# Improving ESB Capabilities through Diagnosis Based on Bayesian Networks and Machine Learning

Roberto Koh-Dzul
Universidad Autónoma de Yucatán, Facultad de Matemáticas, Mérida, México
CNRS, LAAS, Toulouse, France
Email: jose.koh@fmat.uady.mx

Mariano Vargas-Santiago and Codé Diop and Ernesto Exposito
CNRS, LAAS, Toulouse, France
Email: {mvargass, cdiop, eexposit}@laas.fr

Francisco Moo-Mena and Jorge Gómez-Montalvo
Universidad Autónoma de Yucatán, Facultad de Matemáticas, Mérida, México
Email: {mmena, jgomez}@uady.mx

***Abstract*— The growing complexity and scale of systems implies challenges to include Autonomic Computing capabilities that help maintaining or improving the performance, availability and reliability of nowadays systems. In dynamic environments, the systems have to deal with changing conditions and requirements; thereby the autonomic features need a better technique to analyze and diagnose problems, and learn about the functioning conditions of the managed system. In the medical diagnostic area, the tests have included statistical and probabilistic models to aid and improve the results and select better medical treatments. We propose a probabilistic approach to implement an analysis process. The base of our approach is building a Bayesian network as model representing runtime properties of the Managed Element and their relationships. The Bayesian network is initially built from monitored data of an Enterprise Service Bus platform under different workload conditions, by means a structure learning algorithm. We aim to improve the functionalities of an Enterprise Service Bus platform integrating monitoring and fault diagnosis capabilities. A case study is presented to prove the effectiveness of our approach.**

***Index Terms*— Autonomic computing, bayesian network, probabilistic reasoning, diagnostic, machine learning, SOA**

## I. Introduction

The evolution of Internet-based applications has impacted the design and administrative task of current systems. Service Oriented Architecture (SOA) has been one of the most followed paradigms to design and built the distributed and heterogeneous systems. This approach is based in the service concept, involving service consumers that use features offered by service providers. Standard solutions have been proposed for the integration and the mediation of all these components that can be heterogeneous and disparate. Even so, distributed systems have weaknesses that could limit their advance.

In a SOA context where a large number of concurrent services can be provided and consumed, the competition for using this shared services and resources can lead to unpredictable events such as service unavailability, high response time, decrease of reliability, etc. Such anomalies need to be addressed by proposing efficient strategies able to guarantee or improve both the performance and the reliability offered. The complexity of these systems can require a lot of time, people and skills to be configured, managed and repaired; it becomes mandatory to avoid managing them manually since this becoming more difficult and expensive.

The Autonomic Computing initiative proposed by IBM aims to design and built systems capable of monitoring themselves, evaluating its current state, and applying changes to improve or recover its performance, reliability, security and availability properties [1]. This initiative introduces the Autonomic Elements, which consists of an Autonomic Manager and components named as Managed Elements. The Autonomic Manager is an entity that manages the Managed Elements. The Autonomic Manager is composed by a MAPE (Monitoring, Analysis, Planning and Execution) control loop with a shared knowledge base, where observations, analysis results and self-managing plans are contained.

The analysis phase takes observations from the monitoring, makes inferences and diagnoses the overall state of the managed system. The analysis results are used for planning the adaptive action in an autonomous way. The knowledge component in the Autonomic Manager must describe relevant aspects of the Managed Element to make precise analysis and diagnosis.

In this paper, we propose a diagnostic model based on probabilistic reasoning and Bayesian networks as

Manuscript received August 30, 2013; revised September 30, 2013; accepted November 25, 2013.


doi:10.4304/jsw.9.8.2206-2211

knowledge representation. Our model also integrates an algorithm that allows learning about the functioning of the Managed Element.

This paper is organized as follows. The section 2 presents related works on Autonomic Computing and probabilistic diagnostic. The section 3 describes the proposed approach to construct the diagnostic model based on a probabilistic model. Section 4 presents a case of study and experimental results of our diagnostic model. Conclusion and future work are discussed in section 5.

## II. Related Work

Many works have proposed solutions applying different strategies to implement the MAPE control loop phases, for various kinds of systems [2]. Some techniques to deal with the detection and diagnosis of problem are generally based on artificial intelligence and soft-computing principles [3] [4], such as: utility functions [5], active probing [6], advanced behavioral models, pattern recognition, decision trees, probabilistic models, probabilistic reasoning [6] [7] [8], and machine learning [9].

Rish et al. in [6] present an active probing algorithm for probabilistic diagnosis based on the probes selection. The algorithm allows selecting probes in order to minimize the set of monitored measurements, while maintaining high diagnostic accuracy. With the information sent from the probes, the system state is updated using probabilistic inference. Dependency matrix and Bayesian networks are used to represent how the probes are related, and determine which probes should be selected.

