REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE MOHAMED KHIDER - BISKRA

الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
جامعة محمد خيضر بسكرة

Faculté des sciences exactes et des sciences de la nature et de la vie
Département d'informatique

N°d'ordre:

THÈSE

Pour l'obtention de grade de

DOCTEUR 3ème CYCLE EN INFORMATIQUE

Option : Intelligence Artificielle

Titre

Gestion de contexte pour l'Internet des objets

Par

Farida RETIMA

Soutenue le : 24/02/2019

Devant le jury composé de

Président : Pr. Okba KAZAR, professeur, université de Biskra
Rapporteur : Dr. Saber BENHARZALLAH, maître de conférences A, université de Batna 2
Co-Rapporteur: Dr. Laid KAHLOUL, maître de conférences A, université de Biskra.
Examinateur : Dr. Djamel NESSAH, maître de conférences A, université de Khenchela.
Examinateur : Pr. Hammadi BENNOUI, maître de conférences A, université de Biskra

Biskra, Algérie
Année universitaire : 2018 – 2019

Remerciements

En premier lieu, je remercie le bon dieu de m'avoir donné la force et la patience nécessaire pour achever ce travail de thèse.

Je tiens à remercier vivement mon encadreur Dr. Benharzallah Saber, Maître de conférences à l'Université de Batna 2, de m'avoir accueillie au sein de son équipe pour réaliser ma thèse ainsi que pour m'aider et pour sa grande contribution et sa ténacité avec moi tout au long de cette thèse à partir de l'état d'art jusqu'à la contribution et l'implémentation. J'ai apprécié ainsi sa disponibilité, ses expériences et les nombreuses discussions qui ont animé ces années de thèse. Je le remercie pour son dynamisme qui m'a été très profitables et qui m'ont permis d'avancer tout au long de cette thèse. Je remercie également Dr. Kahloul Laid, Maître de conférences à l'Université de Biskra, pour ses conseils pertinents.

Je remercie le Pr. Okba KAZAR professeur d'université de Biskra, pour m'avoir fait l'honneur de présider mon jury de thèse, et je le remercie également pour m'avoir accueilli dans le laboratoire de LINFI et m'avoir permis d'effectuer cette thèse dans de bonnes conditions.

Je remercie également les membres du jury Dr. Djamel NESSAH, maître de conférences à l'université de Khenchela, et Dr. Hammadi BENNOUI maître de conférences à l'université de Biskra de m'avoir fait l'honneur d'accepter de participer à mon jury de thèse.

Enfin, je remercie tous ceux qui ont contribué de près ou de loin à l'aboutissement de ce travail de recherche.

ملخص

تتطلب التطورات السريعة في مجال تكنولوجيا المعلومات والاتصالات أن تقوم الأنظمة الموزعة بدمج بنية تحتية مبتكرة تلبي احتياجات المستخدمين. على وجه الخصوص, فأن ظهور نموذج إنترنت الأشياء يضع المستخدم في بيئة ذكية ومتصلة بالإنترنت. في هذه الرؤية الجديدة للإنترنت، سيتم ربط مليارات الكائنات الذكية والديناميكية بالإنترنت وستتفاعل مع بعضها البعض لتحقيق التطبيقات المشتركة.

يتطلب نموذج إنترنت الأشياء بنية تحتية مناسبة توفر السمات الرئيسية لإنترنت الأشياء وتسهل تطوير تطبيقات إنترنت الأشياء. بالإضافة إلى ذلك ، من الضروري توفير معلومات السياق للتطبيقات لتسهيل تطوير تطبيقات جديدة ، يمكن توفيرها بواسطة وسيط. إن مثل هذا الوسيط في إنترنت الأشياء، تسمى مدير السياق، تعالج كمية هائلة من البيانات غير المتجانسة الناتجة عن مصادر معلومات مختلفة، وتحولها إلى مستوى عالٍ من التجريد لتزويد المستخدمين بنتائج مفيدة.

مع الزيادة الهائلة في عدد مصادر المعلومات (مثل المستشعرات والشبكة الاجتماعية وخدمات الويب. الخ) التي تقدم نفس النوع من المعلومات لتلبية احتياجات التطبيقات. لذلك اصبح من الضروري استخدام صفة المعلومة

كمعيار أساسي لاختيار مصدر السياق الذي يضمن الجودة المطلوبة من قبل التطبيق.

في هذه الاطروحة,نهتم باختيار صفة المعلومة الموافقة لاحتياجات المستخدم. لكن، التزايد الكبير في حجم المعلومات و مصادرها و عدد معايير صفة المعلومات، يضع تحدي كبير في مجال انترنت الاشياء. من اجل حل هذا الاشكال، في هذه الاطروحة نهتم بالمساهمات التاليةّ:

- MapReduce Skyline: من اجل تسريع عملية المعالجة وتكريس مفهوم الموازاة لمعالجة العدد الكبير من المعلومات
- Logique floue: لتقييم صفة المعلومة المقدمة من طرف مصادر السياق ومن طرف التطبيقات. ان هذا المنطق الضبابي مناسب لتمثيل المعلومات الغامضة و المبهمة و الغير صحيحة و الغير دقيقة المقدمة من طرف مختلف مصادر السياق.

الكلمات المفتاحية :انترنت الاشياء، مسير السياق،صفة السياق، المنطق الضبابي، مصدر السياق،التطبيقات.

Résumé

La rapidité de l'évolution dans le domaine des Technologies de l'Information et de la Communication (TIC) nécessite aux systèmes distribués d'intégrer des middlewares innovantes répondant aux besoins d'utilisateur. En particulier, l'émergence du paradigme de l'Internet des objets (IdO) place l'utilisateur au sein d'un environnement intelligent et connecté à l'Internet. Dans cette nouvelle vision d'internet, des milliards d'objets intelligents et dynamiques seront connectés à Internet et interagissent entre eux pour réaliser des applications communes.

Le paradigme de l'Internet des objets nécessite un middleware approprié qui offre les fonctionnalités principales de l'Internet des objets pour faciliter le développement des applications IdO. En outre, il est nécessaire d'offrir les informations de contexte aux applications afin de faciliter le développement de nouvelles applications, cela peut être assuré par un middleware. Un tel middleware IdO, appelé gestionnaire de contexte, traite une énorme quantité de données hétérogènes générées par différentes sources d'information, et les transforme à un haut niveau d'abstraction pour fournir aux utilisateurs des résultats utiles.

Avec l'augmentation exponentielle du nombre de sources d'information (ex: capteurs, réseaux sociaux, services web. etc.), qui offrent le même type d'informations pour répondre aux besoins des applications. Il est devenu important d'utiliser la qualité de contexte (QoC) comme un critère essentiel pour sélectionner la source de contexte qui garantit la qualité requise par l'application.

Dans notre thèse, nous considérons le problème de sélectionner la qualité de contexte en fonction des préférences de satisfaction du consommateur. Cependant, le nombre croissant de sources de contexte et de données posent un grand défi dans IdO. Pour résoudre ce problème, notre thèse fournit les contributions suivantes selon deux axes : premièrement, nous avons proposé un Framework MapReduce Skyline pour accélérer le calcul et introduire le parallélisme pour traiter une grande quantité de sources d'information. Deuxièmement, nous avons utilisé la logique floue pour évaluer la QoC fournie par les sources de contexte et la QoC requise par l'application. La logique floue est appropriée pour représenter les informations ambiguës, imprécises, et erronées fournies par différentes sources d'information.

Mots-clés : Internet des objets, Gestionnaire de contexte, Qualité de contexte, Logique floue, Map Reduce Skyline, Source de contexte, Application.

Abstract

The rapid developments in the field of Information and Communication Technology (ICT) requires the distributed systems integrate innovative middleware that satisfy the user needs. In particular, the emergence of the Internet of Things (IoT) paradigm places the user in an intelligent environment and connected to the Internet. In this new vision of the Internet, billions of intelligent and dynamic objects will be connected to the Internet and will interact with each other to achieve common applications.

The paradigm of the Internet of Things requires an appropriate middleware that offers the main features of the Internet of Things and facilitate the development of IoT applications. In addition, it is necessary to offer context information to applications to facilitate the development of new applications, this can be provided by a middleware. Such an IoT middleware, called context manager, processes a huge amount of heterogeneous data generated by different information sources, and transforms them into a high level of abstraction to provide users with useful results.

With the exponential increase in the number of sources of information (eg sensors, social network, web services .etc.) That offer the same type of information to satisfy the needs of applications. It has become important to use context quality (QoC) as an essential criterion for selecting the context source that guarantees the quality required by the application.

In our thesis, we consider the problem of selecting the quality of context according to consumer satisfaction preferences. However, the growing numbers of context and data sources, pose a great challenge in IoT. To solve this problem, our thesis provides the following contributions along two axes: firstly, we proposed a MapReduce Skyline Framework to accelerate the computation and introduce the parallelism to treat a big quantity of sources of information. Second, we used fuzzy logic to evaluate the QoC provided by context sources and the QoC required by the application. Fuzzy logic is appropriate for representing ambiguous, imprecise, and erroneous information provided by different sources of information.

Keywords: Internet of things, Context manager, Quality of context, Fuzzy logic, Map Reduce Skyline, Context source, Application.

Liste des publications

Revues internationales

Retima Farida, Benharzallah Saber, Kahloul Laid, Kazar Okba, "A comparative analysis of context-management approaches for the internet of things", International Arab Journal of Information Technology (IAJIT), Volume 14, Number A4, May 2017. (indexed isi thomson IF=0.72).

Retima Farida, Benharzallah Saber, Kahloul Laid, Kazar Okba, "A Quality-Aware Context Information Selection Based Fuzzy Logic in IoT Environment", International Arab Journal of Information Technology (IAJIT), Vol. 15, No. 3A, Special Issue 2018. (indexed isi thomson IF=0.72)

Conférences Internationales

Retima Farida, Benharzallah Saber, Kahloul Laid ,Kazar Okba, "A comparison study of context-management approaches for the Internet of Things", 17th International Arab Conference on Information Technology, Sultan Moulay Slimane University, Beni Mellal, December 6-8, 2016, Morocco.

Retima Farida, Benharzallah Saber, Kahloul Laid ,Kazar Okba, "A quality-aware context information selection based fuzzy logic in IoT environment",18th International Arab Conference on Information Technology (ACIT'2017), Tunisia, indexed in IEEE Explore.

Sommaire

Liste des figures

Liste des tableaux

Liste des algorithmes

Chapitre 1 : Introduction

1.1 Cadre général

Le nombre croissant d'objets mobiles utilisés dans notre vie quotidienne a donné naissance à un certain nombre de nouveaux paradigmes tels que l'Internet des objets (IdO), qui repose sur une large gamme de matériels, d'infrastructures réseau, de protocoles de communication et d'applications. Un nouvel aspect comme la gestion du contexte est une exigence fondamentale pour le développement de tels systèmes. La gestion de contexte est un problème traité généralement dans le cadre de réseaux ambiants. Dans le cadre de réseaux multi-échelles (ambiant, Internet, cloud) et de l'Internet des objets, la gestion de contexte devient autrement plus complexe. Elle doit prendre en compte l'hétérogénéité des données, la répartition des traitements et des flux d'informations, le passage à l'échelle, la gestion des informations de qualité de contexte pour permettre des prises de décision appropriées, le respect de la vie privée lors de la transmission des informations de contexte, et l'adaptation à des environnements dynamiques. Ces principaux défis imposent le déploiement des composants distribués, appelés le gestionnaire de contexte. Ces composants sont chargés d'acheminer les informations de contexte depuis les sources vers les applications. Les différentes entités qui composent le gestionnaire de contexte prennent en charge : de collecter les informations de contexte provenant des différentes sources de contexte et traiter les informations collectées, puis les présenter aux différentes applications selon leurs besoins [1].

Pour permettre à l'Internet des objets d'atteindre son plein fonctionnel, plusieurs problématiques doivent être étudiées. Parmi ces défis, nous avons focalisé nos recherches au niveau de gestionnaires de contexte, afin de prendre en charge la qualité des informations (Quality of Context QoC en anglais) qu'ils traitent. Ce dernier correspond à qualifier l'information de contexte par des critères aussi divers que la précision, la complétude, exactitude…etc.

1.2. Motivation et la description de la problématique

Un grand nombre des objets dans l'IdO peuvent être connectés, à tout moment de leur cycle de vie, de manière temporaire ou permanente, au réseau mondial. Par conséquent, des énormes quantités de données sont générées dans l'environnement d'IdO. Elles peuvent être utilisées par les applications pour satisfaire leurs besoins. La solution pour répondre alors à ce problème consiste à utiliser un middleware distribué, qui est chargé de collecter, traiter et disséminer, vers les applications, les informations dont elles ont besoin. Avec le grand nombre des informations disponibles dans l'environnement d'IdO, le défi crucial consiste à sélectionner efficacement les meilleures informations en fonction des besoins des utilisateurs. À cet effet, il est important d'utiliser une méthode pour accélérer le processus de recherche et de sélection de meilleures informations.

L'IdO offre de nouvelles sources d'informations de contexte où celles-ci sont hétérogènes et ayant différentes modes de communications, de détection, stratégies de traitement et divers logiciels. En plus, les objets intelligents dans IdO ne sont pas statiques et donc sujettes à des dysfonctionnements. Comme les informations de contexte provenant de plusieurs sources de contexte (les capteurs, réseaux sociaux, des services. etc) sont intrinsèquement inexactes, erronées et ambiguës. Le risque est de prendre une décision sur la base des informations erronées, et donc le comportement des applications est fortement affecté par la qualité de contexte (QoC). Les applications sensibles au contexte attendent des données correctes et fiables pour adapter leurs fonctionnalités. En fait, les raisons de l'imperfection et l'ambiguïté de l'information de contexte issues de capteurs sont liées aux leurs caractéristiques intrinsèques dues à une résolution limitée du capteur, ou ils sont influencés par les caractéristiques extérieures de l'environnement. De même, les données produites par des capteurs

logiques, tels que les profils d'utilisateur récupérés par une base de données, peuvent présenter des informations partielles et incomplètes, ainsi que des erreurs lorsqu'elles sont produites par des techniques de raisonnement. Pour les informations provenant par les réseaux sociaux, la qualité de l'information dépend aux êtres humains qui partagent l'information. Pour les informations fournies par des logiciels ou des services, une mauvaise qualité de ces informations peut-être à cause à des dysfonctionnements et inefficacité de ces sources d'informations.

En conséquence, ces données de contexte nécessitent des mécanismes de gestion appropriés pour évaluer leurs qualités. Plus précisément, les applications sensibles à la QoC nécessitent une gestion efficace de la QoC ce qui permet à ces applications de s'adapter leurs réactions selon la QoC délivrée par le gestionnaire de contexte.

1.3 Objectif de recherche

L'objectif de cette thèse est la proposition d'une solution pour évaluer la QoC fournie par chaque source de contexte et la QoC requise par l'application au sein d'un gestionnaire de contexte. Un autre objectif est d'accélérer le calcul et d'introduire le parallélisme pour traiter de grandes quantités de sources de contexte et des données dans l'IdO.

1.4 Contribution

L'Internet des objets (IdO) a gagné beaucoup d'attention au cours de la dernière décennie. Un nouvel aspect comme la gestion du contexte est une exigence fondamentale pour le développement de tels systèmes.

Dans la littérature, il existe différentes approches permettant la gestion du contexte pour l'internet des objets. Dans la première contribution, nous avons évalué les approches de gestion de contexte les plus populaires dans l'IdO en utilisant un ensemble de critères proposés pour cet objectif. Notre contribution propose une étude exhaustive des solutions des middlewares pour la gestion de contexte basées sur différentes défis: hétérogénéité, mobilité, influence du monde physique, passage a échelles, sécurité, la protection de la vie privé, déploiement autonome des entités, caractérisation multi-échelles, interopérabilité, la qualité de contexte, acquisition de contexte, modélisation de contexte, raisonnement de contexte, distribution de contexte, méthode de conception, les outils de l'implémentation. L'ambition de cette contribution est de caractériser les solutions existantes dans la littérature en focalisant sur plusieurs aspects, ce qui permet de déterminer les points clés de nos contributions principales.

La deuxième contribution est la proposition d'un middleware basé sur la logique floue au sein d'un gestionnaire de contexte. Cette proposition vise à représenter la QoC fournie par les sources de contexte et la QoC requise par l'application sous forme des termes linguistiques. Ce middleware permet l'évaluation des valeurs incertaines de QoC mesurées, et aide à effectuer la sélection de l'information la plus proche au besoin d'utilisateur. Dans cette contribution, la logique floue a démontré qu'on peut l'utiliser comme un outil puissant dans le processus d'évaluation des attributs de QoC.

Le principal défi consiste à tenir en compte l'énorme quantité des informations générée par le déploiement des différentes sources de contexte. Par conséquent, la dernière contribution est la proposition d'un MapReduce skyline Framework pour accélérer le calcul et introduire le parallélisme pour traiter ces grandes quantités de sources de contexte. Cette dernière contribution vise à appliquer la logique floue à un petit nombre de sources de contexte résultant de MapReduce Skyline plutôt qu'un grand nombre des sources de contexte, qui existent énormément dans le paradigme de l'internet

des objets. Cette proposition permet d'améliorer l'efficacité de sélection des sources de contexte candidates pour satisfaire le besoin de l'utilisateur.

Nous avons consacré à la validation de nos contributions en présentant les résultats obtenus à partir d'une série de tests effectués en utilisant des simulations. Les résultats de l'évaluation obtenus sont satisfaisants et montrent que notre approche permet d'effectuer la gestion de la QoC avec un temps raisonnable.

1.5 Organisation de la thèse

La thèse est structurée comme suit :

Le premier chapitre présente le cadre général et la problématique de la recherche. Ce chapitre identifie également les objectifs de la thèse et présente brièvement les principales contributions.

Le deuxième chapitre est consacré à la présentation de concepts fondamentaux pour la proposition d'un gestionnaire de qualité de contexte dans le paradigme d'internet des objets.

Dans le troisième chapitre nous présenterons notre première contribution. Nous commençons ce chapitre par une définition de différents défis actuellement posés dans la littérature qui doivent être étudiés dans le paradigme de l'internet des objets. Ce chapitre introduit également les différentes approches existantes dans la littérature pour permettre la gestion du contexte dans l'IdO. Nous terminerons ce chapitre par une étude comparative exhaustive entre ces approches de gestion de contexte en se basant sur un ensemble de critères bien ciblés.

Le quatrième chapitre est consacré à nos contributions principales. Nous présenterons une méthode de sélection de l'information de contexte en fonction de sa qualité en se basant sur la logique floue et MapReduce Skyline.

Le cinquième chapitre décrit la simulation du gestionnaire de qualité de contexte proposé. Nous allons présenter dans ce chapitre les résultats d'implémentation et nous allons discuter les résultats obtenus par rapport à un ensemble de paramètres tel que le nombre des critères de QoC et le nombre des sources de contexte.

Enfin pour valider notre approche proposée nous l'avons comparé avec d'autres travaux voisins. Nous terminons cette thèse par une conclusion générale et des quelques perspectives sur des travaux futurs

Chapitre 2 : Concepts généraux

Dans ce chapitre nous résumons les concepts de base utilisés pour la proposition d'un gestionnaire de qualité de contexte dans le paradigme d'internet des objets.

2.1 L'internet des objets

2.1.1 Historique d'IdO

L'Internet des objets vise à relier des objets entre eux via un réseau. Chaque objet possède un identifiant unique. Le système d'identification par radiofréquence (RFID) est le plus utilisé dans l'environnement d'IdO.
En fait, il y a quelques développements importants qui ont contribué à la création du paradigme de l'Internet des objets :

En 1984, l'informaticien de l'Université de Tokyo, Ken Sakamura imagine le monde comme des ordinateurs seraient de plus en plus "absents", c'est à dire que ces ordinateurs deviennent invisibles dans notre environnement et sont intégrés dans les objets de la vie courante. Il appellera cette notion «ordinateur partout» (où computers everywhere). Puis il la nommera «informatique ubiquitaire». Le professeur Sakamura a donc montré que l'implantation de puces électroniques dans des objets physiques permettrait de configurer un réseau de capteurs capables de comprendre l'environnement.

Quatre années plus tard, Mark Weiser, chef scientifique de Xerox Parc a considéré l'ordinateur comme un élément important pour aider l'homme. Cette notion permet de penser à l'Internet des objets.

En 1998, le terme "intelligence ambiante" a été utilisé pour la première fois, par la société Philips. L'intelligence ambiante affirme la volonté de rendre l'informatique invisible en le fusionné dans l'environnement quotidien. Elle regroupe plusieurs idées : l'ubiquité qui se réfère à la présence dans l'environnement de multiples appareils distribués et interconnectés, la sensibilité au contexte qui correspond à collecte et au traitement d'informations concernant l'utilisateur et son environnement, et l'intelligence, qui caractérise la capacité du système à produire des inférences à partir des informations sur l'utilisateur et son environnement.

En 1998, Kevin Ashton, gestionnaire de marque chez Procter et Gamble, a utilisé la notion d'Internet des objets pour la première fois. Il souhaité d'améliorer l'efficacité de la chaîne d'approvisionnement de Procter et Gamble grâce à cette notion. Il s'est intéressé au développement d'un système universel ouvert qui permettrait de connecter des objets à l'Internet [2].

2.1.2 Définition de l'internet des objets

La notion d'Internet des Objets a été créée en 1999 par Kevin ASHTON en État Unit. On trouve à l'heure actuelle plusieurs définitions pour internet des objets ont été proposées dans la littérature. Par conséquent, il n'y a pas de définition standard pour l'IdO. Les définitions suivantes ont été fournies par différents chercheurs :

L'EPoSS (2008) définit l'origine sémantique de l'expression « internet des objets» qui est composée de deux concepts : « internet » et « objets» où l'internet peut être défini comme un réseau de réseaux informatiques interconnectés, basé sur un protocole de communication standardisé (TCP / IP). Alors que «les objets» sont des objets non identifiables. Par conséquent, «Internet des objets» signifie «un réseau mondial d'objets interconnectés adressables uniquement, basé sur des protocoles de communication standard» [3].

Vermesan (2009) exprime qu'Internet des objets (IdO) fait partie intégrante de Internet Future, y compris les développements internet et de réseaux existants. IdO pourrait être défini comme une infrastructure de réseau mondial dynamique avec des capacités de configuration automatique basées sur des protocoles de communication standards et interopérables. Les objets physiques et virtuels possèdent des identités, des attributs physiques et des personnalités virtuelles, utilisent des interfaces intelligentes et sont intégrées au réseau d'information [4].

Lu Tan (2010) définit IdO comme des objets ayant des identités et des personnalités virtuelles opérant dans des espaces intelligents utilisant des interfaces intelligentes pour se connecter et communiquer dans des contextes sociaux, environnementaux et utilisateurs. Une définition différente, qui met l'accent sur l'intégration transparente, pourrait être formulée comme «Objets interconnectés ayant un rôle actif dans ce que l'on pourrait appeler l'Internet du Futur [5].

Chen (2012) définit IdO comme un paradigme qui évolue régulièrement possédant des centaines de milliards de capteurs et de dispositifs intelligents qui interagissent entre eux sans intervention humaine. Ils généreront une énorme quantité de données, fournissant aux humains l'information en contrôlant les événements et les objets, même dans des environnements physiques distants [6].

L'Internet des objets (IdO) est le réseau de divers objets physiques intégrés à des logiciels et à divers types de capteurs. La capacité d'interconnexion et de partage de données entre eux pour obtenir plus de valeur et de services. L'intégration d'IdO dans notre vie personnelle peut certainement apporter de nombreux avantages aux individus, aux entreprises et à la société de diverses manières [7].

2.2 L'information de contexte

De nombreux chercheurs ont tenté de formaliser la signification du contexte dans l'environnement informatique. Cependant, une définition universellement acceptée doit encore être convenue.

Selon le dictionnaire d'Oxford anglaise, le mot contexte fait référence aux «circonstances qui forment le cadre d'un événement, d'une déclaration ou d'une idée».

Des définitions plus formelles et plus génériques utilisaient soit l'environnement de l'utilisateur, soit l'environnement de l'application comme une base pour établir la signification du contexte comme suit :

Schilit et al (1995) ont classé le contexte dans des catégories plus précises: physiques, informatiques et types d'informations sur le contexte d'utilisateur. Le type de contexte physique est lié à des facteurs environnementaux qui peuvent généralement être évalués en utilisant des mécanismes matériels spécialisés. La lumière, et la température sont des exemples de types de données de contexte physique. Le contexte informatique fait référence à l'information qui décrit les ressources disponibles dans l'infrastructure informatique. Cela inclut des informations telles que la connectivité réseau et ses caractéristiques (bande passante, etc.), les ressources proches (imprimantes, projecteurs vidéo, etc.) et les détails concernant la disponibilité de la mémoire, l'utilisation du processeur, etc. Enfin, le contexte de l'utilisateur fait référence au profil de l'utilisateur en se concentrant sur les besoins, les préférences, l'humeur, etc. par exemple, pour utiliser un ordinateur de bureau plutôt qu'un PDA au travail [8].

Brown (1997) voit le contexte comme des éléments de l'environnement dont le dispositif informatique de l'utilisateur est conscient [9].

Ward et al (1997) considère le contexte comme l'état de l'environnement dans lequel l'application est fonctionné. De même, Schmitd et al (1999) ont décrit le contexte comme suit: "la connaissance de l'état de l'utilisateur et du dispositif informatique, y compris l'environnement, la situation, l'emplacement". [10].

Pascoe (1998) est l'un des premiers chercheurs à avoir généralisé la notion de contexte en proposant la définition : Le contexte est un concept subjectif qui est défini par l'entité qui le perçoit. Par exemple, une entité peut concevoir son contexte comme emplacement tandis qu'un autre peut le voir dans une perspective temporelle. Par conséquent, le contexte pourrait être généralement décrit

comme le sous-ensemble des états physiques et conceptuels ayant un intérêt pour une entité particulière » [11].

Chen et Kotz (2000) ont élargi la taxonomie de Schilit et Theimer, en introduisant la classe de temps, qui représente les paramètres, tels que l'heure du jour, la semaine, le mois et la saison de l'année [12].

Dey (2001) définit le contexte comme toute information qui peut être utilisée pour caractériser la situation des entités (à savoir, une personne, un lieu ou un objet) qui sont considérées comme pertinentes pour l'interaction entre un utilisateur et une application, y compris l'utilisateur et l'application elles-mêmes. Le contexte est généralement l'emplacement, l'identité et l'état des personnes, des groupes et des objets informatiques et physiques, les conditions environnementales [13]. Cette information est dans la plupart des cas, implicite et recueillie à partir de capteurs et de plugins déployés dans l'environnement entourant l'utilisateur.

Pour Chaari et al (2005), définit le contexte comme l'ensemble des paramètres externes à l'application qui pouvant influer sur le comportement d'une application en définissant de nouvelles vues sur ses données et ses services. Ces paramètres ont un aspect dynamique qui leur permet d'évoluer durant le temps d'exécution. Ils ne sont pas « parlants » à l'utilisateur final et doivent donc être transparents. Une nouvelle instance de ces paramètres caractérise une nouvelle situation contextuelle qui ne modifie pas les données de l'application mais qui peut mener à les traiter d'une façon différente [14].

Dans le cadre de notre thèse nous nous sommes intéressés à la définition proposée par l'auteur Dey en 2001.

2.3 La qualité de contexte

La QoC a reçu beaucoup d'attention ces dernières années et plusieurs définitions ont été émergées dans la littérature :

La QoC c'est toute information décrivant la qualité d'une information de contexte. Ainsi, QoC se réfère à l'information et non au processus ou au composant matériel qui fournit éventuellement l'information [15].

Une autre définition considère la QoC comme l'ensemble des paramètres utiles pour exprimer les propriétés et les exigences de qualité sur les données de contexte, par exemple, précision, up-to-dateness, trust-worthiness, etc. En d'autres termes, La QoC ne consiste pas à avoir des données de contexte parfaites, telles que des données sans erreurs, mais à avoir une caractérisation correcte de la qualité des données [16].

QoC a également été défini dans [17] comme "toute information qui décrit les informations contextuelles et peut être utilisée pour déterminer la valeur de l'information pour une application spécifique"[18].

L'une des caractéristiques spécifiques du contexte est son imperfection, car il repose souvent sur des propriétés d'entités du monde réel. La qualité du contexte peut varier considérablement en fonction de la source de données. Les capteurs sont sujets à des défaillances et, par conséquent, les données contextuelles peuvent souvent être incorrectes, incohérentes et incomplètes. Les applications sensibles au contexte doivent tenir compte de ces inexactitudes et incertitudes. Les indicateurs de qualité de l'information visent à permettre aux applications de spécifier leurs exigences en termes de qualité des données.

Dey (2000) a proposé un modèle de qualité de l'information, qui comprend les paramètres suivants: accuracy, reliability, coverage, resolution, frequency, and timeliness. Ebling et al ont défini deux indicateurs de qualité de l'information. L'un est la freshness, qui indique quand la valeur du contexte a

été mise à jour pour la dernière fois et l'autre est la confidence, qui décrit dans quelle mesure la valeur est exacte [12].

Manzoor et al (2008) a classé la QoC en sources et paramètres. Les sources de QoC sont les quantités qui sont détectées à partir de l'environnement. Mais comme ces valeurs ne sont pas appropriées pour les utiliser par une application, elles sont transformées en paramètres de QoC de niveau supérieur qui sont plus appropriée pour une utilisation computationnelle. Les paramètres de QoC sont dérivés de sources de QoC et sont représentés sous une forme, qui peut être, utilisée par une application. Les paramètres de QoC génériques sont les paramètres requis par la plupart des applications, tels que la precision, trust, validity, representation consistency, and completeness. Les paramètres de QoC spécifiques à un domaine sont les paramètres importants pour certains domaines d'application spécifiques, tels que la significance and access security [18].

De nombreux paramètres ont été ainsi proposés dans la littérature. Les paramètres de QoC les plus importants sont les suivants :

- Reliability : Manzoor (2010) définit la Reliability pour indiquer dans quelle mesure le contexte peut être considéré comme crédible [19].
- Precision : Buchholz et al (2003) définit la précision pour décrire exactement comment les informations de contexte fournies reflètent la réalité [15].
- Accuracy : Kim and Lee (2006) définit accuracy comme la probabilité de correction, ce qui signifie que les données sont précises et fiables : la probabilité qu'une partie de l'information de contexte soit correcte [20].
- Up-to-dateness: Buchholz et al (2003) décrit Up-to-dateness comme l'âge de l'information de contexte. En général, Up-to-dateness est spécifiée en ajoutant un timestamp aux informations de contexte [15].
- Resolution : Carr and Brown (2001) décrit la résolution pour indiquer le degré de détail avec lequel un capteur peut collecter les données où le plus petit changement détectable de l'environnement pouvant être détecté dans la sortie du capteur [19].
- Coverage : Dey (2000) définit ce paramètre comme l'ensemble de toutes les valeurs possibles pour un attribut de contexte [21]
- Significance : Manzoor A et al (2008) décrit cette mesure de qualité pour indiquer la valeur de l'information de contexte dans une situation spécifique. Sa valeur est particulièrement importante dans les scénarios impliquant des situations mettant en danger la vie de l'homme [18].
- Completeness : Kim & Lee (2006) définit ce paramètre comme la mesure dans laquelle l'information de contexte est disponible, suffisante et non absente c.-à-d, Tous les aspects du phénomène dans l'environnement ont été montrés [22].
- Access Right : Manzoor (2010) décrit le paramètre access right pour indiquer dans quelle mesure le propriétaire du contexte permet au consommateur final d'accéder au contexte [19].
- Integrity : fait référence à la crédibilité et à la fiabilité de la source du contexte [20].
- Probability of correctness : Buchholz et al (2003) définit ce paramètre comme la probabilité qu'une information de contexte soit correcte [20].
- Trustworthiness: Buchholz et al (2003) décrit Trustworthiness comme la probabilité que les informations fournies soient correctes. En autre terme, ce paramètre est utilisé par un producteur de contexte pour évaluer la qualité de l'autre entité qui a produit l'origine de l'information de contexte [15].
- Representation Consistency : Manzoor et al (2010) décrit ce critère comme la mesure dans laquelle le format de représentation du contexte est conforme aux formats de représentation pour le besoin du consommateur de contexte. En autre terme, ce critère dépend de la quantité d'effort nécessaire

pour transformer l'information de contexte selon le modèle de données présenté par le consommateur de contexte [19].

- Granularity : Manzoor et al (2010) désigne ce critère comme degré de détail de l'information de contexte [19].
- Usability : Manzoor et al (2010) désigne à quel niveau cette information de contexte est appropriée à utiliser pour un usage prévu. Ce paramètre considère le niveau de granularité de l'information de contexte collecté et le niveau de granularité requis par un consommateur de contexte [19].

2.4 Gestionnaire de contexte

Un gestionnaire de contexte est une entité logicielle responsable de la collecte des données produites par les différents objets répartis autour de l'IdO. Puis, la gestion (fusion, agrégation, interprétation, inférence) et de la présentation des informations de contexte aux applications sensible au contexte. Ces applications sont classiquement nommées en tant que consommateurs de données de contexte finaux, tandis que les entités fournissant des données brutes sont considérées comme des producteurs. Un composant dans un gestionnaire de contexte qui opère la transformation de données de contexte est considéré à la fois comme un consommateur de données de contexte intermédiaire et comme un producteur.

La différence essentielle entre les gestionnaires de contexte dans IdO et ceux dans l'environnement ambiant est qu'un couplage direct et fort entre un fournisseur de contexte et un consommateur de données final n'existe plus dans IdO. Les gestionnaires de contexte dans l'IdO fonctionnent sur des dispositifs distants et ils n'ont pas besoin de chacun à connaître l'autre. En d'autres termes, étendre le cadre des gestionnaires de contexte de l'environnement ambiant local à l'Internet des Objets (IdO) introduit un découplage spatio-temporel entre les producteurs et les consommateurs de contexte [23, 24].

Dans l'environnement ambiant où les entités de gestionnaire de contexte (les sources et les applications) sont connues à l'avance et développées par la même équipe des développeurs. Or cela n'est plus envisageable pour l'internet des objets. Les sources, les applications et les entités de traitement dans IdO sont conçues et développées par différentes équipes qui sont temporellement, spatialement et administrativement distribuées.

2.5 Application sensible au contexte

La recherche sur le développement d'applications sensible au contexte a attiré beaucoup d'attention depuis 1990.

La première définition des applications sensibles au contexte a été donnée par Schilit et Theimer 1994 [25] où ils ont limité la définition des applications qui sont simplement informées du contexte aux applications qui s'adaptent au contexte. Ils ont proposé un système d'active map service, fournissant des informations à un utilisateur en fonction de sa localisation.

De nombreux chercheurs définissent les applications sensibles au contexte comme des applications qui modifient ou adaptent dynamiquement leurs comportements en fonction du contexte de l'application et de l'utilisateur. Plus précisément :

Abowd et al. (1999) définit un système comme un système sensible au contexte s'il utilise un contexte pour fournir des informations et / ou des services pertinents à l'utilisateur, lorsque la pertinence dépend de la tâche de l'utilisateur [26].

Ryan 1998 [21] définit les applications sensibles au contexte comme des applications qui surveillent les données provenant de capteurs environnementaux et permettent aux utilisateurs de

sélectionner parmi une gamme de contextes physiques et logiques en fonction de leurs intérêts ou activités actuels [27].

Pascoe (1998) a présenté un ensemble de capacités génériques de base pour un système sensible au contexte :

- Détection contextuelle : La détection contextuelle est le niveau le plus fondamental de la sensibilité du contexte. Un système détecte simplement divers états environnementaux et les présenté à l'utilisateur.
- Adaptation contextuelle : Ce n'est pas seulement l'utilisateur qui peut être intéressé par les données de contexte d'un tel système. Les applications peuvent tirer parti de ces connaissances contextuelles en adaptant leur comportement pour s'intégrer plus facilement dans l'environnement de l'utilisateur.
- Découverte de ressources contextuelles : vise à découvrir d'autres ressources dans le même contexte et exploiter ces ressources tant qu'elles restent dans le même contexte.
- Augmentation contextuelle : consiste a enrichi l'environnement avec des informations en fonction d'un contexte précis [11].

