Projet : Application mobile de capture des besoins informatiques d'utilisateurs non-informaticiens

Rapport Final

Date de rédaction : 22 mai 2016

Sommaire

I) Présentation du sujet

A) Contexte de réalisation du projet

Afin de structurer nos idées, nous commencerons par vous décrire l'intérêt et les fonctionnalités principales, très brièvement, de l'application.

Le projet consiste à concevoir et développer une application mobile qui permettra à un utilisateur, lors d'un rendez-vous avec un client, de recueillir les besoins de cette personne et de les centraliser au même endroit. Il pourra à la fois enregistrer du texte, du son et de la photo dans un premier temps. L'application doit permettre l'ajout de nouvelles fonctionnalités.

Cette idée d'application nous a été présentée dans le cadre de notre Master 1 MIAGE. En effet, plus qu'un projet de groupe lambda, ce projet a pour vocation d'être utilisé à la fois par des professionnels mais aussi des particuliers. Nous devrons apporter une attention particulière à l'élaboration de la documentation de notre application, pour qu'elle soit à la fois fonctionnelle et technique.

Afin de décrire l'intérêt de notre application, il est judicieux d'utiliser un exemple : Prenons le cas d'un commercial d'une entreprise informatique. Sa mission consiste à réaliser des rendez-vous avec des clients afin de recueillir des informations sur leur(s) projet(s), pour ensuite réaliser un ou plusieurs devis, reprenant les données recueillies et apportant des solutions aux besoins de ces personnes. Il arrive qu'il y ait des pertes d'informations lors de ces rendez-vous. Ces problèmes résultent des maigres outils de récolte d'informations, utilisés lors de ces entretiens. La plupart du temps la simple prise de note via un ordinateur ou une feuille de papier semble suffire. Malheureusement, dans la plupart des cas, ces solutions ne sont pas suffisantes, elles ne sont pas optimales.

C'est ici que nous entrons en jeu. Muni de son téléphone, à l'aide de notre application, le commercial pourra recueillir les informations comme il le souhaite. Que ce soit en écrivant du texte, en enregistrant du son ou bien en prenant des photos, l'utilisateur aura à sa disposition tous les outils nécessaires à un bon recueil des besoins.

Vous êtes face à un client qui parle extrêmement, vous n'arrivez pas à noter tout ce qu'il dit et vous n'osez pas l'interrompre ? Utiliser donc notre outil d'enregistrement audio ! Vous pourrez conserver par la suite votre écoute sur votre téléphone, ou bien le transférer sur votre ordinateur, pour le réécouter plus tard.

Votre client souhaite vous montrer le schéma de son projet ? Vous n'avez qu'à prendre une photo, avec notre application, pour la conserver et la montrer, par la suite, à vos dirigeants.

Nous ne détaillerons pas plus chacune des fonctionnalités car cela sera fait par la suite dans le dossier. Néanmoins nous avons ici voulu vous faire comprendre l'intérêt de notre application. Nous sommes conscients que les fonctionnalités de notre application sont primaires et que vous pouvez, à l'aide d'un ordinateur, retrouver la plupart de ces dernières. Cependant, l'idée ici est de n'utiliser qu'un seul outil pour recueillir les besoins d'un client et de pouvoir centraliser toutes les ressources, issues du ou des rendez-vous, au même endroit à savoir : notre application.

B) Tâches planifiées

Monsieur CRESCENZO, notre client, souhaitait pouvoir, depuis l'application, créer un projet, y ajouter des documents (en passant par les plugins), pouvoir les tagger, stocker ces documents en local mais aussi sur un cloud.

Le client souhaitait qu'au terme du projet nous réalisions une version stable et fonctionnelle de l'application, compatible avec les téléphones Android. Autrement dit il souhaitait que l'application soit utilisable dans sa version 1.0, disposant de quelques plugins primaires d'utilisation. Le cœur de notre livrable ne fut pas de développer de gros modules complexes, mais plutôt de créer une structure saine et solide afin que, plus tard, d'autres utilisateurs puissent développer des modules à leur tour, compatible avec notre application, et les intégrer dans cette dernière. Nous devions donc réaliser la structure de manière à laisser possible la greffe de plugins par des utilisateurs extérieurs.

Il était logique que nous réalisions un document explicatif permettant de comprendre facilement l'application et expliquant comment il fallait s'y prendre pour créer un plugin. Autrement dit, notre projet ne fut pas qu'un « simple » développement d'application. Nous devions livrer également de nombreux documents afin de pouvoir faire évoluer notre solution par la suite, ou bien encore de comprendre son fonctionnement interne facilement, grâce à une conception détaillée et schématisée, par des diagrammes UML notamment, ou encore des tutoriels d'utilisation et de compréhension.

C) Solution préalablement existante

Notre tâche initiale consistait à rechercher si des solutions de notre projet existaient déjà. Nous nous sommes donc penchés sur les applications « couteaux suisses » disponibles. Nous avons été surpris par le fait qu'aucune application de ce type n'existait déjà. Cette information a permis à notre équipe de concevoir entièrement l'application de manière saine, sans être tenté de copier certains bouts d'une autre application potentiellement semblable à la nôtre.

L'absence d'exemples de notre application peut se traduire par plusieurs facteurs. Le premier c'est que les plugins, comme exprimé précédemment, ont des fonctionnalités très primaires et que chaque personne est capable, soit à l'aide de son smartphone, soit à l'aide de son ordinateur ou encore à l'aide d'une feuille de papier et d'un stylo, de capturer les besoins d'un client. Néanmoins la problématique qui fut la nôtre était de proposer un outil regroupant tous ces fonctionnalités. Pour conclure nous sommes fiers d'avoir pu réaliser une application stable et complète, accompagnée de sa riche documentation, nécessaire pour l'évolution de notre application vers une nouvelle version.

II) Travail accompli

A) Cahier des Charges

Détail du projet

Architecture globale

La charte graphique à concevoir et développer une application mobile (pour ordinateurs, tablettes et smartphones Android) qui permettra à un spécialiste de recueillir les besoins sans être forcément bloqué derrière un clavier d'ordinateur.

Avec sa tablette ou son smartphone, il pourra interroger les utilisateurs directement sur leur poste de travail : enregistrement audio ou vidéo, prise de notes libres, rédaction des scénarios, dessins de diagrammes, etc. (tout cela ne fera pas forcément partie du projet).

Charte graphique

La charte graphique ci-dessous est valable pour l'application mais n'est pas encore définitive

Couleur dominante : Rouge.

Logo : Les ondes présentes sur le logo sont les informations que nous « capturons » de nos utilisateurs. Le nom « capture » symbolise la capture des besoins.

Contenu de l'application (visuel) : L'utilisateur pourra jongler entre 3 fenêtres :

- Fenêtre d'accueil : cette fenêtre listera la liste des projets de l'utilisateur. Elle permettra également soit d'ouvrir un projet existant, soit d'en créer un nouveau ;
- Fenêtre d'un projet : cette fenêtre indiquera les informations relatives à ce projet (nom, description, fichiers) et permettra de créer de nouveaux documents en proposant à l'utilisateur de choisir avec quel plugin il souhaite faire cette action ;
- Fenêtre d'un plugin : cette fenêtre dépendra du plugin qui sera lancé. Elle permettra à l'utilisateur de créer un document, dont le type est fonction du plugin choisi.

Fonctionnalités/éléments

Développement du module de création de projet

But du module : Permettre aux utilisateurs de créer un nouveau projet

Caractéristiques du module : Ce module permettra à un utilisateur de créer un projet, en spécifiant son nom. Ce projet sera présenté « physiquement » sur le téléphone par un dossier au même nom. Par la suite viendront s'ajouter des documents, créés à partir des différents plugins. De plus chaque projet disposera de son propre fichier XML dans lequel on enregistrera des informations utiles sur celui-ci :

- Description du projet ;
- Les fichiers présents dans le dossier du projet. Chaque fichier aura comme informations :
  - Son nom ;
  - Son type : type qui indique le plugin avec lequel ce fichier a été créé.
  - Ses tags : Une liste des tags ajoutés à ce fichier (pour plus tard)

On devra avoir une page (Activity) indiquant la liste des projets et permettant d'en créer un nouveau. Ensuite, en cliquant sur un des projets existants, on sera dirigé vers une nouvelle page : la page du projet sélectionné, indiquant son nom et sa description pour le moment.

Développement du module d'architecture en plugins

But du module : Permettre à l'application d'intégrer des plugins qui permettront de créer des documents de tous types.

Caractéristiques du module : Depuis la page d'un projet, on pourra créer un nouveau document. Cependant, la création de ce document dépendra du plugin que l'on aura choisi. L'application devra être capable, au moment où on clique sur « Créer un nouveau document », de rechercher les plugins installés sur le téléphone et de lancer celui sélectionné par l'utilisateur. Un plugin correspondra donc à une application sur le téléphone. Cependant, nous ne souhaitons pas qu'un plugin puisse être lancé en dehors de l'application. Il sera donc masqué sur le téléphone (non présent dans la liste des applications).

Il faudra également rendre possible la communication d'informations entre l'application principale et les plugins. Cette communication est importante car on souhaite indiquer, au plugin lancé, le nom du projet en cours et le chemin du dossier où sont répertoriés les documents relatifs au projet.

Pour suivre la même logique que précédemment (un projet = un dossier), nous souhaitons que chaque plugin ait son dossier (dans le dossier du projet) et que les documents créés soient stockés dans celui-ci. C'est-à-dire que les fichiers issus du plugin Texte seront stockés dans le dossier Texte (*MonProjet/Texte*).

Développement du module de traitement de texte

But du module : Permettre aux utilisateurs de saisir du texte directement via l'application, de l'enregistrer et de pouvoir le visualiser plus tard.

Caractéristiques du module : Ce module se comportera comme un éditeur de texte simple et efficace, permettant de créer un document, le sauvegarder et le charger. Quand un fichier a déjà été sauvegardé, l'éditeur ne demandera plus à chaque enregistrement le nom du fichier. De plus les fichiers se termineront toujours en « .txt », même si l'utilisateur utilise une extension différente ou n'en n'utilise pas.

Les fichiers textes ainsi créés seront stockés dans un dossier Texte, lui-même contenu dans le dossier du projet en cours (*MonProjet/Texte*).

Développement du module d'enregistrement et d'écoute audio

But du module : Permettre aux utilisateurs d'enregistrer une conversation audio et de l'écouter.

Caractéristiques du module : Ce module permettra d'enregistrer un fichier audio et de le réécouter ensuite. Nous allons pour cela utiliser les outils fournis par l'API Android à savoir MediaRecorder.

Les fichiers audios ainsi créés auront pour extension 3gp et seront stockés dans un dossier Son, lui-même contenu dans le dossier du projet en cours (*MonProjet/Son*).

Développement du module de capture d'image

But du module : Pouvoir capturer des photos lors d'un échange ou d'un rendez-vous.

Caractéristiques du module : Ce module utilisera, tout comme pour l'audio, une API Android et non externe. Les photos ainsi prises ne seront pas visibles dans la Galerie du téléphone, pour ne pas gêner l'utilisateur.

Les fichiers photos ainsi créés auront pour extension jpg et seront stockés dans un dossier Photo, lui-même contenu dans le dossier du projet en cours (*MonProjet/Photo*).

Développement du module de connexion au compte Google et liaison au Drive

But du module : Permettre aux utilisateurs de se connecter via leur compte Google afin de pouvoir enregistrer et synchroniser les données de l'application sur le drive.

Caractéristiques du module : Ce module devra tout d'abord permettre à un utilisateur de se connecter à l'application avec son compte Google. Il pourra également se déconnecter pour se connecter avec un autre compte afin de profiter des données Drive d'un autre utilisateur par exemple.

La fonction « upload » permet d'envoyer sur le Drive les données locales de l'application, en conservant un semblant (semblant, car le Drive utilise un système de fichier différent de l'ordinaire, tous les fichiers sont au même niveau et différencier par un DriveID) d'architecture de fichiers. La fonction « download » effacera le contenu local puis parcourra récursivement le contenu de l'application sur le Drive pour télécharger tous les fichiers dans leurs dossiers respectifs sur le smartphone.

Développement du module de galerie

But du module : Permettre à un utilisateur d'avoir un visuel de tous les documents présents dans un projet.

Caractéristiques du module : Ce module devra ajouter à la fenêtre d'un projet une galerie répertoriant la liste des documents associés à celui-ci. Cette galerie devra ajouter de nouvelles fonctionnalités :

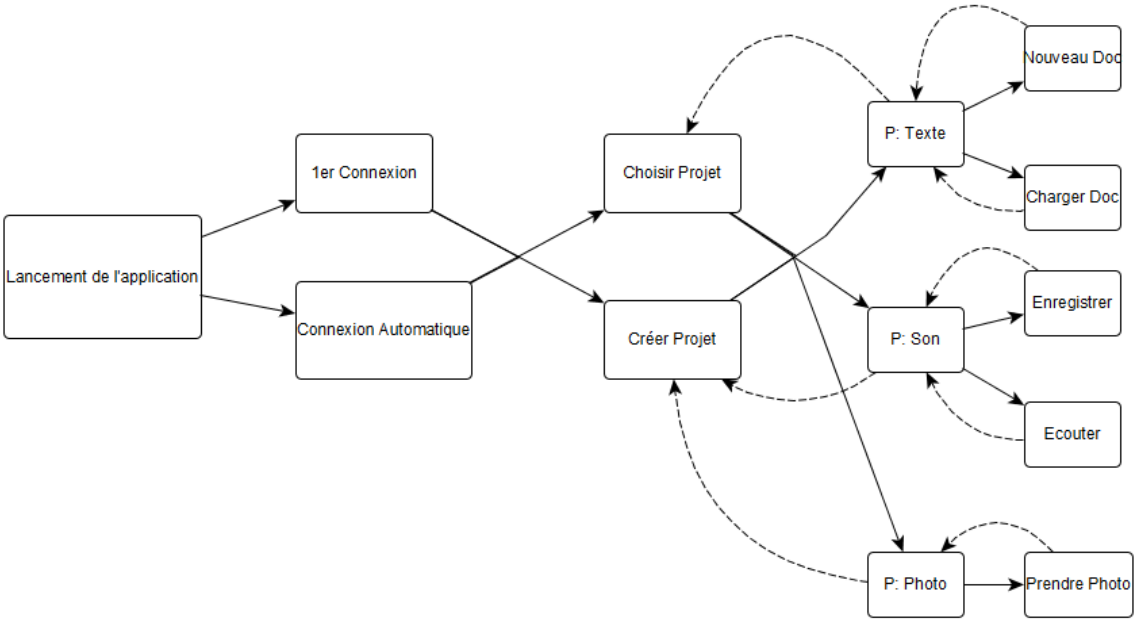
- Ajouter/Supprimer des tags à un document ;
- Ouvrir un document : ouvrir ce document dans le plugin avec lequel il a été créé. Il faudra donc modifier l'application, ainsi que tous les plugins, pour maintenant indiquer, quand on souhaite lancer un plugin, si l'on souhaite ouvrir un document ou non ;
- Supprimer le document.

Base de données

Notre application ne nécessite pas une base de données, vu la quantité minime d'informations que l'on a besoin de stocker. C'est pour cela que nous avons décidé d'opter pour des fichiers XML.

