GWD-R-P.203
JSDL-WG

Philipp Wieder, GWDG
Alexander Papaspyrou, Adesso AG
Andreas Savva, Fujitsu Laboratories Ltd
Donal Fellows, The University of Manchester
Shahbaz Memon, Juelich Supercomputing Centre
May 06, 2013

Activity Instance Container Specification
Version 1.0

Status of This Document

This document provides information to the Grid community regarding the specification of the Activity Instance Container. Distribution is unlimited.

Copyright Notice

Abstract

The purpose of this document is to specify the syntax and semantics of the Activity Instance Container. An *activity*, in this context, is a unit of work to be executed on a distributed system. It can be a job, a task, a data processing operation, a data access operation, an application execution, a program execution, or a Web Service invocation. The Activity Instance Container captures the information related to this unit of work. Systems, which are designed based on this specification, can provide a holistic view of an activity, for example for monitoring, auditing or check-pointing purposes. The specification pursues two different but complementary objectives: (i) it specifies a generic and extensible framework to capture the information related to an activity; and (ii) it specializes this generic framework using a number of Open Grid Forum specifications.

Contents

1. Introduction

The Activity Instance Container captures the information related to an *activity* – a unit of work processed within a distributed system. This description can contain whatever information is relevant to the application domain of the activity. This might be execution-related status data, quality-of-service information, or error messages. All this is captured in an activity instance container and therefore systems built based on the Activity Instance Container specification can provide a holistic view of an activity, for example for monitoring, auditing, or check-pointing purposes.

This document standardizes the description of an activity instance container and, to this extend, syntactically and semantically defines the elements of it. To achieve this, this document pursues two different but complementary objectives: (i) it specifies a generic and extensible framework to capture the information related to an activity (see Section 4); and (ii) it specializes this generic framework using a number of Open Grid Forum (proposed) recommendations (see Section 5). However, this specification does not define how to create an activity instance and how to manage it, as this is implementation-specific.

1.1 Motivation

Information related to an *activity*, such as resource usage, security data, activity state, or data requirements, is captured throughout the lifecycle of an activity using a variety of schemata. Furthermore, such information is stored in different ways and by different logical components. This dispersion of activity-related information leads to management, security, and logistical overhead in discovering, accessing, and using that information. Moreover, it results in an environment where activity information is managed by many systems.

The Job Specification Description Language [JSDL], for example, comprises a core Resource Request Language (RRL) and exists, as pictured in Figure 1, within an environment of other languages like a Scheduling Description Language (SDL), WS-Agreement (WS-AG) [WSAG], a Job Policy Language (JPL), and potentially many more. The activity instance can be used in this context to keep track all of the information related to a job described in JSDL (i.e. the activity) and trace historic information.

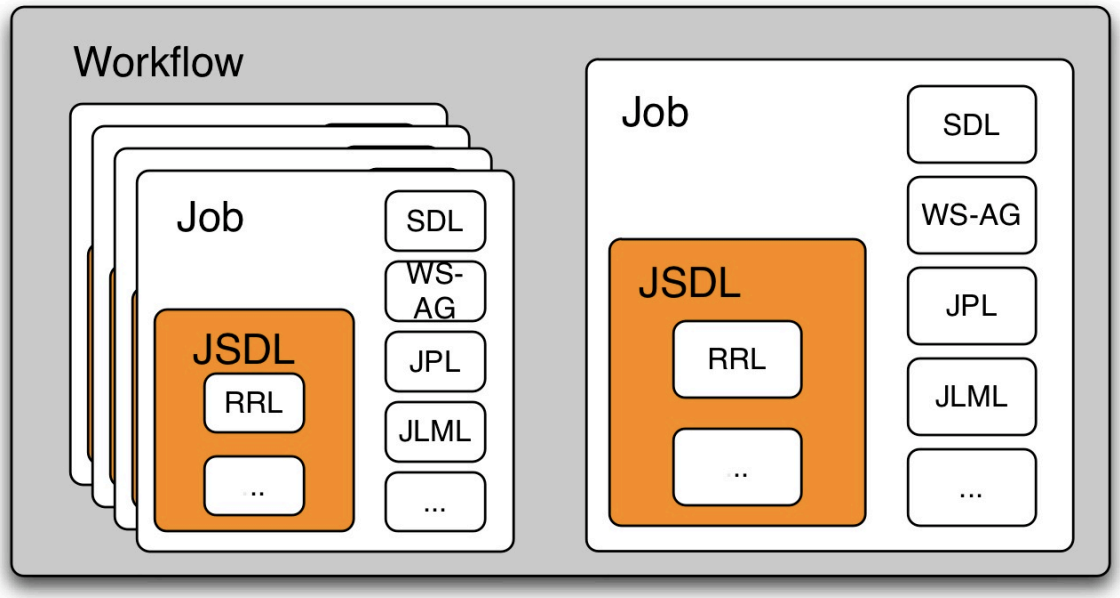

Figure 1. Relationship between JSDL and other specifications **[JSDL]**

1.2 Definition of the Term Activity within the Scope of this Document

An *activity* is a unit of work to be executed on a distributed system. It can be a job, a task, a data processing operation, a data access operation, an application execution, a program execution, a Web Service invocation, or something else that a user or application needs to do, take care of, or execute.

An *activity* is atomic. This means that an activity is an indivisible unit of work from an activity management perspective. When you stop an activity, you stop all of it, not some part of it. Moreover, activities can be composed together to form chains of activities that may be managed conditionally, sequentially, or in parallel. In other words, activities can be the atomic nodes in a workflow. Furthermore, they can be conditionally used to process data in a data centric process.

1.3 Information potentially captured by an Activity

The specification provided by this document comprises activity-related information gathered from a number of use cases. Since the Activity Instance Container can potentially capture all information related to an activity and since the kind of information depends on the application domain, the specification includes a number of extension points.

These extension points may be used to add information like the following:

- Dependencies on data and other activities for the composition and management of activities for workflow, scheduling and brokering processes
- Contextual information, such as:
  - Domain-specific (like for financial markets, weather forecasting, etc.),
  - Security-related (regarding the owner of an activity, the manager of a service, etc.), or
  - SLAs, QoS and other policies.
- Monitoring information, such as:
  - accounting or
  - policy conformance.

1.4 Motivating Use Case – Delegating Activities between Schedulers

This non-normative use case is included in this document to provide an example of an activity and the steps involved in processing it. The use case also serves as the source for the examples given in the normative sections that specify the XML representation of the Activity Instance Container.

1.4.1 Actors

The following actors are involved in the delegation use case as shown in Figure 2:

- The *client*, which can be a user accessing the *primary scheduler* directly or a component doing it on the user's behalf
- The *primary scheduler* is the entity that receives the activity template and generates the activity instance
- Depending on the outcome of the different delegations, one or more *secondary schedulers* to which the Activity is delegated
- The *Basic Execution Service* (BES) [BES] represents the work unit's execution endpoint. Executes the job related to the activity
- The *activity store* is a potentially distributed instance where all information about the activity is stored

1.4.2 Activity Flow

A client sends an activity request to a scheduler describing the requirements of the submitted work unit[1].

[1] Note that the activity request can be submitted in any supported format, for example JSDL. In general, the client does not have to be aware of the concept of an activity instance as specified

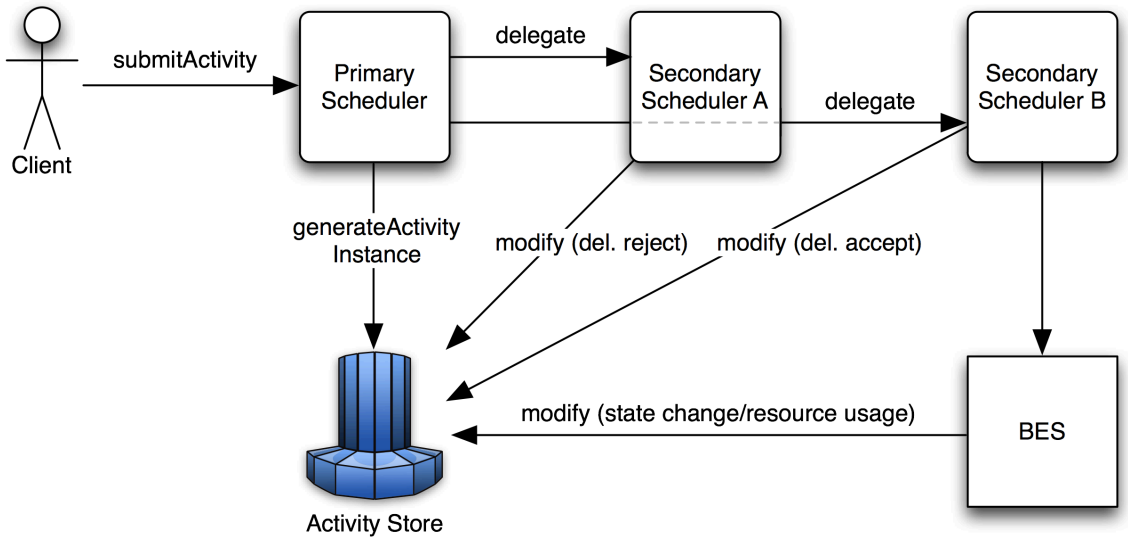

Figure 2. Delegating Activities between Schedulers

The initially receiving (primary) scheduler takes the template and, if it is willing to handle it, creates an activity instance container for it, storing the initial activity request and, if applicable, additional information. The latter should at least include a provenance record, which denotes that the current scheduler has taken over responsibility for the execution of the given activity. Other information may include scheduling attributes, dependencies on other activities, and the current state of the activity.

On activity delegation, the delegator acts like a client towards the potential delegate, offering the job to another scheduler. Again, if the delegate is willing to accept the job, it takes over responsibility and the provenance records and depending information (e.g. the expected BES container) are updated. If necessary, the activity template is modified, as long as the manipulation history is kept. Such modifications may include, as depicted in Figure 2, whether a secondary scheduler rejects or accepts a delegation request, the state transition of the activity, or the resources consumed.

Throughout the whole process, state information is constantly updated. After activity completion, the resource consumption is written to the activity instance container. The corresponding entries and dependent parts of the activity instance could then be marked final to denote the completion of the activity.

