Gestion de la qualité de service des systèmes publier/souscrire déployés sur un réseau mobile ad hoc

Imene Lahyani Abdennadher

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *L'Institut National des Sciences Appliquées/(INSA) de Toulouse*

Présentée et soutenue le *17/12/2015* **par :**

IMENE LAHYANI ABDENNADHER

Gestion de la qualité de service des systèmes publier/souscrire déployés sur un réseau mobile ad-hoc

JURY

PHILIPPE ROOSE	Maître de Conférences HDR	Rapporteur
HABIB YOUSSEF	Professeur d'Université	Rapporteur
MICHELLE SIBILLA	Professeur d'Université	Examinateur
NICOLAS VAN WANBEKE	Ingénieur Docteur	Examinateur
CHRISTOPHE CHASSOT	Professeur d'Université	Co-directeur de thèse
KHALIL DRIRA	Directeur de Recherche	Co-directeur de thèse
MOHAMED JMAIEL	Professeur d'Université	Co-directeur de thèse

Écoles doctorales :

MITT : Domaine STIC : Réseaux, Télécoms, Systèmes et Architecture

EDST : Ecole Doctorale Sciences et Technologies

Laboratoires de Recherche :

LAAS-CNRS - (UPR 8001)

ReDCAD-ENIS - (LR13ES26)

Directeurs de Thèse :

Christophe Chassot, *Khalil Drira* et *Mohamed Jmaiel*

Rapporteurs :

Habib Youssef et *Philippe Roose*

Dédicace

À tous ceux qui comptent pour moi...

Remerciement

C'est avec un grand plaisir que je réserve cette page en signe de gratitude et de reconnaissance à tous ceux qui ont assisté à ce travail.

Je voudrais tout d'abord remercier mes directeurs de thèse M. *Mohamed Jmaiel*, professeur à l'Ecole Nationale d'Ingénieurs de Sfax, M. *Khalil Drira*, directeur de Recherche CNRS au LAAS de Toulouse, et M. *Christophe Chassot*, professeur à l'INSA de Toulouse, qui n'ont pas épargné le moindre effort dans leurs encadrements de cette thèse. C'est grâce à leur aide précieuse et leurs conseils que j'ai pu réaliser ce travail.

Je tiens à témoigner ma gratitude et mes vifs remerciements à M. *Habib Youssef*, professeur à ISITC de Hammam Sousse, et M. *Philippe Roose*, professeur à l'IUT de Bayonne/LIUPPA-T2, d'avoir accepté d'être les rapporteurs de ce travail.

J'exprime également ma profonde gratitude à Mme. *Michelle SIBILLA*, professeur à l'UPS de Toulouse et à M. *Nicolas Van Wambeke*, ingénieur en Communications Aéronautiques à Thales Alenia Space de Toulouse, qui ont accepté d'être membre de mon jury de thèse.

Effectuer cette thèse au sein des deux équipes de recherche LAAS-CNRS et ReDCAD (ENIS) m'a donné l'occasion de faire la connaissance de plusieurs amis. Le temps passé avec ces amis ainsi que les nombreuses discussions, souvent vives et intéressantes, ont enrichi mes connaissances.

Il est donc indispensable de ne pas rater cette occasion pour exprimer ma reconnaissance aux membres du laboratoire de recherche ReDCAD-Sfax et du laboratoire LAAS-CNRS de Toulouse pour leurs efforts qui ont guidé mes pas et enrichi mes connaissances.

Résumé

Les réseaux MANET, grâce aux avancées technologiques ont vécu une évolution très importante. La mobilité continue et imprévue des nœuds dans ces réseaux crée un changement dynamique de topologie. Ce déplacement a bien évidemment un impact sur la qualité de service (QdS) du réseau. Dans cette situation, fournir une bonne QdS demeure un défi réel. Les systèmes publier/souscrire, grâce au découplage tridimensionnel, ont été proposés et ont assuré la connectivité entre les parties communicantes dans le réseau mobile.

L'objectif de cette thèse est la gestion de la QdS des systèmes publier/souscrire sur MANET. Ceci provoque un maintien permanent de la connectivité entre les parties communicantes du système.
Plus précisément, les contributions de cette thèse, sont organisées autour de deux grands axes relatifs aux modules de *monitoring* et d'analyse de la QdS de ces systèmes dans un contexte ad-hoc. Ceci englobe les étapes de collecte des paramètres de QdS au cours du fonctionnement du système, et de détection des dégradations pouvant l'affecter.

Contrairement à ce qui existe dans la littérature, notre module d'analyse, basé sur des méthodes statistiques, se base sur des seuils adaptatifs qui tiennent en comptes de la dynamicité du réseau et des changements affectant le contexte ad-hoc. En outre, le module d'analyse proposé offre à l'utilisateur une variété de choix entre une analyse réactive, proactive et hybride. Notre module permet aussi une localisation des pannes une fois détectées ou prédites. Des expérimentations menées à l'aide du simulateur *Jist/Swans* ont montré l'efficacité des modules développés. De plus, une mesure des paramètres de performance du système avant et après introduction des modules développés a montré leur efficacité. Finalement, une étude de la complexité spatiale et temporelle du module analytique proposé a été réalisée.

Abstract

Mobile ad-hoc networks (MANET) is a set of nodes, communicating meaning wireless channel without any centralized control. The main characteristic of MANET is the frequent mobility of its nodes leading to some dynamic changes of topology. Nodes mobility in such networks introduces possible disconnections between adjacent nodes, and more generally quality of service (QoS) degradation issues that do not fit well with QoS requirements of QoS sensitive applications.

Publish/subscribe communicating system has been proposed at the middleware layer in order to communicate mobile entities in MANET. Such system is a decoupled interaction between publishers (or producers) and receivers (or consumers).

Our goal in this thesis is to provide a QoS management for publish/subscribe systems when deployed on MANET while assuming a best effort flux at the routing layer.
This thesis proposes an analytical model for latency aware publish/subscribe systems on mobile ad-hoc networks. The proposed approach combines both proactive and reactive statistical analysis.
On the one hand, the reactive analysis, suitable for multimedia applications, detects failures by approximating latency series with Gumbel distribution. On the other hand, the proactive analysis, suitable for crisis management applications, forecasts failures occurrence relying on Auto Regression or Auto Regressive Integrated Moving Average Formulas. Finally, a hybrid analysis was proposed by dynamically switching from reactive to predictive forms of analysis whenever quality of service violations are noticed.
In order to extract failure cause, we refer to the correlation method once failure was detected or predicted. Simulations done, using Jist/Swans simulator, under different scenarios proved the efficiency and accuracy of the proposed scheme.

Table des matières

Table des figures

Liste des tableaux

Introduction générale

L'évolution de la technologie dans le domaine des communications sans fil, a permis aux utilisateurs, munis d'un équipement mobile tel qu'un PDA ou un portable, d'accéder continuellement à l'information indépendamment des deux facteurs : temps et lieu.
L'environnement mobile ainsi crée, en particulier le réseau mobile ad-hoc, offre une grande flexibilité d'emploi. En effet, l'utilisateur n'est plus restreint à une localisation fixe, mais il bénéfice d'une libre mobilité tout en restant connecté au réseau. Ceci a poussé les développeurs à mettre en place des applications distribuées bénéficiant de ces nouvelles technologies à la pointe. Cependant, la nature dynamique de ces réseaux mobiles et de ses ressources limitées représentent un défi pour ces applications. Ceci inclut les capacités limitées des nœuds du point de vue batterie et mémoire ainsi que les variations fréquentes des ressources du réseau et de la connectivité des liens.

Afin d'assurer la communication entre les entités mobiles d'une manière efficace, dynamique et flexible, les systèmes publier/souscrire ont répondu à ces exigences et ont pu remédier aux limites des modèles de communications traditionnels. En effet, le modèle *client/serveur* et le modèle *push/pull* ne sont pas suffisamment prometteurs pour la nouvelle génération d'applications distribuées à large échelle. En effet, ces deux modèles exigent un couplage fort entre les entités communicantes ce qui s'oppose à la scalabilité et empêche le développement des applications distribuées à large échelle. En outre, les modèles traditionnels précités souffrent d'un manque d'extensibilité du fait qu'il est pénible d'ajouter ou de supprimer des entités à la communication. Les systèmes publier/souscrire sont apparus donc pour adresser les problèmes de scalabilité et d'extensibilité des applications à large échelle.

D'ailleurs, ces systèmes appelés aussi systèmes basés évènements forment un support

de communication totalement asynchrone. Ils offrent un découplage tridimensionnel, à savoir un découplage dans le temps, dans l'espace et un découplage de synchronisation. Cette caractéristique présente une motivation très importante pour déployer les systèmes publier/souscrire sur les réseaux mobiles ad-hoc (MANET). Toutefois, ceci présente un défi en plein essor.

En effet, adapter ces systèmes sur des réseaux mobiles ad-hoc engendre des problèmes liés à la qualité de service. Ces problèmes sont dus essentiellement à la mobilité fréquente et imprédictible des nœuds engendrant une rupture continue des liens de communication.

En effet, un lien entre deux nœuds logiques (appelés *brokers*), initialement court, peut devenir assez long à cause de la mobilité d'un ou des deux entités. Encore plus pire, un des deux brokers peut se trouver hors de la portée de ses voisins, ce qui engendre la rupture totale de la communication.

Les couches basses, plus précisément la couche réseau, vise à assurer une stratégie qui garantit, la connexion entre n'importe quelle paire de nœuds faisant partie du réseau. La stratégie de routage prend en considération les changements de la topologie ainsi que les autres caractéristiques du réseau ad-hoc pour assurer la communication entre les entités. Par contre, il n'y a pas de garantie de trouver la route, et même si elle a été trouvée, il se peut que cette route ne satisfait pas les exigences des applications en termes de qualité de service.

À cet issu, la couche middleware doit fournir des solutions efficaces à ces problèmes et doit réagir afin d'assurer la qualité de service des applications en vue des mobilité des nœuds. L'objectif de cette thèse est donc d'introduire une solution au niveau middleware permettant d'assurer la qualité de service des systèmes publier/souscrire sur MANET.

L'approche générale proposée s'inscrit dans le cadre des systèmes autonomes qui sont capables de s'auto-contrôler et de réagir façe aux problèmes sans intervention humaine. Ce processus, selon IBM, met en œuvre quatre étapes à savoir : *monitoring*, analyse, planification et exécution.

Notre travail cible exactement les deux premières phases du processus d'auto-adaptation et propose un modèle analytique permettant au système de se contrôler au cours de son

exécution et de détecter, voire prédire, les dégradations de QdS pouvant l'affecter.

Une étude exhaustive de l'existant a montré que les systèmes publier/souscrire adressent le problème de la gestion de la QdS dans chacune des phases du processus d'auto-adaptation. En effet, au niveau de la phase d'exécution, deux catégories de systèmes visent à remédier aux défaillances subies par le système.

Les systèmes appartenant à la première catégorie [CP06, DSM07, PEKS07] apportent des actions de reconfiguration, essentiellement des actions de substitution de liens, dans le but de réparer le système et rétablir son fonctionnement.

Ayant le même but que ces systèmes, le système [MB08] appartenant à la deuxième catégorie, offre des méthodes optimisées de routage au niveau logique vis à vis de la fiabilité des liens afin d'utiliser les routes les plus fiables et échapper des pannes pouvant se produire.

Les actions de reconfiguration introduites par ces systèmes présentent quelques insuffisances vu qu'elles sont bloquantes et qu'elles engendrent la perte de messages au cours de la réparation.

D'autres systèmes publier/souscrire ont adressé la gestion de la QdS au niveau des phases de monitoring et d'analyse.

Une étude détaillée de ces travaux nous a amené à identifier les techniques de monitoring et d'analyse proposées par ces systèmes.

D'une part, certains de ces systèmes adoptant une stratégie réactive [MMH04], utilisent des messages spécifiques de type *Ping* ou *Heartbeat* afin de calculer les paramètres de QdS traités ou afin de détecter les pannes dans le système. Ceci apporte une surcharge au système et crée un trafic supplémentaire qui n'est pas adéquat surtout dans un réseau ad-hoc à ressources limitées.

D'autres systèmes [JFF07], se basent sur des comparaisons des paramètres de QdS, une fois identifiés, par rapport à des seuils fixes et figés. Cependant, le recours à des seuils fixes pourrait être une source de manque de précision dans l'analyse de l'état du système.

En effet, les performances du système varient en fonction de la dynamicité du réseau et des fluctuations pouvant toucher ses performances.

D'autre part, les systèmes [CS05, MA05, KKY+10], adoptent une analyse préventive afin de prédire les défaillances en comparant les valeurs réelles de QdS par rapport à des valeurs estimées. Ceci a pour objectif d'anticiper les pannes et de prédire leurs occurrences afin de prendre les mesures correctives nécessaires.

Notre contribution dans cette thèse, cible les deux premières phases du processus d'auto-adaptation à savoir *monitoring* et analyse, et propose un module analytique orienté QdS.

Ainsi, nous proposons de mettre en place une stratégie innovante permettant de mesurer instantanément la QdS et de détecter, voire prédire, les pannes imminentes au sein du système publier/souscrire.

Pour ce faire, notre approche se base sur une architecture distribuée qui supervise et enregistre la latence comme paramètre de QdS en utilisant les messages propres au système publier/souscrire, dans une première étape, et analyse ces paramètres afin de caractériser l'état du système tout en faisant recours à des seuils dynamiques qui s'adaptent aux variations caractérisant le contexte ad-hoc.

Le module d'analyse proposé n'impose pas une forme d'analyse bien précise (réactive ou proactive). Par contre, il offre au développeur une variété de choix :

◇ une analyse réactive : l'approche proposée dans ce cadre vise à détecter les dégradations de QdS par comparaison des valeurs de latence avec des seuils adaptatifs. Le calcul des seuils débute par une approximation de la série formée par les valeurs de latence à des distributions empiriques à savoir la loi de *Gumbel*. Ensuite, à l'aide de la formule de la *Moyenne Mobile Exponentielle*, la valeur du seuil initialement calculée est mise à jour dynamiquement.

L'analyse réactive se base aussi sur des chroniques temporelles permettant de surveiller l'évolution des valeurs de latence pour éviter de considérer les violations transitoires de QdS. Donc, une dégradation de QdS est détectée après M dépassements successifs des valeurs de latence.

Afin de calculer une valeur pertinente de M, nous avons utilisé le fait que la probabilité de panne doit être inférieure ou égale à une certaine valeur spécifiée par l'utilisateur et qui représente le risque toléré par l'utilisateur.

◇ une analyse proactive : cette forme d'analyse a pour finalité de prédire les pannes pouvant affecter le système. Elle s'appuie sur une estimation de la valeur de la latence à l'aide de la Méthode d' *Auto Regression Linéaire* et sur une estimation de la valeur du seuil. La comparaison de ces deux valeurs estimées permet de prédire les pannes affectant éventuellement le système.
Afin d'améliorer la précision du prédicteur proposé, nous avons eu recours, dans une deuxième étape, à la méthode ARIMA *(Auto Regressive Integrated Moving Average Formula)* qui permet de calculer les termes d'erreurs finement dans le but de minimiser les erreurs de prédictions.

◇ une analyse hybride : cette analyse est dite hybride du coup qu'elle combine les deux formes d'analyse pré-décrites. Le système déclenche d'abord une analyse réactive durant laquelle il s'auto contrôle. S'il note une violation de la contrainte spécifiée par l'utilisateur, le système déclenche une analyse proactive. Une fois que le système se stabilise, il revient de nouveau vers l'analyse réactive et le cycle se répète.

Ce processus se poursuit par l'identification de la source de la dégradation. Le diagnostic est déclenché une fois que le module d'analyse a détecté ou a prédit une dégradation. Dans ce cadre, nous avons commencé par énumérer les causes possibles de la dégradation. Nous avons identifié deux catégories de causes possibles à savoir la mobilité qui se traduit par une variation du nombre de sauts, et la charge du lien logique. Afin d'identifier la cause réelle de la dégradation, nous avons eu recours à la méthode de la corrélation qui permet de mesurer l'intensité de la liaison entre la variation de la latence d'une part et la variation au niveau du nombre de sauts ou la charge d'autre part.

Des expérimentations menées à l'aide du simulateur *Jist/Swans*, sous différentes conditions, ont montré l'efficacité des modules développés. De plus, une mesure des paramètres de performance du système avant et après l'introduction des modules développés a montré leur efficacité. Finalement, une étude de la complexité spatiale et temporelle du module analytique proposé a été réalisée .

Le rapport se compose de quatre chapitres : dans le premier chapitre, nous commençons par introduire les notions de bases relatives aux systèmes publier/souscrire et aux réseaux

mobiles ad-hoc. Ensuite, nous détaillons les problèmes de qualité de service des systèmes publier/souscrire déployés sur les réseaux MANET. Par la suite, nous passons en revue des systèmes publier/souscrire traitant la problématique mentionnée.

Dans le deuxième chapitre, nous étudions les systèmes publier/souscrire adressant la gestion de la QdS dans chacune des phases de *monitoring*, d'analyse et d'exécution. Nous détaillons ensuite les techniques de *monitoring* et d'analyse utilisées par ces systèmes.

Nous introduisons, dans le troisième chapitre, notre modèle analytique d'assurance de la QdS des systèmes publier/souscrire déployés sur MANET. Nous présentons ainsi toutes les fonctionnalités offertes par notre modèle. Puis, nous détaillons le module de *monitoring*, d'analyse et les méthodes statistiques dont ils reposent. Finalement, nous détaillons l'approche de diagnostic proposée.

Dans le quatrième chapitre, nous illustrons notre approche par deux études de cas. La première représente une application de gestion de crise, alors que la deuxième représente une application multimédia. Les résultats de simulations menées à large échelle sont présentés et analysés dans chacune de ces études de cas.

Nous clôturons ce document par une conclusion générale tout en rappelant les contributions majeures de la thèse. Ces contributions aboutissent à d'autres axes de développement intéressants présentant les perspectives de cette thèse.

CHAPITRE 1

Prérequis et problématique

1.1 Introduction

Le déploiement croissant d'applications mobiles, suite aux avancées technologiques a facilité la création des réseaux sans fil notamment le réseau MANET [CM99]. La topologie de ces réseaux ne bénéficie d'aucune infrastructure préexistante.
L'approvisionnement en termes de qualité de service (QdS) est l'une des problématiques majeures confrontées dans les MANET. En effet, l'impact de la mobilité engendre par excellence un changement fréquent de la topologie du réseau. Ce qui influe sur la qualité de transmission des données entre sources et destinations. Ajoutons à tout cela les ressources limitées des nœuds qui peuvent engendrer des déconnexions imprévues.

Le paradigme publier/souscrire, caractérisé par un découplage tridimensionnel, a fait ses preuves comme étant un moyen de communication sur les réseaux en particulier les MANET[CB12]. Notre étude se focalise donc sur les problèmes de QdS affectant les systèmes publier/souscrire sur un réseau MANET dus principalement aux caractéristiques de ces réseaux. En effet, les problèmes critiques liés à la mobilité fréquente des nœuds, notamment lors de la transmission de messages à travers le réseau ont inévitablement un impact sur la qualité de service.

Ce chapitre introduit dans une première partie les réseaux mobiles ad-hoc (MANET) ainsi que les problèmes de QdS confrontés par ces réseaux. Nous clôturons cette première partie par une étude sur les différents niveaux de gestion de la QdS dans les MANET.
La deuxième partie de ce chapitre est consacrée à présenter les systèmes publier/souscrire, leurs caractéristiques, ainsi que les différents types de topologies formées par ces systèmes. Finalement, nous nous focalisons sur une description de la problématique étudiée à travers cette thèse, et nous illustrons nos idées par une étude de cas.

1.2 Contexte : Réseau mobile ad-hoc (MANET)

Dans cette section, nous introduisons les réseaux mobile ad-hoc, leurs caractéristiques primordiales, ainsi que les problèmes de QdS survenant dans ce type de réseau.

1.2.1 Définition

Un réseau mobile ad-hoc (MANET) [1], est formé d'équipements mobiles qui s'interconnectent moyennant une technologie sans fil dans le but de former un réseau temporaire sans aucune infrastructure fixe.

Le groupe MANET de l'IETF [2] fournit une définition précise en introduction de la RFC [CM99] : « Un réseau mobile ad-hoc est constitué de plates-formes, actuellement appelées nœuds, qui sont libres de se déplacer arbitrairement. Un réseau mobile ad-hoc est donc un système autonome de nœuds mobiles. Ce système peut fonctionner soit d'une manière isolée, soit en s'interfaçant à des réseaux fixes à l'aide de passerelles. »
Devenant de plus en plus populaires, les réseaux MANET incitent de plus l'attention des chercheurs. Leurs usages augmentent continuellement grâce à la rapidité et à la simplicité de leurs déploiements. Initialement utilisés dans des applications militaires, ils ont rapidement gagné de nouveaux champs d'application où l'infrastructure de communication n'est plus disponible tels que les opérations tactiques, les missions d'exploration, les opérations de secours, etc [RT99].

La Figure 1.1 illustre la structure d'un réseau MANET.

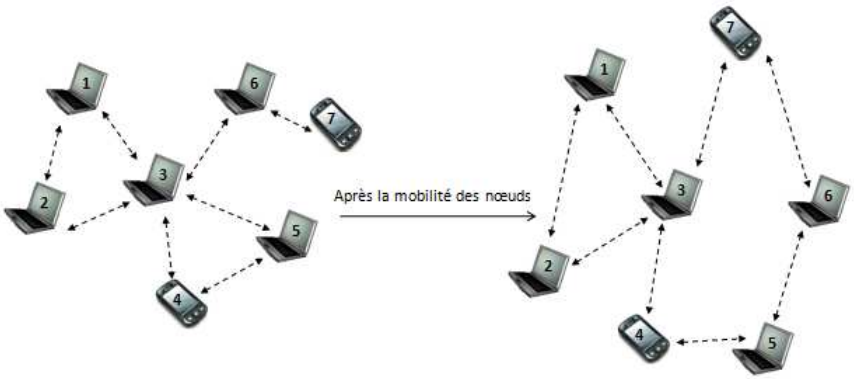

FIG. 1.1 – Réseau mobile ad-hoc

Cette Figure présente un réseau mobile ad-hoc formé de 7 nœuds. Le nœud 2 peut

[1] Mobile Ad-hoc NETWORK
[2] Internet Engineering Task Force

communiquer directement avec les nœuds 1 et 3. Pour atteindre d'autres destinations, par exemple le nœud numéro 5, le nœud 2 doit s'appuyer sur un ensemble de nœuds intermédiaires qui vont, eux mêmes, relayer les messages de proche en proche jusqu'à aboutir à la destination (nœud 5).
Contrairement aux réseaux sans fil à infrastructure, la où le maintien de la connectivité est assurée à travers des équipements spécifiques, dans les réseaux MANET, chaque nœud devra identifier et calculer les routes pour aboutir aux destinations qu'il souhaite atteindre.

Dans la partie suivante, nous présentons les caractéristiques des MANET les distinguant des autres réseaux.

1.2.2 Caractéristiques

Selon [Mer05], les caractéristiques principales des réseaux MANET sont :

- Absence d'infrastructure : Une caractéristique majeure qui distingue les réseaux ad-hoc des autres réseaux mobiles est l'absence d'infrastructure préexistante. Par ailleurs, les hôtes en se déplaçant, sont responsables d'établir et de maintenir la connectivité du réseau de manière continue. Ainsi, chaque nœud agit en tant que routeur constituant ainsi un relai de communication [KTKS09].
- Mobilité des nœuds : La mobilité continue des nœuds engendre un changement fréquent et dynamique de la topologie. En effet, un nœud peut rejoindre le réseau, se déplacer ou même se détacher du réseau à tout moment. Par conséquent, la topologie du réseau peut changer à des instants imprévisibles.
- Contraintes d'énergie : Dans un réseau ad-hoc, les nœuds mobiles fonctionnent principalement à l'aide de batteries. Ceci restreint la durée de vie d'un nœud à la capacité de sa batterie.
- Bande passante limitée : Le réseau MANET repose sur un médium de communication partagé entre les différents nœuds faisant partie du réseau. Ceci rend la bande passante associée à chaque nœud faible et modeste.
- Hétérogénéité des nœuds : Un nœud dans un réseau ad-hoc peut être équipé d'une

ou plusieurs interfaces radio ayant des capacités de transmissions diverses. En outre, les nœuds formant le réseau MANET sont inéquivalents du point de vue capacité de traitement, taille, mobilité ....

– Sécurité limitée : Les réseaux ad-hoc sont sensibles aux problèmes de sécurité. En effet, un nœud malicieux peut s'insérer dans le réseau et pourra intercepter tout ce qui se passe dans le médium physique ce qui s'oppose fortement à la sécurité.

Outre les déconnexions fréquentes et accidentelles causées par la portée limitée et par l'instabilité des liaisons sans-fil, les nœuds du réseau MANET peuvent à tout instant quitter ou rejoindre le réseau. Les conséquences combinées de tous ces comportements et de la mobilité des nœuds se traduisent par des changements fréquents de la topologie. Ces derniers ont un impact sur la qualité de service (QdS) offerte par ces réseaux.

1.2.3 Applications des MANET et besoin en QdS

Divers types d'applications peuvent s'exécuter sur des réseaux MANET. Cependant, ces applications imposent des contraintes et des exigences en termes de QdS que le réseau doit satisfaire.

1.2.3.1 Applications des MANET

Les réseaux ad-hoc étaient dédiés au départ au domaine militaire. Cependant, suites aux avancées considérables des recherches dans le domaine des réseaux et à l'émergence des technologies sans fil, d'autres types d'applications civiles sont apparues telles que :

– Les services d'urgence : ceux-ci incluent les opérations de secours des personnes suite à des catastrophes naturelles tels que les tremblements de terre, les incendies, les inondations, etc.
– Les applications collaboratives : dans le cadre d'une conférence ou d'une réunion de travail par exemple, les participants peuvent échanger des informations et des messages via des communications au sein du réseau MANET.
– Les réseaux de capteurs : dans des situations environnementales, telle que les activités

de la terre, il est possible de faire recours à ces réseaux en exploitant des dispositifs appelés capteurs.

Cette large émergence des réseaux MANET et des applications temps réel, nous incite à bien réfléchir à résoudre les problèmes de QdS affectant ce type de réseau tels que la prolongation du délai d'acheminement des messages ou même leurs pertes.

1.2.3.2 Notion de QdS

La qualité de service est un concept de gestion ayant pour objectif d'optimiser les ressources d'un réseau ou d'un processus et de garantir de bonnes performances aux applications. Particulièrement, au niveau réseau, la QdS est la capacité de véhiculer un trafic donné dans de bonnes conditions, notamment en termes de disponibilité, débit, délai de transmission, gigue, taux de perte de paquets, etc.

Selon la recommandation ITU-X.902 [3], la QdS est définie comme « un ensemble d'exigences dans le comportement collectif d'un ou de plusieurs objets ».

Selon l'IETF [4], la QdS est « un ensemble de besoins à satisfaire par le réseau afin de transmettre des informations de l'expéditeur vers le destinataire ». La QdS est donc l'aptitude d'un service à répondre convenablement à des exigences, visant à satisfaire ses usagers.

La RFC 2386 [CNRS98] définit aussi la QdS comme étant « l'ensemble des besoins à assurer par le réseau pour le transport d'un trafic d'une source vers une destination ».

1.2.3.3 Paramètres de QdS dans les MANET

Pour supporter la QdS, il existe un certain nombre de paramètres usuels bien spécifiques pour les réseaux MANET. En effet, toute application se déroulant sur MANET désire avoir des garanties en termes de gigue, de débit de transmission, de débit, de bande passante, de taux de perte etc. Dans ce qui suit, nous définissons les paramètres de QdS dans les

[3] The International Telecommunication Union (ITU) standard X.902, Information technology - Open distributed processing - Reference Model

[4] Internet Engineering Task Force

MANET.

∗ Le délai : selon (RFC 2679) [AKZ99] le délai est le temps séparant l'envoi du premier bit d'un paquet et le temps de réception du dernier bit de ce même paquet par la destination. Si le paquet n'était pas reçu au moment de l'expiration du temporisateur, le paramètre délai sera infini. Dans ce cas, il y aura une perte de connectivité.

∗ La gigue : est définie comme la variation du délai de réception des paquets. Du coté de l'émetteur, les paquets sont envoyés en flot tout en respectant une période de temps fixe. Dans ce cas, la gigue est nulle. A cause de la congestion survenant dans le réseau, ce flot de paquets peut subir des perturbations et le délai entre les paquets devient non constant.

∗ La bande passante disponible entre deux entités mobiles : est le débit maximal pouvant être émis entre deux nœuds (source et destination) sans causer aucune dégradation des flux sur le réseau.

La passante disponible = capacité de la bande passante - bande passante utilisée.

avec :

- La capacité de la bande passante : est la quantité maximale de données par unité de temps pouvant être supportée par un lien.
- La bande passante utilisée : est la capacité globale consommée à un instant sur un lien ou un chemin.

∗ Le taux de perte : sur le réseau, un paquet qui n'arrive pas à la destination sera considéré perdu. De ce fait, le taux de perte des paquets (RFC 2680)[SFV+08] sur un lien est considéré nul si tous les paquets émis par la source sont reçus par la destination dans le temps approprié.

Afin de prendre en compte ces paramètres et de remédier au problème de QdS, divers travaux de recherche sur les réseaux ad-hoc se sont intensifiés depuis des années. En effet, les applications déployées sur un réseau, plus particulièrement MANET, spécifient des contraintes et des exigences que le réseau doit satisfaire pour que leurs exécutions correspondent à un niveau de qualité bien défini. Du point de vue réseau, ces critères se traduisent souvent en termes de paramètres de QdS. La prise en compte de ces paramètres

peut s'effectuer dans différents niveaux du modèle OSI [5]. Nous présentons dans ce qui suit, deux niveaux de gestion de la QdS dans les MANET.

1.2.3.4 Niveaux de gestion de la QdS dans les MANET

Différentes solutions sur des niveaux différents du modèle OSI ont étés proposées afin de remédier au problème de QdS dans les réseaux MANET.

- 1. Gestion de la QdS au niveau routage

  Au niveau routage (niveau 3 du modèle OSI), divers protocoles de routage traitant la QdS conçus comme étant des extensions optionnelles des protocoles appelés best-effort, ont été proposés [MKSM08, HT07]. En tenant compte d'un ou de plusieurs paramètres de QdS, ces protocoles cherchent à trouver la meilleure route qui pourrait transférer les données entre sources et destinations. De ce fait, les nouvelles recherches travaillent sur l'extension des protocoles de routage existants afin de permettre la recherche de chemins en tenant compte de plusieurs paramètres de QdS à la fois [Pet11].

  Les protocoles de routage se subdivisent principalement en trois grandes catégories :

  - Routage proactif : les protocoles de routage proactif essaient de préparer et de maintenir l'ensemble des meilleurs chemins existants vers toutes les destinations possibles. Ces routes seront sauvegardées d'une façon permanente à l'aide d'un échange permanent des messages de mise à jour des chemins. Le protocole OLSR [MK12] est l'exemple le plus connu de ces protocoles.
  - Routage réactif : les protocoles de routage réactif déclenchent une procédure de recherche de route à chaque fois que le besoin se présente. Les protocoles de routage appartenant à cette classe maintiennent les routes à la demande. A chaque fois que le réseau a besoin d'une route, une procédure de découverte globale de routes est lancée. AODV [LYW13] est un exemple de protocole réactif.
  - Routage hybride : cette classe est une combinaison entre les deux classes

[5] Open System Interconnection model

prédécrites : les routes vers les nœuds relativement proches sont obtenues par un schéma proactif, alors que celles vers les nœuds lointains sont calculées par schéma réactif. ZRP [MK12] est un exemple de protocole hybride.

Ces familles de protocoles ont été augmentées avec des contraintes de QdS, tels que la bande passante, le délai de transfert, le coût, la batterie. Les protocoles de routage avec QdS sont appelés donc à fournir, non seulement les routes disponibles entre source et destination, mais aussi celles qui satisfont les contraintes de QdS souhaitées. Par exemple, le protocole QoS-AODV [Pin09], apparu comme extension du protocole AODV, a eu comme objectif de choisir les routes qui offrent un délai minimum pour transmettre une information d'une source vers une destination tout en garantissant un minimum de bande passante. Les tables de routage ont été donc étendues par les sources qui demandent des garanties de délai avec celles exigeant des garanties de bande passante.

Le protocole Cedar [SSB99] est aussi un exemple de protocole de routage réactif qui assure la QdS des liens en termes de bande passante.

– 2. Gestion de la QdS au niveau middleware

Toutefois la gestion de la QdS dans les MANET se limite au niveau routage, quelques applications ont introduit une nouvelle couche (couche middleware) au dessus de celle du routage pour remédier à des situations non gérables par telle couche.

Au niveau de cette couche, divers paradigmes de communication ont été proposés [Col14] tels que les modèles *remote procedure calls*, *tuple spaces* et *publish/subscribe*.

Le découplage entre les parties communiquantes est une caractéristique primordiale qui servirait à déterminer si ces paradigmes pourraient être adoptés sur un réseau mobile ad-hoc. En effet, si le paradigme offre un découplage temporel, c'est que les deux parties communicantes n'ont pas besoin d'être connectées en même temps. De ce fait, le paradigme de communication utilisé pourrait tolérer les déconnexions et les changements de topologie qui surviennent fréquemment dans le réseau MANET et il sera capable d'envoyer les messages aux destinations dès qu'ils seront disponibles.

Le paradigme *remote procedure calls* est caractérisé par un couplage temporel du coup

que les méthodes invoquées doivent être disponibles au moment de l'appel par la machine distante. Au contraire, les deux paradigmes *tuple spaces* et *publish/subscribe* permettent un découplage temporel.
Une étude comparative [CB12] entre ces modèles de communication a montré que les systèmes publier/souscrire ont fait leur preuve quant à divers paramètres de performance. Les systèmes publier/souscrire ont été donc bien évalués comme étant une façon efficace de connecter les entités constituant le réseau MANET et de remédier à certains problèmes de QdS.
Dans cette optique, certains travaux se sont focalisés sur les systèmes publier/souscrire sur MANET et ont essayé de résoudre les problèmes liés à la prolongation des délais d'acheminement des messages au niveau middleware.
Il semble donc intéressant d'étudier ces systèmes et dans quelle mesure ils pourraient assurer la connectivité entre les parties communicantes dans un réseau MANET.

1.3 Systèmes publier/souscrire

Dans ce qui suit, nous présentons les systèmes publier/souscrire, leurs architectures ainsi que leurs caractéristiques.

1.3.1 Définition et architecture

Les systèmes publier/souscrire, ou encore systèmes basés évènements, représentent un nouveau paradigme de communication offrant aux fournisseurs de l'information (les producteurs) la possibilité d'envoyer des notifications aux récepteurs de l'information (les consommateurs) dans un réseau à large échelle suite à des souscriptions exprimées à priori par ces derniers.

Les systèmes publier/souscrire sont structurés autour d'un réseau distribué de serveurs appelés service d'évènements comportant des *brokers* jouant le rôle d'intermédiaires entre les producteurs et les consommateurs. Le service d'évènement est donc responsable de faire

la correspondance entre les souscriptions des consommateurs d'une part et des évènements publiés par les producteurs d'autre part. La Figure 1.2 présente la structure d'un système publier/souscrire.

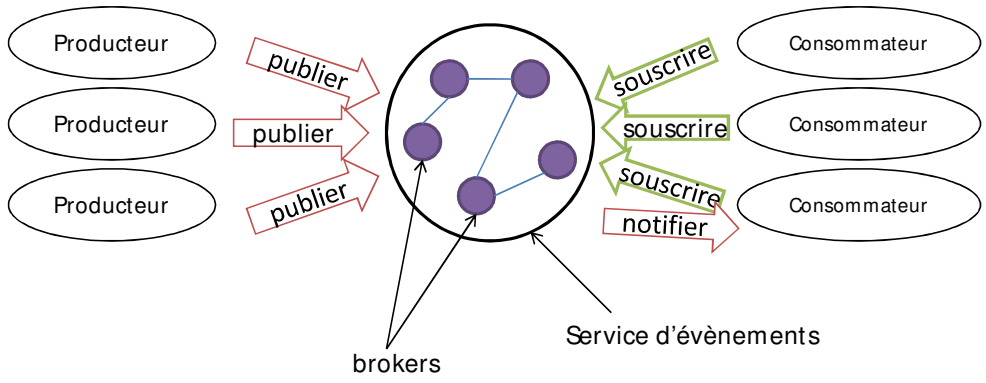

FIG. 1.2 – Système publier/souscrire

Un système publier/souscrire met en place principalement un ensemble d'opérations décrites ci-dessous :

- l'opération publier : déclenchée par le producteur d'évènements pour rendre disponible sur le service d'évènements l'information qu'il possède.
- l'opération souscrire : utilisée par le consommateur pour exprimer ses demandes auprès du réseau de *brokers*.
- l'opération notifier : exécutée au niveau des *brokers* faisant partie du service d'évènements dès qu'il existe une correspondance entre la souscription du consommateur et l'évènement du producteur. Cette opération est clôturée par la livraison des notifications vers le consommateur adéquat.

1.3.2 Caractéristiques des systèmes publier/souscrire

Le modèle publier/souscrire offre un découplage tridimensionnel, à savoir un découplage dans le temps, dans l'espace et de synchronisation.

1.3.2.1 Découplage dans le temps

Les producteurs et les consommateurs n'ont pas besoin d'être connectés en même temps. En effet, comme le montre la Figure 1.3, les producteurs peuvent produire des évènements pendant que les consommateurs soient déconnectés. Inversement, les consommateurs peuvent être notifiés même si les producteurs ne sont pas attachés au système.

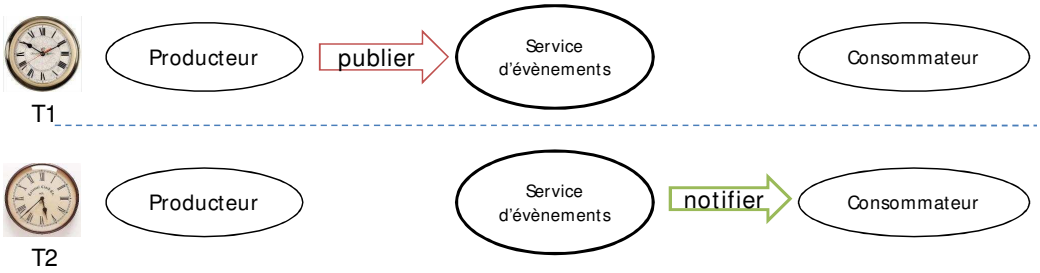

FIG. 1.3 – Découplage dans le temps [EFGK03]

1.3.2.2 Découplage dans l'espace

Les consommateurs et les producteurs n'ont pas intérêt à se reconnaître les uns les autres (Figure 1.4). En effet, les consommateurs accèdent au service d'évènements afin de consommer les messages sans connaître ni la source de ces évènements (les producteurs), ni le nombre de producteurs. Ces derniers, de leurs parts produisent des évènements sans connaître ni les récepteurs de ces messages, ni le nombre des consommateurs intéressés. On parle donc d'une communication anonyme (Figure 1.4).