Processus d'utilisation de l'application

Nous avons schématisé le processus d'utilisation de l'application le plus simplement possible afin de le rendre compréhensible par n'importe quelle personne.



B) Diagrammes UML

Dans cette partie nous nous attarderons sur la conception de notre projet et plus particulièrement sur les diagrammes UML. Nous avons attaché une attention toute particulière à la conception du projet qui pour nous représente les fondations solides. Cette rigueur nous a permis de mettre en commun, sous forme de schémas, nos axes de développement et nos idées, afin qu'au final nous choisissions ensemble les axes définitifs. Ci-dessous nous allons vous exposer les diagrammes les plus importants à nos yeux de notre projet. Néanmoins, en annexe de ce document, vous retrouverez la totalité de nos diagrammes si vous souhaitez pleinement comprendre nos axes de développement.

1) Use case

Nous avons réalisé un diagramme « Use Case », autrement dit un diagramme recensant la totalité des actions qu'un utilisateur peut (comme par exemple : Ouvrir un projet -> Modifier un document -> Ajouter un tag. Il nous a semblé nécessaire de vous exposer ce diagramme en premier afin que vous puissiez avoir un aperçu du large panel d'actions que peut faire un utilisateur.

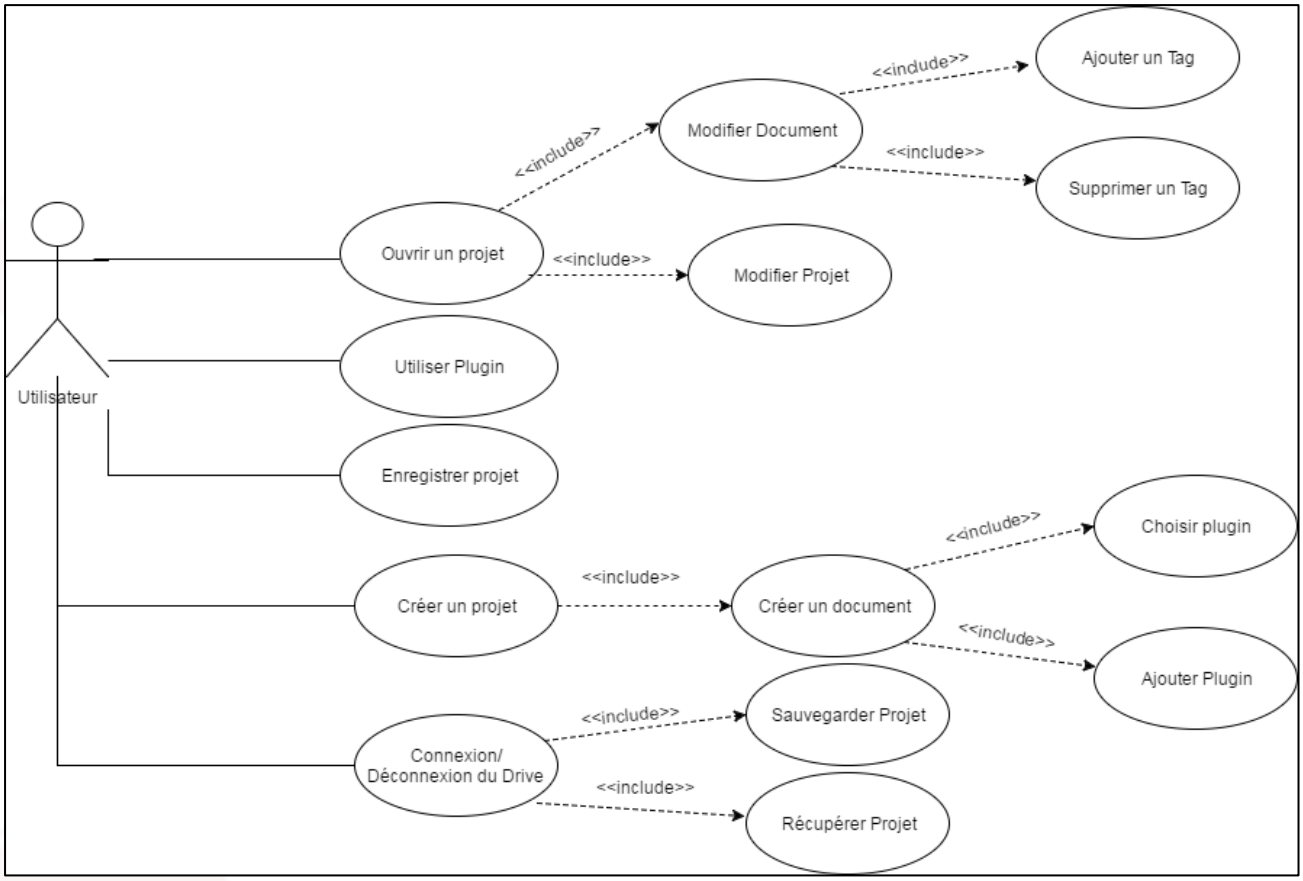


Figure 1 : Diagramme de Cas d'Utilisation

2) Diagramme de séquences

Nous avons également souhaité être rigoureux en réalisant un grand nombre de diagrammes de séquences pertinents, mettant en lien les actions d'un utilisateur et les conséquences de ses actes sur le ou les systèmes. Comme expliqué précédemment, le reste des diagrammes est disponible en annexe.

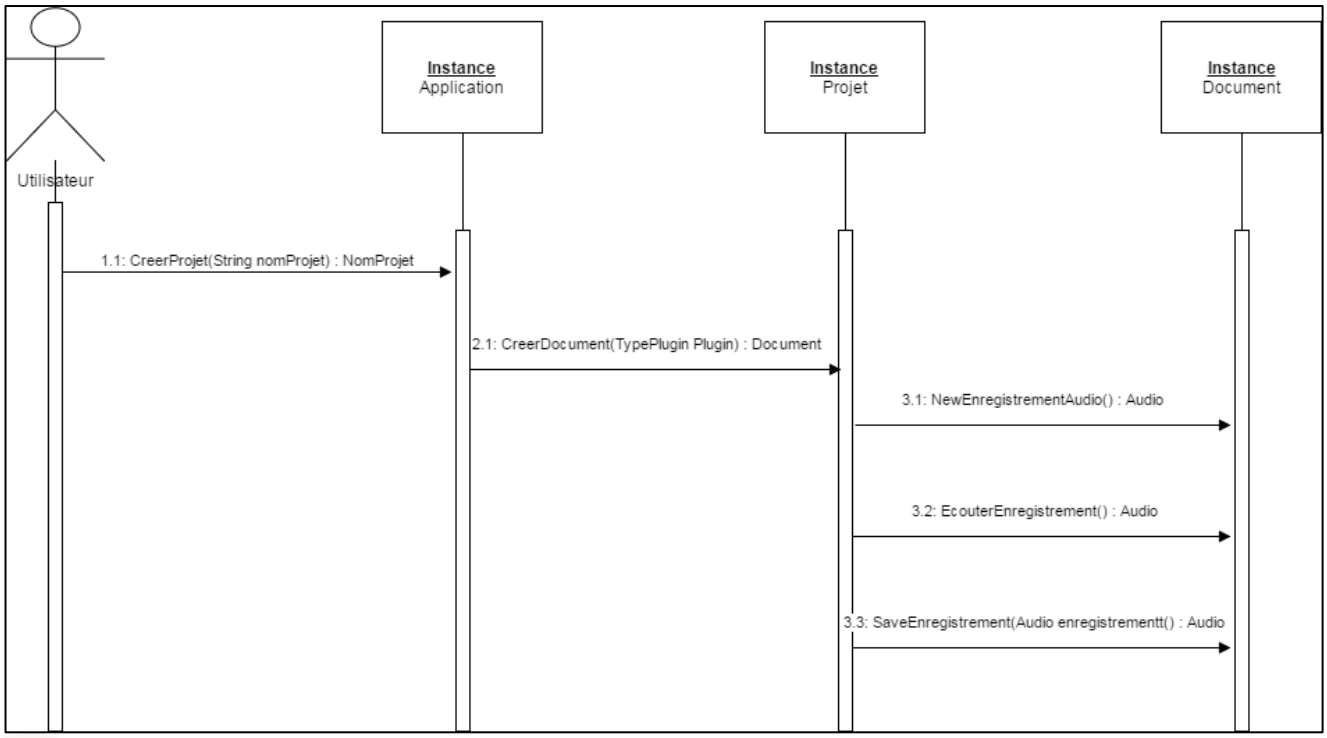


Figure 2 : Diagramme de séquence - Plugin Audio

Sur le diagramme ci-dessus, vous pouvez voir les systèmes impactés par les actions de l'utilisateur. Nous avons également fait apparaitre sur ce diagramme dans les signatures des méthodes les objets en adéquation avec les actions qui leurs sont associés.

Nous avons également voulu exposer un autre de nos diagrammes de séquence afin de vous montrer de façon plus schématique les actions de l'utilisateur lors de la sauvegarder de ses fichiers sur le Cloud. Le diagramme n'est certes pas compliqué à comprendre mais il montre encore une fois l'impact des actions de la sauvegarde sur les différents systèmes.

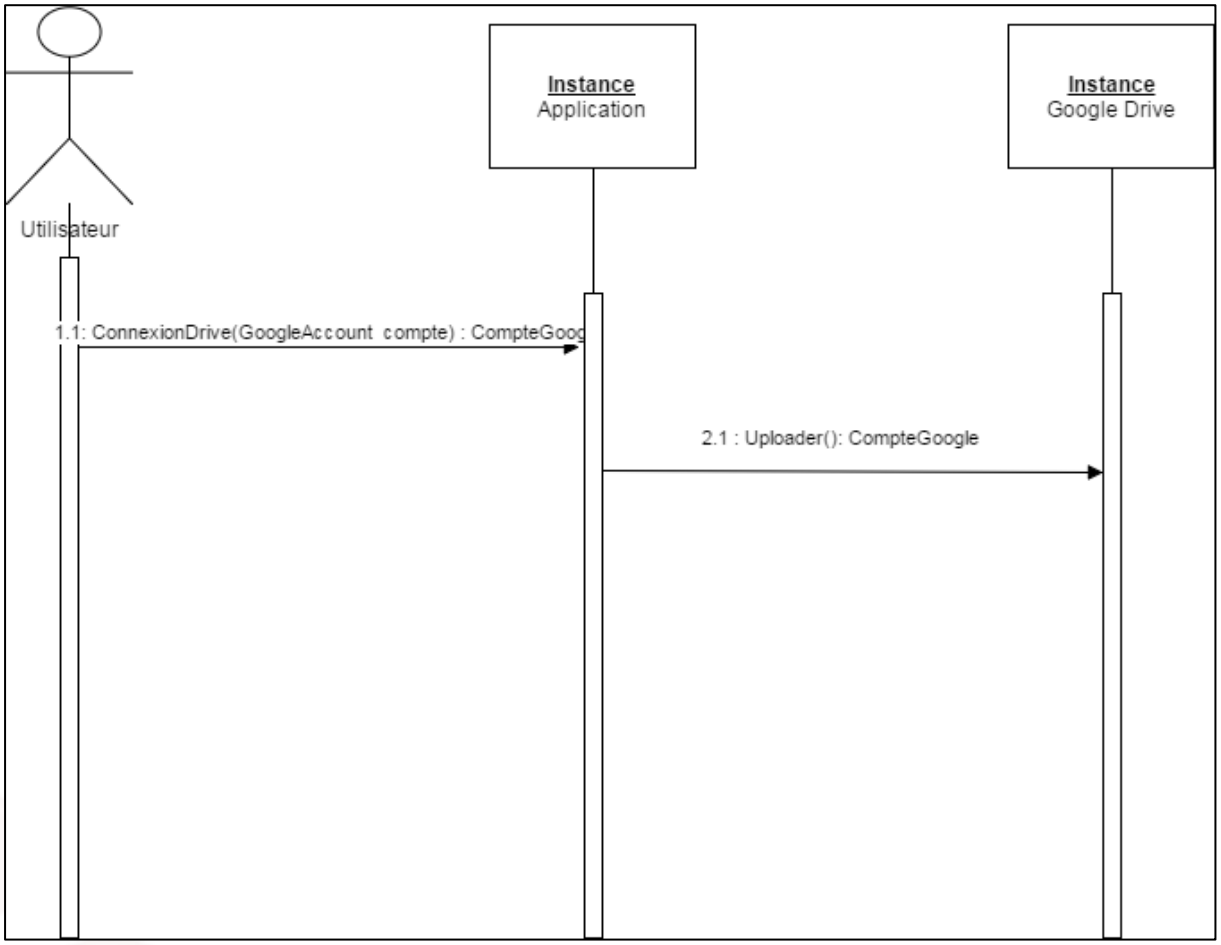


Figure 3 : Diagramme de séquence - Upload

Nous avons également réalisé d'autres types de diagrammes, moins communs que les autres mais tout aussi intéressants pour la compréhension et la structure de nos idées.

3) Diagramme Point de décision

Vous trouverez ci-dessous un diagramme point de décision. Ce diagramme met en évidence une situation bien particulière : la connexion à un compte Google.

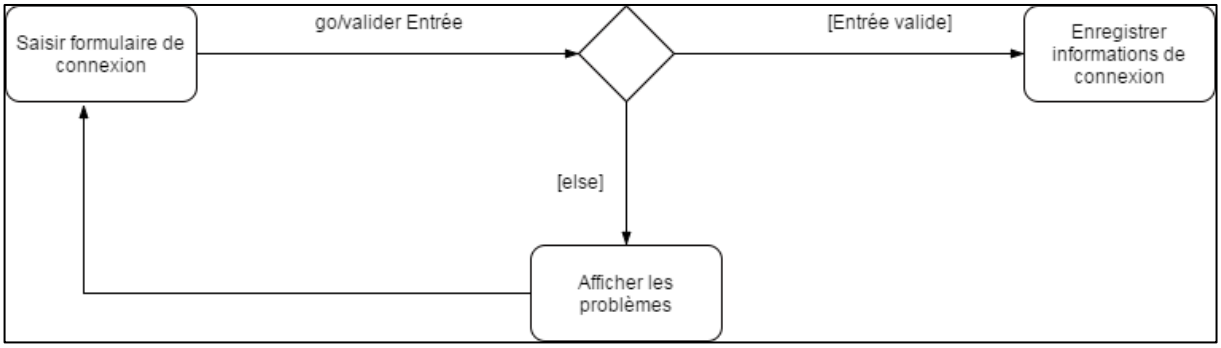


Figure 5 : Point de décision - Connexion à un compte

Dans ce diagramme, ce qui est mis en évidence est l'action du système en fonction de la véracité des informations remplies par l'utilisateur lors de la connexion. En cas de problème, le système va lui afficher l'erreur et le renvoyer à l'écran du saisi du formulaire de connexion.

4) Diagramme d'Etat

Enfin, nous voulions également présenter un diagramme d'Etat, symbolisant l'action d'affichage de documents d'un projet existant dans l'application d'un utilisateur. Nous avons une fois encore réduit notre diagramme à une action bien précise afin d'observer toutes les actions conduisant à ce but.

