Accepted Manuscript

Title: Cloud computing infrastructure for the VPH community

Authors: Piotr Nowakowski, Marian Bubak, Tomasz Bartyński, Tomasz Gubała, Daniel Harężlak, Marek Kasztelnik, Maciej Malawski, Jan Meizner

PII: S1877-7503(17)30733-0
DOI: http://dx.doi.org/doi:10.1016/j.jocs.2017.06.012
Reference: JOCS 711

To appear in:

Received date: 5-1-2017
Revised date: 18-5-2017
Accepted date: 19-6-2017

Please cite this article as: Piotr Nowakowski, Marian Bubak, Tomasz Bartyński, Tomasz Gubała, Daniel Harężlak, Marek Kasztelnik, Maciej Malawski, Jan Meizner, Cloud computing infrastructure for the VPH community, Journal of Computational Sciencehttp://dx.doi.org/10.1016/j.jocs.2017.06.012

Cloud computing infrastructure for the VPH community

Piotr Nowakowski[1], Marian Bubak[2,1], Tomasz Bartyński[1], Tomasz Gubała[1], Daniel Harężlak[1], Marek Kasztelnik[1], Maciej Malawski[2,1], Jan Meizner[1]

[1]*ACC Cyfronet AGH, Nawojki 11, Krakow, Poland*

[2]*Department of Computer Science, AGH University of Science and Technology, al. A. Mickiewicza 30, Krakow, Poland*

Correspondence: p.nowakowski@cyfronet.pl, ul. Nawojki 11, 30-950 Kraków, Poland.

Highlights'

The presented paper includes the following highlights:

- Thorough analysis of computational and storage requirements of the VPH community
- Conceptual description of using cloud resources to share simulation modules
- Practical implementation of a hybrid cloud platform to facilitate development, sharing and management of computational services
- Successful deployment of important application workflows (@neurist, MySpine and others), with hundreds of users spawning thousands of service instances on a monthly basis

ABSTRACT

As virtualization technologies mature and become ever more widespread, cloud computing has emerged as a promising paradigm for e-science. In order to facilitate successful application of cloud computing in scientific research – particularly in a domain as security-minded as medical research –several technical challenges need to be addressed. This paper reports on the successful deployment and utilization of a cloud computing platform for the Virtual Physiological Human (VPH) research community, originating in the VPH-Share project and continuing beyond the end of this project. The platform tackles technical issues involved in porting existing desktop applications to the cloud environment and constitutes a uniform research space where application services can be developed, stored, accessed and shared using a variety of computational infrastructures. The paper also presents examples of application workflows which make use of the presented infrastructure – both internal and external to the VPH community.

HIGHLIGHTS

- Thorough analysis of computational and storage requirements of the VPH community
- Conceptual description of using cloud resources to share simulation modules
- Practical implementation of a hybrid cloud platform to facilitate development, sharing and management of computational services

- Several important application workflows (@neurist, MySpine and others) have successfully exploited the platform, with hundreds of users spawning thousands of service instances on a monthly basis

Keywords: Cloud computing; distributed scientific infrastructures; e-Science; virtualization

1. INTRODUCTION

The advent of modern distributed computing technologies, cloud computing being chief among them, offers new opportunities for developers and users of scientific applications, including in particular the domain referred to as “midrange science” or “long tail science” [1]. This is a category of applications which do not require vast computational resources yet must still be provisioned in a stable and managed environment in order to yield useful results. While computational clouds do not constitute a replacement for traditional high performance computing (HPC) or high throughput computing (HTC) infrastructures, such as those provided by computing grids, they can and do fill an important and growing niche which includes several types of scientific applications. In this regard, the first category comprises desktop applications, i.e. tools which are intended for execution on desktop computers (or enterprise servers) and do not yield themselves easily to parallelization or do not carry requirements characteristic of HPC/HTC systems. This category has expanded significantly over the recent years and will likely continue to grow along with the growing capabilities of desktop solutions (i.e. better processing power, operating memory, network throughput etc.) Additionally, some applications are developed with the intention of being shared within the scientific community. These are predominantly based on the Web application/Representational State Transfer (REST) Services paradigm and provide a means of enabling external users to interact with executable code and scientific datasets in a manner which is suitable both for human users and client software executing in a workflow-like fashion across organizational boundaries.

In this paper the authors present their experience with deploying, provisioning and operating a computational cloud platform for the VPH community, set up under the auspices of the VPH-Share project [2]. While the project has concluded, the platform remains in operation and is actively exploited by a growing VPH user community. This community consists of users who were originally partners of the VPH-Share project consortium, as well as additional users brought on board during the course of the project and not originally affiliated with any VPH-Share project partners.

The main contributions of the paper are as follows:

- We analyse the problems faced by the biomedical community in procuring adequate computing resources to further its research and to share the computational tools and datasets involved in such research.
- We present the Atmosphere cloud platform as a solution to the abovementioned problems
- We show how the platform has been used to address real-life use cases and what tools and features it provides for the VPH community.

The paper is organized as follows: Section 2 defines the problem we’re addressing and highlights some specific challenges which need to be resolved in developing a computational platform for VPH applications. Section 3 provides an overview of related work relevant to the problem area. Section 4 contains a conceptual description of the proposed computational platform while Section 5 details the implementation and deployment of the platform. Section 6 shows how the platform has been used to serve the VPH community, and also provides specific usage examples and statistics. Section 7 summarizes the paper and presents further development prospects going beyond the VPH community.

2. PROBLEM DEFINITION

As highlighted above, the goal of the presented research was to resolve the problem of provisioning distributed resources for medical research and end-user applications. In designing a platform for medical applications care should be taken to ensure the ability to redeploy a number of existing applications (considered “black box” components) upon distributed resources, as well as to guide the development of further applications in a manner consistent with best practices in distributed computing. We believe that this situation is fairly typical of any organization in which an infrastructural change is to take place without affecting day-to-day business operations carried out by end users. As a case in point, the VPH community consists of approximately 50 research teams, each working on a set of scientific applications. Even though many of these applications were initially designed as standalone tools, with no thought given to distribution and remote access, it soon became clear that a fundamental redesign of each and every one of these workflows in order to match an idealized “checklist” of features would be impractical. Rather, the computational infrastructure would need to provide a generic toolkit to enable any existing software to be transferred to the cloud with minimum effort. This immediately led to a list of issues which had to be carefully considered in developing the platform.

A. User interfaces

In terms of user access, standalone applications – such as those being exploited by the VPH community – run the gamut from command-line clients, through web services overlaid by lightweight web clients all the way to “rich client” software, providing users with three-dimensional representations of medical objects, visualized with help from 3D graphics adapters and relying on the capabilities of state-of-the-art graphics accelerators. The cloud platform in question would need to support each of those without forcing developers to prepare customized application releases, tailored for use in computational clouds (a prohibitively complex task in the case of advanced applications and toolkits – e.g. Gimias [3], ANSYS [4] GUIs etc.)

B. Data access

Standalone applications often process data which is stored and accessed locally, in the form of files saved in the local filesystem (DICOM images, VTK, mesh files etc.) While distributed data repositories do exist (see for example Amazon S3 at http://aws.amazon.com/s3/ and RackSpace Cloud Files at http://www.rackspace.com/cloud/public/files/), forcing an existing application to make use of such tools instead of local file storage is often a nontrivial task, requiring access to the application source code.

C. Security

It is not common for a standalone application – and certainly not one developed for the purposes of scientific research – to impose an additional layer of security, restricting access to a group of authorized users. When deploying such applications in the cloud it becomes imperative to ensure that only authorized parties may access the application’s features and that the input (and output) data which the application processes is only exposed to members of that application’s user group. In addition, the legal guidelines concerning processing of medical data must be respected – to this end, the platform should conform to EU guidelines related to end-to-end data encryption and auditability in cross-border healthcare [5]. It is recognized that some data may only be processed within the confines of the institution which produced it or is responsible for maintaining it – a distributed platform should honor this restriction.

3. RELATED WORK

A significant quantity of research effort has gone into making cloud infrastructures applicable to scientific application scenarios and trying to address the issues mentioned previously. The aim of this section is to present the state of the art in the area of cloud resource provisioning for e-Science, both in terms of general-purpose IaaS/PaaS platforms (enabling arbitrary code to be deployed in cloud infrastructures) and highly customized solutions for specific applications, fulfilling their functional and non-functional requirements regarding accessibility, performance, security and ease of use. This information will then constitute a background against which our proposed architecture can be compared and evaluated.

