Préparation de la roadmap wide

Ce document est une prospective des fonctionnalités de Wide. Nous avons déjà évoqué la plupart d'entre elles. Pour chacune d'elle je propose un temps de développement approximatif. Dans la réalité, ce sera probablement plus long puisque nous devons assurer d'autres tâches en parallèle (correction de bugs, gestion de projet etc.).
Nous pensons qu'il serait mieux de faire des releases (nouvelles versions) fréquemment. Les études montrent que ça limite les risques. Nous pensons que des cycles de 3 mois sont intéressants, suffisamment longues pour produire quelques nouveautés et suffisamment courtes pour limiter la casse si on a pris une mauvaise direction.
Il faudra donc exprimer des priorités par rapport aux éléments présentés ci-dessous.

I. Exigences fonctionnelles

A. Application web

En ligne :

http://outils.mri.cnrs.fr/

Se logguer puis « File Manager », en bas.

1. Dé-convolution

Le système actuel de déconvolution est accessible depuis le site web de MRI. Pour l'utiliser il est nécessaire de déposer ses images sur le serveur THOT puis de se rendre sur l'interface web. La demande est alors placé dans une queue de travail et sera effectuée dès que les calculateurs sont disponibles.

Cet outil peut être ajouté à Cicero et deviendrait accessible pour image depuis un menu de l'interface Web. Selon Volker, c'est un travail assez long, entre 3 et 6 mois et dans lequel il devra s'impliquer.

Cette intégration permettrait « d'obliger » les utilisateurs à se saisir de l'outil Cicero et revêt donc une importance stratégique.

2. gestion des méta-données (lecture, écriture)

La plupart des images acquises sur les microscopes sont accompagnées de méta-données décrivant le contexte d'acquisition (matériels, réglages etc.). Ces méta-données évoluent dans le temps en même temps que les formats d'image. Ce travail constitue donc une tâche de fond qu'il conviendra de poursuivre.

Un certain nombre de méta-données sont nécessaires à l'exécution de la déconvolution. Un travail dans ce sens devra être réalisé avant le travail sur la déconvolution (quelques semaines).

3. Upload

Pour le moment l'ajout de fichier se fait par l'intermédiaire du client lourd. Il serait souhaitable de fournir cette fonctionnalité depuis le navigateur Web, avec comme contrainte une limitation de la taille (par exemple 100Mo).

Une alternative consiste à créer une « applet », une application Java s'exécutant depuis le navigateur et permettant les mêmes facilités que pour le client lourd. Une autre alternative consiste à utiliser Flash.

Le développement de cette fonction pourrait faire l'objet d'un stage d'informatique.

Version web (taille limité) : ~1 semaine

Version applet : ~ 2 semaines (ou 3 mois de stage)

4. Prise en charge des fichiers nd et scan (et autres formats regroupant plusieurs fichiers pour une image)

Certain format d'image dans la microscopie utilise plusieurs fichiers pour représenter une image. La version actuelle du client d'upload gère certains fichiers .nd, les plus simples. Cicero n'a donc pas été installé sur les postes d'acquisition reliés à des microscopes produisant des formats non supportés.

1 à 2 semaines

5. Fonction historique

La fonction « historique » prendra tout son sens lorsque il sera possible de réaliser des analyses d'images. Elle s'inscrit dans une démarche de traçabilité et donc de qualité. L'idée est de pouvoir revenir sur les différentes versions d'une image.

2 à 3 semaines

6. Fonction recherche (avec tags et / ou sans)

Le système actuelle permet d'associer un ou plusieurs tags (mot clé) à une image. La fonction recherche permettra de rechercher à partir des tags (une façon de filtrer ou de catégoriser les images) mais également sur d'autre champs (nom du fichier, du répertoire, méta-données etc.)

Le temps pour développer dépend de la complexité du moteur. En partant d'une base simple comme un champs texte, il faut compter 2 à 3 semaines.

C'est également une activité proposable à un stagiaire.

7. Autres opérations d'analyse d'image

C'est l'aspect le plus intéressant de Cicero mais aussi celui sur lequel il y a le plus d'inconnu. L'objectif est de pouvoir réaliser des analyses habituellement effectué avec d'autres logiciels avec Cicero, depuis l'interface Web. Dans le contexte du navigateur, certaines choses sont très compliqués à réaliser puisque nous ne disposons pas de la même inter-activité qu'avec un logiciel spécifique, cependant les choses ont évolué et continuent d'évoluer. Une réunion devra être organiser avec Volker pour essayer de mieux cerner l'envisageable.

