July 6, 2004

Agenda

- Document Status
- Setting up a Tracker
- Use Cases
- XML and XSD
- Logical Operators

Attendees

Darren P
Anuj M
Subir R
Fred B
Donal F
Andreas S
An Ly
Steve M

Action Items

==========

- Document is currently in Darren's Hands.
- Everyone to use the Tracker to get changes into the document.
- Everyone to try and write and XML document to submit the types of jobs they are working with.
- Steve to Make changes to XSD with new changes from the Resource section.
- Software Requirements are still weak. We need to add more to this section. Licensing would be an example. Versions of the application or libraries.

[Text Slide A]

XML and XSD

```
<?xml version="1.0" encoding="UTF-8" ?>
- <jsdl:job xmlns:jsdl="http://www.gridforum.org/JSDL"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.gridforum.org/JSDL jsdl.xsd">
- <!--
=============================================================================
  -->
- <!--  The first section of the JSDL document describes the general features of
  -->
- <!--  a job. In the common case this section is all that you will need.
  -->
- <!--
=============================================================================
  -->

- <jsdl:JobIdentification>
   <jsdl:JobName>MyFirstJob</jsdl:JobName>
   <jsdl:JobAnnotation>This is the first job ever submitted through JSDL</jsdl:JobAnnotation>
  <jsdl:JobAnnotation>This job won't really do anything</jsdl:JobAnnotation>
-  <jsdl:ExecutionUserID>
    <jsdl:User name="steve" group="myGroup" />
   </jsdl:ExecutionUserID>
   <jsdl:JobProjectName>WritingTheSpecProject</jsdl:JobProjectName>
  <jsdl:JobCategory>writingJob</jsdl:JobCategory>
   <jsdl:Extend name="PBSQueueName">myQueue</jsdl:Extend>
  </jsdl:JobIdentification>
```

[Text Slide C]

Resource XML

==========

- <jsdl:Resource>
- <!-- The following describes the general architecture that should be used -->
- <!-- for all resources in the job. Ie all boxes will fall within these -->
- <!-- requirements. -->
- <!-- Unless the job is complex (eg MPI) then this is all that is needed -->
- <!-- Only use intel and spark processors for this job -->
<jsdl:CPUDescription>jsdlprocessor:intel8086</jsdl:CPUDescription>
<jsdl:CPUDescription>jsdlprocessor:sparc</jsdl:CPUDescription>
- <!-- Each box used must have at least 1 processor -->
<jsdl:CPUCount>1</jsdl:CPUCount>
- <!-- All boxes must run at at least 8Mhz -->
<jsdl:CPUSpeed>8Mhz</jsdl:CPUSpeed>
- <!-- Each box must have at least 1024 Mb of memory -->
<jsdl:PhysicalMemory>1024Mb</jsdl:PhysicalMemory>
- <!-- See usage record for units -->
</jsdl:Resource>

<jsdl:Resource>
<jsdl:CPUDescription xmlns:foobar="???">foobar:hp</jsdl:CPUDescription>
- <! - Do we need to have a way to specify where the name is coming from? This will give us the ability ->
- <! - to add predefined valid values for specific element types ->
</jsdl:Resource>

