Formation intégration d'outils sous Galaxy

11/09/2014

Cyril Monjeaud & Yvan Le Bras

Plateforme Bio-informatique GenOuest

CNRS UMR 6074 IRISA-INRIA, Campus de Beaulieu, 35042 Rennes Cedex

http://github.com/cmonjeau/formation_galaxy

PLAN

Présentation du toolshed

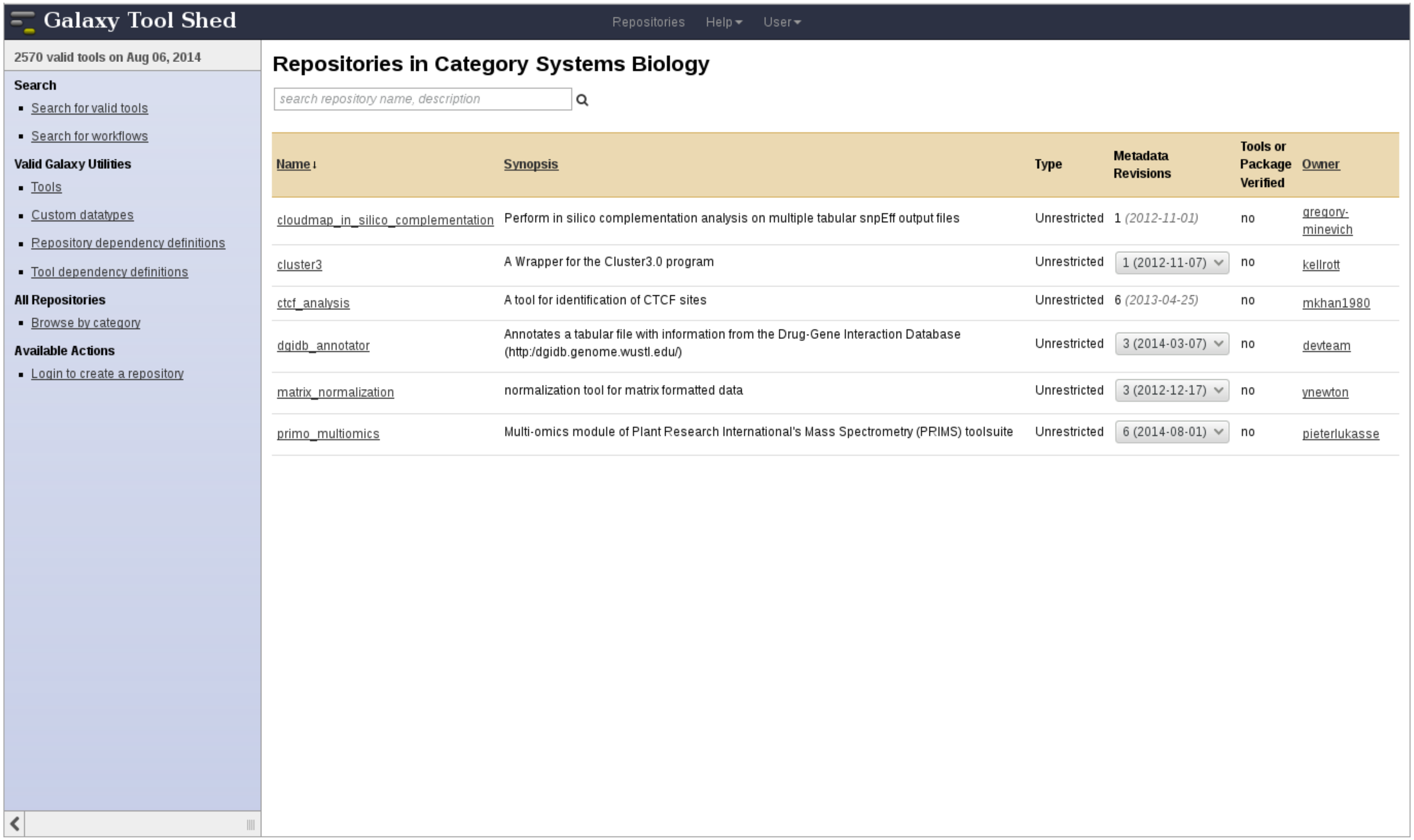
Le toolshed Galaxy

- Appstore d'outils pour Galaxy
  - centralisation des outils
  - intégration rapide dans son instance

- Contient des répertoires contenant :
  - des outils
  - des datatypes
  - des workflows
  - des data managers

- Deux toolsheds principaux
  - toolshed main -> https://toolshed.g2.bx.psu.edu
  - toolshed test -> https://testtoolshed.g2.bx.psu.edu/

Catégorie “Systems Biology”



2570 valid tools on Aug 06, 2014

Search
- Search for valid tools
- Search for workflows

Valid Galaxy Utilities
- Tools
- Custom datatypes
- Repository dependency definitions
- Tool dependency definitions

All Repositories
- Browse by category

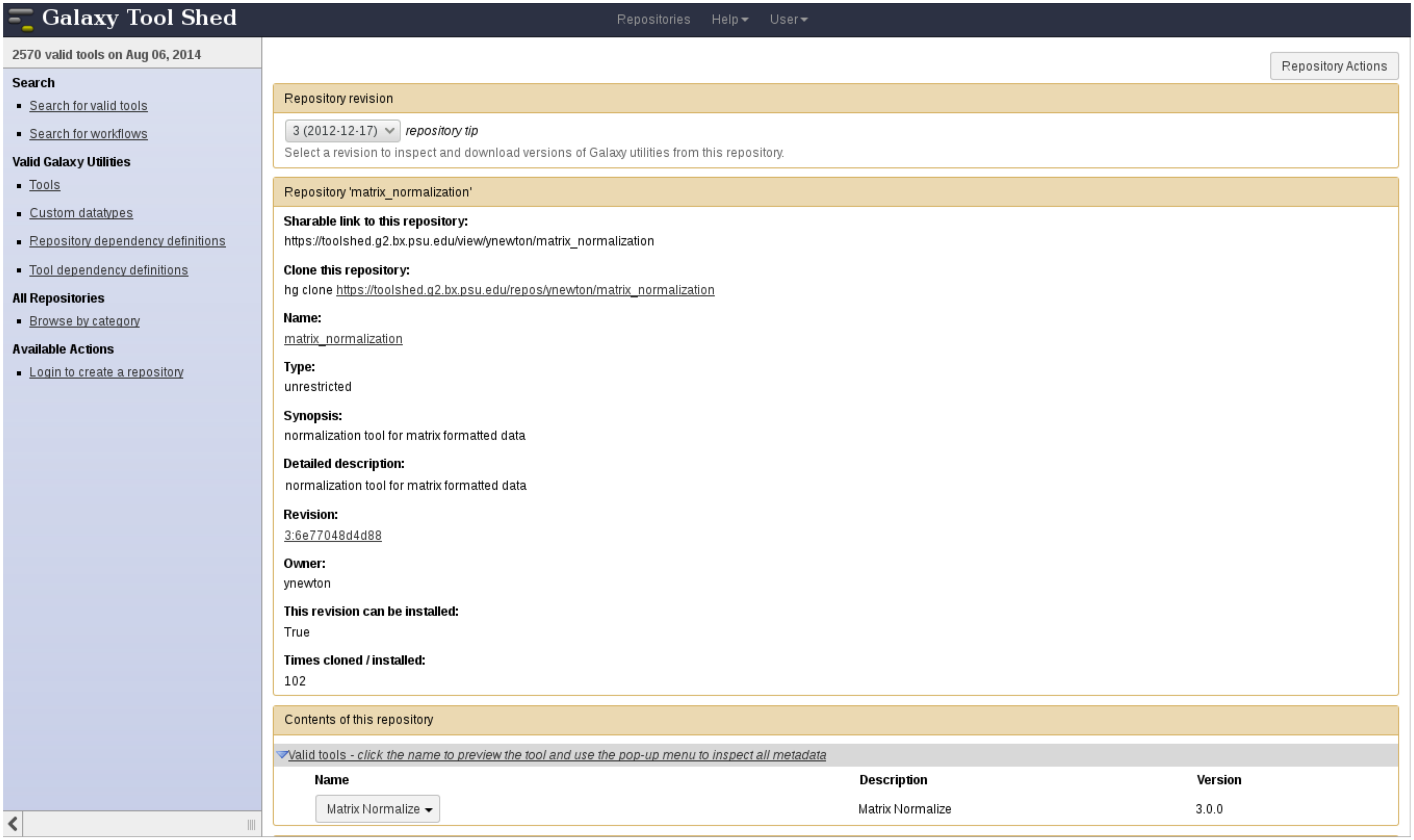
Available Actions
- Login to create a repository

Repositories in Category Systems Biology

search repository name, description

Name↓	Synopsis	Type	Metadata Revisions	Tools or Package Verified	Owner
cloudmap_in_silico_complementation	Perform in silico complementation analysis on multiple tabular snpEff output files	Unrestricted	1 *(2012-11-01)*	no	gregory-minevich
cluster3	A Wrapper for the Cluster3.0 program	Unrestricted	1 (2012-11-07)	no	kellrott
ctcf_analysis	A tool for identification of CTCF sites	Unrestricted	6 *(2013-04-25)*	no	mkhan1980
dgidb_annotator	Annotates a tabular file with information from the Drug-Gene Interaction Database (http:/dgidb.genome.wustl.edu/)	Unrestricted	3 (2014-03-07)	no	devteam
matrix_normalization	normalization tool for matrix formatted data	Unrestricted	3 (2012-12-17)	no	ynewton
primo_multiomics	Multi-omics module of Plant Research International's Mass Spectrometry (PRIMS) toolsuite	Unrestricted	6 (2014-08-01)	no	pieterlukasse

Outil “matrix_normalization”



Repository revision

3 (2012-12-17) *repository tip*

Select a revision to inspect and download versions of Galaxy utilities from this repository.

Repository 'matrix_normalization'

Sharable link to this repository:
https://toolshed.g2.bx.psu.edu/view/ynewton/matrix_normalization

Clone this repository:
hg clone https://toolshed.g2.bx.psu.edu/repos/ynewton/matrix_normalization

Name:
matrix_normalization

Type:
unrestricted

Synopsis:
normalization tool for matrix formatted data

Detailed description:
normalization tool for matrix formatted data

Revision:
3:6e77048d4d88

Owner:
ynewton

This revision can be installed:
True

Times cloned / installed:
102

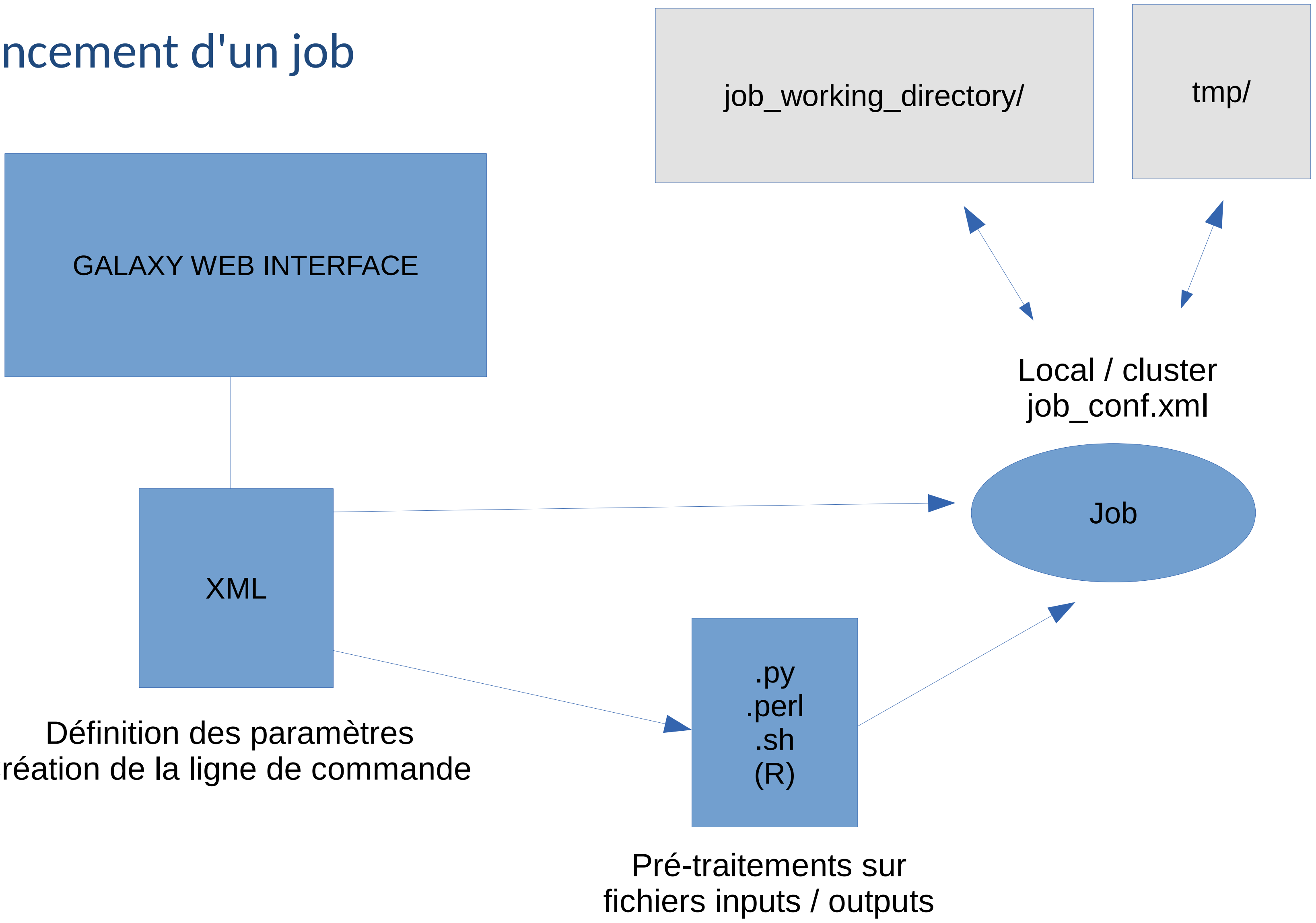
Contents of this repository

Valid tools - *click the name to preview the tool and use the pop-up menu to inspect all metadata*

