

LibX

Logiciel

libre

en page 3

Actualités

hc2.ch, a success story made in EPFL

Christian Kauth

1

DIT-info

Yves Despond

2

Où l'informatique s'expose

Appoline Raposo de Barbosa

2

Analyse

Mot-croisé: NŒUD

Vittoria Rezzonico, Esteban Rosales
& Yves Roucher

8

Découvrir la virtualisation avancée

Laurent Kling

11

Déploiement continu, cas du moteur de recherche de l'EPFL

Maciej Marcowicz

15

Logiciel libre

LibX, la recherche documentaire en deux clics

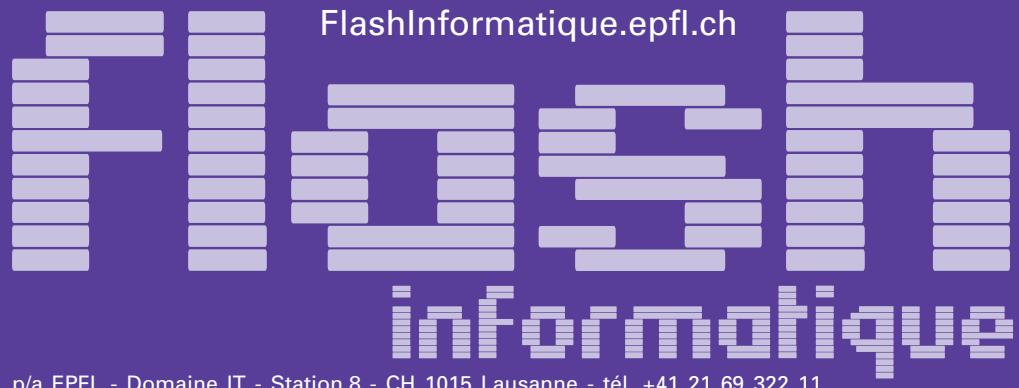
Raphaël Grolimund

3

Prochaines parutions

No	Délai de rédaction	Parution
4	14.05.13	11.06.13
5	27.06.13	16.07.13
6	15.08.13	13.09.13

tout public
public averti
expert



p/a EPFL - Domaine IT - Station 8 - CH 1015 Lausanne - tél. +41 21 69 322 11

» Actualités



hc².ch, a success story made in EPFL

Christian.Kauth@epfl.ch, EPFL - STI, coach de PolyProg et doctorant en microsystèmes et microélectronique

The 125 brave students who participated in the 2013 edition of the Helvetic Coding Contest found in the fifteen proposed algorithmic tasks a hard nut to crack. To be successful, one needs to be up to every trick! Allow yourself a short trip to the amazing world of brain-teasers...

Les 125 étudiants intrépides qui ont participé à l'édition 2013 de l'Helvetic Coding Contest se sont brisé les dents sur une quinzaine de problèmes algorithmiques très exigeants. Pour y réussir, il fallait avoir plus d'un tour dans son sac ! Accordez-vous une petite escapade au monde des casse-têtes...



.../.. Suite page 9



DIT-info

Changement à la TI



Jaouher Garreb nous a quitté à fin mars pour se consacrer à un nouveau travail hors de l'EPFL avec nos vœux de plein succès pour la suite de sa carrière.



Nicolas Repond quittera son poste aux Services de Base pour rejoindre le groupe Téléinformatique début mai et occuper le poste laissé vacant par Jahouer. Nous souhaitons la bienvenue à Nicolas.

Yves.Despond@epfl.ch, Domaine IT



Où l'informatique s'expose

Appoline.Raposo@epfl.ch, EPFL - DIT

Le futur site du Musée cantonal des Beaux-Arts abrite une exposition avant même les travaux de transformation. La page www.visartevaud.ch vous donnera toutes les informations au sujet de cette exposition.

De l'inachevé

**Halles CFF – gare de Lausanne
du 4 au 19 mai 2013**

Vernissage vendredi 3 mai – 18h00

Une halle en presque déshérence. Un lieu entre deux usages. Des rails récemment à l'abandon, un site ferroviaire qui veut devenir lieu de culture. Visarte Vaud vient se poser sur cet entre-deux, ouvrir la friche au regard de la population, déposer une empreinte, un souffle...

Christian Jelk

Cinq longues tables déroulées sur les rails de ces anciennes halles CFF exposeront plusieurs centaines d'œuvres sélectionnées par

cinq artistes qui agissent en curateurs. Leurs choix s'opèrent selon cinq indices programmatiques autour de la notion de l'inachevé, dessin dépouillé, embryon, pulsion, sismique, vandale. L'ensemble trace la géographie d'un art à l'état natif, surgissant à la périphérie de la culture achevée et en avance sur le futur musée.

Table embryon

Patricia Glave, curatrice de cette table, invite entre autres des scientifiques: «*Sur ma table d'exposition, mes invités, artistes, sculpteurs, plasticiens, peintres, mais aussi mathématiciens, architectes, designers... Je pourrai ainsi montrer: comment, on peut faire des liens entre un artiste et un scientifique quand ils dessinent une pensée.*

Parmi ses invités se trouvent quelques scientifiques de l'EPFL comme Kathryn Hess Bellwald, Varvara Karpova et Antonin Danovalet qui a écrit dans le Fl2/2013: **Quand le WiFi se met au service du réseau piétonnier**, flashinformatique.epfl.ch/spip.php?article2642. ■

Impressum

Revue consacrée aux technologies de l'information, éditée par le Domaine IT de l'EPFL (DIT). Les articles n'engagent que leurs auteurs, sauf ceux qui concernent de façon évidente des prestations officielles (sous la responsabilité du DIT ou d'autres entités). Toute reproduction, même partielle, n'est autorisée qu'avec l'accord de la rédaction et des auteurs.

Rédacteur en chef a.i.:

Appoline Raposo de Barbosa, fi@epfl.ch

Mise en page & graphisme:

Appoline Raposo de Barbosa

Comité de rédaction:

Jean-Daniel Bonjour, Sébastien Ferrara, Patrice Fumasoli, Florence Hagen, Laurent Kling, Julia Paolini, François Roulet, Christophe Salzmann & Predrag Viceić

Impression: Atelier de Reprographie EPFL

Tirage: 4000 exemplaires

Adresse Web: flashinformatique.epfl.ch

Adresse: Domaine IT EPFL

Station 8, CH-1015 Lausanne

Téléphone: +41 21 69 32246 & 32247

Abonnement au Fl par e-mail à:
fi-subscribe@listes.epfl.ch



LibX, la recherche documentaire en deux clics

Raphael.Grolimund@epfl.ch, EPFL, bibliothécaire



Wouldn't it be nice to search for papers (or books) directly from the web browser without visiting the website? LibX makes it possible.

Ne serait-il pas agréable de pouvoir effectuer une recherche documentaire directement depuis son navigateur sans devoir se rendre sur le site Web interrogé ? Avec LibX, c'est possible.

Fiche descriptive

LibX		
Domaine		
◆ barre de recherche documentaire pour navigateur Web		
Licence	langue	version
◆ Mozilla Public License	◆ multilingue	◆ 2.0
Alternatives non libres		
◆ MyLibraryToolbar		
Sites Web		
◆ Projet: www.libx.org		
Plates-formes supportées		
◆ (extension pour les navigateurs Firefox et Chrome)		

Introduction

LibX est un module complémentaire, respectivement une extension pour Firefox et Chrome¹, qui a été développé à Virginia Tech (États-Unis).

LibX est un logiciel libre particulier car, en plus des développeurs et des utilisateurs, intervient une troisième catégorie, que nous appellerons les gestionnaires de bibliothèques (universitaires), qui jouent un rôle important. Alors que les développeurs s'occupent du logiciel LibX (le contenant), de sa compatibilité avec le navigateur et de sa mise à jour, le gestionnaire crée une une barre d'outils² dans laquelle il personnalise les bases de données et les champs interrogables (le contenu) pour ses besoins ou ceux de sa communauté. N'importe qui peut créer une édition et choisir

les ressources qu'il souhaite interroger, mais pour schématiser, on peut dire que le développement du module LibX est un travail d'informaticien, alors que la personnalisation du contenu de la barre d'outils est du ressort d'un(e) bibliothécaire. LibX a d'ailleurs été développé conjointement par le département d'informatique et les bibliothèques de Virgina Tech. Il est en effet destiné à interroger des catalogues de bibliothèques (comme NEBIS) et des bases de données documentaires (comme Web of Science). Nous verrons toutefois que LibX n'est pas limité aux ressources documentaires. Le code de LibX est déposé sur Github: <https://github.com/godmar/LibX>. Un outil en ligne, appelé Edition Builder, est mis à disposition des gestionnaires de bibliothèques pour créer leur barre d'outils: libx.org:8080/editionbuilder/src/zul/.

Les utilisateurs, eux, peuvent trouver toutes les éditions publiques (et donc téléchargeables) dans l'annuaire: libx.org/editions. L'annuaire compte plus de 1000 éditions publiques.

Dans cet article, nous allons d'abord voir comment utiliser LibX en tant qu'utilisateur final. Ensuite, nous verrons comment créer une édition et y inclure les ressources qu'on souhaite pouvoir interroger.

Installation

The screenshot shows the LibX Main Page. At the top, there's a welcome message: "Welcome to LibX! This site may look sparse, but LibX 2.0 is fully alive and has a large and devoted user community." Below this, a message says: "All our development efforts right now is devoted to LibX 2.0, which is available for the Firefox and Google Chrome browsers. You can give LibX 2.0 a spin right here and check if an edition exists for your community. If not, keep in mind that anyone can create editions using the Edition Builder." There are two buttons: "Install Bibliothèque de l'EPFL (Barre d'outils avancée) for Firefox" and "Search for an edition for my community". The main area features a search bar with the placeholder "Search > NEBIS for...". Below the search bar are links for "Mots-clés", "Search", "Links", and "About". To the right, there's a section titled "Bibliothèque de l'EPFL (Barre d'outils avancée)" with a search bar containing "NEBIS for...". Below this are buttons for "Reset fields" and "Search". At the bottom, there's a link to "Simple view". A note at the bottom left says: "The following editions were recommended for you based on your IP address of 128.178.195.196:" followed by a list of recommended editions.

fig. 1 – propositions de barres d'outils sur le site Web de LibX et aperçu de l'une d'entre elles

LibX s'intègre dans les navigateurs Web Mozilla Firefox et Google Chrome. Il fonctionne donc sur tous les systèmes d'exploitation où ces navigateurs peuvent être installés. Pour le moment, LibX n'est pas disponible sur les plates-formes mobiles, mais la meilleure manière de proposer LibX sur mobile est en discussion au sein de la communauté.

