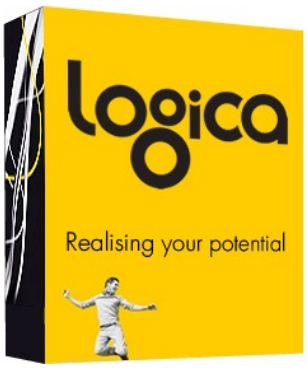
Baptiste CARLIER
Stage IUT
31 Mars 2008 – 20 Juin 2008

Tuteur Entreprise :
M. Nicolas Chapurlat
Tuteur Pédagogique :
M. Jean-Marie Thibaut

Tests de montée en charge d'applications JBoss Seam



Logica - Releasing your potential
Immeuble Mercury
30, Place Salvador Allende
59 650 Villeneuve d'Ascq
Tel: 03 59 56 60 00

IUT 'A' - USTL
Département Informatique
Boulevard Paul Langevin - BP 179
59 650 Villeneuve d'Ascq
Tel: 03 59 63 22 22

Remerciements

- Je souhaite tout d'abord remercier mon tuteur pédagogique, **M. Jean-Marie Thibaut**, mais aussi mes enseignants durant ces deux années d'études, le reste de **l'équipe du Département Informatique de l'IUT** et **mes camarades** de DUT auprès de qui j'ai beaucoup appris aussi.

- Je tiens aussi à remercier **M. Nicolas Chapurlat**, mon tuteur entreprise, qui a su m'aider et me conseiller quand j'en avais besoin. Il est resté très à l'écoute et a su me faire confiance.

- Je remercie aussi toute l'équipe de **Logica** et **le plateau Legrand/Leroy Merlin** qui m'ont permis d'en apprendre plus sur le fonctionnement d'une l'entreprise, ainsi que **la cellule architecture** pour leurs conseilles et leurs avis. Je n'oublie pas de mentionner **les autres stagiaires** avec qui les échanges ont été très intéressants.

- J'aimerais aussi remercier **ma famille et mes amis** qui ont compris quelle importance ce travail avait pour moi.

Résumé

J’ai effectué mon stage dans la société de service Logica qui m’a placé au sein de la cellule Architecture de l'agence Nord, plus précisément sur l'étude d'une nouvelle solution dans la création d'applications web, JBoss Seam sur J2EE. Cette technologie dépassant mes compétences, il m'a fallut d'abord l'étudier pour ensuite modifier ou réaliser quelques applications simples. Après cela, je me suis concentré sur un outil qui m'a permis de tester la capacité à résister à de forte charges de ce type d'applications.

Abstract

I completed my internship in the service company Logica, which placed me on the Architecture cell, more specifically on a study of a new solution of web application creation, JBoss Seam written in J2EE. This technology surpassing my skills, at first I owed to study it for afterwards modify or produce some simple applications. After that, I had to focus on a specific tool which would allow me to test the load this type of applications can support.

Sommaire

Introduction.....1

I. Présentation de l'entreprise Logica.....2

A. Une entreprise européenne.....3

B. En France et à la direction opérationnelle Nord 4

C. La cellule architecture.....5

II. Aspect technique.....6

A. Étude de J2EE et JBoss Seam.....7

1. J2EE.....7

2. JBoss Seam.....8

B. Les tests de montée en charge.....11

1. Qu'est-ce qu'un test de montée en charge ?.....11

2. Quel programme utiliser ?.....12

3. Mes premiers essais.....12

4. Ma documentation.....13

5. Le test final.....14

C. Mes conclusions.....14

III. Ma place dans l'entreprise.....16

A. Difficultés rencontrées.....17

B. Un apport mutuel.....17

1. Une réponse à une problématique.....17

2. L'étude de Seam.....18

C. Ce que je retiens du stage.....18

1. Mes premiers pas dans une SSII.....18

2. Le monde du travail.....19

Conclusion.....21

Annexes.....22

Glossaire.....32

Bibliographie.....33

Introduction

Dans le cadre de la dernière année de DUT Informatique à l'Institut Universitaire et Technologique de l'université Lille1, un stage de deux mois et demi en entreprise est obligatoire. J'ai effectué ce stage de fin d'études à la direction opérationnelle Nord de l'entreprise Logica. J'ai dû étudier les mécanisme d'un framework applicatif nommé *JBoss Seam* édité récemment.

J'ai rejoint la cellule architecture conduite par M. Pascal Stalin, architecte en informatique. M. Nicolas Chapurlat, mon responsable au sein de Logica, avait commencé l'étude de ce framework l'année précédente. Le but de ce stage était d'obtenir des informations sur JBoss Seam quant à la tenue d'une application web conçu avec cet outil face à une montée en charge.

Je commencerai par vous présenter la société Logica, issue de la fusion récente de LogicaCMG et Unilog puis je vous exposerai les étapes par lesquelles je suis passé lors de cette étude. En travaillant en autonomie, mon rôle a été dans un premier temps de comprendre le fonctionnement de Seam et donc d'une technologie qui m'était peu connue J2EE. J'ai ensuite du prendre en main un nouvel outil de tests nommé NeoLoad pour après ça pouvoir satisfaire ma problématique initiale : comment réagit une application classique de Seam lors d'une montée en charge ? Enfin je vous exposerai ce que j'ai pu retenir de ce premier stage en entreprise.

Chapitre 1

Présentation de l'entreprise Logica

A. Une entreprise européenne

Logica est une entreprise européenne de services en ingénierie informatique (SSII). Une des entreprises leaders dans son secteur au niveau européen, le groupe Logica rassemble plus de 40 000 personnes dans 41 pays.

Historiquement **Logica est une entreprise d'origine britannique qui a vue le jour en 1969**. Elle est le résultat de plusieurs fusions avec d'autres entreprises. En 1995, *CMG* est coté en bourse et fusionne avec Logica en 2002. En 1985 *WM-data* entre en bourse et se fera racheter en 2006.

Une année avant la création de Logica, en 1968, l'entreprise « *Informatique et Entreprise* » était créée par 5 fondateurs se regroupant ensuite en 1982 sous le nom d'*Unilog*. Société française, elle fut introduite en bourse en 1988. Après quelques années, en 2001 Unilog s'implante aux Royaume-Uni. Le groupe *LogicaCMG* – issu de la fusion de Logica et de CMG cité précédemment – décide donc de faire une OPA amicale sur Unilog. **C'est ainsi qu'Unilog devient en 2006 « *Unilog, a LogicaCMG company* »**.

Certes très long, ce nom de compagnie contenant à la fois « Unilog » et « Logica » était pratique pour permettre aux clients de retenir que la fusion s'opère. Ainsi, quelques semaines avant le début de mon stage, **le 27 Février 2008, Unilog a changé de nom pour Logica**. Le groupe adoptera cette nouvelle marque à l'échelle internationale.

Une grande entreprise comme Logica **opère donc dans plusieurs secteurs d'activité** :

- Secteur public
- Énergie
- Finance
- Télécommunication
- Industrie, distribution, transports
- Aéronautique
- Agroalimentaire
- Banque & Assurance
- Défense
- Pharmacie

B. En France et à la direction opérationnelle Nord

Logica réunit 8 500 collaborateurs en France répartis sur 20 agences. Il y avait 350 employés à la DO Nord en 2007.

La direction opérationnelle intervient dans les secteurs d'activités cités précédemment. Dans le Nord, les clients du groupe sont regroupés en 3 catégories appelées aussi SBL* :

- La grande distribution
- L'industrie
- La finance et le service public

Les principales enseignes françaises de la **grande distribution** étant installées dans la région, ces entreprises sont les principaux clients de la société. Logica compte parmi ces grands comptes une partie des enseignes du groupe Auchan (Auchan, Leroy Merlin, Décathlon, …) mais aussi d'autres grands noms du secteur de la distribution française ou spécialistes de vente par correspondance (Castorama, 3 Suisess, Damart).

Les comptes du secteur **industrie** sont représentés par de grands groupes internationaux comme Arcelor-Mittal, Roquette et ainsi que de grands groupes français.

Le troisième grand secteur est le secteur de **la finance et du service public**. Ce secteur

* Sector Business Line

couvre le secteur bancaire, les organismes de crédit et d'assurance mais également les organismes publics tels que Voies Navigables de France ou la SNCF. Les grands noms de ce secteur sont : Banque Populaire, ING Direct, Finaref, Cofidis, La Mondiale, Swiss Life, GAN, ou encore AGF.

Sur les **350 employés de la direction opérationnelle Nord**, environ la moitié travaille dans les locaux de l'agence sur des projets en forfait et l'autre moitié réalise des prestations en régie chez les différents clients de la région.

C. La cellule architecture

L'objet de mon stage m'a permis d'incorporer **la nouvelle cellule architecture** de Logica. Créée depuis peu de temps, cette cellule s'efforce d'avoir un esprit de collaboration en interne et externe.

