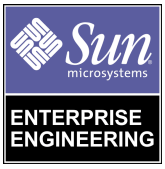
Service Level Definitions and Interactions

By Adrian Cockcroft - Enterprise Engineering

Sun BluePrints OnLine - April 1999



http://www.sun.com/blueprints

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303 USA
650 960-1300 fax 650 969-9131

Part No.: 806-3836-10
Revision 01, April 1999

Service Level Definitions and Interactions

Computer systems are used to provide a service to their end users. System managers are responsible for the quality of this service. System and application vendors provide a range of components that can be used to construct a service. A service must be available when it is needed and must have acceptable performance characteristics. This section defines service level management, starting with a high level view and devines a terminology based on existing practices.

Service Level Management involves interactions between end users, system managers, vendors, and computer systems, as shown in FIGURE 1 on page 2. A common way to capture these interactions is via a Service Level Agreement (SLA) between the system managers and the end users. In reality there are many additional interactions and assumptions that often are not captured formally.

The workload definition includes a *schedule* of what work is run at different times of the day. For example, daytime interactive use, overnight batch, backup and maintenance periods. For each period, the workload mix is defined in terms of applications, transactions, numbers of users, and work rates. The service level definition includes availability and performance for *service classes* that map onto key applications and transactions. Availability is specified as maximum downtime over a period of time. Performance may be specified as response time for interactive transactions, or throughput for batch.

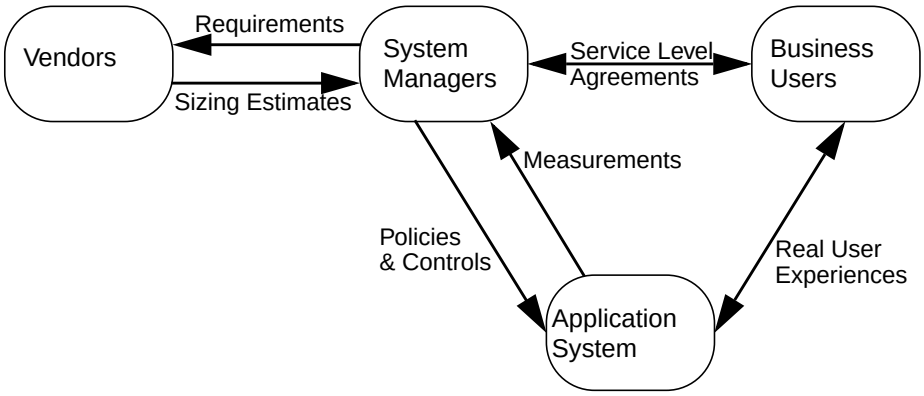


FIGURE 1 Service Level Management Interactions

Requirements

System managers establish a set of Service Level Requirements and a workload definition. The workload is defined by the managers and is often the first description to be created. The requirements are communicated to vendors, who respond by proposing a system to meet these requirements.

Sizing Estimates

Vendors measure the performance of their system using either generic benchmarks or a customer-specific benchmark test. Vendors provide a sizing estimate, based on the service level requirements and workload definition. Vendors provide reliability data for system components. They can also provide availability guarantees and performance guarantees for production systems with a defined workload. Vendors cannot provide unqualified guarantees because there are too many application and environmental dependencies that are outside their control.

Service Level Agreements

A service level agreement (SLA) establishes a user-oriented view of the workload mix and the service levels required. This may take the form “95th percentile response time of under 2 seconds for the new-order transaction with up to 600 users online during the peak hour”. It is important to specify the workload (in this case

the number of users at the peak period), both to provide bounds for what the system is expected to do, and to be precise about the measurement interval. Performance measures averaged over shorter intervals will have higher variance and higher peaks.

The agreed service levels could be either too demanding or too lax. The system may be quite usable and working well, but still failing its SLA. It could also be unusable, but still working within the agreed service levels. There needs to be a continuous process of updating and refining the SLA.

Real User Experiences

The actual service levels experienced by users with a real workload is a subjective measure that is very hard to capture. It is quite common for problems to occur that affect parts of the system not covered explicitly by the SLA, or for the workload to vary from that defined in the SLA. One of the biggest challenges in performance management is to obtain measurements that have a good correlation with the real user experience.

Measurements

The real service levels cannot always be captured directly, but measurements are taken as representative of the real user experience. These measurements are compared against the SLA to decide whether a problem exists. For example downtime during the interactive shift is measured and reported. A problem could occur in the network between the users and the application system which causes poor service levels from the end user point of view, but not from the system point of view. It is much easier to measure service levels inside a back-end server system than at the user interface, but it is important to be aware of the limitations of such measurements. This problem must be carefully considered when the SLA is made and when the service level measurements are being defined.

