SCIDIP-ES
SCIENCE DATA INFRASTRUCTURE FOR PRESERVATION - EARTH SCIENCE

D21.3 Storage Service Installation, Deployment and User Manual

Work package	WP21 Services/Toolkits Development and Adaptation	
Task	Task 21.9 Storage Service Development	
Author (s)	Shirley Crompton	STFC
Author (s)		
Author (s)		
Author (s)		
Author (s)		
Author (s)		
Author (s)		
Authorized by		
Reviewer	Name Surname	Company
Doc Id		
Dissemination Level	~~CONFIDENTIAL/~~PUBLIC	
Issue	1.0	
Date	27/02/2014	

SCIDIP-ES EC Grant Agreement n°. 283401

Abstract:

This document represents the Installation, Deployment and User Manual for the Storage Service developed by the SCIDIP-ES project. This document contains relevant information on how to install, configure and use the Storage Service.

Document Log

Date	Author	Changes	Version	Status
27/02/2014	Shirley Crompton	Created base doc from D21.4 and updated for M30	1.0	Published

TABLE OF CONTENTS

1 Introduction

1.1 *Purpose and Scope*

This document provides an overview of the SCIDIP-ES Storage Service focusing in particular to its installation and usage.

1.2 *Who should read this document*

Users who wish to understand, install and operate the Storage Service.

1.3 *System Context*

The Storage Service is developed by the SCIDIP-ES project to allow existing or new repositories to interact with their holdings as OAIS[1] Archival Information Packages (AIP). An AIP captures all the information needed for the long term preservation of data. For a detailed description of AIP, see Section 3.4 of the D21.3 Master Document. Within the context of the SCIDIP-ES e-infrastructure, Storage Service is used by the Packaging Toolkit to upload packaging information (OAIS AIP) and its associated elements such as Preservation Description Information (PDI) like provenance etc.

2 Design Overview

The Storage Service puts an interface on top of an existing repository to expose an AIP-view of the data holdings irrespective of the physical storage system and information model used. An AIP permits auxiliary preservation information such as metadata, provenance information etc. to be associated with the science data (see Figure 1Table 1). Repositories can use the Storage Service to achieve logical or knowledge preservation, or to render their existing logical preservation processes OAIS-conformant. It should be noted that some additional customization is required to fully integrate Storage Service with existing storage systems.

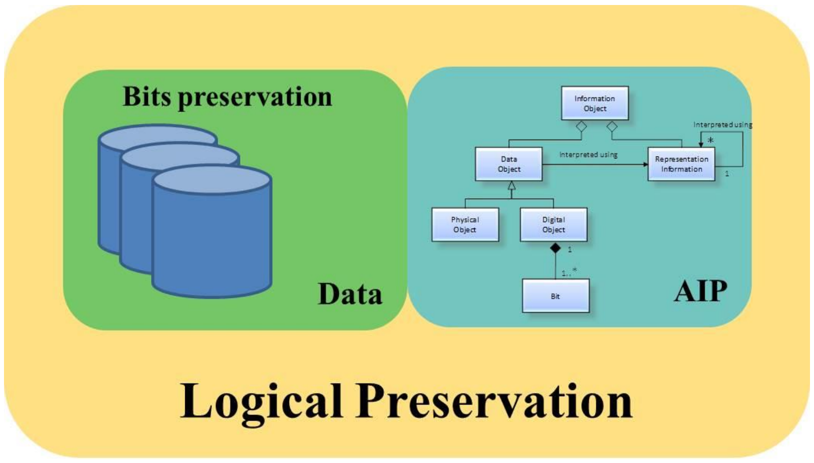


Figure 1 Providing an AIP view of science data.

[1] OAIS - http://public.ccsds.org/publications/archive/650x0m2.pdf

Storage Service is implemented as a multi-modules Maven[2] application written in Java. It allows client to store and retrieve AIPs in exploded or self-contained formats. An exploded AIP contains only the manifest file; the objects referenced by the manifest are stored outside the AIP. A self-contained AIP contains both the manifest file as well as the referenced objects.

Storage Service uses Java interfaces extensively to abstracts the interactions between the different component modules and the underlying information model. Java Service Provider Interface[3] (SPI) mechanism is used to deliver an extensible architecture. Repositories can provide their own custom implementations to support specific business policies and technology (e.g. communication protocol, database/store systems).

The main interfaces used are (see Figure 2):

1. ISSManager – Storage Service Manager Interface. This is the entry point to the application and provides methods for managing AIPs and Information Objects.
2. IDatastoreManager – Datastore Manager Interface. This provides methods for interacting with the persistent data storage
3. IMCATManager - Metadata Catalogue Manager Interface. This provides methods for managing the metadata of the stored items which include their locations within the datastore.
4. ISSid – Interface for the Storage Service Identifier entity.
5. ArchivalInformationPackageInterface – Interface for the Storage Service AIP entity.

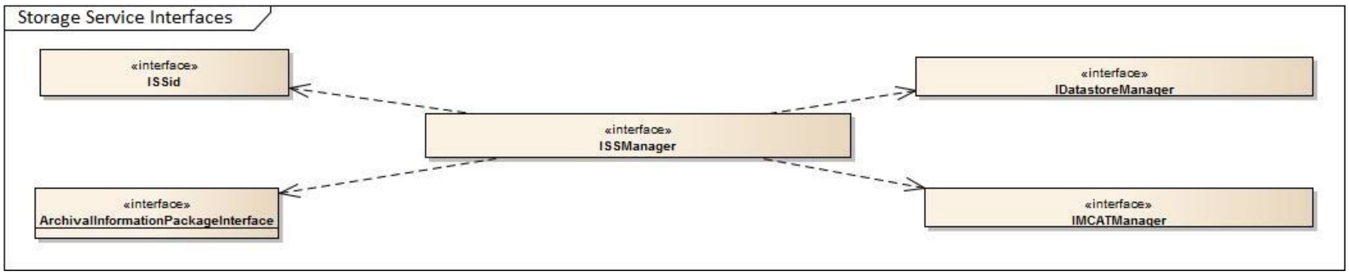

Storage Service Interfaces
«interface» ISSid
«interface» IDatastoreManager
«interface» ISSManager
«interface» ArchivalInformationPackageInterface
«interface» IMCATManager

Figure 2 Storage Service interfaces.

The application contains the following modules (see Figure 3):

1. Storageservice – the Storage Service MAVEN parent module which controls the project build.
2. Storageservice-core – the core component contains the main business logic and the SSManagerImpl class which implements the ISSManager interface and serves as the entry point the the application.
3. Storageservice-datastore – this module contains an implementation of the IDatastoreManager interface and currently provide partial support to the CDMI specifications. It uses the local file system to store the deposited objects.
4. Storageservice-interface – this module contains the interface specifications to enable the modular design adopted by the application.

[2] Apache Maven - http://maven.apache.org/
[3] Service Provider Interface - http://docs.oracle.com/javase/tutorial/ext/basics/spi.html

5. Storageservice-mcat – this module contains an implementation of the IMCATManager interface and is responsible for maintaining metadata associated with the objects, eg. identifiers, internal location, description of the objects etc.
6. Storageservice-rs – this module provides a REST service interface for exposing, via the SSManagerImpl class, the Storageservice-core functionalities to HTTP protocol.
7. Storageservice-demo - this module provides a Swing GUI demo client of the M18 release. This is not packaged with the release but the source code can be downloaded from SourceForge (see Section 3.4).