Le responsable de cette cellule, Pascal Stalin, s'emploie à faire participer de façon collégiale les architectes entre eux dans le but de pouvoir être plus efficace. Afin de se faire mieux reconnaître par l'entreprise, la cellule a mis en place sur Internet un **workspace**[*]. Cet outil permet de fluidifier les connaissances. Chaque membre de la cellule a accès à ce workspace et peut y ajouter des fichiers et des commentaires personnels sur leurs études et leurs travaux. Cela permet de **capitaliser les données** et donc de pouvoir s'y retrouver plus facilement. C'est très pratique pour archiver des informations à propos de telle ou telle technologie. Lorsqu'il est mis à jour régulièrement, ce workspace permet aussi de connaître les emplois du temps de chacun : un nouveau **gain de temps dans l'organisation** d'entretiens ou réunions. Une **liste de diffusion** a aussi été mise en place. Aussi efficace et plus expéditif que le workspace, cette liste permet à chacun de demander des conseils, des points de vue, de présenter rapidement des résultats.

La première réunion de la cellule début Avril aura aussi permis à chacun d'exposer leurs idées à propos de l'intégration de cette cellule architecture dans l'entreprise et des missions qu'elle se devait d'accomplir. En outre il a été question de la création d'un framework spécial conçu pour Logica. Projet très novateur pour l'entreprise, ce framework pourrait être gratuit mais, étant conçu par Logica, le support technique qu'elle apporterait serait le meilleur du marché pour cet outil.

* Site web d'organisation de travail en groupe

Chapitre 2

Aspect technique

A. Étude de J2EE et JBoss Seam

J'ai donc intégré la cellule architecture dans le but de tester le framework applicatif JBoss Seam pour permettre à Logica de voir si cet outil peut être commercialisable. Les architectes souhaiteraient l'utiliser pour créer des applications web pour les clients.

1. J2EE

JBoss Seam utilise la technologie **J2EE 5 – Java 2 Platform, Entreprise Edition**. Java représente plus qu'un simple langage de programmation, car il englobe une technologie de machine virtuelle qui permet aux programmes compilés de fonctionner sans modifications sur des machines différentes. Il inclut aussi des outils de compilation, d'analyse, de débogage et de déploiement des programmes Java et d'autres composants. Cette plate-forme est plus appropriée pour le développement de sites web (*e-commerce* principalement en entreprise) que les deux autres plates-formes Java sous-jacentes J2ME et J2SE

Lors de ma toute première semaine de travail, j'ai donc dû commencer par les bases : j'ai installé la SDK* 6 de Java sur ma machine. Je pouvais ainsi compiler, exécuter, déboguer des programmes Java.

J2EE fonctionne avec trois grandes catégories d'API**. En généralisant, je pourrais dire qu'une API est une librairie.

- Les **composants** :
  Ils offrent une approche modulaire au niveau de l'architecture d'un projet. On distingue les **composants web** (*Servlets*, *JSP*), qui touche à la partie présentation, des **composants métier** (*EJB****).

- Les **services d'infrastructure** :
  Ils sont utiles pour, comme le nom l'indique, s'occuper de l'infrastructure du programme.

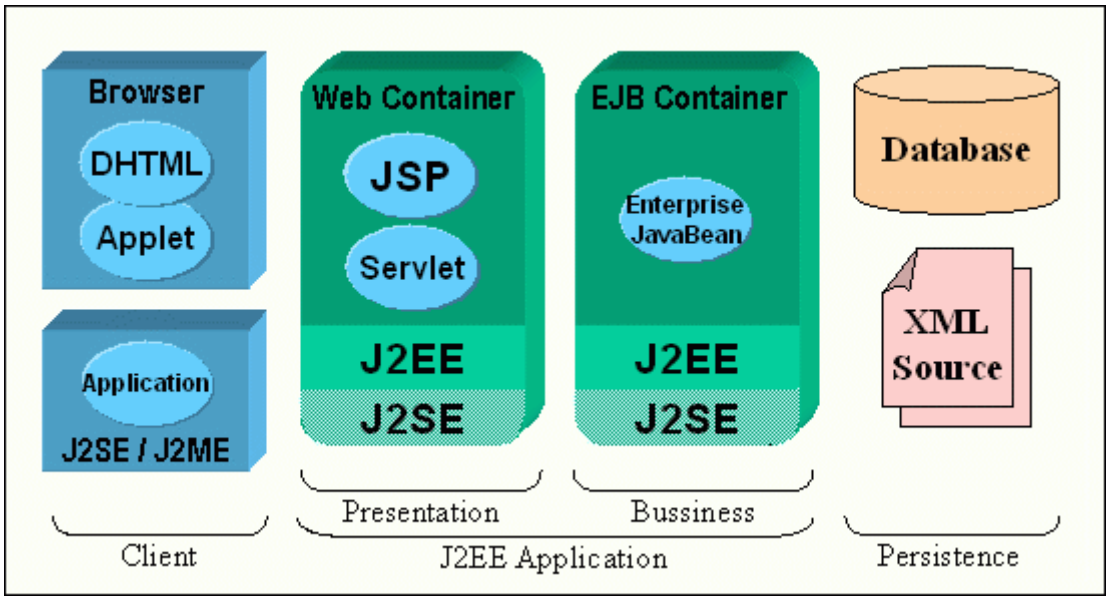
* Software Development Kit

** Application Programming Interface

*** Entreprise Java Beans

JDBC (Java DataBase Connectivity) permet l'accès aux bases de données relationnelles ; JNDI (Java Naming and Directory Interface) est une API d'accès aux services de nommage et aux annuaires ; JPA (Java Persistance API) est une API qui normalise le mapping entre les objets et les bases de données relationnelles ; …

- Les **services de communication** :
  Ils gèrent toutes les communications système : JAAS (*Java Authentication and Authorization Service*) permet l'authentification des utilisateurs et l'association de permissions ; JavaMail qui permet l'envoie de courriel ; …



Simplification du fonctionnement de J2EE

J'ai aussi dû télécharger et installer quelques outils nécessaires. Il m'a fallu mettre en place sur ma machine un serveur *JBoss AS* pour faire tourner les applications J2EE. J'ai aussi installé l'environnement de développement intégré *Eclipse Europa* ainsi que l'outil de développement *Apache Ant* facilitant le déploiement des applications sur le serveur.

Il est nécessaire de bien comprendre le fonctionnement de J2EE pour comprendre l'utilité de JBoss Seam.

2. JBoss Seam

Seam est **un framework – un espace de travail modulaire comprenant un ensemble de**

briques logicielles et de conventions – qui a pour objectif d'obtenir des applications pour Internet abouties mais aussi faciles à maintenir tout en utilisant une grande partie des composants de J2EE 5 (*EJB3*, *JSF*, *jBMP*, ...).

L'initiateur de ce projet est Gavin King. C'est l'un des principaux acteurs de la société *JBoss, Inc* rachetée par *Red Hat* qui a aussi réalisé *Hibernate* (framework de persistance pour Java) ou encore JBoss AS, le serveur Java que j'ai utiliser tout le long de mon stage. Gavin King a aussi participer à la mise en place de la technologie *EJB 3.0*. On comprend alors création de JBoss Seam. L'équipe de Gavin King a voulu harmoniser les technologies de J2EE en créant un seul outil qui puisse unifier les technologies Java dont certaines que King connait bien.

Les atouts du développement J2EE par Seam sont nombreux. La listes est longue mais en voici quelques uns :

- Le développement basé sur **EJB** :
  EJB 3.0 a révolutionné les idées reçues sur les EJB, jusqu'alors considérés comme des objets lourds et grossiers. JBoss Seam est composé d'EJB uniquement et adopte le concept Web 2.0 selon lequel le Web est une plate-forme. Ainsi, JBoss Seam élimine la distinction qui existe entre les composants de couche de présentation et ceux de la logique commerciale.

- La couche d'appels distants reposant sur **AJAX** :
  JBoss Seam Remoting permet aux EJB 3.0 Session d'être appelés directement à partir du navigateur web client via Ajax. Pour les développeurs *JavaScript*, les beans* de session apparaissent comme étant de simples objets JavaScript, masquant ainsi la complexité de la sérialisation XML et de l'API *XMLHttpRequest*.

- L'utilisation de **JSF** :
  JSF est conçu de manière à ce qu'une architecture solide au niveau de la couche de présentation soit obtenue. Seam vise à rendre accessible cette technologie en intégrant bon nombre de fonctionnalités à JSF. Il est possible de lier le bouton d'envoi d'un formulaire à une opération d'un composant Seam et de lier la valeur d'un champ du formulaire à celle d'un attribut. Il est aussi possible d'utiliser les valeurs des composants de Seam afin de conditionner l'affichage des composants graphiques.