A. Clouds in the biomedical engineering computational landscape

As stated in the introductory section, cloud computing does not, and will not in foreseeable future, provide a suitable solution to all VPH type of scientific applications. On the forefront of new physiological models development are applications which require massively parallel computing environments, on the order of millions of processing units at once [6]. Such resources are required to break new boundaries in spatial and temporal simulation resolutions, and to test more complex solvers with anticipation of better result quality. However, with time, these new models enter a class of applications for middle-range computing environments, including clouds.

This is caused not only by the growing capabilities of commodity computing installations (i.e., denser packing of processing power and data storage per unit of space). Re-implementations for more suitable architectures, like GPGPU [7], and model complexity reduction techniques [8] play a decisive role in this process. Consequently, the relevance of cloud computing, based on more performant or specialised hardware for biomedical engineering applications, is on the rise.

One additional aspect which makes such computational solutions attractive to the type of applications contemplated in this paper, is that they are naturally well suited for the diversity of tools used in the biomedical domain [9]. Development of new physiological models, and refinement and validation of existing ones, require a great deal of experimentation and call for an exploratory approach in terms of computational tools (libraries, packages, solvers) [10]. Strong autonomy of cloud-based computing environments (beneficial for both the user and the resource provider, especially security-wise), solid separation of computing instances, and relative freedom in administration of own problem solving environments (notably better than centrally-managed cluster and HPC installations) assist domain scientists in pursuing an efficient exploratory approach to various research challenges.

B. Projects and initiatives providing support for custom scientific applications

Despite the fact that the cloud market is largely dominated by commercial service providers, some clouds are dedicated purely to scientific research. Early examples of this approach with the use of small clouds included Nimbus at the University of Chicago or Stratus at the University of Florida. More production-oriented cloud services are now offered e.g. by the EGI Cloud Compute, available to European researchers [11]. On the other hand, Helix Nebula project aims at building a scientific cloud as a public-private partnership, including European cloud service providers and commercial vendors [12].

Another example of a much larger scientific cloud platform was the set of cloud services provided to scientists by the FutureGrid Project [13] and continued in Chameleon project. It offers access to various cloud platforms such as OpenStack or deployment of bare-metal machines and allows conducting computational experiments on the infrastructure it controls; however it requires users to deal directly with low-level cloud middleware.

Significant interest in cloud computing can also be observed in bioinformatics and data-intensive astronomy. The Open Commons Consortium (formerly the open Cloud Consortium) [14] is an initiative to build an open cloud infrastructure for researchers, focusing mainly on data mining applications in both of the aforementioned disciplines. Participants are required to contribute a container-based datacenter and network links. The Open Science Data Cloud supports large-scale data networks, including BioNimbus and the Sloan Digital Sky Survey. Finally, dedicated tools and consolidation frameworks have emerged that provide unified interfaces and access to various bioinformatics tools, including Galaxy [15], BioLinux [16] and IMENSE [17]. These frameworks make bioinformatics tools available and accessible to researchers; however they still require a computational infrastructure to run the analyses – the CloudMan [18] and CloudBioLinux [19] projects address this need.

As highlighted above, analysis of the available cloud and distributed computing platforms has led us to the conclusion that while individual projects and systems tackle selected issues and requirements stated in the previous section, there is currently no unified cloud-based execution platform for scientific applications which would respect the data privacy and protection issues inherent in medical data processing. In light of this fact we decided to create a customized infrastructure, tailored to the needs of medical research while drawing upon the experience and outcome of existing development initiatives.

C. General-purpose IaaS (Infrastructure as a Service) and PaaS (Platform as a Service) offerings

The synergy between computational clouds and non-HPC scientific applications (sometimes referred to as “midrange computing”) is perceived by the scientific community worldwide, as evidenced by the institutional reports concerning this issue – early ones such as Magellan Report on Cloud Computing for Science [20] and the CERN Strategic Plan for a Scientific Cloud Computing Infrastructure for Europe [21] or recent ones such as the EU report on Open Science Cloud [22]. The goals set forth by such publications are being pursued by a number of research projects, addressing various aspects of cloud software integration.

Research on the support of scientific application on clouds includes the work on providing support for scientific workflow applications [23], or on-demand provisioning of virtual computing clusters on the basis of cloud resources [24]. An interesting cost and performance study discussing various obstacles and advantages of using cloud for HPC is presented in [25], where such issues as total turnaround time (which includes job waiting time) is compared for HPC and cloud infrastructures. However, most of the reported research focuses on typical large-scale workflows of MPI-based HPC applications, while our focus is on mid-range science.

On the web services front, the Taverna Workbench [26], a well-known workflow composition and deployment platform is being extended to provide the ability to interface with services deployed in computing clouds, managed by an external Remote Execution Server. While the actual deployment of services on cloud resources is out of scope of Taverna, the system enables such services to be orchestrated and managed in a coherent manner and therefore complements the work described in this paper. In fact, one of the requests from the VPH user community is to enable the emerging cloud platform to provide support for Taverna workflows.

eScience Central [27] is a platform for sharing computational services and data in a cloud environment, based on the Microsoft Azure platform. It provides mechanisms for deployment and enactment of application workflows. By using a browser, researchers can upload data, share it in a controlled way with colleagues, and analyse it using either a set of pre-defined services, or custom services which can be uploaded for execution and sharing. Unfortunately, custom workflow blocks can only be developed in Java, R or Octave and thus the platform is not generic enough to support easy integration of arbitrary legacy applications with the cloud.

Another example of PaaS solution which can be applied to both Web applications and batch workloads is Microsoft Azure Batch [28]. It allows submitting batch processing workloads to the Azure cloud, and provides a REST interface for job submission. However, in addition to being tightly couple to proprietary Windows-based technologies, no support for autoscaling or easy deployment of legacy applications is provided.

As far as deployment of application services on distributed resources is concerned, mention should be made of the prior work on RealityGrid [29] and UniGrids [30] projects, which attempt to establish guidelines for development of scientific (and other) applications on distributed infrastructures. UniGrids in particular proposes the notion of atomic services, akin to the application components envisioned in the VPH-Share project, although in this case focus is on Grid services, with specific types of APIs (accessible via the UNICORE platform), rather than on generic application components with no prior requirements related to implementation methods and user requirements.

It should also be noted that the wide variety of cloud computing stacks available on the market (OpenStack, Eucalyptus, OpenNebula and Nimbus to name just a few) complicates the development of cross-cloud and hybrid platforms. Thus, a number of projects and initiatives aim specifically at integrating resources from various providers into coherent computing infrastructures: hybrid clouds are based on connecting in-house (private) clouds with public clouds. This allows exercising tight control over critical data (processed in-house) while allowing some less confidential data to be processed in the public realm. Although hybrid clouds are not yet very popular [31], some solutions have specifically been developed to support them. OpenNebula and Nimbus provide special EC2 drivers which allow deploying images in the Amazon public cloud. This makes it possible to create hybrid clouds, bridging local installations and Amazon. On the service providers' side, there are such initiatives as the Amazon Virtual Private Cloud (Amazon VPC) [32] which can isolate sections of AWS resources and connect them with regular private clouds using VPN. If VPN is not sufficient Amazon offers a solution based on physical connection and 802.1q VLANs called AWS Direct Connect for secure and efficient connection to the VPC and other services. Datapipe GoGrid [33] also offers a hybrid solution, based on using the vendor-provided Dedicated Server as the "private" part and additional Cloud Instances as the "public" part of the infrastructure (when more resources are needed, or for less critical data processing).

As already mentioned, security is crucial for the applications deployed in the cloud. This include aspects of authentication, authorization, accounting (AAA) as well as data security (both in transit and at rest). IaaS cloud providers are responsible only for the bottommost layer, covering security of physical hardware and the hypervisors. Solution developers need to ensure higher-level AAA functions as well as classify and protect processed data. There are numerous identity management solutions such as OpenID [34] and authorizations frameworks like OAuth [35]. For data protection, encryption mechanisms based on strong AES may be used combined with a classification determining which kinds of data may be processed in private and/or public cloud. Furthermore, providers may offer some solutions to ensure better control over cryptographic materials such as Hardware Security Modules – such as in the case of AWS CloudHSM [36].

