Recommandations pour Rapport de fin d'études

Taha Ben Salah

June 24, 2017

Version	Responsable	Date	Description
0.6.1	TBS	2017/06/22	Mise à jour
0.6	TBS	2017/06/18	Corrections linguistiques
0.5	TBS	2013/06/08	Version Initiale

1 Structuration recommandée

1.1 Dédicaces

La dédicace concerne un remerciement personnel (à un ou plusieurs membres de la famille) à qui on dédie (on offre) le travail. On ne dédie pas à ses camarades donc.

Les dédicaces ne sont pas obligatoires.

1.2 Remerciements

Généralement on remercie tous ceux ou celles qui on pu contribuer de près ou de loin à la mise en oeuvre du projet (c'est purement non personnel). Exemple, l'école, le jury, les camarades de classe, les encadrants, etc... On ne remercie pas sa famille donc. Les remerciements sont obligatoires.

1.3 Résumé

1.3.1 Contenu

Une description brève contenant les élements suivants

- problématique
- réalisation

Un résumé ne doit pas dépasser les 100 mots.

Un résumé doit contenir les mots clés du systèmes (les concepts et les technos)

1.3.2 Disposition

- Sans numérotation
- Page orpheline

1.4 Abstract (Résumé en anglais)

1.4.1 Contenu

- Les mêmes idée en anglais.
- Eviter une traduction mot pour mot.

1.4.2 Disposition

- Sans numérotation
- Page orpheline, mais peut figuer sur la même page que le résumé en français

1.5 Résumé en arabe

1.5.1 Contenu

- Les mêmes idées en arabe
- Eviter une traduction mot pour mot

1.5.2 Disposition

- Sans numérotation
- Page orpheline, mais peut figuer sur la même page que le résumé en français

1.6 Introduction Générale

1.6.1 Contenu

- Commencer par introduire le contexte général du projet
- Eventuellement introduire la société (1 page max)
- Terminer par étaler la suite : plan du rapport (le 1er chapitre s'intéresse à ...)
- Ne contient pas ce qui va être fait mais plutôt le contexte/cadre du projet

1.6.2 Disposition

- 1 page (max deux pages)
- Sans titre (ou “Préambule”)

1.7 Table des matières

Utiliser le format standard automatique

1.8 Liste des figures

Utiliser le format standard automatique

1.9 Liste des tabeaux

Utiliser le format standard automatique

1.10 Introduction

1.10.1 Concepts Généraux

Reprendre plus en détails les concepts généraux du projet.
Exposer les technologies au sens large.

1.10.2 Problématique

Répondre à la question :
Pourquoi le projet a été posé

- maintenant
- par le client

1.11 Etude de l’existant et Etat de l’art

Le problème étant posé, chercher comment le client et les tiers ont répondu au même besoin/problème.

1.11.1 Etude de l’existant

Les solutions partielles ou globales dans l’entreprise (le client). Autrement dit, avant d’essayer de réaliser ce projet, comment l’entreprise gérait-elle son problème. Insister sur les inconvénients de la solution adoptée et ses limitations. On se limitera aux captures écran des applications existantes sans entrer dans leur technicité (conception, architecture, choix techniques)

1.11.2 Etat de l'art

Les solutions partielles ou globales par les tiers autrement dit les autres projets commerciaux ou libres qui répondent au probème (même en partie).

- Donner des captures écran de projets similaires.
- Collecter les caractéristiques fonctionnelles et technique
- Comparer et critiquer les solutions (idéalement celà se conclut par un tableau comparatif)
- NE PAS proposer une solution : il est uniquement question de détecter les éléments positifs et négatifs de ce qui existe

1.11.3 Formatisme

Présenter brievement les paradigmes et les formalismes adoptés

- Formalismes de description textuelle : Merise,UML,ODT, ...
- Processus de développement : Agile, Scrum, ...
- On parlera aussi ici du formalise de Retro-ingénierie si l'on traite un projet de rétro ingénierie

1.12 Rétro ingénierie

Ce chapite ne doit figurer que dans le cas de projets de reverse-engineering

1.12.1 Retro-Realisation

Discuter les choix techniques, Frameworks, etc... adoptés dans l'application existante

1.12.2 Retro-Conception

1.12.3 Architecture Physique

1.12.4 Architecture Logique

Modèles Conceptuels Discuter les choix conceptuels (design patterns, architectures, ...) adoptés dans l'application existante

Décomposition Modulaire Discuter des modules existants avec des diagrammes de packages (les modules) et de classes (si besoin) pour la partie statique. Au besoin on décriera les intéractions entre modules, dynamiquement, avec un diagramme de séquence

Retro-Specification des besoins Détecter les fonctionnalités existantes par une description brève en diagrammes de cas d'utilisation

1.13 Spécifications

Définition des besoins

- statique (use cases)
- dynamique (séquence, etat transition, activité, collaboration, ...)