8. Ergonomie et meilleur feedback utilisateur

Cette rubrique regroupe diverses petites améliorations :

- téléchargement des archives lourdes asynchrones (avec barre de progression), lorsqu 'on utilise la fonction Download->zip pour télécharger un dossier comprenant beaucoup de données, l'utilisateur se retrouve bloqué pour un temps indéterminé et long en attendant que le serveur est créé l'archive. On peut améliorer le feedback en proposant une barre de progression ~ quelques jours
- une vue sur l'interface web des upload en cours : lorsqu'une requête d'upload a été lancé depuis un poste d'acquisition par exemple, l'utilisateur ne peut plus en suivre l'évolution. On peut proposer un onglet sur l'interface web permettant ce suivi. ~ 1 à 2 semaines.

9. Viewer

Le viewer est une application complémentaire, qui n'existe pas pour l'instant dans Cicero et qui permettrait de visualiser les images un peu comme on le fait dans imageJ. Ce serait le support pour réaliser des opérations d'analyse d'image.

Les fonctions que nous y voyons dans un premier temps reprendraient celles du viewer d'OMERO (il pourrait d'ailleurs servir de base à notre travail), auquel on rajouterait rapidement un outil pour faire des sélections. Puis selon les besoins nous ajouterons d'autres fonctionalités.

~ entre 1 et 3 mois selon ce qu'on peut réutiliser d'OMERO

B. Client lourd

Le client lourd dispose d'une aide en ligne ici :

http://dev.mri.cnrs.fr/wiki/cicero/Upload_client

1. Conservation de la hiérarchie des répertoires

Actuellement lors de l'upload, l'utilisateur sélectionne un répertoire et l'ensemble des fichiers à partir de ce répertoire et de ses sous-répertoires seront uploadés et se retrouveront dans un seul répertoire sur le serveur.

~ 1 semaine

2. Esthétique – ergonomie

Un travail de fond sur l'ergonomie doit être effectué en tenant compte des remarques des utilisateurs.

3. Communication avec le Daemon

Pour ne pas saturer inutilement le serveur, les Daemon installés sur les postes se mettent en veille durant 5 mn s'il n'y a plus de fichiers à transférer. Ce mécanisme a pour effet que lors d'une requête d'upload de nouveaux fichiers, le téléchargement ne commence pas nécessairement immédiatement. Ce qui peut contrarier certains utilisateurs (ou faire croire à une panne).

On peut développer un système de communication entre le client et le daemon pour que le premier informe le second qu'un téléchargement doit commencer au plus tôt.

~ 1 semaine

4. Possibilité de l'utiliser hors du réseau MRI (logging à prévoir)

L'authentification sur le LDAP de MRI a lieu lors de l'ouverture de la session Windows. De ce fait le client d'upload n'assure pas l'authentification ce qui entraine que si on installe le système sur une machine hors du réseau administré par MRI, l'authentification n'aura pas lieu.

< 1 semaine

5. Possibilité de créer une arborescence de répertoire depuis le client lourd (via le bouton « set »)

Le bouton « set » du client permet de définir le répertoire sur le serveur où seront installés les fichiers. Il ne permet pas cependant de créer de répertoire. On ne peut le faire que sur l'interface web.

