D21.3 SCIDIP-ES HAPPI Software: Installation, Deployment and User Manual

Work package	WP21 Services/Toolkits Development and Adaptation	
Task	T21.11 Authenticity, Provenance and Integrity Toolkit T21.12 Persistent Identifiers Service	
Author (s)	Luigi Briguglio, Paul Kristian Dion Pacquing	ENG
Authorized by	Name Surname	Company
Reviewer	Name Surname	Company
Doc Id		
Dissemination Level	PUBLIC	
Issue	1.0	
Date	21/02/2014	

Abstract:

This document represents the User Manual for the software (called HAPPI) built for supporting Authenticity, Provenance, Integrity and Persistent Identifiers developed in the frame of SCIDIP-ES project. This updated and living document contains all useful information on how to install, configure and use SCIDIP-ES HAPPI.

Document Log

Date	Author	Changes	Version	Status
14/02/2014	Luigi Briguglio	Base document created from D21.4 v1.4. Documentation version aligned with the HAPPI release 1.5.0.	0.9	Final
21/02/2014	Luigi Briguglio	Revision based on feedback from Shirley	1.0	Final

TABLE OF CONTENTS

1 Introduction

1.1 Purpose and Scope

This document provides an overview of the software designed, developed and tested for supporting Authenticity, Provenance, Integrity and Persistent Identifiers in the frame of SCIDIP-ES project. The software is called HAPPI, acronym for Handling Authenticity Provenance and Persistent Identifiers. This document updates specification and manual, based on new results achieved at month 30, according to the defined development plan and it focuses in particular to models, specification and usage scenarios behind the software SCIDIP-ES HAPPI.

1.2 Who should read this document

Users who may want to understand and operate on digital archives/libraries, by addressing specifically Authenticity, Provenance, Integrity and Persistent Identifiers. Mainly, the SCIDIP-ES HAPPI is designed for providing standard interfaces for capturing, managing and exchanging crucial information for supporting the preservation activities. These interfaces may be implemented by each archive, based on its specific needs, policies and existing systems. In practice, the main HAPPI "users" are archive managers, curators, software architects and developers.

1.3 System Context

The SCIDIP-ES HAPPI is a component of the SCIDIP-ES e-infrastructure. Its main role is to provide support for handling Authenticity, Provenance, Integrity and Persistent Identifiers on data objects being preserved.

2 Design Overview

The design of SCIDIP-ES HAPPI is based on research and development results coming from SCIDIP-ES Authenticity-Provenance-Integrity Toolkit[1] and Persistent Identifiers Service development tasks. The general overview of this deliverable (D21.3 Master) has already introduced the role of HAPPI which is reiterated as follows:

- **Problem**: Preservation requires, besides keeping bits, ensuring the information encoded in a digital object continues to be usable, and there is **evidence** that the digital object is what it is claimed to be. This evidence provides details on any transformation and/or changes that have been carried out on the digital object and ensures that the **integrity and authenticity** of the object itself.

- **Standard approach**: ISO:14721:2003 (OAIS) identifies and specifies Preservation Description Information (**PDI**) as the information about where to find and obtain the evidence (see previous point) and other required information for supporting the preservation activities. PDI includes **Provenance**, **Fixity**, **Reference**, Context and, in the last version of OAIS, Access Rights.

[1] Part of models, presented in this document, have been submitted and accepted to ECLAP 2013 conference [Briguglio2013]

- **SCIDIP-ES Solution**: The SCIDIP-ES e-infrastructure provides an integrated solution called **SCIDIP-ES HAPPI to** support archive managers and curators in this critical preservation operation.

Essentially, **SCIDIP-ES HAPPI** helps archive managers and curators to capture and manage part of the PDI; the part which covers provenance, reference, evidences of changes and integrity, and is collectively referred to as Evidence History. Figure 1 shows the relationship between the concepts of OAIS PDI and Evidence History.

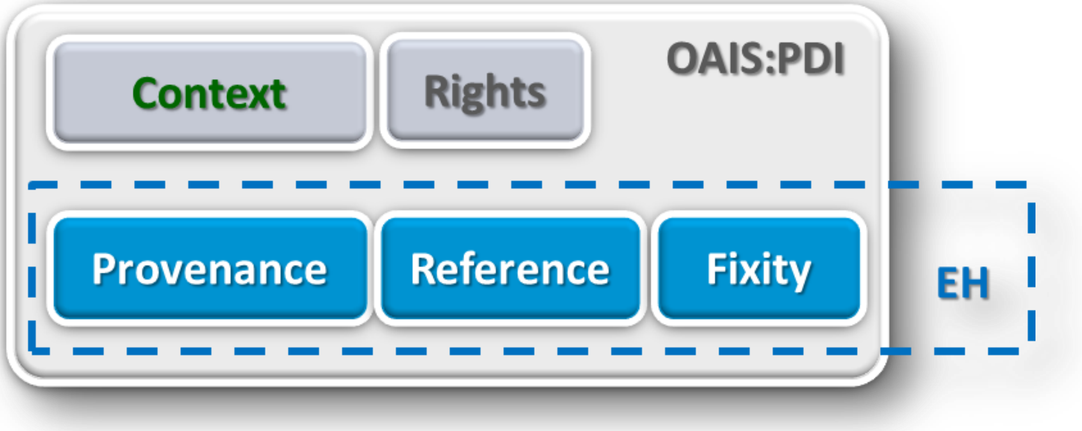


Figure 1 Relationship between evidence history and OAIS PDI

Evidence History is the set of information capturing all the transformations and events that a digital object has undergone since its creation and each single transformation and event is documented by a single Evidence Record. Thus, each Evidence History comprises a set of Evidence Records (as shown in Figure 2).

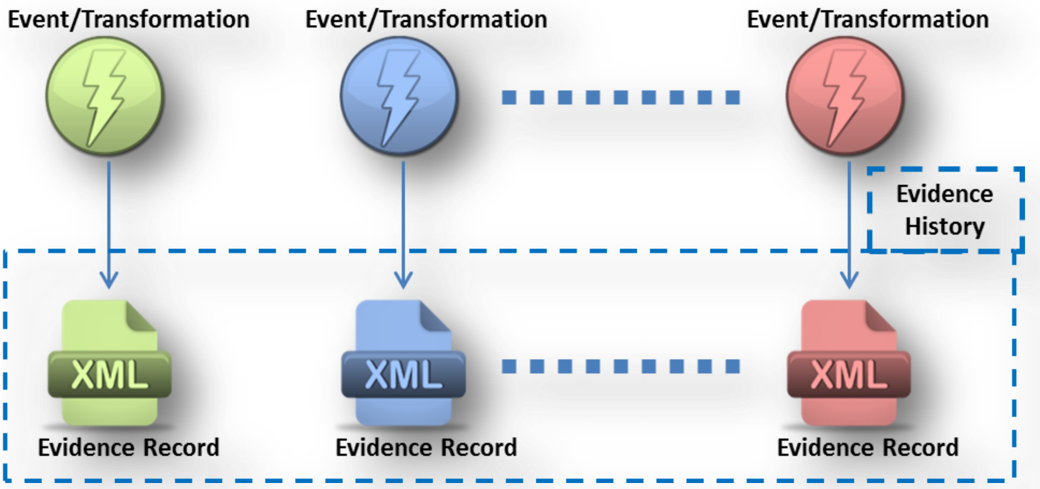


Figure 2 Events, evidence records and evidence history

The structure of an evidence record is mainly characterised by three main sections:

- **Header** – includes reference to the content information which is the preservation target, as well as external references used in describing the current evidence record;
- **Entities** – defines entities involved in the event/transformation: e.g. the result (called representation) from the event/transformation, the transformation that generated the result, and the agent who controlled the transformation;
- **Description** – describes the event/transformation by providing details which could be used later to assess authenticity: e.g. timestamp, fixity, annotations.

