MMP

Deliverable D1.6

Deliverable:	D1.6 Framework release ALPHA Version (TRL5)
Responsible Partner:	CTU
Contributors:	All
Dissemination Level:	PU
Distribution List:	Consortium members, European Commission, PTA
Due Date:	31.10.2016
Preparation Date:	25.10.2016
Report Status:	FINAL

NMP.2013.1.4-1:	Multiscale Modelling Platform: Smart design of nano-enabled products in green technologies
Project nr:	604279, collaborative project
Project Duration:	01.01.2014 – 31.12.2016
Project website:	http://www. mmp-project.eu
Coordinator:	Ir. Sjoukje Wiegersma, sjoukje.wiegersma@tno.nl
Project partners:	TNO, TU/e, CTU, Access, VTT, CelSian, Philips, ~~Abengoa~~

<u>Deliverable description from the project proposal</u>

Framework release ALPHA version (TRL5): The source codes of the platform will be available for public download and experimenting from project site under LGPL license, including documentation (user and reference manuals).

Table of content

1 Introduction

The approach followed in the MMP project is based on a system of distributed, interacting objects designed to solve given problem, implemented in Python [1]. The individual objects represent entities in the problem domain, including individual simulation packages, but also the data, such as fields and properties. The abstract classes are introduced for all entities in the model space [2]. They define a common interface that needs to be implemented by any derived class, representing particular implementation of specific component. Such interface concept allows using any derived class on a very abstract level, using common services defined by abstract class, without being concerned with the implementation details of an individual software component. This essentially allows manipulating all simulation tools using the same interface. Moreover, as the simulation data are represented by objects as well, the platform is independent on particular data format(s), as the exchanged data (such as fields and properties) can be manipulated using the same abstract interface.

The complex simulation pipelines consist of top-level script in Python language enriched by newly introduced classes. The top-level script can be also generated using a graphical tool.
To facilitate execution and development of the simulation workflows, the platform provides the transparent communication mechanism that will take care of the network communication between the objects. An important feature is the transparency, which hides the details of remote communication to the user and allows working with local and remote objects in the same way. The communication layer is built on Pyro4 library [3], which provides a transparent distributed object system fully integrated into Python.

The platform is designed to work on virtually any distributed platform, including grid and cloud infrastructure [4]. For the purpose of performing simulations within the MMP project, the individual simulations and therefore the individual simulation packages have been distributed over the network, running on dedicated servers provided by individual partners, forming grid-like infrastructure.

This deliverable documents the public release of the MMP platform source codes together with its documentation, internal tests and dedicated website.

2 Platform license

The MuPIF platform is available under GNU Lesser General Public License version 3.0 (LGPL v3). The license is available from Free Software Foundation website (https://www.gnu.org/licenses/lgpl-3.0.en.html).

3 Platform source code

The platform development is hosted on SourceForge (http://sourceforge.net/projects/mupif/), which provides collaboration tools and software repository for open source projects, since the very beginning of the project. The GIT software repository of the MMP platform (http://sourceforge.net/p/mupif/code/ci/master/tree/) is used to track the day-by-day development of the platform and allows to retrieve any version. The software repository of the platform is publically accessible.

In certain stages of development, a new version is released. The releases are made available as Python PIP packages. The pip is a package management system allowing simple installation and management of software packages. The advantage of the package is that in addition to sources itself, it is carrying all the necessary dependencies that are satisfied automatically during installation. The platform packages have been uploaded and registered on Python Package Index (https://pypi.python.org/pypi). The package version related to the ALPHA framework release, described by this document, is **1.0.0**.

The platform installation is a collection of several directories and files. They are organized in the hierarchical structure, which is maintained in the repository and is transferred to a computer running the platform. Such a structure allow various installation setups, e.g. user installation with non-administrative rights, installation from a tar file, root installation for all user etc. The platform offers high degree of versatility, allowing easy installation on Windows/Linux/Unix/Mac operating systems.

```
MuPIF_TOP_DIR - contains source code and other files of the MuPIF package
  +--mupif - contains source code of the MuPIF package
  |       +--doc - documentation (reference manual and user guide)
  |       +--examples - examples and tests
  |       +--tests - unit tests
  |       +--Physics - module for units
  |       +--tools - various supportive tools
  |       +--APIs - implemented APIs
  |       +--*.py - MuPIF classes
  |       +--__init__.py - description of MuPIF module
  +--README.txt - general description
  +--setup.py - support for setuptools
  +--MANIFEST.in - support for setuptools
```

