

ETSI TS 102 828 V1.1.1 (2008-08)

Technical Specification

**GRID;
Grid Component Model (GCM);
GCM Application Description**



Reference

DTS/GRID-0004-2

Keywords

interoperability, service, architecture, network,
application

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Individual copies of the present document can be downloaded from:

<http://www.etsi.org>

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at

<http://portal.etsi.org/tb/status/status.asp>

If you find errors in the present document, please send your comment to one of the following services:

http://portal.etsi.org/chaicor/ETSI_support.asp

Copyright Notification

No part may be reproduced except as authorized by written permission.
The copyright and the foregoing restriction extend to reproduction in all media.

© European Telecommunications Standards Institute 2008.
All rights reserved.

DECTTM, **PLUGTESTS**TM, **UMTS**TM, **TIPHON**TM, the TIPHON logo and the ETSI logo are Trade Marks of ETSI registered for the benefit of its Members.

3GPPTM is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners.

Contents

Intellectual Property Rights	4
Foreword.....	4
Introduction	4
1 Scope	5
2 References	5
2.1 Normative references	5
2.2 Informative references.....	5
3 Definitions	6
4 Overall Structure of the GCM Application Description.....	6
4.1 Principles.....	6
4.2 Overall XML structure	6
4.2.1 Environment	7
5 Application Specification	7
5.1 Executable	8
5.2 ProActive.....	9
5.2.1 Overall structure	9
5.2.2 ProActive definition.....	9
5.2.3 Technical Services	12
5.3 GCM Component	13
5.4 MPI.....	13
6 Resource Specification	13
6.1 Node Provider	13
Annex A (normative): Schema	14
Annex B (informative): Example of applicationdescriptor	17
Annex C (informative): Bibliography.....	18
History	19

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: "*Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards*", which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (<http://webapp.etsi.org/IPR/home.asp>).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee GRID (TC Grid).

The present document is related to document TS 102 827 [1] (GCM Interoperability Deployment).

Introduction

The purpose of the present document is to offer a standard, uniform way of describing grid-based applications, and the resources they need.

When building a grid, one is confronted with several problems, among which:

- choosing a grid framework, as many are available;
- when running a grid-enabled application, one must rewrite configuration directives when switching to a different grid framework;
- a grid can be built on widely different hardware infrastructures, from an heterogeneous set of desktop machines to a dedicated cluster, or even any combination of those;
- grid infrastructures can also be very dynamic, it should be easy to add or remove machines, and such changes should be transparent to users.

To simplify this, the GCM Interoperability Deployment standard (TS 102 827 [1]) offers a uniform way to describe grid resources. On the application side, the present document describes how various resources can be used by an application, providing gridification of applications in a very portable manner.

1 Scope

The present document describes an XML schema for describing application constraints and resources to be used for deploying these applications on distributed and parallel infrastructures, like enterprise and scientific grids or job schedulers.

The standard will help enterprises and laboratories to manage their IT software within large-scale computer and telecom infrastructures with the necessary virtualization.

Its primary audience are users of grid-based applications who will need to write application descriptors so they can run them on the grid for their grid.

2 References

References are either specific (identified by date of publication and/or edition number or version number) or non-specific.

- For a specific reference, subsequent revisions do not apply.
- Non-specific reference may be made only to a complete document or a part thereof and only in the following cases:
 - if it is accepted that it will be possible to use all future changes of the referenced document for the purposes of the referring document;
 - for informative references.

Referenced documents which are not found to be publicly available in the expected location might be found at <http://docbox.etsi.org/Reference>.

For online referenced documents, information sufficient to identify and locate the source shall be provided. Preferably, the primary source of the referenced document should be cited, in order to ensure traceability. Furthermore, the reference should, as far as possible, remain valid for the expected life of the document. The reference shall include the method of access to the referenced document and the full network address, with the same punctuation and use of upper case and lower case letters.

NOTE: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long term validity.

2.1 Normative references

The following referenced documents are indispensable for the application of the present document. For dated references, only the edition cited applies. For non-specific references, the latest edition of the referenced document (including any amendments) applies.

- [1] ETSI TS 102 827: "GRID; Grid Component Model (GCM); GCM Interoperability Deployment".

2.2 Informative references

The following referenced documents are not essential to the use of the present document but they assist the user with regard to a particular subject area. For non-specific references, the latest version of the referenced document (including any amendments) applies.