The modularity of the evidence record is justified by one of the most important requirements emerged during the analysis of user requirements [D12.1] and service robustness research [D32.2]. Indeed, SCIDIP-ES HAPPI is designed mainly for supporting operational activities of archive managers which requires interaction with their specific existing legacy systems. In particular, modularity of evidence history is important for supporting the activity "Change of Custody" (i.e. the custody of a digital repository content is transferred to a new repository) which may require custodians to exchange an "interoperable" evidence history.

For this reason, we surveyed the Earth Science Community to quantify their existing data archiving and managing processes. This survey captured the most widely used information (i.e. a core set of information) for gathering evidences and tracing events and transformations impacting on digital assets. An analysis of the survey result indicates that SCIDIP-ES HAPPI needs to interact with an archive's existing systems which deal with:

- **provenance/event tracing**: usually managed information contains timestamp, reference to data impacted by the event/transformation, type of event/transformation, title of the event/transformation, responsible/controller of the event/transformation, annotation for explaining the event/transformation;
- **identifier assignment**: usually identifiers assigned according to organisational policy and tools;
- **fixity management:** usually a checksum assigned according to organisational policy and tools.

Based on this result, in order to address the "interoperability" requirement, the model of SCIDIP-ES HAPPI has been defined by adopting two prominent standards for exchanging information in the Long Term Digital Preservation community: [OPM1.1] and [PREMIS2.2]. Both OPM1.1 and PREMIS2.2 have been analysed in D32.2. The analysis helped identify elements and terms useful for describing evidence of event/transformation. Moreover, to ensure interoperability between different archives, HAPPI adopts XML to facilitate the import/export of information.

Figure 3 illustrates the context in which SCIDIP-ES HAPPI works and its key functionalities.

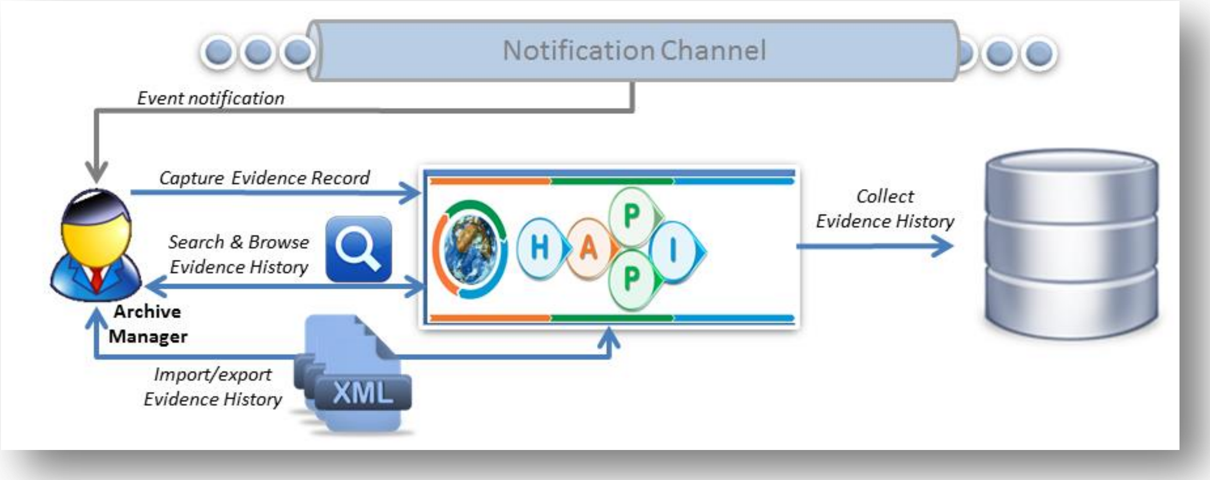


Figure 3 SCIDIP-ES HAPPI - system context

2.1 SCIDIP-ES HAPPI Implementation

SCIDIP-ES HAPPI is a Java software package that mainly manages two key concepts of HAPPI Information Model: Intellectual Entity and Evidence History. SCIDIP-ES HAPPI provides its functionality through web (REST) interfaces.

2.1.1 SCIDIP-ES HAPPI Information Model

Figure 4 shows the concepts (Java classes) of SCIDIP-ES HAPPI and their relationships. It should be noted that each item of the evidence record (class EvidenceItem) is characterised by a label, an identifier and an annotation. Annotation is used for descriptive and retrieval purposes.

The class Reference, used for assigning an identifier, is composed by:

i) Type – type of reference (e.g. URI, EventId);
ii) Value – value of reference;
iii) Organisation – responsible to assign the reference.