Name	Description	Version
Matrix Normalize	Matrix Normalize	3.0.0

L'arborescence d'un serveur Galaxy

Lancement d'un job



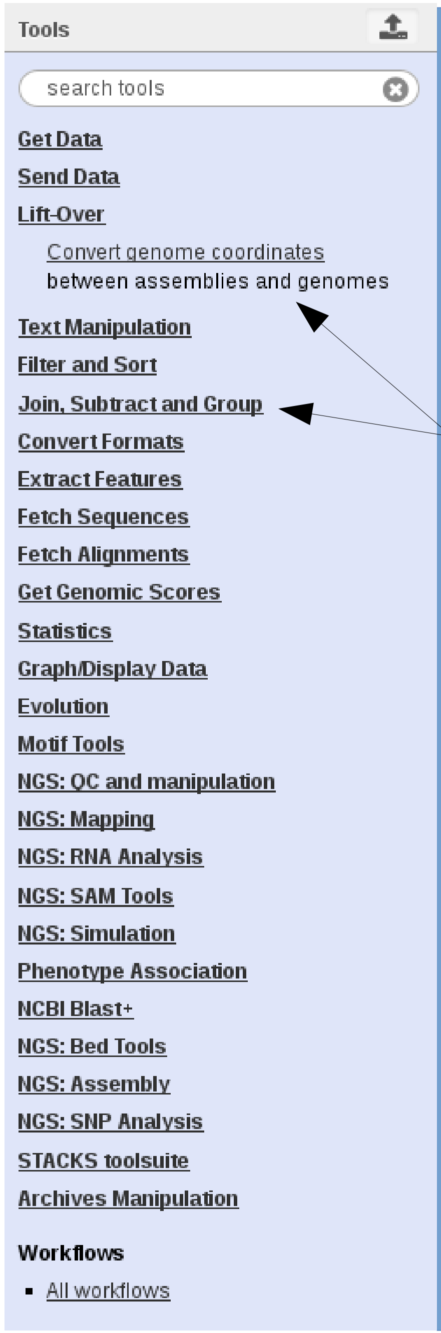
Serveur Galaxy : /opt/galaxy-dist

- **universe_wsgi.ini** $\longrightarrow$ fichier de configuration (database, admin, server, ...)
- tool_conf.xml
- shed_tool_conf.xml
- datatypes_conf.xml
- database/
  - files/
    - 000/
    - 001/
- tools
  - alignment/
    - outil1.xml
    - outil1.py
    - outil2.xml
    - outil2.sh
  - mapping/

Serveur Galaxy : /opt/galaxy-dist

- universe_wsgi.ini
- **tool_conf.xml**
- **shed_tool_conf.xml**
- datatypes_conf.xml
- database/
  - files/
    - 000/
    - 001/
- tools
  - alignment/
    - outil1.xml
    - outil1.py
    - outil2.xml
    - outil2.sh
  - mapping/

$\rightarrow$ ensemble des tools
(emplacements des descripteurs xml, structure du panel)


Tools

search tools

Get Data

Send Data

Lift-Over

Convert genome coordinates between assemblies and genomes

Text Manipulation

Filter and Sort

Join, Subtract and Group

Convert Formats

Extract Features

Fetch Sequences

Fetch Alignments

Get Genomic Scores

Statistics

Graph/Display Data

Evolution

Motif Tools

NGS: QC and manipulation

NGS: Mapping

NGS: RNA Analysis

NGS: SAM Tools

NGS: Simulation

Phenotype Association

NCBI Blast+

NGS: Bed Tools

NGS: Assembly

NGS: SNP Analysis

STACKS toolsuite

Archives Manipulation

Workflows

- All workflows

/opt/galaxy-dist

tool_conf.xml

tools/descripteur.xml

```
<label id="monlabel" text="MonTitre" />
<section id="masection" name="MonTitreDeSection">
   <tool file="tools/descripteur.xml" />
</section>
```

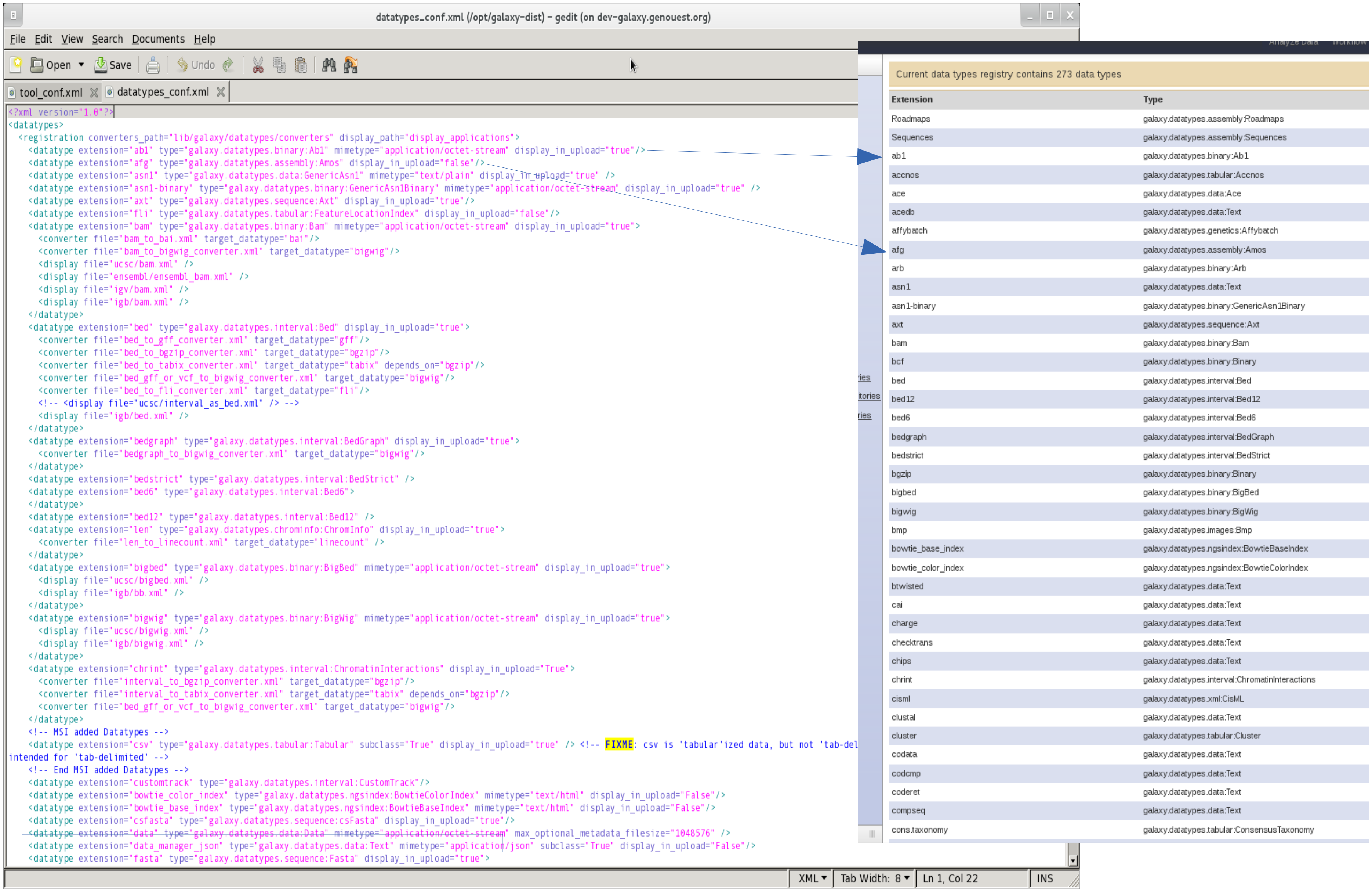
shed_tool_conf.xml

```
<section id="getext" name="Get Data" version="">
  <tool file="cloud-
45.genouest.org/toolshed/repos/cmonjeau/test/e0266fbac9ec/test/discoSNP.xml"
guid="cloud-45.genouest.org/toolshed/repos/cmonjeau/test/discosnp/1.0.0">
      <tool_shed>cloud-45.genouest.org/toolshed</tool_shed>
        <repository_name>test</repository_name>
        <repository_owner>cmonjeau</repository_owner>
        <installed_changeset_revision>e0266fbac9ec</installed_changeset_revision>
        <id>cloud-45.genouest.org/toolshed/repos/cmonjeau/test/discosnp/1.0.0</id>
        <version>1.0.0</version>
    </tool>
</section>
```

Serveur Galaxy : /opt/galaxy-dist

- universe_wsgi.ini
- tool_conf.xml
- shed_tool_conf.xml
- **datatypes_conf.xml** $\rightarrow$ ensemble des datatypes (emplacements vers des classes python)
- database/
  - files/
    - 000/
    - 001/
- tools
  - alignment/
    - outil1.xml
    - outil1.py
    - outil2.xml
    - outil2.sh
  - mapping/

datatypes_conf.xml

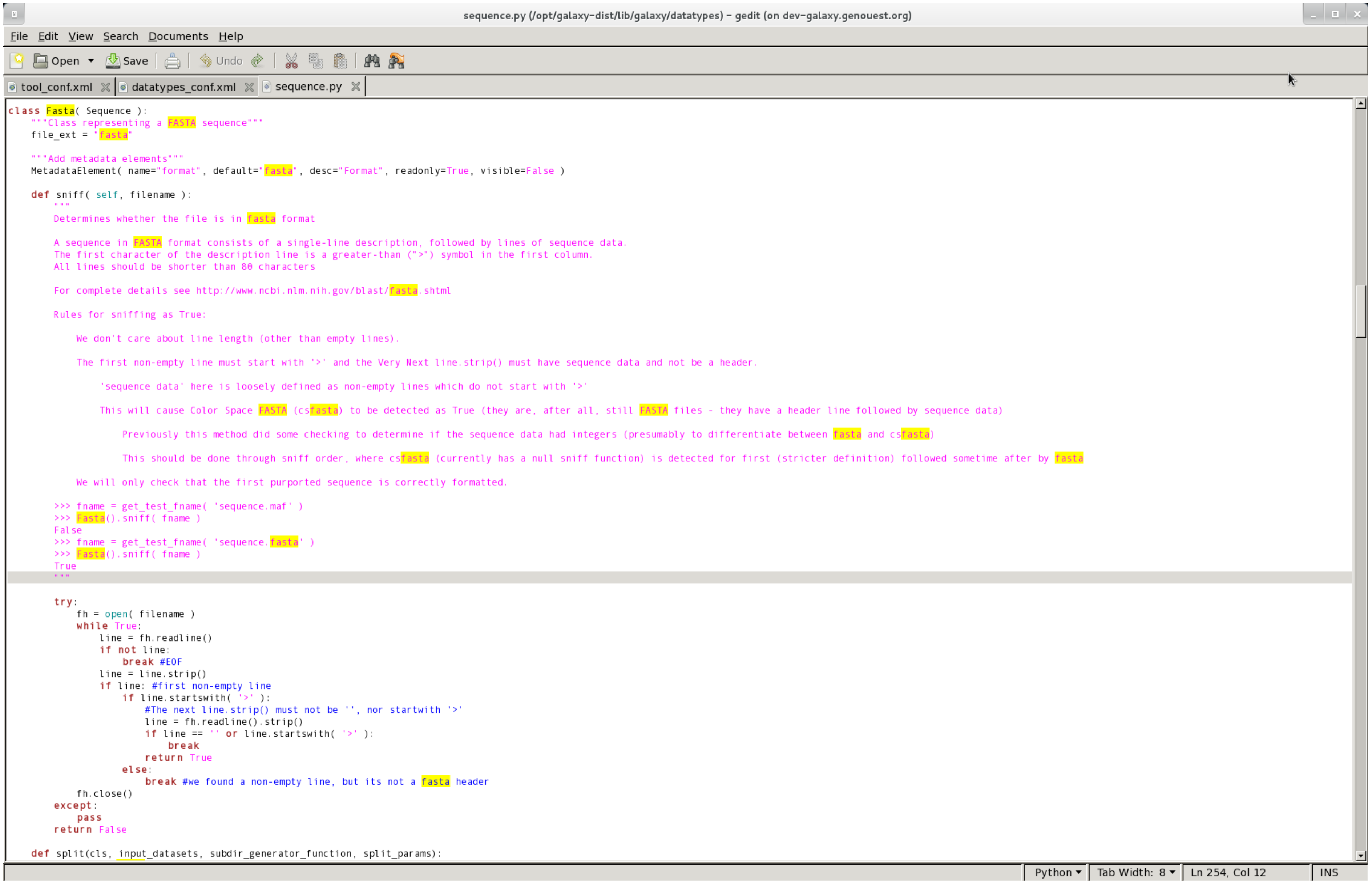

datatypes_conf.xml (/opt/galaxy-dist) - gedit (on dev-galaxy.genouest.org)
File Edit View Search Documents Help
Open Save Undo
tool_conf.xml
datatypes_conf.xml
XML Tab Width: 8 Ln 1, Col 22 INS

```xml
<?xml version="1.0"?>
<datatypes>
  <registration converters_path="lib/galaxy/datatypes/converters" display_path="display_applications">
    <datatype extension="ab1" type="galaxy.datatypes.binary:Ab1" mimetype="application/octet-stream" display_in_upload="true"/>
    <datatype extension="afg" type="galaxy.datatypes.assembly:Amos" display_in_upload="false"/>
    <datatype extension="asn1" type="galaxy.datatypes.data:GenericAsn1" mimetype="text/plain" display_in_upload="true" />
    <datatype extension="asn1-binary" type="galaxy.datatypes.binary:GenericAsn1Binary" mimetype="application/octet-stream" display_in_upload="true" />
    <datatype extension="axt" type="galaxy.datatypes.sequence:Axt" display_in_upload="true"/>
    <datatype extension="fli" type="galaxy.datatypes.tabular:FeatureLocationIndex" display_in_upload="false"/>
    <datatype extension="bam" type="galaxy.datatypes.binary:Bam" mimetype="application/octet-stream" display_in_upload="true">
      <converter file="bam_to_bai.xml" target_datatype="bai"/>
      <converter file="bam_to_bigwig_converter.xml" target_datatype="bigwig"/>
      <display file="ucsc/bam.xml" />
      <display file="ensembl/ensembl_bam.xml" />
      <display file="igv/bam.xml" />
      <display file="igb/bam.xml" />
    </datatype>
    <datatype extension="bed" type="galaxy.datatypes.interval:Bed" display_in_upload="true">
      <converter file="bed_to_gff_converter.xml" target_datatype="gff"/>
      <converter file="bed_to_bgzip_converter.xml" target_datatype="bgzip"/>
      <converter file="bed_to_tabix_converter.xml" target_datatype="tabix" depends_on="bgzip"/>
      <converter file="bed_gff_or_vcf_to_bigwig_converter.xml" target_datatype="bigwig"/>
      <converter file="bed_to_fli_converter.xml" target_datatype="fli"/>
      <!-- <display file="ucsc/interval_as_bed.xml" /> -->
      <display file="igb/bed.xml" />
    </datatype>
    <datatype extension="bedgraph" type="galaxy.datatypes.interval:BedGraph" display_in_upload="true">
      <converter file="bedgraph_to_bigwig_converter.xml" target_datatype="bigwig"/>
    </datatype>
    <datatype extension="bedstrict" type="galaxy.datatypes.interval:BedStrict" />
    <datatype extension="bed6" type="galaxy.datatypes.interval:Bed6">
    </datatype>
    <datatype extension="bed12" type="galaxy.datatypes.interval:Bed12" />
    <datatype extension="len" type="galaxy.datatypes.chrominfo:ChromInfo" display_in_upload="true">
      <converter file="len_to_linecount.xml" target_datatype="linecount" />
    </datatype>
    <datatype extension="bigbed" type="galaxy.datatypes.binary:BigBed" mimetype="application/octet-stream" display_in_upload="true">
      <display file="ucsc/bigbed.xml" />
      <display file="igb/bb.xml" />
    </datatype>
    <datatype extension="bigwig" type="galaxy.datatypes.binary:BigWig" mimetype="application/octet-stream" display_in_upload="true">
      <display file="ucsc/bigwig.xml" />
      <display file="igb/bigwig.xml" />
    </datatype>
    <datatype extension="chrint" type="galaxy.datatypes.interval:ChromatinInteractions" display_in_upload="True">
      <converter file="interval_to_bgzip_converter.xml" target_datatype="bgzip"/>
      <converter file="interval_to_tabix_converter.xml" target_datatype="tabix" depends_on="bgzip"/>
      <converter file="bed_gff_or_vcf_to_bigwig_converter.xml" target_datatype="bigwig"/>
    </datatype>
    <!-- MSI added Datatypes -->
    <datatype extension="csv" type="galaxy.datatypes.tabular:Tabular" subclass="True" display_in_upload="true" /> <!-- FIXME: csv is 'tabular'ized data, but not 'tab-del
intended for 'tab-delimited' -->
    <!-- End MSI added Datatypes -->
    <datatype extension="customtrack" type="galaxy.datatypes.interval:CustomTrack"/>
    <datatype extension="bowtie_color_index" type="galaxy.datatypes.ngsindex:BowtieColorIndex" mimetype="text/html" display_in_upload="False"/>
    <datatype extension="bowtie_base_index" type="galaxy.datatypes.ngsindex:BowtieBaseIndex" mimetype="text/html" display_in_upload="False"/>
    <datatype extension="csfasta" type="galaxy.datatypes.sequence:csFasta" display_in_upload="true"/>
    <datatype extension="data" type="galaxy.datatypes.data:Data" mimetype="application/octet-stream" max_optional_metadata_filesize="1048576" />
    <datatype extension="data_manager_json" type="galaxy.datatypes.data:Text" mimetype="application/json" subclass="True" display_in_upload="False"/>
    <datatype extension="fasta" type="galaxy.datatypes.sequence:Fasta" display_in_upload="true">
```

