École de technologie supérieure

Département de génie logiciel et des TI

Cours : LOG210 - Analyse et conception de logiciels

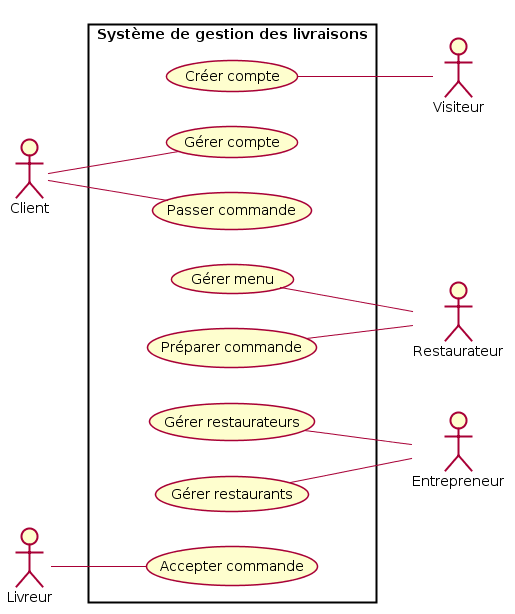
Date de création : 2014-04-25

Date de la dernière modification : 2015-05-06

# Étude de cas : Système de gestion des livraisons

Un entrepreneur veut offrir un service de commande et de livraison aux clients qui désirent commander de la nourriture auprès d'un restaurant n'offrant pas de service de livraison. Un service de livraison indépendant est proposé où les livreurs récupèrent du restaurant ce que les clients ont commandé, puis ils le leur livrent.

Le diagramme de cas d'utilisation suivant (Figure 1) illustre les besoins de l'entrepreneur.

  
**Figure 1**

L’entrepreneur a beaucoup d’exigences et sait de toute évidence que les développeurs ne peuvent pas toutes les réaliser à cause des contraintes de budget et de temps. Il vous a donc préparé une liste d’exigences en incluant son évaluation de l’impact que ces divers facteurs peuvent avoir sur son application. Il se fie au bon jugement et à l’expérience des membres de votre équipe pour organiser chacune de vos itérations de telle sorte que le projet avance le plus, malgré tous les risques. C’est la force d’une approche itérative et évolutive, après tout.

Le rendement au niveau des laboratoires sera calculé à 50% sur l’approche itérative et évolutive, correspondant aux activités d’analyse et de conception selon le Processus Unifié, alors que le défi technologique ou, si vous préférez, la satisfaction du client correspondra à l’autre 50%.

Votre équipe a donc l’entière responsabilité de choisir soigneusement les exigences que vous jugez réalisables dans une itération. N’oubliez pas de tenir compte des risques et de l’expérience de votre équipe dans vos choix. Toutes exigences non satisfaites durant une itération pénaliseront la note de l’équipe. Veuillez toutefois noter que cette pénalité sera moindre si l’exigence est quand même satisfaite à la fin du projet.

Prenez note que l'entrepreneur est ouvert à toute suggestion, de la part des ingénieurs en formation de votre entreprise, qui permettraient d’améliorer la qualité du produit ou du service. N’hésitez pas à en discuter avec le représentant de l’entrepreneur (Chargé de laboratoire) qui pourra évaluer votre proposition et en faire part à l’entrepreneur (Professeur ou chargé de cours) si celles-ci s’avèrent intéressantes pour le projet.

## Conseils pour tous les membres de votre équipe:

Puisque les choix technologiques faits au début du projet peuvent avoir un impact majeur sur toute la session, il faut bien réfléchir à ces choix.

