Analytical framework for QoS aware publish/subscribe system deployed on MANET

Imene Lahyani, Lamia Ben Amor, Mohamed Jmaiel, Khalil Drira, Christophe Chassot

Analytical framework for QoS aware publish/subscribe system deployed on MANET

Imene Lahyani[1,2,3], Lamia Ben Amor[3], Mohamed Jmaiel[3], Khalil Drira[1,2]and Christophe Chassot[1,4]
[1] CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France
[2] Univ de Toulouse, LAAS, F-31400 Toulouse, France
[3] University of Sfax, ReDCAD, B.P.W, 3038 Sfax, Tunisia
[4] Univ de Toulouse, INSA, F-31400 Toulouse, France
{iabdenna,khalil.drira,chassot}@laas.fr, lamia.benamor@redcad.org, mohamed.jmaiel@enis.rnu.tn

***Abstract*—In this paper, we propose an analytical framework for monitoring and analyzing QoS of publish/subscribe systems on MANET. Our framework copes with intermittent connectivity frequently occurring in MANET and provides statistical methods allowing detecting QoS degradations affecting links between brokers at the middleware layer. Besides, our analytical framework identifies QoS degradation source which enables to repair the system. The proposed framework is extensively evaluated with simulation experiments. Simulations results show its efficiency.**

I. Introduction

A publish/subscribe (Pub/Sub) [17] system is a networked communication infrastructure where messages are provided by publishers and then delivered to receivers whose subscriptions match the messages. The broker, named also dispatcher or channel manager, represents the most important component in the system. It has to perform the matching between the incoming events and subscriptions in the system and it also has to send to the subscribers all the events for which they have expressed interest. Such a communication paradigm is promising in mobile environments owing to its decoupling nature ensuring anonymity and decoupling between communicated entities. However, deploying pub/sub systems on MANET [5] is a challenging task due to QoS degradation problems caused by node's mobility especially for delay sensitive applications. In order to face QoS degradations, network routing protocols in MANET always search for a route connecting brokers. However, there is no guarantee to find this route and it is possible that the route found does not fit application requirements. Thus, network routing protocols cannot always solve QoS degradation problems occurring at this layer. Consequently, a solution should be introduced at the middleware layer. Facing this problem, several studies [10] react at this level by taking into account some QoS parameters. These studies seek to address the challenge of managing latency performance for pub/sub systems deployed on MANET. However, they did not provide an efficient solution to this problem since they compare parameters values with fixed threshold and are restricted to a specific subscription language or to a specific topology.

In this paper, we deal with these problems and we propose a delay-aware dynamic framework aiming at providing best adapted QoS in the system according to the available resources by managing a permanent connectivity guarantee between brokers. To achieve the mentioned goal, adaption can be accomplished by migrating a faulty broker to another node in the network ensuring better QoS. Adaption may also be achieved by splitting an overloaded broker in order to minimize his load.

The proposed analytical framework integrates in each node modules which extract QoS parameters and analyzes them in order to detect QoS degradations and identify its causes. The first step of our approach is monitoring QoS parameters by intercepting events. Second, statistical indicators of failure occurring in the system are computed by the analysis module. This module does not refer to fixed thresholds. However, we use an adaptive procedure in order to update the threshold values according to the latest available information. The statistical method used in the analysis step to detect failures is based on the Extreme Values Theory (EVT) [7] as well as the test of Shapiro and Wilk [19]. Our approach approximates QoS measured parameter with Gumbel [6] and Gaussian distributions [6], then tracks mean and max values of the QoS traces by following fluctuations in the time series using updated thresholds. This enables to identify system state: well, acceptable or not well. Once failures are detected, the analysis module identifies their sources using the correlation method [12] which studies the dependence between latency increase and possible factors. In order to validate our approach, we carry on simulations experiments. Simulation results show that our approach provides QoS guarantee for pub/sub systems on MANET and detect failure which preserves system survivability. As a result, our framework is not restricted to a specific topology or a specified subscription language. However, it 's a generic framework which may be used in cooperative applications.

The rest of the paper is organized as follows. The design and analysis of our analytical framework is described in Section II. Section III presents the performance evaluation of our approach under different link failure conditions. Section IV summarizes the related work and Section V concludes the paper.

