ViroLab Virtual Laboratory

Tomasz Gubała[1,2], Bartosz Baliś[3], Maciej Malawski[3], Marek Kasztelnik[1], Piotr Nowakowski[1], Matthias Assel[4], Daniel Harężlak[1], Tomasz Bartyński[1], Joanna Kocot[1], Eryk Ciepiela[1], Dariusz Król[1], Jakub Wach[1], Michał Pelczar[1], Włodzimierz Funika[3], Marian Bubak[1,3]

[1] Academic Computer Center CYFRONET, ul. Nawojki 11, 30-950 Kraków, Poland
[2] Section Computational Science, Informatics Institute, University of Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
[3] Institute of Computer Science, AGH, al. Mickiewicza 30, 30-059, Kraków, Poland
[4] HLRS High Performance Computing Center Stuttgart, University of Stuttgart, Nobelstrasse 19 70550 Stuttgart, Germany
email: gubala@science.uva.nl

Abstract

The ViroLab Virtual Laboratory presented in this work brings a platform of cooperation for scientists of multiple expertise and locations on common scientific goals. The main objective is to deliver an environment that could combine efforts of computer scientists, virology and epidemiology experts and experienced physicians to support future advances in HIV-related research. The paper explains the challenges of building modern, inter-organizational platform to support science and gives the overview of solutions to these challenges. The examples of real-world problems being applied in the presented environment are also described to prove the feasibility of the solution.

Keywords: e-Science, collaborative applications, virtual laboratory, ViroLab, grid

1 Introduction

Today scientific research frequently appears as a joint effort of multiple scientific teams and institutions. The integration of those parties usually takes place in a form of scientific projects and the process of collaboration takes place in different channels, both direct and indirect, virtual. In order to reach their scientific goal the researchers also use computational models and simulations - in fact, more and more of their tools, documents, results and contacts are performed, stored and shared in computers. The purpose of a common, shared environment for scientists comes from the evolution of tools and methods of today's science. The important investigations are made no longer by isolated individuals, most of them not even by single institutions or laboratories. In many cases different research organizations need to join their effort and expertise for the most complex tasks. Therefore a cross-organizational, well-integrated space is needed. Such environment should allow scientists discuss common goals, plan collaborative tasks to reach these goals, perform the tasks combining their knowledge with

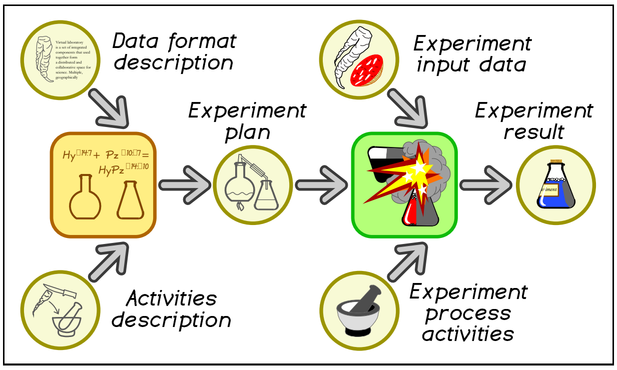

Fig. 1: Conceptual view of the experiment pipeline.

complex simulations running on powerful computers and then to share, evaluate and document results. This work presents the vision, design and prototype implementation of a virtual laboratory that would serve as such a collaborative environment for eScience [15].

The main strength of the virtual laboratory comes from the support for different types of users and mechanisms to integrate their tasks together (like in reality different workers are required for a laboratory to function). The presented system provides means for a scientific programmer to plan in-silico experiments and for a scientist to use these experiment plans to perform complex research. Therefore a set of dedicated tools need to be provided (as different types of users need specific, targeted approach) in an integrated fashion (as they all still work together in the same research endeavor).

2 Architecture and design of the virtual laboratory

The centerpiece of the presented virtual laboratory solution [6] is the concept of the experiment pipeline (see Fig. 1). The two main subjects of cooperative efforts in the laboratory are experiments and their results. Here we distinguish two classes of users: the experiment developers who use their technical skills and the knowledge of the modelled domain to plan new simulation, to search for data sources and to combine them together into new in-silico experiments. The other group, so-called experiment users, are people who actually use previously prepared experiments and execute them using real input data to to obtain specific scientific results.