* Comprendre ici « EJB »

- Le **pageflow** :

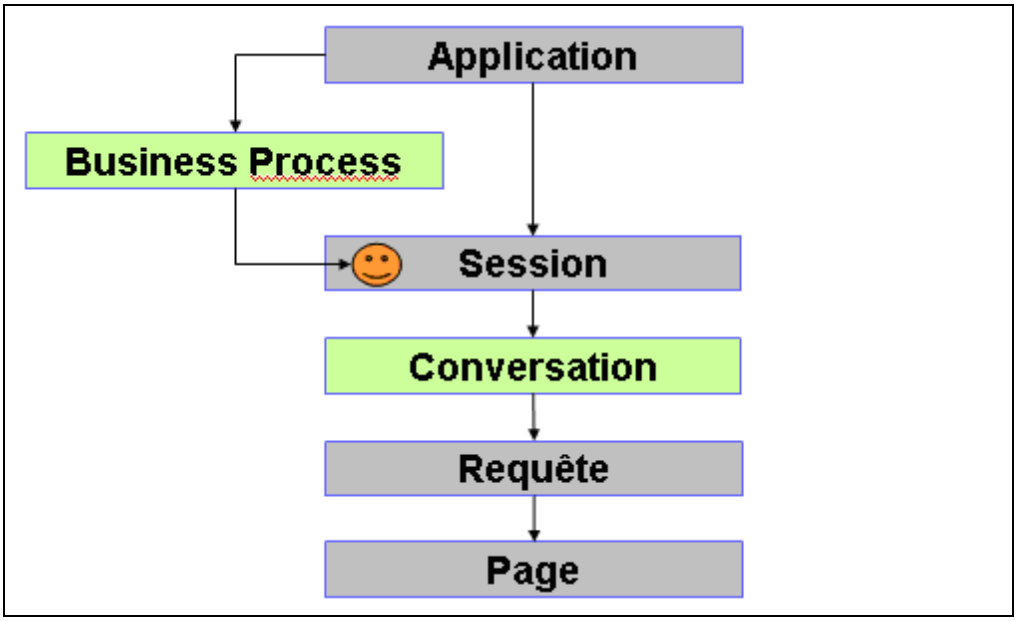
  Il représente une abstraction des interactions entre les pages web et les contrôles métier. La manipulation de ces composants de pageflow simplifie grandement le développement d'une application Web qui, pourtant, met bel en bien en œuvre un modèle MVC. Ces outils, qui permettent de s'abstraire des préoccupations purement technique – donc répétitives – ponctuent l'ensemble des développements.

- Les **annotations** :

  Ce sont des méta-données qui sont ajoutées au code source Java et qui permettent de préciser ou d'ajouter des comportements aux classes Java. Elles peuvent remplacer des fichiers XML afin de centraliser les informations relatives à un composant.

- Les **niveaux de contexte** :

  Les contextes sont des espaces logiques où évoluent les objets dans les applications web. Il en existe originellement quatres : l'application, la session, la requête, la page. Il y a plusieurs pages par requêtes, plusieurs requêtes par session et plusieurs session pour une application. Seam offre deux autres contextes : le *business process* et la *conversation* qui s'incluent dans le modèle de cette façon : l'application, le business process, la session, la conversation, la requête, la page. Le mécanisme de la conversation permet d'envisager la conception du point de vue des cas d'utilisation grâce à une abstraction de la conservation des données entre plusieurs pages.

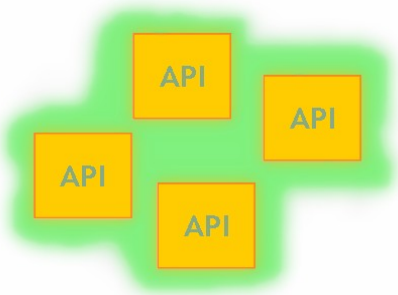


Niveaux de contextes sous JBoss Seam

- Le **générateur de code** :

  Seam intègre un outil de génération de code nommé *seam-gen*. Il permet par exemple de générer une application complète à partir d'une base de données existante. Je l'ai utilisé avec succès pour faire des applications peu fonctionnelles mais bien conceptualisées.

Seam s'occupe de d'unifier les technologie J2EE à disposition pour facilité le développement d'applications. « Seam » signifiant « couture » en anglais, le framework lie les API entre elles.



B. Les tests de montée en charge

1. Qu'est-ce qu'un test de montée en charge ?

Le test de montée en charge, aussi appelé *test de charge*, est un **type de test de performance**. Les tests de performance vont avoir pour objectif de mesurer les temps de réponse d'un système en fonction de sa sollicitation. Le TMC* est un test qui a pour but de **simuler une charge importante d'utilisateurs sur une durée relativement longue**, pour voir si le système testé est capable de supporter une activité intense sur une longue période sans dégradations des performances et des ressources applicatives du système.

Parmi les tests de performance, il y a aussi :

- Le test de capacité au cours duquel on va simuler **un nombre d'utilisateurs sans cesse croissant** (par paliers) de manière à déterminer quelle charge limite le système est capable de supporter.
- Le test en stress au cours duquel on va simuler **l'activité maximale attendue en heures de pointe de l'application**, pour voir comment le système réagit au maximum de l'activité des utilisateurs.
- Test aux limites au cours duquel on va simuler **une activité bien supérieure à l'activité normale**, pour voir comment le système réagit aux limites du modèle d'usage de l'application.

* Test de montée en charge

Le test en charge permet de s'assurer qu'une application fonctionnera correctement une fois mise en production. JBoss Seam étant assez récent, on ne connaît pas encore bien s'il peut être performant et permettre la mise en place d'applications web pouvant tenir des charges importantes d'utilisateurs. Seule la simulation de nombreux utilisateurs simultanés permet de détecter ces problèmes avant une mise en production.

2. Quel programme utiliser ?

Il existe plusieurs programme connus pour pouvoir effectuer des TMC sur des applications web, les plus connus étant *LoadRunner* (payant), *OpenSTA* (gratuit). Cependant ces deux logiciels ont chacun au moins un défaut compte tenu du stage que j'ai effectué.

LoadRunner est un excellent programme très réputé pour la qualité et la diversité des tests qu'il peut mettre en place. Étant un programme propriétaire, le principale désavantage de ce logiciel est le prix de la licence. OpenSTA pourrait alors être une alternative à LoadRunner. Malheureusement, ce programme n'est pas facile d'accès. Il requière beaucoup de pratique et est difficile à prendre en main pour un débutant comme dans un laps de temps aussi court qu'est celui du stage. Il se trouve aussi que ce programme est plutôt limité par rapport aux tests à effectuer.

Nicolas Chapurlat m'a alors demandé de me pencher sur un nouveau logiciel que Logica ne maîtrise pas encore beaucoup, même si certains architectes travaillent dessus en dehors de leur temps de travail. **NeoLoad** est un programme très récent conçu par la société française Neotys créée en 2005. Il offre beaucoup de statistiques qui sont très utiles pour les conclusions à faire après les tests et permet d'exporter des données sous un format simple d'un aspect très professionnel.

3. Mes premiers essais

Au moyen d'un simple navigateur internet, NeoLoad permet d'enregistrer des actions métiers sur un serveur web comme la soumission d'un formulaire ou l'exécution d'une recherche. Ces actions sont rejouées par autant d'utilisateurs virtuels que nécessaire afin de simuler la charge à laquelle sera soumis l'application et le serveur. L'agent du programme s'occupe de récolter les informations que le serveur renvoie (erreurs, débit, utilisation de la mémoire).

La prise en main de NeoLoad fut assez intuitive. En effet, en plus d'être disponible en

français, il est assez facile d'accès. On peut très vite comprendre comment enregistrer un scénario. Le scénario est, comme son nom l'indique, le scénario de pages que l'utilisateur va parcourir sur l'application. Après avoir saisie la licence provisoire, il ne m'a pas fallut longtemps avant d'enregistrer mon premier scénario. J'ai commencé graduellement pour pouvoir bien maîtriser cet outil.

Dans un premier j'ai enregistrer des scénarios sur une application extrêmement basique. Il s'agissait de trois pages HTML reliées entre elles par des simples liens hypertextes. Je me suis en suite attaqué à la montée en charge d'applications Seam sans grandes complexités.

L'application *seam-drools* est très basique. Il faut rentrer dans un formulaire un nombre entre 1 et 100 pour trouver celui choisit par l'application. On a 10 tentatives avant de perdre. L'application *seam-messages* liste les objets *Message*, permet de les lire, voir lesquels ont déjà été lu, les supprimer. L'application *seam-contactlist* est un peu plus évoluée que les précédentes. On peut ajouter à la base de données des contacts (nom, prénom, e-mail, …), effectuer des recherches, afficher/éditer/supprimer les fiches des contacts et y ajouter des commentaires. *(voir Annexes)*

4. Ma documentation

Lors de la prise en main de ce programme, NeoLoad, Nicolas Chapurlat m'a demandé de réaliser un document de quelques pages permettant à d'autres novices de ce programme de pouvoir facilement et rapidement débuter avec celui-ci. Le travail en question fut assez simple : présenter le programme et son installation ; expliquer comment enregistrer rapidement un scénario simple ; expliquer comment mettre en place et exécuter des tests ; expliquer comment utiliser certaines fonctionnalités utiles. *(voir Annexes)*

Cette tâche m'a permis de mieux saisir cette nécessité de capitalisation du travail. Étant un des premiers chez Logica à utiliser NeoLoad et donc à passer du temps dessus, il est plus logique de laisser une trace facile à comprendre pour que les prochains à s'en servir. Si je consacre du temps à comprendre les fonctionnalités et finalités de cet outil, j'acquière des connaissances que je peux transmettre à mon tour.

