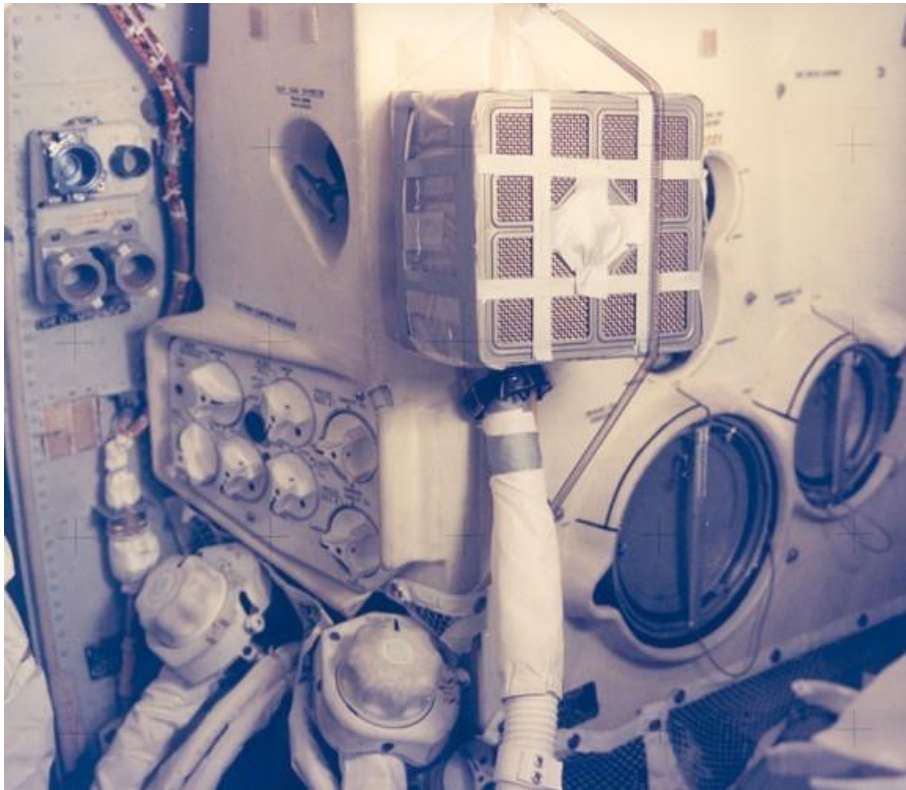


Exécution de Mandats

Table des matières

1	Introduction	3
2	Analyse du mandat par Use Cases et Scénarii.....	4
2.1	Use Cases (cas d'utilisation)	4
2.2	Scénarii	4
2.2.1	Identification	5
2.2.2	Détails	5
3	Planification	7
3.1	Préparation du fichier de planning	7
3.2	Planification initiale.....	10
3.3	Mise à jour	10
4	Détails de réalisation	12
5	Tests	13
5.1	Niveau de test	13
5.1.1	Le Test unitaire	13
5.1.2	Le Test d'intégration.....	13
5.1.3	Le Test Système	14
5.2	Type de test	14
5.3	Stratégie de test	16
5.4	Cas-tests	17
5.5	Plan de test	17
5.6	Résultats de test	17
6	Journal de Travail et Journal de Bord	18
6.1	Le Journal de Travail.....	18
6.2	Le Journal de Bord.....	18
7	Documentation de Projet	19
8	Communiquer sur l'avancement des travaux	20
9	Livraison	22
10	Présenter le résultat final.....	23

1 Introduction



Cela vous dit quelque chose ? Non ? Un petit peu d'histoire alors...

Pour le programme Apollo, la NASA avait mis sur pied deux équipes : une pour concevoir le Module Lunaire et l'autre pour le Module de Commande. Lorsque les choses tournèrent mal pour la mission Apollo 13, les trois astronautes se retrouvèrent face à un sérieux problème : ils devaient réparer le module de commande endommagé avec des pièces du module lunaire. Chacun des deux modules disposait d'un système de filtrage/recyclage de l'air, mais pas de chance : les filtres conçus par une équipe avaient une entrée carrée et les bonbonnes choisies par l'autre avaient une sortie ronde ! Un « détail » qui a bien failli coûter la vie des trois membres d'équipage.

Une des leçons à tirer de cette aventure est qu'un minimum de communication et de convention est nécessaire au bon déroulement d'un projet.

Le but de ce document est de décrire les pratiques appliquées au sein de la filière informatique du CPNV, juste au cas où on nous demanderait de construire une fusée.

2 Analyse du mandat par Use Cases et Scénarii

« Rien ne sert de commencer à construire un escalier là où une échelle suffit. »

Avant de se mettre au travail, il est capital de s'assurer que ce que nous (mandataires) prévoyons de faire correspond aux attentes du mandant.

Pour valider notre compréhension du mandat, nous utilisons la méthode des Use Cases / Scénarii qui s'applique en deux temps.

2.1 Use Cases (cas d'utilisation)

Dans cette phase, il s'agit de bien déterminer à quoi va servir le système à réaliser.

Pour ce faire, nous complétons la phrase :

« On utilise [le système] pour ... ».

Exemple : si on nous demande de réaliser un smartphone simplifié, on peut dire :

1. On utilise un SmartPhone pour téléphoner
2. On utilise un SmartPhone pour envoyer et recevoir des SMS
3. On utilise un SmartPhone pour prendre des photos

Nous avons ainsi identifié trois use cases : « Téléphoner », « Envoyer et recevoir des SMS » et « Prendre des photos ». Au passage, nous avons également établi le fait que notre smartphone simplifié ne permettra pas de surfer sur Internet, puisqu'on n'a pas listé ce cas d'utilisation !

La liste des use cases doit être validée avec le mandant du projet !

2.2 Scénarii

Ensuite, nous détaillons chaque use case au moyen de plusieurs scénarii.

A ce stade, on doit s'imaginer en train d'utiliser le système fini. Chaque scénario raconte une histoire : celle d'une variante particulière d'un cas d'utilisation.

Exemple pour le cas d'utilisation « Téléphoner », on pourrait avoir les scénarii :

- « Appeler une personne listée dans mes contact »,
- « Retourner un appel en absence »,
- « Composer un numéro »,
- « Prendre un double appel »,
- etc...

