RÉPUBLIQUE TUNISIENNE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

UNIVERSITE VIRTUELLE DE TUNIS

Projet de fin d'études

Pour l'obtention du diplôme :

Master professionnel en Nouvelles Technologies des Télécommunications et Réseaux

Par
Aymen HEDIDAR

CONCEPTION ET REALISATION D'UNE APPLICATION MOBILE M-BANKING

Enseignant encadreur : Mr *Riadh BOUHOUCH,*
Encadreur à l'établissement : *Mr Mohamed Meddeb*

Année Universitaire 2011-2012

Dédicace

Mes premiers remerciements

sont adressés à ma famille et mes amis

qui m'ont supporté tout au long de ces longues journées passées à réaliser ce travail.

Remerciements

Mes premiers remerciements sont adressés à mes enseignants et enseignantes à L'UVT pour leur contribution à notre formation de master et je remercie particulièrement Mr Adnen CHERIF, coordinateur du master N2tr.

Ensuite, bien sûr, je tiens à remercier mon encadreur Monsieur Riadh BOUHOUCH, enseignant au sein de l'Université Virtuelle de Tunis, pour son encadrement, ses recommandations précieuses et sa disponibilité, ainsi que Mr Mohamed Meddeb qui m'a soutenu pour réaliser ce stage.

Enfin je tiens à remercier les membres du jury pour avoir assisté à cette soutenance.

Aymen HEDIDAR

Table des matières

Table des figures

Introduction générale

L'avantage compétitif des banques repose sur leur capacité à répondre rapidement aux changements des besoins du marché et augmenter leurs bénéfices en produisant des produits des biens ou des services qui satisfaits le besoin de la clientèle.

Dans un monde en constante évolution et un environnement concurrentiel, un avantage compétitif est de bénéficier de l'information et du service voulu, au moment et à l'endroit voulus.

La réponse *:* Le **m-banking** qui constitue une solution possible à cette situation.

On rencontre plusieurs définitions et divers acronymes qui s'intéressent aux services bancaires par mobiles. Le terme Mobile Banking désigne une spécificité du Mobile Business adaptée au métier de la Banque. C'est en quelque sorte une adaptation du désormais canal de banque à distance classique de « e-banking » sur terminal. Le e-banking est l'ensemble des services bancaires assurés par voie électronique « electronic banking » et donc par Internet : consultation de comptes, virements, achats de produits financiers.

Il est à noter que l'e-banking est un terme générique désignant tous les services assurées par internet sur support portable et non portable (ordinateur de bureau) par contre le m-banking Le mobile banking regroupe toutes les techniques permettant de réaliser des opérations bancaires à partir du support « téléphone portable ».

En fonction des capacités techniques (téléphone voix, données, internet, SMS, WAP) et de la diversité des terminaux mobiles (téléphone mobile, Pocket PC, PDA, **Smartphone**).

Plusieurs banques ont adopté des solutions de **Mobile Banking**, du fait de la meilleure couverture du réseau mobile par rapport au réseau filaire, et au nombre important de téléphones mobiles utilisés. On peut ou l'on trouve des solutions de paiement par SMS également.

Les banques ont exploité les potentialités des outils de communication en particulier la téléphonie mobile en un accès mobiles comme canal de distribution et comme service afin de concrétiser son objectif.

De nos jours, les banques offrent des services à distance via l'équipement du Smartphone grâce à sa richesse qui le rend idéal pour gérer plusieurs processus relatifs à la banque auxquels les clients d'une banque se connectent à leurs comptes bancaires par l'intermédiaire de leur téléphone portable aux services de paiement par téléphonie mobile, qui couvrent un plus grand nombre de services de paiement et permet aux banque d'entrer en relation avec leur clientèle et informer ses clients de nouveaux services. Pour les clients, les services bancaires par mobile représentent un équilibre difficile entre un concept au potentiel considérable (pouvoir faire des transactions n'importe où, n'importe quand) et des obstacles de nature pratique.

Les **Smartphones** permettent à la clientèle de la banque de simplifier le support clients, d'économiser et d'augmenter la satisfaction client.

Dans ce cadre, nous allons essayer de réaliser une application de mobile destinée pour les banques afin de constituer une interface entre le client et sa banque et offrir un ensemble de services pour la clientèle.

Pour atteindre cet objectif, nous commençons par comprendre le fonctionnement du système Mobile et son déploiement pour la mise en place de la banque à distance.

Dans un deuxième lieu, nous choisissons une architecture Client/serveur afin d'implémenter les différents modules logiciel de l'application. Ensuite, nous procédons à la conception. Et enfin, nous développons l'application.

Notre œuvre se subdivisera en quatre principaux chapitres :

Dans le premier chapitre intitulé Aperçu général et cadre général nous allons présenter l'entreprise d'accueil et notre application, et dans le deuxième chapitre qui a pour titre Analyse et spécification des besoins, nous commencerons par comprendre le contexte du système, déterminer les principaux cas d'utilisation, déterminer les besoins fonctionnels et les besoins non fonctionnels. Puis, dans le troisième chapitre relatif de ce travail conception, nous tenterons d'approfondir la compréhension du système et d'obtenir une spécification, une analyse et une conception détaillées des cas d'utilisation.

Au niveau du quatrième chapitre intitulé Réalisation, l'architecture étant stable, le produit s'apparente alors à l'application, satisfaisant les exigences des utilisateurs.

Finalement, nous terminons par une conclusion générale et quelques perspectives intéressantes concernant ce travail.

CHAPITRE I

Aperçu général et cadre du projet

INTRODUCTION

Dans ce chapitre, nous allons présenter l'organisme d'accueil Digital Engineering System, la société au sein de laquelle nous avons effectué notre projet de fin d'études. Ensuite la présentation du cadre du projet permettra de mieux comprendre le problème étudié et présenter le principe de fonctionnement des systèmes de mobile Smartphone et ses applications dans le domaine métier des banques.

Enfin, ce chapitre présentera la solution retenue qui sera détaillée plus loin dans le rapport.

I.1 PRESENTATION DE L'ENTREPRISE D'ACCEUIL

Digital Engineering System est spécialisé dans le développement de solutions clés en main des applications informatiques déployées dans un grand nombre d'institution. Ces solutions trouvent aujourd'hui dans plusieurs domaines : télécommunication, banque, transport, éducation,….

Face à un marché qui n'a pas cesser de proposer de nouvelle technologies Digissys et tenue de maintenir son avancée de technologique avec standard, pragmatisme et pérennité sur des valeurs sûres et innovantes.

En effet, la gestion de projet et services consulting en qualité de maîtrise d'œuvre, nous prenons en charge l'ensemble des tâches inhérentes à la conduite et à la réalisation de projets qui lui ont été confiées.

Bien que ses solutions répondent en standard à un large éventail de besoins, elle demeure toujours à l'écoute de ses clients afin de leur proposer si nécessaire la réalisation de solution sur mesure, au plus juste prix et dans les meilleurs délais.

Le développement spécifiques ou l'intégration de produits partenaires sont alors réalisés conformément aux cahiers des charges et cadencés en cycle de vie, depuis les spécifications jusqu'aux mises en service.

Ses solides bases informatiques entretenues par la culture du génie logiciel et de la veille technologique elle confère le recul nécessaire au conseil et à la conception de systèmes performants en parfaite adéquation avec les besoins. Notre organisation, nos moyens et les mesures que nous appliquons garantissent l'obtention d'un niveau de qualité requis à chaque échelon des projets.

I.2 CADRE DE PROJET

De plus en plus nombreuses sont les applications mobiles qui reposent sur une la plate-forme de mobilité en équipant les périphériques mobiles consommateur les plus récents avec la connectivité en temps réel et l'intégration aux systèmes service client des banque. C'est pourquoi la clientèle peut bénéficier des services bancaires en ligne.

Ces applicatifs réduisent les coûts et augmentent la satisfaction du client. En effet, le client n'a pas besoins de se déplacer à sa banque et attendre aux guichets pour bénéficier de service, c'est sa banque qui lui rend visite.

Et au niveau de la banque, abaissent les coûts de transaction et chargent de service, en particulier le coût de mise en place et de maintien d'un réseau de distribution pour les banques.

I.2 1 Le m-banking

Le Mobile Banking désigne une spécificité du Mobile Business adaptée au métier de la Banque. C'est en quelque sorte une adaptation du canal de banque à distance classique de « e-banking » sur terminal mobile. D'autre part Le e-banking est l'ensemble des services bancaires assurés par voie électronique « electronic banking » et donc par Internet :demander informations, commande chéquier, service bancaire, consultation de comptes, virements et achats de produits financiers.

De nos jours, les réseaux de connexion mobile permettent d'obtenir une connectivité plus sûre et moins coûteuse. En fonction des capacités techniques (téléphone voix, données, internet, SMS, WAP) et de la diversité des terminaux mobiles (téléphone mobile, Pocket PC, PDA, Smartphone). , **on peut classifier le Mobile Banking comme suit** :

Le PDA banking

Le SMS banking, le SMS Payment

Le WAP banking

Le Mobile Internet banking

Le Mobile Embedded Software banking

Le paiement par SMS est classé dans le Mobile Banking et non dans le Mobile Payment, car il ne se substitue par à une carte de paiement bancaire de type carte bleue ou carte VISA.

Les solutions de Mobile Banking peuvent servir à consulter les informations personnelles d'un client au travers d'un mobile. Elles ont ensuite permis d'effectuer des transactions bancaires de type virement et achat/revente d'actions. C'est le Mobile Banking transactionnel.

Les applications mobiles « toujours disponibles et toujours connectées » vous fournissent une communication très personnalisée et ciblée par l'envoie automatique des notifications à votre clientèle. Vos applications de gestion de comptes peuvent ainsi transmettre toute la gamme d'informations, du moment d'une panne locale jusqu'aux estimations de temps de résolution. Avec l'accès automatique au support critique, vous gagnez du temps en évitant des appels inutiles.

I.2.2 Les techniques de m-banking

Réseaux mobiles et transmission de données les réseaux mobiles disponibles utilisent la technologie GSM et depuis UMTS). Ces normes internationales permettent aux équipements mobiles de bien sûr téléphoner, mais aussi d'effectuer des transmissions de données.

Les techniques disponibles pour la transmission de données sont:

• l'échange de données par SMS : bien adapté à la transmission de données textuelles de petite taille.

• l'établissement d'une liaison TCP/IP qui selon la technologie (GSM, UMTS) et les options d'abonnement peut être à différentes vitesses:

- CSD (circuit switched data), proche d'un protocole modem traditionnel: permet l'échange de données mais interdit les appels téléphoniques durant la connexion en mode data.

- GPRS (General Packet Radio Service): permet l'échange de données à
- 64 kbps (dans la pratique plutôt 40 kbps) tout en restant disponible pour recevoir un appel: néanmoins, l'échange simultané de données et la téléphonie n'est pas possible.
- UMTS: échange de données à haute vitesse (de l'ordre de 300 kbps).
- Sur la liaison TCP/IP ainsi établie, les données peuvent être échangées selon tous types de protocoles:
- le WAP, version simplifiée et plus légère du Web, dont le navigateur Wap est embarqué dans la plupart des portables commercialisés actuellement.
- le Web, embarqué dans les portables haut de gamme.
- Email (SMTP), également disponible sur les portables haut de gamme.
- échange de messages multimédia MMS permettant la transmission de messages composites intégrant texte, photo, sons, vidéo, etc.
- toute application spécifique, qui peut être soit embarquée sur le portable (par exemple sous forme d'application Java J2ME téléchargée

Si le portable le supporte ou sur un autre équipement (PC, Palm, etc.) utilisant le portable pour accéder au réseau.

Ces différentes techniques permettent de réaliser de nombreuses applications mobiles professionnelles dans le domaine bancaire offrir les fonctionnalités suivantes en temps réel:

Authentification de l'utilisateur, mise à disposition des informations utiles aux clients concernant la banque, services à la clientèle des outils liés à la banque, synthèse des comptes bancaires de type chèque, carte bancaire, placements liste des opérations d'un compte, virements internes et externes, services professionnels à forte valeur ajoutée, services destinés aux opérateurs télécoms mobiles et intégration au système d'information bancaire : messagerie, bases de données

Les Smartphones puissants actuels vous permettent à étendre vos outils de service client en ligne aux périphériques mobiles. Au fait, plusieurs processus sont rationalisés grâce à la mobilité, car les applications peuvent identifier et localiser automatiquement le client, en lui fournissant toutes autres données pertinentes.

I.2.3 Solution retenue

L'application envisagée, dans le secteur bancaire est une application mobile client riche serveur sur une plateforme **Android.**

Cette application permet à un client d'accéder à partir d'un terminal mobile pour bénéficier des services informationnels et services transactionnels sur sa banque en toute sécurité après vérification des paramètres d'accès au niveau serveur distant.

Le réseau internet via la suite des protocoles TCP/IP permet la connexion entre le terminal mobile et serveur d'application et la transmission de message se fait par le langage JSON qui constitue un moyen de communication. Le terminal mobile envoi les informations afin s'authentifier et une demande chèque de au serveur qui décode ces données et les analyse pour décision.

Android est un système d'exploitation open-source pour smartphone conçu par Google. Le SDK fournit les outils et l'API nécessaires pour commencer à développer sur mobile sur la plateforme Android en Java. Le développement mobile ouvre de larges perspectives. En effet, les mobiles possèdent maintenant des accéléromètres, des connexions sans-fil et des GPS. Ce framework était nouveau pour nous et c'était très enrichissant d'apprendre son fonctionnement. Avec ce projet, j'ai appris comment utiliser une architecture 3-tier et les webservices. Cette architecture divise l'application en trois parties. Le client (téléphone) se connecte à un serveur (Middleware) via des webservices et ce serveur interroge la base de données.

Figure 1 Architecture de la solution proposée

CONCLUSION

Ce chapitre introductif nous a permis de détailler le cadre général du système. Et ce en présentant l'entreprise d'accueil, l'application que nous allons concevoir et développer, ainsi que de présenter les objectifs visés. Dans ce qui suit nous allons entamer la première phase de la conception de notre projet « Analyse et spécification de besoin » pour identifier les différentes fonctionnalités de l'application.

CHAPITRE II

Analyse et spécification des besoins

INTRODUCTION

Une étape essentielle de tout cycle de développement logiciel ou conceptuel consiste à effectuer une étude préalable. Le but de cette phase est de comprendre le contexte du système. Il s'agit d'éclaircir au mieux les besoins fonctionnels et non fonctionnels, apparaitre les acteurs et identifier les cas d'utilisation.

Dans ce chapitre, nous allons essayer d'exprimer les besoins sous forme de diagrammes de cas d'utilisation.

II.1 SPECIFICATION DES BESOINS

La spécification de besoins constitue la phase de départ de toute application à développer dans laquelle nous allons identifier les besoins de notre application. Nous distinguons des besoins fonctionnels qui présentent les fonctionnalités attendues de notre application et les besoins non fonctionnels pour éviter le développement d'une application non satisfaisante ainsi de trouver un accord commun entre les spécialistes et les utilisateurs pour réussir le projet.

I.1.1 Spécification des besoins fonctionnels

Après une étude détaillée de système, cette partie est réservée à la description des exigences fonctionnelles des différents acteurs de l'application. Ces besoins se regroupent dans les diagrammes des cas d'utilisation.

Les besoins utilisateur :

- L'authentification de client.
- Cours devise.
- Change devise.
- Consultation annuaire réseau des agences
- Commande chéquier.
- Contact banque par mail ou par téléphone.

I.1.2 Spécification des besoins non fonctionnels

Les besoins non fonctionnels décrivent toutes les contraintes techniques, ergonomiques et esthétiques auxquelles est soumis le système pour sa réalisation et pour son bon fonctionnement. Et ce qui concerne notre application, nous avons dégagé le besoins suivants :

- La disponibilité : l'application doit être disponible pour être utilisé par n'importe quel utilisateur.
- La sécurité de l'accès aux informations critiques : nous devons prendre en considération la confidentialité des données de clients surtout au niveau de l'authentification. Pour cela nous devons restreindre l'accès à ces informations à l'administrateur.
- La fiabilité : les données fournies par l'application doivent être fiables.
- La convivialité de l'interface graphique : l'application doit fournir une interface conviviale et simple pour tout type d'utilisateur car elle présente le premier contact de l'utilisateur avec l'application et par le biais de celle-ci on découvrira ses fonctionnalités.
- Une solution ouverte et évoluée : l'application peut être améliorée par l'ajout d'autres modules pour garantir la souplesse, l'évolutivité et l'ouverture de la solution.
- La possibilité de retourner au menu principal de l'application à partir de n'importe quelle fenêtre de celle-ci.

