# Managing Hypothesis of Scientific Experiments with PhenoManager[1]

Leonardo Ramos[1], Fabio Porto[2], Daniel de Oliveira[1]

[1] Institute of Computing – Universidade Federal Fluminense (IC/UFF), Niterói, Rio de Janeiro, Brazil
leoslramos@id.uff.br, danielcmo@ic.uff.br
[2] National Laboratory of Scientific Computing (LNCC), Petrópolis, Rio de Janeiro, Brazil
fporto@lncc.br

**Abstract.** Scientific research based on computer simulations is complex since it may involve managing the enormous volumes of data and metadata produced during the life cycle of a scientific experiment, from the formulation of hypotheses to its final evaluation. This wealth of data needs to be structured and managed in a way that makes sense to scientists so that relevant knowledge can be extracted to contribute to the scientific research process. In addition, when it comes to the scope of the scientific project as a whole, it may be associated with several different scientific experiments, which in turn may require executions of different scientific workflows, which makes the task rather arduous. All of this can become even more difficult if we consider that the project tasks must be associated with the execution of such simulations (which may take hours or even days), that the hypotheses of a phenomenon need validation and replication, and that the project team may be geographically dispersed. This article presents an approach called PhenoManager that aims at helping scientists managing their scientific projects and the cycle of the scientific method as a whole. PhenoManager can assist the scientist in structuring, validating, and reproducing hypotheses of a phenomenon through configurable computational models in the approach. For the evaluation of this article was used SciPhy, a scientific workflow in the field of bioinformatics, concluding that the proposed approach brings gains without considerable performance losses.

Categories and Subject Descriptors: H.2.4 [**Database Management**]: Systems; H.2.8 [**Database Management**]: Database Applications

Keywords: Scientific Experiment, Phenomena, Hypothesis, Scientific Workflows, Project Management

## 1. INTRODUCTION

In recent years, there has been a growth in the use of computer simulations in scientific experiments [Hey et al. 2012; de Oliveira et al. 2019]. According to de Oliveira *et al.* [2019], today's scientific experiments rely heavily on the analysis of data generated from complex computer simulations. There are several existing approaches to modeling, managing, monitoring, and debugging simulation-based experiments. Many users implement their own scripts and programs, while others model their experiments using Workflows Management Systems (WfMS) [Deelman et al. 2007; Deelman et al. 2009], Scientific Gateways [Gesing et al. 2019; Ocaña et al. 2020] or Data Intensive and Scalable Computing (DISC) frameworks like Apache Spark [Zaharia et al. 2016] or Hadoop [Karau et al. 2015].

However, none of these approaches is able to represent all the concepts involved in the scientific method as discusssed by Mattoso *et al.* [2010] and Allen *et al.* [2017]. Currently, most of existing approaches focus only on representing the simulations that are executed in the context of a given experiment [de Oliveira et al. 2019; Marinho et al. 2017]. However, the starting point of a scientific

[1]This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. This study was also financed in part by CNPq and FAPERJ.

investigation is the description of a phenomenon. The studied phenomenon occurs in some space-time, in which selected physical quantities are observed [Porto et al. 2015]. Scientific hypotheses conceptually interpret the studied phenomenon through its representation and through mathematical models. The *in silico* hypothesis testing involves performing experiments, representing the mathematical models, and confronting data generated from complex computer simulations [Porto et al. 2015]. That is, in order to confirm or refute a hypothesis, experiments must be defined, and these experiments may demand the execution of several simulations implemented in different ways, as presented in Figure 1 (a workflow in WfMS Pegasus, a Python script and an application implemented in Spark).

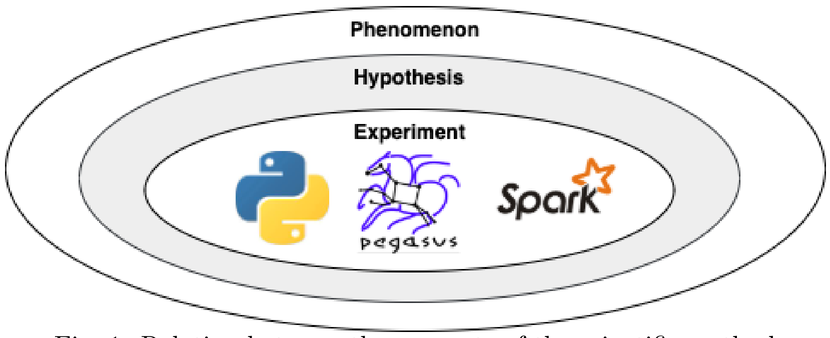


Fig. 1: Relation between the concepts of the scientific method

Since there is no association between the execution of the simulations, the observed phenomena and the defined hypotheses, it is up to the scientist to manage all this knowledge in a manual and *ad-hoc* (commonly error-prone) way. In other words, if the scientist needs to discover which executions of a particular script or workflow are essential to refute the $\alpha$ hypothesis, one will have to record this information on his own. In an even more complex scenario, the validation of a scientific hypothesis may require the execution of several distinct workflows, scripts or MapReduce applications that may execute in distributed, high-performance computing (HPC) environments, such as computing clouds and supercomputers [Vaquero et al. 2008].

Thus, it would be interesting for scientists to have access to an approach that helps in the management of the scientific project as a whole and in supporting the scientific method, helping in the documentation, sharing of the obtained data and in easing the reproduction of the performed experiments, which represents an open challenge. Therefore, this article presents an approach called `PhenoManager` that aims to support the management and validation of scientific hypotheses in an integrated way with the execution of experiments, either via workflows, scripts or MapReduce applications. The `PhenoManager` manages the scientific project since the design stage, from setting up the execution model, to the validation and reproduction of the experiments, using provenance data [Freire et al. 2008]. In addition, the `PhenoManager` allows for the scientist to execute experiments in HPC environments (such as the Santos Dumont supercomputer[2] of the National Laboratory for Scientific Computing - LNCC) and integrates with existing tools such as *SciManager* [Ramos et al. 2016] system, which manages team tasks in scientific projects, in a same {software ecosystem.

This article extends our previous study performed in [Ramos et al. 2019] entitled "Phenomanager: um Sistema de Gerência de Hipóteses de Fenômenos Científicos". In this extended version we enriched the Experimental Evaluation Section with brand new analysis. We have also added a Background section that discusses important concepts. The remainder of the article is organized into three sections in addition to the introduction. Section 2 brings backgound. Section 3 presents an overview of the `PhenoManager`. Section 4 presents the evaluation of `PhenoManager`. Section 5 brings related work, and finally, Section 6 concludes this article.

[2]`https://sdumont.lncc.br/`

## 2. BACKGROUND

In the context of this article, there are important concepts to be explained: (i) scientific projects, (ii) scientific experiments, (iii) scientific hypothesis, (iv) phenomena, and (v) scientific workflows. In addition, it is important to define the life cycle of a scientific experiment and the life cycle of a scientific project.

### 2.1 Basic Concepts

A *scientific project* is the highest level unit of work and it works as a planning instrument. In this way, a scientific project must define the goal(s) of the research and how the research will be conducted. Each project has a group of people to whom tasks and responsibilities concerning the project will be assigned. Depending on the number of goals to be achieved, a scientific project may consist of one or more *scientific experiments.* A scientific experiment, according to the Oxford dictionary, may be defined as "a test performed under controlled conditions, which is performed to demonstrate a known truth, examine the validity of a hypothesis, or determine the effectiveness of something previously unexplored". There are several types of scientific experiments, namely: *in vivo, in vitro, in virtuo and in silico* [Travassos and Barros 2003]. In this article, the focus is on supporting *in silico* experiments, *i.e.* those that are based on computer simulations. In the context of this article, the term *scientific experiment* will be consistently used to refer to simulation-based scientific experiments. This type of experiment is found in many different domains, *e.g.*, bioinformatics [de Oliveira et al. 2013; Coutinho et al. 2011; Ocaña et al. 2013], deepwater oil exploration [de Oliveira et al. 2009], mapping of celestial bodies [Porto et al. 2018], *etc.* All of these examples can be considered large-scale because they consume and produce a large volume of data.

A scientific experiment is associated with a set of controlled actions (*i.e.*, a protocol) with a well-defined goal. These actions include multiple tests, and their results are usually compared with each other to accept or refute a *scientific hypothesis.* In order to confirm or refute a hypothesis, experiments must be defined, and these experiments can demand the execution of several simulations that may be implemented in different ways, *e.g.*, a workflow in WfMS Pegasus, a Python script and a MapReduce application implemented in Hadoop. Large-scale experiments can be executed multiple times to produce a result. Since each execution may demand many resources and time to finish, these experiments usually require parallel techniques and distributed environments (HPC and DISC) to produce results in a feasible time. Commonly, these scientific experiments are composed by the chaining and execution of different programs, which may consume multiple combinations of parameters and large amounts of data. Thus, scientific experiments can be modeled as *scientific workflows.* Scientific workflows can be defined as an abstraction for modeling the flow of activities and data in an experiment. Thus, we have the following interconnection between the concepts: a scientific workflow is part of a given experiment, which, in turn, is in the context of a scientific project. A workflow is the concrete representation of a scientific experiment and symbolizes one of the possible executions of the experiment (*i.e.*, trials). Each experiment follows a well-defined life cycle as defined by [Mattoso et al. 2008], as discussed following.

