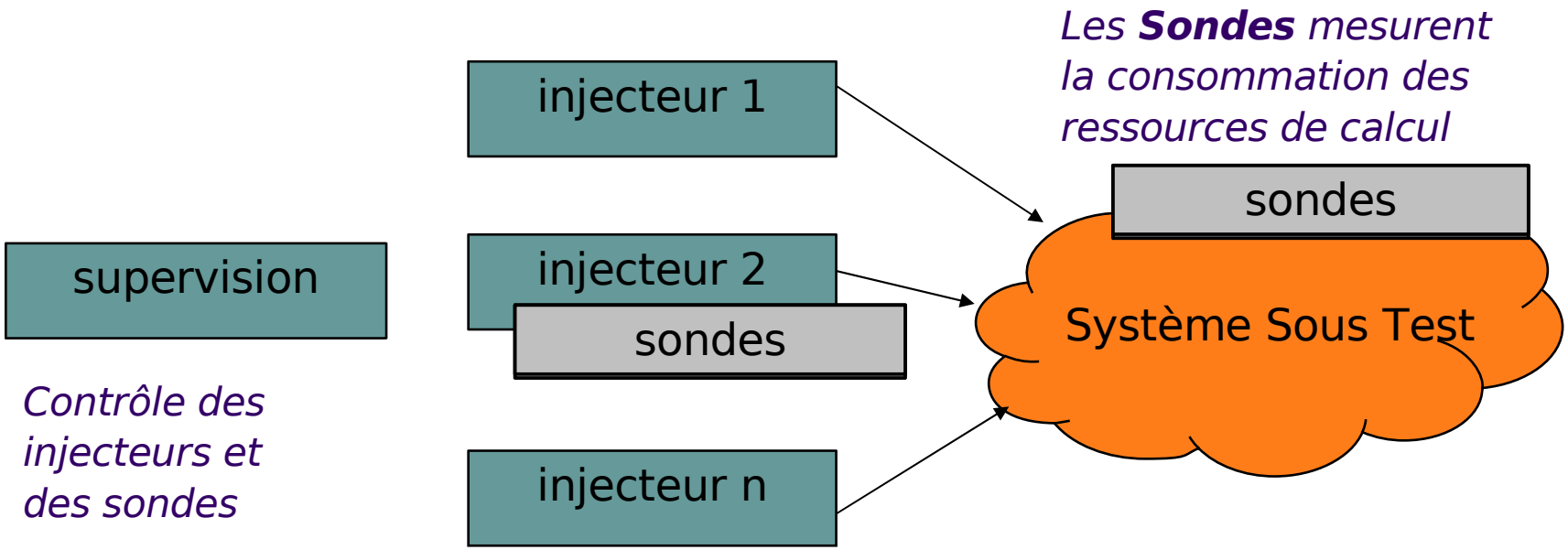
Pratique de la prémétrologie à Orange Labs à travers l'utilisation de la plate-forme de test en charge CLIF

Bruno Dillenseger,
Orange Labs, laboratoire MAPS/AMS
28, chemin du Vieux Chêne,
38243 Meylan cedex
bruno.dillenseger@orange-ftgroup.com

Principe du test de performance par injection de charge



*Les **injecteurs de charge :***

- *émettent des requêtes, attendent les réponses, et mesure le **temps de réponse***
- *selon un certain **scénario***
- *par exemple, en émulant le trafic d'utilisateurs réels par des **utilisateurs virtuels***

Plan

(1) Problématique de la prémétrologie des performances

(2) Le canevas logiciel libre CLIF pour le test en charge

(3) Exemples d'application de CLIF à France Télécom R&D

(4) Axe de recherche : vers des tests autonomes

1. Problématique de la prémétrologie

Besoin de test de performance *agile*...