Brown (1996) définit les applications contextuelles comme des applications qui fournissent automatiquement des informations et effectuent des actions en fonction du contexte actuel de l'utilisateur détecté par les capteurs [28].

Fickas et al définissent des applications sensibles au contexte comme des applications qui surveillent les changements dans l'environnement et adaptent leur fonctionnement selon des directives prédéfinies ou définies par l'utilisateur [29].

Fournir ces applications avec la capacité d'identifier et de comprendre le contexte de leur interaction avec les utilisateurs peut grandement améliorer la communication entre les utilisateurs et les machines. Le contexte apparaît comme une clé fondamentale permettant aux systèmes de rendre les informations disponibles en informations pertinentes. L'application idéale doit être en mesure de fournir des informations à la fois exactes et pertinentes sans que l'utilisateur ait besoin de rechercher activement cette information et de déterminer sa pertinence. Plus généralement, les systèmes sensibles au contexte ont l'avantage de s'adapter dynamiquement aux événements qui produisent dans le système et ses environs. L'une des principales caractéristiques de ces systèmes est d'ajuster le comportement du système sans intervention de l'utilisateur [30].

L'architecture logicielle pour les systèmes contextuels doit avoir la capacité de percevoir le changement de l'environnement externe ou d'état interne, de comprendre le contexte actuel et de fournir des services au bon endroit au bon moment. Par conséquent, la prise en charge du déploiement de divers périphériques, la manipulation d'informations de contexte et la découverte de services sont des considérations de base lors de la conception d'un système sensible au contexte [31].

En général, trois approches typiques ont été très utiles pour développer des applications sensibles au contexte:

(1) Chaque application interagit, obtient, traite et utilise le contexte de son intérêt à sa manière ;

(2) Certaines bibliothèques / toolkits visant à acquérir et à traiter le contexte peuvent être ajoutées et réutilisées pour créer des applications sensibles au contexte ;

(3) Les applications sont construites sur la base d'un middleware sensible au contexte dans lequel la gestion de contexte est activée [32].

2.6 Conclusion

Dans ce chapitre nous avons survolés les principaux concepts et les points clés de l'environnement de l'internet des objets. Nous avons commencé par présenter le contexte historique de l'IdO et sa définition. Ensuite nous avons défini quelques concepts tel que l'information de contexte,

la qualité de contexte, l'application sensible au contexte, et un gestionnaire de contexte, qui ont un impact sur le paradigme d'IdO. Dans le prochain chapitre nous commencerons à explorer le sujet de la thèse en présentant les différentes approches proposées pour la gestion de contexte dans l'internet des objets avec une étude comparative entres telles approches.

Chapitre 3 : Les différentes approches proposées pour la gestion de contexte dans l'internet des objets

3.1 Introduction

Nous évoluons vers un nouveau paradigme de l'internet : l'Internet des objets (IdO), où le nombre des objets intelligents sont interconnectés pour générer des données volumineuses et les transmettre sur internet pour atteindre beaucoup de services. Pour gérer l'énorme quantité d'informations, on a besoin d'un Middleware pour la gestion de l'information de contexte. Ce Middleware offre un service de collecte, de traitement et de présentation de l'information de contexte aux applications. Avec la naissance des middlewares de gestion de contexte, de nombreux problèmes sont apparus et actuellement posés dans la littérature qui doit être étudiés. Parmi ces défis, nous nous intéressons aux suivants : l'hétérogénéité des objets, la mobilité des objets, l'influence du monde physique, le passage à échelle, la sécurité, la protection de la vie privée, le déploiement autonome des entités, la caractérisation multi-échelles, l'interopérabilité, la qualité de contexte, l'acquisition de contexte, la modélisation de contexte, le raisonnement sur le contexte, la distribution de contexte, etc. Ces critères ont une influence directe sur le développement d'un gestionnaire de contexte dans l'IdO.

Dans la littérature, différentes approches ont été proposée pour permettre la gestion du contexte dans l'IdO. Parmi ces approches les plus connus sont : INCOME, SITAC, FIWARE, CAIoT, SAMURAI, HYDRA, AURA, SAI, and WISeMid. Plusieurs défis ont été détectés dans le développement des futures architectures et plus d'efforts devraient être pris en compte pour conduire les propositions de middlewares actuels vers le meilleur. Ce chapitre commence par décrire les différentes problématiques posées dans l'IdO. Nous mettons ensuite une revue des dernières solutions pour la gestion de contexte dans l'IdO. Nous terminons ce chapitre par une étude comparative exhaustive entre les différentes approches présentées basée sur un ensemble de critères bien ciblés.

3.2 Gestionnaire de contexte pour les environnements ambiants

Ces dernières années, la gestion de contexte pour les environnements ambiants suscite l'intérêt d'une communauté de recherche de plus en plus importante. Un gestionnaire de contexte est une entité logicielle utilisée pour gérer les informations de contexte provenant par divers fournisseurs de contextes et traite les demandes des consommateurs de contexte. Le système de gestion de contexte est divisé en trois parties : un ensemble de sources de contexte fournit des informations de contexte. Ces informations sont extraites de capteurs, d'applications locales ou de services externes. Le gestionnaire de contexte agrège ces informations par l'utilisation des méthodes de raisonnement, d'inférence et d'interprétation. Le consommateur de contexte pose sa requête au gestionnaire de contexte en spécifiant les propriétés de l'information qu'il a besoin.

Les gestionnaires de contexte dans les environnements ambiants ne peuvent pas être appliqués pour l'Internet des Objets à cause de nouvelles contraintes apportées par l'IdO qui sont : les sources d'informations et applications très mobiles et hétérogènes, les producteurs d'informations et les applications sont incohérents et pourtant ils doivent échanger des données entre eux, afflux massif de données, besoin de traitement en temps réel des données, prise en compte de l'autonomie limitée des entités de l'Internet des Objets. Par conséquent, les travaux existants dans la littérature dans le paradigme d'IdO visent à étendre le cadre des gestionnaires de contexte de l'environnement ambiant local à l'Internet des Objets (IdO). Pour ce but, un découplage spatio-temporel entre les producteurs et les consommateurs de contexte est nécessaire.

3.3 Problématiques posées par l'Internet des objets

3.3.1 Hétérogénéité de l'Internet des objets

Les propriétés et les capacités des objets varient significativement, contribuant à faire de l'Internet des objets un écosystème certes riche, et aussi très hétérogène. L'IdO est composé d'objets très différents se situent à plusieurs niveaux : matériels, logiciels et protocoles de communication. L'hétérogénéité du système peut venir de différences de protocoles et des réseaux impliqués, de différences d'espace de stockage ou de nature des équipements, ou de variations de la dispersion des entités. Au sein d'un même réseau, les postes de travail dotés de capacités de calcul variés et d'un stockage important peuvent coexister avec d'autres périphériques disposant de ressources limitées. Cette variation nécessite que le middleware de l'IdO ait un niveau d'abstraction plus élevé approprié afin de masquer l'hétérogénéité des modules et la complexité de communication [33].

L'hétérogénéité apparaît à deux niveaux : l'hétérogénéité sémantique des objets, et l'hétérogénéité technologique qui demeure une première difficulté à surmonter pour leur permettre d'interagir [34]:

- L'hétérogénéité sémantique des objets et des dispositifs : ce type d'hétérogénéité est plus difficile à cause de l'indisponibilité de toutes les entités logicielles lors de la conception. Ce qui contraint de devoir les intégrer dès qu'ils soient disponibles et leurs sémantiques doit être connue. Dans ce cadre, pour résoudre ce problème on doit fournir un mécanisme permettant de connaître la sémantique de différentes entités.
- Hétérogénéité technologique : Les technologies matérielles et logicielles utilisées pour construire les objets sont multiples qui compromettent l'idéal de collaboration autonome entre objets. Afin de rendre les entités technologiquement hétérogènes interopérables, l'utilisation des mécanismes de communication standards est impérativement nécessaire. Ce type d'hétérogénéité nécessite des middlewares distribués sur des serveurs, des stations de travail et des appareils mobiles portables.

3.3.2 Influence du monde physique sur l'Internet des objets

Le monde physique est un environnement qui évolue naturellement au cours du temps. Dans le paradigme d'IdO, les objets connectés à Internet sont beaucoup plus qu'une simple connectivité ou un message à transporter. En d'autres termes, l'IdO est ouvert, ce qui signifie que tout le monde peut ajouter et supprimer des objets. La capacité de détection des smartphones et la mobilité des utilisateurs influent sur le monde physique de l'IdO qui nécessite que les développeurs prennent en compte un mécanisme de représentation et de traitement des données. Par conséquent, l'internet des objets est fondamentalement influencé par les caractéristiques du monde physique et des outils qui permettent de le mesurer. Dans ce cas, il s'agira d'adapter les applications aux évolutions du monde qui les entourent.

Nous pouvons diviser cette problématique en trois sous-critères:

- Techniques spécifiques de détection, de correction ou d'atténuation d'erreur: les grands volumes de données issus d'un grand nombre de sources différentes, posent un problème en ce qui concerne leurs mesures qui sont erronées, imprécises ou incomplètes. Il s'agit donc un mécanisme de détection de ces erreurs et leur correction continu en temps réel pour que les applications s'adapter aux changements qui surviennent au cours du temps de manière pertinent.
- La capacité des objets mobiles à s'adapter aux changements : la mobilité des objets induisent une topologie dynamique du réseau donc ils doivent être en mesure de s'adapter dynamiquement aux changements qui surviennent au cours du temps.
- La capacité des objets mobiles à gérer leurs limites énergétiques : Les applications d'IdO nécessitant un grand nombre des dispositifs qui sont souvent difficiles à mettre en œuvre en

raison des contraintes de temps, de mémoire, de traitement et d'énergie. Le défi d'énergie est largement disponible car les objets sont connectés au réseau. Un aspect de l'IdO va donc consister à réaliser la connectivité internet avec des solutions techniques très basse énergie pour fonctionner avec une simple pile, ou en récupérant uniquement l'énergie dans l'environnement [35].

3.3.3 Sécurité

L'internet des objets (IdO) est une intégration de plusieurs réseaux hétérogènes, il devrait traiter des problèmes de compatibilité entre différents réseaux qui est sujette à des problèmes de sécurité**.** La sécurité représente un élément essentiel pour permettre l'adoption généralisée des technologies et des applications d'IdO. Les applications de l'IdO peuvent améliorer la vie quotidienne des personnes. Mais, si elles ne peuvent pas assurer la sécurité des informations et de protocole de communication, ces informations peuvent être divulguées à tout moment. Donc, la sécurité de l'IdO ne peut être ignorée. Avec la diffusion volumineuse de l'IdO qui fournira plus vaste riche d'information, le risque d'exposition de ces informations augmentera. Si l'IdO ne peut pas avoir une bonne solution pour les problèmes de sécurité, cela limitera grandement son développement. L'Internet des objets souffre de plusieurs vulnérabilités inhérentes aux technologies utilisées. En effet, de nombreux objets possèdent des capacités matérielles réduites, et de fait, ne peuvent pas directement mettre en œuvre des techniques de sécurisation modernes (technologie de cryptage, partage de clé.etc). Ces dernières nécessitant des ressources de calcul conséquentes. De la même façon, le coût énergétique élevé des communications sans fils et la puissance de calcul complexifie les processus d'authentification des objets [33].
La sécurité peut être mise en œuvre de deux façons:

- une communication de haut niveau sécurisée entre les pairs de réseau qui permet à la couche supérieure de communiquer entre ses pairs de manière sécurisée et abstraite.
- la gestion sécurisée de la topologie qui traite l'authentification des nouveaux pairs, des autorisations pour accéder au réseau et à la protection des informations de routage échangées dans le réseau [36].

La sécurité concerne la protection contre les accès non autorisés à une infrastructure, il faut s'assurer que les ressources matérielles et/ou logicielles d'un middleware IdO sont utilisées dans le cadre prévu. Le bloc fonctionnel de sécurité permet de sécuriser le système d'IdO en fournissant des fonctions telles que l'authentification, l'autorisation, la confidentialité, l'intégrité, la sécurité des données, et de la non-répudiation : La confidentialité indiquant la garantie que seules les entités autorisées peuvent accéder aux données et les modifier comme l'utilisation des techniques de cryptage. L'intégrité garantit que les informations ne sont pas modifiées ou corrompues [37,38].

3.3.4 Protection de la vie privée

Dans la vision d'IdO, les technologies de communication sans fil joueront un rôle important. L'adoption omniprésente du support sans fil pour l'échange de données peut poser un nouveau problème en matière de violation de la vie privée. En fait, le canal sans fil augmente le risque de violation en raison des capacités d'accès à distance, ce qui expose potentiellement le système à des attaques d'écoute. La vie privée représente donc un véritable problème qui peut limiter le développement de l'IdO. Pour assurer la protection de la vie privée dans le paradigme d'IdO, il faut définit un ensemble de règles et de protocoles qui définissent quels dispositifs ou quelles applications ont le droit d'accéder aux informations de contexte [34].
La protection de la vie privée doit être traitée à différents niveaux:

- Au niveau le plus bas, la couche matérielle doit assurer la sécurité et la confidentialité pendant la collecte et le stockage temporaire dans l'appareil.
- Au niveau communication ou les protocoles sécurisés doivent s'assurer que la communication est bien protégée.
- Au niveau application : Une fois les données reçues, la protection au niveau de l'application doit être en place pour surveiller et contrôler qui peut voir ou utiliser ces données [5].

3.3.5 La mobilité

Une vision cohérente et stable de la topologie du réseau est un facteur important pour la gestion des applications d'IdO. Cependant, la mobilité liée à tout (objets, utilisateur.etc) rend cette cohérence difficile et inefficace [39]. Parce que les objets intelligents dans l'IdO ne sont pas statiques et changent leur emplacement physique, cela entraîne des changements rapides dans la topologie du réseau. Ce changement à son tour affecte l'efficacité du routage car de nombreux chemins deviennent indisponibles. La mise à jour des informations de routage après la mobilité des objets peut entraîner des coûts généraux importants, notamment lorsque de nombreux nœuds ont changé leur emplacement physique simultanément. De nombreuses applications de contexte sont également mobiles et dépendent des sources d'informations mobiles. Donc il faut un mécanisme pour que le provisionnement de l'information de contexte doive être adaptable à l'environnement changeant [40].

3.3.6 Qualité de contexte

Dans l'Internet des objets (IdO), l'un des nombreux facteurs importants est la sensibilité de contexte. Mais les informations de contexte peuvent ne pas être fiables ou utiles, devenant un problème en termes de qualité. Par conséquent, un aspect important est que ces informations de contexte doivent être fiables et la qualité doit être assurée. Dans ce nouvel paradigme, les données collectées à partir d'un déploiement mondial d'objets intelligents constituent la base pour prendre des décisions intelligentes et fournir des services aux utilisateurs.

Les limites des différentes sources d'information, la qualité de la communication et du traitement des données, et la situation d'une mesure spécifique peuvent affecter sur la qualité des informations de contexte collectées. Les applications d'IdO doivent traiter d'énormes volumes de données, avec une grande vélocité, dynamicité et une grande variété de types de données. De plus, divers applications ont des exigences différentes qui détermineront l'efficacité de l'utilisation des données. Si les données sont de mauvaise qualité ou ils ne sont pas facilement utilisé et compris par l'utilisateur, les décisions risquent d'être non pertinentes et mène vers une dégradation des performances du système sensible au contexte [41].

Les sources de contexte dans l'IdO peuvent également avoir des capacités différentes pour observer l'environnement qui offrent le même type d'information avec des différentes qualités fonctionnelles et non fonctionnelles. L'un des principaux défis auxquels sont confrontés les développeurs pour réaliser la vision d'infrastructure d'IdO est de sélectionner la source de contexte le plus approprié disponible qui a une meilleur qualité et adéquate au besoin de l'application. La gestion de la QoC au sein des gestionnaires de contexte doit donc s'effectuer tout le long du cycle de vie des informations, depuis leur collecte jusqu'à leur acheminement vers les applications.

Par conséquent, on peut diviser ce problème en des sous-problèmes comme suit:

- QoC pour les informations de contexte de la phase d'acquisition : évaluer la qualité des informations produites au niveau des sources d'informations peut être réalisée par analyser les caractéristiques des sources de contexte.

- QoC pour les informations de contexte de la phase de traitement : dans ce niveau, évaluer la qualité de contexte consiste à augmenter le niveau d'abstraction des informations de QoC. Prend en compte l'évaluation dans ce niveau permet d'éliminer les informations de contexte qui ont la QoC est mal et faible et donc augmenter la performance dans le processus de traitement des informations.
- QoC pour les informations de contexte de la phase de présentation : dans ce niveau, l'information de contexte avec sa qualité sont présentés aux applications selon leurs besoins pour suggérer de bonnes décisions aux utilisateurs finaux [42].

3.3.7 Passage à l'échelle

L'évolution rapide de la technologie d'IdO définit de nouveaux défis d'architecture en termes de passage à l'échelle. Ce nouvel paradigme est composé d'un nombre croissant d'objets intelligents, ces objets communiquent ensemble. Le middleware proposé devrait conserver sa stabilité et devrait avoir la capacité de gérer efficacement le nombre croissant d'objets et d'exécuter leurs fonctions avec l'efficacité requise, bien que le nombre d'appareils connectés augmente et varie d'un endroit à l'autre. Dans ce cas, une architecture décentralisée est requise qui distribue les informations provenant de différents d'objets d'IdO afin de ne pas se retrouver dans une situation de goulot d'étranglement [43].

Les différents problèmes liés au problème de passage à l'échelle sont résumés comme suit :

- Adressage et nommage : En raison de la croissance rapide des appareils d'IdO, cela implique un espace d'adressage important pour identifier ces objets et la découverte de ces derniers.
- Communication de données et réseaux : En raison du haut niveau d'interconnexion entre de nombreuses entités qui nécessitent un réseau mondial d'objets uniquement adressable basé sur des protocoles de communication standard.
- La gestion de l'information et des connaissances : différents types d'objets hétérogènes génèrent des données en fonction de leurs lectures ou de la communication avec d'autres objets. Une énorme quantité de données continuera à générer à partir de différentes parties du monde d'IdO et cela nécessite une bonne gestion du stockage et de traitement pour faire face à un tel volume de données.
- Gestion des services: en raison du nombre important de services qui pouvant être disponibles dans l'IdO, cela implique une bonne gestion des ressources hétérogènes [38].

3.3.8 Caractérisation multi-échelles

Une caractérisation des scénarios multi-échelle vise à cibler spécifiquement les exigences de conception du gestionnaire de contexte distribué, y compris les exigences pour multi-échelle. L'objectif principal est de concevoir et mettre en œuvre un middleware de gestion de contexte multi-échelle pour l'IdO. Le processus de caractérisation multi-échelle permet de proposer une vision simplifiée d'un système réparti complexe.

Sam Rottenberg (2015) propose un Framework de caractérisation multi-échelle d'IdO appelé MuSCa. Ce Framework inclut un processus de caractérisation fondé sur les concepts de points de vue, dimensions et échelles. Il vise à décrire pour chaque système complexe étudié, ses caractéristiques multi-échelles. La fonctionnalité multi-échelle d'un système dans ce Framework peut être considérée en fonction de plusieurs vues comme les réseaux utilisés, les matériels déployés, les informations échangées, et les utilisateurs. Pour chaque vue, on peut sélectionner une ou plusieurs dimensions. Chaque dimension est associée à une mesure numérique ou sémantique. Les échelles sont des

intervalles de valeurs de cette mesure. Par conséquent, chaque système peut conduire à une caractérisation multi-échelle spéciale [44].

3.3.9 Déploiement autonome des entités

Le déploiement est un processus complexe qui a pour objectif la mise à disposition puis le maintien en condition opérationnelle d'un système logiciel. Le déploiement implique deux catégories de sites: un site producteur qui héberge des composants logiciels et des procédures d'installation, et un site consommateur qui est la cible du déploiement, sur lequel un élément logiciel doit être exécuté. Tous ces sites sont des domaines de déploiement nécessitant l'écriture de la distribution des composants, ce que l'on appelle un plan de déploiement (association entre les composants logiciels et les appareils).

Dans le paradigme d'IdO, un gestionnaire de contexte est un élément essentiel pour traiter les informations de contexte. Ce gestionnaire de contexte est un système de composants répartis qui doivent être déployés sur différents appareils : des appareils pour l'interaction avec l'utilisateur, des appareils (serveurs) pour traiter les informations collecté, les appareils pour produire et collecte les informations.

Les solutions traditionnelles de déploiement consistent à utiliser le mode centralisé dans lequel un opérateur humain réalise les différentes tâches du déploiement. Ces solutions utilisent dans la majorité des cas pour les environnements qui sont limités et leurs composants sont connus à l'avance. Par conséquent, ces solutions de déploiement sont globalement inadaptées et impossible à réaliser dans l'environnement d'IdO du fait de l'hétérogénéité, de la dynamique, de l'ouverture et de la décentralisation de ce système. Ainsi, dans l'IdO, le réseau de machines sur lesquelles on fait le déploiement n'est pas connu à l'avance du fait de l'apparition et la disparition d'appareils, ce qui rend la tâche du déploiement difficile à réaliser [45].

3.3.10 Interopérabilité

L'Internet des Objets est constitué de dispositifs hétérogènes qui interagissent et collaborent entre eux pour réaliser une tâche commune et pour échanger les informations. Comme un nombre croissant d'objets sont ajoutés à l'internet des objets, l'interaction de ces composants ou plusieurs systèmes pour échanger les informations devient nécessaire à réaliser. Le middleware qui sera utilisé dans tels cas devrait être aussi interopérable que possible afin qu'il puisse supporter les objets hétérogènes existants ainsi que d'autres nouveaux objets intelligents qui peuvent se produire dans le futur.
Deux types d'interopérabilité possibles peuvent être atteints :

- D'un côté, les composants internes peuvent interagir entre eux et partager les informations.
- D'un autre côté, différents middlewares peuvent communiquer entre eux et utilisent les informations échangées ce qu'on appelle l'interopérabilité externe [32].

3.3.11 Cycle de vie du contexte

Cycle de vie du contexte se compose de quatre phases :

1. Acquisition de contexte : le contexte doit être acquis de diverses sources à travers la détection, l'observation ou la mesure de certains faits. Le but de l'acquisition de contexte est d'obtenir un maximum de données, de sorte que les possibilités d'applications intelligentes puissent être maximisées grâce à des informations de contexte plus riches. Nous discutons trois facteurs:
   - Technique d'acquisition : la technique utilisée pour collecter les informations provenant de différentes sources de contexte : soit par un middleware : Les applications peuvent récupérer les

données des capteurs par un middleware et non par le dispositif capteur directement, ou les acquérir directement par le dispositif capteur.

- Support de source de contexte : Il y a différentes sources qui sont capables de fournir les informations de contexte. Ce critère permet d'indiquer les sortes de sources de données (soit : capteurs physiques, capteurs virtuels ou capteurs logiques)
- Découverte de contexte : Une fois que les capteurs sont connectés à une solution logicielle, on a besoin d'avoir une méthode pour comprendre automatiquement les données produites par les capteurs et les données connexe. Ce critère est une tâche difficile à cause de la variation considérable dans les domaines d'application en IdO [32,5].

2. Modélisation de contexte : La modélisation de contexte est aussi largement appelée représentation de contexte. Lors de la modélisation des informations, il faut prendre en considération l'hétérogénéité, la mobilité, les relations et les dépendances entre les différents types d'informations, la fraîcheur et l'imperfection des données et la facilité d'utilisation des formalismes de modélisation. Toutes ces problématiques compliquent la phase de la modélisation de contexte dans l'IdO. Il existe plusieurs techniques de modélisation de contexte populaire utilisé dans l'informatique sensible au contexte [5].
3. Raisonnement de contexte: les informations de contexte doit être traitées pour dériver des informations de contexte de haut niveau à partir de données de sources de contexte brutes de bas niveau. Cette méthode vise à déduire de nouvelles connaissances. Le raisonnement de l'information confronte des difficultés à cause de l'imperfection et l'incertitude des données provenant de différentes sources de contexte. Il y a un grand nombre de techniques pour le raisonnement de contexte utilisés dans la littérature. Dans le paradigme IdO, le contexte peut ne pas être précis et complet. Par conséquent, dans la phase de raisonnement, les nouvelles informations de contexte doivent être définies en termes d'attributs de la qualité de contexte. Les problèmes de la technologie de différentes sources de contexte et les problèmes de techniques de raisonnement contribuent de façon remarquable aux conflits de contexte [5].
4. Distribution de contexte :
   Elle est responsable de la diffusion des informations de contexte utiles aux applications correspondantes. Dans IdO, la distribution d'énormes quantités de données aux applications nécessite des protocoles de distribution de données avec des délais de livraison maximum et une différenciation du trafic. Cette phase est basée sur un ensemble des méthodes pour délivrer les données aux consommateurs de contexte [32].

3.4 Les différentes approches proposées pour la gestion du contexte dans l'IdO

Dans la littérature, de nombreuses approches existent permettant la gestion du contexte dans l'Internet des objets. Dans cette section nous allons présenter les approches les plus représentatives et populaires, et ainsi on les étudie exhaustivement en utilisant nos critères.

3.4.1 Approche INCOME

3.4.1.1 Description

Le paradigme d'IdO ouvre la voie à une augmentation exponentielle du nombre d'objets connectables nécessitant de nouvelles solutions capables de faire face à ce changement d'échelle. INCOME [46] propose de concevoir des solutions de gestion de contexte pour adresser non seulement les réseaux ambiants mais aussi l'IdO et le cloud dans un cadre multi-échelle. Ces solutions capables de fonctionner à différentes échelles et de faire face au passage d'une échelle à l'autre. Le projet INCOME, financé par l'Agence Nationale de la Recherche (ANR) de février 2012 à octobre 2015.

INCOME vise à offrir une infrastructure de gestion de contextes Multi-échelles pour l'Internet des Objets. Dans le contexte des réseaux multi-échelles (ambiant, Internet, cloud) et pour l'Internet des Objets, la gestion du contexte devient beaucoup plus complexe. Le but du projet INCOME est de répondre à trois défis scientifiques:

- La gestion de contextes multi-échelles par un middleware qui est déployé sur différents dispositifs ou serveurs répartis sur différents réseaux ou le cloud, nommé MDCM (Multiscale Distributed Context Manager).
- La gestion des préoccupations extra-fonctionnelles (qualité du contexte et la protection de la vie privée).
- Le déploiement autonome des entités qui composent le gestionnaire de contexte [30].

Les principales composantes de l'architecture INCOME : La figure 3.1 montre les principales composantes de l'architecture INCOME assurant la distribution des données entre les producteurs et les consommateurs de contexte.

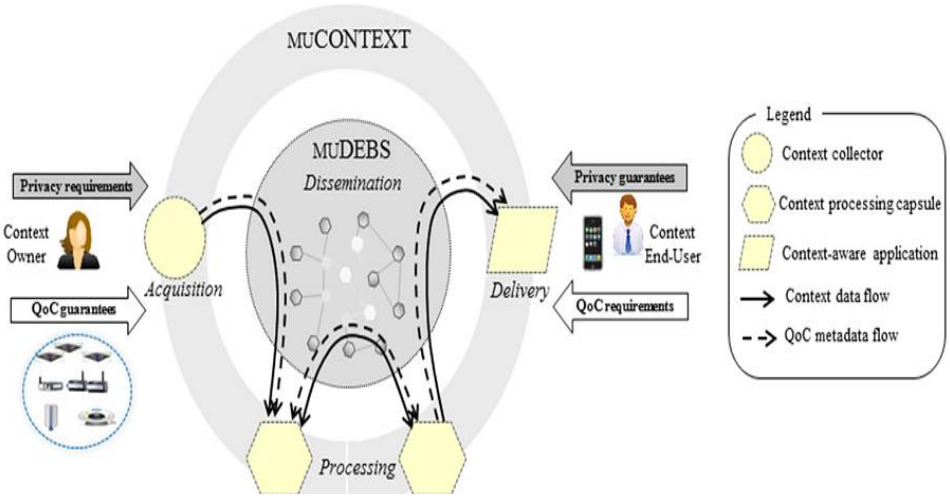

Figure 3.1: L'architecture logique d'INCOME [23].

La figure 3.1 montre l'architecture d'approche INCOME avec ses différentes couches. Chaque couche est décrite ci-dessous :

- Le framework MUCONTEXT : se compose de trois types d'entités logicielles: collecteur, capsule et application. Le collecteur est responsable de l'acquisition des informations de contexte de bas niveau, avec une signification sémantique faible. Il est dans la plupart des cas associé aux capteurs physiques, mais peut aussi, des réseaux sociaux (facebook, twitter.etc), des services web ou d'autres sources d'information. La capsule analyse et transforme les informations de contexte produites par des collecteurs ou d'autres capsules pour produire une nouvelle information ou des informations avec un plus haut niveau abstraction correspondant aux exigences des applications. Ces derniers sont responsables de présenter aux utilisateurs les informations produites par les capsules [47].
- Le cadre MUDEBS prend en charge le routage des informations entre les entités de la couche MUCONTEXT. Ce cadre définit deux rôles, pour le producteur d'information et aussi pour le consommateur. Les entités logicielles "collecteur" et "capsule" ayant le rôle de producteur, tandis que les entités logicielles "capsule" et "application" ont un rôle de consommateur. Les informations reçues par les consommateurs peuvent être transmises en utilisant le "push" ou "pull". Ce mode de communication basé sur "observation-notification" où les applications peuvent

parfois demander de consommer des données de contexte par émission de requête (observations). Les messages qui répondent à ces demandes sont des notifications appelées le mode push [42].

Les échanges d'informations sont fournis dans MUDEBS par un réseau de Brokers. MUDEBS est basé sur des contrats, appelés contrat de production et contrat de consommation, pour fournir la bonne information aux consommateurs de contexte. Un contrat de producteur est défini comme un filtre (filtres d'annonce) indiquant quelle est la nature de l'information fournie par un producteur. De même, un contrat de consommation est exprimé en tant que filtre (filtres de souscription) par les consommateurs pour spécifier les informations dont ils ont besoin [23].

3.4.1.2 L'étude de l'approche INCOME selon différents critères

- **Hétérogénéité et mobilité des objets :** Cela se fait à travers des «contrats de production de contexte» et aussi «contrats de consommation de contexte» qui définissent les conditions que les producteurs et les consommateurs acceptent à partager / recevoir des données. Cela permet donc une connexion entre les exigences de l'application et les garanties de producteurs de contexte qui sont à la fois complètement mobiles et hétérogènes et fortement découplés d'une manière spatio-temporel. La figure 3.2 indique l'accord et la connexion entre les producteurs et les consommateurs de contexte.

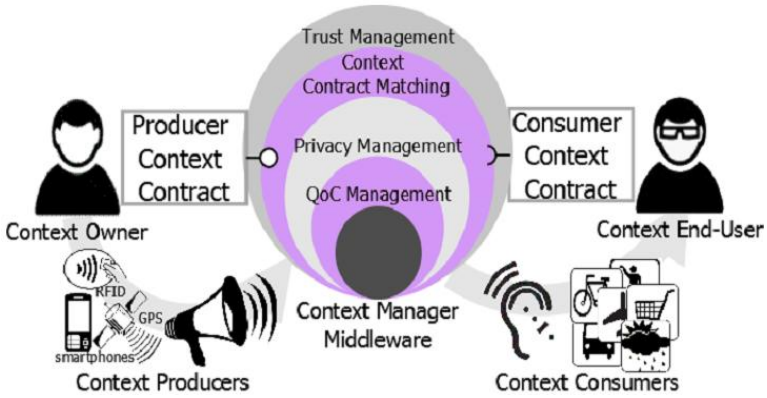

Figure 3.2: Contrat des producteurs et des consommateurs de contexte dans INCOME [23].

- **Influence du monde physique sur l'Internet des objets :** L'Internet des objets est fondamentalement influencé par les caractéristiques du monde physique et des outils qui permettent de le mesurer. Les différentes sources de contexte fournissent des informations inexactes, erronées et ambiguës. Pour ce but, INCOME fournit un mécanisme pour la gestion de la qualité de contexte à chaque étape du cycle de vie de l'information au sein d'un gestionnaire de contexte. Les processus d'analyse de la QoC permettent de fournir une technique de correction des erreurs en éliminant les informations dont le niveau de la QoC est trop faible et garder les informations qui ont une meilleure qualité. En conséquence, les applications nécessitent une gestion efficace de la QoC pour prendre les meilleures décisions par leurs adaptations aux changements de contexte qui se produisent au cours de temps.

- **Caractérisation multi-échelle :** La fonctionnalité multi-échelle d'un système peut être envisagée à différents points de vue. Pour le projet INCOME, les vues suivantes ont été sélectionnées: équipement, réseau, données, géographie, administration des utilisateurs. Pour chacune d'entre elles, des dimensions pertinentes et une échelle significative sont choisies. MuScA (MultiScale Characterization framework) [48] est la cadriciel de la caractérisation multi-échelle orienté modèle proposé par l'INCOME. Ce cadre vise, d'une part, à mettre en œuvre le processus de caractérisation

multi-échelle, et d'autre part, un mécanisme pour la production d'artefacts logiciels qui deviennent conscientes à des échelles lors de la phase d'exécution.

- **Sécurité et la protection de la vie privée :** INCOME fournit une bonne sécurité pour les pairs de communication et pour les protocoles de transmission. Il assure la protection de la vie privée sur trois niveaux de matériel, les protocoles et d'application. Ce projet fournit un contrat de production et de consommation de contexte, qui permet à l'utilisateur d'indiquer quelles informations peuvent être collectées et quelles sont ses exigences de manière sécurisée. Ces contrats peuvent contrôler qui peut utiliser l'information et par quel type d'applications et en même temps assurer une communication sécurisée entre les producteurs et les consommateurs de contexte.

- **Passage à l'échelle :** Avec le nombre croissant de producteurs et de consommateurs de contexte, une solution centralisée n'est pas possible. Pour assurer un fonctionnement stable (gestion de l'information et des connaissances, gestion des services, communications de données et réseaux), le gestionnaire de contexte distribué MDCM (Multiscale Distributed Context Manager) [24] offre cette possibilité par la distribution de ses entités logicielles. Certaines entités seront affectées par la collecte de données brutes, elles seront déployées à proximité des capteurs, les entités de traitement peuvent être déployées sur de nombreuses machines différentes, telles que des ordinateurs personnels, des serveurs dédiés ou dans le cloud, et finalement les applications qui peuvent être déployé aux utilisateurs.

- **L'interopérabilité :** Le projet INCOME fournit une interopérabilité fiable et cela peut être obtenu en échangeant les informations entre les différentes entités du gestionnaire de contexte (les entités de collection, traitement, et de présentation).

- **Déploiement autonome d'entités :** Pour répondre aux besoins d'expression et d'abstraction des plans de déploiement des systèmes distribués multi-échelle, les membres du projet INCOME ont défini un langage dédié appelé MuScADeL [48]. Ce langage a été conçu pour permettre l'expression des contraintes de déploiement d'un système logiciel dans un environnement multi-échelle, comme dans ce cas un gestionnaire de contexte distribué.

- **Qualité de contexte :** INCOME propose une solution pour la gestion de la QoC en sein d'un MDCM consiste à utiliser un modèle commun pour modéliser les critères de QoC. A cet effet, le méta-modèle QoCIM [49] a été proposé récemment pour définir un méta-modèle générique pour la QoC afin de définir tout type de paramètre de qualité de contexte. Les gestionnaires de contexte distribués doivent prend en compte la QoC à chaque étape de cycle de vie du contexte: de son acquisition par les différentes sources de contexte à sa livraison aux applications. Ce projet définit deux types de contrats de contexte [50] : le contrat de production de contexte qui a pour objectif de définir les garanties de QoC que le producteur de contexte est prêt à fournir. Le contrat de consommation de contexte qui indique les exigences de QoC que le consommateur attend. Comme l'illustre la figure 3.3, la nouvelle génération des gestionnaires de contexte doivent prendre en charge la mesure de la qualité de contexte à chaque étape du cycle de vie des données de contexte.