J'ai eu aussi rapidement l'occasion de montrer le fonctionnement du programme à un autre stagiaire de la cellule architecture, Régis Legrand, qui en aura besoin d'ici quelques semaines.

5. Le test final

Nous nous sommes mis d'accord qu'il serait difficile que je crée une application complète exploitant toutes les ficèles de JBoss Seam et que je fasse les tests de montée en charge pendant la durée du stage. Nous avons décidé de faire les tests sur une application exemple du framework, ***booking***, qui couvre la major partie des fonctionnalités de Seam. Je me suis donc penché sur cette application pour comprendre comment elle fonctionne, quels principes elle utilise et quels scénarios sont typiques de son utilisation.

Booking est un site qui sert à réserver des chambres dans des hôtels. Lorsqu'on va sur le site on doit s'authentifier à l'aide d'un login/mot de passe. On peut aussi s'enregistrer. Une fois authentifié on peut faire quelques actions telles que changer son mot de passe, effectuer une recherche d'hôtels, réserver une chambre dans les hôtels souhaités (en pré-payant), afficher la liste de ses réservations et en annuler. *(voir Annexes)*

Cette application est assez typique de JBoss Seam car elle couvre une bonne partie des ces spécificités : les conversations, les évènements, les EJB, JPA, le mapping O/R, JAAS, … On peut donc un terrain idéale pour tester Seam en quasi-totalité.

C. Mes conclusions

Malheureusement, à l'heure qu'il est je n'ai pas encore eu l'occasion de faire les tests finaux sur cette application, *booking*.

Pour finir cette étude il m'a fallut demander une licence pour le programme NeoLoad à Nicolas Chapurlat. La licence d'évaluation est bien pour prendre en mains le programme. Elle permet de faire des testes sans avoir plus de 10 utilisateurs virtuels en même temps et sans limite de temps, hormis les 30 jours d'utilisation du programme. J'ai d'ailleurs dû solliciter à la société Neotys, éditrice de NeoLoad, une extension de licence car la période initiale de 30 jours était dépassée.

Dans ces conditions, il n'est pas possible d'effectuer des TMC intéressants. Il faudrait au minimum simuler un nombre de 100 utilisateurs virtuels pour que cela puisse refléter une utilisation

vraisemblable de l'application et donc qu'il puisse être interprété.

Les scénarios étant près il ne reste plus qu'à utiliser la licence, installer un serveur JBoss AS, déployer les applications en mode *cluster* pour qu'en cas de crash d'une application le test puisse continuer.

Chapitre 3

Ma place dans l'entreprise

A. Difficultés rencontrées

Le sujet de mon stage portant sur **J2EE, il m'a été vraiment difficile les premiers jours de bien comprendre le fonctionnement de cette plate-forme Java**.

En effet, lors de ma formation à l'IUT j'ai eu l'occasion de programmer sur la plate-forme J2SE[1] pendant presque les 2 années du DUT. Nous avons aussi eu la chance d'avoir en seconde année un projet sur la plate-forme J2ME[2]. Malheureusement, même si nous sommes passés par une technologie J2EE (les servlets) en fin d'année pour produire un projet web, nous n'avons pas du tout pu découvrir cette autre plate-forme plus intéressante pour une entreprise comme Logica.

J2EE est typiquement destinée aux applications d'entreprise. Même si elle contient un ensemble d'extensions au framework standard de Java, il n'est pas facilement de débuter dessus sans certaines notions.

J'ai donc dû beaucoup prendre du temps pour me documenter sur ce premier point pour comprendre comment Seam fonctionnait.

B. Un apport mutuel

Lors de ce stage, la cellule architecture de Logica m'a confié une mission. Cela m'a ouvert une porte sur un nouveau domaine pour moi. C'est en cela qu'il y a eu un apport mutuel.

1. Une réponse à une problématique

J'ai tenté de **répondre à un problème** simple à comprendre : est-ce que les applications conçues avec JBoss Seam peuvent supporter une charge standard d'utilisateurs. Cette question renvoyant directement à une question bien plus globale : peut-on créer une offre autour de ce framework ?

Ainsi, même si mon travail ne se présente pas sous la forme d'un projet pour une entreprise

1 Standard Edition

2 Micro Edition

cliente chiffré en « jour par homme » et n'est donc pas rentable à cours terme, je suis heureux d'avoir pu offrir à Logica une réponse à leur question. Sans cette étude et la réponse qu'elle apportera, il pourrait être très dangereux de mettre en place des applications et de les commercialiser.

2. L'étude de Seam

Si j'ai donné à Logica une réponse technique débouchant sur une commercialisation ou non d'applications web développées avec Seam, l'entreprise m'a aussi apporté quelque chose.

Les difficultés que j'ai rencontré au début de ce stage ne furent pas veine. Si j'ai passé beaucoup de temps sur cette étude cela m'a offert des connaissances techniques sur la plate-forme J2EE et sur ce framework qu'est Seam. J'ai obtenu un nouveau savoir à travers ce travail et ce n'est pas négligeable à mon age d'**avoir de nouvelles compétences**, surtout sur J2EE qui est très utilisé en entreprise.

Aussi, un travail en autonomie m'a permis de **devenir plus responsable**. Sans avoir un planning strict à suivre, il a quand même fallut que je m'organise. La compréhension du fonctionnement de J2EE et de JBoss Seam était importante avant de passer à la prise en main de NeoLoad, pour en suite passer aux phases de tests. J'ai tenté d'être autonome sans devenir indépendant.

C. Ce que je retiens du stage

1. Mes premiers pas dans une SSII

N'ayant peu d'expérience dans le monde du travail, ce stage a mis en évidence mon souhait de vouloir travailler dans une SSII. **Le facteur *service*** est un facteur très motivant. Bien sûr je n'ai pas pu travailler en groupe sur un projet en rapport avec un client mais j'ai eu beaucoup d'échos là dessus : le plateau Legrand/Leroy Merlin où se trouve mon poste de travail officie souvent avec la clientèle ; les autres employés ou stagiaires ont aussi eu l'occasion de me parler de leurs clients.

L'aspect *société de service* donne un but à atteindre mais aussi il cadre en matière de temps

et de répartition des tâches ce qu'il faut faire et quand le faire. Cela demande une réflexion sur quoi mettre en œuvre pour satisfaire le client.

Aussi, travailler dans une SSII offre la possibilité d'utiliser ses capacités sur différents projets avec différents clients et non un seul et même. Cette diversité de clientèle, même si les secteurs auxquels elles sont associées sont souvent semblables, me paraît être une chose plus intéressante.

2. Le monde du travail

Mon entré dans le monde du travail m'a permis d'en apprendre plus sur le **fonctionnement hiérarchique et relationnel en entreprise**.

Les relations humaines en entreprises sont un paramètre important du résultat qu'on peut obtenir des équipes. J'ai pu me rendre compte, par exemple sur le plateau Legrand/Leroy Merlin, qu'il faut avoir de la verve lorsqu'on traite avec le client ou même avec les commerciaux qui s'occupe de notre projet. Il peut être fréquent que le client demande comment le projet avance alors que visuellement, on ne peut rien lui montrer de différent par rapport à la dernière fois. Dans ces cas là il faut savoir expliquer et faire des efforts dans son sens. A contrario, on doit aussi savoir expliquer au client qu'il faut laisser le temps à l'équipe pour faire les choses dans l'ordre pour que le résultat puisse être cohérent.

C'est aussi à partir de là que j'ai appris d'autres choses que des compétences techniques sur J2EE ou sur les tests de montée en charge. Le monde de l'entreprise et d'une SSII ne peut se résumé à faire un programme à partir d'un cahier des charges définitif. On ne peut pas toujours partir d'un point A pour aller à un point B sans embuches. Il faut parfois remettre en cause le chemin pour passer en C, étape où l'on présente l'avancement du projet au client par exemple, plusieurs fois et revoir quelques détails du point B.

Il faut aussi savoir construire un **esprit d'entreprise**. Nous avons eu l'occasion, quelques semaines après le début de mon stage, de participer une soirée stagiaire qui réunissait des stagiaires, des tuteurs et des managers de Logica. Après une présentation des différentes filières qui se proposent aux stagiaires pour les années à venir dans l'entreprise, nous avons été mélangés et répartis dans des équipes différentes totalement au hasard pour participer à un jeu physique et

intellectuel, le *InQuest*.

Cette soirée fut l'occasion de découvrir d'autres projets sur lesquels la DO Nord est impliquée. C'est autour d'un cocktail dînatoire bien mérité que s'est clôturée la soirée. La bonne humeur au rendez-vous, c'est pendant plusieurs jours que les couloirs de la DO ont entendu parler de cet événement. Durant cette rencontre ludique, nous avons pu développer notre relationnel professionnel et personnel tout en nous amusant !

Je suis heureux d'avoir pu faire un stage pour compléter ma formation à l'IUT. Il s'agit là d'un passage plus que nécessaire pour terminer une instruction professionnalisante.