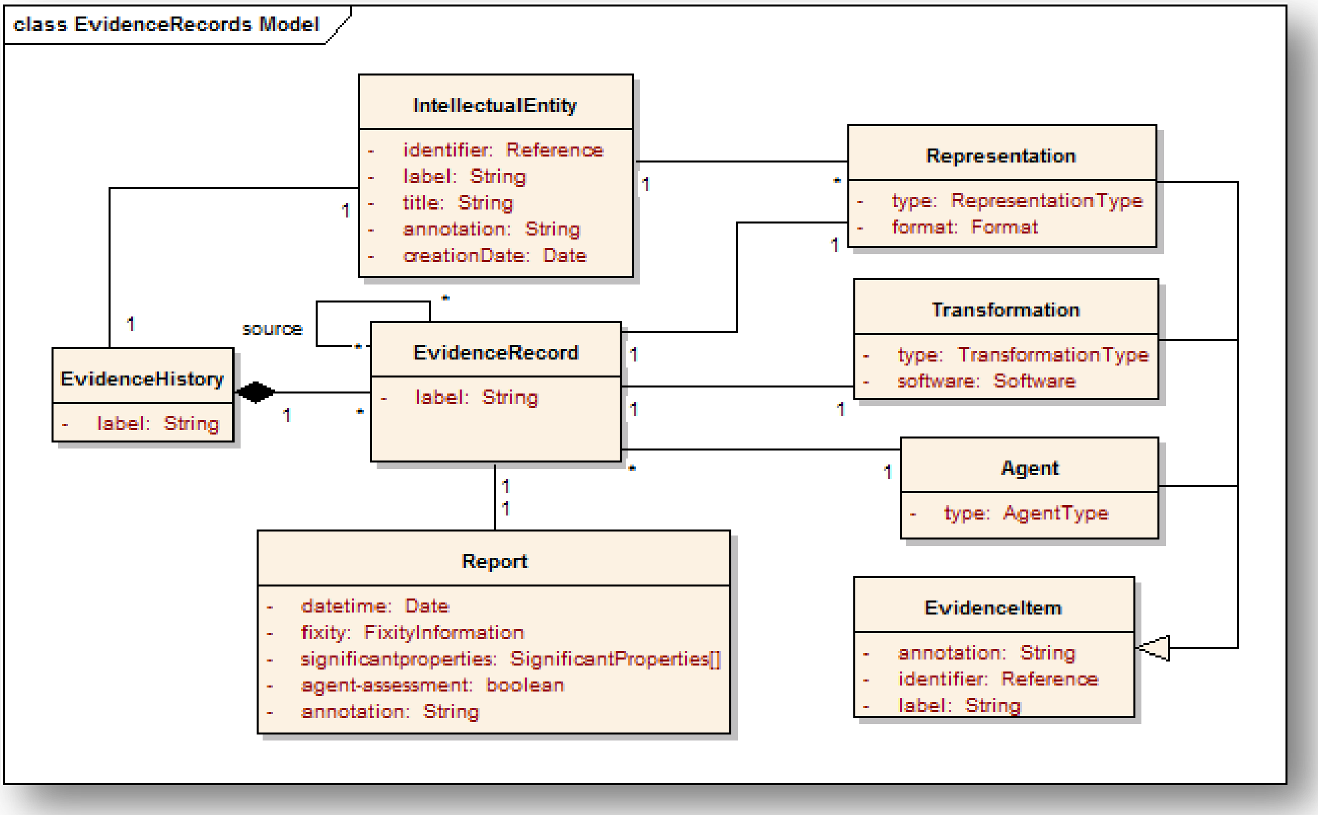


Figure 4 SCIDIP-ES HAPPI Information Model

For managing the classes IntellectualEntity and the EvidenceHistory, SCIDIP-ES HAPPI provides two interfaces: IntellectualEntityManager and EvidenceHistoryManager. Figure 5 below shows the UML class diagram of the SCIDIP-ES HAPPI interfaces and their implementations.

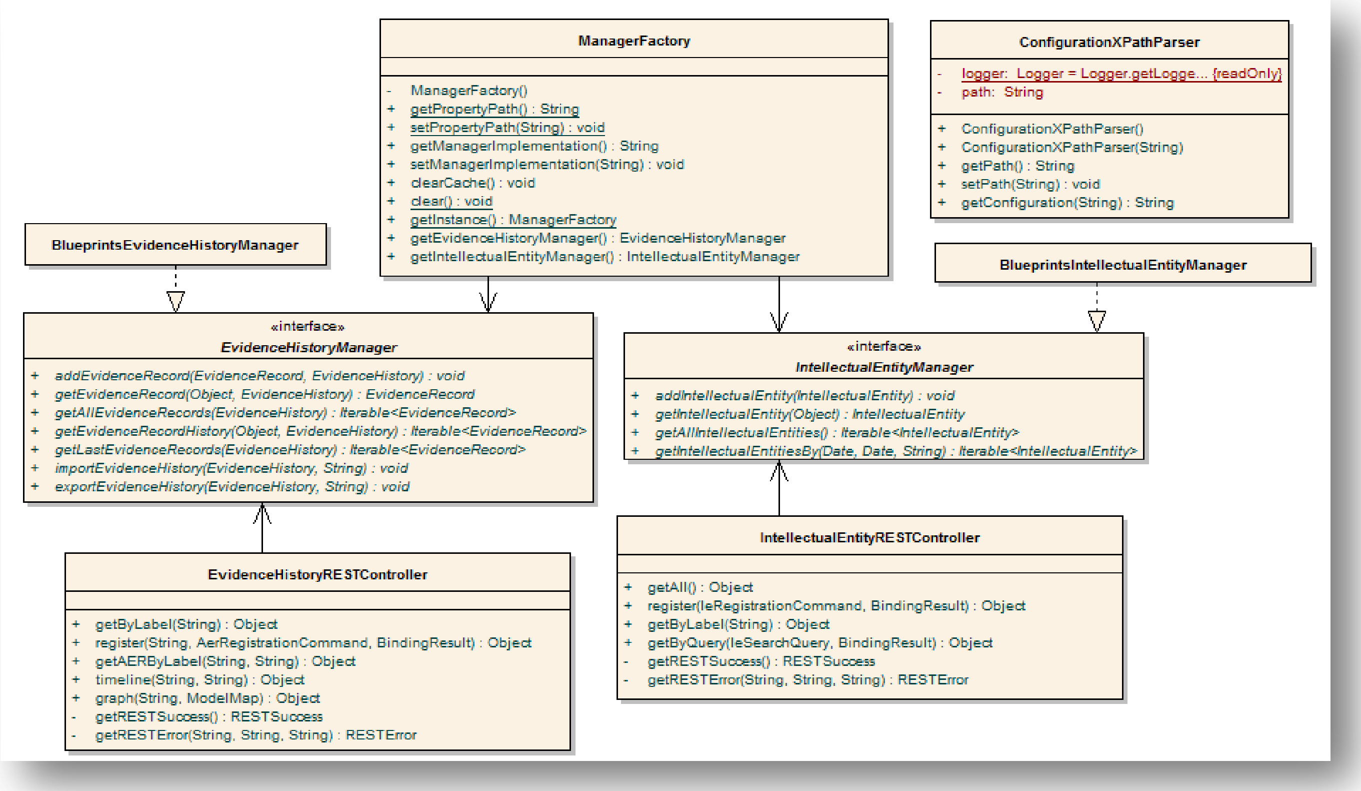

ManagerFactory
- ManagerFactory()
+ getPropertyPath() : String
+ setPropertyPath(String) : void
+ getManagerImplementation() : String
+ setManagerImplementation(String) : void
+ clearCache() : void
+ clear() : void
+ getInstance() : ManagerFactory
+ getEvidenceHistoryManager() : EvidenceHistoryManager
+ getIntellectualEntityManager() : IntellectualEntityManager
ConfigurationXPathParser
- logger: Logger = Logger.getLogge... {readOnly}
- path: String
+ ConfigurationXPathParser()
+ ConfigurationXPathParser(String)
+ getPath() : String
+ setPath(String) : void
+ getConfiguration(String) : String
BlueprintsEvidenceHistoryManager
BlueprintsIntellectualEntityManager
«interface»
EvidenceHistoryManager
+ addEvidenceRecord(EvidenceRecord, EvidenceHistory) : void
+ getEvidenceRecord(Object, EvidenceHistory) : EvidenceRecord
+ getAllEvidenceRecords(EvidenceHistory) : Iterable<EvidenceRecord>
+ getEvidenceRecordHistory(Object, EvidenceHistory) : Iterable<EvidenceRecord>
+ getLastEvidenceRecords(EvidenceHistory) : Iterable<EvidenceRecord>
+ importEvidenceHistory(EvidenceHistory, String) : void
+ exportEvidenceHistory(EvidenceHistory, String) : void
«interface»
IntellectualEntityManager
+ addIntellectualEntity(IntellectualEntity) : void
+ getIntellectualEntity(Object) : IntellectualEntity
+ getAllIntellectualEntities() : Iterable<IntellectualEntity>
+ getIntellectualEntitiesBy(Date, Date, String) : Iterable<IntellectualEntity>
EvidenceHistoryRESTController
+ getByLabel(String) : Object
+ register(String, AerRegistrationCommand, BindingResult) : Object
+ getAERByLabel(String, String) : Object
+ timeline(String, String) : Object
+ graph(String, ModelMap) : Object
- getRESTSuccess() : RESTSuccess
- getRESTError(String, String, String) : RESTError
IntellectualEntityRESTController
+ getAll() : Object
+ register(IeRegistrationCommand, BindingResult) : Object
+ getByLabel(String) : Object
+ getByQuery(IeSearchQuery, BindingResult) : Object
- getRESTSuccess() : RESTSuccess
- getRESTError(String, String, String) : RESTError

Figure 5 SCIDIP-ES HAPPI Interfaces

Figure 5 highlights the most important classes behind the implementation of SCIDIP-ES HAPPI:

i) ManagerFactory and ConfigurationXPathParser – two implementations that are responsible for instantiating SCIDIP-ES HAPPI and EvidenceHistoryManager based on property settings specified in META-INF/conf.properties file;

ii) BlueprintsEvidenceHistoryManager and BlueprintsIntellectualEntityManager – two implementations of interfaces (i.e. EvidenceHistoryManager and IntellectualEntityManager) based on Thinkerpop Blueprints and consequently supporting any Blueprints Graph DB;

iii) EvidenceHistoryRESTController and IntellectualEntityRESTController – two implementations of interfaces (i.e. EvidenceHistoryManager and IntellectualEntityManager) for supporting REST style.