II.2 METHODOLOGIE ET APPROCHE ADOPTEE

Avant de programmer l'application et se lancer dans l'écriture du code : il faut tout d'abord organiser les idées, les documenter, puis organiser la réalisation en définissant les modules et les étapes de la réalisation. Cette démarche antérieure à l'écriture que l'on appelle modélisation ; son produit et un module.

La modélisation consiste à créer une représentation virtuelle d'une réalité de telle façon à faire ressortir les points auxquels on s'intéresse. Dans le cadre de notre projet on a utilisé la méthodologie UML pour la modélisation des différents diagrammes.

II.2.1 Présentation d'UML

En regardant les objectifs fixés pour la réalisation du projet, nous remarquons que nous sommes face à une application modulaire et qui devra rester ouverte pour les améliorations futures. De ce fait, il est très important d'utiliser un langage universel pour la modélisation afin de clarifier la conception et de faciliter les échanges. Notre choix est porté sur le langage

UML puisqu'il convient pour toutes les méthodes objet et se prête bien à la représentation de l'architecture du système.

UML :

UML (Unified Modeling Language) est un langage de modélisation unifié permet de modéliser une application logicielle d'une façon standard dans le cadre de conception orienté objet.

UML permet de couvrir le cycle de vie d'un logiciel depuis la spécification des besoins jusqu'au codage en offrant plusieurs moyens de description et de modélisation des acteurs et de utilisation système, du comportement des objets, du flot de contrôle internes aux opérations, des composants d'implémentation et leurs relations, de la structure matérielle et de la distribution des objets et des composants indépendamment des techniques d'implémentation et peut être mis à jour selon les besoins.

II.2.2 Les avantages d'UML

- Universel.
- Adopté par les grandes entreprises.
- Notation unifié
- Facile à comprendre.
- Adopté par plusieurs processus de développement
- Limite les risques d'erreur.
- N'est pas limité au domaine informatique.

II.3 LES DIAGRAMMES DE CAS D'UTILISATION

Le diagramme de cas d'utilisation a pour but de donner une vision globale sur les interfaces de future application. C'est le premier diagramme UML constitué d'un ensemble d'acteurs qui agit sur des cas d'utilisation et qui décrit sous la forme d'actions et des réactions, le comportement d'un système du point de vue utilisateur.

Acteur : un acteur est un utilisateur qui communique et interagit avec les cas d'utilisation du système. C'est une entité ayant un comportement comme une personne, système ou une entreprise.

Système : cet élément fixe les limites du système en relation avec les acteurs qui l'utilisent (en dehors de système) et les fonctions qu'il doit fournir (à l'intérieur du système).

Cas d'utilisation : un cas d'utilisation représente un ensemble de séquences d'actions à réaliser par le système et produisant un résultat observable intéressant pour un acteur particulier représenté par des ellipses et limité par un rectangle pour représenter le système.

II.3.1 Identification des acteurs

Acteur	Rôle
Client banque (utilisateur mobile)	- L'authentification de client. - Cours devise. - Change devise. - Consultation annuaire réseau des agences - L'envoi de demande chéquier. - Contact banque par mail ou téléphone.

Les acteurs de notre système

II.3.2 Diagramme de cas d'utilisation globale

Ci-dessous, nous présentons le diagramme de cas d'utilisation pour la compréhension du fonctionnement du système.

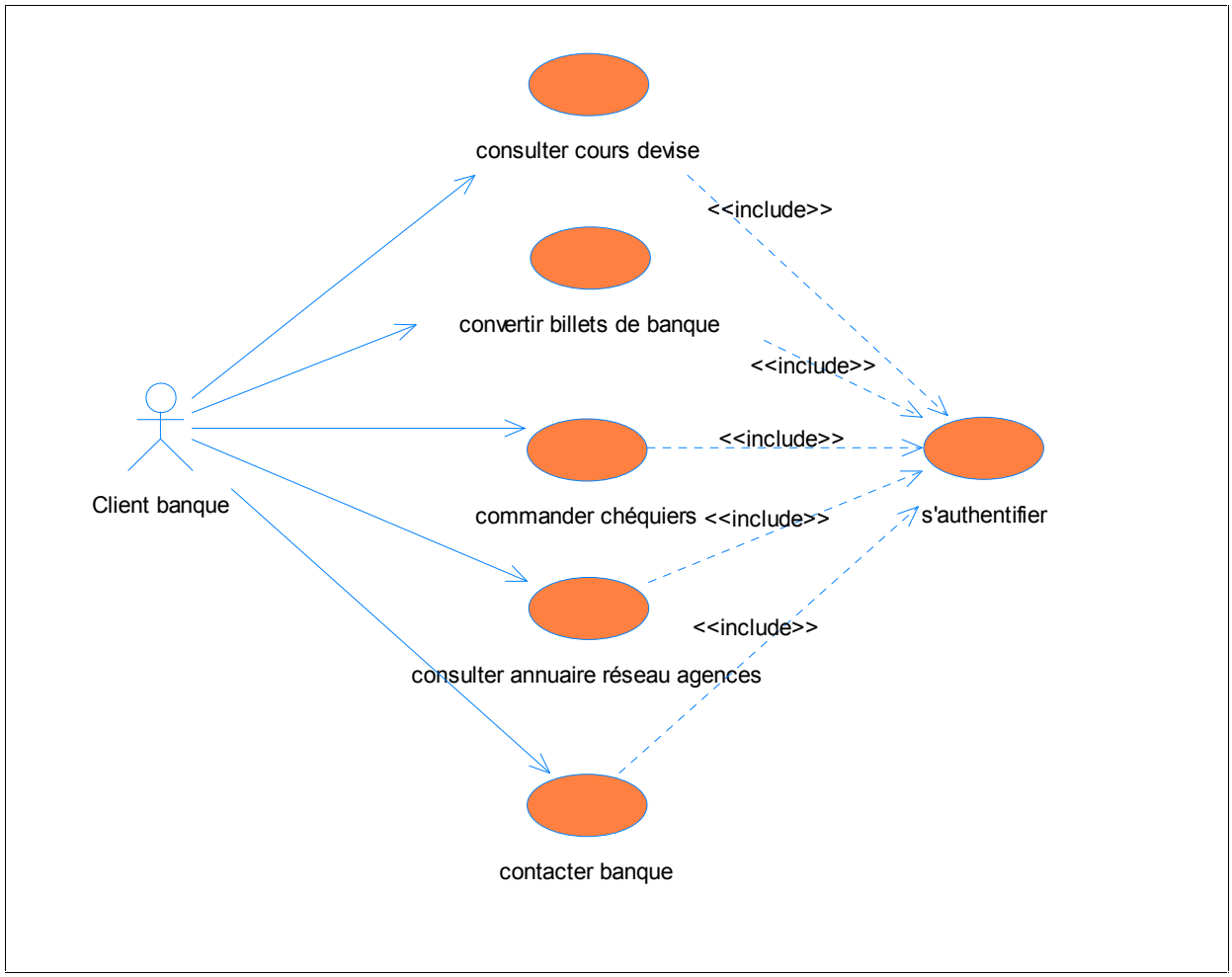

Figure 2 Diagramme de cas d'utilisation globale

Description du cas d'utilisation « **consulter cours devises** »

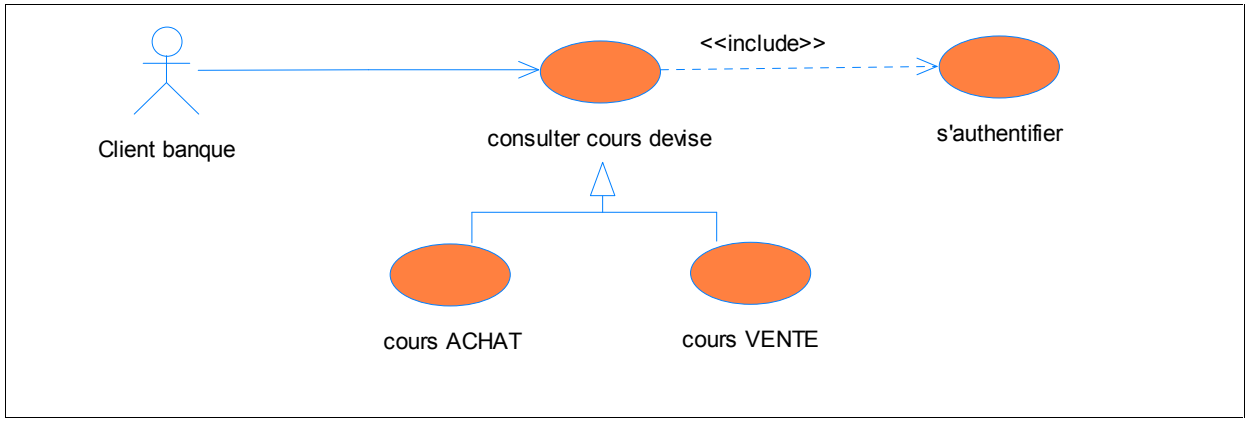

Figure 3 Diagramme de cas d'utilisation «consulter cours devise»

SOMMAIRE	
Titre :	**Consulter cours devise**
But :	Lister le cours de chaque devise par type opération Achat ou vente.
Résumé :	L'utilisateur clique sur le bouton « cours devise » l'action se déclenche et le système liste cours de vente ou achat.
Acteur :	Client banque
DESCRIPTION DES ENCHAINEMENTS	
Pré conditions	Post conditions
- client est authentifié	- affichage cours devises par sens opération ACHAT ou VENTE.
Scénario nominal	
1. Le client se connecte au système de la banque à travers son application mobile par login et un mot de passe. 2. Le client valide la requête d'affichage du cours de change via le bouton correspondant celui de « cours devises » 3. Le système expose l'affichage cours devises correspondants aux opérations d'achat ou vente. 4. Le client à la possibilité de basculer entre les opérations achat/vente.	
Enchaînement alternatif	
1.a Client n'a pas rempli champ ou les données sont incorrectes. 1. Le système affiche un message d'erreur. 2. Retour à l'étape 1 du scénario nominal pour lancer à nouveau la connexion.	

Tableau 1 : Fiche de description du cas d'utilisation : **Consulter cours devises**

Description du cas d'utilisation **Convertir billet de banque**

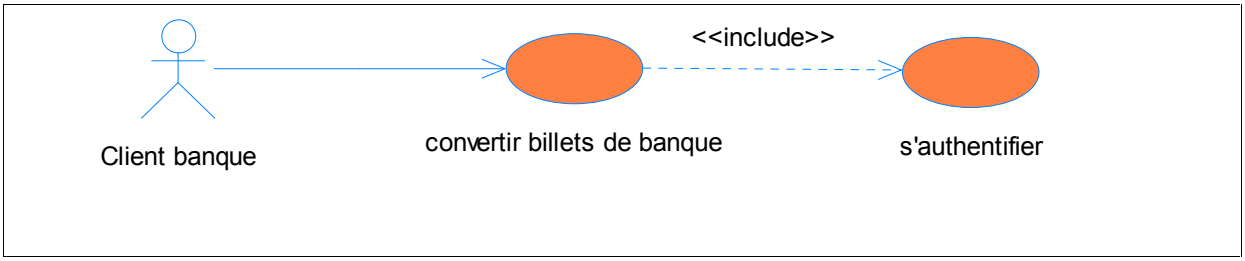

Figure 4 Diagramme de cas d'utilisation « convertir billet de banque »

SOMMAIRE	
Titre :	**Convertir billet de banque**
But :	Le client de banque peut convertir deux devises différentes. « Change devise »
Résumé :	L'utilisateur choisit la devise à convertir et la devise cible.
Acteur :	Client banque
DESCRIPTION DES ENCHAINEMENTS	
Pré conditions	Post conditions
- L'utilisateur est authentifié.	- Affichage le résultat de conversion devise.
Scénario nominal	
1. L'utilisateur se connecte à l'application à par login et un mot de passe. 2. Le client clique sur l'icône « change devise » 3. Le système présentera une page permettant de choisir la devise à convertir et la devise cible. 4. L'utilisateur choisit les deux devises et valide la conversion. 5. Système appelle un service web distant afin de récupérer le résultat de la fonction de conversion. 6. Le système affiche résultat conversion.	
Enchaînement alternatif	
1.a Client n'a pas rempli champ ou les données sont incorrectes. 1. Le système affiche un message d'erreur. 2. Retour à l'étape 1 du scénario nominal pour lancer à nouveau la connexion. 6.a Client n'a pas précisé les devises en entrées 1. Le système affiche un message d'erreur. 2. Retour à l'étape 1 du scénario nominal pour lancer à nouveau pour choisir les devises et valider la conversion.	

Tableau 2 : Fiche de description du cas d'utilisation : **Convertir billet de banque**

Description du cas d'utilisation **«« commander chéquiers»**

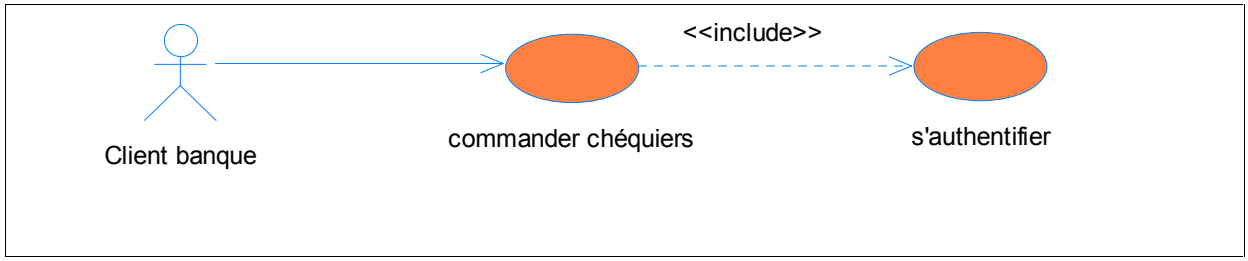

Figure 5 Diagramme de cas d'utilisation « commander chéquiers»

SOMMAIRE	
Titre :	**Commander chéquiers**
But :	Le client peut commander un chéquiers.
Résumé :	Commander chéquiers
Acteur :	Client banque
DESCRIPTION DES ENCHAINEMENTS	
Pré conditions	Post conditions
- Le client est authentifié. - Consultation compte utilisateur.	- Message de confirmation de prise en charge de la demande client du chéquier.
Scénario nominal	
1. Le client se connecte à système de la banque à travers son application mobile par login et un mot de passe. 2. Le système présente une page permettant de connecter au menu de l'application. 3. L'utilisateur lance la commande de chéquiers via l'icône « commander chéquiers » 4. Système effectue la mise à jour à la base de données du compte utilisateur et affiche message de succès.	
Enchaînement alternatif	
1.a Le client n'a pas rempli champ ou les données sont incorrectes. 1. Le système affiche un message d'erreur. 2. Retour à l'étape 1 du scénario nominal pour lancer à nouveau la connexion.	

Tableau 3 : Fiche de description du cas d'utilisation **: Commander chéquiers**

Description du cas d'utilisation **«Consulter annuaire réseau agences»**

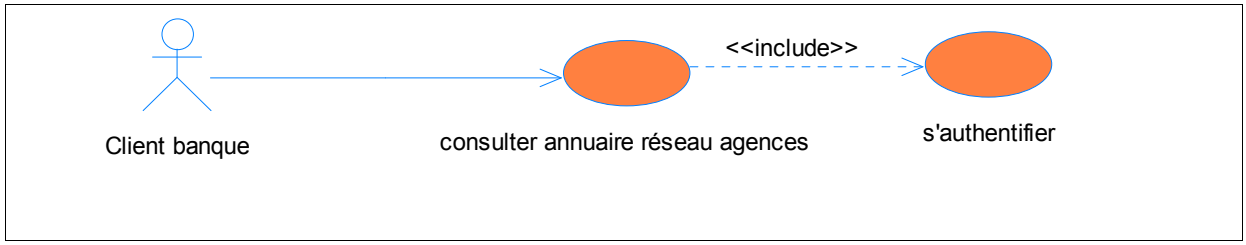

Figure 6 Diagramme de cas d'utilisation « Consulter annuaire réseau agences »

SOMMAIRE	
Titre :	**Consulter annuaire réseau agences**
But :	Le client peut consulter la liste des agences et leurs coordonnées.
Résumé :	Exposer la liste des agences commerciales de la banque.
Acteur :	Client.
DESCRIPTION DES ENCHAINEMENTS	
Pré conditions	Post conditions
- Le client est authentifié.	- Affichage liste des agences avec leurs cordonnées.
Scénario nominal	
1. Le client se connecte à l'application par login et un mot de passe. 2. Le système affiche le menu de l'application. 3. Le client clique sur l'icône « Réseau agences » 4. Le système présente une page permettant de consulter la liste des agences.	
Enchaînement alternatif	
1.a le client n'a pas rempli champ ou les données sont incorrectes. 1. Le système affiche un message d'erreur. 2. Retour à l'étape 1 du scénario nominal pour lancer à nouveau la connexion.	

