Science Gateways in HPC: Usability meets Efficiency and Effectiveness

Sandra Gesing

1.1 Introduction

HPC (High-Performance Computing) infrastructures provide the me-ans for compute-intensive modeling and simulations to achieve results in reasonable time. Efficiency and effectiveness are the traditional key targets for the optimization of such applied scientific methods and they are major drivers for research and developments in HPC. In the last years a further target has arisen driven by the needs of user communities to enable them to focus on their research questions without becoming deeply acquainted with the complex technical details of HPC: usability of modeling and simulations in HPC. Science gateways address this aspect as end-to-end solutions providing intuitive user interfaces while connecting to the underlying complex infrastructures and hiding the technical details as far as feasible and desired from the users. This trend is reflected in quite a few web development frameworks, containerizations, science gateway frameworks and APIs with different foci and strengths, which have evolved to support the developers of science gateways in implementing an intuitive solution for a target research domain. Science gateways have evolved into a new era since 2014 when providers of distributed computing infrastructures reported the first time that the computing and storage resources have been applied more often via science gateways than via command line [1]. Part of this success can be credited to the development of reliable and mature science gateway frameworks over the last decade [2]. Especially the rise of larger data amounts and the importance of workflows for user communities have been recognized and sophisticated data and workflow management solutions [3] have found their way into such frameworks.

The challenges for developers of specific science gateways, which apply HPC infrastructures, are manifold: from intuitive user interfaces for the targeted research domain and security features through efficient job, data and workflow management to parallelization of applications employing parallel and distributed architectures. The knowledge about existing science gateway technologies and their distinctive features and strengths helps developers to select a suitable framework or API without the need to re-invent the wheel and to start the development of a specific solution from scratch.

In the area of science gateways several sources are available to get a well-informed impression of the state-of-the-art technologies and novel developments. Yearly science gateway workshops are established in Europe, the US and Australia, which are partnering and form an international platform to shape future directions for research, foster the exchange of ideas, standards and common requirements and push towards the wider adoption of science gateways in science [4, 5]. The peer-reviewed publications of the workshops and the joint special issues reflect the international standard in this field [6]. IEEE has also observed the importance of science gateways and the IEEE Technical Area on Science Gateways is a further source of information on events, publications and projects [7]. Besides such community-driven resources, the US National Science Foundation (NSF) [8] - as one of the main funding bodies in the US - has recognized the significance of science gateways and is funding the Science Gateways Community Institute [9]. The Science Gateway Institute provides among other services an excellent contact for general information on projects and technologies [5]. The selection of a suitable technology for a specific use case is essential and helps reducing the effort in implementing a science gateway by reusing existing software or frameworks. Thus, a solution for a user community can be provided more efficiently. Additionally, novel developments in web-based technologies and agile web frameworks allow for supporting developers in efficiently creating web-based science gateways.

1.2 Science Gateways and Usability

The overall goal of science gateways is to provide an end-to-end solution and increase the usability of applications especially for researchers who are not necessarily IT specialists. The significance of usability and graphical user interfaces is evident in the history of IT developments in the last 50 years: Doug Engelbart's Augmentation of Human Intellect project, which developed a mouse-driven cursor and multiple windows in the 60s [10], Apple's designs starting in the 70s and resulting in a hype in the last 10 years around smartphones and tablets, the first web browser [11] and an ISO standard on usability for "visual display terminals" in the 90s [12]. The Internet revolutionized research in the last 25 years with increasingly more sophisticated and efficient distributed computing infrastructures and data management solutions having evolved to maintain and increase their usability. Novel developments in web-based technologies as well as agile web frameworks allow for supporting developers in efficiently creating user interfaces for web-based science gateways.

On the user interface side many libraries and frameworks have evolved and we will only mention a few without the claim of completeness. In general, JavaScript libraries, CSS and HTML5 with Ajax [13] allow for dynamic websites focusing on the frontend with advanced features. jQuery [14] is a widely used JavaScript library with standard user interface methods for HTML document traversal and manipulation and event handling. jsPlumb [15] is also a JavaScript library with focus on the illustration of graphs and workflows with many implemented features for the appearance of nodes and edges and corresponding annotations. 3D graphics can be seamlessly created and edited in web browsers via the JavaScript API WebGL [16] without the need of installation of further software. The front-end framework Semantic UI [17] makes use of JavaScript library jQuery, while providing intuitive classes for designing web user interfaces based on the philosophy “everything arbitrary is mutable”. The web application frameworks ReactJS [18], Foundation [19], AngularJS [20] uses declarative programming and follows the MVC concept (Model-View-Controller) [21] to separate data, presentation and logical components in a clean design.

