Emerging Standards and Organizational Patterns in Cloud Computing

IN THIS COLUMN, I PROVIDE SOME EXAMPLES OF DIFFERENT VALID STYLES OF CLOUD STANDARDS DEVELOPMENT. I chose these to illustrate current topics that have led, or are leading, to new, useful methods. I hope these examples will shed light on the differences between standards development styles in terms of content and of the communities that come together to create and publish them.

ALAN SILL

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

One of many difficulties in talking about standards is that there are so many paradigms for their use. This very positive characteristic plays out in the old saw, and to me not very funny joke, that "the greatest thing about standards is that there are so many to choose from." This statement misses one of the most important characteristics of standards: they emerge naturally, predictably, and reliably from intrinsically complex conditions.

Multiple technological options are bound to grow out of conditions in which a diversity of approaches exists. Standards tend to emerge in situations in which complexity can be reduced by pursuing a common method. The fact that standards tend to arise out of complex situations is expected, and the fact that different standards arise to deal with different aspects of these situations is also expected.

Applying DevOps to Standards

Rapid technological change is presently the most important factor limiting effective creation and adoption of cloud standards. I discussed this situation in detail and presented some methods to identify conditions under which standards can be expected to emerge in the most recent column in this series.[1]

As summarized in my earlier column, the number, diversity, and complexity of the solutions to a given problem govern the potential for standards to emerge. Once the technology choices become sufficiently complex and diverse, the opportunity grows for standards to provide practical approaches to common problems, but only if those standards can simplify or clarify the overall set of available solutions.

The term "DevOps" became popular late last decade to denote the combination of development and operations portions of systems administration resulting from applying agile project management methods. Patrick Debois, one of the principal initiators of this movement, recently observed that DevOps has helped support teams to be less risk averse, changing attitudes from avoidance of change to situations in which administrators respond by saying,

2325-6095/15/$31.00 © 2015 IEEE

"Let's do it more often if it is hard." In Debois's words,

> "This change comes partly from the fact that, in modern infrastructures, we're dealing less with hardware and more with an API or programmable infrastructure. So admins have evolved to thinking more programmatically about infrastructure, which is a natural fit with the current development process."[2]

To deal with the rapid pace of change in cloud computing, I've argued for some time that we need a DevOps approach to standards formulation and testing, just like the one that's popular for combining development and operations in software deployment. By this I mean we should pursue simultaneous creation and testing of standards and examples of the software needed to put them into use, with rapid mutual feedback between these two areas of invention.

This approach is a much better match to cloud software conditions than the traditional, highly decoupled, sequential, slow-feedback methods that have dominated standards up to now. Such an approach doesn't mean that standards documents should be written without any foresight or plan for long-term permanence, but any less ambitious approach will fall short of the need to keep up with modern rapidly changing and evolving cloud technologies.

There is good news to report here. DevOps for standards is starting to take hold and even to show signs of becoming the preferred method for new standards development in the world of cloud computing.

Cloud-Native Services, Applications, and Containers

A recent example of the "DevOps for standards" trend is provided by the emergence of the Cloud Native Computing Foundation (CNCF, https://cncf.io), sponsored by a large collection of companies and open source projects. The goals of this effort are to facilitate collaboration on technologies for deploying cloud native applications and services, and to work closely with the also recently formed Open Container Initiative (OCI, www.opencontainers.org), which focuses on container-based virtualization.

Rapid technological change is presently the most important factor limiting effective creation and adoption of cloud standards.

These two projects are examples of efforts to create specifications and standards that consolidate progress in new areas of cloud computing. Each results from earlier code-first software development projects that realized the need to gather contributing groups together to develop formal specifications. Neither intends to abandon its code-first approach, but instead plans to augment this approach by formalizing agreements so others can understand and depend on them.

Consensus-driven combined standards and software development efforts such as these are the natural course of evolution mentioned in previous "Standards Now" columns. The conditions under which previous developments led to the desire to create common standards are precisely those that I discussed in my last column[1] and summarized at the beginning of this column. They are the expected response of any field to the inevitable need to bring into harmony multiple approaches resulting from earlier stages of development.

