eSciMart:  Web Platform for Scientific Software Marketplace

A P Kryukov\textsuperscript{1} and A P Demichev\textsuperscript{2}

Skobeltsyn Institute of Nuclear Physics, Lomonosov Moscow State University, Leninskie gory, 1, Moscow, 119991, Russia

E-mail: \textsuperscript{1}kryukov@theory.sinp.msu.ru, \textsuperscript{2}demichev@theory.sinp.msu.ru

Abstract. In this paper we suggest a design of a web marketplace where users of scientific application software and databases, presented in the form of web services, as well as their providers will have presence simultaneously. The model, which will be the basis for the web marketplace is close to the customer-to-customer (C2C) model, which has been successfully used, for example, on the auction sites such as eBay (ebay.com). Unlike the classical model of C2C the suggested marketplace focuses on application software in the form of web services, and standardization of API through which application software will be integrated into the web marketplace. A prototype of such a platform, entitled eSciMart, is currently being developed at SINP MSU.

1. Introduction

In many cases researchers need to run a large number of similar computing tasks to solve some problem in a particular application area. In this case, pre-installation and configuration of commonly used application packages on a computing resource and the availability of specialized web services as well as appropriate user web interfaces allow owners of computing resources to increase efficiency of their use (Software as a Service model; SaaS). The set of specialized interconnected web services and web-application interfaces is called a web platform for remote access to computing resources. Development of web platforms for scientific research in different areas are widely underway around the world (see, e.g., [1] and refs therein). Examples of such developments are nanoHUB (nanohub.org), Yabi (ccg.murdoch.edu.au/yabi) and e-Science Central (www.esciencecentral.co.uk). In particular, such a platform could be very useful for Monte Carlo simulation problems with parameters sweep in the field of high energy physics.

There exist many software applications that were created by scientists and engineers. However, most of them are not available for public access for various reasons. A number of web portals, including those mentioned in the preceding paragraph, are intended for providing free software. It is clear that a distribution on such a basis of a software suitable for a full-fledged research, especially for commercial use, is impossible. Among other things, this is the reason that NanoHUB has focused on educational purposes and, accordingly, the relatively simple software that is unsuitable for commercial use. On the other hand, many developers of application software or databases in various branches of knowledge have the intention to provide its product for use by other researchers, but under certain conditions, in particular, of receiving a revenue for the use of the provided product. The problem is that the software developers
(individual programmers or small teams) as a matter of rule have no time as well as financial and administrative resources to develop and support market infrastructure in full scope.

Therefore, the basic idea of the work, presented in this paper, is to develop a web platform where users of resources (application software and databases), presented in the form of web services, and their providers be able to directly interact, while the web platform have to perform a range of tasks typical for the free open market. A prototype of such a platform, entitled eSciMart, is currently being developed at SINP MSU.

2. Web-Platform Basic Principles, Architecture and Algorithms

The Internet witnesses the unprecedented boom of e-commerce in the framework of customer-to-customer (C2C) model. This type of e-commerce involves electronically-facilitated transactions between individuals, often through a third party. One common example is online auctions, such as Ebay, where an individual can list an item for sale and other individuals can bid to purchase it. Another example is the site italki.com which provides a free marketplace for teachers and learners of a foreign language (distance learning via Internet, e.g. via Skype; more than 2000000 learners of foreign languages have used this site). C2C is expected to increase in the future because it minimises the costs of using third parties.