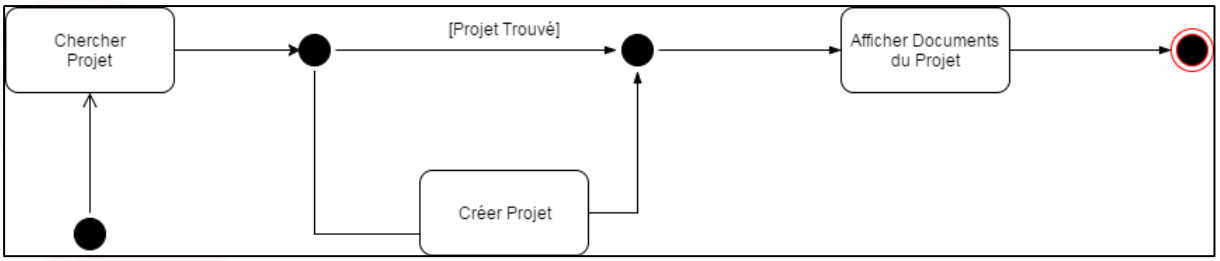


Figure 6 Diagramme d'Etat - Recherche de documents

C) Scénarios

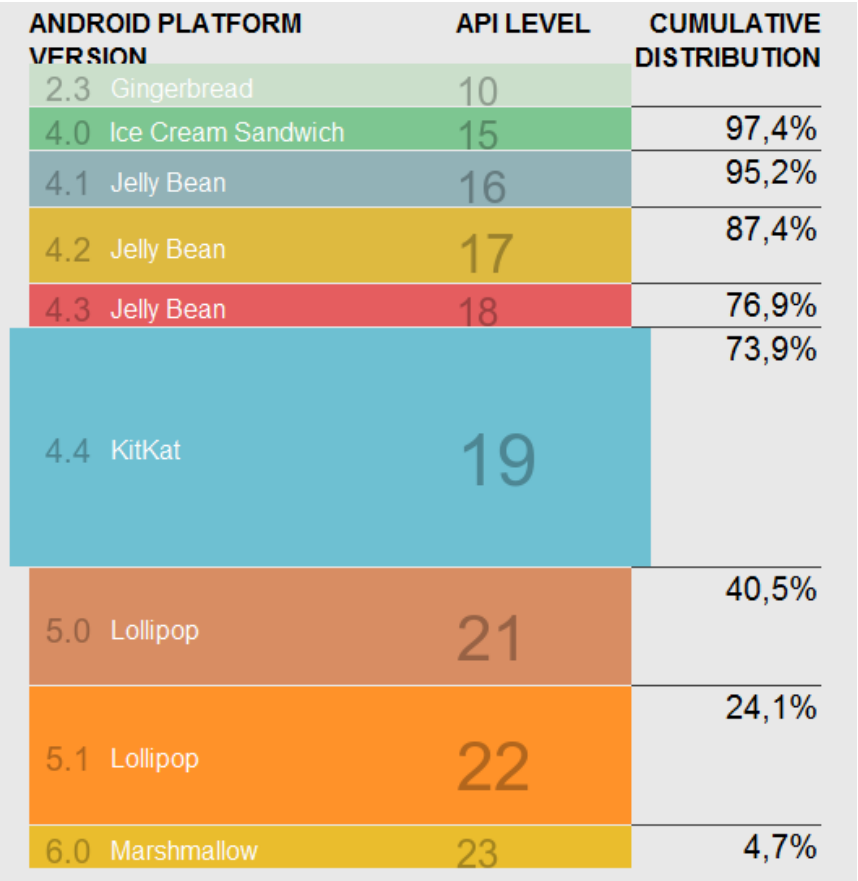
Nous avons également souhaité appliquer ce que nous avons appris, durant nos heures d'Ingénierie des besoins, en réalisant une dizaine de scénarios couvrant le spectre des situations relatives à notre projet. Ces scénarios nous permettent d'avoir des suites d'actions bien précises, que nous pouvons tester à travers notre application, et d'imaginer d'autres scénarios possibles. Vous retrouverez également la liste de tous ces scénarios en annexe (exploratoire, négatif, erroné, etc…), et vous trouverez le scénario général ci-dessous :

IDENTIFIANT	SPA06
NOM	Enregistrement des besoins via l'application
AUTEUR	Mr VALVERDE & Mr DEI
VERSION	V 1.0.0
HISTORIQUE	-
PRIORITE	High
CRITIQUE	High
SOURCE	Projet d'année
RESPONSABLE	Mr Crescenzo
DESCRIPTION COURTE	L'utilisateur souhaite capturer des besoins via l'application
TYPE DE SCENARIO	Scénario Général
BUTS ASSOCIES	Recueillir les besoins du client
ACTEURS	Utilisateur, application, drive, base de données
PRE-CONDITIONS	L'utilisateur possède l'application et un compte Google
POST-CONDITIONS	L'utilisateur enregistre le projet
RESULTAT	Les besoins ont été recueillis
ETAPES DU SCENARIO	1. L'utilisateur se connecte à l'application 2. L'utilisateur crée un projet 3. L'utilisateur recueille les besoins du client via les différents plugins disponibles 4. L'utilisateur sauvegarde son projet sur son drive
QUALITES ATTENDUES	Capturer les besoins du client de manières précises
LIENS AVEC D'AUTRES SCENARIOS	
INFORMATIONS SUPPLEMENTAIRES	

Nous nous sommes efforcés de garder la rigueur que nous avions pendant les TPs afin de l'appliquer ici pour obtenir des scénarios de qualités et intuitifs pour le lecteur.

D) Choix techniques

Nous allons vous détailler dans cette partie les points techniques relatifs à ce projet. Notre premier choix technique était de décider sur quelles versions notre application serait utilisable.



Distribution des versions d'Android

Pour toucher un maximum d'utilisateurs potentiels, nous avons décidé que la version minimale d'Android serait la 4.4 KitKat. Concernant la version maximale, nous sommes allé jusqu'à la version 7.1 Nougat.

a. Architecture en plugins

Le fait de disposer d'une architecture en plugins était un des points les plus importants, imposés par Monsieur CRESCENZO. Au départ, nous pensions que nous allions réutiliser ce que nous avions fait en Programmation Avancée au semestre dernier. Cependant, en recherchant sur internet, très peu d'informations étaient disponibles sur les applications Android à plugins (de nombreux tutoriels étaient soient incomplets, soient erronés). Une chose qui paraissait facile au départ s'est très vite transformer en quelque chose de compliqué.

Nous avons, après maintes recherches, trouvé une solution qui « simule » un fonctionnement en plugins. Le procédé utilisé est le même que celui que l'on retrouve quand une application nous demande : « Avec quelle application souhaitez-vous ouvrir ce document ? » et qu'une liste apparaît avec un certain nombre d'application. En cliquant sur le bouton pour créer un nouveau document (c'est-à-dire ouvrir un plugin), voici ce qui s'affiche :

L'application « principale » va chercher sur le téléphone toutes les applications possédant l'Intent : captureDesBesoins.plugin. Pour qu'un plugin soit visible de l'application, il doit donc spécifier ce fameux Intent dans son AndroidManifest.xml :

```
<!-- Intent pour pouvoir lancer ce Plugin depuis l'application principale -->
<intent-filter>
    <!-- Nom qui sera appelé par l'application principale -->
    <action android:name="captureDesBesoins.plugin" />
    <category android:name="android.intent.category.DEFAULT" />
</intent-filter>
```

La ligne « category android :name » permet d'indiquer qu'on ne souhaite pas que l'application soit visible dans le menu des applications du téléphone (pour ne pas qu'un plugin soit lancé à l'extérieur de l'application principale).

Voici donc le code qui permet de trouver les plugins et de lancer l'Activity de celui qui sera sélectionné par l'utilisateur :

```
Intent intent = new Intent("captureDesBesoins.plugin");

// Bundle pour passer en paramètre le nom du projet
Bundle b = new Bundle();

// Nom du projet sélectionné
b.putString("nomProjet", nomProjet);
// Chemin du projet sélectionné
b.putString("cheminProjet",
GestionnaireFichiers.getCheminProjet(this, nomProjet));

intent.putExtras(b);

String title = "Choisir un Plugin";
// Un chooser pour choisir le plugin
Intent chooser = Intent.createChooser(intent, title);

if (intent.resolveActivity(getPackageManager()) != null) {
    // Lancement du Chooser
    startActivity(chooser);
}
```

Le Bundle nous permet de passer des paramètres à l'Activity du plugin sélectionné, indispensable pour indiquer, par exemple, le chemin où tous les documents doivent être stockés.

Le Guide du développeur (fourni en annexe) indique comment créer un plugin et reprend ces informations-là.

b. Stockage de données : base de données/XML

Au départ, il était prévu d'intégrer à l'application une base de données pour enregistrer des informations sur les projets. Cependant, très vite nous nous sommes rendu compte que nous n'avions pas besoin de stocker énormément de données. Ce que Monsieur CRESCENZO souhaitait c'est qu'on puisse tagger des fichiers. Du coup nous avons choisi de stocker ces informations dans un fichier XML.

Chaque projet possède son propre fichier XML, comportant comme informations :

- Description du projet
- Liste des fichiers associés :
  - Fichier1
    - Nom
    - Type (correspond au plugin avec lequel il a été créé)
    - Tags
  - …

Pour créer ce fichier XML, nous avons choisi d'utiliser XmlSerializer qui propose une syntaxe claire. Voici à quoi ressemble le code qui nous permet de créer le fichier XML au départ (quand le projet vient tout juste d'être créé) :

```
public static String createXML(String nomProjet) throws IOException {
    XmlSerializer xmlSerializer = Xml.newSerializer();
    StringWriter writer = new StringWriter();
    xmlSerializer.setOutput(writer);

    // Début du document XML
    xmlSerializer.startDocument("UTF-8", true);

    // <projet nom = '..'>
    xmlSerializer.startTag("", BalisesXML.PROJET.xml);
    xmlSerializer.attribute("", BalisesXML.NOM.xml, nomProjet);

    //      <description>
    xmlSerializer.startTag("", BalisesXML.DESCRIPTION.xml);
    xmlSerializer.text("Pas de description...");
    //      </description>
    xmlSerializer.endTag("", BalisesXML.DESCRIPTION.xml);

    //      <fichiers>
    xmlSerializer.startTag("", BalisesXML.FICHIERS.xml);
    //      </fichiers>
    xmlSerializer.endTag("", BalisesXML.FICHIERS.xml);

    // </projet>
    xmlSerializer.endTag("", BalisesXML.PROJET.xml);

    // Fin du document XML
    xmlSerializer.endDocument();

    return writer.toString();
}
```

Si la création d'un document est simple, la récupération des informations est plus compliquée. Pour cela nous avons dû utiliser XmlPullParser qui analyse le fichier spécifié en parsant par EventType. Le parser différencie une balise ouvrante (START_TAG), du texte (TEXT) et une balise fermante (END_TAG). Avec ces données et en analysant le nom du tag à chaque fois, nous pouvons récupérer les informations de ce fichier XML.

c. Connexion au compte Google et liaison vers Google Drive

Au départ les données (documents/fichiers XML) étaient sauvegardées uniquement en local sur le téléphone. Pour permettre de visualiser ces données sur un autre téléphone, sur un ordinateur ou bien même de les partager avec quelqu'un d'autre, il a été nécessaire de trouver une solution pour envoyer ces informations sur une plateforme de stockage en ligne. Notre choix s'est vite porté sur l'utilisation de Google Drive, en accord avec Monsieur CRESCENZO.

Avant qu'une personne puisse envoyer des informations sur son Drive, il faut qu'elle se connecte avec son compte Google. Pour manipuler l'API Google, il faut d'abord s'enregistrer sur le site https://developers.google.com et donner plusieurs informations sur notre projet. Ensuite, nous devons choisir quelles API nous souhaitons utiliser, dans notre cas l'API Drive suffisait. Suite à cela, le site nous fournit une clé API unique que nous devons ajouter au AndroidManifest.xml de notre application :

```
<meta-data
        android:name="com.google.android.geo.API_KEY"
        android:value="AIzaSyC4hoGv1rxoz6gRNOryy7XXXXXXXXXXXXX" />
<meta-data
```

Cette clé nous permet de nous connecter avec notre propre compte Google pour tester notre programme. Chaque membre du projet devait donc générer sa propre clé.

Un des problèmes rencontrés était que lorsqu'on se déconnecter, l'application se souvenait de notre compte Google. Du coup, quand on souhaitait se reconnecter, l'application ne nous demandait pas de redonner nos informations. Cela était problématique dans le cas où on voudrait changer de compte. Une petite astuce a dû être mise en place pour, qu'au moment de la déconnexion, l'application « oublie » le compte de l'utilisateur. Par conséquent, à chaque fois que l'on souhaite se connecter, on repasse par la fenêtre de connexion.

Après s'être connecté, l'utilisateur peut choisir d'envoyer ou de récupérer des fichiers de son Drive. Nous avons dissocié ces deux fonctionnalités, un peu à l'image d'un Git où l'on peut soit Pull soit Push. Les classes relatives au Download et à l'Upload implémentent Runnable. L'utilisation de Threads est obligatoire car nous faisons des appels synchrones qui rendent le code bloquant sinon. Ces deux classes fonctionnent de façon récursive.

Un des problèmes rencontrés étaient que le Drive Google ne fonctionne pas de la même manière qu'un système de fichier classique. Sur le Drive tous les fichiers/dossiers sont au même niveau et sont différenciés par un DriveID unique. Pour simuler la notion de parent/fils, il faut indiquer au(x) fil(s) le DriveID du parent, rendant la recherche des fichiers à télécharger plus compliquée.

E) Résultat livré au client

Nous rendons un dépôt git contenant tous les sources du projet, accompagné de tous nos documents de conception.

Pour avoir un visuel de l'application finale, nous vous invitons à consulter le Guide de l'application, disponible en annexe, qui présente en détails chaque fenêtre.

Le code livré est entièrement commenté. Un Guide du développeur (annexe) est également disponible pour permettre à n'importe quelle personne de créer son propre plugin.

III) Gestion de projets

Des réunions régulières étaient organisées avec Monsieur CRESCENZO pour évaluer l'avancement du projet et nous conseiller sur les livrables à fournir. Vous trouverez en annexe les compte-rendu de chaque réunion. Ces documents, ainsi que les notes personnelles de l'encadrant, nous permettaient de visualiser ce qui avait été fait d'une réunion à une autre.

Très vite, nous avons décidé que deux personnes s'occuperaient en grande partie du code, et les deux autres de la conception. Ce choix s'est fait naturellement et a été validé par notre encadrant.

Pour pouvoir travailler tous ensemble sur le projet, nous avons utilisé un répertoire Git. La plateforme GitHub nous a permis de découper notre projet en tickets et ainsi pouvoir développer notre application petit à petit. Chaque module du Cahier des Charges étaient prévus pour, qu'à la fin de l'un d'entre eux, on puisse fournir un livrable complet et fonctionnel.

Nous avons développé sur Android Studio et nous nous servions de nos téléphones pour tester l'application.

