A version of this paper was published in the Proceedings of SCREAM'15: The Science of Cyberinfrastructure: Research, Experience, Applications and Models, June 16, 2015, Portland, Oregon, USA.

Lessons from industry for science cyberinfrastructure: Simplicity, scale, and sustainability via SaaS/PaaS

Ian Foster
Computation Institute
Department of Computer Science
Mathematics and Computer Science Division
University of Chicago and Argonne National Laboratory
Chicago, IL 60637
foster@uchicago.edu

Abstract

Commercial information technology has changed dramatically over the past decade, with profound consequences for both software developers and software consumers. Software-as-a-service (SaaS) enables remote use of powerful capabilities, from accounting and payroll to weather alerts and transporation logistics, that used to require expensive in-house facilities and expertise. Platform-as-a-service (PaaS) offerings from cloud providers simplify the development and operation of SaaS software. These developments have slashed costs, reduced barriers to access and entry, and spurred innovation. Science cyberinfrastructure, in contrast, seems stuck in the 20th Century. I discuss lessons from industry that the scientific community might consider when designing cyberinfrastructure for the 21st Century.

1. The Cyberinfrastructure Challenge

Growing recognition of the transformative impact of ubiquitous digital data, computational modeling, and communications on scientific practice has spurred new *cyberinfrastructure* programs in the US and elsewhere. These programs aim to enable "broad and open access to leadership computing; data and information resources; online instruments and observatories; and visualization and collaboration services"—and to create broad-based "distributed knowledge communities that collaborate and communicate across disciplines, distances and cultures" [1].

After close to a decade of cyberinfrastructure programs, we can point to many successes that have benefited thousands. Yet we remain far from realizing what must surely be the ultimate cyberinfrastructure goal: enabling the entire scientific community to make effective use of new computational methods and tools.

Achieving this grand goal requires that we overcome challenges of *complexity* and *cost*. Much scientific software is too complex for general use, requiring specialized expertise and/or facilities to install, use, and maintain. Furthermore, the scientific community has not successfully assembled the resources needed to sustain this often costly software, especially with the degree of support needed for broad adoption. We must find new software development, delivery, and funding methods that can simultaneously achieve **simplicity** and **sustainability**. As we explain in this brief paper, industry experience with SaaS and PaaS suggest that **scale** can be the secret to bridging these goals.

2. The Emergence of SaaS and PaaS

Commercial information technology (IT) also used to suffer from excessive complexity and cost, in much the same way as science cyberinfrastructure. But over the past decade, commercial IT has undergone the most consequential transformation since the invention of the electronic computer: namely, the emergence of cloud-hosted software as a service (SaaS) for large-scale outsourcing, and platform as a service (PaaS) for large-scale automation. The result has been a remarkable reduction in both complexity and per-user costs.

A SaaS provider maintains and operates a single copy of their software, configured so that many remote users can use it concurrently. Users run the software over the Internet from their Web browser. Intuitive Web interfaces and no local software to install mean that the barriers to the use of software delivered by SaaS can be far lower than for conventional software.

The impact of SaaS on society has been profound. Consumers and companies alike increasingly hand off time-consuming and tedious activities (e.g., organizing photos, booking travel, ordering products, managing payroll) to SaaS providers, who perform them reliably, efficiently, and cost effectively. Because the software is easy to use and per-use costs are low, usage increases. Network effects further encourage adoption.

The impact on the software industry is also significant. There is no need to support multiple platforms or versions: instead, every user runs up-to-date software. The software developer has full visibility into how software is used, simplifying problem determination and facilitating learning. On the other hand, software developers must implement remote access interfaces that scale to support many users, and SaaS providers must operate services in a way that achieves high availability. Fortunately, these latter tasks are simplified by PaaS systems from Amazon Web Services (AWS), Google, Salesforce, and others, which automate many aspects of SaaS operations: in effect, they provide SaaS for SaaS. For example, AWS provides services for deploying applications, replicating state, scaling capacity, queuing requests, and content delivery, to mention just a few.

SaaS is more than just software accessed via a Web browser: SaaS offerings typically also support programmatic access via REST APIs. Thus, for example, Google Docs provides both a Web interface for users who want to edit documents interactively and also a REST API that applications can use to operate on those same documents. Programmers can use such APIs to combine functionality from multiple SaaS providers. For example, a programmer at a manufacturing company might combine customer data (from Salesforce, a SaaS customer relationship provider), stock data (from Unleashed, a SaaS online inventory management provider), and weather data, and generate a report (in a Google spreadsheet) showing how sales and inventory vary with weather. This example illustrates how a componentized view of

software allows SaaS to implement service-oriented architecture (SOA) without the complexity that has bedeviled SOA in the past.

SaaS and PaaS have also driven major changes in the economics of software distribution, based on the interplay between simplicity, scale, and sustainability:

Simplicity: Intuitive Web 2.0 interfaces, designs that focus on essentials, and outsourced operations make software delivered via SaaS easier to use than conventional software, and thus reduce barriers to entry for new users.