Current data types registry contains 273 data types

Extension	Type
Roadmaps	galaxy.datatypes.assembly:Roadmaps
Sequences	galaxy.datatypes.assembly:Sequences
ab1	galaxy.datatypes.binary:Ab1
accnos	galaxy.datatypes.tabular:Accnos
ace	galaxy.datatypes.data:Ace
acedb	galaxy.datatypes.data:Text
affybatch	galaxy.datatypes.genetics:Affybatch
afg	galaxy.datatypes.assembly:Amos
arb	galaxy.datatypes.binary:Arb
asn1	galaxy.datatypes.data:Text
asn1-binary	galaxy.datatypes.binary:GenericAsn1Binary
axt	galaxy.datatypes.sequence:Axt
bam	galaxy.datatypes.binary:Bam
bcf	galaxy.datatypes.binary:Binary
bed	galaxy.datatypes.interval:Bed
bed12	galaxy.datatypes.interval:Bed12
bed6	galaxy.datatypes.interval:Bed6
bedgraph	galaxy.datatypes.interval:BedGraph
bedstrict	galaxy.datatypes.interval:BedStrict
bgzip	galaxy.datatypes.binary:Binary
bigbed	galaxy.datatypes.binary:BigBed
bigwig	galaxy.datatypes.binary:BigWig
bmp	galaxy.datatypes.images:Bmp
bowtie_base_index	galaxy.datatypes.ngsindex:BowtieBaseIndex
bowtie_color_index	galaxy.datatypes.ngsindex:BowtieColorIndex
btwisted	galaxy.datatypes.data:Text
cai	galaxy.datatypes.data:Text
charge	galaxy.datatypes.data:Text
checktrans	galaxy.datatypes.data:Text
chips	galaxy.datatypes.data:Text
chrint	galaxy.datatypes.interval:ChromatinInteractions
cisml	galaxy.datatypes.xml:CisML
clustal	galaxy.datatypes.data:Text
cluster	galaxy.datatypes.tabular:Cluster
codata	galaxy.datatypes.data:Text
codcmp	galaxy.datatypes.data:Text
coderet	galaxy.datatypes.data:Text
compseq	galaxy.datatypes.data:Text
cons.taxonomy	galaxy.datatypes.tabular:ConsensusTaxonomy

lib/galaxy/datatypes/sequence.py


sequence.py (/opt/galaxy-dist/lib/galaxy/datatypes) - gedit (on dev-galaxy.genouest.org)
File Edit View Search Documents Help
Open Save Undo
tool_conf.xml
datatypes_conf.xml
sequence.py
Python
Tab Width: 8
Ln 254, Col 12
INS

```python
class Fasta( Sequence ):
    """Class representing a FASTA sequence"""
    file_ext = "fasta"

    """Add metadata elements"""
    MetadataElement( name="format", default="fasta", desc="Format", readonly=True, visible=False )

    def sniff( self, filename ):
        """
        Determines whether the file is in fasta format

        A sequence in FASTA format consists of a single-line description, followed by lines of sequence data.
        The first character of the description line is a greater-than (">") symbol in the first column.
        All lines should be shorter than 80 characters

        For complete details see http://www.ncbi.nlm.nih.gov/blast/fasta.shtml

        Rules for sniffing as True:

            We don't care about line length (other than empty lines).

            The first non-empty line must start with '>' and the Very Next line.strip() must have sequence data and not be a header.

                'sequence data' here is loosely defined as non-empty lines which do not start with '>'

                This will cause Color Space FASTA (csfasta) to be detected as True (they are, after all, still FASTA files - they have a header line followed by sequence data)

                    Previously this method did some checking to determine if the sequence data had integers (presumably to differentiate between fasta and csfasta)

                    This should be done through sniff order, where csfasta (currently has a null sniff function) is detected for first (stricter definition) followed sometime after by fasta

            We will only check that the first purported sequence is correctly formatted.

        >>> fname = get_test_fname( 'sequence.maf' )
        >>> Fasta().sniff( fname )
        False
        >>> fname = get_test_fname( 'sequence.fasta' )
        >>> Fasta().sniff( fname )
        True
        """

        try:
            fh = open( filename )
            while True:
                line = fh.readline()
                if not line:
                    break #EOF
                line = line.strip()
                if line: #first non-empty line
                    if line.startswith( '>' ):
                        #The next line.strip() must not be '', nor startwith '>'
                        line = fh.readline().strip()
                        if line == '' or line.startswith( '>' ):
                            break
                        return True
                    else:
                        break #we found a non-empty line, but its not a fasta header
            fh.close()
        except:
            pass
        return False

    def split(cls, input_datasets, subdir_generator_function, split_params):
```

Serveur Galaxy : /opt/galaxy-dist

- universe_wsgi.ini
- tool_conf.xml
- shed_tool_conf.xml
- datatypes_conf.xml
- **database/**
  - **files/** $\longrightarrow$ ensemble des fichiers de données (input/output)
    - **000/**
    - **001/**
- tools
  - alignment/
    - outil1.xml
    - outil1.py
    - outil2.xml
    - outil2.sh
  - mapping/

Serveur Galaxy : /opt/galaxy-dist

- universe_wsgi.ini
- tool_conf.xml
- shed_tool_conf.xml
- datatypes_conf.xml
- database/
  - files/
    - 000/
    - 001/
- **tools**
  - **alignment/**
    - **outil1.xml**
    - **outil1.py**
    - **outil2.xml**
    - **outil2.sh**
  - **mapping/**

$\longrightarrow$ Emplacement des descripteurs / wrappers intégrés directement dans Galaxy

Partie toolshed : /opt/shed_tools

- /opt/shed_tools/
  - **cloud-45.genouest.org/** $\longrightarrow$ Source du toolshed
    - toolshed/repos/
      - cmonjeau/
      - **ylebras/**

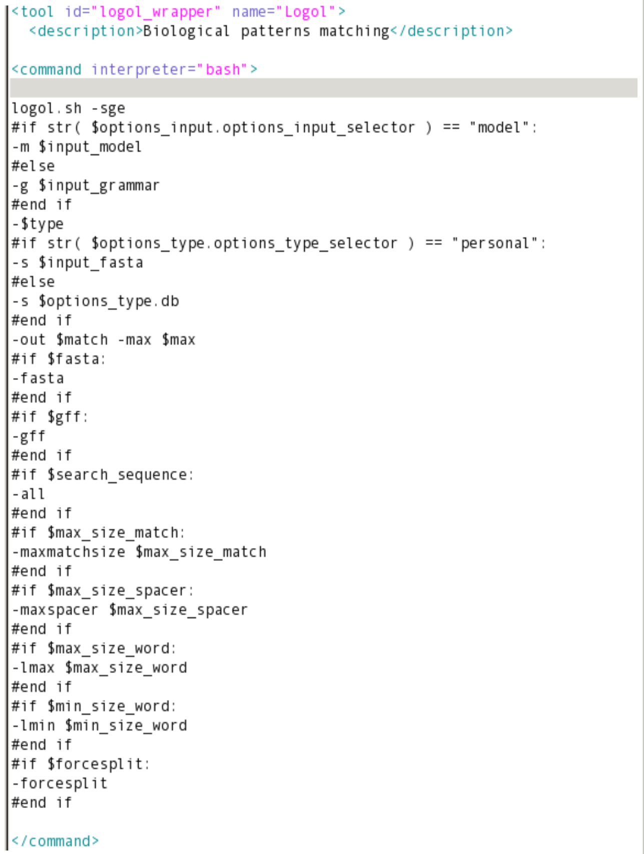
        $\longrightarrow$ Propriétaire du repository
        - discosnp/
        - **takeabreak/**

          $\longrightarrow$ Repositories
          - **e45d1faade2c/** $\longrightarrow$ Révision
            - takeabreak
              - **takeabreak.xml**
              - **takeabreak.py**