Nous formulons un scénario au moyen d'une partie identification et d'une partie de détails

2.2.1 Identification

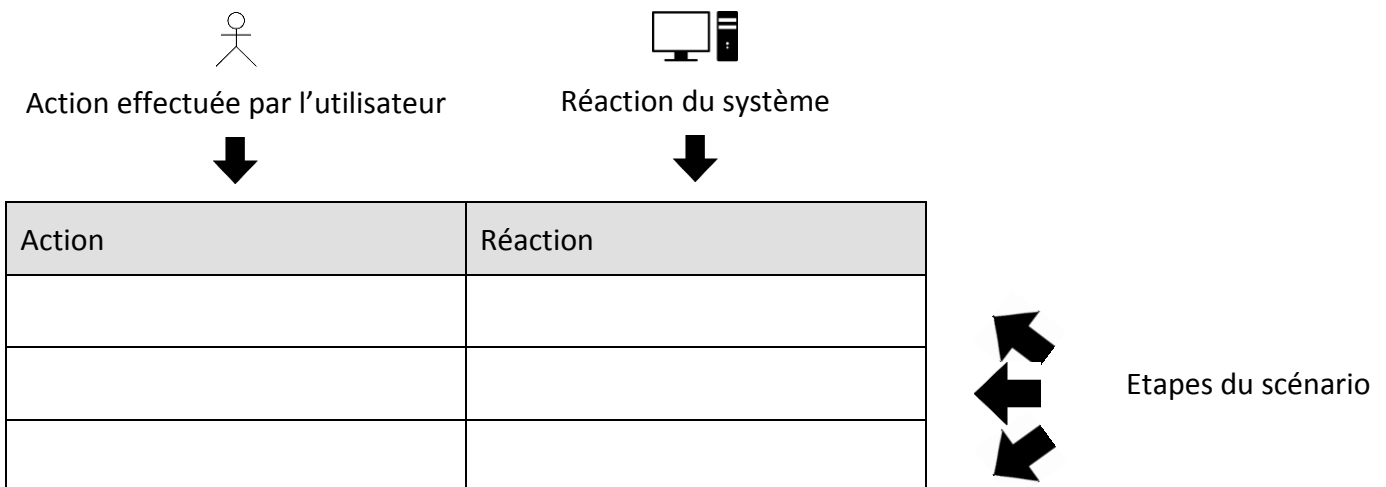
Il s'agit d'une description courte du scénario.

Identifiant		Un code unique à travers tout le projet
En tant que		Rôle
Je veux		Quelque chose
Pour		But
Charge estimée		En heures
Priorité		Selon MoSCoW

2.2.2 Détails

Cette partie raconte étape par étape l'histoire de notre scénario.

Cela se fait dans un tableau à deux colonnes :



Les règles suivantes s'appliquent pour assurer une bonne formulation du scénario :

1. Dans la première colonne, nous ne faisons mention que de l'utilisateur et de l'interface du système, jamais du comportement du système ;
2. Dans la deuxième colonne, nous ne faisons jamais mention de l'utilisateur ;
3. Rappelez-vous que le scénario raconte une histoire ! Chaque case du tableau décrit quelque chose de précis ; il ne peut pas y avoir – entre autres – de « si » ;
4. Les actions peuvent faire référence à des schémas (maquette d'interface, architecture du système, schéma de réseau, ...) ;
5. On peut décrire une action pour laquelle le système n'a pas de réaction. Ceci afin de mieux raconter notre histoire ;
6. Une action peut engendrer plusieurs réactions.

Exemple :

Identifiant	SMP0012
En tant que	Utilisateur
Je veux	Téléphoner
Pour	Appeler une personne de mes contacts
Charge estimée	40
Priorité	M

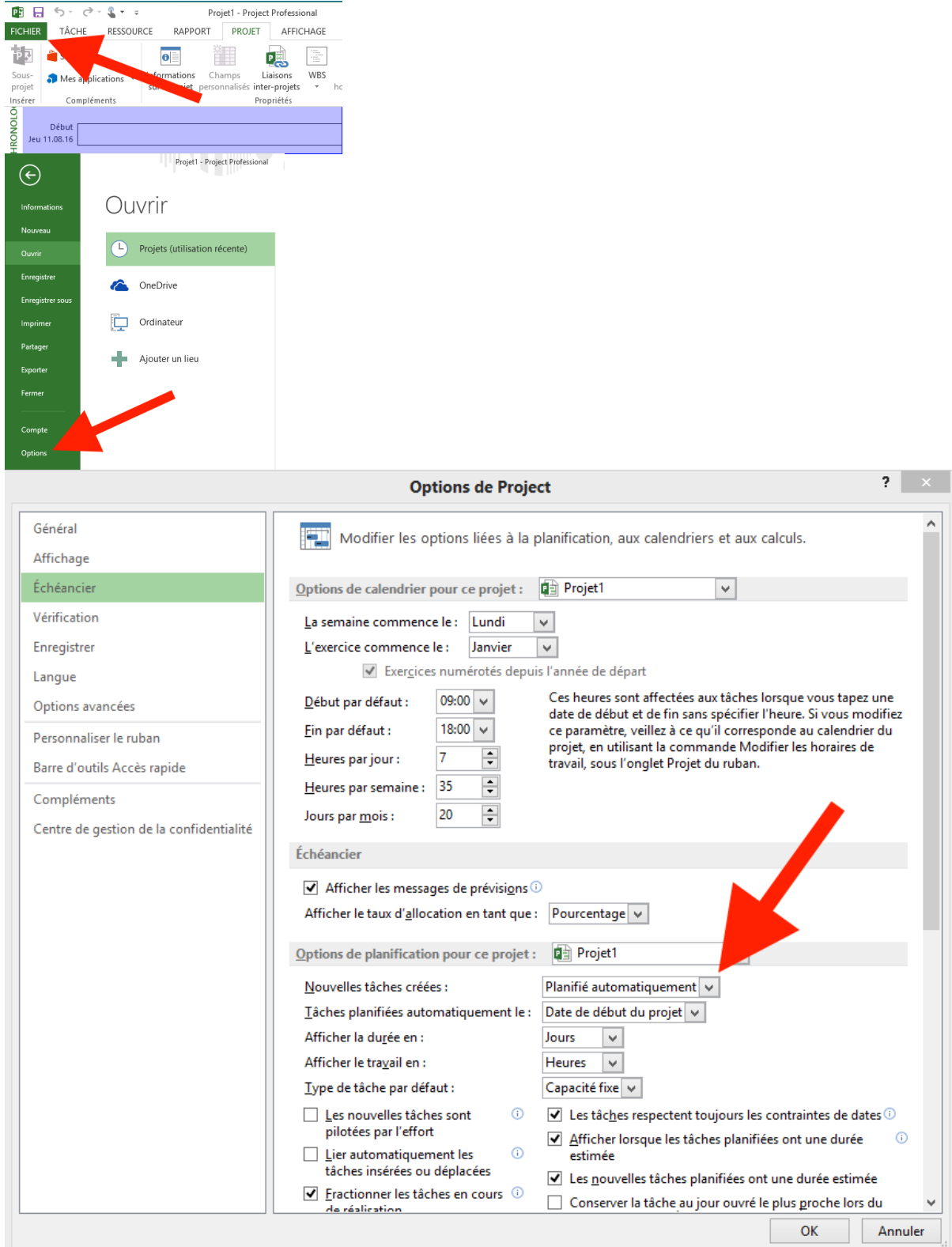
Action	Réaction
J'allume mon smartphone	L'écran s'allume et le système d'exploitation démarre
Je touche l'icône de l'application « téléphone »	L'application démarre
Je touche le bouton « contact »	La liste des contacts personnels s'affiche
Je touche le contact « John »	Les trois numéros de John s'affichent (portable, maison et travail)
Je touche le numéro « maison »	Composition du numéro de domicile de John
	Ça sonne mais John ne décroche pas dans les 30 secondes
	L'appel se termine, on retourne aux trois numéros de John
Je touche le numéro « portable »	Composition du numéro de portable de John
	Ça sonne et John décroche
Je discute de mes vacances avec John	
Je touche le bouton « raccrocher »	La communication s'interrompt
	L'application affiche un résumé de l'appel: Personne appelée, date, heure, durée et coût de l'appel
Je touche « OK »	L'application se referme