II. The Analytical framework

In this paper, a QoS based framework is proposed for QoS insurance of Pub/Sub systems on MANET. The major goal of the proposed framework is to bypass link failures and increase the chance to meet QoS requirement for such systems. The proposed approach monitors QoS parameter values by intercepting events. Then, analyzes these values in order to decide about system state. Finally, it reasons about an occur degradation source whenever a QoS degradation is identified. Our framework based on mathematical models relies on statistical methods to analyze the system state and provides a strong QoS support. It approximates latency values with Gaussian and Gumbel distributions, then compute adaptive thresholds. Failures are detected by comparing latency values with appropriate thresholds. Moreover, QoS degradation source is identified using the correlation method as a measure of dependency between two distributions. Figure 1 describes the proposed approach.

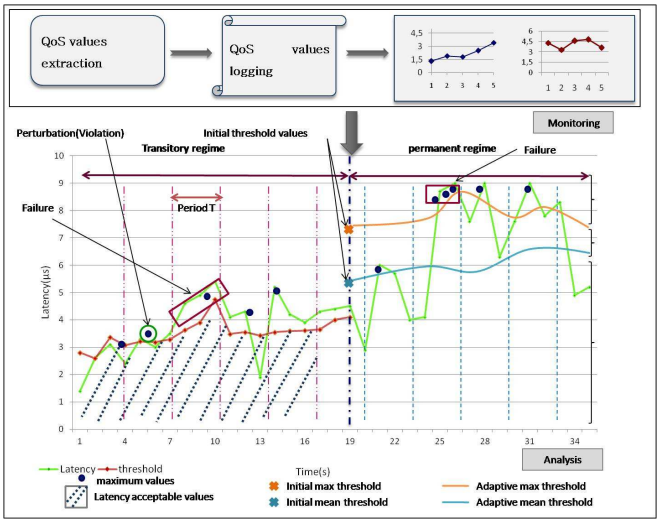

Fig. 1. The proposed approach

A. Monitoring

Monitoring constitutes a fundamental step in the proposed framework. It has to provide to the analysis level the information useful for its operation. A first level of Monitoring step is calculating latency values relying on event system messages. In fact, a monitoring module integrated in each broker intercepts event when translating between neighboring brokers. However, if there is no event delivery during a long time, specific messages are used in order to analysis QoS link state. Latency is calculated locally by the broker receiving the event. It' is defined as the time elapsed between the time, T1, when the first broker B1 sends the first bit of a packet to his neighbor B2 and the time, T2, when the second receives the last bit of that packet at $T1+\Delta T$. Thus, the latency that takes the event between B1 and B2 is defined with the following formula:

$$latency(B_2, B_1) = T_2 - T_1 \pm (offset) \quad (1)$$

where the clock offset refers to difference in the time reported by the two clocks. After computing latency values, the monitoring module saves these values in a log file.

B. Analysis approach

QoS analysis lays on the monitoring step. It exploits QoS parameters extracted from log files in order to analyze the system and detect QoS degradation. Implementing effective analysis needs to reason about the degradation source once QoS violations are noticed. In fact, identifying source failure creates opportunities to execute the appropriate reparation action in order to repair the system. Statistical analysis is used to detect QoS degradation. It's based on comparing QoS values with adaptive thresholds. Thresholds should be adaptive to the changes in network conditions (such as delay and failures), and are computed dynamically referring to latency series approximation with Gumbel and Gaussian distributions. Finally, when QoS degradations are detected, the correlation method is used to study the relation between latency increase and other perturbation factors.

A) Mathematical models for QoS analysis:

Let the analyzed QoS parameter be described by a sequence collected over time and denoted $QoS(t)$. The sequence is divided into periods composed of M samples. Then, the *max*, as well as *mean* values are extracted from each period. In this way, two new time series are obtained from the sequence of periods forming $QoS(t)$ as follows:

$$(QoS_{\max}(i)) = \max QoS_{iM+k} \quad (2)$$

Maximal values where k in [0,M-1]

$$\overline{QoS(i)} = (1/M)\Sigma QoS_{iM+k} \quad (3)$$

Mean values where k in [0,M-1]

Statistical properties of the *max*, and the *mean* are approximated by empirical distributions. In fact, the *Gumbel* distribution for the *max* and the *Gaussian* distribution for the *mean* sequence. Approximations enable to calculate initial thresholds for each series. Failure detection is then performed by comparing latency values with thresholds.