Rappeler le problème et exposer le cahier des charges de façon semi-formelle (éviter les ambguités) souvent avec les cas d'utilisation au sens UML.

Remarque : Dans un docuement technique officiel (en industrie), TOUS les cas d'ulisation doivent être décrits. Mais dans un rapport de fin d'étude, uniquement les cas d'utilisation les plus pertinents seront traités (éviter de refaire des cas d'utilisation équivalents).

1.13.1 Définition des acteurs

Définir chacun des acteurs qui peuvent interagir avec la solution. Explicter les types d'acteurs :

- Acteur principal vs Acteur secondaire

  L'acteur principal : l'utilisateur. Celui pour qui le système a initialement été conçu.

  L'acteur secondaire : Les autres utilisateurs : l'administrateur, ...

- Acteur Humain vs Acteur Logique
  - L'acteur humain est une personne physique : le client, l'administrateur,....
  - L'acteur logique est un acteur matériel (périphériques ...), logiciel (serveur, une autre application), ou temporel (une échéance).
- Acteur actif vs Acteur passif
  - L'acteur actif utilise le système
  - L'acteur passif est utilisé par le système.
- Acteur externe vs Acteur interne
  - L'acteur externe ne fait pas partie du système, il l'utilise ou est invoqué par le système
  - L'acteur interne fait partie du système. Il décrit un échéance (Scheduler), ou un événement interne au système qui déclanche un Use Case

1.13.2 Système Cible

Commencer par le diagramme des cas d'utilisation général (en termes de fonctionnalités globales) qui reprend sans détails tous les acteurs et toutes les fonctions vues par chacun des acteur.

1.13.3 Fonctionnalité 1

Définir le cas d'utilisation

Dresser un diagramme de cas d'utilisation détaillé pour cette fonctionnalité. On ajoutea aussi un diagramme de séquance (ou tout autre diagramme dynamique) pour expliciter la dynamicité du système. Celà inclut souvant trois éléments : l'acteur principal, le systèmes et les acteurs externes (passifs)

Description Textuelle Une description explicative du cas d'utilisation

Cartouche de description Un tableau à deux colonnes sans titre

- Nom : Utiliser une tournure à l'infinitif (ex : Réceptionner un colis).
- Objectif : Une description résumée permettant de comprendre l'intention principale du cas d'utilisation. Cette partie est souvent renseignée au début du projet dans la phase de découvertedes cas d'utilisation.
- Acteurs principaux : Ceux qui vont réaliser le cas d'utilisation (la relation avec le cas d'utilisation est illustrée par le trait liant le cas d'utilisation et l'acteur dans un diagramme de cas d'utilisation)
- Acteurs secondaires : Ceux qui utilisent le systèmes dans des conditions particulières (remplacement de role, ...) et ceux qui ne font que recevoir des informations à l'issue de la réalisation du cas d'utilisation
- Dates : Les dates de créations et de mise à jour de la description courante.
- Responsable : Le nom des responsables.
- Version : Le numéro de version
- Les préconditions : elles décrivent dans quel état doit être le système (l'application) avant que ce cas d'utilisation puisse être déclenché.
- Des scénarii : Ces scénarii sont décrits sous la forme d'échanges d'évènements entre l'acteur et le système en une suite de logique d'actions.On distingue
  - le scénario nominal, qui se déroule quand il n'y a pas d'erreur
  - les scénarii alternatifs qui sont les variantes du scénario nominal
  - les scénarii d'exception qui décrivent les cas d'erreurs.
- Des postconditions : Elle décrivent l'état du système à l'issue des différents scénarii.

Exemple :