1.4.3 Sequence Diagram

Figure 3 shows the sequence of messages occurring in the example use case according to Figure 2. The primary scheduler accepts the activity and, since it cannot process it (the reason for which is not of interest), delegates it first to the secondary scheduler A, which rejects the request, and then to secondary scheduler B, which accepts the request, schedules it and hands it over to the BES container for execution. During this process, the following (asynchronous) messages are sent to the activity store:

- The primary scheduler informs the activity store about the new activity (following the acceptance of the activity request) which results in the generation of an Activity Instance Container

---

by this document at all. To this end, the term activity is used in two different ways: the unit of work submitted by the client (an arbitrary unit of work or activity) and the activity instance compliant with the activity instance description specified within this document.

- The secondary scheduler A informs the activity store about it rejecting the delegation request (using a modification message)
- The secondary scheduler B informs the activity store about the acceptance of the delegation request
- The secondary scheduler B notifies the activity store about the hand-over of the activity to the BES container
- The BES container modifies the Activity Instance Container according to the state of the activity's execution, informs it about the resources used, etc.

The use case shows, for the sake of simplicity, only the steps until the activity is executed. Further steps that occur during the processing of the activity, like feed-back of results, are not show, nor are potential activity monitoring or assessment steps shown.

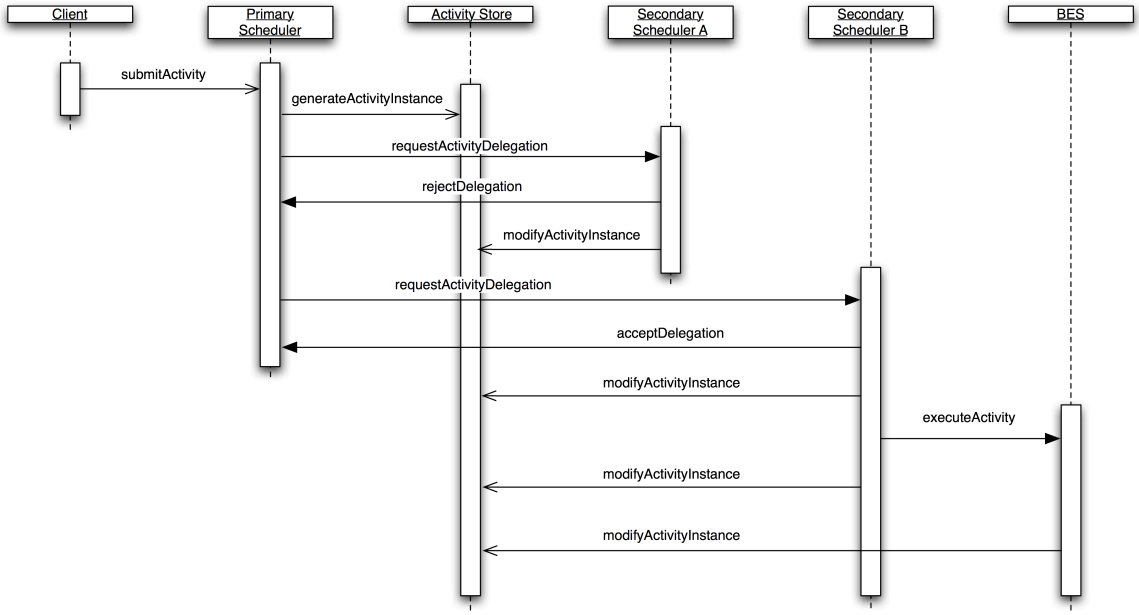

Figure 3. Sequence Diagram of the Activity Delegation Use Case

1.5 Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119 [BRADNER].

This document describes XML Information Sets and inherits the square bracket notation of [INFOSET].

When describing concrete XML schemas [SCHEMA1], [SCHEMA2], this specification uses the notational convention of WS-Security
[WSSEC]. Specifically, each member of an element's [children] or [attributes] properties, is described using an XPath-like notation (e.g.: **/x:MyHeader/x:SomeProperty/@value1**). The use of **{any}** indicates the presence of an element wildcard (<*xsd:any/*>). The use of **@{any}** indicates the presence of an attribute wildcard (<*xsd:anyAttribute/*>).

Pseudo-schemas are provided for each component, before the description of the component. They use BNF-style conventions for attributes and elements: '?' denotes zero or one occurrences; '*' denotes zero or more occurrences; '+' denotes one or more occurrences. Attributes (other than the *abstract* and *substitutes* special attributes) are conventionally assigned

a value that corresponds to their type, as defined in the normative schema.

```
<!-- sample pseudo-schema -->
<defined_element>
      required_attribute_of_type_string="xsd:string"
      optional_attribute_of_type_int="xsd:int"? >
   <required_element />
   <optional_element />?
   <one_or_more_of_this_element />+
</defined_element>
```

1.6 Namespaces

This specification uses namespace prefixes throughout; they are listed in Table 1-1. Note that the choice of any namespace prefix is arbitrary and not semantically significant.

Table 1-1. Prefixes and namespaces used in this specification

Prefix	Namespace
aic	http://schemas.ogf.org/jsdl/2012/12/activity-instance-container
aic-ogf	http://schemas.ogf.org/jsdl/2010/06/activity-instance-container-ogf
bes-factory	http://schemas.ggf.org/bes/2006/08/bes-factory
jsdl	http://schemas.ggf.org/jsdl/2005/11/jsdl
jsdl-posix	http://schemas.ggf.org/jsdl/2005/11/jsdl-posix
ur	http://schemas.ogf.org/urf/2003/09/urf
wsa	http://www.w3.org/2005/08/addressing
xsd	http://www.w3.org/2001/XMLSchema
xsi	http://www.w3.org/2001/XMLSchema-instance

2. Scope of the Specification

The Activity Instance Container specification defines the overall structure of the activity instance container and the semantics associated with each element. It is expected that a number of other specifications will provide the detailed information contained in each container element, depending on the domain in which the Activity Instance Container specification is used. An overview of specifications from the Open Grid Forum that are used with this specification is given here.

2.1 On Basic Execution Service

The Basic Execution Service (BES) [BES] defines a service to which clients can send requests to initiate, monitor, and manage computational activities. In addition to an information model and a set of port types, the BES specification also defines an extensible state model for activities. The Activity Instance Container specification uses the BES state model to record the various states the activity goes through during its lifetime.

2.2 On Job Submission Description Language

The Job Submission Description Language (JSDL) [JSDL] is a language for describing the requirements of computational jobs for submission to resources, particularly in Grid environments where interactions between a number of different types of job management systems is common. A JSDL document may be transformed by intermediaries or refined further by information not available to the initial submitter of that job. Therefore in the context of the Activity Instance Container specification, JSDL can be used to describe both the initial submission that created the activity instance as well as the results of transformations to the activity definition due to processing steps like delegation or negotiation.

2.3 On Usage Records

The Usage Record (UR) [UR] is an XML document language for describing units of accounting system data (e.g., batch scheduler log entries) in an interoperable exchange format. Usage records are focused on describing what a particular unit of work (e.g., computational job) actually consumed in terms of resources, and are used on Grids particularly when the organization that actually carried out the job is different from the organization that is paying for it, and are already widely used in production Grid deployments, such as TeraGrid and DEISA. Within the context of the Activity Instance Container specification, URs can be used to describe the resources (both hardware and software) that have been actually consumed/used by the activity instance over time, as well as sample points of current allocation levels, allowing both system monitoring and accounting within the context of the overall activity lifecycle.

3. Activity Instance Container Structure

An Activity Instance Container is organized as follows: The root element, *ActivityInstanceDescription,* contains an optional human readable description of the activity (i.e. the *ActivityDescription*); and a sequence of one or more history entries, i.e. a list of *ActivityHistoryEntry* elements. Each *HistoryEntry* element contains information about the activity at a specific point in time: status, definition, dependencies, reference to the activity's manager, and resource usage. The pseudo schema definition is given below.

```
<ActivityInstanceDescription>
   <ActivityDescription .../>?
   <ActivityHistory .../>
      <ActivityHistoryEntry>*
         <Status ...>
         <ActivityDefinition ... />?
         <ActivityDependency ... />*
         <ManagerReference .../>
         <ResourceUsage .../>?
         <xsd:any##other/>*
      </ActivityHistoryEntry>
   <xsd:any##other/>*
</ActivityInstanceDescription>
```

Normative definitions of the Activity Instance Container elements are given in Sections 4 and 5. The Activity Instance Container XML schema is listed in Appendix A, examples of Activity Instance Descriptions are given in Appendix B, Appendix C, and Appendix D. The examples given in the following sections are non-normative and are solely added for illustrative purposes.

The Activity Instance Container schema supports an open content model. Entities accessing information contained in Activity Instance Containers MAY not necessarily recognize all the extensions used. Implementations SHOULD ignore all extensions they do not support.

4. The Abstract Activity Instance Container Element Set

4.1 The ActivityInstanceDescription Element

4.1.1 Definition

This element is the root of a single Activity Instance Container, which contains an activity's meta-data and history, and which provides the entry point for every activity. While the meta-data part MAY carry information about the activity's creator, purpose, and further references (i.e. to other activities), the history part SHOULD describe the full lifecycle of the activity.

4.1.2 Multiplicity

The multiplicity of this element is one.

4.1.3 Type

This is an `xsd:complexType`. It MUST support the following elements:

- *ActivityDescription* (cf. 4.2)
- *ActivityHistory* (cf. 4.3)

4.1.4 Attributes

The following attributes are defined:

- *id* - An identifier for the activity, which MUST be globally unique. It is RECOMMENDED to use Universally Unique Identifiers (UUID) as described in RFC4122 [UUID].

4.1.5 XML Representation

The *ActivityInstanceDescription* is rendered in XML as:

```
<ActivityInstanceDescription id=”xsd:string” xsd:any##other*>
   <ActivityDescription ... />?
   <ActivityHistory ... />
   <xsd:any##other/>*
</ActivityInstanceDescription:>
```

Where:

/aid: ActivityInstanceDescription
Represents the *ActivityInstanceDescription* element.

/aid: ActivityInstanceDescription/ActivityDescription
Represents the *ActivityDescription* element as defined in Section 4.2. This element MAY appear zero or one time.

/aid: ActivityInstanceDescription/ActivityHistory
Represents the *ActivityHistory* element as defined in Section 4.3. This element MUST appear exactly once.

