

ABSTRACT

Today scientists are working in e-Science environments and carry out *in silico* experiments. A powerful approach, with proven capabilities to facilitate the design process of computational experiments is based on Scientific Workflows, which are receiving considerable interest in recent years. This thesis thoroughly reviews the Scientific Workflows Management Systems field and investigates in detail popular open source workflow systems from a scientific applicability perspective. Moreover, a complex computational experiment from the life sciences field is implemented using current workflow technology in order to better assess their strengths and weaknesses. Emphasis is placed on features which make these systems attractive for scientific use, e.g. user friendliness, use of distributed resources, reusability, provenance, collaboration, data integration, etc. Our conclusions indicate that although Scientific Workflow Management Systems have open issues, discussed in detail in the context of this thesis, their strong momentum clearly suggests that is only a matter of time before they are adopted by even more scientific fields.

**OPEN SOURCE WORKFLOW SYSTEMS FOR THE
DEVELOPMENT OF COMPLEX COMPUTATIONAL
EXPERIMENTS**

Kleo G. Achilleos

A Thesis

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

at the

University of Cyprus

Recommended for Acceptance

by the Department of Computer Science

June, 2012

APPROVAL PAGE

Master of Science Thesis

OPEN SOURCE WORKFLOW SYSTEMS FOR THE DEVELOPMENT OF COMPLEX COMPUTATIONAL EXPERIMENTS

Presented by

Kleo G. Achilleos

Research Supervisor

Research Supervisor's Name

Committee Member

Committee Member's Name

Committee Member

Committee Member's Name

University of Cyprus

June, 2012

ACKNOWLEDGMENTS

First of all I would like to thank my supervisor, Professor Constantinos S. Pattichis for his support and guidance. Secondly I would like to acknowledge the members of the GRANATUM team for their contribution to this work, and especially Mr. Christos Kannas for his valuable help. I would also like to thank Professor Vasilis J. Promponas for his thorough review and detailed feedback on the thesis.

Most importantly I would like to express my sincere gratitude to Dr. Christos A. Nicolaou for his continuous guidance, encouragement and assistance that enabled and facilitated the completion of this thesis.

Finally, I would like to thank my husband, my two daughters, my parents, my siblings and my entire family to whom I am greatly indebted for all their help and support in preparing this thesis and for the duration of my MSc degree studies.

TABLE OF CONTENTS

Chapter 1: Introduction.....	1
1.1 Aims of this study.....	5
1.2 Guide to Thesis contents	5
Chapter 2: Scientific Workflow Technology Review	6
2.1 Scientific workflow paradigm	6
2.2 Types and subcategories	9
2.3 Scientific workflow management systems	11
2.4 Scientific workflow life cycle	12
2.5 SWMS's architecture.....	14
2.6 SWMS Review	15
2.7 Scientific workflow collaboration	20
2.8 Current projects	21
Chapter 3: Current Open Source Scientific Workflow Systems	24
3.1 KNIME	24
3.2 Taverna	33
3.3 Galaxy.....	41
3.4 Summary	55
Chapter 4: Problem Statement	57
4.1 Chemo informatics Background.....	58
4.2 Problem Description	60

4.3 GRANATUM.....	62
Chapter 5: Experimental Design and Implementation	64
5.1 Experimental design	64
5.1.1 Data	65
5.1.2 Diagram	66
5.2 KNIME implementation	67
5.3 Galaxy implementation	80
Chapter 6: Results/Discussion	89
Chapter 7: Conclusions/Future Work.....	98
Bibliography.....	102

LIST OF TABLES

Table 2.1 – List of popular Workflow applications	17
Table 3.1 – Port types for KNIME nodes	28
Table 3.2 – Node’s Status in KNIME	29
Table 3.3 – Comparing SWMSs	55
Table 4.1 – Details of each process in the VS workflow	62
Table 5.1 – VS Datasets details	65
Table 5.2 – VS workflow process functionality	66
Table 5.3 – Short descriptions of secondary applications in the KNIME implementation	68
Table 5.4 – Detailed description of nodes in the prediction model construction workflow	71
Table 5.5 – Confusion matrix	72
Table 5.6 – Learning Methods used	74
Table 5.7 – Results of testing prediction models depicting F–measure	75
Table 5.8 – Detailed Description of each node in the workflow of Figure 5.5.....	79
Table 5.9 – Short descriptions of additional applications for the Galaxy workflow 81 implementation	81
Table 5.10 – Detailed Description of each Tool of Figure 5.11	82
Table 5.11 – New Definition types in Galaxy	83
Table 6.1 – Results table	90
Table 6.2 – Comparing KNIME and Galaxy implementations	94

LIST OF FIGURES

Figure 1.1 – Application domains of SWMS	4
Figure 1.2 – Applying future Healthcare workflow technology on the cloud	4
Figure 2.1 – Sample workflow in KNIME platform.....	7
Figure 2.2 – Sample workflow in Taverna workbench	7
Figure 2.3.1 – Remote service call workflow in Taverna	8
Figure 2.3.2 – Remote service details as displayed by the Taverna workbench.....	8
Figure 2.4 – Scientific workflow life cycle as proposed by [4]	13
Figure 2.5 – SWMS high level components interaction.....	14
Figure 2.6 – myExperiment’s interface.....	21
Figure 3.1 – KNIME workspace from www.knime.org.....	26
Figure 3.2 – KNIME workspace with project details.....	27
Figure 3.3 – Sample workflow in KNIME	29
Figure 3.4 – KNIME’s Example Flow Server workspace.....	30
Figure 3.5 – Table view of a k-means prediction algorithm run on a sample file.....	31
Figure 3.6 – Scatter Plot of the same file	31
Figure 3.7 – Nodes data input and output.....	32
Figure 3.8 – Control flow nodes	32
Figure 3.9 – Taverna workbench	35
Figure 3.10 – myExperiment perspective.....	35
Figure 3.11 – Sample Workflow in Taverna workbench	36
Figure 3.12 – Processor Details in Taverna.....	37
Figure 3.13 – Validation report in Taverna	38
Figure 3.14 – Results perspective in Taverna.....	39
Figure 3.15 – A simplified view of Taverna processors activities	40

Figure 3.16 – Galaxy’s Analyze Data Interface.....	43
Figure 3.17 – UCSC Table Browser	44
Figure 3.18 – Options Dialog in Shared Data and User menu.....	44
Figure 3.19 – A shared Galaxy page.....	45
Figure 3.20 – A sample workflow	46
Figure 3.21 – Options in Workflow panel.....	46
Figure 3.22 – Galaxy’s Workflow editor space.....	47
Figure 3.23 – Galaxy’s email notification.....	47
Figure 3.24 – Invoking execution in Galaxy	48
Figure 3.25 – Running a workflow in Galaxy	49
Figure 3.26 – History records in Galaxy	49
Figure 3.27 – Saved Histories in Galaxy.....	50
Figure 3.28 – Text view of dataset provided by Galaxy	50
Figure 3.29 – Visualization by the online Genome browser Ensembl	51
Figure 3.30 – Visualization on chrX from Trackster, Galaxy’s visualization environment.....	51
Figure 3.31 – A simplified view of the objects in Galaxy	52
Figure 3.32 – Galaxy objects in action	53
Figure 3.33 – Sample Taverna workflows embedded as tools in the Galaxy server instance at http://galaxy.nbic.nl/galaxy/	54
Figure 4.1 – Properties of a candidate breast cancer chemo preventive compound.....	61
Figure 4.2 – The chemoprevention virtual screening workflow to be implemented	61
Figure 5.1 – Experimental Design Diagram	67
Figure 5.2 – Dataset A containing molecule structures and their toxicity values as displayed by KNIME.....	69
Figure 5.3 – The prediction model construction workflow	70
Figure 5.4 – Models build and saved by KNIME.....	71
Figure 5.5 – The VS workflow as displayed in KNIME.....	77
Figure 5.6 – Input File as displayed by KNIME.....	78

Figure 5.7 – Output File as displayed by KNIME	78
Figure 5.8 – Experiment’s Report in a spreadsheet sorted as pleased.....	80
Figure 5.9 – New tools in the Galaxy’s Tool panel	83
Figure 5.10 – Data types and restrictions enforced among the tools defined	84
Figure 5.11 – The workflow as created in GALAXY	85
Figure 5.12 – Input parameter file as displayed in Galaxy	86
Figure 5.13 – Results of the experiment in an Excel worksheet	86
Figure 5.14 – Output File as displayed in Galaxy.....	87
Figure 5.15 – Published Chemoprevention Workflow in the Galaxy server instance at UCY.	87
Figure 5.16 – Published Chemoprevention Workflows in myExperiment.....	88
Figure 6.1 – Oral Drug Likeness Filter – KNIME – CDK	91
Figure 6.2 – Oral Drug like filter – Galaxy – RDKit	91
Figure 6.3 – Toxicity Prediction on all compounds – KNIME SVM	91
Figure 6.4 – Toxicity Prediction on all compounds – Galaxy SVM.....	91
Figure 6.5 – Overlap results of Oral Drug Like Filter for KNIME-CDK and Galaxy-RDKit implementations.....	91
Figure 6.6 – Overlap results of Toxicity Model prediction for KNIME-Weka SMO and Galaxy-Scikit-learn Nu-SVCimplementations	91
Figure 6.7 – Binding Affinity score on all compounds	92
Figure 6.8 – Toxicity Prediction on Drug Like compounds	92
Figure 6.9 – Binding Affinity score of Drug like Inactive compounds.....	91
Figure 6.10 – Results summary on KNIME implementation	91
Figure 6.11 – 4135 Topics in Galaxy Development list and 188 only in April 2012 from http://dev.list.galaxyproject.org	95
Figure 7.1 – SW and its constituent parts.....	98

TABLE OF ACRONYMS

CSV	Character Separated File
DAG	Directed Acyclic Graph
DPML	Discovery Markup Language
HPC	High Performance Computing
NCI	National Cancer Institute, USA
NEPTUNE	North East Pacific Time-integrated Undersea Networked Experiments
Pan-STARRS	Panoramic Survey Telescope and Rapid Response System
PP	Pipeline Pilot
PMML	Predictive Model Markup Language
SW	Scientific Workflows
SWMS	Scientific Workflow Management System
UCY	University of Cyprus
VS	Virtual Screening
WFRM	Workflow Reference Model
WSDL	Web Services Description Language
WSFL	Web Service Flow Language
XML	Extensible Markup Language
XOML	Extensible Orchestration Markup Language

Chapter 1

Introduction

“e-Science is all about furthering technology in order to advance the scientific discipline”¹

Today scientists are working in e-Science environments and carry out *in silico* experiments. In other words scientists use environments that support global collaboration, involve multidisciplinary science and utilize modern technology infrastructure² to carry out their experiments *in silico*.³ A powerful approach, with proven capabilities to facilitate the design process of computational experiments is based on Scientific Workflows (SW). This approach enables scientists to plug together problem solving computational components [1] and implement complex in-silico experiments such as the analysis of datasets of multi-Terabyte magnitude that arise from sensors or computer simulations, and, the design and execution of complicated algorithms requiring numerous computationally intensive steps. Scientific workflow management systems (SWMS) can potentially accelerate scientific discovery by incorporating data management, analysis, simulation, and visualization tools. They provide an interactive visual interface that facilitates the design, execution and management of workflows. Moreover, scientific workflow management systems enable remote access as well as data and services sharing, making possible collaborations among geographically distributed researchers.

¹ <http://www.escience-grid.org.uk/>

² <http://www.lesc.ic.ac.uk/admin/escience.html>

³ *In silico*, i.e. via computer simulations; not *in-vivo* (in living organisms) or *in-vitro* (in glass tubes).

Traditionally, many scientists have been using batch files, shell scripts, and programs written in general-purpose scripting languages (e.g., Perl, Python) to automate their tool-integration tasks [2]. This approach provides high flexibility, and is therefore appealing, to expert users but makes it difficult for the average user to implement scientific tasks requiring the integration of multiple computational components and data resources. Scientific workflows provide a promising alternative to all scientific users facing the above problem because of several inherent advantages [3]. Two main advantages of the SW approach are visual representation of the task flow and visual channeling of data as opposed to lines of code directing the flow in the case of scripts. Provenance⁴ information, which is very important for the reproducibility of the experiments as well as for backtracking and resolution of errors, is an additional characteristic of workflows not present in scripting tools. Reusability and transparency is easily achieved by the reuse of a workflow or the use of a workflow inside a workflow. Finally complex implementation details such as parallelism, pipelining and High Performance Computing (HPC) are handled transparently by SWMS systems in order to achieve maximum efficiency for execution time [22].

Fundamentally, a scientific workflow is a tool that automates the execution of an experiment. As such it can offer multiple benefits for all the phases of an experiment's lifecycle. During the composition phase, a repository of tried and tested workflows is available to the scientists to choose from. During the execution phase, as experimenting is by definition a repeatable process, workflows can relieve the scientists of repetitive tasks but at the same time keep track of all the intermediary steps and data. These traces can be used at a later stage to enable the reproducibility of the experiment. Provenance information[25] is also useful during the analysis phase to assess the evolution of the research effort, trace the origin of an error or go back to a previous stage and change the direction of investigation. Visualization tools are provided for this phase as well for assisting in the evaluation of the results [25].

⁴ In the scientific workflow research community, the information that describes the details of data processing history is referred to as "provenance" (also "lineage" or "pedigree") (Simmhan et al., 2005).

Moreover, being a promising tool for End-to-End scientific data management, scientific workflows enable scientists to cope with big data as petabytes of data are produced either by sensors or by simulations executing scientific algorithms. Grid technologies allow workflows to implement parallel executions enabling large scale data processing. In this case, workflows are used as a parallel programming model for data-parallel applications. Web services allow ease of access to local and distributed data sources as well as data aggregation from highly heterogeneous environments. Even HPC technology can be made available to scientists who may have limited or no computing resources. Finally, collaboration between scientists is encouraged and achieved both within and across disciplines. Implemented similarly to the trend of social networks, scientists share workflows and their corresponding services. All of the above can optimize the implementation of experiments in a transparent way for the domain scientist.

Currently over 50 different representatives of scientific workflow management systems exist [4]. The most popular in scientific literature being Taverna [5],[6],[7], Triana [8], Kepler [9], Pegasus[10] and KNIME [11],[12] which are open source software and Pipeline Pilot [13], InforSense KDE [14] and Microsoft Trident [15] which are commercial products. On the other hand, Galaxy [16], is a more recent Web based SWMS dedicated to biomedical research that is increasingly gaining popularity.

As in the case of many other tools, SWMS quickly found application in a great number of diverse scientific domains, although they were originally developed with a specialized domain application in mind. Figure 1.1 illustrates some of the main application domains of SWMS. This domain independence is mainly owed to the abstraction that characterizes the workflow paradigm.



Figure 1.1 – Application domains of SWMS

Recent advances do not yet match the expectations of scientists, as noted in [2]. However they are a step towards a future where we can imagine a doctor preparing a checkup workflow of a patient containing complicated DNA analysis, statistical prediction models, image analysis algorithms, inference rules engines and database search all from his tablet pc only to be executed somewhere in the cloud. Or a computer programmer selecting from a pool of cross platform resources created by other computer programmers that read a simple file, scan a database, search the web, send email, calculate formulas, build statistical models, implement complicated algorithms, or executed built-in workflows to create a work-flowed application that can enrich the resources of the initial pool.

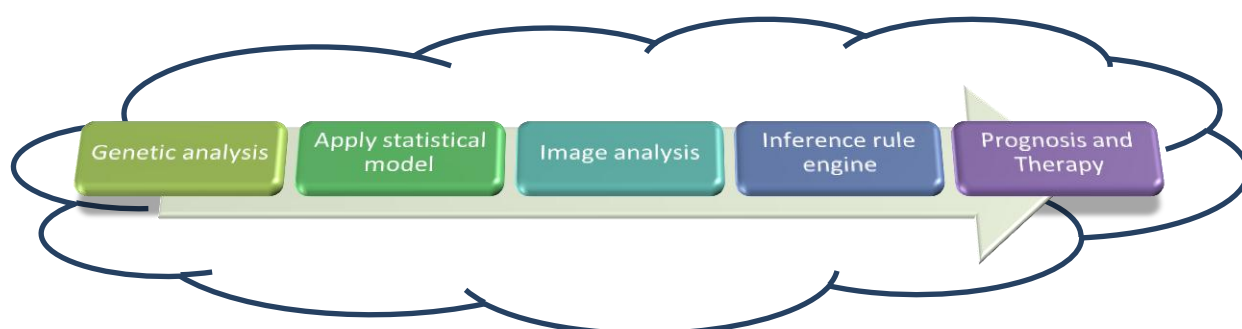


Figure 1.2 – Applying future Healthcare workflow technology on the cloud

1.1 Aims of this study

A main goal of this thesis is to thoroughly review the SWMS field and investigate in detail popular open source workflow systems from a scientific applicability perspective. To achieve this goal a general overview of the field is prepared as well as a more detailed review of representative systems. A second goal is to design a scientific workflow addressing the needs of complex *in silico* experiments from the life sciences field, specifically, the chemoprevention domain. The designed workflow will also be implemented using two of the most promising open source SWMS available. This task involves the preparation of appropriate nodes/tools for each of the SWMS, the implementation and execution of the workflows and the analysis and presentation of the results obtained. The third main goal of the work presented in this thesis is to assess progress in the SWMS field. We will concentrate on what workflow technology offers currently, how we can benefit from it, how it can be improved and what difficulties arise when in use. Further assessment will take place through the evaluation and discussion of the experiences and results obtained from the workflows developed with respect to the features which make SWMS's attractive, e.g. user friendliness, use of distributed resources, reusability, provenance, etc.

1.2 Guide to Thesis contents

The rest of thesis paper is structured as follows: Chapter 2 focuses on the scientific workflow paradigm, its main categories and introduces some of the main SWMS representatives in the scientific literature as well as example use cases. Chapter 3 presents in detail three open source SWMS and discusses their components and functionality. In Chapter 4 a test case workflow is developed which the next chapter, Chapter 5, implements and applies as a test case on two separate platforms. In Chapter 6 the results are discussed while Chapter 7 sums up this thesis by presenting the conclusions and future work.

Chapter 2

Scientific Workflow Technology Review

<i>2.1 Scientific workflow paradigm</i>	<i>2.5 SWMS's architecture</i>
<i>2.2 Types and subcategories</i>	<i>2.6 SWMS Review</i>
<i>2.3 Scientific workflow management systems</i>	<i>2.7 Scientific workflow collaboration</i>
<i>2.4 Scientific workflow life cycle</i>	<i>2.8 Current projects</i>

Scientific workflows help tackle the problem of excessive complexity of in silico experimentation by helping scientists model what an experiment is set to achieve, while abstracting out how it will be executed⁵

2.1 Scientific workflow paradigm

A workflow (WF) is a general, widely used term used to describe the actions that need to be taken in order to complete a complex task. An abstract scientific workflow is represented as a directed graph where each node represents a step⁶ implemented by a software component. This component can be either the execution of a local program or a remote web service (e.g. a query to a database). The edges of the graph represent either data flow or execution dependencies between nodes [55]. The links coordinate the inputs and outputs of the individual steps, forming the data flow. Control flow links occur when two tasks have no data dependencies and therefore the order must be explicitly defined.

In Figure 2.1 a sample workflow, designed using the KNIME platform, is depicted. The sample workflow reads a file, interchanges rows with columns, executes a local script and

⁵ <http://www.taverna.org.uk/>

⁶ **Synonyms:** activities, components, processors

saves the results in a file in the .csv⁷ format. Each step is represented by a node which is clearly named. The links symbolize the flow of the data from one node to the next. The order of execution is determined by the data dependencies.

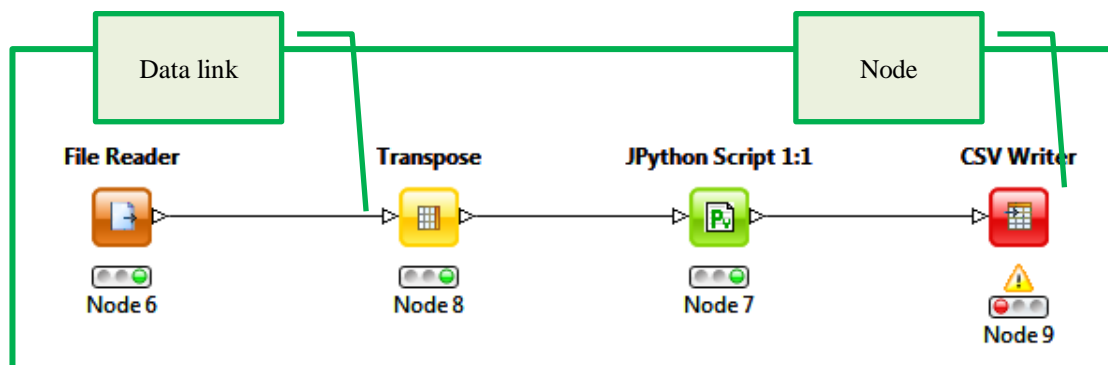


Figure 2.1 – Sample workflow in KNIME platform

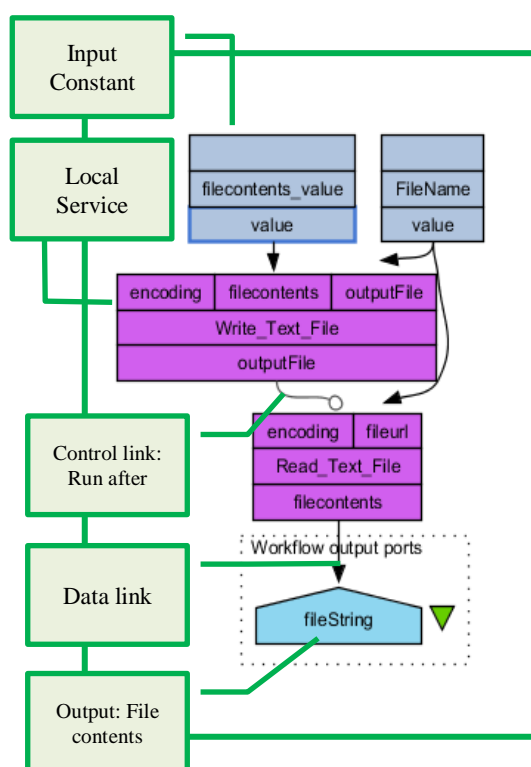


Figure 2.2 – Sample workflow in Taverna workbench

A second example from the Taverna workbench is shown in Figure 2.2. The workflow writes to a text file a value and then it reads the value from the file and presents it as output. The nodes in purple color are the tasks or services to be executed. In this case both services are local programs. The pointed arrows represent the data flow while the rounded ones are control flow links necessary to define the order so that writing the file proceeds reading it.

⁷ csv is a comma separated value plain text file format

A third more useful example from the chemistry domain is given in Figure 2.3.1. This WF has also been designed in the Taverna environment. The workflow converts a file containing chemical substances from one format to another. Three inputs must be specified: input-format, output-format, and the file containing the chemical substance. Next, the format conversion is taking place and the result is passed to the output port. The conversion task this time is not a local program but a network service. More specifically it is a WSDL service. WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information⁸. In Figure 2.3.2. the details of this service are given as displayed by the Taverna environment.

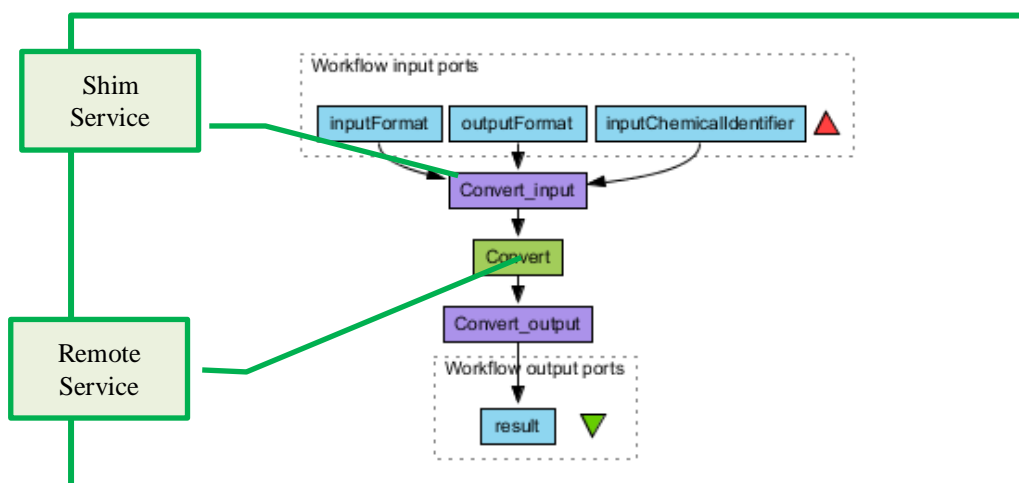


Figure 2.3.1 – Remote service call workflow in Taverna

WSDL-based service	
WSDL	http://www.chemspider.com/OpenBabel.asmx?WSDL
Operation	Convert
Secure	false
Inputs	
parameters	Depth:0 {http://www.chemspider.com/}Convert
Outputs	
parameters	Depth:0 {http://www.chemspider.com/}ConvertResponse
attachmentList	Depth:1

Figure 2.3.2 – Remote service details as displayed by the Taverna workbench

⁸ <http://www.w3.org>

Usually, real life scientific workflows are more complex with calls to more services, usage of shim services to convert between inputs and additional parameters for sequence of execution, looping and error handling.

Abstract workflows are sometimes described using special languages or XML schemas e.g. BPEL[19][21] in the Trident system, DAG[20] in Pegasus, t2flow[52] in Taverna or even simple database values as in Galaxy [16],[55]. Once the abstract workflows are translated into machine readable language they can be fed into workflow execution engines.

2.2 Types and subcategories

Workflow technology is not new. It has long been adopted by the business community. Business workflow management and business process modeling are mature research areas, whose roots go far back to the early days of office automation systems [22]. However, the term “Scientific Workflow” became popular after the year 2000, as the existing technology could not support the special characteristics of scientific processes which are data and computationally intensive, highly repetitive and reproducible. In the Workflow Reference Model (WFRM), published in 1995 by the Workflow Management Coalition industry consortium [23], there is a clear definition for the term workflow.

“A workflow is defined as the computerized facilitation of automation of a business process, in whole or part.”

In the case of scientific workflow however, experts in the field like Ludascher et al. [22] point out that “there seems to be no single set of characteristic features that would uniquely define what a scientific workflow is and isn’t.”

Flow control can be considered the most important classification characteristic of scientific workflows. A workflow is either data-flow or control-flow oriented. In control-driven workflows the connections between the tasks represent a transfer of control from one task to the next one. In data-driven workflows connections represent the flow of data from one task to the next one. The workflow representation is centered on data products. As mentioned in [1]

most of the current scientific workflows are data-flow oriented as opposed to their predecessors and business workflows which are control-flow. According to [24], the reason is that data-flow modeling is the natural way of composing scientific workflows, because they often comprise numerous data transformation steps applying massive parallelism.

Another important distinguishing feature of workflows is pipeline parallel processing. A pipeline consists of a collection of steps. Parallelism is achieved by executing these steps simultaneously on different input data sets. The tasks are executed in separate threads, processing input immediately and not waiting for the previous task to complete. The drawback is that pipelined workflows are harder to restart in the case of unforeseen events as the current state of the executed workflow is not as easy to describe and restore [25].

A popular categorization is based on the distinction between high level scientific oriented workflows and lower-level engineering resource oriented (or “plumbing”) workflows [1]. The first are an implementation of an experimental protocol or a data analysis method where each task corresponds to the high level tasks of the scientific method. The latter are concerned mostly with the “plumbing tasks” such as data movement and replication and job management.

A Workflow Model (also called workflow specification) defines a workflow including its task definition and structure definition. There are two types of workflow models, namely abstract and concrete [28]. The abstract model defines a workflow in an abstract form without referring to any resources for task execution. On the contrary in the concrete model the workflow tasks are bound to the designated resources. The user creates the abstract workflow in the workflow modeler component. Mapping the resources is done transparently by the enactment engine to create a concrete executable workflow.

Scientific workflows can also be differentiated based on the design focus. In the initial discovery stages of a scientific method, a non-mature workflow is constantly changing while

the designer is trying out different approaches and solutions. The design considerations are ease of change and reusability. Later on, as the workflow becomes mature, it obtains a steady form and can then be used as a production workflow executed on a regular basis. At this point the design considerations shift to speed and efficiency.

2.3 Scientific workflow management systems

In theory a SWMS is a combination of a workflow modeling component using an abstract language and a workflow enacting component empowered by an execution engine. In practice a SWMS enables a user to create and then monitor the execution of a workflow by providing the necessary infrastructure. The modeling component enables the user to design, reuse and store workflow models while the enacting component invokes, executes and monitors workflow instances [4] deploying them either on a local desktop computer, a web server, or a distributed computing environment. Embedded in the workflow design, is the order of the tasks to be executed. The coordination process of this execution is known as orchestration. The execution engine adds the transparency required to allow the domain scientist to model a solution without any concerns of how the solution will be carried through.

