VRE infrastructure and Services – Final Version

Workpackage	**5**	**VRE Infrastructure and Services Design and Development**
Task		
Author (s)	Pedro Gonçalves	Terradue
	Serena Avolio	ACS
	Raul Palma	PSNC
	Simone Mantovani	MEEO
	José Manuel Gómez	ESI
Reviewer (s)	Simone Mantovani	MEEO
	Ugo Di Giammatteo	ACS
Approver (s)	Cristiano Silvagni	ESA
	Pedro Gonçalves	Terradue
Authorizer	Mirko Albani	ESA
Document Identifier	EVER-EST DEL WP5-D5.11	
Dissemination Level	Public	
Status	Approved by the EC	
Version	1.1	
Date of Issue	14 December 2018	

Document Log

Date	Author	Changes	Version	Status
31/03/2017	Pedro Gonçalves	Initial version based on D5.10	0.1	Draft
04/04/2017	Serena Avolio	Update on the Common Services	0.2	Draft
06/04/2017	Andrés Garcia, Raul Palma	Update on RO-related services and Taverna Server	0.3	Draft
10/04/2017	Pedro Gonçalves	Update on the Cloud Controller	0.4	Draft
16/05/2017	Pedro Gonçalves	Update with reviewers comments	0.5	Draft
17/05/2017	Pedro Gonçalves	Final review	1.0	Draft to be approved by the EC
14/12/2018	Cristiano Silvagni	Document status	1.1	Approved by the EC

Table of Contents

List of Figures

Definitions and Acronyms

Acronym	Description
CMS	Content Management System
COTS	Commercial Off-The-Shelf
EO	Earth Observation
GUI	Graphical User Interface
RODL	Research Object Digital Library
RO	Research Object
UI	User Interface
VC	Visual Components
VM	Virtual Machine
VRC	Virtual Research Community
VRE	Virtual Research Environment
WP	Work Package

Applicable Documents

Document ID	Document Title
EVER-EST DEL WP5-D5.2	Technical Note on Common Services, Intermediate version
EVER-EST DEL WP5-D5.3	Technical Note on Digital Information and E-Collaboration Services, Intermediate version
EVER-EST DEL WP5-D5.4	Technical Note on e-Research Applications Services, Intermediate version
EVER-EST DEL WP5-D5.5	Technical Note on e-Learning Services, Intermediate version
EVER-EST DEL WP5-D5.6	Technical Note on Common Services, Final version
EVER-EST DEL WP5-D5.7	Technical Note on Digital Information and E-Collaboration Services, Final version
EVER-EST DEL WP5-D5.8	Technical Note on e-Research Applications Services, Final version
EVER-EST DEL WP5-D5.9	Technical Note on e-Learning Services, Final version

1 Introduction

1.1 Purpose of the document

This report accompanies and describes the delivery of the final version of the VRE infrastructure and services. Its main purpose is to list the software components selected for the consolidated design and the development of the EVER-EST Virtual Research Environment (VRE).

1.2 Background

As detailed within the proposal, a consistent part of the infrastructure is based on modules and experience derived from previous Earth Science EU projects including, among others, SCIDIP-ES, GEOWOW and EarthServer. D5.1 provides details concerning the EVER-EST VRE architecture and final set of corresponding software components. This document joins those elements and lists the software originating from the following tasks:

- Task 5.2 - Common services;
- Task 5.3 - Digital Information and E-Collaboration Services;
- Task 5.4 - E-Research Application Services;
- Task 5.5 - E-Learning application services.

1.3 Document structure

This introductory chapter aims to provide key information to readers that do not belong to the EVER-EST technical team in order to provide the context and placement of this document in the overall WP5 activities. For a more general perspective the reading of D5.1 is recommended.

Chapter 2 provides a high-level overview of the EVER-EST VRE architecture focusing and lists the adopted software.

2 EVER-EST VRE Software

2.1 EVER-EST VRE architecture

Figure 1 provides a high-level overview of the EVER-EST VRE architecture.