4. HYBRID CLOUD PLATFORM FOR VPH RESEARCH – CONCEPTUAL DESCRIPTION

The approach adopted in the design and implementation of the integrated computational platform for the VPH community is based on the authors' prior work on cloud resource management [37][38].

The platform fulfils a set of basic requirements formulated with regard to three separate user groups, as shown in Figure 1. We define a cloud application as a collection of services where each service occupies a virtual machine deployed somewhere in the cloud. Services are created by developers (who are familiar with software development tools but not necessarily familiar with cloud resource management) and used by end users (who are not IT experts). The entire platform is managed by a dedicated team of administrators (cloud experts) employed by IT centers which form part of the VPH community.

A layered overview of the platform which fulfils the above-mentioned requirements is provided in Figure 2. Below we explain the purpose and specifics of each layer shown in the figure.

A. Infrastructure layer: computing and storage resources

Given that the infrastructural resources available to the community consist of a number of disparate HPC sites (contributed by participating centres) as well as bought-in external resources procured from commercial operators, the platform must be able to interface various types of physical resources. In the context of the presented solution a private cloud site is understood as a collection of physical hardware managed by an institution which is a member of the VPH community, although this definition can be generalized to cover all deployments where the physical resources are directly supervised by a known stakeholder. This is in contrast to a public cloud, such as Amazon EC2 or RackSpace, where users do not control (and indeed aren't even aware of) the physical location where their computational jobs and data are stored and staged.

The division between public and private cloud sites is particularly important for research projects dealing with sensitive data – as is the case with the VPH applications – due to legal restrictions placed on the distribution and handling of such data. For instance, patient EHRs (Electronic Health Records) are usually not permitted to leave their originating institution (e.g. research hospitals) and must therefore be maintained on physical resources which comprise a local cloud site (an example of which is the Sheffield Teaching Hospital, itself part of the wider VPH community). The cloud platform must be aware of these restrictions and must respect them when deciding upon deployment of application services on physical resources. Satisfying this requirement calls for a metadata repository covering application services and medical datasets.

B. Middleware layer: cloud computing stacks

The middleware layer comprises low-level software which needs to be installed on the physical resources in order to set up a computing and/or data cloud. Two categories of software should be mentioned here: virtualization middleware and cloud computing stacks. The former category provides a layer of abstraction over physical resources upon which the computational cloud can be deployed. It manages the lifecycle of virtual machines but does not, by itself, constitute a federated infrastructure.

In contrast to virtualization middleware, a cloud computing stack can be defined as a collection of tools which enable the virtualized resources to be managed as a unit usually referred to as a cloud site. There are numerous cloud computing stacks available and while some cross-site compatibility is usually implemented (e.g. support for Amazon EC2 or, more recently, for OpenStack APIs), there are no industry-wide standards in this field.

It should also be noted that the VPH consortium operates a distributed data storage infrastructure which unifies access to file-based storage, and that in order to supply meaningful application services, the contents of this federation must be securely accessible to computing resources where data processing takes place. More information regarding the integration between data processing and storage is provided in section 6, while an overview of collaborative data management in the VPH-Share project can be found in [39].

C. Services layer: the Atmosphere cloud platform

The cloud service management platform known as Atmosphere is our primary original contribution in the field. It enables application services (further referred to as Atomic Services – following the introductory description presented above) to be implemented, managed, deployed on demand and accessed – either directly or indirectly, by workflow engines acting on behalf of the end user.

An Atomic Service is defined as an arbitrary application deployed on top of an operating system which is hosted in a virtual machine and can be preserved (and stored) by the cloud middleware services mentioned in the previous subsection. Thus, Atomic Services do not consume hardware resources (except the limited amount of disk space needed to store VM images) until their usage is requested by end users, at which point they can provide a template for Atomic Service Instances (ASI). This is consistent with the standard “pay per use” cloud model although not sufficient for applications which involve a tight request-response loop, where the overhead resulting from spawning and booting a VM proves detrimental to user QoE (Quality of Experience). Hence the platform provides the ability to tag selected Atomic Services as “static” which means that at least one instance of a service is kept online at all times, ready to serve user requests. For other types of services, Atmosphere acts as a factory: it instantiates services when requested by end user or by workflow development tools (including Taverna), and returns the instance endpoint to the requestor.

D. UI layer: user interface extensions for Atmosphere

All Atmosphere UIs assume the form of web applications and can be either served as standalone tools or integrated into portals. For the purposes of the VPH collaboration, Atmosphere management portlets are aggregated by the so-called Master Interface which provides an entry point to the system for all members of the VPH community.

5. IMPLEMENTATION AND DEPLOYMENT OF THE ATMOSPHERE CLOUD PLATFORM

In the course of the VPH-Share project, and with help from members of the Project consortium, we have developed and deployed a production-level cloud infrastructure comprising the above-mentioned Atmosphere cloud platform and overlaid by a set of end-user interfaces through which Atomic Services can be created, registered, shared, provisioned and accessed in a secure manner. Additionally, we carried out integration of Atmosphere with a number of heterogeneous cloud sited, both private and public. The outcome of this process is presented in Figure 3.

Figure 3 presents an overview of the architecture of the VPH-Share Cloud Platform and ancillary services. The platform itself was designed in such a way as to fulfill a number of requirements voiced by the VPH community both prior to and during the VPH-Share development lifecycle [2]:

- Integrate disparate computational resources and provide a consistent interface where the entire VPH infrastructure can be accessed in a uniform manner by users,
- Provide access to computational resources as well as an extensive data storage infrastructure. In addition, ensure that the data storage infrastructure is integrated with the computational resources in such a way that application services can directly and securely access/store data items, reading their input data directly from the infrastructure and outputting results back to the infrastructure. Results should be immediately available to end users who can either download them or visualize them using a set of inbuilt GUIs.
- Support deployment of a variety of applications in the VPH computational infrastructure: in particular, the system should enable development of new application services as well as deployment of existing applications (possibly developed in other projects or contributed by individual partners). While the VPH-Share platform is inherently web-based, the list of supported application types must not be limited to web applications and services.
- Provide single sign-on (SSO) security integration across the entire computational cloud stack. Ensure that application services and data items can only be accessed by properly authorized users, and that owners of application/data have control over who can access which elements of the computational infrastructure.
- In addition to integration with VPH client GUIs, the platform should also expose a set of APIs so that third-party tools (such as workflow management mechanisms) can make use of the platform features.

At the core of the cloud platform lies the Atmosphere Core Services Host that interprets requests issued via its API (these usually arrive by way of Master Interface extensions but can, in principle, be invoked by any external software authorized to do so, i.e. by passing a valid security token along with the request).

As already remarked, each VPH application (or component thereof) is registered in Atmosphere as a so-called Atomic Service [37], which can be spawned and provisioned on demand. Atmosphere itself hosts its own metadata model where it stores information on the available cloud hardware resources and Atomic Service images. It is capable of deploying, copying, saving and annotating Atomic Services as well as configuring access to Atomic Service Instances (ASIs) on cloud sites. The underlying computing resources are drawn from a hybrid resource pool, consisting of hardware contributed by Project partners as well as – where feasible – public machines leased from mainstream commercial cloud providers.

Among the facilities provisioned by the Atmosphere platform to its users are numerous services not found in either commercial or private cloud platforms, including:

- the ability to mount data storage resources as part of the local filesystem of virtual machines (facilitating deployment of applications which have been developed with local workstations in mind, without the need to reengineer their source code)
- mechanisms used to test API endpoints and signal potential service outages
- redirections to services deployed in private address spaces
- a sophisticated billing model, capable of tracking the consumption of financial resources by individual platform users and user teams, integrated with the platform-wide security framework and enabling service administrators to introduce service surcharges on top of what is charged by the proprietors of cloud sites themselves (thus facilitating commercial deployments of the presented solution)
- Components which enable migration of Atomic Services from one underlying cloud site to another (even across heterogeneous site types, such as OpenStack and Amazon Elastic Compute Cloud) – this facilitates smooth operation when e.g. a given site is being phased out, or an application must be transferred over to a public site for scale-out purposes. Such migration can be performed automatically and does not require expert knowledge on the part of the application

owner. Please note, that the migration is supported for Atomic Services, not their running instances, since live migration on running VMs is not currently supported by the underlying cloud middleware and providers.

