Project Acronym: MuG

Project title: Multi-Scale Complex Genomics (MuG)

Call: H2020-EINFRA-2015-1

Topic: EINFRA-9-2015

Project Number: 676556

Project Coordinator: Institute for Research in Biomedicine (IRB Barcelona)

Project start date: 1/11/2015

Duration: 36 months

Deliverable 5.1: Computational infrastructure set-up

Lead beneficiary: Barcelona Supercomputing Center (BSC-CNS)

Dissemination level: PUBLIC

Due date: 01/11/2016

Actual submission date: 07/11/2016

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 676556.

Document history

Version	Contributor(s)	Partner	Date	Comments
0.1	Josep Ll. Gelpi	BSC	12/10/2016	First draft
0.2	Javier Conejero	BSC	18/10/2016	Added COMPSs and PMES sections
0.3	Javier Conejero	BSC	20/10/2016	Added COMPS and PMES roadmap actions
0.6	Adam Hospital	IRB	26/10/2016	Added SGE related sections
0.7	Josep Ll. Gelpi	BSC	30/10/2016	Complete version
1.0			07/11/2016	Approved by technical and supervisory board

Table of contents

1 EXECUTIVE SUMMARY

MuG Virtual Research Environment should provide the members of the 3D/4D genome community with an adequate combination of relevant information, data, and computational tools. The combination should help, with a friendly access, the researcher to analyse data, either from repositories, or obtained from experiment or simulation; combine and compare such analysis results with related studies and reference data.

This document describes the initial prototype of MuG basic computational infrastructure. The initial installation provides a cloud-based environment able to perform a series of operations packed as virtual machines, and several interfaces of access, including specific web portals, and programmatic access. Initial data repository is built on top of IRB's Molecular Modelling and Bioinformatics Data repository. An expected roadmap of future developments is included in the last section of the document.

2 INTRODUCTION

2.1 Motivation and strategy

The concept of Virtual Research Environments could be interpreted in several ways, from a simple web portal where users are just able to browse in a series of existing data items, to a fully developed computational infrastructure where users can add data and tools, provided either by the infrastructure or users themselves. In the case of homogenous communities, the degree of expertise of potential users usually defines the most adequate approach. However 3D/4D genomics community is not homogeneous. Potential members of MuG VRE include researchers from biophysics to cell biology, each of them with expertise in one of the data levels that the project is scoping (see D3.1 for a more formal discussion). In their day-to-day activity, researchers work in a specific level probably being experts in such a level, but mostly naive in the others. MuG main contribution to the 3D/4D community is its **multi-scale scope**. MuG VRE will put together data coming from atomistic simulations, genome annotation, middle and high scale 3D genomics, and cell biology imaging data, and will establish the necessary relationships among the different levels to build an integrated view of the biological phenomena under study. From the computational point of view, this constitutes an extraordinary challenge, as users are expected to be active in generating and analysing data in a given level, and interested in browsing and integrating data from the other levels. This breaks the common paradigm in bioinformatics where most users are data consumers, while data is produced by a reduced and controlled community or even a single group. As a general rule, all members of the MuG VRE community should be provided with access to data and tools to perform analyses, but also with the necessary tools to upload their own data and analyze and integrate it in the community knowledgebase. The computational infrastructure should assure interoperability of analysis tools and generate an integrated environment with a seamless transition among the available data levels. The design of MuG computational infrastructure has been split in several components, a 3D/4D browser (WP3), a data infrastructure (WP4), and a collection of interoperable analysis tools (WP6). The basic computational infrastructure described here has the mission of managing the above components, and integrating them in a single user environment, assuring the best efficiency in data mobilization and process. The chosen strategy, outlined below, will allow VRE members to i) browse the available data in an integrated way, ii) incorporate raw data to the VRE that will perform the appropriate analysis, and incorporate results to MuG's data repository, iii) use the VRE as an analysis infrastructure using the available tools on existing or uploaded data, and iv) download data in the appropriate formats for in-house further analysis. Finally, MuG infrastructure should be aligned with present trends in Europe, leading to reinforce the use of Europe wide e-infrastructures, namely, EGI[1], EUDAT[2], or PRACE[3]. Although the initial infrastructure is local, it will be already linked at the level of data with data repositories hosted at EMBL-EBI, and gradually moved to use EUDAT facilities. On the process side, a limited version of the presented prototype is already available in EGI AppDB.

2.2 Background

MuG VRE computational infrastructure is using already existing components, which will be adapted to work in an integrated way, and will be complemented with the necessary MuG specific tools. The infrastructure layout has been designed as an evolution of the cloud-infrastructure built for the project transPLANT[4], located at http://transplantdb.bsc.es. This layout has been merged with the existing MMB-IRB's infrastructure to manage web-based applications. MuG's data repository will be largely based on MMB-IRB's noSQL databases that include already the Protein Data Bank[5], Uniprot[6], and most remarkably BiGNASim[7], a database for nucleic acids trajectories simulation that will hold atomistic level data in MuG.

3 COMPUTATIONAL INFRASTRUCTURE INITIAL DESIGN

MuG computational infrastructure has been designed to fulfil the following principles:

1. Flexible environment, able to adapt to the specific needs of the analysis tools (from WP6), both in terms of software requirements and computational resources.
2. Software scheduler(s), able to manage analysis workflows, and computational resources in a transparent and adaptable manner. This will be an elastic infrastructure with automatic adaptation to user loads.
3. Multi-scale execution. Analysis workflows could be executed either at the cluster level, in HPC environments, or distributed infrastructures like EGI.
4. Web-based access centered in the MuG multi-scale browser (designed in WP3). This will be complemented by programmatic access using well-established interfaces including Galaxy. User access will integrate the Authentication and Authorization Infrastructure being designed within the Elixir initiative[8].
5. The infrastructure will be eventually interfaced to European e-infrastructures, including the EGI for computation, and EUDAT for shared storage.

Figure 1 shows a general schema of the projected infrastructure.

