TIW1 – Intergiciels et Services – Examen

Durée : 1 h 30 – Documents autorisés (4 pages max) – Ordinateurs, calculatrices, tablettes, téléphones portables… interdits

Questions de cours (barème : 12 points)

Toutes les questions ont au moins une bonne et une mauvaise réponse.
Il faut cocher toutes les bonnes propositions d'une question pour avoir un point.
Si vous cochez une mauvaise proposition, la question est annulée.
Remplissez au stylo noir ou bleu la case de la ou des bonne(s) réponse(s) ; une croix ne suffit pas.
Ne barrez pas une mauvaise réponse, mettez du blanc.
Ne redessinez pas une case que vous avez effacée, laissez blanc.
La première question n'est pas prise en compte dans la notation, mais elle est indispensable pour la notation des autres questions.

1. Identification du sujet :
   Cochez les cases B et C

2. Le pattern Object Pool :
   a) s'appuie sur les méthodes de service des composants
   b) s'appuie sur les méthodes de gestion du cycle de vie des composants
   c) s'appuie sur les méthodes de gestion de la persistance des composants
   d) nécessite un pattern Inversion de Contrôle
   e) améliore la scalabilité d'une application

3. Parmi les propositions suivantes, cochez celles qui sont des référentiels de dépendances « résolvables »
   (les lettres en majuscules indiquent des types, celles en minuscules indiquent des instances et la flèche indique une dépendance) :
   a) A → B ; B → C ; C → A
   b) A → B ; B → C ; C → C
   c) A → b ; b → C ; C → D
   d) A → C ; B → C ; C → d
   e) A → C ; D → E ; B → t

4. Le pattern Annuaire :
   a) supporte le pattern faible couplage
   b) supporte le principe d'Inversion de Dépendances
   c) nécessite d'avoir mis en place un pattern Inversion de Contrôle
   d) est implémenté dans Tomcat
   e) supporte le pattern Chaîne de Responsabilités

5. Dans le framework que vous avez réalisé (idéalement) en TP, les annotations pouvaient être utilisées pour :
   a) injecter des dépendances dans des composants
   b) injecter des POJOs en tant que composants
   c) spécifier les comportements de certains composants
   d) instancier des objets
   e) déployer des applications

6. Parmi les propositions suivantes, cochez celles qui correspondent à des méthodes d'injection de dépendances :
   a) Par proxy
   b) Par interface
   c) Par classe
   d) Par getter
   e) Par constructeur

7. Un framework entièrement conforme à la spécification Jakarta EE :
   a) comporte 3 conteneurs
   b) comporte 2 conteneurs
   c) est interrogeable en HTTP
   d) est interrogeable en RMI
   e) est interrogeable par appel de méthodes Java

8. La Programmation Orientée-Aspects :
   a) permet de traiter les préoccupations d'une application une par une
   b) permet de mélanger les différentes préoccupations d'une application pour qu'elle puisse fonctionner
   c) s'appuie sur le principe de séparation des responsabilités
   d) nécessite un faible couplage entre les composants
   e) s'appuie sur des processeurs d'annotations

9. La notion de composant dynamique :
   a) s'appuie sur un pattern annuaire
   b) s'appuie sur les méthodes de gestion du cycle de vie de ces composants
   c) s'appuie sur le principe de Design by Contract
   d) nécessite un pattern Inversion de Contrôle
   e) améliore la scalabilité d'une application

10. Le Domain-Driven Design (DDD) :
   a) s'appuie sur la notion de framework
   b) s'appuie sur la notion de bounded context
   c) met en valeur les processus métier
   d) permet de diminuer la quantité de code produit
   e) favorise l'utilisation du pattern DTO

11. Les architectures à base de micro-services favorisent :
   a) la réutilisabilité du code
   b) les cycles de développement courts
   c) l'utilisation de bus de services
   d) la communication entre les équipes de développement
   e) la scalabilité d'une application

12. Pour mettre en place des compositions de services, on peut utiliser :
   a) un sémaphore
   b) une pipeline de services
   c) de l'orchestration
   d) du réordonnancement
   e) une chorégraphie

13. Quelles sont les causes qui peuvent amener à une démarche d'urbanisation du SI ?
   a) modifier les objectifs stratégiques à l'aide d'une approche bottom-up
   b) modifier les objectifs stratégiques à l'aide d'une approche top-down
   c) modifier l'architecture fonctionnelle pour intégrer une nouvelle infrastructure technique
   d) identifier des « impondérables » (contraintes extérieures) qui nécessitent de redéfinir l'architecture fonctionnelle
   e) anticiper un changement de processus métier

Étude de cas (barème : 11 points)

Répondre sur la copie d'examen.

Chaque fin d'année, les services du **Père Noël** s'activent pour faire en sorte que les **enfants** puissent obtenir les **jouets** dont ils rêvent. Depuis des temps immémoriaux, cela se passe de la manière suivante :

- Les enfants écrivent des **lettres au Père Noël** et les envoient au Pôle Nord, où habite le Père Noël.
- Le Père Noël lit ces lettres, prend des notes et tient à jour une liste des demandes ; il a récemment abandonné les papyrus qui résistent mal au climat local, et acheté un ordinateur qui lui permet de stocker la liste dans un fichier Excel.
- Le Père Noël est entouré de **Lutins**, à qui envoie la liste pour qu'ils fabriquent les jouets et planifient la « **Grande Livraison** ».
- Le 24 décembre au soir, tout doit être prêt : les jouets dans la **hotte**, les **rennes** attelés au **traîneau**, mais surtout, un itinéraire très précis doit avoir été défini pour ne rater aucune livraison et ne pas perdre de temps.