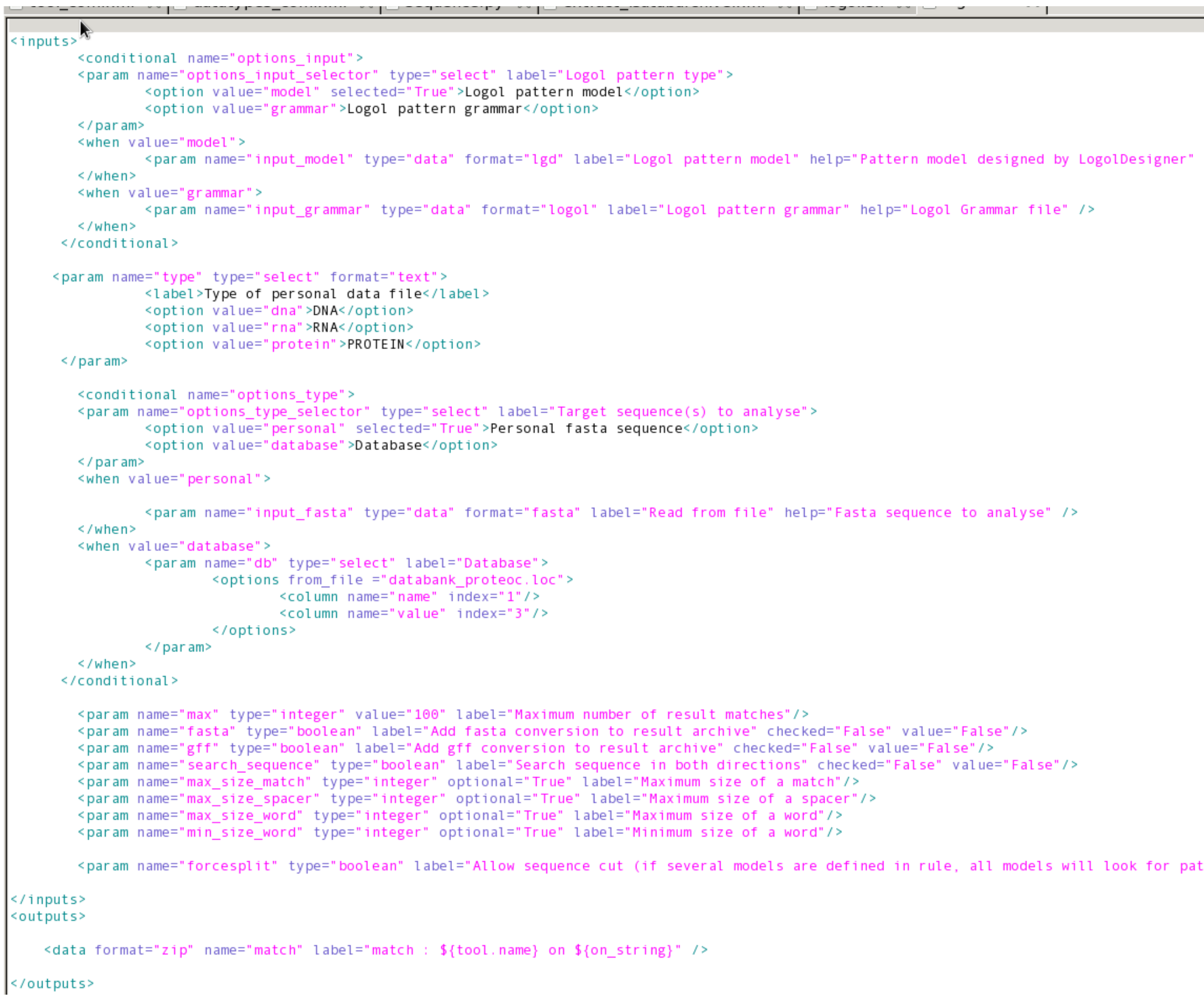
              $\longrightarrow$ Outils

Exemple d'outil : Logol

Le descripteur : logol.xml



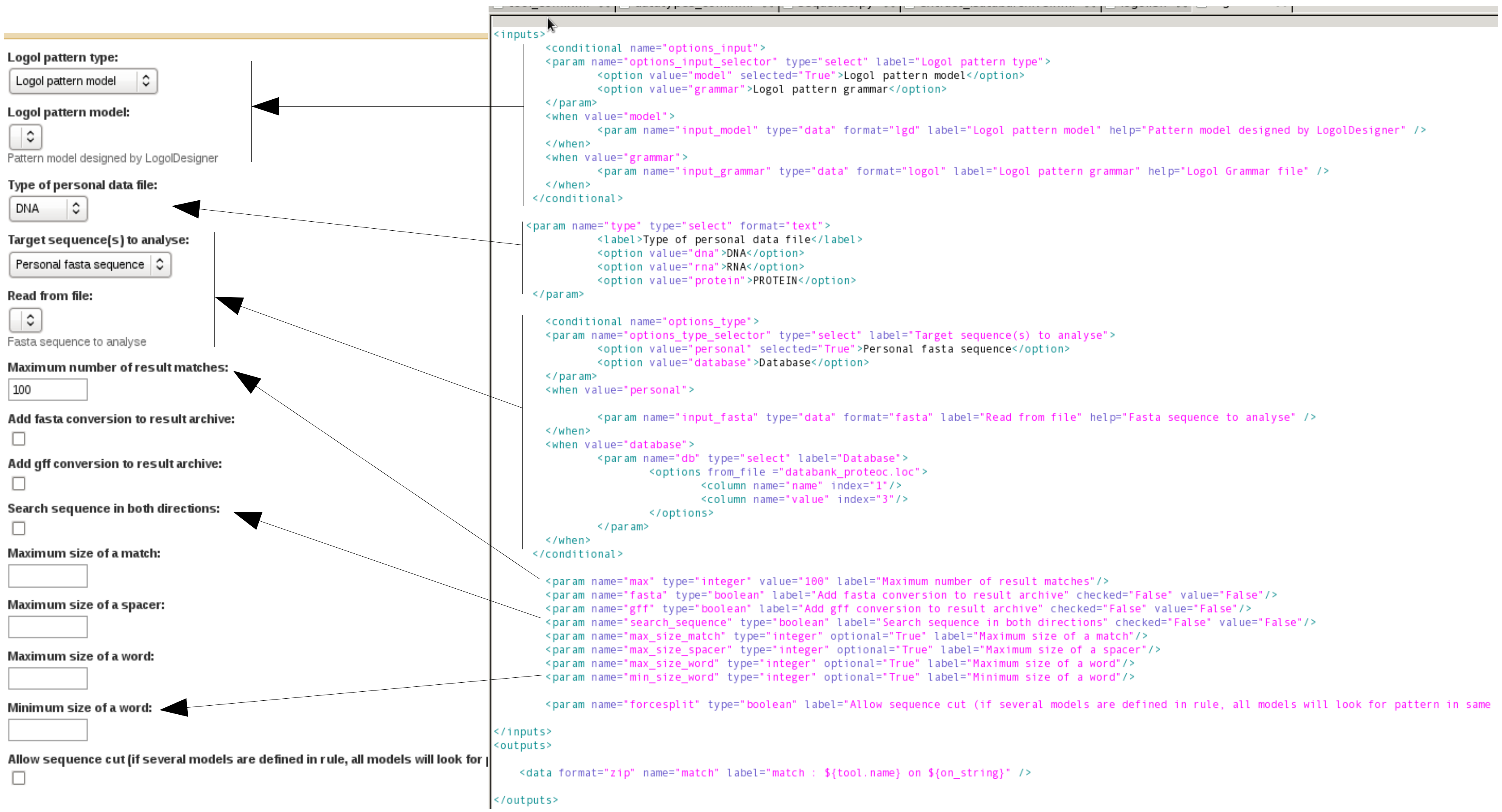
```
<tool id="logol_wrapper" name="Logol">
  <description>Biological patterns matching</description>

<command interpreter="bash">

logol.sh -sge
#if str( $options_input.options_input_selector ) == "model":
-m $input_model
#else
-g $input_grammar
#end if
-$type
#if str( $options_type.options_type_selector ) == "personal":
-s $input_fasta
#else
-s $options_type.db
#end if
-out $match -max $max
#if $fasta:
-fasta
#end if
#if $gff:
-gff
#end if
#if $search_sequence:
-all
#end if
#if $max_size_match:
-maxmatchsize $max_size_match
#end if
#if $max_size_spacer:
-maxspacer $max_size_spacer
#end if
#if $max_size_word:
-lmax $max_size_word
#end if
#if $min_size_word:
-lmin $min_size_word
#end if
#if $forcesplit:
-forcesplit
#end if

</command>
```



```
<inputs>
        <conditional name="options_input">
        <param name="options_input_selector" type="select" label="Logol pattern type">
                <option value="model" selected="True">Logol pattern model</option>
                <option value="grammar">Logol pattern grammar</option>
        </param>
        <when value="model">
                <param name="input_model" type="data" format="lgd" label="Logol pattern model" help="Pattern model designed by LogolDesigner"
        </when>
        <when value="grammar">
                <param name="input_grammar" type="data" format="logol" label="Logol pattern grammar" help="Logol Grammar file" />
        </when>
      </conditional>

     <param name="type" type="select" format="text">
                <label>Type of personal data file</label>
                <option value="dna">DNA</option>
                <option value="rna">RNA</option>
                <option value="protein">PROTEIN</option>
      </param>

        <conditional name="options_type">
        <param name="options_type_selector" type="select" label="Target sequence(s) to analyse">
                <option value="personal" selected="True">Personal fasta sequence</option>
                <option value="database">Database</option>
        </param>
        <when value="personal">

                <param name="input_fasta" type="data" format="fasta" label="Read from file" help="Fasta sequence to analyse" />
        </when>
        <when value="database">
                <param name="db" type="select" label="Database">
                        <options from_file ="databank_proteoc.loc">
                                <column name="name" index="1"/>
                                <column name="value" index="3"/>
                        </options>
                </param>
        </when>
      </conditional>

        <param name="max" type="integer" value="100" label="Maximum number of result matches"/>
        <param name="fasta" type="boolean" label="Add fasta conversion to result archive" checked="False" value="False"/>
        <param name="gff" type="boolean" label="Add gff conversion to result archive" checked="False" value="False"/>
        <param name="search_sequence" type="boolean" label="Search sequence in both directions" checked="False" value="False"/>
        <param name="max_size_match" type="integer" optional="True" label="Maximum size of a match"/>
        <param name="max_size_spacer" type="integer" optional="True" label="Maximum size of a spacer"/>
        <param name="max_size_word" type="integer" optional="True" label="Maximum size of a word"/>
        <param name="min_size_word" type="integer" optional="True" label="Minimum size of a word"/>

        <param name="forcesplit" type="boolean" label="Allow sequence cut (if several models are defined in rule, all models will look for pat

</inputs>
<outputs>

    <data format="zip" name="match" label="match : ${tool.name} on ${on_string}" />

</outputs>
```

http://wiki.galaxyproject.org/Admin/Tools/ToolConfigSyntax

Le descripteur : logol.xml



Logol pattern type:

Logol pattern model

Logol pattern model:

Pattern model designed by LogolDesigner

Type of personal data file:

DNA

Target sequence(s) to analyse:

Personal fasta sequence

Read from file:

Fasta sequence to analyse

Maximum number of result matches:

100

Add fasta conversion to result archive:

☐

Add gff conversion to result archive:

☐

Search sequence in both directions:

☐

Maximum size of a match:

Maximum size of a spacer:

Maximum size of a word:

Minimum size of a word:

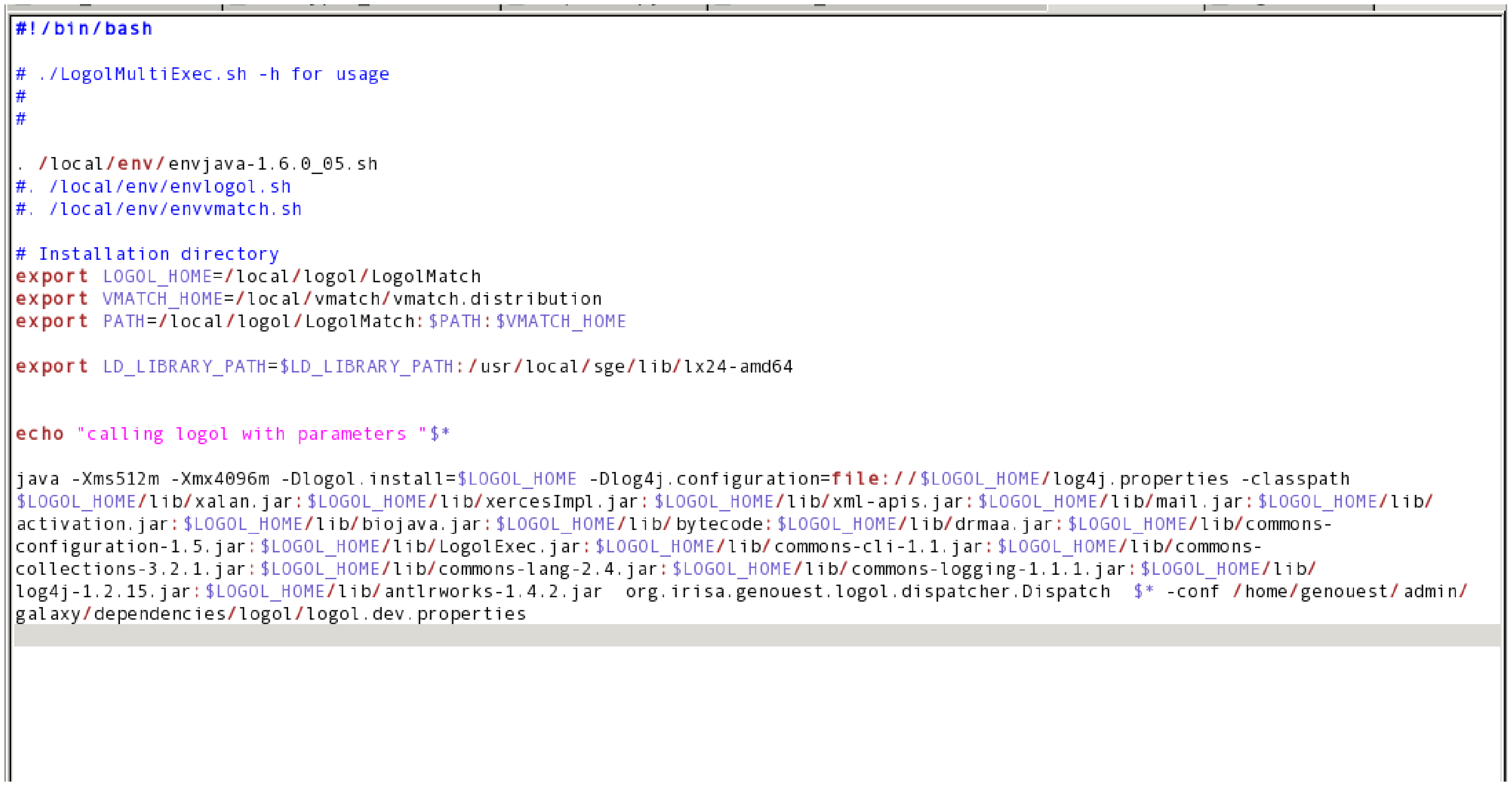
Allow sequence cut (if several models are defined in rule, all models will look for

☐

```
<inputs>
    <conditional name="options_input">
    <param name="options_input_selector" type="select" label="Logol pattern type">
            <option value="model" selected="True">Logol pattern model</option>
            <option value="grammar">Logol pattern grammar</option>
    </param>
    <when value="model">
            <param name="input_model" type="data" format="lgd" label="Logol pattern model" help="Pattern model designed by LogolDesigner" />
    </when>
    <when value="grammar">
            <param name="input_grammar" type="data" format="logol" label="Logol pattern grammar" help="Logol Grammar file" />
    </when>
  </conditional>

  <param name="type" type="select" format="text">
            <label>Type of personal data file</label>
            <option value="dna">DNA</option>
            <option value="rna">RNA</option>
            <option value="protein">PROTEIN</option>
  </param>

    <conditional name="options_type">
    <param name="options_type_selector" type="select" label="Target sequence(s) to analyse">
            <option value="personal" selected="True">Personal fasta sequence</option>
            <option value="database">Database</option>
    </param>
    <when value="personal">

            <param name="input_fasta" type="data" format="fasta" label="Read from file" help="Fasta sequence to analyse" />
    </when>
    <when value="database">
            <param name="db" type="select" label="Database">
                    <options from_file ="databank_proteoc.loc">
                            <column name="name" index="1"/>
                            <column name="value" index="3"/>
                    </options>
            </param>
    </when>
  </conditional>

    <param name="max" type="integer" value="100" label="Maximum number of result matches"/>
    <param name="fasta" type="boolean" label="Add fasta conversion to result archive" checked="False" value="False"/>
    <param name="gff" type="boolean" label="Add gff conversion to result archive" checked="False" value="False"/>
    <param name="search_sequence" type="boolean" label="Search sequence in both directions" checked="False" value="False"/>
    <param name="max_size_match" type="integer" optional="True" label="Maximum size of a match"/>
    <param name="max_size_spacer" type="integer" optional="True" label="Maximum size of a spacer"/>
    <param name="max_size_word" type="integer" optional="True" label="Maximum size of a word"/>
    <param name="min_size_word" type="integer" optional="True" label="Minimum size of a word"/>

    <param name="forcesplit" type="boolean" label="Allow sequence cut (if several models are defined in rule, all models will look for pattern in same

</inputs>
<outputs>

  <data format="zip" name="match" label="match : ${tool.name} on ${on_string}" />

</outputs>
```

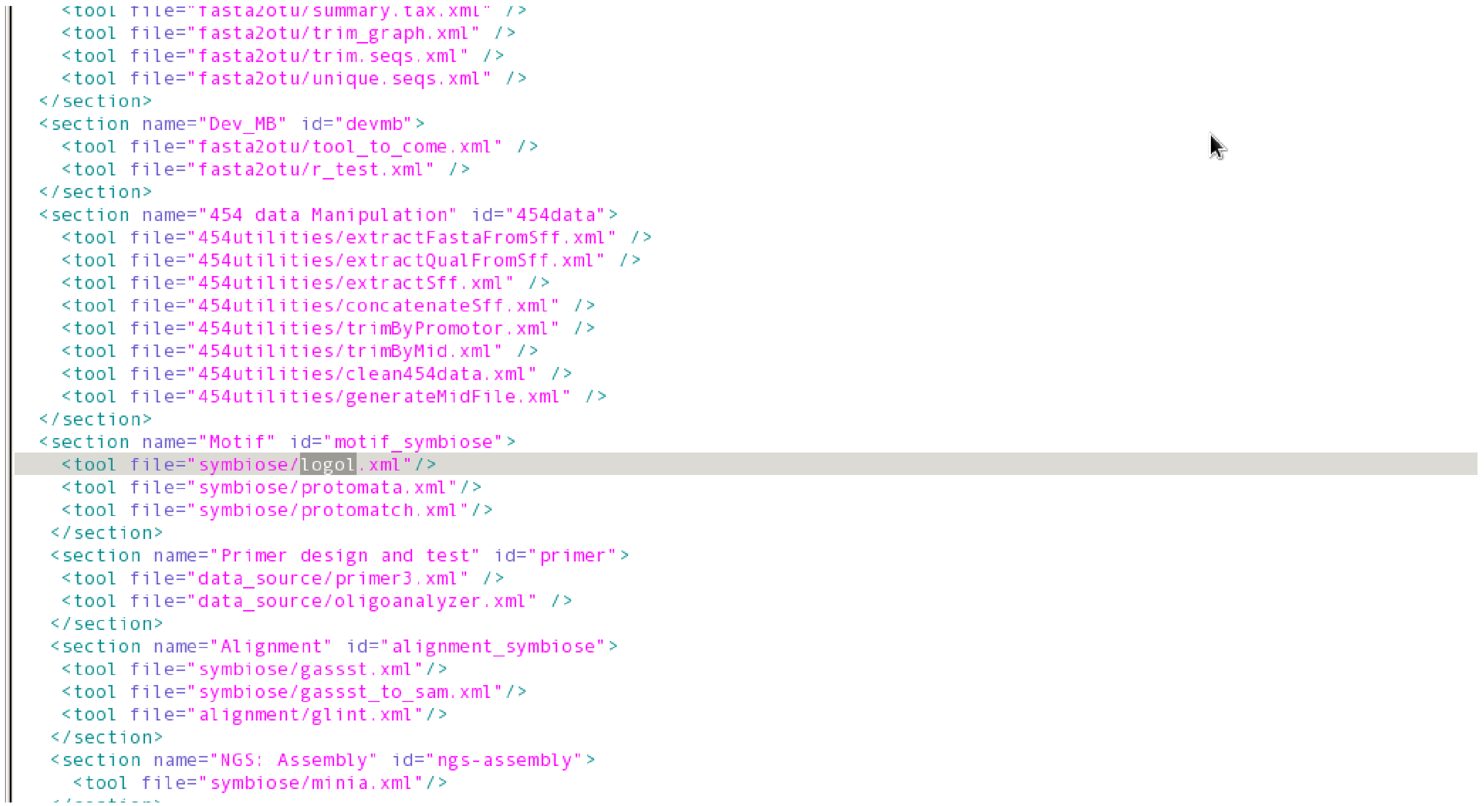
http://wiki.galaxyproject.org/Admin/Tools/ToolConfigSyntax

