TABLES DES MATIERES

TABLES DES FIGURES

TABLES DES TABLEAUX

Introduction générale

Les entreprises ainsi que leurs systèmes d'information sont de plus en plus exposés à un ensemble d'attaques complexes et engendrant des dégâts multiples qui peuvent même mener à mettre fin à l'activité assurée. Ces attaques exploitent des vulnérabilités disponibles pour les composants du système d'information du service jusqu'au la documentation tel que la politique de sécurité.

Dans ce contexte, l'audit de sécurité est parmi les disciplines qui deviennent primordiale pour évaluer le niveau de sécurité des systèmes d'information par l'identification des vulnérabilités et la proposition des corrections nécessaires afin de réduire les dégâts.

D'un point de vue global, l'audit de sécurité concerne l'état des lieux d'une entreprise, ses moyens humains et techniques, son fonctionnement, la conformité de ses installations et son organisation interne en termes de sécurité et de respect des normes ou règlements. Les menaces directes et indirectes sont évidemment prises en compte ainsi que l'identification des points faibles, des valeurs critiques de sécurité et autres dangers potentiels. L'audit de sécurité consistera dès lors à établir un bilan, une synthèse et des recommandations d'ordre technique ou de fonctionnement (interne et externe) pour améliorer la sécurité des entreprises.

Actuellement, le marché de la téléphonie portable connaît une véritable révolution, menée par Android (Google), iOS (Apple), Windows Phone (Microsoft), RIM Bada et Symbian. L'Android Os de Google atteint une part de marché de 46.19 pour cent en mars 2016, c'est le plus grand des fabricants mobile [1].

De même, il est vrai que les systèmes mobiles évoluent rapidement et en particulier en matière de téléchargement des applications. Selon les dernières statistiques, Google Play Store compte 1 600 000 applications avec 50 milliards d'applications téléchargées [2].

Dans ce cadre, l'objectif de ce projet, intitulé «KMCT Audit» sera ainsi de contribuer à élaborer un service d'audit de sécurité des équipements réseaux développé sous le système mobile Android, permettant essentiellement d'évaluer un équipement réseau afin de déterminer ses vulnérabilités et proposer les recommandations nécessaires pour la protection.

Ce présent rapport sera ainsi composé de sept chapitres selon la méthodologie 2TUP [1]:

- Un premier chapitre sera dédié à la présentation du cadre du projet, l'établissement d'accueil « KMCT » et les services qu'elle offre.

[1]2TUP : 2 Track Unified Process

- Le chapitre suivant présentera une étude de l'existant soit un état de l'art,
- Le troisième chapitre sera consacré à la spécification de la démarche et la méthodologie qui seront adaptées tout au long du projet,
- Le quatrième chapitre intitulé « Spécification des besoins » mettre en évidence les besoins fonctionnels et les besoins techniques,
- Le cinquième chapitre présentera la phase d'analyse de la méthode 2TUP,
- Le chapitre qui suit exposera la vue statique et dynamique de l'application soit la partie « Conception ».
- Et le dernier chapitre sera consacré à la réalisation et les tests qui suivront.

Chapitre I. Cadre du projet

Introduction

Dans cette partie nous présentons tous d'abord le cadre dans lequel s'est déroulé notre projet de fin d'études ensuite nous décrivons minutieusement l'entreprise d'accueil ; ses domaines d'activités ainsi que les services qu'elle offre.

I. Présentation du cadre

Dans le but d'obtenir un Mastère professionnel en Nouvelles Technologies des Télécommunications et Réseaux à l'université virtuelle de Tunis, nous avons effectué notre projet de fin d'études au sein de l'établissement Khamis Mushait College of Technology (KMCT) qui est une institution de formation technique et professionnelle à l'Arabie saoudite au niveau de leur département informatique.

Lors du déroulement de ce stage, nous avons essayé d'employer et d'exploiter au mieux nos connaissances et nos compétences acquises dans le domaine de l'informatique et dans les langages de programmation, tout en veillant à bien développer et structurer nos idées.

Ainsi, cela nous permettra de mieux nous préparer soit à une poursuite d'études soit à l'insertion dans la vie professionnelle.

II. Entreprise d'accueil

Khamis Mushait College of Technology est l'une des institutions de la corporation de formation technique et professionnelle de l'Arabie saoudite (Technical and vocational training corporation : TVTC) qui vise à devenir la lumineuse et la distincte des formations techniques et professionnelles dans la région. L'établissement a trois principales spécialités : « Sécurité alimentaire » qui est une spécialité au département de l'environnement de la technologie, la spécialité « Réseaux » au département informatique et la spécialité « Gestion de l'entrepôt » au département administratif de la technologie.

L'établissement KMCT a exercé beaucoup d'efforts pour répondre aux besoins du marché du travail local en lui fournissant des cadres qualifiés et formés pour permettre aux différents secteurs un accès au marché concurrentiel et une croissance efficace et efficiente.

Le diplômé en science informatique va obtenir un diplôme dans l'une des spécialités suivantes :

- Les réseaux informatiques de la technologie (Computer Networks Technology)
- La technologie de gestion de systèmes de réseau (Network Technology Management Systems)
- Multimédia et graphique des sites web (Multimedia and Website Graphics)

L'institution est fière des spécialistes qualifiés, des ingénieurs et de nombreux laboratoires et elle appelle chacun de bénéficier des potentialités de l'école en particulier les services fournis

aux secteurs public et privé par l'intermédiaire du centre de services communautaires.

II.1. Fiche d'identité de l'entreprise d'accueil

Le tableau ci-dessous représente la fiche d'identité de l'entreprise d'accueil KMCT.

Caractéristique	Valeur
Dénomination sociale	Khamis Mushait College of Technology
Identité visuelle	المؤسسة العامة للتدريب التقني والمهني Technical and Vocational Training Corporation
Secteur d'activité	Formation Technique et Professionnelle
Site Web	http://www.tvtc.gov.sa/english/trainingunits/collegesoftechnology/kmct/Pages/default.aspx
Siège	Khamis mushait, Abha, Arabie Saoudite
QR Code	

Tableau 1. Fiche d'identité du KMCT

II.2. Fondation et développement du TVTC

Le début de la formation technique et professionnelle dans le Royaume remonte à une époque reculée. A cette époque, il a été distribué entre les trois autorités gouvernementales : le Ministère de l'éducation a couru les écoles de formation (industrielle, agricole et commerciale), le Ministère du travail et des affaires sociales a couru la formation professionnelle « Centres de formation professionnelle » et le ministère des Municipalités et des Affaires rurales a couru les assistants instituts.

Avoir l'intérêt pour la préparation de la main-d'œuvre pour les domaines techniques et professionnels et le besoin croissant de qualification de la jeunesse saoudienne pour les domaines techniques et industriels, le Royaume a recueilli tous les domaines de la formation technique et professionnelle dans une organisation faîtière - le TVTC.

Ensuite, un arrêté royal n ° 30/m, en date du 08/10/1400, prévoyait la création du TVTC et de regrouper tous les instituts techniques et centres de formation professionnelle dans le cadre de l'TVTC. En conséquence, TVTC a commencé à poursuivre sa fonction, le développement de ses programmes de façon continue à obtenir avec les besoins du Royaume et d'améliorer les ressources humaines pour répondre aux besoins du marché du travail [3].

II.3. Problématique

Aujourd'hui, votre réseau informatique possède un niveau de sécurité élevé. Vos serveurs et machines sont à jour, aucune vulnérabilité connue touche vos systèmes, mais qu'en est-il pour demain ? En effet, tous les jours de nouvelles vulnérabilités sont découverts. Ces failles peuvent toucher les systèmes d'exploitation ou services que vous possédez au sein de votre infrastructure. Comment être alerté au plus tôt de la présence d'une vulnérabilité qui affecte vos équipements ? Cette problématique trouve sa réponse dans les audits de vulnérabilités récurrents et l'adoption d'une démarche proactive.

Pourquoi attendre que votre service de veille vous avertisse de la sortie d'une nouvelle faille alors que vous pourriez, a peine quelques heures après sa découverte, en tester la présence réelle sur vos machines et détecter instantanément si vous êtes vulnérable ou non ?

Compte tenu de l'importance de garantir la sécurité des systèmes d'information, ce projet de fin d'étude, invite ainsi à la réflexion sur les avertissements et les recommandations de sécurité au sein de l'audit mobile qui permet aux auditeurs de pouvoir déterminer les vulnérabilités de leurs systèmes d'information partout et à tout moment.

Conclusion

Dans ce chapitre, nous avons présenté le cadre du projet, soit une présentation du KMCT et ces différents services et par la suite une exposition de la problématique de notre projet. Le deuxième chapitre sera consacré à l'état de l'art.

Chapitre II. Etat de l’art

Introduction

Suite à une présentation de l'entreprise d'accueil et une description définie le cadre de notre projet, ce chapitre étalera les différentes phases du processus d'audit ainsi que les normes utilisées durant une mission d'audit de sécurité. Par la suite, nous allons présenter les systèmes d'exploitation pour les mobiles, les applications mobiles et leur tendance sur le marché. Enfin, nous allons étudier et synthétiser quelques solutions existantes et qui concernent en particulier les scanners de vulnérabilités.

I. Audit de sécurité

Dans cette section, nous allons définir les différents processus liés à un audit de sécurité.

I.1. Définition

Le terme «**Audit**» est apparu dans les années 70 et a été utilisé de manière relativement aléatoire. Nous considérons par la suite un "audit sécurité de l'information" comme une mission d'évaluation de conformité par rapport à une politique de sécurité ou à défaut par rapport à un ensemble de règles de sécurité. Une mission d'audit ne peut ainsi être réalisée que si l'on a défini auparavant un référentiel, c'est-à-dire en l'occurrence, un ensemble de règles organisationnelles, procédurales ou/et techniques de référence. Ce référentiel permet au cours de l'audit d'évaluer le niveau de sécurité réel du " **terrain** " par rapport à une cible [4].

Pour évaluer le niveau de conformité, ce référentiel doit être :

- **Complet** (mesurer l'ensemble des caractéristiques : il ne doit pas s'arrêter au niveau système, réseau, télécoms ou applicatif, de manière exclusive, de même, il doit couvrir des points techniques et organisationnels).
- **Homogène** : chaque caractéristique mesurée doit présenter un poids cohérent avec le tout.
- **Pragmatique** : c'est-à-dire, aisé à quantifier (qualifier) et à contrôler. Ce dernier point est souvent négligé.

I.2. Contenu de l'audit

L'opération d'audit prend notamment en compte les éléments suivants :

- Descriptif du matériel, des logiciels et de la documentation.
- Appréciation globale de l'adéquation entre les besoins et le système d'information existant.
- Examen des méthodes d'organisation, de contrôle et de planification des services informatiques.
- Appréciation de la formation, de la qualification et de l'aptitude du personnel.

- Appréciation de la qualité, de l'accès, de la disponibilité et de la facilité de compréhension de la documentation.

I.3. Les approches d'audit de sécurité

Un audit peut être mené en suivant deux approches (non exclusives) :

- **Une approche " boîte blanche " :** les auditeurs ont accès à l'organisation, aux données et traitements réalisés, aux documents et processus appliqués. Ils font appels aux interlocuteurs autant que de besoin.
- **Une approche " boîte noire " :** l'audit est alors mené en partant d'une connaissance limitée du système d'information cible. Les auditeurs opèrent sans accès à priori aux systèmes et données. Les " tests d'intrusion " ou " tests intrusifs " font partie de cette catégorie d'audit.

Les deux approches précédentes sont complémentaires, en effet :

- **Une approche " boîte blanche " :** est en général un audit plus homogène, l'évaluation possède une caractéristique d'analyse " en complétude " et les aspects techniques et organisationnels sont traités de manière uniforme.
- **Une approche " boîte noire " :** est en général un audit avec une vue plus parcellaire, révélant plutôt des lacunes ciblées à forte orientation technique. Il est plus délicat de cerner la "complétude " de cette approche mais cela permet de simuler des incidents ou attaques logiques sur le système d'information. Les "tests d'intrusion " font partie de cette seconde catégorie.

I.4. Cycle de vie d'un audit de sécurité

La mission d'audit de sécurité informatique est effectuée selon un processus cyclique permettant d'étudier le niveau de sécurité du système d'information d'un point de vue :

- **Technique :** les points d'entrées sur le réseau, les équipements de sécurité, les protocoles mis en œuvre, etc.
- **Organisationnel :** étude des procédures de définition, de mise en place et de suivi de la politique de sécurité, etc.

La Figure 1 suivante présente le cycle de vie d'un audit de sécurité.

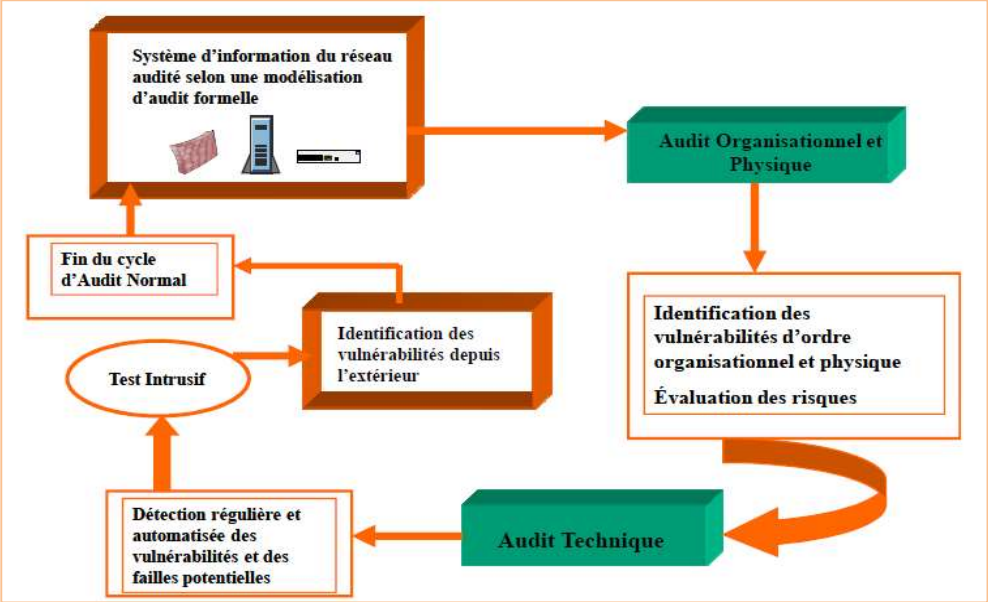

Figure 1. Cycle de vie d'un audit de sécurité

Principalement, la mission sera subdivisée en deux volets : Les tableaux suivants (Tableau 2 et Tableau 3) dégagent les objectifs et le déroulement de chaque niveau de sécurité.

	Audit Niveau 1 : Audit Organisationnel et Physique, Analyse de Risque
Objectifs	▪ S'intéresser à l'aspect physique et organisationnel de l'organisme cible, à auditer. ▪ Avoir une vue globale de l´état de sécurité du système d´information et d´identifier les risques potentiels sur le plan organisationnel.
Déroulement	▪ Suivre une approche méthodologique qui s'appuie sur « une batterie de questions ». ▪ Le questionnaire préétabli devra tenir compte et s'adapter aux réalités de l'organisme à auditer. ▪ L'auditeur évalue les failles et apprécie le niveau de maturité en termes de sécurité de l'organisme, ainsi que la conformité de cet organisme par rapport à la norme référentielle de l'audit.

Tableau 2. Audit niveau 1

	Audit Niveau 2 : Audit Technique
Objectifs	▪ Réalisé suivant une approche méthodique allant de la découverte et la reconnaissance du réseau audité jusqu'au sondage des services réseaux actifs et vulnérables. ▪ Faire apparaître les failles et les risques, les conséquences d'intrusions ou de manipulations illicites de données.

	▪ L'auditeur apprécie l'écart avec les réponses obtenues lors des entretiens. Il testera la robustesse de la sécurité du système d'information et sa capacité à préserver les aspects de confidentialité, d'intégrité, de disponibilité et d'autorisation.
Déroulement	▪ Vu les objectifs escomptés lors de cette étape, leurs aboutissements ne sont possibles que par l'utilisation de différents outils. ▪ L'ensemble des outils utilisés doit couvrir entièrement ou partiellement la liste non exhaustive des catégories ci-après : - Outils de sondage et de reconnaissance du réseau. - Outils de test automatique de vulnérabilités du réseau. - Outils spécialisés dans l'audit des équipements réseau (Routeurs, Switchs). - Outils spécialisés dans l'audit des systèmes d'exploitation. - Outils d'analyse et d'interception de flux réseaux. - Outils de test de la solidité des objets d'authentification (fichiers de mots clés). - Outils de test de la solidité des outils de sécurité réseau (firewalls, IDS, outils d'authentification). - Outils de scan d'existence de connexions dial-up dangereuses (wardialing). - Outils spécialisés dans l'audit des SGBD existants.

Tableau 3. Audit niveau 2

I.5. Démarche de réalisation d'un audit de sécurité

La figure suivante (Figure 2) présente les différentes phases de processus d'audit, il peut être découpé en Cinque phases [5] :

Figure 2. Les phases de processus d'audit

I.5.1. Phase 1 : Définition de la mission d'audit

Durant cette phase, l'objectif de l'audit est fixé ainsi que les objectifs pour mener l'audit. Cette phase permet de mettre en œuvre l'audit de sécurité en définissant les champs d'étude, les périmètres de la mission ainsi que le planning de réalisation.

Un audit ne peut être réalisé que suite à la :

- Définition du sujet d'audit.
- Fixation les domaines sujet d'audit, des objectifs de l'audit.
- Définir le périmètre d'audit.
- Fixation des raisons pour lesquelles l'audit doit être mené.
- Fixation des résultats attendus suite à l'exécution de l'audit.
- Définition des buts à atteindre, l'étendu de l'audit.
- Identification des fonctions et des systèmes à auditer pour les domaines sujets d'audit.
- La bonne définition de la mission a un effet sur toutes les phases du processus d'audit.

I.5.2. Phase 2 : Préparation à l'audit

Cette phase est aussi appelée la phase de pré audit. Elle constitue une phase importante pour la réalisation de l'audit sur terrain. En effet, c'est au cours de cette phase que se dessinent les grands axes qui devront être suivis lors de l'audit sur terrain. Elle se manifeste par des rencontres entre auditeurs et responsables de l'organisme à auditer. Au cours de ces entretiens, les espérances des responsables vis-à-vis de l'audit devront être exprimées. Aussi, le planning de réalisation de la mission de l'audit doit être fixé.

a. Connaître l'environnement à auditer

L'auditeur doit commencer par prendre connaissance du contexte, effectuer un pré-audit rapide afin de repérer les objectifs de la mission.

Cette mission préliminaire comprend :

- L'étude de la documentation disponible.
- Collecte des informations à partir des entretiens menés avec les responsables concernés.

L'auditeur est amené à établir :

- Des objectifs précis : ce qu'il faut rechercher, vérifier, prouver.
- Les moyens et actions : information à rassembler et analyser, contrôles, niveau de confiance à atteindre.
- Un plan de travail, selon la nature et le volume des tâches, avec des délais prévisionnels.

- La liste des contacts nécessaires pour l'accomplissement de chaque phase du processus d'audit.

b. Classification des domaines/zones

Durant la phase de préparation à l'audit, l'auditeur doit fixer les domaines qui doivent être audités :

- Les systèmes de production.
- Les systèmes en cours de développement.
- La gestion des composants du SI.
- Les contrôles mis en place (procédures, politiques, mécanismes de sécurité, etc.).

L'auditeur doit regrouper les composants du SI par domaine/zone tout en affectant à chaque domaine un niveau de risque. La classification des zones par niveau de risque sert pour décider sur le volume de tests et d'opérations menées sur une zone par rapport à une autre. Elle sert aussi pour décider si les tests se font sur des échantillons ou sur la totalité des biens.

c. Notification/Plan de travail

Un audit est généralement réalisé par un groupe d'auditeurs (équipe). Les auditeurs doivent notifier les audités de la date de début de l'audit et ce qui est nécessaire pour le bon déroulement de la mission. Cette notification permet aux responsables (audités) de se préparer en arrangeant l'accès aux ressources du SI. Suite à ces négociations, l'équipe d'audit prépare le plan de travail qui détaille les étapes à réaliser durant le processus d'audit et qui inclut :

- La définition des tâches.
- L'affectation des auditeurs aux tâches définies.
- La fixation des délais pour chaque phase.

Ce plan de travail peut être sujet de modifications suite à l'apparition de nouvelles informations (ex. par entretien) sur le système audité. Ce plan s'accompagne d'un budget exprimé soit en jours soit en termes monétaires.

I.5.3. Phase 3 : Exécution

Dans cette phase, l'auditeur doit collecter des données et des preuves à travers des tests et des opérations. Les données collectées et des résultats des tests seront analysés et examiner afin de dégager les insuffisances et faire des jugements.

Les résultats préliminaires seront documentés et communiqués aux responsables.

a. La preuve

C'est toute information utilisée par l'auditeur pour justifier/renforcer un jugement porté sur un composant du SI audité. C'est le résultat des tests menés sur les composants du SI. Les jugements faits par l'auditeur doivent être basés sur des preuves pertinentes et suffisantes.

La preuve peut inclure :

- Les observations de l'auditeur.
- Les notes prises suite aux entretiens menés avec le personnel concerné.
- Les informations collectées à partir de la documentation relative à l'entreprise.
- Les résultats obtenus suite à l'exécution des procédures de test et l'exécution des outils d'audit.

b. Classification des preuves

Il existe plusieurs catégories de preuves qui sont :

- **Preuve physique :** obtenue généralement suite à l'observation du personnel, d'un événement et peut être sous forme de photos, cartes, plans, etc.
- **Preuve par témoignage :** peut-être des comptes rendus, des questionnaires. Ces types de preuves ne sont pas concluants en elles-mêmes car elles représentent l'opinion d'une autre personne.
- **Preuve sous forme de document :** peut-être des lettres, des accords, contrats, directives, note, et d'autres documents utilisés au sein de l'entreprise. La source du document affecte le niveau de fiabilité du document.
- **Preuve analytique :** généralement dégagée à partir des traitements sur des données, suite à des comparaisons par rapport à des standards, par rapport à des opérations précédentes ou similaires.

c. Critères de fiabilité de la preuve

Les critères qui doivent être satisfont pour considérer une preuve fiable :

- L'indépendance du fournisseur de la preuve : les preuves obtenues à partir des sources externes sont plus fiables par rapport à celles de l'organisme audité.
- Information ou preuve fournie par une personne qualifiée.
- Une preuve qui ne nécessite pas un jugement ou une interprétation considérable.
- Conservation de la validité de la preuve à travers le temps.

Les preuves dégagées à partir des données stockées dans des structures telles que les fichiers, peuvent ne pas être récupérées après une période de temps si les changements sur ces derniers ne sont pas contrôlés ou si les preuves ne sont pas sauvegardées.

d. Protection de la preuve

L'auditeur doit être conscient de l'importance des contrôles qui doivent être appliqués sur les preuves collectées.

Des contrôles doivent être appliqués sur ses preuves assurant la protection de :

- La confidentialité.
- La disponibilité.
- L'intégrité.

Exemples de moyens pour assurer ses propriétés pour les preuves électroniques :

- Le cryptage pour assurer la confidentialité.
- La sauvegarde pour assurer la disponibilité.
- La signature pour assurer l'intégrité.

e. Techniques de collecte de preuves

La collecte des preuves se fait en examinant:

- **Les politiques et procédures du SI :** Vérifier si les politiques et les procédures appropriées sont élaborés. Déterminer si le personnel comprend les procédures et les politiques implémentées. Vérifier si le personnel suit/respecte les procédures et politiques existantes au sein du SI. Vérifier que la révision régulière est réalisée pour ces documents.
- **Les structures organisationnelles du SI :** Vérifier si les structures organisationnelles (SO) sont mises en place et qu'ils appliquent de façon adéquate la séparation des responsabilités. Vérifier le niveau de contrôle assuré par ces SO.
- **La documentation du SI :** Cette documentation inclut au moins :
  - Les documents de spécification et de développement.
  - Les plans de test.
  - Les manuels d'utilisation des applications.
  - Les documents de sécurité (les plans de sécurité, l'analyse des risques, etc.).
- **Le personnel :** Faire des entretiens avec le personnel approprié. Préparation des questionnaires à l'avance.
- **Les processus:** Observation du déroulement des processus du SI.

f. La documentation d'audit

La documentation est un élément essentiel durant l'exécution des tâches d'audit. Les résultats obtenus, les activités et les tests doivent être documentés. Les documents peuvent être écrits sur papier ou sous un format électronique. Ils doivent être correctement nommés, datés,

détaillés (claires). Les documents ne doivent pas être : Manipuler par des personnes non autorisées (lecture, modification, détérioration). Diffuser à des entités internes ou externes à l'organisme audité. Les mesures de protection nécessaires doivent être appliquées. La documentation d'audit inclut :

- Le plan de travail.
- Une description ou un diagramme représentant l'environnement à auditer.
- Les étapes d'audit réalisées et les preuves collectées.
- Les résultats des tests. Les conclusions.
- Les recommandations.
- Les rapports élaborés suite à l'exécution des tâches d'audit.

La documentation doit être conservée pour une période de temps fixée par les lois et les réglementations en vigueur.

g. Les tests de conformité

Après avoir identifié les points de contrôles, des tests de conformité doivent être menés.
Un test de conformité vérifie que les contrôles dans l'environnement audité fonctionnent correctement. Il vérifie que les contrôles fonctionnent conformément aux procédures et politiques de l'entreprise.

Les tests incluent :

- L'examen des documents (schéma stratégique, politique de sécurité, etc.).
- La réalisation des questionnaires avec les responsables et quelques membres du personnel.
- L'observation des opérations.
- L'examen des biens.

Les opérations de test incluent :

- L'analyse des fichiers (log, de configuration, etc.).
- Application des procédures de test d'efficacité des contrôles mis en place (antivirus, détecteur d'intrusion, etc.).
- L'examen des installations, des systèmes fonctionnels ou en développement.
- L'examen des processus de développement.
- L'examen de la sécurité physique, logique associée au SI.
- L'examen de la sécurité des systèmes de communication.

