Editor:
Keisuke Fukui, Fujitsu

Application Contents Service Specification 1.0
Revision 1

Status of This Document

This document provides information to the community regarding the Application Contents Service Specification. Distribution is unlimited.

Abstract

In order to install and operate complex systems such as three-tier systems more efficiently and automatically, it is essential to transfer and manage as a unit of different application-related information, which we call Application Contents. Application Contents may include deployable contents such as program binaries, configuration data, and descriptions of the hardware resources typically required for application execution as well as other contents such as descriptions of procedures for lifecycle management and management policies applied to the running system.

In this document, we define Application Contents Service (ACS) to manage Application Contents. ACS is an OGSA service, which maintains Application Contents as an Application Archive. The ACS repository provides functions to retrieve the contents and their change histories. We also define a standard format of an Application Archive for its management and exchanging.

Editor:
Keisuke Fukui, Fujitsu

Contents

1. Introduction

OGSA-compliant Grid systems provide a rich set of functionalities such as dynamic resource allocation, provisioning, service level management, and automation of administrative tasks, as shown in OGSA [OGSA doc]. In order to utilize these functionalities in executing applications on Grid systems, applications require certain preparation. That is, developers of Grid applications need develop various components including, but not limited to, deployment procedure scripts, requirement descriptions on resources, and policy rules, which are all handled by OGSA services, as well as program binaries, initial data, and configuration information, which are commonly necessary for application execution.

In this document, we use the term a Grid Application, or simply an application, to refer an application executable on Grid systems. And we use the term Application Contents for a set of information which collectively makes the application deployable and runnable on Grid systems, then optionally helps system do provisioning autonomously. The Application Contents don't include information updated by a job instance and information describing a status of a job instance. In any way, ACS doesn't interpret or execute information in the contents; rather it just manages them for use by other OGSA services.

An Application Archive may include a stack of the software, for example, middleware and/or operating system binaries, in order to build the hosting environment required to run a job. Alternatively they may be provided by the system, based on the resource requirement description specified by Application Contents in the Archive. The system may provide with the hosting environment, out of the internal pool of the various hosting environments or creating it on demand.

The necessity for the ACS concept lies within its potential for contributing to practical configuration/repository management services, a technology aid for automated provisioning, jobs managers, etc, as well as to ease the submission and the following management of the application related files as described above. Exchangeability and interoperability of Application Contents is important so that the complexity of efforts do not impede grid adoption throughout industry.

More complicated application, such as three-tier system, makes it difficult to describe information handled by Grid systems for the purpose of deployment and execution management, and requires many files to be handled. Consistent management of Application Contents throughout the lifetime of application, including its version-upgrade, is difficult and prone to human errors.

Scenarios for resource sharing and disaster recovery across sites may be realized through the distributing a set of jobs among the separate grid systems if they are so designed as to be runnable in the distributed environment. For those jobs, automated exchange of Application Contents between Grid systems will facilitate effective deployment of the job. There already exist many products for automated deployment of application on multiple hosts and change management on them. However, those products don't target to run in dynamic and heterogeneous environments like the Grid environments, or to realize resource sharing across multiple administrative domains, which is one of the goals of OGSA. Some products, such as J2EE, utilize mechanisms for specific platforms, and some products take proprietary and/or nonpublic approaches.

Application Contents Service (ACS), defined by this document, will serve to realize above scenarios more effectively. ACS is an OGSA service, which maintains Application Contents for a unit of task processed in Grid systems as an Application Archive. The ACS repository provides functions to retrieve the contents and/or their change histories. We also define a standard format of an Application Archive for its management and exchanging.

By storing a deployable logical set of Application Contents in a single archive, ACS simplifies creation and updating processes of them and enables integrated management to maintain their

consistency. The standard description of Application Contents archive also enables automated administration of Grid Application. By providing change history information and restoration capability of Application Contents, ACS can provide with assistance for troubleshooting of problems arising from erroneous operations performed on the existing Archives.

In this document, we describe:

- In Chapter 2, analysis and definition of the set of requirements for ACS.
- In Chapter 3, concept and architecture of ACS based on the requirements, and relationship to the other standard activities.
- In Chapter 4, some example usage scenarios to present the basic idea.
- In Chapter 5, normative specification of Application Repository Interface (ARI).
- In Chapter 6, normative specification of Application Archive Format (AAF).
- In Chapter 7, security consideration.
- In Chapter 8, non-normative samples of ACS request messages and Application Archive Descriptor.
- In Chapter 9, normative schemas of messages and interfaces (xsd and wsdl).
- In Chapter 10, open issues toward the future enhancement to this specification.

This specification is designed utilizing the experience in the Business Grid Project [BizGrid] in Japan, which makes reference with OGSA architecture. Most of the terms in this document are included in OGSA documents [OGSA doc][OGSA glossary][OGSA usecase]. See glossary section at the end of this document for terms specific to this document.

1.1 Goals

This document is intended to:

- define the standard way to manage and handle Application Contents as a deployable logical unit so as to maintain their consistency, reduce management overheads, and enable automation throughout the lifetime of the application
- define the standard format of Application Archive so as to simplify management processes and to automate administration of application

1.2 Out of scope

ACS will define the specification necessary for management of Application Archive, but doesn't interpret or execute Application Contents contained in them. That is, ACS handles Application Contents as opaque data except for Application Archive Descriptors. Information necessary for deployment and execution management of application need to be defined by OGSA services that process the information.

1.3 Notational Conventions

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

When describing abstract data models, this specification uses the notational convention used by the [XML Infoset]. Specifically, abstract property names always appear in square brackets (e.g., [some property]).

When describing concrete XML schemas, this specification uses the notational convention of [WS-Security]. Specifically, each member of an element's [children] or [attributes] property is described using an XPath-like notation (e.g., /x:MyHeader/x:SomeProperty/@value1). The use of {any} indicates the presence of an element wildcard (<xsd:any/>). The use of @{any} indicates the presence of an attribute wildcard (<xsd:anyAttribute/>). Characters are appended to attributes, elements, and {any} to indicate the number of times they may occur as follows: ? (0 or 1), * (0 or more), + (1 or more). No character indicates exactly 1 occurrence. The characters [and] are used to indicate that contained items are to be treated as a group with respect to the ?, *, and + characters. Attributes, elements, and values separated by | and grouped with (and) are meant to be syntactic alternatives.

1.4 Namespaces

The following namespaces are used in this document:

Prefix	Namespace
xsd	http://www.w3.org/2001/XMLSchema
wsa	http://www.w3.org/2005/03/addressing
wsrf-bf	http://docs.oasis-open.org/wsrf/bf-1
wsrf-rp	http://docs.oasis-open.org/wsrf/rp-1
wsrf-r	http://docs.oasis-open.org/wsrf/r-1
ref	http://ws-i.org/profiles/basic/1.1/xsd
ds	http://www.w3.org/2000/09/xmldsig#"
ari	http://schemas.ggf.org/acs/2005/10/ari
aaf	http://schemas.ggf.org/acs/2005/10/aaf

2. Requirements

In this chapter, we define a set of requirements that ACS is intended to address. The analysis is based on the demand of manipulating Application Contents inside/outside of the Gird system.

2.1 Functional Requirements

2.1.1 Everything required for a task in a single archive

In order to install and operate complex systems, a number of various application related information must be specified. Keeping everything required for a task into a single archive will contribute to ease of use, reduction of the administrative cost and higher consistency. Examples of such information are the following:

- Program binaries and initial data: These are the main contents of the systems and/or the Grid tasks.
- Middleware and/or OS: These may be required for building execution environments in a data center required for Grid task execution.
- Policy rules: OGSA services for task execution management use this information to manage Grid tasks dynamically in case of, for example, erroneous conditions and/or increasing load.
- Fail-over and/or load-sharing configuration: It is an important Grid requirement to be able to cope with disaster recovery and load-sharing among geographically separated sites.
- Scheduling information or operational policy rules, etc.

Merits:

- (Ease of use) Business activity manager of a complex system can handle application contents easily, resulting in less human errors.
- (Administrative cost reduction) Administrative costs will be reduced by simplifying the handling procedures of application contents.
- (Consistency) Consistency of application contents can be validated more easily if they are in a single archive.

Requirements:

- Application contents must be logically bundled in a single and consistent archive.
- ACS should allow diversity of information as its contents, including application, middleware, OS, or firmware.
- ACS should allow bundling a set of applications which are to be deployed and executed in geographically and/or administratively separated sites.

2.1.2 Everything required for Grid in a single archive

Grid application ready to be executed comprises various files such as the descriptive information to be used by the various services for scheduling and resource management. To operate the system more efficiently and automatically, those files need to be presented in a standard format. Examples of such information are the following:

- Configuration Description, Deployment and Lifecycle Management information, presented as CDL.
- Resource requirement description for hardware and/or software, presented as JSDL or WS-Agreement.

Merits:

- (Completeness of the contents) Itemizing and collecting the requisite information in an archive contribute to clarify the completeness of the required information both for user and system.

Requirements:

- All information required for Grid task execution must be logically bundled in a archive.

2.1.3 A service comprising a part of systems

The ACS will be a service comprising a part of systems, such as a Grid system. An archive repository can facilitates a stable source of the archive to collaborate with other services in the system. Complex system composed from multiple applications is at risk for troubles due to missing and/or inconsistency of application contents. It will execute a consistency check, especially at the time of modification of the application.

Changes, upgrades and bug-fixes of the application are envisioned during the lifecycle of Application Archive. The archive repository facilitates lifecycle management of Application Contents including version control, distributed deployment such as copying and relaying a software stack onto multiple machines when sharing application archives across administrative domain.

Merits:

- (Service Integration) Storing the contents of the Application Contents in a repository and providing to other services contributes to the system efficiency in executing a task.
- (Lifecycle management support) Version management, if combined with repository management, can reduce the cost of lifecycle management of application.

Requirements:

- Archives must be stored in a repository, with a standard set of the CRUD functions, i.e. Create, Read, Update, and Delete. Once stored, operations must be allowed in a smaller unit in the archive.
- Repository interface must provide reference to archives, with appropriate access control management for actors in both inside and outside of the system.
- Incremental upgrading of the application archive must be supported.
- Consistency check of the archive contents should be assisted.

2.1.4 Reuse of an existing archive

Considering situations such as repeating the same calculation with different parameters or performing similar services, in parallel or in different points of time, a business activity manager may feel convenient if the system allows creating a new task reusing the content for a running task or its subset. ACS should enable to create a new task using an existing archive in the repository that is previously stored for a similar task. A business activity manager may want to apply some updates on the archive to modify a task or to create a different sort of task. A business activity manager may also want to use the archive created by another business activity manager besides improving his/her own archive.

Merits:

- (Cost reduction) Reuse of an archive can reduce the development/administrative cost when creating similar or repetitive tasks.
- (Incremental improvement) Reuse of an archive enables incremental improvement of functionality and/or reliability.
- (Ease of use) Reuse of an archive contributes to the over all ease of use for the systems

Requirements:

- Retrieving an archive by multiple different actors, under appropriate access control to the archive, must be allowed.

2.2 Non-functional Requirements

2.2.1 Exchangeability, Interoperability

An application archive is expected to be used for various administrative domains and/or heterogeneous environments. In order to address to diversity of the environments, standard specification needs to define a reliable universal format and/or interface which ensure reusability and exchangeability.

Requirements:

- Standard format of archive and interface to the repository needs to be defined.

2.2.2 Extensibility

Specification should allow incremental evolution of itself, to be adaptable for future technologies that are under development or not predicted today, including those in GGF. In addition, specification is desired to allow broader range of the industry to share its single framework. Application archives should accommodate various use cases, such as application installation on a single machine, copying a software stack onto multiple machines, and task execution on a Grid system. It should allow for diversity of system implementations to handle the application archives.

Requirements:

- In addition to minimum set of requirements, ACS must allow optional or extension sets.

2.2.3 Making use of the external ingenuity

There are and will be ingenuity in the world to implement the feature, while satisfying the common set of the requirements and interface specifications. The specification should make full advantage of the external ingenuity in the field.

Requirements:

- ACS specification will not define how to implement; it should provide common interfaces and mechanisms for implementing.

2.2.4 Efficiency

Complex application comes to be a larger set of files and data. In order to address to this, the references to the external storage that is persistent and stable should be allowed in the both archive

and repository. Efficiency in data storage and efficiency in transport is also expected. Transport on network is expected in creation and retrieval of an archive file. ACS should seek to method to take advantage of the state of the art of various technologies in both storage and transport of archives.

Requirements:

- To reduce the size of archives, reference to external storage/files should be allowed in them.
- ACS must provide mechanisms for minimizing the network traffic by allowing differential application archives.
- When managing contents of archives in the repository, mechanisms for eliminating redundancy and minimizing the disk usage should be allowed.
- To make use of the efficient transport technologies available, the third party transports must be allowed in the repository interface.
- ACS should allow ARI implementers to select a communication protocol for transport.

2.3 Security

Business/commercial applications may include confidential and/or critical information, and integrity of contents is vital.

Requirements:

- Security features must be deployable according to the business requirements.
- Appropriate access control to archives is required.
- Secure transport of archives must be realized. For example, mechanisms for keeping confidentiality of archive contents and detection of alteration and spoofing are required.

The security design of ACS follows that of OGSA-WG Security Design Team.

2.4 OGSA-compliant

It is important that the ACS is an OGSA service since the OGSA will be a harness framework for the future grid systems and ensure the interoperability of the diversity of the Grid system implementations.

Requirements:

- The service must be implemented as an OGSA service based on OGSA infrastructure.
- ACS must cooperate with other OGSA services and must not overlap in capabilities with them.
- Archive components must be retrievable by other OGSA service entity.
- Archive contents can be any opaque entities that are required for deployment and execution of application.

3. Architecture

3.1 Concepts

3.1.1 Application Archive and Repository

ACS provides a repository for grid applications. Application providers create Application Archive (AA) instances accompanied with the meta-information describing the contents. The AA is a logical bundle of files that is used for provisioning and executing a task in a Grid system. Such information may include, but not limited to, the application executable code, configuration data necessary for initial deployment of the application and the deployment descriptor documents. The internal form of AA inside the ACS repository is called an AA instance and the wire format of AA for creating or downloading an AA instance is called AA document.

While one may imagine any granularity of the application, a single AA might only contain a single unit of a task deployed in a Grid system. This would be a simple and useful example for a Commercial Data Center use case [OGSA usecase]. A basic scenario for ACS would include an application provider using ACS to create an AA instance in the ACS repository, thereby making the application available for system services that implement deployment or execution in a Grid system. ACS defines a Web Service interface called Application Repository Interface (ARI) to manage the AA instance for both application producers and consumers which supports operations to create, read, update and delete. Also, through ARI, the meta-information of AAs, such as change histories can be handled. Users of ARI can retrieve parts of an AA instance in a granularity of an Application Content. They can make subscription to the event notifications so that they are notified when the AA is updated or deleted. The detailed interface is explained later in Chapter 4.1.

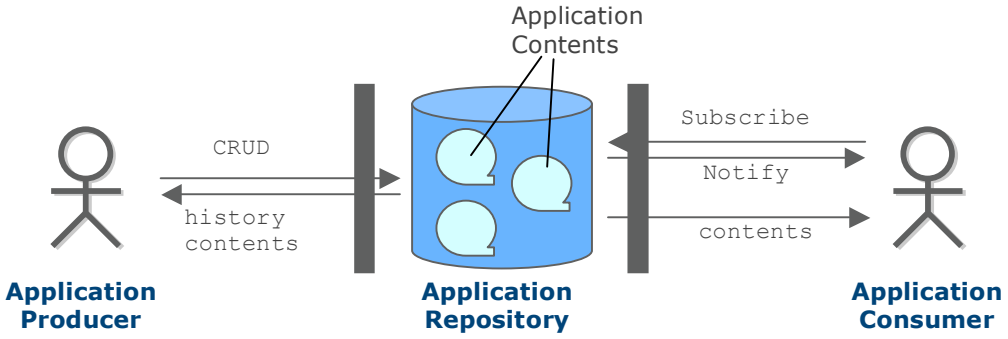

Figure 3-1 ACS concepts diagram

3.1.2 AA Structure

An AA consists of an Application Archive Descriptor (AAD) and zero or more Application Contents. AAD is an XML document, which describes properties of the AA. ACS repository will parse and make use of the AAD. AAD may contain information such as identification, access constraint policy, and information associated with the structure of the AA contents. In order to serve to `GetContents` operation, ACS repository needs to know the meta-information of the Application Content files, such as the names and types of the files. They should be provided by creator and included in AAD.

Application Contents are other items contained in an AA, which may be anything to be used for provisioning and executing applications. A typical Application Content would likely be a Deployment Descriptor (DD). ACS does not parse or interpret Application Contents, including the DD, rather it handles them as the opaque payloads to the AA.

In order to ensure its integrity, AA may include a signature.

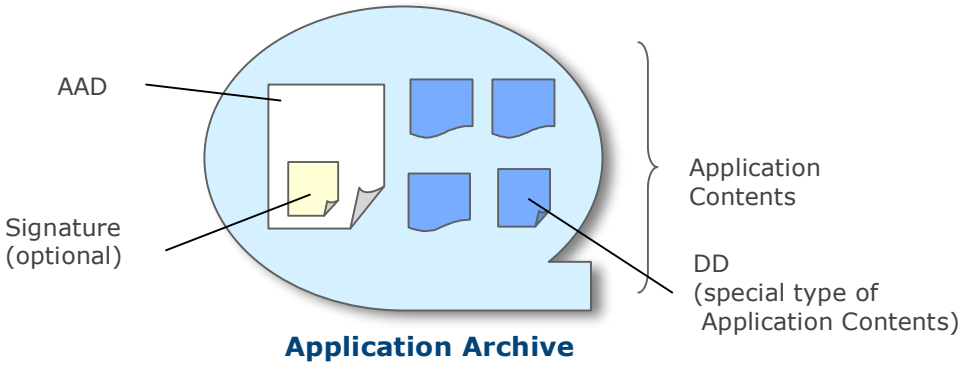

Figure 3-2 AA structure

3.1.3 Physical Format of the AA

It will be useful to bundle contents in an AA into a single file, making distributing the Grid application outside of the repository easier. Also, for those that are larger in size and in binary in nature, such as application programs or data, it may also be useful do compression. One of the purposes of the specification is to create a few sets of example formats in order to increase the chance of interoperability among the implementations. As such, we define a standardized external format of an AA which can make use of the off the shelf prevailed technologies for bundling and compression such as zip. The `Create` operation can accept either a bundled file or discrete files, both containing equivalent AA information. See Section 6.3 for physical format of AA and Section 5.1.2 for `Create` operation.

