

# Factors in Development and Adoption of New Cloud Software and Standards

AS CLOUD SERVICES INCREASINGLY BECOME PART OF HOW WE DO BUSINESS AND SOLVE PROBLEMS IN COMPUTING, IT'S BECOMING IMPORTANT TO EVALUATE RIGOROUSLY IN REAL TIME THE LEVEL OF SUCCESS OF DIFFERENT APPROACHES. This column explores the process of selecting cloud software and

standards in modern deployment environments and calls out some important characteristics specific to cloud computing.

## Innovation and (or Versus) Uncertainty

Let's start by looking at several factors related to general software and standards adoption. As Everett Rogers pointed out in his multi-decade study of factors limiting diffusion of innovations,

"Getting a new idea adopted, even when it has obvious advantages, is difficult. Many innovations require a lengthy period of many years from the time when they become available to the time when they are widely adopted. Therefore, a common problem for many individuals and organizations is how to speed up the rate of diffusion of an innovation."<sup>1</sup>

According to Rogers, an innovation is fundamentally an idea, practice, or object that presents an individual or an organization with new or alternative approaches to solving problems, but also generally carries with it uncertainty as to whether the new idea is superior to previous approaches. The response of many people to this uncertainty is to seek further information about the new idea to cope with the uncertainty it creates. In this classical view, communication of information about an idea among participants is the primary limiting factor governing its uptake and adoption.


In cloud-based development and deployment, the availability of information about new concepts is no longer the limiting factor in building a user base. Multiple avenues exist to disseminate, publish, and receive information, including working code that can be adopted with essentially no delay. Instead, we've reached a state in which the capability of innovations to reach widespread adoption is limited primarily by competition for attention by other new ideas of a similar nature. We are thus in a situation of managing an abundance of information, rather than one in which simple diffusion of information limits uptake.

Adoption of new ideas in cloud computing must now take place by creating a self-sustaining ecosystem in which improvements to an original idea are

ALAN SILL

Texas Tech University,  
alan.sill@standards-now.org





incorporated seamlessly into implementations of the concept at the same time as interactions with other ideas in the same conceptual space are being explored. The ecosystem of interest to any given enterprise is one of true end-to-end solutions for their problems, and isn't limited to any one particular product or software stack; deployment of ideas in this space requires all components of the solution to work and to be tested at once.

I call this process “deployment-based evaluation” and have explored in a previous column using the combined development and operations (DevOps) approach to push new ideas as rapidly as possible into use for evaluation and testing. Cloud computing is particularly well-suited to this type of approach. This situation applies equally well to both software and standards.

The environment in which innovations occur and take hold in cloud computing has changed fundamentally from the one-dimensional situation explored by Rogers. No longer is change simply held back by the process of learning about new advancements or communication about their advantages, or even by proponents of change being able to reach critical mass by interacting with early-adopting users.

Cloud computing is now intrinsically a multi-dimensional environment involving the interaction and simultaneous use of multiple competing ideas, often from completely different portions of the organization. In this way, it differs from previous arenas for standards development, such as CORBA or SOAP-based Web services, in which solutions from a single family of closely related products was often sufficient, and they could be evaluated in isolation.

Unlike the simple situation of selecting electrical voltages or frequencies or even protocols for an interface, in cloud computing it's now possible, and even necessary, to evaluate multiple API and usage alternatives simultaneously and to make rapid performance-driven choices in switching from one approach to another. Innovation in cloud computing isn't usually limited simply by raw uncertainty as to whether an approach is available or will work, or even by the presence of multiple alternatives that must be evaluated to arrive at a conclusion. Instead, tools exist to evaluate and compare multiple available approaches rapidly.

The range of successful cloud computing solutions is now sufficiently broad that a solution to a given problem can be assembled from a wide range of available methods. Selecting an optimal approach requires that we take advantage of the distributed, multicomponent nature of cloud computing itself to evaluate and compare solutions, and often to combine alternatives in ways that go beyond previously available individual methods.

Adoption of a new idea is therefore limited in cloud computing not only by fitness for use or a superior design, but also by the ability to adapt to new uses and new situations. A successful deployment-based evaluation approach consists of implementing and testing new ideas and evaluating their success in real time compared to other methods, when equally valid alternatives exist that are also being explored, using data where possible to assess the results.