¹ D'après les données qui m'ont été fournies par Nathalie Meystre du KIS (qui en est remerciée ici), Firefox et Chrome étaient les deux navigateurs les plus utilisés pour accéder au site Web de l'EPFL en 2012 (35.5% pour Chrome, 22.5% pour Firefox).

² Dans cet article, les termes **édition de LibX** et **barre d'outils** sont utilisés comme synonymes.

LibX, la recherche documentaire en deux clics

L'installation de LibX se fait depuis libx.org. Le site vous propose une sélection de barres d'outils en se basant sur votre adresse IP. Si vous êtes à l'EPFL, ce sont les barres d'outils développées l'année dernière par la Bibliothèque de l'EPFL qui vous seront proposées³. En choisissant une édition, le site vous en présente un aperçu (fig. 1).

Puis, en cliquant sur le bouton **Install Bibliothèque de l'EPFL...**, c'est l'installation classique de n'importe quel module complémentaire pour Firefox⁴ ou extension dans Chrome⁵.

Utilisation

Fonctionnalités de base

LibX permet de chercher dans des ressources sans se rendre sur le site Web en question: cliquez sur le bouton **LibX** (ici, frappé de l'icône EPFL) dans la barre de navigation, choisissez la base de données et vous êtes prêts à chercher (fig. 2). Le résultat de la requête s'affiche dans un nouvel onglet.

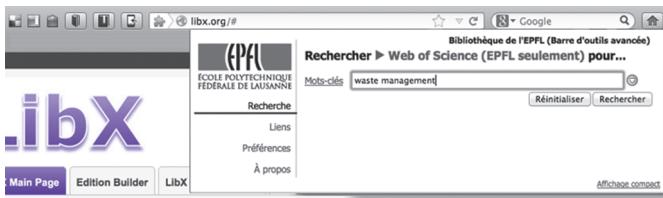


fig. 2 – recherche simple dans Web of Science depuis Firefox

Le moteur de recherche intégré au navigateur peut très bien le faire. Alors quel intérêt d'utiliser LibX ? Réponse dans la figure 3.



fig. 3 – recherche multicritère dans Google Scholar depuis Firefox

LibX permet de lancer des requêtes multicritères en cliquant sur l'icône circulaire (v) à droite du champ de recherche. De plus, relancer la même requête dans une autre ressource est aussi simple que de la sélectionner et valider. Les champs et les valeurs sont conservés. LibX permet également au gestionnaire de bibliothèques de sélectionner des liens qu'il estime utile et de les rendre accessibles dans la section Liens (fig. 4).

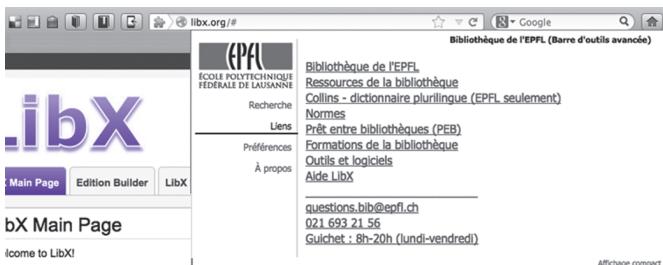


fig. 4 – sélection de liens proposée par la Bibliothèque de l'EPFL

³ Elles sont également disponibles ici: library.epfl.ch/tools/?pg=toolbar

⁴ Autorisation de l'installation, téléchargement, installation et redémarrage du navigateur.

⁵ Sélection de l'extension, installation et ajout au navigateur (pas de redémarrage).

Les comportements décrits ici peuvent être modifiés par l'utilisateur dans le panneau de **Préférences** (accessible par la section Préférences).

Fonctionnalités plus avancées

Mais LibX ne se contente pas de vous permettre de chercher depuis la barre d'outils. Si vous souhaitez effectuer une recherche sur un terme (ou une expression) lu sur une page Web, vous n'avez qu'à le sélectionner, faire un clic-droit (menu contextuel du navigateur) et choisir la ressource dans laquelle chercher (fig. 5).

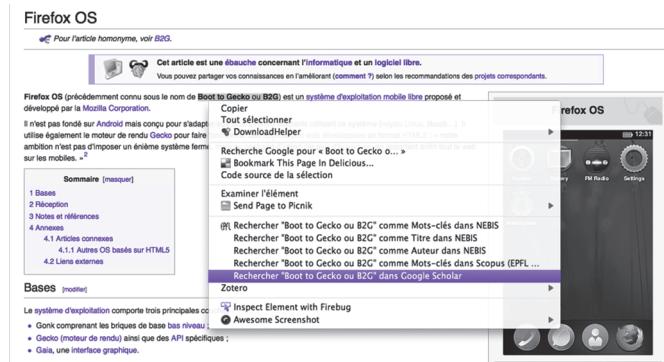


fig. 5 – recherche directement à partir d'une page Web

LibX peut aussi vérifier si votre bibliothèque possède le livre que vous cherchez. En passant le pointeur sur l'ISBN, LibX vérifie et affiche sa disponibilité et, s'il le trouve, donne accès d'un simple clic à sa notice dans le catalogue de la bibliothèque. Pratique pour savoir si vous pouvez emprunter (gratuitement) le livre que vous êtes sur le point d'acheter !

Détails sur le produit

Rélié: 208 pages

Éditeur: Orbit (22 février 2012)

Collection: orbit

Langue: Français

ISBN-10: 2360510452

ISBN-13: 2360510452 refers to: "Survivre à une invasion

Dimension: robot : manuel pratique" Daniel H. Wilson ; traduit de l'anglais (États-Unis) par Patrick Imbert., Orbit,

Moyenne: 4.5 (100 évaluations) (Voir les 100 premiers en Livres)

Classement: (Voir les 100 premiers en Livres)

fig. 6 – livre trouvé sur Amazon détecté dans le réseau NEBIS

Là encore, les ressources qui apparaissent dans le menu contextuel peuvent être personnalisées par l'utilisateur (section **Préférences**). Si vous êtes en échange dans une autre université, vous pouvez facilement changer d'édition en allant dans la section **À propos** (fig. 7). En cas de problème ou de question, c'est également là que vous trouvez les informations de contact du gestionnaire de l'édition que vous utilisez.

Le changement d'édition ne nécessite pas de redémarrage du navigateur. LibX charge simplement les données de la nouvelle édition à la place de l'ancienne.

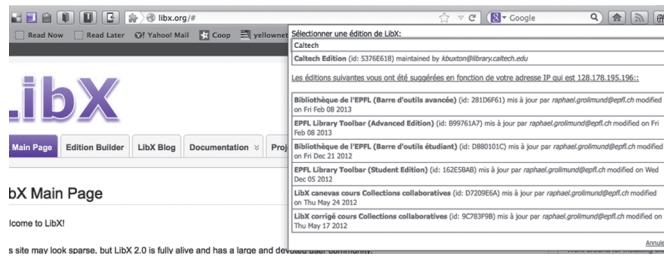


fig. 7 – changement d'édition directement dans LibX (sans téléchargement)

LibX, la recherche documentaire en deux clics

Finalement, si vous trouvez que la fenêtre de LibX est trop grande et que vous n'avez pas besoin de toutes les options en permanence (Affichage complet), vous pouvez opter pour l'affichage compact, réduit au strict minimum (fig. 8).



fig. 8 – affichage complet (à gauche) et compact (à droite) de LibX

Configuration

Passons maintenant à la création et la personnalisation d'une édition de LibX.

La création d'une édition nécessite un compte (gratuit) sur l'Edition Builder à l'adresse libx.org:8080/editionbuilder/src/zul/. Lorsque vous avez déjà un compte, vous entrez simplement vos nom d'utilisateur et mot de passe. Si c'est la première fois que vous vous rendez sur l'Edition Builder, entrez une adresse e-mail comme nom d'utilisateur et un mot de passe. Lorsque vous cliquez sur le bouton **Log on**, la plate-forme vous indique que l'utilisateur n'existe pas et vous propose de créer un compte (fig. 9). Cliquez donc sur le bouton **Register**. Une fenêtre vous demande si vous souhaitez vous inscrire à la liste de diffusion. Faites votre choix et validez.



fig. 9 – fenêtre de connexion et de création d'un compte

Maintenant que vous avez un compte, passons à la création d'une édition personnalisée.

S'il est possible de créer une édition en partant de zéro, vous pouvez aussi vous baser sur l'une des éditions publiques existantes (fig. 10).

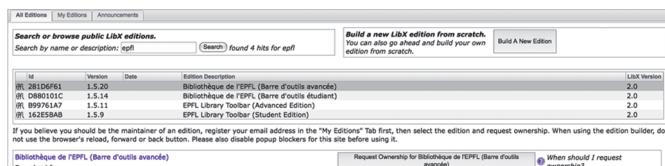


fig. 10 – choix de départ entre une édition préexistante (en bas à droite) ou une édition vierge (en haut à droite)

En partant d'une édition préexistante, il est possible de s'épargner beaucoup de travail. Comme nous allons le voir un peu plus loin, les bases de données comme **Web of Science** ou **Engineering Village** (entre autres) peuvent poser des problèmes de configuration. En trouvant une édition proche de vos besoins, vous gagnerez donc beaucoup de temps. Pour les autres, la liste de diffusion⁶ (et ses archives) seront d'une grande utilité.

⁶ Inscription sur <https://www.mozilla.org/mailman/listinfo/libx> (consulté le 30 mars 2013)

Pour connaître le contenu d'une édition publique, copiez-la dans votre compte grâce au bouton **Make copy of ...** (voir fig. 10). Une copie de l'édition est ajoutée dans la liste de vos éditions et huit nouveaux onglets apparaissent en haut de la page (fig. 11). C'est le point de départ de la création de votre barre d'outils personnalisée, mais c'est aussi comme ça que vous pouvez découvrir la configuration de n'importe quelle barre d'outils publique.

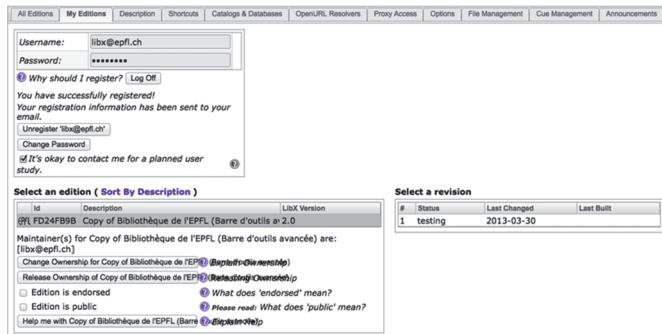


fig. 11 – édition préexistante de LibX copiée dans son propre compte

Passons rapidement en revue le contenu de chacun de ces onglets.