Les enjeux de la maîtrise des performances

Fiabilité et rapidité des services

- échelle des conséquences de problèmes de performance sur le service :
  - lenteur,
  - plantage,
  - perte d'intégrité
- gains ou pertes de transactions
- image de confiance (gains ou pertes d'audience)

Rentabilité et faisabilité opérationnelle

- réduction des coûts de dépannage/maintenance
- dimensionnement optimal des infrastructures

  nombre de serveurs, consommation d'énergie, refroidissement, surface occupée

Forces et limites de la métrologie des performances

- La métrologie des performances est une affaire de spécialistes qualifiés et expérimentés.
  - utilisation d'outils logiciels puissants/riches et réputés fiables
  - utilisation d'une infrastructure technique complexe et potentiellement à large échelle, qu'il faut maîtriser
  - détection et identification de problèmes éventuels
  - recherche des origines des problèmes identifiés
- A l'instar du test en général :
  - la métrologie des performances prend du temps et coûte de l'argent
    - compromis avec le retour sur investissement (ROI) 
    - compromis avec le délai de mise sur le marché (TTM) 
  - la correction des défauts est d'autant plus difficile que la métrologie intervient tardivement dans le cycle de développement (qualification finale)

Enjeux de la prémétrologie

- La prémétrologie vise à anticiper les problèmes de performance en généralisant la pratique du test de performance au cours du cycle conception-développement.
  - tests généralement réalisés par les développeurs ou intégrateurs
  - tests d'éléments avant intégration, ou de parties d'éléments logiciels en cours de développement
  - utilisation possible d'outils d'injection de charge ad hoc, qui aident à révéler certains défauts liés aux performances
- On peut alors détecter et corriger certains problèmes au plus tôt
  - La correction est plus rapide, moins difficile, moins coûteuse.
- La prémétrologie peut se compléter par un travail de modélisation puis d'analyse ou de simulation
  - e.g. pour valider une architecture, détecter des goulets d'étranglement

Le besoin d'un outillage « agile »

La généralisation de la prémétrologie suppose de disposer d'outils « agiles » :

- pour s'adapter à des profils d'utilisateur variés
  - définition des scénarios (graphique, programmatique, scripts)
  - interface utilisateur (interface graphique, ligne de commande...)
- pour s'adapter à des cas de tests très variés et particuliers
  - systèmes cibles (protocoles d'invocation, sondes)
  - degrés de passage à l'échelle (e.g. de la dizaine aux millions d'utilisateurs virtuels)
- pour d'adapter à des objectifs de qualification différents
- sans contrainte lourde de déploiement
  - limitation sur l'environnement d'exécution (e.g. système d'exploitation)
  - termes de la licence
- sans contrainte lourde de budget

Aperçu de l'outillage disponible

Constat

- Les outils de mesure de performance par injection de charge existent par centaines.
- Quelques produits commerciaux phare et une myriade de projets gratuits voire en logiciel libre

Question

- Pourquoi tous ces développeurs refont-ils leur plate-forme ?

Réponses possibles

- Coût élevé et contraintes de licence des outils commerciaux
- Besoin de maîtriser ce qu'on injecte et ce qu'on mesure pour interpréter correctement (plus difficile qu'il n'y paraît a priori !)
- Adaptabilité de l'outil (SST, passage à l'échelle, objectif particulier de qualification, profil d'utilisateur)

Vers un canevas logiciel pour le test en charge « agile »

Le défi est de disposer d'un outil de test sur mesure pour chaque nouveau cas de test

- techniquement accessible pour les utilisateurs
- rapidement disponible
- au moindre coût

Approche :

Un canevas logiciel (libre) hautement et rapidement adaptable permettant de factoriser de façon optimale des fonctions communes

- infrastructure répartie de déploiement/supervision des sondes et injecteurs
- interfaces utilisateur de contrôle des tests (graphique, ligne de commande...)
- définition de scénarios de charge
- stockage des mesures
- analyses statistiques
- production de rapports de test

2. Le canevas logiciel CLIF

CLIF is a Load Injection Framework

CLIF is a Load Injection Framework

CLIF est un canevas logiciel à composants pour l'injection de charge et la mesure de performance

- adaptable et extensible :
  - systèmes sous test variés (protocoles, sondes...)
  - modes de définition des scénarios de charge
  - interfaces utilisateurs et intégrations diverses
  - multi-système d'exploitation (basé sur Java)
- architecture à composants Fractal
  - modèle récursif (hiérarchie + partage de composants)
  - adaptabilité à l'échelle du composant de niveau optimal
  - configuration à l'aide d'un langage de description d'architecture (« ADL »)
- grande puissance de charge
  - injection de charge répartie
  - jusqu'à plusieurs millions d'utilisateurs virtuels par injecteur (cf. module complémentaire ISAC)

