IBM PRESS

Developing and Hosting Applications on the Cloud

Alex Amies, Harm Sluiman,
Qiang Guo Tong, Guo Ning Liu

Developing and Hosting Applications on the Cloud

This page intentionally left blank

Developing and Hosting Applications on the Cloud

Alex Amies, Harm Sluiman, Qiang Guo Tong, Guo Ning Liu

IBM Press
Pearson plc
Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Cape Town • Sydney • Tokyo • Singapore • Mexico City

Ibmpressbooks.com

IBM Press Program Managers: Steven M. Stansel, Ellice Uffer
Cover design: IBM Corporation

Editor-in-Chief: Dave Dusthimer
Marketing Manager: Stephane Nakib
Acquisitions Editor: Mary Beth Ray
Publicist: Heather Fox
Managing Editor: Kristy Hart
Designer: Alan Clements
Project Editor: Betsy Harris
Copy Editor: Krista Hansing Editorial Services, Inc.
Senior Indexer: Cheryl Lenser
Compositor: Nonie Ratcliff
Proofreader: Language Logistics, LLC
Manufacturing Buyer: Dan Uhrig

Published by Pearson plc
Publishing as IBM Press

IBM Press offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact

U. S. Corporate and Government Sales
1-800-382-3419
corpsales@pearsontechgroup.com.

For sales outside the United States, please contact

International Sales
international@pearsoned.com.

ISBN-13: 978-0-13-306684-5
ISBN-10: 0-13-306684-3

This book is dedicated to all the members of the IBM® SmartCloud™ Enterprise development team whose hard work and professionalism has made this large and challenging project a reality.

Contents

Part II: Developing Cloud Applications

Preface

We are writing this book to share our experience over the past several years of developing the IBM SmartCloud™ Enterprise. We hope that readers will not just learn more about that cloud, but also be inspired to build solutions using it or other clouds as a platform. We hope that people using other clouds will benefit from this book as well.

Acknowledgments

Thanks to many dedicated colleagues at IBM who have worked on IBM SmartCloud Enterprise and other related products and projects. In particular, thanks to all the customers and people inside IBM who are using the IBM SmartCloud Enterprise, for their feedback and questions, especially the Rational® team. We gained a great deal of insight about the use of the cloud from these questions and discussions, and it forced us to look at the cloud from an outside-in point of view.

Thanks also to the entire IBM SmartCloud development team for its hard work and dedication in building this wonderful platform, working through unreasonable schedules and difficult technical problems in the process.

Thanks to these specific people who helped with suggestions and review:

- Chris Roach, Program Manager, Cloud Technology, IBM
- Doug Davis, Senior Technical Staff Member, Web Services and Cloud Standards, IBM
- Dikran Meliksetian, Senior Technical Staff Member, Integrated Technology Delivery, IBM
- Jamshid Vayghan, PhD, IBM Distinguished Engineer and Director, CTO Sales Transformation, IBM
- Michael Behrendt, Cloud Computing Architect, IBM
- Prasad Saripalli, PhD, Principal Architect, IBM Cloud Engineering
- Scott Peddle, Advisory Software Engineer, IBM Global Technology Services®
- Shane Weeden, Senior Software Engineer and IBM Tivoli® Federated Identity Manager development lead, who helped us understand OAuth and FIM.
- Stefan Pappe, IBM Fellow, Cloud Services Specialty Area, IBM

This was a personal effort by the authors and is not representative of IBM or its views. IBM did not participate in and does not endorse this work. However, the authors thank IBM for access to the IBM SmartCloud Enterprise system and the opportunity to work on such a challenging and satisfying project.

About the Authors

Alex Amies is a Senior Software Engineer with IBM and an architect on the IBM SmartCloud Enterprise development team.

Harm Sluiman is a Distinguished Engineer with IBM and the technical lead for SmartCloud Enterprise.

Qiang Guo Tong is an Advisory Software Engineer with IBM and one of the lead developers for SmartCloud Enterprise.

