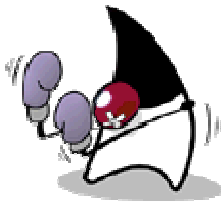
Laboratoire de Technologies de e-commerce (XML & Java) : projet "Inpres-Holidays"- suite

3ème Informatique de gestion 2012-2013

Claude Vilvens et Ludovic Kuty
- en collaboration avec Christophe Charlet



1. Préambule

Le contexte de ce laboratoire est le même que celui du laboratoire de Réseaux et technologies Internet, à savoir celui de "Inpres-Holidays" qui vise à la gestion informatique des activités de la société du même nom. Il conviendra donc de se référer à cet énoncé. Le présent document n'est que la première partie de l'énoncé de laboratoire de "Technologies e-commerce" et correspond à une première évaluation (en novembre 2012). Un second document (fourni vers fin octobre) contiendra la deuxième partie et correspondra à deux évaluations (une en décembre 2012 et une durant la session d'examen en janvier 2013).

2. Règles d'évaluation du laboratoire

A propos de l'évaluation du cours de Technologies de l'E-commerce, les règles d'évaluation suivantes seront utilisées par les enseignants de l'équipe responsable du cours.

1) L'évaluation établissant la note du cours de "Technologie de l'E-commerce" est réalisée de la manière suivante :

♦ théorie : un examen écrit en janvier 2011 (sur base d'une liste de questions fournies en décembre et à préparer) et coté sur 20;
♦ laboratoire : 3 évaluations (aux dates précisées ci-dessous), chacune cotée sur 20; la moyenne de ces 3 cotes fournit une note de laboratoire sur 20;
♦ note finale : **moyenne pondérée de la note de théorie (poids de 5/10) et de la note de laboratoire (poids de 5/10)**.

Cette procédure est d'application tant en 1ère qu'en 2ème session, ainsi que lors d'une éventuelle prolongation de session.

2) *Dans le cas où les travaux sont présentés par une équipe de deux étudiants,* chacun d'entre eux doit être capable d'expliquer et de justifier l'intégralité du travail (pas seulement les parties du travail sur lesquelles il aurait plus particulièrement travaillé).

3) *Dans tous les cas,* tout étudiant doit être capable d'expliquer de manière générale (donc, sans entrer dans les détails) les notions et concepts théoriques qu'il manipule dans ses travaux (par exemple: socket, signature électronique, certificat, etc).

4) En 2ème session et en prolongation de session, un **report de note** est possible pour chacune des deux notes de laboratoire ainsi que pour la note de théorie **pour des notes supérieures ou égales à 10/20**.
Toutes les évaluations (théorie ou laboratoire) ayant des **notes inférieures à 10/20** sont **à représenter dans leur intégralité**.

5) En prolongation de session, un **report de note** est possible **pour des notes supérieures ou égales à 10/20** :
- pour la note de laboratoire mais **seulement pour sa totalité**, pas pour l'une ou l'autre partie;
- pour la note de théorie.

Les évaluations (théorie ou laboratoire dans sa totalité) ayant des **notes inférieures à 10/20** sont donc **à représenter dans leur intégralité**. Mais de plus :
- pour l'examen théorique : les réponses écrites seront présentées et explicitées oralement à deux professeurs responsables du cours;
- pour l'examen de laboratoire : on gardera le même contexte, mais des fonctionnalités différentes pourront être demandées en lieu et place d'anciennes.

Le laboratoire de "Technologies e-commerce" comportera trois évaluations. La première partie des travaux (XML, protocoles, architecture serveurs et XSLT) sera **évaluée** par l'un des professeurs du laboratoire **à partir du 5 novembre 2012** (avec rentrée d'un dossier papier limité - le délai est à *respecter* impérativement).

La deuxième partie de ces travaux (statistiques avec JFreechart, Mail, complémnets WEB …) sera **évaluée** par l'un des professeurs du laboratoire **à partir du 3 décembre 2012** (avec rentrée d'un dossier papier limité - le délai est toujours à *respecter* impérativement).

