A Web-Based Development Environment for Collaborative Data Analysis

M Erdmann, R Fischer, C Glaser, D Klingebiel, M Komm[a], G Müller, M Rieger[*], J Steggemann[b], M Urban and T Winchen

III. Physics Institute A, RWTH Aachen University, Aachen, Germany

[a] now at Centre for Cosmology, Particle Physics and Phenomenology, Université catholique de Louvain, Louvain-la-Neuve, Belgium

[b] now at CERN, European Organization for Nuclear Research, Geneva, Switzerland

[*] corresponding author, rieger@physik.rwth-aachen.de

E-mail: vispa@lists.rwth-aachen.de

Abstract. Visual Physics Analysis (VISPA) is a web-based development environment addressing high energy and astroparticle physics. It covers the entire analysis spectrum from the design and validation phase to the execution of analyses and the visualization of results. VISPA provides a graphical steering of the analysis flow, which consists of self-written, re-usable Python and C++ modules for more demanding tasks. All common operating systems are supported since a standard internet browser is the only software requirement for users. Even access via mobile and touch-compatible devices is possible. In this contribution, we present the most recent developments of our web application concerning technical, state-of-the-art approaches as well as practical experiences. One of the key features is the use of workspaces, i.e. user-configurable connections to remote machines supplying resources and local file access. Thereby, workspaces enable the management of data, computing resources (e.g. remote clusters or computing grids), and additional software either centralized or individually. We further report on the results of an application with more than 100 third-year students using VISPA for their regular particle physics exercises during the winter term 2012/13. Besides the ambition to support and simplify the development cycle of physics analyses, new use cases such as fast, location-independent status queries, the validation of results, and the ability to share analyses within worldwide collaborations with a single click become conceivable.

1. Introduction

The demand for distributed resources and global access to data experienced a rapid growth in the past years. In fact, both, the way we use instrumental technology and the requirements we claim on hardware and software currently undergo a change of paradigms. The development work of scientists is no longer subject to limitations such as locally restricted availability of computing power and data access. Instead, the change of infrastructure becomes the focus of attention in order to fulfill individual needs concerning place and time independence. This is especially achieved by the progress made in the field of networking technology.

The consequent advantages of network (and internet) communication can be applied to scientific workflows. We present a new approach that is based on the desktop version of the Visual

Physics Analysis (VISPA) project, which has been successfully used in various high-energy and astroparticle physics data analyses [1]. The new concept implements the VISPA platform for the application within a standard web-browser. The underlying server-client-approach is based on central software, data, and resource deployment as well as time- and location-independent accessibility. It also provides the opportunity of cooperative analysis creation. Therefore, it can be used as a platform for various scientific applications, e.g. for novel blended learning concepts in a wide scope of physical teaching.

2. The Visual Physics Analysis project

The Visual Physics Analysis (VISPA) project provides a development environment for physics analyses covering the entire analysis spectrum from the design and validation phase to the execution of analyses and the visualization of results. It utilizes the C++ Physics eXtension Library (PXL) [2] as underlying analysis framework.

Analysis that are based on PXL are subdivided into logically separated modules, i.e. blocks of code that are designated to carry out well defined tasks. In addition to predefined I/O modules, PXL provides the possibility to include reusable, user-defined modules, written either in C++ or Python [3]. Connections between these modules describe the data-flow of the analysis, which is subject to an event-by-event approach.

VISPA provides a graphical representation of analyses. It not only reflects their structure, i.e. the sequence of modules and the connections between them, but it also allows for graphical steering. Therefore, the web-based implementation of VISPA includes a variety of extensions, aimed to cover a wide range of functionality.

3. Technical implementation

3.1. Server-client approach

The VISPA platform follows a server-client approach that is typical for modern web applications. A web browser (client) sends a specific request to a web server via the Hypertext Transfer Protocol (HTTP). The server processes the request to create a response that is sent back to the client. There, it can be parsed to either display new content, usually via HTML markup [4], or to update already existing content using the AJAX concept [5].

The server is built on the basis of the Python web framework CherryPy [6]. This allows for a convenient integration of various scientific software packages written in Python, such as PXL, ROOT [7], and NumPy [8]. The user management is realized using a SQL database that stores login information for all registered users. The SQL dialect, e.g. MySQL, PostgreSQL, or SQLite, can be altered as the object-relational mapping (ORM) framework SQLAlchemy [9] is incorporated. The content of HTTP responses is created and cached using the Mako templating library [10].