4.1.6 Example

The following example shows the *ActivityInstanceDescription* element, which carries an id attribute following the UUID specification. Examples for ActivityDescription and ActivityHistory are given in Section 4.2 and Section 4.3, respectively.

```
<ActivityInstanceDescription id=”ea196512-9cb7-4a14-91b0-2dde749a5f7d”>
   <ActivityDescription> ... </ActivityDescription>
   <ActivityHistory> ... </ActivityHistory>
</ActivityInstanceDescription>
```

4.1.7 Extensibility

This element provides two extension points, one for element extensibility and one for attribute extensibility. The former extension point MAY be used to attach XML digital signatures [XMLDSIG] to the container (see Section 7). Implementations SHOULD ignore unsupported extensions.

4.2 The ActivityDescription Element

4.2.1 Definition

This element contains a natural-language description of the activity and offers means for storing additional information on the activity for displaying purposes (e.g. in a user interface).

4.2.2 Multiplicity

The multiplicity of this element is zero or one.

4.2.3 Type

This type of this element is `xsd:string`.

4.2.4 Attributes

No attributes are defined.

4.2.5 Pseudo schema

The *ActivityDescription* is rendered in XML as:

```
<ActivityDescription> xsd:string </ActivityDescription>
```

Where:

/aid: ActivityInstanceDescription/ActivityDescription
Represents the *ActivityDescription* element.

4.2.6 Example

```
<ActivityDescription>
   This activity instance has been generated due to an activity request
   submitted to the scheduling service with the
   following URI: http://tempuri.org/services/activityscheduler. The
   activity request has been received at 2010-05-10T11:11:11.11. The
   activity instance has been created 2010-05-10T11:11:44.44 by the
   organization's activity store with the following URI:
   http://tempuri.org/services/activitystore.
</ActivityDescription>
```

The example shows a human-readable description of an activity following the example given in Section 1.4. There, an activity request is accepted by a scheduler for processing, which then triggers the creation of an activity instance.

4.2.7 Extensibility

This element provides no extension points.

4.3 The ActivityHistory Element

4.3.1 Definition

This element keeps a record of the history of an activity. This record MUST contain one initial and, at most, one final record (see Section 4.4 for the *ActivityHistoryEntry* element and for the

different category attributes). Note that, although a final record MAY be written already, the activity itself MAY still be modified.

4.3.2 Multiplicity

The multiplicity of this element is one.

4.3.3 Type

This is an `xsd:complexType`. It MUST support the following elements:

- *ActivityHistoryEntry*

4.3.4 Attributes

No attributes are defined.

4.3.5 Pseudo schema

The *ActivityHistory* is rendered in XML as:

```
<ActivityHistory>
   <ActivityHistoryEntry/>*
</ActivityHistory>
```

Where:

/aid: ActivityInstanceDescription/ActivityHistory
: Represents the *ActivityHistory* element.

/aid: ActivityInstanceDescription:/ActivityHistory/ActivityHistoryEntry
: Represents the *ActivityHistoryEntry* element as introduced in Section 4.4. This element MAY appear zero or one time. within the *ActivityHistory* element.

4.3.6 Example

```
<ActivityHistory>
   <ActivityHistoryEntry> ... </ActivityHistoryEntry>
   <ActivityHistoryEntry> ... </ActivityHistoryEntry>
</ActivityHistory>
```

This example shows a history of an activity with currently only two entries. Following the example in Section 1.4, the history would contain exactly two entries after the first delegation attempt of the primary scheduler to secondary scheduler A. An example for the *ActivityHistoryEntry* is given in Section 4.4.

4.3.7 Extensibility

This element provides no extension points.

4.4 The ActivityHistoryEntry Element

4.4.1 Definition

This element stores a single event in an activity's history. It denotes an event in the history of an activity, containing its properties at the time the event occurred. Every entry MUST contain at least a timestamp (as attribute), the status of the activity at this timestamp, and a WS-Addressing [WSADDR] endpoint reference to the managing service. Once an *ActivityHistoryEntry* is written, it MUST NOT be altered. Additional information about the respective activity has to be appended to the ActivityHistory by adding a new *ActivityHistoryEntry* element.

4.4.2 Multiplicity

The multiplicity of this element is zero or more.

4.4.3 Type

This is an `xsd:complexType`. It MUST support the following elements:

- *Status*
- *Event*
- *ActivityDefinition*
- *ActivityDependency*
- *ManagerReference*
- *ResourceUsage*

4.4.4 Attributes

The following attributes are defined:

- *timestamp* – the timestamp of the entry. It MUST appear exactly once. Its type is `xsd:dateTime`. It keeps the timestamp at which this event has occurred in the activity's history. The entries in the whole activity history SHOULD be ordered ascending to their timestamp.
- *category* – the category of an entry. The attribute MAY appear zero or one time. Its type is `aid:ActivityHistoryEntryCategoryType`. Denotes the category of this history record. The possible options are "initial", "intermediate", and “final”. Note that there MUST exist at least an initial and a final entry to describe the entire lifecycle of an activity. The semantics of the options is as follows:
  - *initial* denotes the initial history record for a given activity, which MUST be the first one created in the whole record. Note that this is not necessarily the first action taken on a certain activity instance; which events are to be recorded as a history record is implementation-specific. The initial ActivityHistoryEntry MUST NOT have an empty ActivityDefinition element.
  - *intermediate* denotes an intermediate history record for a given activity. Such entry MAY NOT be the first or last one created in the whole record.
  - *final* denotes the final history record for a given activity, which MUST be the last one created in the whole record. Note that this does not imply that the activity as a whole MAY NOT be modified any more.

4.4.5 Pseudo schema

The *ActivityHistoryEntry* is rendered in XML as:

```
<ActivityHistoryEntry timestamp=”xsd:dateTime
                      category=”ActivityHistoryEntryCategoryType”?
                      xsd:any##other*>
   <Status/>
   <ActivityDefinition/>?
```

```
   <ActivityDependency/>*
   <ManagerReference/>?
   <ResourceUsage/>?
   <Event>?
</ActivityHistoryEntry>
```

Where:

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry
Represents the *ActivityHistoryEntry* element.

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/Status
Represents the *Status* element as defined in Section 4.5. It MUST be present exactly once.

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/Event
Represents the *Event* element as defined in Section 4.57. It is an optional element, which
MAY appear zero or one times.

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/ActivityDefinition
Represents the *ActivityDefinition* element as defined in Section 4.9. It is an optional element, which MAY appear zero or one times.

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/ActivityDependency
Represents the *ActivityDependency* element as defined in Section 4.10. It is an optional element, which MAY appear zero or more times.

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/ManagerReference
Represents the *ManagerReference* element as defined in Section 4.11. It is an optional element, which MAY appear zero or one times.

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/ResourceUsage
Represents the *ResourceUsage* element as defined in Section 4.12. It is an optional element, which MAY appear zero or one times.

4.4.6 Example

```
<ActivityHistoryEntry timestamp="2010-05-10T11:11:44.44"
                      category="initial">
   <Status> ... </Status>
   <Event> ... </Event>?
   <ActivityDefinition> ... </ActivityDefinition>
   <ActivityDependency> ... </ActivityDependency>
   <ManagerReference> ... </ManagerReference>?
   <ResourceUsage> ... </ResourceUsage>
</ActivityHistoryEntry>
```

This example shows the *timestamp* and *category* attributes for the initially generated activity instance according to the example given in Section 1.4. All the other elements have examples in the respective sections below.

4.4.7 Extensibility

This element provides two extension points, one for element extensibility and one for attribute extensibility. The former extension point MAY be used to attach XML digital signatures [XMLDSIG] to a container (see Section 7). Implementations SHOULD ignore unsupported extensions.

4.5 The Status Element

4.5.1 Definition

This element stores the status of the activity with respect to the enclosing history record. The status of an activity comprises its current state (defined by an appropriate state model) and, if necessary, information. Every status record for an activity MUST contain at least the current state; if an exceptional condition occurs during the activity's lifetime, it SHOULD be also recorded here. Note that the existence of an exception entry is not necessarily coupled to a corresponding exceptional state; a possible connection between these is left to the implementor and SHOULD be described in the concrete state model's documentation.

4.5.2 Multiplicity

The multiplicity of this element is one.

4.5.3 Type

This is an `xsd:complexType`. It MUST support the following elements:

- *State*
- *Exception*

4.5.4 Attributes

No attributes are defined.

4.5.5 Pseudo schema

The *Status* is rendered in XML as:

```
<Status>
   <State/>
   <Exception/>?
</Status>
```

Where:

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/Status
Represents the *Status* element.

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/Status/State
Represents the *State* element as defined in Section 0. It is a mandatory element, which MUST appear exactly once.

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/Exception
Represents the *Exception* element as defined in Section 4.8. It is an optional element, which MAY appear zero or one time.

4.5.6 Example

```
<Status>
   <State> ... </State>
   <Exception> ... </Exception>
</Status>
```

An example for *State* is given in Section 5.3 while Section 6.1 features an example for the *Exception* element.

4.5.7 Extensibility

This element provides no extension points

4.6 The State Element

4.6.1 Definition

This element denotes details on the state of the activity with respect to the enclosing history record. More specifically, it stores a state model instance for the given activity state.

4.6.2 Multiplicity

The multiplicity of this element is one.

4.6.3 Type

This is an abstract type and has to be substituted by an appropriate definition (see Section 5.3).

4.6.4 Attributes

No attributes are defined.

4.6.5 Pseudo schema

The abstract *State* has no rendering:

```
<State abstract=”true”/>
```

Where:

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/Status/State

Represents the *State* element. Since the *State* element is an abstract XML element it MUST NOT occur in a XML document by itself. It MUST be substituted by a valid *State* substituent instead.

4.6.6 Extensibility

States are specific to activities. The minimal definition provided in Section 5.3 SHOULD be used if there is no applicable specification.

4.7 The Event Element

This element encapsulates an additional information on a particular state of an activity.

4.7.1 Definition

The Event element denotes an additional information relating to an event occurred within a specific activity state. It is useful if an entity responsible of managing ActivityHistory, is expected to provide more information about the activity's state rather than only the status and timestamp attributes. This will help ActivityInstanceDescription consumers, such as users or client applications to better analyze activity runs or failures during the activity lifecycle. A more specific example is, when an activity is failed due to a staging-in failure, in this case the error details will be captured in an Event instance.

