Situation assessment in autonomous systems

J.Aguilar[1] **A. Subias**[2,4] **L. Travé-Massuyès**[2,3] **K. Zouaoui**[2,3]

[1] *Cemisid; La hechicera, Ncleo Pedro Rincon Gutierrez, Universidad de los Andes, Mrida - Venezuela*
[2] *CNRS, LAAS, 7, avenue du Colonel Roche, F-31400 Toulouse, France*
[3] *Univ de Toulouse, LAAS, F-31400 Toulouse, France*
[4] *Univ de Toulouse, INSA, LAAS, F-31400 Toulouse, France*
aguilar@ula.ve;subias,louise@laas.fr

Abstract:
For autonomous systems, it is essential to be able to identify the current state i.e to perform *situation assessment*, in order to determine the action to execute to ensure correct functionalities. Specifically, an autonomous system requires a supervision component with the goal to detect and to diagnose the current situation, and then to determine the self-adaptive operations. We propose a self-adaptive architecture for autonomous communicating systems in which the situation assessment process feeds the reconfiguration system with an estimation of the traffic situation so that it can decide about the reactions appropriate to cope with the dynamic changes. The estimation of the traffic situation is implemented at the transport protocol level from the time-stamped quality of service (QoS) parameters by using a learning approach. A fuzzy clustering method is used to classify the system states in classes called *primitive patterns* and the transitions between classes are expressed in term of *events*. A set of simulated network traffic scenarios are used to illustrate the main principles of the approach.

Keywords:

1. INTRODUCTION

Autonomic Computing is a concept that links many fields of computing in order to create computing systems with self-management properties, e.g. self-protection, self-optimization, self-reconfiguration, etc. (Huebscher and McCann (2008)). This paradigm is necessary for the new generation of emergent applications with self-organizing properties, which require to solve several problems at the level of semantic characterization of the environment (context modeling), optimal discovery and selection of ubiquitous objects, adaptive mechanisms (learning and reasoning capabilities), etc. in order to generate an autonomic and context aware behavior.

In this paper we propose an approach for autonomic computing and more precisely a situation assessment approach for autonomous communicating systems at the transport level Situation assessment is a relevant challenge in autonomous communicating systems as it is an essential way to provide self-adaptive capabilities. Our approach put the situation assessment component at the core of the proposed autonomy architecture (Aguilar-Martin et al. (2011)). This architecture relies on a component-based approach such as the one proposed by *MPTCP* (Multi- Path Transport Control Protocol) and generalized by *ETP* (Enhanced Transport Protocol) Van Wambeke et al. (2008) Guennoun et al. (2008) to widely facilitate the design and development of new composed transport services. Indeed, new transport services are expected to result of the combination of pluggable components offering specific basic functionalities in terms of reliability, ordering, delay control, and congestion control. The problem considered in this paper is to characterize from the network available data the situations that call for specific basic functionalities and specific compositions.

The problem of having an explicit assessment of the communication situation is similar to a monitoring problem as considered in the data driven diagnosis community. Our claim is that we can track the evolution of the traffic from well-identified Quality of Service (*QoS*) properties. We propose to use learning techniques to identify the operation states describing the dynamically changing context situations arising from the distributed and collaborative mobile applications associated to communicating systems.

The feasability of our approach is demonstrated using simulated network traffic obtained from NS - 2 (Network Simulator V2). Although it has not been experimented on real Internet network traffic yet, the results are quite convincing given that the traffic provided by NS - 2 is rather realistic.

The paper is organized as follows. Section 2 presents a self-adaptive architecture for autonomous communicating system. Section 3 is dedicated to the method for situation assessment. A set of simulated network traffic scenarios are used to illustrate the main principles of the approach in section 3. Finally, concluding remarks are drawn in section 4 and the prospects for future work are discussed.