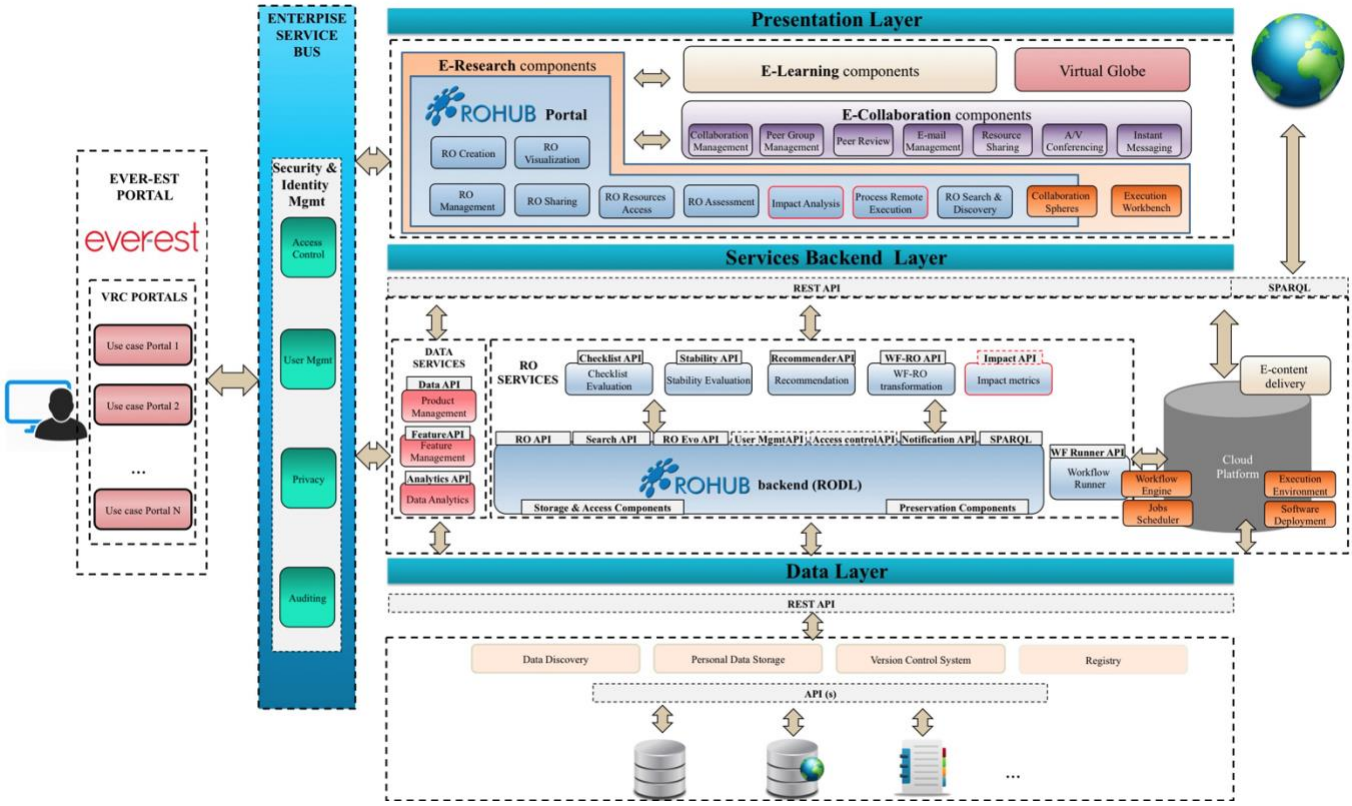

Figure 1 – EVER-EST VRE architecture

The architecture reflects the organization of functions across the various layers and with regards to the core element of the design the following can be identified:

1. **Presentation Layer** - on the top part of the architectural diagram - contains the core elements and technologies that shall guarantee the availability of those services and functions, which shall be directly used by the different communities. It includes the ROHub, the e-collaboration, e-learning and e-research services along with the mechanisms for Earth Science data discovery.
2. **Service Layer** - in the central part of the architectural diagram - provides both generic VRE services and Earth Science specific services. These components represent the reasoning engine of the e-infrastructure and actually orchestrate and manage the services available to the VRE final users.
3. **Data Layer** - bottom part of the design - references the data holdings made available to the Virtual Research Communities (VRCs): data is linked and proper means are provided, where feasible, to access it from the VRE. As a default setting, data will not be copied or duplicated, but will continue to reside on the provider's local servers unless it is directly retrieved by the user.