4.7.2 Multiplicity

The multiplicity of this element is zero or one.

4.7.3 Type

This element is of type s*tring*.

4.7.4 Attributes

No attributes are defined.

4.7.5 Pseudo schema

The Event element is rendered in XML as,

```
<Event>....</Event>
```

Where:
/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/Event
Represents the *Event* element.

4.7.6 Extensibility

The Event element has no extensibility points.

4.8 The Exception Element

4.8.1 Definition

This element denotes details on an exception for the activity with respect to the enclosing record. More specifically, it stores an exception model instance for the given activity fault.

4.8.2 Multiplicity

The multiplicity of this element is zero or one.

4.8.3 Type

This is an abstract type and has to be substituted by an appropriate definition (see Section 6.1).

4.8.4 Attributes

No attributes are defined.

4.8.5 Pseudo schema

The abstract *Exception* has no rendering:

```
<Exception abstract="true"/>
```

Where:
/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/Status/Exception

Represents the *Exception* element. Since the *Exception* element is an abstract XML element it MUST NOT occur in a XML document by itself. It MUST be substituted by a valid *Exception* substituent instead.

4.8.6 Extensibility

Exceptions are specific to activities. The minimal definition provided in Section 6.1 SHOULD be used if there is no applicable specification. Other possible substitutions may include, for example, SOAP faults.

4.9 The ActivityDefinition Element

4.9.1 Definition

This element stores the definition of an activity with respect to the enclosing history record. More specifically, it defines the requirements of an activity, for example, the description template used to create the activity. The initial definition (template) MAY change over time due to refinement of the requirements as a result of scheduling, delegation, or negotiation processes, etc. Therefore this element MAY appear in more than one *ActivityHistoryEntry*.

4.9.2 Multiplicity

The multiplicity of this element is zero or one. Every ActivityHistory record MUST contain at least one *ActivityHistoryEntry* with a non-empty ActivityDefinition element. In particular, the “initial” ActivityHistoryEntry MUST NOT have an empty ActivityDefinition element.

4.9.3 Type

This is an abstract type and has to be substituted by an appropriate definition (see Section 5.1).

4.9.4 Attributes

No attributes are defined.

4.9.5 Pseudo schema

The *ActivityDefinition* is rendered in XML as:

```
<ActivityDefinition abstract="true"/>
```

Where:

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/ActivityDefinition

Represents the *ActivityDefinition* element. Since the *ActivityDefinition* element is an abstract XML element it MUST NOT occur in a XML document by itself. It MUST be substituted by a valid *ActivityDefinition* substituent instead.

4.9.6 Extensibility

The definition provided in Section 5.1 SHOULD be used unless a more applicable specification is available.

4.10 The ActivityDependency Element

4.10.1 Definition

This element stores the dependency definitions for the activity with respect to the enclosing history record. More specifically, it describes links to associated activities within a dependency structure (such as a workflow). It does not have to contain the entire dependency structure, just the dependencies to other activities.

4.10.2 Multiplicity

The multiplicity of this element is zero or more.

4.10.3 Type

This is an abstract type and has to be substituted by an appropriate definition.

4.10.4 Attributes

No attributes are defined.

4.10.5 Pseudo schema

The abstract *ActivityDependency* has no rendering:

```
<ActivityDependency abstract="true"/>
```

Where:

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/ActivityDependency

Represents the *ActivityDependency* element. Since the *ActivityDependency* element is an abstract XML element it MUST NOT occur in a XML document by itself. It MUST be substituted by a valid *ActivityDependency* substituent instead.

4.10.6 Extensibility

No concrete extensions are provided for this element. It is however RECOMMENDED to use available standards.

4.11 The ManagerReference Element

4.11.1 Definition

This element keeps the endpoint reference of the activity's managing service at the time denoted by the enclosing record. The corresponding service SHOULD expose an interface for managing the activity's state, lifecycle, and execution.

4.11.2 Multiplicity

The multiplicity of this element is zero or one.

4.11.3 Type

This is an external type. Refer to the WS-Addressing specification [WSADDR] for further details.

4.11.4 Pseudo schema

The external *ManagerReference* has no rendering:

```
<ManagerReference external="true"/>
```

Where:

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/ManagerReference

Represents the *ManagerReference* element.

4.11.5 Extensibility

This element provides no extension points.

4.11.6 Example

```
<wsa:EndpointReference>
   <wsa:Address>http://tempuri.org/services/activitystore</wsa:Address>
</wsa:EndpointReference>
```

This endpoint shows the address of the activity store (see Section 1.4) formatted according to the Web Services Addressing standard.

4.12 The ResourceUsage Element

4.12.1 Definition

This element stores the resource usage for this activity with respect to the enclosing history record. It describes the resource consumption/usage of an activity, e.g., the number of CPUs used or maximum memory needed for some part of the activity. This element may appear multiple times for an activity, depending on the monitoring policies of the system generating them; the system may choose to perform averaging over the execution time, averaging over several periods that cover the execution time, sampling of the system, etc.

Concretizations of this element SHOULD describe the time instant it was generated or the time period they apply to. Because several monitoring systems may be feeding usage information into the activity instance description, the time points/periods MAY be overlapping and MAY be non-contiguous.

Note that there is no requirement for the information in the activity instance description to be either accurate or timely.

4.12.2 Multiplicity

The multiplicity of this element is zero or more.

4.12.3 Type

This is an abstract type and has to be substituted by an appropriate definition (see Section 5.2).

4.12.4 Attributes

No attributes are defined.

4.12.5 Pseudo schema

The abstract *ResourceUsage* has no rendering:

```
<ResourceUsage abstract="true"/>
```

Where:

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryEntry/ResourceUsage
: Represents the *ResourceUsage* element. Since the *ResourceUsage* element is an abstract XML element it MUST NOT occur in a XML document by itself. It MUST be substituted by a valid *ResourceUsage* substituent instead.

4.12.6 Extensibility

The definition provided in Section 5.2 SHOULD be used unless a more applicable specification is available.

5. The Open Grid Forum-related Activity Instance Element Set

This section defines how existing Open Grid Forum specifications are used in an Activity Instance Description document as substitutions for the respective abstract elements defined in Section 4.

5.1 The JobDefinition Element

The *JobDefinition* element substitutes the abstract type *ActivityDefinition* (see Section 4.9) and defines the activity's requirements using the Job Submission Description Language (JSDL) [JSDL]. All elements of the JSDL specification and extensions MAY be used when defining an Activity.

5.1.1 Multiplicity

The multiplicity of this element is one.

5.1.2 Type

The type of this element is `jsdl:JobDefinition_Type`.

5.1.3 Attributes

No attributes are defined.

5.1.4 Pseudo schema

The *JobDefinition* is rendered in XML as:

```
<JobDefinition substitutes="aid:ActivityDefinition"/>
```

Where:
/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryType/JobDefinition
Represents the *JobDefinition* element.

5.1.5 Example

The following example is derived from Appendix 4 of GFD.136 [JSDL] and has been adapted to the needs of this specification.

```
<aid-ogf:JobDefinition>
   <jsdl:JobDescription>
      <jsdl:JobIdentification>
         <jsdl:JobName>My gnuplot invocation</jsdl:JobName>
         <jsdl:Description>
            Simple application invocation
         </jsdl:Description>
      </jsdl:JobIdentification>
      <jsdl:Application>
         <jsdl:ApplicationName>gnuplot</jsdl:ApplicationName>
         <jsdl-posix:POSIXApplication>
            <jsdl-posix:Executable>/usr/local/bin/gnuplot
            </jsdl-posix:Executable>
            <jsdl-posix:Argument>control.txt</jsdl-posix:Argument>
            <jsdl-posix:Input>input.dat</jsdl-posix:Input>
            <jsdl-posix:Output>output1.png</jsdl-posix:Output>
         </jsdl-posix:POSIXApplication>
      </jsdl:Application>
      <jsdl:Resources>
         <jsdl:IndividualPhysicalMemory>
            <jsdl:LowerBoundedRange>1293942784.0
            </jsdl:LowerBoundedRange>
```

```
         </jsdl:IndividualPhysicalMemory>
         <jsdl:TotalCPUCount><jsdl:Exact>1.0</jsdl:Exact>
         </jsdl:TotalCPUCount>
      </jsdl:Resources>
      <jsdl:DataStaging>
         <jsdl:FileName>control.txt</jsdl:FileName>
         <jsdl:CreationFlag>overwrite</jsdl:CreationFlag>
         <jsdl:DeleteOnTermination>true</jsdl:DeleteOnTermination>
         <jsdl:Source>
            <jsdl:URI>http://tempuri.org/~me/control.txt</jsdl:URI>
         </jsdl:Source>
      </jsdl:DataStaging>
   </jsdl:JobDescription>
</aid-ogf:JobDefinition>
```

This JSDL job definition is the actual activity submitted in the first step to the primary scheduler (according to Figure 2).

5.2 The UsageRecord Element

The *UsageRecord* element substitutes the abstract type *ResourceUsage* (as defined in Section 4.12) and defines a particle of usage information in a format that is compatible with the OGF Usage Record specification [UR]. All elements defined by the specification and extensions MAY be used when describing an activity's resource usage. Where multiple Usage Records are created during the execution of an activity, multiple *UsageRecord* elements MAY be created within multiple *ActivityHistoryEntry* elements. Newer Usage Records do not necessarily make older ones obsolete. Consumer-side processing of Usage Records is outside the scope of this document. It should be noted, however, that according to the UR specification, the Usage Record's `ur:StartTime` and `ur:EndTime` elements MUST be used to determine the periods that individual *UsageRecord* elements refer to.

5.2.1 Multiplicity

The multiplicity of this record is zero or more (there may be an arbitrary number of usage records associated with an activity instance).

5.2.2 Type

The type of this element is `ur:UsageRecordType`.

5.2.3 Attributes

No attributes are defined.

5.2.4 Pseudo schema

The *UsageRecord* is rendered in XML as:

```
<UsageRecord substitutes="aid:ResourceUsage">
   ...
</UsageRecord>
```

Where:
/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryType/UsageRecord
Represents the *UsageRecord* element.

For further elements included in the *UsageRecord* element, please refer to the respective specification [UR].