SCIDIP-ES HAPPI uses a GraphDB (i.e. OrientDB) for the persistence of data. The instance of GraphDB, namely OrientDB, may be replaced with a different one (e.g. Neo4J): this flexibility is guaranteed by the adoption of Tinkerpop Blueprints. The Figure 6 below shows the persistence layer of SCIDIP-ES.

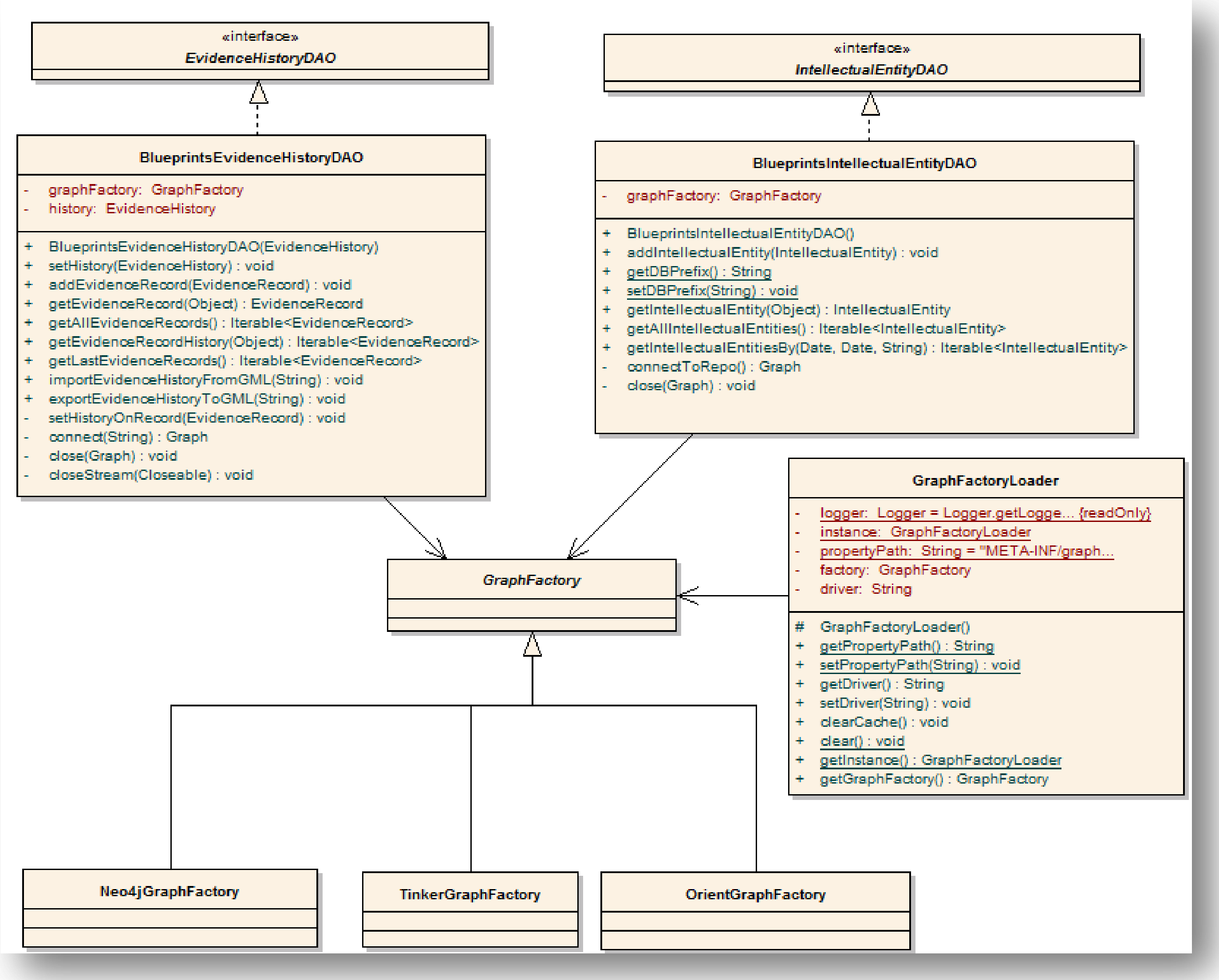

«interface»
EvidenceHistoryDAO
«interface»
IntellectualEntityDAO
BlueprintsEvidenceHistoryDAO
- graphFactory: GraphFactory
- history: EvidenceHistory
+ BlueprintsEvidenceHistoryDAO(EvidenceHistory)
+ setHistory(EvidenceHistory) : void
+ addEvidenceRecord(EvidenceRecord) : void
+ getEvidenceRecord(Object) : EvidenceRecord
+ getAllEvidenceRecords() : Iterable<EvidenceRecord>
+ getEvidenceRecordHistory(Object) : Iterable<EvidenceRecord>
+ getLastEvidenceRecords() : Iterable<EvidenceRecord>
+ importEvidenceHistoryFromGML(String) : void
+ exportEvidenceHistoryToGML(String) : void
- setHistoryOnRecord(EvidenceRecord) : void
- connect(String) : Graph
- close(Graph) : void
- closeStream(Closeable) : void
BlueprintsIntellectualEntityDAO
- graphFactory: GraphFactory
+ BlueprintsIntellectualEntityDAO()
+ addIntellectualEntity(IntellectualEntity) : void
+ getDBPrefix() : String
+ setDBPrefix(String) : void
+ getIntellectualEntity(Object) : IntellectualEntity
+ getAllIntellectualEntities() : Iterable<IntellectualEntity>
+ getIntellectualEntitiesBy(Date, Date, String) : Iterable<IntellectualEntity>
- connectToRepo() : Graph
- close(Graph) : void
GraphFactoryLoader
- logger: Logger = Logger.getLogge... {readOnly}
- instance: GraphFactoryLoader
- propertyPath: String = "META-INF/graph...
- factory: GraphFactory
- driver: String
# GraphFactoryLoader()
+ getPropertyPath() : String
+ setPropertyPath(String) : void
+ getDriver() : String
+ setDriver(String) : void
+ clearCache() : void
+ clear() : void
+ getInstance() : GraphFactoryLoader
+ getGraphFactory() : GraphFactory
GraphFactory
Neo4jGraphFactory
TinkerGraphFactory
OrientGraphFactory

Figure 6 SCIDIP-ES HAPPI Persistence Layer

For each SCIDIP-ES HAPPI interface (i.e. IntellectualEntityManager and EvidenceHistoryManager) is defined a DAO interface that is implemented by using Thinkerpop Blueprints. GraphFactoryLoader is responsible for instantiating the GraphFactory that maybe, based on configuration settings:

- a OrientGraphFactory - for supporting the Orient DB;
- a Neo4jGraphFactory - for supporting the Neo4j DB.