L'auditeur utilise un ensemble d'outils et de techniques pour la réalisation des tests sur les composants du SI.

h. Test par sélection d'échantillons

L'échantillonnage est adopté lorsque des contraintes de temps et de coût empêchent une vérification complète de la population.

La population représente la totalité des biens à auditer. Un échantillon : un sous ensemble de cette population.

L'échantillonnage est utilisé pour déduire/inférer les caractéristiques d'une population en se basant sur les résultats d'examen d'un échantillon de la population. L'auditeur doit auditer l'échantillon en utilisant les procédures de test nécessaires et évaluer les résultats obtenus afin d'obtenir des preuves fiables et suffisantes.

Les techniques basées sur l'échantillonnage requièrent un jugement de la part de l'auditeur pour la définition des caractéristiques de la population pourtant ça peut représenter un risque car l'auditeur peut faire un faux jugement en se basant sur un échantillon.

i. Les ressources de l'auditeur

Tous les documents existants sont susceptibles d'être consultés, et doivent être accessibles :

- Comptes rendus de réunion. Procédures écrites.
- Documents depuis la conception initiale jusqu'au détail de l'exploitation, organigrammes du personnel et fiches de fonctions.
- Schémas des bases de données Documents des constructeurs et prestataires.
- Contrats Les rapports d'audit précédents.

La documentation est variable en quantité et qualité selon l'entreprise.

- L'auditeur utilise aussi l'outil informatique tel que :
- Des sondes connectées au réseau à auditer.
- Des logiciels de simulation, d'analyse de fichiers, d'analyse des performances, de détermination de taux d'occupation et de charge des réseaux, des outils de test de configuration des systèmes d'exploitation, etc.

L'auditeur peut aussi programmer des scripts et des applications pour l'automatisation de certaines tâches d'audit.

Observation : La manière selon laquelle le personnel exécute les tâches accordées.

Questionnaires : Vérifier le respect des procédures et politiques mis en place.

I.5.4. Phase 4 : Elaboration du rapport

Cette phase est consacrée à la rédaction du rapport final et sa distribution. Puis, il y aura une présentation et discussion.

a. La rédaction du rapport

Le rapport d'audit est un document de référence. L'objectif de l'audit et de son livrable (le rapport) est d'assister les responsables de l'organisme audité à améliorer les contrôles pour le SI mis en place. Il est nécessaire de définir à qui ce rapport est destiné et comment il sera diffusé. La rédaction du rapport prend du temps, donc elle ne doit pas être laissée à la fin de la mission. Il faut se baser sur des éléments de preuve pour justifier les insuffisances dégagées. Il faut porter des appréciations claires et non ambigües. Il faut faire des références à des référentiels et des standards.

b. Structure d'un rapport d'audit

Généralement, un rapport d'audit comporte :

- Une introduction : qui inclut les objectifs d'audit, son étendu, la durée d'audit, la date de début de l'audit, la démarche suivie.
- Une synthèse du rapport.
- Une synthèse des recommandations.
- Les observations détaillées.
- Les recommandations détaillées.
- Annexes.

Une page de garde doit être ajoutée au rapport, qui inclut :

- Le titre du rapport, le nom de l'organisme audité.
- La date la mention « document confidentiel » si nécessaire.

c. Les règles de rédaction

Il faut que le document soit clair et compréhensible par le personnel concerné. La taille du rapport dépend de la taille du SI audité et de l'ensemble des insuffisances dégagées. Il faut expliquer les termes et les concepts utilisés. Il faut faire des synthèses et des récapitulations.
Les recommandations doivent être classées en mesures à court terme, à moyen terme et à long terme. Il faut utiliser des règles de nommages et de numérotation qui aident à faire le lien entre l'insuffisance relevée et la recommandation associée (référence).

d. Distribution du rapport d'audit

Il faut considérer la confidentialité des données contenues au niveau du rapport d'audit lors de sa remise aux d audit personnes concernées. Le rapport est livré au premier responsable sur l'organisme audité. Il se charge de le distribuer aux personnes concernées par l'étude et l'implémentation des recommandations. Les moyens de distribution du rapport : De main en main ou par courrier (recommandé). En utilisant la messagerie électronique Il faut appliquer

les mécanismes nécessaires pour assurer la confidentialité et l'intégrité du message échangé. Si le document est hautement confidentiel, des contrôles détectés supplémentaires doivent être appliqués.

I.5.5. Phase 5 : Suivi des recommandations

C'est l'étape la plus importante. Durant laquelle l'auditeur s'assure que les recommandations sont implémentées. Puis il y a rédaction d'un rapport décrivant l'état d'implémentation des recommandations.

a. Les recommandations

Les recommandations peuvent avoir trois formes :

- **Pas de changement introduits pour les contrôles mis en place** : si l'auditeur juge que les contrôles mis en place sont efficaces et offrent un niveau de protection acceptable par rapport aux risques identifiés.
- **Amélioration des contrôles et réduction du risque** : ceci par la modification des contrôles mis en place ou par l'ajout de nouveaux.
- **Transfert du risque (assurance, externalisation)** : peut-être recommandé pour un groupement de composants du SI où le risque est élevé et l'implémentation des contrôles est irréalisable ou le coût est important.

b. Le suivi des recommandations (Following-up)

Un audit de sécurité est sans effet sur le « SI » si les recommandations ne sont pas implémentées. L'objectif du suivi est de vérifier si toutes les recommandations suggérées sont bien appliquées. Le suivi de l'implémentation des recommandations doit se faire en respectant les délais fixés au niveau du rapport d'audit (court, moyen et long terme). Un état (rapport de quelques pages) doit être élaboré par l'auditeur indiquant les recommandations qui ont été appliquées totalement ou partiellement ou celles qui ne sont pas traitées.

I.6. Les références d'audit

L'audit des systèmes d'information est un moyen de vérifier l'écart d'un système d'information par rapport à une référence donnée. Une mission d'audit s'appuie donc impérativement sur une référence. Dans ce domaine, il existe différentes normes sur lesquelles se basent les missions d'audit de sécurité des systèmes d'information. Nous vous présentons les plus utilisées.

I.6.1. La norme BS 7799

Le British Standard 7799 (BS 7799) est composé de deux guides, l'ISO/IEC 17799:2000 et le BS 7799-2 :2002. L'ISO/IEC 17799:2000 est un catalogue regroupant 36 objectifs de

contrôle, décomposés en 127 mesures de contrôle, et relatifs à 10 domaines (politique de sécurité, sécurité du personnel, contrôle des accès…).

Les objectifs de contrôle présentent un but à atteindre et ce qu'il faut entreprendre pour y parvenir. Ils sont ensuite décomposés en mesures de contrôle qui expliquent avec plus ou moins de détails les points à mettre en œuvre pour implémenter ces mesures. Le BS 7799-2:2002 présente un système de gestion de la sécurité de l'information (Information Security Management System - ISMS) en quatre étapes récurrentes (planifier, mettre en œuvre, vérifier, améliorer) se rapprochant des normes de qualité ISO 9001 et ISO 14001. L'étape de planification préconise d'employer l'ISO/IEC 17799:2000.

I.6.2. La norme ISO 13335

Cette norme qui est apparue en 1996 se présente aujourd'hui en quatre parties. Il S'agit notamment :

- ISO 13335-1 : Concepts et modèles pour la gestion de la sécurité des technologies de l'information et des communications (2004)
- ISO 13335-3 : Techniques pour la gestion de la sécurité IT (1998)
- ISO 13335-4 : Sélection de sauvegardes (2000)
- ISO 13335-5 : Guide pour la gestion de la sécurité du réseau (2001).

I.6.3. La norme ISO 15408

Apparue en 1996, la norme ISO 15408 (également baptisée « Common Criteria » ou «Critères Communs ») permet d'avoir une assurance de sécurité sur des critères précis pour un produit ou un système (matériel de sécurité, firewalls…). Elle se compose actuellement de 3 parties qui sont :

- ISO 15408-1 : Introduction et modèle général
- ISO 15408-2 : Exigences fonctionnelles de sécurité
- ISO 15408-3 : Exigences d'assurance de sécurité

I.6.4. La norme ISO 15446

L'ISO/IEC TR 15446:2004 fournit des conseils touchant à la construction de Profils de Protection (PPs) et des Cibles de Sécurité (CS ou ST en anglais pour Security Target) qui sont destinées à être de pair avec ISO/IEC 15408 ("les Critères Communs"). Elle propose également des suggestions sur la façon de développer chaque section d'un PP ou ST.

I.6.5. La norme ISO 17799

Cette norme a constitué le socle de notre mission d'audit. La norme ISO 17799 est un ensemble de recommandations pour la gestion de la sécurité de l'information et un référentiel

en matière de bonnes pratiques de sécurité. Elle émane de la norme britannique BS7799. La norme ISO 17799 couvre aussi bien les aspects techniques, organisationnels et physiques et peut être utilisée par n'importe quel organisme quelle que soit son activité et sa dimension. Les aspects qu'elle recouvre sont structurés suivant onze grandes thématiques. Les clauses constitutives de l'ISO/IEC 17799:2005 sont les suivantes, ordonnées tel qu'évoqué dans la norme :

- Politique de sécurité Organisation de la sécurité.
- Classification et contrôle des actifs.
- Sécurité des ressources humaines.
- Sécurité physique et environnementale.
- Gestion des communications et de l'exploitation.
- Contrôle d'accès Acquisition, développement et maintenance des systèmes d'information.
- Gestion des incidents de sécurité.
- Gestion de la continuité d'activité.
- Conformité.

I.6.6. La norme ISO 27001

La norme ISO 27001 représente la nouvelle famille de normes de sécurité informatique. Il s'agit d'une série de normes spécifiquement réservées par ISO pour des sujets de sécurité de l'information. La norme ISO 27001 est, alignée avec un certain nombre d'autres matières, y compris ISO 9000 pour la gestion de qualité et ISO 14000 pour la gestion environnementale.

II. Les systèmes d'exploitation pour mobile

La guerre des **mobiles** passe essentiellement par celui des systèmes d'exploitation. Sur ce secteur, deux grandes sociétés mènent la danse en termes de parts de marché : **Google** et **Apple** avec respectivement 81,5 % et 14,8 %. Le premier, père d'Android, voit son système déployé sur une belle ribambelle d'appareils venant de nombreuses marques. L'autre, avec iOS, est bien plus exclusif puisqu'il est réservé aux terminaux qu'il vend : les iPhone.

Android et iOS ne sont pas seuls pour autant. D'autres challengers sont dans la partie et ils ne manquent pas d'intérêt. C'est le cas de **Windows Phone 10**, le penchant mobile de Windows 10, ainsi que **BlackBerry 10**, la dernière tentative du constructeur mythique de téléphone pour professionnel, qui est néanmoins progressivement remplacé par **Android**.
Dans cette section, nous allons présenter quelques systèmes d'exploitation pour les systèmes mobiles les plus utilisés sur le marché.

II.1. Android OS

Android, prononcé à la française /ã.dʁɔ.id/ (androïde), en anglais /ˈænˌdɹɔɪd/, est un système d'exploitation mobile basé sur le noyau Linux et développé actuellement par Google. Le système a d'abord été conçu pour les smartphones et tablettes tactiles, puis s'est diversifié dans les objets connectés et ordinateurs comme les télévisions (Android TV), les voitures (Android Auto), les ordinateurs (Android-x86) et les smart Watch (Android Wear). Le système a été lancé en juin 2007 à la suite du rachat par Google en 2005 de la startup du même nom. En 2015, Android est le système d'exploitation le plus utilisé dans le monde avec plus de 80 % de parts de marché dans les Smartphones [6].

II.2. iOS

iOS, anciennement iPhone OS, est le système d'exploitation mobile développé par Apple pour plusieurs de ses appareils. Il est dérivé de OS X dont il partage les fondations (le noyau hybride XNU basé sur le micro-noyau Mach, les services Unix et Cocoa, etc.).

Ce système d'exploitation n'avait aucun nom officiel avant la publication du kit de développement iPhone (SDK) le 6 mars 2008. Jusqu'à cette date, Apple se contentait de mentionner que « l'iPhone tourne sous OS X », une référence ambiguë au système d'exploitation source d'iOS, OS X. Ce n'est qu'à cette occasion que Scott Forstall[2] présenta l'architecture interne du système d'exploitation, et dévoila alors le nom d'iPhone OS. Ce nom a été changé le 7 juin 2010 pour iOS.

La marque commerciale « IOS » était utilisée par Cisco depuis plus de dix ans pour son propre système d'exploitation, IOS, utilisé sur ses routeurs. Pour éviter toute poursuite judiciaire, Apple a acquis auprès de Cisco une licence d'exploitation de la marque « IOS » [7].

II.3. Windows Phone

Windows Phone est un système d'exploitation mobile développé par Microsoft pour succéder à Windows Mobile, sa précédente plateforme logicielle qui a été renommée pour l'occasion en Windows Phone Classique. Contrairement au système qu'il remplace, Windows Phone est d'abord principalement destiné au grand public plutôt qu'au marché des entreprises. Cependant depuis Windows Phone 8, Microsoft propose des fonctions avancées pour les entreprises en offrant, par exemple, un espace d'applications réservé aux entreprises. Il a été lancé le 21 octobre 2010 en Europe, à Singapour, en Australie et en Nouvelle-Zélande, le 8 novembre 2010 aux États-Unis et au Canada, puis le 24 novembre 2010 au Mexique [8].

[2]**Scott Forstall :** est le vice-président des logiciels iPhone d'Apple

II.4. BlackBerry

BlackBerry est une ligne de téléphones intelligents, créée et développée par Mike LAZARIDIS depuis 1999 puis rejoint par Jim Balsillie1, d'abord sous le nom de « **RIM Research In Motion** », puis du produit dénommé BlackBerry, utilisant le système d'exploitation propriétaire BlackBerry OS, puis à partir de janvier 2013 le passage sous le système d'exploitation BlackBerry 10 fait que l'entreprise, dans un but de clarté, a adopté le nom unique de BlackBerry pour l'entreprise et les produits.

À ce propos, le Système d'exploitation BlackBerry 10 est fondé sur le micro noyau **QNX**, logiciel de la société éponyme rachetée en 2010 par la société canadienne RIM [9].

III. Les applications mobiles

Dans cette section nous allons décrire les applications mobiles, leurs avantages ainsi que la tendance de téléchargement de ces applications.

III.1. Description

Une application mobile est un programme téléchargeable de façon gratuite ou payante exécutable à partir du système d'exploitation du téléphone. Toutefois, elle est une application mobile spécifique à un système d'exploitation mobile, développée avec le langage et les outils associés fournis par l'éditeur du système d'exploitation mobile, et installée directement sur le Smartphone.

Pour télécharger une telle application sur un téléphone portable mobile, il existe de différentes possibilités :

- Depuis un ordinateur en utilisant un câble de connexion ;
- A partir d'un service mobile ;
- Via une boutique logicielle accessible depuis un téléphone mobile (App Store d'Apple, Windows Market Place, Nokia OVI, Android Market, etc.).

III.2. Avantages

Les principaux avantages des applications mobiles sont :

- **Utilisation en mode connecté/déconnecté :** les jeux sont un bon exemple du cas d'utilisation de l'utilisation des applications mobiles en mode déconnecté.
- **Les effets « Waouh » :** il s'agit de la capacité d'une application mobile de surprendre son utilisateur par des effets visuels ou d'usages. Avec l'arrivée des téléphones équipés d'écrans tactiles et la révolution des usages, de nouveaux effets sont apparus. Parmi ces effets, nous pouvons parler de la réalité augmentée, la 3D, l'effet « Shaker » (secouer, vibration), l'effet « Tilt » (basculement en mode paysage), l'effet

du glissage, l'effet du pointer et celui du « Slideshow », etc.

- **Utilisation des fonctionnalités du téléphone :** Il est plus facile pour un développeur d'application mobile de faire appel aux fonctionnalités natives du téléphone mobile et aux données natives enregistrées sur l'appareil.

III.3. Tendance de téléchargement des applications mobiles

Tous ces avantages cités au-dessus expliquent la forte croissance du nombre de téléchargement des applications mobiles comme la montre le Tableau 4ci-dessous. Selon Gartner[3], en 2017, les applications mobiles devraient être téléchargées plus de 268 milliards de fois, générant un chiffre d'affaires de près de 77 milliards de dollars et faisant des applications l'un des vecteurs de communication les plus populaires au monde. Chaque utilisateur mobile devrait, d'ici là, envoyer des informations personnalisées à plus de 100 applications et services chaque jour. Dans une industrie plus que jamais dominée par le modèle "Freemium", 94,5% de ces téléchargements concerneront des applications gratuites [10].

	2012	2013	2014	2015	2016	2017
Téléchargements gratuits	57,331	82,876	127,704	167,054	211,313	253,914
Primes de Téléchargements	6,654	9,186	11,105	12,574	13,488	14,778
Total des Téléchargements	63,985	102,062	138,809	179,628	224,801	268,692
Téléchargements gratuits %	89,6	91,0	92,0	93,0	94,0	94.5

Tableau 4. Téléchargements App Store Mobile (Téléchargements en millions)

IV. Etude de l'existant

Le principal objectif de notre projet est de créer un service d'audit qui permet de remédier aux problèmes posés par les outils d'audit (les scanners de vulnérabilités).

Actuellement, les solutions existantes tournes sur les plateformes Windows, Linux ou Mac OS.

Dans cette partie, nous allons présenter les plus importants et préciser les insuffisances de chacun pour avoir une intuition sur notre solution finale qui sera une application mobile selon le besoin de l'établissement KMCT.

[3] **Gartner :** http://www.gartner.com/technology/home.jsp

IV.1. ISS

ISS Internet Scanner ™ est la solution préférée du secteur de la sécurité réseau pour l'analyse de la vulnérabilité du réseau et aide à la décision. Internet Scanner effectue prévue et sélective des sondes de services de communication de votre réseau, les systèmes d'exploitation, les applications clés, et les routeurs à la recherche de ces vulnérabilités le plus souvent utilisé par des menaces sans scrupule de sonder, d'enquêter, et d'attaquer votre réseau. Internet Scanner analyse ensuite vos conditions de vulnérabilité et fournit une série de mesures correctives, l'analyse des tendances, avec sursis, des rapports de configuration et des ensembles de données [11].

Plate-forme : Windows

Les insuffisances d'Internet Scanner sont :

- Les vulnérabilités mentionnées dans le résultat de test d'audit par cet outil ne renvoient pas à des liens **CVE**« Common Vulnerabilities and Exposures » qui est une liste de noms standardisés pour les vulnérabilités publiquement connues et les révélations relatives à la sécurité. (Lorsque une faille de vulnérabilité identifiée associée à un identifiant **CVE**, nous allons obtenir davantage d'informations sur le Web).
- Résultats stockés dans la base non chiffrés par défaut.
- Pauvreté des détails du rapport technique d'audit.
- Ne permet pas une correction automatique des vulnérabilités.
- Ne permet pas de faire des tests personnalisés.
- L'envoi des rapports n'est pas sécurisé.

IV.2. Saint

SAINT (Security Administrator's Integrated Network Tool) est un outil d'évaluation de la sécurité basée sur SATAN. Les caractéristiques comprennent la numérisation à travers un pare-feu, des contrôles de sécurité mis à jour à partir des bulletins CERT & CIAC, 4 niveaux de gravité (rouge, jaune, brun, et vert) et une fonction d'interface au format HTML enrichi [11].

Plate-forme : Windows / Linux 2.x Solaris / Mac OS X

Les insuffisances de SAINT sont :

- Ne permet pas une correction automatique de vulnérabilité sélectionnée
- Ne détecte pas autant de services présentant des failles
- N'indique pas le port et le service présentant l'anomalie

IV.3. Retina

Retina Network Security Scanner est un scanner de vulnérabilité avancée. Il peut analyser chaque machine de votre réseau y compris une variété de plates-formes d'exploitation du système, dispositifs de réseau, bases de données et des tiers ou des applications personnalisées [11].

Plate-forme : Windows

Les insuffisances de Retina sont :

- Il n'y a pas de scan automatique des nouveaux utilisateurs
- Pas de détection d'un changement de machine (sans changer l'adresse IP)
- Gestion des correctifs via un module en option
- Manque de finesse du classement des vulnérabilités
- L'envoi des rapports n'est pas sécurisé

IV.4. Nessus, NewT

L'outil **Nessus**, et sa version Windows appelée NewT est un outil open source. Basé sur une architecture client/serveur, Nessus permet aux utilisateurs d'exécuter la console d'administration, qui exécute des analyses de vulnérabilité et tient des bases de données sur une machine autre que le serveur [11].

Plate-forme : Windows / Linux

Les insuffisances de l'outil Nessus sont :

- L'interface utilisateur, bien que graphique, est peu synthétique, et il n'y a pas d'aide en ligne, ni d'assistant logiciel.
- Manque de suivi en temps réel du déroulement de l'audit
- Pas de stockage sécurisé des résultats
- Il n'y a pas non plus de mode découverte : le premier scan va détecter la configuration de chaque machine du parc, et ajouter cet inventaire dans la base de données.

V. Synthèse pour le choix de la solution

L'étude de l'existant et l'analyse critique nous a permis d'avoir une vision claire sur les objectifs et les erreurs à éviter au cours du développement du service d'audit de sécurité des équipements réseaux mobile que nous allons réaliser.

Cette application sera basée sur l'idée de satisfaire une majorité d'utilisateur. Nous devons répondre aux besoins suivants :

- Evaluer le niveau de sécurité des équipements réseau afin de déterminer ses vulnérabilités et proposer les recommandations nécessaires pour la protection.
- Effectuer une tâche d'audit d'un équipement réseau selon des fichiers de configurations personnalisés ou des questionnaires préconçus par l'administrateur du réseau.
- Détecter des failles de vulnérabilités qui n'en sont pas (faux positifs) ou l'inverse.
- Corriger les problèmes de sécurité qui ne pouvant pas être résolus automatiquement.
- Evaluer les systèmes d'information à tout moment et de n'importe où.

Conclusion

Dans ce chapitre nous avons introduit la notion d'audit de sécurité ainsi son cycle de vie. Nous avons décrit les différentes phases de l'audit de sécurité en présentant le livrable de chaque phase. De plus l'étude et l'analyse de l'existant ont permis de mieux cerner les avantages et les limites des produits existants afin d'apprendre à bien se servir de ces premiers et éviter au maximum certaines erreurs lors de la mise en œuvre d'un produit similaire qui correspond plus aux besoins.

Le troisième chapitre sera consacré à la spécification de la méthodologie et la démarche que nous allons suivre tout au long de ce rapport.

Chapitre III. Démarche et méthodologie du projet

Introduction

Après une étude bien détaillée de l'existant et une comparaison entre les solutions d'audit de sécurité qui existent sur le marché, le présent chapitre sera consacré à la présentation de la démarche du projet ; description du sujet, précision du planning et choix de la méthodologie qui sera adaptée tout au long de ce rapport.

I. Description du sujet

Le projet consiste à réaliser un service d'audit de sécurité des équipements réseaux pour les clients mobiles. Ce service permet aux auditeurs d'évaluer le niveau de sécurité des équipements réseau afin de déterminer ses vulnérabilités et proposer les recommandations nécessaires pour la protection. Le service doit être capable de collecter un grand nombre d'informations pertinentes et détaillées sur le composant pour éviter d'avoir une fausse évaluation : détecter des failles de vulnérabilités qui n'en sont pas (faux positifs) ou l'inverse.

Ce service permet aussi d'évaluer les systèmes d'information à tout moment et de n'importe où ce qui permet de corriger les problèmes de sécurité qui ne pouvant pas être résolus automatiquement.

Les données échangées devront être chiffrées afin de garantir la confidentialité de la communication.

II. Planification du projet

La planification d'un projet est un outil incontournable pour le management du projet. Elle permet de :

- Définir les travaux à réaliser.
- Fixer des objectifs.
- Coordonner les actions.
- Maitriser les moyens.
- Diminuer les risques.
- Suivre les actions encours.
- Rendre compte de l'état d'avancement du projet.

Le Tableau 5 ci-dessous, définie la liste des tâches planifiées tout au long du projet.

Name	Begin date	End date	Duration
Découverte de l'environnement	3/21/16	4/1/16	10
Discussion sur le projet	4/4/16	4/4/16	1
Recherche et collecte d'informations	4/4/16	4/11/16	6
Etude de l'existant	4/7/16	4/11/16	3
Choix de la méthodologie	4/7/16	4/11/16	3
Cahier des charges	4/12/16	4/26/16	11
Génération des cas d'utilisation	4/13/16	4/18/16	4
Conception graphique	4/19/16	4/27/16	7
Conception technique	4/25/16	5/27/16	25
Réalisation	5/2/16	6/24/16	40
Tests et Validation	5/4/16	7/5/16	45
Rédaction du rapport	4/1/16	7/7/16	70

Tableau 5. Planifications des tâches du projet

Le diagramme de GANTT, présenté dans la Figure 3ci-dessous, permet de planifier le projet et de rendre plus simple le suivi de son avancement.