Scale: Low barriers to entry, when combined with attractive features, spurs demand. Multi-tenant, cloud-based deployments and self-help interfaces enable economies of scale in SaaS providers, as the incremental cost per additional user is low.

Sustainability: Economies of scale allow providers of SaaS software to achieve sustainable revenue with low per-use or subscription charges, and/or freemium models in which basic services are free and advanced features involve modest subscriptions. Low charges in turn spur increased adoption, leading to a virtuous cycle in which simplicity and low prices feed large-scale use, which in turn permits lower prices.

Similar issues apply in the case of platforms, although here the positive returns to scale are yet greater due to network effects. (Indeed, platforms only succeed when broadly adopted.)

SaaS and PaaS have implications not only for software development and business models, but also for policy. As Weyl and White observe [2]: "The primary policy problem in platform markets is usually considered to be excessive lock-in to a potentially inefficient dominant platform. … Instead the greater market failure is excessive fragmentation and insufficient participation. These problems, in turn, call for a very different policy response: aiding winners in taking all, ensuring they and not their copycats profit from success, subsidizing adoption and regulating the resulting 'One' dominant firm."

3. Lessons for Scientific Cyberinfrastructure

I argue that the scientific community must apply SaaS/PaaS methods to the delivery of science IT if it is to overcome challenges of complexity and cost. Today, SaaS is rarely used in science, and with few exceptions (e.g., Globus, iPlant, kBase), simple, scalable, and sustainable science-focused platforms are lacking. This situation leads to fragmentation due to replication of function, which in turn hinders interoperability and prevents network effects. To overcome these problems, the scientific community and science funders must pursue substantial changes to software delivery, architecture, funding, and related areas.

Delivery: Most scientific software today is downloaded, installed, and run locally. This degree of complexity introduces a significant barrier to entry: in many fields, installing, learning, and maintaining state-of-the-art data analysis or simulation software can be a full-time job. SaaS can overcome these barriers to entry and thus enable much broader use. Scientific communities need to determine which software can be delivered via SaaS and build capacity in developing and operating SaaS.

Architecture: Much scientific software is organized as complex vertically integrated packages. Where PaaS is appropriate (e.g., in collaborative environments and data management), developers can reduce costs and increase interoperability by outsourcing to platform services. Investigators need to use existing platform services and consider what new platform services could benefit their community.

Sustainability: Much scientific software is sustained by dedicated individuals or small teams, supported by a patchwork of grants. This approach offers no positive returns to scale: if usage doubles, support costs increase but resources do not. SaaS/PaaS subscription models can overcome this problem, but need policies that encourage SaaS and PaaS adoption at a sustainable scale. Funders should consider policies that support the establishment and sustenance of platforms. For example, awards could encourage SaaS/PaaS, reward use of PaaS, and encourage researchers to budget SaaS and PaaS usage charges in grants.

Norms: Research institutions are not typically accustomed to the pay-per-use or consumption-based models that are typical of SaaS/PaaS. Also, funders tend not to be cognizant of this approach, which makes it difficult for investigators to incorporate SaaS/PaaS into their grants. Funders and institutions should promote new policies that overcome these challenges.

Open source: The community should also consider whether open source requirements are appropriate in all settings. Initially established with the laudable goal of spurring collaboration, open source requirements can also facilitate copycatting, which can prevent the large user bases that are essential to sustainability.

These and other changes to practice and policy need to be discussed and debated broadly, to build consensus for action.

4. The Globus example

I present an example to demonstrate that unique features of scientific software need not preclude the use of SaaS/PaaS. Globus (globus.org) applies SaaS methods to deliver powerful research data management services: transfer, sharing, publication, discovery. Its Web 2.0 interfaces enable intuitive use by non-experts without software installation; users proclaim that the service is wonderful, that it "just works." A cloud-hosted, multi-tenant, and self-service implementation provides for scalability, with more than 25,000 registered users and more than 90PB and 10B files transferred. As a step towards sustainability, Globus implements a fremium subscription model: transfer services are free but certain advanced features and management capabilities are accessible only to subscribing institutions. This has attracted more than 30 subscriptions as of May 2015.

Globus also functions as a platform. Its REST APIs allow other applications and services to outsource research data management, identity and group management, and other functions to Globus. For example, the NCAR Research Data Archive outsources data transfer and sharing functions to Globus.

5. Summary

With effort and vision, we can ensure that a broad spectrum of software is accessible in this way, greatly simplified due to outsourcing of functionality and sustained by a broad community of subscribers. We will thus promote values we hold dear, such as accessibility, reproducibility, and more money for research.

Acknowledgments

Supported in part by DOE DE-AC02-06CH11357, NSF OCI 10-53575 and NIH 1U54EB020406-01. I thank Dan Katz, Brigitte Raumann, Steve Tuecke, and Vas Vasiliadis for their comments.

References

1. NSF Cyberinfrastructure Vision for 21st Century Discovery, http://www.nsf.gov/pubs/2007/nsf0728/, 2007.
2. Weyl, E.G. and White, A. Let the Right "One" Win: Policy Lessons from the New Economics of Platforms. *Competition Policy International*, 10(2):29-51, 2014.