EXECUTIVE BRIEFING SERIES

WHITE PAPER

Volume 1

ASPeace of Mind:
Crafting Service Level Agreements for Application Service Providers

Geneer®

NET INNOVATION

VOLUME I

Crafting Service Level Agreements for Application Service Providers

Contents:

Introduction

Ever since the development of web-based software applications delivered over the Internet, customers have been increasingly interested in the quality and reliability of the Application Service Provider (ASP) model. How potential customers evaluate ASPs in this department often determine whether or not they are willing to purchase services offered under this model.

The predictability and quality of service offered by ASPs is called a service level. Contracts that specify service levels are called Service Level Agreements (SLAs). In this white paper, we examine the elements that make up SLAs and the strategies ASPs can utilize in addressing service level agreements. We propose that ASPs think beyond the technicalities of service levels and instead focus on achieving a broader understanding of the entire customer experience and in this way achieve a high quality service experience for their customers.

The Internet value chain from top to bottom is made up of dozens of individual links that all must work together for an application to function properly. Today no one company can deliver the entire chain on its own. For a business to take advantage of a hosted application on the Internet, many different technologies must come together and work as one.

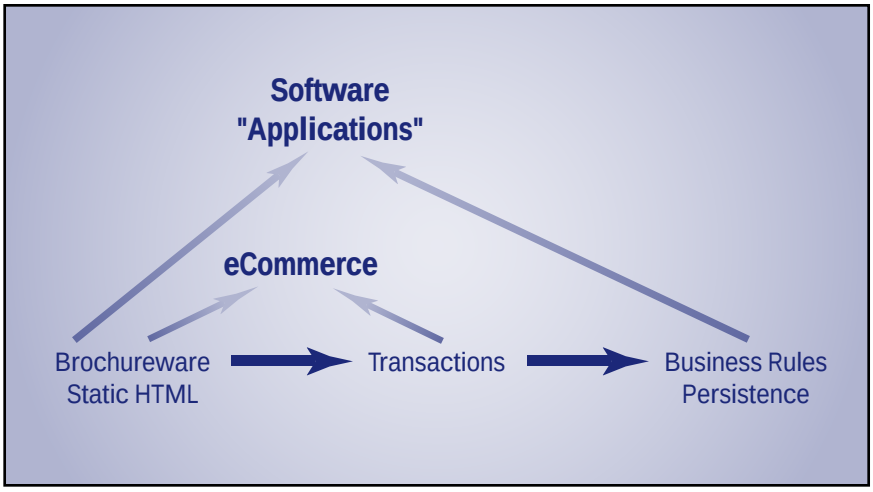
Definition

The acronym ASP stands for Application Service Provider. There are almost as many definitions of "ASP" as there are ASPs themselves. For purposes of this paper we use a simple, functional definition:

> *"An Application Service Provider (ASP) is an organization that provisions software applications over the Internet."*

This definition serves us well since it wraps a potentially diverse bundle of businesses into a single group for purposes of looking at the service issues common to all.

FIGURE 1: INTERNET APPLICATION CONTINUUM



One possible ambiguity in the above definition is the term "software." In the web world, it can be difficult to distinguish crisply among static web pages, e-commerce transaction systems, and "application software." For purposes of our discussion, it is not necessary to settle on a rigorous definition. When we refer to "applications" in this paper, we mean web-based software systems that:

1. Offer users a relatively large number of choices through the user interface
2. Contain greater algorithmic complexity in the business rules layer
3. Are made up of longer workflow patterns among screens

When applications contain more robust blends of these traits, users begin to perceive the web-based system as more "application-like." Moreover, from the perspective of technical architecture, performance, and service levels, such systems are more difficult to build and often are more challenging to deliver with a high level of predictable service.