Guo Ning Liu is a Staff Software Engineer with IBM and worked on development of the public APIs, provisioning services, and security for SmartCloud Enterprise.

This page intentionally left blank

Introduction

The goal of this book is to help enterprises develop and operate services on the cloud. In particular, we hope that independent software vendors will be inspired to build value-add services on public clouds. Additionally, we hope that developers of applications who make heavy use of Infrastructure as a Service (IaaS), such as developers of Platform as a Service, Software as a Service, and Business as a Service, will find this book useful. The target audience is developers who use cloud-management application programming, architects who are planning projects, and others who want to automate the management of IT infrastructure. The book is intermediate in level but still offers a broad overview of the entire topic of IaaS clouds and aims to give a basic background on most of the prerequisites needed to understand the topics discussed.

The book makes special reference to the IBM SmartCloud Enterprise. However, the principles are general and are useful to anyone planning to automate the management of IT infrastructure using cloud technology. In contrast to technical product documentation, the book tells a story about why you might want to use the technologies described and includes sufficient background material to enable you to build the cloud applications described without having to consult numerous external references. The references are listed as suggestions for further reading, not as prerequisites to understanding the information presented.

Today cloud computing is bringing application development, business, and system operations closer together. This means that software developers need to better understand business process and system operations. It also means that business stakeholders and operations staff have to consume more software. The promise of cloud computing is that centralization, standardization, and automation will simplify the user experience and reduce costs. However, fully achieving these benefits requires a new mindset. The scope of this book is intentionally broad, to cover these aspects of application development and operation. In addition, the book is quite practical, providing numerous code examples and demonstrating system utilities for deployment, security, and maintenance.

The plan of the book runs from simple to more challenging. We hope that it gives application developers an idea of the different possible applications that can be developed. As a result, we look at some adjacent areas and related standards. Many of the topics discussed are not new; however, they are strategic to cloud computing and, when necessary, we review them so that readers do not need to seek background information elsewhere. We also will demonstrate several relatively older technologies, such as Linux services and storage systems, that are finding new uses in cloud computing.

Above all, this book emphasizes problem solving through cloud computing. At times you might face a simple problem and need to know only a simple trick. Other times you might be on the wrong track and need some background information to get oriented. Still other times, you might face a bigger problem and need direction and a plan. You will find all of these in this book.

We provide a short description of the overall structure of a cloud here, to give the reader an intuitive feel for what a cloud is. Most readers will have some experience with virtualization. Using virtualization tools, you can create a virtual machine with the operating system install software, make your own customizations to the virtual machine, use it to do some work, save a snapshot to a CD, and then shut down the virtual machine. An Infrastructure as a Service (IaaS) cloud takes this to another level and offers additional convenience and capability.

Using an IaaS cloud you can create the virtual machine without owning any of the virtualization software yourself. Instead, you can access the tools for creating and managing the virtual machine via a web portal. You do not even need the install image of the operating system; you can use a virtual machine image that someone else created previously. (Of course, that someone else probably has a lot of experience in creating virtual machine images, and the image most likely went through a quality process before it was added to the image catalog.) You might not even have to install any software on the virtual machine or make customizations yourself; someone else might have already created something you can leverage. You also do not need to own any of the compute resources to run the virtual machine yourself: Everything is inside a cloud data center. You can access the virtual machine using secure shell or a remote graphical user interface tool, such as Virtual Network Computing (VNC) or Windows® Remote Desktop. When you are finished, you do not need to save the virtual machine to a CD; you can save it to the cloud storage system. Although you do not have to own any of the infrastructure to do all this yourself, you still have to pay for it in some way. The cloud provider handles that automatically as well, based on the quantity of resources that you have used. This is the cloud pay-as-you-go concept.

The cloud provider has to invest in a lot of infrastructure to support this. Figure I.1 shows a high-level overview of an Infrastructure as a Service cloud.