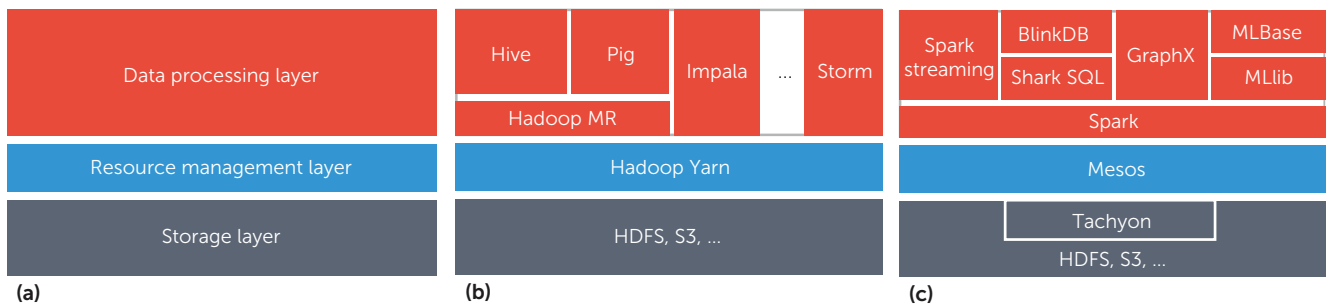
### Software Products: What's in a Name, and Where Does It Fit?

Names are, of course, an important tool in recognizing new ideas. According to an unfortunately apocryphal quote attributed to pioneering computer scientist Donald Knuth, “The most important thing in the programming language is the name. A language will not succeed without a good name. I have recently invented a very good name and now I am looking for a suitable language.”

It's possible that Professor Knuth actually said this, but I cannot substantiate it. The best source I can find is at <http://www.cs.virginia.edu/~evans/cs655/manifests/0227.html> with the notation that this was made in 1967 in reference to the then-impending failure of Algol 68.

Whether or not Knuth actually said this, the art of naming software has now become an important factor for success, a rampant source of confusion, and an organizing component for coordinating new efforts. As an example, let's look at the components of the Hadoop and Berkeley Data Analytics Stack (BDAS) software ecosystems (Figure 1).

Clearly the name itself is not really as important as the environmental niche or region in which a given solution fits in comparison to the problem space of solutions to which it applies. As the diagrams in Figure 1 show, different software components can fit into the same space of solutions as other prod-



**FIGURE 1.** Software data processing ecosystems: (a) layer functionality, (b) Hadoop stack, and (c) Berkeley Data Analytics Stack (BDAS). (Adapted from a talk by Ion Stoica at the NSFCloud Workshop on Experimental Support for Cloud Computing.<sup>2</sup>)

ucts, and the best approach is to understand the general nature of the problem that each component solves.

Even though these software products share some background in terms of development, already there are multiple choices for software components that occupy the same place in the ecosystem in terms of functionality. Naming these efforts allows development to con-

gies and to define related items of terminology, and are best applied to allow interchangeable components to be assembled as part of an overall system.

In some cases, such as the choice of physical voltage on an electrical interface, or frequency of radio transmission, the interaction between systems is immediate and direct. Arbitrary but definite choices must be made in these

tributed systems. Adoption rates typically start out slowly in such situations, even for very good ideas, growing at a slow rate until enough people are in the user pool that the average user can communicate with many more people than with other preceding methods. As Rogers observes,

“An interactive innovation is of little use to an adopting individual unless other individuals with whom the adopter wishes to communicate also adopt. Thus, a critical mass of individuals must adopt an interactive communication technology before it has much utility for the average individual in the system.”<sup>1</sup>

Standards have their greatest value when used to specify interfaces between technologies and to define related items of terminology.

centrate around a certain point in the ecosystem with increased clarity for all concerned.

## When Are Standards Needed?

The purpose of standards isn't necessarily to bring order to the chaos of software products and their associated user bases. Instead, as I've pointed out previously in this column, standards have their greatest value when used to specify interfaces between technolo-

situations, often resolved by involving a government regulatory agency. These situations tend to be regional. Different choices can be, and often are, made in different regions of the globe when the choice in one region only affects the suppliers and users in that region.

Interactivity creates especially important considerations for technologies involving communication. Cloud computing fits in this category, because it depends on technologies that span dis-

Cloud computing follows this paradigm, but differs slightly in that a technology can be successful even if it benefits only a small number of highly distributed users. The key for success is that the approach be workable for each set of users and sustainable in an environment of rapidly changing deployments of other related methods. Technologies that don't build in sufficient ability to adapt to other methods will eventually fail, even if they have at some point a large user base, because of the requirement to function in a physi-

cally distributed, rapidly changing, interdependent environment.

