18/09/2014

Crépin Aurélia, Levrat Gael, Ribierre Paul, Sagon Nicolas Veys Arthur

CPE Groupe E

Plan d’assurance qualité

Projet : Les trésors de Java

#### Objet du document :

Ce plan d'assurance qualité (PAQ) sert à décrire l'ensemble des dispositions spécifiques prises pour assurer la qualité du produit fourni dans le cadre du projet “Les trésors de java” ainsi que la qualité du processus de développement.

#### Responsabilité du document :

|  |  |  |
| --- | --- | --- |
| Rédacteurs | Vérificateurs | Validateur |
| CREPIN Aurélia - Expert Environnement et Qualité | CREPIN Aurélia - Expert Environnement et Qualité | Comité de projet |
| LEVRAT Gaël - Expert  Modélisation | RIBIERRE Paul - Chef de Projet |  |
| RIBIERRE Paul - Chef de Projet  (Gestion et Management de projet) |  |  |
| SAGON Nicolas - Expert Programmation Objet |  |  |
| VEYS Arthur - Expert Technique et Environnement |  |  |

#### Historique des modifications et validation du document :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Date | Parties modifiées | Nature du  changement | Statut |
| V0 | 16/09/2014 |  | Création | Obsolète |
| V1 | 16/09/2014 |  | Publication | Applicable |

#### Diffusion du document :

|  |  |
| --- | --- |
| Entités | Noms |
| Comité de projet | CREPIN Aurélia  LEVRAT Gaël  RIBIERRE Paul  SAGON Nicolas  VEYS Arthur |
| Examinateur CPE | TROUILLOT Xavier |

#### État du document : Applicable

Table des matières

[1. But, domaine d’application et responsabilités 4](#_Toc398844638)

[1.1. Objectifs du Projet 4](#_Toc398844639)

[1.2. Domaine d’application 4](#_Toc398844640)

[1.3. Responsabilités 4](#_Toc398844641)

[2. Documents applicables et documents de référence 5](#_Toc398844642)

[2.1. Documents applicables 5](#_Toc398844643)

[2.2. Documents de référence 5](#_Toc398844644)

[3. Terminologie 5](#_Toc398844645)

[4. Organisation 5](#_Toc398844646)

[4.1. Organisation client 5](#_Toc398844647)

[4.2. Relation avec le client 6](#_Toc398844648)

[4.3. Organisation de l’équipe 6](#_Toc398844649)

[4.3.1. Matrice des compétences 6](#_Toc398844650)

[5. Rôles et responsabilités 7](#_Toc398844651)

[6. Responsabilités des principaux livrables 8](#_Toc398844652)

[7. Démarche de développement 9](#_Toc398844653)

[7.1. Cycle de développement 9](#_Toc398844654)

[7.1.1. Analyse des besoins 9](#_Toc398844655)

[7.1.2. Spécifications 9](#_Toc398844656)

[7.1.3. Conception 9](#_Toc398844657)

[7.1.4. Développement 10](#_Toc398844658)

[7.1.5. Tests unitaires 10](#_Toc398844659)

[7.1.6. Intégration 10](#_Toc398844660)

[7.1.7. Validation 10](#_Toc398844661)

[7.2. Gestion des risques 11](#_Toc398844662)

[8. Documentation 12](#_Toc398844663)

[8.1. Processus de rédaction et de validation des documents 12](#_Toc398844664)

[8.2. Nomenclature des documents 12](#_Toc398844665)

[8.3. Gestion de la Documentation 12](#_Toc398844666)

[8.3.1. États des documents 12](#_Toc398844667)

[8.3.2. Identification des documents 12](#_Toc398844668)

[8.3.3. Structuration d’un document 12](#_Toc398844669)

[8.3.4. Validation d’un document 12](#_Toc398844670)

[8.3.5. Péremption d’un document 12](#_Toc398844671)

[8.3.6. Archivage des documents 13](#_Toc398844672)

[8.3.7. Documents applicables 13](#_Toc398844673)

[8.3.8. Documents de référence 13](#_Toc398844674)

[9. Gestion de configuration 13](#_Toc398844675)

[10. Gestion des modifications 14](#_Toc398844676)

[11. Méthodes, outils et règles 15](#_Toc398844677)

[11.1. Outils de pilotage 15](#_Toc398844678)

[11.2. Méthode et outils de développement 15](#_Toc398844679)

[11.2.1. Conception 15](#_Toc398844680)

[11.2.2. Standard de développement 15](#_Toc398844681)

[11.2.3. Environnement de développement 16](#_Toc398844682)

[11.2.4. Forme 17](#_Toc398844683)

