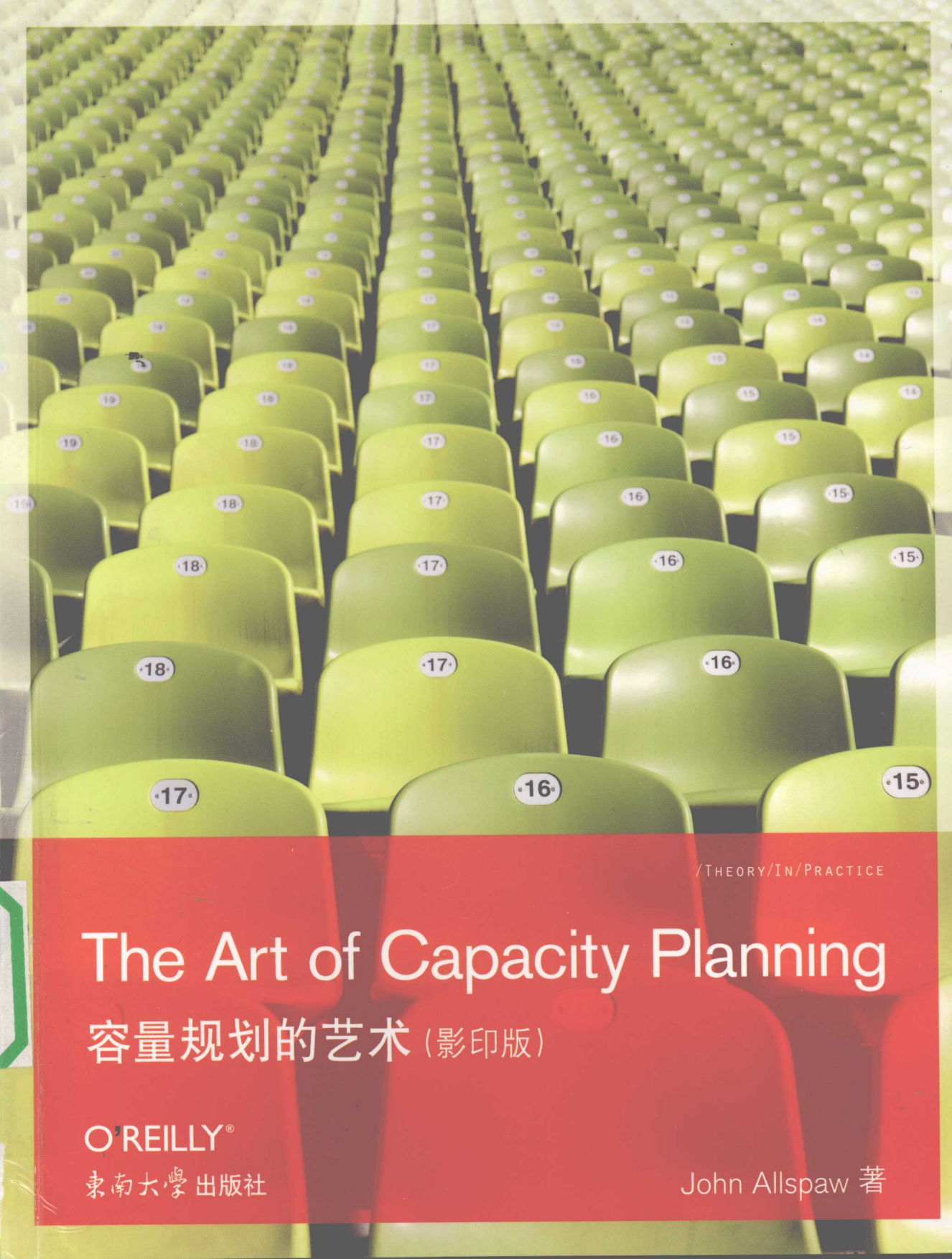


/THEORY/IN/PRACTICE

The Art of Capacity Planning

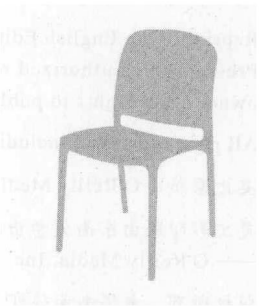
容量规划的艺术（影印版）

O'REILLY®
東南大學出版社

John Allspaw 著

容量规划的艺术(影印版)

