Auteur : Spinelli Isaia et

Simonet Yoann

Prof : [Dutoit Fabien](https://mse.hes-so.ch/consultation/horaires/?annee=2019&trimestre=1&id=8329&type=1)

Assist. : Christophe Greppin

Date : 11.10.2019

Classe : SYM-A

Laboratoire 2 : Protocoles applicatifs

Systèmes mobiles

Table des matières

[Introduction - 2 -](#_Toc21169614)

[Réponse aux questions - 2 -](#_Toc21169615)

[Conclusion - 6 -](#_Toc21169616)

[Difficultés rencontrées - 6 -](#_Toc21169617)

[Compétences acquises - 6 -](#_Toc21169618)

[Résultats obtenus - 6 -](#_Toc21169619)

## Introduction

Ce laboratoire propose une introduction aux techniques de programmation réparties asynchrones. Beaucoup plus complexes à maîtriser que les techniques synchrones, la programmation asynchrone est connue surtout dans le monde des interfaces utilisateurs.

Dans ce laboratoire nous allons illustrer l’utilisation de différentes techniques de protocoles asynchrones pour du mobile.

Pour ce laboratoire, un serveur applicatif accessible sur <http://sym.iict.ch/> à été mis à disposition. Plusieurs services sont définis sur ce serveur, entre autre le plus simple (http://sym.iict.ch/rest/txt) qui implémente un service écho, vous lui POSTez du texte (Content-Type: text/plain) et il vous l’envoie en retour accompagné de certaines informations sur le serveur.

## Manipulation

Pour réaliser cette manipulation, il faudra au minimum une petite application avec une première activité proposant 5 boutons permettant de lancer 5 activités implémentant les 5 points suivants.

La manipulation proposée ici implique une communication asynchrone avec un serveur, sur la base d'une méthode synchrone comme le protocole HTTP (à implémenter) qui aurait, par exemple, pour signature :

***public String sendRequest(String url, String request) throws Exception***

Il existe aussi des librairies permettant de simplifier le développement sur Android de la communication HTTP, comme par exemple Volley6 . Mais dans le cadre de ce laboratoire, nous souhaitons mettre en avant l’asynchronisme et les difficultés associées, il nous est demandé de ne pas utiliser de librairie « clé-en-main ». Celles-ci permettent souvent de faciliter la mise en œuvre des cas « simples », mais dans les cas plus complexes ou sortant de l’ordinaire (par exemple la compression des requêtes sortantes) elles peuvent compliquer le problème.

Dans le cadre de cette manipulation vous pouvez rester sur une solution simple « en mémoire », vous indiquerez toutefois dans votre rapport les limitations de cette façon de faire et proposerez des outils et techniques mieux adaptés (sans forcément réaliser l’implémentation). (page 8)

### Activité Asynchrone

### Activité Différée

Pour cette activité, il a fallu réaliser des requêtes différées. En l'absence de connexion avec le serveur, l'application fonctionne normalement, sans que l'utilisateur n'éprouve une gêne quelconque. Dès que la connexion avec le serveur est rétablie, les informations qui avaient été fournies par l'utilisateur sont transmises au serveur.

Pour faire ceci, lors d’une demande d’envoi d’une requête nous créons un thread si ce n’est pas déjà fait. Celui-ci checkera toutes les 5 secondes s’il y a une connexion à internet. Quand la connexion sera bien présente, il enverra toutes les requêtes précédemment envoyées. Donc quand il y aucune demande de requête, le thread ne prend aucune ressource CPU.

Afin de tester cette activité, nous avons coupé notre wifi et envoyé plusieurs requêtes. Ensuite, nous avons réactivé le wifi pour voir toutes les réponses des requêtes dans les LOGs.

Cette méthode pourrait être problématique si le réseau est stable et qu’on modifie plusieurs fois le même élément car l’ordre d’arrivée des requêtes n’est pas garanti.

On pourrait utiliser le multiplexage de toutes les connexions vers un même serveur en une seule connexion. Plus d’information sur le principe ici : <https://connect.ed-diamond.com/GNU-Linux-Magazine/GLMF-115/Multiplexage-des-connexions-SSH>.

### Activité Sérialisation

### Activité Compressé

### Activité GraphQL

## Questions

### Traitement des erreurs

*Les classes et interfaces SymComManager et CommunicationEventListener utilisées au point 3.1 restent très (et certainement trop) simples pour être utilisables dans une vraie application : que se passe-t-il si le serveur n’est pas joignable dans l’immédiat ou s’il retourne un code HTTP d’erreur ? Veuillez proposer une nouvelle version, mieux adaptée, de ces deux classes / interfaces pour vous aider à illustrer votre réponse.*

Comme dit dans la question, SymComManager et CommunicationEventListener ne gère aucun des cas d’erreurs mentionnés. En effet si le serveur n’est pas joignable ou s’il retourne un code http d’erreur.