IV) Conclusion

Notre application comprend les fonctionnalités suivantes :

- Création d'un projet ;
- Architecture en plugins pour la création de documents ;
- Un plugin pour créer un fichier texte ;
  - Enregistrer un fichier texte ;
  - Charger un autre fichier texte ;
- Un plugin pour enregistrer du son ;
  - Enregistrer/écouter un son ;
- Un plugin pour prendre une photo ;
- Une « galerie » listant tous les documents du projet ;
  - Ajouter/Supprimer des tags d'un document ;
  - Ouvrir le document avec le plugin adéquat ;
  - Supprimer le document ;
- Ajouter une description au projet ;
- Sauvegarde des informations relatives au projet dans un fichier XML ;
- Pouvoir se connecter à son compte Google ;
- Pouvoir upload ses fichiers sur le drive ;
- Pouvoir download des fichiers du Drive.

Nous avons répondu à toutes les attentes de notre client, Monsieur CRESCENZO. Cependant, notre démarche était de livrer un produit qui puisse être reprit facilement par d'autres personnes. De nouvelles fonctionnalités peuvent être ajoutées pour rendre l'application encore plus complète :

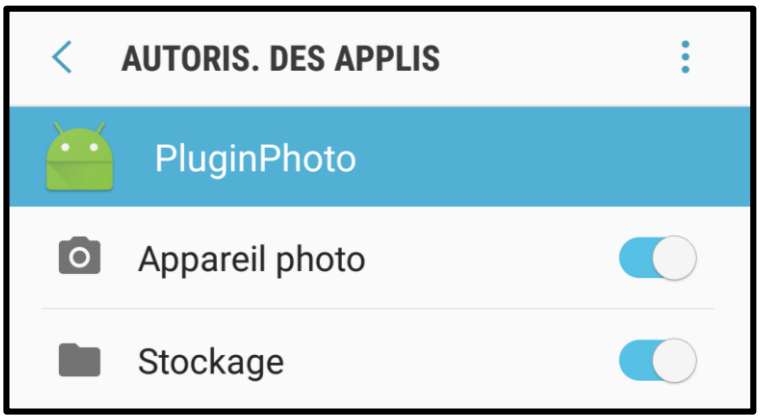
- Ajouter des barres de chargement au moment de l'upload/download ;
- Développer de nouveaux plugins (en prenant soin de suivre à la lettre le Guide du développeur) ;
- Ajouter des fonctionnalités en lien avec les tags sur les documents ;
- Développer un outil (plugin ?) permettant d'extraire automatiquement des données des documents enregistrés. Par exemple : récupérer des informations importantes d'un document texte.

V) Annexes

Guide d’utilisation

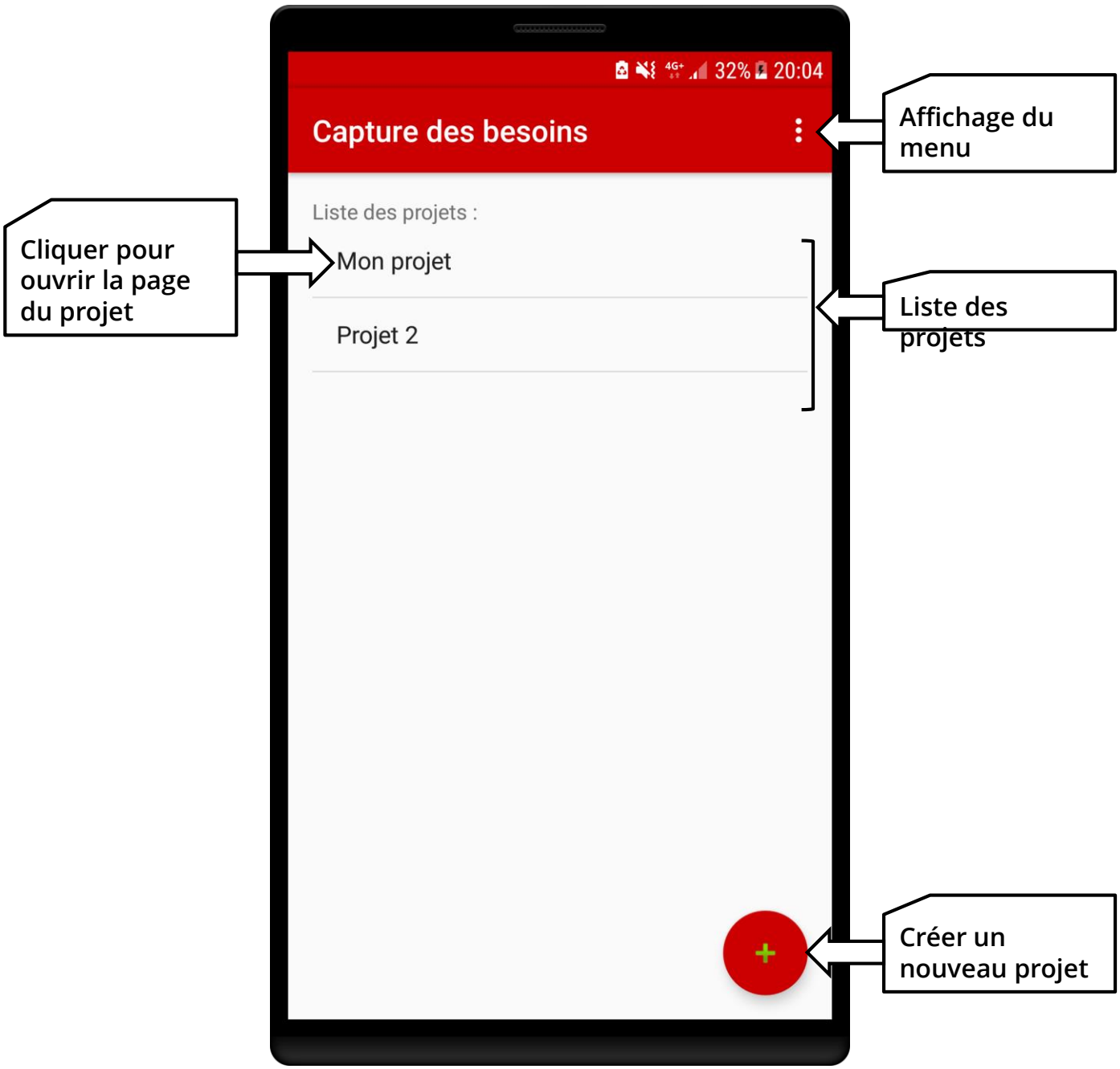
Informations sur l'application

- Application disponible sur Android
  - Version minimale : Android 4.4 KitKat (19)
  - Version maximale : Android 7.1 Nougat (25)
- Téléchargement des plugins :
  - Uniquement via Android Studio pour le moment. Il faut lancer le projet pour qu'il s'installe sur le téléphone.
  - Une fois le plugin installé, il faut activer les Autorisations (car en passant par Android Studio, les autorisations ne sont pas demandées au lancement). Pour cela, il faut se rendre dans Paramètres → Applications → *NomPlugin* → Autorisations

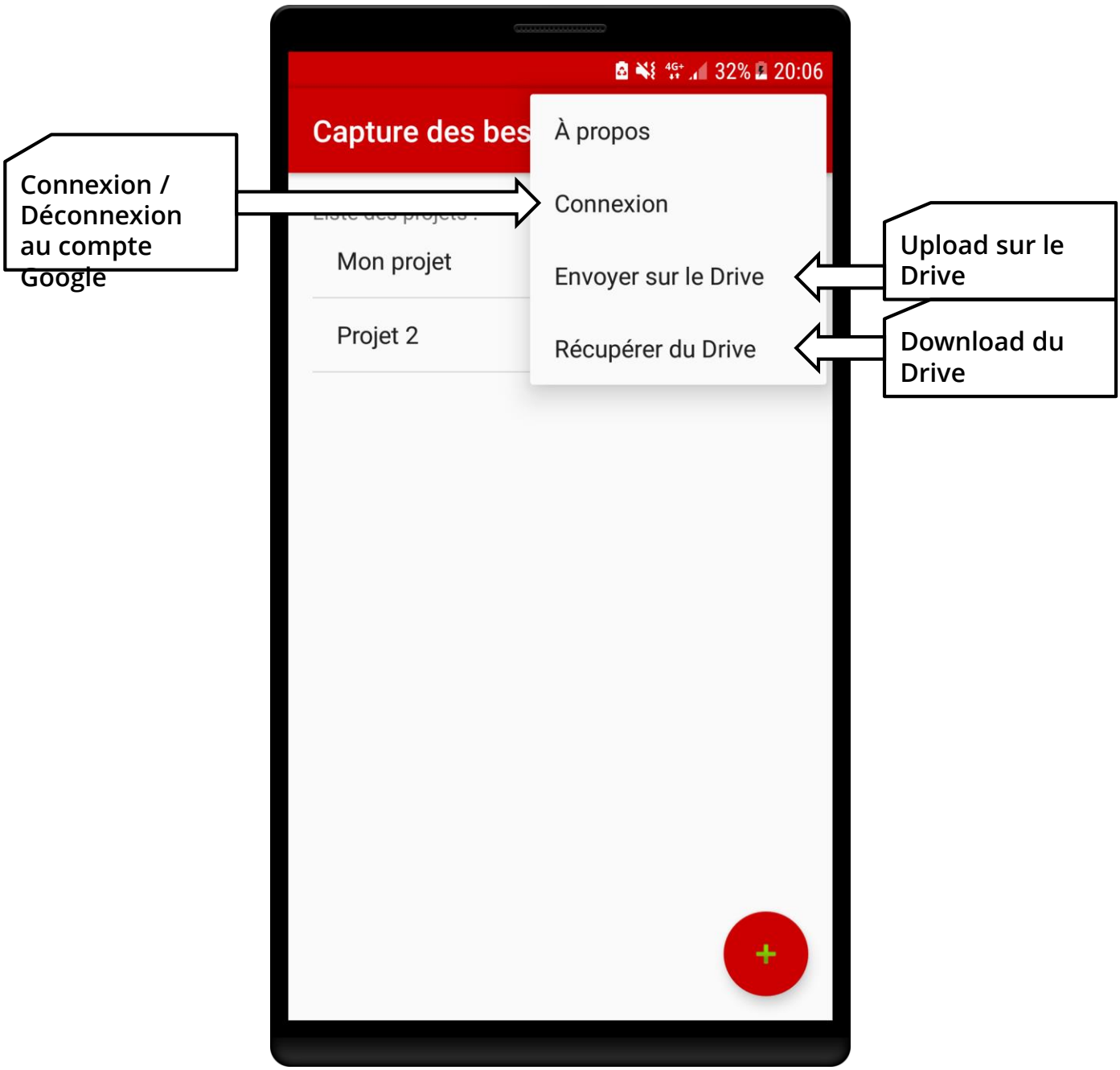


  - Le plugin est maintenant installé et sera utilisable depuis l'application principale.

