République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et Recherche Scientifique
Université Saad Dahlab de Blida 1

Faculté des sciences
Département d'Informatique
En vue d'obtenir le diplôme de master
Domaine : Mathématique et informatique
Filière : Informatique
Spécialité : Informatique
Option : Ingénieure de logiciel

La sélection des entités logicielles adaptatives pour la composition des applications mobiles.

Présenté par :
- HIFI Amira
- BOUSBAA Amine

Promoteur :
- Dr. AFRAH

Soutenu le :

Devant le jury :

-M.GHEBGHOUB	Président
-M. NASSRIE	Examinateur
-M. afrah	Promoteur

Remerciements

Tout d’abord, nous remercieront Allah de nous avoir aidé et donné la force et la volonté de réaliser ce travail.

Ensuite, nous tenons à exprimer nos plus vifs remerciements et gratitude à notre promotrice Md DJEDDAR pour son encadrement continu malgré les complications de santé quel a eu au coure de cette année universitaire, et pour les remarques constructives qu’elle nous a fournies ainsi que pour ses précieuse conseils durant toute la période de notre travail. On la remercie également pour la confiance qu’elle nous a accordée et pour la grande liberté d’idées et de travail qu’elle nous a donnée. Nous n’oublierons pas aussi de la remercier pour ses qualités humaines, son hospitalité et son soutien qui ont permis de bien mener cet ouvrage.

Nous tenons à remercier Md Nacrie et Md Rebghoube les membres du jury d'avoir bien voulu participer à l'évaluation de ce travail.

Quelques personnes ont contribué à la réalisation de ce travail et méritent des remerciements.

Nous sommes également très redevables à nos familles qui croyait en notre capacité à réaliser quelque chose de grand. En nous donnant l'opportunité de poursuivre ce rêve et de le réaliser. Et pour leur encouragement, leur aide et leur grande patience avec nous.

Avec un cœur plein de gratitude MERCI.

ملخص

تطوير تطبيقات الهاتف المحمول المخصصة استنادًا إلى آلية التكوين (أي باستخدام كيانات البرمجيات الحالية) تلقى الكثير من الاهتمام في العامين الماضيين. يوضح عدم تجانس الأجهزة المحمولة أن متطلبات قابلية النقل تلعب دورًا مهمًا في مجال تطوير تطبيقات الأجهزة المحمولة. خلاف ذلك، تطبيقات الهاتف المحمول تعتمد بقوة على ميزات بيئة التنفيذ .وبالتالي، من أجل التأكد من النشر الصحيح والتشغيل الصحيح للتطبيق المحمول المركب، من الضروري التأكد من أن مكوناتها قابلة للتكيف مع السياق الحالي للجهاز المحمول .للتعامل مع هذه القضية ويرجع ذلك إلى حقيقة أنه يمكن استخدام العديد من كيانات البرامج لتنفيذ المحدد لمتطلبات التطبيق المحمول المطلوب نقترح في هذه الورقة خوارزمية اختيار تعتمد على السياق تهدف إلى اختيار كيانات البرامج التكيفية بين جميع الكيانات المماثلة. أيضا، يستهدف تحديد مسارات تكوين مختلفة لإنشاء تطبيقات محمولة مخصصة .لتحقيق هذا الهدف، نقترح أوصاف تستند إلى علم الوجود لتحديد سياق كيانات البرامج المقابلة وبيئة التنفيذ

الكلمات الرئيسية -تطبيقات الهاتف المحمول، خوارزمية الاختيار، السياق، علم الوجود، مسار التكوين.

Résumé

Le développement d'applications mobiles personnalisées en se basant sur le mécanisme de composition entités (en utilisent des entités existantes) a suscité beaucoup d'attention au cours des deux dernières années. L'hétérogénéité des appareils mobiles montre que la portabilité des exigences joue un rôle important dans le domaine de développement d'applications mobiles. Par ailleurs, les applications mobiles sont fortement dépendantes des fonctionnalités de l'environnement d'exécution. Ainsi, dans l'ordre de s'assurer du bon déploiement et du bon fonctionnement de l'application mobile composite, il est nécessaire de s'assurer que leurs électeurs sont adaptables au contexte actuel de l'appareil mobile. Pour faire face à ce problème et en raison du fait que plusieurs entités logicielles peuvent être utilisé pour mettre en œuvre les exigences pour une application mobile souhaitée, nous proposons dans ce papier un algorithme de sélection basé sur le contexte qui vise à sélectionner les entités logicielles adaptatives parmi toutes celles correspondantes. En outre, il vise à déterminer les différentes compositions possibles de chemins pour créer des applications mobiles personnalisées. Pour y parvenir à cet objectif, nous proposons des descriptions basées sur des ontologies pour définir les contextes des entités logicielles correspondantes et l'environnement de l'exécution.

Mots-clés— Applications mobiles, algorithme de sélection, contexte,

Ontologies, Chemins de composition.

Abstract

The development of customized mobile applications basing on the composition mechanism (i.e. using existing software entities) has received a lot of attention in the last couple of years. The mobile devices heterogeneity shows that the portability requirements play an important role in the mobile applications development domain. Otherwise, mobile applications strongly depend on the execution environment features. Thereby, in order to make sure the correct deployment and the proper functioning of the composite mobile application it is necessary to ensure that their constituents are adaptable to the current context of the mobile device. To cope with this issue and due to the fact that several software entities can be used to implement the identified requirements for a desired mobile application, we propose in this paper a context-driven selection algorithm that aims at selecting the adaptive software entities among all corresponding ones. Also, it targets to determine the different possible composition paths to build customized mobile applications. To achieve this objective, we propose ontology based descriptions to define the context of the corresponding software entities and the execution environment.

Keywords— Mobile applications, Selection algorithm, Context, Ontology, Composition path.

Table des matières

Liste des figures

Liste de tableau

Introduction générale

Depuis ces dernières années, la technologie mobile est en train de devenir de plus en plus répandue, et les recherches dans ce domaine sont de plus en plus nécessaire [1].

les appareils mobiles disponibles sur le marché tels que les smartphones et les tablettes sont caractérisés par plusieurs caractéristiques hétérogènes [2] qui sont importantes : vie de la batterie, capacité de stockage, modes de saisie, etc. l'adoption massive et l'utilisation des appareils mobiles pour effectuer nos tâches dans la vie quotidienne est juste une démonstration de leur croissance et de la demande croissante de nouvelles applications mobiles. Alors et, afin de répondre aux besoins des utilisateurs et de prendre avantages des services existants, composant de nouveaux applications soumet à plusieurs contraintes du mobile environnement.

L'hétérogénéité des appareils mobiles et la limitation ressources offertes par eux (c'est-à-dire la puissance de calcul limitée, les limites de la connexion réseau, la faible capacité de stockage, la taille de l'écran, etc.) [4] créent des difficultés et donne à plusieurs défis avec la composition d'applications mobiles. Le point le plus important est la portabilité des applications obtenues sur les appareils mobiles hétérogènes [3] puisque ce n'est pas économiquement viable pour produire des applications logicielles pour quelques périphériques (c'est-à-dire limiter la convivialité des applications mobiles). [5]

Face à ce problème, le développeur mobile devra créer plusieurs versions de l'application mobile souhaitée, où chaque d'entre eux seront spécifiques à un environnement mobile spécifique (c.-à-d. contexte spécifique). Ainsi, pour composer des applications mobiles adaptatifs ; nous devons toujours traiter avec soin les caractéristiques de différents appareils mobiles au moment de sélection des entités constitutives du logiciel.

Généralement, une technique de sélection est basée sur certains critères ou des métriques. Dans nos travaux de recherche, nous proposons de classer ces critères à l'intérieur de deux catégories :

a) les contraintes d'exécution (tels que le niveau de la batterie, la taille de la mémoire, la disponibilité de certains appareils (Wi-Fi, GPS, etc.), afin que nous puissions sélectionner les entités logicielles pouvant être déployées correctement et fonctionner correctement sur le périphérique cible. Ce genre de critère représente les conditions nécessaires pour que l'entité puisse remplir sa tâche.

b) caractéristiques de qualité (telles que : la force du signal du réseau en fonction du périphérique mobile réseau, le temps de réponse en fonction de la vitesse du processeur, etc.), de manière à assurer une meilleure performance et une bonne qualité d'application obtenue. Plusieurs auteurs ont abordé articles et points de vue sur la tâche de sélection dans le domaine mobile [6] [7] [9] [10], où la plupart d'entre eux se sont concentrés essentiellement sur les caractéristiques de qualité pour effectuer la sélection tâche, juste dans le but de réaliser une meilleure application performances (par exemple performances du réseau [6]

   .Même si, plusieurs faits importants doivent également être pris en compte, en particulier et d'une manière importante pour assurer le bon déploiement et le bon fonctionnement de l'application telle que comme : la taille de la mémoire, la taille de l'écran, le niveau de la batterie.

Pour cette raison et ainsi pour atteindre notre objectif, qui est de fournir une application mobile composite compétente et efficace qui fonctionne correctement dans un appareil mobile spécifique, nous proposons une description basée sur une ontologie pour le contexte de chaque cible appareil mobile et entités logicielles adéquates. Sur la base de ces modèles d'ontologies, nous proposons un algorithme de sélection qui vise à

sélectionner les entités logicielles contextuelles, mais aussi à déterminer les différents chemins de composition possibles pour obtenir une application mobile spécifique.

CHAPITRE I : Appareils et applications mobiles

I.1 INTRODUCTON

Les utilisateurs sont de plus en plus mobiles et veulent pouvoir accéder à leurs systèmes d'information quel que soit le lieu où ils se trouvent. Pour ce faire, plusieurs appareils mobiles ont été développés pour soutenir ces systèmes afin de répondre aux exigences des utilisateurs en mobilité. Les constructeurs de ces appareils mobiles, notamment les Smartphones, fournissent des appareils ayant tous des architectures matérielles et logicielles différentes. Cependant, le développement d'applications mobiles se heurte au frein majeur qui est l'hétérogénéité des appareils mobiles et les ressources limitées offertes par eux. Pour cela, avant de se lancer dans le développement d'une application mobile, il est important de connaitre les caractéristiques techniques et logicielles de l'appareil sur lequel elle va s'exécuter. Donc, elle doit être adaptée à son contexte d'utilisation afin de pouvoir répondre correctement aux besoins souhaités. Ce chapitre vise à faire une présentation des notions clés permettant de définir et comprendre le cadre général de note travail de recherche à savoir les applications mobiles, les appareils mobiles et la description du contexte.

I.2 Les Appareils mobiles

Le domaine du mobile est devenu, depuis le 28 novembre 2007, avec la sortie pour le grand public des smartphones, un domaine de l'informatique à part entière. Le premier smartphone qui a révolutionné ce domaine est l'iPhone d'Apple. Aujourd'hui, la population mondiale est mieux équipée en smartphones et tablettes qu'en ordinateur de bureau. Le cabinet Gartner a publié une étude en 2013 1 montrant qu'en 2014, le nombre de smartphones et tablettes vendus dans le monde,

I.2.1 L'hétérogénéité des appareils mobiles

Les entreprises de développement d'applications mobiles se heurtent à deux freins majeurs dans le déploiement de leurs applications. Le premier est l'hétérogénéité des architectures matérielles sur lesquelles leurs applications peuvent être exécutées. Le deuxième est l'hétérogénéité des logiciels installés sur les smartphones. Les constructeurs de smartphone, parmi lesquels les plus populaires sont Samsung, Apple Huawei et LG 1, fournissent des smartphones ayant tous des architectures matérielles différentes. Ainsi, les smartphones ont tous des processeurs différents, des écrans différents (résolution, taille), des caméras différentes (résolution) ... Même s'il existe beaucoup de différences matérielles, celles-ci sont en partie cachées par le système d'exploitation installé sur le smartphone. Par exemple, Android est installé sur une multitude d'appareils. Les entreprises doivent donc uniquement implémenter leurs applications pour Android et non pour un smartphone en particulier. Malgré cela, toutes les différences matérielles doivent être prises en compte lors de la conception et le développement d'une application. Il faut toujours se souvenir qu'une application mobile peut être exécutée sur des smartphones plus ou moins performants. Ainsi, une application permettant de faire de la reconnaissance optique de caractères en temps réel ne fonctionnera pas si la caméra du smartphone sur lequel elle s'exécute a une résotion trop faible. De la même manière, une application ne doit pas lancer un traitement de plusieurs centaines de milliers d'itérations sur un smartphone ayant un processeur trop lent.

I.2.1.1 Les caractéristiques matérielles

Écran : La taille d'un écran est exprimée en pouces (1 pouce = 2,54 cm), elle mesure la diagonale d'un écran. Les smartphones ont généralement un écran entre 3,5 et 5,5 pouces. La taille de l'écran est un élément déterminant lors du choix d'un appareil puisqu'elle a un impact sur la taille du smartphone. Un écran plus grand facilite la lecture et est plus confortable pour jouer ou regarder un film. Cependant, les plus grands modèles sont difficiles à manipuler à une main et surtout, ne peuvent pas toujours se glisser dans

une poche. Autre critère important : la résolution, c'est-à-dire le nombre de points composant une image. Plus la résolution est élevée, meilleure est la qualité d'affichage

Mémoire : Les appareils mobiles ont une mémoire sur laquelle vous pouvez stocker vos applications, vos photos, votre musique… Elle peut varier fortement d'un modèle à l'autre : les modèles de base ont généralement 4 gigas alors que pour les modèles plus couteux, cela peut monter à 64 gigas. Si vous pensez stocker des photos, des vidéos et des fichiers musicaux, n'oubliez pas de vérifier si vous pouvez ajouter une carte mémoire (ex. : une carte microSD) pour augmenter la capacité de stockage.

Puissance : Comme les ordinateurs, les smartphones ont un processeur. Plus celui-ci est puissant, plus les applications s'exécutent rapidement. C'est un critère important pour ceux qui aiment jouer sur leur smartphone. Généralement, les smartphones bon marché ont un processeur moins puissant ou plus ancien.

Bluetooth : est une norme de connexion sans fil peu énergivore, de courte portée et qui offre une bande passante de 1 Mb/s. Ce standard de transfert a pour objectif de transférer de fichiers d'informations, d'agendas électroniques, flux audio entre un téléphone et un casque, interconnexion sans fils des lecteurs mp3 à d'autres dispositifs pour le téléchargement, etc. Un appareil Bluetooth est caractérisé généralement par deux informations : la version de la norme et sa classe 1 . Il définit aussi un certain nombre de profils d'applications (i.e. Bluetooth profiles 2) permettant aux appareils Bluetooth de se connecter entre eux d'une façon spécifique. Donc, en plus de la classe et la version de la norme d'un appareil Bluetooth, les appareils mobile permettent aussi se différencier en terme de gestion de Profils Bluetooth. A titre d'exemple, l'iPhone et l'iPad Touch qui sont dotés par la version 8 d'iOS peuvent prendre en charge plusieurs Profils Bluetooth. Un tableau comparatif et illustratif présenté en (Apple, 2015) fournit des informations sur les profils pris en charge par chaque appareil.