Note that defining physical format of AA does not mean that ACS specifies the internal format of the AA instance in the Application Repository. How to store the AA in the repository is out of ACS scope so that the implementers can contrive ways to be efficient in its operation, for example, to save disk space in the repository.

3.1.4 Relationship between AA and Job

An AA can be used to create a job, i.e. a unit of task, in the Grid system. But maintaining the relationship between an AA and a job is out of ACS's scope. AA management is independent of execution management of jobs, however, ACS may serve to deliver contents to an execution management system. The user of a grid system can create new jobs, reusing an individual AA to support many jobs. Thus, multiple jobs may refer to a same AA instance, but the management related to the job instance is up to the execution management system, which is a consumer of the ACS.

In another case, the user can make a copy of an AA and create a new AA document or AA instance or extend it by modifying or overwriting some of the contents. ACS doesn't parse the DD, therefore the ACS is neutral to any language describing DD.

3.1.5 AA Identifier and Endpoint Reference

An AAID is an element in the AAD that uniquely identifies an AA. Two instances of AA with the same AAID are equivalent in contents. Different versions for the same application will have different AAIDs. Two different AA EPRs does not necessarily mean that the two AA instances they refer to are different. To access an AA instance, one must know the endpoint reference of the AA (AAEPR). The AAEPR can be looked up using the key provided by the ACS implementation, such as AAID.

3.1.6 Version Control

A new AA instance can be created based on the existing AA instance, producing another new AA instance. The operation to accomplish this is called `Update`. It is performed on an existing AA instance, and replaces, removes, or adds to the original in a granularity level starting at the Application Content. The AAD will be always replaced with the execution of the `Update` operation since the version information, a part of its AAID element, must have a different value and reflect the modification. The implementation of the ACS repository must verify that the two AA instances do not have the exact same combination of the version and name in AAID discussed later. It may maintain internal version information in addition to the producer supplied version information, but the format or semantics of the internal version information is out of scope of this specification.

On updates of the existing AA instance, the original AA instance was kept in the repository. AA instances will not be deleted until they are explicitly deleted through the `Destroy` operation. The deletion may be limited by the system design or by other entities, such as job management services. This may ensure the consumers of the AA instance keep using the referencing AA instance after creation of newer versions of the AA instance. Consumers and producers will have an access for any of the instance in the history as long as they have appropriate access permission to the AA instance.

The base version of AA instance and the newly created AA instance have bidirectional references in their Resource Properties; `baseAA` and `newerAA`. The newly created AA instance also has Resource Property called `DifferentialAAD`, which describes differential information from base version. Consumers and producers can trace entire version history using these Resource Properties. In addition, they can subscribe events on update of AA instances to receive the notifications. See Section 5.2.1and 5.3 for more detail.

When implementations allow to do delete the AA pointed by the `baseAA` of another AA instance, the resource properties of those referencing it should be updated and maintained in accordance by the repository. The implementations need to calculate the appropriate values.

3.2 Design Policy

3.2.1 OGSA WSRF Basic Profile Compliant

The OGSA WSRF Basic Profile 1.0 (hereafter, "the Basic Profile") [OGSA WSRF Basic Profile] is a set of normative profiles which defines uses of widely accepted specifications in order to ensure interoperability among Grid service implementations. The Basic Profile addresses issues relating to distributed resource management and Grid computing extending WS-I Basic Profile 1.1[WS-I BP

1.1]. It is strongly recommended that ACS implementation should be compliant with the Basic Profile.

The ACS specification is designed so as to be compatible and consistent with the Basic Profile. That is:

- ACS service is defined as WS-Resources; ApplicationArchive WS-Resource to represent an AA and ApplicationRepository WS-Resource to manage a repository. These are defined as distinct WS-Resources because they have different life spans.
- The ACS WS-Resources are referenced using WS-Addressing endpoint references (EPRs) as specified in the Basic Profile.
- The ACS WS-Resources support certain resource properties and WS-RF operations that are mandated to implement in the Basic Profile. For details, see Section 5.1.4 and 5.2.5.
- The ACS WS-Resources use WS-Notification mechanism for subscribing and publishing notifications as specified in the Basic Profile. For details, see Section 5.1.4 and 5.2.5.
- The ACS WS-Resources raise errors compliant with WS-BaseFault.
- The ACS specification specifies security mechanisms only for ACS-specific domains and follows the Basic Profile with regard to generic security issues. For details, see Chapter 7.

3.2.2 Data Handling

Some Application Contents such as binary executables and user data are likely to be huge in size. The ACS specification is designed to allow flexible implementation for handling such data.

3.2.2.1 Reference to the external storage

AA bundles files for a unit of task in logical sense, which means it allows some of the files can be stored outside of the ACS repository and let the ACS repository or consumer of the ACS resolve the reference, i.e. retrieve actual contents of the files.

There are distinct two places where the references to the external storage may appear for this version of specification:

1) SOAP messages carrying ARI operations

2) Application Contents

For the case 1), ACS repository will validate and resolve the reference on receipt of URL, i.e. the actual entity referred is retrieved and the reference is replaced by the one for the other part of the AA instance, which is an actual entity in the repository. The ACS repository will return fault if any will fail in the process. The topic will be covered more in detail in the Data Transport section below.

For the case 2), URLs will appear in the any part of the Application Contents, which is opaque to ACS repository. It is not recommended since it may cause some ambiguity for the consistency among the contents in AA and pose a question what will happen if the reference cannot be resolved. This could happen where the external storage referred was there but has gone away when it is to be retrieved. It is out of the ACS scope how these external references are resolved. The ACS specification does not ensure the existence or validity of the referenced entities, either.

The references should be accompanied with appropriate digest value of the actual entity, in order to detect the change or tampering of the actual entity after the reference was made. For the case 1), the implementation of the ACS repository may choose and specify the specific digest method to be used from the prevailed technologies, such as MD-5 or SHA-1. The reference in the ARI request

must conform to the XML Signature standard in its representation where it is applicable. For the case 2), it is out of scope of the ACS specification and is left to the implementations how those are presented inside the Application Contents.

3.2.2.2 Data Transport

In order to enable flexible upload and download of data, ACS allows the implementations of repository and its clients to choose methods and type of transport to be used for ARI operations. Minimum set of transport types and methods are specified as normative to provide the maximum interoperability, with extension points in ARI SOAP messages for use with the implementation specific extension which will rather limit the interoperability between the implementation of the ACS repository and its client.

Transport Type

ACS specifies the two type of transport of AA, *discrete* and *bundled*: With the *discrete* type of transport, files constituting an AA, are embedded or attached to the SOAP message as discrete files. With the *bundled* type, files constituting an AA are bundled into a single file in advance and then it is embedded or attached to the SOAP message, e.g. files can be bundled with various technologies such as zip, tar, and so on. ACS will also define the URIs within its namespace for popular one of those to indicate what is used for bundling. An implementation can use its own bundling method and define the URI for it. For the maximum interoperability, implementations of ACS repository are required to implement the discrete type. Implementations of ACS should expose the list of the URIs as its resource properties, indicating the supported transport types. ACS client selects one to be used from the list. See Section 5.1.1 for more details.

Transport Method

ACS defines *Simple* and *SwA* transport methods: with the *Simple* method files are BASE64-encoded and embedded into the SOAP message. With the *SwA* method, files are attached to the SOAP message as proposed as in [SwA]. For maximum interoperability, implementations of ACS are required to implement the *Simple* method. However, SwA would be preferred method for most of the implementations. It is also allowed to support implementation-specific transport methods such as file, ftp, gridFtp, http, etc. Implementations of ACS should expose the list of the URIs as its resource properties, indicating the supported transport method. ACS client selects one to be used from the list. See Section 5.1.1 for more details.

The figure below illustrates a non-normative example of implementation-specific transport method, where an external http server is used to upload data to the ACS repository.

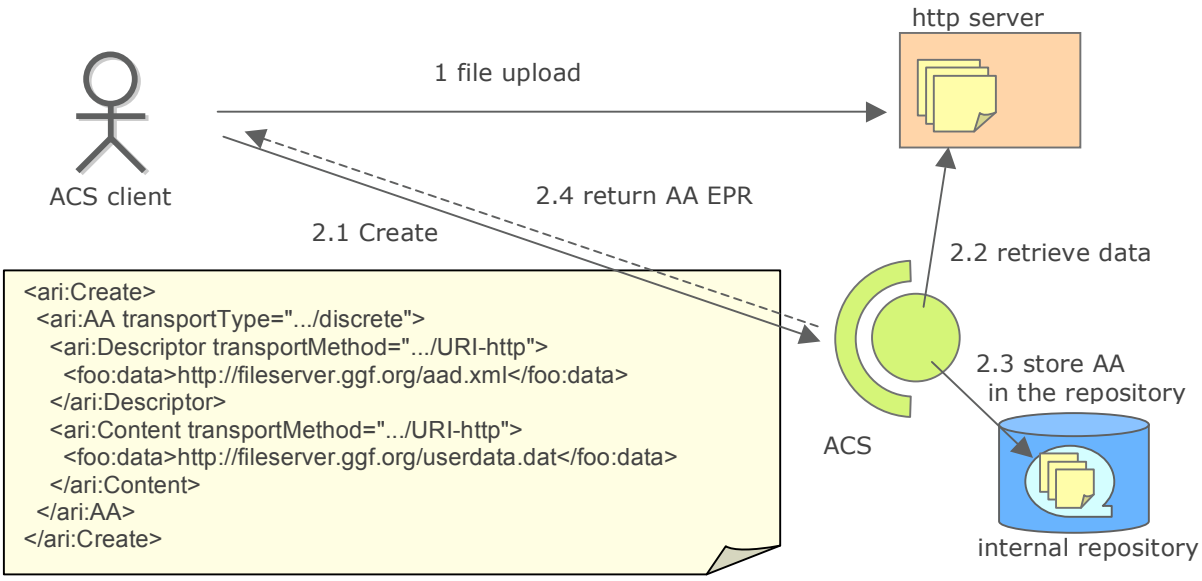

Figure 3-3 Example of Create operation using external http server.

In this case, the client specifies the *discrete* as the `transportType` and URI-http as the `transportMethod`, indicating the http protocol is used for the data transport. As such, the URLs to the real entities are specified in the `Create` message. The implementation of ACS repository is responsible for resolving the contained URLs and collecting real entities of the data.

3.2.3 Asynchronous Messaging

It may consume much time to collect necessary data and create a valid AA instance. Therefore, the ACS service is recommended to implement `Create`/`Update` operation as an asynchronous one. That is, the Implementation of the ACS repository creates an empty AA instance on accepting a request message and returns its EPR before collecting required information to initialize the AA instance. The AA instance is in the initializing state at this moment and does not accept any operations except for `Destroy` operation.

The ACS client should be implemented assuming that the newly created AA instance may be in initializing state. The ACS client can check whether the initialization is successfully completed in the following ways:

- The ACS client attempts to access an arbitrary operation of the AA instance except for `Destroy`. If it is in the initializing state or it has failed to be initialized, an error is returned to the client.
- The ACS client subscribes to the implementation of the ACS repository so that it is notified when initialization of the AA Instance is completed or failed.

3.3 Relationship to other activities

3.3.1 OASIS SDD TC

OASIS SDD TC is a Technical committee "*defining a standardized way to express software installation characteristics required for lifecycle management in a multi-platform environment" on its charter.* [1] The SDD TC chair is also a co-chair of the ACS-WG. The table below depicts the baseline comparison between SDD and ACS.

Table 3-1 Relationship summary

	Solution Deployment Descriptor (SDD)	Application Contents Service (ACS)
Purpose	To define a standard archive format for multiple applications to be installed on the multiple heterogeneous platforms.	
Descriptor (Manifest file)	An XML document to describe its contents. (Solution Deployment Descriptor in SDD, Archive Descriptor in ACS)	
Target platform	A set of Heterogeneous and multiple platforms.	
Scope	Grid application/system is among the scope	Grid application/system is the scope
Emphasis	Package format and Deployment Descriptor.	Application Archive Format and Application Repository Interface
Contents in an archive	Any combination of applications that needs to be installed.	A set of applications that works together to fulfill the job submitted to the grid system.
Structure	Hierarchically layered installable units	Flat files and logical description of those relationships.

Even though the motivation and the scope in detail varies each other, there are potentials that both share single descriptor to meet requirements on both, either can generalize to accommodate the other, or both stands separately keeping capability to transformation or convert each other to gain the interoperability. There were long discussions in the ACS WG on this.

At the time of creation of this specification, the scope and milestones of the SDD TC is yet to be adjusted from its original description. So the current thought for the relationship is to best describe the functional requirements (or use case) on SDD from the perspective ACS, presented it to SDD TC, and wait for their output to see how the two can fit each other.

3.3.2 OGSA EMS Design Team

The Execution Management Services (OGSA-EMS) is currently be worked by OGSA EMS design team and another OGSA related WGs, including the Basic Execution Services working group (BES-WG), RSS-WG, CDDLM-WG, GRAAP-WG, and ACS-WG. EMS encompasses all the services needed to schedule, execute, and manage jobs executing on the grid. ACS repository is among the

[1] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=sdd

sources that EMS could gain access to meta task information indirectly or directly prior to execution in order to meet optimal resources utilization.

The diagram below depicts possible interactions with EMS components. Please note this is a snapshot of the ongoing discussion and the final outcome of this topic should be found in their document. The service container would house the installed application contents through the execution of the job. A possible interaction between the provisioning service and ARI would be to obtain a list of supported transport protocols and security policies.

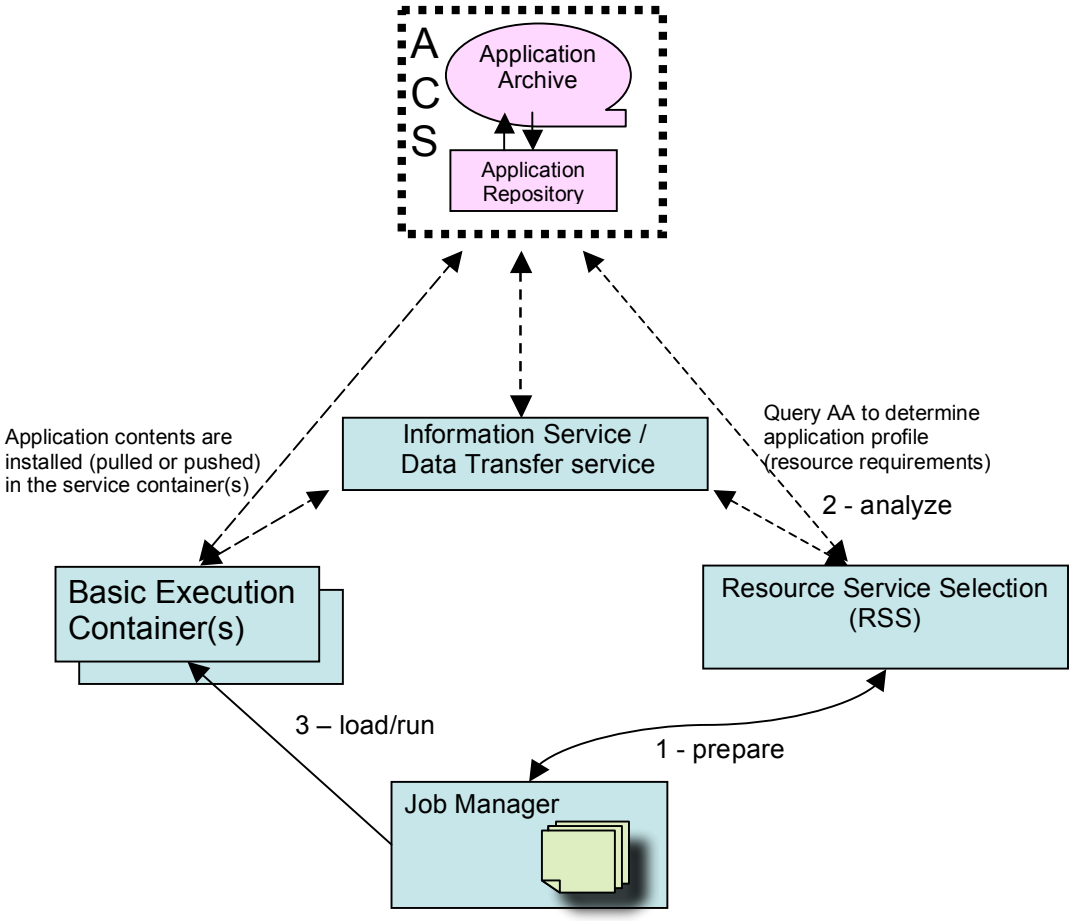

Figure 3-4 Possible interactions with services in EMS

3.3.3 CDDLM WG

CDDLM-WG is a working group in the Global Grid Forum (GGF)[1]. CDDLM stands for Configuration Description, Deployment, and Lifecycle Management. It is expected to be an OGSA service to constitute a grid system and collaborate with diversity of other OGSA compliant services and constituted five documents [CDDLM-FND], [CDDLM-SF], [CDDLM-CDL], [CDDLM-CMP], [CDDLM-API].

We see potential interactions of implementations ACS with those of the CDDLM component. CDDLM-WG also assumes, in Section "3.2 File upload" of the "Deployment APIs Specification", that the existence of a full asset store outside of their implementations, in which components are stored, and the API of the asset store should be used by implementations of the CDDLM instead.

As such, here we present an example that shows a possible interaction between CDDLM and ACS, based on the outcome of our joint discussions. The descriptions below constitute a non-normative part of this specification and the details are to be decided by the design of the system implementation.