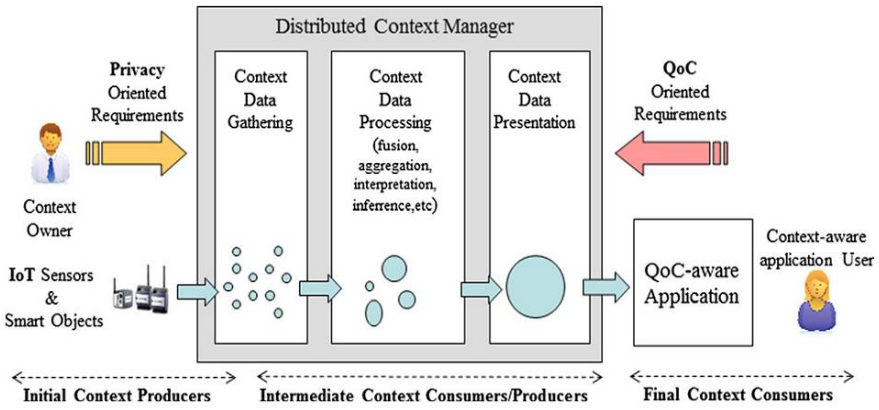

Figure 3.3: Gestionnaire de contextes distribué dans projet INCOME [49].

- **Cycle de vie de contexte**

  - Acquisition de contexte: le projet INCOME utilise le Context Collector: est un logiciel représentatif d'un producteur de contexte qui encapsule le comportement d'acquisition en mode push ou pull de données brutes issues du monde observé.
  - Modélisation de contexte: Le projet INCOME utilise l'ontologie et le méta-modèle QoCIM pour la modélisation.
  - Raisonnement de contexte : Le projet INCOME utilise l'ontologie pour le raisonnement.
    En raison de la fusion de deux ou plusieurs éléments de contexte qui caractérise une situation de différentes dimensions de la même entité observée, cela peut conduire à des conflits de contexte. INCOME est basé sur la résolution des conflits pour la qualité du contexte et aussi la validation, cela assure que les données recueillies sont correctes et significatif.
  - Distribution de contexte : Le projet INCOME utilise le mode pull et push pour disséminer les informations entre les producteurs et les consommateurs de contexte.

3.4.2 Project SITAC (Internet of Things – Applications by and for the Crowd)

3.4.2.1 Description

SITAC est un projet international d'un programme européen ITEA 2 (Information Technology for European Advancement) fondée en 2012 en Espagne. Il est financé par le Ministère de l'Economie et de la Compétitivité (programme INNPACTO). Le projet vise à permettre un accès à grande échelle à des milliards d'appareils connectés en exploitant trois paradigmes: les réseaux sociaux, les applications basées sur la Crowd et l'analyse de données, comme l'indique la figure 3.4 [51]. Le projet ITEA2 SITAC propose de relever le défi de créer une architecture et un «écosystème» unifiant comprenant des plates-formes, des outils et des méthodologies. Ce projet permet la connexion et la coopération de plusieurs types d'entités connectées au réseau: systèmes, machines, appareils ou dispositifs. La gestion de Big Data, la gestion d'un grand nombre d'entités et la gestion avancée du contexte sont des rôles spécifiques du SITAC [52].

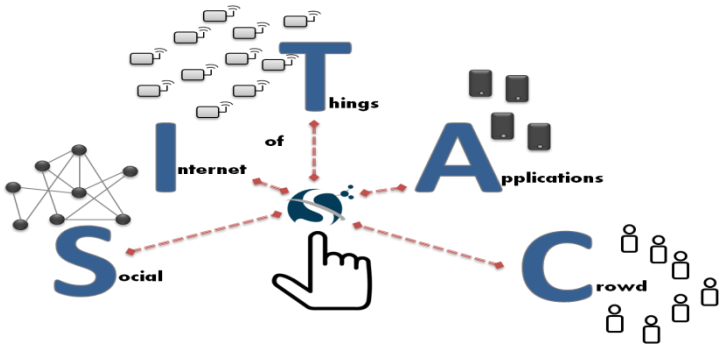

Figure 3.4: SITAC: Social Internet of Things: Apps by and for the Crowd [53].

L'Architecture fonctionnelle de SITAC : Le projet SITAC vise à développer une architecture avec des technologies permettant de créer une plateforme commune pour des réseaux hétérogènes capables de relever les défis de passage à l'échelle, l'adaptabilité et la sécurité. Cette architecture permettra d'exposer des réseaux (et des infrastructures) hétérogènes afin de permettre au propriétaire d'un périphérique de sélectionner consciemment (ou de construire) une infrastructure de réseau appropriée en fonction de besoins spécifiques. De plus, l'utilisation de techniques d'interopérabilité sémantique et la sensibilité du contexte facilitera la diffusion de l'information au bon endroit et au bon moment. Le défi consiste à déterminer les outils et les architectures pour la collecte et le partage des données afin de stimuler la création de services et d'applications par le Crowd.

Une vue fonctionnelle de l'architecture SITAC est représentée dans la figure 3.5. En termes de fonctions, l'architecture peut être représentée par : SITAC- Core, où les fonctions principales et génériques consistent principalement de traitement, stockage, gestion des périphériques, couche de service et gestion de la communication. Ces fonctions devraient être disponibles au format API, afin de permettre l'interrelation. SITAC- Front-end, où les services de base sont placés ensemble pour fournir des fonctionnalités aux utilisateurs. Ceux-ci comprennent principalement des applications et des services qui peuvent être consultés et des capacités pour l'analyse des données [51].

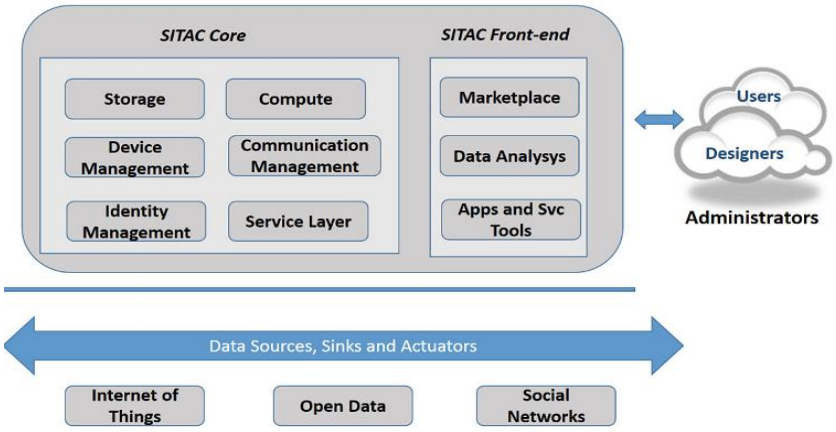

Figure 3.5: La vue fonctionnelle du projet SITAC [51].

3.4.2.2 L'étude de l'approche SITAC selon différents critères

- **Hétérogénéité et mobilité des objets :** L'utilisation de techniques d'interopérabilité sémantique et la sensibilité du contexte facilitent la diffusion de l'information au bon endroit au bon moment. Cette technique permet de créer un lien de connexion entre les objets qui sont complètement découplés en spatio-temporel et fortement hétérogènes et ainsi mobiles.

- **Influence du monde physique sur l'Internet des objets :** Le projet SITAC a résolu les problèmes d'adaptation des objets mobiles aux changements qui survient par le monde extérieur via le composant « Device Management» appartenant au composant principal SITAC Core.

- **Caractérisation multi-échelle:** Elle n'a pas été étudiée dans ce projet.

- **Sécurité:** ce projet capable de résoudre les problèmes de sécurité.

- **La protection de la vie privée :** Elle n'a pas été citée dans ce projet.

- **Passage à l'échelle :** Pour résoudre le problème du très grand nombre d'objets, cette approche offre une architecture structurée en deux composants principaux SITAC Core et SITAC Front-end. Chacun de ces composants contient des entités fonctionnelles utilisées pour distribuer des tâches (stockage d'information, traitement, gestion de communication ...). Cette architecture permet de réaliser la stabilité de fonctionnement quelque soit l'augmentation de ces objets. Le projet STAC est basé sur le cloud pour réduire la complexité de la gestion des entités et la quantité d'informations, la gestion des services, la communication de données et de réseaux.

- **Interopérabilité :** Le projet SITAC donne la possibilité que les entités de SITAC Core ou SITAC Front-end deviennent interopérable en échangeant des informations entre eux et avec le monde extérieur.

- **Déploiement autonome des entités :** Il n'a pas été étudié dans ce projet.
- **Qualité de contexte** : Elle n'a pas été étudiée dans ce projet.
- **Cycle de vie de contexte :** le détail de ce problème n'a pas été cité dans ce projet.

3.4.3 FIWARE Orion Context Broker

3.4.3.1 Description

La Fondation FIWARE crée par une industrie européenne en 2015 pour promouvoir la tendance vers les plates-formes de services ouverts. Il est considéré comme un engagement visible par les entreprises européennes pour proposer des services internet plus innovants aux consommateurs, aux citoyens, aux entreprises et au secteur public [54].

Le principal objectif du projet FIWARE est de construire une plate-forme de base de l'internet du futur en introduisant une infrastructure innovante pour la création et fournir des services numériques polyvalents, basée sur une qualité de service élevée et des garanties de sécurité. Le pilier central du projet FIWARE est la plate-forme FIWARE, constituée d'une bibliothèque complète d'API puissantes visant à faciliter le développement d'applications complexes. Ces API sont rassemblées sur des éléments réutilisables (Generic Enablers GE) centrés dans différents domaines d'utilisation, appelés FIWARE chapters. L'architecture générale de FIWARE couvre les domaines suivants : Data\ context management, IoT Services Enablement, Advanced Web-based User Interface, Security, Interface to Network and Devices, Architecture of Applications\ Services Ecosystem and Delivery Framework, et Cloud Hosting. Cette architecture permet d'apporter des APIs bien définies qui construisent des applications et des services capables de recevoir des données de contexte en temps réel.
La figure 3.6 décrit la composition du chaque chapter de Framework FIWARE [55].

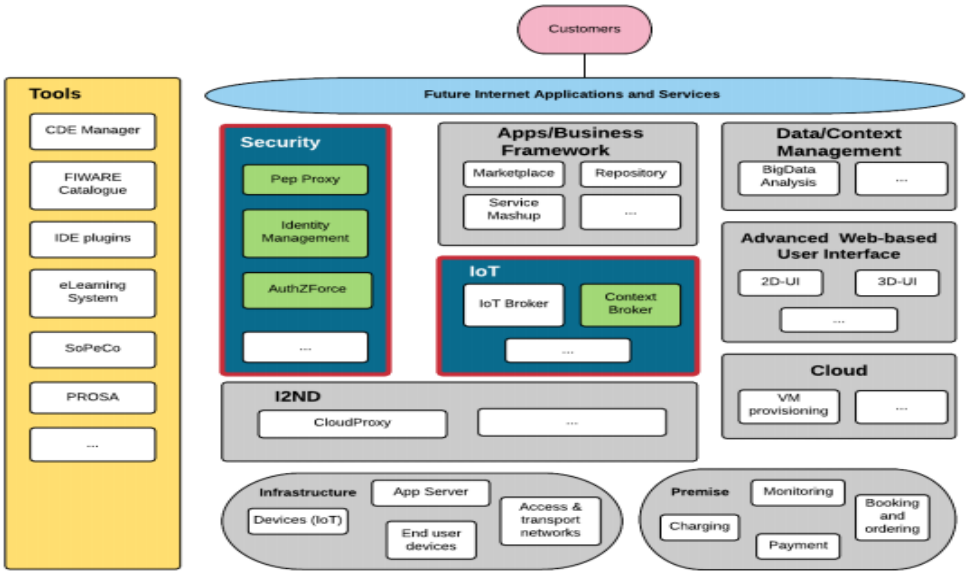

Figure 3.6: La composition des chapitres de Framework FIWARE [55].

Dans cette approche nous avons intéressé à l'Orion Context Broker GE qui appartient au chapter « data \ context management » comme l'illustré la figure 3.7.

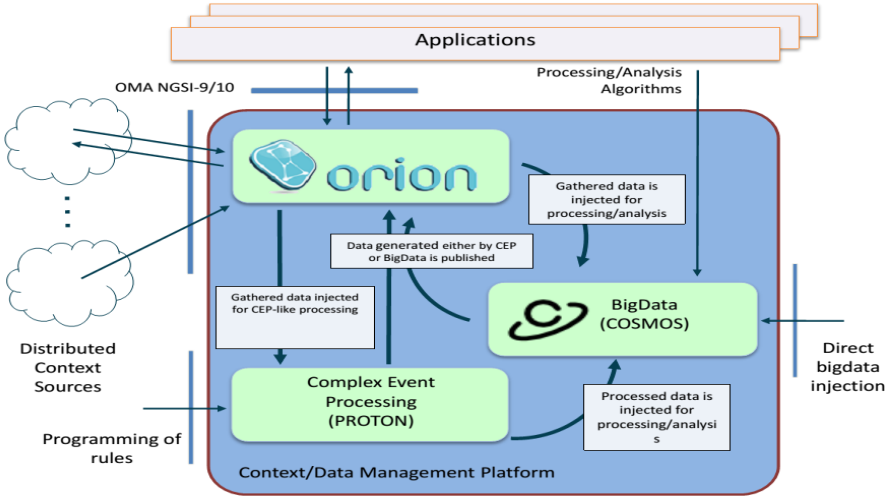

Figure 3.7: Plateforme de context\data management [56].

Gestionnaire de contexte « « Orion Context Broker»

Orion Context Broker, également appelé «publish/subscribe context broker», est une excellente passerelle pour permettre aux applications externes de gérer les événements liés à l'Internet des objets. Le Context Broker fonctionne de manière simple en cachant la complexité des mesures de collecte par les ressources d'IdO qui pourraient être distribuées ou impliquant un accès via plusieurs protocoles de communication. La fonctionnalité de Context Broker peut être simplifiée par la méthode suivant : Les informations de contexte sont publiées par certaines entités, référencées en tant que producteurs de contexte, rendant ces informations disponibles pour d'autres entités, appelées consommateurs de contexte. Ce dernier pouvant extraire les informations de contexte à partir de context broker, ou sinon, ce context broker peut publier les informations aux consommateurs de contexte qui ont souscrit à certaines conditions [55].

Orion context broker est un service basé sur les interfaces OMA NGSI 9/10. Il est capable de gérer l'envoi et la réception des informations de contexte et aussi de gérer un grand nombre de messages d'entités et de gérer les mises à jour, les requêtes, l'enregistrement, notifier les changements d'informations de contexte, et la souscription [57]. Ces fonctionnalités sont indiquées dans la figure 3.8.

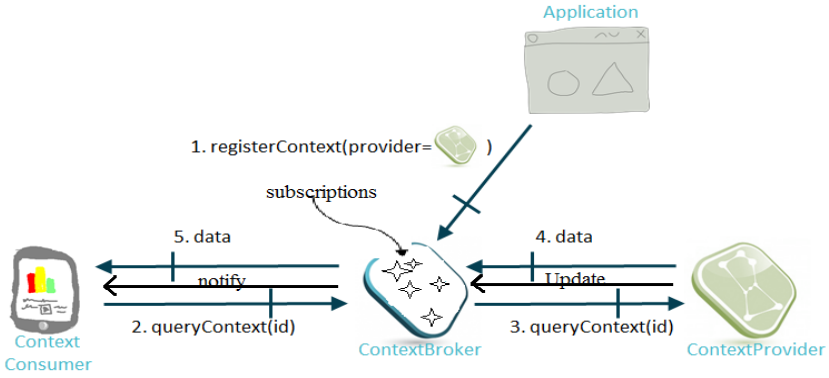

Figure 3.8: Opérations fonctionnelles d'Orion Context Broker [58].

3.4.3.2 L'étude de l'approche FIWARE selon différents critères

- **Hétérogénéité et mobilité des objets :** L'utilisation des interfaces NGSI10 / NGSI9 permet d'échanger les informations entre les différentes objets. Ces interfaces vise à créer un protocole de communication qui ne se soucie pas au l'hétérogénéité ou le changement d'emplacement des objets.

- **Influence du monde physique sur l'Internet des objets :** Ce projet ne supporte pas une technique de correction d'erreur ou un mécanisme de traitement de la limitation d'énergie mais grâce aux interfaces NGSI10 / NGSI9, il permet aux objets mobiles de s'adapter aux changements qui surviennent au cours de temps.

- **Caractérisation multi-échelle:** Elle n'a pas été étudiée dans ce projet.

- **Sécurité et la protection de la vie privée :** Fiware fournit une plate-forme pour assurer la sécurité. Les GEs qui appartiennent à cette plate-forme ont fait la livraison et l'utilisation de services fiables en conditions des exigences de sécurité et de protection de la vie privée.

- **Passage à l'échelle :** En raison du grand nombre d'entités impliquées dans l'Internet des Objets, cette approche offre des interfaces pour gérer un grand nombre d'entités et de messages afin d'assurer la stabilité de fonctionnement. Ces interfaces décrivent la fonctionnalité de base de l'Orion context broker : interface NGSI10 pour la gestion des contextes (c'est-à-dire des informations sur les entités) et la gestion de la disponibilité du contexte assurée par l'interface NGSI9 (des informations non sur les entités elles-mêmes, mais sur les fournisseurs de cette information).

- **Interopérabilité :** Dans le projet FIWARE, l'interopérabilité peut être obtenue en échangeant les informations entre les chapters de la plateforme et entre les composants de chaque chapters eux-mêmes.

- **Déploiement autonome des entités :** Il n'a pas été étudié dans ce projet.

- **Qualité de contexte** : Elle n'a pas été étudiée dans ce projet.

- **Cycle de vie de contexte**
  - Acquisition de contexte: le projet FIWARE utilise des entités « context provider »: comme un entité intermédiaire responsable de la collecte, la gestion et la diffusion des informations de contexte.
  - Modélisation de contexte: Le projet FIWARE utilise l'ontologie pour la modélisation.
  - Raisonnement de contexte: le projet FIWARE utilise l'ontologie pour le raisonnement.
  - Distribution de contexte: ce projet utilise le mode pull et push pour distribuer les informations.

3.4.4 Approche CA4IOT (Context Awareness for Internet of Things)

3.4.4.1 Description

Est une architecture proposée en 2013 pour aider les utilisateurs en automatisant la tâche de sélection des capteurs en fonction des problèmes / tâches plutôt que de fournir une solution middleware complète pour gérer les données de contexte. CA4IOT [59] peut être adopté dans n'importe quelle solution de middleware d'IdO. Une vue d'ensemble de l'architecture de CA4IOT est présentée à la figure 3.9.
L'architecture CA4IOT se compose de quatre couches:

- SDAL (Sensor Data Acquisition Layer): Les composants principaux situés dans cette couche sont les sensor wrappers, wrapper repository, wrapper generator, sensor device definition (SDD) local repository et SDD cloud repository. Cette couche est responsable d'acquisition d'une variété de données de contexte.
- CSDL (Context et Semantic Discovery Layer): Cette couche est responsable de la gestion du contexte et des découvreurs sémantiques, ce qui inclut la génération, la configuration et le stockage. Les composants pertinents sont : context and semantic discoverers, context and semantic discoverer generator, and context and semantic discoverers repository.
- CPRL (Processing and Reasoning Layer): Une collection de fonctions importantes est distribuée dans cette couche comme le traitement de données, le raisonnement de haut niveau pour les informations de contexte, fusion des données, génération de connaissances et le stockage. Les composants appartenant à cette couche sont: context registry, context knowledgebase, reasoning engine, context and semantic discoverer generator, primary context processing, secondary context processor, context provider registry, data fusion operator, and data fusion repository.
- DSCDL (Data, Semantics, and Context Dissemination Layer): les utilisateurs peuvent effectuer des demandes via des interfaces multi-modèles. Un dépôt local peut interagir avec des dépôts qui résident dans le cloud ou ouvrir des données liées pour fournir de meilleures réponses à ces requêtes grâce à l'analyse de Big Data [32].

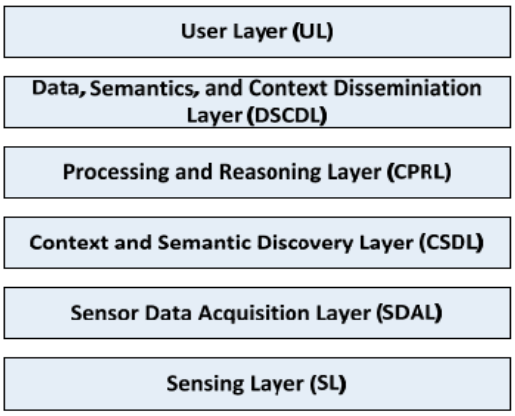

Figure 3.9: Architecture de CA4IOT [32].

L'objectif principal de cette architecture est représenté dans la figure 3.10 en trois étapes. À l'étape 1, l'utilisateur fournit ses besoins à CA4IOT. Ce dernier comprend le problème de l'utilisateur et sélectionne les capteurs appropriés comme indiqué à l'étape 2. À l'étape 3, CA4IOT combine les données extraites à partir des capteurs sélectionnés en un seul flux de données et les envoie à l'utilisateur [59].

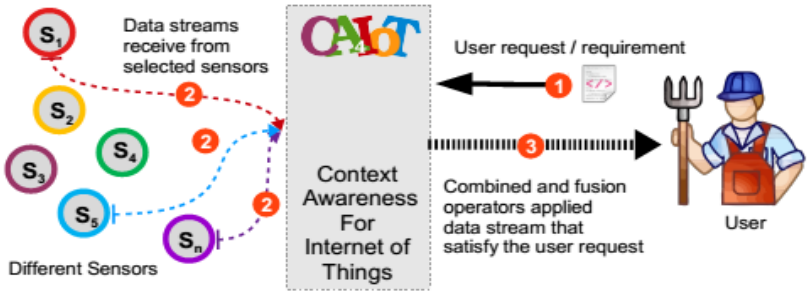

Figure 3.10: L'objectif fonctionnel de CA4IOT [59].

3.4.4.2 L'étude de l'approche CA4IOT selon différents critères

- **Hétérogénéité et mobilité des objets :** l'architecture de CA4IOT est basée sur des couches (Sensor Data Acquisition Layer (SDAL), Context and Semantic Discovery Layer (CSDL), Processing and Reasoning Layer (CPRL), Data Semantics, and Context Dissemination Layer (DSCDL)). La diffusion d'informations de manière successives entre ces couches est considérée comme un protocole de communication entre les différentes entités. Ce type d'architecture permet de réaliser un lien de connexion entre les objets quel que soit leur hétérogénéité et leur mobilité.

- **Influence du monde physique sur l'Internet des objets:** Elle n'a pas été étudiée dans ce projet

- **Caractérisation multi-échelle:** Elle n'a pas été étudiée dans ce projet.

- **Sécurité et la protection de la vie privée :** Elle n'a pas été étudiée dans ce projet.

- **Passage à l'échelle :** Les techniques orientées Cloud sont des solutions élégantes pour résoudre le problème de passage à l'échelle à partir de différents aspects. Le middleware CA4IoT offre de

multiples avantages du Cloud : stockage de données volumineuse, améliore les capacités de traitement grâce au Cloud Computing et extrait les données grâce à l'analyse de Big Data.

- **Interopérabilité :** Avec CA4IoT, l'interopérabilité peut être obtenue en échangeant les informations entre le middleware CA4IoT et avec des capteurs et des applications même entre les couches de ce middleware eux-mêmes.

- **Déploiement autonome des entités :** Il n'a pas été étudié dans ce projet.

- **Qualité de contexte** : Elle n'a pas été étudiée dans ce projet.

- **Cycle de vie de contexte**

  - Acquisition de contexte: CA4IoT utilise un middleware (SDAL): Cette couche est responsable de l'acquisition des données.
  - Modélisation de contexte: Le projet CA4IoT utilise l'ontologie pour la modélisation.
  - Raisonnement de contexte: le projet CA4IoT utilise l'ontologie et une forme statistique pour le raisonnement.
  - Distribution de contexte: Ce projet utilise le mode requête/réponse où les utilisateurs sont capables de faire des requêtes à tout moment et le CA4IoT comprend le problème de l'utilisateur et sélectionne les capteurs appropriés pour répondre aux demandes.

3.4.5 Approche SAMURAI: A Streaming Multi-tenant Context-Management Architecture for Intelligent and Scalable Internet of Things Applications

3.4.5.1 Description

Est une architecture d'event-based stream mining développée dans le projet FP7 BUTLER en 2014 pour le but d'une architecture sécurisée et sensible au contexte pour l'Internet des objets. Cette approche intègre et expose des composants bien connus : complex event processing, machine learning, knowledge representation, NoSQL persistence et in-memory data grids.
Elle offre des fonctionnalités de publication / souscription pour que les clients soient notifiés lorsque des événements particuliers se produisent avec des notifications push implémentées comme callbacks REST. L'architecture de SAMURAI a trois composants de base pour organiser des événements comme l'illustré la figure 3.11:

- In-memory Data Grid: il s'agit d'un conteneur de mémoire distribué pour les événements.
- Event Queue: les clients qui ne prennent pas en charge les notifications push via le callbacks REST peuvent enregistrer une file d'attente pour stocker des événements et les interroger.
- NoSQL Store: Les événements peuvent être stockés de manière persistante en tant que base de données RESTful. Les API RESTful suivent le mappage CRUD sur les méthodes HTTP pour créer des événements (POST), read (GET), update (PUT) ou delete (DELETE) [60].

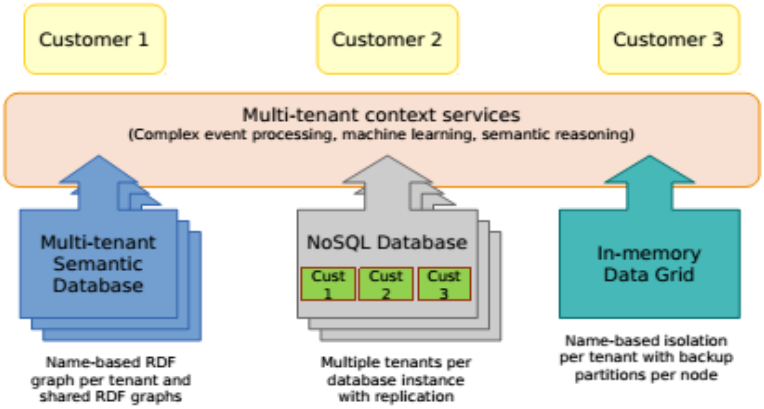

Figure 3.11: Architecture de SAMURAI [60].

Il y a d'autres composants dans cette architecture:

- Complex event processing: Cette composante permet l'extraction des caractéristiques à partir d'événements de bas niveau et appliquer l'interaction « publier / souscription » avec des applications ou des sous-systèmes SAMURAI. Dans l'écosystème d'IdO, de nouveaux types d'événements peuvent être créés à tout moment. Il expose donc une API RESTful pour enregistrer dynamiquement de nouveaux types d'événements lors de l'exécution.
- Semantic database: SAMURAI peut tirer parti des connaissances de base stockées dans une base de données sémantique pour augmenter la signification d'un événement.
- Learning and mining**:** Lorsque la relation entre les événements co-occurrents ne peut pas être établie à l'avance, nous avons besoin de mécanismes de classification et de regroupement pour déduire de manière probabiliste ces dépendances. SAMURAI intègre la bibliothèque d'apprentissage automatique Weka pour ce but et expose ses fonctionnalités grâce aux API RESTful [60].

3.4.5.2 L'étude de l'approche SAMURAI selon différents critères

- **Hétérogénéité et mobilité des objets :** l'architecture offre des API RESTful pour enregistrer les instructions (des requêtes) et les listeners. Cette approche fournit des résultats aux listeners lorsque de nouveaux événements arrivent. Pour que les applications ou les sous-systèmes soient notifiés sur les événements, SAMURAI ajoute un listener à cette instruction. Par conséquent, quelle que soit hétérogénéité et la mobilité des objets, API RESTful permet de relier ces objets entre eux.

- **Influence du monde physique sur l'Internet des objets:** Elle n'a pas été étudiée dans ce projet.

- **Caractérisation multi-échelle:** Elle n'a pas été étudiée dans ce projet.

- **Sécurité :** dans l'architecture SAMURAI, les données sont provisionnées dans In-memory Data Grid, the Semantic Database, the NoSQL Store and the Event Queues. L'utilisation de serveurs dédiés par chaque tenant offre une méthode fiable pour la sécurité.

- **La protection de la vie privée :** Elle n'a pas été citée dans ce projet

- **Passage à l'échelle :** SAMURAI intègre et expose des blocs logiciels pour complex event processing, machine learning and knowledge representation. Chacun de ces blocs est intégré de

manière faiblement couplée, ce qui permet de déployer facilement plusieurs instances de l'architecture dans le cloud. Cette approche poursuit deux méthodes pour résoudre le problème de passage à l'échelle :
1) Déploiement distribué avec une évolutivité horizontale.
2) Partage des ressources grâce à la multi-tenancy.

- **Interopérabilité :** SAMURAI donne la possibilité d'échanger les informations entre les composantes de l'architecture multi-tenancy.

- **Déploiement autonome des entités :** Les sous-systèmes de SAMURAI exposent leurs fonctionnalités via des API RESTful, ce qui simplifie le déploiement distribué de chaque sous-système sur un nœud différent. Il y a deux expériences pour cet effet :
  - La première expérience compare un déploiement de tous les sous-systèmes sur un seul nœud avec un déploiement distribué ayant chaque sous-système sur un nœud dédié.
  - La deuxième, il déployé plusieurs instances d'un système complet de SAMURAI sur différents nœuds. Il utilise un simple équilibreur de charge avec des sessions persistantes pour toujours rediriger les appels consécutifs des clients vers la même instance SAMURAI.

- **Qualité de contexte** : SAMURAI offre une sensibilité de la situation de haut niveau en supportant la QoC dans chaque situation de contexte. Il peut convertir les données et les événements de bas niveau en caractéristiques qui sont plus significatives pour l'utilisation par l'application sensible au contexte ou pour comparer la qualité du contexte. MLContext [61] est un langage textuel spécifique au domaine (DSL) conçu pour modéliser le contexte dans SAMURAI. Il a été utilisé pour générer automatiquement des artefacts logiciels liés à la gestion de contexte pour certains Framework sensible au contexte. Dans cette approche, l'extension MLContext est utilisée pour la modélisation de la qualité de contexte.

- **Cycle de vie de contexte**
  - Acquisition de contexte: SAMURAI utilise des API RESTful pour extraire les données de capteurs par la requête GET.
  - Modélisation de contexte: SAMURAI utilise le modèle MLContext pour la modélisation.
  - Raisonnement de contexte: SAMURAI utilise le modèle MLContext pour le raisonnement.
  - Distribution de contexte: Ce projet utilise le mode «pull and push» pour diffuser les informations de contexte.

3.4.6 Approche HYDRA

3.4.6.1 Description

Cette approche financée dans le cadre du programme FP6 IST en 2010. Hydra [62] est un middleware bien connu pour le système basé sur l'Internet des objets. Ce middleware a été conçu pour faciliter l'interaction avec les appareils par l'abstraction des informations détaillées sur ces appareils et leurs réseaux. Il Permet aux développeurs d'intégrer des périphériques physiques hétérogènes en offrant des interfaces de service web faciles à utiliser pour contrôler n'importe quel type d'appareil physique indépendamment de sa technologie de réseau comme Bluetooth, ZigBee, WiFi.etc. Hydra est un middleware basé sur SOA (Service Oriented Architecture). Il considère chaque périphérique comme un service et utilise des langages d'ontologie, par ex. OWL, OWL-s et SAWSDL, pour définir des descriptions sémantiques de ces dispositifs.

Le but de cette approche est de créer un middleware le plus largement déployé pour les systèmes embarqués intelligents qui permettra aux producteurs de développer des applications embarquées rentables et innovantes pour les appareils nouveaux ou qui sont déjà existants [62]. La figure 3.12 montre les composants de l'architecture Hydra.

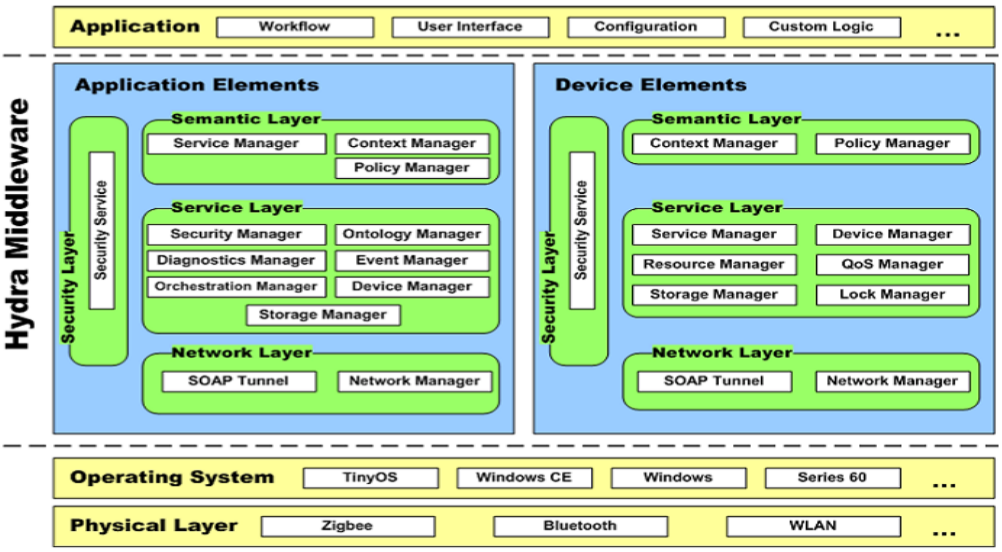

Figure 3.12: Les composants internes et externes du middleware Hydra [63].

3.4.6.2 L'étude de l'approche HYDRA selon différents critères

- **Hétérogénéité et mobilité des objets :** Le middleware Hydra permet d'intégrer des périphériques physiques hétérogènes et mobile en proposant des interfaces de service web faciles à utiliser pour contrôler n'importe quel type d'appareil physique, quelle que soit sa technologie de réseau. Cette méthode fournit un accès interopérable aux données, informations et connaissances sur des plates-formes hétérogènes dans laquelle la couche de communication est transparente.

- **Influence du monde physique sur l'Internet des objets:** En raison de l'évolution rapide d'environnement, des utilisateurs mobiles, des services ambiants, ce Middleware et les objets connectés devraient permettre aux développeurs de mettre en œuvre des applications qui dépendent et d'adapter à l'information de contexte. D'autre part, le middleware HYDRA est déployé sur des nouveaux ou existants réseaux de périphériques distribués, qui fonctionnent avec des ressources limitées en termes de puissance de calcul, d'énergie et d'utilisation de la mémoire.

- **Caractérisation multi-échelle:** Elle n'a pas été étudiée dans ce projet.

- **Sécurité et la protection de la vie privée** : Hydra prend en charge la gestion des clés et concentre sur les préférences de sécurité et de la vie privée dans la sélection sensible au QoS. Ce middleware utilise la description sémantique qui devrait permettre aux dispositifs individuels de spécifier le type et le niveau de services de sécurité qu'ils exigent et qu'ils fournissent. En d'autres termes, les appareils et les services dans Hydra peuvent être sécurisés et fiables grâce à des composants distribués de sécurité.

- **Passage à l'échelle :** Hydra support une gestion supérieure de tous les dispositifs d'une manière intégrée, assurant le passage à l'échelle. Pour faciliter l'utilisation de ces appareils, Hydra a développé un middleware orienté services évolutif.

- **Interopérabilité :** La description sémantique prend en charge l'interopérabilité entre les périphériques de middleware Hydra qui sont hétérogènes et distribués.