L'autonomie : L'avantage d'un smartphone c'est que l'on peut faire beaucoup de choses avec. Malheureusement, vous risquez d'épuiser rapidement la batterie. Comparez

donc l'autonomie proposée par les différents modèles. L'utilisation intensive d'un smartphone peut vous amener à recharger votre appareil tous les jours ! Pour économiser votre batterie, vous pouvez par exemple diminuer un peu la luminosité de votre écran ou désactiver certaines fonctionnalités gourmandes comme le WiFi, la 3G, le GPS…

I.2.1.2 Les caractéristiques Logicielles

Comme les ordinateurs, les tablettes et les Smartphones fonctionnent grâce à un système d'exploitation. Le système d'exploitation (OS en anglais) est l'interface qui permet de faire le lien entre l'utilisateur et son smartphone. Il existe plusieurs systèmes d'exploitation pour les smartphones. Si vous comparez 2 systèmes d'exploitation, vous verrez donc qu'il y a des différences au niveau de l'interface, des menus, des icônes… Même en comparant 2 smartphones avec le même système d'exploitation, vous verrez probablement des différences. En effet, ces systèmes ont évolué avec le temps, il existe donc différentes versions d'un même système. Chaque système d'exploitation dispose également de son propre « Store », c'est-à-dire une boutique en ligne où vous pouvez télécharger de nouvelles applications. Voici les principaux systèmes d'exploitation :

Android : Android est le système d'exploitation le plus répandu. Développé par Google, il n'est pas rattaché à une marque de smartphones (contrairement à iOS). Vous avez donc un large choix de marques qui proposent ce système : Samsung, Acer, HTC, Alcatel, LG, Huawei, Sony… Les fabricants intègrent Android à leurs smartphones et y apportent quelques modifications. C'est pourquoi vous pouvez voir des différences entre 2 smartphones de marques différentes, mais qui ont pourtant la même version d'Android. Certaines applications sont cependant intégrées automatiquement : Google, Gmail, Google Maps, YouTube… Chaque version d'Android porte un numéro et un nom. Par exemple, en 2017, les versions généralement installées sur les smartphones récents sont les versions : 7 (Nougat) et 8 (Oreo).

iOS :Après Android, iOS est le deuxième système d'exploitation le plus répandu. Il équipe uniquement les « iPhone », c'est-à-dire les smartphones de la marque Apple. Le premier iPhone fut commercialisé en 2007. Depuis, plusieurs modèles sont sortis,

généralement accompagnés d'une nouvelle version du système d'exploitation. Par exemple, fin 2014, la sortie de l'iPhone 6 a coïncidé avec l'arrivée de l'iOS 8. La sortie des nouveaux appareils est généralement très attendue, car elle amène souvent beaucoup de nouveautés. Certains fans de la marque se ruent dans les magasins pour obtenir les nouveaux smartphones dès leur sortie.

Les ressources limitées : La majorité des appareils mobiles sont conçus pour être petits en termes de mémoire, de la taille de l'écran d'affichage et de la puissance de la batterie, etc. Cependant, les performances d'un téléphone mobile sont plus modestes, même si la tendance actuelle des constructeurs est de proposer des appareils mobiles tout-en-un de plus en plus puissants disposant des mêmes fonctionnalités qu'un PDA 5. Donc, en dépit de l'accroissement de la puissance de ces appareils (i.e. produire des batteries de plus en plus puissantes, des capacités de stockage de plus en plus grandes) et quelque soit leurs performances, leurs capacités restent limitées face à celles des ordinateurs de bureau et portables. Certaines limitations techniques inhérentes à leur taille d'écran, à la puissance des processeurs, aux interfaces de connectivité et à la capacité de mémorisation influencent le comportement des applications mobiles. Dans cette section nous discutons ces ressources limitées qui sont offertes par les appareils mobiles et qui doivent être prises en compte lors du développement d'applications mobiles (Meier, 2010). Comparés aux ordinateurs de bureau ou portables, les appareils mobiles ont : ◦ De petits écrans avec de faibles résolutions, ◦ De connexions réseaux moins fiables, ◦ Des capacités de stockage permanent limitées, ◦ Une RAM limitée, ◦ Des batteries à autonomie limitée, ◦ Une puissance processeur plus faible. Les écrans des appareils mobiles à petite taille et avec une résolution faible limitent grandement le nombre d'informations qui peuvent être affichées à l'écran. Ils nous rendent se contenter juste de l'essentiel ce qui n'est pas forcément un mal. Les nouvelles générations de téléphones améliorent ce point en proposant de nouveaux écrans avec des résolutions incroyablement meilleurs permettant d'afficher beaucoup d'informations facilement compréhensibles. Cependant, en tenant compte du grand nombre d'appareils mobiles existants et afin de garantir un meilleur affichage du contenu des applications installées sur eux ; il est recommandé de prévoir dès la conception les caractéristiques les plus favorables, i.e. la taille d'écran

recommandée, parce que l'interface de l'application mobile doit être adaptée à la taille de l'écran de l'appareil utilisé. Afin de s'assurer que l'application mobile est élégante et pourra se présenter bien sur pas mal d'appareils mobiles, il est possible de la concevoir pour de petits écrans mais également de prévoir que son interface s'adaptera à de plus grandes tailles en utilisant des techniques d'optimisation d'interfaces.

Avec les services de connectivité et plus particulièrement le service Wi-Fi, les déconnexions restent fréquentes, i.e. connectivité instable, et rien ne garantit que l'utilisateur sera toujours dans une zone couverte. De plus, la connexion sans fil utilisée par les appareils mobiles a généralement une bande passante limitée par rapport à une connexion filière. Cependant, la limitation de la bande passante, la vitesse relativement lente et la non fidélité de cette technologie par rapport à un réseau câblé représentent des obstacles et donc influencent sur l'exécution des applications sur ces appareils. Pour cela, il est préférable de prévoir ces limitations dès la conception et de supposer lors du développement d'applications pour Internet que la connexion réseau sera lente, intermittente, coûteuse et de s'assurer également que l'application développée pourra gérer la perte de connexion. Les progrès dans les mémoires flash et les disques SSD (Solid State Drive) ont entrainé un accroissement considérable des capacités de stockage des appareils mobiles, i.e. diminuer l'utilisation des disques flash ou des cartes SD et donner la possibilité de stocker même les collections mp3, photos, et jeux sur l'appareil. Les disques optiques de ces appareils proposent plus de 32 Go d'espace alors qu'en revanche le Téraoctet est maintenant courant sur les disques pour PC. Dû au fait que les applications mobiles doivent être installées dans la mémoire interne et non sur des cartes SD externes ainsi que l'essentiel de l'espace de stockage d'un mobile sera probablement utilisé pour stocker de la musique et des films, les appareils mobiles offrent à nos applications une capacité de stockage limitée. Les appareils mobiles offrent des batteries qui ne sont pas assez performantes avec une énergie limitée. La manipulation des appareils mobiles consomme cette énergie avec le temps et tout dépend de l'utilisation que nous en faisons. La consommation énergétique d'un Smartphone reflète par la quantité d'énergie utilisée par le Smartphone afin de faire fonctionner les services qu'il propose. A titre d'exemple, les jeux ou les lecteurs vidéo videront bien plus vite la batterie que la consultation des

sites web. Ce qui démontre que l'exécution d'une telle application ou d'une tâche quelconque nécessite forcement la consommation d'une proportion x % d'énergie disponible. Cela prouve que l'exécution des applications mobiles dépend fortement de l'énergie de la batterie disponible actuellement en fonction de sa capacité. En terme de la puissance du processeur, les appareils mobiles les plus performantes dont la vitesse de ces processeurs dépassent le Gigahertz. La puissance du processeur a une dépendance directe avec l'exécution d'une telle application mobile où cette dernière ne doit pas lancer un traitement de plusieurs centaines de milliers d'interactions sur un appareil ayant un processeur très lent .

I.3 Les applications mobiles

I.3.1 Définitions

Une application mobile est un logiciel applicatif développé pour être installé sur un appareil électronique mobile, tel qu'un téléphone portable, un « smartphone ».

Une application mobile est un programme téléchargeable de façon gratuite ou payante et exécutable à partir du système d'exploitation du téléphone.

Ce petit logiciel s'appuie d'une manière générale sur le principe de widgets que nous connaissons sur nos ordinateurs. Pour télécharger une application sur un téléphone mobile, il existe différentes possibilités :

♣ Transfert depuis un ordinateur via un câble de connexion,

♣ À partir d'un service mobile,

♣ via une boutique logicielle accessible depuis un téléphone mobile (App Store d'Apple, Windows Market Place, Nokia OVI, AndroidMarket, etc.)

♣ Le cas échéant l'application est dite native ; elle est déjà dans le téléphone lors de l'achat du téléphone (l'opérateur ou le fabricant l'a ajouté comme fonction de base) [60] [61].

I.3.2 Caractéristiques d'applications mobiles

Des contraintes techniques qu'il est nécessaire de prendre en compte lors de la conception d'une application mobile :

- Tailles d'écrans variables, pouvant dans certains cas être assez réduite
- Possibilité limitée de saisie de données
- Puissance du processeur, pouvant être limité sur les premiers smartphones
- Tailles de la mémoire pouvant varier
- Autonomie du smartphone
- Débits variables de la bande passante Internet. Outre ces aspects techniques, il est également important de prendre en compte l'ergonomie de l'application mobile à réaliser, et c'est un point crucial à ne pas négliger Une application mobile doit respecter certaines règles

: • Utiliser des images petites et légères

- Utiliser des éléments facilement accessibles
- Maitriser l'utilisation du javascript pour économiser la batterie
- Adapter le mode de saisie des informations. [4] Les Solutions mobiles adaptées aux exigences et contraintes des clients. Mobile repose sur cinq domaines de compétences:
- Techniques iOS, Android, Windows, Objective-C, Java, C#, XAML, …
- Architecture : Performance, fiabilité, Intégration, sécurité, Evolutivité ,
- Design & Ergonomie
- Fonctionnelles spécifiques à la mobilité
- Démarche Projet et aux conseils relatifs aux projets de mobilité.[62]

I.3.3 Domaines d'application mobile

Avec les possibilités matérielles incorporées aux terminaux (caméra, GPS, gyroscope, …), les applications Smartphones et Tablettes peuvent intégrer des fonctionnalités spécifiques et dédiées pour les utilisateurs, permettant ainsi d'enrichir le spectre fonctionnel et imaginer des usages non couvert jusqu'à présent par les systèmes d'information

- Géo-localisation, Itinéraires
- Scan de Code barre, Flash, QR Code
- Réalité augmentée
- M-commerce, Paiement mobile
- Push et notification
- Gestion de documents, dématérialisation, Workflow
- Analyse d'Audience
- Gestion et Sécurisation de parc et de déploiement de terminaux mobiles.[63]

I.4 Le contexte dans un environnement mobile

Le contexte est considéré comme un aspect primordial pour le développement d'applications issues de trois domaines de l'informatique qui jouissent à présent d'une attention importante de la part de la communauté informatique : l'informatique ubiquitaire (ou ambiante) (Weiser, 1991), l'informatique pervasive [11] et l'informatique sensible au contexte qui est connu sous le terme anglais "context-awarnes computing" [12]. [13]a déclaré que l'adaptation au contexte dans un système ubiquitaire a pour objectif de fournir aux utilisateurs des applications adaptées à la diversité des dispositifs qu'ils ont à leur disposition, tandis que l'objectif de l'adaptation au contexte dans un système pervasif est d'utiliser le plus de ressources possibles localisées autour de l'utilisateur afin de lui fournir une application adaptée sans qu'il intervienne dans le processus. Contrairement au

paradigme de l'informatique mobile dans lequel les applications peuvent découvrir et utiliser l'information contextuelle, l'adaptation au contexte sert à fournir aux utilisateurs des applications adaptées qui prennent en compte le changement du contexte d'utilisation inhérent au déplacement de l'utilisateur. Notre travail inclus dans ce cadre de recherche : l'adaptation au contexte. Plus précisément, il se focalise sur les applications mobiles déployées sur des Smartphones, tablettes, etc. qui font partie du domaine de l'informatique sensible au contexte. Dans cette section nous allons aborder la notion du contexte dans cet axe. Présentons dans un premier temps des définitions générales concernant cette notion, passons vers la notion d'adaptation. Après, et vu qu'un contexte quelconque nécessite une spécification nous présentons quelques modèles et approches de modélisation dédiés pour la description du contexte**.**

I.4.1 Définition du contexte

Dans les deux dernières décennies, avec l'évolution de l'informatique mobile, la notion du contexte est également devenue très importante dans la recherche. Dans cette section, nous nous intéressons aux définitions de deux notions précieuses contexte et context-aware qui existent dans la littérature.

Qu'est-ce qu'un contexte ?

De nombreux travaux issus de l'informatique sensible au contexte servent à donner une définition du contexte afin d'élaborer un socle pour la tâche d'adaptation. Il existe en fait à peu près autant de définitions du contexte qu'il y a d'équipes de recherche. Malgré ça, une définition à la fois générique et pragmatique de la notion de contexte reste encore manquante. Plus précisément, les chercheurs dans le cadre de recherche de l'adaptation au contexte n'ont pas encore abouti à une définition des paramètres constituant le contexte. En fait, toutes les définitions déjà présentées sont soit très abstraites ce qui rend la modélisation du contexte très difficile, soit très relatif à un domaine particulier. Les auteurs dans ont déclaré quatre familles de contexte où chacune d'entre elles apporte ses propres définitions du contexte : la notion du contexte généralisé, la notion du contexte

géolocalisé, la notion du contexte unifié et la notion du contexte environnemental. Nous nous intéressons aux définitions inclues à la catégorie du contexte environnemental dû au fait qui est celle la plus adoptée dans le domaine de l'informatique sensible au contexte. En 1995, Brown a défini le contexte comme un ensemble d'éléments de l'environnement de l'utilisateur [15] . Deux années après, les auteurs [15] dans ajoutent à la première définition des entités telles que l'heure, la saison, la température, l'identité et la localisation de l'utilisateur. En 2000, les auteurs dans [12] ont déclaré que le contexte détermine soit le comportement d'une application soit les évènements issus de l'application et qui intéressent l'utilisateur. Par conséquent, ils définissent le contexte comme étant un ensemble d'états et de configurations observées de l'environnement. Des travaux très pertinents sont fournis en 1999-2001 par Dey et Abowd [17] afin d'améliorer la compréhension des notions du contexte et context-aware tout en faisant une synthèse des travaux existants à l'époque. Grâce à ces travaux, les auteurs ont donné leur propre définition pour qu'elle soit celle qui généralise les précédentes, soit la plus complète et la plus adoptée par les chercheurs dans leurs travails (ex. nous citons comme exemple [18].Donc, ils ont défini le contexte comme suivant :

« Le contexte est toute information qui peut être utilisée pour caractériser la situation d'une telle entité. Une entité est une personne, un lieu ou un objet qui est considéré pertinent pour l'interaction entre l'utilisateur et l'application, y compris l'utilisateur et l'application. »

Contrairement aux travaux précédents [19] ; [16] qui se contentaient d'énumérer des éléments de contexte tels que (la position, l'identité des personnes et des objets voisins, les changements de voisins, la saison, la température, le temps, etc.), la définition donnée par Dey et Abowd permet de juger si une information donnée fait partie du contexte ou non. De plus, le contexte est considéré par rapport à chaque application (ou activité). Dans le travail de recherche actuel nous appuyons cette définition pour caractériser le contexte de composition de l'application mobile où nous sélections dans un premier temps les entités du contexte (l'appareil mobile cible et les entités logicielles à

composer), après nous présentons toutes les informations pertinentes qui peuvent être utilisées pour définir la situation de d'une telle entité.

Que veut dire context-aware ?

Le terme anglais "aware" signifie conscient. La notion context-aware application qui reflète la notion d'une application sensible au contexte en terme français exprime qu'une application peut être consciente de son contexte et réagir en conséquence. En 1992, un système de localisation des employés dans les pièces d'un bâtiment utilisant des badges actifs [20] était le premier travail de recherche en informatique qui intègre le context-aware. La notion context-aware computing qui veut dire en français l'informatique contextuelle est abordée pour la première fois dans le travail de [19]. Ces auteurs ont défini l'informatique contextuelle comme la capacité d'une application mobile à découvrir et à réagir à des changements dans l'environnement. La définition du contexte donnée par ces auteurs porte sur des variables bien déterminées comme la localisation, l'identité des personnes proches et des objets ainsi que les changements dans ces objets. Cette définition est apparue trop spécifique pour Dey et Abowd dû au fait qu'il est impossible pour un élément pris dans l'environnement de savoir avec certitude qu'il fait partie du contexte ou non. D'après eux, il est impossible d'énumérer de façon exhaustive toutes les variables d'environnement qui peuvent influer sur le contexte. Ces auteurs proposent en 1999 [17] qu'un système soit sensible au contexte s'il utilise des informations du contexte pour doter l'utilisateur d'informations et/ou des services pertinents ou la pertinence dépend de la tâche de l'utilisateur. De nouveau, ces définitions ne sont pas encore assez suffisantes pour inclure tous les aspects. La sensibilité au contexte peut être spécialisée en dépendance de son utilisation.