By extending the GraphFactory class, it is possible to allow the integration of any new Blueprints-compliant Graph DB.

The software is delivered as a Java Web Archive (i.e. happi-server.war) for the server side deployment. SCIDIP-ES HAPPI provides a basic web GUI that exposes functionality supported by its interfaces. Details on its public APIs and information model are provided in Section 5.

3 Installation Guide

This section provides the installation guide and configuration details for deploying and starting SCIDIP-ES HAPPI.

3.1 Overview

3.2 Prerequisites

3.2.1 Software prerequisites

According to [D32.2], development of SCIDIP-ES HAPPI, as well as any other software of the SCIDIP-ES e-Infrastructure, is based on Java technology and underlying common framework (e.g. communication protocols, common libraries).

Current release of SCIDIP-ES HAPPI has been built on top of the following SCIDIP-ES e-Infrastructure common constraints:

- **Oracle JDK1.7.0_45**: currently it is testing new release on JDK7;
- **MAVEN 4.0.0**;
- **Tomcat 7.0.47**: currently it is testing new release on Tomcat7.

Specific prerequisites are:

- **OrientDB 1.5.0 Graph Edition**: that is the graph database adopted by default in SCIDIP-ES HAPPI.
  Available on
  http://www.orientdb.org/portal/function/portal/download/unknown@unknown.com/-/-/-/-/-/orientdb-graphed-1.5.0.zip.

3.2.2 Hardware prerequisites

SCIDIP-ES HAPPI has no specific hardware prerequisites.

3.3 OSS/COTS Installation

N/A

3.4 License Information and Terms of Use

SCIDIP-ES HAPPI (as most of the services and toolkits in the SCIDIP-ES project) is licensed under the Apache License, Version 2.0 (the "License"). A copy of the License could be obtained at: http://www.apache.org/licenses/LICENSE-2.0. Copyright reserved by Engineering Ingegneria Informatica S.p.A.

3.5 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/toolkits/authenticity/trunk

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/toolkits/authenticity/

3.6 SCIDIP-ES HAPPI Installation

Assuming that JDK1.7.0_45 (or greater) and Tomcat 7.0.47 (or greater) are already deployed, installation of SCIDIP-ES HAPPI covers these 3 main steps:

1. **Installation and Configuration of OrientDB 1.5.0 Graph edition.** After the installation, it is necessary to add and enable an orientdb user as follows:
   a. Add user “orient” on **orientdb-server-config.xml** (under config folder of OrientDB installation): the deployer can choose any user name and password. The following credentials are the default ones used by the HAPPI server side component:

```
<users>
     <user resources="*" password="[GENERATED]" name="root"/>
     <user resources="*" password="s3cret" name="orient"/>
 </users>
```

2. **Deployment and Configuration of SCIDIP-ES HAPPI WAR package in Tomcat**:

   a. Add happi-server-1.5.0.war under the webapps folder of Tomcat and restart Tomcat;
   b. Check that the happi-server is properly launched by accessing http://localhost:8080/happi-server-1.5.0 on a web browser. After the home page is shown, stop Tomcat;
   c. Revise the graph.properties under the folder (TOMCAT_HOME/webapps/happi-server-1.5.0 /WEB-INF/classes/META-INF) to configure user credentials and the OrientDB installation path as follows:

```
driver = orient
# driver = neo4j
user = orient
password = s3cret
host = localhost
# Property "directory" is for neo4j
# directory = /opt/neo4j-community-1.9.3/
```

   d. HAPPI 1.5.0 supports Neo4j graph database. To use this, uncomment the row “driver = neo4j” and property “directory”. In this case, orient driver and host property have to be commented out.

3. **Restart Tomcat and use SCIDIP-ES HAPPI Server at http://localhost:8080/happi-server-1.5.0 from your web browser** (The HAPPI GUI has been optimised for Google Chrome and Firefox). That's all!

This new release 1.5.0 supports encrypted password for modifying data when accessing the web application. Administrator can change the password by using the utility java.eu.scidipes.api.utils.Encoder available on the happi-server. The encryption is guaranteed by the spring-security libraries (i.e. spring-security-*-3.1.4.RELEASE.jar). The code for encrypting the password for the default installation (i.e. admin/admin) is shown below:

```
package eu.scidipes.api.utils;

import org.springframework.security.crypto.password.StandardPasswordEncoder;

public class Encoder {

     public static void main(String[] args) {

           StandardPasswordEncoder encoder = new StandardPasswordEncoder();

           System.out.println(encoder.encode("admin"));

           }

}
```

The old password has to be replaced with the encrypted one in TOMCAT_HOME/webapps/happi-server-1.5.0-snapshot/WEB-INF/spring/spring-security.xml as shown below:

```
<authentication-manager>

  <authentication-provider>

    <password-encoder ref="encoder" />

       <user-service>

       <user                                                          name="admin"
password="a79bbf0f0b2bf21210c6f0f78736d9d0171566dac2358956956a3a4b28c1232e7
dbd6aaa1ff1e4da" authorities="admin" />

       </user-service>

  </authentication-provider>

</authentication-manager>
```

3.7 Uninstallation

To uninstall SCIDIP-ES HAPPI requires undeploying it from Tomcat and uninstalling the OrientDB.

4 Using SCIDIP-ES SCIDIP-ES HAPPI

4.1 Getting Started

In line with the development plans defined in [D32.1] and [D32.2] SCIDIP-ES HAPPI currently provides:

- Models for specifying the content of an Authenticity Evidence Record, which represents part of OAIS Preservation Description Information;
- Definition and implementation of two SCIDIP-ES HAPPI interfaces (i.e. EvidenceManager and IntellectualEntityManager, see Section 5) and related methods for supporting archive managers in capturing and managing the Evidence History, as well as registration/search/browse of Intellectual Entities.

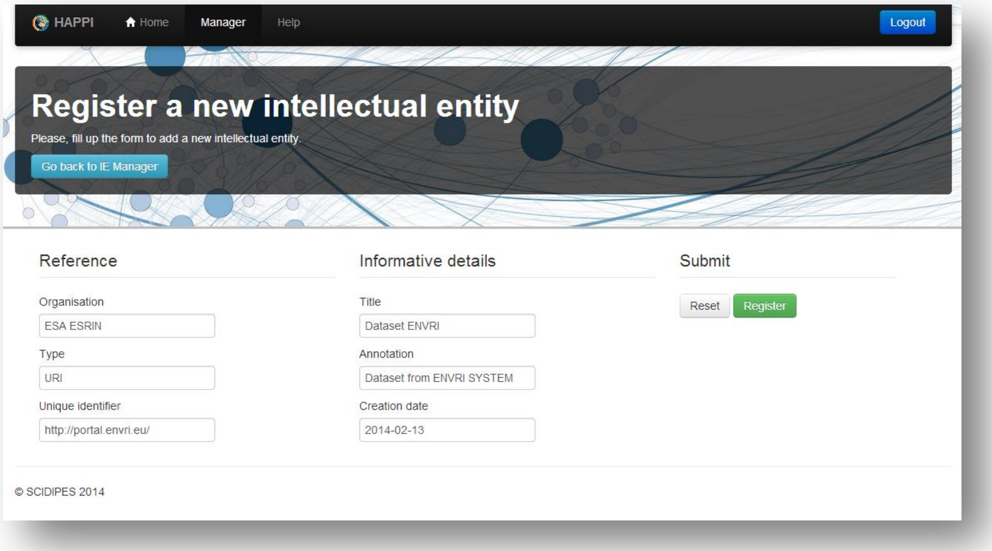
The following sub-sections describe the main usage scenarios supported by SCIDIP-ES HAPPI.

