Jobs/Apps/Systems Parameter Matrix

Cloud and Interactive Computing

Exported on 04/28/2021

Table of Contents

1 Common attributes for Jobs, Applications and Systems

Attribute	Job	App	Sys	Notes
System Definition Values[§]				
host			x	the host associated with this system
canExec			x	boolean flag indicating if system can be used to execute jobs.
jobWorkingDir			x	See Directory Semantics(see page 7); TACC systems will often assign HOST_EVAL($SCRATCH)
jobExecClass			x	One of: NORMAL, PROTECTED
jobContainerRuntimes			x	Zero or more: Docker, Singularity, etc.
jobEnvVariables	x	x	x	environment variables passed into application containers
jobMaxJobs		x	x	Apps can lower the limit set by Systems
jobMaxJobsPerUser		x	x	Apps can lower the limit set by Systems
jobIsBatch			x	boolean flag indicating if system uses a batch scheduler to run jobs. By default this is *false*.
batchScheduler			x	slurm, pbs, condor, etc.
batchLogicalQueues			x	Tapis-defined queues
batchDefaultLogicalQueue			x	Default tapis-defined queue
batchHPCQueue			x	Name of actual queue on HPC system
jobCapabilities			x	parameters.execSystemConstraints field in SubmitJobRequest[1]
dtnSystemId			x	An alternate system to use as a Data Transfer Node (DTN).

1 https://github.com/tapis-project/tapis-java/blob/dev/tapis-jobsapi/src/main/resources/edu/utexas/tacc/tapis/jobs/api/jsonschema/SubmitJobRequest.json

Attribute	Job	App	Sys	Notes
dtnMountPoint			x	mount point on the execution system (maps dtnMountSourcePath from DTN system)
dtnMountSourcePath			x	the relative path that defines the DTN source directory relative to the DTN rootDir.
isDtn			x	boolean flag indicating if system will be used as a DTN.
Job Request Values[§]				
job name	x			User supplied job name
job description	x	x		Long description
archiveOnAppError	x	x		default = true
dynamicExecSystem	x	x		default = false, use constraints to dynamically choose a best fit execution system
execSystemId	x	x		required
execSystemExecDir	x	x		See Directory Semantics(see page 7)
execSystemInputDir	x	x		See Directory Semantics(see page 7)
execSystemOutputDir	x	x		See Directory Semantics(see page 7)
execSystemLogicalQueue	x	x	*	Defaults to system's *batchDefaultLogicalQueue*
archiveSystemId	x	x		Set to enable archiving; *archiveSystemDir* must also be set.
archiveSystemDir	x	x		See Directory Semantics(see page 7)
nodeCount	x	x		
coresPerNode	x	x		
memoryMB	x	x		

Attribute	Job	App	Sys	Notes
maxMinutes	x	x		
fileInputs	x	x		See SubmitJobRequest[2]
parameterSet	x	x		See SubmitJobRequest[3]
execSystemConstraints	x	x		See SubmitJobRequest[4], (array of sql strings)
subscriptions	x	x		See SubmitJobRequest[5] (array of notification subscriptions)
tags	x	x		list of strings

§ All App fields that can be overridden in a Job request are embedded in a jobAttributes object in App's JSON schema and Java model.

In general, the values supplied in the job submission request take priority over the same value supplied in the application, which has priority over values supplied in the system.

The Apps service defines a ***jobAttributes*** object in its App model with two different manifestations. First, it's defined in the JSON schema that generates the App objects returned by various AppsClient methods. Second, it defined in the App server's Java model and returned by various App endpoints. The jobAttributes object encapsulates all job parameters that are set in an application but can be overridden in a Job submission request. The fields in jobAttribute objects have the same names and types as those used in Job submission requests. The goal is to share as much as possible JSON schema and Java code between the two services.

The ***fileInputs*** field is an array of JSON objects as defined in the SubmitJobRequest[6] schema. The schema also defines ***parameterSet*** as a JSON object that contains arrays of other JSON objects. ***execSystemConstraints*** filters prospective execution systems based on their capabilities. When these fields are defined in an application, they are merged into the definitions in the job submission request. The merge performs a logical AND of the sql-like expression from the application definition with that of the job request.