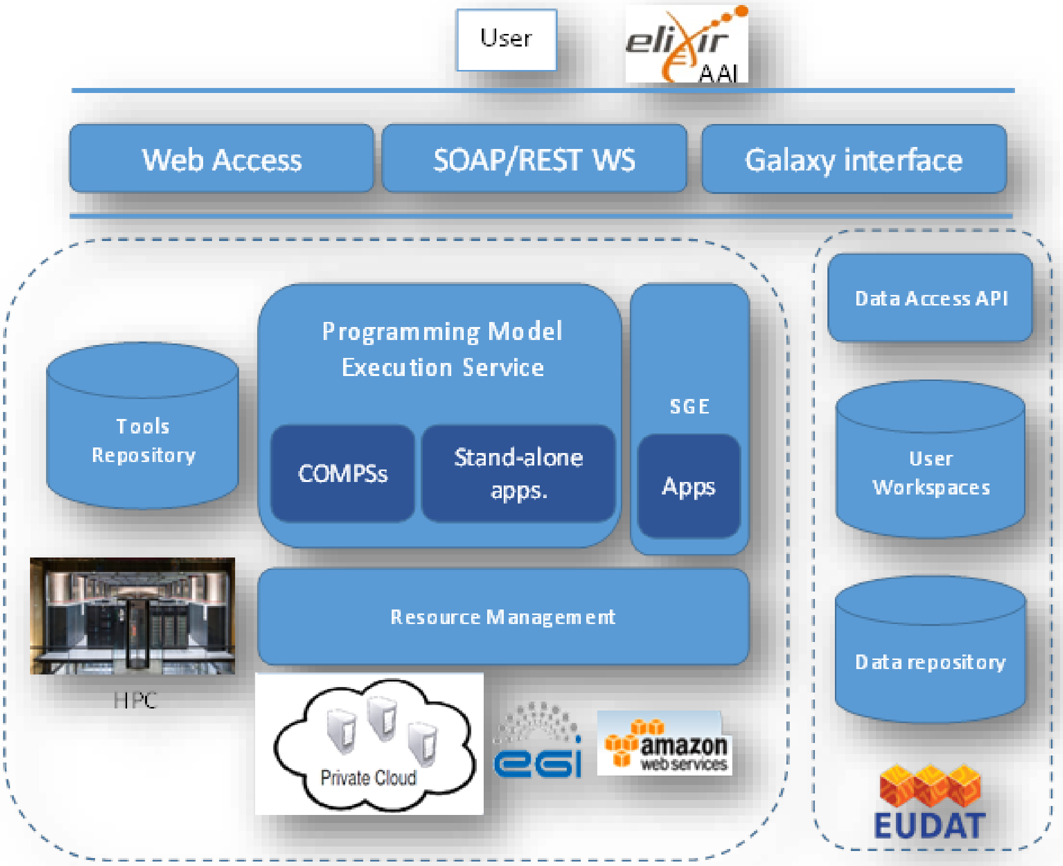


Figure 1. Projected layout of MuG's computational infrastructure

3.1 Projected Data repository

MuG's data repository will store data related to the project, user workspaces, and metadata pointing to relevant data stored in public repositories, either produced by MuG or by others (see D4.1 and D4.2 for details). Principles used in the design of MuG's data repository are:

1. Following MuG's Data management plan (D4.2), data considered in MuG will be stored in the most appropriate data repository, including sites like BioSamples[9] or ArrayExpress[10] for experimental data, BigNASim[7] for simulation data, and others. Data types that lack any appropriate and stable public repository will be stored locally by MuG data repository.
2. MuG data repository will maintain an updated copy of the necessary metadata to locate such types of data (local and remote). Metadata will be integrated to allow for global searches.
3. Data mobilization will include both local and remote data in an integrated and transparent way.
4. Due to the heterogeneity of stored data, the metadata/data repository will need a flexible structure.
5. A REST API will be provided to access the data.
6. Data repository will integrate a personal user workspace. User workspace will contain user's data and results, and information about users' activity.

7. MuG will explore the synchronisation of data repository with European facilities like EUDAT to provide sustainability to the VRE.

To fulfil those principles, MuG with extend initially MMB-IRB's data repository based in two NoSQL database managers (MongoDB[11] and Cassandra[12]) that already manage reference data (PDB, Uniprot) and the MD nucleic acids database BIGNASim. Figure 2 shows a general schema. Present repository provides data access through a REST API that will be extended to cover MuG specific data. As part of the future roadmap (Task 5.3) the use of different types of active storage will be explored to improve efficiency in data retrieval and process.

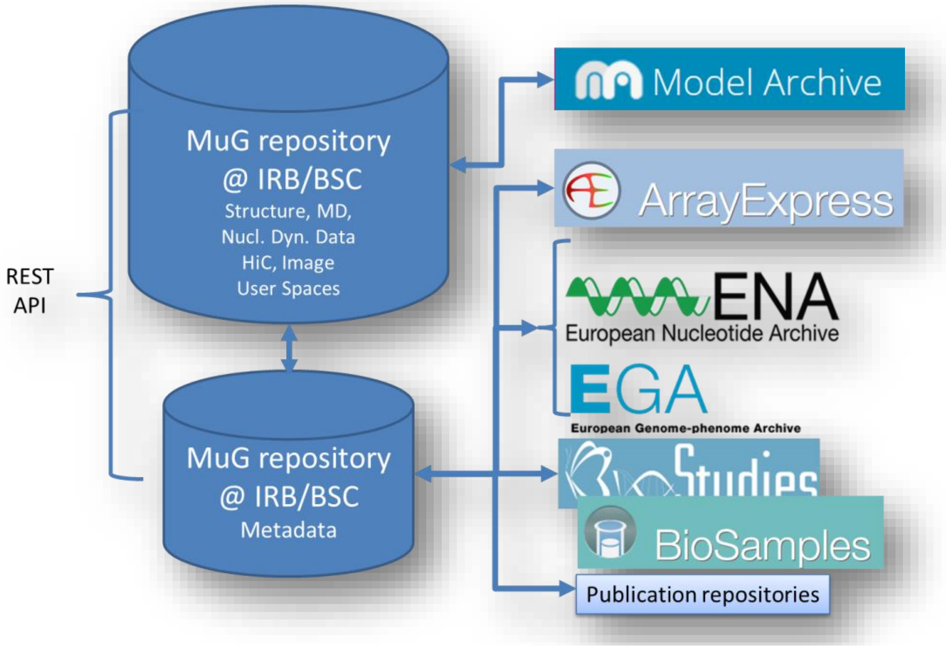


Figure 2. Projected Data Repository Components

4 SOFTWARE COMPONENTS

The following section describes individually the software components used in the initial installation and their specific function.

4.1 OpenNebula. Cloud infrastructure

The prototype's cloud software stack consists of a local infrastructure managed by the Cloud middleware OpenNebula[13]. OpenNebula enables an easy management of a private Cloud infrastructure. It is composed by a front-end that offers different interfaces, such as REST, XML-RPC and Web based, and that is able to manage virtual machines over multiple remote hosts running different hypervisors (e.g. Xen[14]; KVM[15]; or VMWare[16]). The MuG prototype is based in the use of KVM hypervisor. Moreover, using OpenNebula as infrastructure provider also enables the use of Amazon EC2[17] resources if the local infrastructure is not enough for running a specific application.

The OpenNebula cloud middleware is in charge of managing directly the virtualization environment. Management is dynamic and can be adjusted to the specific requirements of the applications by the use of the Open Cloud Computing Interface (OCCI[18]). Additionally, the OpenNebula infrastructure manages a series of virtual machines performing housekeeping functions like database provision, or web-based applications that are instantiated at boot time and maintained permanently in the system. Although a fully operational OpenNebula is used, the infrastructure is built in a way its infrastructure is fully controlled by process management middleware(s) (PMES, and SGE in the initial prototype), hence the user does not need to interact with the cloud manager.

4.2 Process management

Management of processes in the initial MuG infrastructure uses two different environments. Sun Grid Engine will be in charge to manage web application backends (i.e. processes involving well defined and stable workflows), where the major requirement is to be able to manage variable levels of user demand. Workflows that show a more complex structure where, for instance, computational resources should be adjusted at run time, will be managed by COMPSs/PMES. COMPSs programming model will be used when appropriate to develop new workflows in MuG environment.