Included in the Presentation Layer is also the VRE portal that is the main access the components giving shortcut to each VRC user interface. While some services and functions can be seen as common and transversal to all the VRCs, a set of features and functions are specific for each of the communities and required an ad-hoc design.

The implementation of the EVER-EST VRE was divided in 4 main tasks:

- Task 5.2 - Common services;
- Task 5.3 - Digital Information and E-Collaboration Services;
- Task 5.4 - E-Research Application Services;
- Task 5.5 - E-Learning application services.

Each of these tasks focused on different software components and addressed development and configuration tasks at different levels. The following sections list the software used for the definition of the EVER-EST VRE according to those tasks.

2.2 Common services

The Common Services category represents the family of services, which are used by different elements of the EVER-EST infrastructure. They refer to the category of services that will guarantee the correct functioning between the various infrastructure components and between the infrastructure and the final users.

The common services implemented spread among user authentication, API protection, account provisioning, identity administration, security, message routing, message transformation, auditing , data discovery, data access and data Storage.

These services can be grouped by macro-topics coupled with software components. Therefore we have: services related to the identity management, services related to the message mediation, services related to the auditing and services related to data management.

Detailed information on Common Services can be found on the [EVER-EST DEL WP5-D5.2] document and on the [EVER-EST DEL WP5-D5.6] document.

2.2.1 Identity management services

The services related to the identity management are supported by the WSO2 Identity and Access Management Server[1]. The WSO2 Identity Server provides secure Identity Management for the EVER-EST Web Application services and APIs by managing identity and entitlements of the users. It operates through different authentication and authorization protocols as OpenId Connect, OAuth2 and SAML20 ensuring a SSO scenario. It is a COTS configured and customized for the EVER-EST environment.

Message Mediation Services

The mediation services requested by the EVER-EST infrastructure are supported by WSO2 Enterprise Service Bus[2]. The process of routing requests to correct service provider and the process of transforming message formats between different applications are centralized and offered to the end user of the VRE Portal using the mediation of the ESB that acts as a middleware to integrate the ROHub and Taverna functionalities with the VRE portal.

WSO2 Enterprise Service Bus is a COTS configured and customized for the EVER-EST environment.

[1] https://github.com/wso2/product-is

[2] https://github.com/wso2/product-esb

Auditing Services

The auditing services requested by the EVER-EST infrastructure are related to data collection, data aggregation and data analytics. These services are supported by WSO2 Data Analytic Server[3]. The Data Analytics Server offers data analytics functionalities for the EVER-EST environment.

WSO2 Data Analytic Server is a COTS configured and customized for EVER-EST.

Data Management

The services related to personal data management are supported by SeaFile[4]. Seafile offers a repository for the personal data storage. Seafile has a web interface, web API and a desktop client. It guarantees file synchronization, version control, public link sharing, desktop client and web API.

2.3 Digital information and e-collaboration services

2.3.1 EVER-EST and VRC portals

The EVER-EST portal and the VRC portals are developed using Django [5] (source code available at: https://github.com/django/django) and the Django-CMS [6] Content Management System technologies (source code available at: https://github.com/divio/django-cms).

Information about the portals can be found in chapter 2 of the [EVER-EST DEL WP5-D5.3] document and of the [EVER-EST DEL WP5-D5.7] document. The VRE portals source code is available at:

- https://github.com/ec-everest/vre-portal/

Visual Components

The EVER-EST Visual Components (VCs) constitute the core of the EVER-EST Portal and VRC Portals. The VCs are developed as independent objects that could be added to a web page by adding just a few lines of code. The client side of the VC is developed using open web technologies (such as HTML5 standard, Javascript, jQuery, Bootstrap 3 Framework, and CSS 3).

The VCs implemented are:

- Discovery Visual Component, it provides datasets and data discovery functionalities via OpenSearch queries on OpenSearch enabled repositories;
- RO Manager Visual Component, it provides RO creation, visualization and editing functionalities;
- Upload to Seafile Visual Component, it provides upload to Seafile and basic file management functionalities within the VRC portal;
- Virtual Globe Visual Component, it provides 2D/3D map viewer functionalities to visualize EO and non-EO resources;
- WPS Manager Visual Component, it allows WPS retrieval, execution and input/output management.