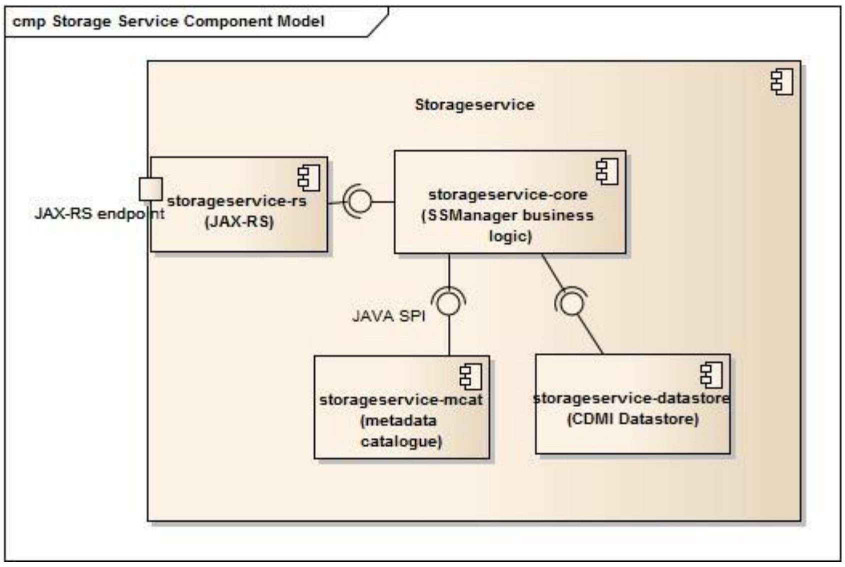

cmp Storage Service Component Model
Storageservice
JAX-RS endpoint
storageservice-rs (JAX-RS)
storageservice-core (SSManager business logic)
JAVA SPI
storageservice-mcat (metadata catalogue)
storageservice-datastore (CDMI Datastore)

Figure 3 – Storage Service component diagram.

2.1 Storage Service MCAT Information Model

The storageservice-mcat module uses a MySQL database to store the metadata including locations of the objects that it manages. The schema is presented in Figure 4 and it contains these entities:

1. SSInfoObj – the generic Information Object entity
2. SSInfoObjGrp – a container entity for grouping Information Objects
3. CDMI_Ref – the CDMI reference entity containing the CDMI identifiers and dereference-able URIs of stored objects provided by the datastore.
4. Type – a lookup table for the OAIS classification associated with an Information Object.

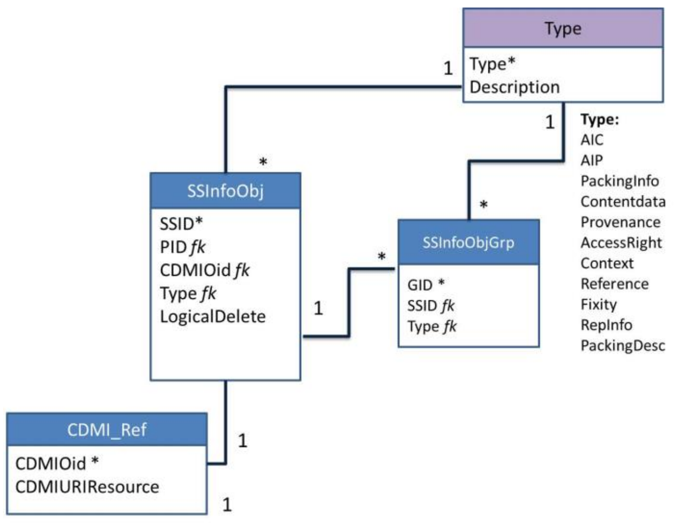


Figure 4 Storageservice-mcat database schema.

3 Installation Guide

3.1 Overview

3.2 Release Notes

The M30 release provides a vanilla implementation for users to try the application and provide feedback. This implementation supports the storage of unpackaged information objects and self-contained OAIS Archival Information Packages (AIPs). It implements part of the CDMI specifications.

3.3 License Information and Terms of Use

SCIDIP-ES Storage Service is licensed under the Apache License, Version 2.0[4] (the "License"). You may not use this software except in compliance with the License. You may obtain a copy of the License at

```
http://www.apache.org/licenses/LICENSE-2.0
```

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

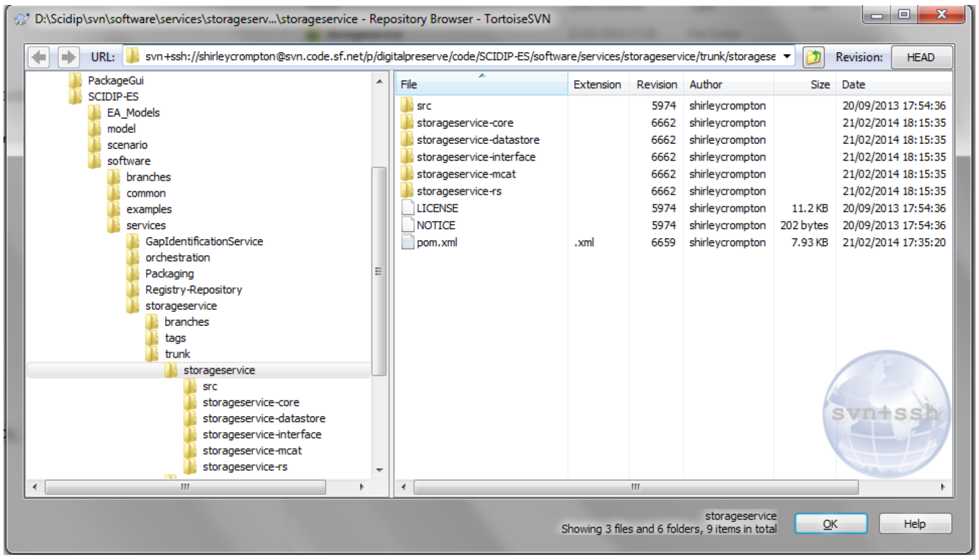
3.4 Download information

The recent stable source code could be accessed from SVN at *Sourceforge*. The URL to the svn trunk is:

```
svn://svn.code.sf.net/p/digitalpreserve/code/SCIDIP-
ES/software/services/storageservice/trunk
```

[4] Apache 2 License - http://www.apache.org/licenses/LICENSE-2.0.html

The repository folder is organised as shown in Figure 5:



D:\Scidip\svn\software\services\storageserv...\storageservice - Repository Browser - TortoiseSVN

URL: svn+ssh://shirleycrompton@svn.code.sf.net/p/digitalpreserve/code/SCIDIP-ES/software/services/storageservice/trunk/storagese   Revision: HEAD

```
PackageGui
SCIDIP-ES
  EA_Models
  model
  scenario
  software
    branches
    common
    examples
    services
      GapIdentificationService
      orchestration
      Packaging
      Registry-Repository
      storageservice
        branches
        tags
        trunk
          storageservice
            src
            storageservice-core
            storageservice-datastore
            storageservice-interface
            storageservice-mcat
            storageservice-rs
```

File	Extension	Revision	Author	Size	Date
src		5974	shirleycrompton		20/09/2013 17:54:36
storageservice-core		6662	shirleycrompton		21/02/2014 18:15:35
storageservice-datastore		6662	shirleycrompton		21/02/2014 18:15:35
storageservice-interface		6662	shirleycrompton		21/02/2014 18:15:35
storageservice-mcat		6662	shirleycrompton		21/02/2014 18:15:35
storageservice-rs		6662	shirleycrompton		21/02/2014 18:15:35
LICENSE		5974	shirleycrompton	11.2 KB	20/09/2013 17:54:36
NOTICE		5974	shirleycrompton	202 bytes	20/09/2013 17:54:36
pom.xml	.xml	6659	shirleycrompton	7.93 KB	21/02/2014 17:35:20

storageservice
Showing 3 files and 6 folders, 9 items in total

Figure 5 SVN structure of Storageservice@SourceForge.

