ScienceSoft: Open Software for Open Science

An Open Community of People, Software and Services for Scientific Research

v. 0.6

Authors

The ScienceSoft Steering Committee

Geneva, 29 February 2012

Table of Contents

Executive summary

There is a wealth of open source software in use across scientific communities but the value of its contribution to science is under-estimated, under-utilised and often poorly coordinated. Some websites such as ohloh (http://www.ohloh.net/) offer directories that attempt to rate the quality and impact of open source software projects, but currently lack the means of attracting developers and users from academic communities and harvesting a large enough body of essential data to make their results meaningful for the scientific research environments. Being able to aggregate the power of cataloguing services, trends and statistics would provide a sound basis for judging the popularity of specific software, enable social-networking amongst users and developers, create active communities and promote citizen science. Rating software and providing a means by which it can be cited in a similar manner to publications and datasets would enable the authors to gain merit and career advancement for their work and accelerate the open source software movement in scientific communities. Being able to quantify the impact of open source software would allow funding agencies, companies and venture capitalists to better target their investments leading to a more vibrant and sustainable open source market for open science.

This document outlines the state-of-the-art of open source activities in the scientific research environments and describes a number of problems and potential solutions identified by discussing with user and developers of existing projects and communities. The potential benefits of the proposed solutions are described and how scientists, developer, administrators, managers and funding bodies could exploit them to take decisions and plan their activities.

History

During September 2011 the EMI project started a discussion on a general proposal to investigate, design and establish an open source software initiative as part of the EMI long-term sustainability plans. The main objective of this proposal was initially to create the conditions for the continuing development, support and use of the EMI software products after the end of the EMI project by establishing a broad open source community of developers and users.

However, it became rapidly clear from discussing with users and developers from scientific communities and research projects that similar concerns about long-term sustainability were shared by many other projects developing or using middleware, applications, tools and related services of critical interest for the European scientific research communities.

The discussion on establishing a broad open source community around software for scientific research was therefore extended to a number of interested parties. The goal is to define and setup a truly open community of software developers, administrators and users in the context of global scientific research by including scientists, developers, service providers, research institutes and commercial companies in the process since its inception.

In December 2011, the ScienceSoft Steering Committee (SC) was formed with participants from a number of projects, communities and SMEs. The mandate of this group was to bring into the discussion their particular perspective and refine the initial concept and ideas. On February 8th, 2012 most SC members and a number of additional interested participants have attended a workshop at CERN (https://indico.cern.ch/conferenceDisplay.py?ovw=True&confId=160503) where the initial definitions of problems, solutions and possible next steps have been discussed.

This document summarizes the work done since September 2011 with particular reference to the outcome of the CERN Workshop.

Market Analysis

As a starting point for the discussion, a brief market analysis was performed. This analysis is probably biased towards current research grids more than other types of scientific computing and data environments. However it is general enough to be easily adapted and extended as necessary.

Five types of computing and data environments have been considered:

1) Grid (High-Throughput) computing
2) High-Performance Computing
3) Cloud computing
4) Desktop or volunteer computing
5) Stand-alone computing

Combination of these basic types can be considered as well, like for example the provision of grid services using dynamically provisioned IaaS clouds or the provision of cloud-like access to grid resources.

The software layers involved in the use of the selected types of environments include:

1) Operating system software
2) Middleware software
3) Service management software
4) Applications and community specific services
5) User interfaces and portals
6) Tests, tools, benchmarks and other supporting software

The roles involved in the use of the software can further be classified as:

1) Developers
2) Users
3) Service providers
4) System engineers, platform integrators
5) Institutes and Companies
6) Collaborations (projects, initiatives, communities, virtual organizations, etc.)
7) Funding bodies

The roles are not mutually exclusive. On the contrary the complex relationship among people in various roles is indeed one of the fundamental issues that ScienceSoft proposes to investigate. For example the developer of a software product can be at the same time the user of another product (a dependency) or a service (a testing service or a cloud IaaS service) and a provider of applications for a group of scientific researchers.

In this classification, the different roles have different problems and are looking for different specific solutions. A consistent proposal must be able to take into account and express the diversity across the market (vertical dimension), while at the same time recognising the commonalities across software layers (horizontal dimension). The distinction between vertical and horizontal dimensions will be further explored later on in this document.

Figure 1 illustrates the previously defined categories and their relationships.