Conclusion

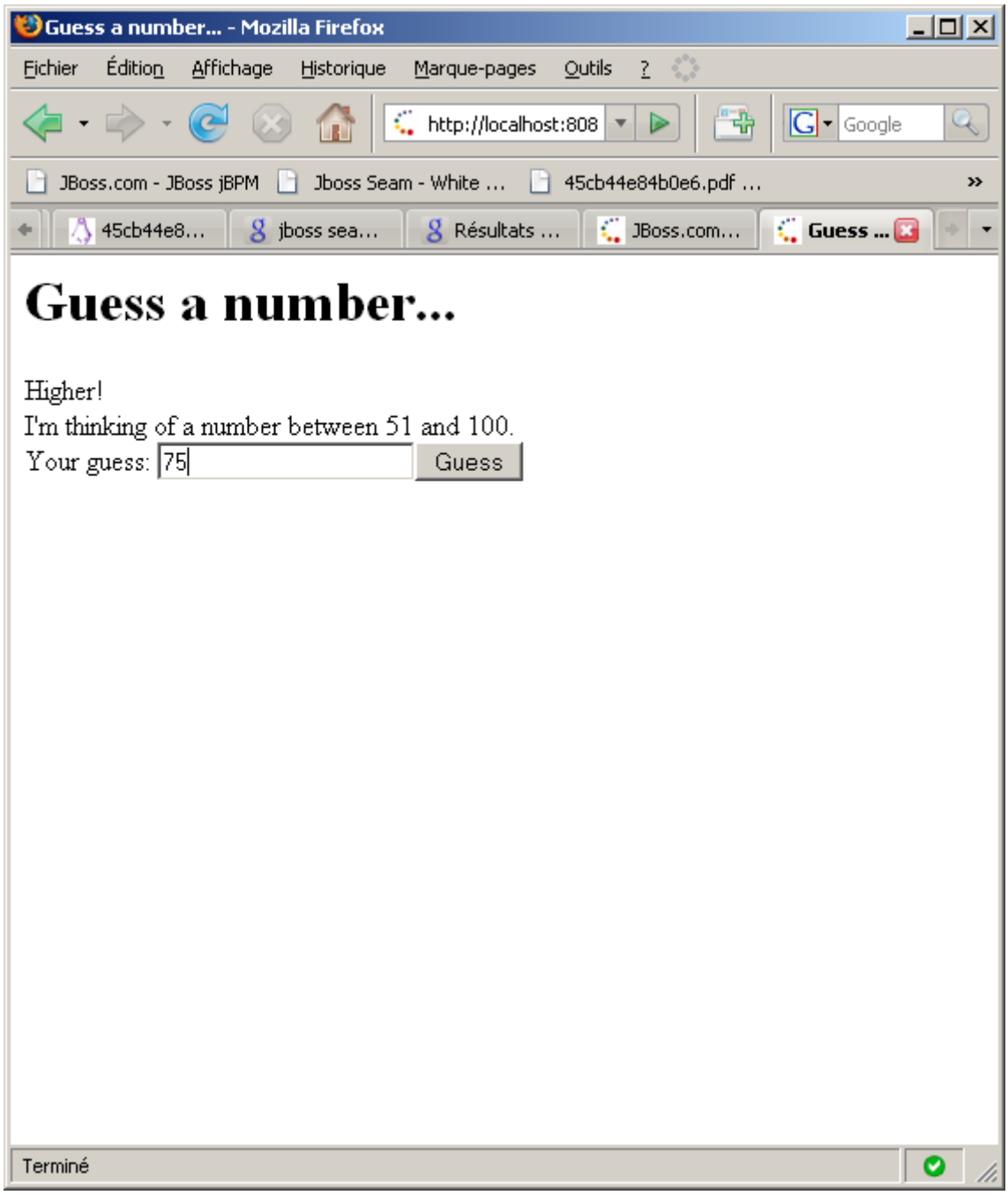
Ce stage m'a beaucoup apporté et m'a fait comprendre le rôle des informaticiens dans une société de service. J'ai apprécié travailler sur cette étude pour la cellule architecture et pouvoir apporter une solution qui sera capitaliser par Logica.

Il est cependant très dommage de n'avoir pas pu, à ce jour, effectuer des tests finaux avec la licence payante. Mais je reste à la disposition de Logica pour les effectuer avant la fin de mon stage où même après. S'approprier des connaissances dans de nouveaux domaines comme la programmation J2EE et les mécanismes et les outils des tests de montée en charge sera certainement utile pour mes prochaines années d'études où lors de ma vie professionnelle.

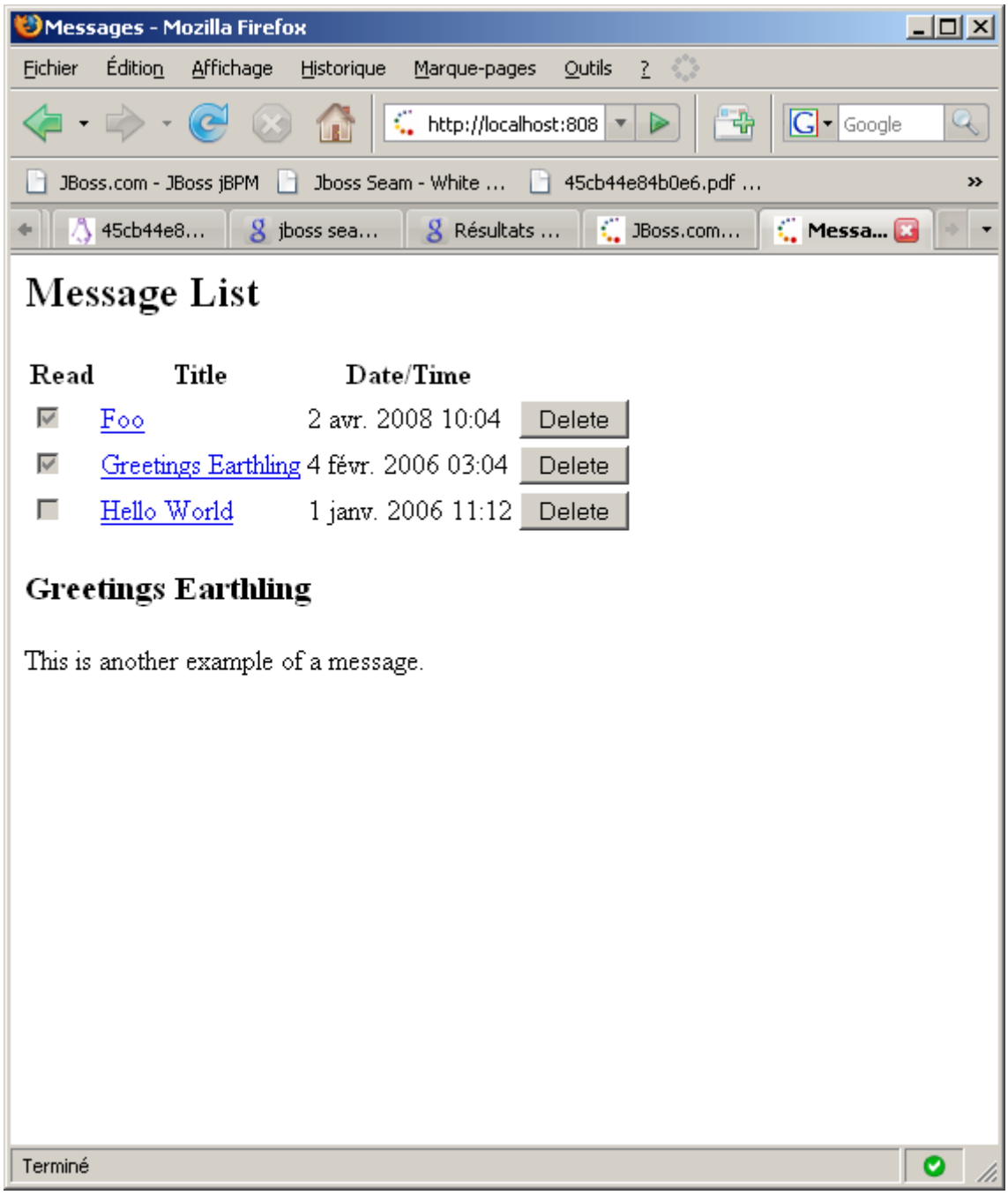
J'ai vécu ce stage comme un employé faisant son travail de tous les jours en autonomie plutôt qu'une personne de passage. Cette expérience a été très enrichissante d'un point de vue humain, technique et fonctionnel, notamment car il s'agit là de ma première expérience en entreprise. La DO Nord de Logica est très accueillante et est accompagnée d'une ambiance très agréable, ce sont là des facteurs motivant.