Par exemple, deux personnes dans l’équipe sont très motivées pour utiliser PHP pour la solution, mais deux autres personnes ne connaissent pas cette technologie et ne sont pas trop motivées pour l’apprendre. Si l’équipe décide d’aller avec un choix comme celui-là, il y a un risque de conflit plus tard dans la session lors de la livraison des fonctionnalités. Ceux qui connaissent moins une technologie auront une mise à niveau à faire et vont faire plus d’erreurs, voire être presque inutile dans le projet. Même s’ils sont motivés pour l’apprendre, ils seront moins performants que les autres. Certains types d’étudiants aiment ce genre de défi malgré l’effort supplémentaire, d’autres se contentent de faire un travail adéquat moins risqué.

Le choix de l’équipe doit être un **consensus**. Vous devez considérer tous ces aspects (expérience, effort, disponibilité, etc. de chaque membre) en tant qu’équipe et non en tant que plusieurs membres individuels.

Considérez l’approche suivante: le choix avec le moins de risque (et qui peut avoir une note raisonnable) est de faire tout en Java avec les technologies simples. Commencez avec cette idée de base et élargissez les objectifs un par un, selon la disponibilité, la motivation et les connaissances de l’équipe **ensemble**.

# Grille des exigences FURPS+

|  |  |  |
| --- | --- | --- |
| Points | Facteur | Exigence |
| 10 |  | F1 – Gérer les restaurants |
| 5 |  | F2 – Créer un compte |
| 5 |  | F3 - Gérer un compte |
| 15 |  | F4 – Passer une commande |
| 10 |  | F5 – Gérer les menus |
| 10 |  | F6 – Préparer une commande |
| 15 |  | F7 – Accepter une commande |
| 10 |  | F8 – Gérer les restaurateurs |
|  | 1.025 | U1 – Conception de sites web adaptatifs |
|  | 1.025 | U2 - Ajax |
|  | 1.025 | U3 – Internationalisation |
| 2.5 |  | P1 – Rapport de temps de réponse |
|  | 1.025 | P2 – Utilisation de cache local |
|  | 1.025 | P3 – Utilisation d’un serveur de type Node.js |
|  | 1.000 | S1 – Prise en charge web |
|  | 1.025 | S2 – Prise en charge REST |
|  | 1.300 | S3 – Prise en charge mobile (Application native) |
|  | 0.750 | S4 – Aucune prise en charge mobile ou web |
|  | 1.025 | S5 – Persistance avec Hibernate |
|  | 1.025 | S6 – Persistance avec MongoDB |
|  | 1.000 | S7 – Persistance avec MySql, Derby, Postgres |
|  | 1.000 | S8 – Persistance avec une base de données XML |
|  | 0.500 | S9 – Persistance dans des fichiers texte |
|  | 1.050 | Im1(Implémentation) – environnement de test |
| 5 |  | In1 (Interface) – Changement d’état par SMS |
| 10 |  | In2 (Interface) – Paiement par PayPal |
| 5 |  | I1 – Utilisation de Patron GOF |
|  | 1.025 | E1 – Déploiement sur un serveur externe |

F – Fonctionnalité, U – Convivialité, R – Fiabilité, P – Performance, S – Possibilité de prise en charge, +I – Implémentation, +I - Interface, +E – Exploitation, +C – Conditionnement, +A – Aspect juridique

Évaluation: Pour le calcul de la note, voir la section Évaluation.

# Exigences fonctionnelles

## F1 – Cas d’utilisation – Gérer les restaurants

Pré condition(s) :

* L’entrepreneur est authentifié

Post conditions(s) :

* Un restaurant a été ajouté dans le système

Acteur principal :

* Entrepreneur

Scénario principal

1. L’entrepreneur entame l’ajout d’un restaurant.
2. L’entrepreneur fournit l’information requise pour l’ajout d’un restaurant.
3. L’entrepreneur assigne un restaurateur au restaurant et confirme l’ajout.
4. Le système enregistre le restaurant et son restaurateur.

Les étapes 1 à 4 sont répétées tant que l’entrepreneur désire ajouter des restaurants et leur affecter un restaurateur dans le système.

