PROCESS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 777533.

PROviding Computing solutions for ExaScale ChallengeS

D4.4	**Updated Report on architecture evaluation and dissemination**		
Project:	PROCESS H2020 – 777533	**Start / Duration:**	01 November 2017 36 Months
Dissemination[1]:	Public	**Nature[2]:**	**R**
Due Date:	31.10.2019	**Work Package:**	**WP 4**
Filename[3]	PROCESS_D4.4_UpdatedArchitectureEvaluation_v1.0.docx		

ABSTRACT

Deliverable 4.4 is an update of the architecture and evaluation described in deliverable 4.2. D4.4 is structured around three main parts - data services (Section 2), computing services (Section 3) and service orchestration and user interfaces (Section 4). Each of these sections describes the critical changes introduced to PROCESS architecture since the release of deliverable 4.2.

In data services, no significant changes have been made. The core of data services is still a distributed virtual file system provided by the LOBCDER component, with a WebDAV interface as the main access point to the file system. LOBCDER encapsulates parts of the PROCESS file system, including separate storage infrastructures available in the project.

In computing services and service orchestration, the main change is the "flattening" of the architecture by making the cloud computing infrastructure a separate branch of the computing infrastructure. This design choice has been introduced to increase the efficiency as well as security of the whole PROCESS platform and has been enabled by the availability of a mature and complex cloud controller - Cloudify.

[1] PU = Public; CO = Confidential, only for members of the Consortium (including the EC services).

[2] R = Report; R+O = Report plus Other. Note: all "O" deliverables must be accompanied by a deliverable report.

[3] eg DX.Y_name to the deliverable_v0xx.   v1 corresponds to the final release submitted to the EC.

Deliverable Contributors:	Name	Organization	Role / Title
Deliverable Leader[4]	Bubak, Marian	AGH	Coordinator
Contributing Authors[5]	Bobák, Martin	UISAV	Writer
	Habala, Ondrej	UISAV	Writer
Reviewer(s)[6]	Belloum, Adam	UvA	Reviewer
	Meizner, Jan	AGH	Reviewer
Final review and approval	Höb, Maximilian	LMU	Reviewer

Document History

Release	Date	Reasons for Change	Status[7]	Distribution
0.0	09-09-2019	The initial structure of the document	Draft	All
0.1	17-09-2019	The final structure of the document	Draft	All
0.2	11-10-2019	First version of the document	Draft	All
0.3	15-10-2019	Finalization of computing services evaluation	Draft	All
0.4	15-10-2019	Finalization of service orchestration evaluation	Draft	All
0.5	16-10-2019	Review of the draft version	In Review	All
0.6	25-10-2019	Language correction, review of the pre-final version	In Review	All
1.0	29-10-2019	Final Version	Released	All

4 Person from the lead beneficiary that is responsible for the deliverable.

5 Person(s) from contributing partners for the deliverable.

6 Typically, person(s) with appropriate expertise to assess the deliverable quality.

7 Status = “Draft”; “In Review”; “Released”.

Table of Contents

Executive Summary

Deliverable D4.4 is the follow up of the three deliverables D4.1 (M6), D5.1 (M9) and D4.2 (M12). The objective of deliverable D4.4 is to describe the changes introduced to the PROCESS architecture since the release of D4.2. In D4.2, we described extensively the method followed to connect all the pieces of PROCESS to create the first PROCESS Pilot Architecture. The first pilot was not meant to cover all the computing and storage resources available within PROCESS, as one of the primary outcomes of the PROCESS architecture is to demonstrate that the PROCESS data and computing services can be scaled out across sites and technologies (HPC and Cloud resources) with minimal manual configuration, which is a key exascale requirement.

After the first pilot has been released, its evaluation has led to changes in the PROCESS architecture - hence the need for deliverable D4.4. The first pilot focused on building the PROCESS platform on HPC resources, we selected thus the two PROCESS HPC sites which are not planning upgrades or major maintenance tasks during the reporting period. Since then, progress has been made in utilizing the cloud infrastructure available in PROCESS. As a consequence in D4.4, we describe a “flattening” of the computing services architecture achieved by upgrading the cloud branch of our infrastructure into a component on the same level as the HPC infrastructure.

