GFD-R(ggf-ogsa-bes-spec-1.0)
Open Grid Services Architecture Basic Execution Service

Authors:
I. Foster, Argonne National Laboratory
A. Grimshaw, U. Virginia
P. Lane, Argonne National Laboratory
W. Lee, Imperial College London
S. Newhouse, U. Southampton
S. Pickles, U. Manchester
Darren Pulsipher, EMC
C. Smith, Platform
M. Theimer, Microsoft

http://forge.gridforum.org/projects/ogsa-bes-wg
3/17/2007

OGSA Basic Execution Service

Version 1.0

Abstract

This document presents a specification for a basic execution service (BES): a service to which clients can send requests to initiate, monitor, and manage computational activities. The specification defines an extensible *state model* for activities; an extensible *information model* for a BES and the activities that it creates; and three *port-types*; BES-Management, BES-Factory, and BES-Activity. BES-Management defines operations for managing the BES itself. BES-Factory defines operations for initiating, monitoring, and managing sets of activities, and for accessing information about the BES. BES-Activity defines operations for for monitoring and managing individual activities.

Contents

1 Introduction

This Basic Execution Service (BES) specification defines Web Services interfaces for creating, monitoring, and controlling computational entities such as UNIX or Windows processes, Web Services, or parallel programs—what we call *activities*. Clients define activities using the Job Submission Description Language (JSDL) **[JSDL 1.0]**. A BES implementation executes each activity that it accepts on an appropriate computational resource, which—depending on the BES implementation and the type(s) of activities supported—may be a single computer; a cluster managed through a resource manager such as Load Leveler, Sun Grid Engine, Portable Batch System, or Condor; a Web Service hosting environment; or even another BES implementation.

More specifically, the BES specification defines three Web Services port-types (see Table 1), aimed at different types of client:

- the **BES-Factory** port-type allows clients to create, monitor, and control sets of activities, and to monitor BES attributes, such as a characterization of the resources the BES makes available to activities and the number of activities it currently has instantiated. This port-type is intended for use by ordinary clients.
- the **BES-Activity** port-type allows clients to create, monitor, and control individual activities. This port-type is intended for use by ordinary clients.
- the **BES-Management** port-type allows clients to monitor the details of and control the BES itself..This port-type is intended for use by system administrators.

In defining these three port-types, the OGSA-BES Working group had to balance competing needs for (a) achieving commonality across BES implementations, so that clients can easily access different BES implementations in standard ways; while also, when necessary, (b) allowing BES implementations to expose differences that may be important to clients. To this end, the BES specification allows for extensibility in three areas; namely state model, information model, and resource model.

State model: During its execution, an activity passes through various states. Different BES implementations may support different sets of states and allowable transitions. To accommodate such differences, the BES specification defines (a) a basic state model in which every successful activity is initially *Pending*, then *Running*, and finally one of *Finished*, *Terminated*, or *Failed*, and (b) mechanisms by which this basic state model can be extended, by defining sub-states, within a specific BES implementation.

Information model: Clients need to be able to inspect various properties of a BES and its activities. Different BES implementations may want to expose different properties, depending (for example) on the computational resources to which they provide access and the types of activity that they support. To accommodate such differences, the BES specification (a) defines a set of attributes that any implementation of the BES port-types must recognize, while (b) allowing many of those attributes to be optional, and (c) allowing this set to be extended (by defining additional attributes) within a specific BES implementation.

Resource model: The OGSA-BES Working Group supports different “resource models” (as they are sometimes called) through what they term profiles. Consequently, the BES specification requires that all BES implementations MUST support a simple operation for retrieving all attributes in a single document. At the same time, the specification does not preclude the option that a specific BES implementation MAY support other access mechanisms. In particular, an implementation MAY compose appropriate port-types—e.g., those defined in the WSRF/WS-Notification, WS-Transfer/WS-Eventing, or WS-ResourceTransfer families of specifications—with the port-types defined in the BES specification.

Because additional resource model renderings are provided compositionally rather than explicitly, all schema and WSDL included as rendering information in this document should be interpreted to be common or profile-agnostic, normative schema. That is, these schema should be

interpreted as the schema and WSDL necessary in addition to, or on top of any types, operations, or other properties required by various resource model renderings, such as for example the OGSA WSRF Basic Profile 1.0 [OGSA WSRF BP]. For illustrative purposes, we specify in Appendix I the WSRF Resource Properties type definitions that map naturally to BES attributes.

Table 1: Summary of the three BES port-types and their operations

BES-Management Port-type	
StopAcceptingNewActivities	Request that the BES stop accepting new activities
StartAcceptingNewActivities	Request that the BES start accepting new activities
BES-Factory Port-type	
CreateActivity	Request the creation of a new activity
GetActivityStatuses	Request the status of a set of activities
TerminateActivities	Request that a set of activities be terminated
GetActivityDocuments	Request the JSDL documents for a set of activities
GetFactoryAttributesDocument	Request XML document containing BES properties
BES-Activity Port-type (optional)	
GetStatus	Request the status of an activity
Terminate	Request that an activity be terminated
GetDocument	Request the JSDL document for an activity
GetActivityAttributesDocument	Request XML document containing activity properties

1.1 Terminology

The keywords “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, “OPTIONAL” in this document are to be interpreted as described in RFC 2119 [**RFC 2119**].

Additional terms used in this document are defined in the Glossary.

When describing abstract data models, this specification uses the notational convention used by the XML Infoset specification [**XML Infoset**].

When describing concrete XML schemas, this specification uses the notational convention of the WS-Security specification [**WS-Security**]. Specifically, each member of an element’s [children] or [attributes] property is described using an XPath-like notation **[XPath]** (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/>).

1.2 Namespaces

The following namespaces are used in this document.

Prefix	Namespace
s11	http://schemas.xmlsoap.org/soap/envelope
xsd	http://www.w3.org/2001/XMLSchema
wsa	http://www.w3.org/2005/03/addressing

wse	http://schemas.xmlsoap.org/ws/2004/08/eventing
jsdl	http://schemas.ggf.org/jsdl/2005/11/jsdl
bes-mgmt	http://schemas.ggf.org/bes/2006/08/bes-management
bes-factory	http://schemas.ggf.org/bes/2006/08/bes-factory
bes-activity	http://schemas.ggf.org/bes/2006/08/bes-activity
bes-ext	http://schemas.ggf.org/bes/2006/08/bes-extensions

2 Job Submission Description Language

We review JSDL briefly, as our choice of JSDL as an activity description language implies certain assumptions about the types of activities that BES implementations can support and the ways in which these activities can execute.

A JSDL document describes a single activity. With the exception of stage in and stage out activities (discussed next), the execution of this activity is (from the perspective of an external observer) an atomic operation. If it can be decomposed further, such decomposition is not visible through the service interface.

JSDL provides a stage-in/stage-out model and associated "local" file system capability in which data is copied to and from the locus of execution. JSDL presumes that there is a universally visible, local file system, regardless whether this locus is a single host or a large cluster.

A BES that supports the stage-in/stage-out model may want to define state model extensions to allow clients to monitor the status of stage-in/stage-out operations. Such extensions can be defined, similarly to those described in Section 4.2, but are out of scope for this specification.

3 Naming Activities: Endpoint References

As we describe in Section 6.2.1, the BES CreateActivity operation returns a WS-Addressing Endpoint Reference (EPR), which clients can subsequently use to refer to the new activity. Some BES implementations may wish to use the extended "WS-Naming" EPR syntax defined within the WS-Naming specification **[WS-Naming]**. In order to allow for the use of this extended syntax, without mandating its adoption, we specify that:

- All operations that generate EPRs (e.g., the "CreateActivity" operation) can generate any EndpointReferenceType-compliant data type. That is, the data type MUST be compliant with WS-Addressing EndpointReferenceTypes, but MAY include profiled data in addition to the required elements, as is the case with WS-Names.
- All operations that operate on EPRs as arguments (e.g., the GetActivityStatuses operation) MUST accept any EndpointReferenceType-compliant data type but MAY ignore any additional elements.
- A BES implementation MUST always present one or more XML anyURIs to clients and these anyURIs MUST be from a set of URIs that represent the types of endpoints that the BES implementation deals with. For example:
  - http://schemas.ggf.org/bes/2006/08/bes/naming/BasicWSAddressing
  - http://www.ogf.org/naming/2006/08/naming-uwsep-pf[1]

[1] Defined by WS-Naming Specification. See WS-Naming for further conformance URIs.

4 State Model

Activities transition through a number of states. It is a challenge to construct a state model that is simple, expressive, and extensible without confusing clients that are unaware of the extensions. The BES specification addresses these challenges by defining first a *basic* state model, and then mechanisms for extending this state model.

4.1 Basic State Model

All activities MUST traverse the following set of states, employing only those transitions between states shown in Figure 1:

- *Pending*: The service has created a record for an activity but not yet instantiated it on a suitable computational resource or enabled it to start execution on such a resource.
- *Running*: The activity is executing on some computational resource.
- *Finished* (a terminal state): The activity has terminated successfully. Successful termination implies that the activity exited of its own accord rather than due to some failure in the BES or of the computational resources on which the activity was running. Note that a successfully terminating activity may nevertheless return an error code as its return value.
- *Terminated* (a terminal state): The client – which might be some system administrator (and hence not necessarily the client who originated the request to create the activity) – has issued a TerminateActivity request.
- *Failed* (a terminal state): The activity has failed due to some system error/failure event, such as failure of a computational resource that the activity was running on.

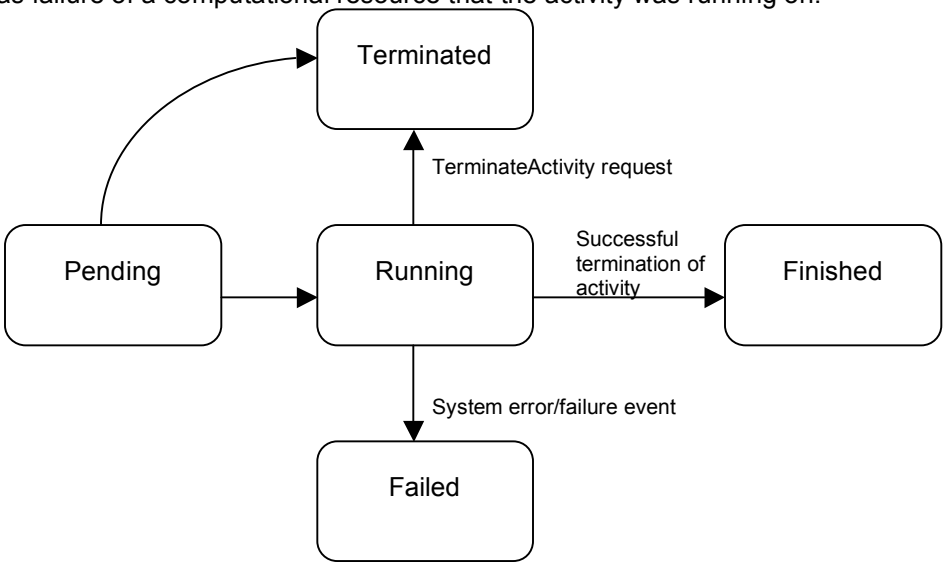

Figure 1. Basic state model.

4.2 State Specialization

To enable interaction among clients and BESs that understand differing levels of functionality, the notion of specialization of states is introduced. This mechanism enables a simple client who only understands basic BES states to interact with more complex service instances, albeit only understanding such services' activities in basic terms. Similarly, it provides a way for more complex clients to "down-grade" their interactions with simpler BESs in a well understood, interoperable manner. Specific specializations of the basic state model—or of other specializations thereof that have been previously defined—should be defined using profiles.

The following is a simple example of state specialization: A *data staging profile* extends the basic state model to support the notion of staging input data from a client to a BES before an activity runs, and then staging result data from the BES back to the client after the activity has finished. Figure 2 illustrates this specialization.

