

A dark blue vertical bar runs down the left side of the page. A blue arrow points to the right from this bar, containing the date.

23/02/2023

# Livable 3

EasySave

Several thin, curved lines in dark blue and light grey originate from the bottom left corner and sweep upwards and to the right.

L.Barreto, V.Goulier, A.Martinez, A.Milhas



## Table des matières

I - Introduction :	3
Parties prenantes :	3
Contexte :	3
II - Version actuelle EasySave :	4
Etat de l'art :	4
Evolutions :	5
II – Cahier des charges :	6
Cahier des charges – Version 3.0 :	6
III – Diagrammes :	8
Diagramme d'activité :	8
Diagramme de classe :	9
Interactions :	11
III – Réalisation :	12
Console déportée et communication :	12
Mono-instanciation de l'application :	12
Type de sauvegarde :	12
Conclusion :	14

## I - Introduction :

Parties prenantes :

### **Société ProSoft :**

- L.Barreto
- V.Goulier
- A.Martinez
- A.Milhas

Contexte :

Pour rappel, notre équipe de développement a été embauchée par la société ProSoft le mois dernier. Nous avons été assignés au projet EasySave, un projet de logiciel de sauvegarde.

Précédemment, nous nous sommes assurés de la création de l'architecture de l'application, la réalisation de diagrammes et autres outils de prévisualisation pour notre future application. À la suite de cela, nous avons développé la version 1.0 de l'application qui était une version console. Nous avons rapidement été missionnés pour développer une version 2.0 de l'application qui comprend une interface graphique pour améliorer l'appréhension de l'application par nos clients.

Aujourd'hui, l'application avec interface graphique est développée et est entrée en production. Nous avons reçu de la part de nos supérieurs la demande pour le développement d'une version 3.0 qui se veut être une amélioration de la version 3.0. Nous allons détailler dans ce livrable l'ensemble des modifications qui ont été apportées sur cette version 3.0 de l'application EasySave.

## II - Version actuelle EasySave :

### Etat de l'art :

Actuellement, la version 2.0 de l'application EasySave est en production et est utilisée par de nombreux clients. Pour rappel, voici un résumé des fonctionnalités majeures de cette version 2.0 afin de mieux comprendre la pertinence des évolutions de la version 3.0.

Premièrement, l'application dispose d'une interface graphique pour permettre à l'utilisateur d'effectuer les actions sur ses différents travaux de sauvegarde. Il n'y a aucune limite quand au nombre de travaux de sauvegarde possibles. Pour la sécurité et la facilité du support, les fichiers logs sont disponibles en deux formats différents et comprennent des informations comme l'horodatage, les dénominations, les adresses sources et destinations ou encore le temps de chiffrement. En parlant de cryptage, il est important de signaler que la version 2.0 comporte plusieurs mécanismes de sécurité comme le chiffrement. Celui-ci est effectué par le biais d'un autre logiciel ProSoft, CryptoSoft. Finalement, afin d'éviter tout problème de sécurité avec un logiciel métier du client, nous avons installé une sécurité qui interdit le lancement de nouveau travail de sauvegarde si un logiciel métier est détecté en fonctionnement.

Pour plus de détail et pour une visualisation simplifiée des informations présentées ci-dessous veuillez trouver un tableau extrait du cahier des charges du projet qui résume les informations.

Fonction	Version 2.0
Interface Graphique	WPF
Multi-langues	Anglais et Français
Travaux de sauvegarde	Illimité
Fichier Log journalier	Oui ( JSON , XML) (Information supplémentaire sur le temps de cryptage)
L'utilisateur peut mettre en pause une ou Plusieurs taches	Non
Fichier Etat	Oui
Type de fonctionnement Sauvegarde	Mono ou Séquentielle
Arrêt si détection du logiciel métier	Oui (impossible de lancer un travail)
Utilisation du logiciel de cryptage externe «CryptoSoft »	Oui

## Evolutions :

La version 3.0 est aujourd'hui pressentie pour être la version finale de l'application et ce pour une grande période. En effet, elle intègre toutes les fonctionnalités qui ont été demandées par nos clients lors des différentes entrevues que les chefs de produit ont pu avoir avec eux. Néanmoins, nous souhaitons aller constamment de l'avant et faire encore progresser l'expérience utilisateur de notre application. C'est pour cela que lors de la revue finale de projet, notre équipe vous présentera des pistes d'amélioration pour une éventuelle version 4.0 de l'application EasySave.