2. A SELF-ADAPTIVE ARCHITECTURE FOR AUTONOMOUS COMMUNICATING SYSTEMS

Coping with context changes in networked systems requires to provide adaptability to the traffic control and management system. This can be achieved through self-adaptive communication protocols that dynamically reconfigure the communication system according to the user's requirements and to the load of the communication resources.

In the well-known Open System Interconnection (OSI) referential model (Zimmerman (1980)), composed of seven layers, the transport layer is the lowest layer operating on an end-to-end basis between two or more communicating hosts. This layer is located between the applications and the network layer. Transport services enable applications to abstract the communication services and protocols provided by the lower network and MAC (Media Access Control) layers. Transport protocols specify the mechanisms to be implemented in order to offer the required transport services. Because the communication Quality of Service (QoS) is highly impacted by the specific transport protocol in use, our self-adaptation architecture targets the transport level and proposes to adapt the transport protocol as the communication context evolves.

Dealing with this problem not only requires a proper characterization of the alternative protocol properties but also the capability of monitoring the *QoS* to assess the communication context. These are at the basis of the decision to dynamically modifying the behavior of the communication protocol for each new context situation and executing the appropriate reconfiguration actions.

Figure 1 illustrates the architecture that has been proposed in (Aguilar-Martin et al. (2011)), and that is foreseen to provide a solution to this problem.

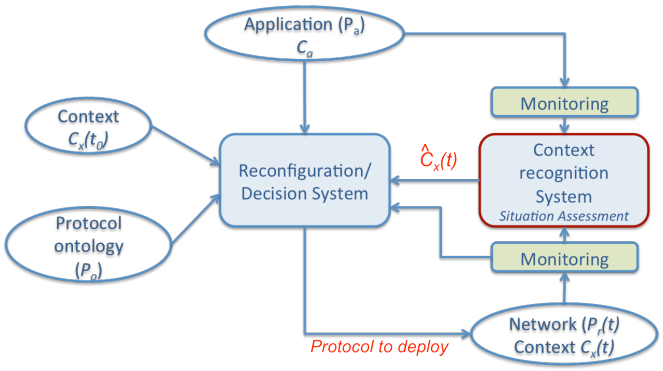

Fig. 1. Architecture for self-adaptation

The Reconfiguration/Decision System outputs the protocol to be deployed. This decision is taken upon several inputs:

- the properties P_o of the different available protocols gathered through an ontology
- the communication context at time t_0 $C_x(t_0)$
- the properties P_a required by the application and the application context C_a
- the current context $C_x(t)$ recognized by the *Context Recognition System*, i.e. $\hat{C}_x(t)$.

Our approach focuses on the Context Recognition System, which monitors and assesses the communication context and related QoS, receiving information from monitors observing, on one hand, the application context C_a and, on the other hand, the network properties $P_r(t)$.

3. SITUATION ASSESSMENT FOR AUTONOMOUS SYSTEMS

3.1 Principles

In the proposed strategy, situation assessment aims at alerting the Reconfiguration/Decision System every time a new situation arises in the network and at identifying the new situation. A situation is related to an evolution of the *QoS* parameters of the studied communicating system. Therefore, situation assessment induces the capability to detect different relevant traffic situations taking the discriminating features in terms of QoS indicators of such situations into account.

The entire situation assessment strategy is based on the analysis of time-stamped QoS parameters issued from the communicating system that can be logged in the form of time series. Two different but non-independent steps can be pointed out:

- an off-line step in which historical time series are analyzed and processed along an unsupervised classification approach to characterize the behavior of the system according to different clusters. This characterization is performed by determining *primitive patterns* corresponding to the clusters, and *successions* corresponding to sets of chronologically neighboring system states as illustrated in Figure 3. This step allows us to characterize the system evolution in terms of high level states, also called *operation states*. An *event identification* phase allows us to characterize the transitions between operation states and to give semantics in terms of QoS parameters to the transitions.
- an on-line step, during which the cluster model identified at the previous step is used as a classifier to determine the actual state of the process from on-line acquired data. The classifier model hence constitutes the situation assessment mechanism of the Context Recognition System in the architecture given in Figure 1.