4.2.1 Sun Grid Engine queuing system

Sun Grid Engine (SGE), developed originally by Sun Microsystems (and now maintained by Oracle), was designed to manage distributed software executions in heterogeneous computational environments. SGE is normally used in cluster-based infrastructures as a general process scheduler. Capabilities of SGE include, among others, resource management, remote execution, parallel execution management, interactive processes, monitoring and accounting, integration with Amazon EC2 or Hadoop. MMB-IRB's infrastructure for web applications has been using SGE as scheduler for their backends. To adapt to MuG general infrastructure (Figure 1), a specific connection with OpenNebula cloud manager has been set up through the use of oneFlow, a component of the OpenNebula framework that allows managing Multi-VM application and auto-scaling. Figure 3 shows a schema of the structure implemented in the MuG prototype.

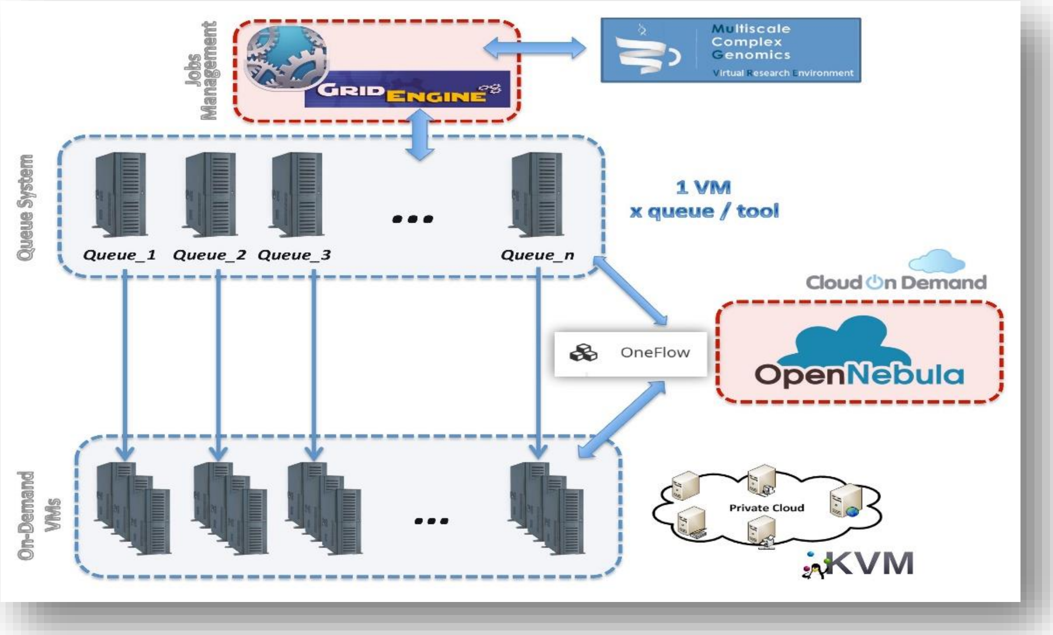


Figure 3. Layout of the integration of Sun Grid engine in MuG computational infrastructure

BackEnds of applications run under the control of SGE are placed in Virtual Machines that are managed by OpenNebula. In the present installation, a single VM is allocated for every application, matching the SGE queue configurations. Such VM is managing the response of the web application, and eventually a certain amount of load. In the case of increased demand, oneFlow instructs openNebula to allocate additional instances of the VMs which simply increase the amount of hosts available for SGE queues, thus allowing to manage the increased demand. OneFlow is monitoring VMs load in a way that instances can be eventually stopped when the demand is decreased.

4.2.2 COMPSs programming model

COMP Superscalar (COMPSs)[19] is a programming model designed to ease the development of applications for distributed infrastructures (e.g. Clusters, Grids and Clouds). To this end, COMPSs features a runtime system able to exploit the inherent parallelism of applications at execution time, hiding its complexities to programmers.

The COMPSs model has four key characteristics: sequential programming, infrastructure unaware, no APIs, and standard programming languages support and application lifecycle management.

The first key characteristic is provided by COMPSs through hiding the complexities of parallelization to developers. Consequently, programmers do not need to deal with the duties of parallelization, such as thread creation and synchronization, data distribution, messaging or fault tolerance. Instead, the model is based on **sequential programming**, which makes it appealing to users that either lack parallel programming expertise or are looking for better programmability. Although programming is sequential, execution is parallel, since at execution time COMPSs builds a workflow composed by the tasks of the application, which are connected through edges that denote data dependencies between them, determined by annotations (that can include specifications of the needs of each task). From this workflow, the COMPSs runtime is able to execute different application tasks at a time within a master-workers architecture, exploiting the inherent parallelism. This architecture is characterized by a master node and multiple worker nodes (Figure 4). In this scenario, the user submits its application to the master node, which orchestrates the parallelization and launches the tasks in the available resources, distributing the input data and collecting the results. Therefore, COMPSs provides an easy way of developing scientific workflow as the ones tackled in MuG.

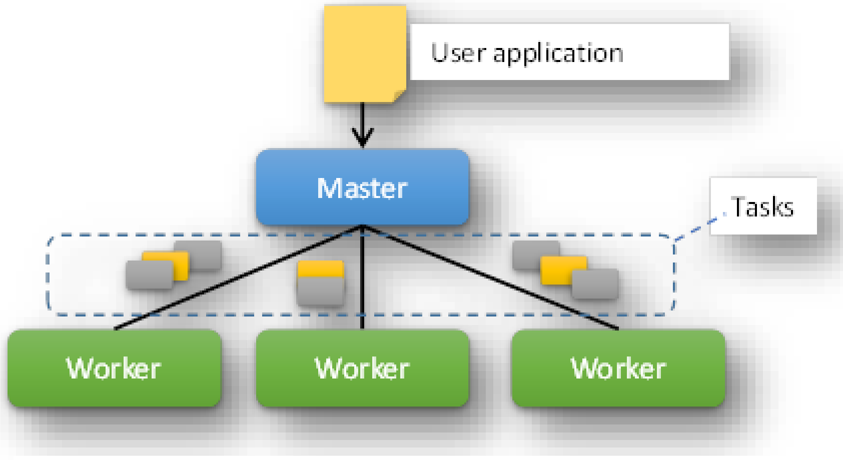


Figure 4. COMPSs deployment infrastructure

The next key characteristic is focused on the multiple infrastructures aimed at providing a distributed environment. COMPSs offers a model that abstracts the application from the underlying distributed infrastructure, making it **infrastructure agnostic**. This characteristic also enables the portability of the applications between different infrastructures. COMPSs runtime takes care of the necessary adaptation to the underlying infrastructure in a completely transparent way. The present implementation of COMPSs, however, allows including specific requirements to guide OpenNebula in the selection of the necessary virtual machines.

Regarding the **no APIs and standard programming languages** characteristic, COMPSs is based on popular programming languages, such as Java, C/C++ and Python. This facilitates the adoption of the model, since programmers can reuse most of their previous code, and existing scripts can be easily adapted. To this end, COMPSs is able to provide direct support for Java applications, while for other languages, like C/C++ and Python, it provides bindings that interface with the runtime.

