], is one of simplest detection strategies, and is able to trigger a warning on Monitoring Interface as soon as an upper or lower value is reached by a system measure.

4.1.2 Multiple Detections

Some events could not be identified as a direct consequence of an isolate parameter value, but they could be recognized when two or more parameters cross some specific thresholds.

In this kind of analysis different original sensed information will be compared with some specific values to detect some symptoms on a multivariable space. Using sets of comparisons, this method could detect more complex behavior than the Simple Detection does. For example, if there is a region with scare coverage, (small amount on sensing node with low redundancy) and some low level node battery event were reported, a warning of near disconnection could be triggered to notify the user about this threat. Any of these prior facts itself, is not a clear evidence of the conclusion, but it simultaneous detection does. The knowledge about domain of study can make a good usage of this technique to identify different parameter values as symptom of a high level event that is taking place on the system.

4.1.3 Beyond a Simple Detection

Threshold rules can be used to translate quantitative data to qualitative information, classifying parameter values in level as "high" and "low", or "normal" and "dangerous" values. This kind of conditions have to be satisfied in order to fire some high level events, they can be represented in an ACTION ontology. For example, in a system that detects if the emissions produced by a refinery violate local regulations or environmental conditions, it can be represented by saying that the status of pollution produced by the refinery varies depending on the concentration of the contaminant emissions and on the location where the refinery is placed. Thus, ACTION can represent that the active property *Qualitative_Level* reacts to the values of NO2 emissions and *Max_Value* of Regulation by using an event *e*. This event fires the association of a coast-located refinery with the value *Dangerous* using the property *Qualitative_Level*, when the active *property Local_Total_Emission of NO2* reaches a mean value of 3 ppm.

4.2 Changes on System Behavior

An event could be recognized if the system does not follow its expected value. When a model of a system behavior is built, comparison of predicted and collected values can be used to detect the occurrences of some hidden events.

4.2.1 Detecting Changes on a Normal Behavior of a Parameter:

Some event occurrences could be detected when a system parameter breaks with its own value trends. Trends can be determined from a set of historical data of some relevant parameters. As soon as some deviation of its trends is detected, this situation can generate some kind of warning, in order to alert about possible effects of this change. Some attention to variance or tolerance has to be considered because it is difficult to get always the same values on measurements of the real systems.

4.2.2 Detecting Changes on the Whole System Normal Behavior:

Some events could be recognized if the system does not follow the behavior of other systems that show reasonable common characteristics. Once a system is instrumented and its data collected, a comparison with the past can be made. However, other possibility is to use a model, built with data that comes from other system(s), which could have many common characteristics, and that could be used as a pattern of comparison. In this case models can be used to predict future behavior of some values and to perform activities of capacity planning. These models can be generated by modeling techniques such as: time series analysis, queuing or simulation models, where metrics can be estimated based on other similar system data. If collected system values do not match model values, this could be interpreted as an indication of a change in its global behavior; and, this change may need to be analyzed by the system operator in other to defined strategies that ensure the system operability.

4.2.3 Additional Features on Using System Behavior Values

Once a model of system behavior is built, it can be used to verify that current sensed data respect a normal behavior. But in some cases there is a gap in the collected system information. Due to problems or partial failures on sensing and transmitting infrastructure, a user can find some time period without collected data. In order to consider reasonable values during these periods, we propose to use interpolation methods in order to fill up gaps in the collected data. This technique will facilitate the inference of a more precisely idea of which would be the system behavior on similar circumstances, even in none fully reliable systems.