### 2.2 The Life Cycle of Scientific Experiments and Scientific Projects

The life cycle of a large-scale scientific experiment, according to Mattoso *et al.* [2008] , consists of multiple interactions, to be performed by scientists during the course of an experiment. Since there may be several experiments coexisting within the same scientific project, we may have more than one life cycle, overlapping or interacting with each other. Fig. 2(a) presents a simplified version of the cycle proposed by [Mattoso et al. 2008] where the main phases can be identified: (i) Composition, (ii) Execution and (iii) Analysis. In the composition phase, scientists define the structure of the scientific experiment, *i.e.*, the logical sequence of activities, the types of input/output data and parameters that must be provided. The execution phase is responsible for executing a workflow specification on a given WfMS. Finally, the analysis phase supports the interpretation and evaluation of the data generated by the composition and execution phases. This phase is highly dependent on provenance data [Freire et al. 2008]. Provenance data represents the history of the experiment, from its specification, to the parameters used, and the execution times for each activity in the experiment. With these data scientists can audit multiple executions and guarantee the reproducibility of the experiment.

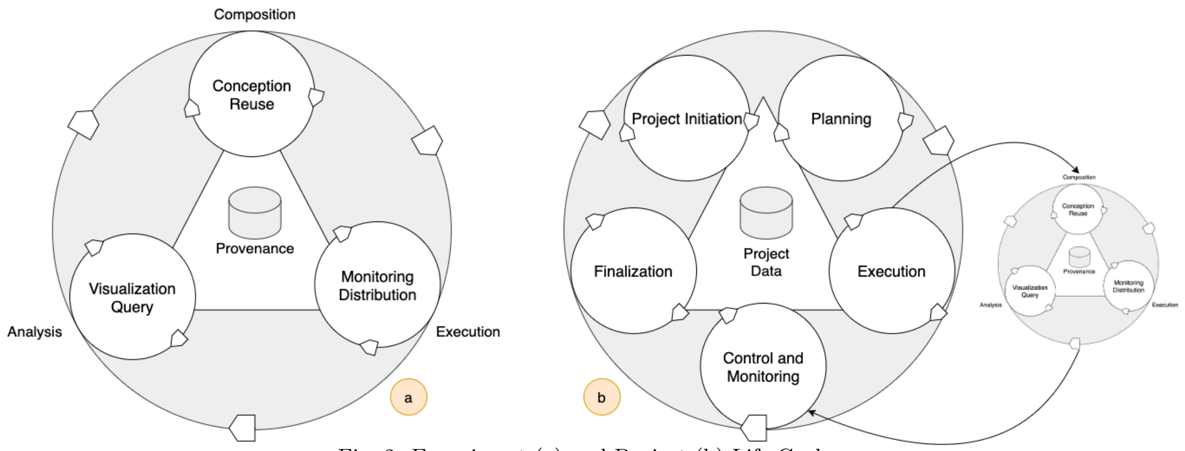


Fig. 2: Experiment (a) and Project (b) Life Cycles

In the same way that the life cycle of an experiment presents defined phases and characteristics, the life cycle of a scientific project (when based on project management techniques/methodologies) consists of a set of phases that are performed by the project members, as presented in Fig. 2(b). Each of the phases presented in Fig. 2(b) has several activities performed by project members. These activities are considered to be *finished* when the users provide the *deliverables*. In the first phase of this cycle, called *Project Initiation*, the project goals are discussed, which methodologies will be used and who will be the members (scientists) and managers (*e.g.*, principal investigators). As a result of this phase a document is produced that represents a opening term of the project. The opening term contains the goals, the components of the team, available budget, *etc.* The second phase, called *Planning*, discusses the high-level requirements to create more specific requirements that guide the creation of tasks. These requirements allow for scientists to define one or more scientific hypotheses. To confirm or refute the hypotheses, one or more tasks are created and assigned to project members. For example, in a bioinformatics project a high-level requirement is “Identify drugs to fight Malaria”. This high-level goal can be detailed in specific goals such as “Perform Phylogenetic Analysis” and “Molecular Modeling Study”. Each of these more specific goals must be associated with an experiment, and one or more tasks can be created that are associated with the goal. For example, for the “Perform Phylogenetic Analysis” we can create tasks entitled “Create the Phylogenetic Analysis Workflow”, “Select the Input Data”, *etc.* Each of these tasks is associated with an experiment and a workflow or script, and must be assigned to one or more members of the project. The third phase is the *Execution* phase of the project. In this phase, the tasks specified in the previous phase are actually executed; the workflows or scripts are implemented, tested, executed, and the data is analyzed. The fourth phase is the *Control and Monitoring* phase, where the tasks performed in the execution phase phase are checked and audited by the project managers. Finally, in the *Finalization* phase, a final document is obtained that presents both the results generated and some project insights.

## 2.3 A Model for Representing Scientific Hypothesis

In this subsection, we present the data model for representing scientific hypotheses used in this article [Porto et al. 2015]. The conceptual model interprets the role of *in silico* data, highlighting formulation and validation of scientific hypotheses, as presented in Fig. 8.

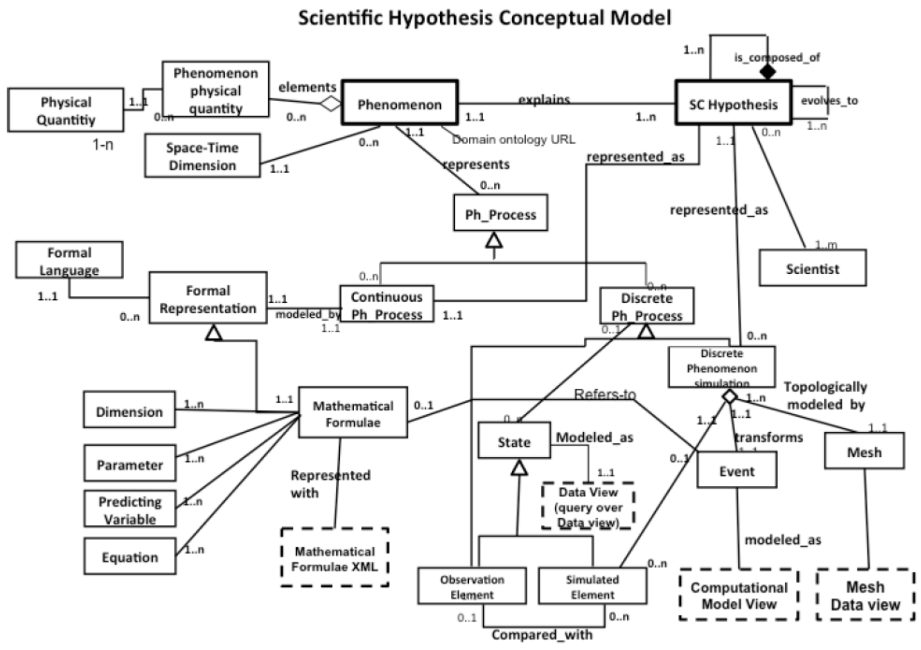


Fig. 3: Scientific Hypothesys Conceptual Model [Porto et al. 2015]

One of the applications of the conceptual model is its implementation as a database that includes data and metadata about the scientific life cycle of the experiment [Porto et al. 2015]. This model reflects the entities involved during an *in silico* experiment life cycle. The domain is structured around the following main concepts: *Phenomenon*, *Hypotheses*, and *Process of Hypotheses and Phenomena*. A scientific hypothesis conceptually represents a formal model that provides a reasonable interpretation for the studied phenomenon. A hypothesis is formally expressed and modeled in a computational representation. The formalization of a scientific study can be provided by a mathematical model.

The representation of the mathematical model is relevant as it is the basis for a consistent mapping between its formal mathematical representation and its data representation. In this model, the specification of a scientific hypothesis is performed from two perspectives: a continuous and a discrete process. The first refers to the mathematical model, discussed previously, representing the studied phenomenon. The latter corresponds to the computational representation of the hypothesis that induces discrete transformations of the state-phenomenon that lead to the generation of simulation data.

## 3. PROPOSED APPROACH: PHENOMANAGER

In this section, we present the architecture of `PhenoManager` and a brief guide on how to use `PhenoManager` in practice.

### 3.1 Architecture

Fig. 4 presents the architecture of the `PhenoManager` and its main components. The `PhenoManager` can be organized into six functional layers: (i) Authentication Layer, (ii) Environment Management Layer, (iii) Execution Layer, (iv) Data Layer, (v) Query Layer, and (vi) Web Portal. The proposed architecture is based on the layered pattern of software architecture [Bass et al. 2003]. The core idea is to split up the software code into layers, where each layer has a certain responsibility and provides a service to a different layer. Each layer may have several components, and each component has a specific task in the system and can be interchangeable according to the needs of the user. In the proposed approach we group the components that present similar and interchangeable goals into the same layer.