La troisième partie (J2ME, SSL, applets sécurisées, Java Web Start, Struts, …) sera évaluée lors de l'examen de laboratoire en janvier-février 2013 (le dossier papier n'est plus nécessaire).

Les travaux de l'évaluation 1

3. La création des menus des restaurants : XML et arbre DOM (1)

Les villages de vacances de Inpres-Holidays proposent bien évidemment de somptueux repas à leurs habitants. Pour ne pas faire de jaloux, ce sont les mêmes repas journaliers qui sont proposés dans tous les restaurants. Le responsable général des restaurants de la chaîne compose chaque jour un menu avec ses variantes (le menu-bar géant self-service est proposé chaque jour et son contenu ne varie pas – il n'intervient donc pas ici). Dans chaque village, le chef de cuisine du restaurant reçoit chaque soir, à sa demande, la composition du menu du lendemain, ceci sous forme d'un fichier XML.

Le responsable général dispose donc d'une application fenêtrée **Applic_Head_Food** qui lui permet de confectionner un menu grandiose à partir des denrées alimentaires en stock (la gestion de celles-ci relève d'un autre responsable et ne sera pas traitée ici). Au démarrage de l'application, la liste des denrées disponibles pour chaque type de plat est téléchargée d'un serveur **Serveur_Restaurants**. Ce serveur écrit en Java est un simple serveur monothread qui attend sur un port PORT_HEAD et qui utilise le protocole VSMEAP (Very Simple MEAls management Protocol) dont les commandes, côté responsable général, sont :

protocole **VSMEAP** (responsable général)		
commande	**sémantique**	**réponse éventuelle**
LOGINHEAD	le responsable ou l'un de ses délégués veut se connecter *paramètres* : nom et mot de passe	oui ou non + cause
DOWPROD	DOWnload PROducts : demande le téléchargement de la liste de denrées *paramètres* : une indication sur le type de plats concernés (PLATS, DESSERTS, ALL, …)	oui + liste ou non + pourquoi (par exemple : livraison de denrées en cours – réessayer plus tard)
UPMENU	UPload MENU : envoie le menu composé au moyen de l'application *paramètres* : une indication sur le type de plats concernés (PLATS, DESSERTS, ALL, …)	oui ou non + pourquoi (par exemple : livraison de denrées en cours – réessayer plus tard)
SENDMENU	SEND MENU : envoie un menu contenu dans un fichier XML (créé de manière quelconque) *paramètres* : une indication sur le type de plats concernés (PLATS, DESSERTS, ALL, …)	oui ou non + pourquoi (par exemple : nom de fichier invalide)

Le GUI a un aspect du type suivant :

Le rôle de ce GUI est en fait de construire en arrière plan, au moyen d'un arbre DOM, un fichier XML (élément "menu") qui peut comprendre, outre le nom donné au menu (par exemple "Délices de Neptune" ou "Au bonheur du cochon") et le nom de la vedette du jour honorée par ce menu (optionnel), les 3 éléments principaux "entree", "plat", "dessert". Pour chacun de ces items, il faut

♦ absolument fournir
- le prix global pour les clients qui ne sont pas en "all-inclusive"
- de 1 à 4 propositions, chacune comportant un nom, un identifiant, une liste d'un certain nombre d'ingrédients, le fait d'être chaud ou froid
♦ éventuellement fournir
- le montant d'un supplément éventuel à payer;
- le risque d'allergie consécutif à un ingrédient.

Lorsque le responsable appuie sur le bouton "Valider menu", l'arbre DOM est transformé en fichier XML et est envoyé par réseau au serveur Serveur_Restaurants : c'est la commande UPMENU du protocole.