Finally, COMPSs is designed to manage the **application lifecycle**. This involves the COMPSs' runtime to request for resources (e.g. virtual machines) to the underlying infrastructure and distributes the computational work of the application amongst them ensuring that data dependencies are kept when a request arrives to the system.

4.2.3 Programming Model Enacting Service

The Programming Model Enacting Service (PMES) allows users to submit job execution requests to remote servers (Figure 5). To this end, PMES requires information about the job that the users want to submit, such as an application's name and its input parameters, and starts the COMPSs' runtime within a virtual machine in a Cloud environment in order to execute it or directly execute the application. Since PMES is also deployed within a virtual machine, as well as the COMPSs runtime deployed for the application execution, both of them interact with the Cloud provider (e.g. OpenNebula) through the use of an OCCI connector. PMES also provides a graphical interface designed to enable the interaction with the service directly from a web browser.

The PMES is able to manage the execution of two different kinds of applications within virtualized resources: stand-alone and COMPSs applications. This functionality enables to use the pre-existing applications, which can be run on the cloud platform without modification.

The use of COMPSs as workflow manager is considered within PMES in order to improve the performance on workflow executions. These applications are executed by launching the COMPSs runtime in a virtual machine; whereas to execute standalone applications, the PMES is also able to run any specific command with certain inputs, either directly for low demanding applications or launching the necessary VM's.

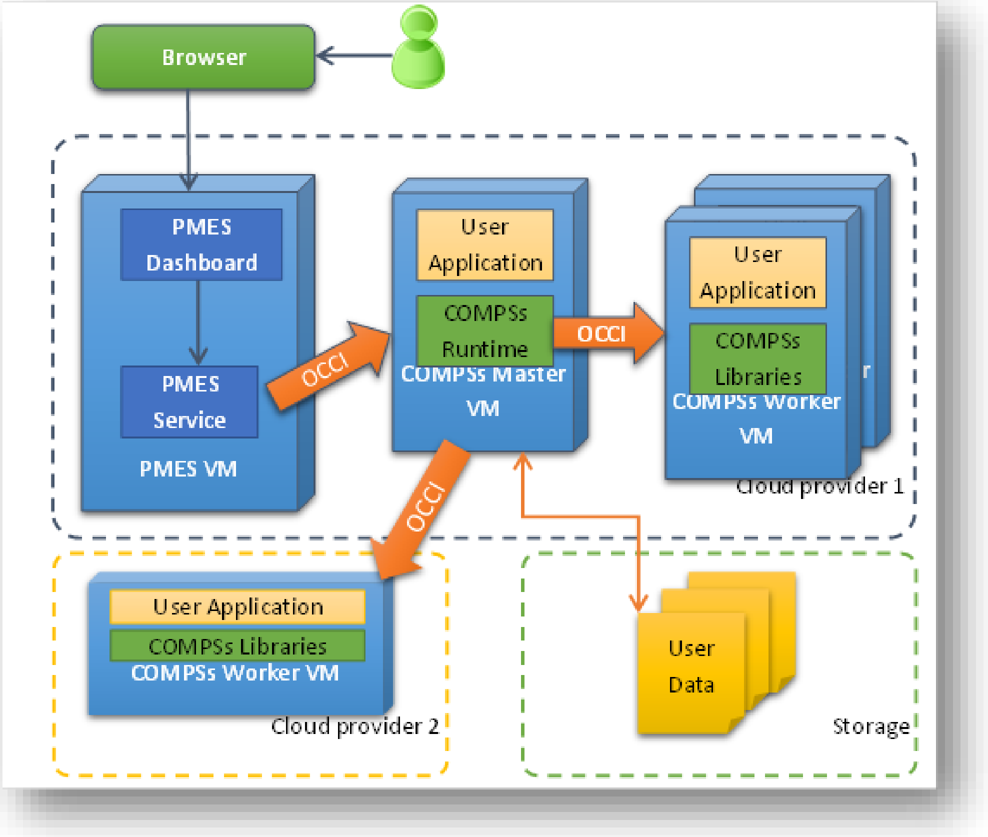


Figure 5. PMES overview

In the context of the MuG project, the PMES will be refactored and extended. Among the main updates we include: the migration to the MEAN stack[20] (which is a framework composed by MongoDB, Express, AngularJS and Node for building dynamic web sites and applications), redesign of the dashboard and of the API, implementation of a new web service and of additional functionalities (e.g. security and management per user).

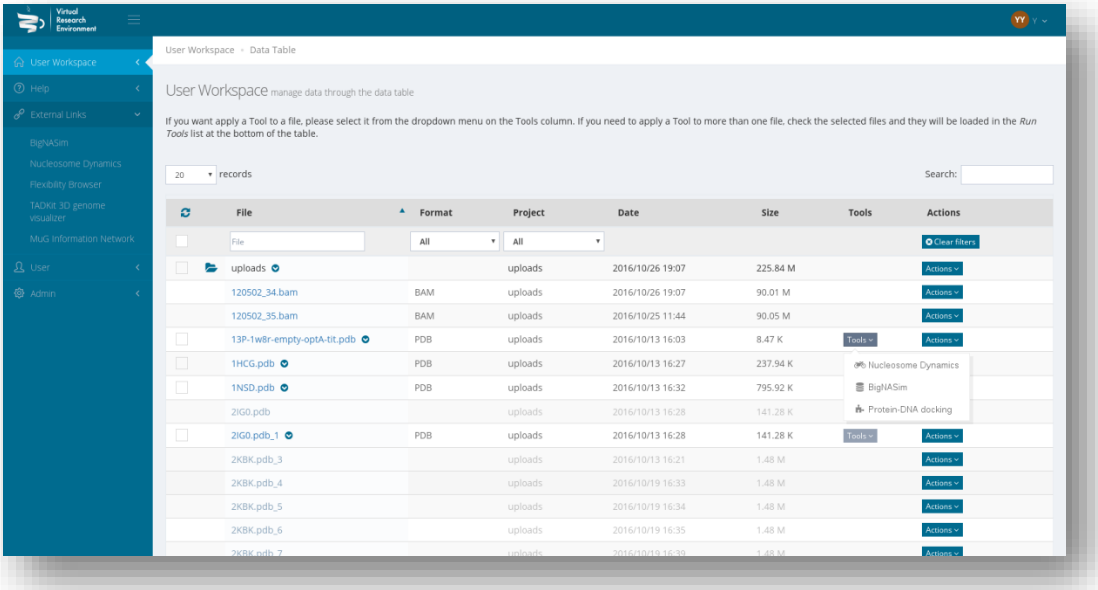
4.3 Data repository components

4.3.1 Database managers

MMB-IRB Data Repository is based on the combination of two database engines, MongoDB[11], and Cassandra[12]. At present it contains reference databases (Section 5.3), especially the Protein Data Bank (PDB), and BigNASim[7], the largest database for nucleic acids simulation trajectories, which corresponds to the atomistic level in MuG data management. Cassandra is a distributed and highly scalable key-value database with a strong user community. Cassandra implements a non-centralized architecture, based on peer-to-peer communication, in which all nodes of the cluster are able to receive and serve queries. Data is stored in tables by rows, which are identified by a key chosen by the database user. In each row, users can add different attributes also identified by a chosen name. To each node of the cluster, a token is assigned and becomes responsible for hosting specific set of rows. The target node for each row is chosen through the partitioner algorithm and the decision is based on the row key and the node token. This characteristic makes Cassandra an excellent manager when data has a fixed and relatively simple structure like simulation trajectories. The MongoDB database in MMB-IRB repository holds reference databases, simulation metadata and pre-calculated analysis results. MongoDB is a fully flexible engine and can store heterogeneous collections of documents. The internal structure of each document does not need to be defined beforehand and can match the data structure used in the interacting software, thus simplifying the use of database documents and external analysis software. The flexibility in structure makes MongoDB the best choice to store heterogeneous data collections. MuG infrastructure will use MongoDB to hold the metadata store, allowing several metadata schemas to coexist. Data hold at MuG repository will use either MongoDB or Cassandra depending on its structure. Other storage models will be also explored (Task 5.3).

