M1 : Ingénierie du Logiciel

UNIVERSITE PIERRE & MARIE CURIE (PARIS VI)

Examen Réparti 1ere partie

15 novembre 2017 (2 heures avec documents : tous SAUF ANNALES CORRIGEES). Barème indicatif sur 20 points.

1. Questions de cours [5 Pts]

Répondez de façon précise et concise aux questions.

Barème : VALABLE sur toutes les questions de cours : -25 à -50% si la réponse inclut la bonne idée, mais qu'elle est noyée dans des infos ou autres réponses fausses/inappropriées.

Q1.1 : Quel est la nature d'un *bus logiciel* dans une plateforme orientée composants ? Quels problèmes en particulier sont traités par un bus logiciel ?

Offre une Api de communication entre les composants : empaqueter les requêtes, les véhiculer à la cible (système de nommage), les désérialiser, retransmettre la réponse

Permet de s'affranchir de l'hétérogénéité : plateforme d'éxecution + langages réalisant les composants

Barème :

50% : nature : véhicule les requêtes, assure la comunication entre les composants

50% gestion de l'hétérogénéité, de la diversité des plateformes et des langages réalisant les composants. On accepte aussi 50% les réponses qui détaillent les responsabilités du bus : acheminement des requêtes, service de nommage, proxy pour les composants distants…

Q1.2 : Comment peut-on s'assurer que le jeu de tests de validation est complet ?

On ne peut pas être « complet ».

On ne peut que se satisfaire de métriques, e.g. couverture des SN, ALT et EXC des fiches détaillées avec au moins 1 jeu de données.

70% test = incomplet

30% métriques pertinentes suggérées : de quoi se satisfait-on en pratique, chercher à couvrir le besoin client…

0 dès qu'on suggère qu'il y a effectivement une solution pour être « complet » avec un jeu de tests.

Q1.3 : On considère le diagramme de cas d'utilisation d'un système de vente en ligne. Comment modéliser que le cas d'utilisation « mettre en vente » précède nécessairement le cas d'utilisation « acheter produit » ?

On ne peut pas le faire directement sur le diagramme ; ni extends ni include ne répondent à cette question de séquencement temporel.

On peut s'en sortir avec différents acteurs (e.g. connexion + acteur Utilisateur Connecté)

Ou avec des préconditions/postconditions :

- mettre en vente => post : il y a des produits.
- Et « acheter produit » précondtion « il y a des produits »

80% ni include ni extends ne fonctionnent ; on ne peut pas le modéliser directement

20% on suggère une façon via les fiches détaillée (pré/post) ou les acteurs, ou on détaille sa réponse en expliquant pourquoi on ne fait pas ça dans ces diagrammes.

Q1.4 : Pourquoi ne peut-on se contenter du code pour décrire une application ? Pourquoi le code reste-t-il cependant nécessaire ?

Diversité des intevenants, niveau d'abstraction unique (faible) : le code n'est pas l'artefact qui est adapté au métier de tous les intervenants

Importance pour discuter entre intervenants « code », e.g. algorithmes, couches technique. Ca reste un artefact important du développement pour les développeurs + construire l'executable.

Barème :

50% le code n'est pas adapté à tous les intervenants, e.g. Client

50% le code reste essentiel pour les développeurs et le développement

2. Problème: Analyse de eServer [15 Pts]

eServer est une application dédiée aux restaurateurs, qui leur facilite la gestion quotidienne du restaurant.

L'application se décompose en quatre parties :

- Les serveurs sont équipés d'un smartphone ; ils saisissent dessus les commandes des clients (qui sont automatiquement transmises aux cuisines) tout au long du service, leur apportent les plats et l'addition, et desservent les tables. Bien sûr, les clients déjà installés peuvent ajouter des plats à leur commande. Les serveurs sont notifiés quand les plats commandés sont prêts.
- Les cuisines sont équipées d'une interface tactile simple où les commandes des clients peuvent être visualisées par ordre chronologique. Le cuisinier prépare les plats, puis peut signaler quand les plats sont prêts à l'application, ce qui notifie le serveur concerné (vibration du téléphone) et les supprime de la liste des commandes en cours.
- La partie caisse et facturation permet d'imprimer un reçu pour un groupe de clients. Il est également possible d'imprimer plusieurs copies du reçu si nécessaire.
- L'interface de gestion du restaurant permet de suivre et d'organiser l'activité du restaurant. Avant le début du service, on y saisit la carte du restaurant (plats, formules…). A tout moment, on peut inspecter des statistiques sur l'activité du restaurant : chiffre d'affaire quotidien et mensuel, plats les plus populaires…