1.1 Macro Substitution

One way to reduce the complexity of running Tapis jobs is to simplify the job submission request. By specifying default values in the application and/or system definitions, users are often relieved from supplying those values on job submission. To increase flexibility, Jobs defines a set macros[7] that can be used as placeholders for values that

2 https://github.com/tapis-project/tapis-java/blob/dev/tapis-jobsapi/src/main/resources/edu/utexas/tacc/tapis/jobs/api/jsonschema/SubmitJobRequest.json
3 https://github.com/tapis-project/tapis-java/blob/dev/tapis-jobsapi/src/main/resources/edu/utexas/tacc/tapis/jobs/api/jsonschema/SubmitJobRequest.json
4 https://github.com/tapis-project/tapis-java/blob/dev/tapis-jobsapi/src/main/resources/edu/utexas/tacc/tapis/jobs/api/jsonschema/SubmitJobRequest.json
5 https://github.com/tapis-project/tapis-java/blob/dev/tapis-jobsapi/src/main/resources/edu/utexas/tacc/tapis/jobs/api/jsonschema/SubmitJobRequest.json
6 https://github.com/tapis-project/tapis-java/blob/dev/tapis-jobsapi/src/main/resources/edu/utexas/tacc/tapis/jobs/api/jsonschema/SubmitJobRequest.json
7 https://confluence.tacc.utexas.edu/display/CIC/Job+User+Macros

get substituted at job submission time. For example, in the following string the macro ${JOB_OWNER} will be replaced with the actual user submitting a job:

> MySequencer is being run by *${JOB_OWNER}* to determine the species of a DNA sample.

If the above string was assigned to the *job description* field in the MySequencer application definition, then users that ran MySequencer would automatically inherit that job description from the application definition. A user could override the application's default job description by supplying their own in the job request.

2 Directory Semantics

2.1 System Directories

The required *jobWorkingDir* value in a system definition specifies the root directory where most I/O occurs during job execution. The following syntax can be used:

1. **/{path}** - the path relative to *rootDir* in the System definition. An empty {path} designates the system's *rootDir* itself. Macro[8] substitution is applied to calculate the path.

2. **HOST_EVAL($SOME_ENV_VARIABLE)** - the value of the specified environment variable when evaluated on the host under the job's effective user ID. This is a dynamic value determined at job submission time. This function extracts specific environment variable values for use during job setup. In particular, the TACC-specific values of $HOME, $WORK, $SCRATCH and $FLASH can be referenced.

   The specified environment variable name is used *as-is:* it is not subject to macro substitution. However, the function call can have a path string appended to it, such as in HOST_EVAL($SCRATCH)/my/dir, and macro substitution is applied to the path string.

2.2 Application and Job Directories

The following directories can be set in an Application definition or in a Job submission request. If set in both, the Job value takes precedence. The assignments use the same syntactic formats defined above for System directories.

- *execSystemExecDir* - the directory where application assets are staged. It's also the current directory from which the application is launched.
  Default: ***${jobWorkingDir}/jobs/${jobID}***

- *execSystemInputDir* - the directory where application input files are staged.
- Default: ***${jobWorkingDir}/jobs/${jobID}***

- *execSystemOutputDir* - the directory where applications write their intermediate and final results.
- Default: ***${jobWorkingDir}/jobs/${jobID}/output***

- *archiveSystemDir* - the directory where applications copy their outputs for long-term storage.
  Default: none

If *archiveSystemId* is not set by either Applications or Jobs, then archiving is not attempted and application output is left unchanged where the application wrote it. A job that assigns an *archiveSystemId* without also assigning an *archiveSystemDir* will fail.

Additional directory defaults are specified in the DTN proposal[9].

2.3 Conditional Path Assignment

All of the directory assignments in Systems, Applications and Job requests discussed above can use the **HOST_EVAL($var)** function to dynamically extract an environment variable's value from an execution or archive host. The environment variable's value is retrieved by logging into the host as the Job owner and issuing

8 https://confluence.tacc.utexas.edu/display/CIC/Job+User+Macros
9 https://confluence.tacc.utexas.edu/display/CIC/Tapis+V3+Jobs+Proposal+for+DTN+Usage

"echo $var". To increase application portability, an optional default value can be passed into the **HOST_EVAL** function. The function's complete signature with the optional *path* parameter is:

HOST_EVAL($VAR, path)

If the environment variable **VAR** does not exist on the host, then the literal **path** parameter is returned by the function. This added flexibility allows applications to run in different environments, such as on TACC HPC systems that automatically expose certain environment variables and VMs that might not. If the environment variable does not exist and no optional *path* parameter is provided, the job fails due to invalid input.