CLIF : la synergie du logiciel libre

CLIF est un logiciel libre du consortium ObjectWeb/OW2

ObjectWeb
Open Source Middleware

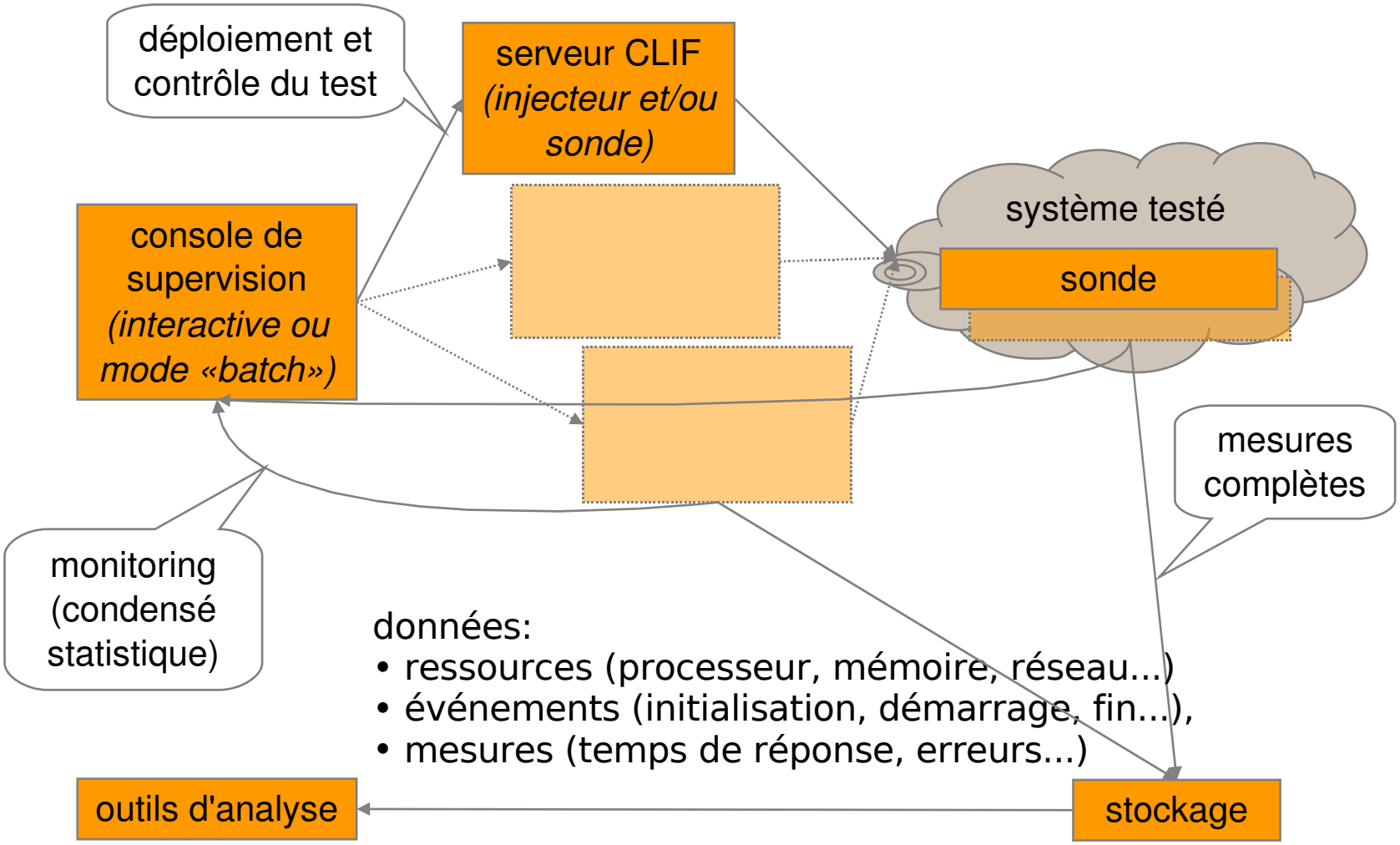
OW2
Consortium

- Licence LGPL
- Lancé en 2002 par France Télécom R&D et l'INRIA
- Lutèce d'or 2007 du *meilleur projet libre réalisé par un grand groupe*