Delivering Predictable Service Levels – the ASP Value Chain

In the world of network and computing systems, the Internet still is in its "wild west" stage of development. Having grown up in an environment of laissez-faire evolution, the Internet value chain from top to bottom is made up of dozens of individual links that all must work together for an application to function properly. Today no one company can deliver the entire chain on its own. For a business to take advantage of a hosted application on the Internet, many different technologies must come together and work as one:

- The user's desktop environment, including operating system and browser
- The networking connection inside the corporation
- The corporate firewall, including routers and switches from various vendors
- The Internet connectivity vendor (e.g. the physical T-1 line)
- The Internet Service Provider
- The backbone (peering) Internet Service Provider
- The ASP's ISP
- The ASP's Internet connectivity vendor
- The ASP's firewall
- The ASP's networking connections
- The ASP's hosting hardware (including backup and failover)
- The ASP's software, including operating system and application

Each link in this chain may be the responsibility of a different vendor, and delivered under a its own SLA. Thus, ASPs looking to deliver a predictable, high quality service level to their customers must orchestrate that service level across many partners and technology categories.

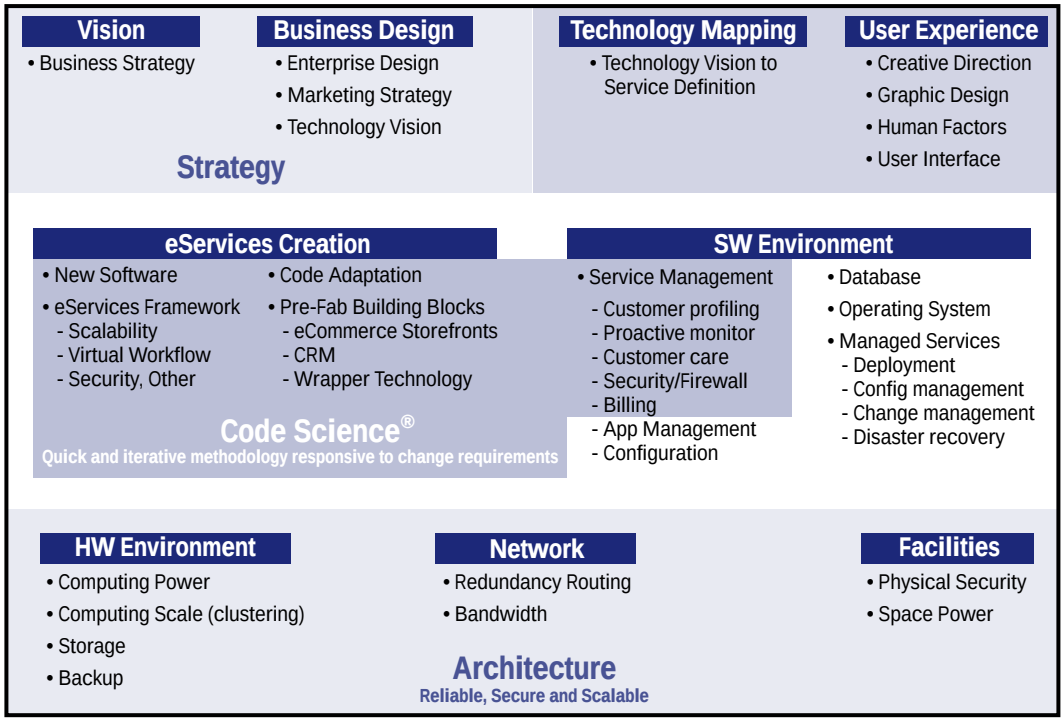
The ASP value chain can be viewed as a hierarchy of layers, each layer building upon those below it. In Figure 2, we group these layers into nine functional categories. Each category can be addressed independently, even though at delivery time, they must interoperate correctly. Let's take a quick look at each, beginning at the bottom:

Facilities

This layer refers to the physical site where the equipment running the application is located.

This includes equipment to house the computing and network gear, reliable sources and quality of electricity to run that gear, and appropriate security procedures to safeguard the systems it houses.

FIGURE 2: THE ASP VALUE CHAIN



Network

This layer refers to the bandwidth provided within the data center, its connection to one or more Internet backbones, and its eventual delivery to the customer premises. Network connectivity design should include fail over provisions. Extremely important here are high performance "peering points" – network meeting points in which data is handed off from one network to another. Poorly designed or overloaded peering points are a prominent source of ASP performance problems.