The presented implementation carries a number of advantages both from the viewpoint of the service developer and the end user. First, it provides full support for the application development lifecycle: rather than having to manage the availability of each Atomic Service individually, the developer may trust Atmosphere to spawn the services whenever required and shut down instances which are no longer needed. Moreover, Atmosphere includes a uniform authentication and authorization model. Each Atomic Service template available to VPH-Share service developers comes with a preinstalled security proxy which intercepts incoming calls and validates them on the fly with the use of predefined security policies which can be injected into running instances. This means that services are only available to authorized users and that security is seamlessly integrated into the service development lifecycle. It also means that applications deployed as Atomic Services (i.e. the service “payload”) do not to be made security-aware.

Atmosphere itself provides a secure RESTful endpoint. All operations supported by the platform can be performed by invoking appropriate API actions. Online documentation is provided for developers. While Atmosphere can be interacted with programmatically by any tool capable of issuing REST requests, the VPH-Share project also provides a dedicated user interface layer (called the Master Interface) which sits atop Atmosphere and permits services to be located, shared and invoked with a web browser. Separate views are provided to service developers and end users, enabling each group to carry out actions described in Section 6. The Atmosphere visualization component can be embedded in arbitrary web portals for seamless integration with external infrastructures, such as those mentioned in Section 7.

An interesting issue which needed to be resolved was the lack of public IP addresses in private cloud deployments. Such cloud sites are typically composed of cluster nodes linked via a private network. Even when a pool of public IPs can be assigned to the cloud site by its managing institution, this pool cannot be expected to cover all the virtual machines which may be spawned by the cloud platform (note that a single physical host may easily run 10 or more separate VMs). For this reason, Atmosphere provides a custom reverse proxy service, to be installed alongside the cloud middleware on the site’s head node. This service is automatically configured by Atmosphere runtime whenever a fresh Atomic Service instance is spawned, and it provides port forwarding features for all end-user application components deployed on the given VM. External clients may interact with the Atomic Service instance by calling the reverse proxy on custom ports. If necessary, more than one instance of the reverse proxy may be spawned to avoid creating a communications bottleneck.

6. RESULTS

Over the four-year duration of the VPH-Share project we have succeeded in deploying a production-level hybrid computational cloud infrastructure; something which had not heretofore been achieved (earlier systems tended to focus on a single cloud infrastructure provider). In fact, by the end of VPH-Share the Atmosphere cloud platform was able to instantiate and deploy Atomic Services to a number of private and commercial clouds, including Amazon EC2, OpenStack, RackSpace, Google Compute and Microsoft Azure.

A. VPH cloud application lifecycle and usage examples

The lifecycle of VPH-Share application services, as designed and supported by the Atmosphere platform, is depicted in Figure 4 and Figure 5. All Atomic Services which have been created with the use of VPH-Share cloud tools follow the same procedure:

1. The service developer begins work by checking out a “blank” OS template and instantiating it on a cloud site of his/her choice. The instance can then be used to deploy and configure application components which constitute a single Atomic Service.

2. Once the developer is satisfied with the state of implementation of the service, he/she instructs Atmosphere to store it as an Atomic Service which can be further instantiated by end users (as well as by other developers). At this point the developer provides certain ancillary information – including a specification of resource requirements (number of CPUs/RAM/HDD space required by service instances), WSDL files (for SOAP services), RESTful endpoints (if available) and a textual description of the service. The developer is also free to flag the service as “shared” (meaning multiple users can share a single instance) or “scalable” (where multiple instances are used to serve requests coming from an individual user).
3. The new Atomic Service is officially published by the developer, whereupon it can be queried for in the VPH-Share user interfaces (using a variety of search tools).
4. An end user who wishes to make use of the Atomic Service issues a sharing request, which can be either approved or rejected by the service owner (i.e. the developer). If approved, the end user is free to instruct the cloud platform to spawn instances of the service and to interact with its features.

Some examples of Atomic Services managed by Atmosphere are listed below. Additional applications are being deployed (including tools coming from partner projects such as P-Medicine) and the platform itself is sufficiently generic to support most e-Science workflows and standalone tools, not necessarily tied to biomedical science.

- @neurist: This application models the process of aneurism formation in the human brain on the basis of CT images. It is used to predict the likelihood of aneurism rupture and suggest possible surgical treatment options. @neurist is based on a Taverna workflow which interfaces application services [40].
- VPHOP: An osteoporosis modeling tool capable of prediction of the mechanical competence and risk of fracture at a given skeletal location (e.g. femur, lumbar vertebrae) via analysis of tissue samples using mechanical models [41].
- EUHeart: This application analyzes the implications for altered coronary blood flow in cardiac ischemia. Using the tagged MRI and enhanced MR perfusion imaging, measures of ventricular mechanics and blood flow are estimated. Cardiac efficiency metrics can be extracted and linked back to patient prognosis [42].
- ViroLab Drug Ranking System: An application which enables clinicians to simulate the resistance of particular strains of HIV to selected drugs and suggest optimal treatment strategies for each patient. DRS is a web application which interfaces external services and data sources (e.g. virus strain susceptibility databases) to produce a user-friendly report. The application itself, as well as the computational services it relies on, is managed as a VPH-Share Atomic Service [43, 44].
- OpenLabyrinth: A web application used to conduct medical training courses at the Karolinska Institutet in Stockholm, Sweden. The application consists of a core database service along with a variable number of dynamic UI instances spawned on demand depending on the current load [45].

In total, over 250 Atomic Services were developed with the use of Atmosphere. This list comprises components of VPH projects such as @neurIST (2 tools, 14 services, 3 workflows), euHeart (2 tools, 4 services), ViroLab (2 services), VP-HOP (7 services, 2 workflows), p-Medicine (1 service, 1 workflow), DARE (4 tools, 4 workflows), CHIC, MD-Paedigree, ARTReat (2 tools), MySpine (1 tool), RT3s (1 tool), vFFR (3 tools, 2 workflows) and many others. In addition, we also integrated approximately 20 external applications, i.e. services which were not created with the use of VPH tools, but which were imported into the Share platform and can be provisioned in the same way as any other Atomic Service.

Figure 6 provides an overview of the usage statistics of the VPH-Share deployment of the Atmosphere platform (note that separate instances have been deployed for other research communities, as explained in Section 7). While the VPH-Share project itself concluded in mid-2015, the platform remains in active use and continues to provide computational resources for a variety of research projects, both internal and external to the VPH community.

B. *Common issues encountered while transferring existing applications to the cloud*

In the course of our development work in VPH-Share the Atmosphere team extended support and assistance to developers and proprietors of existing application services looking to deploy them in the cloud. Many of the extensions and specific tools mentioned in Section 6 were designed in response to specific deployment challenges. The following issues bear mentioning due to their recurring nature and their impact on the system architecture.

1) *Licensing issues*

In certain cases, the deployment of existing tools was hampered by the lack of appropriate software licenses or billing models. For example, several applications requiring the use of Microsoft Windows (such as the GIMIAS OsteoSuite environment employed in the MySpine workflow [46]) cannot be instantiated on private cloud sites operated by VPH partners due to the lack of Service Provider License Agreements (SPLA) which need to be negotiated with Microsoft. To overcome this problem, we opted to rely on public clouds, particularly Amazon EC2, where Windows instances can be procured on a pay-as-you-go basis (albeit at extra cost).

2) *Problems with timely instantiation of Atomic Services participating in complex workflows*

Due to the nature of cloud computing a service instance is not immediately available once an instantiation request has been dispatched – the cloud platform must first allocate hardware resources for the VM and boot up its operating system. Even at this point some application services (particularly ones meant to be accessed via an API) may not be functional – a service request, if sent too early, will trigger an error until the service container has properly started. This issue prompted us to implement a monitoring component which can return status information for selected service endpoints – the component periodically sends simple requests to a newly booted VM and notifies its clients of the status (up/down/unknown) of each endpoint.