Tableau 4 : Fiche de description du cas d'utilisation : **Consulter annuaire réseau agences**

Description du cas d'utilisation **contacter banque**

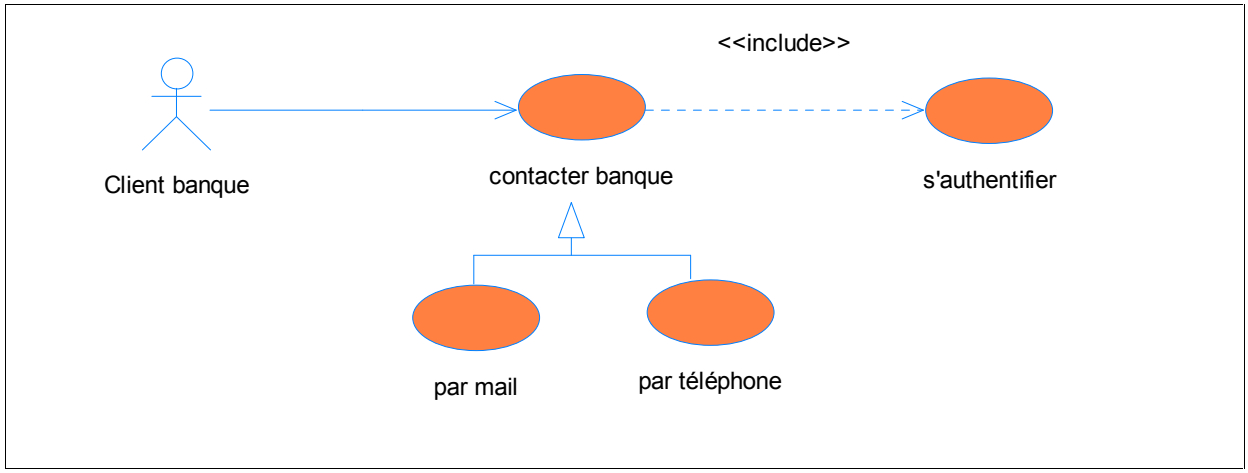

Figure 7 Diagramme de cas d'utilisation «contacter banque»

SOMMAIRE	
Titre :	**Contacter banque**
But :	Il permet consulter les coordonnées de la banque et contacter la banque par mail ou par téléphone.
Résumé :	Consulter coordonnées de la banque. Contacter banque par téléphone ou appeler banque.
Acteur :	Client banque
DESCRIPTION DES ENCHAINEMENTS	
Pré conditions	Post conditions
- Le client est authentifié.	- Présenter coordonnées de la banque et possibilité de le joindre par mail ou par téléphone.
Scénario nominal	
1. Le client se connecte à système de la banque à travers son application mobile par login et un mot de passe. 2. Le système affiche le menu de l'application. 3. Le client clique sur l'icône « Réseau agences » 4. Le système présente une page permettant de contacter sa banque. 5. .le client peur envoyer un mail à la banque ou appeler le service clientèle de sa banque.	
Enchaînement alternatif	

Tableau 5 : Fiche de description du cas d'utilisation : **Contacter banque**

CONCLUSION

Après avoir décrit les besoins fonctionnels et techniques attendus de notre application qui consistent à mettre en place une démarche de développement. Lors de cette dernière phase nous avons essayé d'exprimer le fonctionnement de notre système en se basant principalement sur les diagrammes de cas d'utilisations. Nous pouvons ainsi entamer la prochaine étape qui consiste à présenter la phase de conception.

CHAPITRE III

Conception

INTRODUCTION

Après avoir tracé les grandes lignes de phase de spécification de besoins, mettons l'accent maintenant sur une phase fondamentale dans le cycle de vie d'un logiciel, la phase de conception. Cette phase a pour objectif de déduire la spécification de l'architecture de système.

En premier lieu, la méthodologie de conception sera présentée, l'organisation des sections suivantes de ce chapitre suivra alors la logique de cette technologie.

Cette phase aboutira à la conception et la représentation des diagrammes de séquences et d'activités en se basant sur le langage de modélisation UML.

III.1 LA CONCEPTION GENERALE

III.1.1 Le cycle de développement en v

De nos jours, la méthodologie adoptée dans l'analyse et la conception des systèmes représente un choix stratégique pour le bureau d'études afin de mener à terme les projets tout en respectant les délais annoncés au client et avec la qualité demandée.

Vu l'évolution des besoins des utilisateurs finaux, les applications d'entreprise deviennent de plus en plus complexes et difficiles à con concevoir et à développer.

Pour la conception, le développement et la réalisation de notre application, nous avons opté pour l'application du processus de développement V qui demeure actuellement le cycle de vie le plus connu et certainement le plus convenable aux projets complexes.

Ce processus nous a accompagné du début de projet jusqu'à l'implémentation. Son principe est qu'avec toute décomposition doit être décrite la recomposition, et que toute description d'un composant doit être accompagnée de test qui permettront de s'assurer qu'il correspond à sa description. Ceci rend explicite la préparation des dernières phases (validation-vérification)

par les premières (construction du l'application) et on sait progressivement si on s'approche de ce que le client désire.

Le schéma ci-dessous représente les différentes phases du modèle en V :

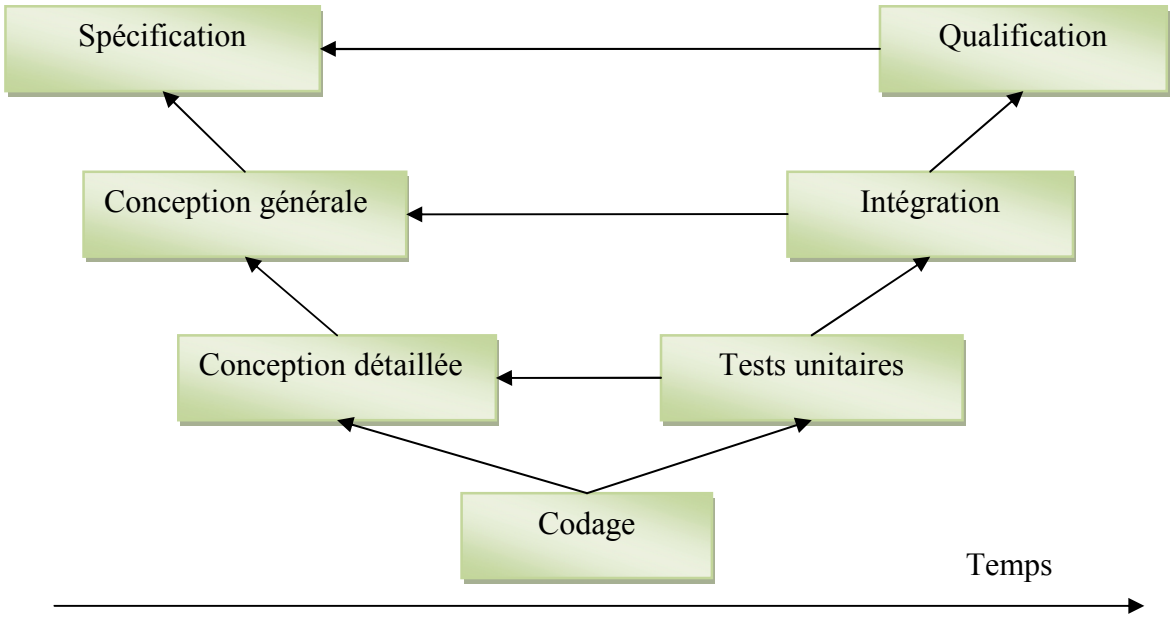

Figure 8 Modèle en v

III.2 LA CONCEPTION DETAILLEE

La conception détaillée met en œuvre itérativement un micro-processus de construction et c'est en cette phase que l'on génère le plus de volume d'informations.

En tant que concepteurs, nous allons élaborer le modèle de conception qui va donner une image « prête à coder » de notre solution.

Cette étape se fera par étape afin d'aboutir à un système fonctionnel reflétant une réalité physique.

III.2.1 Le diagramme de déploiement

Le diagramme de déploiement définit l'architecture matérielle de l'application. Il présente les périphériques utilisés et la répartition du système sur ces différents éléments. Il montre aussi les liens de communication entre ces diverses entités.

Le diagramme de déploiement de notre application est représenté par le diagramme ci-après :

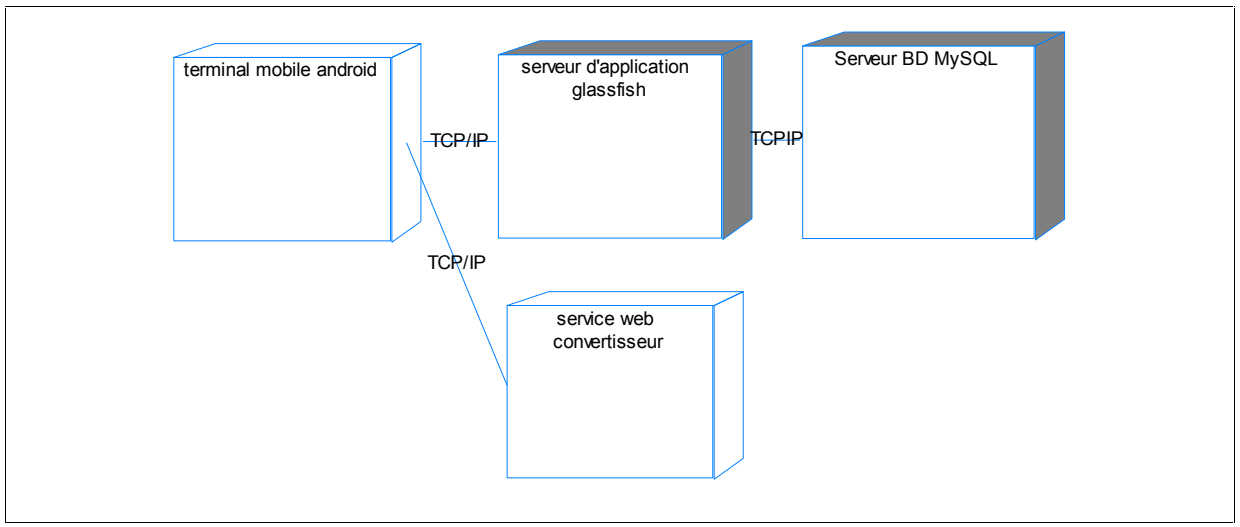

Figure 9 Diagramme de déploiement

III.2.2 les diagrammes de séquence

Les diagrammes de séquence peuvent servir à illustrer les cas d'utilisations décrits dans le chapitre précédent. Ils permettent de représenter la succession chronologique des opérations réalisées par un acteur et qui font passer d'un objet à un autre pour représenter un scénario.

Dans cette partie, nous allons décrire les scénarios les plus importants ainsi que leurs représentations par les diagrammes de séquence

III.2.2.1 le diagramme de séquence « s'authentifier »

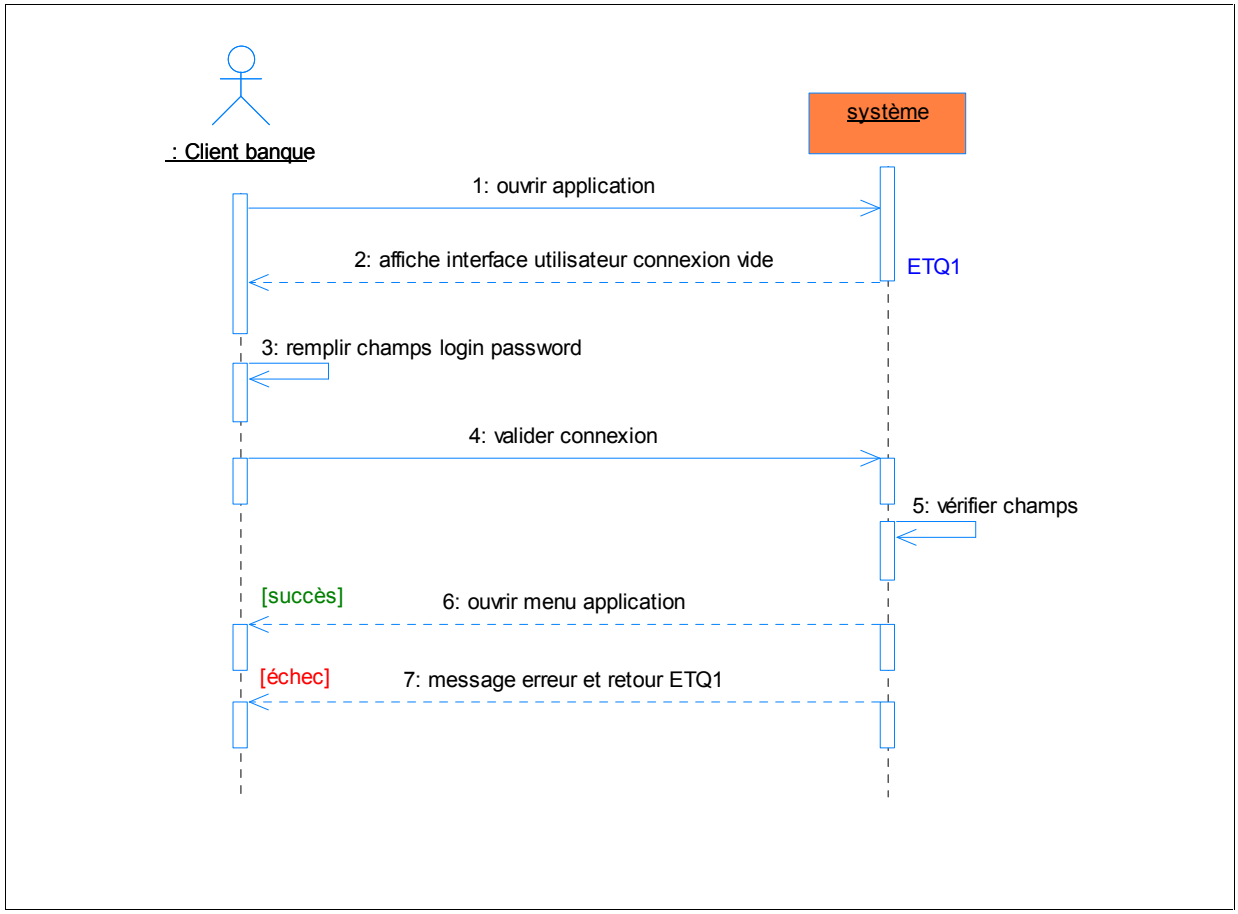

Figure 10 Diagramme de séquence « s'authentifier »

III.2.2.2 le diagramme de séquence « Consulter cours devise »