Dai et al. in [7] present a self-healing approach applying a Multivariable Decision Process (MDD) and a Naïve Bayes (NB) model to implement a two-step diagnostic process. The main goal of this work is to get a consequence oriented diagnostic, the first step is to assign a severity level to the observed symptoms using the MDD model, further the NB takes the symptoms and its classification results are used to infer the possible consequences. The diagnostic process helps to select the most adequate healing action.

In [9], a machine learning approach to construction of Bayesian networks from semantic models is proposed. The Bayesian network servers to monitor and learns about the effects of managing actions, improving decision making for the self-configuring property.

Unlike the related work, we propose a diagnosis process based on a probabilistic model that allows represent state of the Managed Element from its runtime parameters. The diagnosis can determines when a parameter value is indicating an issue. A Bayesian network helps to define the stochastic properties among the parameters, which is useful to know effects and causes of an issue, besides suggest the adaptive actions required.

### A. Probabilistic Reasoning

Probabilistic reasoning identifies relevant variables in a problem domain and builds a probabilistic model that represents the relationships among them. The variables can be in different states. The probability of each state is defined in the model, and can change depending on the relationships with other variables.

The probabilistic reasoning introduces observed evidence in some variables of the model, and computes the posterior probabilities of the remaining variables; this is the probabilities of the states conditioned to the evidence. The result is the interpretation of the most probable states, depending on their meaning in the domain.

Bayesian networks (BN) are a probabilistic graphical model that consists of a Directed Acyclic Graph (DAG) representing causal hypothesis, and sets of probability distributions over the set of variables $V$ [10] as Conditional Probability Tables (CPT).

The BN properties and the Bayes' theorem allow computing the influence of a variable over the other variables in a BN given evidence [11].

In a diagnostic model, evidences represent alerts or symptoms as states in the variables. The diagnosis is the interpretation of the most probable posterior states in the remaining variables, after to introduce the evidences. The diagnosis can show other variables in undesirable states.

Bayesian networks and the probabilistic reasoning are used widely, from troubleshooting and expert reasoning to machine learning in different areas, such as statistics, data mining, medical differential diagnosis, etc; mainly in making decisions and prediction [12].

### B. PC Algorithm

In Autonomic Computing, a learning process is important because the Managed Element could change over the time due to variations in its workload, in the user requirements, or in its environment. Learning capabilities can help to improve the analysis and the diagnostic results. Learning also could be used to assess and improve the effectiveness of adaption actions in a feedback loop of the Autonomic Manager.

In a BN, changes in the Managed Element may reflect modifications in the relations among the variables, and the CPTs. One approach to learn about the stochastic properties, and build a probabilistic model of a problem domain is based on tests of independence. The discovered model represents the underlying independence conditions among the variables.

The PC Algorithm is based in this approach to build the DAG associated to a BN. It performs tests for independence between pairs of variables, finding the edges and determining their directions in the DAG [13]. The CPTs are calculated for each variable applying statistics and the principle of the conditional probability.

The PC algorithm can be applied to any data sample and, under ideal conditions, it warrants discovering the most precise causal structure.

## III. Diagnostic Model Based on Probabilistic Reasoning

In a SOA context, element such as service consumer and service providers, are under dynamic and unpredictable conditions, and this aspect must be considered by the Autonomic Manager. We propose using a Bayesian network as knowledge representation of the Managed Ele-

ment, in order to use this model in probabilistic diagnostic and learning processes.

In order to offer more autonomous capabilities, our diagnostic module includes the PC algorithm as learning mechanism. This allows discovering of the BN related to initial data samples to start the autonomic behavior. Later, the algorithm updates the knowledge base periodically with data coming from the monitoring phase.

The variables and categories, or states, represent interesting runtime aspects in the Managed Element that describes its overall state. The variables in the BN are defined in the monitoring data achieving a low coupling between the Autonomic Manager and the Managed Element. Thus, to change the probabilistic model is not necessary to make changes to the Autonomic Manager.

The BNs represent relationships between variables, and allow making inferences about the overall state of the Managed Elements from evidences on some variables. For example, a high CPU load detected from the Monitoring would suggest instability in the system.

BNs can represent the Managed Element from different perspectives, depending on the variables included. For example, to give a diagnosis based on QoS, parameters such as Response Time or Throughput will be part of the variables in the model.

Three main parts of our proposed diagnostic model are: Data pre-processing module, Learning module and Querier Module. The Fig. 1 shows a schematic representation of the diagnostic process.