Recent release of the software may also be downloaded from the SCIDIP-ES maven nexus repository at:

> http://nexus.scidip-es.eu/content/repositories/releases/eu/scidipes/services/storageservice/

and the nightly Snapshot build at:

> http://nexus.scidip-es.eu/content/repositories/snapshots/eu/scidipes/services/storageservice/

3.5 Prerequisites

3.5.1 Software prerequisites

- MySQL Server 5.5[5]
- Java SE Development Kit 7[6]
- Tomcat Server 7.0.12 or upwards

In the next section it will be assumed that the MySQL server is already installed on the target machine.

3.5.2 Hardware prerequisites

None.

3.6 OSS/COTS Installation

None.

[5] MySQL Server 5.5 - http://dev.mysql.com/downloads/mysql/5.5.html#downloads
[6] Java SE Development Kit 7 - http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html

3.7 Storage Service Installation

Storage Service is deployed via installing the restservice-<release>.war file into Tomcat (see Section 3.7.3). First, you need to build a custom version configured to your specific deployment environment. The process is explained below.

3.7.1 Setting up the MySQL Database for MCAT

The MCAT database (see Figure 4) needs to be set up and populated with look up data for the Type table. The DDL (storageservice.sql) for creating the schema and SQL (data.sql) for loading the lookup data are provided in Annex C and Annex D respectively.

To create the schema, assuming that the MySQL server is installed and running, run the MySQL command in a standard shell:

```
<shell> mysql –u <username> -p <password> < storageservice.sql
```

Next, populate the Type table with lookup data:

```
<shell> mysql –u <username> -p <password> < data.sql
```

The database connection parameters are set during the Maven build process. Therefore, before you build the module you should amend the live profile in the POM and provide the property values appropriate to your environment. Then, you should be able to automatically deploy and populate the database by running the ant:run goal in the Storageservice.mcat.POM. The *dev profile* in the current release uses these parameters as default:

Property	Value
mysql-url	jdbc:mysql://locahost:3306/storageservice
mysql-dbdriver	com.mysql.jdbc.Driver
mysql-usn	scidipes
mysql-pswd	sc1d3p-3sMl8

Table 1 Storageservice-mcat MySQL connection parameters.

To run the SWING demo client out of the box and assuming that the *dev profile* is active, you need to set up the MySQL database using the above parameters.

3.7.2 Setting up the CDMI File Store

This implementation provides a partial implementation of a CDMI datastore backed by the filesystem. The filesystem location is a property (datastore.basefolder) specified in the profiles of the Storageservice.datastore.POM and set during the Maven build process. You should change this property to suit your environment and then rebuild the application.

The *dev profile* of the current release has the property pre-set to:

```
d:/scidip/storageservice
```

The storageservice.datastore module writes the objects to a subfolder (cdmi_objectid) under this base folder. Please ensure that the following two folders exist and that storageservice-datastore has the privilege to read and write to these locations:

```
<datastore.basefolder>/scidip/storageservice
<datastore.basefolder>/scidip/storageservice/cdmi_objectid
```

3.7.3 Setting up the Rest Service Deployment Properties (optional)

The Storage Service SSManagerImpl (see Section 2) is exposed via a REST service interface for storing and retrieving information objects and AIPs. If you wish to use the tomcat7-maven-plugin to un/re/deploy the service on your tomcat server, you need to edit the relevant profile in the Storageservice.restservice.POM and provide values for the properties listed in Table 2 below. Then you should be able to call the relevant tomcat7 goal to un/re/deploy the serviced during the Maven build process, eg. tomcat7:deploy.

Property	Value
tomcat-server (this points to the server.id property hidden in the Maven .settings.xml)	scidip-tomcat
tomcat-url (this points to the tomcat admin web endpoint)	`http://localhost:8080/manager/text`
username (tomcat admin web property in the Maven .settings.xml)	<yourTomcatAdminUser>
password (tomcat admin web property in the Maven .settings.xml)	<yourTomcatAdminUserPassword>

Table 2 Tomcat properties for un/deploying the REST service using the tomcat7-maven-plugin.

3.7.4 Building the release

Please check out the modules and update the properties as described above. Then use the following Maven command to build the storageservice project:

```
mvn -X clean compile package install -P <your profle>
```

3.7.5 Setting up the Demo client (optional)

The demo client file chooser dialogues use the following pre-defined locations on the file system:

```
Read : c:/scidip

Write: c:/scidip/storageservice/output
```

You may wish to create these locations and grant read/write permission on these to storageservice-demo application. The application should still run even if the locations are not set. It should possible to use the file chooser dialogues to select another location.

3.8 Uninstallation

Storage Service can be uninstalled by performing the following steps:

1. Delete the downloaded files
2. Delete the datastore folder and its content, see Section 3.7.2.
3. Drop the MySQL storageservice schema created in Section 3.7.1.
4. Drop the MySQL user 'scidipes' created in Section 3.7.1.
5. Undeploy the storageservice.war from our Tomcat server, see Section 3.7.3.

4 Using the SCDIP-ES Storage Service Rest Interface

The Rest service currently provides the following endpoints:

Endpoint	Available HTTP Method	Description
http://<host>:<port>/storageservice/infoobject/pid/<pid>	GET/POST	Storing and retrieving information objects using pid. A Storage Service SSID is returned for the POST operation.
http:// <host>:<port>/storageservice/aip/pid/<pid>	GET/POST	Storing and retrieving AIP objects using PID. A Storage Service SSID is returned for the POST operation.

Table 3 Storage Service Rest Service Endpoints

4.1 Using the Rest Service

We can use cURL[7] to test the Rest interface without a Rest service client. To store an object (note the use of the –data-binary flag):

```
curl –X POST --data-binary @<yourfile.txt>  <service endpoint>
```

and to retrieve an object:

```
curl –X GET –o <outputFile> <service endpoint>
```

[7] cURL - http://curl.haxx.se/

Here is an example of storing a text.xml object with a pid of 123 via a proxy using the deployment at the SCIDIP-ES interactive platform @ `http://int-platform.scidip-es.eu/storageservice/`:

```
curl -x wwwcache.dl.ac.uk:8080 -X POST --data-binary @text.xml
http://int-platform.scidip-es.eu/storageservice/infoobject/pid/123
```

Here is an example of retrieving the same object;

```
curl -x wwwcache.dl.ac.uk:8080 –X GET –o text_out.xml http://int-
platform.scidip-es.eu/storageservice/infoobject/pid/123
```

Here is an example of storing an AIP object with a pid of 1234:

```
curl -x wwwcache.dl.ac.uk:8080 -X POST --data-binary @aip.xml
http://int-platform.scidip-es.eu/storageservice/aip/pid/1234
```

And finally an example of retrieving the AIP object:

```
curl -x wwwcache.dl.ac.uk:8080 –X GET –o aip_out.xml http://int-
platform.scidip-es.eu/storageservice/aip/pid/1234
```

5 Using the SCIDIP-ES Storage Service Demo Client

First ensure that the MySQL database service is running and unpack the zip file if not already done. Start the demo client by running the following command (without line break) in a standard shell where the storageservice-demo.jar file is located:

```
java –cp storageservice-demo.jar
com.stfc.scidipes.storageservice.demo.Console
```

The demo client is displayed (see Figure 6).

