

# Blog Xebia France

J2EE, Agilité et SOA

<http://blog.xebia.fr>

---

**Revue de Presse Xebia**

La revue de presse de l'actualité Java/J2EE hebdomadaire proposée par Xebia.

## Actualité éditeurs / SSII

[Kenai, fermera, fermera pas ?](#)

### Le coin de la technique

[Groovy++: bientôt dans votre JVM ?](#) [InfoQ sort un ebook gratuit sur GrailsMaven vers une injection de dépendances via Guice](#) Actualité éditeurs / SSII [Kenai, fermera, fermera pas ?](#)

Comme il fallait le prévoir au vu de l'engouement d'Oracle pour les projets communautaires, la ferme de projets Kenai vit ses derniers jours en tant que service public. La plateforme lancée en 2008 et construite autour de JRuby, Glassfish et MySQL fournissait un environnement de suivi de projet complet avec la gestion de version, des wikis, des mailing lists et que sais-je encore. Avec NetBeans 6.7, il est même possible de maintenir entièrement son projet depuis son IDE. Le site héberge encore aujourd'hui quelques 40 000 projets communautaires parmi lesquels on retiendra JRuby et Hudson. Oracle a d'abord annoncé la fermeture du site sous 60 jours afin de permettre aux projets de migrer vers d'autres solutions. L'idée de départ devait être de ré-utiliser la plateforme en interne mais plus du tout ouverte au public.

Faut il y voir les effets de la réaction d'une communauté importante ou un gros problème de communication : Oracle change son fusil d'épaule à travers la voix de Ted Farrell qui a clarifié l'avenir de la plateforme en annonçant que ce serait finalement une migration vers java.net. Dans son annonce, Farrell laisse entendre que tous les efforts seront poussés sur java.net et que la communauté Kenai pourrait continuer à travailler sans perte sur la nouvelle version de cette ancienne plateforme de développement hosté.

Comme le dit la chanson, trois pas en avant, trois pas en arrière, ...

[Kenai to close sur InfoQ](#) [Le post de Ted Farrell](#) Le coin de la technique [Groovy++: bientôt dans votre JVM ?](#)

Nous avons éhontément passé sous silence nombre de news du monde Groovy récemment:

[Groovy 1.7](#) [Grails 1.2](#) La sortie du plugin Groovy Eclipse [en version 2.0](#) La sortie de [nouvelles versions de Griffon](#), le framework pour clients lourds qui reprend les idées de Grails

Alors pour nous rattraper, nous ne passerons pas à coté de la nouvelle suivante : le développement de Groovy++ ! Alex Tkachman, l'initiateur du projet nous explique [dans une interview](#) de quoi il retourne et Roshan Dawrani [nous explique ce qu'il retient du projet](#). En fait, malgré tous ses avantages, Groovy est encore assez lent. En tout cas beaucoup plus que Java, à cause de son coté dynamique. L'idée de Groovy++ est de continuer à utiliser Groovy normalement, mais en pluggant un compilateur spécial pour certaines parties du code annotées avec @Typed. Le code ainsi annoté devra respecter de légères contraintes (contraintes par rapport au laxisme de Groovy, mais légères par rapport au typage fort de Java). Le compilateur pourra donc effectuer des optimisations, notamment grâce à [l'inférence de type](#), et nettement améliorer les performances. Les premiers résultats semblent encourageant en tout cas : 197 fois plus rapide d'après [ce micro-bench](#) et 33% plus rapide [d'après cet article lu sur le Touilleur Express](#).

Alors ce Groovy++, une révolution ? Difficile à dire ! En effet, d'un coté Groovy est de plus en plus utilisé (notamment à travers Grails) et un coup de boost ne peut pas lui faire de mal. Mais d'un autre coté, l'interaction de Groovy et Java étant très avancée, on peut se contenter d'écrire en Java les parties du programme nécessitant des performances optimales. On peut aussi voir en Groovy++ l'ajout d'une énième librairie dans le développement, avec ses contraintes, ses bugs... D'autant que, pour des problèmes de licence sur certaines parties de code, le compilateur n'est pas encore OpenSource (ce qui est prévu pour la suite). Donc, si vous sous sentez l'âme d'un aventurier ou que vous avez désespérément besoin de gains de performances tout en gardant la « Groovy attitude », n'hésitez pas à nous faire vos retours sur l'utilisation de Groovy++ !

InfoQ sort un ebook gratuit sur Grails

InfoQ vient de sortir, dans sa série « [Minibooks](#) » la seconde version de son « [Getting started with Grails](#) ».

La première version, datant d'il y a déjà 3 ans, était basée sur Grails 0.3.1 alors que celle-ci s'appuie sur le tout récent 1.2. Le principe reste identique : les auteurs s'appuient sur une application de gestion de course de chevaux pour nous faire découvrir au fur et à mesure nombre des possibilités offertes par le framework. A la vue du sommaire, largement remanié, on peut penser que les changements ont été profonds. Sans avoir

encore lu cette seconde édition, nous pouvons d'ores et déjà la recommander chaudement à toute personne qui s'intéresse à Grails et voudrait rapidement avoir un éventail clair des possibilités qu'il offre (si elle est dans la même veine que la précédente). D'autant qu'au format électronique, le livre est gratuit (juste besoin de s'inscrire à InfoQ)!

Maven vers une injection de dépendances via Guice

Sonatype ([via InfoQ](#)) a annoncé la migration progressive de Maven vers une injection de dépendances gérée par Google Guice. Ce qui signifie l'abandon à moyen terme de Plexus, le framework quelque peu obsolète (et abscons, faute de documentation) utilisé depuis les débuts du framework de build. Dans un premier temps, la migration sera gérée via un bridge créé sous Guice.

De plus, l'introduction de Guice devrait se faire via les annotations de la JSR 330, ce qui rendra Maven moins adhérent à Guice dans le cas d'une nouvelle migration vers un autre framework d'IoC.

Cette nouvelle devrait réjouir les développeurs de plugins, qui n'auront plus à maîtriser les arcanes de Plexus. Autre conséquence, plus directe, les développeurs Maven n'auront plus à jouer les comittees Plexus pour débbugger et ou faire avancer cet outil (qui devrait rapidement se diriger vers une fin de vie en l'absence de ce soutien de poids).

On peut noter que Jason van Zyl, le créateur de Maven, et Bob Lee, l'un des principaux artisans de Guice, avaient ?uvré ensemble à imposer la JSR 330 face à la JSR 299 ([la fameuse bataille @Inject contre WebBeans](#)).

<http://blog.xebia.fr/2010/02/08/revue-de-presse-xebia-146/>

---

## Just Deployt !

Le 1er février dernier, [XebiaLabs](#) a mis en ligne une [version dite « Personnelle » de « Deployt »](#), sa solution d'automatisation des déploiements J2EE.

Cette version est gratuite et possède toutes les fonctionnalités de la version Enterprise, exception faite des aspects sécurité. Elle inclut une licence permettant à un utilisateur unique et identifié d'utiliser l'outil, sans limitations. Deployt Personal Edition inclut en standard des plugins pour IBM WebSphere AS, Oracle WebLogic Server et JBoss AS. Cette version peut être téléchargée gratuitement avec sa documentation et des tutoriels permettant de comprendre son fonctionnement.