Scénarios alternatifs

1a. L’entrepreneur désire supprimer un restaurant

1. L’entrepreneur entame la suppression d’un restaurant
2. Le système supprime le restaurant

1b. L’entrepreneur désire modifier les informations d’un restaurant

1. L’entrepreneur entame la modification d’un restaurant
2. L’entrepreneur modifie les informations.
3. Les informations sont enregistrées dans le système.

3a. L’entrepreneur n’assigne pas de restaurateur

1. Le système enregistre le restaurant sans restaurateur et avertit l’entrepreneur.
2. Retour à 4.

## F2 – Cas d’utilisation – Créer un compte

Pré condition(s) :

* Aucune

Post condition(s) :

* Un compte client a été créé

Objectifs des parties prenantes :

* Visiteur : désire avoir un compte client pour pouvoir passer des commandes

Acteur principal :

* Visiteur

Scénario principal

1. Le visiteur entame la création d’un compte client
2. Le visiteur saisit son nom complet, sa date de naissance, son adresse complète, son numéro de téléphone, son courriel[[1]](#footnote-1) et le mot de passe2 qu’il désire avoir.
3. Le visiteur confirme la création du compte client
4. Le système enregistre le nouveau compte client et affiche un message de confirmation avec l’information de celui-ci.

## F3 – Cas d’utilisation – Gérer un compte

Pré condition(s) :

* Le client est authentifié

Post condition(s) :

* Les informations du client ont été modifiées

Objectifs des parties prenantes :

* Client : désire avoir le plein contrôle sur les informations retrouvées sur son compte.

Acteur principal :

* Client

Scénario principal

1. Le client entame la modification des informations de son compte
2. Le client modifie son adresse complète, son numéro de téléphone ou son mot de passe.
3. Le client confirme les modifications.
4. Le système enregistre les modifications et affiche un message de confirmation avec les nouvelles informations enregistrées.

## F4 – Cas d’utilisation – Passer une commande

Pré condition(s) :

* Le client est authentifié.

Post conditions(s) :

* Une commande a été ajoutée dans le carnet des commandes.

Objectifs des parties prenantes :

* Client : désire se faire livrer la nourriture qu’il a commandée.
* Restaurateur : désire planifier la préparation des commandes pour en assurer la livraison.

Acteur principal :

* Client

Scénario principal

1. Le client commence une nouvelle commande de nourriture.
2. Le système présente les restaurants participants.
3. Le client sélectionne un restaurant.
4. Le système affiche le menu du restaurant.
5. Le client sélectionne un repas et entre la quantité désirée.
6. Le client indique que la commande est complète.
7. Le système affiche le sommaire de la commande.
8. Le client entre la date et l’heure de livraison et confirme la commande.
9. Le système enregistre la commande et affiche un numéro de confirmation.
10. Le système envoie un courriel de confirmation au client avec les informations sur la commande

*L’étape 5 est répétée tant que le client désire commander d’autres repas.*

Scénarios alternatifs

8a. La commande doit être livrée à une nouvelle adresse du client

1. Le client entre une nouvelle adresse de livraison.
2. Le système remplace l’adresse de livraison par défaut.
3. Retour à l’étape 8.

8b. La commande doit être livrée à une adresse prédéfinie par le client

1. Le client choisit l’adresse de livraison dans sa liste d’adresse possible.
2. Le système remplace l’adresse de livraison par défaut.
3. Retour à l’étape 8.

## F5 – Cas d’utilisation – Gérer les menus

Pré condition(s) :

* Le restaurateur est authentifié
* Le restaurateur est associé à un restaurant

Post condition(s) :

* Un menu a été défini

Objectifs des parties prenantes :

* Restaurateur : désire offrir des menus aux clients
* Client : désire connaître la nourriture offerte par un restaurant

Acteur principal :

* Restaurateur

Scénario principal

1. Le restaurateur entame la définition d’un menu pour un restaurant.
2. Le restaurateur saisit un nom au menu.
3. Le restaurateur ajoute un plat à son menu.
   1. Le restaurateur définit un nom au plat.
   2. Le restaurateur définit une description au plat.
   3. Le restaurateur définit un prix au plat.
   4. Le restaurateur confirme l’ajout du plat
4. Le restaurateur confirme le nouveau menu
5. Le système enregistre le menu.