The three basic modules of the PROCESS architecture - data services, computing services and service orchestration - are evaluated independently in D4.4. For the data services, no critical changes have been introduced. The core of the data services is still provided by the LOBCDER component, implementing a virtual distributed file system over all of PROCESS’ data storages and some data-producing services (like the Dispel Gate). For the computing services, the most important change is the separation of cloud computing from HPC services and establishing it as an independent branch of computing services. The IEE portal still provides the single point of entry for users, so usage complexity has not increased. In service orchestration, the same change (elevation of cloud computing to a separate branch) has led to the direct connection of IEE and Cloudify, our main cloud controller. This leads to a simplified access to the infrastructure, increased effectiveness, and also increased the security of the PROCESS infrastructure.

In user interfaces, there has been no changes. The core is still a Docker image of Jupyter notebook with PROCESS-specific services and libraries, on which users can easily build their custom application user interfaces.

List of Figures

1 Introduction

The objective of the deliverable D4.4 is to describe the critical changes to PROCESS architecture since the release of D4.2. The deliverable D4.4 can be regarded as an extension of the deliverable D4.2. D4.4 follows the same three-step evaluation of its predecessor:

1. technology and performance analysis of standalone components
2. applicability to the PROCESS use cases
3. definition of the application programming interface

The structure of the deliverable D4.4 is affected by the evaluation approach. Every main architecture component is evaluated in a separate section:

- data services (Section 2)
- computing services (Section 3)
- service orchestration and user interfaces (Section 4)

1.1 Pilot architecture

The pilot architecture presented in the deliverable D4.2 (see Figure 1 from the deliverable D4.2) was extended several times. It is still structured as three main components: virtualization layer, data services, and computing services. Users of the PROCESS platform access it through application-oriented scientific gateways.

1.2 Pilot use cases

The PROCESS approach is evaluated by five use cases described in the deliverables D2.1 and D2.2. UC#1, UC#3, and UC#4 are focused on computing services. UC#2 and UC#5 provide special challenges for data services. UC#4 provides also challenges related to data streams. Following the requirements derived from the use cases, the PROCESS architecture is modelled as programmable infrastructure exploiting HPC and cloud resources, and creating of a virtual infrastructure that can be customised easily for every use case.

2 Data services

The data services remain unchanged. Their core (LOBCDER) is modelled as micro-infrastructure (i.e. a set of adaptor containers, data containers, interface containers, and logic containers which is created according to the requirements of the PROCESS user communities). Pre/post processing is done by DISPEL which is a part of the micro-infrastructure containers as it was described in the previous version of this deliverable. Meta-data are managed by DataNet accessed by the micro-infrastructure through DataNet-Adaptor. The data services are described within Subsections 2.1 - 2.6 from the deliverable D4.2 page 9-23.

3 Computing services

To increase utilization efficiency of the computing services, the cloud resources are managed directly by Cloudify. Usage of another tool or layer for cloud resource management will cause unnecessary overhead. Also, this approach increases flexibility and security of the whole PROCESS platform, because it requires less components, thus less potential vulnerabilities.