I.4.2 Vers la notion d'adaptation :

Les auteurs dans [21] proposent une définition pour la notion d'interaction du contexte comme étant l'intersection entre les connaissances contextuelles de l'ordinateur et celles de l'utilisateur dont l'idée est que l'utilisateur se fait des connaissances contextuelles du système. Tirant profit de la définition de la sensibilité au contexte présentée en [17] ,nous allons s'inspirer et rediriger la notion d'interaction du contexte

apportée par ces auteurs vers notre axe de recherche qui porte sur l'adaptation d'applications mobiles au contexte de l'appareil mobile qui les supporte, à savoir spécialiser la définition donnée pour le domaine de développement mobile pour qu'elle soit :

« Le contexte d'interaction est l'interaction entre les informations contextuelles de l'application mobile et celles de l'appareil mobile cible où l'idée que l'application doit toujours avoir des connaissances contextuelles sur son environnement d'exécution afin de mettre à disposition des informations et des services à l'utilisateur » (cf. Figure 1.1).

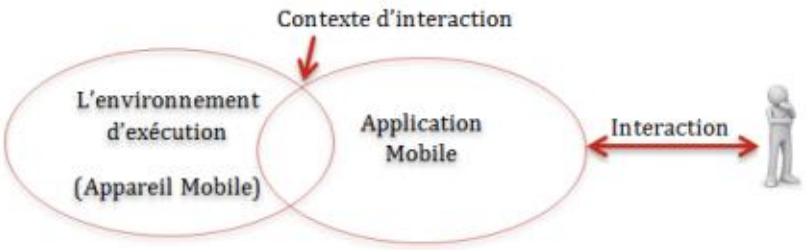

Figure 1: zone d'interaction application et appareil mobile

Par conséquence, le contexte d'interaction dans l'environnement mobile représente la conformité des contraintes d'exécution de l'application avec le contexte d'exécution de l'appareil mobile. En se basant sur les définitions du contexte et context-aware présentées dans la section précédente, nous devons distinguer entre les applications qui utilisent le contexte comme par exemple un service de météo qui aura besoin d'informations de localisation et de temps pour produire un bulletin et d'autres applications qui adaptent leur comportement en fonction du contexte. Une autre définition du context-aware plus orientée vers l'adaptation au contexte est donné par Brown en [22] Il dit qu'une application sensible au contexte doit automatiquement extraire de l'information ou effectuer des actions en fonction du contexte utilisateur détecté par les capteurs. D'un côté, une application peut juste se concentrer sur la détection, la perception et l'interprétation des informations contextuelles de l'environnement d'exécution. Ce qui démontre que l'utilisation simple du contexte n'implique pas une modification de son

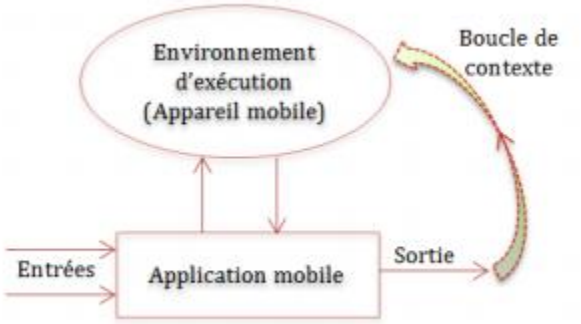

Figure 2 : l'adaptation du contenu en fonction du terminal

comportement. Par exemple, les applications dans les travaux [23] ; [24] font juste des connaissances sur les éléments de l'environnement de l'utilisateur. D'un autre côté, les applications peuvent changer dynamiquement leur comportement en fonction du contexte de l'environnement d'exécution [16] ; [25]. En conséquence, nous avons abouti à une définition beaucoup plus orientée vers notre axe de recherche pour les deux notions d'adaptation et d'auto-adaptation. Afin de pouvoir obtenir des applications adaptatives, le développeur acquiert des connaissances sur l'environnement d'exécution pour garantir si les contraintes d'exécution d'une application mobile sont conformes au contexte d'exécution de l'appareil mobile cible, i.e. l'adaptation du contenu en fonction du terminal (cf. Figure 1.2). Par conséquent, l'adaptation est le fait de s'assurer que l'application souhaitée pourra fonctionner correctement dans un tel environnement mobile.

I.4.3 Modélisation du contexte

Une application sensible au contexte doit fournir des informations et des services pertinents par rapport à la tâche de l'utilisateur en dépendance des informations contextuelles de l'environnement d'exécution de cette application. Une tâche de modélisation du contexte est donc extrêmement importante et indispensable pour atteindre cet objectif. Dans cette section, nous abordons dans un premier temps les modèles dédiés pour la description du contexte. Après, nous présentons quelques travaux qui s'inscrivent dans le domaine de la modélisation du contexte.

I.4.3.1 Modèle du contexte

Afin de pouvoir utiliser les informations contextuelles dans un système sensible au contexte, Dey nous a proposé dans [26] de suivre trois étapes nécessaires. En premier lieu, capturer ces informations. La deuxième étape sert à effectuer une interprétation du contexte pour passer à une représentation de haut niveau plus exploitable pour l'application. Finalement, fournir ces informations interprétées à l'application. Pour avoir une représentation fiable et riche des données capturées et aussi garder une trace de l'historique de ses valeurs, plusieurs chercheurs ont proposé de stocker le contexte avant sa diffusion à l'application, i.e. la modélisation de contexte. Dans cette section, nous nous intéressons aux modèles de la littérature pour la prise en compte du contexte. Les modèles du contexte ont pour objectif de stocker et échanger les valeurs des paramètres contextuels. D'après les travaux de [27] ;[12], nous avons trouvé que les structures de données utilisées pour décrire et diffuser le contexte s'organisent selon six catégories : le modèle clés-valeurs, le modèle à balise, le modèle orienté objet, le modèle basé sur la logique, l'ontologie et la logique de situation. Cependant, nous portons une attention particulière aux modèles les plus adaptés pour les environnements mobiles :

• Le modèle Clés-Valeurs : ce modèle consiste à stocker le contexte en utilisant un ensemble de couples (attribut, valeur). Par exemple : (location, "hospital"). Dans cet exemple, l'attribut de l'environnement est location. Ce dernier, représente l'information de l'emplacement qui porte la valeur hospital.

• Le modèle à balise : l'équivalent de ce terme en anglais est Markup model. Il contient des données organisées dans les structures hiérarchiques grâce aux balises. Ce modèle sert à représenter le contexte en utilisant le RDF 6 qui est développé par le W3C.

• L'ontologie : les ontologies sont considérées parmi les modèles les plus utiles pour représenter et traiter le contexte. Une ontologie permet de créer des vocabulaires partagés entre les systèmes, i.e. supporter l'interopérabilité, comme elle permet aussi d'ajouter l'aspect sémantique aux données contextuelles.

Chaque modèle est recommandé pour être utilisé pour modéliser le contexte d'un type particulier des applications. Comme par exemple les applications destinées aux

ordinateurs de bureau où des serveurs peuvent utiliser des modèles basés sur la logique afin de déduire de nouvelles informations alors que les applications dédiées aux téléphones mobiles nécessitent des modèles plus légers leur permettant de réagir avec le changement du contexte.

I.4.3.2 Approches de modélisation

Dans cette section, nous nous focalisons sur les travaux qui portent sur la phase du stockage du contexte, i.e. description du contexte. La modélisation du contexte est définie par comme une tâche de spécification de toutes les entités et les relations entre ces entités qui sont nécessaires pour décrire le contexte [29]. Plusieurs représentations contextuelles portant sur les modèles du contexte cités précédemment ont été proposées. Parmi eux nous présentons : Le contexte Toolkit de Dey est l'une des premières architectures qui prend en considération la sensibilité au contexte [26] .Suivant le principe de cette architecture, une application s'abonne au serveur pour être au courant d'un changement de contexte où les données contextuelles sont capturées par des senseurs et interprétées via des widgets avant de les transférer au serveur. Le contexte Toolkit de Dey est basé sur le modèle clef-attribut. La représentation contextuelle la plus connue qui repose sur le modèle à balise c'est CC/PP 7. Elle est proposée par le W3C [30] et basée sur le langage général de description des méta-données RDF et encodée en XML. La spécification CC/PP permet de construire un profil, tenant compte des capacités du terminal mais aussi de l'utilisateur. Les auteurs dans [et [29], [30] ont utilisé ce modèle pour la prise en compte du contexte. Mukhtar et al. Dans leurs travaux effectués respectivement en 2008 et 2009 [40] étendent le modèle CC/PP afin de l'adopter pour effectuer la tâche de sélection des composants constituants d'une application multimédia dynamiquement tout en prenant compte leurs ressources nécessaires et/ou les préférences de l'utilisateur. En 2012, Derdour a proposé un diagramme de classe pour la description du contexte des applications multimédias [32]. Ce contexte est représenté par un profil utilisateur, réseau et plateforme où cette dernière est composée d'un profil matériel, logiciel et environnement. Cette description est basée sur le profil CC/PP pour décrire les caractéristiques physiques de l'utilisateur et les objectifs de transfert des flux multimédias. De plus, il existe de nombreux travaux décrits

dans la littérature qui attaquent l'aspect de modélisation du contexte où chacun fournit une description contextuelle dédiée pour traiter une problématique spécifique dans un système particulier. Dû au fait qu'une personne n'aura pas besoins à tout moment de tous les services auxquels elle pourra accéder, les auteurs dans [30] ont proposé un modèle formel de contexte pour choisir les services les plus pertinents. Ce modèle est basé sur les préférences de l'utilisateur et les descriptions de services pour faire une recherche pro-active de services appropriés, i.e. découverte personnalisée de services. Toutes les approches de modélisations indiquées en dessus touchent le domaine de l'informatique mobile. Par conséquent, elles sont dédiées aux systèmes sensibles au contexte. Plusieurs d'autres descriptions contextuelles ont été proposées pour la spécification du contexte dans les systèmes pervasives et ubiquitaires. Dans le travail [31], les auteurs font une évaluation des modèles du contexte. Ils concluent que les ontologies sont les plus adaptées pour modéliser le contexte dans des environnements ubiquitaires. Le modèle d'ontologie CoOL (Context Ontology Language) proposé dans [32] représente le contexte comme un ensemble d'entités ayant des aspects décrivant leurs caractéristiques. L'auteur dans [33] a proposé une ontologie de contexte commune comme une solution pour l'hétérogénéité des environnements. De plus, nous trouvons le modèle d'ontologie CONON (CONtexte ONtologies) [34] qui est destiné pour modéliser le contexte dans les systèmes pervasives et l'ontologie de contexte CoDAMos (Context-Driven Adaptation of Mobile Services) qui est proposée en 2005 dans [35] et dédiée aux systèmes ambiants intelligents. En se basant sur les principes de ces différents travaux de modélisation du contexte, nous pensons que le modèle d'ontologies est le plus adéquat même pour définir le contexte de développement des applications mobiles. Dû au fait que ce type d'applications ainsi que leurs environnements d'exécution présentent plusieurs points d'hétérogénéité, les ontologies permettent de fournir un vocabulaire commun partagé entre eux. Donc, la modélisation du contexte en utilisant ce type de modèles du contexte est considérée comme une solution pour l'hétérogénéité présentée par les appareils mobiles et les applications mobiles.

I.5 Conclusion

Dans ce chapitre nous avons passé en revue quelques informations théoriques sur les notions clés liés à notre travail de recherche incluant des définitions présentées dans la littérature pour chacune des applications mobiles, appareils mobiles et contexte. Nous avons présenté quelques définitions dédiées aux applications mobiles. Ainsi. Nous avons constaté que contrairement aux dispositifs classiques, i.e. les ordinateurs, les différents appareils mobiles sont caractérisés par des ressources limitées et présentent plusieurs points de différences que ce soit du point de vue matériel ou logiciel. Cependant, nous avons montré aussi que ces caractéristiques et ressources limitées forment les informations contextuelles de l'environnement d'exécution des applications mobiles et que le contexte de l'environnement d'exécution change souvent. Face à ce constat, nous avons essayé de comprendre ce qui est le contexte pour les applications mobiles et son intérêt pour garantir un meilleur fonctionnement et finalement en présentant quelques approches de modélisation du contexte. Dans le chapitre qui suit, nous aborderons toutes les notions et les différentes techniques ainsi que les travaux existants qui sont liés à nos contributions et qui reflètent un socle théorique pour le processus de sélection proposé.

Chapitre 2 : méthode de sélection des entités logicielles

II.1 Introduction

La hausse du nombre d'applications où des services disponibles sur les différentes plateformes des dispositifs mobiles est indéniable. A la portée l'utilisateur, Ce dernier a plusieurs exigences qui seront répondu avec des différentes applications.

. Un besoin naissant est alors de pouvoir combiner différentes applications - entités logicielles - afin de pouvoir tirer la meilleur partie de leurs fonctionnalités pour obtenir une application composite comblant les besoins de l'utilisateur.

Dû au fait que les entités logicielles existantes ainsi que ses plateformes d'exécution présentent différents points d'hétérogénéité, Notre approche aborde le choix de l'adaptation entités logicielles afin de construire un composite spécifique d'application pour l'environnement mobile.

Nous visons tout d'abord à modéliser le contexte de la composition que ce dernier reflète la spécification des informations contextuelles de l'entité logicielles constitutives (c'est-à-dire entités pouvant être utilisées pour mettre en œuvre les différents fonctionnalités), y compris son rôle et toutes les conditions pour son exécution. Par conséquent, toutes les caractéristiques associées au dispositif mobile dans lequel les applications souhaitées seront déployées, ainsi que l'état d'exécution.

Nous avons choisi comme langage de description de modéliser le contexte de composition « ontologie » [36] Ceci fournit plus tard une description sémantique qui permet de déterminer le comportement de toute entité logicielle.

En outre, étant donné qu'une entité logicielle peut être implémentée soit avec un service, soit avec un composant, etc. l'ontologie fournit une description standard de ces entités hétérogènes. Donc, il prend en charge l'interopérabilité des entités logicielles hétérogènes comme un vocabulaire commun.

Après, et à travers ces descriptions de contexte, nous proposons un algorithme de sélection qui vise à sélectionner un logiciel contextuel entités.

Il s'agit de composer une application mobile adaptative pour un environnement spécifique, et aussi de fournir tous les différents chemins de composition valides possibles. Les ontologies sont destinées à représenter les informations contextuelles des entités logicielles, et le périphérique mobile cible étant interprété à la fois par le développeur de l'application mobile et notre mécanisme de sélection proposée. L'extraction des informations nécessaires sera à travers les descriptions XML fournies par ces ontologies.

II.2 Processus pour la Composition d'Applications Mobiles (CAMAP)

Tout d'abord nous donnons une définition générale de la composition d'applications présentée par Christian Brel dans son travail de recherche [37] : « La composition d'applications est le moyen de combiner plusieurs morceaux d'applications existantes pour en construire une nouvelle. Cette composition réutilise tout ou une partie des applications existantes. ».

une application peut être représentée par des services ou des composants, plusieurs définitions plus précises et spécialisées pour des domaines particuliers ont été proposées pour la notion de composition. Plusieurs définitions ont été proposées pour la composition qui porte sur les services web.