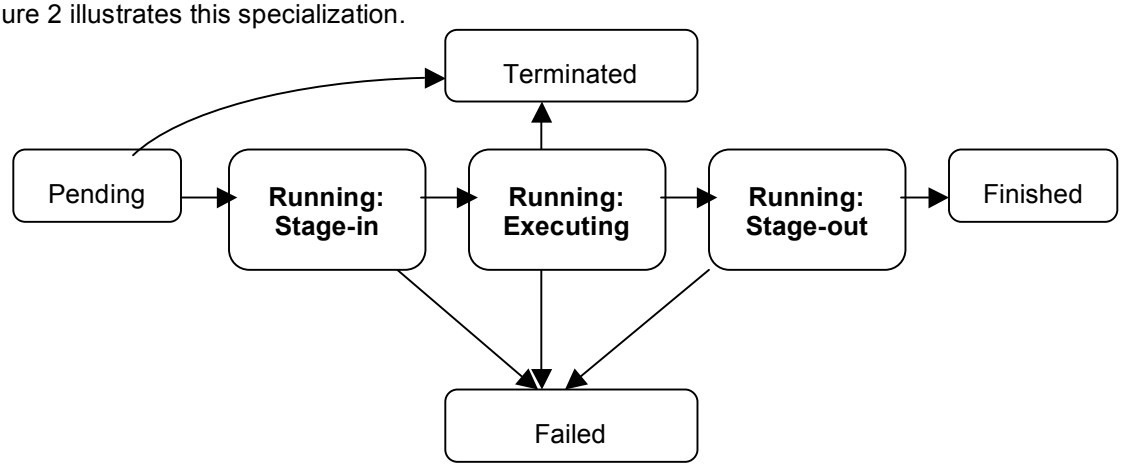

Figure 2. Data staging specialization profile: Extends the basic state model to support the notion of staging in data from a client user to the BES before an activity runs and staging data out back to the client user after the activity has finished.

A client understanding only the basic state model could still create activities on a BES that implements the data staging profile; it would simply ignore the ”Running:Stage-in,” “Running:Executing,” and “Running:Stage-out” sub-states of the “Running” state. (See Section 4.3 for how sub-states are represented in an extensible manner that oblivious clients can easily ignore.) Similarly, a client who understands the specialization profile would still understand how to interact with a BES implementing only the basic state model since the basic states are a strict subset of those defined in the specialization profile.

4.2.1 Defining Valid Specializations

To ensure interoperability between clients and BESs with differing understandings of various specializations, restrictions must be introduced on how specializations may be defined. To illustrate why restrictions are needed, consider a variation of the data staging profile in which the failure of a data staging operations returns an activity from the “Running:Stage-in” sub-state to a “Pending” state. Figure 3 illustrates this alternative data staging profile. This specialization profile has the undesirable property that it exposes an unexpected and hence uninterpretable behavior to clients who are unfamiliar with the specialization. In this case, a client who understands only the basic state model could see an unexpected transition from “Running” to “Pending.”

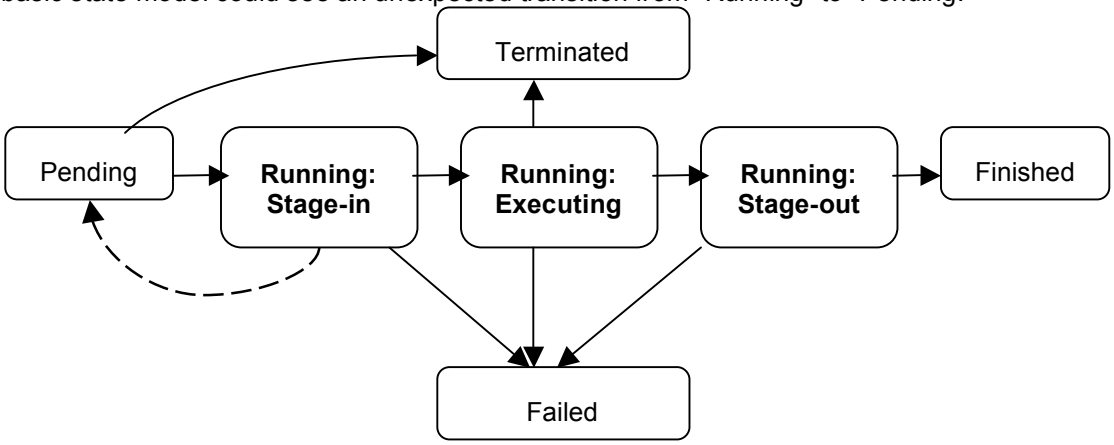

Figure 3. An illegal variation of a data staging specialization profile: Failed data stage-ins result in an activity being placed in the "Pending" state. This profile is illegal because it adds a state transition that would be unexpectedly visible to clients who only understand the basic state model.

BESs MUST not implement such specialization profiles. More specifically, any specialization profile that a BES implements MUST obey the following rules regarding sub-state definitions and allowed state transitions:

1. A specialization can introduce sub-states only by replacing a state in the state model that it is specializing (which itself may be a specialization of some other state model) with a graph of sub-states and state transitions among those sub-states.
2. A state transition from any sub-state in the specialization to another state, S, in the unspecialized state model may only occur if a corresponding state transition already existed in the unspecialized state model from the state that has been replaced, R, to that state S.

4.2.2 Specialization Example: Suspend/Resume

In another specialization example, the "Running" state is split into "Proceeding" and "Suspended" sub-states. Figure 4 illustrates this specialization.

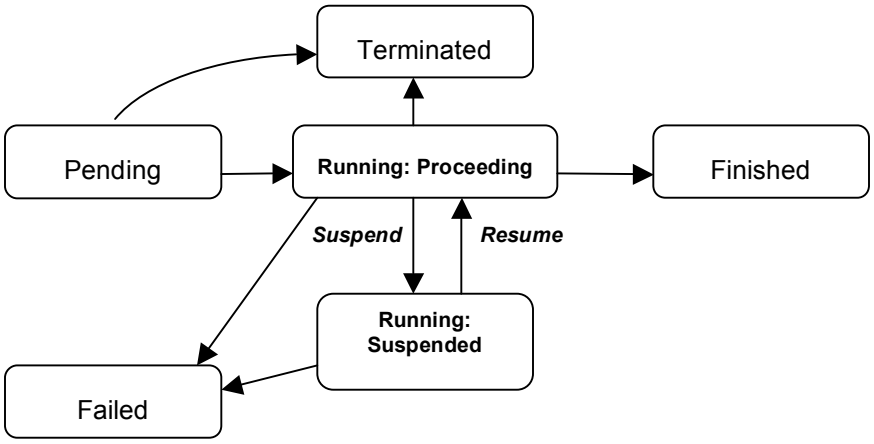

Figure 4. Suspend/resume specialization profile: Activities can be suspended and resumed as a result of appropriate client requests.

In contrast to the previous example: the transition from "Running:Proceeding" to "Running:Suspended" is assumed to occur in response to a client request. Thus, the specialization profile will presumably also define additional interface operations that enable clients to request "suspend" and "resume" state transitions.

4.2.3 Composition of Specializations

If multiple independent specialization profiles are defined the question arises of what it means for any given BES to implement multiple such profiles. That is, how does composition of multiple specialization profiles work? A related question is how clients that may not understand all profiles implemented by a BES can still interact with that BES in a meaningful manner.

Consider the two specialization examples presented so far and their composition. A BES implementing both would yield a service capable of performing data staging for its activities as well as allowing clients to suspend and resume activities. The composition of the two capabilities raises a number of issues that must be addressed:

- The *Suspend* operation is not necessarily applicable to all the sub-states of "Running" that are defined in the data staging profile. For example, the *Suspend* operation might be supported for an activity that is in state "Running:Executing", but not for activities in sub-states "Running:Stage-in" and "Running:Stage-out." Thus, a client must be prepared to

receive a fault response to a request, indicating that a request is inapplicable to the activity's current sub-state.

- A variation on the theme of inapplicability is the case where a requested client operation is applicable to a sub-state that an activity will eventually transition to of its own accord—i.e., without requiring further interaction with the client. In such a case, a BES might choose to apply the client's request eventually. For example, if it receives a *Suspend* request for an activity currently in the "Running:Stage-in" state then it might apply that *Suspend* request once the activity has transitioned to the "Running:Executing" state.

  This case raises the question of what to tell a client about their request, and when. The client may wish to receive a response to its request immediately, telling it that the request will eventually be applied once the relevant activity has progressed to a suitable sub-state. Alternatively, the client may wish to receive a response to its request only when the requested change has actually occurred. To support both response cases requires that clients can specify in a request whether they wish to receive an immediate response back or whether they wish to only receive a response once the request has actually been acted upon. In the former case a client must be prepared to receive back a fault response indicating that their request will eventually be applied.

- Different BESs may or may not support the notion that suspension is applicable to the "Running:Stage-in" and "Running:Stage-out" states. This example illustrates the fact that composition of specialization profiles can yield composition sub-states whose semantics are not fully defined by the specialization profiles being composed. Note that clients who are prepared to receive "request-inapplicable" and "request-will-be-applied-eventually" faults can still determine the semantics of any given BES they interact with in an unambiguous manner at interaction time. However, the behavior of two different BESs with which a client interacts may vary with respect to whether they implement a given client request or return a fault response instead.

  Unambiguous composition of specialization profiles requires the definition of an additional profile that precisely defines the semantics of each compositional sub-state when two or more specific specialization profiles are composed. Any given BES can choose to implement a compositional specialization profile or not, just as it can choose to implement individual specialization profiles or not. Clients interacting with a BES that implements a composition profile (and advertises the fact) will know precisely what behaviors to expect. Clients interacting with a BES that does not implement a composition profile – either because such a profile has not yet been defined or because the BES provides different composition semantics – can still interoperate with the BES. However, the set of potential behaviors a client might encounter will not be statically determinable and hence clients of such BESs will need to be prepared for a variety of different behaviors whose precise nature will only be determinable at interaction time with any given BES.

Consider next the composition of three profiles: the data staging profile, the suspend/resume profile, and a third migration profile. Figure 5 illustrates the migration profile, in which an activity in state "Running" can be migrated on request to another computational resource.

4.2.4 Representing Sub-states

A client can query a BES to determine a given activity's state. However, the question remains of how to represent that state. In particular, how should the compositional sub-states derived from different specialization profiles be represented? There is no obvious canonical way to represent the power set of sub-states that result from composing multiple specialization profiles unless one requires that an explicit composition profile be defined for every composition that a BES may ever implement.

Consequently a set-based "*union state*" approach is taken to representing sub-state information. For example, an activity that is in the process of staging data out and is also suspended, would be represented by a set of sub-state labels as follows: {Running;Stage-out, Running:Suspended}. Section 4.3 describes how union states should be encoded as XML elements.

Consider next the composition of three profiles: the data staging profile, the suspend/resume profile, and a third migration profile. Figure 5 illustrates the migration profile, in which an activity in state “Running” can be migrated on request to another computational resource.

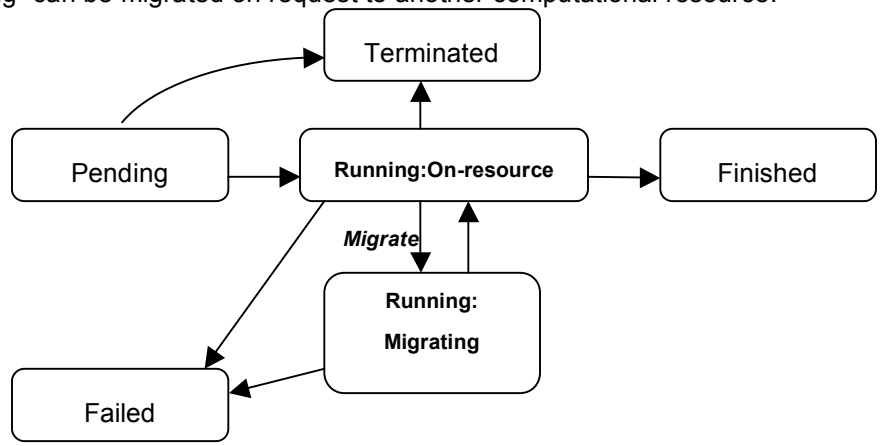

Figure 5. Migration specialization profile: Activities can be migrated upon client request to another computational resource.

Thus, an activity that is in the process of staging data out, and is both suspended and being migrated, would have this state: {Running;Stage-out, Running:Suspended, Running:Migrating}.

Suppose now that a composition profile exists that specifies that migration is inapplicable to the “Running:Stage-out” sub-state of the data staging profile and that a client who understands the data staging and migration profiles, but not the suspend/resume profile, wishes to determine whether requesting a *Migrate* operation is appropriate to a given activity in the “Running” state. If the client receives a union state of the form {Running:Suspended, Running;Stage-out} in response to a query request, it can examine each element of the set, ignoring those it does not recognize. It will skip over the “Running:Suspended” sub-state label and will recognize the “Running:Stage-out” sub-state label. Consequently it will be able to determine that a *Migrate* request would be inappropriate.