While the look-and-feel of the user interface is especially important for the acceptance in the user community, the backend and the integration with the underlying infrastructure, which is mostly hidden from the users, is the more complex task from the technical point of view. Some technologies are widely used for web-based science gateways but are lacking standard libraries for the support of HPC infrastructures such as the open source content management systems Drupal [22] and Joomla [23] and the high-level framework Django [24]. Thus, the developers are creating such integrations from scratch. The lack of HPC support out of the box also applies to portal frameworks such as Liferay [25] and Pluto [26] but offering the advantage of reusability of so-called portlets. The portal frameworks are implementations of the JSR168/JSR286 [27, 28] standards and they enable to implement portlets once and deploy them in every portal framework, which supports these standards. Especially Liferay is widely used for science gateways in the HPC community. In the last eight years a couple of science gateway frameworks have been developed on top of Liferay, benefitting from the available authentication and authorization mechanisms and layout features, e.g., gUSE/WS-PGRADE [29].

In general, the architecture of science gateway technologies for distributed systems consists of four layers: 1) the user interface layer, 2) the application layer, 3) the high-level services layer such as job management and data management and if applicable workflow management and 4) the connected cluster, grid and cloud infrastructures (see Fig. 1). While the first two layers may be specific for each science gateway developed via a science gateway technology, the third and fourth layer are generic and can be re-used for any science gateway irrespective of its target domain. The generic requirements on such layers have led to the development of multiple mature science gateway technologies. We refer to examples here, which are free and available as open source and can be as such further developed by the community and are not based exclusively on a business model.

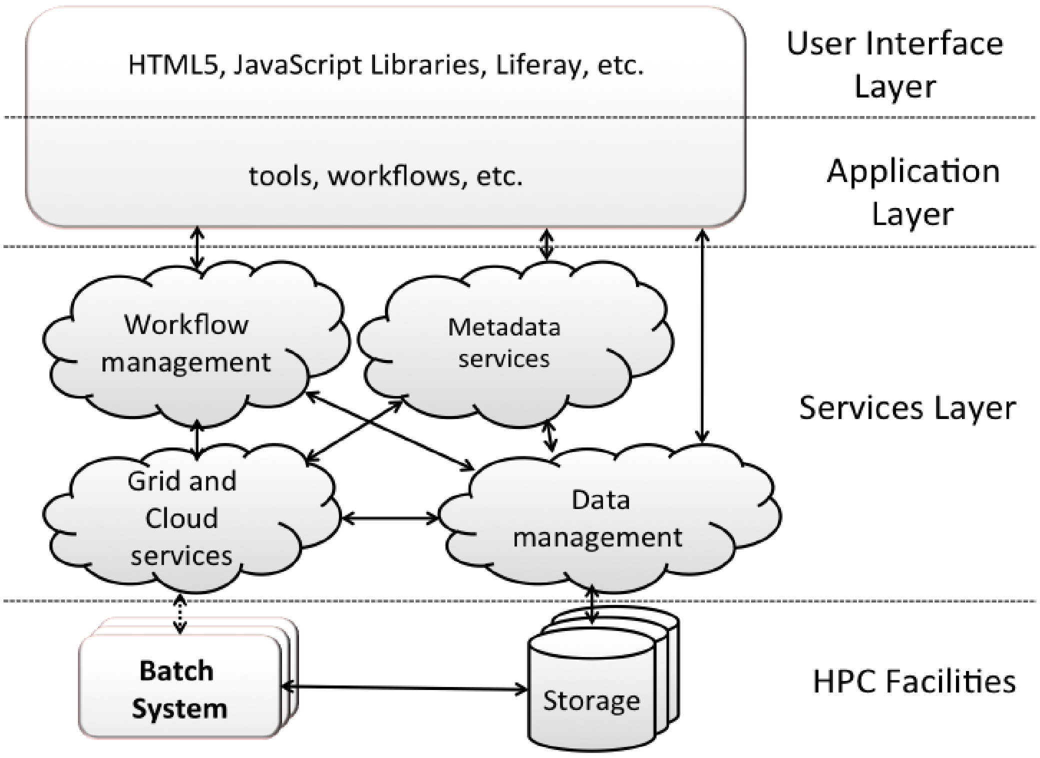

Figure 1: The general infrastructure for science gateways with providing access to HPC resources. For each layer are examples of technologies provided.