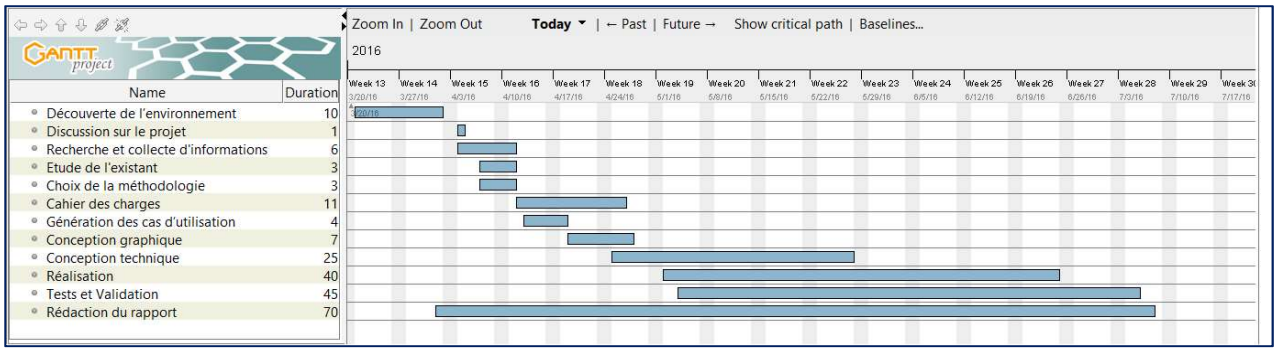

Figure 3. Diagramme de GANTT

III. Méthodologie du projet

Tout développement de logiciel nécessite de reposer sur une méthode. Il existe différentes méthodes de développement mais aussi divers types de cycles de développement entrant dans la réalisation d'un logiciel. Ces cycles prendront en compte toutes les étapes de la conception d'un logiciel.

IV. Les méthodologies existantes

L'objectif du chef de projet est de pouvoir mener son projet à terme en respectant les délais et le budget alloué. Pour atteindre cet objectif, il doit prendre en compte les 3C (voir Figure 4 ci-dessous) qui sont les trois contraintes que constitue le projet.

Figure 4. Les contraintes 3C

Pour atteindre son objectif, le chef de projet peut avoir recours à plusieurs méthodes de gestion de projet. Il existe deux grandes familles de méthode de développement ; nous citons les méthodes classiques ou traditionnelles et les méthodes agiles.

IV.1. Les méthodes classiques

Une méthode classique est une méthode de développement figée. Le client expose sa problématique au début et le développeur s'en charge entièrement de lui livrer à la fin de la période programmée au projet la solution complète. Aucun retour vers client ne se fait tout au long de la période du développement de la solution.

Les cycles classiques de développement d'applications sont : les modèles en cascade, les modèles en V, le prototypage et les modèles en spirale.

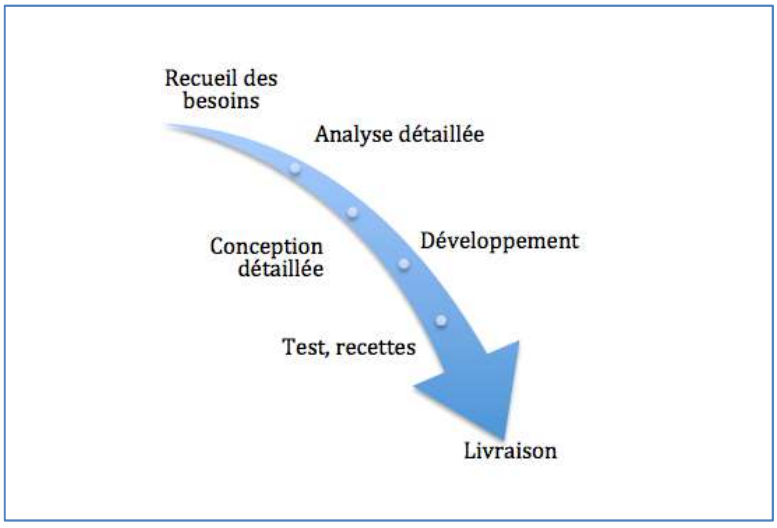

Figure 5. Cycle de vie des méthodes classiques

Dans un cycle « en cascade » (comme indiqué dans la Figure 5 ci-dessus) **les risques sont détectés tardivement** puisqu'il faut attendre la fin du développement pour effectuer la phase de test. Plus le projet avance, plus l'impact des risques augmente : **il sera toujours plus**

difficile et coûteux de revenir en arrière lorsqu'on découvre une anomalie tardivement.

IV.2. Les méthodes agiles

Les méthodes de développement dites « **méthodes agiles** » (en anglais Agile Modeling, noté AG) visent à réduire le cycle de vie du logiciel en développant une version minimale, puis en intégrant les fonctionnalités par un processus itératif basé sur une écoute client et des tests tout au long du cycle de développement.

Les méthodes agiles prônent quatre valeurs fondamentales (entre parenthèse, les citations du manifeste) :

- **L'équipe** (« Personnes et interaction plutôt que processus et outils »).
- **L'application** (« Logiciel fonctionnel plutôt que documentation complète »).
- **La collaboration** (« Collaboration avec le client plutôt que négociation de contrat »).
- **L'acceptation du changement** (« Réagir au changement plutôt que suivre un plan »).

Parmi les méthodes agiles, nous distinguons 2TUP, RUP[4] et XP[5].

IV.2.1. La méthode 2TUP

- **Définition :** préconise un cycle de vie en Y qui dissocie la résolution des questions fonctionnelles et techniques. Le cycle de vie de 2TUP s'apparente à un cycle de développement en cascade mais introduit une forme itérative interne à certaines tâches.

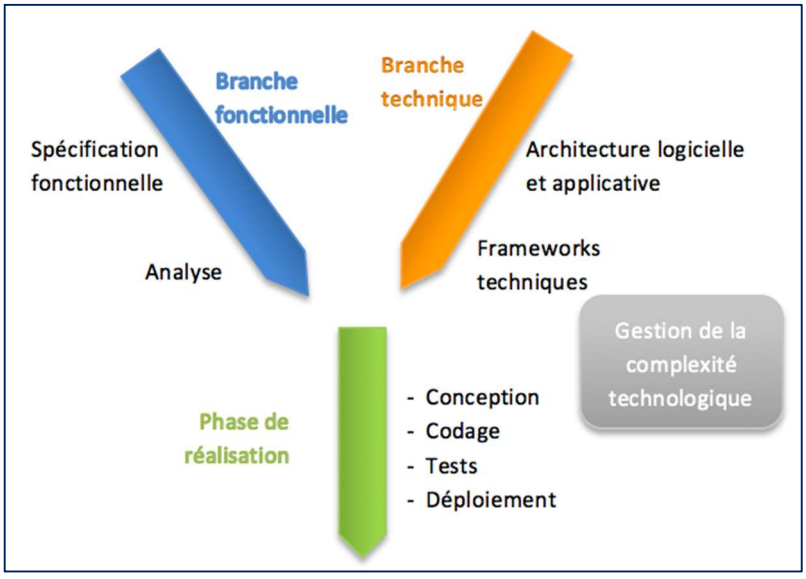

Figure 6. Cycle de vie en Y

[4] **RUP:** Rational Unified Process
[5] **XP:** eXtreme Programming

- **Avantages :** cette méthode permet essentiellement la séparation entre les besoins fonctionnels et techniques (voir Figure 6 ci-dessus).

IV.2.2. La méthode RUP

- **Définition :** se caractérise par une approche globale nommée « Vue 4+1 ». Les 5 composants de cette vue sont : la vue des Cas d'utilisation, la vue Logique, la vue d'Implémentation, la vue du Processus, la vue du Déploiement (voir Figure 7ci-dessous).

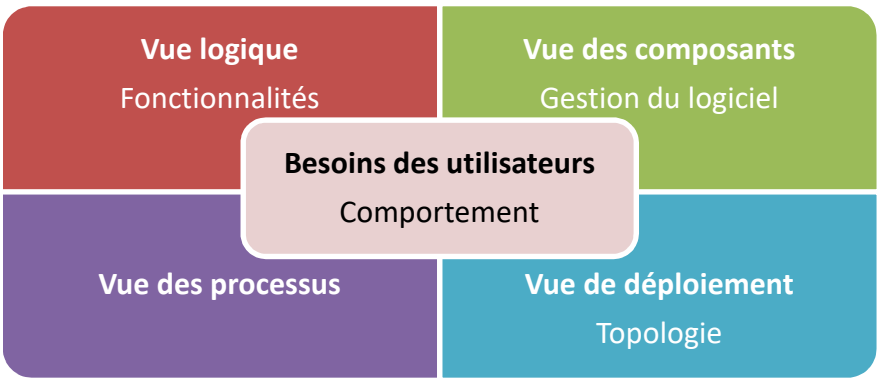

Figure 7. La vue 4+1 de RUP

- **Avantages** : RUP offre un processus guide formel et adaptable comme guide d'activité.

IV.2.3. La méthode XP

- **Définition** : est très axée sur la partie construction de l'application. Une de ses originalités réside dans l'approche de planification qui se matérialise sous la forme d'un jeu intitulé « Planning Game » et qui implique simultanément les utilisateurs et les développeurs. La Figure 8 ci-dessous illustre le cycle de vie de la méthode XP.

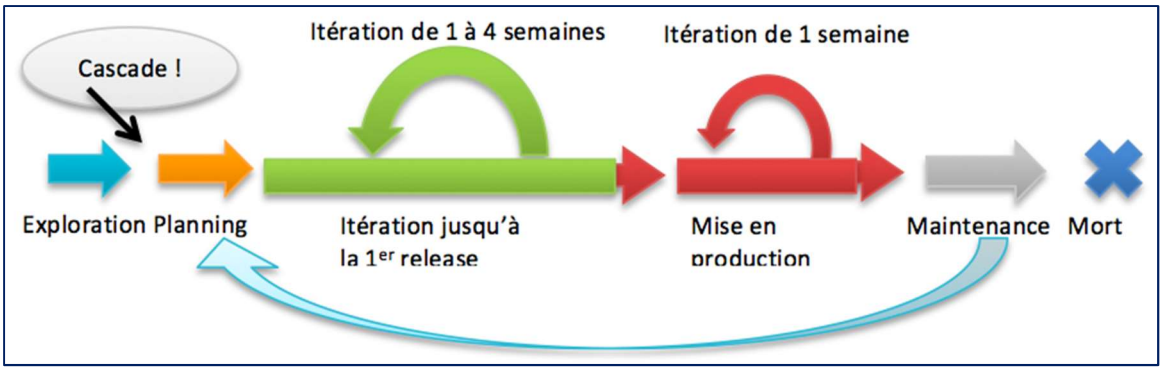

Figure 8. Cycle de vie de XP

- **Avantages** : On notera des techniques particulières liées à la production du code comme la programmation en binôme, l'appropriation collective du code, la Ré factorisation et l'Intégration continue.

IV.3. Comparaison entre les deux méthodes de travail

Ci-dessous un tableau récapitulatif des différences entre la méthode traditionnelle et la méthode agile [12].

Thème	Approche traditionnelle	Approche agile
Cycle de vie	En cascade ou en V, sans rétroaction possible, phases séquentielles.	Itératif et incrémental.
Planification	Prédictive, caractérisée par des plans plus ou moins détaillés sur la base d'un périmètre et d'exigences définies et stables au début du projet.	Adaptative avec plusieurs niveaux de planification (macro- et micro planification) avec ajustements si nécessaires au fil de l'eau en fonction des changements survenus.
Documentation	Produite en quantité importante comme support de communication, de validation et de contractualisation.	Réduite au strict nécessaire au profit d'incréments fonctionnels opérationnels pour obtenir le feedback du client.
Equipe	Une équipe avec des ressources spécialisées, dirigées par un chef de projet.	Une équipe responsabilisée où l'interactive et la communication sont privilégiées, soutenue par le chef de projet.
Qualité	Contrôle qualité à la fin du cycle de développement. Le client découvre le produit fini.	Un contrôle qualité précoce et permanent, au niveau du produit et du processus. Le client visualise les résultats tôt et fréquemment.
Changement	Résistance voire opposition au changement. Processus lourds de gestion des changements acceptés.	Accueil favorable au changement inéluctable, intégré dans le processus.
Suivi de l'avancement	Mesure de la conformité aux plans initiaux. Analyse des écarts.	Un seul indicateur d'avancement : le nombre de fonctionnalités implémentées et le travail restant à faire.
Gestion des risques	Processus distinct, rigoureux, de gestion des risques.	Gestion des risques intégrée dans le processus global, avec responsabilisation de chacun dans l'identification et la résolution des risques. Pilotage par les risques.

Mesure de succès	Respect des engagements initiaux en termes de coût, de budget et de niveau de qualité.	Satisfaction client par la livraison de valeur ajoutée.

Tableau 6. Comparaison entre méthodes classiques et méthodes agiles

Les méthodes agiles seront plus utilisées pour les gros projets car elles offrent une meilleure adaptabilité, visibilité et gestion des risques. Elles pourraient tout aussi bien être utilisées pour les projets où il n'y pas de documentations détaillées, le client peut alors voir l'évolution du projet et l'adapter selon ses besoins.

En revanche, les **méthodes classiques** seront plus utilisées si nous avons une idée très précise du projet avec un cahier des charges et planning très détaillé où nous avons anticipé tous les risques possibles.

Le chef de projet se retrouve souvent seul face à lui-même lors de la prise de décision, de gérer les retours client, devant l'incertitude afin de satisfaire le client tout en respectant les 3C. On pourrait alors l'assimiler à un art martial car il doit avoir une grande maitrise de soi et de l'environnement de chaque projet auquel il est amené à gérer.

V. Choix de la méthodologie

Devant le nombre de méthodes disponibles, le choix parmi elles devient difficile, beaucoup de questions peuvent se poser à un chef de projet lors d'un démarrage de projet :

- Comment vais-je organiser les équipes de développement ?
- Quelles tâches attribuer à qui ?
- Quel temps faudrait-il pour livrer le produit ?
- Comment faire participer le client au développement afin de capter les besoins de celui-ci ?
- Comment éviter des dérives et de mauvaises estimations qui vont allonger les coûts et le temps de développement ?
- Comment vais-je procéder pour que le produit soit évolutif et facilement maintenable ?

Nous pouvons citer à ce propos les méthodes de développement objet suivantes : 2TUP, RUP, XP, AUP[6] et Open UP.

C'est ainsi que notre choix s'est orientée vers la méthode agile en particulier la méthode 2TUP, du fait de son approche nouvelle et originale est essentiellement de la séparation entre

[6] **AUP :** Agile Unified Process

les besoins techniques et fonctionnels.

Notre projet est basé sur un processus de développement bien défini qui va de la détermination des besoins fonctionnels attendus du système jusqu'à la conception et le codage final. Ce processus se base lui-même sur le Processus Unifié (Unified Process) qui est devenu un standard général réunissant les meilleures pratiques de développement.

Cette méthode ne se base aucunement sur un processus linéaire mais bien, sur un développement itératif et incrémental.

Nous allons d'abord définir les différents concepts et étapes constituant la méthode 2TUP.

V.1. Définition de 2TUP

2TUP (Two Track Unified Process) implémente le processus unifié, méthode de développement qui considère le cycle de vie d'un logiciel sous-forme incrémentale et itérative afin de s'adapter aux changements continuels dans l'organisation du système d'information à représenter. En ce sens, il renforce le contrôle sur les capacités d'évolution et de correction de tels systèmes.

V.2. Description de la méthode 2TUP

La méthode 2TUP sépare initialement les aspects techniques des aspects fonctionnels avant de les regrouper dans la phase de réalisation. La figure ci-dessous (Figure 9) détaille les différentes phases du développement.

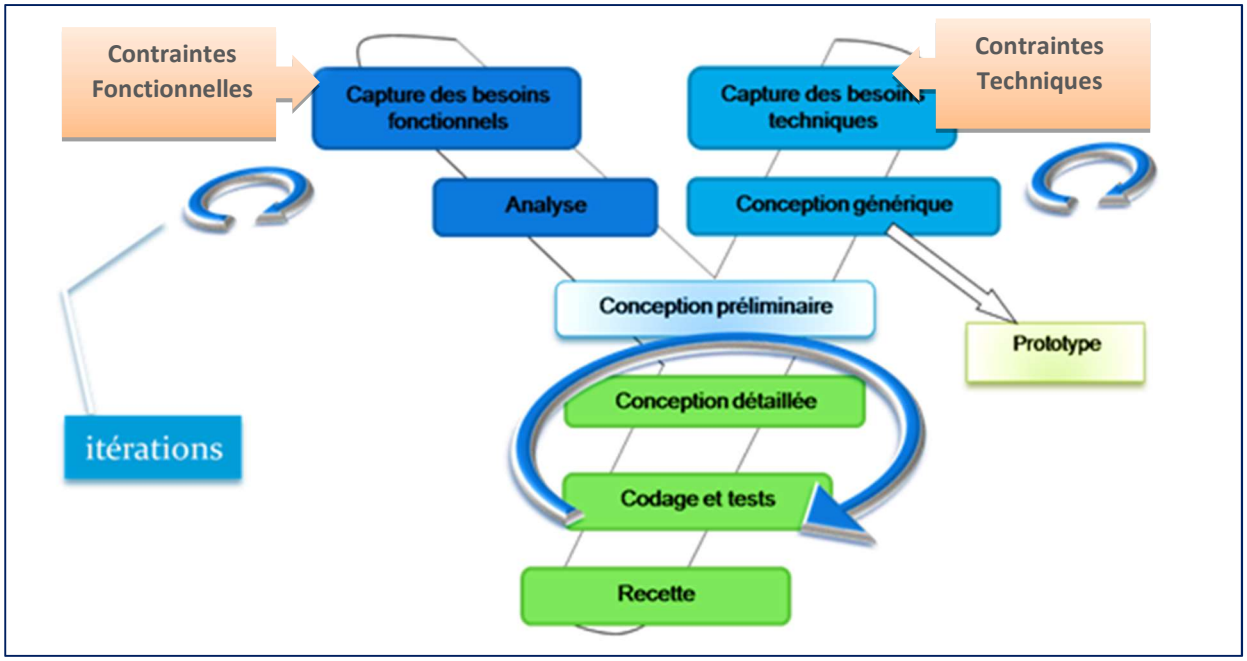

Figure 9. Processus de développement en Y

La branche de gauche (branche fonctionnelle) contient deux phases :

- **Capture des besoins fonctionnels**, qui produit le modèle des besoins focalisés sur le métier des utilisateurs. Elle qualifie, au plus tôt le risque de produire un système inadapté aux utilisateurs
- **L'analyse**, qui consiste à étudier précisément la spécification fonctionnelle de manière à obtenir une idée de ce que va réaliser le système en terme de métier.

La branche de droite (branche technique) intègre deux phases :

- **La capture des besoins techniques**, qui recense toutes les contraintes sur les choix de dimensionnant et la conception du système. Il s'agit de la spécification des outils (logiciels), de la structure des matériels à exploiter ainsi que la prise en compte des contraintes d'intégration avec l'existant (pré requis d'architecture technique).
- **La conception générique**, qui définit ensuite les composants nécessaires à la construction de l'architecture technique. Cette conception est complètement indépendante des aspects fonctionnels spécifiés dans la branche gauche. Elle a pour objectif de d'uniformiser et de réutiliser les mêmes mécanismes pour tout un système. L'architecture technique construit le squelette du système, son importance est telle qu'il est conseillé de réaliser un prototype de manière à valider les principes par le codage et les tests.

La branche du milieu se compose de quatre phases :

- **La conception préliminaire**, qui consiste à appliquer les concepts liés aux fonctionnalités du système et à intégrer les composants techniques au système. Il s'agit d'intégrer les fonctions métiers et applicatives dans l'architecture technique définie dans la phase de conception générique.
  - **La conception détaillée**, qui étudie ensuite comment réaliser chaque composant.
  - **L'étape de codage**, qui produit ses composants et tests au fur et à mesure les unités de code réalisées.
  - **L'étape de recette**, qui consiste enfin à valider les fonctionnalités du système développé.

Conclusion

Dans ce chapitre, nous avons évoqué la démarche et la méthodologie 2TUP que nous allons suivre tout au long de ce rapport afin de bien aboutir aux objectif préalablement prédéfinis. Le chapitre qui suit va présenter l'étude préliminaire et la capture initiale des besoins soit un cahier de charges délivré et les spécifications des besoins.

Chapitre IV. Spécification des besoins

Introduction

Au niveau de ce chapitre, nous allons entamer les premières étapes de la méthode 2TUP, il s'agit de l'étude préliminaire suivie par la capture des besoins fonctionnels et par la suite les besoins techniques.

I. Etude préliminaire

L'étude préliminaire (ou Pré étude) est la toute première étape du processus 2TUP. Elle consiste à effectuer un premier repérage des besoins fonctionnels et opérationnels, en utilisant principalement du texte, ou bien des diagrammes très simples. Elle prépare les activités plus formelles de capture des besoins fonctionnels et de capture techniques.

I.1. Présentation du projet à réaliser

C'est un scanner de vulnérabilité des équipements réseaux mobile pour Android destiné à toutes les directions du KMCT permettant à chacune l'accès à différents besoins. Le projet doit permettre essentiellement la gestion de l'opération d'audit, la gestion des équipements réseaux, la gestion des vulnérabilités, le suivi des recommandations, l'édition des questionnaires, etc.

I.2. Recueil des besoins fonctionnels

Pour le recueil des besoins, nous avons effectué plusieurs recherches pour identifier au mieux les besoins de l'application, et ceci afin de répondre aux attentes des utilisateurs. Nous sommes allés chercher les informations au sein de la direction « Computer Networks Technology ». Cette phase correspond à une recherche sur le terrain pour bien définir le cadre de notre système.

I.3. Identification des acteurs

Les acteurs du système identifiés dans un premier temps sont :

- **Le client :** C'est un acteur principal et déclencheur de l'application qui possède un espace d'authentification (un login et un mot de passe). Il représente la partie qui veut auditer les équipements de son réseau. Le client peut être un client simple ou un auditeur dans une entreprise.
- **L'administrateur :** il a pour rôle de gérer : les comptes des utilisateurs, les ressources matérielles, les tâches d'audit, les vulnérabilités, les questionnaires, les recommandations et afficher l'historique des vulnérabilités. Il accède par l'intermédiaire d'un login et un mot de passe.

I.4. Modélisation du contexte

La modélisation du contexte consiste à déterminer les fonctionnalités qu'apporte le système pour chaque utilisateur. Les tableaux suivants donnent les différents services que notre système offre à chaque acteur qui y participe.

Utilisateur	Description des besoins fonctionnels
Client	Le système offre à un client la possibilité de : ▪ S'authentifier ▪ Se déconnecter ▪ Gérer son profil ▪ Auditer un équipement ▪ Consulter un rapport d'audit ▪ Afficher un historique d'audit

Le système en permettant à l'administrateur de profiter des fonctionnalités offertes à un client, il lui permet également de bénéficier des droits supplémentaires.

Utilisateur	Description des besoins fonctionnels
Administrateur	Le système offre à un administrateur la possibilité de : ▪ Gérer les équipements ▪ Gérer les tâches d'audit ▪ Gérer les utilisateurs ▪ Gérer les vulnérabilités ▪ Gérer les questionnaires ▪ Gérer les recommandations ▪ Consulter l'historique des utilisateurs

Après ce travail de répartition, nous allons maintenant par le biais de la capture des besoins fonctionnels de formaliser de plus en plus notre système.

II. Capture des besoins fonctionnels

La capture des besoins fonctionnels consiste à identifier les différents cas d'utilisations par ses acteurs afin de les décrire, les organiser et de relever les classes candidates du modèle d'analyse.

II.1. Identification des cas d'utilisation

L'identification des cas d'utilisation une première fois, nous donnons un aperçu des fonctionnalités futures que doit implémenter le système. Cependant, il nous faut plusieurs itérations pour ainsi arriver à constituer des cas d'utilisation complets. D'autres cas d'utilisation vont apparaître au fur à mesure de la description de ceux-là, et l'avancement dans le « recueil des besoins fonctionnels ».

Pour constituer les cas d'utilisation, il faut considérer l'intention fonctionnelle de l'acteur par rapport au système dans le cadre de l'émission ou de la réception de chaque message. En regroupant les intentions fonctionnelles en unités cohérentes, on obtient les cas d'utilisations.

Ci-dessous une description détaillée de chaque cas d'utilisation identifié :

Cas d'utilisation	S'authentifier
Acteurs	Client / Administrateur
Messages émis/reçus par les acteurs	Emis : login + mot de passe Reçus : Accès autorisé ou pas

Cas d'utilisation	Auditer un équipement
Acteurs	Client / Administrateur
Messages émis/reçus par les acteurs	Emis : se connecter, répondre au questionnaire / télécharger un fichier de configuration Reçus : rapport d'audit, ne peut pas se connecter
Description	L'utilisateur client doit en premier lieu répondre au questionnaire ou télécharger le fichier de configuration afin de collecter des informations sur cet équipement, il peut choisir une tâche audit pour avoir un résultat plus précis, puis paramétrer le format du rapport et enfin lancer l'audit. Le résultat est le rapport d'audit qui contient : - La liste exhaustive des vulnérabilités recensées par le service sur l'équipement analysé. - La liste des recommandations permettant de supprimer les vulnérabilités trouvées.

Cas d'utilisation	Consulter un rapport d'audit
Acteurs	Client / Administrateur
Messages émis/reçus par les acteurs	Emis : se connecter Reçus : rapport d'audit, ne peut pas se connecter

Cas d'utilisation	Afficher un historique d'audit
Acteurs	Client / Administrateur
Messages émis/reçus par les acteurs	Emis : saisie, consulte Reçus : historique des audits réalisés auparavant

Cas d'utilisation	Gérer les équipements
Acteurs	Administrateur
Messages émis/reçus par les acteurs	Emis : se connecter, créer / modifier / supprimer Reçus : liste des équipements
Description	Chaque équipement est caractérisé par son type (serveur, routeur, firewall, …) et son emplacement dans le réseau, afin de bien simuler l'équipement réel que le client souhaite auditer.

Cas d'utilisation	Gérer les utilisateurs
Acteurs	Administrateur
Messages émis/reçus par les acteurs	Emis : se connecter, ajouter / modifier / supprimer Reçus : liste des utilisateurs

Cas d'utilisation	Gérer les tâches d'audit
Acteurs	Administrateur
Messages émis/reçus par les acteurs	Emis : se connecter, ajouter, supprimer Reçus : liste des tâches
Description	Les tâches d'audit sont des tâches que le client peut utiliser pour auditer un équipement, ils représentent des types de scan et des commandes qui seront exécutés soit par le service ou par un outil externe (exemple Nmap).