Par ailleurs, avec Deployt Personal Edition, vous avez tout à disposition pour développer vos propres plugins : la documentation de l'API de plugin, les tutoriels et le code source des plugins existants. Enfin, en plus du support technique fourni via le web, notre équipe support peut être contactée gratuitement pendant 90 jours !

Si [Deployt Personal Edition](#) permet de bénéficier des avantages de Deployt (y compris en Production), il conviendra d'opter pour la version Enterprise incluant ses fonctionnalités de sécurité, son support technique complet et sa licence multi-utilisateurs dans le cadre d'un environnement plus complexe d'Entreprise.

Deployt Personal Edition peut être obtenue en cliquant sur le lien suivant :

<http://www.xebialabs.com/deployit-personal-edition-request>.

<http://blog.xebia.fr/2010/02/04/just-deployit/>

---

## Tomcat load balancing ? mod\_proxy vs mod\_jk le match

Dans notre article sur l'[utilisation de HTTPS avec Tomcat en production](#), nous avons étudié les solutions reposant sur la mise en place d'un reverse proxy HTTP. Nous n'avons pas oublié pour autant le protocole AJP. Ce protocole est né pour faciliter et accélérer les communications entre un serveur web frontal et le serveur d'application JServ en back-end d'Apache. Avec le temps, Tomcat a remplacé Apache JServ mais AJP est resté. Jusqu'en 2003, AJP était la seule solution viable permettant de placer le serveur d'application derrière un serveur Apache. Avec la maturation de la fonctionnalité Proxy dans Apache est née la solution tout HTTP. Nous avons donc décidé d'organiser un match opposant la solution AJP à la solution HTTP.

Un peu d'histoire

Tout commence en 1997 avec la création d'Apache JServ. A l'époque, il s'agit d'un serveur de Servlet qui supporte uniquement le protocole AJP créé pour l'occasion. Dans l'architecture initiale, c'est Apache 1.1 qui fournit le serveur web et transfère les requêtes par socket au moteur de Servlet. C'est la naissance du protocole [AJP dans sa première version](#) implémentée par le mod\_jserv. L'Apache JServ Protocol fonctionne au départ comme un proxy qui redirige le flux vers JServ. Le protocole est en texte clair et utilise un caractère en début de ligne pour distinguer les différents éléments de la requête.

Rapidement, le protocole initial est considéré comme trop limité car il fonctionne uniquement en loopback et possède une authentification pauvre. En 1998, le protocole [AJP 1.1](#) permet à JServ de tourner sur une autre machine que le serveur web et d'assurer une authentification forte basée sur md5. Avec le développement de JServ apparaît un problème de performance important : le coût d'ouverture d'une socket et la vitesse des réseaux. Ils sont considérés à l'époque comme les principaux goulets d'étranglement. Pour résoudre ces problèmes, le protocole passe à la version 1.2 dont vous trouverez le draft initial [ici](#). AJP 1.2 est un protocole binaire orienté paquets qui permet de recycler la ou les sockets connectées à JServ. Le passage au binaire permet aussi d'améliorer les performances car il diminue la taille des données qui transitent et simplifie le traitement.

En 1999, Sun offre son implémentation de référence des Servlets à la fondation Apache. C'est le point de départ des projets Tomcat et Ant. Commence alors une période de transition qui finira par l'abandon d'Apache JServ. Pendant cette transition, AJP sera porté sur Tomcat qui bénéficiera d'emblée d'une facilité d'interconnexion avec Apache. Aux alentours de 2000, le mod\_jk est développé pour étendre AJP qui pourra supporter le transport des données SSL. C'est la version 1.3 que l'on retrouve aujourd'hui supportée par les dernières générations de Tomcat. Après toutes ces années de développement, le mod\_jk est maintenu par le projet [Tomcat Connectors](#) et n'a jamais été intégré aux projets Apache.

La création d'AJP résulte donc de la simplicité et de la rapidité de développement souhaitées par les auteurs. Il fallait aller vite, et implémenter un proxy pleinement compatible HTTP aurait été trop long. Le module mod\_proxy existe depuis 1996 dans Apache 1.1. Mais il s'agissait d'une fonctionnalité expérimentale qui n'offrait ni performance ni stabilité. A la sortie d'Apache 2, le proxy a même été dé-scopé car il ne fonctionnait plus du tout. Les développeurs vont pourtant rapidement le corriger et le réintégrer comme solution pour faire des reverse proxies. Pour la sortie d'Apache 2.2, le mod\_proxy est entièrement réécrit pour supporter le load-balancing. Il offre, pour la première fois dans Apache, une solution capable de concurrencer le mod\_jk en performance et en scalabilité. Cerise sur le gâteau, Apache a décidé de supporter nativement le protocole AJP en ajoutant un mod\_proxy\_ajp à la solution mod\_proxy.

Installation mod\_proxy

Le mod\_proxy fait partie de la distribution standard d'Apache HTTPD. Il est livré avec le mod\_proxy\_http, le mod\_proxy\_ajp et le mod\_proxy\_balancer. Il suffit donc de s'assurer que ces modules sont bien chargés au démarrage d'Apache.

mod\_jk

Si mod\_jk était autrefois très délicat à installer avec la compilation du code sur un serveur similaire au serveur cible, la situation s'est grandement simplifiée. Le projet [Apache Tomcat Connector](#) fournit désormais [les binaires](#) pour les principales plateformes (Linux, Windows, Free BSD, Mac, Solaris, AIX, etc). Il faut donc télécharger le binaire du module et le copier dans le répertoire contenant les modules Apache.

Configuration

Pour mieux comparer les deux solutions, nous avons choisi de prendre comme exemple l'utilisation d'Apache en front desservant deux Tomcat en load-balancing. Nous ne nous intéresserons ici qu'à la configuration d'Apache HTTPD. La configuration AJP de Tomcat étant déjà largement documentée sur le web et celle via HTTP dans nos articles précédents, nous n'aborderons pas ces problématiques.

`mod_proxy_http & mod_proxy_balancer`

La configuration du `mod_proxy` consiste d'abord à s'assurer que les modules soient bien chargés avec les directives `LoadModule`. Nous activons ensuite le `server-status` ainsi que le `balancer-manager` pour obtenir une interface de surveillance / administration du load-balancer. Enfin, nous avons configuré le Proxy balancer nommé `my-application-cluster` pour contenir nos 2 serveurs Tomcat. La dernière ligne de configuration active le reverse proxy pour que les requêtes sur `/my-application` soient redirigées vers le `/my-application` du load-balancer.