One category includes workbenches such as Taverna [30], the Kepler workbench [31], KNIME (the Konstanz Information Miner) [32], and the UNICORE Rich Client [33]. These examples are additionally offering workflow management capabilities. They necessitate the installation of software on the user side and offer a workflow canvas to graphically create and edit workflows and submit them to the underlying infrastructure. Each user interface layer provides the same look-and-feel for all applications. The target infrastructures are quite different for these examples though. Taverna supports the workflow management of available web services, whereas Kepler targets command line tools like R scripts or compiled C implementations. KNIME is intuitive, versatile, widely used, and currently being extended to support generic access to HPC resources. The UNICORE Rich Client focuses on the exploitation of compute and data infrastructures, which are integrated via the UNICORE grid middleware [34].

The second category of science gateways contains web-based science gateway frameworks. gUSE/WS-PGRADE, Galaxy [35], HubZero [36] and the Catania Science Gateway Framework [37] belong to this category. The first two offer workflow editing features and workflow management as well whereas HubZero provides workflow management options in the backend via the workflow management system Pegasus [38] but focuses more on the integration of single applications and collaboration tools analogous to the Catania Science Gateway Framework. gUSE/WS-PGRADE and Galaxy offer generic workflow canvasses capable of managing command line tools and web services. The concepts behind creating the workflows are quite different though. While WS-PGRADE includes the option for the users to upload and invoke scripts and computational tools, Galaxy is designed as toolbox, which is configured by an administrator and users can select from a list of available tools. The extension of the science gateway frameworks with user interfaces especially tailored to a specific application can be performed in WS-PGRADE as portlets developed on top of Liferay. Galaxy is not developed on top of a standard framework and thus does not directly support the implementation of specific user interfaces but since it is available as open source, developers are able to extend the framework to communities' demands.

The third category is concerned with the development of science gateways and includes mature APIs and libraries offering features for the implementation of the first three layers of the science gateway architecture. Examples are Apache Airavata [39], the Agave Platform [40] and the Vine Toolkit [41], which aim at reducing the effort on the developer side while enabling to apply novel user interface technologies and frameworks. All three frameworks are supporting diverse programming languages and the basic concept is the same.

1.3 Designing Science Gateways

The close collaboration with the respective user community is crucial to gather all necessary information and requirements on a science gateway that is intended to serve for the specific use case. This usually underestimated design task is more often than not the most challenging one. While users are mainly experts in their research domain, they may be not aware of the implications of using specific software, the availability of a computational tool, security demands or concepts such as workflows. The exact layout for the science gateway is usually a continuous and iterative process with suggestions from developers for the layout and feedback and comments from the user community.

Through the experience with several projects and communities, we have created a checklist for important topics to discuss and address in collaboration with the communities [42]. This checklist can be used for supporting the creation of a Software Requirement Specification (SRS), for example, following the 830-1998 - IEEE Recommended Practice for Software Requirements Specifications [43]. According to the recommendation a SRS should be

a) Correct;

b) Unambiguous;

c) Complete;

d) Consistent;

e) Ranked for importance and/or stability;

f) Verifiable;

g) Modifiable;

h) Traceable

It should address the software product – not the process of producing a software product. In the MoSGrid project (Molecular Simulation Grid) [44], for example, we created first a survey answered by about 50 domain researchers in computational chemistry, with questions such as which tools they use, whether they know about computational workflows and may use already workflows and whether they would like to share their data and/or workflows. In the design process have been directly involved the domain partners of the project (five domain researchers from three affiliations).

The topics can be distinguished in the three main categories: domain-specific topics, organizational topics and technical topics. Domain-specific topics can be again divided in three major groups:

1. Requirements, which lay in the nature of the research topic.
2. Requirements, which refer directly to the target community and their specific needs, their diversity in experience and knowledge regarding the research topic and/or computational tools and their analysis steps.
3. Requirements, which result from available specific resources from lab instruments to local, on-campus, national to international distributed computing infrastructures.

Topics of the three groups are in detail explained in Table 1.