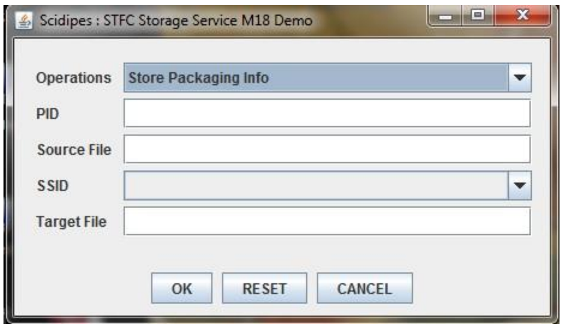

Scidipes : STFC Storage Service M18 Demo
Operations: Store Packaging Info
PID
Source File
SSID
Target File

Figure 6 Storage Service demo GUI.

The Operations combo box provides a choice of operations (see Figure 7) supported by the demo GUI.

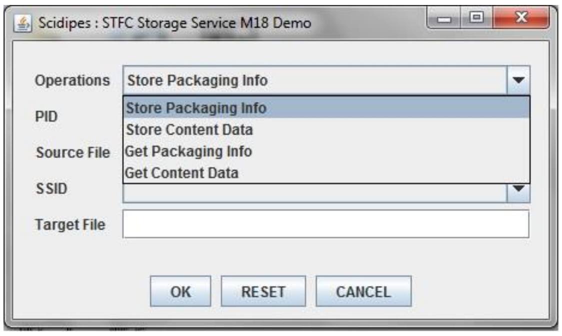


Figure 7 Operations supported by the demo GUI.

5.1 *Store Packaging Info*

To store a packaging information (AIP) object:

1. enter the PID of the object. Please make sure that you press the <enter> key afterwards.
2. click on the Source File input textbox to bring up the File Chooser to select the file (see Figure 8)
3. click the OK button to proceed.

The Store Packaging Info method is the default selection; there is no need to choose it.

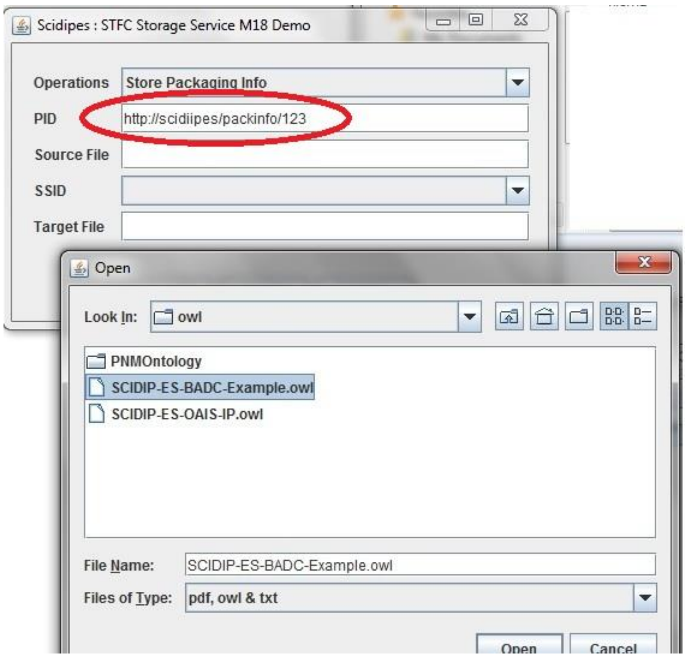
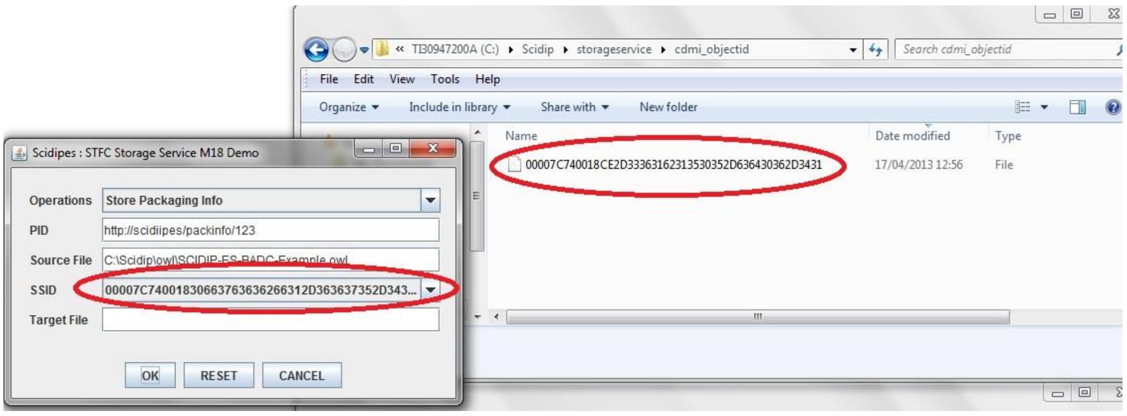


Figure 8 Store a packaging information object.

The file is stored in the CDMI datastore (the file folder located at <datastore.basefolder>/scidip/storageservice/cdmi_objectid) as shown in Figure 9. The file name is the CDMI id generated by the storageservice-datastore module whereas the SSID displayed in the demo GUI is generated by the storageservice-core module. The former is internal to Storage Service and the latter can be used by external clients to retrieve the stored objects from Storage Service.



Scidipes : STFC Storage Service M18 Demo

Operations: Store Packaging Info
PID: http://scidiipes/packinfo/123
Source File: C:\Scidip\owl\SCIDIP-ES-BADC-Example.owl
SSID: 00007C7400183066376363626631 2D363637352D343...
Target File:

OK | RESET | CANCEL

« TI30947200A (C:) ▸ Scidip ▸ storageservice ▸ cdmi_objectid

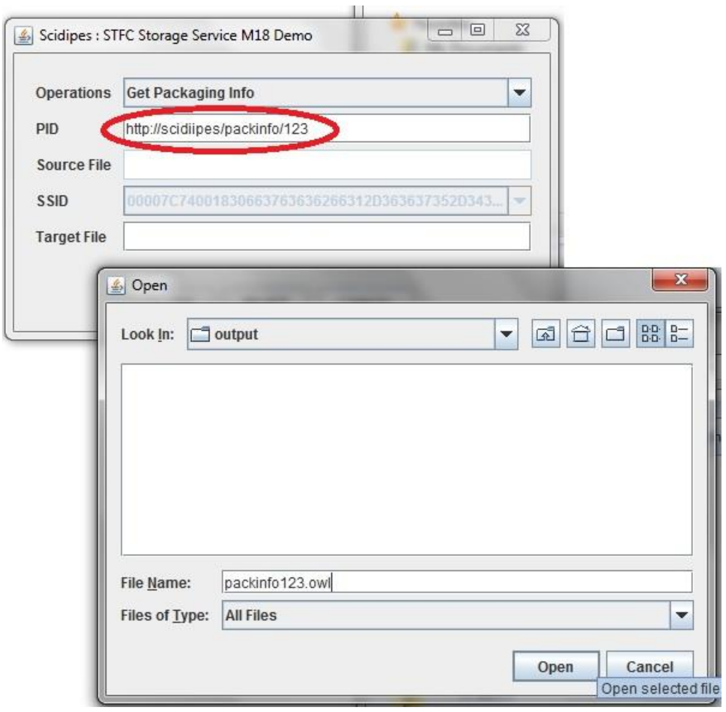
Name	Date modified	Type
00007C740018CE2D33363162313530352D636430362D3431	17/04/2013 12:56	File

Figure 9 Packaging information object written to the CDMI datastore.

