Transatlantic TUmour MOdel Repositories

D5.2
Final Integrated Environment prototype

Project Number: FP7--IST-247754
Deliverable id: D5.2
Deliverable name: Final Integrated Environment prototype
Submission Date: 25/010/2013

COVER AND CONTROL PAGE OF DOCUMENT	
Project Acronym:	TUMOR
Project Full Name:	Transatlantic TUmour MOdel Repositories

Document id:	D5.2
Document name:	Final integrated environment prototype
Document type (PU, INT, RE)	RE
Version:	1.1
Submission date:	
Editor: Organisation: Email:	Stelios Sfakianakis FORTH-ICS ssfak@ics.forth.gr

Document type PU = public, INT = internal, RE = restricted

ABSTRACT: This deliverable document describes the final integrated prototype of the project and provides the results of its evaluation based on a tutorial and questionnaire submitted to a number of users.

KEYWORD LIST: scientific workflows, integration, VPH, evaluation

MODIFICATION CONTROL			
Version	Date	Status	Author
0.2	9 September 2013	Draft	Stelios Sfakianakis
0.5	20 September 2013	Draft	Vangelis Sakkalis
1.0	10 October 2013	Pre-Final	Stelios Sfakianakis
1.1	25 October 2013	Final	Stelios Sfakianakis

List of Contributors

- Stelios Sfakianakis (FORTH)
- Vangelis Sakkalis (FORTH)
- Fay Misichroni (ICCS)
- David Johnson (UOXF.BL)
- Steve McKeever (UOXF.BL)
- Norbert Graf (USAAR)
- Thomas Taylor (INFOTECH Soft)

Contents

1 EXECUTIVE SUMMARY 5
2 ARCHITECTURE 5
3 A WALKTHROUGH OF THE TUMOR WORKFLOW ENVIRONMENT 10
3.1 INTRODUCTION 10
3.2 LOGGING INTO THE SYSTEM 10
3.3 MODEL SEARCH AND FILTERING 11
3.4 MODEL INSPECTION AND LINKING 13
3.5 POPULATING MODELS WITH PATIENT DATA 16
3.6 SAVING AND LOADING WORKFLOWS 18
3.7 WORKFLOW EXECUTION AND MONITORING 19
4 EVALUATION AND USABILITY RESULTS 22
4.1 USABILITY TESTING PROCESS 22
4.1.1 Usability Measurement Tools 24
4.2 EVALUATION RESULTS 26
4.2.1 System Satisfaction Scale Results (SUS) 26
4.2.2 User Success Rate Results (SAS) 27
APPENDIX A – USER SUCCESS RATE FORM 29
APPENDIX B–SYSTEM SATISFACTION SCALE QUESTIONNAIRE (SUS) 30

1 Executive Summary

According to its Description of Work, the TUMOR project aims to build an integrated, interoperable transatlantic research environment offering the best available models and tools for clinically oriented cancer modelling and serving as an international validation/ clinical translation platform for predictive, in-silico oncology.

This document describes the final version of the TUMOR integrated environment in terms of its architecture. IN addition, it presents the results of its evaluation based on a detailed walkthrough tutorial and questionnaire-based usability tools submitted to a number of users.

2 Architecture

The TUMOR integrated environment comprises a distributed software system and therefore it is essential to describe its architectural characteristics, i.e. its components, the interactions between these components, and the principles and non-functional characteristics of these interactions. A general overview of the system is shown in the figure below:

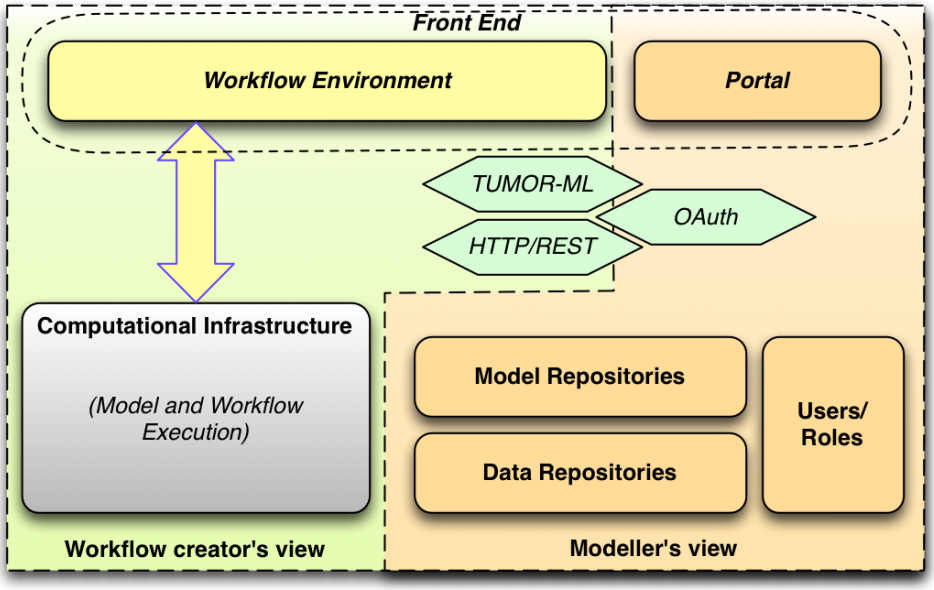


Figure 1 The main components of the TUMOR platform

We consider two main “facades” of the system based on its main use cases (Figure 2):

- In the “Modellers’ view” the main functionality of the system deals with the management of computational models and relevant data in the Model and Data repositories of the platform. In this façade users upload/register models and data, and the system through its Portal supports their discovery, navigation, and download. The primary entry point for this view of the system is the EU Model and data repository through its portal, which is also the “Common Access Point” for the whole platform.
- In the “Workflows creator view” the system supports the linking and execution of the linked models in a “software as a service” way. The actual execution happens in the TUMOR’s servers backend without imposing any load in the users local machines. The entry point for this view of the platform is the TUMOR workflow environment.

In the Modellers’ view the users interact only with the model and data repository and therefore there is not an “integration agenda” to consider (please see Deliverable 3.1.2 for the final version of the EU repository). The real integration happens in the second view of the system, where the different components (Workflow environment, model and data repositories) need to cooperate and communicate with each other. Therefore in

the rest of the document we primarily focus on this second group of user scenarios and interactions within the platform.

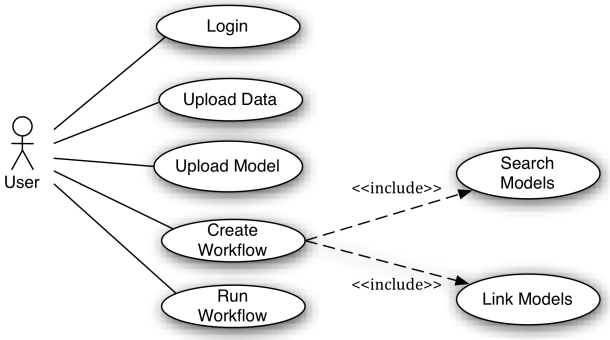


Figure 2 The main use cases for the TUMOR integrated platform

The decomposition of the platform into components and its actual deployment is shown in the next figure:

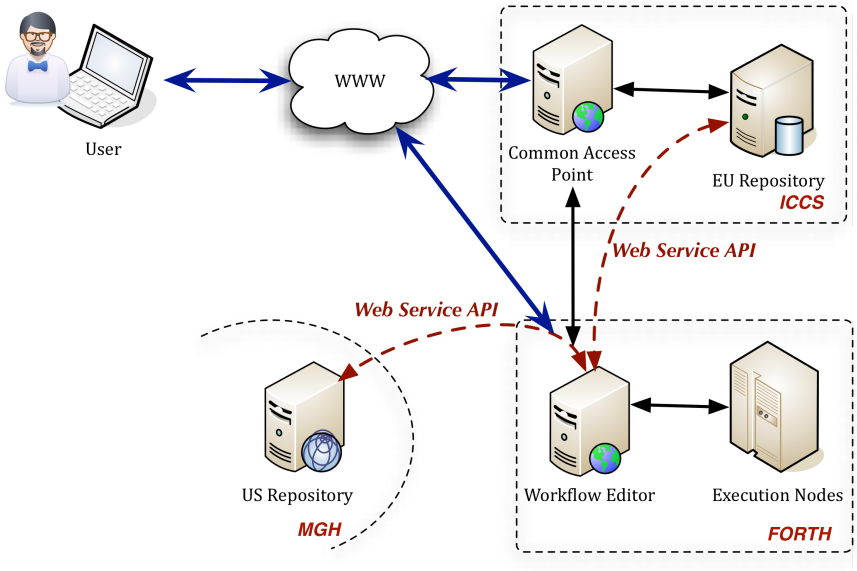


Figure 3 The deployment of TUMOR components.

The EU Repository and its Common Access Point (CAP) are hosted in the ICCS premises. The Workflow Editor and Engine alongside with its supporting execution infrastructure is deployed in FORTH. The US Repository is hosted on the other side of the Atlantic, in the CViT.org infrastructure.

The TUMOR integrated environment is therefore a multi-component distributed system, where each component serves part of the platform's functionality. For example, the workflow environment is a separate web application that stores neither user login information, nor the models and the accompanied data. So there is a need for *Single Sign On*, so that the users are not required to signup twice or to provide the same credentials twice when they access the CAP/EU Repository and the Workflow Environment. Additionally the users should be allowed to make secure and authenticated requests to the model repositories through the workflow environment. To address both of these concerns, TUMOR uses the OAuth 2.0 (Open Authorization, version 2.0) protocol[1] that is also supported by Google, Microsoft, and Facebook in their web applications. The interactions among the Common Access Point and the

[1] http://oauth.net/2/

Workflow environment are shown in the next figure. After the successful authentication of the user, the workflow environment can access the model repositories on the users' behalf without knowing their passwords or other authenticating information.