Cas d'utilisation	Gérer les vulnérabilités
Acteurs	Administrateur
Messages émis/reçus par les acteurs	Emis : se connecter, ajouter / modifier / supprimer Reçus : liste des vulnérabilités
Description	Stocker dans la base de données une liste la plus exhaustive possible des vulnérabilités d'un équipement réseau (serveur, firewall, routeur, etc.). Cette liste des vulnérabilités est dégagée à partir des standards de sécurité.

Cas d'utilisation	Gérer les questionnaires
Acteurs	Administrateur
Messages émis/reçus par les acteurs	Emis : se connecter, ajouter / modifier / supprimer Reçus : questionnaire
Description	Le questionnaire permet d'automatiser la collecte de données sur l'équipement que l'utilisateur souhaite auditer. Il contient des questions qui sont dégagées à partir des standards de sécurité.

Cas d'utilisation	Gérer les recommandations
Acteurs	Administrateur
Messages émis/reçus par les acteurs	Emis : se connecter, ajouter / modifier / supprimer Reçus : liste des recommandations
Description	Une liste des recommandations est présentée par le service à la fin d'un audit qui est lancé par un client afin de supprimer les vulnérabilités trouvées.

Cas d'utilisation	Consulter l'historique des utilisateurs
Acteurs	Administrateur
Messages émis/reçus par les acteurs	Emis : se connecter, consulter, Reçus : historiques
Description	Toute action faite par un utilisateur sera stockée dans une table log dans la base de données et seul l'administrateur de l'application peut y a accéder. Cela permet d'identifier facilement les erreurs, s'il y en aura.

II.2. Diagramme des cas d'utilisation

La Figure 10 représentée ci-dessous, illustre le diagramme des cas d'utilisation pour donner une vision globale du comportement fonctionnel d'un système logiciel.

Dans un diagramme de cas d'utilisation, les utilisateurs sont appelés « acteurs », ils interagissent avec les « cas d'utilisation ». Un acteur et un cas d'utilisation sont mis en relation par une association représentée par une ligne.

Ce diagramme est composé de deux acteurs principaux, chacun regroupe certaines fonctionnalités. Ces fonctionnalités ont été détaillées dans le tableau présenté dans le paragraphe ci-dessus.

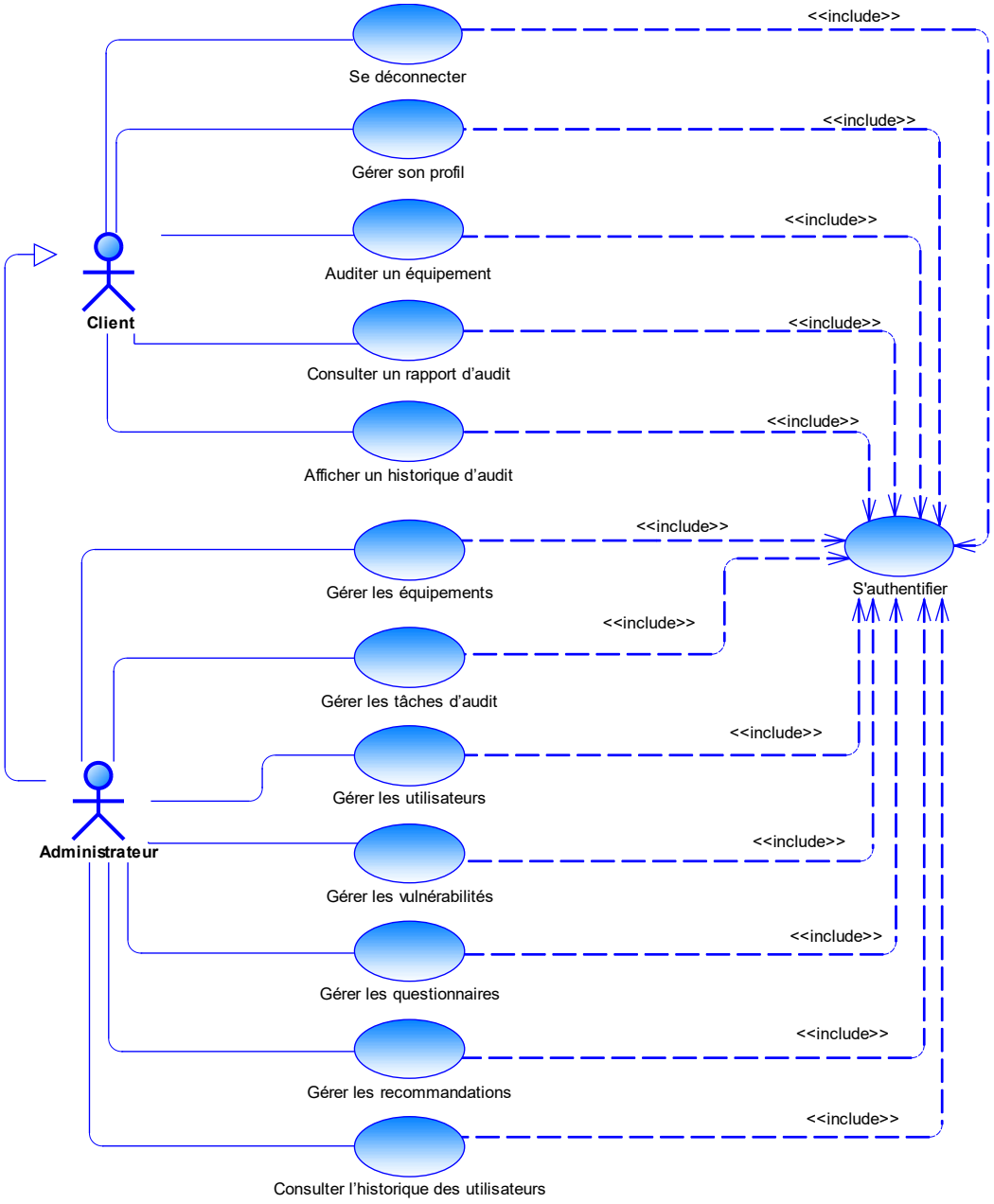

Figure 10. Diagramme des cas d'utilisation

III. Capture des besoins techniques

Dans cette partie, nous allons détailler les besoins techniques de notre système.

III.1. JEE

Java Enterprise Edition, ou Java EE (anciennement J2EE), est une spécification pour la technique Java de Sun plus particulièrement destinée aux applications d'entreprise. Ces applications sont considérées dans une approche multi-niveaux. Dans ce but, toute implémentation de cette spécification contient un ensemble d'extensions au Framework Java standard (JSE, Java Standard Edition) afin de faciliter la création d'applications réparties [13].

III.2. DAO

Un objet d'accès aux données (en Anglais Data Access Object ou DAO) est un patron de conception utilisé dans les architectures logicielles objet.

III.3. Bouncy Castle

Bouncy Castle est une bibliothèque de cryptographie libre et open-source. Elle s'apparente à la librairie **C openssl** qui est conforme aux différents standards en vigueur.

IV. Choix de la plateforme

Cette section est consacrée pour le choix de plateforme Android que nous avons utilisé.

IV.1. Android SDK

Le kit de développement (SDK) d'Android est un ensemble complet d'outils de développement. Il inclut un débogueur, des bibliothèques logicielles, un émulateur basé sur QEMU, de la documentation, des exemples de code et des tutoriaux.

Les plateformes de développement prises en charge par ce kit sont les distributions sous Noyau Linux, Mac OS X 10.5.8 ou plus, Windows XP ou version ultérieure.

L'IDE officiellement supporté était Eclipse combiné au plugin d'outils de développement d'Android (ADT), mais depuis 2015, Google officialise Android Studio qui devient alors l'IDE officiel pour le SDK Android. Les développeurs peuvent utiliser n'importe quel éditeur de texte pour modifier les fichiers Java et XML, puis utiliser les outils en ligne de commande (Java Development Kit et Apache Ant sont obligatoires) pour créer, construire et déboguer des applications Android ainsi que contrôler des périphériques Android (pour déclencher un redémarrage, installer un logiciel à distance ou autre) [14].

IV.2. Les composants d'une application Android

Une application Android est un groupe de cinq composants applicatifs qui sont :

- **Activité:** est la constituante principale d'une application sous Android. C'est le seul composant qui soit visible à l'utilisateur final dans lequel est programmé la logique IHM (Interaction Homme Machine).
- **Service :** Les services n'ont pas d'interface graphique et tournent en tache de fond (en arrière-plan). C'est-à-dire qu'une application s'exécute quand une autre application est en train de s'exécuter comme les services de lecture de musique.
- **Intent :**est un message envoyé au système Android pour lui indiquer que nous avons l'intention de faire quelque chose, Il permet d'invoquer d'autres activités ou services.
- **Broadcast receiver :** Il écoute et réagi aux annonces broadcast (réactions sur les évènements extérieurs). Par exemple appel entrant, changement de fuseau horaire.
- **Content provider :** Gestion du partage de données entre applications.

Android permet le partage de composant entre applications ainsi que de gérer leur cycle de vie.

Le schéma suivant (Figure 11ci-dessous), représente le cycle de vie d'une activité sous Android et les méthodes appelées lors des changements d'état. Ainsi, les différentes méthodes sont présentées comme suit :

- **onCreate() / onDestroy():** permet de gérer les opérations à faire avant l'affichage de l'activité, et lorsqu'on détruit complètement l'activité de la mémoire. On met en général peu de code dans onCreate() afin d'afficher l'activité le plus rapidement possible.
- **onStart() / onStop():** ces méthodes sont appelées quand l'activité devient visible/invisible pour l'utilisateur.
- **onPause() / onResume():** une activité peut rester visible mais être mise en pause par le fait qu'une autre activité est en train de démarrer, par exemple B. onPause() ne doit pas être trop long, car B ne sera pas créé tant que onPause() n'a pas fini son exécution.
- **onRestart():** cette méthode supplémentaire est appelée quand on relance une activité qui est passée par onStrop(). Puis onStart() est aussi appelée. Cela permet de différencier le premier lancement d'un ré-lancement.

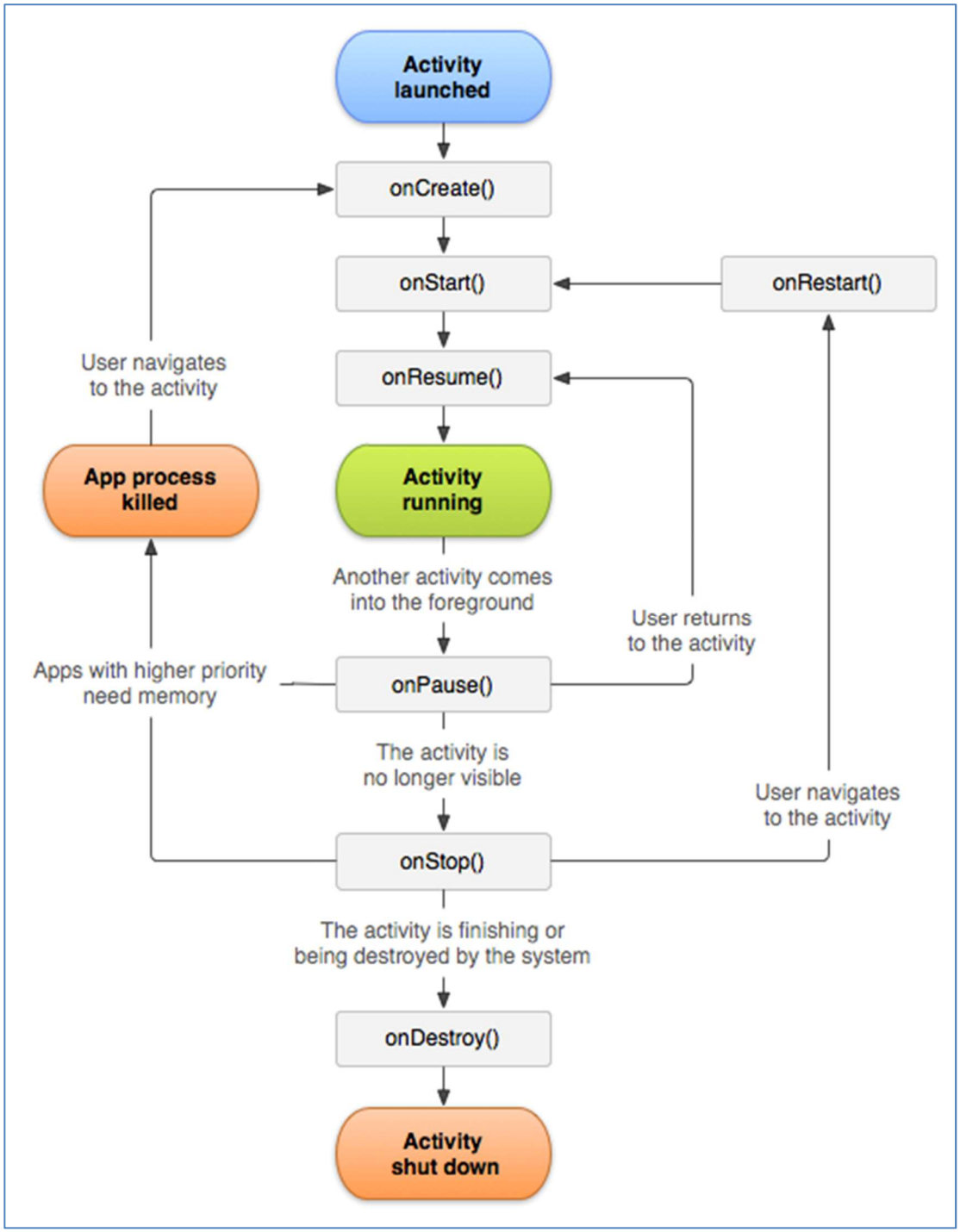

Figure 11. Cycle de vie d'une Activité sous Android

Conclusion

Au niveau de ce chapitre, nous avons bien définit le cahier des charges du projet et spécifié les besoins fonctionnels et techniques de ce dernier.

Cette phase nous a permis de préparer l'analyse orientée objet qui fera le concept du chapitre suivant et marquera le démarrage de l'analyse objet du système à réaliser.

Chapitre V. Analyse

Introduction

Après une spécification détaillée des besoins fonctionnels et techniques de l'application, ce présent chapitre sera consacré entièrement à la phase d'analyse de la méthode 2 TUP.

Toute phase d'analyse est composée d'une vue statique illustrée par les diagrammes de classe participantes et une vue dynamique présentée par des diagrammes de séquences.

I. Vue statique

Dans cette section, nous avons identifié les classes d'analyse qui vont participer à la réalisation des cas d'utilisation.

Il y a trois types de classes : « entité », « contrôle » et « dialogue ». Ces classes seront représentées dans les diagrammes de classes participantes.

Les classes « **entité** » vont seulement posséder des attributs. Ceux-ci représentent généralement les informations persistantes de l'application.

Les classes « **contrôle** » vont seulement posséder des opérations. Ceux-ci correspondent à la logique de l'application, au travail de règles.

Les classes « **dialogue** » vont posséder des attributs et des opérations. Les attributs vont représenter des champs de saisie.

Nous représenterons et détaillerons certains d'entre eux dans la section suivante.

I.1. Diagramme de classes participantes

Le diagramme des classes participantes est un diagramme de classes décrivant les classes d'analyse et dans lequel on ajoute les acteurs.

A ce point du développement, seules les classes dialogue ont des opérations (actions de l'utilisateur sur l'IHM).

Ces opérations correspondent aux opérations système, c'est à dire aux messages entrants que seules les classes de dialogues sont habilitées à intercepter.

Associations :

- Les dialogues ne peuvent être reliés qu'aux contrôles ou à d'autres dialogues (en général, associations unidirectionnelles).
- Les classes métier ne peuvent être reliées qu'aux contrôles ou à d'autres classes métier.
- Les contrôles ont accès à tous les types de classes.

I.1.1. Cas d'utilisation « Créer un utilisateur »

Le diagramme de la Figure 12 ci-dessous présente une vue statique du cas d'utilisation « Créer un utilisateur ».

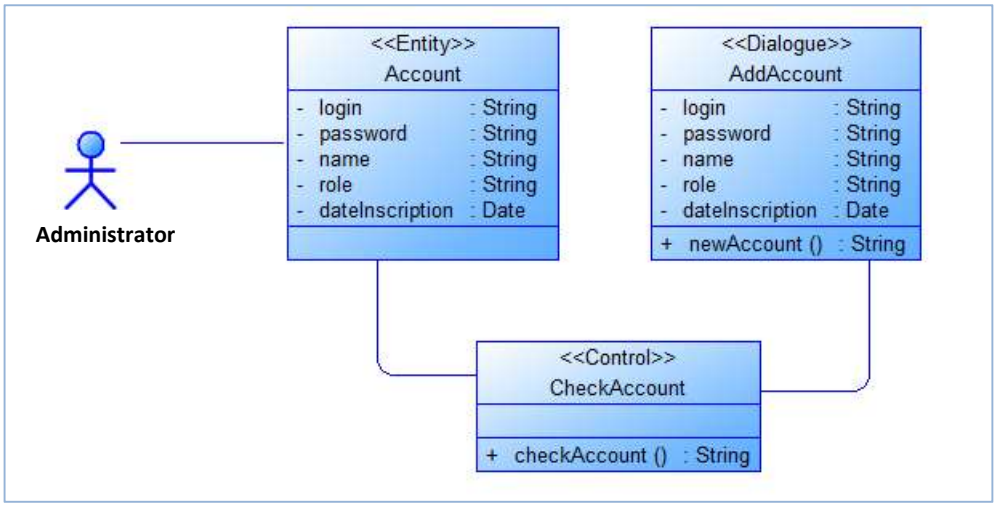

Figure 12. Diagramme de classes participantes du cas « Créer un utilisateur »

L'administrateur fait entrer l'ensemble des attributs relatifs au compte qu'il veut créer à travers une classe de type « Dialogue ». Ces champs seront vérifiés via une classe de type « Control » ainsi une classe « Entity » appelée « Account » sera créée.

I.1.2. Cas d'utilisation « Lancer un audit »

Le diagramme représenté ci-dessous est une vue statique du cas d'utilisation « Lancer un audit ».

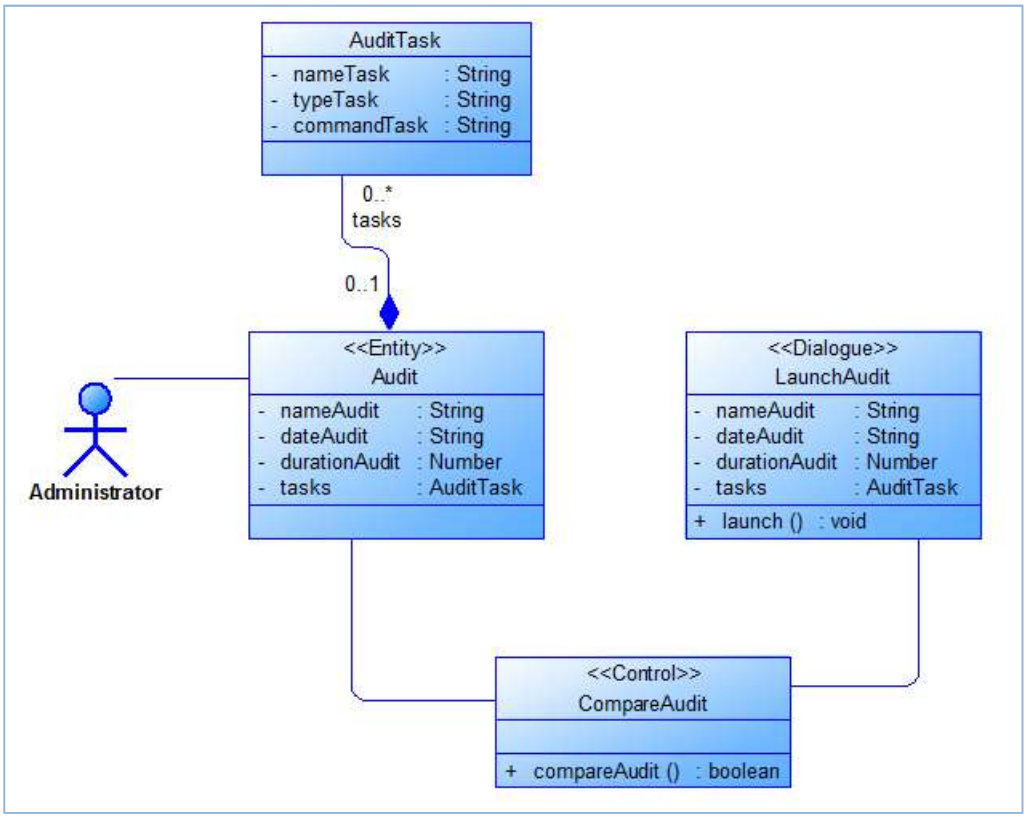

Figure 13. Diagramme de classes participantes du cas «Lancer un audit»

L'administrateur fait créer les tâches relatives à un audit. L'opération « compare » va vérifier les tâches entrées via une classe de type « Control » ainsi une classe « Entity » appelée « Audit » sera créée selon les tâches sélectionnées.

II. Vue dynamique

Dans cette partie, nous utiliserons des diagrammes de séquence pour illustrer les principales tâches effectuées par l'application, les méthodes invoquées ainsi que les messages retournés.

II.1. Diagrammes de séquences

Les diagrammes de séquences sont la représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique dans la formulation UML.

Dans ce qui suit, nous présentons le diagramme de séquence pour chaque cas d'utilisation dans notre système.

II.1.1. Diagramme de séquence du cas « S'authentifier »

L'authentification est la procédure qui consiste à vérifier l'identité d'une entité afin de lui spécifier les permissions qui lui ont été accordées. Le diagramme de séquence, de la Figure 14 ci-dessous, décrit la séquence d'authentification ainsi que les classes impliquées.

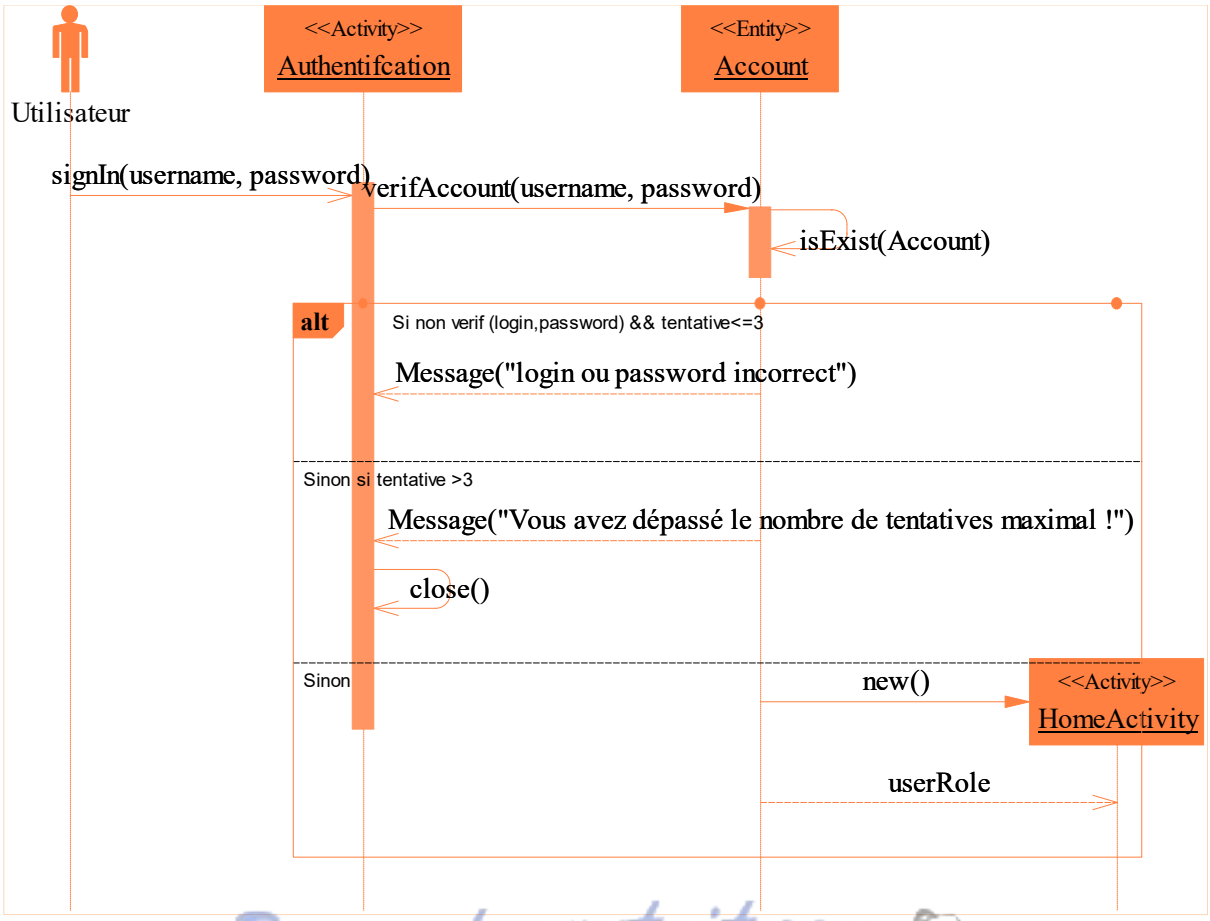

Figure 14. Diagramme de séquence du cas « S'authentifier »

Tout utilisateur de l'application afin de se connecter doit introduire un mot de passe et un login correct sinon des messages d'alertes sera affichés. Cependant, selon son profil, il accèdera à une page et un menu bien spécifique. Il existe deux principaux profils :

- Profil « Client »
- Profil « Administrateur »

II.1.2. Diagramme de séquence du cas « Gérer les équipements »

Le diagramme de séquence, de la Figure 15 présentée ci-dessous, décrit la séquence de l'ajout d'un nouvel équipement ainsi que les classes impliquées.