- [i.1] <http://proactive.inria.fr>.
- [i.2] <http://www.mpi-forum.org/docs/>.
- [i.3] <http://fractal.objectweb.org/>.

3 Definitions

For the purposes of the present document, the terms and definitions given in TS 102 827 [1] and the following apply:

component: abstraction of a software entity with a well-defined interface for both server and client parts

component model: specification of how components are defined and interact together

executable: arbitrary program which can be the launch target of an GCM Application Descriptor

fractal: modular and extensible component model

NOTE: (see <http://fractal.objectweb.org/> [i.3]).

Message Passing Interface (MPI): a computer specification and is an implementation that allows many computers to communicate with one another

NOTE: (see <http://www.mpi-forum.org/docs/> [i.2]).

ProActive: Java Grid middleware for parallel, distributed and multi-threaded computing

NOTE 1: (see <http://proactive.inria.fr> [i.1]).

NOTE 2: ProActive is part of the ObjectWeb/OW2 consortium and developed by INRIA, CNRS and University of Nice Sophia Antipolis, with Open Source code under the GPL license.

4 Overall Structure of the GCM Application Description

4.1 Principles

Like a deployment descriptor, an application descriptor is written in XML. Each element of the descriptor is defined as a single node, which can have attributes and child nodes.

Also, see clause 6.1 of TS 102 827 [1].

4.2 Overall XML structure

A deployment descriptor has the following structure:

```
<environment>
  <descriptorVariable .../>
  <programVariable .../>
  <descriptorDefaultVariable .../>
  <programDefaultVariable .../>
  <includePropertyFile ...>
  ...
</environment>

<application>
...
</application>

<resources>
  <nodeProvider>
    <file .../>
    ...
  </nodeProvider>
  ...
</resources>
```

The elements must be specified in this order. The `<environment>` element can be omitted, while the other ones are mandatory.

The `<environment>` element is a table of variable names and their values, which is used to give some amount of flexibility to the descriptor.

The `<application>` element describes the application that will be run on the grid, and the amount of resources it will use (the number of processing nodes).

Finally, the `<resources>` element describes the resources made available by the grid, by referencing deployment descriptors through `<nodeProvider>` elements.

4.2.1 Environment

The `<environment>` element can have the following children elements:

- **descriptorVariable:** the value has to be set in the descriptor, and cannot be specified in the program;
- **descriptorDefaultVariable:** a default value must be specified in the descriptor. The program has the ability to change the value at execution. If the value is changed in the program, then this new value will have precedence over the one defined in the descriptor;
- **programVariable:** the value must be set in the program, and cannot be specified in the descriptor;
- **programDefaultVariable:** a default value must be specified by the program as a constant, or computed (based on input size, for instance). The descriptor has the ability to change the value. If the value is changed in the descriptor, then this new value will have precedence over the one defined in the program;
- **includePropertyFile:** read variable definitions from a property file. A property file contains variable definitions as "NAME = VALUE" lines, for example:
 - variable1 = value1;
 - variable2 = value2.

There can be any number of these children elements. They cannot have child elements.

`<descriptorVariable>`, `<descriptorDefaultVariable>` and `<programDefaultVariable>` have two attributes, "name" and "value", each accepting string values.

`<programVariable>` only has a single "name" attribute (since it can only be set in the program).

`<includePropertyFile>` only has a single "location" attribute.

For example:

```
<environment>
  <descriptorVariable name="usertype" value="admin" />
  <descriptorDefaultVariable name="username" value="jsmith" />
  <programVariable name="nbNodes" />
  <programDefaultVariable name="nbJobs" value="10" />
  <includePropertyFile location="otherVariableDefs.property" />
</environment>
```

5 Application Specification

The `<application>` element is the core of the descriptor, as it describes the application which will be run on a GCM-described grid. It contains a single child which type determines the kind of application that will be launched on the grid. The types currently defined are:

- 1) Executable.
- 2) ProActive.
- 3) GCM Fractal Component.
- 4) MPI.

5.1 Executable

An executable defines an application which is actually a simple command to be run.

It can have the following child elements:

- **nodeProvider** (empty element with a single "refId" attribute): the id of a node provider (defined in the <resources> part). There can be any number of such element;
- **command** (command): the command which will be run on the portion of the grid defined by the specified node providers.

