Best practices to develop SLAs for cloud computing

Develop a standard way to create service level agreements that multiple partners can use

Judith M. Myerson

January 07, 2013

You can't rush the process of developing service level agreements (SLAs) between cloud consumers and providers. What service guarantees do consumers expect? What terms and conditions can cloud computing providers and consumers agree on? What terminologies will they use? Plus, the cloud provider must evaluate its relationships and SLAs with vendors, enterprise data centers, network providers, and content providers. There's much to consider. In this article, the author discusses some best practices and how SLAs can be standardized.

TM Forum defines SLAs as expectations among two or more parties regarding service quality, priorities, and responsibilities. The Cloud Standards Customer Council views cloud SLAs as written expectations for service between cloud consumers and providers. It provides guidance to decision makers on what to expect and what to be aware of as they evaluate and compare end user SLAs from cloud computing providers. The decision makers should also evaluate the SLAs that a cloud computing provider has with vendors, enterprise data centers, network providers, and content providers.

An SLA is not a mandate when it's driven by a major reorganization, downsizing, service consolidation, or transition to a cloud services environment. It doesn't have the inputs from all pertinent parties that must be involved.

An SLA is not a one-way solution. One party — the cloud service provider, for example — should not impose decisions about how things should be done, particularly when the other party — the cloud service customer, for example — has different expectations about how the SLA should be formulated.

An SLA isn't a quick solution. Rushing the process of negotiating the terms and conditions in the SLA doesn't give enough time for the parties to understand each other's expectations particularly when each party has a different perception of what a certain terminology stands for.

A real-life story on SLA

Several years ago, I published a seven-part article series (no longer available) on developerWorks about using SLAs in a web services context. Afterward, the chief of web services and services oriented architecture for a major United States federal agency contacted me. He particularly liked the part about mitigating risk for vulnerability with an SLA guarantee. This part described interruption thresholds that can spell the difference between saving or not saving an enterprise from a complete denial of service (DoS) and destruction of critical system assets. It also explained how large interruption thresholds can affect SLA guarantees.

While corresponding with the agency chief, I mentioned another part in that series about guaranteeing a web service with an SLA. That part described six types to test web services features before making a service public:

- **Statefulness.** Does the server respond in the correct steps?
- **Access.** Can an unauthorized user successfully access a control that only administrators are authorized to use?
- **Response time.** Is the application taking too long to respond?
- **Time-out.** What happens when the application times out?
- **Versioning.** Can a new build break an existing application's functions?

The part also described the following exceptions that can be potentially included in the SLA:

- Failure: Hardware, telecommunications, software, or performance monitor
- Network issues not within direct control of the service provider
- Denial of service: Client negligence or willful conduct; acts of God; war strikes; unavailability of telecommunications; inability to get supplies or equipment needed for the provisions of the SLA
- Scheduled maintenance: Hardware and software upgrades and backups

The agency chief asked me to do a presentation and hold a systems engineering discussion with his team. In my presentation, I included the points mentioned above. Plus, I added the importance of an exit clause in the SLA so that customers can exit from the SLA when the guarantees can't be met at numerous times.

After my presentation was reviewed, the chief sent me a copy of his agency's SLA template for review and asked me to contribute to the first and later versions of the template. For privacy reasons, I cannot discuss the details of the template, but it was eventually made part of the federal system.

This SLA template can be extended to cloud-based services. All SLAs are about service guarantees, regardless of the type of service that is provided: web services, cloud services, or network access services. Penalties are imposed if service guarantees are not met. The level of guaranteed services differs from one partner to another.

Not all web services (such as service-oriented architecture (SOA) cloud-based applications) are cloud-based. SOA is a design pattern that is composed of loosely coupled, discoverable, reusable, interoperable, platform-agnostic services by using web services standards. SaaS is not SOA. SaaS is a consumption model; it uses resources that are hosted by a cloud service provider. SOA is a design model in which there is no restriction on who the consumer is.

The importance of standardizing SLAs in a multi-cloud environment

While SLAs have traditionally been a contract between a service provider and a cloud service customer (an enterprise, business, or government agency), the expanding value chain for other services has made SLAs important for a myriad of often complex relationships between partnerships.