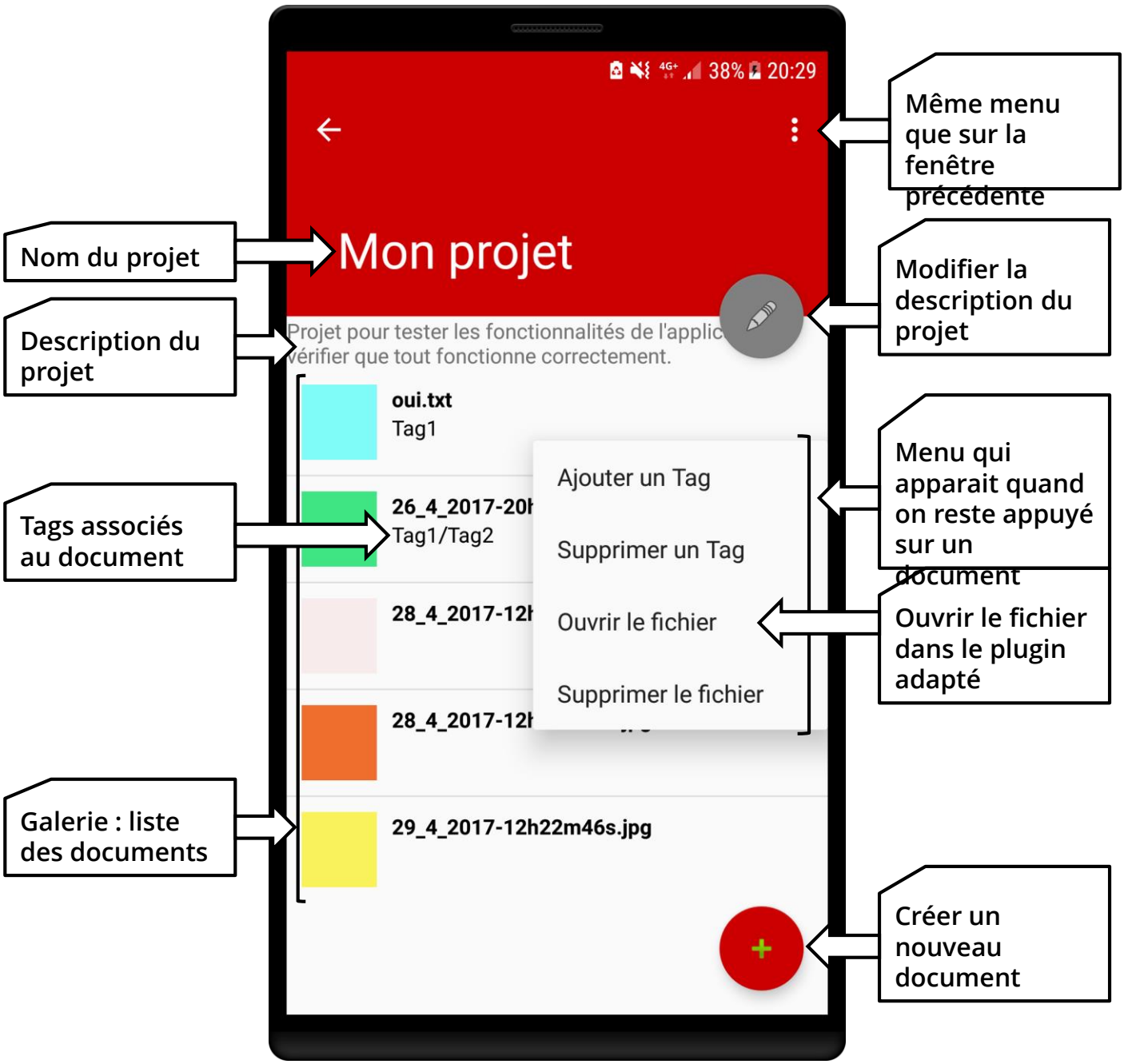
Fenêtre principale



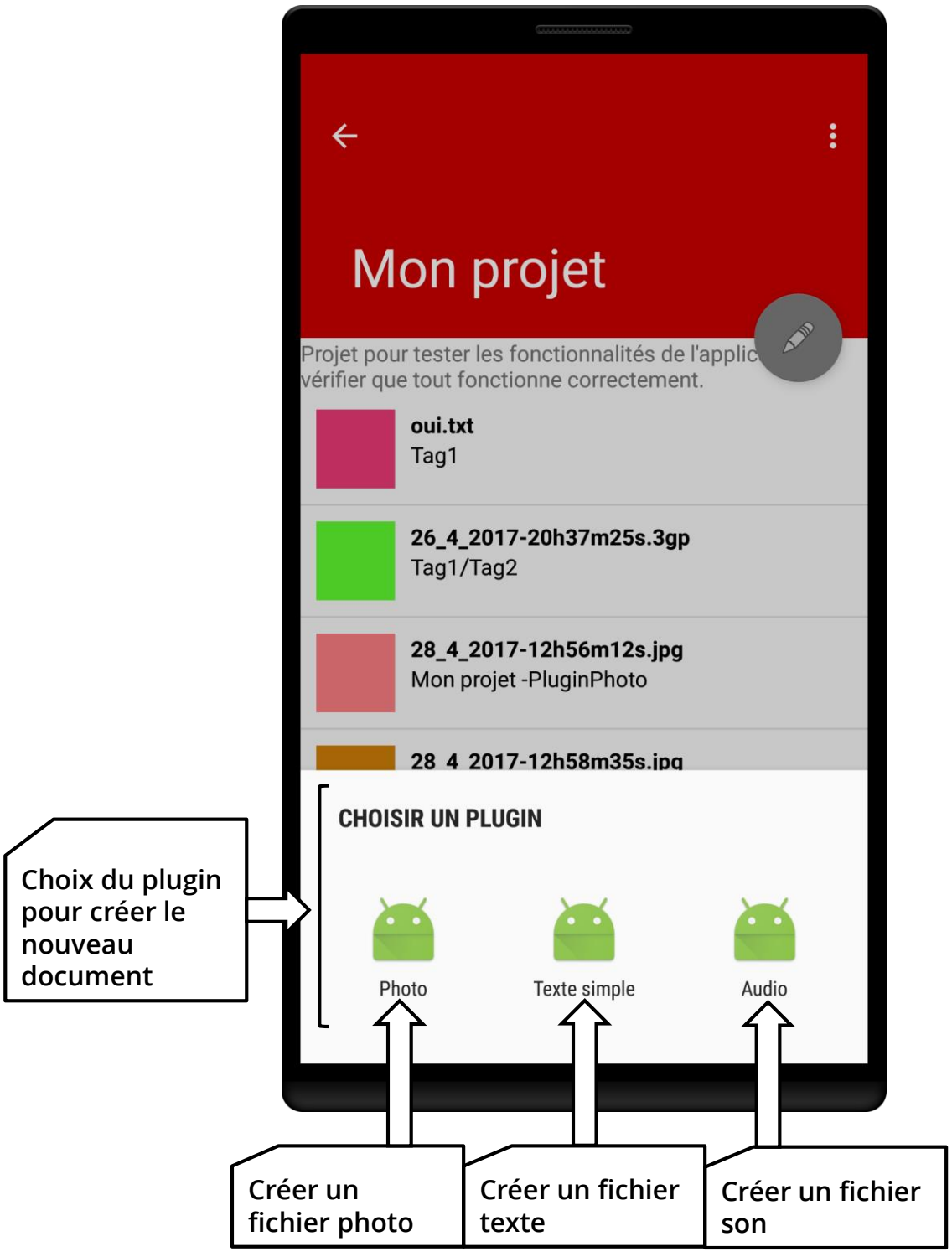
Fenêtre principale - Menu



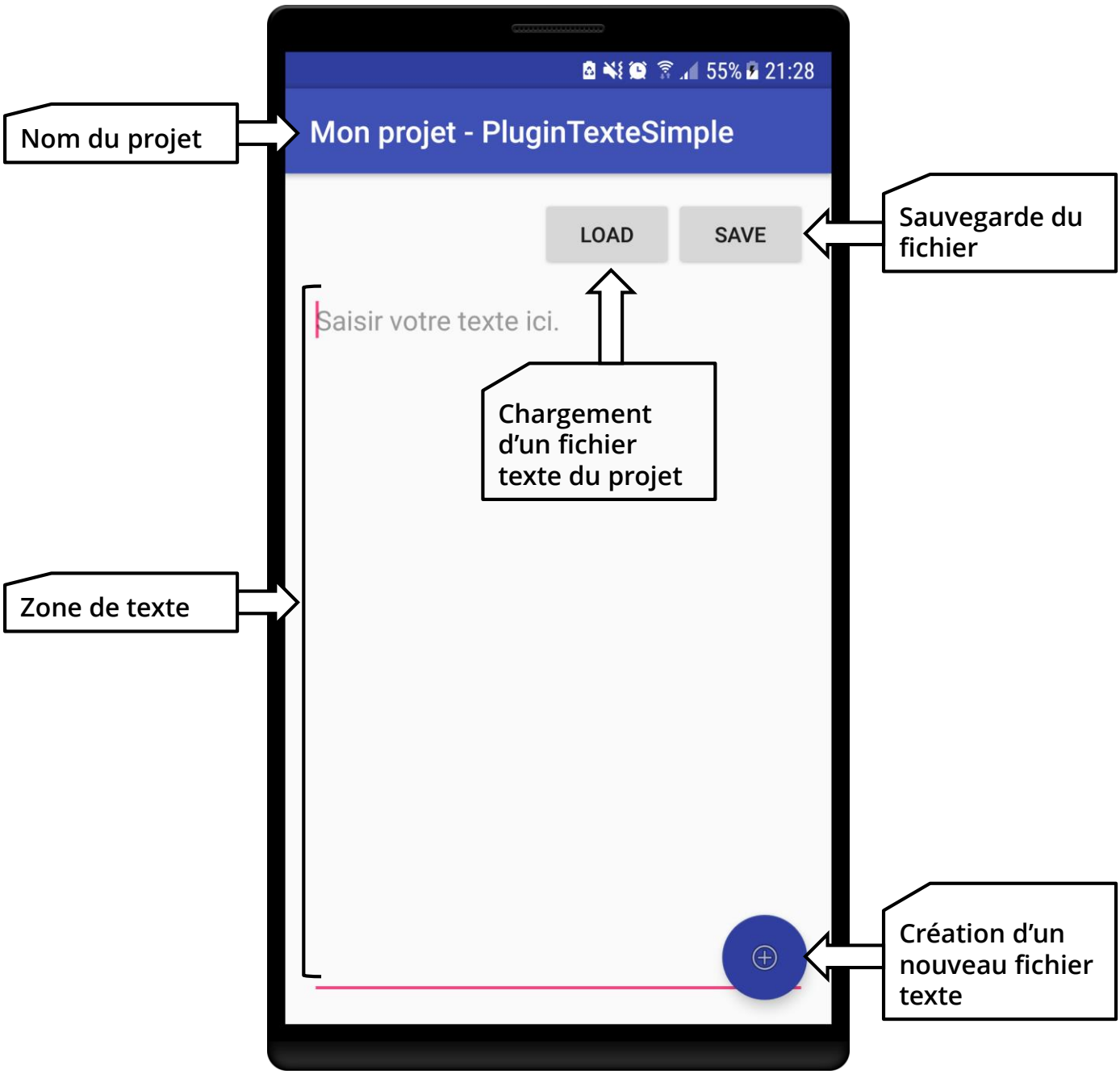
Fenêtre d’un projet



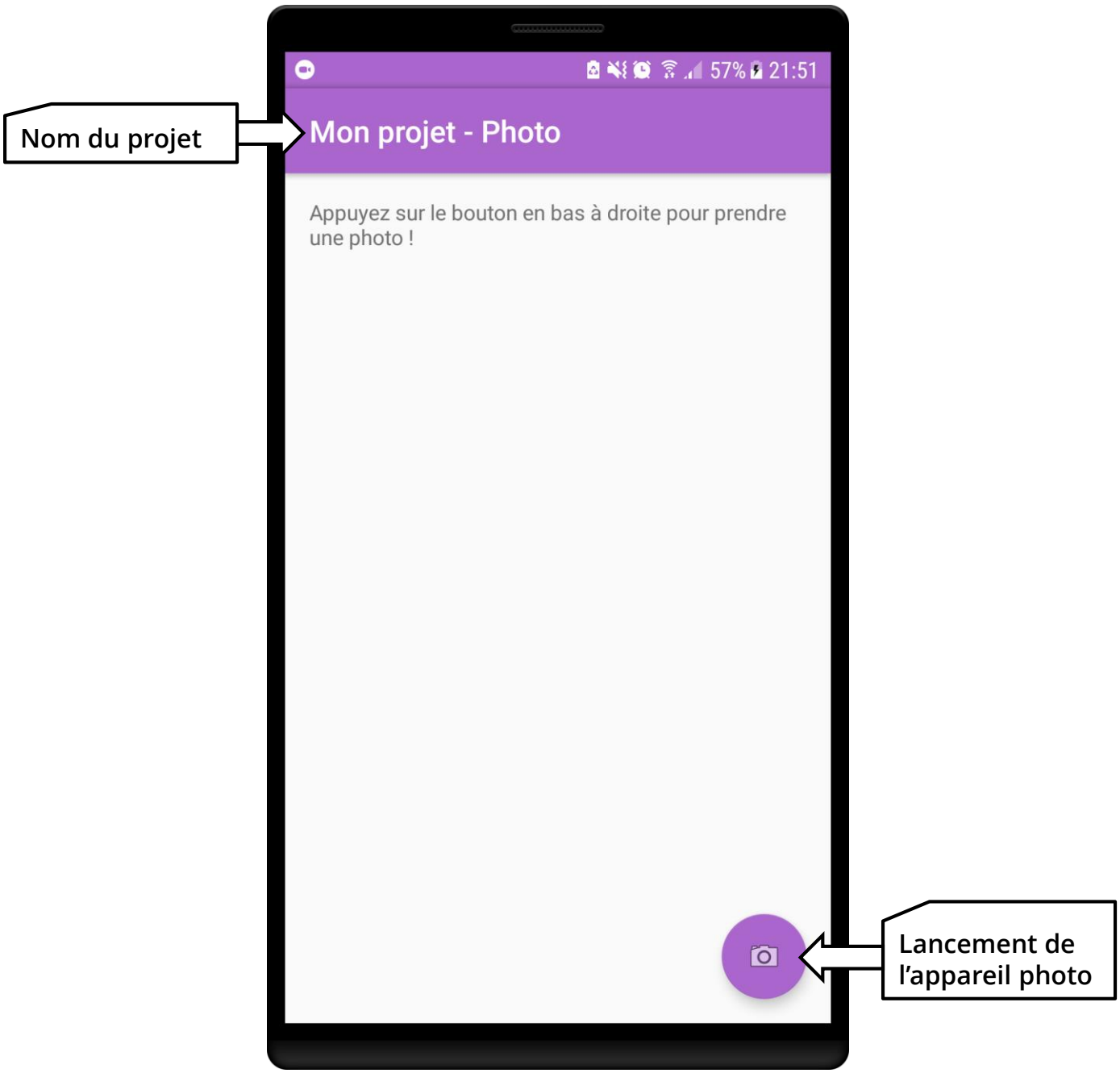
Fenêtre d’un projet – créer un nouveau document