Le menu du restaurant est composé de plats. Chaque plat est qualifié par

- Son nom et sa description (e.g. « Crêpe Grand-Père », « chocolat noir, amandes »)
- Sa catégorie (entrée, plat, dessert…)

- Les extras qui peuvent l'accompagner et pour chaque extra le prix (e.g. supplément chantilly 1,5 eu) .

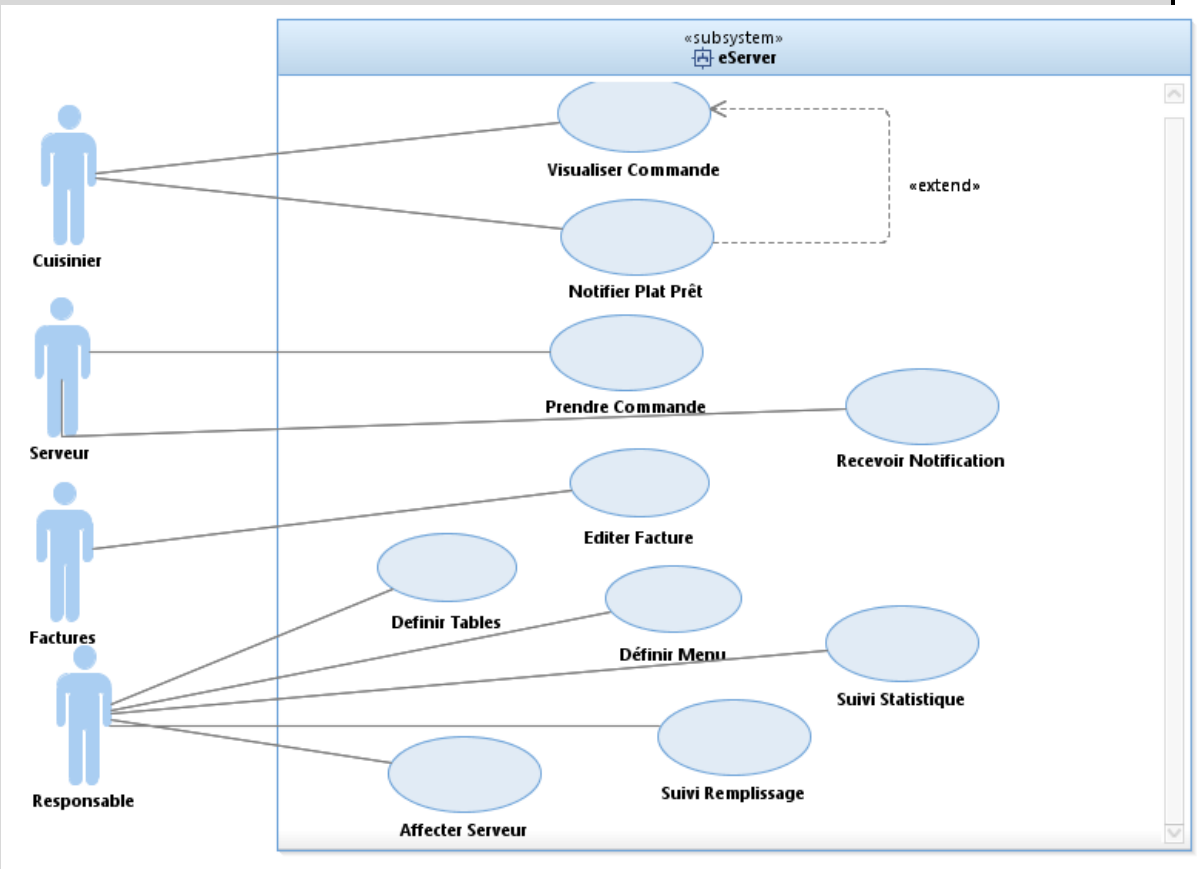
La carte du restaurant contient le prix des différents produits à la vente :