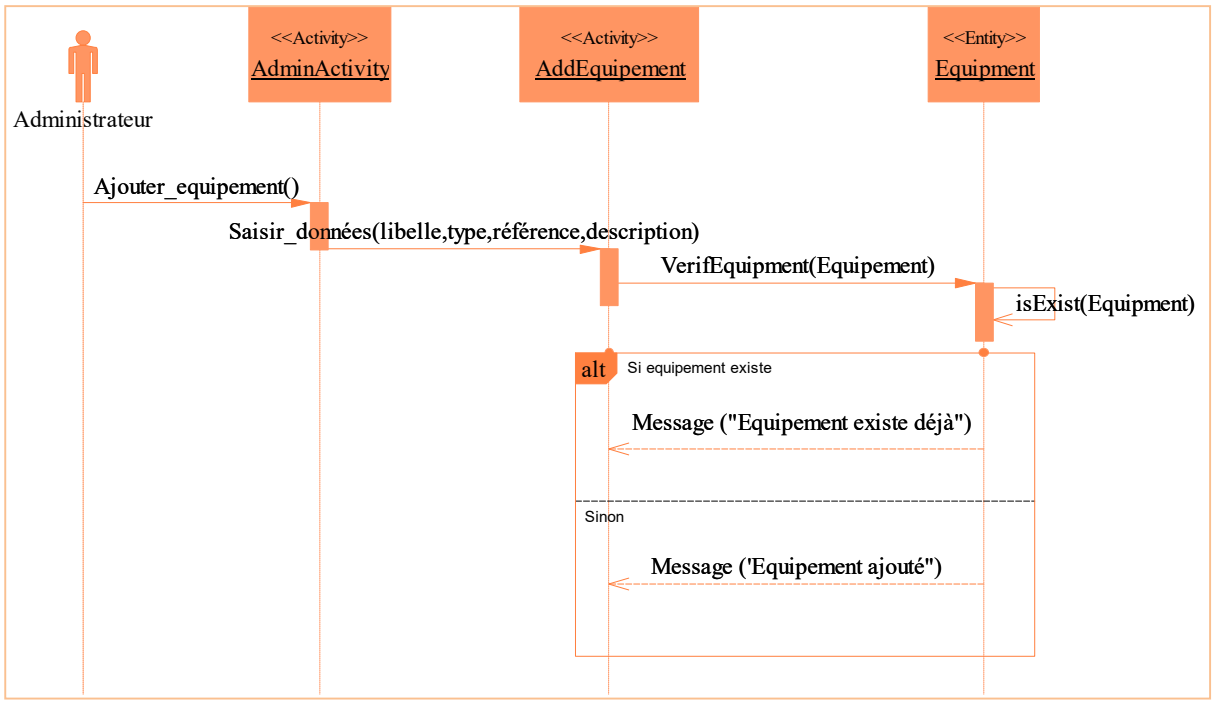

Figure 15. Diagramme de séquence du cas « Ajouter un équipement »

L'administrateur s'authentifie et accède au service pour ajouter un équipement, il saisit le libellé, le type et les caractéristiques de l'équipement et enfin confirme l'ajout. Un message s'affiche « Equipement ajouté » si l'équipement n'existe pas, sinon un message d'erreur s'affiche « Equipement existe déjà » et le système le redirige vers l'interface de saisie d'un nouvel équipement.

II.1.3. Diagramme de séquence du cas « Gérer les utilisateurs »

Le diagramme de séquence, de la Figure 16 présentée ci-dessous, décrit le scénario de la suppression d'un utilisateur.

A travers l'interface de gestion des utilisateurs, l'administrateur sélectionne l'utilisateur qui veut le supprimer parmi la liste des utilisateurs existants. A ce stade l'administrateur décide s'il veut confirmer ou annuler la suppression.

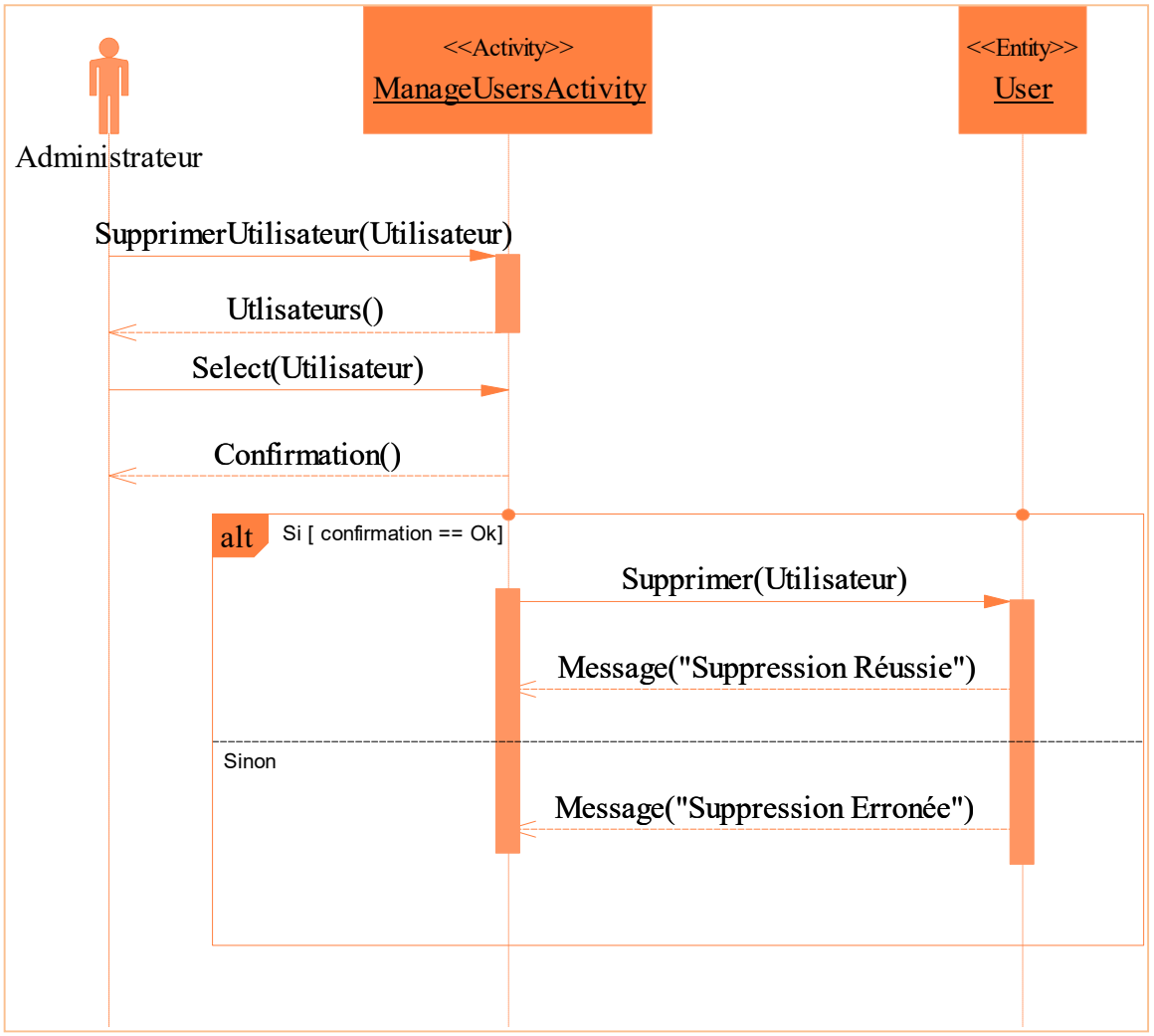

Figure 16. Diagramme de séquence du cas « Supprimer un utilisateur »

II.1.4. Diagramme de séquence du cas « Gérer les vulnérabilités »

Le diagramme de séquence, de la Figure 17 présentée ci-dessous, décrit la séquence de la modification d'une vulnérabilité déjà existant ainsi que les classes impliquées.

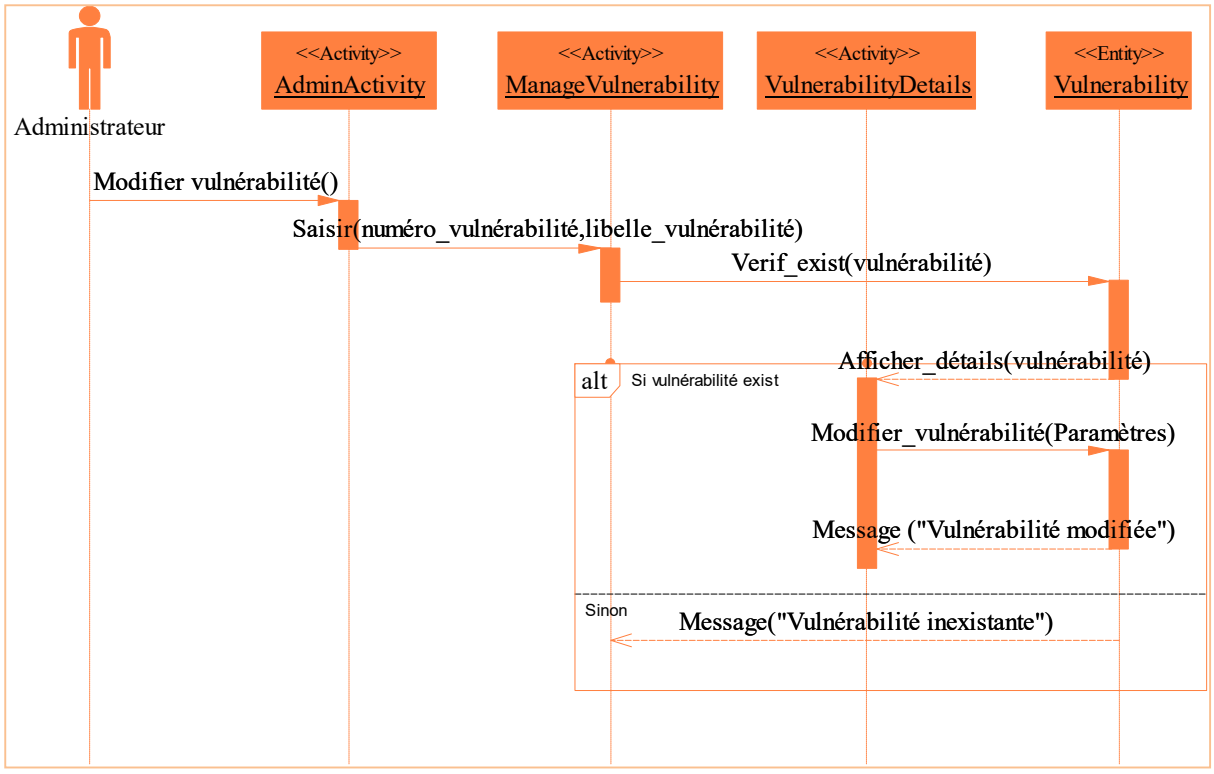

Figure 17. Diagramme de séquence du cas « Modifier une vulnérabilité »

Afin de modifier une vulnérabilité, il faut préciser tout d'abord son identifiant. Les données relatives à la vulnérabilité seront affichées. L'administrateur modifie les informations, puis seront transmises à la base afin de changer les champs de données d'une entité « Vulnerability » déjà existant. Ainsi le message retourné sera affichée sur l'écran informant l'utilisateur que la vulnérabilité a été modifiée avec succès.

II.1.5. Diagramme de séquence du cas « Auditer un équipement »

La Figure 18 ci-dessous, détaille les cas d'utilisation liés au cas « Auditer un équipement ».

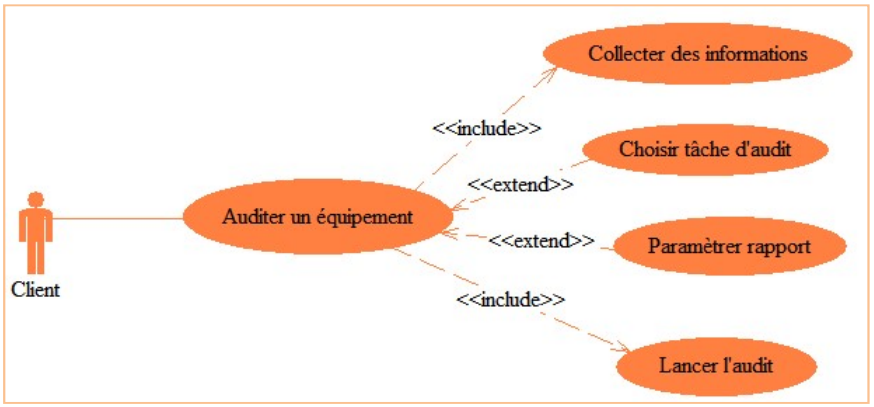

Figure 18. Cas d'utilisation détaillé du cas « Auditer un équipement »

Le Tableau 7 et la Figure 19 ci-dessous décrivent le scénario du lancement d'un audit.

Cas d'utilisation	Auditer un équipement
Acteur	Client
Précondition	Le client s'authentifie et accède au service pour auditer un équipement
Scénario nominal	**1.** Le client clique sur « Audit » dans le menu **2.** Le client clique sur «Collecter des informations» **3.** Une interface de collecte des informations s'affiche **4.** Le client choisi un équipement à auditer et un type de source de données **5.** Le client lance la collecte des informations **6.** Le système enregistre le résultat de la collecte **7.** Le client clique sur « Audit » dans le menu puis il clique sur «Choisir une tâche» **8.** Une interface s'affiche qui contient une liste des tâches d'audit qui peuvent être exécutées sur l'équipement choisi par le client **9.** Le client choisi une tâche ou un ensemble de tâches **10.** Le système enregistre la tâche choisi par le client **11.** Le client clique sur « Audit » dans le menu puis il clique sur «Paramétrer le rapport» **12.** Une interface pour choisir un format du rapport s'affiche **13.** Le client choisi un format pour le rapport d'audit qui sera généré à la fin de l'audit **14.** Le système enregistre le format choisi par le client **15.** Le client clique sur « Audit » dans le menu puis il clique sur «Lancer audit» **16.** Une fenêtre qui contient les paramètres choisies par le client aux étapes précédentes s'affichent **17.** Le client lance l'audit **18.** Le système sélectionne la liste la plus exhaustive possible des vulnérabilités de l'équipement choisi par le client **19.** Le système exécute les tâches choisies **20.** Le système compare le résultat de la collecte de l'information de **l'étape 6** et le résultat de l'exécution des tâches à la liste des

	vulnérabilités sélectionnées à **l'étape 18** **21.** Le système enregistre le résultat de la comparaison qui représente les vulnérabilités de l'équipement dégagées par l'audit **22.** Le système sélectionne les recommandations nécessaires **23.** Le système génère le rapport et l'enregistre **24.** Le système affiche le rapport au client avec le format qu'il a choisi
Scénario alternatif	**12.** Le client n'a pas réalisé l'étape de collecte des informations Le système lui affiche un message d'erreur « Vous devez choisir un équipement et lancer la collecte des informations » **Retour à l'étape 1**

Tableau 7. Scénario de « Lancer un audit »

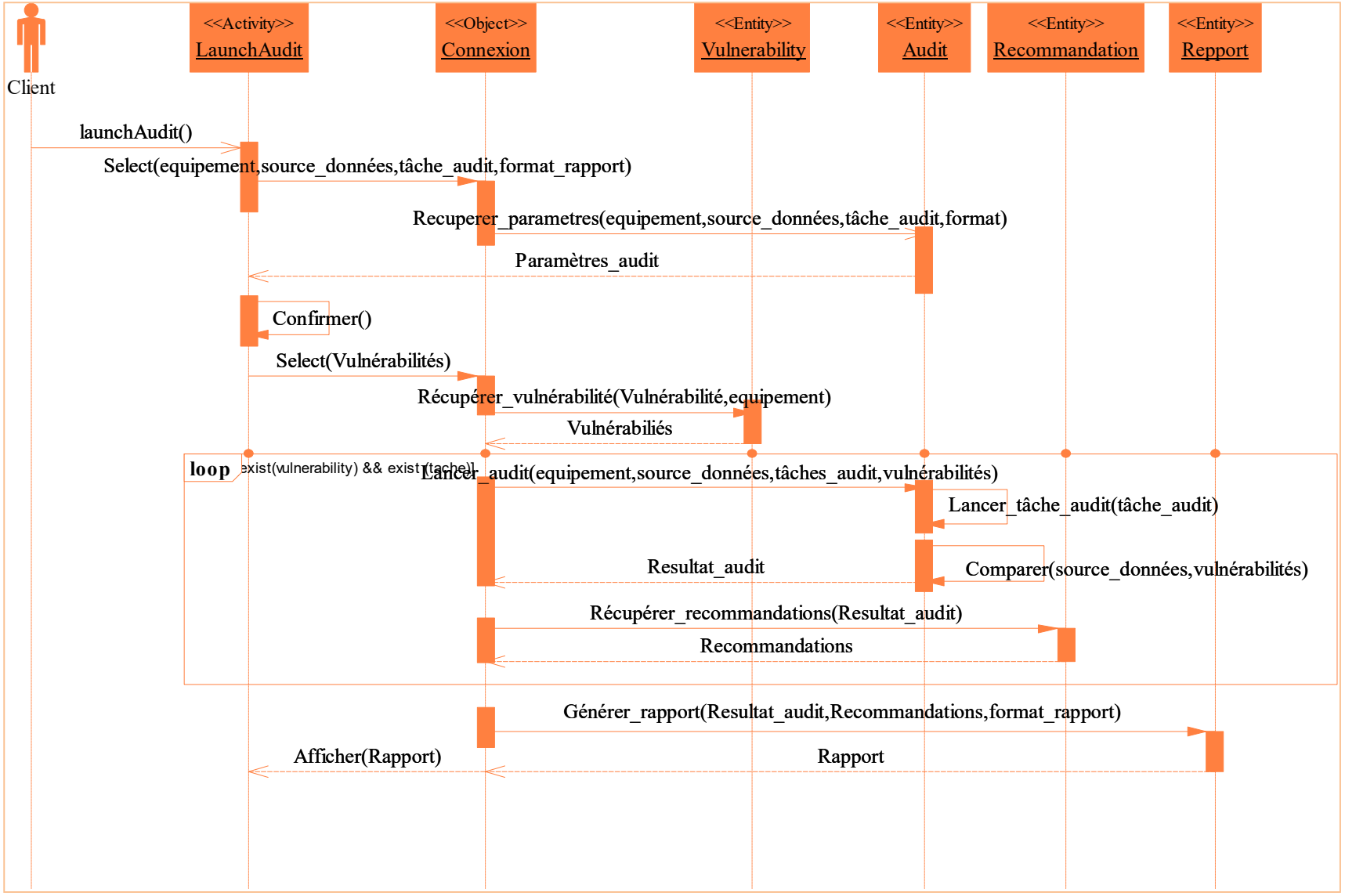

Figure 19. Diagramme de séquence du cas « Lancer un Audit »

II.1.6. Cas d'utilisation « Collecter les informations »

La Figure 20 ci-dessous, détaille les cas d'utilisation liés au cas « Collecter les informations ».

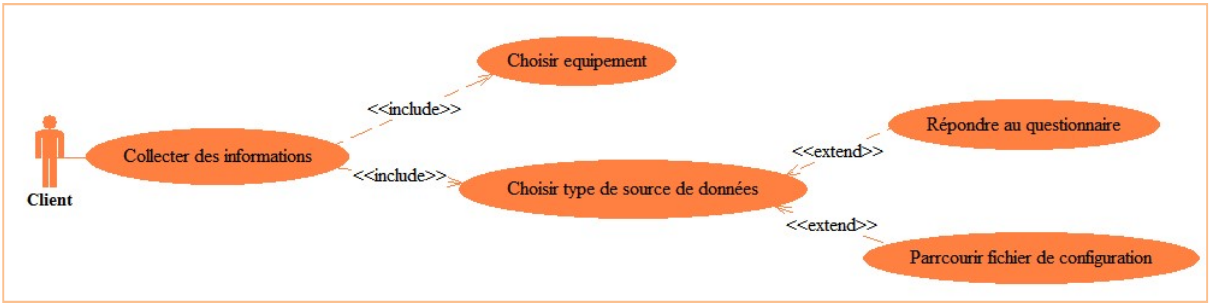

Figure 20. Cas d'utilisation détaillé du cas « Collecter les informations »

Le Tableau 8 Tableau 7 et la Figure 21 ci-dessous décrivent le scénario de la collecte des informations.

Cas d'utilisation	Collecter les informations
Acteur	Client
Précondition	Le client s'authentifie et accède au service pour auditer un équipement
Scénario nominal	**1.** Le client clique sur « Audit » dans le menu **2.** Le client clique sur « Collecter des informations » **3.** Une interface de collecte des informations s'affiche **4.** Le client choisi un équipement à auditer, il doit choisir son type et ses caractéristiques **5.** Le client choisi un type de source de données **a.** Il choisit de répondre à un questionnaire, une interface qui s'affiche avec une liste des questions, le client y répond **b.** Il choisit comme source de données le fichier de configuration de l'équipement choisi, le client parcourt le fichier
Scénario alternatif	**5.a.** Le client n'a pas répondu à toutes les questions : Le système affiche un message « Vous devez répondre à toutes le questions » **Retour à l'étape 5** **5.b.** Le client a donné un fichier qui ne correspond pas à un fichier de configuration : Le système affiche un message « Vous devez ajouter un fichier de configuration » **Retour à l'étape 5**

Tableau 8. Scénario de « Collecter les informations »

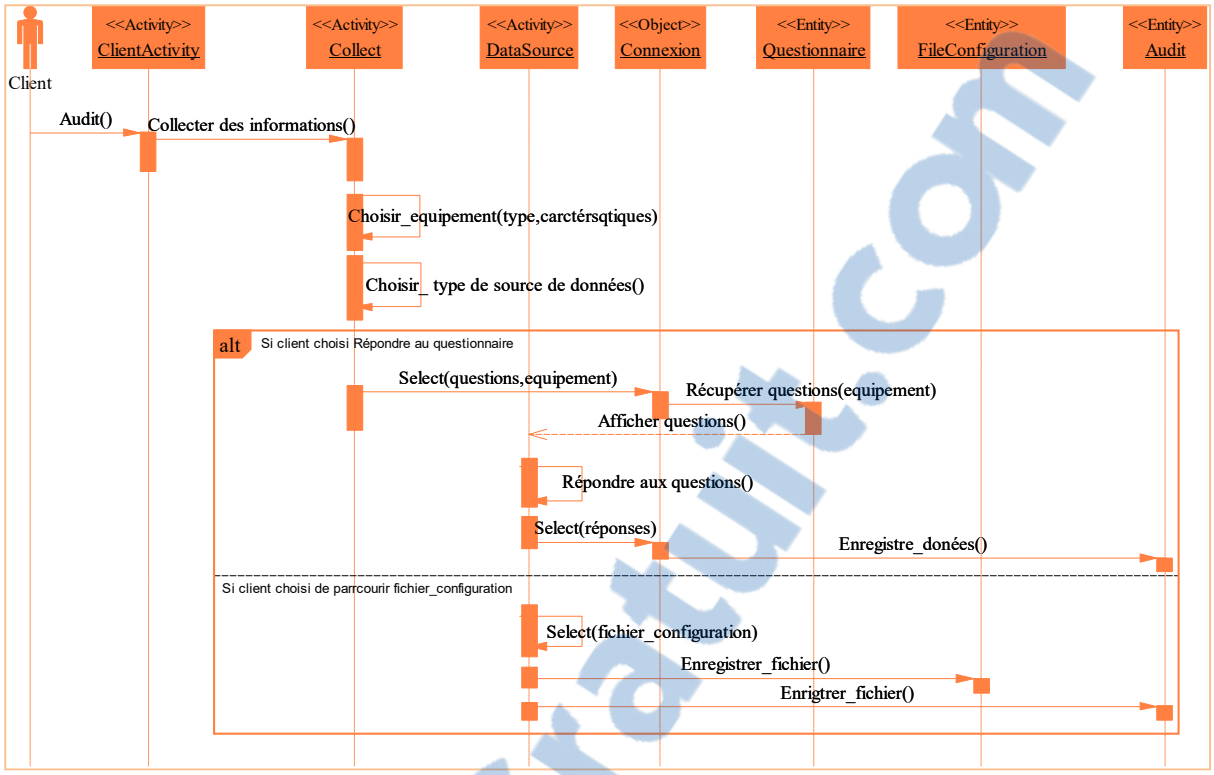

Figure 21. Diagramme de séquence du cas « Collecter des informations »

II.2. Diagramme d'activités

Un diagramme d'activités permet de modéliser un processus interactif, global ou partiel pour un système donné (logiciel, système d'information). Il est recommandable pour exprimer une dimension temporelle sur une partie du modèle, à partir de diagrammes de classes ou de cas d'utilisation, par exemple.

Le diagramme d'activités est une représentation proche de l'organigramme ; la description d'un cas d'utilisation par un diagramme d'activités correspond à sa traduction algorithmique.

Une activité est l'exécution d'une partie du cas d'utilisation, elle est représentée par un rectangle aux bords arrondis.

Dans ce qui suit, nous présentons les diagrammes d'activités pour quelques cas d'utilisation dans notre système.

II.2.1. Cas d'utilisation « Créer un utilisateur »

La Figure 22 ci-dessous représente le diagramme d'activités du cas d'utilisation « Créer un utilisateur ».