Information about the Digital Information Services and the Visual Components can be found in chapter 3 of the [EVER-EST DEL WP5-D5.7] document.

[3] https://github.com/wso2/product-das

[4] https://github.com/seafile

[5] https://www.djangoproject.com/

[6] https://www.django-cms.org/en/

The VC source codes are available at:

- https://github.com/ec-everest/vre-portal/

E-Collaboration Services

Django, the Content Management System (CMS) on which the EVER-EST Portal and the VRC Portals are based, allows the integration of most of the e-collaboration capabilities through third party plugins, while other e-collaboration services have been integrated as JavaScript extensions:

- Forum capabilities are provided by the django-machina 7 plugin (source code available at: https://github.com/ellmetha/django-machina);
- Internal messaging and notification capabilities are provided by the django-postman8 plugin (source code available at: https://bitbucket.org/psam/django-postman/wiki/Home);
- Calendar/scheduling capabilities are provided by the django-todo9 plugin (source code available at: https://github.com/shacker/django-todo);
- Chat capabilities are provided using EXtensible Messaging and Presence Protocol (XMPP)10 standard, eJabberd 11 server side (source code available at: https://github.com/processone/ejabberd) and JavaScript XMPP Client (JSXC)12 client side (source code available at: https://github.com/jsxc/jsxc).

Information about the E-Collaboration Services can be found in chapter of the [EVER-EST DEL WP5-D5.7] document.

2.4 e-research application services

2.4.1 RODL

Research Object Digital Library (RODL) is the backend service provided by the ROHub system. It exposes a set of RESTful APIs, being the two primary ones the RO API and the RO Evolution API. The RO API defines the formats and links used to create and maintain ROs in the digital library. It implements the RO model, hence recognizing concepts such as aggregations, annotations and folders, as well as relations between different resources. ROHub supports content negotiation for metadata, including formats like RDF/XML, Turtle and TriG. The RO Evolution API defines the formats and links used to change the lifecycle stage of a RO (based on the RO evolution model), to create an immutable snapshot or archive from a mutable Live RO, as well as to retrieve their evolution provenance. Additionally, ROHub provides a SPARQL endpoint, a Notification API, a search API (Opensearch), a user management API, and an access control API. RODL source code is available at:

- https://github.com/rohub/rodl

[7] https://django-machina.readthedocs.io/en/stable/

[8] http://django-postman.readthedocs.io/en/stable/

[9] https://github.com/shacker/django-todo

[10] http://xmpp.org/

[11] https://www.ejabberd.im/

[12] https://www.jsxc.org/

2.4.2 ROHub portal

ROHub Portal provides User Interface (UI) for the RODL in the form of web page. ROHub Portal implements a set of UI components that were designed to support user dealing with his research objects. ROHub Portal source code was divided into two separate parts:

- Portal REST API, which provides data and user oriented functionality (login/logout/user session management):

  https://github.com/rohub/portal-restapi
- Portal web UI, which implements actual user interface visible components:

  https://github.com/rohub/portal-webui

2.4.3 Cloud platform

The EVER-EST VRC members will be able to access execution environments with capabilities enabling to specify, launch and terminate Virtual Machines (VMs). These capabilities are provided by Terradue Cloud Platform, a Hybrid Cloud infrastructure associating PSNC resources with a cloud controller managing a scalable data management framework that are exploited by a Developer's application to deliver a Cloud Appliance. The software of the Cloud Platform is available at the OpenNebula GitHub - https://github.com/OpenNebula

The Terradue Cloud Controller uses the OpenNebula Sunstone GUI intended for both end users and administrators that simplifies the typical management operations in private and hybrid cloud infrastructures. This GUI allows easy management of all resources and performing typical operations on them. This software is available here: https://github.com/OpenNebula/one

2.4.4 Taverna server

Taverna server exposes a RESTful API supporting the remote execution of workflows (in taverna format). Additionally, taverna server provides the capability to generate a research object encapsulating the execution provenance, which can be imported into ROHub.