Description: informations générales sur cette édition;

Shortcut: liste de liens utiles (favoris);

Catalogs & Databases: le cœur d'une barre d'outils et cet article (développé un peu plus loin);

OpenURL Resolver: configuration du résolveur de liens qui fait le lien entre une référence bibliographique et le texte intégral (à l'EPFL, nous utilisons SFX d'Ex Libris⁷ – voir encadré);

Proxy Access: paramètres du proxy (inutile à l'EPFL, car l'accès aux ressources est basé sur l'adresse IP);

Options: gestion des paramètres particuliers liés au fonctionnement de la barre d'outils;

File Management: gestion des logos apparaissant dans l'interface;

Cue Management: gestion des paquets permettant d'étendre les fonctionnalités d'une barre d'outils.

Pourquoi un résolveur de liens ?

Contrairement au site Web d'un éditeur (comme ScienceDirect d'Elsevier), une base de données bibliographique ne comprend que des références des articles et non le texte intégral. Les bases de données les plus utilisées à l'EPFL (Web of Science, Scopus et Engineering Village) sont des bases de données bibliographiques. Or, le chercheur qui consulte ces plates-formes ne s'intéresse pas seulement à la référence. Il a besoin de pouvoir lire l'article. C'est là que le résolveur de liens entre en scène. S'il trouve la référence dans sa base de connaissances (où les abonnements souscrits par la bibliothèque sont enregistrés), il affiche un bouton dans la base de données bibliographique indiquant que le texte intégral est accessible. Ce bouton est un lien menant vers le texte intégral sur sa plate-forme d'origine.

⁷ www.exlibrisgroup.com/category/SFXOverview (consulté le 30 mars 2013)

LibX, la recherche documentaire en deux clics

Le cœur de l'Edition Builder est dans l'onglet **Catalogs & Databases**. C'est là que les ressources sélectionnées sont paramétrées. Il y a deux types de ressources: les catalogues de bibliothèques et les autres.

Les catalogues de bibliothèques sont préconfigurés dans LibX en fonction du **système intégré de gestion de bibliothèque (SIGB)** utilisé. À la Bibliothèque de l'EPFL (et dans l'ensemble du réseau NEBIS), ce logiciel est ALEPH d'Ex Libris (fig. 12).

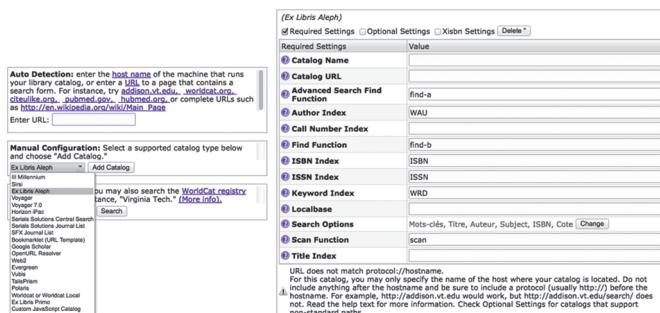


fig. 12 – la liste des logiciels pris en charge (Manual Configuration, à gauche) et la grille ALEPH préremplie (à droite).

La plupart des logiciels utilisés en bibliothèque sont pris en charge par LibX. Ce n'est toutefois pas le cas de Virtua de VTLS⁸, le SIGB utilisé par nos voisins de la Bibliothèque cantonale et universitaire (BCU) de Lausanne et l'ensemble du réseau **RERO**.

Dans leur cas, c'est la liste des membres de WorldCat qui offre la solution (*WorldCat Registry*, fig. 13). L'import du profil se fait ensuite via l'outil Auto Detection (illustré dans la fig. 14).



fig. 13: - profil trouvé en interrogeant la liste des membres de WorldCat.

La configuration d'un catalogue de bibliothèque, c'est bien. Mais là où LibX devient vraiment intéressant, c'est dans la configuration d'une base de données. Le travail repose sur l'analyse de l'url d'une requête. Voyons cela avec une base de données documentaire propre à l'EPFL: **Infoscience**.

Une recherche dans Infoscience génère une url contenant trois paramètres: <https://infoscience.epfl.ch/search?ln=fr&p=grolimund&f=>. Les deux premiers nous intéressent, car ils permettent d'inclure la langue (ln) et le mot-clé entré par l'utilisateur (p) dans la requête. Pour ajouter Infoscience dans une édition de LibX, il faut passer par l'outil Auto Detection, taper une url (infoscience.epfl.ch) et cliquez sur le bouton **Add**. Notez que vous trouverez Infoscience, car il a déjà été ajouté dans les barres d'outils de la Bibliothèque de l'EPFL. Si vous ne trouvez rien de satisfaisant, copiez n'importe quelle ressource et modifiez tous les paramètres.

Dans la figure 14, vous pouvez voir que l'url qui renseigne le champ *Bookmarklet URL Template* est un peu différent de celui que j'ai indiqué plus haut: infoscience.epfl.ch/search?ln=fr&p=%Y. La valeur %Y s'est glissée dans l'url. C'est ainsi que LibX gère l'endroit où les mots-clés entrés par l'utilisateur sont inclus.

Prenons un deuxième exemple: **Google Scholar**. C'est l'url d'une recherche avancée qui nous intéresse: scholar.google.com/scholar?hl=fr&lr=&as_publication=College%20Research%20Libraries&q=zotero&author=Puckett. Cette fois-ci, nous avons passé trois valeurs dans la requête: zotero comme mot-clé (q), Puckett comme auteur (author) et College Research Libraries comme nom de journal (as_publication). Une fois dans LibX, cette url ressemble à cela: scholar.google.ch/scholar?hl=fr&q=%Y&author=%oa&as_publication=%jt.

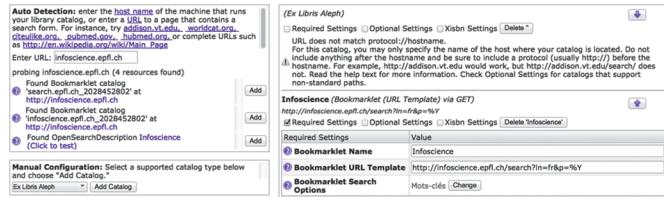


fig. 14 – Infoscience et les 3 champs qui le décrivent (nom, url et options)

Cet exemple met en lumière la capacité de LibX à effectuer des requêtes multicritères. Les valeurs %Y, %oa et %jt permettent d'inclure la bonne valeur au bon endroit. Les libellés de ces paramètres sont gérés à l'échelle de l'édition. Par défaut, %Y s'applique au paramètre mot-clé, mais vous pouvez très bien le remplacer par le libellé %keyword. De même, %jt pourrait être %journal.

Le mécanisme de LibX dévoilé, vous comprenez maintenant que n'importe quel outil dont l'url est analysable de la sorte peut être intégré dans une édition de LibX. Voici quelques exemples⁹:

- Annuaire de l'EPFL (recherche des doctorants du CRAFT): search.epfl.ch/search.action?unit=craft&klass=Doctorant;
- Twitter (tweet de @lessig incluant le hashtag #cc): <https://twitter.com/search?q=%23cc%20from%3Alessig>;
- Flickr (recherches d'images de l'EPFL sous licences Creative Commons): www.flickr.com/search/?q=epfl<=cc&ct=0&tmt=al&tadv=1;
- Mozilla Firefox (modules complémentaires compatibles avec Firefox 19 pour Mac OS X faisant des captures d'écran): <https://addons.mozilla.org/fr/firefox/search/?q=screenshot&apvver=19.0&platform=mac>;

Mais que se passe-t-il si l'url ne contient pas les paramètres de la requête ? Là, ça devient problématique. C'est le cas de ressources documentaires importantes comme *Web of Science* et *Engineering Village* dont les urls ressemblent respectivement à: apps.webofknowledge.com/summary.do?SID=R15bF7dDOAM1GPMPLgB&product=UA&qid=1&search_mode=GeneralSearch et www.engineeringvillage.com/controller/servlet/Controller?CID=expertSearch *CitationFormat*.

Ces urls sont absolument inutilisables. Comme il s'agit de ressources documentaires importantes, des personnes ont cherché une solution à ce problème et y sont parvenues. Leurs éditions étant publiques, leur travail bénéficie à tout le monde. Par contre, dans le cas de ressources non documentaires, il y a peu de chance que quelqu'un ait cherché une solution pour l'inclure dans LibX.

⁹ Pour des raisons de lisibilité, certaines urls ont été nettoyées. Seuls les paramètres utiles pour l'exemple ont été conservés. Vous n'obtiendrez donc pas forcément les mêmes urls si vous essayez par vous-mêmes.

⁸ www.vtls.com/products/virtua (consulté le 30 mars 2013)

À l'EPFL

L'année dernière, la Bibliothèque de l'EPFL a créé quatre éditions de LibX: deux en français et deux en anglais. À chaque fois, l'une des éditions est destinée aux étudiants de l'EPFL, alors que l'autre (édition avancée) est prévue pour les doctorants et chercheurs. Cette dernière intègre les trois bases de données les plus utilisées à l'EPFL (*Web of Science, Scopus, Engineering Village*), en plus des ressources de l'édition pour les étudiants.

Il va de soi que ces barres d'outils ne répondent pas à tous les besoins particuliers de chaque section de l'EPFL. Mais le contenu d'une barre d'une telle barre d'outils peut s'adapter à vos besoins et c'est la raison d'être de cet article. Si vous souhaitez créer une barre d'outils personnalisée, lancez-vous ! Les barres d'outils de la Bibliothèque de l'EPFL seront sans doute un bon point de départ.

Aide et support

En plus de l'aide que vous pouvez requérir auprès de la Bibliothèque¹⁰, vous pouvez vous appuyez sur la (maigre) documentation en ligne (libx.org/documentation/), les lectures indiquées à la fin de l'article et surtout la très bonne liste de diffusion (très utile et réactive).

LibX est un outil simple et efficace. Il faut tout de même noter que l'interface de l'*Edition Builder* est déroutante de prime abord. Mais une fois passé la surprise du début, on se fait au fonctionnement de l'outil.

Autre point étonnant de prime abord: les mises à jour d'une édition se font sur une plate-forme en ligne. Pas besoin de télécharger quoi que ce soit. La mise à jour est déployée depuis les serveurs de LibX.

¹⁰ Instructions d'utilisation disponibles:
library.epfl.ch/tools/?pg=toolbar-use

Pour aller plus loin...

