Créer son Web-Service BioMOBY @ LIPM

http://lipm-bioinfo.toulouse.inra.fr/formations

Sébastien Carrere
sebastien.carrere@toulouse.inra.fr

Laboratoire Interactions Plantes Micro-organismes

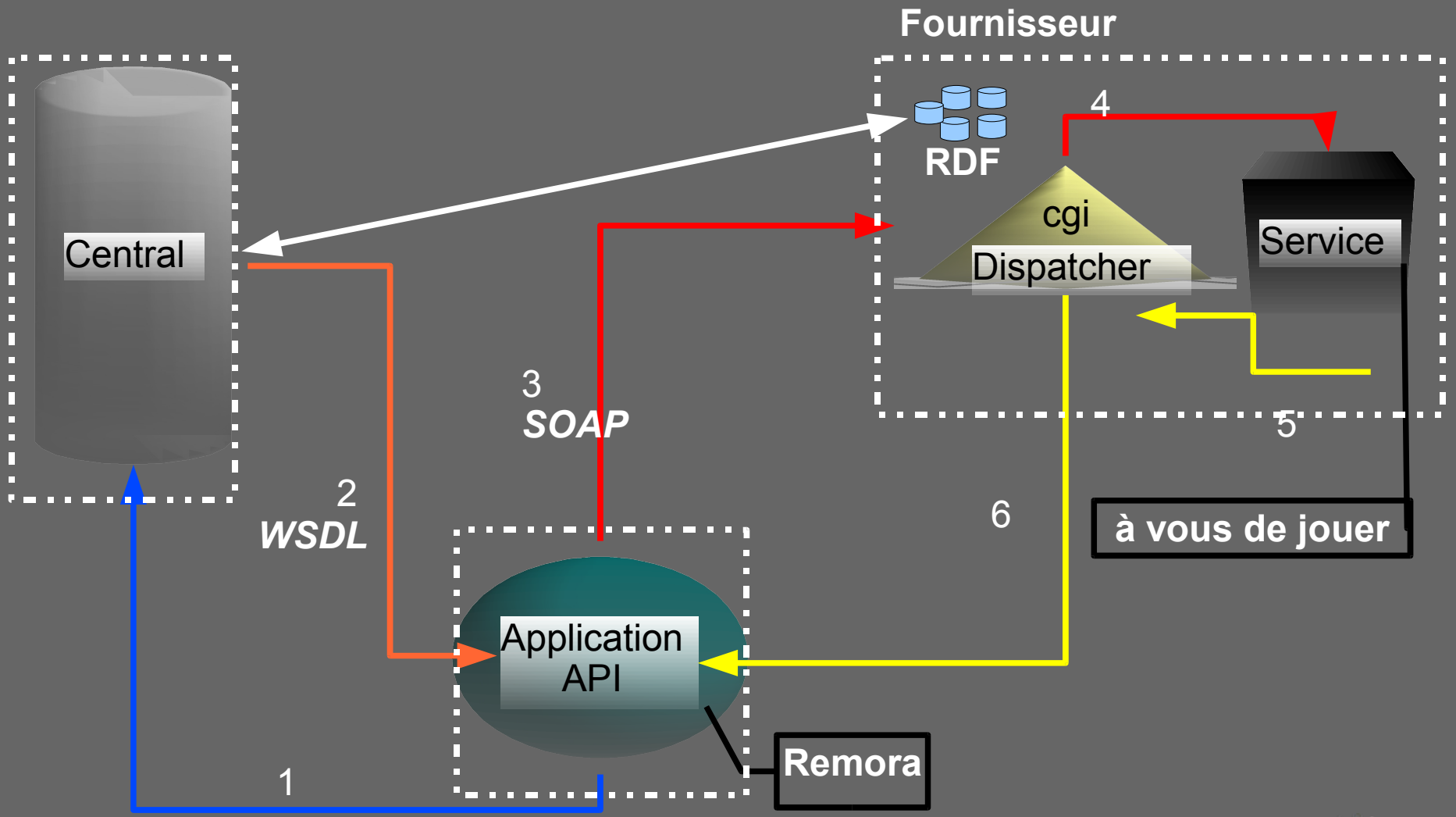
Plan

- **1) Fonctionnement de BioMoby**
  - le Central, le Dispatcher et le Web-service
  - les Articles Primaires et les Paramètres
  - les NameSpaces, le typage des Objets et services

- **2) Principe de fonctionnement d'un Web-Service BioMoby**
  - à quoi ressemble un message XML::BioMoby
  - comment est structuré un web-service BioMoby écrit en Perl