This architecture is applied in the Trident SWMS [26]. This Microsoft system allows for independent components for workflow modeling and for execution. Firstly the scientist creates the workflow in an independent workflow composer. Then the workflow is executed in Trident. This is known as centralized execution architecture. Other systems follow a less strict decentralized architecture. For example, in Taverna 2 [5], each processor independently starts its own execution as soon as the input data are available. This allows for inter-processor parallelism as the tasks are executed in separate threads. The need for coordination however exists, so the system offers a façade pattern that relays messages to and from the central monitor. As SWMS are software environments created specifically for workflows, they encompass a number of functionalities for their management including workflow design, re-engineering, allocation of resources, task scheduling, data movement, data formats,

optimizations, execution, monitoring, fault management, analysis, provenance data, storage, collaboration, reuse. Moreover, SWMS is typically run over middleware that provides infrastructure for accessing the applications or resources consumed by the workflow, and facilities like security and access control [4].

2.4 Scientific workflow life cycle

The main design goal of SWMS is to support the workflow lifecycle. Detailed analysis of how each step of the lifecycle can be supported provides an improved understanding of the functionalities that any SWMS must accommodate. The lifecycle of a scientific workflow begins with the Design phase where a new workflow is created either from scratch or from existing workflows. During the following Planning phase the workflow is validated and optimized to user requirements. This phase also includes resource allocation and task scheduling if required. The Execution phase involves invoking and monitoring the workflow, retrieving the data, error handling and keeping measurements. The results of the execution are visualized and tagged in the Analysis phase. Finally, in the Storage phase the workflow is stored along with its provenance data and enabled for sharing [4]. Slightly different scientific workflow life cycles were proposed by experts in the field in [4], [22], [26], [25], [27]. In Figure 2.4 the scientific workflow life cycle is given as presented in [4].

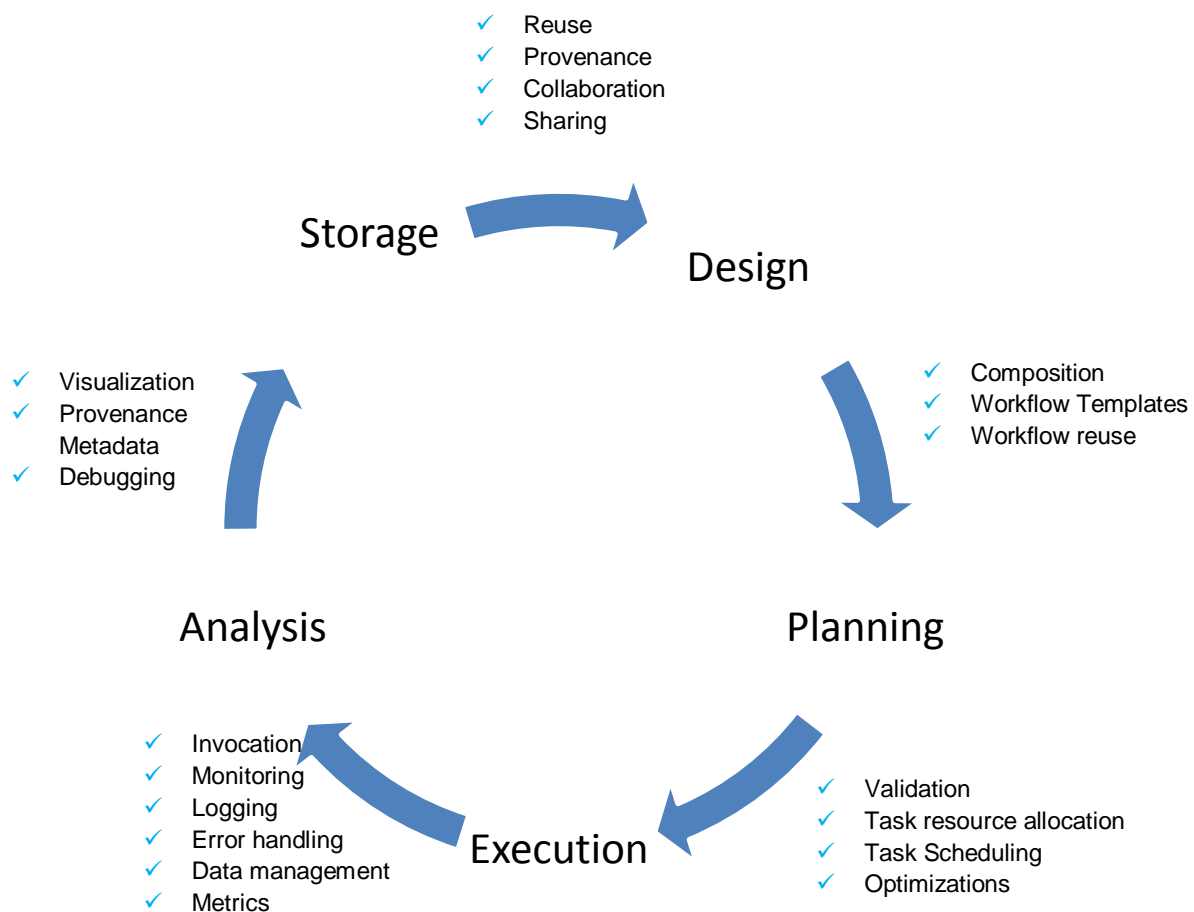


Figure 2.4 – Scientific workflow life cycle as proposed by [4]

2.5 SWMS's architecture

A simplified architecture of the SWMS high level components is presented in Figure 2.5. It is a merger of proposed architectures in [4], [22], [27], [26], [44], [45], [46].

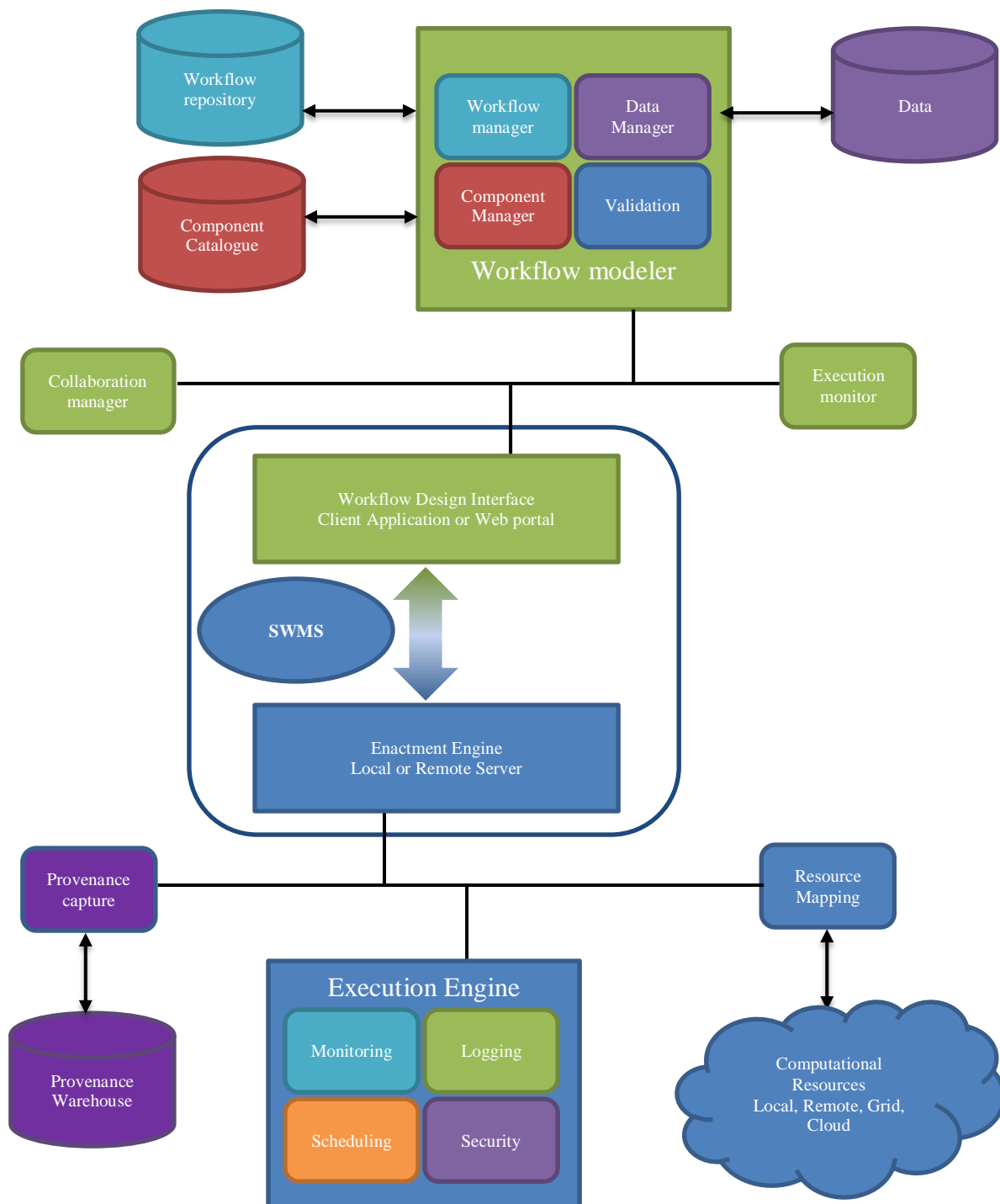


Figure 2.5 – SWMS high level components interaction

An SWMS is generally considered to be a combination of two main constituent parts. The Workflow Design Interface and the Enactment Engine. The Workflow Design Interface is usually a client application or a web portal that enables a user to model workflows (workflow modeler), execute them (execution monitor) and share them (collaboration manager). The workflow modeler enables the user to create workflows from the software components catalogue available and validates the workflow in terms of data type consistency or service availability. The Enactment Engine is responsible for executing the workflow (execution engine), allocating resources and keeping track or provenance data. Scheduling, monitoring, logging the workflow's execution details and any security issues are handled by the Execution Engine.

2.6 SWMS Review

The field of SWMS has been receiving considerable interest in recent years. Consequently, a number of implementations have been reported and several reviews of such systems have been published. Early on, in 2005, Yu and Buyya [28] presented a taxonomy of grid workflow systems. In 2006, Taylor et al. [29] published a book on E-Science workflows, presenting several systems and defining research questions. Tiwari and Sekhar [30] surveyed workflow systems for life sciences. The research questions set down at the National Science Foundation Workshop on Scientific Workflows of 2006 were recorded by Gil et al. [31]. In 2008, Barker and van Hemert [1], presented a concise survey of existing workflow technology from the business and scientific domain and made a number of key suggestions. At the same year V. Curcin and M. Ghanem [44] reviewed six systems considered state of the art in the field. McPhillips et al. [2] prepared a list of Desiderata for scientific workflow systems for scientists. Finally, C. Goble et al. [4] presented the challenges to be met by the advancing workflow technology. In 2009, Ludascher et al. [22] in his survey compared Scientific Workflows to the well-established Business Workflows. At the same year, the same author provides an overview of the characteristic features of scientific workflows and outlines their life cycle [25].

Deelman et al. [27] extracts a taxonomy of features from the end users view for the current scientific workflow systems. Sonntag et al. in their work in 2011 [26], after reviewing contemporary systems, proposed a conceptual architecture for scientific workflow systems based on business workflow systems, an approach encouraged by the Sixth International Workshop on Scientific Workflows (SWF 2011). This section provides an updated review of the main, most popular SWMS in order to present the current state of the art in the field.

The list of different workflow management tools used routinely is considerably large, exceeding 50 items[4]. This list includes popular SWMS like Taverna, Triana, Kepler, Pegasus, KNIME, Galaxy, Pipeline Pilot, InforSense KDE and Microsoft Trident but also BioWBI [32], GridBus [33], ICENI [34], and Magenta [35], GridNexus [36], ASKALON [37] and others. Table 2.1 presents a snapshot of the main popular representatives of workflow management tools and their main characteristics. A short informative description follows.

❖ **Taverna[5][6][7]**

Taverna is an open-source, Grid-aware workflow management system. It is used primarily by the bioinformatics, computational chemistry, medical imaging, social science and astronomy communities Taverna is actually domain independent. It is comprised of the Taverna Workbench graphical workflow authoring client, together with a workflow representation language, and an appropriate enactment engine. Taverna is implemented as a service-oriented architecture, based on Web service standards. Provenance plays an integral part in Taverna, allowing users to capture and inspect details such as who conducted the experiment, what services were used, and what the results of services provided. Taverna has been created by the myGrid [51] team.

❖ **Galaxy[16][17][18]**

Galaxy is a Web-based platform for data intensive biomedical research. It provides a framework for integrating computational tools and the environment for interactive data analysis, reuse, sharing and other. It allows nearly any tool that can be run from the command-

List of Scientific Workflow Applications					
	Application	URL	TECHNOLOGY/ PARADIGM	SCIENTIFIC FIELD	Last updated version
O P E N S O U R C E	Taverna [5][6][7]	http://www.taverna.org.uk/	Java based	Bioinformatics, chemistry, astronomy, data and text mining, music,	May - 12
	Galaxy [16]	http://galaxy.psu.edu/	Python based	Life Sciences, Bioinformatics	May - 12
	Pegasus [10]	http://pegasus.isi.edu/	Java based DAX Condor DAGMan	Bioinformatics, Astronomy, Botany, Chemistry, Physics, Ocean science, Neuroscience, Limnology, Genome analysis, Earthquake science, Climate modeling, Computer science, Helioseismology	Feb - 12
	Triana [8]	http://www.trianacode.org/	Java based WSFL TrianaService	Signal, text and image processing	Oct - 11
	Kepler [9]	https://kepler-project.org/	Java based MoML Ptolemy	Ecology and Geology	Jun - 11
	Knime [11][12]	http://www.knime.org/	Java based	Life Sciences, Chemo- and Bioinformatics, but also high performance data analysis	Mar-12
C O M M E R C I A L	Discovery Net , Inforsence, IDBS [14]	http://www.idbs.com/	--	life sciences, healthcare, financial services, sales & marketing analytics Life Science, Environmental monitoring, geo-hazard modeling	--
	Pipeline Pilot, Accelrys [13]	http://accelrys.com/products/pipeline-pilot/	--	Biology, Chemistry, Material Science	--
	Microsoft Trident [15]	http://research.microsoft.com/en-us/collaboration/tools/trident.aspx	Microsoft Workflows Engine XOML	Oceanography, Astronomy	--

Table 2.1 – List of popular Workflow applications

line to be wrapped in a structured well defined interface. It is open source and specially designed for the needs of bioinformaticians supporting sequence manipulation with built in libraries. It does not support any control flow operations or remote services. Additionally it does not use a workflow language but rather a relational database. The Galaxy workflow system allows for analysis using multiple tools incorporated to the system through WF that may be built and run or extracted from past runs, and rerun. All of Galaxy's operations can be performed using nothing more than a web browser. A recent Taverna-Galaxy integration allows the automatic generation of Galaxy tools from Taverna 2 workflows. The tools can

then be installed in a Galaxy server and become part of a Galaxy pipeline. Galaxy can also be instantiated on cloud computing infrastructures or can be interfaced with grid clusters.

❖ **Pegasus [10]**

Pegasus is a framework for mapping scientific workflows onto distributed resources including Grid and Cloud-based systems. It has found application in a number of fields such as Bioinformatics, Astronomy, Botany, Chemistry, Physics, Ocean Science, Neuroscience and others. The user defines an abstract workflow and then Pegasus maps and executes it onto available distributed computer resources through the use of Artificial Intelligence scheduling techniques. The mapping is done automatically concealing any technical and middleware details. Pegasus [10] is rather a workflow compiler than a SWMS. It uses a proprietary language at the abstract level which is the execution independent XML representation of a directed acyclic graph.

❖ **KNIME [11][12]**

KNIME (Konstanz Information Miner) is an open-source platform that supports data integration from various sources, processing, modeling, analysis and mining, as well as parallel execution. The user can create data flows visually, then execute the analysis steps and study the results. KNIME is primarily used in pharmaceutical research with some applications reported in other areas like customer resource management and data analysis (CRM), business intelligence and financial data analysis. KNIME is based on the Eclipse open source platform and is developed at the University of Konstanz, Germany.

❖ **TRIANA [8]**

Triana is an open source visual workflow-based problem solving environment that focuses on supporting services and workflow execution in distributed environments. It is designed to define, process, analyse, manage, execute and monitor workflows. For workflow and service execution Triana supports peer-to-peer systems and Grid environments that enable dynamic resource allocation. The environment has been used in a range of tasks, such as signal, text and image processing. Once designed, Triana workflows can be imported from or saved in a

variety of forms. Triana supports multiple abstract workflow languages by allowing different workflow readers/writers to be plugged in. Triana is developed at Cardiff University.

❖ **KEPLER [9]**

Kepler is an open-source scientific workflow engine used primarily in geology and ecology projects. Kepler provides an intuitive graphical user interface and an execution engine to help scientists edit and manage scientific workflows, collect provenance information related to the developed workflows, and generate reports on their executions over time. The execution engine is separated from the graphical user interface enabling the execution of workflows in batch, centralized or distributed mode. Kepler provides a large variety of computational models inherited from the Ptolemy II [49] system based at the University of California at Berkeley and uses a proprietary language. Kepler is developed by a cross-project collaboration to serve scientists from different disciplines and is based upon work supported by the US National Science Foundation.

❖ **DiscoveryNet [14]**

DiscoveryNet is a visual component integration-based workflow system that provides a graphical user interface to build workflows out of existing third-party tools and services. It also allows for service providers to publish their software components for data analysis and mining. The developed workflows are saved in an XML-based format called Discovery Markup Language (DPML). The DiscoveryNet technology and system was later commercialized as a series of products through InforSense4.

❖ **Pipeline pilot [13]**

Pipeline Pilot (PP) is a commercial package implementing a scientific workflow management system [13]. A limited version of it is provided free to academic communities. PP was built by Accelrys to automate scientific data management, analysis and reporting processes powered by a data pipelining engine. Pipeline Pilot provides a large set of domain-specific component libraries with an emphasis on bioinformatics and computational chemistry and so it is largely used as a data pipelining framework and reporting platform in these scientific domains.

❖ TRIDENT [15]

Trident, implemented on top of the Windows Workflow Foundation[50], uses a control flow-oriented modeling language based on the Extensible Orchestration Markup Language (XOML). The Microsoft system consists of independent components for workflow modeling and execution. It provides a workflow modeling tool like a text editor, a graphical composer and domain-specific workflow packages for astronomy, biology, meteorology or oceanography. The workflow is executed in Trident, but can also be executed by compiling it into a usual application and run on Microsoft .Net platforms. Trident is part of a collaborative project between The University of Washington, Monterey Bay Aquarium Research Institute and Microsoft.

2.7 Scientific workflow collaboration

As previously mentioned, one of the main advantages of these tools is their ability to promote scientific collaboration through sharing of workflows. An example of such initiatives is myExperiment [38]. myExperiment is a social networking site for researchers providing a Virtual Research Environment (VRE) designed for users to share, discover and reuse workflows [52].

Experts stress that workflows “*encapsulate scientific intellectual property*”. As such they must be stored, organized and easily retrieved. myExperiment aims to be an online scientific workflow repository for organizing, sharing and discovering analogous to online research paper management applications. Similar to the method used by an author to publish and distribute a paper, a researcher can publish the workflow to be easily accessed by interested scientists through myExperiment. Furthermore, users can tag and comment workflows and, create and join groups and exchange messages. Initially, myExperiment was built as part of the myGrid and Taverna projects for supporting bioinformaticians but, just like Taverna, it is

now used by a wider range of disciplines and supports different types of workflows for various scientific domains.

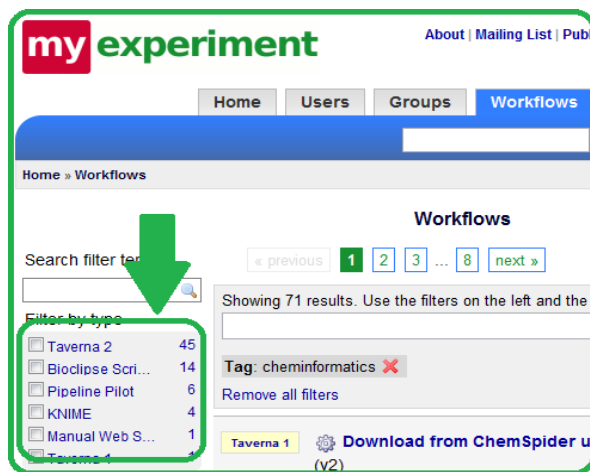


Figure 2.6 – myExperiment’s interface on <http://www.myexperiment.org/>

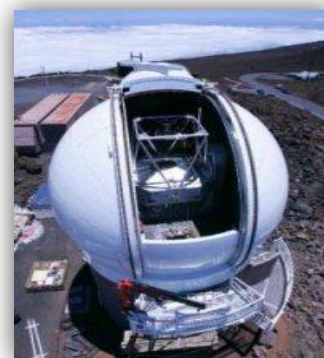
Currently myExperiment has over 5000 members, 250 groups, and 2000 workflows⁹. The main users of myExperiment are Taverna users, as the two systems are tightly integrated. However, myExperiment now stores workflows of other SWMSs like KNIME, KEPLER and Pipeline Pilot and more recently Galaxy.

2.8 Sample Current Projects

The following are some examples to illustrate the wide applicability and strength of the tools:

🌐 Pan-STARRS Sky Survey [39]

The Panoramic Survey Telescope and Rapid Response System (Pan-STARRS) is an example of a project utilizing workflow technology. The workflows are being built on the Trident workflow workbench. The project uses workflows to ingest multiple terabytes of data that come out of a panoramic

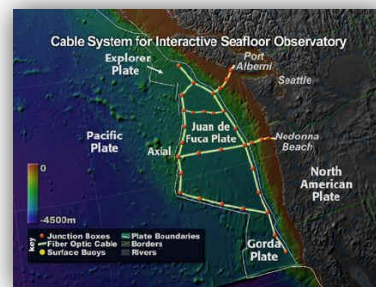


⁹ Source: www.myexperiment.org on 12/05/2012

survey telescope at Hawaii into a SQL Server database. The survey is building a time series of detections of five billion astronomical objects by scanning the entire visible sky in four nights, producing about 1 TB of data per week, after initial processing to extract object-specific data from the raw images.

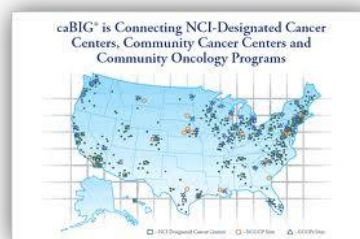
🌐 NEPTUNE oceanography project[39]

North East Pacific Time-integrated Undersea Networked Experiments (NEPTUNE) is a Regional Cabled Observatory on the Juan de Fuca plate that extends a 1500 Km long fiber optic cable to a network of sensors widely distributed across, above, and below the seafloor. The sensors are continuously streaming data back to shore for analysis by oceanographers. Trident, is NEPTUNE's scientific workflow workbench enabling scientists to explore and visualize oceanographic data in real-time while providing an environment to compose, run and catalog workflows.



🌐 Biomedical Informatics Grid (caBIG)[40]

caBIG, sponsored by the US National Cancer Institute (NCI)[47], is an information network enabling cancer researchers and physicians to share data and knowledge, and thus accelerate the discovery of new cancer treatment methods. Cancer Grid (caGrid)[48], the underlying infrastructure of caBIG, is an extension to the Taverna workflow system designed and implemented to ease building and running caGrid workflows.



BioVeL – Biodiversity Virtual e-Laboratory [41]

BioVeL is a virtual e-laboratory that supports research on biodiversity issues using large amounts of data from cross-disciplinary sources.

BioVeL uses Taverna workflows for data processing and myExperiment for workflow sharing and collaboration. The

BioCatalogue (an online registry of biological Web Services) [42] is

also used. The project is led by the School of Computer Science and Informatics at Cardiff University.

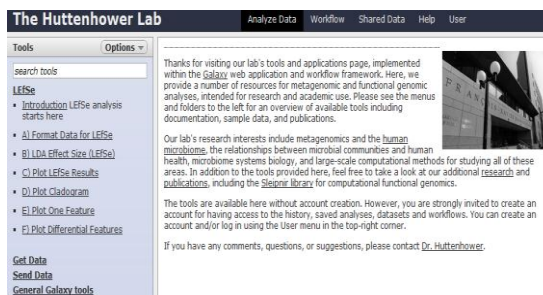


Galaxy – Huttenhower Lab [43]

The Galaxy platform is used to provide

access to metagenomic and functional genomic analyses, intended for research and academic use. The LDA Effect Size (LEfSe) project (Segata et. al 2011)-aiming

to enable high-dimensional biomarker discovery is implemented using the Galaxy SWMS. The Galaxy server is maintained by the Huttenhower Lab, led by Dr. Huttenhower at the Department of Biostatistics, Harvard School of Public Health.



Chapter 3

Open Source Scientific Workflow Systems

3.1 KNIME


3.2 Taverna

3.3 Galaxy

3.4 Summary

“...One of the hottest technology areas out there: making sense of data!”¹⁰

3.1 KNIME

KNIME  was developed by the Chair for Bioinformatics and Information Mining at the University of Konstanz, Germany. It is a modular, expandable and highly scalable platform encompassing various data loading, transformation, analysis and visual exploration models. KNIME has found application in life sciences, pharmaceutical industry, financial services, publishers, Retailers and E-tailers, manufacturing consulting firms, government and research¹⁰.

The platform enables the user to visually create data flows, execute selected analysis steps, and later investigate the results through interactive views on data and models. At the same time it serves as an integration platform enabling scientists to use different projects in a single environment and facilitating developers design and implement new tools.

¹⁰www.knime.org

KNIME is written in Java and its graphical workflow editor is implemented as an Eclipse[54] plug-in.

As KNIME is based on the Eclipse platform it is easily extensible. KNIME functionality is enriched by integrating the functionality of different open source projects that essentially cover all major areas of data analysis such as WEKA (Witten and Frank (2005)) for machine learning and data mining, the R environment (R Development core team (2007)) for statistical computations and graphics, JFreeChart (Gilbert (2005)) for visualization (various line, pie and histogram charts), CDK -Chemistry Development kit and other.

KNIME is an open source software available as a desktop version to all users. It is also possible to set a KNIME server for executing workflows through a portal interface and for maintaining an online workflow repository for workflow sharing. There is also support for cluster execution of the workflows. However the latter are part of a professional package.

One highlight of KNIME's latest additions is the ability to support PMML. The Predictive Model Markup Language (PMML) is an XML-based markup language that enables applications to define models related to predictive analytics and data mining and to share those models between PMML-compliant applications.¹¹ As a result a model developed by KNIME can be exported and then used in another data mining engine. Another characteristic is the addition of database ports that are JDBC-compliant that work directly in the database enabling even preview of the actual data inside the database tables. JDBC is a Java-based data access technology that provides methods for querying and updating data in a database.

Although written in Java, KNIME, permits running Python, Perl and other code fragments through the use of special scripting nodes. This is extremely useful as a lot of scientific work is currently under the form of Python or Perl scripts.

¹¹ Data Mining Group <http://www.dmg.org>

3.1.1 KNIME ENVIRONMENT

Figure 3.1 below describes the section of the KNIME workspace (image from www.knime.org).

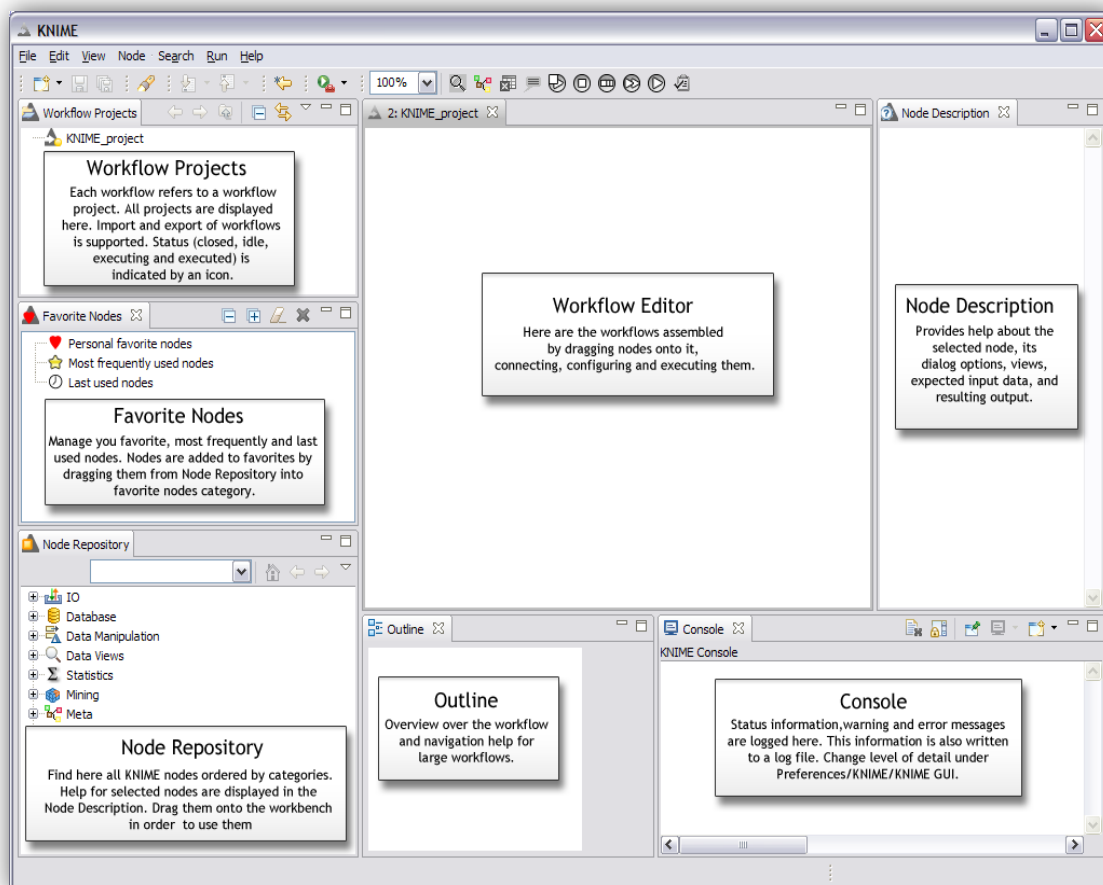


Figure 3.1 – KNIME workspace from www.knime.org

The user can work on several workflow projects. Each workflow can be edited in the Workflow Editor area. The user can create a workflow by dragging nodes into the Editor area either from the Node Repository or from the Favorites Nodes section. When a node is selected the Node Description displays relevant information about the node. Large workflows can be navigated by using the Outline panel. Finally the Console displays valuable information like

warning and error messages on node configuration and execution. Figure 3.2 shows the KNIME workspaces when a project is loaded.