3 Planification

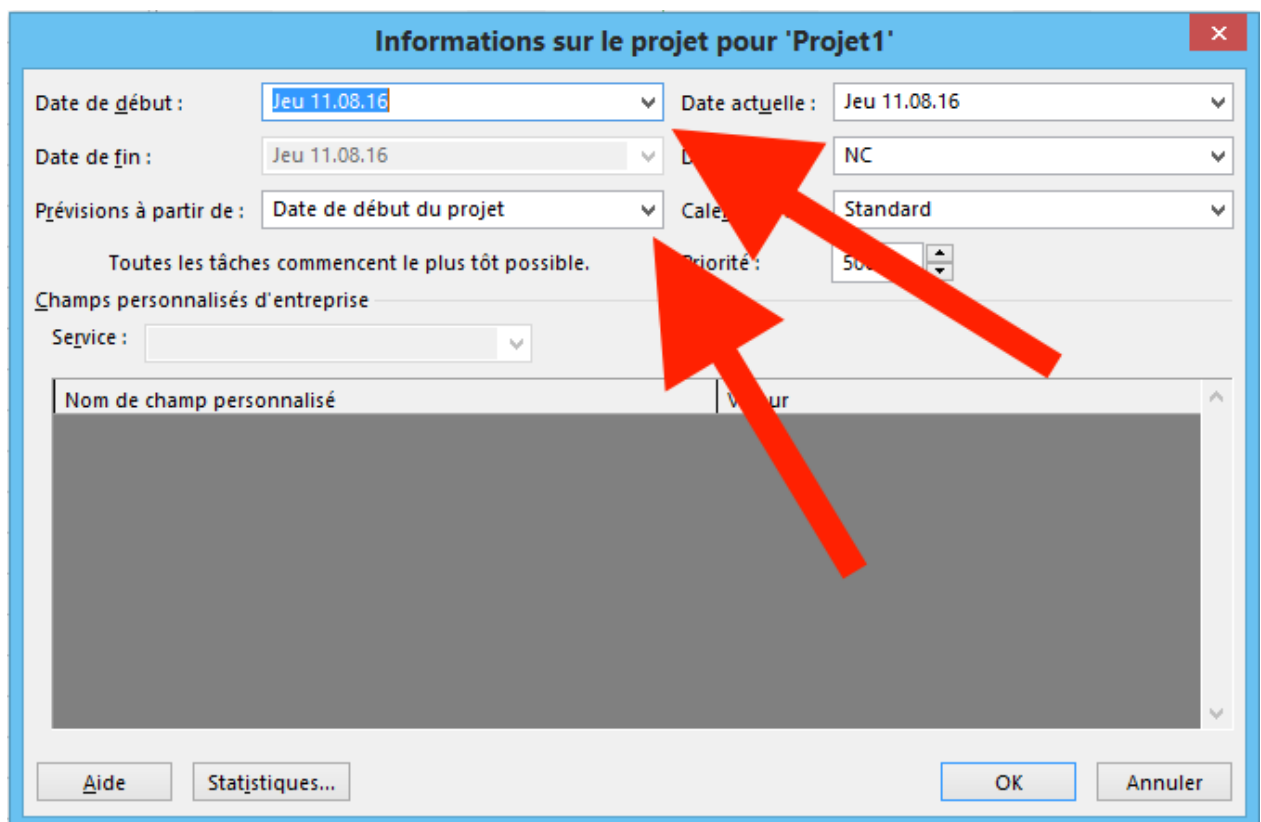
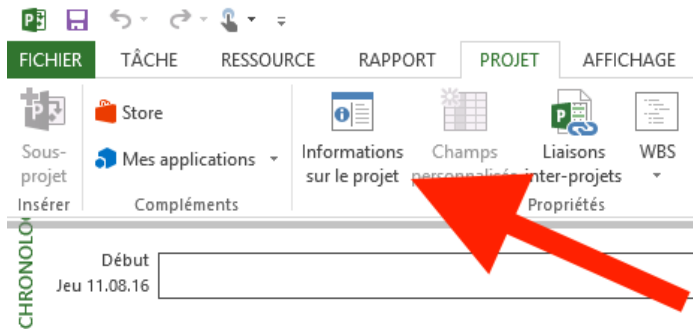
Nous gérons la planification avec MS-Project.