This paper focuses on the off-line step decomposed in the following tasks :

(1) *Generation of a training data base* for collaborative communicating systems,
(2) *Specification of primitive patterns and successions* : this is the clustering phase in which each data sample is assigned to a class according to its feature values. A class can be seen as the operation state of the communication system during a time epoch. Therefore the behavior of the system is represented by a chronologically ordered sequence of classes. In this sequence each class constitutes a *primitive pattern*. Each primitive pattern is associated to a *succession of samples* (belonging to the corresponding primitive pattern) and the time interval during which all

samples belong to the same class. During this time interval, the system operation state is assumed to be the same.

(3) *Determination of events*: an event is defined for each transition between primitive patterns i.e. when the communication system evolves from one operation state to another. *Transition zones* are defined and characterized by specific durations, initial class, destination class etc., by using the information referring to the concerned samples.

3.2 Generation of a training data base for collaborative communication systems

In the context of networked communicating systems, the data is obtained from a simulator of communication systems that can be parameterized by a given protocol within a set : TCP (Transmission Control Protocol), TCP and UDP (User Datagram Protocol), etc.

The selected features must be relevant for characterizing the different situations to be detected. In our case, we use standard QoS parameters, namely delay, bit rate, jitter and percentage of lost packets. Every data sample has hence a time stamp and four dimensions.

Several scenarios inducing representative behavior, have been defined and simulated, e.g. router congestion, connection loss, etc. These scenarios are described by the communication protocols used (TCP, UDP, etc.), the sources of packet emission, etc. They are given to the simulator, whose simulation traces provide all kind of information, i.e. ID packet, packet size, emission time, emission address, reception address, etc. for the different sites of the network.

These data is filtered from the point of view of the source and the destination of the packets for which one wishes to analyze the faulty situations (emission and reception addresses), and from the point of view of the information that is required to calculate the feature values. For example, to calculate the delay, the emission and reception time of a packet are required. This information is determined/searched for from the trace file. A script must be elaborated for each feature.

Fig. 2. Classification results for the UDP / TCP scenario

3.3 Specification of primitive patterns and successions

The process characterization step, which leads to the classifier to be used for on-line situation assessment, corresponds to a learning stage based on the clustering methodology LAMDA (Learning Algorithm for Multivariate Data Analysis) known in the Data Mining research community (Kempowsky et al. (2003)). LAMDA is a fuzzy methodology for conceptual clustering and classification. It is based on finding the global membership degree of a sample to an existing class, considering all the contributions of each of its features. This contribution is called the *marginal adequacy degree* (MAD). $MADs$ are combined using "fuzzy mixed connectives" as aggregation operators in order to obtain the *global adequacy degree* (GAD) of an sample to a class (Aguilar-Martin and de Mantaras (1982)). The $LAMDA$ methodology enables classifying samples represented with quantitative and/or qualitative, features. When the feature is of numerical type, the MAD is calculated by selecting one out of different distribution function possibilities. When the feature is qualitative, the observed frequency is used to evaluate the MAD. To avoid the assignment of a poor representative sample to a class, i.e a sample with low GAD, a minimum GAD threshold is used. This threshold corresponds to the GAD given by the Non-Informative Class (NIC). Therefore, if the maximum value of all the $GADs$ of a sample is less than the NIC threshold, it is not classified in any of the existing classes. The $LAMDA$ method allows one working simultaneously with multiple features of differnet types, i.e. corresponding to numerical information and qualitative information (Carrete and Aguilar-Martin (1991)). Moreover, the characteristics of the resulting classes after a learning stage are easily interpretable. This method allows one mixing different learning strategies as well as combining recognition. $LAMDA$ accepts the creation of new classes given a number of observations that have been rejected (unrecognized), keeping the previously created classes. It also accepts the evolution of the existing classes as well as the creation of new classes. For these reasons, the strategy described in this paper has been implemented using the $LAMDA$ method via a software tool called $SALSA$ (Situation Assesment using the $LAMDA$ claSsification Algorithm) which implements the method and provides an easy to use user interface (Kempowsky et al. (2003)) (see figure 2). $SALSA$ has been designed to enhance the knowledge of the expert. If supervised learning is used (i.e. the learning can be done with a priori knowledge of the different possible categories of samples), the expert, according to some criteria, pre-defines a number of significant groups or classes to a set of measured data from the process.