<! - We cannot see the need to use the explicity AND and OR at this time ->
<! - The rules for ANDing and ORing things together is described below ->
<jsdl:Resource>
<jsdl:AND>
<jsdl:OR>
<jsdl:AND>
<jsdl:CPUDesc>i86</>
<jsdl:PhysMem>8MB</>
</jsdl:AND>
<jsdl:AND>

```
            <jsdl:CPUDesc>Alpha</>
            <jsdl:PhysMem>16MB</>
         </jsdl:AND>
      </jsdl:OR>
      <jsdl:CPUCount>1</>
   </jsdl:AND>
</jsdl:Resource>
<! - Do not use this above ->

<!- -->
<jsdl:Resource name="darrensResource">
    <jsdl:Resource name="bar">
            <jsdl:CPUDesc>i86</>
            <jsdl:PhysMem>8MB</>
    </jsdl:Resource>

    <jsdl:Resource name="foo">
            <jsdl:CPUDesc>Alpha</>
            <jsdl:PhysMem>16MB</>
    </jsdl:Resource>
    <jsdl:CPUCount>1</>
  <!-- Copy contents of foo and then add extra bits -->
   <jsdl:Resource name="foo2" derivedfrom="foo bar">
       <jsdl:CPUSpeed>200Mhz</>
  </jsdl:Resource>
</jsdl:Resource>

<!- -->

- <!-- Where we have two elements of the same type they are combined with an OR operator -->
- <!-- Where we have elements that are of a different type they are combined with an AND operator --
>
- <! - derivedfrom is a attribute that is a reference to an identifier attribute of an exsisting element ->
- <! - Do we want to have the ability to have more than one derivedfrom attribute for a Resource
element? ->
- <! - If we have multiple inheritance then the resource sets are just copied into the Resource
- <! - and the rules above apply.  So the Resource foo2 would actually be: ->
```

```
<jsdl:Resource name="foo2">
            <jsdl:CPUDesc>i86</>
            <jsdl:PhysMem>8MB</>
            <jsdl:CPUDesc>Alpha</>
            <jsdl:PhysMem>16MB</>
      <jsdl:CPUSpeed>200Mhz</>
</jsdl:Resource>
```

- <! - We are still having reservations about using multiple inheritance ->

[Text Slide D]

Not discussed today
==========

- <!-- Process topology - we need to define this some people may like to
  -->
- <!-- try writing some examples based on their needs here.
  -->
  <jsdl:ProcessTopology>?????</jsdl:ProcessTopology>

