



# Managing User Expectations with Service Level Agreements

By Rachel Carmichael

**W**hen people talk about availability, the discussion almost always begins with hardware. Numbers and sizes of servers, disk arrays, communication lines and, on occasion, additional physical data center sites. Once the hardware is out of the way, the talk turns to software. Do we have a backup of our data? What about the programs that we use to manipulate and/or access the information?

I believe that the discussion should start one step back with a Service Level Agreement (SLA). What is an SLA? Whether you know it or not, you participate in dozens of SLAs each day. Do you expect the coffee shop on the corner to be open at the time they have posted on the door? What about the train or bus you take to get to your office? The schedule they run (or sometimes do not run) on is an SLA.

You, yourself have a Service Level Agreement with the company you work for. You are expected to be at your desk at the same time every day, and to put in a certain number of hours. You expect to be allowed to take a lunch hour and to have a certain number of vacation days. All of these are SLAs. I am sure if you stopped to think about it, you could come up with quite a number!

## What is an SLA? Why do you need one?

So what exactly is an SLA from a system user perspective? A Service Level Agreement is a document or set of documents that define what you expect to provide or have provided to you. The SLA should spell out, in detail, the agreement you have made with your users and with your service providers. Remember that anything you leave out can, and probably will, come back to haunt you at a later time. When you write the SLA, pretend that you are paying triple for everything that is not included that has to be done anyway. Just as it has always been more expensive to go back and fix code when you do not plan properly, it is expensive, if only in terms of your time and energy, if you do not properly plan your SLA. It is always better to be more, rather than less, detailed in this document.

Before discussing what belongs in the SLA, think about when you should be writing this document and getting user buy-in. Most people think of writing any sort of documentation after the design, coding and testing are complete and it is time to release for production. In the case of the SLA, that is exactly the wrong time to begin thinking about what you should include. If you wait until the very end of the project, the following negative results will occur:

- You will have no time to investigate options and costs.
- You will have no time to educate the user population about what they can realistically expect.
- You will have no time to negotiate something you can actually provide.

The best time to start working on the SLA is at the beginning of the project, when you can design the hardware and software systems to meet the availability, response time and contingency requirements. The SLA should be a living document throughout the life of the project.

The SLA can also serve as the entrance criteria for preproduction and operational readiness testing. The test cases and scenarios should be adapted with an eye to verifying that the SLA is an achievable benchmark.

You might have to start with a very basic outline, just a list of the items you want to remember to include. As the project progresses and you get more and more information, you can “fill in the blanks.” You cannot discuss backup schedules or recovery procedures until you have some idea of how much data will be backed up, how often it has to be backed up, what you are using to do the actual backup, what is allowable data loss, and what is acceptable recovery time. But you can include a section that discusses the backup and recovery strategy. Once you have a better idea of what you are dealing with, you can go back and update that section with the information.

What should you include in your SLA? You should include sections for normal availability, maintenance, backup and recovery, and performance. Also think about the following questions:

- How much time it will take a vendor to arrive on-site with technical support and parts in case of problems?
- When do you escalate the problem? To whom?
- When do you go to your contingency plan?

There may be other things you need to include, depending on your particular system and data center. If your company uses a hosting company to handle your production systems, you may not need to include all the items as they may already be covered in the contract with the hosting company. They should, however, be covered somewhere.

## Normal Availability Times

The first part of the SLA should discuss the normal availability times for your application and database:

1. Is this a system that needs to be available 24 hours a day, 7 days a week, 365 days a year, like a web site?
2. Or is this a system that is only used during business hours, like the computers used to manage the cash registers of a brick and mortar store?
3. What hours is the system really going to be used?
4. Will you have users in other time zones?

Think back to when you were a child. You wanted to do something “because everyone else is doing it” and your mom would reply “Just because everyone jumps off a bridge, would you jump too?” Remember that advice when it comes to defining the availability times for your system. If the system is used by a local back office and they do not work weekends, there is no reason to promise 24x7x365 availability “because we can.” If the system is a development system, you can and should work into the agreement that while the system will be available for use on weekends, there will not be on-call support or immediate response to problems, unless management wants this included.

Make sure that your users understand the reality behind their availability requests. The following chart lists the maximum number of minutes and hours your systems can be down to make the associated availability percentages. They should understand that these numbers include downtime for regular maintenance and code changes, as well as unforeseen problems. While it may be prestigious to say “We meet five 9’s,” there is a cost associated with the higher availability in terms of physical hardware and software systems to support the requirement.