Pour faire face à l'augmentation de la population et du prix des matières premières, cette année, le Père Noël se modernise et a décidé de tirer parti des dernières évolutions technologiques pour répondre aux exigences toujours croissantes des enfants.

Zoe, 10 ans, experte métier de la commande de jouets, précise les contours de cette application révolutionnaire :

- Les enfants peuvent discuter avec un **LLM** pour imaginer des jouets qui n'existent pas encore. À l'issue d'une séance de chat avec un enfant, le LLM génère un nom et une description pour un nouveau jouet.
- Dans une lettre au Père Noël figurent le nom, l'âge et l'adresse de l'enfant, ainsi que le nom du jouet généré. Le système le relie automatiquement à sa description, et l'enrichit des dispositions réglementaires en vigueur en fonction de l'âge et de la zone géographique de l'enfant. Le tout forme un document appelé : **demande de jouet**.
- Les lutins ont également accès à l'interface du LLM, qui peut retrouver le contexte de la discussion avec un enfant pour préciser des points qui ne seraient pas clairs dans les spécifications du jouet.
- Le Père Noël a constaté que tous ses lutins ne sont pas également performants. Afin de rationaliser ses processus, il les a donc structurés en plusieurs équipes : les **manuels** s'occupent de la fabrication des jouets, alors que les **intellectuels** gèrent la logistique et planifient les livraisons. Ceux qui n'entrent dans aucune de ces deux catégories sont dans l'équipe « illettrés, désœuvrés, lents ou étourdis » (**idle**) : ils passent le plus clair de leur temps à soigner les rennes et entretenir le traîneau (ou pas), mais peuvent temporairement venir renforcer les deux autres équipes en cas de besoin ponctuel.
- Le Père Noël a donc besoin de savoir à tout moment qui fait quoi (et qui ne fait rien), combien il y a de demandes de jouets en cours de fabrication ou en cours de planification, etc., et souhaite donc avoir un **dashboard** qui met ces Key Performance Indicators (**KPI**) à sa disposition, pour que tout soit prêt pour la Grande Livraison du 24 décembre au soir.
- Malgré toute la bonne volonté des lutins, il arrive tout de même que des jouets ne fonctionnent pas ou ne soient pas livrés correctement. Dans ce cas, le Père Noël a tout prévu : il souhaite également mettre en ligne un **outil de suivi des livraisons et de la satisfaction** des enfants grâce auquel on peut retrouver si un jouet a été correctement testé et livré. Si ce n'est pas le cas, il s'engage à le faire re-fabriquer par ses meilleurs lutins manuels, et à le renvoyer par La Poste car il n'a pas le droit de circuler au milieu du ciel en traîneau les autres jours de l'année.

Malheureusement, aucun lutin n'est capable de concevoir ni de développer un tel système. C'est pourquoi vous avez été embauché.e pour mener à bien la démarche d'urbanisation du SI du Père Noël. Écrivez :

1. Le référentiel d'objectifs stratégiques (diagramme d'Ishikawa). (2pts)
2. Un diagramme représentant les processus de l'architecture cible (diagramme BPMN). (4pts)
3. La liste des paliers d'urbanisation qui vont permettre de passer du SI actuel au SI cible. (2pts)

Une fois que la démarche a [bien/un peu] avancé, vous en arrivez à l'étape cruciale de **préparation de la hotte** du Père Noël. Pour simplifier, **on suppose que tous les jouets rentrent dans la hotte**, mais qu'il ne faut pas que le Père Noël perde du temps à fouiller dedans pour retrouver un jouet : ils doivent avoir été empilés et pouvoir être dépilés dans l'ordre de la livraison. Ce sont quelques lutins oisifs de l'équipe idle qui s'occupent de cette tâche, juste avant la Grande Livraison ; il est donc important de leur faciliter la tâche au maximum, en leur indiquant les jouets qu'ils doivent placer dans la hotte **un par un, dans l'ordre inverse du dépilement**. Pour qu'un jouet puisse être placé dans la hotte, il faut :

- que l'équipe logistique ait déterminé l'**ordre de livraison** de ce jouet en fonction de l'adresse de l'enfant
- que le jouet ait été fabriqué et mesuré : on doit connaître ses **dimensions** pour savoir combien de lutins idle il faut pour le transporter depuis la caverne de production (où travaillent les lutins manuels) à la piste de décollage du traîneau et le placer dans la hotte

Un serveur centralisé est disponible et contient toutes les informations sur la logistique et en particulier l'ordre dans lequel seront livrés les jouets. Il s'appuie sur le framework que vous avez réalisé en TP (idéalement), et est interrogeable en HTTP. Vous disposez également d'un broker de messages RabbitMQ.

Chaque jouet est identifié et traité indépendamment par les lutins idle (sinon, ils s'embrouillent et la livraison se passe mal).

4. Écrivez un diagramme de processus (BPMN) qui positionne les différentes tâches entre la fabrication et la livraison. Indiquez de manière claire les flux physiques (lutins, jouet), les requêtes/réponses HTTP, les échanges de messages via RabbitMQ et les appels de méthodes Java. (3pts)