Les auteurs dans [38] définissent la composition comme suivant « La composition est le processus de la sélection, la combinaison et l'exécution des services web pour atteindre l'objectif de l'utilisateur. ».

Les auteurs dans [39] [40] ont dit que « Si l'objectif du concepteur d'une application n'est pas atteint par l'invocation et l'exécution d'un simple service web élémentaire, alors le concepteur doit combiner les fonctionnalités d'un ensemble de services.

Dans ce cas-là cette tâche est appelée composition de services web ». Les auteurs considèrent la composition de services web comme étant un moyen efficace pour créer,

exécuter et maintenir des services qui dépendent d'autres services où les services web invoqués lors de la composition sont appelés services web composants.

D'autres travaux ont mis l'objet de composition des entités logicielles du type service. Les auteurs dans [40] ont défini le processus de composition de services comme la spécification d'un service composite à partir d'autres services. Une autre définition plus significative présentée par [41] est la suivante : « La composition de services désigne une interaction entre deux ou plusieurs services en vue d'accomplir des objectifs déterminés. ». D'après ces travaux [41];[42];[43] et [44]), la composition de services nécessite au préalable d'établir le lien d'exploitation avec le fournisseur de services et après combiner les différents services réutilisés qui sont identifiés sous la responsabilité du consommateur de services afin de réaliser l'application. Par conséquent, ce mécanisme de composition vise dans un premier temps à découvrir les services disponibles qui peuvent répondre aux besoins du consommateur ainsi qu'à sélectionner ceux les plus adaptés aux préférences ; après à définir et à gérer les collaborations entre les services effectivement utilisés. En effet, la composition d'applications est un processus complexe qui fait intervenir un ensemble d'étapes intermédiaires. Ces étapes se répartissent suivant les préoccupations classiques liées à la réutilisation d'entités logicielles existantes commençant par l'identification des entités qui seront composées puis la réalisation effective de cette composition. Par conséquent, le processus de composition d'applications peut être divisé en trois étapes : (1) la découverte, (2) la sélection et (3) la composition d'entités logicielles.

Ce processus vise à effectuer la tâche de composition en commençant par l'identification de l'ensemble des fonctionnalités souhaitées tout en ignorant comment elles seront implémentées ainsi que les différentes dépendances entre ces fonctionnalités. Ces dépendances décrivent les ordres d'invocation ainsi que les données échangées entre elles. Après, il associe chaque fonctionnalité identifiée aux entités logicielles concrètes correspondantes qui peuvent être utilisées pour l'implémenter et qui peuvent se trouver dans différents emplacements (ex. internet, emplacement locaux, d'autres appareils, etc.). L'étape suivante permet de sélectionner parmi les entités correspondantes trouvées les plus appropriées qui sont adaptables au contexte courant de l'appareil mobile. La tâche de

sélection est basée sur les conditions d'exécution des entités logicielles et la description contextuelle de l'appareil mobile cible. Ensuite, il vise à composer les entités contextuelles sélectionnées tout en incluant les adaptateurs nécessaires dans le cas d'une coordination hétérogène. La dernière étape consiste à générer l'application mobile composite concrète vers une plateforme spécifique sur laquelle elle sera exécutée. Cependant, nous devons mettre l'accent sur l'acronyme CMA représenter le terme Composite Mobile Application. Nous allons maintenant aborder une définition spécifique pour chacune de ces processus: la découverte, la sélection et la composition ainsi que quelques travaux apportés pour chacune d'entre elles.

II.2.1 Processus de découverte :

La découverte est un point clé nécessaire pour le développement d'applications par la réutilisation de l'existant. Elle désigne le processus qui permet d'identifier les entités logicielles qui peuvent répondre aux besoins exprimés par l'architecte. Ce processus de découverte repose, selon le cas, soit sur une recherche dans un entrepôt ou un registre local ou public, i.e. chercher des composants ou des services, soit sur une découverte globale sur le web, i.e. chercher des services web. La tâche de découverte sert à comparer les descriptions d'entités logicielles dont il a accès avec les exigences fonctionnelles et non fonctionnelles qui lui sont fournies afin de déterminer les entités candidates. Ces dernières sont les entités existantes d'après leurs descriptions peuvent répondre aux exigences. Afin d'avoir cette liste des candidates, le processus de découverte doit passer par deux étapes essentielles :

(1) L'accès aux entités logicielles disponibles [45]; [46].

(2) La description de ces entités et les besoins puis la détermination d'une solution comme adéquate [47] ; [48].

Donc, la découverte est permise par la description préalable des entités et leur publication au sein d'un registre ou d'autres emplacements. Dans le cas où les écarts avec les exigences sont toujours expressifs, les solutions les plus proches possibles sont proposées puis laissées aux choix de l'architecte de les utiliser ou non. Cependant, nous

abordons une définition plus orientée vers notre axe de recherche et convenable aux applications mobiles pour la notion de découverte : « Le processus de découverte vise à chercher et choisir les entités logicielles correspondantes pour chaque fonctionnalité désirée en les téléchargeant via internet où certaines d'entre elles sont gratuites tandis que d'autres sont payantes. Ce processus exploite l'ensemble des fonctionnalités identifiées qui représente l'application mobile désirée afin de trouver les différentes entités logicielles qui satisfont les exigences des utilisateurs décrites par ces fonctionnalités. » Plusieurs travaux ont été effectués sur cet axe de recherche. Les auteurs dans [49] ont présenté une approche pour la découverte automatique des services dans les environnements d'exécution distribuée (WSMX : Web Service Modelling eXecution environment). Ce processus de découverte vise à comparer la description de la requête avec la description des services stockés dans un répertoire localisé dans le pair ; ce répertoire est dit le registre local des services. La plateforme WSMX proposée dans ce travail est constituée de deux composants principaux communication discovery sub-component et local discoverysub-component. Le premier composant gère la communication avec les autres contrairement au deuxième composant qui est dédié à la découverte local support dirigé par les mots clés. Cette approche supporte la sémantique de similarité entre la requête et le service mais il ne prend pas en considération le processus composition, dans le cas où la requête n'est pas servie par un seul service. Merla dans [50] propose une plateforme pour la découverte sémantique des services web. Cette plateforme effectue la tâche de découverte avec la prise en compte des pré-conditions comme les informations du contexte des services. Cette tâche supporte la priorité inter-contexte entre les services. La plateforme proposée est alors capable de sélectionner le service web qui répond le mieux au contexte indiqué par l'utilisateur sur un appareil mobile à partir de la liste des fonctions des services Web similaires. D'autre part, la déduction de la sémantique de similarité entre les paramètres de la requête et les services fournis est basée sur une ontologie générique d'un domaine particulier spécifié en OWL-S. Les auteurs dans [51] proposent une approche dynamique pour la découverte des services web dans les systèmes pair à pair structurés. Les différentes approches d'alignement existantes entre les besoins et les entités logicielles qui peuvent répondre à ces besoins peuvent se différencier en termes de deux critères

importants : le degré de pertinence de la solution et le type de correspondance. Les algorithmes de découverte peuvent varier selon leurs capacités à gérer la pertinence des candidats déterminés par rapport aux exigences fonctionnelles et non fonctionnelles définies. Afin d'améliorer significativement cette pertinence, la formulation des besoins et des descriptions d'entités logicielles nécessite la prise en compte des sémantiques par l'emploi d'ontologies de domaines d'applications[52] ; [53] En outre, ces algorithmes de découverte peuvent conduire à l'identification d'une unique entité logicielle existante pour répondre à un besoin spécifique, i.e. l'approche 1-1 [54]; [45]. Ils peuvent aussi identifier une composition de plusieurs entités pour supporter ce besoin, i.e. l'approche 1-N [46]; [55] Quelques approches de découverte sont conçues pour être capable de construire des solutions qui n'existent pas directement dans le système en composant jusqu'à N entités logicielles disponibles. Ainsi, ces approches supportent la spécification des bonnes collaborations entre ces N entités en étroite liaison avec les exigences du consommateur.

II.2.2 Processus de sélection

Dans notre travail, nous nous focalisons en particulier qu'à l'étape de sélection. Pour la composition, le résultat de la tâche de découverte est un ensemble d'entités logicielles correspondantes pour chaque besoin défini ou fonctionnalité désirée. Les entités trouvées mappées à la même fonctionnalité sont équivalentes fonctionnellement mais elles peuvent varier en plusieurs aspects non fonctionnels. La seconde étape de la composition correspond au processus de sélection où ce dernier sert à identifier parmi les entités candidates celles les plus appropriées aux besoins de l'application à construire. En général, ces entités candidates répondent tous aux conditions minimales d'acceptation posées par l'architecte de l'application. La sélection se base donc sur les préférences des utilisateurs ou d'autres critères/contraintes pour identifier celle avec la plus haute qualité et/ou la plus adaptée au contexte d'utilisation. Ces contraintes de sélection diffèrent selon le domaine d'application. Plusieurs travaux ont été effectués à ce stade de recherche. Maintenant, nous présentons parmi eux quelques travaux proposés précisément pour les environnements mobiles. Dans [56] les auteurs ont proposé une approche transversale du choix du service. Ils se concentré sur la manière dont le client et le serveur sont appariés,

en tenant compte des performances du réseau aspect. Cette approche permet au client de changer les meilleurs serveurs la topologie du réseau change (c'est-à-dire qu'il regroupe le client avec serveurs), basé sur le mécanisme de routage ad-hoc du réseau. Ils également montré que la sélection efficace peut améliorer la le débit du réseau a augmenté de 40%. Alors que, les auteurs dans [7] montré que la sélection du fournisseur de services présentant le plus faible compte (c'est-à-dire choisir les fournisseurs de services les plus proches) n'est pas suffisante pour obtenir une meilleure performance du système. A ce niveau, ils ont proposé une sélection intégrée qui prend plus de évaluation complète de chaque fournisseur de services, y compris la capacité du fournisseur de service, le risque de défaillance du service dans le réseau, temps de réponse, etc. Auteurs dans, proposé le protocole de découverte, qui adopte deux métriques pour sélectionner un instance de service. Ces métriques sont: le nombre de sauts entre demandeur de service et fournisseur de service, et la capacité de service (c'est-à-dire que la métrique de la CoS exprime la capacité nominale d'un instance de service). Les auteurs dans [9] montré que les résultats obtenus la localisation des communications est un facteur qui conduit à améliorer les performances du réseau. Ils ont démontré à travers simulations qui déclenchent la re-sélection des serveurs, après détecter les modifications de la topologie du réseau, est très fiable réduire la congestion et les retards. Les approches mentionnés ci-dessus considèrent l'impact de la sélection du service mécanisme côté client sans prendre en compte les fonctionnalités de la cible de l'appareil. Pour traiter ce problème, plusieurs les travaux ont été développés. En raison du fait que la publicité mobile Les périphériques réseau ad hoc se caractérisent par une très grande contrainte, et aussi la situation des ressources peut changer rapidement; Les auteurs de [8] discutent de la sélection du service pour le service composite. Ils ont effectué la tâche de sélection en fonction des informations dépendantes du service et de l'appareil informations dépendantes (niveau de la batterie, charge actuelle, réseau puissance du signal, etc.). Pendant ce temps, dans[3], un logiciel infrastructure appelée AppSpotter, qui permet la dynamique et composition automatisée des composants logiciels de Une application mobile est proposée. Dans cette AppSpotter, un composant nommé "composant selector" récupère de la Actifs SPL (composant logiciel stocké) le logiciel composants qui répondent ou remplissent un ensemble de critères, qui sont

représenté avec les fonctionnalités de l'appareil mobile: plate-forme, écran, entrée, clavier et accéléromètre. Tâche de sélection dans les deux [8][3] est piloté avec des fonctionnalités d'appareil mobile mais des critères de sélection ne sont pas les mêmes. Le tableau suivant (cf. tableau 1) donne une comparaison des œuvres citées selon un ensemble de Critères. Ces critères sont divisés en deux catégories. expliqué précédemment. Le premier représente la cible de l'appareil caractéristiques, tandis que le second comprend un ensemble de caractéristiques qui représentent l'impact du mécanisme de sélection sur le client côté.

	Execution Constraints					Quality Characteristics			
	Battery Level	Memory size	Screen size	platform	Wireless connectors	Network performance	Response time	Hop-count	CoS
[6]	-	-	-	-	-	*	-	-	-
[7]	-	-	-	-	-	*	*	*	*
[10]	-	-	-	-	-	-	-	*	*
[9]	-	-	-	-	-	*	-	-	-
[8]	*	-	-	-	-	*	-	-	-
[3]	-	-	*	*	-	-	-	-	-

Tableau 1 syntheses de selection

La plupart de ces approches de sélection se concentrent uniquement sur la qualité caractéristique. Donc, aucun d'entre eux ne traite de critères importants tels que: la disponibilité des connecteurs sans fil nécessaires ou la taille de la mémoire pendant la tâche de sélection. Si une entité logicielle besoin d'un GPS ou d'un périphérique Wi-Fi pour effectuer sa tâche, nécessaire de s'assurer d'abord que le périphérique cible possède cette connecteur sans fil, sinon cela ne fonctionne pas. Aussi, la taille de l'entité logicielle

ne doit pas dépasser la capacité de rappel stockage du périphérique cible afin d'assurer son correct déploiement. Notre travail est une solution à combler ce manque. Il vise à prendre en compte tous les contraintes d'exécution pour effectuer le processus de sélection.

II.2.3 Processus de composition

Après l'identification des besoins souhaités. Nous abordons dans un premier temps un scénario de composition. Un exemple de composition est : une combinaison de deux services web nommés GeoIPService et GlobalWeather dont l'objectif est de retourner à partir de la localisation les conditions météorologiques courantes. Le résultat de cette composition est un service composite de météorologie nommé MyMeteo. Ce composite agit comme suivant : le service GeoIPService obtient l'emplacement courant de l'utilisateur, après le service GlobalWeather fournit l'utilisateur par les conditions météorologiques courantes en se basant sur les cordonnées obtenues par le premier service. Par conséquent, l'étape de composition correspond à l'établissement de la collaboration entre les entités logicielles qui ont été sélectionnées. Le résultat final est la construction de la composition d'entités encapsulées sous la notion d'une entité composite [47] Il s'agit des compositions de services à travers les orchestrations de services pour avoir un service composite, des compositions dans les approches à base de composants à travers les assemblages de composants afin d'arriver finalement à un composant composite ou également la composition des entités hétérogènes .

II.3 Modélisation de contexte basée sur une ontologie

II.3.1 La nécessité de décrire le contexte

Le développement d'applications pour des environnements mobiles est soumis à plusieurs conditions d'exécution. Dans ce contexte, les applications mobiles sont restreintes par les ressources limitées offertes par les appareils mobiles sur lesquels elles

seront déployées comme : une mémoire limitée, un environnement mobile alimenté par batterie a une capacité limitée et de la disponibilité de certains services (ex. un accès Wi-Fi, un service GPS, un appareil photo avec une fonction d'autofocus, etc.) [57]. Les ressources limitées et l'hétérogénéité des appareils mobiles montrent que la portabilité des exigences joue un rôle important dans le domaine de développement des applications mobiles. Autrement dit, les applications mobiles dépendent fortement des caractéristiques de l'environnement d'exécution. De ce fait, pour assurer le déploiement correct et le bon fonctionnement de l'application mobile composite, il est nécessaire de faire en sorte que leurs entités constitutives soient adaptables au contexte actuel de l'appareil mobile cible. Cependant, une telle fonctionnalité souhaitée définie à un niveau conceptuel peut être implémentée avec plusieurs entités logicielles équivalentes mais capable de fonctionner dans des conditions différentes. Pour cela, afin de composer une application mobile adaptative nous devons toujours prendre précisément en considération les différentes caractéristiques des appareils mobiles lors du choix des entités logicielles. De ce fait, pour l'adaptation des applications mobiles au contexte il est important de la prendre en considération au niveau conceptuel, i.e. le contexte d'exécution doit être fourni lors de la conception de l'application mobile. Donc, l'adaptation contextuelle d'une application repose sur le fait d'observer le contexte de composition. Par conséquent, le processus proposé a la possibilité d'observer l'environnement de composition, i.e. d'identifier et de récupérer les informations pertinentes sur les objets de composition et l'environnement d'exécution. L'originalité de notre processus repose entre autres sur le fait que nous proposons un moyen compatible à nos objectifs pour décrire le contexte de composition. Cependant, nous avons proposé une définition du contexte cohérente par rapport à nos objectifs et une formalisation de la définition du contexte de composition. Ce dernier reflète :

◦ La spécification des informations contextuelles des entités logicielles constituantes de l'application, i.e. les entités logicielles correspondantes qui peuvent être utilisées pour implémenter les différentes fonctionnalités requises, y compris leurs rôles et toutes les conditions nécessaires pour leur exécution.