Four our case study an unsupervised learning strategy has been used since no prior knowledge about the possible situations is given i.e. there are no pre-defined classes. After several tests with different parameter tunings (selection of different membership functions, exigency index, etc.), a suitable partition is obtained such as the one presented in fig 3. In this case 57 samples have been classified and the resulting classification is composed by five classes.

Each class is a *primitive pattern* (in the example 5 primitive patterns are defined). A *succession*is a continuous

sequence of samples assigned to the same class. That is, *successions* correspond to time intervals in which all individuals belong to the same class (they have the same primitive pattern label) i.e. time intervals where the system operation state is similar (in the example, 8 successions are defined).

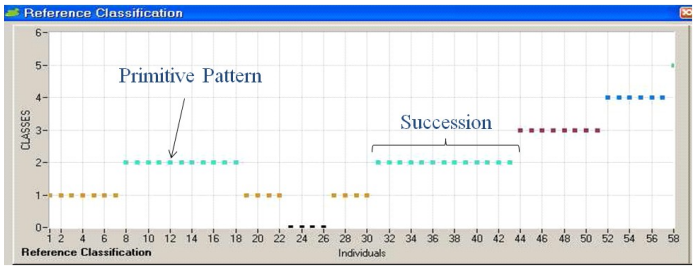

Fig. 3. Primitive Patterns and Successions

3.4 Determination of events

The change of primitive pattern, or transition, indicates an evolution of the system behavior, which can be associated to an event. By using the classification previously established, all the transitions are easily identified in terms of source and destination primitive patterns. The problem is then to characterize the transitions in order to get more insight into their semantics. For a given transition between a source primitive pattern and a destination primitive pattern, one simple solution is to characterize the transition by the concerned individuals (i.e. the last individual of the source and the first individual of the destination). In this case the events are derived from only two individuals, which may be questionable.

Another solution that we have adopted is to define a *transition zone*. The idea is to consider more than these two individuals to characterize an event (for that, we consider the associated *successions* respectively to the source and destination patterns). To determine the set of samples that define a transition zone, we make use of the classification method LAMDA is fuzzy. Indeed, the classification method $LAMDA$ assigns to each individual a membership degree for each existing class, the GAD, and a sample is assigned to the class for which its GAD is maximum.

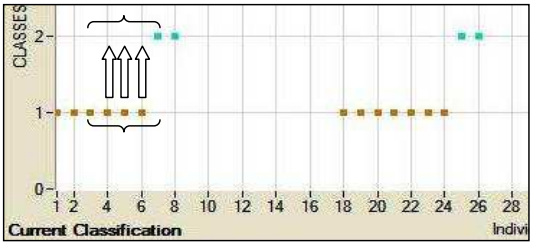

Fig. 4. Determination of a transition zone

So, to determine the transition zones a threshold is first assigned to each class. Then for each individual its GADs to the different classes are compared to the corresponding thresholds. For a given transition, the transition zone is defined through all the samples of the source or destination primitive patterns (i.e. their associated *successions*), such that their $GADs$ are superiors to the threshold of the destination or source primitive patterns, respectively. By defining the transitions zones in this way, more information is used to derive the events. However, determining the threshold definition is an open problem. A first proposal is to define the threshold of a class ($Threshold_\alpha$) from the characteristics of this class in terms of the number of samples assigned to the class (N_α), the minimal value of GAD for the assigned samples of the class (GAD_α^{min}) and the maximal value of GAD for the assigned samples of the class (GAD_α^{max}).

$$Threshold_\alpha = GAD_\alpha^{max} - \frac{GAD_\alpha^{max} - GAD_\alpha^{min}}{N_\alpha}$$

This threshold allows one to establish the transition zone by taking samples of a class only if this class includes a significant number of individuals.