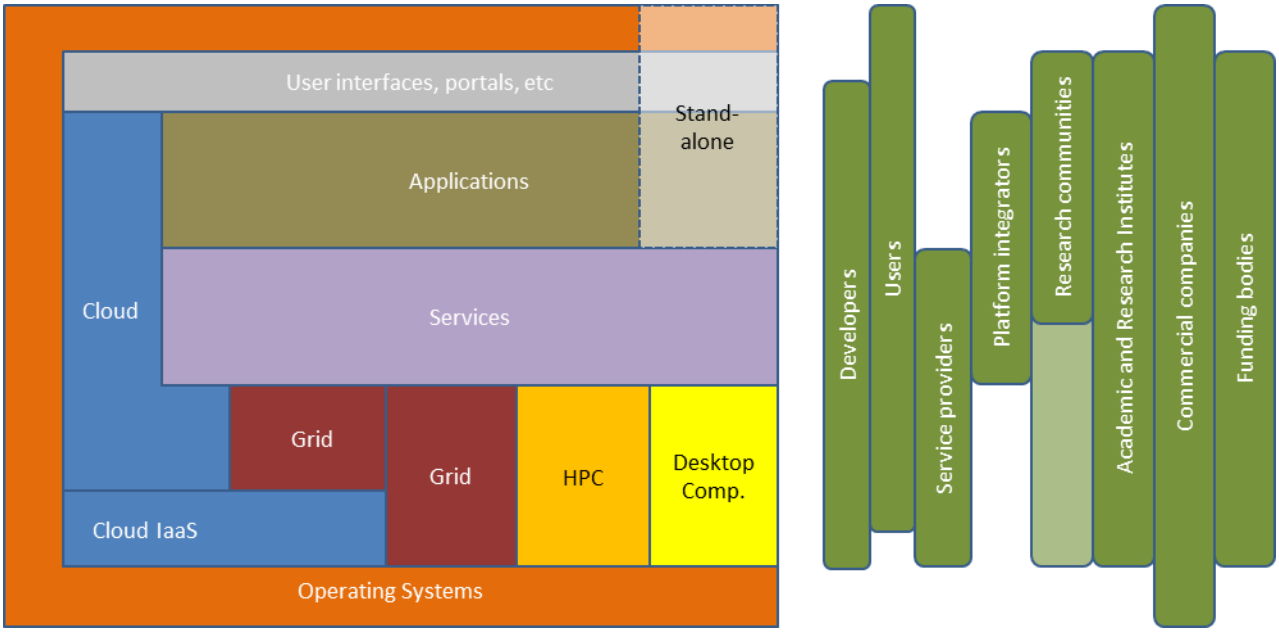

Figure 1: The software and services market classification

Identified Problems

Users and developers of software for scientific research projects have been contacted during conferences and presentations and asked about what problems they are confronted with in their normal working activities. Among the identified problems the following are considered the most critical:

- **Lack of continuity in support, development, coordination of software**: this issue is mostly felt by users and developers of software developed by short-term projects. Users face difficulties in adopting or relying on software without longer term support commitments. At the same time developers, who may be willing to continue developing and supporting the software, find themselves without a repository or release channels and have no easy way of keeping contact with their users.
- **Non-optimal communication between users and developers**: most projects tend to be rather horizontal and focus on specific, limited aspects of the scientific research software stack. Although this is a quite natural approach to setting up projects, it has the drawback of creating boundaries between different categories of users. The interaction happens often in occasion of conferences or other public events, but many users have expressed the need for a more direct and continuous relation with the developers in order to discuss requirements, features, issues, etc. It is felt that there is excessive formalization and loss of information due to being in different projects.
- **Lack of consistent real usage information**: although it is relatively easy to count downloads from a web site, information on actual usage is much more difficult to collect. In a number of cases, information systems are available to provide this data, but not for all potential services

and applications. The lack of this information prevents understanding the actual user base and evaluating the relative importance and impact of the produced software