In order to prove that the series can be approximated with empirical distributions, we start with a first step named transitory regime.

1) Transitory regime: Demonstrating the perfect matching with the Gaussian distribution requires executing a normality test. So, we resort to the test of "SHAPIRO and Wilk" [19].

However, in order to prove that max values are best suited to the Gumbel distribution, we execute the following steps: First of all, we have to note that the cumulative distribution function is defined as:

$$F(x) = e^{-e^{(-(x-x_0)/S))}} \quad (4)$$

where x_0 is the location parameter and S is the scale one. Hence we have to:

- Sort the n samples forming our $\max$ distribution $QoS_{\max}(i)$ where i belongs to [1,n] in an increasing order;

- attribute a rank r to each value;
- measure the empirical frequency defined as:

$$ef = (r - 0,5)/n \tag{5}$$

- calculate the Gumbel reduced variable defined as:

$$u = -\ln(-\ln(F(x)) \tag{6}$$

- plot the curve defined by $(u_i, QoS_{\max}(i))$ couples.

Once we can fit the plotted point by a straight line, then, we make sure that our distribution is well suited to the Gumbel one.

In order to extract the Gumbel parameters, we calculate both the scale parameter S and the location parameter x_0 using the method of moments as shown in the following formulas .

$$x_0 = \mu - S\gamma \tag{7}$$

where μ is the mean of the $QoS_{\max}$ distribution and γ is the Euler constant defined as γ= 0,5772.

$$S = \frac{\sqrt{6}}{\pi}\sigma \tag{8}$$

where σ denotes the standard deviation of the $QoS_{\max}$ distribution

Based on the extracted parameter values, we can at the end of the first step, calculate initial threshold values as denoted in the following table.

TABLE I
FORMULAS USED TO CALCULATE INITIAL THRESHOLD VALUES

	Cumulative distribution Function	Threshold value
Max values	$G(z) = \exp(-\exp(-(z - x_0)/S)$	$Th_{QoS_{max}} = x_0 - S\ln[\ln(\frac{1}{p})]$
Mean values	$F(z) = \Phi((z - \mu_{QoS_{mean}})/\sigma_{QoS_{mean}}$	$Th_{QoS_{mean}} = f(\phi^{-1}(p), \mu_{QoS_{mean}}, \sigma_{QoS_{mean}})$

A simple method to detect the presence of peaks in this phase consists in setting thresholds. Appropriate threshold values are first obtained by calculating the conditional mean and the associated confidence interval. These values are calculated using the following formulas.

$$\overline{QoS} = \frac{1}{n}\sum_{i=0}^{n-1} QoS_i \tag{9}$$

$$CI^+ = \overline{QoS} + t_\alpha \hat{\sigma}/\sqrt{n} \tag{10}$$

The first formula defines the mean calculated at a point of time, however, the second presents the confidence interval relative to that mean at the same instant.

In order to perform an occur analysis, chronicles are used to detect QoS degradations in this phase. In fact, after N successive QoS violations, our analysis module triggers an alarm. A key question to ask is whether it is possible to mathematically compute N, by considering that the probability of N messages in failed state is less than the probability of failure tolerated by the user [18]. Let :

- N :denotes the number of successive violations leading to QoS degradation detection.
- LV: corresponds to a measured Latency higher than a threshold
- LOK: corresponds to a measured value under the threshold.
- $Risk$ is the defect probability specified by the user.