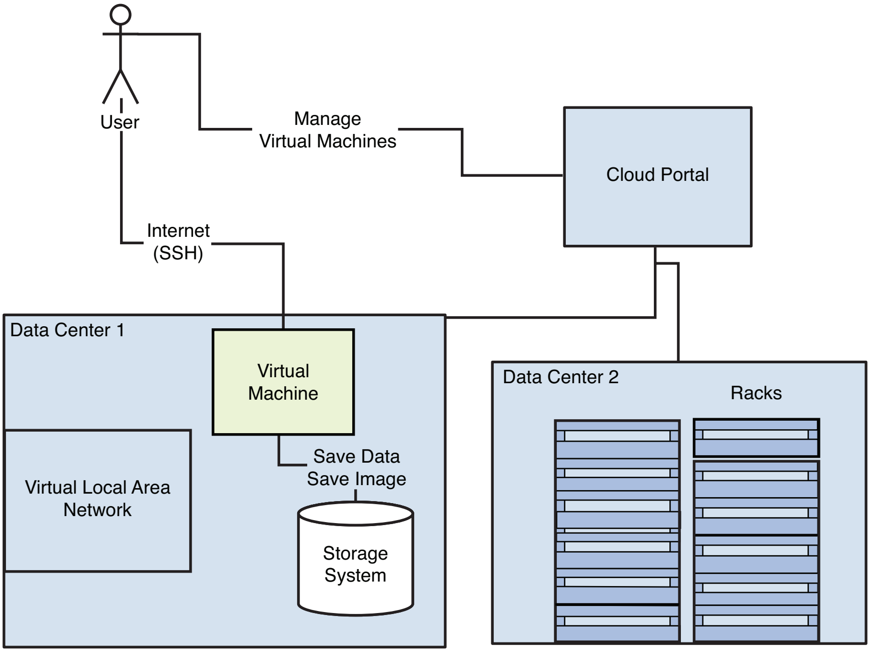

Figure I.1 Conceptual diagram of an Infrastructure as a Service cloud

The figure shows two cloud data centers with rack-based servers. Each server has many CPUs and can support multiple virtual machines of different sizes. This is a major investment for the cloud provider and the first advantage that a cloud user might think of, compared to in-house virtualization: With a cloud, you can have as many computing resources as you need for as short or long of a duration as desired; you are not limited by the computing capacity of your local facilities. We refer to this characteristic as elasticity. You also connect to the cloud via the Internet, which is convenient if you are hosting a web site but requires you to consider security. This is where the virtual local area network shown in Figure I.1 can help you.

The cloud also provides a network storage system, which you can use for storing either virtual machine images or data. Although the cost of ownership of network storage systems is declining, owning your own network storage system is still expensive and affordable to usually only medium to large companies. Blocks of the storage system can be carved off and made available as block storage volumes that can attach to virtual machines. Another aspect of data storage and backup in cloud environments is that multiple data centers are available for making redundant copies of data and providing high availability for mission-critical applications.

The cloud portal provides all this self-service as an additional aspect of cloud computing, which is a great savings for enterprises. No need to ask an administrator every time you need a new server, IP address, or additional storage—the cloud portal provides a control panel that gives you an overview of resources that end users can manage on demand. Not only are fewer administrators needed, but the consumers of the resources also have access to the resources more quickly. This results in both a savings in capital and staff needed and a more agile business.

Another aspect of cloud computing that is immediately apparent to independent software vendors is that public clouds provide a platform for a marketplace. Visibility of resources and services on the cloud can be categorized at three levels: private, shared, and public. Publicly visible resources, especially virtual machine images, provide an opportunity for independent software vendors to sell services.