This element can have the following attribute:

- **instances** (one of "onePerHost", "onePerVM", "onePerCapacity"): the number of instances of the command which will be run.

The <command> element can have the following children (in this specified order):

- **path** (path string): the path of the executable;
- **arg** (string): the arg string which will be passed to the command. There can be any number of such element;
- **filetransfer** (file transfer): the files which should be transferred prior to running the command.

It can have the following attribute:

- **name** (string): name of the executable. If a <path> child element is present, the value of this attribute will be appended to the value of the <path> child element.

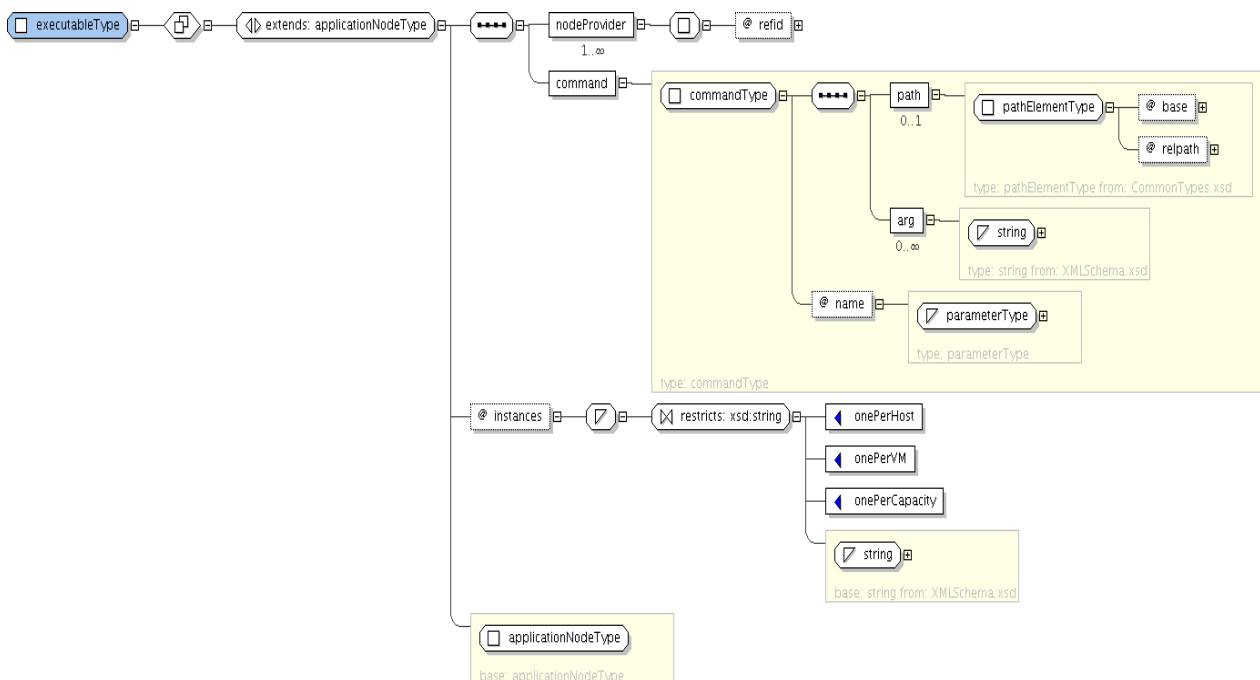


Figure 1

Example of a call to the "ls" Unix command:

```
<executable>
  <command name="ls">
    <arg>-l</arg>
    <arg>-h</arg>
    <arg>--sort=time</arg>
    <arg>-r</arg>
  </command>
  <nodeProvider refid="INRIA_LAN" />
</executable>
```

5.2 ProActive

5.2.1 Overall structure

The structure of a <proactive> element is as follows:

```
<proactive>
  <technicalServices>
  </technicalServices>

  <virtualNode>
    <nodeProvider/>
    ...
  </virtualNode>
  <virtualNode>
  ...
  </virtualNode>

  ...
</proactive>
```

5.2.2 ProActive definition

A <proactive> element can have the following children:

- **configuration:** various configuration parameters. This element can have the following children:
 - **bootClasspath** (simple classpath): the boot classpath for the JVM;
 - **java** (path string): the path to the Java executable;
 - **jvmarg** (string): arguments passed to the JVM;
 - **applicationClasspath** (classpath): classpath for the application;
 - **proactiveClasspath** (classpath): classpath used to override the standard ProActive classpath computed from its installation location;
 - **securityPolicy** (relative path): path to the Java security policy file;
 - **proactiveSecurity:** security policy for application and runtime. This element has two children:
 - **applicationPolicy** (relative path): path to Java security policy file that will be applied on the application's objects deployed at runtime, like nodes and active objects;
 - **runtimePolicy** (relative path): path to Java security policy file that will be applied on the ProActive Runtime.
 - **log4jProperties** (relative path): path to the Java log4j configuration file;
 - **userProperties** (relative path): path to the Java properties file;
 - **virtualNode** (virtual node): description of a virtual node. There can be any number of such element.
- **technicalServices** (technical services): the set of technical services global to this instance of ProActive. Technical Services are described in clause 5.2.3.

The <proactive> element can have the following attributes:

- **relpath** (path string): the location of the ProActive installation;
- **base** (one of "HOME", "ROOT"): base location of the ProActive installation: HOME is the user's home directory, ROOT is the root directory of the system.

A <virtualNode> element can have the following children:

- **nodeProvider** (reference to a node provider): the node provider which will provide the ProActive nodes for this virtual node - see below for description;
- **technicalServices** (technical service): a technical service specific to this virtual node. There can be any number of such children. Technical Services are described in clause 5.2.3.

A virtualNode element can also have the following attributes:

- **id** (string): a string identifying this virtual node;
- **capacity** (positive integer or "max"): the capacity requested by this virtual node (that is, the total number of nodes it will request from the node providers which are affected to it).

A <nodeProvider> within a <virtualNode> can only have <technicalServices> child elements (Technical Services are described in clause 5.2.3). These describe technical services specific to this node provider. A <nodeProvider> can also have the following attributes:

- **refid** (string): the id of the node provider (as defined in the resources element);
- **capacity** (positive integer or "max"): the capacity of this ProActive node provider (that is, the number of ProActive nodes which will be requested from it).