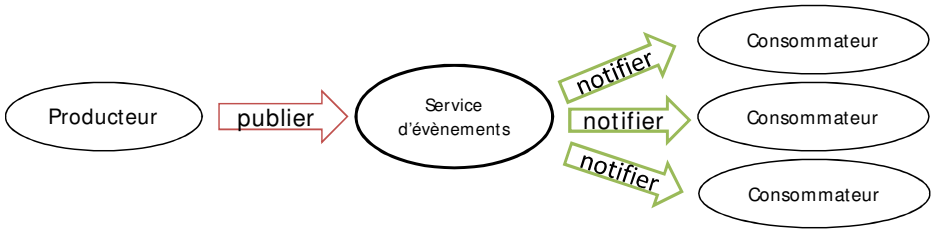

FIG. 1.4 – Découplage dans l'espace [EFGK03]

1.3.2.3 Découplage de synchronisation

La communication entre clients est complètement asynchrone (Figure 1.5). En effet, les consommateurs peuvent recevoir des notifications dans le temps où les producteurs publient d'autres évènements. De leurs parts, les producteurs peuvent fournir leurs évènements dans le temps où les consommateurs expriment les demandes sous forme de souscriptions.

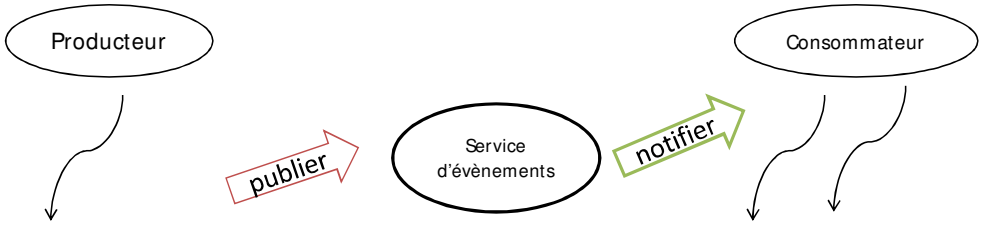

FIG. 1.5 – Découplage de synchronisation [EFGK03]

Ce découplage tridimensionnel offre aux consommateurs et aux producteurs une flexibilité et une grande facilité de communication. Cette caractéristique rend par excellence le paradigme publier/souscrire un moyen de communication adéquat pour les réseaux MANET.

Prenons l'exemple d'une application militaire pour illustrer cette constatation. Dans telle application, plusieurs soldats sont éparpillés sur le champ formant le réseau MANET. De nouveaux agents peuvent s'ajouter à tout moment sur le terrain de bataille. Les agents et les soldats collaborent ensemble afin de défendre contre l'ennemie.

Diverses contraintes s'imposent dans cette application afin d'assurer la communication entre différentes entités formant cette application :

- Les agents ne connaissent pas les uns les autres.
- Les soldats doivent être capables de communiquer en mode asynchrone pour faciliter la communication.
- De plus, les attaques subies par ces soldats introduisent des changements fréquents de leurs localisations. Mais ça doit en aucun sens inhiber leurs communications.

Tous ces facteurs mettent en relief le découplage tridimensionnel et motivent le recours

au paradigme publier/souscrire pour assurer la communication dans cette application sur un réseau MANET.

1.3.3 Classification des systèmes publier/souscrire

Les systèmes publier/souscrire pourraient être classés selon deux critères : selon le langage de souscription ou selon la topologie suivant laquelle les *brokers* sont connectés.

1.3.3.1 Le langage de souscription

Différentes techniques sont exploitées par les consommateurs afin d'exprimer leurs intérêts pour un évènement donné. En d'autres termes, ces techniques spécifient la manière avec laquelle les souscriptions sont exprimées. Ceci donne naissance à trois types de systèmes : des systèmes basés sujet, basés contenu et d'autres basés type.

– 1. Systèmes basés sujet

Les consommateurs spécifient leur intérêt sous forme de thèmes ou aussi sujets. De ce fait, chaque souscrit est associé à un thème, il reçoit par la suite toutes les notifications liées à ce sujet.

Par exemple, les consommateurs intéressés au thème "*sport*" expriment leurs demandes sous la forme suivante : *topic ="sport"*. Comme réponse, ces clients vont recevoir à partir du service d'évènements tous les évènements ayant le mot clef *sport* dans leurs sujets.

Parmi les systèmes basés sujet, nous citons : *Scribe* [CDK02] et *Mabcross* [DSM07].

– 2. Systèmes basés type

Dans ces systèmes, la classification des évènements est fondé sur la structure des évènements plutôt que sur leurs sujets. Ces systèmes [Pat07] utilisent un filtrage de l'évènement selon son type. De ce fait, le filtrage spécifie le type de l'évènement s'il s'agit d'un type simple et identifie sa classe s'il s'agit d'un objet.

Parmi les systèmes publier/souscrire basés type, nous citons : *Indiqos* [CAR05] et *Q* [MA05].

– 3. Systèmes basés contenu

Les systèmes publier/souscrire basés contenu sont apparus pour pallier au problème d'expressivité posés par les systèmes basés type et basés sujet. Le système basé contenu offre un filtrage basé sur le contenu réel de l'évènement.

En fait, un filtre est composé par un ensemble d'attributs auxquels correspond un ensemble de contraintes ayant la forme suivante : (type, nom, valeur, opérateur). Différents opérateurs peuvent être utilisés tels que ($>$, $<$, $=$, ?,$> *$,$< *$,etc.).

Une souscription d'attribut a = (type_a, nom_a, valeur_a) vérifie une notification d'attribut b = (type_b, nom_b, valeur_b, operateur_b) si et seulement si

– $\text{type}_a = \text{type}_b$

– $\text{nom}_a = \text{nom}_b$

– operateur_b (valeur_a,valeur_b) retourne vrai.

Dans ce cas, on peut affirmer que la souscription d'attribut a lie une notification d'attribut b (ou bien la notification d'attribut b couvre la notification d'attribut a).

Plus concrètement, un souscrit délivrant au sein du système une souscription de la forme "price > 40", aura comme notifications tous les évènements qui ont une valeur du champ "price" supérieure à 40.

Parmi les systèmes basés contenus, nous citons : *Elvin* [SAB+00] et *Siena* [CRW01].

1.3.3.2 La topologie

Les *brokers*, connectés les uns aux autres au sein du service d'évènements, forment une topologie bien déterminée. Réellement, il existe quatre types de topologies [CRW01] mettant en relief la façon avec laquelle les *brokers* sont connectés.

– 1. Topologie client/serveur hiérarchique

Suivant cette topologie, chaque *broker* est responsable d'un ensemble de clients. De ce fait, tous les messages vont être envoyés vers le *broker*, supérieur hiérarchique, ou encore *master*. Dans cette topologie, l'ensemble des *brokers* et des clients forment un graphe asymétrique.

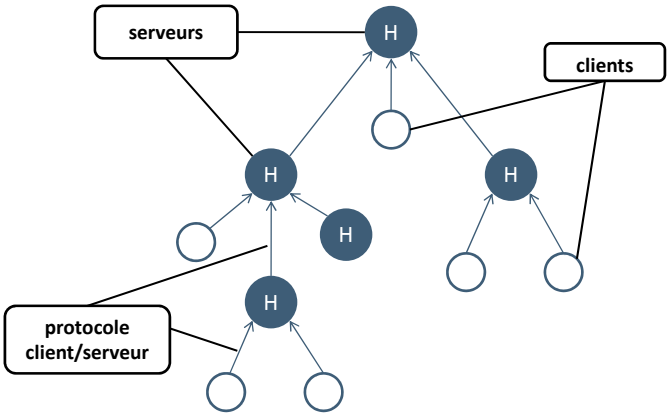

FIG. 1.6 – Topologie hiérarchique [CRW01]

Il est à noter ici qu'une panne affectant un nœud de la hiérarchie cause la panne de tous les nœuds situés au dessous de lui dans la hiérarchie.

– 2. Topologie pair à pair acyclique

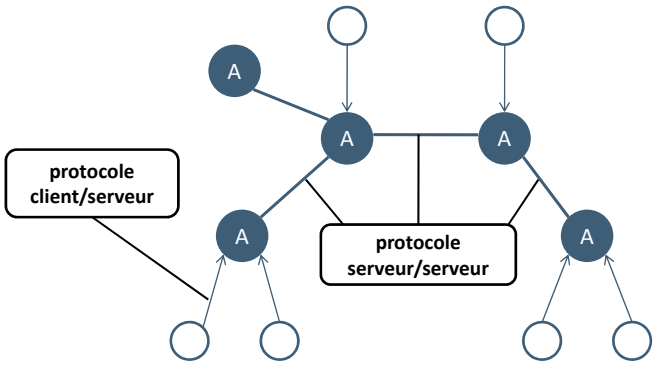

FIG. 1.7 – Topologie pair à pair acyclique [CRW01]

Dans cette topologie [CRW01], la communication entre les *brokers* se fait dans les deux sens (représentée sur le graphe par des arcs non orientés). L'ensemble des *brokers* forme un graphe acyclique. Suivant cette topologie, il existe un seul chemin entre chaque paire de *brokers* ce qui évite la formation de cycle.

Il est à noter que la panne d'un *broker* dans la hiérarchie cause l'échec de tous les nœuds connectés à lui.

– 3. Topologie pair à pair générique

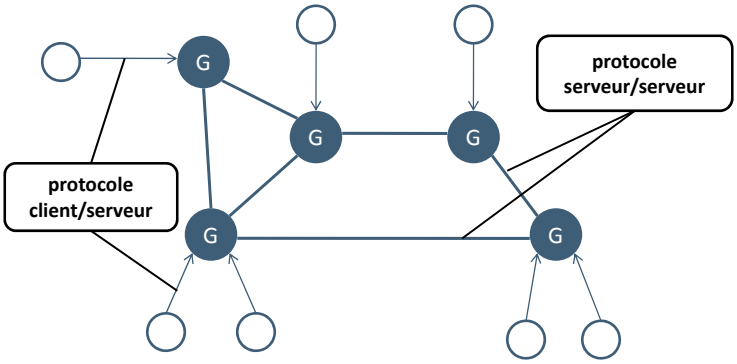

FIG. 1.8 – Topologie pair à pair générique [CRW01]

La topologie pair à pair générique [CRW01] permet aux *brokers* une sorte de communication bidirectionnelle. Selon cette topologie, il peut y exister un ou plusieurs chemins entre chaque paire de *brokers*, ce qui induit à une redondance de chemins dans le système. Ceci facilite la communication entre les parties constituant l'arbre logique (Figure 1.10).

– 4. Topologie hybride

Cette topologie combine les deux concepts prédéfinis, celui de la topologie pair à pair et de la topologie hiérarchique. Il existe deux types de topologies hybrides : une formée par une combinaison entre la topologie pair à pair générique et hiérarchique alors que la deuxième est une combinaison de la topologie pair à pair acyclique avec la topologie hiérarchique.

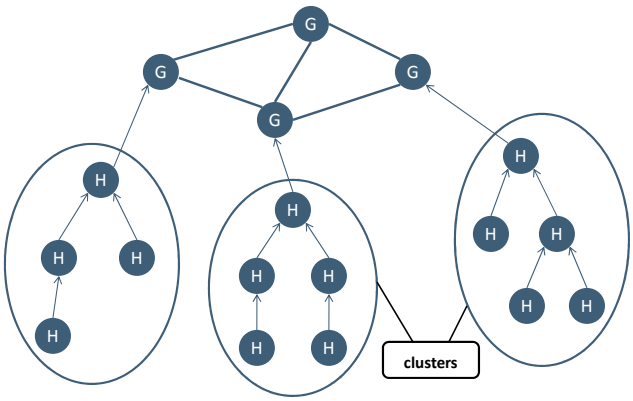

FIG. 1.9 – Topologie hybride [CRW01]

Par exemple, sur la Figure 1.9, les *brokers* du même groupe ou *cluser* sont connectés à travers une topologie hiérarchique, alors que celle entre les différents groupes est pair à pair générale.

1.4 Problématique et positionnement

Indépendamment de la classe à laquelle appartient un système publier/souscrire, ce concept reste convenable pour assurer la connectivité entre les parties régissantes dans n'importe quelle application. Ce paradigme présente un outil de communication prometteur pour les applications distribuées. En fait, le découplage tridimensionnel le caractérisant fait que ce type de système s'adapte parfaitement à des applications distribuées déployées sur MANET. Bien qu'ils paraissent dédiés, au départ, aux applications militaires, les systèmes publier/souscrire peuvent être déployés à bon escient dans des situations d'urgence, telles que les missions de secours : les incendies, les catastrophes naturelles (inondations, tremblements de terre, etc.). Néanmoins, divers défis en termes de QdS peuvent affecter les systèmes publier/souscrire sur MANET.
Le scénario suivant, représentant une application militaire, l'explique.
Le réseau, représenté par la Figure 1.10, est formé par des robots, des agents militaires, des hélicoptères... Toutes ces entités disposent de communication sans fil et sont éparpillées sur le terrain. Afin d'assurer la communication entre ces entités mobiles, il est judicieux d'y déployer des *brokers* et de mettre en place un système publier/souscrire.

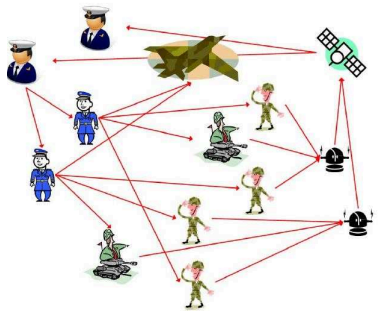

FIG. 1.10 – Application militaire

Ce déploiement met en place deux niveaux (Figure 1.11) :

- un niveau physique composé par l'ensemble des nœuds physiques ou dispositifs hétérogènes formant le réseau MANET.
- un niveau logique ou encore appelé niveau middleware composé par les nœuds logiques formant le système publier/souscrire. Ces nœuds sont de type producteur, consommateur et *broker*.

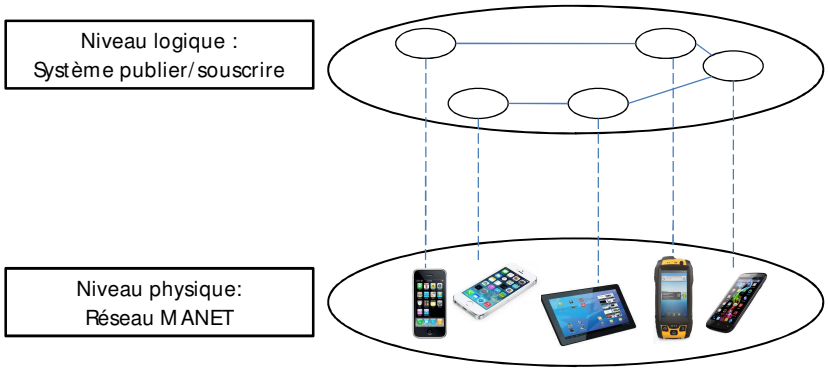

FIG. 1.11 – Déploiement d'un système publier/souscrire sur MANET.

La communication débute donc lorsqu'un agent produit une information qui indique la présence d'un risque qu'il l'a détecté à l'aide de son caméscope. Cette information sera véhiculée au sein du service d'évènements vers tous les hélicoptères et les autres agents chargés de défendre. Ces derniers comme étant connectés via le service d'évènements à travers des souscriptions, sont à l'écoute de notifications. Le défi majeur de ces agents est de défendre et de faire face à toutes les menaces qui pourraient se produire dans le champ d'attaque.

Au système publier/souscrire déployé sur le réseau MANET, s'ajoutent de nouvelles problématiques liées aux spécificités d'un tel environnement. En effet, aux défaillances possibles des entités formant notre système viennent se greffer les déconnexions de ces entités dues principalement à leur mobilité. Par ailleurs, un agent militaire peut se trouver loin de ses voisins ce qui nuit à la qualité de la liaison avec ces derniers et pourrait même menacer tout le champ de bataille.

Dans l'exemple de la Figure 1.12, les deux agents a et b mettent en place une liaison physique formée d'un seul saut. Ceci assure une qualité de communication acceptable. Après déplacement de l'agent a, le chemin entre les deux devient formé de 6 sauts (Figure 1.13). Ceci engendre bien évidemment une augmentation du temps d'acheminement et nuit aux performances du système.

Pour faire face à cette dynamicité, la couche réseau cherche bien évidemment une solution connectant cet agent à son voisin, mais le chemin trouvé pourrait être assez long, ce qui provoquera une prolongation du délai d'acheminement des messages entre les entités en question. Ceci entraînera certainement une violation des contraintes de l'application en termes de QdS.

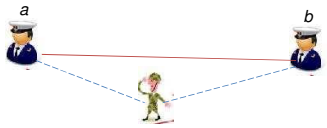

FIG. 1.12 – Avant déplacement des nœuds

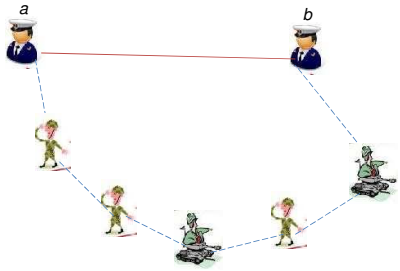

FIG. 1.13 – Après déplacement des nœuds

Ces problèmes ne cessent de s'aggraver par le fait que l'agent mobile a peut quitter complètement la zone de couverture de ses voisins. Dans ce cas, la liaison entre a et b est interrompue et le délai d'acheminement des messages est infini. Face à ces perturbations, la couche réseau se trouvera donc incapable d'établir la liaison entre les nœuds communicants. Dés lors, il est légitime de chercher à fournir à l'application des garanties sur le délai, appelé au niveau middleware la latence, afin de répondre aux fortes contraintes de ces applications.

Afin de pallier à ces problèmes, il est envisageable de fournir et de maintenir un support de qualité de service de ces systèmes. Ainsi, l'introduction d'une nouvelle solution au niveau de la couche *middleware* pour prendre en charge la notion de QdS s'avère nécessaire. Cette constatation a été démontrée via une publication [LJ11].
Dans le cadre de cette thèse, nous cherchons à fournir une gestion de la QdS au niveau de la couche middleware tout en supposant un flux *best effort* au niveau routage.

Au niveau de la couche middleware, divers travaux ont été proposés. Tous ces travaux partent d'un point commun qui suppose que le maintien de la QdS peut être réalisé grâce à un placement dynamique des *brokers*. Ayant le même point de départ, nous considérons que le placement des *brokers* sur le réseau est guidé par la QdS (dans notre cas la latence) observé entre les nœuds physiques hébergeant les *brokers* dans le chemin transférant les évènements. Ceci pourrait être achevé soit par la mobilité de ces nœuds, soit par leur surcharges. De ce fait, quand la mobilité d'un nœud augmente, on considère qu'une dégradation de QdS a affecté le lien entre le *broker* en question et ses voisins. Ceci indique un risque de dégradation de la QdS, et nous incite à revoir la distribution des *brokers* sur les nœuds.
Pour remédier à ces dégradations, des actions de reconfiguration doivent être introduites en réponse à chaque cause de panne afin d'assurer la survivabilité du système. Par exemple, façe à la mobilité, une action d'adaptation pourrait être accomplie en migrant un *broker* non stable vers un autre nœud du réseau offrant une meilleure QdS [LMJ09]. D'autres actions de reconfiguration pourraient être envisagées pour diminuer la charge d'un nœud *broker* en répartissant sa charge sur deux *brokers*.

Dans notre étude, cette approche générale est accomplie selon le processus des systèmes autonomes défini par IBM.
Par définition, un système autonome est capable de s'apercevoir qu'il est en état de disfonctionnement et de prendre les mesures nécessaires afin de restituer son état normal sans intervention humaine.
La solution proposée par IBM en 2001 met en place, au sein des systèmes autonomes, un processus d'auto-adaptation comportant quatre phases, à savoir (Figure 1.14).

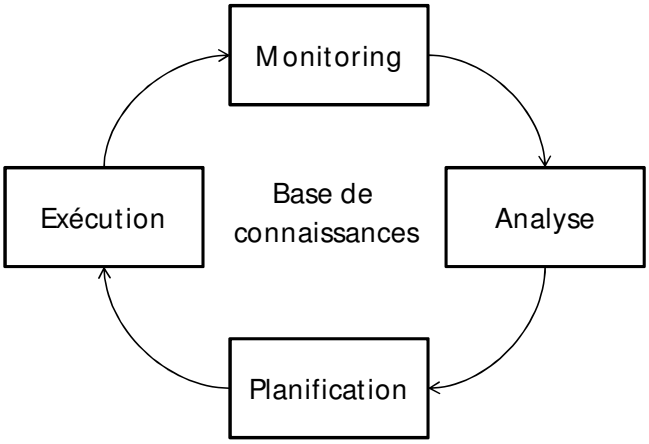

FIG. 1.14 – Processus d'auto-adaptation

- *Monitoring* : Cette phase se base sur l'observation et le stockage des valeurs des paramètres de QdS dans le but de les exploiter par les phases suivantes dans le processus d'auto-adaptation. Les paramètres de performance collectés reflètent l'état du système.
- *Analyse* : Cette étape présente la phase d'exploitation des mesures de QdS issues de la phase précédente. La phase d'analyse repose sur le suivi des mesures extraites afin de pouvoir caractériser l'état du système. Cette phase a pour finalité de détecter les dégradations de QoS considérées comme symptôme d'une panne affectant le système. La phase d'analyse est clôturée par l'identification de la cause de la dégradation ce qu'on appellera plus tard diagnostic.
- *Planification* : Dans le cas d'une dégradation de QdS identifiée par la phase d'analyse, la phase de planification se charge d'examiner les rapports de diagnostic afin de prendre des décisions de reconfiguration nécessaires et réparer le système. La phase de planification propose donc pour chaque source de panne une action corrective adéquate. Ceci pourrait être modélisé par des règles de type (if-then).
- *Exécution* : La phase d'exécution constitue la dernière étape dans le processus d'auto-adaptation. Elle consiste à exécuter les décisions prises lors de la phase de planification pour maintenir et rétablir une QdS acceptable fournie par le système.

Notre contribution par le biais de cette thèse cible les étapes de *monitoring* et d'analyse du processus d'auto-adaptation. Dans un premier temps, le *monitoring* consiste à calculer la latence, comme paramètre de QdS, entre les *brokers* voisins sur le chemin transférant les évènements. Ensuite, l'analyse consiste tout d'abord à prédire ou à détecter les dégradations de QdS, puis à chercher la cause derrière la dégradation. Cette dernière étape est appelée diagnostic.

Les solutions proposées actuellement pour la gestion de la qualité de service des systèmes publier/souscrire sur MANET, dans les phases de *monitoring*, d'analyse et d'exécution du processus d'auto-adaptation, sont détaillées dans le chapitre suivant.
Nous présentons donc un état de l'art autour de ces systèmes et plus particulièrement sur les méthodes adoptées pour le *monitoring* et l'analyse de la QdS tout en mettant en relief des critères de comparaison et d'évaluation de ces approches. Finalement, nous énumérons l'apport de notre approche vis à vis de ces travaux.

1.5 Conclusion

Dans ce chapitre, nous avons présenté le contexte dans lequel se situe ce sujet de thèse. Pour ce faire, nous avons d'abord commencé par décrire les réseaux mobile ad-hoc. Nous avons également étudié les caractéristiques primordiales de ces réseaux. Particulièrement, nous avons abordé le problème de QdS qui confronte ces réseaux à cause de la mobilité fréquente des nœuds.

Dans la deuxième partie de ce chapitre, nous avons donné un aperçu sur les systèmes publier/souscrire assurant la communication entre les différentes parties communicantes. Ensuite, nous avons énuméré les problèmes de QdS de ces systèmes sur les réseaux MANET. Ceci nous a amené à détailler la problématique découlée du déploiement des systèmes publier/souscrire sur MANET.

L'approche générale que nous proposons s'inscrit dans le cadre des systèmes autonomes, qui s'auto-contrôlent et s'auto-adaptent pour remédier aux problèmes rencontrés.

Le chapitre suivant expose un état de l'art autour des systèmes publier/souscrire qui ont traité le problème de la gestion de la QdS dans chacune des phases du processus d'auto-adaptation.

CHAPITRE 2

Etat de l'art

2.1 Introduction

La mobilité continue et imprévue des nœuds du réseau MANET crée un changement dynamique de topologie. Par ailleurs, un nœud peut joindre le réseau, changer sa position voire même quitter le réseau. Ce déplacement a bien évidemment un impact sur la morphologie du réseau et pourrait changer le comportement du canal de communication. Ajoutons à tout cela les ressources limitées des nœuds qui peuvent engendrer des déconnexions imprévues.
Dans cette situation, fournir une bonne QdS demeure un défi réel. Les systèmes publier/souscrire, grâce au découplage tridimensionnel, ont été proposés afin d'assurer la connectivité entre les parties communicantes dans le réseau mobile.

L'objectif de cette thèse est d'assurer un niveau de QdS acceptable qui provoque un maintien permanent de la connectivité des systèmes publier/souscrire sur MANET. Plus précisément, les contributions de cette thèse, sont organisées autour de deux grands axes relatifs aux modules de *monitoring* et d'analyse de la QdS de ces systèmes dans un contexte ad-hoc. Ceci englobe les étapes de collecte des paramètres de QdS au cours du fonctionnement du système, et de détection des dégradations pouvant l'affecter. Ces dégradations peuvent être réparées grâce à diverses actions de réparation qui pourraient être déclenchées lors de la phase d'exécution.

Dans ce chapitre, nous présentons un état de l'art autour des systèmes publier/souscrire adressant le problème de gestion de la QdS sur les réseaux MANET. Plus particulièrement, nous passons en revue ces systèmes dans les phases de *monitoring*, d'analyse et d'exécution du processus d'auto-adaptation.

Dans un premier temps, nous focalisons notre étude sur les systèmes publier/souscrire proposant des actions de reconfiguration ou de réparation du système. Puis, nous présentons les méthodes de *monitoring* et d'analyse proposés par les systèmes en question tout en mettant en évidence les critères de comparaison les plus importants entre les approches étudiées.

Une synthèse des travaux présentés nous a permis d'identifier les problèmes posés par ces systèmes et d'apporter des éléments de réponse à ces problèmes au sein de l'approche proposée.

2.2 Systèmes publier/souscrire adressant le problème de gestion de la QdS au niveau de la phase d'exécution

Une étude exhaustive sur les travaux existants dans la littérature, ayant pour finalité l'assurance de la QdS des systèmes publier/souscrire, montre que les systèmes en question proposent des actions de réparation des pannes affectant les *brokers* ou les liens [CP06, DSM09, KJ09, MCP11, ECR12, MBCK12, ZW13, PB14, HMS09].
Des techniques de routage au niveau logique ont été aussi proposées afin d'anticiper les pannes et d'empêcher leur production. Ceci a montré la présence de deux catégories de systèmes. Dans la première catégorie, les systèmes réagissent après la production de pannes par l'introduction de diverses actions de reconfiguration. Ayant le même but que la première catégorie, les systèmes appartenant à la deuxième suivent une approche préventive et cherchent à améliorer d'une façon proactive leurs performances à travers des techniques de routage optimisées au niveau logique.

2.2.1 Première catégorie : reconfiguration réactive

La première catégorie regroupe les systèmes publier/souscrire qui agissent au niveau middleware en réparant les pannes se produisant dans le système par l'apport de différentes actions de reconfiguration.

2.2.1.1 Le système *Reds*

Reds [CP06] est l'abréviation de *REconfigurable Dispatching System*. C'est le premier système basé évènement qui agit au niveau de l'arbre logique formé par l'ensemble des *brokers* afin de réparer le système. Le système ainsi formé s'auto-organise face aux changements dynamiques affectant la topologie.

En se basant sur les principes du protocole de routage MAODV, *Reds* construit et met à jour l'arbre logique formé par les *brokers*. Ce mécanisme commence par une sorte d'initialisation des niveaux des nœuds dans l'arbre. L'arbre est ainsi formé par un nœud racine ayant le niveau 0. Chaque nœud racine a un ensemble de nœuds fils regroupés sous forme de groupe *multicast* ayant des niveaux variés. Après avoir bâti un arbre logique représentant le système publier/souscrire, *Reds* traite le problème de repartitionnement de son arbre. A cause d'une rupture au niveau du lien logique dicté soit par la mobilité des nœuds du réseau physique, soit par leurs défaillances, l'arbre est repartitionné en deux ou plusieurs sous arbres indépendants.

Pour remédier à ce problème, *Reds* initie un processus de réparation et de rétablissement des liaisons en appliquant les principes du protocole MAODV. Dans l'exemple de la Figure 2.1, la liaison *(A-B)* est interrompue, le nœud B comme étant le nœud le plus loin de la racine, cherche un nœud possédant un niveau inférieur à celui de B. Le nœud D répond bien à cette contrainte. De ce fait, la nouvelle liaison *(B-D)* est établie et la mise à jour des profondeurs des nœuds est effectuée.

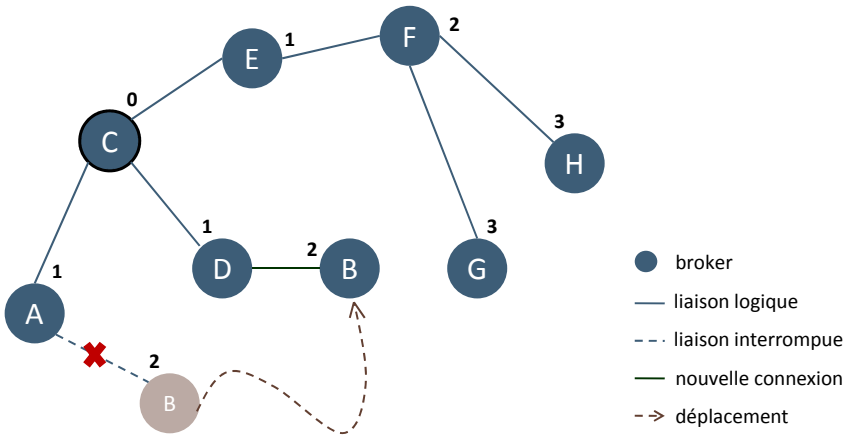

FIG. 2.1 – Exemple d'échec d'une liaison dans *Reds*

✓ Synthèse

Le système *Reds* maintient la connectivité de l'arbre tout en introduisant des actions de reconfiguration au système. Néanmoins, il présente quelques limites. En effet, la procédure de recherche de liaisons en remplacement de celle défaillante n'est pas garantie. Dans l'exemple de la Figure 2.1, le nœud B pourrait ne pas trouver le nœud qui répond à l'exigence imposée par le protocole MAODV. Dans ce cas, l'arbre demeure partitionné et le système échoue à rétablir la connexion.

De plus, *Reds* présente un problème au niveau des points d'accès. En effet, un client consommateur connecté à un point d'accès ne reçoit pas l'évènement répondant à ses besoins si le point d'accès auquel il est connecté tombe en panne. En fait, sa souscription n'est enregistrée qu'au niveau du point d'accès auquel il est rattaché. Encore plus grave, si le point d'accès tombe en panne, la souscription du client auquel il est lié, sera complètement perdue puisqu'elle ne figure que sur ce nœud.

Au niveau de l'opération de réparation, *Reds* souffre d'un blocage lors de cette opération. En effet, elle engendre un temps d'attente qui fait bufférisers les messages et engendre bien évidemment l'augmentation au niveau du temps d'acheminement de l'information.

2.2.1.2 Le système *Mabcross*

Mabcross [DSM07] est l'abréviation de *A Mobility-Aware and Cross-layer Based Middleware for Mobile Ad-Hoc Networks*. Ce système publier/souscrire utilise la notion des annonces afin de détecter les échecs survenues dans le système et de les réparer. L'appui sur ces messages au niveau de la réparation a pour but de réduire le nombre de messages spécifiques (de contrôle) et aussi minimiser la surcharge de ces messages.

Une annonce ou encore "*advertisment*" en anglais, est un message émis par les producteurs et transmis de près en près par les *brokers* au sein du service d'évènements afin d'indiquer la nature des publications qu'ils pourraient émettre. Lorsqu'un client consommateur émet une souscription à travers un point d'accès, sa souscription sera véhiculée seulement vers les *brokers* contenant une annonce correspondante. Finalement, quand le producteur émet son évènement, cette notification sera routée via le chemin activé par les souscriptions.

Une annonce est composée de :

- un numéro de séquence,
- l'identifiant du producteur noté ID,
- le sujet de l'évènement,
- les identifiants de tous les *brokers* qui appartiennent à la même route.

Envoyant périodiquement une annonce, chaque producteur dans *Mabcross* pourrait se rendre compte des déconnexions et des défaillances pouvant affecter le système. L'exemple présenté dans les Figures 2.2 et 2.3 détaille cette stratégie.

Comme hypothèse, on peut partir du fait que D_1 cherche un évènement qui va être publié par P_1 (Figures 2.2 et 2.3). La communication commence lorsque P_1 publie une annonce vers D_1. Ce dernier, ayant reçu précédemment une souscription qui vise à récupérer l'évènement de la part de P_1, met à jour l'annonce en insérant l'ID du *broker* D_1 dans la liste des souscrits. Ayant reçu l'annonce, D_1 change l'ID du producteur par son propre ID, et diffuse ce message en multicast.

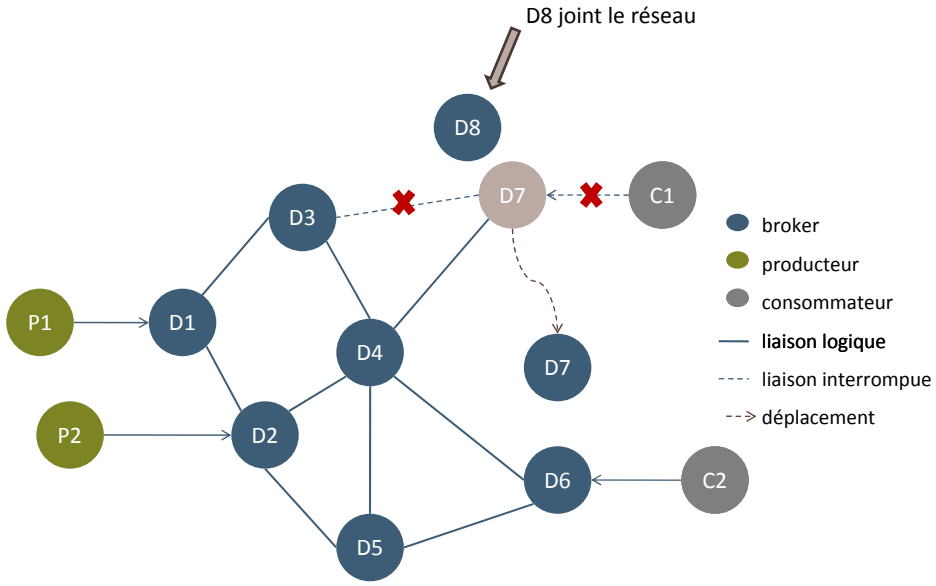

FIG. 2.2 – Tolopogie initiale

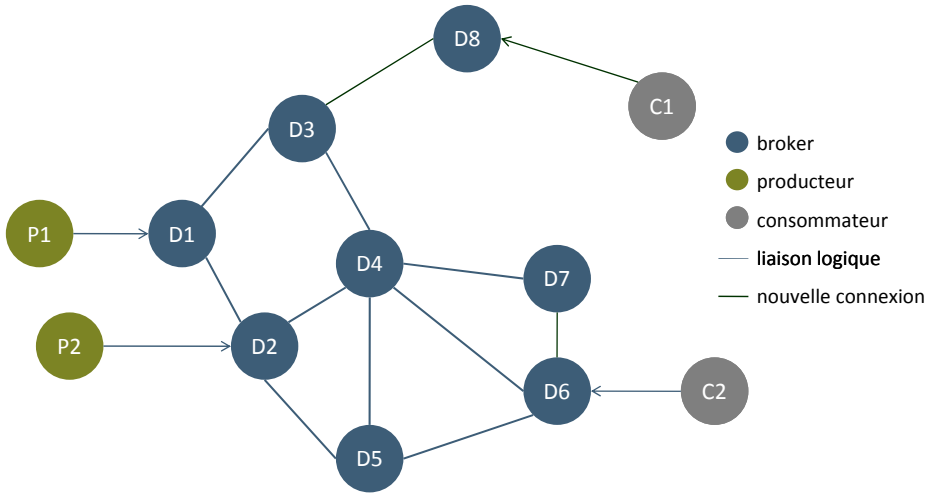

FIG. 2.3 – Topologie après la réaction de *Mabcross*

Supposons que D_7 se déconnecte et que D_8 vient de joindre le réseau. Ce dernier reçoit donc l'annonce envoyée périodiquement par le producteur. Il met à jour le champ ID et finit par l'envoyer au consommateur. A son tour, le consommateur reçoit l'annonce et compare le couple (ID- sujet) par celle reçue précédemment avant la déconnexion de D_7. Il note une incohérence au niveau des champs ID, de ce fait il envoie une désouscription à D_7 et bâtie une liaison avec D_8.

✓ Synthèse

En cas d'absence d'un nœud substituant le nœud partant, les consommateurs risquent de ne pas être notifiés. De plus, la stratégie adoptée dans ce système est basée sur l'envoi périodique d'annonces. Donc, un échec de liaison ou de *broker* n'est détecté qu'après la diffusion des annonces. Ceci augmente le risque de perte de messages si jamais une panne se produit avant l'envoi de ces messages. En outre, le problème des points d'accès mentionné dans *Reds* reste aussi persistant dans *Mabcross*.

2.2.1.3 Le système *Hermes*

Hermes [PEKS07] est un système publier/souscrire basé type. Il est bâti sur un réseau pair à pair de *brokers*. Selon [PEKS07], tous les *brokers* formant *hermes* portent connaissance de l'état de tous les *brokers* voisins et des informations de routage au niveau logique. *Hermes* utilise la notion de nœuds *Rendez vous* qui sont des *brokers* spécifiques servant à lier les annonces aux souscriptions. Ainsi, pour chaque type d'évènement, un nœud *Rendez vous* est crée.
Afin de garantir la non défaillance du nœud *Rendez vous*, *hermes* crée des duplications de ces derniers sur le réseau logique de *brokers*.