The Art of Capacity Planning



John Allspaw

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Taipei • Tokyo

O'Reilly Media, Inc.授权东南大学出版社出版

东南大学出版社

图书在版编目（CIP）数据

容量规划的艺术：英文 /（美）奥斯帕（Allspaw, J.）著. —影印本. —南京：东南大学出版社，2009.5
书名原文：The Art of Capacity Planning
ISBN 978-7-5641-1652-1

Ⅰ. 容… Ⅱ. 奥… Ⅲ. 网站－开发－英文 Ⅳ. TP393.092

中国版本图书馆 CIP 数据核字（2009）第 065398 号

江苏省版权局著作权合同登记
图字：10-2009-139 号

容量规划的艺术（影印版）

出版发行：东南大学出版社
地　　址：南京四牌楼 2 号　　邮编：210096
出 版 人：江　汉
网　　址：http://press.seu.edu.cn
电子邮件：press@seu.edu.cn
印　　刷：扬中市印刷有限公司
开　　本：787 毫米 × 980 毫米　16 开本
印　　张：9.75 印张
字　　数：164 千字
版　　次：2009 年 5 月第 1 版
印　　次：2009 年 5 月第 1 次印刷
书　　号：ISBN 978-7-5641-1652-1
印　　数：1~2200 册
定　　价：29.00 元（册）

本社图书若有印装质量问题，请直接与读者服务部联系。电话（传真）：025-83792328

Preface

SOMEWHERE AROUND 3 A.M. ON JULY 7TH, 2005, MY COWORKER, CAL HENDERSON, AND I WERE FINISHING up some final details before moving all of the traffic for our website, Flickr.com, to its new home: a Yahoo! data center in Texas. The original infrastructure in Vancouver was becoming more and more overloaded, and suffering from serious power and space constraints. Since Yahoo! had just acquired Flickr, it was time to bring new capacity online. It was about an hour after we changed DNS records to point to our shiny new servers that Cal happened to glance at the news. The London subway had just been bombed.

Londoners responded with their camera phones, among other things. Over the next 24 hours, Flickr saw more traffic than ever before, as photos from the disaster were uploaded to the site. News outlets began linking to the photos, and traffic on our new servers went through the roof.

It was not only a great example of citizen journalism, but also an object lesson—sadly, one born of tragedy—in capacity planning. Traffic can be sporadic and unpredictable at times. Had we not moved over to the new data center, Flickr.com wouldn't have been available that day.

Capacity planning has been around since ancient times, with roots in everything from economics to engineering. In a basic sense, capacity planning is resource management. When resources are finite, and come at a cost, you need to do some capacity planning.

When a civil engineering firm designs a new highway system, it's planning for capacity, as is a power company planning to deliver electricity to a metropolitan area. In some ways, their concerns have a lot in common with web operations; many of the basic concepts and concerns can be applied to all three disciplines.

While systems administration has been around since the 1960s, the branch focused on serving websites is still emerging. A large part of web operations is capacity planning and management. Those are *processes*, not *tasks*, and they are composed of many different parts. Although every organization goes about it differently, the basic concepts are the same:

- Ensure proper resources (servers, storage, network, etc.) are available to handle expected and unexpected loads.
- Have a clearly defined procurement and approval system in place.
- Be prepared to justify capital expenditures in support of the business.
- Have a deployment and management system in place to manage the resources once they are deployed.

Why I Wrote This Book

One of my frustrations as an operations engineering manager was not having somewhere to turn to help me figure out how much equipment we'd need to keep running. Existing books on the topic of computer capacity planning were focused on the mathematical *theory* of resource planning, rather than the practical *implementation* of the whole process.

A lot of literature addressed only rudimentary models of website use cases, and lacked specific information or advice. Instead, they tended to offer mathematical models designed to illustrate the principles of queuing theory, which is the foundation of traditional capacity planning. This approach might be mathematically interesting and elegant, but it doesn't help the operations engineer when informed he has a week to prepare for some unknown amount of additional traffic—perhaps due to the launch of a super new feature—or seeing his site dying under the weight of a link from the front page of Yahoo!, Digg, or CNN.

I've found most books on web capacity planning were written with the implied assumption that concepts and processes found in non-web environments, such as manufacturing or industrial engineering, applied uniformly to website environments as well. While some of the theory surrounding such planning may indeed be similar, the practical application of those concepts doesn't map very well to the short timelines of website development.

