INSA de Rennes
Quatrième année Informatique

Rapport de Bilan de planification

Projet VaR

Benjamin Bouguet - Paul Chaignon
Eric Chauty - Ulysse Goarant

Hamdi Raissi - Ivan Le Plumey
Quentin Giai Gianetto

Mai 2014

Table des matières

Chapitre 1

Introduction

Dans le milieu de la finance, les prises de décisions liées aux investissements sont en grande partie motivées par une estimation des risques associés. De plus, l'établissement des accords de Bâle dont le troisième et dernier a été publié fin 2010, obligent les banques à s'organiser dans ce sens. En effet, il s'agit d'accords de réglementations bancaires qui visent à garantir un niveau minimum de capitaux afin d'assurer la solidité financière des banques. Afin d'être en règle vis-à-vis de ces accords, des outils de gestion de risque, tels que la Value-at-Risk (VaR) peuvent être utilisés. La VaR correspond à la perte associée à un actif qu'il est probable de ne pas excéder pour un niveau de confiance et un horizon de temps donnés. Par exemple, une banque peut déterminer une VaR de 100,000 € pour un niveau de confiance de 95% (équivalent à un risque de 5%) et un horizon de temps d'un mois. Cela signifie qu'il y a une probabilité de 5% de perdre plus de 100,000 € d'ici un mois.

Notre logiciel sera justement un outil d'aide à la décision qui permettra de calculer la VaR par diverses méthodes statistiques. Il proposera également un ensemble de tests, comparaisons, tableaux récapitulatifs et backtesting [1] entre les différentes méthodes statistiques. Avec cette multitude d'outils, les financiers seront en mesure de travailler sur leurs portefeuilles [2] aisément, au moyen d'une interface simple, intuitive et ergonomique, pensée pour l'utilisation au quotidien.

En décembre 2013, nous avions établi une estimation du temps de travail à réaliser sur chaque partie du logiciel, que ce soit du développement ou des tests, chaque point avait été détaillé. Durant les quelques mois qui ont suivi, un décompte du temps passé sur chaque étape du développement ou des tests a été effectué.

Aujourd'hui nous sommes en mesure d'effectuer un bilan de notre planification initiale. Nous reprendrons les fonctionnalités précedemment détaillées dans le rapport de planification, en passant en revue sur les éventuelles différences observées entre la durée estimée et la durée réelle.

Nous avions également effectuer des prévisions de temps de travail sur la conception, la page HTML, la réalisation du rapport final et la préparation à la soutenance. Nous n'en parlerons pas dans ce document car ces dernières sont

1. Le backtesting ou test rétroactif de validité consiste à tester la pertinence d'une modélisation ou d'une stratégie en s'appuyant sur un large ensemble de données historiques réelles.

2. Un portefeuille est une composition de plusieurs actifs financiers (actions, obligations, matières premières).

secondaires dans les enseignements à retenir des erreurs de planification. En effet, ces étapes se sont toutes bien déroulées ou ne sont pas encore terminées.

Chapitre 2

Développement

Le développement est composé de sept grandes parties :
— les données
— les calculs
— le backtesting
— l'IHM
— les graphiques
— les rapports
— les sessions

Nous dresserons dans cet ordre le bilan des différentes phases du développement. Comme prévu, nous avons eu l'avantage que beaucoup de nos tâches ont été indépendantes les unes des autres. Ceci nous a permis d'avoir une plus grande flexibilité dans la répartition des tâches et leur réalisation. Cependant, nous avons eu un surplus de travail (conflits) lors de la fusion des différentes branches de développement.

2.1 Gestion des données

Le calcul de la Value-At-Risk s'effectuant sur des données financières, il est nécessaire de commencer par implémenter la gestion des données.

2.1.1 Actifs

Les actifs constituent les éléments de base des données manipulées dans le logiciel. La première étape a été l'implémentation du modèle obtenu lors de la phase de conception. C'est à dire de créer une classe et les méthodes associées.

Notre logiciel est capable d'importer des données provenant de Yahoo Finance aux formats CSV et TXT. L'importation est consitutée d'un ensemble de traitements sur le fichier des actifs et résulte en la création d'un nouveau fichier ne contenant que les données sélectionées.

Tâches	Durée prévue	Durée réelle
Modélisation des actifs	10h	11h
Importation des actifs	20h	21h
Exportation des actifs	10h	4h

TABLE 2.1 – Durées des tâches liées aux actifs

Nous avons passé presque exactement le temps prévu sur l'importation des données. Cependant nous aurions pu être plus rapide. Cela s'expliquant par le fait qu'elle a été réalisée en trois itérations, chacune ayant demandé des vérifications et des relectures par les membres du groupe. Cela a été nécessaire car nous avons dû modifier le code pour le besoin d'autres branches de développement.

L'exportation a été beaucoup plus rapide que prévu, car tout le travail a été réalisé lors de l'étape d'importation.