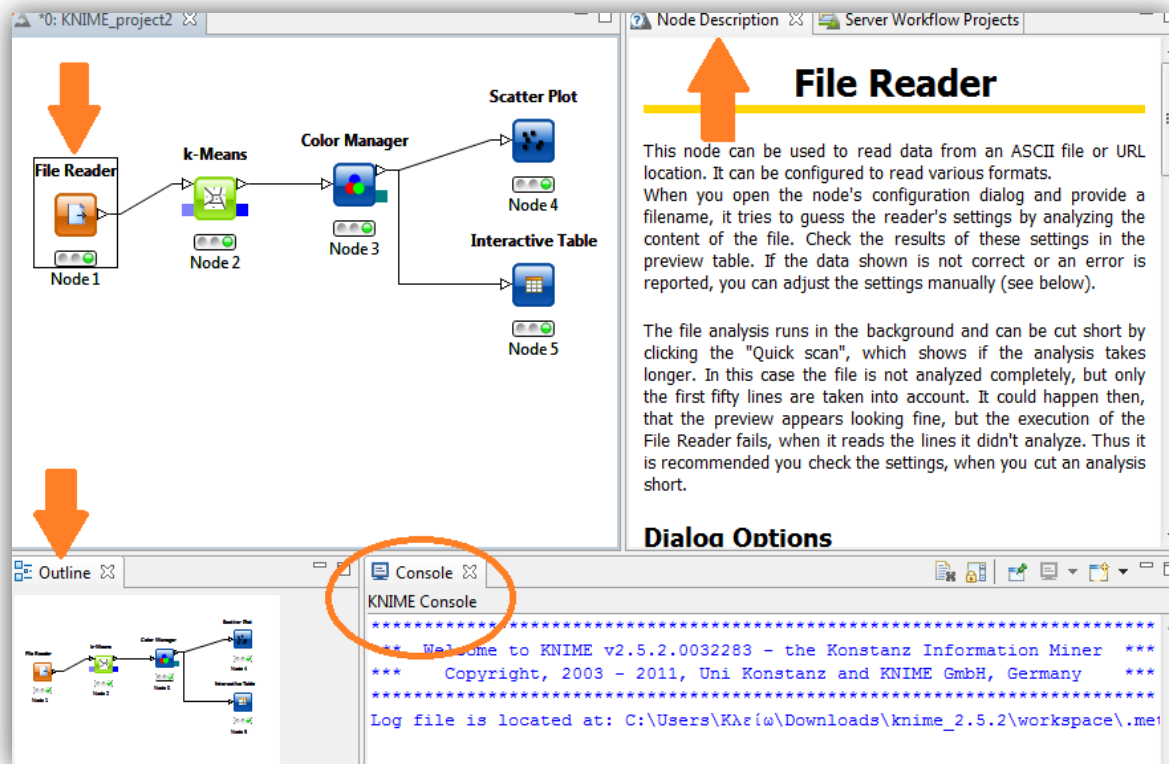


Figure 3.2 – KNIME workspace with project details

3.1.2 COMPONENTS OF A KNIME WORKFLOW

A workflow in KNIME consists of nodes and directed edges forming a directed acyclic graph (DAG). Each node is a processing unit with one or more input and/or output ports. It processes data or data models. The type of processing varies from basic data operations to reporting to computationally intensive data modeling e.g. Decision Trees. All the categories of nodes available are listed in the Node Repository. Along the directed edges data or models are transferred from an out-port to the in-port of another node.

The type of ports supported is shown in Table 3.1. The data ports of each node are also marked differently according to the type supported. The editor allows the connection only between ports of the same type.

Sometimes, to maintain, a workflow design simple, or to wrap a functionality in one node, sub-workflows can be declared. These are known as Meta nodes. Meta nodes are nodes that are actually a workflow of nodes.






Port Types			
Description / Data type	image	Description / Data Type	image
White triangle / flat data tables	<p>File Reader</p>  <p>Node 1</p>	Dark Cyan square / General purpose port for structured data	<p>Naive Bayes Predictor</p>  <p>The model to use Node 21</p>
Blue square / PMML Data model	<p>PMML Reader</p>  <p>Model Node 22</p>	Grey square / Unknown type	<p>Model Reader</p>  <p>Object Node 18</p>
Brown square / Database port	<p>Database Column Filter</p>  <p>Database Connection Node 16</p>		

Table 3.1 – Port types for KNIME nodes

3.1.3 BUILDING A WORKFLOW

In order to build a workflow you simply drag a node from the Node Repository to the Workflow Editor space. Once placed, a user can click on it to get its description in the node description section. Using right click, a list of selected options appears which enable the user

to configure the parameters of the node and execute it among other things. Each node can be executed on its own, once the data are loaded.

When a node is dragged in the workflow editor its status is set to red color meaning it needs to be configured. After making the necessary configurations the status is turned to yellow. Green is the color stating that a successful execution has already taken place (Table 3.2).




Node status		
<p>File Reader</p>  <p>Node 6</p>	<p>File Reader</p>  <p>Node 1</p>	<p>File Reader</p>  <p>Node 1</p>
needs to be configured	ready to be executed	successfully executed

Table 3.2 – Node's Status in KNIME

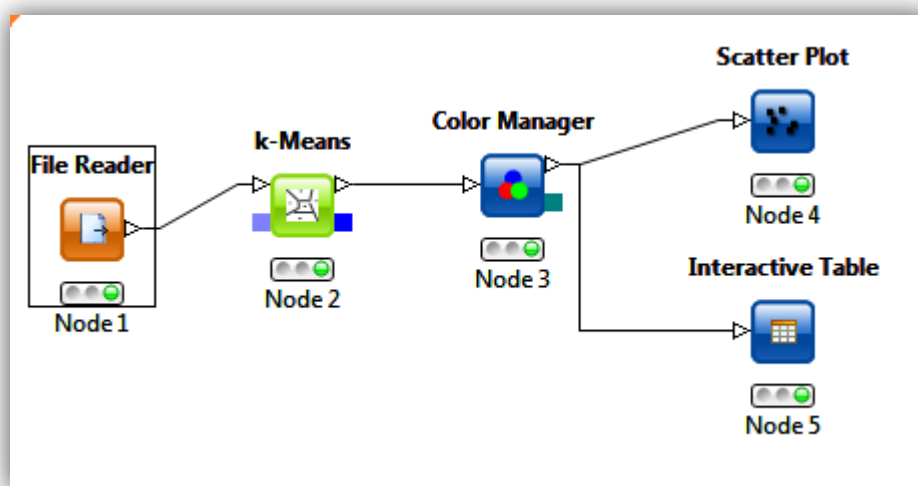


Figure 3.3 – Sample workflow in KNIME

In the sample workflow in Figure 3.3 a file is read, passed through a k-mean modeler and then colored so that the scatter plot can show comprehensive results.

KNIME also has a KNIME Example Flow Server window panel that gives access to an online public database of example workflows that can be directly downloaded into the workbench.

(Fig. 3.4)

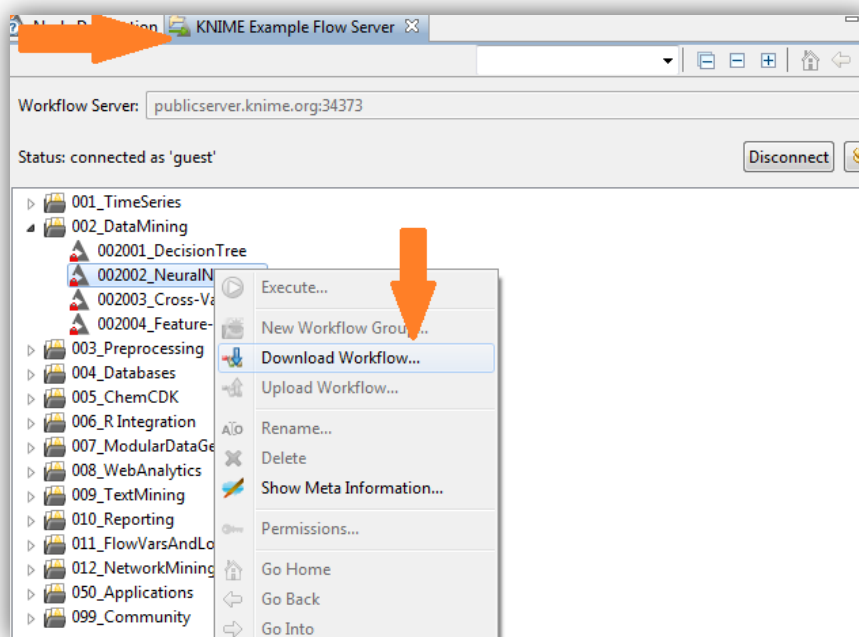


Figure 3.4 – KNIME’s Example Flow Server workspace

3.1.4 RUNNING A WORKFLOW

In KNIME the workflow is executed either node by node or as a pipeline. In order for a node to pass the data to the successor node it must process all the input and finish execution. By design each node stores its workflow structure, its settings and result data. Therefore is possible to add a new node without the need for preceding nodes to be executed again.

Also, every time the workflow is restored in the workflow editor, its previous execution status, its results data and settings are loaded as well. If the workflow is exported again (if selected) then all this information is exported as well. So, when imported to another KNIME workbench the workflow maintains its previous execution information.

3.1.5 VIEWING THE RESULTS

The results can be in any possible data form. KNIME provides a data table view for each node displayed in Figure 3.5. Dedicated Data view nodes can plot line plots, pie charts, histograms etc (Fig. 3.6). Moreover, it is possible to create reports with embedded chart images.

Row ID	D sepal le...	D sepal w...	D petal le...	D petal wi...	S class
Row40	5	5.5	1.3	0.3	Iris-setosa
Row41	4.5	2.3	1.3	0.3	Iris-setosa
Row42	4.4	3.2	1.3	0.2	Iris-setosa
Row43	5	3.5	1.6	0.6	Iris-setosa
Row44	5.1	3.8	1.9	0.4	Iris-setosa
Row45	4.8	3	1.4	0.3	Iris-setosa
Row46	5.1	3.8	1.6	0.2	Iris-setosa
Row47	4.6	3.2	1.4	0.2	Iris-setosa
Row48	5.3	3.7	1.5	0.2	Iris-setosa
Row49	5	3.3	1.4	0.2	Iris-setosa
Row50	7	3.2	4.7	1.4	Iris-versicolor
Row51	6.4	3.2	4.5	1.5	Iris-versicolor
Row52	6.9	3.1	4.9	1.5	Iris-versicolor
Row53	5.5	2.3	4	1.3	Iris-versicolor
Row54	6.5	2.8	4.6	1.5	Iris-versicolor
Row55	5.7	2.8	4.5	1.3	Iris-versicolor

Figure 3.5 - Table view of a k-means prediction algorithm run on a sample file

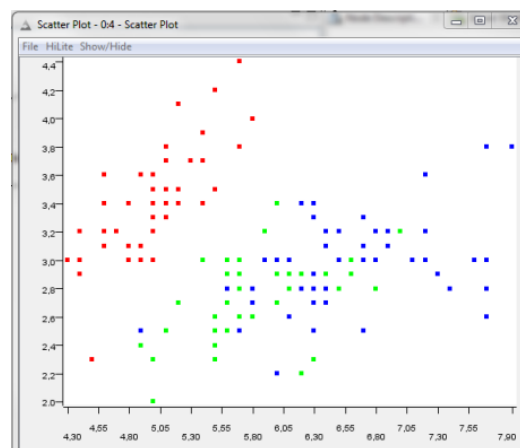


Figure 3.6 - Scatter Plot of the same file

3.1.6 ARCHITECTURE

In KNIME, a flow starts with a node that imports data from an input source such as a text file or a database. The data is stored in an internal table-based format consisting of columns with a certain data type (integer, string, image, molecule, etc.) and an arbitrary number of rows conforming to the column specifications. These data tables are sent along the connections to other nodes that modify, transform, model, or visualize the data [12]. Figure 3.7 summarizes data flow between nodes.

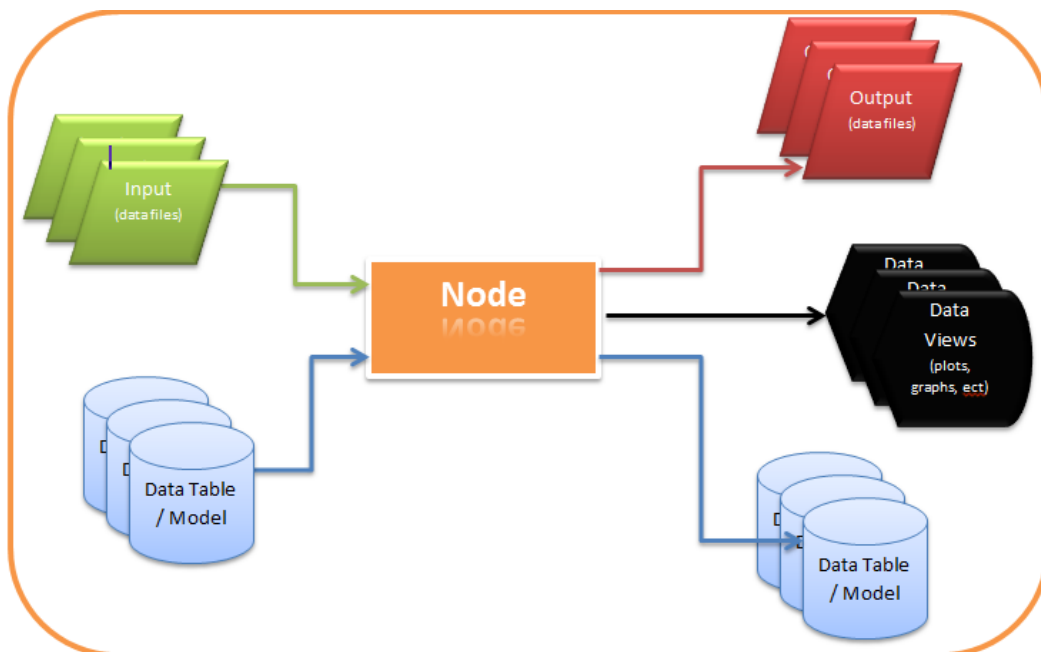


Figure 3.7 – Nodes data input and output

Dedicated special node types enable control flow operations such as iterations, if and case statements. (Fig. 3.8)

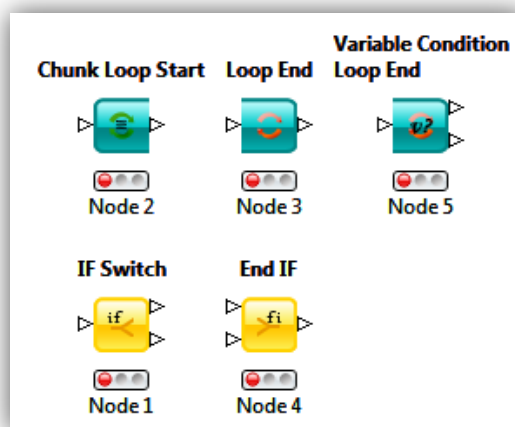



Figure 3.8 – Control flow nodes

Unlike, other SWMS, e.g. Taverna described in section 3.2 of this chapter, nodes in KNIME first process the entire input table and then forward the results to successor nodes. The advantages that arise from having each node store its results permanently are numerous. Workflow execution can be stopped and resumed later on at any node and any intermediate results are available for viewing. Any modifications, e.g. new nodes, can make use of

previous data runs and therefore re-running the entire workflow is not necessary. Also, by keeping track of the status of nodes (configured or executed) the workload can be distributed accordingly to threads, clusters or the grid to achieve parallelism.

3.2 Taverna

The Taverna  SWMS is a more general tool than KNIME, aiming in supporting the automation of service-based and data-intensive processes as explained in the white paper “Taverna: a tool for building and running workflows of services” [6]. Being more general also makes it more difficult to accomplish specialized tasks. By design, Taverna has a distinct web flavor.

Taverna is an open source domain-independent Workflow Management System. It was developed as part of the myGrid consortium with the Life Sciences field in mind but it has found application in numerous domains including Arts, Astronomy, Bioinformatics, Chemistry, Data and text mining, Education, Engineering, Geoinformatics, Music, Social Sciences, Physics [52]. Taverna is usually deployed as a Standalone Workbench. However it can also be deployed as a server, on a grid, on a cloud and behind a portal to be used by a number of projects.

From the advent of its design Taverna was an application that applied Web Services technology to workflow design. That meant that tools created using different programming languages (e.g. Java, PERL, Python, etc) or platforms (Unix, Windows, etc) could now be accessed via a web service interface eliminating any need for integration. The same applied for the databases available on the web. As a result, researchers could now design and execute a pipeline of web services, without any programmatic knowledge.

In the new release of Taverna (2010) important new features have been included. Parallelism, both intra-process and inter-process, asynchronous service support and separation of data and process spaces to support scaling to arbitrary data volumes.

The Taverna suite is written in Java. It includes the Taverna Engine (used for enacting workflows) that powers both the Taverna Workbench (the desktop client application) and the Taverna Server (which allows remote execution of workflows). Taverna is also available as a Command Line Tool for a quick execution of workflows from a terminal.¹²

A vital component of Taverna's open architecture is the plug-in functionality. Various plugins for Taverna have been developed including the XPath plugin, REST plugin, BioCatalogue plugin, PBS plugin, SADI plugin, CDK plugin, Opal plugin, caGrid plugin, XWS plugin, gLite plugin.

3.2.1 TAVERNA ENVIRONMENT

To create a workflow a user must use the Design perspective of the workbench. There one can select a service from the Service panel, and then drop it on the Workflow diagram area. Each service can be either a remote or local service. The Details pane displays information about the service, while the Validation pane displays the validation results of all the services. Figure 3.9 displays the design view of the workbench.

Workflow Explorer is unique in Taverna. It provides an alternative view of the entire workflow that allows certain actions to be taken on the services but at the same time it serves as a navigation tool. The input and output parameters of each service are clearly shown. By right clicking on an input port for example, one can define the input port's constant value. Remember that KNIME has the Outline View used for navigating large workflows.

¹² www.taverna.org

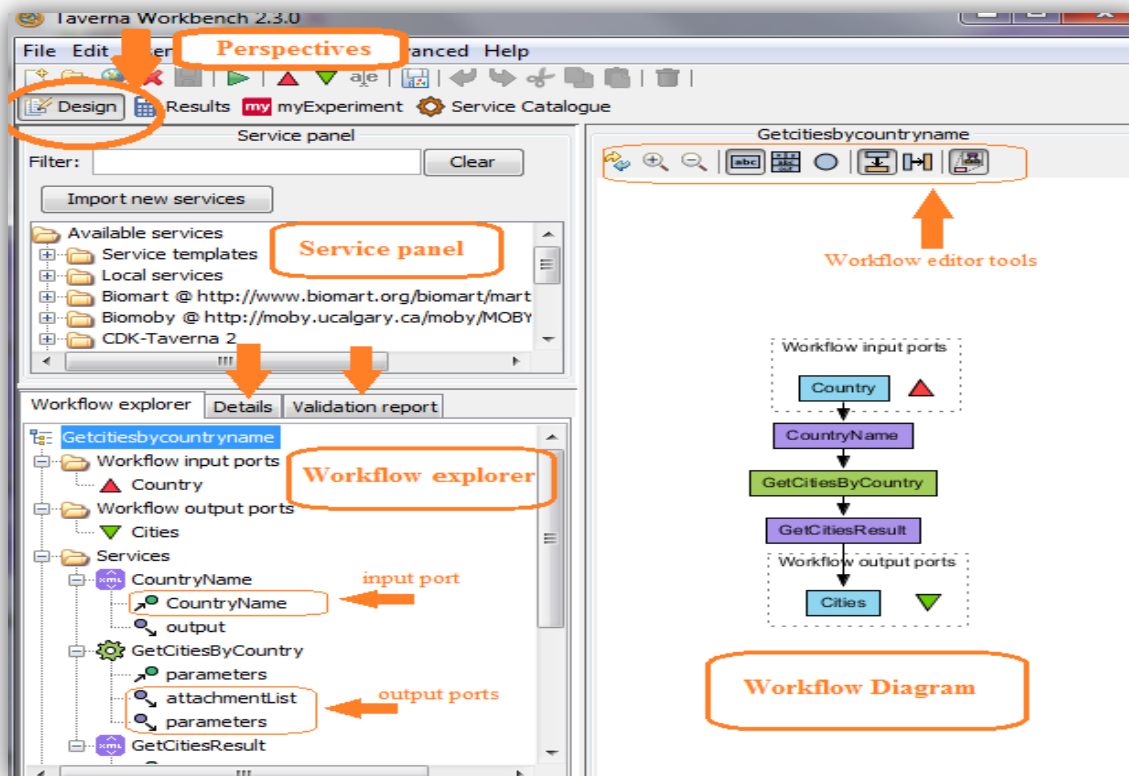


Figure 3.9 – Taverna workbench

One feature of Taverna, also present in the KNIME desktop, is workflow sharing. The user can have direct access to an online repository of workflows and directly open them inside the Workflow diagram, by selecting the myExperiment perspective (Fig. 3.10). Unlike, KNIME's repository, myExperiment is a social collaboration site, where the example workflows are uploaded by the users themselves.

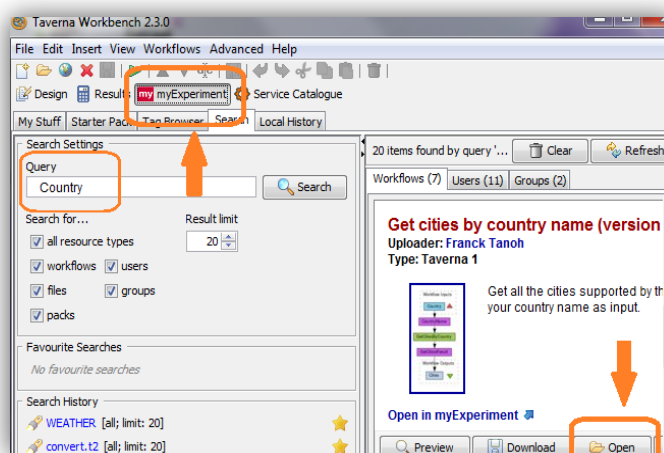


Figure 3.10 – myExperiment perspective

Service catalogue functionality can be accessed from within the Taverna Workbench by selecting the Service Catalogue Perspective [53].

3.2.2 COMPONENTS OF A TAVERNA WORKFLOW

The nodes in a Taverna workflow are called processors. They represent software components, usually local or remote services. Each processor has its input and output ports. Each edge between two processors is either a data link or a control link. The workflow computation proceeds by passing data from one processor to the next through the data links. The processors can have a data dependency, and the output of the first processor is fed in the input port of its dependents. The processors have a preset order of execution denoting that the second can only be executed after the first is finished. Special links called control flow links can also be defined, setting order of execution when no data dependency exists between two processes. To encourage reuse and modular design, nested workflows can be added to a workflow, having input and output ports just as any other service. This feature resembles the KNIME Meta nodes discussed previously.

In Figure 3.11a sample workflow in the Taverna workbench is displayed. The workflow can be downloaded from the myExperiment website. It has 3 input ports, one output port and it converts chemical identifiers from one format to another.

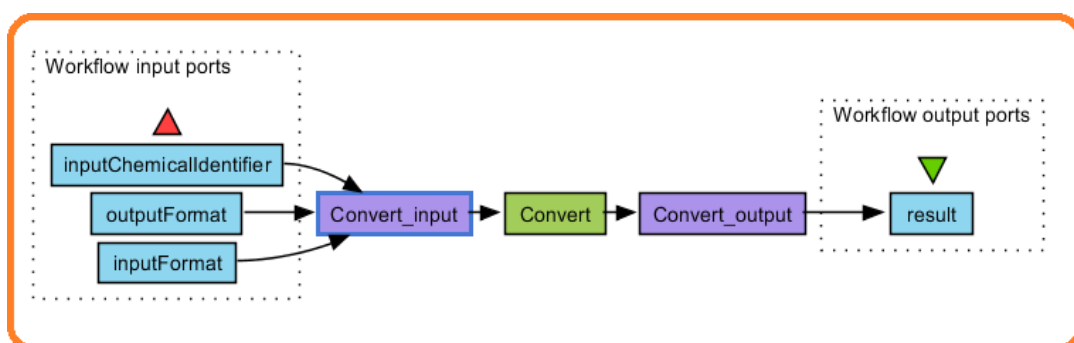
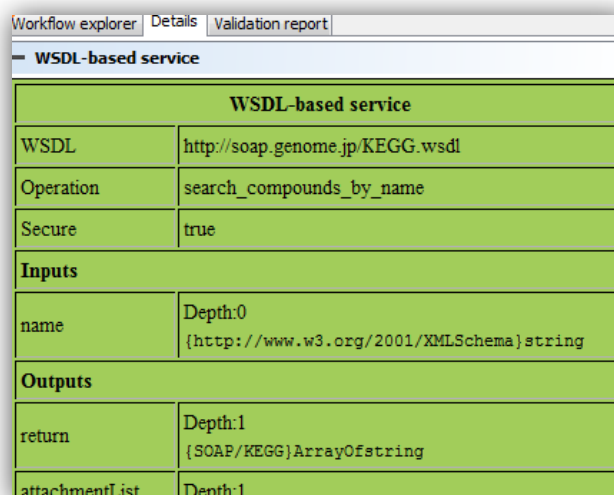


Figure 3.11 – Sample Workflow in Taverna workbench

3.2.3 BUILDING A WORKFLOW

In order to build a workflow you simply drag a processor/service from the Service Panel either to the Workflow diagram area or to the Workflow explorer area. For every service added both areas are updated. The user can use both areas as he pleases. A disappointing difference to the KNIME editor is that the placement of the processors is done automatically and no manual configuration is possible. Instead a set of tools are available for setting horizontal or vertical alignment.

Each service type has a distinct color. For example, a standard SOAP service is green, and local Java class operation is purple. When a service is clicked upon, its details appear in the Details pane. Compared to KNIME, the level of detail about the processor in concern is lower. For example, for most services the details displayed are the input and output ports, the path of the service and the service provider while for other no details are visible. In Fig 3.12 such details are shown for the SOAP WSDL type service of shown previously in Fig 3.11.



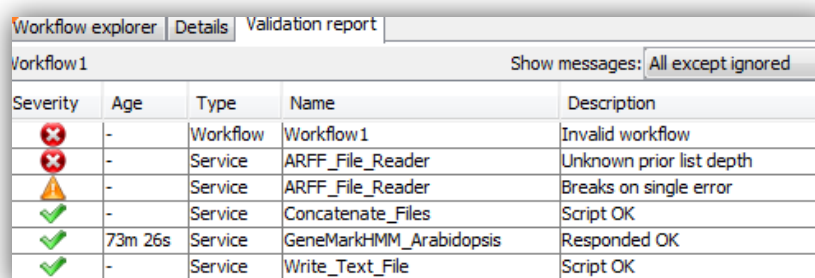
WSDL-based service	
WSDL	http://soap.genome.jp/KEGG.wsdl
Operation	search_compounds_by_name
Secure	true
Inputs	
name	Depth:0 {http://www.w3.org/2001/XMLSchema}string
Outputs	
return	Depth:1 {SOAP/KEGG}ArrayOfstring
attachmentList	Depth:1

Figure 3.12 – Processor Details in Taverna

The links of the workflow can be either drawn using the mouse or placed using right clicks. Depending on the kind of service, a set of options are available when using right click including Validate, Configure Running (creates loops and adds parallelism), Configure

security (where applicable handles user authentication for accessing online tools), Run after (creates control flow links), Show/ Hide ports, Link to and Link from (setting the data flow links).

Prior to running the workflow, validating the services is an option to make sure the workflow is in proper order and to identify possible problems. Taverna can check the validity of data types, input and output ports, scripts which are involved in services, and whether invoked external Web services are online. Any issues detected are reported in the validation window. In Fig 3.13 a random “Validation report” is presented.



Severity	Age	Type	Name	Description
✘	-	Workflow	Workflow1	Invalid workflow
✘	-	Service	ARFF_File_Reader	Unknown prior list depth
⚠	-	Service	ARFF_File_Reader	Breaks on single error
✔	-	Service	Concatenate_Files	Script OK
✔	73m 26s	Service	GeneMarkHMM_Arabidopsis	Responded OK
✔	-	Service	Write_Text_File	Script OK

Figure 3.13 – Validation report in Taverna

3.2.4 RUNNING A WORKFLOW

By pressing the run button the workbench switches to the Results perspective (Fig. 3.14). If the workflow has any inputs defined then the *Run Dialog* pops up to enable the user to specify values for the input ports of the workflow. All of the details concerning the execution of a workflow are shown on the results view of the workbench. Firstly, the progress of the current run can be monitored. Secondly, for each service called the execution details, e.g. the execution time, are shown on the progress report tab. Moreover, the user can select and preview any of the previous runs. At the bottom of the window the workflow results are displayed. Input, output and result data can be previewed.