[12. Contrôle des fournisseurs. 18](#_Toc398844684)

[13. Reproduction, protection, livraison 19](#_Toc398844685)

[13.1. Reproduction, protection 19](#_Toc398844686)

[13.2. Livraison, protection 19](#_Toc398844687)

[14. Vérifications et Suivi du Plan Qualité 19](#_Toc398844688)

[14.1. Résolution des non-conformités 20](#_Toc398844689)

# But, domaine d’application et responsabilités

## Objectifs du Projet

Le projet vise au développement d’un jeu numérique basé sur le jeu de plateau “Attention Pirates!”. Les attentes et contraintes liées à la mise en œuvre de ce logiciel sont définies dans le cahier des charges fourni par le maître d’ouvrage Mr Xavier TROUILLOT représentant la société CPE.

Cette alternative numérique sera nommée “Trésors de Java”.

## Domaine d’application

Ce document décrit les procédures qualité afférentes au projet “Trésors de Java”. Les dispositions décrites dans ce plan d'assurance et de contrôle de la qualité couvrent tout processus de conception et de développement du projet, ainsi que les livrables concernés par ce projet.

## Responsabilités

La maîtrise d’Ouvrage est représentée par Mr Xavier TROUILLOT – CPE (MOA)

La maîtrise d’Œuvre est représentée par le groupe E composé des personnes suivantes: (MOE)

CREPIN Aurélia : Responsable Qualité et Outils de Développement

LEVRAT Gaël : Responsable Modélisation et Architecture

RIBIERRE Paul : Chef de Projet et référent Modélisation

SAGON Nicolas : Responsable Technique Programmation Objet

VEYS Arthur : Responsable Environnement, Outils, et Veille Technologique

# Documents applicables et documents de référence

Les documents applicables définissent les documents qui seront créés pendant la vie du projet. Les documents de référence sont les documents qui ne sont pas créés durant le projet, mais sur lesquels les documents applicables s’appuient. Ces documents doivent être définis dans le plan qualité associé au projet.

## Documents applicables

* Le plan qualité
* Le dossier de spécification
* Le standard de conception détaillée
* Les fiches de compte rendu
* Le cahier de recette
* La documentation utilisateur

## Documents de référence

* Les règles du Jeu
* Les documentations techniques des environnements et des outils de développement utilisés.

# Terminologie

* CDC : Cahier des Charges
* CDP : Chef de Projet
* CNQ : Cout d’obtention de la non-qualité
* PAQ : Plan d’assurance
* RQ : Responsable Qualité
* MCO : Maintien en conditions opérationnelles
* IDE : Environnement de développement
* Comité : l’ensemble de l’équipe du projet

# Organisation

## Organisation client

Le déroulement du projet est décrit par le document fourni par la maîtrise d’ouvrage : https://e-campus.cpe.fr/mod/resource/view.php?id=14447