Although the server-client system is intended to operate on separate computers via network communication, the possibility to install the VISPA platform locally on a single computer is preserved. In this operation mode, the user does not need to register to the platform, which accordingly is accessible only from the local machine.

3.2. Workspaces

The implementation of workspaces is a new key ingredient of the VISPA platform. Scientific analyses often depend to a great extent on the availability of computing power and data access. Hence, in some scientific environments a single machine may not be appropriate for acting as both, web server and processing unit at the same time. For the purpose of splitting the architecture into logically and physically separated units, all user-related computational tasks are sourced out into workspaces.

Every machine running Python that is accessible via SSH over the network can act as a workspace, regardless of whether it is, e.g., a mobile phone, a desktop machine, or a computing cluster. Access is granted if the client is able to connect to the workspace using password or public key authentication. At the beginning of a connection, all necessary files are copied over to the workspace where a receiving process is started. The communication between this process and the VISPA server is subject to the JSON-RPC 2.0 specification [11]. This way, the entire host system of the workspace can be utilized including file access, permissions settings, and CPU usage.

In this model, the VISPA server itself takes on the role of an intermediate node: it serves requests and delivers files related to the VISPA platform on the one hand, and handles connections as well as distributes computational tasks to workspaces on the other hand. A generic overview of this approach is shown in figure 1.

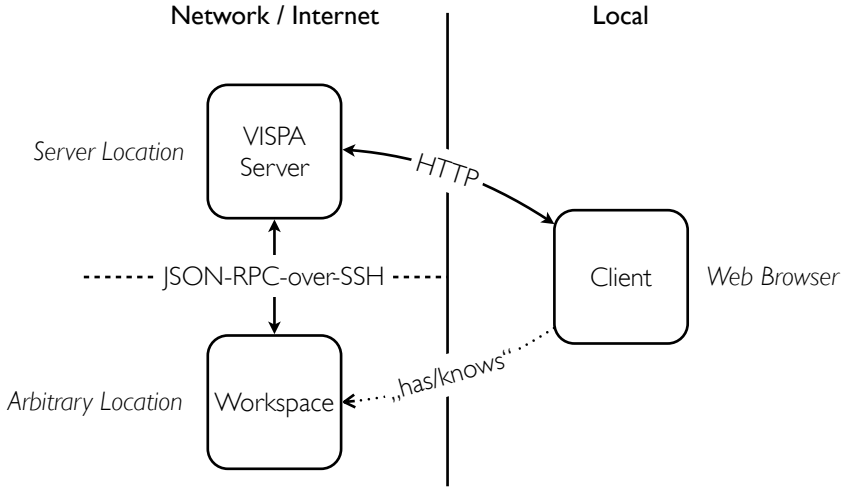

Figure 1. Schematic overview of the workspace approach.

3.3. User interface and extensions

The graphical user interface (GUI) is based on the HTML5 markup language and the JavaScript scripting language, which are both supported by common web browsers. Advanced interaction components and compatibility for mobile devices are implemented using the DOJO framework [12] and the jQuery UI framework [13]. This software foundation is provided by the web server and initially transfered to the web browser on startup. It is then utilized by both, the VISPA platform itself and its extensions. A selection of extensions is illustrated in figure 2.

The *Analysis Designer* provides the GUI for developing physics analyses based on the PXL toolkit. Modules and connections between them can be created and/or changed by drag & drop gestures in order to steer the analysis flow.

The *File Browser* implements the possibility to browse through personal and public files that are available on the currently selected workspace. It provides common features like, e.g., up- and download, copy, and paste functionality as well as the possibility to select and open files in other extensions.

In high energy physics, a data file may contain sophisticated information of multiple events. Files of the format `pxlio`, introduced by PXL, can be inspected event-by-event with the *PXL Browser*. The properties of the constituents can be obtained by clicking on their graphical representations.

For the purpose of directly inspecting, creating, and steering analysis code within the web browser, a *Code Editor* is incorporated based on the Ace Editor project [14].