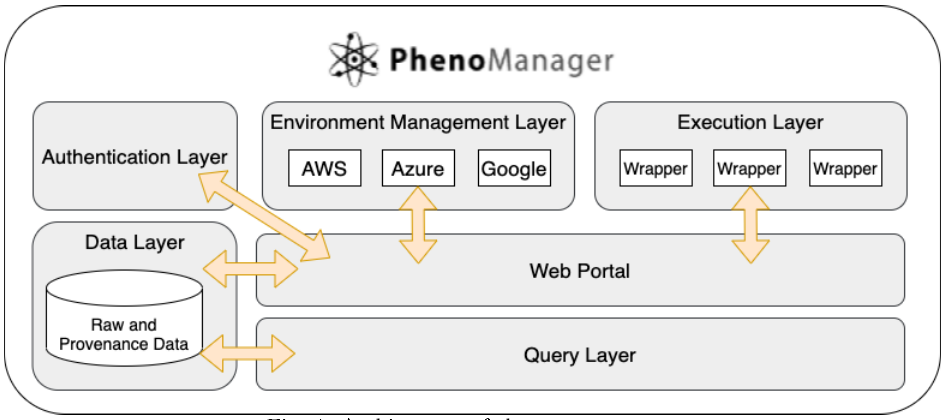


Fig. 4: Architecture of the PhenoManager

The *Authentication Layer* is responsible for managing the access credentials to the PhenoManager. This layer is critical since unpublished search data is handled by PhenoManager. Credential control is performed at two levels. At the *Personal Profile* level, each user of the tool configures/completes their personal information and loads credentials for access to the various environments (*e.g.*, Amazon AWS). At the *User Groups* level, the administrator (with privileges to create groups) search, select and group existing user profiles, and, once the group is created, share the same privileges. Each user or group can access the data and functionalities provided by PhenoManager according to the following privileges: (i) Read permission: the user and/or group can only view the data; (ii) Write permission: the user and/or group can read and submit information; (iii) Administrator permission: in addition to the aforementioned permissions, the user and/or group can register permissions for other users and groups;

The *Environment Management Layer* is responsible for configuring distributed environments for executing simulations. For each different environment, a component must be developed with the calls to its specific API. The PhenoManager already provides integration with three different types of environments: Cluster, Cloud (Amazon AWS) and SSH. Additionally, one can configure a VPN connection for these environments by selecting between Cisco VPN and VPN default. For the Cloud environment, one can configure in detail, the types and images of the virtual machines that will be built in the Amazon AWS environment.

The *Execution Layer* is responsible for invoking WfMSs, scripts or external applications in the HPC environments. For each different system or application to be invoked, a specific wrapper must be provided (since the PhenoManager needs to be aware of the invocation process of the external application). The call to the wrappers is asynchronous, so the service of this layer can be scaled to more instances, thus increasing the throughput of parallel executions of scientific simulations for different users. Thus, through this artifice, we try to guarantee parallelism and high availability of the PhenoManager.

The *Data Layer* contains the provenance database and the raw data produced by the simulations. In addition, this layer contains a series of extractors responsible for accessing the provenance database or log of the external application and loading the information into the provenance database of the PhenoManager. In its current version, the provenance database is modeled is PostgreSQL DBMS and the raw data is loaded into Google Drive. The *Query Layer* is responsible for allowing the scientist to submit queries to the data managed by the system. This layer provides an API that abstracts the queries to the data in a way that makes it easy for scientists and other applications that consume this service to handle them. The API allows for the scientist to submit queries containing filters, sorting, aggregation functions, and field projections on all the entities exposed in the `PhenoManager` data model. Another important aspect is that the *API* only responds successfully if the user passes correct credentials in the request header. In addition, the Query Layer is responsible for exporting packages called *Research Objects* (RO) [Holl et al. 2013], which contain both the queried metadata and the raw data produced by the simulation. By means of ROs, scientists can reproduce a certain simulation, an experiment or check if a hypothesis has been effectively validated.

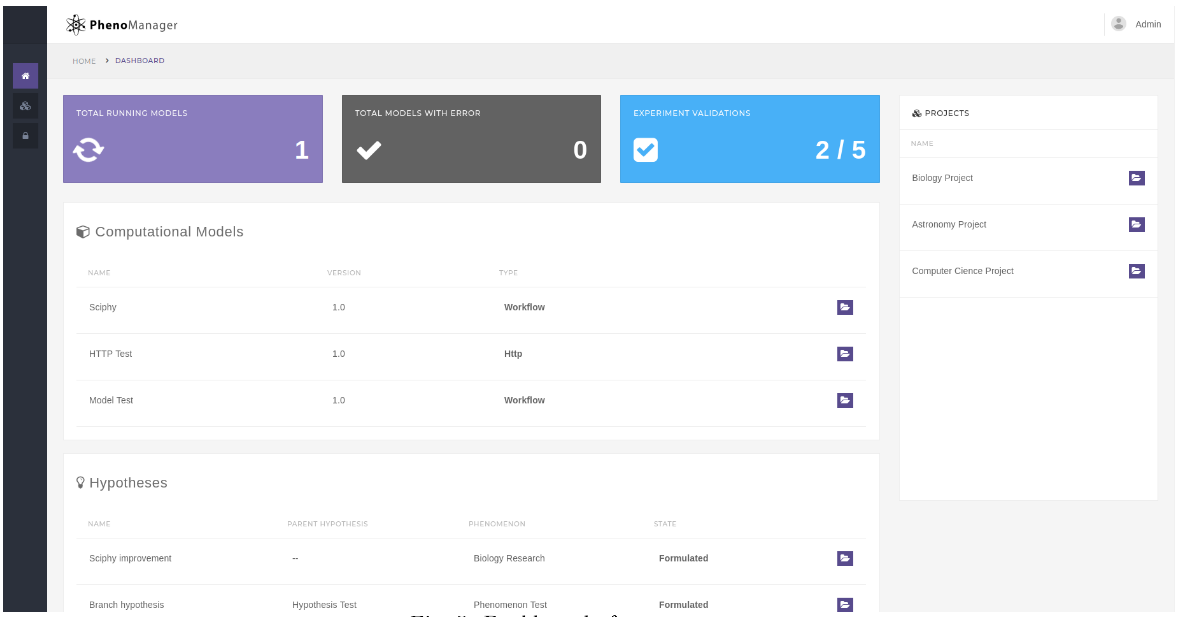
Finally, the *Web Portal* is responsible for all the interface with the scientist and the integration with the other layers. Using it the scientist registers the observed phenomena, the associated hypotheses, his/her experiments and the models that execute the simulations of each experiment (*e.g.*, workflow, script or application). Furthermore, through the Web Portal, the scientist is able to effectively execute his/her simulations and query the collected provenance data in an integrated manner,*i.e.,* if a same experiment is composed of several workflows and applications, the queries to the provenance base consider all the simulations as part of the same experiment, which does not occur in the existing tools that manage the computational models in an isolated manner [Deelman et al. 2009].

`PhenoManager` was developed in Java and follows the architectural pattern of APIs as microservices, *i.e.*, each component is a small, autonomous web service that provides only one functionality [Newman 2015]. All microservices were built using the Spring Boot framework, which already provides support for developing applications in this pattern in a fast and low verbose manner. For data security and authentication between components, the Spring Security framework was used. The development of the interfaces, templates and screens of the Web Portal was done with AngularJs. Since each service that composes the `PhenoManager`architecture is completely isolated from the others, scalability becomes one of the key points of this ecosystem. To ensure the invocation of external applications asynchronously, the open source RabbitMQ message Broker was chosen. The souce code of `PhenoManager` can be obtained at `https://github.com/UFFeScience/Phenomanager`.

## 3.2 `PhenoManager` in Practice

The Web Portal of `PhenoManager` is presented in Fig. 5. The scientist first accesses the *dashboard* (Fig. 5(a)) that presents all the simulation executions in progress, finished, *etc.*, but the scientist can also register a phenomenon and hypotheses in the `PhenoManager`. After defining the access credentials, profiles and user groups, the `PhenoManager` features are enabled. Thus, the first action that the user can perform in the system is to register a scientific project, which is the highest level work unit. A project has a name, a description/documentation and phenomena associated. If the user is an administrator or has write priviledge, it can create a project, edit, as well as create/edit the phenomena that one intends to study. Users with read-only priviledges are only able to view project information.