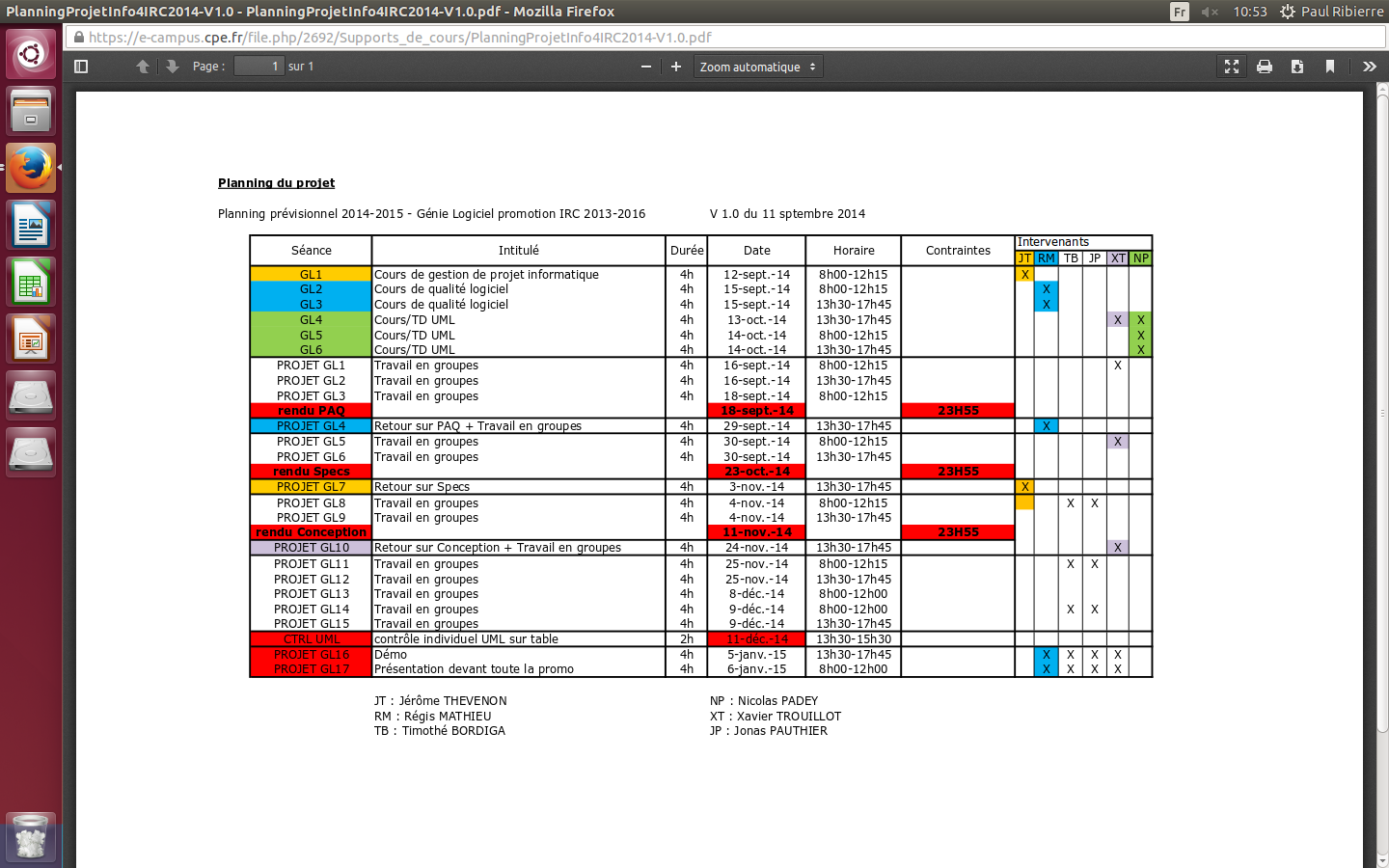


Figure 1: planning prévisionnel du projet

## Relation avec le client

Il est prévu des réunions avec le client selon le planning ci-dessus. Les documents devants faire l’objet d’une présentation sont, le plan qualité, les spécifications, et la modélisation UML du projet. L’application fera l’objet d’une présentation et démonstration et devra répondre aux contraintes dictées par ces documents.

## Organisation de l’équipe

Dans le cadre de ce projet, le comité de projet est constitué de : CREPIN Aurélia, LEVRAT Gaël, RIBIERRE Paul, SAGON Nicolas et VEYS Arthur.

### Matrice des compétences

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Compétence ou domaine | CREPIN Aurélia | LEVRAT Gaël | RIBIERRE Paul | SAGON Nicolas | VEYS Arthur |
| Java | TBC | TBC | TBC | TBC | TBC |
| Conception UML | CT | BC | BC | BC | TBC |
| Qualité | TBC | BC | CT | CT | CT |
| Communication | BC | BC | BC | BC | BC |

Légende : CT : Connaissance Théorique BC : Bonne Connaissance

TBC : Très Bonne Connaissance EX : Expert

# Rôles et responsabilités

#### Chef de projet :

Le chef de projet doit veiller à la direction et à la coordination des ressources d’un projet de manière à atteindre les objectifs de celui-ci. Il a en charge la rédaction et la vérification des documents de gestions de projet associés.

#### Responsable Qualité :

Le responsable qualité doit établir et mettre à jour le plan d’assurance qualité. Il doit veiller également à l’application du plan d’assurance qualité lors de toutes les étapes du projet.

#### Responsable Technique Programmation Objet:

Le responsable technique est un référent expert en programmation en charge du code source de l’application.

#### Responsable de la Modélisation:

Le responsable modélisation doit spécifier, modéliser et construire les documents nécessaires au bon développement d'un logiciel orienté objet.

#### Responsable Environnement, Outils, et Veille Technologique:

Le responsable d’environnement est chargé de déployer et manager un ensemble d'outils pour augmenter la productivité des programmeurs.