En cas d'échec d'un lien, *Hermes* introduit des opérations de reconfiguration au niveau logique visant à rétablir le lien défaillant. De plus, *Hermes* fait recours à diverses techniques qui visent à assurer la consistance de l'information en cours d'adaptation.
Pour ce faire, le réseau logique essaie de trouver des primitives remplaçant les routes défaillantes. Ceci pourrait causer un changement dans le chemin actif transférant les annonces et les souscriptions.
Lors de l'échec d'un *broker*, les messages de type souscriptions et annonces expirent bien évidemment et seront éliminés de la table de routage logique.
Afin de recouvrir cette défaillance, le nœud *broker* qui a détecté la défaillance est appelé à renvoyer les messages précédemment véhiculés au nœud défaillant vers le nœud *rendez-vous* via un nouveau lien. Cette information sera diffusée sur le réseau logique causant une mise

à jour de la table de routage logique de tous les *brokers*.

L'exemple de la Figure 2.4 montre l'opération de réparation dans *Hermes* en cas d'échec d'un nœud *broker*.

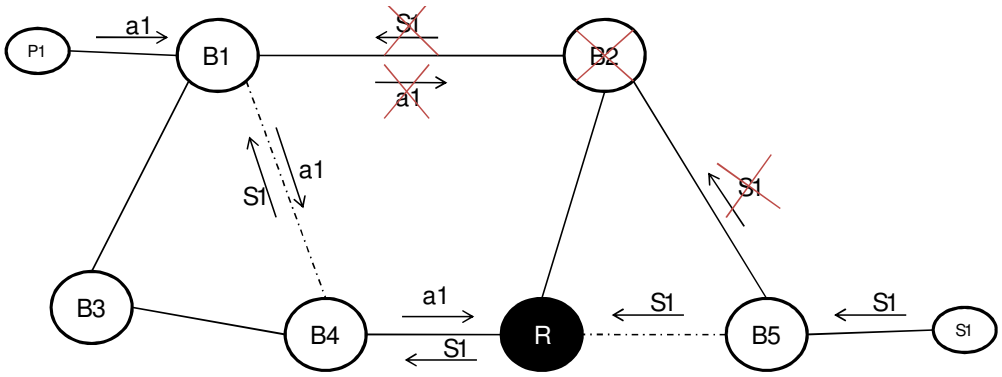

FIG. 2.4 – Opération de réparation dans *Hermes*

Le *broker* B_1 détecte la défaillance de B_2. Pour cela, il envoie l'annonce a_1 qui a été précédemment propagée vers le nœud *Rendez-vous* R. De sa part, le réseau logique, fait face à l'échec de B_2 en mettant à jour la table de routage logique de B_1 par élimination du lien $B_1 - B_2$ et l'ajout du lien $B_1 - B_4$. Le même mécanisme est exécuté au niveau du *broker* B_5 en propageant la souscription S_1 vers le nœud *Rendez-vous*. Ce dernier se charge de faire la correspondance entre l'annonce a_1 et la souscription S_1 et finit par acheminer la publication de P_1 vers le consommateur souscrit sur le chemin (R-B_5-S_1).

✓ Synthèse

Le modèle de tolérance aux pannes dans *Hermes* est basé sur l'utilisation des nœuds *Rendez vous*. Utiliser ces nœuds ainsi que les réplications de ces derniers pourrait ajouter un trafic supplémentaire sur le réseau logique et causer même sa saturation.

2.2.1.4 Récapitulation

Bien que les systèmes de la première catégorie offrent des solutions de réparation de l'arbre logique à travers une technique de substitution de nœud ou de lien , ils partent d'une hypothèse forte qui suppose que la topologie des *brokers* pourrait être toujours représentée

sous forme d'un arbre. Chose qui impose des restrictions et rend ces approches exploitables que pour les systèmes à topologie pair à pair cyclique. De plus, les actions de réparation engendrent un blocage du système et peuvent mener vers un dysfonctionnement du système jusqu'à la fin de la réparation. Finalement, l'inconvénient majeur de ces systèmes est la production de pannes vu leurs effets néfastes sur le système.

2.2.2 Deuxième catégorie : planification proactive des routes

Outre les systèmes décrits dans la première catégorie, il existe d'autres systèmes qui réagissent avant la production de pannes par l'introduction d'opérations périodiques d'optimisation.

2.2.2.1 Le système *Relrout*

Le système *Relrout* [MB08] propose un routage au niveau logique qui permet de réduire au maximum l'échec de livraison des notifications vers les consommateurs adéquats. Pour ce faire, le système part d'une hypothèse qui suppose qu'il existe plusieurs chemins entre chaque source et destination avec différents niveaux de fiabilité.
Afin d'acheminer l'information, le système calcule d'abord une estimation de la fiabilité de chaque route reliant une source à une destination. Egalement, le système propose un algorithme de mise à jour de ces fiabilités estimées en réponse aux changements de topologie. Ensuite, il envoie le message aussi bien sur le chemin ayant la fiabilité la plus élevée, et sur d'autres chemins ayant différents niveaux de fiabilités.
Moyennant cette stratégie, *Relrout* introduit une opération d'optimisation proactive des routes logiques entre chaque paire de clients et garantit une livraison fiable des messages.

✓ Synthèse

L'approche proposée dans [MB08] réduit l'occurrence de pannes dans le système. *Rel-Rout* prédit la panne des liens en adoptant les meilleures routes en termes de fiabilité. Cependant, le système considère uniquement la panne des liens, ainsi, il ne traite pas la panne des nœuds ou leur dynamicité. En outre, le système reste incapable de récupérer les

notifications perdues.

2.2.3 Synthèse

En vue de répondre aux exigences des applications en termes de QdS, les opérations de réparation deviennent de plus en plus importantes. Toutefois, quelques actions de réparation peuvent engendrer des problèmes de blocage et de perte de messages. Ce qui provoquera le dysfonctionnement du système pendant la reconfiguration.

Le tableau 2.1 récapitule l'étude présentée dans la section précédante et présente une comparaison de ces approches cherchant à maintenir la connectivité au sein des systèmes publier/souscrire sur MANET.

TAB. 2.1 – Synthèse

Systèmes	Langage de souscription	Topologie	Type du réseau	QdS	Techniques d'adaptation
Reds [CP06]	basé contenu	pair à pair acyclique	ad-hoc	-	reconfiguration réactive
Mabcross [DSM07]	basé sujet	pair à pair cyclique	ad-hoc	-	reconfiguration réactive
Hermes [PEKS07]	basé type	pair à pair cyclique	ad-hoc	-	reconfiguration réactive
Relrout [MB08]	basé type	pair à pair cyclique	ad-hoc	fiabilité	planification proactive des routes

Les différentes approches détaillées ci-dessus traitent le problème de QdS dans les systèmes publier/souscrire sur MANET. [CP06] et [DSM07], [PEKS07] réagissent après pannes et introduisent des techniques de substitution des liens logiques. En contre partie, [MB08] réagit avant la panne par construction dynamique du réseau de *brokers* d'une façon optimisée en termes de fiabilité des liens. Ceci a pour effet d'échapper de la production des pannes.

D'une façon générale, toutes les approches étudiées visant à introduire des actions

correctives dans le système d'une façon réactive ou préventive, partent d'une hypothèse qui suppose que la gestion de la QdS pourrait être traitée grâce à un placement dynamique des *brokers* sur les nœuds physiques.
Notre point de départ est similaire, et considère que le placement des *brokers* est régie par la latence comme paramètre de QdS. Notre vision corrective s'inscrit donc dans le même cadre que ces travaux, et opte pour un maintien permanent de la QdS au sein du système à travers l'élaboration des deux premières phases du processus d'auto-adaptation.

Dans la section suivante, nous présentons notre étude autour des systèmes publier/souscrire adressant la gestion de la QdS au niveau des étapes de *monitoring* et d'analyse.

2.3 Systèmes publier/souscrire adressant la gestion de la QdS au niveau des phases de *monitoring* et d'analyse

Au niveau des phases de *monitoring* et d'analyse du processus d'auto-adaptation, quelques systèmes adressent la gestion de la QdS. Notre travail cible exactement ces deux étapes et propose de nouvelles démarches théoriques et techniques.
Une étude exhaustive de ces travaux nous a permis de les classer suivant la méthode adoptée pour le *monitoring* ou pour l'analyse. L'analyse, comme étant composée de deux étapes fortement indépendantes, sera présentée en deux parties : la première présente les méthodes de détection des dégradations, alors que la deuxième résume les méthodes de localisation des pannes ou de diagnostic.

2.3.1 Paramètres de QdS traités dans les systèmes publier/souscrire

Divers systèmes publier/souscrire ont eu comme objectif la gestion de la QdS. Dans la littérature, les approches traitant la QdS ont ciblé différents paramètres [MKB07] (Figure 2.5).

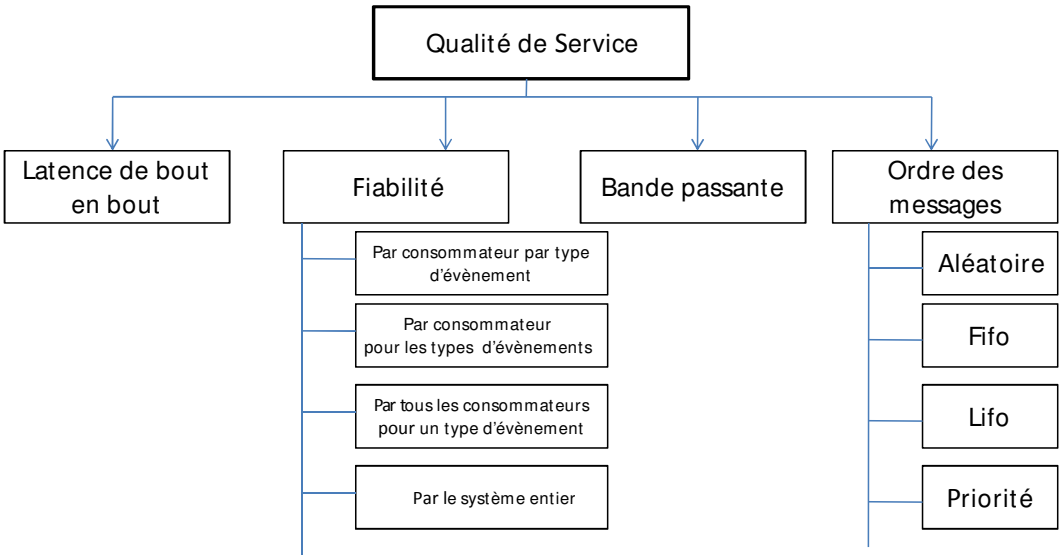

FIG. 2.5 – Paramètres de QdS dans les systèmes publier/souscrire [MKB07]

1/ Bande passante : représente les ressources disponibles sur un chemin transférant un message.

Au niveau de la couche middleware, la bande passante représente le nombre d'évènements véhiculés entre les consommateurs et les producteurs par unité de temps.

2/ Ordre des messages : représente la façon dont les messages sont transmis. Les systèmes publier/souscrire identifient cinq classes, à savoir : un ordre aléatoire, un ordre FIFO, un ordre LIFO, et un ordre personnalisé spécifié par le producteur.

3/ Fiabilité : qualifie l'opération d'envoi et de réception des messages.
La fiabilité représente le rapport entre le nombre de notifications qui ont été reçues par un consommateur « c » noté n(e_τ) pour un évènement particulier et le nombre de publications

relatives à cet évènement et produites par le producteur p(e_τ).

La fiabilité R atteinte par un consommateur « c » est donc exprimée à l'aide de la formule suivante :

$$R[e_\tau] = \frac{n(e_\tau)}{p(e_\tau)} \tag{2.1}$$

4/ Latence : c'est le temps qui sépare l'instant de production d'un évènement par un producteur et l'instant de réception de cet évènement par le consommateur correspondant.

La latence sur un chemin est la somme des délais de propagation dans tous les nœuds *brokers* appartenant au chemin avec le délai de transmission et de bufférisation dans chacun des nœuds *brokers* appartenant au chemin considéré.

$$L(e_\mu) = b * (P_d) + \sum_{i=1}^{y-1} (T_d[i] + Q_d[i]) \tag{2.2}$$

où :

- $L(e_\mu)$ est la latence prise par une notification pour un évènement de type μ entre un producteur et un consommateur.
- T_d est le délai de transmission,
- P_d est le délai de propagation (valeur constante sur le réseau),
- Q_d est le délai de bufférisation,
- e est un évènement correspondant à un type μ,
- b est le nombre total de nœuds appartenant au chemin considéré,
- y est le nombre de nœuds *brokers* appartenant au chemin considéré.

2.3.2 Méthodes de *monitoring* et d'analyse

Une étude exhaustive de l'existant a montré que les méthodes de *monitoring* et d'analyse se subdivisent en deux grands axes. Le premier concerne les méthodes réactives qui visent à détecter les pannes et à identifier les défaillances une fois subies par le système. Alors que le deuxième axe englobe les méthodes proactives qui ont pour objectif la prévention des pannes avant leurs occurences.

Ces deux méthodes identifiées à travers une étude bibliographique seront détaillées ci-dessous.

2.3.2.1 *Monitoring* et analyse réactive : méthodes de détection de pannes basées sur des messages spécifiques

Divers travaux ont pour but de détecter la panne des nœuds dans les réseaux de communication en se basant sur des messages spécifiques de type *Ping* ou *Heartbeat*.

– 1. Architectures *Ping*

Un message *Ping* est un message envoyé par le nœud source vers le nœud destination dans le but de lui poser la question suivante : "Es tu vivant ?". Dès réception, le nœud récepteur répond par un acquittement "Oui, je suis vivant". La détection de défaillance peut donc se faire suivant deux stratégies différentes :

– un nœud est considéré en panne s'il ne répond pas à un nombre de messages bien déterminé de type *Ping*.
– un nœud est défaillant s'il sera incapable de produire des acquittements durant un temps prédéfini.

Dans [MMH04], *Musolesi et al.* ont proposé le système publier/souscrire *EMMA*[1] comme extension du système JMS sur les réseaux mobiles ad-hoc. Le système fait recours aux algorithmes épidémiques dans le but d'assurer la livraison de messages. Le système se base sur le fait que chaque message est répliqué. De ce fait, des copies de chaque messages sont éparpillés dans tous les nœuds du système.

Afin de détecter les pannes dans ce système, *Musolesi et al.* utilisent des détecteurs actifs qui se basent sur l'échange de messages spécifiques de type *Ping*. Ainsi, si un nœud *broker* n'a pas reçu un acquittement du message "Es tu vivant ?" pendant une période de temps fixe, ce nœud est considéré défaillant et une panne est identifiée dans le système.

[1]Epidemic Messaging Middleware for Ad-hoc networks

✓ Synthèse

En s'appuyant sur des détecteurs de pannes actifs et moyennant des messages *Ping*, *Emma* permet de détecter la panne des *brokers* sur un réseau mobile. Cependant, ce système n'offre pas une support de détection de la panne des liens fréquemment rencontrés dans des réseaux dynamiques.

En outre, le recours aux acquittements pour vérifier la vivacité d'un nœud, cause un gaspillage en termes de bande passante et d'enérgie dans un réseau à ressources limités.

Finalement, le recours à une borne fixe pour la réception de l'acquittement s'oppose aux caractéristiques des réseaux dynamiques et pourrait probablement mener vers des fausses détections de pannes.

La Figure 2.6 illustre un exemple de ce problème.

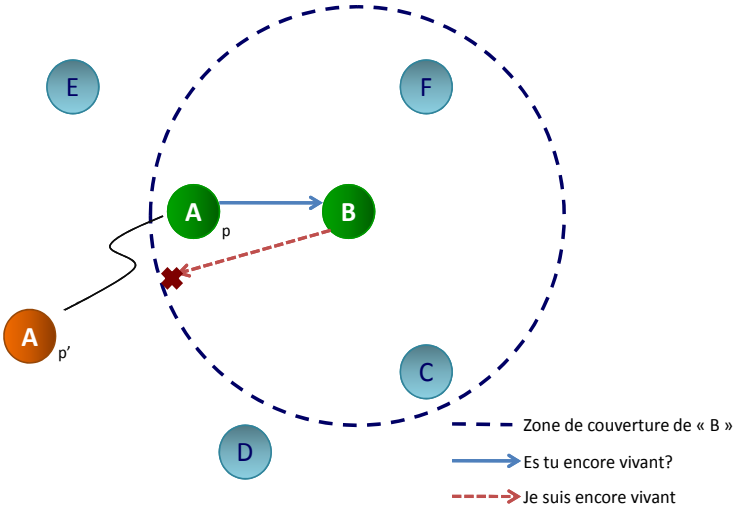

FIG. 2.6 – Problème posé par les messages *Ping*

En se basant sur le principe expliqué précédemment, le nœud A envoie un message *Ping* vers son voisin B. Une requête est donc envoyée de A vers B à l'instant $t = t_e$.

Vu la dynamicité du réseau, le nœud A change de position pendant que le nœud B est entrain de lui envoyer l'acquittement "Je suis vivant". A est devenu hors de la porté de B et ne recevra pas la réponse dans la période de temps tolérée. En conséquence, le nœud A conclue que le nœud B est défaillant et génère une fausse alarme via à vis de son voisin B.

– 2. Architectures *heartbeat*

La technique *heartbeat* présente une amélioration des techniques *Ping* par élimination des acquittements. Selon cette technique (voir Figure 2.7), le nœud A envoie périodiquement une pulsation "Je suis vivant" vers le ou les nœuds destinations. Si le nœud destinataire ne reçoit pas ce message au bout d'une période fixe Δ_{t_0}, le nœud A est considéré défaillant.

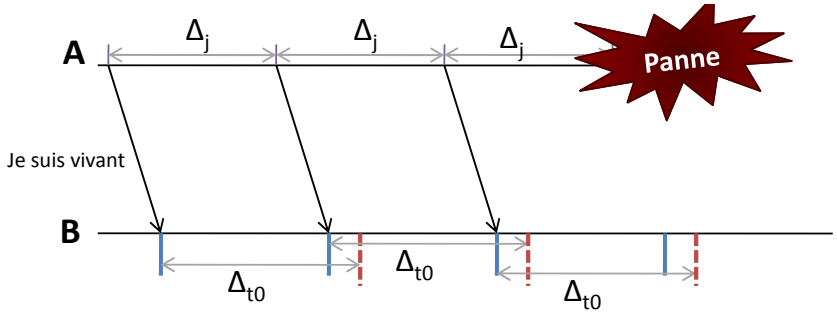

FIG. 2.7 – Principe de l'architecture *heartbeat*

Dans la littérature, diverses stratégies peuvent être adoptées dans l'utilisation des messages *heartbeat*. Les systèmes publier/souscrire décrits ci dessus le témoignent.

– 2.1 Architectures "tous à tous"

Dans cette classe, tout détecteur de panne incorporé au sein de chaque nœud du réseau émet périodiquement des messages de type *heartbeat* vers tous les nœuds du réseau, ce qui explique l'appélation "Tous à Tous".

De ce fait, si un nœud récepteur nommé B ne reçoit pas un message de son voisin A au bout d'une certaine durée de temps préfixée, le détecteur de panne de B assume que le nœud A est devenu défaillant.

Un parmi les systèmes qui utilisent cette architecture est *Hermes* [PB02]. C'est un système publier/souscrire basé type tolérant aux pannes. Il met en place, au sein de chaque *broker*, un détecteur de panne actif qui prend en compte la détection de la panne des *brokers* et des liens. De ce fait, chaque *broker* dans *Hermes* envoie périodiquement des messages de type *heartbeat* à ses voisins. Si un nœud sera incapable d'interagir avec ses voisins, il sera considéré en panne.

✓ Synthèse

Hermes offre un module de détection de pannes se basant sur l'architecture *heartbeat*. En revanche, *Hermes* adopte un contexte statique qui ne tient pas compte de la dynamicité, caractéristique primordiale des réseaux mobiles. En outre, l'envoie de messages *heartbeat* vers tous les nœuds voisins requière une bande passante plus importante.

– 2.2 Architectures en anneau

Dans cette classe, les nœuds sont interconnectés entre eux formant un anneau. Les messages *heartbeat* sont donc envoyés d'un nœud vers son successeur dans l'anneau. Pour détecter les pannes, la même principe de détection de pannes expliquée précédemment, est adoptée.

Le système [JMV08] faisant recours à cette architecture, est un système basé contenu qui intègre un module d'analyse responsable de détecter la panne des nœuds en se basant, essentiellement, sur l'échange de messages de type *heartbeat*.

Le système est organisé sous forme de groupes (*clusters*) et d'anneaux comme le montre la Figure 2.8.

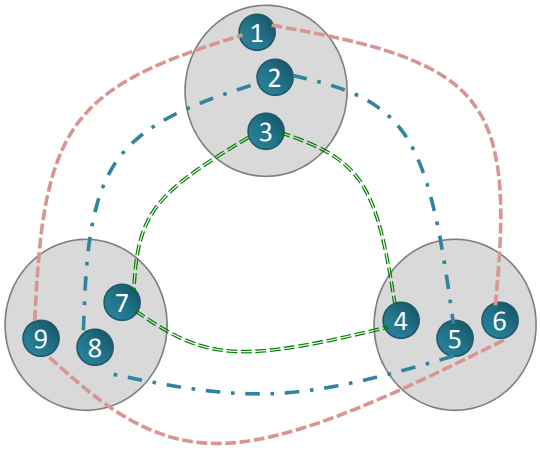

FIG. 2.8 – Architecture du système [JMV08]

Dans ce système, *Jafarpour et al.* se basent sur une hypothèse forte qui suppose que tous les liens sont fiables. C'est ainsi qu'ils détectent uniquement la panne des *brokers* par perte de messages *heartbeat*.

– 2.3 Architecture en *clusters*

Dans cette catégorie, le réseau est organisé sous forme de groupes (*clusters*). Tout *cluster* est surveillé par un maitre. La détection de pannes dans une telle classe se base sur l'usage des messages *heartbeat* dans le même groupe et entre les maitres associées aux différents groupes (*clusters*).

L'approche proposée par *Ben Khedher et al.* [KGD07] se basant sur le principe de l'architecture en *clusters*, vise à détecter les problèmes de panne des nœuds et de perte de messages. Afin de satisfaire leurs objectifs, *Ben Khedher et al.* organisent le réseau en un ensemble de couches comme le montre la Figure 2.9.

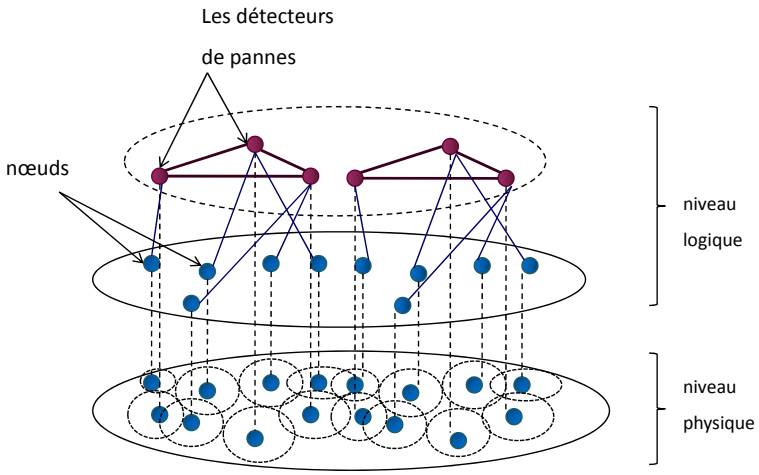

FIG. 2.9 – Architecture du réseau dans l'approche de *Ben Khedher et al.*

Selon cette Figure, les détecteurs de pannes occupent la couche supérieure. Chaque détecteur est responsable de détecter la panne d'un ensemble de nœuds *brokers* qui lui sont accordés. Pour ce faire, les détecteurs se basent sur les messages *heartbeat* comme le montre la Figure 2.10.

Ainsi, chaque nœud *broker* envoie périodiquement des messages de type *heartbeat* au détecteur qui lui est attaché. Si une période de temps s'est écoulée sans que le maître du groupe ou le détecteur reçoit ce message, il véhicule un message de type *Warning* vers le nœud émetteur. Si ce dernier est encore vivant, il notifie le détecteur par un message *correct*. A défaut, le détecteur admet que le nœud est devenu défaillant.

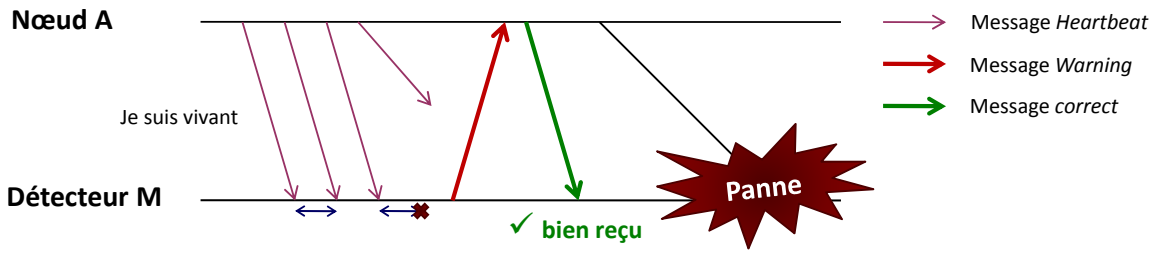

FIG. 2.10 – Principe de détection dans l'approche de *Ben Khedher et al.*

Pour détecter l'échec du détecteur, chaque paire de détecteurs envoie périodiquement des messages *heartbeat*. De ce fait, si un détecteur M_1 ne reçoit pas un *heartbeat* venant d'un autre détecteur M_2 sur un laps de temps pré-établie, M_1 véhicule un message *warning* vers tous les détecteurs. De ce fait, si M_2 est encore vivant, il diffuse un message *correct* vers tous les détecteurs. A défaut, M_2 est considéré défaillant si M_1 ne reçoit de sa part ni des messages *heartbeat* ni des messages *correct*.

✓ Synthèse

Bien qu'elle a apporté des améliorations par rapport à l'architecture *Ping*, l'architecture *Hearbeat* présente aussi des inconvénients. En effet, comme le montre l'exemple de la Figure 2.11, le nœud A émet périodiquement un message "Je suis encore vivant" à son voisin B.

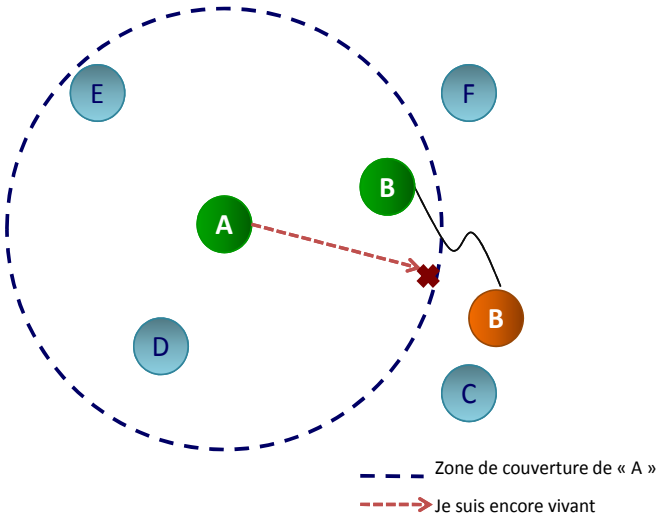

FIG. 2.11 – Problème posé par les architectures *heartbeat*

Entre temps, B peut changer de localité et devenir hors de la zone de couverture de son voisin A. Après une période de temps fixe, B conclut que A est défaillant. Cette synthèse erronée pourrait déclencher des fausses détections au sein du système.

– 3. Récapitulation

Le *monitoring* et l'analyse basée sur un échange de messages de types *Ping* et *Hearbeat* risque de causer un gaspillage en termes de bande passante et d'énergie.
De plus, l'utilisation d'un intervalle de temps fixe pour la réception d'acquittement n'est pas adéquate au contexte dynamique et peut engendrer, par la suite, des fausses alarmes.
En outre, dans un réseau MANET caractérisé par une forte dynamicité, un nœud désirant envoyer un message peut changer de localité pendant que le nœud récepteur est entrain de lui envoyer l'acquittement.
Par conséquent, l'acquittement ne sera pas reçu correctement par le nœud émetteur pendant la durée de temps Δ_t. De ce fait, le nœud émetteur considère que le nœud récepteur est défaillant, ce qui n'est pas le cas. Ceci peut engendrer bien évidemment des fausses détections.

2.3.2.2 *Monitoring* et analyse réactive : méthodes de détection de pannes se basant sur la comparaison par rapport à un seuil

Cette catégorie se base sur la comparaison des paramètres de performances du système par rapport à une valeur seuil dans le but d'identifier les dégradations de QdS et les pannes survenant dans le système.
Dans ce qui suit, nous énumérons les systèmes publier/souscrire utilisant cette technique.

– 1. Jerzak et al. [JFF07]

Jerzak et al. ont proposé un middleware qui tient compte du délai comme paramètre de QdS appelé *fail-aware*. Le délai est calculé en utilisant des messages spécifiques de type *Ping*. Le système utilise une marge de valeurs de seuils allant d'une borne inférieure à une borne supérieure sur le délai de transmission d'un évènement.
La méthode d'analyse incorporée au sein de ce système vise à décider si un évènement

satisfait les contraintes ou les bornes temporelles spécifiées dés le départ. Pour ce faire, *Jerzak et al.* ont recours à la formule suivante.

$$t_d(m) \leq u_b(m) \tag{2.3}$$

Ceci fait que, si le temps de transmission $t_d(m)$ d'un évènement est inférieur à la borne supérieure $u_b(m)$, le système est dans un état acceptable. Sinon, le système souffre d'une violation. Chaque message véhiculé au sein du système devra donc obéir aux contraintes temporelles imposées par le système.

✓ Synthèse

Bien que le système impose des contraintes liées au délai d'acheminement des messages, il présente quelques insuffisances. En Effet, le système se base sur l'usage de bornes ou de seuils fixes. Une valeur seuil fixée à un instant t peut devenir non valable après un certain temps vu la dynamicité du réseau. Par conséquent, il s'avère inadéquat de faire référence à des seuils fixes dans un contexte mobile vu que leurs valeurs fixes peuvent devenir non valables quand le réseau change de topologie ou de contexte.

2.3.2.3 Récapitulation

Les méthodes de *monitoring* et d'analyse utilisant des messages spécifiques ainsi que celles qui comparent les valeurs de QdS par rapport à des seuils fixes se rejoignent pour assurer la QdS des systèmes publier/souscrire sur MANET. Ces approches permettent de détecter les pannes dans de tels systèmes mais elles présentent quelques limites.

Par ailleurs, les messages spécifiques, que ce soit de type *Ping* ou *Heartbeat*, introduisent des flux supplémentaires sur le réseau. Cependant, ces flux sont inappréciables dans des réseaux mobile ad-hoc caractérisés par des ressources limitées.

Dans la même directive, d'autres approches d'analyse se basant sur la comparaison par rapport à un seuil ont été proposées. Ces approches font recours à des seuils figés qui contredisent le contexte dynamique. En effet, un seuil préfixé à un instant t peut devenir inadéquat après un certain temps puisque le réseau subit fréquemment des changements .

Le tableau récapitulatif suivant (Tab 2.2) résume ces approches tout en mettant en relief les éléments de comparaison les plus pertinants.

TAB. 2.2 – Synthèse des travaux étudiés

Travaux	Réseau	Langage de sous-cription	Type de pannes	Source de pannes	Architecture du détecteur	Technique
Hermes [PB02]	statique	basé type	nœuds	mobilité	distribuée	*heartbeat*
Emma [MMH04]	ad-hoc	Basé sujet	nœuds	mobilité	distribuée	*Ping*
Fail-aware [JFF07]	statique	basé contenu	-	-	distribuée	comparaison par rapport à un seuil fixe
Jafarpour et al.[JMV08]	statique	basé contenu	nœuds	connexion et déconnexion	distribuée	*heartbeat*
Ben Khedher et al. [KGD07]	ad-hoc	hybride	nœuds	mobilité	hybride	*heartbeat*

2.3.2.4 *Monitoring* et analyse préventive

Dans la littérature, d'autres systèmes publier/souscrire adoptant une analyse préventive visent à anticiper l'occurrence des pannes et inhiber leurs productions. Dans ce qui suit, nous présentons ces systèmes tout en mettant l'accent sur les méthodes de *monitoring* et d'analyse exploitées.

– 1. Le système *Opportunistic Overlays* [CS05]

Ce système [CS05] est bâti sur le système basé évènement Jecho [ZSEC01]. Il résout les problèmes de changement fréquent de topologie rencontrés dans les réseaux dynamiques par une mise à jour périodique de l'arbre logique. En effet, le système consulte d'une façon permanente les routes logiques. S'il s'aperçoit qu'il existe une route plus courte que celle active, il reconstruit le réseau logique en adoptant la meilleure variante en termes de nombre de sauts logiques.

Chaque *broker* maintient une connaissance de la topologie de tout le réseau logique dans une table de topologie notée T. Périodiquement, chaque *broker* reçoit les tables de topologie de ses voisins, met à jour sa table, et propage toutes les modifications de topologie si elles existent.

La reconstruction de l'arbre logique passe par les quatre étapes suivantes :

- Découverte des *brokers* voisins : Chaque *broker* met à jour périodiquement sa table en utilisant une méthode de découverte de routes appelée "Expanding Ring Search". Si un nœud se déplace très loin de ses voisins, le lien initial entre ce nœud et son voisin *broker* sera éliminée de l'arbre initial. En contre partie, si un nœud se déplace dans la zone de couverture d'un autre voisin *broker*, une nouvelle liaison est ajoutée entre ces deux nœuds.
- Propagation de la topologie de l'arbre logique : A chaque fois qu'un *broker* met à jour sa table de voisins *brokers*, il envoie à ses voisins toutes les nouvelles données. Un numéro de séquence est attribué à chaque mise à jour.
- Quand un *broker* reçoit une information de mise à jour, il compare le numéro de séquence du message contenu dans l'information venante avec celui contenu dans sa table. Il marque le message s'il a un numéro de séquence plus grand.
- Reconstruction de la table de voisins *brokers* : La table des voisins *brokers* T change si une mise à jour est efféctuée dans le voisinage par le *broker* ou lors de la reception d'une information de mise à jour des voisins *brokers*.

Pour réaliser ceci, le système exploite quatre tables maintenues au niveau de chaque *broker* à travers un composant appelé *broker manager* contenant quatre tables :

- La première table appelée "*Broker Neighbor Table*"(BNT) sert à stocker toutes les informations relatives aux *brokers* voisins avec leurs noms ainsi que la distance en termes de sauts physiques le séparant de ses voisins. A chaque modification se produisant au niveau des routes, le *broker* met à jour cette table.
- La deuxième table appelé "*Broker Information Table*" contient l'adresse IP du *broker*, sa position actuelle ainsi que la capacité de la mémoire disponible.
- La troisième table appelée "*Broker Network Topology Table*" (BTT) sauvegarde la

topologie du réseau logique tout en précisant la distance entre *brokers*.

– La quatrième table "*Broker Routing Table*" (*BRT*) contient les routes les plus courtes menant vers tous les *brokers* du réseau.

Dans le but de reconstruire l'arbre logique à chaque changement de topologie, ces tables sont mises à jour périodiquement. En effet, chaque *broker* détecte périodiquement son voisinage et propage en conséquence la topologie du réseau à tous ces voisins noté S_D. Chaque mise à jour de la table sera donc accompagnée par une attribution d'un numéro de séquence. Ceci lance bien évidemment une mise à jour au niveau des tables *BNT* et *BTT* des voisins S_D dans le cas ou le numéro de séquence des messages reçus par D est plus grand que celui existant dans les tables appartenant à S_D. Finalement, la table *BRT* est mise à jour en recalculant la route la plus courte menant vers les voisins.

Les Figures 2.12 et 2.13 montrent le processus de construction dynamique du réseau logique (de *brokers*) dans le réseau MANET. Sur ces Figures, la table *BNT* est présentée par une matrice qui décrit le nombre de sauts logiques entre les *brokers*.

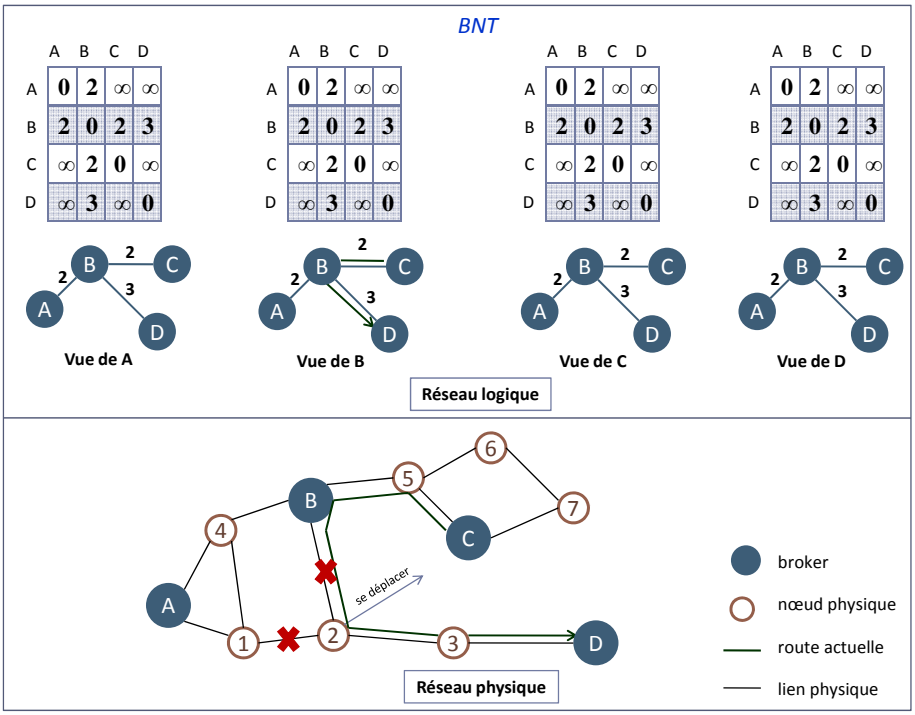

FIG. 2.12 – Topologie initiale

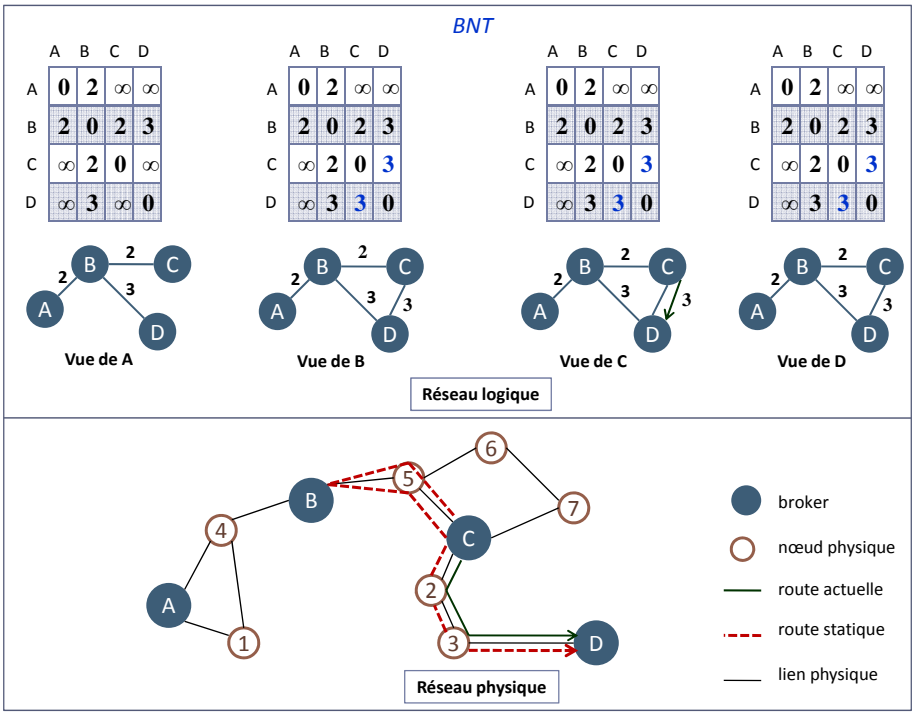

FIG. 2.13 – Topologie après le déplacement du nœud 2

Partant de la topologie initiale ou les quatre *brokers* ont le même arbre logique (*A*-*B*, *B*-*C* et *B*-*D*), le réseau subit des changements par mobilité du nœud 2. Ceci engendre la rupture des deux liaisons 2-*B* et 2-1 et l'établissement d'un nouveau lien 2-*C*. Le nœud *C*, étant le premier qui a commencé la procédure de mise à jour, s'aperçoit que le nœud *D* devient son voisin. Pour cela, il ajoute le lien *C*-*D* à sa table *BNT*, et propage ce changement à ses voisins *B* et *D*. Ces derniers mettent à jour leurs tables tout en ajoutant le lien *C*-*D*. Ensuite, les *brokers* *B*, *C* et *D* reconstruisent leurs tables *BRT* tout en se basnat sur la topologie actuelle du réseau logique composé comme suit : (*A*-*B*, *B*-*C*, *B*-*D*, et *C*-*D*). En conséquence, la route qui sera considérée afin d'acheminer les évènements entre *C* vers *D* est :$C \longrightarrow 2 \longrightarrow 3 \longrightarrow D$. Cette route est différente à celle de l'approche statique qui correspond à : $C \longrightarrow 5 \longrightarrow B \longrightarrow 5 \longrightarrow C \longrightarrow 2 \longrightarrow 3 \longrightarrow D$.