5.2.5 Example

This example is derived from Section 14.1 of GFD.98 [UR] and has been adapted to the needs of this specification.

```
<aid-ogf:UsageRecord>
   <ur:RecordIdentity
      ur:recordId="http://tempuri.org/mscf/colony/PBS.1234.0"
      ur:createTime="2010-05-10T11:44:44.44" />
   <ur:JobIdentity>
   <ur:LocalJobId>PBS.1234.0</ur:LocalJobId>
   </ur:JobIdentity>
   <ur:UserIdentity>
      <ur:LocalUserId>scottmo</ur:LocalUserId>
   </ur:UserIdentity>
   <ur:Charge>2870</ur:Charge>
   <ur:Status>completed</ur:Status>
   <ur:Memory ur:storageUnit="MB">1234</ur:Memory>
   <ur:ServiceLevel ur:type="QOS">Gold level</ur:ServiceLevel>
   <ur:Processors>1</ur:Processors>
   <ur:ProjectName>mscfops</ur:ProjectName>
   <ur:MachineName>Colony</ur:MachineName>
   <ur:WallDuration>PT1S</ur:WallDuration>
   <ur:StartTime>2010-05-10T11:22:22.22</ur:StartTime>
   <ur:EndTime>2010-05-10T11:33:33.33</ur:EndTime>
   <ur:NodeCount>1</ur:NodeCount>
   <ur:Queue>batch</ur:Queue>
   <ur:Resource ur:description="quoteId">1435</ur:Resource>
   <ur:Resource ur:description="application">gnuplot</ur:Resource>
   <ur:Resource ur:description="executable">gnuplot</ur:Resource>
</aid-ogf:UsageRecord>
```

5.3 The ActivityStatus Element

The *ActivityStatus* element substitutes the abstract type State (see Section 4.5) and contains the activity's state using the Basic Execution Service (BES) state model. It supports the same states and state extensibility model as the BES specification. For details refer to Section 4 of GFD.108 [BES].

5.3.1 Multiplicity

The multiplicity of this element is one.

5.3.2 Type

The type of this element is `bes-factory:ActivityStatusType`.

5.3.3 Attributes

No additional attributes are defined.

5.3.4 Pseudo schema

The *ActivityStatus* is rendered in XML as:

```
<aid-ogf:ActivityStatus substitutes="aid:State"/>
```

Where:

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryType/Status/ActivityStatus
Represents the *ActivityStatus* element.

5.3.5 Example

The activity is in the BES Running state, and in an activity-specific sub-state of staging in files.

```
<aid-ogf:ActivityStatus state="Running">
   <n00:Staging-In/>
</aid-ogf:ActivityStatus>
```

With respect to the example in Section 1.4, this state is reached after the activity has been handed over to the BES for execution.

6. The Activity Instance Element Set not related to the Open Grid Forum

This section provides definitions for abstract element substitutions where no OGF specifications exist.

6.1 The Exception Element

6.1.1 Definition

The Exception element substitutes the *Exception* abstract type (see Section 4.8) and provides additional information about abnormal state change of the Activity. This is a basic definition as there is no applicable OGF specification at this time.

6.1.2 Multiplicity

The multiplicity of this element is one.

6.1.3 Type

This is an `xsd:complexType`. It MUST support the following elements:

- *Identifier*
- *Reason*

6.1.4 Attributes

No attributes are defined.

6.1.5 Pseudo schema

The *Exception* is rendered in XML as:

```
<Exception substitutes="aid:Exception">
   <Identifier> xsd:string </Identifier>
   <Reason> xsd:string </Reason>
</Exception>
```

Where:

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryType/Status/Exception
: Represents the *Exception* element.

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryType/Status/Exception/Identifier
: Represents the *Identifier* element as defined in Section 6.2. It is a mandatory element, which MUST appear exactly once.

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryType/Status/Exception/Reason
: Represents the *Reason* element as defined in Section 6.3. It is a mandatory element, which MUST appear exactly once.

6.1.6 Example

An exception was raised because the activity ran out of storage.

```
<aid-ogf:Exception>
   <aid-ogf:Identifier>InsufficientStorage</aid-ogf:Identifier>
   <aid-ogf:Reason>Storage quota reached</aid-ogf:Reason>
</aid-ogf:Exception>
```

6.2 The Identifier Element

6.2.1 Definition

This element identifies the raised exception by name. It provides information on the kind of exception raised. There are no format requirements.

6.2.2 Multiplicity

The multiplicity of this element is one.

6.2.3 Type

The type of this element is `xsd:string`.

6.2.4 Attributes

No attributes are defined.

6.2.5 Pseudo schema

The *Identifier* is rendered in XML as:

```
<Identifier> xsd:string </Identifier>
```

Where:
/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryType/Status/Exception/Identifier
Represents the *Identifier* element.

6.2.6 Example

A component of the activity could not write its data to storage due to insufficient storage.

```
<aid-ogf:Identifier>InsufficientStorage</aid-ogf:Identifier>
```

6.3 The Reason Element

6.3.1 Definition

This element provides additional information about the raised exception. There are no format requirements.

6.3.2 Multiplicity

The multiplicity of this element is one.

6.3.3 Type

The type of this element is `xsd:string`.

6.3.4 Attributes

No attributes are defined.

6.3.5 Pseudo schema

The *Reason* is rendered in XML as:

```
<aid-ogf:Reason> xsd:string </aid-ogf:Reason>
```

Where:

/aid:ActivityInstanceDescription/ActivityHistory/ActivityHistoryType/Status/Exception/Reason

Represents the *Reason* element.

6.3.6 Example

The reason for the component failing to write to storage was that its quota was reached.

```
<aid-ogf:Reason>Storage quota reached</aid-ogf:Reason>
```

7. Security Considerations

There are two key security considerations in relation to activity instance documents: the privacy of the data within the document and the integrity of that data.

Because an activity instance document can contain much information that is in need of being secured, it is important that services and resources handling these documents ensure that appropriate access controls are applied. The definition of such rules lies outside the scope of this specification, as is the description of how those rules are to be attached to or associated with the activity instance document.

Because an entire activity instance document, or parts thereof, may be passed between many systems between its originating source system and the eventual consumers of the data (e.g., a principal investigator or funding organization) and the fact that the document may be used for making decisions on payments for work done, it is important for the consumers of the activity instance document to be able to determine that the document they see is what was originally provided. The source system may attach XML digital signatures [XMLDSIG] to individual *ActivityHistoryEntry* elements; or to the overall *ActivityDescription* element. Signing the overall document ensures its integrity (as well as provides the ability to check who was responsible for creating it). It also has the effect of sealing that particular version of the document (though future versions of the document may also be created, at a cost of requiring some entity to recreate the signature once again).

This specification does not recommend any specific normalization or signing algorithms, though it is noted that algorithms that depend on the presence of ID attributes on elements or which depend on absolute XPath addressing [XPATH] are NOT RECOMMENDED as that makes those documents difficult to aggregate. It is RECOMMENDED that in order to gain maximal efficiency, originating source systems delay generating a signature for a document until they believe they have accumulated all the relevant *ActivityHistoryEntry* elements.

8. Contributors

Donal Fellows
Research Computing Services
The University of Manchester
Devonshire House, Precinct Centre, Oxford Road
Manchester M13 9PL
donal.k.fellows@manchester.ac.uk

Alexander Papaspyrou
Adesso Mobile Solutions GmbH
Stockholmer Allee 24
44269 Dortmund

Andreas Savva
Cloud Computing Research Center
Fujitsu Laboratories
4-1-1, Kamikodanaka, Nakahara, Kawasaki City, Japan
Email: andreas.savva@jp.fujitsu.com

Philipp Wieder
Service Computing Group/ITMC
TU Dortmund University
44227 Dortmund
philipp.wieder@udo.edu

Shahbaz Memon
Juelich Supercomputing Centre
Forschungszentrum Juelich GmbH
52428 Juelich
m.memon@fz-juelich.de

The authors would also like to thank Ali Anjomshoaa, Fred Brisard, Steve McGough, Neil Chue Hong, Shiraz Memon, Henning Mersch, Chris Smith, Wolfgang Ziegler and the people from the NextGRID project for their valuable input and the time they have committed to this specification or the foundations of it.

9. Intellectual Property Statement

10. Disclaimer

of the information herein will not infringe any rights or any implied warranties of merchantability or fitness for a particular purpose.

11. Full Copyright Notice

12. References

[BES] Foster, I., Grimshaw, A., Lane, P., Lee, W., Morgan, M., Newhouse, S., Pickles, S., Pulsipher, D., Smith, C., and Theimer, M. OGSA® Basic Execution Service Version 1.0, Grid Forum Document, GFD.108, Open Grid Forum, Lemont, Illinois, U.S.A. August 2007.

[BRADNER] Bradner, S., Key Words for Use in RFCs to Indicate Requirement Levels, RFC 2119. March 1997.

[BRAY] Bray, T., Hollander, D., Layman, A., Tobin, R., and Thompson, H.S., Namespaces in XML 1.0 (Third Edition), W3C Recommendation. December 2009.

[INFOSET] Cowan, J. and Tobin, R., XML Information Set (Second Edition), W3C Recommendation. February 2004.

[JSDL] Anjomshoaa, A., Brisard, F., Drescher, M., Fellows, D., Ly, A., McGough, S., Pulsipher, D., and Savva, A., Job Submission Description Language (JSDL) Specification, Version 1.0, Grid Forum Document, GFD.136, Open Grid Forum, Lemont, Illinois, U.S.A. July 2008.

[SCHEMA1] Thompson, H.S., Beech, D., Maloney, M., and Mendelsohn, N., XML Schema Part 1: Structures (Second Edition), W3C Recommendation. October 2004.

[SCHEMA2] Biron, P. and Malhotra, A., XML Schema Part 2: Datatypes (Second Edition), W3C Recommendation. October 2004.

[UR] Mach, R., Lepro-Metz, R., Jackson, S., and McGinnis, L., Usage Record Format, Grid Forum Document, GFD.98, Open Grid Forum, Lemont, Illinois, U.S.A. February 2007.

[UUID] Leach, P., Mealling, M., and Salz, R., A Universally Unique IDentifier (UUID) URN Namespace RFC 4122. July 2005.

[WSADDR] Gudgin, M., Hadley, M, and Rogers, T. (eds.), Web Services Addressing 1.0 – Core (WS-Addressing), W3C Recommendation. May 2006.