In most web development settings, it's been my observation that change happens too fast and too often to allow for the detailed and rigorous capacity investigations common to other fields. By the time the operations engineer comes up with the queuing model for his system,

new code is deployed and the usage characteristics have likely already changed dramatically. Or some other technological, social, or real-world event occurs, making all of the modeling and simulations irrelevant.

What I've found to be far more helpful, is talking to colleagues in the industry—people who come up against many of the same scaling and capacity issues. Over time, I've had contact with many different companies, each employing diverse architectures, and each experiencing different problems. But quite often they shared very similar approaches to solutions. My hope is that I can illustrate some of these approaches in this book.

Focus and Topics

This book is not about building complex models and simulations, nor is it about spending time running benchmarks over and over. It's not about mathematical concepts such as Little's Law, Markov chains, or Poisson arrival rates.

What this book is about is *practical* capacity planning and management that can take place in the real world. It's about using real tools, and being able to adapt to changing usage on a website that will (hopefully) grow over time. When you have a flat tire on the highway, you could spend a lot of time trying to figure out the cause, or you can get on with the obvious task of installing the spare and getting back on the road.

This is the approach I'm presenting to capacity planning: adaptive, not theoretical.

Keep in mind a good deal of the information in this book will seem a lot like common sense—this is a good thing. Quite often the simplest approaches to problem solving are the best ones, and capacity planning is no exception.

This book will cover the process of capacity planning for growing websites, including measurement, procurement, and deployment. I'll discuss some of the more popular and proven measurement tools and techniques. Most of these tools run in both LAMP and Windows-based environments. As such, I'll try to keep the discussion as platform-agnostic as possible.

Of course, it's beyond the scope of this book to cover the details of every database, web server, caching server, and storage solution. Instead, I'll use examples of each to illustrate the process and concepts, but this book is not meant to be an implementation guide. The intention is to be as generic as possible when it comes to explaining resource management—it's the process itself we want to emphasize.

For example, a database is used to store data and provide responses to queries. Most of the more popular databases allow for replicating data to other servers, which enhances redundancy, performance, and architectural decisions. It also assists the technical implementation of replication with Postgres, Oracle, or MySQL (a topic for other books). This book covers what replication means in terms of planning capacity and deployment.

Essentially, this book is about measuring, planning, and managing growth for a web application, regardless of the underlying technologies you choose.

Audience for This Book

This book is for systems, storage, database, and network administrators, engineering managers, and of course, capacity planners.

It's intended for anyone who hopes (or perhaps fears) their website will grow like those of Facebook, Flickr, MySpace, Twitter, and others—companies that underwent the trial-by-fire process of scaling up as their usage skyrocketed. The approaches in this text come from real experience with sites where traffic has grown both heavily and rapidly. If you expect the popularity of your site will dramatically increase the amount of traffic you experience, then please read this book.

Organization of the Material

Chapter 1, *Goals, Issues, and Processes in Capacity Planning*, presents the issues that arise over and over on heavily trafficked websites.

Chapter 2, *Setting Goals for Capacity*, illustrates the various concerns involved with planning for the growth of a web application, and how capacity fits into the overall picture of availability and performance.

Chapter 3, *Measurement: Units of Capacity*, discusses capacity measurement and monitoring.

Chapter 4, *Predicting Trends*, explains how to turn measurement data into forecasts, and how trending fits into the overall planning process.

Chapter 5, *Deployment*, discusses concepts related to deployment; automation of installation, configuration, and management.

Appendix A, *Virtualization and Cloud Computing*, discusses where virtualization and cloud services fit into a capacity plan.

Appendix B, *Dealing with Instantaneous Growth*, offers insight into what can be done in capacity crisis situations, and some best practices for dealing with site outages.

Appendix C, *Capacity Tools*, is an annotated list of measurement, installation, configuration, and management tools highlighted throughout the book.

Conventions Used in This Book

The following typographical conventions are used in this book:

Italic
: Indicates new terms, URLs, filenames, Unix utilities, and command-line options.

`Constant width`
: Indicates the contents of files, the output from commands, and generally anything found in programs.

`Constant width bold`

Shows commands or other text that should be typed literally by the user, and parts of code or files highlighted to stand out for discussion.

`Constant width italic`

Shows text that should be replaced with user-supplied values.