Availability needed (%)	Downtime (min/year)	Downtime (hours/year)
99.00	5256	87.6
99.25	3942	65.7
99.50	2628	43.8
99.75	1314	21.9
99.90	526	8.76
99.99	53	0.88
99.999	5	0.08

### Performance Requirements

Performance requirements can be one of the most difficult factors to define. Most people do not think of response time in seconds. They just want it “fast.” But “fast” can mean different things to different people, depending upon the amount of information you are processing and returning, the format in which you are returning the information and what else is going on. If your system is an online web site, response time should be a high priority. It is better to return some of the information quickly than to wait for all of the data to be collected and then sent to the browser. If you are working with a reporting system, especially if it is a data warehouse that is processing large amounts of data, your end users are more likely to be willing to wait several minutes for responses.

In general, people tend to wait no more than three seconds before they start getting impatient and hitting the send or back button on a web site. There was a time when 1200 baud modem response time was considered fast.

Nowadays, with the proliferation of cable modems and DSL connections, the time people are willing to wait for a response has dropped. You need to factor in, not only the time it will take for your database to return the first rows of the request, but also the network lag you may have and how many intervening servers there are between the database and the end user. You might have sub-second response time from the database, but that does you no good if the network connection between the database server and the application server or the application server and the Web server is slow. Make sure that you have tested the connection response times under both heavy and light use, before you agree to a performance response time.

When you think of performance, do you think only of how quickly you can get the information out? Do you ever consider how long it will take to get the data into your database? Load time is an often forgotten component of performance requirements. Be sure to run tests on all of the parts of your system, both insert and extract, before you commit to performance times.

### Regular Maintenance Schedule

As anyone who has worked with Oracle for even a short time knows, there are always patch releases and new versions being released. Operating systems release new versions and patches as well. In addition, businesses rarely remain static and requirements change and grow. Every time you make a change to a program or series of programs, you will need to release the change to production.

If you have a regular maintenance schedule, your users will know when there will be an outage of service and how long it will last. It is always much easier to remind someone in advance of a disruption in service than it is to explain why the system was unavailable. Your users can plan for the downtime and you will not have to spend time sending out maintenance notices and explaining why things are being done.

With each new release of the database kernel, more and more maintenance functions are available while the database is up and running. There are, however, still some things that must be done while the database is not available. There are 137 initialization parameters in Oracle9iR2 for Windows

2000 that cannot be modified without shutting down the database and restarting. For example, switching to or from Automatic Undo Management, will require a shutdown of the database in order to edit the initialization file and restart the database. Changes like these should be scheduled under the regular maintenance agreement.

It is a good idea to have a regular maintenance window scheduled, even if there are no maintenance jobs for a particular time slot. Users do not usually complain if the system is unexpectedly available.

### Recoverability

When was the last time you backed up your PDA? What would you do if the disk was damaged and you lost all of the data you have stored in it? How easily can you reconstruct the address book and daily calendar?

Now multiply the horror of that experience by a thousand percent. Most database servers have hundreds of gigabytes of disk storage. What happens if a disk or controller fails? While you might feel the pain of not having your records and you might have to spend some time with a phone book to reconstruct the information, imagine the loss of time and revenue if your company does not have its data backed up.

Think about what it will take to recover from a server crash, a disk crash or even the total loss of a data center. Can you handle the “Oops, I didn’t mean to commit that change” call from a user? Define “data loss” with your end users as well as what is “acceptable loss” in terms of system downtime and plan a backup strategy and recovery plan to deal with those requirements. Test the recovery plan often. Oracle provides some features, such as LogMiner and Flashback Query, to help deal with accidental data loss. You can do table-space-point-in-time recovery if you have planned your database and recovery strategy properly. These features work to allow you to keep as much of your database available as possible while you recover from problems.

You may have to design and write your own backup scripts. You may have the luxury of using RMAN or a third party backup product. You can do exports, offline backups, online backups or some combination of all of them.

This is the place to list not only what you are backing up, but how and when. Decide whether or not you want to do incremental backups, whether you need to do daily backups or if you can manage with weekly backups. Do you need to backup the archived logs immediately or can you wait a day or an hour? The amount of data that the users are willing to lose will determine how often the backups are done. This section should be written in conjunction with the contingency section, as the types and timings of the backups and recovery plans will determine how quickly you can move to contingency.

### Response Time for Hardware/Software Problems

Most applications and servers have both internal and external support services. The internal support for the application can be the developers and the DBA while the external support might be your support contract with Oracle. The same goes for the hardware itself, as you will have system and network administrators to manage and diagnose the operating system and/or network problems, while your company should have contracts with the companies that supply the hardware.

While it would be wonderful if, like magic, as soon as a problem developed you had the personnel onsite to fix it, this is wishful thinking. However, you can write into your SLA the maximum acceptable wait time for someone to begin to acknowledge the problem. You can include penalties (for the external support personnel) and time limits before you call in additional help. You cannot set a time limit within which the problem must be fixed, but you may be able to say something like “If the hardware is not functional after x hours, ABC Company will replace the part instead of repairing it.”