## II – Cahier des charges :

### Cahier des charges – Version 3.0 :

Comment expliqué précédemment, nous allons détailler le cahier des charges de la version 3.0 de l'application EasySave et donc détailler tous les changements et améliorations que nous avons pu apporter à cette application.

#### **Sauvegarde en parallèle :**

Nous avons supprimé le mode de sauvegarde séquentiel standardiser un mode de sauvegarde en parallèle.

#### **Gestion des fichiers prioritaires :**

L'utilisateur a désormais la possibilité de définir dans les paramètres de l'application des extensions de fichiers prioritaires. Aucune sauvegarde de fichier considéré comme non prioritaire ne pourra être effectuée tant que des fichiers prioritaires seront en file d'attente.

#### **Limite de taille lors des transferts :**

Nous avons ajouté une distinction concernant la taille de fichiers. En effet, nous considérons comme « gros fichiers » des fichiers dont la taille est supérieure à n Ko.

Pendant l'exécution, dans un souci de saturation de la bande passante, nous ne transférons pas en même temps plusieurs « gros fichiers ». En parallèle, il reste possible de continuer le transfert de plus petits fichiers tout en respectant les règles de priorité.

#### **Interaction avec les travaux :**

Pour chaque travail ou ensemble de travaux, l'utilisateur a la possibilité de :

- Pause : mettre en pause après le transfert du fichier en cours
- Play : démarrer ou reprendre un travail
- Stop : arrêter immédiatement le travail et la tâche en cours

L'utilisateur peut voir en temps réel l'avancement de ses travaux.

#### **Pause en cas de détection de logiciel métier :**

Si un logiciel métier est détecté sous statut « actif », les opérations de l'application sont mises en pause.

#### **Console déportée :**

Nous avons ajouté une console déportée pour agir sur les travaux de sauvegarde et suivre leur avancement en temps réel depuis un poste distant.

Cette console déportée comporte une interface homme-machine qui permettra une prise en main facile pour l'utilisateur.

## **Application mono-instance :**

Afin d'éviter que l'utilisateur lance plusieurs fois l'application, une sécurité est présente.

Pour conclure sur les besoins de l'application, vous retrouvez ci-dessous un tableau récapitulatif des besoins pour l'ensemble des versions EasySave.

Fonction	Version 1.0	Version 1.1	Version 2.0	Version 3.0
Interface Graphique	Console	Console	WPF	WPF
Multi-langues	Anglais et Français	Anglais et Français	Anglais et Français	Anglais et Français
Travaux de sauvegarde	Limité à 5	Limité à 5	Illimité	
Fichier Log journalier	Oui en JSON uniquement	Oui ( JSON , XML)	Oui ( JSON , XML) (Information supplémentaire sur le temps de cryptage)	Oui ( JSON , XML)
L'utilisateur peut mettre en pause une ou Plusieurs taches	Non	Non	Non	Oui
Fichier Etat	Oui	Oui	Oui	Oui
Type de fonctionnement Sauvegarde	Mono ou séquentielle	Mono ou séquentielle	Mono ou Séquentielle	Parallèle
Arrêt si détection du logiciel métier	Non	Non	Oui (impossible de lancer un travail)	Oui (arrêt de tous les transferts en cours)
Utilisation du logiciel de cryptage externe «CryptoSoft»	Non	Non	Oui	Oui
Gestion de fichiers Prioritaires	Non	Non	Non	OUI, avec attente des autres taches
Interdiction de sauvegardes Simultanées pour les fichiers volumineux	Non	Non	Non	OUI
Interface de visualisation déportée	Non	Non	Non	Oui
Application Mono-instance	Non	Non	Non	Oui
Surveillance charge Réseau	Non	Non	Non	Réduction automatique des flux

Dans les parties suivantes, nous allons détailler avec des diagrammes ainsi que des explications le fonctionnement des nouveautés implémentées dans la version 3.0 de l'application EasySave.



### III – Diagrammes :

#### Diagramme d'activité :

Le diagramme d'activité présenté ci-dessous montre que l'ensemble des actions utilisateur passent dans un premier temps par la vue. C'est donc elle qui va gérer les actions de l'utilisateur. Elle va donc appeler les méthodes du modèle pour effectuer les actions demandées par l'utilisateur.