Fig. 1: Installation structure of MuPIF

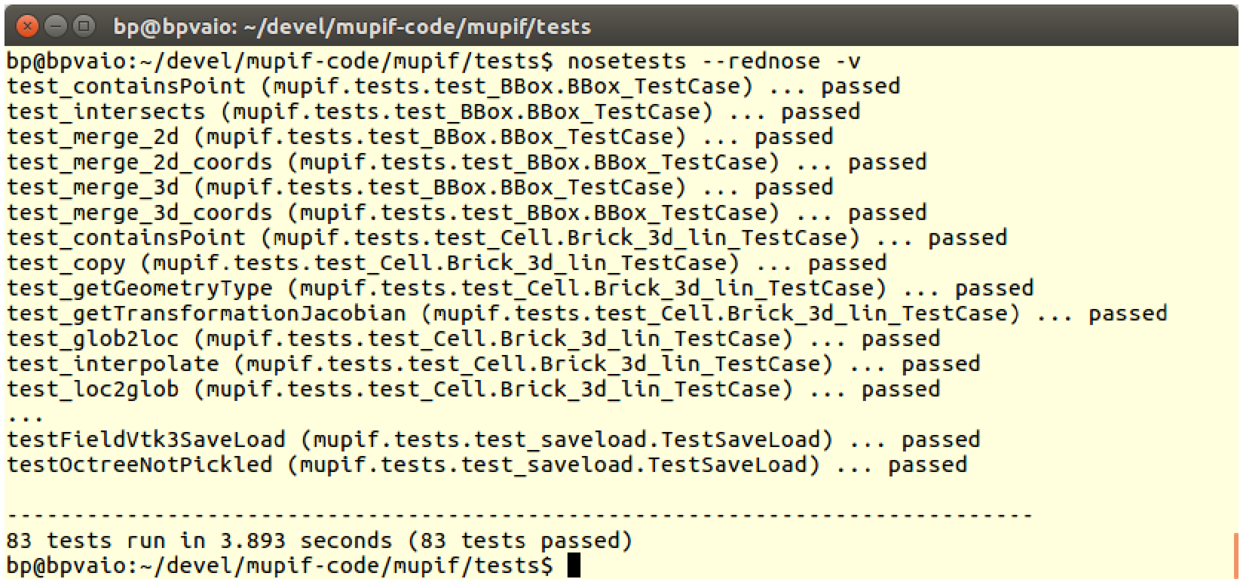
4 Readiness level and software quality measures

The Technology Readiness Level (TRL) for this final release complies with TRL5, which is defined as “Specific software components are integrated with reasonably realistic supporting elements so that it can be tested in a simulated environment”.

The release version has been tested in industrial and realistic environments, as documented in milestone MS3 [5], and Deliverables D2.6, D2.7, where prototype simulation chains have been executed in a distributed configurations set up by the project partners utilizing the available hardware and software resources at their disposition. The platform runs have been also successfully demonstrated live on 3rd MMP webinar (D4.12) and during plugfest in Cecam workshop [7].

At present, the platform is being also actively used outside the MMP project which provides another independent evaluation. For example, it is used in the frame of the research project funded by the Czech Grant Agency [8] to simulate fire propagation and its influence on mechanical behavior of steel frames, or to analyze the effect of humidity and temperature on water condensation inside tunnels [9].

To ensure continuous quality of the platform development, a series of unit tests has been developed. For each framework class and its individual methods the corresponding unit test has been developed. Each test tests the individual method by providing different inputs and comparing the outputs with expected values. Altogether, more than 80 independent tests have been developed. These tests are run regularly, to ensure the software quality and stability.



```
bp@bpvaio:~/devel/mupif-code/mupif/tests$ nosetests --rednose -v
test_containsPoint (mupif.tests.test_BBox.BBox_TestCase) ... passed
test_intersects (mupif.tests.test_BBox.BBox_TestCase) ... passed
test_merge_2d (mupif.tests.test_BBox.BBox_TestCase) ... passed
test_merge_2d_coords (mupif.tests.test_BBox.BBox_TestCase) ... passed
test_merge_3d (mupif.tests.test_BBox.BBox_TestCase) ... passed
test_merge_3d_coords (mupif.tests.test_BBox.BBox_TestCase) ... passed
test_containsPoint (mupif.tests.test_Cell.Brick_3d_lin_TestCase) ... passed
test_copy (mupif.tests.test_Cell.Brick_3d_lin_TestCase) ... passed
test_getGeometryType (mupif.tests.test_Cell.Brick_3d_lin_TestCase) ... passed
test_getTransformationJacobian (mupif.tests.test_Cell.Brick_3d_lin_TestCase) ... passed
test_glob2loc (mupif.tests.test_Cell.Brick_3d_lin_TestCase) ... passed
test_interpolate (mupif.tests.test_Cell.Brick_3d_lin_TestCase) ... passed
test_loc2glob (mupif.tests.test_Cell.Brick_3d_lin_TestCase) ... passed
...
testFieldVtk3SaveLoad (mupif.tests.test_saveload.TestSaveLoad) ... passed
testOctreeNotPickled (mupif.tests.test_saveload.TestSaveLoad) ... passed

-----------------------------------------------------------------------------
83 tests run in 3.893 seconds (83 tests passed)
bp@bpvaio:~/devel/mupif-code/mupif/tests$
```