The screenshot displays the Taverna Workbench 2.3.0 interface in the Results perspective. The top menu bar includes File, Edit, Insert, View, Workflows, Advanced, and Help. The toolbar contains various icons for workflow management. The 'Workflow runs' panel shows a table of runs with the following data:

Name	Status	Que...	Itera...	Itera...	Aver...	First ...	Last ...
Getci	Finished	-	-	-	3,4 s	03:47:00	03:47:03
C	Finished	0	1	0	18 ms	03:47:02	03:47:03
G	Finished	0	1	0	623 ms	03:47:03	03:47:03
G	Finished	0	1	0	5 ms	03:47:03	03:47:03

The 'Workflow results' panel shows a tree view with 'Value 1' selected, displaying XML data for 'Country' and 'City'.

```

<Table>
  <Country>Cyprus</Country>
  <City>Larnaca Airport</City>
</Table>
<Table>
  <Country>Cyprus</Country>
  <City>Athalassa</City>
</Table>

```

Figure 3.14 – Results perspective in Taverna

3.2.5 VIEWING RESULTS

In the example above in Figure 3.14 the results are shown as text. Taverna, also displays images, graph plots, graph diagrams and several chemical data formats. Taverna is able to do this by applying a renderer to the output.

3.2.6 ARCHITECTURE

When the user designs a workflow, a workflow specification is created. For this specification to be executed it is subsequently translated into a Taverna object model. In this model the processors of the workflow are represented by objects and the data links by method invocations. Each processor object can implement one or more activities where an activity is an executable component. Each component can be as simple as a local service call or it may be a workflow itself. A simplified view of Taverna processors activities is given in Figure 3.15.

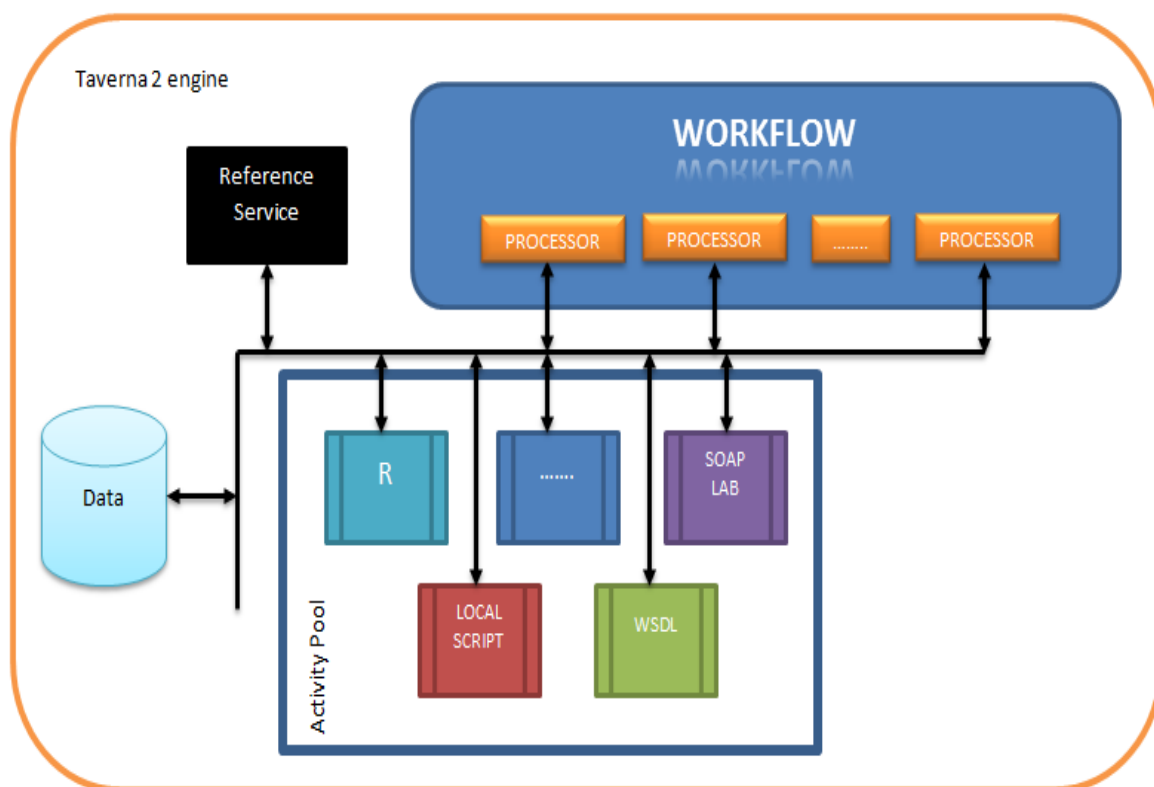



Figure 3.15 – A simplified view of Taverna processors activities

A processor independently starts its own execution as soon as all of its input ports are populated by a data item. Upon completing the execution the new data produced at the output ports are propagated to the next processor.

Taverna uses a Reference Service that proxies the actual data by references to them. During a workflow run, instead of moving the actual data, the references are passed along the data links. To access the data the Reference service is called. When two processors have no data dependencies they can begin parallel execution as soon as their input data is received. This achieves inter-processor parallelism. Intra-process parallelism is achieved by the feature of Taverna called the implicit iteration framework. If you connect a set of data objects to a process that expects a single data item at a time, the process will iterate over each sequence. Each iteration can be considered independent and thus is processed independently. Additionally, pipelining is supported when there is a chain of processors iterating and forwarding the output as soon as it is created.

In KNIME, each node stores its results permanently. In Taverna, this is true for each workflow. Each time a workflow is executed provenance information is generated by the Taverna Engine. The information is kept in memory for the current session. By having the appropriate options enabled this information can be saved in a database to be available for any further sessions. Taverna provenance data contains information about the workflow run, such as date and time of the run, intermediate values generated by services in the workflow during the run, as well as the final results¹³.

3.3 Galaxy

In order to use Galaxy  all you need is a web browser and of course a sound knowledge of bioinformatics. Galaxy was built to simplify the process of genomic analysis by providing a set of easy to use tools for the retrieval and analysis of large amounts of data. Actually the main server is instantiated with genomic dedicated tools. It is however possible to extend the set of tools for other tasks as well. And this is done easily, in a straightforward way.

¹³ www.taverna.org

Galaxy, as stated in [16],[18] is an open, web based platform. Its primary design considerations were accessibility, reproducibility and transparency. Galaxy is accessible by scientists with no programming knowledge through the use of galaxy tools. It produces reproducible computational analysis by generating metadata for each analysis step through the automated production of Galaxy History items. It promotes transparency by enabling sharing of data, tools, workflows, results and report documents.

Galaxy is set up as a free, public, internet accessible resource at UseGalaxy.org. It can even be used using a guest account. Creating an account is a simple process and allows a user to run a maximum of 8 concurrent jobs and use a total of 250GB for storage. In the top right corner of the Galaxy interface the percent of quota limit used by a user account is noted within a bar icon. Data transfer and data storage are not encrypted. There are also other Galaxy servers installed by institutions either public or semi-public. Alternatively local Galaxy servers can be set up by downloading and customizing the Galaxy application. Galaxy is available as a standalone package only for Linux environments. It includes an embedded web server and an SQL database. It is also possible to instantiate Galaxy instances on the cloud.

The Python programming language is the primary implementation language of the Galaxy framework. Galaxy was developed with the cooperation of the Center for Comparative Genomics and Bioinformatics at Penn State, and the Biology and Mathematics and Computer Science departments at Emory University.¹⁴

3.3.1 GALAXY ENVIRONMENT

The Galaxy portal consists of the analysis workspace, the workflow editor, the public repositories and the visualization environment. The user can navigate from one to the other using the navigation bar at the top of the Galaxy window. Through this portal the user can

¹⁴ <http://galaxy.psu.edu/>

create, share and view Galaxy objects: Histories, Workflows, Datasets, Pages and Visualizations.

The analysis workspace (Fig. 3.16) enables the user to run individual tools and workflows. On the left, the tool panel lists all the possible tools the user can select just by clicking. The list of tools contains specialized genome processes which are either data retrieval or data analysis tasks. On the right hand side of the window the history panel, displays the most recent history items. Every tool run by the user generates a new history item. Every history item can be re-executed, used in subsequent analyses, visualized, downloaded and annotated. Moreover, the sequence of the history items is used to automatically create workflows through the option Extract workflow. The middle column is the details pane that displays the tool's interfaces. When a workflow is run, the details panel displays the workflows details.

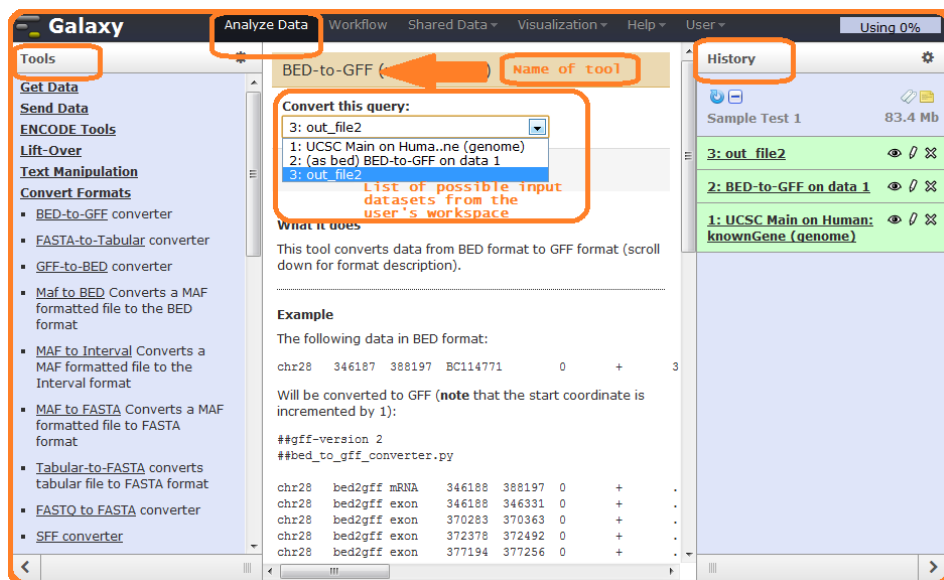


Figure 3.16 - Galaxy's Analyze Data Interface

The analysis workspace is also used to create datasets, so that, later on these datasets will be used as input to other tools or workflows. Support for retrieval of data from online data libraries is an important feature of Galaxy. In Figure 3.17 the tool Table browser is used to query the UCSC genome library. UCSC is a library hosted by the University of California,

Santa Cruz which contains the reference sequence and working draft assemblies for a large collection of genomes¹⁵.

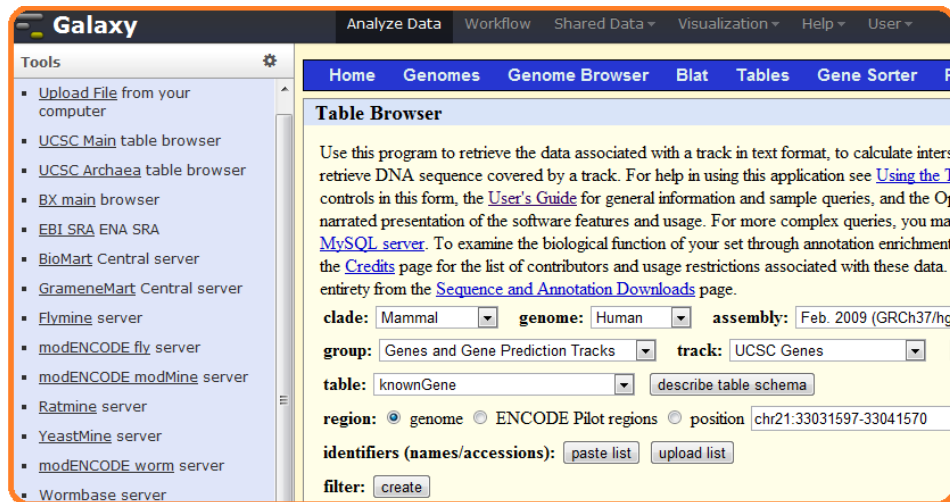


Figure 3.17 - UCSC Table Browser

On the Navigation Bar other useful links are the Shared Data and the User menu. The Shared Data menu gives access to public Data Libraries, and Published Histories, Workflows, Visualizations and Pages of other galaxy users. On the other hand, the User menu gives access to the user's Histories, Datasets and Pages. (Fig. 3.18)

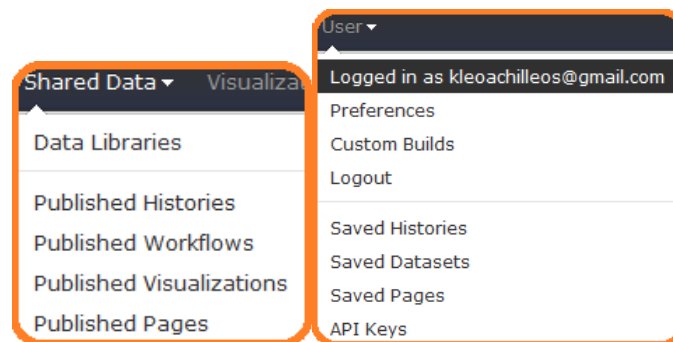


Figure 3.18 - Options Dialog in Shared Data and User menu

Workflows are accessible through the Workflow editor. The workflow editor is the graphical interface for creating and editing workflows. Running workflows is done through the analysis workspace. The Visualization menu is Galaxy's visualization and visual analysis

¹⁵ <http://genome.ucsc.edu>

environment. It supports genome specific datasets from within Galaxy. This feature is enabled on the server but not by default on local instances.

Pages are a feature unique to Galaxy. They are online documents used to describe the analysis performed but also to provide links to the Galaxy objects Histories, Workflows, Datasets that were used in the analysis. This enables the reader of the document to have direct access to the dataset used, to import the workflow and reproduce the experiment himself. And it makes it even easier for another scientist to continue and build upon a previous work. In Figure 3.19 a sample page shared by a user is shown.

The screenshot shows a web browser window with the Galaxy logo and 'Analyze Data' button. The page title is 'Windshield splatter analysis with the Galaxy metagenomic'. Below the title, there is author information: 'SERGEI KOSAKOVSKY POND^{1,2,*}, SAMIR WADHAWAN^{3,6*}, FRANCESCA CHIAROMONTE⁴, GURUPRASAD ANANDA^{1,3}, WE'. A note says 'CORRESPONDENCE SHOULD ADDRESSED TO [SKP](#), [JT](#), OR [AN](#)'. The 'How to use this document' section explains that it is a live copy of supplementary materials. Below this, there is a 'Galaxy History' section with a plus sign icon and a link to 'Galaxy History Comparison of G'. A text line says 'This is the Galaxy history showing a generic analysis of metagenomic data. (This corresponds to the "A complete metagenom'. Below this is a 'Dataset' table with three rows, each with a plus sign icon and a link to 'Galaxy History'. The table rows are: '1: 454 reads', '2: 454 qualities', and '3: Select high quality segments on data 2 and data 1'. The 'Galaxy History' and 'Dataset' sections are highlighted with orange boxes.

Figure 3.19 - A shared Galaxy page

3.3.2 COMPONENTS OF A GALAXY WORKFLOW

A Galaxy workflow consists of tools. Each tool has its own distinct interface to set and manage its parameters for execution, input and output. A link associates two tools, and can be set only if the corresponding data types agree. A sample workflow is presented in Figure 3.20 that converts a dataset from one format into another.

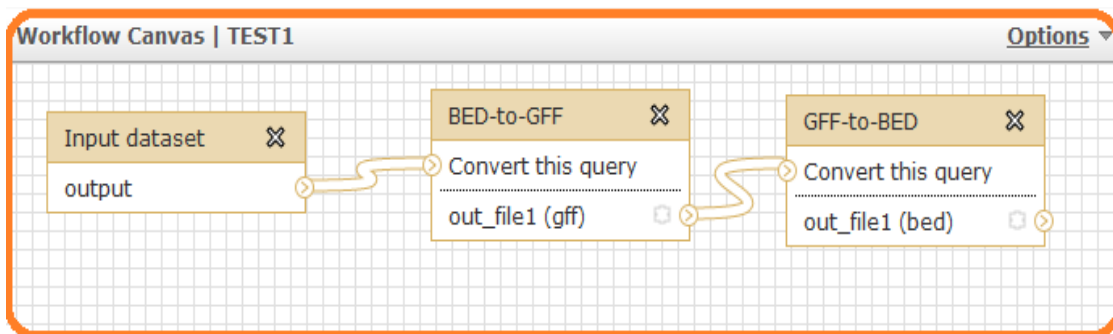


Figure 3.20 - A sample workflow

A list of available workflows appears when the Workflow menu option is selected. Each of the workflows can be edited, run, downloaded, deleted etc.(Fig. 3.21) Or even a new workflow can be created or imported.

Name	# of Steps
imported: bwa-versi	67
imported: imported:	12
TEST1	2
Workflow constructe	2
TEST 2	0
Unnamed workflow	3
test1	0

Figure 3.21 - Options in Workflow panel

3.3.3 BUILDING A WORKFLOW

In order to create a workflow the user must go to the workflow editor and select the option of creating a new workflow. The workflow editor has the tool panel on the left, the same as in the analysis workspace except for the Get Data tools (Fig. 3.22). This group of tools is available only in the analysis workspace.

To insert a tool the user clicks on it from the list of tools. Automatically, the tool appears on the canvas area where the user can place it where he wants. When a tool is selected the interface details of the tool are displayed on the right hand side panel. The interface details are different for every tool. One of the possible actions is to “Rename Dataset”, which renames the output data set. Another useful action is the email notification which sends an email to the user when this step is finished as shown in Figure 3.23.

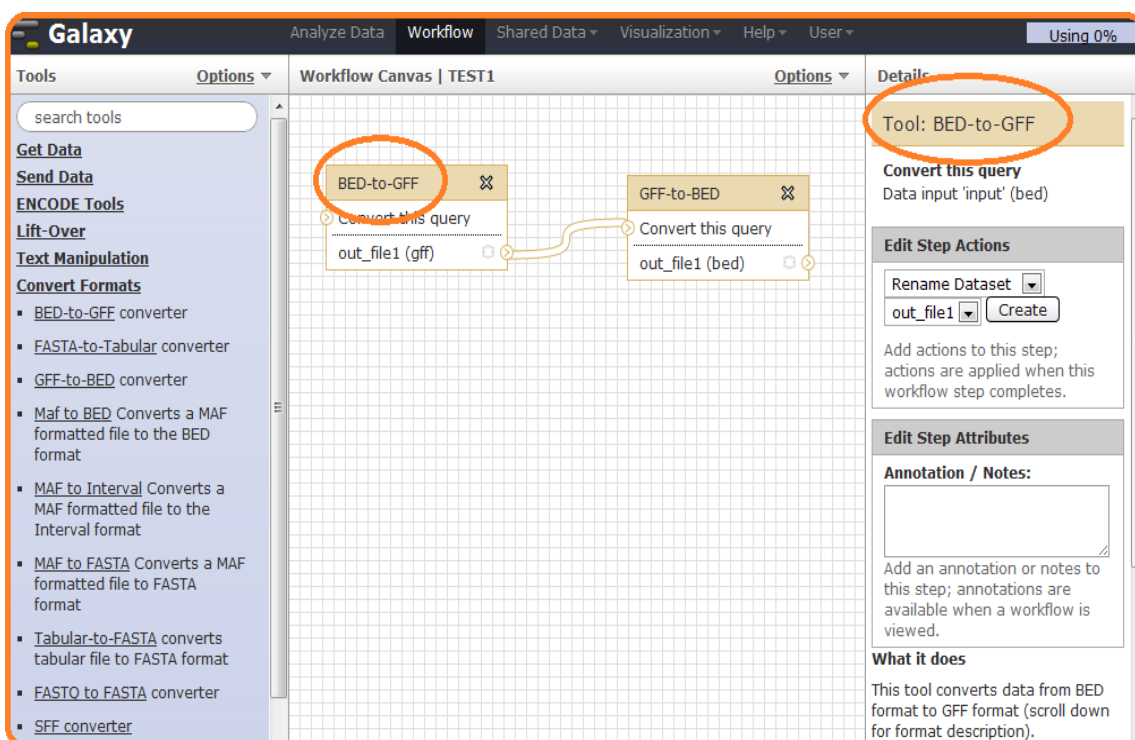


Figure 3.22 - Galaxy’s Workflow editor space

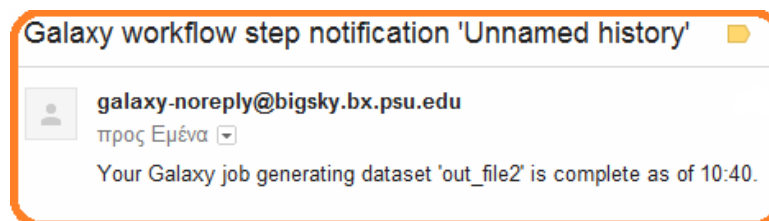


Figure 3.23 - Galaxy’s email notification

User annotations are a property of each tool that helps to indicate the intended purpose of the analysis step, something extremely useful when sharing workflows. To connect two tools, the user simply connects the output of first task to the input of another. Galaxy does not allow connection of incompatible formats. The input to the first tool is set up at run time in the analyze workbench by selecting data from a list of compatible data from the users saved datasets. These can be datasets prepared in previous steps in the analyze workbench or in previous runs of workflows.

Another, way to create a workflow is to extract one from a list of history items. Galaxy automatically creates a workflow by selecting which history items to include and it determines the links based on the input/output data sets. Also, it is possible to import a shared workflow and make the desired changes.

3.3.4 RUNNING A WORKFLOW

Invoking the execution of workflow is done in the workflow editor (Fig. 3.24).

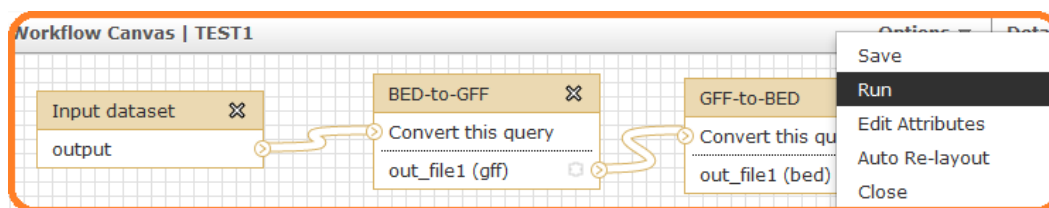


Figure 3.24 – Invoking execution in Galaxy

The execution, however, takes place from within the analysis workspace (Fig. 3.25). Prior to running the workflow, Step 1 which is selecting the input dataset or datasets, must be completed. The input datasets available for selection are the ones in the user's dataset space that have the appropriate format.

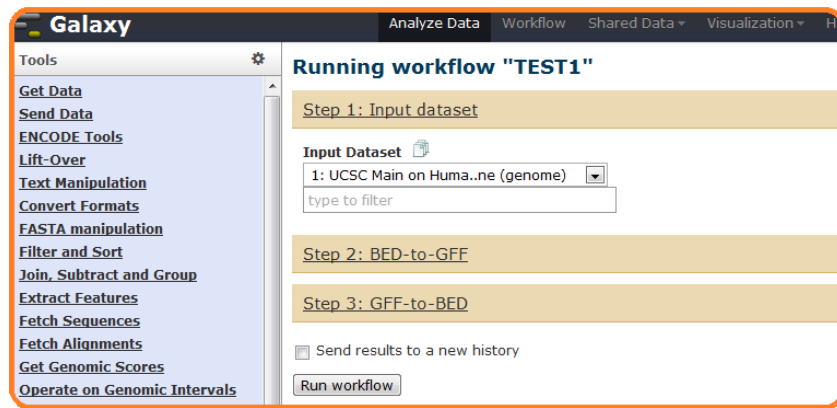


Figure 3.25 - Running a workflow in Galaxy

The History details of each run are shown on the right hand side panel (Fig. 3.26). In example below, the first two tools are executed and the third is currently running.

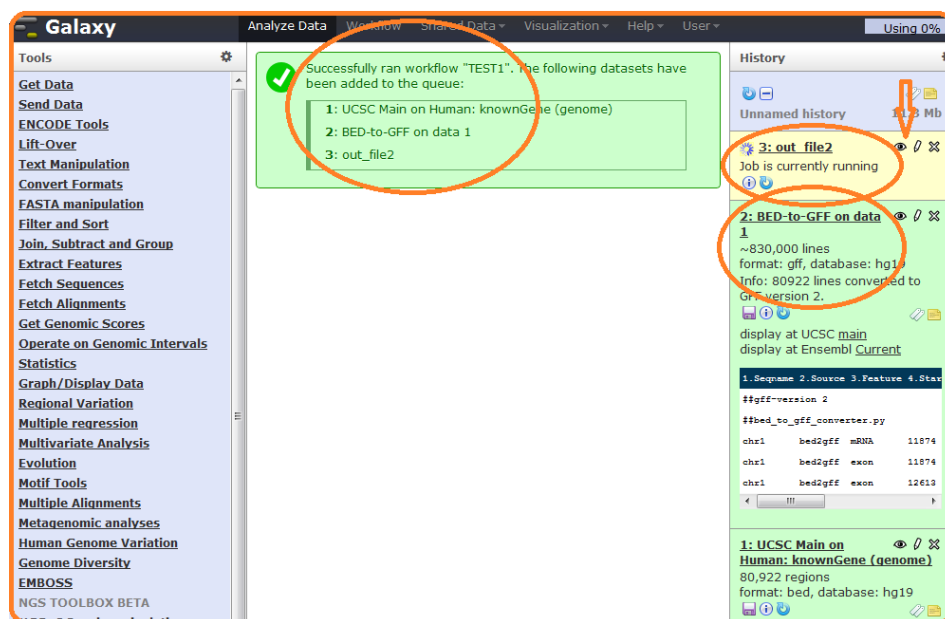


Figure 3.26 – History records in Galaxy

As mentioned in the previous paragraphs, Galaxy automatically creates a history log for each tool of the workflow that executes. The history of each run can be given a name, and retrieved as pleased (Fig. 3.27). Each history contains one or more execution of tools.

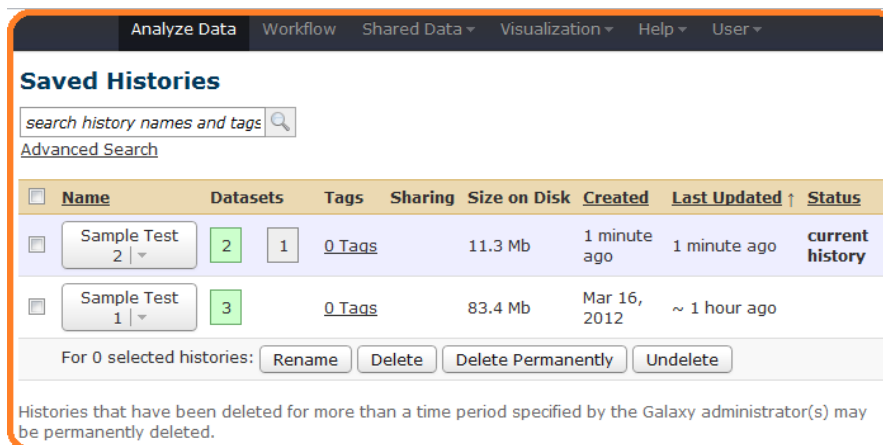


Figure 3.27 - Saved Histories in Galaxy

3.3.5 VIEWING RESULTS

Directly through the history panel or through the User menu the datasets are available for viewing in their text format or in the format created by online genome browsers when this is supported. In Figure 3.28 a text view of the result dataset and in Figure 3.29 a visualization by the online genome browser Ensembl¹⁶.

Figure 3.28 - Text view of dataset provided by Galaxy

¹⁶ <http://www.ensembl.org>

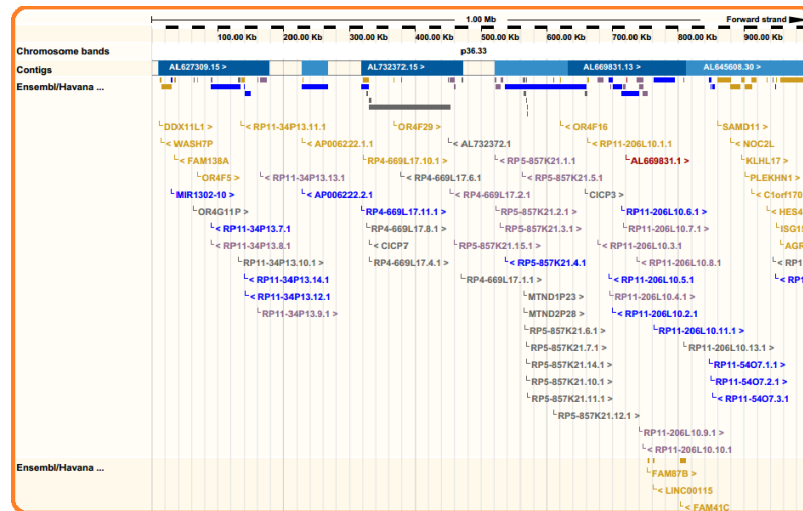


Figure 3.29 - Visualization by the online Genome browser Ensembl

For visualization and visual analysis the user can go to the Visualization menu where more sophisticated visual tools are supported as shown in Figure 3.30 where a chrX is visualized in Galaxy's native Trackster environment. Trackster is not available on the local instances, only on the main server.