✓ Synthèse

Le système décrit précédemment propose un mécanisme d'optimisation périodique des routes tout en considérant celles les plus courtes moyennant une reconstruction dynamique

de l'arbre logique. Cependant, ce système est bâti sur une hypothèse forte qui suppose que le chemin le plus performant est toujours le plus court.

Par contre, le système n'utilise aucun mécanisme de surveillance de l'état des liens. En effet, lorsqu'un *broker* s'éloigne de ses voisins *brokers*, une dégradation de la qualité d'un ou de plusieurs liens avec ses voisins pourrait se produire. Cette dégradation n'est pas perceptible par le système et ne sera pas traitée.

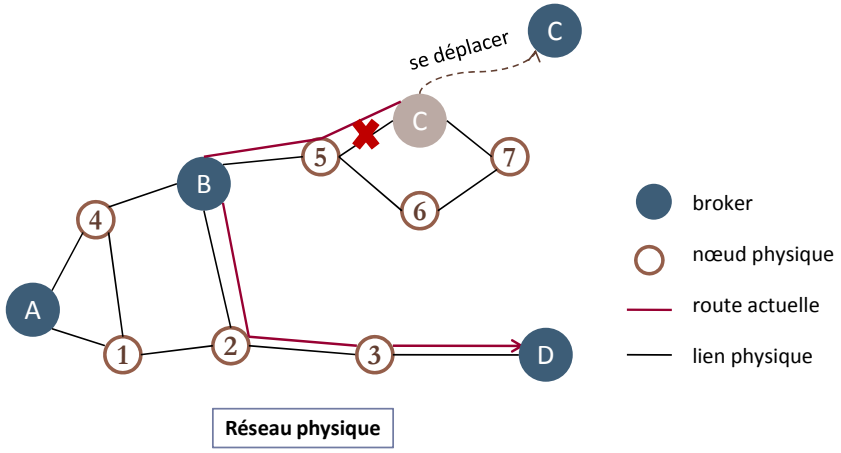

FIG. 2.14 – Problèmes posés par [CS05]

L'exemple de la Figure 2.14 fait que si le *broker* C s'éloigne de ses voisins, une dégradation de la latence du lien B-C se produit. Sur le réseau, aucune autre route plus courte n'est disponible. Face à ce problème, le système ne pourrait pas réagir et sera obligé d'adopter la route B-C même avec des performances moyennes ou médiocres en termes de latence.

– 2. Le système *Harmony*

Harmony [KKY+10, YKK+09] est un système publier/souscrire qui gère la mobilité des nœuds *brokers* en adoptant une étape d'analyse proactive. Harmony assure une gestion de la QdS en termes de latence et taux de perte. Pour achever ceci, chaque *broker* contient un agent responsable de collecter des mesures de latence et de taux de perte. Périodiquement, Harmony cherche proactivement la route optimale. Il compare la latence relative à la route actuelle avec celle assurée par le meilleur chemin. Le système échange donc de route si

la différence entre les deux mesures de latence est inférieure à un seuil prédéfini par l'application. Le seuil utilisé dans cette étape est fixé par l'application et associé à un sujet donné.

✓ Synthèse

Le système Harmony assure une satisfaction des exigences de l'application en termes de QdS notamment au niveau de la latence. Néanmoins, le système utilise des seuils figés et spécifiques à un sujet bien déterminé. De ce fait, les techniques utilisées ne pourraient pas être appliquées pour un système basé contenu. De plus, l'envoi sur les routes multiples peut causer la saturation du réseau. Finalement, l'étape de sélection de routes multiples restreint son utilisation aux topologies cycliques.

– 3. Le système *Q*

Le système *Q* [MA05] est un système basé évènement qui maintient un niveau de qualité de service acceptable moyennant des optimisations périodiques au niveau des liens logiques. *Q* s'adapte aux changements de la topologie moyennant une interaction *cross layer*. Cette interaction se résume à travers une communication établie entre le niveau logique avec le niveau réseau afin d'obtenir l'information concernant la topologie du réseau. Equipé de cette information, le système introduit des opérations de reconfiguration afin de garantir une efficacité de communication.

À travers cette communication *cross-layer*, une comparaison est affectée entre la distance logique (nombre de *brokers* intermédiaires) séparent deux nœuds avec la distance physique entre les mêmes nœuds en termes de nombre de sauts physiques. Donc, si le résultat de comparaison indique qu'il existe un chemin physique plus court (en termes de nombre de sauts) que celui du chemin logique adopté, le système procède à une substitution de ces deux liens. De cette manière, *Q* adopte toujours les liens logiques les plus courts.

La technique de détermination des longueurs des chemins logiques est achevée moyennant des messages de type *Ping* entre les producteurs et les consommateurs. Dans le système *Q*, un producteur émet périodiquement un message de type *Ping*. A la réception d'un message de type *Ping*, chaque *broker*, inclut son identifiant dans l'entête du message. Ce message est transmis de près en près vers les *brokers* voisins. De cette façon, chaque

broker sera capable à travers les messages *Ping* de déterminer son niveau dans l'arbre logique. Cette valeur offre donc au *broker* la possibilité de déterminer le nombre de sauts logiques le séparant de ses voisins. Comparée avec le nombre de sauts physiques entre les mêmes nœuds, cette valeur pourrait être un indicateur d'une optimisation possible pour le système Q. En fait, si le *broker* se rend compte à travers cette comparaison qu'une meilleure solution existe, il déclenche une opération de reconfiguration. L'exemple de la Figure 2.15 décrit cette opération.

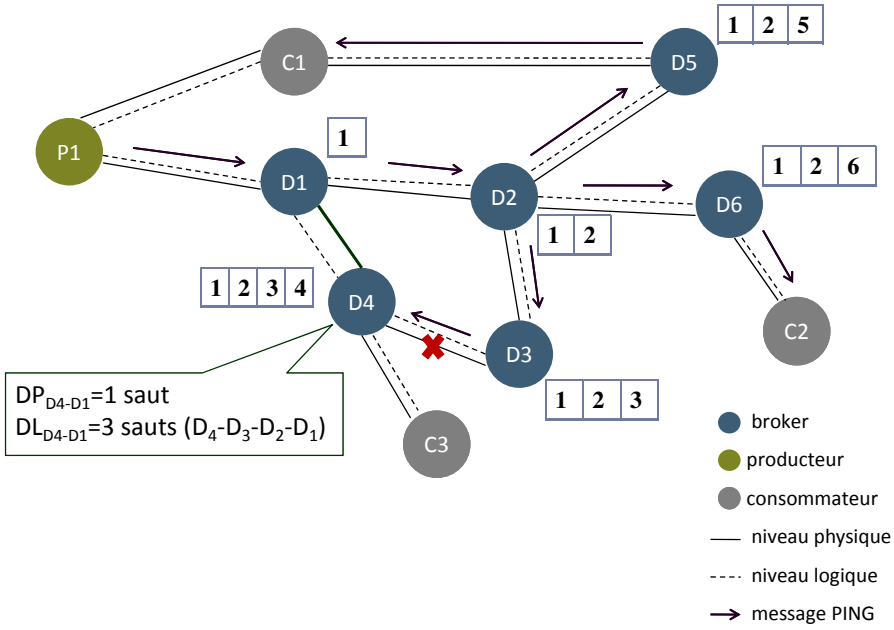

FIG. 2.15 – Optimisation des routes par cross layer dans Q

Sur cet exemple, le producteur P envoie périodiquement des messages de type *Ping* vers ses voisins. Ce message est transmis de près en près (via le lien $P - D1 - D2 - D4 - S2$) vers le consommateur S_2.

Ce dernier D_1 en consultant les identifiants figurant sur le message reçu, s'aperçoit qu'il est à trois sauts de D_1. En consultant sa table de routage, il note qu'il est à un seul saut de D1. S_2 émet donc une désouscription de D_4 et se souscrit sur D_1. Cette stratégie offre au système Q la possibilité d'améliorer les routes d'une façon périodique et donc d'économiser la latence prise par les évènements en adoptant toujours les routes présentant un minimum

de distance entre les nœuds logiques.

✓ Synthèse

Le système Q opère à travers une reconfiguration préventive afin d'adopter les routes les plus courtes du réseau. Ceci pourrait minimiser la latence d'une part, et engendrer la dégradation d'autres paramètres de QdS, d'autre part.

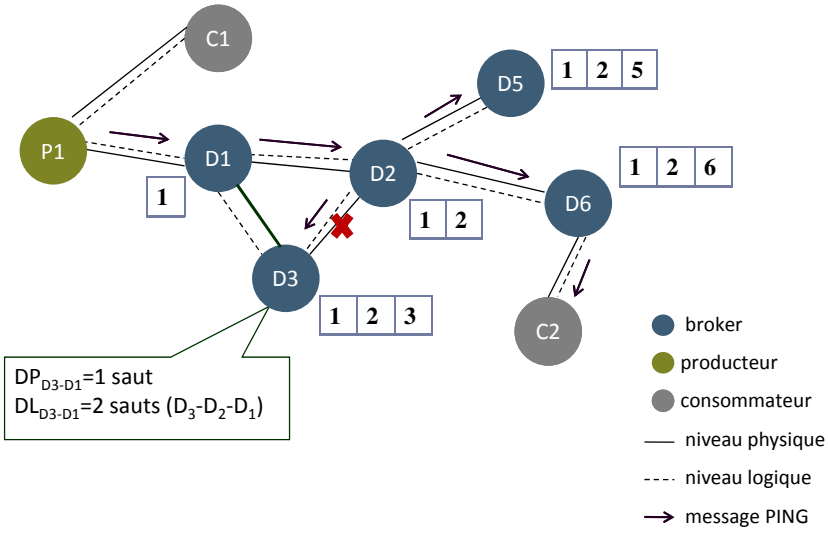

FIG. 2.16 – Limite du système Q

Dans la Figure 2.16, le *broker* D_3 se détache de son voisin D_2 et joint D_1.
Cette opération pourrait être inutile si la latence sur le chemin (D_1-D_2- D_3) est inférieure à celle trouvée par celle de la nouvelle route résultante de l'opération d'optimisation. On peut donc déduire que le système ne contient pas un module de supervision du système et considère que le seul facteur influant sur la latence est le nombre de sauts.

– 4. Le système *dynamic publish/subscribe* [TKKR09]

Dans [TKKR09], les auteurs offrent une démarche probabiliste qui permet de satisfaire les contraintes exigées par les souscrits en termes de latence. Pour ce faire, chaque client consommateur spécifie des bornes concernant la latence requise pour son besoin sur le chemin lui séparant du producteur. D'autre part, chaque consommateur maintient une connaissance sur les valeurs de latence sur chacun des liens le reliant avec ses voisins dans

le réseau logique de *broker*s. Ceci est déterminé grâce à des interactions inter-couches (*cross layer*) au sein du système. Une comparaison entre ces deux valeurs de latence (celle exigée et celle fournie par le système) mène le consommateur à calculer la probabilité de satisfaction de ses contraintes sur le chemin de bout en bout partant du producteur. L'analyse présentée dans ce travail est préventive du coup qu'elle permet au consommateur d'avoir une idée à l'avance sur le service qui pourrait être fourni par le système afin de prendre les mesures correctives nécessaires pour la satisfaction de ses contraintes.

2.3.3 Synthèse

Les méthodes de *monitoring* et d'analyse présentées proposent, d'une part, une collecte des mesures de QdS à travers des messages spécifiques de type *Ping* qui apportent un trafic supplémentaire au système. D'autre part, l'analyse pourrait suivre soit une approche réactive, soit une approche proactive. La première met en place des détecteurs de pannes actifs à l'aide des messages *Ping* et *Heartbeat*. De plus, l'analyse réactive fait appel à des seuils fixes qui ne tiennent pas compte des changements de l'environnement. L'approche proactive part du même objectif que l'analyse réactive, et propose des traitements préventifs qui calculent des valeurs estimées de certains paramètres de QdS dans le but d'anticiper les pannes.

Face aux problèmes de QdS, sans cesse grandissante, des systèmes publier/souscrire sur MANET, nous proposons une gestion de la QdS qui combine les deux aspects proactif et réactif de l'approche d'analyse.

Notre approche, contrairement à ce qui existe dans la littérature, fait recours aux messages propres du système publier/souscrire afin de collecter les mesures de QdS.

En outre, notre approche n' impose pas une forme prédéfinie d'analyse. Au contraire, nous offrons au développeur la possibilité de choisir la forme d'analyse la plus adéquate aux exigences de son application en termes de QdS.

Du point de vue langage de souscription, notre approche n'est pas restreinte à un système publier/souscrire bien déterminé, d'ailleurs, elle est générique et supporte tout type de

langage de souscription. Finalement, notre approche se base sur des seuils adaptatifs, au lieu des seuils fixes, qui tiennent en compte du contexte ad-hoc et des variations fréquentes en termes de QdS affectant ce type de réseau.

D'autres solutions dans la littérature adressent une gestion de la QdS aux niveaux des phases de *monitoring* et d'analyse à travers un équilibrage de charge.
Ces systèmes utilisent, de même que les travaux cités, des techniques de comparaison par rapport à des seuils fixes afin de détecter les dégradations de charge.
Dans ce qui suit, nous donnons un aperçu sur quelques systèmes publier/souscrire traitant la charge comme paramètre de QdS dans les réseaux MANET.

– 1. Le système *Padres*

Dans [YHA10], *Jacobson et al.* ont mis en place un système appelé *Padres*. C'est un système publier/souscrire basé contenu développé par le groupe de recherche sur les systèmes "Middleware" de l'Université de Toronto [FJLM05]. Ce système prend en charge différentes fonctionnalités. En effet, *PADRES* comporte des algorithmes de détection de pannes capables de gérer différentes classes d'échecs et de garantir le fonctionnement continu du système. Outre, ce système offre un mécanisme d'équilibrage de charge dans le réseau assurant qu'une charge excessive est également répartie entre les différents *broker*s dans le but d'améliorer la disponibilité et la robustesse du système.

En général, le composant responsable de l'équilibrage détecte non seulement la surcharge des nœuds *broker*s mais aussi le déséquilibre entre ces différents nœuds. Cette détection est assurée en passant obligatoirement par une phase de *monitoring*. Cette dernière est élaborée en ayant recours à des messages spécifiques PIE[2] transmis afin d'obtenir des informations concernant les paramètres de performances. Ces paramètres seront utilisés par la suite par les algorithmes de décharge dans le but d'établir la session d'équilibrage. Parmi ces paramètres, nous distinguons le taux d'utilisation des messages en entrée I_r. En fait, ce paramètre reflète le temps de traitement ainsi que le taux d'utilisation CPU[3] mais aussi la durée de temps pendant laquelle les messages entrants restent dans la

[2]Padres Information Exchange
[3]Central Processing Unit

file d'attente du *broker*.

$$I_r = \frac{i_r}{m_r} \tag{2.4}$$

avec I_r représente le taux des publications (en nombre de messages par secondes) et m_r présente le nombre de messages traités par seconde. A partir de là, si I_r dépasse une valeur seuil prédéfinie qui vaut 1, une panne est notifiée et une session d'équilibrage est immédiatement déclenchée.

✓ Synthèse

Padres offre un mécanisme d'équilibrage de charge intégré dans chaque *broker* incorporant un composant de *monitoring* ainsi qu'un détecteur de pannes visant à assurer un état meilleur pour tout le système. Outre, il traite aussi bien la congestion des nœuds que la saturation des liens. Néanmoins, ce système Publier/Souscrire n'est pas appliqué dans des contextes dynamiques comme les réseaux ad-hoc. De plus, *Padres* ne tient pas compte de la charge provenant des publications, en effet, le mécanisme d'équilibrage agit seulement sur les souscrits en les migrant d'un *broker* saturé à un *broker* moins chargé. D'autre part, *Padres* considère des paramètres statiques intervenant dans les calculs des métriques de performances. Ceci s'oppose à la dynamicité des réseaux ad-hoc.

– 2. Le système *HyperSub*

Dans [YZH07], *Yang et al.* ont proposé un système publier/souscrire basé contenu appelé *HyperSub*. Ce système offre la possibilité d'assurer un équilibrage de charge entre les *broker*s dans le réseau. De ce fait, le paramètre à surveiller dans ce système est la charge d'un *broker* qui est déterminée en fonction du nombre de souscriptions que le *broker* maintient dans sa table. Pour ce faire, le *monitoring* dans *HyperSub* s'effectue en se basant sur l'échange de messages spécifiques. Ainsi, pour obtenir l'information concernant la charge de ses voisins, un *broker* B_1 envoie un message contenant sa charge à l'ensemble des voisins situés à un saut ainsi que ceux situés à deux sauts de B_1. Par la suite, le nœud B_1 est désormais chargé ou saturé si sa charge dépasse une valeur seuil définie par :

$$S = \overline{L} \times (1 + \delta_n) \tag{2.5}$$

avec $\overline{L}$ dénote la moyenne des charges des voisins à un sauts et à deux sauts de B_1. Confronté avec ce problème de surcharge, le nœud B_1 choisit un ensemble de nœuds voisins non chargés auxquels il migre sa charge.

✓ Synthèse

Bien que *HyperSub* offre un mécanisme d'équilibrage de charges entre l'ensemble des *broker*s dans le réseau, la technique de *monitoring* qu'il utilise se base sur l'échange de messages spécifiques. Cet échange risque de causer un trafic supplémentaire ce qui peut engendrer la congestion du réseau. D'autre part, la technique d'analyse mise en œuvre se base essentiellement sur la comparaison par rapport à une valeur seuil fixe. Ceci reste non convenable dans des contextes dynamiques dans lesquels les métriques de performances subissent des fluctuations continues.

2.3.4 Localisation des pannes

Le module d'analyse intègre aussi un sous module de diagnostic qui cherche à localiser la panne une fois identifiée afin de remédier à la défaillance et empêcher tout échec du système.

2.3.4.1 Définition de la panne

Selon [SLM10], une panne est un évènement qui se produit dans le système et qui cause une déviation du service offert par le système par rapport au service correct. Cet évènement cause aussi une ou plusieurs performances non désirées du comportement du système. Une panne peut se produire d'une façon soudaine ou peut se produire lentement (par étapes).

Dans notre contexte, une panne est une augmentation progressive des valeurs de latence sur les liens logiques transférant les évènements.

2.3.4.2 Types de pannes dans les systèmes publier/souscrire

Une étude de l'existant nous a amené à identifier les catégories de pannes prises en compte par les systèmes publier/souscrire [CER13]. Cependant, ces systèmes ne précisent

pas la technique utilisée pour la localisation des pannes.

Trois catégories de pannes peuvent se produire dans ces systèmes (Figure 2.17).

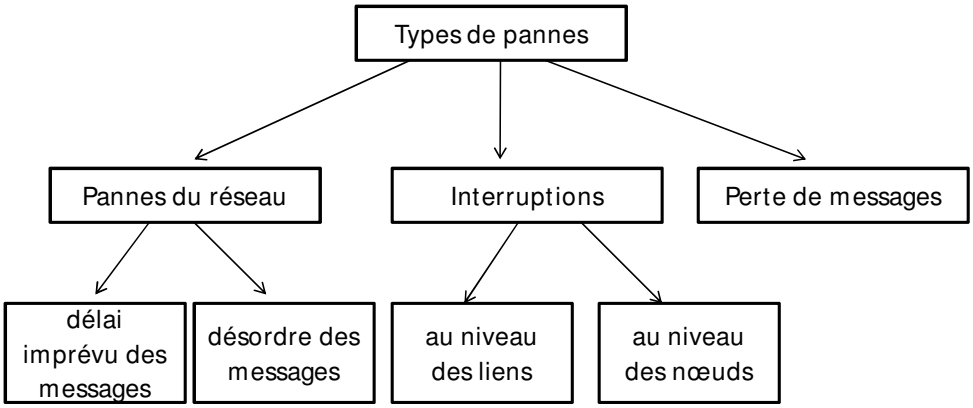

FIG. 2.17 – Types de pannes pouvant affecter les systèmes publier/souscrire

- Pannes provenant du réseau : un comportement anormal du réseau peut engendrer des pannes au sein du système publier/souscrire. Cette anomalie peut être causée par :
  - a) un délai imprévu des messages : le temps nécessaire pour échanger des évènements sur le réseau pourrait être plus grand que celui offert par le lien. Ceci pourrait survenir à cause d'une congestion ou une surcharge des nœuds ou des liens.
  - b) un désordre au niveau des messages : les messages peuvent être reçus suivant un ordre différent de celui qui été utilisé lors de l'envoi par les producteurs. En effet, les messages arrivent aux souscrits selon différents chemins. De ce fait, les messages ayant traversés les plus court chemins arrivent aux destinations avant les autres.
- Interruptions des liens ou des nœuds : les pannes matérielles ou logicielles peuvent affecter les nœuds causant ainsi un mal fonctionnement du système. De même, les liens peuvent devenir inadéquats lors de la perte de connectivité.
- Perte de messages : les évènements peuvent être perdus, soit au moment de l'envoi d'un message, de la réception d'un message ou sur le canal transférant le mesage

entre la source et la destination.

2.3.4.3 Méthodes de diagnostic

D'après la littérature, il n'existe pas de méthodes de diagnostic spécifiques pour les systèmes basés évènements. Par contre, ces méthodes ont fait preuve de leur efficacité dans d'autres domaines. Les méthodes de diagnostic ou de localisation des pannes peuvent être classées comme suit.

– 1. Diagnostic basé règles

Les techniques de diagnostic à base de règles [SPKG12, WD09] se basent sur les systèmes experts fondés sur un ensemble de règles prédéfinies par un expert du domaine. La formulation des règles se basent sur des normes et prennent la forme (si-alors). De ce fait, chaque règle est composée d'une première partie (la partie si) appelée prémisse, et d'une deuxième partie (partie alors) appelé conclusion. Par exemple, la règle suivante :

"Si le taux d'utilisation de la charge CPU dépasse la valeur 90 alors le nœud est chargé." indique que : dès que la valeur de CPU devient supérieure à 90, le système détecte une surcharge qui provient exactement de la charge CPU.

La mise en place de plusieurs règles fait appel à des méthodes d'inférence qui permettent leurs combinaisons afin de déduire la ou les causes de pannes.

✓ Synthèse

Bien que ces règles sont construites et interprétées par des experts, elles ne peuvent pas identifier des pannes imprévues ou utiliser des règles qui n'étaient pas spécifiées dans la base de règles. De ce fait, le rôle du diagnostic à base de règles reste limité, vu qu'il ne peut pas couvrir toutes les pannes possibles.

– 2. Diagnostic basé sur des méthodes statistiques

Ces méthodes de diagnostic [SPKG12, WD09] exploitent des méthodes statistiques telles que la corrélation, la comparaison des histogrammes, les théories de probabilités, etc. Les techniques de corrélation, fréquemment utilisées dans ce contexte, analysent des données historiques afin de découvrir automatiquement les relations qui existent entre

chaque paire de métriques. Plus précisément, la corrélation permet de déduire les relations de causalité qui pourraient exister entre les symptômes de pannes et les causes.

✓ Synthèse

Les méthodes statistiques doivent être utilisées avec précision. En effet, une erreur peut engendrer des fausses localisations de pannes. Par exemple, si une série de données était faussement estimée à une série normale, ceci peut engendrer des taux élevés d'erreurs de précision.

– 3. Diagnostic basé sur des machines d'apprentissage

Les machines d'apprentissage [SPKG12, WD09] se basent sur une première phase d'apprentissage et une deuxième phase de validation ou de test qui tire des résultats en se basant sur le modèle d'apprentissage. Ces techniques font recours à diverses techniques statistiques afin de localiser les pannes. En effet, afin de réaliser l'apprentissage, le système utilise des données connues de l'état de bon fonctionnement et d'échec du système, dans le but d'apprendre quelles sont les métriques les plus corrélées avec les états d'échec. Ceci permet, donc, de former une base d'apprentissage contenant les métriques ou les composants du système les plus responsables de l'occurrence de pannes. De ce fait, le système sera guidé plus tard à localiser finement les pannes affectant le système.

✓ Synthèse

Les techniques d'apprentissage permettent une localisation des pannes en identifiant les composants du systèmes ou les métriques les plus corrélés avec les états de défaillance. Cependant, ces techniques peuvent devenir non précises quand le nombre de métriques ou de symptômes devient important.

De plus, un réapprentissage est toujours requis dans la cas ou le comportement du système varie fréquemment. A défaut, le diagnostic sera erroné. De ce fait, ce type de diagnostic ne pourra pas être appliqué dans les situations ou le système change fréquemment de comportement et subit toujours des perturbations et des dégradations.

2.3.4.4 Récapitulation

Notre étude effectuée autour des méthodes de diagnostic a identifié trois classes principales : des méthodes basées sur des règles, d'autres basées sur l'apprentissage, et des méthodes basées sur des lois statistiques.
Adopter les méthodes basées règles ou celles basées sur des machines d'apprentissage dans notre contexte pourrait donner des résultats erronées dans certaines conditions. Par ailleurs, une règle qui est définie dans un contexte d'utilisation demeure inadéquate quand le réseau change d'environnement. Egalement, un apprentissage effectué à un instant t pourrait donner des résultats invalides à l'instant $t + \Delta_t$ vu la dynamicité du réseau.
Par conséquent, il serait presque impossible d'établir un modèle de référence ou d'appliquer les méthodes basés règles ou celles basées sur l'apprentissage dans un contexte caractérisé par la dynamicité et le changement fréquent de l'état du système.
Nous avons donc opté pour les méthodes statistiques afin de localiser les pannes dans notre système. Plus particulièrement, nous avons adopté la méthode de corrélation afin de mesurer l'intensité de la liaison entre la variation de la latence d'une part et les causes possibles de cette variation.

2.4 Conclusion

Dans ce chapitre, nous avons fourni un état d'art concernant les méthodes de *monitoring* et d'analyse existantes ainsi que les systèmes publier/souscrire adoptant ces méthodes.

Quant au *monitoring*, nous avons décidé de faire recours aux messages propres du système publier/souscrire pour échapper de la surcharge engendrée par les messages spécifiques tels que les messages *ping* et *hearbeat* cités dans la littérature.
De plus, la nature dynamique du réseau MANET nous a incités à adopter une méthode d'analyse qui se base sur la comparaison des paramètres de QdS par rapport à une valeur seuil. Contrairement à ce qui existe dans le littérature, cette valeur devra être dynamique pour répondre à la mobilité fréquente et au changement d'état qui pourrait affecter le

réseau ad-hoc.

Quant aux méthodes de diagnostic, une étude approfondie nous a incités à adopter les modèles de diagnostic qui font pas référence à un modèle préétabli telles que les méthodes à base de règles ou d'apprentissage vu la dynamicité du réseau MANET. Plus particulièrement, nous avons adopté une méthode statistique dans l'élaboration de notre module de diagnostic mesurant la dépendance et les relations de causalité entre les symptômes et les causes.

Dans le chapitre suivant, nous présentons les modules de *monitoring* et d'analyse englobés dans le modèle analytique orienté QdS proposé.

CHAPITRE 3

Modèle analytique orienté QdS

3.1 Introduction

Face à ses topologies dynamiques, ses capacités limités, les réseaux MANET requièrent des contraintes additionnelles par rapport aux réseaux filaires. Combinés avec l'accroissement des applications multimédias ainsi qu'une large gamme d'applications mobiles, le fort besoin de garantir la qualité de service nous a mené à proposer notre modèle analytique orienté QdS. Le modèle analytique orienté QdS est proposé comme solution aux problèmes de (QdS) des MANET et ses contraintes particulières.

L'objectif fondamental du modèle proposé est de permettre au système de se superviser et d'analyser son état afin de détecter, voire prédire, les pannes pouvant l'affecter.
Dans notre contexte, une panne est une dégradation continue de QdS qui touche les liens logiques du système. Elle représente les situations dans lesquelles les valeurs de latence dépassent d'une façon continue le seuil.

Notre modèle analytique orienté QdS [LJC15] intègre au sein de chaque *broker* deux modules interdépendants. Le premier module se charge d'exécuter la phase de *monitoring*, alors que le deuxième est responsable de la phase d'analyse visant à analyser les dégradations affectant les liens logiques séparant les *brokers* voisins au niveau middleware. Le module d'analyse comporte aussi une étape de diagnostic cherchant à identifier la cause de la dégradation.

Ce chapitre présente l'aspect théorique de nos contributions dans le cadre de cette thèse. Dans une première partie, nous exposons le module de *monitoring* proposé. En seconde partie, nous détaillons le module d'analyse avec les trois volets d'utilisation possibles à savoir l'analyse réactive, proactive et hybride.

3.2 Approche de *monitoring* proposée

En raison de la mobilité fréquente dans les réseaux MANET, le contrôle de la topologie joue un rôle primordial dans le fonctionnement du système. Ce poids accordé à la mobilité se traduit par une supervision permanente de l'état des liens à travers le calcul de la latence

comme paramètre de performance de la qualité des liens.
De ce fait, nous visons dans le cadre de cette thèse, à maintenir une connectivité permanente tout en supervisant la qualité des liens du réseau. Le but est, donc, d'atteindre un ensemble d'objectifs tels que : assurer une connectivité permanente du réseau, offrir un support de QdS acceptable pour les applications et augmenter la durée de vie du système.

Ainsi l'approche de *monitoring* que nous proposons est une approche distribuée, qui met en place au sein de chaque *broker* un composant nommé 'moniteur'. Le moniteur proposé utilise le trafic du système publier/souscrire pour mesurer les paramètres de performance du système. De ce fait, il n'ajoute pas un trafic supplémentaire de type *Ping* ou *Hearbeat*.

La latence, considérée comme un paramètre de QdS pour la surveillance de la qualité des liens, correspond au temps que requière un évènement pour transiter dans le réseau d'une source à une destination.

L'approche que nous adoptons se base donc sur des calculs locaux (càd au sein de chaque nœud *broker*) de la latence pour garantir un état meilleur du système.
La latence considérée est celle prise par un évènement pour transiter entre deux *brokers* voisins au sein du service d'évènements. Ainsi, quand un évènement E_1 transite d'un *broker* B_1 vers son voisin B_2, la latence est calculée par le *broker* récepteur B_2 à l'aide de la formule suivante :

$$Latence = t_{réception} - t_{envoi} \pm offset \tag{3.1}$$

Où :

- $t_{réception}$: est le temps de réception du message par le *broker* B_2.
- t_{envoi} : est le temps d'envoi du message par le *broker* B_1.
- *offset* : désigne la différence entre les temps indiqués par les horloges des deux *brokers*.

Dans le but de calculer l'*offset* [SBK05], le nœud B_2 envoie un message *synchronisation_pulse* à B_1 contenant la valeur de T_1. Le *broker* B_1 reçoit ce message à l'instant T_2, il envoie un message d'acquittement vers B_2, contenant T_1, T_2 et T_3. B_2 reçoit cet acquittement à l'instant T_4. En supposant que le délai de propagation et le

décalage horaire sont constants pendant une durée de temps faible, B_2 calcule le décalage d'horloge qui le sépare de B_1 moyennant l'equation 3.2 :

$$offset = \frac{(T_2 - T_1) - (T_4 - T_3)}{2} \tag{3.2}$$

Sur la Figure 3.1, T_1 et T_4 présentent les temps mesurés par l'horloge de B_2. De même, T_2 et T_3 présentent les temps mesurés par l'horloge de B_1.

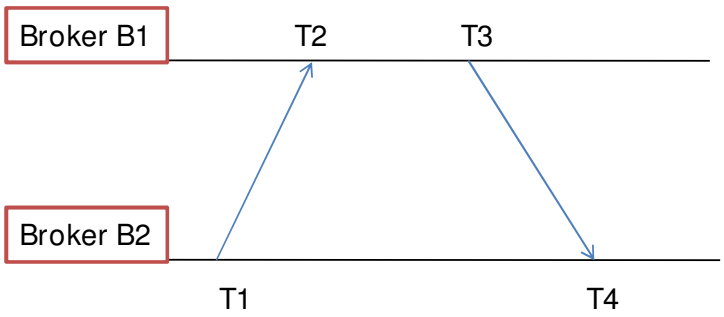

FIG. 3.1 – Synchronisation des horloges

En l'absence d'un trafic dans le système publier/souscrire, le module de *monitoring* envoie spécialement des messages vides (qui ne contiennent pas de notifications) servant à mesurer la latence éventuelle entre *brokers* voisins et à superviser le système.

A la fin de la phase du *monitoring*, chaque *broker* dispose d'un fichier Log contenant l'ID du *broker* récepteur de l'évènement, l'évènement, sa taille, le nombre de sauts physiques séparant les deux *brokers* (celui qui a envoyé le message et celui qui l'a reçu) et la latence mesurée.

3.3 Approche d'analyse proposée

Le module d'analyse développé dans le cadre de cette thèse exploite les valeurs de latence issues du module de *monitoring* afin de décider concernant l'état du système. Il permet au système de s'auto-contrôler afin de détecter les dégradations de QdS.
Chaque dégradation ou anomalie détectée sera traduite dans le cas échéant par des alarmes.

L'analyse comporte aussi une étape de diagnostic qui vise à localiser la panne une fois identifiée.

Bien que les applications cibles aient des contraintes variées en termes de QdS, notre modèle traite toutes ces applications et offre une solution particulière pour chacune. Nous ciblons ainsi toutes les applications allant des plus critiques ayant des exigences dures en QdS vers celles les plus tolérantes en QdS.

L'approche proposée offre donc au développeur une variante de fonctionnalités et lui permet un choix varié entre trois formes d'analyse.

- une analyse réactive [LKJ12] : destinée pour les applications les moins sensibles aux pannes telles que les applications multimédias, ou le vidéo-streaming. En effet, sans qualité de service, les flux vidéos se dégradent, l'image devient saccadée, la voix et le vidéo deviennent désynchronisées. Ceci fait que ces applications requirent aussi de la QdS et présentent des besoins de QdS afin d'assurer une livraison fiable des données multimédia.
- une analyse proactive [LGJC12, LMC12] : cette approche préventive est destinée surtout pour les applications les plus sensibles aux pannes telles que les applications de gestion de crise. Ces applications requièrent des garanties strictes en termes de qualité de service et présentent des contraintes qui mettent l'accent sur le délai ou encore la latence. En effet, la panne causée par un retard d'acheminement des données dans de telles situations engendre des dégâts et même des catastrophes.
- une analyse hybride [LJDC15] : c'est une combinaison entre les deux formes d'analyse précitées. Cette forme d'analyse est une variante d'amélioration de l'analyse réactive. Elle pourrait être utilisée quand le système présente des dégradations même en présence de l'analyse réactive. La solution est donc de migrer vers l'analyse proactive afin d'anticiper les pannes et stabiliser le système.

La Figure 3.2 illustre le module d'analyse proposé.

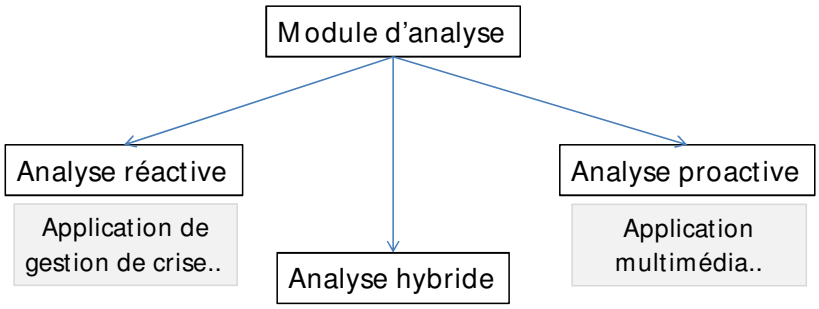

FIG. 3.2 – Module d'analyse proposé

Dans ce qui suit, nous détaillons ces trois formes d'analyse chacune à part.

3.3.1 Analyse réactive

Les applications multimédias, comme étant des applications de traitement de l'information de tout type (vidéo, texte, son...) présentent des contraintes en termes de QdS. Ces contraintes mettent l'accent sur le délai. En effet, un flux vidéo dont la voix et le son ne sont pas synchronisés est jugé de mauvaise qualité. De plus, une vidéo dont le mouvement apparaît lent entrainera la désatisfaction des clients.

Afin de remédier à ces insuffisances, nous proposons au sein de notre approche un module d'analyse réactive destiné pour cette gamme d'applications.