3.1 Architecture

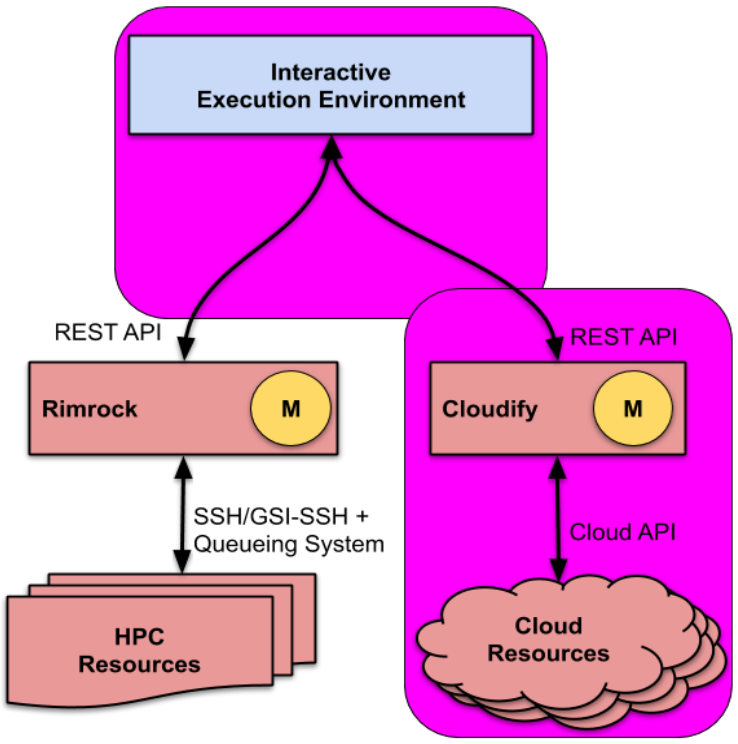

Figure 1: PROCESS computing services schema. The yellow “M” token represents a connection with a monitoring system. The PROCESS platform uses Zabbix as a monitoring agent. All the changes are highlighted in magenta.

The new version of computing services is depicted in Figure 1. During the requirement analysis, two main types of computing resources were recognized: high performance computing (HPC) resources, and cloud resources. The first version of the computing subsystem was mainly targeting the HPC part and cloud resources were managed by the Atmosphere tool which was connected to the PROCESS platform by REST-based API. However, Atmosphere was abandoned after further performance analysis, because it introduced unnecessary overhead. Also, the two resource types of the computing services are better isolated. The whole computing subsystem is integrated into the Interactive Execution Environment via REST-based API.

3.2 Implementation

The previous approach for infrastructure accessing remains the same from the user viewpoint. Users define the computation properties of their workflows through the Interactive Execution Environment which are performed on HPC resources via Rimrock or cloud resources via Cloudify. The integration of the submodules of the computing subsystem is carried out by the RESTful API described in Deliverables D4.2, and D4.3.

3.3 Evaluation and limitations

A practical result of the evaluation of the initial deployment of PROCESS technology and use case prototypes is the division of computing resources into HPC resources controlled by Rimrock and cloud resources controlled by Cloudify. Both Rimrock and Cloudify can be enacted from the IEE, which thus still provides a single unified point of entry for users wishing to perform computations in the PROCESS infrastructure.

3.3.1 Heterogeneity of the infrastructures

The fundamental heterogeneity of PROCESS infrastructure components - that of division into HPC and cloud resources, with their specific application and modes of use, has been agreed upon within the PROCESS consortium and presented above. While there are further sources of heterogeneity - hardware capabilities, methods of access and control, methods of task queuing and task enactment, these do not present too many difficulties, as both the HPC controller - Rimrock - and the cloud controller – Cloudify – provide REST API to access their services. Thus the single point of entry, IEE, is able to efficiently enact jobs on both branches of the infrastructure without having to account for specific properties of its components.

3.3.2 Software dependencies

Software dependency in PROCESS has been addressed by wrapping user code in self-contained executable modules - containers. These containers can be deployed in the infrastructure on-demand and they contain all the software necessary to execute the user's task. The principal software dependency here is twofold - for the HPC infrastructure the containers are hosted by Singularity and CharlieCloud, for the cloud infrastructure Docker is used. These dependencies are dictated by the goal of PROCESS to use – and build on – pre-existing infrastructure.

User applications have more software dependencies of their own. From the point of view of the PROCESS computing services applications software dependencies are invisible because of the containers abstraction and thus will not be reviewed here.

3.4 Applicability to use cases

Since the changes of the computing subsystem are only technical, the previous analysis of the applicability to the use cases (subsection 3.4 from the deliverable D4.2) is valid for the current version of the computing subsystem also.

3.5 API

The API of the computing services is REST-based API described in the deliverable D4.2 page 27.

4 Service Orchestration and User Interfaces

The approach for user interfaces remains the same. They are modelled as containerized microservices that are based on modularity, reusability, easy-to-use, openness, and interoperability. The users are able to interact with the PROCESS platform through the command-line interface, graphical user interface, interactive execution environment, and script API (see Section 4 page 27-31 in deliverable 4.2).