Nous retrouvons ensuite les 3 grandes parties du programme :

- La partie de création des Jobs avec l'ensemble des opérations associées.
- La partie paramètres avec l'ensemble des opérations associées.
- La partie de suppression, modification, démarrage et arrêt des Jobs.

Ce diagramme d'activité vous permet donc de comprendre le fonctionnement du programme.

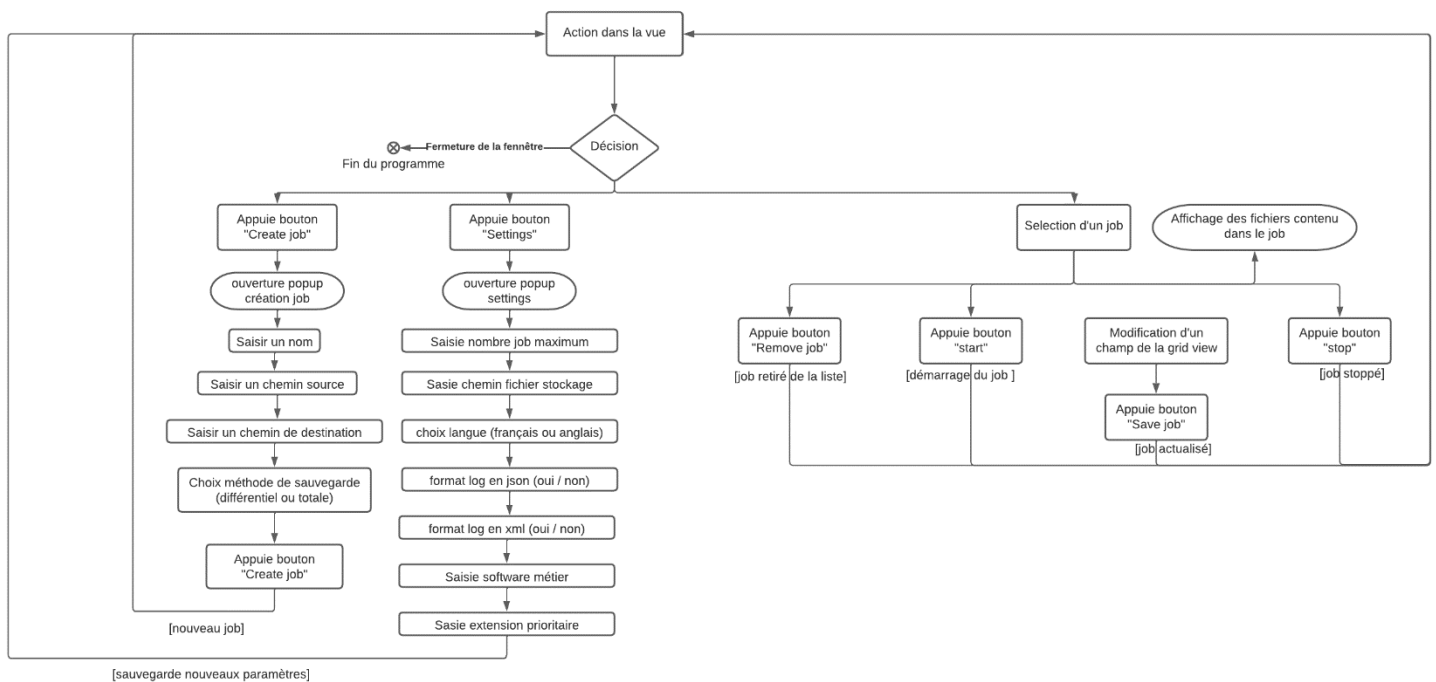
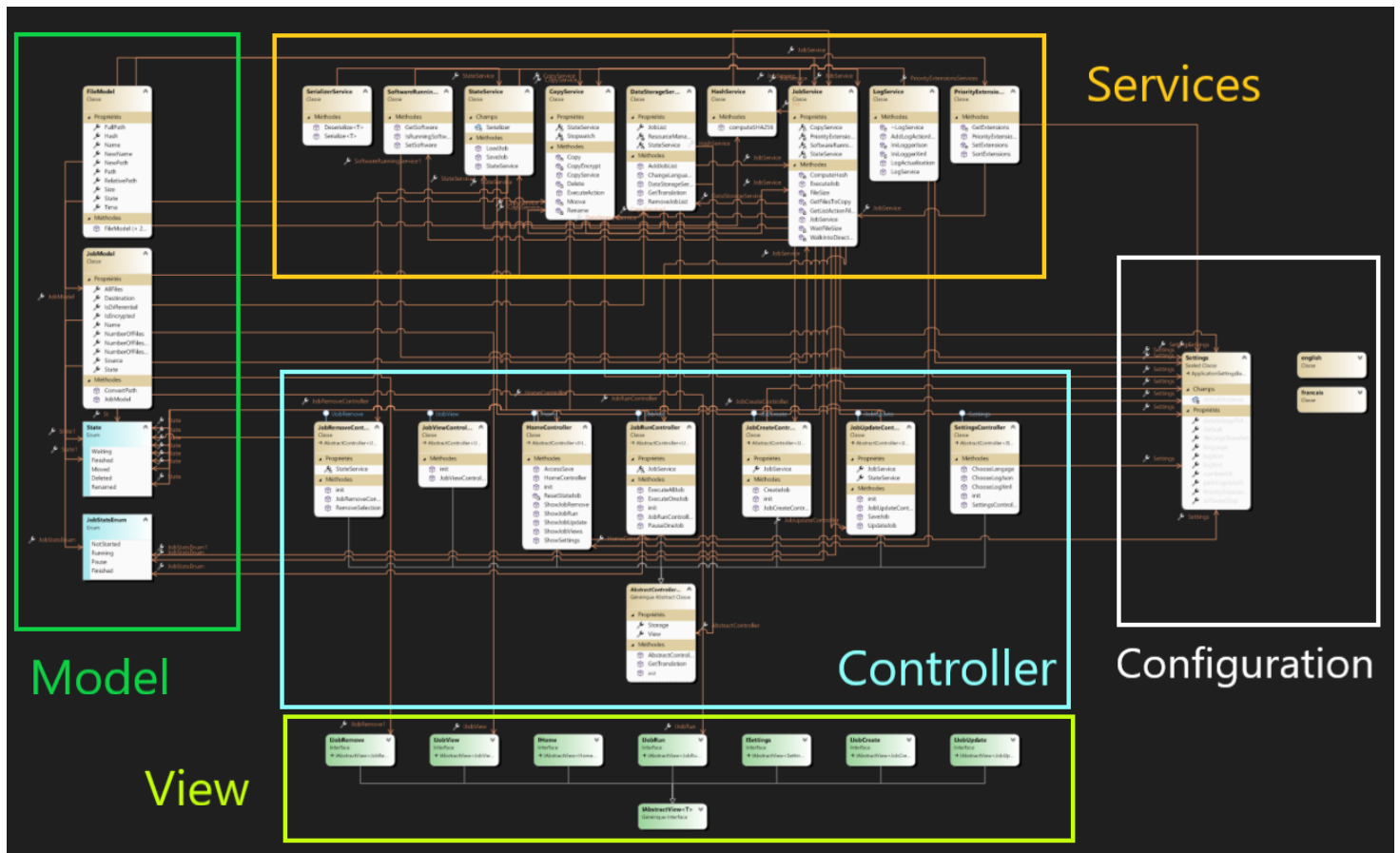