5.2 *Get Packaging Info*

To retrieve the Packaging info object stored in the previous section:

1. press the reset button to clear the existing input
2. select the Get Packaging Info operation
3. provide the PID of the object. Please make sure that you press the <enter> key afterwards
4. click on the Target File input textbox to open the File Chooser and provide an output file name (see Figure 10)
5. click the OK button to proceed.



Scidipes : STFC Storage Service M18 Demo

Operations: Get Packaging Info
PID: http://scidiipes/packinfo/123
Source File:
SSID: 00007C74001830663763636266312D363637352D343...
Target File:

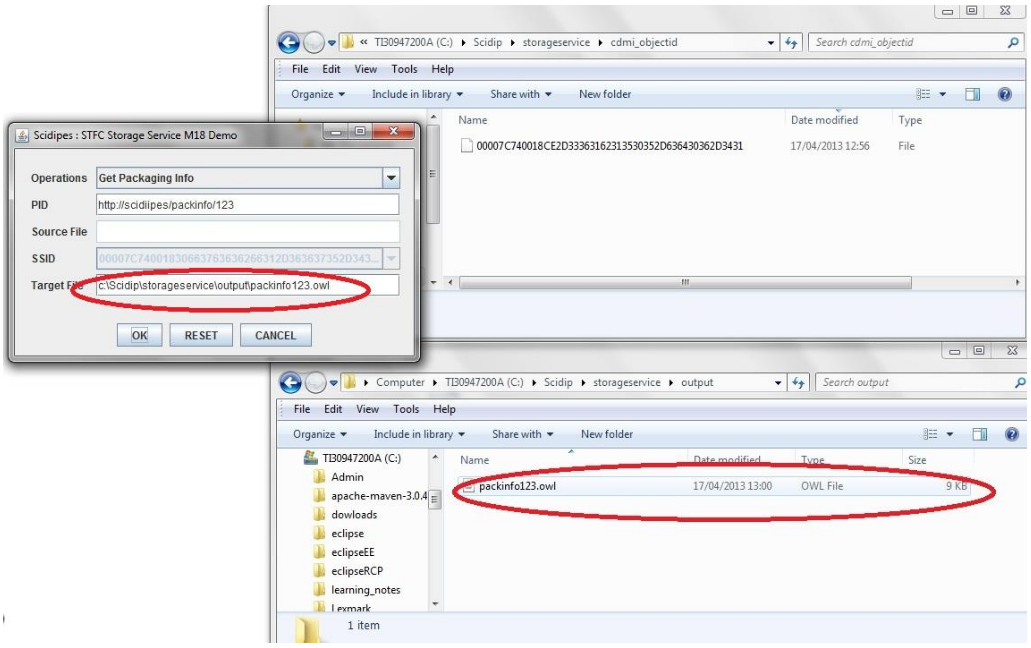
Open

Look In: output
File Name: packinfo123.owl
Files of Type: All Files

Open | Cancel
Open selected file

Figure 10 Retrieve the stored packaging information object by its PID.

Figure 11 shows that the object has been retrieved and written with the specified name to the target location. Figure 12 shows the stored and retrieved file side by side.



Scidipes : STFC Storage Service M18 Demo

Operations	Get Packaging Info
PID	http://scidiipes/packinfo/123
Source File	
SSID	00007C740018306637636302663120363637352D343...
Target File	c:\Scidip\storageservice\output\packinfo123.owl

OK RESET CANCEL

« TI30947200A (C:) ▸ Scidip ▸ storageservice ▸ cdmi_objectid

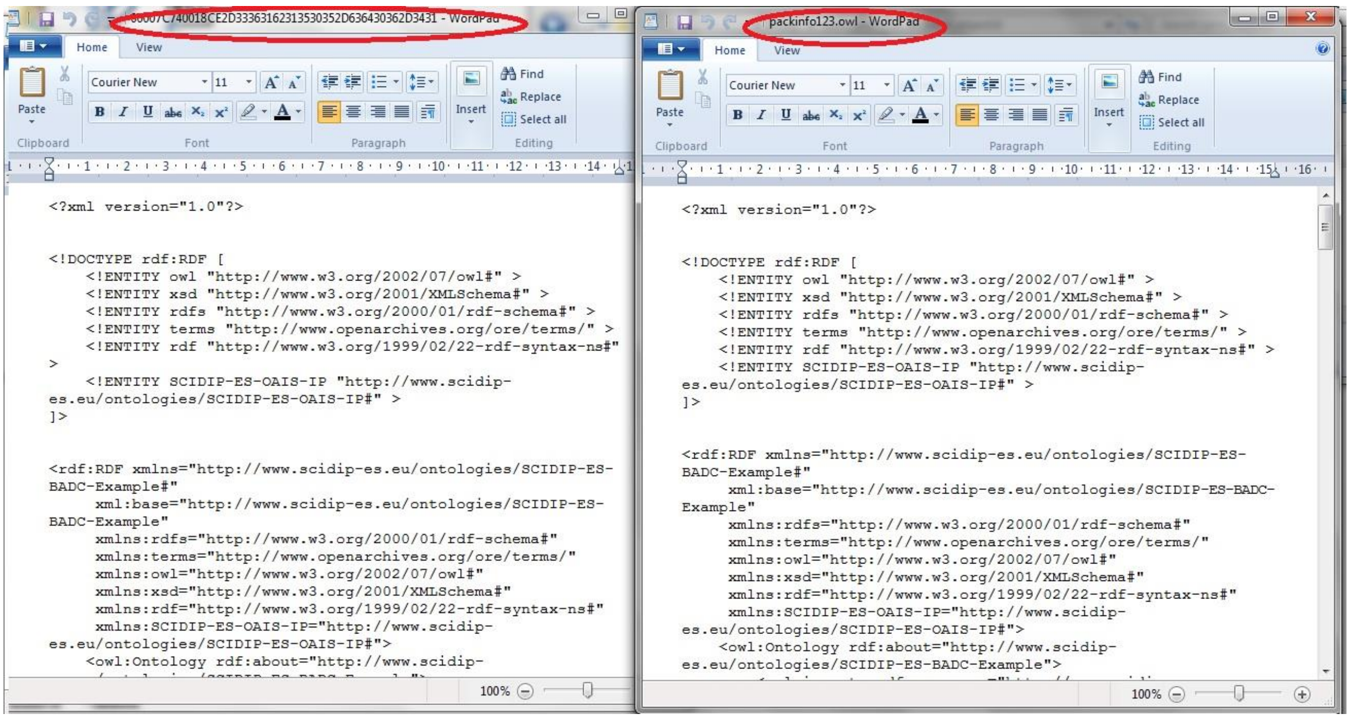
Name	Date modified	Type
00007C740018CE2D333631623135303352D636430362D3431	17/04/2013 12:56	File

▸ Computer ▸ TI30947200A (C:) ▸ Scidip ▸ storageservice ▸ output

Name	Date modified	Type	Size
packinfo123.owl	17/04/2013 13:00	OWL File	9 KB

1 item

Figure 11 The retrieved file is written to the output folder.