4. Le traitement des menus reçus : parsing SAX

Le chef de cuisine de chaque village (ou son délégué) dispose d'une application fenêtrée **Applic_Village_Food** qui lui permet de demander le menu construit par le responable pour la journée du lendemain. Ce menu est, en principe, disponible dès midi. L'application converse avec le serveur avec d'autres commandes du protocole VSMEAP (Very Simple MEAls management Protocol), cela sur un port PORT_MENU selon un modèle multithread en threads à la demande :

protocole **VSMEAP** (chef de cuisine d'un village)		
commande	**sémantique**	**réponse éventuelle**
LOGINVILLAGE	le chef ou l'un de ses délégués veut se connecter *paramètres* : nom et mot de passe	oui ou non + cause

DOWMENU	DOWnload MENU : demande le téléchargement du menu prévu pour le lendemain *paramètres* : une indication sur le type de plats concernés (PLATS, DESSERTS, ALL, …)	oui + fichier XML ou non + pourquoi (par exemple : pas encore disponible – réessayer plus tard)

Lorsque le fichier XML est reçu, il est parsé avec SAX - si des erreurs sont détectées, le client envoie au serveur la commande :

protocole **VSMEAP** (chef de cuisine d'un village)		
commande	**sémantique**	**réponse éventuelle**
WRONGMENU	WRONG MENU : signale que le menu reçu n'est pas conforme à la structure attendue *paramètres* : une indication sur le type de problème (CHECK, VERIFY_DTD)	oui - ACK

Mais nécessité fait force de loi : si certains ingrédients sont manquants dans le village, force sera de supprimer le plat correspondant des propositions : un nouveau fichier XML sera donc créé en ne retenant que ce qui est possible de préparer (le réapprovisionnement est assuré par une autre voie que l'on ne traitera pas ici). A priori, ce nouveau fichier sera créé au moyen de la technologie DOM.

5. La publication de la réponse : XSLT

Il restera alors à l'application **Applic_Village_Food** à appliquer une transformation XSLT au moyen d'un fichier XSL pour générer

1) une page HTML de présentation claire du menu pour le grand public; cette page sera placée dans le répertoire ROOT d'un serveur Tomcat : le voyageur n'aura plus qu'à utiliser son browser pour la visualiser;
2) trois fichiers contenant seulement les tags HTML de présentation des entrées, des plats principaux et des desserts; une servlet **Servlet_Food** utilisera ces trois fichiers pour répondre à un client qui la sollicite avec son browser : elle fournit en plus des indications météos (trouvées dans une base de données BD_METEO – à construire mais de manière élémentaire) et le nom des GEOs disponibles ce jour (information trouvée dans la base BD_HOTELS déjà connue par l'énoncé de "Réseaux et technologies Internet").

Les travaux de l'évaluation 2

1. Data mining et informations statistiques

Le serveur **Serveur_Activites**, évoqué dans la première partie, va se voir doté de fonctionnalités complémentaires importantes. En fait, il attend aussi sur un port PORT_STAT des requêtes formulées par les analystes marketing d'Inpres-Holidays. Ceux-ci manipulent une application **Applic_Data_Mining**, dont le rôle est de demander le traitement des informations disponibles dans BD_HOTELS. Ceci sous-entend que la base de données contient des informations supplémentaires, comme par exemple la date de naissance ou la nationalité des voyageurs (afin, par exemple, d'établir une relation activités-âge ou activités-nationalité).

On ici utilise le protocole **HIDP** (**H**olidays **I**nformation and **D**ecision **P**rotocol) dont les commandes sont :

protocole **HIDP**		
Commande	**Sémantique**	**réponse éventuelle – résultat attendu**
LOGIN	démarrage de l'application : un analyste se fait reconnaître *paramètres* : nom, password	oui ou non – validation au moyen d'un digest dans la base BD_ HOTELS
GET_STAT_DESCR_ACTIV	demande du nombre d'inscriptions, pour une saison donnée (= les mois de mai à octobre d'uen année donné), à l'une des activités "cracra-hontah", "extreme trek" et "orgiac island" par quinzaine, avec calcul de la moyenne et l'écart-type *paramètres* : l'année et l'activité	les données brutes et les paramètres statistiques demandés
GET_GR_ACTIV_COMP	demande d'un graphique statistique réalisé à l'aide de la librairie **JFreechart** : diagramme sectoriel ou histogramme pour le nombre d'inscriptions pour une saison donnée aux trois activités "cracra-hontah", "extreme trek" et "orgiac island" *paramètres* : l'année, le type de graphique = SEC, HIST	histogramme comparé ou diagramme en camembert
GET_GR_ACTIV_EVOL	demande d'un graphique statistique réalisé à l'aide de la librairie **JFreechart** : diagramme linéaire montrant l'évolution du nombre d'inscriptions pour une saison donnée aux trois activités "cracra-hontah", "extreme trek" et "orgiac island"	histogramme comparé ou diagramme en camembert

