Projet Encaisse

TPV-Mobile-Web

Analyse Fonctionnelle :
Communication réseau TCP entre le serveur et la caisse.

Aktaour
04/11/2010

Sommaire

But :

Nous sommes dans une ère de l'automatisation des développements. Nous faisons de plus en plus confiance à des protocoles de communication basée principalement sur les développements Internet.

Qui dit automatisation dit rendement accru. Mais attention, un gain de productivité au démarrage d'un projet peut vite devenir un calvaire si l'on ne maitrise pas toute la chaine.

Il est compréhensible de vouloir mettre en place ce genre de solution, mais ne nuit-elle pas à terme au processus d'intégration et de développement.

Il est préférable de réfléchir à une solution pérenne dont on maitrise le code de part en part plutôt que de privilégier des boites noires qui risquent de bloquer en production.

Protocole HTTP – Web services

Les web services se basent sur le protocole HTTP qui est lui même basé sur le protocole TCP.

Les web services se développent comme des petits pains, on rentre les paramètres, on compile et c'est cuit. Le problème est qu'on ne maitrise pas le temps de cuisson, le degré de chaleur et le four utilisé.

Pour des sites internet où la performance n'est pas primordiale, ce type de protocole est idéal.

Pour un logiciel de caisse, cela est incohérent voir même suicidaire, c'est un environnement qui se doit être stable et rapide.

Pour un logiciel gérant les données centrales, il peut être utilisé pour envoyer et recevoir des données à un système tiers. Mais pourquoi câbler deux fonctionnalités utilisées par le même logiciel avec du http si ce n'est pour donner le bâton pour se faire battre ?

Je ne suis pas pour ce protocole et ceci pour plusieurs raisons :

- Système ouvert
    - N'importe qui peut agir sur votre système d'information en y faisant n'importe quoi.
    - En général, il y a des centaines voir des milliers de fonctions ouvertes, qui pourrait garantir qu'un appel ne doit pas être consécutif à un autre pour fonctionner ?
    - Dès lors que quelqu'un peut agir librement sur votre système d'information, comment pourriez-vous garantir l'intégrité de vos données ?
- Protocole http.
    - Il est difficile et limité de transférer des fichiers binaires.
    - Si le web est fait pour fonctionner dans un écosystème très diversifié et indépendant de la plateforme c'est qu'il repose sur du TCP.
    - Une connexion TCP est ouverte à chaque demande puis fermée après réception de la réponse.
    - On ne connait l'état de disponibilité du serveur qu'au moment de l'appel.
    - Toujours à l'initiative du client vers le serveur. Le serveur ne peut envoyer de lui même des données, il ne fait que répondre aux demandes du client.
    - Ce protocole est lent surtout quand vous devez faire des appels successifs.
- Service Web
    - Une lourdeur incroyable
        - Lors des développements.
        - Quantité mémoire consommé ahurissante lors des appels.
        - Volumétrie dans les sources répercutées dans la taille de l'exécutable.
        - Volumétrie très importante de données transitant sur le réseau pour des appels qui ne nécessite théoriquement que quelques octets.
    - Le mystère
        - A part un expert, personne n'est capable de vous dire ce qui transite exactement sur le réseau.

        - Il est très difficile de lire et d'interpréter les flux envoyés.
    - Un système ouvert plutôt fermé
        - Un web service avec des objets complexes ne peut être lu par tous les langages de développement, il dépend fortement de comment le Framework génère son web service. En effet, un standard n'est pas forcément interpréter par tous de la même manière.

Protocole TCP

L'un des meilleurs et des plus fiables des moyens de communications. Il existe depuis la création des réseaux TCP/IP. Il assure l'intégrité des données envoyées et assure l'unicité dans l'ordre des paquets reçus.

Il est utilisé lors d'une connexion à une base de données distante, à un serveur Web, dans le SCO d'IBM.

Les inconvénients :

- Manque d'évolutivité dans l'implémentation du protocole pour un logiciel. En effet, TCP est décrit comme une succession de trame (données) circulant sur un réseau internet ou intranet. Chaque trame envoyée doit pouvoir être lue et interprétée par le client, hors pour cela, il faut que les trames soient de taille fixées à l'avance.
- Il faut pour chaque fonctionnalité publiée du logiciel en définir une ou plusieurs trames spécifiques en binaire. Ce qui est très fastidieux à créer et à maintenir.

Les avantages sont énormes :

- Une connexion persistante entre le client et le serveur.
- Une réponse instantanée du serveur et du client (hors temps de traitement de la demande).
- Multi plateforme.
- Envoie natif des flux binaires.
- Flux bi directionnel, client vers serveur ou serveur vers client.
- On envoie que ce dont on a besoin.