Nous proposons ainsi un module d'analyse réactive dont l'objectif est de détecter les dégradations de QdS affectant le système. Le détecteur de panne ainsi proposé cible les situations où les valeurs de QdS plus précisément les valeurs de latence dépassent d'une façon continue le seuil des valeurs acceptables. Ceci nous mène à utiliser les chroniques temporelles qui empêchent de considérer les violations transitoires de QdS.

L'analyse réactive proposée se base donc sur une comparaison des valeurs de latence issues à partir du module de *monitoring* par rapport à une valeur seuil adaptative. Cette valeur évolue dynamiquement au cours du temps en fonction des variations du contexte ad-hoc.

A ce stade, il est légitime de se poser la question : Comment déterminer le seuil des

valeurs de latence ?

La façon traditionnelle la plus simple est de calculer les valeurs du seuil est d'utiliser un seuil égal à la moyenne des valeurs de latence augmentés par un intervalle de confiance. Cependant, ces calculs demeurent incohérents vu que les valeurs de la latence subissent souvent des fluctuations transitoires associées à la mobilité fréquente des nœuds déclenchant par la suite un processus de recherche de routes.
La considération de ces violations et de ces valeurs non stables de latence produisent des calculs biaisés du seuil (Figure 3.3).

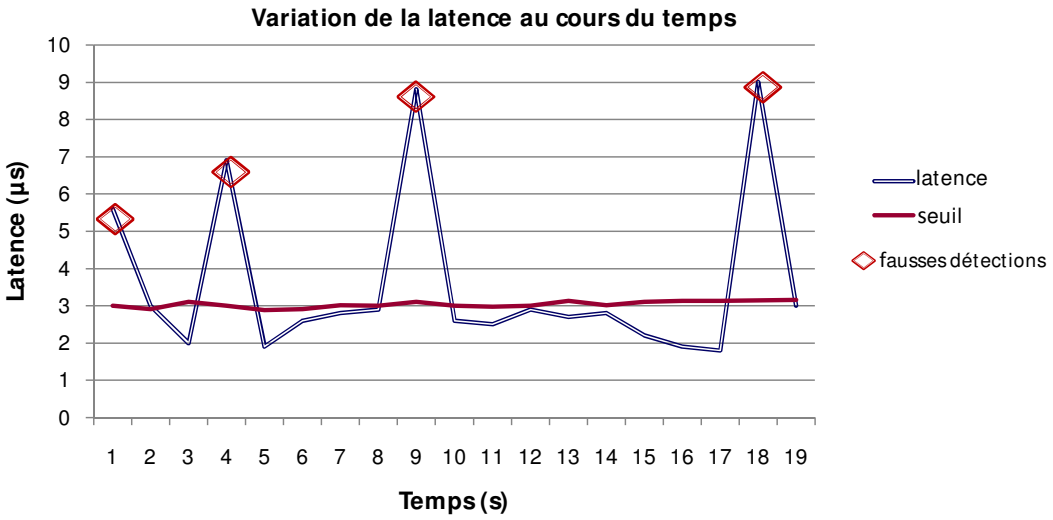

FIG. 3.3 – Fluctuations observées des valeurs de latence

L'idée donc derrière le détecteur de panne proposé est de considérer séparément les valeurs maximales de la série pour former le seuil. Ces valeurs maximales présentent une frontière infranchissable par les mesures de latence. C'est ainsi qu'apparait le besoin d'appliquer le Théorème des Valeurs Extrêmes (TVE) dans le but de produire une modélisation particulière des valeurs extrêmes de la série temporelle formée par les valeurs de latence.

Le Théorème des Valeurs Extrêmes permet d'étudier d'une façon quantitative les enlèvements et les baisses associées à la série et représentant les valeurs extrêmes d'une distribution [Lon95]. Suivant cette théorie, une nouvelle série représentant les valeurs maximales de la distribution est formée.

Cette nouvelle série formée des valeurs maximales de latence, pourrait être modélisée à l'aide d'une des trois distributions suivantes :*Gumbel*, *Fréchet* et *Weibull* [KN00] appelées respectivement valeur extrème type I, II, et III. Ces modèles ont été largement utilisés dans les domaines de finance, d'économie, de télécommunications, ...

Il est important de noter que les lois de *Gumbel* et *Fréchet* servent à modéliser les valeurs extrèmes les plus larges (valeurs maximales), tandis que la loi de *Weibull* est reliée aux valeurs minimales d'une série donnée. Ainsi, les lois qui peuvent modéliser les valeurs maximales de latence sont la loi *Gumbel* et la loi de *Fréshet*.

La représentation graphique de la variable réduite en fonction de la série des valeurs maximales est un bon indicateur de la loi adéquate (voir Section 3.3.1.1). En effet, si les points représentant les valeurs maximales de la distribution sont approximativement alignées, alors ces valeurs peuvent être modélisées à l'aide de la loi de *Gumbel*. Si en revanche, les valeurs maximales ont tendance à se disperser vers le haut, les données sont vraisemblablement issues d'une loi de *Fréchet*.

Après ajustement de la série des valeurs maximales à la loi de correspondante, nous pouvons calculer la valeur initiale du seuil à travers des formules mathématiques liées à la loi.

Une fois obtenue, cette valeur initiale du seuil est mise à jour dynamiquement moyennant la formule de la Moyenne Mobile Exponentielle. Nous formons ainsi une série de seuils que nous appelons le seuil des valeurs maximales.

Toutes ces opérations mettent en place deux phases : une phase à régime transitoire et une phase à régime permanent.

3.3.1.1 Phase à régime transitoire

Cette phase a pour finalité de déterminer la loi correspondante à la série des valeurs maximales afin de calculer la valeur initiale du seuil. La longueur de cette phase représente le nombre d'échantillons nécessaires afin de démontrer l'adéquation de la série des valeurs maximales de latence à la loi.

1. Extraction de la série des valeurs maximales

Notons par Latency(t) la série formée par les mesures de latence, issues du module de *monitoring*. Tout d'abord, nous décomposons la série originale des valeurs de latence en un ensemble de périodes de longueurs égales, noté T. Chaque période contient un nombre N d'échantillons, fixé à base d'expérimentations. L'extraction de la série des valeurs maximales de latence à partir de la série originale des mesures de latence est achevée à l'aide de la formule suivante :

$$(QdS_{max})_i = \max_k latency_{(i,k)} \tag{3.3}$$

avec k dans *[1..N]*, et sachant que :

- $(QdS_{max})_i$ est la valeur maximale de la latence extraite sur chaque période i. (i varie entre 1 et le nombre de périodes dans la série des valeurs de latence.)
- $latency_{(i,k)}$ est la valeur mesurée de la latence.

2. Détermination de la loi correspondante à la série des valeurs maximales : la loi de *Gumbel*

Le but de cette étape est de déterminer la loi correspondante à la série des valeurs maximales de latence et de s'en servir pour calculer la valeur seuil initiale.

Afin de déterminer la loi correspondante à la série des valeurs maximales de latence ; il fallait d'abord trier les valeurs maximales de la latence récupérées sur chaque période T suivant un ordre croissant.

Ensuite, nous attribuons un rang r à chaque valeur de la série. Finalement, on calcule, pour chaque valeur de rang, une fréquence empirique moyennant la formule suivante :

$$F_e = \frac{r - 0.5}{\acute{n}} \tag{3.4}$$

Avec :

- r est le rang attribué à chaque valeur maximale de latence.
- $\acute{n}$ est le nombre d'échantillons formant la série des valeurs maximales.

Ceci nous mènera à calculer la variable réduite U définie par :

$$U = \ln(-\ln(F_e))) \tag{3.5}$$

A la fin de cette étape, nous modélisons les couples (U, QdS_{max}) par une représentation graphique. Cette modélisation est très importante dans l'étape de détermination de la loi des maximums. En effet, si les points du graphique sont approximativement alignés, alors la serie des valeurs maximales correspond à la loi de *Gumbel*. Au contraire, si les points du graphique ont tendance à se disperser vers le haut, alors la série des valeurs maximales pourrait être modélisée à l'aide de la loi de *Fréshet*.

La droite obtenue dans notre cas, modélisant le couple (U, QdS_{max}), est un bon indicateur de l'adéquation de la série des valeurs maximales à la loi de *Gumbel*. Cette doite s'ecrit sous la forme suivante :

$$QdS_{max} = S * U + x_0 \tag{3.6}$$

Le gradex S et le paramètre de position [bib] x_0 représentent les deux paramètres de la loi. Ces deux paramètres sont utiles pour le calcul de la valeur initiale du seuil.
Sur un graphique de *Gumbel*, nous pouvons retrouver les paramètres de la loi comme suit :

- pour $U = 0$: on a $QdS_{max} = x_0$.
- la pente de la droite est égale au gradex S.

Le calcul des paramètres de la loi de *Gumbel* est aussi possible à l'aide de la méthode des moments qui consiste à confondre les moments des échantillons avec ceux théoriques de la loi.

A l'aide de la méthode des moments, les paramètres x_0 et S seront exprimés à l'aide des formules suivantes :

$$x_0 = \mu - S\gamma \tag{3.7}$$

$$S = \frac{\sqrt{6}}{\pi}\sigma \tag{3.8}$$

avec :

- σ est l'écart type des valeurs composant l'échantillon des valeurs maximales de latence.
- μ est la moyenne de l'échantillon des valeurs maximales de latence.
- $\gamma = 0.5772$ est la constante de l'Euler.

Une fois calculés soit à l'aide de la méthode graphique ou à l'aide de la méthode des moments, ces paramètres nous serviront à déduire les valeurs seuils initiales de la distribution de *Gumbel* étudiée. De ce fait, la valeur du seuil associée à cette distribution formée par les valeurs maximales est exprimée par l'Equation suivante [GN08] :

$$S_{QdS_{max}} = x_0 - S\ln[\ln(\frac{1}{p})] \tag{3.9}$$

Où p est la probabilité de non dépassement du seuil.

En conclusion, la phase à régime transitoire est clôturée par le calcul de la valeur initiale du seuil correspondant à la distribution formée par les valeurs de latence.
Cette valeur servira dans la phase à régime permanent pour tirer les valeurs du seuil à chaque instant.
Le recours à une phase à régime transitoire nécessite aussi la mise en œuvre d'un mécanisme de détection de panne pendant cette phase. Le modèle utilisé se base sur un seuil égal à la moyenne des valeurs de latence augmentée par un intervalle de confiance. Ceci nécessite que la série des valeurs moyennes extraites à partir de la série des valeurs de latence suivrait la loi Gaussienne.

3. Test d'adéquation de la série des valeurs moyennes à la loi de *Gauss*

Nous procédons d'abord à une extraction des valeurs moyennes de la série formée par les valeurs de latence à l'aide de la formule suivante :

$$QdS_{moy\,i} = \frac{1}{N} \sum_{k=1}^{N} latency_{i,k} \tag{3.10}$$

avec k dans *[1..N]*, et sachant que :

- $QdS_{moy\,i}$ est la valeur moyenne de la latence extraite sur chaque période i. (i varie entre 1 et le nombre de périodes dans la série des valeurs de latence.)
- $latency_{i,k}$ est la valeur mesurée de la latence.
- N est le nombre d'échantillons par période.

Ensuite, nous rangeons ces valeurs moyennes dans l'ordre croissant, nous obtenons ainsi une nouvelle série notée $QdS_{moy_{triés}}$. La moyenne de la série ainsi formée est calculée moyennant la formule suivante :

$$\overline{QdS_{moy_{triés}}} = \frac{1}{\acute{n}} \sum_{k=1}^{\acute{n}} QdS_{moy\,i} \tag{3.11}$$

Où $\acute{n}$ est le nombre d'échantillons formant la série des valeurs moyennes.

Dans le but de réaliser le test de normalité de la série des valeurs moyennes, nous nous sommes basés d'abord sur le test de Chapiro et Wilk. Ce test se base sur la comparaison d'une valeur W avec une valeur W_{crit} extraite de la table de Chapiro et Wilk (Table 2 (Annexe .1)) tout en adoptant un risque compris entre 1% et 5%. Par conséquent, si W est supérieur à W_{crit}, l'hypothèse de normalité sera valide. Sinon, l'hypothèse de normalité sera rejetée.

Le nombre W, utilisé dans le test de Chapiro et Wilk, est calculé à l'aide de la formule suivante :

$$W = \frac{\sum_{j=1}^{q} (a_j \times d_j)^2}{Z} \tag{3.12}$$

sachant que :

- q est la partie entière du rapport $\acute{n}/2$, avec $\acute{n}$ est le nombre d'échantillons des valeurs

moyennes.

- les valeurs a_j sont extraites de la Table 1 de l'annexe .1.
- Z est calculé à l'aide de la formule :

$$Z = \sum_{j=1}^{\acute{n}} (QdS_{moy_j} - \overline{QdS_{moy_{triés}}})^2 \tag{3.13}$$

- d_j est calculé comme suit :

$$\begin{aligned} d_1 &= QdS_{moy_{\acute{n}}} - QdS_{moy_1} \\ d_2 &= QdS_{moy_{\acute{n}-1}} - QdS_{moy_2} \\ &\vdots \\ d_j &= QdS_{moy_{\acute{n}+1-j}} - QdS_{moy_j} \end{aligned} \tag{3.14}$$

Ces différences sont calculées avec $1 \leq j \leq q$ et

Le test de normalité est achevé sur les cinq premiers échantillons. Le même processus est déroulé sur les échantillons qui suivent. En s'appuyant sur le théorème de la limite centrale [Jol07], on pourrait affirmer que le reste de la distribution formée par les valeurs moyennes pourrait être ajusté à la loi de *Gauss* de paramètres :

- μ : moyenne calculée comme suit :

$$\mu_{QdS_{moy}} = \frac{1}{\acute{n}} \sum_{i=1}^{\acute{n}} QdS_{\ moy_i} \tag{3.15}$$

- σ : valeur d'écart type calculé à l'aide de la formule suivante :

$$\sigma_{QdS_{moy}} = \sqrt{\frac{1}{\acute{n}} \sum_{i=1}^{\acute{n}} (QdS_{\ moy_i} - \mu_{QdS_{moy}})^2} \tag{3.16}$$

4. Détermination de la valeur du seuil correspondant à la distribution *Gaussienne*

Après avoir montré l'adéquation de la distribution formée par l'ensemble des valeurs moyennes à l'instant t_i à la loi de *Gauss* de moyenne $\mu_{QdS_{moy}}$ et d'écart type $\sigma_{QdS_{moy}}$, reste à calculer l'intervalle de confiance associé comme présenté dans la formule 3.17 :

$$intervalle\_confiance(t_i) = [\mu_{QdS_{moy}}(t_i) - t_\alpha \times \frac{\sigma_{QdS_{moy}}(t_i)}{\sqrt{g}}, \mu_{QdS_{moy}}(t_i) + t_\alpha \times \frac{\sigma_{QdS_{moy}}(t_i)}{\sqrt{g}}] \tag{3.17}$$

Avec

- $\sigma_{QdS_{moy}}(t_i)$ et $\mu_{QdS_{moy}}(t_i)$ sont respectivement l'écart type et la moyenne de l'ensemble des échantillons formant l'historique de l'instant t_i.
- g est le nombre d'échantillons formant l'historique de l'instant t_i
- t_α est la constante extraite de la table de Student (Voir Annexe .2)

En conclusion, nous obtenons le seuil à l'instant t_i calculé en sommant la moyenne et l'intervalle de confiance associés à l'instant considéré.

$$Seuil(t_i) = \mu_{QdS_{moy}}(t_i) + intervalle\_confiance(t_i) \tag{3.18}$$

5. Détection de pannes dans la phase à régime transitoire

La détection de pannes durant la phase à régime transitoire est achevée en comparant la valeur de la latence avec la valeur du seuil déterminé à l'aide de la Formule 3.18.

De ce fait, si la valeur de la latence à l'instant t_i est inférieure au seuil, le système est dans un bon état. Sinon, le système signale la présence d'une violation.
Un cumul de violations pourrait déclencher une alarme dans le système signalant la présence de pannes durant la période de traitement. Le nombre de violations tolérés est fixé par expérimentation.

3.3.1.2 Phase à régime permanent

Dans cette phase à régime permanent, la détection des dégradations de QdS affectant la série des valeurs de latence est réalisée en comparant les valeurs réelles de latence avec celle du seuil. Nous rappelons que nous avons procédé au calcul de la valeur initiale du seuil durant la phase à régime transitoire.
La détection est effectuée sur des périodes de temps. Durant chaque période, nous faisons l'extraction de la série des valeurs maximales afin de calculer la valeur du seuil relative à chaque période. Une fois calculée, nous comparons la valeur du seuil avec les valeurs de la latence de la même période.

La phase à régime permanent est illustrée par la Figure 3.4 :

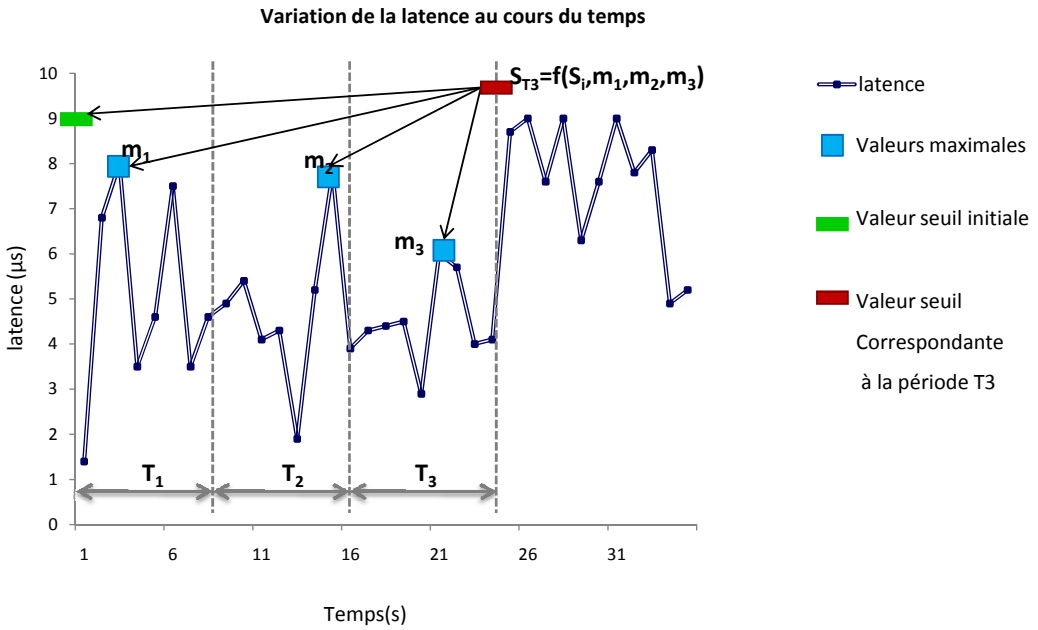

FIG. 3.4 – Phase à régime permanent

Durant cette phase, nous avons utilisé des seuils adaptatifs qui prennent en compte des dernières valeurs réelles de la latence. Ceci a pour objectif d'avoir plus de précision dans les détections de pannes et d'éviter de considérer des seuils fixes qui pourraient devenir inadéquats après une période de temps dans un contexte fréquemment mobile.

Le recours à un seuil adaptatif durant la phase à régime permanent nous a incité à utiliser la technique de la Moyenne Mobile Exponentielle pondérée $EWMA$[1] qui met à jour la valeur du seuil sur chaque période de temps T. Cette technique permet de déduire la valeur seuil à l'instant t en tenant compte des valeurs précédentes [LS90] :

$$Seuil_{max_i} = \lambda \sum_{j=0}^{i-1} (1-\lambda)^j \, QdS_{max_{i-j}} + (1-\lambda)^i S_{QdS_{max}} \tag{3.19}$$

Avec :

- $Seuil_{max_i}$ est la valeur du seuil correspondante à la période i.
- i est l'indice de la période.
- $QdS_{max_{i-j}}$ est la valeur maximale de latence obtenue sur la période $i-j$.
- $S_{QdS_{max}}$ est la valeur du seuil initial définie par la formule 3.9.
- λ est le poids qui permet de pondérer soit la valeur initiale du seuil soit les valeurs précédantes du seuil. λ est compris entre 0 et 1.

Donc à la fin de chaque période T, nous procédons à une comparaison des valeurs de latence relatives à chaque instant avec la valeur du seuil associée à la même période.

Dans le but d'éviter les fausses détections, nous avons recours à la notion de chroniques temporelles. De ce fait, nous considérons que M dépassements successifs de la valeur du seuil maximal est un signe d'une panne affectant les liens logiques entre les *brokers* voisins au sein du système.

Afin de calculer une valeur pertinente de M, nous avons utilisé le fait que la probabilité d'avoir M mesures de latence en état de violations doit être inférieure ou égale à une valeur de risque spécifiée par l'utilisateur [HGDJ08].

$$P[MsuccLV] \leq Risk \tag{3.20}$$

Soit :

- M : le nombre de violations successives menant à une détection de la dégradation de

[1] Exponentially Weighted Moving Average

QdS.

- LV : correspond à la latence mesurée supérieure au seuil.
- LOK : corresponds à la valeur mesurée au dessous du seuil.
- *Risk* est la probabilité de défaillance spécifiée par l'utilisateur.

La probabilité que le message suivant soit en état de violation noté LV, est égale à la probabilité d'être dans un état acceptable LOK multiplié par la probabilité de transition de l'état LV, plus la probabilité d'être à l'état LV multiplié par la probabilité d'être dans le même état à l'instant prochain.

$$P[anystate \rightarrow LV] = P[LOK] * P[LOK \rightarrow LV] + P[LV] * P[LV \rightarrow LV] \tag{3.21}$$

De plus :

$$P[MsuccLV] = P[anystate \rightarrow LV] * P[LV \rightarrow LV]^{M-1} \tag{3.22}$$

En remplaçant $P[Msucc \rightarrow LV]$ dans la première equation, nous obtenons l'expression donnant la plus petite valeur de N :

$$P[anystate \rightarrow LV] * P[LV \rightarrow LV]^{M-1} \leq (Risk) \tag{3.23}$$

D'ou, nous déduisons : $M \geq \lambda$ avec

$$\lambda = 1 + \frac{ln\frac{Risk}{P[anystate \rightarrow LV]}}{ln(P[LV \rightarrow LV])} \tag{3.24}$$

En choisissant une valeur de M supérieure à la plus petite valeur trouvée, nous obtenons une précision de la détection.

3.3.1.3 Synthèse

L'analyse réactive proposée dans le cadre de cette thèse se base sur la comparaison de la série temporelle formée par les valeurs de latence avec les valeurs du seuil des valeurs maximales dans le but de détecter les dégradations de QdS pouvant affecter le système.

Le seuil est calculé en se basant sur le Théorème des Valeurs Extrêmes et mis à jour dynamiquement à l'aide de la Formule de la Moyenne Mobile Exponentielle.
Donc, si les valeurs de latence dépassent successivement le seuil, le module d'analyse signale une alarme présentant une dégradation de QdS affectant le lien logique en question. Les chroniques temporelles utilisés permettent une détection plus précise et éliminent les fausses détections.

3.3.2 Analyse proactive

Certaines applications, telles que les applications de gestion de crise, sont sensibles aux pannes et leurs productions pourraient mener le système à des situations graves voire même catastrophiques. Dans ce contexte, notre approche propose un module d'analyse proactive qui servira à prédire les pannes et empêcher leur production dans un système présentant des risques importants.
L'analyse ainsi proposée est préventive du coup qu'elle se base sur une comparaison de la valeur de la latence estimée avec la valeur du seuil estimée à un instant futur.
Une étude de l'existant [CHA, SLM10] a montré qu'il existe plusieurs méthodes statistiques de prévision. Ces méthodes ont fait leurs preuve dans d'autres domaines et peuvent être classées sous deux grands axes :

- Méthodes qualitatives [LTJ12] : ces méthodes font appel à une méthodologie non mathématique. Elles utilisent des données provenant de l'expérience pour former une base de connaissance utile pour réaliser des prévisions.
- Méthodes quantitatives [Wal12] : ces méthodes utilisent les observations du passé de la variable pour prévoir son futur. Elles sont principalement utilisées pour réaliser des prévisions à court terme. Parmi ces méthodes : la régression linéaire [Nau15], la modélisation ARIMA [Tsa08].

Afin d'estimer les valeurs de latence, nous avons eu recours à des techniques de prévisions quantitatives. En effet, la mobilité, comme étant la caractéristique majeure du réseau ad-hoc, introduit des changements fréquent du comportement du système. Ce qui

rend impossible l'utilisation des méthodes qualitatives.
En particulier, nous avons utilisé la méthode d'Auto Régression linéaire, dans un premier lieu, et la méthode ARIMA *(Integrated Weigted Moving Average Formula)* [Tsa08], dans un second lieu. L'utilisation de ces méthodes d'estimation nécessite le recours à une fenêtre d'historique formée par les valeurs de latence à des instants précédents sur un intervalle de temps noté Δt_d.

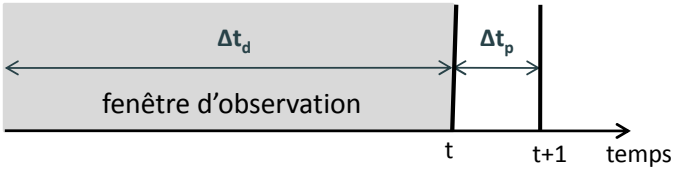

FIG. 3.5 – Relation entre le temps actuel et le temps de prédiction

Le module d'analyse proactive intégré dans notre modèle analytique orienté QdS est responsable de calculer la valeur de la latence à l'instant futur $t + 1$ à partir de celle à l'instant présent t et de comparer cette valeur avec le seuil correspondant au même instant.

Pour ce faire, nous avons utilisé des seuils adaptatifs qui s'ajustent dynamiquement en réponse à la dynamicité du réseau. Pour cela, nous avons utilisé la Méthode de la *Moyenne Mobile Exponentielle*. Cette dernière s'appuie sur les valeurs de la latence stockées dans le fichier log ainsi que les valeurs du seuil calculées précédemment afin de déterminer la valeur du seuil à l'instant $t + 1$.

Doté des valeurs estimées, le module d'analyse se charge de comparer ces deux valeurs pour déduire l'état futur du lien logique et par conséquent prévenir sa défaillance. L'approche ainsi développée a pour but d'anticiper les pannes et garantir le bon fonctionnement du système.

L'analyse proactive met en place deux phases : une phase à régime transitoire, là où nous collectons les valeurs de la latence pour former une base capable d'effectuer les estimations, et une phase à régime permanent durant laquelle nous effectuons les estimations et les prédictions.

3.3.2.1 Phase à régime transitoire

L'estimation des valeurs de la latence ne peut s'effectuer qu'après le stockage des valeurs de latence pour former une base d'estimation. Donc le but de cette phase est de collecter suffisamment de valeurs de latence afin d'effectuer les prédictions.

Durant cette phase, le système peut subir des fluctuations des valeurs de latence engendrant des dégradations de la qualité du lien entre les *brokers*. Ceci nous a incité à proposer une méthode de détection de pannes afin de réagir face aux dégradations de QdS. Pour ce faire, nous nous sommes basés sur le même principe de la méthode de détection utilisée dans la phase à régime transitoire de l'analyse réactive expliquée précédemment.
Notre objectif dans cette étape est de s'assurer de l'adéquation de la série des valeurs moyennes de la latence à la loi Gaussienne. Une fois que nous obtenons les 5 premières valeurs de la moyenne des valeurs de latence, le test de normalité est entamé à l'aide du test de chapiro et Wilk. Le calcul du seuil est donc possible à partir de la cinquième valeur de latence. A l'aide du théorème de la limite centrale, nous affirmons que la série considérée est ajustée à la loi Gaussienne. Dés lors, il est possible de comparer les valeurs de latence par rapport à la valeur du seuil calculé à l'aide de la formule (3.18).

3.3.2.2 Phase à régime permanent

Comme expliqué précédemment, l'analyse proactive s'appuie sur des méthodes statistiques afin d'estimer les valeurs de latence, ainsi que celle du seuil, et finit par comparer ces deux valeurs afin de prédire une panne éventuelle.
Le digramme présenté à la Figure 3.6 montre les étapes constituant cette phase.

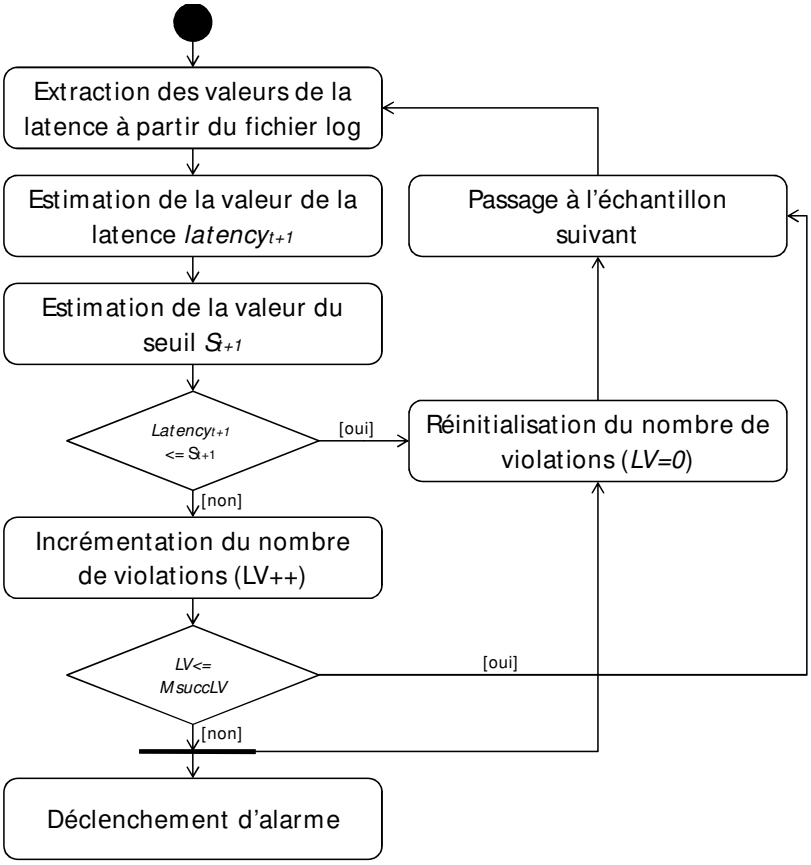

FIG. 3.6 – Phase à régime permanent

1. Estimation de la valeur de la latence

– 1.Estimation basée sur l'Auto-Régression Linéaire

L'Auto-Régression [eBR06] linéaire est une méthode statistique qui permet de prédire la valeur prochaine d'une variable tout en se basant sur une série temporelle de données. En appliquant la formule d'Auto-Régression linéaire sur la valeur de la latence mesurée entre deux *brokers* noté $latency_t$, nous pouvons calculer la valeur future de la latence qui s'écrit comme suit :

$$\widehat{latency_{t+1}} = b_0 + \sum_{i=1}^{c} b_i \times latency_t + e_(t+1) \tag{3.25}$$

avec :

- b_i sont les coefficients d'Auto-Régression,
- c est l'ordre de l'Auto-Régression représentant le nombre d'observations à considérer,
- $e_(t+1)$ est un termes d'erreur.

$$e_(t+1) = latency_t - \widehat{latency_t} \tag{3.26}$$

- $latency_t$ est la latence à l'instant $t - i$.

Afin d'estimer la prochaine valeur de la latence noté $\widehat{latency_{t+1}}$, nous nous sommes basées sur une fenêtre d'observations glissante dans l'axe du temps dans le but de tenir compte des dernières valeurs de latence. La taille de cette fenêtre est fixée par expérimentation.

L'étape de détermination de l'ordre c de l'Auto-Régression est importante vu qu'elle permet d'identifier la dimension temporelle offrant le moyen d'obtenir de bonnes estimations sur les valeurs prochaines. Une étude faite nous a permis de choisir le critère *Akaike Information Criterion* AIC [Kni89] pour l'estimation du bon ordre c de l'Auto-Régression. Ce critère permet d'estimer l'ordre c qui se base sur le choix des minimum des AIC.
Ainsi, selon ce critère, c vérifie :

$$c = arg\min AIC_c \tag{3.27}$$

Les AIC de chaque ordre variant entre 1 et 10 sont calculés à l'aide de la formule :

$$AIC_c = -2I[\ln(\hat{\sigma_c})^2] + 2c \tag{3.28}$$

Où

- I est la période d'observation càd le nombre d'échantillons considérés pour réaliser les estimations.
- $\hat{\sigma_c}$ est calculé comme suit :

$$\hat{\sigma_c}^2 = (I - c - 1)^{-1} \sum_{t=c}^{I} {e_t}^2 \tag{3.29}$$

L'application de l'équation 3.28, nous permet d'obtenir les valeurs adéquates à chaque ordre. Nous choisissons donc la valeur minimale des AIC afin de préciser le bon ordre.

Cette première étape est clôturée par la détermination de l'ordre c, nous passons ensuite à la deuxième étape permettant de déterminer les coefficients de l'Auto-Régression linéaire.

◦ Calcul des coefficients de l'Auto-Régression

Les coefficients d'Auto-Régression faisant la forme d'un vecteur b dans l'équation 3.25, sont calculés à l'aide de la méthode des moindres carrés [Ber86] basée essentiellement sur les équations de Yule-Walker.

Afin de déterminer le vecteur b, nous avons exploité l'estimateur des moindres carrées représenté par l'équation 3.30.

$$b = (X^I X)^{-1} X^I L \tag{3.30}$$

Avec

- L : est un vecteur formé par les valeurs mesurées de latence de $c + 1$ à n (n est le nombre d'échantillons formant la fenêtre d'observations. $n = 50$ dans notre expérimentation).
- X : est une matrice s'écrivant comme suit :

$$\mathbf{X} = \begin{pmatrix} 1 & latency_c & latency_{c-1} & \dots & latency_1 \\ 1 & latency_{c+1} & latency_c & \dots & latency_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & latency_{n-1} & latency_{n-2} & \dots & latency_{n-c} \end{pmatrix}$$

Finalement, moyennant l'équation 3.30, nous pouvons déduire les coefficients de l'Auto-Régression linéaire.

Dotée de tous les paramètres de l'équation 3.25, cette partie est clôturée par le calcul des valeurs estimées de la latence à l'aide de la méthode d'Auto-Régression linéaire.

– 2.Estimation basée sur la méthode ARIMA

La méthode ARIMA est l'abréviation de *Auto-Régression à Moyenne Mobile Intégrée*. C'est une méthode de prédiction statistique qui a pour but de calculer finement les erreurs de prédictions afin de produire des prédictions plus précises.

ARIMA représente un cas particulier de la méthode d'Auto-Régression à Moyenne Mobile (ARMA (f,r)) qui a la forme suivante :

$$(1 - \sum_{i=1}^{f}(\varphi_i H^i))latency_t = (1 + \sum_{i=1}^{r}\theta_i H^i)\varepsilon_t \tag{3.31}$$

avec :

– φ_i sont les coefficients de la partie AR (Auto-Régression).

– θ_i sont les coefficients de la partie MA (Moyenne Mobile).

– ε_t est un termes d'erreur.

– H est appelé opérateur de retard défini comme suit :

$$H^i latency_t = latency_{t-i} \tag{3.32}$$

avec

$$H^0 = 1 \tag{3.33}$$

Un modèle ARIMA(c,d,r) est étiqueté des trois paramètres :

– f : le nombre de termes auto-régressifs.

– r : le nombre de moyennes mobiles.

– d : le nombre de différenciations.

Dans notre étude, nous avons considéré un niveau de différentiation égal à 1. D'ou, ARIMA (f,1,r) est équivalent à :

$$latency_t = \mu + latency_{t-1} + \sum_{i=1}^{f}(\varphi_i) * (latency_{t-i} - latency_{t-i-1}) + \varepsilon_t + \sum_{i=1}^{r}(\Theta_i * \varepsilon_{t-i}) \quad (3.34)$$

avec :

– la constante μ est un facteur défini par la formule suivante :

$$\mu = Mean * (1 - \sum_{i=1}^{f}(AR(f)) \quad (3.35)$$

Où :

$$Mean = (1/I - 1) \sum_{i=1}^{I-1}(latency_i - latency_{i-1}) \quad (3.36)$$

avec I est le nombre d'échantillons considérés pour réaliser les estimations.

Finalement, en appliquant la formule 3.34, nous obtenons les valeurs de latence estimées à l'instant futur à l'aide de la Méthode ARIMA.

– 3.Estimation de la valeur du seuil

Afin de suivre les variations dynamiques de l'environnement, nous avons recours à des seuils adaptatifs. Cette adaptation fait appel, à la Méthode de la Moyenne Mobile Exponentielle EWMA [LS90]. Par définition, la méthode EWMA calcule les valeurs futures du seuil en partant d'une combinaison entre la dernière valeur réelle d'observation avec la dernière valeur prédite du seuil calculée à l'instant t-1.

A l'aide d'une moyenne pondérée de ces valeurs, EWMA calcule la valeur prochaine du seuil S_{t+1} :

$$S_{t+1} = \alpha latency_t + (1 - \alpha)S_t \quad (3.37)$$

Où

– α : constante souvent comprise entre 0 et 1. α permet de pondérer soit la valeur de la latence, soit la valeur du seuil. (dans notre cas, α est égale à 0,25).

– $latency_t$: la valeur mesurée de la latence à l'instant t.

– S_t : la valeur du seuil à l'instant t.

3. Prédiction de pannes

Une fois calculée, la valeur prédite du seuil est comparée avec la valeur du seuil correspondante au même instant.

Pour déclencher une alarme signalant la présence d'un risque éventuel, nous avons utilisé la notion de chroniques temporelles. Considérons donc qu'une panne ne résulte qu'après M dépassements successifs du seuil. Afin d'avoir une valeur précise de M, nous avons utilisé l'inégalité suivante :

$$P[MsuccLV] \leq (1 - accurancy) \tag{3.38}$$

La valeur de la précision notée *accurancy* est fixée par expérimentation. En effectuant le même calcul que celui fait par les équations 3.21, 3.22, 3.23, nous obtenons :

$M \geq \lambda$ avec

$$\lambda = 1 + \frac{ln\frac{accurancy}{P[anystate \rightarrow LV]}}{ln(P[LV \rightarrow LV])} \tag{3.39}$$

En atteignant la valeur M de violations permises, le module d'analyse proactive déclenche une alarme signalant qu'une panne va éventuellement affecter le système.

3.3.2.3 Synthèse

Le module d'analyse proactive intégré au sein de l'approche proposée permet de réaliser des prédictions des pannes dérivantes des fluctuations des valeurs de latence. Par ailleurs, nous avons fourni dans ce module deux méthodes statistiques qui permettent de calculer les valeurs estimées de la latence. En outre, les valeurs des seuils sont calculées et mises à jour à l'aide de la méthode de la Moyenne Mobile Exponentielle EWMA. Des chroniques temporelles sont utilisées dans le but d'effectuer des prévisions plus précises.