After configuring the project, the user needs to configure the phenomena and hypotheses from which a described phenomenon refers. Similarly to the scientific project, the phenomenon and the hypothesis have a name and a description. In the case of the hypothesis, it may contain several other derived hypotheses that describe derivations of a study [Gonçalves and Porto 2013]. According to the results obtained during executions of experiments associated with a given hypothesis, the scientist can change the hypothesis' state. A hypothesis can assume the following states in `PhenoManager`: *Formulated*: a newly created hypothesis; (i) *Validated*: a hypothesis validated by experiments; (ii) *Confirmed*: hypothesis proved to be true, however, it lacks validation; (iii) *Improved*: a hypothesis that had an improvement in its formulation; (iv) *Refuted*: a hypothesis that has been refuted;


PhenoManager

Admin

HOME > DASHBOARD

TOTAL RUNNING MODELS 1

TOTAL MODELS WITH ERROR 0

EXPERIMENT VALIDATIONS 2 / 5

Computational Models

| NAME | VERSION | TYPE |
|---|---|---|
| Sciphy | 1.0 | Workflow |
| HTTP Test | 1.0 | Http |
| Model Test | 1.0 | Workflow |

Hypotheses

| NAME | PARENT HYPOTHESIS | PHENOMENON | STATE |
|---|---|---|---|
| Sciphy improvement | -- | Biology Research | Formulated |
| Branch hypothesis | Hypothesis Test | Phenomenon Test | Formulated |

PROJECTS

| NAME |
|---|
| Biology Project |
| Astronomy Project |
| Computer Cience Project |

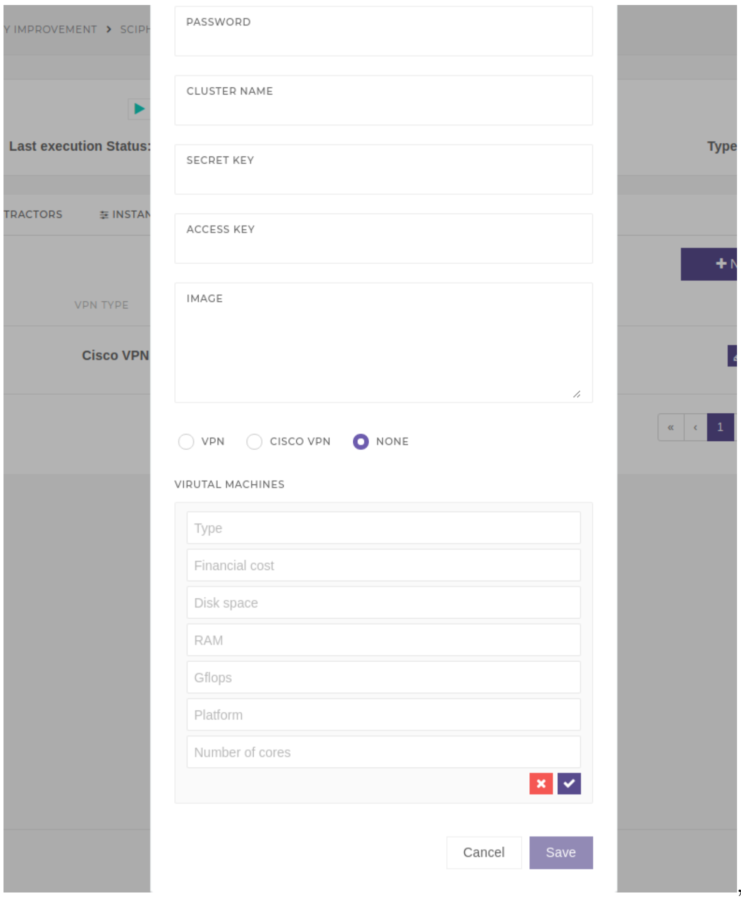
Fig. 5: Dashboard of PhenoManager

The next step is to register the scientific experiments that play a fundamental role of validating (or not) the hypothesis. The scientific experiment also contains name, description, and the computational models that are the concrete representation of the experiment. Furthermore, as it is a conceptual modeling of an execution model, it may have a guideline of the parameters that the computational model, in turn, uses for its execution. In addition, it is possible to create checkpoints for the validation of an experiment. Validation items are entities that aim at determining a guideline on how an experiment can be validated. The same experiment may contain several validation items and by selecting an item as validated, the scientist is able to *upload* files that evidence that this checkpoint was in fact validated.

The creation and configuration of an execution model inside the PhenoManager must follow these steps: (i) Registration of the Computational Environment in which the model will execute (*SSH*, *Cluster*, *Amazon*), as shown in Fig. 6; (ii) Registration of the artifact that will be executed in the environment configured in the previous step (*e.g.*, a program, a workflow); (iii) Registration of the parameters, which represent the values consumed by the execution of an artifact; and (iv) Registration of the execution data extractor(s), which are components responsible for extracting the provenance data.

In the execution environment configuration step, the scientist has the option to configure three different environment types: *Cluster*, *Cloud (Amazon AWS)* and *SSH*. Furthermore, it is possible to configure *VPN* connection for such environments, being able to select between *Cisco VPN* and *default VPN*. When the user selects the *Cloud* environment, it is possible to configure, in detail, the types and images of the virtual machines that will be deployed in the *Amazon AWS* environment (Fig. 6). When the user selects the type *Cluster*, she has to specify the resources that will be allocated and used inside the *batch* file of the *job*.

After configuring the environment, the user can start configuring the artifacts to be executed. One can choose three types of artifacts (called Executor in the interface): *WebService*, *Workflow*, *Command* and *Executable*. A *WebService* executor can perform *REST* and *SOAP* calls. For the *Workflow* type, a compressed *.zip* file is expected, with the programs and the workflow system responsible for managing the *Workflow* execution. If the user chooses the *Executable* type, an executable code is expected. Finally, the type *Command*, expects a text with the command line that will be executed in the



PASSWORD

CLUSTER NAME

SECRET KEY

ACCESS KEY

IMAGE

( ) VPN ( ) CISCO VPN [✓ NONE]

VIRUTAL MACHINES

Type

Financial cost

Disk space

RAM

Gflops

Platform

Number of cores

Fig. 6: Defining a Cloud Computational Environment in PhenoManager.

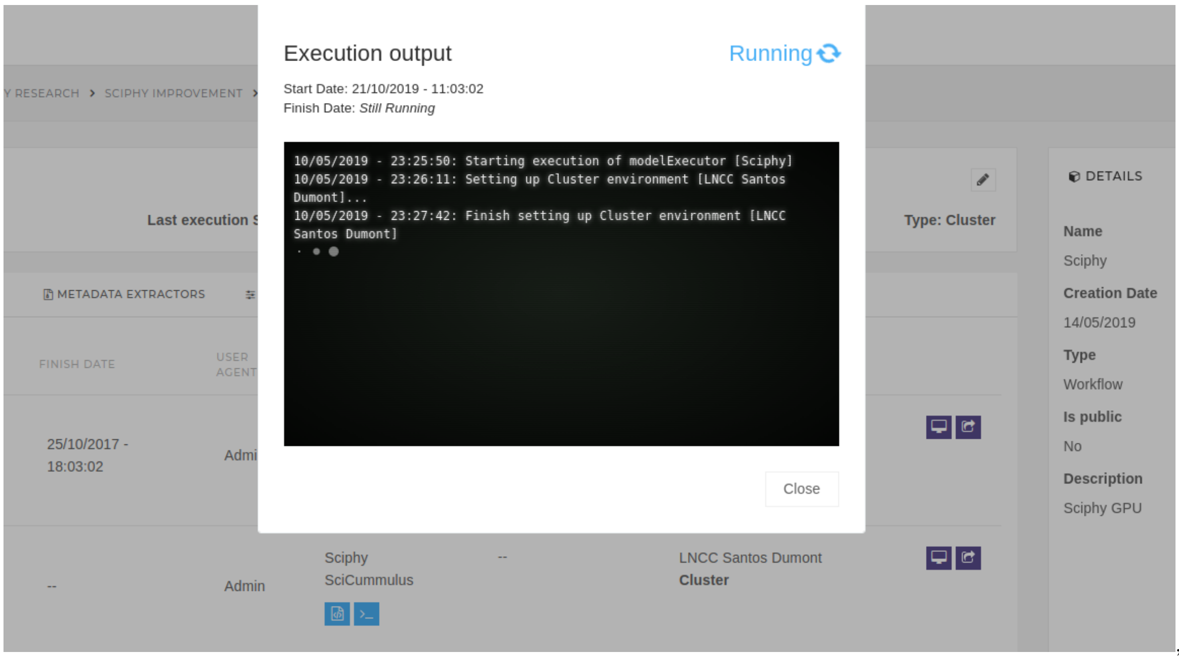
configured environment.

After the artifact configuration is performed, it is possible to start its execution and the asynchronous execution service will be in charge of orchestrating the execution. The provenance data and *logs* of the execution are displayed in real time as the execution proceeds, as one can see in Fig. 7. In Fig. 8, we show the provenance of a specific computational model.

Once the executions have started or have finished, the user is able to query the PhenoManager provenance database. An example query that can be submitted is "*What are the names and versions of the models used in the validation of a hypothesis with a specific name ("sciphy")?*" After processing the query, a *Research Object* containing the query response and associated data is generated for download. Fig. 9 presents a fragment of the generated *Research Object*.

## 4. EVALUATION

This section presents the experimental evaluation of the proposed approach. We use a scientific workflow from the bioinformatics domain as a case study. The experiments were separated into two parts. In the first part, we evaluated the overhead of PhenoManager when executing a workflow. In the second, we evaluated PhenoManager in terms of its usability, using the Technology Acceptance Model (TAM). Thus, the key idea of this section is to evaluate the efficiency and usability of the PhenoManager as well as its limitations.