Hardware

This layer refers to the equipment itself on which the application is running. This includes computing servers, storage devices, and network routing gear. Each of these elements must be reliable, and each element should have an associated backup in case of failure. Moreover, each element must be designed to accommodate increased usage loads as the ASP increases the number of customers using the application. For example, a single application may begin with one server each to run the application code and the database. As usage increases, "clustering" and "load balancing" techniques may be used to distribute the processing workload among additional servers.

Software Environment

This label refers to four categories of support software that run conceptually "underneath" the application itself. These support layers are:

• Operating System: The standard software such as Unix and Windows that connect applications to basic computing services such as disk I/O, memory management, and multitasking.

• Database: Software services that provide high performance access and storage of data, as well as reliable backup, reorganization, and maintenance of the application's databases.

• Managed Services: Software tools that facilitate disaster recovery, version control, and configuration management (e.g., seamlessly adding new customers and users to the system).

• Service Management: Software capabilities that automate tasks directly related to the proper administration of the customer's experience of the application. Examples include application monitoring, billing, online customer support, and directory services.

Each link in the chain represents a potential point of failure, and the overall chain may perform only as well as its weakest link. From the customer experience, then, each link in the chain is equally important.

Application

This is the application service offering itself. Applications may be custom-written by the ASP or by an ASP specialist like Geneer on behalf of the ASP. The application might also be a port of a client-server application using a "wrapper" technology like Citrix Metaframe. Such technologies are useful for getting a quick "stake in the ground" in the market. However, they have the disadvantage of scaling poorly and thus seldom are thought of as a long-term solution. Problems with the wrapper approach include:

• They aren't hardware efficient – wrapper solutions will always require more machine resources to host than purpose-built ASP applications due to the overhead necessary to support the wrapper

• They aren't "multi-tenancy" capable – very often hardware and other resources must be dedicated to a single client rather than hosting multiple clients on a single installation

• They weren't built on Internet protocols and technologies – because the original application wasn't built to run on the Internet, there tends to be a high level of communication between the host and the user interface. This is fine in an installed software model but turns these applications into bandwidth hogs when run using wrappers.

• They weren't built with security in mind – most packaged software assumes a closed network environment, with low latency and high physical security. The

application may therefore not be secure enough for use on a public network like the Internet.

• They aren't workflow based and intuitive – many packaged software applications are harder to use than browser-based apps. This becomes a training hurdle that doesn't make it easy for occasional users to "drop by and use it once and a while."

• They have accumulated the baggage of non-core functionality – some packaged software suffers from feature bloat from playing the old world "feature matching" game needed to get in the door for enterprise product selection.

• They don't share data well with external systems – many packaged software systems are islands that don't integrate well with other applications, or which can't integrate using the Internet.

Given these problems, it's easy to see why many ASPs use ASP specialists like Geneer to create purpose-built ASP applications through component assembly and/or custom development.

User Experience

This refers to the design of the user interface and graphic elements of the application, and how those elements are combined into a productive and satisfying workflow experience. Design considerations include user profile, browsers to be supported, bandwidth constraints, and the positioning of the application service versus its competitors.

Technology Mapping

This is the step of translating a business vision into technology choices and architecture that technically can deliver on that vision. Not giving due thought to how a business model might evolve can lead to missteps that force expensive rework and delays in getting to market. Thus, explicitly laying out both the knowns and unknowns and incorporating the correct amount of flexibility into a design is a vital step that should offer a high return on investment.

Vision and Business Design

The ASP market continues to evolve and become more competitive. Having a clear starting vision of market niche, revenue sources and strategies, pricing, and target segment makes the job of building the right technologies possible.