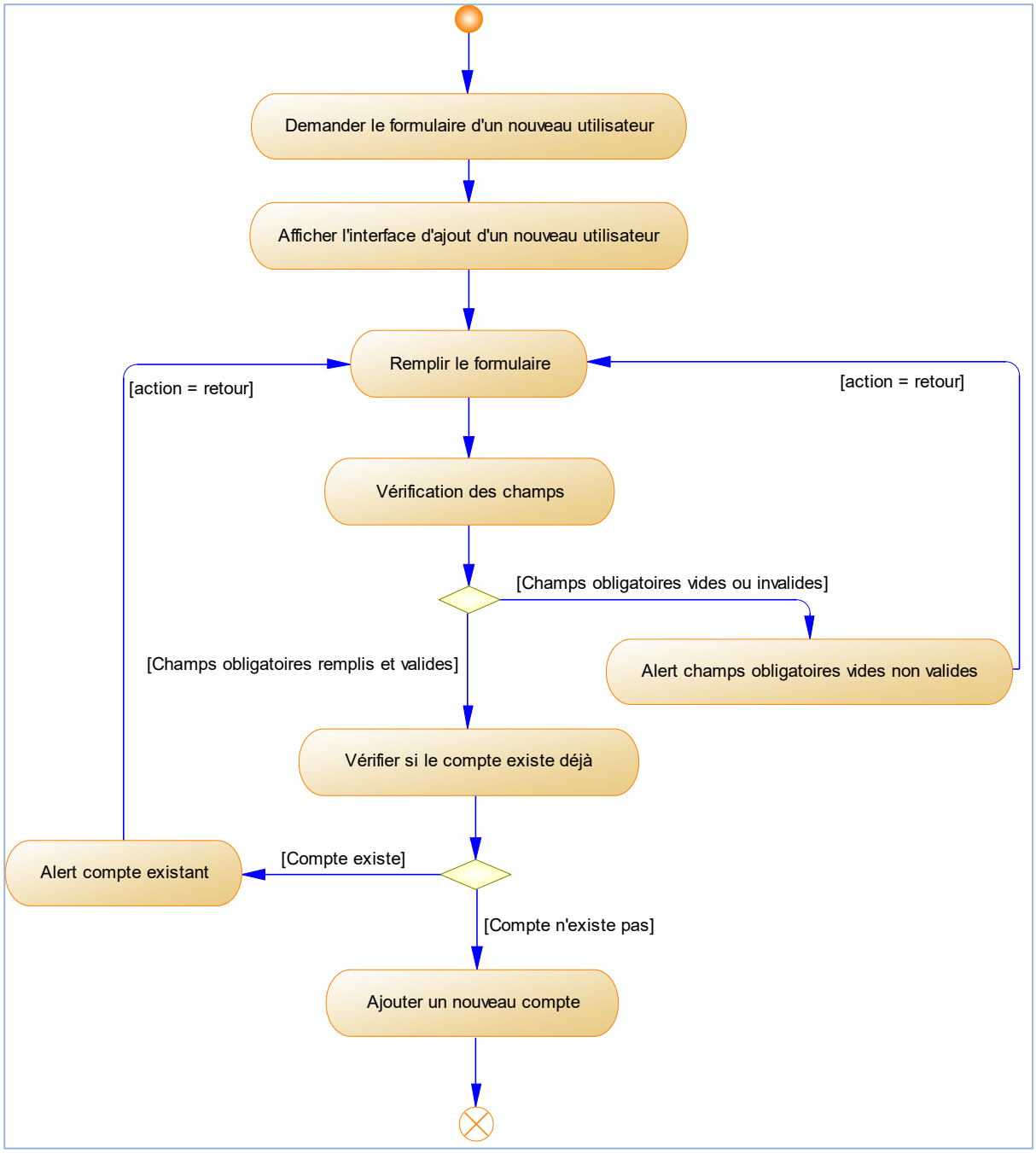

Figure 22. Diagramme d'activités du cas d'utilisation « Créer un utilisateur »

II.2.2. Cas d'utilisation « Collecter les informations »

La figure ci-dessous représente le diagramme d'activités du cas d'utilisation « Collecter les informations ».

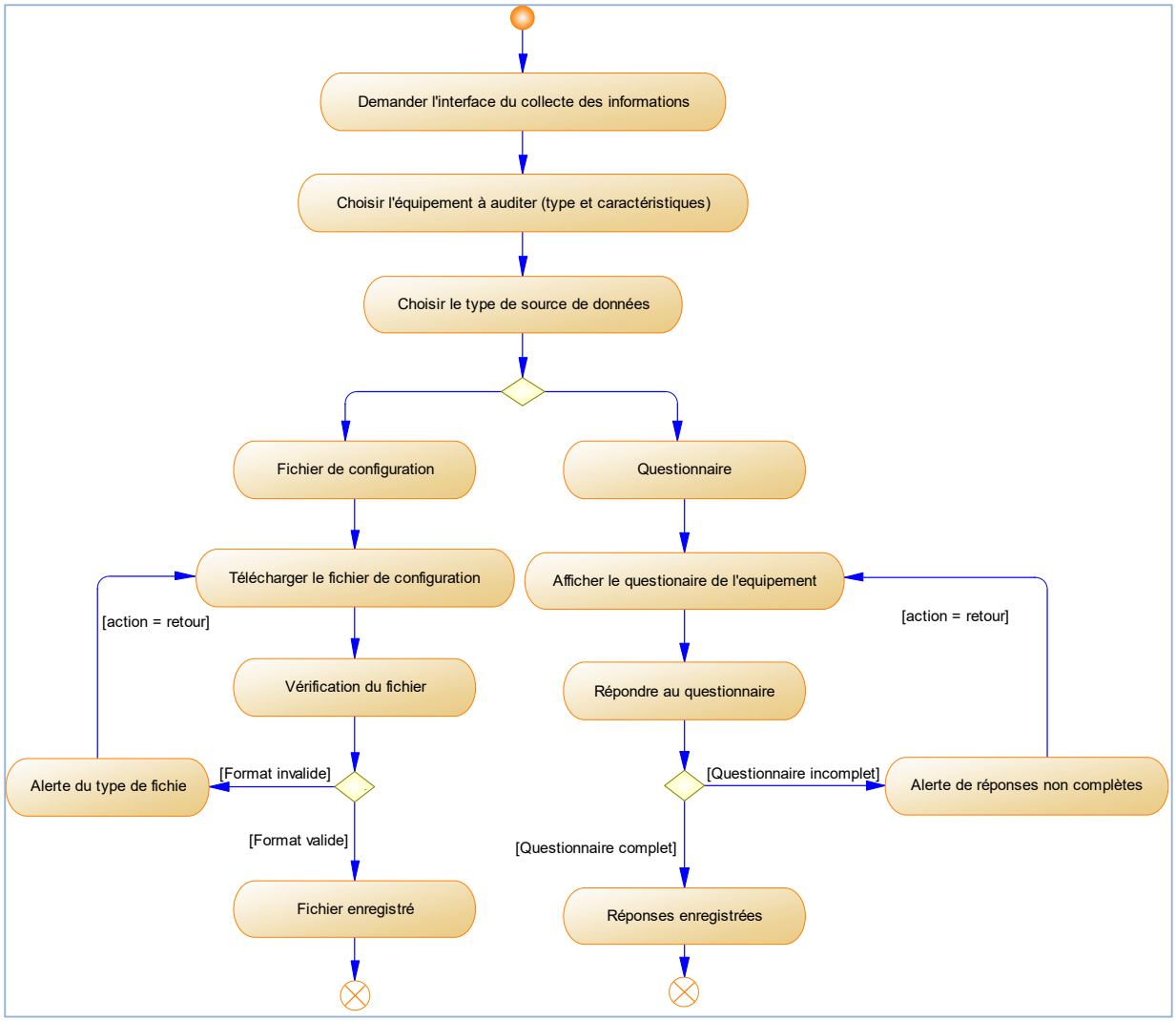

Figure 23. Diagramme d'activités du cas d'utilisation « Collecter les informations »

Conclusion

Dans ce chapitre, nous avons décrit l'étape d'analyse de la méthode 2TUP de point de vue statique avec la présentation de quelques diagrammes de classe participante et de point de vue dynamique en précisant certains diagrammes de séquence. Le chapitre suivant présentera l'étape de la conception.

Chapitre VI. Conception

Introduction

Après la phase d'analyse, nous allons entamer, dans ce chapitre la conception du projet. En utilisant la méthode 2TUP, dans un premier temps, nous commenceront par une conception préliminaire présentant le logo et la conception de l'application mobile et la charte graphique par la suite nous passeront à une conception détaillée du projet en tenant compte des contraintes fonctionnelles et technologiques rencontrées.

I. Conception préliminaire

Cette partie est consacrée pour la présentation de la charte graphique de notre application qui est un document de travail qui contient l'ensemble des règles d'utilisation des signes graphiques qui constitue l'identité graphique pour pouvoir les décliner sur différents support d'une même identité.

I.1. Le logo

Le logotype de notre solution, représenté dans la Figure 24 ci-dessous, est à l'origine de relation de lettrage « Audit », le mot « KMCT » abréviation de « Khamis Mushait College of Technology » et une forme d'une loupe sur le logo de l'établissement, ceci est inspiré du fait qu'au cours d'un audit il y a une analyse et une recherche afin de mettre en lumière les failles à corriger par la suite.

Figure 24. Logo "KMCT Audit"

I.2. L'identité visuelle

La gamme de couleurs est inspirée de la charte graphique de l'établissement. Pour garder une tendance visuelle nous avons recourt à des nuances de bleu qui est la couleur de la détente physique et mentale et procure un sentiment de sécurité et de confiance. En ayant recourt à l'orange nous avons visé à produire un contraste visuelle. L'orange est audace, optimiste et rayonnant. On associe cette couleur à la créativité et à la communication car l'orange favorise la sécurité, l'estime de soi, le combat de dépression et ravive la bonne humeur.

I.3. Gabarit de mis en page

C'est la dernière étape avant la réalisation graphique, qui se présent par une version papier du produit final où nous avons schématisé le produit écran par écran. La Figure 25 suivante illustre le gabarit de l'interface principale de notre produit.

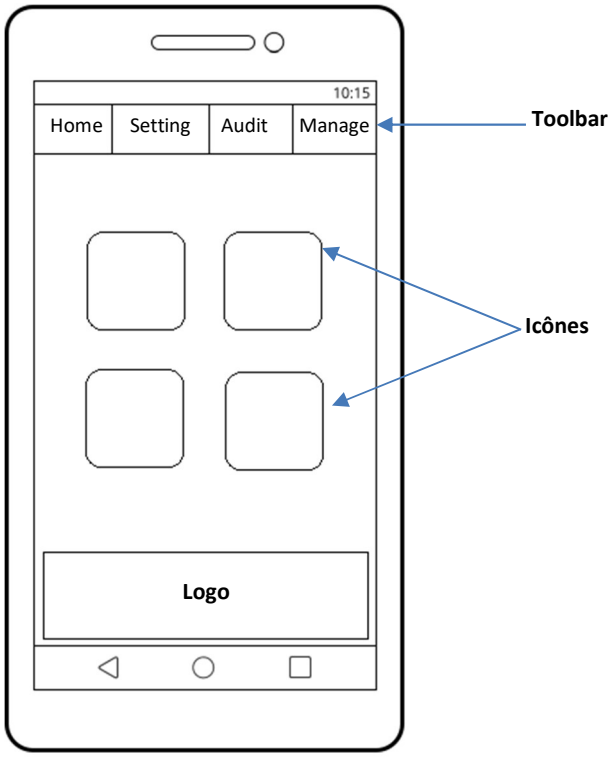

Figure 25. Gabarit de l'interface principale

II. Conception détaillée

Dans cette partie, nous allons présenter le diagramme de classe général de l'application, soit une description détaillée des attributs et méthodes utilisés au cours du développement du projet.

II.1. Diagramme de classes côté serveur

Le diagramme de classes est composé de onze classes. Chaque classe décrit les responsabilités, le comportement et le type d'un ensemble d'objets. Les éléments de cet ensemble sont les instances de la classe. C'est aussi un ensemble de fonctions et de données (attributs) qui sont liées ensembles par un champ sémantique. Les classes sont utilisées dans la programmation orientée objet. Elles permettent de modéliser un programme et ainsi de découper une tâche complexe en plusieurs petits travaux simples.

La Figure 26 ci-dessous représente le diagramme de classes côté serveur de notre solution « KMCT Audit ».

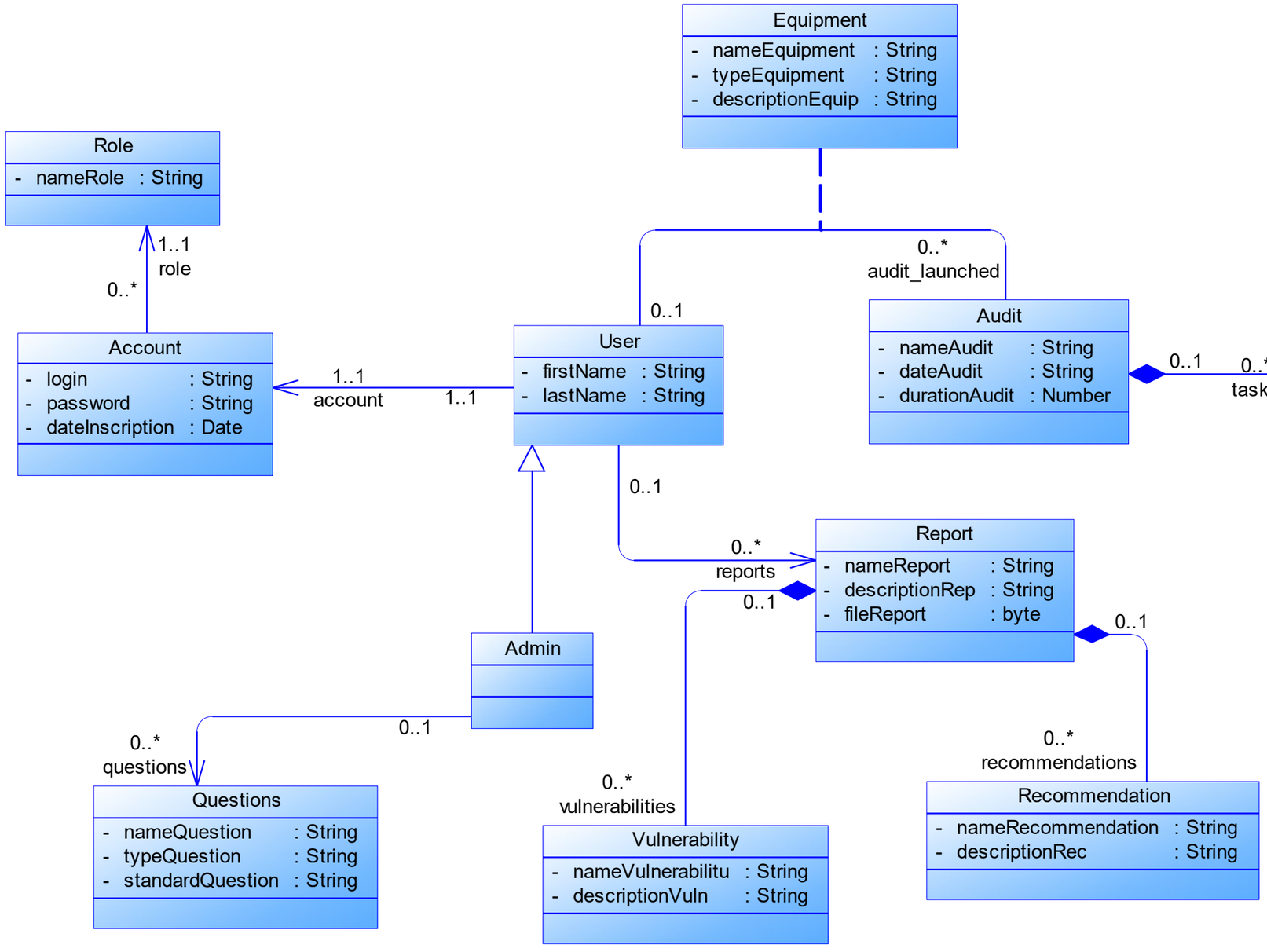

Figure 26. Diagramme de classe côté serveur

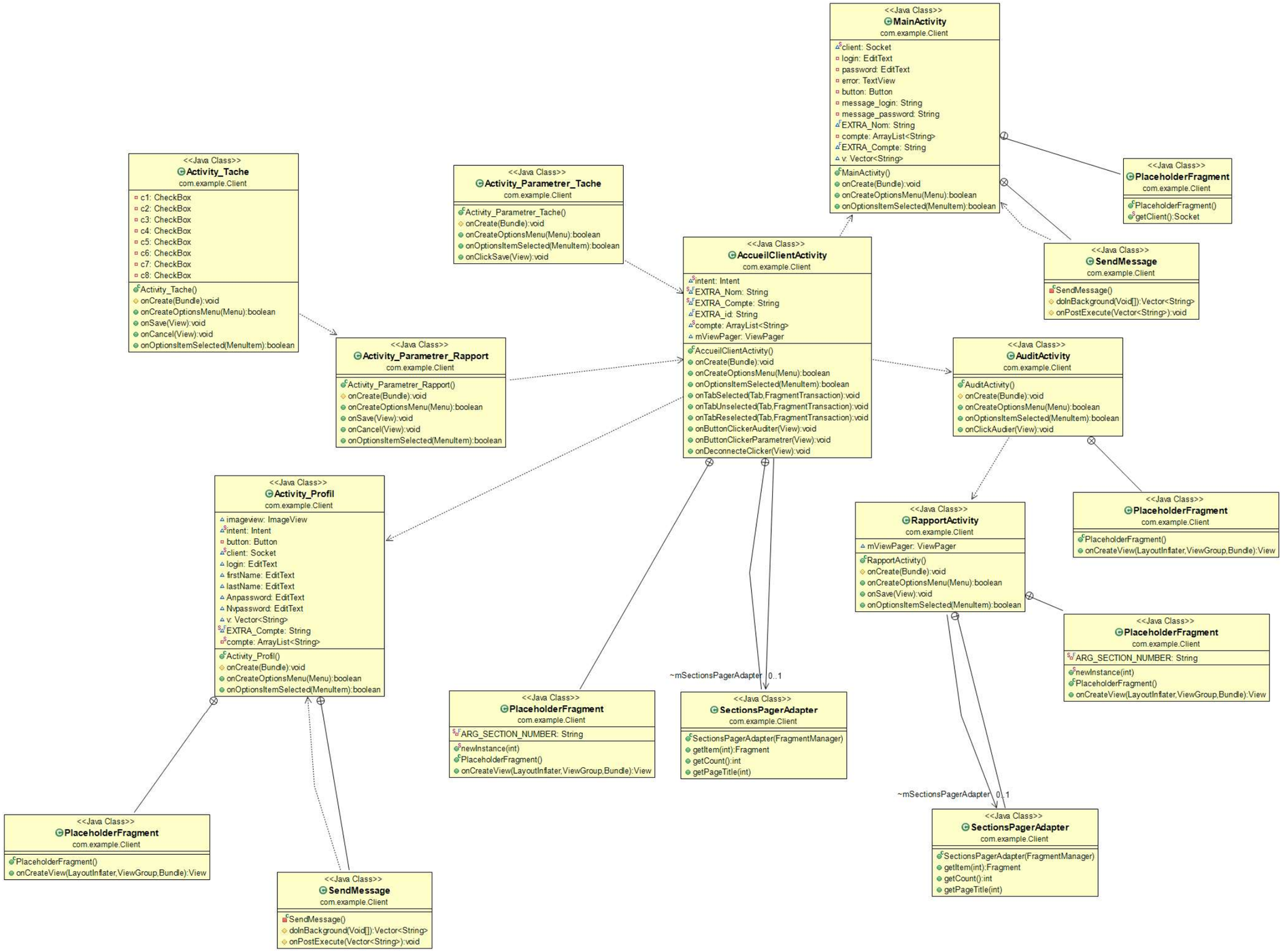

Figure 27. Diagramme de classe côté client - Partie 1

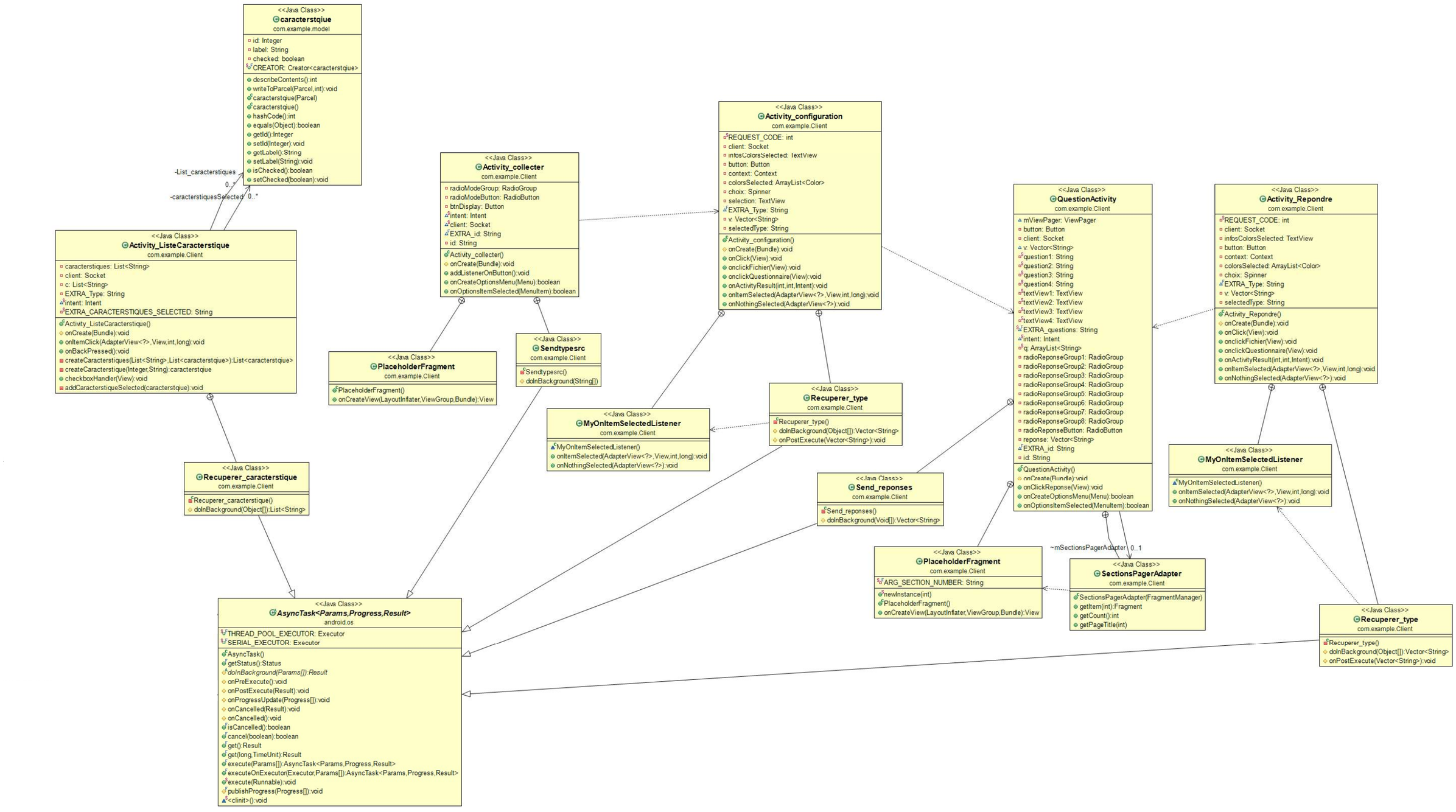

Figure 28. Diagramme de classe côté client - Partie 2

II.2. Diagramme de classes côté client

La Figure 27 et la Figure 28 ci-dessus illustrent une partie du diagramme de classes générale côté client.

III. Architecture des plateformes utilisées

La technologie Objet requiert une architecture. C'est cette architecture qui organise les interactions entre objets. Nous avons l'habitude de regrouper ces objets en classes, ces classes en domaines, et ces domaines en couches. Les couches permettent de présenter l'architecture de l'application. Les équipes de réalisation s'attribuent alors des responsabilités sur le développement de chaque couche. Aussi, si modéliser est indispensable, construire une architecture à couche est un critère de qualité dans le cadre d'un développement Objet. Reste à choisir le nombre de couches et à définir leur contenu.

III.1. Côté serveur

La plateforme Java EE 5 est basée sur un modèle en couches. Chaque couche communique avec sa couche inférieure et sa couche supérieure. Le langage JAVA étant totalement portable, le développement du système ne dépendra absolument pas de la machine sur laquelle il fonctionnera. La Figure 29 ci-dessous représente les couches de l'architecture Java EE 5.

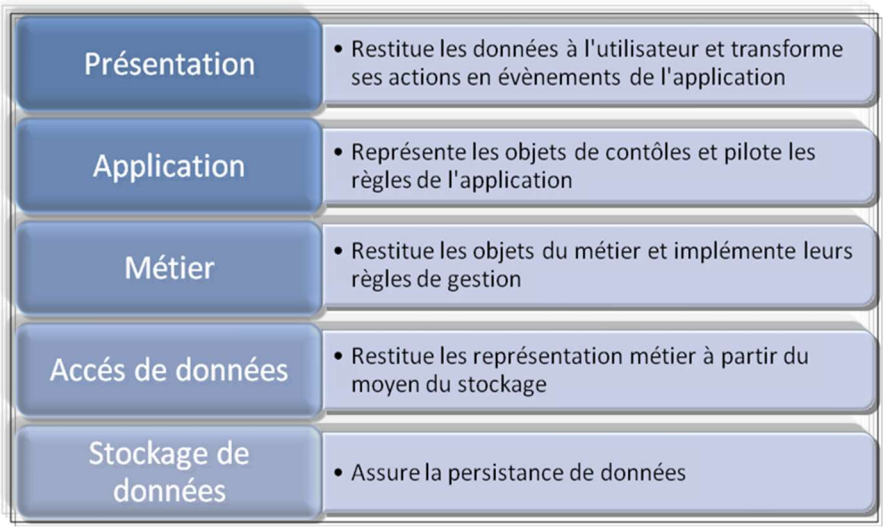

Figure 29. Les couches de l'architecture Java EE 5

Nous n'avons pas travaillé sur la couche « Présentation » de cette architecture car dans notre cas la partie présentation est développée sur la partie mobile.

III.2. Côté client

Le système Android est une pile logicielle complète qui est divisée en quatre zones comme le montre la Figure 30 suivante.