The individuals considered to define a transition zone are no longer assigned to a class. Therefore, the successions of the different primitive patterns must be updated.

Each transition zone is then characterized by a starting date (t_s), an ending date (t_e), a source primitive pattern (PP_i) and a destination primitive pattern (PP_j). t_s and t_e are, respectively, the minimal and maximum time instant of the samples of the transition zone.

An event e_{ij} is associated to each transition zone, where PP_i is the source primitive pattern and PP_j is the destination primitive pattern of the transition. The occurence date t_k of e_{ij} is given by the mean of t_s and t_e.

3.5 Event interpretation

In our approach, we only consider the information provided by data to estimate the events which caused a transition from an operation state of the communication system during a time epoch to another. Then, we detect significant differences between the qualitative values of the features of the two classes concerned in a transition by using the class profiles provided by the classification tool $SALSA$. Moreover, we use a simple semantics introduced by Kempowsky et al. (2005) characterizing the detected variations between the features of the classes, whether they have increased, decreased or remained practically unchanged. For each event associated to a transition between a class C_k and C_l the values of each feature i.e percentage of losses, throughput, end to end delay and jitter, in the two class profiles are analyzed and a marginal event about the feature d_i takes values $\leftrightarrow$, $\uparrow$ or $\downarrow$, depending on the definition of a user threshold δ previously chosen as follows:

- if $|d_i(C_k) - d_i(C_l)| < \delta$ the marginal event is $\leftrightarrow$
- if $d_i(C_k) > d_i(C_l) + \delta$ the marginal event is $\downarrow$
- if $d_i(C_k) < d_i(C_l) - \delta$ the marginal event is $\uparrow$

At the end of this step, a sequence of dated events is obtained which models the communicating system QoS evolution qualitatively and each event is interpreted through the marginal event associated to each of its features. This information is useful because during the recognition step it can guide the analysis towards the feature(s) whose variations best characterize the observed change of the communicating situation. This aims to provide a better

understanding of the communicating situation in order to guide the reconfiguration system in its choice of the new composed transport services suitable to the recognized situation.

4. APPLICATION TO THE TRANSPORT LAYER OF A COMMUNICATION NETWORK

In this section we present our off-line learning algorithm is tested against several scenarios using simulated traffic.

4.1 Scenario description

As a first step, we have considered the particular situation of a communication system that is the situation of packet loss by congestion. In order to test our approach we used $NS-2$ (Network Simulator V2) and $OTCL$ (Object Tools Command Language) to describe and simulate two identical scenarios of packets loss by congestion unlike the flows generated by the simulated network nodes causing congestion as mentioned in the figure 5. In the first simulation, the congestion is caused by two UDP flows; in the second congestion, it is caused by a TCP flow and one UDP flow. Each simulation contains three times the congestion operation state and the length of each simulation is 20 seconds. The simulated network congestion is caused