3.3.3 Analyse hybride

Le modèle analytique orienté QdS intègre aussi un module d'analyse hybride qui combine les deux formes d'analyse pré-décrites. Cette analyse pourrait être appliquée si le besoin à une analyse réactive se présente et que la contrainte de risque spécifiée par l'uti-

lisateur n'est pas vraiment garantie. Dans ce cas, l'introduction de l'analyse préventive pourrait stabiliser le système et aider à la satisfaction de l'utilisateur.

Le fonctionnement du système serait donc comme suit : le système déclenche d'abord une analyse réactive durant laquelle il s'auto-contrôle en vérifiant la contrainte spécifiée par l'utilisateur (Formule (3.20)). S'il note une violation de cette contrainte, le système migre vers une analyse proactive.

Une fois que le système se stabilise, il revient de nouveau vers l'analyse réactive et le cycle se répète. Le module d'analyse forme donc un cycle qui varie entre analyse réactive et proactive.

Ceci a pour but de satisfaire toujours les contraintes spécifiées par l'utilisateur et d'assurer un état acceptable du système.

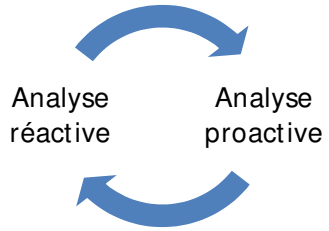

FIG. 3.7 – Analyse hybride

3.3.4 Approche de diagnostic

Outre les modules d'analyse, le modèle analytique proposé dans le cadre de cette thèse, introduit un module de diagnostic qui a pour but de chercher la cause de la dégradation de QdS, une fois prédite ou détectée [LAJ+12].

Selon notre étude, l'élévation de la valeur de la latence, entre deux *brokers* voisins B_1 et B_2 par exemple, est due soit à un déplacement de l'un des nœuds participant à la communication, soit au chargement du lien qui les relie.

Ceci nous permet donc de classer les pannes selon trois catégories comme le montre le tableau suivant.

TAB. 3.1 – Sources de pannes

Catégories de pannes	Source de pannes
mobilité	B_1 se déplace loin de B_2
	B_2 se déplace loin de B_1
	B_1 et B_2 se déplacent
charge	charge du système publier/souscrire
	charge d'une application extérieure
autres	obstacle, bruit

L'étude élaborée et présentée dans (Section 2.3.4) nous a permis de décider concernant la méthode utilisée pour identifier finement la cause des pannes. Ainsi, afin d'étudier la dépendance entre la variation de la latence d'une part et la variation des facteurs la causant, nous avons eu recours à l'auto-corrélation[cor].
Mathématiquement, la corrélation sert à étudier l'intensité de la liaison qui peut exister entre deux variables.
La corrélation s'articule autour de deux étapes principales :

3.3.4.1 Calcul du coefficient de *Pearson*

Le coefficient de corrélation de Pearson PCC mesure le degré de liaison et de dépendance entre deux variables quantitatives $X1$ et $X2$. Ce coefficient est décrit à l'aide de la formule suivante :

$$PCC = \frac{cov(X1, X2)}{\sqrt{var(X1) - var(X2)}}. \tag{3.40}$$

3.3.4.2 Test de significativité du coefficient de *Pearson*

La phase de test du coefficient de corrélation requière les étapes suivantes :

- Considérer l'hypothèse H_0 : il n'existe pas de relations entre les deux distributions $X1$ et $X2$.
- Préciser un risque d'erreur pour le rejet de H_0.
- Procéder au calcul de la valeur absolue du coefficient de corrélation.

- Trouver la valeur théorique, notée *Valth*, issue de la table des valeurs critiques du coefficient de corrélation de *Pearson*.
- Tester H_0 : l'hypothèse est vraie si $Valth >| PCC |$.
- Accepter ou rejeter H_0.

Selon le test de significativité du coefficient de *Pearson* prédécrit, deux situations peuvent en résulter :

- Si $Valth >| PCC |$, nous acceptons l'hypothèse H_0 et nous affirmons que les deux distributions $X1$ et $X2$ sont indépendantes.
- Sinon, si PCC est négatif, alors nous jugeons que $X1$ et $X2$ sont négativement corrélées. Par contre, si PCC est positif, nous déclarons que $X1$ et $X2$ sont positivement et significativement corrélées.

Notre module de diagnostic intégré au sein de notre approche, illustré dans la Figure 3.8, exécute en parallèle deux algorithmes. Le premier algorithme consiste à mesurer la corrélation entre la variation de la latence et celle du nombre de sauts.
Alors que le deuxième traite la corrélation entre la variation de la latence et celle de la charge. Selon la significativité des coefficients de corrélations calculés entre ces différentes distributions, nous nous trouvons face à trois scénarios différents.

- Si la latence et le nombre de sauts sont significativement et positivement corrélés, nous pouvons donc déduire que la variation du nombre de sauts est un parmi les facteurs qui ont engendré la dégradation. D'ou, nous pouvons déduire que la catégorie de panne est la mobilité. Reste à déterminer la cause exacte de la dégradation en vérifiant les positions des nœuds et leur vitesses.

  Nous passons donc à déterminer lequel des deux *brokers* est entrain de se déplacer en examinant la variation de leurs positions en fonction du temps. Pour ce faire, nous étudions la corrélation qui existe entre la variation de la distance de déplacement du *broker* et la variation de sa vitesse de déplacement.
- Si la latence est la charge sont significativement et positivement corrélées, cela veut dire que la charge est un parmi les facteurs qui ont causé la dégradation.
- Si la latence entre les paramètres n'est pas significative, cela implique que la

dégradation est causée par une source extérieure qui pourrait inclure un bruit, un obstacle...

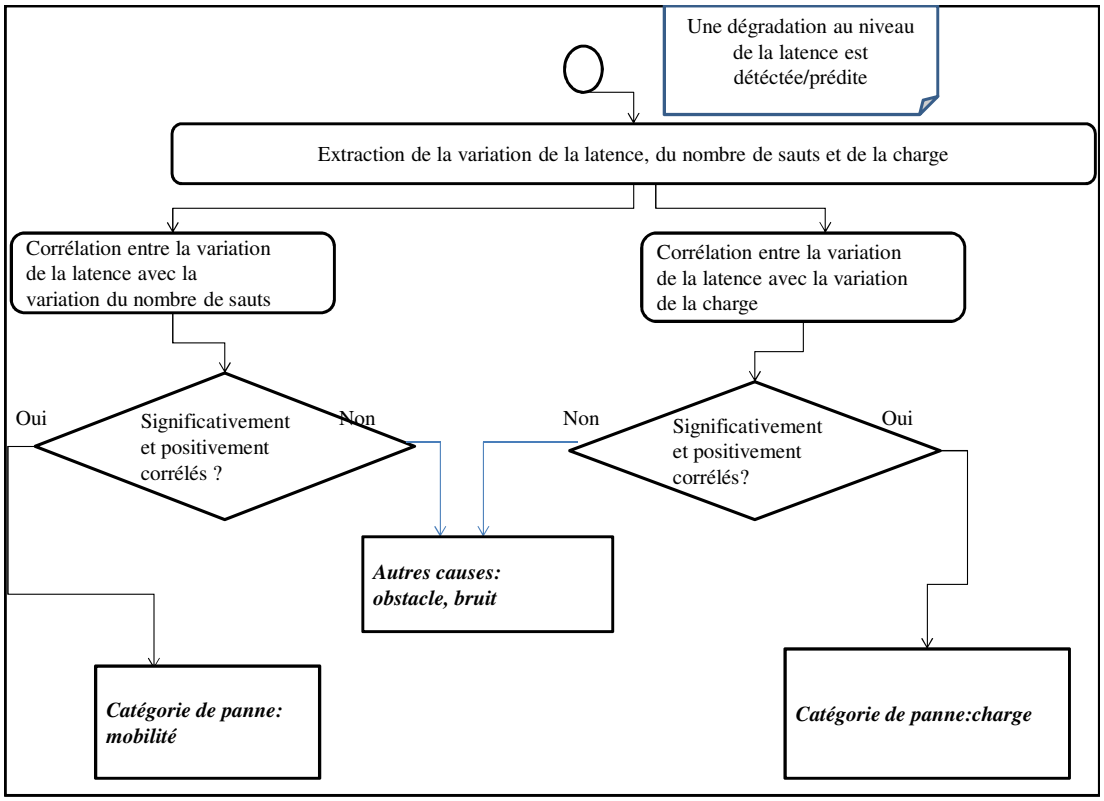

FIG. 3.8 – Approche de diagnostic

3.3.4.3 Synthèse

Le module de diagnostic proposé a pour but de localiser les pannes et d'identifier leurs sources. Ceci permet de planifier les actions correctives adéquates durant les phases de planification et d'exécution du processus d'auto-adaptation.

La corrélation, utilisée au sein du diagnostic, nous a permis de mesurer l'intensité de la liaison entre la variation de la latence et les sources de pannes correspondantes.

3.4 Conclusion

Dans ce chapitre, nous avons proposé un modèle analytique orienté QdS. Ce modèle est incorporé au sein de chaque nœud du service d'évènements. En se basant sur des méthodes statistiques, notre approche assure une supervision continue du système et une détection/prédiction des pannes affectant les liens logiques au niveau middleware. L'approche ainsi proposée permet aussi d'identifier la cause de la dégradation et de reconnaitre la source exacte de la panne.

Dans le chapitre suivant, nous présentons les scénarios d'expérimentations réalisés afin de valider notre approche.

CHAPITRE 4

Évaluation des performances

4.1 Introduction

La démarche scientifique adoptée dans le cadre de cette thèse, part de la proposition d'un modèle analytique qui sert à contrôler le système et à détecter voire prédire les dégradations de QdS pouvant l'affecter. Le modèle ainsi proposé n'impose pas de restrictions au développeur, par contre, il lui offre la possibilité de choisir le type d'analyse adéquat à son application.
En effet, d'après notre étude, les applications les plus sensibles aux pannes, nécessitent un module d'analyse proactive, afin de prédire l'occurrence des pannes et préserver le bon fonctionnement du système.
D'autres applications devraient aussi faire recours au module d'analyse réactive et hybride afin d'assurer un transfert non dégradé des flux multimédias par exemple.
Les expérimentations élaborées dans ce chapitre visent à valider la démarche théorique proposée.

Nous présentons dans un premier temps, le contexte d'expérimentation adopté, puis nous passons à décrire les cas d'études considérés pour la validation du module proposé.
Ainsi, nous visualisons les résultats des modules de *monitoring* et d'analyse reflétant l'état du système. Ensuite, nous comparons les performances du système sans et avec l'introduction du module proposé.
Une étude de la complexité spatiale et temporelle du module analytique proposée a été aussi menée au sein de cette partie.

4.2 Contexte d'expérimentation

La partie expérimentale du module proposé se base sur le simulateur *Jist/Swans* qui est un simulateur de réseaux mobiles ad-hoc. Sur ce simulateur, nous avons mis en œuvre le système *Siena*, comme étant un système basé évènements opérant au niveau middleware.

4.2.1 Le Simulateur *Jist/Swans*

Jist est un simulateur à évènements discrets auquel est associé le module *Swans* afin de simuler des réseaux mobiles ad-hoc supportant un grand nombre de nœuds.
Le simulateur *Jist/Swans* a fait ses preuves en comparaison avec d'autres simulateurs tels que NS, GlomoSim [1]. En effet, *Jist/Swans* est reconnu par la scalabilité qu'il offre. En plus, il n'exige pas un langage de programmation bien particulier tel que GlomoSim. Ce dernier impose la réécriture du code ce qui peut gêner l'utilisateur du simulateur. Par ailleurs, *Jist/Swans* exploite le langage Java.
De plus, il a montré son efficacité du point de vue temps d'exécution en comparaison avec les autres simulateurs réseaux tel que NS.

Jist/Swans [BHvR] est formé d'une pile qui met en œuvre les différents constituants du simulateur (Figure 4.1).

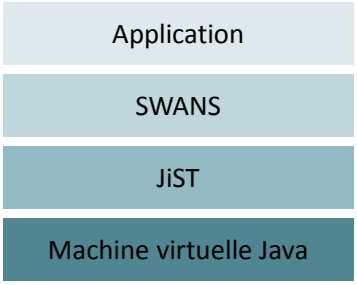

FIG. 4.1 – Pile de simulation

◇ *'JVM'* : La machine virtuelle Java est localisée en bas de la pile. C'est le lieu ou se déroulent toutes les simulations.

◇ JiST [BHvR05] : localisé juste au dessus de la machine virtuelle Java. Il présente le moteur de simulation des évènements discrets.

◇ SWANS [BHR05] : construit juste au dessus de la plate-forme JiST, *Swans* présente un simulateur de réseaux sans fil. Grace à *Swans*, il est possible de simuler des réseaux mobiles ad-hoc supportant un nombre important de nœuds. Ceci met en relief son aspect

[1] Global Mobile Simulator

de scalabilité.

◇ Application : juste en haut de la pile, se trouve l'application de l'utilisateur, qui pourrait fonctionner sur chacun des nœuds simulés.

Du point de vue code, l'ensemble des classes de *Swans* est organisé comme l'illustre la Figure suivante :

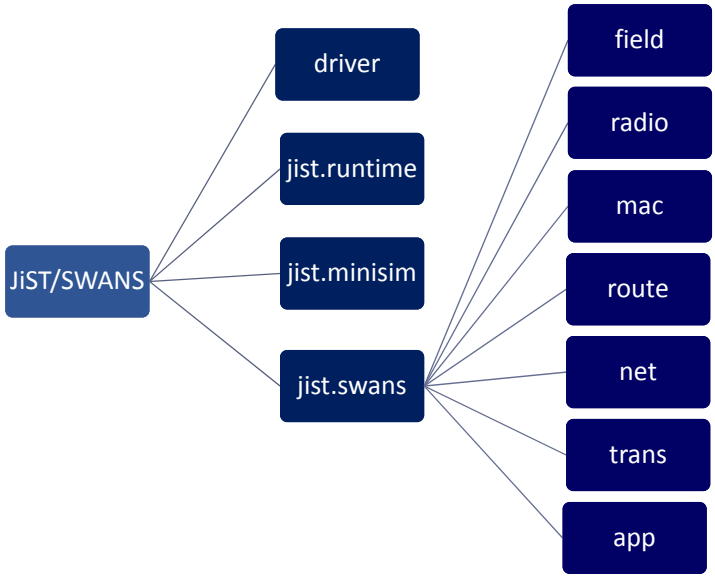

FIG. 4.2 – Arborescence du simulateur *Jist/Swans*

Les classes du simulateur *Swans* sont organisées sous forme de packages et sont bien ordonnées afin de former les différentes couches du modèle OSI.

- La couche physique : représentée par le *package jist.swans.field* et *jist.swans.radio*. Elle représente la propagation du signal qui pourrait venir de divers équipements radio.
- La couche liaison de données : modélisée par le *package jist.swans.mac*. Elle met en œuvre les protocoles d'accès au médium.
- La couche routage : définie par la *package jist.swans.route*. Elle englobe les protocoles de routage adoptés par le simulateur tels que *AODV*, *ZRP*, *DSR*.
- La couche transport : présentée par la *package jist.swans.trans* et met en œuvre les protocoles de transport (*TCP* et *UDP*).

- La couche application : présentée par le *package jist.swans.app* et définit l'ensemble des applications à simuler.

Pour simuler un ensemble de nœuds sur un réseau ad-hoc, le simulateur procède à la création de ces nœuds et à la mise en place des différentes couches précitées dans chacun des nœuds simulés (Figure 4.3).

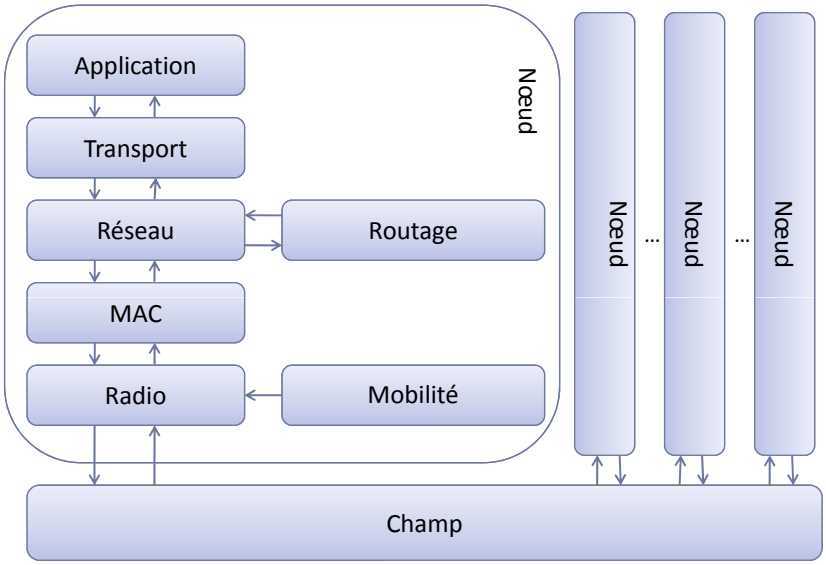

FIG. 4.3 – Architecture interne d'un nœud simulé

4.2.2 Le Système publier/souscrire *Siena*

Siena est un système publier/souscrire basé contenu. C'est un projet de recherche *Open Source*. Il présente l'avantage d'être scalable. La version disponible de *Siena*, écrite en Java, supporte la topologie hiérarchique.

Nous avons d'abord étendu cette version pour qu'elle supporte la topologie pair à pair acyclique. Ensuite, nous avons étendu *Siena* moyennant le module proposé. Finalement, nous avons intégré et déployé *Siena* sur le simulateur *Jist/Swans*.

4.3 Application de gestion de crise

Cette application nous a permis de valider l'analyse proactive proposée. Dans ce qui suit, nous présentons un aperçu de l'application, ainsi que les résultats d'expérimentation obtenus.

4.3.1 Description de l'application

Les applications de gestion de crise (CMS) peuvent être utilisées dans des situations d'urgence et de catastrophes naturelles.

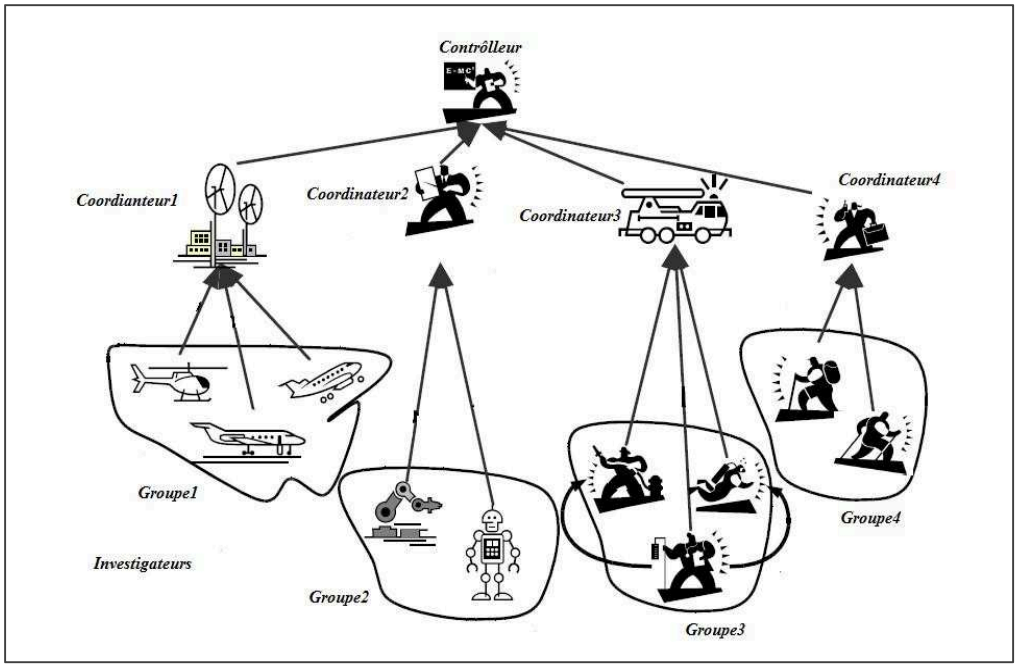

FIG. 4.4 – Application de gestion de crise

Comme le montre la Figure 4.4, l'application (CMS) inclut des groupes de robots et de personnels militaires qui coopèrent ensemble afin de réaliser une mission commune.

Parmi ces participants, nous distinguons des entités de type contrôleur, coordinateur et investigateur.

- Le contrôleur supervise l'application. Il collecte les rapports de la part des coordinateurs.
- Le coordinateur gère un groupe d'investigateurs en distribuant des tâches à exécuter. Il synthétise les informations de la part des investigateurs. De ce fait, le coordinateur est appelé à envoyer les informations au contrôleur.
- Les investigateurs observent, analysent et soumettent des rapports décrivant la situation au coordinateur assigné.

Dans la suite, nous nous intéressons à un groupe ou une section d'investigateurs mobiles formant ainsi le réseau mobile ad-hoc.

Dans le but d'assurer la communication entre les investigateurs mobiles, nous avons mis en place un système publier/souscrire. Nous nous intéressons, particulièrement, à la communication entre deux investigateurs incorporant des *brokers* notés B_1 et B_2.

Le degré de risque ou d'urgence de cette application nécessite le recours à une analyse proactive.

Nous adoptons ainsi une stratégie de prévention de pannes faisant référence aux méthodes statistiques proposées.

4.3.2 Paramètres de simulation

Le tableau 4.1 récapitule les données adoptées dans nos simulations.

TAB. 4.1 – Paramètres de simulation

Paramètres de simulation	**Valeur**
champ de simulation	2000*3000m
nombre de nœuds	200
modèle de mobilité	*Continuous Random Walk*
temps de pause	30s
vitesse maximale	30m/s
portée de transmission	300m
couche liaison de données	802.11 b
couche transport	UDP
taille du message	120 octets
durée de simulation	4000s

Dans les scénarios de test réalisés, nous considérons que tous les nœuds se déplacent selon le modèle de mobilité *Continuous Random Walk* avec une vitesse maximale égale à 30m/s. Dans ce modèle de mobilité, chaque nœud choisit aléatoirement un angle de direction et une vitesse aléatoire entre $[V_{min}, V_{max}]$. Lorsque ce nœud atteint le nœud destination, il reste en repos pour un certain temps (temps de pause=$30s$) avant de commencer le mouvement suivant.
Dans les expérimentations faites, nous nous intéressons à la liaison entre les *brokers* voisins B_1 et B_2. Dans ce qui suit, nous présentons deux scénarios de test dans lesquels, nous affectons la qualité des liens en déplaçant l'un des deux *brokers*.
Les résultats présentés correspondent à une trace de vitesse moyenne égale à $5m/s$.

Au delà du réseau physique, un réseau logique est bâti. Ce réseau est composé de 30 *brokers* placés d'une façon aléatoire sur les nœuds physiques et connectés via une topologie pair à pair acyclique. Le réseau logique contient aussi 10 producteurs et 10 consommateurs. Un producteur P publie périodiquement des messages (chaque $30s$) vers le point d'accès auquel il est connecté, dans notre cas B_1. Ce dernier transmet l'évènement vers son voisin B_2. Ce message est transmis entre les *brokers* du service d'évènements jusqu'à aboutir au consommateur adéquat.

Au départ, les deux *brokers* B_1 et B_2 étaient séparés par un seul saut sur le réseau physique. Cependant, après déplacement des nœuds, ces deux *brokers* deviennent éloignées l'un par rapport à l'autre, ce qui augmente le nombre de sauts à l'intermédiaire.

4.3.3 Détections de pannes dans la phase à régime transitoire

Parallèlement au stockage des valeurs de la latence, nous procédons à un contrôle de la QdS en comparant la valeur enregistrée de la latence avec celle du seuil. Le seuil est calculé après avoir prouver l'adéquation de la série formée par les valeurs moyennes à la loi Gaussienne.
La figure 4.5 illustre les résultats obtenus pendant la phase à régime transitoire.

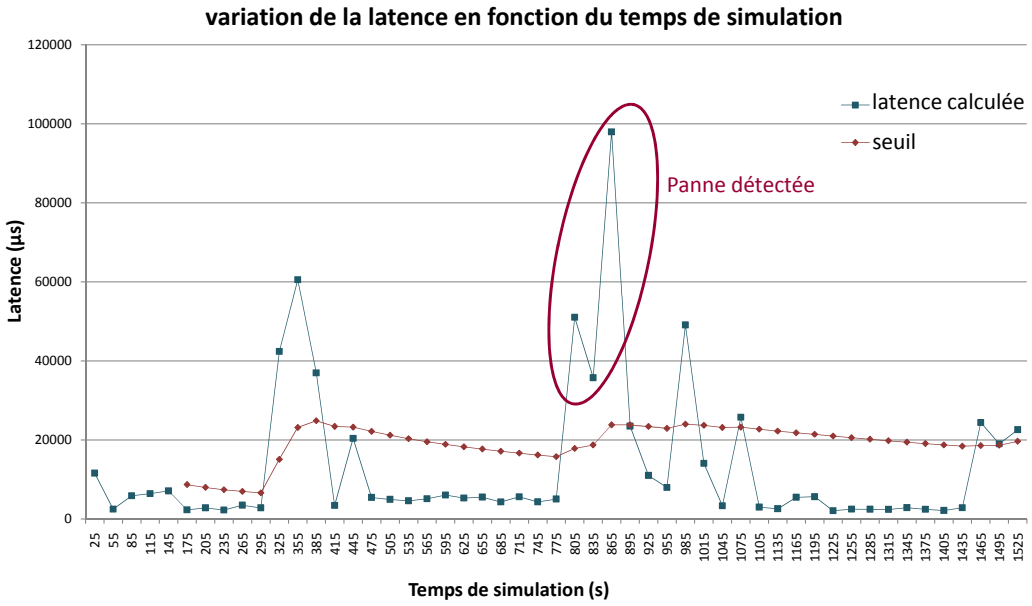

FIG. 4.5 – Phase à régime transitoire

A l'instant $t = 780s$, nous déplaçons le *broker* B_2 progressivement loin de ses voisins nécessitant une nouvelle recherche de routes. Les valeurs de la latence croissent de $51046\,\mu\,s$ à l'instant $t = 805s$ à $97952\,\mu\,s$ à l'instant $t = 865s$. Le module d'analyse a détecté l'occurrence de la panne à l'instant approprié.

4.3.4 Prédictions de pannes dans la phase à régime permanent

Dans cette partie, nous exposons les résultats relatifs aux estimations basées AR et celles basées ARIMA.

4.3.4.1 Estimation basée AR

Partant du scénario décrit, nous considérons une fenêtre d'observations égale à 50 évènements. La fenêtre considérée est glissante au cours du temps, donc à chaque nouvelle valeur mesurée de latence, nous considérons les derniers 50 messages pour réaliser les estimations.

Afin de tester l'efficacité du module d'analyse proactive, nous avons injecté des pannes dans

le système dans le but d'affecter la qualité des liens. Notre but durant cette expérimentation (Figure 4.6) est de vérifier si le système a bien réussi à prévoir ces pannes pendant la phase à régime permanent. Pour ce faire, nous avons éloigné progressivement le *broker* B_2 de ses voisins et nous avons suivi la qualité du lien $B_1 - B_2$.

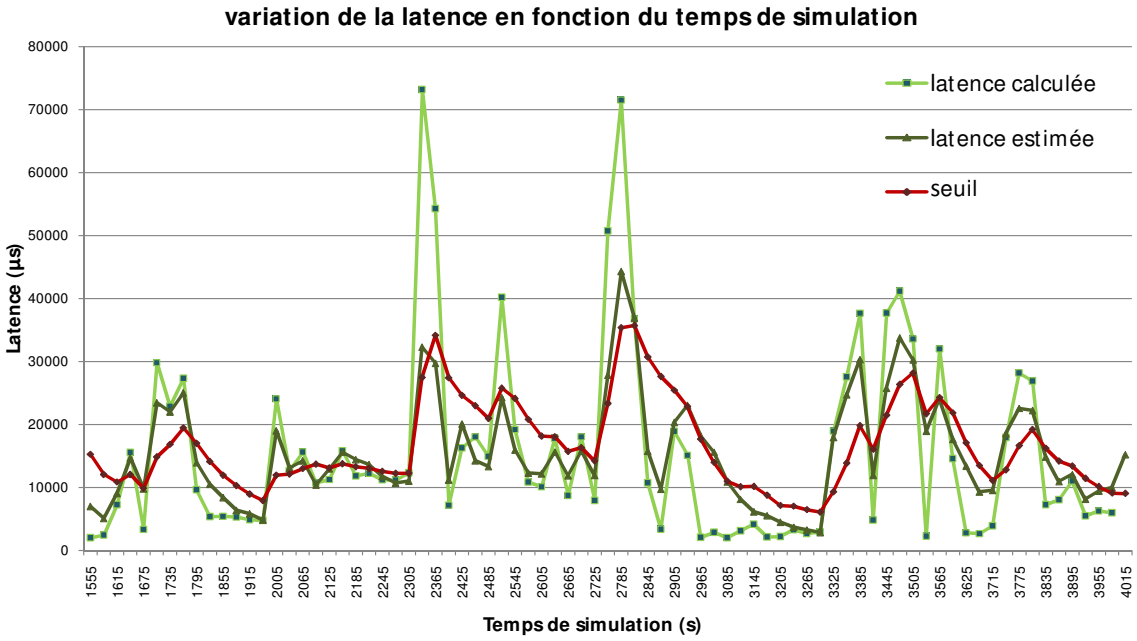

FIG. 4.6 – Résultats de simulations relatifs à l'analyse proactive

En consultant le fichier *log* du *broker* B_2, nous remarquons que le nombre de sauts croit progressivement de 2 sauts à 3 sauts pour aboutir à 4 sauts au troisième déplacement.

En accord avec ces déplacements, les valeurs de la latence augmentent progressivement. De ce fait, la valeur de la latence s'élève de 18936μs à t=3325s jusqu'à 27549μs à l'instant t=3355s. Ceci affirme la présence d'une panne.

Le fichier log du *broker* B_2 fait preuve que le module d'analyse proactive proposé s'est aperçu de la panne et a bien réussi à la prédire à l'avance (Figure 4.7) en produisant une alarme.

```
Etat du lien logique avec le dispatcher 1 : dégradé
Instant de la prédiction de la dégradation : 3295s
Le nombre de violations successifs : 1
La valeur estimée de la latence: 17930.997696634797
La valeur estimée du seuil: 9300.463322603646
Le dépassement mesure 8630.534374031151
-----------------------------------------------------
Etat du lien logique avec le dispatcher 1 : dégradé
Instant de la prédiction de la dégradation : 3325s
Le nombre de violations successifs : 2
La valeur estimée de la latence: 24720.380317757892
La valeur estimée du seuil: 13862.597491952734
Le dépassement mesure 10857.782825805158
-----------------------------------------------------
Etat du lien logique avec le dispatcher 1 : En panne
Instant de la prédiction de la panne : 3355s
La valeur estimée de la latence: 30313.78321385468
La valeur estimée du seuil: 19813.198118964552
Le dépassement mesure 10500.585094890128
```

FIG. 4.7 – Extrait du fichier log montrant la prédiction de la panne

4.3.4.2 Estimation basée ARIMA

Dans ce scénario, nous déplaçons le *broker* B_2 avec une grande vitesse à partir de l'instant $t = 1760s$. Ceci cause une augmentation des valeurs de latence de 27961 μs à 31181 μs à l'instant $t = 1790s$ et à 61723 μs à l'instant $t = 1820s$ (Figure 4.8).

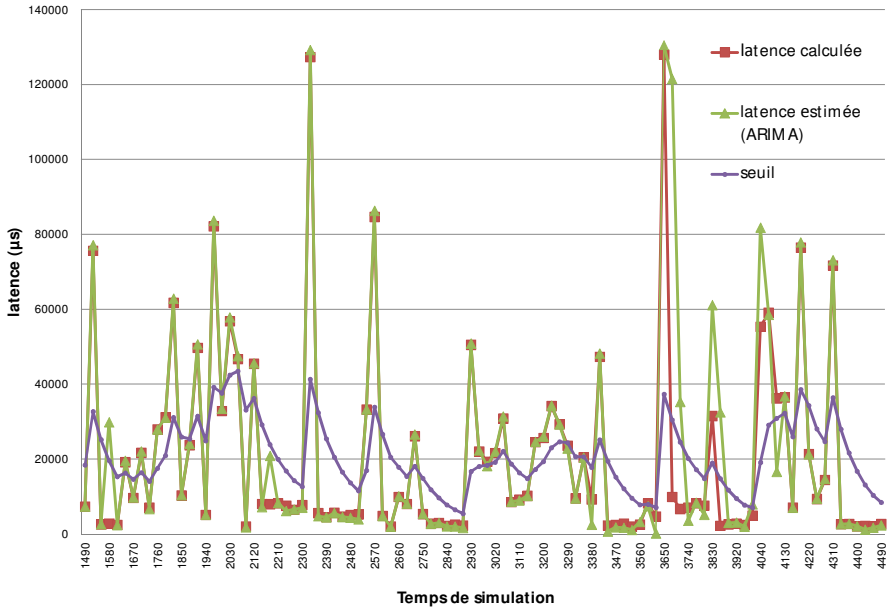

FIG. 4.8 – Prédiction basée ARIMA

Le module d'analyse proactive a pu s'apercevoir qu'une panne va probablement affecter

le système à un instant futur proche.

Après le déclenchement d'alarmes, notre algorithme cherche la source de la dégradation. Le fichier Log (Tableau 4.2) généré par le *broker* B_2 détaille ceci. En effet, l'algorithme de diagnostic vérifie les relations de corrélation qui existent entre la latence et le nombre de sauts d'une part et la latence et la charge d'autre part. Le calcul effectué par cet algorithme montre que la valeur du coefficient de corrélation mesuré entre la latence et le nombre de sauts est supérieure à la valeur théorique. De ce fait, Le module d'analyse proactive déduit que la source de panne provient d'un mouvement du *broker* B_2.

TAB. 4.2 – Fichier Log représentant la recherche de la source de pannes

Instant de prédiction	1820 s
Nombre de violations successives	3
Latence à l'instant t-2	27961.0 μ s
Latence à l'instant t-1	31181.0 μ s
Latence à l'instant t	61723.0 μ s
Nombre de sauts à l'instant t	3.0
Nombre de sauts à l'instant t-1	3.0
Nombre de sauts à l'instant t-2	4.0
PCC	0.996
Valeur Théorique de *PCC*	0.9877
Catégorie de pannes	mobilité
Cause	variation du nombre de sauts
	B_2 est en mouvement
	B_2 se déplace avec une grande vitesse

4.3.5 Évaluation des prédicteurs de pannes basés AR et ARIMA

Dans le but d'évaluer l'efficacité des prédicteurs de pannes proposés au sein du modèle analytique proposé, nous procédons à une évaluation graphique ainsi qu'à une évaluation moyennant les paramètres de performance.

4.3.5.1 Évaluation graphique

En adoptant toujours les mêmes paramètres de l'expérience précédente, nous effectuons une comparaison entre les prédicteurs de pannes basés AR et ARIMA. La Figure 4.9 trace

les valeurs de la latence mesurées, avec celles estimées à l'aide de AR et ARIMA.

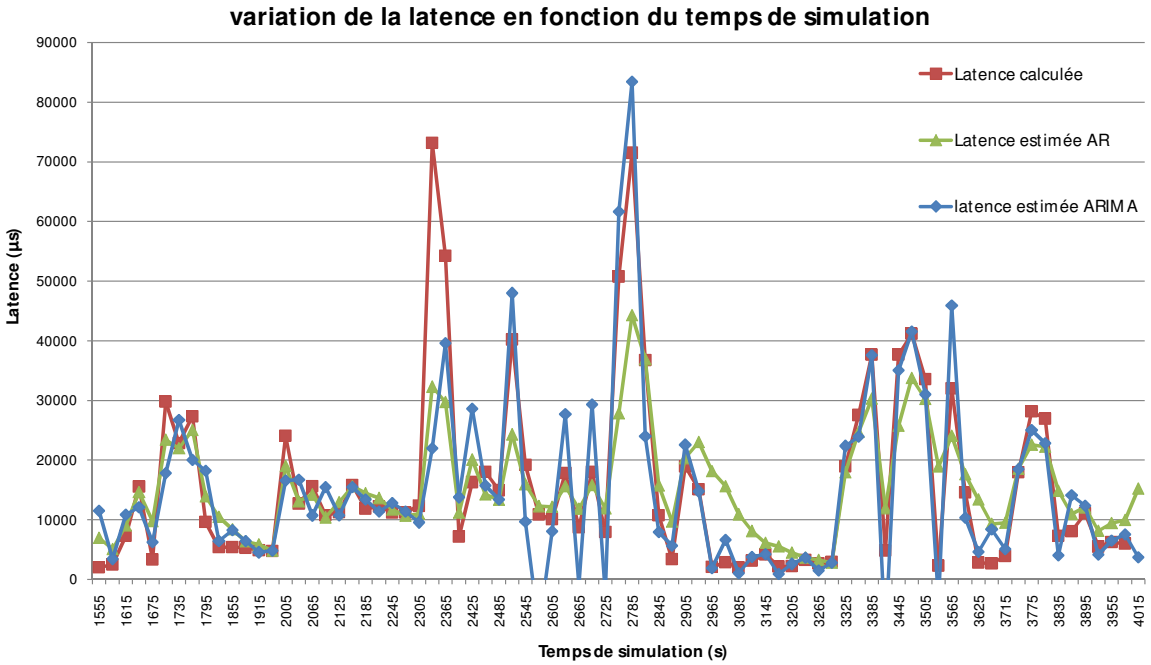

FIG. 4.9 – Comparaison entre le prédicteur basé AR et celui basé ARIMA

On déduit à partir de la Figure 4.9 que les deux courbes suivent la courbe réelle des valeurs mesurées. Ceci montre la cohérence entre les mesures réelles et les mesures estimées.
En outre, la tendance de la courbe basée AR présente une légère déviation par rapport à celle basée ARIMA en comparaison avec la courbe des valeurs réelles.
Nous déduisons donc que le prédicteur de pannes basé ARIMA présente des performances légèrement meilleures que celui basé AR dans notre contexte.
La partie suivante présente une preuve plus solide de cette constatation.

4.3.5.2 Évaluation selon les critères de performances

Afin d'évaluer l'efficacité du prédicteur de pannes proposé au sein du modèle analytique proposé, nous faisons référence aux paramètres de performances de la table de contingence (Table 4.3). En fait, l'objectif de l'algorithme proposé est de réaliser une prédiction de pannes d'une manière précise qui s'aperçoit au maximum des pannes et qui génère au minimum des fausses détections.

Le tableau 4.3 définit les quatre cas possibles.