00007C740018CE2D333631623135303352D636430362D3431 - WordPad

```
<?xml version="1.0"?>


<!DOCTYPE rdf:RDF [
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY terms "http://www.openarchives.org/ore/terms/" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
>
    <!ENTITY SCIDIP-ES-OAIS-IP "http://www.scidip-
es.eu/ontologies/SCIDIP-ES-OAIS-IP#" >
]>


<rdf:RDF xmlns="http://www.scidip-es.eu/ontologies/SCIDIP-ES-
BADC-Example#"
     xml:base="http://www.scidip-es.eu/ontologies/SCIDIP-ES-
BADC-Example"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:terms="http://www.openarchives.org/ore/terms/"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:SCIDIP-ES-OAIS-IP="http://www.scidip-
es.eu/ontologies/SCIDIP-ES-OAIS-IP#">
    <owl:Ontology rdf:about="http://www.scidip-
```

packinfo123.owl - WordPad

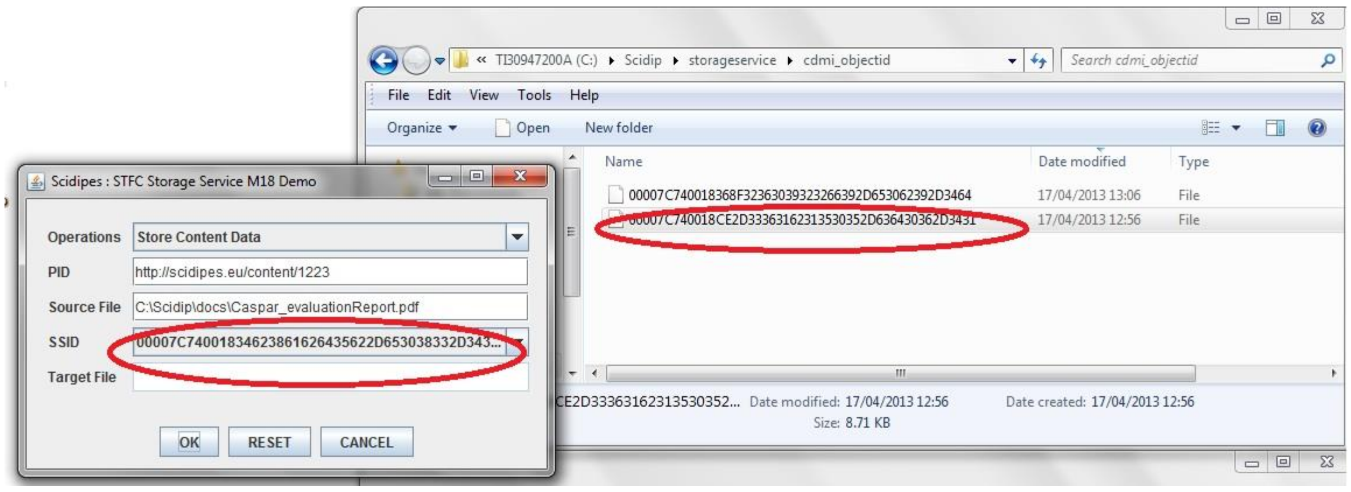
```
<?xml version="1.0"?>


<!DOCTYPE rdf:RDF [
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY terms "http://www.openarchives.org/ore/terms/" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
    <!ENTITY SCIDIP-ES-OAIS-IP "http://www.scidip-
es.eu/ontologies/SCIDIP-ES-OAIS-IP#" >
]>


<rdf:RDF xmlns="http://www.scidip-es.eu/ontologies/SCIDIP-ES-
BADC-Example#"
     xml:base="http://www.scidip-es.eu/ontologies/SCIDIP-ES-BADC-
Example"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:terms="http://www.openarchives.org/ore/terms/"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:SCIDIP-ES-OAIS-IP="http://www.scidip-
es.eu/ontologies/SCIDIP-ES-OAIS-IP#">
    <owl:Ontology rdf:about="http://www.scidip-
es.eu/ontologies/SCIDIP-ES-BADC-Example">
```

Figure 12 Comparison of the stored and retrieved file.

5.3 *Store Content Data*

This operation works in the same way as Store Packaging Info (Section 5.1):

1. enter the PID of the object. Please make sure that you press the <enter> key afterwards.
2. click on the Source File input textbox to bring up the File Chooser to select the file
3. click the OK button to proceed.


Scidipes : STFC Storage Service M18 Demo

Operations	Store Content Data
PID	http://scidipes.eu/content/1223
Source File	C:\Scidip\docs\Caspar_evaluationReport.pdf
SSID	00007C74001834623861626435622D653038332D343...
Target File	

TI30947200A (C:) ▸ Scidip ▸ storageservice ▸ cdmi_objectid

Name	Date modified	Type
00007C740018368F32363039323266392D653062392D3464	17/04/2013 13:06	File
00007C740018CE2D33363162313530352D636430362D3431	17/04/2013 12:56	File

CE2D33363162313530352... Date modified: 17/04/2013 12:56 Date created: 17/04/2013 12:56
Size: 8.71 KB

Figure 13 Storing a Content Data object.

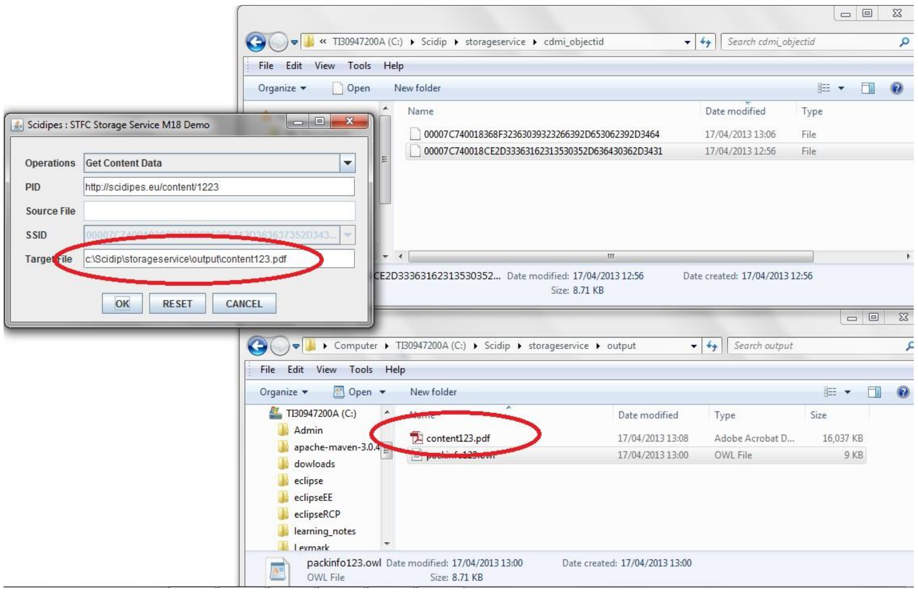
Figure 13 shows that the object is written to the CDMI datastore. Note that we have chosen as Source File the Caspar_evaluationReport.pdf. We will retrieve this in the next section.

5.4 *Get Content Data*

To retrieve the stored Content Data object, follows the same procedure in Section 5.2:

1. press the reset button to clear the existing input
2. select the Get Content Data operation
3. provide the PID of the object. Please make sure that you press the <enter> key afterwards
4. click on the Target File input textbox to open the File Chooser and provide an output file name
5. click the OK button to proceed.

Figure 14 shows that the file has been retrieved successfully and written to the required location.