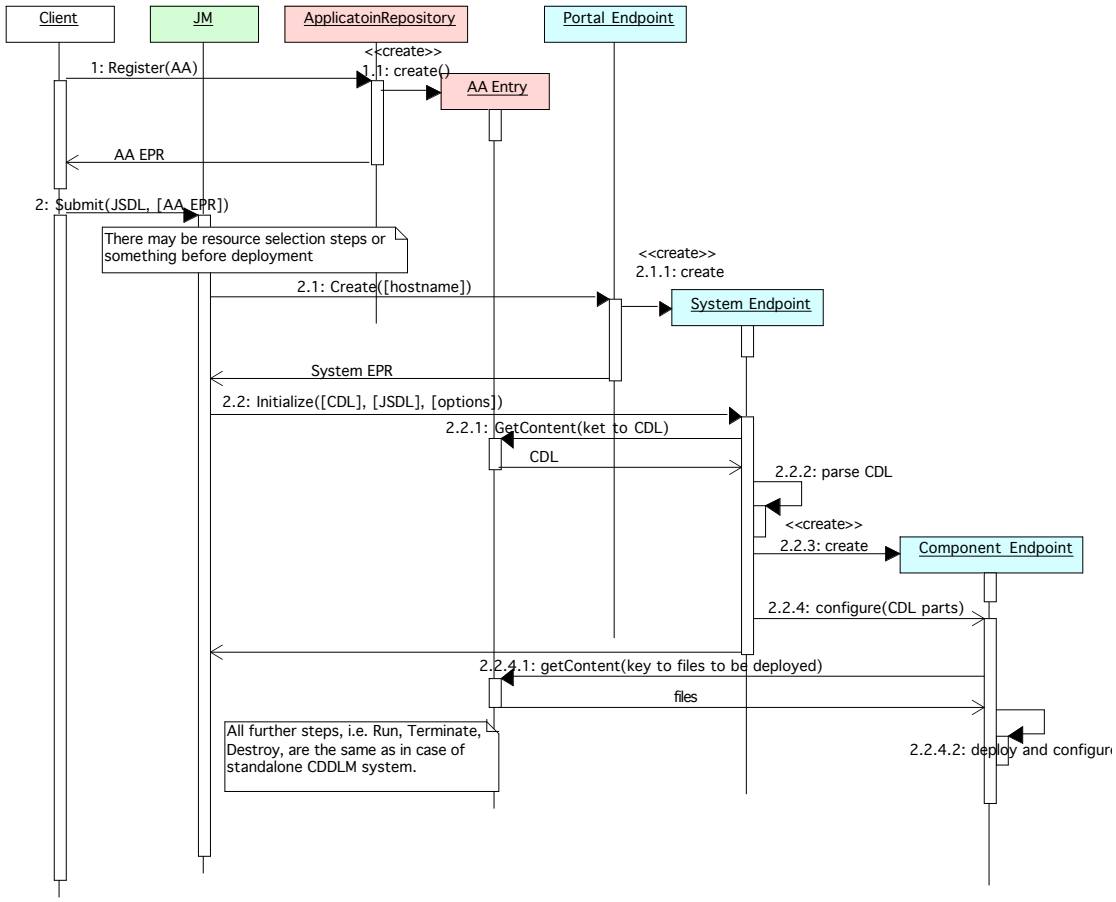

Figure 3-5 Possible Interaction with CDDLM (example)

[1] https://forge.gridforum.org/docman2/ViewCategory.php?group_id=130&category_id=550

3.3.4 OGSA Data WG

The OGSA Data Working Group defines the data services which provide for access to data on the grid. ACS can utilize protocols being specified such as Byte-IO, GridFTP, and possibly Replication. ACS provides an extension point for protocols which allow implementations to specify which protocols are supported. The ACS team has reviewed their briefings and draft documents with interest. Interaction with the OGSA Data Working group is expected to increase during the development of the next version of this specification.

3.3.5 OASIS WSDM TC

The OASIS Web Service Distributed Management (WSDM) TC defines two set of specifications for distributed resource management: Management Using Web Services (MUWS) and Management Of Web Services (MOWS). MUWS provides interoperable, base manageability for monitoring and control managers using Web services. It defines a basic set of manageability capabilities such as identification, discovery, monitoring and so on. MOWS defines the manageability model for managing Web services as a resource and how to describe and access that manageability using MUWS.

We understand that the manageability defined by WSDM would the appropriate standards for implementations of ACS as for the manageability aspect. Thus, the ACS specification leaves the manageability aspect to the standards set by external references such as OGSA profiles.. The ACS specification focuses on the functional aspect of the ACS, i.e. on the service interface and archive format in order to gain the maximum interoperability among implementations.

4. Example Usages

4.1 Roles

This section describes roles of ACS users. Note that these roles are used purely for the purposes of describing the usage of ACS. These roles are not explicitly identified or enforced by ACS. The roles are:

- *Producer* – owner of an application that is to be archived into ACS. The producers create, update, and delete AA instances.
- *Consumer* – users of ACS interface that want to retrieve information about AA instances or retrieve contents of AA.
- *Replicator* – a special version of consumer that enables getting a formatted version of the complete AA for the purposes of recreating the AA in another instance of ACS.

4.2 Creation of an AA Instance

Creation of AA Instance means to create an AA instance in an ACS repository. This is realized via `Create` operation defined in the ARI. To invoke the operation, producers need to build an AA document and send it to the ACS repository via `Create` request message. The AA document is a logical bundle of an AA, which consists of an AAD (XML document file) and zero or more Application Content file(s). The process to build an AA document is not specified in ACS, but some possible scenarios are described below.

Case 1: Creating an AA instance through web-based portal service (Figure 4-1). Application Content files, such as executables and user data, and administrative information, such as identification and access control policy, are transmitted by Producers to the portal service. The portal service generates AAD from the given information and invokes `Create` operation of ACS, accompanied with AAD and all Application Content files.

Case 2: Building an AA document using developer's tool, i.e. an ACS compliant IDE. The tool assists the application developer to assemble application content files, to describe an AAD and to generate an AA document. In addition to building an AA document, the tool may invoke `Create` operation (Figure 4-2). If this is the case, the tool acts the same role as portal service in case 1. More typically, the producer and developer may belong to different organization and be separated geographically or in some way (Figure 4-3). In that case, the `Create` operation will be invoked by producer taking a bundled AA document as an input.

As explained in Section 3.2.2.2, there are two physical types of AA document; bundled and discrete. The bundled type of AA document is a single archived file which contains an AAD and Application Content files, whose file structure is compliant with AAF defined in this specification. In the "Case 2" described above, the bundled type of AA document is naturally expected since it is easier to move around a single file than dozens of files. Note that both types are logically equivalent and the portal service and tool may support either type for `Create` operation. Also, ACS defines mechanism for ACS implementation to select appropriate transport method in transferring AA document. It is not described here.

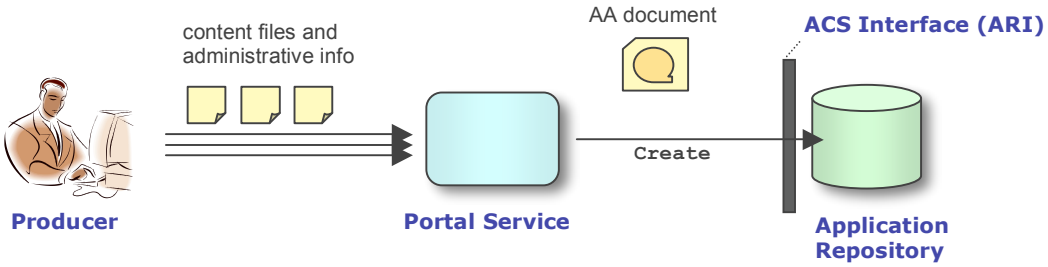

Figure 4-1 Case 1: Creating AA instance through portal service

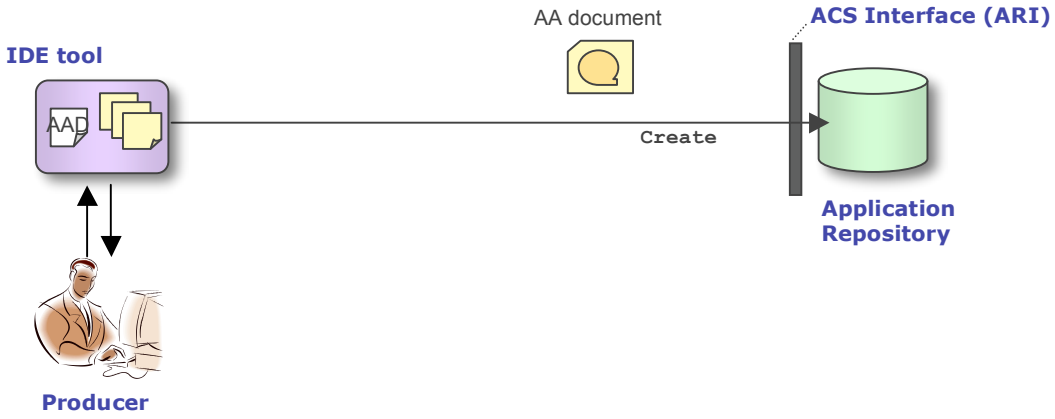

Figure 4-2 Case 2: Building AA document using IDE tool

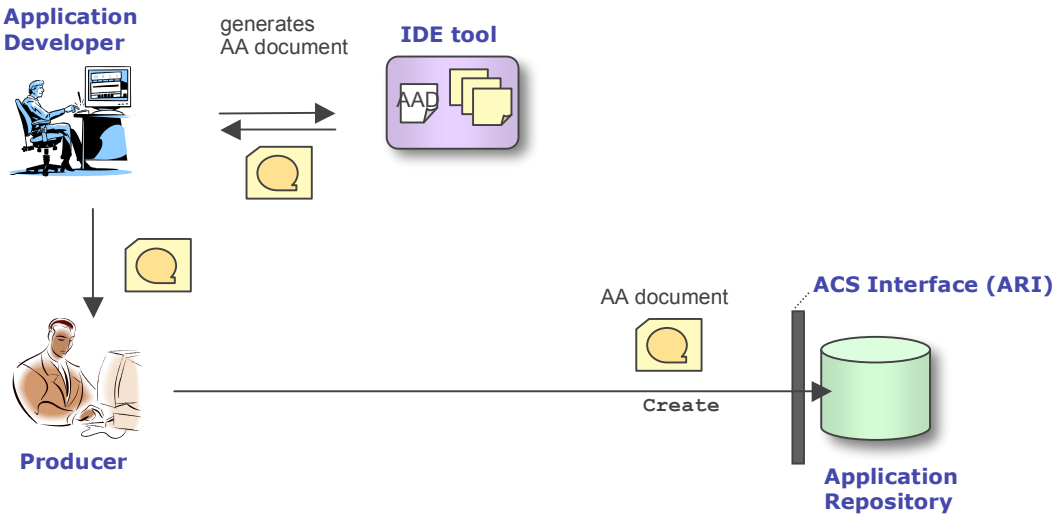

Figure 4-3 Case 2: Building AA document using IDE tool (Developer and Producer are separated)

4.3 Retrieval of Application Contents

Consumers can retrieve arbitrary Application Contents from an AA instance through `GetContents` operation. In order to retrieve Application Contents, consumers need to have an EPR of the AA instance. Though ACS doesn't specify how consumers obtain the EPR returned to the producer of the AA instance, there can be multiple possibilities depending on the design of the system. At the creation time of an AA instance, the key information to select the necessary contents such as types or names of the contents should be provided in an AAD by producers, and exposed by ACS repository as Resource Properties of the AA instance. Consumers can get them through WS-ResourceProperties operations and specify the key information as parameter of `GetContents` operation.

Here we illustrate how a provisioning service in a grid system can work with ACS repository. Note that the description here is not normative and does not limit the use of ACS.

Possible sequence of grid application provisioning:

Step 1: When a producer wants to submit a job into a grid system, it creates an AA instance in the ACS repository for the system and obtains its AA EPR. After that, it can send the AA EPR to Job Manager at the time of job submission.

Step 2: Job Manager determines nodes on which the job is executed, using the Resource Selection Service (RSS). In this step, RSS services may utilize information stored in the AA instance via ARI operations on the AA instance identified by the AA EPR. Then Job Manager retrieves a deployment descriptor from the AA instance, using `GetContents` operation with the corresponding type of contents as key information. The `GetContents` message would be like below.

```
<ari:GetContents>
  <ari:queryExpression
      dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
boolean(/aaf:contents/aaf:content/@type
= "aaf:DeploymentDescriptor")
  </ari:queryExpression>
  <ari:transportMethod>
     http://www.gridforum.org/acs/ari/transport-method/embedded
  </ari:transportMethod>
</ari:GetContents>
```

Step 3: Job Manager sends the obtained deployment descriptor to deployment service. The deployment service parses the descriptor and instantiates application specific deployment engines. Though the ACS doesn't limit the type of the deployment service, if it was the CDDLM, for example, the deployment engines are called Component services. The code files for the Component are also contained in the AA instance as an Application Contents and CDDLM service retrieves them through another `GetContents` operation. The information to be used in order to specify the code files should be provided in the deployment descriptor. The actual deployment processes are performed by Component services.

4.4 Updating of an AA instance

Updating of an AA Instance means to create another AA instance based on an existing AA instance in the ACS repository. This is realized via `Update` operation defined in the ARI. In order to update an AA instance, producers build an AA document that contains the difference from the base AA

instance and send it to ACS via `Update` request message, accompanied by change information that tells which files should be replaced, added or removed. The AA document here consists of an AAD, content files that are to replace the original or to be simply added. This type of AA document is called partial AA document since it may not contain all contents required to create new AA instance.

When accepting the request, the ACS repository creates a new version of AA instance and returns the new AA EPR. Note that the original AA instance remains unmodified and consumers can access the old version of AA instance. The old and new versions of AA instances have bidirectional references on their Resource Properties. One can invoke multiple `Update` operations on single AA instance, resulting in multiple versions of the AA instances that share the same AA instance for a base version Resource Property.

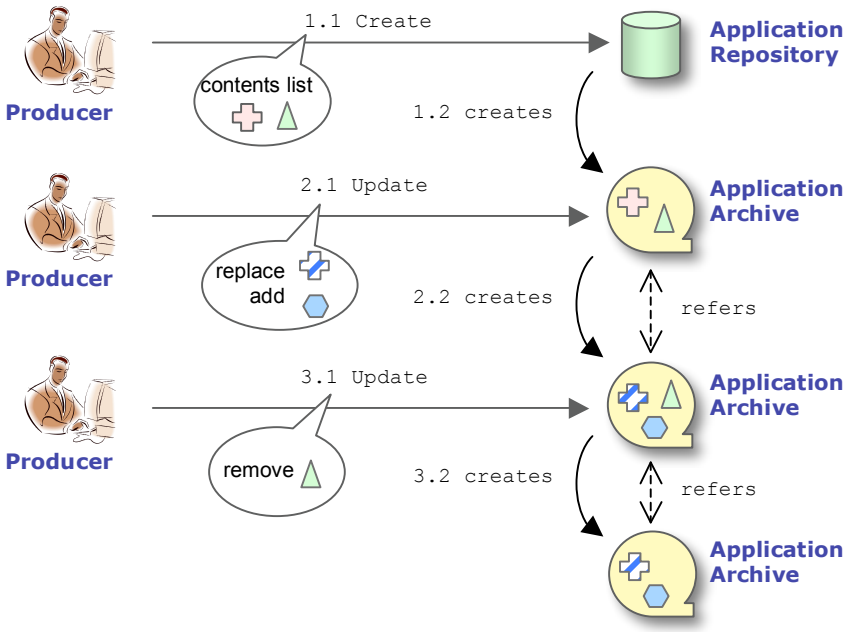

Figure 4-4 Updating AA instances

Alternatively, Producers may invoke `Create` a new AA instance copying from an existing AA. In this case, producer downloads a complete AA document from the existing AA instance and creates an AA instance with modified one. Instead of Update, the standard Create operation can be used in this case. The base AA and newly created AA don't have references each other. Additional version management can be explored by the implementations of this scenario, but the ACS doesn't specify the detail in this point.

4.5 Replication of an AA instance

Replication/synchronization of AA instance is one of the advanced ACS use cases. Many times, an enterprise system may wish to distribute non-realtime or infrequently updated data to many locations for WAN load distribution, different physical site redundancy, and reducing routes/long hauls. ACS is a good fit for this type of behavior. ACS defines the minimum set of operations enabling these but leaves the sophisticated mechanisms for the designers of implementations. ACS provides notification mechanism so that replicators can catch the event notification on changes to an AA instance. Also, ACS provides an access to the event history in a given period of time. The notification mechanism to be used is based on the WS-Notification specification. The partial update mechanism described in Section 5.2.2 can be used to save replication costs in time and bandwidth. For example, if a part of an AA instance is changed, replicators can update only the changed contents. Note that the update in smaller granularity than Application Contents is out of ACS scope, though it may be handled in a consumer dependent way, for example, providing a binary patch.

The simplest examples of replicating AA instance and its contents are shown below.