- Une prédiction est dite *true positive* si une panne se produit et que le prédicteur arrive à la prévoir.
- Une prédiction est dite *false positive* s'il y a pas de pannes et qu' une alarme est générée.
- Une prédiction est dite *false negative* si le prédicteur ne réussit pas à prédire une panne existante.
- Une prédiction est dite *true negative* s'il y a pas de pannes et que le prédicteur ne génère pas d'alarmes.

TAB. 4.3 – Table de contingence [SLM10]

	Panne existante	Panne inexistante	Somme
Prédiction :panne	*True positive(TP)*	*False positive(FP)*	Positives(POS)
Prédiction :pas de panne	*False negative(FN)*	*True negative(TN)*	Negatives(NEG)
Somme	Pannes(P)	Pas de pannes(PP)	Total(T)

À la lumière de ces définitions, les paramètres de performance du système pourraient être définis comme suit :

$$precision = \frac{TP}{TP + FP} = \frac{TP}{POS} \tag{4.1}$$

Le deuxième paramètre d'évaluation du prédicteur est le rappel connue sous son appellation anglaise *recall*. C'est le rapport entre le nombre d'alarmes produites et le nombre total de pannes.

$$rappel = \frac{TP}{TP + FN} = \frac{TP}{P} \tag{4.2}$$

Le tableau 4.4 présente les performances de notre prédicteur basé AR et ARIMA tirées des expérimentations faites en termes de précision et de rappel. En considérant ces deux métriques de performance et en appliquant les formules sur l'ensemble des pannes survenant dans le système, nous trouvons une valeur de précision allant de 75% en exploitant la méthode AR vers 80% en exploitant la méthode ARIMA.

Nous pouvons conclure d'une part, que le module d'analyse proactive produit en grande partie de correctes alarmes. D'autre part, le module d'analyse basé ARIMA offre une

précision légèrement plus importante par rapport à celui basé AR. Ceci est conforme aux résultats graphiques expliqués ci dessus.

Le tableau 4.4 montre aussi les performances du prédicteur en termes de rappel. En effet, le module d'analyse proposé réussit en grande partie (88%) à s'apercevoir aux pannes existantes dans le système. Ceci fait preuve des performances du prédicteur et de l'analyse proactive d'une façon générale.

TAB. 4.4 – Comparaison des paramètres de performance des prédicteurs

	Précision	Rappel
Prédiction basée ($ARIMA$)	80%	88%
Prédiction basée (AR)	75%	85.71%

4.4 Application multimédia

Dans cette partie, nous détaillons l'application multimédia ainsi que les résultats illustrant l'efficacité des modules d'analyse réactive et hybride incorporés au sein de cette application.

4.4.1 Description de l'application

Les applications multimédia imposent des exigences en termes de QdS. En effet, les applications de vidéo-streaming imposent que les flux soient envoyés et reçus à temps sans aucun retard de livraison. Dans le but d'assurer une application multimédia interactive, on considère un système publier/souscrire la où les participants à la communication échangent des données de type multimédia. Les restrictions imposées par le vidéo-streaming exigent que le message expire après un certain délai. Afin de respecter ces exigences, nous intégrons notre modèle analytique afin que les flux arrivent aux destinations à temps sans dégradation.

4.4.2 Paramètres de simulation

Le tableau 4.5 suivant regroupe les paramètres de simulation. Le réseau logique bâti au dessus du réseau physique est analogue à celui introduit précédemment (dans la Section 4.3).

TAB. 4.5 – Paramètres de simulation

Paramètres de simulation	**Valeurs**
Taille du champs	2000m×3000m
Nombre de nœuds	200
Modèle de mobilité	*Continuous Random Walk*
Temps de pause	30s
Vitesse maximale	30m/s
Portée de transmission	300m
Liaison de données	802.11
Protocole de transport	UDP
Durée de simulation	9000s

4.4.3 Expérimentations relatives à l'analyse réactive

Nous présentons dans une première partie le calcul du nombre de dépassements tolérées pour déclencher une alarme. En deuxième partie, nous décrivons les résultats obtenus.

4.4.3.1 Calcul du nombre de dépassements tolérés par l'analyse réactive

Considérons une valeur du risque, introduite par l'utilisateur, égale à 0,01 au sein de l'application multimédia par exemple. À l'issu de cette valeur et des valeurs tirées des expérimentations, nous appliquons les formules expliquées dans le chapitre précédant (Section 3.3.1). Nous aurons ainsi :

$P[LOK] = 0.74$, $P[LOK \rightarrow LV] = 0.127$ et $P[LV] = 0.236$, $P[LV \rightarrow LV] = 0.104$ donc :

$$P[Toutetat \rightarrow LV] = 0.74 \times 0.127 + 0.236 \times 0.104 \tag{4.3}$$

$\Longleftrightarrow M \succeq 2.2$. Ce qui signifie que le nombre de dépassements tolérés M doit être au moins

égale à 3.

4.4.3.2 Résultats issus du module d'analyse réactive

Dans le scénario de test réalisé, nous considérons que tous les nœuds se déplacent selon le modèle de mobilité *Continuous Random Walk* avec une vitesse maximale égale à 30m/s.

1. Scénario 1 : la mobilité source de pannes

Ce scénario de test consiste à provoquer des pannes au niveau des liens logiques dans le but de vérifier que notre module d'analyse parvient à les détecter et que l'algorithme de diagnostic pourrait bien identifier les causes des dégradations injectées.

Au niveau de la phase à régime transitoire (Figure 4.10), nous injectons des pannes au niveau du lien B_1- B_2 à l'instant $t = 260s$.
La latence augmente progressivement de 2628 μs jusqu'à 41422 μs. Face à cette dégradation, le module d'analyse génère une alarme notifiant la présence d'une dégradation à l'instant approprié.

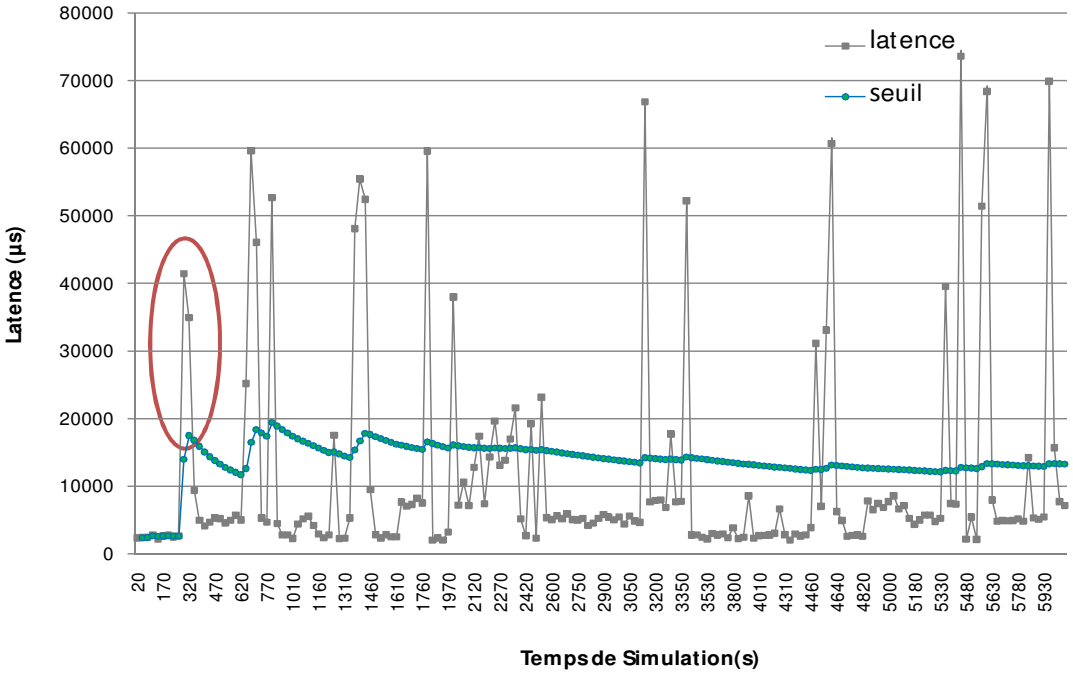

FIG. 4.10 – Résultats relatifs à l'analyse réactive (phase à régime transitoire)

Au niveau de la phase à régime permanent, plus précisément, à l'instant $t = 6350$, nous déplaçons le *broker* B_2 loin de son voisin B_1 causant ainsi une augmentation des valeurs de latence de 95519 μ s à 75610 μ s.

Une autre panne est injectée à $t = 7820s$ du temps de simulation. Ceci cause aussi une variation de la latence de 29029 μs à 81567 μs. La Latence augmente donc et dépasse le seuil adaptatif. Comme illustré dans la Figure 4.11, le module d'analyse réactive réussit à détecter ces dégradations et génère des alarmes aux moments appropriés.

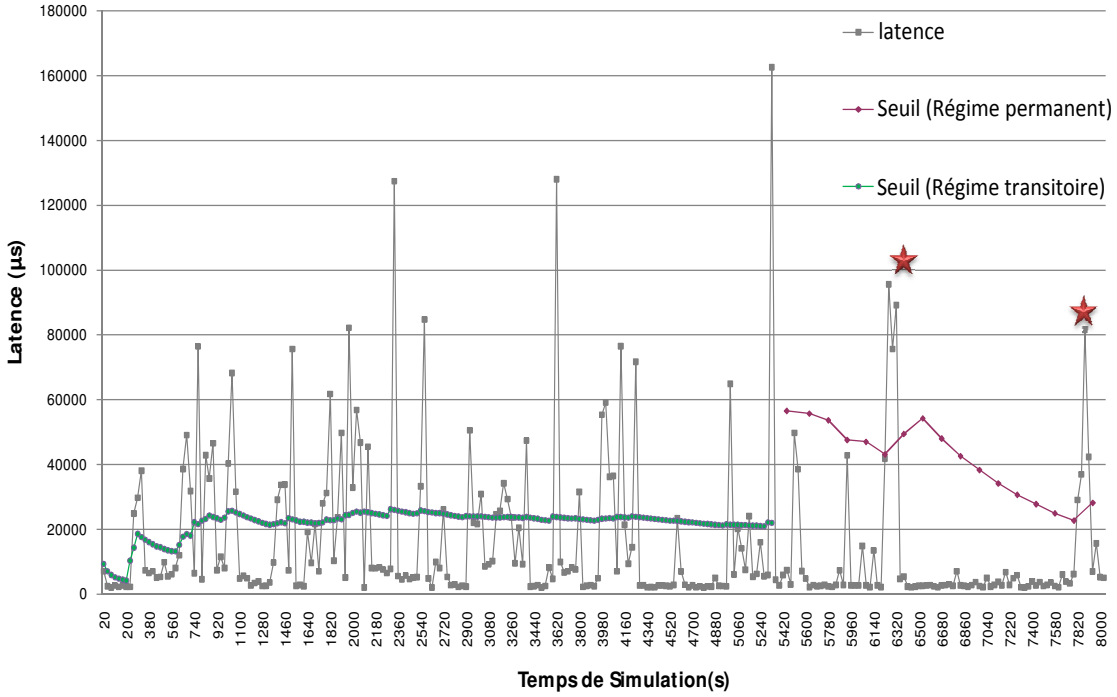

FIG. 4.11 – Variation de la latence en fonction du temps de simulation (scénario1)

Notre module de diagnostic a pu identifier également la cause de la dégradation détectée à l'instant $t = 7820$ qui est la mobilité. Nous présentons le fichier log du *broker* B_2 correspondant à ce scénario dans la Table 4.6.

TAB. 4.6 – Fichier log relatif au premier scénario de test

Instant de détection	$7880s$
Nombre de dépassements successives	3
Latence à l'instant t-2 Latence à l'instant t-1 Latence à l'instant t	$29029\,\mu\,s$ $36130\,\mu\,s$ $81567\,\mu\,s$
Nombre de sauts à l'instant t-2 Nombre de sauts à l'instant t-1 Nombre de sauts à l'instant t	5 5 6
PCC calculé	0.99027
Valeur théorique	0.9877
Catégorie	Mobilité
Cause	Le *broker* B_2 s'éloigne de son voisin avec une grande vitesse

Le module de diagnostic a calculé la corrélation entre la variation de la latence et la variation du nombre de sauts. Une comparaison entre la valeur calculé du PCC et celle théorique a montré que la mobilité est une parmi les causes qui ont engendré la dégradation des valeurs de latence.

2. Scénario 2 : la charge source de pannes

Ce scénario de test consiste à provoquer des pannes dans le système en surchargeant le lien B_1-B_2. Réellement, cette surcharge engendre une augmentation des valeurs de la latence variant entre $8235\,\mu\,s$ et $9503\,\mu\,s$.

La Figure 4.12 montre que ces valeurs de latence présentent trois dépassements consécutifs du seuil.

Le module de diagnostic parvient à identifier la cause de la dégradation détectée à l'instant approprié ($t = 256$). En effet, le module de diagnostic procède à un calcul du coefficient de corrélation mesurant la dépendance entre la variation de la latence et la charge d'une part, et la variation de la latence et le nombre de sauts d'autre part.

D'après le calcul du coefficient de corrélation mesuré entre la latence et la charge, l'algorithme de diagnostic déduit que la charge est une parmi les causes qui ont mené vers la dégradation de la latence.

Nous présentons ainsi le fichier log correspondant à ce scénario dans la Table 4.7.

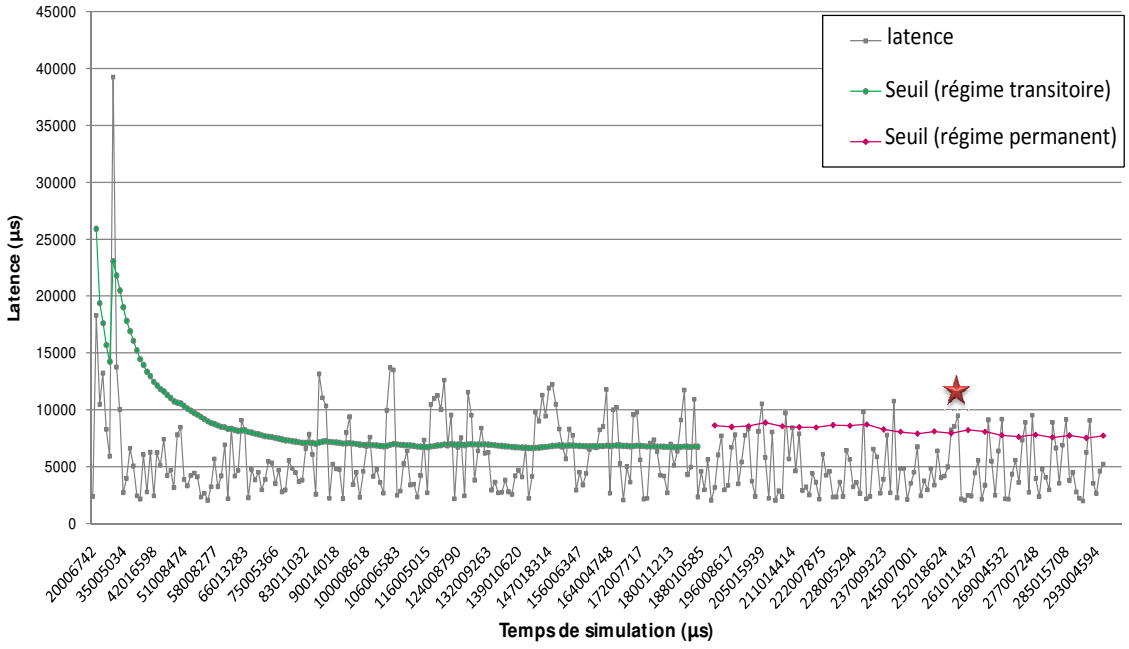

FIG. 4.12 – Variation de la latence en fonction du temps de simulation (scénario2)

TAB. 4.7 – Fichier log de diagnostic du deuxième scénario de test

Instant de détection	$256s$
Nombre de dépassements successives	3
Latence à l'instant t-2 Latence à l'instant t-1 Latence à l'instant t	$8235\,\mu\,s$ $8460\,\mu\,s$ $9803\,\mu\,s$
Charge à l'instant t-2 Charge à l'instant t-1 Charge à l'instant t	$164msg/s$ $191msg/s$ $233msg/s$
PCC calculé	0.9879
Valeur théorique	0.9877
Catégorie	Charge
Cause	Charge du lien logique entre les deux *brokers* communicants

4.4.4 Expérimentations relatives à l'analyse hybride

Partant du même scénario de mobilité (scénario 1), nous affectons encore le système prédécrit. Face à ces perturbations, notre module d'analyse s'aperçoit que la probabilité de pannes dans le système est devenue supérieure au risque spécifié par l'utilisateur (à

l'instant t=8840s du temps de simulation).

Le système est devenu donc non stable et ne répond plus aux exigences spécifiées par l'utilisateur, ce qui présente un risque élevé de mal-fonctionnement.

Cependant, garder le même état pourrait mener le système vers un état de dysfonctionnement. C'est ainsi que se présente le besoin de faire une opération de migration vers l'analyse proactive afin d'anticiper les pannes voire même satisfaire les contraintes exigées par l'utilisateur de l'application.

En se basant sur ce principe, le système déclenche automatiquement une opération de migration vers l'analyse réactive (basée ARIMA) afin de répondre aux exigences de l'utilisateur.

La Figure 4.13 décrit cette opération de migration vers l'analyse proactive.

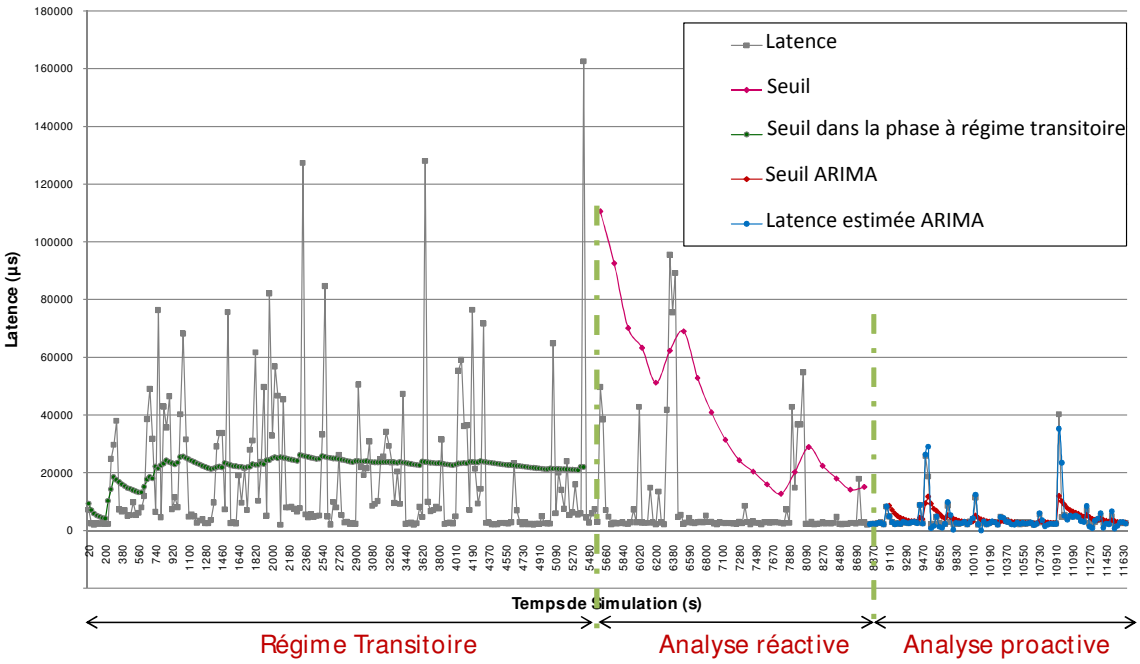

FIG. 4.13 – Analyse hybride

4.4.5 Etude des paramètres de performance du système

Afin d'évaluer la performance du modèle proposé, nous avons considéré la disponibilité du système comme un paramètre de performance.

Par définition, la disponibilité est la probabilité que le système fonctionne normalement durant une période de temps.

Mathématiquement, la disponibilité est estimée en utilisant l'équation 4.4 dans le cas d'un système non réparable.

$$availability = N\_received/N\_sent \tag{4.4}$$

avec *N_received* est le nombre de messages reçus correctement par un *broker* et *N_sent* est le nombre de messages envoyés par le *broker* voisin.

Dans le cas d'un système réparable, la disponibilité est exprimée comme suit :

$$availability = MTBF/(MTTR + MTBF) \tag{4.5}$$

Avec :

– MTBF est le temps moyen entre pannes.

– MTTR est le temps moyen jusqu'à la réparation.

Afin de montrer l'efficacité de l'analyse hybride, nous comparons la disponibilité dans cinq scénarios différents :

– 1. Premier scénario

Dans ce scénario, nous considérons que le système est non réparable, càd, que nous n'avons pas introduit le module analytique proposé au sein du système *Siena*. De ce fait, la mobilité des nœuds engendre une perte de connectivité et l'interruption du fonctionnement du système.

La Figure 4.14 décrit le comportement du système non réparable.

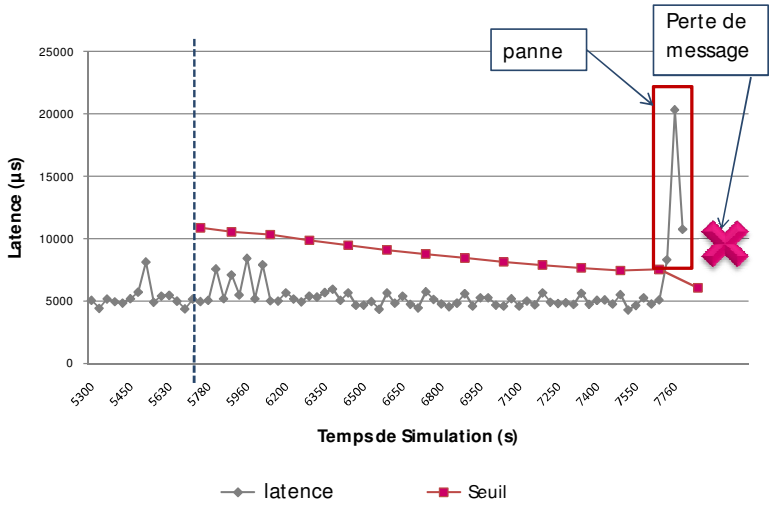

FIG. 4.14 – Système non réparable

– 2. Deuxième scénario

Dans ce scénario (Figure 4.15), nous adoptons le système publier/souscrire *Siena* avec une méthode de détection qui se base sur un seuil fixe faible. Dans ce cas, le système détecte toutes les violations, et introduit des opérations de réparation pour faire face à ces violations.

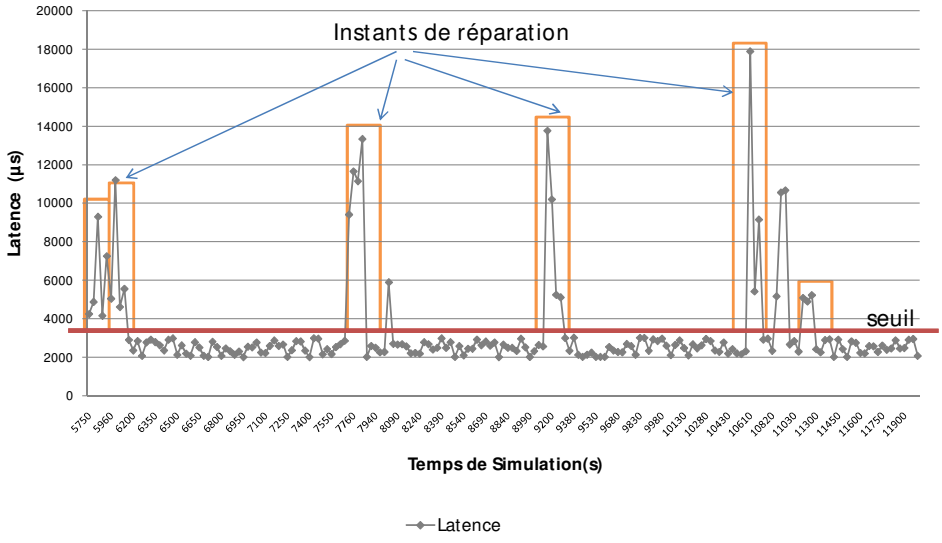

FIG. 4.15 – Une analyse basée sur un seuil fixe faible est incluse dans les *brokers*

– 3. Troisième scénario

Dans ce scénario, nous adoptons le système publier/souscrire *Siena* avec une méthode de détection qui se base sur un seuil fixe élevé. Dans ce cas, le système ne s'aperçoit pas aux pannes et n'introduit aucune action de réparation.

– 4. Quatrième scénario

Grâce au modèle analytique adoptant l'analyse réactive incluse dans les *brokers*, le système a pu détecter trois pannes respectivement à t=7820 $\mu\,s$, t=9260$\mu\,s$, t= 10730$\mu\,s$. Le déclenchement d'alarmes dans le système nous a incité à introduire une action de reconfiguration de durée 30 ms afin de réparer le système. Dans ce cas, la disponibilité du système est égale à 99,78 %.

– 5. Cinquième scénario

Un modèle analytique adoptant l'analyse hybride est inclus dans les *brokers*. Le système a fait une opération de migration vers l'analyse proactive dès qu'il s'aperçoit que le risque spécifié par l'utilisateur n'est plus garanti. Grâce à l'analyse proactive mise en place après l'opération de migration, les pannes ne surviennent plus au système. De ce fait, la disponibilité augmente à 99,84%.

Les résultats montrant le bénéfice d'utiliser l'analyse réactive et hybride sont récapitulés dans la Table 4.8.

TAB. 4.8 – Disponibilité du système publier/souscrire étudié

Système publier/souscrire	disponibilité
sans modèle analytique	55,45%
analyse se basant sur un seuil fixe faible	73,45%
analyse se basant sur seuil fixe élevé	55,45%
analyse réactive proposée	99,78 %
analyse hybride proposée	99,84 %

Nous déduisons à partir de cette table que le modèle analytique proposé augmente la disponibilité du système et réduit par conséquent le temps de dysfonctionnement. De plus, l'analyse hybride assure plus de disponibilité au système en comparaison avec l'analyse réactive.

En outre, les résultats issues de ce tableau démontrent que notre approche s'appuyant sur un seuil adaptatif offre une meilleure disponibilité du système en comparaison avec les approches s'appuyant sur des seuils fixes.
Par ailleurs, un seuil fixe et faible cause une détection fréquente des violations qui ne sont pas réellement des pannes dans le système, ceci induit à introduire des actions de réparation inutiles.
D'autre part, quand le seuil est fixe et élevé, le système ne s'aperçoit pas aux pannes lui affectant. De ce fait, aucune action n'est déclenchée par le système. Ceci cause bien évidemment la perte des messages et le dysfonctionnement total du système. Ce scénario sera donc presque équivalent au scénario 1, au cours duquel le système détecte la panne et n'introduit aucune action corrective.

En conclusion, nous pouvons affirmer que le module analytique proposé offre un état meilleur du système et contribue en grande partie à assurer son bon fonctionnement.

4.5 Etude de la complexité du module analytique proposé

Dans cette section, nous évaluons d'une façon théorique et pratique la complexité du module analytique proposé.

4.5.1 Etude théorique de la complexité du module analytique proposé

Afin d'estimer théoriquement la complexité de l'algorithme de *monitoring* et d'analyse, nous sommes amenés à évaluer la durée moyenne d'exécution de l'algorithme.
Etant donné que notre algorithme est composé de deux parties dépendantes (*monitoring* et analyse) dont l'exécution se fait d'une façon séquentielle, la complexité résultante du module analytique proposé est la somme de la complexité de l'algorithme de *monitoring* avec celle de l'algorithme d'analyse.

L'algorithme de *monitoring*, calcule pour chaque évènement reçu la latence comme étant la différence entre le temps de réception de l'évènement et le temps d'envoi de cet évènement par le broker voisin. Une fois calculé, le module de *monitoring* sauvegarde ces valeurs dans des fichiers logs, là où il enregistre aussi le broker émetteur, l'évènement, sa taille ainsi que le nombre de sauts séparant les deux brokers émetteur et récepteur.

La complexité de l'algorithme de *monitoring* pour chaque évènement correspond à la complexité du module de calcul du nombre de sauts car la récupération des autres paramètres correspond à une complexité de O(1). Ainsi la complexité du module de *monitoring* pour chaque évènement est de l'ordre de O(t) où t est la taille du réseau.

D'autre part, l'algorithme d'analyse proactive proposé (basé AR ou ARIMA) calcule la valeur de la latence prédite en effectuant des estimations qui dépendent de la taille de la fenêtre d'observations. Cette fenêtre d'observations est glissante dans l'axe du temps. De ce fait, à chaque évènement reçu, nous faisons m traitements appliqués sur les m dernières valeurs de latence, où m est la taille de la fenêtre d'observations.

Ainsi la complexité du module d'analyse proactive pour chaque évènement reçu est O(m) Où m est le nombre d'échantillons considérés pour effectuer les estimations.

Quant à l'analyse réactive, elle est composée de deux parties : La première est une période de prétraitement durant laquelle nous montrons l'adéquation de la série des valeurs maximales à la loi de *Gumbel*. Cette étape est cloturée par le calcul du seuil initial maximal. La période de prétraitement se déroule une seule fois pour tout le processus d'analyse.

La période de traitement, comme expliqué dans le chapitre 3, fait l'extraction de la valeur maximale de latence sur chaque période. Une mise à jour du seuil est ensuite effectuée en prenant en considération la valeur initiale du seuil ainsi que le seuil précédant. De ce fait, le traitement correspondant au n^{ieme} évènement est proportionnel à n où n est le nombre d'échantillons des valeurs de latence.

En conclusion, les traitements d'analyse réactive effectués pour l'ensemble des (n) évènements vaut :

$$\sum_{i=1}^{n}(traitement_{e_i}) \tag{4.6}$$

Où $traitement_{e_i}$ est le traitement effectué pour l'évènement e_i. La complexité résultante est donc $O(n * (n+1)/2) = O(n^2)$ qui est une complexité quadratique.

4.5.2 Etude pratique de la complexité temporelle du module analytique proposé

Afin d'étudier d'une façon pratique la complexité temporelle du module analytique proposé, nous avons mesuré le surcoût introduit par les modules de *monitoring* et d'analyse. Le but donc derrière cette étude est de mesurer la rapidité de l'algorithme de *monitoring* et d'analyse en fonction de la taille des évènements. Le scénario consiste à faire transiter des évènements de différentes tailles entre deux *brokers* voisins sans et avec l'introduction du module analytique proposé. Pendant le test, un producteur émet des évènements vers un premier *broker* B_1, ce dernier transmet ce message vers son *broker* voisin B_2. Le temps d'exécution mesuré est donc le temps qui sépare la début de simulation avec le temps de réception de ces évènements par le *broker* voisin B_2.

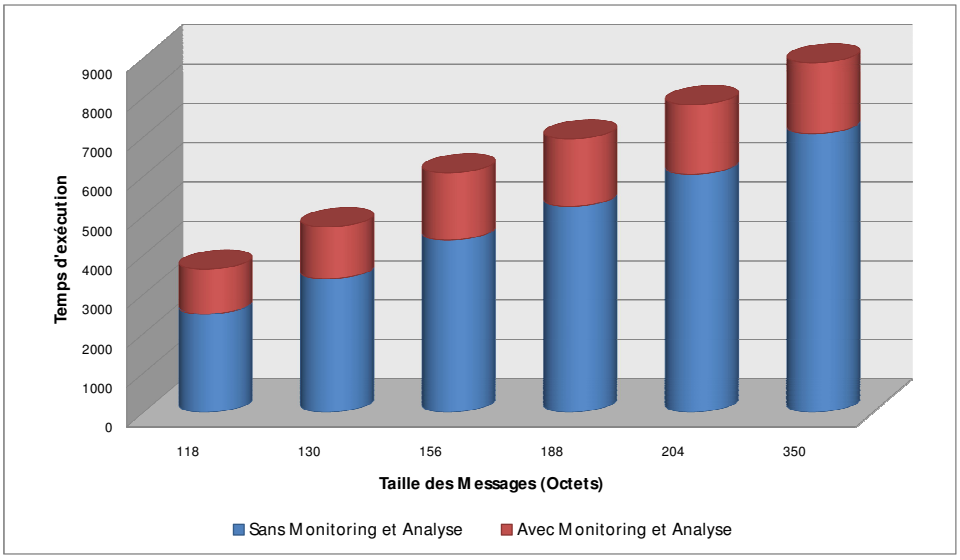

FIG. 4.16 – *Overhead* introduit par le module analytique proposé

Comme le montre la Figure 4.16, le temps d'exécution après l'introduction du module

analytique est légèrement supérieur à celui mesuré sans l'introduction du module proposé. En effet, pour un message comportant 346 octets, le temps d'exécution est de 7100 μs sans modules de *monitoring* et d'analyse et il mesure environ 8200 μs après avoir introduit ces deux modules. Nous pouvons donc conclure que le module analytique proposée offre un surcoût relativement faible.

4.5.3 Etude pratique de la complexité spatiale du module analytique proposé

L'étude de la complexité spatiale a pour but de mesurer l'occupation mémoire ou l'espace mémoire alloué par le calcul de l'algorithme de *monitoring* et d'analyse en fonction du nombre d'évènements.

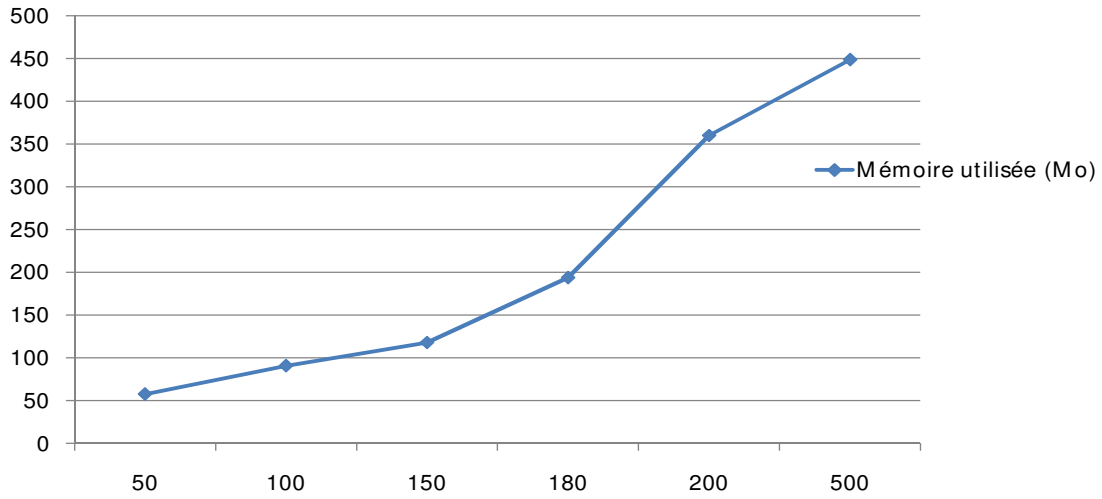

FIG. 4.17 – Mémoire utilisée en fonction du nombre d'évènements

D'après les résultats tirés (Figure 4.17), la consommation mémoire requise pour l'algorithme sur le simulateur *Jist/Swans* ne dépasse pas 360 Mo pour un nombre d'évènements égal à 200. Cependant, elle varie entre 57 Mo et 117 Mo lorsque le nombre d'évènements varie entre 50 et 150 évènements.

Nous pouvons donc conclure que le module analytique orienté QdS proposée introduit

une consommation mémoire négligeable qui n'entraine auncune perturbation sur le système.

4.6 Etude de la stabilité combinatoire du module analytique proposé

Afin d'étudier le comportement du système après l'introduction du module analytique proposé, nous nous sommes intéressés à la notion de stabilité combinatoire, connue sous la nomunation anglaise "combinatorial stability".

D'après [CM01], l'approche est combinatoirement stable si et seulement si les changements de la topologie surviennent lentement de manière à ce que le système analyse et gère correctement la situation.

D'une façon analytique, l'approche est combinatoirement stable seulement si et seulement si $t_\mu > t_\omega$, avec :

- t_μ : l'intervalle de temps entre deux changements de la topologie.
- t_ω : le temps requis par l'approche pour l'analyse de la situation.

Dans ce cas, l'approche sera capable d'analyser correctement la situation avant qu'il n'y ait un changement significatif de l'état du système.

D'après les expérimentations faites, le temps requis par l'algorithme de *monitoring* et d'analyse est négligeable (de l'ordre de 1500 μs). De ce fait, la relation $t_\mu > t_\omega$ demeurre vérifiée. Notamment, nous pouvons aussi affirmer le comportement de la stabilité combinatoire de l'approche tant que les deux valeurs t_μ et t_ω sont proches l'une à l'autre.
Par conséquent, le système arrive dans le cas ordinaire (lorsque le déplacement des nœuds est non brusque) à analyser les situations avant l'occurence des changements de positions. Entre autres, les changements de positions affectant les nœuds non figurant dans le chemin des évènements n'influent pas sur le *monitoring* et l'analyse.

Cependant, les mouvements très rapides et brusques des nœuds appartenant au chemin des évènements peuvent contredire la loi de la stabilité combinatoire de l'approche. En effet, dans ce cas, il y aura une élévation brusque des valeurs de latence qui sera non

perceptible par l'approche.

4.7 Conclusion

Tout au long de ce chapitre, nous avons présenté les résultats des expérimentations réalisées autour du module analytique proposé sur le simulateur *Jist/Swans* intégrant le middleware *Siena*.

Nous avons présenté dans un premier temps un cas d'étude représentant un risque élevé, sur lequel, nous avons testé notre module d'analyse proactive. Dans un deuxième temps, nous avons présenté une application multimédia présentant un risque moins élevé que la première, et nous avons présenté les résultats issus des deux modules d'analyse réactive et hybride. Ces résultats présentent une preuve expérimentale du modèle analytique orienté QdS proposé.

Ensuite, un test de disponibilité du système, sans et avec, le module d'analyse a montré la valeur ajouté par le module proposé. Finalement, une étude de la complexité spatiale et temporelle du module analytique proposée a été réalisée et présentée à la fin de cette partie.

Conclusion générale

La principale motivation qui régit ce travail est le développement d'un modèle analytique orienté QdS pour les systèmes publier/souscrire déployés sur MANET. Le modèle ainsi proposé s'appuie sur des méthodes statistiques et englobe une phase de *monitoring*, d'analyse et de diagnostic. L'approche proposée est une approche distribuée qui intègre au sein de chaque *broker* des composants responsables de réaliser les fonctionnalités offertes par le modèle introduit dans cette thèse.