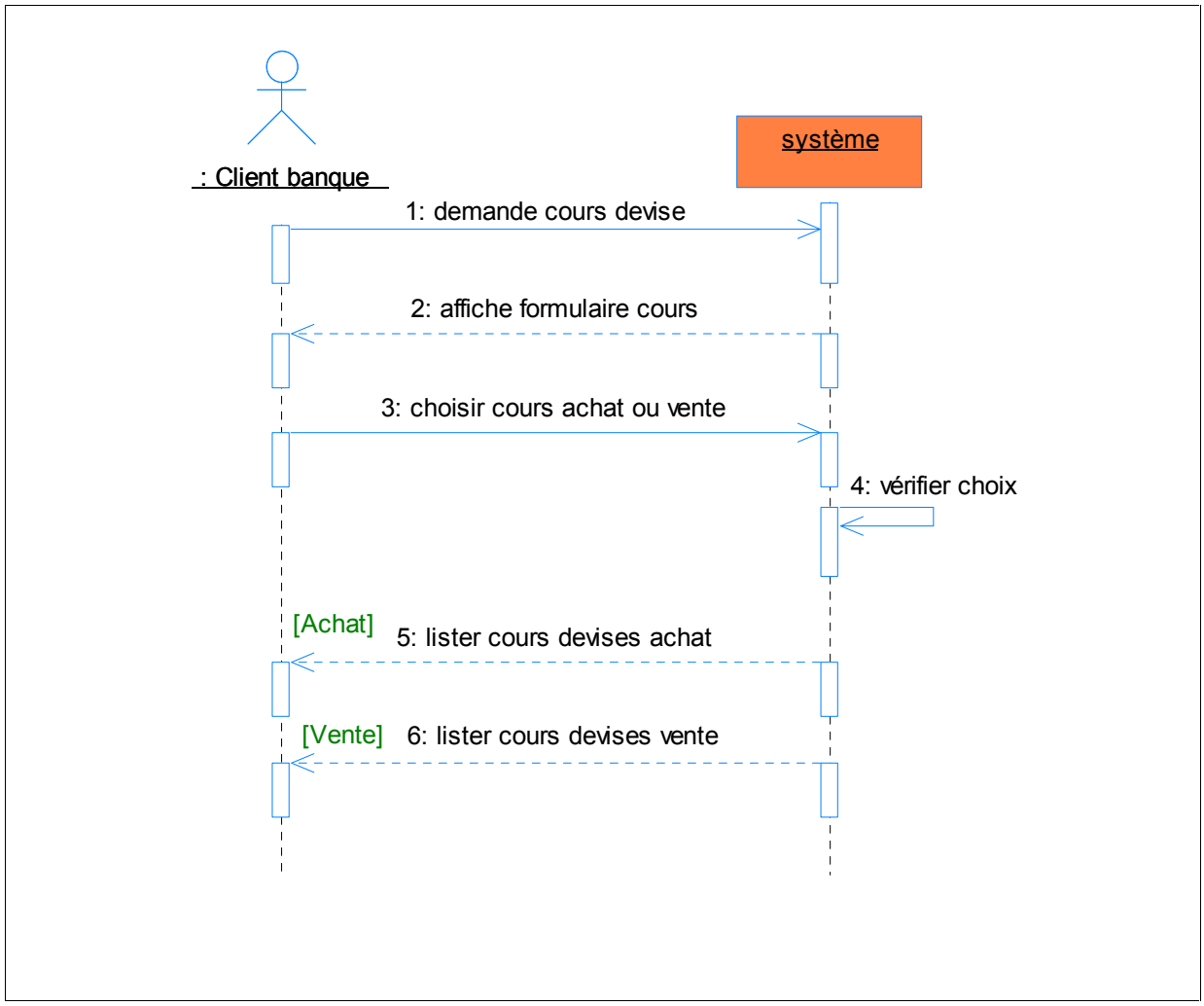

Figure 11 Diagramme de séquence « consulter cours devise »

III.2.2.3 le diagramme de séquence « Convertir billet de banque »

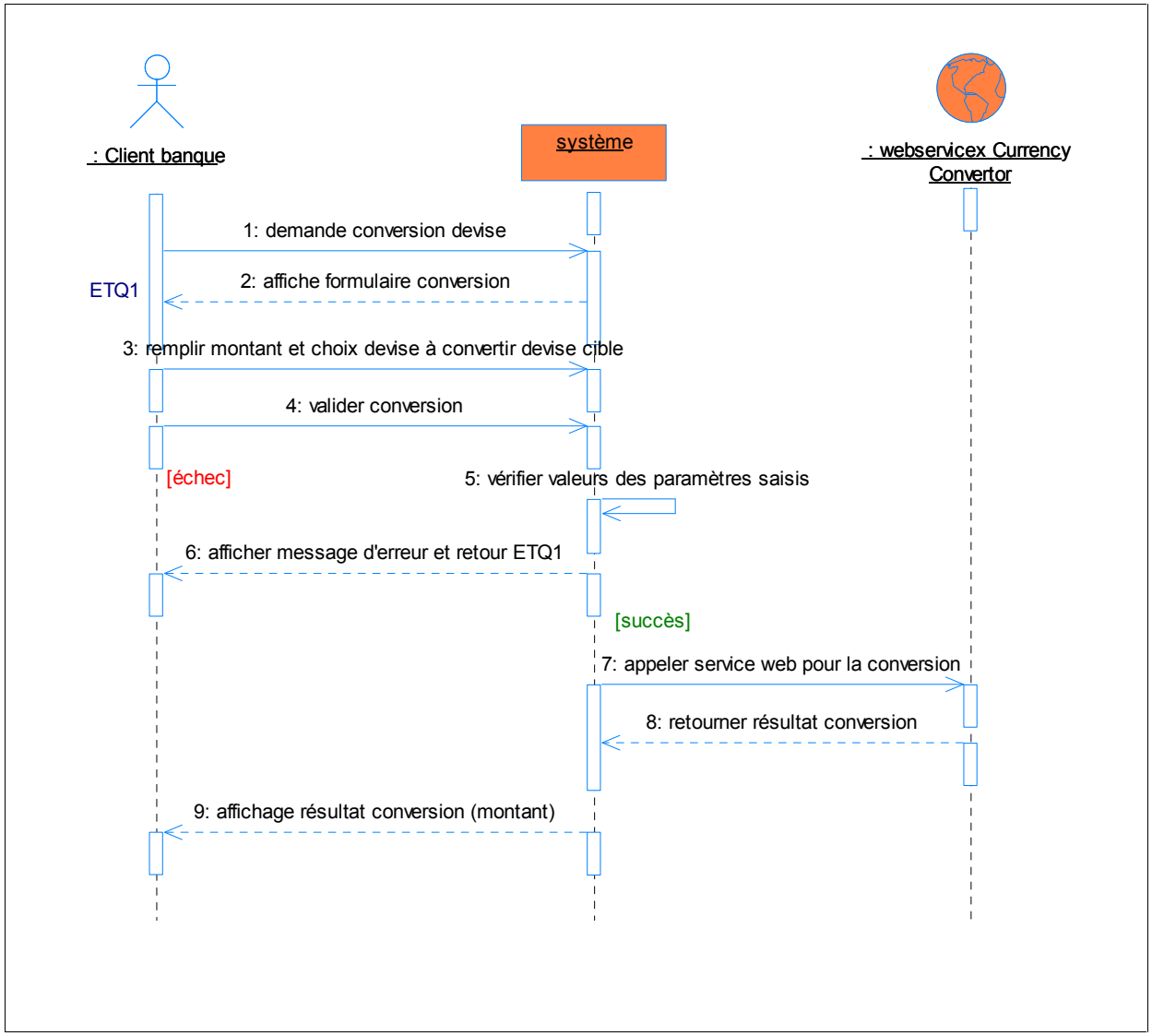

Figure 12 Diagramme de séquence « Convertir billet de banque »

III.2.2.4 le diagramme de séquence « Commander chéquiers »

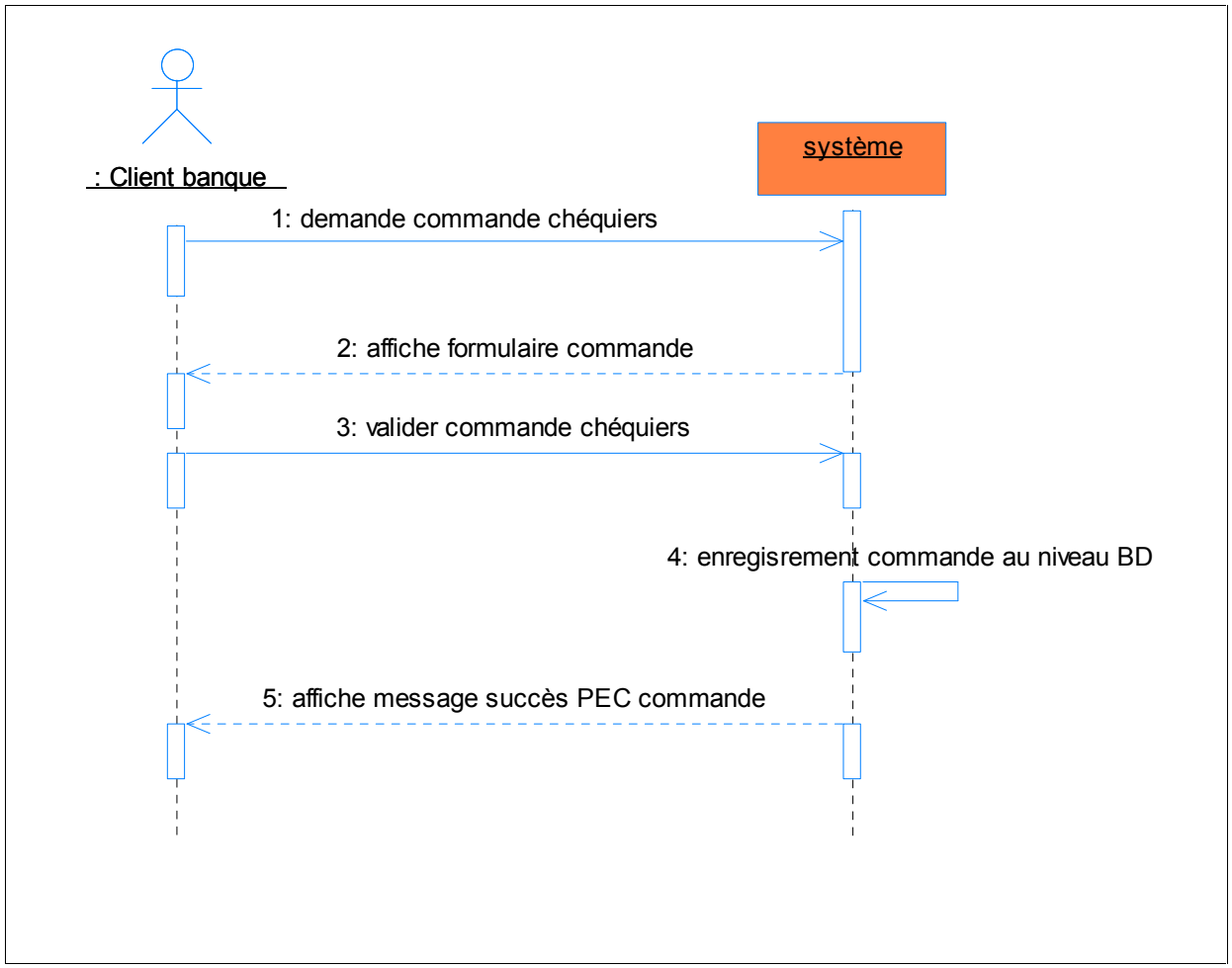

Figure 13 Diagramme de séquence «Commander chéquiers»

III.2.2.5 le diagramme de séquence « Consulter annuaire réseau agences »

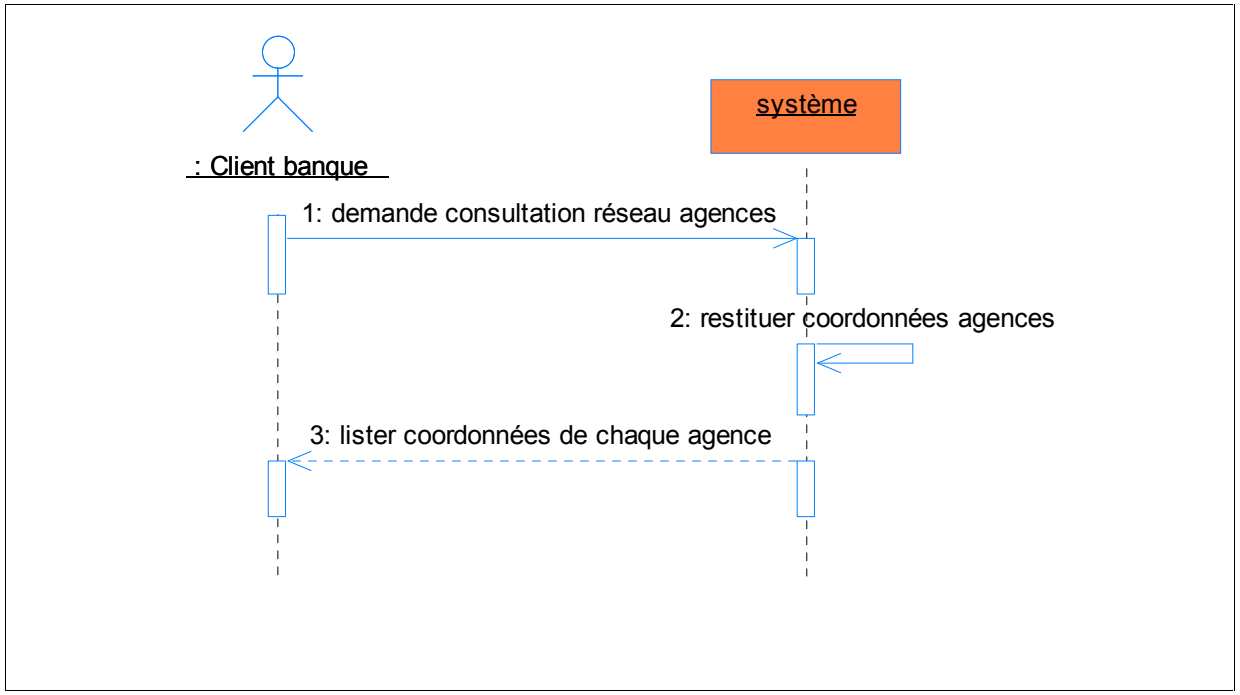

Figure 14 Diagramme de séquence «Consulter annuaire réseau agences »

III.2.4.6 le diagramme de séquence « Contacter banque »

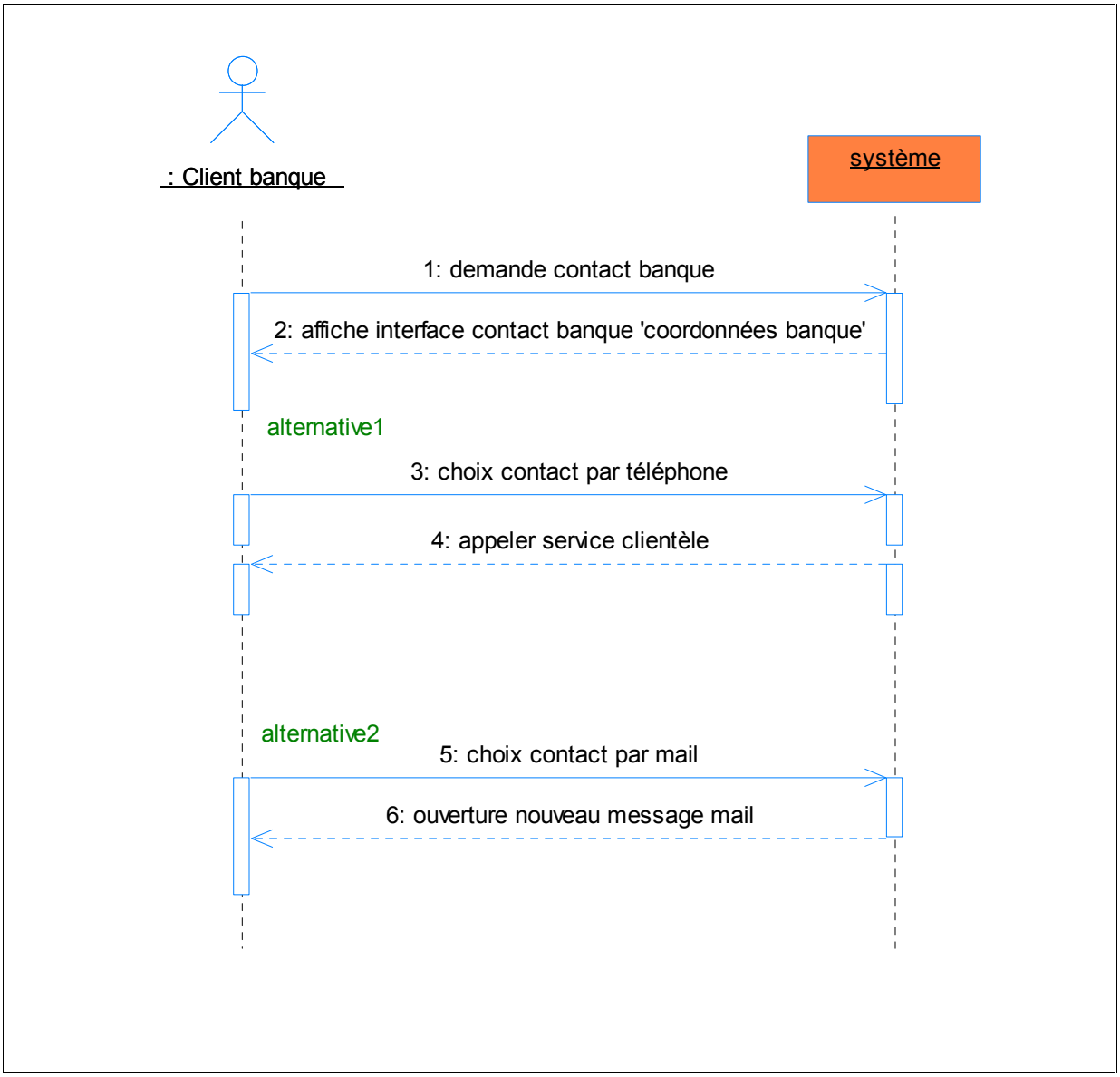

Figure 15 Diagramme de séquence «Contacter banque »

III.2.3 Les diagrammes d'activité

Le diagramme d'activité permet de représenter le déclenchement d'évènements en fonction des états du système et de modéliser des comportements parallélisables. Il donne une vision des activités propres à une opération ou à un cas d'utilisation.