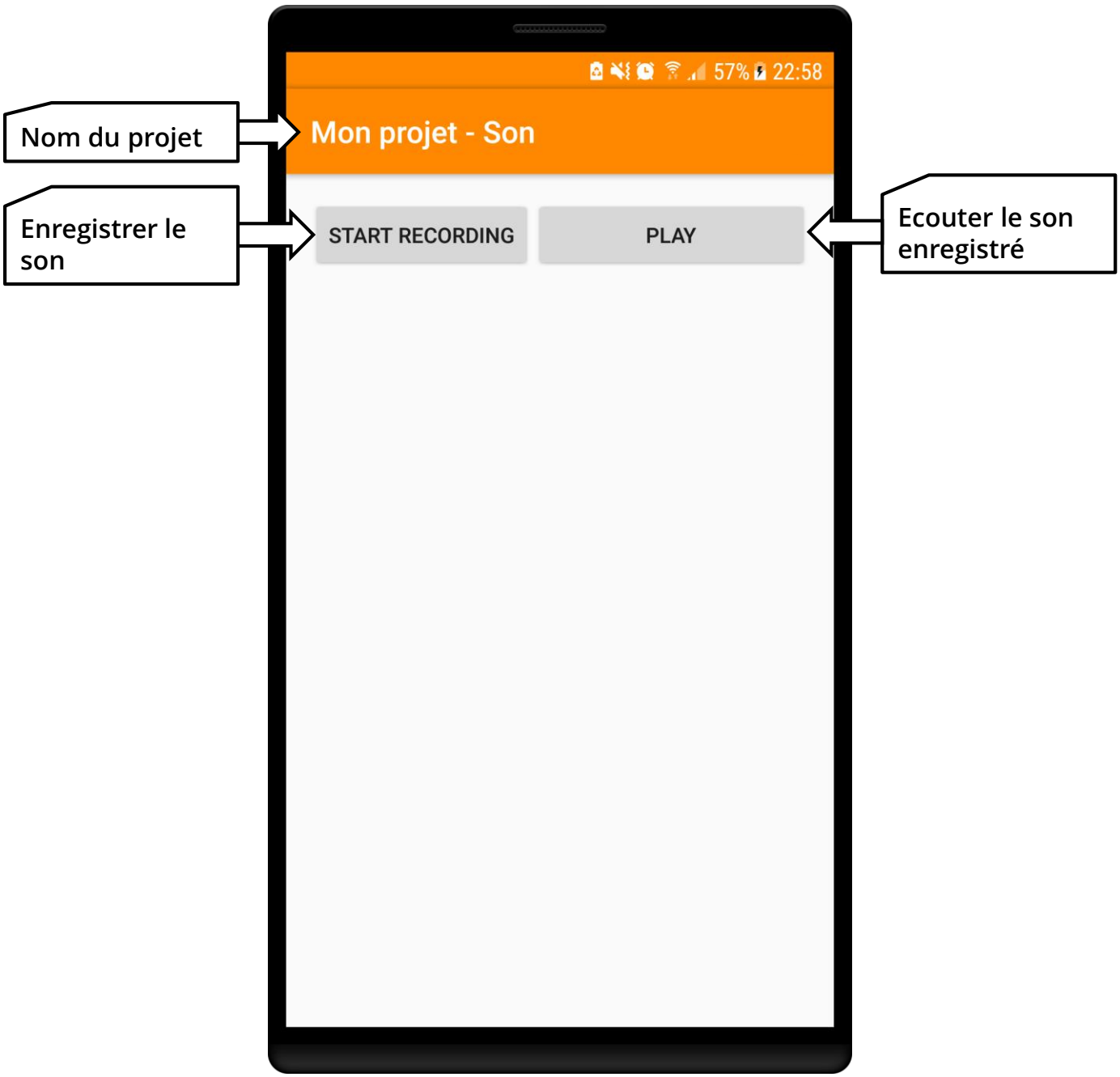
Fenêtre du plugin texte

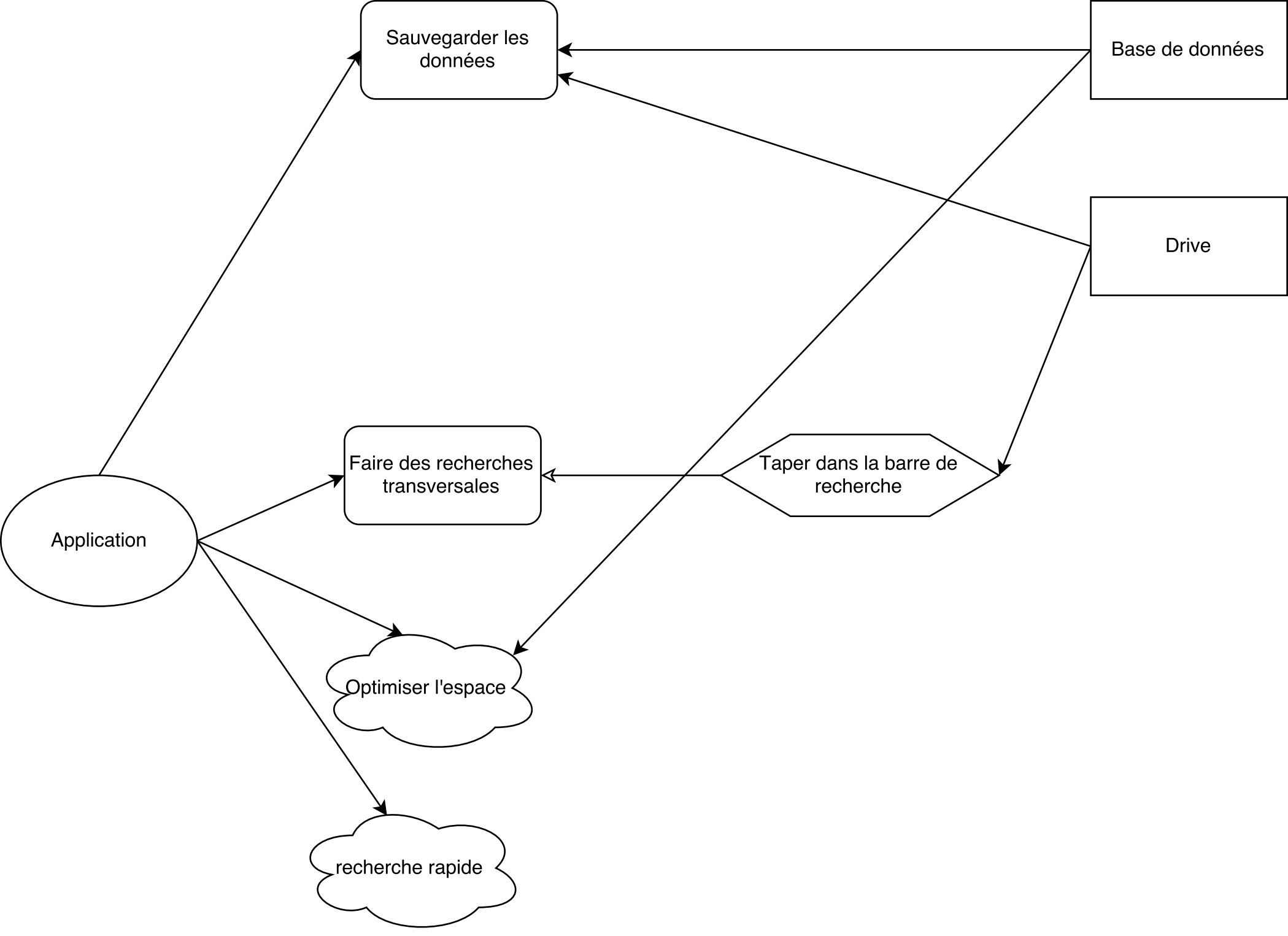


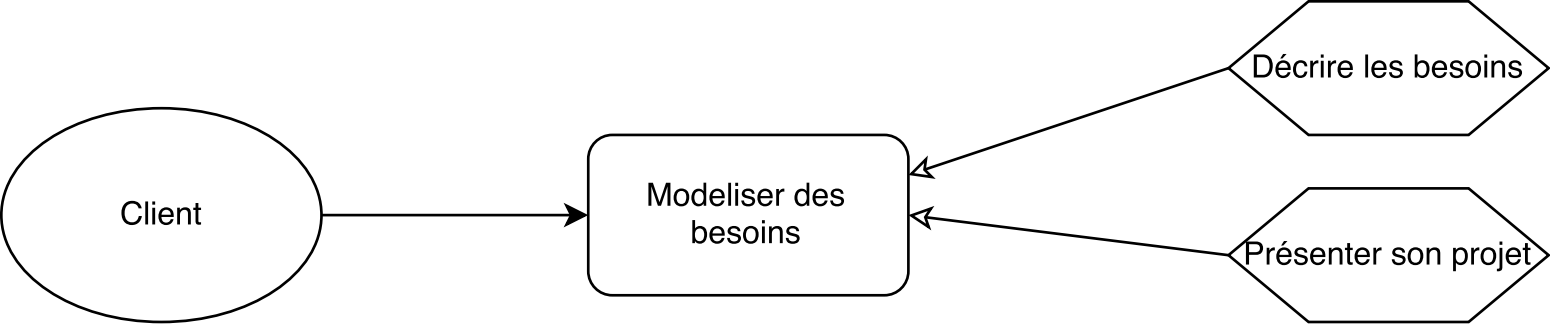
Fenêtre du plugin photo

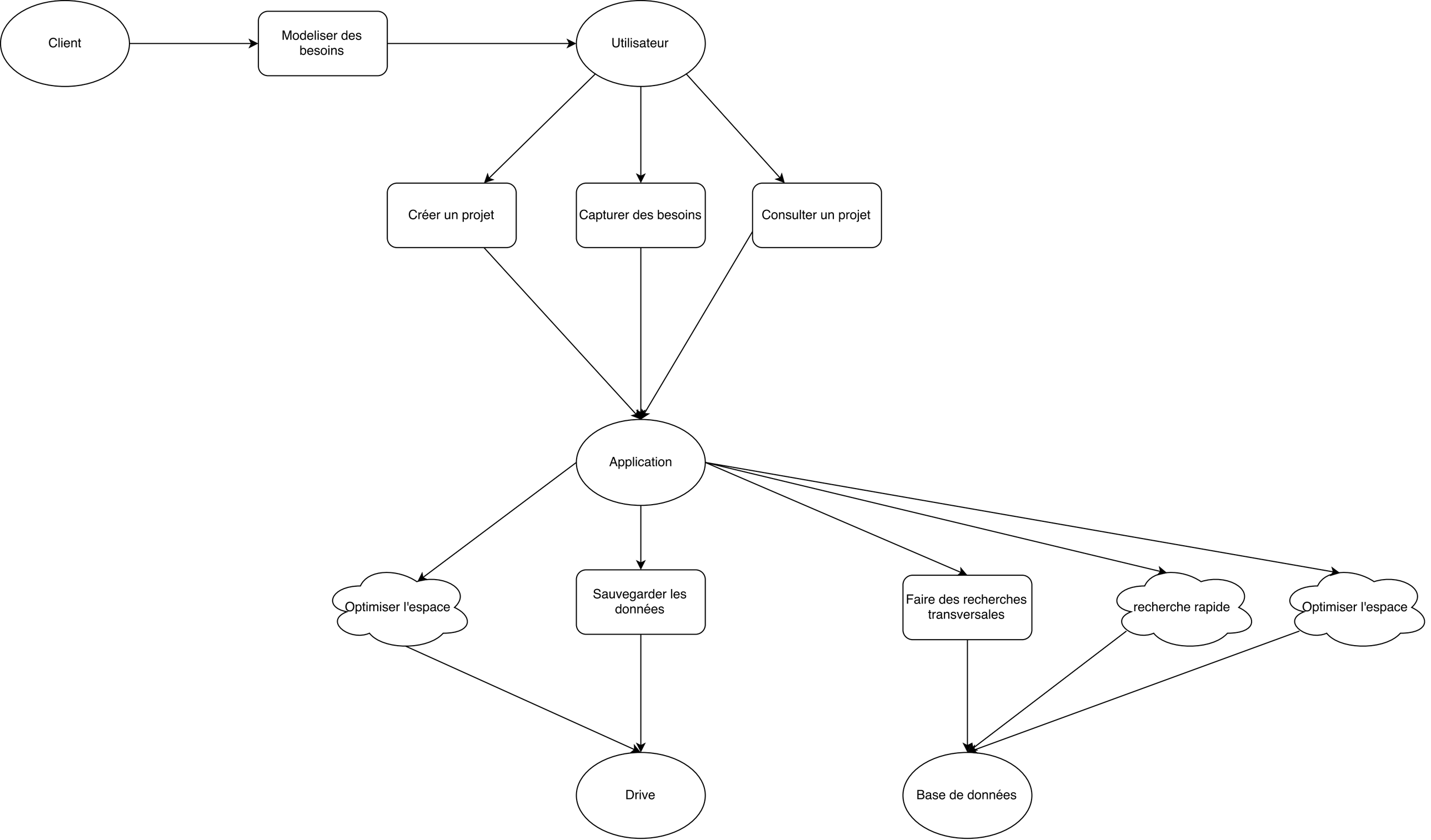


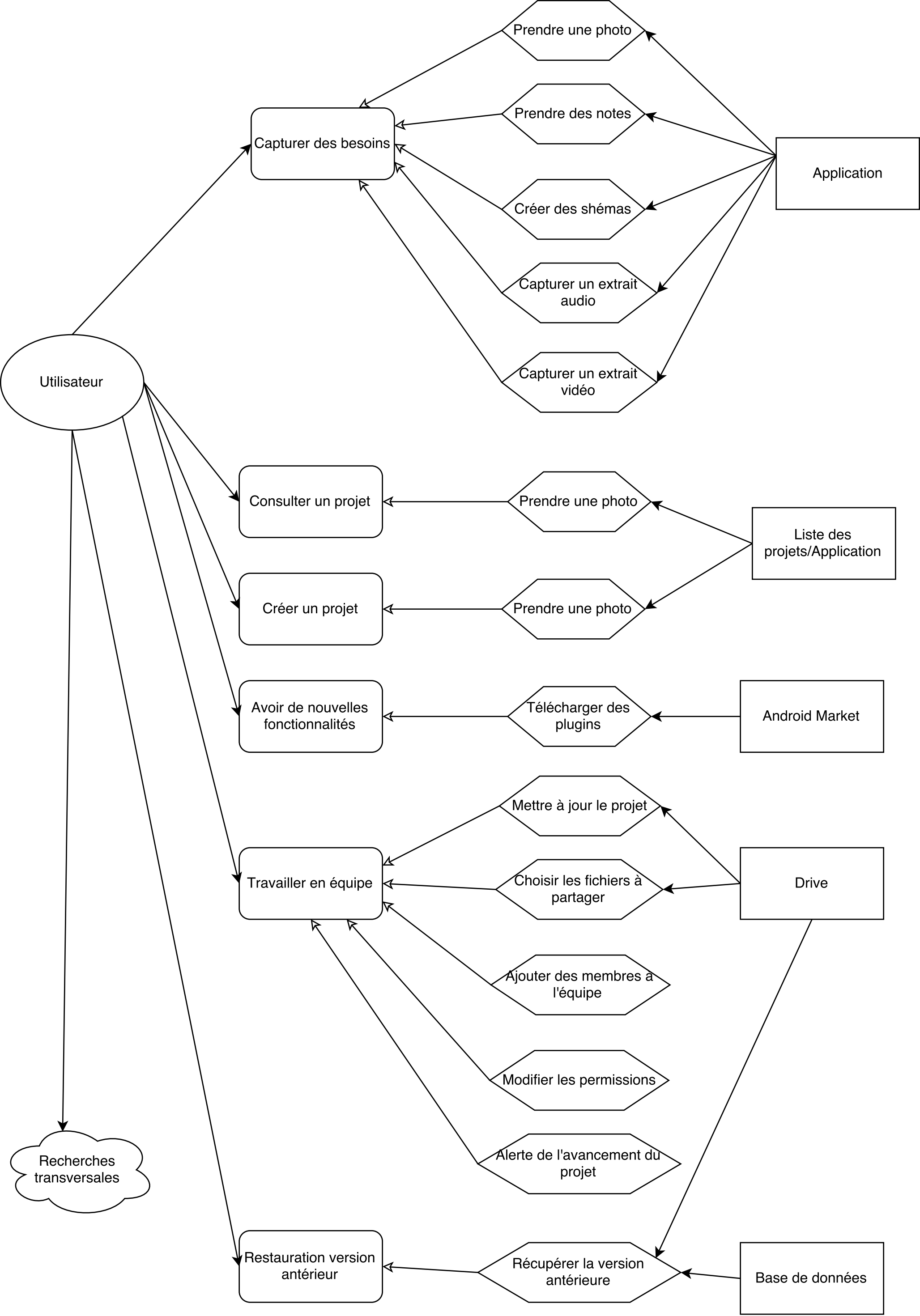
Fenêtre du plugin audio

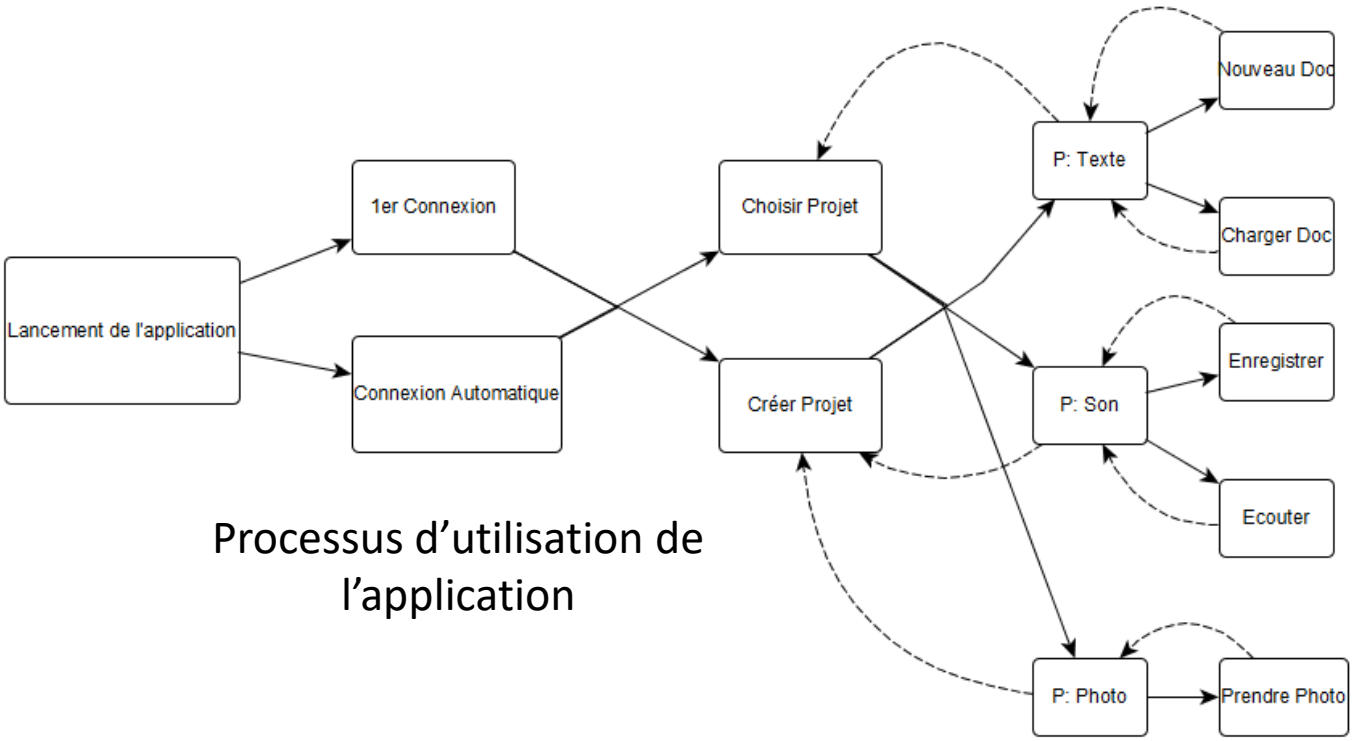



Sauvegarder les données
Base de données
Drive
Faire des recherches transversales
Taper dans la barre de recherche
Application
Optimiser l'espace
recherche rapide