Policies and Controls

System managers create policies that direct the resources of the computer system to maintain service levels according to the workload definition specified by the policy. This workload definition is closely related to the SLA workload definition, but may be modified to satisfy operational constraints. It is translated into terms that map onto system features. Example policies include “A maximum of 600 interactive users of the order entry application at any time”, “order entry application has a 60% share

of CPU, 30% share of network, and 40% share of memory", "if new-order response time is worse than its target, steal resources from other workloads that are over-achieving."

The policy works only in terms of the measurements it has available. If the wrong things are being measured, the policy will be ineffective. The policy can control resources directly or indirectly. For example direct controls on CPU time and network bandwidth usage might be used to implement indirect controls on disk I/O rates by slowing or stopping a process.

Capacity Planning and Exception Reporting

The measured workload and service levels should be analyzed to extract trends. A capacity planning process can then be used to predict future scenarios and determine action plans to tune or upgrade systems, modify the SLA, and proactively avoid service problems.

In cases where the measured workload from the users exceeds the agreed workload definition or the measured service level falls short of the agreed level, an exception report is produced.

Resource Management Control Loop

For a resource to be managed it must be both measurable and controllable. A control loop is setup that measures the performance of an application subsystem, applies policies and goals to decide what to do, then uses controls to change the resources being provided to that subsystem. This loop may be implemented manually on a timescale measured in days or weeks by reconfiguring and upgrading entire systems, or it may be automated in software and run as often as every few seconds. The loop is shown in FIGURE 2.

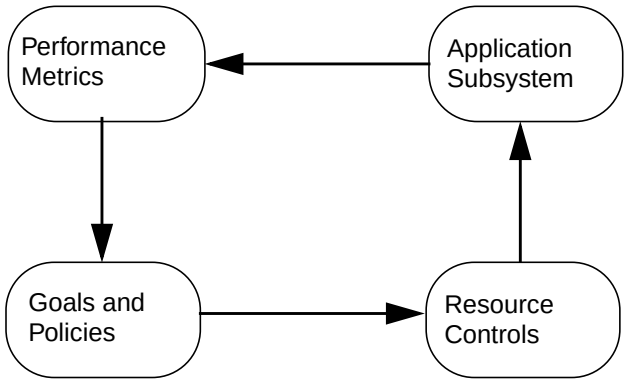


FIGURE 2 Resource Management Control Loop

A complete system implements many control loops. A brief digression into basic control theory is provided at this point to help explain the behavior of such systems.

A Simple Approach to Control Theory

We spend so much of our lives operating control loops that it is actually quite intuitive to most people. Designing a control loop is more complex and requires a more explicit understanding of what is going on.

You start with an objective, such as to steer a car around a corner while staying in lane. You apply a control input by turning the steering wheel to your best guess of the amount that will get you around the corner, then after a delay the car responds, you measure the response, compare it with what you wanted, and obtain the error difference. If the difference is zero you don't need to change the control input, if the turn is too tight you need to reduce the input, if the turn is too wide you need to increase the input. You have to decide how much extra correction is needed to compensate for being wrong the first time, and also whether the car has finished responding fully to the initial input.

The first time you tried a car driving game on a computer, you probably swung wildly from side to side. This is caused by over correcting too late, because you don't have the same motion sensing inputs you have in a real car. When the car is oscillating, your corrections may be delayed to the point where you are turning the wheel the wrong way at the wrong time and you could spin off or crash. Eventually, you learn to react based on what you see on the screen and make smaller corrections more quickly to keep the car on track in a stable manner.

In control terms, you are applying *negative feedback* to the system. You take the error difference between what you wanted and what you got, and apply the inverse of the error to the system to reduce the error in the future. The rate at which you measure and apply corrections is called the *control interval* and the rate at which the system responds to changes is called the *time constant* for the loop. The amount of the error that you feed back changes the characteristic behavior of the control loop. If you feed back a large proportion of the error with a short control interval the system is *lightly damped* and will be very responsive to sudden changes but will probably oscillate back and forth. If you feed back a small proportion of the error over a longer control interval the system is *heavily damped* and will tend to be sluggish and unresponsive with a large time constant.

When we apply these principles to computer system resource management you can see that it is important to average measurements over an appropriate time scale and get the damping factor right. The resource manager needs to respond quickly enough to cope with sudden changes in the workload, such as many simultaneous user logins at the start of a shift, while maintaining a steady flow of resources to all the workloads on the system so that response times are consistent and predictable.

Author's Bio: Adrian Cockcroft

The author of Sun Performance And Tuning, Adrian is an accomplished performance specialist for Sun Microsystems and recognized worldwide as an expert on the subject.