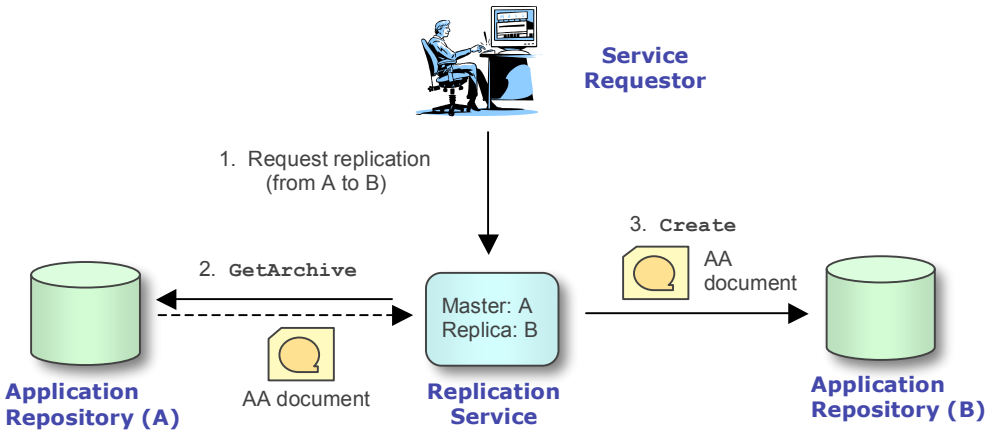

Figure 4-5 Simplest replication sample: Creating new replica

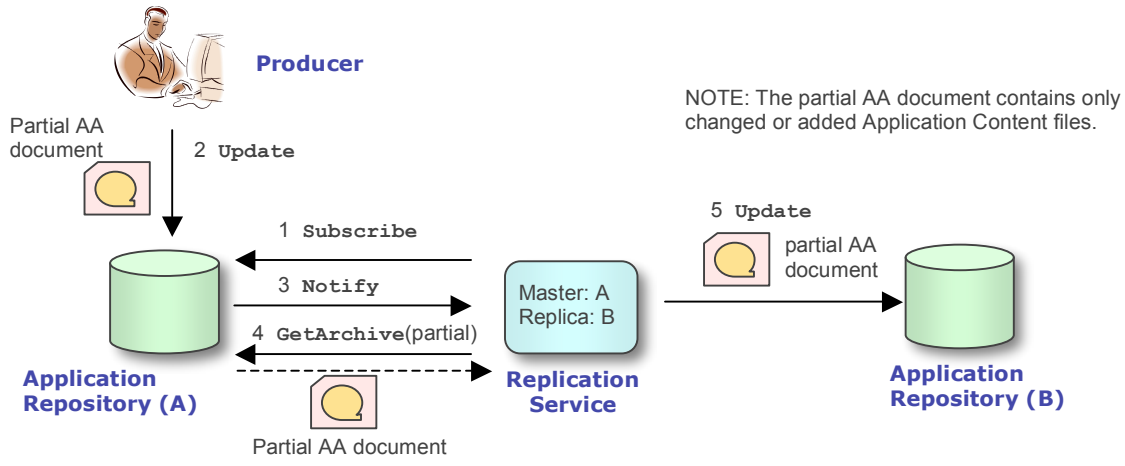

Figure 4-6 Simplest replication sample: Synchronizing master and replica

5. Application Repository Interface Specification

5.1 ApplicationRepository portType

The ApplicationRepository is a WS-Resource which represents an implementation of the ACS repository. This section describes about the Resource Properties, Operations and other requirements that this portType must support.

5.1.1 Resource Properties

The format of the Resource Properties document defined by ApplicationRepository portType is shown below:

```
<ari:Version>xsd:anyURI</ari:Version>
<ari:TransportType>xsd:anyURI</ari:TransportType>+
<ari:TransportMethod>xsd:anyURI</ari:TransportMethod>+
<ari:QueryExpressionDialect>
  xsd:anyURI
</ari:QueryExpressionDialect>+
```

The elements in the Resource Property document are described as follows:

/ari:Version
: This REQUIRED element indicates the version of the ACS specification which the repository implementation is based on. Currently, the following URI is available as the version:
  - `http://schemas.ggf.org/acs/2005/10/ari`

/ari:TransportType
: This REQUIRED element indicates ranges of the transport types that are supported by the implementation of the ACS repository as a list of URIs. In this specification the following transport type URIs are reserved:
  - `http://schemas.ggf.org/acs/2005/10/ari/transport-type/discrete`
    Data is transported as discrete files. For the maximum interoperability, all implementations of the ACS repository MUST support this transport type.
  - `http://schemas.ggf.org/acs/2005/10/ari/transport-type/bundled/zip`
    Data is transported as a bundled file of multiple contents using ZIP format.
    Other transport type using another bundling technologies and a corresponding URI MAY be defined and supported by implementations of ACS repository.

/ari:TransportMethod
: This REQUIRED element indicates ranges of the transport methods that are supported by the implementation of the ACS repository as a list of URIs. In this specification the following transport method URIs are reserved:
  - `http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded`
    Data is BASE64-encoded and embedded in the SOAP body. For the maximum interoperability, all implementations of the ACS repository MUST support this transport method.
  - `http://schemas.ggf.org/acs/2005/10/ari/transport-method/SwA`
    Data is attached to the SOAP message conforming to [SwA].
    Other transport method and a corresponding URI MAY be defined and supported by implementations of ACS repository.

/ari:QueryExpressionDialect
: This REQUIRED element indicates ranges of query expression dialects supported by the

implementation of the ACS repository for use with LookupArchives operation as a list of URIs. There are two well-known dialects identified by this specification, corresponding to two versions of the XPath language.

- `http://www.w3.org/TR/1999/REC-xpath-19991116`
  This URI identifies the XPath 1.0 language. The contents of the QueryExpression MUST be a string containing a valid XPath 1.0 expression.
- `http://www.w3.org/TR/2003/WD-xpath20-20031112`
  This URI identifies the Xpath 2.0 (working draft) language. The contents of the QueryExpression MUST be a string containing a valid XPath 2.0 expression. NOTE: It is described that "additional URI will be added to represent the W3C Recommendation form of the XPath 2.0 language.2" in WSRF specification. As such, the described URI is not final and subject to be updated.
  The requestor specifies query expression dialect URI as well as query expression itself in the LookupArchives operation. It is allowed for implementations of the ACS repository to support additional value for dialects with URIs in addition to the above.

5.1.2 Create operation

This operation creates an AA instance in the repository.

An AA document, in the form of either a bundled archive or discrete files, is transported in the Create request message. The transportType is specified for an AA and the transportMethod is specified for a bundled file or each of the discrete files.

5.1.2.1 Request message

The format of the Create request message, in case that the AA document is presented as a bundled file, is shown below:

```
<ari:Create>
  <ari:AA transportType="URI of bundled file transport type">
    <ari:Bundle>
      ari:TransportedDataType
    </ari:Bundle>
  </ari:AA>
  {any}*
</ari:Create>
```

The format of the Create request message, in case that the AA document is presented as discrete files, is shown below:

```
<ari:Create>
  <ari:AA transportType="URI of discrete files transport type">
    <ari:Descriptor>
      ari:TransportedDataType
    </ari:Descriptor>
    <ari:Content pathname="aaf:RelativePathnameType">
      ari:TransportedDataType
    </ari:Content>*
  </ari:AA>
  {any}*
</ari:Create>
```

The elements in the Create request message are described below:

/ari:Create/ari:AA
This REQUIRED element contains the bundle file for an AA document.

/ari:Create/ari:AA/@transportType
This REQUIRED attribute specifies the URI for the transport type the requester of the Create operation, which is chosen from the supported transport types by the implementation of the ACS repository.

/ari:Create/{any}
Provides an extension point for the implementation-specific information of the request.

/ari:Create/ari:AA/ari:Bundle
This is a REQUIRED element for bundled file transport type. It contains information of the bundled archive. This must appear exactly and only once. The details of TransportedDataType are explained later.

/ari:Create/ari:AA/ari:Descriptor
This is a REQUIRED element for discrete files transport type. It contains information of the AAD in the AA. The details of TransportedDataType are explained later.

/ari:Create/ari:AA/ari:Content
This is a REQUIRED element for discrete files transport type. It contains information of an Application Content in the AA. The details of TransportedDataType are explained later.

/ari:Create/ari:AA/ari:Content/@pathname
This is a REQUIRED attribute for discrete files transport type. It specifies the path of the Application Content described in the AAD. This attribute is for specifying which Application Content the ari:Content element is describing.

ari:TransportedDataType

ari:Bundle, ari:Descriptor, and ari:Content elements above are of ari:TransportedDataType, which is shown below:

```
<some-element transportMethod="xsd:anyURI">
 (
  <ari:Binary>xsd:base64Binary<ari:Binary/> |
  <ari:AttachmentRef>ref:swaRef</ari:AttachmentRef> |
  {any}*
 )
</some-element>
```

/@transportMethod
This REQUIRED attribute specifies the URI of the transport method for bundled archive, AAD, or Application Content. Depending on the value for this attribute, different child elements are contained.

/ari:Binary
This OPTIONAL element specifies the data of bundled archive, AAD, or Application Content to be transported. This element is REQUIRED if the value of transportMethod attribute is "`http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded`".

/ari:AttachmentRef
This OPTIONAL element specifies the URI of bundled archive, AAD, or Applicaiton Content attached to the SOAP envelope. This element is REQUIRED if the value of transportMethod attribute is "`http://schemas.ggf.org/acs/2005/10/ari/transport-method/SwA`".

/{any}
Provides an extension point for the implementation-specific information when implementation-specific transport method is used.

5.1.2.2 Response message

The format of the response message to the Create request message is shown below:

```
<ari:CreateResponse>
  <ari:ArchiveEPR>wsa:EndpointReferenceType</ari:ArchiveEPR>
</ari:CreateResponse>
```

The contents of the CreateResponse message are described below:

/ari:CreateResponse/ari:ArchiveEPR
This REQUIRED element specifies the endpoint reference of the created AA instance. The actual creation of AA instance may be executed asynchronously to the response, and the requestor of the Create should prepare that following operation using the EPR may result in ResourceNotReadyFault.

5.1.2.3 Faults

The types of faults for the Create operation are as follows:

ResourceUnknownFault
The WS-Resource (ApplicationRepository) identified in the message is not known to the Web service.

IllegalDescriptorFault
The specified AAD does not conform to the Application Archive Format part of the ACS specification. Note that it does not include the case that the Create message has syntax errors with respect to WSDL definition. The ApplicationRepository portType must at least check that:

1. the AAD in the AA document conforms to the specification.
2. files which are listed in the AAD are contained in the AA document.
3. Full AA document is specified in the request message. Note that differential AA document is allowed only in the Update request message defined in ApplicationArchive portType.

TransportTypeNotSupportedFault
The specified transport type is not supported by the repository.

TransportMethodNotSupportedFault
The specified transport method is not supported by the repository.

CreationFailedFault
The operation failed due to internal reasons.

Implementations of ACS repository MAY augment other fault types that are implementation dependent. Requester SHOULD prepare for the implementation specific faults.

5.1.3 LookupArchives operation

This operation retrieves the list of AA instances that meet the given query condition in the ACS repository. The query is performed on the Resource Properties Documents of the AA instances in the repository.

This operation can be used, for example, for the following purposes:

- in order to retrieve the list of AAs which have the specified AAID
- in order to retrieve the list of AAs which belong to a certain owner
- in order to retrieve the list of all the versions of archives of a certain application

5.1.3.1 Request message

The format of the LookupArchives request message is shown below:

```
<ari:LookupArchives>
  <ari:QueryExpression dialect="xsd:anyURI">
     xsd:any
  </ari:QueryExpression>
</ari:LookupArchives>
```

The contents of the LookupArchives request message are described below:

/ari:LookupArchives/ari:QueryExpression
: This REQUIRED element indicates a query expression evaluated for the Resource Property Documents of all the AA instances which the requester can access.

/ari:LookupArchives/ari:QueryExpression/@dialect
: This REQUIRED attribute indicates a URI that identifies the language of the query expression.

5.1.3.2 Response message

The format of the response message to the LookupArchives request is shown below:

```
<ari:LookupArchivesResponse>
  <ari:AAInfo>
    <ari:EPR>wsa:EndpointReferenceType</ari:EPR>
    {any}*
  </ari:AAInfo>*
</ari:LookupArchivesResponse>
```

The contents of the LookupArchivesResponse message are described below:

/ari:LookupArchivesResponse/ari:AAInfo
: Indicates information of one of the matching AA instances.

/ari:LookupArchivesResponse/ari:AAInfo/ari:EPR
: This REQUIRED element indicates the endpoint reference of the AA instance.

/ari:LookupArchivesResponse/ari:AAInfo/{any}
: Provides an extension point for the implementation-specific information of the request.

If there is no AA matching the query expression, a response message without an ari:AAInfo element MUST be returned.

5.1.3.3 Faults

The types of faults for the LookupArchives operation are as follows:

ResourceUnknownFault
: The WS-Resource (ApplicationRepository) identified in the message is not known to the Web service.

UnknownQueryExpressionDialectFault
The given query expression has a dialect that is unknown to the repository service.

InvalidQueryExpressionFault
The given query expression is invalid within the query expression language identified by the dialect attribute.

LookupFailedFault
The operation failed due to internal reasons.

Implementations of ACS repository MAY augment other fault types that are implementation dependent. Requester SHOULD prepare for the implementation specific faults.

5.1.4 Other requirements for the ACS implementation

To be conformant to OGSA WSRF Basic Profile, the ApplicationRepository MUST implement the following portTypes.

portType name	Purpose
wsrp:GetResourceProperty	in order to enable to read the Resource Properties
wsrp:GetMultipleResourceProperties	in order to enable to read the Resource Properties
wsrl:ImmediateResourceTermination	in order to enable repository destruction

If the implementation of the ACS repository supports event notification, ApplicationRepository MUST implement the following portType.

portType name	purpose
wsnt:NotificationProducer	in order to enable to subscribe for AA creation/update notifications

5.2 ApplicationArchive portType

The ApplicationArchive is a WS-Resource which represents an Application Archive (AA) instance in the repository. Update of an AA instance should result in a different AA instance with a different version inside its AAID, leaving the original AA instance as it is until it is explicitly deleted. This section describes about the Resource Properties, Operations and other requirements that this portType must support.

5.2.1 Resource properties

The format of the Resource Properties document defined by ApplicationArchive portType is shown below:

```
<aaf:AAD>aaf:AADType</aaf:AAD>
<aaf:DifferentialAAD>aaf:DiffAADType</aaf:DifferentialAAD>?
<ari:CreationDateTime>xsd:dateTime</ari:CreationDateTime>
<ari:BaseAA>wsa:EndpointReferenceType</ari:BaseAA>?
<ari:NewerAA>wsa:EndpointReferenceType</ari:NewerAA>*
<ari:Repository>wsa:EndpointReferenceType</ari:Repository>
<ari:QueryExpressionDialect>
  xsd:anyURI
</ari:QueryExpressionDialect>+
```

The elements in the Resource Property document are described below:

/aaf:AAD
This REQUIRED element indicates the AAD of this AA instance.

/aaf:DifferentialAAD
This OPTIONAL element indicates the Differential AAD of this AA instance from the base AA. This element is REQUIRED when and only when this AA instance was created using Update operation on another AA instance.

/ari:CreationDateTime
This REQUIRED element indicates the date and time when this AA instance was created.

/ari:BaseAA
This OPTIONAL element indicates the endpoint reference of the AA instance which this AA instance is based on. This element is REQUIRED when and only when this AA instance was created using `Update` operation on another AA instance.

/ari:NewerAA
Indicates the list of AA instances which are created by Update operations on this AA instance.

/ari:Repository
This REQUIRED element indicates the endpoint reference of the repository which holds this AA instance.

/ari:QueryExpressionDialect
This REQUIRED element indicates ranges of a query expression dialect that is used in the GetContents operation and supported by the AA instance as a list of URIs. There are two well-known dialects identified by this specification corresponding to two versions of the XPath language (for details, see Section 5.1.1). The requestor specifies query expression dialect URI as well as query expression itself in the `GetContents` operation.

5.2.2 Update operation

This operation creates a new AA instance with a difference from the existing AA instance in the repository.

A differential AA document, in the form of either a bundled archive or discrete files, is transported in the Update request message. The Update request message is syntactically the same as Create request but the differential AA document only contains Application Contents to be added to or replaced with those in base AA instance and Differential AAD which describes the differential information. Like Create operation, both the type and method for transport are specified for a bundled file or each of the discrete files from the transport types and methods supported by the Application Repository.

5.2.2.1 Request message

The format of the Update request message is shown below:

```
<ari:Update>
  <ari:AA ...>...</ari:AA>
  {any}*
</ari:Update>
```

The contents of the Update request message are described below:

/ari:Update/ari:AA
This REQUIRED element indicates an AA document of the updating AA. The format of this element, including type and method for transport, is as same as that in Section 5.1.2.1.

/ari:Update/{any}
Provides an extension point for the implementation-specific information of the request.

5.2.2.2 Response message

The response to the Update request message is a message of the following form:

```
<ari:UpdateResponse>
  <ari:ArchiveEPR>wsa:EndpointReferenceType</ari:ArchiveEPR>
</ari:UpdateResponse>
```

The contents of the UpdateResponse message are described below:

/ari:UpdateResponse/ari:ArchiveEPR
This REQUIRED element specifies the endpoint reference of the AA instance which is newly created by updating.

5.2.2.3 Faults

The types of faults for the Update operation are as follows:

ResourceUnknownFault
The WS-Resource (ApplicationArchive) identified in the message is not known to the Web service.

ResourceNotReadyFault
The WS-Resource (ApplicationArchive) identified in the message is not ready to be used.

ResourceUnavailableFault
The WS-Resource (ApplicationArchive) identified in the message is not available because the Create (Update) operation which instantiated the WS-Resource failed. The wsrf-bf:FaultCause element of this fault SHOULD describe the Creation(Update)FailedFault.

IllegalDescriptorFault
The specified AAD does not conform to the Application Archive Format part of the ACS specification. Note that it does not include the case that the Create message has syntax errors with respect to WSDL definition. The ApplicationRepository portType must at least check that:

1. the AAD in the AA document conforms to the specification.
2. files which are listed in the AAD are contained in the AA document.
3. a differential AA document is specified in the request message.

TransportTypeNotSupportedFault
The specified transport type is not supported by the repository.

TransportMethodNotSupportedFault
The specified transport method is not supported by the repository.

UpdateFailedFault
The operation failed due to internal reasons.

Implementations of ACS repository MAY augment other fault types that are implementation dependent. Requester SHOULD prepare for the implementation specific faults.

5.2.3 GetContents operation

This operation retrieves one or more Application Contents that satisfy the given query expression over the AAD contained in the specified AA instance.

Requesters can specify the preferred transport method, from supported methods found in the Resource Properties of the Application Repository.

5.2.3.1 Request message

The format of the GetContents request message is shown below:

```
<ari:GetContents>
  <ari:QueryExpression dialect="xsd:anyURI">
     xsd:any
  </ari:QueryExpression>
  <ari:TransportMethod>xsd:anyURI</ari:TransportMethod>
</ari:GetContents>
```

The contents of the GetContents request message are described below:

/ari:GetContents/ari:QueryExpression
: This REQUIRED element indicates a query expression evaluated for the AAD document of the AA instance. The query expression MUST be constructed so that the evaluation result represents a set of aaf:Content elements in the AAD document.