4.3.2 Personal workspace

A key structure in MuG VRE will be the personal workspace. A significant amount of data types involved in 3D/4D genomics are contained in large files, where management shows clear difficulties, not only from the transmission point of view, but also due to conversion operations that can be computationally expensive. This requires that MuG VRE accepts asynchronous operations. SGE and PMES (see section 4.2) are already in place to manage asynchronous process management. Asynchronous data management is achieved through a specific application, the personal workspace that in turn is the central landing place when accessing the infrastructure (Figure 6). The personal workspace has a private filesystem-style structure and will allow the user to manage data, including to upload/download data from the system, recover reference data or data from experiments in public repositories, incorporate such data to workflow analysis, and send data to MuG's multi-scale browser. Access to the workspace is currently based on HTTP communications, although in the future other protocols will be explored. Access to the workspace is restricted (controlled through the appropriate authentication protocols).



User Workspace manage data through the data table

If you want apply a Tool to a file, please select it from the dropdown menu on the Tools column. If you need to apply a Tool to more than one file, check the selected files and they will be loaded in the *Run Tools* list at the bottom of the table.

20 records

Search:

	File	Format	Project	Date	Size	Tools	Actions
□	File	All	All				Clear filters
□	uploads		uploads	2016/10/26 19:07	225.84 M		Actions
	120502_34.bam	BAM	uploads	2016/10/26 19:07	90.01 M		Actions
	120502_35.bam	BAM	uploads	2016/10/25 11:44	90.05 M		Actions
□	13P-1w8r-empty-optA-tit.pdb	PDB	uploads	2016/10/13 16:03	8.47 K	Tools	Actions
□	1HCG.pdb	PDB	uploads	2016/10/13 16:27	237.94 K		
□	1NSD.pdb	PDB	uploads	2016/10/13 16:32	795.92 K		
	2IG0.pdb		uploads	2016/10/13 16:28	141.28 K		
□	2IG0.pdb_1	PDB	uploads	2016/10/13 16:28	141.28 K	Tools	Actions
	2KBK.pdb_3		uploads	2016/10/13 16:21	1.48 M		Actions
	2KBK.pdb_4		uploads	2016/10/19 16:33	1.48 M		Actions
	2KBK.pdb_5		uploads	2016/10/19 16:34	1.48 M		Actions
	2KBK.pdb_6		uploads	2016/10/19 16:35	1.48 M		Actions
	2KBK.pdb_7		uploads	2016/10/19 16:39	1.48 M		Actions

Figure 6 Screenshot of MuG personal user workspace

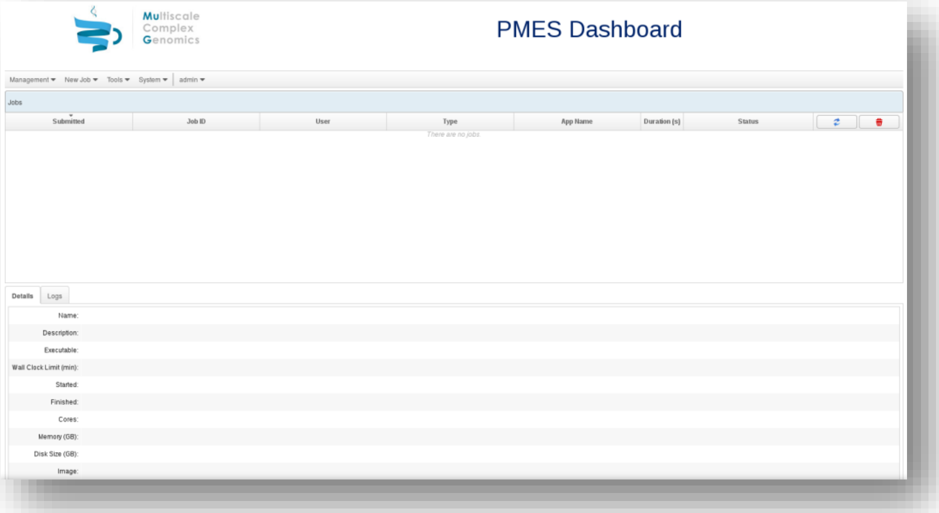
4.4 User access interfaces

Both interactive and programmatic access are available in MuG infrastructure. Users are expected to use web tools for data browsing and manual application management, but also programmatic access for large scale analysis and also for data retrieval will be available. In the initial version, existing applications are using their original user access interfaces. The aim of the development roadmap is to integrate web access to make it data-centric around the personal workspace (section 4.3.2), and the multi-scale browser developed at WP3. The present implementation uses standard authentication procedures. It is expected to incorporate the authentication and authorization infrastructure defined by the Elixir initiative.

4.4.1 PMES Dashboard

PMES provides a graphical web interface which can be used in order to interact with. This interface was designed in order to ease the user interaction and experience with the platform (Figure 7). This web interface represents a dashboard which is useful for simple execution purposes, where users can directly get the information about the infrastructure, manage the applications and jobs, interact for the job submission and track the submissions operations.

Moreover, the PMES dashboard also provides the accounting of the jobs submitted to the infrastructure, as well as some details about its execution.



PMES Dashboard

Jobs

Submitted	Job ID	User	Type	App Name	Duration (s)	Status

There are no jobs.

Name:
Description:
Executable:
Wall Clock Limit (min):
Started:
Finished:
Cores:
Memory (GB):
Disk Size (GB):
Image:

Figure 7. Screenshot of PMES Dashboard

4.4.2 Programmatic access

In order to allow a programmatic access, PMES implements a SOAP based web service according to the Basic Execution Service (BES) standard[21]. BES is an Open Grid Forum (OGF) standard specification for a SOAP service to which clients can send requests to start, monitor, and manage computational activities.

Briefly, the basic operations considered are:

Operation	Description
Create activity	Creates and submits an activity to the infrastructure.
Terminate activity	Terminates a previously created activity.

Get activity status	Retrieves the status of the activity (e.g. running, failed, etc.)
Get activity documents	Gets the activity documents of an activity giving the: JSDLs, jobs status, execution progress, elapsed time and error messages.

Each activity is a set of jobs, and they are described with the Job Submission Description Language (JSDL)[22]. In this description, a set of characteristics of the job can be specified, such as: application name, executable and arguments. However, it can also be used to specify the hardware requirements (CPU, memory, storage, etc.), CPU time, or data staging.