- [1] BAILEY, Annette et BACK, Godmar, 2006. *LibX – a Firefox extension for enhanced library access*. In: Library Hi Tech [en ligne]. 1 avril 2006. Vol. 24, n° 2, pp. 290-304. [Consulté le 30 mars 2013]. DOI 10.1108/07378830610669646. Disponible à l'adresse: www.emeraldinsight.com/journals.htm?issn=0737-8831&volume=24&tissue=2&articleid=1558881&show=abstract.
- [2] NICHOLSON, Brian Robert, 2011. *LibX 2.0* [en ligne]. Thèse de Master. Blacksburg, VA: Virginia Polytechnic Institute and State University. Disponible à l'adresse: scholar.lib.vt.edu/theses/available/etd-12202011-204944/unrestricted/Nicholson_BR_T_2011.pdf.
- [3] PUCKETT, Jason, 2010. *Superpower your browser with LibX and Zotero: Open source tools for research*. In: College and Research Libraries News [en ligne]. 2010. Vol. 71, n° 2, pp. 70-74 97. Disponible à l'adresse: crln.acrl.org/content/71/2/70.full.pdf+html.
- [4] RITTERBUSH, Jon, 2007. *Supporting Library Research with LibX and Zotero*. In: Journal of Web Librarianship [en ligne]. 2007. Vol. 1, n° 3, pp. 111-122. DOI 10.1300/J502v01n03_08. Disponible à l'adresse: www.tandfonline.com/doi/abs/10.1300/J502v01n03_08.

Cette liste de référence est disponible et mise à jour sur go.epfl.ch/libx.



Article du FI-EPFL 2013 sous licence CC BY-SA 3.0 / R. Grolimund

GLOSSAIRE

Engineering Village: base de données bibliographique d'Elsevier regroupant des dizaines de millions de publications dans les domaines de l'ingénierie.

Google Scholar: moteur de recherche de Google affichant un sous-ensemble de résultats ne comprenant que des publications académiques et scientifiques.

Infoscience: dépôt institutionnel de l'EPFL où les collaborateurs de l'EPFL peuvent déposer leurs publications.

ISBN (International Standard Book Number): numéro unique à 10 chiffres permettant d'identifier chaque édition de chaque livre publié. L'ISBN ayant été

introduit à la fin des années 1970, les livres parus avant n'en possèdent pas. L'ISBN est passé à 13 chiffres en 2007.

RERO: réseau des bibliothèques universitaires romandes. Ce réseau inclus notamment des universités de Lausanne, Genève et Fribourg.

Scopus: base de données bibliographique d'Elsevier contenant les publications parues dans plus de 19'000 journaux scientifiques.

Système intégré de gestion de bibliothèque (SIBG): logiciel utilisé pour gérer tous les aspects de la gestion d'une bibliothèque, de l'achat de nouveaux documents jusqu'à leur prêt et la gestion de comptes des lecteurs.

Web of Science: bases de données bibliographiques de Thomson Reuters regroupant toutes les publications faites dans une sélection de journaux et conférences. Le fameux Impact Factor est basé sur les publications présentes dans cette base (et celle-là seulement !).

WorldCat: métacatalogue comprenant les données des quelques 10'000 bibliothèques membres dans le monde. Ce catalogue affiche aujourd'hui plus de 1.5 milliards de documents.



Un mot: nœud – quelques regards: mathématique, étymologique, marine et illustration.

Nœud de calcul, nœud de login, ... – VR

En HPC (*Calcul à Hautes Performances*), les nœuds sont le cœur et le cerveau d'un supercalculateur. Les *clusters* sont composés de nœuds (de calcul, de login ou d'administration) ainsi que de stockage spécialisé. Le tout est relié par du réseau, qui peut être haute vitesse, basse latence ou bien du Gigabit Ethernet standard. Le (ou les) nœud d'administration s'occupe de tout ce qui est gestion de services (*backup*, installation,...).

Le (ou les) nœud de login, aussi appelé *frontend*, est la partie visible sur le réseau, accessible par les utilisateurs. Un *cluster* de calcul, même s'il s'agit d'un ensemble d'entités, n'apparaît sur le réseau que comme une unique entité.

Enfin, nous avons les nœuds de calcul, nombreux, et tous les mêmes. Si l'un d'entre eux est défaillant et doit être remplacé, une fois le matériel arrivé, le nœud d'administration rend le remplaçant immédiatement opérationnel en l'installant et l'intégrant dans l'environnement. La distribution du travail sur les nœuds est gérée par l'ordonnanceur de tâches (*scheduler*), qui connaît à tout instant la charge des nœuds et s'occupe d'assigner les tâches en attente selon des règles dictées par une politique donnée, mais aussi selon le bon sens afin d'optimiser l'utilisation des ressources. Les supercalculateurs les plus puissants au monde comptent des dizaines de milliers de nœuds de calcul. Les supercalculateurs se trouvant sur le site de l'EPFL comptent, respectivement:

- Electra: 24 nœuds (mais aussi des GPU)
- Antares: 56 nœuds
- Aries: 44 nœuds
- Bellatrix: 408 nœuds
- BlueGene/Q Lemanicus: 1024 nœuds

En plus de cela, ces *clusters* comportent tous au moins un nœud de login. Certains ont aussi des nœuds d'administration ainsi que des nœuds spécialisés.

Références

- hpc.epfl.ch/ressources.php
- hpc-dit.epfl.ch
- bluegene.epfl.ch
- Du nouveau pour le HPC: Blue Gene/Q Lemanicus (Fl 2/2013, flashinformatique.epfl.ch/spip.php?article2639)
- Bellatrix – le nouveau cluster de calcul de l'EPFL géré par le DIT (Fl 8/2012, flashinformatique.epfl.ch/spip.php?article2607)

Nœud de vipère – ER



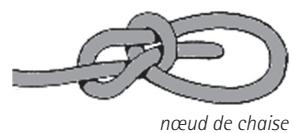
Nœud marin – YR

Parlons seulement ici du nœud qui sert à attacher sur un bateau, pas de celui qui sert à mesurer, car là c'est encore une autre histoire.

Et des histoires de nœuds, les marins pourraient vous en raconter pendant des soirées entières sans se lasser.

Quoi de plus beau qu'un nœud marin bien fait ? On peut l'admirer sous tous ses angles. Chaque nœud a son esthétique personnelle et son utilité aussi sur un bateau et bien sûr un nom. On ne va pas tous les énumérer ici, ce serait trop long.

L'art des nœuds dans le monde maritime s'appelle le matelotage. Mais que peut-on faire avec un nœud sur un bateau ?



Votre bateau arrive au port, il faut bien l'attacher. Dans ce cas, on fera un **nœud de chaise**. Et croyez-moi, vous pouvez tirer sur le cordage, le nœud de chaise ne va pas se défaire qu'il soit

mot-croisé: NŒUD

sec ou mouillé. Vous pouvez être sûr que votre bateau est bien attaché. Et à votre retour, avant de repartir sur l'eau, vous pourrez le défaire facilement sans problème, car il n'est pas serré. C'est là toute sa force. Ce nœud de chaise se classe dans la catégorie des **nœuds d'amarrage**.

Et dans cette catégorie, on trouve aussi le **nœud de cabestan**, le **nœud en huit**, le **nœud de capucin**. Ils ont tous leur particularité et leur utilité.



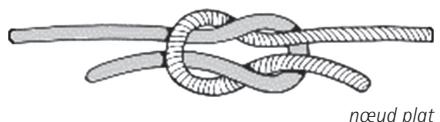
nœud en huit



nœud de capucin

Dans une autre catégorie, on trouve les **nœuds d'arrêt**. Par exemple, le demi-nœud qui, lui, ne sert pas contre pas à grand-chose. Et une fois serré, il est impossible à défaire. Si on veut un nœud d'arrêt plus sérieux, ce sera le **nœud en huit** qui se fait rapidement, mais peut aussi se défaire sans difficulté, même après avoir tiré dessus.

Peut-être vous arrive-t-il d'avoir un cordage (un bout pour les marins) trop court. Alors pour rallonger temporairement avec un autre bout, quoi de mieux qu'un **nœud plat**. Il est facile à faire, s'adapte à tous les diamètres de cordage et ne glissera pas.



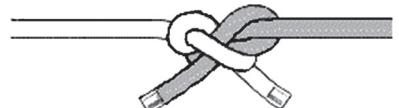
nœud plat

Dans cette même catégorie des **nœuds d'assemblage**, on trouvera le **nœud de pêcheur** plus facile à redéfaire, mais tout aussi efficace.



nœud de pêcheur

Un petit dernier pour le voyage, le **nœud de carrick** plus difficile, mais aussi très élégant.



nœud de carrick

Que de nœuds ! On ne peut que vous souhaiter: Bon vent quand vous prendrez la mer pour de longs voyages et bon apprentissage des nœuds ! ■

hc².ch, a success story made in EPFL

.../... Suite de la première page

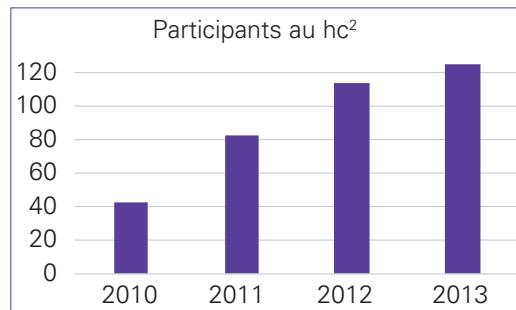
Il est devenu culte et le rendez-vous annuel par excellence pour tout programmeur qui se respecte – l'**Helvetica Coding Contest** (hc²) a rassemblé en mars sur le campus 125 étudiants afflués des quatre coins du pays. L'équipe d'organisation de PolyProg a su, une fois de plus, servir ce qui va droit au cœur de tout programmeur: une série de casse-têtes algorithmiques stimulants disponibles en trois langues, assaisonnée par un riche buffet et des présentations techniques, le tout dans une ambiance amicale et agréable.