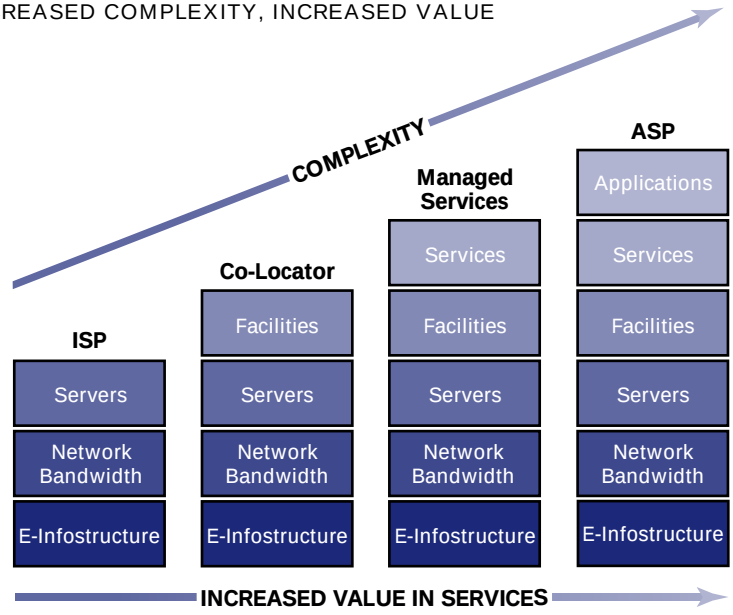
Observations Regarding the ASP Value Chain

The value chain helps us understand important aspects of an ASP service. Elements toward the top of the chart are strategically prior in importance to those below it. It is not uncommon for ASPs to begin building applications without adequate forethought to vital strategic elements like pricing models or identification of target market segments. While

time-to-market always is important, the market no longer rewards undifferentiated offerings without a coherent business path to shareholder value. Thus, there is a balance to be struck between too much planning and too little. The danger of the latter, however, can mean building a service offering that cannot evolve as the market matures. Therefore each layer on the value chain is generally constrained by the quality of what happens in the layers above it.

The layers toward the bottom of the chart are rapidly evolving toward commodity status and usually should be outsourced. The bottom three layers (facilities, network, and hardware) are commonly offered under the label "hosting" by outsourcing vendors like Exodus, Digex and Data Return. These should be insourced only when they directly tie back to some competitive advantage. While the level of reliability of these layers when outsourced is still not comparable to electrical and water utilities, they provide as a group a level of service significantly above what most ASPs can achieve on their own. The gap here will continue to widen. It is becoming more common also to outsource elements of the software environment and this trend is sure to continue. Competitive advantage in building ASP applications and software environments derives primarily from good alignment with business design and methodological process. The technology used must clearly support the business vision of the enterprise. When an ASP's product marketing, sales, and technology staff clearly understand the mandate and speak with one voice, they already have a competitive advantage over their competitors.

FIGURE 3: INCREASED COMPLEXITY, INCREASED VALUE



Methodological process refers to a software building process that shows results quickly and is responsive to changes as the marketplace evolves. Methodologies like Geneer's Code Science® are specifically designed to help ASPs get to market quickly, get early feedback from customers, and quickly incorporate market feedback into subsequent versions of the

application service. This also lets the developer prove its worth and deliver value quickly – reducing risk to the ASP whether it is insourcing or outsourcing the development work.

Our final observation regarding the value chain is to point out just how many links in the ASP value chain there really are. This is a key point in understanding why it is difficult to deliver high quality service levels with Internet-based applications. Each link in the chain represents a potential point of failure, and the overall chain may perform only as well as its weakest link. From the customer experience, then, each link in the chain is equally important. Since it is not practical for any one company to control every link, ASPs must identify the links that are most vital for supporting their value proposition, and then seek out partners for the rest with whom they can deliver a seamless customer experience.

While this is the difficult challenge of delivering web-based service offerings, it is also why the ASP industry exists. The illustration above shows the relationship between value and complexity. As a single entity is willing to support more and more value chain elements, the value it delivers to customers correspondingly increases. By supporting the application and the technical segments below it, ASPs deliver high value to customers by lowering support costs, reducing capital investment, and increasing customer focus on core competencies.

Having reviewed the ASP value chain, let's now look at what is a service level agreement and how ASPs can approach the challenge of crafting service level claims.

What is a Service Level Agreement?