- Les plats vendus à la carte, chacun ayant un prix
- Des formules à prix fixe, qui offrent des alternatives pour chaque étape, choisies parmi les plats. e.g. formule entrée + plat 9,40 : œuf mayo **ou** salade **ou** charcuterie + poulet **ou** steak **ou** poisson. Les formules peuvent contenir un nombre arbitraire d'étapes et de plats au sein de chaque étape.

La gestion des tables du restaurant permet d'ajouter les éléments suivants :

- Chaque serveur est responsable d'un sous-ensemble des tables, qui lui sont affectées dans l'interface de gestion. Quand il démarre la commande d'un groupe de clients, il note dans l'application les tables concernées par ce groupe. Ces tables seront notées occupées jusqu'à l'édition de la facture.
- Dans l'interface de gestion, on peut définir un plan de table graphiquement. Pendant le service, on définit l'affectation des tables aux serveurs. Enfin on peut visualiser l'état du restaurant en temps réel : tables occupées ou disponibles, serveur affecté à chaque table.

Question 2.1 : (3 pts) Réalisez le diagramme de cas d'utilisation de la phase d'analyse. Vous justifierez tous vos choix, par un texte ou des annotations sur le diagramme.



Cuisine :

* visualiser commande

* noter commande prête

Serveur

- Démarrer commande
- Modifier commande/ajouter
- Recevoir notification plats prêts

Gestionnaire

- Définir menu/carte
- Consulter stats (CA,…)

Facturation

- Editer recu

+

- Définir plan de table
- Affecter serveur
- Visualiser état restaurant
- (extends) noter tables prises sur prendre commande (optionnel)

Barème :

10% par use case * 10 use case du diagramme du corrigé

Il doit être lié à un acteur « raisonnable ». La séparation entre Responsable et Serveur doit apparaitre (affecter serveurs n'est pas un use case de Serveur + état du resto, stats). Les serveurs ne définissent pas la carte etc… L'acteur « Factures » ou Caissier n'est pas nécessairement identifié, on peut affecter ce cas au serveur ou au Responsable. Si l'acteur n'est pas raisonnable 0% pour ce use case.

On donne 5% pour le use case du serveur « apporter plats » s'il est présent à la place de « recevoir notification plat prêt ». On peut supposer que ça couvre la notification.

On accepte des extends et/ou includes raisonnables (un minimum commentés/justifiés) par exemple un « imprimer copies recu » qui étend « imprimer reçu »

-5% par use case en trop/redondant : se loger,

Fautes fréquentes :

-10% use cases hors cadre : mettre la table, desservir la table, sourire au client… On ne fait pas -10% par use case hors cadre, mais -10% dès qu'il y a un use case HS.

-10 si tout est détaillé (indexer ou publier tweet en use case qui sont inclus dans tweeter par exemple…) ça découle des use case, mais ce n'en sont pas.

-5 à -15% pour les fautes

-5 par use case mal formulé : on veut un verbe qui exprime l'action du point de vue de l'acteur.

-10% aucun commentaire/aucun texte pour accompagner le diagramme, diagramme sec

Jusqu'à -20% par héritage, include ou extend injustifiable ou autre incohérence/mésusage d'UML.

-10% si on ne précise pas qui fait l'action dans le scenario (use case sans acteur lié)

Question 2.2 : (3 pts) Précisez la ou les fiches détaillée(s) (acteurs concernés, pré-conditions, post-conditions, scénario nominal, alternatives, exceptions) du (ou des) cas d'utilisation(s) correspondant aux interactions possibles avec le système depuis les cuisines.

2 use case a priori :

consulter commandes + notifier plat prêt

Fiche 1

Titre : Visualiser Commandes

Acteur : Cuisinier

Hypothèse : aucune

Pré : aucune

Post : aucune

Scénario :

1. L'utilisateur clique sur « commandes en cours »
2. Le système affiche l'ensemble des commandes en cours : elles sont groupées par table/groupe de client, et triées chronologiquement (plus ancien d'abord). On peut voir le titre du plat, l'heure de la commande et les extras commandés.