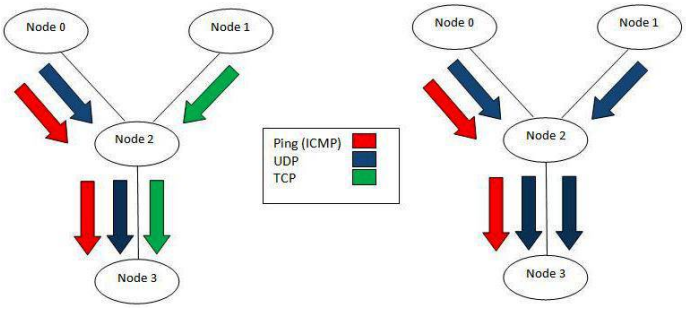

Fig. 5. Physical architecture of the simulated network

by UDP/TCP or UDP/UDP respectively issued by the node 0 and node 1, so the above scenario shows three successive situations of congestion. Ping packets from the "Node 0" are the sensors of our system. We will follow the traces of Ping packets only then we will release the descriptors of our system in a state of congestion through these traces. The results of these simulations are very different. Indeed, TCP provides congestion control, i.e., trial to resolve the situation when network congestion occurs, and decrease the frequency of emission. By against, a UDP node continues broadcast regardless of the losses caused.

4.2 Training data set

The features that we have chosen are conventional parameters in terms of network QoS. This is the *percentage of losses, throughput, end to end delay* and *jitter*. To calculate these features, the following assumptions are considered:

- The $ICMP$ traffic (PING packets) is assumed regular (64 bytes every 0.1 seconds)
- The clocks of different nodes are synchronized [1]
- Each package contains a unique sequence number in its flow
- Each acknowledgment carries the sequence number of the acquitted package
- Each package contains a temporary stamp field (11) which indicates the time of its issue
- Each acquittal includes information about the time of sent (temporary stamp) and time of receipt of acquitted package

The tool $NS-2$ provides simulation results as a log file collected from four nodes. This file has the following structure (figure 6):

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]
Action	Instant	Nsrc	Ndest	Type	Taille	Flag	IDflux	@src	@dest	N°Seq	IDpaquet	Timestamp

Fig. 6. Structure of the log file generated by NS-2

Our goal is to extract from this file information about each PING packet sent from node 0 and received by node 3. To do this we have developed two calculation scripts using the Perl language. The first script makes the calculation of the features at the transmitter side and the second at the receiver side.
The resulting file has the following structure (figure 7):

[0]	[1]
N° Individual	Instant

[0]	[1]	[2]	[3]
Jitter	Delay	throughput	% loss

Fig. 7. File structure resulting from the calculation script

Each line provides a sample computed at a given time and contains values of the four features. A feature is calculated at time of receipt of the tenth ping packet. The file resulting from the simulation UDP/UDP contains 57 samples and the one resulting from the simulation TCP/UDP contains 173 individuals.

4.3 Primitive patterns and successions

The file resulting from each calculation script developed in the preprocessing phase is the entry of SALSA. As already said, the file contains lines of the four values of the features (% loss, throughput, delay and jitter), i.e. each line is a sample. The Figure 2 shows the classification results obtained for the UDP / TCP simulation. The profile of each class (lower part of figure 2) is given with normalized values of the four features. The top of figure 2 shows the results of the classification and primitive pattens are easily identifiable (the first primitive pattern includes 17 first samples.

4.4 Events determination

According to section 3.4, after the classification we need to define the transition zones to determine the events. For that we use the threshold defined in section 3.4 to determine the samples belonging to the different transition

[1] The $NS-2$ simulator used a unique clock shared by all the nodes. In a real network, solutions exist such as the *Network Time Protocol* (NTP) so that all the nodes operate in the same temporal space.

zones. Then, with the previous data we determine the event for each transition zone. Each event(e_{ij}, t_k) is characterized as described in section 3.4. Table 1 is an extract of the results obtained for the TCP/UDP simulation.

Table 1. Events for TCP/UDP simulation

event	t_k
e_{13}	0.0049
e_{31}	0.0112
e_{32}	0.0123
e_{21}	0.0203
...	...

Moreover an interpretation is associated to the different events. For instance, the first event e_{12}, characterized as indicated by table 2, traduces the evolution from the normal situation of the communicating system to the start of the congestion.