- **Déploiement autonome des entités :** Hydra facilite le processus de déploiement en fournissant une interface permettant d'interagir avec les périphériques considérés en tant que fournisseurs de services au moment du déploiement. Pour les appareils qui n'ont pas une puissance de calcul suffisante pour être un fournisseur de services, Hydra utilise un proxy qui permet à ces appareils d'interagir avec le middleware Hydra via le protocole IP [63].

- **Qualité de contexte** : Elle n'a pas été étudiée dans ce projet.

- **Cycle de vie de contexte**

  - Acquisition de contexte: HYDRA a développé un middleware basé sur une architecture orientée services (SOA), pour lequel la méthode d'acquisition des données est faite par des interfaces API basé sur SOAP.
  - Modélisation de contexte: Hydra considère chaque périphérique comme un service et utilise des langages d'ontologie, par ex. OWL, OWL-s et SAWSDL, pour modéliser ces périphériques par une description sémantique.
  - Raisonnement de contexte: Hydra a utilisé les ontologies pour faire le raisonnement.
  - Distribution de contexte: Cette approche a utilisé le mode «requête-réponse » pour diffuser les informations de contexte.

3.4.7 Approche AURA

3.4.7.1 Description

Middleware d'Aura [63] a été proposé en 2002 et se concentré sur l'élaboration et la manipulation des données collectées à partir des périphériques dans l'IdO**.** Son but est de fournir une facilité de configuration et de déploiement pour l'utilisateur final et les développeurs. Ce middleware prend en charge l'interaction avec des dispositifs complexes et leur intégration.

AURA définit un proxy, appelé AURA personnel qui permet aux utilisateurs d'utiliser un périphérique indépendant de leurs emplacements physiques. La figure 3.13 montre les composants de l'architecture du Framework AURA et leurs interactions. Il existe quatre types de composants : 1) le gestionnaire de tâches (Task Manager), appelé Prism, pour incarner le concept d'aura personnelle. 2) l'observateur de contexte (Context Observer) rassemble les informations sur le contexte physique et déclenche un événement au Prism et le gestionnaire d'environnement (Environment Manager). 3) le gestionnaire de l'environnement incarne la passerelle vers l'environnement ; et 4) les fournisseurs (Suppliers) produisent des services abstraits dont les tâches ont composées : édition de texte (text editing), lecture de vidéos (video playing) [63, 64].

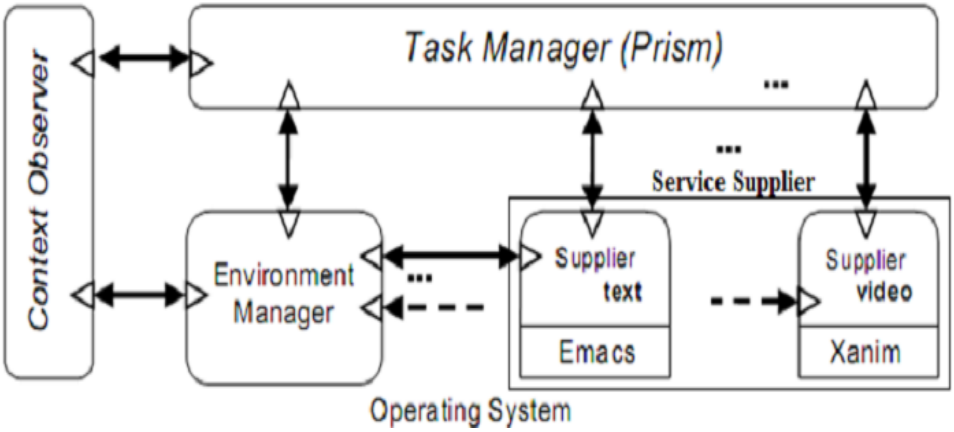

Figure 3.13: Les composants de l'architecture du Framework AURA et leurs interactions [65].

3.4.7.2 L'étude de l'approche AURA selon différents critères

- **Hétérogénéité des objets :** les différentes fonctionnalités d'AURA est encapsulée dans des composants de la structure architecturale (le gestionnaire de tâches, le gestionnaire d'environnement, l'observateur de contexte, fournisseurs de services). Les interactions entre ces parties sont effectuées par des connecteurs explicites qui cachent les détails de la distribution des informations et encapsulent l'hétérogénéité des fournisseurs de services.

- **Mobilité des objets :** AURA a développé une solution pour le problème de la mobilité des utilisateurs basé sur le concept d'Aura personnelle. L'objectif derrière une aura personnelle est qu'elle agit comme un proxy pour l'utilisateur mobile qu'elle représente. L'architecture d'Aura utilise le composant gestionnaire de tâches (Prism) pour incarner le concept d'une aura personnelle. Ce composant permet de contrôler la mobilité des utilisateurs et de s'adapter au changement des contextes ou des taches.

- **Influence du monde physique sur l'Internet des objets:** Les éléments clés d'AURA sont des représentations explicites des tâches de l'utilisateur sous la forme de collections de services, et des observations du contexte. Il permet de configurer la tâche de manière appropriée à l'environnement, et la gestion de monde extérieur qui facilite le suivi et l'adaptation des ressources.

- **Caractérisation multi-échelle:** Elle n'a pas été étudiée dans ce projet.

- **Sécurité :** Elle n'a pas été étudiée dans ce projet

- **Protection de la vie privée :** AURA fournit la protection de la vie privée de l'utilisateur avec un minimum de frais généraux sur Big Data [66].

- **Passage à l'échelle :** Il n'a pas été étudié dans ce projet

- **Interopérabilité :** le middleware d'AURA permet de fournir une interopérabilité et un moyen uniforme de communiquer entre les parties du sa système : le gestionnaire de tâches, l'observateur de contexte, gestionnaire d'environnement, et les fournisseurs. Il existe quatre types de connecteurs dans Aura: entre Prism et un fournisseur, entre Prism et le gestionnaire d'environnement, entre

l'observateur de contexte et Prism, et entre l'observateur de contexte et le gestionnaire d'environnement. Chacun de ces types de connecteurs est défini par un protocole d'interaction approprié au type de composant qu'il connecte [65].

- **Déploiement autonome des entités :** Le composant gestionnaire d'environnement (Environnement Manager) sait quels fournisseurs sont disponibles pour fournir les services et où ils peuvent être déployés. Lorsque le composant gestionnaire de taches (Prism) migre une tâche d'un environnement à un autre, le déploiement de fournisseurs entre les périphériques peut être très différent. De plus, même dans le même environnement, ce déploiement peut changer de façon dynamique, à mesure du changement de l'accessibilité des composants.

- **Qualité de contexte** : Elle n'a pas été étudiée dans ce projet.

- **Cycle de vie de contexte :**
  - Acquisition de contexte: l'observateur de contexte (Context Observer) fournit des informations sur le contexte physique.
  - Modélisation de contexte: Elle n'a pas été citée dans ce projet
  - Raisonnement de contexte: Il n'a pas été cité dans ce projet
  - Distribution de contexte: AURA utilise le mode «requête-réponse» pour diffuser les informations.

3.4.8 Approche SAI : Service Application Integration

3.4.8.1 Description

Est un middleware orienté services (SOA) et orienté Grid proposé en 2010 pour les données hétérogènes distribués et l'intégration de systèmes dans les domaines de la sensibilité au contexte et dans l'IdO. Le système SAI est considéré comme un middleware orienté message prenant en charge le développement d'applications sensible au contexte.

L'objectif est de permettre aux clients de rechercher des données sans aucune connaissance de leur emplacement physique dans l'environnement distribué, sans se préoccuper de la cohérence des données. Ce middleware a été doté d'une infrastructure Grid pour offrir la distribution de travail et l'équilibrage de charge pour le traitement des données de contexte, tout en préservant l'état de sécurité du système et la cohérence des données [67]. La structure logique de l'architecture SAI est représentée dans la figure 3.14.

Il existe cinq types de composants :

- Front-End Portal : est un conteneur qui établit des exigences pour des «modules d'applications». Ces modules peuvent être composés au moment de l'exécution et livrés en tant qu'application web entièrement intégrée et accessibles via un PC ou un téléphone mobile.
- Infrastructure Grid : fournit la distribution de la charge de travail aux applications et aux services de système.
- Basic Back-End Services : ce groupe de composants contient des services offrant des capacités supportant le fonctionnement de l'ensemble de l'infrastructure de SAI.
- Bus de messages (Message Bus) : est une infrastructure qui fournissant des capacités de messagerie au niveau de l'application aux composants du système SAI.
- The Adaptors Framework, permet l'interfaçage de l'SAI avec des systèmes d'information hétérogènes [67].

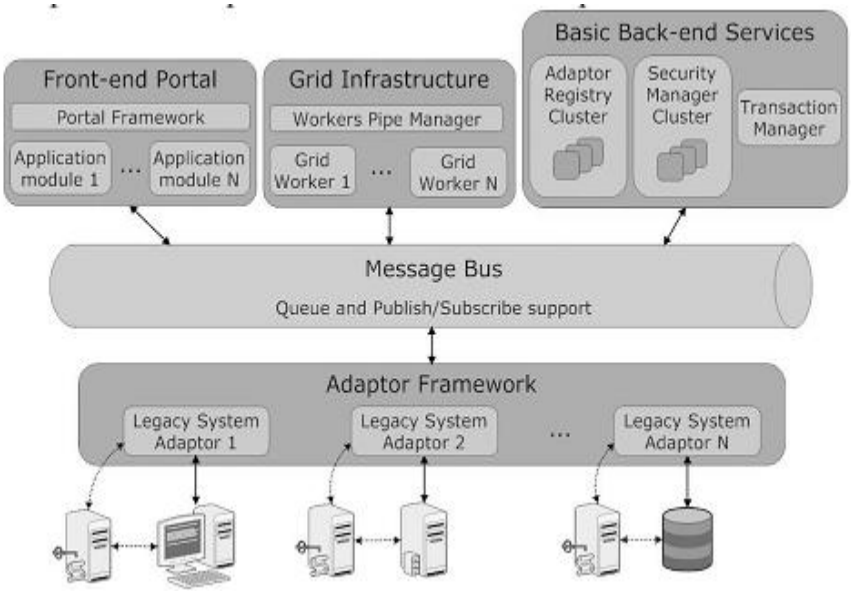

Figure 3.14: Architecture de middleware SAI [67].

3.4.8.2 L'étude de l'approche SAI selon différents critères

- **Hétérogénéité des objets :** l'adoption d'une approche SOA (Service Oriented Architecture) dans SAI permet de faciliter l'intégration de ressources hétérogènes (par exemple, capteurs, actionneurs, systèmes d'information d'entreprise) pour le développement d'applications sensible au contexte.

- **La mobilité des objets :** Un adaptateur (Adapter) de SAI peut être configuré pour écouter les messages de requête sur un protocole de transport réseau changeante à cause de la mobilité de différents objets au cours de temps. Cet adaptateur peut prendre en charge la demande / réponse synchrone, unidirectionnelle asynchrone, la notification-réponse asynchrone pour satisfaire le besoin de applications.

- **Influence du monde physique sur l'Internet des objets:** Elle n'a pas été étudiée dans ce projet

- **Caractérisation multi-échelle:** Elle n'a pas été étudiée dans ce projet.

- **Sécurité et la protection de la vie privée :** le composante de gestionnaire de sécurité (Security Manager) dans le middleware SAI fournis des mécanismes pour valider l'identité des entités du système, pour accorder ou refuser l'autorisation des actions effectuées sur les ressources gérées par SAI, et ainsi pour garantir l'intégrité et la confidentialité des messages.

- **Passage à l'échelle :** La composante « bus de messages » de l'infrastructure SAI fournis des capacités de messagerie au niveau de l'application aux composants du système SAI. Les interfaces SAI sont complètement découplées de toute implémentation concrète de courtier (broker) de messagerie, afin de permettre une flexibilité et un passage à l'échelle maximale du middleware. L'infrastructure de Grid dans SAI fournit la distribution de charge de travail aux applications et aux services de système pour améliorer les performances globales du système.

- **Interopérabilité :** les capacités du middleware SAI permet de développer une plateforme de services qui vise à fournir des services d'information à ces différentes catégories d'utilisateurs. Ce middleware supporté l'interopérabilité entre les organisations concernées et les autorités publics, tout en cachant à ses clients les hétérogénéités des infrastructures technologiques sous-jacentes (par exemple, des capteurs et des actionneurs, et les appareils mobiles).

- **Déploiement autonome des entités :** Il n'a pas été étudié dans ce projet

- **Qualité de contexte** : Elle n'a pas été étudiée dans ce projet.

- **Cycle de vie de contexte :**
  - Acquisition de contexte : Le Framework des adaptateurs dans SAI permet l'interfaçage de SAI avec les sources d'information hétérogènes.
  - Modélisation de contexte: SAI utilise le modèle XML pour modéliser les informations
  - Raisonnement de contexte: pas cité dans ce projet.
  - Distribution de contexte: SAI utilise le mode «publication-souscription» pour diffuser les informations.

3.4.9 Approche WISeMid (Wireless sensor network's and Internet's Services integration Middleware)

3.4.9.1 Description

WISeMid [68] est un middleware d'IdO sensible à l'énergie pour intégrer les réseaux de capteurs sans fil et Internet au niveau de service. Dans ce contexte, les applications exécutées dans les nœuds Internet / WSN peuvent jouer le rôle de fournisseurs de services ou d'utilisateurs de services. L'idée est de fournir une infrastructure qui permet d'intégrer des applications, qui sont considérées comme des services, de manière transparente. Dans un système basé sur l'Internet des objets, il est important d'économiser l'énergie dans l'interaction entre les appareils, car ils ont généralement des fournisseurs d'énergie limités. En outre, le système basé sur l'IdO est une communication basée sur IP.

La structure logique de l'architecture WISeMid est représentée dans la figure 3.15. Il existe trois couches :1) Couche de services communs (Common services layer) : Cette couche contient des services généraux (agrégation de données de capteurs, Regroupement pour définir des clusters dans un WSN, Nommer et stocker les informations requises pour accéder à un service), qui ne sont pas pour un but particulier ou un domaine d'application spécifique.2) Couche de distribution : Cette couche définit les composants requis pour utiliser un service. Par exemple, Requestor est un composant qui effectue une invocation à distance avec des paramètres. WISeMid utilise WISeMid Inter-ORB Protocol (WIOP) pour effectuer interactions requête-réponse. 3) Couche d'infrastructure : Cette couche comprend le gestionnaire de demande de serveur (Server Request Handler) et le gestionnaire de demande de client (Client Request Handler). Ces gestionnaires fournissent une communication réseau pour interagir avec les périphériques [63].

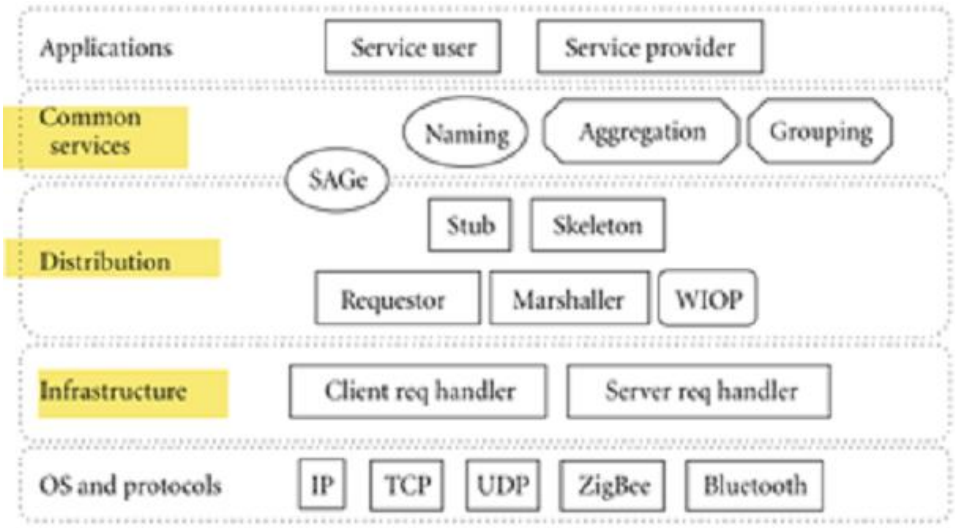

Figure 3.15: Architecture de WISeMid [68].

3.4.9.2 L'étude de l'approche WISeMid selon différents critères

- **Hétérogénéité des objets :** un service offert par un nœud de capteur dans un WSN ou par un hôte sur Internet doit être accessible de manière uniforme, quel que soit le client ou l'emplacement du service. Par conséquent, les développeurs d'applications ont seulement besoin de connaître le nom du service pour accéder à ses opérations. WISeMid prend la responsabilité de cacher l'hétérogénéité de tous les mécanismes de bas niveau du réseau. WISeMid devrait traiter quatre types différents d'hétérogénéité, à savoir le système d'exploitation, le réseau, le matériel et le langage de programmation.

- **La mobilité des objets :** Elle n'a pas été étudiée dans ce projet.

- **Influence du monde physique sur l'Internet des objets:** Il y a deux versions de protocole WIOP: 1) WIOPi prend en charge la communication via Internet. 2) Les WIOPs prennent en charge la communication dans un réseau de capteurs sans fil (WSN) [68]. Ces deux versions de protocole WIOP permet d'éviter le gaspillage d'énergie lors de la transmission radio.

- **Caractérisation multi-échelle:** Elle n'a pas été étudiée dans ce projet.

- **Sécurité et la protection de la vie privée :** Elle n'a pas été étudiée dans ce projet

- **Passage à l'échelle :** Elle n'a pas été citée dans ce projet.

- **Interopérabilité :** WIOP est le protocole d'interopérabilité qui définit les messages de demande / réponse entre les clients et les serveurs. Pour résoudre ce problème, il y a aussi SAGe (Sensor Advanced Gateway) est un élément important de l'architecture WISeMid. Il fonctionne dans l'hôte Internet connecté au nœud récepteur WSN via un port série. La fonction principale de SAGe est de servir de proxy de service entre les deux réseaux en permettant la communication entre les services qui s'exécutent sur les hôtes Internet et les nœuds WSN de manière transparente.

- **Déploiement autonome des entités :** WISeMid se concentre sur l'intégration d'Internet et du réseau de capteurs sans fil au niveau du service en assurant la transparence de l'accès, localisation et de la technologie. Grâce a cette Caractéristique, ce middleware peut fournir une facilité de déploiement, car on n'a pas besoin d'avoir les informations détaillées telles que l'adresse de capteurs pour déployer un service**.**

- **Qualité de contexte** : Elle n'a pas été étudiée dans ce projet.

- **Cycle de vie de contexte**
  - Acquisition de contexte : Couche d'infrastructure fournis une communication réseau pour interagir avec les périphériques pour collecter les données.
  - Modélisation de contexte: Le protocole WIOP définit un format pour les messages de demande ou de réponse entre les clients et les serveurs. Ainsi, ce middleware utilisé WISeMid IDL (langage de définition d'interface) qui permet de définir des interfaces de service de manière uniforme.
  - Raisonnement de contexte: Il n'a pas été cité dans ce projet.
  - Distribution de contexte: WISeMid utilise WISeMid Inter-ORB Protocol (WIOP) pour effectuer des interactions requête / Réponse.

3.5 Comparaison des différentes approches existantes pour la gestion du contexte dans l'IdO

Le tableau 3.1 présente notre étude comparative exhaustive entre les différentes approches existantes basée sur un ensemble de critères bien ciblés. Les critères considérés dans cette étude sont: l'hétérogénéité des objets, l'influence du monde physique, la sécurité, la protection de la vie privée, la mobilité des objets, le passage à l'échelle, la caractérisation multi-échelle, le déploiement autonome des entités, l'interopérabilité, la qualité du contexte, l'acquisition de contexte, la modélisation de contexte, le raisonnement de contexte, la distribution de contexte, la méthode de conception, les outils d'implémentation. Cette étude comparative [69] a été publiée en International Journal of Information Technology (IAJIT) Vol. 14, No. 4A, en 2017.

Nous avons utilisé quelques signes pour faire cette étude :

✓ Ce symbole permet de signifier qu'un défi est résolu par une approche.

- Ce symbole permet de signifier qu'un défi n'est pas résolu par une approche

?? Ce symbole permet de signifier qu'un défi n'est pas mentionné en détail dans le projet

		Context management approaches for IoT								
The comparison criteria	Sub criteria	INCOME Approach [41]	SITAC Approach [51]	FIWARE Approach [55]	CA4IOT Approach [59]	SAMURAI Approach [60]	HYDRA Approach [62]	AURA Approach [65]	SAI Approach [67]	WISeMid Approach [68]
Quality of context	QoC of acquisition phase	✓	-	-	-	✓	-	-	-	-
	QoC of transformation Phase	✓	-	-	-	✓	-	-	-	-
	QoC of de delivery phase	✓	-	-	-	✓	-	-	-	-
Influence of the physical world	detection, correction, error attenuation techniques	✓	✓	-	-	-	✓	-	-	✓
	objects mobiles adaptation	✓	✓	✓	-	-	✓	✓	-	✓
	energy limitation treatment	-	✓	-	-	-	✓	✓	-	✓
Heterogeneity of objects	semantic heterogeneity	✓	✓	✓	✓	✓	✓	✓	✓	✓
	technological heterogeneity	✓	✓	✓	✓	✓	✓	✓	✓	✓
Security	Peers communication	✓	✓	✓	-	✓	✓	-	✓	??
	Topology management	✓	✓	✓	-	✓	✓	-	✓	??
Privacy Protection	Hardware level	✓	??	✓	-	??	✓	✓	✓	??
	Protocol level	✓	??	✓	-	??	✓	✓	✓	??
	Application level	✓	??	✓	-	??	✓	✓	✓	??
Standalone deployment of entities		✓	-	-	-	✓	✓	✓	-	✓
information processing style		QoC at every step of the information life cycle	Semantic interoperability	Performs operations (update, query, notify, registration)	Query-response	Performs request: Create (POST), Read (GET), Update (PUT) Delete (DELETE) events.	Semantic models	Query-response	functional profile for each information	Query-response
Scalability	Scale of IoT	✓	✓	✓	✓	✓	✓	-	✓	??
	data and network communication	✓	✓	✓	✓	✓	✓	-	✓	??
	Management information	✓	✓	✓	✓	✓	✓	✓	✓	??
	Service management	✓	✓	✓	✓	✓	✓	✓	✓	??

Multi-scale characterization		✓	-	-	✓	-	-	-	-	-
Architectural style		Distributed	Distributed	Distributed	Distributed & layered	distributed	Distributed & layered	Distributed	Distributed	Distributed & layered
Context acquisition	Data source support	Any sensors	Any sensors	Any sensors	Any sensors	Any sensors	Any sensors	Any sensors	Any device	Any device
	Acquisition technique	Context collector (pull& push)	??	Context provider	Acquire through a middleware(SDAL)	RESTful APIs	SOAP-based API	Context Observer (CO)	Adaptors Framework	The Infrastructure layer
	Context discovery	✓	??	✓	✓	✓	✓	✓	✓	✓
Context modeling	Modeling Techniques	Ontology based(context information)QoCIM (QoC)	??	Ontology	Ontology	MLContext model	Ontology	??	XML data model	WIOP message format
Context reasoning	Reasoning Techniques	Ontology	??	ontology	Ontology and statistical	MLContext model	Ontology	??	??	??
	Quality of context	Conflict resolution & Validation	??	✓	✓	??	✓	??	✓	??
Context Distribution	Distribution Techniques	Publish and Subscribe	??	Publish and Subscribe	Query	Publish and Subscribe	Query	Query	Publish and Subscribe	Query
Interoperability	Internal	✓	✓	✓	✓	✓	✓	✓	✓	✓
	External	✓	✓	✓	✓	✓	✓	✓	✓	✓
mobility of objects		✓	✓	✓	✓	✓	✓	✓	✓	-
Design Method		Model-Driven Engineering (MuSCa, QoCIM)+ Distributed Event-Based System(DEBS)	Semantic interoperability	Based on Generic Enablers	learning by understanding, maintaining context data in knowledge bases; follows a layered architecture	Multi-Tenant event-based streaming architecture	SOA-based Middleware + semantic-based Model-Driven Architecture	architecture-specific connectors	Based on distributed grid architecture, Follows service oriented architecture (SOA)	Integration WSNs and the Internet at service level(WSN and Internet nodes)
Implementation tools		EMF(Eclipse Modeling Framework)	Social – media based platform	Open stack	OpenIoT software Project	Esper(Java POJOs), GeoSPARQL RESTful APIs	Source Forge + OSGi platform	Wrapped Microsoft Word and Emacs, Media Player and Xanim, Coda or AFS(distributed file systems)	JOTM(an open and standalone implementation of the JTA (Java Transaction API), ActiveMQ	WISeMid Interface Definition Language, WISeMid Inter-ORB Protocol, programming languages (Java, nesC).

Tableau 3.1 : Étude comparative selon différents critères

3.6 Discussion

Les sections précédentes ont présenté une revue sur les solutions de gestion de contexte dans l'IdO. Ces solutions basées sur le découplage entre les participants de l'IdO, à savoir les producteurs d'informations de contexte et les consommateurs de telles informations.

Actuellement, dans la plupart des solutions proposées, les fonctionnalités du middleware sont réalisées en répartissant les tâches dans une architecture en couches / distribuées. Parmi ces solutions: INCOME, SITAC, FIWARE, CAIoT, SAMURAI, HYDRA, AURA, SAI et WISeMid. Seules quatre architectures de middleware prennent en compte la sécurité et la protection de la vie privée en appliquant différentes stratégies, par exemple, les projets INCOME, FIWARE, HYDRA, SAI. Cependant, la sécurité, la protection de la vie privée et la qualité de contexte sont absentes dans les middlewares CAIoT et WISeMid.

Le critère du passage à l'échelle concerne la stabilité du fonctionnement lorsque le nombre d'entités augmente. La plupart des approches étudiées traitent ce défis en utilisant des techniques orientées Cloud. Il devient possible de garantir le bon fonctionnement des applications qui assurent la gestion de la QoC et la confiance entre les participants de l'IdO. Cet aspect est pris en compte dans certains projets comme INCOME et SAMURAI qui ont réussi à résoudre ce problème en proposant une architecture logicielle pour appliquer les contrats de QoC entre les producteurs et les consommateurs de contexte. Chaque solution middleware se concentre sur différents aspects de l'Internet des objets tels que l'interopérabilité, la caractérisation multi-échelle, le déploiement autonome d'entités et bien d'autres.

L'objectif majeur de la plupart des approches est de concevoir une solution pour aider les utilisateurs à automatiser la tâche de sélection des sources de contexte en fonction des tâches à accomplir. Plusieurs défis techniques ont été détectés dans le développement des futures architectures et plus d'efforts devraient être pris en compte pour conduire les propositions de middlewares actuels vers de meilleurs. Le middleware attendu devrait prendre en compte le niveau de la sensibilité de contexte en combinant l'architecture, des modèles et des techniques dans le middleware d'IdO existantes. Une solution middleware générique doit protéger la confidentialité des informations, assurer la sécurité de l'information pendant le traitement ou créer probablement des clouds privés dédiés aux secteurs publics, et garantir la résolution de la QoC.

À partir de cette étude comparative de différentes approches de gestion de contexte dans l'IdO selon plusieurs défis; nous déduisons que la QoC est manquée dans la plupart des solutions logicielles. Prendre en considération la QoC lors de la gestion de contexte est très utile qui mène les applications à des réactions pertinents et des bonnes décisions. Ainsi, le nombre croissant des objets et des données nécessitent une méthode de représentation et de traitement efficace de ces données.

3.7 Conclusion

Dans ce chapitre, nous avons démontré que les nouveaux défis apparus avec l'IdO en termes d'objets hétérogènes à grand échelle, le passage à l'échelle, l'interopérabilité, la sécurité et la protection de la vie privée et autres problèmes nécessitent la conception d'une solution logicielle pour la gestion de contexte multi-échelle. Cette solution commence depuis la production des informations de contexte jusqu'à leurs utilisations par une application. Ce chapitre a présenté les deux grandes parties suivantes:

1. Un aperçu des approches existantes dans la littérature telles que les projets INCOME, SITAC, FIWARE, CA4IoT, SAMURAI, HYDRA, AURA, SAI, WISeMid.
2. Une étude comparative exhaustive entre les approches étudiées basée sur un ensemble de critères bien ciblés.

L’étude comparative de ces travaux est à la base de nos autres contributions qui seront présentées dans le prochain chapitre. En fait, la plupart des travaux étudiés n’offrent aucune solution pour tenir en compte la QoC, qui permet à une application de prendre de bonnes décisions. Le prochain chapitre présente nos contributions pour la gestion de la QoC au sein d’un gestionnaire de contexte.

Chapitre 4 : Le Gestionnaire de Qualité de Contexte Proposé

4.1 Introduction

Dans l'internet des objets (IdO), il existe nombreuses applications qui sont réalisées par les collaborations des objets. Ces applications utilisent des informations de contexte. Il est nécessaire d'offrir des informations de contexte aux applications afin de faciliter leurs adaptations aux environnements. Cela peut être assuré par un middleware. Un tel middleware d'IdO appelé un gestionnaire de contexte. Ce dernier gère une quantité énorme de données hétérogènes générées par différentes sources de contexte, et les transforme à un haut niveau d'abstraction pour fournir aux consommateurs une sortie précieuse.

L'augmentation exponentielle du nombre de sources de contexte qui offrent le même type d'informations pour répondre aux exigences des applications, pose un grand défi dans l'IdO. Donc, Il est devenu important d'utiliser la qualité de contexte (QoC) comme un critère essentiel pour sélectionner la source de contexte qui garantit la qualité requise par une application.

Dans le chapitre précédent, nous avons évalué les approches de gestion de contexte les plus représentatives dans l'IdO en utilisant un ensemble de critères: hétérogénéité, mobilité, influence du monde physique, passage à l'échelle, sécurité, protection de la vie privée, la QoC, déploiement autonome d'entités, caractérisation multi-échelles, interopérabilité, acquisition de contexte, modélisation du contexte, le raisonnement du contexte, la distribution de contexte, la méthode de conception et les outils de mise en œuvre proposés à cet effet. Un défi important comme le critère de QoC manque dans la plupart des approches étudiées.

Quelques travaux ont présenté des revues pour la gestion de la QoC [70, 71, 72, 73, 49, 74, 75, 76, 77].

Dans [49], les auteurs ont proposé un modèle d'information générique, expressif pour la QoC (QoCIM) pour gérer tout critère de QoC dans les gestionnaires de contexte distribués et les applications sensibles au QoC dans l'IdO. QoCIM facilite l'implémentation des fonctions génériques de transformation de QoC. Le but de ces fonctions est simplement de transformer des informations de bas niveau en informations de haut niveau d'abstraction. Cependant, ces fonctions ne reflètent pas le fait de l'incertitude de QoC fournie par différentes sources de contexte. Ainsi, ce travail ne gère pas une solution efficace pour l'augmentation de nombre de données afin d'améliorer le calcul de ces fonctions.

Dans [71, 76, 77], les auteurs ont traité uniquement la catégorisation et la modélisation de la qualité de l'information de contexte pour faciliter la programmation d'applications sensibles au contexte dans les environnements ambiants. Ces travaux n'expriment aucune méthode d'évaluation pour les différents critères de QoC.

Dans [75], les auteurs ont identifié des mécanismes tels que la logique probabiliste, la logique floue et les réseaux bayésiens pour traiter uniquement le critère de l'incertitude. Cependant, cette méthode n'est pas suffisante pour vraiment exprimer la qualité globale de l'information, en fait, il existe différents critères qui représentent entièrement la QoC.

Dans [73, 74], les auteurs ont utilisé l'ontologie pour modéliser le contexte et sa qualité de manière formelle. Cette solution réalise une compréhension sémantique pour les concepts et les relations existant entre eux. Cette méthode est centralisée et nécessite des ressources informatiques et de stockages importants.

Dans [70], les auteurs ont été basés sur l'intégration de la gestion de QoC au sein de la plate-forme COSMOS pour les environnements ambiants. Il s'agit d'un processus basé sur un ensemble de composants logiciels (nœud de contexte, nœud de QoC) organisés en architecture hiérarchique.

Cependant, cette méthode nécessite du temps et des ressources pour créer un nouveau composant d'information de contexte, et sa qualité ou pour en mettre à jour un qui existe déjà. Ce travail ne permet pas une gestion efficace de la qualité du contexte car il traite la QoC comme un composant qui a un "entré-sortie". Il ne gère pas aussi l'imprécision de la description de QoC fournie par différentes sources de contexte.

Dans [72], les auteurs ont présenté une technique pour combiner différentes métriques de QoC pour déduire la valeur de la confiance sur le contexte en utilisant la logique floue. Ce travail ne gère pas une solution pour l'augmentation de nombre de données et la QoC pour améliorer le calcul de leur système d'inférence de confiance.

Dans notre contribution, nous considérons le problème de la sélection de qualité de contexte en fonction des préférences de satisfaction du consommateur. Cependant, le nombre croissant de sources de contexte et de données posent un grand défi dans l'IdO. Pour résoudre ce problème, nous proposons dans ce chapitre un Framework MapReduce Skyline pour accélérer le calcul et introduire un parallélisme pour traiter de grandes quantités de sources de contexte. En outre, une autre proposition dans notre approche est l'utilisation de la logique floue. Cette dernière proposition vise à évaluer la QoC fournie par les sources de contexte et la QoC requise par l'application. Notre approche est proposée pour une raison essentielle : nous appliquons la logique floue à un petit nombre de sources de contexte résultant de l'opération MapReduce Skyline plutôt qu'un grand nombre. Par conséquent, cette opération sélectionne les sources de contexte optimales à partir d'un grand nombre de sources de contexte en réduisant le temps et le coût de calcul.

4.2 Description de l'approche

Notre approche montre que le Middleware basé sur la logique floue, le Map reduce skyline sont réalisable et nécessaires pour supporter l'évaluation de la qualité de contexte (QoC) dans l'IdO.

Cette thèse ne vise pas à fournir une solution logicielle complète qui répond à toutes les exigences nécessaires pour un middleware d'IdO. Nous nous concentrons plutôt sur un seul problème qui gère les informations de QoC. L'architecture proposée est distribuée avec des composants pour la collecte et le traitement des informations de contexte. Notre gestionnaire de contexte se concentre sur le traitement de contexte en supposant que toutes les données de contexte sont produites par différentes sources de contexte. Par conséquent, l'architecture du gestionnaire de qualité de contexte proposé est illustrée à la figure 4.1. Ce gestionnaire ne visait qu'à résoudre ce problème de QoC.

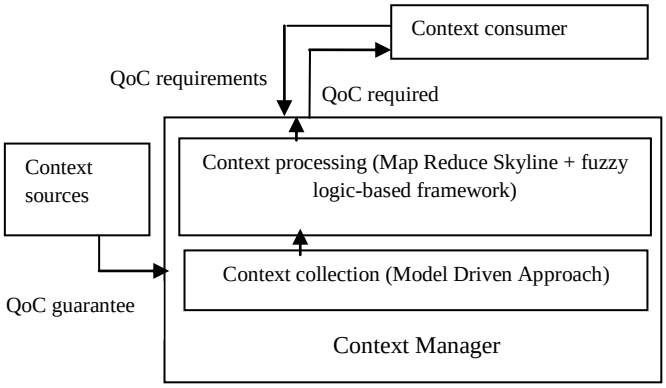

Figure 4.1: Architecture de notre gestionnaire de QoC

Le gestionnaire de QoC se compose de deux couches :

1. Couche de collection de contexte : Cette couche apparaît dans la plupart des solutions de middleware de gestion de contexte pour l'IdO avec des terminologies différentes. Ce middleware est responsable de la collection des informations de contexte de bas niveau à partir de diverses sources disponibles [24]. Ces informations de contexte et les caractéristiques des sources de contexte doivent être modélisées dans cette couche. Un modèle à ce niveau s'appliquera à une source de contexte spécifique pour fournir une solution de modélisation générique pour exprimer un fournisseur de contexte en utilisant l'ingénierie dirigé par les modèles (IDM).
L'approche IDM vise à séparer la conception en deux niveaux [44]:

- Au niveau supérieur : Platform Independent Model (PIM) qui décrit l'architecture logicielle du système, mais sans prendre en compte les spécificités d'une plate-forme particulière.
- À un niveau inférieur : Platform Specific Model (PSM) est une extension du PIM avec des concepts spécifiques à une plate-forme cible. Dans cette couche, la transformation de modèles (Model To Text (M2T)) extrait le code source à partir d'un modèle PSM.