Le wrapper : logol.sh



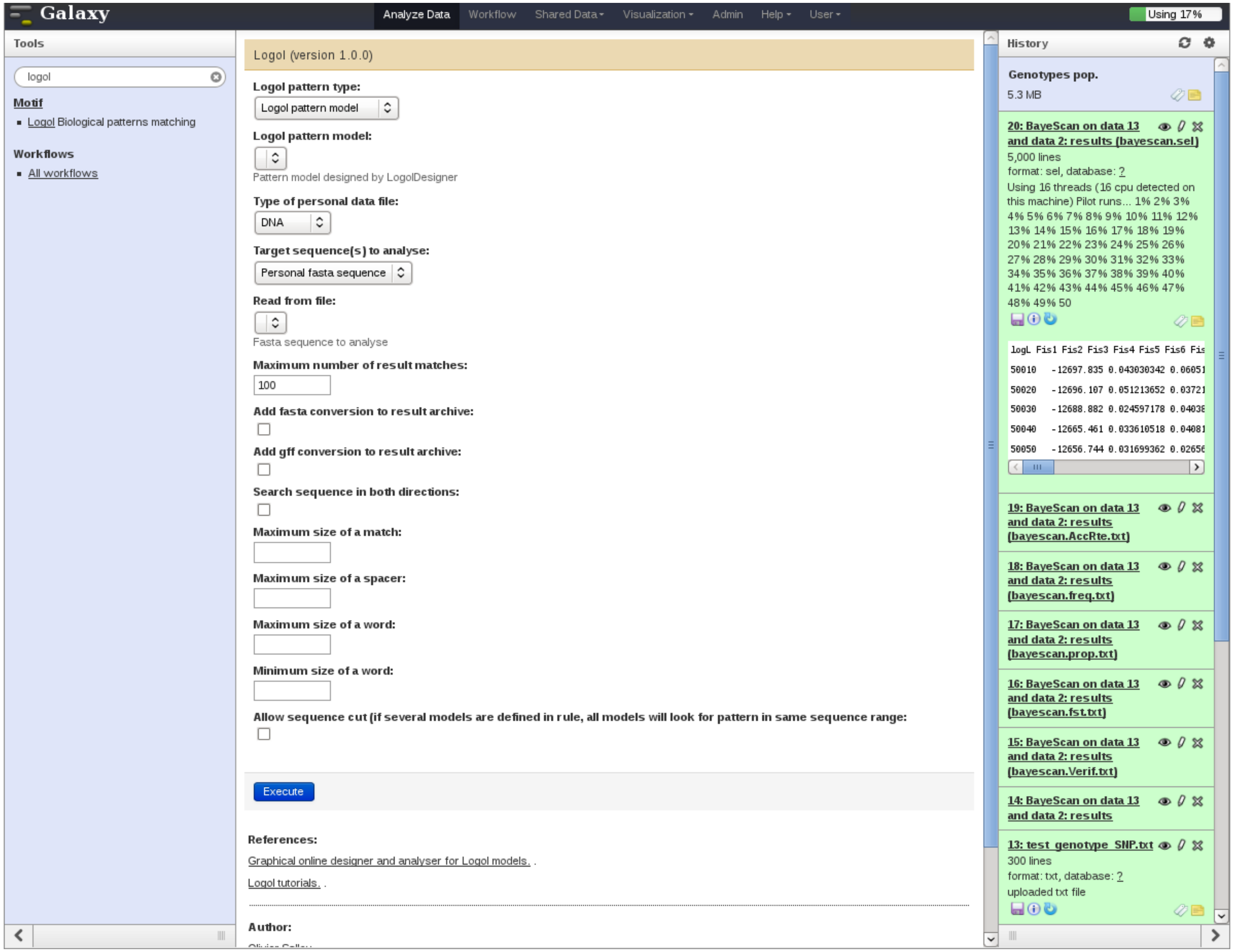
```
#!/bin/bash

# ./LogolMultiExec.sh -h for usage
#
#

. /local/env/envjava-1.6.0_05.sh
#. /local/env/envlogol.sh
#. /local/env/envvmatch.sh

# Installation directory
export LOGOL_HOME=/local/logol/LogolMatch
export VMATCH_HOME=/local/vmatch/vmatch.distribution
export PATH=/local/logol/LogolMatch:$PATH:$VMATCH_HOME

export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/sge/lib/lx24-amd64


echo "calling logol with parameters "$*

java -Xms512m -Xmx4096m -Dlogol.install=$LOGOL_HOME -Dlog4j.configuration=file://$LOGOL_HOME/log4j.properties -classpath
$LOGOL_HOME/lib/xalan.jar:$LOGOL_HOME/lib/xercesImpl.jar:$LOGOL_HOME/lib/xml-apis.jar:$LOGOL_HOME/lib/mail.jar:$LOGOL_HOME/lib/
activation.jar:$LOGOL_HOME/lib/biojava.jar:$LOGOL_HOME/lib/bytecode:$LOGOL_HOME/lib/drmaa.jar:$LOGOL_HOME/lib/commons-
configuration-1.5.jar:$LOGOL_HOME/lib/LogolExec.jar:$LOGOL_HOME/lib/commons-cli-1.1.jar:$LOGOL_HOME/lib/commons-
collections-3.2.1.jar:$LOGOL_HOME/lib/commons-lang-2.4.jar:$LOGOL_HOME/lib/commons-logging-1.1.1.jar:$LOGOL_HOME/lib/
log4j-1.2.15.jar:$LOGOL_HOME/lib/antlrworks-1.4.2.jar  org.irisa.genouest.logol.dispatcher.Dispatch  $* -conf /home/genouest/admin/
galaxy/dependencies/logol/logol.dev.properties
```

Le fichier de configuration : tool_conf.xml



```
    <tool file="fasta2otu/summary.tax.xml" />
    <tool file="fasta2otu/trim_graph.xml" />
    <tool file="fasta2otu/trim.seqs.xml" />
    <tool file="fasta2otu/unique.seqs.xml" />
  </section>
  <section name="Dev_MB" id="devmb">
    <tool file="fasta2otu/tool_to_come.xml" />
    <tool file="fasta2otu/r_test.xml" />
  </section>
  <section name="454 data Manipulation" id="454data">
    <tool file="454utilities/extractFastaFromSff.xml" />
    <tool file="454utilities/extractQualFromSff.xml" />
    <tool file="454utilities/extractSff.xml" />
    <tool file="454utilities/concatenateSff.xml" />
    <tool file="454utilities/trimByPromotor.xml" />
    <tool file="454utilities/trimByMid.xml" />
    <tool file="454utilities/clean454data.xml" />
    <tool file="454utilities/generateMidFile.xml" />
  </section>
  <section name="Motif" id="motif_symbiose">
    <tool file="symbiose/logol.xml"/>
    <tool file="symbiose/protomata.xml"/>
    <tool file="symbiose/protomatch.xml"/>
   </section>
   <section name="Primer design and test" id="primer">
    <tool file="data_source/primer3.xml" />
    <tool file="data_source/oligoanalyzer.xml" />
   </section>
   <section name="Alignment" id="alignment_symbiose">
    <tool file="symbiose/gassst.xml"/>
    <tool file="symbiose/gassst_to_sam.xml"/>
    <tool file="alignment/glint.xml"/>
   </section>
   <section name="NGS: Assembly" id="ngs-assembly">
     <tool file="symbiose/minia.xml"/>
```

Aperçu dans Galaxy



logol

Motif

- Logol Biological patterns matching

Workflows

- All workflows

Logol (version 1.0.0)

Logol pattern type: Logol pattern model

Logol pattern model:

Pattern model designed by LogolDesigner

Type of personal data file: DNA

Target sequence(s) to analyse: Personal fasta sequence

Read from file:

Fasta sequence to analyse

Maximum number of result matches: 100

Add fasta conversion to result archive:

Add gff conversion to result archive:

Search sequence in both directions:

Maximum size of a match:

Maximum size of a spacer:

Maximum size of a word:

Minimum size of a word:

Allow sequence cut (if several models are defined in rule, all models will look for pattern in same sequence range:

References:

Graphical online designer and analyser for Logol models. .

Logol tutorials. .

Author:

Genotypes pop.

5.3 MB

20: BayeScan on data 13 and data 2: results (bayescan.sel)

5,000 lines

format: sel, database: ?

Using 16 threads (16 cpu detected on this machine) Pilot runs... 1% 2% 3% 4% 5% 6% 7% 8% 9% 10% 11% 12% 13% 14% 15% 16% 17% 18% 19% 20% 21% 22% 23% 24% 25% 26% 27% 28% 29% 30% 31% 32% 33% 34% 35% 36% 37% 38% 39% 40% 41% 42% 43% 44% 45% 46% 47% 48% 49% 50

```
logL Fis1 Fis2 Fis3 Fis4 Fis5 Fis6 Fis
50010   -12697.835 0.043030342 0.06051
50020   -12696.107 0.051213652 0.03721
50030   -12688.882 0.024597178 0.04038
50040   -12665.461 0.033610518 0.04081
50050   -12656.744 0.031699362 0.02656
```

19: BayeScan on data 13 and data 2: results (bayescan.AccRte.txt)

18: BayeScan on data 13 and data 2: results (bayescan.freq.txt)

17: BayeScan on data 13 and data 2: results (bayescan.prop.txt)

16: BayeScan on data 13 and data 2: results (bayescan.fst.txt)

15: BayeScan on data 13 and data 2: results (bayescan.Verif.txt)

14: BayeScan on data 13 and data 2: results

13: test_genotype_SNP.txt

300 lines

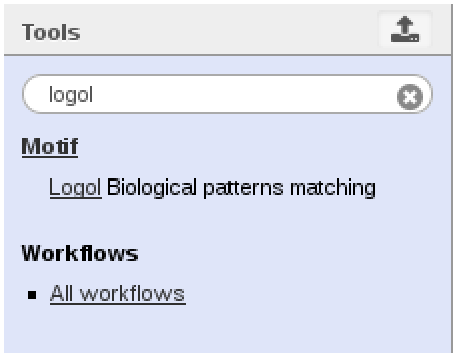
format: txt, database: ?

uploaded txt file

La syntaxe des descripteurs

http://wiki.galaxyproject.org/Admin/Tools/ToolConfigSyntax

Balises les plus importantes


Tools

logol

Motif

Logol Biological patterns matching

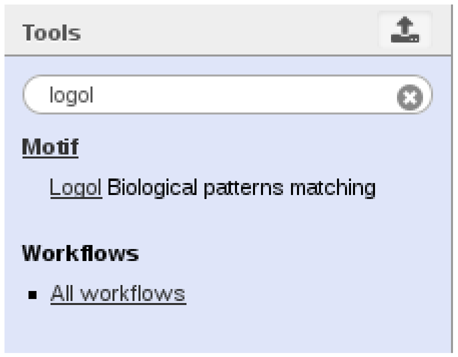
Workflows

- All workflows

```
<tool id="logol_wrapper" name="Logol">
    <description>Biological patterns matching</description>
    <command interpreter="bash">
            logol.sh $options.input.options_input_selector $option_input.input_model $type > $match
    </command>
    <inputs>
        <conditional name="options_input">
                <param name="options_input_selector" type="select" label="Logol pattern type">
                  <option value="model" selected="True">Logol pattern model</option>
                  <option value="grammar">Logol pattern grammar</option>
                </param>
                <when value="model">
                  <param name="input_model" type="data" format="lgd" label="Logol pattern model" help="Pattern model designed by LogolDesigner" />
                </when>
        </conditional>
        <param name="type" type="select" format="text" label="type">
            <option value="dna">DNA</option>
            <option value="rna">RNA</option>
            <option value="protein">PROTEIN</option>
        </param>
        <param name="max" type="integer" value="100" label="Maximum number of result matches"/>
        <param name="fasta" type="boolean" label="Add fasta conversion to result archive" checked="False" value="False"/>
    </inputs>
    <outputs>
          <data format="text" name="match" label="match : ${tool.name} on ${on_string}" />
    </outputs>
    <help>
    </help>
</tool>
```

Syntaxes les plus importantes

```
<tool id="logol_wrapper" name="Logol">
    <description>Biological patterns matching</description>
    <command interpreter="bash">
            logol.sh $options.input.options_input_selector $option_input.input_model $type > $match
    </command>
    <inputs>
        <conditional name="options_input">
                <param name="options_input_selector" type="select" label="Logol pattern type">
                  <option value="model" selected="True">Logol pattern model</option>
                  <option value="grammar">Logol pattern grammar</option>
                </param>
                <when value="model">
                  <param name="input_model" type="data" format="lgd" label="Logol pattern model" help="Pattern model designed by LogolDesigner" />
                </when>
        </conditional>
        <param name="type" type="select" format="text" label="type">
            <option value="dna">DNA</option>
            <option value="rna">RNA</option>
            <option value="protein">PROTEIN</option>
        </param>
        <param name="max" type="integer" value="100" label="Maximum number of result matches"/>
        <param name="fasta" type="boolean" label="Add fasta conversion to result archive" checked="False" value="False"/>
    </inputs>
    <outputs>
          <data format="text" name="match" label="match : ${tool.name} on ${on_string}" />
    </outputs>
    <help>
    </help>
</tool>
```

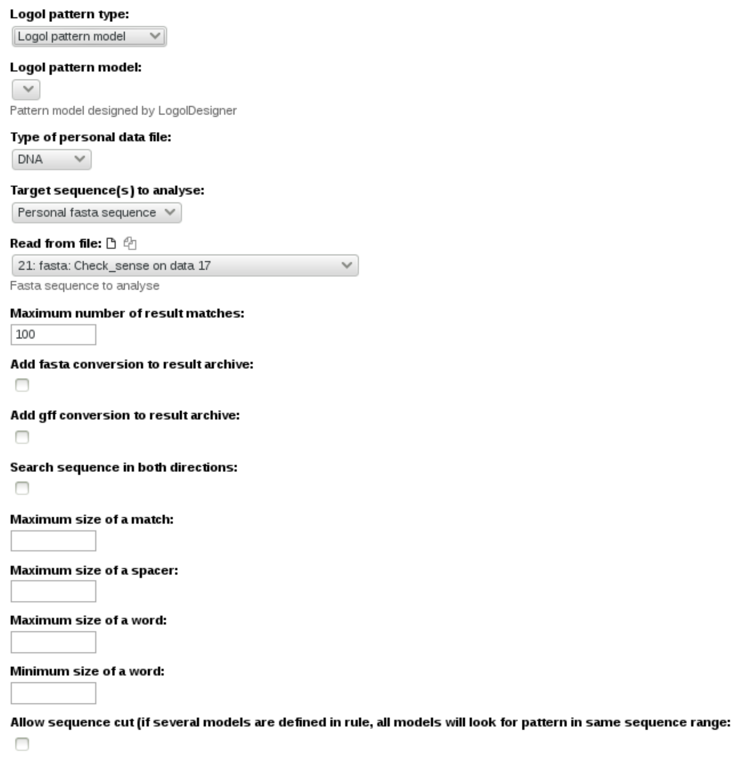


Syntaxes les plus importantes

```
<tool id="logol_wrapper" name="Logol">
    <description>Biological patterns matching</description>
    <command interpreter="bash">
            logol.sh $options.input.options_input_selector $option_input.input_model $type > $match
    </command>
    <inputs>
        <conditional name="options_input">
                <param name="options_input_selector" type="select" label="Logol pattern type">
                  <option value="model" selected="True">Logol pattern model</option>
                  <option value="grammar">Logol pattern grammar</option>
                </param>
                <when value="model">
                  <param name="input_model" type="data" format="lgd" label="Logol pattern model" help="Pattern model designed by LogolDesigner" />
                </when>
        </conditional>
        <param name="type" type="select" format="text" label="type">
            <option value="dna">DNA</option>
            <option value="rna">RNA</option>
            <option value="protein">PROTEIN</option>
        </param>
        <param name="max" type="integer" value="100" label="Maximum number of result matches"/>
        <param name="fasta" type="boolean" label="Add fasta conversion to result archive" checked="False" value="False"/>
    </inputs>
    <outputs>
          <data format="text" name="match" label="match : ${tool.name} on ${on_string}" />
    </outputs>
    <help>
    </help>
</tool>
```