Groups	Topics	Examples
Requirements referring to the research topic	Goal and target area of the envisioned science gateway	Workflows for computational drug design using docking tools
	Scale and format of the	Molecular structures in

	available data	PDB format
Requirements referring to the target user community	Groups of users distinguished via their experience in the research domain	Novel users to the research topic such as students or experienced users in the research domain
	Groups of users distinguished via their experience with computational tools	Wet-lab researchers mostly familiar with working with Excel or researchers familiar with command line usage of computational tools
	Layout and feature requirements	Strictly pre-configured user interfaces, possibilities for changing parameter configurations or possibilities to process own scripts
	Priorities of features and options	A list ranging from must-have to great-to-have options
	Integration of existing applications or development of new applications from scratch	Computational tools already used in the community, e.g., Gromacs, or developing statistical approaches with R
	Visualization	Browsing of data or interactive modules like a molecule editor

	Workflow management	Pre-configuration of connected tasks for a certain purpose such as optimizing molecular structures for a docking workflow
	Security and privacy management	Private space for research results before publication or patents and shared spaces for results afterwards
Requirements referring to available infrastructures	Hardware	External disk at a lab containing the input data
	Credentials	Access via on-campus accounts or Grid certificates for national resources
	Batch systems, Grid middlewares or Cloud systems	PBS, UNICORE, etc.
	Data management systems	dCache, iRODS etc.

Table 1: The checklist illustrates the domain-specific groups and the topics, which can be used by principal investigators and/or developers for designing and implementation of a successful science gateway.

In contrast to domain-specific topics, organizational topics refer to measures for a successful collaboration in general, which are influenced via external factors of a project such as time constraints of a grant or internal factors such as availability of alpha- or beta-testers (see Table 2). It may be not feasible to receive information on all topics or to set up all organizational aspects from the beginning but important is to raise the topics and start the conversation. The information is essential for the correct choice of technologies and may prevent a significant amount of refactoring. While in industry the creation of a system specification is the common approach to define all requirements, projects in academia work differently and are often more dynamic and less clearly specified. One reason lies in the research nature of the projects.

Groups	Topics	Measures
External topics	Limited funding and time constraints	Project plan with deliverables and milestones
	Availability of data and computational tools from third party affiliations	Communicating via emails and calls with third party affiliations
Internal topics	Concurrent collection of requirements and features	Weekly meetings of researchers and developers or development team
	Concurrent feedback during the development	Agreement on alpha- and beta-testers in the community
	Location of teams	Distributed teams compared to local teams necessitate the use of conference calls as well as emails to a larger extent, maybe under consideration of different time zones

Table 2: The checklist illustrates the external and internal topics and measures on the organizational side, which can be used by principal investigators and/or developers for designing and implementation of a successful science gateway.

Besides the topics deriving from the research domain and the interdisciplinary collaboration, also technical topics of the design of available infrastructures and considerations regarding the involved development team or single developers are important to examine. Such subjective project analyses are complemented by the investigation of objective conditions such as available support of suitable technologies. See Table 3 for a comprehensive list of technical topics. This checklist is intended to support the design decision process and is general applicable. Each use case is different and there is not one technology, which fits them all but various mature solutions, which can form the basis for diverse science gateways.

Groups	Topics
Subjective factors	Experience with existing frameworks, programming languages and data access methods
	Effort for extending existing frameworks compared to novel developments for the specific use case
	Synergy effects with other science gateway projects
	Available infrastructure in the hosting environment including security infrastructure and resources
Objective factors	Available support of suitable technologies
	Scalability of suitable technologies
	Technologies of the applications, which have to be integrated
	Technical requirements of the applications and/or of access to input data
	Performance measures of applications

Table 3: The checklist illustrates the subjective and objective factors and the topics, which can be used by principal investigators and/or developers for designing a successful science gateway.

1.4 Reusability of Scientific Methods and Reproducibility of Science

One of the goals of science gateways is to offer methodologies for performing analyses, which can be re-used for different data sets and by different users. Thus, many science gateways offer sharing possibilities within a community, between different science gateway instances of one technology or even between diverse technologies, e.g., via the SHIWA platform [45, 46]. Even though such sharing options are available, reusability of methodologies and reproducibility of science are mainly dependent on two main aspects:

1. The willingness of researchers to share methodologies and data.
   Even with easy-to-use sharing options, researchers need to perform further steps to provide their methodologies and data to a community, which cost time and effort. They might have invested a large amount of time and computing power to create these and see an advantage to keep the knowledge in their group and between collaborators for creating further results with this part of their research. A survey in the MoSGrid [47] community elucidated that 70% would share their results and molecular structures in a repository after they have published them or own a patent. The disposition to share tools and workflows was higher with nearly 90%. If these results can be transferred to researchers in general, the 70% or 90%, respectively, would be a promising result to achieve reproducibility of science at a high rate.