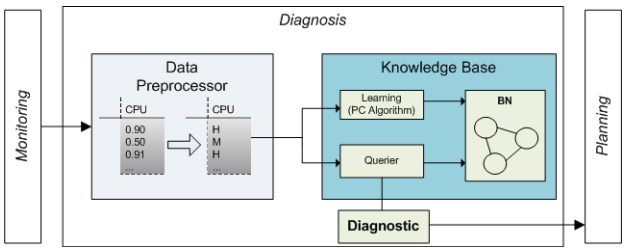


Figure 1. Diagnostic Process

### A. Data Pre-processing Module

The monitoring part can observe continuous or discrete parameters in the Managed Element. We consider that discrete values are better to represents qualitative criteria for the system properties. So, the continuous values from monitoring are mapped to discrete values, which are states for the variables in the knowledge base.

The pre-processing module is responsible for carrying out the first task of our diagnostic process, the values mapping. For this, each variable related to a parameter has states defined as numeric intervals that cover all possible values. The states of the variables must give proper meaning to intervals of values. With more states the knowledge representation can be more precise, but making inferences could be very complex.

All data sent from the monitoring phase are classified to states for each variable through interval matching. By this way, a value *v* for the variable *A*, represents the variable *A* in the state *a*, if the interval of the state *a* includes the value *v*. The intervals must be mutually exclusive.

The pre-processed data are sent to the Querier Module for analysis, and stored in a monitoring database for later use by the Learning Module.

### B. Querier Module

The Querier module inserts the data received from the previous module in the BN as evidence, and gets the most probable states in the variables through the BN properties. The diagnostic result is the set of the most probable states of all variables, representing the general conditions of the managed system.

The Planning phase takes the diagnosis and, based on policies, determines the necessary actions. If the policies indicate that all the variables are in correct states, then no action is applied in the system. Otherwise, the planned actions will aim to restore normal states of the variables, maintaining the normal functioning of the system.

At runtime it can be hard to monitor and get all information about the variables defined in the BN, because it implies using more resources. With BNs it is possible to choose part of data related to some variables taking from the monitoring phases, the data are passed to the Querier to infer the states of other variables and detect undesirable states. It means less cost for monitoring.

### C. Learning Module

The learning module is the responsible for building the initial knowledge base and keeping it updated during the execution time. This module uses the PC algorithm to discover the structure of the BN from a data sample. The data sample only contains pre-processed values that are discrete values (states) for the variables. The resulting BN is a knowledge base that offers a probabilistic representation of the reality.

To initialize the diagnostic module, a data sample is required from which a first BN is discovered. This first BN is used to make inferences about the state of the Managed Element through the Querier module.

After a while, the knowledge base needs to be updated, to do so the Learning module takes the most recent pre-processed data in the monitoring database and executes the PC algorithm in order to update the underlying BN.

## IV. Approach Application and Assessment

### A. Study Case

In order to show the effectiveness of our approach in the SOA paradigm, we integrate the Diagnosis Model to an Enterprise Service Bus (ESB) platform[14], as Managed Element.

The diagnosis model creates a Bayesian network that represents the behavior of the ESB when it is working in a high demand environment. This representation is constructed from data samples collected from tests. With this knowledge base, we show how make inferences about the overall state of the Managed Element.

The Fig. 2 shows the experimental environment, the main components are: Client Generator, Service Providers, ESB platform and the Autonomic Manager. The client generator can generate clients that send many simultaneous request messages to service providers, these re-

quests are mediated by the ESB. The mediation process consists in routing the request and response messages to the providers and the consumers, respectively. Under these conditions the ESB becomes a bottleneck, and its behavior is observed by the Autonomic Manager building a knowledge base that helps to diagnose problems by monitoring only some aspects.

The environment was deployed in a PROXMOX Virtual Environment[1] which allows configure multiple Virtual Machines where the Client Generator, Service Provider and the ESB were deployed. Each VM has Linux Ubuntu 12.04 LTS as operating system and a QEMU Virtual CPU. The ESB product used is a WSO2 ESB. The service providers were deployed in a WSO2 AS.

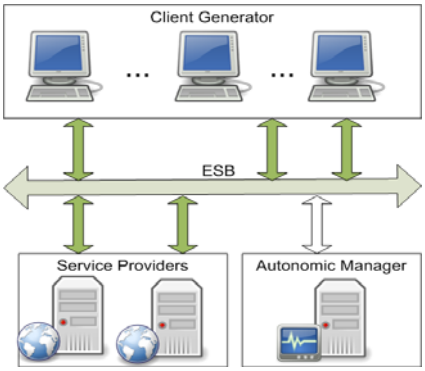


Figure 2. Experimental Environment.