4.2.5 Multi-Level Specializations

This composition model allows for “component” specializations to be defined that can, for the most part, be mixed and matched with each other. However, some component specializations require definition relative to both the basic state model and other specializations. For example, consider a hold/release specialization that introduces the notion of being able to hold an activity from progressing to its next state until an explicit client *release* request is issued. Figure 6 illustrates this specialization. The semantics of *Hold* are that an activity will not progress from its “Running” state to a “Finished” or “Terminated state until a corresponding *Release* operation is issued by the client.

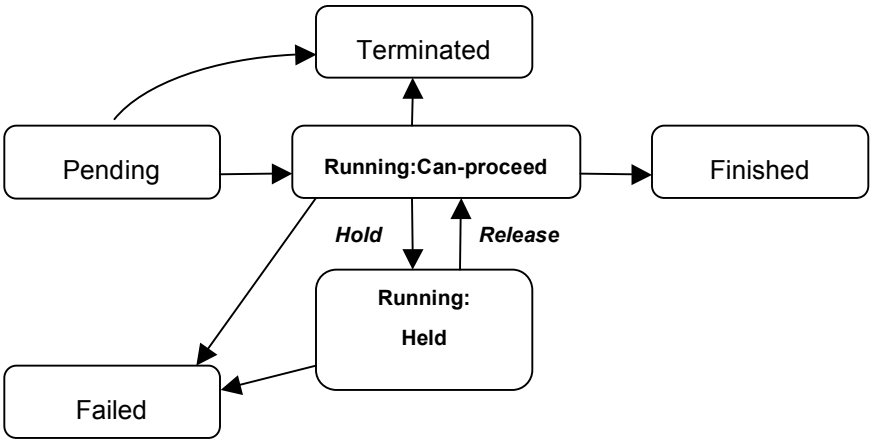

Figure 6. Hold/release specialization profile: Activities can be held from progressing to their next state until a client *release* request is issued.

Now suppose that a BES implements the data staging profile and that we would like to introduce the notion of being able to hold an activity after each sub-state of the data staging profile. Unfortunately the hold/resume specialization cannot be employed to achieve this aim since composition of the two profiles would only result in an implementation that would allow holding an activity from leaving its "Running:Stage-out" sub-state. To achieve the desired functionality would require defining a second hold/release specialization profile that is a specialization of the data staging profile rather than the basic state model.

4.2.6 Activity Sub-State XML Representation

While a simple state model—for example, one in which all states are represented by strings, enumerations, or other such basic types—might seem a reasonable choice, the fact that a single BES specification cannot anticipate all possible BES uses leads to a more complex state representation. BES instead represents state with a complex type that allows for the addition of sub-states. The example below shows the representation of the *Running* state in which the author has added a custom sub-state called *Staging-In*. In so doing, he has indicated that the target activity is in the basic *Running* state, but he has additionally indicated that it is also in the *Staging-In* sub-state of that *Running* state. This extra information may be of use to some clients, but need not be understood by other clients who care only that the activity is in the *Running* state.

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

4.3 Specialization Fault Responses

Given the fact that there can be multiple sub-states active at a given time, and given that clients can call operations that might not make sense given the current composite state of an Activity, two faults have been defined in order to indicate to clients when requested state transitions are either not allowed, or when a requested state change can only happen at a later time, when an Activity has transitioned out of a given sub-state.

The CantApplyOperationToCurrentStateFault is used to indicate that a requested state transition is not allowed given the current Activity state. For example, an Activity that is in a sub-state of "Running:Stage-out" can not be put into a "Running:Suspended" sub-state, since the Activity has actually finished executing.

The OperationWillBeAppliedEventuallyFault is used by the BES to indicate to the client that the requested state operation is allowed, but that the operation can not be applied immediately given the current Activity state. By throwing this fault, the BES indicates to the client that it will apply the requested operation when the Activity state allows. For example, an Activity in Running:Migrating

sub-state can not be put into Running:Suspended, until the Activity has completed the migration operation, and is back in the Running:On-resource sub-state.

5 BES-Management Port-type (Attributes and Operations)

The BES-Management port-type defines two operations for BES management. The intent of the BES-Management port-type is to define an interface for clients who are system administrators of a BES. Operations are specified using a combination of English and IDL. (A normative rendering is presented in Appendices A and C.)

5.1 BES-Management Operations

The following sections give (non-normatively) the total set of operations defined on the BES-Management port type. Normative information is provided in Appendix D.

5.1.1 StopAcceptingNewActivities

This operation is used to request that the BES stop accepting new activities.

5.1.1.1 Input(s)

- None.

5.1.1.2 Output(s)

- None.

5.1.1.3 Fault(s)

- **None.**

5.1.2 StartAcceptingNewActivities

This operation is used to request that the BES start accepting new activities.

5.1.2.1 Input(s)

- None.

5.1.2.2 Output(s)

- None.

5.1.2.3 Fault(s)

- **None.**

6 BES-Factory Port-Type

The BES-Factory port type contains operations that support the creation and manipulation of activities controlled or maintained by the BES. It also contains an operation for retrieving attribute information about the BES itself. Although one might argue that such attribute information, and the operation to retrieve it, belong in the BES-Management port-type, it is included here because the BES-Factory port-type is intended to be the interface that ordinary clients of a BES employ. Ordinary clients will often wish to inspect BES state in order to determine whether or not to employ a particular BES for executing activities.

6.1 Attributes

The BES specification may be used in a wide range of scenarios, including management of a single computer (such as a UNIX or Windows host); the management of clusters of resources through cluster management software such as LSF, Load Leveler, SGE, or PBS; and situations in which one BES service acts as a façade for one or more other BESs (i.e., hierarchical

arrangements of BES services). Quite different attributes may be useful in these different situations. Thus, this specification adopts a flexible and extensible mechanism for describing both backend resources, and front-end BES semantics.

Attributes are categorized as either Basic Resource or BES specific. Appendix B shows the schemas for this attribute document. All BESs provide a BES-specific attribute document. That document may (through the use of extensibility elements included in the schema) contain zero or more sub-documents, each corresponding to either contained BES attribute documents or Basic Resource attribute documents.

While the schema for these attributes is normatively given in the appendices of this document, a formal description and definition of those elements is given here (refer to Appendix B for cardinality and exact type). For each attribute, the abbreviation BES refers to a BES specific attribute while BR indicates a basic resource attribute (i.e., one that applies only to underlying resources).

Table 2: BES-Factory attributes

Name	Type	Number	Data Type	Description
IsAcceptingNewActivities	BES	1	Boolean	True if BES is accepting new activities
CommonName	BES	0 or 1	String	Short human-readable name for BES
LongDescription	BES	0 or 1	String	Longer human-readable description.
TotalNumberOfActivities	BES	1	Integer	Number of activities current active in BES
ActivityReference	BES	≥0	EPR	EPRs to activities currently active in BES
TotalNumberOfContainedResources	BES	1	Integer	Number of contained resources accessible by the BES
ContainedResource	BES	≥0	anyType	Currently expected to be either of type BasicResourceAttributesDocumentType or FactoryResourceAttributesDocumentType
NamingProfile	BES	≥1	URI	URI's of Naming profiles used by BES
BESExtension	BES	≥0	URI	URI's of supported BES extensions
LocalResourceManagerType	BES	1	URI	Resource's local resource manger type
ResourceName	BR	0 or 1	String	Resource's name
OperatingSystem	BR	0 or 1	String	Resource's operating system type
CPUArchitecture	BR	0 or 1	String	Resource's CPU architecture type
CPUcount	BR	0 or 1	Double	Resource's CPU count
CPUSpeed	BR	0 or 1	Double	Resource's CPU speed, in Hertz
PhysicalMemory	BR	0 or 1	Double	Resource's physical memory, in bytes
VirtualMemory	BR	0 or 1	Double	Resource's virtual memory, in bytes

6.1.1 IsAcceptingNewActivities

A BES attribute that Indicates whether or not the target BES is currently accepting new activities. If this value is false, then the BES MUST throw a NotAcceptingActivitiesFault fault for every CreateActivity request that it receives. If this value is true, then the BES MUST process every CreateActivity request that it receives.

6.1.2 CommonName

A BES attribute that indicates a common or human readable name for the service. The vagueness of this definition is intentional as the attribute is meant primarily for human consumption (for example, by users browsing with client tools). Likely values may include “machine.domain.name” and “My Organization’s Big Iron machine.”

6.1.3 LongDescription

A BES attribute meant as a human-readable string geared towards human consumption. Again, the purpose of this attribute is primarily for client side browsing capabilities and may include values such as “The big iron machine located in room 1138 in building THX” and “Mark’s personal desktop machine.”

6.1.4 TotalNumberOfActivities

This BES attribute indicates the total number of activities currently managed (regardless of the states of those activities) by the target BES.

6.1.5 ActivityReference

A BES attribute that lists all of the WS-Addressing endpoint reference types for all of the currently contained activities.

6.1.6 TotalNumberOfContainedResources

This BES attribute indicates the total number of contained resources that are accesible by the BES factory.

6.1.7 ContainedResource

This attribute is a list of sub-attribute documents whose types must all either be bes:BasicResourceAttributeDocumentType, or bes:FactoryResourceAttributeDocumentType. This attribute is itself BES specific, thus implying that BasicResourceAttributeDocumentTypes are leaf nodes in an attribute document hierarchy.

6.1.8 NamingProfile

A BES attribute that indicates what types of Endpoint References this BES returns from it’s CreateActivity operation. Valid values include (but are not limited to):

- http://schemas.ggf.org/bes/2006/08/bes/naming/BasicWSAddressing
- http://www.ogf.org/naming/2006/08/naming-uwsep-pf

6.1.9 BESExtension

A BES attribute that indicates what, if any, BES extensions are supported. Valid values include (but are not limited to):

- http://schemas.ggf.org/bes/2006/08/bes-extensions/IdempotentActivityIDLifetime
- http://schemas.ggf.org/bes/2006/08/bes-extensions/SupportsSubscriptions
- http://schemas.ggf.org/bes/2006/08/bes-extensions/SupportsLifetimes

6.1.10 LocalResourceManagerType

This attribute is both a BES and a BR attribute type which indicates (where applicable) the type of resource/resources that is/are being managed. For example, the values may indicate that the resources are single UNIX or Windows machines, clusters of machines behind LSF, LoadLeveler, SGE, or PBS queues, etc. Valid values for this attribute are URIs which SHOULD be defined on a per-resource type basis. In other words, it is expected that various companies participating in the BES specification will provide unique, company-specific and product specific namespaces for URIs identifying their resource types. Examples might include:

- http://vcgr.cs.virginia.edu/bes/2006/08/resources/queuing
- http://sun.com/bes/2006/08/resources/queues/SGE

6.1.11 ResourceName

A BR attribute that is used to name a particular instance of a ContainedResource. This attribute MAY be used to provide the FQDN of a resource that is being made accessible via the BES factory.

6.1.12 OperatingSystem

A BR attribute that indicates the operating system information for the managed resource. The type for this attribute is taken directly from the JSDL **[JSDL 1.0]** documentation.

1.1.3 CPUArchitecture

A BR attribute that indicates the architecture of the machine for this managed resource. The type for this attribute is taken directly from the JSDL **[JSDL 1.0]** documentation.

1.1.4 CPUCount

A BR attribute that indicates the number of CPUs managed or provided by the target resource. This number is equal to the total number of CPUs, cores, etc. available on the resource indicated.

1.1.5 CPUSpeed

An attribute available in the BR document which indicates the clock speed of an individual CPU on the resource given. This value is in units of Hertz.

1.1.6 PhysicalMemory

A BR attribute giving the total amount of memory (in bytes) managed by the indicated resource.

1.1.7 VirtualMemory

A BR attribute giving the total amount of memory (in bytes) available as swap or virtual memory on the indicated resource.

1.2 BES-Factory Operations

The following sections non-normatively describe the operations supported by the BES-Factory port type. Normative descriptions are provided in Appendix E.