2. Technical dependencies of methodologies and data.

   Methodologies are among others dependent on operating systems, tools in diverse versions and local or distributed data. A study on the social marketplace MyExperiment [48] for sharing Taverna workflows, for example, presents that only 20% of the workflows are reproducible and reusable out of the box.

> For solving such problems in science gateways, the different sharing possibilities have to be analyzed and – where necessary – tools, data and workflows have to be provided in diverse infrastructures and via various job and data management systems.

While research areas and the science gateway technologies are independent of institutional, state or national boundaries, this does often not apply to research infrastructures, which offer HPC infrastructures such as EGI [49] and PRACE [50] in Europe or XSEDE [51] in the US. The acquisition and maintenance of resources depend on funding, which can be institutional, national or on international level. Thus, the use of such resources is bound to policies and rules of the funding bodies. To support user communities across such boundaries, it is essential for science gateway creators to understand the effects of applying different research infrastructures.

Science gateways consists in general of three layers (see Figure 1) and thus form a science gateway infrastructure:

1. User interface layer – This layer determines the layout and design of the user interface visible to the community.

2. Application layer – This layer is responsible for the features offered in the science gateway, e.g., generic applications such as security features for authentication to and authorization in the science gateway as well as domain-oriented applications such molecular structure editors. In case of workflow-enabled science gateways, this might be a workflow editor.

3. Services layer – This layer connects to services of the science gateway framework such as data repositories and services such as adaptors to apply batch, grid or cloud systems or distributed data management systems.

In a well-designed science gateway, the first layer is independent of the underlying research infrastructure while the latter influences the second and third layer.

Policies of research infrastructures add another complexity layer such as the application process to receive allocations on and access to available resources. The German National Grid Infrastructure, for example, only gives access to users of German universities and their collaborators. Other research infrastructures provide a more international approach such as InCommon [52], for example. InCommon is an international initiative for global interfederation for security credentials joined by over 40 national federations. However, policies of research infrastructures are quite diverse from each other, which thus hampers to determine generic challenges for policies of research infrastructures. The conclusion is that policies themselves can form a technical and organizational challenge dependent on restrictions resulting from them.

The following challenges in the application layer have to be considered.

A1. Security

Since one of the goals of science gateways is to create an easy-to-use interface to underlying resources, a single sign-on feature for accessing the science gateway via the same security credentials as the resources is highly desirable. Thus, the authentication mechanism of the science gateway or an additional feature in the science gateway has to support the credentials needed in a research infrastructure and corresponding authorization to compute and data resources.

A2. Available tools

Diverse research infrastructures have instantiated diverse policies regarding available tools. While some allow uploading own tools for submission to the research infrastructure, some only allow using pre-installed tools on the research infrastructure. In case of large software packages the installation has to be performed on the infrastructure for an efficient use. It is inefficient to install the whole package during each submission of a job referring to a tool in such a software package. Virtualizations like Docker containers [53] form efficient solutions here.

A3. Available data

Input data for a job or workflow may be available locally in the science gateway or in one research infrastructure. If the data is available in a certain research infrastructure and mandatory for the effective and efficient application of a tool or workflow in another research infrastructure, it needs to be added to the targeted infrastructure.

Similar challenges are faced in the services layer, though they take place on a different technical layer. Thus, the following challenges have to be addressed.

S1. Job management system

Each research infrastructure supports at least one job submission system. It might be a local, batch, grid or cloud system, which includes authentication, authorization and accounting mechanisms. The services available in the science gateway infrastructure have to be analyzed whether they support one of the available job management systems and its security demands. While diverse hardware architectures might be available in the research infrastructure, the differences are handled by the job management system.

S2. Data management system

The application of an available data management system in the targeted research infrastructure results in a more efficient performance of tools and workflows since it relieves the users from unnecessary uploads and downloads, which could be very time-consuming in data-intensive analyses. Thus, the possibility to directly access the supported data management system via services in the science gateway infrastructure is beneficial for the efficiency of applications. Analogue to job management systems, each data management system possesses its own security features with authentication, authorization and accounting mechanisms.