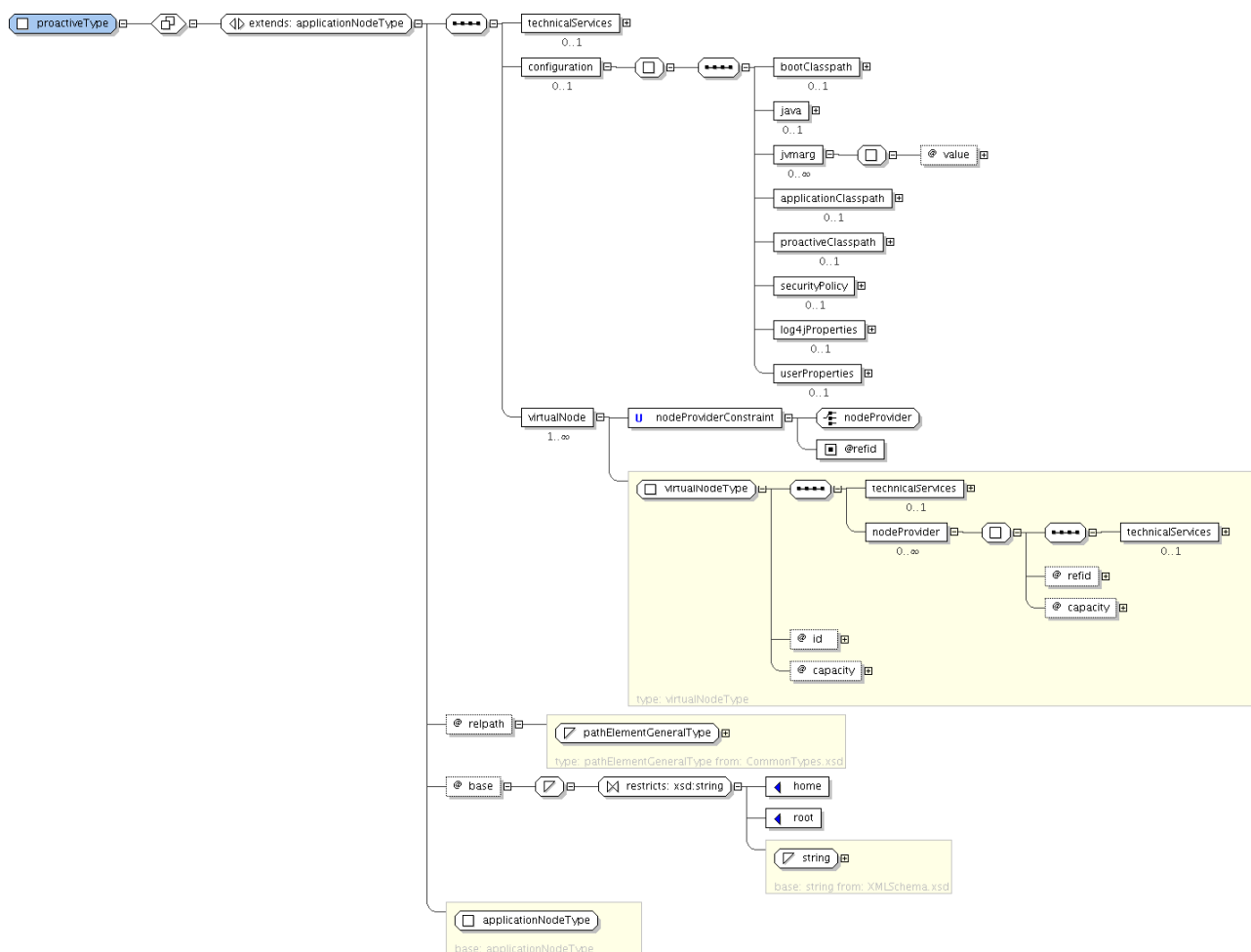


Figure 2

EXAMPLE:

```

<proactive base="root" relpath="/usr/local/proactive">
  <technicalServices>
    <class name="proactiveservices">
      <property name="myprop" value="avalue"/>
    </class>
    <class name="otherservices">
      <property name="x" value="12"/>
      <property name="y" value="15"/>
    </class>
  </technicalServices>

  <virtualNode id="master" capacity="1">
    <technicalServices>
      <class name="vnodeservices">
        <property name="myprop" value="avalue"/>
      </class>
    </technicalServices>

    <nodeProvider refid="INRIA_LAN">
      <technicalServices>
        <class name="nodeprovidertechservices">
          <property name="myprop" value="avalue"/>
        </class>
      </technicalServices>
    </nodeProvider>
  </virtualNode>

  <virtualNode id="slaves" capacity="max">
    <nodeProvider refid="INRIA_LAN"/>
    <nodeProvider refid="INRIA_CLUSTERS"/>
  </virtualNode>

```

```

<virtualNode id="slaves2" capacity="max">
  <nodeProvider refid="INRIA_LAN"/>
  <nodeProvider refid="INRIA_CLUSTERS"/>
</virtualNode>
</proactive>

```

5.2.3 Technical Services

A technical service is a non-functional requirement that may be dynamically fulfilled at runtime by adapting the configuration of selected resources. Such services are specified with a `<technicalServices>` element. This element holds a sequence of `<class>` children elements, which themselves hold a sequence of `<property>` elements.

A `<class>` element has a single attribute, "name", of type string. A `<property>` element has two attributes, "name" and "value", both of type string.

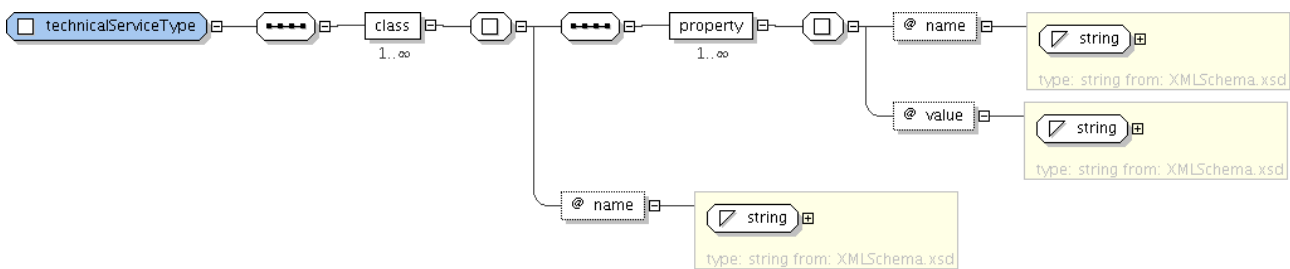


Figure 3

EXAMPLE:

```

<technicalServices>
  <class name="proactiveservice">
    <property name="myprop" value="avalue"/>
  </class>
  <class name="otherservice">
    <property name="x" value="12"/>
    <property name="y" value="15"/>
  </class>
</technicalServices>

```

Technical services can be defined at several levels of the ProActive definition:

- globally;
- for a given virtual node;
- for a given node provider for a virtual node.