Alternative A1 : Valider Préparation

A1.1 En SN3, l'utilisateur peut sélectionner un des plats commandés ; cf UC « Notifier Plat prêt »

Fiche 2 :

Titre : Notifier Plat prêt

Acteur : Cuisinier

Hypothèse : aucune

Pré : des plats commandés mais pas encore préparés sont visibles dans l'interface

Post : les plats sélectionnés sont notés « préparé », le serveur est notifié

Scénario :

1. L'utilisateur sélectionne (coche) un ou plusieurs des plats à l'état « commandé » appartenant à un même groupe de clients
2. L'utilisateur clique sur « plats prêts »
3. Le système affiche la liste des plats concernés et demande une validation
4. L'utilisateur valide
5. Le système notifie le serveur affecté aux tables de ce groupe de client que les plats sont prêts
6. Le système met à jour les commandes : les commandes concernées passent à l'état « préparé »

Exception E1 :Annuler

E1.1 En SN4, l'utilisateur peut abandonner l'opération ; le système revient à l'affichage des commandes en cours.

Barême :sur 100%

10% Cohérent avec le diagramme de use case, une ou deux fiches. On doit voir les include/extends du diagramme dans les fiches (include => étape du SN, extends = ALT).

20% visualiser commandes on doit décrire un scenario raisonnable qui fait que le système affiche les commandes, par ordre chronologique (10% sur « chronologique », élément du CdC)

20% il y a une procédure permettant au cuisinier de sélectionner les plats qui sont prêts dans une liste 10% s'il fait la sélection sans qu'on ait spécifié l'affichage d'une liste pertinente par le système. 0% si on ne voit pas l'information sur « quels plats/commandes sont prêts » circuler du cuisinier vers le système. On donne les 20% si le lien est fait via une post-condition cohérente dans « Visualiser » avec la pré-condition de « notifier »

20% Notifier plat prêt spécifie bien qu'on notifie le Serveur concerné. On donne 10% si c'est dit en post-condition mais pas dans le scenario nominal. On ne sanctionne pas si ce n'est pas correctement précisé quel serveur on doit signaler.

20% La mise à jour de l'état des commandes est modélisé : on précise que l'affichage se fait sur les commandes « commandé »/en cours (10%), et que les commandes basculent à l'état « préparé » ou sont retirées de la liste quand on notifie prêt (10%). On accepte la modélisation dans le scenario nominal et/ou dans les post-conditions pour ces mise à jour.

10% exception ou alternative « annuler » dans « notifier prêt »

On peut éventuellement donner 10% pour des ALT ou exception bien spécifiées qui traitent d'autres cas pertinents. Attention à la cohérence avec les dia de use case, il y a 10% dans le barème pour la cohérence.

Cette question est très délicate à corriger. Il faut donc vérifier les points suivants.

-10 pour incohérence globale du texte, utilisation incorrecte des champs Pré/Post/Scenario etc... En particulier, -10% si les préconditions/hypothèses sont testées dans le scenario et ou si les étapes ne sont pas bien affectées à acteur ou système