### **Configuration avec `mod_proxy_http` & `mod_proxy_balancer` - `httpd.conf`**

`# LOAD MODULES`

`LoadModule proxy_module libexec/apache2/mod_proxy.so`

`LoadModule proxy_http_module libexec/apache2/mod_proxy_http.so`

`LoadModule proxy_balancer_module libexec/apache2/mod_proxy_balancer.so`

`# STATUS AND MONITORING`

`# Display proxy balancer status in /server-status page`

`ProxyStatus On`

`SetHandler server-info`

`Order deny,allow`

`Deny from all`

`Allow from localhost`

`SetHandler balancer-manager`

`Order Deny,Allow`

`Deny from all`

`Allow from localhost`

`# APPLICATIONS CONFIGURATION`

`BalancerMember http://node-1:8080 route=node-1 disablereuse=On`

`# ...`

`BalancerMember http://node-n:8081 route=node-n disablereuse=On`

`ProxyPreserveHost On`

`ProxyPass /my-application balancer://my-application-cluster/my-application stickysession=JSESSIONID`

Vous pouvez le constater, la configuration est parfaitement intégrée à la configuration standard d'Apache HTTPD. Les équipes de production n'auront à priori pas de grandes difficultés à prendre en main ce type de configuration qui nécessite seulement de savoir parcourir la [documentation Apache](#) déjà bien connue des administrateurs.

`mod_jk`

La première étape consiste à configurer le serveur Apache pour qu'il utilise le `mod_jk`. Tout commence par le

chargement du module avec la directive LoadModule. Ensuite nous fournissons le chemin du deuxième fichier de configuration définissant les workers. La directive JkMount permet ensuite d'associer un worker du mod\_jk à un pattern d'url du serveur. Pour chaque requête dont l'URL correspond au pattern, Apache va déléguer le traitement au mod\_jk. Nous montons d'abord le worker jkstatus sur /jkmanager en autorisant l'accès uniquement depuis le système local. C'est ensuite au tour du loadbalancer qui servira notre application.

### **Configuration avec mod\_jk - httpd.conf**

```
# LOAD MODULES

LoadModule jk_module libexec/apache2/mod_jk.so

# MOD_JK CONFIGURATION FILE
JkWorkersFile /etc/apache2/other/workers.properties

# MOD_JK PROPRIETARY LOG FILE
JkLogFile /var/log/apache2/mod_jk.log

# NEEDED ON MAC SNOW LEOPARD
JkShmFile /var/log/apache2/

# STATUS AND MONITORING
JkMount /jkmanager/* jkstatus

    Order deny,allow
    Deny from all
    Allow from localhost

# APPLICATIONS CONFIGURATION
JkMount /my-application/* loadbalancer
```

Il faut maintenant configurer le mod\_jk proprement dit dans son fichier de configuration spécifique. Il s'agit d'une liste de propriétés commençant toujours par worker.. La propriété worker.list fournit les noms des workers actifs. Le nom est ensuite utilisé pour paramétrer le worker avec des clés de la forme worker.nomWorker.parametre. Le premier worker activé jkstatus utilise le type spécial status qui correspond à l'interface de suivi et de gestion du mod\_jk. Le deuxième worker loadbalancer est en fait chargé de répartir les sessions entre le worker1 et le worker2 via le type lb. Ce sont finalement les workers 1 à n qui assurent le transport des requêtes sur AJP1.3 vers le port 8009 d'un Tomcat distant.

### **Configuration avec mod\_jk - workers.properties**

```
# WORKERS AND PSEUDO WORKER
worker.list=jkstatus, loadbalancer

# STATUS AND MANAGEMENT PSEUDO WORKER
worker.jkstatus.type=status

# WORKER 1 TO N
worker.worker1.type=ajp13
worker.worker1.host=node-1
worker.worker1.port=8009

# ...
```

```
worker.worker2.port=8009
worker.worker2.host=node-n
worker.worker2.type=ajp13
```

```
# LOAD BALANCER PSEUDO WORKER
```

```
worker.loadbalancer.type=lb
worker.loadbalancer.balance_workers=worker1,worker2
```

Avec ses deux fichiers de configurations séparés et la syntaxe "rustique" du worker.properties, la configuration est définitivement le point faible du mod\_jk. L'un des seuls avantages réside dans le nombre impressionnant de paramètres supportés qui permet un paramétrage fin si le besoin s'en fait sentir. Si seulement nous n'étions pas forcés à tant de verbosité !

#### Interfaces graphiques

Les deux modules fournissent des interfaces web pour assurer la supervision du load-balancer. Cette fois, le mod\_proxy se contente de fournir des statistiques réduites dans une interface minimaliste. L'interface liste principalement le statut de chaque nœud du load-balancer ainsi que la quantité de données envoyée et reçue par nœud.

Outre le suivi des statistiques du load-balancer, le Balancer Manager permet aussi de faire des modifications à chaud, éditer les workers, les passer en offline, voire même changer la méthode de répartition utilisée.

De son côté, le mod\_jk prouve sa maturité avec son interface rustique elle aussi, mais plus voire trop complète. Elle est composée de plusieurs pages dont une page d'accueil similaire en tout point à l'interface du mod\_proxy. L'avantage réside dans la fourniture d'une page détaillant l'état complet pour chaque nœud.

Mais le mod\_jk ne se contente pas de cela : il fournit aussi une page permettant de modifier à chaud la configuration du load-balancer. Il est parfaitement envisageable pour le déploiement en production d'une nouvelle version de l'application de couper les nœuds un à un au moment de leur mise à jour puis de les réactiver sans interruption du service.

Attention toutefois, car les modifications faites par ce biais sont uniquement enregistrées en mémoire, la nouvelle configuration sera perdue au prochain redémarrage d'Apache.

La mince différence vient probablement de la jeunesse de la solution de load-balancing du mod\_proxy face à la longue expérience de production du mod\_jk. Mais elle ne saurait justifier une préférence pour l'un des deux modules, qui sur ce point sont très proches.

#### LoadBalancing et gestion d'erreur

Outre la configuration et l'interface graphique, les deux solutions disposent de quelques fonctions avancées notamment en ce qui concerne la gestion d'erreur et la gestion des sockets réseaux. Les deux modules utilisent des pools de connexion rangés par Thread du serveur Apache et par membres du cluster.

Avec le mod\_proxy, il est possible de choisir parmi trois algorithmes de répartition de charge :

Par requête : la charge est répartie pour chaque requête entrante en fonction de la session si elle existe. Par trafic réseau : les requêtes sont envoyées vers le membre du cluster ayant reçu le moins de trafic. Par taux d'occupation : la charge est répartie en fonction du nombre de requêtes en cours de traitement ou en attente.

En ce qui concerne la gestion d'erreur, il n'y a pas grand chose à dire car il n'y a pas grand chose de fait. Si un timeout claque, un certain nombre de tentatives de reconnexion seront réalisées jusqu'à ce que le noeud soit marqué en erreur et n'en décolle plus.

C'est sur la gestion d'erreur que le mod\_jk possède une plus grande sophistication que son récent concurrent.

Tout d'abord le protocole AJP fournit un mécanisme de health check (CPing/CPong) qui permet de tester l'état du lien entre le serveur frontal et un membre du cluster. Cependant, c'est beaucoup plus limité que les heart beat des load balancer hardware, il n'est pas possible de tester une url pour détecter des indisponibilités applicatives (échec de démarrage de la web app, web app KO à cause de l'indisponibilité d'un backend clef).

