projet Programmation Système

Mise à jour du sujet et des attentes

09/12/2018

Préambule

Ce document est une mise à jour du sujet du projet programmation système. En effet il s’avère que le travail demandé était trop important pour être convenablement réalisé dans les temps. Cette modification impacte à la fois le contenu des livrables ainsi que la forme de l’évaluation. Cependant l’ensemble des informations présentes dans le premier document reste valide. Aussi, toutes les indications du chapitre 1 au chapitre 6 inclus sont inchangées.

Pour plus de clarté, l’ensemble des chapitres 7 et 8 ont été repris. Les éléments supprimés sont ~~barrés~~ et les éléments ajoutés ou modifiés sont soulignés.

Sommaire

[7 Cahier des charges et contraintes 2](#_Toc532154709)

[Fonctionnel 2](#_Toc532154710)

[Technique 3](#_Toc532154711)

[8 Organisation du projet et livrables attendus 4](#_Toc532154712)

[9 Grille d’évaluation 6](#_Toc532154713)

# Cahier des charges et contraintes

## Fonctionnel

* Diagrammes de cas d’utilisation
* Diagrammes d’activité de chaque poste de travail.
* ~~Simulation graphique du fonctionnement du restaurant en temps réel. Pour les besoins de la démonstration (et des tests), nous vous demandons d’avoir un mode simulation en temps accéléré à l’échelle (1’ = 1’’). Cela veut dire que vous serez capables de nous montrer en 7’ de démonstration le fonctionnement d’une équipe de travail de 7 heures.~~
* L’application doit permettre de visualiser l’état de chaque personne (salariés ou clients) et de chaque objet modélisé ainsi que les situations limites~~, et mettre des alertes sur le manque de telle ou telle ressource (presque plus d’assiettes ou plus de verres du tout, …). Pour cela vous devez avoir une fenêtre de contrôle (vous pouvez vous inspirer des pages de contrôle des superviseurs comme Nagios ou Centreon avec lesquels vous avez travaillé dans l’UE précédente).~~
* Prévoir le mode « **PAUSE** » de votre application pour pouvoir stopper tous les processus et analyser la situation en cours.
* ~~Modélisation~~/implémentation de plusieurs catégories de clients avec des comportements différents.
* ~~Modélisation~~/implémentation de tous les postes décrits dans la partie « Description de postes ».
* ~~Modélisation/implémentation de tout le matériel décrit (et nécessaire pour la réalisation des recettes)~~
* ~~Préconisations sur le dimensionnement du restaurant et des ressources : les chiffres donnés sont corrects, surdimensionnés ou sous-dimensionnés. Dans ce cas-là, vous devez donner les postes à pourvoir ou à supprimer et le matériel à garder, à supprimer ou à racheter. Egalement, on attend de vous des préconisations sur le processus global de la gestion du restaurant.~~
* Les temps de chaque tâche, les quantités ~~d’objets ou~~ des postes, le nombre de clients par type, le temps en mode accéléré, … tout doit être paramétrable dans l’application.

En bref :

* La simulation graphique n’est plus demandée
* La gestion du matériel n’est plus demandée.
* Les préconisations pour les aménagements optimums du restaurant ne sont plus demandées.
* L’interface à produire n’affiche uniquement que les logs des différents postes instanciés.

## Technique

* (**OBLIGATOIRE**) Diagrammes de composants ~~et classes de l’application (ou applications).~~
* (**OBLIGATOIRE**) Diagramme(s) de séquences
* (**OBLIGATOIRE**) Au moins un IPC par type (synchro, échange de données ou les deux) doit être utilisé. Les threads vous seront indispensables également. A vous de décider sur l’utilisation d’un processus léger ou pas en fonction des situations. Voir ressource « **Aide-mémoire** » IPC sur Moodle.
* (**OBLIGATOIRE**) Utilisation des pools
* (**OBLIGATOIRE**) Utilisation des sockets pour les échanges entre la salle de restauration et la cuisine.
* (**OBLIGATOIRE**) Utilisation du langage C# .NET.
* **(OBLIGATOIRE)** L’ensemble des scénarii d’activité doit être enregistré en BDD
* ~~(~~**~~OBLIGATOIRE~~**~~) Base de données de stocks en SQL Serveur, actualisée avec les livraisons et mise à jour en temps-réel en fonction des commandes.~~
* (**OBLIGATOIRE**) Utilisation des DP ~~(au moins 5 au choix - observer, strategy, builder, factory, decorator, singleton, … -~~ en plus du MVC qui est obligatoire).
* ~~(~~**~~OBLIGATOIRE~~**~~) Utilisation de Git et du TDD pendant tout le développement du projet (tous les membres du groupe doivent démontrer leur travail sur Git. Le tuteur regardera les statistiques d’utilisation des fichiers tout au long du projet).~~
* (**BONUS 1**) Chaque tâche de chaque processus ou thread doit être horodaté dans un log (en BDD ou en fichier log unique)
* **(BONUS 2**) Utilisation de 2 machines Windows distinctes : une pour la cuisine et une pour la salle de restauration
* **(BONUS 3**) Utilisation de 2 machines distinctes pour la cuisine et la salle de restauration : une sur Windows avec .NET et l’autre sur Linux en Java.
* **(BONUS 4)** Possibilité de sauvegarde une situation déterminée dans un service et de pouvoir la « **rejouer** » plus tard. Cela peut être très utile en complément du mode **PAUSE**.