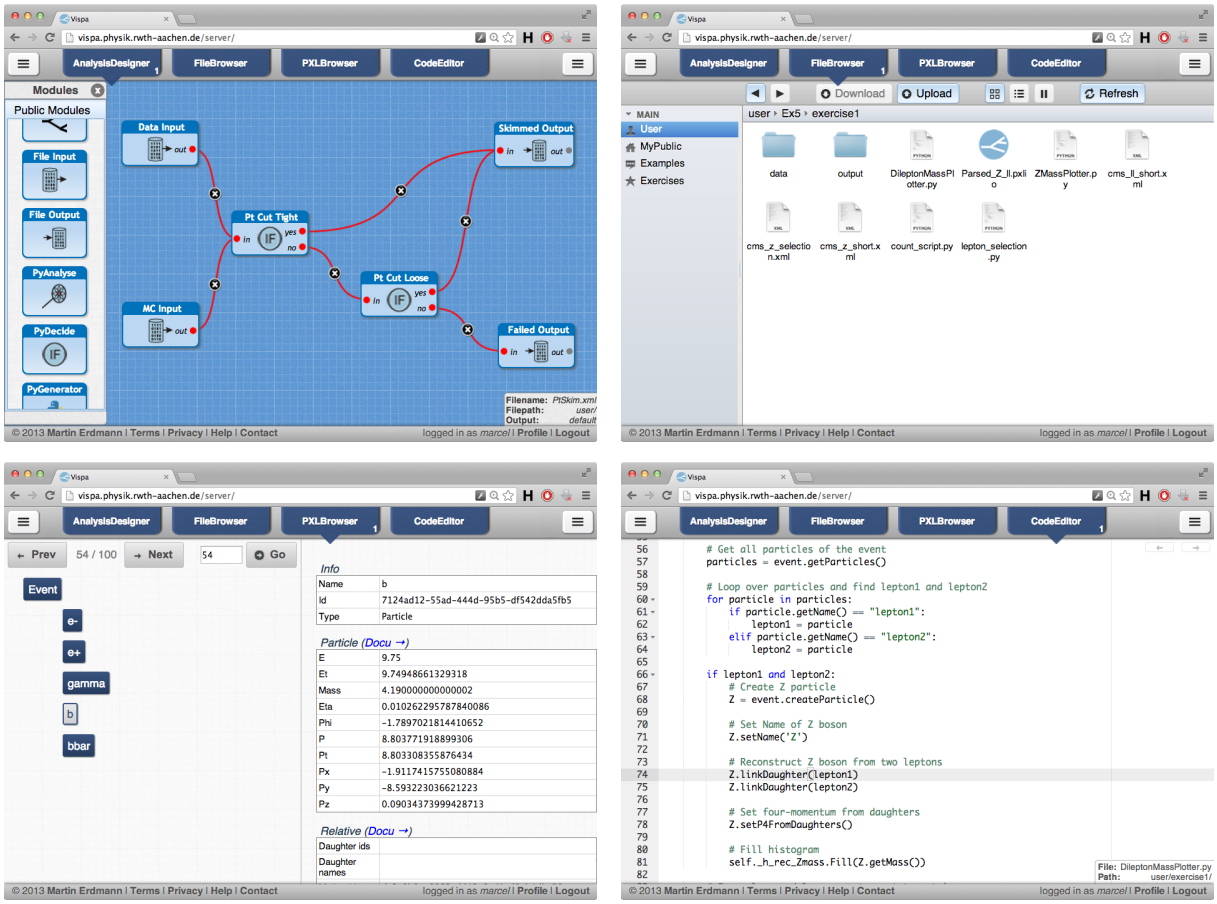

Figure 2. Extensions of the web-based implementation of VISPA (from top-left to bottom-right): Analysis Designer, File Browser, PXL Browser, and Code Editor.

4. A field test with third year students

The practical use of the VISPA platform was tested in the scope of the course "Particle Physics and Astrophysics" held during the winter term 2012/13 with 3 hours of lectures per week and exercises due every 2 weeks. The course consisted of more than 100 third year physics students. 25 % of the points of every exercise could be achieved by successfully working on data analysis tasks using VISPA. A total of 50 % of all exercise points was mandatory to receive the permission to participate in the final exam.