Une activité est une opération d'une certaine durée qui peut être interrompue.

Dans ce cas, on va représenter ci-après des diagrammes d'activités qui décrivent :

L'authentification d'un utilisateur :

III.2.3.1 diagrammes d'activité « s'authentifier »

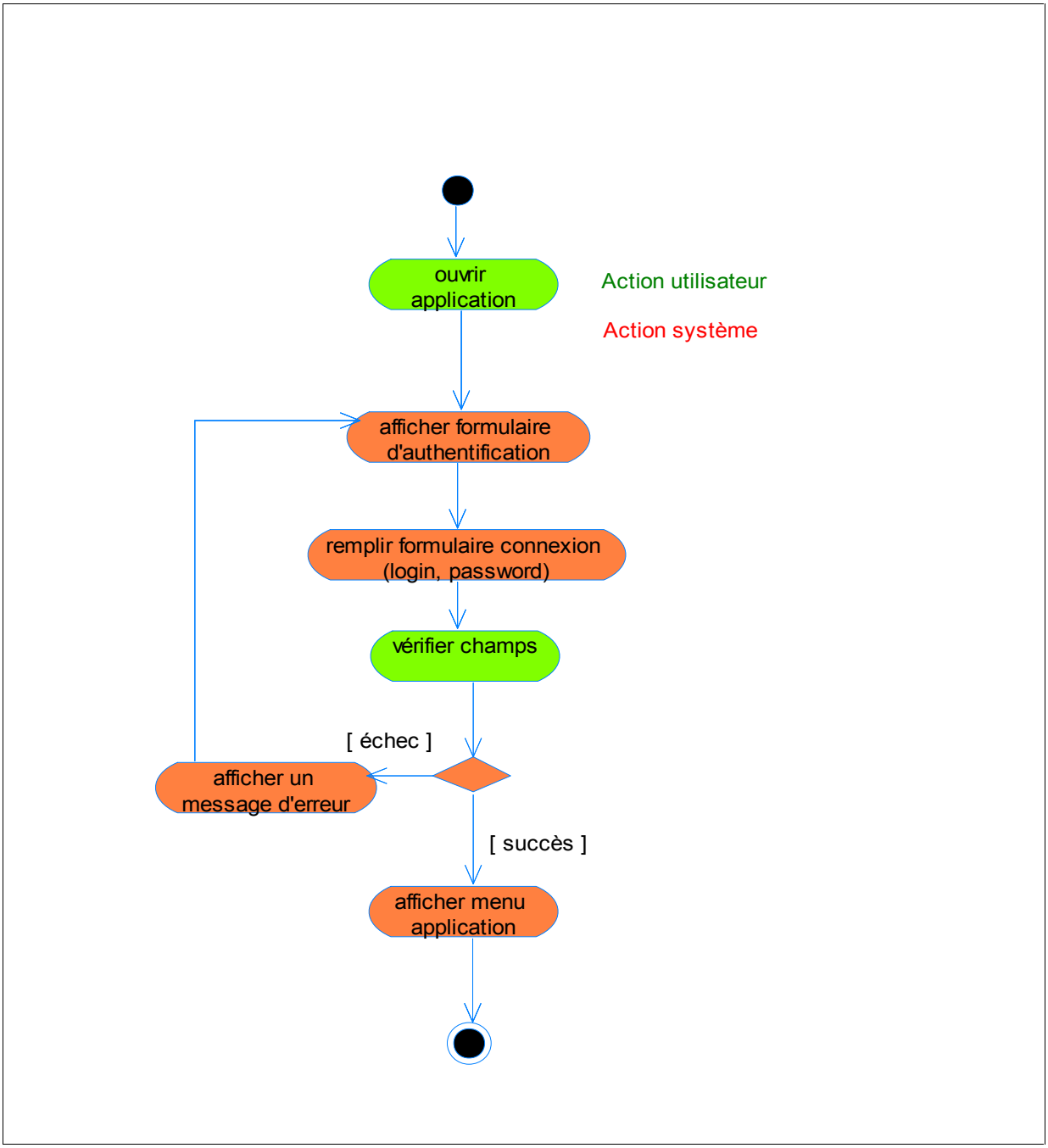

Figure 16 Diagramme d'activité « s'authentifier »

III.2.3.2 diagrammes d'activité «Consulter cours devise»

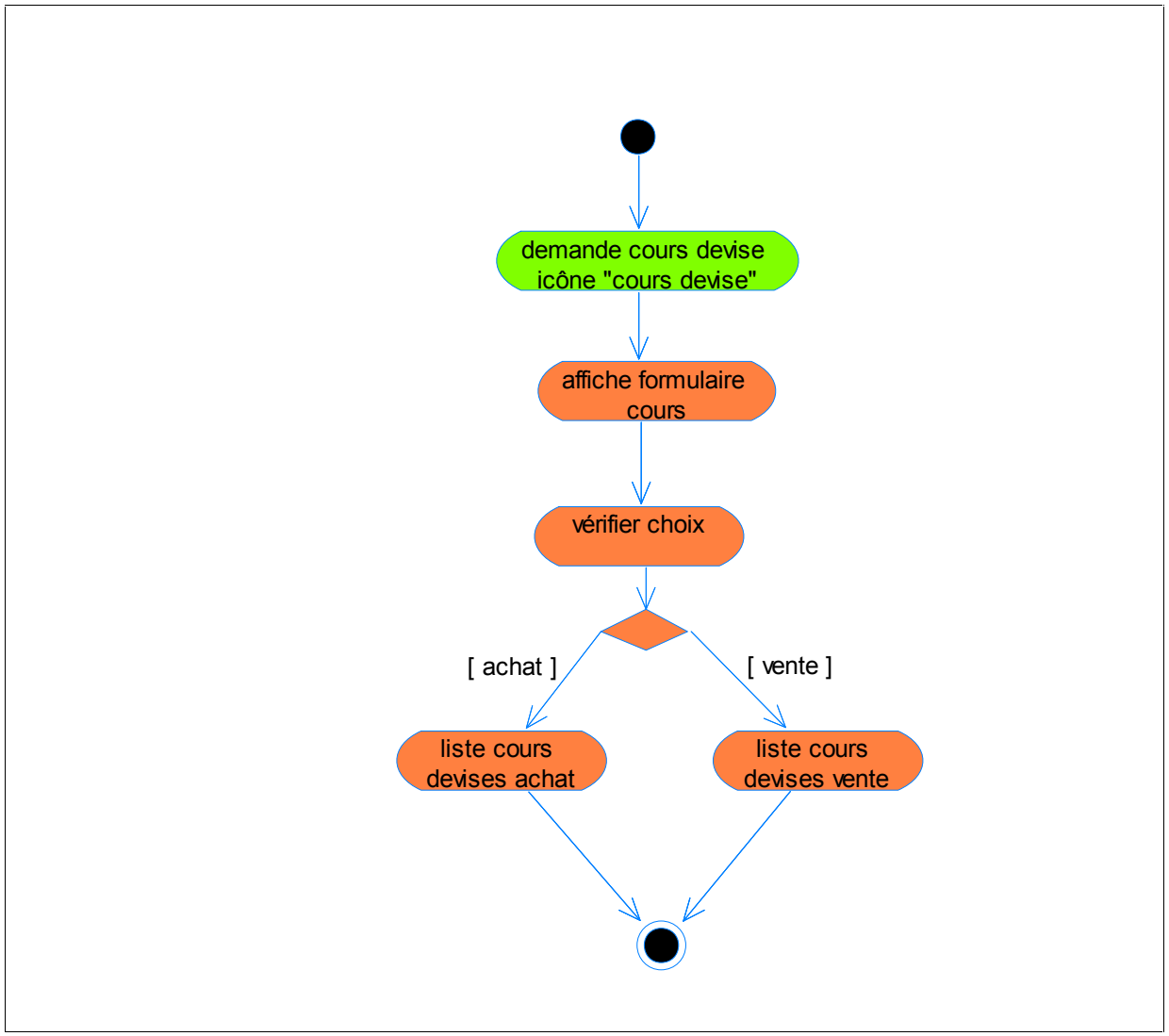

Figure 17 Diagramme d'activité «Consulter cours devise»

III.2.3.3 diagrammes d'activité «Convertir billet de banque»

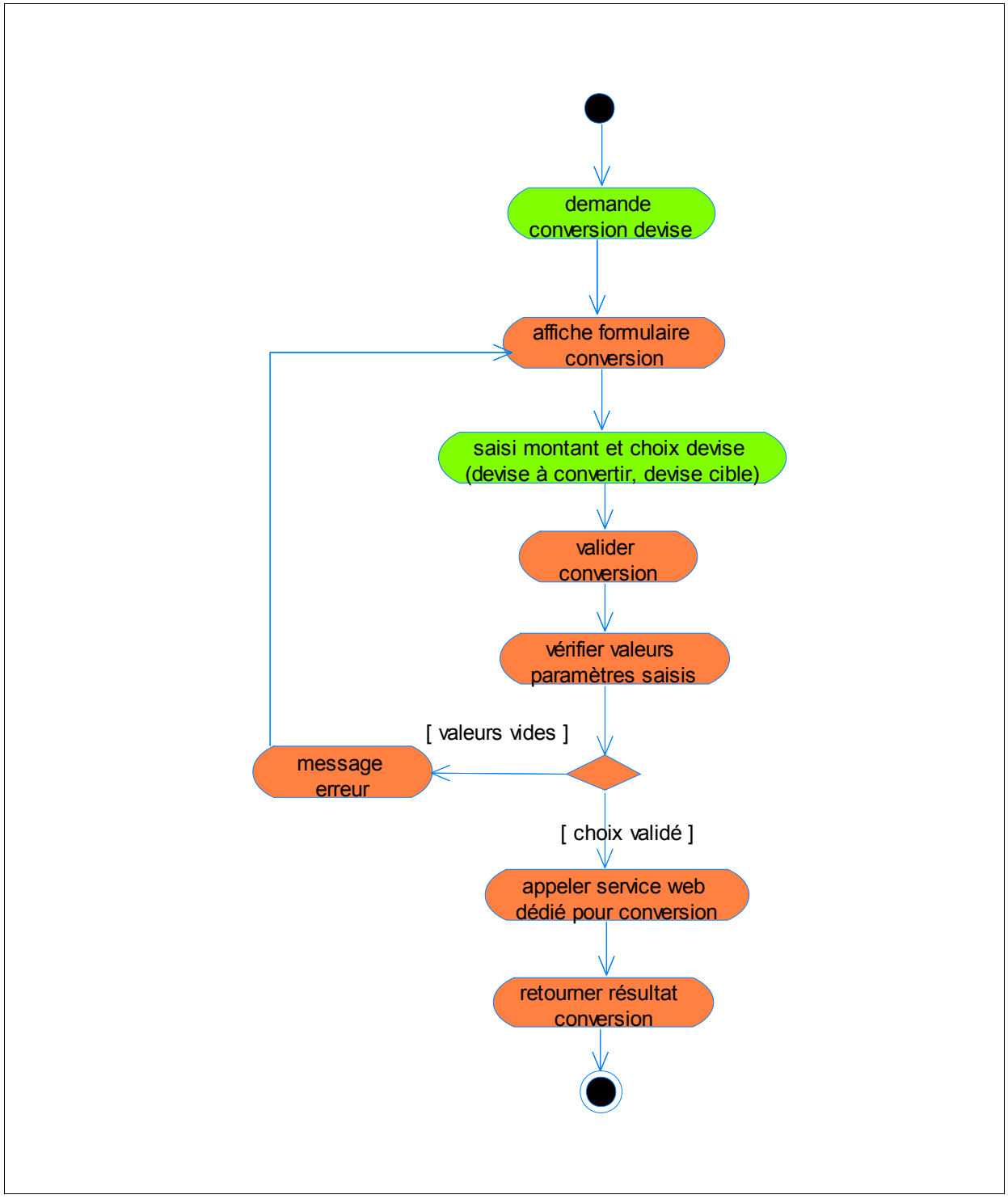

Figure 18 Diagramme d'activité «Convertir billet de banque »

III.2.3 4 diagrammes d'activité «Commander chéquiers»

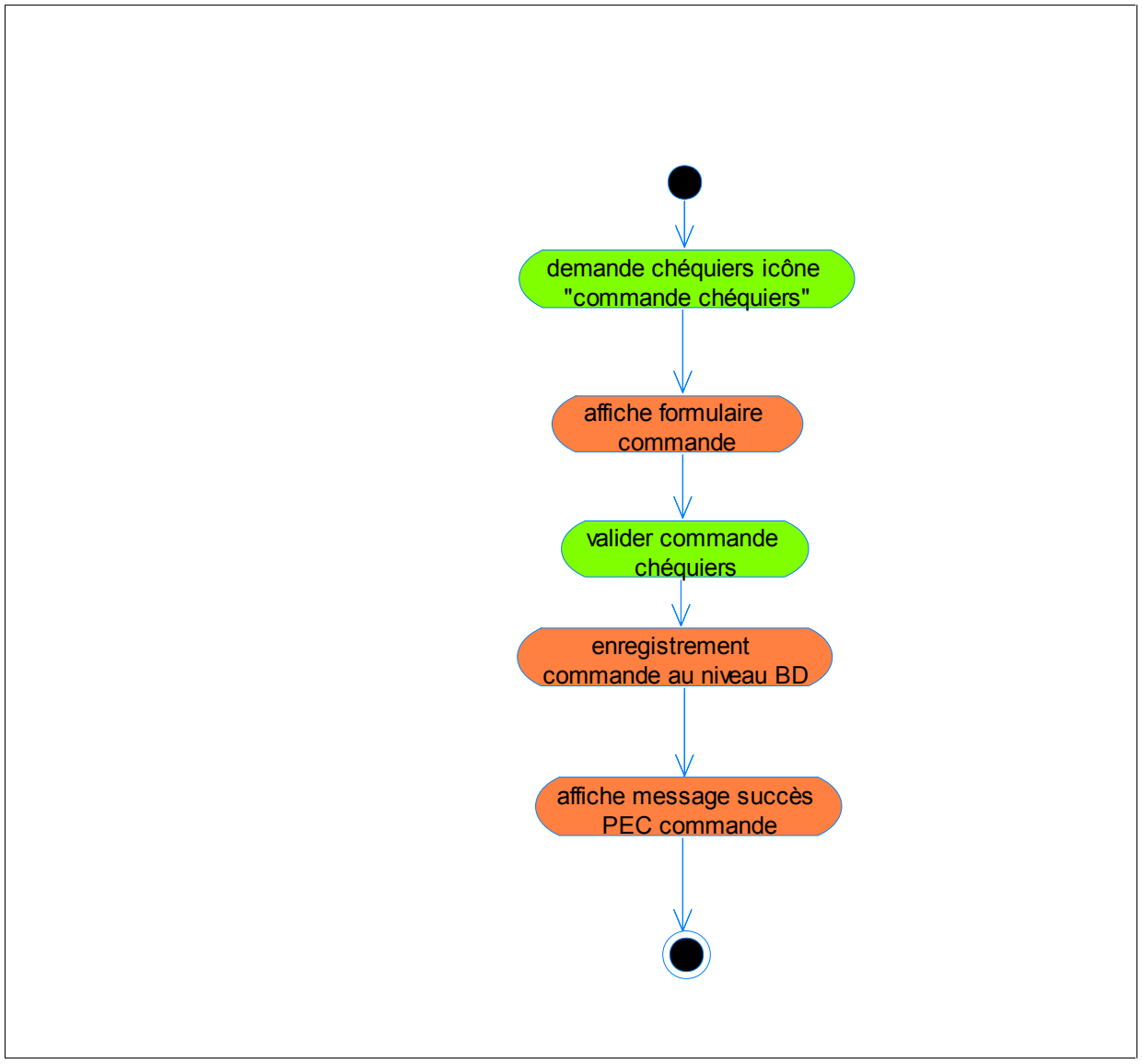

Figure 19 Diagramme d'activité «Commander chéquiers»