Unlike the classical model of C2C the marketplace, suggested in our work, focuses on application software in the form of web services. We shall use the following nomenclature: consumer is a platform user requesting software as a service via the platform, provider is a user providing a software to consumers by means of the platform. There are two ways for offering an application software by means of the web platform. In the first case, the software is deployed on provider’s own resources. Thus the provider offers not only a software, but also the computing resources. In this case the provider can use different types of resources, e.g., supercomputers, clusters, standalone servers, cloud systems. Of course, these computing facilities are not necessarily in property of the provider but can be leased resources (e.g., in a public cloud). This way is especially suits those providers who want to exclude completely access both users and platform administrators to the software (both to source code and binary executables). It is worth noting that in the case of provider’s own resources a problem of reliability of delivery of services may appear. In the second case, the software is deployed on platform resources. In this case provider offers only a software as a service; a fee for the use of computing resources goes to the owners of the platform resources. The web platform have to provide a service for remote installation of the application software by users (providers) themselves. In this case the type of the resources is practically limited to the cloud systems. Indeed, an independent remote installation of application software on supercomputers by users is highly problematic both in terms of security and from a technical point of view: for a deployment of a software the user must deeply know the features of the architecture and software of the specific supercomputer. A natural solution to these problems is the use of virtual machines [2] that provide both safety (isolation of installed software) and feasibility of using the operating system required for the application software to be installed. Thus, since cloud infrastructure provides tools for managing virtual machines, the web platforms allowing software installation must provide access to the cloud systems in the framework “Platform as a Service” (PaaS) model.

General architecture of the web platform for the marketplace have the three-layer architecture (see fig. 1).

The first layer is the frontend that provides the user web interface; the second layer consists of the platform engine responsible for job submitting, obtaining results, execution of workflows, etc., the system of remote software deployment, the administration system (responsible for user management, tool configuration, audit trails, etc.) and the system of providers/consumers interactions; the third layer is a resource manager that exposes data and compute resources to
Figure 1. A general architecture of the web marketplace

the preceding layer. Also the web platform has security infrastructure (see below).

Each of the systems of the second layer, in turn, consists of a set of the functional modules. For example, the system of job submission consists of modules of task launching, workflow processing, access to scientific databases, etc. (see, e.g., [1] and refs therein). The most important system from the point of view of the purposes and tasks of the web marketplace is the system of providers/consumers interactions.

The list of modules of this system includes: module of software registration; service catalog with search engine; logging (event tracking) module; billing module; integrator of conditions for workflows; module of verifying compliance with the terms of service; software deployment module; module of a tool setup; module of e-wallets; repository of the software codes; module for review and blog posting and for assigning rating marks.

A standardization of API through which application software is integrated into the web marketplace is realized by means of the process of software-into-tool transformation. An individual prepared job (launch of an application software package, access to a storage, etc.) is called a “tool”. After a deployment of a software on a resource, the creation of a associate tool starts from its description in a machine readable form via a special web interface (web forms) including the following information:

- unique name of a tool;
- name of an application package installed;
- parameters of the tool (and their acceptable values);
- formats of input and output files for the tool;
- description of particular requirements for input files;
- control of a user (or user group) access to the tool;

as well as presetting all technical details, in particular,

- web addresses of resources where the software is installed,
- storage to save output files,
- particular formats of database queries, etc.

In fact, the creation of a tool together with the platform functionality convert the associated application software or database into SaaS.

Having access to tools provided by a web platform, the user only specifies particular values of input parameters or files containing input data; the rest of the job description is generated automatically. Thus, the preparation of a job, its submission, execution control, and obtainment
of the result are done using a common web browser. In addition to launching application packages, a tool can include other operations, for example, database query, visualization of results, and obtaining of on-line data from a scientific installation. All that is left for users to do is to formulate in the natural language the essence of a particular request. It is especially important that web services and corresponding interfaces allow users to avoid dipping into the intricacies of installing and launching each of these packages, so they can concentrate only on research problems.

Algorithm of the platform operation when a provider is offering his SaaS is depicted in figure 2. This figure corresponds to the case when the provider uses the cloud resources of the platform for software deployment. In the case of provider’s own resources, the provider deploys the software on this resources and provide an access to the software via the SSH protocol. If the resource is not a standalone server but an HPC system with a batch manager, the latter should be in the list of managers supported by the platform. It is worth noting that the platform supports workflows, which consist of sequences of tasks (for example, access to databases with subsequent use of results as input data for computing tasks). Thus a provider may not only a single tool but also debugged and ready to execution workflows, constructed of provider’s own application software or/and software from third-party providers in the spirit of the BPaaS model [3].