For purposes of this discussion, a Service Level Agreement (hereafter, SLA) is a contract between two parties that specifies the performance and quality metrics of an application service offering – and specifies what happens when those metrics are not met.

We segment the metrics of the offering into two groups: the "technical" metrics and the "operations and maintenance" metrics. Let's look at each.

Technical Service Level Metrics

Technical metrics are the most familiar and widely debated of SLAs. The most common of these metrics are uptime and quality of service.

UPTIME

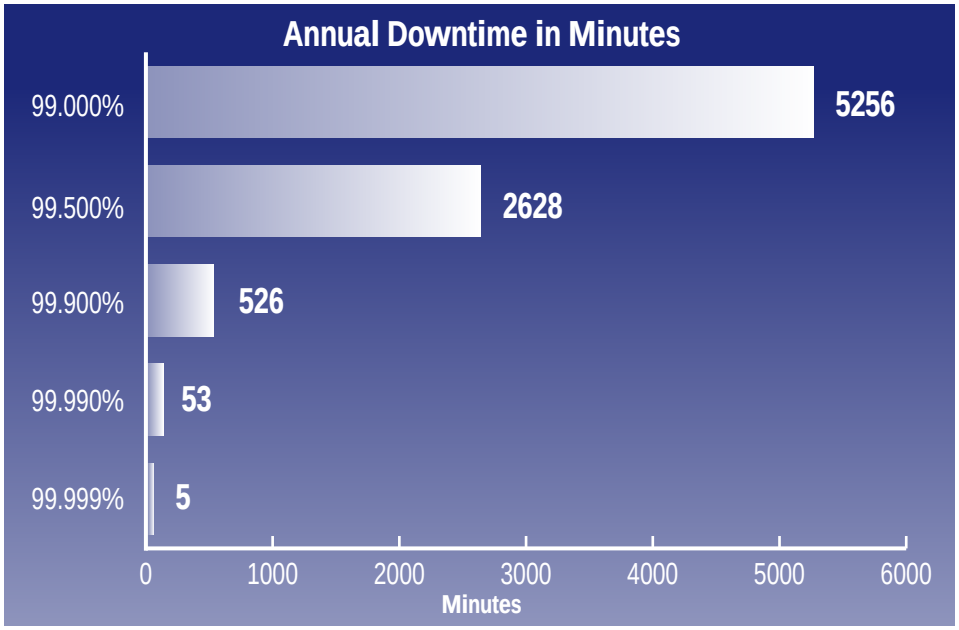
Uptime refers to what the customer can expect regarding the application service's availability. SLAs may specify this by individual technical components such as "server uptime," "network uptime," and "operating system uptime," although the more holistic "service availability" is more common. Uptime most often is expressed as a percentage of all available time. Common SLA percentages start at 98.5%. SLAs above 99% are common, but even this seemingly high percentage can be inadequate

for mission-critical applications as Figure 4 shows.

As can be seen, 99% uptime measured against a 24x7 standard implies the service can be unavailable for nearly four full days per year and still meet the metric. It is not until one achieves "four nines" (99.99%) and "five nines" (99.999%) uptime that the application begins to meet the standard of electrical utilities. This SLA is an important one to position carefully, both contractually and in terms of setting customer expectations.

Many ASPs try to aggregate partners' SLAs into a single offering from the customer's point of view. This lets them better control the customer experience, but requires accepting risk against partner performance.

FIGURE 4: THE FIVE 9s PARADIGM



QUALITY OF SERVICE (QOS)

QOS commonly refers to response time, latency, and effective throughput thresholds. QOS assumes application availability and then seeks additionally to define a minimum "customer experience" standard by employing metrics that nail down "time to response" from the customer's point of view. Anyone who has ever used a web application that – while available – was excruciatingly slow, understands the importance of this metric.

QOS SLAs can be difficult to deliver because problems can occur at many points throughout the network. In fact, some of the biggest risks in the customer experience are in the "last mile" – the network connection from the local provider to the end customer. Here a poor ISP or an overloaded peering point can ruin the customer experience of an otherwise well-crafted architecture.

There are a few basic rules that go a long way to help understand how hard it can be to offer a high quality technical SLA.