The CNCF and OCI projects also illustrate a new paradigm in which the right to contribute to the technical steering of a common standard is partly determined by previous successful demonstration of relevant key technologies through actual code.

SDO Work Patterns

Three main types of organizations develop standards:

- industry-based special-purpose consortia composed mostly of large-scale companies and sometimes organized around the foundation model;
- formal standards developing organizations (SDOs) that exist primarily to develop standards; and
- informal community organizations that can be open to single-person developer or user input.

Of course, any particular organization can have characteristics in more than one of these categories.

There are also many organizational patterns that govern participation and contribution. Some SDOs clearly aim to serve as vehicles for discussion among projects belonging entirely to their respective funding sources. Others limit decision making to representatives from contributing sponsors and/or have restrictions in place for accepting external unsolicited input. Some operate as entirely closed organizations, whereas others are open to public input and participation.

Regardless of their participation model, all organizations that offer central services have operating costs, and thus need sources of funding. Among existing SDOs, there is a wide variety of methods for raising the needed funds. Most have tiered membership levels, with various privileges depending on the level chosen. The higher tiers are often aimed at companies or governments, although some SDOs offer discounted or free memberships to nonprofit or academic organizations. Some also offer individual memberships, and a few SDOs require no membership payment at all for participation.

To produce and publish their standards under appropriate copyright statements, all SDOs require some form of agreement among their members regarding contributions. In today's patent-driven competitive landscape, such provisions are essential to protect the opportunity for fair use of specifications resulting from common development efforts. As code-first examples become increasingly important, the rules for contributions and licensing of technologies and for establishing consensus will have to adapt to keep pace with the need for rapid cycling between specifications and implementations.

Organizational Output and Products

SDOs sometimes develop branded or otherwise closely interlinked sets of specifications that they use not only for organizational uniqueness, but also to distinguish their output from that of other organizations. One risk of this type of approach is that the resulting documents might be too strongly unique to that SDO to allow them to be reused by other organizations or efforts. Although it can be easy to create a new group within an SDO to explore a new standard, and even to foster implementations, this process might be closed off or obscure to nonmembers.

A standard's overall purpose is usually to identify a common approach that can be reused either by attribution or by reference.

The same risk of ingrown isolation exists, of course, in large software projects. Without explicit efforts to minimize this problem, each such piece of software can tend to be a closed system with little to no provisions for external projects to connect with and interoperate among other software being used in the same problem space. In each case, both with standards and software, a mechanism for establishing consensus when multiple parties are involved must exist.

A standard's overall purpose is usually to identify a common approach that can be reused either by attribution or by reference to save time and effort when developing implementations. In the case of software, efforts to make it possible to call or connect communication channels between different modules make interoperability possible in principle.

When formulated in clear, publicly documented terms, such efforts lead naturally to the creation of custom-designed APIs, protocols, and standards. Examples of SDO products that are seeing uptake in cloud computing include the Open Cloud Computing Interface (OCCI), Topology and Orchestration Specification for Cloud Applications (TOSCA), Cloud Data Management Interface (CDMI), Open Virtualization Format (OVF), Cloud Application Management for Applications (CAMP), and Cloud Infrastructure Management Interface (CIMI) standards that I've explored in previous columns.

A "profile" is a specific interpretation of a more general specification used to adapt a given standard to a particular use case. Each such profile is a customized way to use a published standard in a particular area that resolves ambiguities or options left by the original specification. Profiles are thus a valuable tool that can be created either by a specification's original producer or by other parties or organizations, referencing the original publication. The European CloudWATCH project has gathered and analyzed examples of profiles (see www.cloudwatchhub.eu/cloudwatch-clustering-standards-profiles).