/ari:GetContents/ari:QueryExpression/@dialect
: This REQUIRED attribute indicates a URI that identifies the language of the query expression.

/ari:GetContents/ari:TransportMethod
: This REQUIRED element indicates the transport method of the matching Application Contents. The content of this parameter is a transport method URI. The matching Application Contents MUST be returned in the transport method specified in this parameter.

5.2.3.2 Response message

The format of the GetContents response message is shown below:

```
<ari:GetContentsResponse>
  <ari:Content ... />*
</ari:GetContentsResponse>
```

The contents of the GetContentsResponse message are described below:

/ari:GetContentsResponse/ari:Content
: Information of an Application Content in the AA. The format of this element is as same as that in Section 5.1.2.1.

If there is no Application Content matching the query expression, a response message without an ari:Content element MUST be returned.

5.2.3.3 Faults

The types of faults for the GetContents operation are as follows:

ResourceUnknownFault
The WS-Resource (ApplicationArchive) identified in the message is not known to the Web service.

ResourceNotReadyFault
The WS-Resource (ApplicationArchive) identified in the message is not ready to be used.

ResourceUnavailableFault
The WS-Resource (ApplicationArchive) identified in the message is not available because the Create (Update) operation which instantiated the WS-Resource failed. The wsrf-bf:FaultCause element of this fault SHOULD describe the Creation(Update)FailedFault.

UnknownQueryExpressionDialectFault
The given query expression has a dialect that is unknown to the AA instance.

InvalidQueryExpressionFault
The given query expression is invalid within the query expression language identified by the dialect attribute.

TransportMethodNotSupportedFault
The specified transport method is not supported by the AA instance.

Implementations of ACS repository MAY augment other fault types that are implementation dependent. Requester SHOULD prepare for the implementation specific faults.

5.2.4 GetArchive operation

This operation retrieves an AA document of the AA instance which the request is made for. Requesters can specify the preferred transport type and method, from supported types and methods found in the Resource Properties of the Application Repository.

5.2.4.1 Request message

The format of the GetArchive request message is shown below:

```
<ari:GetArchive>
  <ari:Differential>xsd:boolean</ari:Differential>?
  <ari:TransportType>xsd:anyURI<ari:TransportType>
  <ari:TransportMethod>xsd:anyURI</ari:TransportMethod>
</ari:GetArchive>
```

The contents of the GetArchive request message are described below:

/ari:GetArchive/ari:Differential
"true" indicates that the producer supplied AA document is requested if the specified AA instance is created by Update operation, otherwise, if it is created by Create operation, the GetArchiveFailedFault will be returned. "false" indicates that the complete AA instance generated by the Update operation over the base AA instance. "false" is assumed by default.

/ari:GetArchive/ari:TransportType
This REQUIRED element indicates the transport type of the AA document. The content of this element is a transport type URI. The AA document MUST be returned in the transport type specified in this parameter.

/ari:GetArchive/ari:TransportMethod
This REQUIRED element indicates the transport method of the AA document. The content of

this element is a transport method URI. The AA document MUST be returned in the transport method specified in this parameter.

5.2.4.2 Response message

The format of the GetArchive response message is shown below:

```
<ari:GetArchiveResponse>
  <ari:AA ...>...</ari:AA>
</ari:GetArchiveResponse>
```

The contents of the GetArchiveResponse message are described below:

/ari:GetArchiveResponse/ari:AA
This REQUIRED element indicates the AA document. The format of this element is as same as that in Section 5.1.2.1.

5.2.4.3 Faults

The types of faults for the GetArchive operation are as follows:

ResourceUnknownFault
The WS-Resource (ApplicationArchive) identified in the message is not known to the Web service.

ResourceNotReadyFault
The WS-Resource (ApplicationArchive) identified in the message is not ready to be used.

ResourceUnavailableFault
The WS-Resource (ApplicationArchive) identified in the message is not available because the Create (Update) operation which instantiated the WS-Resource failed. The wsrf-bf:FaultCause element of this fault SHOULD describe the Creation(Update)FailedFault.

TransportTypeNotSupportedFault
The specified transport type is not supported by the AA instance.

TransportMethodNotSupportedFault
The specified transport method is not supported by the AA instance.

GetArchiveFailedFault
The operation failed due to internal reasons.

Implementations of ACS repository MAY augment other fault types that are implementation dependent. Requester SHOULD prepare for the implementation specific faults.

5.2.5 Other requirements for the ACS implementation

To be conformant to OGSA WSRF Basic Profile, the ApplicationArchive implementation MUST implement the following portTypes.

portType name	purpose
wsrp:GetResourceProperty	in order to enable to read the Resource Properties
wsrp:GetMultipleResourceProperties	in order to enable to read the Resource Properties
wsrl:ImmediateResourceTermination	in order to enable AA instance destruction

If the implementation of the ACS repository supports event notification, ApplicationArchive MUST implement the following portType.

portType name	purpose
wsnt:NotificationProducer	in order to enable to subscribe for AA instance destruction notifications

5.3 Notifications

Based on WS-Topics, the topic space of ACS notifications is defined as follows:

The name of the topic space is “ACSTopicSpace”.

The URI of the topic space is “`http://schemas.ggf.org/acs/2005/10/ari/topicspace`”.

There are two topics in the topic space:

ApplicationArchiveCreated with ApplicationArchiveCreatedMessageType
ApplicationArchiveUpdated with ApplicationArchiveUpdatedMessageType

Note that the notifications will be filtered with those that specified in the parameters of subscribe messages and subject to be limited by access policy in ACS repository.

5.3.1 ApplicationArchiveCreatedMessageType

This type of notification message is published when a new AA (i.e., AA of the beginning version) is created through `Create` operation. If the implementation of ApplicationRepository portType supports event notification, it SHOULD raise this type of notification message.

The format of notification messages of ApplicationArchiveCreatedMessageType is shown below:

```
<ari:DateTime>xsd:dateTime</ari:DateTime>
<ari:AAID>aaf:AAIDType</ari:AAID>
<ari:ArchiveEPR>
  wsa:EndpointReferenceType
</ari:ArchiveEPR>
```

The contents of notification messages of ApplicationArchiveCreatedMessageType are described below:

/ari:DateTime
The date and time when the AA was created.

/ari:AAID
The ID of the created AA.

/ari:ArchiveEPR
The endpoint reference of the created AA.

5.3.2 ApplicationArchiveUpdatedMessageType

If the implementation of ApplicationArchive portType supports event notification, it SHOULD raise the notification message of the type described here when newer version of AAs (that is, all the descendant AAs) are created as results of `Update` operations. This notification SHOULD be raised not only when the immediate child of the AA instance is created, but also when its descendants are created.

The format of notification messages of ApplicationArchiveUpdatedMessageType is shown below:

```
<ari:DateTime>xsd:dateTime</ari:DateTime>
<ari:AAIDNew>aaf:AAIDType</ari:AAIDNew>
<ari:AAIDOld>aaf:AAIDType</ari:AAIDOld>
<ari:ArchiveEPRNew>
  wsa:EndpointReferenceType
</ari:ArchiveEPRNew>
<ari:ArchiveEPROld>
  wsa:EndpointReferenceType
</ari:ArchiveEPROld>
<aaf:DifferentialAAD>aaf:DifferentialAADType</aaf:DifferentialAAD>
```

The contents of notification messages of ApplicationArchiveUpdatedMessageType are described beow:

/ari:DateTime
The date and time when the AA was updated.

/ari:AAIDNew
The ID of the updating AA.

/ari:AAIDOld
The ID of the updated AA (i.e., the AA of the previous version).

/ari:ArchiveEPRNew
The endpoint reference of the updating AA.

/ari:ArchiveEPROld
The endpoint reference of the updated AA.

/aaf:DifferentialAAD
Indicates the Differential AAD of updating AA instance from the base (updated) AA.

6. Application Archive Format Specification

6.1 Application Archive Descriptor

The Application Archive Descriptor (AAD) is an XML document describing meta-information of AA including a list of Application Contents, identification, access constraint information and so on. AAD is described by the application producer and sent to ACS as a part of AA document in `Create` operation so that ACS can verify consistency of the AA and provide functionality for accessing the contents in the AA.

All valid AAD MUST conform to the XML Schema defined in Section 9.1.

6.1.1 AAD structure

The aaf:AAD element is the root element of the AAD. This element contains the below elements that describe the AA: aaf:AAID, aaf:Author, aaf:Descriptions, aaf:AccessConstraint, aaf:Contents, ds:Signature and an extension point so that application producers can append any implementation-specific properties of the AA. ACS implementation need not understand the semantics of the extended properties. The details of each element are specified in the following sections. Since the AAD is exposed as a Resource Property of AA instance, ACS client can retrieve them using the interfaces defined in the WS-ResourceProperties specification. The pseudo schema definition follows:

```
<aaf:AAD>
  <aaf:AAID>aaf:AAIDType</aaf:AAID>
  <aaf:Author>aaf:AuthorType</aaf:Author>
  <aaf:Descriptions>aaf:DescriptionsType</aaf:Descriptions>?
  <aaf:AccessConstraint>
    aaf:AccessConstraintType
  </aaf:AccessConstraint>?
  <ds:Signature>ds:SignatureType</ds:Signature>?
  <aaf:Contents>aaf:ContentsType</aaf:Contents>
  {any}*
</aaf:AAD>
```

/aaf:AAD/aaf:AAID
: This REQUIRED element is a identification of the AA.

/aaf:AAD/aaf:Author
: This REQUIRED element provides human-readable information about the author of the AA.

/aaf:AAD/aaf:Descriptions
: This OPTIONAL element provides human-readable information about the AA.

/aaf:AAD/aaf:AccessConstraint
: This OPTIONAL element provides access control policy over the AA.

/aaf:AAD/ds:Signature
: This OPTIONAL element provides digital signature over AAD and Application Contents in AA by means of XML-Signature. For the details of this element, see Section 6.4.

/aaf:AAD/aaf:Contents
: This REQUIRED element contains the list of Application Contents in the AA.

/aaf:AAD/{any}
Provides an extension point for the implementation-specific information of the AA.

6.1.2 AAID element

This element specifies the identification of the AA which consists of name and version information. AAID MUST be unique in an ACS repository. The AAID has the following syntax:

```
<aaf:AAID>
  <aaf:Name>xsd:anyURI</aaf:Name>
  <aaf:Version>xsd:string</aaf:Version>
</aaf:AAID>
```

/aaf:AAID/aaf:Name
This REQUIRED element indicates URI which represents the AA name. This is for grouping the same applications with different versions.

/aaf:AAID/aaf:Version
This REQUIRED element indicates version string of the AA. This specification does not define the format of the version string.

6.1.3 Author element

This element specifies information about the AA author. This is descriptive, human-readable information. This element has the following syntax. An extension point for the implementation-specific information is provided.

```
<aaf:Author>
  <aaf:Name>xsd:string</aaf:Name>
  <aaf:Description xml:lang="xsd:language"?>
    xsd:string
  </aaf:Description>*
  <aaf:Location>
    <aaf:Country>xsd:string</aaf:Country>?
    <aaf:Address>xsd:string</aaf:Address>?
  </aaf:Location>?
  {any}*
</aaf:Author>
```

/aaf:Author/aaf:Name
This REQUIRED element indicates the name of the author.

/aaf:Author/aaf:Description
This OPTIONAL element indicates the description about the author.

/aaf:Author/aaf:Description/@xml:lang
This OPTIONAL attribute indicates the language of the description. “en” is the default value for this attribute.

/aaf:Author/aaf:Location
This OPTIONAL element indicates the address of the author.

/aaf:Author/aaf:Location/aaf:Country
This OPTIONAL element indicates the country of the author.

/aaf:Author/aaf:Location/aaf:Address
This OPTIONAL element indicates the address of the author.

6.1.4 Descriptions element

This element provides human-readable description about the AA. This element has the following syntax:

```
<aaf:Descriptions>
  <aaf:Description xml:lang="xsd:language"?>
    xsd:string
  </aaf:Description>*
</aaf:Descriptions>?
```

/aaf:Descriptions/aaf:Description
`aaf:Descriptions` element MUST contain zero or more `aaf:Description` element.

/aaf:Descriptions/aaf:Description/@xml:lang
This OPTIONAL attribute indicates the language of the description. “en” is the default value for this attribute.

6.1.5 AccessConstraint element

This element specifies policy for access control over the AA and its contents. The policy description MUST be described in the language specified by the `dialect` attribute. XACML is RECOMMENDED to be accepted by implementations of ACS repository as policy description language.

This element has the following syntax:

```
<aaf:AccessConstraint dialect="xsd:anyURI">
  {any}*
</aaf:AccessConstraint>?
```

/aaf:AccessConstraint/@dialect
This REQUIRED attribute indicates a URI that identifies the language of the access control policy description over the AA. There is a well-known dialect identified by this specification, corresponding to the XACML language.

- `urn:oasis:names:tc:xacml:1.0:policy`
  This URI identifies the XACML policy language.

/aaf:AccessConstraint/{any}
Access control policy description.

6.1.6 Contents element

The element contains list of Application Contents in the AA. All of the Application Content files MUST be listed in this element.

This element has the following syntax:

```
<aaf:Contents>
  <aaf:Content type="xsd:QName"? {@any}*>
    <aaf:Pathname>aaf:RelativePathnameType</aaf:Pathname>
    <ds:DigestMethod>ds:DigestMethodType</ds:DigestMethod>?
    <ds:DigestValue>ds:DigestValueType</ds:DigestValue>?
  </aaf:Content>*
</aaf:Contents>
```

/aaf:Contents/aaf:Content
`aaf:contents` element MUST contain zero or more `aaf:content` element.

/aaf:Contents/aaf:Content/@type
This OPTIONAL attribute indicates the type of Application Content. This attribute can be used as key information in the `GetContents` operation in ApplicationArchive portType. The following QName is reserved for the value of this attribute:

- `aaf:DeploymentDescriptor`
  The type of Application Contents which describe the configuration of the application.

/aaf:Contents/aaf:Content/@{any}
This provides extension points for the implementation-specific information of the Application Content. This attribute can be used as key information in the `GetContents` operation in ApplicationArchive portType.

/aaf:Contents/aaf:Content/aaf:Pathname
This REQUIRED element specifies the relative path of the Application Content relative to the root directory in the AA, which MUST NOT begin with "./". Note that this element restricts the physical format of AA if AA document is provided as bundle type in `Create/Update` operation. This element can be used as key information in the `GetContents` operation.

/aaf:Contents/aaf:Content/ds:DigestMethod
This OPTIONAL element specifies the digest algorithm to be applied to the Application Content.

/aaf:Contents/aaf:Content/ds:DigestValue
This OPTIONAL element specifies the Base64-encoded value of the digest of the Application Content.

6.2 Differential Application Archive Descriptor

Differential Application Archive Descriptor (Differential AAD) is an XML document describing differential information of AA from the base AA, which can be used as an input for `Update` operation only and cannot be used with `Create` operation. The name of the element to be used is `aaf:DifferentialAAD` and is typed `aaf:DifferentialAADType`. The difference from the standard AADType is:

- All the child element of the `aaf:DifferentialAAD`, except for `aaf:AAID` and `aaf:Contents`, are OPTIONAL. If the optional elements are given, they should replace those in the new AAD created for the updated AA.
- `aaf:AAID` element MUST have `aaf:BaserVersion` element as its child.
- Each `aaf:content` element MUST have a REQUIRED attribute `operation`.

The rest of the elements and attributes remain same as standard AAD and not described here. Valid Differential AAD MUST conform to the XML Schema defined in Chapter 9 .

6.2.1 AAID element

`aaf:AAID` element MUST have `aaf:BaserVersion` element as its child in addition to the standard element described in section 6.1.2. The AAID has the following syntax:

```
<aaf:AAID>
  <aaf:Name>xsd:anyURI</aaf:Name>
  <aaf:Version>xsd:string</aaf:Version>
  <aaf:BaseVersion>xsd:string</aaf:BaseVersion>
```

```
</aaf:AAID>
```

/aaf:AAID/aaf:BaseVersion
: This REQUIRED element indicates version string of the base AA (that is, the AA which this AA is based on). This specification does not define the format of the version string.

6.2.2 Contents and Content element

`aaf:Contents` element in the Differential AAD contains list of Application Contents to be modified in the AA. All `aaf:Content` elements MUST have `operation` attribute for the Differential AAD. These elements have the following syntax:

```
<aaf:Contents>
  <aaf:Content type="xsd:QName"?
               operation=("add" | "delete" | "replace") {@any}*>
    <aaf:Pathname>xsd:string</aaf:Pathname>
    <ds:DigestMethod>ds:DigestMethodType</ds:DigestMethod>?
    <ds:DigestValue>ds:DigestValueType</ds:DigestValue>?
  </aaf:Content>*
</aaf:Contents>
```

/aaf:Contents/aaf:Content
: `aaf:Contents` element MUST contain zero or more `aaf:Content` element. The corresponding `aaf:Content` element in the new AAD will be replaced with this `aaf:Content` element except for the `operation` attribute.

/aaf:Contents/aaf:Content/@operation
: This REQUIRED attribute specifies the operation on the Application Content which is needed to construct full AA from the base AA and this differential AA. The attribute value MUST be "`add`", "`delete`", or "`replace`".

6.3 Physical format

The representation of AA instances inside the ACS repository is implementation dependent and ACS doesn't specify it. If an AA is stored or transported outside of the ACS repository, a single file bundled with zip would be preferred and the corresponding URI value for the transportType is provided as described in ARI part of the ACS specification. The AAD should include the entire list of the contents in the AA accompanied with relative pathnames in the bundle or on the file system. The representation of an AA outside of the ACS repository is called an AA document.

AAD MUST be located in the root directory of AA and its filename MUST be "aad.xml". Similarly, differential AAD MUST be located in the root directory of differential AA with the filename "aad.xml".