In order to determinate a precise value of N, we use the following expression :

$$P[NsuccLV] \preceq Risk \tag{11}$$

In addition, the probability that the next message is in violation state denoted by LV, is equal to the probability of being in acceptable state LOK multiplied by the probability of going from this state to the state LV added to the probability of being in state LV multiplying the probability of staying in this state in the next instant.

$$P[anystate \rightarrow LV] = P[LOK] * P[LOK \rightarrow LV] + P[LV] * P[LV \rightarrow LV] \tag{12}$$

Besides, the probability of obtaining N successive violations is equal to the probability to reach a LV state, after leaving any state, multiplied by the probability of staying in LV state (N-1) times:

$$P[NsuccLV] = P[anystate \rightarrow LV] * P[LV \rightarrow LV]^{N-1} \tag{13}$$

When we replace the probability $P[Nsucc \rightarrow LV]$ in the first inequality, we obtain the following expression providing the lower bound for N:

$$P[anystate \rightarrow LV] * P[LV \rightarrow LV]^{N-1} \leq (Risk) \tag{14}$$

Thus, we obtain $N \geq \lambda$ where

$$\lambda = 1 + \frac{ln\frac{Risk}{P[anystate \rightarrow LV]}}{ln(P[LV \rightarrow LV])} \tag{15}$$

When we choose an N's value greater than the smallest one,we accurate the degradation detection rate.

2) Permanent regime: Towards detecting the out of range values and deciding about system state, latency values are compared to the corresponding thresholds. Adaptive thresholds are dynamically updated using the exponential weighted moving average formula [16]:

$$Th_i = \lambda\Sigma(1 - \lambda)^j * QoS_{i-j} + (1 - \lambda)^i * Th_0 \tag{16}$$

where j in $\{1, i - 1\}$ and Th_0 denotes the initial Threshold.

Using the same formula used in the previous phase, chronicles are computed in order to perform an occur analysis. Thus, degradation detection algorithm trigger alarms when N successive violations happen denoted by $Max\_Nbr\_Succ\_Violation$ in Algorithm 1. As illustrated in Algorithm 1, to each measured latency value (line 5), the detection algorithm increases the number of violations $Nb\_Violation$, in (line 7) whenever the new value exceeds the

Algorithm 1 failure detection algorithm

1: $max\_threshold = Gumbel.update\_threshold$ $(maxValues, initial\_max\_thres)$
2: $mean\_threshold = Gauss.update\_threshold$ $(meanValues, initial\_mean\_thres)$
3: Idle: $Nb\_Violation = 0$
4: **loop**
5: $On\_New\_Measured\_Latency()$
6: **if** $(Latency > max\_threshold)$ **then**
7: $Nb\_Violation++$;
8: **else**
9: goto: Idle
10: **end if**
11: **if** $(Nb\_Violation > Max\_Nbr\_Succ\_Violation)$ **then**
12: identify_source()
13: **end if**
14: **end loop**

Algorithm 2 Diagnostic algorithm (first scenario)

1: **if** isCorrelated (latency, hopcount) **then**
2: degradation_category=mobility
3: degradation_cause = hop count variation
4: **if** verify_position($Broker_i$) **then**
5: possible_cause= $Broker_i$ is moving
6: **if** verify_speed($Broker_i$) **then**
7: possible_cause= $Broker_i$ speed is high
8: **end if**
9: **end if**
10: **if** verify_position($Broker_j$) **then**
11: possible_cause= $Broker_j$ is moving
12: **if** verify_speed($Broker_j$) **then**
13: possible_cause= $Broker_j$ speed is high
14: **end if**
15: **end if**
16: **end if**

threshold value (line 6). In case of non successive violations (line 9), the execution will jump to the initialization at line 3. Otherwise, it increments the number of successive violation until triggering alarms and reasoning about degradation source (line 12).

B)Degradation source identification approach:diagnostic approach

After detecting a QoS degradation, we start the reasoning about its source. We start by identifying degradation category which may be either mobility or overload. Finally, we identify precisely the factor causing the failure. Table 2 resumes possible failure causes that may degrade QoS of the link between two brokers denoted by B_1 and B_2.

TABLE II
FAULTS CATEGORY AND POSSIBLE CAUSES

Fault category	Fault source
Mobility	B_1 is moving far from B_2
	B_2 is moving far from B_1
	B_1 and B_2 are moving
Load	load of the P/S system
	load of an external application
Others	obstacle, noise

The key idea of source degradation identification is to study the dependency to any statistical relationship between two random variables or two sets of data. Doing this, we use the correlation method [12] towards studying the relation between latency value variation and parameters affecting latency such as hop count variation or overload variation.