*L’étape 3 est répétée tant que le restaurateur désire ajouter un plat dans un menu.*

*Les étapes 2 à 4 sont répétées tant que le restaurateur désire définir des menus dans le système.*

Scénarios alternatifs

2a. Le restaurateur n’assigne pas de nom au menu

1. Le restaurateur ajoute un plat à son menu
2. Le restaurateur n’assigne pas de nom au menu et confirme l’ajout
3. Le système avertit le restaurateur que le nom du menu est obligatoire et empêche l’ajout du menu

3a. Le restaurateur désire ajouter un plat sans assigner une description

1. Le restaurateur ajoute un plat à son menu
2. Le restaurateur définit un nom et un prix pour le plat
3. Le restaurateur confirme l’ajout du plat
4. Le système enregistre le menu et avertit le restaurateur qu’aucune description n’a été fournie

3b. Le restaurateur ajoute un plat sans assigner un nom et/ou un prix

1. Le restaurateur ajoute un plat à son menu
2. Le restaurateur ne définit pas de nom et/ou de prix
3. Le restaurateur définit une description
4. Le restaurateur confirme l’ajout du plat

Le système avertit le restaurateur que le nom et le prix sont obligatoires et empêche l’ajout du plat.

## F6 – Cas d’utilisation – Préparer une commande

Pré condition(s) :

* Le restaurateur est authentifié
* Une commande a été passée chez un restaurant associé au restaurateur

Post condition(s) :

* Une commande est prête

Objectifs des parties prenantes :

* Restaurateur : désire planifier les commandes et la préparation de la nourriture afin d’assurer la fraîcheur des plats.
* Livreur : désire connaître les commandes prêtes à être livré

Acteur principal :

* Restaurateur

Scénario principal

1. Le restaurateur entame la préparation d’une commande.
2. Le restaurateur choisit la commande à préparer.
3. Le système met à jour le statut de la commande à « En préparation ».
4. Le restaurateur termine la préparation de la commande.
5. Le système met à jour le statut de la commande à « Prête ».

*Les étapes 2 à 5 sont répétées tant que le restaurateur désire préparer une commande.*

## F7 – Cas d’utilisation – Accepter une commande

Pré condition(s) :

* Le livreur est authentifié.

Post condition(s) :

* Une commande a été inscrite au carnet des livraisons.

Objectifs des parties prenantes:

* Livreur : désire optimiser ses déplacements en sélectionnant les commandes prêtes les plus proches de son emplacement courant.
* Restaurateur : cherche à minimiser la durée de temps où une commande prête attend d’être livrée.

Acteur principal :

* Livreur

Scénario principal

1. Le livreur entame une demande de livraison et entre son emplacement actuel.
2. Le système affiche l’ensemble des commandes actuellement prêtes.
3. Le livreur sélectionne une commande.
4. Le système communique avec un système cartographique puis affiche le trajet optimal pour récupérer la commande et la livrer.
5. Le livreur accepte de livrer la commande.
6. Le système inscrit la commande au carnet des livraisons à la date (heure) actuelle.

*Les étapes 3 et 4 sont répétées tant que le livreur n’accepte pas la commande sélectionnée.*

Scénarios alternatifs

5a. Un autre livreur a déjà accepté la commande sélectionnée.

1. Le système affiche un message d’information.
2. Retour à l’étape 2.

## F8 – Cas d’utilisation – Gérer les restaurateurs

Précondition(s) :

* L’entrepreneur est authentifié

Postconditions(s) :

* Un restaurateur a été ajouté dans le système

Acteur principal :

* Entrepreneur

Scénario principal

1. L’entrepreneur entame l’ajout d’un restaurateur.
2. L’entrepreneur fournit les informations requises pour l’ajout d’un restaurateur.
3. Le système inscrit le restaurateur.