The GetActivityStatuses, TerminateActivities, and GetActivityDocuments operations take a vector of activity EPRs as input and operate on all the referenced activities. Since an input vector may contain EPRs that are either unknown to a BES or for which the BES cannot execute the requested operation, the following failure semantics MUST be provided by a compliant BES:

- If a request fails for some reason that applies to all the specified activities—e.g., due to an authorization fault—then the BES MUST respond with an appropriate fault response message.
- If a request can succeed for one or more of the specified activities then the BES MUST respond with a vector of response elements, where each element corresponds to the

equivalent activity designated in the input EPR vector. Each response element MUST be either an element describing the results of the request, as applied to the designated activity, or a SOAP-1.1 fault element describing why the request could not be applied to the designated activity (e.g., because the EPR could not be resolved to any known activity within the BES).

1.2.1 CreateActivity

This operation is used to request the creation of a new activity. In the following subsections, we define this operation's inputs, outputs, and faults.

1.2.1.1 Input(s)

- **ActivityDocumentType ActivityDocument**: An XML document describing a single activity that is to be created.

The ActivityDocument element is used to describe either the desired or the actual representation of a single activity and has the following structure. In all cases, a jsdl:JobDefinition element MUST be present. In a CreateActivity request this element describes desired/required aspects of the activity to be created. When returned in response messages for GetActivityStatuses and GetActivityDocuments requests the element describes the current representation of a existing activity. Other elements may also be present in an ActivityDocument, for example, to describe desired/actual aspects of an activity that correspond to one or more BES extension specifications.

```
<bes:ActivityDocument>
  <jsdl:JobDefinition>
    ...
  </jsdl:JobDefinition>
  <xsd:any/> *
</bes:ActivityDocument>
```

1.2.1.2 Output(s)

- **CreateActivityResponseType Response**: On success, Response contains an ActivityIdentifier (EPR) identifying the requested activity. An ActivityDocument element MAY be included representing the current representation of the requested activity.

1.2.1.3 Fault(s)

- **NotAuthorizedFault:** Indicates that while the front-end (i.e. BES web service) was able to validate the incoming user credentials, the back-end would not permit the operation given the credentials supplied. This can happen for example when a BES implements its activities by fork/exec'ing those applications using **su**.
- **NotAcceptingNewActivities**: A fault that indicates that the BES is not currently accepting new activities.
- **UnsupportedFeatureFault**: A fault indicating either**:**
    - A well-formed, supported JSDL document input element containing a sub-element that is not implemented by this BES implementation.
    - A non-JSDL input element that is not implemented by this BES implementation.

  The features that are not implemented are described in the body of the fault.
- **InvalidRequestMessageFault:** An element in the request message is not recognized. The elements that are not recognized are described in the body of the fault. This does not mean that the element itself is in error, but rather that it specifies a syntactically correct value which does not in fact make sense. For example, the number of CPUs is represented by a double, so fractional values are syntactically correct, but would cause this fault to be thrown as they do not make sense in the context given.

1.2.2 GetActivityStatuses

This operation allows a client to request the status of zero or more activities previously initiated by CreateActivity operations. Each activity for which status information is required is specified by an EPR; the response is a vector of ActivityStatus elements, one per requested activity.

1.2.2.1 Input(s)

- **EPR[] ActivityIdentifier**: A vector of zero or more EPRs (obtained from previous CreateActivity operations), indicating activities from which we require status information.

1.2.2.2 Output(s)

- **GetActivityStatusResponseType[] Response**: An XML document containing a vector of ActivityStatus elements (see below), one for each EPR provided in the input. Note that because many activities can be queried, this document may become large.

The ActivityStatus element used to represent the status of a single activity has the following structure. Its two components are the EPR for the activity and either a description of the overall state of the activity referenced by that EPR (OverallStatus) or a SOAP-1.1 fault element. This SOAP fault may indicate that the activity identifier was unknown with the InvalidActivityIdentifier fault message, or it can also indicate a back end security exception with the NotAuthorizedFault. The NotAuthorizedFault in particular may be indicated when the container allows a query operation to take place, but the activity in question is protected from a given user.

```
<ActivityStatus>
   <ActivityIdentifier> {wsa:EndpointReferenceType} </ActivityIdentifier>
   <ActivityStatus> ActvityStateType </ActivityStatus> ?
   |
   <SOAP-1.1:fault> fault </SOAP-1.1:fault> ?
</ActivityStatus>*
```

Since the BES specification allows for extensible activity state diagrams, it is possible that not all states within the state diagram will be relevant/meaningful to a particular client. BES requires that all legal state transitions are transitioned even if they are not relevant to a particular client. For instance, if an empty JSDL document is submitted to the BES then all the states from *New* to *Finished* will be transitioned through even though there is no underlying specified activity.

1.2.2.3 Fault(s)

- **InvalidRequestMessageFault:** An element in the request message is not recognized. The elements that are not recognized are described in the body of the fault. This does not mean that the element itself is in error, but rather that it specifies a syntactically correct value which does not in fact make sense. For example, the number of CPUs is represented by a double, so fractional values are syntactically correct, but would cause this fault to be thrown as they do not make sense in the context given.

1.2.3 TerminateActivities

This operation requests that a specified set of activities be terminated. The BES attempts to terminate each activity specified. As a consequence of this operation, the specified activities MAY be terminated. If an activity cannot be terminated immediately, the eventual success of this operation (i.e., to move the activity into the *Terminated* state) must be determined through other operations (e.g., GetActivityState) or by subscribing to any generated events, if a BES supports subscription.

If a request is successful, then each specified activity will eventually enter the *Terminated* state. Invoking this operation on a *Terminated* activity has no further effect. How long the activity remains in the *Terminated* state before the EPR no longer returns a reference to the activity is not defined.

1.2.3.1 Input(s)

- **EPR[] activities**: A vector of zero or more EPRs identifying the activities that are to be terminated.

1.2.3.2 Output(s)

- **TerminateActivityResponseType[] Response**: A vector detailing the responses of the BES to the termination requests. The Terminated element is a boolean value indicating whether the BES successfully (true) terminated the activity or not (false). If true is returned, then the associated activity is now in the *Terminated* state. If false is returned then the activity MAY eventually transition to the *Terminated* state. If an activity specified in the input cannot be located or cannot be terminated for some reason then the TerminateResponse MUST contain a SOAP-1.1 fault element instead of a Terminated element.

```
<TerminateActivityResponse>
   <ActivityIdentifier>EPR</ActivityIdentifier>
   <Cancelled> xsd:Boolean </Cancelled> ?
   |
   <SOAP-1.1:fault> ... </SOAP-1.1:fault> ?
</TerminateActivityResponse> *
```

1.2.3.3 Fault(s)

- **InvalidRequestMessageFault:** An element in the request message is not recognized. The elements that are not recognized are described in the body of the fault. This does not mean that the element itself is in error, but rather that it specifies a syntactically correct value which does not in fact make sense. For example, the number of CPUs is represented by a double, so fractional values are syntactically correct, but would cause this fault to be thrown as they do not make sense in the context given.

This operation requests ActivityDocument descriptions for a set of specified activities. These ActivityDocuments may be different from those initially passed to the BES in the CreateActivity operation, as the BES may alter its contents to reflect policy or process within the service. That is, the ActivityDocument for a given activity reflects the activity being run as opposed to what the BES was originally asked to run. (If the submitter wishes to access the latter document, they should retain their own copy.)

1.2.3.4 Input(s)

- **EPR[] ActivityIdentifiers**: A vector of EPRs identifying the activities for which ActivityDocuments are requested.

1.2.3.5 Output(s)

- **GetActivityDocumentResponseType [] Response**. The output from this operation returns a vector of ActivityDocumentResponse elements.

The ActivityDocumentResponse element used to represent information about a single activity has the following structure. Its two components are the EPR for the activity and either an ActivityDocument for the activity referenced by that EPR or a SOAP-1.1 fault element. This fault element might be either an InvalidActivityIdentifier fault indicating that the request activity was not recognized by this container, or it might be a NotAuthorizedFault indicating that the client in question, while possessing sufficient credentials to query the container, does not have permission to view the indicated activitie's Activity Document.

```
<GetActivityDocumentResponse >
   <ActivityIdentifier> {wsa:EndpointReferenceType} </ActivityIdentifier>
   <ActivityDocument> {jsdl:JobDefinition} </ActivityDocument> ?
   |
   <SOAP-1.1:fault> fault </SOAP-1.1:fault> ?
```

</GetActivityDocumentResponse >*

1.2.3.6 Fault(s)

- **InvalidRequestMessageFault:** An element in the request message is not recognized. The elements that are not recognized are described in the body of the fault. This does not mean that the element itself is in error, but rather that it specifies a syntactically correct value which does not in fact make sense. For example, the number of CPUs is represented by a double, so fractional values are syntactically correct, but would cause this fault to be thrown as they do not make sense in the context given.

1.2.4 GetAttributesDocument

This operation is used to request a document containing the BES-Management attributes.

1.2.4.1 Input(s)

None

1.2.4.2 Output(s)

- **BESResourceAttributesDocumentType BESResourceAttributesDocument**: An XML document containing the various attributes listed above in the attributes section. This document MUST be of type bes-factory:BESResourceAttributeDocumentType.

1.2.4.3 Fault(s)

- **InvalidRequestMessageFault:** An element in the request message is not recognized. The elements that are not recognized are described in the body of the fault. This does not mean that the element itself is in error, but rather that it specifies a syntactically correct value which does not in fact make sense. For example, the number of CPUs is represented by a double, so fractional values are syntactically correct, but would cause this fault to be thrown as they do not make sense in the context given.

2 BES-Activity Port-type

The BES-Activity port-type defines operations for monitoring and managing individual activities. These operations are intended to be applied to EPRs returned by previous CreateActivity operations. Returning an EPR that supports the BES-Activity Port-type is optional in the CreateActivity operation of the BES-Factory Port-type.

2.1 BES-Activity Attributes

An implementation of the BES-Activity port-type MUST support the attributes listed in Table 3, so that clients can determine current activity status.

Table 3: BES-Activity attributes

Name	N	Type	Description
Status	1	bes-factory:ActivityStatusType	Status of the activity
ActivityDocument	1	bes-factory:ActivityDocumentType	Document containing activity's JSDL document and resource-specific parameters
FactoryReference	1	wsa:EndpointReferenceType	EPR to the BES-Factory resource that created this activity

2.2 BES-Activity Operations

No operations are defined for the BES-Activity port type.

3 Optional Extensions

To determine supported extension one can check the attribute BESExtension.

3.1 Idempotent Execution Semantics

If idempotent execution is required, the following element MUST be used to uniquely identify an activity to the BES using a client-generated identifier. If present in an activity description, the BES MUST NOT execute the activity containing this identifier more than once.

```
<bes-ext:IdempotentActivityID>
    wsa:AttributedURI
</bes-ext:IdempotentActivityID>
```

Should a BES receive a second CreateActivity request that includes the same identifier as a previously received request, the BES MUST not create the requested activity a second time if it already created the activity for the first request. Furthermore, if the first request resulted in a created activity then the BES MUST return the same response message as was returned to the first CreateActivity request.

By default, the lifetime of the identifier SHOULD be equal to the lifetime of the activity. A TerminateAcitvities request SHOULD, by default, also invalidate the identifier. This may not be acceptable depending on the BES implementation and/or client logic. The activity requester MAY request an explicit identifier lifetime by specifying the following additional extension element:

```
<bes-ext:IdempotentActivityIDLifetime>
    xsd:dateTime
</bes-ext:IdempotentActivityIDLifetime>
```

3.2 Subscription to Notification Events

A BES that allows its clients to subscribe for messages concerning activity state changes MUST do so using either the WS-Eventing or WS-Notification protocols.

To avoid both additional round-trip messages and the race-condition inherent in first creating and then subscribing to an activity's state changes, compliant BESs implementing this extension MUST support the (optional) inclusion of one of the following subscription request elements as an extension to the ActivityDocument element in the input to the CreateActivity request:

```
<wsnt:Subscribe/>
      OR
<wse:Subscribe/>
```

If either of these extensions is specified, an ActivityDocument MUST appear in the response to CreateActivity. Additionaly, one of the following associated subscription reference elements MUST be added as an extension to the ActivityDocument element returned in the response to CreateActivity:

```
<wsnt:SubscribeResponse/>
      OR
<wse:SubscribeResponse/>
```

The subscription reference element SHOULD also be included in the actual ActivityDocument used by the activity so that activity resource attributes and/or operations which are out of scope of this specification are consistent with the ActivityDocument returned by CreateActivity.