3) *Transferring Atomic Services across cloud sites*

On occasion an Atomic Service developed in a private cloud environment had to be transferred to a public cloud – usually due to performance reasons or because the service required a certain type of hardware not provided by the private site. In order to avoid having to manually recreate the service in a new environment, we implemented a component which can be used (by platform administrators) to transfer service images across sites. The component supports cloud sites which provide an OpenStack or AWS-compliant management APIs and ensures automatic on-the-fly conversion of virtual machine images as necessary.

4) *GPU-intensive applications*

It is worth noting that cloud instances do not typically provide graphics acceleration. This is due to technical reasons – 3D accelerators are specialized and expensive pieces of hardware which consume a lot of power and generate a lot of heat – all very detrimental to the operation of a server farm. Given the unexpected reliance of certain VPH workflows on 3D visualization we've been compelled to provide efficient remote desktop access to customized service instances operating in a public cloud (EC2 G2 family).

5) *Local access to a distributed filesystem*

Most standalone applications adapted for use with the cloud rely on reading input/storing results in the local filesystem. Given the sensitive nature of data processed by the VPH community, such information cannot be saved along with each Atomic Service template, and the cost of adapting existing applications to interface with an external data service often proves prohibitive. As a solution, we came up with a way to automatically mount the VPH-Share data storage federation (called LOBCDER [47, 48]), in a secure manner, as part of the local system VM. LOBCDER exposes an interface which externally mimics a

WebDAV-compliant server, and can be accessed securely using the current user's security credentials, which are passed to the newly spawned VM instance through an appropriate cloud initial configuration mechanism. The service instance is thereby provided with access to its creator's federated data space, and can upload/download files by interacting with the local filesystem, obviating the need to reengineer the resident application. The decision to implement the service in this manner was dictated by user convenience rather than performance optimization; however no performance complaints related to the storage federation have been voiced by end users.

6) Performance implications of running applications in the cloud

Cloud platforms represent a specific type of computing environment which is uniquely suited to running scale-out studies and "long tail science" applications. While most standalone applications can be readily redeployed in the cloud with minimal reengineering, certain performance issues apply and must be communicated to users prior to cloud deployment. In the course of our work on the VPH-Share project we have found that the delay in instantiating virtual machines hosting application components impacts user experience – however, this should be weighed against the cost of operating "hot" VMs (which Atmosphere also supports) where an application is ready at the click of a button but the underlying VM must be kept up at all times. There are also issues related to processing 3D images using remote desktop access. Having specifically solicited feedback from users on this issue it was concluded that with modern remote UI protocols the inherent lag is manageable and does not result in reduced QoE. Finally, it should be noted that – given the ability to spawn an application on multiple cloud sites – some sites will offer inherently better performance due to their proximity to the sources of data manipulated by the Atomic Services. This can be communicated to end users in service metadata, and additionally service owners have the option to exclude instantiation of VMs on sites where performance (or cost of operation) is deemed unacceptable. Regarding resource utilization – a cloud monitoring interface is available for platform administrators and we assume that – with regard to hardware resources provided by commercial vendors – the principal limiting factor is cost rather than unavailability of physical hardware. While this is not currently implemented, a smart scheduler/optimizer could potentially take momentary load information into account when executed without clear instructions where to instantiate a given service. Atmosphere supports pluggable optimizers and the matter remains a potential avenue of future development.

7. DISCUSSION AND CONCLUSIONS

From the point of view of the features delivered by the VPH-Share cloud platform, we believe that the decision to create a hybrid IaaS offering – which is not specifically tailored to any particular application but is instead envisioned as an open infrastructure, capable of integrating additional types of software services as well as hardware infrastructures (cloud sites) – was justified and necessary. Over the course of the VPH-Share project we were faced with the need to deploy a vast array of dissimilar applications and services, including web applications, RESTful services, SOAP Web Services (with proper WSDL specifications), command-line tools, services exposing custom APIs on nonstandard ports, rich client applications with 3D GUIs (running under Linux as well as Windows), and more. That all of these application components can be successfully deployed, provisioned and utilized via the VPH cloud platform is testament to the platform's robustness and extendibility.

The robustness of the presented architecture is also evidenced by the fact that a separate set of hardware and software resources can be provisioned to the VPH-DARE project [50] with no changes in the design or implementation of Atmosphere itself. The platform is now used in production mode for scientific and educational purposes by a growing user community, consisting not only of the original members of VPH-Share and associated projects, but also by researchers affiliated with former VPH-Share project partners.

Our success in bridging the gap between large-scale computational services and the user communities which require such services to perform scientific research (whom we refer to as "domain scientists") has resulted in Atmosphere being deployed beyond the context of VPH-Share. In particular, the platform is now the core computational component of VPH-DARE, as well as of the Polish national HPC infrastructure – PL-Grid [51]. Work is also underway to deploy a separate instance of the cloud platform in support of the computational services being developed at the Insigneo Institute [52] – a collaborative initiative between the University of Sheffield and Sheffield Teaching Hospitals NHS Foundation Trust. Finally, Atmosphere is used to provision computational resources for the EurValve project [53] where certain computationally intensive – but sporadic – operations must be performed on the fly as part of clinical workflows.

Acknowlegements. The authors are very grateful to Prof. Rod Hose, Dr. Keith McCormack, and Dr. Steven Wood for fruitful discussions and suggestions. This research was funded by the VPH-Share and EurValve EU projects, and by the PL-Grid Core Polish project.

References

1. Mitchum R, Unwinding the 'Long Tail' of Science; editorial available at https://www.ci.uchicago.edu/blog/unwinding-long-tail-science (accessed 20 May 2016
2. The VPH-Share project website; http://www.vph-share.eu/ (accessed 20 May 2016)
3. Graphical Interface for Medical Image Analysis and Simulation, http://www.gimias.org (accessed 26 Apr 2017)
4. ANSYS project homepage, http://www.ansys.com (accessed 26 Apr 2017)
5. European Commission, Overview of the national laws on electronic health records in the EU Member States, https://ec.europa.eu/health/sites/health/files/ehealth/docs/laws_report_recommendations_en.pdf (accessed 26 Apr 2017)
6. A. Randles, E. W. Draeger, P. E. Bailey, Massively parallel simulations of hemodynamics in the primary large arteries of the human vasculature, Journal of Computational Science, Volume 9, July 2015, Pages 70-75, ISSN 1877-7503, http://dx.doi.org/10.1016/j.jocs.2015.04.003
7. R. M. Amorim, R. W. dos Santos, Solving the cardiac bidomain equations using graphics processing units, Journal of Computational Science, Volume 4, Issue 5, September 2013, Pages 370-376, ISSN 1877-7503, http://dx.doi.org/10.1016/j.jocs.2012.06.007
8. Chnafa, C. et al. Improved reduced-order modelling of cerebrovascular flow distribution by accounting for arterial bifurcation pressure drops, Journal of Biomechanics, Volume 51, 83 – 88
9. V. Díaz-Zuccarini, A. Narracott, G. Burriesci, et al. Adaptation and development of software simulation methodologies for cardiovascular engineering: present and future challenges from an end-user perspective. Philosophical transactions Series A, Mathematical, physical, and engineering sciences. 2009;367(1898):2655-2666. doi:10.1098/rsta.2009.0052.
10. A. Caiazzo, D. Evans, J-L. Falcone, J. Hegewald, E. Lorenz, B. Stahl, D. Wang, J. Bernsdorf, B. Chopard, J. Gunn, R. Hose, M. Krafczyk, P. Lawford, R. Smallwood, D. Walker, A. Hoekstra, A Complex Automata approach for in-stent restenosis: Two-dimensional multiscale modelling and simulations, Journal of Computational Science, Volume 2, Issue 1, March 2011, Pages 9-17, ISSN 1877-7503, http://dx.doi.org/10.1016/j.jocs.2010.09.002.
11. EGI Cloud Compute, available at https://www.egi.eu/services/cloud-compute/ (accessed 18 December 2016)
12. Helix Nebula project, available at http://www.helix-nebula.eu, (accessed 10 December 2016)
13. FutureGrid, https://portal.futuregrid.org/, 2012 (accessed 10 December 2016)
14. The Open Commons Consortium, http://occ-data.org/ (accessed 16 December 2016)
15. E. Afgan, J. Goecks, D. Baker, N. Coraor, A. Nekrutenko, and J. Taylor, J., "Galaxy: A gateway to tools in e-Science," in Guide to e-Science: Next Generation Scientific Research and Discovery (X. Yang, L. Wang, and W. Xie, eds.) pp. 145-177. Springer, New York.
16. D. Field, B. Tiwari, T. Booth, S. Houten, D. Swan, N. Bertrand, and M. Thurston, "Open software for biologists: from famine to feast," in Nat. Biotechnol. 24:801-803.
17. S. Zasada, T. Wang, A. Haidar, E. Liu, N. Graf, G. Clapworthy, S. Manos, P. Coveney, "IMENSE: An e-infrastructure environment for patient specific multiscale data integration, modelling and clinical treatment", Journal of Computational Science vol. 3, issue 5, pp. 314-327, Elsevier 2012; http://dx.doi.org/10.1016/j.jocs.2011.07.001
18. CloudMan, https://wiki.galaxyproject.org/CloudMan (accessed 30 December 2016)
19. CloudBioLinux, http://cloudbiolinux.org/ (accessed 30 December 2016)
20. K. Yelick, S. Coghlan, B. Draney, and R.S. Canon, "The Magellan Report on Cloud Computing for Science", U.S. Department of Energy Office of Science, Office of Advanced Scientific Computing Research (ASCR) December, 2011; available at http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Magellan_Final_Report.pdf (accessed 16 December 2016).
21. Strategic Plan for a Scientific Cloud Computing infrastructure for Europe, published by CERN/ESA, CERN-OPEN-2011-036; available at