◦ Toutes les caractéristiques associées à l'appareil mobile sur lequel l'application mobile souhaitée sera déployée ainsi que l'état d'exécution des ressources disponibles.

Nous avons choisi d'utiliser les ontologies comme langage de description afin de modéliser le contexte de composition. Ceci fournit une description sémantique qui permet de déterminer le comportement des entités logicielles et les caractéristiques de l'appareil mobile.

En outre, dû au fait qu'une entité logicielle peut être implémentée soit avec un service soit un composant, etc. et les appareils mobiles présentent des configurations hétérogènes. L'ontologie fournit une description standard de ces entités et ces appareils mobiles hétérogènes. Par conséquent, elle prend en charge l'interopérabilité entre eux comme un vocabulaire commun. Les sous-sections qui suivent ont pour objectif de modéliser le contexte de composition d'une application mobile Modélisation du contexte de l'appareil mobile cible Dans le domaine des applications mobiles

II.3.2 Modélisation du contexte de l'appareil mobile cible

Concernant le logiciel mobile, une composition réussie dépend fortement de l'environnement de l'application. Ainsi, pour assurer le bon déploiement et le bon fonctionnement des applications mobiles composites, il doit prendre en compte les informations de contexte différentes de l'appareil mobile dans lequel sera déployé lors de l'exécution de la tâche de composition.

De ce fait, il est nécessaire d'identifier les différentes caractéristiques de l'environnement de déploiement. Pour cela, nous proposons une description basée sur les ontologies pour modéliser le contexte actuel de l'environnement de déploiement comme illustré à la Fig.1

. En fait, nous pouvons trouver différents appareils avec différentes capacités (c.-à-d.

Caractéristiques hétérogènes). En outre, les fonctionnalités de l'appareil mobile sont temporaires et peuvent être changés à tout moment (par exemple niveau de la batterie

de l'appareil de l'utilisateur devient faible; Wi-Fi est désactivé, etc.). Par conséquent, le contexte courant du dispositif mobile est représenté avec ses caractéristiques matérielles et logicielles ainsi que l'état actuel des ressources disponibles, nous proposons de modéliser les informations contextuelles d'un appareil mobile selon trois catégories :

• contexte de déploiement(A1): représente le matériel caractéristiques de l'appareil mobile.

• Plateforme d'exécution(A2): représente le type de la plateforme installé dans l'appareil mobile.

• Contexte d'exécution(A3) : représente l'état actuel des ressources disponibles sur l'appareil mobile de l'utilisateur.

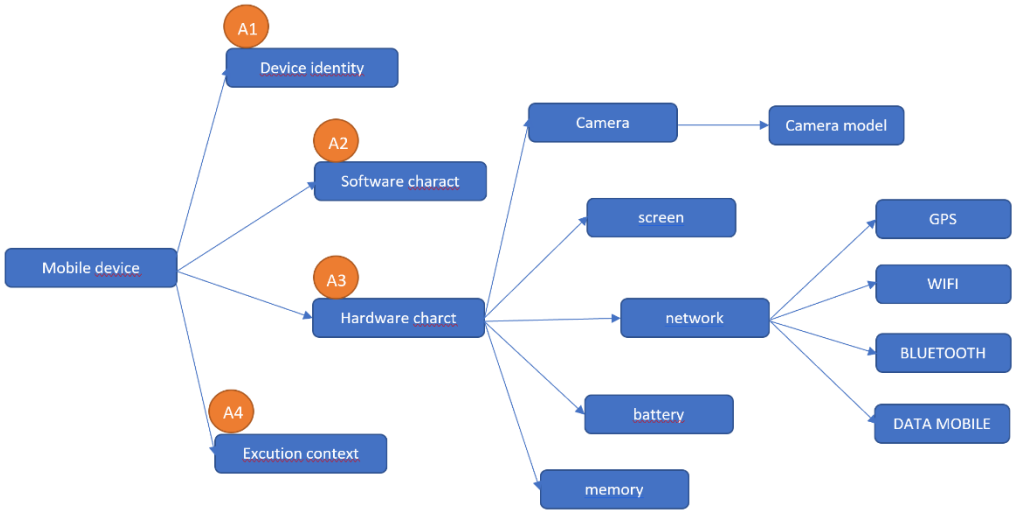

Figure 3 : L'ontologie du contexte de l'appareil mobile

Nous avons associé chaque appareil mobile par une identité ((A) : MobileDevice -Identity) qui reflète le nom du dispositif. La disponibilité de certains équipements (ex. une connexion Wi-Fi, GPS et d'une caméra) et également les valeurs de certaines fonctions de l'appareil (ex. la taille de l'écran, le type de la plateforme installée, la capacité de stockage restante, etc.) sont les critères sur lesquels nous somme basés pour assurer le bon déploiement de l'application mobile désirée (A1) : SoftwareCharacteritics, (A2) :

HardwareCharacteristics). Ainsi, la modélisation de l'état actuel des ressources disponibles (i.e. l'état de Wi-Fi, l'état de GPS et le niveau actuel de la batterie) est un aspect crucial pour atteindre notre objectif qui est d'assurer le bon fonctionnement de l'application mobile composite désirée ((A3) : ExecutionContext). Supposons que nous ayons une entité logicielle (constituante d'une telle application mobile) qui a besoin d'un accès Wi-Fi pour qu'elle puisse effectuer sa tâche et nous supposons qu'elle consomme 3% d'énergie pendant sa durée d'exécution moyenne, i.e. l'exécution de cette entité logicielle implique que la batterie de l'appareil mobile perdra 3% de sa capacité. De plus, nous supposons que l'appareil mobile cible possède le service Wi-Fi. Néanmoins, son état actuel est désactivé et le niveau actuel de la batterie est de 2%. Dans ce cas, cette entité logicielle ne fonctionnera pas correctement tant que l'accès Wi-Fi est désactivé et le niveau de la batterie est faible. Un autre exemple, si on trouve une entité logicielle constituante qui a besoin de 120Mb de mémoire et la capacité de stockage disponible actuellement sur l'appareil mobile est plus faible que celle qui est nécessaire, cette entité logicielle ne sera pas déployée avec succès.

II.3.3 Modélisation du contexte des objets de composition

Plusieurs entités logicielles peuvent être utilisées pour mettre en œuvre la même fonctionnalité souhaitée. Par conséquent, ces entités logicielles sont fonctionnellement équivalentes ((B) : FunctionalAspects) mais elles peuvent varier dans plusieurs aspects non fonctionnels ((C) : NonFunctionalAspects), i.e. des conditions d'exécution différentes. Les caractéristiques fondamentales liées à l'entité logicielle sont représentées par : son type d'implémentation (service, composant, etc.) et son rôle (description de la tâche à effectuer). Notre processus de composition repose sur l'utilisation des entités logicielles qui sont adaptables au contexte courant de l'appareil mobile cible pour composer et obtenir une application adaptative. Afin de choisir les entités logicielles contextuelles nous proposons d'associer chaque entité avec un profil d'exécution spécifique. Ce profil d'exécution contient tous les aspects non-fonctionnels qui représentent les conditions nécessaires pour son exécution (ex. la plateforme nécessaire (C1), les connectivités réseaux requis avec leurs caractéristiques (C2), les caractéristiques

requises pour un appareil photo (C3), consommation d'énergie (C4), la taille de l'écran nécessaire (C5), la capacité nécessaire pour la déployer (C6), etc.). Dans ce Ainsi, les profils d'exécution contiennent diverses mesures associées avec des valeurs. Ces métriques sont les critères que nous proposons de assurer le bon déploiement et le bon fonctionnement du entité logicielle.

Ainsi, la description de l'entité logicielle concerne non seulement caractéristiques fonctionnelles, mais doit aussi être assez riche pour fournir une visibilité sur la possibilité d'effectuer une meilleure sélection tâche. Le modèle d'ontologie présenté à la Fig.2 décrit la description fonctionnelle et toutes les conditions nécessaires à l'exécution d'une entité logicielle.

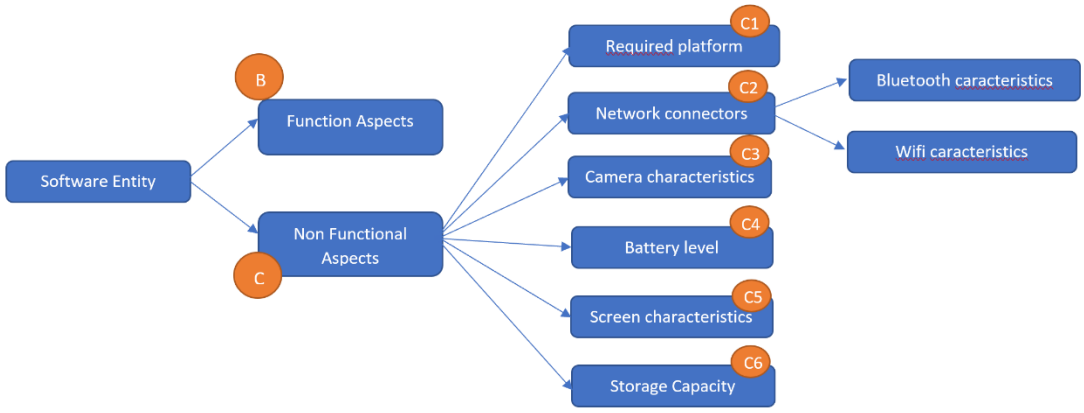

Figure 4: L'ontologie du contexte de l'entité logicielle

A titre d'exemple, nous supposons que nous avons une fonctionnalité requise Fun2 : Read Barcode qui vise à lire le code-barres d'un produit. Cette fonctionnalité peut être réalisée avec trois entités logicielles différentes (cf. Figure 3.7) :

◦ SE1 : Service (reconnaissance du code-barres à distance).

◦ SE2 : Composant (reconnaissance locale du code-barres).

◦ SE3 : Service (http ://searchupc.com/).

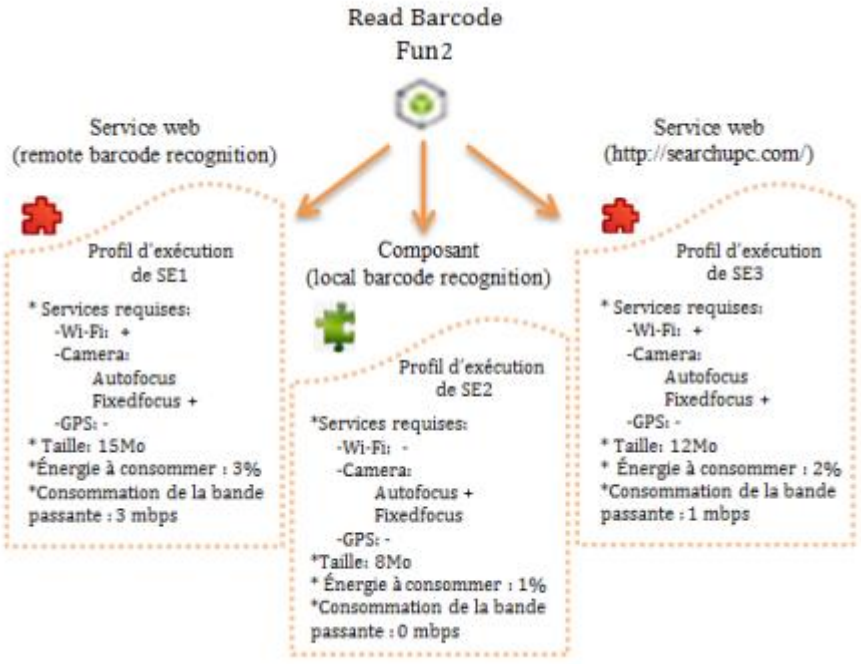

Figure 5: Profil d'exécution des entités logicielles

Chacune de ces entités logicielles doit avoir un ensemble de contraintes d'exécution, i.e. liste des conditions requises pour exécuter une entité logicielle. Par exemple, l'entité logicielle SE1 a besoin d'une caméra en mode fixedfocus, un accès Wi-Fi et 3% d'énergie à consommer pour qu'elle puisse effectuer sa tâche et donc fonctionner correctement. L'image du produit sera prise par un appareil équipé d'une caméra avec la fonction fixedfocus ce qui empêchera la reconnaissance locale du code-barres ce qui nécessite l'invocation d'un service à distance pour traiter l'image acquise. En outre, il a besoin de 15Mo d'espace mémoire pour peut être déployée correctement sur l'appareil mobile, etc. En revanche, l'entité logicielle SE2 a besoin d'une caméra avec une fonction d'autofocus et 1% de l'énergie de la batterie pour pouvoir fonctionner correctement. En outre, il a besoin de 8Mo de la capacité du stockage pour être déployée correctement sur l'appareil mobile, etc. Ici, nous ne présentons pas les valeurs réelles mais seulement les différences relatives entre les aspects non fonctionnels. Toutes les conditions nécessaires sont exprimées dans un profil d'exécution comme illustré dans la Figure 3 où les métriques

qui portent le signe + ou – reflètent le besoin ou le non besoin du service/ressource pour l'exécution de l'entité et les autres métriques portent des valeurs changeables.

II.4 Algorithme de sélection basé sur le contexte

Notre algorithme proposé est destiné à effectuer le sélection entités logicielles contextuelle après avoir recu la liste des entités logicielles qui correspondent au besoin de l'utilisateur décrit exigences (voir pseudo-code). Le mécanisme de sélection est basé a la fois sur les informations dépendantes de l'entité logicielle et sur les informations dépendantes du dispositif mobile. Les caractéristiques de dispositifs mobile et leur état actuel peuvent être déduits par le système mobile ou obtenu par le développeur. Il sera enregistré via un fichier XML conformément a l'ontologie de contexte de dispositif propose. Par conséquent, un fichier auteur conforme à l'ontologie de contexte SE est nécessaire pour décrire toutes les caractéristiques fonctionnelles non fonctionnelles des entités logicielles qui corresponds les fonctionnalités souhaitées. Le mécanisme de sélection vise à extraire des chemins de la liste SE (c.-à-d. un ensemble d'entités logiciels appropriés) de chaque fonctionnalité appartenant au chemin de la liste Fun (ie ensemble de fonctionnalités identifiées) celles qui répondent à un ensemble de critères (cf. Fig. 4). Ainsi, à partir de l'ensemble de SE fournie, seules les entités logicielles compatibles avec le contexte du dispositif (c.-à-d. un SE contextuelle) seront sélectionnées pour intégrer et assembler l'application mobile souhaitée. Nous associons cette capacité à la méthode AdaptabilityTest qui vise à comparer le profil d'exécution SE avec les fonctionnalités de l'appareil mobile afin de sélectionner le SE qui sont conformes au dispositif mobile. Elle renvoie la valeur vraie si toutes les contraintes d'exécution SE sont satisfaites. Cette méthode reflète un ensemble de règles d'influence qui visent à extraire les informations contextuelles nécessaires a partir des modèles ontologiques proposés, afin d'effectuer la tâche de comparaison. La liste suivante représente un exemple de règle d'inférence permettent de vérifier la compatibilité de SE (y) qui implémente la fonctionnalité Fun (x) avec l'appareil mobile pour la fonction caméra.