For a successful virtual experiment three main factors are needed: the input data that is being analyzed, the analytic modules that perform some task of that data (simulation, transformation, inference, etc.) and the ”glue” logic that combines these elements into a well-defined process of specific experiment. Among other components (see Fig. 2 for the entire architecture), the laboratory provides the Data Access Service [2] to obtain important HIV-related data from

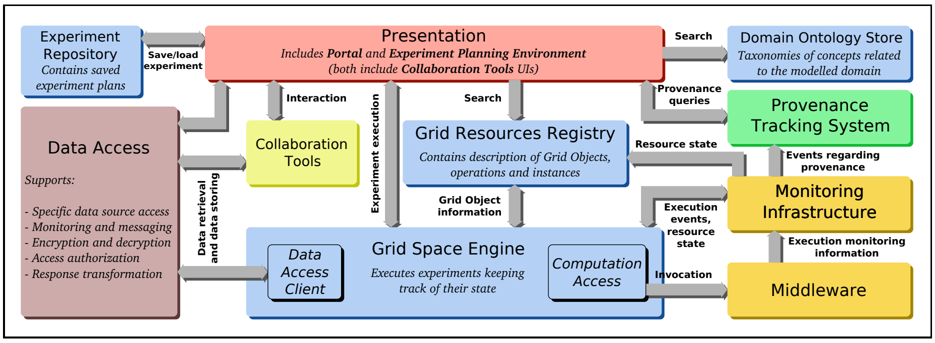
Experiment Repository
Contains saved experiment plans
Save/load experiment
Presentation
Includes Portal and Experiment Planning Environment
(both include Collaboration Tools UIs)
Search
Domain Ontology Store
Taxonomies of concepts related to the modelled domain
Provenance queries
Provenance Tracking System
Data Access
Supports:
- Specific data source access
- Monitoring and messaging
- Encryption and decryption
- Access authorization
- Response transformation
Interaction
Collaboration Tools
Experiment execution
Search
Grid Resources Registry
Contains description of Grid Objects, operations and instances
Resource state
Events regarding provenance
Monitoring Infrastructure
Data retrieval and data storing
Grid Object information
Execution events, resource state
Execution monitoring information
Grid Space Engine
Executes experiments keeping track of their state
Data Access Client
Computation Access
Invocation
Middleware

Fig. 2: The components that constitute the ViroLab Virtual Laboratory

remote databases (e.g. hospitals) and the Grid Operation Invoker [4] to call remote computing elements that perform particular tasks on the data. Since many sophisticated tools (simulators) require significant computing power the laboratory provides access to both powerful computational testbeds using the Grid technology (like the EGEE project [8] testbed) and stand-alone, specialized computation servers (using MOCCA [13] or WebServices technologies). Moreover, a specialized, built-in optimization module (GridSpace Application Optimizer [12]) takes care of optimal load-balancing on smaller computation servers and machines. As the notion of experiment repeatability and the provenance of results is very important for scientists the provenance tracking, recording and publishing system PROToS [3] is also provided in the presented platform. Users are able to find past experiments, browse archived data and track origins of certain experiment results.

Since the system is fairly complex there are dedicated tools to help the users to perform their tasks. The Experiment Planning Environment is an Eclipse-based development tool that aims specifically at supporting quick yet not restricted process of experiment plan development: it uses the Ruby scripting language as the main development platform for the future experiments. On the other hand, the scientists use their web browsers to access the web-based Experiment Management Interface where one may load an experiment, execute it and collect results for off-line analysis. This interface is accompanied by a QUaTRO [5] query tool that allows the use of the provenance system to ask for past experiments and their results. A set of web-based collaboration tools that help scientists exchange such results, observations and opinions is also being prepared to enrich the environment functionality.

In order to achieve the sufficient level of integration of all these components, several additional modules are in place. The Experiment Repository stores experiment plans prepared by developers and published for scientists. The GridSpace engine [11] provides the experiment execution capabilities and the Grid Resources Registry holds information on what computing elements are available and where. The planned Laboratory Data Base will hold the results

```
patientID = DataRequester.new.getData("Provide patient\'s ID")
region = DataRequester.new.getData("Region (\"rt\" or \"pro\")")

nucleoDB = DACConnector.new("das",
   "angelina.hlrs.de:8080/wsrf/services/DataAccessService","","","")
sequences = nucleoDB.executeDistributedQuery(
   "select nucleotides from nt_sequence where
    patient_ii=#{patientID.to_s};")

mutationsTool = GObj.create("RegaDBMutationsTool")
mutationsTool.align(sequences, region)
mutations = regaDBMutationsTool.getResult

drs = GObj.create("DrugResistanceService")
puts drs.drs("retrogram", region, 100, mutatations)
```

Fig. 3: Listing of a sample experiment plan.

obtained by scientists so they are not forced to save them locally on their own in order to use them later.

3 Example virology experiment in the virtual laboratory

The development of the virtual laboratory is quite advanced and the first versions of the core modules and the end-user tools are already released [7]. The developers are closely cooperating with virology scientists and scientific programmers to steadily improve the quality and usability of the platform. The current prototype was already used to plan and execute important virology experiments.

One such experiment is the analysis of a certain HIV virus genotype information in order to find its specific characteristics and therefore to be able to better advise in the process of treatment.

Fig. 3 shows the experiment plan. It is divided into four sections. It starts with some interaction with the user (probably a clinical virologist) who provides the anonymized ID of the target patient and the region of interest (the protein) to be analyzed. The second part connects to the federated database querying system to perform a distributed query over multiple data sources and to retrieve appropriate HIV sequences. Later on, the experiment contacts a mutation detection tool in order to learn what specific mutations the virus in the target patient evolved and, finally, on the basis of those mutations the experiment contacts the Drug Resistance Service to learn how the mutations influence treatment possibilities.