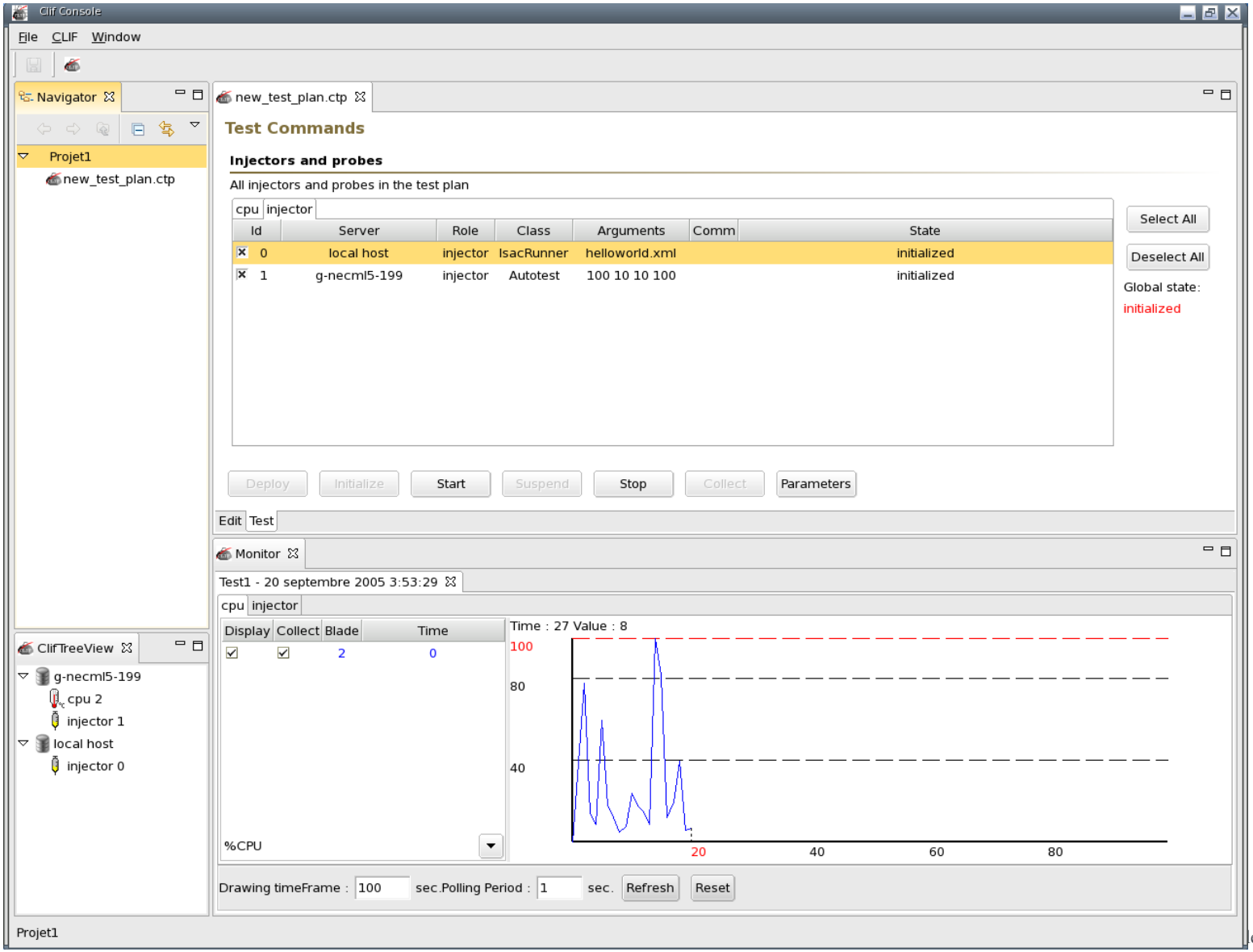
CLIF réutilise un grand nombre d'autres projets libres

Fractal (OW2), HttpClient (Apache), Htmlparser (Source Forge), Eclipse, PostgreSQL JDBC driver, DnsJava, MaxQ (Tigris), JDOM, OpenLDAP, JAIN-SIP...