	paramètres : l'année	
GET_STAT_INFER_ACTIV	estimation de l'âge moyen et de la fourchette de distribution des participants, pour une saison donnée, à l'une des activités "cracra-hontah", "extreme trek" et "orgiac island" *paramètres* : l'année et l'activité	les estimations de paramètres demandés
GET_GR_2D_DEPENSES	demande d'un graphique statistique réalisé à l'aide de la librairie **JFreechart** : nuage de points et paramètres de régression corrélation pour un échantillon de données de type age-dépenses des voyageurs pour l'ensemble des activités d'un jour (ski nautique, cours de survie, initiation aux orgies romaines, …) pratiquées durant un séjour. *paramètres* : effectif de l'échantillon à prélever	nuage de points, coefficient de corrélation, paramètres de régression (supposée linéaire)
TEST_COMP	demande un test d'hypothèse de comparaison des moyennes des dépenses pour les activités d'un jour par séjour pour deux nationalités différentes de voyageurs *paramètres* : les nationalités, le seuil utilisé	différence significative ou non

Pour simplifier, on supposera qu'un voyageur ne réalise qu'un seul séjour par saison (d'ailleurs, au prix où c'est … ;-))

Remarque

La dernière commande suppose que l'on dispose d'une librairie statistique fournissant les probabilités et les quantiles de la loi normale (par exemple). On pourra utiliser pour cela la Apache Commons Mathematics Library (téléchargeable sur http://commons.apache.org/math/download_math.cgi - la version 2.2 suffit). Un exemple d'utilisation est :

StatistiquesWithApacheCommons.java

```
package statistiqueswithapachecommons;

import org.apache.commons.math.MathException;
import org.apache.commons.math.distribution.NormalDistribution;
import org.apache.commons.math.distribution.NormalDistributionImpl;
```

```
public class StatistiquesWithApacheCommons
{
   private static NormalDistribution d;

   public static void main(String[] args) throws MathException
   {
      d = new NormalDistributionImpl(1000, 100);
      System.out.println(d.cumulativeProbability(1200));
      d = new NormalDistributionImpl();
      System.out.println(d.inverseCumulativeProbability(0.95));
   }
}
```

Résultat :

```
0.9772498680518208
1.6448536269514737
```

On pourra comparer la dernière valeur obtenue avec ce qu'on lit dans une table classique de la loi normale (par exemple : http://www.agro-montpellier.fr/cnam-lr/statnet/tables.htm).

2. SMTP/POP3 en Java et en C/C++

a) D'une part, il s'agit tout d'abord de développer une petite application **Applic_Mail** permettant de traiter le courrier électronique relatif aux informations échangées par les responsables de villages de vacances et les divers gestionnaires d'activités ou de matériel. Plus précisément, elle comporte une fonctionnalité GUI **Java** de gestion classique d'e-mail (type "**mini-outlook**"), permettant d'envoyer et de recevoir un mail
- soit simple de type texte;
- soit composite avec des pièces attachées qui sont des images gif ou jpg, ou encore un paquet de bytes (tiens ? comme un digest ?) ou même des objets sérialisés (instance de la petite classe **PieceAttachee** qui se limite à encapsuler les composantes de l'information, selon sa nature).