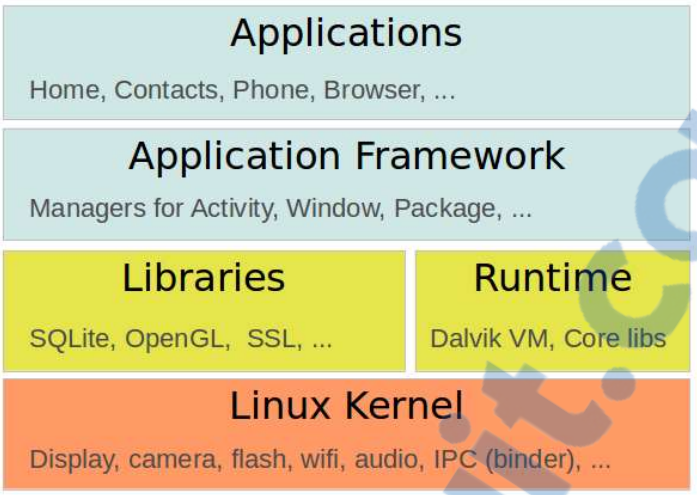

Figure 30. Les composants de la plate-forme Android

Ces éléments peuvent être décrits comme :

- **Applications :** Le projet Open Source Android contient plusieurs applications par défaut comme le navigateur, l'appareil photo, la galerie, la musique, le téléphone et plus encore ;
- **Application Framework :** Une API qui permet aux applications Android d'interagir avec le système Android ;
- **Libraries and runtime :** Les bibliothèques pour de nombreuses fonctions communes (par exemple : le rendu graphique, le stockage de données, la navigation sur le Web, etc.) de l'Application Framework et de la machine virtuelle **Dalvik**, ainsi que le noyau de bibliothèques Java pour exécuter des applications Android ;
- **Linux kernel :** La couche de communication avec le matériel sous-jacent.

Le noyau Linux, les bibliothèques et le moteur d'exécution sont encapsulés par l'Application Framework. Le développeur d'applications Android fonctionne généralement avec les deux couches supérieures pour créer de nouvelles applications Android.

Conclusion

Dans ce chapitre, nous avons présenté l'étape de la conception soit le diagramme de classe descriptif du projet.

Le chapitre qui suit sera consacré spécialement à la phase de réalisation et aux différents tests qui ont été faits tout au long de ce projet.

Chapitre VII. Réalisation et Tests

Introduction

Après avoir finalisé l'étape de conception, nous passons dans ce chapitre à l'implémentation de notre application. Nous commençons par la définition de notre environnement de développement. Par la suite, nous enchainons par la présentation et la description détaillée des différentes étapes de la solution finale.

I. Description de l'environnement de développement

Au niveau de la phase de la conception, nous avons eu recours à Power Designer pour la modélisation UML.

Par rapport à la phase de l'implémentation de la partie mobile, nous avons utilisé le kit de développement d'Android pour Eclipse.

Et pour l'implémentation de la partie serveur, nous avons utilisé l'IDE Eclipse et MySQL comme système de gestion de la base de données.

II. Architecture de l'application

La Figure 31 ci-dessous, illustre l'architecture de notre application. Le client mobile communique avec serveur d'audit à travers les sockets et l'échange entre eux sera sécurisé par le chiffrement des données.

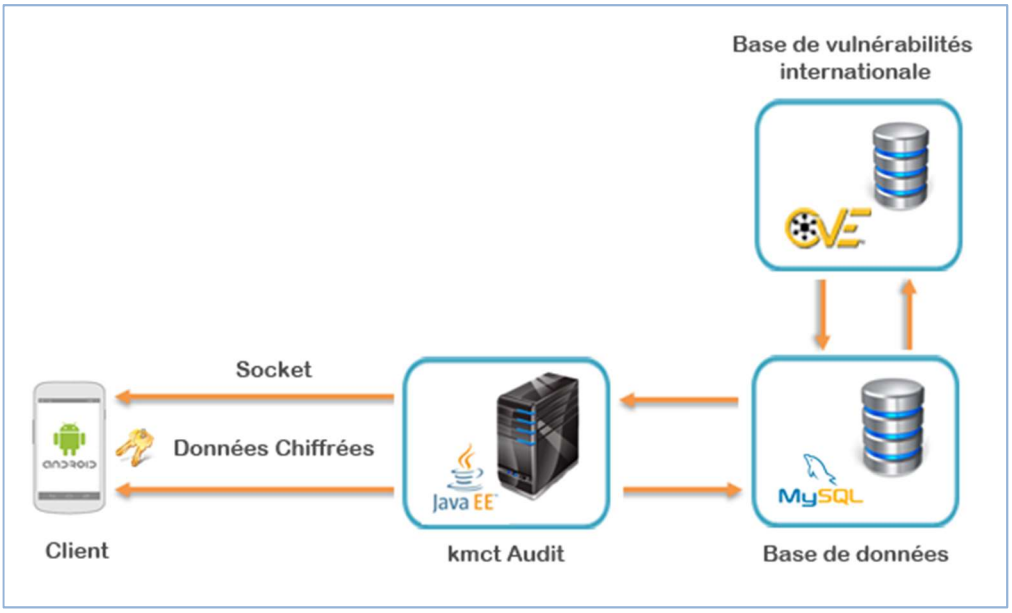

Figure 31. Architecture de l'application

La base de vulnérabilité « CVE » contient une liste des noms standardisés pour les vulnérabilités assurément connues et les révélations relatives à la sécurité et associe à chaque vulnérabilité un identifiant et un lien sur le web.

Nous consultons cette liste pour associer aux vulnérabilités détectées et enregistrées dans

notre base de données des liens sur le web et lorsqu'on génère le rapport ces liens seront y ajoutés.

Cela aide l'auditeur à avoir davantage d'information sur les vulnérabilités détectées sur son équipement.

Cette liste est disponible sur le site http://cve.mitre.org/ au format XML, donc on a besoin d'utiliser un parseur pour analyser son contenu et récupérer les **balises**, leurs contenus et leurs **attributs.**

III. Description de la solution finale

Dans cette section, nous allons présenter la solution finale détaillée. A ce stade, nous allons présenter notre application mobile pour Android à travers les différentes interfaces de nos utilisateurs, « le client » et « l'administrateur » à travers des imprimes-écrans réalisés sur l'application.

III.1. Interface d'authentification

Les utilisateurs de notre service d'audit « KMCT Audit » sont synthétisés en deux catégories : Administrateur et Simple Utilisateur.

Chaque utilisateur peut consulter son propre compte après son authentification ainsi que les fonctionnalités qu'il peut exécuter selon son type. Donc il doit saisir son « nom d'utilisateur » et son « mot de passe » de la façon figurée dans la Figure 32 ci-dessous.

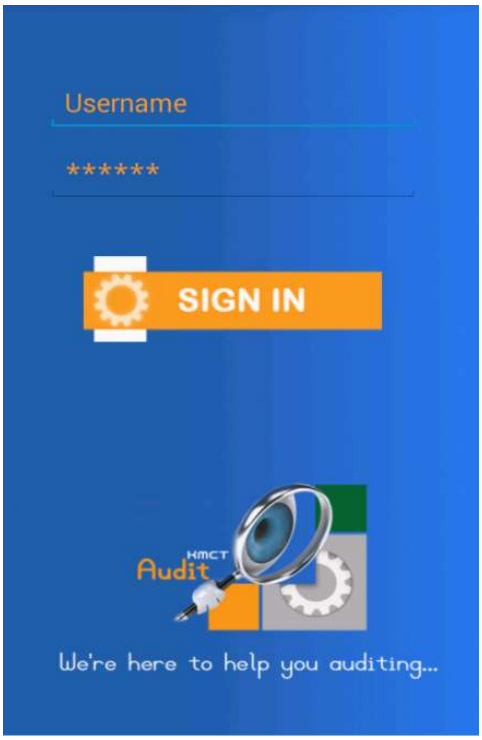

Figure 32. Interface d'authentification

Si l'utilisateur saisie un « Nom d'utilisateur » ou un « Mot de passe » non valide un message d'erreur sera affiché (voir Figure 33 ci-dessous).

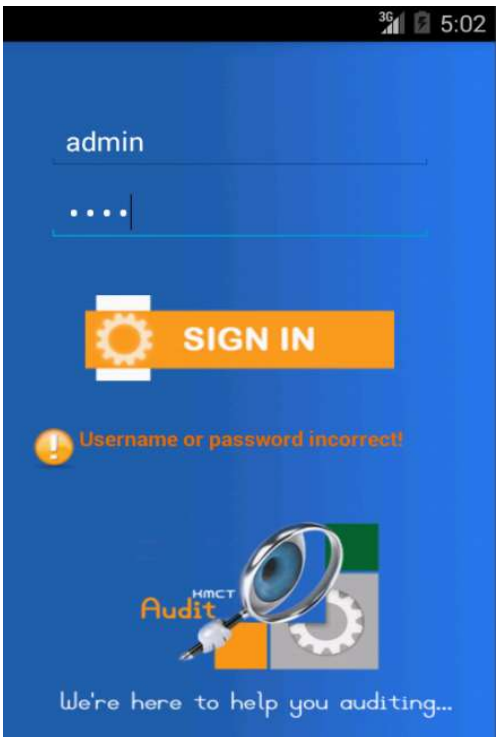

Figure 33. Erreur d'Authentification

Si l'utilisateur essaie de se connecter plus que trois fois en utilisant un mot de passe ou un login erroné un message d'erreur s'affiche pour l'informer qu'il a dépassé le nombre de tentatives autorisées, le programme se fermera automatiquement par la suite.

III.2. Interface d'accueil

La Figure 34, représentée ci-dessous, est la première interface de notre application « KMCT Audit » sous Android. Elle est composée essentiellement d' :

- un menu en haut avec les fonctionnalités offertes pour un client,
- un message de bienvenu personnalisé,
- un bouton de déconnexion,
- une description de l'objectif du service,

et en bas le logo et le slogan de notre service.

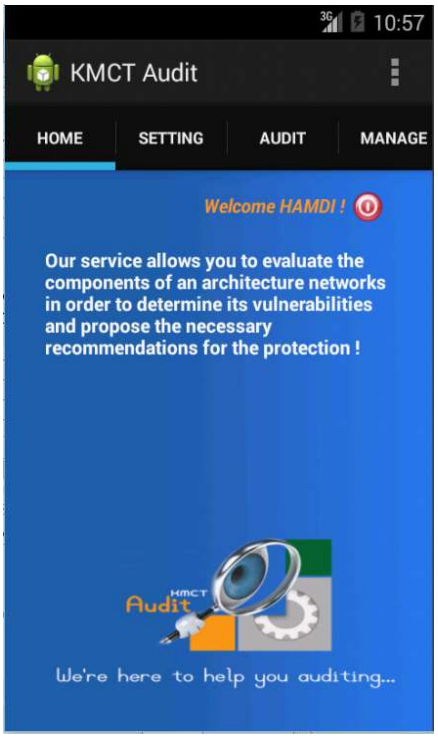

Figure 34. Interface d'accueil

III.3. Interface « Settings »

La Figure 35 représentée ci-dessous, permet au client de paramétrer son profil (ses informations personnelles) et de paramétrer les tâches d'audit qu'il souhaite exécuter en cours d'audit.

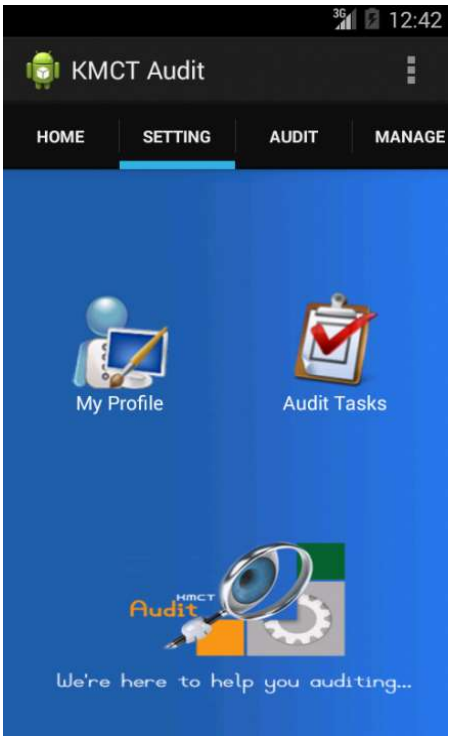

Figure 35. Interface "Setting"

III.3.1. Interface « My Profile »

A travers l'interface « My Profile » (Figure 36 ci-dessous), chaque utilisateur a le droit de modifier ses informations personnelles telles que (Nom, Prénom, Login et Mot de passe).

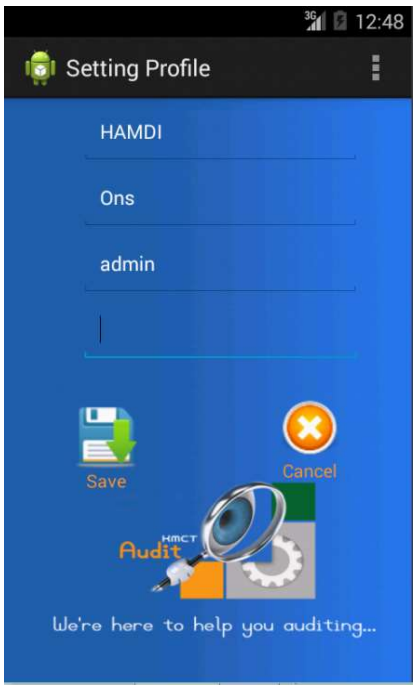

Figure 36. Interface "My Profile"

III.3.2. Interface « Audit Tasks »

Pour auditer un équipement réseau, l'auditeur doit tout d'abord paramétrer les tâches qu'il souhaite appliquer comme indiqué dans la Figure 37 suivante.

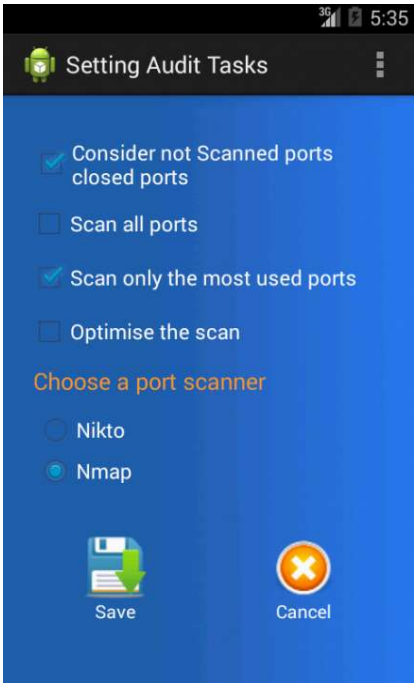

Figure 37. Interface "Audit Tasks"

Pour le scan des ports on peut utiliser l'outil « Nmap » ou « Nikto ». Ces outils seront lancés à partir de notre servie à travers l'invite de commande.

Dans notre cas nous avons choisi les paramètres suivants :

- Considérer les ports non scannés ports fermés
- Ne scanner que les ports les plus utilisés
- L'outil de scan Nmap

III.4. Interface des services « KMCT Audit »

La Figure 38, illustrée ci-dessus, expose les principaux services de notre application qui sont comme suit :

- **« Collect »** pour lancer la collecte des informations pour un équipement donné.
- **« Tasks »** pour choisir les tâches paramètres auparavant.
- **« Setting »** pour paramétrer le format du rapport.
- **« Launch Audit »** pour lancer l'audit d'un équipement selon les paramètres sélectionnés par l'auditeur.

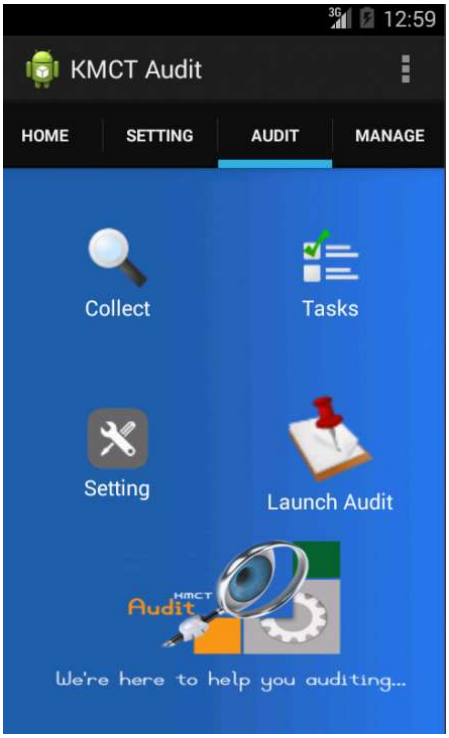

Figure 38. Interface des services "KMCT Audit"

III.4.1. Interface « Collect »

L'auditeur doit en premier lieu collecter des informations sur l'équipement qu'il souhaite auditer soit par un questionnaire soit par un fichier de configuration (voir Figure 39).

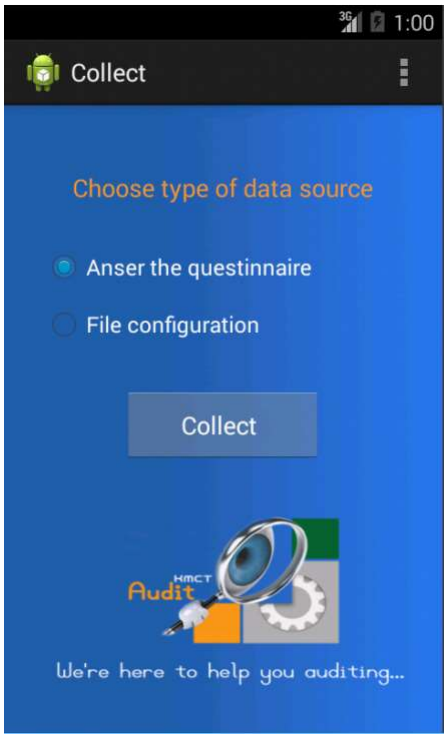

Figure 39. Interface "Collect"

Si l'auditeur choisi 'Répondre au questionnaire' comme source de donnée pareillement dans la Figure 40 ci-dessous, il répond à une liste des questions qui dépondent du type de l'équipement choisi à auditer. Ces questions sont retenues de la « Check List » de la norme ISO 27001. Les réponses à ces questions permettent d'identifier les failles.

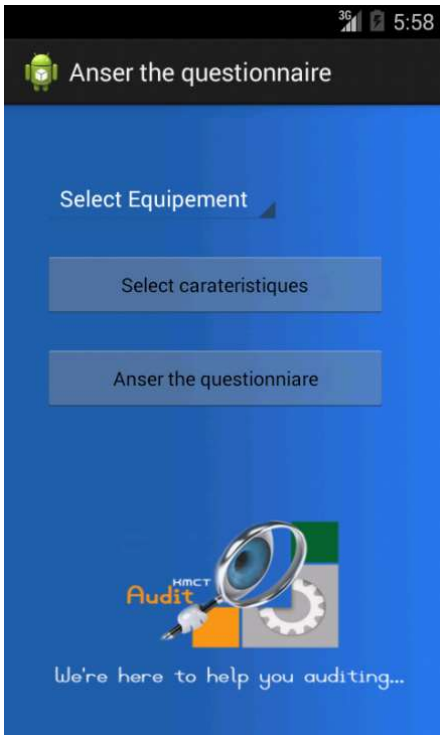

Figure 40. Interface "Anser the questionnaire"

L'utilisateur doit choisir le type d'équipement et ses caractéristiques qui permettent de simuler l'équipement en réel.

Les figures suivantes Figure 41 et Figure 42, illustrent le cas de sélection du type de l'équipement « Routeur ». En cliquant sur le bouton « Select Caracteristiques » la liste des caractéristiques du routeur sera affichée.

Figure 41. Interface "Select Equipment"

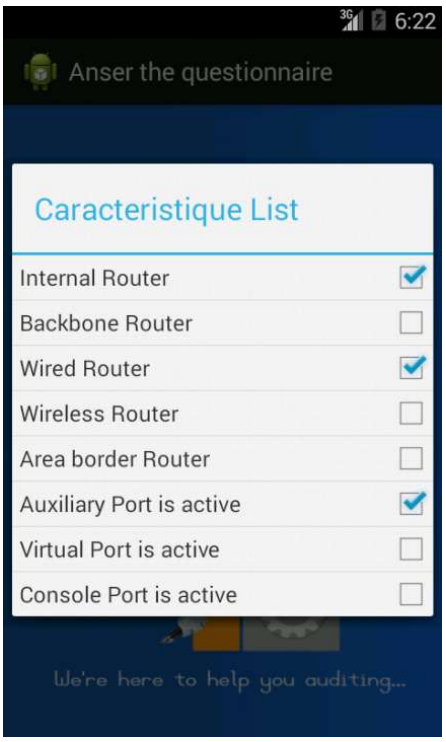

Figure 42. Interface " Caracteristique List"

Après le choix de l'équipement et la sélection des caractéristiques, l'auditeur répond au « Questionnaire » par 'Yes' ou 'No' pour chaque question (voir Figure 44 et Figure 45 ci-dessous).

Si l'auditeur choisi « Fichier de configuration » comme source de donnée, il parcourt la mémoire de son Smartphone afin de préciser le bon fichier qui sera exploité dans la phase de l'identification des failles.

Pour permettre l'utilisateur à accéder à la mémoire de téléphone, nous avons ajouté cette permission dans le fichier « AndroidManifest.xml » comme indiqué dans la Figure 43 suivante :

```
uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"
/uses-permission>
```

Figure 43. Permission d'accès à la mémoire de téléphone

Figure 44. Interface "Questionnaire" Page 1

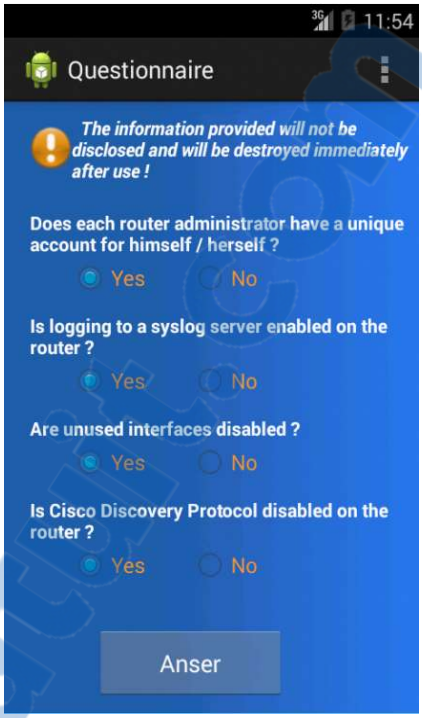

Figure 45. Interface "Questionnaire" Page 2

III.4.2. Interface « Tasks »

A travers l'interface « Tasks », exposée par la Figure 46 ci-dessous, l'utilisateur choisi les tâches qu'il a paramétré via l'interface « Audit Tasks » (voir Figure 37 ci-dessus).

Dans notre cas, nous avons déjà choisi d'utiliser le scanner de port « Nmap ».

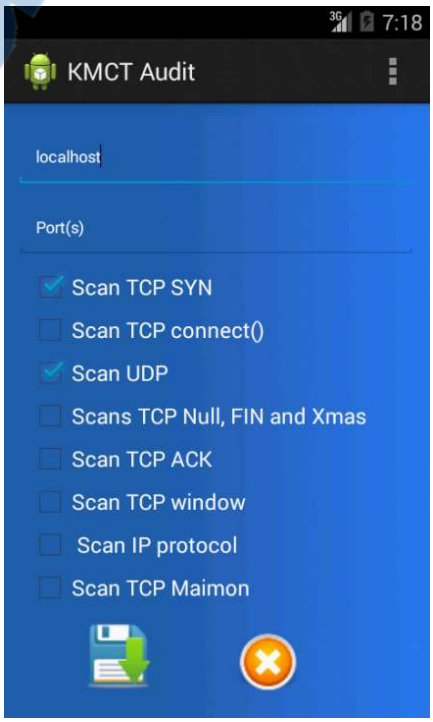

Figure 46. Interface "Tasks"

III.4.3. Interface « Setting Report »

Afin d'offrir une documentation lisible, l'auditeur peut choisir le format adéquat (PDF ou XML) du rapport d'audit selon son besoin ultérieur (voir Figure 47).

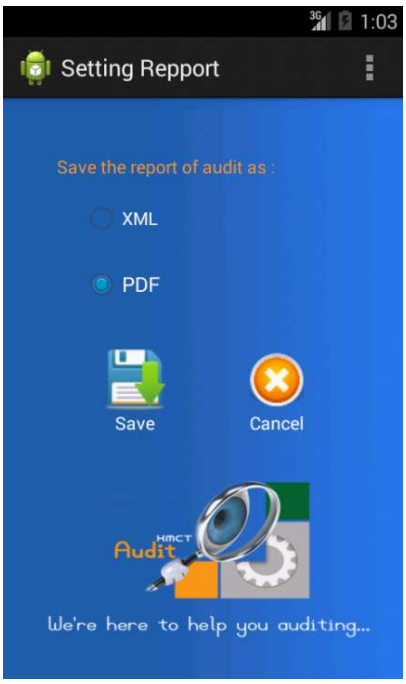

Figure 47. Interface "Setting Report"

III.4.4. Interface « Launch Audit »

Les trois phases que nous avons mentionnées ci-dessus sont exigées pour le lancement de l'audit d'un équipement réseau. La Figure 48 ci-dessous affiche tous les paramètres choisis.

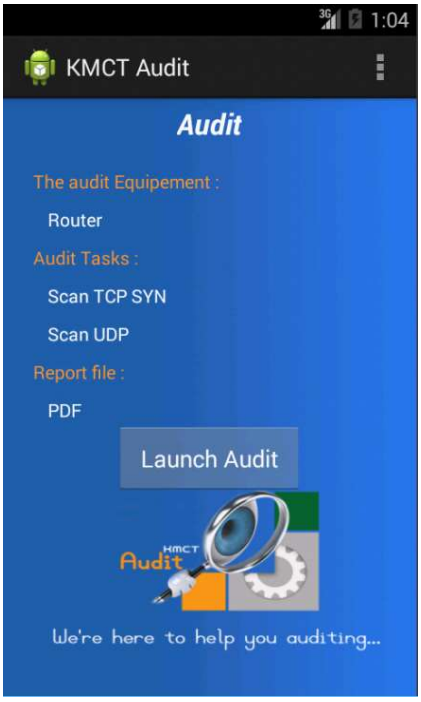

Figure 48. Interface "Launch Audit"

III.5. Interface « Audit Report »

Dans le cas de bon déroulement des tâches choisis précédemment, le rapport d'audit sera affiché en indiquant les vulnérabilités détectées avec les recommandations nécessaires (voir Figure 49 ci-dessous).