# Responsabilités des principaux livrables

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Numéro de Modèle | Documents du kit de gestion de projet | Chef de projet | Responsable qualité | Responsable Technique Programmation | Responsable de la modélisation | Responsable Environnement |
| 1 | Plan d’Assurance Qualité | Rédaction  Vérification | Rédaction  Vérification | Rédaction  Vérification | Rédaction  Vérification | Rédaction  Vérification |
| 2 | Dossier de spécification | Rédaction  Vérification | Vérification | Rédaction | Rédaction | Rédaction |
| 3 | Le standard de conception détaillée | Rédaction  Vérification | Vérification | Rédaction | Rédaction | Rédaction |
| 4 | Les fiches de compte rendu | Rédaction  Vérification | Rédaction  Vérification | Rédaction | Rédaction | Rédaction |
| 5 | Le cahier de recette | Vérification | Vérification | Rédaction | Rédaction | Rédaction |
| 6 | La documentation utilisateur | Rédaction  Vérification | Rédaction  Vérification | Rédaction  Vérification | Rédaction  Vérification | Rédaction  Vérification |

# Démarche de développement

## Cycle de développement

Le cycle de développement mis en place dans le cadre du projet est un cycle de développement en V. Il permet, en cas d'anomalie, de limiter un retour aux étapes précédentes. Les phases de la partie montante doivent renvoyer de l'information sur les phases en vis-à-vis lorsque des défauts sont détectés, afin d'améliorer le logiciel.

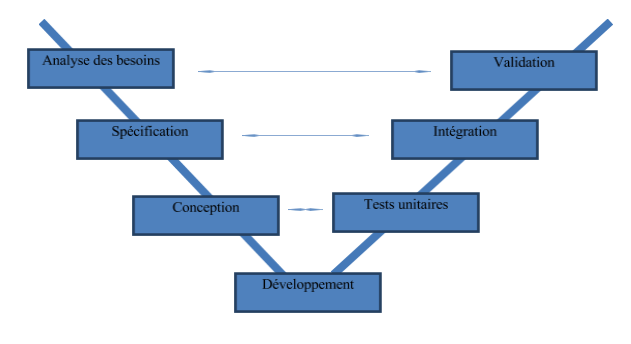


Figure 2:Description des étapes du cycle en V

### Analyse des besoins

Cette étape permet de prendre en compte les besoins du client ainsi que son cahier des charges, elle décrit l'existant, les attentes et les exigences générales exprimées par le client. Elle permet aussi de définir un planning prévisionnel, de mettre en place un environnement de programmation ainsi qu’une démarche qualité propre au sujet.

### Spécifications

.Une spécification fonctionnelle décrit comment un produit fonctionnera, entièrement du point de vue de l'utilisateur. Elle ne s'attache pas à la façon d'implémenter la chose. Elle parle de fonctionnalités. Elle spécifie les écrans, les menus, les dialogues et ainsi de suite. Les spécifications fonctionnelles d’attention pirates sont  l’ensemble de ses règles. Elles se trouvent dans le document suivant: règles du jeu.pdf se trouvant sur l’e-campus.

### Conception

Cette étape met en place l'organisation des différents éléments du système défini dans le dossier de spécification et des relations entre ces éléments. Dans le cadre de ce projet, la structure du système informatique sera  représentée sous forme de graphiques UML tel que les diagrammes de classe, les diagrammes de séquence et les cas d’utilisation.

### Développement

C’est l'ensemble des activités liées à la création du logiciel. Cette étape correspond à la rédaction du code source du logiciel en rapport avec le dossier de spécification et le dossier de conception UML.

### Tests unitaires

Cette étape permet de s'assurer du fonctionnement correct d'une partie du logiciel, elle permet de confronter une réalisation à sa spécification, et de s'assurer qu'elle fonctionne correctement en toutes circonstances. En cas d’anomalie ou de différence avec les résultats attendus, le programme sera modifié pour répondre aux spécifications initiales.

### Intégration

Une fois que les développeurs ont chacun validé leurs développements lors des tests unitaires, cette étape va regrouper leurs modifications dans le cadre d'une livraison. Les tests d'intégration ont pour but de valider le fait que toutes les parties développées indépendamment fonctionnent bien de façon cohérente.

### Validation

Cette étape permet de vérifier si toutes les exigences client décrites dans le document de spécification sont respectées. Elle permet la livraison du logiciel final.

## Gestion des risques

Les principaux risques liés au projet sont :

|  |  |  |  |
| --- | --- | --- | --- |
| Description du risque | Gravité  (1-5) | Impact | Action |
| Retard Livraison | 5 | Non validation du projet | Gestion du projet avec redmine. Responsabilité du chef de projet |
| Fonctionnalités non-implémentées | 3 | Non-respect des spécifications | Répartition des tâches. Diagramme de Gantt |
| Application non- fonctionnelle | 3 | Fonctionnalités limitées ou inexistantes | Test unitaire avec JUnit.  Test de non-régression avec Jenkins |
| Perte de données | 4 | Grave retard | Utilisation de Git sur un serveur dédié |