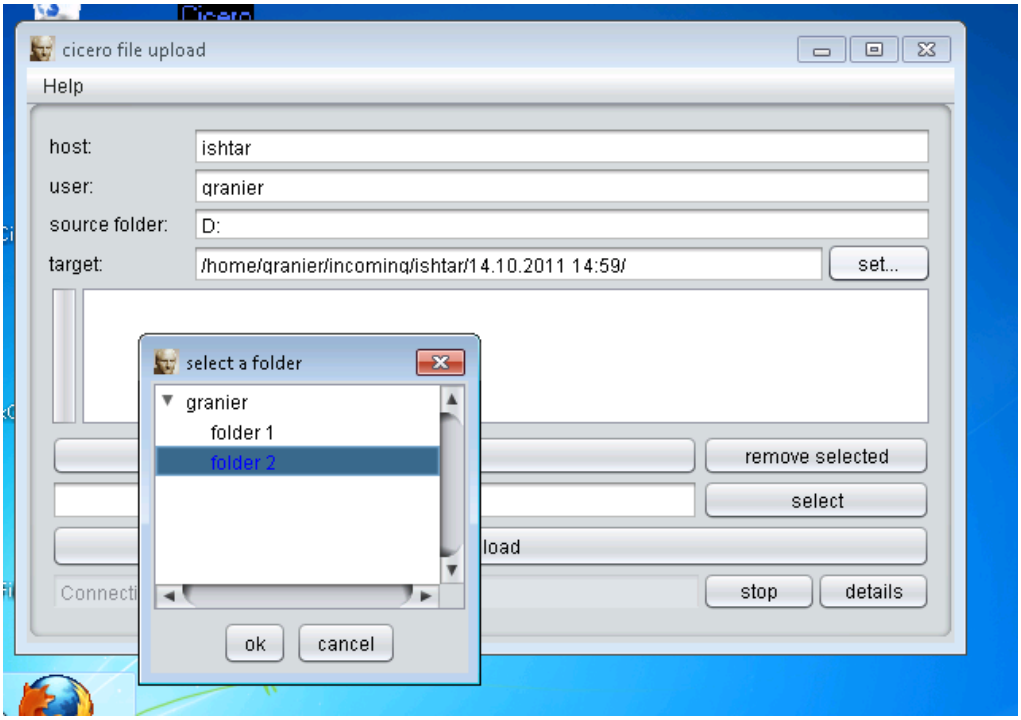

Illustration 1: Le fenêtre pour sélectionner un répertoire sur le serveur

~ 1 semaine (synergie avec le point 1)

6. Donner une estimation du temps d'upload (si par exemple il y a plusieurs utilisateurs sur l'hôte)

Proposer une information indiquant à l'utilisateur qui vient de soumettre des fichiers indiquant dans combien de temps ses téléchargements devraient commencer et combien de temps ils vont durer.

~ quelques jours

7. Divers petits changements très court

- Virginie : le répertoire par défaut `/login/incoming/nom de machine/date` ne convient pas (le nom de la machine et la date ne correspondra pas forcément à la date de manip)
- Indiquer le dossier en cours d'upload
- Virginie : supprimer la ligne d:\utilisateurs
- Scan de départ trop long : à l'ouverture du client, ce dernier parcours l'arborescence des utilisateurs et repère les nouveaux fichiers depuis la dernière session. Cela permet de les ajouter par défaut comme candidat à l'upload.

C. ImageJ

L'idée de ce point est de permettre d'ouvrir une image hébergé sur le serveur cicero depuis imageJ. Puis, après l'avoir modifié, de la renvoyer sur le serveur.

D. Daemon

Le Daemon est le service (qui tourne en permanence y compris quand personne n'utilise le poste) qui assure l'upload des fichiers vers le serveur.

II. Exigences non fonctionnelles

Les exigences non fonctionnelles regroupent des exigences de résilience (robustesse, résistance aux incidents), de performance, de sécurité et de supportabilité (capacité à être maintenue, configurée et étendu).

Ici je retiens la maintenance envers les framework que nous utilisons. L'enjeu est de ne pas se faire distancer par rapport à une technologie.

1. Migration Glassfish 3

Glassfish est le serveur d'application sur lequel « tourne » cicero.

A priori peu d'impact sur le projet ~ entre 2 jours et 2 semaines (des imprévus peuvent arriver). Cependant d'autres applications gérées par Volker utilisent Glassfish donc risques d'effet de bord.

2. Migration richfaces 4.x, JSF 2.x

Richfaces est la librairie permettant de faire une interface utilisateur « riche » sur le web. Nous utilisons la version 3 qui n'évoluera plus, il nous paraît très important de passer à la version 4 dont les développements sont assez dynamiques.

~ 1 à 3 semaines

3. Migration loci 4.3.2

Loci est la librairie de référence pour manipuler les formats de microscopie. Elle est utilisé par OMERO et développé par les mêmes équipes.

~ de 1 jour à une semaine (en général pas de problèmes)

III. Proposition de Roadmap

Ce n'est qu'une proposition.

A. Release fin janvier 2012

À titre indicatif, d'ici fin janvier, je dispose d'environ 60 jours. J'en réserve 10 pour les activités de support. Volker peut mobiliser 12 jours.

Fonctionnalités	Temps jours-hommes
(MetaDec) Gestion des méta-données nécessaires à la déconvolution I.A.2.	12
(Upload) Upload depuis le navigateur (version petits fichiers)I.A.3.	5
(MultiF)Prise en charge des fichiers .nd et .scan I.A.4.	8
(ErgWeb) Ergonomie interface web I.A.8.	10
(HierFold) Conservation de la hiérarchie des répertoires I.B.1.	5
(Login) Log-in hors MRI I.B.4.	7
(Mig) Migration glassfish 3 II.A.1.	6
(Deconv) Déconvolution I.A.1.	12

Fonctionnalités	Temps jours-hommes
Total	**65 j.h**

B. Release fin mai 2012

Environ 75 jours à répartir dont 10 en fonction de support.

On peut mobiliser 36 jours de Volker. Soit 65 + 36 = 101 jours disponibles.

Fonctionnalités	Temps jours-hommes
(Deconv) Déconvolution I.A.1.	60
(UplApp) Upload applet I.A.3.	12
(Hist) Fonction historique I.A.5.	12
(Mig) Migration richfaces 4 II.A.2.	7
(ErgCl)Esthétique / ergonomie I.B.2.	7
Total	**98 j.h**