For example, the same service provider can provide services to:

- Cloud service customers (SaaS end users, PaaS developers and IaaS infrastructure specialists)
- Vendors
- Large enterprises
- Businesses
- Government agencies

The same vendor provides services to:

- Network providers
- Cloud service providers
- Web service providers
- Enterprises
- Businesses
- Government agencies

The same network provider provides network access services to cloud service providers.

The same enterprise provides private cloud services to SaaS end users, PaaS developers, and IaaS infrastructure specialists.

As part of pre-disaster planning, a requesting provider needs to discover if reserved resources would be available from a substitute provider. It can be a many-to-many relationship between requesting and substitute providers. A requesting provider can have more than one substitute provider. A substitute provider can associate with one or more requesting providers.

The cloud service provider is the substitute cloud service provider who accepts the request for data transfer services from a requesting cloud service provider so that the requesting provider can fix the problem with outage or load balancing. The substitute provider can limit the number of data transfer requests from more than one requesting service providers.

To compete successfully, companies must proactively manage the quality of their services. Since provisioning of those services is dependent on multiple partners, management of SLAs become critical for success. More important is the standardization of terminologies in the SLAs among the partners.

When terminologies are not standardized, the definition of a terminology that a partner (a requesting service provider) uses may differ from the definition that another partner (a substitute service provider or vendor) uses. Partners, for instance, may define SLA parameters (such as failure to meet performance) in different ways.

When the differences aren't ironed out during a negotiation, they might impact the SLA process (for example, what is a penalty and when to impose the penalty). One partner might impose the penalty of a 10 percent rebate of service fees for cloud service downtime exceeding one hour while another partner might impose the penalty of 12 percent for downtime exceeding half an hour.

SLA best practices

Best practices are used to describe the process of developing and following a standard way of doing things that multiple partners can use. When the partners agree on the standard way of using terminologies when negotiating SLAs, they contribute to the process of SLA standardization.

The Cloud Standards Custom Council provides cloud consumers with the seven steps they should take when evaluating cloud SLAs to help them know what to expect when comparing cloud service providers or negotiating terms with a provider, as detailed in *Practical Guide to Cloud Service Level Agreements*, published by the Cloud Standards Customer Council in April 2012 (see Resources.

1. Identify the cloud actors

The first step is to identify the cloud actors. The National Institute of Standards and Technology (NIST) Reference Architecture (see Resources) identifies five unique cloud actors:

- Cloud consumer
- Cloud provider
- Cloud carrier
- Cloud broker
- Cloud auditor

Each actor has a unique role and responsibilities. Only the first three have relationships with one another regarding the terms and conditions in the SLAs. One drawback to note is that the NIST Cloud Computing Reference Architecture doesn't identify other actors such as cloud vendor and web services providers. Another drawback is that it doesn't say if these actors have many-to-many relationships.

2. Evaluate business-level policies

The policies expressed in the SLA should be evaluated against the business strategy and policies. The data policies that consumers need to consider for inclusion in the cloud SLA when reviewing a cloud SLA are data preservation, redundancy, location, seizure and privacy.

The business level policies that should be considered for inclusion in the SLA include guarantees, list of services not covered, excess usage, payment and penalty methods, subcontracted services, licensed software and industry specific standards.

3. Understand SaaS, PaaS and IaaS

The third step is to understand what SaaS, PaaS, and IaaS are about and which type of cloud it is running on (private public or hybrid). Terms and conditions in the SLA depend on the complexity of control variables that the provider gives to the consumers.

The SLA for the SaaS is not as complex as the SLA for the PaaS. The only control the SaaS consumer (end users) has is to access the SaaS application, while the PaaS consumers (developers) have controls over the application development life cycle but not the virtual machines. The SLA for the IaaS is the most complex as the IaaS consumers (infrastructure specialists) have control over the virtual machines but not physical infrastructure.

4. Metrics

The fourth step is to identify what metrics should be used to achieve performance objectives. Some examples of availability and response time metrics are:

- Metric name in SLA
- Constraints
- Method and frequency of collection

5. Security

Consider key security requirements for cloud SLAs, including:

- Asset sensitivity
- Legal/regulatory requirements
- Cloud providers' security capabilities

Each country has a different set of privacy regulations than another country. For this reason, the consumers should know in which country the data would be stored in the cloud. One country may prohibit certain privacy data from outside the country, while another country may allow external privacy data.