[WSAG] Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Nakata, T., Pruyne, J., Rofrano, J., Tuecke, S., and Xu, M., Web Services Agreement Specification (WS-Agreement), Grid Forum Document GFD.107; The Open Grid Forum, Joliet, Illinois, United States. 2007.

[WSSEC] Nadalin, A., Kaler C., Hallam-Baker, P., and Monzillo, R. (eds.), Web Services Security: SOAP Message Security 1.1, OASIS Standard. February 2006.

[XMLDSIG] Eastlake, D., Reagle, J., Solo, D., Hirsch, F., and Roessler, T. (eds.), XML Signature Syntax and Processing (Second Edition), W3C Recommendation. June 2008.

[XPATH] Berglund, A., Boag, S., Chamberlin, D., Fernández, M.F., Kay, M., Robie, J., and Siméon, J., XML Path Language (XPath) 2.0, W3C Recommendation. January 2007.

Appendix A The Abstract Activity Instance Description Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema attributeFormDefault="unqualified"
elementFormDefault="qualified"
targetNamespace="http://schemas.ogf.org/jsdl/2010/06/activity-instance-
description" version="1.0"
xmlns:aid="http://schemas.ogf.org/jsdl/2010/06/activity-instance-
description"
xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<xsd:annotation>
   <xsd:documentation xml:lang="en">
      The OGF takes no position regarding the validity or scope of
      any intellectual property or other rights that might be claim
      to pertain to the implementation or use of the technology
      described in this document or the extent to which any license
      under such rights might or might not be available; neither does
      it represent that it has made any effort to identify any such
      rights. Copies of claims of rights made available for
      publication and any assurances of licenses to be made
      available, or the result of an attempt made to obtain a general
      license or permission for the use of such proprietary rights by
      implementers or users of this specification can be obtained
      from the OGF Secretariat.

      The OGF invites any interested party to bring to its attention
      any copyrights, patents or patent applications, or other
      proprietary rights which may cover technology that may be
      required to practice this recommendation. Please address the
      information to the OGF Executive Director.

      This document and the information contained herein is provided
      on an "As Is" basis and the OGF disclaims all warranties,
      express or implied, including but not limited to any warranty
      that the use of the information herein will not infringe any
      rights or any implied warranties of merchantability or fitness
      for a particular purpose.

      Copyright (C) Open Grid Forum (2010). All Rights Reserved. This
      document and translations of it may be copied and furnished to
      others, and derivative works that comment on or otherwise
      explain it or assist in its implementation may be prepared,
      copied, published and distributed, in whole or in part, without
      restriction of any kind, provided that the above copyright
      notice and this paragraph are included on all such copies and
      derivative works. However, this document itself may not be
      modified in any way, such as by removing the copyright notice
      or references to the OGF or other organizations, except as
      needed for the purpose of developing Grid Recommendations in
      which case the procedures for copyrights defined in the OGF
      Document process must be followed, or as required to translate
      it into languages other than English.
```

```
      The limited permissions granted above are perpetual and will
      not be revoked by the OGF or its successors or assignees.
   </xsd:documentation>

   <xsd:documentation xml:lang="en">
      Abstract Activity Instance Description schema document
      according to the Activity Instance Description Specification
      Version 1.0 (GFD.X)

      Authors:
           Philipp Wieder, GWDG
           Alexander Papaspyrou, Adesso AG
           Andreas Savva, Fujitsu Laboratories Ltd
           Donal Fellows, The University of Manchester
           Shahbaz Memon, Juelich Supercomptuting Centre
   </xsd:documentation>
</xsd:annotation>

<xsd:import
   namespace="http://www.w3.org/2005/08/addressing"
   schemaLocation="http://www.w3.org/2005/08/addressing/ws-addr.xsd"/>

<xsd:element abstract="true" name="State">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         Stores a state model instance for the given activity state. In
         its abstract form, the state model is arbitrary. This is an
         abstract type and has to be substituted by an appropriate
         definition (see also GFD.X, Sections 4.6 and 5.3).
      </xsd:documentation>
   </xsd:annotation>
</xsd:element>

<xsd:element abstract="true" name="Exception">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         Stores an exception model instance for the given activity
         fault. This is an abstract type and has to be substituted by a
         appropriate definition (see also GFD.X, Sections 4.7 and 6.1).
      </xsd:documentation>
   </xsd:annotation>
</xsd:element>

<xsd:complexType name="ActivityStatusType">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         Describes the status of the activity with respect to the
         enclosing history record. The status of an activity comprises
         its current state (defined by an appropriate state model) and,
         if necessary, exception information. Every status record
         for an activity MUST contain at least the current state; if an
         exceptional condition occurs during the activity's lifetime, it
         SHOULD be also recorded here. Note that the existence of an
         exception entry is not necessarily coupled to a corresponding
         exceptional state; a possible connection between these is left
         to the implementor and SHOULD be described in the concrete
 state model's documentation.
```

```
      </xsd:documentation>
   </xsd:annotation>
   <xsd:sequence>
      <xsd:element ref="aid:State">
         <xsd:annotation>
            <xsd:documentation xml:lang="en">
               Denotes details on the state of the activity instance
               with respect to the enclosing history record. See also
               abstract element State and GFD.108, Section 6.6.
            </xsd:documentation>
         </xsd:annotation>
      </xsd:element>
      <xsd:element ref="aid:Exception" minOccurs="0">
         <xsd:annotation>
            <xsd:documentation xml:lang="en">
               Denotes the exception raised for the activity instance
               with respect to the enclosing record. See also abstract
               element Exception.
            </xsd:documentation>
         </xsd:annotation>
      </xsd:element>
   </xsd:sequence>
</xsd:complexType>

<xsd:element abstract="true" name="ActivityDefinition">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         Defines the requirements of an activity, for example the
         description template used to create the activity. The initial
         definition (template) MAY change over time due to refinement of
         the requirements as a result of scheduling, delegation, or
         negotiation processes, etc. Therefore, this element MAY appear
         in more than one ActivityHistoryEntry. This is an abstract type
         and has to be substituted by an appropriate definition (see
         GFD.X, Sections 4.8 and 5.1).
      </xsd:documentation>
   </xsd:annotation>
</xsd:element>

<xsd:element abstract="true" name="ActivityDependency">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         Describes links to associated activities within a dependency
         structure (such as a workflow). This is an abstract type and
         has to be substituted by an appropriate definition (see GFD.X,
         Section 4.9).
      </xsd:documentation>
   </xsd:annotation>
</xsd:element>

<xsd:element abstract="true" name="ResourceUsage">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         Describes the resource consumption/usage of an activity, e.g.,
         the number of CPUs used or maximum memory needed for some part
         of the activity. This element may appear multiple times for an
         activity, depending on the monitoring policies of the system
```

```
         generating them; the system may choose to perform averaging
         over the execution time, averaging over several periods that
         cover the execution time, sampling of the system, etc.
         Concretizations of this element SHOULD describe what time
         instant or time period they apply to. Because several
         monitoring systems may be feeding usage information into the
         activity instance document, the time points/periods MAY be
         overlapping and MAY be non-contiguous. Note that there is no
         requirement for the information in the activity instance
         document to be either accurate or timely. This is an abstract
         type and has to be substituted by an appropriate definition
         (see GFD.X, sections 4.11 and 5.2).
      </xsd:documentation>
   </xsd:annotation>
</xsd:element>

<xsd:simpleType name="ActivityHistoryEntryCategoryType">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         Describes the category of a historic entry for an activity. The
         possible options are "initial", "intermediate", and "final".
      </xsd:documentation>
   </xsd:annotation>
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="initial">
         <xsd:annotation>
            <xsd:documentation xml:lang="en">
               Denotes the initial history record for a given activity,
               which MUST be the first one created in the whole record.
            </xsd:documentation>
         </xsd:annotation>
      </xsd:enumeration>
      <xsd:enumeration value="intermediate">
         <xsd:annotation>
            <xsd:documentation xml:lang="en">
               Denotes an intermediate history record for a given
               activity. Such entry MUST NOT be the first one
               created in the whole record.
            </xsd:documentation>
         </xsd:annotation>
      </xsd:enumeration>
      <xsd:enumeration value="final">
         <xsd:annotation>
            <xsd:documentation xml:lang="en">
               Denotes the final history record for a given activity.
               Note that this does not imply that the activity on its
            </xsd:documentation>
         </xsd:annotation>
      </xsd:enumeration>
   </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="ActivityHistoryEntryType">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         Denotes an event in the history of an activity, containing its
         properties at the time the event occurred. Every entry MUST
```

```
        contain at least a timestamp (as attribute), the status of the
        activity at this timestamp, and a WS-Adressing endpoint
        reference to the managing service.
     </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
     <xsd:element name="Status" type="aid:ActivityStatusType">
        <xsd:annotation>
           <xsd:documentation xml:lang="en">
               Stores the status of the activity instance at the time
              described by the surrounding history entry. The status
              of an activity comprises its current state
              (defined by an appropriate state model) and, if
              necessary, exception information. Every status record for
              an activity MUST contain at least the current state; if
              an exceptional condition occurs during the activity's
              lifetime, it SHOULD be also recorded here. Note that the
              existence of an exception entry is not necessarily
              coupled to a corresponding exceptional state; a possible
              connection between these is left to the implementor and
              SHOULD be described in the concrete state model's
              documentation. See also ActivityStatusType and GFD.X,
              Section 4.5.
           </xsd:documentation>
        </xsd:annotation>
     </xsd:element>
     <xsd:element ref="aid:ActivityDefinition" minOccurs="0">
        <xsd:annotation>
           <xsd:documentation xml:lang="en">
               Stores the definition of the activity with respect to
              the enclosing history record. See also
               ActivityDefinition and GFD.X, Section 4.8.
           </xsd:documentation>
        </xsd:annotation>
     </xsd:element>
     <xsd:element ref="aid:ActivityDependency" minOccurs="0"
     maxOccurs=”unbound”>
        <xsd:annotation>
           <xsd:documentation xml:lang="en">
              Stores the dependency definitions for the activity with
              respect to the enclosing history record. See also
              ActivityDependency and GFD.X, Section 4.9.
           </xsd:documentation>
        </xsd:annotation>
     </xsd:element>
     <xsd:element name="ManagerReference"
     type="wsa:EndpointReferenceType" minOccurs="0">
        <xsd:annotation>
           <xsd:documentation xml:lang="en">
              Keeps the endpoint reference of the activity's managing
              service at the time denoted by the enclosing record. The
              corresponding service MAY expose an interface for
              managing the activity's state, lifecycle, and execution.
              See also GFD.X, Section 4.10.
           </xsd:documentation>
        </xsd:annotation>
     </xsd:element>
```