This is better explained with the following skeleton:

```

<proactive>
  <technicalServices>
    <class name="proactiveFaultTolerance">
      <property name="enable" value="true" />
    </class>
  </technicalServices>

  <virtualNode>
    <technicalServices>
      <class name="...">
        <property name="..." value="..." />
      </class>
    </technicalServices>
  </virtualNode>
  <nodeProvider>
    <technicalServices>
      <class name="...">

```

```

        <property name="..." value="..." />
      </class>
    </technicalServices>
  </nodeProvider>
  ...
</virtualNode>
...
</proactive>

```

5.3 GCM Component

For further study.

5.4 MPI

For further study.

6 Resource Specification

The `<resources>` element has no attributes and only contains a sequence of `<nodeProvider>` elements. There can be any number of such elements.

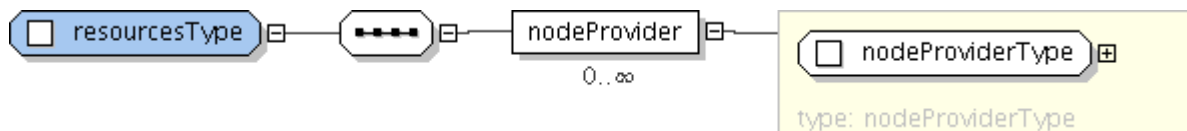


Figure 4

6.1 Node Provider

A `<nodeProvider>` can have the follow children:

- **file** (file): this element can only have a single attribute ("path") which contains the path to the deployment descriptor file which the node provider uses. There can be any number of such elements.

It can also have the following attribute:

- **id** (id): a string identifying the ProActive node provider.

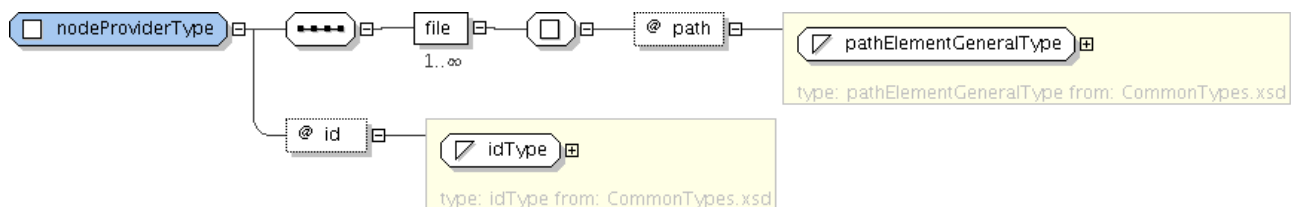


Figure 5

Annex A (normative): Schema

```

<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" targetNamespace="urn:gcm:application:1.0"
xmlns="urn:gcm:application:1.0"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="./CommonTypes.xsd" />

  <xsd:attribute name="OSAttr">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="Unix"></xsd:enumeration>
        <xsd:enumeration value="Windows"></xsd:enumeration>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>

  <xsd:attribute name="PriorityAttr">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="Low"></xsd:enumeration>
        <xsd:enumeration value="Normal"></xsd:enumeration>
        <xsd:enumeration value="High"></xsd:enumeration>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>

  <xsd:simpleType name="parameterType">
    <xsd:restriction base="xsd:string"></xsd:restriction>
  </xsd:simpleType>

  <xsd:complexType name="jreType">
    <xsd:sequence>
      <xsd:element name="parameter" type="parameterType" />
    </xsd:sequence>
    <xsd:attribute name="path" type="xsd:string"></xsd:attribute>
  </xsd:complexType>

  <xsd:complexType name="valueType">
    <xsd:attribute name="value" type="xsd:string" />
  </xsd:complexType>

  <xsd:complexType name="technicalServiceType">
    <xsd:sequence>
      <xsd:element name="class" maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="property" maxOccurs="unbounded">
              <xsd:complexType>
                <xsd:attribute name="name" type="xsd:string" use="required" />
                <xsd:attribute name="value" type="xsd:string" use="required" />
              </xsd:complexType>
            </xsd:element>
          </xsd:sequence>
          <xsd:attribute name="name" type="xsd:string" use="required" />
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="nodeProviderType">
    <xsd:sequence>
      <xsd:element name="file" maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:attribute name="path" type="pathElementGeneralType"></xsd:attribute>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="id" type="idType"></xsd:attribute>
  </xsd:complexType>