CLIF : infrastructure répartie



Console CLIF basée sur Eclipse RCP



Sondes CLIF

- Sondes système disponibles pour Linux, Windows et MacOSX:
  - *cpu* / utilisation des processeurs
  - *memory* / utilisation mémoire et swap
  - *network* / réception, émission
  - *disk* / lectures, écritures disques

- Sondes *JVM* pour suivre l'utilisation de la mémoire et le Garbage Collector des machines virtuelles Java

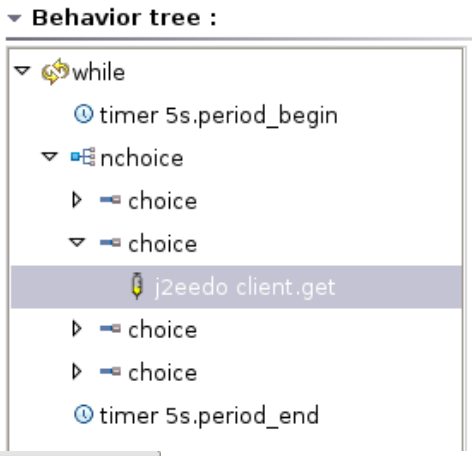
- Cadre pour définir ses propres sondes

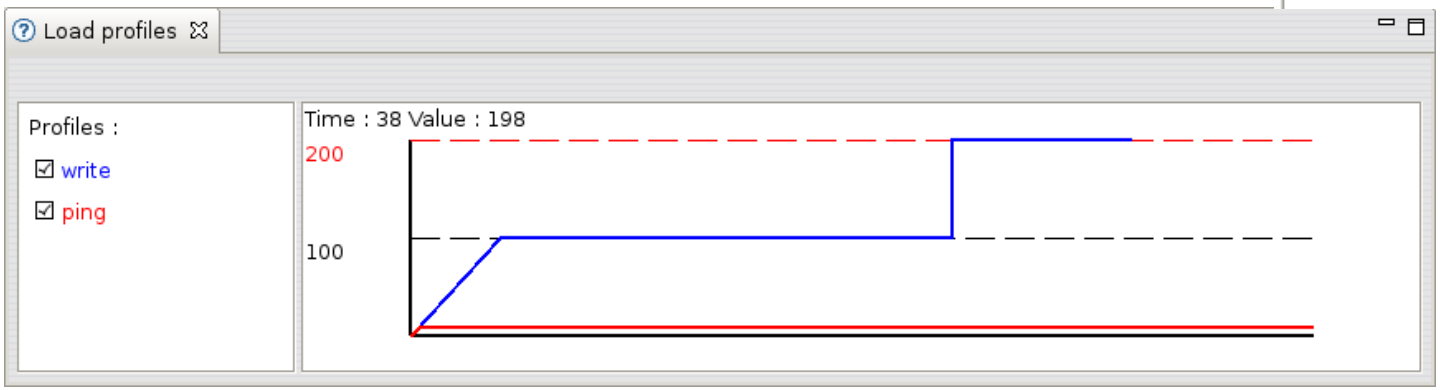
  sondes SNMP, JMX, ...

ISAC is a Scenario Architecture for CLIF

- outil générique de définition de scénarios formels
- un scénario combine :
  - la définition de comportements séquentiels (e.g. comportement des utilisateurs du SST)
    - requêtes sur le SST, temps de pause (attente/*think time*)
    - structure logique : *if-then-else*, boucle, choix probabiliste, préemption
  - et un profil de charge pour chaque comportement

    évolution du nombre d'instances actives (i.e. utilisateurs virtuels) avec le temps





Plug-ins ISAC

Les comportements définissent une structure logique qui utilise des plug-ins pour réaliser concrètement les tests :

- Injecteurs

  TCP, UDP, TCP, DNS, HTTP(S), JDBC, JMS, DHCP, LDAP, SIP...