Esprit d'équipe, créativité et know-how

Une inscription via hc2.ch, un petit problème de qualification, nécessaire pour harmoniser la demande avec l'offre, et tout allait comme sur des roulettes. À la sortie du métro, il suffisait de se laisser guider par les flèches pour s'immerger dans l'ambiance hc² et ce, dès l'entrée dans les bâtiments du Centre Ouest. Check-in rapide, photo d'équipe et un café-croissant pour bien démarrer la journée en causette. Une sonnerie de cor annonça la partie officielle de la journée, qui commença par un mot de bienvenue et quelques présentations des sponsors Open Systems et AdNovum. Elle fut suivie d'un léger concours à blanc pour se familiariser avec l'environnement: chaque équipe tricéphale disposait d'un seul ordinateur pour résoudre les problèmes et y coder une solution en C/C++ ou Java. Le code était à soumettre via une interface Web, qui répondait quelques secondes plus tard avec le verdict. Dès que le code avait résolu les tests correctement dans le délai imparti, un



nouveau problème, plus épique, devenait accessible à l'équipe. À la fin de cet échauffement, un buffet bien garni attendait les participants. Le moment de faire le plein en énergie pour quatre heures et demie de concours intensives. Esprit d'équipe, créativité et compétences algorithmiques furent de mise pour résoudre un maximum de problèmes. Des posters présentés en fin de journée révélèrent les solutions. La journée se clôtura par la remise des prix pour les meilleurs et pour les plus chanceux. Les vainqueurs 2013 étudient à l'ETHZ et se classent parmi les jeunes programmeurs les plus talentueux du monde.

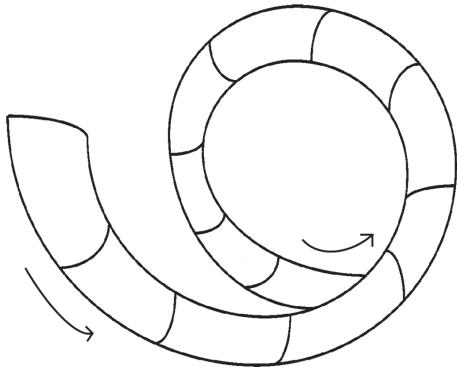


Nombre de participants aux concours hc² depuis 2010

Si la mémoire fait défaut

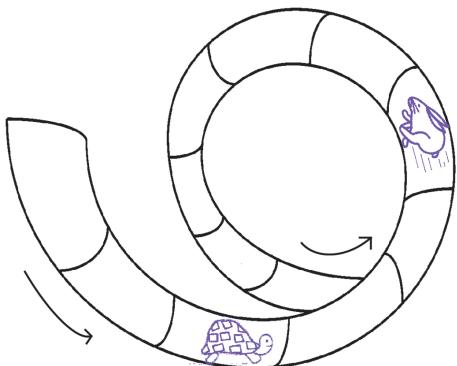
Je vous propose de saisir papier et stylo et d'essayer de résoudre un problème du concours qui a piégé deux tiers des équipes qui s'y sont attaquées. Un vrai régal dont la solution ne présuppose

aucune connaissance en programmation. Imaginez une liste de chiffres uniques, qui est rebouclée sur elle-même, comme l'illustre la figure ci-dessous. Cette liste ne peut être traversée que dans un sens (indiqué par la flèche) et consiste en une partie linéaire suivie d'une partie cyclique, qui fait que vous pouvez passer l'éternité à la traverser. On vous donne le début de cette liste et vous demande combien d'éléments uniques elle contient. Pour ce faire, vous pouvez demander l'identité de la case suivante pour n'importe quelle case de votre choix. Cette information vous permet de traverser la liste. Puisque chaque case est caractérisée par un identifiant unique, vous avez moyen de distinguer les cases. L'approche qui saute à l'œil consiste à mémoriser les identifiants, et d'arrêter la traversée dès qu'un duplicita est trouvé, car ce serait la fermeture du cycle. Ceci est correct, mais malheureusement vous ne disposez de loin pas d'assez de mémoire pour mémoriser tous les identifiants des cases – bienvenu(e) à l'hc² !



Le lièvre et la tortue

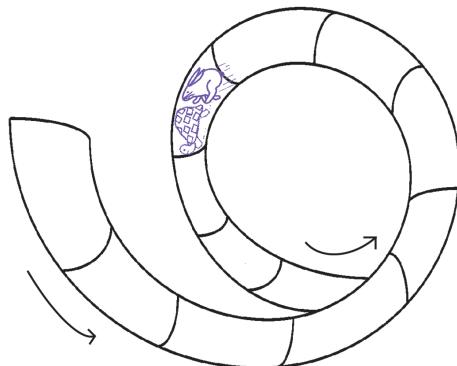
L'astuce consiste en plusieurs traversées de la liste à des vitesses différentes. Imaginez une tortue et un lièvre, qui partent en même temps, avec des vitesses de 1 et 2 respectivement. Supposez que la partie linéaire de la liste contienne L (3 dans l'exemple) éléments, tandis que la partie cyclique en est composée de C (9 dans notre cas). Vu que le lièvre court deux fois plus vite que la tortue, et que la partie linéaire a une longueur L , il aura une avance de L sur la tortue au moment où celle-ci s'apprête à entrer dans le cycle. À vrai dire, le lièvre court en cercle de longueur C , donc son avance effective vaut L modulo C .



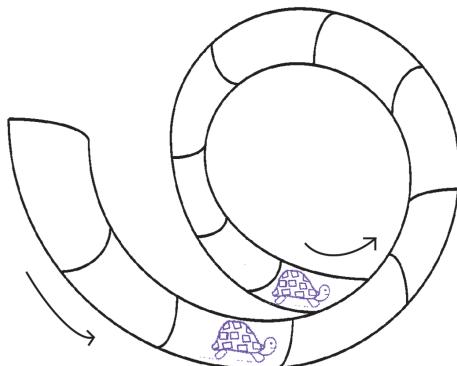
À partir de ce moment, les deux courront en rond, et le lièvre finira par doubler la tortue, après que cette dernière aura fait X pas dans le cycle. Étant donné que le lièvre avait une avance de L modulo C pas et court deux fois plus vite, X doit vérifier la congruence suivante:

$$X = L + 2X \pmod{C}$$

dont la solution est $X = -L \pmod{C}$. Ceci veut dire que les deux se rencontrent à L pas devant l'entrée au cycle !



Au moment de la rencontre du lièvre et de la tortue, faisons partir une deuxième tortue depuis le début de la liste. L pas plus tard, cette tortue aura parcouru la partie linéaire, et l'autre viendra d'achever son premier cycle. Ainsi les deux *carapacés* se rencontreront à l'entrée du cycle, observation cruciale pour résoudre l'éénigme ! Il en suit que la longueur de la partie linéaire correspond à la distance parcourue par la deuxième tortue avant qu'elle ne rencontre la première.



Reste à déterminer la longueur du cycle. Pour ce faire, organisons encore une course entre tortue et lièvre depuis l'entrée du cycle. Comme vous l'aurez deviné, le lièvre doublera la tortue à ce même endroit deux tours plus tard. Ainsi la longueur du cycle correspond à la distance parcourue par la tortue dans cette course, information qui vous suffit pour résoudre notre petite énigme !

En résumé

- Faites partir ensemble le lièvre et la tortue 1.
- Au moment du doublement, faites partir la tortue 2 et démarrez le chrono.
- Au moment de la rencontre des deux *carapacés*, le chrono indique $L+1$. Remplacez une tortue par un lièvre.
- Au moment du doublement, le chrono indique $L+C+1$.
- La réponse au problème est $L+C$.

Si cette petite excursion au monde des casse-têtes a suscité votre intérêt, n'hésitez pas à partir à la découverte de la gamme entière des colles posées aux participants de l'Helvetic Coding Contest sur hc2.ch.

PolyProg, polyprog.epfl.ch, sera de retour avec de nouveaux défis algorithmiques dès la rentrée académique! ■



Découvrir la virtualisation avancée

Laurent.Kling@epfl.ch, EPFL -STI, coordinateur informatique à la Faculté des Sciences et Techniques de l'Ingénieur

How do modern virtualization tools allow magical operations?

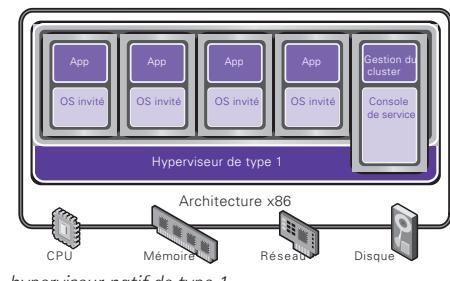
Comment la virtualisation permet des opérations magiques ?

Au premier degré, la virtualisation est un formidable outil de coexistence pacifique. Le Macintosh peut parler Linux et Windows; et réciproquement, Linux et Windows peuvent articuler le langage de l'autre. Au deuxième degré cette technique permet des opérations quasi magiques:

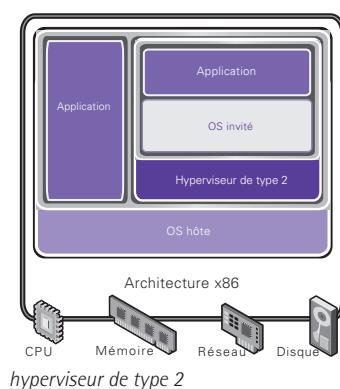
- dépasser la mort physique d'un ordinateur,
- transformer une configuration en mort-vivant,
- reprendre une expérience après six mois ou dix ans,
- déplacer une machine en fonctionnement entre serveurs.

Ces emplois semblent sortis d'un manuel vaudou ou de sorcellerie. Tel un alchimiste des temps modernes, il faut mettre en place les éléments du rituel:

- un hyperviseur natif de type 1,
- un serveur cobaye,
- un chef d'orchestre zombie.



Le premier ingrédient est alchimique, l'hyperviseur natif de type 1 est dénommé **bare metal** qu'on peut traduire par métal à nu. Dans notre utilisation, cela représente un logiciel de virtualisation connecté directement au matériel.



Le type 2 correspond au logiciel nécessitant un système d'exploitation sous-jacent (solutions décrites dans l'article **Virtualisation pour tous**).

Le deuxième élément de la potion est un serveur vitaminé. Dans ce type de service, c'est la place en mémoire qui fait défaut en premier, une fourchette entre 48 à 128 Go de mémoire vive avec correction d'erreur ECC est conseillée. Pour la puissance de calcul, deux processeurs avec 6 noyaux (*core*) chacun sont adéquats, car chaque machine virtuelle consomme une fraction de la capacité totale. Pour simplifier la configuration, la mémoire de stockage est conservée directement sur le serveur. Une capacité utile de 2 à 5 To est un bon point de départ. Les connexions réseau sont multiples pour séparer les trafics et offrir une bande passante cumulée suffisante (minimum 2 x 1 Gbit/s).

Le sorcier vaudou est primordial pour énoncer les recettes. Le paradoxe provient du fait que la partie matérielle est simple, mais ce serveur contient de nombreuses machines virtuelles. De ce fait, la complexité physique diminue pour être remplacée par plus de composants logiciels, métaphores du monde réel:

- un ordinateur devient une machine virtuelle;
- des prises réseau et leurs connectiques (switch) deviennent une entité virtuelle;
- le stockage devient abstrait.