```

```

<xsd:complexType name="virtualNodeType">
  <xsd:sequence>
    <xsd:element name="technicalServices" type="technicalServiceType" minOccurs="0" />
    <xsd:element name="nodeProvider" minOccurs="0" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="technicalServices" type="technicalServiceType"
minOccurs="0" />
        </xsd:sequence>
        <xsd:attribute name="refid" type="idType" use="required" />
        <xsd:attribute name="capacity" type="capacityType" />
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:string" use="required"></xsd:attribute>
  <xsd:attribute name="capacity" type="capacityType"></xsd:attribute>
</xsd:complexType>

<xsd:complexType name="commandType">
  <xsd:sequence>
    <xsd:element name="path" type="pathElementType" minOccurs="0" />
    <xsd:element name="arg" type="xsd:string" minOccurs="0" maxOccurs="unbounded" />
  </xsd:sequence>
  <xsd:attribute name="name" type="parameterType"></xsd:attribute>
</xsd:complexType>

<xsd:complexType name="applicationNodeType" abstract="true"></xsd:complexType>

<xsd:complexType name="proactiveType">
  <xsd:complexContent>
    <xsd:extension base="applicationNodeType">
      <xsd:sequence>
        <xsd:element name="technicalServices" type="technicalServiceType" minOccurs="0"
/>
        <xsd:element name="configuration" minOccurs="0">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="bootClasspath" type="pathType" minOccurs="0" />
              <xsd:element name="java" type="pathElementType" minOccurs="0"
maxOccurs="1" />
              <xsd:element name="jvmarg" minOccurs="0" maxOccurs="unbounded">
                <xsd:complexType>
                  <xsd:attribute name="value" type="xsd:string" />
                </xsd:complexType>
              </xsd:element>
              <xsd:element name="applicationClasspath" type="classpathType"
minOccurs="0" maxOccurs="1" />
              <xsd:element name="proactiveClasspath" type="classpathType"
minOccurs="0" maxOccurs="1" />
              <xsd:element name="securityPolicy" type="pathElementType"
minOccurs="0" maxOccurs="1" />
              <xsd:element name="log4jProperties" type="pathElementType"
minOccurs="0" maxOccurs="1" />
              <xsd:element name="userProperties" type="pathElementType"
minOccurs="0" maxOccurs="1" />
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="virtualNode" type="virtualNodeType" maxOccurs="unbounded">
          <xsd:unique name="nodeProviderConstraint">
            <xsd:selector xpath="nodeProvider"></xsd:selector>
            <xsd:field xpath="@refid"></xsd:field>
          </xsd:unique>
        </xsd:element>
      </xsd:sequence>
      <xsd:attribute name="relpath" type="pathElementGeneralType" use="required" />
      <xsd:attribute name="base" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="home" />
            <xsd:enumeration value="root" />
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:extension>