6.4 Signature

AAD and Application Contents in an AA can be digitally-signed. It is RECOMMENDED that they be digitally-signed by means of XML-Signature. In the case, they MUST be digitally-signed in the following two steps:

1. For each Application Content which is to be digitally-signed, describe ds:DigestMethod and ds:DigestValue child elements of the corresponding aaf:Content element in AAD. The digest value MUST be calculated over the whole of each Application Content. Any canonicalization MUST NOT be performed on AAD or Application Contents in text format before digest calculation.

2. In AAD, describe ds:Signature child element in aaf:AAD (or aaf:DifferentialAAD) element which represents the enveloped signature over the AAD. Note that each Application Content above is digitally-signed indirectly via the digest information in the corresponding aaf:Content element.

For example, AAD of digitally-signed AA is as follows:

```
<aaf:AAD>
  ...
  <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    <ds:SignedInfo>
      <ds:CanonicalizationMethod Algorithm=
        "http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
      <ds:SignatureMethod Algorithm=
        "http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
      <ds:Reference URI="">
        <ds:Transforms>
          <ds:Transform Algorithm=
        "http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
        </ds:Transforms>
        <ds:DigestMethod Algorithm=
             "http://www.w3.org/2000/09/xmldsig#sha1"/>
        <ds:DigestValue>Njc4OTAxMjM...</ds:DigestValue>
      </ds:Reference>
    </ds:SignedInfo>
    <ds:SignatureValue>NTY3ODkwMTI...</ds:SignatureValue>
  </ds:Signature>
  <aaf:Contents>
    <aaf:Content type="aaf:DeploymentDescriptor">
      <aaf:Pathname>deploy/dd.xml</aaf:Pathname>
      <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <ds:DigestValue>MTIzNDU2Nzg...</ds:DigestValue>
    </aaf:Content>
    <aaf:Content type="bar:ApplicationBinary">
      <aaf:Pathname>app/foo.exe</aaf:Pathname>
      <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <ds:DigestValue>MjM0NTY3ODk...</ds:DigestValue>
    </aaf:Content>
  </aaf:Contents>
</aaf:AAD>
```

This specification does not specify digest, signature, canonicalization, or transform algorithms used in XML-Signature.

The signature of (differential) AA can be validated in the following two steps:

1. Validate the enveloped signature (ds:Signature element) in (differential) AAD.
2. Calculate the digest value of the Application Content in the digest method specified in ds:DigestMethod child element of the corresponding aaf:Content element. Check that the digest value coincides with that specified in ds:DigestValue child element.

Note that the process of digital-signature by means of XML-Signature described above is independent of whether AA document, in Create/Update request and GetArchive response messages, is transported in the form of a bundled archive or discrete files.

7. Security Considerations

A common understanding of security and an agreement on the security model is required in order for grid services to interoperate[1]. While WS-Security[2] architecture provides for a secure SOAP message exchange, complexity is increased when other required specifications are incorporated such as SAML and policy mechanisms of XACML[3].

ACS implementations MUST adhere to the OGSA™ Basic Security profiles which provide the security foundation. Since there can be multiple levels of security, each ACS implementations MUST also expose one or more OGSA Security conformance claims in order to advertise them. It is RECOMMENDED that ACS implementations adhere to the OGSA Secure Channel Security Profile. This helps clients determine the transport and security mechanisms provided by the ACS node implementation.

Several optional and extensible mechanisms for implementing nodes are described within this specification. The main areas of security policies and mechanisms are related to repository access control and integrity of data.

ACS permits the producer of an archive to specify an optional security policy document which is RECOMMEND to be an XACML compliant. An extensibility element in the AAD can be used to specify the policy.

In order to authorize or restrict access to an archive, an implementation of an ACS node MAY provide its own implementation of a SAML compliant Policy Enforcement Point (PEP). This would help to reduce the dependencies on external services and would allow an ACS node to accept or reject service requests to the repository. An ACS implementation could therefore have a generous policy that grants access to all interfaces and contents or strict policy which enforces access based on the optionally provided XACML document.

ACS also supports XML Digital Signatures on archive contents to enable inquirers to validate the integrity of the data with respect to the publisher.

[1] http://www.cs.virginia.edu/~humphrey/ogsa-sec-wg/OGSA-SecArch-v1-07192002.pdf
[2] http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf
[3] There are some efforts to simplify the process via the Web-Service Policy language (WSPL) specification.

8. Samples

8.1 Create

In this section, we consider a sample AA containing:

- AAD file named aad.xml in the root directory
- deployment descriptor file named dd.xml in the “deploy” subdirectory
- application binary files named foo.exe and foo.dll in the “app” subdirectory
- user data file named init.dat in the “data” subdirectory
- document file named ReadMe.txt in the “doc” subdirectory

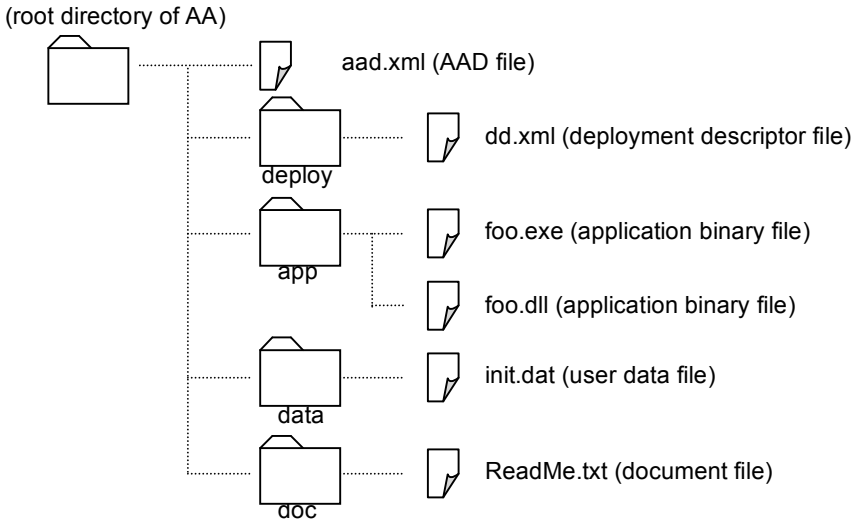

Figure 8-1 Structure of the sample AA

8.1.1 AAD File

The AAD file in the sample AA is shown below:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aaf:AAD xmlns:aaf="http://schemas.ggf.org/acs/2005/10/aaf"
         xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns:bar="http://www.example.ggf.org/">
  <aaf:AAID>
    <aaf:Name>http://www.foo.ggf.org/sample-application</aaf:Name>
    <aaf:Version>1.0.0</aaf:Version>
  </aaf:AAID>
  <aaf:Author>
    <aaf:Name>Foo Software Inc.</aaf:Name>
    <aaf:Description xml:lang="en">
    home page is http://www.foo.org/
    </aaf:Description>
    <aaf:Location>
      <aaf:Country>United States</aaf:Country>
    </aaf:Location>
```

```
  </aaf:Author>
  <aaf:Descriptions>
    <aaf:Description>
    sample app of foo application
    </aaf:Description>
  </aaf:Descriptions>
  <aaf:AccessConstraint
       dialect="urn:oasis:names:tc:xacml:1.0:policy">
    <xacml:Policy
      xmlns:xacml="urn:oasis:names:tc:xacml:1.0:policy">
      ...
    </xacml:Policy>
  </aaf:AccessConstraint>
  <ds:Signature>...</ds:Signature>
  <aaf:Contents>
    <aaf:Content type="aaf:DeploymentDescriptor">
      <aaf:Pathname>deploy/dd.xml</aaf:Pathname>
      <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <ds:DigestValue>MTIzNDU2Nzg...</ds:DigestValue>
    </aaf:Content>
    <aaf:Content type="bar:ApplicationBinary">
      <aaf:Pathname>app/foo.exe</aaf:Pathname>
      <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <ds:DigestValue>MjM0NTY3ODk...</ds:DigestValue>
    </aaf:Content>
    <aaf:Content type="bar:ApplicationBinary">
      <aaf:Pathname>app/foo.dll</aaf:Pathname>
      <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <ds:DigestValue>MzQ1Njc4OTA...</ds:DigestValue>
    </aaf:Content>
    <aaf:Content type="bar:UserData">
      <aaf:Pathname>data/init.dat</aaf:Pathname>
      <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <ds:DigestValue>NDU2Nzg5MDE...</ds:DigestValue>
    </aaf:Content>
    <aaf:Content type="bar:Document">
      <aaf:Pathname>doc/ReadMe.txt</aaf:Pathname>
      <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <ds:DigestValue>NTY3ODkwMTI...</ds:DigestValue>
    </aaf:Content>
  </aaf:Contents>
</aaf:AAD>
```

8.1.2 Create request message

In the Create request message, we can consider several combinations of transport type and method. The example Create messages described below have the following combinations:

	transport type	transport method
Example 1	bundled in zip	embedded
Example 2	bundled in zip	SwA
Example 3	discrete	embedded
Example 4	discrete	SwA
Example 5	bundled in zip	user-defined ftp

8.1.2.1 Example 1 (transport type: bundled, transport method: embedded)

The example Create request message, when the sample AA described in 8.1 is transported in the bundled type and the embedded method, is shown below:

```
<ari:Create>
  <ari:AA transportType=
   "http://schemas.ggf.org/acs/2005/10/ari/transport-type/bundled/zip">
    <ari:Bundle transportMethod=
    "http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded">
      <ari:Binary>ODkwMTIzNDU...</ari:Binary>
    </ari:Bundle>
  </ari:AA>
</ari:Create>
```

8.1.2.2 Example 2 (transport type: bundled, transport method: SwA)

The example Create request message, when the sample AA described in 8.1 is transported in the bundled type and the SwA method, is shown below:

```
MIME-Version: 1.0
Content-Type: Multipart/Related; boundary=MIME_boundary; type=text/xml;
        start="<soap-message@example.com>"
Content-Description: This is an example message description.

--MIME_boundary
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
Content-ID: <soap-message@example.com>

<?xml version="1.0" encoding="UTF-8" ?>
<SOAP-ENV:Envelope
          xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
..
<SOAP-ENV:Body xmlns:ari="http://schemas.ggf.org/acs/2005/10/ari"
               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xmlns:bar="http://www.example.com/"
>
<ari:Create>
  <ari:AA transportType=
   "http://schemas.ggf.org/acs/2005/10/ari/transport-type/bundled/zip">
    <ari:Bundle transportMethod=
     "http://schemas.ggf.org/acs/2005/10/ari/transport-method/SwA">
      <ari:AttachmentRef>cid:foo@example.com</ari:AttachmentRef>
    </ari:Bundle>
  </ari:AA>
```

```
</ari:Create>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

--MIME_boundary
Content-Type: binary/octet-stream
Content-Transfer-Encoding: binary
Content-ID: <foo@example.com>

(binary data)
--MIME_boundary--
```

8.1.2.3 Example 3 (transport type: discrete, transport method: embedded)

The example Create request message, when the sample AA described in 8.1 is transported in the discrete type and the embedded method, is shown below:

```
<ari:Create>
  <ari:AA transportType=
      "http://schemas.ggf.org/acs/2005/10/ari/transport-type/discrete">
    <ari:Descriptor transportMethod=
    "http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded">
      <ari:Binary>MTIzNDU2Nzg...</ari:Binary>
    </ari:Descriptor>
    <ari:Content transportMethod=
     "http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded"
            pathname="deploy/dd.xml">
      <ari:Binary>MjM0NTY3Odk...</ari:Binary>
    </ari:Content>
    <ari:Content transportMethod=
     "http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded"
            pathname="app/foo.exe">
      <ari:Binary>MzQ1Njc4OTA...</ari:Binary>
    </ari:Content>
    <ari:Content transportMethod=
     "http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded"
            pathname="app/foo.dll">
      <ari:Binary>NDU2Nzg5MDE...</ari:Binary>
    </ari:Content>
    <ari:Content transportMethod=
     "http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded"
            pathname="data/init.dat">
      <ari:Binary>NTY3ODkwMTI...</ari:Binary>
    </ari:Content>
    <ari:Content transportMethod=
     "http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded"
            pathname="doc/ReadMe.txt">
      <ari:Binary>Njc4OTAxMjM...</ari:Binary>
    </ari:Content>
  </ari:AA>
</ari:Create>
```

8.1.2.4 Example 4 (transport type: discrete, transport method: SwA)

The example Create request message, when the sample AA described in 8.1 is transported in the discrete type and the SwA method, is shown below:

```
MIME-Version: 1.0
Content-Type: Multipart/Related; boundary=MIME_boundary; type=text/xml;
        start="<soap-message@example.com>"
Content-Description: This is an example message description.

--MIME_boundary
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
Content-ID: <soap-message@example.com>

<?xml version="1.0" encoding="UTF-8" ?>
<SOAP-ENV:Envelope
          xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
..
<SOAP-ENV:Body xmlns:ari="http://schemas.ggf.org/acs/2005/10/ari"
               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xmlns:bar="http://www.example.com/"
>
<ari:Create>
  <ari:AA transportType=
      "http://schemas.ggf.org/acs/2005/10/ari/transport-type/discrete">
    <ari:Descriptor transportMethod=
         "http://schemas.ggf.org/acs/2005/10/ari/transport-method/SwA">
      <ari:AttachmentRef>cid:foo1@example.com</ari:AttachmentRef>
    </ari:Descriptor>
    <ari:Content transportMethod=
          "http://schemas.ggf.org/acs/2005/10/ari/transport-method/SwA"
            pathname="deploy/dd.xml">
      <ari:AttachmentRef>cid:foo2@example.com</ari:AttachmentRef>
    </ari:Content>
    <ari:Content transportMethod=
          "http://schemas.ggf.org/acs/2005/10/ari/transport-method/SwA"
            pathname="app/foo.exe">
      <ari:AttachmentRef>cid:foo3@example.com</ari:AttachmentRef>
    </ari:Content>
    <ari:Content transportMethod=
          "http://schemas.ggf.org/acs/2005/10/ari/transport-method/SwA"
            pathname="app/foo.dll">
      <ari:AttachmentRef>cid:foo4@example.com</ari:AttachmentRef>
    </ari:Content>
    <ari:Content transportMethod=
          "http://schemas.ggf.org/acs/2005/10/ari/transport-method/SwA"
            pathname="data/init.dat">
      <ari:AttachmentRef>cid:foo5@example.com</ari:AttachmentRef>
    </ari:Content>
    <ari:Content transportMethod=
          "http://schemas.ggf.org/acs/2005/10/ari/transport-method/SwA"
            pathname="doc/ReadMe.txt">
      <ari:AttachmentRef>cid:foo6@example.com</ari:AttachmentRef>
```

```
    </ari:Content>
  </ari:AA>
</ari:Create>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

--MIME_boundary
Content-Type: binary/octet-stream
Content-Transfer-Encoding: binary
Content-ID: <foo1@example.com>

(binary data)
--MIME_boundary--
Content-Type: binary/octet-stream
Content-Transfer-Encoding: binary
Content-ID: <foo2@example.com>

(binary data)
--MIME_boundary--
Content-Type: binary/octet-stream
Content-Transfer-Encoding: binary
Content-ID: <foo3@example.com>

(binary data)
--MIME_boundary--
Content-Type: binary/octet-stream
Content-Transfer-Encoding: binary
Content-ID: <foo4@example.com>

(binary data)
--MIME_boundary--
Content-Type: binary/octet-stream
Content-Transfer-Encoding: binary
Content-ID: <foo5@example.com>

(binary data)
--MIME_boundary--
Content-Type: binary/octet-stream
Content-Transfer-Encoding: binary
Content-ID: <foo6@example.com>

(binary data)
--MIME_boundary--
```

8.1.2.5 Example 5 (transport type: bundled, transport method: user-defined ftp)

The example Create request message, when the sample AA described in 8.1 is transported in the bundled type and the user-defined ftp method using the extensibility of the transport method, is shown below:

```
<ari:Create>
  <ari:AA transportType=
   "http://schemas.ggf.org/acs/2005/10/ari/transport-type/bundled/zip">
```

```
    <ari:Bundle transportMethod=
                "http://www.example.com/transport-method/ftp">
      <bar:URL>ftp://myserver.com/sample.aa</bar:URL>
      <bar:username>john</bar:username>
      <bar:password>some-password</bar:password>
    </ari:Bundle>
  </ari:AA>
</ari:Create>
```

8.2 Update

In this section, we consider a sample differential AA containing:

- differential AAD file named diffAad.xml in the root directory
- revised version of deployment descriptor file named dd.xml in the “deploy” subdirectory
- additional application binary file named bar.exe in the “app” subdirectory
- revised version of document file named ReadMe.txt in the “doc” subdirectory

We assume that the user data file named init.dat is not revised and that the application binary file named foo.dll is unnecessary and to be deleted in the revised version of AA.

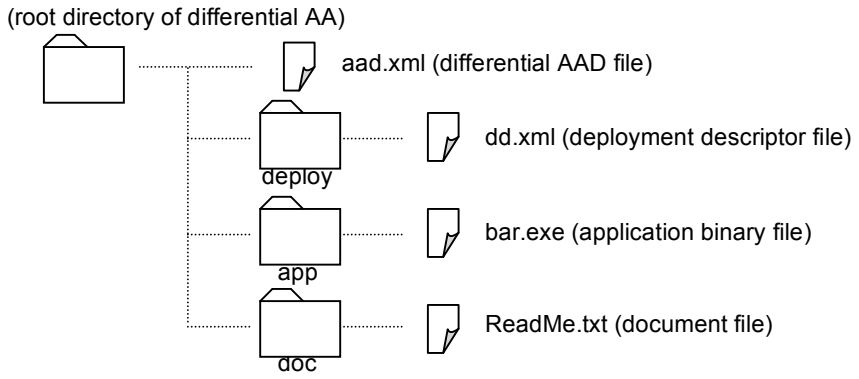

Figure 8-2 Structure of the sample differential AA

8.2.1 Differential AAD File

The Differential AAD file in the sample differential AA is shown below:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aaf:DifferentialAAD
     xmlns:aaf="http://schemas.ggf.org/acs/2005/10/aaf"
     xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xmlns:bar="http://www.example.com/">
  <aaf:AAID>
    <aaf:Name>http://www.foo.org/sample-application</aaf:Name>
    <aaf:Version>1.0.1</aaf:Version>
    <aaf:BaseVersion>1.0.0</aaf:BaseVersion>
  </aaf:AAID>
  <ds:Signature>...</ds:Signature>
  <aaf:Contents>
    <aaf:Content type="aaf:DeploymentDescriptor" operation="replace">
```

```
      <aaf:Pathname>deploy/dd.xml</aaf:Pathname>
      <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <ds:DigestValue>MjM0NTY3ODk...</ds:DigestValue>
    </aaf:Content>
    <aaf:Content type="bar:ApplicationBinary" operation="add">
      <aaf:Pathname>app/bar.exe</aaf:Pathname>
      <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <ds:DigestValue>MzQ1Njc4OTA...</ds:DigestValue>
    </aaf:Content>
    <aaf:Content type="bar:ApplicationBinary" operation="delete">
      <aaf:Pathname>app/foo.dll</aaf:Pathname>
    </aaf:Content>
    <aaf:Content type="bar:Document" operation="replace">
      <aaf:Pathname>doc/ReadMe.txt</aaf:Pathname>
      <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <ds:DigestValue>NDU2Nzg5MDE...</ds:DigestValue>
    </aaf:Content>
  </aaf:Contents>
</aaf:DifferentialAAD>
```

8.2.2 Update request message

In the Update request message, we can consider several combinations of transport type and method. For example, the Update request message, when the sample differential AA described in 8.2 is transported in the discrete type and the embedded method, is shown below:

```
<ari:Update>
  <ari:AA transportType=
      "http://schemas.ggf.org/acs/2005/10/ari/transport-type/discrete">
    <ari:Descriptor transportMethod=
    "http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded">
      <ari:Binary>MTIzNDU2Nzg...</ari:Binary>
    </ari:Descriptor>
    <ari:Content transportMethod=
     "http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded"
            pathname="deploy/dd.xml">
      <ari:Binary>MjM0NTY3Odk...</ari:Binary>
    </ari:Content>
    <ari:Content transportMethod=
     "http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded"
            pathname="app/bar.exe">
      <ari:Binary>MzQ1Njc4OTA...</ari:Binary>
    </ari:Content>
    <ari:Content transportMethod=
     "http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded"
            pathname="doc/ReadMe.txt">
      <ari:Binary>Njc4OTAxMjM...</ari:Binary>
    </ari:Content>
  </ari:AA>
</ari:Update>
```

Note that ari:Content element is not described for Application Content which is to be deleted in the updating AA.