Annexes

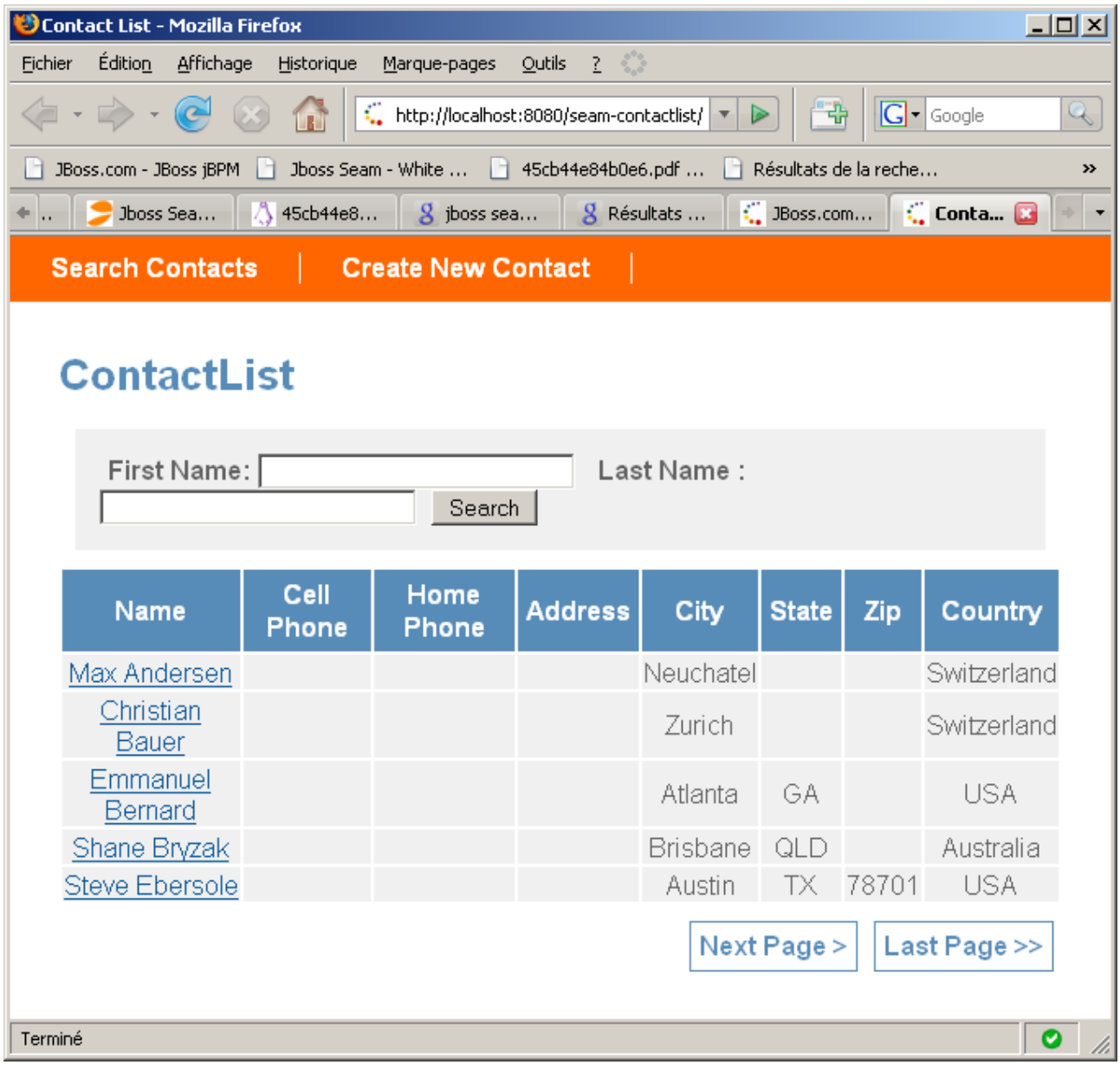
Capture d'écran : application « seam-drools »



Capture d'écran : application « seam-messages »



Capture d'écran : application « seam-contactlist »



NeoLoad

Sommaire

Présentation de programme 1
Mise en place 1
- Téléchargement 1
- Installation 1
Conception d'un enregistrement 2
- Préliminaire 2
- Enregistrer 2
- Assistant de fin d'enregistrement 2
Test de montée 3
- Valider un utilisateur virtuel 3
- Teste 3
Personnaliser son scénario 4
- Les variables 4
- Multi-utilisateur 6
A retenir 6

Présentation de programme

NeoLoad est un logiciel de test de montée en charge des applications web. Le test en charge vous permet de fiabiliser des applications web lors de la phase de développement, avant leur passage en production.

NeoLoad est un outil capable de générer une charge importante et réaliste sur une application web. Il permet d'étudier les statistiques sur le comportement du serveur et de montrer les erreurs et les problèmes de performance d'infrastructures web comme J2EE, .NET, PHP, ASP, CGI, AJAX, SOAP... en enregistrant et rejouant les requêtes HTTP et HTTPS envoyées au serveur.

A savoir : ce document est viable pour la **version d'évaluation 2.4.1** de NeoLoad.

Mise en place

➢ Téléchargement

Il faut dans un premier temps télécharger le programme sur le site de Neotys, l'éditeur du programme : http://www.neotys.fr/test-en-charge/download.html. Il va falloir remplir un formulaire pour pouvoir, par e-mail, obtenir la licence d'évaluation. Quelques secondes plus tard, vous recevrez un fichier *licence.lic*.

➢ Installation

Lors du premier lancement du programme, il faut activer la licence. Pour cela, le menu des paramètres généraux apparaît (sinon l'ouvrir : *Aide* > *Gestion de la licence...*) :

Sélectionner *Utiliser un fichier licence*, diquer sur *Entrer une clé de licence…* et *Appliquer*.

Conception d'un enregistrement

➢ Préliminaire

Pour paramétrer le programme il suffit de faire *Edition* > *Préférences*. Le port d'enregistrement par défaut est 8090.

Il est possible de configurer une navigation utilisant un proxy, d'ajouter/sélectionner par défaut un navigateur, et d'autre paramètres intéressant comme la gestion des paramètres dynamiques ou des options pour l'affichage des graphiques.

➢ Enregistrer

Commencer l'enregistrement en cliquant sur le bouton rouge qui se trouve sur le menu ou sur bouton *Conception* (cf. *Image 3*) > onglet *Bibliothèque* > *Bibliothèque* ou encore par *Enregistrer* > *Démarrer l'enregistrement*. (cf. *Image 1*)

Image 1

Un encart l'ouvre demandant le choix du navigateur et la suppression et le vidage ou non des cookies et du cache.

Le navigateur s'ouvre sur la page d'enregistrement de NeoLoad (http://neoload/), il faut alors saisir l'url de l'application web à tester. Terminer l'enregistrement en fermant le navigateur.

➢ Assistant de fin d'enregistrement

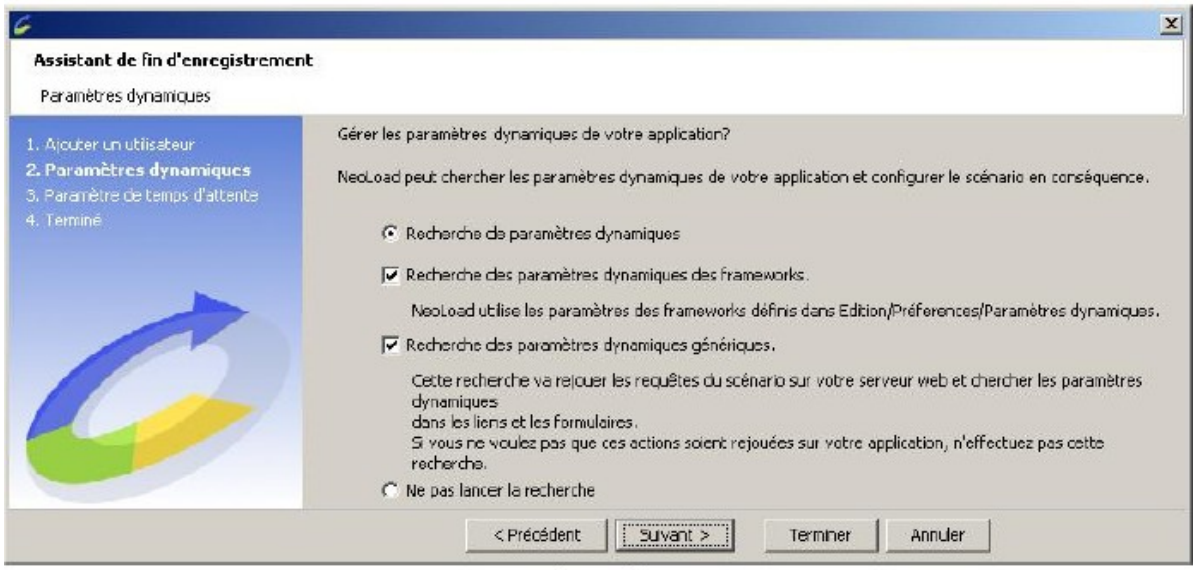


Image 2

Une fois l'enregistrement terminé, si tout s'est déroulé comme prévu, l'*Assistant de fin d'enregistrement* apparaîtra automatiquement. Il faut :
1 – Sélectionner l'utilisateur à qui l'enregistrement sera attribué (le créer au cas échéant)
2 – Rechercher les paramètres dynamiques si nécessaire (cf. *Image 2*)
3 – Inscrire un temps d'attente entre deux pages (simulation du temps de réflexion de l'utilisateur sur la page
4 – Terminer l'utilisation de l'Assistant

Test de montée

➢ Valider un utilisateur virtuel

Une fois l'enregistrement terminé, on peut re-paramétrer la les pages parcourus et les requêtes envoyées dans la bibliothèque si c'est nécessaire : bouton *Conception* > onglet *Bibliothèque* > *Bibliothèque*.