```

```

    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="executableType">
    <xsd:complexContent>
      <xsd:extension base="applicationNodeType">
        <xsd:sequence>
          <xsd:element name="command" type="commandType"></xsd:element>
          <xsd:element name="nodeProvider" maxOccurs="unbounded">
            <xsd:complexType>
              <xsd:attribute name="refid" type="xsd:string" use="required" />
            </xsd:complexType>
          </xsd:element>
        </xsd:sequence>
        <xsd:attribute name="instances">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:enumeration value="onePerHost"></xsd:enumeration>
              <xsd:enumeration value="onePerVM"></xsd:enumeration>
              <xsd:enumeration value="onePerCapacity"></xsd:enumeration>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:attribute>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="abstractApplicationElement" type="applicationNodeType" />

  <xsd:element name="executable" type="executableType"
  substitutionGroup="abstractApplicationElement" />
  <xsd:element name="proactive" type="proactiveType"
  substitutionGroup="abstractApplicationElement" />

  <xsd:complexType name="applicationType">
    <xsd:sequence>
      <xsd:element ref="abstractApplicationElement" maxOccurs="1" minOccurs="1"></xsd:element>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="resourcesType">
    <xsd:sequence>
      <xsd:element name="nodeProvider" type="nodeProviderType" minOccurs="0"
  maxOccurs="unbounded"></xsd:element>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="GCMAApplicationType">
    <xsd:sequence>
      <xsd:element name="environment" type="environmentType" minOccurs="0" />
      <xsd:element name="application" type="applicationType" minOccurs="1"
  maxOccurs="1"></xsd:element>
      <xsd:element name="resources" type="resourcesType" minOccurs="1"
  maxOccurs="1"></xsd:element>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:element name="GCMApplication" type="GCMAApplicationType">
    <!-- resource providers refid/id key correctness -->
    <!-- -->
    <xsd:key name="idnodeProviders">
      <xsd:selector xpath="resources/nodeProvider" />
      <xsd:field xpath="@id" />
    </xsd:key>
    <xsd:keyref name="refIdKeynodeProvider" refer="idnodeProviders">
      <xsd:selector xpath="application/executable/nodeProvider"></xsd:selector>
      <xsd:field xpath="@refid"></xsd:field>
    </xsd:keyref>
    <xsd:keyref name="refIdKeynodeProvider2" refer="idnodeProviders">
      <xsd:selector xpath="application/proactive/virtualNode/nodeProvider"></xsd:selector>
      <xsd:field xpath="@refid"></xsd:field>
    </xsd:keyref>
  </xsd:element>

</xsd:schema>

```


Annex B (informative): Example of applicationdescriptor

```

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

<GCMAApplication
  xmlns="urn:gcm:application:1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www-sop.inria.fr/oasis/ProActive/schemas
http://proactive.inria.fr/schemas/gcm/1.0/ApplicationDescriptorSchema.xsd">

  <application>
    <proactive relpath="Scratch/ProActive/">

      <configuration>
        <java relpath="myApps/Java/bin/java" />
        <proactiveClasspath>
          <pathElement base="proactive"
            relpath="lib/log4j.jar" />
          <pathElement base="proactive"
            relpath="lib/fractal.jar" />
          <pathElement base="home"
            relpath="myApps/ganymed-ssh.jar" />
        </proactiveClasspath>
        <applicationClasspath>
          <pathElement relpath="myApps/nqueens.jar" />
        </applicationClasspath>
        <securityPolicy base="proactive"
          relpath="config/security.policy" />
        <log4jProperties base="home" relpath="log4j.properties" />
      </configuration>

      <virtualNode id="master" capacity="1">
        <nodeProvider refid="COMPANY_LAN" />
      </virtualNode>

      <virtualNode id="slaves" capacity="max">
        <nodeProvider refid="COMPANY_LAN" />
      </virtualNode>
    </proactive>
  </application>

  <resources>
    <nodeProvider id="COMPANY_LAN">
      <file path="deployment.xml" />
    </nodeProvider>
  </resources>

</GCMAApplication>

```

Annex C (informative): Bibliography

- Programming, Composing, Deploying for the Grid Baude F., Baduel L., Caromel D., Contes A., Huet F., Morel M. and Quilici R., in "GRID COMPUTING: Software Environments and Tools", Jose C. Cunha and Omer F. Rana (Eds), Springer Verlag, January 2006
- Snir, Marc; Otto, Steve; Huss-Lederman, Steven; Walker, David; Dongarra, Jack (1995) "MPI: The Complete Reference" MIT Press Cambridge, MA, USA. ISBN 0-262-69215-5
- Gropp, William; Lusk, Ewing; Skjellum, Anthony. (1999) Using MPI, 2nd Edition: portable Parallel Programming with the Message Passing Interface. MIT Press In Scientific And Engineering Computation Series, Cambridge, MA, USA. 395 pp. ISBN 978-0-262-57132-6
- E. Bruneton, T. Coupaye, M. Leclercq, V. Quéma, and J.-B. Stefani. The Fractal Component Model and Its Support in Java . Software Practice and Experience, special issue on Experiences with Auto-adaptive and Reconfigurable Systems. 36(11-12), 2006.
- E. Bruneton, T. Coupaye, M. Leclercq, V. Quéma, and J.-B. Stefani. An Open Component Model and Its Support in Java . Seventh International Symposium on Component-Based Software Engineering (CBSE-7). LNCS 3054, pp. 7-22, May 2004.

History

Document history		
V1.1.1	August 2008	Publication