8.3 GetContents

8.3.1 GetContents request message

The example GetContents request message is shown below:

```
<ari:GetContents>
  <ari:QueryExpression
       dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
     aad:content[@type="bar:ApplicationBinary"]
  </ari:QueryExpression>
  <ari:TransportMethod>
    http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded
  </ari:TransportMethod>
</ari:GetContents>
```

In the example above, Application Contents whose type is “bar:ApplicationBinary” are requested.

8.3.2 GetContents response message

The GetContents response message is shown below:

```
<ari:GetContentsResponse>
  <ari:Content transportMethod=
     "http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded"
          pathname="app/foo.exe">
    <ari:Binary>MzQ1Njc4OTA...</ari:Binary>
  </ari:Content>
  <ari:Content transportMethod=
     "http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded"
          pathname="app/foo.dll">
    <ari:Binary>NDU2Nzg5MDE...</ari:Binary>
  </ari:Content>
</ari:GetContentsResponse>
```

8.4 GetArchive

8.4.1 GetArchive request message

The example GetArchive request message is shown below:

```
<ari:GetArchive>
  <ari:Differential>false</ari:Differential>
  <ari:TransportType>
    http://schemas.ggf.org/acs/2005/10/ari/transport-type/bundled/zip
  </ari:TransportType>
  <ari:TransportMethod>
    http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded
  </ari:TransportMethod>
</ari:GetArchive>
```

8.4.2 GetArchive response message

The GetArchive response message is shown below:

```
<ari:GetArchiveResponse>
  <ari:AA transportType=
   "http://schemas.ggf.org/acs/2005/10/ari/transport-type/bundled/zip">
    <ari:Bundle transportMethod=
    "http://schemas.ggf.org/acs/2005/10/ari/transport-method/embedded">
      <ari:Binary>ODkwMTIzNDU...</ari:Binary>
    </ari:Bundle>
  </ari:AA>
</ari:GetArchiveResponse>
```

9. Schema Definition

This chapter describes schema definitions for:

- Application Archive Descriptor (AAD.xsd)
- Message schema for Application Repository Interface (ARI.xsd)
- Application Repository Interface (ARI.wsdl)

9.1 Application Archive Descriptor (AAD.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:aaf="http://schemas.ggf.org/acs/2005/10/aaf"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            targetNamespace="http://schemas.ggf.org/acs/2005/10/aaf"
            elementFormDefault="qualified"
            attributeFormDefault="unqualified">

  <xsd:import namespace="http://www.w3.org/XML/1998/namespace"
              schemaLocation="http://www.w3.org/2001/xml.xsd"/>
  <xsd:import namespace="http://www.w3.org/2000/09/xmldsig#"
              schemaLocation=
          "http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd"/>

  <xsd:element name="AAD" type="aaf:AADType">
    <xsd:annotation>
      <xsd:documentation>The root element of AAD.</xsd:documentation>
    </xsd:annotation>
  </xsd:element>
  <xsd:element name="DifferentialAAD" type="aaf:DiffAADType">
    <xsd:annotation>
      <xsd:documentation>
        The root element of differential AAD.
      </xsd:documentation>
    </xsd:annotation>
  </xsd:element>
  <xsd:complexType name="AADType">
    <xsd:sequence>
      <xsd:element name="AAID" type="aaf:AAIDType"/>
      <xsd:element name="Author" type="aaf:AuthorType"/>
      <xsd:element name="Descriptions" type="aaf:DescriptionsType"
                minOccurs="0"/>
      <xsd:element name="AccessConstraint"
                type="aaf:AccessConstraintType" minOccurs="0"/>
      <xsd:element ref="ds:Signature" minOccurs="0"/>
      <xsd:element name="Contents" type="aaf:ContentsType"/>
      <xsd:any namespace="##other" processContents="lax"
                minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="DiffAADType">
```

```
    <xsd:sequence>
      <xsd:element name="AAID" type="aaf:DiffAAIDType"/>
      <xsd:element name="Author" type="aaf:AuthorType" minOccurs="0"/>
      <xsd:element name="Descriptions" type="aaf:DescriptionsType"
                minOccurs="0"/>
      <xsd:element name="AccessConstraint"
                type="aaf:AccessConstraintType" minOccurs="0"/>
      <xsd:element ref="ds:Signature" minOccurs="0"/>
      <xsd:element name="Contents" type="aaf:DiffContentsType"/>
      <xsd:any namespace="##other" processContents="lax"
                minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="AAIDType">
    <xsd:sequence>
      <xsd:element name="Name" type="xsd:anyURI"/>
      <xsd:element name="Version" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="AuthorType">
    <xsd:sequence>
      <xsd:element name="Name" type="xsd:string"/>
      <xsd:element name="Description" type="aaf:DescriptionType"
                minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element name="Location" minOccurs="0">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="Country" type="xsd:string"
              minOccurs="0"/>
            <xsd:element name="Address" type="xsd:string"
              minOccurs="0"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:any namespace="##other" processContents="lax"
                minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="DescriptionType">
    <xsd:simpleContent>
      <xsd:extension base="xsd:string">
        <xsd:attribute ref="xml:lang" use="optional" default="en"/>
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>
  <xsd:complexType name="DescriptionsType">
    <xsd:sequence>
      <xsd:element name="Description" type="aaf:DescriptionType"
                minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="AccessConstraintType" mixed="true">
    <xsd:sequence>
```

```
      <xsd:any namespace="##other" processContents="lax"
               minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="dialect" type="xsd:anyURI" use="required"/>
  </xsd:complexType>
  <xsd:group name="DigestInfoGroup">
    <xsd:sequence>
      <xsd:element ref="ds:DigestMethod"/>
      <xsd:element ref="ds:DigestValue"/>
    </xsd:sequence>
  </xsd:group>
  <xsd:simpleType name="RelativePathnameType">
    <xsd:restriction base="xsd:string">
      <xsd:pattern value=
        "([^\s./]+([\s]+[^\s/]+)*)(/([^\s/]+([\s]+[^\s/]+)*))*"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:complexType name="ContentType">
    <xsd:sequence>
      <xsd:element name="Pathname" type="aaf:RelativePathnameType"/>
      <xsd:group ref="aaf:DigestInfoGroup" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="type" type="xsd:QName" use="optional"/>
    <xsd:anyAttribute namespace="##any" processContents="lax"/>
  </xsd:complexType>
  <xsd:complexType name="ContentsType">
    <xsd:sequence>
      <xsd:element name="Content" type="aaf:ContentType"
               minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="DiffAAIDType">
    <xsd:complexContent>
      <xsd:extension base="aaf:AAIDType">
        <xsd:sequence>
          <xsd:element name="BaseVersion" type="xsd:string"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:simpleType name="OperationType">
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="add"/>
      <xsd:pattern value="delete"/>
      <xsd:pattern value="replace"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:complexType name="DiffContentType">
    <xsd:complexContent>
      <xsd:extension base="aaf:ContentType">
        <xsd:attribute name="operation" type="aaf:OperationType"
               use="required"/>
      </xsd:extension>
```

```
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="DiffContentsType">
    <xsd:sequence>
      <xsd:element name="Content" type="aaf:DiffContentType"
                minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

9.2 Message schema for Application Repository Interface (ARI.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:ari="http://schemas.ggf.org/acs/2005/10/ari"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:aaf="http://schemas.ggf.org/acs/2005/10/aaf"
            xmlns:wsa="http://www.w3.org/2005/03/addressing"
            xmlns:wsrf-bf="http://docs.oasis-open.org/wsrf/bf-1"
            xmlns:ref="http://ws-i.org/profiles/basic/1.1/xsd"
            targetNamespace="http://schemas.ggf.org/acs/2005/10/ari"
            elementFormDefault="qualified"
            attributeFormDefault="unqualified">

  <xsd:import namespace="http://schemas.ggf.org/acs/2005/10/aaf"
              schemaLocation="./AAD.xsd"/>
  <xsd:import namespace="http://www.w3.org/2005/03/addressing"
              schemaLocation="http://www.w3.org/2005/03/addressing/"/>
  <xsd:import namespace="http://docs.oasis-open.org/wsrf/bf-1"
              schemaLocation="http://docs.oasis-open.org/wsrf/bf-1"/>
  <xsd:import namespace="http://ws-i.org/profiles/basic/1.1/xsd"
              schemaLocation="../other/swaref.xsd"/>
  <!-- The schema "swaref.xsd" can be retrieved from:
       http://www.ws-i.org/Profiles/AttachmentsProfile-1.0-2004-08-
24.html#Referencing_Attachments_from_the_SOAP_Envelope -->

  <xsd:complexType name="AAType">
    <xsd:choice>
      <xsd:element name="Bundle">
        <xsd:complexType>
          <xsd:complexContent>
            <xsd:extension base="ari:TransportedDataType"/>
          </xsd:complexContent>
        </xsd:complexType>
      </xsd:element>
      <xsd:sequence>
        <xsd:element name="Descriptor">
          <xsd:complexType>
            <xsd:complexContent>
              <xsd:extension base="ari:TransportedDataType"/>
            </xsd:complexContent>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="Content"
```

```
                    minOccurs="0" maxOccurs="unbounded">
          <xsd:complexType>
            <xsd:complexContent>
              <xsd:extension base="ari:TransportedDataType">
               <xsd:attribute name="pathname"
                     type="aaf:RelativePathnameType" use="required"/>
              </xsd:extension>
            </xsd:complexContent>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
   </xsd:choice>
   <xsd:attribute name="transportType" type="xsd:anyURI"
                  use="required"/>
</xsd:complexType>
<xsd:complexType name="TransportedDataType">
   <xsd:choice>
     <xsd:element name="Binary" type="xsd:base64Binary"/>
     <xsd:element name="AttachmentRef" type="ref:swaRef"/>
     <xsd:any namespace="##other" processContents="lax"
              minOccurs="0" maxOccurs="unbounded"/>
   </xsd:choice>
   <xsd:attribute name="transportMethod" type="xsd:anyURI"
              use="required"/>
</xsd:complexType>
<xsd:complexType name="QueryExpressionType" mixed="true">
   <xsd:sequence>
     <xsd:any namespace="##other" processContents="lax"
              minOccurs="0"/>
   </xsd:sequence>
   <xsd:attribute name="dialect" type="xsd:anyURI"/>
</xsd:complexType>
<xsd:element name="RepositoryProperties">
   <xsd:complexType>
     <xsd:sequence>
       <xsd:element name="Version" type="xsd:anyURI"/>
       <xsd:element name="TransportType" type="xsd:anyURI"
                    maxOccurs="unbounded"/>
       <xsd:element name="TransportMethod" type="xsd:anyURI"
                    maxOccurs="unbounded"/>
       <xsd:element name="QueryExpressionDialect" type="xsd:anyURI"
                    maxOccurs="unbounded"/>
     </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<xsd:element name="ArchiveProperties">
   <xsd:complexType>
     <xsd:sequence>
       <xsd:element ref="aaf:AAD"/>
       <xsd:element ref="aaf:DifferentialAAD" minOccurs="0"/>
       <xsd:element name="CreationDateTime" type="xsd:dateTime"/>
       <xsd:element name="BaseAA" type="wsa:EndpointReferenceType"
                    minOccurs="0"/>
```

```
        <xsd:element name="NewerAA" type="wsa:EndpointReferenceType"
                     minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="Repository"
                     type="wsa:EndpointReferenceType"/>
        <xsd:element name="QueryExpressionDialect" type="xsd:anyURI"
                     maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="Create">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="AA" type="ari:AAType"/>
        <xsd:any namespace="##other" processContents="lax"
                     minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="CreateResponse">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="ArchiveEPR"
                     type="wsa:EndpointReferenceType"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="LookupArchives">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="QueryExpression"
                     type="ari:QueryExpressionType"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="LookupArchivesResponse">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="AAInfo"
                     minOccurs="0" maxOccurs="unbounded">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="EPR"
                           type="wsa:EndpointReferenceType"/>
              <xsd:any namespace="##any" processContents="lax"
                           minOccurs="0" maxOccurs="unbounded"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="Update">
    <xsd:complexType>
```

```
      <xsd:sequence>
        <xsd:element name="AA" type="ari:AAType"/>
        <xsd:any namespace="##other" processContents="lax"
                 minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="UpdateResponse">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="ArchiveEPR"
                     type="wsa:EndpointReferenceType"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="GetContents">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="QueryExpression"
                     type="ari:QueryExpressionType"/>
        <xsd:element name="TransportMethod" type="xsd:anyURI"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="GetContentsResponse">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="Content"
                     minOccurs="0" maxOccurs="unbounded">
          <xsd:complexType>
            <xsd:complexContent>
              <xsd:extension base="ari:TransportedDataType">
               <xsd:attribute name="pathname"
                     type="aaf:RelativePathnameType" use="required"/>
              </xsd:extension>
            </xsd:complexContent>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="GetArchive">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="Differential" type="xsd:boolean"/>
        <xsd:element name="TransportType" type="xsd:anyURI"/>
        <xsd:element name="TransportMethod" type="xsd:anyURI"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="GetArchiveResponse">
    <xsd:complexType>
      <xsd:sequence>
```

```
        <xsd:element name="AA" type="ari:AAType"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="IllegalDescriptorFault">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="wsrf-bf:BaseFaultType"/>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="TransportTypeNotSupportedFault">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="wsrf-bf:BaseFaultType"/>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="TransportMethodNotSupportedFault">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="wsrf-bf:BaseFaultType"/>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="CreationFailedFault">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="wsrf-bf:BaseFaultType"/>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="LookupFailedFault">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="wsrf-bf:BaseFaultType"/>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="ResourceNotReadyFault">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="wsrf-bf:BaseFaultType"/>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="ResourceUnavailableFault">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="wsrf-bf:BaseFaultType"/>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
```

```
  <xsd:element name="UpdateFailedFault">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="wsrf-bf:BaseFaultType"/>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="GetArchiveFailedFault">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="wsrf-bf:BaseFaultType"/>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="ApplicationArchiveCreatedMessage">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="DateTime" type="xsd:dateTime"/>
        <xsd:element name="AAID" type="aaf:AAIDType"/>
        <xsd:element name="ArchiveEPR"
                     type="wsa:EndpointReferenceType"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="ApplicationArchiveUpdatedMessage">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="DateTime" type="xsd:dateTime"/>
        <xsd:element name="AAIDNew" type="aaf:AAIDType"/>
        <xsd:element name="AAIDOld" type="aaf:AAIDType"/>
        <xsd:element name="ArchiveEPRNew"
                     type="wsa:EndpointReferenceType"/>
        <xsd:element name="ArchiveEPROld"
                     type="wsa:EndpointReferenceType"/>
        <xsd:element ref="aaf:DifferentialAAD"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

9.3 Application Repository Interface (ARI.wsdl)

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="ARIportTypes"
                  xmlns:ari="http://schemas.ggf.org/acs/2005/10/ari"
                  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
                  xmlns:wsrf-rp="http://docs.oasis-open.org/wsrf/rp-1"
                  xmlns:wsrf-r="http://docs.oasis-open.org/wsrf/r-1"
                targetNamespace="http://schemas.ggf.org/acs/2005/10/ari">
  <!-- ============================================================ -->
  <wsdl:documentation>
    This is the WSDL Describing the ApplicationRepository and
ApplicationArchive portTypes.
```

```
  </wsdl:documentation>

  <!-- ========================================================== -->
  <!-- importing WSDLs -->
  <!-- ========================================================== -->
  <wsdl:import namespace="http://docs.oasis-open.org/wsrf/rp-1"
               location="http://docs.oasis-open.org/wsrf/2005/03/wsrf-
WS-ResourceProperties-1.2-draft-06.wsdl"/>
  <wsdl:import namespace="http://docs.oasis-open.org/wsrf/r-1"
               location="http://docs.oasis-open.org/wsrf/2005/03/wsrf-
WS-Resource-1.2-draft-03.wsdl"/>