```
      <xsd:element name="Event" type="xsd:string" minOccurs="0">
         <xsd:annotation>
            <xsd:documentation xml:lang="en">
             The Event element denotes an additional information relating
             to an event occurred within a specific activity state. It
             is useful if an entity responsible of managing
             ActivityHistory, is expected to provide more information
             about the activity's state rather than only the status and
             timestamp attributes. This will help
             ActivityInstanceDescription consumers, such as users or
             client applications to better analyze activity runs or
             failures during the activity lifecycle. A more specific
             example is, when an activity is failed due to a staging-in
             failure, in this case the error details will be captured in
             an Event instance.
            </xsd:documentation>
         </xsd:annotation>
      </xsd:element>
      <xsd:element ref="aid:ResourceUsage" minOccurs="0"
      maxOccurs="unbounded">
         <xsd:annotation>
            <xsd:documentation xml:lang="en">
               Stores the resource usage for this activity with respect
               to the enclosing history record. See also ResourceUsage
               and GFD.X, Section 4.11.
            </xsd:documentation>
         </xsd:annotation>
      </xsd:element>
      <xsd:any namespace="##other" processContents="lax" minOccurs="0"
      maxOccurs="unbounded">
         <xsd:annotation>
            <xsd:documentation xml:lang="en">
               Provides an extension point for additional elements in
               the ActivityInstanceDescriptionType. Implementations
               SHOULD ignore unsupported extensions.
            </xsd:documentation>
         </xsd:annotation>
      </xsd:any>

   </xsd:sequence>
   <xsd:attribute name="timestamp" type="xsd:dateTime"
   use="required">
      <xsd:annotation>
         <xsd:documentation xml:lang="en">
            Keeps the timestamp at which this event has occurred in
            the activity's history. The entries in the whole activity
            history SHOULD be ordered ascending to their timestamp.
            See GFD.X, Section 4.4.4,
         </xsd:documentation>
      </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="category"
   type="aid:ActivityHistoryEntryCategoryType" use="optional">
      <xsd:annotation>
         <xsd:documentation xml:lang="en">
            Denotes the category of this history record. See
            ActivityHistoryEntryCategoryType and GFD.X, Section
```

```
            4.4.4, for possible values.
         </xsd:documentation>
         </xsd:annotation>
      </xsd:attribute>
   <xsd:anyAttribute namespace="##other" processContents="lax"/>
</xsd:complexType>

<xsd:complexType name="ActivityHistoryType">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         Keeps track of the record of the activity's history. This
         record MUST contain one initial record and, at most, one final
         record see also ActivityHistoryEntryCategoryType). Note that,
         although a final record MAY have been written already, the
         activity document MAY still be modified.
      </xsd:documentation>
   </xsd:annotation>
   <xsd:sequence>
      <xsd:element name="ActivityHistoryEntry"
      type="aid:ActivityHistoryEntryType" minOccurs="0"
      maxOccurs="unbounded">
         <xsd:annotation>
            <xsd:documentation xml:lang="en">
               Stores a single event in an activity's history. It
               denotes an event in the history of an activity,
               containing its properties at the time the event
               occurred. Every entry MUST contain at least a
               timestamp (as attribute), the status of
               the activity at this timestamp, and a WS-Addressing
               [WSADDR] endpoint reference to the managing service.
               Once an ActivityHistoryEntry is written, it MUST NOT
               be altered. Additional information about the
               respective activity has to be appended to the
               ActivityHistory by adding a new ActivityHistoryEntry
               element. See also ActivityHistoryEntryType and GFD.X,
               Section 4.4.
            </xsd:documentation>
         </xsd:annotation>
      </xsd:element>
   </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="ActivityInstanceDescriptionType">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         Contains an activity's meta-data and history and provides a
         root element for every activity. While the meta-data part MAY
         carry information about the activity's creator, purpose, and
         references (i.e. to other activities), the history part SHOULD
         describe the full lifecycle of the activity.
      </xsd:documentation>
   </xsd:annotation>
   <xsd:sequence>
      <xsd:element name="ActivityDescription" type="xsd:string"
      minOccurs="0">
         <xsd:annotation>
            <xsd:documentation xml:lang="en">
```

```
                Contains a natural-language description of the activity
                and offers means for storing additional information on
                the activity for displaying purposes (e.g. in a user
                interface). See also ActivityDescriptionType and GFD.X,
                Section 4.2.
             </xsd:documentation>
          </xsd:annotation>
       </xsd:element>
       <xsd:element name="ActivityHistory"
       type="aid:ActivityHistoryType">
          <xsd:annotation>
             <xsd:documentation xml:lang="en">
                Stores the history of an activity. See also
                ActivityHistoryType and GFD.X, Section 4.3.
             </xsd:documentation>
          </xsd:annotation>
       </xsd:element>
       <xsd:any namespace="##other" processContents="lax" minOccurs="0"
       maxOccurs="unbounded">
          <xsd:annotation>
             <xsd:documentation xml:lang="en">
                Provides an extension point for additional root elements
                in the activity document. Implementations SHOULD ignore
                unsupported extensions.
             </xsd:documentation>
          </xsd:annotation>
       </xsd:any>
    </xsd:sequence>
    <xsd:attribute name="id" type="xsd:string">
       <xsd:annotation>
          <xsd:documentation xml:lang="en">
             An identifier for the activity, which MUST be globally
             unique. It is RECOMMENDED to use Universally Unique
             Identifiers as described in RFC 4122, "A Universally Unique
             IDentifier (UUID) URN Namespace". See also GFD.X, Section
             4.1.4.
          </xsd:documentation>
       </xsd:annotation>
    </xsd:attribute>
    <xsd:anyAttribute namespace="##other" processContents="lax">
       <xsd:annotation>
          <xsd:documentation xml:lang="en">
             Provides an extension point for additional root attributes
             in the activity document. Implementations SHOULD ignore
             unsupported extensions.
          </xsd:documentation>
       </xsd:annotation>
    </xsd:anyAttribute>
</xsd:complexType>

<xsd:element name="ActivityInstanceDescription"
type="aid:ActivityInstanceDescriptionType">
    <xsd:annotation>
       <xsd:documentation xml:lang="en">
          The document root of a single activity instancewhich contains
           an activity's meta-data and history and provides the entry
           point for every activity. While the meta-data part MAY carry
```

```
        information about the activity's creator, purpose, and
         references (i.e. to other activities), the history part
         SHOULD describe the full lifecycle of the activity. See also
        ActivityInstanceDescriptionType and GFD.X, Section 4.1.
      </xsd:documentation>
   </xsd:annotation>
</xsd:element>

</xsd:schema
```

Appendix B Open Grid-Forum-related Activity Instance Description Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema attributeFormDefault="unqualified"
elementFormDefault="qualified"
targetNamespace="http://schemas.ogf.org/jsdl/2010/06/activity-instance-
description-ogf" version="1.0"
xmlns:aid="http://schemas.ogf.org/jsdl/2010/06/activity-instance-
description"
xmlns:aid-ogf="http://schemas.ogf.org/jsdl/2010/06/activity-instance-
description-ogf"
xmlns:bes-factory="http://schemas.ggf.org/bes/2006/08/bes-factory"
xmlns:jsdl="http://schemas.ggf.org/jsdl/2005/11/jsdl"
xmlns:urf="http://schema.ogf.org/urf/2003/09/urf"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<xsd:annotation>
   <xsd:documentation xml:lang="en">
      The OGF takes no position regarding the validity or scope of
      any intellectual property or other rights that might be claim
      to pertain to the implementation or use of the technology
      described in this document or the extent to which any license
      under such rights might or might not be available; neither does
      it represent that it has made any effort to identify any such
      rights. Copies of claims of rights made available for
      publication and any assurances of licenses to be made
      available, or the result of an attempt made to obtain a general
      license or permission for the use of such proprietary rights by
      implementers or users of this specification can be obtained
      from the OGF Secretariat.

      The OGF invites any interested party to bring to its attention
      any copyrights, patents or patent applications, or other
      proprietary rights which may cover technology that may be
      required to practice this recommendation. Please address the
      information to the OGF Executive Director.

      This document and the information contained herein is provided
      on an "As Is" basis and the OGF disclaims all warranties,
      express or implied, including but not limited to any warranty
      that the use of the information herein will not infringe any
      rights or any implied warranties of merchantability or fitness
      for a particular purpose.

      Copyright (C) Open Grid Forum (2010). All Rights Reserved. This
      document and translations of it may be copied and furnished to
      others, and derivative works that comment on or otherwise
      explain it or assist in its implementation may be prepared,
      copied, published and distributed, in whole or in part, without
      restriction of any kind, provided that the above copyright
      notice and this paragraph are included on all such copies and
      derivative works. However, this document itself may not be
      modified in any way, such as by removing the copyright notice
      or references to the OGF or other organizations, except as
      needed for the purpose of developing Grid Recommendations in
```

```
      which case the procedures for copyrights defined in the OGF
      Document process must be followed, or as required to translate
      it into languages other than English.

      The limited permissions granted above are perpetual and will
      not be revoked by the OGF or its successors or assignees.
   </xsd:documentation>
</xsd:annotation>
<xsd:annotation>
   <xsd:documentation xml:lang="en">
      OGF-specific Activity Instance schema document according to the
      Activity Instance Description Specification Version 1.0 (GFD.X).

      Authors:
           Philipp Wieder, GWDG
           Alexander Papaspyrou, Adesso AG
           Andreas Savva, Fujitsu Laboratories Ltd
           Donal Fellows, The University of Manchester
           Shahbaz Memon, Juelich Supercomptuting Centre
   </xsd:documentation>
</xsd:annotation>

<xsd:import
namespace="http://schemas.ogf.org/jsdl/2010/06/activity-instance-
description"
schemaLocation http://schemas.ogf.org/jsdl/2010/06/activity-instance-
description />

<xsd:import
namespace="http://schemas.ggf.org/bes/2006/08/bes-factory"
schemaLocation="http://schemas.ggf.org/bes/2006/08/bes-factory"/>

<xsd:import
namespace="http://schemas.ggf.org/jsdl/2005/11/jsdl"
schemaLocation="http://schemas.ggf.org/jsdl/2005/11/jsdl"/>

<xsd:import
namespace="http://schema.ogf.org/urf/2003/09/urf"
schemaLocation="http://schemas.ogf.org/urf/2003/09/url"/>

<!-- ===== ELEMENTS WITHIN SUBSTITUTION GROUPS ===== -->

<xsd:element name="ActivityStatus" substitutionGroup="aid:State"
type="bes-factory:ActivityStatusType"/>

<xsd:element name="Exception" substitutionGroup="aid:Exception">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="Identifier" type="xsd:string">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">
                   Identifies the raised exception by name.
                   It provides information on the kind of exception
                   raised. There are no format requirements.
               </xsd:documentation>
            </xsd:annotation>
         </xsd:element>
```

```
        <xsd:element name="Reason" type="xsd:string">
           <xsd:annotation>
              <xsd:documentation xml:lang="en">
                  Provides additional information about the raised
                  exception. There are no formal requirements.
              </xsd:documentation>
           </xsd:annotation>
        </xsd:element>
        <xsd:any namespace="##other" processContents="lax"
        minOccurs="0" maxOccurs="unbounded">
           <xsd:annotation>
              <xsd:documentation xml:lang="en">
                  Provides an extension point for additional root
                  elements ib the activity document. Implementations
                  SHOULD ignore uupported extensions.
              </xsd:documentation>
           </xsd:annotation>
        </xsd:any>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>

<xsd:element name="JobDefinition"
substitutionGroup="aid:ActivityDefinition"
type="jsdl:JobDefinition_Type"/>

<xsd:element name="UsageRecord"
substitutionGroup="aid:ResourceUsage" type="urf:UsageRecordType"/>

</xsd:schema>
```