D'autre part, le module fait une distinction entre les erreurs locales temporaires qui sont sans impact particulier, un code d'erreur HTTP par exemple, et les erreurs globales qui indiquent que le serveur est clairement en erreur et ne recevra plus de nouvelles requêtes. Il existe un système d'escalade qui, en cas de répétition trop fréquente d'erreurs locales sur un noeud se charge de le passer en erreur globale. Le module se charge de réactiver le noeud en mode recovery après un délai paramétré.

Côté répartition de charge, mod\_jk apporte un dernier algorithme de répartition reposant sur le nombre de sessions HTTP en cours par serveur. Cette méthode est relativement récente puisqu'elle est apparue dans la version 1.2.20 du mod\_jk, actuellement en version 1.2.29. Elle est recommandée pour les applications chargeant fortement la session et de ce fait, supportant un nombre limité de sessions par serveur ; ce cas d'utilisation est assez marginal.

Dans les deux modules, l'algorithme de répartition est pondéré par un facteur nommé "lbfactor", qui permet d'appliquer des quotas de travail aux serveurs. Ce système s'avère utile si les serveurs sont de puissances différentes par exemple.

Le mod\_jk marque ici un petit point sur le mod\_proxy grâce à sa gestion d'erreur plus fine. Attention aux effets de bord, le retry sur timeout en a surpris plus d'un et l'éviction de serveurs tomcat pour cause de répétition d'erreur est un beau sujet de déni de service

Pour le reste, le mod\_proxy colle au fonctionnement du mod\_jk, ce qui ne manquera pas de faciliter son utilisation aux habitués du mod\_jk.

Exploitation et diagnostic des problèmes

mod\_proxy\_http présente sur mod\_jk le très grand avantage d'utiliser un protocole standard, HTTP, connu de tous les acteurs d'un système d'information alors que mod\_jk repose sur le protocole AJP que quasiment personne ne connaît. Les administrateurs systèmes et réseaux sont habitués à HTTP et notamment à ses connections persistantes (aka HTTP keepAlive) ; ils savent ajuster leurs algorithmes de load balancing, les timeouts des firewalls et le dimensionnement des piles tcp/ip des serveurs (ulimit, tcp\_keepalive\_intvl, tcp\_tw\_bucket, etc).

Un autre atout de mod\_proxy est la facilité de diagnostic. N'importe quel acteur du système d'information peut utiliser curl, wget, telnet, elinks voire wireshark pour troubleshoot un problème de communication HTTP; ce n'est pas le cas avec le protocole AJP qui n'est pas human readable et encore moins human writable. Choisir AJP mérite de former les administrateurs systèmes et réseaux et de prévoir des outils de tests permettant de requêter un connecteur AJP mais ce n'est hélas que très rarement fait.

En cas de problème réseau avec HTTP comme avec AJP, n'oubliez pas que désactiver les connections persistantes simplifie grandement les investigations et n'a rien de scandaleux en 2010 (cf [HAProxy](#)) ; c'est "disablereuse=On" pour mod\_proxy\_http et "JkOptions +DisableReuse" pour mod\_jk

Conclusion

Avec sa simplicité de configuration, sa répartition de charge calquée sur le mod\_jk, le mod\_proxy conviendra parfaitement dans la grande majorité des cas. Il s'accommode de clusters répartissant la charge sur plusieurs noeuds ou bien en tant que simple reverse proxy. La cerise sur le gateau étant l'utilisation du protocole HTTP

qui permet de garantir la portabilité des services et facilite grandement l'analyse du trafic.

**A nos yeux, avec l'intégration native de mod\_proxy\_http et mod\_proxy\_balancer à Apache, il n'y a plus de justification à ajouter le module additionnel mod\_jk ni d'introduire le protocole méconnu AJP. Les optimisations d'AJP ne sont plus de mise aujourd'hui et ne justifient donc pas l'utilisation d'un mod\_jk.**

Il reste, dans les deux cas, quelques efforts à faire pour permettre de plus facilement monter des serveurs à chaud dans un cluster. Pour la haute disponibilité sans perte, il faudra mettre en place un [cluster Tomcat](#) assurant la réplication des sessions utilisateur entre les serveurs. Encore faut-il en avoir vraiment besoin ...

Attention, la recommandation de Tomcat est encore le mod\_jk et, il est important de garder fonctionnel ce qui marche déjà. Donc, quoiqu'il arrive, si vous avez déjà une solution fonctionnelle avec le mod\_jk, inutile de migrer. Pour ce qui est de l'interface et de la gestion d'erreur, nous parions sur l'avenir du mod\_proxy qui est en développement intensif et bénéficie des corrections de bug de son aîné. Reste un nouveau venu dans le paysage développé par Jboss qui attire déjà notre attention c'est le [mod\\_cluster](#) actuellement, il n'est utilisable qu'avec Jboss AS. L'avantage de cette nouvelle solution en devenir est de suivre le cycle de vie des applications via un protocole spécifique MCMP reposant tout de même sur HTTP.

[http://blog.xebia.fr/2010/02/03/tomcat-load-balancing-mod\\_proxy-vs-mod\\_jk-le-match/](http://blog.xebia.fr/2010/02/03/tomcat-load-balancing-mod_proxy-vs-mod_jk-le-match/)

---

**Ma rencontre avec Ken Schwaber**

Le French Scrum User Group a eu le grand privilège de rencontrer Ken la semaine dernière grâce à notre sponsor Microsoft qui le recevait dans le cadre de formations organisées en France.

Je souhaite relater ici les propos qui se sont tenus lors de cet échange avec Ken.

Quant à mes impressions personnelles, je ne les partagerai pas avec vous car, en tant que Président du SUG, je souhaite rester neutre face aux différents courants de pensée, organismes certificateurs ou non, canal historique ou dissidents ?

Le premier contact avec Ken est surprenant. Le co-fondateur de Scrum avec [Jeff Sutherland](#), la figure légendaire de l'agilité est fatiguée, un peu diminuée physiquement également. Son accident de vélo a laissé des traces.

Ken m'a glissé à l'oreille : "c'est parce que je fais du vélo que j'ai failli mourir mais c'est également parce que je fais du vélo (et donc que je suis en forme physique) que j'ai pu m'en sortir."

Derrière son visage marqué par les décalages horaires permanents, se cache un personnage qui n'a rien perdu de sa verve, de son dynamisme, de son humour, de ses convictions.

Ken est un passionné de développement logiciel, cela se voit.

Nous lui avons demandé de nous présenter [Scrum.org](#) dont il est le fondateur.

Première définition : [Scrum.org](#) a pour objectif de "préserver l'intégrité et la consistance de Scrum" (sic !).

Allons plus loin.

[Scrum.org](#) propose des tests en ligne (assessments) et des cours destinés aux praticiens Scrum.

Parmi les tests en ligne, notons l'arrivée de :

Scrum Developer (SD), test destiné aux développeurs. Scrum Open, test en ligne destiné à évaluer le premier niveau de connaissances Scrum d'un utilisateur. Scrum Level II, le test pour les expérimentés. Attention !: Pour réviser et réussir ces tests, il faudra avoir lu le Scrum Guide dont l'auteur est ? Ken Schwaber.