5 Protocols for Integration of Architecture Components

We propose the use of a WSN as an entry point of data collected from real systems, and to use it as first data sources of a whole system. This data can be used to detect and notify to system operators the occurrence of specific events on the system as it was describes in [18]. We also proposed the use of Web Services [19] [20] [21] as a mechanism to unify the communication between each system component even on WSNs. The communications of the system components that is show in Figure 2, from layer 1 to layer 2, such as is being proposed on Sensor Web [22]; from layer 2 to layer 3, and also from layer 3 to layer 4, could be used by means of Web Services, which will generate a common reference mark to collect and transmit information between these components. This strategy has been also explored previously for connecting WSN to Grids environment [23][24]. With this integration, we can combine different solutions on each component of the architecture through a unique syntax and semantic. Additionally, our proposal includes components to analyze data collected by other Web Services. This interaction will facilitate the discovery of relevant facts hidden in the raw data. It is important to notice that this task would be almost impossible to perform if knowledge encoded in the ontology were not available. The use of Web Services could help to integrate new nodes at any time, in order to improve the functionalities of the discovery of relevant system event in the proposed architecture.

In presence of limited capabilities devices, wrappers and proxies can be developed to encapsulate real world programs that provide these facilities on top of traditional and less complex request-response WSNs platforms. The main goal is that each data source will offer a common API that any programmer can easily understand and work with it, in order to facilitate the developing of new services based on previously available ones.

6 Conclusions and Future Work

WSNs have had a great impact on current information management. The usage of networks of many motes (tiny and cheap devices) to monitor physical systems will become a reality, pervasive and ubiquity computing. For this subject we propose an architecture that uses WSN data sources as a component of a whole information system, providing an integration of monitors, data storage platforms, and knowledge management components to support operator decision processes, while supporting them in the detection of relevant events.

Web services can be used as common interfaces between different system components. This enables the creation of more powerful Web services, resulting from the orchestration of the HAEDEP architecture service and its components, with other published Web services.

The integration of active knowledge with detailed information in data sources enable operators to recognize, and a wide range of relevant events, ranging from

a simple metric that crosses a specific threshold to none trivial, high level events, which could use domain knowledge and inference about different parameters values to be identified.

The main contribution of this paper is an architecture that orchestrates different components and strategies, and represents the properties of the data measured by monitored systems in a knowledge base. Reasoning techniques are implemented, to support the detection and prediction of relevant events. Finally, the usage of Web services allows a unified communication mechanism between the architecture components.

References

[1] Karl, H., Willing, A.: *Protocols and Architectures for Wireless Sensor Network*. John Wiley & Sons. West Sussex, England (2006)

[2] Kuorilehto, M., Hannikainen, M., Hamalainen, T. D. (2005) *A Survey of Application Distribution in Wireless Sensor Networks*. EURASIP Journal on Wireless Communications and Networking. vol 5, pp 774--788

[3] Romer, K., Mattern, F. (2004) *The design space of wireless sensor networks*. IEEE Wireless Communications. vol 11. num 6 pp 54--61

[4] Case, J., Fedor, M., Schoffstall, M., Davin, J. (1990) *A Simple Network Management Protocol (SNMP) RFC 1157*. http://www.ietf.org/rfc/rfc1157.txt Consulted January 6th of 2009

[5] Larry L Peterson, L. L. and Davies B. S.: Computer Networks: A System Approach. Morgan Kaufmann Publishers. 3th Edition. 2003. pp 657-658.

[6] Canadian Council of Ministers of the Environment. (2005) *Emission Monitoring and Reporting Strategy – Summary and Background Framework for Petroleum Refinery Emission Reductions* (NFPRER). http://www.ccme.ca/assets/pdf/e_m_strategy_summary_final_eng.pdf Consulted January 9th of 2009

[7] Goldin D., Srinivasa S., Srikanti, V. (2004) *Active Databases as Information Systems*, Proceedings of the International Database Engineering and Applications Symposium (IDEAS'04), p.123-130, 07-09 July, (2004)

[8] W3C. (2004) *OWL Web Ontology Language Overview*. http://www.w3.org/TR/owl-features. Consulted January 10th of 2009

[9] Tovar E. L. and Vidal M. E. (2008) *Magic Rewritings for Efficient Processing Reactivity on Web Ontologies*. Lecture Notes in Computer Science. Vol. 11, pp. 1 – 18

[10] Medizade M., Ridgely J. R., Nelson D. (2004) *Marginal Expense Oil Well Wireless Surveillance MEOWS - Phase II* Final Technical Report. Petrolects, LLC and Vaquero Energ. November.