3.3 Lifetime Management

A BES that implements the WS-ResourceLifetime operation SHOULD use the following element to indicate the requestor's suggestion for the initial setting of the termination time resource property [WS-ResourceLifetime] of the activity:

```
<bes-ext:InitialTerminationTime>
    xsd:dateTime
</bes-ext:InitialTerminationTime>
```

This value SHOULD NOT be assumed to be linked with any other limits specified in the ActivityDocument (e.g., jsdl-posix:WallTimeLimit). The value of the element is to be interpreted as an "absolute time", That said time in general is assumed to be relative to the time source used by the BES implementation.

If the BES implementation is unable or unwilling to set the TerminationTime attribute of the activity to the given value or a value greater, then the CreateActivity request MUST fault. If the value is not "in the future" relative to the current time as known by the BES implementation, the request MUST fault. The use of the xsi:nil attribute with value "true" indicates there is no scheduled termination time requested for the activity. If the element does not include the time zone designation, the value of the element MUST be interpreted with respect to universal time (UTC). If a fault is returned, the operation MUST NOT have an effect.

If this element is not included, the initial value of the TerminationTime resource property is dependent on the BES implementation.

4 Management

Currently no generic management infrastructure has been specified for OGSA services. We would expect such an infrastructure to support the termination of a BES following a duly authorized request. In terminating a BES, the impact on the activities taking place within the BES is undefined.

5 Security Considerations

Security considerations are significant in execution management. A BES will typically want to control who can invoke its various operations. There may also be a need for identity mapping between a grid credential and the BES's authorization and authentication space. These considerations are outside the scope of this document.

6 Author Information

Ian Foster
Globus
Argonne National Labs
foster@mcs.anl.gov

Andrew Grimshaw
Global Bio Grid
University of Virginia
grimshaw@cs.virginia.edu

Peter Lane
Globus
Argonne National Labs
lane@mcs.anl.gov

William Lee
London eScience
Imperial College London

Steven Newhouse
OMII-UK
Imperial College London
s.newhouse@omii.ac.uk

Stephen Pickles
Manchester Computing
University of Manchester
stephen.pickles@manchester.ac.uk

Darren Pulsipher
EMC
Pulsipher_Darren@emc.com

Chris Smith
Platform Computing
csmith@platform.com

Marvin Theimer
Microsoft
theimer@microsoft.com

7 Contributors

We gratefully acknowledge the contributions made to this document by

8 Acknowledgments

We are grateful to numerous colleagues for discussions on the topics covered in this document, and to the people who provided comments on the public drafts. Thanks in particular to (in alphabetical order, with apologies to anybody we have missed) ….

9 Intellectual Property Statement

10 Full Copyright Notice

11 References

[RFC2119] S.Bradner, Key words for use in RFCs to Indicate Requirement Levels, http://www.ietf.org/rfc/rfc2119.txt, IETF RFC 2119, March 1997.

[XML-Infoset] XML Information Set (Second Edition) W3C Recommendation 4 February 2004 - http://www.w3.org/TR/xml-infoset/

[XPATH] XML Path Language (XPath) Version 1.0 W3C Recommendation, 16 November 1999 -http://www.w3.org/TR/xpath

[WS-Addressing] D. Box and F. Curbera (ed.) Web Services Addressing 1.0 – Core (WS-Addressing), W3C Last Call 31 March 2005, http://www.w3.org/TR/2005/WD-ws-addr-core-20050331

[SOAP 1.1] Simple Object Access Protocol (SOAP) 1.1 , W3C 08 May 2000 – http://www.w3.org/TR/soap11

[OGSA WSRF BP] OGSA WSRF Basic Profile definition 1.0, GGF February, 2006

[WS-Security] Web Services Security (WS-Security), Version 1.0 05 April 2002 - http://www-128.ibm.com/developerworks/webservices/library/ws-secure/

[WS-Naming] WS-Naming Specification 1.0, GGF February 2006

[JSDL 1.0] Job Service Description Language 1.0, GGF November 2005

Appendix A. Normative BES-Management XSD

None needed.

Appendix B. Normative BES-Factory XSD

```
<xsd:schema
    targetNamespace="http://schemas.ggf.org/bes/2006/08/bes-factory"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:bes-factory="http://schemas.ggf.org/bes/2006/08/bes-factory"
    xmlns:jsdl="http://schemas.ggf.org/jsdl/2005/11/jsdl"
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">

  <xsd:import
    namespace="http://schemas.xmlsoap.org/soap/envelope/"
    schemaLocation="http://schemas.xmlsoap.org/soap/envelope/"/>

  <xsd:import
      namespace="http://www.w3.org/2005/08/addressing"
      schemaLocation="http://www.w3.org/2002/ws/addr/ns/ws-addr"/>

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

  <!-- Attribute Document Types -->
  <xsd:complexType name="BasicResourceAttributesDocumentType">
    <xsd:sequence>
      <xsd:element name="ResourceName" type="xsd:string" minOccurs="0"/>
      <xsd:element name="OperatingSystem" type="jsdl:OperatingSystem_Type"
minOccurs="0"/>
      <xsd:element name="CPUArchitecture" type="jsdl:CPUArchitecture_Type"
minOccurs="0"/>
      <xsd:element name="CPUCount" type="xsd:double" minOccurs="0"/>
      <xsd:element name="CPUSpeed" type="xsd:double" minOccurs="0"/>
      <xsd:element name="PhysicalMemory" type="xsd:double" minOccurs="0"/>
      <xsd:element name="VirtualMemory" type="xsd:double" minOccurs="0"/>
      <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:anyAttribute namespace="##other" processContents="lax"/>
  </xsd:complexType>

  <xsd:complexType name="FactoryResourceAttributesDocumentType">
    <xsd:sequence>
      <xsd:element ref="bes-factory:BasicResourceAttributesDocument" minOccurs="0"/>
      <xsd:element name="IsAcceptingNewActivities" type="xsd:boolean"/>
      <xsd:element name="CommonName" type="xsd:string" minOccurs="0" maxOccurs ="1"/>
      <xsd:element name="LongDescription" type="xsd:string" minOccurs="0" maxOccurs
="1"/>
      <xsd:element name="TotalNumberOfActivities" type="xsd:long"/>
      <xsd:element name="ActivityReference" type="wsa:EndpointReferenceType"
minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element name="TotalNumberOfContainedResources" type="xsd:long"/>
      <xsd:element name="ContainedResource" type="xsd:anyType" minOccurs="0"
maxOccurs="unbounded"/>
      <xsd:element name="NamingProfile" type="xsd:anyURI" maxOccurs="unbounded"/>
      <xsd:element name="BESExtension" type="xsd:anyURI" minOccurs="0"
maxOccurs="unbounded"/>
      <xsd:element name="LocalResourceManagerType" type="xsd:anyURI"/>
      <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:anyAttribute namespace="##other" processContents="lax"/>
  </xsd:complexType>

  <!-- Message Helper Types -->
  <xsd:complexType name="ActivityDocumentType">
    <xsd:sequence>
```

```
      <xsd:element ref="jsdl:JobDefinition"/>
      <xsd:any namespace="##other" processContents="lax"
          minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:anyAttribute namespace="##other" processContents="lax"/>
  </xsd:complexType>

  <xsd:simpleType name="ActivityStateEnumeration">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="Pending"/>
      <xsd:enumeration value="Running"/>
      <xsd:enumeration value="Cancelled"/>
      <xsd:enumeration value="Failed"/>
      <xsd:enumeration value="Finished"/>
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:complexType name="ActivityStatusType">
    <xsd:sequence>
      <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="state" type="bes-factory:ActivityStateEnumeration"
use="required"/>
    <xsd:anyAttribute namespace="##other" processContents="lax"/>
  </xsd:complexType>

  <xsd:complexType name="GetActivityStatusResponseType">
    <xsd:sequence>
      <xsd:element name="ActivityIdentifier" type="wsa:EndpointReferenceType"/>
      <xsd:element ref="bes-factory:ActivityStatus" minOccurs="0"/>
      <xsd:element name="Fault" type="soap:Fault" minOccurs="0"/>
      <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:anyAttribute namespace="##other" processContents="lax"/>
  </xsd:complexType>

  <xsd:complexType name="GetActivityDocumentResponseType">
    <xsd:sequence>
      <xsd:element name="ActivityIdentifier" type="wsa:EndpointReferenceType"/>
      <xsd:element name="JobDefinition" type="jsdl:JobDefinition_Type" minOccurs="0"/>
      <xsd:element name="Fault" type="soap:Fault" minOccurs="0"/>
      <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:anyAttribute namespace="##other" processContents="lax"/>
  </xsd:complexType>

  <xsd:complexType name="TerminateActivityResponseType">
    <xsd:sequence>
      <xsd:element name="ActivityIdentifier" type="wsa:EndpointReferenceType"/>
      <xsd:element name="Cancelled" type="xsd:boolean"/>
      <xsd:element name="Fault" type="soap:Fault" minOccurs="0"/>
      <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:anyAttribute namespace="##other" processContents="lax"/>
  </xsd:complexType>

  <!-- Message Helper Elements -->
  <xsd:element name="BasicResourceAttributesDocument"
          type="bes-factory:BasicResourceAttributesDocumentType"/>
  <xsd:element name="FactoryResourceAttributesDocument"
          type="bes-factory:FactoryResourceAttributesDocumentType"/>
  <xsd:element name="ActivityDocument"
          type="bes-factory:ActivityDocumentType"/>
  <xsd:element name="ActivityStatus"
          type="bes-factory:ActivityStatusType"/>

</xsd:schema>
```

Appendix C. Normative BES-Activity XSD

```
<xsd:schema
    targetNamespace="http://schemas.ggf.org/bes/2006/08/bes-activity"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:bes-factory="http://schemas.ggf.org/bes/2006/08/bes-factory"
    xmlns:bes-activity="http://schemas.ggf.org/bes/2006/08/bes-activity"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">

    <xsd:import
        namespace="http://www.w3.org/2005/08/addressing"
        schemaLocation="http://www.w3.org/2002/ws/addr/ns/ws-addr"/>

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

    <xsd:complexType name="ActivityResourceAttributesDocumentType">
        <xsd:sequence>
            <xsd:element name=”Status” type="bes-factory:ActivityStatusType"/>
            <xsd:element name=”ActivityDocument”
                type="bes-factory:ActivityDocumentType"/>
            <xsd:element name="FactoryReference" type="wsa:EndpointReferenceType"/>
            <xsd:any namespace=”##other” processContents=”lax”
                minOccurs=”0” maxOccurs=”unbounded”/>
        </xsd:sequence>
        <xsd:anyAttribute namespace=”##other” processContents=”lax”/>
    </xsd:complexType>

    <xsd:element name=”ActivityResourceAttributesDocument”
        type="bes-activity:ActivityResourceAttributesDocumentType”/>

</xsd:schema>
```