S3. Data transfer protocols

Additionally to the aforementioned data management systems, research infrastructures provide data transfer protocols for transferring data in general - whether it consists of executables, scripts, small or large input and output data sets or databases. Thus, the science gateway infrastructure needs to support at least one of the available data transfer protocols applied in the research infrastructure to be able to transfer files at all.

The aforementioned challenges are considered a minimal set regarding the technical access to research infrastructures via science gateways.

1.5 Conclusion

The overall goal of science gateways in HPC is the increased usability of modeling of data and simulations using complex underlying computing infrastructures. This chapter introduces the generic architecture of science gateways and examples for mature solutions. It outlines the importance of gathering information for designing science gateways for domain researchers, who want to apply HPC infrastructures and distributed data management. We have presented checklists for developers in interdisciplinary projects considering domain-related, organizational and technical aspects independent of a specific selected technology but as starting point for selecting technologies and designing as well as implementing a science gateway for a community. These checklists can be used to prepare the interaction with domain researchers and to make informed decisions about technologies suitable for specific science gateways. A wide range of mature and maintained web frameworks and science gateway technologies are available to aid developers in designing and implementing such solutions. While each of them have their own communities, they also have their specific advantages and disadvantages for use cases. Aspects such as scalability and feature availability narrow the scope and help to choose the most suitable technology.

Bridging the differences between research infrastructures via science gateways is a promising way to set the stage for reusability of scientific methodologies and reproducibility of research on an international stage. However, current technical implementations are not sufficient to achieve such goals but also the researchers themselves have to be willing to open up their methodologies and data to the community. Science gateways can be beneficial for this purpose and ease the required steps – especially if they are available in diverse research infrastructures. We have elucidated the challenges faced when science gateways are ported to various research infrastructures in general. While the technical challenges can be summarized in a minimal set consisting of security mechanisms, tool and data availability as well as data management and data transfer protocols, the challenges resulting from policies are dependent on the policies themselves.

References

[1] K. A. Lawrence, N. Wilkins-Diehr, J. A. W.ernert, M. Pierce, M. Zentner, S. Marru. 2014. Who cares about science gateways?: a large-scale survey of community use

and needs. In *Proceedings of the 9th Gateway Computing Environments Workshop* (GCE '14). IEEE Press, Piscataway, NJ, USA, 1-4. DOI=10.1109/GCE.2014.11 http://dx.doi.org/10.1109/GCE.2014.11.

[2] R. Dooley, M. R. Hanlon. Recipes 2.0: building for today and tomorrow, Concurrency Computat.: Pract. Exper., 27, 258 (2015).

[3] J. Liu, E. Pacitti, P. Valduriez and M. Mattoso: A survey of data-intensive scientific workflow management, Journal of Grid Computing, Springer, 2015, 13, 457-493.

[4] IWSG (International Workshop on Science Gateways), http://iwsg.info/

[5] Gateway Workshops, http://sciencegateways.org

[6] Gesing, S., Wilkins-Diehr, N., Barker, M. and Pierantoni, G. Special Issue on Science Gateways. Journal of Grid Computing, 14(4):495–703, 2016

[7] IEEE Technical Area on Science Gateways, http://ieeesciencegateways.org

[8] National Science Foundation (NSF), http://nsf.gov

[9] Gesing, S., Wilkins-Diehr, N., Dahan, M., Lawrence, K., Zentner, M., Pierce, M., Hayden, L.B., and Marru, S. Science Gateways: The Long Road to the Birth of an Institute. Proc. of HICSS-50 (50th Hawaii International Conference on System Sciences), 4-7 January 2017, Hilton Waikoloa, HI, USA, http://hdl.handle.net/10125/41919

[10] Engelbart, D.C., Augmenting Human Intellect: A Conceptual Framework, Summary Report AFOSR-3233, Stanford Research Institute, Menlo Park, CA, October 1962

[11] The WorldWideWeb Browser. http://www.w3.org/People/Berners-Lee/WorldWideWeb.html, 2016 (accessed 29.02.2016).

[12] ISO 9241-1:1992, http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_detail_ics.htm?csnumber=16873, 2016 (accessed 29.02.2016).

[13] AJAX, http://www.w3schools.com/ajax/, 2016 (accessed 29.02.2016).