- Timers

  délai fixe ou aléatoire avec diverses distributions, délais calculés...

- Fournisseurs de jeux de données

  compteur, lecture de champs dans un fichier au format CSV ou lecture d'un fichier entier, transformation et extraction de chaînes de caractères, valeurs aléatoires...

- Fournisseurs de fonctions booléennes pour les instructions conditionnelles (if/while/preemption)

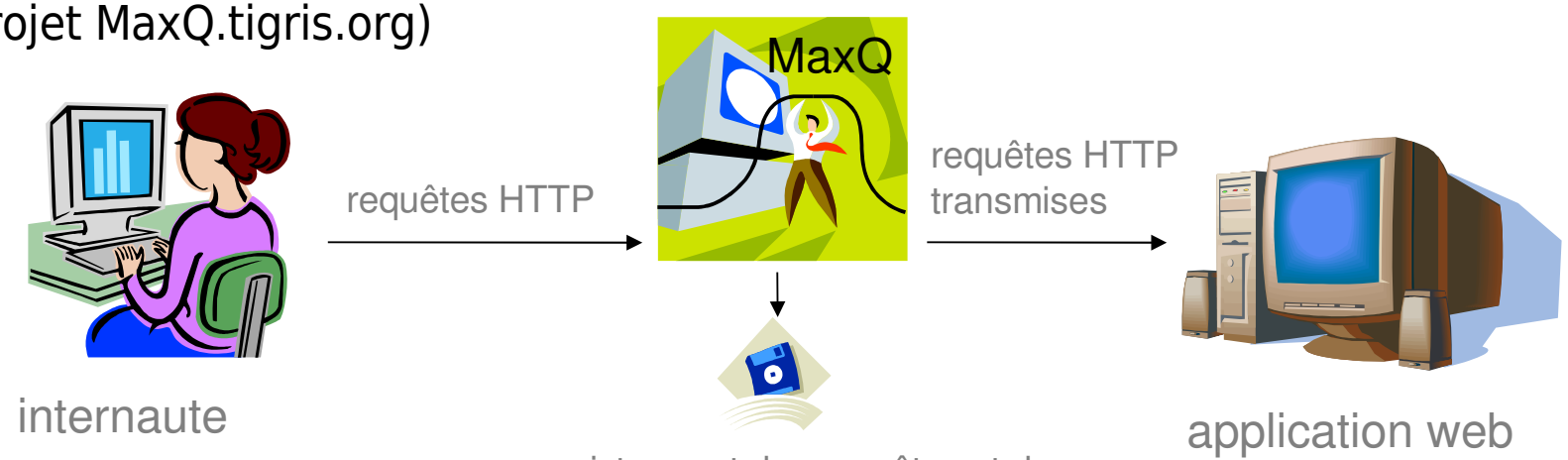
Un assistant Eclipse (*wizard*) permet de créer ses *plug-ins* ISAC.

Outils de capture de comportement

Chaque plug-in ISAC injecteur peut être associé à un outil de capture produisant des comportements ISAC pour rejouer des sessions réelles.

Exemple :

proxy HTTP produisant des scénarios ISAC pour le plug-in HttpInjector (projet MaxQ.tigris.org)



3. Premiers retours d'expérience

Mise en œuvre de CLIF à France Télécom R&D dans des contextes techniques divers

Le contexte du Groupe France Télécom

En tant qu'opérateur de télécommunication intégré, Orange rassemble une grande diversité :

- réseaux fixes/mobiles/sans fil/domestique
- trafic voix, voix sur IP, vidéo, données...
- systèmes répartis à large échelle : infrastructures réseau, systèmes d'information et de commande, plates-formes de service
- équipes R&D spécialisées sur les réseaux fixes, réseaux mobiles, services, middleware...
- une multitude de protocoles/technologies, d'objectifs de performance, de compétences

Echelle : plus de 161 millions d'abonnés dans 220 pays

- La qualité de service est la priorité numéro 1 du groupe
- Le dimensionnement et le test en charge sont donc essentiels