Using Code Examples

This book is here to help you get your job done. In general, you may use the code in this book in your programs and documentation. You do not need to contact us for permission unless you're reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing a CD-ROM of examples from O'Reilly books *does* require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount of example code from this book into your product's documentation *does* require permission.

We appreciate, but do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example: "*The Art of Capacity Planning* by John Allspaw. Copyright 2008 Yahoo! Inc., 978-0-596-51857-8."

If you feel your use of code examples falls outside fair use or the permission given above, feel free to contact us at *permissions@oreilly.com*.

We'd Like to Hear from You

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at:

http://www.oreilly.com/catalog/9780596518578

To comment or ask technical questions about this book, send email to:

bookquestions@oreilly.com

For more information about our books, conferences, Resource Centers, and the O'Reilly Network, see our website at:

http://www.oreilly.com

Safari® Books Online

When you see a Safari® Books Online icon on the cover of your favorite technology book, that means the book is available online through the O'Reilly Network Safari Bookshelf.

Safari offers a solution that's better than e-books. It's a virtual library that lets you easily search thousands of top tech books, cut and paste code samples, download chapters, and find quick answers when you need the most accurate, current information. Try it for free at *http://safari.oreilly.com*.

Acknowledgments

It's simply not possible to thank everyone enough in this single, small paragraph, but I will most certainly mention their names. Most of the material in this book was derived from experiences in the trenches, and there are many people who have toughed it out in those trenches alongside me. Peter Norby, Gil Raphaelli, Kevin Collins, Dathan Pattishall, Cal Henderson, Aaron Cope, Paul Hammond, Paul Lloyd, Serguei Mourachov and Chad Dickerson need special thanks, as does Heather Champ and the entire Flickr customer care team. Thank you Flickr development engineering: you all think like operations engineers and for that I am grateful. Thanks to Stewart Butterfield and Caterina Fake for convincing me to join the Flickr team early on. Thanks to David Filo and Hugo Gunnarsen for forcing me to back up my hardware requests with real data. Major thanks go out to Kevin Murphy for providing so much material in the automated deployment chapter. Thanks to Andy Oram and Isabel Kunkle for editing, and special thanks to my good friend Chris Colin for excellent pre-pre-editing advice.

Thanks to Adam Jacob, Matt St. Onge, Jeremy Zawodny, and Theo Schlossnagle for the super tech review.

Much thanks to Matt Mullenweg and Don MacAskill for sharing their cloud infrastructure use cases.

Most important, thanks to my wife, Elizabeth Kairys, for encouraging and supporting me in this insane endeavor. Accomplishing this without her would have been impossible.

O'Reilly Media, Inc.介绍

O'Reilly Media, Inc.是世界上在 UNIX、X、Internet 和其他开放系统图书领域具有领导地位的出版公司，同时是联机出版的先锋。

从最畅销的《The Whole Internet User's Guide & Catalog》（被纽约公共图书馆评为二十世纪最重要的50本书之一）到GNN（最早的Internet门户和商业网站），再到WebSite（第一个桌面PC的Web服务器软件），O'Reilly Media, Inc.一直处于Internet发展的最前沿。

许多书店的反馈表明，O'Reilly Media, Inc.是最稳定的计算机图书出版商 —— 每一本书都一版再版。与大多数计算机图书出版商相比，O'Reilly Media, Inc.具有深厚的计算机专业背景，这使得O'Reilly Media, Inc.形成了一个非常不同于其他出版商的出版方针。O'Reilly Media, Inc.所有的编辑人员以前都是程序员，或者是顶尖级的技术专家。O'Reilly Media, Inc.还有许多固定的作者群体 —— 他们本身是相关领域的技术专家、咨询专家，而现在编写著作，O'Reilly Media, Inc.依靠他们及时地推出图书。因为O'Reilly Media, Inc.紧密地与计算机业界联系着，所以O'Reilly Media, Inc.知道市场上真正需要什么图书。

出版说明

随着计算机技术的成熟和广泛应用，人类正在步入一个技术迅猛发展的新时期。计算机技术的发展给人们的工业生产、商业活动和日常生活都带来了巨大的影响。然而，计算机领域的技术更新速度之快也是众所周知的，为了帮助国内技术人员在第一时间了解国外最新的技术，东南大学出版社和美国O'Reilly Meida, Inc.达成协议，将陆续引进该公司的代表前沿技术或者在某专项领域享有盛名的著作，以影印版或者简体中文版的形式呈献给读者。其中，影印版书籍力求与国外图书“同步”出版，并且“原汁原味”展现给读者。

