M1 : Ingénierie du Logiciel

UNIVERSITE PIERRE & MARIE CURIE (PARIS VI)

Examen Réparti 2eme partie

9 Janvier 2014 (2 heures avec documents : tous SAUF ANNALES CORRIGEES).
Barème indicatif sur 24 points (donne le poids relatif des questions) (max 20/20).

1. Questions de cours [2 Pts]

Répondez de façon précise et concise aux questions.

Q1.1 :

a) Comment réaliser une transformation modèle à modèle M2M, pour passer d'un **modèle** MA exprimé dans un langage A à un **modèle** MB exprimé dans un langage B ? Quels méta-modèles faut-il manipuler et de quelle manière ?

b) Quelles transformations T2M ou M2T faut-il ajouter si les modèles MA et MB sont exprimés dans les syntaxes **concrètes** de A et B (et pas directement en XMI).

2. Problème: Conception d'un Mailer POP [Barème sur 22 Pts]

Le protocole POP (Post Office Protocol) est un des standards permettant la communication par email. C'est un protocole textuel (ASCII) qu'offrent les serveurs de mail de façon standard sur le port 110. Pour interagir avec le serveur, il faut d'abord s'identifier et s'authentifier :

- USER *nom_de_votre_compte* : généralement, ce qui se trouve avant le « @ » de l'adresse électronique (erreur si le nom de compte est inconnu)
- PASS *mot_de_passe* :le mot de passe pour accéder à la boîte de courrier (erreur si le mot de passe est incorrect ou si la commande précédente n'était pas USER)

Puis, il est possible d'utiliser l'une des commandes POP3 listées ci-dessous (elles rendent toutes une erreur si l'utilisateur n'est pas correctement connecté).

- `LIST` : sans argument; donne sur une ligne le nombre **n** de messages et la taille totale **k** des messages, puis sur **n** lignes pour chaque message son identifiant et sa taille.
- `LIST` *numéro_du_message* : donne la taille du message (erreur si le message n'existe pas ou plus)
- `RETR` *numéro_du_message* : récupère le message indiqué et l'affiche (erreur si indice de message inexistant);
- `DELE` *numéro_du_message* : efface le message spécifié (erreur si message inexistant);
- `QUIT` : quitter la session en cours

On considère un protocole simplifié dans cet énoncé où les autres commandes POP3 ne seront pas utilisées.

Les divers arguments sont passés sous la forme de chaînes de caractères. Les réponses du serveur sont également des chaînes de caractère. Les réponses en plusieurs lignes (LIST sans argument, RETR) sont terminées par une ligne comportant seulement la chaîne « **.\n** ». Le serveur débute chaque réponse par la ligne « **+OK commentaire** » (nominal, suivi de la réponse à la commande) ou « **-Error commentaire** » (problème, pas de réponse possible). Le commentaire explique le message.

Modélisation POP3 (6 points)

Question 2.1 : (2 pts) Proposez une interface IPOP3 qui permette de capturer la fonctionnalité d'un serveur de mail. On construira des signatures pour toutes les opérations qui capturent le cas nominal ; les erreurs seront traitées via une hiérarchie de POPProtocolException qu'on ne demande pas de modéliser. On utilisera des entiers pour typer les identifiants de message ; les réponses multi-lignes seront traitées par un ensemble de chaînes de caractère.

Question 2.2 : (1 pt)

a) Représentez un composant représentant un serveur de mail sur un diagramme de composants. Représentez également un composant représentant un client pour le service (un outil pour lire ses mails par exemple).
b) Représentez sur un diagramme de structure interne une instanciation de ces composants qui les met en communication.

Question 2.3 : (3 pt) En cohérence avec les questions précédentes, représentez par un diagramme de séquence inter-composant le dialogue client-serveur retranscrit ici.

```
S: <en attente de la connexion TCP sur le port 110>
C: <ouverture de la connexion sur pop.lip6.fr, port 110>
S:    +OK POP3 server pop.lip6.fr ready <@lip6.fr>
C:    USER toto
S:     +OK User accepted
C:    PASS soleil
S:     +OK Pass accepted
C:    LIST
S:    +OK 2 messages (320 octets)
S:    1 120
S:    2 200
S:    .
C:    RETR 1
S:    +OK 120 octets
S:    From : bob@jmail.com
S:    <etc… le serveur POP3 envoie le message 1>
S:    .
C:    DELE 1
S:    +OK message 1 deleted
C:    QUIT
S:    +OK toto POP3 server signing off
C:  <fermeture de la connexion>
S:  <en attente de la prochaine connexion>
```