Les principales contributions de ce travail sont :

- Le *monitoring* : se fait par collection des valeurs de la latence et leurs sauvegardes dans des fichiers logs. Ceci est réalisé par interception d'évènements. .
- L'analyse : correspond à la phase d'exploitation des valeurs issues du *monitoring*. Cette fonction permet de s'assurer du bon fonctionnement de l'application, et le cas échéant déclencher des alarmes. L'approche développée offre à l'utilisateur une variété de choix. Les fonctionnalités offertes peuvent se résumer comme suit :

  ◇ une analyse réactive : cette approche vise à détecter les dégradations de QdS par comparaison des valeurs de latence avec des seuils adaptatifs. Le calcul des seuils débute par une approximation de la série formée par les valeurs de latence à des distributions empiriques à savoir la loi de *Gumbel*. Ensuite, à l'aide de la formule de la *Moyenne Mobile Exponentielle*, la valeur du seuil initialement calculée est mise à jour dynamiquement.

  ◇ une analyse proactive : Cette forme d'analyse a pour finalité de prédire les pannes pouvant affecter le système. Elle s'appuie sur une estimation de la valeur de la latence à l'aide de la méthode d' Auto Régression Linéaire et sur une estimation de la

valeur du seuil. La comparaison de ces deux valeurs estimées permet de prédire les pannes affectant éventuellement le système. La méthode ARIMA *(Auto Regressive Integrated Moving Average Formula)* a été aussi utilisée afin de minimiser les erreurs de prédictions.

◇ une analyse hybride : Cette analyse est dite hybride du coup qu'elle combine les deux formes d'analyse pré-décrites. Le système déclenche d'abord une analyse réactive durant laquelle il s'auto contrôle. S'il note une violation de la contrainte spécifiée par l'utilisateur, le système déclenche une analyse proactive. Une fois que le système se stabilise, il revient de nouveau vers l'analyse réactive et le cycle se répète.

Ce processus se poursuit par l'identification de la source de la dégradation.

– Le diagnostic est déclenché suite à la détection d'une dégradation de QdS. Dans ce cadre, la méthode de la *corrélation* a été utilisé afin de mesurer l'intensité de la liaison entre la variation de la latence d'une part et la variation au niveau du nombre de sauts ou la charge d'autre part.

Des expérimentations menées à l'aide du simulateur *Jist/Swans*, sous différentes conditions, ont montré l'efficacité des modules développés. De plus, une mesure des paramètres de performance du système avant et après introduction des modules développés a montré leur efficacité.

Le perspectives des travaux présentés dans ce mémoire suivent différent axes de réflexion. À court terme, nous prévoyons l'extension du module d'analyse pour qu'il supporte les dégradations brusques au niveau de la latence. Ceci nécessite certainement des approximations de la série des valeurs de latence à d'autres lois mathématiques. Pour réaliser de tel ajustement, il est nécessaire de faire une étude exhaustive de ces lois.

Nous prévoyons aussi achever les deux autres phases du processus d'auto-adaptation à savoir planification et exécution. Dans ce sens, nous avons commencé à proposer et à valider des actions de reconfigurations afin de réparer le système [LRJC12]. De tels mécanismes peuvent avoir recours à la substitution ou à la migration d'un *broker* vers un autre nœud du réseau offrant une meilleure QdS. D'autres actions de reconfiguration peuvent être envisagées telles que la fusion ou la duplication de *brokers*. Nous avons donc tracé l'arbre des

solutions des différentes opérations de reconfiguration possibles qui peuvent être effectuées pendant l'exécution afin de réparer le système. Ceci nous a amené à proposer un protocole de gestion dynamique de l'architecture tout en définissant des règles de transformations décrivant les éléments déclencheurs des différentes reconfigurations possibles de l'architecture. Afin de réaliser ça, nous avons eu recours à la technique formelle des grammaires de graphes pour pouvoir décrire l'architecture. L'approche ACG (Abstract Component Graph) pourrait aussi être utilisée pour traiter l'évolution dynamique des architectures logicielles par la transformation de graphe. Le moteur de transformation de graphes (GMTE), a joué le rôle de moteur de transformation de graphe afin de trouver toutes reconfigurations pouvant être appliquées sur le système.

Dans le but d'enrichir le modèle analytique proposé, nous envisageons aussi inclure un autre paramètre de performance du système tel que le taux de perte. L'idée derrière cette extension est de se baser sur la théorie qui affirme la corrélation existante entre la latence et la perte de message. Ceci permet donc d'utiliser les résultats du module de *monitoring* dans sa version actuelle et d'en servir pour prédire la perte de messages. Une étude approfondie de la série des valeurs de latence et l'étude de ses tendances permettra d'appuyer le module d'analyse proposé. En effet, en utilisant les tendances à court et à long terme, et en se basant sur les mêmes seuils adaptatifs proposés au cours de la phase d'analyse, nous envisageons calculer la probabilité de perte du message à l'instant futur. Le prédicteur est donc une somme pondérée des indicateurs de tendance à court et à long terme et d'une variable qui mesure la position de la valeur de latence dans l'intervalle formé par les seuils.

À moyen terme, nous planifions rendre standard le modèle analytique proposé afin de l'utiliser dans différentes applications et plus spécifiquement au niveau des services Web.

En outre, nous prévoyons mesurer le niveau de satisfaction du client consommateur vis-à-vis du service offert par le système après l'ajout du module analytique proposé. Dans ce sens, nous visons la qualité d'expérience ou la qualité de perception du client envers les notifications reçus via le système. Ceci traduit l'appréciation du service offert par le système de la part du consommateur. La qualité d'expérience permet donc une mesure

de bout en bout du service offert par le système tout en faisant référence à un ensemble d'utilisateurs.

Ceci est possible à travers la mesure de la note d'opinion moyenne (MOS [2]) permettant d'attribuer une note au service reçu à travers le système afin de caractériser sa qualité. Concrètement, un consommateur pourrait attribuer une note d'opinion moyenne concernant le flux multimédia qu'il a reçu du système basé évènements reflétant la qualité de la vidéo.

[2]mean opinion score

Publications de Imene LAHYANI

Publications dans des journaux

[**J1**] Imene Lahyani, Mohamed Jmaiel and Christophe Chassot. Analytical decisional model for latency aware publish/subscribe systems on MANET. *Journal of Systems and Software*. November 10, 2015. Elsevier.(Impact Factor : 1,485)

[**J2**] Imene Lahyani, Mohamed Jmaiel, Khalil Drira and Christophe Chassot. Latency Aware Publish/Subscribe Systems on MANET.*The International Journal of Wireless and Mobile Computing (IJWMC)*. 2015, Vol.8, No.3, pp.236 - 248.

Manifestations Internationales avec actes

[**C1**] Imene Lahyani, Mohamed Jmaiel, Khalil Drira and Christophe Chassot. Analytical decisional model for publish/subscribe systems on MANET. *the 23 st International Conference Collaboration Technologies and Infrastructure (WETICE 2014). 2nd Track on Adaptive and Reconfigurable Service-oriented and component-based Applications and Architectures (AROSA 2014)*. June 23-25, 2014. Parma, Italy. IEEE Computer Society. (Class B)

[**C2**] Imene Lahyani, Lamia Ben Amor, Mohamed Jmaiel, Khalil Drira and Christophe Chassot. Analytical Framework for QoS aware Publish/Subscribe System deployed on MANET.*the 10th IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA 2012)*. July 10-13, 2012. Leganés, Madrid. IEEE Computer Society. (Class B)

[**C3**] Imene Lahyani, Mouna Gassara and Mohamed Jmaiel. Predictive schemes for QoS awareness of publish/subscribe systems on MANET.*the 11th International Symposium on Parallel and Distributed Computing (ISPDC 2012)*. June 25-29, 2012. Munich, Germany.

IEEE Computer Society. (Class C)

[C4] Imene Lahyani, Wafa Makki and Christophe Chassot. Failure Prediction for Publish/Subscribe System on MANET. *the 21 st International Conference Collaboration Technologies and Infrastructure (WETICE 2012). 2nd Track on Adaptive and Reconfigurable Service-oriented and component-based Applications and Architectures (AROSA 2012).* June 25-27, 2012. Toulouse, France. IEEE Computer Society. (Short paper) (Class B)

[C5] Imene Lahyani, Ismael Bouassida Rodriguez, Mohamed Jmaiel and Chrisptophe Chassot. Towards self healing publish/subscribe system on MANET. *the 21 st International Conference Collaboration Technologies and Infrastructure (WETICE 2012). 2nd Track on Collaborative Technology for Coordinating Crisis Management (CT2CM 2012).* June 25-27, 2012. Toulouse, France. IEEE Computer Society. (Class B)

[C6] Imene Lahyani, Nesrine Khabou, and Mohamed Jmaiel. QoS Monitoring and Analysis Approach for Publish/Subscribe Systems deployed on MANET, *The 20th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP 2011). February 15-17, 2012, Garching, Germany.* pages 120-124. IEEE Computer Society.(Short paper) (Class C)

[C7] Imene Lahyani and Mohamed Jmaiel. Experimentations for QoS Evaluation of Publish/Subscribe Systems deployed on Ad-hoc Network, *the 6 th International Workshop on Performance Analysis and Enhancement of Wireless Networks (PAEWN'11) : IEEE Workshops of International Conference on Advanced Information Networking and Applications*. March 22-25, 2011, Biopolis, Singapore. pages 676-681, IEEE Computer Society. (Class B)

[C8] Imene Lahyani, Soumaya Marzouk, and Mohamed Jmaiel. QoS aware publish/subscribe system deployed on a mobile ad-hoc network. *Journal of SIWN - The Systemics and Informatics World Network, Volume 7 :84-90, May 2009. In the 5th International Conference on Self-organization and Adaptation of Computing and Communications (SACC 2009).*

.1 Les tables de Shapiro et Wilk

La première partie de cet annexe représente les tables de Shapiro et Wilk.

– les tables des coefficients a_i

n / j	2	3	4	5	6	7	8	9	10
1	0.7071	0.7071	0.6872	0.6646	0.6431	0.6233	0.6052	0.5888	0.5739
2	0.000	0.000	0.1677	0.2413	0.2806	0.3031	0.3164	0.3244	0.3291
3	0.000	0.000	0.000	0.000	0.0875	0.1401	0.1743	0.1976	0.2141
4	0.000	0.000	0.000	0.000	0.000	0.000	0.0561	0.0947	0.1224
5	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.0399

N / j	11	12	13	14	15	16	17	18	19	20
1	0.5601	0.5475	0.5359	0.5251	0.5150	0.5056	0.4963	0.4886	0.4808	0.4734
2	0.3315	0.3325	0.3325	0.3318	0.3306	0.3290	0.3273	0.3253	0.3232	0.3211
3	0.2260	0.2347	0.2412	0.2460	0.2495	0.2521	0.2540	0.2553	0.2561	0.2565
4	0.1429	0.1586	0.1707	0.1802	0.1878	0.1939	0.1988	0.2027	0.2059	0.2085
5	0.0695	0.0922	0.1099	0.1240	0.1353	0.1447	0.1524	0.1587	0.1641	0.1686
6	0.000	0.0303	0.0539	0.0727	0.0880	0.1005	0.1109	0.1197	0.1271	0.1334
7	0.000	0.000	0.000	0.0240	0.0433	0.0593	0.0725	0.0837	0.0932	0.1013
8	0.000	0.000	0.000	0.000	0.000	0.0196	0.0359	0.0496	0.0612	0.0711
9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.0163	0.0303	0.0422
10	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.0140

n / j	21	22	23	24	25	26	27	28	29	30
1	0.4643	0.4590	0.4542	0.4493	0.4450	0.4407	0.4366	0.4328	0.4291	0.4254
2	0.3185	0.3156	0.3126	0.3098	0.3069	0.3043	0.3018	0.2992	0.2968	0.2944
3	0.2578	0.2571	0.2563	0.2554	0.2543	0.2533	0.2522	0.2510	0.2499	0.2487
4	0.2119	0.2131	0.2139	0.2145	0.2148	0.2151	0.2152	0.2151	0.2150	0.2148
5	0.1736	0.1764	0.1787	0.1807	0.1822	0.1836	0.1848	0.1857	0.1064	0.1870
6	0.1399	0.1443	0.1480	0.1512	0.1539	0.1563	0.1584	0.1601	0.1616	0.1630
7	0.1092	0.1150	0.1201	0.1245	0.1283	0.1316	0.1346	0.1372	0.1395	0.1415
8	0.0804	0.0878	0.0941	0.0997	0.1046	0.1089	0.1128	0.1162	0.1192	0.1219
9	0.530	0.0618	0.0696	0.0764	0.0823	0.0876	0.0923	0.0965	0.1002	0.1036
10	0.0263	0.0368	0.0459	0.0564	0.0610	0.0672	0.0728	0.0965	0.0822	0.0862
11	0.000	0.0122	0.0228	0.0539	0.0403	0.0476	0.0540	0.0778	0.0650	0.0697
12	0.000	0.000	0.000	0.0321	0.0200	0.0284	0.0358	0.0598	0.0483	0.0537
13	0.000	0.000	0.000	0.0107	0.000	0.0094	0.0178	0.0424	0.0320	0.0381
14	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.0253	0.0159	0.0227
15	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.0084	0.0076

– La table des valeurs du nombre W

N	W (0.05)	W (0.01)
5	0.772	0.686
6	0.988	0.713
7	0.803	0.730
8	0.818	0.749
9	0.929	0.764
10	0.842	0.781
11	0.850	0.792
12	0.859	0.805
13	0.866	0.814
14	0.874	0.825
15	0.881	0.835
16	0.887	0.844
17	0.892	0.851
18	0.897	0.858
19	0.901	0.863
20	0.905	0.868
21	0.808	0.873
22	0.911	0.878
23	0.914	0.881
24	0.916	0.884
25	0.918	0.888
26	0.920	0.891
27	0.923	0.894
28	0.924	0.876
29	0.926	0.898
30	0.927	0.900

.2 La table de Student

La deuxième partie de cet annexe représente la table de Student.

	0.50	0.20	0.10	0.05	0.02	0.01	0.005	0.002	0.001	0.0001
1	1.000	3.078	6.314	12.706	31.281	63.657	127.32	318.31	636.62	6366.2
2	0.816	1.886	2.920	4.303	6.965	9.925	14.089	22.327	34.599	99.992
3	0.765	1.638	2.353	3.182	4.541	5.841	7.453	10.215	12.924	28.000
4	0.741	1.533	2.132	2.776	3.747	4.604	5.598	7.173	8.610	15.544
5	0.727	1.476	2.015	2.571	3.365	4.032	4.773	5.893	6.869	11.178
6	0.718	1.440	1.943	2.447	3.143	3.707	4.317	5.208	5.959	9.082
7	0.711	1.415	1.895	2.365	2.998	3.499	4.029	4.785	5.408	7.885
8	0.706	1.397	1.860	2.306	2.896	3.355	3.833	4.501	5.041	7.120
9	0.703	1.383	1.833	2.262	2.821	3.250	3.690	4.297	4.781	6.594
10	0.700	1.372	1.812	2.228	2.764	3.169	3.581	4.144	4.587	6.211
11	0.697	1.363	1.796	2.201	2.718	3.106	3.497	4.025	4.437	5.921
12	0.695	1.356	1.782	2.179	2.681	3.055	3.428	3.930	4.318	5.694
13	0.694	1.350	1.771	2.160	2.650	3.012	3.372	3.852	4.221	5.513
14	0.692	1.345	1.761	2.145	2.624	2.977	3.326	3.787	4.140	5.363
15	0.691	1.341	1.753	2.131	2.602	2.947	3.286	3.733	4.073	5.239
16	0.690	1.337	1.746	2.120	2.583	2.921	3.252	3.686	4.015	5.134
17	0.689	1.333	1.740	2.110	2.567	2.898	3.222	3.646	3.965	5.044
18	0.688	1.330	1.734	2.101	2.552	2.878	3.197	3.610	3.922	4.966
19	0.688	1.328	1.729	2.093	2.539	2.861	3.174	3.579	3.883	4.897
20	0.687	1.325	1.725	2.086	2.528	2.845	3.153	3.552	3.850	4.837
21	0.686	1.323	1.721	2.080	2.518	2.831	3.135	3.527	3.819	4.784
22	0.686	1.321	1.717	2.074	2.508	2.819	3.119	3.505	3.792	4.736
23	0.685	1.319	1.714	2.069	2.500	2.807	3.104	3.485	3.768	4.693
24	0.685	1.318	1.711	2.064	2.492	2.797	3.091	3.467	3.745	4.654
25	0.684	1.316	1.708	2.060	2.485	2.787	3.078	3.450	3.725	4.619
30	0.683	1.310	1.697	2.042	2.457	2.750	3.030	3.385	3.646	4.482
35	0.682	1.306	1.690	2.030	2.438	2.724	2.996	0.340	3.591	4.389
40	0.681	1.303	1.684	2.021	2.423	2.704	2.971	3.307	3.551	4.321
45	0.680	1.301	1.679	2.014	2.412	2.690	2.952	3.281	3.520	4.269
50	0.679	1.299	1.676	2.009	2.403	2.678	2.937	3.261	3.496	4.228
60	0.679	1.296	1.671	2.000	2.390	2.660	2.915	3.232	3.460	4.169
70	0.678	1.294	1.667	1.994	2.381	2.648	2.899	3.211	3.435	4.127
80	0.678	1.292	1.664	1.990	2.374	2.639	2.887	3.195	3.416	4.096
90	0.677	1.291	1.662	1.987	2.368	2.632	2.878	3.183	3.402	4.072
100	0.677	1.290	1.660	1.984	2.364	2.626	2.871	3.174	3.390	4.053
150	0.676	1.287	1.655	1.976	2.351	2.609	2.849	3.145	3.357	3.998
200	0.676	1.286	1.653	1.972	2.345	2.601	2.839	3.131	3.340	3.970
300	0.675	1.284	1.650	1.968	2.339	2.592	2.828	3.118	3.323	3.944
500	0.675	1.283	1.648	1.965	2.334	2.586	2.820	3.107	3.310	3.922
1000	0.675	1.282	1.646	1.962	2.330	2.581	2.813	3.098	3.300	3.906
infini	0.674	1.282	1.645	1.960	2.326	2.576	2.807	3.090	3.291	3.891

Bibliographie

[AKZ99] Guy Almes, Sunil Khalidindi, and Matt Zekauskas, *A one-way delay metric for ippm, rfc 2679*, 1999.

[Ber86] Richard Bertrand, *Pratique de l'analyse statistique des données*, Presses de l'Université du Québec, 1986.

[BHR05] Rimon Barr, Zygmunt Haas, and Robbert V. Renesse, *Scalable wireless ad hoc network simulation*, ch. 19, pp. 297–311, CRC Press, aout 2005.

[BHvR] Rimon Barr, Zygmunt Haas, and Robbert van Renesse, *JiST/SWANS : Java in Simulation Time / Scalable Wireless Ad hoc Network Simulator*, http ://jist.ece.cornell.edu/.

[BHvR05] Rimon Barr, Zygmunt J. Haas, and Robbert van Renesse, *Jist : an efficient approach to simulation using virtual machines.*, Softw., Pract. Exper. (2005), 539–576.

[bib] *La loi de gumbel*, http ://www.jybaudot.fr/Probas/gumbel.html.

[CAR05] Nuno Carvalho, Filipe Araujo, and Luis Rodrigues, *Scalable qos-based event routing in publish-subscribe systems*, Proceedings of the Fourth IEEE International Symposium on Network Computing and Applications (Washington, DC, USA), IEEE Computer Society, 2005, pp. 101–108.

[CB12] Justin Collins and Rajive Bagrodia, *A quantitative comparison of communication paradigms for manets*, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering **73** (2012), 261–272.

[CDK02] Miguel Castro, Peter Druschel, and Anne-Marie Kermarrec, *Scribe : a large-scale and decentralized application-level multicast infrastructure*, Selected Areas in Communications, IEEE Journal on **20** (2002), no. 8, 1489 – 1499.

[CER13] Domenico Cotroneo Christian Esposito and Stefano Russo, *On reliability in publish/subscribe services*, Computer Networks **57** (2013), no. 5, 1318–1343.

[CHA] ARTHUR CHARPENTIER, *Cours de series temporelles theorie et applications*, Université Paris Dauphine.

[CM99] Christina Corso and Joseph Macker, *Mobile ad hoc networking (manet) : Routing protocol performance issues and evaluation considerations*, 1999.

[CM01] Satyabrata Chakrabarti and Amitabh Mishra, *Qos issues in ad hoc wireless networks*, IEEE Communications Magazine **39** (2001), 142–148.

[CNRS98] Eric Crawley, Raj Nair, Bala Rajagopalan, and Hal Sandick, *A framework for qos-based routing in the internet*, 1998.

[Col14] Justin Scott Collins, *Communication paradigms for mobile ad hoc networks*, Ph.D. thesis, University of California, Los Angeles, 2014.

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

[CP06] Gianpaolo Cugola and Gian Pietro Picco, *Reds : a reconfigurable dispatching system*, Proceedings of the 6th international workshop on Software engineering and middleware (New York, NY, USA), SEM '06, ACM, 2006, pp. 9–16.

[CRW01] Antonio Carzaniga, David S. Rosenblum, and Alexander L. Wolf, *Design and evaluation of a wide-area event notification service*, ACM Trans. Comput. Syst. **19** (2001), 332–383.

[CS05] Yuan Chen and Karsten Schwan, *Opportunistic overlays : efficient content delivery in mobile ad hoc networks*, Proceedings of the ACM/IFIP/USENIX 2005 International Conference on Middleware (New York, NY, USA), Middleware '05, Springer-Verlag New York, 2005, pp. 354–374.

[DSM07] Mieso K. Denko, Elhadi Shakshuki, and Haroon Malik, *A mobility-aware and cross-layer based middleware for mobile ad hoc networks*, 21st International Conference on Advanced Information Networking and Applications, 2007., may 2007, pp. 474 –481.

[DSM09] Mieso K. Denkoa, Elhadi Shakshukib, and Haroon Malikc, *Enhanced cross-layer based middleware for mobile ad hoc networks*, Journal of Network and Computer Applications **32** (2009), 490–499.

[eBR06] Emmanuel César et Bruno Richard, *Les séries temporelles*, Mars 2006.

[ECR12] Christian Esposito, Domenico Cotroneo, and Stefano Russo, *A tutorial on reliability in publish/subscribe services*, Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems, July 2012, pp. 399–406.

[EFGK03] Patrick Eugster, Pascal Felber, Rachid Guerrapui, and Anne-Marie Kermarrec, *The many faces of publish/subscribe*, ACM Comput. Surv. **35** (2003), 114–131.

[FJLM05] Eli Fidler, Hans-Arno Jacobsen, Guoli Li, and Serge Mankovskii, *The padres distributed publish/subscribe system*, In 8th International Conference on Feature Interactions in Telecommunications and Software Systems, 2005, pp. 12–30.

[GN08] Giada Giorgi and Claudio Narduzzi, *Detection of anomalous behaviors in networks from traffic measurements*, IEEE Transactions on Instrumentation and Measurement **57** (2008), no. 12, 2782 –2791.

[HGDJ08] Riadh Ben Halima, Karim Guennoun, Khalil Drira, and Mohamed Jmaiel, *Providing predictive self-healing for web services : A qos monitoring and analysis-based approach*, Journal of Information Assurance and Security (2008).

[HMS09] Joe Hoffert, Daniel Mack, and Douglas Schmidt, *Using machine learning to maintain pub/sub system qos in dynamic environments*, Proceedings of the 8th International Workshop on Adaptive and Reflective MIddleware, 2009.

[HT07] Lajos Hanzo and Rahim Tafazolli, *A survey of qos routing solutions for mobile ad hoc networks*, IEEE Communications Surveys and Tutorials **9** (2007), no. 2, 50–70.

[JFF07] Zbigniew Jerzak, Robert Fach, and Christof Fetzer, *Fail-aware publish/subscribe*, Sixth IEEE International Symposium on Network Computing and Applications, 2007., july 2007, pp. 113 –125.

[JMV08] Hojjat Jafarpour, Sharad Mehrotra, and Nalini Venkatasubramanian, *A fast and robust content-based publish/subscribe architecture*, Seventh IEEE International Symposium on Network Computing and Applications, 2008., july 2008, pp. 52 –59.

[Jol07] Jean Michel Jolion, *Probabilités et statistique*, INSA Lyon, 2007.

[KGD07] Dhafer Ben Khedher, Roch Glitho, and Rachida Dssouli, *A novel overlay-based failure detection architecture for manet applications*, Networks, 2007. ICON 2007. 15th IEEE International Conference on, nov. 2007, pp. 130 –135.

[KJ09] Reza Sherafat Kazemzadeh and Hans-Arno Jacobsen, *Reliable and highly available distributed publish/subscribe service*, Proceedings of the 28th IEEE International Symposium on Reliable Distributed Systems, September 2009, pp. 41 –50.

[KKY+10] Minkyong Kim, Kyriakos Karenos, Fan Ye, Johnathan Reason, Hui Lei, and Konstantin Shagin, *Efficacy of techniques for responsiveness in a wide-area publish/subscribe system*, Proceedings of the 11th International Middleware Conference Industrial track (New York, NY, USA), Middleware Industrial Track '10, ACM, 2010, pp. 40–45.

[KN00] Samuel Kotz and Saralees Nadarajah, *Extreme value distributions theory and applications*, Imperial College Press, 2000.

[Kni89] Keith Knight, *Consistency of akaike's information criterion for infinite variance autoregressive processes*, The Annals of Statistics **17** (1989), 824–840.

[KTKS09] Tae-Hoon Kim, David Tipper, Prashant Krishnamurthy, and Lee Swindlhurst, *Improving the topological resilience of mobile ad hoc networks*, in Proceedings of the 7th International Workshop onDesign of Reliable Communication Networks,DRCN 2009, oct 2009, pp. 191 –197.

[LAJ+12] Imene Lahyani, Lamia Ben Amor, Mohamed Jmaiel, Khalil Drira, and Christophe Chassot, *Analytical framework for qos aware publish/subscribe system deployed on MANET*, 10th IEEE International Symposium on Parallel and Distributed Processing with Applications, ISPA 2012, Leganes, Madrid, Spain, July 10-13, 2012, 2012, pp. 325–331.

[LGJC12] Imene Lahyani, Mouna Gassara, Mohamed Jmaiel, and Christophe Chassot, *Predictive schemes for qos awareness of publish/subscribe systems on MANET*, 11th International Symposium on Parallel and Distributed Computing, ISPDC 2012, Munich, Germany, June 25-29, 2012, 2012, pp. 19–25.

[LJ11] Imene Lahyani and Moahmed Jmaiel, *Experimentations for qos evaluation of publish/subscribe systems deployed on ad-hoc network*, Proceedings of the IEEE Workshops of International Conference on Advanced Information Networking and Applications, Middleware Industrial Track '10, 2011.

[LJC15] Imene Lahyani, Mohamed Jmaiel, and Christophe Chassot, *Analytical decisional model for latency aware publish/subscribe systems on manet*, Journal of Systems and Software (2015).

[LJDC15] Imene Lahyani, Mohamed Jmaiel, Khalil Drira, and Christophe Chassot, *Latency-aware publish/subscribe systems on MANET*, IJWMC **8** (2015), no. 3, 236–248.

[LKJ12] Imene Lahyani, Nesrine Khabou, and Mohamed Jmaiel, *Qos monitoring and analysis approach for publish/subscribe systems deployed on MANET*, Proceedings of the 20th Euromicro International Conference on Parallel, Distributed and Network-Based Processing, PDP 2012, Munich, Germany, February 15-17, 2012, 2012, pp. 120–124.

[LMC12] Imene Lahyani, Wafa Makki, and Christophe Chassot, *Failure prediction for publish/subscribe system on MANET*, 21st IEEE International Workshop on Enabling Technologies : Infrastructure for Collaborative Enterprises, WETICE 2012, Toulouse, France, June 25-27, 2012, 2012, pp. 98–100.

[LMJ09] Imene Lahyani, Soumaya Marzouk, and Mohamed Jmaiel, *Qos aware publish/subscribe system deployed on a mobile ad-hoc network*, Proceeding of the 5th International Conference on Self-organization and Adaptation of Computing and Communications (SACC 2009) (Leipzig, Germany), March 2009.

[Lon95] François Longin, *La théorie des valeurs extrêmes : Présentation et premières applications en finance*, 77–97.

[LRJC12] Imene Lahyani, Ismael Bouassida Rodriguez, Mohamed Jmaiel, and Christophe Chassot, *Towards self healing publish/subscribe system on MANET*, 21st IEEE International Workshop on Enabling Technologies : Infrastructure for Collaborative Enterprises, WETICE 2012, Toulouse, France, June 25-27, 2012, 2012, pp. 385–390.

[LS90] James Lucasa and Michael Saccuccib, *Exponentially weighted moving average control schemes : properties and enhancements*, Technometrics **32** (1990), 1–29.

[LTJ12] Stephen Lapan, MaryLynn T.Quartaroli, and Frances J.Riemer, *Qualitative research, an introduction to methods and designs*, Jossey-Bass, 2012.

[LYW13] Sheng Liu, Yang Yang, and Weixing Wang, *Research of aodv routing protocol for ad hoc networks1*, Science direct, Elsevier **5** (2013), 21–31.

[MA05] Giovanni Turi Marco Avvenuti, Alessio Vecchio, *A cross-layer approach for publish/subscribe in mobile ad hoc networks*, Mobility Aware Technologies and Applications (Thomas Magedanz, Ahmed Karmouch, Samuel Pierre, and Iakovos Venieris, eds.), Lecture Notes in Computer Science, vol. 3744, Springer Berlin / Heidelberg, 2005, 10.1007/11569510-20, pp. 203–214.

[MB08] Shruti Mahambre and Umesh Bellur, *An adaptive approach for ensuring reliability in event based middleware*, Proceedings of the Second International Conference on Distributed Event-based Systems (DEBS 08), 2008, pp. 157–168.

[MBCK12] TobiasR. Maye, Lionel Brunie, David Coquil, and Harald Kosch, *On reliability in publish/subscribe systems : a survey*, International Journal of Parallel, Emergent and Distributed Systems archive **27** (2012), no. 5, 369–386.

[MCP11] Amirhossein Malekpour, Antonio Carzaniga, and Fernando Pedone, *End-to-end reliability for best-effort content-based publish/subscribe networks*, Proceedings of the 28th IEEE International Symposium on Reliable Distributed Systems, July 2011.

[Mer05] Rabah Meraihi, *Gestion de la qualité de service et contrôle de topologie dans les réseaux mobiles ad hoc*, Ph.D. thesis, Paristech ;ENST ;INFRES Informatique et Réseaux, 2005.

[MK12] Selli Mohapatra and Priyadarshi Kanungo, *Performance analysis of aodv, dsr, olsr and dsdv routing protocols using ns2 simulator*, Science direct, Elsevier **30** (2012), 69–76.

[MKB07] Shruti Mahambre, Madhu Kumar, and Umesh Bellur, *A taxonomy of qos-aware adaptive event-dissemination middleware*, IEEE Internet Computing **11** (2007), 35–44.

[MKSM08] P. Macharla, R. Kumar, A.K. Sarje, and M. Misra, *A qos routing protocol for delay-sensitive applications in mobile ad hoc networks*, Communication Systems Software and Middleware and Workshops, 2008. COMSWARE 2008. 3rd International Conference on, january 2008.

[MMH04] Mirco Musolesi, Cecilia Mascolo, and Stephen Hailes, *Adapting asynchronous messaging middleware to ad hoc networking*, Proceedings of the 2nd workshop on Middleware for pervasive and ad-hoc computing (New York, NY, USA), MPAC '04, ACM, 2004, pp. 121–126.

[Nau15] Robert Nau, *Statistical forecasting : notes on regression and time series analysis*, Fuqua School of Business, Duke University, November 2015.

[Pat07] Eugster Patrick, *Type-based publish/subscribe : concepts and experiences*, ACM Transactions on Programming Languages and Systems **29** (2007), 1–50.

[PB02] Peter Pietzuch and Jean Bacon, *Hermes : a distributed event-based middleware architecture*, Proceedings of the 22nd International Conference on Distributed Computing Systems Workshops, 2002, pp. 611 – 618.

[PB14] Marco Platania and Roberto Berald, *Reliable and timely event notification for publish/subscribe services over the internet*, Journal IEEE/ACM Transactions on Networking (TON) archive **22** (2014), no. 1, 230–243.

[PEKS07] Peter Pietzuch, David Eyers, Samuel Kounev, and Brian Shand, *Towards a common api for publish/subscribe*, Proceedings of the 2007 inaugural international conference on Distributed event-based systems (New York, NY, USA), DEBS '07, ACM, 2007, pp. 152–157.

[Pet11] Magula Peter, *Quality of service in mobile ad hoc networks*, IEEE Communications Surveys and Tutorials **9** (2011), no. 2, 50–70.

[Pin09] Yu Ping, *A revised aodv protocol with qos for mobile ad hoc network*, Proceeding of the 2nd IEEE International Conference on Computer Science and Information Technology, IEEE computer Society, August 2009, pp. 241 – 244.

[RT99] Elizabeth M. Royer and Chai Keong Toh, *A review of current routing protocols for ad hoc mobile wireless networks*, Personal Communications, IEEE **6** (1999), no. 2, 46–55.

[SAB+00] Bill Segall, David Arnold, Julian Boot, Michael Henderson, and Ted Phelps, *Content based routing with elvin4*, in Proceedings of AUUG2K, 2000, pp. 55–65.

[SBK05] Bharath Sundararaman, Ugo Buy, and Ajay D. Kshemkalyani, *Clock synchronisation for wireless sensor network : A survey.*

[SFV+08] Liqi Shi, Abraham Fapojuwo, Neil Viberg, Wendy Hoople, and Norbert Chan, *Methods for calculating bandwidth, delay, and packet loss metrics in multi-hop IEEE802.11 ad hoc networks*, Proceedings of the 67th IEEE Vehicular Technology Conference,Spring 2008, 2008, pp. 103–107.

[SLM10] Felix Salfner, Maren Lenk, and Miroslaw Malek, *A survey of online failure prediction methods*, ACM Comput. Surv. **42** (2010), 10 :1–10 :42.

[SPKG12] Kaustubh Joshi Soila P. Kavulya and Felicita Di Giandomenico, *Failure diagnosis of complex systems*, ch. Resilience Assessment and Evaluation of Computing Systems, pp. 239–261, Springer Berlin Heidelberg, 2012.

[SSB99] Prasun Sinha, Raghupathy Sivakumar, and Vaduvur Bharghavan, *Cedar : a core-extraction distributed ad hoc routing algorithm*, IEEE Journal on Selected Areas in Communications **17** (1999), 1454 – 1465.

[TKKR09] M. Adnan Tariq, Boris Koldehofe, Gerald G. Koch, and Kurt Rothermel, *Providing probabilistic latency bounds for dynamic publish/subscribe systems*, In Proceeding of the 16 th ITG/GI Conference on Kommunication in Verteilten Systemen. Springer, 2009.

[Tsa08] Ruey Tsay, *Arma models*, Autumn Quarter, 2008.

[Wal12] Steven R. Walk, *Quantitative technology forecasting techniques, technological change*, 2012.

[WD09] W. Eric Wong and Vidroha Debroy, *A survey of software fault localization*, Tech. report, The University of Texas at Dallas, 2009.

[YHA10] Cheung Alex King Yeung and Jacobsen Hans-Arno, *Load balancing content-based publish/subscribe systems*, ACM Trans. Comput. Syst. **28** (2010), 9 :1–9 :55.

[YKK+09] Hao Yang, Minkyong Kim, Kyriakos Karenos, Fan Ye, and Hui Lei, *Message-oriented middleware with qos awareness*, Proceedings of the 7th International Joint Conference on Service-Oriented Computing (Berlin, Heidelberg), ICSOC-ServiceWave '09, Springer-Verlag, 2009, pp. 331–345.

[YZH07] Xiaoyu Yang, Yingwu Zhu, and Yiming Hu, *A large-scale and decentralized infrastructure for content-based publish/subscribe services*, International Conference on Parallel Processing, ICPP 2007, sept. 2007, p. 61.

[ZSEC01] Dong Zhou, Karsten Schwan, Greg Eisenhauer, and Yuan Chen, *Jecho - interactive high performance computing with java event channels*, Symposium on Parallel and Distributed Processing, 2001.

[ZW13] Yaxiong Zhao and Jie Wu, *Building a reliable and high-performance content-based publish/subscribe system*, Journal of Parallel and Distributed Computing archive **73** (2013), no. 4, 369–386.

Imene LAHYANI ABDENNADHER

Directeurs de thèse : Christophe Chassot, Khalil Drira – Université de Toulouse

Mohamed Jmaiel – Université de Sfax

Lieu et date de la soutenance : Toulouse, le 17 Décembre 2015

Titre : Gestion de la qualité de service des systèmes publier/souscrire déployés sur un réseau mobile ad-hoc

Résumé :

Les systèmes publier/souscrire déployés sur des réseaux mobiles ad-hoc présentent des défis importants en termes de qualité de service (QdS). L'objectif de cette thèse est d'assurer une gestion de la QdS des systèmes publier/souscrire sur MANET.

Les contributions de cette thèse sont organisées autour de deux grands axes relatifs aux modules de monitoring et d'analyse de la QdS de ces systèmes dans un contexte ad-hoc. Ceci englobe les étapes de collecte des paramètres de QdS au cours du fonctionnement du système, et de détection des dégradations pouvant l'affecter.

Le module d'analyse proposé offre à l'utilisateur une variété de choix entre une analyse réactive, proactive et hybride. Notre module permet aussi une localisation des pannes une fois détectées ou prédites.

Des expérimentations menées à l'aide du simulateur Jist/Swans ont montré l'efficacité des modules développés. De plus, une mesure des paramètres de performance du système avant et après introduction des modules développés a montré leur efficacité. Finalement, une étude de la complexité spatiale et temporelle du module analytique développé a été réalisée.

Mots-clés : publier/souscrire, MANET, qualité de service, monitoring, analyse.

Title : Quality of service aware publish/subscribe system on mobile ad-hoc newtork

Abstract :

Publish/subscribe systems when deployed on mobile ad hoc network present significant challenges in terms of quality of service (QoS). The goal of this thesis is to ensure QoS management of publish/subscribe systems on MANET. The contribution of this thesis is to introduce monitoring and analysis modules. This requires collecting QoS parameters during system functionning in order to detect and prevent degradations. The proposed analysis module enables the user to choose among a variety of analyses including reactive, proactive and hybrid analysis. Our approach allows also locating failures once detected or predicted.

The efficiency and accuracy of the proposed scheme were validated by performing simulation under the Jist/Swans simulator. Besides, performance parameters were studied before and after the introduction of the developed modules.

Keywords : publish/subscribe, MANET, quality of service, monitoring, analysis.

Ecoles doctorales : MITT : Domaine STIC , Réseaux , Télécoms, Systèmes et Architecture

EDST : Ecole Doctorale Sciences et Technologie

Laboratoire de recherche : LAAS - CNRS