Performance Goals

Performance goals depend on your application's realm and architecture more than anything else. When you have finished gathering requirements, you should determine general performance goals. Depending on your software development process, you might need to adjust these goals as requirements change and new business and operation needs arise. We review some examples of performance goals and guidelines for several archetypal applications, but, as with anything performance-related, these guidelines need to be adapted to your software's domain.

First, here are some examples of statements that *are not* good performance goals:

- The application will remain responsive when many users access the Shopping Cart screen simultaneously.
- The application will not use an unreasonable amount of memory as long as the number of users is reasonable.
- A single database server will serve queries quickly even when there are multiple, fully-loaded application servers.

The main problem with these statements is that they are overly general and subjective. If these are your performance goals, then you are bound to discover they are subject to interpretation and disagreements on their frame-of-reference. A business analyst may consider 100,000 concurrent users a "reasonable" number, whereas a technical team member may know the available hardware cannot support this number of users on a single machine. Conversely, a developer might consider 500 ms response times "responsive," but a user interface expert may consider it laggy and unpolished.

A performance goal, then, is expressed in terms of *quantifiable performance metrics* that can be *measured* by some means of performance testing. The performance goal should also contain some information about its *environment*—general or specific to that performance goal. Some examples of well-specified performance goals include:

- The application will serve every page in the "Important" category within less than 300 ms (not including network roundtrip time), as long as not more than 5,000 users access the Shopping Cart screen concurrently.
- The application will use not more than 4 KB of memory for each idle user session.
- The database server's CPU and disk utilization should not exceed 70%, and it should return responses to queries in the "Common" category within less than 75ms, as long as there are no more than 10 application servers accessing it.

Note These examples assume that the "Important" page category and "Common" query category are well-known terms defined by business analysts or application architects. Guaranteeing performance goals for every nook and cranny in the application is often unreasonable and is not worth the investment in development, hardware, and operational costs.

We now consider some examples of performance goals for typical applications (see Table 1-1). This list is by no means exhaustive and is not intended to be used as a checklist or template for your own performance goals—it is a general frame that establishes differences in performance goals when diverse application types are concerned.

Table 1-1. Examples of Performance Goals for Typical Applications

System Type	Performance Goal	Environment Constraints
External Web Server	Time from request start to full response generated should not exceed 300ms	Not more than 300 concurrently active requests
External Web Server	Virtual memory usage (including cache) should not exceed 1.3GB	Not more than 300 concurrently active requests; not more than 5,000 connected user sessions
Application Server	CPU utilization should not exceed 75%	Not more than 1,000 concurrently active API requests
Application Server	Hard page fault rate should not exceed 2 hard page faults per second	Not more than 1,000 concurrently active API requests
Smart Client Application	Time from double-click on desktop shortcut to main screen showing list of employees should not exceed 1,500ms	–
Smart Client Application	CPU utilization when the application is idle should not exceed 1%	–
Web Page	Time for filtering and sorting the grid of incoming emails should not exceed 750ms, including shuffling animation	Not more than 200 incoming emails displayed on a single screen
Web Page	Memory utilization of cached JavaScript objects for the "chat with representative" windows should not exceed 2.5MB	–
Monitoring Service	Time from failure event to alert generated and dispatched should not exceed 25ms	–
Monitoring Service	Disk I/O operation rate when alerts are not actively generated should be 0	–

Note Characteristics of the hardware on which the application runs are a crucial part of environment constraints. For example, the startup time constraint placed on the smart client application in Table 1-1 may require a solid-state hard drive or a rotating hard drive speed of at least

7200RPM, at least 2GB of system memory, and a 1.2GHz or faster processor with SSE3 instruction support. These environment constraints are not worth repeating for every performance goal, but they are worth remembering during performance testing.

When performance goals are well-defined, the performance testing, load testing, and subsequent optimization process is laid out trivially. Verifying conjectures, such as “with 1,000 concurrently executing API requests there are less than 2 hard page faults per second on the application server,” may often require access to load testing tools and a suitable hardware environment. The next chapter discusses measuring the application to determine whether it meets or exceeds its performance goals once such an environment is established.

Composing well-defined performance goals often requires prior familiarity with performance metrics, which we discuss next.