- **Limited access to other users' experience**: before using an application or a software service and investing time and money in them, users would like to know what other people think of those applications or services, what problems have been found, if and how they have been solved. Problems solved in one case can be applied in other cases and save time for both users and support people
- **Limited or complex ways of finding what exists already**: a lot of duplication is often found in the software production. In many cases the duplication is due to real needs of producing something that works better or fits better a community need, but quite often it is due to the lack of knowledge of what exist already. Having information about existing software and services, who develops and supports them and who uses them may help taking informed decisions and quickly bootstrap new projects without waste of time and resources
- **No way of influencing the production of software**: many users feel that they have little saying on what functionality software should have or what software should be supported and which should be stopped. Although functionality and quality are always a matter of negotiation and priorities, the lack of the possibility of expressing opinions often drives users away or prompt them to develop their own solutions in the hope of having more control
- **Lack of visibility of the software activities**: many software developers working in research institutes produce fundamental tools used by scientists to make important discoveries. Academic recognition is usually measured in terms of publications and the contributions of programs and tools cannot be formally acknowledged with a reliable citation system. Having formal ways of citing and listing software used in conjunction with published scientific results would increase motivation and help career recognition and development
- **No way of assessing the user "market"**: the main of foremost barrier to the involvement of commercial ventures into the development and support of software and services for scientific research is the difficulty of sizing the market and the potential revenue streams. Having ways of performing realistic market analysis and offering targeted services to the scientific communities may help SMEs if not larger companies to define achievable business plans

Most of the software developed today by research institutes, university, research projects, etc. is typically stored in local source and binary repositories and readily available for the duration of the project lifetime only. Finding software based on given functional characteristics or field of application is very difficult especially for new projects or young researchers. Binaries to be run on the most used operating systems are available from many different places ranging from local university repositories to mainstream community repositories like EPEL or Debian. Cases of conflicts are often found between different versions distributed by different people from different places. Source code is even more difficult to locate and access and contributing with comments, patches and fixes, which is a very common activity in the open source world, is traditionally very difficult to do in the research communities. This has been for years a primary complaint from users.

Similar requirements have been expressed by application developers, infrastructure managers and users. Within the HPC community the organization of common repositories of application code is a known concern. Although such code is usually highly dependent on the hardware on which they are run, the lack of code sharing and availability is considered one of the reasons why the European HPC efforts are falling behind similar activities in the US and Asia (see the recent IDC report presented at the EGI Technical Forum in Lyon in September 2011[1]).

Most of the reported problems can be categorized as a lack of consistent and transparent information about the software being used in scientific research. The problem is not necessarily a lack of technical information (such as documentation or user guides, although this has also been described as a problem in many cases), but rather a lack of metadata.

Information about who develops, contributes and uses a given program is very difficult to find out and yet the widespread availability of such information would give more visibility and credibility to the software products. In addition, the EC invests considerable amounts of money into funding projects that directly or indirectly need to develop software. A single repository of metadata information about software products would allow projects to avoid re-developing existing solutions and would provide valuable statistics about software usage. Such information could be used also by the EC to monitor the outcome and impact of funded projects, the extension of adoption of open source software and the compliance with OSI licenses and possible as input to future EC calls objectives and framework programmes. In the same way, the information could give more strength and credibility to project proposals, which could be backed by realistic information about usage, impact and exploitation of the software.

Possible solutions and benefits

Functionality

From the discussions with users and developers and the outcome of the CERN Workshop in February, a number of desirable functionality has been defined:

- **Software, services and people catalogues**: the first and foremost desired functionality is the provision of catalogues of information about software products, software-related services and people. The catalogues should provide information based on tags or taxonomies that allow to group together sets of related products, services and people based on flexible search criteria. The provision of technical metrics about software and services is also desired, for example license, programming languages, compatibility, supported systems, etc.

[1] Steve Conway (IDC Research Vice-President, HPC), Financing a Software Infrastructure for Highly Parallelized Codes, 9th eConcertation Meeting Lyon, September 22-23, 2011