L'utilisateur devrait idéalement être prévenu dans un délai de 5 minutes de l'arrivée d'un nouveau message. A cet effet, l'application utilisera un thread de polling qui "interrogera" périodiquement la boîte aux lettres.

b) D'autre part, lorsque le **Serveur_Activites** traite avec succès une commande ACKACTFUN ou BTREKFUN, un e-mail de confirmation est envoyé au client. Plus précisément :

* pour ACKACTFUN : le mail est un simple mail de texte qui récapitule la réservation;
* pour BTREKFUN : outre un récapitulatif, le mail transporte en pièce attachée

- ♦ un document reprenant le règlement de l'activité : pour éviter les réclamations, le client devra renvoyer un mail avec son numéro de carte d'identité indiquant qu'il accepte ce règlement à l'adresse d'un autre membre du groupe de laboratoire;
- ♦ une carte pour cracra-hontah" et "extreme trek";
- ♦ un objet Java contenant la phrase qui sert de sésame pour participer à la séance choisie de "orgiac island";
- ♦ une chaîne de bytes représentant le digest du nom du client.

c) Dans un esprit similaire, le serveur **Serveur_Villages** envoie un e-mail de confirmation aux responsables des villages de vacances lorsqu'une commande BMAT a été prise en compte avec succès.

3. Amélioration de Web_Applic_Reservations

Il s'agit ici de faciliter les réservations en permettant aux voyageurs

3.1 L'internationalisation

L'internationalisation va être prise en compte afin de permettre aux clients étrangers de fréquenter sans difficulté le shop du ferry. On imagine donc que l'application WEB démarre par une JSP dédiée au **login et choix de langues.** Le client va contacter le site WEB en invoquant le Java Server Page **Contexte.jsp** : il reçoit ainsi une page WEB de type formulaire – en fait, c'est une **applet**. Ce formulaire comporte une boîte de listes contenant

- la liste des langues utilisables sur le site;
- des zones d'entrée pour son nom, son code de réservation et son adresse e-mail;
- un bouton "Connect".

Les langues utilisables se trouvent dans une table *Langues* de la base BD_HOTELS. L'internationalisation sera réalisée pour les JSPs suivants, et pour les messages courants, en utilisant la technique des "bundles".

Toutes les pages suivantes utiliseront la balise personnalisés sans corps **<status:date-heure>** pour réaliser dans chaque JSP l'affichage (dans la langue choisie) de la date et heure.

3.2 Le choix des réservations

L'affichage des pages catalogues du site de Inpres-Holidays se réalise à présent au moyen d'une balise personnalisés sans corps <**status:show**>.

3.3 Le paiement des réservations

La page de confirmation-paiement des achats est une JSP dédiée à la **confirmation des réservations et l'introduction des données complémentaires**. Cette JSP utilise la balise avec corps **<status:list>** qui affiche dans le JSP la liste des achats effectués. Mais surtout, le paiement, après introduction du numéro de carte de crédit, va provoquer la mise en route d'une séquence authentifiée (au moyen du digest du code PIN de la carte) : les informations de paiements (nom client, numéro de carte, montant) sont envoyées à un **Serveur_Banque** sur son port PORT_PAY et, bien sûr, le serveur va vérifier si le paiement peut être effectué ou pas au moyen d'une base de données BD_COMPTES. Le protocole qu'il convient ainsi de définir est désigné par **BDP** (**B**ank **D**ialog **P**rotcol). En cas de succès, une facture est envoyée par mail au client. Cette facture va servir de preuve au client pour aller rechercher ses achats au shop du ferry.

3.4 La réponse et la clôture de la transaction (réussie ou non)

Quelle que soit l'issue de la transaction, le client reçoit une dernière JSP visant la **clôture des opérations**. Outre le résultat, cette page affiche le temps d'utilisation total de l'application au moyen de la balise sans corps **<status:using-time>.**

Les travaux de l'évaluation 3

4. J2ME

L'idée est d'élargir le plus possible les activités d'Inpres-Holidays, et donc de permettre au grand public d'effectuer des réservations pour les motels et les villages au moyen de son smartphone.