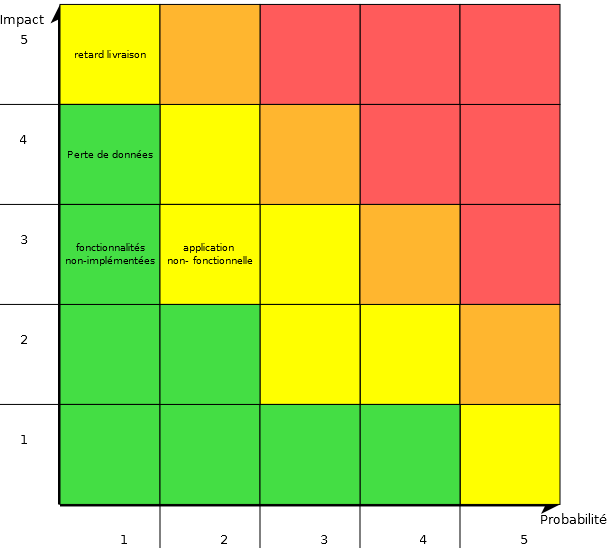


Figure 3: Evaluation des risques

# Documentation

## Processus de rédaction et de validation des documents

Chaque document produit doit répondre aux exigences du plan qualité. Un nouveau document doit être soumis à la validation du Comité Projet et particulièrement du responsable qualité et du chef de projet. Après validation des modifications apportées, le nouveau document est publié et envoyé aux parties par le Responsable Qualité.

## Nomenclature des documents

Tout document est nommé en fonction de la nomenclature suivante :

#### Les\_tresors\_de\_JAVA-[nom\_du\_document]-[version]-[date]

Ainsi, par exemple, le nom du Plan d'Assurance Qualité en sa première version et datant du 16/09/2014 serait « Les\_trésors\_de\_JAVA-Plan\_Assurance\_Qualité-V0-16092014».

## Gestion de la Documentation

Le processus interne d’assurance qualité traite de la maîtrise et du cycle de vie des documents, depuis leur création jusqu'à leur destruction.

### États des documents

Tous les documents sont susceptibles d’évoluer et possèdent un des états suivant :

* Brouillon
* En attente de Validation
* Document Applicable
* Document Obsolète

Cet état se trouve en entête du document.

### Identification des documents

Tout document possède un nom unique.

### Structuration d’un document

Tout document contient des informations obligatoires (Rédacteur, état, version,…).

### Validation d’un document

Tout document doit être validé par la personne responsable du document.

### Péremption d’un document

Un document obsolète ne doit pas pouvoir être utilisé par inadvertance. Seule la version courante sera disponible. Une possibilité de récupérer un document précédent existe grâce à l’outil de gestion de versions.

### Archivage des documents

Les documents susceptibles d’être consultés ultérieurement doivent être archivés sur le Git.

### Documents applicables

Les procédures de type applicables sont des documents dont l'application est imposée et vérifiable. Les documents applicables du projet sont tous ceux dont le statut est “Document Applicable”.

### Documents de référence

Les documents de référence sont des documents permettant d'effectuer le développement mais qui ne sont pas imposés.

Les documents de référence du projet sont les documentations :