Syntaxes les plus importantes

```
<tool id="logol_wrapper" name="Logol">
    <description>Biological patterns matching</description>
    <command interpreter="bash">
            logol.sh $options.input.options_input_selector $option_input.input_model $type > $match
    </command>
    <inputs>
        <conditional name="options_input">
                <param name="options_input_selector" type="select" label="Logol pattern type">
                  <option value="model" selected="True">Logol pattern model</option>
                  <option value="grammar">Logol pattern grammar</option>
                </param>
                <when value="model">
                  <param name="input_model" type="data" format="lgd" label="Logol pattern model" />
                </when>
        </conditional>
        <param name="type" type="select" format="text" label="type">
            <option value="dna">DNA</option>
            <option value="rna">RNA</option>
            <option value="protein">PROTEIN</option>
        </param>
        <param name="max" type="integer" value="100" label="Maximum number of result matches"/>
        <param name="fasta" type="boolean" label="Add fasta conversion to result archive" checked="False" value="False"/>
    </inputs>
    <outputs>
         <data format="text" name="match" label="match : ${tool.name} on ${on_string}" />
    </outputs>
    <help>
    </help>
</tool>
```



Logol pattern type:
Logol pattern model

Logol pattern model:

Pattern model designed by LogolDesigner

Type of personal data file:
DNA

Target sequence(s) to analyse:
Personal fasta sequence

Read from file:
21: fasta: Check_sense on data 17
Fasta sequence to analyse

Maximum number of result matches:
100

Add fasta conversion to result archive:

Add gff conversion to result archive:

Search sequence in both directions:

Maximum size of a match:

Maximum size of a spacer:

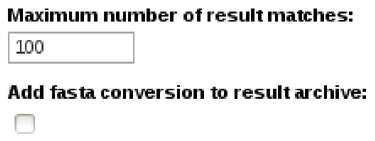
Maximum size of a word:

Minimum size of a word:

Allow sequence cut (if several models are defined in rule, all models will look for pattern in same sequence range:

Syntaxes les plus importantes

```
<tool id="logol_wrapper" name="Logol">
    <description>Biological patterns matching</description>
    <command interpreter="bash">
            logol.sh $options.input.options_input_selector $option_input.input_model $type > $match
    </command>
    <inputs>
        <conditional name="options_input">
                <param name="options_input_selector" type="select" label="Logol pattern type">
                  <option value="model" selected="True">Logol pattern model</option>
                  <option value="grammar">Logol pattern grammar</option>
                </param>
                <when value="model">
                  <param name="input_model" type="data" format="lgd" label="Logol pattern model" help="Pattern model designed by LogolDesigner" />
                </when>
        </conditional>
        <param name="type" type="select" format="text" label="type">
            <option value="dna">DNA</option>
            <option value="rna">RNA</option>
            <option value="protein">PROTEIN</option>
        </param>
        <param name="max" type="integer" value="100" label="Maximum number of result matches"/>
        <param name="fasta" type="boolean" label="Add fasta conversion to result archive" checked="False" value="False"/>
    </inputs>
    <outputs>
          <data format="text" name="match" label="match : ${tool.name} on ${on_string}" />
    </outputs>
    <help>
    </help>
</tool>
```


Maximum number of result matches:

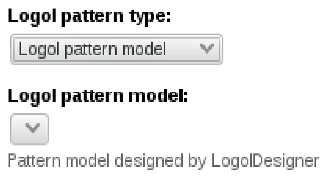
100

Add fasta conversion to result archive:

☐

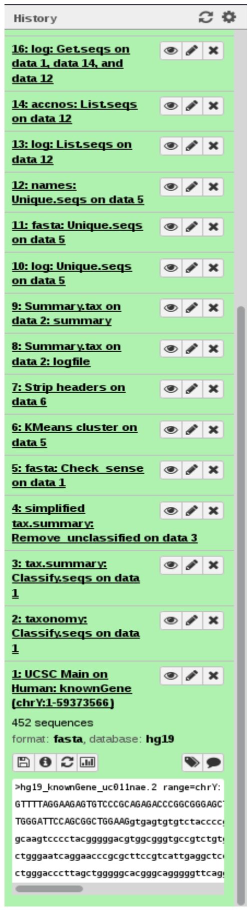
Syntaxes les plus importantes

```
<tool id="logol_wrapper" name="Logol">
    <description>Biological patterns matching</description>
    <command interpreter="bash">
            logol.sh $options.input.options_input_selector $option_input.input_model $type > $match
    </command>
    <inputs>
        <conditional name="options_input">
                <param name="options_input_selector" type="select" label="Logol pattern type">
                    <option value="model" selected="True">Logol pattern model</option>
                    <option value="grammar">Logol pattern grammar</option>
                </param>
                <when value="model">
                    <param name="input_model" type="data" format="lgd" label="Logol pattern model" help="Pattern model designed by LogolDesigner" />
                </when>
        </conditional>
        <param name="type" type="select" format="text" label="type">
            <option value="dna">DNA</option>
            <option value="rna">RNA</option>
            <option value="protein">PROTEIN</option>
        </param>
        <param name="max" type="integer" value="100" label="Maximum number of result matches"/>
        <param name="fasta" type="boolean" label="Add fasta conversion to result archive" checked="False" value="False"/>
    </inputs>
    <outputs>
        <data format="text" name="match" label="match : ${tool.name} on ${on_string}" />
    </outputs>
    <help>
    </help>
</tool>
```



Syntaxes les plus importantes

```
<tool id="logol_wrapper" name="Logol">
    <description>Biological patterns matching</description>
    <command interpreter="bash">
            logol.sh $options.input.options_input_selector $option_input.input_model $type > $match
    </command>
    <inputs>
        <conditional name="options_input">
                <param name="options_input_selector" type="select" label="Logol pattern type">
                  <option value="model" selected="True">Logol pattern model</option>
                  <option value="grammar">Logol pattern grammar</option>
                </param>
                <when value="model">
                  <param name="input_model" type="data" format="lgd" label="Logol pattern model" help="Pattern model designed by LogolDesigner" />
                </when>
        </conditional>
        <param name="type" type="select" format="text" label="type">
            <option value="dna">DNA</option>
            <option value="rna">RNA</option>
            <option value="protein">PROTEIN</option>
        </param>
        <param name="max" type="integer" value="100" label="Maximum number of result matches"/>
        <param name="fasta" type="boolean" label="Add fasta conversion to result archive" checked="False" value="False"/>
    </inputs>
    <outputs>
        <data format="text" name="match" label="match : ${tool.name} on ${on_string}" />
    </outputs>
    <help>
    </help>
</tool>
```



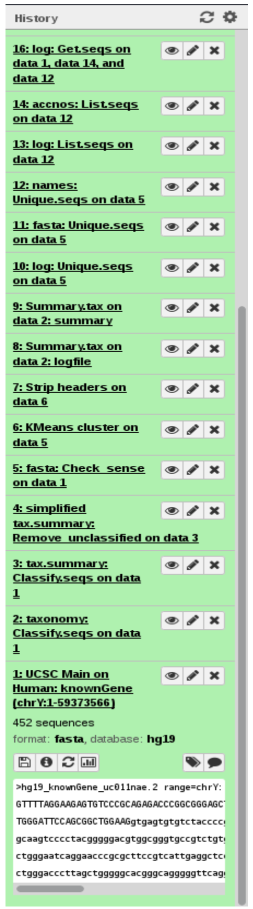
16: log: Get.seqs on data 1, data 14, and data 12
14: accnos: List.seqs on data 12
13: log: List.seqs on data 12
12: names: Unique.seqs on data 5
11: fasta: Unique.seqs on data 5
10: log: Unique.seqs on data 5
9: Summary.tax on data 2: summary
8: Summary.tax on data 2: logfile
7: Strip headers on data 6
6: KMeans cluster on data 5
5: fasta: Check_sense on data 1
4: simplified tax.summary: Remove_unclassified on data 3
3: tax.summary: Classify.seqs on data 1
2: taxonomy: Classify.seqs on data 1
1: UCSC Main on Human: knownGene (chrY:1-59373566)

452 sequences
format: **fasta**, database: **hg19**

```
>hg19_knownGene_uc011nae.2 range=chrY:
GTTTTAGGAAGAGTGTCCCGCAGAGACCCGGCGGGAGC
TGGGATTCCAGCGGCTGGAAGgtgagtgtgtctacccc
gcaagtcccctacgggggacgtggcgggtgccgtctgt
ctgggaatcaggaacccgcgcttccgtcattgaggctc
ctgggacccttagctgggggcacgggcagggggttcag
```

Syntaxes les plus importantes

```
<tool id="logol_wrapper" name="Logol">
    <description>Biological patterns matching</description>
    <command interpreter="bash">
            logol.sh $options.input.options_input_selector $option_input.input_model $type > $match
    </command>
    <inputs>
        <conditional name="options_input">
                <param name="options_input_selector" type="select" label="Logol pattern type">
                    <option value="model" selected="True">Logol pattern model</option>
                    <option value="grammar">Logol pattern grammar</option>
                </param>
                <when value="model">
                    <param name="input_model" type="data" format="lgd" label="Logol pattern model" help="Pattern model designed by LogolDesigner" />
                </when>
        </conditional>
        <param name="type" type="select" format="text" label="type">
            <option value="dna">DNA</option>
            <option value="rna">RNA</option>
            <option value="protein">PROTEIN</option>
        </param>
        <param name="max" type="integer" value="100" label="Maximum number of result matches"/>
        <param name="fasta" type="boolean" label="Add fasta conversion to result archive" checked="False" value="False"/>
    </inputs>
    <outputs>
        <data format="text" name="match" label="match : ${tool.name} on ${on_string}" />
    </outputs>
    <help>
    </help>
</tool>
```



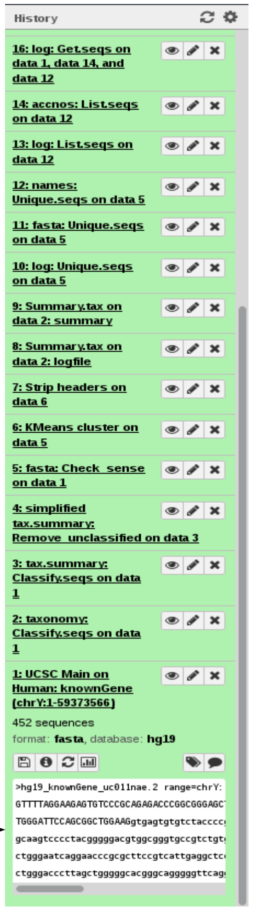
16: log: Get.seqs on data 1, data 14, and data 12
14: accnos: List.seqs on data 12
13: log: List.seqs on data 12
12: names: Unique.seqs on data 5
11: fasta: Unique.seqs on data 5
10: log: Unique.seqs on data 5
9: Summary.tax on data 2: summary
8: Summary.tax on data 2: logfile
7: Strip headers on data 6
6: KMeans cluster on data 5
5: fasta: Check_sense on data 1
4: simplified tax.summary: Remove_unclassified on data 3
3: tax.summary: Classify.seqs on data 1
2: taxonomy: Classify.seqs on data 1
1: UCSC Main on Human: knownGene (chrY:1-59373566)

452 sequences
format: **fasta**, database: **hg19**

```
>hg19_knownGene_uc011nae.2 range=chrY:
GTTTTAGGAAGAGTGTCCCGCAGAGACCCGGCGGGAGC
TGGGATTCCAGCGGCTGGAAGgtgagtgtgtctacccc
gcaagtcccctacgggggacgtggcgggtgccgtctgt
ctgggaatcaggaacccgcgcttccgtcattgaggctc
ctgggacccttagctgggggcacgggcagggggttcag
```

Syntaxes les plus importantes

```
<tool id="logol_wrapper" name="Logol">
    <description>Biological patterns matching</description>
    <command interpreter="bash">
            logol.sh $options.input.options_input_selector $option_input.input_model $type > $match
    </command>
    <inputs>
        <conditional name="options_input">
                <param name="options_input_selector" type="select" label="Logol pattern type">
                    <option value="model" selected="True">Logol pattern model</option>
                    <option value="grammar">Logol pattern grammar</option>
                </param>
                <when value="model">
                    <param name="input_model" type="data" format="lgd" label="Logol pattern model" help="Pattern model designed by LogolDesigner" />
                </when>
        </conditional>
        <param name="type" type="select" format="text" label="type">
            <option value="dna">DNA</option>
            <option value="rna">RNA</option>
            <option value="protein">PROTEIN</option>
        </param>
        <param name="max" type="integer" value="100" label="Maximum number of result matches"/>
        <param name="fasta" type="boolean" label="Add fasta conversion to result archive" checked="False" value="False"/>
    </inputs>
    <outputs>
        <data format="text" name="match" label="match : ${tool.name} on ${on_string}" />
    </outputs>
    <help>
    </help>
</tool>
```



History

16: log: Get.seqs on data 1, data 14, and data 12

14: accnos: List.seqs on data 12

13: log: List.seqs on data 12

12: names: Unique.seqs on data 5

11: fasta: Unique.seqs on data 5

10: log: Unique.seqs on data 5

9: Summary.tax on data 2: summary

8: Summary.tax on data 2: logfile

7: Strip headers on data 6

6: KMeans cluster on data 5

5: fasta: Check_sense on data 1

4: simplified tax.summary: Remove_unclassified on data 3

3: tax.summary: Classify.seqs on data 1

2: taxonomy: Classify.seqs on data 1

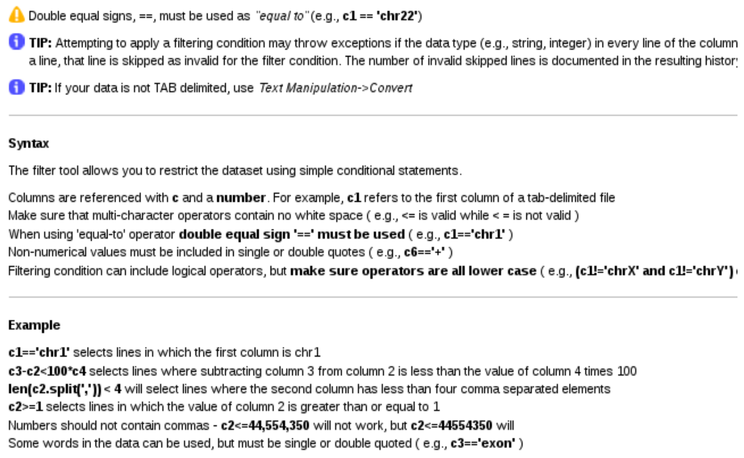
1: UCSC Main on Human: knownGene (chrY:1-59373566)

452 sequences

format: **fasta**, database: **hg19**

```
>hg19_knownGene_uc011nae.2 range=chrY:
GTTTTAGGAAGAGTGTCCCGCAGAGACCCGGCGGGAGC
TGGGATTCCAGCGGCTGGAAGgtgagtgtgtctacccc
gcaagtcccctacgggggacgtggcgggtgccgtctgt
ctgggaatcaggaacccgcgcttccgtcattgaggctc
ctgggacccttagctgggggcacgggcagggggttcag
```

Syntaxes les plus importantes



⚠ Double equal signs, ==, must be used as *"equal to"* (e.g., **c1 == 'chr22'**)

TIP: Attempting to apply a filtering condition may throw exceptions if the data type (e.g., string, integer) in every line of the column
a line, that line is skipped as invalid for the filter condition. The number of invalid skipped lines is documented in the resulting histor

TIP: If your data is not TAB delimited, use *Text Manipulation->Convert*

Syntax

The filter tool allows you to restrict the dataset using simple conditional statements.