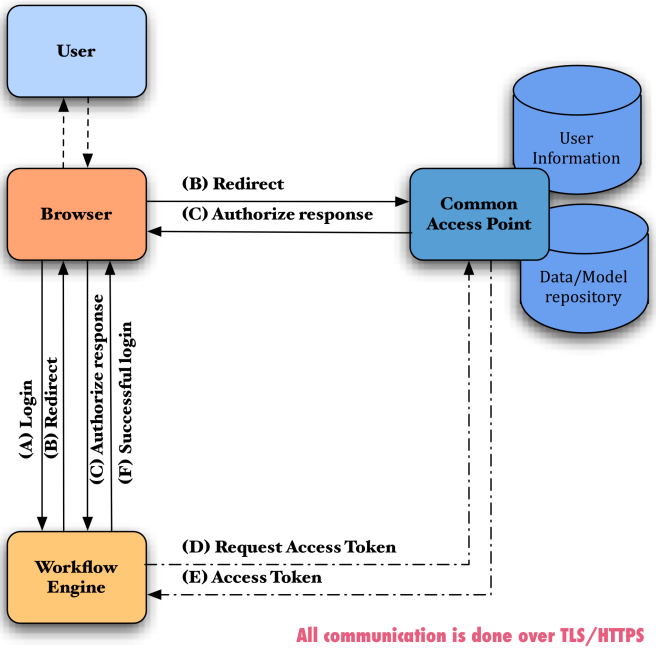


Figure 4 The OAuth-based interactions between the components at user's login

After the authentication phase finishes, the workflow environment retrieves the models from the US and EU model repositories using the designed web services API (see Deliverable 4.1.2). The models themselves are described using TumorML (see Deliverable 4.2.2) that describes their abstract interfaces and also the execution requirements of their published implementations. An example of such descriptions can be seen in Figure 5.

For the linking of models the metadata annotations of their parameters are very important. In addition to the syntactic annotation related to the data type (e.g. double precision numbers, boolean values, etc) the final version of the platform supports also semantic annotation. In particular the TumorML model description also incorporate and integrate with the MIRIAM guidelines[2] in terms of providing links to semantic terms. An example can be shown in Figure 5 where the metadata of a parameter include the MIRIAM URI (`urn:miriam:uniprot:P19174`) of the protein that the parameter refers to.

[2] Le Novère N., Finney A., Hucka M., Bhalla U., Campagne F., Collado-Vides J., Crampin E., Halstead M., Klipp E., Mendes P., Nielsen P., Sauro H., Shapiro B., Snoep J.L., Spence H.D., Wanner B.L. (2005) Minimum Information Requested In the Annotation of biochemical Models (MIRIAM) Nature Biotechnology, 23: 1509-1515.

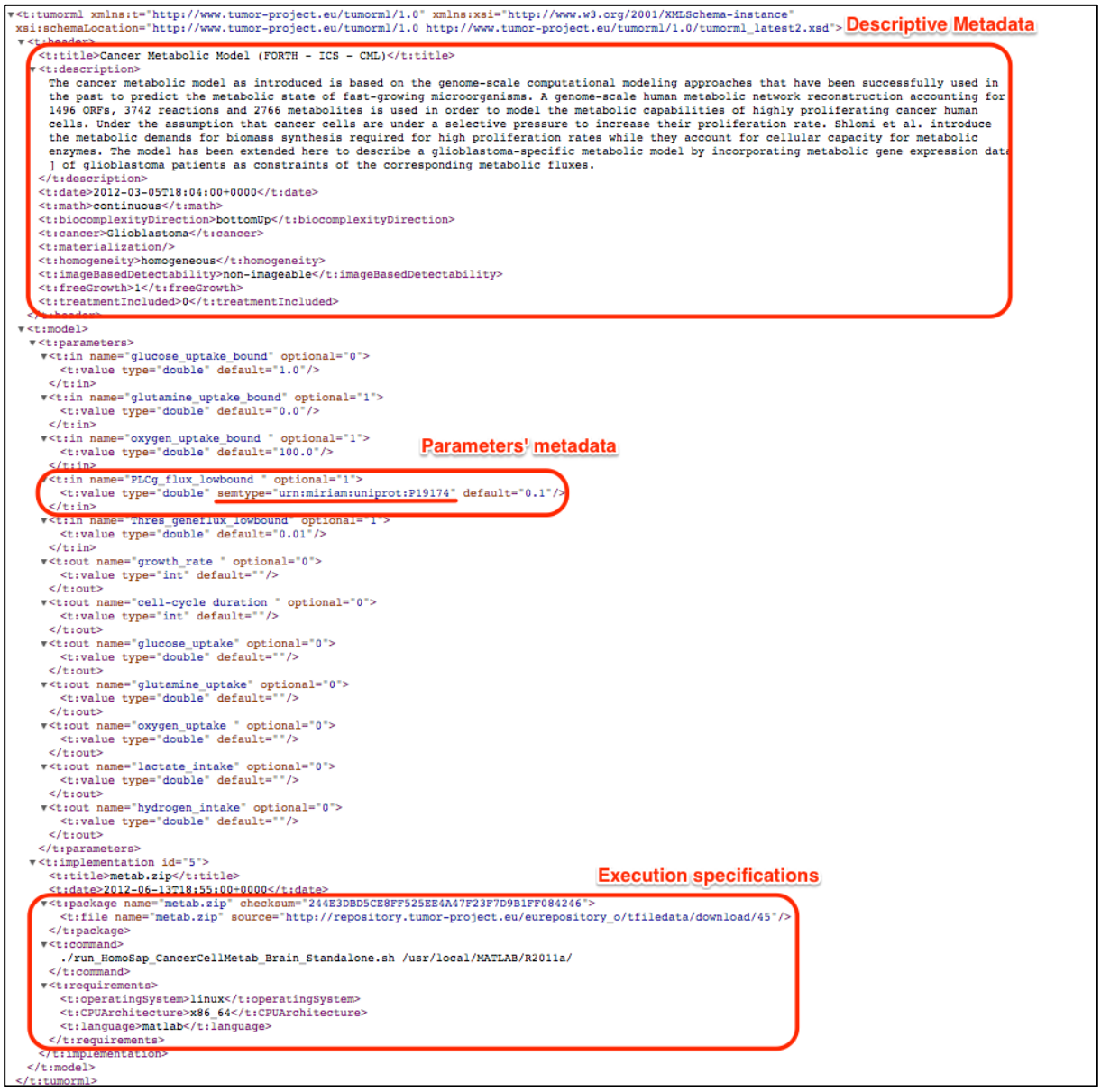


Figure 5 An exemplar model described in TumorML

The execution of a model happens in the backend of the workflow environment effectively relieving the user's local machine from any load. The way a model is executed depends on the type of its implementation. We support two main types: The model can either be implemented as a "command line" tool in whatever language the modeller chooses or it can be specified using the Systems Biology Markup Language (SBML[3]). SBML is the one of the most widely used markup languages to model the molecular and sub-cellular processes.

So there are two main execution frameworks in the TUMOR platform. The one is based on the SBML description of a model whereas the other is more generic in the sense that a model can be provided as a self-contained executable. An SBML description of a model is a declarative artifact. It describes the mathematics required, typically in the form of ordinary differential equations (ODEs), to implement the model and nothing else. In order to implement the model a solver is required to numerically resolve the equations, and execute the corresponding reactions based on the kinetic laws and the prescribed parameter values. This solver can be a simulation environment, a compiler that links the SBML file with numerical library and generates a standalone executable or a partial evaluator that attempts to unfold the ODEs with respect to known solving algorithms. In general the SBML models can be classified as deterministic or

[3] http://sbml.org

stochastic, with the latter using Monte Carlo simulation and related methods. The TUMOR execution infrastructure supports deterministic and stochastic models, through the incorporation of the COPASI simulator[4]. The use of COPASI software allows the parsing of SBML models and their execution but nevertheless there are a couple of parameters that need to specified prior to the execution:

- The simulation time for the model
- The algorithm to be used, e.g. deterministic, stochastic, or hybrid.

These parameters are not specified by SBML but they are essential in order for the models to produce the desired results. In order to support flexibility, the users can input values for both parameters at runtime. These parameter values are then passed to the COPASI solver for simulating the models.

In the more generic case, the model is provided with no information on its internals. The supplied code, either in binary or in source format, should be able to be run as a command line program with its inputs and outputs specified either as command line options or as files. For example, if the execution framework (as in our case) is a Linux 64bit environment, the supplied executable code should be compliant with it. Of course, in the case where the source code of the model is available in the form a scripting language, like Python or Perl, there are fewer restrictions imposed to the model creators.

Irrespective of the models' type (SBML or generic/command line formats), TumorML offers a generic metadata "envelope" to describe both their interface, i.e. input parameters and output results, and execution requirements. The interface definition provides valuable information for *linking* models in the workflow editor, based on the required input and the generated output. On the other hand the execution information is utilized from the workflow's runtime, when the models are simulated or executed.