Figure 49. Interface "Audit Report"

A ce stade, l'auditeur peut enregistrer le rapport dans la mémoire de son Smartphone pour une utilisation ultérieure.

IV. Déploiement d'une application Android

La publication d'une application n'est que le commencement du cycle de vie de cette dernière car une fois en ligne, nous aurons besoin de la mettre à jour régulièrement, soit pour corriger des bogues notifiés par nos utilisateurs, soit pour proposer de nouvelles fonctionnalités ou adapter notre création aux différentes versions du SDK.

Avant de publier notre application sur le marché Android, il nous faut :

- S'inscrire et créer un compte développeur sur l'Android Market ;
- Payer un enregistrement de 25 dollars (environ 20 euros au moment de l'écriture de cet ouvrage) ;
- Signer la charte de distribution du développeur du marché Android ;

- Vérifier l'application et signer le fichier .apk [7];
- Et enfin publier l'application.

Pour en savoir plus sur les différentes étapes, voir Annexe A.

Conclusion

Ce dernier chapitre a présenté certaines parties de l'application mobile pour Android «KMCT Audit » qui constitue la partie réalisation de notre solution pour la plateforme Android avec les différentes interfaces des utilisateurs.

Il est vrai que, ce projet a répondu parfaitement aux objectifs fixés au niveau du cahier des charges mais toutefois et comme tout projet réalisé, il y a toujours des améliorations qui peuvent être rajoutée.

[7] **APK** : Android Package : est un format de fichier Android

Conclusion générale

Face à l'ampleur des risques qui menacent aujourd'hui les systèmes d'information des entreprises, les dirigeants sont devenus plus préoccupés par cette problématique et demandeurs de missions d'audit et de conseil dans ce domaine, un service qui permet d'auditer les équipements de n'importe où et à tout moment demeure utile voire nécessaire pour faciliter la mission d'audit.

C'est dans ce cadre que s'inscrit ce projet de fin d'études dans lequel il a été confié de concevoir et de réaliser un service d'audit de sécurité des équipements réseaux mobile pour Android intitulé « KMCT Audit » qui constitue une preuve de concept ou POC (en anglais : Proof Of Concept) correspondant à une démonstration de faisabilité d'un projet en cours de réalisation.

Après avoir fixé le cahier des charges et les spécifications, nous avons procédé à l'analyse suivie par la conception de l'application et cela selon la méthodologie 2TUP. Le présent rapport met en évidence le déroulement et le développement de ces étapes.

L'élaboration de ce projet nous a permis d'approfondir nos connaissances en informatique et surtout dans le domaine du développement mobile ainsi que dans le domaine de sécurité réseau. De même, il nous a donné l'occasion de manipuler des nouvelles technologies de développement, de consolider les technologies déjà acquises et d'acquérir une bonne expérience au niveau de la réalisation pratique.

De ce fait, nous avons eu l'occasion au cours de ce projet d'utiliser un ensemble diversifié de logiciels en particulier ; Android SDK pour l'implémentation de l'application cliente, la simulation sur des émulateurs Android, la construction des interfaces graphiques, Power Designer pour la conception technique de l'application et Adobe Photoshop pour la conception graphique des interfaces.

Pour conclure, l'expérience vécue était en adéquation avec nos attentes puisque les principaux objectifs qui ont été définis au préalable sont atteints et que nous avons réussi à mettre en œuvre et donner naissance à notre application « KMCT Audit ».

Toutefois, ce projet s'arrête à des améliorations significatives en matière de sécurité de données qui permet à l'administrateur de choisir le type de chiffrement : asymétrique, symétrique ou hybride.

Références

[1] Statista, «Android market share in the United Kingdom (UK) 2011-2016 (fee-based),» [En ligne]. Available: http://www.statista.com/.

[2] Statista, «App stores,» [En ligne]. Available: http://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/.

[3] TVCT, «About Us,» [En ligne]. Available: http://www.tvtc.gov.sa/English/AboutUs/Pages/default.aspx.

[4] A. RHARRAB, «« Audit Sécurité des Systèmes d'Information». Mémoire de Projet de Fin d'Etudes».

[5] D. L. Cannon, CISA, Certified Information Systems Auditor Study Guide, SYBEX, 2011.

[6] «Android- Wikipédia,» [En ligne]. Available: https://fr.wikipedia.org/wiki/Android.

[7] «IOS_(Apple)- Wikipédia,» [En ligne]. Available: https://fr.wikipedia.org/wiki/IOS_(Apple).

[8] «Windows Phone,» [En ligne]. Available: https://fr.wikipedia.org/wiki/Windows_Phone.

[9] «BlackBerry- Wikipédia,» [En ligne]. Available: https://fr.wikipedia.org/wiki/BlackBerry.

[10] N. Jaimes, «Chiffre d'affaires des applications mobiles dans le monde,» 2015. [En ligne]. Available: http://www.journaldunet.com/ebusiness/internet-mobile/1125256-chiffre-d-affaires-des-applications-mobiles-dans-le-monde/.

[11] «10 meilleurs scanner de vulnerabilite,» [En ligne]. Available: http://01howto.blogspot.com/2010/12/10-meilleurs-scanner-de-vulnerabilite.html.

[12] AD34, «LA GESTION DE PROJET : METHODES CLASSIQUES VS METHODES AGILES,» 2013. [En ligne]. Available: http://www.access-dev.com/access-dev/la-gestion-de-projet-methodes-classiques-vs-methodes-agiles/.

[13] «JEE- Wikipédia,» [En ligne]. Available: http://fr.wikipedia.org/wiki/J2EE.

[14] «Android SDK - Wikipédia,» [En ligne]. Available: https://fr.wikipedia.org/wiki/Android_SDK.

Annexes

A- Déploiement d'une application Android

L'Android Market

L'Android Market – le marché d'applications Android – permet aux développeurs du monde entier de mettre leurs applications à disposition des utilisateurs d'Android. Ceux-ci pourront ainsi télécharger, gratuitement ou moyennant une rétribution, n'importe quelle application déposée dans cet espace.

Comme nous le verrons, certaines applications ne sont disponibles que dans un ou plusieurs pays spécifiques. Le choix des pays dont les utilisateurs seront autorisés à télécharger votre application sera à faire lors de la publication.

S'inscrire en tant que développeur sur le marché Android

Ouvrir un compte développeur est la première chose à faire (les heureux possesseurs d'un tel compte peuvent passer directement à la partie suivante). Pour vous rendre sur le site officiel du marché Android et vous y inscrire, saisissez l'adresse http:// www.android.com/market dans votre navigateur favori. Vous obtiendrez une page du type :

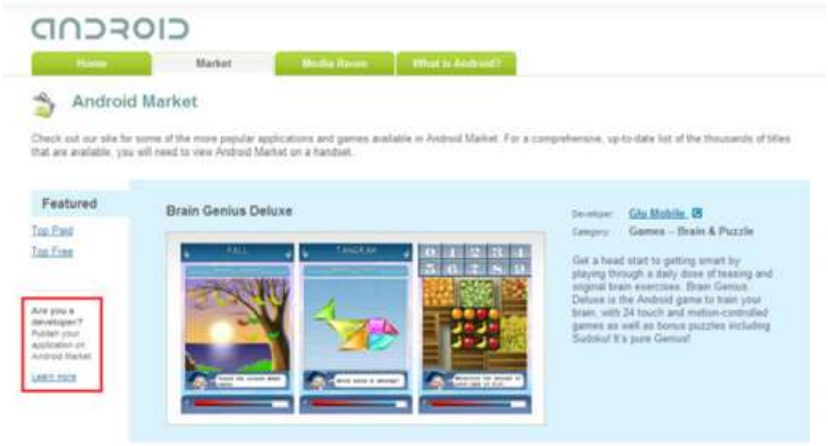

Figure A- 1. La page de l'android Market sur Internet

Pour accéder à l'espace développeur, cliquez sur *Learn More* dans le menu gauche du site ou saisissez directement l'adresse suivante dans votre navigateur : http:// market.android.com/publish/Home.

> **À NOTER Mise à jour des procédures d'inscription**
>
> Les procédures décrites dans la suite de ce chapitre peuvent être susceptibles d'évoluer en fonction des mises à jour des sites.

> **À SAVOIR Création obligatoire d'un compte Google pour s'inscrire sur le marché Android**
>
> L'inscription sur le marché Android nécessite d'avoir un compte Google. Si vous n'en avez pas, il vous faudra donc en créer un, soit depuis la page http://market.android.com/publish/Home (en cliquant sur *Create an account now*), soit depuis la page https://www.google.com/accounts/NewAccount (qui a l'avantage d'être dans la langue de votre navigateur).

Une fois muni de votre identifiant Google, vous pouvez vous authentifier sur la page d'inscription du marché Android. Si vous n'êtes pas encore inscrit en tant que développeur, la procédure d'inscription commencera automatiquement. La page d'inscription débute par la création du profil du développeur : nom, adresse mail, site internet et téléphone de contact. Saisissez bien tous les champs de façon à ce que l'on puisse vous contacter si un souci de paiement avec un autre utilisateur se produisait.

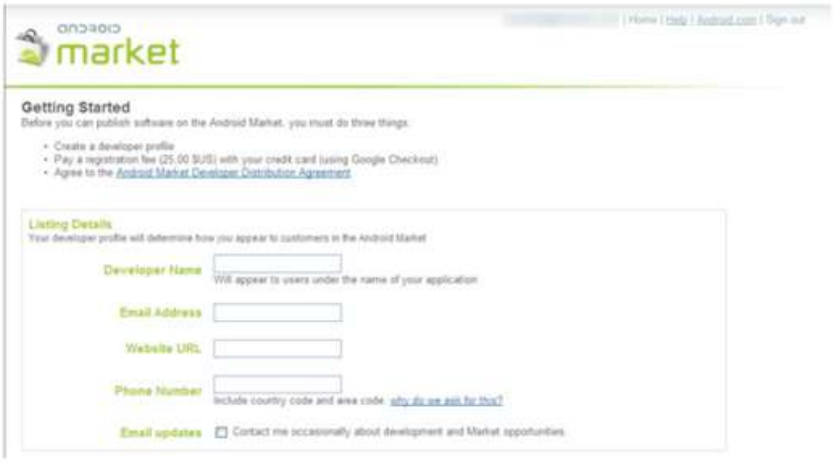

Figure A- 2. Création du profil du développeur

Une fois les informations de profil renseignées, cliquez sur *Continue* en bas de la page. L'assistant d'inscription vous proposera de payer les droits d'utilisation du service (fixé à 25 $ lors la rédaction de cet ouvrage)

Figure A- 3. Paiement des droits d'enregistrement

Cliquez sur *Continue* à nouveau pour arriver sur la page d'ajout d'une carte bancaire à votre compte Google Checkout et l'approbation de la charte de distribution du développeur du marché Android.

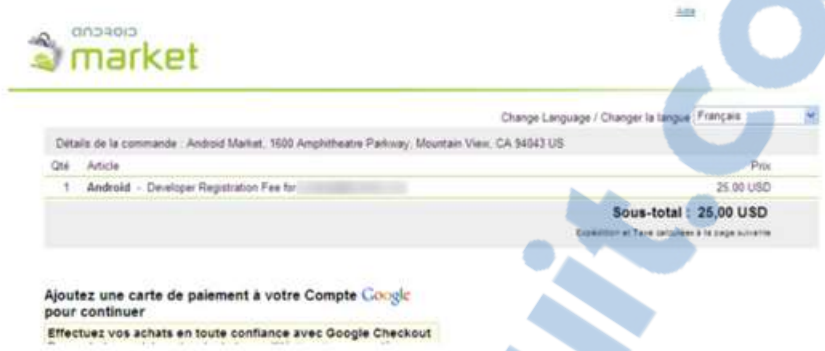

Figure A- 4. L'enregistrement d'un compte développeur coûte 25 $ géré avec Google Checkout

Une fois que vous avez rempli les informations de paiement, cliquez sur *Accepter et continuer* pour terminer le processus. Vous devriez recevoir les informations récapitulant l'ensemble du processus et votre facture dans votre boîte de messagerie.

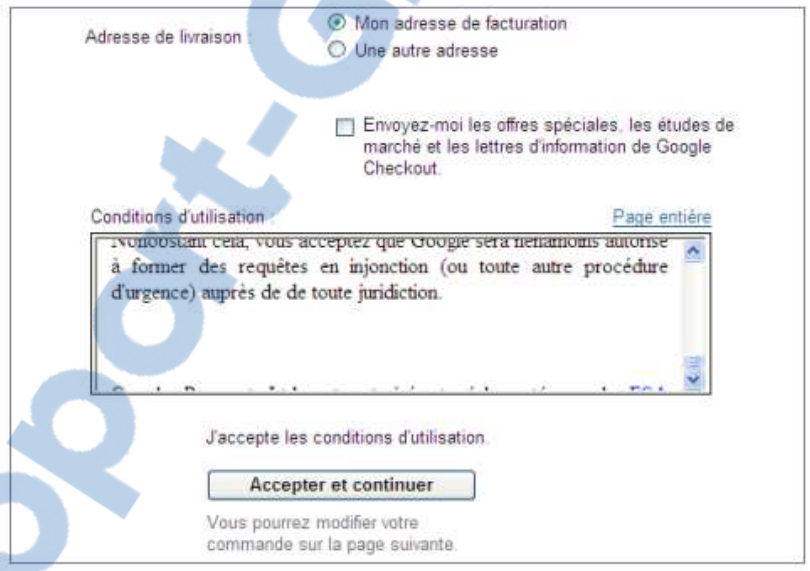

Figure A- 5. Acceptation de la licence d'utilisation pour devenir développeur

> **REMARQUE Interface anglophone du marché Android**
>
> À l'heure de l'écriture de cet ouvrage, l'interface proposée depuis le page d'accueil du marché Android n'est pas intégralement francisée. Vous devrez donc évoluer dans un interface écrite en anglais jusqu'aux premières étapes de votre inscription.

Félicitations, vous faites maintenant partie des développeurs Android et allez bientôt pouvoir rejoindre cette grande communauté et mêler vos talents de programmateur et de graphiste à ceux des autres.

Préparer son application pour la publication

La publication d'une application Android nécessite la création d'un paquetage – un fichier d'extension .apk contenant tout le code et les ressources de l'application – qui servira au système Android pour lire les propriétés et installer l'application sur le système. Pour garantir à vos utilisateurs une bonne installation et utilisation de votre application, ne succombez pas tout de suite à la précipitation en vous ruant sur le marché Android pour y publier votre application. Prenez le temps qu'il faut pour vérifier celle-ci et ne pas rater votre lancement !

Vérifier son application

En termes de méthodologie, publier une application Android à destination des autres utilisateurs n'est pas très différent de la publication d'autres types d'applications à destination des ordinateurs de bureau ou des serveurs. L'application devra d'abord être testée, et fournie sous la forme d'un paquetage avant d'être livrée sur le marché Android. Voici une liste des étapes les plus importantes à valider avant d'empaqueter votre application:

1. Testez votre application de façon exhaustive (tests unitaires, tests fonctionnels, tests de charge, etc.).
2. Ajoutez, si vous le pensez nécessaire, une licence d'utilisation de votre application.
3. Prenez le temps de créer une icône et d'internationaliser le titre (si votre application est disponible en plusieurs langues).
4. Retirez toutes les routines et informations de débogage (alertes, fenêtres de messages, sortie console, etc.).
5. Vérifiez tous les fichiers de ressources de l'application (fichiers d'internationalisation présents, images en résolution adéquate et de taille correcte pour un affichage optimal, etc.).
6. Indiquez les numéros de version de l'application et du code (voir plus loin).
7. Obtenez une clé privée adéquate pour signer l'application. Après avoir passé toutes ces étapes, vous pouvez enfin empaqueter l'application :
8. Signez l'application avec la clé privée précédemment obtenue.
9. Testez votre application signée dans l'émulateur.
10. Testez l'application sur un appareil réel !

N'oubliez surtout pas cette dernière étape indispensable avant de livrer votre application sur le marché Android ! Sitôt toutes ces étapes passées, vous serez fin prêt à publier votre application et peut-être à gagner un peu d'argent avec.

Ajouter des informations de version à une application

Une application Android est, comme toutes les applications informatiques, une création évolutive. À ce titre, elle possède un cycle de vie commençant par une première version, qui se transforme ensuite au fil du temps et des évolutions fonctionnelles et technologiques, créant ainsi de multiples versions. Que ce soit pour des raisons marketing, pour notifier l'ajout de corrections et/ou de nouvelles fonctionnalités ou pour suivre l'évolution de l'application par l'équipe de développement, vous devez spécifier la version de votre application. Un numéro de version d'une application permet notamment de faire le lien avec de potentielles dépendances vers d'autres briques logicielles et de permettre de gérer la compatibilité avec les différentes versions de la plate-forme Android. Une application Android possède donc, au même titre que d'autres plates-formes, une gestion des versions. Vous pouvez spécifier deux valeurs de version (*versionCode* et *versionName*) dans le manifeste de l'application : ces deux attributs doivent être renseignés avant de compiler votre code :

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    paquetage="com.eyrolles.android.demo" android:versionCode="3"
    android:versionName="2.5">
```

L'attribut *versionCode* est un nombre entier spécifiant la version de votre code. De façon générale, vous publierez votre application avec une version de code égale à 1, puis vous incrémenterez ce chiffre de 1 à chaque évolution, qu'elle soit majeure ou mineure. Ce numéro de version peut être consulté par d'autres applications de façon à vérifier la compatibilité, sans pour autant être affiché à l'utilisateur. Notez que la version du code n'a pas de lien direct avec la version de l'application – attribut *versionName*– qui est visible quant à elle par l'utilisateur. L'attribut *versionName* est une chaîne de caractères qui représente le numéro de version de l'application qui sera affiché à l'utilisateur. Cette chaîne permet de spécifier n'importe quel format de version. Généralement, les développeurs utilisent le formalisme *<majeur>.<mineur>.<révision>* pour spécifier un numéro de version. Vous pouvez également compléter celui-ci d'un qualificatif quant à la maturité de l'application, par exemple « 1.0.1 Beta ». Ce numéro de version est purement informatif et à destination des utilisateurs : celui-ci ne sera pas utilisé par le système et/ou les services de publication.

À chaque publication sur le marché Android, n'oubliez pas d'incrémenter les deux valeurs.

Compiler et signer une application avec ADT

Toutes les applications Android doivent être numériquement signées pour être installées sur le système Android. La signature, à l'aide d'un certificat contenant votre clé privée, permet d'identifier le développeur et de créer une relation de confiance entre celui-ci et le système. Le certificat n'a pas besoin d'être signé par une autorité certifiée ; un certificat auto-signé est accepté. Le système Android n'exécutera pas et n'installera pas une application non signée. Cette règle est tout aussi vraie dans le cadre du développement. En effet, pour pouvoir signer votre application, vous devez utiliser un certificat. Par défaut, lorsque le complément ADT compile un projet, il utilise un certificat généré automatiquement.

Afin de distribuer votre application sur le marché Android, ce certificat de débogage devra être remplacé par celui du développeur ou de l'entreprise. Toutes les applications d'un même développeur ou d'une même entreprise sont en général signées avec le même certificat. Il existe plusieurs façons de signer vos applications pour les publier sur le marché Android, les deux plus connues sont avec ADT et avec les outils standards *keystore/ jarsigner*. C'est avec ADT que nous allons procéder pour la suite de ce chapitre. La première étape pour signer votre application est de la compiler sans la signer. Ensuite vous devez créer ou récupérer une clé privée adéquate avec laquelle vous signerez l'application. Les utilisateurs d'Eclipse et d'ADT peuvent utiliser l'assistant d'export pour compiler l'application en la signant ou non avec une clé privée. Vous pouvez aussi générer un nouveau trousseau de clés et une nouvelle clé privée directement depuis cette interface. Pour compiler votre application et la signer en une seule action, effectuez un clic droit sur votre projet dans la fenêtre d'exploration de paquetage – vue Eclipse nommée *Package Explorer*– puis sélectionnez *Android Tools > Export Signed Application package*.

Figure A- 6. Exportez votre application et signez-la en une seule action grâce à ADT

L'assistant d'export et de signature de l'application vous propose alors de choisir une application Android dans votre espace de travail :

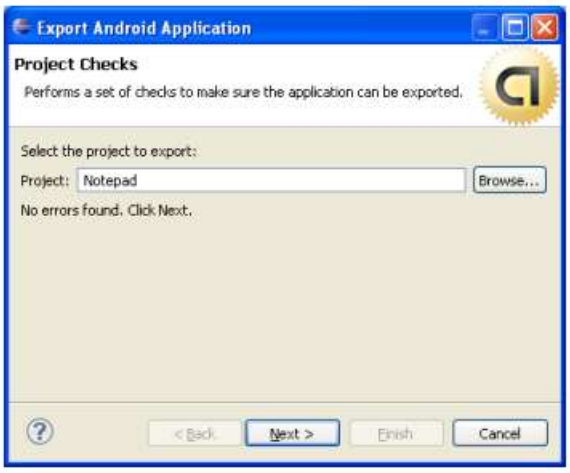

Figure A- 7. Sélectionnez le projet que vous désirez exporter, puis appuyez sur 'Next'

Si vous possédez déjà un fichier *.keystore* – que vous auriez généré avec l'outil keystore du SDK Java – contenant la clé privée avec laquelle vous souhaitez signer votre application, sélectionnez l'option *Use existing keystore*. Sinon créez un nouveau trousseau de clés en spécifiant son emplacement et le mot de passe de protection.

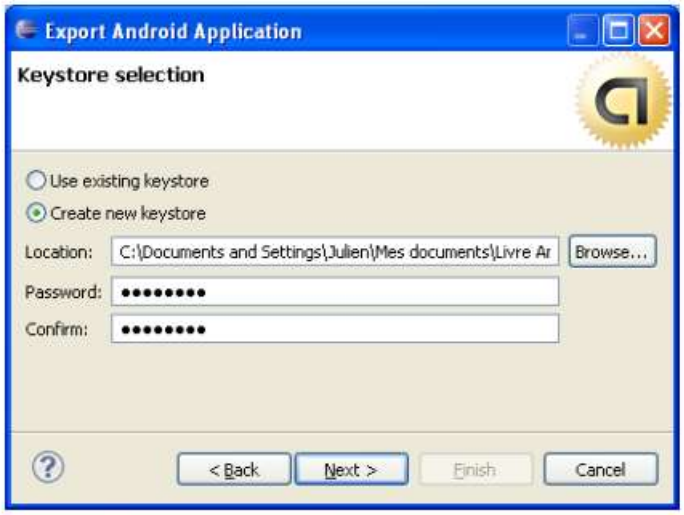

Figure A- 8. Création d'un nouveau trousseau de clés

Spécifiez ensuite l'alias de la clé à générer – de façon à pouvoir l'identifier par la suite – ainsi qu'un mot de passe de façon à protéger votre clé privée.

> À RETENIR **Notez votre mot de passe de clé !**
>
> Notez votre mot de passe ! Celui-ci vous sera redemandé à chaque fois que vous souhaiterez signer une application.

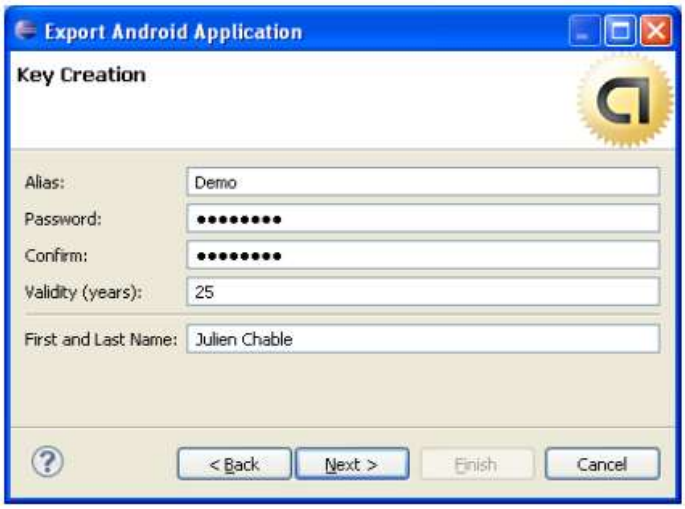

Figure A- 9. Création d'une clé valable 25 ans

Entrez toutes les informations sur vous ou votre entreprise pour cette clé. Ces informations seront stockées et permettront à vos utilisateurs d'identifier l'origine de l'application et de son auteur.

Figure A- 10. Les informations concernant votre clé sont à remplir rigoureusement

Pour compiler, générer votre clef et signer l'application, cliquez sur *Finish*.

Figure A- 11. Dernier écran de l'assistant permettant de générer une clé, de compiler et de signer l'application

Le fichier .apk résultant de cette opération n'est ni plus ni moins que votre application signée numériquement et prête à rejoindre les milliers d'applications du marché Android.

Publier son application sur le marché Android

Vous avez vérifié et signé votre application, tous les feux sont ainsi au vert pour publier votre application sur le marché Android.

Présentation de la console de publication des applications

Connectez-vous sur le marché Android (http://market.android.com/publish/Home). Une fois authentifié avec les identifiants que vous avez saisis lors de l'enregistrement, vous serez redirigé sur la console du développeur.

Figure A- 12. La console de développeur Android

Cette dernière offre une vue sur l'ensemble des développements déjà publiés ou en cours de publication. Vous pouvez également modifier votre profil (nom, adresse de messagerie, téléphone, etc.) si celui-ci a évolué depuis votre enregistrement sur le marché Android.

Source : http://www.cegepsherbrooke.qc.ca/~gagnonju/Fichiers255/Programmation_Android_-_De_la_conception_au_deploiement_avec_le_SDK_Google_Android_2.pdf