- <!-- Set up environmental variables
  -->

- <jsdl:Environment>
- <!-- These variables apply to all boxes used
  -->
  <jsdl:EnvironmentVariable name="TASKID">Some Value</jsdl:EnvironmentVariable>
  <jsdl:EnvironmentVariable name="PROBLEMSIZE">1000</jsdl:EnvironmentVariable>
  </jsdl:Environment>
- <jsdl:SoftwareRequirements>
- <jsdl:OperatingSystem>
  <jsdl:OperatingSystemDescription>Windows2000</jsdl:OperatingSystemDescription>
  <jsdl:OperatingSystemVersion>2000</jsdl:OperatingSystemVersion>
  </jsdl:OperatingSystem>
- <jsdl:OperatingSystem>
  <jsdl:OperatingSystemDescription>Linux</jsdl:OperatingSystemDescription>
  <jsdl:OperatingSystemVersion>RedHat7.2</jsdl:OperatingSystemVersion>
  </jsdl:OperatingSystem>
- <jsdl:Limits>
  <jsdl:ProcessVirtualMemoryLimit>1Gb</jsdl:ProcessVirtualMemoryLimit>
  <jsdl:VirtualMemoryLimit>1Gb</jsdl:VirtualMemoryLimit>
  <jsdl:DataSegmentSizeLimit>10Gb</jsdl:DataSegmentSizeLimit>
  <jsdl:CoreDumpSizeLimit>0</jsdl:CoreDumpSizeLimit>
  <jsdl:CPUTimeLimit>10Hours</jsdl:CPUTimeLimit>
  <jsdl:WallTimeLimit>24Hours</jsdl:WallTimeLimit>
  </jsdl:Limits>
  <jsdl:Queue>myQueue</jsdl:Queue>

```
  <jsdl:Queue>anotherQueue</jsdl:Queue>
  </jsdl:SoftwareRequirements>
- <jsdl:Application>
  <jsdl:ExecutableDescription>Run the vi command</jsdl:ExecutableDescription>
  <jsdl:ExecutableName type="bash">vi</jsdl:ExecutableName>
  <jsdl:Argument>-noX</jsdl:Argument>
  <jsdl:Argument>myFile.txt</jsdl:Argument>
  <jsdl:StdIn>myin.txt</jsdl:StdIn>
  <jsdl:StdOut>myOut.txt</jsdl:StdOut>
  <jsdl:StdErr>myErr.txt</jsdl:StdErr>
  <jsdl:WorkingDirectory>myBrainDump</jsdl:WorkingDirectory>
  <jsdl:Log>processLog.txt</jsdl:Log>
  </jsdl:Application>
- <!--
========================================================================
  -->
- <!--  In this section of the document we describe a single host that can be
  -->
- <!--  used to run the job on. This section defines a resource either
  -->
- <!--  explicitly named or defined by the characteristics defined within
  -->
- <!--  note that the following should be seen as extending the definition of a
  -->
- <!--  resource specified above in the general section.
  -->
- <!--
========================================================================
  -->
- <jsdl:Host name="bar" description="This is a specific host type">
- <!--  List of host names we know match the requirements
  -->
  <jsdl:HostName>myFastBox.ggf.org</jsdl:HostName>
  <jsdl:HostName>anotherGoodBox.ggf.org</jsdl:HostName>
- <!--  Valid user id's that can be used on these boxes
  -->
- <jsdl:ExecutionUserID>
```

```
  <jsdl:User name="asm" group="secondGroup" />
  </jsdl:ExecutionUserID>
- <jsdl:Resource>
- <!--  The following describes the general architecture that should be
  -->
- <!--  used for this resources in the job. Ie boxes matching this
  -->
- <!--  description can be used for this host "id".
  -->
- <!--  Only use intel and spark processors for this job
  -->
  <jsdl:CPUDescription>itel8486</jsdl:CPUDescription>
  <jsdl:CPUDescription>spark2</jsdl:CPUDescription>
- <!--  Each box used must have at least 2 processors
  -->
  <jsdl:CPUCount>2</jsdl:CPUCount>
- <!--  All boxes must run at at least 1024Mhz
  -->
  <jsdl:CPUSpeed>1024Mhz</jsdl:CPUSpeed>
- <!--  Each box must have at least 4096 Mb of memory
  -->
  <jsdl:PhysicalMemory>4096Mb</jsdl:PhysicalMemory>
- <!--  See usage record for units
  -->
  </jsdl:Resource>
- <!--  Process topology - we need to define this some people may like to
  -->
- <!--  try writing some examples based on their needs here.
  -->
  <jsdl:ProcessTopology>?????</jsdl:ProcessTopology>
- <!--  Set up environmental variables
  -->
- <jsdl:Environment>
- <!--  These variables apply to all boxes used
  -->
  <jsdl:EnvironmentVariable name="TASKID">Some new Value</jsdl:EnvironmentVariable>
  <jsdl:EnvironmentVariable name="PROBLEMSIZE">2000</jsdl:EnvironmentVariable>
```

```
  <jsdl:EnvironmentVariable name="CORRECTION">-1</jsdl:EnvironmentVariable>
  </jsdl:Environment>
- <jsdl:SoftwareRequirements>
- <jsdl:OperatingSystem>
  <jsdl:OperatingSystemDescription>Windows2000</jsdl:OperatingSystemDescription>
  <jsdl:OperatingSystemVersion>2000.0.1.3</jsdl:OperatingSystemVersion>
  </jsdl:OperatingSystem>
- <jsdl:OperatingSystem>
  <jsdl:OperatingSystemDescription>Linux</jsdl:OperatingSystemDescription>
  <jsdl:OperatingSystemVersion>RedHat7.2</jsdl:OperatingSystemVersion>
  <jsdl:Extend name="patchVersion">October2003</jsdl:Extend>
  </jsdl:OperatingSystem>
- <jsdl:Limits>
  <jsdl:ProcessVirtualMemoryLimit>2Gb</jsdl:ProcessVirtualMemoryLimit>
  <jsdl:VirtualMemoryLimit>0.5Gb</jsdl:VirtualMemoryLimit>
  <jsdl:DataSegmentSizeLimit>15Gb</jsdl:DataSegmentSizeLimit>
  <jsdl:CoreDumpSizeLimit>2bytes</jsdl:CoreDumpSizeLimit>
  <jsdl:CPUTimeLimit>11Hours</jsdl:CPUTimeLimit>
  <jsdl:WallTimeLimit>48Hours</jsdl:WallTimeLimit>
  </jsdl:Limits>
  <jsdl:Queue>brokenQueue</jsdl:Queue>
  <jsdl:Queue>workingQueue</jsdl:Queue>
  </jsdl:SoftwareRequirements>
- <!--  Allow a process topology to be specified here
  -->
  </jsdl:Host>
- <!--
==========================================================================
  -->
- <!--  In this section of the document we describe a host grouping that can be
  -->
- <!--  used to run the job on. This group defines a collection of resources
  -->
- <!--  either explicitaly named or defined by the characteristics defined within
  -->
- <!--  note that the following should be seen as extending the definition of a
  -->
```

```
- <!--  resource specified above in the general section.
  -->
- <!--
==========================================================================
  -->
- <jsdl:HostGroup name="foo" description="This host type can be used for our job">
- <!--  Some examples of the resources that match this host type. We do not
  -->
- <!--  limit the system to only use these resources - but they are
  -->
- <!--  probably a good starting point.
  -->
  <jsdl:HostName>mybox.ggf.org</jsdl:HostName>
  <jsdl:HostName>secondbox.ggf.org</jsdl:HostName>
  <jsdl:HostName>resourcePool1</jsdl:HostName>
- <!--  Valid user id's that can be used on these boxes
  -->
- <jsdl:ExecutionUserID>
  <jsdl:User name="asm2" group="thirdGroup" />
  </jsdl:ExecutionUserID>
- <jsdl:Resource>
- <!--  Hosts in this type are more narrowly specified
  -->
- <!--  Only use intel and spark processors for this job
  -->
  <jsdl:CPUDescription>itelPentium4</jsdl:CPUDescription>
- <!--  Each box used must have at least 4 processors
  -->
  <jsdl:CPUCount>4</jsdl:CPUCount>
- <!--  All boxes must run at at least 2048Mhz
  -->
  <jsdl:CPUSpeed>2048Mhz</jsdl:CPUSpeed>
- <!--  Each box must have at least 4096 Mb of memory
  -->
  <jsdl:PhysicalMemory>4096Mb</jsdl:PhysicalMemory>
- <!--  See usage record for units
  -->
```