III.2.3 5 diagrammes d'activité «Consulter annuaire réseau agences»

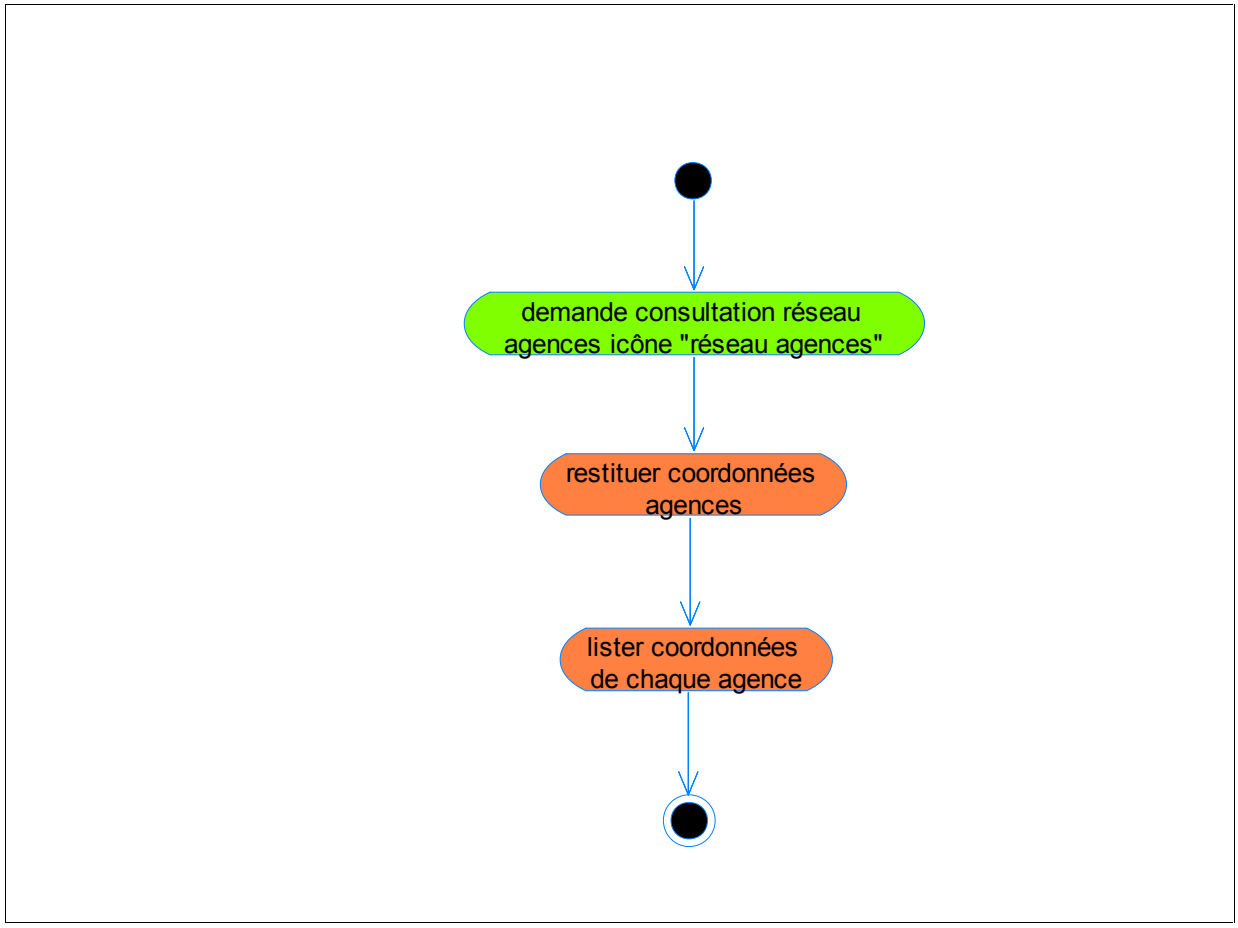

Figure 20 Diagramme d'activité «Consulter annuaire réseau agences»

III.2.3.6 diagrammes d'activité «Contacter banque »

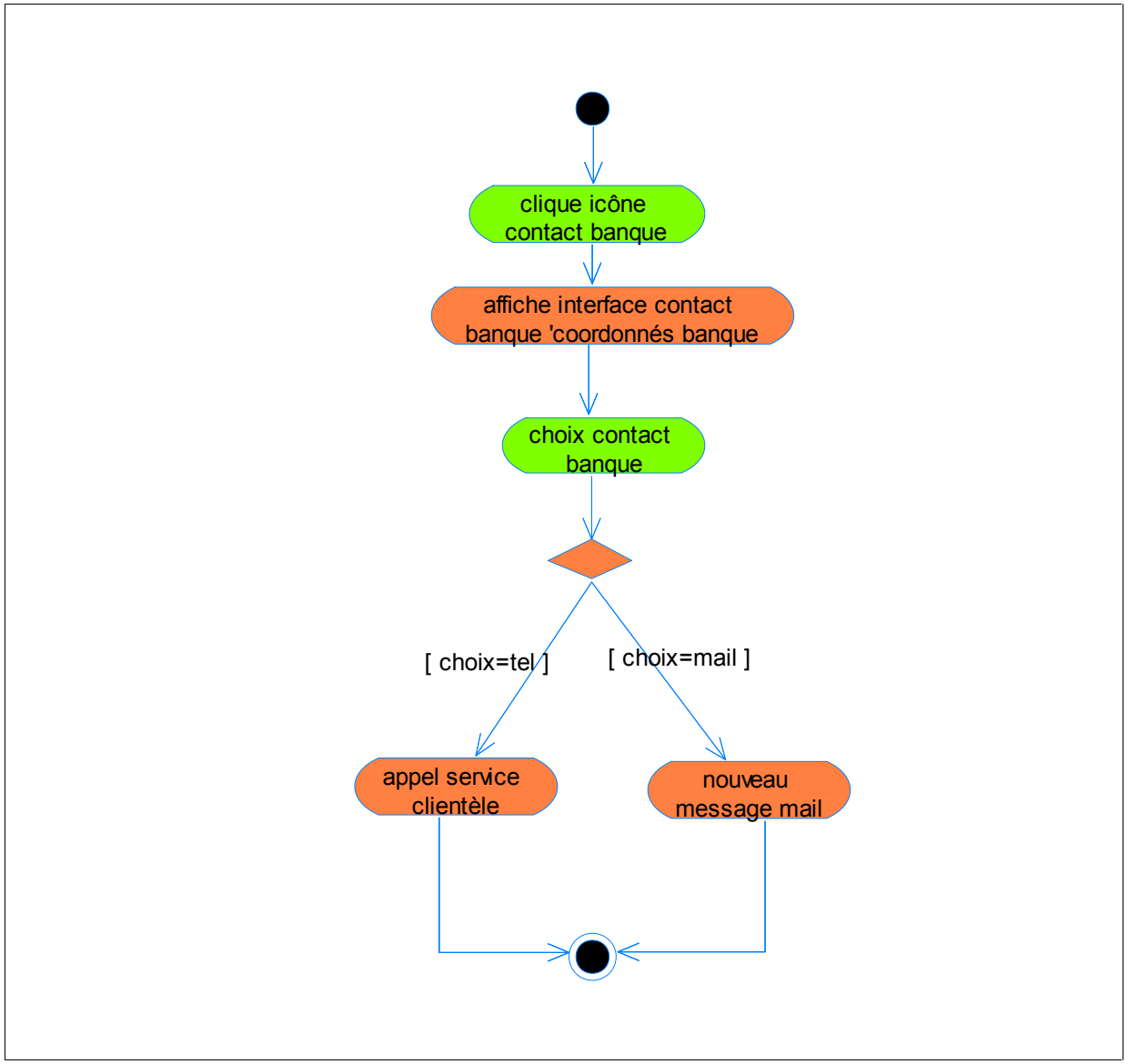

Figure 21 Diagramme d'activité «Contacter banque»

III.2.4 Les diagramme de collaboration

Un diagramme de collaboration montre les interactions entre les objets et leurs liens.

Contrairement aux diagrammes de séquence, un diagramme de collaboration montre les relations entre les objets et l'ordonnancement est explicité par des numéros de séquencement.

On présente ci-dessous les différents diagrammes de collaboration de notre application.

III.2.4.1 Le diagramme de collaboration « s'authentifier »

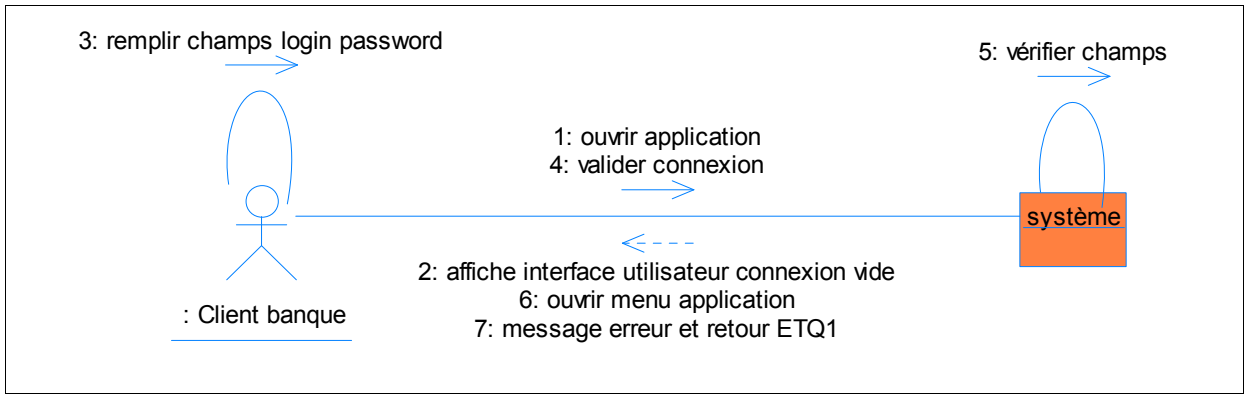

Figure 22Diagramme de collaboration « s'authentifier »

III.2.4.2 Le diagramme de collaboration «Consulter cours devise »

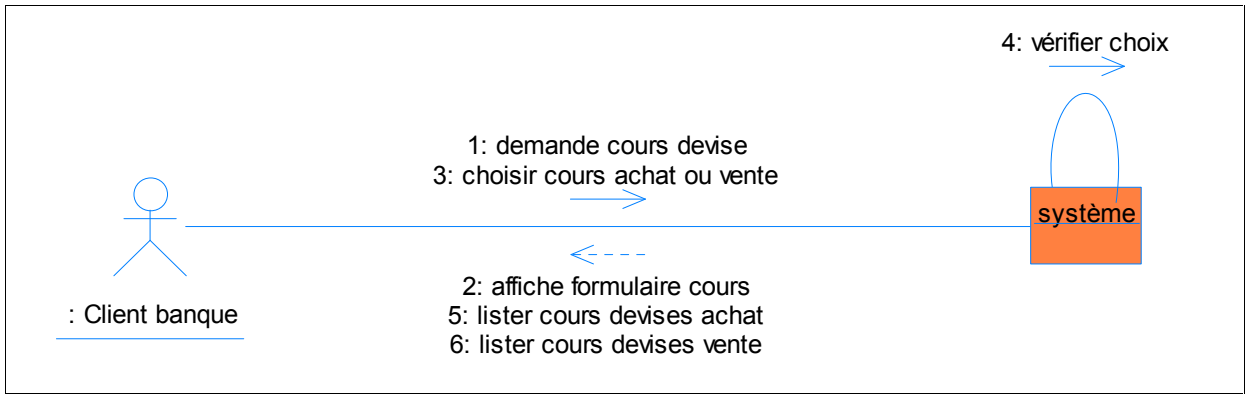

Figure 23 Diagramme de collaboration «Consulter cours devise »

III.2.4.3 Le diagramme de collaboration «Convertir billet de banque »

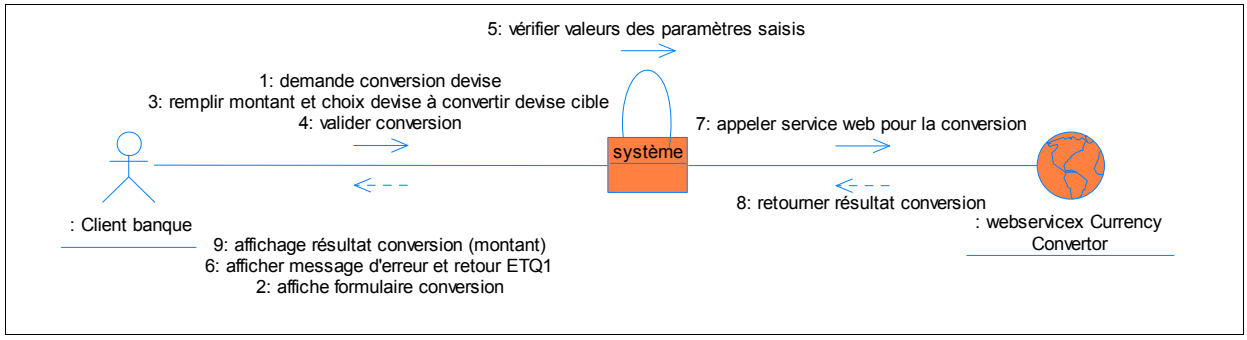

Figure 24 Diagramme de collaboration «Convertir billet de banque »

III.2.4.4 Le diagramme de collaboration «Commander chéquiers »

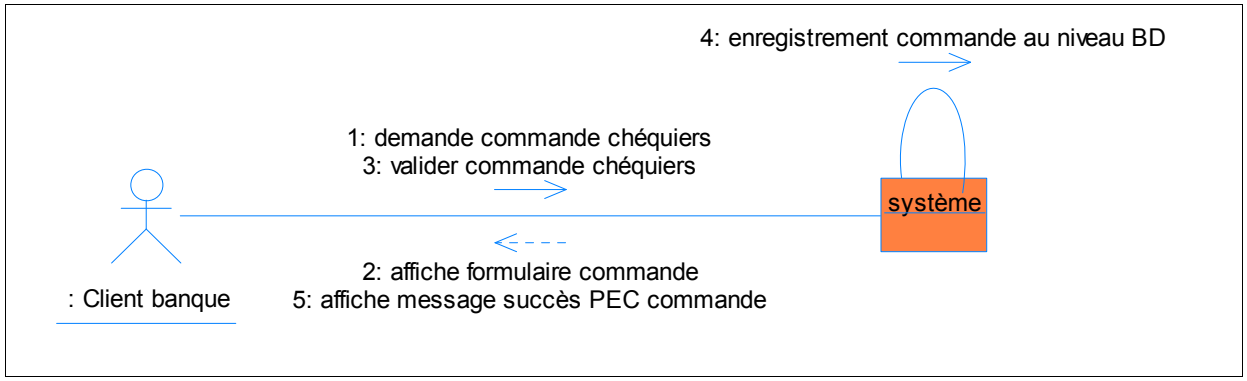

Figure 25 Diagramme de collaboration «Commander chéquiers »