PREFIX MD-Context : URI du modèle d'ontologie de périphérique mobile
PREFIX SE-Context : URI du modèle d'ontologie d'entité logicielle

```
SELECT ? Type? Disponibilité? RequiredType
WHERE {? X SE-Context : NameFun “x” --le nom de l'amusement
? x SE-Context : ID-SE “y” - l'identifiant de la SE
? x MD-Context : CameraType? Type
? x MD-Context : CameraAvailability? Availability
? x SE-Context : RequiredCam? RequiredCam
? x SE-Context : RequiredCameraType? RequiredType}
Si
EXACT : availability = RequiredCam et Type = RequiredType
Else DISJOINT
```
Listing 1. Règle d'influence

Si nous supposons que nous avons un appareil mobile dont Le contexte actuel est le suivant: Has-GPS est désactivé, Has-Wi-Fi est activer .a une Caméra avec un focale fixe, le niveau de batterie est de 50 %, et la capacité de stockage libre est de: 500 Mo. Cependant, le résultat de la tâche de sélection selon l'exemple présenté dans le La section précédente sera le suivant : Contextuelle SE-Fun1 ← { SE1 , SE3 }. Ainsi, tous les concepts spécifiés destiner a êtres comparais aux concepts de dispositifs mobiles concernant les SE1 et SE3 sont équivalent (par exemple, le SE1 a besoin d’une caméra à focale fixe et la dispositif mobile dans lequel il sera déployé a une caméra de type à mise a focale fixe). Pendant ce temps, le SE2 n’est pas adaptable aux contexte actuel de l'appareil mobile cible car il a besoin d'une camera auto-focus pour pouvoir effectuer sa tâche alors que le appareil mobile a juste un fixe. L’algorithme de sélection proposée assure que toute fonctionnalité doit avoir au moins un SE contextuel pour la mettre en œuvre. En outre, il est conçu pour fournir, après la tâche de sélection, les chemins de composition pour obtenir une application mobile adaptative comme illustré à la Fig. 4.

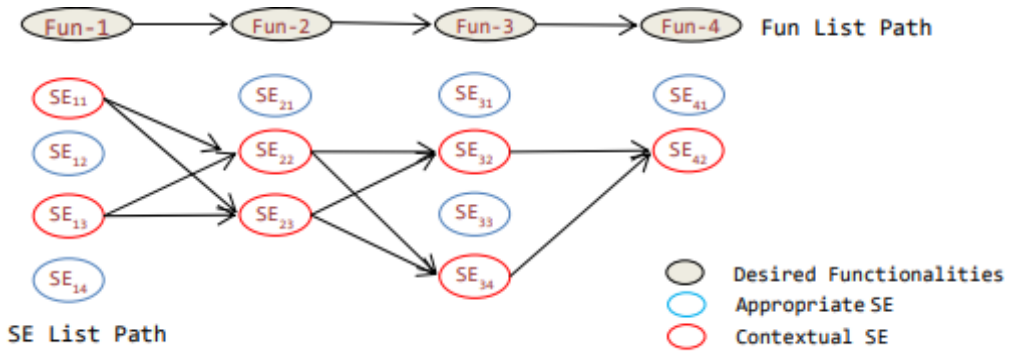

Figure 6: mécanisme de sélection

Pseudo-code: algorithme de sélection

1.
Fun-List: Fonctionnalités souhaitées.
2
SE-List: entités logicielles les mieux adaptées.
3
Téléphone : environnement de déploiement.
4
Comp-list : entités logicielles par niveau
5
--Sélection des SE contextuelle compatible avec le téléphone —
6
Pour chaque SE € SE-List {
8
Profil SE.Exécution comparé aux caractéristiques du téléphone
9
Si la comparaison est fausse alors supprimer le SE de la SE-List}
10.
--calculer le nombre de chemin possible --
11.
Cpt=1
12
Pour chaque fun € fun-List faites {
13
Cpt=cpt*nombre de SE de chaque fonctionnalité}
14
--application de l'algorithme combinatoire pour chaque SE correspondent a une fonctionnalité --
15.
Pour chaque fun € fun-List faites {
16.
Cree une nouvelle liste de comp-list
17
Pour chaque SE € SE-List correspondent à la fonctionnalité, alors {
18
Repeter {
19
Repeter {

Ajouter SE a Comp-list }

20
} jusqu'au nombre de redondance du SE
21
} jusqu'au nombre de chemin possible cpt
22
--transformation de résultat combinatoires en chemin de composition --
23
Pour chaque élément des Comp-liste faire {
24
Cree un chemin de composition}
25
--teste de validité de chemin par rapport à la batterie et l'espace de stockage --
26
Pour chaque chemin de composition faire {
27
Si (somme de consommation de batterie > batterie téléphone restante) alors
28
Suprimier le chemin
29
Sinon {
30
Si (somme consomation de batterie < meilleur chemin de consommation) alors
31.
Récupéré le chemin comme meilleur chemin de consommation}
32
Si (somme d'espace de stockage nécessaire > stockage téléphone restant) alors
33
Supprimer le chemin
34.
Sinon {
35
Si (somme d'espace de stockage nécessaire < meilleur chemin de stockage) alors
36
Récupéré le chemin comme meilleur chemin de stockage}}
37
--teste de validité de chemin par rapport a la compatibilité des SE entre eux--
38
Pour chaque chemin de composition faire {
27
Si (chemin contient SE non compatible avec un autre SE) alors
28
Suprimier le chemin

II.5 CONCLUSION

Dans ce chapitre, nous avons proposé une technique de sélection pour composer des applications mobiles adaptatives. Nous avons adopté la notation de l'ontologie pour spécifier et modéliser la composition le contexte. Nous avons également proposé deux ontologies descriptions, dont la première vise à décrire toute informations pouvant être utilisées pour caractériser la situation des l'appareils mobiles ; tandis que le second reflète la spécification des informations contextuelles des constituantes entités logicielles. Sur la base de ces descriptions, nous avons proposé un algorithme de sélection qui vise à sélectionner l'entités logicielles adaptatives et de découvrir tous les chemins de composition possibles utilisant ces entités sélectionnées.

Nous avons aussi amélioré l'algorithme pour sélectionner le chemin le plus approprié, La sélection sera effectuée en fonction du coût de composition des composants logiciels en termes d'adaptabilité et consommation. Et aussi vérifier et suprmier les chemins ou les SE son incompatibilité entre eux puisque notre travail se concentre que sur la sélection et non pas l'encapsulation des donné de sorite de chaque SE (médiateur)

Notre mécanisme de sélection se concentre essentiellement sur un ensemble de critères qui visent à assurer le bon déploiement et le bon fonctionnement d'une application mobile composite, qui n'est pas le cas pour les approches de sélection existantes.

Chapitre 3 : tests et résultats

III.1 Introduction

Dans le chapitre précèdent, nous avons présenté l'algorithme de notre mécanisme de composition. Cependant, nous visons dans ce dernier chapitre de notre mémoire à implémenter et présenter l'exécution de l'algorithme de sélection qui vise a choisir les chemins de composition. Nous utiliserons le language java pour réaliser notre travail ainsi que l'XML pour décrire les règles de chaque instance (téléphone, application). Ce chapitre vise aussi a tester a travers un exemple notre application « work office share » que nous allons présenter ultérieurement.

L'objectif finale sera d'extraire tous les chemins de composition possible pour la composition de cette application toute en respectent les règles prédéfinies de chaque entité.

III.2 Application de composition

Notre application se compose de 3 rubriques principales, et pour les 2 premier rubrique 2 sous-section chaque un que nous les présenteront en dessous

Rubrique téléphone :

Cette rubrique contient la sous-section liste téléphone qui contient la liste des téléphones mobile avec leurs paramètres que nous pouvons utiliser comme un environnement d'exécution .

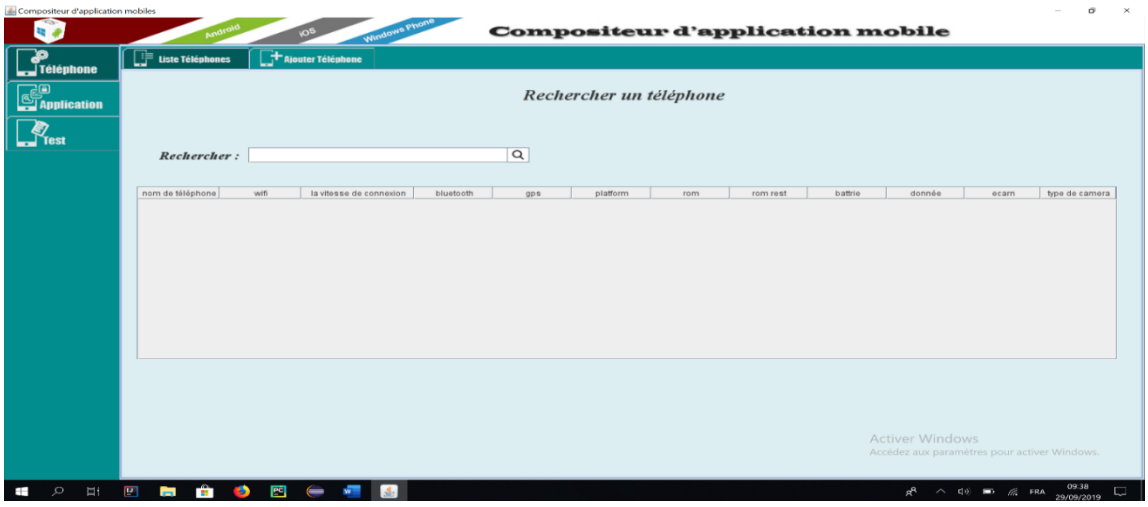

Figure 7 : interface liste des téléphones

Et la sous-section ajout téléphone qui permet de crée et d'ajouter un téléphone à la liste précédente tous ont choisissant les paramètres désirés.

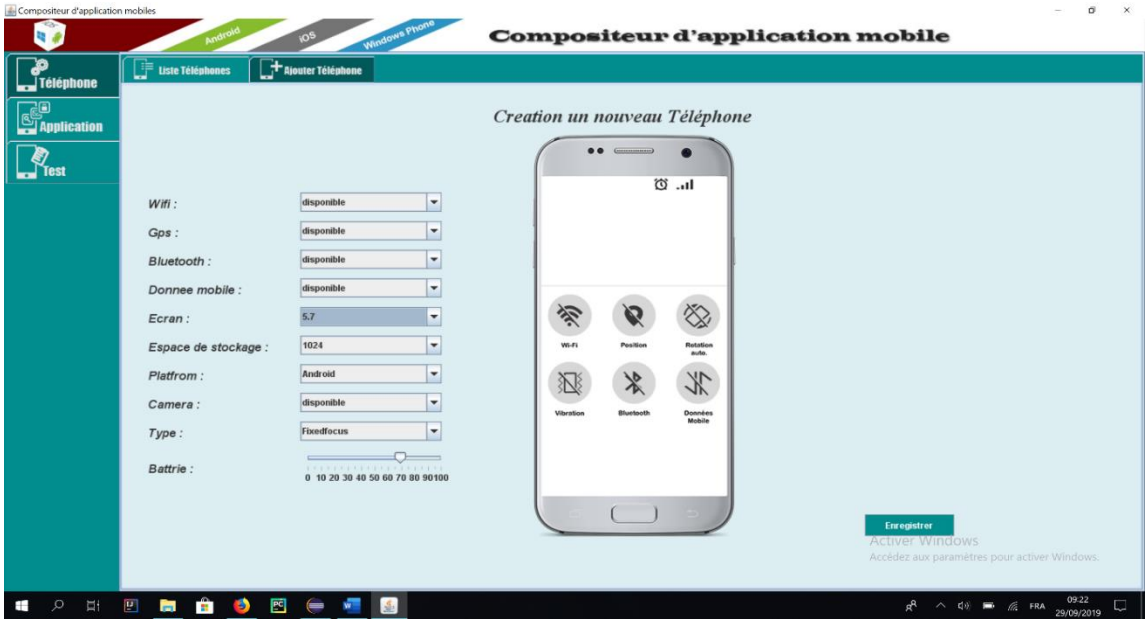

Figure 8 : interface création téléphones

Rubrique application :

La rubrique application contient elle aussi la sous-section liste application qui contient la liste des applications qu'on désire composer a partir des fonctionnalités prédéfinies.

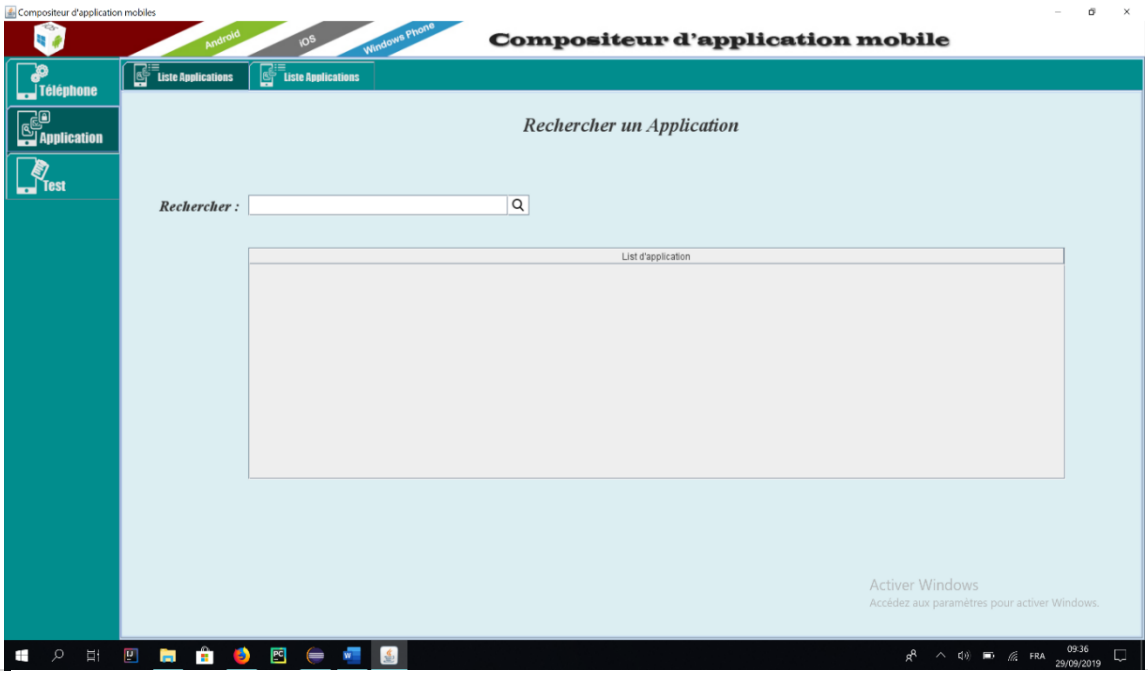

Figure 9 : interface liste des applications

Aussi la sous-section ajout application qui sert a créé et ajouter une application a la liste précédente.