1. The weakest link in the value chain determines the highest technical SLA the ASP can offer. If the network is up only 98% of the time, the uptime of other links is irrelevant – your application will be available no more than 98% of the time.

2. Strictly speaking, you can affect service levels only on the portions of the value chain you directly control. This is one of the great drivers toward consolidation in the lower segments of the value chain: Those who can own control large portions of the value chain should be able to win a competitive advantage by offering a single point of accountability for that value chain. Many ASPs try to aggregate partners SLAs into a single offering from the customer's point of view. This lets them better control the customer experience, but requires accepting risk against partner performance. This must be addressed in partnering agreements and negotiated so that appropriate compensation is addressed if the partner performance falls below committed service levels.

3. Technical SLAs expressed in percentages must be based on sampled historical experience. ASPs that write SLAs expressed in availability percentage without either measuring their performance or aggregating a statistically significant sample put their business model at risk. Unfortunately, this remains a relatively common phenomenon that sooner or later must change.

Improving Weak Service Levels

For vendors that deliver mission-critical application services, there are two delivery strategies that help address the structurally weak position the industry finds itself in with respect to SLAs.

Intranet Delivery

The first means of overcoming problems in delivering Internet-based service levels is to deliver the service via the customer's intranet. This involves installing the application on the customer premises. In theory, a higher service level should be achievable by co-locating at the customer because the application no longer needs to travel across public Internet segments that may offer the lowest SLA segments of the value chain.

There also can be possible integration advantages as well as psychological comfort on the client's part that the application is under its control. However, in practice many intranets perform measurably worse than the average public Internet-delivered service offering. Moreover, there are additional support costs and operational risks in this approach related to the loss of a single, central point of support that defines the ASP model. Many ASPs are designed around a single geographic point of ongoing support and maintenance, and intranet delivery can adversely affect their operational excellence model.

Private Networks

Another common tactic for improving service levels of mission-critical apps is to reduce

reliance on the public Internet. This approach delivers data from a data center with access to a backbone that travels to the customer premises with perhaps only a single hop from the backbone to a private circuit directly to the customer premises. This design avoids sometimes-crowded public peering points and keeps traffic on high bandwidth backbones and private dedicated lines. This approach can be more expensive, but often improves predictability of QoS elements.

Another variant of this approach is to create a private virtual circuit (PVC) that lets the ASP "drill out" a dedicated virtual path through the Internet to the customer premise. This lets ASPs monitor actual performance through the virtual circuit and in some cases provision dedicated bandwidth that should improve predictability.

Most successful ASPs agree that it is the human factors around procedures and processes that can have the biggest impact on both actual application availability, as well as customer satisfaction.

Operations and Maintenance Service Level Metrics

Operations and maintenance metrics, also referred to as "O & M" metrics, are service response norms that ASPs put into writing as a way of increasing customer confidence in the ASP's process maturity. One attractive element of these SLAs is that they are largely under the control of the ASP. Moreover, many of the large-scale outages of application services come from human error resulting from failures in process. Many customers focus on this group of metrics as a means of assessing how sophisticated ASP operations are.

Here is a brief overview of some common "O & M" metrics:

Help desk response time

What is the guaranteed maximum clock time before help desk either responds to a trouble alert?

Time-to-problem identification

Once a problem is discovered, what is maximum time to identify the root cause of the problem. Root causes can be expressed in terms of the value chain layers, such as application, operating system, hardware, and network. Usually this SLA defines an escalation path that brings increasingly higher-level resources to bear as time elapses.

Time-to-restoration

What is the maximum time to restore the service once the root cause is identified?

Backup schedules and procedures

Define minimum timetables and explicit conditions under which backups are performed. Define standard operating procedures for how physical media is stored. Define maximum time a database "restore" procedure should a database become corrupted.

Security procedures

Define standard operating procedures for security-related issues including authorization, access, passwords, and escalation path of security issues.

New version procedures

How are new software versions installed and tested? How are bug fixes implemented? What prior notification takes place?

New customer provisioning

Adding new customers to an existing system potentially affects existing customers. What are the procedures for adding new customers?