Correlation method requires two steps. The first one consists in computing the Pearson correlation coefficient PCC value studying the dependency between two distributions X1 and X2 as described in the following equation.

$$PCC = cov(X1, X2)/\sqrt{(var(X1) - var(X2)}$$

The second step consists in executing a significance test of the correlation coefficient by comparing the computed coefficient with a theoretical value issued from the table of the critical value of the PCC. Thus, if $PCC(X1, X2)$ is positive and greater than theoretical PCC , this means that X1 and X2 are positively and significatively correlated.

The proposed approach executes in parallel two algorithms studying correlation between latency variation and hop count variation as well as between latency variation and load variation. According to the significance of the correlation coefficient computed between these distributions, our algorithm proceeds in three different scenarios.

1) If latency and hop count distributions are significatively and positively correlated, then we can deduce that hop count variation is one among the factors that causes QoS degradation. Thus, as described in algorithm 2, the first possible source causing the degradation may belong to the mobility category (line 2). We move then to identify the occur source causing hop count variation by looking to nodes positions (line 4,10) and speeds (line 6,12). Node speeds are compared with adaptive threshold updated using the EWMA formula.
2) If latency and load distributions are significatively and positively correlated, then we can notice that load variation is one among the factors that causes QoS degradation.
3) Otherwise, if correlation between all these parameters distributions is not significative, we can deduce that degradation is caused by other sources as described in Table 2.

III. EVALUATION

This section shows the performance evaluation of our approach using simulations with various settings.

A. Simulation Setup

To simulate dynamic network environment, we use JiST/SWANS simulator (Java in Simulation Time/Scalable Wireless Ad hoc Network Simulator) [8]. We resort also to the content based publish/subscribe system named SIENA [9](Scalable Internet Event Notification Architecture) to which we add our monitoring and analysis modules. In all simulations, we use 802.11b as the medium access control protocol and AODV (Ad hoc On-demand Distance Vector) as routing protocol which is a reactive one. During simulation, we measure the latency between two neighboring brokers called B_1 and B_2 at each event delivery.

Simulation experiments consist in two parts. Part 1 provides an explanation of how monitoring and analysis modules succeed in detecting failures and identifying the occur source of failure. In part 2, we estimate our analytical framework overhead.

B. Part 1: Efficiency of our approach

To evaluate the performance of the proposed modules, we consider a network of 200 broker nodes, equipped with 802.11 radio interfaces, roam in a $2000 * 3000m^2$ area. The model for mobility is continuous random walk with pause time equal to $30s$ and max speed equal to $30m/s$. Size of event notifications is about 120 Bytes. The radio transmission range of each node is approximately 300 meters. Each simulation spans about 9000 seconds of simulation time. A first scenario consists in highlighting the monitoring module results. So, we trigger the nodes mobility with a low speed varying from 1m/s up to 5 m/s. The traces files describing QoS parameters issued from monitoring module allow us to draw the curve depicted by Figure 2. In fact, in this figure, our system works well.i.e. a failure free one.

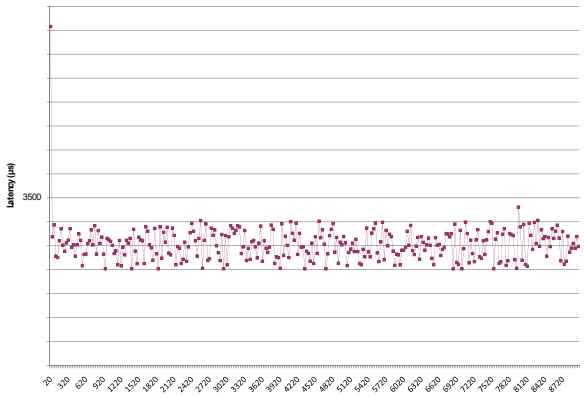

Fig. 2. latency function of simulation Time

We start our evaluation by computing the N value (see section 2), in order to estimate after how much successive violations our analysis module triggers an alarm. The used values and probabilities are deduced from experiments under the simulator. Besides, we consider that the tolerated risk proposed by the user is equal to 0.01%. Thus, we obtain :

$P[anystate \rightarrow LV] = 0,118$

With $P[LOK] = 0,74$,

$P[LV] = 0,236$,

$P[LOK \rightarrow LV] = 0,127$,

and $P[LV \rightarrow LV] = 0,104$

Consequently, $N \geq 2.2$. Thus, N must at least be equal to 3.