JavaDoc :<http://docs.oracle.com/javase/7/docs/technotes/guides/javadoc/>

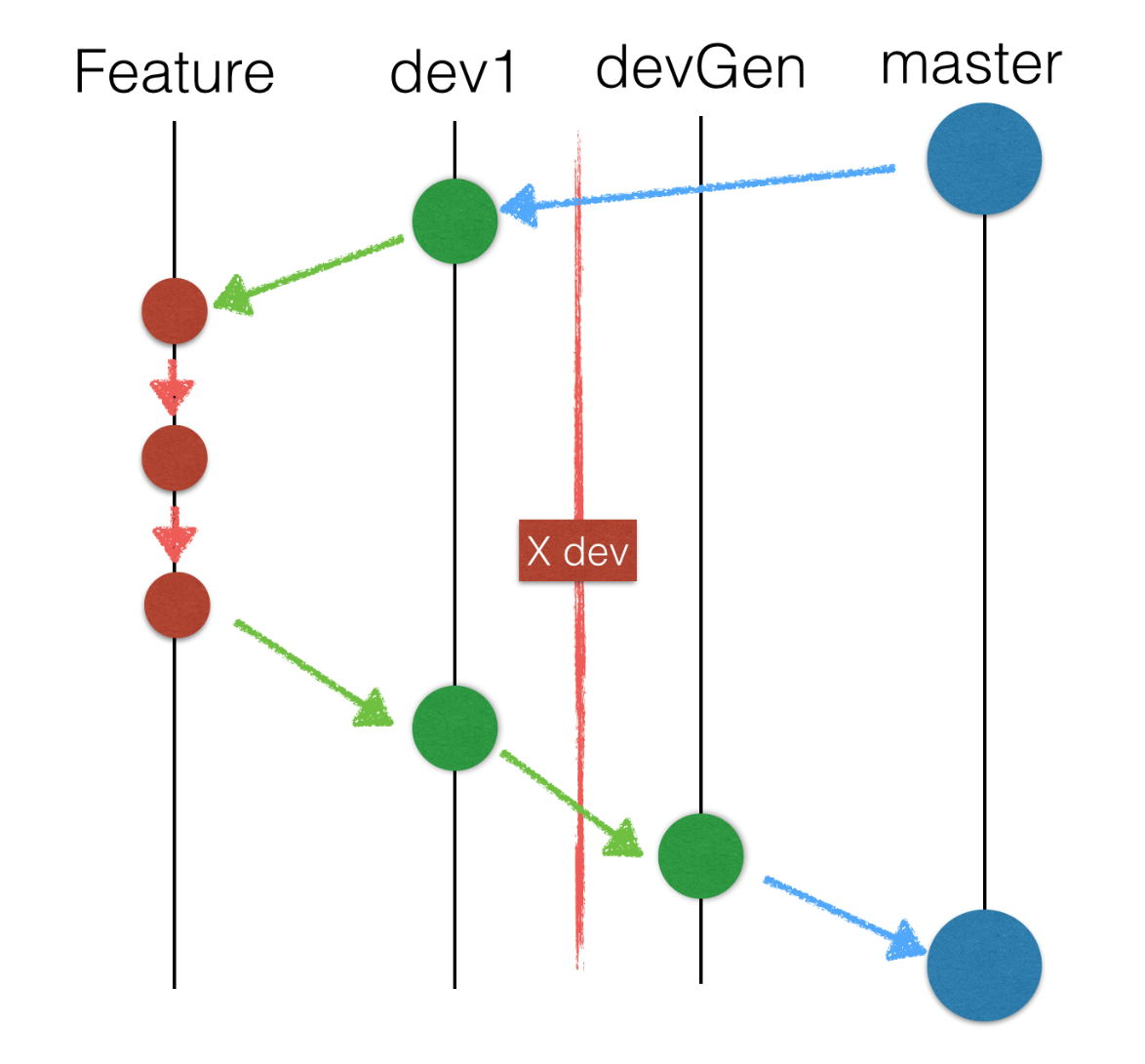
Redmine : <http://www.redmine.org/projects/redmine/wiki>

Git : <http://git-scm.com/doc>

Jenkins: <https://wiki.jenkins-ci.org/display/JENKINS/Home>

# Gestion de configuration

L’outil de gestion de version Git permettant le développement par branche, nous utiliserons une méthode de développement inspirée de Git-Flow :



Chaque développeur va travailler sur sa branche de développement et développer ses fonctionnalités soit directement à l'intérieur de celle-ci soit dans des branches subsidiaires (au choix du développeur). Une fois sa branche développement prête à être intégré, elle est ajoutée dans la branche de développement générale où l’intégration de toutes les branches utilisateurs est effectuée. Une fois les tests d’intégrations terminés, tout est envoyé dans la branche master pour créer une nouvelle version. Jenkins permettra de compiler automatiquement la branche master ou bien la branche de développement générale, ensuite l’outil SonarQube analysera notre code pour rechercher les optimisations possibles et les erreurs éventuelles.

A chaque intégration, la branche master se verra attribué une nouvelle version. L’application commencera à la version 1.0. Ensuite à chaque nouvelle intégration nous incrémenterons le décimal (1.1, 1.2 …). Quand nous livreront une nouvelle version avec beaucoup de nouvelles fonctionnalités, nous incrémenterons le numéro de version (1.0, 2.0..).