Dans ce serveur réel, on se retrouve devant un ensemble de machines interconnectées nécessitant un *maestro*. La multitude de paramètres à gérer rend rapidement cette gestion surhumaine, il faut automatiser de nombreuses tâches. Au final, on obtient cette hiérarchie:

- l'être humain définit le morceau à jouer;
- le chef d'orchestre numérique vérifie la conformité des opérations et organise le déroulement harmonieux de chaque soliste en suivant la partition;
- chaque musicien exécute au moment adéquat les instruments pour produire la mélodie.

Le changement de rôle est enthousiasmant, il requiert un apprentissage pour éviter les fausses notes. Le décor étant posé, il est temps de mettre en œuvre ces nouvelles connaissances.

En pratique

Parmi les licences éducation disponibles, le choix s'est porté sur l'infrastructure de VMware. Voici les étapes à suivre pour la création de l'infrastructure

- le chef d'orchestre, *vCenter*,
- le soliste multi-instrumentiste, *vSphere* qui contient les trois composants primordiaux;
 - ▶ la capacité de calcul, *vCompute*.
 - ▶ le stockage des machines virtuelles, *vStorage*.
 - ▶ la gestion des réseaux virtuels et réels, *vNetwork*

Détaillons rapidement chaque étape.

Découvrir la virtualisation avancée

Le chef d'orchestre, vCenter

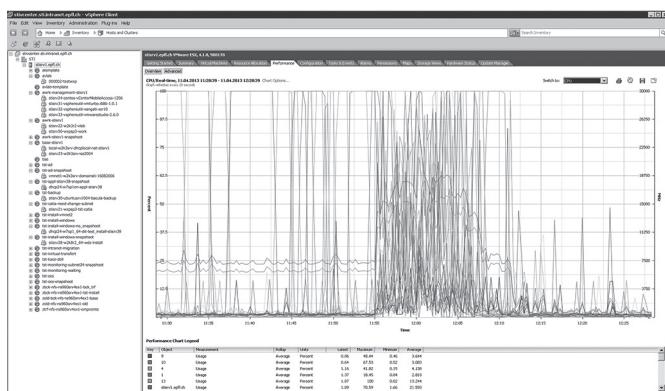
La solution la plus simple est d'utiliser un chef d'orchestre déjà existant. Heureusement, ce service est disponible centralement à l'EPFL avec myvm.epfl.ch. Dans ce cas, on évite sa configuration et sa maintenance. Pour les plus aventureux, ils peuvent le recréer en virtuel. Il est trop risqué de mélanger l'œuf et la poule en voulant utiliser le biotope en train d'être élaboré. Le plus raisonnable est de l'héberger dans un autre environnement virtuel ! Le chef d'orchestre est isolé du test, tout en étant capable de tenir son rôle. J'ai successivement utilisé les deux méthodes :

- incorporer dans le monde virtuel de l'EPFL,
 - créer dans une machine exécutée dans l'EPFL.
- La solution encapsulée permet de mener toutes les expériences imaginables, sans affecter l'environnement de production de l'EPFL et sans le risque de tout perdre par une fausse manœuvre.

Le soliste multi-instrumentiste, vSphere

La deuxième étape consiste à mettre en place le serveur de virtualisation vSphere. C'est probablement la partie la plus simple du processus, on peut même acheter son serveur avec vSphere préinstallée. Il faut vérifier la comptabilité du serveur avec ce produit (www.vmware.com/resources/compatibility/search.php).

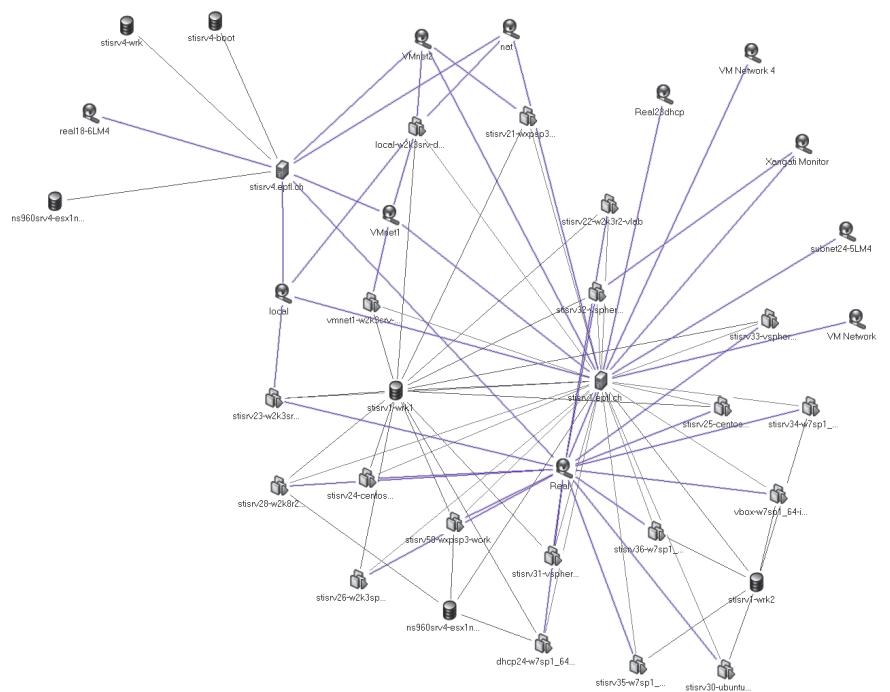
Un bricolage est déconseillé, car cette machine va héberger de nombreuses machines virtuelles. Pour éviter les problèmes de limitation de la taille mémoire avec la licence en version 5, j'ai utilisé la version 4.1. Depuis septembre 2012 ces limitations ont été supprimées et la version 5.1 doit être préférée.



mélodie sur un serveur virtuel vSphere

La troisième étape consiste à utiliser l'environnement ainsi créé

La différence fondamentale avec le modèle classique réside dans le fait qu'on manipule des composants logiques. *Le temps est formidablement accéléré*. Au processus traditionnel de définition du besoin, du choix de la configuration adaptée, de la commande d'un serveur, de l'attente de la livraison, de la mise en service et finalement de l'utilisation se substitue un processus très rapide : *créer un serveur virtuel et l'utiliser !* Immédiatement, les experts vont rétorquer que les étapes de définition du besoin et du choix



relation entre composants actifs visibles dans le chef d'orchestre

de la configuration adaptée sont toujours nécessaires. Je suis le premier à demander ces réflexions. La différence fondamentale est qu'une machine virtuelle est très facilement modifiable :

- si le système d'exploitation supporte la modification à chaud on peut augmenter la mémoire vive ou étendre la taille du disque dur sans éteindre la machine;
- allouer uniquement l'espace utilisé de disque dur (*thin provisioning*);
- si l'OS le permet, on peut changer le nombre de processeurs, néanmoins avec un redémarrage de la machine !

Au second degré, on arrive même à faire des opérations quasi magiques :

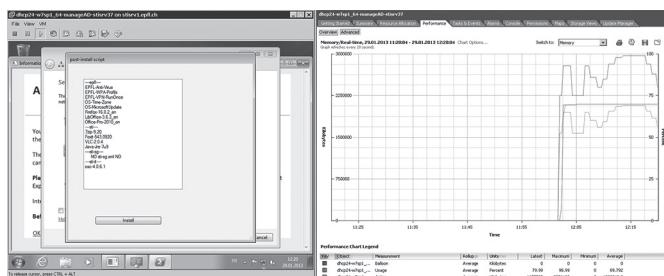
- sauvegarder l'état complet d'une machine en fonctionnement, avec mémoire vive et processus actifs;
- garder un historique des états successifs;
- revenir dans le passé avec une machine ressuscitée;
- sans interruption, déplacer une machine en fonctionnement entre différents serveurs physiques !

Ces exploits semblent être une fuite en avant technologique. Un serveur physique est concret, rassurant, identifiable. Un serveur virtuel possède exactement les mêmes qualificatifs. La seule différence réside dans le niveau de métaphore. Pour mieux exprimer les avantages, je vais prendre une série de cas concrets où la virtualisation représente une avancée.

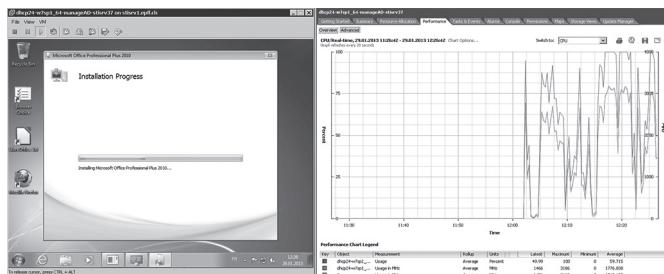
Mesurer les performances

Pour obtenir la meilleure adéquation entre le logiciel et la puissance d'un serveur, on procède par tâtonnements. Quand le serveur est physique, le choix est rapidement fait, il n'est pas possible d'adapter de manière très fine le contenant. Avec une machine virtuelle, la situation est totalement différente. Il est facile d'adapter chacun de ses composants, même après sa mise en service. Cerise sur le gâteau, on peut même visualiser en temps réel le fonctionnement du serveur et sa consommation. Par exemple l'installation automatique d'un poste Windows 7.