Figure 3.30 - Visualization on chrX from Trackster, Galaxy's visualization environment

3.3.6 ARCHITECTURE

Galaxy was designed as a framework for integrating computational tools. Any scripting tool that runs from the command line or has a web interface can be wrapped up to run from inside Galaxy. The tool can be in any programming language other than Python e.g. Perl. The details of each execution are kept and handled as history items along with the datasets used and created. Workflows can be designed for repeating an execution of a series of tools on different datasets. Pages can provide state of the art documentation with live links that can reproduce the analysis. On top of these, collaboration is promoted through the sharing option available for all of the above items.

The Galaxy objects supported by the Galaxy environment are Histories, Workflows, Datasets, Pages and Visualizations. Each of these objects is an accessible named entity within Galaxy.

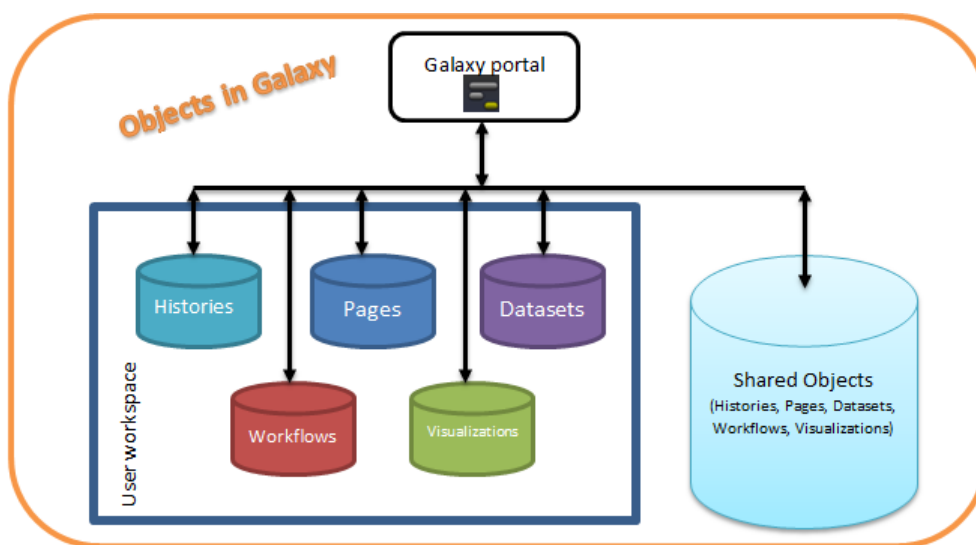


Figure 3.31 – A simplified view of the objects in Galaxy

Provenance information for each workflow and tool executed is automatically generated and stored in history items. A set of history items make up a History. Each History can be given a name, retrieved and re-executed. In this context, a workflow is just a graphical representation of a History. There is no data flow language in Galaxy. The properties of each node and link are simply values in database. That is why there are no control flow operations supported.

Datasets are files that are either uploaded or are the output of a tool. All data sets created by tools are automatically saved and whenever an input dataset to a tool must be selected only the data sets of matching are displayed for selection. Visualizations are graphical representations of datasets. Finally, Pages, as previously mentioned, are online documentation document containing links to other Galaxy objects i.e. Histories, Workflows, Datasets.

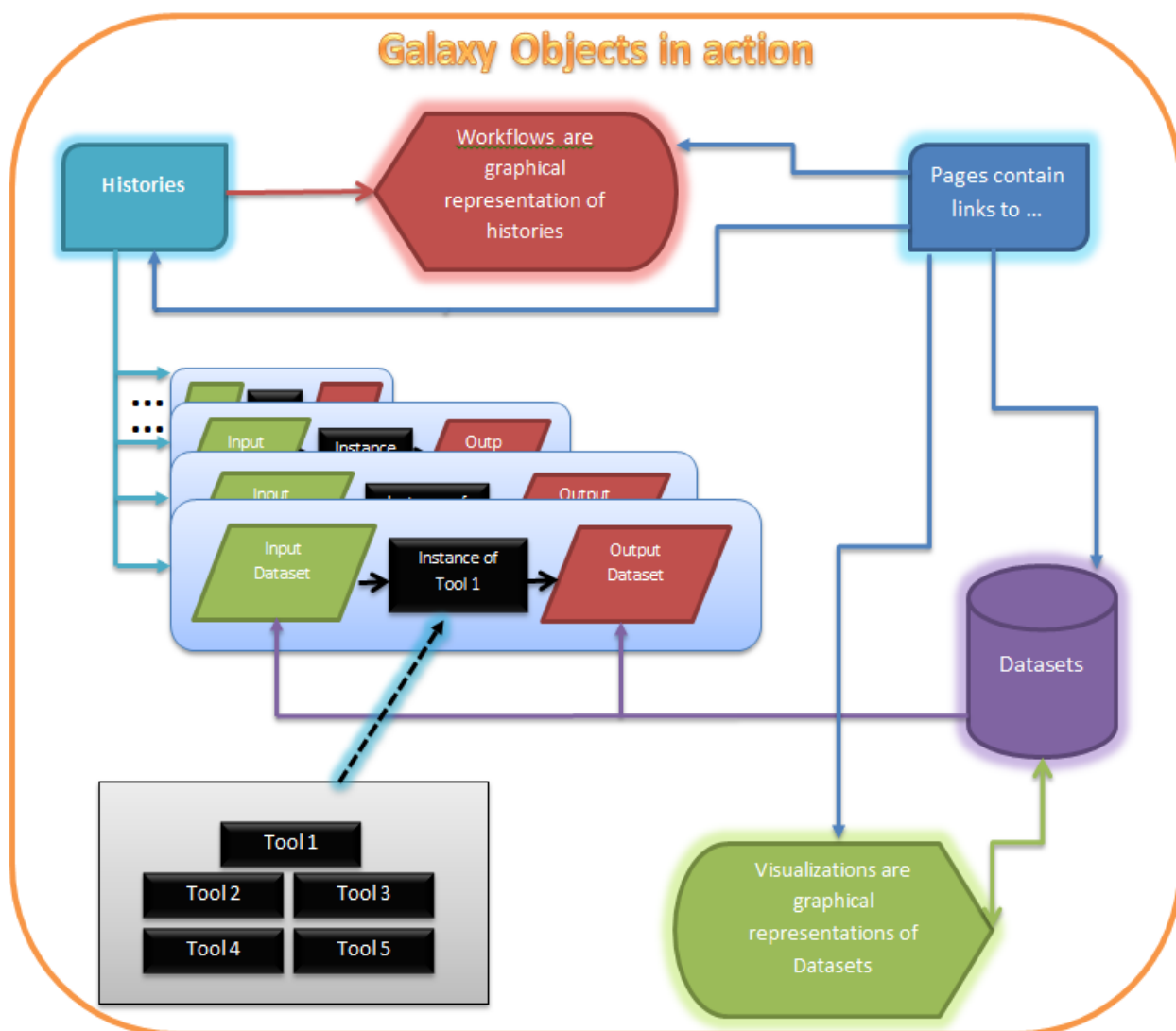


Figure 3.32 – Galaxy objects in action

3.3.7 TAVERNA AND GALAXY INTEROPERABILITY




Taverna workflows can be invoked from the Galaxy environment. They can be downloaded from myExperiment site and embedded in the Galaxy server instance just like all other custom Galaxy tools. Technically speaking, Taverna workflows are translated in Galaxy as Ruby scripts. Running the Ruby script tool initiates a connection to a remote Taverna server, where the workflow is executed. Results are then returned back through to Galaxy via Ruby. One of the reasons for enabling this integration is that Galaxy does not support control flow operations and remote services invocation while Taverna does. [55]

The screenshot displays the Galaxy web interface for the tool 'NCBI Gi to Kegg Pathway Descriptions (version 1.0.0)'. The interface includes a sidebar with a list of tools under the 'Taverna Workflows' category. The main panel shows the tool's configuration options: 'Select source for Gi_numbers' is set to 'Type manually', 'Enter Gi_numbers' contains the values '215422388' and '120407068', and 'Would you also like the raw results as a zip file?' is set to 'No'. An 'Execute' button is visible. Below the form, there is a section 'What it does' and 'Inputs' which lists 'Gi_numbers' as a list of genbank gene identifiers. The 'Outputs' section lists 'gene_descriptions', 'pathway_by_genes', and 'pathway_descriptions'.

Figure 3.33- Sample Taverna workflows embedded as tools
in the Galaxy server instance at <http://galaxy.nbic.nl/galaxy/>

3.4 Summary

A summary of SWMS characteristics can be found on Table 3.3

Table comparison on common SWMS characteristics			
SWMS Characteristics	Galaxy 	Taverna 	KNIME 
Naming of software components	Tool	process	node
Data Flow or Control Flow operations	DF only	both	both
Web portal	Yes	no but possible (done by projects)	yes, for corporate deployment of KNIME
Web server	Yes	yes	yes
Parallelism	Yes	yes intra - process inter - process pipeline	yes node level
HPC scheduling	Yes	separately	separately
Web services	No	yes	yes
Plug in Tools	for genomic research designed by researchers	CDK, R, BIOMOBY	WEKA, R, CDK, JFree Chart
Integration framework	Yes	yes	yes
data representation plots, graphs, images	online genome browsers	yes, in result pane	yes, dedicated nodes
cloud	Yes	possible	n/a
grid	yes, though the instance deployment	possible	yes, for corporate deployment of KNIME
nested workflows	No	yes, named nested workflows	yes, named meta nodes
interoperability	Integrates Taverna workflows as tools	no	Invokes Pipeline pilot web services
command line	no, but it wraps command line scripts into tools	yes	no, only experimental in older version
workflow sharing	yes, portal access myExperiment	yes myExperiment	yes, for corporate deployment of KNIME myExperiment

reproducibility (provenance)	yes, history items workflow, tool level	yes, workflow level	yes, node level
documentation of Tool/Node/Service usage	high level of detail	low level of detail	high level of detail
Chemistry related plug ins	CADD Suite[85]	RDKit[82], CDK[81]	RDKit[82], CDK[81], Erl Wood Cheminformatics[83], Indigo[84]
Community involvement / size (04-12 mail list entries) (aprox. projects running)	Active (188) (20)	Moderate (22) (60)	Moderate (34) (n/a, mostly commercial)

Table 3.3 – Comparing SWMSs

Chapter 4

Problem Description

4.1 Chemoinformatics Background

4.2 Problem Description

4.3 GRANATUM

"A problem well-defined is half-solved." ¹⁷

As discussed in Chapter 1 this thesis has three main objectives. The first is the investigation and review of the SWMS field. The second is the design of a specific scientific workflow implementing a complex *in silico* experiment from the chemoprevention field and its implementation using two of the most promising SWMS. This involves the preparation of nodes/tools for each of the systems, the design, implementation and execution of workflows and the analysis and presentation of the results obtained. The third main objective is to assess progress in the open source SWMS field. This will be done through the evaluation and discussion of the experiences and results obtained from the developed workflows with respect to the features which make SWMS's attractive. This chapter will focus on the description of the experimental problem of this thesis, the presentation of necessary background to the problem and the general design of the solution implemented.

¹⁷ Albert Einstein

4.1 Chemoinformatics background

Chemoinformatics can be defined as the interdisciplinary field that combines elements from chemistry, biology, statistics, mathematics and computer science to solve chemical and biological problems [56]. It implements and employs tools for representing, processing and using chemical information on the computer in order to assist scientists to manage chemical data, explore the chemical space and identify solutions to chemical and biological problems [57]. One of its main applications is in Drug Design where scientists use *in silico* models to expedite the drug discovery process [56].

In chemoinformatics, molecules are represented by a special category of graphs called molecular graphs [58] where nodes and edges represent the atoms and bonds respectively. Graph theory techniques are then successfully applied for the algorithmic analysis of molecular structures (e.g. similarity searching, molecular alignment and superposition, docking). For statistical analysis molecular vectors are typically calculated. These vectors may consist of physicochemical descriptors (e.g. molecular weight, number of hydrogen bond donors, etc) and/or structural descriptors (e.g. presence or absence of specific molecular fragments or other structural features). Once molecules are represented in vector format, statistical methods and machine learning algorithms are used for the analysis of molecular collections, for example, the development of predictive models correlating chemical structure to biological property (e.g. activity). The knowledge extracted can be used for predicting the biological characteristics of new, untested molecules, for selecting promising compounds to test in the lab, for designing new molecules, etc.

Among the methods frequently used is docking, which is based on the presence of detailed knowledge of the protein target structure, and similarity searching using 1D - descriptors (compound properties) and 2D - Binary Descriptors (molecular fingerprints) which is based on the availability of known ligands [59].

Docking is the process by which two molecular structures (e.g. a protein and a small molecule-drug) are *in silico* predicted to bind together. Typically, the method works as follows: it uses the known protein target structure, defines a set of the most likely low energy conformations of the small molecules to test and then attempts to place each small molecule conformation into the protein receptor. Each docking attempt is scored and the value obtained is used as an indication of the experimental binding affinity of the small molecule to the receptor [66].

1-D descriptors calculate whole molecule properties like the molecular weight, molecular surface, number of bond donors, log P and others. Popular rules use such descriptors for the characterization of the molecules. An example is Lipinski's Rule of 5 [60] which uses a subset of these descriptors to identify molecules likely to have poor oral absorption by humans.

In 2-D descriptors, like the so called molecular fingerprints, the presence of structural properties for each position of a molecule is narrated by means of a Boolean bit set to 'one' otherwise to 'zero' [59]. This methodology allows the definition of elaborate molecule bit vectors capturing detailed information on the molecular structure.

Any combination of the above methods, when applied in conjunction with machine learning techniques on a single molecule or a set of molecules, may be used as part of a Virtual Screening (VS) process. VS is the computational analog of biological screening performed in laboratories. It scores, ranks and filters compounds using computational procedures. Its goal is to decrease the number of compounds physically screened by identifying small subsets of large molecular databases that have increased probability to be active against a specific biological target [61]. VS methods are widely used in Drug Design and have great potential for use in new fields such as chemoprevention.

Chemoprevention, is the field of biology that studies the use of chemical substances, either natural or laboratory made, in the prevention of diseases such as diabetes, cancer etc [62]. Cancer Chemoprevention specifically focuses on chemopreventive agents that prevent cancer i.e. block, inhibit or reverse its development [63]. Cancer Chemoprevention is considered one of the most promising areas in current cancer research [63].

4.2 Problem Description

So far a review of scientific workflow tools available for the implementation of *in silico* experiments has been presented. Among them, three (3) representatives from the open source category were selected and discussed in detail. The work described in the remainder of this thesis focuses on the implementation of an *in silico* chemoprevention experiment using the available workflow tools to investigate any weaknesses but more importantly the tool's strong points. The experiences and results obtained will subsequently be used to critically assess the progress of open source SWMS and their ability to be used in production mode by researchers and professionals in a variety of scientific fields.

The experiment chosen comes from the Cancer Chemoprevention field, and is part of the work of the EU FP7 funded GRANATUM project [64]. Details of the GRANATUM project follow in the next section.

4.2.1 Virtual Screening Experiment Description

Virtual screening, as explained in the previous section, is a computerized process that aids in the categorization of chemical compounds. The VS experiment, as designed by GRANATUM team experts, is looking for compounds with cancer chemoprevention characteristics. The successful candidates must have the 3 properties as shown in Figure 4.1.

The candidate molecules will be filtered by 3 VS filters. The first filter, named **OralDruglikeness** filter, filters out molecules that according to the Lipinski's rule[60] have poor oral absorption and therefore are not suitable for administration to patients in the form of a pill. The second filter, designed for **toxicity prediction**, eliminates potentially active/toxic

candidates based on a trained prediction Model. The last filter, named **Binding Affinity Prediction**, scores the molecule's binding affinity through a docking process with an Estrogen Receptor (ER) protein involved in the emergence of breast cancer in women. The result of the experiment will be a list with prioritized compounds according to the filters above.



Figure 4.1 – Properties of a candidate breast cancer chemo preventive compound

4.2.2 Virtual Screening Experiment Workflow

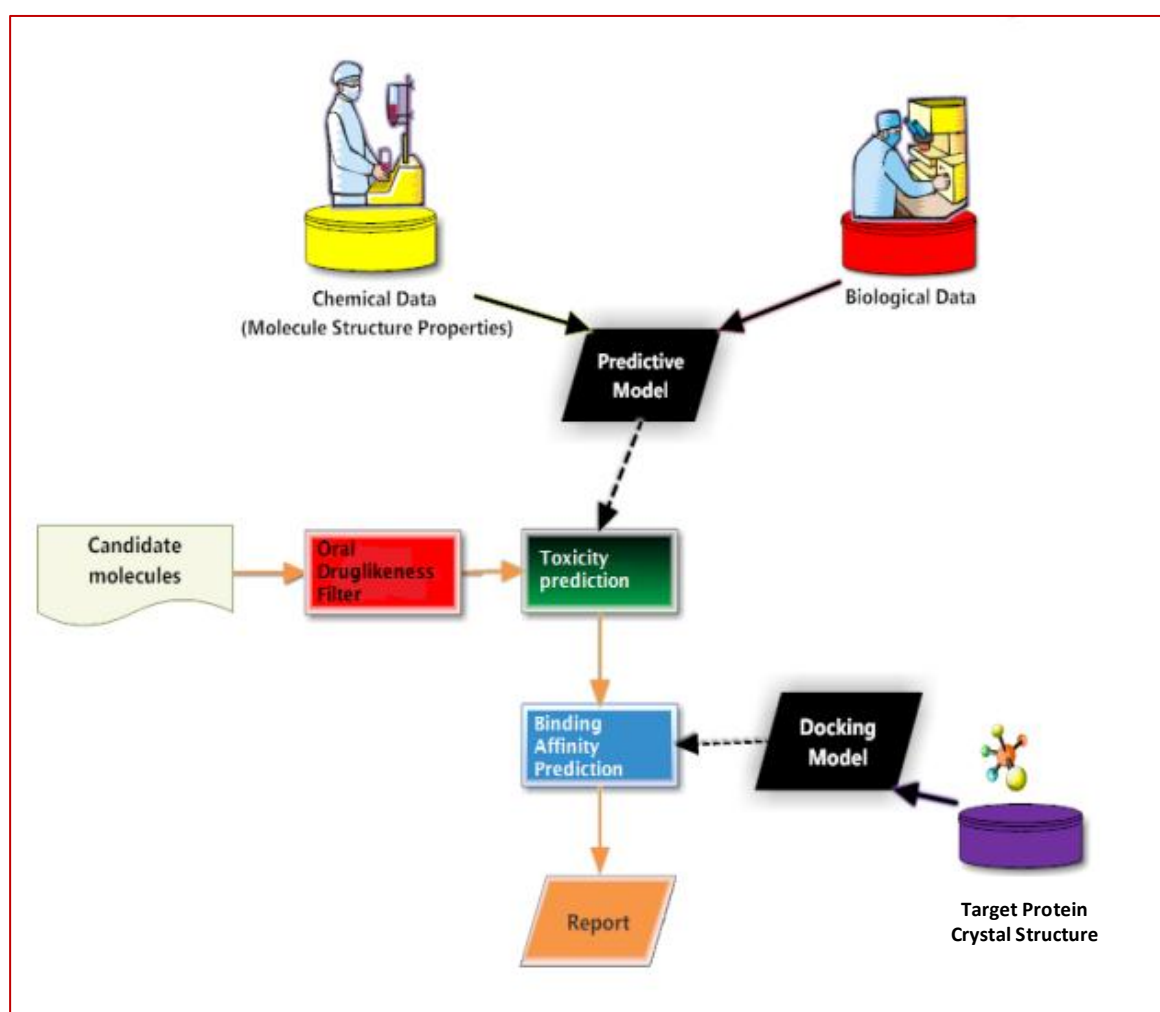


Figure 4.2 - The chemoprevention virtual screening workflow to be implemented

Details of each process in the workflow of Figure 4.2 are given in table 4.1 that follows.

Name of filter	Chemical Method	Description
Oral Druglikeness Filter	1-D descriptors Calculated (Molecular weight, logP, Hydrogen Bonds Donors, Hydrogen Bond Acceptors)	According to Lipinski's rule the ligand must have certain molecular properties. The molecular properties of each molecule are calculated and the successful candidates continue to the next filter. According to Lipinski's rule poor oral absorption or permeation is more likely when: MW > 500, LogP>5, more than 5 H-bond donors (sum of OH and NH groups), more than 10 H-bond acceptors (sum of N and O atoms).
Toxicity Prediction	2-D Descriptors (molecular fingerprints)	Each molecule is characterized by a predictive model either as Toxic (active) or Non-toxic (inactive) based on the molecular fingerprint calculated. Candidates predicted to be toxic are rejected.
Binding Affinity prediction	Docking and Scoring	Each successful candidate is scored by a Docking model for a specific protein involved in the emergence of breast cancer. The likeness of binding of the specific molecule to the target receptor is measured. Molecules are prioritized based on the predicted binding affinity.

Table 4.1 – Details of each process in the VS workflow

The above VS workflow has been implemented within the framework of this thesis using open source scientific workflow systems. Chapter 5 will elaborate on the datasets used, the experimental design and the implementation details. The results obtained are discussed in Chapter 6.

4.3 Granatum



The Granatum Vision as stated in the projects webpage: www.granatum.org

“The vision of the GRANATUM project is to **bridge** the information, knowledge and collaboration gap among biomedical researchers in Europe (at least) ensuring that the

biomedical scientific community has homogenized, integrated access to the globally available information and data resources needed to perform complex cancer chemoprevention experiments and conduct studies on large-scale datasets.”

GRANATUM is a project led by Fraunhofer FIT. The GRANATUM consortium consists of eight partners, from five EU member states, i.e. Ireland, Italy, Germany, Cyprus and Greece including the University of Cyprus.

SWMS are the key element necessary to implement this bridge that will bring together researchers in Europe. As already discussed in Chapter 2, SWMSs offer a number of advantages. They enable sharing and exchange of data, information and methods. They contribute towards the reproducibility of any experiment. Moreover, they can integrate different applications into a single one and provide a common interface and point of access. They also enable scientists without any informatics expertise to easily, transparently use computational resources. Finally, they support alternative processing solutions such as the use of HPC, the Grid and now the Cloud.

Specifically, the GRANATUM project will develop a “Scientific Workflow Management System” for chemoprevention experts to aid in the discovery of new chemical agents with promising chemopreventive characteristics. The system developed will be web-based and will enable scientists to create, update, store and share virtual screening workflows and predictive models. The tool will provide a pool of methods or software components implemented as nodes that may be combined to achieve complex analysis experiments. It will also provide all the necessary data management and processing utilities for reading, cleaning and formatting of data to be used by the processing components as well as writing and storing the analysis results.

Chapter 5

Experimental Design and Implementation

5.1 Experimental design

5.1.1 Data

5.1.2 Overview Diagram

5.2 KNIME implementation

5.3 Galaxy implementation

"Where there is a will, there's a way " ¹⁸

5.1 Experimental Design

This chapter describes the algorithmic implementation carried out and the computational experiments performed for the purpose of this thesis. The VS process was implemented in both the KNIME and Galaxy platforms. An important decision was the selection among a KNIME or a Taverna implementation. Since both use the exact same chemical function libraries the effect was bound to be similar. Subsequently there was no gain in doing both implementations. KNIME was selected over Taverna due to its powerful data visualization tools, friendlier environment and superior documentation and error reporting. It also is the most popular open source SWMS in the chemo informatics field. The advantages of Taverna's online services support were not required for this specific implementation.

Another important consideration in favor of KNIME is that it has integrated 4 different chemical libraries and therefore has a plethora of tools to select from. Moreover it has incorporated Weka[80], a Machine Learning Tool, that gives access to several machine

¹⁸ English proverb http://en.wikiquote.org/wiki/English_proverbs

learning algorithms. Galaxy, on the other hand, was selected with no competitor as it is the only online SWMS available. Galaxy however does not have the tools required for the planned experiment. Its tools are mainly for genetic and general bioinformatics processing. The CADD suite present in the Galaxy platform only has a few of the required chemoinformatics processes. This drawback is in fact irrelevant for the purposes of this thesis, and GRANATUM at large, since our focus is on integrating the GRANATUM tools into Galaxy and to making them available on line.

The data used for the VS process is described in section 5.1.1 whereas in section 5.1.2 a diagram depicting the actual VS process performed is presented. Section 5.2 and 5.3 describe the specific implementations on KNIME and Galaxy respectively.

5.1.1 Data

The experiment uses three datasets as described below in table 5.1.

Details	Dataset A	Dataset B	Dataset C
Purpose	Predictive Model Training	Protein Target Model	VS process testing
Reference	PubChem with AID 464 [94]	PDB database, ID	Indofine data [95]
Contains	706 molecules: 331 active/375 inactive	ER-alpha protein structure (1xpc); co-crystallized with the drug Tamoxifen	2536 molecules
Description	Contains molecular structures labeled as active/toxic or non-active according to laboratory tests. The dataset will be used to train a Toxicity predictive model.	The structure will be used to create a Binding Affinity model for scoring fitness to an ER-receptor	It contains a list of molecules to be passed through the VS process.

Table 5.1 - VS Datasets details

5.1.2 Overview Diagram

The main goal of the experiment is to guide the selection of molecules from the Indofine dataset, a commercially available collection of compounds, for acquisition and biological

screening in the lab. The *in silico* experiment will profile all external compounds and highlight those with predicted favorable characteristics based on the filters/models used. In this manner savings both in time and costs will be achieved as the alternative would require that all Indofine compounds are bought and tested experimentally. The diagram in Figure 5.1 presents the VS experiment performed. It is similar to diagram 4.2 in chapter 4. However, in this case the processes were not used as filters in sequence; rather they were executed in parallel. As a result predictions on all input data are included in the final report and, therefore, more detailed analysis can be performed. The functionality of each VS process is explained in table 5.2.

Process Name	Functionality
Load Compounds	Converts the data from a text file format into internal binary representation of molecules.
Calculate fingerprints	For each molecule a set of binary descriptors (fingerprints) is calculated. Morgan type fingerprints were selected [56]
Predict cytotoxicity	Each of the molecules is passed through a predictive model that predicts a value signifying the molecule's likeness to be cytotoxic.
Calculate Descriptors	For each molecule a set of descriptors is calculated. There is a large set of descriptors to choose from. In our case we are interested in only 4 descriptors values (Molecular weight, logP, Hydrogen Bonds Donors, Hydrogen Bond Acceptors) necessary for the calculation of the Rule-Of-Five.
Oral Drug Likeness Filter	For each molecule the widely used Rule-of-Five also known as the Lipinski rule [60] is applied. The result is a value indicating the number of rules the molecule failed.
Calculating Docking values	Each of the molecules is passed through a model that predicts the binding affinity to an ER receptor [66]
Merge Values	The results of all of the previous steps are summed up in a report. The columns in the report are: molecule, Toxicity (Active/Inactive), Rule-of-five value pass/fail , Docking value. The 4 descriptor values calculated for the purposes of the Rule-of-five are shown next (Molecular weight, logP, Hydrogen Bonds Donors, Hydrogen Bond Acceptors)
Model Calculation	
Predictive model	A Support Vector Machine model[90], build with PubChem AID 464 data.
Docking model	A Binding Affinity scoring model, build for the ER-alpha protein structure (1xpc)

Table 5.2 - VS workflow process functionality

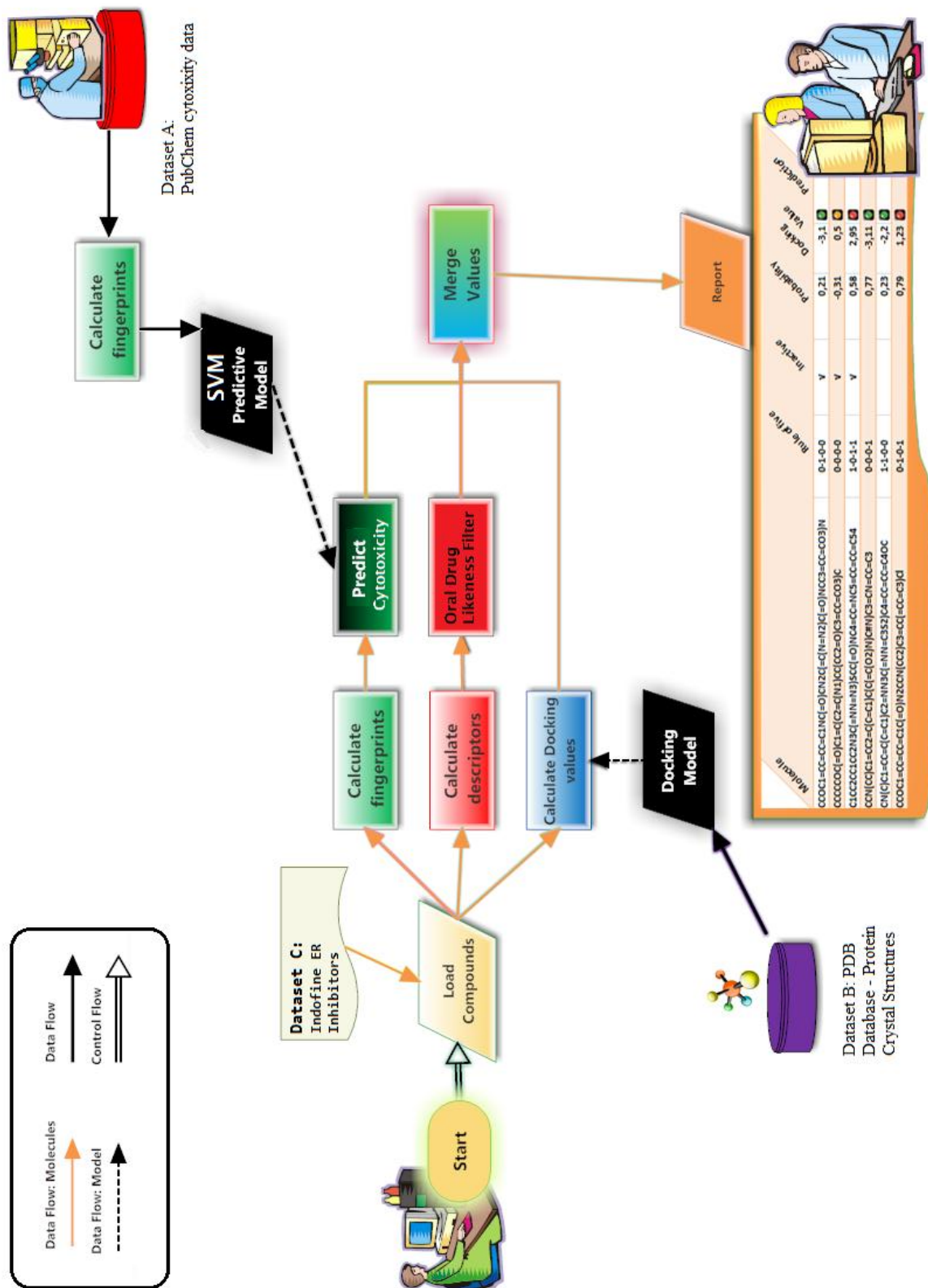


Figure 5.1 - Experimental Design Diagram

5.2 KNIME implementation

The KNIME implementation was performed on a personal desktop computer operating on Windows 7 Home Premium edition with the following hardware characteristics: Intel® Core™ i5-2430M @ 2.40GHz 64Bit, 4GB RAM and 1TB hard drive. The main application used was KNIME 2.5.3. The secondary applications were Python 2.7[78], Weka 3.7.5[80], Java Eclipse 3.6.1[79], RDKit[82], CDK[81]. A short description of the software applications used is given in table 5.3.

