Extensions to JSDL

XtreemOS Consortium

Matej Artac (XLAB), Massimo Coppola (CNR), Toni Cortes (BSC),

Yvon Jegou (INRIA), John Mehnert-Spahn (UDUS), Ramon Nou (BSC)

March 2010

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

XtreemOS

- **XtreemOS is a Grid Operating System**
- **Provides transparent access to the Grid**
  - "run on the grid as if you where on your desktop"
- **To execute a job:**
  - Execute the application JSDL on a desktop
- **XtreemOS project supported by the EU**
  - **`www.xtreemos.eu`**

Outline

- Dynamic behavior in resource selection
- Resource selection tolerance
- Location of files
- Job execution time
- Firewalls
- Interactive jobs
- Credential delegation
- Checkpointing

Managing dynamic behavior

- **XtreemOS allows *physical resource sharing***
  - **Multiple processes, containers or VMs : akin to Clouds**
  - Good application matching is important

App A	App B	App A + APP B
• 30 % CPU load • 50 % RAM load • 20 %Net BW	• 60% CPU load • 30% RAM load • 30% Net BW	• 90% CPU load • 80% RAM load • 50% Net BW

- **XtreemOS allows interactive applications**
  - Dynamic app behaviour, dynamic app start/stop
  - Appropriate policies and resource selection strategies
- **Associate a dynamic meaning to most JSDL tags**
  - E.g. Total RAM ←→ Free RAM
  - "Static" value may differ from "dynamic" value

Resource selection tolerance

- **Constraint on dynamic resource characteristics**
  - Applications may accept some tolerance
    - needs 4GB of memory, but would run with 3GB free
    - needs a 3Ghz Xeon, but would accept a 30% loaded machine
- **Proposed changes**
  - `Resource` tags are extended with a new attribute to indicate tolerance
    - Real change is in the `RangeValue_Type` tag
      - Backward compatible
    - Dynamic threshold value = static bound * tolerance
    - For those attributes whose dynamic semantics is meaningful

Example

- **Comparison with dynamic values can be slacker**

```
<jsdl-srds:IndividualDiskSpace>
   <jsdl-srds:Range>
       <jsdl-srds:LowerBound tolerance="0.1">100000000
       </jsdl-srds:LowerBound>
       <jsdl-srds:UpperBound tolerance="0.9">2400000000
       </jsdl-srds:UpperBound>
   </jsdl-srds:Range>
</jsdl-srds:IndividualDiskSpace>
```

- **Select machines whose**
  - disk space amount X is : 100MB < X < 2400MB
  - *free* disk space Y is : 0.1* 100MB < Y < 2400MB * 0.9

Example

```
<jsdl-srds:IndividualDiskSpace>
   <jsdl-srds:Exact tolerance="0.9">10000000
   </jsdl-srds:Exact>
</jsdl-srds:IndividualDiskSpace>
```

- **Exact requirements turned into tolerance ranges**
- **Select machines whose**
  - Whose disk space amount X is : X == 10MB
  - Whose *free* disk space Y is : **0.9** * 10MB < Y < 10MB

Partially loaded nodes

- **Exclusive access to resources not always needed**
  - Application may be willing to share a CPU
  - Historically reliable machines may be preferred
- **Proposed changes**
  - Add tag `Uptime` (machine uptime in min)
  - Add tag `IdlePercentage` (% of idle cycles)
  - Already dynamic, no tolerance attribute needed
  - Can be combined with other tags possibly using dynamic tolerance
  - Allow to select unloaded, long standing machines

Example

- **Ordinary JSDL use**

```
<jsdl-srds:IdlePercentage><jsdl:Range>
   <jsdl:LowerBound>0</jsdl:LowerBound>
   <jsdl:UpperBound>50</jsdl:UpperBound>
</jsdl:Range></jsdl-srds:IdlePercentage>
<jsdl-srds:Uptime>
   <jsdl:LowerBoundedRange>2</jsdl:LowerBoundedRange>
</jsdl-srds:Uptime>
```

- **Select machines with less than 50% CPU idle**
  - Load is always measured dynamically
- **Refuse machines who have just boot up** (>2 min)

Working implementation

- **XtreemOS employs described extensions**
- **Syntax refined after first prototypes**
- **XML schemata defined extending standard JSDL**
  - Extensions are backward compatible
- **XML validation enforced**