Imaginez que dans une version 1 d'un logiciel on a une trame de longueur 4 et de longueur 8 dans une version 2. Comment une version 2 peut communiquer avec une version1 sans décaler toutes les trames lues?

Exemple version 1 :

Exemple version 2 :

Dans les jeux vidéo réseaux ce n'est pas possible puisque les trames sont lues sur la base de longueurs fixes pour des questions de rapidité.

L'idée est de rendre les trames TCP indépendantes en termes de taille. La longueur des trames ne doivent pas être fixées à l'avance. L'idée est simplement de rajouter un champ en début de chaque trame pour signifier au client le nombre d'octet à lire.

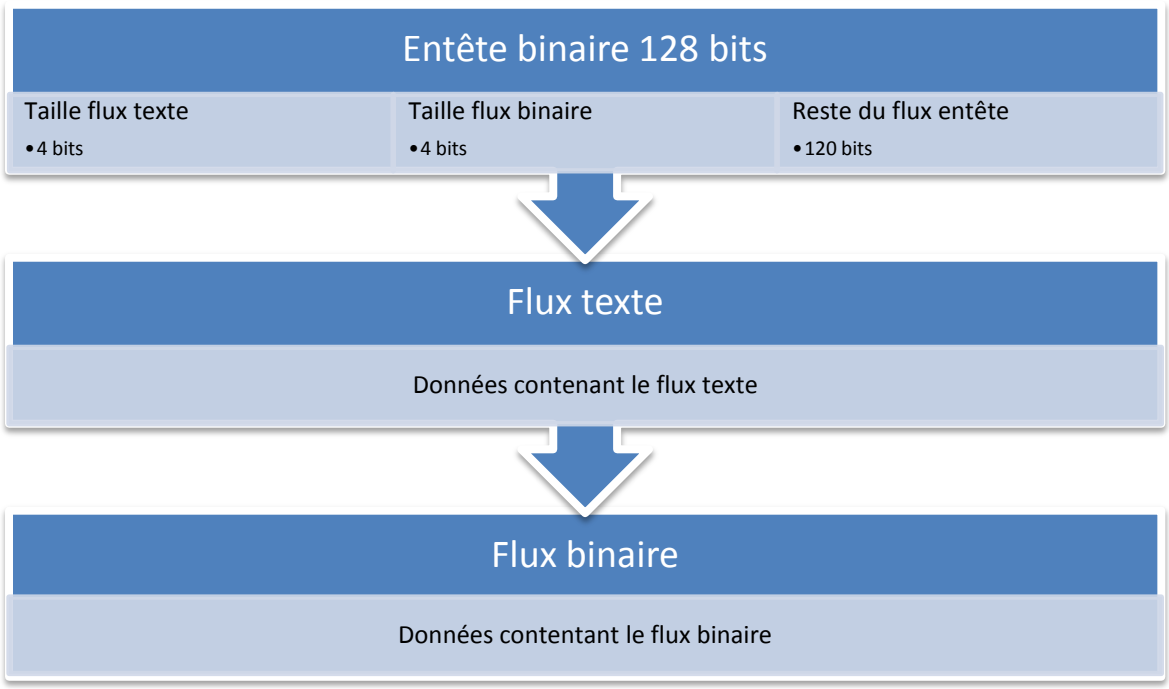
Des lors, un client d'une version 1 peut donc communiquer avec une version 2 et inversement.

Avec cette évolution, il est possible de lire la longueur et les données de la trame binaire envoyé par le client mais pas de les interpréter.

Protocole TCP Maison HDTP : HyperData Transfer Protocol.

Le protocole HDTP (nom donné arbitrairement) permet :

- Avoir les avantages de TCP sans le problème d'évolution et d'écriture de données
- Déterminer la taille des données à lire dans une trame.
- Envoyer en une trame :
    - Entête binaire.
    - Flux texte.
    - Flux binaire.



Ecarts :

Aucune communication réseau existante dans le projet Encaisse

Modifications :

Encaisse doit implémenter le protocole HDTP. Ce protocole sera la pierre angulaire du projet.
Il permettra d'envoyer et de recevoir des données, des fichiers, des actions à effectuer et de les exécuter en temps réel.

Protocole HDTP appliqué au projet

Le projet « Encaisse » utilise le protocole HDTP pour :

- Identifier chaque trame par :
  - Numéro de magasin
  - Numéro de caisse
  - Type d'action (exemple : recherche article)
  - Identifiant de transaction
  - Type de poste (Station, Serveur, Middle Office)
- Crypter ou non les données en RSA (clé public / clé privé).
- Envoyer des mises à jour.
- Envoyer des flux déshydratés.
- Envoyer des fichiers.
- Envoyer des données bi directionnelles entre le client et le serveur.
-