[4] http://www.copasi.org/

3 A walkthrough of the TUMOR workflow environment

In this section we provide a tutorial and walkthrough of the main functionalities of the workflow editor that was designed and implemented in the context of the TUMOR project. This tutorial has been also provided to the potential users that participated in the evaluation of environment (please see Section 4).

3.1 Introduction

The TUMOR Workflow Environment is a web based graphical user interface for connecting computational models for cancer in a way that they work together by exchanging data. This is the notion of *workflows* or *pipelines*, because the models usually work in a line similar to the "assembly lines" used in the manufacturing industry where the different *actors* (models) work in tandem and in sequence to produce the final result.

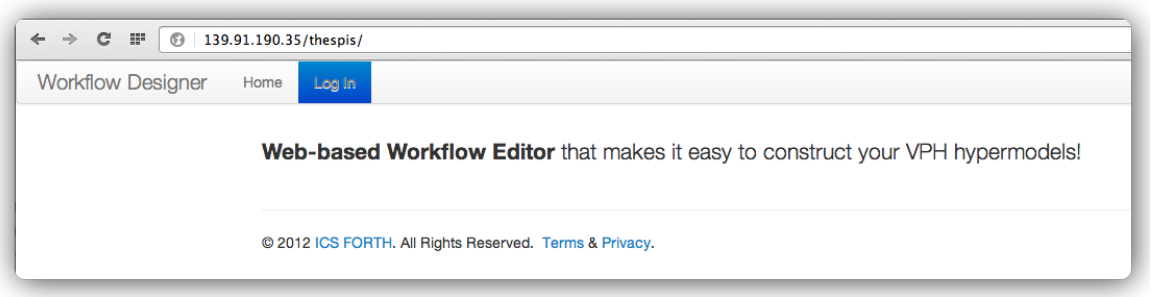
In this tutorial we are walking through the main functionalities of this environment which include:

- Logging into the TUMOR platform and the workflow environment in particular
- Searching and browsing the available models
- Inspecting these models and connecting them, so as to effectively construct multi-model workflows
- Saving and retrieving workflows, and finally
- Executing the workflows, monitoring the execution, and retrieving the results

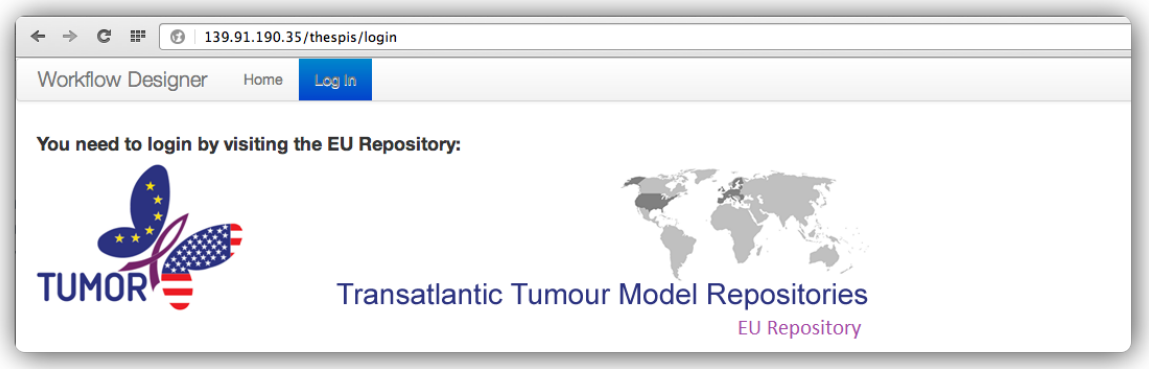
We describe each of these steps in the following paragraphs.

3.2 Logging into the system

The Workflow environment can be accessed at http://139.91.190.35/thespis/

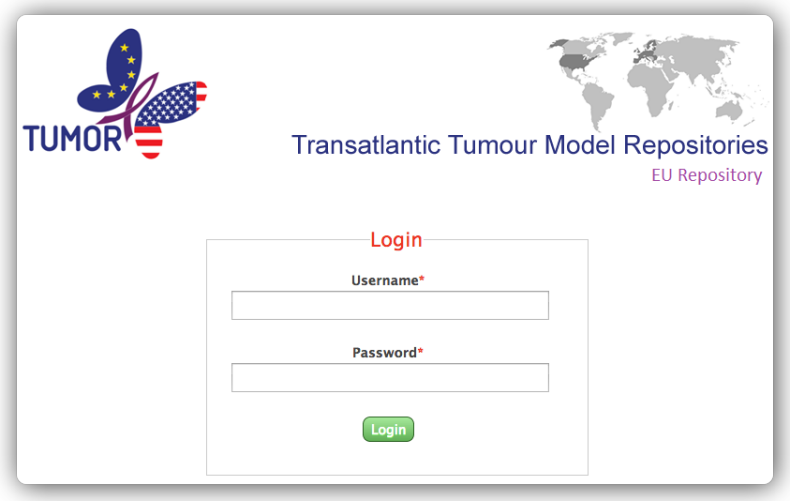


The user needs to "login", by selecting the highlighted menu link, in order to get access to the workflow environment. The login process takes place in another component, the EU Model Repository, which stores and manages the computational models and the user access.



After visiting the EU Model Repository the user is asked to provide her user name and password. For this tutorial the following credentials can be used:

- User name: `guest`
- Password: `tumor`

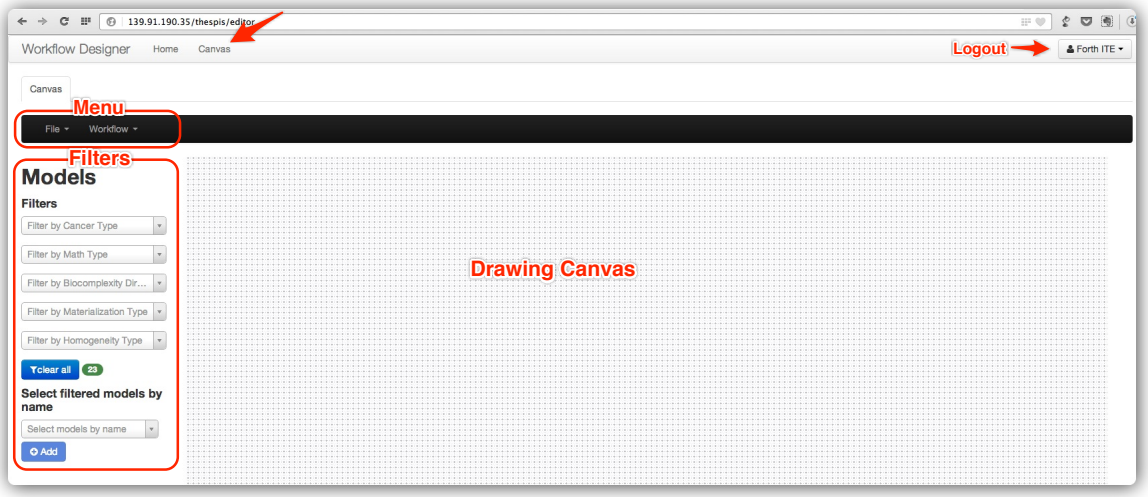


Next the user is asked to allow the Workflow Environment to access the models in the EU model repository on his/her behalf.

After selecting “Yep” the user should be “redirected” back to the workflow environment

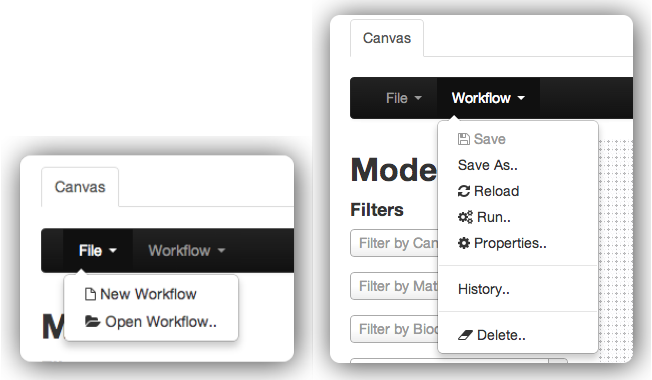
3.3 Model Search and Filtering

After the successful authentication in the EU model repository, the “Canvas” page of the workflow environment appears in the user’s browser. As shown in the next figure the user has access to a canvas specific menu and a filters and search area, while in the middle there’s a “drawing canvas” area. Additionally in the upper right corner there’s a button for logging out the current user.



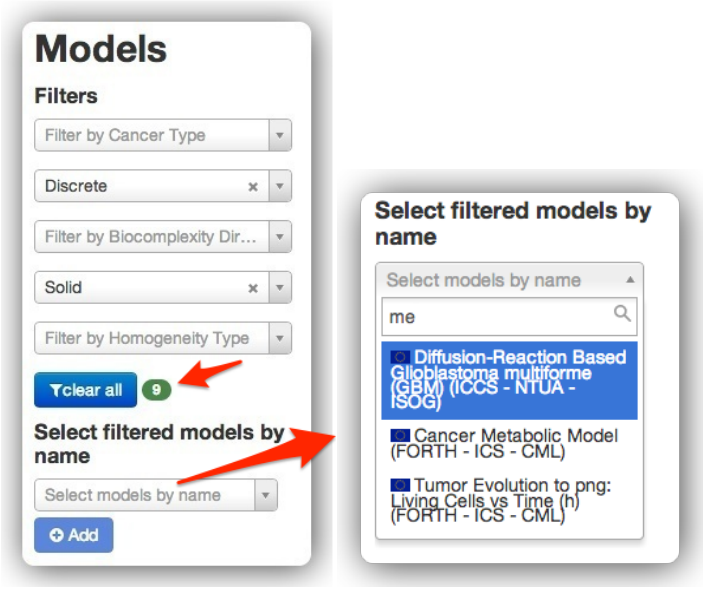
☞*By default the user is initially presented with an empty "canvas", i.e. no workflow is shown, unless the user has already visited the system in a previous login session. In the latter case the previously edited workflow is loaded.*

The menus consists of two main items, the File for opening existing workflows or creating new ones, and the Workflow that includes actions for saving, deleting, running, etc. the current workflow. Please select the "New Workflow" item from the File menu to clear the drawing canvas area and initiate the construction of a new pipeline.