To lightweight the service orchestration, there are three main orchestration modules: distributed virtual file system, HPC resource manager, and cloud resource manager. Originally during the Project preparation we have envisioned Atmosphere as a multi-cloud solution to schedule workloads in the cloud environment of the PROCESS platform. During the scope of the project especially the evaluation of the architecture we have found it to be redundant component as the Cloudify would provide same basic functionality plus additional one needed for the task. The basic functions of Atmosphere which were replaced are:

- providing access to multiple public/private clouds
- launching and monitoring of cloud instances
- managing of cloud images
- providing of a web user interface

Cloudify is arguably a more robust solution, so in the interest of avoiding duplication of functionality, it supplies the functions of Atmosphere. Thus, the services orchestration was re-modelled (see the following subsections).

4.1 Architecture

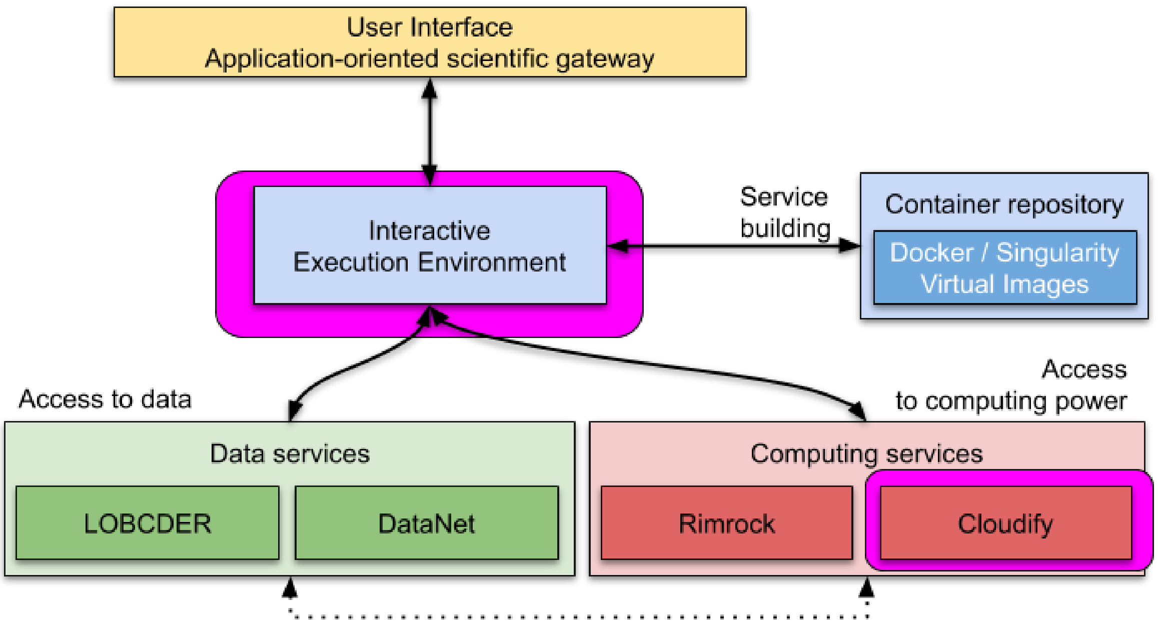
User Interface
Application-oriented scientific gateway
Interactive Execution Environment
Service building
Container repository
Docker / Singularity Virtual Images
Access to data
Access to computing power
Data services
LOBCDER
DataNet
Computing services
Rimrock
Cloudify

Figure 2: PROCESS service orchestration schema. All the changes are highlighted in magenta.

Figure 2 depicts the new version of the service orchestration. The new design models it as follows: the data services (LOBCDER representing distributed virtual file system, and DataNet acting as meta-data manager) are integrated directly into the Interactive Execution Environment via RESTful API and WebDAV. Access to the computing resources is provided by Rimrock (HPC manager), and Cloudify (cloud manager). Both tools are also integrated into the Interactive Execution Environment by RESTful API.

4.2 Implementation

The implementation of the individual modules is unchanged (see the deliverable D4.2). The new is the way of the connection and integration of the individual modules. All of the individual modules support RESTful API which provides the uniform way for accessing both the data and to computing resources.

4.3 Evaluation and limitations

The basic analysis of evaluation and limitations of PROCESS' approach to service orchestration and user interfaces remains unchanged comparing to the one presented in deliverable 4.2. The decision to stop using Atmosphere has not created any added limitation or complexity to our work. Challenges related to software dependencies observed during deployment and testing are described below.

