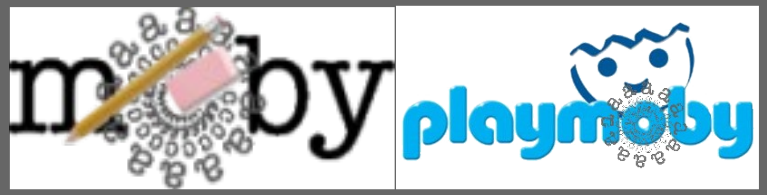
Créer son Web-Service BioMOBY @ LIPM



http://symbiose.toulouse.inra.fr/tp_playmoby/

Sébastien Carrere
sebastien.carrere@toulouse.inra.fr

Laboratoire Interactions Plantes Micro-organismes

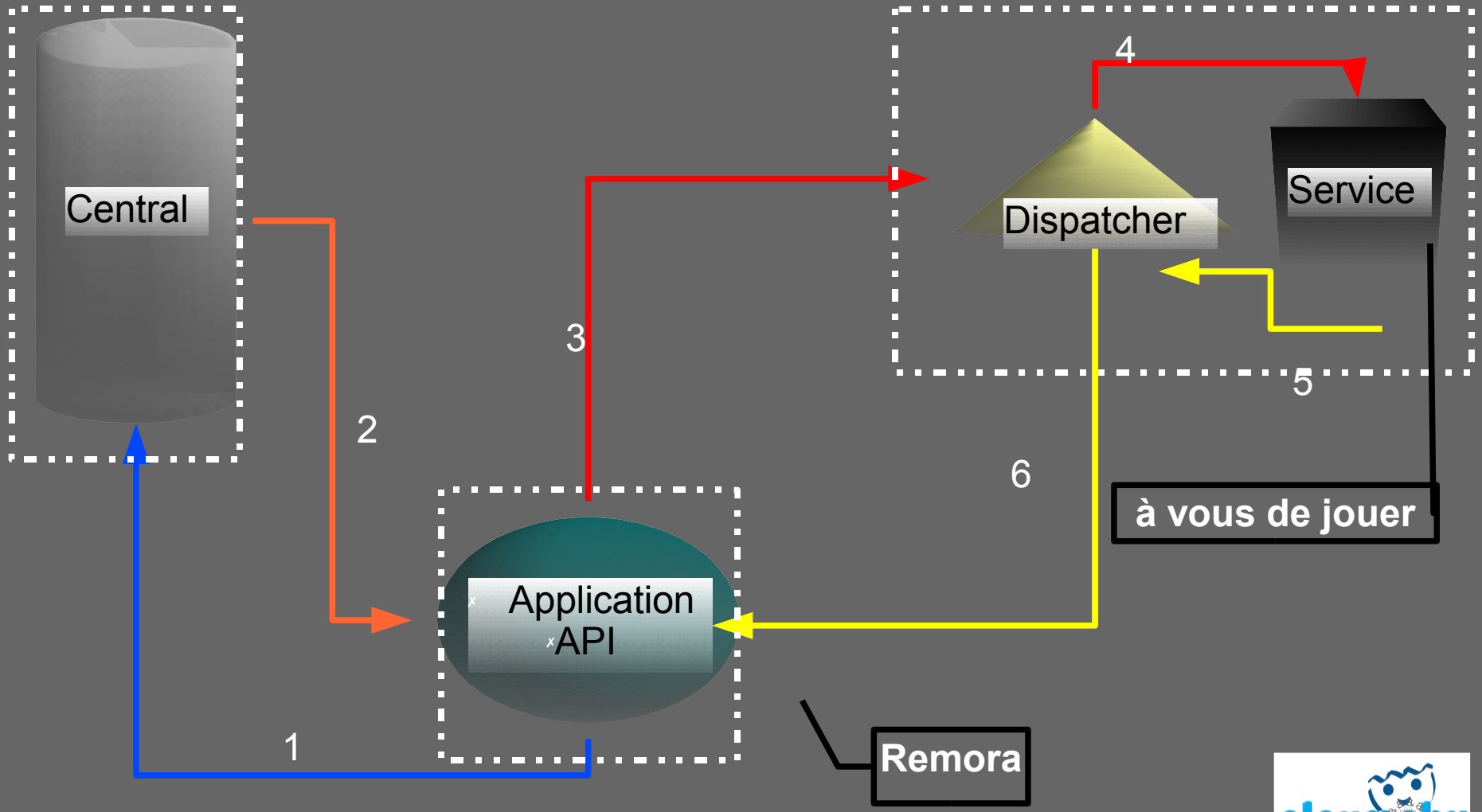
Plan

- **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
  - enregistrement/suppression

- **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 les 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 etre definit par un ID et un *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,..)**

à quoi ressemble 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>
```

comment est structuré 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
}
```