2. Couche de traitement du contexte : Il est responsable de l'extraction des informations de contexte à partir de la couche de collection de contexte [24]. Nous avons développé au sein de cette couche un programme parallèle pour traiter de grandes quantités des données afin d'améliorer l'efficacité en utilisant le paradigme MapReduce Skyline. Ensuite, nous avons appliqué la logique floue sur la liste de résultats finale générée par MapReduce Skyline.

- **Côté Application** : Initialement, l'utilisateur final demande au gestionnaire de contexte de rechercher la liste des sources de contexte disponibles pour satisfaire son besoin. Les besoins des utilisateurs sont généralement non quantifiables. Ces besoins peuvent être capturés en fonction de variables linguistiques.
  La QoC pour la demande d'application est définie par deux éléments :
  - Critères de qualité : sont les critères de qualité dans la requête d'application.
  - La valeur de la qualité de contexte : le besoin d'un demandeur concernant le paramètre de qualité repose sur des termes linguistiques.
- **Côté source de contexte** : La garantie de la QoC pour la source de contexte est définie par deux éléments:
  - Le critère de qualité pour la garantie de source de contexte.
  - Le niveau de qualité des données que le producteur est en mesure de fournir. Ce niveau de qualité prend une valeur entre [0, 1].

4.3 Modélisation de source de contexte au niveau couche de collection

4.3.1 Motivation

La modélisation de source de contexte permet de définir, pour chaque source, les contextes qui ont une influence sur cette source et les interactions entre le contexte et la source de contexte. Par conséquent, il est intéressant d'avoir un support de modélisation commun. Le cœur de modélisation doit être ouvert pour que des extensions puissent être ajoutées pour différentes sources de contexte. Il est ainsi nécessaire que le modèle de contexte puisse évoluer pendant le cycle de vie de source de contexte. Donc, l'approche adoptée consiste à intégrer dans le processus de modélisation de source de contexte une phase de définition de contexte et les caractéristiques de cette source en utilisant l'ingénierie dirigée par les modèles. Cette approche peut être considérée comme un bon choix en ce qui concerne le traitement des limites des approches actuelles de modélisation du contexte. Les

informations contextuelles, y compris la QoC, pourraient donc être représentées à un niveau élevé d'abstraction,

4.3.2 Le méta-modèle proposé pour les sources de contexte

Dans le but de prendre en charge la diversité des sources de contexte existantes dans l'IdO, nous proposons une solution basée sur la méta-modélisation. Elle fournit un support unifié aux développeurs pour modéliser tout type de source de contexte. Dans la figure 4.2, nous décrivons notre méta-modèle utilisé pour modéliser la source de contexte. Le cœur de ce méta-modèle est composé de cinq classes : Context Source, Context Information, QoC Criteria, QoC Value, QoC Criteria Definition.

1. La classe Context Source est définie par quelques champs : Le champ *id* identifie de manière unique chaque source de contexte modélisée dans ce modèle. Ainsi, le champ *Type* qui désigne le type de cette source (capteur ou réseau social ou application. etc). Et le champ *Characteristic* qui rapporte les différentes caractéristiques de cette source de contexte.
2. La modélisation des informations de contexte, avec la classe Context Information. Cette classe possède le champ *id* qui identifie de manière unique l'information, *Content* qui représente le type de l'information, *Date* qui contient la date de création de l'information et le type de source qui fournit cette information.
3. L'évaluation de la qualité d'une information de contexte par différents critères de la QoC est représentée par la classe QoC Criteria. Cette classe joue un rôle important car elle permet d'évaluer la qualité de l'information selon des critères bien définis. Elle permet aussi d'identifier de manière unique le critère qui est modélisé par le champ *id*.
4. La valeur d'un critère de QoC est représentée par la classe QoC Value. Le champ *Value* de cette classe désigne la valeur d'un critère de qualité de contexte. Ce champ peut être utilisé par les applications pour sélectionner parmi plusieurs sources de contexte celui le plus approprié au besoin d'utilisateurs.
5. La classe QoC Criteria Definition permet de définir chaque critère de QoC. Le champ *id* dans cette classe identifie de manière unique chaque définition de critère modélisé dans ce modèle. Le champ *value* indique les valeurs possibles pour ce critère. Les champs *Min value* et *Max value* désignent respectivement la valeur minimale et maximale autorisée pour ce critère. Le champ *Critical Value* désigne les limites critiques pour ce critère. Ce champ vise à alerter à certaines situations, si nécessaire.

La classe Context Source permet de représenter la source de contexte. Elle peut fournir une ou plusieurs informations de contexte représentées par la classe Context Information. À l'inverse, logiquement l'information de contexte peut être obtenue par différents sources de contexte. Cette information peut qualifier selon différents paramètres de qualité de contexte par la classe QoC Criteria. Cette classe représente un critère de QoC, et qui est liée à une définition bien ciblée, représentées par la classe QoC Criteria Definition. La classe QoC Criteria est liée ainsi à la classe QoC value où une valeur sera calculée à partir aux caractéristiques de source de contexte. La valeur de critère de QoC est représentée par la classe QoC Value.

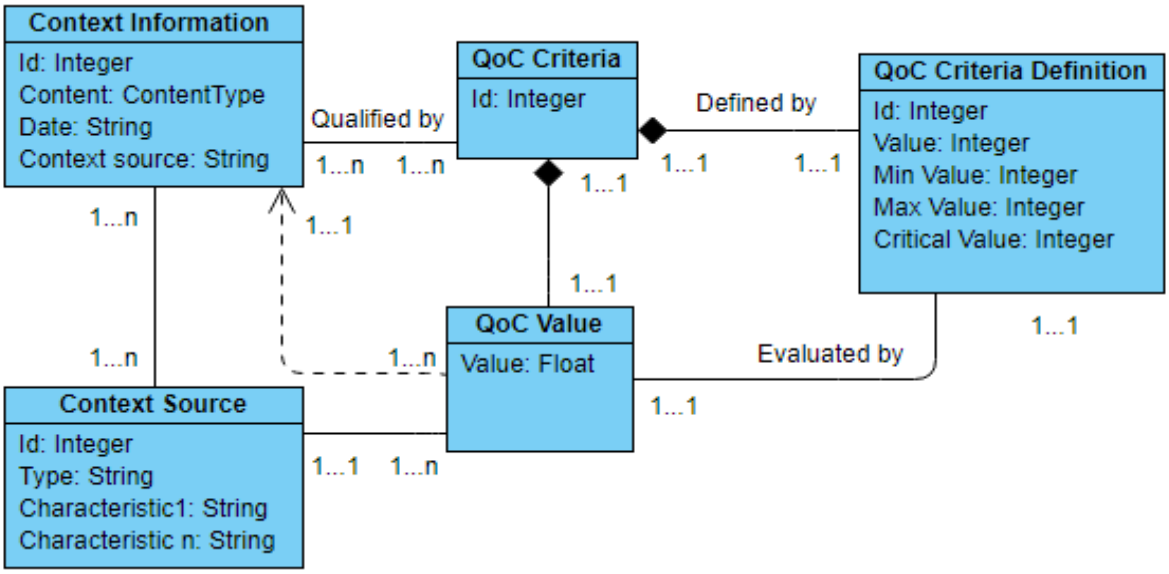

Figure 4.2: Le méta-modèle de source de contexte

4.4 Sélection de la qualité de contexte au niveau couche de traitement basée sur la logique floue et MapReduce Skyline

4.4.1 Motivation d'utiliser la logique floue et le MapReduce Skyline

1. Motivation d'utiliser la logique floue

L'IdO permet la collecte d'une grande variété de données de contexte provenant de différentes sources de contexte. Ces données de contexte peuvent ensuite être exploitées par les applications pour détecter la situation actuelle des utilisateurs et leur fournissent les services pertinents correspondant à leurs besoins. Cependant, ces données de contexte sont connues pour être imparfaites et incertaines par nature. L'incertitude des données de contexte est un défi crucial pour permettre aux applications d'avoir des réactions pertinentes. Un moyen de limiter cette incertitude est de manipuler des informations supplémentaires associées aux données de contexte sous la forme de métadonnées ce qu'on appelle la qualité de contexte (QoC). L'utilisation de la technique de logique floue permet de sélectionner la source de contexte approprié en considérant les exigences non fonctionnelles de l'application. Cette méthode évalue des contraintes et des préférences imprécises de QoC.

L'utilisation de la logique floue est motivée par sa méthodologie de résolution de problèmes. Elle peut être mise en œuvre sur n'importe quel système de gestion indépendamment de la taille et de la complexité du problème. Cette solution est appropriée pour être utilisée dans la représentation d'une description d'attribut non fonctionnelle. Il permet de raisonner non pas sur des variables numériques, mais sur des variables linguistiques, c'est-à-dire, sur des variables qualitatives. Le fait de raisonner sur ces variables linguistiques va permettre de pouvoir manipuler des connaissances en langage naturel. Cela ressemble à un raisonnement humain pour l'utilisation d'informations approximatives fournies par différentes sources contextuelles [81]. La logique floue est établie de manière à pouvoir traiter des variables inexactes de valeurs. Il peut donc traiter efficacement le flou et l'imprécision de la description de QoC tant du côté du consommateur que du côté du fournisseur de contexte : Les ensembles flous sont adaptés pour spécifier à la fois les exigences de QoC demandée par une application et une estimation approximative fournie par un producteur de contexte. La logique floue a été utilisée pour aider le consommateur à choisir la QoC la plus adaptée en donnant des poids aux

sources de contexte disponibles. Ces poids peuvent être tirés comme des conclusions à partir d'un raisonnement sur un système flou (une base de règle) exprimée en langage naturel.

2. Motivation d'utiliser MapReduce Skyline

Le nombre croissant des sources de contexte dans l'environnement d'IdO qui fournissent la même fonctionnalité mais diffèrent en termes des paramètres de QoC, mène vers un problème de gestion de QoC. Ce problème est transformé à la sélection de meilleures sources de contexte candidats en ce qui concerne la QoC requie par l'application.

Cette étude considère le problème de la sélection d'un petit nombre de sources de contexte important dans l'environnement d'IdO. Basé sur certains critères de QoC à partir de besoin de l'application, le calcul Skyline sélectionne des sources de contexte qui ne sont pas dominées par d'autres. En raison de la structure complexe de l'IdO, la sélection d'une source de contexte la plus adéquate au besoin d'application est plus compliquée. Nous devons ainsi prendre en compte divers critères de qualité de contexte. En outre, l'IdO contient des données massives, et l'augmentation des données est énorme. La méthode de skyline est adoptée pour réduire l'espace de recherche dans le but d'améliorer l'efficacité de la gestion de QoC.

Inspiré par le traitement parallèle distribué, nous proposons d'appliquer la technologie MapReduce pour améliorer l'efficacité avec des performances évolutives dans la sélection de sources de contexte à grande échelle. Le Framework MapReduce vise à accélérer le calcul et introduire le parallélisme dans le traitement.

4.4.2 Définition de la logique floue

La logique floue a été introduite par Lotfi Zadeh en 1965 de l'université de Californie de Berkeley. La logique conventionnelle basée sur qu'un ordinateur peut comprendre et prend une entrée précise. Il produit un résultat défini comme vrai ou faux, ce qui équivaut à oui ou non de l'humain. L'inventeur de la logique floue, Lotfi Zadeh, a observé que contrairement aux ordinateurs, la prise de décision humaine comprend une gamme de possibilités entre oui et non. Ce concept vise à fournir des modèles approximatifs pour des systèmes incertains dont la modélisation avec des approches mathématiques précises est très difficile. Le système d'inférence floue basée sur la base des règles d'inférence floue peuvent être utilisées pour surmonter l'utilisation des données mathématiques ou des structures algorithmiques complexes dans la modélisation du système. Les entrées et les sorties d'un système d'inférence floue sont fournies dans une présentation linguistique et ont des valeurs linguistiques telles que faible, moyen et grand. Les règles d'inférence floue sont un ensemble de règles Si-Alors qui peut décider en fonction des valeurs d'entrée formelles et donner des valeurs de sortie [82].

Le principe d'un système flou, c'est de pouvoir calculer des paramètres de sorties en fournissant au système un ensemble de règles formulés en langage naturel. Un système flou est composé en trois étapes consécutives :

- La fuzzification est la première étape pour appliquer un système d'inférence floue. La plupart des variables existant dans le monde réel sont des variables classiques. Il traduire ces variables classiques en des variables linguistiques. Grâce à une fonction d'appartenance créée par le concepteur du système flou, on va pouvoir transformer une donnée quantitative en variable linguistique qualitative. Généralement, la fuzzification implique deux processus: dériver les fonctions d'appartenance pour les variables d'entrée et de sortie et les représenter avec des variables linguistiques. Ce processus équivaut à convertir ou mettre en correspondance un ensemble classique avec un ensemble flou à des degrés divers. Les fonctions d'appartenance peuvent avoir

plusieurs types différents, tels que la forme triangulaire, trapézoïdale, ou gaussienne.etc. Le type exact de fonction d'appartenance dépend sur des applications réelles. Pour les systèmes qui nécessitent une variation dynamique significative sur une période courte de temps, une forme triangulaire ou trapézoïdale doit être utilisée. Pour les systèmes nécessitant une précision de contrôle très élevée, une forme gaussienne doit être choisie [83].

- Le moteur d'inférence : chargée d'appliquer des règles d'inférences dû à l'expertise humaine. La règle floue est représentée par une séquence de la forme SI ALORS. Il conduisant à des algorithmes décrivant quelle action ou sortie doit être prise en termes d'information actuellement observée. Une règle floue SI ALORS associe à une condition décrite en utilisant des variables linguistiques et des ensembles flous à une conclusion. La partie SI est principalement utilisée pour capturer les connaissances en utilisant des conditions, et la partie ALORS peut être utilisée pour donner la conclusion sous forme de variable linguistique.

- La Défuzzification : Pour rendre les conclusions des règles d'inférences disponibles pour des applications réelles, un processus de défuzzification est nécessaire. La conclusion ou la sortie floue est toujours une variable linguistique, et cette variable doit être convertie en variable classique via le processus de défuzzification. Il permettant de fusionner les différentes commandes générés par le moteur d'inférence pour le donner qu'une seule commande de sortie et de transformer cette variable linguistique de sortie en donnée classique. Trois techniques de défuzzification sont couramment utilisées, à savoir: la méthode Mean of Maximum, la méthode Center of Gravity et la méthode Height [84].

4.4.3 MapReduce Skyline

4.4.3.1 Définition MapReduce

Les systèmes d'entreprise traditionnels ont un serveur centralisé pour stocker et traiter les données. Le modèle traditionnel n'est pas adapté pour traiter d'énormes volumes de données simultanément et ne peut pas être pris en charge par les serveurs de données standard. Le modèle de programmation MapReduce introduit par Google en 2004. Il permet à résoudre ce défi en traitant un grand ensemble de données de manière massivement parallèle. MapReduce divise une tâche en petites parties et les affecte à plusieurs ordinateurs. Plus tard, les résultats sont collectés et intégrés pour former le résultat de l'ensemble de données.

MapReduce (Dean & Ghemawat, 2008) est un outil populaire pour le traitement distribué et évolutif des données volumineuses. Il est de plus en plus utilisé dans différentes applications principalement en raison de ses caractéristiques importantes, y compris l'évolutivité, la tolérance aux pannes, la facilité de programmation et la flexibilité [85]. MapReduce est un modèle de programmation et une implémentation associée pour le traitement et la génération de grands ensembles de données.

Les utilisateurs spécifient une fonction Map qui prend une paire d'entrées et produit un ensemble de paires clé / valeur intermédiaires. La bibliothèque MapReduce regroupe toutes les valeurs intermédiaires associées à la même clé intermédiaire et les transmet à la fonction Reduce. La fonction Reduce fusionne toutes les valeurs intermédiaires associées à la même clé intermédiaire [86].

La qualité de l'environnement de l'internet des objets peut être grandement améliorée par le traitement plus rapide de MapReduce. Le modèle de MapReduce accélère le processus de sélection de Skyline en explorant le parallélisme distribué.

4.4.3.2 Définition de calcul Skyline

Börzsönyi et al. [87] sont les premiers à avoir abordé la problématique du calcul des Skylines dans le contexte des bases de données. Ils ont proposé d'étendre les systèmes de bases de données par une opération Skyline. Cette opération filtre un ensemble de points intéressants à partir d'un ensemble potentiellement important de points de données.

Le calcul Skyline a attiré beaucoup d'attention au cours des dernières années. Il a été largement appliqué dans les environnements distribués et mobiles. Cette méthode aide les utilisateurs à prendre des décisions intelligentes sur des données complexes, où des nombreux critères contradictoires sont considérés [88]. Le Skyline est utiles pour extraire des points intéressants à partir d'un grand ensemble de données selon plusieurs critères [89]. Un point est considéré comme intéressant, s'il n'y a pas d'autre point mieux que cela dans tous les critères d'évaluation.

4.4.3.3 Définition de la relation de dominance

Un concept nécessaire à la définition formelle des Skylines est la relation de dominance.

Définition (Relation de dominance) : On dit qu'un point $p \in P$ domine un autre point $q \in P$, noté $p \prec q$, si (1) sur chaque dimension $d_i \in D$, $p_i \leq q_i$; et (2) sur au moins une dimension $dj \in D$, $p_j < q_j$, où P représente l'ensemble de données sur l'espace de dimensions D.

Donc, un ensemble de points $SKY(P) \subseteq P$ qui ne sont dominés par aucun autre point de P. Les points de SKY(P) sont appelés points de Skyline.

Supposons que on a un ensemble de N serveurs S_i participant au calcul parallèle. L'ensemble de données P est distribué aux partitions en utilisant une technique de partitionnement de l'espace, telle que P_i est l'ensemble des points stockés par le serveur S_i.

Chaque serveur S_i calcule localement la liste des points Skylines (les Skylines locaux) en se basant sur les points P_i stockés localement. Ces Skylines locaux sont fusionnées pour obtenir un ensemble de Skylines globaux [90].

4.4.3.4 Étude de différentes méthodes de partitionnement Skyline

Un défi important est la façon de partitionner l'ensemble de données sur les N serveurs, afin d'augmenter l'efficacité du traitement de calcul Skyline. Des techniques de partitionnement de l'espace, telles que la division récursive de l'espace de données, ont été utilisées pour le traitement des Skyline dans des paramètres parallèles et distribués.

Nous allons présenter les méthodes de Skyline basées sur MapReduce pour partitionner l'espace de données qui ont été utilisées pour les calculs de Skyline dans la littérature.

4.4.3.4.1 Méthode de partitionnement aléatoire

Une approche simple pour partitionner un ensemble de données parmi un certain nombre de serveurs consiste à choisir au hasard l'un des serveurs pour chaque point. Cette méthode partitionne les points de données de manière aléatoire parmi les serveurs (chaque partition contient un échantillon aléatoire des données). Le schéma de partitionnement aléatoire est facile à implémenter et n'ajoute aucun surcoût de calcul pour décider dans quelle partition chaque point appartenu.

Le principal inconvénient de cette approche est que la taille des ensembles de résultats Skylines locaux n'est pas minimisée. Nombreux points appartenant aux ensembles de Skylines locaux n'appartiennent pas aux ensembles de résultats Skylines globaux. En général, il n'y a aucun moyen d'atténuer ce défi afin de réduire le coût de transfert des Skylines locaux à partir des serveurs au

coordinateur. Ainsi, la performance du partitionnement aléatoire se dégrade de manière significative dans le cas de distributions de données spécifiques. Si l'ensemble de données est anti-corrélé, le partitionnement aléatoire dispatché le coût élevé de traitement à toutes les partitions [90].

4.4.3.4.2 Méthode de partitionnement par grille (Grid Partitioning)

Le schéma de partitionnement basé sur la grille est basé sur une division récursive des dimensions de l'espace de données. Si l'espace de données a deux dimensions, le résultat du partitionnement basé sur la grille est constitué de partitions de forme rectangulaire. Afin de définir les limites des partitions dans chaque dimension, l'équation $x = \sqrt[d]{N}$ est utilisée, où x est le nombre de limites dans chaque dimension, d est le nombre de dimensions et N est le nombre désiré de partitions. Dans ce type de partitionnement, chaque serveur est responsable d'une seule partition.

Le principal avantage de cette approche est que chaque partition a approximativement la même cardinalité de points de données, donc la charge de travail de chaque serveur est équilibrée. De plus, chaque partition a la même distribution de données et calcule les points Skylines locaux de manière proportionnelle. Grâce à une communication partiellement sérialisée entre les serveurs dans cette méthode, ils garantissent que les bons résultats sont renvoyés. Il permet d'éviter de contacter tous les serveurs, d'autant plus que plusieurs ne contribuent pas au résultat final. Cependant, cette technique de partitionnement présente plusieurs inconvénients. Tout d'abord, le serveur correspondant à l'origine des axes contribue le plus à l'ensemble de résultats, alors que plusieurs autres, en particulier ceux éloignés à des axes, ne contribuent pas du tout. En outre, chaque serveur renvoie une quantité égale des Skylines locaux, mais la plupart d'entre eux ne contribuent pas au résultat de Skylines globaux. En conséquence, la phase de fusion possède une charge de travail redondante et le coût de communication n'est pas minimisé [91].

4.4.3.4.3 Méthode de partitionnement par l'angle (Angle Partitioning)

Les auteurs ont proposé une approche de partitionnement de l'espace basée sur l'angle. Il utilise des coordonnées hyper-sphériques afin de partitionner l'espace de manière à augmenter l'efficacité du traitement des requêtes de Skyline distribuées parallèles. Cette méthode transforme l'espace de coordonnées cartésiennes dans un espace de coordonnées hyper-sphérique et applique un partitionnement basé sur les coordonnées angulaires [92].

Les coordonnées cartésiennes d'un point de données $x = [x_1, x_2, ..., x_d]$, où x_i désigne la valeur de x dans la i ème dimension de l'attribut et d indique la dimension de l'espace cartésien, sont mappées à des coordonnées hyper-sphériques. Ces coordonnées hyper-sphériques sont constitués d'une coordonnée radiale r et (d-1) coordonnées angulaires $\varphi_1, \varphi_2, ..., \varphi_{d-1}$, en utilisant l'ensemble d'équations 1 et 2 respectivement :

$$r = \sqrt{x_n^2 + x_{n-1}^2 + \cdots x_1^2} \quad (1)$$

$$\tan(\varphi_1) = \frac{\sqrt{x_n^2+x_{n-1}^2+\cdots+x_2^2}}{x_1}, \ldots\ldots, \tan(\varphi_{d-2}) = \frac{\sqrt{x_n^2+x_{n-1}^2}}{x_{n-2}}, \ldots, \tan(\varphi_{d-1}) = \frac{x_n}{x_{n-1}} \quad (2)$$

Étant donné le nombre de partitions N et un espace de données D de d dimensions, le partitionnement angulaire attribue à chaque partition une partie de l'espace de données D_i ($1 <= i <= N$). L'espace de données de la i éme partition est défini par l'équation 3 comme suit:

$$D_i = [\varphi_1^{i-1}, \varphi_1^i], \times \ldots \times, \varphi_{d-1}^{i-1}, \varphi_{d-1}^i \quad (3)$$

Où φ_j^0=0 et $\varphi_j^N = \frac{\pi}{2}$ (1<= j <= d), alors que φ_j^{i-1}et φ_j^i sont les limites de la coordonnée angulaire φ_j pour la partition i [89].

Après avoir assigné à chaque partition une partie de l'espace de données D_i, nous pouvons facilement distribuer les points de données à ces partitions. D'abord, les coordonnées cartésiennes de chaque point sont mappées aux coordonnées hyper-sphériques. Ensuite, les d-1 coordonnées angulaires sont comparées aux limites de chaque partition et la partition correspondante est trouvée.

Dans ce schéma de partitionnement, toutes les partitions partagent la zone proche de l'origine des axes. Ceci est important car cela augmente la probabilité que les points de Skylines globaux qui existent près de l'origine des axes soient répartis de manière égale entre les partitions. Le partitionnement basé sur l'angle ne devrait renvoyer que quelques Skylines locaux. Par conséquent, la réalisation d'un petit ensemble de résultats locaux entraîne des coûts de communication réseau plus faibles et des coûts de traitement réduite pour la phase de fusion. Mais, le coût de traitement pour déterminer les limites de partition n'est pas minimisé. La charge de travail dans cette méthode est répartie uniformément entre tous les serveurs disponibles. Il dépend de limites de chaque partition, qui influencent à la performance globale du calcul Skyline parallèle [91].

La méthode de partitionnement proposée dans notre approche vise à classer les sources de contexte en des catégories. Cette catégorisation est conçue à partir de la grande diversité observée dans les sources de contexte dans l'IdO. Cette méthode est plus réaliste et n'ajoute aucun surcoût de calcul pour décider dans quelle partition chaque point appartenu. Notre méthode est simple car elle consiste de partitionner l'ensemble des sources de contexte selon son type.

4.4.3.5 Étude de différents algorithmes existants pour le calcul Skyline

Un certain nombre d'algorithmes ont été proposés dans la littérature pour répondre aux calculs Skyline en présence de larges volumes de données. Les travaux les plus populaires peuvent être résumés comme suit :

4.4.3.5.1 Block Nested Loops (BNL)

Börzsönyi et al. [87] ont proposé le premier algorithme de recherche de Skyline Block Nested Loop (BNL). L'algorithme BNL vise à comparer tous les points de l'ensemble de données deux à deux et de retourner comme Skyline tous les points qui ne sont dominés par aucun autre.

L'idée de cet algorithme est de balayer le fichier de données et garder une liste des points skylines candidats dans la mémoire central. Quand un point p est lu comme une entrée, p est comparé à tous les points de la mémoire centrale et, en fonction de cette comparaison, p est éliminé, placé dans la mémoire centrale ou dans un fichier temporaire qui sera considéré dans l'itération suivante de l'algorithme [93]. Lors de la comparaison d'un point p avec les points stockés en mémoire centrale, trois cas se produisent :

- p est dominé par un point dans la mémoire centrale. Dans ce cas, p est éliminé et ne sera pas considéré dans les prochaines itérations.
- p domine un ou plusieurs points dans la mémoire centrale. Dans ce cas, ces points sont éliminés ; c'est-à-dire que ces points sont supprimés de la mémoire centrale et ne seront pas prises en compte dans les prochaines itérations. P est inséré dans la mémoire centrale.
- p est incomparable avec tous les points dans la mémoire centrale. S'il y a assez de place dans la mémoire centrale, p est inséré dans la mémoire centrale. Sinon, p est écrit dans un fichier temporaire. Les points du fichier temporaire seront traités dans l'itération suivante de l'algorithme.

Dans le meilleur des cas, l'algorithme se termine après une seule itération avec une complexité O(n) (n représentant le nombre de points dans la base de données). Alors que dans le pire cas, sa complexité est quadratique O (n^2) [94,95].

4.4.3.5.2 Bitmap

Contrairement à la plupart des algorithmes existants qui nécessitent au moins un passage sur l'ensemble de données pour retourner le premier point intéressant, l'algorithme Bitmap renvoient progressivement les points intéressants qu'il n'a pas besoin d'analyser l'ensemble de données complet. Kian-Lee Tan et al [93] ont proposé l'algorithme bitmap pour résoudre le problème du calcul skyline progressive. Cet algorithme est complètement non-bloquant et exploite une structure bitmap pour identifier rapidement si un point est un point intéressant ou non. Bitmap code toutes les données dans une structure bitmap afin d'identifier les points skylines. L'algorithme Bitmap convertit chaque point p de l'ensemble de données en un vecteur de bits, représentant le nombre de points ayant une plus petite coordonnée que p dans chaque dimension. La longueur du vecteur de bits est déterminé par le nombre de valeurs distinctes sur toutes les dimensions. Les points Skylines sont alors obtenus en utilisant uniquement des opérations binaires sur les vecteurs à bits, permettant une optimisation considérable des temps de calcul des Skylines. De plus, l'algorithme Bitmap retourne les points Skylines au fur et à mesure de leurs calculs [96].

Les bitmaps fonctionnent bien lorsque le nombre de valeurs distinctes pour chaque dimension est petit. Même si Bitmap est un algorithme progressif, il doit considérer tous les points de l'ensemble de données afin de calculer le Skyline complet. Il tend à être une opération coûteuse parce que pour chaque point inspecté doit récupérer les bitmaps de tous les points [97].

4.4.3.5.3 Nearest Neighbor (NN)

L'algorithme Nearest Neighbor (NN) [98] est le premier algorithme qui utilise la structure d'index R*-tree afin d'éliminer massivement des points en évitant les contrôles de dominance redondants. L'algorithme NN est basé sur des requêtes de plus proche voisinage, utilisant une distance donnée (e.g. distance de Manhattan). Ces requêtes sont implémentées à l'aide d'index de type R /*R_-Tree. Cet algorithme calcule d'abord le voisin le plus proche de l'origine du repère, qui est garanti pour faire partie de l'ensemble des résultats skylines. L'origine du repère représentant forcément un point Skyline puisqu'aucun autre point n'a pu le dominer. NN segmente le reste des points en régions qui se chevauchent en appliquant la technique de recherche du plus proche voisin sur le dernier point extrait. De toute évidence, la région dominée par le voisin le plus proche peut être élagué de considération. En regardant répétitivement pour le voisin le plus proche dans les régions non dominées, le skyline complet est déterminé. Un appel récursif de l'algorithme NN se termine lorsque la requête du plus proche voisin correspondant ne récupère aucun point dans l'espace correspondant [99].

Cette algorithme a des entrées et des sorties importants, en particulier dans les espaces de grande dimension, en raison de l'accès récurrent du R*tree. Globalement, l'algorithme NN est un bon algorithme pour le calcul des Skylines à faible dimensions [91].

Nous avons adopté vers l'algorithme BNL dans notre approche à cause de sa grande applicabilité. Il peut fonctionner pour tous les types de domaines. L'algorithme BNL peut être appliqué sans indexation ou de tri du fichier de l'ensemble de données.

4.4.4 Description de l'approche proposée

4.4.4.1 Les méthodes d'évaluation choisies pour les métriques de QoC

Nous commençons d'abord d'évaluer les qualités des sources de contexte selon différents critères (précision, accuracy, complétude, etc.) à partir des propriétés de chaque source de contexte. Chaque critère de QoC prend une valeur comprise entre [0, 1]. Nous introduisons quelques recherches utilisées pour calculer les métriques de QoC en fonction du type de source de contexte (les capteurs physiques, Réseaux sociaux, Composant de Service web, Composant logiciel) comme suit dans le tableau 4.1:

Sources de contexte	**Méthodes d'évaluation pour les métriques de QoC**
Les capteurs physiques [72, 100]	Atif Manzoor et al. (2009) se concentrent sur l'évaluation des métriques de QoC pour les capteurs en utilisant de nombreuses équations en fonction des propriétés des capteurs.
Réseaux sociaux [101]	Parina Alamir (2016) propose une méthode pour évaluer les valeurs de confiance dans les réseaux sociaux basée sur l'assurance de QoS et l'historique des appels entre les utilisateurs de réseau social.
Composant de Service web [102]	Kuyoro Shade O (2012) a décrit les caractéristiques non fonctionnelles des services Web. La qualité des informations fournies par le service web correspond à la qualité de service QoS.
Composant logiciel [103]	Arun Sharma et al. (2008) utilise le processus de hiérarchie analytique (AHP) pour attribuer les valeurs de poids aux caractéristiques du modèle proposé. Ces valeurs de poids sont ensuite utilisées pour évaluer la contribution de qualité des sous-critères et critères. Enfin, ces valeurs de poids seront utilisées ainsi pour calculer la qualité globale du composant logiciel en utilisant les métriques appropriées.

Tableau 4.1 : Méthodes d'évaluation pour les métriques de QoC.

- Pour la première catégorie (les capteurs physiques) : Les caractéristiques des capteurs sont des propriétés différentes des capteurs utilisés pour évaluer les différents critères de QoC. Les caractéristiques d'un capteur comprennent des informations sur les capteurs physiques qui peuvent affecter la qualité des informations fournies par ceux-ci. Il peut y avoir différentes façons de calculer les valeurs des caractéristiques pour les capteurs physiques. Ces caractéristiques incluent les informations sur la precision, accuracy, resolution, Completeness. etc [100].
  Il existe plusieurs efforts pour évaluer la qualité de l'information provenant par différents capteurs physiques soient statiques ou dynamiques. Parmi ces travaux nous sommes intéressés aux travaux présentés par Atif Manzoor et al en 2009. Ils considèrent les deux vues objectives et subjectives de QoC. La vision objective de QoC présente une qualité d'information de contexte indépendante des exigences d'un consommateur de contexte. Le calcul de cette vue implique les caractéristiques du capteur et la mesure de contexte. Tandis que la vue subjective de QoC présente la qualité de l'information de contexte dépendante pour un consommateur de contexte spécifique. Atif Manzoor considère cette nature objective et subjective de QoC et a proposé la définition de la QoC comme suit :

« Qualité du Contexte indique le degré de conformité du contexte collecté par les capteurs à la situation de l'environnement et aux exigences d'un consommateur de contexte particulier »
Atif Manzoor et al, ont évalué les différents critères de QoC par des équations en fonction des propriétés des capteurs physiques [72].

- Pour la deuxième catégorie (réseaux sociaux) : Parina Alamir en 2016 [101] propose l'approche de gestion de confiance en analysant les modèles de comportement des utilisateurs pour les réseaux sociaux. Pour ce faire, il a proposé une méthode de quantification de la relation de confiance basée sur l'analyse du journal des communications des utilisateurs et les facteurs de qualité de service dans les réseaux sociaux. Les facteurs de QoS consistant : accessibility, response ability, tendency to respond et agility et l'historique de journal des communications consistant en : abundance, novelty, et sincerity [101].
- Pour la troisième catégorie (composant de service web) : Dans le domaine des services Web, la qualité de service (QoS) a été discutée dans un certain nombre de recherches pour présenter les caractéristiques non fonctionnelles des services Web. Kuyoro Shade O et al [102] en 2012 ont désigné que la qualité de service (QoS) est généralement utilisée pour décrire les caractéristiques non fonctionnelles des services web et utilisée comme point de différenciation important entre les différents services Web. La qualité des informations fournies par ces services web correspond à la qualité de service (QoS) [102].
- Pour la dernière catégorie (composant logiciel) : Arun Sharma et al [103] en 2008, a utilisé le processus de hiérarchie analytique (AHP). L'AHP [103] est une technique qui aide les décideurs à structurer des décisions complexes, à quantifier des facteurs intangibles et à évaluer les choix dans des situations de décisions multi-objectifs. C'est un Framework de prise de décision complet et rationnel qui fournit une méthodologie puissante pour déterminer la valeur relative parmi un ensemble d'éléments. AHP permet de développer l'importance relative parmi les attributs en utilisant l'opinion des experts ou à travers une analyse de comparaison par paires exhaustive. Ce processus vise ainsi à développer par un algorithme un poids pour chacun des attributs. Dans l'approche proposée par Arun Sharma, AHP a pour le but d'attribuer les valeurs de poids aux caractéristiques et sous-caractéristiques du modèle proposé. Ces valeurs de poids sont ensuite utilisées pour évaluer la qualité globale du composant logiciel en utilisant des métriques appropriées. La formule suivante est utilisée pour évaluer la qualité du système :

$$Q=w_f F+w_r R+w_u U+w_e E+w_m M+w_p P \quad (4)$$

Où w_f , w_r, w_u, w_e, w_m , w_p sont les valeurs de poids attribué par AHP pour les caractéristiques de qualité : F (functionality), R (reliability), U(usability), E(efficiency), M (maintainability) et P (portability) respectivement. En outre, les caractéristiques sont divisées en sous-caractéristiques, les valeurs de F, R, U, E, M et P peuvent être mesurées en utilisant leurs constituants comme illustré par les équations 5,6,7,8,9 et 10 respectivement:

$$F=w_s S+w_a A+w_i I+w_{se} SE+w_c C+w_{co} CO+w_{re} RE \quad (5)$$

$$R=w_{ma} MA+w_{ft} FT+w_{rec} REC \quad (6)$$

$$E=w_{tb} TB+w_{rb} RB+w_{sc} SC \quad (7)$$

$$U=w_{un} UN+w_{le} LE+w_{op} OP+w_{com} COM \quad (8)$$

$$M=w_{st} ST+w_{te} TE+w_{ch} CH+w_{tr} TR+w_{fl} FL \quad (9)$$

$$P=w_{ad} AD+w_{in} IN+w_{rep} REP \quad (10)$$

Où S: Suitability, A: Accuracy, I: Interoperability, SE: Security, C: Compliance, CO: Compatibility, RE: Reusability, MA: Maturity, FT: Fault Tolerance et REC: Recoverability, TB : Time Behavior, RB : Resource Behavior, SC : Scalability, UN : Understandability, LE : Learnability, OP : Operability, COM : Complexity, ST : Stability,TE : Testability, CH : Changeability, TR : Trackability, FL : Flexibility, AD : Adaptability, IN : Installability, REP : Replaceability [103].