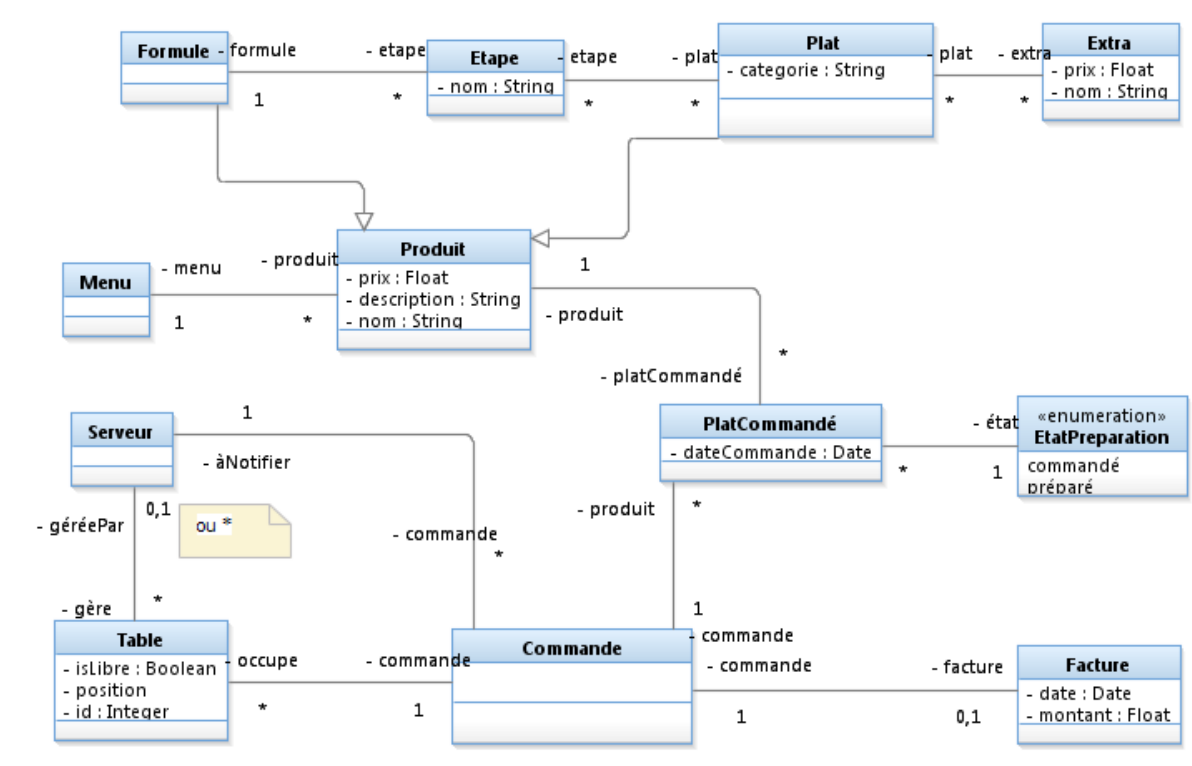
Erreurs fréquentes :

-5% à -10% on spécifie des pré/post conditions qui parlent de l'état du restaurant, pas du système

-10% on ne sait pas clairement qui du système ou de l'acteur fait l'action dans une étape du scenario

-15% :Spécification d'étapes hors système comme étapes du scenario (e.g. le cuisinier prépare un plat.) .

Question 2.3 : (4 points) Réalisez le diagramme de classes métier de la phase d'analyse. Vous justifierez tous vos choix, par un texte ou des annotations sur le diagramme. Ne modélisez pas la classe représentant le « Système », introduite dans l'approche en V du module.



Voici ma correction : si 10% pour un élément, ne donner que 5 si la formulation est incomplète (e.g. cardianlités mal renseignées, il manque des attributs parmi ceux recherchés,…).

Sur 115%: (s'il manque une partie des éléments donner 5%)

10 % : Un plat possède : un prix à la carte, un nom, une description, une catégorie (enum OK),

10% : Un Extra possède un nom, un prix, et est associé Extra *-* plats, ou Extra *-1 plats (au choix)

10% : une formule comporte * étapes, les étapes comportent * plats

10% : une formule a un prix, un nom, et une étape à un nom

10% le menu contient des formules et des plats à la carte

10% Une commande est associé à des produits : plats ou menus munis d'un prix

10% Un plat sur une commande a un état. On accepte que ce soit la commande entière qui ait un état.

10% une facture a un montant, et est associée à une commande

10% le serveur est lié aux commandes d'une manière ou d'une autre : on sait quel serveur notifier sur plat prêt. Le lien entre Commande et Table répond à ce besoin, si le serveur est lié à la table.

10% Table est affectée à serveur,

10% la table possède un état ou une association vers « Client ». NB : le corrigé a fusionné la notion de Client et Commande.

+5% la table a une position

0% lien entre cuisinier et plat préparé, c'est de la dynamique mais on ne sanctionne pas.