Execution output

Running

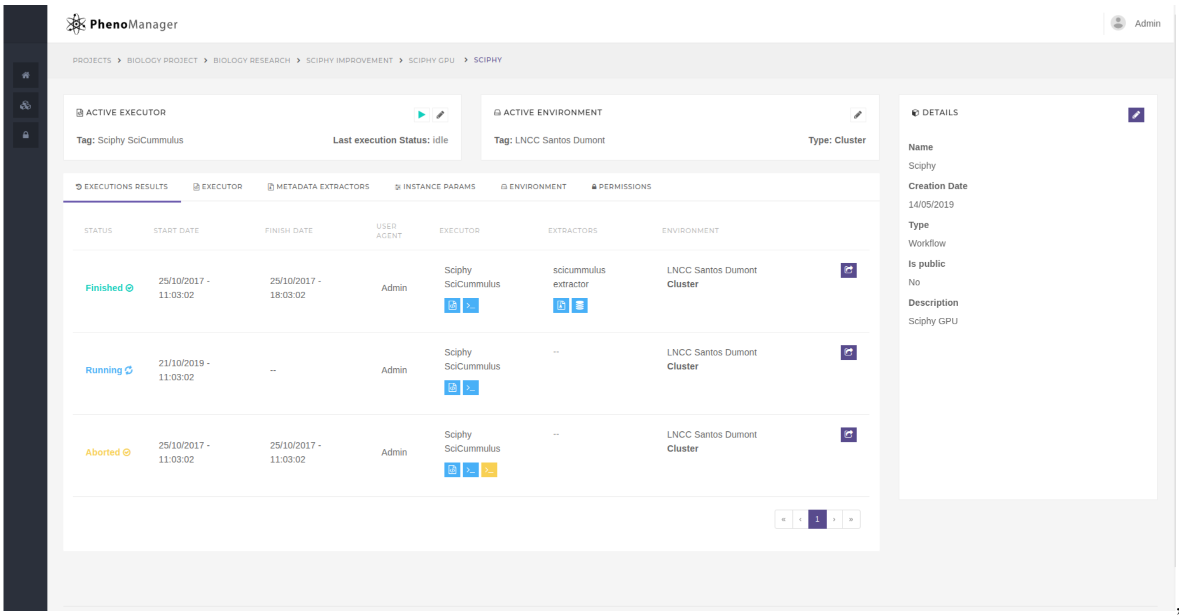
Start Date: 21/10/2019 - 11:03:02
Finish Date: *Still Running*

```
10/05/2019 - 23:25:50: Starting execution of modelExecutor [Sciphy]
10/05/2019 - 23:26:11: Setting up Cluster environment [LNCC Santos
Dumont]...
10/05/2019 - 23:27:42: Finish setting up Cluster environment [LNCC
Santos Dumont]
```

| FINISH DATE | USER AGENT | | | | |
|---|---|---|---|---|---|
| 25/10/2017 - 18:03:02 | Admi | | | | |
| -- | Admin | Sciphy SciCummulus | -- | LNCC Santos Dumont **Cluster** | |

**Name** Sciphy
**Creation Date** 14/05/2019
**Type** Workflow
**Is public** No
**Description** Sciphy GPU

Fig. 7: *Logs* of executions of a computational model displayed in real time.



ACTIVE EXECUTOR

**Tag:** Sciphy SciCummulus — **Last execution Status:** idle

ACTIVE ENVIRONMENT

**Tag:** LNCC Santos Dumont — **Type: Cluster**

DETAILS

**Name** Sciphy
**Creation Date** 14/05/2019
**Type** Workflow
**Is public** No
**Description** Sciphy GPU

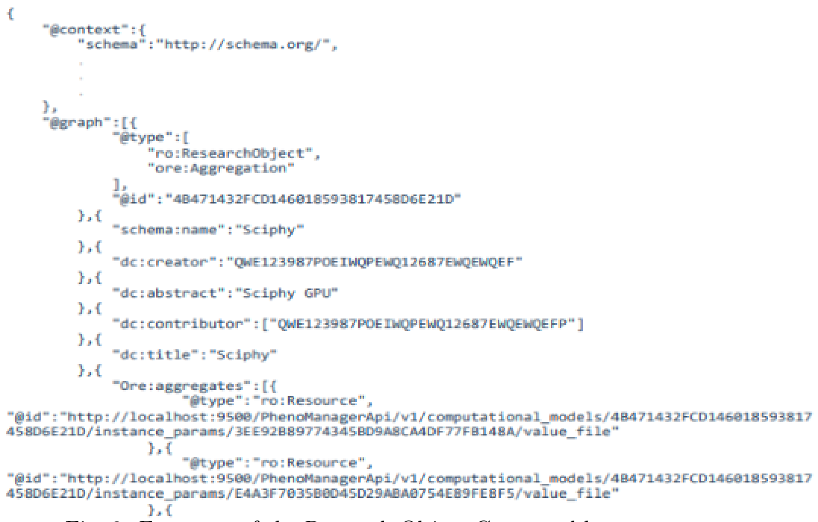
| STATUS | START DATE | FINISH DATE | USER AGENT | EXECUTOR | EXTRACTORS | ENVIRONMENT |
|---|---|---|---|---|---|---|
| Finished | 25/10/2017 - 11:03:02 | 25/10/2017 - 18:03:02 | Admin | Sciphy SciCummulus | scicummulus extractor | LNCC Santos Dumont **Cluster** |
| Running | 21/10/2019 - 11:03:02 | -- | Admin | Sciphy SciCummulus | -- | LNCC Santos Dumont **Cluster** |
| Aborted | 25/10/2017 - 11:03:02 | 25/10/2017 - 11:03:02 | Admin | Sciphy SciCummulus | -- | LNCC Santos Dumont **Cluster** |

Fig. 8: History of executions of a computational model.

## 4.1 Case Study: SciPhy Workflow

Sciphy [Ocaña et al. 2011] is a scientific workflow that is designed to generate phylogenetic trees with maximum likelihood. It was originally designed to work with amino acid sequences, but can be executed consuming other types of biological sequences. SciPhy is composed of four activities:

(1) Multiple Sequence Aligment (MSA): this activity builds individual alignments. It receives a multi-fasta file containing DNA and RNA sequences as input, producing as output an alignment (MSA);
(2) Sequence Conversion (implemented by ReadSeq): Converts the alignment into PHYLIP format;



```
{
    "@context":{
        "schema":"http://schema.org/",
        .
        .
        .
    },
    "@graph":[{
            "@type":[
                "ro:ResearchObject",
                "ore:Aggregation"
            ],
            "@id":"4B471432FCD146018593817458D6E21D"
        },{
            "schema:name":"Sciphy"
        },{
            "dc:creator":"QWE123987POEIWQPEWQ12687EWQEWQEF"
        },{
            "dc:abstract":"Sciphy GPU"
        },{
            "dc:contributor":["QWE123987POEIWQPEWQ12687EWQEWQEFP"]
        },{
            "dc:title":"Sciphy"
        },{
            "Ore:aggregates":[{
                    "@type":"ro:Resource",
"@id":"http://localhost:9500/PhenoManagerApi/v1/computational_models/4B471432FCD146018593817
458D6E21D/instance_params/3EE92B89774345BD9A8CA4DF77FB148A/value_file"
                },{
                    "@type":"ro:Resource",
"@id":"http://localhost:9500/PhenoManagerApi/v1/computational_models/4B471432FCD146018593817
458D6E21D/instance_params/E4A3F7035B0D45D29ABA0754E89FE8F5/value_file"
                },{
```

Fig. 9: Fragment of the Research Object Gerenated by `PhenoManager`

(3) Evolutionary Model Evaluation (implemented by ModelGenerator): In this activity each MSA is tested to find the best evolutionary model to use;
(4) Tree Generation (implemented by RaXml): In this activity both the evolutionary model and the MSA are used to generate phylogenetic trees;

Since the scientists do not know *a priori* which alignment method produces the best final result, they need to execute SchiPhy several times, once for each MSA method. In the configuration of SciPhy used in this article we consider the following MSA programs: MAFFT, Kalign, ClustalW, Muscle, ans ProbCons. Fig. 10 shows the conceptual view of SciPhy, where rectangles represent the conceptual activities and circles represent the programs that implement them. For more details about SciPhy, please refer to [Ocaña et al. 2011].

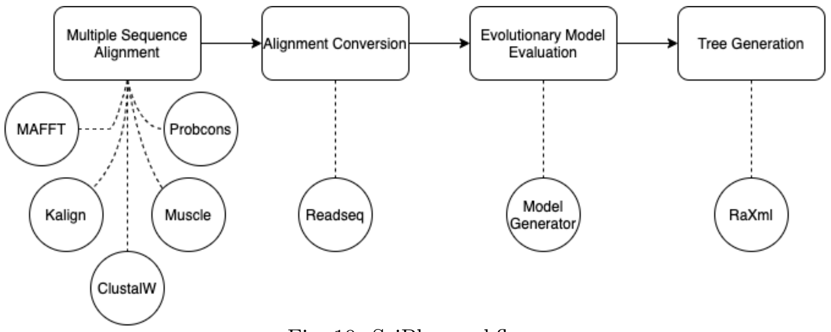


Fig. 10: SciPhy workflow

## 4.2 Overhead Analysis

The environment chosen to perform the overhead analysis was the Santos Dumont Supercomputer at LNCC. The Santos Dumont has an installed processing capacity of about 1.1 Petaflop/s, presenting a hybrid configuration of computational nodes. In addition, it has a total of 18,144 CPU cores distributed on 756 compute nodes (24 cores per node), where each contains mostly exclusively CPUs with multi-core architecture. Furthermore, it also has a differentiated node (fat node), with a high number of cores (240) and large-capacity shared memory architecture (6 Tb in a single address address space).