4 Related work

This sections presents a very short summary of the works that inspired the authors with certain ideas and solutions.

The LEAD project [9] is an example of a virtual laboratory for weather applications. The LEAD environment features a portal with user interfaces (UI) and a set of dedicated, distributed Grid resources (both computational and data-related) that are available through those UIs. The underlying workflow system allows for combining the present resources together to define task-specific processing. The system is provided for weather analysts.

Another interesting example of an experimentation space is the Kepler [1] system. Kepler provides a powerful tool to compose application workflows (that could be, in particular, experiments) out of local and remote parts. Composed workflows could be executed under specific execution regime with an underlying enactment engine.

In successful environment of MyGrid's [10] a bioinformatician uses the Taverna to compose complex experiment processes out of smaller, atomic building blocks. The rich library of those basic elements allow for great flexibility and numerous different solutions. The composed processing schemes are executed by Taverna (the workflow engine) and yield results. The collaborative aspect of this solution is being investigated in the MyExperiment project [14].

5 Summary and planned developments

As one may see the ViroLab Virtual Laboratory is prepared in a way that allows to build complex, distributed and powerful experiments in a relatively small amount of code. The developers may combine different data sources and data analysis modules and provide them for collaborating scientists to perform interesting experiments.

New functionality in future versions of the virtual laboratory, among others, will include the mechanisms for resources and experiment monitoring to gather better status and provenance information, an experiment result management system that helps scientists to store, revoke, share and comment on experiment results and wider support for high-performance computing platforms like EGEE or DEISA. As the ideas and design concepts that are the basis for this laboratory are more general, the future applications in other fields of eScience are possible and planned.

Acknowledgements This work was partly funded by the European Commission, Project ViroLab IST-027446, the related Polish grant SPUB-M and the Foundation for Polish Science.

References

1. Ilkay Altintas, Efrat Jaeger, Kai Lin, Bertram Ludaescher, and Ashraf Memon. A web service composition and deployment framework for scientific workflows. *ICWS*, 0:814, 2004.
2. M. Assel, B. Krammer, and A. Loehden. Management and access of biomedical data in a grid environment. In *Proceedings of Cracow Grid Workshop 2006*, pages 263–270, 2007.

3. B. Balis, M. Bubak, and J. Wach. Provenance tracking in the virolab virtual laboratory. In *Proceedings of the 7th Int. Conf. on Parallel Processing and Applied Mathematics PPAM07*. Lecture Notes on Computer Science, Springer, 2008. to appear.
4. T. Bartynski, M. Malawski, T. Gubala, and M. Bubak. Universal grid client: Grid operation invoker. In *Proceedings of the 7th Int. Conf. on Parallel Processing and Applied Mathematics PPAM07*. Lecture Notes on Computer Science, Springer, 2008. to appear.
5. B.Balis, M.Bubak, and J.Wach. User-oriented querying over repositories of data and provenance. In *Proceedings of the e-Science 2007 Conference*, page to appear. IEEE CS Press, 2007.
6. M. Bubak, T. Gubala, M. Kasztelnik, M. Malawski, P. Nowakowski, and P.M.A. Sloot. Collaborative virtual laboratory for e-health. In *Expanding the Knowledge Economy: Issues, Applications, Case Studies, eChallenges e-2007 Conference Proceedings*, pages 537–544. IOS Press, 2007.
7. The ViroLab Project Consortium. The virolab virtual laboratory website, 2007. http://virolab.cyfronet.pl/.
8. EGEE Project. Website, 2006. http://public.eu-egee.org/.
9. K.K. Droegemeier et al. Service-oriented environments in research and education for dynamically interacting with mesoscale weather. *IEEE Computing in Science and Engineering*, (Nov-Dec), 2005.
10. R. Stevens et.al. Exploring williams-beuren syndrome using mygrid. *Bioinformatics*, 1(20):303–310, 2004.
11. T. Gubala and M. Bubak. Gridspace - semantic programming environment for the grid. In *6-th International Conference on Parallel Processing and Applied Mathematics PPAM'2005*, volume 3911, pages 172–179. Lecture Notes in Computer Science, Springer-Verlag, 2006.
12. J. Kocot, I. Ryszka, and M. Bubak. Optimization of grid application execution. Master's thesis, AGH University of Science and Technology, Krakow, 2007.
13. Maciej Malawski, Marian Bubak, Michal Placek, Dawid Kurzyniec, and Vaidy Sunderam. Experiments with distributed component computing across grid boundaries. In *Proceedings of the HPC-GECO/CompFrame workshop in conjunction with HPDC 2006*, Paris, France, 2006.
14. myExperiment Project Team. myexperiment website, 2007. http://myexperiment.org/.
15. Peter M.A. Sloot, Alfredo Tirado-Ramos, Ilkay Altintas, Marian Bubak, and Charles Boucher. From molecule to man: Decision support in individualized e-health. *Computer*, 39(11):40–46, 2006.