Diagramme de classe :

Ci-dessous se présente le diagramme de classes du projet EasySave. Ce diagramme nous permet de mettre en lumière plusieurs choses quant à l'architecture et au fonctionnement de notre application.



Tout d'abord, avec les segmentations qui ont été faites, on repère très facilement les 3 sections Model, View ainsi que Controller. Cela met en lumière l'utilisation du modèle MVC pour la structure et l'architecture de notre application.

On retrouve donc l'ensemble des vues de notre application, les modèles qui définissent les données nécessaires au bon fonctionnement de l'application et finalement l'ensemble des Controller qui nous permettent d'interfacer les vues avec les modèles.

En plus de ces 3 parties principales de l'application, nous retrouvons les fichiers de configuration. Cette partie contient par exemple les fichiers nécessaires aux différentes langues de l'application ainsi que les autres paramètres de l'application. Finalement, on retrouve toute la partie Service qui stocke donc l'ensemble de nos logiciels d'automatisation.

Au vu du nombre de liens présents dans ce diagramme, il devient complexe de pouvoir clairement suivre chacun des liens mais il est tout de même possible de ressortir une tendance principale. En effet, il est aisé de remarquer que beaucoup de liens rejoignent les Controller ce qui est logique en raison de leur fonction d'interfaçage des autres parties de l'architecture. De plus, on remarque aussi la forte utilisation des services pour les différentes fonctionnalités.

Si vous souhaitez pouvoir facilement étudier les différentes classes ainsi que leur contenu, veuillez noter que le diagramme ci-dessus est disponible en annexe du projet.

## Interactions :

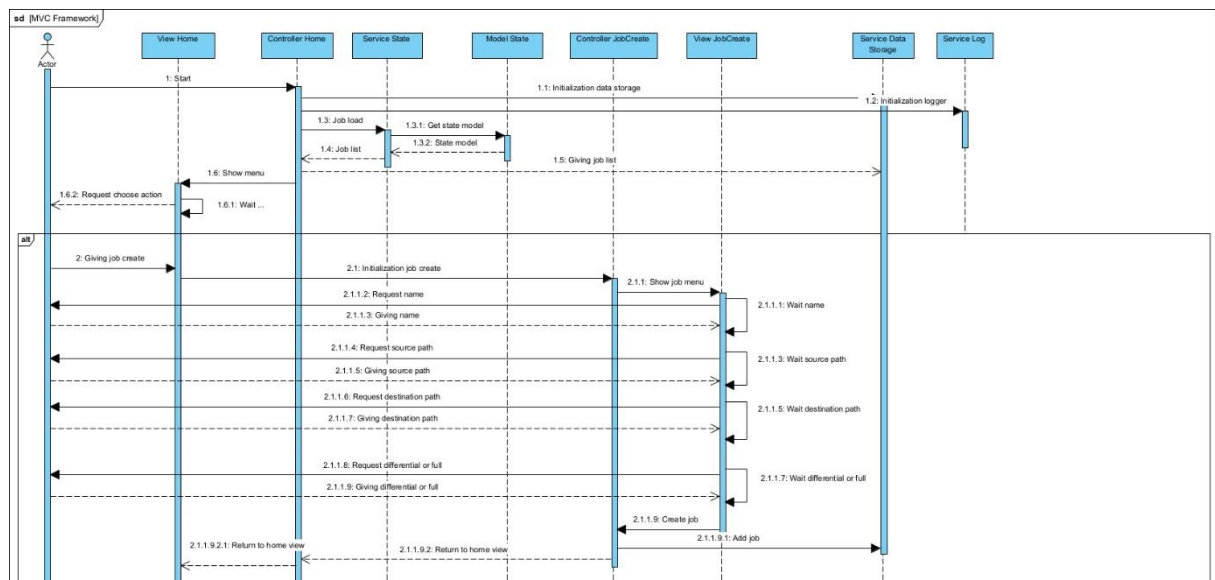
Nous avons donc pu voir grâce au diagramme d'activité l'ensemble des opérations possibles pour l'utilisateur dans l'application EasySave puis avec le diagramme de classes l'ensemble des classes qui compose l'application et donc par extension l'architecture de l'application.

Il est possible en prenant pour base ses deux diagrammes de comprendre l'ensemble des interactions qu'ont les parties de l'application entre elles. En effet, l'architecture MVC que nous avons choisi pour notre projet entraîne un fonctionnement bien spécifique qui mérité d'être détaillé.

Le point d'entrée de notre application est l'interface graphique qui est contenue dans la partie Views du programme comme détaillé dans le diagramme de classes. Depuis ce point d'entrée nous pouvons accéder aux différentes fonctionnalités de l'application. Ces fonctionnalités sont gérées par les Controller. L'ensemble des données que nous manipulons sont stockées dans les Modèles de l'application. Maintenant que nous savons où se trouvent l'ensemble des informations, voici le fonctionnement interne :

Lorsque la vue va être sollicitée par l'utilisateur, le Controller va effectuer l'action en lien avec la demande de l'utilisateur. Il va réceptionner la demande puis aller demander au modèle les données nécessaires pour finalement renvoyer ses données à la vue pour retourner un affichage cohérent. C'est ainsi que notre application est séquencée.