The server setup was specially dedicated to the application with students. An instance of the VISPA platform was implemented on a Linux server with 10 worker nodes in accordance with a security concept created in collaboration with the computing center of the RWTH Aachen University. It turned out that even at peak times, memory and CPU occupancy were sufficient to supply each student request with adequate computing power.

In order to assess the success of the application, a survey was started to obtain feedback about the acceptance and learning benefit of the VISPA platform. 63 students participated the survey that comprised both, predefined questions and the possibility for free text comments. Valuable comments on the workflow were received. The majority concerned the number of interactions, i.e. the number of mouse clicks, needed to perform an iteration of the data analysis exercise. The result of the overall rating for the VISPA platform is shown in figure 3, yielding a positive assessment of the learning project, which is visible on the right side of the distribution.

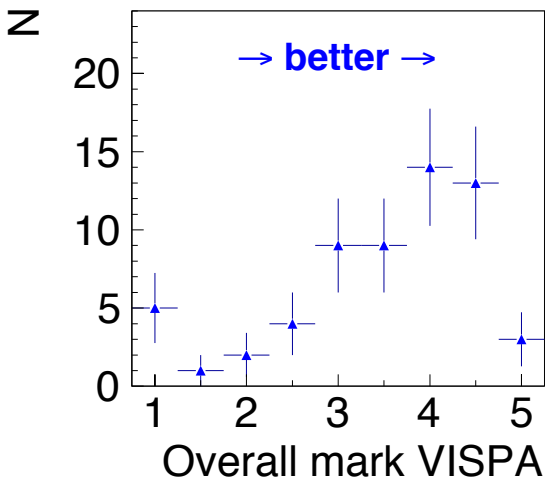

Figure 3. The student's overall rating of the VISPA learning concept on a scale with the highest value being the best mark.

5. Conclusions

We have presented a web-based development environment addressing workflows in high energy and astroparticle physics. It enables the development and execution of physics data analysis in a standard web-browser. Workspaces are a key ingredient of the platform. They allow for remote access to data, software, and resources on arbitrary machines that are accessible via SSH. The potential of the presented web platform is comprehensive. For example, scientists are able to collaborate on analysis concepts and algorithms through the web interface. A new level of transparency may be offered in publications by providing web access to the corresponding analysis. Students may get access to analysis examples exploring public experiment data with professional tools that are standard in todays physics analyses.

Acknowledgments

We wish to thank the organizers of the ACAT2013 conference for their kind support. This work is supported by the Ministerium für Wissenschaft und Forschung, Nordrhein-Westfalen, the Bundesministerium für Bildung und Forschung (BMBF), and the Helmholz Alliance Physics at the Terascale. M Rieger and C Glaser gratefully thank for support by the Deutsche Forschungsgemeinschaft (DFG) and T Winchen gratefully acknowledges funding by the Friedrich-Ebert-Stiftung.

References

[1] Bretz H-P et al. (2012) *A Development Environment for Visual Physics Analysis* JINST **7** T08005 doi:10.1088/1748-0221/7/08/T08005 arXiv:1205.4912v2 `http://vispa.physik.rwth-aachen.de`
[2] Erdmann M et al. (2012) *Physics eXtension Library (PXL)* `http://vispa.physik.rwth-aachen.de/PXL`
[3] *Python Programming Language* `http://www.python.org`
[4] World Wide Web Consortium *HTML5 Specification* `http://www.w3.org/TR/html5`
[5] Holdener A T (2008) *Ajax - The Definitive Guide* (O'Reilly Media, Sebastopol, CA, USA)
[6] The CherryPy Team *CherryPy - A Minimalist Python Web Framework* `http://www.cherrypy.org`
[7] Brun R and Rademakers F (1996) *Nucl. Inst. & Meth. in Phys. Res. A 398 81-86* `http://root.cern.ch`
[8] NumPy Developers `http://www.numpy.org/`
[9] SQLAlchemy Authors and Contributors `http://www.sqlalchemy.org`
[10] Bayer M *Mako Templates for Python* `http://www.makotemplates.org/`
[11] JSON-RPC 2.0 Specification `http://www.jsonrpc.org/specification`
[12] The Dojo Foundation `http://dojotoolkit.org`
[13] The jQuery Foundation `http://jqueryui.com`
[14] Cloud9 IDE and the Mozilla Foundation `http://ace.c9.io`