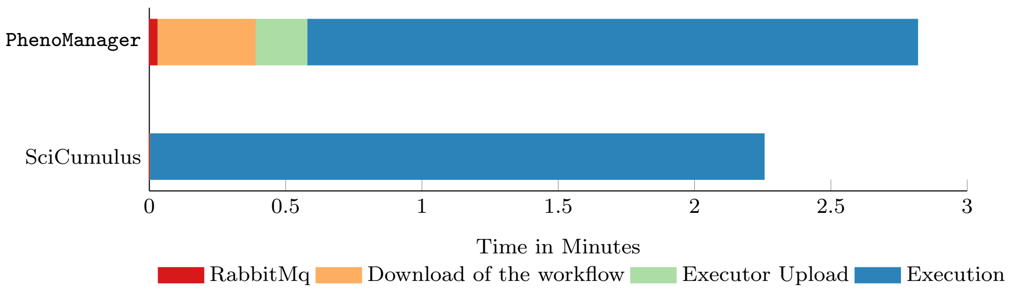


Fig. 11: Execution Times with Empty Queues

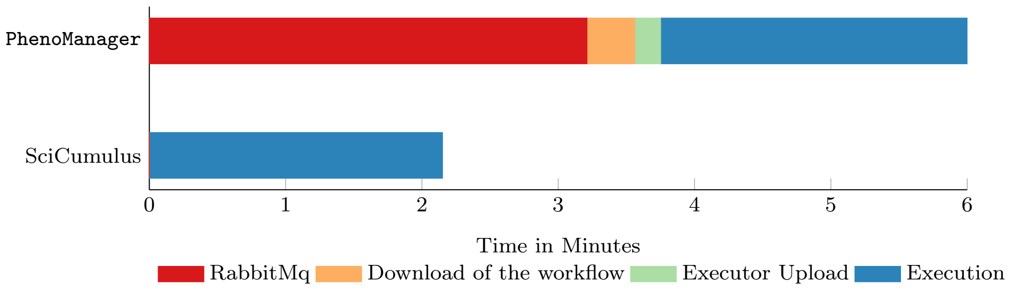


Fig. 12: Execution Times with Full Queues

We executed a small-scale version of SciPhy (consuming only 10 multi-fasta files) in two different scenarios. These 10 multi-fasta files of protein sequences contain 20 sequences (in average) and they were chosen since their processing is not time-consuming (other multi-fasta files may need several hours to be processed). These sequences were extracted from the UniProt database [DBL 2011] and we followed the inclusion criteria defined by [Ocaña et al. 2013]. The first scenario measures the average experiment execution time of 30 executions with empty RabbitMq queues and one consumer. The second scenario measures the average execution time of 30 execution times with a RabbitMq execution queue with one thousand parallel messages and ten consumers with ten threads each. We have executed the experiment 30 times since it is recommended as a minimum number of experiment repetitions to be statistically valid [Walpole et al. 2007].

In each of the scenarios, we compare the average execution time of SciPhy workflow using `PhenoManager` and only a WfMS (SciCumulus [de Oliveira et al. 2012]). In `PhenoManager` we measured the following times:

(1) Time from sending the message to RabbitMq until it consumes the message;

(2) Time to download the workflow specification and programs from Google Drive;

(3) Time to upload the executor to the configured environment (Santos Dumont at LNCC);

(4) Time to effectively execute the workflow;

In Fig. 11, we can see a comparison of the average execution times of a manual execution of SciPhy and using `PhenoManager` with empty queues. One can state that in this execution scenario, the execution time presents has a small and not very relevant variation between the two approaches. On the other hand, in Fig. 12, considering the second evaluation scenario, one can observe that there is a larger increase in the total execution time of the of the experiment. In this case, the workflow in `PhenoManager` remained in a SCHEDULED state while the messages were consumed in the queue.

### 4.3 Evaluation with Users

In order to qualitatively evaluate `PhenoManager`, we used the evaluation model called TAM (Technology Acceptance Model) [Davis 1989]. Inspired by the results in [de Souza et al. 2015], we used the TAM approach to capture the user perception of the proposed system (`PhenoManager`) regarding its *utility* and *ease-of-use* or *convenience*. This qualitative evaluation methodology extends the *Rational Choice Theory* (RCT) [Fishbein and Ajzen 1980], which indicates which user beliefs generate attitudes. Accordingly, the technology acceptance is directly influenced by the users' behavioral intention [Davis 1989], which in turn is motivated by two cognitive beliefs: *(i)* perceived *utility* (PU), and *(i)* perceived *ease-of-use* or *convenience* (PEOU). The TAM methodology suggests collecting a sufficient number of answers from questionnaires whose questions are linked with `PhenoManager` *utility* and *convenience*. Each question within the questionnaires follows a Likert scale [de Souza et al. 2015], and the user must select only one option between *(a)* Very Low, *(b)* Low, *(c)* Medium, *(d)* High, and *(e)* Very High. Five scholars (02 bioinformaticians, 03 computer science experts) confirmed their participation in this evaluation. Table I shows the questions used for the TAM evaluation of `PhenoManager`.

Table I: `PhenoManager` evaluation with TAM.

| **Question** | **Type** | **V. Low** | **Low** | **Medium** | **High** | **V. High** |
|---|---|---|---|---|---|---|
| Which is the applicability degree of `PhenoManager` in your data routines? | 1 - Utility | 0.00% | 0.00% | 16.66% | **50.00%** | 33.33% |
| Which would be the performance enhancement degree if you adopted `PhenoManager`? | 2 - Utility | 0.00% | 0.00% | **50.00%** | 33.33% | 16.66% |
| Which is the information quality level presented by `PhenoManager`? | 3 - Utility | 0.00% | 0.00% | 33.33% | **66.67%** | 0.00% |
| To which extent `PhenoManager` would enhance the quality of your work? | 4 - Utility | 0.00% | 0.00% | 0.00% | **83.33%** | 16.66% |
| Which is the ease degree of using `PhenoManager`? | 5 - Convenience | 0.00% | 0.00% | 0.00% | **50.00%** | **50.00%** |
| Which is the ease degree of learning how to use `PhenoManager`? | 6 - Convenience | 0.00% | 0.00% | 0.00% | **100.00%** | 0.00% |
| Which is the ease degree of remembering `PhenoManager` functionalities? | 7 - Convenience | 0.00% | 0.00% | 0.00% | **33.33%** | 66.66% |
| Which is the ease degree of identifying errors with `PhenoManager`? | 8 - Convenience | 0.00% | 0.00% | **83.33%** | 16.66% | 0.00% |

Table I presents the overall results of TAM questionnaires, in which most of surveyed experts indicate `PhenoManager` applies to their daily duties. Potential users also marked other features of `PhenoManager` as of "high" and "very high" usage. However, more than 80% of them also indicated identifying and fixing errors in `PhenoManager` is not simple. Notice that `PhenoManager` was first implemented as a proof of concept rather than a final product. Therefore, those indications are being used to enhance internal routines, such as error spotting and fixing.

## 5. RELATED WORK

To the best of the authors' knowledge, there is not another approach that manages a scientific project, a scientific experiment, the associated phenomena, and the hypothesis as `PhenoManager` does. This way, in this section, we present some approaches that present a level of intersection with `PhenoManager`, *i.e.*, this section discusses related work that aims at assisting in the scientific method, either by helping in the reproduction of experiments, in the management, modeling or execution of computational models.

Pardi and Russo [2019] propose a global storage Ecosystem to manage large-scale distributed datasets in the context of scientific projects. The idea is to create a project where scientists can aggregate multiple storage areas and datasets to simplify data retrieval. Differently from `PhenoManager`, the approach proposed by Pardi and Russo only considers data produced during experiments, but it does not manage the metadata associated with the experiment and its associated hypothesis. Kluge and Friedel [2018] propose Watchdog, a workflow system to automate complex analysis of large-scale experimental data. Although Watchdog allows for scientists to define their pipelines using pre-defined modules as well as their scripts, it considers workflows in an isolated way, decoupled from the concept of experiment and scientific project.

Dayibas *et al.* [2019] propose an approach to automate labor-intensive parts of simulation-based experiments. Dayibas *et al.* focus on the reusability of simulation experiments. Their work considers the concept of experiment and simulation to manage results. Lepperod *et al.* [2020] propose an approach named Expipe for data management in neuroscience experiments. The authors assume that produced and consumed datasets are too large to be stored locally and thus they demand analysis in distributed environments. Expipe stores and organizes experimental data and metadata captured during the execution of a pipeline. Marinho *et al.* [2017] propose an approach named ExpLine that allows for scientists to model their experiment in multiple levels of abstraction. It is based on the concept of Experiment Lines [de Oliveira et al. 2010]. Based on an abstract representation of the experiment, the scientist can derive concrete and executable workflows. Although it represents a step forward, ExpLine does not support managing hypothesis either phenomena.

Goble *et al.* [2018] highlight recent developments and approaches in the ResearchObject community. Research Object (RO) is a framework composed of several elements in a scientific project that can be packaged in a single unit of work. ROs contain programs, data, and metadata associated with the experiments and projects. It is already used in well-known platforms such as MyExperiment [Goble et al. 2010; Roure et al. 2010]. MyExperiment is a social platform for sharing Research Objects such as scientific workflows. Unlike existing workflow systems, its goal is only to share ROs and does not propose the execution and configuration of experiments or hypothesis evaluation. This tool could be integrated with `PhenoManager`, as long as the computational model is set as public. Wf4Ever Toolkit [Page et al. 2012] has the same purpose of sharing research objects. Like MyExperiment, this tool could be easily integrated with `PhenoManager`, taking advantage of the benefits of both approaches.