At $t = 6350$ s in simulation time, we inject a failure by moving quickly broker B_2 far from his neighbors frequently. This causes a high latency values varying from 95519μ s to 75610μ s. Another failure is injected at t= 7820 s in simulation time, This causes also latency variation from 29029μ s to 81567μ s. As shown in Figure 3, latency oversteps the max threshold value for three consecutive times. Consequently, our failure detector generates alarms.

TABLE III
LOG FILE DESCRIBING SOURCE DEGRADATION IDENTIFICATION(FIRST SCENARIO)

Time detection	7880 s
Number of successive violations	3
latency value at t-2	29029.0 μs
latency value at t-1	36913.0 μs
latency value at t	81567.0 μs
hop count at t-2	5.0
hop count at t-1	5.0
hop count at t	6.0
PCC	0.99027
PCC theoretical value	r = 0.9877
failure category	mobility
cause	hop count variation $Broker_2$ is moving $Broker_2$ speed is high

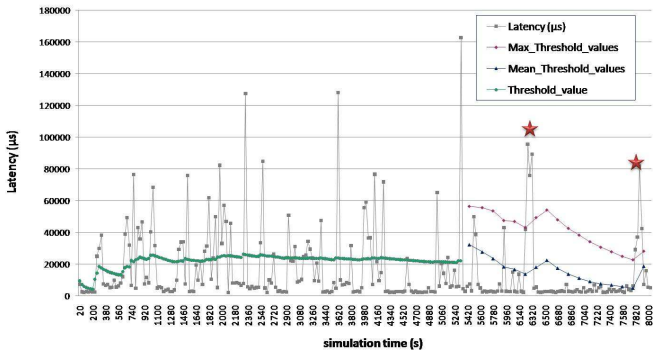

Fig. 3. Reactive analysis

The following log generated by our analysis algorithm describes source degradation identification (Table3).

In second scenario, we inject failure in the system by overloading the broker B_1. This causes also latency variation from 8235μs to 9503μs.

As shown in Figure 4, latency oversteps the max threshold value for three consecutive times. Consequently, our failure detector generates alarms. The following log generated by our

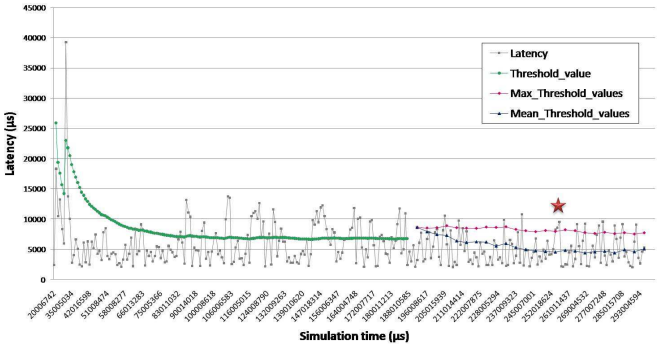

Fig. 4. Reactive analysis

TABLE IV
LOG FILE DESCRIBING SOURCE DEGRADATION IDENTIFICATION(SECOND SCENARIO)

Time detection	256 s
Number of successive violations	3
latency value at t-2	8235 μs
latency value at t-1	8460 μs
latency value at t	9503 μs
load at t-2	164 msg/s
load at t-1	191 msg/s
load at t	233 msg/s
PCC	0.9879
PCC theoretical value	r = 0.9877
failure category	load

analysis algorithm describes source degradation identification (Table 4). We notice from experiments that our algorithm detects QoS degradations and identifies its cause which is a hop count variation in the first scenario and load variation in the second scenario. This illustrates the efficiency of our approach.