4.2 Registering Intellectual Entity

All operations start by identifying the Intellectual Entity: it is the abstraction of each instance of digital representation. For this purpose, the IntellectualEntityManager interface provides the method `addIntellectualEntity`.

By registering the Intellectual Entity, HAPPI creates a dedicated graph database for the specified intellectual entity which in turn allows information about its own instances (digital representations) and related evidence records to be gathered.

Figure 7 below shows the required details for creating and registering a new intellectual entity via HAPPI Web Application.



Register a new intellectual entity

Please, fill up the form to add a new intellectual entity.

Reference

Organisation: ESA ESRIN

Type: URI

Unique identifier: http://portal.envri.eu/

Informative details

Title: Dataset ENVRI

Annotation: Dataset from ENVRI SYSTEM

Creation date: 2014-02-13

Submit

© SCIDIPES 2014

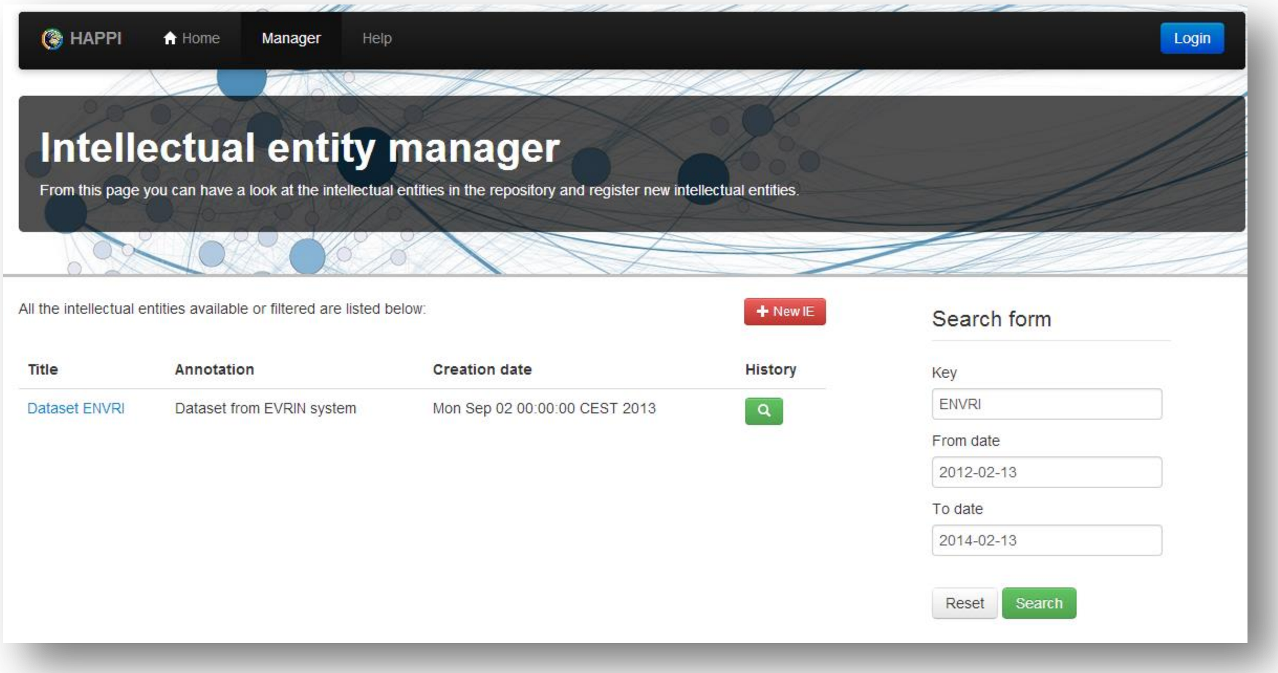
Figure 7 Creation and registration of a new intellectual entity

4.3 Search and Browse Intellectual Entity

Intellectual Entities may be searched by using mainly:

- Keywords within the title or annotation;
- Time period within the timeline of an intellectual entity.

This functionality is provided by the IntellectualEntityManager interface through the method `getIntellectualEntitiesBy`. The HAPPI Web Application allows users to search and browse intellectual entity by using this feature. Here below a screenshot.



Intellectual entity manager

From this page you can have a look at the intellectual entities in the repository and register new intellectual entities.

All the intellectual entities available or filtered are listed below:

Title	Annotation	Creation date	History
Dataset ENVRI	Dataset from EVRIN system	Mon Sep 02 00:00:00 CEST 2013	🔍

Search form

Key: ENVRI

From date: 2012-02-13

To date: 2014-02-13

Figure 8 Search and browse intellectual entity via HAPPI web application

4.4 Capturing Evidence History

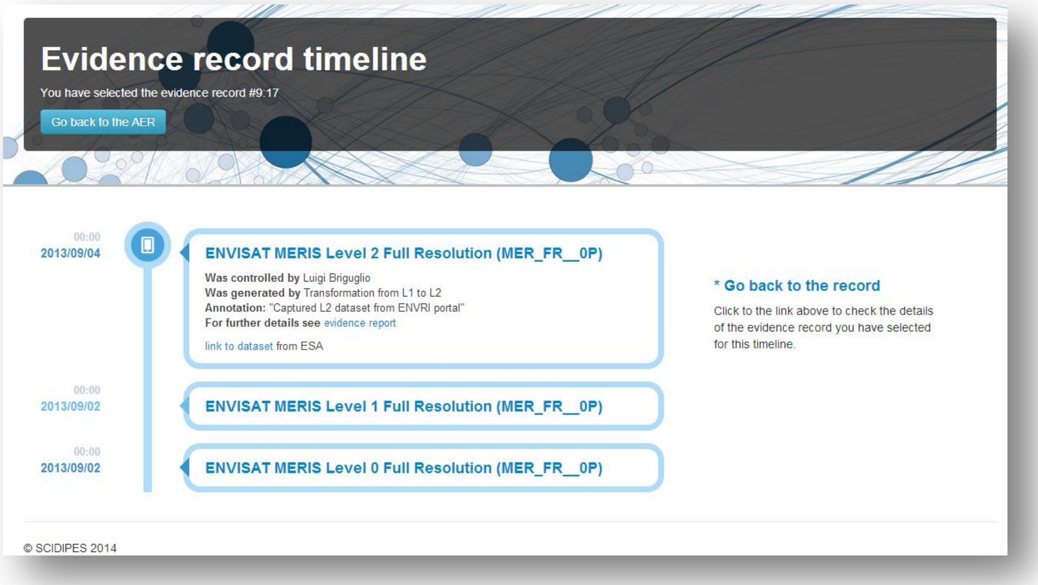
As stated in section 2, each transformation and event that a digital object has undergone is documented by an Evidence Record. The whole lifecycle history of the digital object is consequently collected as a set of Evidence Records within an Evidence History object.

For capturing the Evidence History, the EvidenceManager interface provides the method `addEvidenceRecord`. The method is invoked for each event and transformation in order to capture the set of records which constitutes the evidence history of the digital object.

4.5 Browsing Timeline

Browsing timeline of events/transformations on a digital object, starting from a specific transformation, allows users to visualise the change history of an intellectual entity. The EvidenceManager interface provides the method `getEvidenceRecordHistory` to support this requirement.