# Gestion des modifications

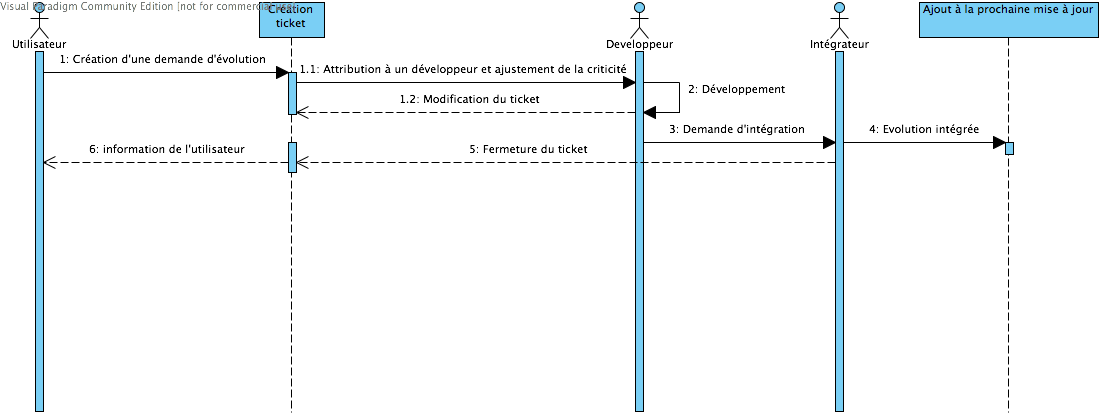
La gestion des configurations s'effectuera principalement avec la création de ticket Redmine et suivra ce processus :

Figure : Processus de gestion des configurations

L’outil SonarQube sera aussi très important dans la gestion des modifications. Quand celui-ci remontera une erreur ou bien une optimisation importante, un ticket sera crée sur le Redmine et ensuite attribué à un développeur. L’erreur ou l’optimisation sera donc automatiquement ajouté au Workflow du développeur.

# Méthodes, outils et règles

## Outils de pilotage

La base de pilotage du projet sera l’outil Redmine, solution flexible et open source de gestion de projets.

Cet outil permettra de visualiser le temps de travail, l’avancement et les objectifs de chaque ressource. De plus, il dispose d’outils permettant de prendre du recul sur le projet et de faire un point sur son état. (Typiquement la génération de Gantt).

Chaque membre du comité projet devra utiliser cet outil et tracer ses temps sur celui-ci.

## Méthode et outils de développement

### Conception

La conception se basera sur la méthode UML pour la partie Java.

### Standard de développement

Lors de la phase de réalisation du projet, nous utiliseront la convention de nommage CamelCase dans notre code. Cette convention consiste à écrire un ensemble de mots en mettant en majuscule la première lettre des mots liés. Elle convient particulièrement à un code JAVA, sensible à la casse dans la déclaration variable/classe.

|  |  |
| --- | --- |
| Variable | maVariable |
| Classe | MaClasse |
| Interface | IMonInterface |
| Package | monpackage |
| Méthode | maMethode |
| Constante | MA\_CONSTANTE |

Figure 5: Convention de nommage

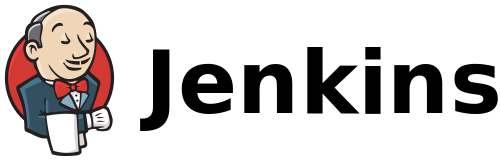
### Environnement de développement

L’environnement de développement sera basé sur un système d’exploitation Linux (debian).



Un serveur dédié (Fedora 20) loué pour l'occasion, hébergera de nombreux outils qui aideront à la réalisation de notre projet :

#### Jenkins:



Jenkins est un outil d’intégration continu développé en Java et basé sur un accès web par Apache Tomcat. Il nous servira de point de repère lors d’erreur à la compilation et sera un indicateur en cas de régression durant notre phase d’intégration.

#### SonarQube

Sonarqube-48x200.png

SonarQube (précédemment Sonar) est un logiciel de contrôle de qualité de code en continu. Il nous permettra d’identifier les points sensibles de notre code et apportera une aide à l’optimisation de celui-ci.

#### IDE

Nous avons choisis  d’utiliser l'environnement de développement Eclipse.



Il nous servira à écrire notre code ainsi que tous les tests unitaires. De plus Eclipse possède un plugin pour gérer directement le dépôt GIT, il nous évitera d’utiliser les lignes de commandes GIT qui sont très fastidieuses et complexes à comprendre. Nous avons tous déjà utilisé Eclipse c’est donc pour ces raisons que nous avons choisit de l’utiliser.

#### Outils de bureautique

Dans le cadre du développement de l’application, nous utiliseront la suite Office de Microsoft pour tous nos besoins de rédaction (Word) et présentation (Powerpoint).

La suite d’outil de bureautique de Google (docs essentiellement) sera aussi ponctuellement utilisée pour la réalisation de documents en collaboration.

Une formation à l’utilisation de Git et JUnit est prévu. Elle sera donnée par un membre de l’équipe maitrisant ces deux outils.

### Forme

Les documents devront être dupliqués et versionnés à la fois sur la plate-forme de gestion de version Git et sur le Redmine.