Client
Modeliser des besoins
Décrire les besoins
Présenter son projet


Client
Modeliser des besoins
Utilisateur
Créer un projet
Capturer des besoins
Consulter un projet
Application
Optimiser l'espace
Sauvegarder les données
Faire des recherches transversales
recherche rapide
Optimiser l'espace
Drive
Base de données


Prendre une photo
Capturer des besoins
Prendre des notes
Application
Créer des shémas
Capturer un extrait audio
Capturer un extrait vidéo
Utilisateur
Consulter un projet
Prendre une photo
Liste des projets/Application
Créer un projet
Prendre une photo
Avoir de nouvelles fonctionnalités
Télécharger des plugins
Android Market
Mettre à jour le projet
Travailler en équipe
Choisir les fichiers à partager
Drive
Ajouter des membres à l'équipe
Modifier les permissions
Alerte de l'avancement du projet
Recherches transversales
Restauration version antérieur
Récupérer la version antérieure
Base de données



Processus d'utilisation de l'application

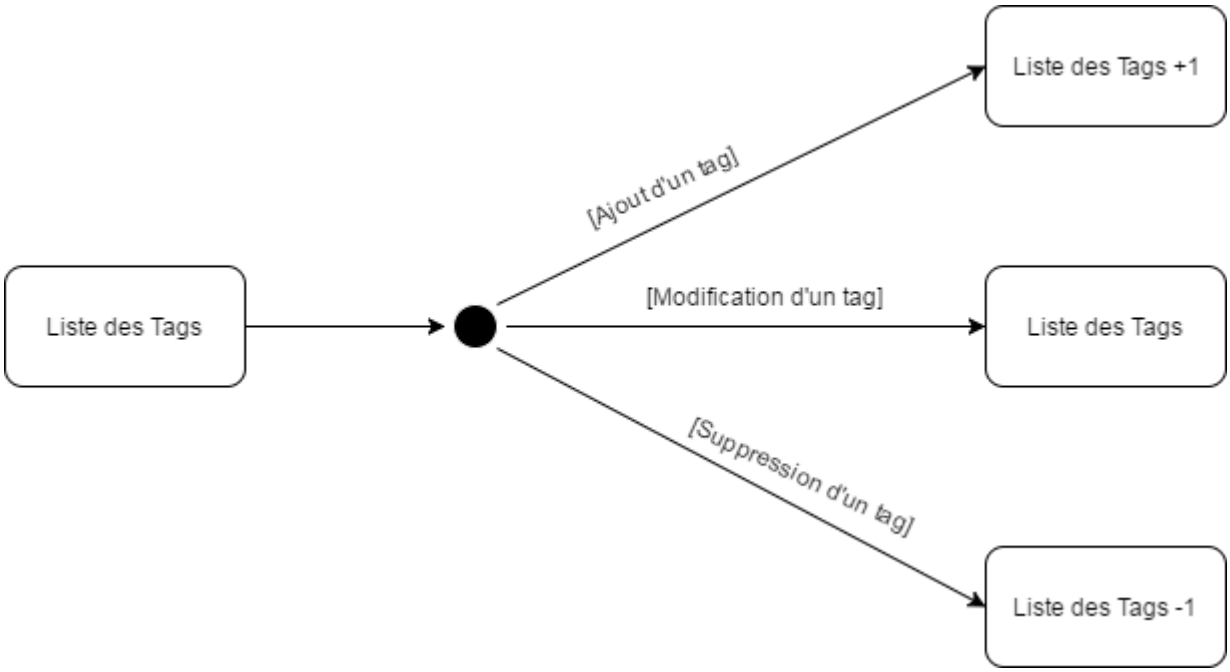


Diagramme état - tags

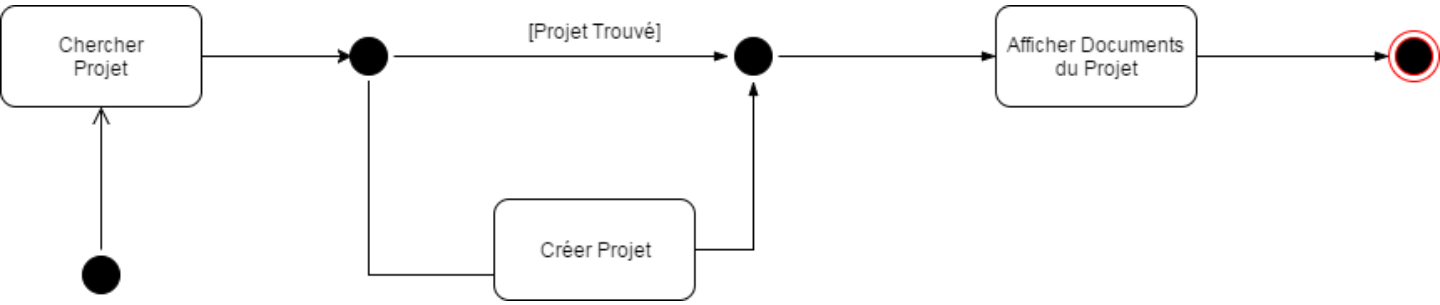
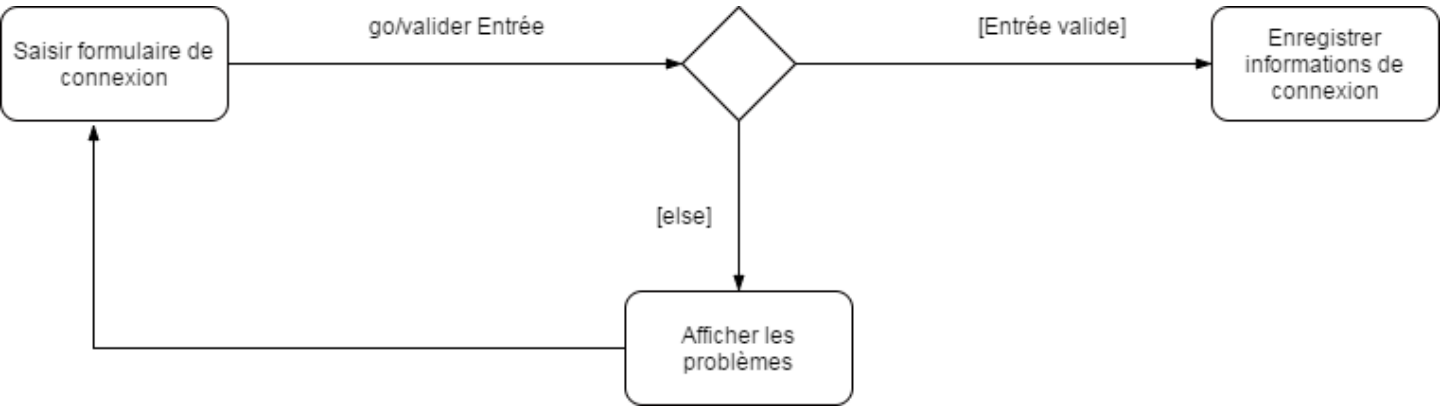
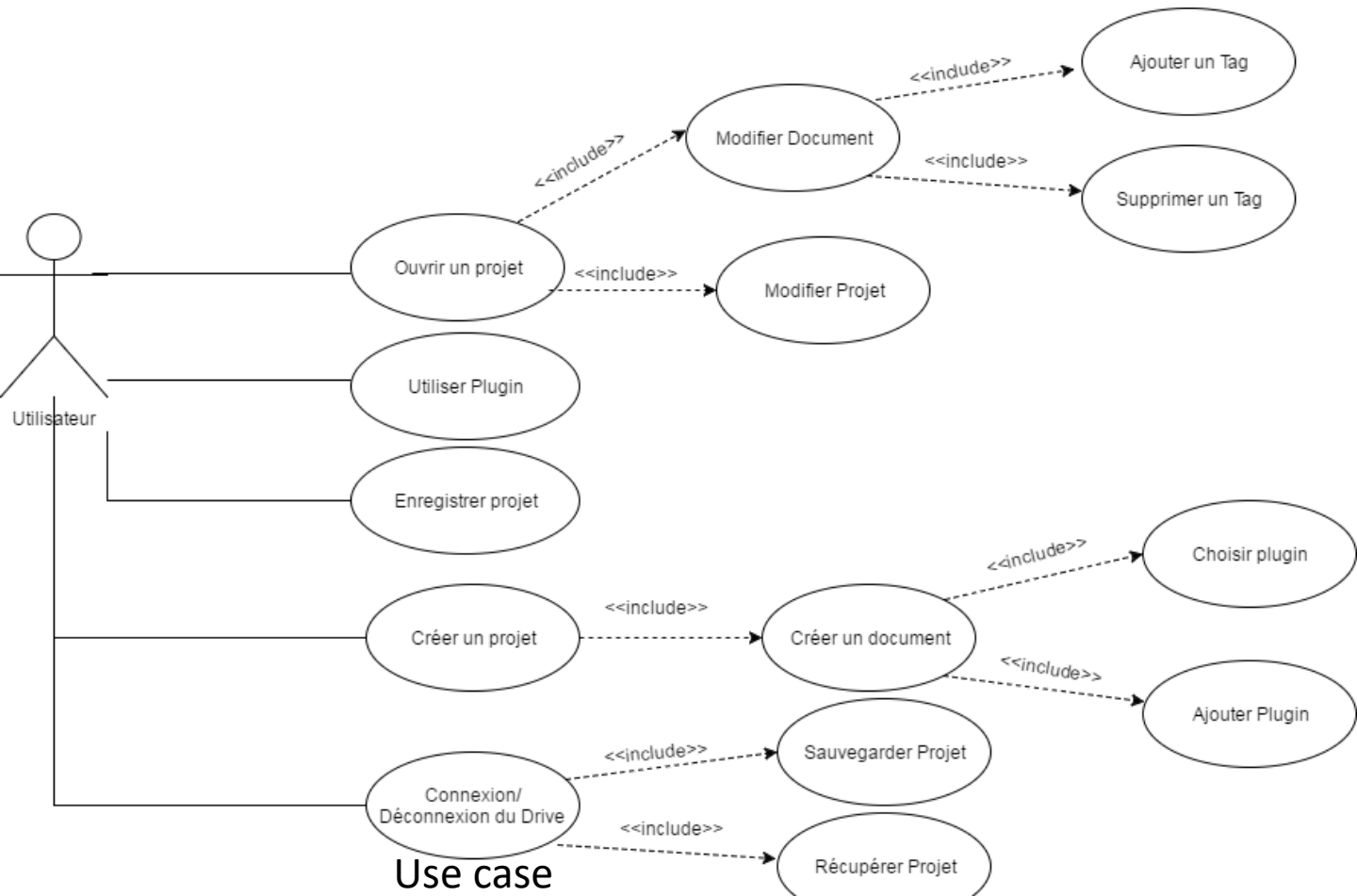