A noter que Scrum n'est plus agnostique et que des cours Scrum Developer purement .net ou purement Java sont au programme.

La prochaine rencontre se fera avec Scott Ambler, le 15 mars grâce à IBM. Je donnerai des informations précises dans le cadre des activités du SUG.

<http://blog.xebia.fr/2010/02/02/ma-rencontre-avec-ken-schwaber/>

---

## Revue de Presse Xebia

La revue de presse de l'actualité Java/J2EE hebdomadaire proposée par Xebia.

## Actualité éditeurs / SSII

<http://sun.com> 301 moved permanently

## Le coin de la technique

[Première Beta pour Scala 2.8.0](#)[Apache JackRabbit implémente maintenant JCR 2.0](#)[Un ?il sur Lucene, Solr et HBase](#)

## Evènements de notre communauté en France et à l'étranger

[2ème anniversaire du Paris JUG](#) Actualité éditeurs / SSII<http://sun.com> 301 moved permanently

C'est officiel, Sun n'est plus. Outre les messages humoristiques (teintés d'amertume, [chez James Gosling](#) par exemple), le redirect de <http://sun.com> vers <http://oracle.com>, le rebranding de tous les logos (sur la page de [téléchargement du sdk](#) par exemple), la concrétisation du rachat de Sun par Oracle fait ressurgir les questions, orientées produit, que l'on se posait déjà lors de l'annonce de ce rachat. Dionysios G. Synodinos offre, via InfoQ, [une liste détaillée de celles ci](#) :

**Java, la JVM, le JCP** : Thomas Kurian a annoncé son intention de tirer le meilleur de chacune des deux JVM de la firme, Hotspot et JRockit, pour en faire le choix par défaut face à la JVM d'IBM. Les développements planifiés sont alléchants, avec entre autres un meilleur support du multi c?urs, suppression du permGen (pour permettre une meilleure intégration des langages type Groovy ou JRuby). Concernant le JCP, aucune annonce claire n'a été faite pour l'instant.**MySQL** : cette base de données est annoncée comme complémentaire de la base historique d'Oracle. Elle bénéficiera, selon Larry Ellison, de plus d'attentions que par le passé, et les préoccupations marketing seront dissociées de celles du grand moteur, avec une équipe commerciale dédiée.**JavaFx et RIA** : Oracle a de nouveau affiché sa volonté d'investir massivement dans ces technologies. Certains analystes avancent que cet investissement viendrait boucher le trou laissé par l'échec d'intégration de Flex avec la suite BEA.**Netbeans** : des produits cités, c'est le plus menacé. Oracle laisse planer le doute, mais serait plus enclin à pousser JDeveloper. Restent les rumeurs d'une survie en tant que second couteau, ou bien un hypothétique reversement du code source à la communauté ...

**Glassfish** : face au mastodonte Weblogic, le serveur de référence JEE 6 serait proposé comme serveur d'appoint (l'expression officielle est departmental solution à opposer à l'entreprise solution Weblogic). Contrairement à MySQL, Oracle n'annonce pas d'équipe marketing ou commerciale dédiée. Certains (comme [Alexis MP](#)) y voient une bonne nouvelle (Oracle présenterait une offre comparable à celle d'IBM - WAS Community Edition / WAS). D'autres au contraire, y voient le premier clou dans le cercueil de ce serveur (avec là encore une analogie à IBM et l'anémique Geronimo).**Service Cloud** : là c'est officiel, les projets de Sun sont abandonnés.**Investissement dans l'Open Source**: Oracle n'a pas fait d'annonce officielle sur la stratégie à venir, mais la majorité des journalistes / bloggers de l'eco système JEE sont très pessimistes vis à vis de la capacité d'investissement d'Oracle dans le libre.**Effectifs** : sur un terrain plus terre à terre, Larry Ellison envisage le départ immédiat de 2000 personnes, et l'embauche de plus de 2000 nouveaux employés, principalement dans des postes d'ingénieurs, ou de commerciaux. Il ne s'est bien sur pas engagé sur un nouveau plan à plus long terme. Côté direction, le départ de Jonathan Schwartz semble entériné, et l'avenir de Scott McNealy est plus que flou. Le coin de la technique

Quelques nouvelles de [Scala](#) avec la sortie de la [première Beta de la version 2.8.0](#), information relayée par [JavaLobby](#) et [InfoQ](#).

[Nouvelles fonctionnalités](#) et [nombreuses corrections de bugs](#) sont au menu :

l'API Collection a ainsi été reconçue, l'API XML a aussi été améliorée, on peut utiliser des arguments nommés et/ou donner une valeur par défaut, les [Packages](#) peuvent désormais contenir des méthodes, des champs ou des types, il est possible d'utiliser des acteurs light, le support des annotations Java nested a été ajouté. De nombreux efforts ont aussi été apportés au niveau de la vitesse de compilation qui est maintenant plus rapide de 50%.

Côté outils, le plugin eclipse a été profondément remanié et de nombreuses opérations sont désormais réalisées par le compilateur Scala (et non plus l'IDE comme c'est le cas aujourd'hui). [Scaladoc 2](#) fait aussi son apparition avec un nouveau look-and-feel et plusieurs autres améliorations. Et pour le REPL, il supporte

la complétion sur les objets, méthodes, champs et bien d'autres.

En bref, une release pas si mineure que ça ! Le téléchargement se passe [ici](#).

A noter aussi la non compatibilité binaire entre la version 2.8 et la branche 2.7.

Apache JackRabbit implémente maintenant JCR 2.0

Sans grand bruit, l'équipe d'Apache JackRabbit a mis à disposition cette semaine la version 2.0 finale de leur projet. JackRabbit 2.0 est maintenant entièrement conforme à la [JSR-283 \(Java Content Repository 2.0\)](#) dont il est l'implémentation de référence.

Java Content Repository n'est pas une technologie très répandue, non pas parce qu'elle n'est pas satisfaisante, mais parce qu'elle adresse des besoins très spécifiques : JCR permet le stockage de documents sous forme arborescente tout en offrant des capacités transactionnelles, le versionning, ou encore une gestion de verrous. Les principaux cas d'utilisation sont les systèmes de gestion de documents, les CMS, ou des systèmes de stockage de fichiers organisés tels que le propose [Artifactory](#), un repository Maven fonctionnant avec JCR.

JCR 2.0 apporte, outre un rafraichissement de son API, quelques améliorations majeures :

**Query Object Model** : un nouveau modèle de requêtes objet à l'image de Criteria chez Hibernate ou de QueryBuilder dans JPA 2.0. Nous vous parlions déjà [il y quelques mois](#) des perspectives liées à cette fonctionnalité au travers l'API proposée par Magnolia. **API de retention et hold** : cette API vient compléter l'API de lock déjà présente afin de permettre des politiques de rétention de documents particulières. **API de contrôle d'accès** : les aspects de sécurité [étaient peu abordés](#) par la JSR-170 (JCR 1.0). Une API plus complète leur est désormais dédiée.

