# CGA XML Description

This document describes the XML files used to describe the flow of a pipeline.

# Pipeline XML Description

Each pipeline will be controlled by an XML description file. The outer tag is <pipeline>.

## pipeline

<pipeline name="...">  
 <input />  
 <output />  
 <tempfile />  
 <process />  
</pipeline>

A <pipeline> tag contains one or more <input> tags and one or more <process> tags.

## input

The <input> tag specifies a file that is to be used in the pipeline. The filespec (complete file path) may be hardcoded in the tag, or refer to a positional command line parameter.

<input id="..." type=”...” filespec="..." />  
<input id="..." type=”...” parameter="..." />

If the filespec form is used, then that specific file is used as the input to the pipeline. This would be used, for instance, for the path to a reference genome.

If the parameter form is used, the value of the attribute is the positional argument's number (1-based) from the command line invoking the pipeline.

***Note:*** *we need to revisit this when we have nailed down the form of the invoking command, for instance, how we specify the XML file describing the pipeline.*

The type attribute specifies the type of data that is required, e.g. fastq, sam, bam, vcf, gtf, fa, etc.

## output

The <output> tag is similar to the <input> tag. In particular, the two tags share an ID space, such that the files can be passed from process to process.

<output id="..." type="..." filespec="..." />  
<output id="..." type="..." parameter="..." />

## tempfile

The <tempfile> tag specifies an id for a file of indeterminate name which is to be discarded at the end of the run. If specified in the lexical scope of a <process> tag, the file may be deleted at the end of the process, before the overall pipeline has completed.  
  
<tempfile id=”...” />

## outputdir

The <outputdir> tag specifies the default output directory for output files specified with a relative path. It is unspecified whether output tempfiles are created in this directory.

<outputdir id=”...” filespec=”...” />  
<outputdir id="..." parameter="..." />

## process

A process is a logical phase of the pipeline. A <process> tag contains one or more each of inputs and tools, and zero or more outputs (a process might be run that only has side-effects, with no output files generated).

<process name="...">  
 <input type="..." id="..." />  
 <output type="..." id="..." />  
 <tool />  
</process>

***Note:*** *Input and output tags here have different attributes than in the* <pipeline> *(they don’t specify the filename nor parameter). Should they be different tags, or is it OK to be context sensitive?*

A process has fixed input and output file types. This allows a pipeline to chain processes together. Tools may have differing input and output file types. If a replacement process is designed using different tools, the new process must maintain the “contract” by converting the files as necessary. The input and output types must be the same across all instances of a particular process. This means that the attribute could be removed, and simply specified in an architecture document, but it is retained for self-documentation purposes. A pipeline implementation may check that the output file type of one process matches the input file type of the following process.

## tool

A tool takes one or more inputs, and creates one or more outputs. For instance, it could be an aligner, or a simple sam to bam converter.

<tool name="..."

input="id [, id]..."

output="id [, id]..."

description="..." />

The description attribute specifies the basename of another XML file which describes the tool and its parameters. The driver will search for the XML file in a set of TBD directories, allowing per-user customization, followed by a system-wide configuration. I am also considering searching for an environment variable <name>.DESCRIPTION, where <name> is the value of the name attribute. Description searching will be described in a separate spec.

The input and output attributes are used to map from the pipeline file id space to a tool file id space. In this way, a tool description file can be utilized in multiple pipelines or processes without having to have a global id assignment registry.

The ids listed in the input or output attributes are in the pipeline /process id space. The order in the list determines the id they will have in the tool’s description file. The first id in the list in this tag will receive the id “in\_1” or “out\_1”, as appropriate, in the tool description XML file.

# Tool XML Description

A tool XML description is separate from a pipeline description. This allows utilizing a tool description in multiple pipelines, and also conveniently tweaking a tool’s configuration without editing a whole pipeline file.

The outer tag is <tool>.

## tool

<tool name=”...” tool\_config\_prefix=”...” threads=”...”>  
 <description />  
 <option />  
 <command />  
</tool>

The tool\_config\_prefix attribute is optional. The threads attribute is optional. The <description> tag is optional. There may be zero or more <option> tags. There must be one or more <command> tags. If multiple <command> tags are specified, they are executed serially.

See the <option> tag for a description of the tool\_config\_prefix attribute.

The threads attribute specifies the maximum number of threads any command in this tool will use. If not specified, the tool will be run in one thread. If executing in a cluster environment, it specifies the number of processors that will be allocated. If the commands in a tool run for a significant length of time and use widely different numbers of threads, consider splitting it into multiple tools within one process, so that each tool can specify the number of threads appropriate to its command(s).

When executing in a cluster environment, a tool will be scheduled as a job.

## description

The <description> tag contains free form information about the tool.

<description> ... </description>

## option

The information in the <option> tag is used in the <command> tag.

<option name="..." command\_text=”...” value="..." id="...">

The <option> tag must contain a name attribute. The command\_text, value and id attributes are optional, but at least one must be specified. See the description of the <command> tag for how these are used.

For each option specified, if the tool\_config\_prefix attribute is specified in the <tool> tag, option processing will search for an configuration file in a manner TBD. If the configuration file exists and the option’s name is listed, the value(s) from the configuration file will be used instead of the values specified in this tag. A full description of configuration processing is in a separate document TBD.

## command

The <command> tag specifies how to construct the command line that will be executed is constructed.

<command  
 delimiters=”...”  
 stdout\_id=”...”  
 stderr\_id=”...”> ... </command>

The text within this tag consists of literal text that will be inserted in the command line, interspersed with option names enclosed within braces ({}). Option names in this context means the values of the name attributes in the <option> tags.

If the optional delimiters attribute is specified, its value is a two character string. The first character is used in place of open brace ({) to indicate the start of an option name, and the second character is used in place of close brace (}). If a literal brace is required in the command line, use the delimiters attribute to specify an alternate pair of characters.

The optional stdout\_id and stderr\_id attributes allow IO redirection.

When a brace-enclosed option name is encountered, the value of the command\_text attribute will be inserted, if present. Then the value of the value attribute if specified, and finally the filename represented by the id attribute, if specified. But see the specification of tool configuration files for processing that overrides the values specified in the XML file.

All of the text within the <command> </command> tags will be reconstructed as a single line, with any line breaks treated as spaces and all spaces collapsed.

## tempfile

The <tempfile> tag specifies an id for a file of indeterminate name which is to be discarded at the end of the tool invocation.

<tempfile id="..." />