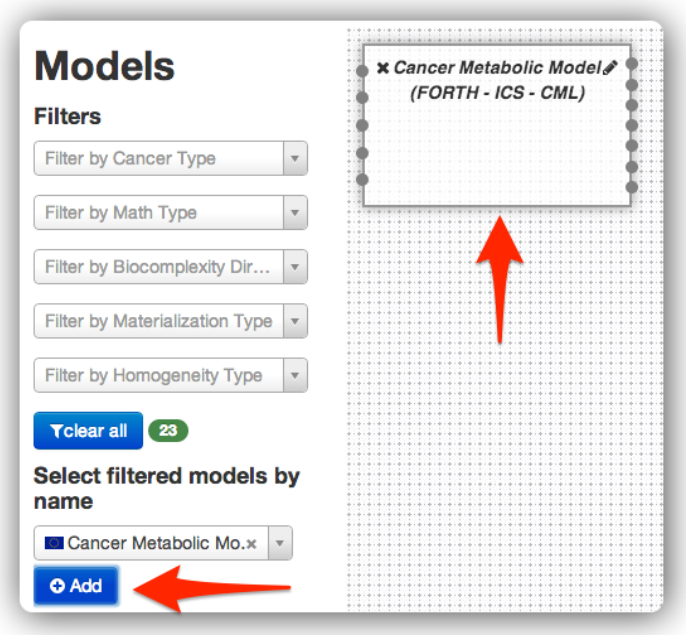


In the filters area the user can provide some filtering criteria, like the Cancer type (e.g. Breast, Lung), the Mathematics type (e.g. Continuous, Discrete), the Materialization type (Solid or Liquid), or the Homogeneity type (homogeneous or non-homogeneous). When the user applies some filters the number of matched models in the repository is shown whereas by pressing the "Clear all" button all filters are removed. Furthermore the user can also search by the models' names. Just by giving a few characters of a model's name the user is able to see the list of matching models.

☞*Please bear in mind that this search-by-name functionality is affected by the other filters so the user may need to "clear all" filters before searching model names in order to include all models as potential matches.*



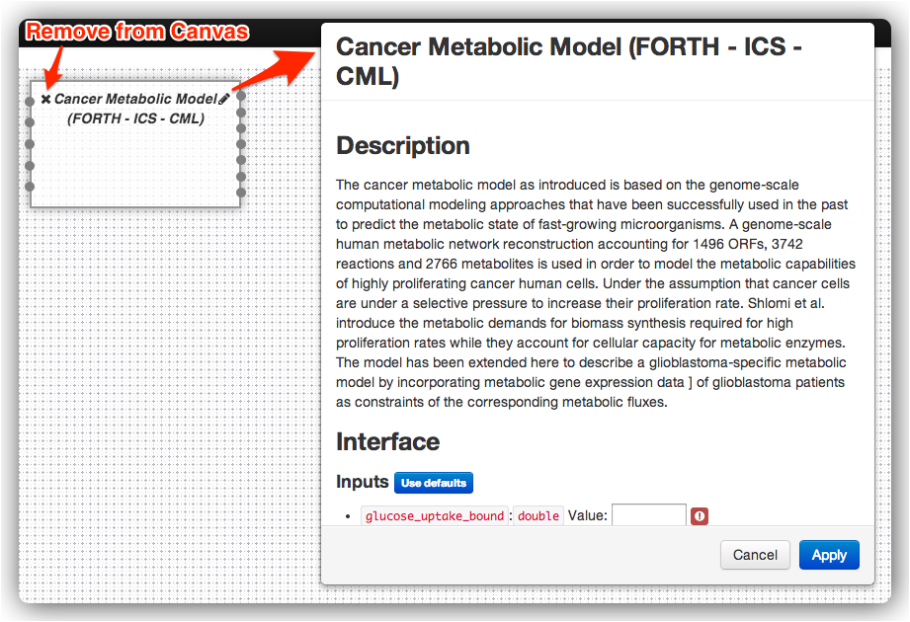
After the user has applied some filtering criteria or given a part of a model's name he/she can selects one of the matching models from the dropdown list and after pressing the "Add" button the model appears as a box in the upper left corner of the drawing area.



3.4 Model Inspection and Linking

The user is able to get more information about a model after it is added in the drawing area. The model's box is shown with a number of "dot" connectors in the left and right side. These are the model's input parameters and data ("what the model needs") and outputs ("what the model provides"). These "dots" serve as connection endpoints for linking models together.

Near the model's name in its container box there are two "handles": the remove "x" icon, which removes the model from the workflow, and the edit "pen" icon. After the user presses the little "pen" in the upper right corner of the model box, he/she can get a detailed description of the model and also information about its interface i.e. its input parameters and outputs.



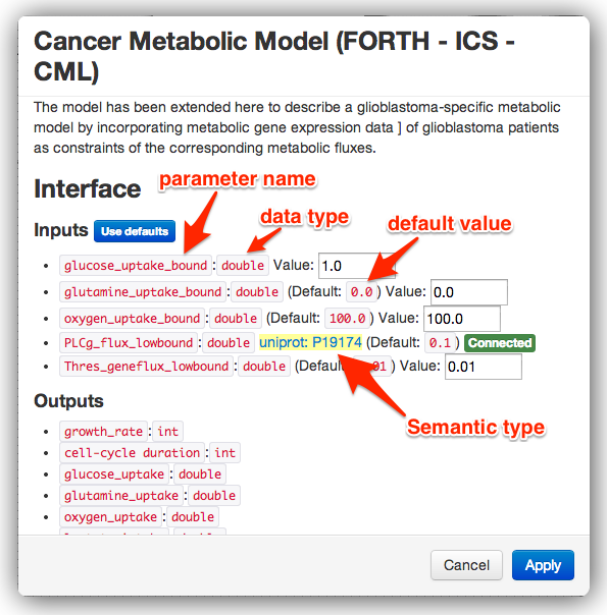
Each input and output parameter has a number of characteristics, such as:

- Name,
- Data type (e.g. “double”, “string” etc.),
- Default value that can be used if not specified otherwise,
- Semantic type, which is a reference to a concept ideally coming from an ontology such as the Gene Ontology (GO) that provides some domain specific annotation to the specific parameter

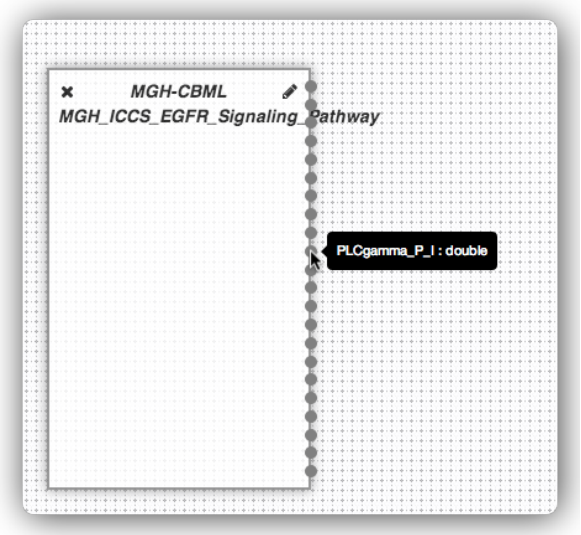
☞ *The data type and semantic type information is of paramount importance to the model linking functionality in the workflow environment. The “semantic type”, wherever exists, is shown as a hyper link to the MIRIAM[5] resolving service[6], that provides human readable information about the linked concept.*

In the same information dialog for a model the user can specify a value for each parameter that will be used when the model is executed in the context of the workflow. An alternative, especially in the case that a model requires lots of parameters, is to press the “Use defaults” button, which means that the default values for all the parameters will be used at execution time.

