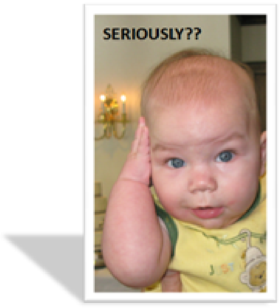
API Jabberwocky: You Say Tomay-to and I Say Potah-to

Lori MacVittie, 2011-25-04

IT as a Service requires commoditization. Commoditization implies standardization. The network needs standardization, and that's only going to happen via a common API and semantic model.



Randy Bias of Cloudscaling apparently set off a firestorm at Cloud Connect 2011, stating with typical Randy forthrightness: "API's don't matter."

It's not something we haven't heard before. In fact, it's not something I haven't said myself, in a way. Randy wasn't really questioning the need for APIs, that's a given. What he was getting at was to question the need for *standardization* of APIs.

Within IT, particularly in development organizations, the API has become been the primary method of integration. Applications needing to invoke the functionality of another application or system leverage API calls to perform some task: add this data, retrieve this data, process this data. When SOA was at its peak, the focus was properly on the abstraction of business functions at the API level, to provide for reuse and consistency of process across applications.

The focus of "cloud" APIs and infrastructure APIs has been on interoperability. While a standardized API is indeed one way to achieve interoperability, i.e. the ability to migrate operational processes across dissimilar environments, there are reasons why such a goal may be considered impossible to achieve. Consider Mike Fratto's recent commentary on APIs and cloud computing , noting that "feature variation between vendors" is too broad to expect agreement on which functions make up a common base from which such an interoperable infrastructure API could be designed.

> Of course APIs matter, but having a functional standardized cloud service management API doesn't and, I'd argue, never will.
>
> …
>
> The semantics of calling a method on vendor A's API or Vendor B's API will be different because of how the API's are implemented, but there is likely enough feature variation between vendors that even agreeing on what functions make up the most common denominator is probably impossible to settle.
>
> -- Mike Fratto, "Standardizing Cloud APIs is Useless", Network Computing (March 2011)

Mike is right; variances across infrastructure solutions in terms of features and functions is broad and makes a common API representing those features and functions a daunting if not impossible task. That said, I would argue that achieving API parity – seen as necessary for interoperability - is not necessarily the only goal of infrastructure APIs today. Let us consider that the goal of infrastructure and cloud APIs is to **present a common interface for the implementation of** *operational functions*.

SERVICE-ORIENTED OPERATIONS

The management of infrastructure components today – in most cases – can be accomplished via a standards-based (i.e. protocol interoperable) service-enabled SDK. These SDKs provide standards-based access to to just about every method necessary to manage the infrastructure.

But these are not "APIs" in the way in which we need them to be APIs; they are not service-oriented themselves, but merely leverage service-oriented protocols as a means to exchange the data necessary to be managed.

What is needed at the infrastructure level is service-oriented operations; the abstraction of operational functions into an API that can be implemented commonly across clouds, environments and infrastructure. Commonality across such operational tasks *do* exist*.* Despite the very broad differences in application network infrastructure (application delivery controllers, load balancers, WAN optimization controllers) there are still common *operational tasks* that can certainly be abstracted and specified by a common API. The differences are in the features and specific configuration, not necessarily the tasks. Configuring a Virtual Server (or Virtual IP Address, VIP) is an operational task that requires the same core functions across multiple vendor implementations. What we have under the hood is semantic differences that exacerbate existing feature disparity: you say tomato, I say potah-to.

Therein lies the opportunity for standardization – at the operational task layer of the IT stack.

DOES it MATTER?

The question was does a **standardized** API matter or not? Assuming we can get to operational function parity across the infrastructure, does it change anything? The differences in features that could be invoked via an API make true seamless interoperability a likely unattainable goal.

Mike goes on to say that if we cannot achieve such a goal, standards are unnecessary: "*The goal of IT standards is for products and technologies to foster interoperation. If standards can't result in interoperation, then the standard is useless*". Mike's view of the application of standards and their benefits is limited to interoperability. I disagree that interoperability is the only benefit provided by standards and that this is part of the transformation of IT – and networking in particular – that must happen ere IT as a Service can come to fruition. The ultimate goal is to provide a portfolio of infrastructure services with the implementation being not irrelevant, but loosely coupled; separated from its interface in such a way that components – or even providers - can be switched out without disruption.

Mike's argument is based upon the premise that there is only one purpose for IT standards: interoperability. Interoperability isn't the *only* raison d'être for APIs and standardization. That may have been true in the past, but that is no longer true. Standardization also, as we are so often reminded by cloud pundits, enables commoditization, which is a key step toward reducing the cost of resources in the data center. APIs and standardization also matter to devops, to the operational developer who needs to automate and integrate infrastructure by codifying operational processes. They need APIs as a means to scale operations along with servers and applications and users. Standardized APIs (or at least a standardized data center infrastructure model) would significantly reduce the time required for integration simply by reducing the number of models and interfaces devops needs to learn. It also subsequently reduces the possibility of introducing repeatable errors into such operations by allowing devops to focus on the process, not specific products or their APIs. While SDKs are available, they often require intimate knowledge of the solution as well as the standards and protocols by which such SDKs are used. The difference in semantics alone is enough to put off even the most stalwart operations developer. APIs mitigate these differences, offering ease-of-use, as it were, and a simpler model of interaction that is necessary for devops to not only automate, but automate **efficiently**. The difference between 5-8 SDK function calls, over the network, and a single API call are huge in terms of performance, reliability and the need for idempotency in infrastructure operations. It's not just an end-user thing, it's an operational risk mitigation thing as well.

I would argue at this point that yes, APIs and a *common operational model (i.e. standardized)* are necessary to the future growth of cloud computing and highly dynamic data center architectures. The commoditization of operational processes necessary to achieve economy of scale requires standardization. Without such common ground the ability to automate operational processes is negatively impacted and reduces the chance and benefits afforded by moving to such a model in the first place.

SOA ultimately failed to achieve on its goals not because it was a poor idea but because implementations failed to recognize that the service interface was about common *business* functions, not application functions. Cloud and infrastructure APIs and integration need to focus on a standardized (or at least de facto standardized) service interface that encapsulates common *operational* functions. APIs – and standardization of the models -do matter at the infrastructure layer of a data center architecture in the sense that they provide the foundation upon which operations can be automated and ultimately integrated in to the data center orchestration engines that will emerge in future iterations to replace many of the home-grown scripts and solutions leveraged today out of necessity.

APIs do matter, but not solely for purposes of interoperability. Rather the goal is to enable the scalability of operations through automation. Standardization of common operational components (functions and the core model) would go along way toward enabling that scalability to occur more efficiently.

- The World Doesn't Care About APIs
- Standardized Cloud APIs? Yes.
- Standardizing Cloud APIs is Useless
- Cloud, Standards, and Pants
- Infrastructure 2.0: Squishy Name for a Squishy Concept
- Despite Good Intentions PaaS Interoperability Still Only Skin Deep
- Cloud interoperability must dig deeper than the virtualization layer
- The API Is the New CLI
- WILS: Automation versus Orchestration
- Magic Virtualization-Fairy Dust and the New Network

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc. Corporate Headquarters info@f5.com	F5 Networks Asia-Pacific apacinfo@f5.com	F5 Networks Ltd. Europe/Middle-East/Africa emeainfo@f5.com	F5 Networks Japan K.K. f5j-info@f5.com