Appendix C Activity Instance Description Example A

This example shows an Activity Instance created due to a job submitted to a scheduler. It contains an document ID (see Section 4.1.4), and description of the activity (see Section 4.2), the initial history entry (see Section 4.4), which contains the status of the activity at the time of submission (Pending) (see Section 4.5), an initial definition of the activity (see Section 4.9), and a reference to the activity manager (see Section 4.11).

```
<?xml version="1.0" encoding="UTF-8"?>
<aid:ActivityInstanceDescription
xmlns="http://schemas.ogf.org/jsdl/2010/06/activity-
instancedescription-ogf"
xmlns:aid-ogf="http://schemas.ogf.org/jsdl/2010/06/activity-instance-
description-ogf"
xmlns:aid="http://schemas.ogf.org/jsdl/2010/06/activity-instance-
description"
xmlns:jsdl="http://schemas.ggf.org/jsdl/2005/11/jsdl"
xmlns:jsdl-posix="http://schemas.ggf.org/jsdl/2005/11/jsdl-posix"
xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://schemas.ogf.org/jsdl/2010/06/activity-
instance-description http://schemas.ogf.org/jsdl/2010/06/activity-
instance-description-ogf.xsd"
id="ea196512-9cb7-4a14-91b0-2dde749a5f7d">

<aid:ActivityDescription>
   This activity instance has been generated due to an activity
   request submitted to the scheduling service with the following
   URI: http://tempuri.org/services/activityscheduler. The activity
   request has been received at 2010-05-10T11:11:11.11. The activity
   instance has been created 2010-05-10T11:11:44.44 by the
   organization's activity store with the following URI:
   http://tempuri.org/services/activitystore.
</aid:ActivityDescription>

<aid:ActivityHistory>
   <aid:ActivityHistoryEntry timestamp="2010-05-10T11:11:44.44"
   category="initial">

      <aid:Status>
         <ActivityStatus state="Pending"/>
      </aid:Status>

      <JobDefinition>
         <jsdl:JobDescription>
            <jsdl:JobIdentification>
               <jsdl:JobName> My gnuplot invocation</jsdl:JobName>
               <jsdl:Description>
                   Simple application invocation
               </jsdl:Description>
            </jsdl:JobIdentification>
            <jsdl:Application>
               <jsdl:ApplicationName>gnuplot</jsdl:ApplicationName>
               <jsdl-posix:POSIXApplication>
                   <jsdl-posix:Executable>
                         /usr/local/bin/gnuplot
                   </jsdl-posix:Executable>
```

```
                  <jsdl-posix:Argument>
                        control.txt
                  </jsdl-posix:Argument>
                  <jsdl-posix:Input>input.dat</jsdl-posix:Input>
                  <jsdl-posix:Output>output1.png</jsdl-posix:Output>
               </jsdl-posix:POSIXApplication>
            </jsdl:Application>
            <jsdl:Resources>
               <jsdl:IndividualPhysicalMemory>
                  <jsdl:LowerBoundedRange>
                        1293942784.0
                  </jsdl:LowerBoundedRange>
               </jsdl:IndividualPhysicalMemory>
               <jsdl:TotalCPUCount>
                  <jsdl:Exact>1.0</jsdl:Exact>
               </jsdl:TotalCPUCount>
            </jsdl:Resources>
            <jsdl:DataStaging>
               <jsdl:FileName>control.txt</jsdl:FileName>
               <jsdl:CreationFlag>overwrite</jsdl:CreationFlag>
               <jsdl:DeleteOnTermination>true</jsdl:DeleteOnTermination>
               <jsdl:Source>
                  <jsdl:URI>
                        http://tempuri.org/~me/control.txt
                  </jsdl:URI>
               </jsdl:Source>
            </jsdl:DataStaging>
         </jsdl:JobDescription>
      </JobDefinition>

      <aid:ManagerReference>
         <wsa:Address>
            http://tempuri.org/services/
         </wsa:Address>
      </aid:ManagerReference>

      <aid:Event>Activity created with ID 72524628.
                 Created with type JSDL</aid:Event>
   </aid:ActivityHistoryEntry>
</aid:ActivityHistory>

</aid:ActivityInstanceDescription>
```

Appendix D Activity Instance Description Example B

This example shows the same Activity Instance as in the previous example (therefore the *JobDefinition* content is not shown in detail). The activity changed from status Pending to Running and then Finished. The final *HistoryEntry* also carries the *UsageRecord* that details the resources consumed by the acivity.

```
<?xml version="1.0" encoding="UTF-8"?>
<aid:ActivityInstanceDescription
xmlns="http://schemas.ogf.org/jsdl/2010/06/activity-
instancedescription-ogf"
xmlns:aid-ogf="http://schemas.ogf.org/jsdl/2010/06/activity-instance-
description-ogf"
xmlns:aid="http://schemas.ogf.org/jsdl/2010/06/activity-instance-
description"
xmlns:jsdl="http://schemas.ggf.org/jsdl/2005/11/jsdl"
xmlns:jsdl-posix="http://schemas.ggf.org/jsdl/2005/11/jsdl-posix"
xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://schemas.ogf.org/jsdl/2010/06/activity-
instance-description http://schemas.ogf.org/jsdl/2010/06/activity-
instance-description-ogf.xsd"
id="ea196512-9cb7-4a14-91b0-2dde749a5f7d">

<aid:ActivityDescription>
   This activity instance has been generated due to an activity
   request submitted to the scheduling service with the following
   URI: http://tempuri.org/services/activityscheduler. The activity
   request has been received at 2010-05-10T11:11:11.11. The activity
   instance has been created 2010-05-10T11:11:44.44 by the
   organization's activity store with the following URI:
   http://tempuri.org/services/activitystore.
</aid:ActivityDescription>

<aid:ActivityHistory>
   <aid:ActivityHistoryEntry timestamp="2010-05-10T11:11:44.44"
   category="initial">

      <aid:Status>
         <ActivityStatus state="Pending"/>
      </aid:Status>

      <JobDefinition>
         ...
      </JobDefinition>

      <aid:ManagerReference>
         <wsa:Address>
            http://tempuri.org/services/
         </wsa:Address>
      </aid:ManagerReference>
   </aid:ActivityHistoryEntry>

   <aid:ActivityHistoryEntry timestamp="2010-05-10T11:22:22.22"
   category="intermediate">
      <aid:Status>
```

```
         <ActivityStatus state="Running"/>
      </aid:Status>

      <aid:ManagerReference>
         <wsa:Address>
            http://tempuri.org/services/activitystore
         </wsa:Address>
      </aid:ManagerReference>
   </aid:ActivityHistoryEntry>

   <aid:ActivityHistoryEntry timestamp="2010-05-10T11:44:44.44"
   category="final">

      <aid:Status>
         <ActivityStatus state="Finished"/>
      </aid:Status>

      <aid:ManagerReference>
         <wsa:Address>
            http://tempuri.org/services/activitystore
         </wsa:Address>
      </aid:ManagerReference>

      <UsageRecord>
         <ur:RecordIdentity
         ur:recordId="http://tempuri.org/mscf/colony/PBS.1234.0"
         ur:createTime="2010-05-10T11:44:44.44"/>
         <ur:JobIdentity>
            <ur:LocalJobId>PBS.1234.0</ur:LocalJobId>
         </ur:JobIdentity>
         <ur:UserIdentity>
            <ur:LocalUserId>scottmo</ur:LocalUserId
         </ur:UserIdentity>
         <ur:Charge>2870</ur:Charge>
         <ur:Status>completed</ur:Status>
         <ur:Memory ur:storageUnit="MB">1234</ur:Memory>
         <ur:ServiceLevel ur:type="QOS">Gold level</ur:ServiceLevel>
         <ur:Processors>1</ur:Processors>
         <ur:ProjectName>mscfops</ur:ProjectName>
         <ur:MachineName>Colony</ur:MachineName>
         <ur:WallDuration>PT1S</ur:WallDuration>
         <ur:StartTime>2010-05-10T11:22:22.22</ur:StartTime>
         <ur:EndTime>2010-05-10T11:33:33.33</ur:EndTime>
         <ur:NodeCount>1</ur:NodeCount>
         <ur:Queue>batch</ur:Queue>
         <ur:Resource ur:description="quoteId">1435</ur:Resource>
         <ur:Resource ur:description="application">gnuplot</ur:Resource>
         <ur:Resource ur:description="executable">gnuplot</ur:Resource>
      </UsageRecord>
   </aid:ActivityHistoryEntry>
</aid:ActivityHistory>

</aid:ActivityInstanceDescription>
```