[5] http://www.ebi.ac.uk/miriam/main/
[6] http://identifiers.org/

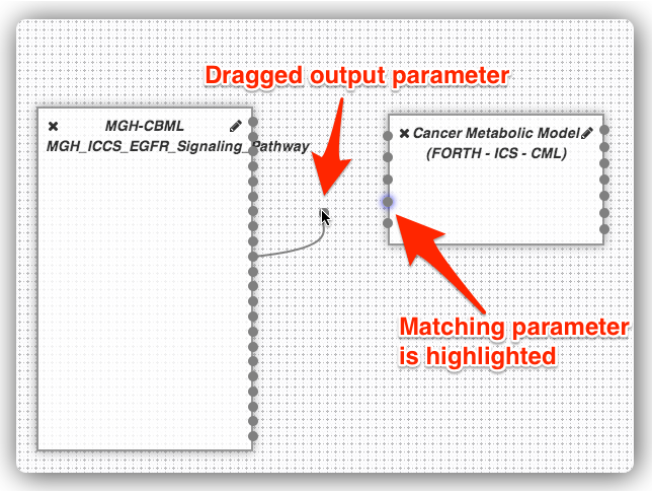


Closing the information dialog of the model (by pressing the Apply button if you want to store the changes or the Cancel button to discard them), the user can have a look in the input or output parameters by "hovering" the mouse pointer over them. A little tooltip appears with the name of the parameter and its data type

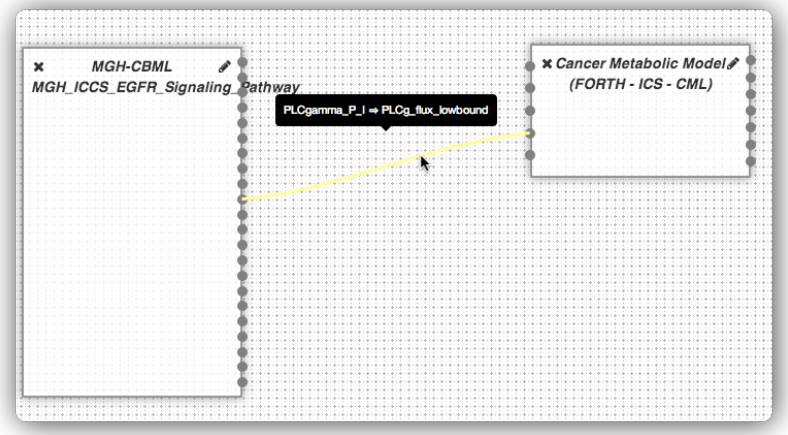


The important functionality of the workflow environment and its "raison d'être" in the first place is to support the linking of the models and the concstruction of multi-model workflows. To this end the user needs to add two models into the canvas as shown in the next figure. The first one is the "EGFR Signaling Pathway" and the second one is the "Metabolic Model" that can easily be found when searched by name. Inspecting the two models the user can see that the `PLCg_flux_lowbound` input parameter of the Metabolic model matches with the `PLCgamma_P_I` output of the EGFR signaling model. The two parameters match both in the data type ("double") and the semantic type (Uniprot P19174[7]). The actual linking of the two models can be done by "dragging" with the mouse the source (output) parameter of the first model and "dropping" it to the target (input) parameter of the second model. During this action input parameters of other models that match (syntactically, i.e. based on the data type, and semantically based on the semantic type) to the "dragged" parameter are highlighted as shown below.

[7] Referring to the "1-phosphatidylinositol 4,5-bisphosphate phosphodiesterase gamma-1" protein, see http://www.uniprot.org/uniprot/P19174



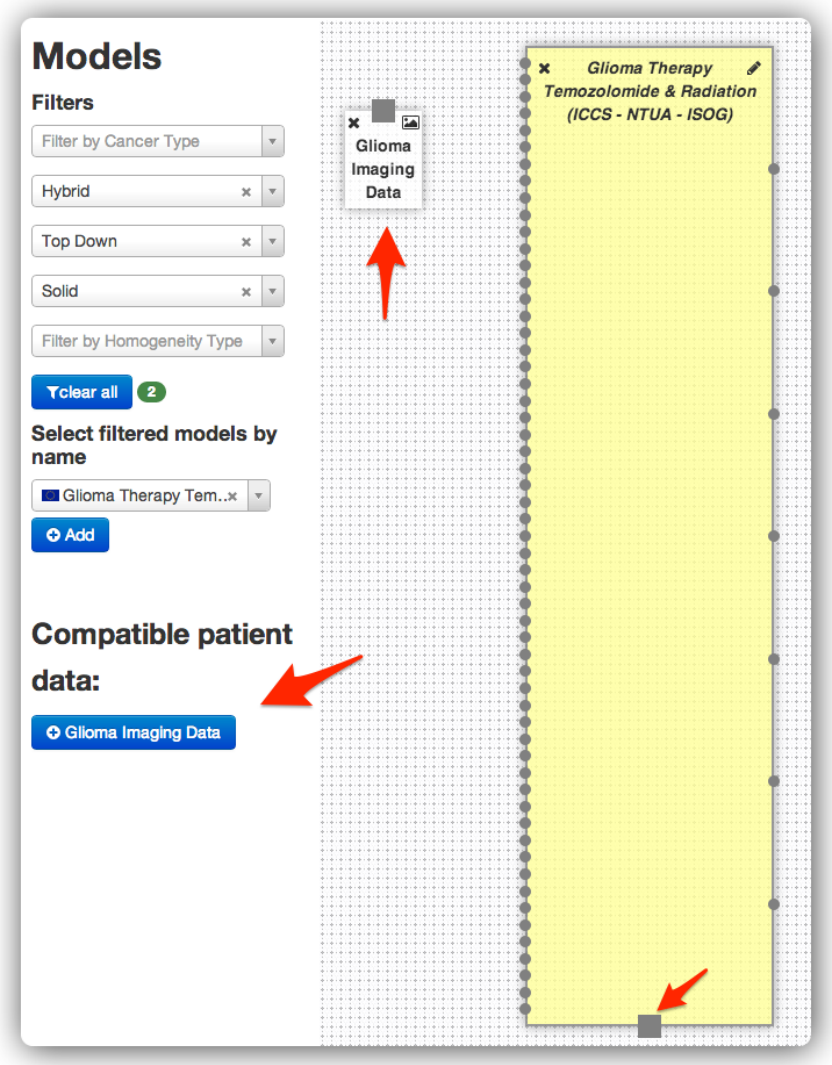
After the “drag and drop” action the two models are now connected and by “hovering” the mouse over the connecting line a small tooltip appears that shows which parameters are connected:



☞*The link connecting two models can be removed by dragging the end of it (in the target (input) parameter) out of its connection with the target model.*

3.5 Populating models with patient data

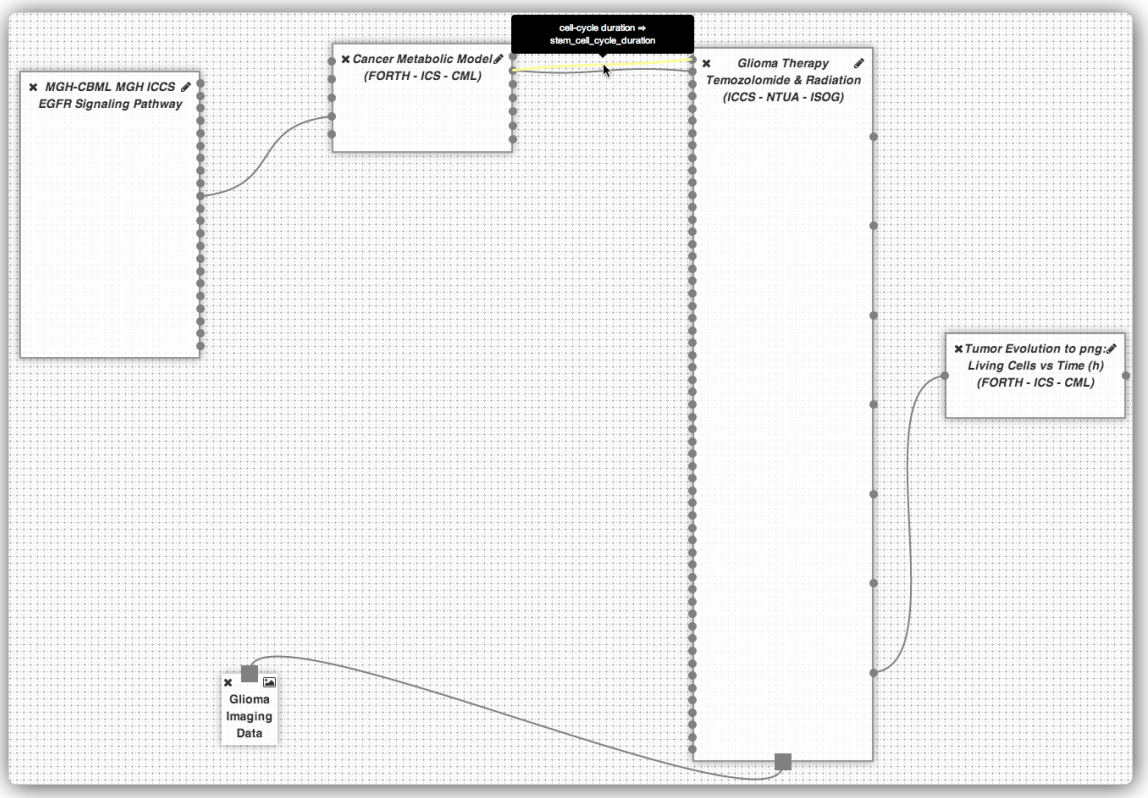
Some models require patient data in order to perform their simulations in the context of a specific patient. An example of this is shown in the next screenshot. The “Glioma therapy” model from ICCS requires patient data as shown with the small square connector in the bottom of the model’s box:



Clicking inside the model's box another button appears on the left side (below the filters) reading "Glioma Imaging Data". When the user clicks this button another box, denoting the patient data, appears in the drawing area with a similarly looking square connector in its top side. This data can be connected to the model in the similar way as described above for linking the models, i.e. by dragging its square connector and dropping it to the corresponding square connector of the model.