Columns are referenced with **c** and a **number**. For example, **c1** refers to the first column of a tab-delimited file
Make sure that multi-character operators contain no white space (e.g., <= is valid while < = is not valid)
When using 'equal-to' operator **double equal sign '==' must be used** (e.g., **c1=='chr1'**)
Non-numerical values must be included in single or double quotes (e.g., **c6=='+'**)
Filtering condition can include logical operators, but **make sure operators are all lower case** (e.g., **(c1!='chrX' and c1!='chrY')**

Example

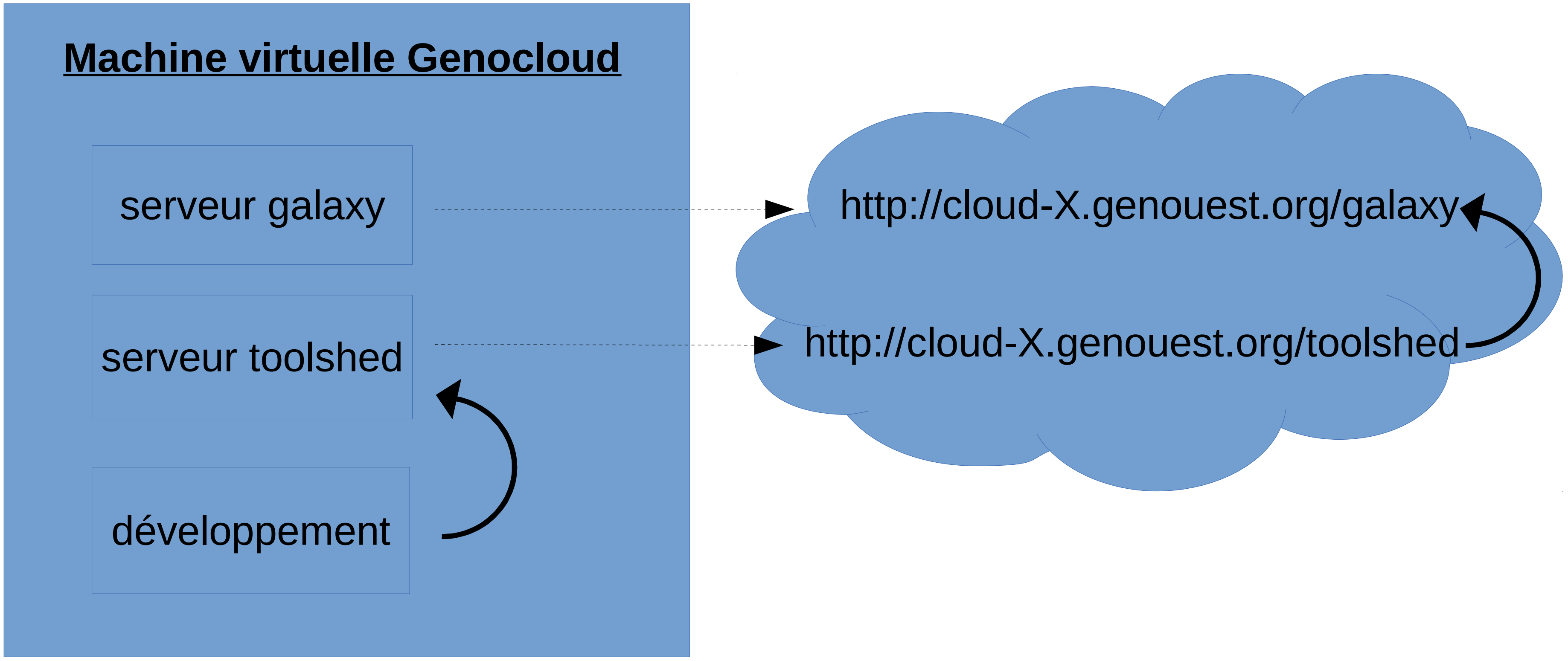
c1=='chr1' selects lines in which the first column is chr1
c3-c2<100*c4 selects lines where subtracting column 3 from column 2 is less than the value of column 4 times 100
len(c2.split(',')) < **4** will select lines where the second column has less than four comma separated elements
c2>=1 selects lines in which the value of column 2 is greater than or equal to 1
Numbers should not contain commas - **c2<=44,554,350** will not work, but **c2<=44554350** will
Some words in the data can be used, but must be single or double quoted (e.g., **c3=='exon'**)

```
<tool id="logol_wrapper" name="Logol">
    <description>Biological patterns matching</description>
    <command interpreter="bash">
            logol.sh $options.input.options_input_selector $option_input.input_model $type > $match
    </command>
    <inputs>
        <conditional name="options_input">
                <param name="options_input_selector" type="select" label="Logol pattern type">
                    <option value="model" selected="True">Logol pattern model</option>
                    <option value="grammar">Logol pattern grammar</option>
                </param>
                <when value="model">
                    <param name="input_model" type="data" format="lgd" label="Logol pattern model" help="Pattern model designed by LogolDesigner" />
                </when>
        </conditional>
        <param name="type" type="select" format="text" label="type">
            <option value="dna">DNA</option>
            <option value="rna">RNA</option>
            <option value="protein">PROTEIN</option>
        </param>
        <param name="max" type="integer" value="100" label="Maximum number of result matches"/>
        <param name="fasta" type="boolean" label="Add fasta conversion to result archive" checked="False" value="False"/>
    </inputs>
    <outputs>
            <data format="text" name="match" label="match : ${tool.name} on ${on_string}" />
    </outputs>
    <help>
    </help>
</tool>
```

TP1. Préparation de l'environnement

Genocloud

Environnement

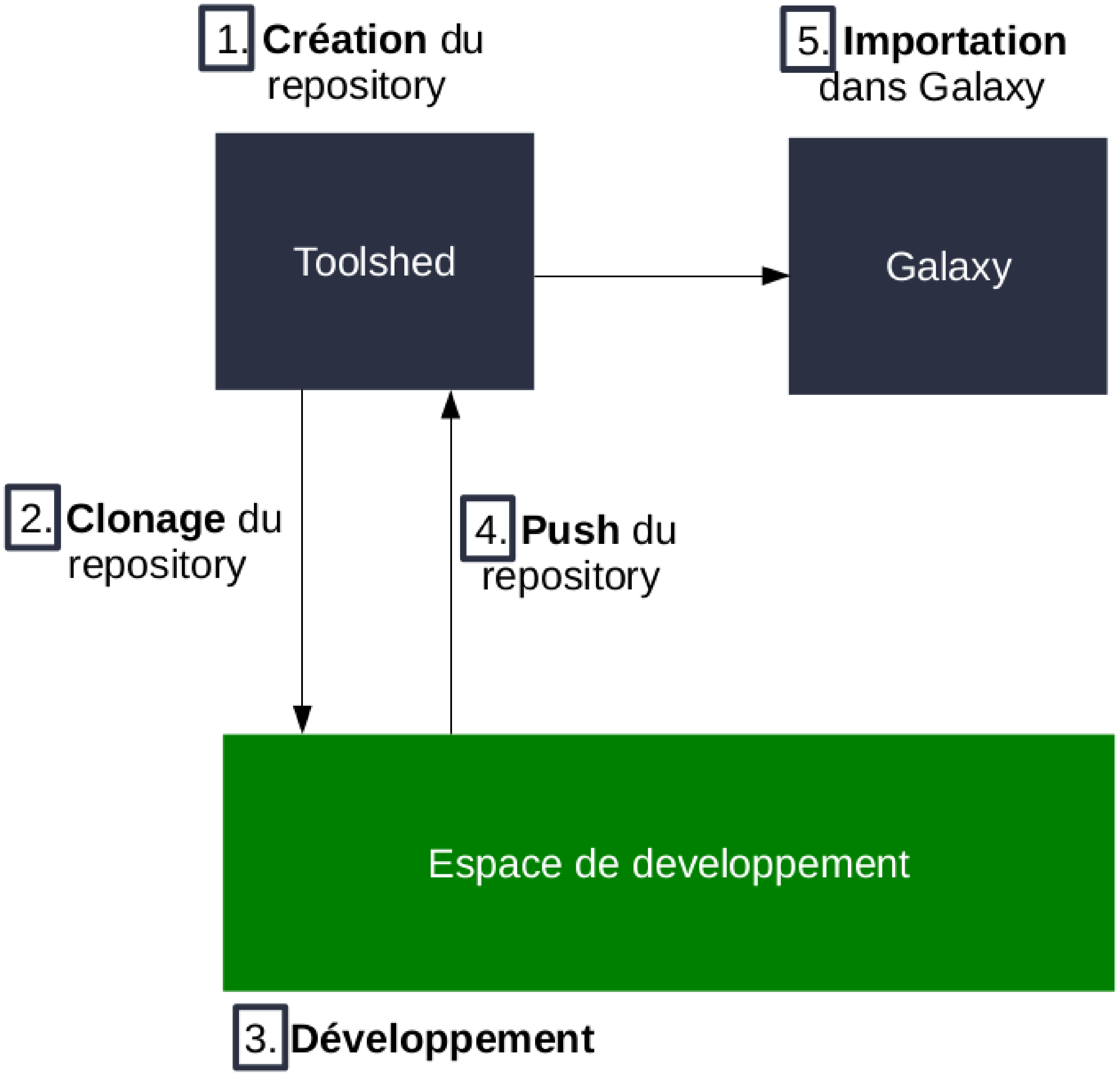

Machine virtuelle Genocloud
serveur galaxy
serveur toolshed
développement
http://cloud-X.genouest.org/galaxy
http://cloud-X.genouest.org/toolshed

Installation de l'environnement

cf. document PDF

TP2. Les différentes étapes pour l'intégration

Toolshed & première intégration d'un outil : takeabreak


1. **Création** du repository
5. **Importation** dans Galaxy
Toolshed
Galaxy
2. **Clonage** du repository
4. **Push** du repository
Espace de developpement
3. **Développement**

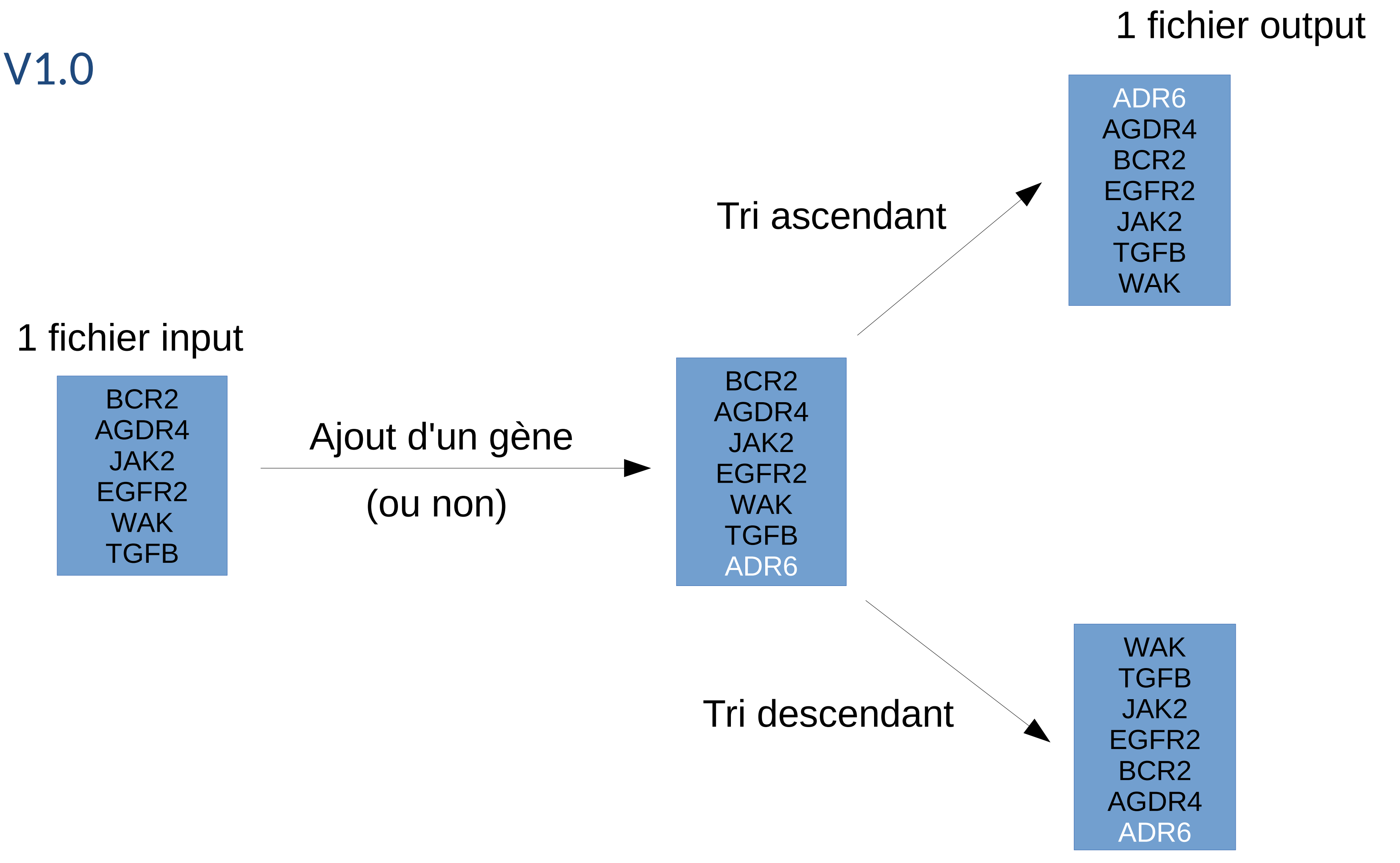
Intégration de TakeABreak

cf. document PDF

TP3. Intégration d'un outil simple

Tri d'une liste de gènes

V1.0



1 fichier output

Difficultés : aucune

TP4. Intégration d'un outil avec plusieurs entrées

Tri d'une liste de gènes issus de plusieurs fichiers

Utilisation du multi-input files

- <param name="gene_files_list" type="data" **multiple="true"** format="txt" label="Gene list file" />