- <jsdl:Network>
- <!-- Boxes of this type must have fast ethernet cards
-->
<jsdl:NetworkDescription>FastEthernet</jsdl:NetworkDescription>
- <!-- and at least two cards per box
-->
<jsdl:NetworkCount>2</jsdl:NetworkCount>
- <!-- Capable of supporting 100MB transfers
-->
<jsdl:NetworkBandwidth>100MB</jsdl:NetworkBandwidth>
</jsdl:Network>
- <!-- Provide a file space called scratch which can store at least
-->
- <!-- 4096Mb of data
-->
<jsdl:FileSystem name="scratch" size="4096Mb" />
- <!-- Have (or be able to have" the following filesystem mounted
-->
<jsdl:FileSystem name="myBrainDump"
size="10GB">nfs:hostname:/export/vol01</jsdl:FileSystem>
- <!-- Have 100GB of swap space
-->
<jsdl:SwapSpace>100GB</jsdl:SwapSpace>
- <!-- Have temp space of 200KB of data
-->
<jsdl:TemporarySpace>200KB</jsdl:TemporarySpace>
- <!-- Our job requires exclusive use of these nodes?
-->
<jsdl:ExclusiveExecution>true</jsdl:ExclusiveExecution>
</jsdl:Resource>
- <!-- Process topology - we need to define this some people may like to
-->
- <!-- try writing some examples based on their needs here.
-->
<jsdl:ProcessTopology>?????</jsdl:ProcessTopology>
- <!-- Set up environmental variables
-->

```
- <jsdl:Environment>
- <!--  These variables apply to all boxes used
  -->
  <jsdl:EnvironmentVariable name="CORRECTION">-2</jsdl:EnvironmentVariable>
  </jsdl:Environment>
- <jsdl:SoftwareRequirements>
- <jsdl:Limits>
  <jsdl:ProcessVirtualMemoryLimit>4Gb</jsdl:ProcessVirtualMemoryLimit>
  <jsdl:VirtualMemoryLimit>0.5Gb</jsdl:VirtualMemoryLimit>
  <jsdl:DataSegmentSizeLimit>75Gb</jsdl:DataSegmentSizeLimit>
  </jsdl:Limits>
  <jsdl:Queue>brokenQueue</jsdl:Queue>
  <jsdl:Queue>workingQueue</jsdl:Queue>
  </jsdl:SoftwareRequirements>
  </jsdl:HostGroup>
- <!--
==========================================================================
  -->
- <!--  In this section the files that are required to be staged in/out of the
  -->
- <!--  resource are defined and the possible locations where they may be staged
  -->
- <!--  to/from can be defined.
  -->
- <!--
==========================================================================
  -->
- <jsdl:DataAttributes>
- <jsdl:File>
  <jsdl:FileName>myFirstFile.txt</jsdl:FileName>
  <jsdl:FileSystemName>myBrainDump</jsdl:FileSystemName>
  <jsdl:Source>http://www.ggf.org/myFirstFile.txt</jsdl:Source>
  <jsdl:CreationFlag>overwrite</jsdl:CreationFlag>
  </jsdl:File>
- <jsdl:File>
  <jsdl:FileName>mySecondFile.data</jsdl:FileName>
  <jsdl:FileSystemName>myBrainDump</jsdl:FileSystemName>
```