3.1 Préparation du fichier de planning

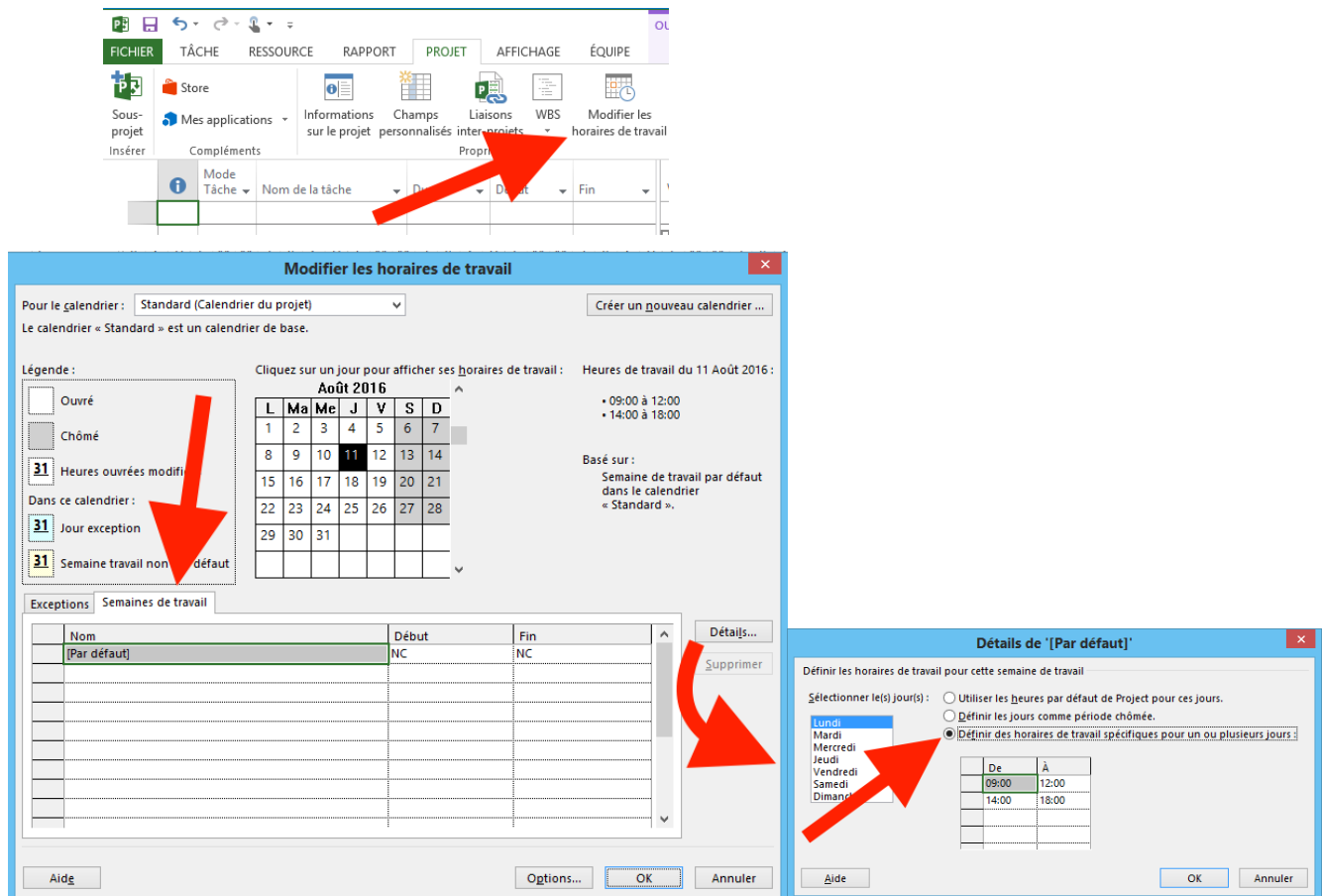
- 1) Dans les options MS-Project, changer « Nouvelles tâches créées » en « Planifié automatiquement »



- 2) Dans « Informations sur le projet », définir la date du début du projet et vérifier qu'on a bien « prévision à partir de » sur « Date de début du projet »



3) Entrer les horaires hebdomadaires



Modifier les horaires de travail

Pour le calendrier : Standard (Calendrier du projet) Créer un nouveau calendrier ...

Le calendrier « Standard » est un calendrier de base.

Légende :

- Ouvré
- Chômé
- 31 Heures ouvrées modifiées
- 31 Jour exception
- 31 Semaine travail non par défaut

Cliquez sur un jour pour afficher ses horaires de travail : **Août 2016**

Heures de travail du 11 Août 2016 :

- 09:00 à 12:00
- 14:00 à 18:00

Basé sur : Semaine de travail par défaut dans le calendrier « Standard ».

Détails de '[Par défaut]'

