BES and JSDL Extensions

Mark Morgan

This document serves as a simple report out of extensions that we (Genesis II) think are necessary going forward for production grid interoperability in XSEDE. All of the extensions in this paper fall into one of two categories; those extensions that Genesis II already has implemented that we see a continued need to maintain, and those that we haven’t implemented yet but that we see a need for as we try and build a production infrastructure with interoperable grid implementations. Further, each extension described exists either on the BES, or in the JSDL specification.

For all QName aliases used in this document are defined as follows

|  |  |
| --- | --- |
| Alias | Namespace |
| jsdl | http://schemas.ggf.org/jsdl/2005/11/jsdl |
| gii-jsdl | http://vcgr.cs.virginia.edu/genesisII/jsdl |
| gii-common | http://vcgr.cs.virginia.edu/genii/common |
| gii-bes | http://vcgr.cs.virginia.edu/genii/2008/03/bes |

# JSDL Extensions

## Unique Name Attribute

Genesis II currently defines an additional attribute that can be placed on the JSDL FileSystem resource request element. The name of this attribute is

/jsdl:JobDefinition/jsdl:JobDescription/jsdl:Resources/jsdl:FileSytem@gii-jsdl:unique-id

Currently this attribute is only used with the Genesis II **SCRATCH** file system and is there to give the BES container a unique name with which to batch the files that are downloaded as part of a job. We use the scratch space as a sort of cache for data staging, but because many different users may choose to stage in files of similar names (a.out, input.dat, etc.), we have to keep those files from conflicting with each other. Unfortunately, Genesis II does not have the notion of a unique id that we could use, so we use this attribute for that purpose[[1]](#footnote-1).

## Grid FileSystem

Genesis II currently supports a file system type that is not defined by the JSDL specification. It’s name is **GRID** and it indicates that a job wishes to have the grid namespace mounted for it before it starts running. In all other respects it is just like any other file system in the JSDL specification.

## WallclockTime Restriction

Genesis II currently supports an additional JSDL element for indicating a job’s wallclock time requirements. For some reason, the original JSDL specification did not specify this element in the resource requirements but rather put it in to the JSDL-POSIX extension. However, because this requirement encompasses more than just posix applications, we had to create a similar resource requirement restriction. It is defined as a range type just like many of the other JSDL restriction elements and it’s name is

/jsdl:JobDefinition/jsdl:JobDescription/jsdl:Resources/gii-jsdl:WallclockTime

## Matching Parameters

Genesis II supports the notion of something called a matching parameter. This is an arbitrary name/value pair that a job can use to indicate a property that it requires or supports (required properties have a value of **requires:**value and supported properties have a value of **supports:**value). BESs likewise can advertise matching parameters that it supports or requires and they are matched to these JSDL matching parameters in the obvious way. For this JSDL extension, the element is defined as

<gii-jsdl:property name=” *xsd:string* “ value=” *xsd:string* “/> **\***

and it occurs 0 or more times inside of

/jsdl:JobDefinition/jsdl:JobDescription/jsdl:Resources

These matching parameters are different from the existing mechanisms built into the JSDL specification in that they allow for a user to make requests for scheduling or matching based on completely arbitrary scheduling parameters that are agreed on outside of the JSDL specification. In theory, this could also have been done by adding new XML elements into the resources section of the JSDL, but by doing it this way, the code can be written to do the simple matching without knowing what the possible space of parameter types and values is ahead of time.

# BES Extesions

Genesis II implements the BES specification, but it also implements the OGSA-WSRF-BP. As such, we have taken the point of view that all BES attributes (as defined by the BES specification) are also implemented as WSRF Resource Properties. For interoperability reasons, we use the BES attributes for scheduling decisions, but for the purposes of this paper, I use the WSRF-RP syntax for naming resource properties, as it is more concise.

## WallclockTimeLimit (gii-bes:WallclockTimeLimit)

Genesis II currently allows for a BES to advertise a maximum wall clock time limit that it will allow for jobs. This was put in place to support batch systems that limit how much time a user’s job can consume before the batch system eliminates that job.

## Supported Filesystems (gii-bes:supported-filesystem \*)

Genesis II BES containers advertise a list of file systems that they support (this list matches the *names* of the file systems as defined in the JSDL specification. This allows our scheduling code to decide whether or not a job requesting certain file systems can run on the target BES container.

## Matching Parameters (gii-common:matching-parameter \*)

Genesis II BES containers (actually, any Genesis II resource) advertise matching parameters in a way similar to the matching parameter support in our JSDL extensions. This fulfills the corresponding role as described in that previous section for scheduling. Jobs that **require** a value get matched against BESs that **support** or **require** it, and jobs that **support** a value are allowed to run on BESs that **require** it. The full table of matching is given below. This table shows the ways that Genesis II evaluates whether or not a given job is allowed to be scheduled on a given BES.

|  |  |  |  |
| --- | --- | --- | --- |
|  | BES Supports Parameter | BES Requires Parameter | BES Does Not Advertize Parameter |
| Job Supports Parameter | Allowed | Allowed | Allowed |
| Job Requires Parameter | Allowed | Allowed | Not Allowed |
| Job Does Not Mention Parameter | Allowed | Not Allowed | Allowed |

## Supported Staging Protocols

There is no mention currently of supported staging protocols in Genesis II. This is because, in the absence of production level interoperability, we know what staging protocols we support. However, going forward under the assumption that production level interoperability will happen, the ability to decide whether or not a given data stage as requested by a job can be supported by a target BES will be crucial for making good scheduling decisions. We might be able to get away with simply listing the URI schemes supported, but my guess is that we will need to go further and also specify what kinds of authentication will be supported by the protocol.

Appendix A: Example JSDL

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<JobDefinition xmlns="http://schemas.ggf.org/jsdl/2005/11/jsdl" xmlns:ns2="http://schemas.ggf.org/jsdl/2005/11/jsdl-posix" xmlns:ns3="http://schemas.ggf.org/jsdl/2006/07/jsdl-hpcpa" xmlns:ns4="http://schemas.ogf.org/jsdl/2007/02/jsdl-spmd" xmlns:ns5="http://vcgr.cs.virginia.edu/jsdl/genii" xmlns:ns6="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd" xmlns:ns7="http://schemas.ogf.org/hpcp/2007/11/ac" xmlns:ns8="http://schemas.ogf.org/jsdl/2009/03/sweep" xmlns:ns9="http://schemas.ogf.org/jsdl/2009/03/sweep/functions">

 <JobDescription>

 <JobIdentification/>

 <Application>

 <ns2:POSIXApplication>

 <ns2:Executable>/bin/ls</ns2:Executable>

 <ns2:Argument>Argument 1</ns2:Argument>

 <ns2:Argument filesystemName="GRID">Argument 2</ns2:Argument> <ns2:Argument filesystemName="SCRATCH">Argument 3</ns2:Argument>

 </ns2:POSIXApplication>

 </Application>

 <Resources> <FileSystem name="GRID">

 <FileSystemType>normal</FileSystemType> </FileSystem>

 <FileSystem xmlns:ns10="http://vcgr.cs.virginia.edu/genesisII/jsdl" name="SCRATCH" ns10:unique-id="unique-scratch-id">

 <FileSystemType>spool</FileSystemType>

 </FileSystem>

 <ns5:property value="Some Value" name="requires:Matching Parameter 1"/>

 <ns5:property value="Another Value"

 name="supports:Matching Parameter 2"/>

 </Resources>

 </JobDescription>

</JobDefinition>

1. Please see Appendix A for an example JSDL file showing these various extensions. [↑](#footnote-ref-1)