2.1.2 Portefeuilles

Les portefeuilles sont constitués d'une composition d'actifs. Il était donc nécessaire d'avoir préalablement implémenté la gestion des actifs avant d'envisager l'implémentation de la gestion des portefeuilles.

Tâches	Durée prévue	Durée réelle
Modélisation des portefeuilles	20h	19h30
Importer un portefeuille	15h	?
Exporter un portefeuille	10h	?

TABLE 2.2 – Durées des tâches liées aux portefeuilles

Aucun souci particulier n'ayant été rencontré durant la réalisation des différentes classes nécessaires à la gestion des portfeuilles, nous avons presque égalé le temps prévu, notre estimation était donc bonne.

L'importation et l'exportation ont quant à elles été plus courtes que prévu à réaliser. Ces deux tâches s'incluent dans une étape d'import et d'export de toutes les données. Cet écart assez important se justifie par le fait qu'un temps important a déjà été passé sur la structuration des données, donc il ne "restait" plus qu'à collecter l'ensemble des données de tous les portefeuilles (fichiers d'actifs, rapports et la base de données en JSON) pour les ajouter dans une archive en zip.

2.2 Calculs

Les fonctionnalités de calcul contiennent principalement le calcul de la Value-at-Risk selon les différentes méthodes. Pour calculer cette dernière, nous avons utilisé le logiciel de statistiques R. Dans un premier temps, nous avons cherché à interfacer notre logiciel avec R en utilisant RInside, conformément aux spécifications. Cependant, cela s'est révélé être impossible malgré toutes nos recherches et essais. Nous avons dû trouver une autre solution.

Les tests de corrélation nous ont été fournis sous forme de scripts R par notre encadrant. Il nous a d'abord fallu les tester (ce qui correspond à la tâche de test des scripts externes) puis les appeler via R. Comme cet appel est différent d'une simple utilisation de R, nous avons prévu une durée un peu plus longue.

Enfin, les calculs statistiques se résument à de simples appels à des fonctions fournies par R. C'est pourquoi cette tâche est relativement courte.

Tâches	Durée prévue	Durée réelle
Interfaçage avec R	20h	14h30
Modélisation GARCH	15h	11h30
Méthode historique	5h	9h
Méthode Riskmetrics	10h	2h
Tests de corrélation	10h	?
Tests scripts externes R	5h	4h
Calculs statistiques	5h	?

TABLE 2.3 – Durées des tâches liées aux calculs

Malgré le fait que nous ayons rencontré de nombreuses difficultés et que l'on ait dû changer notre méthode d'interfacage de notre logiciel avec R, nous avons terminé cette tâche en moins de temps que prévu. La deuxième solution qui est d'utiliser RScript s'est révelée beaucoup plus simple que notre première.

Bien que le temps passé sur la méthode historique excède celui prévu initialement, cela se doit au fait qu'un code en commun à d'autres classes a été développé. De plus, des dépendances avec d'autres branches ont nécessité plusieurs modifications. En elle même, la méthode historique a été rapide à développer.

Comme nous le pensions, la réalisation de la méthode Riskmetrics devait être plus rapide, car grandement inspirée de la modélisation GARCH. Mais elle s'est révelée encore plus rapide, l'estimation a été vraiment sur-évaluée. Il possible qu'un peu plus de travail en amont sur les notions mathématiques à implémenter ait permis une meilleure prévision.

Même remarque que précédemment, la méthode GARCH s'est révelée plus rapide à réaliser, une connaissance moins floue sur Qt et les notions mathématiques utilisées, aurait pu donner une meilleure évaluation.

2.3 Backtesting

Le développement du backtesting a nécessité préalablement le fonctionnement du calcul de la VaR selon les différentes méthodes de calcul.

Tâches	Durée prévue	Durée réelle
Backtesting	30h	3h

TABLE 2.4 – Durées des tâches liées au backtesting

Cette étape a été complètement sur-évaluée. Ceci pourrait peut être s'expliquer de nouveau par le manque de certaines notions mathématiques, mais aussi par la bonne réalisation des calculs de VaR précédemment. Le backtesting étant totalement basé dessus, et ne fait que lancer des tests de comparaisons entre les méthodes de calcul de VaR.

2.4 Rapports

De manière générale, les rapports résument les informations générées par les autres modules du logiciel. Il n'était cependant pas nécessaire que le module correspondant au rapport à générer soit d'abord implémenté. Le développement en parallèle du module de la génération des rapports et des modules correspondants d'où sont extraites les données, a donc été possible.

Tâches	Durée prévue	Durée réelle
Rapport au format *DOCX*	10h	23h
Rapport au format *PDF*	10h	16h
Rapport du calcul de la VaR	5h	2h
Rapport de statistiques générales	5h	2h
Rapport de matrice de covariance	5h	2h
Rapport de backtesting	5h	2h

TABLE 2.5 – Durées des tâches liées aux rapports

Dans le rapport de spécification, nous avions prévu d'utiliser la librairie *libopc.* pour générer les rapports en PDF ou DOCX. Cela s'est révelé être une mauvaise idée, cela était bien trop complexe. Nous avons opté pour l'utilisation et la réalisation d'un programme Java, auquel notre logiciel envoie un ensemble de paramètres représentant les données à ajouter dans le rapport. Cela explique donc pourquoi nous avons dépassé le temps prévu pour la réalisation des rapports en DOCX, car ce fût les premiers réalisés.