Image 3

Il est possible de modifier le scénario attribué à l'utilisateur : bouton *Conception* > onglet *Utilisateurs Virtuels*. Il faut aussi créer une population — onglet *Populations* — et attribuer l'utilisateur de la phase précédente à la population crée.

Ensuite, la validation de l'utilisateur virtuel peut se faire avant le teste pour vérifier le scénario. *Lancer* > *Démarrer la validation*. Dans la nouvelle fenêtre, sélectionner l'utilisateur adéquat, puis *Lancer la validation*. Si des erreurs sont retournées, cela ne sert à rien de lancer le teste : observer les onglets Résumé et Détails pour corriger les nœuds de pages et les requêtes.

➢ Teste

Lors du teste (*Lancer* > *Démarrer le teste*), NeoLoad fournis une pléiade d'information en temps réel. On dispose d'un graphique principal (cf. *Image 4*) et d'informations là encore en temps réel sur l'exécution (nombre de hits, nombre d'erreurs, nombre d'utilisateurs virtuels, temps de réponse, …).

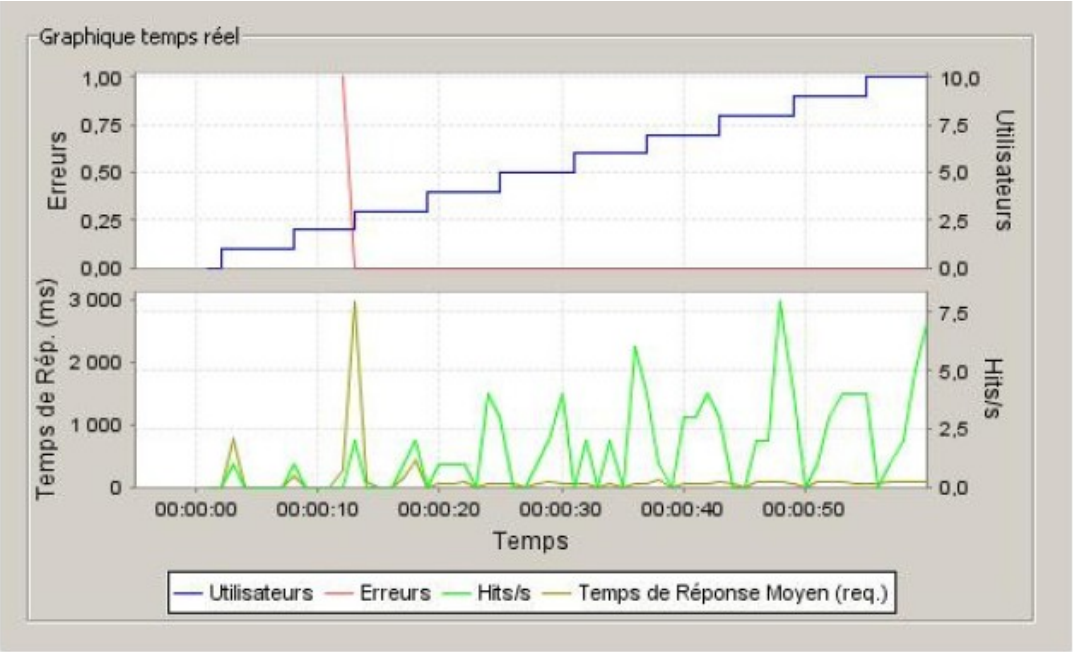


Image 4

NeoLoad fournira une batterie de résultat : une synthèse du teste, des valeurs, des graphiques, une liste d'erreurs. La synthèse apporte statistiques et graphiques. On peut aussi choisir de comparer son teste avec un ancien et en obtenir une nouvelle synthèse. Elle peut être exportée sous différent format RTF, PDF, HTML, XML pour obtenir des documents d'une qualité professionnelle.

Personnaliser son scénario

Pour personnaliser son scénario, on a la possibilité d'utiliser des variables. Actuellement, neuf types de variables sont disponibles : *compteur*, *liste*, *fichier*, *SQL*, *entier aléatoire*, *chaîne aléatoire*, *chaîne constante*, *date* et *date courante*.
Pour éditer des variables : cliquer sur l'icône adéquat (cf. *Image 5*) ou faire *Edition* > *Variables* ; une nouvelle fenêtre s'ouvrira alors. Cliquer sur le bouton *Image 6* (en bas à gauche) pour choisir le type de variable. Une fenêtre supplémentaire intitulée « Variables *Création de variables* » s'ouvrira.

Image 5

Image 6

➢ Les variables

Compteur

Utile pour changer un paramètre dans les requêtes envoyées, ce type de variable est simple à utiliser. Une fois la seconde fenêtre ouverte, sélectionner le type Compteur. Choisissez lui un nom ; des valeurs d'initialisation, d'incrémentation et de plafond ; sa politique de changement de valeur ; sa portée. Il faut paramétrer en fonction de votre scénario. Valider deux fois (*OK*, *OK*) pour fermer les fenêtres et votre variable sera prête pour être utilisée dans votre scénario.

Pour son utilisation, il faut re-paramétrer la ou les pages (souvenez vous : bouton *Conception* > onglet *Bibliothèque* > *Bibliothèque* > **Votre scénario**). Dans la hiérarchie

des pages et requêtes, choisissez celle à éditer. Si votre variable se nomme *aaa*, sa valeur sera *${aaa}*.

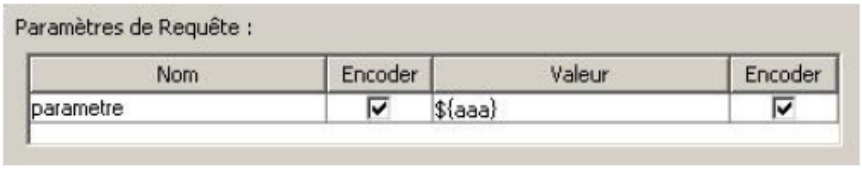


Image 7

En bas de la page, les boutons « + » et « - » servent respectivement à ajouter un paramètre à la requête et en supprimer un.

Liste

Une liste est aussi simple à éditer qu'un compteur. Il faut bien choisir le type Liste lorsque la seconde fenêtre apparaît. Pour paramétrer sa liste on peut ajouter des lignes et des colonnes ou d'en enlever. Pour information, le nombre de lignes et de colonnes peut dépasser 100 unités chacune. On peut obtenir par exemple un tableau d'utilisateurs (cf. *Image 8*).

Là aussi il faut choisir un nom ; l'intitulé des colonnes et les valeurs des cases ; la politique de changement de valeur ; la portée de la variable.

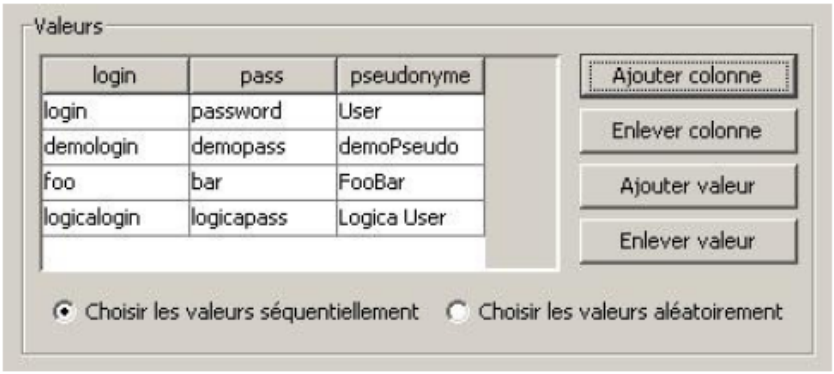


Image 8

On peut utilisé notre variable dans les requêtes envoyées par le programme comme précédemment avec la variable compteur. (Bouton *Conception* > onglet *Bibliothèque* > *Bibliothèque* > **Votre scénario**).
Si la variable compteur *aaa* s'appelle par l'expression *${aaa}*, la variable liste (de l'*Image 8* par exemple) qu'on nommera *bbb* s'appellera par l'expression *${bbb.colonne}*, colonne devant etre remplacé par l'intitulé de la colonne que l'on souhaite lire/écrire. Exemple : *${bbb.login}*, *${bbb.pass}* et *${bbb.pseudonyme}* sont disponibles pour cet exemple.