# Organisation du projet et livrables attendus

Le projet se déroule en groupes de 4 (max. 5 personnes) du 3 décembre au 12 décembre 2018.

* Au plus tard le mercredi 5 décembre avant 12h00, vous devez rendre le « Dossier Architecture (logiciel) » comportant :
  + Tous les diagrammes UML fonctionnels et techniques demandés.
  + Détail et explication de tous les DP utilisés dans l’application, MVC inclus.
  + MCD de la BDD et un script SQL pour sa création.
  + Ce livrable est à déposer sur un répertoire de votre dépôt Git (le nommer obligatoirement **LIVRABLES FINAUX**, svp) et vous informerez votre tuteur pour qu’il puisse le récupérer en lui envoyant un lien vers votre dépôt Git.
  + Ce livrable peut être rendu avant si vous avez terminé la modélisation mais attention à ne pas le négliger. Avoir une bonne modélisation et penser à l’utilisation des DP est indispensable pour réussir ce projet. Cela vous évitera des développements trop précipités et parfois inutiles. Si la modélisation est bien faite, vous gagnerez du temps pour le codage.
* ~~Le 11 avant 17h, mettre dans le répertoire~~**~~LIVRABLES FINAUX~~** ~~de Git, toutes les sources de votre projet (~~**~~pour rappel~~** ~~- Git doit être utilisé pendant le développement et pas seulement comme un dépôt pour les livrables !). C’est uniquement cela qui pourra être analysé et évalué par le jury du projet. N’oubliez pas que vous devez travailler en TDD et donc, il est indispensable de mettre le rapport des tests effectués (soit automatiquement généré par l’IDE, soit un rapport de tests et recette que vous avez effectué vous-mêmes).~~
* ~~La journée du 12 décembre sera consacrée à la soutenance de votre projet. Elle se déroulera pendant 1h15 de la façon suivante :~~ 
  + ~~Démonstration de votre application en temps accéléré. Présentation d’une ou de plusieurs situations normales ou posant des problèmes (n’hésitez pas à utiliser plusieurs PC en parallèle pour cela). Donner vos préconisations. (Max 20’)~~
  + ~~Présentation d’un slide sur la répartition des tâches et les responsabilités de chaque membre du groupe. (Max 1-2’)~~
  + ~~Chaque membre du groupe doit démontrer ses connaissances sur le projet en exposant au jury une explication technique sur sa partie. (Max 2’ par personne)~~
  + ~~Le jury pourra donc vous poser des questions sur tous vos livrables et sur la soutenance. Une partie de la note finale sera basée sur vos réponses aux questions posées par les jurys (Max 15’).~~
  + ~~Délibération des jurys (Max 15’)~~
  + ~~Restitution au groupe (Max 10’)~~
* Une vidéo technique explicative permettant aux jurys qui évaluera votre application, de s’y retrouver dans votre programme. Durant cette vidéo, il est inutile que l’ensemble du groupe prenne la parole. De la même manière, inutile de vous filmer, il sera plus intéressant de capturer votre écran pour illustrer vos explications. L’idée de cette vidéo est de permettre au jury de comprendre la philosophie et l’architecture de votre applicatif. La durée de votre vidéo n’est pas fixée, utilisez le temps qu’il vous semble nécessaire pour expliquer techniquement où se trouvent les différents éléments et comment les utiliser.
* Un rapport de gestion de projet faisant apparaître la charge de travail de chacun des membres ainsi que sa participation et implication dans le projet.
* La deadline de livraison de l’applicatif est le 17 décembre 2018 17h00.

En bref :

* Pas de soutenance
* Livraison le 17 décembre 2018 08h00
* Une vidéo explicative et technique

# Grille d’évaluation

Le projet comporte deux notes différentes dans l’UE 5.2 :

* Note Dossier Architecture (logiciel) du projet Programmation système. (coeff. 1)
* Note de la réalisation du Projet Programmation système. (coeff. 3)

La note finale du projet sera calculée de la manière suivante – voir grille :

* 70% sur la réalisation demandée (certains points obligatoires).
* ~~40% sur les compétences individuelles (intervention technique pendant la soutenance et question(s) posée(s) par les jurys)~~
* 20% sur l’évaluation par les pairs
* ~~10% sur le jury en fonction de la prestation orale lors de la soutenance~~
* 10% sur l’évaluation de la qualité, de la clarté et de la pertinence des explications techniques de la vidéo.

IMPORTANT : le projet est composé d’une partie OBLIGATOIRE. Ce qui est demandé doit être totalement réalisé et présenté le jour de la soutenance. Si cette partie n’est pas réalisée, la note sera C pour chaque personne du groupe (voire D si les jurys constatent un manque important de travail).

Les fonctionnalités optionnelles et/ou BONUS peuvent vous aider à compenser certains manques dans vos réalisations. MAIS attention à ne pas vous disperser pour autant dans trop de choses à la fois. Elles seront prises en compte seulement si la partie obligatoire est réalisée.

~~La grille d’évaluation étudiant détaillée sera disponible à partir du 6 décembre sur Moodle.~~

La grille d’évaluation est en cours de modification. Je reviens vers vous dès que je l’ai finalisée et faite validée.