Au-delà de la conformance à JCR 2.0, JackRabbit 2.0 est maintenant basé sur Java 5, et offre la possibilité de recherche full-text sur les documents grâce à Apache Tika, un sous-projet de Lucene.

Ces changements ne suffiront probablement pas à eux seuls à élargir la communauté d'utilisateurs de JCR ; en revanche le récent engouement pour les technologies NoSQL peut contribuer à attirer vers JCR des équipes qui, jusqu'alors, auraient été plus frileuses à considérer un autre stockage de données qu'un SGBDR.

Un ?il sur Lucene, Solr et HBase

Malgré son jeune age, [HBase](#) suscite un intérêt suffisant pour pousser certaines entreprises [à l'utiliser en l'état](#) en production : les services apportés par cette base de données NoSQL suffisent à faire oublier l'expérience difficile que peut parfois réserver ce projet.

Dans un tel contexte toute source d'information est utile. Le [Jira](#) et la [mailing list](#) du projet deviennent alors des sources d'informations précieuses qu'il s'agit de suivre avec attention pour être au fait des dernières corrections et solutions aux problèmes couramment rencontrés.

Pour répondre à ce besoin, Sematext, une entreprise américaine spécialisée sur Lucene, Solr et HBase [diffuse depuis peu un digest](#) régulier des nouveautés importantes sur chacune de ces technologies.

On apprend ainsi qu'une abstraction est en cours de développement pour HBase. Nommée [HBql](#), elle vise à offrir une API proche de JDBC et un langage de requête SQL, ce qui constituerait un environnement de développement plus familier aux développeurs démarrant avec HBase (mais pas forcément plus efficace...). Du côté de la communauté Lucene / Solr, c'est la recherche géo-spaciale qui semble actuellement monopoliser l'intérêt. Cette possibilité est présente [depuis la version 2.9](#) dans Lucene et correspond à l'air du temps puisque [la géo-localisation devient courante](#).

Evènements de notre communauté en France et à l'étranger  
2ème anniversaire du Paris JUG  
Mardi 9 février 2010 aura lieu le deuxième anniversaire du [Paris Java User Group](#).

Pour l'occasion, l'équipe du Paris JUG nous a concocté une soirée spéciale sur le thème de l'Open Source :  
18h45 à 19h00 : Accueil  
19h00 à 19h10 : Le mot de l'équipe  
19h10 à 20h00 : Keynote de Sacha Labourey  
20h00 à 20h45 : L'open source en France  
Comment Obeo est devenu membre stratégique de la

fondation EclipseXWiki.org vs XWiki.com 20h45 à 21h30 : Buffet 21h30 à 23h00 : L'open source en France : Développons en Java jCaptchaPlay! framework une (r)évolution pour les applications web en Javajax-doclets Présentation d'eXo Platform 23h00 à ... : 3ème mi-temps des juggers Cette soirée aura lieu [dans un amphi de 500 places de la Sorbonne \(108 Boulevard Malesherbes dans le 17ème\)](#).

[Le programme détaillé est disponible sur le site du Paris JUG. Les inscriptions se font par ici. Pour participer à la troisième mi-temps exceptionnelle, c'est par ici.](#)

Xebia est fière d'être sponsor de l'événement.

En espérant vous voir nombreux ...

<http://blog.xebia.fr/2010/02/01/revue-de-presse-xebia-145/>

---

## En aparté avec Jeff Sutherland

Xebia vous invite le mardi 09 février 2010 à une soirée en aparté avec Jeff Sutherland.

Cette soirée se déroulera sur le modèle [Birds of Feather \(BoF\)](#) à partir de 19h00 dans nos locaux (156 Boulevard Haussmann à Paris). Elle sera l'occasion d'**échanger directement avec Jeff Sutherland** et de profiter de son expérience unique en tant que père de la méthode ayant plus de 10 ans d'expérience sur Scrum.

Le même soir que le [deuxième anniversaire du Paris JUG](#) ? Oui, Jeff Sutherland passe rarement à Paris, nous aurions aimé que les événements ne se chevauchent pas mais nous n'avons pas trouvé d'autre solution. Donc nous sponsorisons la soirée anniversaire du JUG et nous organisons cet aparté avec Jeff Sutherland .

Le nombre de place étant limitées, merci de vous inscrire via le formulaire ci-après (premiers arrivés, premiers servis) :

En aparté avec Jeff Sutherland Votre nom(obligatoire) Votre prénom(obligatoire) Votre adresse email(veuillez entrer une adresse email valide) Votre site web Informations complémentaires Captcha anti-spam

<http://blog.xebia.fr/2010/02/01/en-aparte-avec-jeff-sutherland/>

---

## 2ème anniversaire du Paris JUG

Mardi 9 février 2010 aura lieu le deuxième anniversaire du [Paris Java User Group](#).

Pour l'occasion, l'équipe du Paris JUG nous a concocté une soirée spéciale sur le thème de l'Open Source :

18h45 à 19h00 : Accueil  
19h00 à 19h10 : Le mot de l'équipe  
19h10 à 20h00 : Keynote de Sacha Labourey  
20h00 à 20h45 : L'open source en France Comment Obeo est devenu membre stratégique de la fondation Eclipse  
XWiki.org vs XWiki.com  
20h45 à 21h30 : Buffet  
21h30 à 23h00 : L'open source en France : Développons en Java jCaptchaPlay! framework une (r)évolution pour les applications web en Javajax-doclets  
Présentation d'eXo Platform  
23h00 à ... : 3ème mi-temps des juggers  
Cette soirée aura lieu [dans un amphi de 500 places de la Sorbonne \(108 Boulevard Malesherbes dans le 17ème\)](#).

[Le programme détaillé est disponible sur le site du Paris JUG. Les inscriptions se font par ici. Pour participer à la troisième mi-temps exceptionnelle, c'est par ici.](#)

Xebia est fière d'être sponsor de l'événement.

En espérant vous voir nombreux ...

<http://blog.xebia.fr/2010/02/01/2eme-anniversaire-du-paris-jug/>

---

**Performance, les Xebians jouent les démineurs**

Le premier [XKE dans nos nouveaux locaux](#) a donné lieu à de bien curieuses scènes : des bisounours ont hué des poubelles sous le regard moqueur de pokemons ! Et, non, les cartons de déménagement ne nous sont pas tombés sur la tête. Ce n'était là que quelques uns des noms choisis par des équipes de 3 à 4 consultants, qui se sont mesurés dans un concours de tuning de performance, sur une application Java EE standard, buggée (volontairement, pour une fois) par les maîtres de cérémonie, [Guillaume Bodet](#) et [Cyrille Le Clerc](#). Tous les participants se sont vus remettre une VM, contenant un Tomcat, une application (PetClinic de Spring, revue et "corrigée") et des scripts de performance JMeter. Le code source n'a, dans un premier temps, pas été fourni.

Pour tous, un seul but : faire diminuer les temps de réponses de l'application.

Les règles étaient les suivantes :

Un bug n'est considéré comme trouvé que lorsqu'il a été identifié, qu'un correctif a été proposé et que la preuve est faite que ce correctif permet d'améliorer significativement les temps de réponse. Il existe trois niveaux de difficulté, allant du bug évident à l'anomalie la plus fourbe. Le choix des outils est libre.