Le format PDF a lui été plus rapide, car nous avons réutilisé les modèles prédéfinis des rapports DOCX.

La construction des autres rapports n'a pas été compliquée. En effet, cela à juste consité à créer de nouveaux modèles et à leur passer en paramètres les résultats des différents calculs. Notre prévision s'étant établie avec une méthode plus compliquée, il est donc naturel d'observer un temps de réalisation plus rapide.

2.5 Graphiques

Les graphiques nécessitent principalement l'implémentation de la gestion des données. L'implémentation des matrices de corrélation nécessite de plus celle des

calculs correspondants.

Tâches	Durée prévue	Durée réelle
Matrice de corrélation	15h	6h
Histogramme	15h	?

TABLE 2.6 – Durées des tâches liées aux graphiques

2.6 IHM

L'IHM contient toutes les actions que l'utilisateur peut effectuer, les formulaires que l'utilisateur devra remplir ainsi que les éléments graphiques tels que les tableaux. La durée des tâches comprend aussi la gestion des messages d'erreurs ainsi que leur affichage dans l'IHM. Il sera possible de démarrer le développement de l'IHM en parallèle de celui des autres modules.

Tâches	Durée prévue	Durée réelle
Exporter des actifs	5h	?
Importer des actifs	10h	4h
Backtesting	5h	?
Calcul de la VAR	5h	?
Ajouter/Modifier portefeuille	10h	3h
Gestion des sessions	10h	?
Générer rapports	5h	5h
Calculs statistiques	5h	?
Exporter un portefeuille	10h	?
Importer un portefeuille	5h	?
Supprimer un portefeuille	5h	?
Listing des portefeuilles	10h	1h30
Afficher le contenu d'un portefeuille	10h	14h
Afficher les rapports d'un portefeuille	5h	5h

TABLE 2.7 – Durées des tâches liées à l'IHM

Pour la plupart des éléments graphiques, de simples intéractions entre les objets Qt ont été suffisantes. C'est pour cela que le temps prévu s'est révélé trop important.

L'affichage du contenu des portefeuilles été plus compliqué. Nous aurions pu nous limiter à une simple lecture à la base de données puis à un affichage dans un tableau, mais ceci n'était pas très adapté. Nous sommes partis sur la réalisation d'un QAbstractItemModel, car c'est plus propre comme cela. Par

exemple, nous n'avons pas à gérer explicitement le bon nombre de ligne ou de colonnes à afficher.

2.7 Session

Cette fonctionnalité ajoute la sauvegarde dans la base de données des portefeuilles, rapports et actifs. Les modifications sont aussi sauvegardées (changement de nom, suppression d'un rapport, changement de l'emplacement du fichier d'un rapport). Cette tâche a nécessité que la gestion des actifs et des portefeuilles soit en partie déjà réalisée.

Tâches	Durée prévue	Durée réelle
Sauvegarde de la session	10h	10h
Restaurer la session	10h	6h30

TABLE 2.8 – Durées des tâches liées à la session

Chapitre 3

Tests

Nous avons effectué des tests au fûr et à mesure de chaque branche de développement, et entièrement à la fin du projet. De plus, les tests unitaires de la gestion des données (et des algorithmes de calcul) ont été écrits dès qu'une fonctionnalité (respectivement un algorithme) a été écrite. L'IHM étant plus difficile à tester automatiquement, celle-ci a été testée (manuellement) suivant le même déroulement que les autres points de développement.

Tâches	Durée prévue	Durée réelle
Tests de la gestion des données	20h	14h
Tests des calculs (modèle et VaR)	20h	5h
Tests IHM	20h	4h

TABLE 3.1 – Durées des tâches liées aux tests

Chapitre 4

Diagramme de GANTT

Chapitre 5

Conclusion

Le tableau ci-dessous permet de comparer la planification initiale du projet global avec le temps réellement nécessaire à sa réalisation :

Tâches	Durée prévue	Durée réelle
Développement	375h	?
Tests	60h	?
Total	435h	?

Table 5.1 – Durées des tâches principales

Dans ce document, nous avons comparé les temps prévus à la réalisation des différentes parties de notre logiciel, en expliquant dans la mesure du possible les raisons pour lesquelles nous avons observé des différences entre la durée prévue et la durée effective. Globalement, le temps passé sur la réalisation de ce projet est inférieur au temps prévu. Cela ne signifie pas que moins de choses ont été réalisées, mais plutôt qu'elles avaient été sur-évaluées en temps de développement ou que nous avons été plus efficace.