C. Part 2:Analytical framework overhead

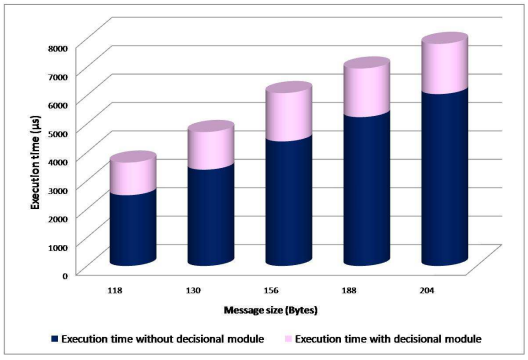

Fig. 5. Overhead introduced by the analytical framework

This part aims to estimate the overhead of the monitoring and the analysis modules.

To reach this goal, we perform simulations experiments with and without the proposed framework and we measure the execution time when sending periodically messages in the system. The noticed overhead represents the time needed to execute the added framework. We can see for instance, as represented in Figure 5, that the latency is about 5840,47μs and 4310,45μs with and without additional components, respectively, when message size . Thus, this overhead stills relatively low compared to the execution time.

IV. RELATED WORK

Failures are inevitable in a distributed system built over a mobile ad hoc network. So that, they should be detected instantly to avoid the whole system failure and to ensure the system survivability. In the following, we present works dealing with failures detection techniques and diagnosis methods used to locate failures.

Several studies addressed the issue of QoS analysis for pub/sub systems in order to detect failures. Researches on failure detection (FD) approaches for pub/sub systems are concentrating on two main topics: FD based on specific messages exchange and FD based on fixed threshold. In the first category, each node sends a specific message such as PING and Heartbeat to its neighbors. If the latter doesn't receive the message within a fixed timeout, then the target node is considered failed. [11] [13] use PING message and [14] [15] rely on Heartbeat message to identify node failure. However using a fixed timeout is not adapted to MANET and so can lead to false detections. Moreover, these techniques don't deal with link failure.

Contrary to the above techniques, our approach doesn't rely on specific messages but rather uses pub/sub messages to detect system failure.

In the second category, the idea consists in comparing QoS parameter value with a fixed threshold. If QoS parameter exceeds the threshold value then an alarm is triggered. In [**?**], authors propose a messaging system called Harmony. Separately, for each message topic, Harmony specifies latency requirements. Therefore, this restrained applying this approach in content based systems. Our approach, however, is not limited to a specific subscription language and is considered as a generic approach. Contrarily to the explained approaches, we use adaptive threshold according to the requirements of applications and the variation of the network. After detecting a failure, we need to reason about source failure which defines diagnosis methods.

Diagnosis methods have been increasingly developed especially for dynamic systems. These methods concern the task to identify and isolate faults in such systems. In the literature, there are two main categories of diagnosis methods: a model based diagnosis and a free model based diagnosis.

Model based diagnosis methods can be categorized into two subclasses: Diagnosis based on quantitative model from the FDI Community [1] relaying on mathematical model describing the behavior of the system, and diagnosis based on qualitative model.

In the second subclass, qualitative models are used such as graphs or if-then rules to describe the system's behavior. These approaches are suitable for systems with qualitative faults [2]

like discrete-event systems, hybrid systems. Although them advantages, model based diagnosis are not usable in some applications where it is difficult or even impossible to obtain the system model. Indeed, only free model based diagnosis are operational in such applications.

Free model based diagnosis are based on information from previous experience and they generally use artificial intelligence [3] and statistical methods [4] to identify faults. In our approach, it's impossible to establish a model that reflects the behavior of the system due to applications requirements and the variation of the network. Thus, we refer to the free model based diagnosis.

V. CONCLUSION

This paper proposes a generic and dynamic framework for QoS aware pub/sub systems deployed on MANET. Different from traditional approaches utilizing a fixed threshold for topics, our framework dynamically update thresholds using statistical methods in order to bypass link failures and increase the chance of delivering messages within the delay requirement. Our study focused on QoS aware pub/sub systems by providing monitoring and analysis modules allowing to continuously supervising the system, detecting QoS degradations and reasoning about source failure. Our approach is evaluated using simulations with various network settings and is shown to present a reasonable performance. Work is also underway to incorporate other QoS parameters to maintain good system performance, while still providing connectivity between brokers. Parallel efforts to evaluate the proposed QoS based framework in a real deployment using android are under development as well.