Finalement et à titre d'exemple, voici un extrait de diagramme de séquence pour illustrer les interactions View – Controller – Model exposées ci-dessus.



### III – Réalisation :

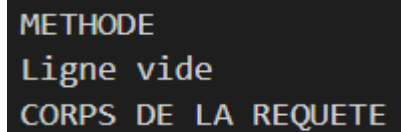
Ici, nous détaillerons nos différents choix techniques et explications à propos de la version 3.0 du logiciel EasySave.

#### Console déportée et communication :

Tout d'abord, un des changements majeurs sur EasySave 3.0 est l'arrivée d'une console déportée qui permet, depuis un poste distant, de visualiser et contrôler les travaux de sauvegarde. Afin de permettre la communication des informations entre l'application locale et l'application distante, nous avons développé un protocole de communication dédié.

Pour un examen en profondeur du protocole de communication, je vous invite à consulter le document « Protocole de communication EasySave.pdf » dans les annexes du projet. Néanmoins, les informations importantes à propos de ce protocole sont les suivantes :

-Premièrement, la forme de la requête a été définie comme présentée sur l'image ci-contre :



```
METHODE
Ligne vide
CORPS DE LA REQUETE
```

-Deuxièmement, nous disposons de 3 méthodes, GET qui permet de récupérer les informations de l'application locale, PAUSE pour mettre en pause un travail de sauvegarde et finalement PLAY pour démarrer ou reprendre un travail de sauvegarde.

-Ensuite, le format utilisé sera le format JSON.

-Nous communiquons en utilisant le protocole TCP.

Pour terminer cette présentation de la console déportée, veuillez noter que celle-ci possède sa propre documentation utilisateur disponible dans les annexes du projet.

#### Mono-instanciation de l'application :

Afin d'éviter tout problème de conflit entre les différents travaux de sauvegarde, nous avons implémenté une sécurité qui rend donc impossible la multi instance. Nous avons donc mis en place une vérification à base de mutex pour vérifier si l'application est déjà lancée ou non. Dans les faits, à chaque démarrage, l'application va générer un mutex. Dans le cas où un mutex est détecté au lancement de l'application, cela veut dire que celle-ci en a déjà généré un et qu'elle est donc déjà en fonctionnement. C'est donc cette vérification de la présence de mutex qui nous permet de sécuriser l'instanciation de notre application.

#### Type de sauvegarde :

A la demande de nombreux clients, nous avons supprimé les types de sauvegarde différentiels et unitaires au profit d'un seul type de sauvegarde en parallèle.

Avec cette implantation de sauvegarde en parallèle s'ajoutent d'autres fonctionnalités à savoir la gestion des tailles de fichiers. En effet, nous avons mis en place un système de classification des fichiers. En effet, nous voulons distinguer deux types de fichiers à savoir les « gros » et « petits » fichiers. Nous ségrégons les fichiers en contrôlant leur taille. Nous avons mis en place une valeur par défaut de 10 Mo pour faire la séparation. En effet, tout fichier ayant une taille supérieure à 10 Mo sera considéré comme gros fichier alors que tous les fichiers ayant une taille inférieure à 10 Mo sera considéré comme un petit fichier.

Avec ces tailles de fichiers pris en compte une règle est en place. Cette règle permet d'interdire la copie en simultané de plusieurs gros fichiers. Cela intervient dans un souci d'éviter la saturation des performances de l'ordinateur de l'utilisateur. Lorsque que plusieurs gros fichiers sont en attente de traitement, l'application va automatiquement autoriser le traitement de plus petits fichiers afin de ne pas rester bloqué sur la file d'attente inutilement. Cette fonctionnalité nous permet donc d'optimiser les flux dans notre application.

## Conclusion :

Nous avons donc pu exposer dans ce document le contexte dans lequel nous évoluons aujourd'hui avec un rappel des fonctionnalités implémentées dans la version 2.0 de l'application. Nous avons ensuite passé en revue les fonctionnalités ajoutées avec la version 3.0. Afin de clarifier le fonctionnement de l'application ainsi que certains points techniques nous sommes revenus sur les points fondamentaux de celle-ci.

Veuillez prendre note du fait que nombre des documents présentés dans ce rapport sont disponibles en annexe aux côtés de la Release Note de l'application ainsi que de sa documentation utilisateur multilingue.