http://indico.cern.ch/event/145537/attachments/137513/195173/European_Cloud_Computing_Strategic_Plan_Final1.0.pdf (accessed 16 December 2016)

22. Realising the European Open Science Cloud, High Level Expert Group report, European Commission, 2016, available at http://ec.europa.eu/research/openscience/pdf/realising_the_european_open_science_cloud_2016.pdf (accessed 16 December 2016)
23. G. Juve and E. Deelman, “Automating application deployment in infrastructure clouds,” 3rd IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2011), 2011; http://dx.doi.org/10.1109/CloudCom.2011.102
24. P. Marshall, H. Tufo, K. Keahey, D. LaBissoniere, and M. Woitaszek, “Architecting a large-scale elastic environment: recontextualization and adaptive cloud services for scientific computing”, Proceedings of the 7th International Conference on Software Paradigm Trends (ICSOFT); Rome, Italy; July 2012; http://www.nimbusproject.org/files/marshall-icsoft2012.pdf (accessed 16 December 2016).
25. A. Marathe, R. Harris, D. Lowenthal, B. de Supinski, B. Rountree, M. Schulz, and X. Yuan, “A comparative study of high-performance computing on the cloud,” in Proceedings of the 22nd international symposium on High-performance parallel and distributed computing SE – HPDC ’13, 2013, pp. 239–250.
26. T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, T. Carver, K. Glover et al., “Taverna: A tool for the composition and enactment of bioinformatics workflows,” Bioinformatics 20 (17): 3045–3054. DOI:10.1093/bioinformatics/bth361.
27. P. Watson, H. Hiden, S. Woodman, D. Leahy, and J. Cała. “e-Science Central: e-Science on the Web, powered by Clouds,” UK e-Science All Hands Meeting 2010.
28. Microsoft Azure Batch, available at https://azure.microsoft.com/en-us/services/batch/ (accessed 16 December 2016)
29. J. Cohen, A.S. McGough, J. Darlington, N. Furmento, G. Kong, and A. Mayer, "RealityGrid: An Integrated Approach to Middleware through ICENI". Philosophical Transactions: Mathematical, Physical and Engineering Sciences, 363(1833):1817--1827,(2005)
30. D. Fellows, J.M. Brooke; An Architecture for Distributed Grid Brokering, Proceedings of the 11th International Euro-Par Conference, Lisbon, Portugal, Eds. José C. Cunha, Pedro D. Medeiros, Springer LNCS 3648, p. 475, 2005
31. L. Shubert et al., The Future Of Cloud Computing – Opportunities For European Cloud Computing Beyond 2010, European Commission; 2011
32. Amazon Virtual Private Cloud (Amazon VPC), http://aws.amazon.com/vpc/ (accessed 16 December 2016)
33. GoGrid, https://www.datapipe.com/gogrid/, GoGrid, 2011 (accessed 20 December 2016)
34. OpenID, http://openid.net/
35. OAuth, https://oauth.net/
36. AWS CloudHSM, https://aws.amazon.com/cloudhsm/
37. Malawski M, Meizner J, Bubak M, Gepner P, Component Approach to Computational Applications on Clouds, *Procedia Computer Science*, 4:432-441, May 2011; DOI http://dx.doi.org/10.1016/j.procs.2011.04.045.
38. Malawski M, Gubała T, Bubak M: Component-based Approach for Programming and Running Scientific Applications on Grids and Clouds, *The International Journal of High Performance Computing Applications*, vol. 26, no. 3, pp. 275-295 (2012).
39. Benkner S., Bisbal J., Engelbrecht G, Hose R., Kaniovskyi Y., Koehler M., Pedrinaci C., Wood S., Towards Collaborative Data Management in the VPH-Share Project, Proc. 2011 International Conference on Parallel Processing, DOI: 10.1007/978-3-642-29737-3_7
40. S. Benkner, A. Arbona, G. Berti, A. Chiarini, R. Dunlop, G. Engelbrecht, A. F. Frangi, C. M. Friedrich, S. Hanser, P. Hasselmeyer, R. D. Hose, J. Iavindrasana, M. Kohler, L. L. Iacono, G. Lonsdale, R. Meyer, B. Moore, H. Rajasekaran, P. E. Summers, A. Wohrer, and S. Wood, “@neurIST: Infrastructure for Advanced Disease Management Through Integration of Heterogeneous Data, Computing, and Complex Processing Services,” IEEE Transactions on Information Technology in Biomedicine, vol. 14, no. 6, pp. 1365-1377, Nov. 2010.
41. M. Viceconti, Multiscale Modeling of the Skeletal System. Cambridge University Press, 2012.
42. N. Smith, A. de Vecchi, M. McCormick, et al. euHeart: personalized and integrated cardiac care using patient-specific cardiovascular modelling. Interface Focus. 2011;1(3):349-364. doi:10.1098/rsfs.2010.0048.
43. PMA Sloot, A Tirado-Ramos, I Altintas, M Bubak, C Boucher From molecule to man: Decision support in individualized e-health. IEEE Computer 39 (11), 40-46
44. A. Bankhead, E. Mancini, A.C. Sims, R.S. Baric, S. McWeeney, P.M.A. Sloot, "A simulation framework to investigate in vitro viral infection dynamics", Journal of Computational Science, Volume 4, Issue 3, May 2013, Pages 127-134, ISSN 1877-7503, http://dx.doi.org/10.1016/j.jocs.2011.08.007.

45. A. Kononowicz, A. Berman, N. Stathakarou, C. McGrath, T. Bartyński, P. Nowakowski, M. Malawski, N. Zary, "Virtual Patients in a Behavioral Medicine Massive Open Online Course (MOOC): A Case-Based Analysis of Technical Capacity and User Navigation Pathways", JMIR Medical Education Journal, vol. 1, No. 2, 2015; available at https://mededu.jmir.org/2015/2/e8/ (accessed 5 Jan 2017).
46. MySpine OsteoSuite, http://www.gimias.org/index.php/clinical-prototypes/osteosuite (accessed 6 Jan 2017).
47. S. Koulouzis, D. Vasyunin, R. Cushing, A. Belloum, M. Bubak, "Cloud Data Federation for Scientific Applications", EuroPar 2013: Parallel Processing Workshops, pp. 13-22, Springer 2013, ISBN 978-3-642-54419-4
48. S. Koulouzis, A. Belloum, M. Bubak, P. Lamata, D. Nolte, D. Vasyunin, C. de Laat, "Distributed Data Management Service for VPH Applications", IEEE Internet Computing 20(2): 34-41 (2016)
49. S. Koulouzis, A. S. Z. Belloum, M. Bubak, Z. Zhao, M. Zivkovic, C. T. A. M. de Laat, "SDN-aware federation of distributed data", Future Generation Comp. Syst. 56: 64-76 (2016)
50. The VPH-Dare@IT project website; http://www.vph-dare.eu/ (accessed 20 May 2016).
51. Bubak M, Kitowski J, Wiatr K (eds.), *eScience on Distributed Computing Infrastructure*, Springer 2014, ISBN 978-3-319-10894-0.
52. The Insigneo Institute for in-Silico Medicine; http://insigneo.org/ (accessed 20 May 2016).
53. EurValve: Personalized Decision Support for Heart Valve Disease; http://www.eurvalve.eu/ (accessed 30 December 2016).