我们真诚地希望，所引进的书籍能对国内相关行业的技术人员、科研机构的研究人员和高校师生的学习和工作有所帮助，对国内计算机技术的发展有所促进。也衷心期望读者提出宝贵的意见和建议。

最新出版的影印版图书，包括：

- 《应用程序性能测试艺术》（影印版）
- 《算法技术手册》（影印版）
- 《学习 OpenCV》（影印版）
- 《Adobe AIR 1.5 Cookbook》（影印版）
- 《构建嵌入式 Linux 系统 第二版》（影印版）
- 《Web 应用程序通用设计》（影印版）
- 《Web 2.0 策划指南》（影印版）
- 《浅入浅出 C#》（影印版）
- 《容量规划的艺术》（影印版）
- 《RESTful.NET》（影印版）
- 《学习 JavaScript 第二版》（影印版）
- 《SQL 技术手册 第三版》（影印版）

CONTENTS

CHAPTER ONE

Goals, Issues, and Processes in Capacity Planning

THIS CHAPTER IS DESIGNED TO HELP YOU ASSEMBLE AND USE THE WEALTH OF TOOLS AND TECHNIQUES presented in the following chapters. If you do not grasp the concepts introduced in this chapter, reading the remainder of this book will be like setting out on the open ocean without knowing how to use a compass, sextant, or GPS device—you can go around in circles forever.

When you break them down, capacity planning and management—the steps taken to organize the resources your site needs to run properly—are, in fact, simple processes. You begin by asking the question: what performance do you need from your website?

First, define the application's overall load and capacity requirements using *specific* metrics, such as response times, consumable capacity, and peak-driven processing. Peak-driven processing is the workload experienced by your application's resources (web servers, databases, etc.) during peak usage. The process, illustrated in Figure 1-1, involves answering these questions:

1. How well is the current infrastructure working?

   Measure the characteristics of the workload for each piece of the architecture that comprises your applications—web server, database server, network, and so on—and compare them to what you came up with for your performance requirements mentioned above.

2. What do you need in the future to maintain acceptable performance?

   Predict the future based on what you know about past system performance then marry that prediction with what you can afford, and a realistic timeline. Determine what you'll need and *when* you'll need it.

3. How can you install and manage resources after you gather what you need?

   Deploy this new capacity with industry-proven tools and techniques.

4. Rinse, repeat.

   Iterate and calibrate your capacity plan over time.

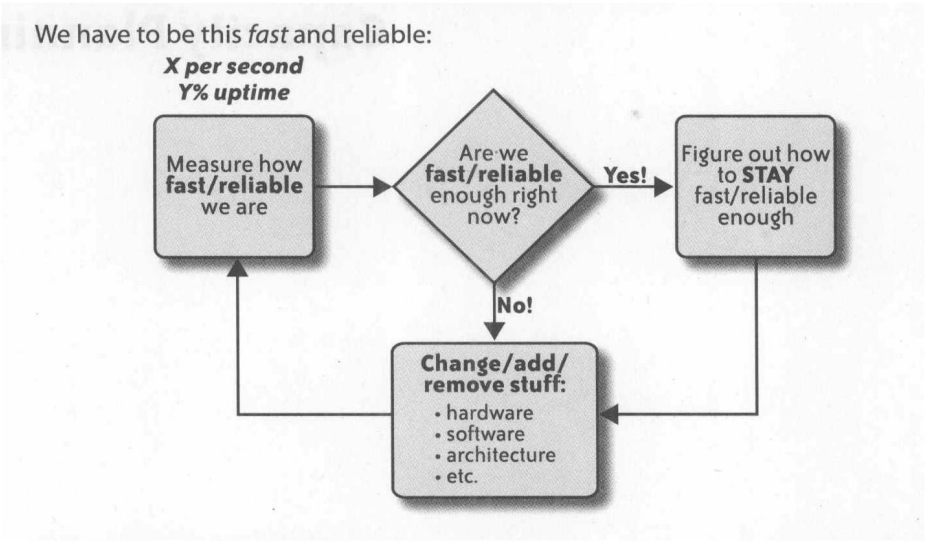


FIGURE 1-1. The process for determining the capacity you need

Your ultimate goal lies between not buying enough hardware and wasting your money on too much hardware.