In this particular study we collected data related for performance parameters of an ESB, namely CPU Load (CPU), Heap Memory Used (MU), Concurrency Level (C), and the Response Time (RT). These measures were recorded for each request message mediated by the ESB.

The CPU load and the Memory Used represent the percentage of usage of processor and memory in a given instant. The Concurrency Level is the number of request that mediated simultaneously by the ESB, and the Response Time is the time elapsed between sending a request and the receiving the response.

Aiming consider most like real conditions, we used configurations for the experimental environments able to generate from 4000 up to 32000 simultaneous requests in order to increase the workload in the ESB.

In each request-response messaging we measure the CPU, Memory load, the Concurrence Level and the Response Time. To calculate the Response Time we use two timestamps, when a request is sent (*t1*) and the time when the correspondent response is received (*t2*). The RT is calculated as $RT=t2-t1$. To calculate the Concurrence Level (C) we calculate the time intersection among the timestamps (*t1* and *t2*) of each request, that is, two request *i*, *j* are concurrent if $t1_i < t1_j$ and $t2_i > t1_j$.

### B. Data Pre-processing

Later running the scenarios and gathering a large data set under the same conditions for managed element, we have to define the states and intervals, for the variables, and classify the data.

As mentioned above, the data set consists of four variables: the CPU Load, the Heap Memory Used (MU), the Response Time (RT), and the Concurrency level (C). The Table I shows these variables and their states. Following these values, the data were classified.

TABLE I
VARIABLES AND STATES

| Variable | States |
|---|---|
| CPU | Low=[0,25), Med=[25,50), High=[50, 75), Very High=[75,100] |
| MU | |
| C | Low=[0,339), Med=[339,967), High=[967, 4219), Very high=[4219,9919] |
| RT | Preferred=[0,25) Acceptable=[25,50) Unacceptable=[50, 75) Very high=[75,100] |

### C. Discovered Knowledge Base

Having the states definition, we can execute the data pre-processing and the PC algorithm for all the collected data in order to build an initial knowledge base that represents the behavior of the ESB.

We used TETRAD IV API[2] to execute the PC algorithm, discover the associated DAG, and calculate the CPTs, the resulting BN is shown in the Fig. 3. This BN shows a strong dependency among the variables monitored and measured, concluding that an evidence in a variable will affect abruptly other ones. Eq. (4) is the global probability distribution associated.

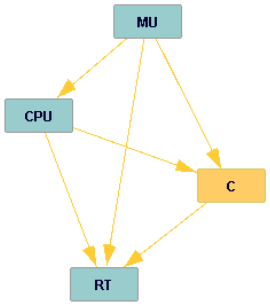


Figure 3. Discovered Bayesian network

$$P(CPU,MU,CL,RT) = P(RT \mid CPU,MU,C) \; P(CPU \mid MU) \; P(C \mid MU) P(MU) \quad . \quad (4)$$

### D. Probabilistic Diagnostic

The knowledge base can be used to make inferences with or without evidences. When no evidence exists, it is possible to calculate the probability that a variable is in a

[1] http://www.proxmox.com/

[2] http://www.phil.cmu.edu/tetrad

particular state regardless of the states of the other variables, this is the marginal probability. For example, the most probable state of the CPU in the generated knowledge base is *P(CPU=Very High) = 0.42*, this is because the high workload conditions.

The posterior probabilities indicate the most probable state in the variables when evidence is observed. In this way the Querier can detect atypical states and request a plan to improve or recover the performance in the Managed Element. For example, to know the effects of having a high concurrency level, we insert in the BN the evidence *C=High* getting the most probable states in the Table II. These values were calculated using TEDRAD IV. With these probabilities, it is possible to infer that a high number of concurrent requests trying to be attended cause unacceptable or error conditions over the components (high CPU load and Heap Memory in error condition), and in its performance (unacceptable Response Time).

TABLE II
MOST PROBABLE STATES OF VARIABLES

| Variable | State | Probability |
|---|---|---|
| CPU | Very High | 36% |
| MU | Error | 73% |
| RT | Unacc | 75% |

These results are useful for the Planning phase; since it is responsible to determine the required actions to recover the normal operation level of the ESB. In a cloud computing environment, the recovery actions could include increasing the number of CPUs and Memory capacity in order to reduce the Response Time.

## CONCLUSION

In this paper we have presented a diagnostic model based on probabilistic reasoning, which offers a closer representation of the real conditions of the Managed Element using a BN as knowledge base. It allows making inferences when evidences are observed giving more precise diagnosis.