Although I can't catalog all cloud computing activity from all of these organizations in a single column, we make an effort in these pages to keep track of recent standards activity of note to the community at large. In addition, several organizations exist in both government and industry sectors that periodically survey progress in the cloud standards area. Among these are the US National Institute of Standards and Technology (NIST, www.nist.gov/itl/cloud), the CloudWATCH project, the Global Inter-Cloud Technology Forum (www.gictf.jp), and IEEE's own p2301 and p2302 cloud-oriented working groups and their associated Intercloud Testbed project (http://standards.ieee.org/news/2011/cloud.html).

Individual Contributions

Individual contributions are also often put forward as standards or proposals for the first stage of standardization. This is the fundamental basis of the request for comment (RFC) process that is the starting point for many important standards, even in cases where a resulting process is carried forward in larger working groups or otherwise formalized in broader settings involving multiple participants.

Individuals operating outside the realm of foundations or SDOs can publish their work under their choice of license. To encourage reuse, many such efforts are offered under open license protocols such as creative commons (CC) or CC by attribution (CC-BY). Some use license agreements from open source projects, such as the Apache license, even when code isn't involved. In contrast, SDOs generally require input to be given according to predetermined contributor agreements with more restrictive terms.

The virtIO development effort provides a good recent example of a standard that has transitioned from a single-person specification to full SDO-based published status based on further work. The SDO in this case is the Organization for the Advancement of Structured Information Standards (OASIS, www.oasis-open.org).

The virtIO specification grew out of the desire to simplify and standardize methods to handle device drivers in virtualized environments. Rusty Russell of IBM, working through Australia's "Oz Labs" open source community, first documented the approach in an informal specification that became the basis of a published paper.[4] The specification was later refined in a series of public repositories, the latest of which (https://github.com/rustyrussell/virtio-spec) was used as the home of the document until mid-2013.

OASIS chartered a technical committee to look at formalizing paravirtualized block, network, console, and other drivers through refinement of the virtIO specification in June 2013, with the stated goals to make virtual devices "look like driver authors expect physical devices to look" and to "keep the good, discard the bad, and make the ugly optional" (www.oasis-open.org/committees/virtio/charter.php). This committee has produced a version 1.0 specification and two updates since then.

The OASIS technical committee released its most recent version of virtIO 1.0 in August 2015, and has further draft changes in the pipeline for future revisions.[5] This standard has grown significantly in importance since its introduction and now forms a standardized basis for handling device drivers in virtualized environments. Comments on new versions are possible through a public feedback license during designated periods, but contributions to the specification during its development require OASIS membership.

The Role of Foundations

Special-purpose foundations are increasingly becoming the basis for combined standards and software development projects, rather than efforts through any of the previously existing SDOs or by independent community efforts. For example, the Linux Foundation is managing both the CNCF and OCI efforts.[3]

The initial members of these two projects are primarily large companies active in cloud computing, along with a selection of open source projects that are also primarily driven by large companies. Large-scale vendors, and not existing SDOs or community-based efforts, are therefore positioned to drive the development of specifications and standards in these areas.

The choice of the foundation model for these new efforts probably results from the previous success of foundations as the basis for several well-known vendor-neutral open source software projects, such as Apache, OpenStack, and the OpenFlow open networking project. It's also possible that the problems that motivated standards to be created in these areas were viewed by their respective communities as detached from the interests of previously existing standards organizations.

Such dedicated formulations, however, also lead to considerations about ownership, guidance, and influence over the directions of the resulting projects. For example, Thierry Carrez, current director of engineering for the OpenStack Foundation, recently posted an exploration of the role of foundations as being the preferred organizational model for large open source projects. In this discussion, he included cautions to "pause and read the fine print" and to "assess how open they are, discover who ends up owning their upstream open source project, and determine their primary reason for existing" (http://ttx.re/the-age-of-foundations.html).

The fact that foundations are playing a role in standards development is neither surprising nor new. The central role of the Linux Foundation in these efforts may, however, be a bit of a surprise. The growth of container use in cloud computing has largely been built around Linux control groups and the related "lxc" code base, which might explain some of the interest of the Linux community, but the goals of the OCI effort, for example, explicitly include non-Linux uses of container technologies.