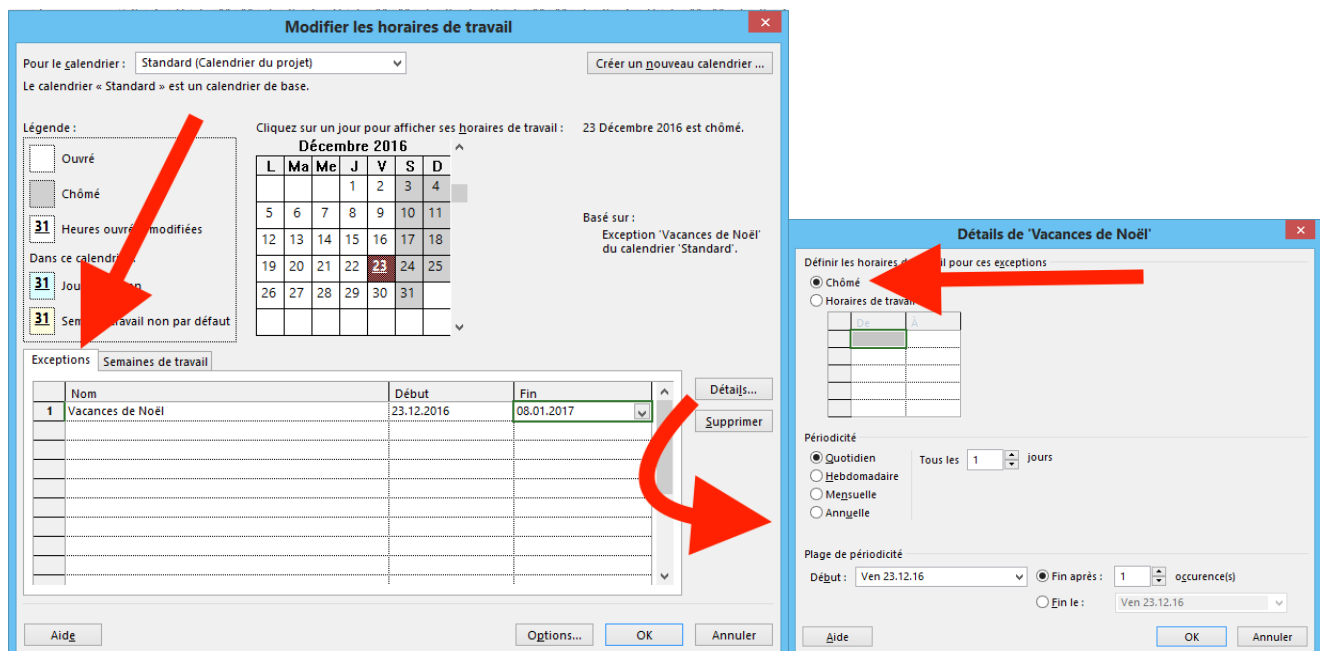
Définir les horaires de travail pour cette semaine de travail

Sélectionner le(s) jour(s) :

- ☐ Utiliser les heures par défaut de Project pour ces jours.
- ☐ Définir les jours comme période chômée.
- ☒ Définir des horaires de travail spécifiques pour un ou plusieurs jours :

De	À
09:00	12:00
14:00	18:00

4) Entrer les exceptions (vacances/jours fériés).



Modifier les horaires de travail

Pour le calendrier : Standard (Calendrier du projet) Créer un nouveau calendrier ...

Le calendrier « Standard » est un calendrier de base.

Légende :

- Ouvré
- Chômé
- 31 Heures ouvrées modifiées
- 31 Jour exception
- 31 Semaine travail non par défaut

Cliquez sur un jour pour afficher ses horaires de travail : **Décembre 2016**

23 Décembre 2016 est chômé.

Basé sur : Exception 'Vacances de Noël' du calendrier 'Standard'.

Détails de 'Vacances de Noël'

Définir les horaires de travail pour ces exceptions

☒ Chômé

☐ Horaires de travail

Périodicité

- ☒ Quotidien
- ☐ Hebdomadaire
- ☐ Mensuelle
- ☐ Annuelle

Tous les 1 jours

Plage de périodicité

Début : Ven 23.12.16 ☒ Fin après : 1 occurrence(s)

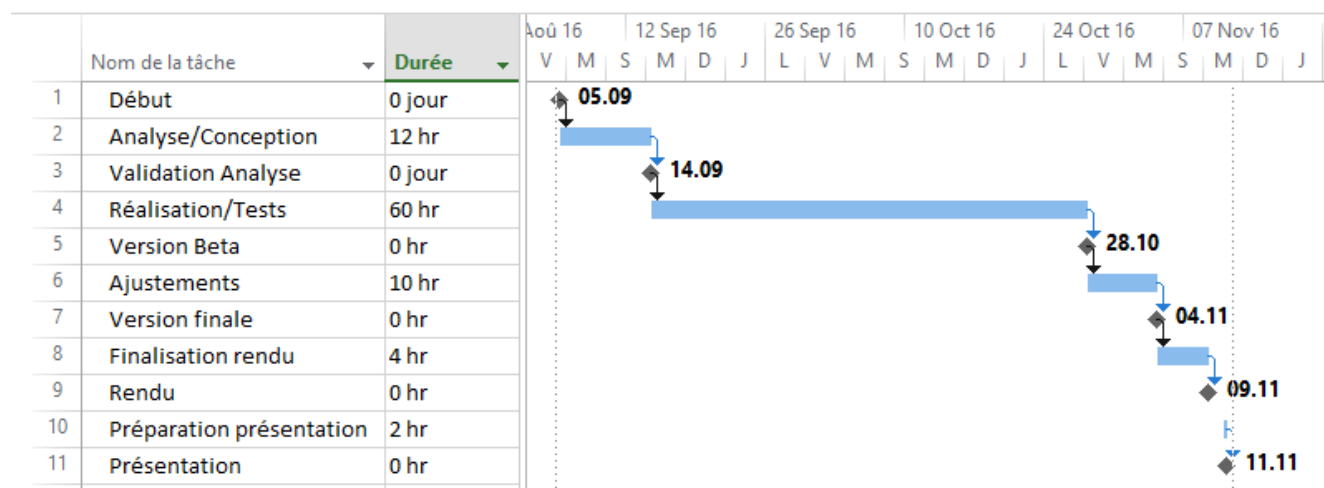
☐ Fin le : Ven 23.12.16

3.2 Planification initiale

Cette planification est grossière car on ne dispose que de peu d'informations à ce stade du projet. Elle n'est typiquement constituée que de phases générales et de jalons majeurs tels que :

- Début/Fin
- Validation des Use cases/scénarios avec le mandant
- Réalisation d'un Use case choisi
- Version Beta
- ...