Dans le premier cas, si le serveur n’est pas joignable l’utilisateur ne verra rien de spécial mise à part qu’aucune réponse arrive. En revanche, une exception sera levée. Afin de remédier à ça, il serait préférable de commencer par s’assurer que le téléphone soit bien connecté à internet. Ensuite, il serait possible d’utiliser le même principe que la méthode différée. Donc, d’attendre un moment en enregistrant la requête et d’essayer plus tard.

Dans le deuxième cas, si le serveur revoie un code d’erreur, il est possible de tester cela. Il y a deux cas d’erreurs possible les erreurs 4xx qui sont du au serveur et 5xx qui sont dû à l’utilisateur. Dans le premier cas, l’erreur est dû au serveur donc il serait possible d’appliquer la même méthode que lors d’un problème de connexion. Pour une erreur 5xx qui est donc dû au client, comme nous faisons dans ce laboratoire, il est possible de récupérer l’erreur et de l’afficher à l’utilisateur afin qu’il puisse prendre conscience de l’erreur et y remédier.

### Authentification

*Si une authentification par le serveur est requise, peut-on utiliser un protocole asynchrone ? Quelles seraient les restrictions ? Peut-on utiliser une transmission différée ?*

Cela peut dépendre de l’application mais en général si une authentification est nécessaire, toute activités suivant l’authentification dépendra de la réponse ce celle-ci. Donc, on protocole asynchrone ne serait pas nécessaire mais tout de même possible.

De plus une transmission différée peut être utiliser mais n’est clairement pas adapté à une authentification. En effet, lors d’une authentification nous souhaitons une réponse rapide et le différé n’assure pas ce principe.

L’une des restrictions d’une authentification est le temps de réponse qui se doit d’être relativement rapide afin que l’expérience utilisateur soit agréable. De plus, lors d’une authentification, des données confidentielles sont envoyées, il soit doit donc que la communication soit spécialement sécurisée.

### Threads concurrents

*Lors de l'utilisation de protocoles asynchrones, c'est généralement deux threads différents qui se préoccupent de la préparation, de l'envoi, de la réception et du traitement des données. Quels problèmes cela peut-il poser ?*

Contrairement aux asynctasks, les threads peuvent s’exécuter au même moment si plusieurs cœurs sont accessibles. Dans ce cas, il est important de prêter une attention particulière aux accès concurrents (sections critiques) si les threads se partagent des ressources ce qui est généralement le cas. De plus, l’ordre d’exécution des threads pourrait varier, il est donc important qu’une synchronisation se fasse afin qu’aucunes tâches s’effectuent avant qu’une dépendance ne soit pas respectée.

### Ecriture différée

Lorsque l'on implémente l'écriture différée, il arrive que l'on ait soudainement plusieurs transmissions en attente qui deviennent possibles simultanément. Voici deux possibilités :

1. Effectuer une connexion par transmission différée
2. Multiplexer toutes les connexions vers un même serveur en une seule connexion de transport.

### Transmission d’objets

1. *Quel inconvénient y a-t-il à utiliser une infrastructure de type REST/JSON n'offrant aucun service de validation (DTD, XML-schéma, WSDL) par rapport à une infrastructure comme SOAP offrant ces possibilités ? Est-ce qu’il y a en revanche des avantages que vous pouvez citer ?*
2. *L’utilisation d’un mécanisme comme Protocol Buffers9 est-elle compatible avec une architecture basée sur HTTP ? Veuillez discuter des éventuelles avantages ou limitations par rapport à un protocole basé sur JSON ou XML ?*
3. *Par rapport à l’API GraphQL mise à disposition pour ce laboratoire. Avez-vous constaté des points qui pourraient être améliorés pour une utilisation mobile ?*

### Transmission compressée

Quel gain peut-on constater en moyenne sur des fichiers texte (xml et json sont aussi du texte) en utilisant de la compression du point 3.4 ? Vous comparerez plusieurs tailles et types de contenu.

## Conclusion

### Difficultés rencontrées

Lecture de la donnée.

### Compétences acquises

### Résultats obtenus

Date : 10.11.19

Nom de l’étudiant : Spinelli Isaia et Simonet Yoann