<i>Application</i>	<i>Website</i>	<i>Short Description (source Wikipedia)</i>
<i>Python</i>	<i>pyhton.org</i>	<i>Programming Language Python is a general-purpose, high-level programming, cross platform language whose design philosophy emphasizes code readability. The reference implementation of Python (CPython) is free and open source software. Python supports multiple programming paradigms: object-oriented, imperative and functional programming. It features a fully dynamic type system and automatic memory management. Like other dynamic languages, Python is often used as a scripting language.</i>
<i>Weka</i>	<i>www.cs.waikato.ac.nz/ml/weka/</i>	<i>Data Mining Tool The Weka workbench contains a collection of visualization tools and algorithms for data analysis and predictive modeling, together with graphical user interfaces for easy access to this functionality. Weka supports several standard data mining tasks, more specifically, data preprocessing, clustering, classification, regression, visualization, and feature selection. Weka uses the text format Attribute Relationship File Format (ARFF).</i>
<i>Java Eclipse SDK</i>	<i>http://www.eclipse.org/</i>	<i>The Eclipse Platform is a multi-language software development environment comprising an integrated development environment (IDE) and an extensible plug-in system. It is written mostly in Java. By means of various plug-ins, it can be used to develop applications in various programming languages. The Eclipse SDK (which includes the Java development tools) is meant for Java developers. Users can extend its abilities by installing plug-ins written for the Eclipse Platform, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules. Eclipse SDK is a free and open source software.</i>
<i>RDKit</i>	<i>http://www.rdkit.org/</i>	<i>A collection of chemoinformatics and machine-learning software written in C++ and Python.</i>
<i>CDK</i>	<i>http://tech.knime.org/community/cdk</i>	<i>The Chemistry Development Kit (CDK) is a Java library for structural chemo- and bioinformatics. It is an open source and open development chemoinformatics package. The CDK KNIME integration was developed in collaboration with the KNIME group.</i>

Table 5.3 – Short descriptions of secondary applications in the KNIME implementation

Based on the experimental design diagram in Figure 5.1, the experiment can be separated into 4 distinct steps. The first step, STEP1, will build the prediction model to be used later in step 3. Similarly STEP2 will build the docking model. In STEP3, a workflow executing the 2 models and the filter will be constructed and run. During STEP4, the workflow is exported and shared in a global collaboration platform. The same workflow is then imported into another machine for testing.

STEP 1: Prepare the Predictive Model

The predictive model is constructed by the workflow in Figure 5.4. The input file is dataset A, described in section 5.1. Dataset A, shown in Figure 5.2, contains molecule structures and their known toxicity value. For each molecule in the file, its molecular descriptors will be calculated and then used to train the prediction model. When the model is build, it will be saved in a suitable format for input in a prediction type node later in STEP3.

Table "cytotoxicity_aid464.smi" - Rows: 706			
Row ID	SMILES	ID	Outcome
Row0	<chem>CCN(CC)C1=CC2=C(C=C1)C(C(=C(O2)N)C#N)C3=CN=CC=...</chem>	658614	Active
Row1	<chem>CN(C)C1=CC=C(C=C1)C2=NN3C(=NN=C3S2)C4=CC=CC=...</chem>	663191	Active
Row2	<chem>CCOC1=CC=CC=C1NC(=O)CN2C(=C(N=N2)C(=O)NCC3=C...</chem>	3236382	Active
Row3	<chem>CCCCOC(=O)C1=C(C2=C(N1)CC(CC2=O)C3=CC=CO3)C</chem>	5307230	Active
Row4	<chem>CCOC1=CC=CC=C1C(=O)N2CCN(CC2)C3=CC(=CC=C3)Cl</chem>	871395	Active
Row5	<chem>C1CC2CC1CC2N3C(=NN=N3)C4=CC=CC=C4</chem>	3067360	Active

Figure 5.2 - Dataset A containing molecule structures and their toxicity values as displayed by KNIME

The workflow in Figure 5.4, consists of 11 preprocessing nodes, responsible for reading the data and transforming it into the suitable type for insertion into the model builders. Two different models are build, an SVM model and a Bayesian. The actual inputs to the classifiers were 1024 columns of bit strings (the fingerprints) and 1 column with the string Active/Inactive property which indicated the classification classes. KNIME provides a model

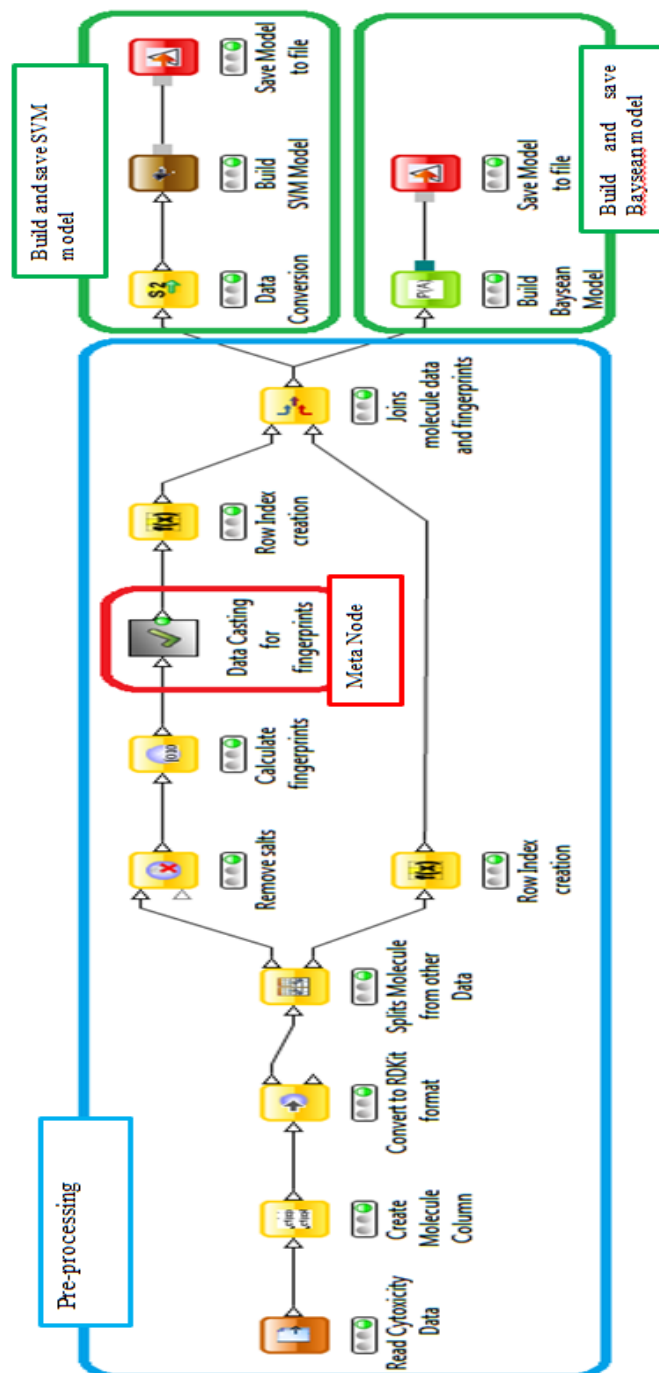


Figure 5.3 – The Prediction Model construction workflow

type format, thus enabling models to be saved. In Figure 5.4 the models prepared and saved are shown. A detailed description of each Node in the workflow follows in table 5.4.

<i>Name</i>	<i>Type /Category</i>	<i>Description</i>
<i>Read cytotoxicity Data</i>	<i>File Reader/Read/ IO</i>	<i>Read the cytotoxicity data provided The data consists on a SMILES's Molecule and a column indicating Active, Inactive toxicity property.</i>
<i>Create Molecule Column</i>	<i>Molecule Type Cast / Translators / Chemistry</i>	<i>Converts the column containing the Molecule string to molecule format</i>
<i>Convert to RDKit</i>	<i>Molecule to RDKit / RDKit / Community Nodes</i>	<i>Generates RDKit molecule column from a string representation (SDF or Smiles). An advantage of this Node is that it removes any Molecules that are incorrect.</i>
<i>Split Molecule from other Data</i>	<i>Splitter/ Column / Data Manipulation</i>	<i>Splits the Molecule column from the rest of the data in order to be processed for fingerprint calculation.</i>
<i>Remove Salts</i>	<i>RDKit Salt Stripper/ RDKit / Community Nodes</i>	<i>This node is used for cleaning the representation of molecules, i.e. removing salts</i>
<i>Calculate Fingerprints</i>	<i>RDKit Fingerprints/ RDKit / Community Nodes</i>	<i>Generates fingerprints for an input RDKit Mol column. Type chosen: Morgan Fingerprints</i>
<i>Data Casting for fingerprints</i>	<i>Meta Node (collection of Nodes)</i>	<i>Currently in KNIME fingerprints are available in bitvector format which cannot be set as input to the Classifiers. This Meta Node takes care of the formatting issues</i>
<i>Add IC50 column</i>	<i>Joiner / Column/ Data Manipulation</i>	<i>Add IC50 column which is needed for classification.</i>
<i>Row index creation</i>	<i>Math Formula / Miscellaneous</i>	<i>A simple math formula to create a row with index values that can be used for joining the data together.</i>
<i>Joining Molecule and Fingerprints</i>	<i>Joiner / Column/ Data Manipulation</i>	<i>Joining the Molecule, its toxicity value and the fingerprints.</i>

Table 5.4 – Detailed description of nodes in the prediction model construction workflow



 Bayesian.zip	14/4/2012 7:26 μμ	WinRAR ZIP archive	41 KB
 SMO.zip	17/4/2012 1:37 μμ	WinRAR ZIP archive	200 KB

Figure 5.4 - Models build and saved by KNIME

An important decision is selecting the suitable classification algorithm for the experiment.

KNIME has a number of build-in suitable classifiers but it also has integrated some of the

methods provided by WEKA. However, classifiers incorporated into Knime are essentially using black boxes to the user. Therefore, in order to access all the capabilities of a data mining dedicated software, I changed workbench. In the WEKA [86] workbench one can easily test different classification algorithms, adjust their parameters but also perform preprocessing functions that are valuable to classification. Also, the results of the classifier are available in a more comprehensible way.

Testing in WEKA

A. Selecting the Algorithms to evaluate

As the number of combinations of different classifiers and parameters is almost infinite, I selected algorithms as recommended in the literature [56].

B. Data Mining Basic concepts necessary for evaluating training results

The confusion matrix

The following table shows the confusion matrix [87] for a two class classifier.

CLASS	A	B
A	TN	FN
B	FP	TP

Table 5.5 – Confusion matrix

A confusion matrix contains information about actual and predicted classifications done by a classification system. The following definitions can be used to clarify any attempt to evaluate the results in the case of classifiers built for the purposes of our VS experiment.

True Negative: An *Inactive* molecule that was correctly classified as *Inactive*.

True Positive: An *Active* molecule that was detected successfully.

False Negative: An *Active* molecule that was not detected successfully.

False Positive: An *Inactive* molecule that was wrongly classified as *Active*.

The performance of each classifier is evaluated using the F-measure metric, using the data described in the previous table.

$$\mathbf{F\text{-measure}} = \frac{2 * TP}{(2 * TP + FN + FP)}$$

The **F measure** is a balanced average of the Precision and the Recall where the optimum score is 1 and the lowest is 0. It gives a measure of the correctness of the classifier.

For an Active filter to be successful it must classify *Active* molecules with high accuracy, i.e. the FN value must be as low as possible. This is measured using recall or sensitivity.

$$\mathbf{Recall = Sensitivity} = \frac{TP}{(TP + FN)}$$

If FN=0 then Recall=1 (All *Active* molecules are identified and classified as *Active*)

On the other hand, it must not classify *Inactive* molecules as *Active*, so FP must be as low as possible.

This is measured in precision.

$$\mathbf{Precision} = \frac{TP}{P} = \frac{TP}{(TP + FP)}$$

If FP=0 then Precision=1 (No *Inactive* molecule is classified as *Active*)

C. Validation Techniques

All the experiments have been carried out using a stratified 10-fold cross-validation [88], a technique for estimating the performance of a predictive model, in order to increase the confidence level of results obtained. Research has shown that $k = 10$ is a satisfactory total (Breiman & Spector 1992) and (Kohavi1995).

It must be stressed that there is asymmetry in misclassification costs.

In our case, **Recall** has a higher negative cost as it is not desirable for *Active* molecules to be classified as *Inactive*, but it's no harm done if some *Inactive* molecules are misclassified as *Actives* since they will be detected as such in later stages of the process during experimental validation

D. Learning Methods

Weka implementation / Reference	Description of Learning Methods
Naive Bayes	<p>A Naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions.</p> <p>In simple terms, a naive Bayes classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature. Even if these features depend on each other or upon the existence of the other features, a naive Bayes classifier considers all of these properties to independently contribute to any other probability.</p>
	<p>George H. John, Pat Langley: Estimating Continuous Distributions in Bayesian Classifiers. In: Eleventh Conference on Uncertainty in Artificial Intelligence, San Mateo, 338-345, 1995.</p>
k-NN WEKA : IbK	<p>The k-Nearest-Neighbour algorithm (k-NN) is an 'instance-based' machine learning method [89]. This approach does not seek to develop a model that describes the structure of the underlying data; rather, all training examples are stored, and test instances are classified by estimating their similarity to the stored examples. The instances are assigned the majority class of the k closest instances. The underlying inductive bias of the algorithm assumes instances that are close in the attribute space are of the same class.</p>
	<p>D. Aha, D. Kibler (1991). Instance-based learning algorithms. <i>Machine Learning</i>. 6:37-66.</p>
Support Vector Machine WEKA: SMO	<p>Support vector machines (SVMs) map training instances into a higher dimensional feature space by some nonlinear function, and then calculate the optimal hyperplane which maximizes the margin between the data points in the positive class and the data points in the negative class [90]. The Sequential Minimal Optimization (SMO) algorithm implemented in WEKA was used. SMO is a fast method of training SVMs. It breaks a large quadratic programming problem down into a series of the smallest possible sub-problems, minimizing processor and memory demands. The inductive bias of the algorithm is largely dependent on the kernel adopted; different kernels will lead to different models.</p> <p>Implements John Platt's sequential minimal optimization algorithm for training a support vector classifier.</p>
	<p>J. Platt: Fast Training of Support Vector Machines using Sequential Minimal Optimization. In B. Schoelkopf and C. Burges and A. Smola, editors, <i>Advances in Kernel Methods - Support Vector Learning</i>, 1998.</p> <p>S.S. Keerthi, S.K. Shevade, C. Bhattacharyya, K.R.K. Murthy (2001). Improvements to Platt's SMO Algorithm for SVM Classifier Design. <i>Neural Computation</i>. 13(3):637-649.</p> <p>Trevor Hastie, Robert Tibshirani: <i>Classification by Pairwise Coupling</i>. In: <i>Advances in Neural Information Processing Systems</i>, 1998.</p>
Decision Tree C4.5 - WEKA: J48	<p>Ensembles of classifiers can often perform better than any individual classifier.</p> <p>Boosting classifiers, manipulates training examples to generate a set of hypotheses. Instead of dividing the training set, it attaches weights to each of the training instances, and on each iteration attempts to minimize the weighted error on the training set. Misclassified instances have their weight increased and correctly classified instances have their weight decreased.</p> <p>AdaBoostM1 method was used with a decision tree as a base classifier.</p> <p>Decision trees use tests on one or more attributes to classify a particular instance. A typical tree has several internal nodes, which represent tests, and several child nodes, which represent all potential classification outcomes; the tree can effectively be described as a series of if-else rules.</p> <p>J48 is an open source Java implementation of the C4.5 algorithm in the weka data mining tool.</p>
	<p>Yoav Freund, Robert E. Schapire: Experiments with a new boosting algorithm. In: Thirteenth International Conference on Machine Learning, San Francisco, 148-156, 1996.</p> <p>Ross Quinlan (1993). <i>C4.5: Programs for Machine Learning</i>. Morgan Kaufmann Publishers, San Mateo, CA.</p>

Table 5.6 – Learning Methods used

E. Start mining

In the table that follows, the testing results confirm that:

- the best 2 options for the Prediction Model are SMO and Naive Bayes
- that attribute selection must be applied as it improves the results by a factor of 10%

Testing Possible Prediction models on WEKA tool												
Test File												
Instances	706											
Class: Active	331											
Class: Inactive	375											
Training Results stratified 10 fold cross validation												
Algorithm	Attribute selection	Precision			Recall			Correctly Classified Instances		Incorrectly Classified Instances		F-Measure
		Active	Inactive	Average	Active	Inactive	Average					
Naive Bayes	yes	0,708	0,735	0,722	0,695	0,747	0,722	510	72%	196	28%	0,722
	no	0,602	0,638	0,621	0,571	0,667	0,662	439	62%	267	38%	0,621
Support Vector Machine	yes	0,725	0,735	0,731	0,686	0,771	0,731	516	73%	190	27%	0,730
	no	0,628	0,672	0,652	0,628	0,672	0,652	460	65%	246	35%	0,652
k-NN	yes	0,614	0,697	0,658	0,698	0,613	0,653	461	65%	245	35%	0,653
	no	0,588	0,696	0,645	0,728	0,549	0,633	447	63%	259	27%	0,643
Boosting	yes	0,662	0,713	0,689	0,686	0,691	0,688	486	69%	220	31%	0,689
	no	0,555	0,614	0,586	0,580	0,589	0,585	413	58%	293	42%	0,585
Decision Tree	yes	0,626	0,660	0,644	0,601	0,683	0,644	455	64%	251	36%	0,613
	no	0,492	0,557	0,527	0,556	0,493	0,523	369	52%	337	47%	0,523

Table 5.7 - Results of testing prediction models depicting F-measure

STEP 2: Prepare the Binding Affinity Model

At the time this experiment was implemented no open source docking model was available to be used in a KNIME workflow. It was therefore decided to run the docking model independently and feed its results into the final report. The docking model run is explained in detail in the Galaxy implementation as it was developed by the GRANATUM team.

STEP 3: Implement the VS experiment workflow

The main workflow of the experiment has 3 input nodes. The main input node reads the file to be screened which is dataset C shown in Figure 5.6. The second input node reads the toxicity prediction model constructed in STEP1 and the third the results of the docking model constructed in STEP2. As in the case of the prediction model workflow, a number of

preprocessing nodes need to be run before the input data (molecules) is put in the right format for filtering through the predictive model. Upon completion of the filtering through the predictive model a new column I appended. This column is labeled winner and can have only two possible values: Active meaning toxic and Inactive meaning non-toxic.

The next nodes to be executed are the ones implementing the oral drug likeness filter. Here the molecules change type as a different software component will be used from the CDK group. Executing the Lipinski's filter node appends a new numeric column that counts the failures of the filter. Number 0 indicates a molecule with 0 failures. The actual values of the 4 descriptors are also calculated by a separate node to be included in the final report as reference. Reading the results from the docking score file requires executing data retrieval and data transformation nodes as the data are is customized for use by KNIME nodes. At the final steps of the workflow the results are merged in the final report. As noted in the experimental design, the nodes are run in parallel and not as sequential filters. As a result the final report contains all the molecules. The entire workflow implemented is shown in Figure 5.5

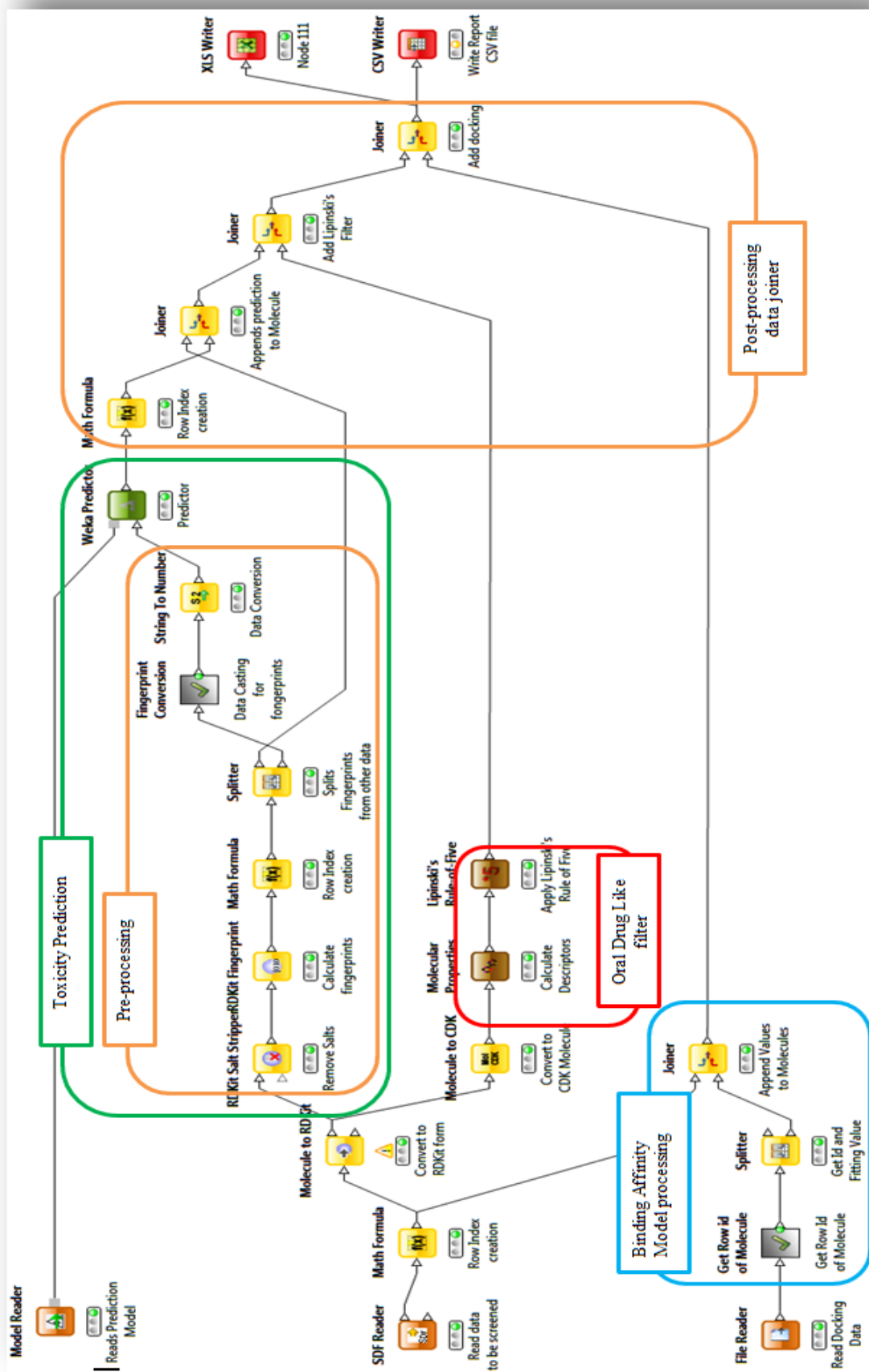


Figure 5.5 - The VS workflow as displayed in KNIME

Table "default" - Rows: 2536 Spec - Column: 1 Properties | Fil

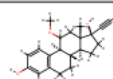
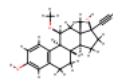
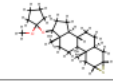
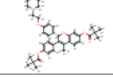
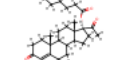
Row ID	sdf Molecule
Row0	
Row1	
Row2	
Row3	
Row4	

Figure 5.6 - Input File as displayed by KNIME

The final report is constructed by merging 3 KNIME tables. The columns selected are the molecule, the Winner column indicating the prediction Active/ Inactive, each of the four descriptors of the Lipinski's rule, a column indicating the number of filters they fail and finally the predicted binding affinity score. The output table constructed is shown in Figure 5.7. The report is written out in a text file (tab delimited) by an output node to match the Galaxy implementation and in an xls format for Excel for easy processing.






sdf Molecule	S Winner	D Mannhold LogP	I Hydrogen Bond Acceptors	I Hydrogen Bond Donors	D Molecular Weight	I Lipinski's Rule of Five	D fitness
	Active	3.44	2	2	326.188	0	-10.426
	Active	3.44	2	2	326.188	0	-10.722
	Active	3.88	2	0	404.275	1	-14.732
	Active	4.98	3	0	625.34	1	-14.811
	Active	3.99	4	0	428.293	1	-16.116

Figure 5.7 - Output File as displayed by KNIME

A detailed description of each Node of the VS workflow in Figure 5.5 follows in table 5.8.

<i>Name</i>	<i>Type /Category</i>	<i>Description</i>
<i>Read input data for screening</i>	<i>SDF Reader / IO / Chemistry</i>	<i>Reads input file to be screened which is in SDF file form.</i>
<i>Row Index Creation</i>	<i>Math Formula / Miscellaneous</i>	<i>Creates a unique row index to be used for joining the docking data with the Molecule</i>
<i>Convert to RDKit</i>	<i>Molecule to RDKit / RDKit / Community Nodes</i>	<i>Generates RDKit molecule column from a string representation (SDF or Smiles) It also removes from the list any non valid molecules</i>
<i>Remove Salts</i>	<i>RDKit Salt Stripper/ RDKit / Community Nodes</i>	<i>This node is used for removing salts from RDKit molecules.</i>
<i>Read Prediction Model</i>	<i>Model Reader / Read / IO</i>	<i>Reads the model created in the previous step from file.</i>
<i>Read Docking Data</i>	<i>File Reader/ Read/ IO</i>	<i>For now, the data is calculated outside KNIME and are fed into the report</i>
<i>Calculate Fingerprints</i>	<i>RDKit Fingerprints/ RDKit / Community Nodes</i>	<i>Generates fingerprints for an input RDKit Mol column. Type chosen: Morgan</i>
<i>Row Index Creation</i>	<i>Math Formula / Miscellaneous</i>	<i>Creates a unique row index to be used for joining the modeling data back with the Molecule</i>
<i>Splits fingerprint from other Data</i>	<i>Splitter / Column/ Data Manipulation</i>	<i>Splits fingerprint from other data in order to be fed into the Predictor.</i>
<i>Datacasting for fingerprints</i>	<i>Meta node (a collection of nodes)</i>	<i>As defined above. Converts the fingerprint bitvector into 1 bit strings.</i>
<i>Data conversion</i>	<i>String to Number</i>	<i>It converts the bits to numbers in order to be accepted by the SVM prediction model.</i>
<i>Predictor</i>	<i>Weka prediction /</i>	<i>Takes as input a previously saved Model and performs a classification. It appends a new column Winner on the data</i>
<i>Row Index creation</i>	<i>Math Formula / Miscellaneous</i>	<i>Creates a unique row index to be used for joining the modeling data back with the Molecule</i>
<i>Append prediction to Molecule</i>	<i>Joiner / Column/ Data Manipulation</i>	<i>Joins the Molecule with its prediction value as returned by the Model</i>
<i>Convert to CDK Molecule</i>	<i>Molecule to CDK</i>	<i>The Lipinski's Rule of five is implemented in the CDK module. So we must convert the molecules to CDK format.</i>
<i>Calculate Descriptors</i>	<i>Molecular properties /</i>	<i>Calculates descriptor for the molecule, of which five are chosen: lopP, Molecular weight, Hydrogen Bond Acceptors, Hydrogen Bond Donors</i>
<i>Apply Lipinski's Rule of five</i>	<i>Lipinski's Rule of 5 / CDK / Community Nodes</i>	<i>Applies Lipinski's Rule of 5 and adds one additional column containing the count of failures of the rule of five</i>
<i>Get row id of Molecule</i>	<i>MetaNode (a collection of Nodes)</i>	<i>A series of transformations are required to get the id of the Molecule in order to be matched for the report</i>
<i>Get Id and Fitting Value</i>	<i>Splitter / Column/ Data Manipulation</i>	<i>The required columns are split from the rest of the data</i>
<i>Append value to molecules</i>	<i>Joiner / Column/ Data Manipulation</i>	<i>This node joins the Molecule with its docking value</i>
<i>Add Lipinski's Filter</i>	<i>Joiner / Column/ Data Manipulation</i>	<i>Adds Lipinski's Filter and descriptors to final output. Join based on Molecule</i>
<i>Add Docking</i>	<i>Joiner / Column/ Data Manipulation</i>	<i>Adds docking value to final output. Join based on Molecule.</i>
<i>Write csv file</i>	<i>CSV Writer / Write / IO</i>	<i>Write the data into a csv type file.</i>

Table 5.8 – Detailed Description of each node in the workflow of Figure 5.5.