![Figure 2. The algorithm of a provider and market modules operation.](image-url)

Each user of the platform (both a customer and a provider) has the electronic (virtual) wallet (e-wallet) containing the conditional means (digital cash) within a platform. Users can transfer their funds to the e-wallets of other users. In reality users also shall have opportunity to transfer funds from the e-wallets to real payment systems (like PayPal) or to bank accounts, but in this work we do not consider this process.

To use the platform tools, a consumer chooses from the catalog appropriate tools; designs out of these tools a workflow or select predefine business process (BPaaS); gets terms of services and price for their use; accepts or deny the terms of conditions; if accepts, transfer digital money from his e-wallet to the provider’s e-wallet (the module of verifying compliance with the terms of services checks that the conditions are accepted and ample means have been transferred to the providers’ e-wallets); starts the workflow and obtains a result.

An administrator of the web platform provides user registration, moderation of software installation, user support and Quality of Service (QoS) obligations from the side of providers offering their own computing resources.

The web marketplace together with computer resources where the application software and databases are deployed is a special case of the distributed computing systems (DCS). To provide secure access to the resources of DCS with the account of the rights of a given user as well as of the politics of a given service or resource a DCS security infrastructure is needed. This
infrastructure should be sufficiently reliable and, at the same time, user friendly. Within creation of the DCS, the new technique of authentication of users [4] which is based on login/password pair together with a sessional key is developed. Key difference of this method from the most popular approach based on public key infrastructures (PKI) [5] is the non-utilization of proxy certificates for execution of user requests. Another new principle which is suggested to be used for security arrangement is the use of hashes for signing requests in the framework of a DCS. For confirmation of legality of each request in DCS signatures in the form of specially created unique hashes with unlimited validity period are used. The use of the unique hashes allows to solve, on the one hand, a problem of protection of request contents while processing within DCS, and, on the other hand, solves the problems caused by limited period of validity of usual proxy certificates. Registration of the unique hashes allows to provide authentication and authorization of requests.

In general the suggested approach leads to essential simplification both the registration of new users in the system, and their operations in the distributed system. Some lowering of the security caused by the sessional key without password is compensated by restriction of its validity time. After validity expiration of the key, the user requests a new key either via the special web interface, or through API of the appropriate service.

3. Conclusion
A prototype of the web marketplace, presented in this paper and entitled eSciMart, is currently being developed at SINP MSU. The basis of the prototype platform is web service technologies and, in particular, the service-oriented architecture (SOA) [6], as well as the REST architectural style [7]. For the data interchange between services the JSON format (http://www.json.com) is used. The approach to creation of this web-platform market of application software is based both on the original solutions and on the synthesis and adaptation of the solutions used in research hubs (e.g., nanoHUB; nanohub.org), cloud and grid systems, as well as in on-line application stores and auctions.

Acknowledgments
This work is supported by the RFBR grant No.15-07-09309.

References
[1] Polyakov S P, Demichev A P, and Kryukov A P 2015 Procedia Computer Science 66 429
[2] James E S and Ravi N 2005 Virtual Machines: Versatile Platforms For Systems And Processes (San Francisco: Morgan Kaufmann)
[3] Barton T and Seel C 2014 Enterprise Modelling and Information Systems Architectures (EMISA 2014). Lecture Notes in Informatics Proceedings Series of the German Informatics society vol P-234 (Bonn: Kollen Druck & Verlag GmbH) p145
[4] Dubenskaya J Demichev A Kryukov A and Prikhodko N 2015 Procedia Computer Science 66 525
[5] Buchmann J A Karatsiolis E and Wiesmaier A 2013 Introduction to Public Key Infrastructures (Heidelberg: Springer)
[6] Erl T 2009 SOA Design Patterns (New York: Prentice Hall)
[7] Richardson L and Ruby S 2007 RESTful Web Services (Cambridge: O’Reilly Media)