Elle est présentée sous forme de diagramme de Gantt :



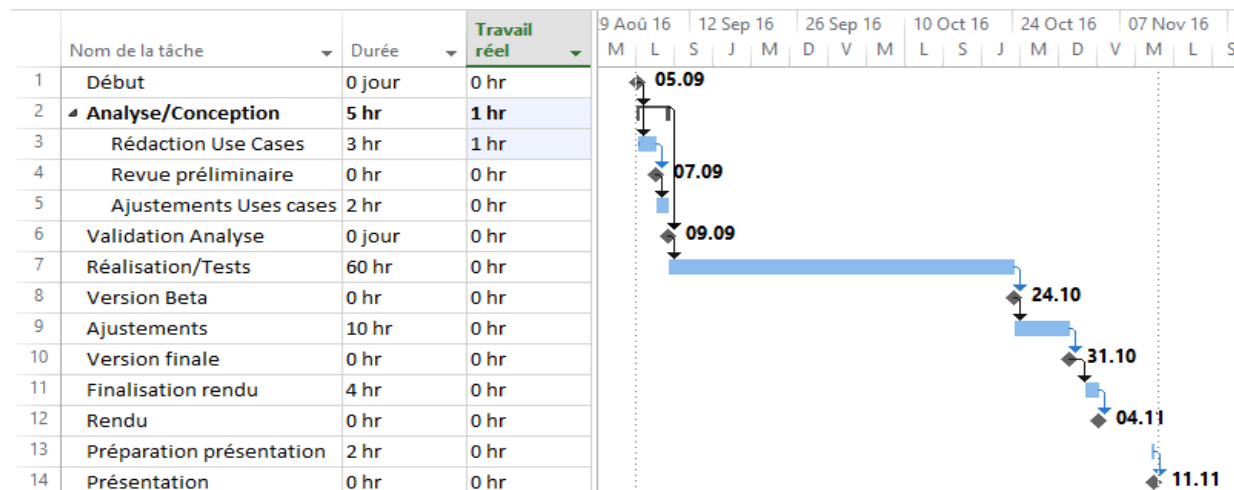
3.3 Mise à jour

A partir de là, nous effectuons au minimum une « rétro-planification » par semaine.

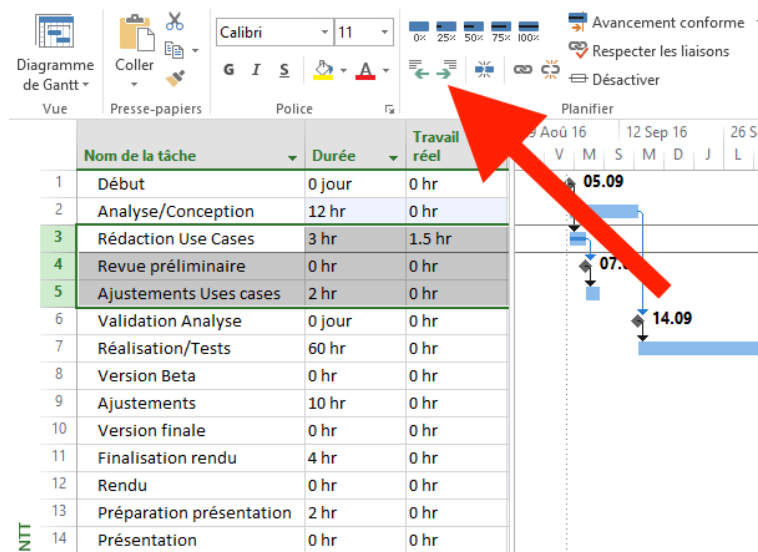
Une rétro-planification consiste à :

- Reporter dans le planning le travail effectué depuis la dernière rétro-planification ;
- Décomposer une tâche existante en deux ou plusieurs tâches SMART (voir section suivante) ;
- Décider des tâches qui seront exécutées dans les jours qui viennent ;
- Consigner les éventuels changements majeurs dans le journal de bord.

Typiquement, une première rétro-planification effectuée sur le planning de la section précédente pourrait mener au planning suivant :



On constate que la grosse tâche du planning initial est maintenant devenue une « tâche récapitulative ». Cela se fait en sélectionnant les tâches qui la suivent et en les « abaissant » :



Les tâches détaillées suivent les recommandations suivantes :

1. Elles sont SMART ¹:

- **Spécifique** : l'énoncé est clair et précis. Quand on le lit, on sait ce que l'on doit faire. La tâche ne définit qu'une seule chose à faire ;
- **Mesurable** : il est possible de vérifier de manière non ambiguë que la tâche est réalisée ou non ;
- **Ambitieuse** : la tâche représente un travail significatif ;
- **Réaliste** : la personne en charge de la tâche a les moyens et les compétences pour l'effectuer ;
- **Temporelle** : on a pu en estimer la durée, en heures.

2. Les plus courtes possibles (max 4-5 heures si possible)

3. Inutile de préciser les tâches faites en parallèle à petite dose (tenue du journal de bord, documentation...), mais en tenir compte dans l'estimation des autres tâches.

4. Prévoir, en revanche, une tâche de finalisation de la documentation (relecture, correction des fautes, brochage), qui prend souvent quelques heures.

¹ Le principe SMART est très répandu, mais pas standardisé. La définition donnée ici est propre au CPNV.

4 Détails de réalisation

Il est rare qu'un projet débouche du premier coup sur un produit fini. Il est donc primordial de fournir avec le travail rendu toutes les informations nécessaires pour qu'une autre personne puisse reprendre et continuer le projet.

Cela consiste au minimum à décrire :

- Les choix technologiques effectués, s'ils ne sont pas évidents ou imposés ;
- Le fonctionnement général du système, ainsi que les algorithmes spécifiques s'il y en a ;
- L'arborescence du répertoire « Résultat » de la livraison. Nous listons tous les fichiers que nous avons créés ou modifiés, avec une rapide description de leur contenu (des noms qui parlent !) ;
- Les librairies ou outils logiciels tiers utilisés (version, utilité, référence à l'éditeur/fabriquant) ;
- La description exacte du matériel ;
- La marche à suivre pour reconstruire le résultat final à partir de ce qui a été rendu ;
- Les éventuels login/password nécessaires pour utiliser le système dans son environnement de développement.

5 Tests

Nous fournissons trois éléments concernant les tests :

1. La Stratégie,
2. les Cas-Tests,
3. les Résultats.

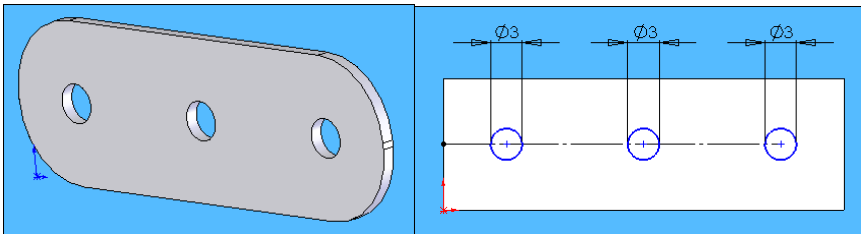
Ces trois points sont détaillés plus loin, mais commençons par quelques définitions.