Let's suppose you're a supermarket manager. One of your tasks is to manage the schedule of cashiers. Your challenge is picking the right number of cashiers working at any moment. Assign too few, and the checkout lines will become long, and the customers irate. Schedule too many working at once, and you're spending more money than necessary. The trick is finding the right balance.

Now, think of the cashiers as servers, and the customers as client browsers. Be aware some cashiers might be better than others, and each day might bring a different amount of customers. Then you need to take into consideration your supermarket is getting more and more popular. A seasoned supermarket manager intuitively knows these variables exist, and attempts to strike a good balance between not frustrating the customers and not paying too many cashiers.

Welcome to the supermarket of web operations.

Quick and Dirty Math

The ideas I've just presented are hardly new, innovative, or complex. Engineering disciplines have always employed back-of-the-envelope calculations; the field of web operations is no different.

Because we're looking to make judgments and predictions on a quickly changing landscape, approximations will be necessary, and it's important to realize what that means in terms of limitations in the process. Being aware of when detail is needed and when it's not is crucial to forecasting budgets and cost models. Unnecessary detail means wasted time. Lacking the proper detail can be fatal.

Predicting When Your Systems Will Fail

Knowing when each piece of your infrastructure will fail (gracefully or not) is crucial to capacity planning. Capacity planning for the web, more often than one would like to admit, looks like the approach shown in Figure 1-2.

FIGURE 1-2. Finding failure points

Including this information as part of your calculations is mandatory, not optional. However, determining the limits of each portion of your site's backend can be tricky. An easily segmented architecture helps you find the limits of your current hardware configurations. You can then use those capacity ceilings as a basis for predicting future growth.

For example, let's assume you have a database server that responds to queries from your frontend web servers. Planning for capacity means knowing the answers to questions such as these:

- Taking into account the specific hardware configuration, how many queries per second (QPS) can the database server manage?
- How many QPS can it serve before performance degradation affects end user experience?

Adjusting for periodic spikes and subtracting some comfortable percentage of headroom (or safety factor, which we'll talk about later) will render a single number with which you can characterize that database configuration vis-à-vis the specific role. Once you find that "red line" metric, you'll know:

- The load that will cause the database to fail, which will allow you to set alert thresholds accordingly.
- What to expect from adding (or removing) similar database servers to the backend.
- When to start sizing another order of new database capacity.

We'll talk more about these last points in the coming chapters. One thing to note is the entire capacity planning process is going to be architecture-specific. This means the calculations you make to predict increasing capacity may have other constraints specific to your particular application.

For example, to spread out the load, a LAMP application might utilize a MySQL server as a master database in which all live data is written and maintained, and use a second, replicated slave database for read-only database operations. Adding more slave databases to scale the read-only traffic is generally an appropriate technique, but many large websites (including Flickr) have been forthright about their experiences with this approach, and the limits they've encountered. There is a limit to how many read-only slave databases you can add before you begin to see diminishing returns as the rate and volume of changes to data on the master database may be more than the replicated slaves can sustain, no matter how many you add. This is just one example where your architecture can have a large effect on your ability to add capacity.

Expanding database-driven web applications might take different paths in their evolution toward scalable maturity. Some may choose to federate data across many master databases. They may split up the database into their own clusters, or choose to cache data in a variety of methods to reduce load on their database layer. Yet others may take a hybrid approach, using all of these methods of scaling. This book is not intended to be an advice column on database scaling, it's meant to serve as a guide by which you can come up with your own planning and measurement process—one that is right for your environment.

Make Your System Stats Tell Stories

Server statistics paint only part of the picture of your system's health. Unless they can be tied to actual site metrics, server statistics don't mean very much in terms of characterizing your usage. And this is something you'll need to know in order to track how capacity will change over time.

For example, knowing your web servers are processing *X* requests per second is handy, but it's also good to know what those *X* requests per second actually mean in terms of your users. Maybe *X* requests per second represents *Y* number of users employing the site simultaneously.

It would be even better to know that of those *Y* simultaneous users, *A* percent are uploading photos, *B* percent are making comments on a heated forum topic, and *C* percent are poking randomly around the site while waiting for the pizza guy to arrive. Measuring those user metrics over time is a first step. Comparing and graphing the web server hits-per-second against those user interaction metrics will ultimately yield some of the cost of