- **Generation of statistics**: the information collected and processed should not only be used to search about software, but also to general relevant and useful statistics. The most requested statistics are for example actual usage information, as opposed to downloads of packages from a web site, geographical distribution of usage and production, involvement of Institutes and Companies in projects based on scientific discipline, etc.
- **Honour system**: community users should be able to rate the registered software and services based on their experience. Ratings can be provided based on predefined categories such as reliability, support quality, documentation, ease-of-use, standards support, etc. Ratings should be supported by comments describing the user experience with the software or service
- **Citation system to allow software to be referenced in papers**: registered software should receive some sort of unique identifier like the DOIs used for papers, so they can be reliably cited in scientific publications.
- **Marketplace for products, services, and people**: this is one of the most interesting features and one that may well define a community. Matching demand and offer of software products, service and people skills should be enabled based on the catalogues maintained by the community tools. Some form of advertising could be considered both for non-profit and for profit activities, although different models should be envisaged in this case
- **Links to technical services**: one of the marketplace-related activities is the provision of technical services. A range of services could be designed and provided by community members and provided to other members for free or for a fee depending on conditions. Such services could go from straight access to IaaS resources, to software testing services, to consultancy and support, etc. Services can be generic or community-specific. The common requirement is to provide a way to advertise, promote and access the services
- **Platform integration support**: using the collected information and the product catalogues, it should be possible to define community-specific software stacks to be supported by platform integrators. These community-specific profiles or stacks can then be pre-packaged and easily deployed using the more and more standard virtualization and cloud technologies
- **Support for creation of ad-hoc communities and groups**: this requirement is what may actually define ScienceSoft as a community-enabler. ScienceSoft per se should act a super-community provide a framework and tools to enable more specific communities to interact without being isolated from other communities. Cross-community groups, for example for common activities like security, configuration, etc. could also be created and populated with the relevant products and people
- **Coordination, collaboration and discussion tools**: collaboration tools typical of distributed online infrastructures should be available through the ScienceSoft community portal. It should be possible for each hosted community to have independent channels, but a general social network for science should be supported allowing direct interaction among users in addition to standard forums
- **Support for organization of technical events**: it should be possible to advertise and manage technical events related to the hosted communities or projects. Support can range from dedicated event pages, to agendas, advertising, collection of material, etc.

Benefits

The creation of links or relationships not only among pieces of software, but equally among the people interacting with the software, would foster a more active community and create the conditions for sharing ideas and skills and a more rapid improvements of the software quality. The use of modern social networking techniques would greatly help the establishment of active open source communities and focused sub-communities around specific scientific and technical interests. For example, sub-communities could be established for people interested in testing or in writing documentation, communities of packagers, or experts of specific standards or security and so on.

The possibility of sizing and profiling the usage of software by user communities would potentially allow commercial companies to offer added-value services based on concrete needs.

The availability of value-added information about software and its usage would bring several benefits. It would allow Institutes to perform more realistic assessment of costs and optimize resources by focusing on unique propositions rather than duplicating existing software. The possibility of concretely sizing the user base of an application or a service would provide tangible supporting evidence for funding requests with concrete impact analysis. Open source software licenses adoption and compliance could be verified in order to enforce legal requirements.

The establishment of a software rating system based on both technical criteria (Is a given platform supported? Is a certain package format available?) and usability criteria (What do existing users think of it? Is documentation up-to-date and well written?) would allow filtering mature products from less mature products and would increase the developers' motivation to improve certain aspects of their software like documentation that traditionally receive less effort that the actual code writing.

An interesting functionality is the possibility of creating community-specific virtual software stacks using the software catalogues. Once the profile is defined, it can be kept updated as the products evolve and dedicated community integrator could provide pre-packaged appliances to be shared for example through appliance marketplaces like the one implemented by StratusLab.

State-of-the-Art

The first approach being considered to address the described issues and provide the desired functionality is to exploit lessons learned from successful open source software communities.

Most of the software used in scientific research and developed by academic institutes is generically "open source" in the sense that it uses some type of OSI license. However, it takes more that source code and a license to have a "community". The general definition of an open source software community is a group of developers and users interacting to produce free-software. The interaction among users and developers, the sharing of resources and common objectives and the benefits deriving from sharing are of course fundamental to have a community and not just software in a repository.

We can distinguish primarily between four different types of open source communities:

- **Technology-specific (or horizontal) projects**: this type of communities includes projects focused around a specific technology or framework which all members contribute to. Usually the membership rules are quite stringent both in technological and legal terms. For example it's not uncommon to have to adopt a mandated IP model and a license for all contributing products. Notable examples of this category are the Apache Foundation, the Eclipse Foundation or the Drupal Association
- **Operating system distributions**: communities focused around different flavours of Linux operating systems have been among the first to emerge and have in many cases enabled most profitable open source business models. Although in general there is no formal membership into these communities, the engagement rules to contribute are quite strict and require a peer-review level of competence and quality for both contributors and products. Most notable examples of this category are Fedora, EPEL, Debian, CentOS, etc.
- **Services and tools**: these open source communities usually provide a software application and often services based on that application. They have dual usage models, whereby access to the service is free for personal or non-commercial use, while professional use is charged a fee. Most notable examples include SourceForge, GitHub, Zarafa and many others.
- **End-to-end (or vertical) open source communities**: at the end of 2011 Andrew Aitken, president of the Olliance Group, a leading open source consulting firms, wrote an article about the appearance of "super-communities" or communities of communities[2]. The super-communities instead of focusing on a particular piece of open source technology are built around the entire end-to-end supply chain of an industrial sector, like the aerospace industry (Polarsys launched by Airbus), stock exchange management (OpenMama launched by the New York Stock Exchange) or OSEHRA (electronic healthcare records launched by the US Department of Veteran Affairs). The Olliance Group predits that this kind of communities will rapidly increase in number. Microsoft launched in 2011 the OuterCurve community to host open source software and communities and provide general-purpose IP management services with a focus on Windows applications. In the scientific communities, portal like NanoHub or CyberSKA focus on the general needs of the nano technologies and radio astronomy communities respectively.