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

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~~)

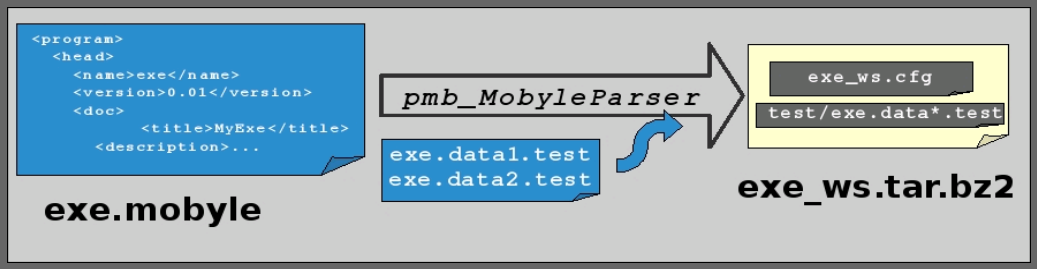
- on peut toujours utiliser ces programmes en ligne de commande
- on peut les encapsuler via d'autres technologies (CGI, Mobyle)

3 étapes

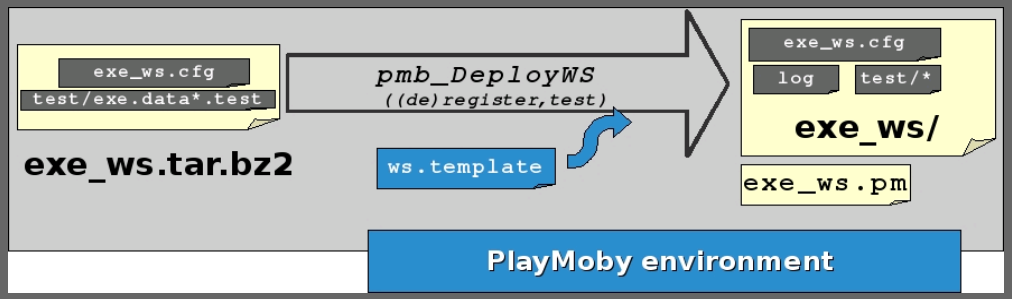
1. Génération d'un fichier de description [Mobyle XML]

- 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



Enregistrement d'un web-service

Parametres:

Description du service: nom, description, URI d'authentification
Interfaces: entrees / sorties / parametres
Acces: signature url (URL RDF), dispatcher url

Script:

pmb_RegisterService.pl –central_conf <central.cfg>
–service_conf <default.cfg>
–module <services.pm>
[–special_conf]

Retour:

if success: **generation du fichier RDF**
ajout au fichier dispatcher

Suppression d'un web-service

Parametres:

Acces: signature url (Url du RDF)

Script:

pmb_DeregisterService.pl –central_conf <central.cfg>
–service_conf <default.cfg>
--remove

Retour:

if success: “vidage” du fichier RDF
suppression dans fichier dispatcher

TP

http://symbiose.toulouse.inra.fr/tp_playmoby/

1. Creer l'arborescence de travail

- connection via ssh sur symbiose.toulouse.inra.fr
- %tcsh
- %cd /www/tp_playmoby/$USER
- recuperer l'archive playmoby.tar.bz2
- %tar xvfj playmoby.tar.bz2
- %cat README

TP

http://symbiose.toulouse.inra.fr/tp_playmoby/

2. Décrire un service

- choisir un programme a interfacer (si pas d'idée voir dans .../playmoby/sample)
- décrire le programme via *Appli.pm*
- déployer le web-service dans l'arborescence

TP

http://symbiose.toulouse.inra.fr/tp_playmoby/

3. Enregistrer / Tester / Debugger

Pieges:

- problemes d'ecriture des fichiers tmp/log (droits apache ?)
- fichier test incorrect
- dispatcher mal configuré
- nom du package Perl incorrect

Récapitulatif des Fichiers créés/modifiés

1. pmb_MobyleParser.pl produit l'archive Webservice.bz2 contenant:
   - le fichier .cfg du webservice
   - les données de test
   - un fichier log (vide mais avec les bons droits)

2. pmb_DeployWS.pl --pm
   - crée le fichier Webservice.pm (LE webservice)

3. pmb_DeployWS.pl --register
   - crée le fichier Webservice.pm
   - crée le fichier RDF (annuaire_webservice.rdf)
   - ajoute une ligne dans le fichier cfg/dispatcher/dispatcher.txt

4. pmb_DeployWS.pl --deregister
   - copie le fichier RDF (.bkp) et vide l'original
   - supprime la ligne dans le fichier cfg/dispatcher/dispatcher.txt

5. pmb_DeployWS.pl --test
   - crée la versionMoby des données de test (XML)
   - écrit le resultat du Webservice sur STDOUT