Les étapes 1 à 3 sont répétées tant que l’entrepreneur désire ajouter des restaurants et restaurateurs dans le système.

Scénarios alternatifs

1a. L’entrepreneur désire supprimer un restaurateur

1. L’entrepreneur entame la suppression d’un restaurateur
2. Le système supprime le restaurateur

1b. L’entrepreneur désire modifier les informations d’un restaurateur

1. L’entrepreneur entame la modification d’un restaurateur
2. L’entrepreneur modifie les informations.
3. Les informations sont enregistrées dans le système.

2a. L’entrepreneur n’assigne pas de restaurant(s)

1. Le système inscrit le restaurateur sans lui assigner de restaurant et avertit l’entrepreneur.
2. Retour à 3.

# Convivialité

## U1 – Conception de sites web adaptatifs

(Nécessite S1) L'entrepreneur veut que l’application soit facilement utilisable sous plusieurs formats de résolution (moniteur 4 :3, 16 :9, iPad, téléphone intelligent). On demande le support d’au moins 2 formats qui peuvent être sur la même plateforme.

Voir <http://fr.wikipedia.org/wiki/Site_web_adaptatif>

## U2 - Ajax

L'entrepreneur aimerait avoir une application fluide en utilisant des technologies évitant un rafraîchissement complet de la page.

Voir <http://fr.wikipedia.org/wiki/Ajax_(informatique)>

## U3 – Internationalisation

L'entrepreneur offre déjà des services dans différents pays. Il demande donc que le système puisse supporter plusieurs langues. Vous devez traduire votre système en au moins une langue supplémentaire et les utilisateurs doivent pouvoir changer celle-ci lors de l’utilisation de l’application. Le texte pour la langue doit être séparé du code source.

# Fiabilité (Reliability)

## R1 – N/A

# Performances

## P1 – Rapport de performance

L'entrepreneur aimerait que l’application ait une bonne rapidité de réaction. Chaque action ne devrait pas prendre plus de deux secondes pour compléter. L'entrepreneur s’attend à un rapport contenant les temps de réponse de chaque fonction du système.

## P2 – Utilisation de cache local

Amélioration de la performance du système en utilisant le processus de mise en cache local selon la section 30.2 du livre de Larman. UML 2 et les designs patterns. Vous devez être en mesure de démontrer la différence de performance pour une requête qui se rend jusqu’au serveur et une requête qui est optimisée grâce à la mise en cache.

## P3 – Utilisation d’un serveur de type Node.js

L'entrepreneur aura besoin d’un serveur ayant une très bonne performance pour les cas où il y a beaucoup de requêtes plutôt légères sur le serveur. L’utilisation de Node.js au niveau du serveur pourrait permettre d’augmenter la performance dans ces cas. Il faut cependant calculer l’impact que cela peut avoir sur le développement des API de la couche-domaine. Il faut absolument que la couche-domaine soit développée en conception orientée objet.

Node.js est plus performant que des serveurs traditionnels lorsqu'il s'agit de traiter beaucoup de requêtes légères. Par exemple, Node.js pourrait accélérer les choses lorsqu'un Client construit sa commande et qu’il navigue les menus pour les plats et qu’il y a beaucoup de requêtes AJAX provenant de beaucoup de clients en même temps. Mais Node.js ne serait pas mieux pour beaucoup de requêtes lourdes, p.ex. Gérer restaurateurs/restaurants/compte/etc.

<http://java.dzone.com/articles/performance-comparison-between>

# Possibilité de prise en charge

## S1 - Prise en charge web

L'entrepreneur aimerait que son application soit utilisable avec un fureteur aux fins d'accessibilité."

## S2 – Prise en charge REST

Pour supporter la réutilisation de modules (p.ex. les commandes), il aimerait que son application soit utilisable au travers d’un API REST. Ainsi, un tiers parti pourra inclure le module de commande client dans son propre site web.