To evaluate the potential for the success and future growth of a given cloud technology, we need to weigh its current location on the adoption growth curve, its capacity to interoperate with other related methods, and its position in the technology spectrum as either a product or an interface between products. We also need to evaluate whether a workable solution limited to a particular environment or specific use is enough, or whether we might instead need portability and interoperability on a wider scale of deployment involving different cloud platforms or providers. The dynamics of cloud computing allow each of these choices to be valid.

These conditions also apply equally to evaluating, testing, and adopting cloud software and standards. When evaluating standards specifically, however, keep in mind their fundamental role in the software ecosystem, which is to permit such evaluation, interoperation, and change to take place. A standard will work well when it promotes and encourages software reuse and interchangeability so that developers and users can make their own selections as to the appropriate software product to use.

**IN PRIOR ISSUES, I'VE CALLED ATTENTION TO A NUMBER OF SPECIFIC STANDARDS FROM DIFFERENT ORGANIZATIONS THAT ARE MAKING THEIR WAY ALONG THE ADOPTION GROWTH CURVE.**

I've also pointed out the need for strong, consistent, and continuous communication between developers adopting these technologies and the standards organizations themselves, in the process observing that traditional docu-

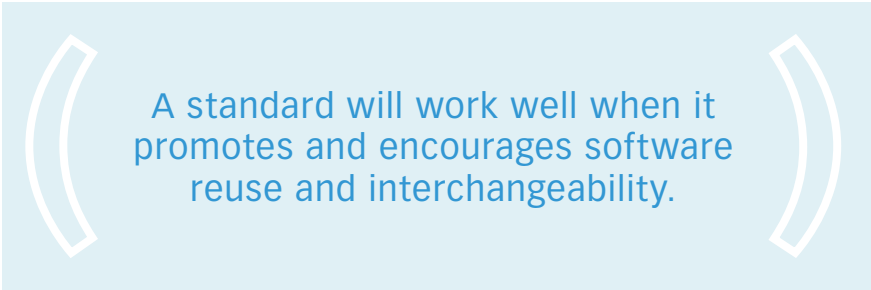
ment development approaches used by isolated standards organizations won't work well in cloud environments because of the highly interconnected nature of cloud development and deployment.

I'm pleased to report that much progress is being made in this area. Future versions of this column will look at individual standards in terms of conceptual functions they can be used to perform, such as image portability, job

able at <https://www.chameleoncloud.org/nsf-cloud-workshop>.

---

**ALAN SILL** directs the US National Science Foundation Center for Cloud and Autonomic Computing site at Texas Tech University, where he is also a senior scientist at the High Performance Computing Center and adjunct professor of physics. Sill holds a PhD in particle physics from American University and is an active



A standard will work well when it promotes and encourages software reuse and interchangeability.

provisioning, and task orchestration, and bring you up to date on their current status. Meanwhile, the discussion here should help illustrate how standards are useful to characterize, identify and connect components of software development.

As always, please respond with your opinions on this or previous columns, and include any news you think the community should know. I can be reached for this purpose at [alan.sill@standards-now.org](mailto:alan.sill@standards-now.org).

#### References

1. E.M. Rogers, *Diffusion of Innovations*, 5th ed., Simon and Schuster, 2003.
2. I. Stoica, "Berkeley Data Analytics Stack (BDAS): Experience and Lessons Learned," talk at the NSFCloud Workshop on Experimental Support for Cloud Computing, 2014; avail-

member of IEEE, the Distributed Management Task Force, TeleManagement Forum, and other cloud standards working groups, and serves either directly or as liaison for the Open Grid Forum on several national and international standards roadmap committees. He serves as vice president of standards for the Open Grid Forum and co-chairs the US National Institute of Standards and Technology's "Standards Acceleration to Jumpstart Adoption of Cloud Computing" working group, and is one of the organizers of the ongoing developer-oriented Cloud Interoperability Plugfest series of meetings. For further details, visit <http://cac.ttu.edu> or contact him at [alan.sill@standards-now.org](mailto:alan.sill@standards-now.org).



Selected CS articles and columns are also available for free at <http://ComputingNow.computer.org>.