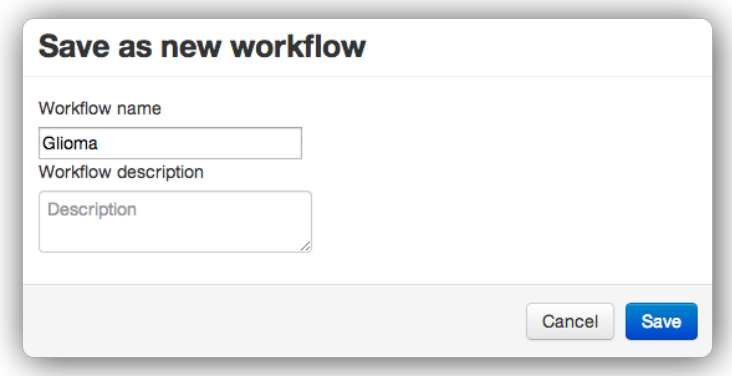
☞*When some patient data are linked with a model it can be the case that some of the model's input parameters take values. The reason for this is that some parameters are constrained to take patient specific values or data specific values (e.g. image size). But even in this case the user is able to overwrite these values in the "edit dialog" of the model (by clinking in the "pen" icon next to the model's name as explained above).*

A complete workflow with 3 models (and a last "visualization" step to produce a PNG image) is shown below. This workflow also features the connection of the same source parameter (the cell-cycle duration output of the Metabolic model) to two different target parameters (the `stem_cell_cycle_duration` and `limp_cell_cycle_duration` inputs of the "Glioma Therapy" model).

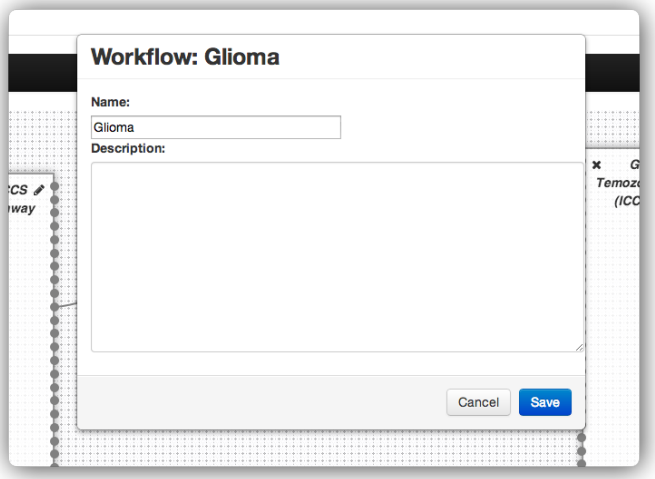


3.6 Saving and loading workflows

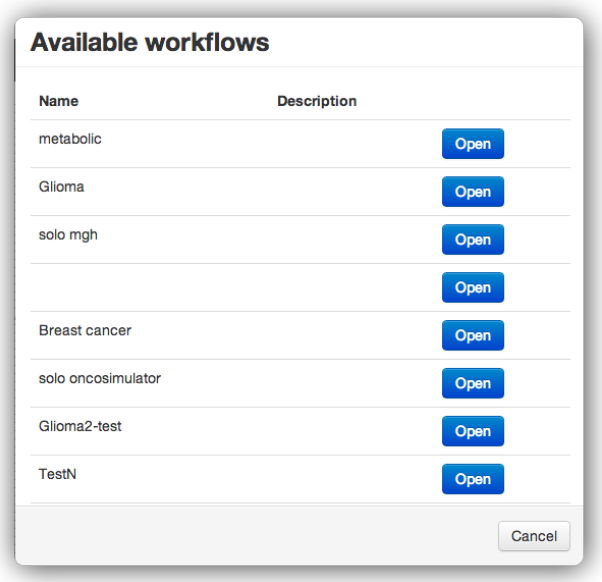
At any time during the workflow editing phase the user can save the workflow. The first time he/she needs to provide a name and (optionally) a description for it. The relevant functionality is available through the Workflow menu (i.e. Workflow > Save as..)



If a workflow has been saved the user can change its name and description from the "Workflow > Properties" menu item or even delete it from the "Delete" menu item.



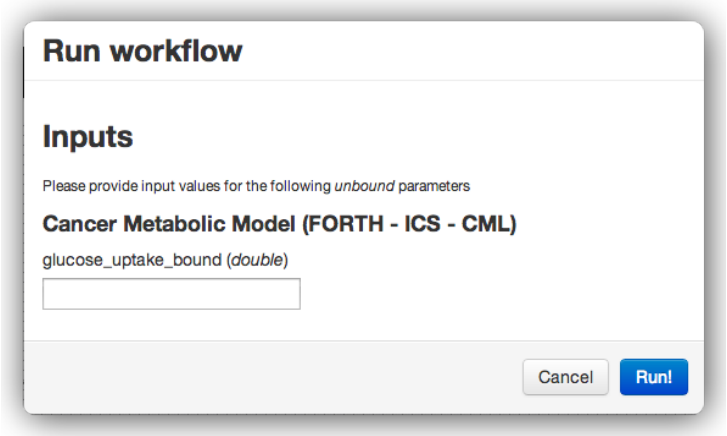
Finally the user can load a previously saved workflow choosing the “File > Open workflow..” menu item.



3.7 Workflow execution and monitoring

The objective of the workflow environment is of course to support the execution of the workflows composed by multiple interlinked models. Therefore after the user has finished the construction of the workflow and has saved it, he/she can run it by choosing the “Workflow > Run..” menu item. Normally the models require values for their input parameters to run and these can be specified in the workflow construction phase as described above, either by linking them to the output of other models or by providing specific values in the models’ edit dialog. In the latter case the values are in fact incorporated into the workflow and cannot be altered unless the workflow is loaded, edited, and saved again.

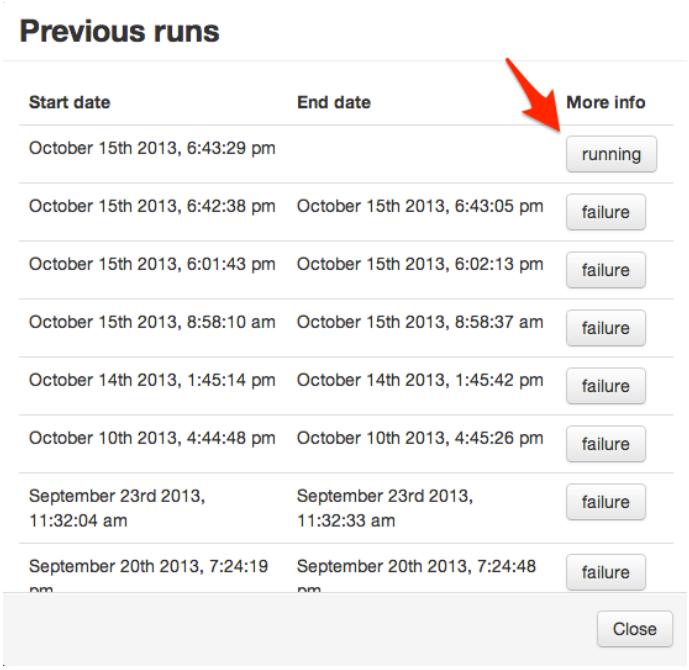
In the case though that there are some “unbound” input parameters for any of the models participating into the workflow, when the user runs the workflow the system will ask him/her to provide values for these parameters, as shown below:



After the user presses “Run” the workflow is assembled in the backend, the data and the models are downloaded from the EU Model Repository, and the execution starts. During the workflow execution the user can have an immediate feedback on the execution status because the currently running models are highlighted at each point in time.

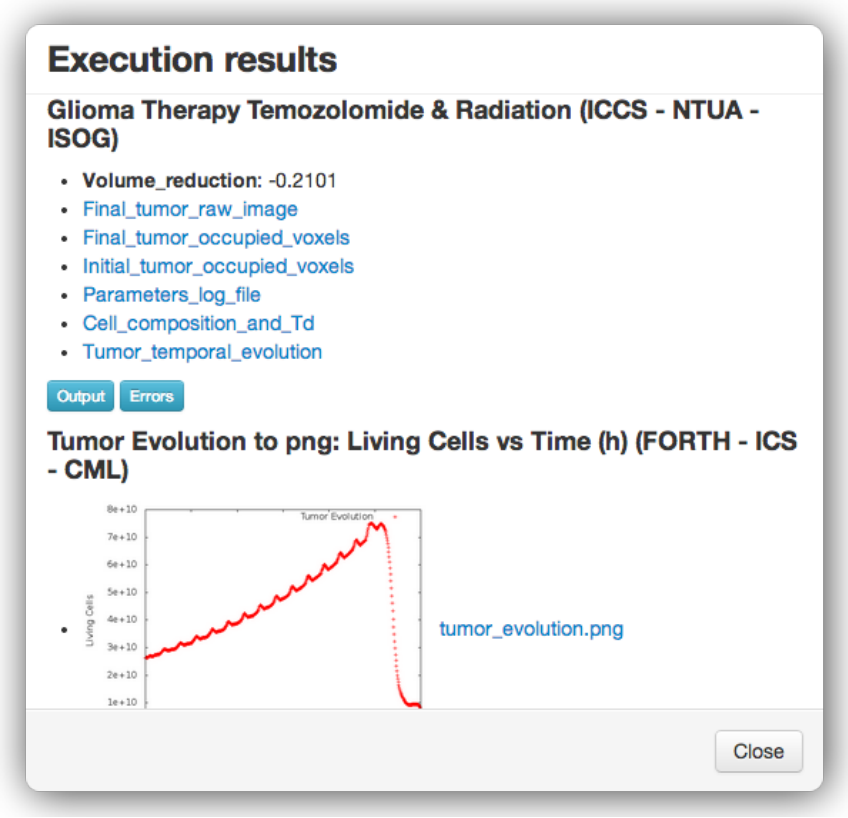
The user can even log out and the workflow will continue its execution in the workflow system’s backend despite that there’s no one monitoring it. The monitoring can be resumed after the user logs in again, opens the very same workflow, and selects the “Workflow > History..” menu item. The user will then be presented with the previous

executions of this workflow and one of them should read “running” under the “More info” column:



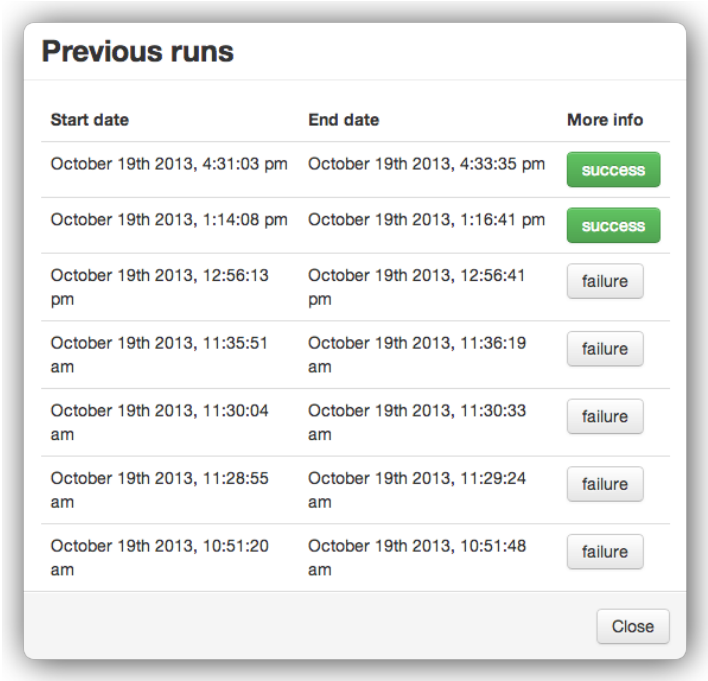
By clicking this “running” button the workflow monitoring should be resumed.

When the workflow execution finishes, the results should be presented as shown next:



The results of each model execution are grouped together and if they are atomic values (like numbers) or images are shown inline. Otherwise they are presented as hyperlinks so that the user can download them in his/her local computer.

The results of the previous runs can also be retrieved from the “Workflow > History...” menu item and by clinking in the corresponding “success” or “failure” buttons:



In the case of successful executions the full result set are shown while in the case of unsuccessful ones the user can see the errors reported by the failed model executions.

4 Evaluation and usability results

This section presents the Evaluation Plan based on a set of usability measurement tools that was used during the usability testing of the TUMOR Workflow prototype. The proposed Evaluation scheme is user-based. As users we consider that Clinicians and Researchers. Having real users test an application increases the effectiveness of usability evaluation and focuses on finding problems that are often overlooked by the developing team. It should be noted thought that finding real users to participate in these evaluations is not an easy task, so the evaluators should try and make the most of their testing. Nevertheless, usability testing is the most fundamental user-based evaluation method because it provides direct input on how the user perceives and interacts with the application or interface that is being tested and on whether the design areas create confusion and other problems to the user. The value of observing a real user trying to go through real case scenarios is truly irreplaceable as the team can have a clear view of the design's shortcomings and successes[8]. Thus, it is imperative that each application is given the chance to be tested against real users.

Evaluation Plan

4.1 Usability testing process

Contrary to common belief, usability testing does not have to be an elaborate and expensive venture. It does not have to happen inside the confines of a lab with expensive equipment and set-ups. We based our evaluation planning on the following steps:

Step 1: Define the test plan. This step involves defining the following:

- **What is going to be tested?** Since the time of the test is limited, the evaluators have to decide on the scope of the test and where the focus is going to be. Is it going to be on the overall design elements, or on specific features of the

[8] Nielsen, J.1993. "Usability Engineering". (pp. 165-205). Academic Press, Boston, MA.

system, navigation, content, or overall usability? Sometimes it is difficult to examine all aspects of a system in one test and that has to be taken into consideration.

- **Why is the test being conducted?** What questions are the evaluators trying to answer?
- **How is the test going to be done?** In this step, details as to how the test is going to be conducted need to be described. Things like how the testers are going to be selected and invited to participate, when is the test going to take place, what is the budget that will be needed to carry it out etc. At this stage of the test plan, the evaluators also need to decide upon the measurements that are going to be used.
  - *The actual tools we used for measuring usability are described on paragraph 4.1.1.*

Step 2: Decide on the location where the test will be conducted i.e., lab, office, remotely, or locally, as well as the equipment that is going to be needed i.e., cameras, software, hardware etc.

- *We conducted the test remotely, since we are referring to a web-based application.*

Step 3: Select the users that are going to participate in the test. Define the profile of the real user of the system that is being tested and then select testers that fit that profile as close as possible.

- *We based our selection on the general rule to select users that are as representative as possible of the intended users of the system. Hence we included Research oriented Clinicians and Modelers.*

Step 4: Prepare test materials. Have printed copies of all the documents that are going to be used during the test, i.e., the list of the tasks to be tested and the questionnaires that users may need to fill out before or after the test.

- *We based our evaluation on the specified tasks described in* Appendix A – User Success Rate Form*. All users have been provided with a detailed walkthrough (see Section 3).*

Step 5: Conduct the tests. The main stages we selected are a subset of those described by Jacob Nielsen:

- **Preparation**: Make sure the software that is tested is working properly, and that all the test materials are ready.
- **Introduction:** At this stage the experimenter welcomes and thanks the tester for accepting to participate in the test and gives a brief explanation of the purpose of the test. Then the experimenter proceeds with explaining the testing process. It is important to clarify to the tester at this point that the evaluation is done on the system and not on his/her performance and that all the acquired data will be treated as confidential information.
- **Running the Test**: During the test, the experimenter remains uninvolved and refrain from making comments, observations, or expressing any opinions on the interface or the system.

Step 6: Analyze data and observations. At this stage we went over the data acquired from the measurement tools that are being described in Section Usability Measurement Tools.

Step 7: Create recommendations. Based on the gathered data from Step 7, the usability specialists will create a list of recommendations on what needs to be changed

in order to make the system more user-friendly and accessible and solutions to problems that were observed. The results will be then shared and discussed with the rest of the development team.

4.1.1 Usability Measurement Tools

4.1.1.1 User Success Rate

User success rate is a very simple usability measure introduced by Jacob Nielsen[9] that measures the overall effectiveness of the system. It is a coarse metric that calculates the percentage of tasks that users complete correctly. Even though it doesn't say anything about why users fail or how well they perform the tasks, it is a quick and easy way to get the bottom line of usability, the success rate. **Appendix A** – User Success Rate Form includes a User Success Rate empty form that was used as a print-out.

The way we calculated the output us the following:

Step 1: Divide the test in N tasks and watch the users trying to complete them.

Step 2: Mark each user attempt with S = success, F = failure, P = partial success

- **Mark with S (success):** those who manage to complete a task.
- **Mark with P (partial success):** those who completed the task, but with some errors, or omissions, or only after they asked for help. There is no firm rule for assigning credit for partial success. Partial scores are only estimates, but they still provide a more realistic impression of design quality than an absolute approach to success and failure. Partial credits get a 50% mark.
- **Mark with F (failure),** those who simply didn't manage to complete the task or were even close at doing so.

Step 3: Calculate the success rate:

Success Rate = (TS + (PS*0.5))/Number of attempts

TS = Total number of successful attempts

PS = Total number of partial success attempts

Example: Assume there are 6 tasks and 4 users who received the following marks:

	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6
User 1	F	P	S	S	F	S
User 2	F	F	P	F	F	S
User 3	F	S	S	P	P	S
User 4	P	S	S	P	P	S

TS = Total number of successful attempts = **10**

PS = Total number of partial success attempts = **7**

Total **Success Rate** = (10 + (7*0.5))/24 = **56%**

4.1.1.2 System Satisfaction Scale Questionnaire (SUS)

This is another measuring tool, which calculates the overall system satisfaction. It is using a 5-point Likert scale (1=strongly disagree and 5=strongly agree). This questionnaire is answered by the users right after the test is finished when their experiences from the interaction with the system is still fresh in their minds.

Here are the questions asked in SUS (*see* Appendix A – User Success Rate Form**)**

[9] Nielsen, J.1993. "Usability Engineering". (pp. 165-205). Academic Press, Boston, MA.

1. I think that I would like to use this system frequently
2. I found the system unnecessarily complex
3. I thought the system was easy to use
4. I think that I would need the support of a technical person to be able to use this system
5. I found the various functions in this system were well integrated
6. I thought there was too much inconsistency in this system
7. I would imagine that most people would learn to use this system very quickly
8. I found the system very cumbersome to use
9. I felt very confident using the system
10. I needed to learn a lot of things before I could get going with this system

How to score SUS

To calculate the SUS score, first sum the score from each question. Each question's score will range from 0 to 4.

Step 1: For questions 1,3,5,7,and 9 the score is the **scale** position minus 1.