Scidipes : STFC Storage Service M18 Demo

Operations	Get Content Data
PID	http://scidipes.eu/content/1223
Source File	
SSID	00007C740018... 312D363637352D343...
Target File	c:\Scidip\storageservice\output\content123.pdf

OK RESET CANCEL

« TI30947200A (C:) ▸ Scidip ▸ storageservice ▸ cdmi_objectid

Name	Date modified	Type
00007C740018368F32363039323266392D653062392D3464	17/04/2013 13:06	File
00007C740018CE2D33363162313530352D636430362D3431	17/04/2013 12:56	File

CE2D33363162313530352... Date modified: 17/04/2013 12:56 Date created: 17/04/2013 12:56
Size: 8.71 KB

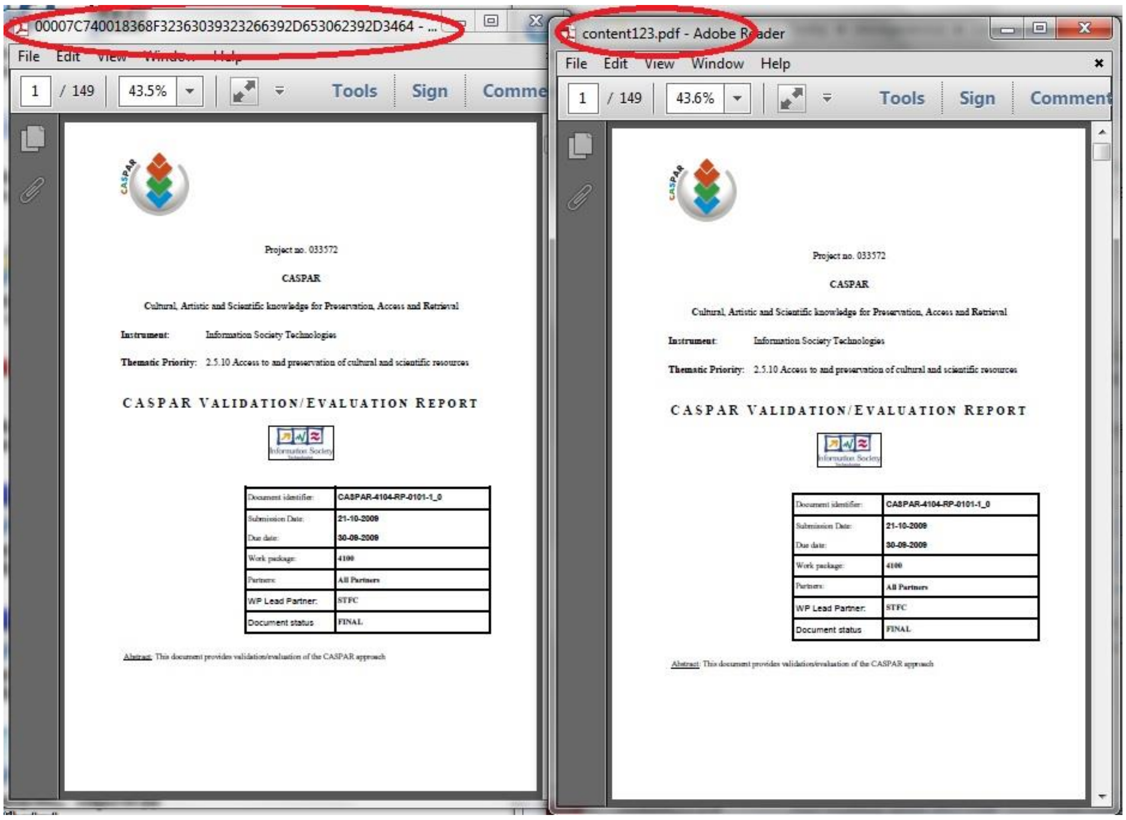
▸ Computer ▸ TI30947200A (C:) ▸ Scidip ▸ storageservice ▸ output

Name	Date modified	Type	Size
content123.pdf	17/04/2013 13:08	Adobe Acrobat D...	16,037 KB
packinfo123.owl	17/04/2013 13:00	OWL File	9 KB

packinfo123.owl Date modified: 17/04/2013 13:00 Date created: 17/04/2013 13:00
OWL File Size: 8.71 KB

Figure 14 Getting the Content Data back.

Figure 15 shows the stored and retrieved Content Data object (the Caspar_evaluationReport.pdf) side by side.


00007C740018368F32363039323266392D653062392D3464 - ...

1 / 149 43.5%

Project no. 033572

CASPAR

Cultural, Artistic and Scientific knowledge for Preservation, Access and Retrieval

Instrument: Information Society Technologies

Thematic Priority: 2.5.10 Access to and preservation of cultural and scientific resources

CASPAR VALIDATION/EVALUATION REPORT

Document identifier:	CASPAR-4104-RP-0101-1_0
Submission Date:	21-10-2009
Due date:	30-09-2009
Work package:	4100
Partners:	All Partners
WP Lead Partner:	STFC
Document status	FINAL

Abstract: This document provides validation/evaluation of the CASPAR approach

content123.pdf - Adobe Reader

1 / 149 43.6%

Project no. 033572

CASPAR

Cultural, Artistic and Scientific knowledge for Preservation, Access and Retrieval

Instrument: Information Society Technologies

Thematic Priority: 2.5.10 Access to and preservation of cultural and scientific resources

CASPAR VALIDATION/EVALUATION REPORT

Document identifier:	CASPAR-4104-RP-0101-1_0
Submission Date:	21-10-2009
Due date:	30-09-2009
Work package:	4100
Partners:	All Partners
WP Lead Partner:	STFC
Document status	FINAL

Abstract: This document provides validation/evaluation of the CASPAR approach

Figure 15 The stored and retrieved Content Data object side by side.

6 Reference Manual

6.1 Keyboard shortcuts

None.

6.2 Command-line commands

None.

6.3 Public APIs

The Storage Service is designed as a web service and the public API for the storageservice-core.SSManagerImpl class is exposed as a JAX-RS service (see Section 3) accessible via HTTP uniform methods. However, you may also use the SSManagerImpl class locally via its public API as illustrated by the Storageservice-demo Client application (see Section 5). Table 4 lists the key ISSManager interface methods relevant for the M30 release. The full interface is specified in the storageservice-interface module available from the SourceForge @ svn://svn.code.sf.net/p/digitalpreserve/code/SCIDIP-ES/software/services/storageservice/trunk/storageservice/storageservice-interface.

Method Summary	
byte[]	**accessInfoObject(ISSid ssid)** Retrieve an Information Object using its storage service identifier (SSID)
byte[]	**accessInfoObject(String pid)** Retrieve an Information Object using its persistent identifier (PID)
byte[]	**accessInfoObjectAsAip(ISSid ssid)** Retrieve an Information Object packaged as an AIP using its SSID.
byte[]	**accessInfoObjectAsAip(String pid)** Retrieve an Informaton Object packaged as an AIP using its PID.
ISSId	**storeInfoObject(byte[] bytes, ISSid ssid)** Store an Information Object using an existing SSID.
ISSId	**storeInfoObject(byte[] bytes, String pid)** Store an Information Object using an existing PID.
byte[]	**accessAIP(ISSid ssid)** Retrieve an object stored as a self-contained AIP using its SSID.
byte[]	**accessAIP(String pid)**

	Retrieve an object stored as a self-contained AIP using its PID.
ISSid	**storeAIP(byte[] bytes, ISSid ssid)** Store a self-contained AIP object using an existing SSID.
ISSid	**storeAIP(byte[] bytes, Sring pid)** Store a self-contained AIP object using an existing PID.

Table 4 ISSManager interface method (relevant for the M30 release).

7 Future Work

The following enhancements are proposed to improve the usability of Storage Service:

1. Provide fuller support to the CDMI specifications by wrapping the backend CDMI datastore as an JAX-RS service and the addition of a storageservice-datastore-client module (see Figure 16). This allows the main storageservice application to be deployed remotely from the datastore.
2. Add more methods in the storageservice-rs module to expose functionalities for getting an information object packaged as an AIP.
3. Enable dynamic configuration of application properties via external configuration files.
4. Rework the de/encoding functions to support the de/serialisation of objects in different formats.
5. Improve error handling in the storageservice-rs module and to implement bugfixes and enhancements identified by the user testing (WP22) and service evolution (WP24) work packages.