Extension Schema fragment

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
         targetNamespace="http://schemas.ggf.org/jsdl/2005/11/jsdl-srds"
         elementFormDefault="qualified"
         xmlns="http://schemas.ggf.org/jsdl/2009/11/jsdl-srds"
         xmlns:jsdl-srds="http://schemas.ggf.org/jsdl/2005/11/jsdl-srds"
         xmlns:jsdl="http://schemas.ggf.org/jsdl/2005/11/jsdl">
<!-- Import normative schema -->
<xsd:import namespace="http://schemas.ggf.org/jsdl/2005/11/jsdl"
         schemaLocation="Jsdl_Normative_OGF.xsd"/>
<!-- COMPLEX TYPES: Definitions for the RangeValueType -->
<xsd:complexType name="Boundary_Type"
         xmlns="http://schemas.ggf.org/jsdl/2005/11/jsdl-srds">
    <xsd:simpleContent>
      <xsd:extension base="xsd:double">
        <xsd:attribute name="exclusiveBound" type="xsd:boolean" use="optional"/>
        <xsd:attribute name="tolerance" type="xsd:double"
         xmlns="http://schemas.ggf.org/jsdl/2005/11/jsdl-srds" use="required"/>
           <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:extension>
     </xsd:simpleContent>
</xsd:complexType>
```

File usage

- **Try to get resources close to data**
  - Applications need a way to specify the important files
    - Can be used by scheduler as a hint
  - Not the same as stage in/out files
    - It is complementary
    - Not all files may be important
    - Depending on file system, some stage in/out may be needed
- **Proposed changes**
  - New tag (`SchedulingHint`) with file information

Example

```
<SchedulingHint>
   <FileSystem type="XtreemFS">
      <Volume id="Default">
         <File>out.txt</File>
         <File>tmpdir/process.txt</File>
      </Volume>
   </FileSystem>
</SchedulingHint>
```

Job time

- **User has little control on when a job is executed**
  - Cannot decide
    - Starting time
    - Days and/or times when a job can be executed
    - Duration (it it takes longer checkpoint and stop)
- **Proposed changes**
  - New tag (`LifeTime`) constrains on job execution

Example

```
<LifeTime>
   <StartTime> datetime </StartTime>
   <ExecutionTime> time </ExecutionTime>
   <Constraints>
      <Constraint>
         <DayOfWeek> dayofweek </DayOfWeek>
            <TimeInterval>
               <Start> time </Start>
               <End> time </End>
            </TimeInterval>
      </Constraint>
   </Constraints>
</LifeTime>
```

Firewall

- **Firewalls make jobs life nearly impossible**
  - We should offer a way to specify network needs
  - Resource selections should take these needs into account
- **Proposed changes**
  - Add some parameters to the resource tag

Example

```
<jsdl:Resources>
    <network:Network
     xmlns:net=http://xtreemos.org/schemas/jsdl/net/2008
     05>
        <net:Netmask>201.123.123.0/24</net:Netmask>
        <net:Ports>1000-2000,60000,65000</net:Ports>
        <net:Proto>TCP</net:Proto>
    </network:Network>
</jsdl:Resources>
```

Interactive jobs

- **Interactive jobs should be executable in a Grid**
  - We need to detect which jobs are interactive
  - We need to set the environment for their interactivity
- **Proposed changes**
  - Extend application element to describe interactivity

Example

- Solution 1: extend existing tags

```
<Input>/dev/tty</Input>
<Output>stdout</Output>
```

- Add new tags

```
<X11>true/false</X11>
```

- Tags need to be analyzed on both client and resource sides

SSO / delegation

- **XtreemOS jobs can execute Grid requests**
  - Interactive jobs
  - Needs credentials from the user (certificates, …)
- **Various delegation schemes defined**
  - Proxy certificates
  - XtreemOS SSO service
  - Dtokens
- **When delegation is not handled in user space**
  - Need to be requested from JSDL
- **Need a new tag**

Checkpointing

- **Checkpointing is a common case in the Grid**
  - Applications should be able to specify its parameters
    - Who should initiate it (i.e user/system)
    - Which checkpointer to use (i.e. BLRC version X.Y)
    - Which container (i.e. cgroups)
    - Application information (i.e. #procs, #threads, …)
    - Type of checkpointing protocol (i.e. coordianted or not)
- **Proposed changes**
  - Add a new tag (`JobFaultTolerance`) with all needed parameters

Example

```
<JobFaultTolerance>
      <Initiator>
            <User> yes </User>
            <OperatingSystem> yes </OperatingSystem>
                  <Application>no</Application>
      </Initiator>

      <Checkpointer> BLCR </Checkpointer>

      <CheckpointerVersion>0.8.2
  </CheckpointerVersion>
  ...
```

Example

```
...
   <ContainerType>cgroups</ContainerType>
   <ContainerParameter>
       <NetworkNS>yes</NetworkNS>
       <PidNS>yes>/PidNS>
       <IpcNS>yes</IpcNS>
       <Uts>yes</Uts>
   </ContainerParameter>
       ...
```

Example

```
...
<ApplicationSoftwareResources>
      <Singleproc>yes</Singleproc>

      <Singlethread>yes</Singlethread>

      <Sysipcshm>yes</Sysipcshm>

      <Sysipcmsgq>yes<Sysipcmsgq>
      <Sysipcsem>yes<Sysipcsem>
      <Files>yes<Files>
   </ApplicationSoftwareResources>
...
```

Example

```
...
   <CheckpointProtocol>Coordinated</CheckpointProtocol>
       <CheckpointProtocolParameter>
       <Periodinseconds>5</Periodinseconds>
       </CheckpointProtocolParameter>
</JobFaultTolerance>
```