[14] jQUERY, https://jquery.com/, 2016 (accessed 29.02.2016).

[15] jsPlumb, https://github.com/sporritt/jsPlumb, 2016 (accessed 29.02.2016).

[16] WebGL. https://www.khronos.org/news/press/khronos-releases-final-webgl-1.0-specification, 2016 (accessed 29.02.2016).

[17] Semantic UI. http://semantic-ui.com/, 2016 (accessed 29.02.2016).

[18] ReactJS, http://reactjs.net/, 2016 (accessed 29.02.2016).

[19] Foundation, http://foundation.zurb.com/, 2016 (accessed 29.02.2016).

[20] AngularJS, https://angularjs.org/, 2016 (accessed 29.02.2016).

[21] G. E. Krasner and S. T. Pope. "A Cookbook for Using the Model-View Controller User Interface Paradigm in Smalltalk-80". Journal of Object-Oriented Programming, 1(3):26{49, 1988.

[22] Drupal. https://drupal.org/, 2016 (accessed 29.02.2016).

[23] Joomla. http://www.joomla.org/, 2016 (accessed 29.02.2016).

[24] Django. https://www.djangoproject.com/, 2016 (accessed 29.02.2016).

[25] Inc. Liferay. “Liferay". http://www.liferay.com, 2016 (accessed 29.02.2016).

[26] Apache Software Foundation. “Pluto", 2016 (accessed 29.02.2016).

[27] A. Abdelnur and S. Hepper. “JSR168: Portlet Specification". http://www.jcp.org/en/jsr/detail?id=168, 2003 (accessed 29.02.2016).

[28] M.S. Nicklous and S. Hepper. “JSR 286: Portlet Specification 2.0". http://www.jcp.org/en/jsr/detail?id=286, 2008 (accessed 29.02.2016).

[29] P. Kacsuk, Z. Farkas, M. Kozlovszky, G. Hermann, A. Balasko, K. Karoczkai and I. Marton: WS-PGRADE/gUSE Generic DCI Gateway Framework for a Large Variety of User Communities, Journal of Grid Computing, Springer Netherlands, 2012, 10, 601-630.

[30] K. Wolstencroft, R. Haines, D. Fellows, A. Williams, D. Withers,S. Owen, S. Soiland-Reyes, I. Dunlop, A. Nenadic, P. Fisher,J. Bhagat, K. Belhajjame, F. Bacall, A. Hardi-

sty, A. Nieva de la Hidalga, M. P. Balcazar Vargas, S. Sufi, and C. Goble, “The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud,” Nucleic Acids Research, vol. 41, no. W1, pp. W557–W561, 2013. [Online]. Available: http://nar.oxfordjournals.org/content/41/W1/W557.abstract

[31] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. A. Lee, J. Tao, and Y. Zhao, “Scientific workflow management and the Kepler system,” Concurrency and Computation: Practice andExperience, vol. 18, no. 10, pp. 1039–1065, August 2006. [Online]. Available: http://dx.doi.org/10.1002/cpe.994

[32] M. R. Berthold, N. Cebron, F. Dill, T. R. Gabriel, T. Kötter, T. Meinl, P. Ohl, C. Sieb, K. Thiel and B. Wiswedel: KNIME: The Konstanz Information Miner, Springer, 2008.

[33] B. Demuth, B. Schuller, S. Holl, J. Daivandy, A. Giesler, V. Huber and S. Sild: The UNICORE Rich Client: Facilitating the Automated Execution of Scientific Workflows, e-Science (e-Science), 2010 IEEE Sixth International Conference on, 2010, 238-245.

[34] A. Streit, P. Bala, A. Beck-Ratzka, K. Benedyczak, S. Bergmann, R. Breu, J. M. Daivandy, B. Demuth, A. Eifer, A. Giesler and others: UNICORE 6 - Recent and Future Advancements, Annals of Telecommunications-annales des Télécommunications, Springer, 2010, 65, 757-762.

[35] Goecks, J, Nekrutenko, A, Taylor, J and The Galaxy Team. “Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences”. Genome Biol. 2010 Aug 25;11(8):R86

[36] M. McLennan, R. Kennell, "HUBzero: A Platform for Dissemination and Collaboration in Computational Science and Engineering," Computing in Science and Engineering, 12(2), pp. 48-52, March/April, 2010.