Diagramme état



Point de décision - connexion



Use case

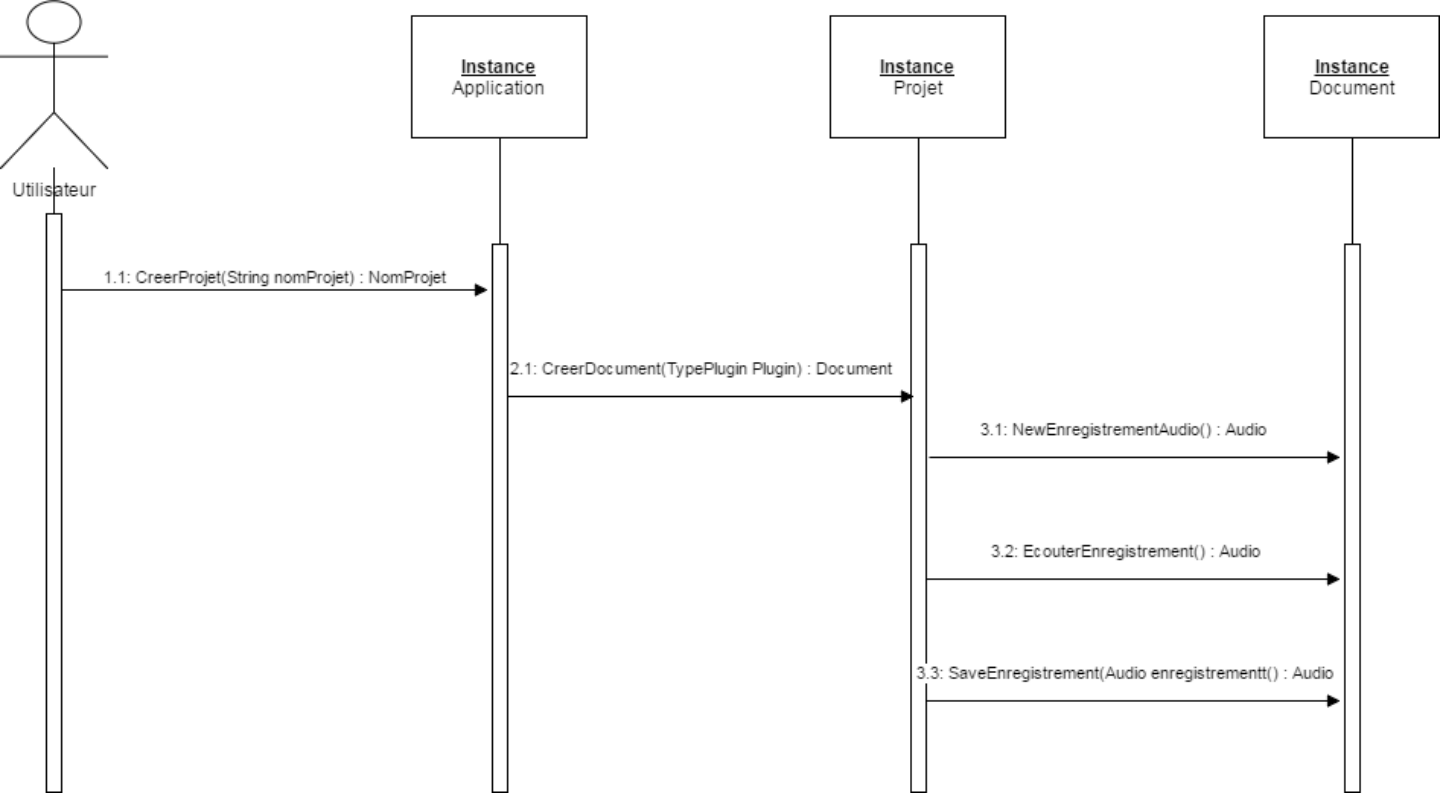


Diagramme séquence audio

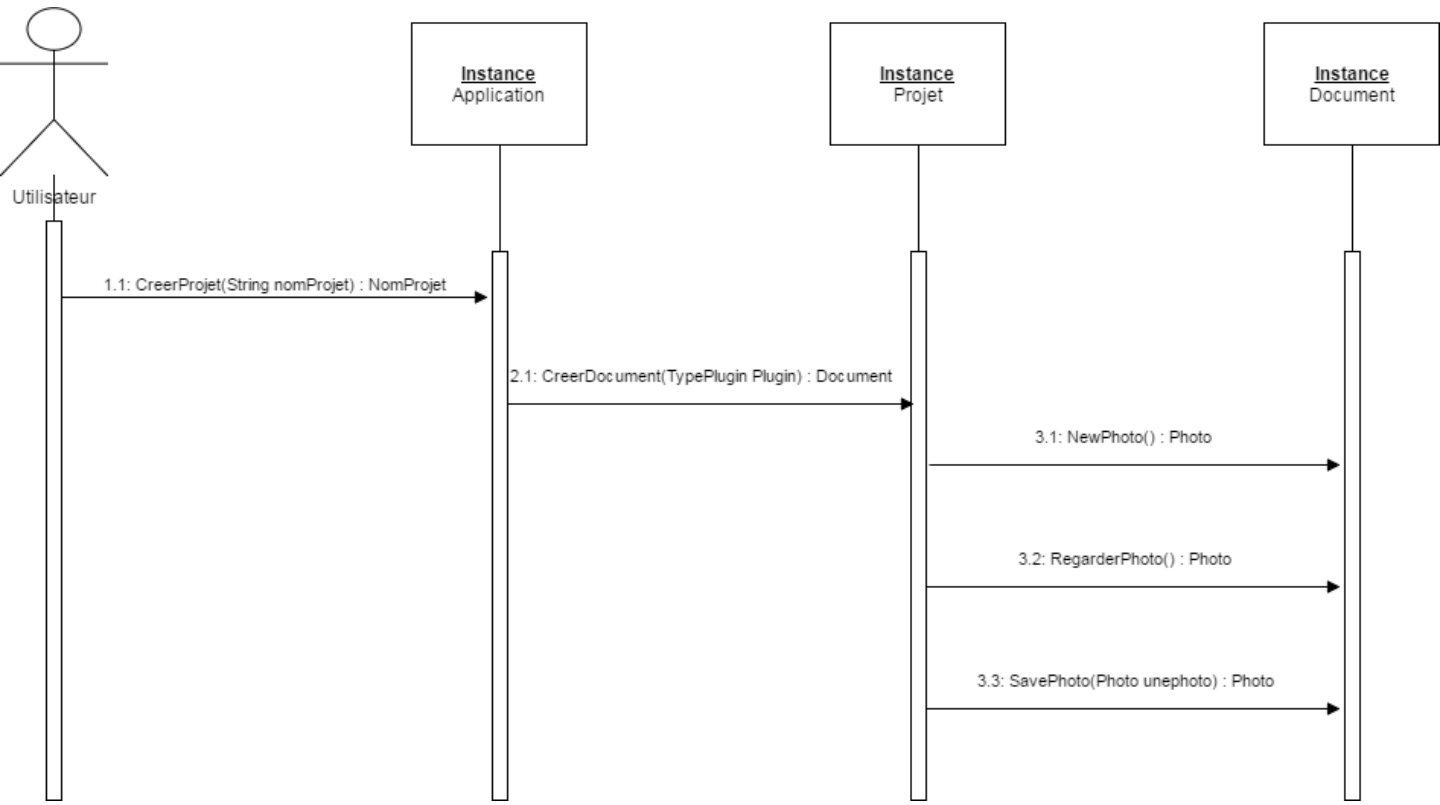


Diagramme séquence photo

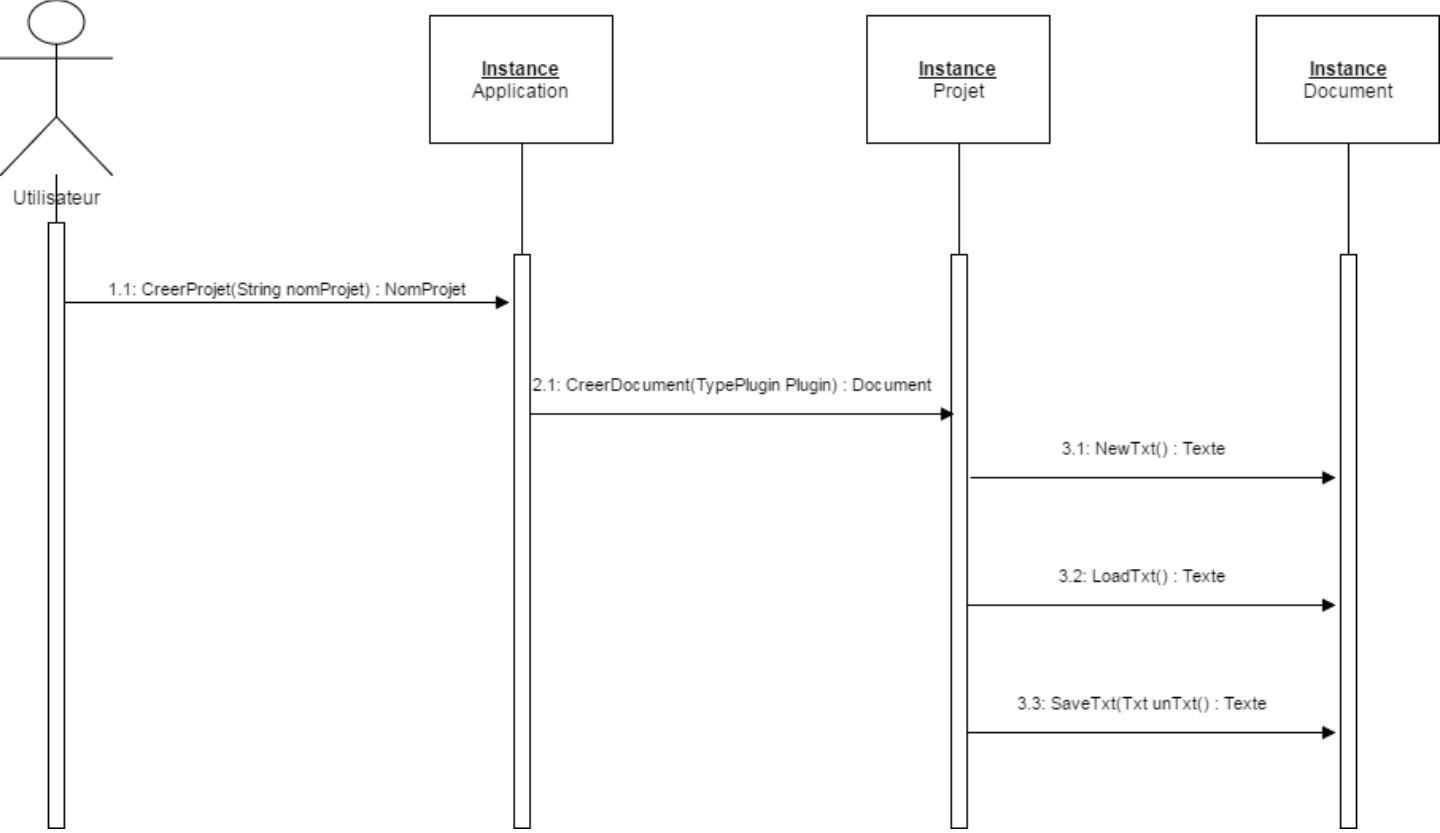


Diagramme séquence Texte

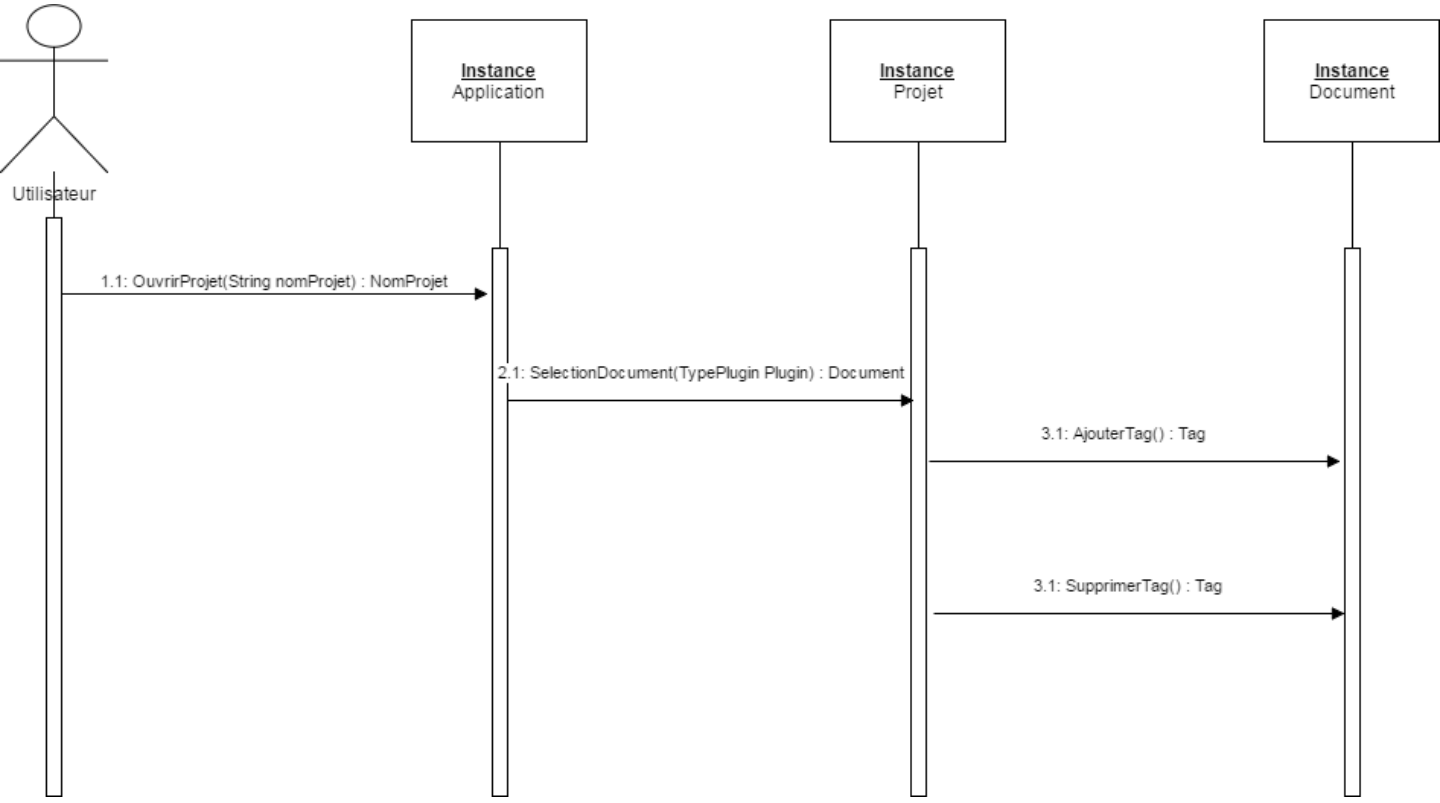


Diagramme séquence Tags

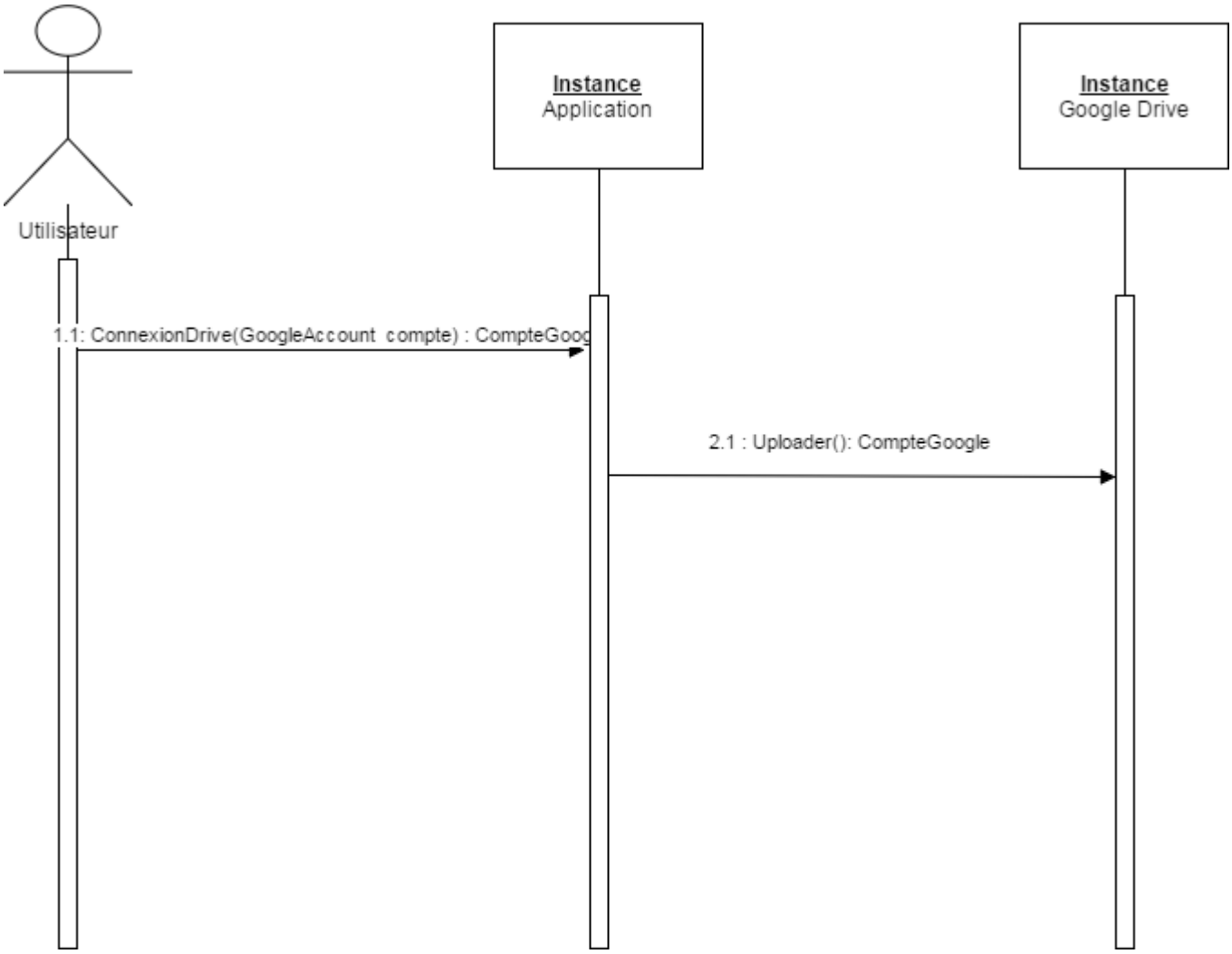


Diagramme séquence Upload