5.1 Niveau de test

Nous appliquons jusqu'à trois niveaux de test :

5.1.1 Le Test unitaire

Il vérifie le bon fonctionnement d'un composant réalisé par une personne.



Exemple : on vérifie qu'une pièce de Mécano respecte les dimensions voulues

5.1.2 Le Test d'intégration

Il vérifie que deux ou plusieurs composants testés unitairement fonctionnent correctement ensemble.



Exemple : on vérifie que diverses pièces de Mécano peuvent bien être assemblées au moyen des vis et écrous.

5.1.3 Le Test Système

Il vérifie le bon fonctionnement du système complet dans l'environnement où il va être vraiment utilisé.



Exemple : on vérifie le bon fonctionnement de notre construction complète.

5.2 Type de test

Nous distinguons également trois types de tests :

- Le Test fonctionnel, qui vérifie que l'élément testé fait ce qu'on attend de lui ;
- Le Test de performance, qui vérifie que l'élément testé peut traiter la quantité d'information voulue dans le temps voulu ;
- Le Test de robustesse, qui vérifie que l'élément testé peut reprendre un fonctionnement normal après avoir passé par une situation anormale.



Exemple : considérons la machine ci-dessus.

Cette machine sert à imprimer des flyers, les plier, les insérer dans une enveloppe, fermer l'enveloppe et faire des paquets de 100 enveloppes fermées.

Un test fonctionnel vérifiera que les flyers sont bien pliés en trois volets égaux et qu'il y a bien 100 enveloppes par paquet au final

Un test de performance vérifiera que la machine est capable de produire 250 paquets de 100 enveloppes en une heure

Un test de robustesse vérifiera que si on fournit des enveloppes qui sont déjà remplies, le système les écarte et reprend son fonctionnement dès que les enveloppes fournies sont vides

5.3 Stratégie de test

Pour tout mandat/projet, nous définissons une stratégie de test. Celle-ci consiste à décrire :

1. Le matériel et logiciel dont on va avoir besoin pour faire nos tests. Chaque membre de l'équipe a son poste de travail, mais on se doit de faire des tests en dehors de l'environnement de développement. Exemples types :
 - Un PC inutilisé dans la salle de classe
 - Un ordinateur privé
 - Un ordinateur équipé d'un autre système d'exploitation
 - Une ou plusieurs machines virtuelles
 - Un hébergeur externe
 - ...
2. Les données de test que l'on va utiliser
3. Les personnes qui vont participer aux tests : camarades de classe, amis, famille, profs, ...
4. Le timing par rapport au projet. Juste avant de rendre le résultat final ? (Oui, mais pas seulement car on n'aura plus de temps pour corriger les erreurs) Dès qu'on a réalisé quelque chose de significatif ? (Oui mais pas seulement car rien ne garantit que ce qui marchait au début du projet marche encore à la fin)
5. Nous rassemblons ces informations dans un tableau « Type / Niveau ». Exemple :

Niveau Type	Unitaire	Intégration	Système
Fonctionnel	Effectué en cours de réalisation par chaque développeur. Aucun rapport n'est fourni.	Chaque développeur dépose son code dans le répertoire « En cours » sur le partage réseau commun. Il reprend le code des autres pour les mettre sur son poste de travail. Aucun rapport n'est fourni.	L'application compilée est installée sur la VM de test « Projet XXX » . A ce moment, les cas-tests sont exécutés afin de garantir le bon fonctionnement du programme.
Performance	Aucun	Aucun	Installation du logiciel sur le serveur, chargement d'une base de données de 10'000 membres. Cas-tests : <ul style="list-style-type: none">• Perf 10'000
Robustesse	Effectué en cours de réalisation par chaque développeur. Aucun rapport n'est fourni.	Aucun	Sur la VM « Projet XXX ». Cas-tests : <ul style="list-style-type: none">• Doublons• Coupure réseau

5.4 Cas-tests

Un cas-test n'est rien d'autre qu'un scénario où on prévoit de la place pour écrire :

- Le contexte dans lequel on l'exécute : quels matériel, logiciel, données sont utilisés (tirés de la stratégie de test)
- La personne qui l'exécute (tirée de la stratégie de test)
- Le résultat **réellement observé** sur le système pour chaque étape du scénario

Exemple :

Synopsis	Réservations d'équipe
Environnement	Visual Studio + SQL Server Management Studio
Objectif	Valider la présentation des réservations faites par le responsable un groupe
Données	Res Ecoliers.txt, Res Ecoliers – erreurs.txt
Pré-requis	
Auteur	X. Carrel

	Action	Résultat attendu	Résultat
1.	Lancement de l'application, création de l'utilisateur JD	L'utilisateur est loggé, mais l'accès à la page « Bookings » n'est pas possible (pas de lien qui y mène)	<input type="checkbox"/>
2.	On change l'URL pour aller directement à la page bookings (localhost/Pages/Bookings)	On se retrouve sur la page d'accueil	<input type="checkbox"/>
3.	Dans SQLServer Management Studio, on définit JD comme entraîneur des Ecoliers. Dans le navigateur : déconnexion/reconnexion de JD	Un lien vers la page « Bookings » est affiché sous le titre de « Réservations d'équipe »	<input type="checkbox"/>
4.	Clic sur « Réservations d'équipe »	La page titre « Réservations pour : Ecoliers », puis le message « Aucune »	<input type="checkbox"/>
5.	Dans SQLServer Management Studio, on crée une réservation à la main dans la table booking.	La réservation apparaît dans un tableau à 4 colonnes :	<input type="checkbox"/>
6.	Cliquer sur « Choisissez un fichier », sélectionner le fichier « Res Ecoliers.txt », cliquer « Importer »	Un message dit que 6 réservations ont été importées. Le tableau est complété	<input type="checkbox"/>
7.	Cliquer sur « Choisissez un fichier », sélectionner le fichier « Res Ecoliers - erreurs.txt », cliquer « Importer »	Un message dit que 2 réservations ont été importées. Les 4 mauvaises réservations sont affichées Le tableau est complété avec deux réservations	<input type="checkbox"/>

5.5 Plan de test

L'ensemble des cas-tests de notre projet est appelé « **Plan de test** ».