Communication norms

Who is informed of scheduled outages? Is the customer informed if the service goes down even if they don't complain? If the primary contact is unavailable, what is the escalation path for communicating with the customer?

The SLA Business Driver: Creating a Customer Experience

A crucial point in crafting a service level policy is to understand that customer interest in SLAs comes from their attempt to nail down a minimum customer experience for the service offering. Yet, a customer experience is much more than just the sum of the contractually agreed upon SLAs. ASPs therefore should pay attention to crafting the overall customer experience, and treat SLAs as one aspect of that experience. When correctly executed, this should lessen customer focus on technical metrics, as well as improve overall customer satisfaction with the offering.

ASPs willing to create an "operational excellence" model for their service have a great opportunity here. By thinking holistically about the customer experience of their service and their company, ASPs can wrap the SLA issue into a larger context that includes the more subjective, but often equally important elements of the service experience. Most successful ASPs agree that it is the human factors around procedures and processes that can have the biggest impact on both actual application availability, as well as customer satisfaction.

How should an ASP make a start at creating an operational excellence model? ASPs should begin by addressing the following questions related to the customer experience:

- What are my core claims for this service?
- What aspects of the delivery chain can I really control?
- What geographical areas am I able to support (including Europe and Latin America)
- How do my customers want to communicate with me?
- What should the customer experience be when I have to take down the service?
- What should be the customer experience when something bad happens?

Even when there is a service fault beyond the ASP's control, customers want assurances as to what actions are underway to address the situation. They want late-breaking information or at least best guesses about when service will be reinstated. Remember that someone at

the customer site may be accountable, and without answers, may be looking bad within the organization. Often it is these "moment of truth" experiences that form the major part of the customer's impression of the ASP service offering.

Once self-assessment questions are answered, O & M strategies can be crafted around customer service. Customer service plans must be sustainable and predictable along the dimensions of support availability (e.g., 24x7, M-F 8am – 8pm, business hours only) and claimed support modes. Support modes include client communication media such as:

- Email
- Phone call in
- Phone call back
- Real-time web monitoring
- Real-time web-based customer service chat
- Self-help facilities like online FAQs and message boards
- Web "crisis" boards – separately hosted apart from the service itself

ASPs often outsource and partner for parts of the service offering. When this occurs, they must pay special attention to where in the value chain the partnerships intersect and overlap in order to create procedures that avoid the perception of finger pointing among partners from the customer perspective. Crucial areas to discuss and agree include:

- Who takes the first call from the end customer?
- Who is accountable for the "triage" (the job of figuring out where the problem resides and who is responsible for fixing it)?
- Who keeps the "run-book" current? The "run-book" refers to the standard operating procedures for running and maintaining the ASP service. While many ASPs create these procedures, they often become lax in documenting changes to those procedures. In an ideal world, the ASP should capture this knowledge in order to improve its service and understanding of customer preferences and requirements.
- Who is accountable for improving processes as the partnership gains experience working together?

One common landmine among start-up ASPs is offering telephone support via outsourced call centers. ASPs should be cautious in outsourcing their telephone support until they have gained sufficient insight into the common topics that generate the customer calls in the first place. Outsourced call center support works best when the ASP clearly understands the most common call-in topics and can train the call center to provide a satisfactory response to the caller. Until an 80/20 rule of thumb is reached, ASPs should consider maintaining their own in-house support. When an ASP eventually does outsource its support, there should be clear escalation paths when the call center cannot resolve the issue.

In ASP partnerships, it is vital that the entity that "owns" the customer relationship also own it in terms of providing a single point of support that takes responsibility for overall customer satisfaction with the service. Thus partnerships must go well beyond "doing the deal," and seek to craft a seamless, unified experience for the customer's perspective.

The Service Sale

We have seen in this paper that SLAs are only one aspect of the overall customer experience and that by addressing the customer's underlying business driver, the focus on technical SLAs may be reduced. This points up the importance of framing service and SLA claims early in the customer relationship.