Note, taverna server replaces the workflow runner service that was originally planned for this purpose (see [EVER-EST DEL WP5-D5.8] for more details).

Taverna server is a third party software and its source code is available at: https://github.com/apache/incubator-taverna-server

2.4.5 Other RO-related services

- RO checklist service: evaluates the RO quality against a predefined set of requirements. This is a third party software, with contributions from PSNC, and its source code is available at: https://github.com/wf4ever/ro-manager/tree/master/src/roweb
- RO stability service: evaluates the RO stability and reliability, i.e., RO quality throughout time; hence it is relies on the checklist service. This is a service from ESI, and its source code is available at: https://github.com/wf4ever/reliability
- RO transformation service: enables the transformation of workflow (currently taverna workflows) into a research object. This is a third party software, with contributions from PSNC, and its source code is available at: https://github.com/wf4ever/wf-ro
- RO enrichment service: annotates automatically research objects with structured metadata elicited from the textual content extracted from their resources. This is an ESI service that uses COGITO software for which Expert System has granted a license of use to the EVER-EST project. The source code of the enrichment service is available at: https://github.com/ec-everest/semantic-enrichment-service

- RO recommendation service and Collaboration Spheres: suggests research objects that might be of interest according to user's research interests. The recommender system consists of an API implemented as a rest service and a web application called the collaboration spheres that serves as user interface. This is an ESI development and the source code is available at: https://github.com/ec-everest/recommender-services

Scholarly Communication Services: harvests research object citation information from Google Scholar and Microsoft Academic, which are specialized sites that gather citation information of research works in scholarly communications. This is an ESI development and its source code is available at: https://github.com/ec-everest/scholarly-communication-services

2.5 e-learning application services

The e-Learning Services for Earth Observation (EO) data are built using two core technologies: Data Cubes and Web Notebooks. The use of these technologies responds to the expectations of an easy integration of multi-source data and a capability to provide a complete analysis of the e-Learning modules in direct contact with the final user. The web context and the provision of services from both components allow participants to interactively execute the courses.

2.5.1 Web notebook jupyter

The Jupyter Notebook is an interactive computing environment that enables users to author notebook documents that include: live code, interactive widgets, plots, narrative text, equations, images and video. The Jupyter Notebook provides a complete and self-contained record of a computation that can be converted to various formats and shared with others. The Jupyter Project source code is available in their GitHub at: https://github.com/jupyter

2.5.2 JupyterHub

JupyterHub allows the creation of a multi-user Hub that spawns, manages, and proxies multiple instances of the single-user Jupyter notebook server. Highly flexible with several customization options, JupyterHub can be used to serve notebooks to individual users or member of a given VRE.

JupyterHub is a multi-user server that manages and proxies multiple instances of the single-user Jupyter notebook server. There are three basic processes involved:

- Multi-user Hub (Python/Tornado)
- Configurable http proxy (node-http-proxy)
- Multiple single-user IPython notebook servers (Python/IPython/Tornado)

JupyterHyb uses a pluggable authentication module (PAM) as a mechanism to integrate multiple low-level authentication schemes into a high-level API. It allows the development of authentication mechanisms to be written independently of the underlying authentication scheme. The custom Authenticator subclasses were adapted to enable the EVER-EST OpenID Connect authentication schema.

The JupyterHub source code is available in their GitHub at: https://github.com/jupyterhub/

2.5.3 Data cubes

The EVER-EST Service Developer is provided with a VM containing the open source Australian Geoscience Data Cube (AGDC) software package. The Data Cube source code is available in specific fork in github at: https://github.com/data-cube/agdc-v2

2.5.4 Web notebooks

The EVER-EST e-Learning Web notebooks are available at: https://github.com/ec-everest/e-learning-modules

This repository contains an evolving list of Web Notebooks that demonstrate their potential to fully support the Earth Observation data life cycle including examples to perform data access, data cleansing, exploration, and reproducibility. It will use information dissemination and data management tools that offer easy access to search and retrieval data repositories operations to allow extraction and distribution of single parameters or combined products. These Jupyter Notebooks take advantage of EO toolboxes (e.g. GDAL, SNAP), access data in HDFS, Docker data buckets and Data Cubes running on top of a Cloud based cluster.