Appendix D. Normative BES-Management WSDL

```
<?xml version="1.0" encoding="utf-8" ?>
<wsdl:definitions name="BESManagement"
    targetNamespace="http://schemas.ggf.org/bes/2006/08/bes-management"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:bes-mgmt="http://schemas.ggf.org/bes/2006/08/bes-management"
    xmlns:wsa="http://www.w3/org/2005/08/addressing"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" >

  <wsdl:types>
    <xsd:schema
        attributeFormDefault="unqualified"
        elementFormDefault="qualified"
        targetNamespace="http://schemas.ggf.org/bes/2006/08/bes-management">

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

      <!-- Message Types -->
      <xsd:complexType name="StopAcceptingNewActivitiesType">
        <xsd:sequence>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="StopAcceptingNewActivitiesResponseType">
        <xsd:sequence>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="StartAcceptingNewActivitiesType">
        <xsd:sequence>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="StartAcceptingNewActivitiesResponseType">
        <xsd:sequence>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <!-- Message Elements -->
      <xsd:element name="StopAcceptingNewActivities"
        type="bes-mgmt:StopAcceptingNewActivitiesType"/>
      <xsd:element name="StopAcceptingNewActivitiesResponse"
        type="bes-mgmt:StopAcceptingNewActivitiesResponseType"/>
      <xsd:element name="StartAcceptingNewActivities"
        type="bes-mgmt:StartAcceptingNewActivitiesType"/>
      <xsd:element name="StartAcceptingNewActivitiesResponse"
        type="bes-mgmt:StartAcceptingNewActivitiesResponseType"/>
    </xsd:schema>
  </wsdl:types>

  <!-- Messages -->
```

```
  <wsdl:message name="StopAcceptingNewActivitiesRequest">
    <wsdl:part name="parameters"
        element="bes-mgmt:StopAcceptingNewActivities"/>
  </wsdl:message>

  <wsdl:message name="StopAcceptingNewActivitiesResponse">
    <wsdl:part name="parameters"
        element="bes-mgmt:StopAcceptingNewActivitiesResponse"/>
  </wsdl:message>

  <wsdl:message name="StartAcceptingNewActivitiesRequest">
    <wsdl:part name="parameters"
        element="bes-mgmt:StartAcceptingNewActivities"/>
  </wsdl:message>

  <wsdl:message name="StartAcceptingNewActivitiesResponse">
    <wsdl:part name="parameters"
        element="bes-mgmt:StartAcceptingNewActivitiesResponse"/>
  </wsdl:message>

  <!-- Port Type -->
  <wsdl:portType name="BESManagementPortType">

    <wsdl:operation name="StopAcceptingNewActivities">
      <wsdl:input
        name="StopAcceptingNewActivities"
        message="bes-mgmt:StopAcceptingNewActivitiesRequest"
        wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
management/BESManagementPortType/StopAcceptingNewActivities"/>
      <wsdl:output
        name="StopAcceptingNewActivitiesResponse"
        message="bes-mgmt:StopAcceptingNewActivitiesResponse"
        wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
management/BESManagementPortType/StopAcceptingNewActivitiesResponse"/>
    </wsdl:operation>

    <wsdl:operation name="StartAcceptingNewActivities">
      <wsdl:input
        name="StartAcceptingNewActivities"
        message="bes-mgmt:StartAcceptingNewActivitiesRequest"
        wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
management/BESManagementPortType/StartAcceptingNewActivities"/>
      <wsdl:output
        name="StartAcceptingNewActivitiesResponse"
        message="bes-mgmt:StartAcceptingNewActivitiesResponse"
        wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
management/BESManagementPortType/StartAcceptingNewActivitiesResponse"/>
    </wsdl:operation>
  </wsdl:portType>

  <!-- Bindings -->
  <wsdl:binding name="BESManagementBinding" type="bes-mgmt:BESManagementPortType">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />

    <wsdl:operation name="StopAcceptingNewActivities">
      <soap:operation soapAction="http://schemas.ggf.org/bes/2006/08/bes-
management/BESManagementPortType/StopAcceptingNewActivities" />
      <wsdl:input name="StopAcceptingNewActivities">
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="StopAcceptingNewActivitiesResponse">
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>

    <wsdl:operation name="StartAcceptingNewActivities">
      <soap:operation soapAction="http://schemas.ggf.org/bes/2006/08/bes-
management/BESManagementPortType/StartAcceptingNewActivities" />
      <wsdl:input name="StartAcceptingNewActivities">
        <soap:body use="literal" />
      </wsdl:input>
```

```
      <wsdl:output name="StartAcceptingNewActivitiesResponse">
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
</wsdl:definitions>
```

Appendix E. Normative BES-Factory WSDL

```
<wsdl:definitions name="BESFactory"
    targetNamespace="http://schemas.ggf.org/bes/2006/08/bes-factory"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:bes-factory="http://schemas.ggf.org/bes/2006/08/bes-factory"
    xmlns:wsa="http://www.w3/org/2005/08/addressing"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" >

  <wsdl:types>
    <xsd:schema
        xmlns:wsa="http://www.w3.org/2005/08/addressing"
        attributeFormDefault="unqualified"
        elementFormDefault="qualified"
        targetNamespace="http://schemas.ggf.org/bes/2006/08/bes-factory">

      <xsd:include schemaLocation="bes-factory.xsd"/>

      <!-- Message Types -->
      <xsd:complexType name="CreateActivityType">
        <xsd:sequence>
          <xsd:element ref="bes-factory:ActivityDocument"/>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="CreateActivityResponseType">
        <xsd:sequence>
          <xsd:element name="ActivityIdentifier" type="wsa:EndpointReferenceType"/>
          <xsd:element ref="bes-factory:ActivityDocument" minOccurs="0"/>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="GetActivityStatusesType">
        <xsd:sequence>
          <xsd:element name="ActivityIdentifier" type="wsa:EndpointReferenceType"
maxOccurs="unbounded" minOccurs="0"/>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="GetActivityStatusesResponseType">
        <xsd:sequence>
          <xsd:element name="Response" type="bes-factory:GetActivityStatusResponseType"
maxOccurs="unbounded" minOccurs="0"/>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="TerminateActivitiesType">
        <xsd:sequence>
          <xsd:element name="ActivityIdentifier" type="wsa:EndpointReferenceType"
minOccurs="0" maxOccurs="unbounded"/>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>
```

```
      <xsd:complexType name="TerminateActivitiesResponseType">
        <xsd:sequence>
          <xsd:element name="Response" type="bes-factory:TerminateActivityResponseType"
minOccurs="0" maxOccurs="unbounded"/>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="GetActivityDocumentsType">
        <xsd:sequence>
          <xsd:element name="ActivityIdentifier" type="wsa:EndpointReferenceType"
minOccurs="0" maxOccurs="unbounded"/>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="GetActivityDocumentsResponseType">
        <xsd:sequence>
          <xsd:element name="Response" type="bes-factory:GetActivityDocumentResponseType"
minOccurs="0" maxOccurs="unbounded"/>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="GetFactoryAttributesDocumentType">
        <xsd:sequence>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="GetFactoryAttributesDocumentResponseType">
        <xsd:sequence>
          <xsd:element ref="bes-factory:FactoryResourceAttributesDocument"/>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <!-- Fault Types  -->
      <xsd:complexType name="NotAuthorizedFaultType">
        <xsd:sequence>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="NotAcceptingNewActivitiesFaultType">
        <xsd:sequence>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="UnsupportedFeatureFaultType">
        <xsd:sequence>
          <xsd:element name="Feature" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
```

```
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="CantApplyOperationToCurrentStateFaultType">
        <xsd:sequence>
          <xsd:element name="ActivityStatus" type="bes-factory:ActivityStatusType" />
          <xsd:element name="Message" type="xsd:string"/>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="OperationWillBeAppliedEventuallyFaultType">
        <xsd:sequence>
          <xsd:element name="ActivityStatus" type="bes-factory:ActivityStatusType" />
          <xsd:element name="Message" type="xsd:string"/>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="InvalidActivityIdentifierFaultType">
        <xsd:sequence>
          <xsd:element name="Message" type="xsd:string"/>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <xsd:complexType name="InvalidRequestMessageFaultType">
        <xsd:sequence>
          <xsd:element name="InvalidElement" type="xsd:string" minOccurs="0" maxOccurs
="unbounded"/>
          <xsd:element name="Message" type="xsd:string"/>
          <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>

      <!-- Message Elements -->
      <xsd:element name="CreateActivity"
        type="bes-factory:CreateActivityType"/>
      <xsd:element name="CreateActivityResponse"
        type="bes-factory:CreateActivityResponseType"/>
      <xsd:element name="GetActivityStatuses"
        type="bes-factory:GetActivityStatusesType"/>
      <xsd:element name="GetActivityStatusesResponse"
        type="bes-factory:GetActivityStatusesResponseType"/>
      <xsd:element name="TerminateActivities"
        type="bes-factory:TerminateActivitiesType"/>
      <xsd:element name="TerminateActivitiesResponse"
        type="bes-factory:TerminateActivitiesResponseType"/>
      <xsd:element name="GetActivityDocuments"
        type="bes-factory:GetActivityDocumentsType"/>
      <xsd:element name="GetActivityDocumentsResponse"
        type="bes-factory:GetActivityDocumentsResponseType"/>
      <xsd:element name="GetFactoryAttributesDocument"
        type="bes-factory:GetFactoryAttributesDocumentType"/>
      <xsd:element name="GetFactoryAttributesDocumentResponse"
        type="bes-factory:GetFactoryAttributesDocumentResponseType"/>

      <!-- Fault Elements -->
      <xsd:element name="NotAuthorizedFault"
        type="bes-factory:NotAuthorizedFaultType"/>
      <xsd:element name="NotAcceptingNewActivitiesFault"
        type="bes-factory:NotAcceptingNewActivitiesFaultType"/>
```

```
    <xsd:element name="UnsupportedFeatureFault"
      type="bes-factory:UnsupportedFeatureFaultType"/>
    <xsd:element name="CantApplyOperationToCurrentStateFault"
      type="bes-factory:CantApplyOperationToCurrentStateFaultType"/>
    <xsd:element name="OperationWillBeAppliedEventuallyFault"
      type="bes-factory:OperationWillBeAppliedEventuallyFaultType"/>
    <xsd:element name="InvalidActivityIdentifierFault"
      type="bes-factory:InvalidActivityIdentifierFaultType"/>
    <xsd:element name="InvalidRequestMessageFault"
      type="bes-factory:InvalidRequestMessageFaultType"/>
  </xsd:schema>
</wsdl:types>

<!-- Messages -->
<wsdl:message name="CreateActivityRequest">
  <wsdl:part name="parameters"
      element="bes-factory:CreateActivity"/>
</wsdl:message>

<wsdl:message name="CreateActivityResponse">
  <wsdl:part name="parameters"
      element="bes-factory:CreateActivityResponse"/>
</wsdl:message>

<wsdl:message name="GetActivityStatusesRequest">
  <wsdl:part name="parameters"
      element="bes-factory:GetActivityStatuses"/>
</wsdl:message>

<wsdl:message name="GetActivityStatusesResponse">
  <wsdl:part name="parameters"
      element="bes-factory:GetActivityStatusesResponse"/>
</wsdl:message>

<wsdl:message name="TerminateActivitiesRequest">
  <wsdl:part name="parameters"
      element="bes-factory:TerminateActivities"/>
</wsdl:message>

<wsdl:message name="TerminateActivitiesResponse">
  <wsdl:part name="parameters"
      element="bes-factory:TerminateActivitiesResponse"/>
</wsdl:message>

<wsdl:message name="GetActivityDocumentsRequest">
  <wsdl:part name="parameters"
      element="bes-factory:GetActivityDocuments"/>
</wsdl:message>

<wsdl:message name="GetActivityDocumentsResponse">
  <wsdl:part name="parameters"
      element="bes-factory:GetActivityDocumentsResponse"/>
</wsdl:message>

<wsdl:message name="GetFactoryAttributesDocumentRequest">
  <wsdl:part name="parameters"
      element="bes-factory:GetFactoryAttributesDocument"/>
</wsdl:message>

<wsdl:message name="GetFactoryAttributesDocumentResponse">
  <wsdl:part name="parameters"
      element="bes-factory:GetFactoryAttributesDocumentResponse"/>
</wsdl:message>

<!-- Fault Messages -->
<wsdl:message name="NotAuthorizedFault">
  <wsdl:part name="Detail"
    element="bes-factory:NotAuthorizedFault"/>
</wsdl:message>

<wsdl:message name="NotAcceptingNewActivitiesFault">
```

```
    <wsdl:part name="Detail"
      element="bes-factory:NotAcceptingNewActivitiesFault"/>
  </wsdl:message>

  <wsdl:message name="UnsupportedFeatureFault">
    <wsdl:part name="Detail"
      element="bes-factory:UnsupportedFeatureFault"/>
  </wsdl:message>

  <wsdl:message name="CantApplyOperationToCurrentStateFault">
    <wsdl:part name="Detail"
      element="bes-factory:CantApplyOperationToCurrentStateFault" />
  </wsdl:message>

  <wsdl:message name="OperationWillBeAppliedEventuallyFault">
    <wsdl:part name="Detail"
      element="bes-factory:OperationWillBeAppliedEventuallyFault" />
  </wsdl:message>

  <wsdl:message name="InvalidActivityIdentifierFault">
    <wsdl:part name="Detail"
      element="bes-factory:InvalidActivityIdentifierFault" />
  </wsdl:message>

  <wsdl:message name="InvalidRequestMessageFault">
    <wsdl:part name="Detail"
      element="bes-factory:InvalidRequestMessageFault" />
  </wsdl:message>

  <!-- Port Type -->
  <wsdl:portType name="BESFactoryPortType">

    <wsdl:operation name="CreateActivity">
      <wsdl:input
         name="CreateActivity"
         message="bes-factory:CreateActivityRequest"
         wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/CreateActivity"/>
      <wsdl:output
         name="CreateActivityResponse"
         message="bes-factory:CreateActivityResponse"
         wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/CreateActivityResponse"/>
      <wsdl:fault name="NotAuthorizedFault"
        message="bes-factory:NotAuthorizedFault"
        wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/Fault"/>
      <wsdl:fault name="NotAcceptingNewActivitiesFault"
         message="bes-factory:NotAcceptingNewActivitiesFault"
         wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/Fault"/>
      <wsdl:fault name="UnsupportedFeatureFault"
         message="bes-factory:UnsupportedFeatureFault"
         wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/Fault"/>
      <wsdl:fault name="InvalidRequestMessageFault"
         message="bes-factory:InvalidRequestMessageFault"
         wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/Fault"/>
    </wsdl:operation>

    <wsdl:operation name="GetActivityStatuses">
      <wsdl:input
          name="GetActivityStatuses"
          message="bes-factory:GetActivityStatusesRequest"
          wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/GetActivityStatuses"/>
      <wsdl:output
          name="GetActivityStatusesResponse"
          message="bes-factory:GetActivityStatusesResponse"
```

```
          wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/GetActivityStatusesResponse"/>
      <wsdl:fault name="InvalidRequestMessageFault"
         message="bes-factory:InvalidRequestMessageFault"
         wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/Fault"/>
    </wsdl:operation>

    <wsdl:operation name="TerminateActivities">
      <wsdl:input
          name="TerminateActivities"
          message="bes-factory:TerminateActivitiesRequest"
          wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/TerminateActivities"/>
      <wsdl:output
          name="TerminateActivitiesResponse"
          message="bes-factory:TerminateActivitiesResponse"
          wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/TerminateActivitiesResponse"/>
      <wsdl:fault name="InvalidRequestMessageFault"
         message="bes-factory:InvalidRequestMessageFault"
         wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/Fault"/>
    </wsdl:operation>

    <wsdl:operation name="GetActivityDocuments">
      <wsdl:input
          name="GetActivityDocuments"
          message="bes-factory:GetActivityDocumentsRequest"
          wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/GetActivityDocuments"/>
      <wsdl:output
          name="GetActivityDocumentsResponse"
          message="bes-factory:GetActivityDocumentsResponse"
          wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/GetActivityDocumentsResponse"/>
      <wsdl:fault name="InvalidRequestMessageFault"
         message="bes-factory:InvalidRequestMessageFault"
         wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/Fault"/>
    </wsdl:operation>

    <wsdl:operation name="GetFactoryAttributesDocument">
      <wsdl:input
          name="GetFactoryAttributesDocument"
          message="bes-factory:GetFactoryAttributesDocumentRequest"
          wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/GetFactoryAttributesDocument"/>
      <wsdl:output
          name="GetFactoryAttributesDocumentResponse"
          message="bes-factory:GetFactoryAttributesDocumentResponse"
          wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/GetFactoryAttributesDocumentResponse"/>
      <wsdl:fault name="InvalidRequestMessageFault"
        message="bes-factory:InvalidRequestMessageFault"
        wsa:Action="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/Fault"/>
    </wsdl:operation>
  </wsdl:portType>

  <!-- Bindings -->
  <wsdl:binding name="BESFactoryBinding" type="bes-factory:BESFactoryPortType">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />

    <wsdl:operation name="CreateActivity">
      <soap:operation soapAction="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/CreateActivity" />
      <wsdl:input name="CreateActivity">
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="CreateActivityResponse">
```

```
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="NotAuthorizedFault">
        <soap:fault name="NotAuthorizedFault" use="literal" />
      </wsdl:fault>
      <wsdl:fault name="NotAcceptingNewActivitiesFault">
        <soap:fault name="NotAcceptingNewActivitiesFault" use="literal" />
      </wsdl:fault>
      <wsdl:fault name="UnsupportedFeatureFault">
        <soap:fault name="UnsupportedFeatureFault" use="literal" />
      </wsdl:fault>
      <wsdl:fault name="InvalidRequestMessageFault">
        <soap:fault name="InvalidRequestMessageFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>

    <wsdl:operation name="GetActivityStatuses">
      <soap:operation soapAction="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/GetActivityStatuses" />
      <wsdl:input name="GetActivityStatuses">
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="GetActivityStatusesResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidRequestMessageFault">
        <soap:fault name="InvalidRequestMessageFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>

    <wsdl:operation name="TerminateActivities">
      <soap:operation soapAction="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/TerminateActivities" />
      <wsdl:input name="TerminateActivities">
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="TerminateActivitiesResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidRequestMessageFault">
        <soap:fault name="InvalidRequestMessageFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>

    <wsdl:operation name="GetActivityDocuments">
      <soap:operation soapAction="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/GetActivityDocuments" />
      <wsdl:input name="GetActivityDocuments">
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="GetActivityDocumentsResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidRequestMessageFault">
        <soap:fault name="InvalidRequestMessageFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>

    <wsdl:operation name="GetFactoryAttributesDocument">
      <soap:operation soapAction="http://schemas.ggf.org/bes/2006/08/bes-
factory/BESFactoryPortType/GetFactoryAttributesDocument" />
      <wsdl:input name="GetFactoryAttributesDocument">
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output name="GetFactoryAttributesDocumentResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="InvalidRequestMessageFault">
        <soap:fault name="InvalidRequestMessageFault" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
```

```
  </wsdl:binding>
</wsdl:definitions>
```