The HAPPI Web Application uses this functionality to let user browse the history of an intellectual entity since its creation. Two modalities are provided for representing the history: timeline (Figure 9) and graph (Figure 10) views.



Evidence record timeline

You have selected the evidence record #9:17

00:00
2013/09/04

ENVISAT MERIS Level 2 Full Resolution (MER_FR__0P)

Was controlled by Luigi Briguglio
Was generated by Transformation from L1 to L2
Annotation: "Captured L2 dataset from ENVRI portal"
For further details see evidence report

link to dataset from ESA

00:00
2013/09/02

ENVISAT MERIS Level 1 Full Resolution (MER_FR__0P)

00:00
2013/09/02

ENVISAT MERIS Level 0 Full Resolution (MER_FR__0P)

* Go back to the record

Click to the link above to check the details of the evidence record you have selected for this timeline.

© SCIDIPES 2014

Figure 9 Intellectual entity history: timeline view

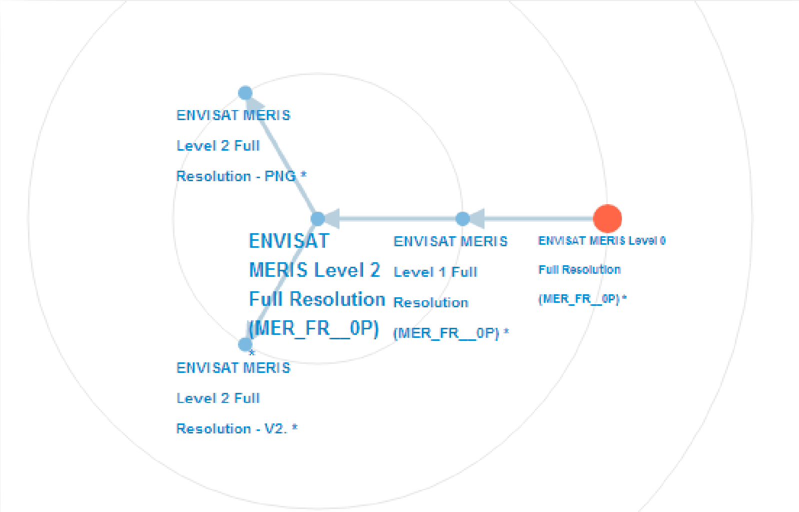

ENVISAT MERIS Level 2 Full Resolution - PNG *
ENVISAT MERIS Level 2 Full Resolution (MER_FR__0P) *
ENVISAT MERIS Level 1 Full Resolution (MER_FR__0P) *
ENVISAT MERIS Level 0 Full Resolution (MER_FR__0P) *
ENVISAT MERIS Level 2 Full Resolution - V2. *

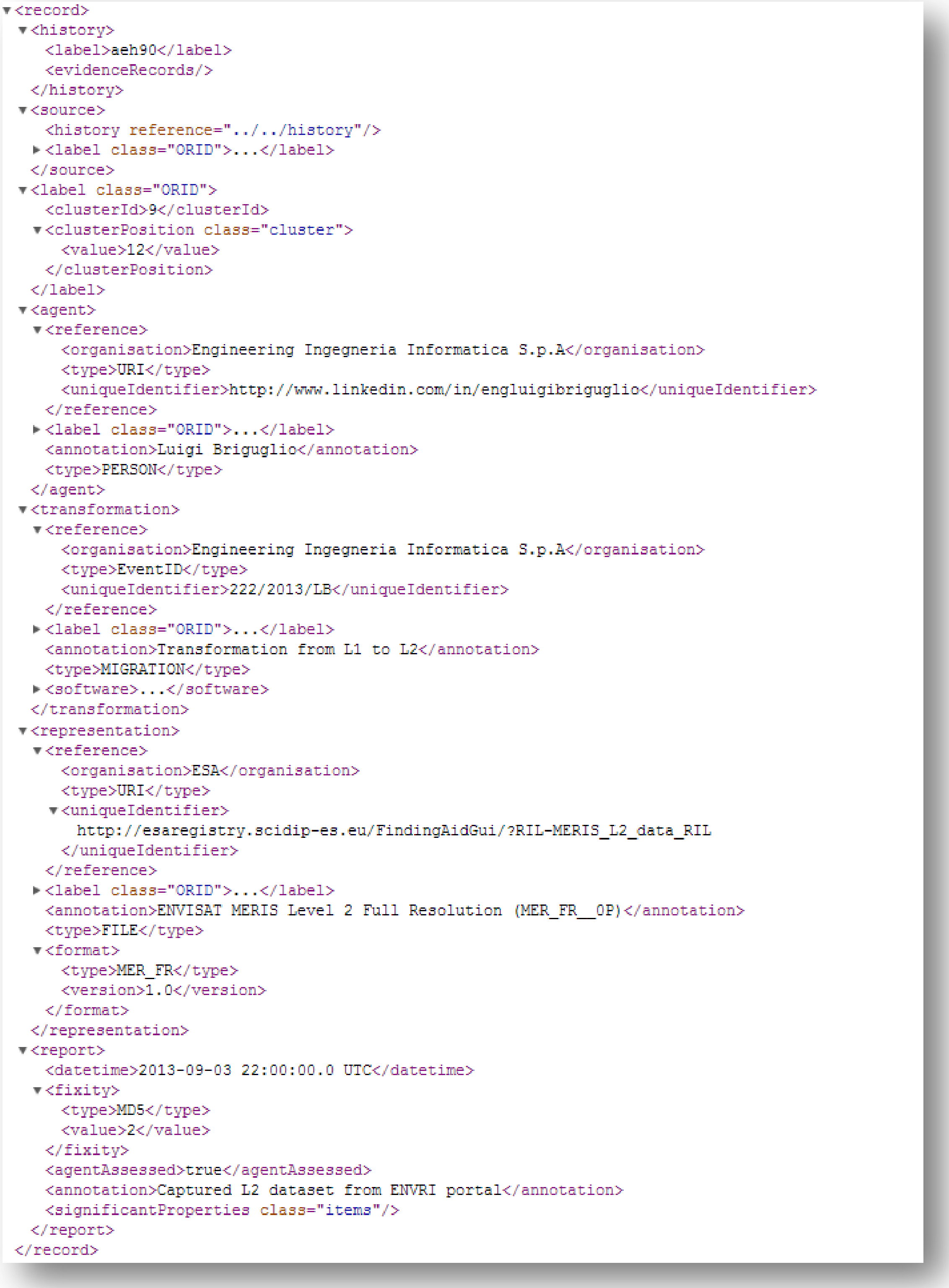
Figure 10 Intellectual entity history: graph view

4.6 *Importing/Exporting Evidence History*

SCIDIP-ES HAPPI allows users to import/export the Evidence History of a digital object from/to a standard XML file (based on graph xml schema).

As stated in section 2, Evidence History is the set of information describing all the transformations and events a digital object has undergone since its creation.