Fig. 2: Screenshot illustrating the unit test run and its results

In addition, a series of 16 demonstration examples have been developed, illustrating the platform usage on real (although simplified) simulation workflows. The Travis CI service (travis-ci.org) has been set-up to establish an automated testing environment. This testing environment allows to automatically run the unit tests and demonstration examples whenever the source repository update is detected. The results are easily accessible (https://travis-ci.org/mmp-project/mupif). When any change breaks any test, the developers are immediately notified.

5 Platform documentation

The documentation of the platform consist of User and Reference manuals. The User manual describes the platform design and usage from the user perspective. It covers the platform installation, set-up and usage. It also covers advanced topics such as the API implementation, distributed computing, security, and monitoring. The User manual is distributed as a part of any release version within corresponding python package and is also available on platform website for a separate download at www.mupif.org.

MuPIF User Manual Table of Content

Snapshot of User MuPIF User Manual Content.

The reference manual serves as reference for platform developers. It contains the documentation of all framework classes and their methods. The core of reference manual is automatically generated form the commented platform sources, essentially allowing to maintain updated documentation all the time.

MuPIF Reference manual

Introduction

    - mupif.IntegrationRule module
    - mupif.JobManager module
    - mupif.Localizer module
    - mupif.Mesh module
    - mupif.Octree module
    - mupif.Property module
    - mupif.PyroFile module
    - mupif.PyroUtil module
    - mupif.RemoteAppRecord module
    - mupif.TimeStep module
    - mupif.Timer module
    - mupif.Util module
    - mupif.ValueType module
    - mupif.Vertex module
    - mupif.VtkReader2 module
    - mupif.fieldID module
    - mupif.functionID module
    - mupif.propertyID module
    - Module contents
- Acknowledgement

Table of content from MuPIF's reference manual.

6 Platform website

As already noted, the platform development is hosted on Source Forge. This site serves not only the software repository, but also provides a complete environment supporting open source project development. This includes dedicated wiki, issue tracking system, and discussion forum. In particular, the issue tracking system allows the platform users to report any problem encountered and track their status. The discussion forum allows to post any question or ask for support the developers and/or other users.

In addition to Source Forge, which is oriented rather to platform developers, the general platform web presentation has been developed, oriented to the general public and potential users. It contains general presentation of the platform, its design principles and features. The platform use is demonstrated on selected examples. The links to platform webinars, documentation and other resources are given. This dedicated website is available at www.mupif.org.

7 References

1. Python Software Foundation. Python Language Reference, version 2.7. Available at http://www.python.org
2. D1.1 Application Interface Specification, MMP project, 2014.
3. Pyro-Python Remote Objects, http://pythonhosted.org/Pyro.
4. D1.4 Hardware and services like simulation pipeline in place,
5. MMP project, 2015.
6. MS3: Running a first simulation chain on a distributed configuration (TRL4), MMP project, 2015.
7. CECAM workshop Multiscale Simulation: from Materials through to Industrial Usage, Sept 5-7, 2016, Dublin, Ireland. http://www.cecam.ie/multiscale-simulation-from-materials-through-to-industrial-usage-5-7-sept-2016/
8. Research project from Czech Science Foundation: Coupled model of structural standard fire test, 2016-2018, No. 16-18448S.
9. Research project CESTI: Centre for effective and sustainable transport infrastructure issued by Technological Agency of the Czech Republic, TE01020168, 2013-2019.