CLIF à France Télécom R&D (1/2)

- CLIF a été appliqué à une trentaine de projets R&D
  - utilisation en prémétrologie avant transfert, métrologie et déploiement
  - utilisation en métrologie en l'absence d'autre outil
- Les capacités d'adaptation ont été largement exploitées
  - Temps classique pour intégrer une sonde ou un injecteur : 2 jours (moyennant la disponibilité d'une librairie cliente pour le protocole ou la ressource cible)

Exemples :

- middleware orienté message (contexte Machine-to-machine)
  - utilisation d'injecteurs et sondes spécifiques afin de récupérer les temps de réponse globaux et intermédiaires dans des échanges de message asynchrones
- plate-forme de réservation de ressources réseau
  - un injecteur spécifique a été défini pour ce protocole propriétaire

CLIF à France Télécom R&D (2/2)

Exemples (suite) :

- Trafic SOAP/HTTP(S) sur un ESB matériel (*Enterprise Service Bus*) dans une architecture SOA
  - le but était de mesurer les performances de l'ESB (pas des services)
  - les services ont été remplacés par des “bouchons”
  - utilisation de sondes SNMP et JMX
  - utilisation de l'injecteur HTTP et rejeu de requêtes SOAP
- DNS en cluster
  - haut débit (8000 requêtes/s par injecteur de charge)
  - test très consommateur de ressource réseau, nécessitant un *tuning* système des injecteurs de charge
- Logiciel de gestion de tests Salomé-TMF
  - injecteurs spécifiques pour l'API Java de Salomé-TMF
  - 40 injecteurs de charge en parallèle
- serveurs DHCP, SGBD mémoire (JDBC)...

4. Axe de recherche

Vers l'autonomie dans les campagnes de test de performance

Pratique classique du test en charge

Problème

On cherche les limites de performance d'un système inconnu en fonction de critères de qualité de service donnés

Pratique

- paramétrage du système sous test et du système d'injection
- exécution du test
- mesure des temps de réponse, des débits...
- mesure de la consommation des ressources
- analyse des résultats (typiquement *post mortem*)

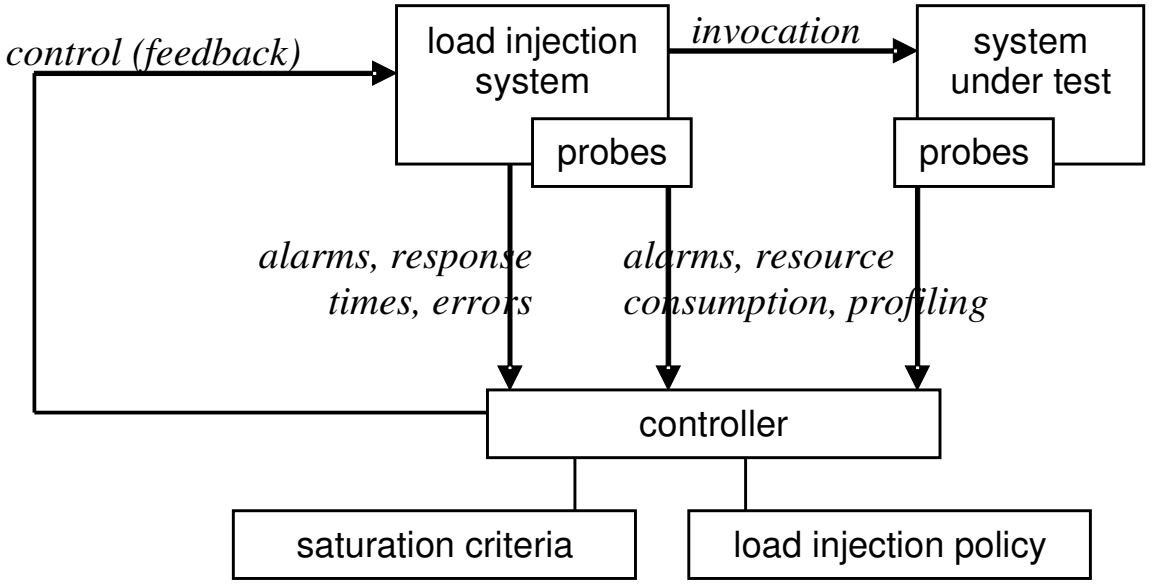
... et on REBOUCLE !