The PMES service can be accessed using web-service clients. Alternatively, PMES also provides a Java API in order to generate the appropriate job definitions and launch their executions.

4.4.3 Galaxy interface

Galaxy[23] is an open source, web-based platform for data intensive bioinformatics research. Galaxy offers a common workspace integrating data and analysis tools, allowing users to design their own workflows managing both public and private data. Galaxy is a very popular interface in bioinformatics, counting with a strong community of developers. Although Galaxy itself can provide some of the intended functionality of MuG VRE infrastructure, this is too complex to be implemented in a single interface. MuG intends to use Galaxy as an additional interface to the infrastructure, allowing to run some of MuG tools and data under a known environment, although using MuG computational infrastructure as backend.

5 PRESENT STATUS AND APPLICATIONS OFFER

The infrastructure is available at http://www.multiscalegenomics.eu/MuGVRE/. The following sections summarize the present status of infrastructure components, applications offer, and data available at the repository. Development of the infrastructure is a continuous process, further improvement will be documented at MuG VRE web site.

5.1 Available components

Table 1 shows a list of the components that are part of the initial prototype, and those that are being currently developed.

Table 1. Status of MuG infrastructure initial deployment

Component	Description	Status
Cloud manager	OpenNebula[13] cloud manager administrates hardware resources & Virtual Machine repository	Installed
COMP Superscalar	Multi-scale programming model	Installed (version 1.2)
Programming Model Enactment system (PMES)[2]	Manages the execution of both stand-alone and COMPSs[19] based workflows. Controls OpenNebula resources through the rOCCI[18] protocol. Provides Programmatic Access to applications.	Installed (version 2.3.2)

Sun Grid Engine	Queuing system to manage workflow execution. Complements PMES for specific applications.	Installed
Personal workspaces	User workspaces managed by traditional file systems, and a noSQL database. User authentication through traditional procedures.	Installed. The initial version maintains application-based workspaces where available. There is ongoing work to integrate them in a unique space. Additional authentication methods under study.
Data Repository	Data storage based in noSQL technology (MongoDB[11] & Cassandra[12])	Data available for 3D atomistic structure and simulation data, powered by the BiGNASim, and MMB-IRB's structural data repositories.
Multi-scale browser	Visual access to data analysis	Being developed at WP3. The prototype is linked to the TADkit tool for chromatin structure visualization, BIGNASim portal for atomistic simulation data, and to sequence data through JBrowse
Data Access APIs	All data will be accessible with the appropriate API based on REST protocol	To be developed

5.2 Applications available

Table 2. Applications available at the prototype infrastructure

Application	Description	Type and Access
Serial-Maker +	General purpose genome annotation tool. Includes Maker[24], Exonerate[25], Augustus[26]	Virtual Machine, access through PMES Dashboard and WS
Bwapipeline	Sequence aligner for ngs data. Includes BWA[27], samtools[28], vcftools	Virtual Machine, access through PMES Dashboard
Bowtie+	Sequence aligner. Includes Bowtie[29], tophat[30], samtools[28]	Virtual Machine, access through PMES Dashboard
BIGNASim	Access to the MuG Section of BIGNASim[7], a general purpose database and analysis portal for Nucleic Acids simulations data	Web access.
Flexibility browser[31]	Access to flexibility data extracted from NA Simulations	Web access.
NucleosomeDynamics	Software suite for analysis of MNase-seq for Nucleosome	Web access.

	positioning related data. Includes NucleR[32].	
TADKit 3D	Access to 3D Representation for chromatin conformation modeled from 3C-Data	Web access.
Genome Browser	Browser for reference genome data for Human, Drosophila and Yeast	Web access.

5.3 MuG Data repository

Table 3 summarizes the data currently available at MuG repositories. Data is accessible through the specific interfaces.

Table 3. Data currently available at MuG repositories

Data Set	Origin and status	Comments
Reference Databases		
Protein Data Bank (MMB-IRB)	RCSB. Weekly update	MongoDB, REST API
Uniprot (MMB-IRB)	EMBL-EBI. Monthly update	MongoDB, REST API
Reference Genomes	EMBL-EBI, Ensembl	Raw files, and specific application formats
Reference Annotation Tracks	Diverse	Displayed in JBrowse. See Annex A for detail
MuG specific data and metadata		
ArrayExpress Nucleosome related experiments	EMBL-EBI ArrayExpress (metadata test set)	MongoDB. Web access
Biosamples Nucleosome related entries	EMBL-EBI Biosamples (metadata test set)	MongoDB. Web access
MuG specific simulation set	MuG partners	Available through BigNASim engine at MuG VRE web site
Nucl. Acids Flexibility Data	MuG partners	Available at MuG VRE web site

6 DEVELOPMENT ROADMAP

This section describes the expected development roadmap regarding the computational and data infrastructures, and its components, for the next twelve months. Results of the development will be included in the future deliverable D5.2 describing the first release of complete MuG infrastructure. The main development objective regarding MuG computational infrastructure will be the **full integration of the software components**. Specific work lines are indicated below.

6.1 User workspace

User workspace is planned as the center of user activity. During the next period we will adapt its initial version to allow a full integration with the different application within MuG. Specific steps:

- Integration of MuG data types in main user workspace. The present version of the workspace interface depends on the enabling applications. A new workspace with a unique interface will be developed to allow all MuG applications and workflows to interact with it. This will also improve the possibility of sharing data between levels.
- Adaptation of the workspace as main scheduling element. PMES and web-based applications will be adapted to use the workspace as data manager.
- Improvement of data transmission will be explored through the use of advanced products like gridFTP (as recommended by the Elixir initiative), or Aspera.
- User workspace will be linked to the newly developed visualization applications, especially MuG multi-scale browser.

6.2 Computational layer

Considering the increase of technologies and infrastructures that are becoming of interest in bioinformatics (e.g. cloud, containers, etc.), one of the main actions that need to be taken is to perform a restructuration of the interfaces with these environments within the available process schedulers, especially the COMPSs runtime. It is expected that developing a shared infrastructure for the different interfaces involved (COMPSs, PMES, software modules) will ease the task.

The availability of several infrastructure interfaces will extend the usability of software modules developed in MuG.

6.3 Storage

MuG's WP4 is in charge of designing the VRE global data model that in turn will define the requirements for storage. In addition to the NoSQL based data repository in use, other storage strategies will be assayed (Task 5.3)

6.4 Software components

The main development tasks will be:

- Harmonize COMPSs and PMES versions. The initial infrastructure as presented here is based on outdated versions of PMES, and COMPSs. Software versions will be updated, and adjusted to each other to produce a stable version of both components.
- Depending on decisions taken regarding storage, or specific requirements of MuG software modules, the optimal characteristics of virtual machines may vary. In case these differences cannot be tackled at run time, a series if virtual machines need to be created. Contextualization procedure will be extended to allow a fine grained control of this issue.
- Design the migration of PMES to the MEAN stack technologies. The web framework and storage database used during the initial implementation of PMES difficult or limits its maintenance and new features addition. In particular the present application framework used in PMES is designed for simple graphical user interfaces, but have little support for emerging technologies like MongoDB, Express, AngularJS and Node, which

form part of the so called MEAN stack. Since the MEAN stack provides newer, easier and simpler mechanisms in order to implement the programmatic access compared to BES, a revision of the programmatic access is envisioned.