Skillful expectation setting with respect to issues common to all ASPs can reduce unrealistic expectations and focus customer attention on other elements that can be written into agreements and fulfilled with higher predictability.

Thus, a key time to ensure effective communication of SLAs is during the sales process. Experience shows that leaving SLA talks as a last issue to be resolved in contract negotiations can cast a negative pall over bringing on a new customer. Skillful expectation setting with respect to issues common to all ASPs can reduce unrealistic expectations and focus customer attention on other elements that can be written into agreements and fulfilled with higher predictability.

At a recent ASP conference, Richard Dowling of EMC described results from a survey conducted among hosting providers. That survey yielded the statistic that an astonishing 20% of customers had SLA penalty clauses in effect for which they were receiving compensation from the provider. This points up the importance of crafting SLAs that the ASP can indeed deliver upon.

Here are some tactics that can help you set realistic expectations with potential customers early in the sales process:

- Educate the client about the value chain common to all ASPs. The goal of this is to coach the client not to negotiate for what is excessively expensive or impossible to deliver.
- Establish clear definitions of technical jargon that map to customer expectations. For example, does the term "network" refer to the

    a) network within the walls of the data center?
    b) the network backbone?
    c) the end-to-end network including "last mile"?

  Many ASP contracts bury these definitions deep within contract language only to create unrealistic expectations that end in unhappy relationships.

- Before discussing SLAs, discuss the client's predicted usage patterns. For example, if a client sees no need for access to the system between midnight and 4am, SLAs perhaps can exclude this period (or specify a reduced target SLA) that then can be used as a window for bring up new versions of the service or provisioning new customers.

• Focus customer attention on the O & M metrics and processes that the ASP enforces that add predictability to its service offering.

• Focus attention on self-monitoring and self-service support and provisioning tools that add a flavor of control for the customer.

• Consider creating tiered levels of service that include premium levels with higher SLAs. This helps the customer think through what SLAs they really do require. It also establishes the value of an increased SLA and helps the recoup the investment in delivering the higher SLA.

• Describe successful case studies in which the positive human experience of the service removes the urgency for nailing down abstract numbers that don't necessarily add up to a user experience in any case.

Getting Started Checklist for ASPs

Here are steps relating to SLAs to consider as you craft your ASP offering.

1. Identify required role of technical SLAs in the offering

   • Is the offering 'mission critical' to client's business on a minute-by-minute basis?
   • Can you negotiate for scheduled downtime?
   • Must you offer a technical service level agreement at all?

2. Determine best technical SLAs that can be offered

   • What are the SLAs offered by your value chain partners?
   • What statistical data do you have regarding partner performance as well as your service offering's peformance?
   • Will enterprise customers accept a delivery approach that relies on private virtual networks or dedicated lines to the customer premise?

3. Determine 'O & M' SLAs

   • What are the operational policies necessary to make the offering predictable?
   • What are the structural management processes required to ensure that policies are enforced effectively?
   • How will you capture and use your learning and experience regarding best delivery and support practices as the service evolves?

4. Create service policies with respect to "moments of truth"

   • How do you handle notification with respect to performance problems?
   • What compensation do you offer customers when failing to meet the SLA?

• How will you communicate to your customer that you are going 'above and beyond' with respect to problems that are not directly under your control?

5. Identify marketing and sales tactics around service levels

• Can you segment the market around service levels?
• Can you offer tiered levels of service with corresponding pricing levels?
• What parts of the customer service strategy can be delivered via the web to give customers a greater sense of control?

Summary

The topic of SLAs is complex and the subject of much jargon and technical ambiguity. You can start to cut through it by focusing on the customer's underlying interest in SLAs: they are trying to quantify a minimum performance threshold for the offering. By addressing the SLA question holistically in terms of both technical and "O & M" SLAs, ASPs provide an improved context for discussing them in a fashion that meets this driver. This in turn creates a better overall customer experience as well as increases the ASP's competitive position.

Geneer®
NET INNOVATION

1400 East Touhy Avenue, Des Plaines, IL 60018-3340
phone: 847.294.0300 • toll free: 800.443.6337 • fax: 847.294.0358 • www.geneer.com