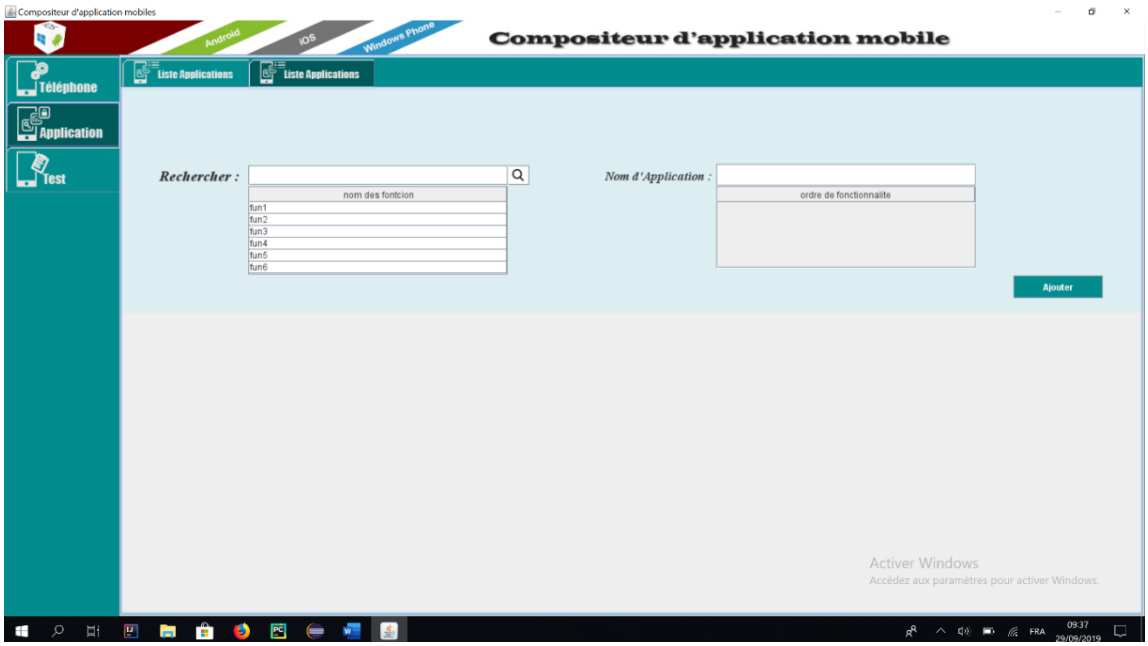

Figure 10 : interface création application

Rubrique teste :

La dernière rubrique test nous permet de choisir une application et un téléphone pour tester la possibilité de composition tout en affichant les chemins sous forme de liste ou de graph.

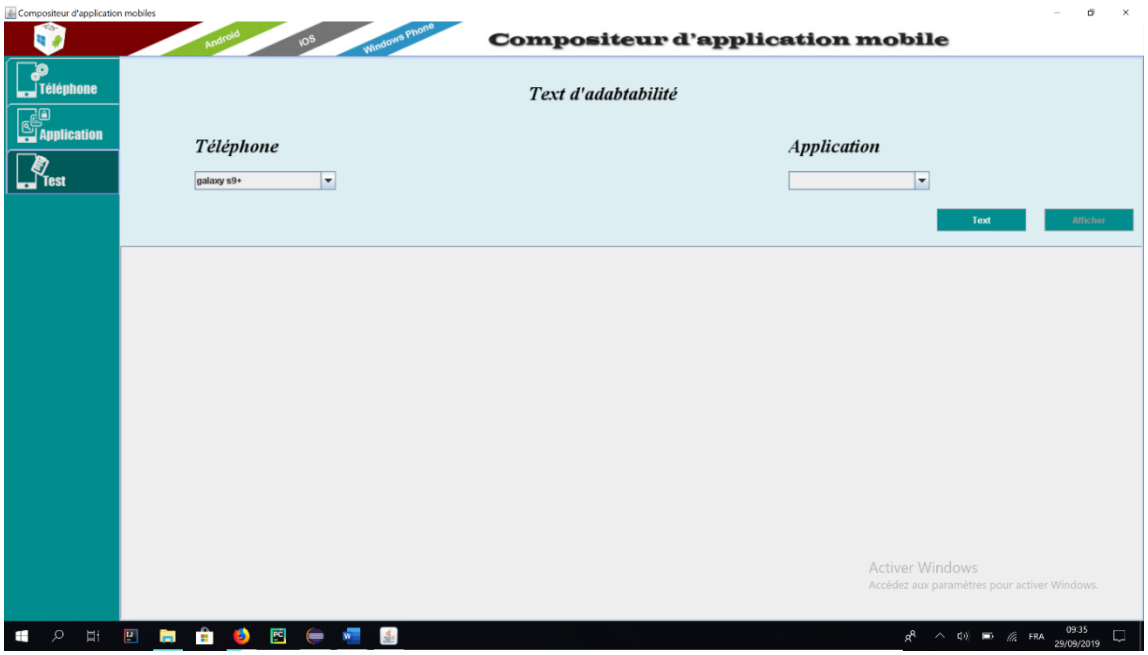

Figure 11 : interface de teste

III.3 Implémentation de l'application de composition

Pour tester notre application qui vas exécuter l'algorithme de sélection nous allons tout d'abord présenter les besoins et les règles d'exécution sur deux parties :

Partie 1 : scenario de besoin

Un utilisateur qui travaille dans un post de transmission de document délicat doit vérifier l'originalité de source des documents qu'il reçoi et retransmet le document a un destinataire dans le même office de travail , cette utilisateur utilisera une application qui traduit le code barre sur le document a un numéro de série que ce dernier serre a vérifié l'originalité de ce document , utilise une application qui se connecte a la base de donné de la société pour vérifier la compatibilité des documents ,et une autre application pour envoyer le document a un destinataire précis

Work office share est l'application que nous allons composer des différentes applications précédentes. Elle lie le code bar, vérifie l'originalité, vérifie la compatibilité des documents avec une base de données externe, envoi le document au destinataire. L'application aura besoin des différents entités logiciel pour effectuer les action présidente et d'un environnement mobile qui permet son déploiement juste et correct.

Partie 2 : exécution de la composition

De ces derniers besoins, nous allons tout d'abord crée 3 différents environnements de déploiement que nous utiliserons pour tester notre composition dans différentes situations.

nom de téléphone	wifi	la vitesse de connexion	bluetooth	gps	platform	rom	rom rest	battrie	donnée	ecarn	type de camera
Galaxy s9+	disponible	3	disponible	disponible	Android	1024	1024	100	disponible	6.3	Autofocus
Galaxy A20	non disponible	3	non disponible	non disponible	Android	128	128	25	disponible	5.1	Fixedfocus
Galaxy A50	disponible	3	disponible	disponible	Android	32	32	14	disponible	4.7	Autofocus

Tableau 2 : liste des téléphones paramétré

L’étape suivante sera de crée la composition de l’application désiré qui représentera dans notre cas l’application office work share.

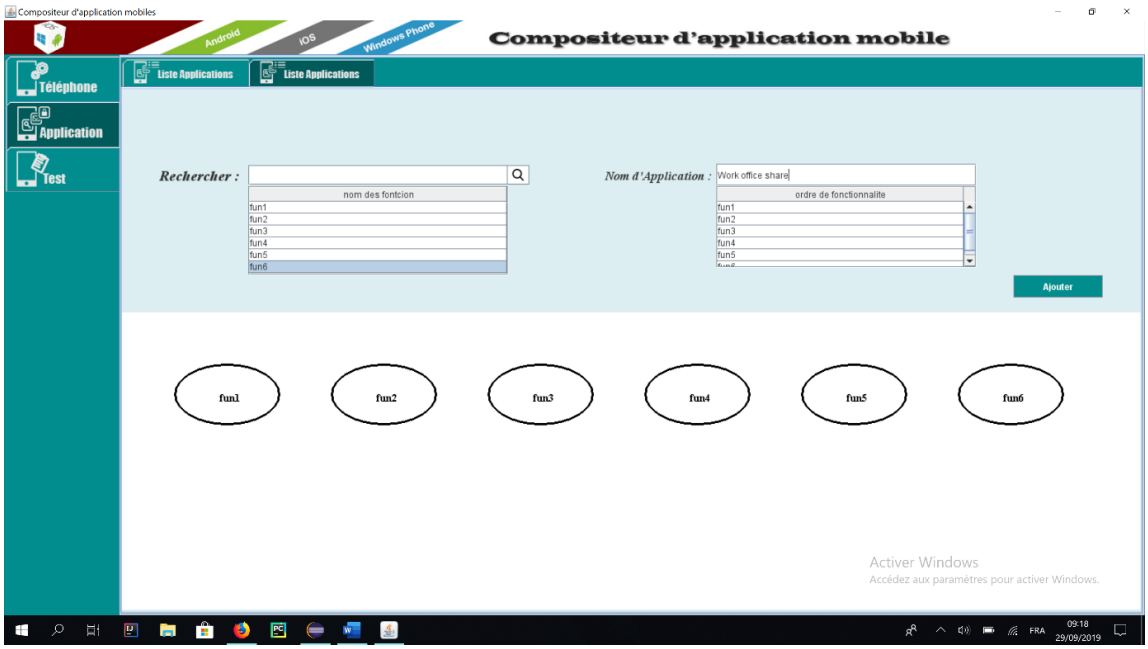

Figure 12 : composition de l’application work office share

III.4 Résultat

Avant de présenter les résultats, nous allons préciser que notre application work office share a besoin des paramètres suivantes :

Au minimum un éléments disponible {WIFI, donné mobile}

Bluetooth : disponible

GPS : non disponible

Platform : Android

Ecran taille minimale :4’7

Vitesse de connexion minimale : 1mb/s

Consommation de batterie minimum : 15%

Espace de stockage libre : 28 Mo

Après avoir fournie tous les besoins nécessaires de tester la composition, et après l'exécution de la sélection des entités logiciel, nous avons aboutie au résultat suivant pour les 3 différente environnement de déploiement sachant que nous nous sommes basés sur la batterie et le stockage restante car ces deux valeurs son variable d'un moment a un autre contrairement aux autres attribue qui son fix pour chaque téléphone.

Nous commençons tout d'abord par le premier cas ou tout les paramètre de dépoilement son optimale pour la composition de notre application :

WIFI, donné mobile, Bluetooth : Disponible

GPS : non disponible

Platform : Android

Ecran taille minimale : 6'3

Vitesse de connexion : 1mb/s

Batterie : 100%

Espace de stockage libre : 1024 Mo

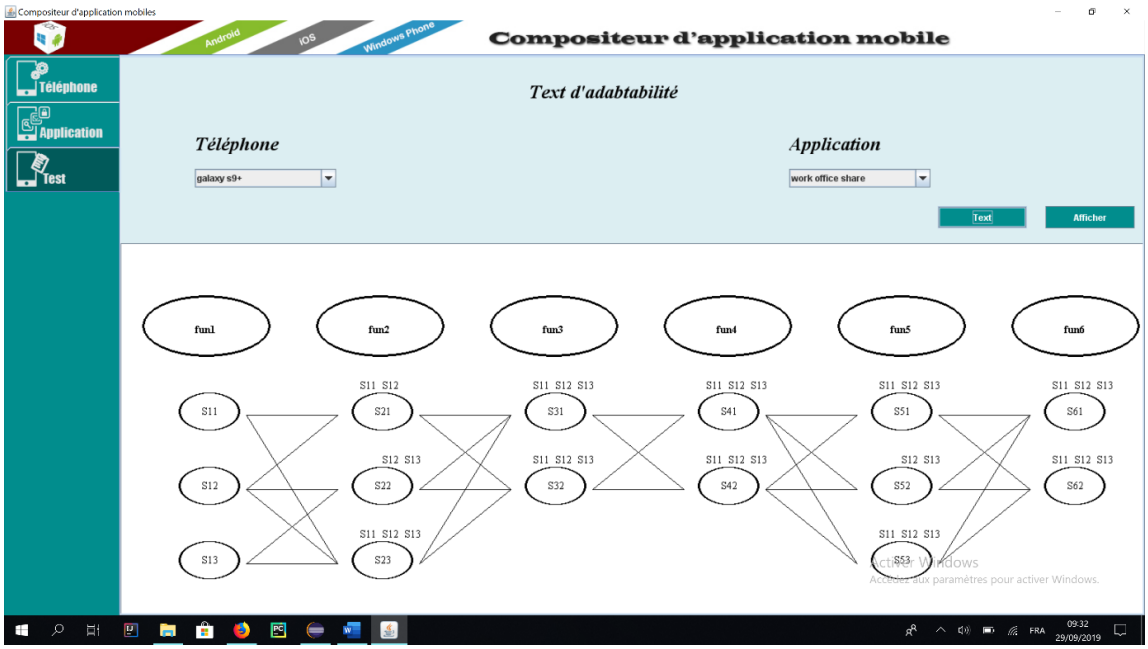

Figure 13 : graph de résultat de teste pour téléphone 1

Nous constatant du graph que presque tous les chemins de composition ont été sélectionnés.

Pour le deuxième cas les paramètre de dépoilement son casi-optimale pour la composition de notre application :

Donné mobile : disponible

WIFI, Bluetooth : non Disponible

GPS : non disponible

Platform : Android

Ecran taille minimale : 6’3

Vitesse de connexion : 1mb/s

Batterie : 25%

Espace de stockage libre : 128 Mo

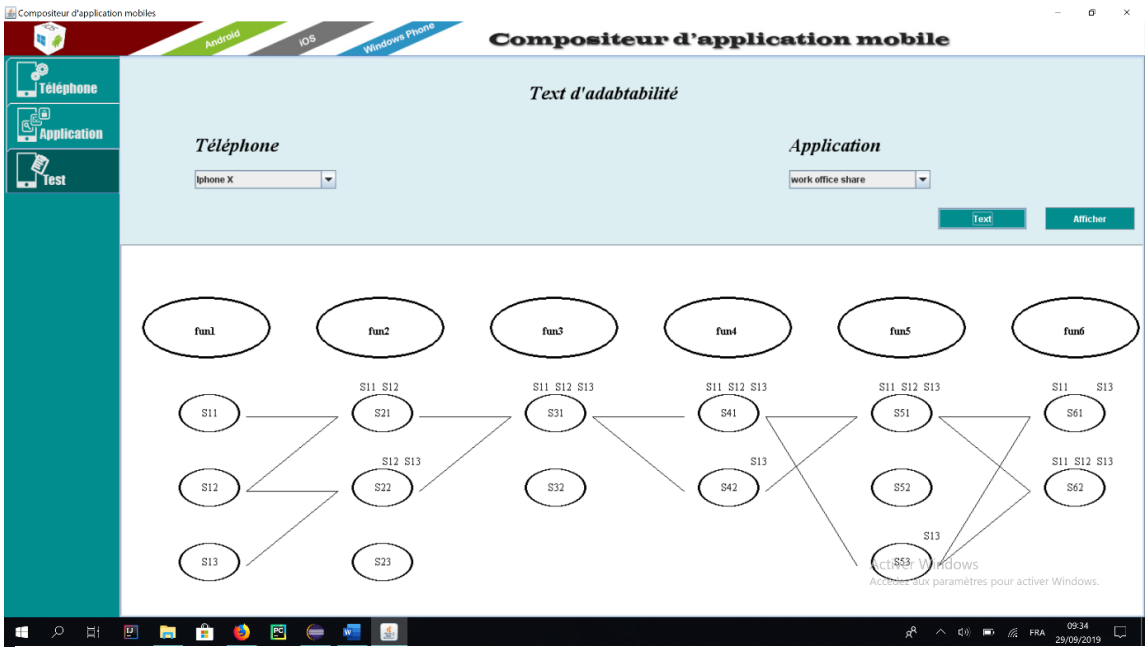

Figure 14 : graph de résultat de teste pour téléphone 2

Nous constatons que peu de chemin de composition son possible car plusieurs SE ne son pas compatible avec l’environnement de déploiement ou a cause d’insuffisance de batterie au de stockage.