# Contrôle des fournisseurs.

Les fournisseurs sont une des principales ressources. Ainsi les engagements pris avec les fournisseurs peuvent impacter la qualité du produit de par la qualité des fournitures. Mais les conséquences principal principale des fournisseurs ce retrouvent sur les délais, et ceux-ci sont cruciale l'accomplissement du projet. Des délais non suivi résulteraient en un produit de mauvaise qualité ou non-fini.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Fournisseur | Rôle | Risque | Conséquence | CNQ | Prévention et contrôle |
| OVH, Kimsufi | Fournisseur de serveur dédié | Perte des données | Perte de la totalité de la production | 5x6jours/homme (totalité) | dépôt décentraliser et sauvegarde régulière toutes les semaines |
|  |  | Crash du système | Incapacité de développement temporairement | minimal | Dépôt décentralisé et sauvegarde régulière toutes les semaines |
| CPE | Fournisseur de matériel informatique | Crash du système | Perte du développement en cour | 5\*0,5jours/homme | Avoir accès à des ordinateurs personnels. |
|  |  | Incapacité à se connecter au serveur | Ralentissement du développement du projet | 5\*0,5jours/homme | Communication active avec M. Trouillot, Mme Perrin et la DSI. |

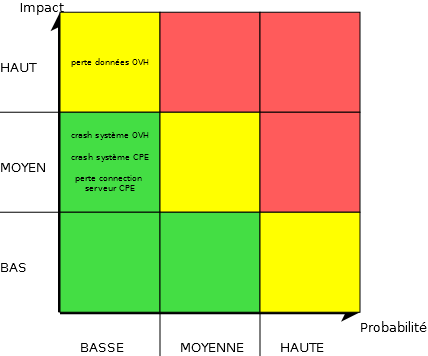


Figure 6: Evaluation des risques engendré par les fournisseurs.

# Reproduction, protection, livraison

## Reproduction, protection

Le Jeu “Attention pirates !” étant la propriété de “Jeux Ravensburger S.A.”, nous déclinons tous droits dessus. Ce logiciel n’est pas développé dans un but lucratif mis dans un but uniquement éducatif et académique, et ne sera pas reproduit. Nous déclinons toute responsabilité sur l’utilisation de ce logiciel en dehors du cadre des droits propriétaire et de la loi.

## Livraison, protection

Pour assurer la protection de notre développement la livraison se fera sous la forme d’un exécutable avec la documentation associée.

# Vérifications et Suivi du Plan Qualité

L'utilisation de ce PAQ doit permettre d'atteindre les objectifs suivants :

* Constituer une référence commune à tous les membres de l'équipe projet. Il permettra d'assurer une bonne cohérence et une homogénéité dans les méthodes de travail.
* Garantir la qualité du produit et des prestations. Cette qualité s'exprime par des critères à respecter dans le cadre de ce projet.  Ces critères sont détaillé tous au long de ce document.
* Définir les procédures à suivre, les outils à utiliser, la méthodologie de développement du produit et les contrôles prévus pour chaque activité.

Le contrôle qualité portera sur :

* Le produit : Pour cela nous vérifierons la conformité des documents dans leur phase finale.  Nous vérifierons également la conformité du développement tous au long de celui-ci grâce aux outils adapté Sonar et Jenkins (voir paragraphe Outils). Le contrôle portera sur le respect des protocoles et des standards et fera l’objet de la rédaction d’un cahier de recette.
* les processus : le contrôle des processus consiste à contrôler l’existence, la production et l’utilisation des fichiers, des méthodes et outils, et des règles de développement. Les dossiers et ces fiches seront alors annotés du nom du contrôleur qui tiendra une fiche récapitulative du contrôle de processus. Le chef de projet sera en charge du suivie de la méthode. Le responsable outils et de leurs MCO. Les règles de développement seront appliquées et vérifié par l'intégrité des développeurs.

Des jalons sous forme de revues informelles compléteront le suivi de l’application du plan qualité, ceux-ci se déroulerons en anticipation des intégrations sur la branche de développement principale et des réunions du comité.

## Résolution des non-conformités

Lorsqu’une anomalie ou un incident apparaît, les développeurs et responsables devrons ce référer à ce PAQ et suivront les recommandations de celui-ci. Toute anomalie devra être signalée au chef de projet. Si le PAQ n’apporte pas de solution, le chef de projet sera responsable de la résolution de l’incident.

La procédure suivante devra être appliquée:

* Identifier l’incident
* Identifier la cause
* Evaluer la portée et les conséquences
* Vérifier le PAQ
* Déclarer l’incident auprès du chef de projet
* Faire évoluer le plan qualité si besoin
* Valider celui-ci par le comité
* Répertorier l’incident dans le cahier de recette