A vos marques... Prêts ? Débuggez !

Préambule

L'un des buts de ce Xke était d'être très didactique, dans les méthodes de recherche et dans l'utilisation des outils. Les bombes placées dans le code permettaient de réaliser une progression linéaire (dans la difficulté comme dans les gains attendus). Cette progression est retranscrite dans cet article et ceux qui vont suivre.

Cependant, pour différentes raisons que nous aurons l'occasion d'expliquer, l'ordre des analyses, et donc de découvertes des bugs, n'est pas celui qu'aurait choisi un champion de la performance, j'ai nommé Kirk Pepperdine. Nous lui avons soumis notre application, et nous vous exposerons sa méthode d'analyse et les raisons qui la motivent dans le dernier article de la série.

Premier contact avec l'application...

Ouverture du navigateur, entrée de l'url, et, pas de surprise, l'application PetClinic s'ouvre.

Premier lancement de JMeter, et première constatation : avec un seul utilisateur, l'application met en moyenne plus de 2 secondes à répondre. Inacceptable pour la plupart des sites web (surtout avec un seul utilisateur actif).

... et premiers réglages

À première vue, la machine ne semble pas à genoux, on peut donc incriminer directement l'application.

Nous allons donc chercher à savoir ce qu'elle fait. Un réflexe classique, qu'un certain nombre d'entre nous a eu, est d'aller ouvrir les logs. Catalina.out est vide, on n'a pas de répertoire de logs applicatives évident, à priori le coupable classique, la configuration log4j, est à écarter. Et pourtant... Pour savoir ce que fait réellement notre application, il est possible de réaliser une série de thread dumps. Pour cela, deux possibilités :

la vieille école, qui va dumper le contenu des threads à l'aide d'un kill -3 sur la jvm. l'école moderne, qui va utiliser [JVisualVm](#) (et [ses plugins](#)) pour réaliser la même opération à l'aide d'une belle interface.

Et là, demie surprise, notre thread actif est souvent surpris dans la méthode

org.apache.log4j.spi.LoggingEvent.getLocationInformation. Nous aurions donc un logger actif. Nos soupçons initiaux se confirment, ne reste plus qu'à débusquer le coupable.

Nous n'avons pas (encore) le code source. Or, cette bombe plombe tellement l'application qu'il n'est pas envisageable que nous ne puissions pas la résoudre de suite. On doit donc avoir un fichier de paramétrage de la log externalisé. Là encore, VisualVm va nous aider. Dans la fenêtre overview, les paramètres de démarrage de la VM sont affichés. Et l'on voit apparaître une directive de configuration log4j, -Dlog4j.configuration.

Un petit grep dans le répertoire de lancement de Tomcat (\$TOMCAT\_HOME/bin), et nous voyons apparaître

dans le setEnv.sh la ligne -Dlog4j.configuration=file:\$CATALINA\_HOME/conf/log.xml. Et si nous ouvrons ce fichier, le root Logger est bien en DEBUG. Mais aucune directive spécifique pour logger ailleurs que dans la console. Tout devrait donc aller dans catalina.out. Ca sent la redirection 'sauvage'. Continuons à parcourir les fichiers de lancement. Un petit tour dans catalina.sh, et surprise, une belle redirection Unix "\$CATALINA\_BASE"/logs/catalina.out 1> /var/log/catalina/tomcat.log &  
Et en ouvrant ce fichier, on comprend mieux le temps perdu : des milliers de lignes de debug !

Celles ci ont d'ailleurs une particularité sympathique : à chaque ligne de debug, le numero de ligne de la classa Java est indiqué.

C'est pratique ! Pourquoi ne met on pas systématiquement en place cette configuration sur nos projets ? Un petit tour dans la documentation Log4J nous apprend :

Log4J nous prévient, ces méthodes sont sous performantes. Si l'on creuse un peu dans le source, on trouve la classe affichant les pattern %M et %L, org.apache.log4j.spi.LocationInfo, qui contient le constructeur suivant :

```
public LocationInfo(Throwable t, String fqncOfCallingClass) {  
if(t == null || fqncOfCallingClass == null)  
return;
```

```
String s;
```

```
// Protect against multiple access to sw.
```

```
synchronized(sw) {
```

```
t.printStackTrace(pw);
```

```
s = sw.toString();
```

```
sw.getBuffer().setLength(0);
```

```
}
```

```
[...]
```

Autrement dit, pour chaque ligne de log, Log4J génère une stackTrace pour pouvoir récupérer le numéro de la ligne et le nom de la méthode. Pas vraiment performant.

Modifions donc le pattern et observons le résultat.

Suppression de %M:%L dans le pattern du Logger Temps moyen de 2,3 s à 2,0 s pour 1 utilisateur 2 points pour l'équipe qui a trouvé

Toujours concernant ce fichier de log, nous voyons apparaître une ligne ne respectant pas le PatternLayout de Log4J, préfixée par Hibernate, qui trace une requête SQL. Gardons cette ligne en mémoire, nous y repenserons quand nous aurons le code source.

Deuxième amélioration immédiate envisageable, une application, qui plus est en production, ne devrait pas avoir besoin d'un niveau de log aussi fin. Passons le donc à ERROR.

Logger de DEBUG à ERROR Temps moyen de 2,0 s à 1,75 s pour 1 utilisateur 1 point pour l'équipe qui a trouvé

Nous avons fait un sort au logger. Mais nous n'en avons pas encore fini avec la configuration out-of-the-box de cette application.

Source

[Image libre de droits](#)

<http://blog.xebia.fr/2010/01/27/performance-les-xebians-jouent-les-demineurs/>

---

**Formation certifiante ScrumMaster par Arlen Bankston**

Xebia propose des formations certifiantes ScrumMaster animées par Arlen Bankston.

Cette formation répondra, entre autres, aux questions suivantes :

Comment planifier et faire des estimations avec Scrum ? Comment un chef de projet traditionnel devient-il un chef de projet agile ? Comment faire travailler l'analyste fonctionnel avec les équipes agiles ? Comment fonctionnent les reportings et les métriques avec Scrum ? Comment travailler avec des équipes distribuées en Scrum ? Comment savoir si un projet est compatible avec Scrum ? Quels sont les outils principaux de Scrum ? Comment s'assurer d'un résultat cohérent ? Comment aménager la salle d'une équipe agile ?

Arlen Bankston (Formateur ScrumMaster certifié et Lean Six Sigma Master Black Belt). La formation a lieu en anglais.

Arlen Bankston est un leader reconnu dans la mise en œuvre des méthodologies de management tel que Lean, Six Sigma, BPM mais aussi dans les processus de développement logiciel SCRUM et XP.

Arlen a mené avec succès des projets agiles d'envergure pour des entreprises telles que Capital One, T.Rowe Price, NBC Universal, Saudi Telecom, United States Army et Freddie Mac pour ne citer qu'elles.

Arlen est aussi le Vice Président de [LitheSpeed](#), un cabinet international spécialisé dans la mise en place des techniques Lean et Six Sigma et des méthodologies Scrum et XP.