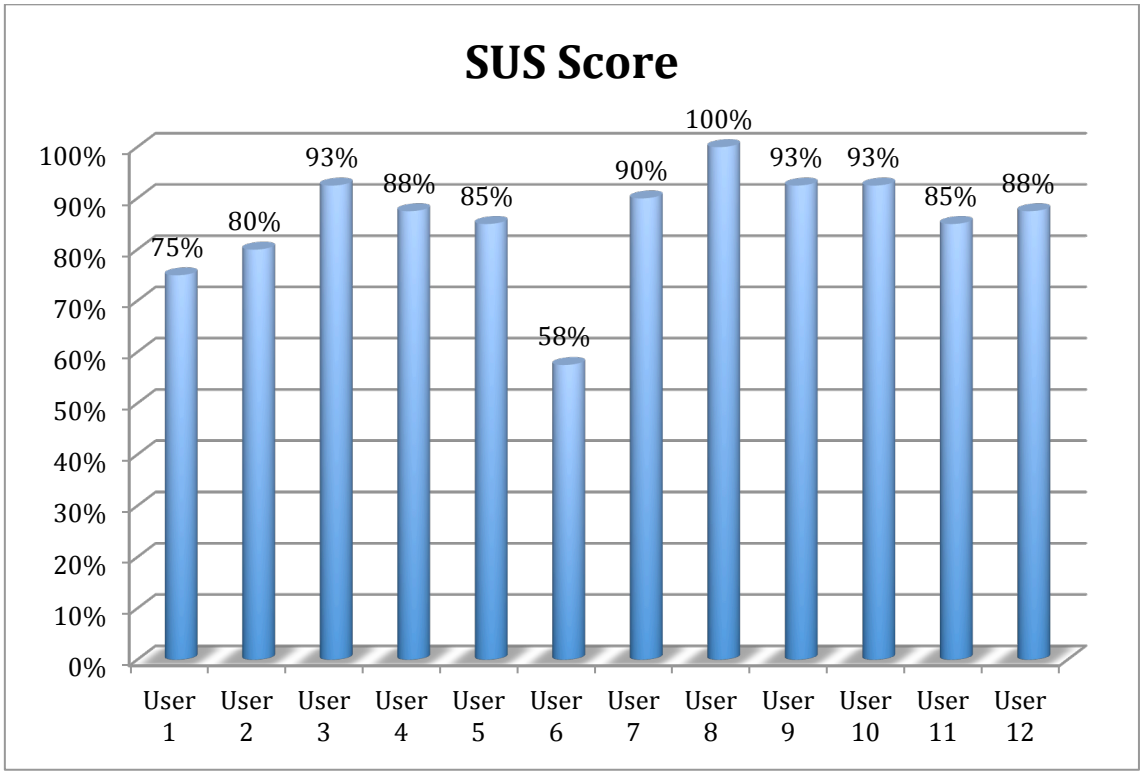
Step 2: For questions 2,4,6,8 and 10, the contribution is 5 minus the **scale** position.

Step 3: Multiply the sum of the scores by 2.5 to obtain the overall value of SU.

4.2 Evaluation results

According to the methodology presented in the previous section we included 12 users (8 researchers and 4 clinicians) who completed the evaluation process based on the User Success Rate and System Satisfaction Scale Questionnaires. The outcome provided by the two user groups (the Researchers and the Clinicians) did not deviate significantly to require separate usability testing schemes and trials. Hence, the results are tabulated and presented together.

4.2.1 System Satisfaction Scale Results (SUS)



According to the SUS results the overall system satisfaction was acceptable reaching more than 85% in most of the cases. The users felt confident enough with the system, but there was a single case (User 6) where technical support was desirable.

4.2.2 User Success Rate Results (SAS)

The results are tabulated in the following table based on the following tasks (see Appendix A):

Task 1: **Login**

Task 2: **Model Search & Filtering**

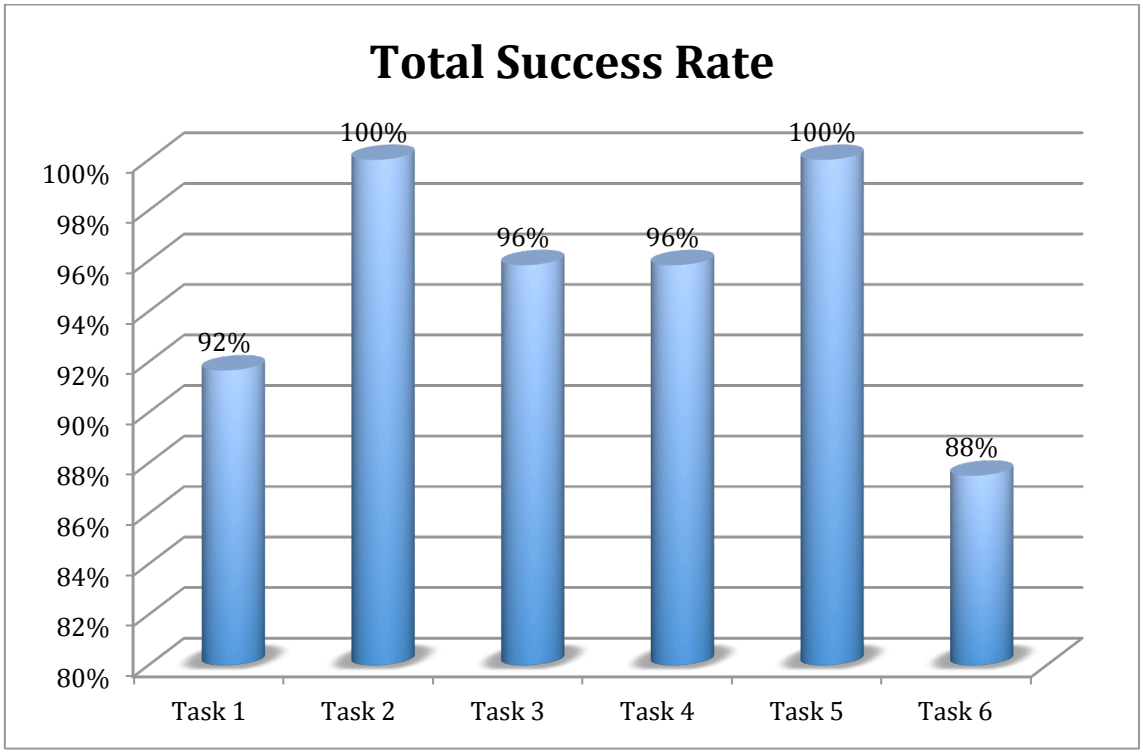
Task 3: **Model Inspection & Linking**

Task 4: **Populating models with patient data**

Task 5: **Saving and Loading Workflows**

Task 6: **Workflow execution & Monitoring**

	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6
User 1	PS	S	S	S	S	PS
User 2	S	S	S	PS	S	S
User 3	S	S	S	S	S	S
User 4	S	S	S	S	S	S
User 5	P	S	S	S	S	S
User 6	S	S	S	S	S	S
User 7	S	S	S	S	S	S
User 8	S	S	S	S	S	P
User 9	S	S	S	S	S	S
User 10	S	S	P	S	S	P
User 11	S	S	S	S	S	S
User 12	S	S	S	S	S	S
TS	10	12	11	11	12	9
PS	2	0	1	1	0	3
Total Success Rate	92%	100%	96%	96%	100%	88%



User success rate measures the overall effectiveness of the system. It calculates the percentage of the aforementioned tasks that users completed correctly. More specifically, it is noted that even if the scores were well above 85%, there was a “difficulty” during the login process (Task 1) and the execution/ monitoring task (Task 6). The first case pointed out that the authentication mechanism involving both the Repository and the Workflow systems is rather confusing for the user. It should be more smooth and transparent. The second case was caused by a technical problem where the user did not have immediate feedback for the results of the workflow execution due to a timeout in the communication between the browser and the server. The results were available on the server side but the user needed to reload the application in his browser to retrieve them.

Both issues have now been resolved.

Appendix A – User Success Rate Form

Based on the following tasks

Task 1: **Login**

Task 2: **Model Search & Filtering**

Task 3: **Model Inspection & Linking**

Task 4: **Populating models with patient data**

Task 5: **Saving and Loading Workflows**

Task 6: **Workflow execution & Monitoring**

Mark with S (success): those who manage to complete a task.

Mark with P (partial success): those who completed the task, but with some errors, or omissions, or only after they asked for help.

Mark with F (failure), those who simply didn't manage to complete the task or were even close at doing so.

User Success Rate						
	Task 1	**Task 2**	**Task 3**	**Task 4**	**Task 5**	**Task 6**
User 1						
User 2						
User 3						
User 4						
User 5						
Calculate Success Rate = (TS + (PS*0.5))/Number of attempts						

TS = Total number of successful attempts

PS = Total number of partial success attempts

Appendix B–System Satisfaction Scale Questionnaire (SUS)

This questionnaire should be given to the user at the end of test.

System Satisfaction Scale Questionnaire					
1=Strongly disagree, 5=Strongly Agree					
I think that I would like to use this system frequently	1	2	3	4	5
I found the system unnecessarily complex	1	2	3	4	5
I thought the system was easy to use	1	2	3	4	5
I think that I would need the support of a technical person to be able to use this system	1	2	3	4	5
I found the various functions in this system were well integrated	1	2	3	4	5
I thought there was too much inconsistency in this system	1	2	3	4	5
I would imagine that most people would learn to use this system very quickly	1	2	3	4	5
I found the system very cumbersome to use	1	2	3	4	5
I felt very confident using the system	1	2	3	4	5
I needed to learn a lot of things before I could get going with this system	1	2	3	4	5