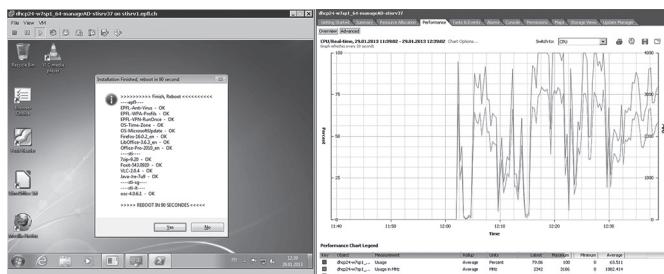
Découvrir la virtualisation avancée



consommation de la mémoire après l'installation de l'OS (7 min 30 s)



utilisation du processeur pendant l'installation de Microsoft Office (20 minutes)



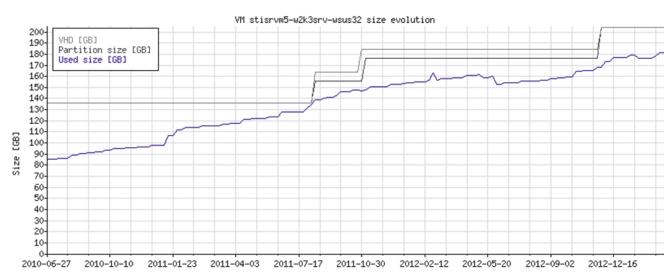
utilisation du processeur après l'installation complexe de la machine (30 minutes)



utilisation du processeur, la machine est disponible pour l'utilisateur (32 minutes)

La machine immortelle

Chaque machine physique possède une durée de vie. Parfois l'obsolescence technologique où la consommation énergétique entraîne l'arrêt d'une machine qui pourrait encore fonctionner. Chaque changement de configuration physique entraîne un coût humain. On se retrouve même parfois avec quatre générations de serveurs dans le même local où chacun possède un élément essentiel du suivant. Cette situation ubuesque a comme origine l'impossibilité de transférer un logiciel entre des machines physiques !



augmentation de la capacité disque d'une machine virtuelle pour les mises à jour Windows

La virtualisation résout ce problème de manière très simple. La machine essentielle n'est jamais détruite, elle se déplace entre des containers physiques. On peut même en lisant l'article [Virtualisation pour tous](#) (Fl 2/2013 - flashinformatique.epfl.ch/spip.php?article2644) la déplacer entre des systèmes concurrents.

Avec ces mécanismes, j'utilise une machine virtuelle de contrôle Windows XP qui a voyagé entre différents environnements physiques et virtuels :

- créée sur un poste de travail PC,
- déplacée sur un poste de travail Macintosh,
- transférée dans quatre générations de serveurs virtuels de type 1 (2.1, 3, 3.5 et 4.1).

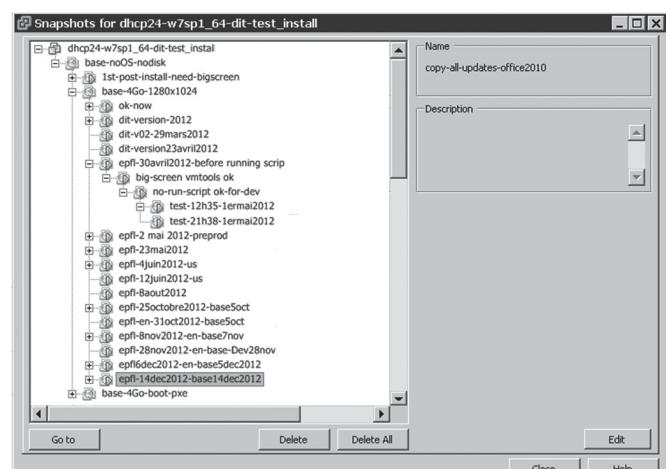
Dans le même ordre d'idée, une machine pour les mises à jour de Windows peut voir sa capacité disque augmenter au fil du temps (90 Go au départ, plus de 200 Go maintenant).

Avec une solution réelle, on aurait probablement dû changer de serveur physique.

La machine arrêtée à chaud

L'année dernière, j'ai dû procéder à une opération à cœur ouvert. Une nouvelle version d'un logiciel de CAO ne fonctionnait plus. Sur une machine virtuelle, j'ai reproduit l'installation de la nouvelle version. Effectivement, le serveur de licence était incompatible. J'ai figé cette machine virtuelle à chaud. Cela signifie que cette machine ne consomme plus de ressources mémoire vive ou de processeur. Elle était en état de mort-vivant. Muni de l'information, j'ai demandé au fournisseur de me donner une licence à jour. J'ai ensuite ressuscité la machine virtuelle et miracle, le logiciel a fonctionné correctement. Cette expérience semble anecdotique. Dans un autre contexte, combien de doctorants doivent recommencer à zéro une expérience en réinterprétant des notes ? Si l'expérience était conservée sur une machine virtuelle gelée, un clic serait suffisant pour redémarrer l'expérience.

Un arbre de développement conservé



arbre de développement avec les instantanés

La pratique d'un outil de gestion de versions est probablement la première étape d'une logique de développement rationnel de logiciel. Quand le travail consiste à mettre en place des systèmes complexes reliant plusieurs serveurs variant dans le temps, le problème semble irrésolu. Comment conserver les étapes successives de développement ? Avec un environnement virtuel, c'est possible en utilisant des instantanés. Il faut faire attention à la stabilité

Découvrir la virtualisation avancée

de la machine virtuelle et la taille occupée sur le stockage. Si on revient dans le passé, c'est peut-être facile pour une Pénélope de tout perdre, mais décourageant pour un développeur. La parade est de conserver les modifications dans un serveur externe. En cas de retour en arrière, on retrouve en effet l'état antérieur, mais rien n'empêche de recopier depuis ce serveur externe la version actuelle. C'est la méthode utilisée pour le développement de l'outil d'installation automatique pour Windows 7 qui doit se dérouler avant le dernier redémarrage. Ce moment très précis du cycle d'installation d'un système d'exploitation n'apparaît qu'une seule fois. Si on gèle ce moment, il est possible de redémarrer l'exécution exactement au même instant. Chaque *come-back* prend moins d'une minute. La prime de cette méthode est de disposer de l'état exact de chaque version dans un état parfaitement reproduit. Si une erreur apparaît en production, il est facile de la reproduire et de l'isoler pour résoudre le problème.

Déplacer une machine virtuelle à chaud

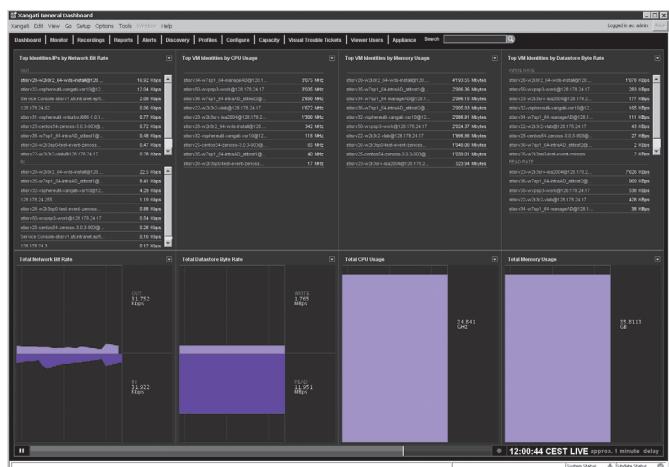
L'accumulation de machines virtuelles sur un seul serveur physique entraîne un risque élevé.

- La première précaution est de conserver une sauvegarde des machines, cette étape sera décrite dans un prochain article.
- La deuxième solution serait de dédoubler l'équipement, ce qui entraînerait l'achat d'un serveur de stockage pour un accès simultané aux données.

Dans un petit environnement de deux ou trois serveurs, la solution pour éviter cet achat est *vSphere Storage Appliance*. On utilise une partie de l'espace disque interne comme un miroir de l'autre machine. La perte de capacité brute qui en découle n'est pas négligeable: 50 % pour deux serveurs, 33 % pour trois serveurs. Cette méthode permet quand même d'éviter un stockage externe en assurant une haute disponibilité.

Finalement, avec la version 5.1 de VMware, il existe l'alternative de déplacer simultanément la machine virtuelle accompagnée de ses données entre des serveurs disjoints. Cette solution présente cependant l'inconvénient de ne pas pouvoir assurer une haute disponibilité.

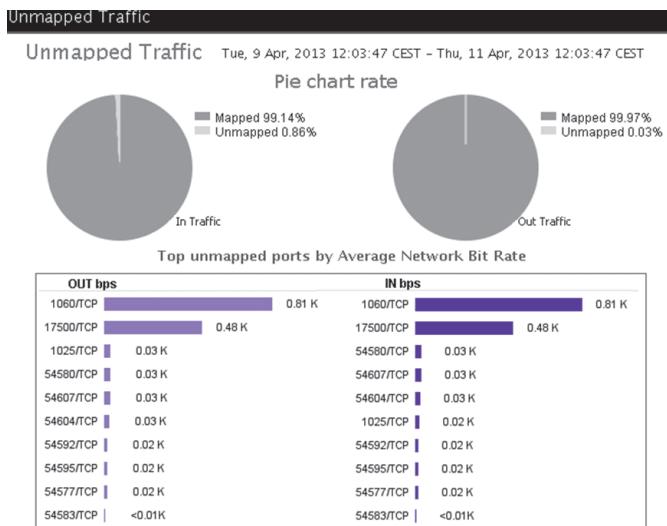
Mesurer un serveur réel avec une machine virtuelle à l'intérieur



Xangati, tableau de bord d'un hyperviseur de type 1

Comme tout est virtuel, on peut imaginer des combinaisons insolites: un serveur réel *vSphere*; exécutant une application encap-

sulée dans une machine virtuelle mesurant sa performance, nommée *Xangati*. Pour complexifier le tout, cette machine mesure le trafic réseau du serveur vers l'extérieur.



application Xangati, mesure du trafic réseau non lié aux machines virtuelles

Conclusion

Les machines virtuelles représentent sur son poste de travail une avancée importante. On se libère du carcan d'un seul système d'exploitation. Avec les environnements virtuels avancés du type hyperviseur 1, on ouvre la boîte de pandore et l'on se retrouve avec uniquement l'espérance. Notre soif de perfection nous pousse à espérer un système capable de satisfaire tous nos besoins. Sans atteindre cet objectif, ces outils nous offrent de nouveaux horizons. La capacité de traiter des serveurs comme des boîtes noires est très intéressante. Le fait de déplacer une machine virtuelle entre des environnements physiques en production représente une simplification importante du travail. La chute vertigineuse de la vente des PC (www.idc.com/getdoc.jsp?containerId=prUS24065413) semble indiquer la fin d'une époque, on se retrouve devant un formidable champ d'expérimentation.

Vendeur	1er trimestre 2013		1er trimestre 2012		Progression sur une année
	livraisons	Parts de marché	livraisons	Parts de marché	
HP	3'570'000	25.1%	4'632'000	28.5%	-22.9%
Dell	3'074'000	21.7%	3'590'000	22.1%	-14.4%
Apple	1'418'000	10.0%	1'533'000	9.4%	-7.5%
Toshiba	1'279'000	9.0%	1'349'000	8.3%	-5.2%
Lenovo	1'274'000	9.0%	1'127'000	6.9%	13.0%
Others	3'581'000	25.2%	4'022'000	24.7%	-11.0%
Total	14'197'000	100.0%	16'255'000	100.0%	-12.7%

IDC: ventes des cinq plus gros fournisseurs de PC au USA, premier trimestre 2013 (rapport préliminaire)

Depuis plus de dix ans, mon ordinateur de travail est le réceptacle de fenêtres provenant d'autres serveurs, la majorité de ceux-ci sont virtualisés. Sur une tablette, la même situation se répète. Finalement cette distance avec le matériel, rend peut être plus évident notre dépendance à ces étranges lucarnes. ■



Déploiement continu, cas du moteur de recherche de l'EPFL

Maciej.Macowicz@epfl.ch, EPFL - Domaine IT- KIS, responsable de search.epfl.ch

We introduce here some principles of continuous delivery and we present their application to the development process of the EPFL search engine.