Rabix [Kaushik et al. 2017] is a project that works as a workflow modeler using Common Workflow Language (CWL)[3]. CWL is an open open standard for describing workflows and analysis tools in a way that makes them portable and and scalable in a variety of software and hardware environments, from workstations to cluster, cloud, and HPC environments. In terms of scientific project management as a whole, there are a number of initiatives that aim to support scientists in managing scientific projects. The LabGuru[4] aims to help the scientist in the management of his or her laboratory. Although it has functionalities for task distribution, LabGuru aims at a better allocation of allocation of resources (people and equipment) in the daily tasks. LabGuru does not concern with the execution of the experiment and the analysis of the data. Pinheiro *et al.* [2006] propose a research project management methodology, which is based on the use of project management techniques in research to obtain products. Although they do not propose any tool, the authors discuss their importance to the process.

Finally, ΥDB [Gonçalves and Porto 2015] is a probabilistic database system for representing numerical simulation output data as hypotheses of the phenomenon they reproduce. The term *hypotheses as data* is coined to refer to the output of simulations and associated uncertainty quantification. The system infers the uncertainty of variables based on the variability of the input values and that of the numerical model itself. ΥDB can be plugged to the output of `PhenoManager`, in case of numer-

[3]`https://www.commonwl.org/#`
[4]`https://www.labguru.com/`

ical simulation experiments, adding uncertainty to the hypothesis description. In addition, existing commercial tools such as Trello[5], Tasker[6], Redmine[7] and Jira[8] can be used for scientific project management. However, as these tools do not aim to serve this type of projects, there is no association with the concept of scientific experiment and there are no mechanisms to specify a workflow or execute it. Table II presents a comparative analysis of the related work.

Table II: Related work comparative analysis

| **Approach** | **License** | **Abstraction Level** | **Data Sharing** | **HPC Support** |
|---|---|---|---|---|
| [Pardi and Russo 2019] | Academic | Project and dataset | Distributed dataset | Yes |
| [Kluge and Friedel 2018] | Academic | Workflow | N/A | Yes |
| [Dayibas et al. 2019] | Academic | Experiment and simulation | No | N/A |
| [Lepperød et al. 2020] | Academic | Experiment and dataset | Distributed dataset | Yes |
| [Marinho et al. 2017] | Academic | Experiment line, abstract and concrete workflows | No | Partial |
| [Goble et al. 2018] | Academic | Concrete workflow | Research Object | N/A |
| [Goble et al. 2010; Roure et al. 2010] | Academic | Concrete workflow | Research Object | N/A |
| [Kaushik et al. 2017] | Academic | Abstract and concrete workflows | N/A | Yes |
| LabGuru | Commercial | Task | Shared dataset | N/A |
| Trello | Commercial | Project and task | Partial | No |
| Tasker | Commercial | Project and task | Partial | N/A |
| Jira | Commercial | Project and task | Partial | N/A |
| Redmine | Commercial | Project and task | Partial | N/A |
| ΥDB | Academic | Hypothesis | Simulation output | N/A |
| PhenoManager | Academic | Project, experiment, hypothesis, workflow and task | Research objects | Yes |

## 6. CONCLUSIONS

The volume and variety of data produced during the execution of a scientific project can be huge. All these data need to be structured and managed in a way that scientists can extract useful knowledge. These data may be associated with the execution of multiple workflows, evaluation of several hypotheses, *etc.* Managing these data can be arduous without an approach to support it.

This article presents the `PhenoManager`, a Scientific Hypothesis Management System that is capable of managing phenomena and hypotheses in conjunction with the execution of their associated experiments and computational simulations. These computational simulations may be implemented as a workflow, a script or invoking a third party program. The `PhenoManager` is based on a microservices architecture, which eases its extension and scalability. In this way, `PhenoManager` provides a valuable starting point for the integrated analysis of provenance data from multiple systems and programs.

In this article, we evaluated `PhenoManager` in a quantitative and qualitative way. In the quantitative evaluation, we measured the execution time overhead imposed by adopting the `PhenoManager`. Results showed that the introduced overhead is acceptable since analytical features are now available for scientists. In the qualitative evaluation, we analyzed the utility and ease-of-use of `PhenoManager` using TAM. Results show that most of the subjects mention that they could use `PhenoManager` in their daily duties. This is a contribution of this article in comparison with previous work [Ramos et al. 2019].

[5]`https://trello.com/pt-BR`
[6]`http://www.tasker.com.br/`
[7]`https://www.redmine.org/`
[8]`https://www.atlassian.com/br/software/jira`

As future work, we plan to provide additional analyses such as performance analyses of the execution of the simulation. In addition, we plan to implement an automatic hypothesis validation mechanism from the collected provenance data. The `PhenoManager` is an open source software and can be obtained at `https://github.com/UFFeScience/Phenomanager`.

## REFERENCES

Ongoing and future developments at the universal protein resource. *Nucleic Acids Res.* 39 (Database-Issue): 214–219, 2011.

Allen, A., Aragon, C. R., Becker, C., Carver, J. C., Chis, A., Combemale, B., Croucher, M., Crowston, K., Garijo, D., Gehani, A., Goble, C. A., Haines, R., Hirschfeld, R., Howison, J., Huff, K. D., Jay, C., Katz, D. S., Kirchner, C., Kuksenok, K., Lämmel, R., Nierstrasz, O., Turk, M. J., van Nieuwpoort, R., Vaughn, M., and Vinju, J. J. Engineering academic software (dagstuhl perspectives workshop 16252). *Dagstuhl Manifestos* 6 (1): 1–20, 2017.

Bass, L., Clements, P., and Kazman, R. *Software Architecture in Practice, Second Edition.* Addison-Wesley Professional, 2003.

Coutinho, F., Ogasawara, E. S., de Oliveira, D., Braganholo, V., Lima, A. A. B., Dávila, A. M. R., and Mattoso, M. Many task computing for orthologous genes identification in protozoan genomes using hydra. *Concurr. Comput. Pract. Exp.* 23 (17): 2326–2337, 2011.

Davis, F. D. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS quarterly*, 1989.

Dayibas, O., Oguztüzün, H., and Yilmaz, L. On the use of model-driven engineering principles for the management of simulation experiments. *J. Simulation* 13 (2): 83–95, 2019.

de Oliveira, D., Cunha, L., Tomaz, L., Pereira, V., and Mattoso, M. Using ontologies to support deep water oil exploration scientific workflows. In *2009 IEEE Congress on Services, Part I, SERVICES I 2009, Los Angeles, CA, USA, July 6-10, 2009.* IEEE Computer Society, pp. 364–367, 2009.

de Oliveira, D., Ocaña, K. A. C. S., Baião, F. A., and Mattoso, M. A provenance-based adaptive scheduling heuristic for parallel scientific workflows in clouds. *J. Grid Comput.* 10 (3): 521–552, 2012.

de Oliveira, D., Ocaña, K. A. C. S., Ogasawara, E. S., Dias, J., de A. R. Gonçalves, J. C., Baião, F. A., and Mattoso, M. Performance evaluation of parallel strategies in public clouds: A study with phylogenomic workflows. *Future Gener. Comput. Syst.* 29 (7): 1816–1825, 2013.

de Oliveira, D., Ogasawara, E. S., Chirigati, F. S., Silva, V., Murta, L. G. P., and Mattoso, M. Gexpline: A tool for supporting experiment composition. In *Provenance and Annotation of Data and Processes - Third International Provenance and Annotation Workshop, IPAW 2010, Troy, NY, USA, June 15-16, 2010. Revised Selected Papers*, D. L. McGuinness, J. Michaelis, and L. Moreau (Eds.). Lecture Notes in Computer Science, vol. 6378. Springer, pp. 251–259, 2010.

de Oliveira, D. C. M., Liu, J., and Pacitti, E. *Data-Intensive Workflow Management: For Clouds and Data-Intensive and Scalable Computing Environments.* Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2019.

de Souza, I. E., Oliveira, P. H. L., Bispo, E. L., Inocencio, A. C. G., and Parreira, P. A. TESE - an information system for management of experimental software engineering projects. In *Proceedings of the Brazilian Symposium on Information Systems*, S. W. M. Siqueira and S. T. Carvalho (Eds.). ACM, pp. 563–570, 2015.

Deelman, E., Gannon, D., Shields, M., and Taylor, I. Workflows and e-science: An overview of workflow system features and capabilities. *Future Generation Computer Systems* 25 (5): 528–540, 2009.

Deelman, E., Mehta, G., Singh, G., Su, M.-H., and Vahi, K. Pegasus: mapping large-scale workflows to distributed resources. In *Workflows for e-Science*. Springer, pp. 376–394, 2007.

Fishbein, M. and Ajzen, I. Understanding attitudes and predicting social behavior, 1980.

Freire, J., Koop, D., Santos, E., and Silva, C. T. Provenance for computational tasks: A survey. *Computing in Science & Engineering* 10 (3), 2008.