4.3.1 Software dependencies

Since the new “flattened” architecture of service orchestration further highlights cloud computing as an independent branch of the infrastructure, we must note that the basic method of deployment of software in the cloud, using Cloudify as the orchestration manager, is via TOSCA templates and Ansible roles. These are pre-configured to hide the complexities of different cloud environments and the IEE can enact them via simple Cloudify's REST API, which means that users do not have to deal with configuration of the Cloud environment and can focus on their applications.

User interfaces for applications are designed as modular and extensible, created as containers (Docker images) derived from a basic Jupyter image[8]. On top of this image we will deploy a set of PROCESS-specific services and libraries, which can be used by the users while developing their application code. This decision introduce two important constraints on PROCESS users - the need to organize their work in the manner of a Jupyter notebook, and to develop at least the highest-level codes in Python. It is possible to forego the use of Jupyter and Python, at the expense of having to integrate PROCESS-specific services and libraries manually.

4.4 Applicability to use cases

Since the changes of the service orchestration are only technical, the previous analysis of the applicability to the use cases (Subsection 4.4 in deliverable D4.2, page 30-31) is still valid for the current version of the service orchestration also.

4.5 API

The core component of the new service orchestration model is the Interactive Execution Environment (see deliverable D4.2 page 31). The API of the data and computing services is REST-based API described in the deliverable D4.2.

[8]https://hub.docker.com/u/jupyter/

5 Dissemination

The main dissemination activity of the PROCESS architecture was the half-day workshop *Platform-driven e-infrastructure innovations* (EINFRA) at the IEEE eScience 2019 International Conference in San Diego, California, USA, where exascale projects and use cases were invited to join. This workshop was a follow up of the eScience 2018 workshop in Amsterdam, The Netherlands.

The PROCESS work was presented in the following papers:

1. Martin Bobák, Ladislav Hluchy, Adam Belloum, Reginald Cushing, Jan Meizner, Piotr Nowakowski, Viet Tran, Ondrej Habala, Jason Maassen, Balázs Somosköi, Mara Graziani, Matti Heikkurinen, Maximilian Höb, Jan Schmidt. Reference Exascale Architecture. In Proceeding IEEE 15th International Conference on eScience (eScience) 2019 : Workshop on Platform-Driven e-Infrastructure Innovations (EINFRA), San Diego, California, USA, IEEE 2019 p. 479-487. IEEE Catalog Number: CFP1978F-ART. ISBN: 978-1-7281-2451-3. DOI: 10.1109/eScience.2019.00063.
2. Hanno Spreeuw, Souley Madougou, Ronald Van Haren, Berend Weel, Adam Belloum, Jason Maassen. Unlocking the LOFAR LTA. In Proceeding IEEE 15th International Conference on eScience (eScience) 2019 : Workshop on Platform-Driven e-Infrastructure Innovations (EINFRA), San Diego, California, USA, IEEE 2019 p. 467-470. IEEE Catalog Number: CFP1978F-ART. ISBN: 978-1-7281-2451-3. DOI: 10.1109/eScience.2019.00061.
3. Reginald Cushing, Onno Valkering, Adam Belloum, Cees de Laat. Towards a New Paradigm for Programming Scientific Workflows. In Proceeding IEEE 15th International Conference on eScience (eScience) 2019 : Workshop on Platform-Driven e-Infrastructure Innovations (EINFRA), San Diego, California, USA, IEEE 2019 p. 604-608. IEEE Catalog Number: CFP1978F-ART. ISBN: 978-1-7281-2451-3. DOI: 10.1109/eScience.2019.00083
4. Valkering O, Belloum A. Privacy-Preserving Record Linkage with Spark. In2019 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID) 2019 May 1 (pp. 440-448).
5. Höb, M., PROCESS -- PROviding Computing solutions for ExaScale challengeS (Poster), In ICCS 2019 Proceedings and Book of Abstracts, Faro, Portugal, June, 2019.
6. Martin Bobák, Ladislav Hluchý, Mara Graziani, Henning Müller. Machine Learning in Medical Imaging. In Proceeding 1st International Conference on New Research and Development in Technical and Natural Science (ICNRDTNS). Radenci, Slovenia, September 18 – 20, 2019, p. 37 – 40. ISBN 978-961-290-461-6.