	A	B	C	D
1	<i>Molecule</i>	<i>Lipinski's Rule of Five</i>	<i>Winner</i>	<i>fitness</i>
2	-ISIS- 09231109272D 10 9 0 0	0	Inactive	-27,06417919
3	-ISIS- 06211114542D 20 22 0 0	0	Inactive	-24,65659327
4	-ISIS- 06211114532D 21 23 0 0	0	Inactive	-24,43903469
5	-ISIS- 06211114542D 21 23 0 0	0	Inactive	-24,35563023
6	-ISIS- 06211114542D 21 23 0 0	0	Inactive	-24,23495013
7	-ISIS- 06211114532D 25 28 0 0	0	Inactive	-23,43293668
8	-ISIS- 06211114532D 21 23 0 0	0	Inactive	-23,12999213
9	-ISIS- 06211114542D 21 22 0 0	0	Inactive	-22,88401723
10	-ISIS- 08101114552D 15 15 0 0	0	Inactive	-22,57762518
11	-ISIS- 08101114552D 10 10 0 0	0	Inactive	-22,57162639
12	-ISIS- 06211114542D 15 16 0 0	0	Inactive	-22,36395694
13	-ISIS- 06211114542D 24 25 0 0	0	Inactive	-22,29732525
14	-ISIS- 06211114532D 15 17 0 0	0	Inactive	-21,55068388
15	-ISIS- 06211114532D 21 23 0 0	0	Inactive	-21,17642107

Figure 5.8 - Experiment's Report in a spreadsheet sorted as pleased

This output can now easily be sorted by users/ domain scientists using a spreadsheet application to select the top candidate molecules indicated by the VS experiment (Fig. 5.8).

STEP 3: Sharing & Collaboration

Saving a KNIME workflow automatically saves provenance data which in KNIME are saved for each individual node separately. Using the Export workflow option, the workflow is exported in a zip format. The file was then uploaded in myExperiment (Fig. 5.19), along with its data. It was later downloaded on another workstation and imported in KNIME for further processing and demonstration.

5.3 Galaxy implementation

The Galaxy workflow was designed and executed on a Galaxy server instance running Ubuntu Desktop 12.04 LTS (Linux version) with the following hardware characteristics: Intel® Pentium® 4 CPU 3.20GHz (1 core - HT enabled 32bit), 1GB ram and 80GB hard disk. The additional applications required were Python 2.7, SciKit-Learn 0.10, RDKit 2011_12_1, Chil2 GlamDock 0.5. In table 5.9 a short description of the additional applications is given.

<i>Application</i>	<i>Website</i>	<i>Short Description (source Wikipedia)</i>
<i>Python</i>	<i>pyhton.org</i>	<i>Programming Language Python is a general-purpose, high-level programming, cross platform language whose design philosophy emphasizes code readability. The reference implementation of Python (CPython) is free and open source software. Python supports multiple programming paradigms: object-oriented, imperative and functional programming. It features a fully dynamic type system and automatic memory management. Like other dynamic languages, Python is often used as a scripting language.</i>
<i>RDKit</i>	<i>www.rdkit.org/</i>	<i>A collection of chemoinformatics and machine-learning software written in C++ and Python.</i>
<i>Scikit-learn</i>	<i>scikit-learn.org/</i>	<i>A Python open-source module integrating classic machine learning algorithms.</i>
<i>Chil2 GlamDock</i>	<i>www.chil2.de/Glamdock.html</i>	<i>The Chil2 platform consists of components used for molecular modeling and screening focusing mainly on structural- and ligand based docking.</i>

Table 5.9 – Short descriptions of additional applications for the Galaxy workflow implementation

Three steps were required in order to perform the VS experiment in Galaxy. Firstly, new tools had to be prepared and inserted in the platform. Secondly the workflow had to be created and then executed. The third and final step was to perform workflow sharing.

STEP 1: Integrate the necessary tools

The GRANATUM team has implemented command line python modules. The task was to create new tools in the Galaxy platform that would call these command line scripts. Creating tools in Galaxy requires the use of xml wrappers around functionalities and their insertion into the Galaxy tool collection. An xml file defines the tools interface which in this implementation calls a python script. Another xml handles the tool's listing in the Tools

panel. Figure 5.9 displays menu configuration of the new set of tools in the Galaxy's Tools panel while the tools required for the experiment are listed in table 5.10.

Tool's List	Description	Granatum Module
	Input / Output	
Load Compounds	Input: molecule file in sdf form Output: molecule file in binary form Description: Creates a binary molecule file from the input file	load_compounds.py
Merge Report	Input: 3 different binary molecule files Output: a tab delimited text file Description: Merges 3 files into a single file with columns as follows: Molecule, ID, Oral_Drug_Like_Result, Oral_Drug_Like_Result_Reason, Prediction, Margin_Distance, Docking_Score	getPrediction.py
	Filters	
OralDruglikeness Filter	Input: Molecule binary file Output: Molecule binary file with appended the filter's result values Description: Calculates the molecular descriptors and appends two new columns. Oral_Drug_Like_Result and Oral_Drug_Like_Result_Reason	oralDrugLikeFilter.py
Clean Molecules	Input: Molecule binary file Output: Molecule binary file Description: Cleans molecules by removing salts.	cleanMols.py
	Descriptors Calculation	
Morgan Fingerprints	Input: Molecule binary file Output: Molecule binary file with fingerprints values Description: Appends new columns for the fingerprints	calcMorganDesc.py
	Models	
SVM Model	Input: Molecule binary file Output: Molecule binary file with prediction values Description: Executes the SVM model on the input file and appends two new columns: Prediction and Margin_Distance	convDataPM.py main_predictSVM.py

Table 5.10 – Detailed Description of each Tool of Figure 5.11

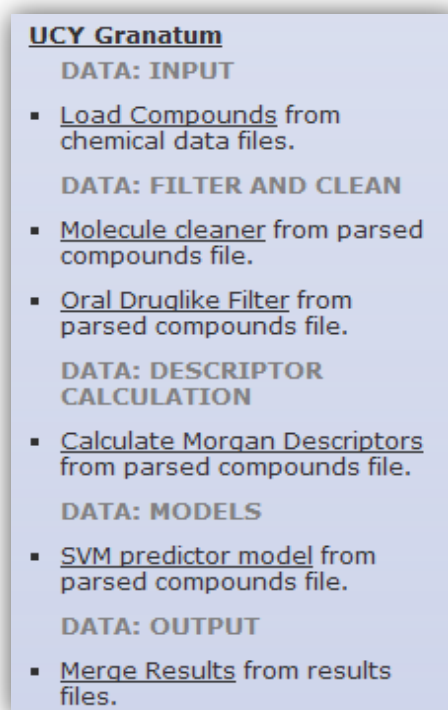


Figure 5.9 – New tools in the Galaxy’s Tool panel

New Definition types

New data types (table 5.11) were defined in order to enable the use of the output of a set of tools as input into another set of tools. This was very important for avoiding unnecessary mistakes in the tools interface by selecting inappropriate datasets. The restrictions as enforced are shown in Figure 5.10.

New definitions types for GRANATUM Tools	
gpkl	binary python data dictionary (g –GRANATUM, pkl - from python data dictionary)
gpklm	as above with appended columns binary Morgan fingerprints (m – morgan)
gpklp	as above with appended column the prediction of the model (p-prediction)
gpklf	as above with appended columns the results of the Oral Drug like filter’s execution (f-filter)

Table 5.11 – New Definition types in Galaxy

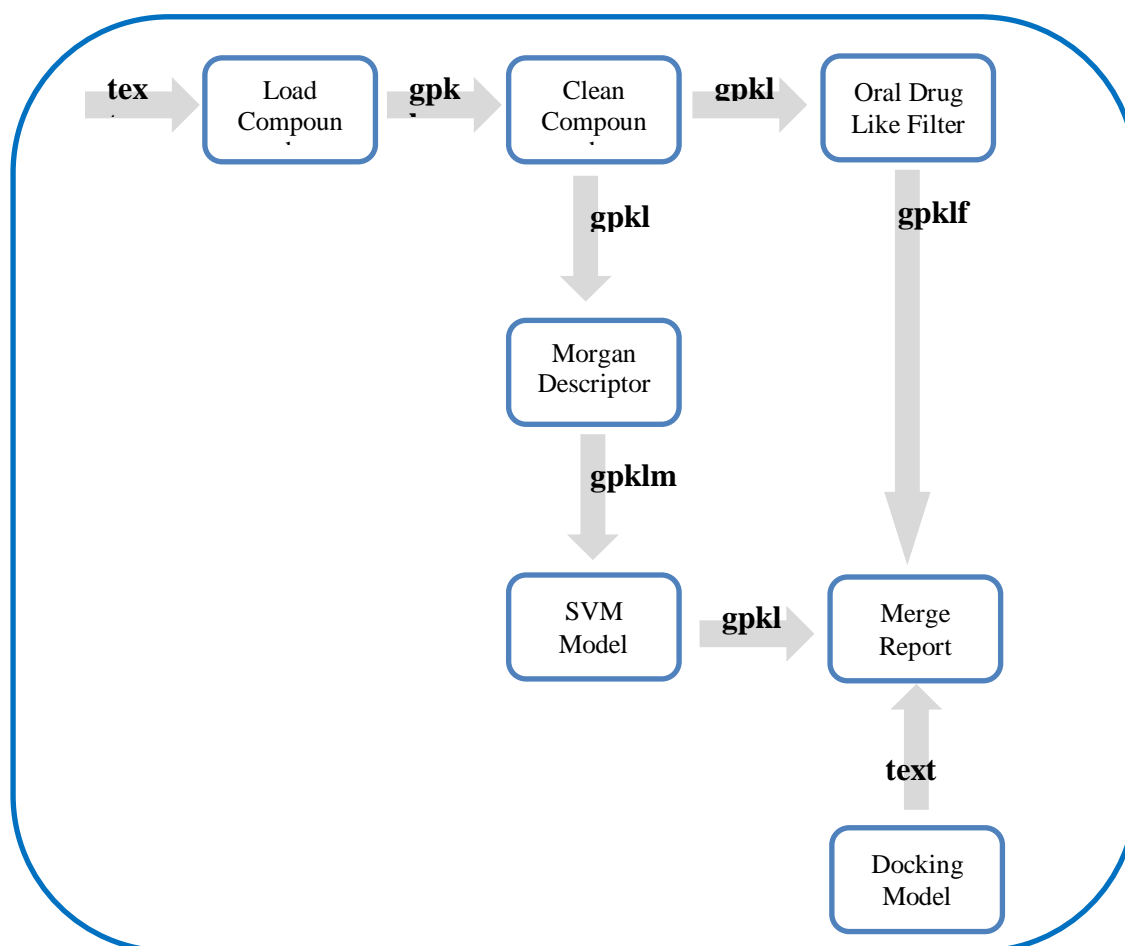


Figure 5.10 – Data types and restrictions enforced among the tools defined

Important Implementation Notes

1. Because the GRANATUM modules were designed to be used as command line scripts, they could not be called directly. Firstly, to conform to Galaxy requirements they needed to have at least one input file and at least one output file for input to the next tool. Therefore the first wrapper reads the files created by the program and feeds them into the output. A second wrapper was required to catch the “stdout” and “stderr” messages produced by the scripts. This second wrapper was provided by the Galaxy community and amended appropriately to accommodate the needs of the present work.
2. Unlike the case of KNIME, where the SWMS was used to build the predictive model, the functionality of creating such model was intentionally not implemented. The model was build outside Galaxy and used for prediction as required. The decision was based on GRANATUM specifications which note that ordinary users will not be allowed to create models. The SVM

prediction Model was prepared using DATASET A, described in section 5.1. The Nu-Support Vector Classification (Nu-SVC) was selected to build the model using 10-fold cross validation using the Scikit-learn[96] module.

3. The Docking scores, similarly to the case of the KNIME implementation, were calculated outside of Galaxy and fed into the final report.(Add info for docking program)

STEP 2: Design and Execute the VS workflow

The implemented Galaxy VS workflow is shown in Figure 5.11. The experiment consists of 7 steps executed by 7 different tools. In the beginning of the process, before executing the workflow, all initial datasets must be imported into the workflow history. Two files are required. The first is the parameter file required by the GRANATUM Load Molecules module where the user sets the location of file to be read and other log files. The second is the file containing the docking results.

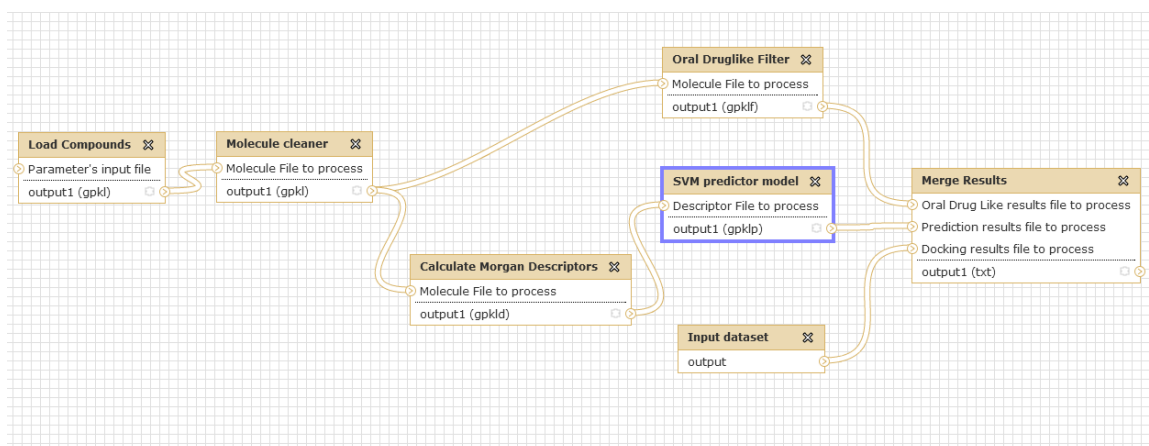


Figure 5.11 - The workflow as created in GALAXY


```

1: galaxy load_compounds.params
6 lines
format: txt, database: ?
Info: uploaded txt file

# load_compounds parameters file
# Designate the path and names for each of the following:
# -- structure (3D or smiles) file name for compounds to be imported
# (.sd, .sdf, .sd.gz, .sdf.gz, or .smi, .smi.gz)      # GZipped not ready yet!!!
# -- log file (.load_compounds.log)

```

Figure 5.12 - Input parameter file as displayed in Galaxy

In the first execution step, the Load Compounds command is executed (Fig. 5.12) which reads the parameters file and returns a file containing molecules. The next step is to execute clean molecules so that salts are removed and molecular validity is confirmed. This is an important procedure as it cleans the molecules preventing subsequent processes from failing because of inappropriate molecule format. Following, the cleaned molecules serve as input to 2 tools: Oral Drug like filter and Calculate Morgan fingerprints. As in the case of KNIME, the predictor model takes as input fingerprint descriptors that must be calculated first. Therefore the predictor model is called with the output of the Calculate Morgan fingerprints tool. The results of the predictor model and the Oral filter along with the docking scores are merged into a report (Fig. 5.14). The report is written out in a tab delimited file (Fig. 5.13).

	A	B	C	D	E	F	G
1	Smiles	ID	Oral_Druglike_Result	Oral_Druglike_Result_Reason	Prediction	Margin_Distance	Docking_Result
2	<chem>COc1ccc(C=O)c(O)c1O</chem>	344	1	["In range for HBD(None,5) with vActive		0.771420364692	-150.339.398.537
3	<chem>COc1c(O)c(O)cc(C=O)c1</chem>	345	1	["In range for HBD(None,5) with vActive		0.233026674221	-111.492.571.483
4	<chem>COc1cc(O)c2c(c1)OC(c1ccc(O)cc1)CC2=O</chem>	346	1	["In range for HBD(None,5) with vActive		0.208824972266	-108.314.805.197
5	<chem>COc1ccc2oc(-c3cc(O)cc(O)c3)cc(=O)c2c1</chem>	347	1	["In range for HBD(None,5) with vInactive		-0.314394440634	-147.943.896.522
6	<chem>COc1ccc(C(=O)c2ccc(OC)cc2O)c(O)c1</chem>	340	1	["In range for HBD(None,5) with vActive		0.785369631198	-10.537.202.617
7	<chem>OCCOC1CCCC(OCCO)C1</chem>	341	1	["In range for HBD(None,5) with vInactive		-0.578557303577	-128.949.413.516

Figure 5.13 – Results of the experiment in an Excel worksheet

21: Merge Results on data 20, data 2, and data 7

2,452 lines
 format: txt, database: ?
 Info: Created property matrix...
 transforming... 2435 39
 side length... 2435 39
 Property matrix with 38 properties for 2435 patterns!

There are 2451 compounds!

Get Prediction: 0% [] ETA: --:--:--
 Get Prediction: 1

📄 ⓘ ↻

Smiles	ID	Oral_Druglike_Result	Oral_Druglike_Result_Reason	Prediction	Margin_Dist:
COc1ccc(C=O)c(O)c1O	344	1	['In range for HBD(None,5) with value: 2.0. In range for HB		
COc1c(O)c(O)cc(C=O)c1	345	1	['In range for HBD(None,5) with value: 2.0. In range for HB		
COc1cc(O)c2c(c1)OC(c1ccc(O)cc1)CC2=O	346	1	['In range for HBD(None,5) with value: 2.0.		
	6				-10.8314805197

Figure 5.14 - Output File as displayed in Galaxy

STEP 3: Sharing and Collaboration

Sharing in Galaxy is easy. The workflow can be shared naturally among Galaxy users (fig. 5.15) and a default detailed panel showing the author, rating, tags and other related workflows is created. If a user wants to test the workflow all he has to do to is import the workflow and run it. Alternatively it is possible to send the workflow directly to myExperiment for publishing (Fig. 5.16). It is worth stressing that the latter approach is the only one feasible for sharing workflows implemented in SWMS systems like Knime.

Galaxy Analyze Data Workflow Shared Data Admin Help User

Published Workflows

search name, annotation, owner, and tags 🔍
[Advanced Search](#)

Name	Annotation	Owner	Community Rating	Community Tags	Last Updated
Chemoprevention Workflow		kleo-achilleos	★★★★★		~ 17 hours ago

Figure 5.15 – Published Chemoprevention Workflow in the Galaxy server instance at UCY

my experiment About | Mailing List | Publications Logout

Home Users Groups **Workflows** Files Packs Services

chemoprevention Workflows Search

Home » Workflows BOOKMARK f t e

Workflows

Search filter terms Sort by: Rank

Showing 3 results. Use the filters on the left and the search box below to refine the results.

chemoprevention

[Remove search query](#)

Filter by type

- KNIME 2
- Galaxy 1

Filter by user

- Kleo Achilleos 3

Filter by licence

- by-sa 2

Filter by group

- GRANATUM ... 1

KNIME **CHEMOPREVENTION WORKFLOW2 WITH DATA (v1)** View

Created: 19/05/12 @ 09:13:59 | **Last updated:** 19/05/12 @ 09:30:08 Download (v1)

Credits: Kleo Achilleos Manage

License: Creative Commons Attribution-Share Alike 3.0 Unported License

Original Uploader

CHEMOPREVENTION WORKFLOW WITH DATA

Rating: 0.0 / 5 (0 ratings) | **Versions:** 1 | **Reviews:** 0 | **Comments:** 0 | **Citations:** 0

Viewed: 0 times | **Downloaded:** 0 times

This Workflow has no tags!

Galaxy **Chemoprevention Workflow (v1)** View

Created: 19/05/12 @ 09:36:24 | **Last updated:** 19/05/12 @ 09:40:53 Download (v1)

License: No license Manage

Original Uploader

UTILIZES GRANATUM SOFTWARE MODULES TO PERFORM A CHEMOPREVENTION EXPERIMENT

Rating: 0.0 / 5 (0 ratings) | **Versions:** 1 | **Reviews:** 0 | **Comments:** 0 | **Citations:** 0

Viewed: 0 times | **Downloaded:** 0 times

This Workflow has no tags!

Figure 5.16 – Published Chemoprevention Workflows in myExperiment

Chapter 6

Results / Discussion

However beautiful the strategy, you should occasionally look at the results.¹⁹

In the preceding chapter, Chapter 5, the second goal, of the thesis was realized. The chemoprevention workflow successfully run in both systems and produced results potentially useful to domain scientists in this case chemoprevention experts. The collection of molecules was virtually screened and prioritized, and a subset of molecules was identified as potential candidates for biological screening. Both implementations produced tab delimited files containing all the necessary information for the domain scientist to make the inferences required. Consequently in both cases, savings in time and costs were achieved. The third goal of this thesis aims to identify differences, similarities and open issues of SWMS through the experiences gained from implementing the above workflows.

Comparison of results from both implementations

Table 6.1 summarizes the results as obtained by the implementations presented in Chapter 5. The results from the two workflow executions are highly similar. The Oral Drug Like filter manages to eliminate unsuitable molecules in the rate of 11% in KNIME's case and 16% in Galaxy's as presented in figures 6.1 and 6.2. The Oral Drug Like filter was implemented using different software libraries. CDK in KNIME and RDKit in the Galaxy implementation. This can explain the small 8% difference seen in Fig 6.5. As far as the Toxicity model is concerned

¹⁹ *Winston Churchill*

the overlap is at 72% shown in Figure 6.6. Again, different software libraries were used and therefore different models were created. The toxicity model manages to single out more than half of the initial molecules as possible candidates. These figures are presented in figures 6.3 and 6.4. A number of 707 out of 2435 had satisfactory binding affinity prediction results (docking score values <-15) shown in Figure 6.7. As previously noted the docking results are identical in both runs as the exact same model implementation was used.

Overall as seen from the Table 6.1 2163 molecules are Drug Like (Fig. 6.1). Of them 1183 are Inactive (Fig. 6.8) and a final of 315 pass all 3 filters (Fig. 6.9) in the KNIME workflow run. The VS process successfully identifies only a 12% of the initial dataset as possible candidates for biological evaluation. Figure 6.10 summarizes these results in a Venn diagram.

Results Table		
	KNIME	Galaxy - GRANATUM
Initial no of molecules	2536	2536
Minus rejected	2514 Stage 1- Reading 2451 Stage 2 - RDKit	2451
Pass Oral Drug Like	2163 /2438 (rejected 13) (37 not scored included in fail)	2049 / 2451
Inactive	1386 /2451 (1183 Pass oral)	1367 /2451 (1145 Pass Oral)
Docking (<-15)	<-15 706 / 2435 (93 not scored, 76 not matched) (315 (>-15) Inactive, Pass Oral)	

Table 6.1 - Results table

Fig 6.1 - Oral Drug Likeness Filter in KNIME - CDK

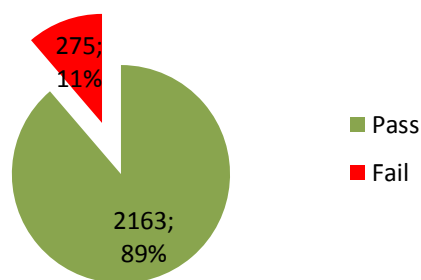


Fig 6.2 - Oral Drug like filter in Galaxy- RDKit

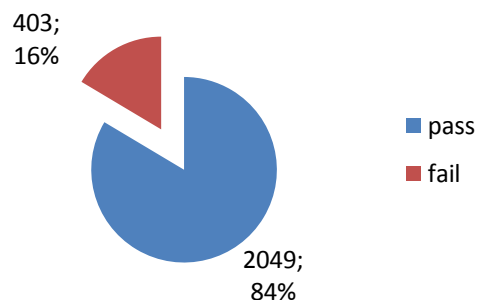


Fig 6.3 - Toxicity Prediction on all compounds - KNIME SVM

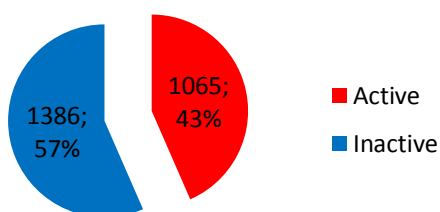


Fig 6.4 - Toxicity Prediction on all compounds - Galaxy SVM

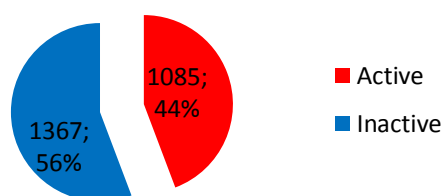
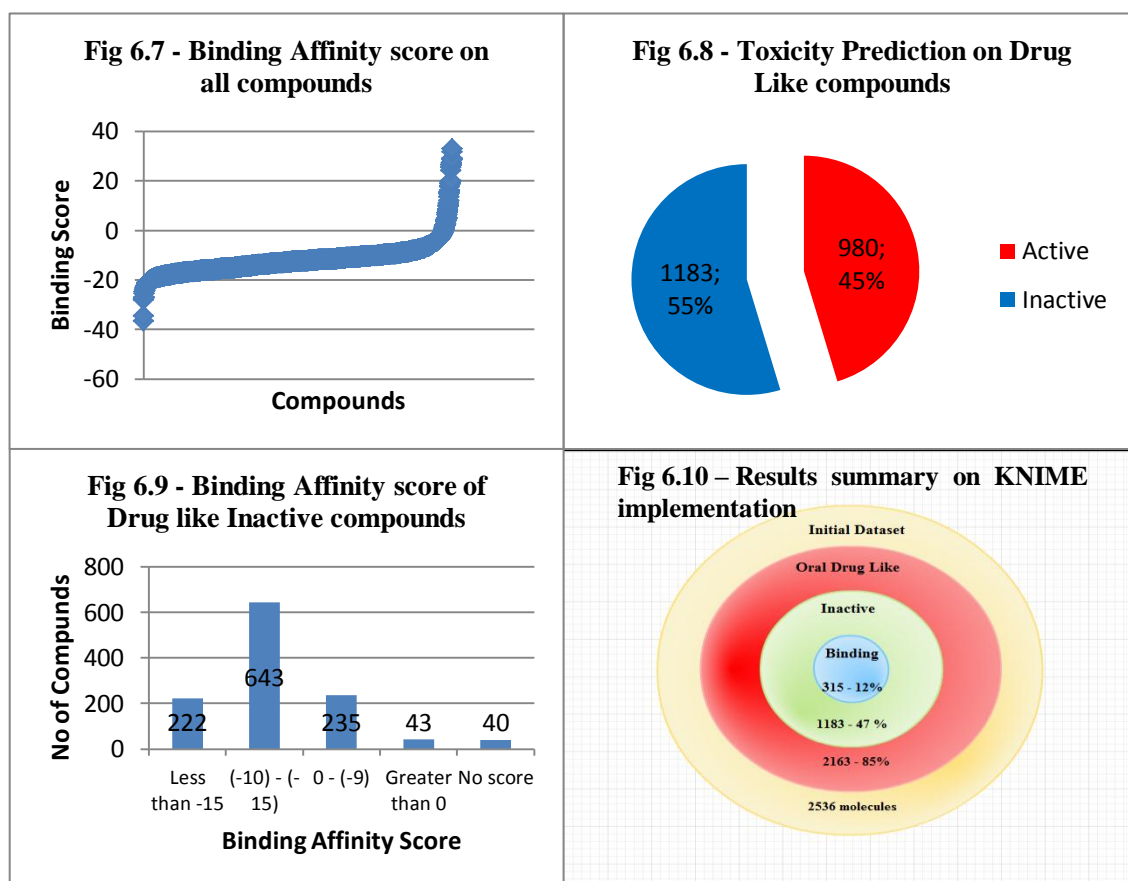


Fig 6.5 - Overlap results of Oral Drug Like Filter for KNIME-CDK and Galaxy-RDKit implementations



Fig 6.6 - Overlap results of Toxicity Model prediction for KNIME-Weka SMO and Galaxy-Scikit-learn Nu-SVC implementations





One striking difference between the two implementations is the complexity associated with each of them. In total, the Galaxy implementation has 7 tools versus the KNIME workflow that contains more than 30 nodes. This is an advantage of developing custom tools. A general tool/node, as in the case of KNIME, requires more data transformation between nodes, especially if the nodes are from different libraries. These data transformations are referred to as shim services by workflow experts. At least half of the nodes created in the VS workflow implemented in KNIME are data transformation nodes.