4.4.4.2 Description des algorithmes de MapReduce Skyline et la logique floue proposées

Le nombre croissant de sources de contexte alternatifs qui fournissent la même fonctionnalité mais qui diffèrent par des paramètres de qualité, pose un défi au niveau de la gestion de la QoC. Ce problème est transformé en sélection de meilleures sources de contexte candidats en ce qui concerne la qualité de l'information.

Dans cette thèse, nous présentons une méthode parallèle de sélection d'une source de contexte Skyline pour améliorer l'efficacité en utilisant le paradigme MapReduce.

Après l'évaluation des métriques de QoC des sources de contexte qui a été expliqué dans la section 4.4.4.1, nous appliquons nos méthodes proposées (MapRedce Skyline et la logique floue qui seront décrites dans les algorithmes 2 et 3).

La première méthode MapReduce Skyline est composée en deux étapes principales :

- Processus de Map: les sources de contexte sont partitionnées par le serveur maître en plusieurs blocs de données selon leur proximité avec l'utilisateur entre plusieurs serveurs esclaves N. La méthode de partitionnement proposée au sein de chaque serveur esclave vise à classer les sources de contexte en des catégories. Ces serveurs esclaves travaillent en parallèle pour accélérer le temps de classification des sources de contexte en des catégories. Cette catégorisation est conçue à partir de la grande diversité observée dans les sources de contexte dans l'IdO. Les catégories utilisées sont : réseaux sociaux, les capteurs physiques, les composantes de services web, les composantes des logiciels. Chaque catégorie est affectée à un autre serveur esclave pour minimiser le temps de calcul des skylines locaux. Ces skylines locaux issus des sources de contexte sont générés dans des blocs de données subdivisés (catégories). Le grand nombre des entités, les multiples attributs de qualité de contexte dans chaque catégorie de sources de contexte posent un grand défi pour calculer les skylines locaux. Pour gérer ce problème, nous avons proposé ainsi d'appliquer MapReduce Skyline au sein de chaque catégorie. En d'autres termes, chaque serveur esclave subdivise chaque catégorie en sous catégories qui fonctionnent en parallèle. Les skylines locaux de ces sous catégories sont fusionnés pour obtenir les skylines locaux de la catégorie correspondante. L'application de MapReduce Skyline en deux niveaux en parallèle permet d'améliorer l'efficacité du processus de skylines locaux et réduire de manière significative le coût de traitement.
    A. Pour la catégorie des capteurs physiques : on applique la fonction Map sur cette catégorie. On subdivise la catégorie des capteurs physiques en deux sous classes selon la situation de détection : (1) les capteurs statiques, (2) les capteurs dynamiques.
    B. Pour la catégorie des réseaux sociaux : on applique la fonction Map sur cette catégorie. On distingue deux sous classes selon le type d'usage : (1) Réseaux sociaux professionnels orientés sur la mise en valeur et les échanges professionnels de ses membres (par exemple : Linkedin, Viadeo), (2) Les réseaux sociaux à grands publics (par exemple : Facebook, twitter, youtube).
    C. Pour la catégorie des services web : on applique la fonction Map sur cette catégorie. On distingue deux sous classes selon le type de technologies : (1) Les services web de type representational state transfer (REST) exposent leurs fonctionnalités comme un ensemble de

ressources (URI) identifiables et accessibles par la protocole HTTP. (2) les services web SOAP exposent leurs fonctionnalités sous la forme de services exécutables à distance. Leurs spécifications reposent sur les standards SOAP et WSDL.

D. Pour la catégorie des composantes logicielles : on applique la fonction Map sur cette catégorie. on expose trois sous classes selon le type de logiciel [105]: (1) logiciels propriétaires sont créés par des sociétés privées et sont vendus aux utilisateurs sur Internet, (2) les logiciels gratuits sont créés par des firmes spécialisées. Ils sont fournis gratuitement aux utilisateurs qui en ont besoin. Ils sont téléchargeables gratuitement sur internet, (3) les logiciels libres où leur code source est ouvert à tous et peut-être changé en toute légalité. Ces logiciels sont souvent gratuits et téléchargeables sur internet.

- Le processus Reduce: cette fonction calcule les skylines globaux à partir des skylines locaux générés dans le processus de Map.

Ces deux processus sont illustrés dans la figure 4.3 :

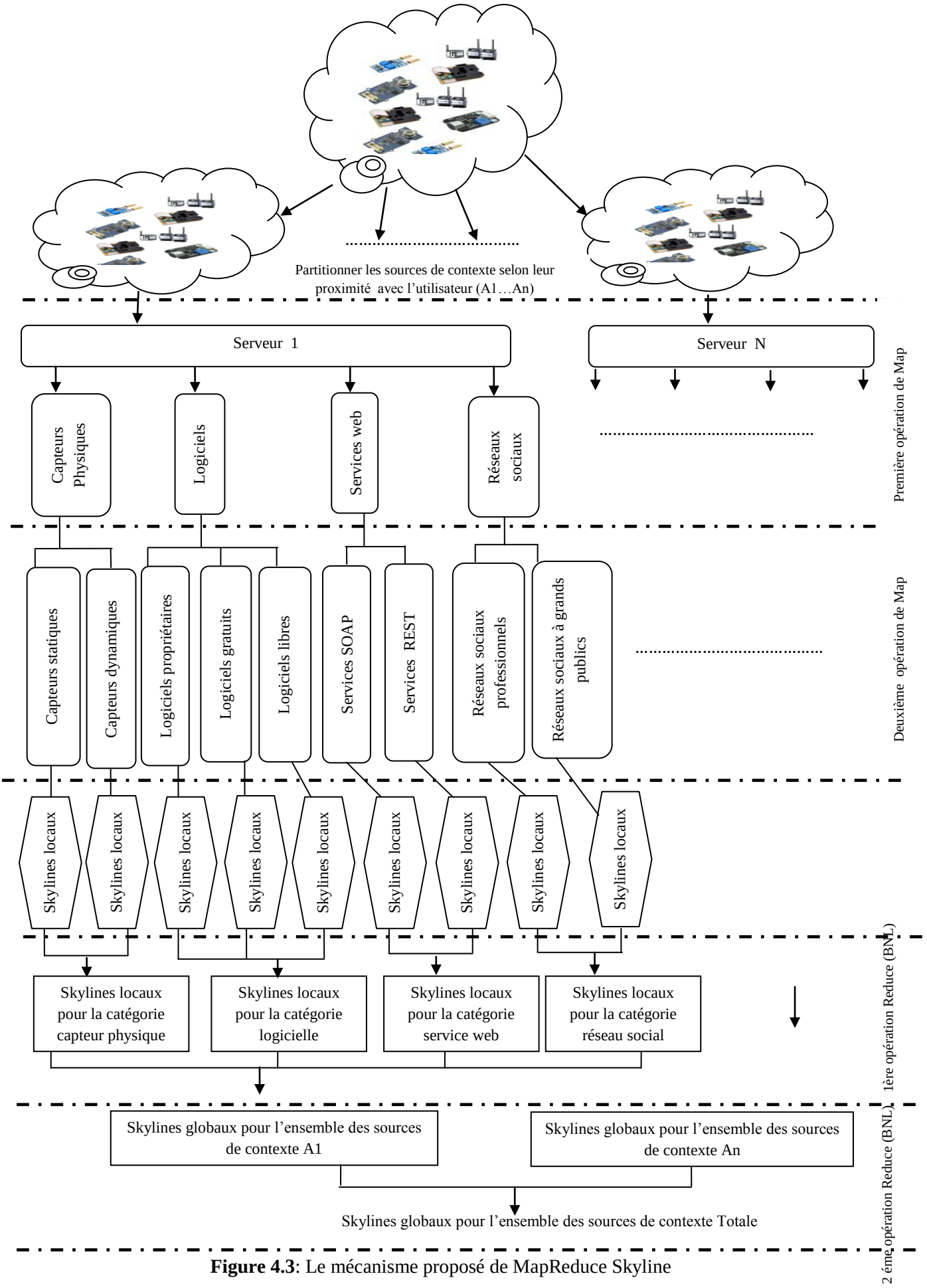

Figure 4.3: Le mécanisme proposé de MapReduce Skyline

La deuxième méthode de la logique floue est basée comme suit :

Nous appliquons la logique floue sur la liste des Skylines globaux générée dans la première méthode de MapReduce Skyline. La logique floue transforme les valeurs de critères de QoC de chaque source de contexte dans la liste des skylines globaux en valeurs floues à l'aide des fonctions d'appartenance. Le résultat de ces fonctions d'appartenance permet aux règles d'inférence de dériver les valeurs d'utilité de chaque source de contexte. La source de contexte qui a le poids le plus élevé est le candidat sélectionné.

La figure 4.4 explique le déroulement de la deuxième méthode pour sélectionner la meilleure source de contexte parmi les sources de contexte dans la liste de skylines globaux:

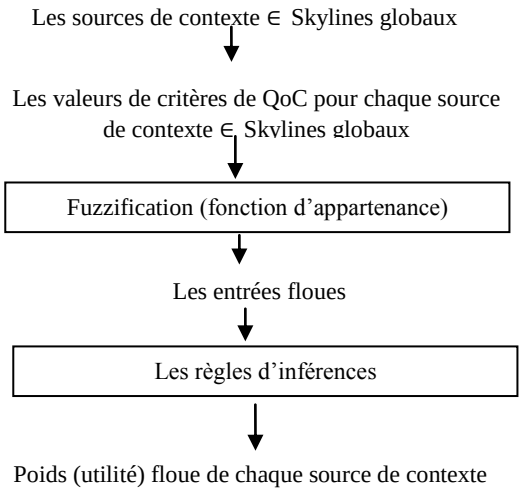

Figure 4.4: Le processus proposé de la logique floue

Nous décrivons ces deux méthodes en appliquant nos algorithmes proposés (Algorithmes 2 et 3). Pour ce faire, nous adoptons l'algorithme BNL (Algorithme 1) proposé par Borzsonyi et al [87] comme suit:

```
Algorithme 1: SkylineBNL
Input: input of the Skyline operation (set of points M)
Output: output of the Skyline operation (set of points R)
Begin algorithm 1
// T: temporary file, S: main memory, p<q: point p is dominated by point q;
// Initialization: R: =∅, T: =∅, S: =∅; CountIn: =0; CountOut:=0;
1 : While: ⅂EOF (M) do begin
2 : For each p ∈S do // transfer points which have been compared to all point;
3 : If TimeStamp (p) = CountIn then save(R,p), release(p);
4 : Load (M,p), TimeStamp(p):=CountOut; //load next point
5 : CountIn: =CountIn+1;
6 : For each q ∈S \ {p} do begin // compare p with all points q;
7 : If p <q then release (p), break;
8 : If p> q then release (q):
9 : End for
10 : If ⅂MemoryAvailable then begin
11 : Save (T,p), release(p);
12 : CountOut: =CountOut+1;
13 : End if
14 : If EOF (M) then begin
15 : M: = T, T: =∅; CountIn: =0, CountOut: =0
16 : End if
17 : End while
18 : For each p∈S do save (R,p), release(p);
19 : Return R;
End Algorithm 1
```

Algorithme 4.1: Algorithme BNL

Algorithme 2: Map reduce Skyline

Input: Different Context Sources (CS) (physical sensors, social network, software applications, web services);
Output: The list of global skylines (GS);
Begin Algorithme 2
// Apply our proposed partitioning method (for the set Ai) which based to classify the context source into categories (physical Sensor, Software component, Web service component, Social Network) to obtain the partitions list Pi (physical Sensor partition, Social Network partition, Software component partition, Web Service component partition);
1 : **For each** CSi∈CS **do Begin**
2 : **If** (CSi== Physical Sensor) **then**
3 : **If begin**
4 : Physical Sensor partition←CSi;
5 : **If** (CSi== Static Physical Sensor) **then**
6 : Static Physical Sensor partition←CSi ;
7 : **Else** Dynamic Physical Sensor partition←CSi ;
8 : **If end**.
9 : **Else begin**
10 :**If** (CSi==Social Network) **then**
11 : **If begin**
12 : Social Network partition←CSi;
13 : **If** (CSi== Professional Social Networks) **then**
14 : Professional Social Networks partition←CSi
15 : **Else** Public Social Network partition←CSi
16 : **If end.**
17 : **Else begin**
18 : **If** (CSi==Software component) **then**
19 : **If begin**
20 : Software component partition←CSi;
21 : **If** (CSi== Proprietary Software) **then**
22 : Proprietary Software partition←CSi
23 : **Else begin**
24 : **If** (CSi== Free Software) **then**
25 : Free Software partition←CSi
26 : **Else** Open Software partition←CSi
27 : **Else end**
28 : **If end**
29 : **Else begin**
30 : **If** (CSi==Service component) **then**
31 : **If begin**
32 : Service component partition←CSi;
33 : **If** (CSi==REST Service) **then**
34 : REST Service partition←CSi ;
35 : **Else** SOAP Service partition←CSi ;
36 : **If end**
37 : **End Else**
38 : **End Else**
39 : **End Else**
40 : **End for.**
41 : **For each** partitioned block Pi (context source category) **do**
42 : **begin**
// Compute Local Skylines LSi of each sub-category (SC) and local skylines of the corresponding category LSj from context sources CS using BNL described in algorithm 1);
43 : **For each** sub-category (SC) ⊆ partitioned block Pi **do**
44 : **Begin for**
45 : LSi←BNL (SC);
46 : **End for**
47 : LSj←BNL (LSi, ..,LSn);
48 : Output (Pi, LSj)
49 : **End for**
// Compute the Global Skylines GSi from the local Skylines LSj using BNL;
50 : GSi←BNL (LSj, ..,LSn);
51: GS←BNL(GSi,…..GSn); // Reduce the global skylines of each set (Ai,…An)
52 : Output (GS);
End Algorithme 2

Algorithme 4.2 : Le processus proposé de MapReduce Skyline

Algorithme 3: Fuzzy logic
Input: GS;
Output: Weigh of each context source in global skyline list based on the linguistic terms (WCS);
1 : **Begin**
// apply the fuzzification using one of the membership functions (ex: triangular membership functions);
2 : **For each** CSi ∈ GS (context source in global skylines) **do**
// Fuzzification (input: QoC Criteria values (QC), output: QoC Criteria based on the linguistic terms (QC_linguistic));
3 : QC_linguistic ←Fuzzification (QC);
// Inference rules
4 : WCS←Inference rules (QC_linguistic);
5 : **End for.**
End Algorithme 3// context source which has the higher weigh is the selected candidate.

Algorithme 4.3 : Le processus proposé de la logique floue

L'algorithme 4.2 est basé sur la partition du grand nombre de sources de contexte CS qui fournissent le même type d'informations de contexte demandées par une application. Notre méthode de partitionnement est basée sur la classification des sources de contexte en catégories (Pi). Chaque catégorie (Pi) est subdivisée en des sous catégories (SC). Ensuite, nous calculons les skylines locaux LSi de chaque sous-catégorie de source de contexte en fonction des préférences de l'application en utilisant la méthode skyline BNL. Les skylines locaux (les LSi) des sous-catégories sont fusionnés dans un skylines locaux (LSj) pour chaque catégorie correspondant en utilisant aussi la méthode skyline BNL. Les skylines locaux (les LSj) pour chaque catégorie sont fusionnés dans un skylines globaux (GSi) pour un ensemble Ai en utilisant aussi la méthode skyline BNL. Enfin, les skylines globaux obtenus de chaque ensemble Ai sont fusionnés en appliquant ainsi le BNL pour obtenir le résultat final de skylines globaux (GS).

Dans l'algorithme 4.3, nous appliquons la logique floue uniquement sur la liste des skylines globaux (GS) qui est le plus adapté à la demande d'application. Grâce à nos règles d'inférences générées, nous sélectionnons la source de contexte CS qui a le poids WCS le plus élevé pour satisfaire le besoin d'application. Notre solution a été publiée en International Journal of Information Technology (IAJIT) Vol. 15, No. 3A, en 2018 [106].

4.4.5 Exemple illustratif

Dans cet exemple, nous intéressons à fournir aux utilisateurs les informations de la température selon leurs préférences en fonction des paramètres de QoC bien définis. Cet exemple vise seulement à clarifier notre proposition de Mapreduce Skyline et la logique floue de manière simplifiée.
Il y a beaucoup de recherches dans la littérature qui ont défini des mesures de QoC. Nous considérons dans cet exemple les critères de QoC suivants : Reliability, Up-to-dateness. Ces deux critères de QoC ont été expliqués dans la section 2.3.
Dans cet exemple, on suppose qu'une application s'intéresse aux informations de haute Reliability et de Up-to-dateness moyenne.

1) Pour les informations de température provenant par des capteurs physiques, les paramètres de QoC Reliability et Up-to-dateness sont évalués selon la méthode d'Atif Manzoor en 2009 [100] qui a été expliquée dans le tableau 4.1 comme suit :
   a) Le paramètre de Reliability est calculé comme suit :

$$\text{Reliability} = \begin{cases} 1 - \frac{d(S,\varepsilon)}{d_{max}} * \left(n * \frac{\prod_{i=1}^{n} r_i}{\sum_{i=1}^{n} r_i}\right) & if\ d(S,\varepsilon) < d_{max} \\ 0 & otherwise \end{cases} \quad (11)$$

Où $d(S,\varepsilon)$ est la distance entre le capteur et l'entité. d_{max} est la distance maximale à laquelle on peut faire confiance sur l'observation de ce capteur. r_i indique toutes les caractéristiques *n* du capteur comme accuracy et precision qui contribuent à la reliability du capteur pour collecter une information particulier. Donc, accuracy et precision selon Manzoor (2009) sont évalués comme suit :

- $accuracy = 1 - \frac{|M-T|}{T}$ (12)

  Où M est la valeur mesurée et T est la vraie valeur de la quantité mesurée, M-T pour calculer l'erreur dans la mesure.
- La précision d'un capteur physique peut être mesurée en répétant les expériences pour un certain nombre de fois et en examinant la variation de données.

b) Le deuxième critère Up-to-dateness est calculé par la différence entre le temps actuel et le temps de la mesure.

2) Pour les informations de température provenant par des réseaux sociaux, les paramètres de QoC Reliability et Up-to-dateness sont évalués selon la méthode de Parina Alamir (2016) [101] qui a été expliquée dans le tableau 4.1 comme suit :

a) La Reliability est mesurée en fonction de paramètre « Abundance » pour les réseaux sociaux. Le critère Abundance indique le niveau des connexions entre les utilisateurs. Si Abundance entre utilisateur A et B a un bon niveau cela indique que l'information partagée dans le réseau social est ainsi crédible. Donc, la Reliability est calculée comme suit :

$$\text{abundance} = \frac{\text{con}_{\text{A,B}}}{\sum_{\text{k=1}}^{\text{n}} \text{con}_{\text{A,K}}} \quad (13)$$

Où $con_{A,B}$ dénote la conversation totale entre «A» et «B». $\sum_{k=1}^{n} con_{A,K}$ indique la conversation totale que 'A' a eu avec tous les autres utilisateurs. Si l'Abundance entre «A» et «B» est inférieure à l'Abundance entre «A» et «C», cela signifie que «A» fait confiance à «C» plus que B, donc l'information partagée par A et C est plus crédible que l'information partagée par A et B.

b) Le deuxième critère Up-to-dateness de l'information est calculée par la différence entre le temps actuel et le temps de partage de l'information dans le réseau social.

3) Pour les informations de température provenant par des services web, les paramètres de QoC Reliability et Up-to-dateness sont évalués selon la méthode proposée par Kuyoro Shade O (2012) [102] qui a été expliquée dans le tableau 4.1, elle est comme suit :

a) Le paramètre de reliability est lié à la livraison assurée et ordonnée pour les messages transmis et reçus par les demandeurs de service et les fournisseurs de services.

b) Up-to-dateness de l'information est calculée par la différence entre le temps actuel et le temps de fournir de l'information par un service web.

4) Pour les informations de température provenant par des logiciels, les paramètres de QoC Reliability et Up-to-dateness sont évalués selon la méthode proposée par Arun Sharma (2008) [103] qui a été expliquée dans le tableau 4.1, elle est comme suit :

a) Le paramètre de reliability est défini par Sharma sur trois sous-caractéristiques comme suit : Maturity, Recoverability, Fault Tolerance.

- Maturity : Nombre de versions publiées jusqu'à présent pour le même composant logiciel.

- Recoverability : il mesure la capacité d'un composant logiciel à récupérer suite à une défaillance inattendue et également de récupérer les données perdues.
- Fault Tolerance : cette sous-caractéristique indique si le composant logiciel peut maintenir un niveau de performance spécifié dans le cas de défaut.

b) Le deuxième critère Up-to-dateness est calculé par la différence entre le temps actuel et le temps de la mesure par un logiciel.

Nous avons choisi ces quatre méthodes pour mesurer la QoC de chaque catégorie de source de contexte, car elles ont un faible coût de traitement et sont faciles à utiliser par rapport à d'autres méthodes.
Supposons que le Tableau 4.2 montre les valeurs de critères de QoC calculées selon les méthodes décrites dans le Tableau 4.1 :

Les sources de contexte disponibles	**Reliability**	**Up-to-dateness**
capteur physique de temperature 1	0.5	0.2
Facebook	0.8	0.1
capteur physique de temperature 2	0.6	0.4
Twitter	0.3	0.6
capteur physique de temperature 3	0.4	0.3
capteur physique de temperature 4	0.1	0.8
Instagram	0.9	0.5
Linkedin	0.7	0.5

Tableau 4.2 : Valeurs des critères de QoC

Afin de vérifier notre solution, nous appliquons l'algorithme 2 (MapReduce skyline) dans cet exemple de données. Nous proposons d'appliquer la MapReduce Skyline en deux niveaux sur l'ensemble des sources de contexte. La première étape de Map Reduce skyline est d'utiliser notre méthode de partitionnement. Cette méthode est basée sur les quatre catégories de source de contexte (capteurs physiques, composant logiciel, composant de service, réseau social). Ensuite, chaque catégorie elle-même traite MapReduce Skyline de manière distribuée et parallèle. Dans ce cas, nous obtenons que l'ensemble de sources de contexte dans le tableau 4.2 est subdivisé en 2 catégories: partition de réseau social= { Facebook , Twitter, Instagram , Linkedin }et la partition de capteurs physiques={ capteur physique de temperature 1, capteur physique de temperature 2, capteur physique de temperature 3, capteur physique de temperature 4}.

Pour la première catégorie de réseau social : nous obtenons que cette catégorie soit subdivisée en deux sous-catégories comme suit : sous-catégorie de réseau social à grand public= {Facebook, Twitter, Instagram}, sous- catégorie de réseau social professionnel= {Linkedin}.

Pour la deuxième catégorie de capteurs physiques : nous constatons que cette catégorie soit subdivisée en deux sous-catégories comme suit : sous- catégorie de capteurs physiques statiques= {capteur physique de temperature 1, capteur physique de temperature 2}, sous- catégorie de capteurs physiques dynamiques= {capteur physique de temperature 3, capteur physique de temperature 4}.

Ensuite, nous calculons au sein de chaque sous-catégorie les skylines locaux de sources de contexte en utilisant l'algorithme BNL. Les skylines locaux de chaque catégorie sont obtenus en fusionnant (fonction Reduce) les skylines locaux des sous-catégories correspondantes.
Dans notre exemple, on suppose qu'une application est intéressée par des informations qui ont une reliability élevée (high reliability) et une Up-to-dateness moyenne (medium Up-to-dateness). Dans ce cas, le calcul skyline basé sur le BNL pour les catégories capteur physique et réseau social suive les étapes suivantes:

1. Selon la définition de la dominance, une source de contexte domine une autre source de contexte parce qu'elle a une reliability et une Up-to-dateness supérieures. Ainsi, les sources de contexte skylines sont les meilleurs compromis possibles entre reliability et Up-to-dateness.
2. Récupérer une source de contexte comme un skyline local si elle est supérieure ou égale à toutes les sources de contexte dans tous les critères QoC.
3. Éliminer les sources de contexte dominées par d'autres dans la partition.
4. Les skylines locaux pour la catégorie de réseau social sont:
   - Les skylines locaux pour la première sous-catégorie de réseau social à grand public : sont les sources de contexte « Twitter et Instagram ».
   - Les skylines locaux pour la deuxième sous-catégorie de réseau social professionnelle : c'est la source de contexte « Linkedin ».

   Donc, les skylines locaux pour la catégorie de réseau social c'est le résultat de fusion entre les skylines locaux de ces sous-catégories en utilisant l'algorithme BNL = {Twitter, Instagram}.
5. Les skylines locaux pour la catégorie de capteur physique en utilisant l'algorithme BNL:
   - Les skylines locaux pour la première sous-catégorie de capteur physique statique : c'est la source de contexte « capteur statique de température 2 ».
   - Les skylines locaux pour la deuxième sous-catégorie de capteur physique dynamique: sont les sources de contexte « capteur dynamique de température 3 et capteur dynamique de température 4 ».

   Donc, les skylines locaux pour la catégorie de capteur physique c'est le résultat de fusion entre les skylines locaux de ces sous-catégories en utilisant l'algorithme BNL = {capteur statique de température 2 et capteur dynamique de température 4}.
6. Les skylines globaux de notre ensemble de données est le résultat de fusion entre les skylines locaux de deux catégories de capteur physique et le réseau social : nous obtenons {Twitter, Instagram et capteur dynamique de température 4}.

Considérons dans cet exemple l'ensemble de données avec deux dimensions (Reliability, Up-to-dateness) représenté sur la figure 4.5. Les points skylines sont : {Twitter, Instagram et capteur dynamique de température 4} qui sont les meilleurs compromis possibles entre le Reliability et Up-to-dateness.

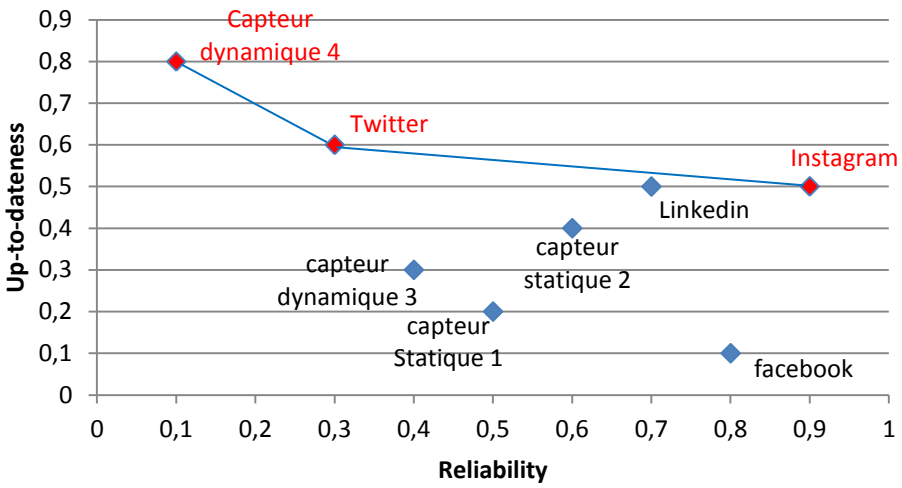

Figure 4.5: Les points Skylines de l'exemple

Basé sur le résultat de skylines globaux, nous sélectionnons la source de contexte qui est la plus adaptée pour la demande de l'application en appliquant l'algorithme 3 (logique floue).
Les valeurs de critères de QoC sont fuzzifiées en utilisant trois fonctions d'appartenance triangulaires: $F_H(x)$, $F_M(x)$ et $F_L(x)$. Ces fonctions d'appartenance sont présentées dans les équations (14), (15) et (16), respectivement. Ils évaluent les valeurs « High, Medium, Low» pour les critères de QoC, respectivement.

$$F_H(x) = \begin{cases} \frac{x-c}{d-c} & \text{if } c \le x \le d \\ \frac{e-x}{e-d} & \text{if } d < x \le e \\ 0 & \text{otherwise} \end{cases} \quad (14)$$

$$F_M(x) = \begin{cases} \frac{x-b}{c-b} & \text{if } b \le x \le c \\ \frac{d-x}{d-c} & \text{if } c < x \le d \\ 0 & \text{otherwise} \end{cases} \quad (15)$$

$$F_L(x) = \begin{cases} \frac{x-a}{b-a} & \text{if } a \le x \le b \\ \frac{c-x}{c-b} & \text{if } b < x \le c \\ 0 & \text{otherwise} \end{cases} \quad (16)$$

Où les constantes: a, b, c, d et e sont fournies par un expert en fonction des données. Cette fonction est simple et prend les valeurs d'utilisateur, puis il transfère directement à l'ensemble flou. Pour les systèmes qui nécessitent une variation dynamique significative sur une courte période de temps, la fonction triangulaire devrait être utilisée [107]. Si on met: a = 0; b = 0,1; c = 0,6; d = 1; e = 0,3, nous obtenons les valeurs floues suivantes (décrites dans le tableau 4.3) :

Les skylines globaux de source de contexte	Reliability	Up-to-dateness
Twitter	0 % high 40% medium 60% low	0 % high 100 % medium 0 % low
Capteur physique de temperature 4	0 % high 0 % medium 100 % low	50 % high 50 % medium 0 % low
Instagram	75 % high 25 % medium 0 % low	0 % high 80 % medium 20 % low

Tableau 4.3 : les valeurs floues calculées pour les skylines globaux

La prochaine étape de la logique floue est constituée par les règles d'inférence écrites par le concepteur du système floue en fonction de ses connaissances. Nous effectuons des règles d'inférence pour les skylines globaux des sources de contexte : Twitter, Capteur physique de température 4, Instagram. Les règles d'inférence générées sont les suivantes:

- Pour la source de contexte « Twitter »:

IF reliability 0% high AND Up-to-dateness 0% high THEN (Weight- Twitter) 0% high.
IF reliability 0% high AND Up-to-dateness 100% medium THEN (Weight- Twitter) 0% high.
IF reliability 0% high AND Up-to-dateness 0% low THEN (Weight- Twitter) 0% low.
IF reliability 40% medium AND Up-to-dateness 0% high THEN (Weight- Twitter) 0% high.
IF reliability 40% medium AND Up-to-dateness 100% medium THEN (Weight- Twitter) 40% medium.
IF reliability 40% medium AND Up-to-dateness 0% low THEN (Weight- Twitter) 0% low.
IF reliability 60% low AND Up-to-dateness 0% high THEN (Weight- Twitter) 0% high.
IF reliability 60% low AND Up-to-dateness 100% medium THEN (Weight- Twitter) 60% low.
IF reliability 60% low AND Up-to-dateness 0% low THEN (Weight- Twitter) 0% low.

- Pour la source de contexte « Capteur physique de température 4 » :

IF reliability 0% high AND Up-to-dateness 50% high THEN (Weight- Capteur physique de température 4) 0% high.
IF reliability 0% high AND Up-to-dateness 50% medium THEN (Weight- Capteur physique de température 4) 0% high.
IF reliability 0% high AND Up-to-dateness 0% low THEN (Weight- Capteur physique de température 4) 0% low.
IF reliability 0% medium AND Up-to-dateness 50% high THEN (Weight- Capteur physique de température 4) 0% medium.
IF reliability 0% medium AND Up-to-dateness 50% medium THEN (Weight- Capteur physique de température 4) 0% medium.
IF reliability 0% medium AND Up-to-dateness 0% low THEN (Weight- Capteur physique de température 4) 0% medium.

IF reliability 100% low AND Up-to-dateness 50% high THEN (Weight- Capteur physique de température 4) 50% high.
IF reliability 100% low AND Up-to-dateness 50% medium THEN (Weight- Capteur physique de température 4) 50% medium.
IF reliability 100% low AND Up-to-dateness 0% low THEN (Weight- Capteur physique de température 4) 0% low.

- Pour la source de contexte «Instagram » :

IF reliability 75% high AND Up-to-dateness 0% high THEN (Weight- Instagram) 0% high.
IF reliability 75% high AND Up-to-dateness 80% medium THEN (Weight- Instagram) 75 % high.
IF reliability 75% high AND Up-to-dateness 20% low THEN (Weight- Instagram) 20% low.
IF reliability 25% medium AND Up-to-dateness 0% high THEN (Weight- Instagram) 0% high.
IF reliability 25% medium AND Up-to-dateness 80% medium THEN (Weight- Instagram) 25% medium.
IF reliability 25% medium AND Up-to-dateness 20% low THEN (Weight- Instagram) 20% low.
IF reliability 0% low AND Up-to-dateness 0% high THEN (Weight- Instagram) 0% low.
IF reliability 0% low AND Up-to-dateness 80% medium THEN (Weight- Instagram) 0% low.
IF reliability 0% low AND Up-to-dateness 20% low THEN (Weight- Instagram) 0% low.

Nous avons basé dans nos règles d'inférence sur l'opérateur AND qui correspond à l'opérateur MIN. Dans nos règles d'inférence, nous avons plusieurs règles qui génèrent plusieurs valeurs de la même variable linguistique, nous avons choisi un opérateur OU pour combiner les valeurs de la même variable.

Par exemple, nous avons quatre règles qui génèrent la variable linguistique «high»: weight-Instagram est high à 0% et 75%. Si on utilise un opérateur OR (opérateur Maximum), la variable " weight- Instagram est high" aura une valeur finale de 75%. Par conséquent, nous obtenons :

- Pour Instagram, nous obtenons: weight- Instagram est high à 75%, medium à 25% et low à 0%.
- Pour Capteur physique de température 4, nous obtenons: weight- Capteur physique de température 4 est high à 50%, medium à 50 % et low à 0%.
- Pour Twitter, nous obtenons: weight-Twitter est high à 0%, medium à 40 % et low à 60%.

Enfin, la source de contexte « Instagram » possède un poids le plus élevé, c'est donc le candidat sélectionné qui est le plus proche au besoin de l'utilisateur.

4.5 Conclusion

Dans ce chapitre, un Middleware basé sur la logique floue a été proposé pour la gestion de la qualité de contexte (QoC) dans l'IdO. Nous avons utilisé les paramètres de QoC pour la sélection des sources de contexte. Le processus de sélection permet d'obtenir les informations les mieux adaptées à la demande d'utilisateur. Ce processus évalue les critères de QoC pour les sources de contexte et les consommateurs de contexte. De plus, dans notre approche, nous avons considéré une méthode parallèle et distribuée « MapReduce Skyline » pour accélérer le processus de calcul et gérer des sources de contexte et des données massives.

5.1 Analyse de scénario

Nous illustrons notre travail en utilisant un scénario simple « Mesure de température ». La température de milieu dans laquelle les personnes vivent est le terme donné à la fourniture de soins aux personnes soit dans leurs propres maisons ou dans les hôpitaux ou dans les lieux de travail...etc. La température de milieu est généralement soutenue par la technologie.