  <!-- ========================================================== -->
  <!-- types for ARI portTypes -->
  <!-- ========================================================== -->
  <wsdl:types>
    <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <xsd:import namespace="http://schemas.ggf.org/acs/2005/10/ari"
                  schemaLocation="./ARI.xsd" />
     </xsd:schema>
  </wsdl:types>

  <!-- ========================================================== -->
  <!-- messages for ApplicationRepository portType -->
  <!-- ========================================================== -->
  <wsdl:message name="CreateRequest">
    <wsdl:part element="ari:Create" name="CreateRequest"/>
  </wsdl:message>
  <wsdl:message name="CreateResponse">
    <wsdl:part element="ari:CreateResponse" name="CreateResponse"/>
  </wsdl:message>

  <wsdl:message name="LookupArchivesRequest">
    <wsdl:part element="ari:LookupArchives"
               name="LookupArchivesRequest"/>
  </wsdl:message>
  <wsdl:message name="LookupArchivesResponse">
    <wsdl:part element="ari:LookupArchivesResponse"
               name="LookupArchivesResponse"/>
  </wsdl:message>

  <!-- ========================================================== -->
  <!-- messages for ApplicationArchive portType -->
  <!-- ========================================================== -->
  <wsdl:message name="UpdateRequest">
    <wsdl:part element="ari:Update" name="UpdateRequest"/>
  </wsdl:message>
  <wsdl:message name="UpdateResponse">
    <wsdl:part element="ari:UpdateResponse" name="UpdateResponse"/>
  </wsdl:message>

  <wsdl:message name="GetContentsRequest">
    <wsdl:part element="ari:GetContents" name="GetContentsRequest"/>
```

```
  </wsdl:message>
  <wsdl:message name="GetContentsResponse">
    <wsdl:part element="ari:GetContentsResponse"
               name="GetContentsResponse"/>
  </wsdl:message>

  <wsdl:message name="GetArchiveRequest">
    <wsdl:part element="ari:GetArchive" name="GetArchiveRequest"/>
  </wsdl:message>
  <wsdl:message name="GetArchiveResponse">
    <wsdl:part element="ari:GetArchiveResponse"
               name="GetArchiveResponse"/>
  </wsdl:message>

  <!-- ============================================================ -->
  <!-- Fault messages -->
  <!-- ============================================================ -->
  <wsdl:message name="ResourceNotReadyFault">
    <wsdl:part element="ari:ResourceNotReadyFault"
               name="ResourceNotReadyFault"/>
  </wsdl:message>
  <wsdl:message name="ResourceUnavailableFault">
    <wsdl:part element="ari:ResourceUnavailableFault"
               name="ResourceUnavailableFault"/>
  </wsdl:message>
  <wsdl:message name="TransportMethodNotSupportedFault">
    <wsdl:part element="ari:TransportMethodNotSupportedFault"
               name="TransportMethodNotSupportedFault"/>
  </wsdl:message>
  <wsdl:message name="TransportTypeNotSupportedFault">
    <wsdl:part element="ari:TransportTypeNotSupportedFault"
               name="TransportTypeNotSupportedFault"/>
  </wsdl:message>
  <wsdl:message name="IllegalDescriptorFault">
    <wsdl:part element="ari:IllegalDescriptorFault"
               name="IllegalDescriptorFault"/>
  </wsdl:message>
  <wsdl:message name="CreationFailedFault">
    <wsdl:part element="ari:CreationFailedFault"
               name="CreationFailedFault"/>
  </wsdl:message>
  <wsdl:message name="LookupFailedFault">
    <wsdl:part element="ari:LookupFailedFault"
               name="LookupFailedFault"/>
  </wsdl:message>
  <wsdl:message name="UpdateFailedFault">
    <wsdl:part element="ari:UpdateFailedFault"
               name="UpdateFailedFault"/>
  </wsdl:message>
  <wsdl:message name="GetArchiveFailedFault">
    <wsdl:part element="ari:GetArchiveFailedFault"
               name="GetArchiveFailedFault"/>
  </wsdl:message>
```

```
<!-- ============================================================ -->
<!-- ApplicationRepository portType definition -->
<!-- ============================================================ -->
<wsdl:portType name="ApplicationRepository"
               wsrf-rp:ResourceProperties="ari:RepositoryProperties">

  <wsdl:operation name="Create">
    <wsdl:input name="CreateRequest" message="ari:CreateRequest"/>
    <wsdl:output name="CreateResponse" message="ari:CreateResponse"/>
    <wsdl:fault name="ResourceUnknownFault"
                message="wsrf-r:ResourceUnknownFault"/>
    <wsdl:fault name="IllegalDescriptorFault"
                message="ari:IllegalDescriptorFault"/>
    <wsdl:fault name="TransportTypeNotSupportedFault"
                message="ari:TransportTypeNotSupportedFault"/>
    <wsdl:fault name="TransportMethodNotSupportedFault"
                message="ari:TransportMethodNotSupportedFault"/>
    <wsdl:fault name="CreationFailedFault"
                message="ari:CreationFailedFault"/>
  </wsdl:operation>

  <wsdl:operation name="LookupArchives">
    <wsdl:input name="LookupArchivesRequest"
                message="ari:LookupArchivesRequest"/>
    <wsdl:output name="LookupArchivesResponse"
                message="ari:LookupArchivesResponse"/>
    <wsdl:fault name="ResourceUnknownFault"
                message="wsrf-r:ResourceUnknownFault"/>
    <wsdl:fault name="UnknownQueryExpressionDialectFault"
               message="wsrf-rp:UnknownQueryExpressionDialectFault"/>
    <wsdl:fault name="InvalidQueryExpressionFault"
                message="wsrf-rp:InvalidQueryExpressionFault"/>
    <wsdl:fault name="LookupFailedFault"
                message="ari:LookupFailedFault"/>
  </wsdl:operation>

</wsdl:portType>

<!-- ============================================================ -->
<!-- ApplicationArchive portType definition -->
<!-- ============================================================ -->
<wsdl:portType name="ApplicationArchive"
               wsrf-rp:ResourceProperties="ari:ArchiveProperties">

  <wsdl:operation name="Update">
    <wsdl:input name="UpdateRequest" message="ari:UpdateRequest"/>
    <wsdl:output name="UpdateResponse" message="ari:UpdateResponse"/>
    <wsdl:fault name="ResourceUnknownFault"
                message="wsrf-r:ResourceUnknownFault"/>
    <wsdl:fault name="ResourceNotReadyFault"
                message="ari:ResourceNotReadyFault"/>
    <wsdl:fault name="ResourceUnavailableFault"
```

```
                    message="ari:ResourceUnavailableFault"/>
      <wsdl:fault name="IllegalDescriptorFault"
                    message="ari:IllegalDescriptorFault"/>
      <wsdl:fault name="TransportTypeNotSupportedFault"
                    message="ari:TransportTypeNotSupportedFault"/>
      <wsdl:fault name="TransportMethodNotSupportedFault"
                    message="ari:TransportMethodNotSupportedFault"/>
      <wsdl:fault name="UpdateFailedFault"
                    message="ari:UpdateFailedFault"/>
    </wsdl:operation>

    <wsdl:operation name="GetContents">
      <wsdl:input name="GetContentsRequest"
                    message="ari:GetContentsRequest"/>
      <wsdl:output name="GetContentsResponse"
                    message="ari:GetContentsResponse"/>
      <wsdl:fault name="ResourceUnknownFault"
                    message="wsrf-r:ResourceUnknownFault"/>
      <wsdl:fault name="ResourceNotReadyFault"
                    message="ari:ResourceNotReadyFault"/>
      <wsdl:fault name="ResourceUnavailableFault"
                    message="ari:ResourceUnavailableFault"/>
      <wsdl:fault name="UnknownQueryExpressionDialectFault"
                   message="wsrf-rp:UnknownQueryExpressionDialectFault"/>
      <wsdl:fault name="InvalidQueryExpressionFault"
                    message="wsrf-rp:InvalidQueryExpressionFault"/>
      <wsdl:fault name="TransportMethodNotSupportedFault"
                    message="ari:TransportMethodNotSupportedFault"/>
    </wsdl:operation>

    <wsdl:operation name="GetArchive">
      <wsdl:input name="GetArchiveRequest"
                    message="ari:GetArchiveRequest"/>
      <wsdl:output name="GetArchiveResponse"
                    message="ari:GetArchiveResponse"/>
      <wsdl:fault name="ResourceUnknownFault"
                    message="wsrf-r:ResourceUnknownFault"/>
      <wsdl:fault name="ResourceNotReadyFault"
                    message="ari:ResourceNotReadyFault"/>
      <wsdl:fault name="ResourceUnavailableFault"
                    message="ari:ResourceUnavailableFault"/>
      <wsdl:fault name="TransportTypeNotSupportedFault"
                    message="ari:TransportTypeNotSupportedFault"/>
      <wsdl:fault name="TransportMethodNotSupportedFault"
                    message="ari:TransportMethodNotSupportedFault"/>
      <wsdl:fault name="GetArchiveFailedFault"
                    message="ari:GetArchiveFailedFault"/>
    </wsdl:operation>

  </wsdl:portType>

</wsdl:definitions>
```

10. Open Issues

The below lists the open issues that needed more study in various aspects and use cases, which are among the items to be addressed in the future version of the specification. The description in this Chapter is non-normative.

- The relationship to the outcomes from the on-going OASIS SDD TC activities should be discussed once after the XML Schema for their deployment descriptor is published. Interoperability need to be considered.
- More extensive or detailed Resource Properties for ApplicationRepository and ApplicationArchive may be explored and be a part of the specification.
- For the retrieval of the version history, the ACS repository may store an entire tree of the sequence may be maintained and provided for the users, while this version only maintain the links to the parent and children.
- More features that assist the replication of AAs may be explored in accordance with the on-going discussion in the other standard activities.

Author Information

The below are the names of persons who authored this document:

Keisuke Fukui (Editor)
Fujitsu Laboratories Ltd.
Email: kfukui@labs.fujitsu.com

Thomas Studwell
IBM
Email: studwell@us.ibm.com

Peter Ziu
Northrop Grumman
Email: peter.ziu@ngc.com

Michael Behrens
R2AD, LLC
Email: behrens@r2ad.com

Sachiko Wada
ASCADE, Inc.
Email: sachiko@ascade.co.jp

Yuichiro Yonebayashi
ASCADE, Inc.
Email: yonebayashi@ascade.co.jp

Acknowledgements

We are grateful to numerous colleagues for discussions on the topics covered in this document. Thanks in particular to members in the CDDLM-WG; they joined our discussion for the interactions with CDDLM. Thanks to members in NAREGI in particular the PSE team for their joining the discussions and presenting valuable input for our design work. We also thank Stephen Pickles, Bill Nitzberg, Hiro Kishimoto, Dave Snelling, John Tollefsrud for their kind advice in operating our activity at GGF.

Glossary

Application Contents
Application Contents are files required for executing grid applications for a unit of a task in Grid systems. They include files such as the application binaries, configuration data and what are needed to create a unit of the task on grid systems. The task includes bare metal grid provisioning.

Application Archive (AA)
Application Archive (AA) is a logical bundle of Application Archive Descriptor (AAD), Application Contents, and their associated Application Deployment Descriptor. Contents in AA may contain pointers to the external storage; i.e. they are logically bundled as a unit.

AA Descriptor (AAD)
AA Descriptor (AAD) is an XML document, which describes properties of the AA contents. AAD may contain information such as identification, access constraint information, and internal data associated with the structure of the AA contents.

AA EPR (AAEPR)
AA EPR is an EndpointReference for an AA instance, which is an ApplicationArchive WS-Resource. Different versions for the same AA instance will have different EPRs.

AA ID (AAID)
AA ID is a element in the AAD, which identifies the AA. Two instances of AA with the same AAID are equivalent in its contents. Different versions for the same AA will have different AAIDs.

Application Contents Service (ACS)
Application Contents Service (ACS) is a set of specifications, which consists of ARI and AAF. One of the goals of ACS is to ease the submission and the following management of the application related files for a unit of a task processed in grid systems, which are needed to deploy and execute applications for a unit of a task.

Application Repository Interface (ARI)
ARI is the name of the specification, which defines interfaces to the Application Repository.

Application Archive Format (AAF)
AAF is the name of the specification, which defines a transport format of the AA.

ApplicationRepository WS-Resource
ApplicationRepository is a repository service for AAs. It is a WS-Resource with an ApplicationRepository portType. An ApplicationRepository is referred by an AR EPR.

ApplicationArchive WS-Resource
ApplicationArchive is an instance in the AR, which represents single AA. It is a WS-Resource with an ApplicationArchive portType. An ApplicationArchive is referred by an AA EPR.

Intellectual Property Statement

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

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

Full Copyright Notice

References

[OGSA doc] "The Open Grid Services Architecture", Version 1.0, I. Foster, H. Kishimoto, A. Savva, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, J. Von Reich,
http://www.ggf.org/documents/GFD.30.pdf

[OGSA glossary] "Open Grid Services Architecture: Glossary of Terms", Version 1.0, J.Treadwell,
http://www.ggf.org/documents/GFD.44.pdf

[OGSA usecase] "Open Grid Services Architecture Use Cases", March 4, 2004, I. Foster, D. Gannon, H. Kishimoto, Jeffrin J. Von Reich,
http://www.ggf.org/documents/GFD.29.pdf

[OGSA WSRF Basic Profile] I. Foster, T. Maguire and D. Snelling OGSA WSRF Basic Profile Version 1.0, Global Grid Forum OGSA-WG, GWD-R, 1 September 2005.
http://forge.gridforum.org/projects/ogsa-wg/document/draft-ggf-ogsa-wsrf-basic-profile/en/36

[WS-I BP 1.1] K. Ballinger, D. Ehnebuske, C. Ferris, M. Gudgin, C.K. Liu, M. Nottingham, and P. Yendluri (ed.) Basic Profile Version 1.1, Web Services Interoperability Organization Final Material, 24 August 2004.
http://www.ws-i.org/Profiles/BasicProfile-1.1.html

[BizGrid] “Business Grid Middleware Goals and Status”, January 20, 2004, Hiro Kishimoto, Takashi Kojo, Fred Maciel,
http://www.globusworld.org/program/slides/3c_1.pdf

[CDDLM-FND] "Configuration Description, Deployment, and Lifecycle Management (CDDLM) Foundation Document", 7 August 2005, D. Bell, T. Kojo, P. Goldsack, S. Loughran, D. Milojicic, S. Schaefer, J. Tatemura, and P. Toft,
http://www.ggf.org/documents/GFD.50.pdf

[CDDLM-SF] "Configuration Description, Deployment, and Lifecycle Management (CDDLM) SmartFrog-Based Language Specification", 9 September 2005, P. Goldsack,
http://www.ggf.org/documents/GFD.51.pdf

[CDDLM-CDL] "Configuration Description, Deployment, and Lifecycle Management (CDDLM) XML-Based Language", Document in preparation

[CDDLM-CMP] "Configuration Description, Deployment, and Lifecycle Management (CDDLM) Component Model", Version 1.0, 28 June 2005, Stuart Schaefer,
http://www.ggf.org/Public_Comment_Docs/Documents/Sep-2005/draft-ggf-cddlm-component-model.pdf

[CDDLM-API] "Configuration Description, Deployment, and Lifecycle Management (CDDLM) Deployment API", 13 September 2005, S. Loughran,
http://www.ggf.org/Public_Comment_Docs/Documents/Oct-2005/draft-ggf-cddlm-deploy-api2.pdf

[WS-ResourceProperties] S. Graham and J. Treadwell (ed.) Web Services Resource Properties 1.2 (WS-ResourceProperties), OASIS Public Review Draft 01, 10 June 2005.
http://docs.oasis-open.org/wsrf/wsrf-ws_resource_properties-1.2-spec-pr-01.pdf

[WS-ResourceLifetime] L. Srinivasan and T. Banks (ed.) Web Services Resource Lifetime 1.2 (WS-ResourceLifetime), OASIS Public Review Draft 01, 13 June 2005.
http://docs.oasis-open.org/wsrf/wsrf-ws_resource_lifetime-1.2-spec-pr-01.pdf

[WS-BaseFaults] S. Tuecke, L. Liu and S. Meder (ed.) Web Services Base Faults 1.2 (WS-BaseFaults), OASIS Public Review Draft 01. 13 June 2005.
http://docs.oasis-open.org/wsrf/wsrf-ws_base_faults-1.2-spec-pr-01.pdf

[WS-BaseNotification] S. Graham, D. Hull and B. Murray (ed.) Web Services Base Notification 1.3 (WS-BaseNotification), OASIS Public Review Draft 01, 07 July 2005.
http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-pr-01.pdf

[WS-Security] A. Nadalin, C. Kaler, P. Hallam-Baker and R. Monzillo (ed.) Web Services Security: SOAP Message Security 1.0 (WS-Security 2004), OASIS Standard, 200401, March 2004.
http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf

[SwA] "SOAP Messages with Attachments", W3C Note, December 2000, John J. Barton, Satish Thatte, Henrik Frystyk Nielsen,
http://www.w3.org/TR/SOAP-attachments

[XML-Signature] “XML-Signature Syntax and Processing”, W3C Recommendation, 12 February 2002, D. Eastlake (ed.), J. Reagle (ed.), D. Solo (ed.), M. Bartel, J. Boyer, B. Fox, B. LaMacchia, E. Simon,
http://www.w3.org/TR/xmldsig-core/

[WS-I SAML 1.0] A. Barbir, M. Gudgin, M. McIntosh, and K. S. Morrison (ed.) SAML Token Profile Version 1.0, Web Services Interoperability Organization, Working Group Draft, 19 January 2005.
http://www.ws-i.org/Profiles/SAMLTokenProfile-1.0.html

[XML Infoset] "XML Information Set (Second Edition)", W3C Recommendation, 4 February 2004, John Cowan (ed.), Richard Tobin (ed.),
http://www.w3.org/TR/xmldsig-core/

[RFC 2119] S. Bradner (ed.) Key words for use in RFCs to Indicate Requirement Levels, The Internet Engineering Task Force Best Current Practice, March 1997.
http://www.ietf.org/rfc/rfc2119