Piotr Nowakowski holds an M.Sc. degree in Information Science from the AGH University of Science and Technology. Currently employed at the Academic Computing Centre CYFRONET AGH in Kraków, Poland. His interests include distributed computing environments with particular focus on cloud computing, as well as management of large data sets. Involved in numerous collaborative research projects on a national and European level as a designer and developer of customized IT solutions for e-Science. Author of several dozen publications on distributed computing and data management.

Marian Bubak has a M.Sc. degree in Technical Physics and Ph.D. in Computer Science. He is an adjunct at the Department of Computer Science and ACC Cyfronet AGH University of Science and Technology, Kraków, Poland, and a Professor of Distributed System Engineering at the University of Amsterdam. His research interests include parallel and distributed computing, distributed computing infrastructures, and e-science; he is the author of about 200 papers in this area, co-editor of 28 proceedings of international conferences and a member of 3 journals; more at http://dice.cyfronet.pl/.

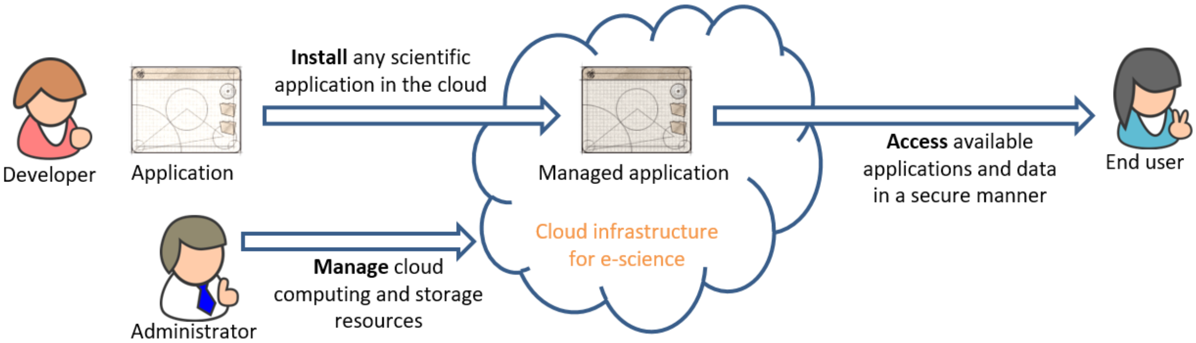


Figure 1: Conceptual view of cloud platform features – user classes and modes of interaction.

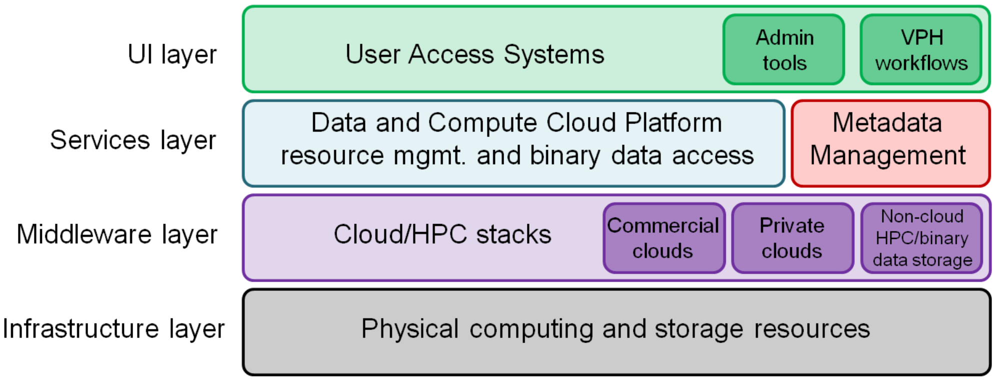


Figure 2: Layered view of the distributed VPH cloud computing environment.

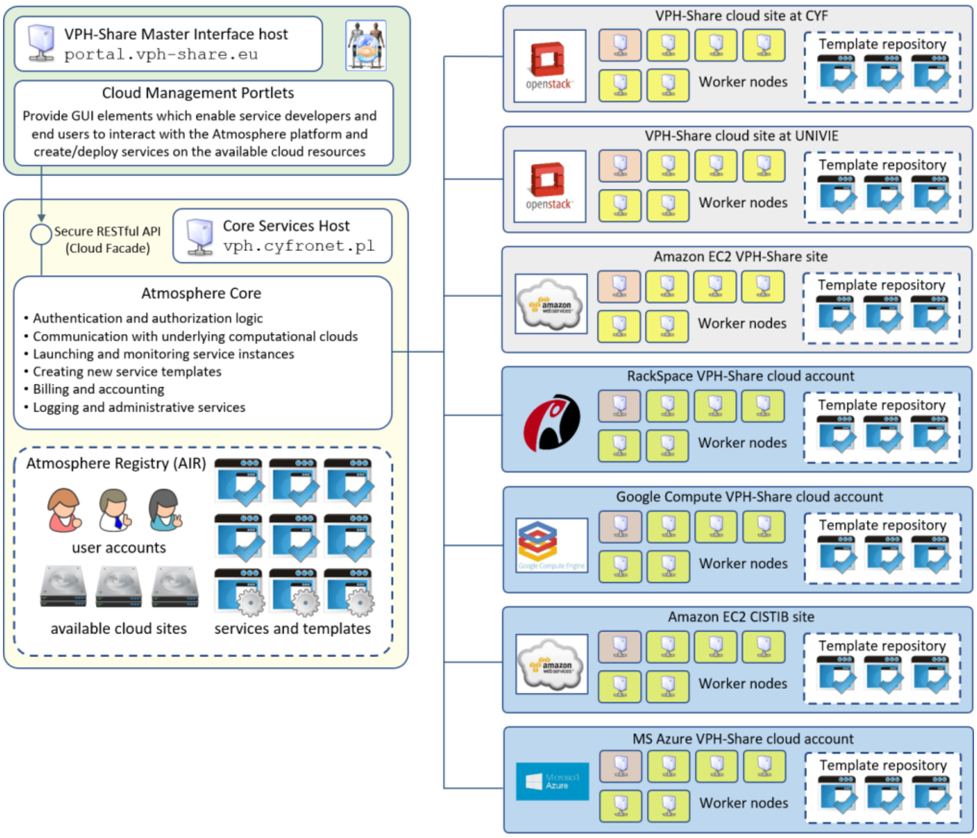
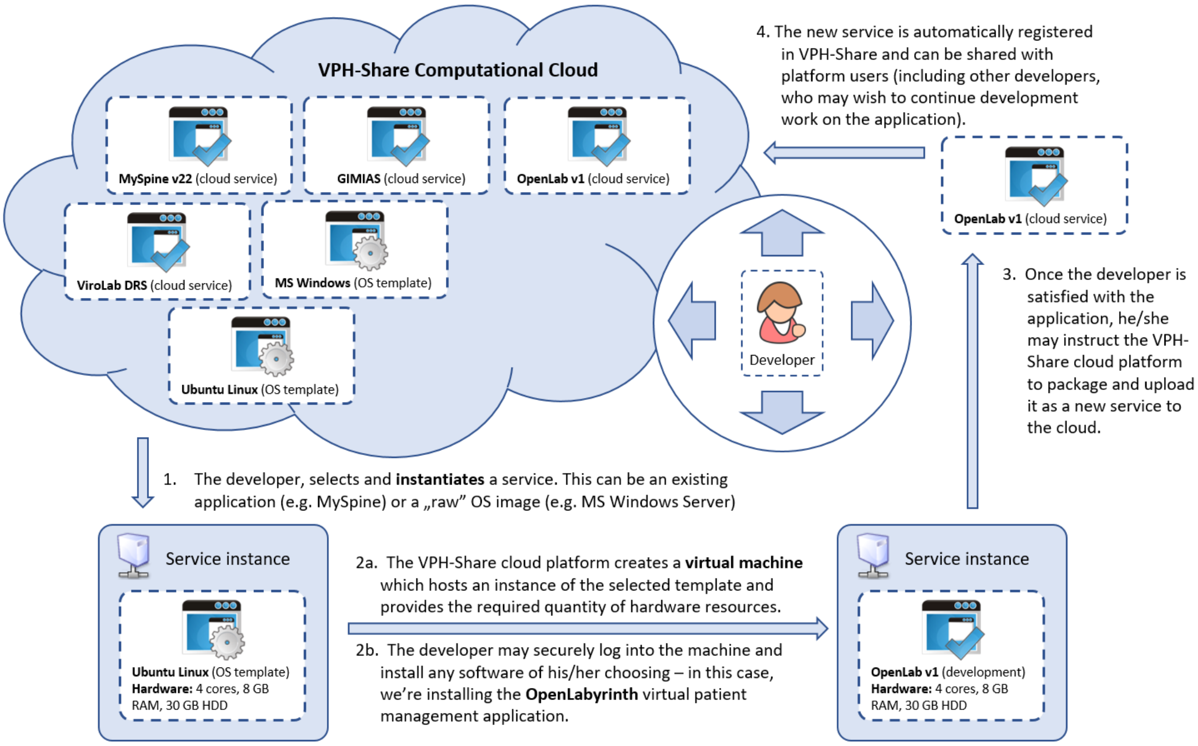