*continued on page 42*

## Escalation Procedures

The tag line for the movie Ghostbusters was “Who’re you gonna call?” That’s exactly what the escalation procedures section of the SLA is meant to define.

This section should outline what happens, when, and in which order, if there are problems. You may want to include a checklist to help determine what the problem is so that you can specifically define who to call. A phone call from the user reporting “Oracle is down” is not helpful in determining if the network administrator, system administrator or database administrator should be called. Or maybe it is just an application problem and the developer on call should be paged. A list of questions will help you to refine the problem area.

After you have defined the proper job function to handle the problem, it is time to identify the actual person to notify. You may have a group of people who rotate taking off-hour calls, so you might just list the “on-call” pager. Or you may have a roster of names and dates and have to go through the list, based on date, to make the phone call.

Is your data center coverage automated? Nowadays, many data centers run “lights out” when there is no actual person there to make a phone call. You should consider automating the notification process, either through homegrown scripts or a third-party package. An advantage of automation is that you can look for the problems yourself, rather than wait for an end user to make a call. You can identify a problem and be in the process of fixing it when the call comes in. You look like a hero, the problem is fixed sooner and everyone is happy.

Whether you send out automated messages or have a person making phone calls, you need to consider the timing of the notification and when you decide to move on to the next person on the list. Be careful not to send a second notice too quickly on the heels of the first one. If you send a page every five minutes, the person on call will not have time to respond and clear the problem before you page them again. The queue of messages to send may grow so quickly that it becomes impossible to log in and clear them.

There should be a clearly defined “chain of command.” At some point in the process, it may become necessary to inform upper management of the problem because it will have an impact on business. Who to call, when to call and why to call should be part of this section of the SLA.

## Contingency Procedures

“No computer ever goes down, no database ever crashes and no programmer ever writes buggy code.”

Now that you have stopped laughing, think about the implications of this. If things break, and break badly, what do you do? This is not a decision you can make on your own. Contingency strategies cost money and can cost a lot of money, if the decision is made to have a complete data center ready and able to instantaneously take over if the primary system fails.

What you can do is present options to your management and business users, outlining the various possibilities and associated costs. Include the costs of extra licenses if you choose to use Data Guard as well as the time it will take to restore backup tapes if the decision is made not to keep a complete data center at the ready. Once the decisions have been made, they should be recorded in this section of the SLA.

Once the decisions about the timing and costs of moving to contingency have been made, you can document the process. There should be a separate highly detailed document that describes exactly what is done, when and by whom. The SLA should include the major points of the contingency document, including answers to the following questions:

1. How long will it take to have a fully-functional data center up and running?
2. Which applications will be brought up and in what order?
3. What personnel will be onsite?
4. How often is the contingency plan tested?

Be sure to document the allowable downtime before you go to contingency, the allowable downtime to get to contingency, as well as the acceptable data loss when moving from a primary site to a secondary site. Don’t forget to include documentation about how to “get home” to primary once the problem has been resolved and what the downtimes and data losses associated with the move back are.

## Conclusions

If you have been reading this article carefully so far, you have noticed that a word appearing frequently throughout this article is “depending,” and with good reason. Like most things, designing a Service Level Agreement is an art, not a science, and the contents of your SLA will be different from those of the SLA for someone else. This article has tried to give you an outline of the items that you will find in almost every SLA, but not all of these may apply to your specific situation.

Your company might decide that your system does not need a contingency plan and that they can survive for several days without the system, until the problem can be resolved. Or you might decide that you want to include a separate section on Recovery Testing. The important thing is to have an SLA, and to refine it over time. Each time something not covered in the SLA occurs, it is a good idea to update the SLA so that the new process is covered. Remember to update the SLA as new application functionality is added to your system. Following all of this advice will make recovering from problems much easier and more efficient.



### About the Author

**Rachel Carmichael** has been an Oracle DBA for over 11 years. She chaired the DBA Special Interest Group for the New York Oracle Users Group, was the meeting agenda coordinator for NYOUG, is both Chauncey and OCP certified and has presented at various international and national conferences. She co-authored the Oracle Press books: *Oracle SQL and PL/SQL Annotated Archives*, 1999 with Kevin Loney, *Oracle DBA 101*, 2000 with Marlene Theriault and James Viscusi, *9i Instant Scripts* with Kevin Loney and Megh Thakkar, and *Oracle9i DBA 101* with Marlene Theriault and James Viscusi. She can be reached at [wisernet100@yahoo.com](mailto:wisernet100@yahoo.com).