The software produced and used in scientific applications is by its nature very diverse. It uses different programming models, technologies, IP and licensing models. In addition the end users, the scientists using the software to perform their research are not overly interest in what technologies are used under the hood, but are very concerned with having a working set of tools. The first three types of open source communities described about could therefore be used for parts of the software produced in the academic world, but would t bring the level of communication and organization needed to provide the

[2] http://opensourcedelivers.com/2011/12/20/the-advent-of-super-communities/

functionality described earlier. A suitable model for ScienceSoft could therefore be the fourth one, where the overall end-to-end software needs of specific scientific communities could be modelled and addressed with a more global approach than just individual pieces of software.

ScienceSoft Organization, Structure, Operations

Based on the preceding analysis the ScienceSoft super-community could therefore be configured as a community of users and developers of software targeted at scientific applications. The community members contribute software and information and provide services to other members. The members within ScienceSoft are organized in focused communities or collaborations around a specific scientific or research topic. People, software, services, etc. are tagged within the super-communities based on their particular focus in order to build community-specific sets of resources. Resource tagging is not exclusive. The same resource can be tagged with more than one community focus, so that it becomes possible to understand the overall usage of that resource within a more general scientific context.

A community portal gives access to the community services like member and product registration and management and the different functionality described earlier in the document.

Most of the common base functionality required to operate the ScienceSoft portal already exists in some form. As a first implementation, the ScienceSoft portal can be configured as an aggregation of such functionality within a coherent container. Functionality like software inventories, source-code repositories, social networking, forums, etc., can be provided in this way using existing open source services like ohloh.net or inventories, Drupal modules for the web based collaboration tools and existing social networks like FaceBook, Linkedin or Google+ for user management.

The community specific services would instead be provided by the members through links or applications running in the portal. Community-specific micro-sites can be easily established from common templates to create well-defined identities and focused sets of people, programs, services, tools, information, etc.

The governance structure for ScienceSoft could therefore be based on two general roles:

- ScienceSoft maintainers: the maintainers are in charge of implementing and supporting the portal and the base common features by integrating as much as possible existing functionality from other compatible open source initiatives, where compatible means that the license and usage rights must allow the integration within the ScienceSoft portal. The maintainers could also provide support for general items like community templates, although it can be envisaged that such functionality would be also contributed by other community members
- ScienceSoft contributors: any ScienceSoft community member providing information, software or services to ScienceSoft or one or more communities hosted within ScienceSoft. This category includes the people responsible to defined and maintain collaborations or communities within

ScienceSoft. For example the EMI Collaboration, the Grid and Cloud Security Group, the Earthquake Modelling and Prediction project could be hosted communities with one or more person managing their definition, memberships, activities, software stacks, etc.

Although the initial organization and governance structure of ScienceSoft should be as lean and lightweight as possible, it is foreseen that depending on the success and evolution of its activities it could become in the future a Not-for-Profit Foundation on the model of existing successful open source foundation.

Current Activities and Timeline

The requirements and possible implementation strategies for Science Soft are currently being investigated by the ScienceSoft Steering Committee. Initial requirements and desired functionality have been discussed in occasion of the ScienceSoft Workshop held at CERN in February 2012. The outcome of the workshop has been incorporated in this document. The workshop participants have agreed to keep providing feedback and collaborate in the definition of implementation priorities. The proposed timeline is as follows:

- March to June 2012: definition of priorities and identification of volunteers ScienceSoft maintainers
- July to December 2012: progressive implementation of a prototype community portal, dissemination, engagement of scientific communities in trying the functionality and providing feedback
- January to April 2013: start of the regular activities, further requirements and implementation cycles. Until this date the community is incubated with the EMI project, which provides overall coordination
- May 2013 onward: regular operations, fund raising for continuing activities based on the success of the initiative. Phase down and discontinuation if no interest has emerged.