6. Identify service management requirements

Identify service management requirements. They include what should be monitored and reported (for example, load performance, application performance), and what should be metered. They also include how rapid provisioning should be (speed, testing, demand flexibility) and how resource change should be managed.

7. Prepare for and manage service failure

The final step is to prepare for and manage service failure, determining what remedies should be provided (for example, service credits) and what are the liability limitations.

Then you need to understand how the disaster recovery plan will work when needed. The plan should define what service outage is, how unexpected incidents will be handled, and what actions to take when service disruption is prolonged.

An exit clause should be part of every cloud SLA in case either the consumer or provider wants to terminate the relationship.

Cloud metrics

In the SPEC Open Systems Group, the Cloud Computing Working Group wrote a Report on Cloud Computing to the OSG Steering Committee. The report provides a list of cloud metrics used to measure how well the tests are performing.

The list includes:

- Elasticity (provisioning internal, agility, scaling up and down)
- Throughput
- Response time

Elasticity metrics include provisioning internal, agility, and scaling up and down. Provisioning internal measures the time needed to bring up or drop a resource. Provisioning internal measures the time to bring up:

- A new instance (resource) on the IaaS
- A new instance of application server on the PaaS
- A new application instance online to meet increasing demand

Agility measures how well the workload can be scaled and how well the system is provisioned to be as close to the needs of the workload as possible.

Throughput is the amount of work the cloud can do per unit time. Response time is the time between when a request is made by a user and when the response is received by the user. Or more specifically how long does it take for the application to respond to the user's request.

What's missing from the cloud metrics

The following levels are missing from the cloud metrics list that could be used in determining guaranteed service levels in the SLA:

- **User threshold level.** Sets the maximum number of users concurrently accessing the application that consists of accepted or restructured service components decomposed from a legacy system.
- **Data requests threshold level.** Sets the maximum number of data requests that users can concurrently sent to the SaaS application.
- **Resources threshold level.** Sets the maximum amount of resources (for example, CPU, storage devices, disk space) that can be allocated to each SaaS user, PaaS developer and IaaS network specialist.

Knowing who sets the threshold levels is important when the consumers evaluate the SLAs. Setting the threshold levels depend on the control variables assigned to the SaaS, PaaS, and IaaS users and providers:

- **SaaS user.** The end user doesn't set any threshold levels. The only control the user has is to access the application from either the provider or the PaaS developer from a desktop, laptop, or mobile.
- **SaaS provider.** The provider sets the user threshold level. At a minimum, the provider controls operating systems, hardware, network infrastructure, SaaS application upgrades, and patches.
- **PaaS application developer.** The developer can set the data request threshold at the SaaS user level for SaaS users who are co-residents on the PaaS. The developer controls the development of all SaaS applications of loosely coupled service components with accepted dependencies in a full business life cycle, and runs them on the PaaS.
- **PaaS provider.** The provider sets the data request and resource thresholds at the developer level. At a minimum, the provider controls operating systems, hardware, network infrastructure, and resource management.
- **IaaS infrastructure and network specialist.** If the specialist hosts the PaaS, he may set the user and data requests for the PaaS developers. The specialist controls the operating systems, network equipment, and deployed applications at the virtual machine level.
- **IaaS provider.** The provider sets user, data requests, and resource threshold levels. At a minimum, the provider controls the infrastructure of traditional computing resources in the cloud environment.

Conclusion

In planning for SLA standardization, consider best practices for developing and following a standard way of doing things that multiple partners can use. New terminologies such as user, data request, and resource thresholds should be considered for use in determining guaranteed service levels. When the partners agree on the standard way of using terminologies when negotiating SLAs, they contribute to the process of SLA standardization. We need to build a team of developers, managers, business analysts, system engineers and make it easier for them to their job of standardizing the terms and conditions in the SLA for each deployment type: SaaS, PaaS and IaaS.

Related topics

- Practical Guide to Cloud Service Agreements Version 2.0
- Cloud computing on developerWorks
- Bluemix Developers Community in developerWorks
- Evaluate IBM products