Appendix F. Normative BES-Activity WSDL

None needed.

Appendix G. Non-normative Examples for BES-Management

a. StopAcceptingNewActivities

i. Request Message

```
<s11:Envelope
    xmlns:s11="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:bes-mgmt="http://schemas.ggf.org/bes/2006/08/bes-management">

    <s11:Header>
        <wsa:Action>
            http://schemas.ggf.org/bes/2006/08/bes-
           management/BESManagementPortType/StopAcceptingNewActivities
        </wsa:Action>
        <wsa:To s11:mustUnderstand=1>
            http://www.bes.org/BESManager
        </wsa:To>
    </s11:Header>

    <s11:Body>
        <bes-mgmt:StopAcceptingNewActivities/>
    </s11:Body>
</s11:Envelope>
```

ii. Response Message

```
<s11:Envelope
    xmlns:s11="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:bes-mgmt="http://schemas.ggf.org/bes/2006/08/bes-management">

    <s11:Header>
        <wsa:Action>
            http://schemas.ggf.org/bes/2006/08/bes-
           management/BESManagementPortTypeStopAcceptingNewActivitiesResponse
        </wsa:Action>
        <wsa:To>
            http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous
        </wsa:To>
    </s11:Header>

    <s11:Body>
        <bes-mgmt:StopAcceptingNewActivitiesResponse/>
    </s11:Body>
</s11:Envelope>
```

b. StartAcceptingNewActivities

i. Request Message

```
<s11:Envelope
    xmlns:s11="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:bes-mgmt="http://schemas.ggf.org/bes/2006/08/bes-management">

    <s11:Header>
        <wsa:Action>
            http://schemas.ggf.org/bes/2006/08/bes-
           management/BESManagementPortType/StartAcceptingNewActivities
        </wsa:Action>
        <wsa:To s11:mustUnderstand=1>
            http://www.bes.org/BESManager
        </wsa:To>
    </s11:Header>

    <s11:Body>
```

```
        <bes-mgmt:StartAcceptingNewActivities/>
    </s11:Body>
</s11:Envelope>
```

ii. Response Message

```
<s11:Envelope
    xmlns:s11="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:bes-mgmt="http://schemas.ggf.org/bes/2006/08/bes-management">

    <s11:Header>
        <wsa:Action>
            http://schemas.ggf.org/bes/2006/08/bes-
           management/BESManagementPortTypeStartAcceptingNewActivitiesResponse
        </wsa:Action>
        <wsa:To>
            http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous
        </wsa:To>
    </s11:Header>

    <s11:Body>
        <bes-mgmt:StartAcceptingNewActivitiesResponse/>
    </s11:Body>
</s11:Envelope>
```

Appendix H. Non-normative Examples for BES-Factory

c. CreateActivity

i. Request Message

```
<s11:Envelope
    xmlns:s11="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:jsdl="http://schemas.ggf.org/jsdl/2005/11/jsdl"
    xmlns:bes-factory="http://schemas.ggf.org/bes/2006/08/bes-factory">

    <s11:Header>
        <wsa:Action>
            http://schemas.ggf.org/bes/2006/08/bes-
           factory/BESFactoryPortType/CreateActivity
        </wsa:Action>
        <wsa:To s11:mustUnderstand=1>
            http://www.bes.org/BESFactory
        </wsa:To>
    </s11:Header>

    <s11:Body>
        <bes-factory:CreateActivity>
            <bes-factory:ActivityDocument>
                <jsdl:JobDefinition>
                    {Any valid JSDL document}
                </jsdl:JobDefinition>
            </bes-factory:ActivityDocument>
        </bes-factory:CreateActivity>
    </s11:Body>
</s11:Envelope>
```

ii. Response Message

```
<s11:Envelope
    xmlns:s11="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:n00="http://tempuri.org/"
    xmlns:naming="http://schemas.ggf.org/naming/2006/08/naming"
    xmlns:bes-factory="http://schemas.ggf.org/bes/2006/08/bes-factory">

    <s11:Header>
        <wsa:Action>
            http://schemas.ggf.org/bes/2006/08/bes-
           factory/BESFactoryPortTypeCreateActivityResponse
        </wsa:Action>
        <wsa:To>
            http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous
        </wsa:To>
    </s11:Header>

    <s11:Body>
        <bes-factory:CreateActivityResponse>
            <bes-factory:ActivityIdentifier>
                <wsa:Address>http://tempuri.org/some-service</wsa:Address>
                <wsa:ReferenceParameters>
                    <n00:id>D4A88953-FFFF-49F6-5145-AE21FF0438AE</n00:id>
                </wsa:ReferenceParameters>
                <wsa:Metadata>
                    <naming:EndpointIdentifier>urn:guid:B94C4186-FFFF-4dbb-AD9C-
           39DFB8B54388</name:EndpointIdentifier>
                </wsa:Metadata>
            </bes-factory:ActivityIdentifier>
        </bes-factory:CreateActivityResponse>
    </s11:Body>
</s11:Envelope>
```

d. GetActivityStatuses

i. Request Message

```
<s11:Envelope
    xmlns:s11="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:n00="http://tempuri.org/"
    xmlns:naming="http://schemas.ggf.org/naming/2006/08/naming"
    xmlns:bes-factory="http://schemas.ggf.org/bes/2006/08/bes-factory">

    <s11:Header>
        <wsa:Action>
            http://schemas.ggf.org/bes/2006/08/bes-
           factory/BESFactoryPortType/GetActivityStatuses
        </wsa:Action>
        <wsa:To s11:mustUnderstand=1>
            http://www.bes.org/BESFactory
        </wsa:To>
    </s11:Header>

    <s11:Body>
        <bes-factory:GetActivityStatuses>
            <bes-factory:ActivityIdentifier>
                <wsa:Address>http://tempuri.org/some-service</wsa:Address>
                <wsa:ReferenceParameters>
                    <n00:id>D4A88953-FFFF-49F6-5145-AE21FF0438AE</n00:id>
                </wsa:ReferenceParameters>
                <wsa:Metadata>
                    <naming:EndpointIdentifier>urn:guid:B94C4186-FFFF-4dbb-AD9C-
           39DFB8B54388</name:EndpointIdentifier>
                </wsa:Metadata>
            </bes-factory:ActivityIdentifier>
        </bes-factory:GetActivityStatuses>
    </s11:Body>
</s11:Envelope>
```