REFERENCES

[1] M.O. Cordier and P. Dague and M. Dumas and F. Levy and Y. Montmain and M. Staroswiecki and L. Trave-Massuyes, *AI and Automatic Control Approaches of Model-Based Diagnosis : Links and Underlying Hypotheses.*, IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes SAFEPROCESS, 2000.

[2] S. Baniardalani and J. Askari and J. Lunze.*Qualitative Model Based Fault Diagnosis Using a Threshold Level*, International Journal of Control, 2010.

[3] P. chandra and B.V. Sanker and K.S.Sarma,*Neuro-Fuzzy approach fr fault location and diagnosis using online learning system*, Journal of Theoretical and Applied Information Technology, 2010.

[4] SP. Bodik and G. Friedman and L. Biewald and H. Levine and G. Candea and K. Patel and G. Tolle and J. Hui and A. Fox and M. I. Jordan andD. Patterson.*Combining Visualization and Statistical Analysis to Improve Operator Confidence and Efficiency for Failure Detection and Localization.*

[5] R. Khakpour, I. Demeure, *Designing and prototyping an event-based communication system on mobile ad hoc networks*,2008.

[6] G.Giorgi and C.Narduzzi, *Detection of Anomalous Behaviors in Networks From Traffic Measurements*, Instrumentation and Measurement, IEEE Transactions on, 2008.

[7] E. Castillo, *Extreme Value Theory in Engineering*, New York:Academic,New York, NY, USA,1998 .

[8] R. Barr and Z.J. Haas and R.v Renesse, *Scalable wireless ad hoc network simulation,handbook on theoretical and algorithmic aspects of sensor, Ad hoc Wireless, and Peer-to-Peer Networks*, CRC Press, pp. 297-311 (Chapter 19). 2005

[9] A.Carzaniga and D.Rosenblum and A.Wolf, *Design and evaluation of a wide-area event notification service.* ACM Transactions on Computer Systems, 2001.

[10] M. Kim and K. Karenos and F. Ye and R. Fan and H. Lei and K. Shagin, *Efficacy of techniques for responsiveness in a wide-area publish/subscribe system*, Proceedings of the 11th International Middleware Conference Industrial track, 2010.

[11] A. Marco and V. Alessio and T.Giovanni A *Cross-Layer Approach for Publish/Subscribe in Mobile Ad Hoc Networks*, Springer Berli,2005.

[12] http://www.nvcc.edu/home/elanthier/methods/correlation.htm.

[13] M. Mirco and M. Cecilia and H. Stephen. *EMMA: Epidemic Messaging Middleware for Ad hoc networks*. Personal Ubiquitous Comput,2005.

[14] D. Ben Khedher and R. Glitho and R.Dssouli, *A Novel Overlay-Based Failure Detection Architecture for MANET Applications*, 15th IEEE International Conference on, 2007.

[15] H. Jafarpour and S. Mehrotra and N.Venkatasubramanian, *A Fast and Robust Content-based Publish/Subscribe Architecture*, Seventh IEEE International Symposium on, 2008.

[16] M. Lucas and S. Saccucci and Jr. Baxley and V. Robert and H. Woodall and D. Maragh and W. Faltin and W. Fedrick and J. Hahn and T. Tucker and William T. and Hunter and J. Stuart and F. MacGregor and J.Thomas J., *Exponentially weighted moving average control schemes: properties and enhancements*, American Society for Quality Control and American Statistical Association,Alexandria, Va, USA,1990 .

[17] P.Eugster and P.Felber and R.Guerraoui and A.Kermarrec, *The many faces of publish/subscribe*, ACM Comput. Surv, 2003.

[18] H.Ben Halima and K.Guennoun and K.Drira and M. Jmaiel, *Providing Predictive Self-Healing for Web Services: A QoS Monitoring and Analysis-based Approach*, Journal of Information Assurance and Security,2008.

[19] R.Rakotomalala, *Tests de normalite : Techniques empiriques et tests statistiques*, 2008.