- **3) TP**
  - Installation de l'environnement : ***playmoby***
  - Description d'un Web-service : ***Appli.pm***
  - Enregistrement, test

le Central, le Dispatcher et le Web-service



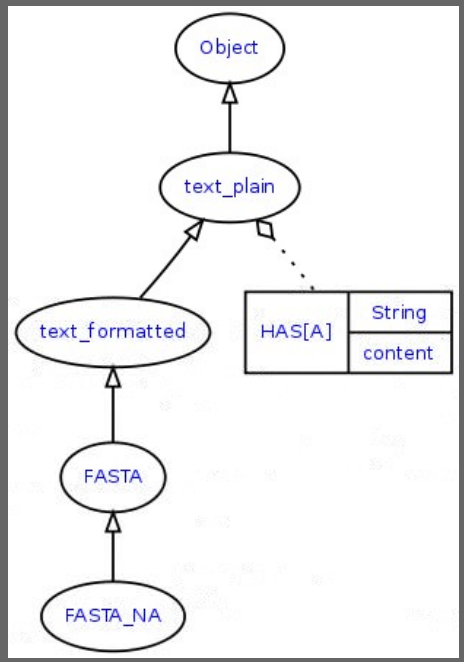
les Articles Primaires et Secondaires (Paramètres)

- **Les articles primaires**
  - Input / Output
  - dans le cas général: OBLIGATOIRES
  - 2 types: Simple objet / Collection d'objets (Homogène ou pas)

- **Les articles secondaires**
  - Paramètres
  - Optionnels
  - Types prédéfinis String, Integer, Float, DateTime
  - Attributs Enum, Min, Max, Default, Description

les NameSpaces, le typage des Objets et services

- **Un objet minimal peut être défini par un ID et un/des *NameSpaces***
  - exple: ID=P10958 NameSpace=SPTR_AC



- **Mais on peut vouloir passer autre chose que des *Objets***
  - typage des données (***ontologie***)
  - permet l'interopérabilité entre services (***workflow***)
  - NameSpace Aware

- **De même on peut typer les services (Parsing, Analyse, Retrieval,..)**

un message XML::BioMoby

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" SOAP- ...">
<SOAP-ENV:Body>
        <namesp3:Multalin xmlns:namesp3="http://biomoby.org/">
        <body>
        <![CDATA[
        <?xml version='1.0' encoding='UTF-8'?>
        <moby:MOBY xmlns:moby='http://www.biomoby.org/moby-s'>
                <moby:mobyContent>
                        <moby:mobyData queryID='1'>
                                <moby:Simple moby:articleName='mes_sequences'>
<moby:FASTA_AA_multi><moby:String articleName='content'><![CDATA[>SMc02591_AA-Y02591
MSVPASSRERKSYWISLVSLLAAVPLAVLVGSRGEFAAWLQRRMEPPLTV
VVELFLVPRQADGFTLSLALTGSPPILLK
>SMc04141_AA-gst9
LSLAIFPVLVLYVIFSRQLIRGITAGAVK]]></moby:String>
</moby:FASTA_AA_multi>
                                </moby:Simple>

                                <moby:Parameter moby:articleName='gapcost'>
                                        <Value>5</Value>
                                </moby:Parameter>

                        </moby:mobyData>
                </moby:mobyContent>
        </moby:MOBY>
        ]]>
        </body>
        </namesp3:Multalin>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

un web service BioMoby écrit en Perl

```
sub MonWebService
{
        my ($caller, $message) = @_;
        #Recuperation de la liste des requetes dans le message

        foreach my $query (@a_queries)
        {
                #recuperation du numero de la requete
                #recuperation des articles

                foreach my $input_article (@a_input_articles)
                {
                        my ($article_name, $article) = @{$input_article};

                        # Recuperation des input
                        # Recuperation des parametres
                }

                #Ecriture des fichiers temporaires de données
                #Execution du traitement

                #Ajout du resultat au message de reponse du WebService
        }
        #Retour du message
}
```

Financé par ANR

Sébastien Letort & Sébastien Carrere

Lipm

Un environnement de développement et de production de web-services bioMOBY

moby

Notre conception d'un web-service

Un web-service est l'encapsulation d'un programme déjà existant.