ii. Response Message

```
<s11:Envelope
    xmlns:s11="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:n00="http://tempuri.org/"
    xmlns:naming="http://schemas.ggf.org/naming/2006/08/naming"
    xmlns:bes-factory="http://schemas.ggf.org/bes/2006/08/bes-factory">

    <s11:Header>
        <wsa:Action>
            http://schemas.ggf.org/bes/2006/08/bes-factory/GetActivityStatusesResponse
        </wsa:Action>
        <wsa:To>
            http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous
        </wsa:To>
    </s11:Header>

    <s11:Body>
        <bes-factory:GetActivityStatusesResponse>
            <bes-factory:Response>
                <bes-factory:ActivityIdentifier>
                    <wsa:Address>http://tempuri.org/some-service</wsa:Address>
                    <wsa:ReferenceParameters>
                         <n00:id>D4A88953-FFFF-49F6-5145-AE21FF0438AE</n00:id>
                    </wsa:ReferenceParameters>
                    <wsa:Metadata>
                        <naming:EndpointIdentifier>urn:guid:B94C4186-FFFF-4dbb-AD9C-
           39DFB8B54388</name:EndpointIdentifier>
                    </wsa:Metadata>
                </bes-factory:ActivityIdentifier>
                <bes-factory:ActivityStatus>
                    <bes-factory:State>Running</bes-factory:State>
                    <n00:Staging-In/>
                </bes-factory:ActivityStatus>
```

```
            </bes-factory:Response>
        </bes-factory:GetActivityStatusesResponse>
    </s11:Body>
</s11:Envelope>
```

e. TerminateActivities

i. Request Message

```
<s11:Envelope
    xmlns:s11="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:n00="http://tempuri.org/"
    xmlns:naming="http://schemas.ggf.org/naming/2006/08/naming"
    xmlns:bes-factory="http://schemas.ggf.org/bes/2006/08/bes-factory">

    <s11:Header>
        <wsa:Action>
            http://schemas.ggf.org/bes/2006/08/bes-factory/TerminateActivities
        </wsa:Action>
        <wsa:To s11:mustUnderstand=1>
            http://www.bes.org/BESFactory
        </wsa:To>
    </s11:Header>

    <s11:Body>
        <bes-factory:TerminateActivities>
            <bes-factory:ActivityIdentifier>
                <wsa:Address>http://tempuri.org/some-service</wsa:Address>
                <wsa:ReferenceParameters>
                    <n00:id>D4A88953-FFFF-49F6-5145-AE21FF0438AE</n00:id>
                </wsa:ReferenceParameters>
                <wsa:Metadata>
                    <naming:EndpointIdentifier>urn:guid:B94C4186-FFFF-4dbb-AD9C-
           39DFB8B54388</name:EndpointIdentifier>
                </wsa:Metadata>
            </bes-factory:ActivityIdentifier>
        </bes-factory:TerminateActivities>
    </s11:Body>
</s11:Envelope>
```

ii. Response Message

```
<s11:Envelope
    xmlns:s11="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:n00="http://tempuri.org/"
    xmlns:naming="http://schemas.ggf.org/naming/2006/08/naming"
    xmlns:bes-factory="http://schemas.ggf.org/bes/2006/08/bes-factory">

    <s11:Header>
        <wsa:Action>
            http://schemas.ggf.org/bes/2006/08/bes-factory/TerminateActivitiesResponse
        </wsa:Action>
        <wsa:To>
            http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous
        </wsa:To>
    </s11:Header>

    <s11:Body>
        <bes-factory:TerminateActivitiesResponse>
            <bes-factory:Response>
                <bes-factory:ActivityIdentifier>
                    <wsa:Address>http://tempuri.org/some-service</wsa:Address>
                    <wsa:ReferenceParameters>
                        <n00:id>D4A88953-FFFF-49F6-5145-AE21FF0438AE</n00:id>
                    </wsa:ReferenceParameters>
                    <wsa:Metadata>
                        <naming:EndpointIdentifier>urn:guid:B94C4186-FFFF-4dbb-AD9C-
           39DFB8B54388</name:EndpointIdentifier>
                    </wsa:Metadata>
```

```
                </bes-factory:ActivityIdentifier>
                <bes-factory:Cancelled>true</bes-factory:Cancelled>
            </bes-factory:Response>
        </bes-factory:TerminateActivitiesResponse>
    </s11:Body>
</s11:Envelope>
```

f. GetActivityDocuments

i. Request Message

```
<s11:Envelope
    xmlns:s11="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:n00="http://tempuri.org/"
    xmlns:naming="http://schemas.ggf.org/naming/2006/08/naming"
    xmlns:bes-factory="http://schemas.ggf.org/bes/2006/08/bes-factory">

    <s11:Header>
        <wsa:Action>
            http://schemas.ggf.org/bes/2006/08/bes-factory/GetActivityDocuments
        </wsa:Action>
        <wsa:To s11:mustUnderstand=1>
            http://www.bes.org/BESFactory
        </wsa:To>
    </s11:Header>

    <s11:Body>
        <bes-factory:GetActivityDocuments>
            <bes-factory:ActivityIdentifier>
                <wsa:Address>http://tempuri.org/some-service</wsa:Address>
                <wsa:ReferenceParameters>
                    <n00:id>D4A88953-FFFF-49F6-5145-AE21FF0438AE</n00:id>
                </wsa:ReferenceParameters>
                <wsa:Metadata>
                    <naming:EndpointIdentifier>urn:guid:B94C4186-FFFF-4dbb-AD9C-
           39DFB8B54388</name:EndpointIdentifier>
                </wsa:Metadata>
            </bes-factory:ActivityIdentifier>
        </bes-factory:GetActivityDocuments>
    </s11:Body>
</s11:Envelope>
```

ii. Response Message

```
<s11:Envelope
    xmlns:s11="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:n00="http://tempuri.org/"
    xmlns:naming="http://schemas.ggf.org/naming/2006/08/naming"
    xmlns:bes-factory="http://schemas.ggf.org/bes/2006/08/bes-factory">

    <s11:Header>
        <wsa:Action>
            http://schemas.ggf.org/bes/2006/08/bes-factory/GetActivityDocumentsResponse
        </wsa:Action>
        <wsa:To>
            http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous
        </wsa:To>
    </s11:Header>

    <s11:Body>
        <bes-factory:GetActivityDocumentsResponse>
            <bes-factory:Response>
                <bes-factory:ActivityIdentifier>
                    <wsa:Address>http://tempuri.org/some-service</wsa:Address>
                    <wsa:ReferenceParameters>
                        <n00:id>D4A88953-FFFF-49F6-5145-AE21FF0438AE</n00:id>
                    </wsa:ReferenceParameters>
                    <wsa:Metadata>
```

```
                        <naming:EndpointIdentifier>urn:guid:B94C4186-FFFF-4dbb-AD9C-
          39DFB8B54388</name:EndpointIdentifier>
                    </wsa:Metadata>
                </bes-factory:ActivityIdentifier>
                <jsdl:JobDefinition> { any valid JSDL document} </jsdl:JobDefinition>
            </bes-factory:Response>
        </bes-factory:GetActivityDocumentsResponse>
    </s11:Body>
</s11:Envelope>
```

g. GetFactoryAttributesDocument

i. Request Message

```
<s11:Envelope
    xmlns:s11="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:bes-mgmt="http://schemas.ggf.org/bes/2006/08/bes-factory">

    <s11:Header>
        <wsa:Action>
            http://schemas.ggf.org/bes/2006/08/bes-
           factory/BESFactoryPortType/GetFactoryAttributesDocument
        </wsa:Action>
        <wsa:To s11:mustUnderstand=1>
            http://www.bes.org/BESManager
        </wsa:To>
    </s11:Header>

    <s11:Body>
        <bes-factory:GetFactoryAttributesDocument/>
    </s11:Body>
</s11:Envelope>
```

ii. Response Message

```
<s11:Envelope
    xmlns:s11="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:jsdl="http://schemas.ggf.org/jsdl/2005/11/jsdl"
    xmlns:n00="http://tempuri.org/"
    xmlns:naming="http://schemas.ggf.org/naming/2006/08/naming"
    xmlns:bes-mgmt="http://schemas.ggf.org/bes/2006/08/bes-factory">

    <s11:Header>
        <wsa:Action>
            http://schemas.ggf.org/bes/2006/08/bes-
           factory/GetFactoryAttributesDocumentResponse
        </wsa:Action>
        <wsa:To>
            http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous
        </wsa:To>
    </s11:Header>

    <s11:Body>
        <bes-factory:GetFactoryAttributesDocumentResponse>
            <bes-factory:FactoryResourceAttributesDocument>
                <bes-factory:BasicResourceAttributesDocument>
                    <bes-factory:OperatingSystem>
                        <jsdl:OperatingSystemType>
                            <jsdl:OperatingSystemName>Inferno</jsdl:OperatingSystemName>
                        </jsdl:OperatingSystemType>
                        <jsdl:OperatingSystemVersion>
                            5.01
                        </jsdl:OperatingSystemVersion>
                    </bes-factory:OperatingSystem>
                    <bes-factory:CPUArchitecture>
                        <jsdl:CPUArchitecureName>sparc</jsdl:CPUArchitecureName>
                    </bes-factory:CPUArchitecture>
                    <bes-factory:CPUCount>4.0</bes-factory:CPUCount>
                    <bes-factory:CPUSpeed>2400000000.0</bes-factory:CPUSpeed>
```

```
                    <bes-factory:PhysicalMemory>2048000000.0</bes-factory:PhysicalMemory>
                    <bes-factory:VirtualMemory>20480000000.0</bes-factory:VirtualMemory>
                </bes-factory:BasicResourceAttributesDocument>
                <bes-factory:IsAcceptingNewActivities>
                    true
                </bes-factory:IsAcceptingNewActivities>
                <bes-factory:CommonName>mooch.home.net</bes-factory:CommonName>
                <bes-factory:LongDescription>
                    Mark&amp;apos;s Personal Desktop Machine
                </bes-factory:LongDescription>
                <bes-factory:TotalNumberOfActivities>
                    2
                </bes-factory:TotalNumberOfActivities>
<bes-factory:ActivityReference>
                    <wsa:Address>http://tempuri.org/some-service</wsa:Address>
                    <wsa:ReferenceParameters>
                        <n00:id>D4A88953-FFFF-49F6-5145-AE21FF0438AE</n00:id>
                    </wsa:ReferenceParameters>
                    <wsa:Metadata>
                        <naming:EndpointIdentifier>urn:guid:B94C4186-FFFF-4dbb-AD9C-
           39DFB8B54388</name:EndpointIdentifier>
                    </wsa:Metadata>
                </bes-factory:ActivityReference>
                <bes-factory:ActivityReference>
                    <wsa:Address>http://tempuri.org/some-service</wsa:Address>
                    <wsa:ReferenceParameters>
                        <n00:id>D4A88953-0000-49F6-5145-AE21FF0438AE</n00:id>
                    </wsa:ReferenceParameters>
                    <wsa:Metadata>
                        <naming:EndpointIdentifier>urn:guid:B94C4186-0000-4dbb-AD9C-
           39DFB8B54388</name:EndpointIdentifier>
                    </wsa:Metadata>
                </bes-factory:ActivityReference>
                <bes-factory:TotalNumberOfContainedResources>
                    0
                </bes-factory:TotalNumberOfContainedResources>
                <bes-factory:NamingProfile>
                    http://schemas.ggf.org/bes/2006/08/bes/naming/WS-Naming
                </bes-factory:NamingProfile>
                <bes-factory:LocalResourceManagerType>
                    http://schemas.ggf.org/bes/2006/08/bes-factory/native
                </bes-factory:LocalResourceManagerType>
            </bes-factory:FactoryResourceAttributesDocument>
        </bes-factory:GetFactoryAttributesDocumentResponse>
    </s11:Body>
</s11:Envelope>
```

Appendix I. OGSA WSRF Basic Profile 1.0 Rendering

We explained in the introduction how the BES specification is designed to enable the creation of BESs that are compliant with OGSA Base Profiles via the composition of BES port-types with other appropriate port-types. To illustrate how this can be done, we explain how an implementor would create a BES that is compliant with the OGSA WSRF Base Profile 1.0. This material is not intended to be normative.

1) A BES that is to be compliant with the OGSA WSRF Base Profile must implement, in addition to the BES-Management and BES-Factory port-types, the WS-ResourceProperties, WS-ResourceLifetime, and WS-BaseNotification port-types, in a manner compliant with the OGSA WSRF Base Profile.

2) Implementers MUST also ensure that all attributes given in the description of the attributes document appear as exactly named WS-ResourceProperties. Exactly named here implies that each resource property must have the QName {http://schemas.ggf.org/bes/2006/08/bes-management}attr-name where attr-name is the name of the attribute used inside the attributes document types (e.g., TotalNumberOfActivities, CommonName, etc.).