<table>
<tr><td>Intitulé</td><td>Authentification</td></tr>
<tr><td>Id</td><td>UC- 001</td></tr>
<tr><td>Révision</td><td>v1-TBS-2011-10-13</td></tr>
<tr><td>Auteur Principal</td><td>Opérateur</td></tr>
<tr><td>Acteurs Secondaires</td><td>N/A</td></tr>
<tr><td>Description</td><td>Ce cas d'utilisation décrit les étapes nécessaires pour pouvoir utiliser le système . Il s'agit une étape nécessaire pour tous les prochains cas d'utilisation.</td></tr>
<tr><td>Pré-conditions</td><td>L'application est installée sur le Mobile</td></tr>
<tr><td>Post-conditions</td><td>Une fois l'application lancée, l'opérateur est invité à entrer son login et son mot de passe puis il entre sur « enter ».</td></tr>
<tr><td>Scénario nominal</td><td>1- l'opérateur lance l'application
2- Il saisit son login
3- Il saisit son mot de passe
4- Il appuie sur le bouton “Se connecter”</td></tr>
<tr><td>Scénario Alternatif</td><td>Quand l'utilisateur utilise plus d'une fois l'application sur le même terminal, il peut ne pas mentionner son login (étape 2) qui est déjà mentionné (mais reste modifiable) car le système se rappelle du dernier login utilisé.</td></tr>
<tr><td>Scénario Alternatif2</td><td>L'utilisateur pourra appuyer sur “Entrer” au lieu de cliquer sur le bouton (étape 4)</td></tr>
<tr><td>Scénario d'exception</td><td>Si le login ou le mot de passe sont incorrects, le système rejette l'authentification avec un message de type « login ou mot de passe incorrect ». Aucun système de blocage n'est prévu si l'authentification échoue plusieurs fois de suites.</td></tr>
<tr><td>Contraintes d'intégrité</td><td><table><tr><th>Donnée</th><th>Type</th><th>Null</th><th>Précisions</th></tr><tr><td>login</td><td>alphanumérique</td><td>oui</td><td>sensible à la casse, longeur:3-20</td></tr><tr><td>mot de passe</td><td>alphanumérique</td><td>oui</td><td>sensible à la casse, longeur:3-20</td></tr></table></td></tr>
<tr><td>Contraintes logiques</td><td>Le login et mot de passe doivent correspondre à un user existant dans le système. Le login et mot de passe ne sont pas puisés depuis SAGE mais depuis une configuration locale.</td></tr>
<tr><td>Contraintes visuelles</td><td>La saisie du mot de passe n'est pas visible.</td></tr>
<tr><td>Spécifications particulières</td><td>Cette page est la page par défaut quand tout problème de sécurité survient</td></tr>
</table>

1.14 Conception

C'est la solution proprement dite.

1.14.1 Architecture physique

Dresser un diagramme de déploiement qui décrit les composante matérielles qui interviennent (les noeuds).

Il est d'usage d'incorposer les composants logiciels dans le diagramme de déploiement aussi.

Le niveau le plus fin serait un “fichier” de déploiement. Par exemple une DLL, un WAR, un EAR.

Le diagramme contient aussi les OS et les serveurs applicatifs (serveur de base de données,serveur d'application) sans oublier les protocoles/frameworks de communication (TCP/IP, JDBC/OLEDB) ni les versions.

1.14.2 Architecture logique

Ici on reprend tous les composants un à un pour les décrire dans le context d'un choix conceptuel bien défini (3tier, MVC, pyramide, etc...)

Modèles conceptuels Enumération des designs patterns utilisés.

Décomposition Modulaire Diagramme de packages qui décrit la décomposition verticale (en terme de modules applicatifs) et horizontale (en terme de couches 3 tiers généralement) .

Un module applicatif est une partie de l'applicaiton décrite par

- une cohésion forte : cohérence et consistance de la décomposition (ne pas mettre des éléments qui qui peuvent évoluer séparément)
- un cloisonnement fort : toute dépendence est bien spécifiée
- un couplage faible : une dépendance faible (sans dépendance ou dépendance mesurable)

Le rapport doit rester compréhensible meme si l'on enleve tous les schémas et les tableaux. Souvent les paragaraphes associés aux schémas décrivent le contenu du schéma en annotant/référençant la figure en question.

1.14.3 Module 1

Modèle de présentation On mettra ici les diagrammes des classes, de sequences ou de collaboration de la couche présentation. Généralement on verrait les classes du modèle MVC si utilisé.

Notons que le diagramme de classe pourra bien représenter les fichiers ressouces de présentations aussi (xhtml, etc...) meme si elles ne sont pas des classes. On rajoutera dans ce cas le stériotype <<view>> sur la classe associée.

Le diagramme de classe présenterait aussi la navigabilité d'un écran à un autre.

Modèle de traitement On mettra ici les diagrammes de classes, de sequences ou de collaboration de la couche métier. Ceci contient les BO (Business Object), EJB, COM, les composants Corba...

Modèle de données On mettra ici les diagrammes de classes, principalement qui décrivent le modèle de persitence ainso que les classes d'accès aux données au besoin (DAO), ou pourrait aussi y retrouver le diagramme de classe des DTO. Souvent on mettrait deux diagrammes de classes un pour les Entités et un autre pour les DAO et les les DTO (en mettant les bon stériotypes pour les séparer).On pourrait envisager de mettre les DTO dans un diagramme séparé si l'on sent le besoin.

Dans certains cas, il n'est pas nécessaire de décrire le diagramme d'accès aux données (DAO) car par exemple, dans le cas d'utilisation de la technologie JPA, ou .Net Entity, la couche DAL est déjà implémentée par le framework et ne sera pas incluse dans les diagramme.

1.14.4 Module N

Suivre la même logique que pour le Module 1

1.15 Réalisation

1.15.1 Choix techniques

Paradigmes Logiciels Introduire si applicable le paradigme utilisé : Programmation orientée Objet/ Programmation fonctionnelle, orientée aspect, ...