Important : les variables sont utilisables (lecture ou écriture) lorsqu'un sélecteur de ce type (à gauche) est disponible dans un champs :

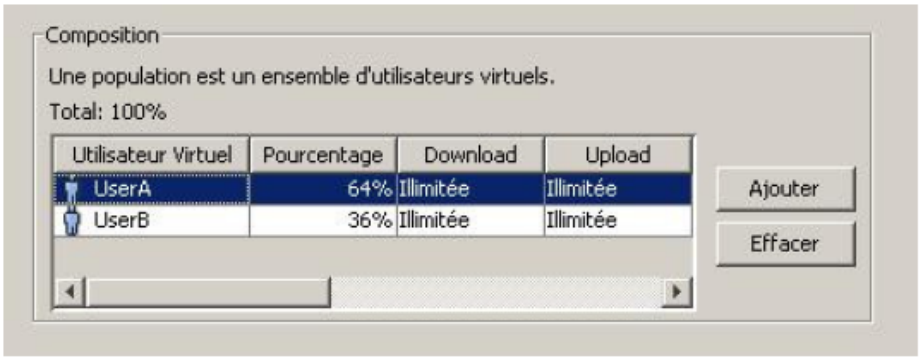
Image 9

En cliquant dessus, une petite fenêtre avec la liste des variables s'ouvre et il est facile de choisir quelle variable utiliser ou d'en créer d'autres via le *Gestionnaire des variables*).

➢ Multi-utilisateur

La navigation sur un site n'est pas linéaire. On peut donc créer plusieurs enregistrements pour qu'ils puissent modéliser plusieurs navigations (correspondant à plusieurs types d'utilisateurs). Une fois les différentes enregistrements créer il faut aller chercher notre population pour gérer ces navigations : bouton *Conception* > onglet *Populations* > **Votre scénario**.

Les boutons *Ajouter* et *Effacer* nous permettent de gérer les utilisateurs. Il faut aussi gérer le pourcentage de participation des utilisateurs sur le test de montée en charge.



On pourrait expliquer une navigation par un exemple comme celui-ci :

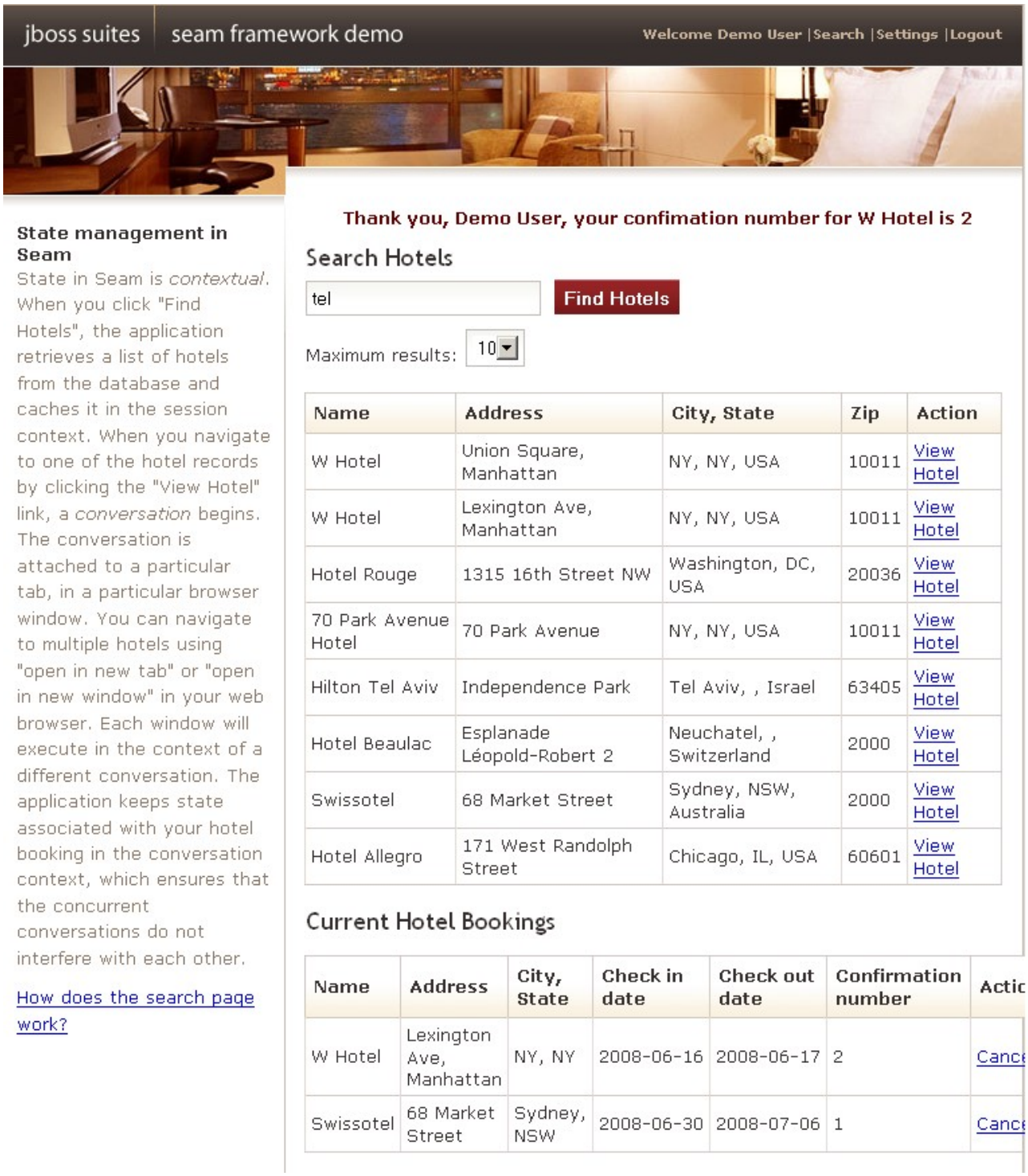
Scénario	**Utilisateur**	**Bibliothèque**
	Client (70%)	Achat
Boutique	Vendeur (20%)	Consultation
	Créateur (10%)	Ajout d'article

A retenir

NeoLoad s'avère être un programme très performant et facile à prendre en main. Les testes peuvent être effectués rapidement et comparés entre eux. La réactivité du support technique est aussi un atout.

Malheureusement, la version d'évaluation est un peu contraignante pour faire des testes de montée de qualités : 30 jours disponibles, nombre maximum d'utilisateurs virtuels simulé de 10, temps d'attente entre deux pages de 3 secondes.

Capture d'écran : application « seam-booking »



Glossaire

AJAX :	*Asynchromous Javascript And XML*, solution informatique libre utilisant conjointement HTML/CSS/DOM/Javascript/XML.
Contexte :	Espaces logiques où évoluent les objets dans les applications web.
Conversation :	Nouveau contexte apporté par Seam.
EJB :	*Entreprise Java Bean*, architecture de composants logiciels côté serveur.
Framework :	Espace de travail modulaire comprenant un ensemble de briques logicielles et de conventions.
Hibernate :	Framework open source gérant la persistance des objets en base de données relationnelle.
HTML :	*HyperText Makeup Language*, langage de balisage pour pages web.
J2EE :	Plate-forme de développement.
JavaScript :	Langage de programmation de script principalement utilisé pour les pages web.
JBoss AS :	*JBoss Application Serveur*, serveur d'applications J2EE.
JBoss Seam :	Framework applicatif développé par la société *JBoss, Inc*.
JPA :	*Java Persistance API*.
JSF :	*JavaServer Faces*, framework de programmation de Java.
JSP :	*JavaServer Pages*, technologie Java permettant de générer dynamiquement du HTML, XML et autre.
NeoLoad :	Programme de test de montée en charge.
Neotys :	Société éditrice de NeoLoad.
Seam :	cf. JBoss Seam.
XML :	*eXtensible Makeup Language*, langage de balisage.

Bibliographie

Informations sur Logica

- *http://buzz.fr.logica.com/* ainsi que beaucoup de documents Intranet
- *http://www.logica.fr*

J2EE, JBoss, JBoss Seam

- *Documentation interne à Logica*
- *http://docs.jboss.com/seam/2.0.0.GA/reference/en/pdf/seam_reference.pdf*
- *http://wiki.objetdirect.com/*
- *http://www.tutorialized.com/view/tutorial/EJB-3-tutorial-using-Eclipse-Ant-and-JBoss/11892*
- *ftp://ftp-developpez.com/valtech/articles/java/javaee/jboss/seam/Jboss-Seam.pdf*
- *https://unilog-jbs.pbwiki.com/*
- *http://www.dotnetguru.org/articles/Persistance/livreblanc/ormapping.htm*
- *http://www.valtech.fr/etc/medialib/pdf/itc/fr/valtech_days_2007.Par.94294.File.dat/Jboss%20Seam%20-%20White%20Paper%20-%20Valtech%20Days%202007.pdf*
- *http://www.solutionslinux.fr/document_conferencier/45cb44e84b0e6.pdf*
- *http://www.horstmann.com/corejsf/jsf-tags.html*

Tests de montée en charge, NeoLoad

- *http://www.neotys.fr/*
- *http://www.netqost.net/*

Divers

- *http://fr.wikipedia.org/*
- *http://en.wikipedia.org/*
- *http://www.commentacamarche.com/*
- *http://www.developpez.com/*