Ce programme manipule des fichiers en entrée et sortie (STDIN & STDOUT)

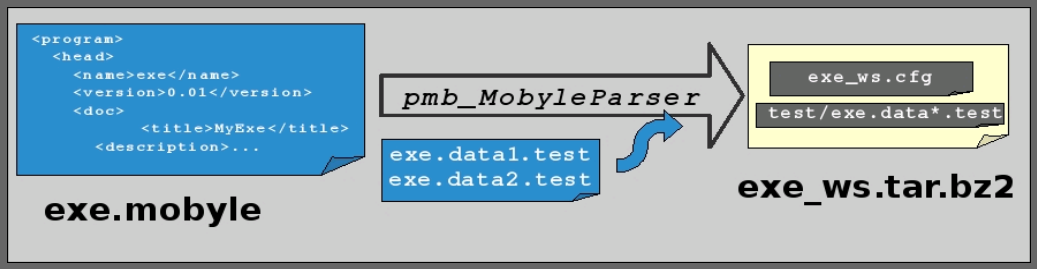
- on peut toujours utiliser ces programmes en ligne de commande
- on peut les encapsuler via d'autres technologies (CGI, Mobyle)

3 étapes

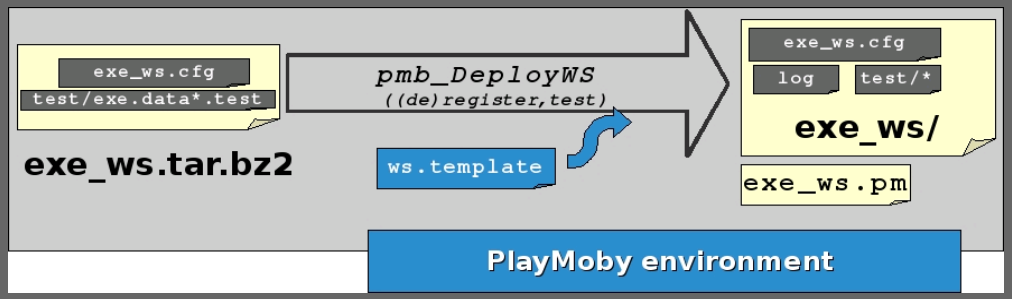
1. Génération d'un fichier de description [Mobyle XML]

- Mobyle: C.Letondal *et al* , successeur de Pise;
  grosse collection de descriptions d'applications
- Appli.pm: un module pour générer ces fichiers XML

2. Génération du web-service à partir de la description XML



3. Enregistrement et test



« en avant les histoires! »

1 . PREPARATIFS

#connexion à la machine de développement (machine avec serveur apache)
#l'arborescence biomoby doit etre sous une arborescence apache

% ssh stageX@lipm-bioinfo.toulouse.inra.fr
% pwd

#recuperation de l'archive playmoby
% wget http://lipm-bioinfo.toulouse.inra.fr/biomoby/playmoby.tar.bz2
% tar xjf playmoby.tar.bz2
% cd playmoby

RDV sur http://lipm-bioinfo.toulouse.inra.fr/biomoby/playmoby#first_install
% ./pmb_configure.pl [...]
% source .setenv

« en avant les histoires! »

2 . ENCAPSULATION

#ecrire un script perl encapsulant votre programme (testé sur lix3)
#ce script doit permettre d'utiliser votre programme avec des E/S fichiers
#decrivez votre programme a l'aide du module Appli.pm et de l'ontologie BioMOBY

#pour le typage des E/S
#htttp://lipm-bioinfo.toulouse.inra.fr/registry/cgi/registry.cgi

#exemples de scripts:

sample/PrintHello
sample/tab2fas
sample/msf2phylip
sample/msf2fasta
sample/compare_list
sample/blastp_swissprot

« en avant les histoires! »

3 . GENERATION XML / WS

générez le fichier Mobyle-XML
générez l'archive du WS (***$PMBBIN/pmb_MobyleParser.pl***)
    - preparez au prealable un jeu de test qui vous sera demandé
décompressez l 'arborescence

4 . DEPLOIEMENT

enregistrez le service dans l'annuaire de test (**opencentral**) (***$PMBBIN/pmb_Deploy.pl***)
    >> creation du fichier .pm et du fichier RDF
testez le service (***$PMBBIN/pmb_Deploy.pl***)
    >> ecriture dans le fichier log
désenregistrez le service de l'annuaire de test (***$PMBBIN/pmb_Deploy.pl***)
    >> vidange du fichier RDF
enregistrez le sur l'annuaire de production (**mobycentral**) (***$PMBBIN/pmb_Deploy.pl***)
visualisez dans Remora (cache mis a jour toutes les heures xh30)