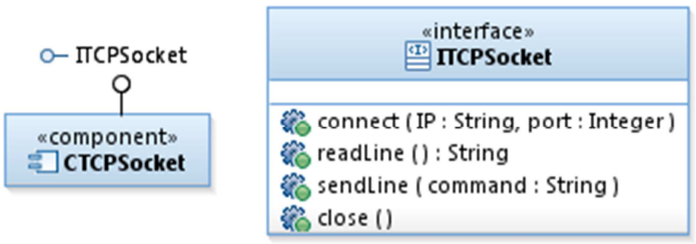
CServeurMail (4 points)

Question 2.4 : (4 pt) Représentez sur un diagramme de classes une conception détaillée pour un composant serveur de mail qui stocke et gère effectivement les messages des utilisateurs. On sera aussi détaillé que possible (utilisations de Map si c'est approprié, commentaires…). Pensez à correctement gérer l'aspect multi-utilisateurs. Les mails eux-même seront modélisés comme des entités très simples de façon à

coller au protocole POP. (En pratique le formatage des mails -- émetteur, date, relais, objet, pièces jointes etc… -- est le sujet d'autres standards, MIME en particulier qui dépassent le cadre de cet énoncé).

Proxy (7 points)

On s'intéresse à présent à la conception d'un composant CPOPPRoxy qui joue le rôle de proxy distant pour un serveur mail POP3 conforme au standard. Ce composant qui réalise l'interface IPOP3 proposée en 2.1 sera instancié sur la machine du client. Il s'appuie lui-même sur une Socket en mode connecté (TCP) décrite par l'interface ITCPSocket. Son rôle est de transformer les demandes formulées sur son interface IPOP3 en échange de trames réseau avec un serveur mail qui se conforme au standard POP3. Sur une socket TCP connectée au serveur (via **connect) sendLine** permet d'envoyer une ligne de texte au serveur et **readLine** (invocation bloquante) permet de lire une ligne de texte (réponse) envoyée par le serveur.



Question 2.5 : (1 point) Modélisez sur un diagramme de composant le composant CPOPProxy.

Question 2.6 : (1,5 pt) Modélisez à l'aide d'un diagramme de structure interne les composants instanciés sur la machine client.

Question 2.7 : (3 points) Modélisez l'interaction permettant à un utilisateur **déjà connecté**, de lister les messages, puis d'effacer le message 1 (on suppose qu'il existe). On utilisera un diagramme de séquence représentant tous les composants instanciés à la question précédente.

Question 2.8 : (1,5 points) Proposez à l'aide d'un diagramme de classes une conception détaillée pour la réalisation du composant Proxy.

Stub (2 points)

Question 2.9 : (2 points) On souhaite à présent réaliser un adaptateur CPOPStub qui permette au composant CServeurMail décrit en question 2.4 d'être conforme au standard POP3. Cet adaptateur peut être vu comme un Stub réseau pour le composant CServeurMail.

Il est instancié sur la machine serveur et attend les connexions sur une socket en mode serveur. On suppose l'existence d'un composant CServerSocket qui offre une interface IServerSocket dont les opérations (proches mais pas exactement symétriques de ITCPSocket) ne seront pas détaillées dans cet énoncé.

Il décode les trames réseau et délègue les traitements sur le CServeurMail.

a) Représentez ce composant sur un diagramme de composants.
b) Modélisez sur un diagramme de structure interne les composants instanciés sur la machine serveur.

SMTP (3 points)

Question 2.10 : (3 points)

a) Que manque-t-il au système pour être réellement opérationnel (par exemple pour permettre d'exécuter la séquence caractéristique identifiée en 2.3) ?
b) Proposez une interface ISMTP et ses opérations pour permettre de compléter notre conception.
c) Quel(s) composants parmi ceux déjà définis devraient logiquement réaliser/utiliser cette interface ?

Crédits : Certains textes de cet énoncé sont basés sur les descriptions des pages wikipedia dédiées à pop.