<jsdl:Target>gsiftp::box.ggf.org/homes/ggf/mySecondFile.data</jsdl:Target>
<jsdl:CreationFlag>append</jsdl:CreationFlag>
</jsdl:File>
- <jsdl:File>
<jsdl:FileName>StageThrough.data</jsdl:FileName>
<jsdl:FileSystemName>scratch</jsdl:FileSystemName>
<jsdl:Source>ftp://ftp.ggf.org/largeData/input.dat</jsdl:Source>
<jsdl:Source>ftp://ftp.slowbox.org/largeDataBackup/record.data</jsdl:Source>
<jsdl:Target>gsiftp::box.ggf.org/homes/ggf/result.data</jsdl:Target>
<jsdl:CreationFlag>overwrite</jsdl:CreationFlag>
</jsdl:File>
</jsdl:DataAttributes>
</jsdl:job>

[Text Slide E]

Use Cases (Donal)
The following are versions of some simple jobs that are the sorts of things that are handled by UNICORE. I'm using English descriptions, not JSDL, but I hope they will be fairly straight-forward to convert (or at least the bits that are applicable.)

* Simple application invokation

  User wants to run the application 'gnuplot' to produce a substantial number of plotted graphical files based on some data shipped in from elsewhere (as part of a workflow) which a front-end application will then build into an animation of spinning data.

  Front-end application knows URL for data file which must be staged in.

  Front-end application wants to stage in a control file that it specifies directly which directs gnuplot to produce the output files.

  Front-end application knows how many output graphic files are produced and what their names are and wants to stage them to itself upon successful completion of the job and would prefer to stage them as some kind of archive file.

  In case of error, messages should be produced on stderr (also to be staged on completion) and no images are to be transferred.

* Slightly more complex application invokation and workflow integration

  User wants to run the application 'octave' to generate an input dataset to another application (bonus points for indicating how this fits with the previous use case.)

  User supplies the *.oct file to configure octave.

  User has no idea where the application is installed; it is not on the path, but the job engine is aware of where it is provided.

Workflow manager wants all components in the workflow to use a common working directory to make transferring intermediate files easy.

Output file is to be left in working directory if program succeeds, and deleted if program fails.

Stderr to be separate from all other components in the workflow to facilitate easier debugging.

(I can't remember how much use of parallelism octave can make; my concrete example happens to be on a single-processor machine...)

* Parallel lightly-coupled composite application invokation

User wants to run a weather simulation application. Application is in several parts.

- o Global weather model (model as external data source/webservice)
- o Topology/data-grid interpolator (local models work with a cartesian coordinate system, global models work with a polar coordinate system)
- o Local weather model
- o Visualization engine

All three job-like components can make reasonable use of parallel processing, but the codes have characteristics that prefer different kinds of processors (e.g. SIMD vs MIMD)

Intermediate datasets may be in multi-gigabyte range; GridFTP (or other parallel transfer scheme) is strongly preferred as a data transfer mechanism, but not required. It is better that the data gets through at all than the workflow fails due to some GridFTP fault.

Applications may only be executed by registered users; they are not on anyone's path and never will be.

Application front-end knows enough about the basic global topology to

be able to provide basic hints to the (first two) applications about how to tile their assigned processors across their input data.

OK, the last one is definitely well beyond JSDL as it currently stands, but it indicates a case that interested real users in the EUROGRID and GRIP projects and it's probably a good idea if we check that we're not going to get in the way of it. :^)

None of the above require specification of the processor or OS; the focus is on the application.

[Text Slide B]

A Sharing Slide

[Share A]