Voir <http://fr.wikipedia.org/wiki/Representational_State_Transfer>

## S3 – Prise en charge mobile (Application native)

Pour cueillir des informations sur le comportement des consommateurs ainsi que la géolocalisation des livreurs, l’entrepreneur aimerait que son application soit utilisable au travers d’un appareil mobile. Certaines fonctionnalités de l’application doivent être disponibles, même si la connexion réseau ne fonctionne pas.

## S4 – Aucune prise en charge web ou mobile

L'entrepreneur a absolument besoin d’une application accessible à distance. Si vous faites une version sans accès à distance, il y a une pénalité.

## S5 – Persistance avec Hibernate

L'entrepreneur prévoit un changement d’infrastructure dans un ou deux ans et la solution ne devrait pas trop dépendre du type de base de données. Utilisez Hibernate pour persister vos données. Cette exigence nécessite l’exigence S7.

## S6 – Persistance avec MongoDB

Utilisez une base de données orientée documents pour persister les données de vos clients. Voir [http://fr.wikipedia.org/wiki/Base\_de\_données\_orientée\_documents](http://fr.wikipedia.org/wiki/Base_de_donn%C3%A9es_orient%C3%A9e_documents)

## S7 – Persistance avec MySql, Derby, Postgres

Utiliser une banque de données relationnelle pour sauvegarder les données.

## S8 – Persistance avec une base de données en XML

Utiliser une base de données XML pour sauvegarder vos données

## S9 – Persistance dans des fichiers texte

Réalisez vous-même la sauvegarde et l’extraction des données à partir de fichiers texte.

# Implémentation

## Im1 - Environnement de test

Un des aspects très importants de l’implémentation est la robustesse du code. Vous devez vous assurer que le code est robuste et le reste tout au long du processus itératif de développement. Une méthode pour y arriver consiste à implanter un environnement de test. Vous devez donc démontrer une progression des tests couvrant la couche logique applicative, et ce à chaque itération.

# Interface utilisateur

## In1 – Changement d’état par SMS

Désirant offrir ses services à une jeune clientèle techno, notre entrepreneur aimerait supporter la notification par SMS de telle sorte que les consommateurs soient rapidement informés de l’état de leur commande.

## In2 – Paiement par PayPal

Dans le but d’accélérer et de faciliter le travail des livreurs, l'entrepreneur aimerait que le consommateur puisse effectuer son paiement par PayPal.

Voir [https://developer.paypal.com/webapps/developer/docs/api/ - api-endpoints](https://developer.paypal.com/webapps/developer/docs/api/#api-endpoints)

Voir <https://developer.paypal.com/webapps/developer/docs/api/>

# Exploitation

## E1 – Déploiement sur un serveur externe

L'entrepreneur désire s’assurer que vous êtes en mesure de déployer votre application sur un serveur externe. Ceci permettra de la rassurer quant au fait que le serveur pourra être porté sur une plateforme infonuagique dans la seconde phase du projet.

# Évaluation

Le travail de laboratoire sera évalué en deux volets, soit la partie processus et la partie implémentation. Cette dernière sera évaluée vers la fin de chaque itération (*e1*, *e2*, *e3*) mais ce n’est que l’évaluation (*e*) de l’implémentation à la dernière itération qui comptera pour la note.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Évaluation | Itération 1 | Itération 2 | Itération 3 | Itération 4 | Total |
| Processus | *a* | *b* | *c* | *d* | (*a*+*b*+*c*+*d*) |
| Implémen- tation | *e1* | *e2* | *e3* | *e* | *e* |