Table 2. Event e_{12} characterization

feature	marginal event
jitter	$\leftrightarrow$
delay	$\uparrow$
throughput	$\downarrow$
loss	$\uparrow$

5. CONCLUSION

In this paper, an approach for self-adaptive autonomous systems based on situation assessment has been proposed. We have investigated the idea of applying a data-driven diagnosis technique for networked systems supervision and management. We elaborated a learning based approach to capture the different traffic context situations that need to be detected for run-time adaptation of the communication protocols. The approach is illustrated with protocols at the transport level.

Our claim is that we can elaborate a model-based diagnosis approach for the correct design of adaptive communication protocol without much expert knowledge but using data mining techniques. The experiments presented in this paper are the first step of our study. They are based on a situation assessment method that uses a simple "static" classifier to recognize the situation, from the analysis of the feature values at some time instant. We believe that more sophisticated structures that include the temporal aspect can add significant power to the recognition system. Indeed, standard transport protocols implement strategies that act upon hard-coded event patterns representing specific traffic situations. These are detected at the source or destination nodes. The events arise from feedback provided by standard parameters stamping the packets and the event patterns generally express temporal relations. Our future work will be oriented towards formalizing the temporal patterns in use in standard protocols using *chronicles* with the aim to express more sophisticated patterns and hence to generalize traffic situation assessment Subias et al. (2010). Currently we are working on the problem of learning chronicles, which is formulated as an extension of the work presented in this paper. The first phase indeed uses the classification method as presented here and the second is based on temporal data mining techniques.

ACKNOWLEDGEMENTS

This work has been developed in collaboration with the research team SARA of LAAS-CNRS. The authors thank Christophe Chassot, Ernesto Exposito and Code Diop from SARA for their collaboration.

REFERENCES

Aguilar-Martin, J. and de Mantaras, R.L. (1982). Approximate reasoning in decision analysis. *The Process of Classification and Learning the Meaning of Linguistic Concepts, North Holland Publishing Company*, 165–175.

Aguilar-Martin, J., Subias, A., Travé-Massuyes, L., and Zouaoui, K. (2011). A chronicle learning approach for self-adapting strategies in collaborative communicating systems. Technical Rapport, LAAS-CNRS, France.

Carrete, N. and Aguilar-Martin, J. (1991). Controlling selectivity in nonstandard pattern recognition algorithms. *Systems, Man and Cybernetics, IEEE Transactions on*, 21(1), 71–82.

Guennoun, K., Drira, K., Van Wambeke, N., Chassot, C., Armando, F., and Expósito, E. (2008). A framework of models for qos-oriented adaptive deployment of multilayer communication services in group cooperative activities. *Computer Communications*, 31(13), 3003–3017.

Huebscher, M. and McCann, J. (2008). A survey of autonomic computing degrees, models and applications. *ACM Computing Surveys Journal*, 20(3), 1–28.

Kempowsky, T., Aguilar-Martin, J., Subias, A., and Le Lann, M. (2003). Classification tool based on interactivity between expertise and self-learning techniques. *International Symposium IFAC-Safeprocess, Washington DC, USA*.

Kempowsky, T., Subias, A., Aguilar-Martin, J., and LeLann, M. (2005). Online continuous process monitoring by means of a finite state machine generated using learning techniques. In *18th International Congress on Condition Monitoring and Diagnostic Engineering Management (COMADEM'2005)*, 221–231. Cranfield (UK).

Subias, A., Exposito, E., Chassot, C., Travé-Massuyès, L., and Drira, K. (2010). Self-adapting strategies guided by diagnosis and situation assessment in collaborative communicating systems. In *International Workshop on Principles of Diagnosis (DX 10)*. Portland (USA).

Van Wambeke, N., Armando, F., Chassot, C., and Expósito, E. (2008). A model-based approach for self-adaptive transport protocols. *Computer Communications*, 31(11), 2699–2705.

Zimmerman, H. (1980). Osi reference model – the iso model of architecture for open systems interconnection. *IEEE Transactions on Communications*, 28, 425–432.