The import/export functionality ensures different archives involved in the preservation process of the whole lifecycle of the digital object can exchange evidence information about it: in practice this could occur for any transfer of custody. The EvidenceManager interface provides the `exportEvidenceHistory` and `importEvidenceHistory` methods to support the export and import of Evidence History. An example of exported XML file is shown in Figure 11.



```
▼<record>
  ▼<history>
     <label>aeh90</label>
     <evidenceRecords/>
   </history>
  ▼<source>
     <history reference="../../history"/>
    ▶<label class="ORID">...</label>
   </source>
  ▼<label class="ORID">
     <clusterId>9</clusterId>
    ▼<clusterPosition class="cluster">
       <value>12</value>
     </clusterPosition>
   </label>
  ▼<agent>
    ▼<reference>
       <organisation>Engineering Ingegneria Informatica S.p.A</organisation>
       <type>URI</type>
       <uniqueIdentifier>http://www.linkedin.com/in/engluigibriguglio</uniqueIdentifier>
     </reference>
    ▶<label class="ORID">...</label>
     <annotation>Luigi Briguglio</annotation>
     <type>PERSON</type>
   </agent>
  ▼<transformation>
    ▼<reference>
       <organisation>Engineering Ingegneria Informatica S.p.A</organisation>
       <type>EventID</type>
       <uniqueIdentifier>222/2013/LB</uniqueIdentifier>
     </reference>
    ▶<label class="ORID">...</label>
     <annotation>Transformation from L1 to L2</annotation>
     <type>MIGRATION</type>
    ▶<software>...</software>
   </transformation>
  ▼<representation>
    ▼<reference>
       <organisation>ESA</organisation>
       <type>URI</type>
      ▼<uniqueIdentifier>
         http://esaregistry.scidip-es.eu/FindingAidGui/?RIL-MERIS_L2_data_RIL
       </uniqueIdentifier>
     </reference>
    ▶<label class="ORID">...</label>
     <annotation>ENVISAT MERIS Level 2 Full Resolution (MER_FR__0P)</annotation>
     <type>FILE</type>
    ▼<format>
       <type>MER_FR</type>
       <version>1.0</version>
     </format>
   </representation>
  ▼<report>
     <datetime>2013-09-03 22:00:00.0 UTC</datetime>
    ▼<fixity>
       <type>MD5</type>
       <value>2</value>
     </fixity>
     <agentAssessed>true</agentAssessed>
     <annotation>Captured L2 dataset from ENVRI portal</annotation>
     <significantProperties class="items"/>
   </report>
 </record>
```

Figure 11 Example of an evidence record exported in XML

5 Reference Manual

5.1 *Keyboard shortcuts*

N/A

5.2 *Command-line commands*

N/A

5.3 *Public APIs*

The SCIDIP-ES HAPPI provides its functionality through the IntellectualEntityManager and the EvidenceManager interfaces. The interface methods are described in the tables below.

5.3.1 IntellectualEntityManager

Method Summary	

Void	`addIntellectualEntity(IntellectualEntity ie)`
	It stores the intellectual entity ie in the repository and creates a new database associated with the evidence history of the ie.
	REST syntax: POST /service/ie/{ie}

IntellectualEntity	`getIntellectualEntity(String label)` Given a string label, it returns the intellectual entity with that label. REST syntax: GET /service/ie/{label}/
Iterable<IntellectualEntity>	`getAllIntellectualEntities()` It searches all the intellectual entity in the archive. REST syntax: GET /service/ie/
Iterable<IntellectualEntity>	`getIntellectualEntitiesBy(Date startDate, Date endDate, String key)` It returns all the intellectual entities created between dates startDate and endDate with the keyword key in the annotation or in the title. REST syntax: GET /service/ie/{query}/

Table 1 SCIDIP-ES HAPPI: IntellectualEntityManager interface

5.3.2 EvidenceManager

Method Summary	
Void	`addEvidenceRecord(EvidenceRecord evidenceRecord, EvidenceHistory eh` Add new evidence for a change occurred on a representation of intellectual entity. REST syntax: POST /service/aeh/{aehLabel}/aer/{aerLabel}/

EvidenceRecord	`getEvidenceRecord(Object label, EvidenceHistory eh)`
	Get evidence for a change occurred on a representation of intellectual entity, based on specific label assigned by the graphdb.
	REST syntax: GET /service/aeh/{aehLabel}/aer/{aerLabel}

Iterable<EvidenceRecord>	`getAllEvidenceRecords(EvidenceHistory eh)` Get all the evidence records of an evidence history. REST syntax: GET /service/aeh/{aehLabel}/
Iterable<EvidenceRecord>	`getEvidenceRecordHistory(Object label, EvidenceHistory eh)` Get part of evidence history by starting from the evidence record identified by label. REST syntax: GET /service/aeh/{aehLabel}/aer/{aerLabel}/timeline
Iterable<EvidenceRecord>	`getLastEvidenceRecords(EvidenceHistory eh)` Return the list of last evidence records. Indeed, in case of branch, more than one evidence records are “last records”.
Void	`importEvidenceHistory(EvidenceHistory eh, String gmlFileName)` Imports the specified evidence history from graph XML file gmlFileName.
Void	`exportEvidenceHistory(EvidenceHistory eh, String gmlFileName)` Exports the evidence history eh of the digital object to graph XML file gmlFileName.

Table 2 SCIDIP-ES HAPPI: EvidenceManager interface

6 Common problems and their correction

Further tests, planned to be extended in the next few months, may uncover potential issues with HAPPI and further details will be provided in D24.2.

The following FAQ is commonly encountered from deployments and tests:

FAQ #	Issue Description	Solution
1	HAPPI provides message “could not connect database”	• Check database server is up and running at http://localhost:2480/studio. • Try to login with user “orient” credentials • If first two steps work fine, check if ports 2424 and 2480 are accessible to tomcat.

Annex A. References

[Briguglio2013] Luigi Briguglio, Silvio Salza, Maria Guercio: Preserving Authenticity Evidence to Assess Provenance and Integrity of Digital Resources. ECLAP 2013: pp.66-77 Springer LNCS volume 7990 (2013) - http://dx.doi.org/10.1007/978-3-642-40050-6_7

[Guercio-Salza] Guercio, M., Salza, S.: Managing Authenticity through the Digital Resources Lifecycles, In Digital Libraries and Archives, 8th Italian Res. Conf., IRCDL 2012, Bari, February 2012. CCIS vol.354, pp.249-260 Springer (2013)

[D12.1] User Requirements – available at http://www.scidip-es.eu/scidip-es/deliverables/

[D32.1] Research on Services and Toolkits - available at http://www.scidip-es.eu/scidip-es/deliverables/

[D32.2] Generic Services/Toolkits and Robustness Research Report and Plan - available at http://www.scidip-es.eu/scidip-es/deliverables/

[OPM1.1] Open Provenance Model Core Specification v1.1 - available at http://eprints.soton.ac.uk/271449/1/opm.pdf

[OrientDB] OrientDB – official website http://www.orientdb.org

[Premis2.2] PREMIS Data Dictionary for Preservation Metadata v2.0 http://www.loc.gov/standards/premis/v2/premis-2-2.pdf

Annex B. Figures and Tables

B.1. List of Figures

B.2. List of Tables

Annex C. Terminology

ACRONYM	DESCRIPTION
AIP	Archival Information Package
ARK	Archival Resource Key
CDMI	Cloud Management Interface
DOI	Digital Object Identifier
ES	Earth Science
GIS	Gap Identification Service
KB	Knowledge Base
OS	Orchestration Service
OWL	Web Ontology Language
PI	Persistent Identifier
PNM	Preservation Network Model

PURL	Persistent Uniform Resource Locator
RDF	Resource Description Framework
RepInfo	Representation Information
SNIA	Storage Networking Industry Association
SWKM	Semantic Web Knowledge Middleware
VM	Virtual Machine
WP	Work Package
XAM	eXtensible Access Method
XML	eXtensible Mark-up Language