Idée : introduire des fonctionnalités autonomes de rétroaction (rebouclage) pour permettre aux testeurs de gagner du temps

Recherche autonome de limite de saturation

On réalise une injection de charge autorégulée recherchant les limites de performance (saturation) d'un système sous test.

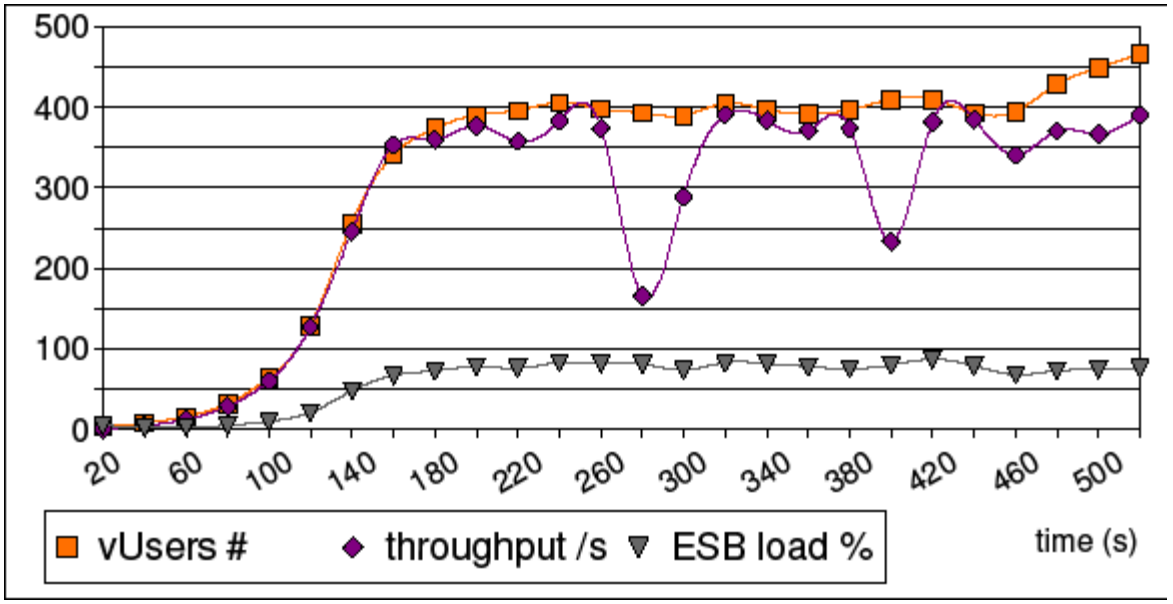
La canevas logiciel CLIF est complété par un composant de contrôle de l'injection.



Exemple : saturation de l'ESB matériel

Expérience

- charge SOAP/HTTPS : ajustement du nombre d'utilisateurs virtuels
- sonde SNMP observant la charge de l'ESB : objectif de 80%



→ résultats conformes aux résultats de la campagne de test classique

Conclusion

- La pratique du test est toujours soumise à un double compromis
  - entre la qualité et le coût (*Return On Investment*)
  - entre la qualité et les délais (*Time-To-Market*)
- La prémétrologie permet de désengorger l'étape de métrologie
  - anticipation sur les problèmes de performance lorsqu'ils sont plus faciles à résoudre, à proximité de ceux qui peuvent les résoudre
  - outils moins éprouvés mais sur mesure
- CLIF outil de prémétrologie “universel”
  - approche architecturale à composants pour une adaptabilité optimale
  - logiciel libre, aux fonctionnalités avancées par rapport aux autres (généricité, sondes, répartition, scénarios ISAC...)
  - disponible sur tout environnement supportant Java
- Travaux en cours :
  - développement d'outils d'analyse et de production de rapports
  - recherche sur l'extension de l'architecture de CLIF pour introduire des capacités de test autonome
  - complémentarité avec la simulation pour les systèmes de grande taille