III.2.4.5 Le diagramme de collaboration «Consulter annuaire réseau agences »

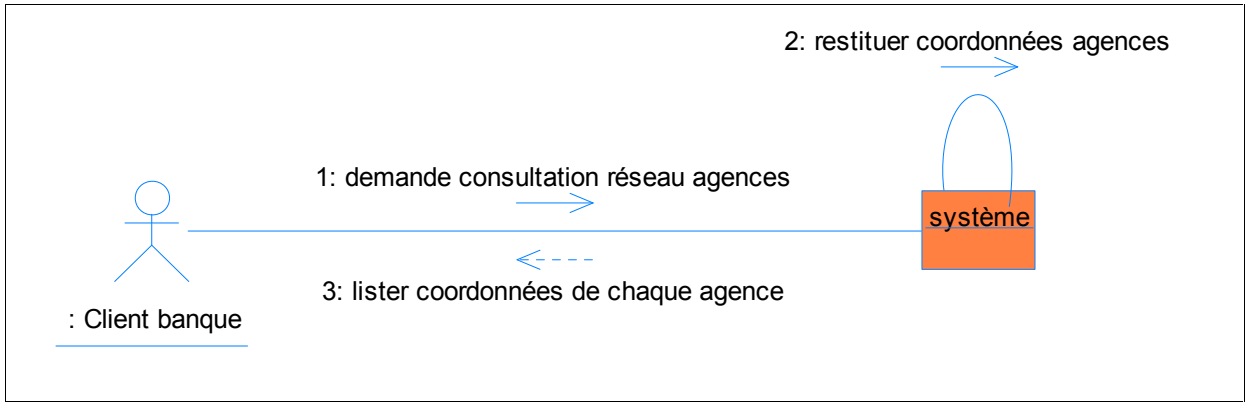

Figure 26 Diagramme de collaboration «Consulter annuaire réseau agences »

III.2.4.6 Le diagramme de collaboration «Contacter banque »

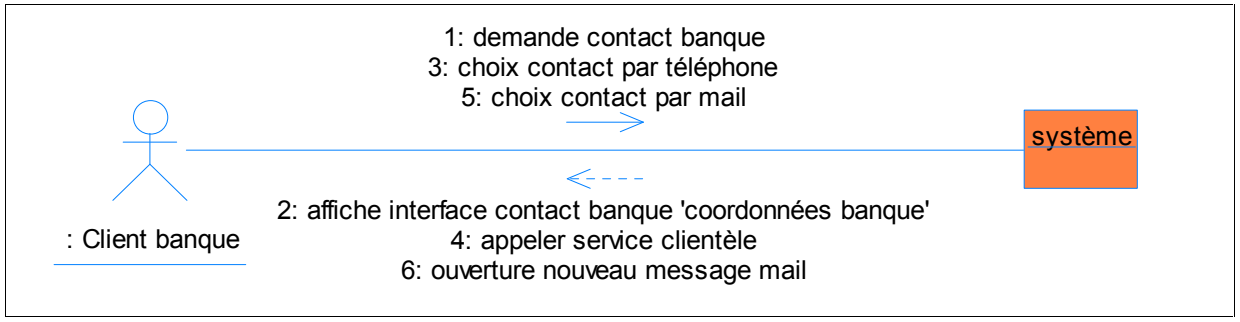

Figure 27 Diagramme de collaboration «Contacter banque »

CONCLUSION

Cette phase de conception avait pour intérêt de présenter les différentes étapes de conception de l'application tout en évoluant dans le niveau de détail, et doit par conséquent aboutir immédiatement à l'implémentation avec une vision claire des aspects fonctionnels ainsi que organisationnels de l'application.

CHAPITRE IV

Réalisation

INTRODUCTION

Cette partie constitue le dernier volet de ce rapport. Après avoir terminé la phase de spécification et conception, la solution étant déjà choisie et étudiée, il nous reste que de se décider dans quel environnement nous allons travailler, exposer les choix techniques utilisés et le langage adopté, et présenter l'implémentation et les tests réalisés.

IV.1 CHOIX TECHNIQUE

IV.1.1 Choix du langage de programmation

Java

Java est un langage de programmation orienté objet, développé par Sun Microsystems et destiné à fonctionner dans une machine virtuelle, il permet de créer des logiciels compatibles avec des nombreux systèmes d'exploitation.

Java et non seulement un langage de programmation puissant conçu pour être sûr, inter plateformes et international, mais aussi un environnement de développement qui est continuellement étendu pour fournir des nouvelles caractéristiques et des bibliothèques permettant de gérer de manière élégante des problèmes traditionnellement complexes dans les langages de programmation classiques, tels que le multithreading, les accès aux bases des données, la programmation réseau, l'informatique répartie.

En plus java est considéré comme un langage adaptable aux plusieurs domaines puisque une application web implémentée par celle-ci peut avoir des extensions ou des modifications dans le futur. De plus, java permet de réduire le temps de développement d'une application grâce à la réutilisation du code développé.

Json

JSON (JavaScript Object Notation – Notation Objet issue de JavaScript) est un format léger d'échange de données. Il est facile à lire ou à écrire pour des humains. Il est aisément analysable ou générable par des machines. Il est basé sur un sous-ensemble du langage de programmation JavaScript (JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999).
JSON est un format texte complètement indépendant de tout langage, mais les conventions qu'il utilise seront familières à tout programmeur habitué aux langages descendant du C, comme par exemple : C lui-même, C++, C#, Java, JavaScript, Perl, Python et bien d'autres. Ces propriétés font de JSON un langage d'échange de données idéal.(cf.annexe1)

IV.1.2 Choix de l'architecture de l'application

Client/ Serveur

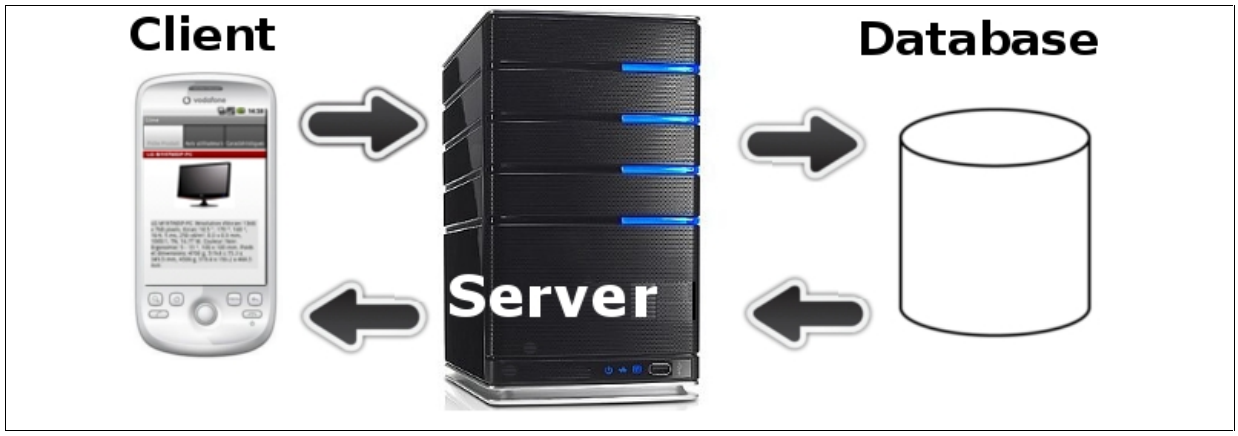

Figure 28 Architecture à 3 niveaux

Dans l'architecture à trois niveaux, les applications au niveau serveur sont délocalisées, c'est-à-dire que chaque serveur est spécialisé dans une tâche (serveur web/ serveur de base de données par exemple). Il permet :

- une plus grande flexibilité/souplesse ;
- une sécurité accrue car la sécurité peur être définie indépendamment pour chaque service, et à chaque niveau ;
- de meilleures performances, étant donné le partage des tâches entre les différents serveurs.

Cette architecture (appelée 3 tiers) fait intervenir trois parties indépendantes les unes des autres :

1. la couche de données liée au serveur de base de données (SGBD) : stockage et accès aux données. Le système de stockage des données a pour but de conserver une quantité plus ou moins importante de données de façon structurée. Nous pouvons utiliser pour cette partie des systèmes très variés qui peuvent être des systèmes de fichiers, des mainframes, des systèmes de bases de données relationnelles, etc.
2. la logique applicative : il se compose généralement d'un script ou d'un programme qui constitue les traitements métier nécessaires sur l'information afin de le rendre exploitable par chaque utilisateur.
3. La couche présentation (ou affichage) associé au client qui de fait est dit « léger » dans la mesure où il n'assume aucune fonction de traitement à la différence du modèle 2-tiers. C'est la partie la plus immédiatement visible pour l'utilisateur. Elle a donc une importance primordiale pour rendre l'information lisible, compréhensible et accessible.

Protocol et format de données

Protocole de communication

Dans notre projet, nous avons utilisé le protocole HTTP, afin de communiquer les données entre la partie cliente mobile et le serveur web. En effet, Le HTTP est un protocole qui définit la communication entre un serveur et un client (facilite le dispatch des fonctions). En général, nous utilisons la méthode Post pour envoyer des données au programme situé à une URL spécifiée. Dans notre cas la requête Post envoyée à partir de l'application client vers le serveur est de la forme suivante :

http://192.168.1.2:8080/nomapplication?parametre=valeur.

Format de données communiquées

JSON (*JavaScript Object Notation*) est un format de données textuel, générique, dérivé de la notation des objets du langage ECMAScript. Il permet de représenter de l'information structurée. Un document JSON ne comprend que deux éléments structurels : des ensembles de paires nom / valeur ; des listes ordonnées de valeurs.

Ces mêmes éléments représentent 3 types de données :

- des objets ;
- des tableaux ;

- des valeurs génériques de type tableau, objet, booléen, nombre, chaîne ou null.

Le principal avantage de l'utilisation de JSON, dans notre application, est qu'il est simple à mettre en œuvre. Au rang des avantages, nous pouvons également citer :

- Facile à apprendre, car sa syntaxe est réduite et non-extensible;
- Ses types de données sont connus et simples à décrire ;
- Peu verbeux et léger, ce qui le rend bien adapté aux terminaux mobiles au contraire au langage XML qui est très verbeux.

IV.2 ENVIRONNEMENT LOGISTIQUE

IV.2.1 Environnement de développement

J2EE : nous avons préparé une plateforme j2ee, qui est la proposition de SUN pour le développement et la mise en œuvre d'applications d'entreprise multi niveaux. Elle permet de simplifier le développement d'applications en environnement client léger et permet aussi au programmeur le développement normalisé de composants modulaires réutilisables.

Cette plate-forme J2EE est une suite robuste de services middleware qui simplifie la tâche des développeurs d'applications côté serveur. Elle repose sur les technologies existantes de la plate-forme J2SE afin de faciliter la création d'applications réparties.

Les principales caractéristiques de J2EE sont :

- C'est une spécification pour le langage de programmation Java plus particulièrement destinée aux applications d'entreprise.
- Une grande variété de plateformes et de systèmes d'exploitation sont capables de faire fonctionner une application J2EE.
- Il existe un grand nombre d'IDE différents pour le développement ;
- Toutes les spécifications sont publiques.

Outils de développement IDE NetBeans (coté serveur) Eclipse (coté client)

NetBeans est un environnement de développement intégré (EDI), placé en open source par Sun en juin 2000 sous licence CDDL et GPLv2 (Common Development and Distribution License).

En plus de Java, NetBeans permet également de supporter différents autres langages, comme Python, C, C++,JavaScript, XML, Ruby, PHP et HTML. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi langage, refactoring, éditeur graphique d'interfaces et de pages Web).

En plus de Java, NetBeans permet également de supporter différents autres langages, comme Python, C, C++,JavaScript, XML, Ruby, PHP et HTML. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refactoring, éditeur graphique d'interfaces et de pages Web).

Conçu en Java, NetBeans est disponible sous Windows, Linux, Solaris (sur x86 et SPARC), Mac OS X ou sous une version indépendante des systèmes d'exploitation (requérant une machine virtuelle Java). Un environnement Java Development Kit JDK est requis pour les développements en Java.

NetBeans constitue par ailleurs une plateforme qui permet le développement d'applications spécifiques (bibliothèque Swing (Java)). L'IDE NetBeans s'appuie sur cette plate-forme.

Eclipse est un environnement de développement intégré libre extensible, universel et polyvalent, permettant de créer des projets de développement mettant en oeuvre n'importe quel langage de programmation. Eclipse IDE est principalement écrit en Java (à l'aide de la bibliothèque graphique SWT, d'IBM), et ce langage, grâce à des bibliothèques spécifiques, est également utilisé pour écrire des extensions.

La spécificité d'Eclipse IDE vient du fait de son architecture totalement développée autour de la notion de plugin : toutes les fonctionnalités de cet atelier logiciel sont développées entant que plug-in.

Dans le cadre de notre projet, nous avons utilisé la version Eclipse Helios, avec le plugin

ADT de Google.

Le SDK Android

L'outil le plus important est le SDK Android. Facile à installer, il permet de télécharger tous les outils indispensables au développement d'applications. Un petit logiciel permet d'abord de télécharger les différentes versions du SDK (une version du SDK par version d'Android : 1.4, 1.5, 1.6, 2.0 etc.). Il permet également de télécharger les différentes versions des Google APIs

(APIs pour intégrer des fonctionnalités liées aux services Google tels que Maps etc.) ou de la documentation JavaDoc. Son fonctionnement est similaire aux gestionnaires de paquets de Linux.

ADT pour Eclipse

Eclipse est l'Environnement de Développement Intégré (ou IDE) le plus largement utilisé pour la programmation Java; très performant, il est de plus gratuit et open source.
Le langage privilégié pour le développement d'applications Android est justement Java.
Google a donc tout naturellement conçu un plugin pour Eclipse (un plugin est un module qui complète un logiciel hôte pour lui apporter de nouvelles fonctionnalités).

Android Development Tools, ou ADT, est très complet et surtout très pratique : conception graphique d'interfaces utilisateur, debug distant sur un téléphone, gestion de l'architecture de fichiers d'une application etc.

Emulateur

Nous l'avons évoqué plus haut, le SDK propose un émulateur Android. Il permet de lancer sur la machine du développeur un terminal virtuel représentant à l'écran un téléphone embarquant Android. C'est bien évidemment un outil indispensable pour le développement mobile. A chaque version d'Android est associée une version de l'émulateur, permettant au développeur de voir exactement à quoi ressemblera son application sur un matériel réel.
Rappelons cependant que l'émulateur ne propose pas toutes les fonctionnalités d'un vrai téléphone. Il ne permet par exemple pas d'émuler la gestion du Bluetooth.

IV.2.2 système de gestion de base de données

MySQL est un système de gestion de base de données (SGBD). Selon le type d'application, sa licence est libre ou propriétaire. Il fait partie des logiciels de gestion de base de données les plus utilisés au monde, autant par le grand public (applications web principalement) que par des professionnels, en concurrence avec Oracle et Microsoft SQL Server.
MySQL AB a été acheté le 16 janvier 2008 par Sun Microsystems pour un milliard de dollars américains. En 2009, Sun Microsystems a été acquis par Oracle Corporation, mettant entre les mains d'une même société les deux produits concurrents que sont Oracle Database et MySQL. Ce rachat a été autorisé par la Commission européenne le 21 janvier 2010.

MySQL est un serveur de bases de données relationnelles SQL développé dans un souci de performances élevées en lecture, ce qui signifie qu'il est davantage orienté vers le service de données déjà en place que vers celui de mises à jour fréquentes et fortement sécurisées. Il est multi-thread et multi-utilisateur.

IV.2.3 Serveur application (glassFish)

GlassFish est le nom du serveur d'applications Open Source Java EE 5 et désormais Java EE 6 avec la version 3 qui sert de socle au produit Oracle GlassFish Server (anciennement Sun Java System Application Server de Sun Microsystems). Sa partie Toplink persistence provient d'Oracle. C'est la réponse aux développeurs Java désireux d'accéder aux sources et de contribuer au développement des serveurs d'applications de nouvelle génération.
Vu que GlassFish inclut un serveur HTTP interne, il est aussi considéré comme un serveur http, donc il est capable de fonctionner en autonomie, pour traiter à la fois les requêtes HTTP simples et les applications web. Il représente les avantages suivants :

- Il est simple, beaucoup plus que les serveurs d'application Open Source « complets».
- Il est donc plus simple d'administrer une instance GlassFish qu'un serveur d'applications complet.
- Il n'occupe que 2 ports sur la machine (8080 et 8009), alors que les autres prennent une dizaine.