Malus :

-10% à -20% éléments dynamiques qui n'ont pas à apparaitre, e.g. Acteurs.

-10% si associations orientées, compositions etc…

-10% si opérations sur les classes

-10% à 20% pour toute autre faute ou aberration

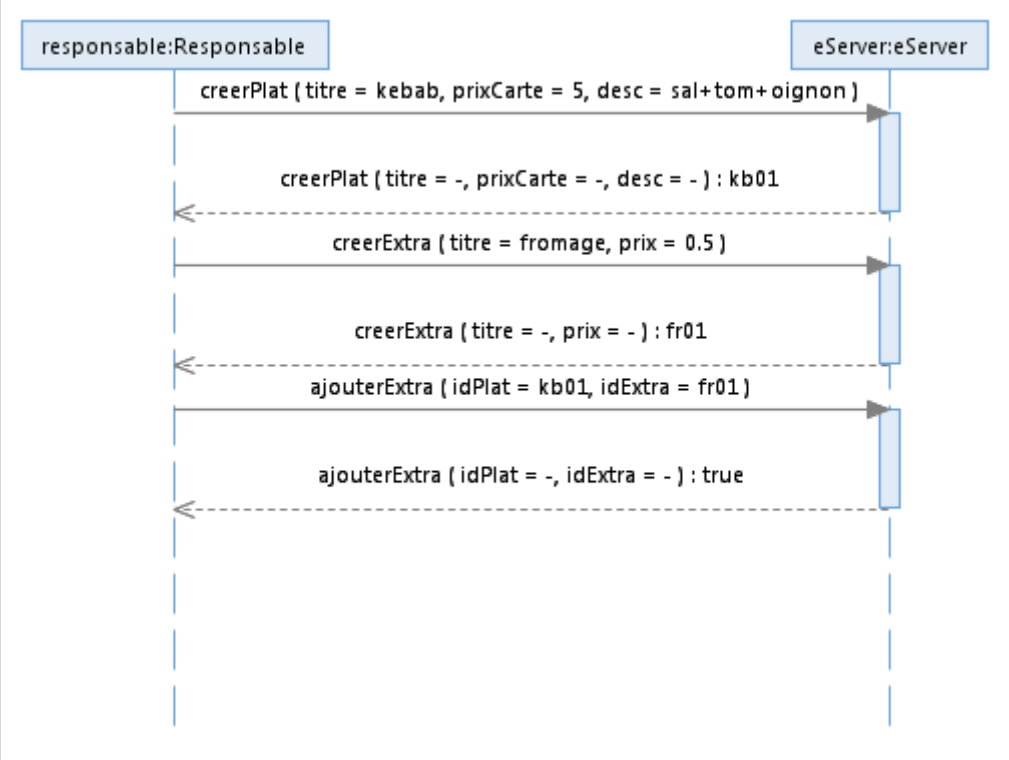
Question 2.4 : (2,5 pts)

A) Réalisez un diagramme de séquence de niveau analyse présentant le déroulement (scénario **nominal**) des étapes permettant à un utilisateur de créer une carte offrant un unique plat mais avec un extra possible. (e.g. Kebab 5 eu + supplément Fromage 0.5 eu). On évitera de sur-spécifier les actions privées du système.

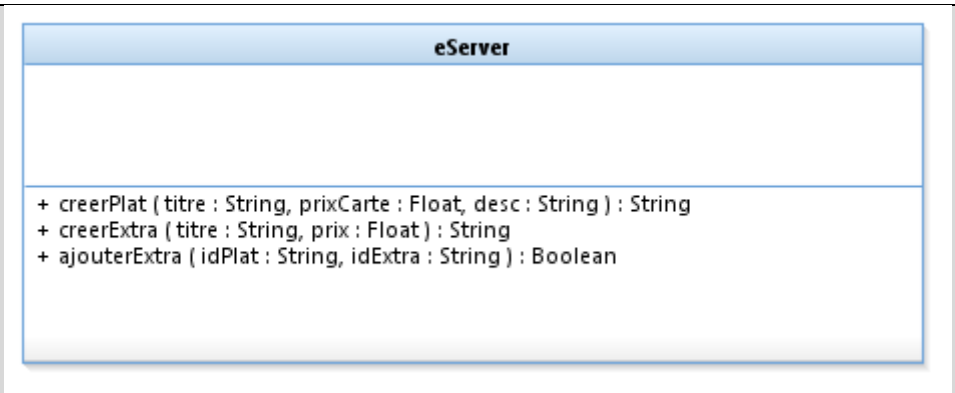
B) Dessinez la classe « système » afin de préciser les opérations identifiées dans cette séquence (signature, visibilité).