Our approach also proposes a diagnostic model able to be adapted to changing conditions and requirements of a specific managed element. It can add new variables of interest and change the perspective from which inferences are made, taking into account the available data set and the costs of monitoring. The knowledge base can be updated in order to consider new conditions by means of the PC algorithm that enables learning tasks including new knowledge that is observed in recent monitored data.

We showed the results of integrating and execution of the diagnostic model monitoring a SOA platform. Using the built BN to make inferences about the state of the ESB, by means probabilistic reasoning.

In the future, the proposed approach could be improved to make predictions with the BN, or including other probabilistic models such as the Markov chains, which allows making predictions by observing changing state in the managed element over the time. Other possible inclusion in the knowledge base is a feedback that could assess the Plans and determine which is the most adequate to solve a problem.

## ACKNOWLEDGMENT

Part of this work has been supported by the Consejo Nacional de Ciencia y Tecnología (CONACYT) from México and by FP7-ICT IMAGINE research and development project, co-funded by the European Commission under the “Virtual Factories and Enterprises” (FoF-ICT-2011.7.3, Grant Agreement No: 285132).

## REFERENCES

[1] IBM Corporation, "An architectural blueprint for autonomic computing", http://www-03.ibm.com/autonomic/pdfs/AC, June 2005.
[2] J. O. Kephart and D. M. Chess, “The Vision of Autonomic Computing", Computer, vol. 36, no. 1, pp. 41–50, 2003. doi: 10.1109/MC.2003.1160055
[3] A. Khalid, M. A. Haye, M. J. Khan, and S. Shamail, “Survey of Frameworks, Architectures and Techniques in Autonomic Computing”, in: Autonomic and Autonomous Systems, 2009. ICAS ‘09. Fifth International Conference on, pp. 220-225, 2009. doi: 10.1109/ICAS.2009.38
[4] M. C. Huebscher and J. A. McCann, "A survey of autonomic computing - degrees, models, and applications", ACM Comput. Surv., vol. 40, no. 3, pp. 7:1–7:28, 2008. doi: 10.1145/1380584.1380585
[5] V. Sharma, A. Thomas, T. Abdelzaher, K. Skadron, and Z. Lu, “Power-aware QoS Management in Web Servers”, in: Proc. of the 24th IEEE International Real-Time Systems Symposium, USA, pp. 63–72, 2003. doi: 10.1109/REAL.2003.1253254
[6] I. Rish, M. Brodie, N. Odintsova, S. Ma, and G. Grabarnik, “Real-time problem determination in distributed systems using active probing”, in: Network Operations and Management Symposium, 2004. NOMS 2004. IEEE/IFIP, vol. 1, pp. 133–146 Vol.1, 2004. doi: 10.1109/NOMS.2004.1317650
[7] Y. Dai, Y. Xiang, and G. Zhang, "Self-healing and Hybrid Diagnosis in Cloud Computing”, in: Proc. of the 1st International Conference on Cloud Computing, China, pp. 45–56, 2009. doi: 10.1007/978-3-642-10665-1_5
[8] J. Dowling, R. Cunningham, E. Curran, and V. Cahill, “Building autonomic systems using collaborative reinforcement learning” Knowl. Eng. Rev., vol. 21, no. 3, pp. 231–238, 2006. doi: 10.1017/S0269888906000956
[9] A. Devitt, B. Danev, and K. Matusikova, “Ontology-driven Automatic Construction of Bayesian Networks for Telecommunication Network Management”, in: 2nd Int. Workshop: Formal Ontologies Meet Industry (FOMI 2006), 2006.
[10] P. Spirtes and C. Meek, “Learning Bayesian Networks with Discrete Variables from Data”, in: Proc. of the First International Conference on Knowledge Discovery and Data Mining (KDD-95), Canada, pp. 294–299, 1995.
[11] D. Barber, “Bayesian Reasoning and Machine Learning”, Cambridge University Press, New York, USA, 2012. ISBN: 978-0-521-51814-7
[12] W. Buntine, “Theory refinement on Bayesian networks”, in: Proc. of the seventh conference (1991) on Uncertainty in artificial intelligence, USA, pp. 52–60, 1991.

[13] J. Abellán, M. Gómez-Olmedo, and S. Moral, "Some Variations on the PC Algorithm", in: Third European Workshop on Probabilistic Graphical Models, 12-15 September 2006, Prague, Czech Republic. Electronic Proceedings, Czech Republic, pp. 1–8, 2006.
[14] D. A. Chappell, "Enterprise Service Bus", O'Reilly Media, Inc., 2004. ISBN: 978-0-596-55650-1