Figure 3: Hybrid cloud infrastructure as implemented for the VPH community. Dashed boxes represent storage of metadata and service templates.



1. The developer, selects and **instantiates** a service. This can be an existing application (e.g. MySpine) or a „raw" OS image (e.g. MS Windows Server)

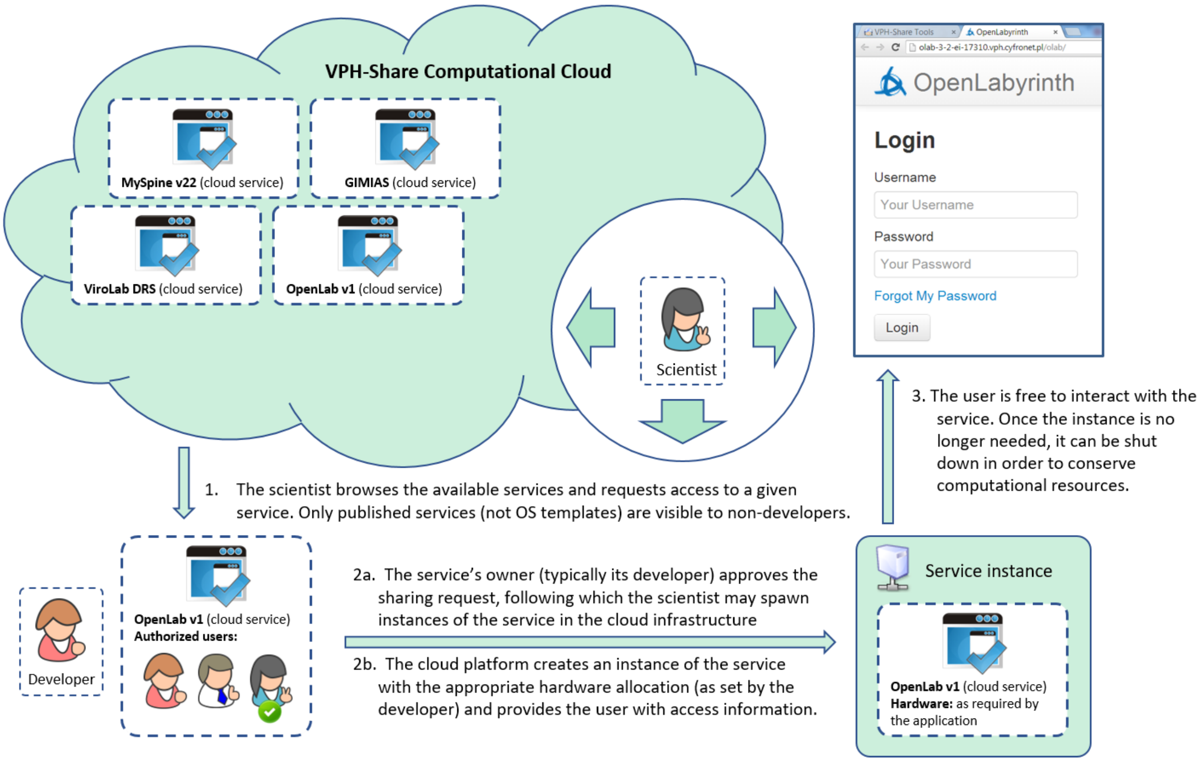
2a. The VPH-Share cloud platform creates a **virtual machine** which hosts an instance of the selected template and provides the required quantity of hardware resources.

2b. The developer may securely log into the machine and install any software of his/her choosing – in this case, we're installing the **OpenLabyrinth** virtual patient management application.

3. Once the developer is satisfied with the application, he/she may instruct the VPH-Share cloud platform to package and upload it as a new service to the cloud.

4. The new service is automatically registered in VPH-Share and can be shared with platform users (including other developers, who may wish to continue development work on the application).

Figure 4: Interacting with the VPH-Share cloud platform – developer's view.



1. The scientist browses the available services and requests access to a given service. Only published services (not OS templates) are visible to non-developers.

2a. The service's owner (typically its developer) approves the sharing request, following which the scientist may spawn instances of the service in the cloud infrastructure

2b. The cloud platform creates an instance of the service with the appropriate hardware allocation (as set by the developer) and provides the user with access information.

3. The user is free to interact with the service. Once the instance is no longer needed, it can be shut down in order to conserve computational resources.

Figure 5: Interacting with the VPH-Share cloud platform – end user's view.

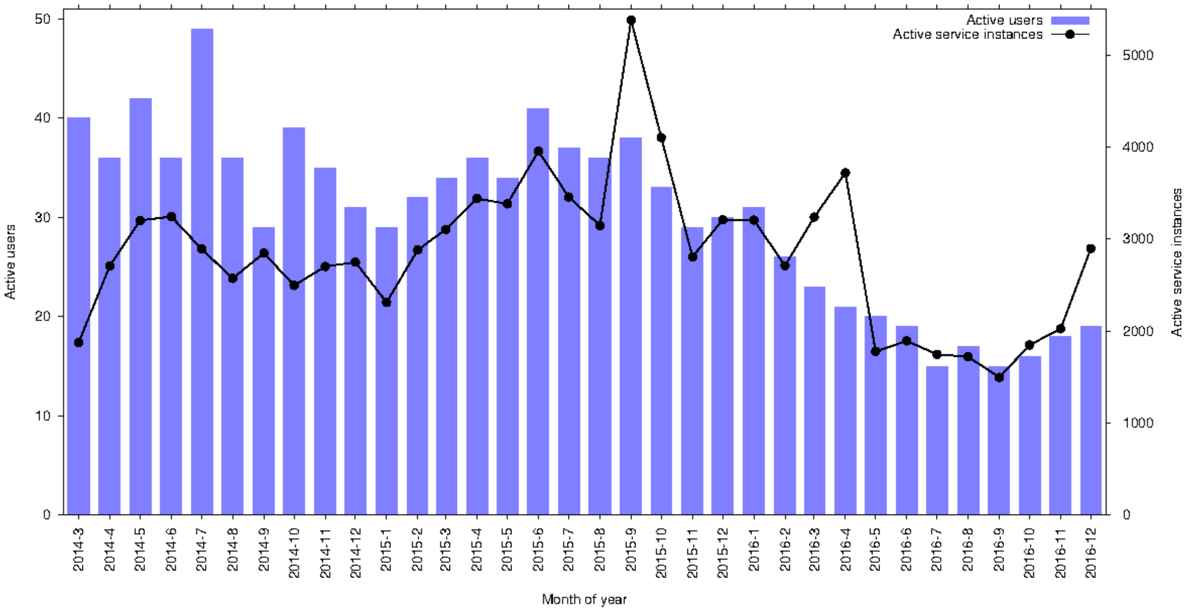


Figure 6: Monthly usage statistics for the VPH-Share cloud platform: active users and service instances.