- Another consequence from the update is related to the PMES web-service communication protocol. In this scope, the current PMES implementation uses SOAP, which limits the interoperability and is implementation dependent. As an alternative, REST provides a more interoperable communication protocol, and will be aligned with MuG's data access interfaces.
- In terms of security: it is expected to improve the access control through the use of the Elixir authentication and authorization infrastructure[8] and local authentication (Figure 8). Elixir provides services for federated authentication and authorization of end users of Elixir tools and services introducing a common identity to manage and control user access to multiple Elixir services, and to verify their identity and access rights. Thus, in order to have a more controlled environment and to empower security through a two level authentication, a local authentication is also envisioned. The local authentication should be used to enable the access to a user that has already logged in through Elixir. Additionally to Elixir, the authentication could be performed through other authentication services, such as OAuth (Google).

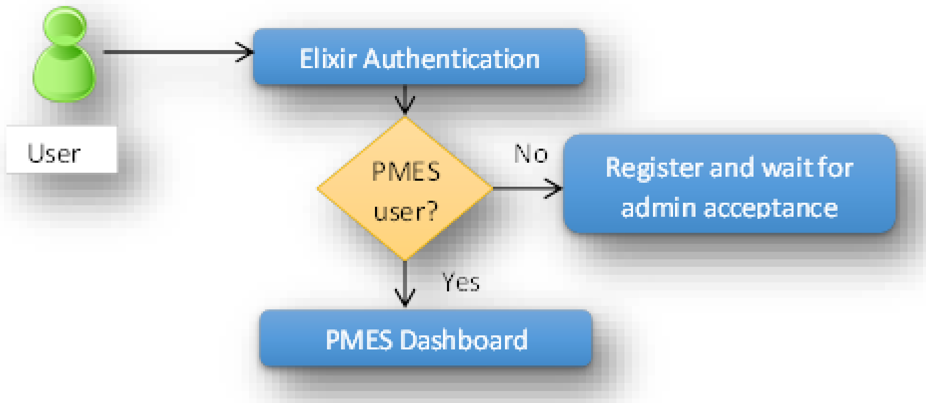


Figure 8. Authentication process

7 REFERENCES

1. EGI. European Grid Infrastructure. https://www.egi.eu
2. EUDAT: the collaborative Pan-European infrastructure providing research data services. https://www.eudat.eu
3. PRACE Research Infrastructure https://www.prace-ri.eu
4. TransNational infrastructure for Plant Genomics. EC FP7 283496, http://www.transplantdb.eu/sites/transplantdb.eu/files/D5.2-transPLANT.pdf
5. Berman, H., Henrick, K., Nakamura, H. and Markley, J.L. (2007) The worldwide Protein Data Bank (wwPDB): ensuring a single, uniform archive of PDB data. *Nucleic Acids Research*, **35**, D301-D303.
6. The Universal Protein Resource (UniProt) in 2010. *Nucleic Acids Res*, **38**, D142-148.
7. Hospital A., Andrio P., Cugnasco C., Codo L., Becerra Y., Dans P.D., Battistini F., Torres J., Goñi R., Orozco M., Gelpí JL. BIGNASim: *A NoSQL database structure and analysis portal for nucleic acids simulation data.* Nucleic Acids Res. 2016 44(D1) p272-8.
8. ELIXIR. A distributed infrastructure for life-science information. https://www.elixir-europe.org/
9. BioSamples. https://www.ebi.ac.uk/biosamples/

10. ArrayExpress. https://www.ebi.ac.uk/arrayexpress/
11. MongoDB. https://www.mongodb.com
12. Cassandra. http://cassandra.apache.org/
13. OpenNebula http://www.opennebula.org
14. Xen, http://www.xen.org/
15. KVM, http://www.linux-kvm.org/page/Main_Page
16. VMWare, http://www.vmware.com/
17. Amazon EC2. http://aws.amazon.com/en/ec2/
18. OCCI, http://occi-wg.org
19. Tejedor E. and Badia RM. *COMP Superscalar: Bringing GRID superscalar and GCM Together*. 8th IEEE International Symposium on Cluster Computing and the Grid 2008.
20. MEAN Stack. http://mean.io/
21. BES. http://www.ogf.org/documents/GFD.108.pdf
22. JSDL. http://www.gridforum.org/documents/GFD.56.pdf
23. Galaxy. http://usegalaxy.org
24. Cantarel B., Korf I., Robb SMC, Parra G., Ross E., Moore B., Holt C., Sanchez Alvarado A., Yandell M. *MAKER: An Easy-to-use Annotation Pipeline Designed for Emerging Model Organism Genomes*. Genome Research. 2008 18(1) p188-96.
25. Slater G.S., Birney E. *Automated generation of heuristics for biological sequence comparison*. BMC Bioinformatics. 2005 6 p31.
26. Stanke M, Waack S. *Gene Prediction with a Hidden-Markov Model and a new Intron Submodel*. Bioinformatics. 2003 19(2) p215-225.
27. Li H. *Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM*. 2013 arXiv:1303.3997v1.
28. Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., Durbin, R., 1000 Genome Project Data Processing Subgroup. *The Sequence Alignment/Map format and SAMtools.* Bioinformatics. 2009 25(16) p2078–2079.
29. Langmead B, Trapnell C, Pop M, Salzberg SL. *Ultrafast and memory-efficient alignment of short DNA sequences to the human genome*. *Genome Biol* 2009 10(3) R25.
30. Trapnell C., Pachter L., Salzberg S. L. *TopHat: discovering splice junctions with RNA-Seq* Bioinformatics 2009 25(9) p1105-1111.
31. Hospital A., Faustino I., Collepardo-Guevara R., González C., Gelpí JL., Orozco M. *NAFlex: a web server for the study of nucleic acid flexibility.* Nucl. Acids Res. 2013 41(1) p47-55.
32. Flores O, Orozco M. *nucleR: a package for non-parametric nucleosome positioning*. Bioinformatics. 2011 27 p2149–50.
33. Saccharomyces Genome Database. http://www.yeastgenome.org
34. Yassour M1, Kaplan T, Fraser HB, Levin JZ, Pfiffner J, Adiconis X, Schroth G, Luo S, Khrebtukova I, Gnirke A, Nusbaum C, Thompson DA, Friedman N, Regev A. *Ab initio construction of a eukaryotic transcriptome by massively parallel mRNA sequencing.* Proc Natl Acad Sci U S A. 2009. 106(9):3264-9.
35. Nagalakshmi U1, Wang Z, Waern K, Shou C, Raha D, Gerstein M, Snyder M. *The transcriptional landscape of the yeast genome defined by RNA sequencing*. Science. 2008 6;320(5881):1344-1349.
36. Zhang Z, Dietrich FS. *Mapping of transcription start sites in Saccharomyces cerevisiae using 5' SAGE.* Nucleic Acids Res. 2005. 33(9):2838-51
37. Kirmizis A, Santos-Rosa H, Penkett CJ, Singer MA, Vermeulen M, Mann M, Bähler J, Green RD, Kouzarides T. *Arginine methylation at histone H3R2 controls deposition of H3K4 trimethylation.* Nature. 2007; 449(7164):928-932.
38. Mavrich TN, Ioshikhes IP, Venters BJ, Jiang C, Tomsho LP, Qi J, Schuster SC, Albert I, Pugh BF *A barrier nucleosome model for statistical positioning of nucleosomes throughout the yeast genome.* Genome Res. 2008 18(7):1073-1083.
39. Hesselberth JR, Chen X, Zhang Z, Sabo PJ, Sandstrom R, Reynolds AP, Thurman RE, Neph S, Kuehn MS, Noble WS, Fields S, Stamatoyannopoulos JA. *Global mapping of protein-DNA interactions in vivo by digital genomic footprinting.* Nat Methods. 2009. 6(4):283-289.
40. Albert I, Mavrich TN, Tomsho LP, Qi J, Zanton SJ, Schuster SC, Pugh BF. *Translational and rotational settings of H2A.Z nucleosomes across the Saccharomyces cerevisiae genome.* Nature. 2007;446(7135):572-576.