[37] V. Ardizzone et al., "The DECIDE Science Gateway", Journal of Grid Computing (2012) 10:689-707; DOI 10.1007/s10723-012-9242-3.

[38] Ewa Deelman, Gurmeet Singh, Mei-Hui Su, James Blythe, Yolanda Gil, Carl Kesselman, Gaurang Mehta, Karan Vahi, G. Bruce Berriman, John Good, Anastasia Laity, Joseph C. Jacob, and Daniel S. Katz. 2005. Pegasus: A framework for mapping complex scientific workflows onto distributed systems. *Sci. Program.* 13, 3 (July 2005), 219-237.

[39] Marru, Suresh, Lahiru Gunathilake, Chathura Herath, Patanachai Tangchaisin, Marlon Pierce, Chris Mattmann, Raminder Singh et al. "Apache airavata: a framework for distributed applications and computational workflows." InProceedings of the 2011 ACM workshop on Gateway computing environments, pp. 21-28. ACM, 2011.

[40] Dooley, Rion et al. "Software-as-a-service: the iPlant foundation API." 5th IEEE Workshop on Many-Task Computing on Grids and Supercomputers (MTAGS) Nov. 2012.

[41] Piotr Dziubecki, Piotr Grabowski , Michał Krysiński, Tomasz Kuczyński, Krzysztof Kurowski, Dawid Szejnfeld. Easy Development and Integration of Science Gateways with Vine Toolkit. Journal of Grid Computing, December 2012, Volume 10, Issue 4, pp 631-645

[42] Sandra Gesing, Rion Dooley, Marlon Pierce, Jens Krüger, Richard Grunzke, Sonja Herres-Pawlis and Alexander Hoffmann. Gathering Requirements for Advancing Simulations in HPC Infrastructures via Science Gateways. Future Generation Computing Systems (accepted).

[43] 830-1998 - IEEE Recommended Practice for Software Requirements Specifications https://standards.ieee.org/findstds/standard/830-1998.html

[44] Krüger, R. Grunzke, S. Gesing, S. Breuers, A. Brinkmann, L. de la Garza, O. Kohlbacher, M. Kruse, W. E. Nagel, L. Packschies, R. Müller-Pfefferkorn, P. Schäfer, C. Schärfe, T. Steinke, T. Schlemmer, K. D. Warzecha, A. Zink and S. Herres-Pawlis. “The MoSGrid Science Gateway – A Complete Solution for Molecular Simulations”, Journal of Chemical Theory and Computation, 2014, 10(6), 2232–2245.

[45] Plankensteiner, K., Prodan, R., Janetschek, M., Fahringer, T., Montagnat, J., Rogers, D., Harvey, I., Taylor, I., Balaskó, A., and Kacsuk, P. “Fine-Grain Interoperability of Scienti-

fic Workflows in Distributed Computing Infrastructures" J Grid Computing (2013) 11: 429. doi:10.1007/s10723-013-9261-8

[46] SHIWA (SHaring Interoperable Workflows for Large-scale Scientic Simulations on Available DCIs). http://www.shiwa-workflow.eu/project, 2016.

[47] Gesing, S., Herres-Pawlis, S., Birkenheuer, G., Brinkmann, A., Grunzke, R., Kacsuk, P., Kohlbacher, O., Kozlovszky, M., Krüger, J., Müller-Pfefferkorn, R., Schäfer, P., and Steinke, T. "A Science Gateway Getting Ready for Serving the International Molecular Simulation Community". Proceedings of Science, PoS(EGICF12-EMITC2)050, 2012.

[48] Jun Zhao, Jose Manuel Gomez-Perez, Khalid Belhajjame, Graham Klyne, Esteban Garcia-Cuesta, Austin Garrido, Kristina Hettne, Maree Roos, David De Roure, and Carole Goble. "Why workflows break understanding and combating decay in Taverna workflows". In E-Science (e-Science), 2012 IEEE 8th International Conference on, pages 1–9. IEEE, 2012.

[49] EGI – European Grid Infrastructure, http://www.egi.eu/, 2016 (accessed 29.02.2016).

[50] XSEDE, https://www.xsede.org/, 2016 (accessed 29.02.2016).

[51] PRACE, http://www.prace-ri.eu/, 2016 (accessed 29.02.2016).

[52] InCommon. https://www.incommon.org/, 2016.

[53] Docker. https://www.docker.com/, 2016.