Voici quelques principes de déploiement continu et leur application aux processus de développement du moteur de recherche de l'EPFL.

Cycle de vie d'une application

Le cycle de vie d'une application informatique commence par la phase de *définition et analyse des besoins*, suivi de la *conception*; dans la phase de *codage* on élabore les *codes sources* de l'application. Les *tests* permettent de trouver des erreurs techniques ou fonctionnelles; ensuite, lors du *déploiement* on installe l'application sur les serveurs de production. Finalement la *maintenance* consiste à corriger des erreurs ou à ajouter des nouvelles fonctionnalités selon les besoins identifiés; les concepteurs de l'application repassent alors par les phases de conception, de codage, refont les tests, et finalement déplacent en production. Les principaux problèmes liés au cycle de vie d'une application sont:

- les différentes activités (construction de l'application, tests, mise en production, etc.) souvent effectuées *à la main*; nécessitant un savoir-faire particulier et pouvant conduire à des erreurs,
- les environnements de test différents de ceux de production; par conséquent certaines erreurs se manifesteront en production sans avoir pu être détectées pendant les tests,
- la mise en production manuelle et fastidieuse et demande beaucoup d'efforts. Elle intervient donc rarement, privant les utilisateurs finaux des corrections ou améliorations durant une longue période,
- l'installation manuelle de nouveaux serveurs pour l'application présente un risque accru, car elle est fastidieuse et demande un savoir-faire très particulier.

Déploiement continu

Les pratiques de *déploiement continu* [1] sont apparues pour pallier ces problèmes, minimiser la distance entre les phases et permettre les mises en production de la manière la plus rapide et la plus efficace possible tout en minimisant les risques. Le terme **continu** signifie qu'une nouvelle version de l'application peut être déployée à tout moment, le terme **déploiement** se réfère à la mise en production d'une nouvelle version de l'application à partir des codes sources, mais aussi à l'installation d'un nouveau serveur pour l'application à partir des configurations nécessaires.

Le déploiement continu s'appuie sur différents principes:

- les tâches répétitives (construction des exécutables, reconfiguration des serveurs, mise en production, tests, ...) sont automatisées et effectuées par des *scripts*,
- les données utilisées pour la construction (codes sources, scripts) et le fonctionnement de l'application (configurations des serveurs, des logiciels tiers, de l'OS) sont stockés dans un *dépôt* permettant la *gestion de versions*,
- l'application est accompagnée d'une *suite de tests automatisés*, qui détectent les erreurs techniques et fonctionnelles,
- les serveurs de production sont accompagnés de serveurs de test de configuration identique,
- le déploiement sur les serveurs de test et de production se fait exclusivement par les scripts, les opérations *manuelles* sont limitées au strict minimum,
- la création d'un nouveau serveur de test ou de production est entièrement automatisée, toutes les informations nécessaires sont disponibles dans le dépôt.

Du point de vue du développeur le déploiement continu n'a rien de révolutionnaire, c'est juste un recueil de bonnes pratiques basées sur des règles de bon sens. Nous allons présenter l'application de ces principes au moteur de recherche de l'EPFL, **search 2010**.

Environnement de développement de search 2010

Search 2010 est une application mixte J2EE/Groovy, déployée sur deux serveurs physiques de production, placés derrière le répartiteur de charge. L'environnement de développement de search2010 est construit conformément aux recettes de déploiement continu et comporte: un **poste de développement**, les **serveurs de construction et de test** et les **serveurs de production**. La plupart des tâches (construction/test/déploiement) sont automatisées par des scripts; un dépôt est utilisé pour les codes sources, les scripts et les configurations des serveurs.

Le **poste de développement** comprend les exécutifs Java et Groovy, un environnement de développement pour élaborer les codes sources, les outils de construction nécessaires (maven et gradle), un gestionnaire de versions (svn) pour accéder au dépôt, une version allégée de serveur d'application (tomcat) pour pouvoir exécuter l'application sans avoir à la déployer, et l'environnement Geb/Selenium [2, 3] pour effectuer des tests fonctionnels (par simulation du comportement de l'utilisateur face à un navigateur Web). On peut enregistrer les modifications des codes sources dans le dépôt uniquement depuis le poste de développement.

Le **serveur de construction** est utilisé pour construire l'application à partir de codes sources rapatriés depuis le dépôt et pour la déployer (automatiquement, par script) sur les **serveurs de test**

Déploiement continu, cas du moteur de recherche de l'EPFL

et/ou de production. Il comporte un gestionnaire de versions, un outil de construction et les scripts de construction/déploiement, qui peuvent être modifiés sur ce serveur et stockés dans le dépôt.

Les **serveurs de test** sont utilisés pour tester l'application dans un environnement identique à la production. Ils comportent un gestionnaire de versions et un serveur d'application (apache et tomcat).

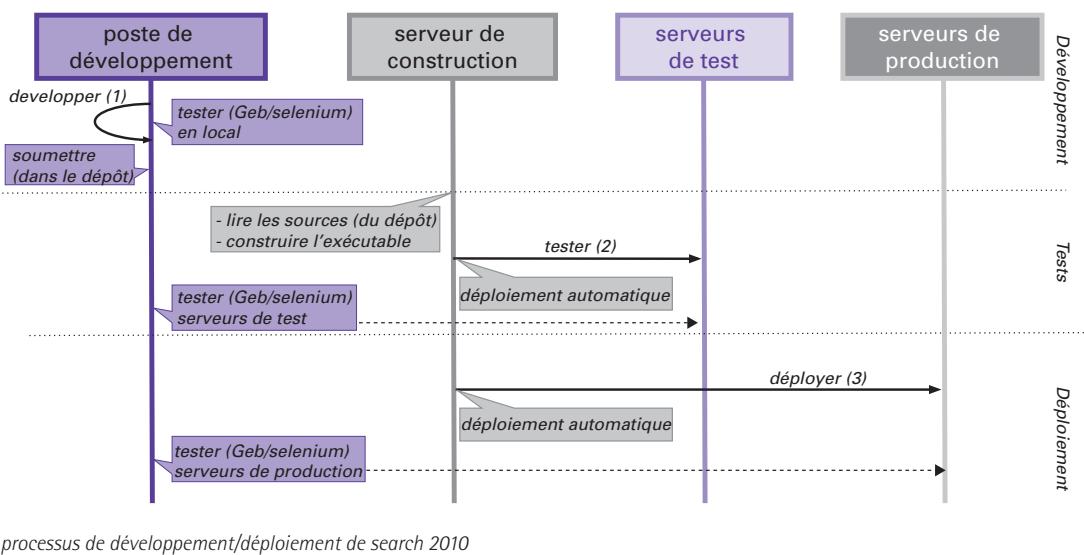
L'application est déployée sur les serveurs de test depuis le serveur de construction, les tests fonctionnels se font alors depuis le poste de développement. Les serveurs de test peuvent également servir pour démonstration d'une nouvelle fonctionnalité de l'application dont le développement n'est pas terminé (depuis une branche particulière du dépôt). Toute modification des configurations du serveur d'application (apache et tomcat) ne peut se faire que sur les serveurs de tests et est stockée dans le dépôt.

Les **serveurs de production** comportent un gestionnaire de versions et un serveur d'application; aucune modification de l'application ni des configurations n'est possible. L'application est déployée depuis le serveur de construction, les configurations sont rapatriées depuis le dépôt.

Il y a trois phases principales du développement (voir schéma):

1. Le **développement** proprement dit se passe sur le poste de développement, le développeur a la possibilité de modifier les codes sources, construire et exécuter l'application en local et lancer une suite de tests en local. Une fois un **jalon de développement** (correction d'une erreur ou mise en place d'une fonctionnalité) atteint, on envoie les sources dans le dépôt et on passe à la phase de tests.
2. Les **tests** ont lieu entre le poste de développement et les serveurs de construction et de test. Durant cette phase on rapatrie les sources depuis le dépôt sur le serveur de construction, on construit l'application et on la déploie sur les serveurs de test. Ensuite on lance les tests automatisés des serveurs de test depuis le poste de développement. On repasse en développement si les tests échouent, sinon on passe au déploiement. Dans cette phase on a la possibilité de modifier les configurations du serveur d'application.
3. Le **déploiement** a lieu entre les serveurs de construction et de production; cette phase consiste à déployer l'application construite et testée dans la phase précédente sur les serveurs de production. Il est également possible de lancer les tests fonctionnels depuis le poste de développement.

L'installation d'un nouveau serveur de construction/test/production (*smoke-test*) est entièrement automatique, on commence par installer le système d'exploitation, on le configure suivant les recettes gérées par le système Puppet [4] et stockées dans le dépôt. Ensuite on rapatrie depuis le dépôt le script d'installation



de search.epfl.ch, qui à son tour transfère les configurations du serveur d'application et l'exécutable de l'application. Le nouveau serveur est alors prêt à l'emploi.

L'installation d'un nouveau poste de développement est plus fastidieuse et nécessite le rapatriement manuel de la plupart des logiciels utilisés; bien documentée, elle peut cependant se faire en moins de deux heures de travail.

Conclusion

Le processus de développement de search 2010 présente des avantages indéniables:

- disponibilité de modifications en production juste après la fin de leur élaboration,
- possibilité de démonstration des fonctionnalités en cours de développement (depuis une branche du dépôt) sans effort (déploiement sur les serveurs de test),
- tests effectués dans un environnement de test identique à la production, donc pertinents,
- répétabilité et prédictibilité de processus de développement/déploiement,
- possibilité de reconstruction rapide de serveurs.

Vu la taille du projet search 2010 (un seul développeur), nous n'avons pas jugé utile d'installer un serveur d'intégration (par exemple Jenkins [5]) pour automatiser les phases de développement.

Références

- [1] HUMBLE, Jez and FARLEY David. *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Addison-Wesley Signature Series (Fowler), 2011.
- [2] Geb: very groovy browser automation. www.gebish.org.
- [3] Selenium Browser Automation. docs.seleniumhq.org.
- [4] fr.wikipedia.org/wiki/Puppet.
- [5] Jenkins: An extendable open source continuous integration server. jenkins-ci.org. ■