Another important difference is the lack of any visualization tools between intermediate steps in the case of the Galaxy implementation. Building custom tools has some negative consequences, one of which is that accessibility to common functionalities, e.g. visualization nodes is no longer available. These too must be customized. In contrast, all KNIME nodes

enable the visualization of their output results in each step which is very useful in assessing quality and correctness, especially when first developing and testing the workflow.

The runtime environment is also different. The KNIME workflow ran on a personal desktop computer while the other ran on an online Galaxy server. Having a server implementation offers benefits for resource management that were not explored in the context of this thesis. However, it is well understood that an online implementation enables the centralized management of the workflow system and its tools/nodes and thus facilitates maintenance by administrators. It also allows users to focus on the design and implementation of their *in silico* experiments and leave technical matters to SWMS administrators. In turn, a well-supported and documented online system can facilitate the adoption of such technology by users not comfortable with software administration, for example, biologists and chemists. On the other hand, the KNIME approach may be more appropriate for users more comfortable with managing computational tools as it allows more control of the SWMS system and the nodes and workflows that the user may develop. In the case of the GRANATUM project it is obvious that a system such as Galaxy would better fit its purpose.

As far as execution time is concerned, it is not safe to make any conclusions. The heaviest processes were the ones involved in building and executing the SVM model in both systems. Indicatively, the KNIME workflow required 3 min and the GALAXY required 3.5 min. However, GALAXY was run on a test server of older technology where other users had concurrent access while KNIME was run on a dedicated high-end personal computer. On the other hand, the KNIME workflow had a lot of data transformations nodes as opposed to custom designed function modules.

A summary of the advantages and disadvantages of each implementation is given in the table

6.2

KNIME	Galaxy
+ Built in chemistry support and Multiple chemistry plug in to choose from	- Limited chemo informatics tools
+ Data visualization in each step	- Limited data visualization
+ Built-in Machine learning models and plug-in	- No machine learning integration
+ Adequate Documentation of Nodes	+ Adequate Documentation of Nodes
+ Easy installation	+ No installation by end user; handled by server administrator
+ Sharing in my Experiment	+ Direct sharing with myExperiment
- Need manually configured updates	+ Updates by the server administrator
- Need of data transformation processes (known as shim)	+ Custom development
- Limited functionality for collaboration	+ Built in sharing and collaboration
- Local resources	+ Online environment
- Available locally	+ Available from anywhere

Table 6.2 – Comparing KNIME and Galaxy implementations

Critical Review of Open Source SWMS

User-friendliness: One of the strong selling points of SWMS technology is the promise to allow and trivialize the implementation of complex scientific experiments by non-expert users. Ideally, users with little background in databases and algorithm implementation will be able to design *in silico* experiments that make use of data with varying formats from distributed resources and analyze it using methods executed on computational resources as

required. Currently, this is clearly not the general case. Most modern SWMS have made significant steps in this direction but still remain sophisticated tools which may be intimidating to the non-computational user. However, a middle solution is using current technology and based on user requirements implement customized solutions with not too much effort. This custom solution will hide all unnecessary complex details from the end user while at the same time provide equal functionality. This is the approach that the GRANATUM team is currently following.

Support mechanisms: The support mechanisms of open source software are typically the wiki pages and mailing lists administered by expert users (Fig. 6.11). As such it is up to the community of each tool to adequately support new users and guide them through their initial usages of the tool. Personally, I have resorted to online resources and support by the community of KNIME and GALAXY and found that both communities were quick to assist although it usually took several iterations of email exchanges to solve the problem. In all SWMS examined, more can be done in the form of tutorials, videos, better documentation of common errors, etc.

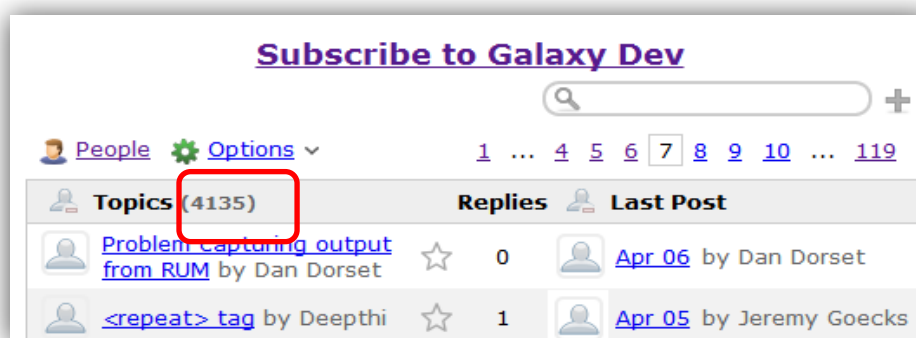


Figure 6.11 – 4135 Topics in Galaxy Development list and 188 only in April 2012 from <http://dev.list.galaxyproject.org>

Error handling: The error messages should be meaningful, as the intended users are not gurus but rather beginners in most cases. KNIME has an excellent error mechanism. It prevents errors from happening by using the configuration menu on the nodes and by setting

the status of node. If something is wrong the node status stays red and cannot be executed. If the data is not readable then it cannot be executed and so on. In Galaxy's case, it depends on the tool's configuration what kind of error it produces. As is the case with KNIME, Galaxy prevents errors by carefully checking the data type of the input into the next tool.

In particular, adequate documentation is a must for the prevention of errors. Not only to describe what each component or tool does but also what are its inputs, outputs and notably common errors and how to deal with them. Both tools are well documented.

Integration of heterogeneous resources: SWMS's have great potential in implementing complex *in silico* experiments integrating computational and data resources from varying sources. Currently, this feature is supported by GALAXY. Support of retrieval of data from online data libraries is an important feature of that tool as is visualizing through online browsers. KNIME, as a desktop tool lacks support in this very important feature. Expert users may be able to prepare KNIME nodes that communicate with e.g. web services to access and use distributed resources and data repositories but this is not a feature KNIME was designed to address or emphasizes.

Inter-operability: The number of open source SWMS is not small, as some SWMS are domain specific while others are domain independent, others are configured for the grid, others for remote services calls etc. It is also obvious that this number will evidently grow. That is not worrying but rather expected, as new technology offers more functional solutions, enabling for more efficient architectures. As noted in [4] the aim is not to reduce the number of SWMS but rather to make sure that these systems can interact. The only feasible way to do this is by the use of standards. In fact, some experts have argued in favor of using the standardized successful business workflow language currently in use. Others have argued against it, as scientific workflows are not a subcategory of business workflows but have distinctive differences such as data flow control rather than control flow, massive

heterogeneous data volumes requiring integration, intensive computation and demanding user interaction and visualization.

If seen from a user perspective, a workflow should be platform independent. This is not feasible without standardization. The way Taverna and Galaxy interact is by creating an executable “black box” that encompasses the functionality of the Taverna workflow. The black box is then executed in the Galaxy platform. KNIME does not address this issue at all. Perhaps, the most attractive model for succeeding interoperability is the one the internet is based upon. The workflows packaged as services themselves. This is the case now for the software components.

Workflow Sharing: The primary example of well thought workflow sharing can be found in myExperiment, an online collaboration environment, designed specifically for the sharing of workflows prepared using the Taverna SWMS. Eventually, myExperiment usage is spreading to other open source workflow systems; currently KNIME workflows can be shared through this platform by an import operation. In the case of Galaxy users can share workflows both within the workbench through the sharing option and by exporting their workflow directly into the myExperiment environment.

Chapter 7

Conclusions / Future Work

*Learning without thought is labor lost; thought without learning is perilous.*²⁰

Scientific workflows and Scientific Workflow Management Systems have changed the dynamics of many scientific disciplines and have accelerated scientific discoveries. They enabled domain scientists to explore, visualize, process, transform, store and model huge volumes of heterogeneous data by the use of functional cross platform software components utilizing enormous processing power as offered by grid or cloud technology (Fig. 7.1)

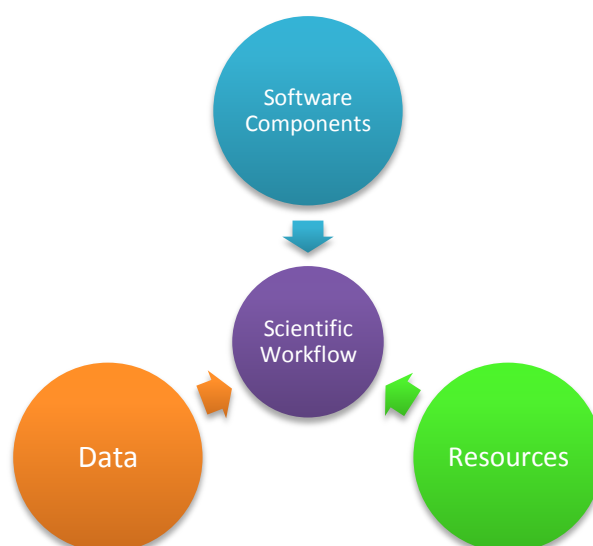


Figure 7.1 – SW and its constituent parts

The main benefit of the SWMS approach is the transparency it offers for Data and Resource management. Domain scientists are not interested in the actual form of the data (binary, text, stream or not) nor where the actual processing will occur (locally or remotely). She/he is interested in the results and the form they will be presented. The second most important benefit is provenance capture. The information gathered by the SWMS during the execution of a workflow enables the reproducibility of the experiment. Additionally, the use of online

²⁰ Confucius

repositories of scientific workflows is documented, knowledge flow is promoted and collaboration encouraged. All together can support and accelerate scientific work and discovery. Finally, current SWMS provide a friendlier, usable and visual working environment supporting easy reuse. As a direct result SWMS are gaining ground and are rapidly accepted and used in the daily work routine of numerous research fields.

Online SWMS offer additional benefits. There is no need to set up installations on local machines or remote servers, no downloads, no conflicts, no updates to worry about. Secondly, the tools are available at any personal computer from anywhere in the world provided that they are connected to the internet. The same applies to data. A scientist can import and use their data in the system available along with the workflow. Moreover the data and work are secure and can be backed up and protected depending always to the system's specifications. Provenance information collected by the system can also serve for documentation purposes and for future reference. Importantly, all data and work can be shared with other collaborators in real time. Some online SWMS even offer more advanced features such as transparent access to HPC, to grid services or the cloud, thus, offering speed and efficiency for scientific processes that are computationally expensive and/or data intensive.

Summing up the results discussed in Chapter 6 the two systems presented are not rivals. KNIME has an attractive interface, it is easily installed, it has built-in chemistry support, multiple chemoinformatics plug-in tools to choose from, it provides data visualization tools, machine learning models and it has adequate documentation. On the other hand, Galaxy's support of chemoinformatics processes is minimal and its interface is not so attractive. However, it is its inner beauty that counts. Galaxy's online environment, except from the fact that it follows the modern trend of online presence, has numerous advantages such as no need for local system support, constant and global availability, built-in sharing and collaboration features. Moreover, Galaxy's architecture enables easy integration of scripting tools. KNIME is considered among the top open source software for chemoinformatics. Galaxy is a

promising platform that is gaining supporters every day. Evidence submitted are the Galaxy-Taverna integration in 2011, the myExperiment integration expected around mid of 2012 and the number of Galaxy servers currently running.

Galaxy is not ready to support chemo-informatics experiments as is. It needs the development and contribution of further tools. It is however ready to support custom code as tools. Its platform offers all the requirements for supporting a scientific workflow experiment: provenance, data and resource transparency, sharing and collaboration, user friendly environment and documentation. KNIME, is ready to support a number of chemo-informatics experiments as it has built in support but also it has contributions of third party tools. It too, can integrate in house code as nodes. However, features such as online interface along with built-in collaboration and sharing required by the large, interdisciplinary projects such as GRANATUM, are not provided by the open source version, and are restricted to the KNIME server which is a commercial product.

My development work is far from complete. One limitation of the GRANATUM tools current implementation is the lack of any visualization tools. Currently, there is no visualization for the intermediate step results. That should be one priority. Another should be to integrate open source molecular visualization tools, so that molecules can be seen also in their structural form. The second important implementation concern, already mentioned in chapter 5, is that the collection of tools implemented is fairly limited and custom to the implementation of a relatively simple proof of concept. Consequently additional tool development needs to be done. The docking module needs to be further developed and properly integrated and additional predictive models and filters need to be implemented for use by chemoprevention experts.

Going beyond the current implementation, I would say that Galaxy has proven itself as a promising platform for supporting scientific experiments. As such I would suggest enabling full functionality of the server as a Galaxy server in the University of Cyprus and listing it in

Galaxy's List of Galaxy servers available all over the globe. Also important is giving access to the Galaxy server to other researchers and organizing a training course to promote its use. Since using Galaxy as a tool integration platform was proven to be a straightforward task, more tools should be integrated either from GRANATUM or from other project and domains. We expect that such development and further integration of tools into Galaxy and similar SWMS systems will surely continue in the immediate future and will conquer even more scientific fields in the coming years.

Bibliography

- [1] A. Barker and J. V. Hemert, "Scientific Workflow : A Survey and Research Directions," pp. 746-753, 2008.
- [2] T. McPhillips, S. Bowers, D. Zinn, and B. Ludäscher, "Scientific workflow design for mere mortals," *Future Generation Computer Systems*, vol. 25, no. 5, pp. 541-551, 2009.
- [3] D. Barseghian et al., "Workflows and extensions to the Kepler scientific workflow system to support environmental sensor data access and analysis," *Ecological Informatics*, vol. 5, no. 1, pp. 42-50, Jan. 2010.
- [4] C. Goble, P. Missier, and D. De Roure, "Scientific workflows," in *Current opinion in drug discovery development*, vol. 11, no. 3, McGraw Hill, 2008, pp. 527-534.
- [5] P. Missier, S. Soiland-Reyes, S. Owen, W. Tan, A. Nenadic, I. Dunlop, A. Williams, T. Oinn, and C. Goble, "Taverna, reloaded," in *SSDBM 2010*, Heidelberg, Germany, 2010.
- [6] D. Hull, K. Wolstencroft, R. Stevens, C. Goble, M. Pocock, P. Li, and T. Oinn, "Taverna: a tool for building and running workflows of services.," *Nucleic Acids Research*, vol. 34, iss. Web Server issue, pp. 729-732, 2006.
- [7] T. Oinn, M. Greenwood, M. Addis, N. Alpdemir, J. Ferris, K. Glover, C. Goble, A. Goderis, D. Hull, D. Marvin, P. Li, P. Lord, M. Pocock, M. Senger, R. Stevens, A. Wipat, and C. Wroe, "Taverna: lessons in creating a workflow environment for the life sciences," *Concurrency and Computation: Practice and Experience*, vol. 18, iss. 10, pp. 1067-1100, 2006.
- [8] I. Taylor, M. Shields, I. Wang, and R. Philp, "Distributed P2P Computing within Triana: A Galaxy Visualization Test Case," in *IPDPS 2003 Conference*, 2003, p. 16.1.

- [9] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludascher, and S. Mock, "Kepler: an extensible system for design and execution of scientific workflows," Proceedings 16th International Conference on Scientific and Statistical Database Management 2004, vol. 59, pp. 423-424, 2004.
- [10] E. Deelman, G. Singh, M.-hui Su, J. Blythe, Y. Gil, and C. Kesselman, "Pegasus : A framework for mapping complex scientific workflows onto distributed systems," *Jet Propulsion*, vol. 13, pp. 219-237, 2005.
- [11] M. R. Berthold et al., "KNIME: The Konstanz Information Miner," in *Data Analysis Machine Learning and Applications*, C. Preisach, H. Burkhardt, L. Schmidt-Thieme, and R. Decker, Eds. Springer Berlin Heidelberg, 2008, pp. 319-326.
- [12] M. R. Berthold et al., "KNIME - the Konstanz information miner: version 2.0 and beyond," *SIGKDD Explor Newsl*, vol. 11, no. 1, pp. 26-31, 2009.
- [13] Accelrys, "Pipeline Pilot," *Methods*. Accelrys, p. 2007, 2010.
<http://accelrys.com/products/pipeline-pilot/>
- [14] A. Rowe, D. Kalaitzopoulos, M. Osmond, M. Ghanem, and Y. Guo, "The discovery net system for high throughput bioinformatics," *Bioinformatics*, vol. 19, no. 90001, p. 225i-231, 2003.
- [15] R. S. Barga et al., "Trident: Scientific Workflow Workbench for Oceanography," 2008 IEEE Congress on Services - Part I, pp. 465-466, Jul. 2008.
- [16] B. Giardine et al., "Galaxy: A platform for interactive large-scale genome analysis," *Genome Research*, vol. 15, no. 10, pp. 1451-1455, 2005.
- [17] D. Blankenberg et al., "Galaxy: a web-based genome analysis tool for experimentalists.," *Current protocols in molecular biology edited by Frederick M Ausubel et al*, vol. Chapter 19, no. January, pp. Unit 19.10.1-21, 2010.
- [18] J. Goecks, A. Nekrutenko, and J. Taylor, "Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences," *Genome Biology*, vol. 11, no. 8, p. R86, 2010.

- [19] Wassermann, B., et al., Sedna: A BPEL-Based Environment for Visual Scientific Workflow Modeling, in *Workflows for E-science: Scientific Workflows for Grids*, I.J. Taylor, et al., Editors. 2007, Springer-Verlag. p. 428-449.
- [20] E. Deelman, G. Singh, M.-hui Su, J. Blythe, Y. Gil, and C. Kesselman, "Pegasus : A framework for mapping complex scientific workflows onto distributed systems," *Jet Propulsion*, vol. 13, pp. 219-237, 2005.
- [21] "Web Services Business Process Execution Language Version 2.0" Internet: <http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.pdf>, [May 01, 2012].
- [22] B. Ludäscher, M. Weske, T. McPhillips, and S. Bowers, "Scientific Workflows: Business as Usual?," *BPM*, vol. 5701. Springer, pp. 31-47, 2009.
- [23] WfMC, "The Workflow Reference Model," *Management*, vol. am, no. 1. pp. 1-55, 1995.
- [24] T. Fleuren, J. Götze, and P. Müller, "Workflow Skeletons: Increasing Scalability of Scientific Workflows by Combining Orchestration and Choreography," 2011 IEEE Ninth European Conference on Web Services, pp. 99-106, 2011.
- [25] B. Ludäscher et al., "Scientific Process Automation and Workflow Management," *Development*, vol. 10, no. 3, pp. 476–508, 2009.
- [26] M. Sonntag and D. Karastoyanova, *Conventional Simulation Workflow Technology for Scientific Conventional Workflow Technology for Scientific Simulation*. 2011, pp. 0-31.
- [27] E. Deelman, D. Gannon, M. Shields, and I. Taylor, "Workflows and e-Science: An overview of workflow system features and capabilities," *Future Generation Computer Systems*, vol. 25, no. 5, pp. 528-540, May 2009.
- [28] J. Yu and R. Buyya, "A Taxonomy of Scientific Workflow Systems for Grid Computing," *System*, vol. 34, no. 3, pp. 44-49, 2005.
- [29] I. J. Taylor, E. Deelman, and D. B. Gannon, *Workflows for e-Science*. Springer, 2007, p. 523.
- [30] A. Tiwari and A. K. T. Sekhar, "Workflow based framework for life science informatics.," *Computational Biology and Chemistry*, vol. 31, no. 5-6, pp. 305-319, 2007.

- [31] Y. Gil et al., "Examining the Challenges of Scientific Workflows," *Computer*, vol. 40, no. 12, pp. 24-32, 2007.
- [32] A. C. Siepel et al., "An integration platform for heterogeneous bioinformatics software components," *IBM Systems Journal*, vol. 40, no. 2, pp. 570-591, 2001.
- [33] R. Buyya and S. Venugopal, "The Gridbus Toolkit for Service Oriented Grid and Utility Computing: An Overview and Status Report," *Source*, p. 11, 2004.
- [34] N. Furmento, A. Mayer, S. McGough, S. Newhouse, T. Field, and J. Darlington, "ICENI: Optimisation of component applications within a Grid environment," *Parallel Computing*, vol. 28, no. 12, pp. 1753-1772, 2002.
- [35] C. Walton and A. Barker, "An Agent-Based e-Science Experiment Builder," in *Proceedings of The 1st International Workshop on Semantic Intelligent Middleware for the Web and the Grid European Conference on Artificial Intelligence ECAI*, 2004.
- [36] J. L. Brown et al., "GridNexus: A Grid Services Scientific Workflow System," *International Journal of Computer and Information Science*, vol. 6, no. 2, 2005.
- [37] T. Fahringer et al., "ASKALON: A Grid Application Development and Computing Environment," *The 6th IEEEACM International Workshop on Grid Computing 2005*, pp. 122-131, 2005.
- [38] D. D. Roure et al., "myExperiment: Defining the Social Virtual Research Environment," *2008 IEEE Fourth International Conference on eScience*, vol. 0, no. July, pp. 182-189, 2008.
- [39] Y. Simmhan, R. Barga, C. V. Ingen, E. Lazowska, and A. Szalay, "Building the Trident Scientific Workflow Workbench for Data Management in the Cloud," *2009 Third International Conference on Advanced Engineering Computing and Applications in Sciences*, pp. 41-50, Oct. 2009.
- [40] W. Tan et al., "caGrid Workflow Toolkit: A Taverna based workflow tool for cancer Grid.," *BMC Bioinformatics*, vol. 11, no. 1, p. 542, 2010.
- [41] "Biodiversity Virtual e-Laboratory" <http://www.biovel.eu>, [May 01, 2012].
- [42] "BioCatalogue.org" Internet: <http://www.biocatalogue.org/>, [May 01, 2012].

- [43] “Galaxy / Huttenhower Lab” Internet: <http://huttenhower.org/galaxy/>, [May 01, 2012].
- [44] V. Curcin and M. Ghanem, “Scientific workflow systems - can one size fit all?,” 2008 Cairo International Biomedical Engineering Conference, pp. 1-9, 2008.
- [45] S. Shumilov, Y. Leng, M. El-Gayyar, and a. B. Cremers, “Distributed Scientific Workflow Management for Data-Intensive Applications,” 2008 12th IEEE International Workshop on Future Trends of Distributed Computing Systems, pp. 65-73, 2008.
- [46] C. L. C. Lin et al., A Reference Architecture for Scientific Workflow Management Systems and the VIEW SOA Solution, vol. 2, no. 1. Published by the IEEE Computer Society, 2009, pp. 79-92.
- [47] “National Cancer Institute” Internet: <http://www.cancer.gov/>, [May 01, 2012].
- [48] “cancergrid” Internet: <http://www.cancergrid.org/>, [May 01, 2012].
- [49] E. A. Lee, “Overview of the Ptolemy Project,” <http://ptolemy.eecs.berkeley.edu>, no. UCB/ERL M03/25. 2003
- [50] D. Chappell, “Introducing Windows Workflow Foundation,” Microsoft, 2009.
- [51] “myGrid” Internet: <http://www.mygrid.org.uk/>, [May 01, 2012].
- [52] “Taverna – open source and domain independent Workflow Management System” Internet: <http://www.taverna.org.uk/>, [May 01, 2012].
- [53] “myGrid” Internet: <http://www.mygrid.org.uk/dev/wiki/display/taverna/>, [May 01, 2012].
- [54] “Eclipse – The Eclipse Foundation open source community website” Internet: <http://www.eclipse.org>, [May 01, 2012].
- [55] M. Abouelhoda, S. Alaa, and M. Ghanem, “Meta-workflows: pattern-based interoperability between Galaxy and Taverna,” in Proceedings of the 1st International Workshop on Workflow Approaches to New Datacentric Science, 2010, pp. 1-8.
- [56] N. Brown, “Chemoinformatics—an introduction for computer scientists,” ACM Computing Surveys, vol. 41, no. 2, pp. 1-38, 2009.

- [57] M. Modeling, C.-aided D. Design, D. J. Wild, and W. I. Consulting, "Getting Started in Chemoinformatics," Wild, no. September, pp. 1-49, 2004.
- [58] A. Varnek and I. I. Baskin, "Chemoinformatics as a Theoretical Chemistry Discipline," Molecular Informatics, vol. 30, no. 1, pp. 20-32, 2011.
- [59] A. S. Reddy, S. P. Pati, P. P. Kumar, H. N. Pradeep, and G. N. Sastry, "Virtual screening in drug discovery -- a computational perspective.," Current protein peptide science, vol. 8, no. 4, pp. 329-351, 2007.
- [60] C. A. Lipinski, F. Lombardo, B. W. Dominy, and P. J. Feeney, "Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings," Advanced Drug Delivery Reviews, vol. 46, no. 1-3, pp. 3-26, 2001.
- [61] C. Wegscheid-Gerlach, Chemoinformatics Approaches to Virtual Screening. Von Alexandre Varnek und Alexander Tropsha (Hrsg.), vol. 38, no. 5. Royal Society of Chemistry, 2009, pp. 473-473.
- [62] G. J. Kelloff et al., "Agents, biomarkers, and cohorts for chemopreventive agent development in prostate cancer.," Urology, vol. 57, no. 4 Suppl 1, pp. 46-51, 2001.
- [63] "German Cancer Research Center" Internet: <http://www.dkfz.de>, [May 01, 2012].
- [64] "GRANATUM" Internet: <http://granatum.org>, [May 01, 2012].
- [65] "CORDIS" Internet: <http://cordis.europa.eu>, [May 01, 2012].
- [66] E. M. Krovat, T. Steindl, and T. Langer, "Recent Advances in Docking and Scoring," Current Computer Aided Drug Design, vol. 1, no. 1, pp. 93-102, 2005.
- [67] "KNIME" Internet: <http://www.knime.org>, [May 01, 2012].
- [68] "KNIMEtech" Internet: <http://tech.knime.org/>, [May 01, 2012].
- [69] "Galaxy" Internet: <http://galaxy.psu.edu/>, [May 01, 2012].
- [70] "UCSC" Internet: <http://genome.ucsc.edu/>, [May 01, 2012].
- [71] "ENSEMBL" Internet: <http://www.ensembl.org>, [May 01, 2012].
- [72] "InforSense" Internet: Internet: <http://www.inforsense.com/>, [May 01, 2012].
- [73] "Accelrys" Internet: <http://accelrys.com>, [May 01, 2012].

- [74] “Askalon Programming Environment for Grid Computing” Internet: <http://www.askalon.org/>, [May 01, 2012].
- [75] Y. Simmhan, “End-to-End Scientific Data Management Using Workflows,” 2008 IEEE Congress on Services - Part I, pp. 472-473, Jul. 2008.
- [76] X. Yang, R. P. Bruin, and M. T. Dove, “Developing an End-to-End Scientific Workflow:,” *Computing in Science & Engineering*, vol. 12, no. 3, pp. 52-61, May 2010.
- [77] W. A. Warr and W. Warr, “Integration , Analysis and Collaboration . An Update on Workflow and Pipelining in Cheminformatics,” *Life Sciences*, pp. 1-7, 2007.
- [78] “Python” www.pyhton.org, [May 01, 2012].
- [79] “Java Eclipse SDK” <http://www.eclipse.org/>, [May 01, 2012].
- [80] “Weka” www.cs.waikato.ac.nz/ml/weka, [May 01, 2012].
- [81] “CDK” Internet: <http://sourceforge.net/projects/cdk/>, [May 01, 2012].
- [82] “RDKit” Internet: www.rdkit.org, [May 01, 2012].
- [83] “Erlwood” Internet: tech.knime.org/community/erlwood, [May 01, 2012].
- [84] “Indigo” Internet: <http://ggasoftware.com/opensource/indigo>, [May 01, 2012].
- [85] “CADDSSuite” Internet: <http://www.ballview.org/caddssuite>, [May 01, 2012].
- [86] Bouckaert, R. R., Frank, E., Hall, M. A., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2010). WEKA-Experiences with a Java Open-Source Project. *Journal of Machine Learning Research*, 11:2533-2541
- [87] Kohavi, R., & Provost, F. (1998). Special Issue on Applications of Machine Learning and the Knowledge Discovery Process. *Machine Learning*, 30(2/3), 127-271.
- [88] Kohavi, R. (1995). A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. *International Joint Conference on Artificial Intelligence*, 14(12), 1137–1145. Citeseer. doi:10.1067/mod.2000.109031
- [89] Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1), 21-27. IEEE. doi:10.1109/TIT.1967.1053964

- [90] Hearst, M. A., Dumais, S. T., Osman, E., Platt, J., & Scholkopf, B. (1998). Support vector machines. (M. Jordan, J. Kleinberg, & B. Scholkopf, Eds.)IEEE Intelligent Systems, 13(4), 18-28. Springer. doi:10.1109/5254.708428
- [91] Mishra, H., Singh, N., Lahiri, T., & Misra, K. (2009). A comparative study on the molecular descriptors for predicting drug-likeness of small molecules. *Bioinformatics*, 3(9), 384-388. Biomedical Informatics Publishing Group.
- [92] Guyon, I., & Elisseeff, A. (2003). An Introduction to Variable and Feature Selection. (L. P. Kaelbling, Ed.)*Journal of Machine Learning Research*, 3(7-8), 1157-1182
- [93] Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data Mining: Practical Machine Learning Tools and Techniques*. *Annals of Physics* (Vol. 54, p. 664). Morgan Kaufmann
- [94] “AID464 – PubChem BioAssay Summary” Internet: <http://pubchem.ncbi.nlm.nih.gov/assay/assay.cgi?aid=464>, [May 01, 2012].
- [95] “Indofine Chemical Company, Inc” Internet: <http://www.indofinechemical.com/>, [May 01, 2012]
- [96] Scikit-learn: Machine Learning in Python, Pedregosa et al., *JMLR* 12, pp. 2825-2830, 2011.