Gesing, S., Dahan, M., Zentner, M. G., Wilkins-Diehr, N., and Lawrence, K. A. The science gateways community institute: Collaborations and efforts on international scale. *Future Gener. Comput. Syst.* vol. 101, pp. 951–958, 2019.

Goble, C. A., Bhagat, J., Aleksejevs, S., Cruickshank, D., Michaelides, D. T., Newman, D. R., Borkum, M., Bechhofer, S., Roos, M., Li, P., and Roure, D. D. myexperiment: a repository and social network for the sharing of bioinformatics workflows. *Nucleic Acids Res.* 38 (Web-Server-Issue): 677–682, 2010.

Goble, C. A., Soiland-Reyes, S., and Bechhofer, S. Research object community update. In *Proceedings of Workshop on Research Objects (RO2018), Amsterdam, The Netherlands, October 29, 2018*, 2018.

Gonçalves, B. and Porto, F. Research lattices: towards a scientific hypothesis data model. In *Conference on Scientific and Statistical Database Management, SSDBM '13, Baltimore, MD, USA, July 29 - 31, 2013*, A. Szalay, T. Budavari, M. Balazinska, A. Meliou, and A. Sacan (Eds.). ACM, pp. 41:1–41:4, 2013.

Gonçalves, B. and Porto, F. Managing scientific hypotheses as data with support for predictive analytics. *Computing in Science Engineering* 17 (5): 35–43, 2015.

Hey, T., Gannon, D., and Pinkelman, J. The future of data-intensive science. *IEEE Computer* 45 (5): 81–82, 2012.

Holl, S., Garijo, D., Belhajjame, K., Zimmermann, O., Giovanni, R. D., Obst, M., and Goble, C. A. On specifying and sharing scientific workflow optimization results using research objects. In *WORKS 2013, Denver, CO, USA, November 17, 2013*. pp. 28–37, 2013.

Karau, H., Konwinski, A., Wendell, P., and Zaharia, M. *Learning spark: lightning-fast big data analysis.* " O'Reilly Media, Inc.", 2015.

Kaushik, G., Ivkovic, S., Simonovic, J., Tijanic, N., Davis-Dusenbery, B., and Kural, D. Rabix: An open-source workflow executor supporting recomputability and interoperability of workflow descriptions. In *Biocomputing 2017: Proceedings of the Pacific Symposium, Kohala Coast, Hawaii, USA, January 3-7, 2017*, R. B. Altman, A. K. Dunker, L. Hunter, M. D. Ritchie, and T. E. Klein (Eds.). pp. 154–165, 2017.

Kluge, M. and Friedel, C. C. Watchdog - a workflow management system for the distributed analysis of large-scale experimental data. *BMC Bioinform.* 19 (1): 97:1–97:13, 2018.

Lepperød, M. E., Dragly, S., Buccino, A. P., Mobarhan, M. H., Malthe-Sørenssen, A., Hafting, T., and Fyhn, M. Experimental pipeline (expipe): A lightweight data management platform to simplify the steps from experiment to data analysis. *Frontiers Neuroinformatics* vol. 14, pp. 30, 2020.

Marinho, A., de Oliveira, D., Ogasawara, E. S., Silva, V., Ocaña, K. A. C. S., Murta, L., Braganholo, V., and Mattoso, M. Deriving scientific workflows from algebraic experiment lines: A practical approach. *Future Gener. Comput. Syst.* vol. 68, pp. 111–127, 2017.

Mattoso, M., Werner, C., Travassos, G., Braganholo, V., and Murta, L. Gerenciando experimentos científicos em larga escala. *SBC-SEMISH* vol. 8, pp. 121–135, 2008.

Mattoso, M., Werner, C., Travassos, G. H., Braganholo, V., Ogasawara, E. S., de Oliveira, D., da Cruz, S. M. S., Martinho, W., and Murta, L. Towards supporting the life cycle of large scale scientific experiments. *Int. J. Bus. Process. Integr. Manag.* 5 (1): 79–92, 2010.

Newman, S. *Building microservices: designing fine-grained systems.* " O'Reilly Media, Inc.", 2015.

Ocaña, K. A., de Oliveira, D., Ogasawara, E., Dávila, A. M., Lima, A. A., and Mattoso, M. Sciphy: a cloud-based workflow for phylogenetic analysis of drug targets in protozoan genomes. In *BSB11*. Springer, pp. 66–70, 2011.

Ocaña, K. A. C. S., de Oliveira, D., Dias, J., Ogasawara, E. S., and Mattoso, M. Designing a parallel cloud based comparative genomics workflow to improve phylogenetic analyses. *Future Gener. Comput. Syst.* 29 (8): 2205–2219, 2013.

Ocaña, K. A. C. S., Galheigo, M., Osthoff, C., Jr., L. M. R. G., Porto, F., Gomes, A. T. A., de Oliveira, D., and de Vasconcelos, A. T. R. Bioinfoportal: A scientific gateway for integrating bioinformatics applications on the brazilian national high-performance computing network. *Future Gener. Comput. Syst.* vol. 107, pp. 192–214, 2020.

Ocaña, K. A., de Oliveira, D., Dias, J., Ogasawara, E., and Mattoso, M. Designing a parallel cloud based comparative genomics workflow to improve phylogenetic analyses. *Future Generation Computer Systems* 29 (8): 2205–2219, 2013. Including Special sections: Advanced Cloud Monitoring Systems The fourth IEEE International Conference on e-Science 2011 — e-Science Applications and Tools Cluster, Grid, and Cloud Computing.

Page, K. R., Palma, R., Holubowicz, P., Klyne, G., Soiland-Reyes, S., Cruickshank, D., González-Cabero, R., García-Cuesta, E., Roure, D. D., Zhao, J., and Gómez-Pérez, J. M. From workflows to research objects: An architecture for preserving the semantics of science. In *Proceedings of the Second International Workshop on Linked Science 2012 - Tackling Big Data, Boston, MA, USA, November 12, 2012*, T. Kauppinen, L. C. Pouchard, and C. Keßler (Eds.). CEUR Workshop Proceedings, vol. 951. CEUR-WS.org, 2012.

Pardi, S. and Russo, G. A big data approach for multi-experiment data management. *Int. J. Grid Util. Comput.* 10 (2): 159–167, 2019.

Pinheiro, A. A. A., Siani, A. A. C., Guilhermino, J. d. F. A., Henriques, M. d. G. A. M. d. O., Quental, C. M., and Pizarro, A. P. B. Metodologia para gerenciar projetos de pesquisa e desenvolvimento com foco em produtos: uma proposta. *Revista de AdministraÃS ÃPÃ* vol. 40, pp. 457 – 478, 06, 2006.

Porto, F., Costa, R. G., de Carvalho Moura, A. M., and Gonçalves, B. Modeling and implementing scientific hypothesis. *J. Database Manag.* 26 (2): 1–13, 2015.

Porto, F., Rittmeyer, J. N., Ogasawara, E. S., Krone-Martins, A., Valduriez, P., and Shasha, D. E. Point pattern search in big data. In *Proceedings of the 30th International Conference on Scientific and Statistical Database Management, SSDBM 2018, Bozen-Bolzano, Italy, July 09-11, 2018*, D. Sacharidis, J. Gamper, and M. H. Böhlen (Eds.). ACM, pp. 21:1–21:12, 2018.

Ramos, L., Ocaña, K., Oliveira, D., Porto, F., and de Oliveira, D. Phenomanager: um sistema de gerência de hipóteses de fenômenos científicos. In *Anais estendidos do XXXIV Simpósio Brasileiro de Bancos de Dados*. SBC, Porto Alegre, RS, Brasil, 2019.

Ramos, L. S., Ocaña, K. A., and de Oliveira, D. Um sistema de informação para gerência de projetos científicos baseados em simulações computacionais. In *Anais do XII Simpósio Brasileiro de Sistemas de Informação*. SBC, pp. 216–223, 2016.

Roure, D. D., Goble, C. A., Aleksejevs, S., Bechhofer, S., Bhagat, J., Cruickshank, D., Fisher, P., Hull, D., Michaelides, D. T., Newman, D. R., Procter, R., Lin, Y., and Poschen, M. Towards open science: the myexperiment approach. *Concurr. Comput. Pract. Exp.* 22 (17): 2335–2353, 2010.

Travassos, G. H. and Barros, M. O. Contributions of in virtuo and in silico experiments for the future of empirical studies in software engineering. In *2nd Workshop on Empirical Software Engineering the Future of Empirical Studies in Software Engineering*. pp. 117–130, 2003.

Vaquero, L. M., Rodero-Merino, L., Caceres, J., and Lindner, M. A break in the clouds: Towards a cloud definition. *SIGCOMM Comput. Commun. Rev.* 39 (1): 50–55, Dec., 2008.

Walpole, R. E., Myers, R. H., Myers, S. L., and Ye, K. *Probability & statistics for engineers and scientists*. Pearson Education, Upper Saddle River, 2007.

Zaharia, M., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M. J., Ghodsi, A., Gonzalez, J., Shenker, S., and Stoica, I. Apache spark: a unified engine for big data processing. *Commun. ACM* 59 (11): 56–65, 2016.