[11] Krasner G.E., Pope S.T. (1988) *A cookbook for using the model view controller user interface paradigm in Smalltalk-80*. Journal of Object-Oriented Programming, vol 1 num 3. pp 26-- 49

[12] Liu, J., Zhao, F. (2005) *Towards Semantic Services for Sensor-Rich Information Systems*. In Proceedings of the 2nd IEEE/CreateNet International Workshop on Broadband Advanced Sensor Networks, Basenets , Boston, MA, Oct. 3. pp 44--51

[13] Intel (2004) *Instrumenting the World An Introduction to Wireless Sensor Networks. Version 1*, http://www.intel.com/research/print/overview_instrument_world.pdf

[14] Minami, M., Saruwatari, S., Kashima, T., Morito, T., Morikawa, H., Aoyama, T. (2004) *Implementation-based approach for designing practical sensor network systems*. In: Proceedings of the 11th Asia-Pacific Software Engineering Conference (APSEC'04) vol 0. 30 Nov-3 Dec. pp 703- 710

[15] Akyildiz I. F., Su W., Sankarasubramaniam Y., Cayirci E. (2002) *Wireless Sensor Networks: A Survey. Computer Networks*, vol 38 num 4. pp 393—422.

[16] Maria Rüther M., Bandholtz T,. Menger M. (2006) *SNS Environmental Vocabulary – from Terms to Ontology*. http://www.semantic-network.de/2006-semantics-TB.pdf. Consulted January 9th of 2009

[17] Merrett, G. V., Harris, N. R., Al-Hashimi, B. M. and White N. M. (2006) *Energy Controlled Reporting for Industrial Monitoring Wireless Sensor Networks*. IEEE Sensors. 22nd-25th October. Daegu, Korea. pp 892--895

[18] Franklin M., Jeffery S., Krishnamurthy S., Reiss F., Rizvi S., Wu E., Cooper O., Edakkunni A., Hong, W. (2005) *Design Considerations for High Fan-in Systems: The HiFi Approach*. CIDR Conference, January, 2005, Asilomar, CA. http://www.cs.berkeley.edu/~franklin/Papers/hifiCIDR05.pdf

[19] Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., Weerawarana, S. (2002) *Unraveling the Web services Web: An introduction to SOAP, WSDL, and UDDI*. IEEE Internet Computing vol 6 num 2. pp 83--93.

[20] Curbera, F., Khalaf R., Mukhi N., Tai S., Weerawarana S. (2003) *The Next Step in Web Services Communication* of the ACM vol 46 num 10. pp 29--34

[21] Radenkovic M. and Wietrzyk B. *Life Science Grid Middleware in a More Dynamic Environment*. In (2005) R. Meersman et al. Editors. OTM Workshops 2005. LNCS vol 3762. pp 264 – 273. Springer-Verlag Berlin Heidelberg

[22] Chu, X.; Buyya, R. (2007) *Service Oriented Sensor Web*. In: Mahalik, N. P. (ed), Sensor Network and Configuration: Fundamentals, Standards, Platforms, and Applications. Springer-Verlag, Germany, January. pp.51--74

[23] Gaynor M., Welsh M., Moulton S., Rowan A., 3LaCombe E., Wynne J. (2004) *Integrating wireless sensor networks with the Grid*. IEEE Internet Computing, special Issue on Wireless Grids, pp 32--39.

[24] Lim H.B, Teo Y. M., Mukherjee P., Lam V. T., Wong W. F., See S. *Sensor Grid: Integration of Wireless Sensor Networks and the Grid*. The IEEE Conference on Local Computer Networks LCN 2005, 15-17 November. Sydney, Australia, (2005)