Le serveur **Serveur_Reservations** doit donc aussi satisfaire ces requêtes en considérant que l'application **Mobile_Applic_Reservations** tournant sur le smartphone est capable d'utiliser le protocole RMP (sans LROOMS). Cependant, cette possibilité n'est offerte qu'aux clients qui sont déjà connus de la société, autrement dit à des voyageurs qui ont déjà séjournés dans un des villages d'Inpres-Holidays et qui ont reçu à cette occasion un mot de passe spécial. En fait, le Serveur_Reservation va dérouter les demandes non signées (ce que demande RMP) sur un module intermédiaire **Serveur_Wellknown_Voyagers** qui vérifiera ce mot de passe et qui en cas de succès, et comme la servlet de l'application Web de réservations, signera la requête au moyen d'une clé privée particulière qui lui a été attribuée : la requête ainsi signée repartira vers Serveur_Reservations.

5. Les communications sécurisées par SSL

Dans le cas d'une commande PROOM de paiement d'une réservation, le serveur Serveur_Reservations va entreprendre deux actions supplémentaires, toutes deux sécurisées au moyen de TLS/SSL :

- d'une part, il va vérifier le numéro de carte de crédit (et le cryptogramme associé) du client payeur auprès du serveur de l'organisme émetteur de la carte (pour simplifier, il n'y en a que deux possibles : VILVISA et MASTERKUTY – ce qui fera donc deux serveurs distincts);
- d'autre part, en cas de succès, il demandera le virement de la somme à payer à sa banque : pour cela, il fournira le numéro de compte de Inpres-Holidays au serveur de VILVISA et MASTERKUTY (selon le cas) qui contactera le serveur de la banque d'Inpres-Holidays (vous suivez ;-) ?); une fois le virement effectué, la transaction est mémorisée dans une base de données et un e-mail sera envoyé à Inpres-Holidays (plus précisément à son comptable en chef assoocié à l'adresse d'un autre membre du groupe de laboratoire).

On aura donc besoin de deux protocoles de communication, respectivement

protocole **GIMP** (Gimme gImme giMme Protocol) [Serveur_Reservations → Serveur crédit]		
commande	**sémantique de la requête**	**réponse éventuelle**
PAY_FOR_CLIENT	pour vérifier la carte de crédit *paramètre* : nom du client, numéro de carte de crédit et cryptogramme associé, référence de la réservation	VERIF_CARD_SUCCESSFULL ou VERIF_CARD_FAILED,

et

protocole **MAMP** (Money Allways Money Protocol) [Serveur crédit → Serveur banque]		
commande	**sémantique de la requête**	**réponse éventuelle**
TRANSFER_POGN	pour transférer le montant de la réservation *paramètre* : somme, nom du client, référence de la réservation	VERIF_INT_SUCCESSFULL ou VERIF_INT_FAILED, réponse signée par le serveur

Les certificats nécessaires seront fabriqués avec OpenSSL et/ou la programmation. Les keystores seront visualisés avec Keytool IUI.

En pratique : Pour ces points 4 et 5, il convient d'analyser les trames au moyen d'un sniffer quelconque. En particulier, on se demandera si une nouvelle commande provoque un nouvel handshake ou pas.

6. Java Web Start

On demande que l'application fenêtrée développée dans la première partie du laboratoire (XML – les menus) devienne une application Java Web Start.

Est-ce également envisageable pour l'application Applic_Mail du point 2 ci-dessus ?

7. Utilisation du framework Struts pour Web_Applic_Reservations

Le site évoqué au point 3 devrait à présent évoluer, quant à son application Web, en une version 2.0 utilisant les mécanismes de la plate-forme Struts. Quels changements faut-il apporter ? Peut-on les qualifier de fondamentaux ou de légers ?

8. Les applets signées

On termine comme on a commencé ;-) Les applets du premier point du laboratoire de "Réseaux et technologies Internet" sont à présent des applets signées par la CA loacl InpresHolidayRoot (à inventer).

Soyez créatifs et imaginatifs … mais restez rationnels et raisonnables, notamment quant aux informations utilisées …

s: CV & LK – avec le concours de *C.C.*