Les utilisateurs exploitent généralement des applications installées dans leurs smart phones pour connaître les informations nécessaires ce qui concerne le dégrée de température. Des métadonnées de QoC sont associées à ces informations (par exemple, la précision des mesures). Les exigences d'applications sont spécifiées par ces métadonnées de QoC sous forme des informations qualitatives. En revanche, plusieurs sources de contexte qui fournissent l'information de température sont disponibles (des capteurs physiques ou logicielles, des réseaux sociaux, des services web). Ces sources de contexte tournent autour de nous tout au long de notre vie quotidienne. Les différentes sources de contexte sont aussi spécifiées leurs garanties en terme de l'information de température fournies par des estimations approximatives (par exemple, la précision et la correction des mesures).

L'objectif de notre scénario est de fournir aux utilisateurs les informations de température nécessaire selon leurs besoins en termes de variables linguistiques. Pour réaliser cet objectif, notre approche vise à sélectionner la source de contexte la plus proche au besoin de l'utilisateur. Les entités responsables de traitement pour les grandes quantités de l'information sont généralement déployées sur des serveurs, où ils utilisent les deux méthodes proposées MapReduce Skyline et la logique floue. L'utilisation de notre approche pour le fonctionnement de ce scenario est illustrée dans la figure 5.1.

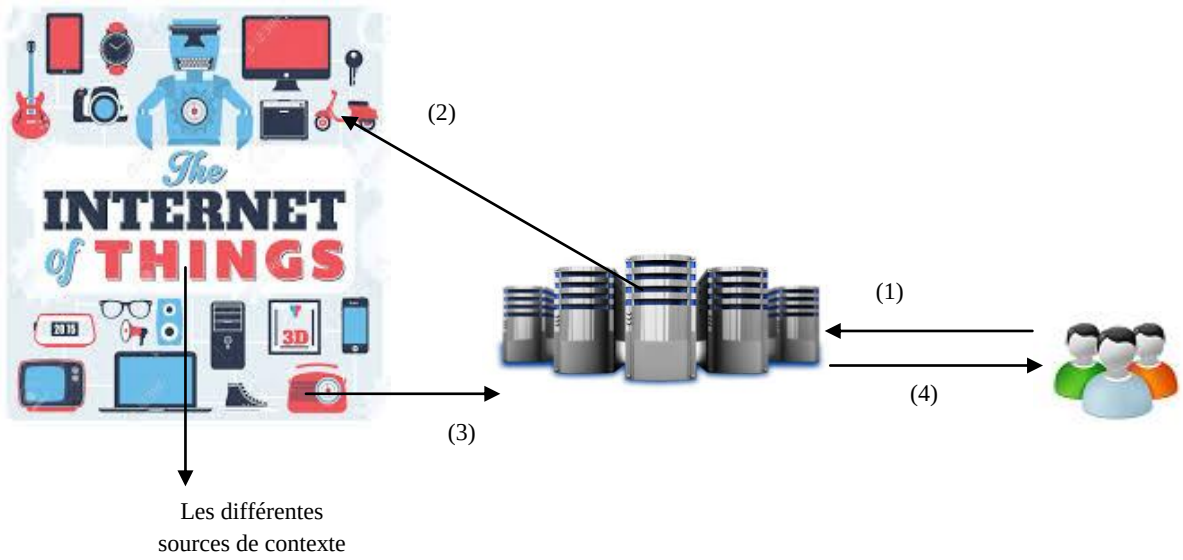

Figure 5.1: L'architecture de scenario

Où:

(1) L'utilisateur pose sa requête au serveur de traitement (gestionnaire de QoC) pour obtenir le dégrée de température selon des termes linguistique.

(2) Le serveur de traitement sélectionne toutes les sources de contexte disponibles qui fournissent le même type d'information (température). Chaque source de contexte fournit ainsi le degré de température selon des termes linguistiques.

(3) Le serveur de traitement utilise MapReduce Skyline et la logique floue pour sélectionner la source de contexte la plus proche au besoin de l'utilisateur.
(4) Le serveur de traitement fournit le degré de température appropriée au l'utilisateur.

5.2 Implémentation du gestionnaire de qualité de contexte proposé

5.2.1 Présentation de l'outil utilisé

Afin de faciliter le développement et la simulation de notre approche, nous avons utilisé l'environnement de développement intégré Eclipse KEPLER basé sur le langage JAVA. Toutes les expérimentations ont été réalisées sur un ordinateur du type Intel (R) Core (TM) i3-2328M, avec 2.20 GHz de fréquence d'horloge et 4 Go de RAM, sous le système d'exploitation Windows 7.

Eclipse est un IDE, Integrated Development Environment (EDI environnement de développement intégré en français), c'est-à-dire un logiciel qui simplifie la programmation en proposant un certain nombre de raccourcis et d'aide à la programmation. Il est gratuit et disponible pour la plupart des systèmes d'exploitation [108].

Eclipse IDE est principalement écrit en Java (à l'aide de la bibliothèque graphique SWT, d'IBM). Un des avantages d'Eclipse et de fournir un environnement intégré qui facilite la création, la compilation, les tests et l'exécution de projets java qui pouvant contenir de nombreux fichiers de code, repartis dans différents paquetages et référençant de nombreuses librairies [109].

5.2.2 Présentation du déroulement du système

Dans cette section nous allons présenter les interfaces permettant aux utilisateurs d'utiliser le gestionnaire de qualité de contexte proposé pour obtenir l'information adéquate aux leurs besoins.

1. Interface de requête de l'application : cette interface permet aux utilisateurs de présenter leur besoins en sélectionnant les différents critères de qualité de l'information souhaité sous forme des termes linguistiques.
   Après la sélection des différents critères de qualité de l'information par l'utilisateur, le Bouton « Web Service» permet d'afficher toutes les sources de contexte de type services web qui fournissent l'information de température. C'est le même principe pour les autres boutons de « Physical Sensor», « Software », « Social Network» qui permettent d'afficher toutes les sources de contexte de type capteurs physiques, logiciels, réseau social, respectivement.
   Le bouton « Start request » permet de lancer la requête au gestionnaire de QoC pour obtenir l'information de température nécessaire.
   Comme l'illustré la figure 5.2, l'utilisateur demande l'information de température qui a une grande précision et une moyen Granularity.

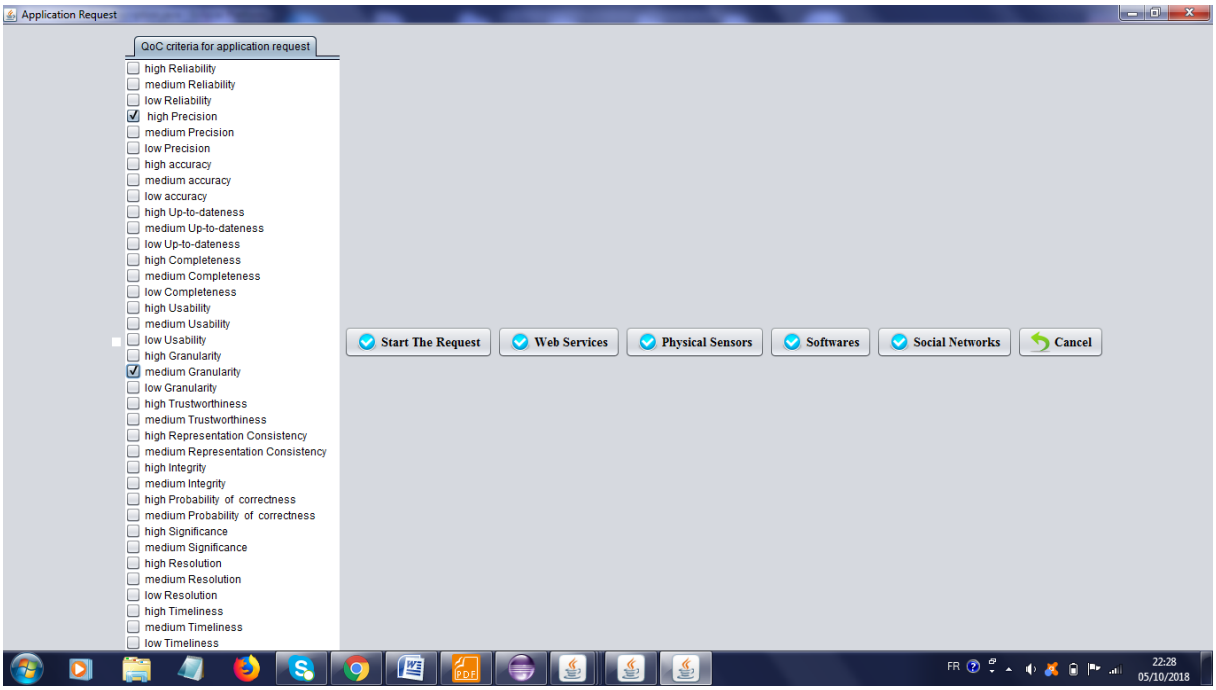

Figure 5.2: Interface de requête de l'application

L'interface indiquée dans la figure 5.3 permet d'afficher tous les services web disponibles qui fournissent l'information de température. Chaque source de contexte disponible possède :

- le numéro de source de contexte.
- La catégorie de source de contexte (services web ou logiciel ou réseau social ou capteur physique).
- La sous catégorie de source de contexte selon la catégorie correspondant.
- Le type de l'information recherché par l'utilisateur (information de température).
- Le besoin de l'utilisateur sous forme des termes linguistiques.
- La garantie de source de contexte au niveau des valeurs de critères de la qualité de l'information.

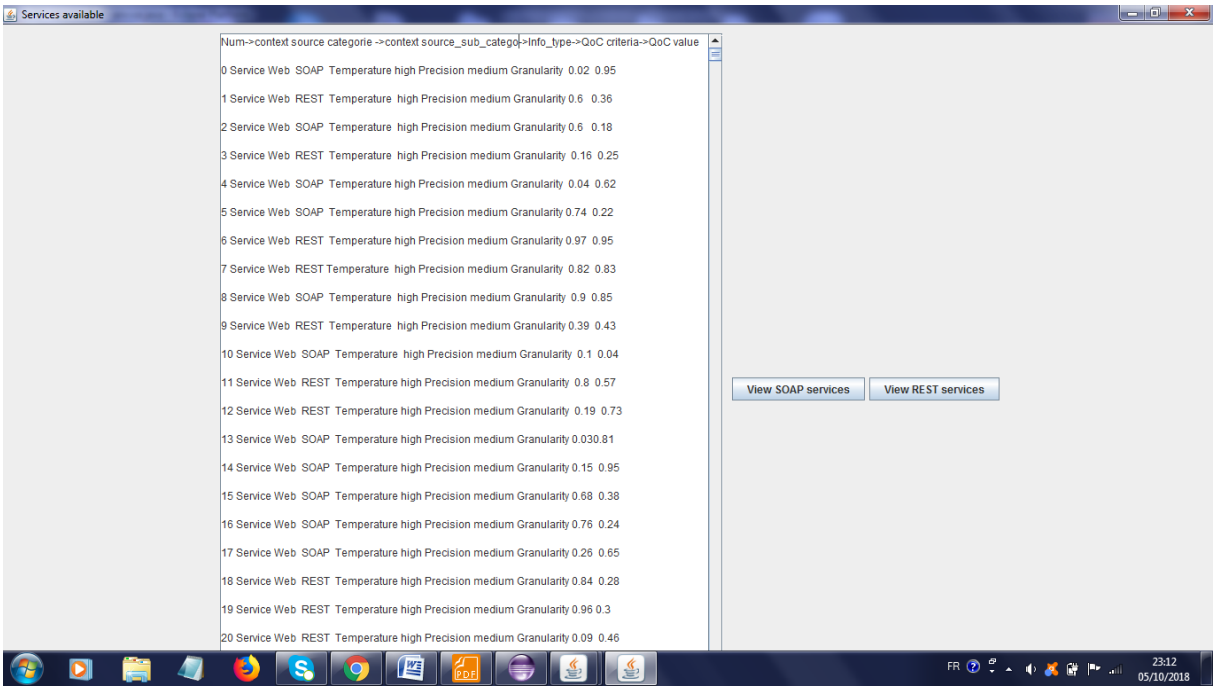

Figure 5.3: Interface des services web disponibles.

Cette interface permet de spécifier ainsi les deux types possibles de services web (REST ou SOAP) comme suit dans la figure 5.4 qui affiche tous les services web de type REST disponibles.

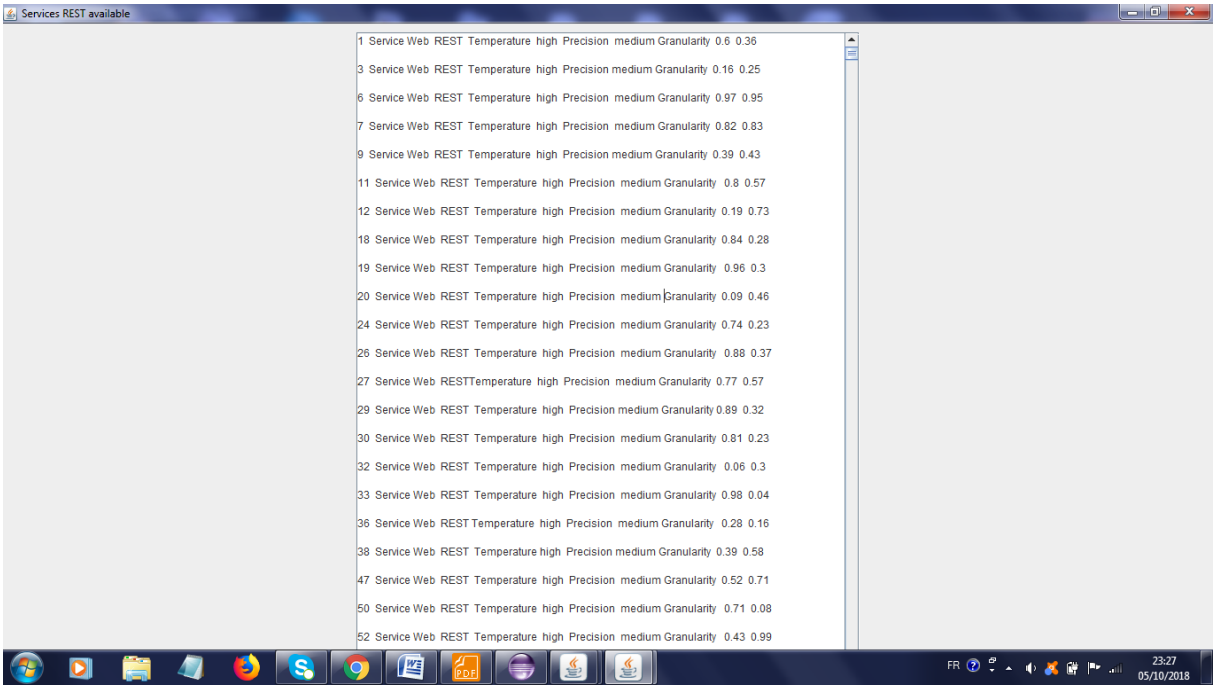

Figure 5.4: Interface des services web de type REST disponibles

2. Interface des règles d'inférences de la logique floue pour les sources de contexte candidats : le bouton « Start Request» permet de faire le traitement sur l'ensemble des sources de contexte disponibles. Ce traitement se concentre sur la MapReduce Skyline avec deux niveaux (subdivision sur des catégories et des sous catégories). Le calcul skyline basé BNL permet d'afficher les sources

de contexte qui sont incomparables et proche à la demande d'utilisateur. Le mécanisme de la logique floue est appliqué sur l'ensemble résultant de MapReduce Skyline. La figure 5.5 montre les règles d'inférences obtenu après la fuzzification des valeurs de critères de QoC pour les sources de contexte candidates.

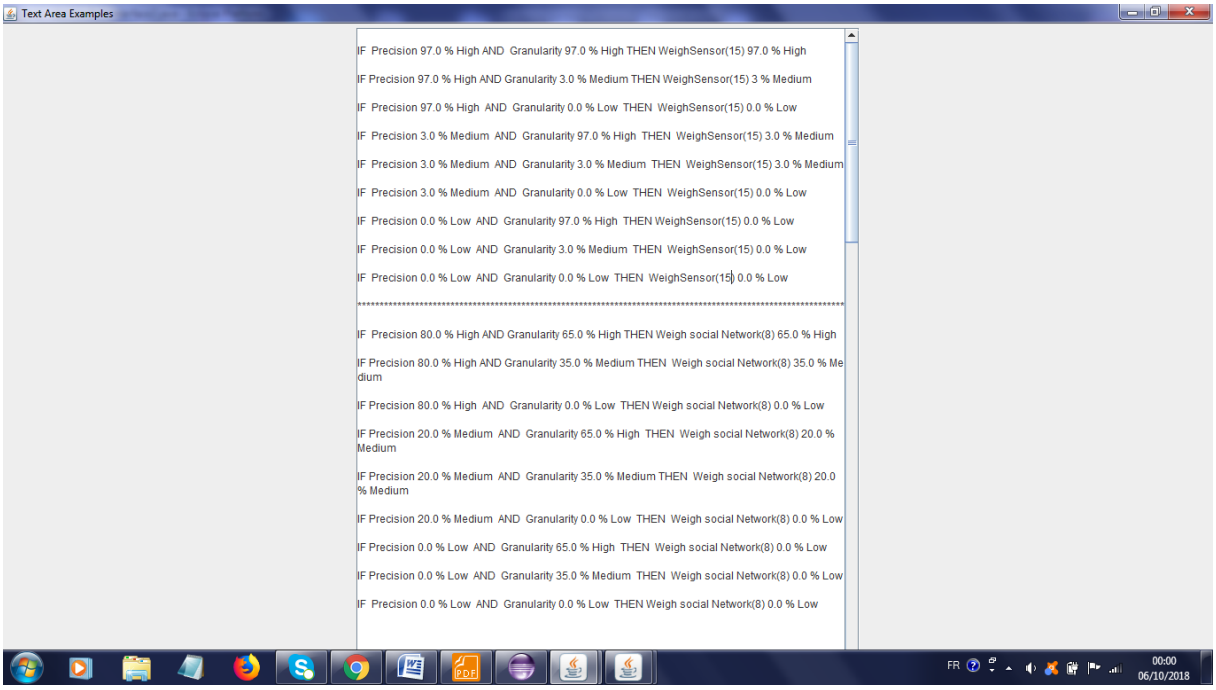

Figure 5.5: Interface des règles d'inférences pour les sources candidates

3. Interface d'affichage de résultat : cette interface permet d'afficher la source de contexte sélectionnée pour satisfaire le besoin de l'utilisateur comme il est indiqué dans la figure 5.6. Les conclusions de ces règles d'inférences permettent de générer le poids de chaque source de contexte candidate en termes des variables linguistiques. La source de contexte choisie possède un poids le plus élevé.

Figure 5.6: Interface d'affichage de résultat

5.2.3 Évaluation des résultats obtenus

Dans cette étude, nous allons aborder le problème de la sélection des sources de contexte parmi un nombre assez grand dans l'environnement d'IdO. Nous envisageons une nouvelle approche pour identifier les sources de contexte candidates. La principale caractéristique de notre approche proposée est qu'il peut extraire des résultats à l'aide d'un calcul skyline. De plus, dans notre approche proposée, nous considérons l'infrastructure parallèle distribuée MapReduce pour accélérer le processus de calcul skyline basé sur BNL et pour gérer des données massives. La figure 5.7 indique le temps d'exécution

nécessaire requis pour sélectionner l'ensemble de skylines globaux en fonction de besoin de l'utilisateur. Cette figure permet d'introduire le temps d'exécution selon :

- Le nombre de sources de contexte
- Le nombre de critères de QoC (deux critères, trois critères et quatre critères).

Le temps d'exécution dans notre proposition est très raisonnable avec l'augmentation du nombre des sources de contexte et les paramètres de QoC.
Notre proposition vise à accélérer le calcul skyline en utilisant le MapReduce. Cette méthode permet de subdiviser les sources de contexte en des catégories et des sous catégories. Le calcul skyline basé sur BNL est appliqué aux petits nombres des sources de contexte qui se trouvent dans ces sous catégories de manière distribuée et parallèle entre plusieurs serveurs esclaves.

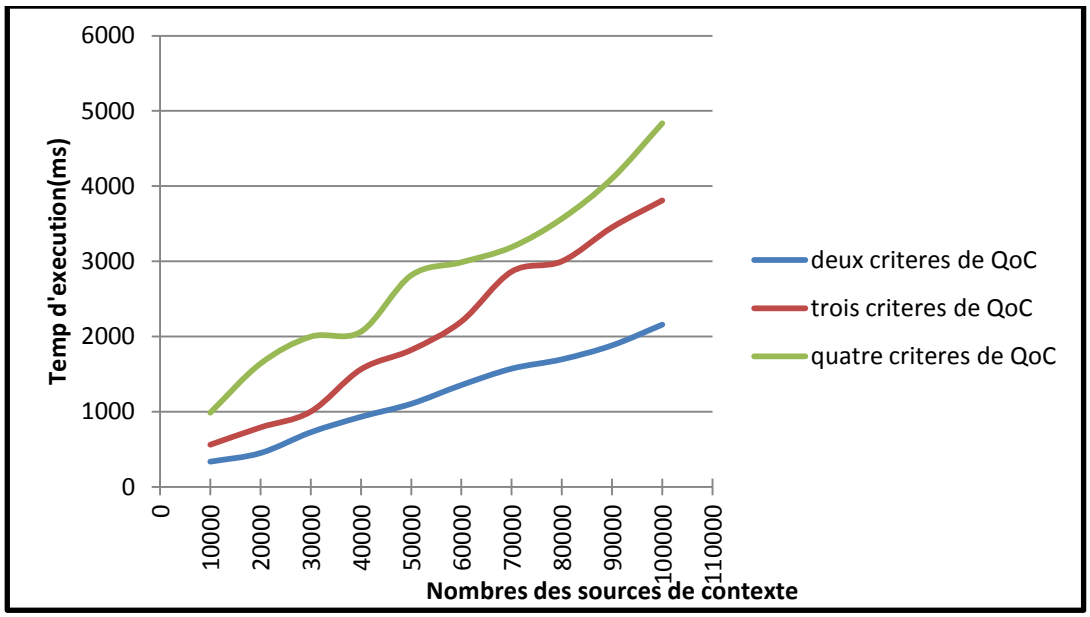

Figure 5.7: Temps d'exécution pour sélectionner les meilleures sources de contexte en utilisant calcul skyline basé sur BNL

Le résultat obtenu montre que l'augmentation des sources de contexte et les paramètres de QoC entraînent une augmentation significative du temps d'exécution pour le calcul skyline basé sur BNL.
La figure 5.8 indique le temps d'exécution pour le calcul skyline requis pour sélectionner les meilleures sources de contexte en fonction des besoins de l'utilisateur dans trois cas :

- Le MapReduce n'a été pas utilisé : dans ce cas, nous avons calculé le temps d'exécution de calcul skyline basé sur BNL sur l'ensemble des sources de contexte sans les subdiviser en catégories et des sous catégories. Cette méthode consomme un temps énorme avec l'augmentation des sources de contexte.
- L'utilisation de Map Reduce avec un seul niveau : dans ce cas, nous avons calculé le temps d'exécution de calcul skyline basé sur BNL sur l'ensemble des sources de contexte en les subdivisant aux quatre catégories (capteurs physiques, logiciels, réseaux sociaux, services web). Cette méthode consomme un temps moyen avec l'augmentation des sources de contexte car elle est appliquée le calcul skyline au sein de chaque catégorie en parallèle.
- Dans le cas où nous avons utilisé le MapReduce avec deux niveaux : nous avons calculé le temps d'exécution de calcul skyline basé sur BNL sur l'ensemble des sources de contexte en les subdivisant aux quatre catégories (capteurs physiques, logiciels, réseaux sociaux, services web). Chaque catégorie est elle-même est subdivisée en des sous catégories. Cette méthode

consomme un temps agréable avec l'augmentation des sources de contexte car elle a appliqué le calcul skyline à un petit nombre des sources de contexte au sein de chaque sous catégorie en parallèle.

Cette figure montre que le temps d'exécution nécessaire pour sélectionner les meilleures sources de contexte en utilisant le calcul skyline basé sur BNL pour deux critères de QoC est toujours meilleur dans notre proposition où le MapReduce est appliqué sur deux niveaux.

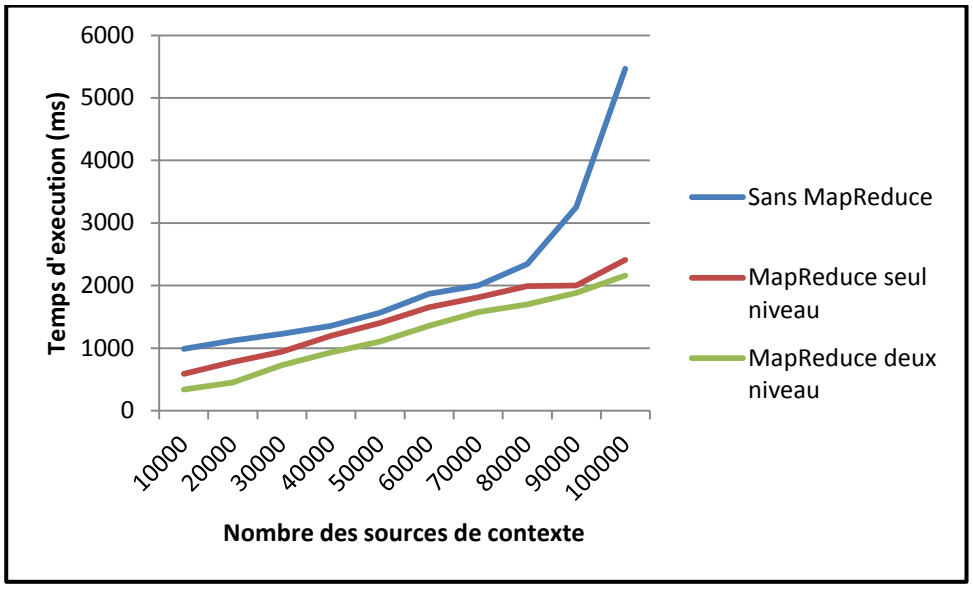

Figure 5.8: Comparaison de performance de la technique proposée avec la technique de MapReduce avec un seul niveau et le cas où le MapReduce n'est pas utilisé pour deux critères de QoC.

Pour choisir la source de contexte candidate satisfaisant le besoin de l'utilisateur, nous avons proposé la logique floue pour gérer l'évaluation de la qualité de contexte (QoC).
Nous appliquons la logique floue à la liste de skylines globaux qui contiennent un nombre petite des sources de contexte générée grâce au MapReduce Skyline. La logique floue transforme les valeurs des critères de QoC de chaque source de contexte dans la liste de skylines globaux en valeurs floues à l'aide de fonctions d'appartenance. Le résultat de ces fonctions d'appartenance permet aux règles d'inférence de dériver les valeurs d'utilité de chaque source de contexte. Ces valeurs d'utilité permettent de choisir la source de contexte candidate.
La figure 5.9 signifie le temps d'exécution nécessaire pour l'évaluation de la QoC en utilisant la logique floue dans le cas où nous n'avons pas utilisé le MapReduce Skyline.

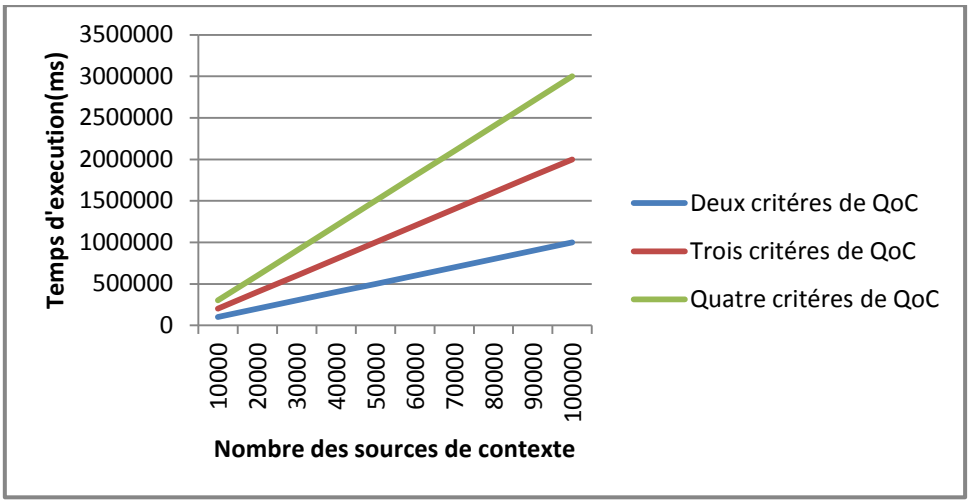

Figure 5.9: Temps d'exécution de la logique floue sans utiliser le MapReduce Skyline

La figure 5.9 exprime que l'augmentation des sources de contexte et les paramètres de QoC entraînent une augmentation significative du temps d'exécution pour la logique floue. Pour cette raison, nous avons proposé d'appliquer la logique floue à un petit nombre de sources de contexte résultant de la MapReduce Skyline plutôt qu'à un grand nombre.

La figure 5.10 indique le temps d'exécution nécessaire requis pour sélectionner la source de contexte candidate pour deux critères de QoC en utilisant la logique floue.
L'ensemble résultant des sources de contexte skylines est le même dans la méthode d'utilisation le MapReduce Skyline avec seul niveau et dans la méthode de MapReduce Skyline avec deux niveaux. Mais, la seule différence entre ces deux méthodes est : le MapReduce Skyline avec deux niveau est plus efficace et consomme peu de temps par rapport à la méthode de MapReduce Skyline avec seul niveau.
Appliquer la logique floue sur l'ensemble des sources de contexte skylines permet de déduire le poids de chacun de ces sources. La source de contexte qui possède un poids plus haut, c'est la source de contexte choisie.

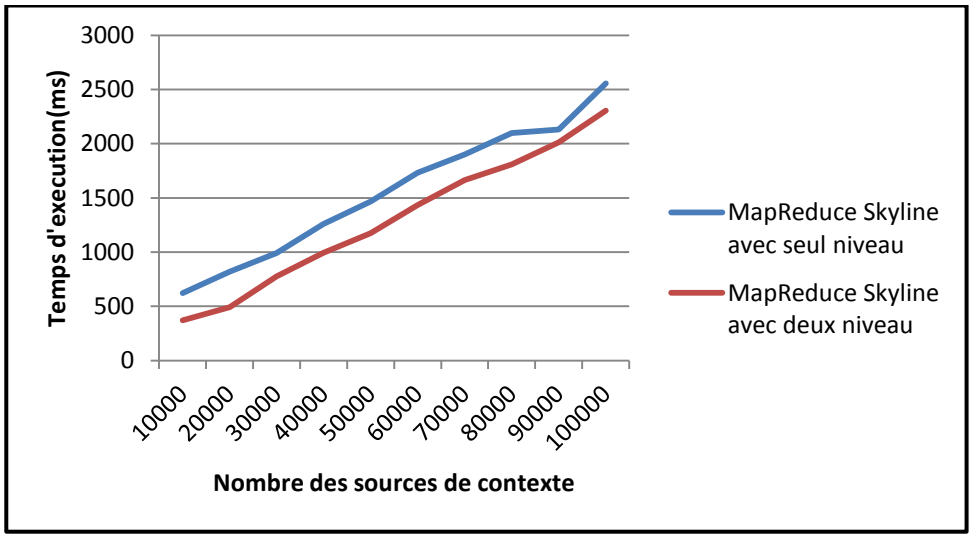

Figure 5.10: Temps d'exécution de la logique floue + le MapReduce Skyline dans notre proposition et dans le cas de MapReduce Skyline avec seul niveau pour deux critères de QoC.

Cette figure montre que le temps d'exécution nécessaire pour sélectionner la source de contexte en utilisant la logique floue est toujours meilleur dans notre proposition où le MapReduce est appliqué sur deux niveaux.

5.3 Comparaison et résultats

5.3.1 Description

Quelques travaux ont été réalisés dans le domaine de la recherche et la sélection des sources de contexte selon le besoin de l'utilisateur dans l'IdO. Tout middleware proposé doit respecter les données de sources de contexte pertinentes sans demander aux utilisateurs de choisir manuellement les sources qui correspondent à leurs besoins. Pour cela, la recherche et la sélection de la meilleure source de contexte reste un grand défi.

Suivant cette vision, nous pouvons montrer l'efficacité et la faisabilité de la méthode de sélection de sources de contexte proposée en la comparant à d'autres techniques. De nombreuses solutions middleware pour l'IdO sont introduites dans la littérature pour la recherche et la sélection des capteurs. Les plus pertinentes pour notre approche sont la CASSARAM, AntClust [110].

1. Le Framework proposé dans le travail de Perera et al. [111] est nommé CASSARAM permet aux utilisateurs de rechercher et de sélectionner les capteurs les mieux adaptés à leurs besoins. CASSARAM prend en compte les préférences de l'utilisateur et une large gamme de caractéristiques du capteur, telles que la fiabilité, la précision, la durée de vie de la batterie, etc. Les capteurs sont classés en fonction des exigences de l'utilisateur basées sur la proximité. Pour ce but, ils ont développé une technique d'indexation basé sur la distance euclidienne, appelée Comparative Priority Based Weighted Index (CPWI). En fait, une méthode heuristique est nommée Comparative Priority-based Heuristic Filtering (CPHF) est appliquée pour réduire la quantité de données nécessaires à traiter pendant la découverte et la sélection. L'idée de base de CPHF est de supprimer les capteurs éloignés à des capteurs idéaux définis par l'utilisateur et de réduire le nombre de capteurs qui doivent être indexés et classés [112].
2. Ebrahimi et al. [113] ont proposé un algorithme méta-heuristique (Antclust) afin de regrouper des capteurs sous la forme Sensor Semantic Overlay Networks (SSONs) dans lesquels des capteurs avec des informations de contexte similaires sont rassemblés dans un cluster. Dans chaque SSON, les capteurs sont regroupés par un algorithme de classification en fonction de leurs propriétés de contexte (précision, fiabilité, énergie, disponibilité, coût, etc.). Ainsi, les requêtes ne seront transmises qu'aux SSON pertinents, et donc, aux clusters associés avec les attributs qui satisfont aux contraintes du contenu de la requête. Cela réduit l'espace de recherche et augmente éventuellement l'efficacité du processus de la sélection. Pour ce but, la distance euclidienne entre les attributs de requête et chaque cluster sera calculée. Enfin, les clusters contenant les capteurs les plus appropriés pour aider les utilisateurs à résoudre leurs propres problèmes seront sélectionnés à la suite du processus de recherche [114].

Dans le tableau 5.1, nous présentons une brève comparaison entre notre technique et les deux techniques CASSARAM et Antclust que nous avons décrit dans le paragraphe ci-dessus.