La prochaine session aura lieu les **8 et 9 mars 2010**.

[Télécharger le programme >>](#)

Renseignements et inscriptions par mail : [info@xebia.fr](mailto:info@xebia.fr) ou en appelant le **06 09 69 05 49**.

<http://blog.xebia.fr/2010/01/27/formation-certifiante-scrum-master-par-arlen-bankston/>

---

**Revue de Presse Xebia**

La revue de presse de l'actualité Java/J2EE hebdomadaire proposée par Xebia.

## Actualité éditeurs / SSII

[Tomcat 6.0.24, version mineure mais non négligeable](#)

## Agilité

[Le pair-programming comment ça marche ?](#)

## RIA

[Firebug passe en version 1.5](#) Actualité éditeurs / SSII Tomcat 6.0.24, version mineure mais non négligeable  
Nous allons commencer par un clin d'oeil humoristique, même les plus grands ont des difficultés à builder, le projet Tomcat a brûlé trois tags subversion pour faire une release sans défaut, il y a eu de nombreux problèmes d'adaptation des sauts de ligne aux plateformes Windows/Unix-Linux .

Ensuite, la sécurité avec la correction de la faille [CVE-2009-3555](#) appelée "SSL-Man-In-The-Middle attack" qui intéressera ceux qui gèrent SSL avec Tomcat.

Enfin, nous avons retenu deux nouvelles fonctionnalités qui nous paraissent particulièrement utiles :

L'intégration de la [RemotelpValve](#) pour connaître l'adresse IP de l'internaute et le protocole (http/https) utilisé lorsque Tomcat est précédé de load balancers et/ou d'un serveur web. Nous en avons longuement parlé dans [Tomcat : Adresse IP de l'internaute, load balancer, reverse proxy et header Http X-Forwarded-For](#) et dans [Tomcat, SSL, communications sécurisées et X-Forwarded-Proto](#).

Exemple :

Fragment de server.xml pour déclarer le RemotelpValve avec utilisation de X-Forwarded-Proto :

L'intégration du [JmxRemoteLifecycleListener](#) qui permet de figer le deuxième port d'écoute RMI et ainsi se connecter avec [VisualVM](#) et un tunnel SSH pour passer les firewalls. A nous le profiling sur les serveurs de production ! Mais attention aux chausse-trappes :

il faut télécharger le [catalina-jmx-remote.jar](#) et le copier sous \$TOMCAT\_HOME/lib, un vilain problème de UnmarshalException/ClassNotFoundException dans les clients comme Hyperic HQ si vous activez useLocalPorts (contournement [ici](#)), les propriétés à définir dans setenv.sh (ou catalina.properties) en plus de la déclaration dans server.xml.

L'occasion d'une contribution ?

Exemple :

Fragment de setenv.sh pour déclarer des variables d'environnement de JmxRemoteLifecycleListener :

```
-Dcom.sun.management.jmxremote.ssl=false  
-Dcom.sun.management.jmxremote.authenticate=false
```

Fragment de server.xml pour déclarer le JmxRemoteLifecycleListener :

Agilité Le pair-programming comment ça marche ?

Stuart Wray de la Royal School of Signals, s'est fendu d'un article sur le fonctionnement du pair-programming dans l'édition de Janvier du magazine de l'IEEE. Dans l'article, il liste quatre bonnes pratiques permettant de garantir selon ses termes, l'efficacité du développement par pair. Il ne s'agit pas de simplement développer à deux l'un au clavier et l'autre le doigt sur l'écran.

Dialogue entre les développeurs :

Il est important de verbaliser les problèmes rencontrés de façon intelligible. Présenter une difficulté à un tiers permet de garder le focus dessus et l'effort de présentation force à clarifier la situation. Il faut sortir la tête du mur pour résoudre les problèmes et parfois le seul fait de présenter le blocage permet de trouver la solution. L'auteur identifie la conversation comme un point clé permettant aux paires d'être et de rester productifs. Voir

plus de détails :

Un phénomène bien connu de tous, deux personnes ne voient pas les mêmes choses au même moment. C'est à cela que sert la relecture sur le blog Xebia. Il y a toujours des fautes d'inattention, liés à des centres d'intérêts différents par exemple. En paire, un développeur trouve plus rapidement certaines erreurs que son collègue qui en voit d'autres. Celui qui ne tape pas au clavier repère toujours beaucoup plus rapidement les coquilles.

Stuart lève notre attention sur la fatigue des paires, en travaillant ensemble, les développeurs commencent à repérer les mêmes erreurs et à fixer leur attention sur les mêmes points. La productivité est alors en chute libre et l'apport du pair-programming se perd. Il faut prévoir des rotations régulières entre les paires pour se prémunir contre cet effet de bord. Combattre les mauvaises pratiques :

Les deux développeurs doivent prendre l'engagement de coder en respectant une convention, en utilisant des bonnes pratiques. Chacun est juge du travail de son voisin, pourtant à terme c'est le travail des deux ensemble qui sera jugé. Cet engagement nécessaire des deux parties augmente la responsabilité de chacun sur la qualité du code produit. Posez-vous la question : faites-vous plus propre en développant seul dans votre cave ou en développant sous le regard attentif de votre paire ? Partager et juger l'expertise :

Aucun individu n'a la même productivité et la différence varie au moins d'un facteur de un à dix entre deux individus. Cela implique des erreurs d'estimation temporelle par exemple si votre héros Java qui code les yeux fermés annonce une journée pour une tâche et qu'elle est réalisée par un autre le temps passé ne sera sûrement pas d'une journée (Tous le monde ne peut pas coder les yeux fermés). C'est seulement en collaborant étroitement avec un développeur qu'il devient possible de juger de ses capacités. Mais avec le pair programming et la rotation des paires chaque développeur connaît les domaines d'expertise des autres et sait se positionner par rapport aux autres. Les estimations seront donc plus réalistes dans une équipe en pair programming. [La news sur InfoQL'article complet](#) RIAFirebug passe en version 1.5

[Firebug](#), le plugin Firefox ultime pour le débogage de nos applications web, passe en [version 1.5](#) (via [Ajaxian](#)).

Pour rappel, cet outil permet d'inspecter notre code HTML, CSS et Javascript, de l'éditer avec répercussions directes sur la page, d'analyser les flux XMLHttpRequest avec le nombre d'appels effectués ainsi que leur durée... En bref, un outil incontournable !

Les corrections de bugs sont [nombreuses](#) et les nouvelles fonctionnalités sont aussi au [rendez-vous](#). On appréciera ainsi un mode d'inspection encore plus robuste, une boîte d'information rapide lors de l'inspection, l'ajout du [bouton de persistance](#), le blocage du Javascript et des événements sur un breakpoint, les nombreux [nouveaux breakpoints](#) ou bien encore l'[explorateur XML](#) pour une réponse de type XML. Ces deux derniers points seront à coup sûr des fonctionnalités clés pour le débogage de nos applications.

Si vous ne le possédez pas encore, c'est par [ici](#) que cela se passe.

<http://blog.xebia.fr/2010/01/25/revue-de-presse-xebia-144/>

---