Essentiellement, on doit voir sur ce diagramme toute l'information circuler de l'acteur vers le système, sous une forme ou une autre. On ne doit pas nécessairement voir les opérations privées en self call

A priori on a un séquence très simple ici. Le corrigé est minimal mais suffisant.



Donc :



Barême :

A) 80%

+10% lignes de vie correctes : acteur vs système

On cherche de l'information qu'on voit circuler de l'acteur vers le système

20 % La description complète du kebab : 10% titre + prix, 10% catégorie + description(c'est un plat)

20% La description de l'extra « fromage » + prix

20% une API raisonnable pour lier les deux est proposée. i.e. c'est l'acteur qui dit au système qu'il y a un lien entre le kebab et le fromage. On donne 10% sur ce point si on voit une seule invocation avec kebab+fromage, mais que la signature permettrait de passer d'autres paires « nom extra, prix ». 0% si on ne peut pas clairement aussi ajouter l'extra « sans oignon » à 0 eu sur le kebab.

10% : modélisation de l'affichage en self loop sur le système, modélisation d'actions pertinentes comme « enregistrer » du système, commentaires …

-10% on ne voit pas clairement que les lignes de vie sont des instances (notation o:Obj)

-20% si appel du système à une opération de l'acteur Agent. L'envoi asynchrone d'un message, ou une note expliquant qu'on considère que Joe représente l'acteur et son IHM => -10%. Cela reste incorrect. On cherche les responsabilités du système, pas des acteurs (donc externes au système).

-20% si on affecte des méthodes à des classes métier

B)

20% : signature(s) cohérentes avec le diag de séquence et réalisables, 0% dès qu'une incohérence est constatée. Les méthodes correspondant à d'autres use case sont tolérées mais ne donnent pas de points (hors sujet)

les self calls doivent être private (-10% si ce n'est pas le cas).

Question 2.5 : (2,5 pts) Ecrivez un test de validation couvrant la commande d'un plat supplémentaire pour un groupe de clients déjà installés.

TV042 : Test ajout plat

Contexte : La commande des clients table 12 est créée (TV12) avec ce smartphone; la carte du restaurant est définie (TV5 à TV8)

Entrée : plat choisi : « kebab », table « 12 »

Scenario :

1. Le serveur sélectionne la commande « table 12 » dans la liste des commandes en cours
2. Il clique sur « ajouter plat »
3. Il sélectionne le plat « kebab »

Résultat attendu : la commande du kebab est passée, le kebab est ajouté aux clients de table 12 et sera facturé. Le cuisinier doit voir la commande du kebab sur son écran.

Moyens de vérification : resélectionner la commande dans le téléphone pour constater que le kebab y figure ; constater dans l'interface « cuisines » que la commande du kebab est présente dans la liste des plats en attente de préparation.

Barême :

20% le contexte précise qu'on a une commande en cours pour cette table

15% on voit l'étape de sélection de la commande dans une liste

15% on voit les données de saisie, précisées dans l'entrée : kebab. On ne choisit pas un plat au hasard.

15% résultat attendu précise que le cuisinier voit la commande

15% résultat attendu précise que le plat en question est ajouté

20% le moyen de vérification est plus que « visuel », on contrôle l'existence du plat commandé, e.g. via la cuisine ou l'interface du serveur.

-10% il existe des données saisies qui ne sont pas mentionnées dans la section « entrée », ou des données dans Entrée qui ne sont pas utilisées

-20% scenario difficilement réalisable, mentionne autre que les actions utilisateur, imprécis (il doit être reproductible sans réfléchir)

-25% le scenario mentionne des actions du système (autre que résultat attendu)