Les techniques de la sélection	Notre technique proposée	CASSARAM	Antclust
Modèle de Représentation des sources de contexte	- Modéliser les sources de contexte en utilisant les modèles de l'MDA.	- Semantic Sensor Network Ontology (SSNO) pour modéliser les descriptions de capteurs et les propriétés de contexte.	- Utiliser de Semantic Sensor Network Ontology (SSN) Pour modéliser les propriétés et les descriptions du contexte des capteurs.
Modèle de Représentation du besoin de l'utilisateur	- Représenter le besoin de l'utilisateur au niveau QoC sous forme d'aspect floue	- Le prototype d'outil CASSARA fournit une interface à l'utilisateur permettant d'exprimer ces informations via des requêtes SPARQL.	- Pas un modèle spécifique.
Méthode de sélection	- MapReduce Skyline pour sélectionner les sources candidates. - La logique floue pour choisir la source de contexte le plus adéquate à l'utilisateur.	- Chaque capteur est tracé dans un espace multidimensionnel où chaque dimension est représentée par une propriété de contexte. - Comparative Priority-based Weighted Index (CPWI) est généré pour chaque capteur en combinant les priorités des utilisateurs et les propriétés de contexte. - Les capteurs sont classés en fonction de CPWI et le nombre de capteurs requis par l'utilisateur est sélectionné à partir en haut de la liste.	- Un algorithme méta-heuristique AntClust pour regrouper les capteurs sous la forme Sensor Semantic Overlay Networks (SSONs). - Calculer la distance euclidienne entre les attributs de requête et chaque cluster. - les clusters contenant les capteurs les plus appropriés seront sélectionnés.
Avantages	- Accélérer le processus de recherche et la sélection grâce à MapReduce Skyline. - Modélisation des sources de contexte en utilisant le modèle de l'IDM - Utiliser la logique floue pour faciliter aux consommateurs de contexte de mentionner ses exigences en matière de QoC en termes	- sélectionner des capteurs en fonction des exigences et des priorités des utilisateurs. - CASSARAM est un Modèle de recherche, de sélection et de classement de capteurs sensible au contexte. - L'ontologie SSN est capable de modéliser une quantité importante d'informations sur les	- Prend en considération l'augmentation spectaculaire du nombre de capteurs dans le futur d'IdO. - Utiliser un algorithme méta-heuristique qui réduit l'espace de recherche du capteur et augmente progressivement l'efficacité du processus de recherche. - Proposer une stratégie

	linguistiques. - Utiliser la logique floue pour faciliter aux producteurs de contexte de représenter ses garanties en matière de QoC en termes linguistiques à cause de l'incertitude des sources de contexte dans IdO. - Temps de traitement est clairement acceptable.	capteurs.	adaptative pour maintenir les performances du système face à la dynamicité de l'IdO.
Inconvénients	On a ignoré les performances de système pour faire face aux changements (joindre de nouvelles sources de contexte) dans l'IdO.	- Indexer et classer un grand nombre des capteurs est inefficace et gaspille une quantité importante de ressources de calcul. - Comparative Priority-based Heuristic Filtering (CPHF) améliore la performance de CASSARAM mais il reste un peu inefficace. - N'a pas considéré la nature dynamique de l'IdO.	- Avec un nombre croissant de capteurs, le nombre d'itérations nécessaires à la création du SSONs augmente. Mais la nature distribuée et la capacité d'auto-organisation de l'algorithme Antclust, vise à améliorer un peu la performance pour la création de SSONs.

Tableau 5.1 : Brève comparaison entre notre technique et les deux techniques CASSARAM et Antclust

5.3.2 Comparaison avec CASSARAM et Antclust

Nous avons comparé le temps d'exécution de notre technique de sélection de source de contexte optimale pour répondre à la requête de l'utilisateur avec CASSARAM et Antclust. La figure 5.11 montre le temps de traitement de la technique proposée pour trois critères de QoC par rapport à CASSARAM et Antclust. Nous constatons que le temps d'exécution de la technique proposée est nettement meilleur que celui de la technique CASSARAM. L'approche CASSARAM gaspille un temps important pour sélectionner les capteurs nécessaires selon les préférences de l'utilisateur. Cette méthode prend un temps considérable pour indexer et classer un grand nombre des capteurs. Même l'utilisation de la méthode Comparative Priority-based Heuristic Filtering (CPHF) pour améliorer la performance de CASSARAM n'a ajuste pas l'efficacité de façon remarquable. En revanche, notre proposition utilise le MapReduce Skyline pour accélérer le calcul de manière distribuée et parallèle. Le modèle de MapReduce accélère le processus de sélection des sources de contexte skylines en explorant le parallélisme distribué.

D'autre part, nous constatons ainsi que le temps d'exécution de la technique proposée est approximativement égal avec une différence négligeable à la technique Antclust, en particulier lorsqu'il y a moins de 5000 sources de contexte. Cependant, le temps d'exécution commence à augmenter par rapport à l'Antclust lorsque le nombre des sources de contexte est supérieur à 5000 en raison du temps obtenu de l'algorithme de skyline BNL pour un grand nombre des sources de contexte.

Dans l'approche Antclust, la requête de l'utilisateur basée sur les priorités de l'utilisateur, et prend ainsi en considération les types et les caractéristiques des capteurs. Il est bien connu que les caractéristiques de capteurs affectent sur les informations fournies. Cependant, Antclust ne propose aucune solution pour évaluer l'inexactitude et l'ambiguïté de ces informations. Elle sélectionne seulement les clusters contenant les capteurs les plus appropriés pour répondre au besoin de l'utilisateur. Par contre, notre proposition vise à évaluer la qualité de l'information fournie par différentes sources de contexte en utilisant la logique floue. Elle peut donc traiter efficacement le flou et l'imprécision de l'information sous forme des termes linguistiques. Pour réduire le temps de traitement et accélérer le processus de la logique floue, nous avons appliqué la logique floue sur l'ensemble des sources de contexte skylines.

Notre proposition et Antclust visent à réduire l'espace de recherche et augmente éventuellement l'efficacité du processus de sélection. Dans Antclust, les requêtes de l'utilisateur ne seront transmises qu'aux SSON pertinents et aux clusters associés. Par ailleurs, dans notre proposition, l'évaluation basée sur la logique floue ne sera appliquée qu'à un ensemble de petit nombre des sources de contexte skylines.

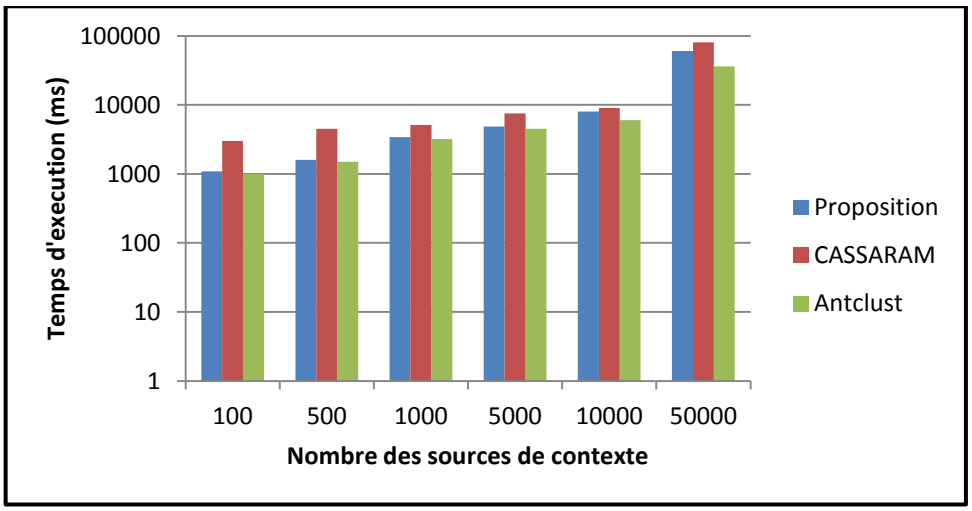

Figure 5.11: La comparaison des performances de notre proposition avec CASSARAM et AntClust

5.4 Conclusion

Le calcul Skyline est utilisé dans la sélection de meilleures sources de contexte pour satisfaire le besoin de l'utilisateur. Cependant, les sources de contexte à grande échelle dans l'environnement d'IdO posent un défi pour la sélection des sources de contexte skylines. Dans ce chapitre, nous avons démontré l'efficacité de notre proposition pour la sélection des sources de contexte Skylines parallèles. Nous avons démontré ainsi l'efficacité d'utiliser la logique floue pour sélectionner la source de contexte candidate parmi l'ensemble des sources de contexte Skylines.

Comme indiqué dans les travaux voisins, quelques travaux ont été réalisés dans le domaine de la recherche et la sélection des sources de contexte dans l'IdO. Afin d'évaluer les performances du

gestionnaire de qualité de contexte proposé, nous l'avons comparé avec ces méthodes de sélection des capteurs les plus connues dans le domaine. Cette comparaison permet de nous donner des résultats très prometteuses pour notre proposition.

Chapitre 6 : Conclusion et perspectives

6.1 Synthèse des contributions

L'Internet s'étend au monde physique en formant d'environnement de l'Internet des objets et en créant des espaces intelligents (des lieux géographiques qui contiennent des objets connectés qui peuvent communiquer ensemble et offrir des services d'IdO).

Dans le contexte de l'Internet des objets sur un grand nombre d'appareils mobiles et hétérogènes, il est devenu difficile pour l'utilisateur d'avoir des informations exactes et correctes. La recherche des informations basée sur une description qualitative permettant une recherche flexible où les informations retournées peuvent être proches à la demande recherchée. Cette proposition semble d'être une bonne approche pour la recherche des informations dans l'IdO. En même temps, les applications d'IdO nécessitent de plus en plus de données contextuelles avec une qualité de contexte (QoC) adéquate pour améliorer le confort de la vie quotidienne.

Cette thèse vise à résoudre le problème de la gestion de la qualité de contexte dans l'IdO. Nous considérons que les consommateurs et les producteurs sont des participants découplés. Les middlewares de gestion de QoC doivent ainsi s'adapter en permanence aux nouvelles exigences de qualité de contexte. Pour ce but, nous avons proposé une architecture logicielle pour un gestionnaire de qualité de contexte. En effet, il est nécessaire de développer des mécanismes pour trouver un compromis entre les garanties de sources de contexte et les exigences des applications d'IdO au niveau de la QoC.

Dans ce travail, nous avons proposé une solution qui est structurée sur deux contributions principales :

- Afin d'évaluer la QoC au niveau d'applications et les sources de contexte, nous avons proposé d'utiliser la logique floue comme une méthode fiable qui vise à traiter les notions imprécises.
- Nous avons considéré ainsi que le MapReduce Skyline comme une technique important pour accélérer le calcul et introduire le parallélisme dans le traitement.

Dans le chapitre 3, Nous avons posé plusieurs problématiques pour la réalisation de l'Internet des objets en termes d'objets hétérogènes à grande échelle, le passage à l'échelle, l'interopérabilité, la sécurité et la protection de la vie privée et autres problèmes qui nécessitent la conception d'une solution logicielle pour la gestion de contexte. Ce chapitre ainsi a présenté les deux grandes parties suivantes:

1. Un aperçu sur les efforts déployés pour résoudre le problème de gestion de contexte dans l'IdO selon un ensemble de critères bien ciblés. Nous examinons les projets INCOME, SITAC, FIWARE, CA4IoT, SAMURAI, HYDRA, AURA, SAI, WISeMid. Nous analysons aussi leur capacité à effectuer une gestion de contexte flexible, exhaustive et évolutive.
2. Une étude comparative exhaustive entre ces différentes approches étudiées. Cette étude montre la conformité de chaque solution avec la liste des critères que nous avons introduit dans le début de ce chapitre.

Dans le chapitre 4, nous avons décrit notre proposition basée sur la logique floue pour la gestion de la qualité de contexte (QoC) dans l'IdO. Ce mécanisme permet de sélectionner l'information la mieux adaptée à la demande d'utilisateur en terme de QoC. Pour rendre l'évaluation de la QoC basée sur la logique floue efficace, nous appliquons le paradigme MapReduce pour résoudre le problème de sélection des sources de contexte skylines. Nous accélérons le processus de sélection des sources skylines en explorant le parallélisme distribué qui permet de gérer des données massives.

Dans le chapitre 5, nous avons simulé le fonctionnement de gestionnaire de qualité de contexte proposé. Cette simulation commence par la présentation de requête de l'utilisateur jusqu'à la sélection de l'information adéquate à la demande de l'application avec des captures d'écrans. Nous avons montré ainsi l'efficacité de notre proposition en la comparant avec d'autres techniques (sans utiliser le

MapReduce Skyline et l'utilisation de MapReduce Skyline avec un seul niveau). L'expérimentation a démontré ainsi l'importance d'utiliser le MapReduce Skyline à cause d'augmentation exponentielles des données et des sources de contexte dans l'environnement d'IdO.

Pour valider l'approche proposée, nous l'avons comparé avec d'autres travaux voisins. La comparaison révèle l'importance de notre contribution présentée dans cette thèse par rapport aux autres approches notamment en terme de temps d'exécutions.

6.2 Perspectives

En plus de nos contributions à cette thèse, certaines défis doivent encore être examinés. En fait, tout au long du travail présenté dans cette thèse, nous avons formulé un certain nombre d'hypothèses qui peuvent être discutées et traitées dans les activités de recherche futures. Nous les énumérons brièvement dans ce qui suit:

- Nous visons à améliorer notre gestionnaire de qualité de contexte proposé, en accélérant le processus de fonctionnement de la logique floue pour l'évaluation de la QoC.
- Notre travail futur consiste à appliquer l'approche proposée sur un véritable ensemble de données.

Références bibliographiques

[1] Sam R., and al. (2012), "Vers une définition d'un système réparti multi-échelle", *8émes journées francophones Mobilité et Ubiquité*, pp.178-183, France.

[2] Charline D. (2011), "Les produits blancs connectés ont-ils un avenir auprès des particuliers ?", *Thèse professionnelle*, MBA spécialisés et formation continue du pôle universitaire léonard de Vinci.

[3] Gérald S., Sebastian L. (2008), "Internet of Things in 2020: A roadmap for the future", *Internet-Of-Things in 2020 EC-EPoSS Workshop Report*.

[4] Ovidiu V et al. (2011). Internet of Things Strategic Research Roadmap. Chapter In book: Internet of Things - Converging Technologies for Smart Environments and Integrated Ecosystems, Publisher: River Publishers, Editors: Ovidiu Vermesan, Peter Friess, pp.10-50.

[5] Charith P., and al. (2014) "Context Aware Computing for The IoTs: A Survey", *IEEE Communications Surveys & Tutorials Journal*, Vol.16, No.1, pp. 414 – 454.

[6] Yen-Kuang Ch. (2012), "Challenges and Opportunities of Internet of Things", *17th Asia and South Pacific Design Automation Conference*, pp.383-388, Australia.

[7] Mayank D., Jitendra K., Rajesh K. (2015), "Internet of Things and its Challenges", *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, pp. 810-814, India.

[8] Nearchos P., Avraam Ch., and George A.P. (2007), "Experiences from Developing a Distributed Context Management System for Enabling Adaptivity", *IFIP International Federation for Information Processing 2007*, pp: 225-238, Nicosia, Cyprus.

[9] Peter J. B., John D.B., Xian Ch. (1997), "Context-Aware Applications: From the Laboratory to the Marketplace", *IEEE Personal Communications Journal*, Vol: 4, No: 5, pp: 58 – 64, University of kent at Canterbury.

[10] Andy W., Alan J., Andy H. (1997), "A new location technique for the active office", *IEEE Personal Communications Journal*, Vol: 4, No: 5, pp: 42 – 47, University of CAMBRIDGE.

[11] Jason P. (1998), "Adding Generic Contextual Capabilities to Wearable Computers", *ISWC '98 Proceedings of the 2nd IEEE International Symposium on Wearable Computers*, pp: 92-99, USA.

[12] Maja V. (2007). "Context aware service composition", *Technical Report*, Cambridge United Kingdom.

[13] Anind K. D., Gregory D. A., and Daniel S. (2001). "A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications", *Human-computer interaction Journal*, Vol: 16, No: 2, pp: 97-166.

[14] Tarak Ch., Frédérique L., André F. (2005). "Adaptation des applications au contexte en utilisant les services WEB", *ConferenceUbiMob'05*, pp: 111-118, Grenoble, France.

[15] Thomas B., Axel K¨ U., Michael Sch. (2003). "Quality of Context: What It Is And Why We Need It", In *Proceedings of the 10th Workshop of the OpenView University Association: OVUA'03*. pp: 1-14, Germany.

[16] Mario F., Luca F., Antonio C., Azzedine B. (2011). "QoC-based Context Data Caching for Disaster Area Scenarios", *2011 IEEE International Conference on Communications (ICC)*, pp: 1-5, Japan.

[17] Michael K., Iris H. (2005). "Challenges in Modelling and Using Quality of Context (QoC)". *International Workshop on Mobile Agents for Telecommunication Applications MATA 2005: Mobility Aware Technologies and Applications*, pp: 324–333, Germany.

[18] Atif M., Hong-Linh T., and Schahram D. (2008). "On the Evaluation of Quality of Context", *European Conference on Smart Sensing and Context EuroSSC 2008: Smart Sensing and Context*, pp 140-153. Berlin.

[19] Schahram D., Van Hai Do., Atif M. (2010). "Quality of Context in Pervasive Systems: Models, Techniques, and Applications", *Conference proceedings*, pp.1-155. Wien.

[20] Debora Cabral N., Mario Antonio Ribeiro D., Jose Leomar T. (2014). "Context management: toward assessing quality of context parameters in a ubiquitous ambient assisted living environment", *Journal of Information Systems and Technology Management*, Vol. 11, No. 3, pp. 569-590.

[21] Anind K. Dey. (2000). "Providing Architectural Support for Building Context-Aware Applications", *Doctoral Dissertation*, USA.

[22] Débora Cabral N., Adroaldo de A., Lucas B., Willian Romeu R., José Leomar T., Mário Antônio Ribeiro D. (2015). "An Enhanced Quality of Context Evaluating Approach in the e-Health Sensor Platform", *Q2SWinet '15 Proceedings of the 11th ACM Symposium on QoS and Security for Wireless and Mobile Networks*, pp.1-7, Mexico.

[23] Léon L., Pierrick M., Denis C, Sophie Ch., Thierry D., Atif M. (2016). "Enhancing Context Data Distribution for the Internet of Things Using Qoc-awareness and Attribute Based Access Control", *Annals of Telecommunications Journal*, Vol. 71, No. 3-4, pp.121-132.

[24] Pierrick M, Thierry D., Sophie Ch., Michelle S., Chantal Ta. (2015). "From Ambient Sensing to IoTbased Context Computing: An Open Framework for End to End QoC Management", *Sensors Journal*, Vol. 15, No. 6, pp. 14180-14206.

[25] Bill N. Schilit., Marvin M. Th. (1994). "Disseminating active map information to mobile hosts", *IEEE Network: The Magazine of Global Internetworking Journal*, Vol.8, No.5, pp. 22-32.

[26] Daniel S., Anind K. Dey., Gregory D.A. (1999). "The Context Toolkit: Aiding the Development of Context-Enabled Applications", *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 434-441.

[27] Nick R., Jason Pa., David M. (1998). "Enhanced Reality Fieldwork: the Context Aware Archaeological Assistant", *Proceedings of the 25th Anniversary Conference*. pp. 269-274, Canterbury.

[28] P.J. Brown. (1995). "The stick-e document: a framework for creating context-aware applications", *Proceedings of EP''96*, pp. 259–272, Canterbury.

[29] Anind K. Dey., Gregory D.A. (1999). "Towards a Better Understanding of Context and Context-Awareness", *HUC '99 Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, pp 304-307, Berlin.

[30] Patrick B., Avelino J. Gonzalez. (2014). "Context in Computing: A Cross-Disciplinary Approach for Modeling the Real World", *Context in Computing*, Springer-Verlag New York.

[31] Wan-rong J., Chi-Chia H., Jane Yung-jen H. (2009). "Context Life Cycle Management in Smart Space Environments", *Proceedings AUPC'09*, pp. 9-14, Taiwan.

[32] Xin Li, Martina E., José-Fernán Martinez., Gregorio Rubio. (2015). "Context Aware Middleware Architectures: Survey and Challenges", *Sensors Journal*, Vol.15, No.8, pp. 20570-20607.

[33] Benjamin B. (2015). "Système de gestion de flux pour l'Internet des objets intelligents". *Thèse de doctorat*, Université de Versailles Saint-Quentin-En-Yvelines.

[34] Ahmed Malek N. (2014). "L'intelligence ambiante et les systèmes de transport intelligents", *Diplôme MAGISTER*, Université Badji Mokhtar Annaba, Algérie.

[35] Benoît P. (2012). "Le Machine to machine : premier pas vers l'internet des objets", *Technical report*.

[36] Soma B. (2011), "Role of middleware for IoTs: a study", *International Journal of Computer Science & Engineering Survey (IJCSES)*, Vol.2, No.3, pp. 94-105.

[37] Qi Jing., Athanasios V.Va., Jiafu W., Jingwei Lu., Dechao Qiu. (2014). "Security of the Internet of Things: perspectives and challenges", *Wireless Networks journal*, Vol. 20, No.8, pp. 2481–2501.

[38] Daniele M., Sabrina S., Francesco De P., Imrich Ch. (2012). "Internet of things: Vision, applications and research challenges", *Ad Hoc Networks Journal*, Vol.10, No.7, pp.1497-1516.

[39] Jun young K., (2015), "Secure and Efficient Management Architecture for the IoTs", *proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, School of Computer Science and Engineering, pp. 499-500, UNSW Australia.

[40] Luigi A., Antonio I., and Giacomo M., (2010). "The IoTs: A survey", *Computer Networks Journal*, *Vol.54*, No.15, pp.2787-2805.

[41] Jean-Paul A., and al, (2014), "Projet INCOME: INfrastructure de gestion de COntexte Multi-Echelle pour l'Internet des Objets", *In Conférence Francophone sur les Architectures Logicielles (CAL)*, pp. 1–2, Université de Toulouse.

[42] Pierrick M., (2015), "Gestion de bout en bout de la Qualité de Contexte pour l'Internet des Objets: le cadriciel QoCIM", *Thèse doctoral*, Université Toulouse.

[43] Ghofrane F. (2015), "Middleware for IoTs: a study", *International Conference Distributed Computing in Sensor Systems (DCOSS)*, pp. 100–112, Tunisia.

[44] Sam Ro., (2015), "Modèles, méthodes et outils pour les systèmes répartis multiéchelle", *Thèse doctoral*, Télécom SudParis école doctorale S&I avec l'Université d'Évry-Val-d'Essonne.

[45] Raja B., (2015), "Déploiement de systémes répartis multi-échelles processus, langage et outils intergiciels", *Thèse Doctoral*, Université de Toulouse.

[46] Jean-Paul A., Amel B., Valérie Ca., Marie-F., Coise Canut., Sophie Ch., Denis C, Thierry D, Romain L., Emmanuel L., Sébastien L., Hervé M., André P., Chantal T., Pascale Z. (2012). "INCOME – Multi-scale Context Management for the Internet of Things", *International Joint Conference on Ambient Intelligence AmI*, pp. 338-347.

[47] Jean-Paul A. and al. (2014). "Gestion de contexte multi échelle pour l'Internet des objets", *Journées francophones Mobilité et Ubiquité*, pp.89-100, France.

[48] Sam R., Sébastien L., Chantal T. (2014). "MuSCa: A multiscale characterization framework for complex distributed systems", *2014 Federated Conference on Computer Science and Information Systems*, pp. 1657–1665, Poland.

[49] Pierrick M., Thierry D., Sophie Ch., Michelle S. (2013). "QoCIM: A Meta-model for Quality of Context", *International and Interdisciplinary Conference on Modeling and Using Context CONTEXT 2013: Modeling and Using Context*, pp. 302-315.

[50] Samer Machara Ma. (2016). "Models and algorithms for managing quality of context and respect for privacy in the Internet of Things", *Thése de Doctorat* , Université Paris-Saclay, Telecom SudParis.

[51] Celestino M., Manuel O., Joaquim B., Tipu Arvind R., Jonathan Ro. (2014). "Social Networks and Internet of Things, an Overview of the SITAC Project", *International Wireless Internet Conference WICON 2014: Wireless Internet.*

[52] Celestino M., Manuel O., Joaquim B., Tipu R., Jonathan R. (2014). "Social Networks and Internet of Things, an Overview of the SITAC Project", *International Wireless Internet Conference WICON 2014: Wireless Internet*, pp 191-196, Aveiro, Portugal.

[53] http://sitac.wp.tem-tsp.eu/

[54] FIWARE Foundation: from research to those digital services you will love ,https://ec.europa.eu/digital-single-market/en/news/fiware-foundation-research-those-digital-services-you-will-love.

[55] Martínez R., Darío. (2015). "Internet of things virtualization and identity management via fiware generic enablers". Springer, Cham, Madrid.

[56] https://fr.slideshare.net/fermingalan/fiware-managing-context-information-at-large-scale.

[57] IoT Tutorial-Orion context Broker & Arduino, https://www.fiware.org/2015/06/10/iot-tutorial-orion-context-broker-arduino/.

[58] FIWARE NGSI APIv1 Walkthrough, http://fiware-orion.readthedocs.io/en/latest/user/walkthrough_apiv1/.

[59] Perera, Ch., Zaslavsky A., Christen P., Georgakopoulos D. (2012). "CA4IOT: Context Awareness for Internet of Things", *Proceedings of the IEEE International Conference on Green Computing and Communications, Conference on Internet of Things, and Conference on Cyber,Physical and Social Computing (iThings/CPSCom/GreenCom), IEEE Computer Society Washington*, pp. 775-782, USA.

[60] Davy P., Yolande B. (2014), "SAMURAI: A Streaming Multi-tenant Context-Management Architecture for Intelligent and Scalable Internet of Things Applications", *IE '14 Proceedings of the 2014 International Conference on Intelligent Environments*, IEEE Computer Society Washington, pp. 226-233, USA.

[61] José R. H., Jesus Garcia-M., Juan A.B, Davy P. (2016). "A Model-Driven Approach for Quality of Context in Pervasive Systems", *Computers & Electrical Engineering Journal*, Vol.55, pp. 39-58, USA.

[62] Sarnovsky M., butka P., kostelnik P., lackova D. (2007), "HYDRA – Network Embedded System Middleware for Ambient Intelligent Devices", *In: ICCC'2007: Proceedings of 8th International Carpathian Control Conference*, pp. 611-614.

[63] Shirin Z. (2013), "Middleware for internet of things". *Thesis doctoral*, University of Twente.

[64] Tomas S., Jan Hreno. (2013). "Semantic middleware in Applications of the Internet of Things", *Conference proceedings*, pp. 24–32.

[65] João Pedro S, David G. (2002), "Aura: An Architectural Framework for User Mobility in Ubiquitous Computing Environments", *Working Conference on Software Architecture WICSA 2002: Software Architecture*, pp.29-43.

[66] Sallauddin M., Mohammad S., Shabana. (2016). "IoT Middleware for Device Privacy on Big Data", *International Journal of Innovative Research in Science*, Engineering and Technology, Vol.5, No.6, pp. 10266- 10273.

[67] David P., Federica P., Dino G., Agostino L. (2010), "A Scalable Grid and Service-Oriented Middleware for Distributed Heterogeneous Data and System Integration in Context-Awareness Oriented Domains", *The Internet of Things: 20th Tyrrhenian Workshop on Digital Communications*, pp.109-118.

[68] Jeisa P.D., Antonio V.L. Damaso., Nelson S. Rosa. (2010). "WISeMid: Middleware for Integrating Wireless Sensor Networks and the Internet", *IFIP International Conference on Distributed Applications and Interoperable Systems DAIS 2010: Distributed Applications and Interoperable Systems*, pp. 70-83.

[69] Retima F., Benharzallah S., Kahloul L., and Kazar O., (2017), "A Comparative Analysis of Context Management Approaches for the Internet of Things" ,*The International Arab Journal of Information Technology*, Vol. 14, No. 4A, pp.578-585.

[70] Abid Z. (2012). "Gestion de la Qualité de Contexte Pour L'intelligence ambiante", *Ph.D Thesis*, Telécom SudParis et Université d' Evry-Vald'Essonne.

[71] Becker S., Blessing A., Durr F., Rothermel K. (2010). "Reference Model for the Quality of Context Information," *Technical Report*.

[72] Dustdar S., Manzoor A. (2010). "Quality of Context in Pervasive Systems: Models, Techniques, and Applications", *Ph.D Thesis*, Technical University of Vienna.

[73] Gu T., Wang X., Pung H., Zhang D. (2004). "An Ontology-based Context Model in Intelligent Environments", *In proceedings of Communication Networks and Distributed System Modeling and Simulation Conference*, Singapore, pp. 270-275.

[74] Preuveneers D., Berbers Y. (2006). "Quality Extensions and Uncertainty Handling for Context Ontologies", *in Proceedings of Context and Ontology Conference*, Trentino.

[75] Ranganathan A., Al-Muhtadi J., Campbell R. (2004). "Reasoning about Uncertain Contexts in Pervasive Computing Environments", *IEEE Pervasive Computing*, Vol. 3, No. 2, pp. 62-70.

[76] Razzaque M., Dobson S., and Nixon P. (2006), "Categorization and Modeling of Quality in Context Information", *Trusted and Autonomic Computing journal*.

[77] Zimmermann A.(2007), Context Management and Personalisation: a Tool Suite for Context-and User-Aware Computing, Shaker Verlag GmbH, Germany.

[78] José Ramón Hoyos Barceló. (2015). "A Model-Driven Approach for Context Modeling in Context-Aware Systems", *A dissertation for the degree of Doctor of Philosophy in the subject of Computer Science*, University of Murcia.

[79] Chantal T., Zakia Kazi-A. (2010). "Building Context-Awareness Models for Mobile Applications", *Journal of Digital Information Management*, Vol. 8, No.2, pp.78-87. France.

[80] Marten J. van Sinderen. (2004). "Model-Driven Architecture with Emphasis on Industrial Applications", *CTIT Technical Report TR-CTIT-04-12*, University of Twente.

[81] http://www.ferdinandpiette.com/blog/2011/08/la-logique-floue-interets-et-limites/.

[82] Tarek B., Sidi-Mohammed S. (2015). "A Fuzzy Logic-Based Communication Medium Selection for QoS Preservation in Vehicular Networks", *IEEE International Conference on Communications (ICC)*, pp.101-108, Malaysia.

[83] Ying B., Dali W. (2006). "Fundamentals of Fuzzy Logic Control – Fuzzy Sets, Fuzzy Rules and Defuzzifications", *Advanced Fuzzy Logic Technologies in Industrial Applications*, Springer, pp.17-36, Berlin.

[84] https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligence_fuzzy_logic_systems.htm.

[85] Ibrahim Abaker Targio H., Nor Badrul A., Abdullah G., Ibrar Y., Feng X, Samee Ullah Kh. (2016). "MapReduce: Review and open challenges", *Scientometrics Journal*, Vol.109, No.1, pp.389–422.

[86] Jeffrey D., Sanjay Gh. (2008). "MapReduce: Simplied Data Processing on Large Clusters", *OSDI'04 proceeding of the 6th conference on symposium on operating systems design & implementation*, pp.1-13, San Francisco.

[87] Stephan B., Donald K., Konrad S. (2001). "The Skyline Operator", *In Proc of the 17th International Conference on Data Engineering*, pp. 421–430. Germany.

[88] Yuanyuan L., Zhiyang Li., Mianxiong D., Wenyu Q., Changqing Ji., Junfeng W.(2016). "Efficient Subspace Skyline Query based on User Preference using MapReduce", *Ad Hoc Networks Journal*, Vol.35, pp.105-115.

[89] Kasper M., Jens Laurits P., Hua L., Yongluan Z. (2014). "Efficient Skyline Computation in MapReduce", *17th International Conference on Extending Database Technology (EDBT)*, pp.37-48, Athens, Greece.

[90] Akrivi V., Christos D., Yannis K. (2008). "Angle-based Space Partitioning for Efficient Parallel Skyline Computation", *SIGMOD'08 proceedings of the 2008 ACM SIGMOD international conference on management of data*, pp. 227-238.

[91] Stella Maropaki. (2013). "Parallel Skyline Computation on Map -Reduce Systems", *Thesis doctoral*, Technical University of Crete.

[92] Christos K., Theodoros T. (2017). "A Survey of Skyline Query Processing", *computing research repository journal (CoRR)*.

[93] Kian-Lee T., Pin-Kwang E., Beng Chin O. (2001). "Efficient Progressive Skyline Computation", *VLDB'01 proceedings of the 27th international conference on Very Large Data Bases*, pp.301-310.

[94] Jan Ch., Parke G., Jarek Gryz., Dongming L. (2005). "Skyline with Presorting: Theory and Optimizations", *in proceedings of Intelligent Information Processing and Web Mining*, pp.595-604.

[95] Jan Ch., Parke G., Jarek G., Dongming Li. (2002). "Skyline with Presorting", *Proceedings 19th International Conference on Data Engineering*, pp. 1-3, India.

[96] Parke G., Ryan S., Jarek G. (2005). "Maximal vector computation in large data sets", *VLDB'05 Proceedings of the 31 st international conference on Very large data bases*, pp. 229–240, Trondheim, Norway.

[97] Patrick Kamnang W. (2017). "Optimisation des requêtes skyline multidimensionnelles", *thesis doctoral*, University of Bordeaux.

[98] Donald K, Frank R, Steffen R. (2002). "Shooting Stars in the Sky: An Online Algorithm for Skyline Queries", *Proceedings of the 28th VLDB Conference*, pp. 1-12, Hong Kong, China.

[99] Dimitris Papadias, Yufei T., Greg Fu., Bernhard S. (2003). "An Optimal and Progressive Algorithm for Skyline Queries", *SIGMOD'2003 Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pp. 467-478 San Diego, California.

[100] Manzoor A., Truong H., Dustdar S. (2014). "Quality of Context: Models and Applications for Context-aware Systems in Pervasive Environments", *The Knowledge Engineering Review Journal*, Vol. 29, No. 2, pp.154-170.

[101] Parina A., Nima Jafari N. (2016). "Trust evaluation between the users of social networks using the quality of service requirements and call log histories", *Information & Knowledge Management*, Vol. 45, No.10, pp.1505-1523.

[102] Kuyoro Shade O., Awodele O. Akinde Ronke O. and Okolie Samuel O. (2012). "Quality of Service (Qos) Issues in Web Services", *IJCSNS International Journal of Computer Science and Network Security*, Vol.12, No.1, pp. 1-4, Babcock University, Nigeria.

[103] Bijay K. J., Peter C. Patton., Ernest H. Forman. (2007). The Analytic Hierarchy Process (AHP) in Software Development, Prentice Hall.

[104] Arun S., Rajesh K., P. S. Grover. (2008). "Estimation of Quality for Software Components – an Empirical Approach", *SIGSOFT Software Engineering Notes Journal*, Vol.33, No.6, pp.1-10.

[105] http://info-acse.blogspot.com/

[106] Retima F., Benharzallah S., Kahloul L., and Kazar O., (2018), "A Quality-Aware Context Information Selection Based Fuzzy Logic in IoT Environment", *The International Arab Journal of Information Technology*, Vol. 15, No. 3A, pp. 522-531.

[107] Yahyaoui H., Almulla M., Maamar Z. (2014). "A Fuzzy Model for Selecting Social Web Services", *in Proceedings of the International Conference on Web Information Systems Engineering*, pp. 32-46, Wise.

[108] http://www.enseignement.polytechnique.fr/informatique/profs/Julien.Cervelle/eclipse/.

[109] http://dept-info.labri.fr/ENSEIGNEMENT/programmation2/intro-eclipse/.

[110] Kertiou I., Benharzallah S., Kahloul L., Beggas M., Euler R., Abdelkader L., Bounceur A. (2018), "A dynamic skyline technique for a context-aware selection of the best sensors in an IoT architecture", *Ad Hoc Networks journal*, Vol.81, pp. 183-196.

[111] Charith P., Arkady Z., Peter Ch., Michael C., Dimitrios G. (2013), "Context-aware Sensor Search, Selection and Ranking Model for Internet of Things Middleware", *IEEE 14th International Conference on Mobile Data Management*, pp. 314-322, Italy.

[112] Charith P., Arkady Z., Chi Harold L., Michael C., Peter Ch., Dimitrios G. (2014), "Sensor Search Techniques for Sensing as a Service Architecture for The Internet of Things", *IEEE Sensors Journal*, Vol: 14 , No: 2, pp. 406 - 420.

[113] Mohammad E., Elaheh Sh., Raymond K.W., Simon F., Jinan F. (2017), "An adaptive meta-heuristic search for the internet of things", *Future Generation Computer Systems Journal*, Vol: 76, pp. 486-494.

[114] Varsha D. (2017), "An Efficient Clustering of Sensors using a Meta Heuristic Algorithm for IoT", *International Journal of Computer Applications*, Vol.173, No.8.