IV.3 TRAVAIL REALISE

La conception des interfaces de l'application m-banking est une étape très importante puisque toutes les interactions avec le couleur de l'application passent à travers ces interfaces, nous devons alors guider l'utilisateur avec les messages d'erreur et avec des notifications si besoin.

IV.3.1 Les jeux de test

Dans cette partie, nous allons présenter quelques cas d'utilisations, sous forme d'un guide utilisateur.

Pour accéder à notre application le client banque doit s'authentifier. Comme toute application, la sécurité d'accès est nécessaire. La figure ci-après donne l'interface à travers laquelle l'utilisateur s'identifie. Il saisit son login et son password puis le serveur vérifie ces informations.

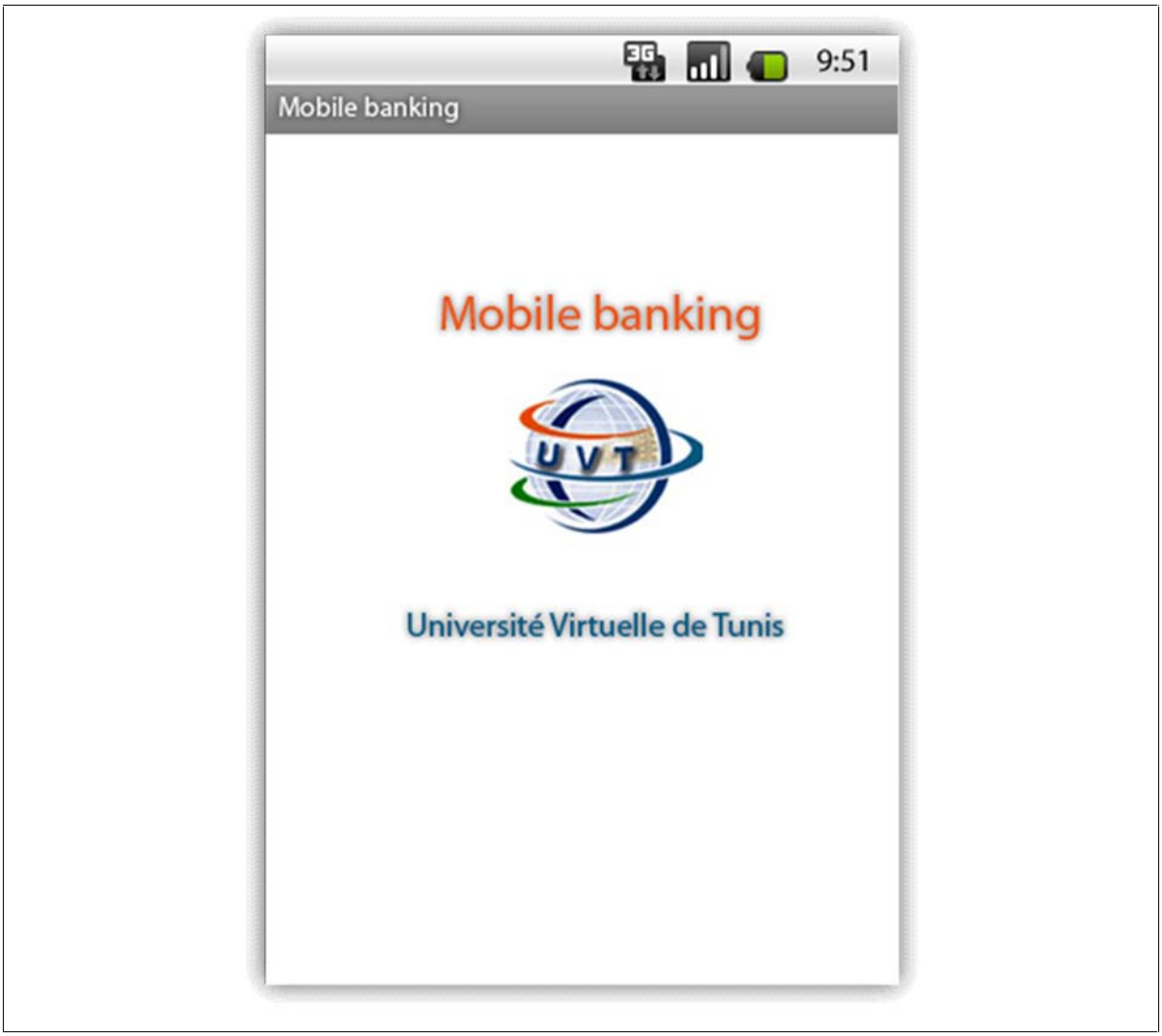

Figure 29 Interface chargement ouverture application

Figure 30 Interface d'authentification

En cas d'erreur, un message d'erreur sera affiché dans le cas où les informations écrites ne sont pas présentes dans notre base de données.

Figure 31 Interface d'authentification en cas d'erreur

Une fois les données sont valides, l'utilisateur accède au menu. Dans cette interface l'utilisateur peut gérer toutes les fonctionnalités de l'application.

Figure 32 Interface menu application

Si l'utilisateur sélectionne l'icône « cours devises » sur le menu principale. Le système affiche l'interface relative au cours devises, d'où le client banque peut choisir l'un des fonctionnalités ; cours achat ou cours vente via les onglets correspondants. Lorsque l'utilisateur choisie onglet achat le système affiche liste cours achat et lorsque il choisit cours vente le système affiche cours vente.

Figure 33 Interface cours devises

Dans l'écran « menu » si l'utilisateur choisit « change devises », le système enchaine à l'interface utilisateur relative au « change devises ».

Figure 34 Menu application /change devises

L'utilisateur saisi le montant et choisie les devises source et cible du formulaire et valide la conversion de billet de banque.

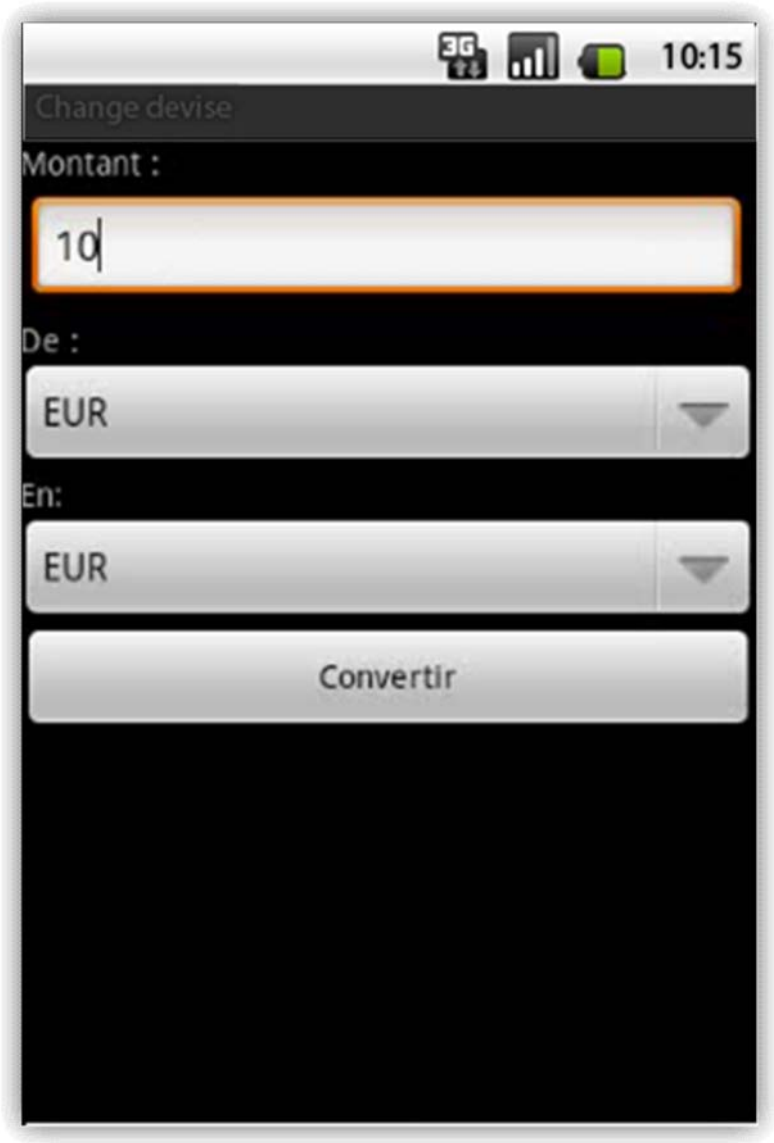

Figure 35 interface change devise

Figure 36 Interface change devise/choix devise

En effet, pour calculer le résultat Le système le service web distant afin de retourner le résultat de conversion. Par la suite le résultat sera affiché par la suite comme indique ci-dessous le masque de change.

Figure 37 résultat Interface change devises

Enfin, si l'utilisateur choisie de sélectionner la fonctionnalité « contact » l'interface suivante sera affiché afin de mettre à la disposition de client les coordonnées de la banque et le choix de contacter sa banque par mail ou par téléphone. Donc l'application prend en charge des actions susvisées.

Figure 38 interface contact

CONCLUSION GENERALE ET PERSPECTIVES

Au terme de ce rapport, nous pouvons conclure que ce stage de fin d'études nous a donné une occasion opportune nous permettant de confronter l'acquis théorique à l'environnement pratique.

En effet, le stage nous a permis de prendre certaines responsabilités, par la suite de consolider de plus en plus nos connaissances théoriques et pratiques. C'est là que réside la valeur d'un tel projet de fin d'études qui combine les exigences de la vie professionnelle aux côtés bénéfiques de l'enseignement pratique que nous avons eu à l'UVT.

Ce travail de conception et de développement d'une application M-BANKING au long du stage nous a été bénéfique sur plusieurs plans : il nous a permis de perfectionner nos connaissances acquises en programmation et en conception.

Du point de vue technique, ce projet nous a permis de nous adapter avec l'environnement du développement informatique, de même il nous a permis de maîtriser la méthode développement ; le Processus unifié et des nouvelles technologies de programmation.

Au début de notre stage, nous avons consacré du temps pour l'étude et recenser les fonctionnalités de notre application. L'étude analytique menée dans les détails nous a permis de prévoir puis contourner les problèmes rencontrés. Et tout au long du développement, nous avons concentré sur les nouvelles technologies utilisés et les techniques de programmation appliquées.

Tout au long de l'élaboration du projet, nous avons rencontré plusieurs difficultés tant au niveau conceptuel qu'au niveau de la réalisation. Tout de même, nous avons réussi à les surpasser pour présenter en fin de compte une application opérationnelle.

Comme perspective, nous espérons voir notre application évoluer par une étape d'approfondir le volet transactionnel avec des services distribués un serveur web et bases données et des services web afin de rendre notre application plus intéressante. Qui permet L'échange, des diverses données en utilisant des réseaux et des formats normalisés.

Nous espérons enfin que le travail que nous avons effectué a été à la hauteur de la confiance qui nous a été donnée.

REFERENCES

[1]http://www.jmdoudoux.fr/accueil_java.htm.

[2]www.bd.enst.fr/dombd/Cours/Applications/3tiers/index.html.

[3] http://developer.android.com/index.html

[4]http://android-developers.blogspot.com/

[5]http://www.json.org/

[6]http://www.w3schools.com/json/default.asp

[7] Programmation Android De la conception au déploiement avec le SDK Google Android

“Damien Guignard;Julien Chable; Emmanuel Robles “

[8] L’art du développement android « Mark Murphy ; PEARSON »

ANNEXES

Annexe 1 JSON ou JavaScript Object Notation

INTRODUCTION

La notation JSON n'est pas une représentation propre au langage Javascript mais elle y est couramment associé ne serait-ce que de part son nom (JavaScript Object Notation soit Notation objet pour Javascript).

Cette notation permet de sérialiser (partiellement) des objets et tableaux. C'est à dire les "transformer" en chaînes de caractères qui pourront être transférées facilement d'un langage à l'autre, sur le reseau, etc. pour être traitées dans un autre environnement que celui d'origine (après désérialisation).

LA NOTATION EN 2 MOTS

Pour résumer, les tableaux sont transcrits sous forme d'une chaîne de caractères délimitée par des crochets. Cette chaîne contient l'ensemble des valeur (entre guillemets pour les chaînes de caractères) séparées par des virgules.sera représenté par la chaîne de caractères **["JSON", 2]**

Et les objets (ou les tableaux associatifs) sont transcrits sous forme d'une chaîne de caractères délimitée par des accolades. Cette chaîne contient l'ensemble des variables membres (nom/valeur) séparés par des virgules. Les variables membres sont représentés par la concaténation du nom de la variable (entre guillemets), le caractère deux-points suivi de la valeur (entre guillemets pour les chaînes de caractères).

Ainsi l'objet schématisé ci-dessous

```
MonObjet {
    nom = "JSON";
    langage = "Javascript";
    version = 2;
}
```

Seront représentés par la chaîne de
caractères {"nom":"JSON","langage":"Javascript","version":2}

POURQUOI FAIRE?

Il est certainement possible de trouver diverses utilisations possibles de la notation JSON mais en pratique elle est utilisée pour passer aisement une multitude de paramètres à une fonction mais avec un seul argument. Comme dans le schéma suivant:

```
afficheMsg({"msg":"Erreur ligne 2", "couleur":"rouge", "police":"12pt"})
```

Il est alors possible d'avoir un grand nombre de paramètres optionnels. Il est possible de les passer dans un ordre quelconque, etc.

Pour être plus précis encore, cette notation est particulièrement pratique lors de l'utilisation de callbacks (fonctions dont le nom est passé en argument d'un traitement pour être appelée au cours ou à la fin du traitement). Avec la notation JSON, il est possible de ne pas avoir à ce soucier du nombre d'arguments de la callback en regroupant tous les paramètres "optionnels" dans un seul argument au format JSON.

Sachant que les callbacks jouent généralement un rôle important dans la programmation AJAX, on comprend aisément toute l'attention portée à la notation JSON.

COMPRENDRE JSON : L'ECHANGE DE DONNEES SIMPLIFIE

"Echange de données" est aujourd'hui, dans de nombreux esprits, synonyme de XML, le format d'interopérabilité par excellence. Des projets faisant appel à des méthodes JavaScript ont pourtant vu le jour, où les données sont transportées non par flux XML, mais directement via des méthodes du navigateur, notamment `HttpRequest` - l'exemple le plus connu étant GMail.

Basé sur ces méthodes, un format émerge qui tente de prendre en compte les besoins, en méthodes simples et légères pour l'échange de données, des applications basées sur le navigateur : JavaScript Object Notation (JSON), basé sur ECMAScript. Une extension est également en cours de spécification : JSON-RPC, qui prend les avantages de JSON face à XML (plus léger), et les englobe dans un protocole similaire à XML-RPC (mais, donc, sans le XML…). JSON-RPC-Java est une implémentation fonctionnelle du protocole en Java, et JSOLait une implémentation JavaScript (les deux sont disponibles sous licence LGPL).

L'objectif de ce projet n'est pas de remplacer XML comme format d'échange, car celui ci

conserve des avantages propres, notamment l'extensibilité. JSON offre simplement aux projets Web légers une méthode d'échange digeste.

En JavaScript, un échange prendrait cette forme :

```
var myJSONObject = {
  "bindings": [
    {"ircEvent": "PRIVMSG", "method": "newURI", "regex": "^http://.*"},
    {"ircEvent": "PRIVMSG", "method": "deleteURI", "regex": "^delete.*"},
    {"ircEvent": "PRIVMSG", "method": "randomURI", "regex": "^random.*"}
  ]
};
```

...où l'on créé un objet binding contenant lui-même trois objets, chacun lançant un message au moyen de ses membres.

L'intérêt consiste principalement à afficher de nouvelles données sur une page Web sans devoir la recharger intégralement : seules les données nécessaires sont chargées. En clair, il s'agit d'une méthode permettant de se passer de Flash, des frames ou iframes, ou d'applets Java, et utiliser JavaScript en visant une balise DIV. Par ailleurs, il n'est plus nécessaire de faire appel à un parseur XML pour gérer les données, étant donné qu'elles sont directement prises en compte par JavaScript.