41. Guillemette B, Bataille AR, Gévry N, Adam M, Blanchette M, Robert F, Gaudreau L. *Variant histone H2A.Z is globally localized to the promoters of inactive yeast genes and regulates nucleosome positioning.* PLoS Biol. 2005; 3(12):e384.
42. Guillemette B, Drogaris P, Lin HH, Armstrong H, Hiragami-Hamada K, Imhof A, Bonneil E, Thibault P, Verreault A, Festenstein RJ. *H3 lysine 4 is acetylated at active gene promoters and is regulated by H3 lysine 4 methylation.* PLoS Genet. 2011;7(3):e1001354.
43. Liu CL, Kaplan T, Kim M, Buratowski S, Schreiber SL, Friedman N, Rando OJ. *Single-nucleosome mapping of histone modifications in S. cerevisiae.* PLoS Biol. 2005; 3(10):e328.
44. Field Y, Kaplan N, Fondufe-Mittendorf Y, Moore IK, Sharon E, Lubling Y, Widom J, Segal E. *Distinct modes of regulation by chromatin encoded through nucleosome positioning signals.* PLoS Comput Biol. 2008; 4(11):e1000216.
45. Schep AN, Buenrostro JD, Denny SK, Schwartz K, Sherlock G, Greenleaf WJ. *Structured nucleosome fingerprints enable high-resolution mapping of chromatin architecture within regulatory regions.* Genome Res. 2015; 25(11):1757-1770.
46. Xu W, Aparicio JG, Aparicio OM, Tavaré S. *Genome-wide mapping of ORC and Mcm2p binding sites on tiling arrays and identification of essential ARS consensus sequences in S. cerevisiae.* BMC Genomics. 2006 26;7:276.
47. Eaton ML, Galani K, Kang S, Bell SP, MacAlpine DM. *Conserved nucleosome positioning defines replication origins.* Genes Dev. 2010;24(8):748-753
48. Rhee HS, Pugh BF. *Genome-wide structure and organization of eukaryotic pre-initiation complexes.* Nature. 2012 18;483(7389):295-301.
49. Mayer A, Lidschreiber M, Siebert M, Leike K, Söding J, Cramer P. *Uniform transitions of the general RNA polymerase II transcription complex.* Nat Struct Mol Biol. 2010; 17(10):1272-1278.
50. Rhee HS, Pugh BF *Comprehensive genome-wide protein-DNA interactions detected at single-nucleotide resolution.* Cell. 2011 9;147(6):1408-1419.
51. Deniz Ö, Flores O, Aldea M, Soler-López M, Orozco M. *Nucleosome architecture throughout the cell cycle.* Sci Rep. 2016 28;6:19729.
52. Filion GJ, van Bemmel JG, Braunschweig U, Talhout W, Kind J, Ward LD, Brugman W, de Castro IJ, Kerkhoven RM, Bussemaker HJ, van Steensel B. *Systematic protein location mapping reveals five principal chromatin types in Drosophila cells.* Cell. 2010 15;143(2):212-24.
53. Mavrich TN, Jiang C, Ioshikhes IP, Li X, Venters BJ, Zanton SJ, Tomsho LP, Qi J, Glaser RL, Schuster SC, Gilmour DS, Albert I, Pugh BF. *Nucleosome organization in the Drosophila genome* Nature. 2008 15;453(7193):358-62.

8 ANNEX A. Annotation tracks available at MuG Genome browser

Data	***Source***
Saccharomyces cerevisiae	
Gene and Gene predictions	Saccharomyces Genome Database[33]
Gene structure / UTRs / transcribed regions	Yassour et al, 2009 [34]
Gene Models / introns / 5' 3' UTR's / unannotated transcripts	Nagalakshimi et al. 2008[35]
Transciption Start sites	Zhang, Z and Dietrich FS. 2005[36]
Chromatin modifications	Kirmizis A. et al. 2007[37]
Nucleosome positions	Mavrich et al. 2008[38]
Digital genomic footprinting	Hesselberth et al. 2009[39]
H2A.Z nucleosome positions	Albert et al. 2007[40]
H2A/H2B, H2A.Z/H2A.Z, H2A.Z/H2B log2 ChIp chip ratio	Guillemette et al. 2005[41]
H3K4ac_set1D_on_WT, set1D_H3K4ac_on_H3, WT_H3K4ac_on_H3, WT_H3K4me3_on_H3	Guillemette et al. 2011[42]
anti-Ac, H2AK7aci, H2BK16ac, H3K14ac, H3K18ac, H3K4me1, H3K4me2, H3K4me3, H3K9ac, H4K12ac, H4K16ac, H4K5ac, H4K8ac, mock, RNA PolII ChIP_chip	Liu et al. 2005[43]
predicted average nucleosome occupancy, predicted nucleosome potential score, nucleosome sequence read count	Field et al. 2008[44]
nucleosome positions, nucleoatac signal, nucleosome calling occurrences	Schep et al. 2015[45]
ORC, Mcm2p binding, ARS sequences	Xu et al. 2006[46]
ORC, ARS, Nucleosome positioning	Eaton et al. 2010[47]
TATA_elements	Rhee and Ough 2012[48]
Bur1, Cet1 (Capping enzyme), Ctk1 Elf1, Kin28 (TFIIH), Paf1, Pcf11, Ser2P (RNA Pol II), Ser5P (RNA Pol II), Ser7P (RNA Pol II), Rpb3 (RNA Pol II), Spn1 (lws1), Spt16, Spt4, Spt5, Spt6, Spt6deltaC, Tfg1 (TFIIF), TFIIB	Mayer et al. 2010[49]
Gal4, Phd1, Rap1, Reb1	Rhee and Pugh 2011[50]
Nucleosome architecture through cell cycle	Deniz et al. 2016[51]
Drosophila melanogaster	
Genes, Transcripts	
Chromatin types through protein binding sites	Filion et al. 2010[52]
Nucleosome organization	Mavrich et al 2008[53]
Homo sapiens	
Refseq Genes	
Gencode Genes	