L’équipe remet les artéfacts pour chaque itération selon le tableau suivant :

|  |  |  |
| --- | --- | --- |
| Itération | Minimum des points d’exigences fonctionnelles[[2]](#footnote-2) | Artéfacts à remettre |
| 1 | 10 | Plan d’itération[[3]](#footnote-3) Modèle du domaine (MDD)[[4]](#footnote-4) Diagrammes de séquence système (DSS)[[5]](#footnote-5)  Contrats d’opération5  Diagramme de séquence pour chaque opération du système[[6]](#footnote-6) |
| 2 | 20 | Plan d’itération3 Modèle du domaine (MDD)4  Diagrammes de séquence système (DSS)5 Contrats d’opération5  Diagramme de séquence pour chaque opération du système6 |
| 3 | 30 | Plan d’itération3 Modèle du domaine (MDD)4  Diagrammes de séquence système (DSS)5 Contrats d’opération5 Réalisation de cas d’utilisation (RDCU)5 |
| 4 | 40 | Plan d’itération3 Modèle du domaine (MDD)4  Diagrammes de séquence système (DSS)5 Contrats d’opération5 Réalisation de cas d’utilisation (RDCU)5 |

*Par exemple, pour l’itération 1, une équipe peut viser les fonctionnalités F2 et F3 (10 points) et ça respectera le minimum. Il y aura un MDD pour l’ensemble des cas d’utilisation F2 et F3, au moins un DSS pour F2 et au moins un DSS pour F3. Il y aura les diagrammes de séquence pour esquisser chacune des opérations systèmes des DSS.   
  
Pour l’itération 2, l’équipe vise F1 et F8 (nouvelles fonctionnalités) ajoutant 25 points pour un total de 35. L’ensemble des artéfacts à remettre comprendra un MDD pour F1, F2, F3 et F8, au moins un DSS pour chaque fonctionnalité (F1, F2, F3, F8), et les diagrammes de séquence respectant les postconditions des contrats, etc.*

*À l’itération 3, l’équipe doit remettre les RDCU’s correspondant aux fonctionnalités des itérations 1 à 3. On peut commencer avec les diagrammes de séquence existants, et y appliquer les principes GRASP. Veuillez noter que les chargés de laboratoire porteront une attention particulière au réusinage que vous aurez fait au code pour qu’il corresponde aux RDCU.*

Critères d’évaluation de la partie processus de chaque itération (*a*, *b*, *c*, *d*)

|  |  |
| --- | --- |
| **Critère** | **Points** |
| Qualité des artéfacts (processus d’analyse et de conception) | 10 |
| Minimum des points pas respecté (processus itératif) | -5 |
| Aucune nouvelle fonctionnalité démontrable (processus évolutif) | -5 |

Le calcul de la note pour la satisfaction du client est le suivant :

e = ∑(Points X (1 + (Facteuri-1)) / 100

*Par exemple, une équipe réalise une solution qui répond adéquatement aux exigences suivantes :*

*F1, F2, F3, F5, F6, U1, U2, S3*

*Le calcul est donc (10 + 5 + 5 + 10 + 10) \* (1 + (0.,05 + 0.05 + 0.3)) =56/100*

Si une équipe ne réussit pas à répondre adéquatement à une exigence, elle ne sera pas utilisée dans le calcul.

Pour les notes dépassant 100%, le maximum est 120%. Bon laboratoire!

1. Paramètres utilisés pour l’authentification de l’usager. [↑](#footnote-ref-1)
2. Points du tableau des exigences. [↑](#footnote-ref-2)
3. Le plan d’itération est remis et vérifié avant l’itération, sauf pour la première où il est remis au début du second laboratoire. [↑](#footnote-ref-3)
4. Le modèle du domaine doit être cohérent pour tous les cas d’utilisation depuis le début de la première itération. Voir la figure 2.4 pour plus d’informations. [↑](#footnote-ref-4)
5. Il doit y avoir un DSS (et un contrat d’opération et une RDCU, le cas échéant) pour chaque cas d’utilisation analysé depuis le début de la première itération. [↑](#footnote-ref-5)
6. Selon la notation du chapitre F14/A15 du livre de Larman. Dans l’itération 2, les postconditions du contrat doivent être satisfaites par le diagramme. [↑](#footnote-ref-6)