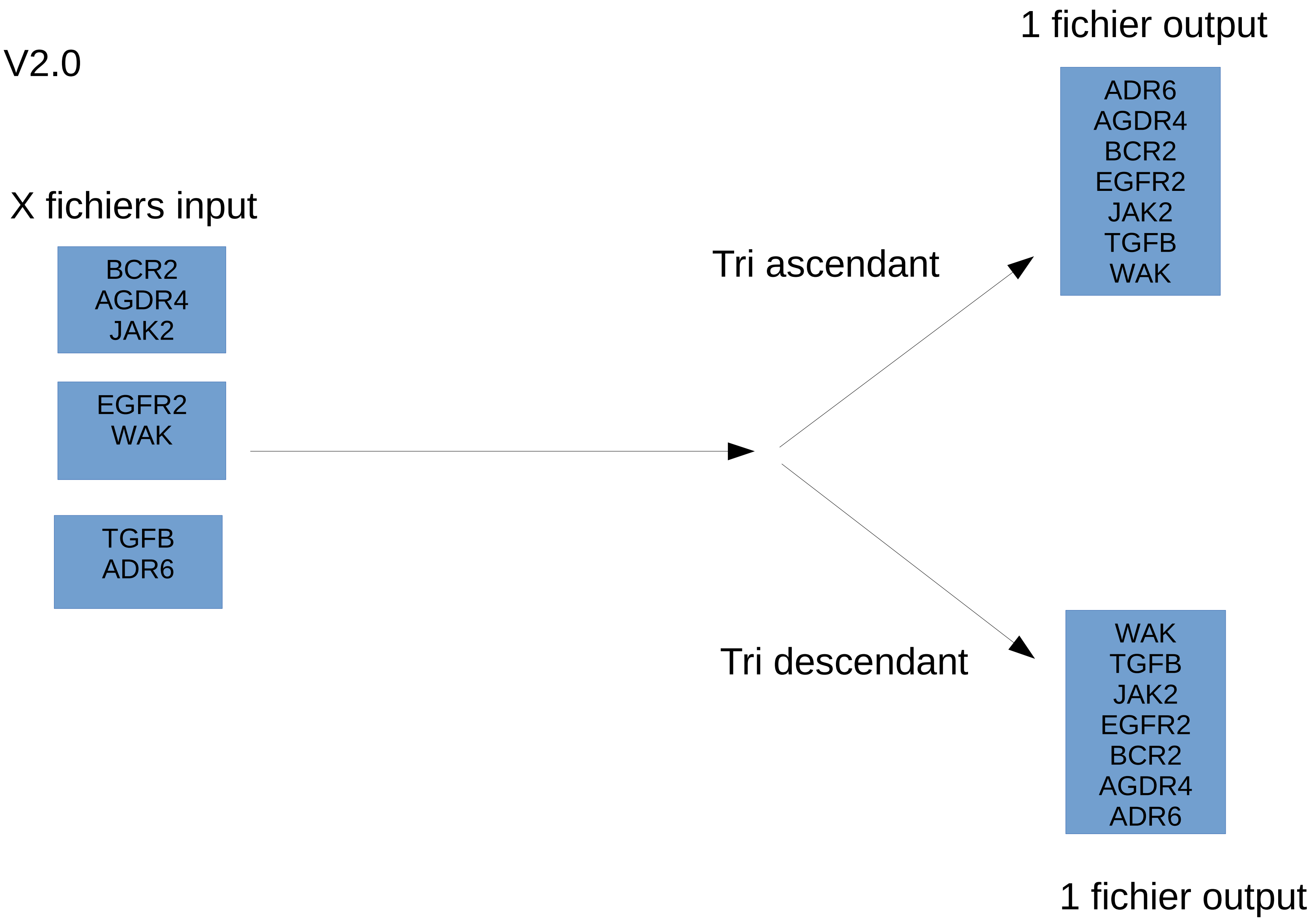
  - Passage directement de $gene_files_list dans la ligne de commande
  - Dans le wrapper, récupération d'une chaine de caractère du type :
    - /opt/galaxy-dist/database/files/000/dataset_001.dat**,**/opt/galaxy-dist/database/files/000/dataset_002.dat

- <param name="gene_files_list" type="data" **multiple="true"** format="txt" label="Gene list file" />
- <configfiles>

```
  <configfile name="gene_files_list_config" >
  #for $file in $gene_files_list:
    ${file}::${file.display_name}
  #end for
  </configfile>
</configfiles>
```

- Passage de $gene_files_list_config dans la ligne de commande
- Dans le wrapper, récupération d'un fichier (chemin) à traiter
  - Contenu du fichier :
    - /opt/galaxy-dist/database/files/000/dataset_001.dat**::nom1**
    - /opt/galaxy-dist/database/files/000/dataset_002.dat**::nom2**

V2.0



Difficultés : multi-input

TP5. Intégration d'un outil avec plusieurs sorties

Tri d'une liste de gènes issus de plusieurs fichiers et limitation d'un nombre de gène par fichier de sortie

Utilisation du multi-output files avec nombre de fichiers inconnus

- <tool id=”monid” name=”myname” force_history_refresh=”true” >
- <outputs>

  <data format="txt" name="output" label=" log.txt" />

  </outputs>

- --output $output

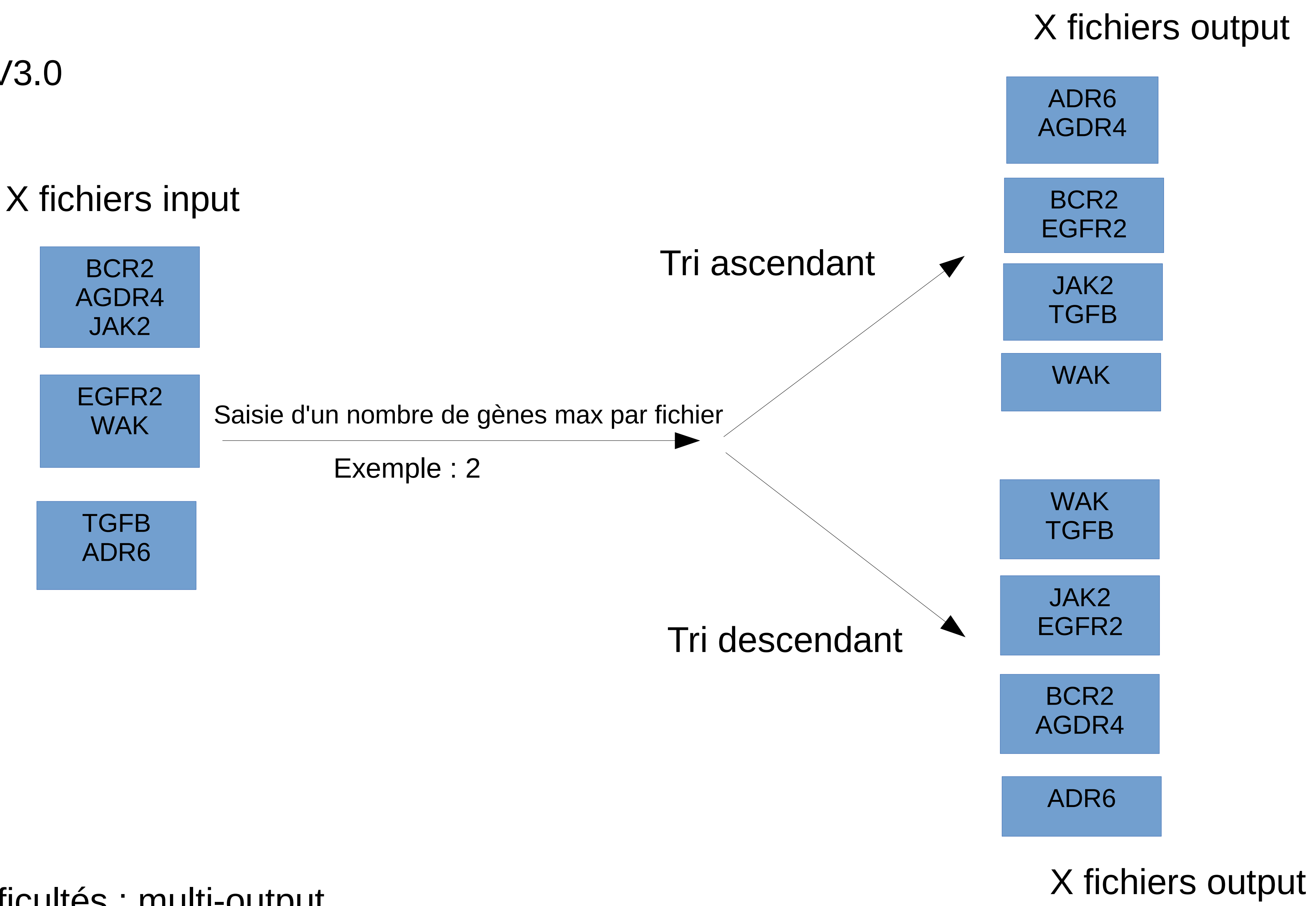
  --output_id $output.id

  --new_file_path $__new_file_path__

- $__new_file_path__ correspond à un répertoire temporaire définit dans universe_wsgi.ini

- Dans le wrapper, les fichiers de sorties doivent être envoyé dans ce répertoire
  - *new_file_path*/**primary**_*output.id_name*_**visible**_*txt*

V3.0



Difficultés : multi-output

TP6. Intégration de dépendances

Exemple de takeabreak

Les dependencies : étape 1/3

- Intégration d'un package spécial dans le toolshed
  - type = tool_dependencies

- Le repository nommé « commet_dependencies » contient uniquement **un fichier tool_dependencies.xml** définissant un package
- Dans ce fichier s'applique une succession d' «actions» :

```
<?xml version="1.0"?>
<tool_dependency>
    <package name="galaxy_commet" version="24.7.14">
        <install version="1.0">
        <actions>
                <action type="download_by_url">http://github.com/pierrepeterlongo/commet/archive/master.zip</action>
                <action type="shell_command">make</action>
               <!-- move directories into $INSTALL_DIR -->
              <action type="move_directory_files">
                    <source_directory>bin</source_directory>
                    <destination_directory>$INSTALL_DIR/bin</destination_directory>
             </action>
             <!-- move files into $INSTALL_DIR -->
             <action type="move_file">
                     <source>dendro.R</source>
                     <destination>$INSTALL_DIR/Rscript</destination>
             </action>
             <!-- create env.sh -->
             <action type="set_environment">
                   <environment_variable name="PATH" action="prepend_to">$INSTALL_DIR/bin</environment_variable>
                   <environment_variable name="RSCRIPTS" action="set_to">$INSTALL_DIR/Rscript/</environment_variable>
         </action>
        </actions>
        </install>
     </package>
</tool_dependency>
```

$\downarrow$

Création d'un fichier env.sh embarqué par l'outil lors de l'exécution

Les dependencies : étape 2/3

- Au sein du repository contenant l'outil Galaxy développé doit s'ajouter un fichier **tool_dependencies.xml**

```
<?xml version="1.0"?>
<tool_dependency>
<package name="galaxy_commet" version="24.7.14">
        <repository toolshed="http://toolshed.genouest.org" name="commet_dependencies"
owner="cmonjeau" prior_installation_required="True" changeset_revision="96f67cab9b21"/>
</package>
</tool_dependency>
```

- Installation du package lors de l'installation de l'outil

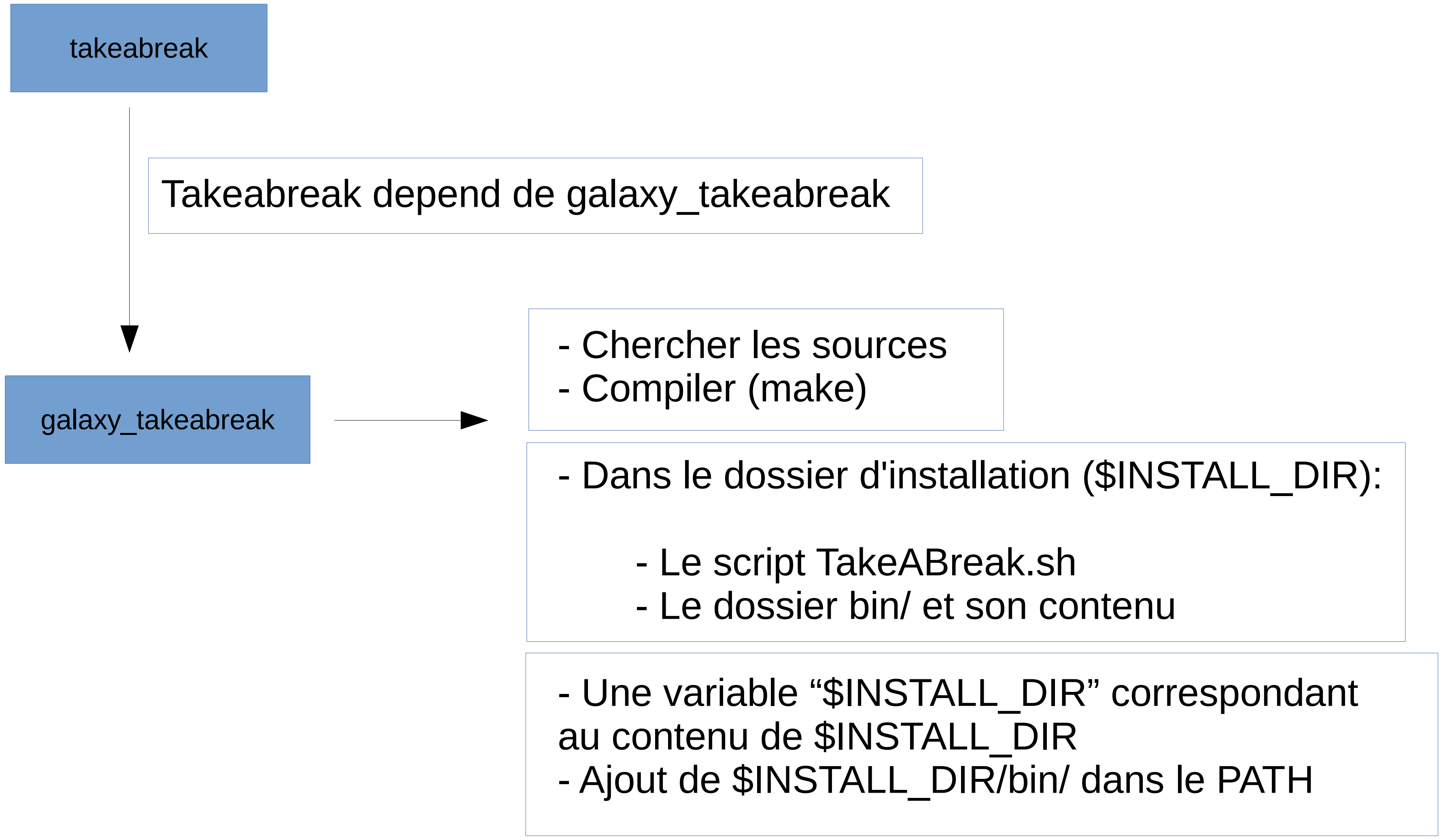
Les dependencies : étape 3/3

- Au sein du descripteur de l'outil :

```
<requirements>
    <requirement type="package" version="24.7.14">galaxy_commet</requirement>
</requirements>
```

- Nécessaire pour l'embarquement automatique de l'environnement

Intégration de la dependance de TakeABreak



takeabreak

Takeabreak depend de galaxy_takeabreak

galaxy_takeabreak

- Chercher les sources
- Compiler (make)

- Dans le dossier d'installation ($INSTALL_DIR):

    - Le script TakeABreak.sh
    - Le dossier bin/ et son contenu

- Une variable “$INSTALL_DIR” correspondant au contenu de $INSTALL_DIR
- Ajout de $INSTALL_DIR/bin/ dans le PATH