5.6 Résultats de test

La publication des résultats consiste à donner après exécution des tests :

1. Un tableau de synthèse des résultats. Exemple :

Cas-test	Date	Personne	Résultat	Commentaire
Inscription	7.10.16	Julien	OK	
MdP Perdu	8.10.16	Julien	KO	L'email n'est pas envoyé
Réservation	9.10.16	Julien	OK	
Doublons	10.10.16	Julien	OK	

2. Le plan de test complet, avec les champs de résultat remplis par le testeur pour chaque cas-test
3. La liste des problèmes connus

6 Journal de Travail et Journal de Bord

Tout au long du projet, le mandataire maintient à jour deux journaux distincts.

6.1 Le Journal de Travail

Il a pour but de savoir qui a passé du temps (et combien) sur quelle tâche. Il sert typiquement à :

- Introduire le nombre d'heures effectives dans le fichier MS-Project durant la rétro-planification ;
- Facturer le travail (s'il y a lieu).

Exemple :

Personne	Date	Tâche	Durée(h)	Commentaire
Julien	7.9.16	Rédaction Use Cases	1.5	25 minutes d'installation de logiciel (Visio)
Julien	8.9.16	Rédaction Use Cases	1	
Julien	9.9.16	Ajustements Use Cases	0.5	
Julien	10.9.16	Codage UC 1	4	30 minutes perdues à cause d'un problème de compatibilité de librairies

6.2 Le Journal de Bord

Il a pour but de retracer l'histoire du projet au travers de ses faits marquants

Exemple :

Date	Événement
8.9.16	Revue des Use cases avec M. Gates : des ajustements sont demandés au niveau de la gestion des membres
9.9.16	Revue des Use cases avec M. Gates : ils sont validés
9.9.16	Rétro-planification de planning : aucun problème en vue
14.9.16	M. Gates demande (par mail) l'ajout d'une page d'historique des achats

7 Documentation de Projet

La documentation d'un projet réalisé dans la filière informatique du CPNV consiste au final en un document (livré en format PDF) contenant :

1. Une introduction avec :
 - Une description générale du projet
 - Les informations sur le mandant du projet
 - Les informations sur le mandataire (nous)
 - Des références (pas de copie !) aux documents fournis en guise d'énoncé (cahier des charges, projet précédent, emails explicatifs, ...)
2. L'analyse, telle que décrite au chapitre 2 de ce document
3. La planification initiale, telle que décrite au chapitre 3.2 de ce document
4. Les détails de réalisation, tels que décrits au chapitre 4 de ce document
5. Les détails de la livraison, telle que décrite au chapitre 9 de ce document
6. Le Journal de Bord, tel que décrit au chapitre 6.2 de ce document
7. Une conclusion, contenant entre autres :
 - Un commentaire critique de comparaison entre ce qui était demandé et ce qui a été réalisé
 - Le « planning réel » résultant des rétro-planifications, ainsi qu'une comparaison de celui-ci avec le planning initial
 - La liste des problèmes connus
 - Un commentaire personnel sur l'ensemble du projet

8 Communiquer sur l'avancement des travaux

Pour tenir le mandant informé de l'avancement des travaux, nous lui communiquons à intervalles réguliers :

- Le Journal de Bord,
- Le Journal de Travail,
- Le Document de Projet.

Sauf cas exceptionnel demandé spécifiquement par le mandant, tous les documents sont toujours transmis en format PDF.

De par leur structure, il est facile pour le mandant de détecter les changements intervenus dans les journaux entre deux envois : il suffit d'aller consulter les bas du tableau, qui est organisé par ordre chronologique.

Il n'en va pas de même pour le Document de Projet, dans lequel des modifications difficilement décelables peuvent se situer dans des endroits que le mandant a déjà lu plusieurs jours auparavant.

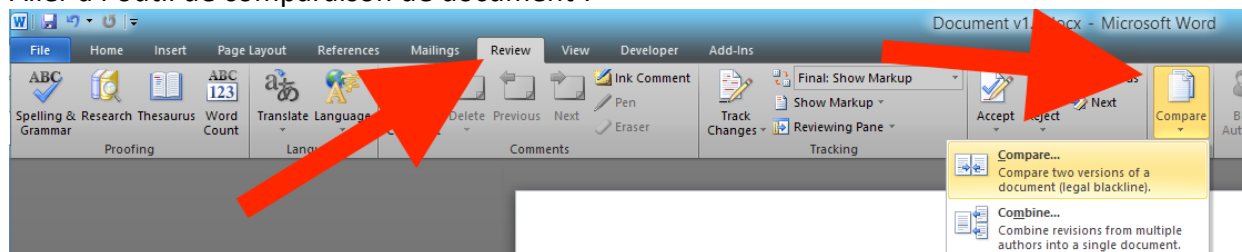
Pour faciliter sa lecture, mais aussi pour nous aider à bien documenter les changements, nous fournissons deux versions : la nouvelle version et une qui met en évidence les modifications de contenu intervenues depuis la version précédente.

Exemple : si l'on travaille sur un document « Projet Alpha.docx » et que l'on publie la version 1.3, on fournira :

1. « Projet Alpha v1.3.pdf »
2. « Projet Alpha v1.3 – différences.pdf »

Pour créer la version « ... - différences » :

1. Aller à l'outil de comparaison de document :








2. Sélectionner les deux versions, tout décocher (sauf « tables ») dans les options :

3. Effectuer la comparaison (OK) et sauver le résultat au format pdf.

9 Livraison

Nous livrons le résultat de notre travail au moyen d'un seul fichier archive (format ZIP).

L'extraction du contenu doit être comme l'exemple ci-dessous :

 Projet Alpha	Un répertoire qui porte le nom du projet
 Documentation	Un répertoire de documentation
 Projet Alpha.pdf	Le document de projet
 Annexes	Tout autre document utile, référencé dans le document de projet
...	
 Résultat	Résultat produit : code, scripts, manuels utilisateur
...	La structure de ce dossier est décrite en section 4

Ce fichier est remis au mandant et au chef de projet (s'il s'agit d'une personne différente) par un moyen convenu à l'avance tel que :

- Un support physique (CD, clé USB, Disque externe, etc.),
- Un emplacement sur un partage réseau,
- Un dossier partagé sur le cloud (Dropbox, Google Drive, ...).

10 Présenter le résultat final