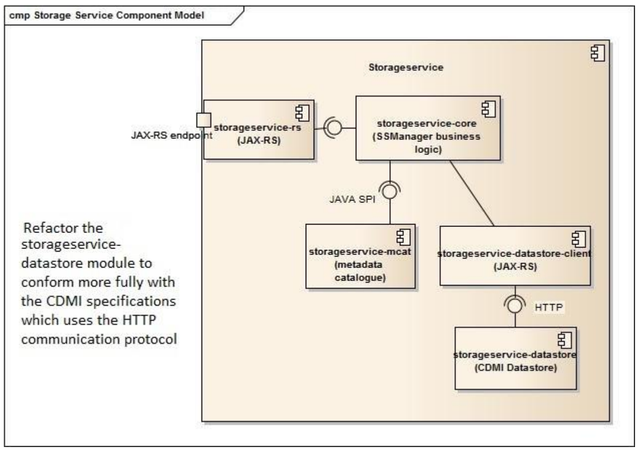

cmp Storage Service Component Model
Storageservice
JAX-RS endpoint
storageservice-rs (JAX-RS)
storageservice-core (SSManager business logic)
JAVA SPI
storageservice-mcat (metadata catalogue)
storageservice-datastore-client (JAX-RS)
HTTP
storageservice-datastore (CDMI Datastore)
Refactor the storageservice-datastore module to conform more fully with the CDMI specifications which uses the HTTP communication protocol

Figure 16 Component diagram illustrating the improvement required.

8 Troubleshooting Common Issues

If the demo GUI displays a no PID error message, please make sure that you have press the <Enter> key after inputting the PID.

Annex A. Figures and Tables

A.1. List of Figures

A.2. List of Tables

Annex B. Terminology

ACRONYM	DESCRIPTION
AIP	Archival Information Package
ARK	Archival Resource Key
CDMI	Cloud Data Management Interface
DOI	Digital Object Identifier
ES	Earth Science
GIS	Gap Identification Service
KB	Knowledge Base
OAIS	Open Archival Information System
OWL	Web Ontology Language
PID	Persistent Identifier
PNM	Preservation Network Model
PURL	Persistent Uniform Resource Locator
RDF	Resource Description Framework
RepInfo	Representation Information
SNIA	Storage Networking Industry Association
SS	Storage Service
VM	Virtual Machine
WP	Work Package
XAM	eXtensible Access Method
XML	eXtensible Mark-up Language

Annex C. Database schema (storageservice.sql)

```
-- MySQL DDL for Storage Servie
--
-- Host: localhost    Database: storageservice
-- ------------------------------------------------------
-- Server version    5.5.29
-- need to run from another database

DROP SCHEMA IF EXISTS storageservice;

CREATE SCHEMA storageservice CHARACTER SET utf8mb4 COLLATE
utf8mb4_unicode_ci;


-- GRANT ALL PRIVILEGES ON storageservice.* TO 'scidipes'@'%';


CREATE TABLE storageservice.cdmi_obj
(
  cdmi_oid                VARCHAR(100)  NOT NULL,
  cdmi_uri_res        VARCHAR(255)    DEFAULT NULL,
  PRIMARY KEY (cdmi_oid)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

CREATE TABLE storageservice.ss_info_obj_type
(
  obj_type            VARCHAR(100)    NOT NULL,
  description    VARCHAR(255)    DEFAULT NULL COMMENT 'Description ',
  PRIMARY KEY (obj_type)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='OAIS type';


CREATE TABLE storageservice.ss_info_obj_grp
(
  gid            int(11)                    NOT NULL,
  ssid           VARCHAR(50)                        NOT NULL,
  grp_obj_type   enum('AIC','AIP')     DEFAULT 'AIP'   COMMENT 'Group obj
type can be AIC or AIP only',
  log_deleted    enum('0','1')         DEFAULT '0'     COMMENT 'Has this been
logically deleted, 0: false, 1: true',
  PRIMARY KEY (gid,ssid)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

CREATE TABLE storageservice.ss_info_obj
(
  ssid                VARCHAR(50)          NOT NULL,
  pid                 VARCHAR(255)         DEFAULT NULL    COMMENT 'The
persistent identifier provided by client on upload',
  obj_type            VARCHAR(100)         NOT NULL        COMMENT 'oais
info obj type',
  cdmi_oid            VARCHAR(100)         DEFAULT NULL    COMMENT 'cdmi
object identifier',
  log_deleted   enum('0','1')         DEFAULT '0'     COMMENT 'Has this been
logically deleted, 0: false, 1: true',
  PRIMARY KEY (ssid)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='Storage Service Info Object
entity, this is the main table';


ALTER TABLE storageservice.ss_info_obj ADD
(
  CONSTRAINT FK_SSIO_TYPE
  FOREIGN KEY(obj_type)
  REFERENCES storageservice.ss_info_obj_type(obj_type)
  ON DELETE NO ACTION
  ON UPDATE CASCADE
);

ALTER TABLE storageservice.ss_info_obj ADD
(
  CONSTRAINT FK_SSIO_CDMI
  FOREIGN KEY(cdmi_oid)
  REFERENCES storageservice.cdmi_obj(cdmi_oid)
  ON DELETE NO ACTION
  ON UPDATE CASCADE
);
/*
ALTER TABLE storageservice.ss_info_obj_grp ADD
(
  CONSTRAINT FK_SSIOGRP_TYPE
  FOREIGN KEY(obj_type)
  REFERENCES storageservice.ss_info_obj_type(obj_type)
  ON DELETE NO ACTION
  ON UPDATE CASCADE
);*/

ALTER TABLE storageservice.ss_info_obj_grp ADD
(
  CONSTRAINT FK_SSIOGRP_SSIO
  FOREIGN KEY(ssid)
  REFERENCES storageservice.ss_info_obj(ssid)
  ON DELETE NO ACTION
  ON UPDATE CASCADE
);

COMMIT;
```

Annex D. Look up data (data.sql)

```
-- first delete all records if exist
DELETE FROM storageservice.ss_info_obj_type;
-- then insert the look up data
INSERT INTO storageservice.ss_info_obj_type VALUES('InfoObject','OAIS
Information Object. Composed of a Data Object and the Representation
Information.');
INSERT INTO storageservice.ss_info_obj_type VALUES('PackagingInfo','OAIS
Packaging Information object. Stores the resource map with pointers to
aggregated resources');
INSERT INTO storageservice.ss_info_obj_type VALUES('AIC','OAIS Archival
Information Package object');
INSERT INTO storageservice.ss_info_obj_type VALUES('AIP','OAIS Archival
Information Collection object');
INSERT INTO storageservice.ss_info_obj_type VALUES('ContentData','OAIS
Content Data object');
INSERT INTO storageservice.ss_info_obj_type VALUES('Provenance','OAIS
Preservation Description Info Provenance object');
INSERT INTO storageservice.ss_info_obj_type VALUES('AccessRight','OAIS
Preservation Description Info Access Right object');
INSERT INTO storageservice.ss_info_obj_type VALUES('Context','OAIS
Preservation Description Info Context object');
INSERT INTO storageservice.ss_info_obj_type VALUES('Reference','OAIS
Preservation Description Info Reference object');
INSERT INTO storageservice.ss_info_obj_type VALUES('Fixity','OAIS
Preservation Description Info Fixity object');
-- Not sure we need the next three
INSERT INTO storageservice.ss_info_obj_type VALUES('RepInfo','OAIS
Representation Information object.');
INSERT INTO storageservice.ss_info_obj_type VALUES('PackageDesc','OAIS
Package Description object');
INSERT INTO storageservice.ss_info_obj_type VALUES('PDI','OAIS Preservation
Description Information');
--
COMMIT;
```