On pourra aussi ici parler de l'intérêt d'utiliser un ORM (sans le nommer) par exemple etc...

Frameworks de programmations Présenter ici les Frameworks utilisés (en les nommants) après une discussion des frameworks existants et une explicitation des choix et contraintes d'applications.

Serveurs d'application Présenter ici les Serveurs d'application, de base de données, etc... utilisés après une discussion des serveurs existants

1.15.2 Problèmes techniques

Exposer les problèmes techniques rencontrés et comment il auraient été dépassés.

1.15.3 Captures écran

Quelques captures d'écran décrivant l'application.

1.15.4 Tests et validation

Parler ici de l'intérêt des plans de tests et de validation

Environment de Test Décrire l'environnement de test en le comparant à l'environnement de déploiement (on décrira un nouveau diagramme de déploiement)

Tests Unitaires Discuter des choix du framwork et outils de test unitaires utilisés (JUnit, NUnit,...) . Décrire un ou deux scénarios de tests en précisant les résultats (généralement positifs) de leur application

Tests d'intégration Discuter kes choix du serveur d'integration continue (CI : Jenkins, Hudson, ...) si applicable. Discuter de la portée des tests des Tests d'integration utilisés (integration entre le module et l'application déjà existante, integration avec un serveur LDAP, etc...) .

Tests de couverture Appliquer un outil de test de couverture et annoncer les résultats (JCov, JaCoCo,...)

Tests de performances Appliquer un outil de stress test et annoncer les résultats (JMeter, HP Load Runner,...)

1.16 Conclusion

1.16.1 Résumé

Résumé de ce qui a été fait , ne pas y inclure des phrase de type "j'ai appris des choses intéressantes ....". C'est souvent une pagent avec le résumé de chaque chapitre précédant.

1.16.2 Perspectives

Proposer des perspectives. Une perspective est une suite de votre travail que vous savez exatement comment faire mais que vous n'avez pas eu le temps de faire. Il ne s'agit pas d'unetache non terminée, mais d'une nouvelle tache que vous auriez aimé rajouter.

Eviter des phrases bateau de type "l'application peut être amiéliorée"

1.17 Références

1.17.1 Bibliographie

Utiliser la forme standard

1.17.2 Netographie

Utiliser la forme standard

1.18 Annexes

Les annexes doivent re référencés dans le texte.

1.18.1 Annexe A

1.18.2 Annexe B ...

2 Check List

2.1 Mise en page

- Les paragraphes doivent être en texte justifiés
- Vérifier l'ortographe
- Entete et pied de page avec numéro de page.
- Définir une conclusion au terme de chaque chapitre
- Il ne faut jamais dépasser les 4 niveaux de numérotation (pas de 1.2.2.3.6 par exemple)
- Un titre de paragraphe ne doit jamais etre "Diagramme de ...."
- Ne jamais mettre un espace avant une ponctuation. Toujours rajouter l'espace après la ponctuation.

2.2 Langue et expression

- Eviter les phrases longues
- Eviter les phrases orphelines
- Un patagraphe >= 2 phrases
- Ne jamais utiliser le 'on' et éviter le 'je' et le 'nous'. Utiliser les formes passives sinon utiliser 'je' quand vous voulez insister sur un travail personnel, et 'nous' quand vous voulez insister sur un travail de groupe.
- Eviter l'espacement superflu
- Ne jamais mettre du code source dans le rapport. Au pire le mettre dans l'annexe et le référencer dans le corps du texte.

2.3 Harmonie

Le volume des chapitres doit être homogène/équivalent. Si un chapitre devient trop court le concaténer à un autre chapitre. S'il est trop grop le diviser en deux;

2.4 Références

- Aucun tableau, ni image ou figure ne doit être non référencé par un label. Chacun devrait aussi être explicitement référence dans le texte (ex : bla bla comme le montre la figure [X].)
- Redessiner personellement les images simples
- Eviter de mettre des figures non personnelles, au besoin ajouter aussi la référence au site source.
- Ancun texte ne doit être repris telquel d'un site sauf citation de définition. Dans le cas d'une citation, la mettre entre guillement et suffixé par une référence [X].
- Dans le cas d'un text repris avec modification il faudra aussi ajouter la référence [X].
- NE JAMAIS PLAGIER un rapport, ou tout autre document. s'il s'agit d'une citation, la mettre entre deux quotes et rajouter une référence à l'auteur.
- Toutes les références de la table des références doivent être utilisée dans le rapport
- Wikipédia n'est pas une référence. Les sources de références sont les documents et rapport dont l'auteur n'est pas anonyme et reconnu pour son expertise dans le domaine.

2.5 Outils

Utiliser un outil de modélisation pour les diagrammes UML (dia, argouml,...)