Linux is also obviously not the only operating system suitable for cloud-native applications. Other forms of Unix are equally viable, and Microsoft has developed a strong interest in standardization efforts for containers for use with its Windows and Azure technologies. Perhaps the Linux Foundation's experience in supporting previous multiple independent implementations of its basic core technologies played a role in this selection.

The characteristics detailed above are neither exclusive to each category of cloud computing organization nor exhaustive. Special-purpose projects supported by foundations, for example, generally accept and incorporate contributions from individual participants even though they are often funded and driven by companies. All open source software projects operate in this way. Many SDOs do so as well, with conditions that depend on their membership requirements. Government or international standards organizations often have more formal rules, sometimes restricting input to their processes to certain types of contributors. It's a mix-and-match environment fueled by rapid technological progress.

The resulting wide variety in terms of rules for participation among different types of organizations, however, is at least as frustrating to the average developer as any other aspect of standards development, and is frequently a barrier to participating in such organizations.

THE EXAMPLES I'VE PRESENTED SHOW THAT THE CLOUD COMPUTING FIELD IS INCREASINGLY ABLE TO BALANCE ITS INTRINSIC NEEDS FOR STABLE DESIGN PATTERNS AND RAPID CHANGE. Practical, workable cloud computing standards are starting to emerge in some areas, with actively growing user communities. Earlier columns catalogued several other related efforts. In the future, I hope to solicit deeper descriptions of each of them to form the basis for a special issue.

Please share your opinions on this topic or on those explored in previous columns. What is the right balance between cloud standards and cloud software development processes? How should choices in this area be presented and pursued, and who should have the opportunity to participate in each stage? When multiple options are available, who chooses among them, and how can the community endorse such decisions? Should there be room for individual or community-based efforts in addition to foundations or formal SDO efforts, and if so, who should organize them, and what is the best mechanism for their discovery and adoption by commercial providers? What about considerations for use and development by scientific organizations, or the nonprofit sector?

Let us know what you think, and please also include any news you think the community should know about the general areas of cloud standards, compliance, or related topics. We're always open to article submissions. I'm also happy to review ideas for such submissions, or for proposed guest columns. I can be reached for this purpose at alan.sill@standards-now.org.

References

1. A. Sill, "Socioeconomics of Cloud Standards," *IEEE Cloud Computing*, vol. 2, no. 3, pp. 8–11.
2. S. Chin, "How DevOps Became DevOps," Java Magazine, July/Aug. 2015, pp. 18–21; www.oraclejavamagazine-digital.com/javamagazine/july_august_2015/?pg=19.
3. "New Cloud Native Computing Foundation to Drive Alignment among Container Technologies," press release, Linux Foundation, 20 July 2015; www.linuxfoundation.org/news-media/announcements/2015/07/new-cloud-native-computing-foundation-drive-alignment-among.
4. R. Russell, "Virtio: Towards a De-Facto Standard for Virtual I/O Devices," *ACM SIGOPS Operating Systems Rev.*, vol. 42, no. 5, 2008, pp. 95–103; doi:10.1145/1400097.1400108.
5. R. Russell et al., eds., Virtual I/O Device (VIRTIO) Version 1.0, OASIS Committee Specification 03, 2 Aug. 2015; http://docs.oasis-open.org/virtio/virtio/v1.0/virtio-v1.0.html.

ALAN SILL *directs the US National Science Foundation Center for Cloud and Autonomic Computing site at Texas Tech University, where he's also a senior scientist at the High Performance Computing Center and adjunct professor of physics. Sill has a PhD in particle physics from American University and is an active member of IEEE, the Distributed Management Task Force, TeleManagement Forum, and the Open Grid Forum, for which he currently serves as President. He is a member of several other cloud standards working groups, and national and international standards roadmap committees. For further details, visit http://cac.ttu.edu or contact him at alan.sill@standards-now.org.*