Pour le troisième cas les paramètre de dépoilement son non-optimale pour la composition de notre application :

Donné mobile, WIFI, Bluetooth : disponible

GPS : disponible

Platform : Android

Ecran taille minimale : 4’7

Vitesse de connexion : 1mb/s

Batterie : 14%

Espace de stockage libre : 32 Mo

Figure 15 graph de résultat de teste pour téléphone 3

De l’erreur afficher nous constatons qu’il n’existe aucuns chemins possibles pour cet environnement de déploiement, ce résultat et inévitable vu que la batterie et inferieur au seuil minimum du déploiement de notre application work office Share.

III.5 Conclusion

L’utilisation croissante des applications mobiles a travers le monde et la variété des besoins pour chaque utilisateur a aboutie à un problème de crée une application mobile pour chaque besoin précis. De cette problématique nous avons crée une application qui permet de combiner les différents besoins d’un utilisateur et généré l’application désiré. Pour cela, nous avons repris des algorithmes de sélection et de composition dans le but de les amélioré et les automatisé pour répondre a la composition d’une application grâce à une suite de besoin introduite par l’utilisateur tout en vérifiant la compatibilité des attribue nécessaire du téléphone et aussi des entités logiciels.

REFERENCE

[1] Economides, Anastasios A., and Amalia Grousopoulou. "Students' thoughts about the importance and costs of their mobile devices' features and services."Telematics and Informatics 26.1 (2009): 57-84.

[2] Zhang,X., et al., Towards an elastic application model for augmenting the computing capabilities of mobile devices with cloud computing. Mobile Networks and Applications, 2011. 16(3): p. 270-284.

[3] Rosa, Ricardo Erikson VS, and Vicente F. Lucena Jr. "Smart composition of reusable software components in mobile application product lines."Proceedings of the 2nd International Workshop on Product Line Approaches in Software Engineering. ACM, 2011.

[4] Hofer, T., Schwinger, W., Pichler, M., Leonhartsberger, G., Altmann, J., & Retschitzegger, W. (2003, January). Context-awareness on mobile devices-the hydrogen approach. In System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on (pp. 10-pp). IEEE.

[5] Alves, V., Santos, G., Calheiros, F., Nepomuceno, V., Pires, D., Costa Neto, A., & Borba, P. (2006). Beyond code: Handling variability in art artifacts in mobile game product lines.

[6] Varshavsky, Alex, Bradley Reid, and Eyal de Lara. "A cross-layer approach to service discovery and selection in MANETs." Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on. IEEE, 2005.

[7] Zhang, Zhuoyao, et al. "An Integrated Approach to Service Selection in Mobile Ad Hoc Networks." Wireless Communications, Networking and Mobile Computing, 2008. WiCOM'08. 4th International Conference on. IEEE, 2008.

[8] Prochart, Guenter, et al. "Fuzzy-based support for service composition in mobile ad hoc networks." Pervasive Services, IEEE International Conference on. IEEE, 2007.

[9] P. E. Engelstad et al., "Service Discovery Architectures for OnDemand Ad Hoc Networks," Int'l. J.Ad Hoc and Sensor Wireless Networks, vol. 2. no. 1, Mar. 2006, pp. 27–58.

[10] V. Lenders, M. May, and B. Plattner, "Service Discovery in Mobile Ad Hoc Networks: A Field Theoretic Approach," Elsevier J. Pervasive and Mobile Computing (PMC), vol. 1, no. 3, Sept. 2005, pp. 343–70.

[11] Saha, D. and A. Mukherjee (2003). "Pervasive computing : a paradigm for the 21st century."Computer, IEEE 36(3) : 25-31. ISSN : 0018-9162.

[12]Chen, G. and D. Kotz (2000). A survey of context-aware mobile computing research,Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College. Vol. 1, No. 2.1,pp. 2-1.

[13] Lopez-Velasco (2009). Sélection et composition de services Web pour la générationd'applications adaptées au contexte d'utilisation, Thése de doctorat en informatique. GrenobleI : Université Joseph-Fourier. 18 novembre 2008. 252

[14] Tigli, J. and S. Lavirotte (2006). "Adaptation dynamique à l'environnement d'exécution : un enjeu pour l'informatique mobile et ambiante. [talk]." Séminaire invité à l'IMAG, Grenoble, France, novembre 2006

[15] Brown, P. J. (1995). "The stick-e document : a framework for creating context-aware applications." ELECTRONIC PUBLISHING-CHICHESTER-, Citeseer 8(2-3) : 259-272. ISSN : 0894-3982.

[16] Brown, P. J. et al. (1997). "Context-aware applications : from the laboratory to the marketplace." Personal Communications, IEEE 4(5) : 58-64. ISSN : 1070-9916.

[17]. Abowd, G. D. et al. (1999). Towards a better understanding of context and context-awareness. In : Gellersen Hans-W, Handheld and ubiquitous computing, Springer Berlin Heidelberg. p. 304-307. ISBN : 3540665501.

[18]Bettini, C. et al. (2010). "A survey of context modelling and reasoning techniques." Pervasive and Mobile Computing, Elsevier 6(2) : 161-180. ISSN : 1574-1192.

[19] Schilit, B. N. and M. M. Theimer (1994). "Disseminating active map information to mobile hosts." Network, IEEE 8(5) : 22-32. ISSN : 0890-8044

[20] Want, R. et al. (1992). "The active badge location system." ACM Transactions on Information Systems (TOIS), ACM 10(1) : 91-102. ISSN : 1046-8188.

[21] Rey, G. and J. Coutaz (2002). Le contexteur : une abstraction logicielle pour la réalisation de systémes interactifs sensibles au contexte. In Proceedings of the 14th French-speaking conference on Human-computer interaction (Conférence Francophone sur l'Interaction Homme-Machine), ACM. p. 105-112. ISBN : 1581136153.

[22] Brown, A. W. and K. C. Wallnau (1998). "The current state of CBSE." IEEE software, IEEE 15(5) : 37-46. ISSN : 0740-7459.

[23] Pascoe, J. (1998). Adding generic contextual capabilities to wearable computers. Wearable Computers, 1998. Digest of Papers. Second International Symposium on, IEEE. p. 92-99. ISBN : 0818690747.

[24] Pascoe, J. et al. (1998). Human-Computer-Giraffe Interaction-HCI in the Field. In Workshop on Human Computer Interaction with Mobile Devices.

[25] Cheverst, K. et al. (2000). Providing tailored (context-aware) information to city visitors. In Adaptive hypermedia and adaptive web-based systems, Springer. p. 73-85. ISBN : 3540679103.

[26]Dey, A. K. et al. (2001). "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications." Human-computer interaction, L. Erlbaum Associates Inc. 16(2) : 97-166. ISSN : 0737-0024.

[27] Hoareau, C. and I. Satoh (2009). "Modeling and processing information for context-aware computing : A survey." New Generation Computing, Springer 27(3) : 177-196. ISSN : 0288-3635.

[28] Chen, G. and D. Kotz (2000). A survey of context-aware mobile computing research, Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College. Vol. 1, No. 2.1, pp. 2-1.

[29]Buchholz, S. et al. (2004). Comprehensive structured context profiles (cscp) : Design and experiences. In Pervasive Computing and Communications Workshops, Proceedings of the Second IEEE Annual Conference on, IEEE. p. 43-47. ISBN : 0769521061.

[30] Indulska, J. et al. (2003). Experiences in using cc/pp in context-aware systems. Mobile Data Management, Springer. p. 247-261. ISBN : 3540003932.

[31](Mukhtar, H. et al. (2008). A policy-based approach for resource specification in small devices. In Mobile Ubiquitous Computing, Systems, Services and Technologies 2008 (UBICOMM’08), The Second International Conference on, IEEE. p. 239-244. ISBN : 0769533671.

Mukhtar, H. et al. (2009). User preferences-based automatic device selection for multimedia user tasks in pervasive environments. Networking and Services, 2009. ICNS’09. Fifth International Conference on, IEEE.

[32] Derdour, M. (2012). Modélisation et implémentation d'un systéme d'information de gestion de flux multimedia pour des architectures logicielles intégrant des appareils sans-fil mobiles. Thése de doctorat en Science : Réseaux., Annaba : Univérsié Badji Mokhtar. 220 p

[33]Rasch, K. et al. (2011). "Context-driven personalized service discovery in pervasive environments." World Wide Web, Springer 14(4) : 295-319. ISSN : 1386-145X.29..Abi-Char et al., 2010

[30] Kiss, C. (2007). "Composite capability/preference profiles (cc/pp) : Structure and vocabularies 2.0." W3C working draft 30

[31]Strang, T. and C. Linnhoff-Popien (2004). A context modeling survey. In Workshop on Advanced Context Modelling,Reasoning and Management, UbiComp 2004, The Sixth International Conference on Ubiquitous Computing, Nottingham/England, 2004. (En ligne). Disponible sur : http ://elib.dlr.de/7444/1/Ubicomp2004ContextWSCameraReadyVersion.pdf

[32]Strang, T. et al. (2003). CoOL : A Context Ontology Language to enable Contextual Interoperability. In : J.-B. Stefani, I. Dameure and D. Hagimont (eds.), Distributed Applications and Interoperable Systems, Paris. p. 236–247. ISBN : 978-3-540-20529-6.

[33.] Ay, F. (2007). Context modeling and reasoning using ontologies. (en ligne). University of Technology Berlin. Disponile sur : http ://www.ponnuki.de/cmaruo/cmaruo.pdf.

[34] Wang, X. H. et al. (2004). Ontology based context modeling and reasoning using OWL. In Pervasive Computing and Communications Workshops, Proceedings of the Second IEEE Annual Conference on, IEEE. p. 18-22. ISBN : 0769521061.

[35] Preuveneers, D. et al. (2004). Towards an extensible context ontology for ambient intelligence. In Proceedings of the 2nd European Symposium on Ambient Intelligence (EUSAI 2004), Springer : p. 148-159. ISBN : 3540237216.

[36] Ay, F. (2007). Context modeling and reasoning using ontologies. (en ligne). University of Technology Berlin. Disponile sur : http ://www.ponnuki.de/cmaruo/cmaruo.pdf.

[37]Brel, C. (2013). Composition d'applications multi-modèles dirigée par la composition des interfaces graphiques, Thése de doctorat en informatique. Antipolis : Université Nice Sophia, 28 juin 2013. 204 p.

[38] Martin, D. et al. (2005). Bringing semantics to web services : The OWL-S approach. In : J. Cardoso and A. Sheth, Semantic Web Services and Web Process Composition, Springer : p. 26-42. ISBN : 978-3-540-24328-1.

[39] Benatallah, B., et al. (2005). Service Composition : Concepts, Techniques. In : Z. Stojanovic and A. Dahanayake, Service-Oriented Software System Engineering : Challenges and Practices, Idea Group. p. 48-66. ISBN : 1-59140-428-2.

[40] Alonso, G. et al. (2004). Web services : Concepts, Architectures and Applications, Springer Verlag. p. 320 . ISBN : 3642078885.

[41] Abi Lahoud, E. (2010). Composition dynamique de services : application à la conception et au développement de systémes d'information dans un environnement distribué, Thése de doctorat en informatique. Dijon : Université de BOURGOGNE, 11/02/2010. p. 235.

[42] Papazoglou, M. P. and W.-J. Van Den Heuvel (2007). "Service oriented architectures : approaches, technologies and research issues." International Journal on Very Large Data Bases (VLDB), Springer 16(3) : 389-415. ISSN : 1066-8888

[43] Di Nitto, E. et al. (2008). "A journey to highly dynamic, self-adaptive service-based applications." Automated Software Engineering 15(3-4) : 313-341. ISSN : 0928-8910.

[44] Chang, Y.-C. et al. (2008). Solving the service composition puzzle. In Proceedings of the 2008 IEEE International Conference on Services Computing (SCC 2008). IEEE Computer Society : Washington, DC, USA. Vol. 2. p. 387-394. ISBN : 0769532837

[45] Verma, K. et al. (2005). The METEOR-S approach for configuring and executing dynamic web processes. Technical Report TR6-24-05, The University of Georgia, Computer Science Department, LSDIS Lab, Athens, Georgia

[46] Kalasapur, S. et al. (2007). "Dynamic service composition in pervasive computing." Parallel and Distributed Systems, IEEE Transactions on 18(7) : 907-918. ISSN : 1045-9219.

[47] Lee, J. et al. (2008). "Dynamic service composition : A discovery-based approach." International Journal of Software Engineering and Knowledge Engineering, World Scientific 18(2) : 199-222. ISSN : 0218-1940

[47] Li, L. et al. (2009). Trust-oriented composite service selection and discovery. In : L. Baresi, C. Chi and J. Suzuki, Service-Oriented Computing, Springer Berlin Heidelberg : p. 50-67. ISBN : 978-3-642-10382-7.

[48]Suraci, V. et al. (2007). Context-aware semantic service discovery. In Mobile and Wireless Communications Summit, 2007. 16th IST, 1-5 july., IEEE. p. 1-5. ISBN : 9638111666

[49]Toma, I. et al. (2005). A P2P discovery mechanism for web service execution environment. In Second WSMO Implementation Workshop.

[50] Merla, C. B. (2010). "Context-aware match-making in semantic web service discovery." International Journal of Advanced Engineering Sciences and Technologies 9(2) : 243-247. Meyer, B. (1985). "On formalism in specifications." IEEE software 2(1) : 6-26. ISSN : 0740-7459.

[51] He, Q. et al. (2013). "A decentralized service discovery approach on peer-to-peer networks." Services Computing, IEEE Transactions on 6(1) : 64-75. ISSN : 1939-1374.

[52] Rake, J. et al. (2009). Enhancing semantic service discovery in heterogeneous environments. In : W. Abramowicz, Business Information Systems, Springer Berlin Heidelberg. p. 205-216. ISBN : 978-3-642-01189-4.

[53] Ma, Q. et al. (2008). A semantic QoS-aware discovery framework for web services. In Web Services 2008 (ICWS'08). IEEE International Conference on, IEEE. p. 129-136. ISBN : 0769533108.

[54] Gu, X. et al. (2004). SpiderNet : An integrated peer-to-peer service composition framework. In High performance Distributed Computing, Proceedings. 13th IEEE International Symposium on, IEEE. p. 110-119. ISBN : 0769521754.

[55]Beauche, S. and P. Poizat (2008). Automated service composition with adaptive planning. In : F. George, L. Winfried, Service-Oriented Computing-ICSOC 2008, Springer. p. 530-537. ISBN : 9783642012471

[56] Varshavsky, A. et al. (2005). A cross-layer approach to service discovery and selection in MANETs. In Mobile Adhoc and Sensor Systems Conference(MASS), IEEE International Conference on, 7-7 Nov. 2005, Washington, DC, IEEE Computer Society. 8 pp.-466. ISBN : 0780394658

[57] Zhang, X. et al. (2011). "Towards an elastic application model for augmenting the computing capabilities of mobile devices with cloud computing." Mobile Networks and Applications, Springer-Verlag New York, Inc. 16(3) : 270-284. ISSN : 1383-469X

[58] Ay, Feruzan. "Context Modeling and Reasoning using Ontologies (July 2007)."

[59] Dey, A. K. (2001). "Understanding and using context." Personal and ubiquitous computing, Springer-Verlag 5(1) : 4-7. ISSN : 1617-4909

[60] Définition Application mobile. Date de consultation : 03 2015. http ://www.denitions-webmarketing.com/Denition-Application-mobile- application embarquee ou service mobile.

[61] http ://www.mobileenfrance.com/2009/08/18/application-embarquee-ou-service-mobile- %E2%80%93-partie-1-denitions/

[62] aborder un projet de mobilité.

http ://fr.clever-age.com/veille/blog/comment-aborder-un-projet-de-mobilite.html

[63] conceptiondéveloppement d'application pour SmartphoneTablettes.

http ://www.synertic.fr/sites/default/les/pdf/synertic%20Mobile_v2.pdf