Terminology

This section gives some of the basic terminology for cloud computing, to give readers a common resource for the terms used. Upcoming chapters explain the terminology in more detail for specialized aspects of cloud computing.

instance—A virtual machine instance. Sometimes referred to as a node.

image—A template for creating a virtual machine. A large file that saves the state of a virtual machine so that a new virtual machine can be created from it.

virtual local area network (VLAN)—An abstraction of the traditional local area network that does not depend on physical connections. A VLAN usually is a resource that a cloud user uses and is isolated from the Internet.

public cloud—A cloud from which multiple enterprises or individuals can consume services. IBM SmartCloud Enterprise is a public cloud that allows only enterprises as customers.

private cloud—A cloud that an enterprise operates for its sole use.

multitenant—A service that multiple tenants share. In this context, a tenant is usually an enterprise, and separation of the tenants' resources is implied.

compute size—The number of virtual CPUs, amount of memory, and hard disks dedicated to a virtual machine.

elasticity—The capability to scale resources on demand, such as dynamically adding virtual machines or IP addresses.

Organization of the Book

The book is divided in to three parts.

Background Information

The first part of the book covers background knowledge on cloud computing. It begins with Chapter 1, "Infrastructure as a Service Cloud Concepts," and covers the basic reasons for using cloud computing by looking at some use cases. This chapter then explains some basic cloud concepts and the resource model of the entities we are managing. The chapter provides a context and language for the chapters that follow. It is followed by a description of how to set up development environments in the cloud. To this point, all the concepts apply equally to any Infrastructure as a Service cloud.

Developing Cloud Applications

The second part of the book describes how to use cloud tools and develop simple cloud applications, and it explores potential cloud application areas. It includes chapters on developing on the cloud, developing with the IBM SmartCloud Enterprise, leveraging standards, and creating cloud services and applications. The chapters also describe the command-line toolkit, Java, and REST APIs for managing resources specifically for IBM SmartCloud Enterprise, as well as provide a number of code examples. In addition, this part discusses standards that relate to cloud computing and some open source projects and covers how to leverage those standards to interoperate between clouds. Following that, this part describes several application areas that are becoming important in cloud computing, such as image customization, network services, software installation and management, storage, and remote desktops.

Exploring Hosting Cloud Applications

The third section of the book discusses hosting applications on the cloud. This includes chapters on security; monitoring, performance, and availability; and operations and maintenance on the cloud. First, we provide an overview of relevant security areas and techniques for hardening applications. We then discuss monitoring, performance, and availability. Finally, we discuss business support systems and maintenance.

The book uses a scenario to illustrate and tie together the different concepts discussed. Throughout, we focus on a hypothetical company called IoT Data that provides a data storage service for Internet-enabled devices.

Disclaimer

Any recommended solutions contained in this book are not guaranteed. Warranty is not implied for any source code. All source code should be understood as sample for illustrative purposes only. IBM does not support or endorse any information in this book.

This page intentionally left blank

CHAPTER 1

Infrastructure as a Service Cloud Concepts

This chapter discusses Infrastructure as a Service (IaaS) concepts with the goal of giving cloud application developers background knowledge and helping them explore why they might want to use cloud computing.

The United States National Institute for Standards and Technology (NIST) defines cloud computing as a model for convenient and rapid network access to a shared pool of computing resources that can be provisioned with minimal management effort [Mell and Grance, 2009]. According to this definition, cloud computing has five essential characteristics:

- On-demand self-service
- Broad network access
- Multitenancy
- Rapid elasticity
- Measured service (pay as you go)

NIST also describes four deployment models:

- **Private cloud**—An organization operates a cloud for its own use. A private cloud can be either on-site at an enterprise's own premises or off-site at the cloud provider's location, with network connectivity and isolation from the outside using a virtual private network (VPN). A private cloud does not need multitenant capability, even though this is one of the five essential characteristics listed earlier.
- **Community cloud**—Several organizations use the cloud. For example, several government organizations might share both goals and resources.
- **Public cloud**—A cloud provider offers cloud services to the public-at-large.
- **Hybrid cloud**—Two or more clouds are federated by some enabling technology.

The content in this book applies to each of these models. However, some of the technologies are more applicable to one of more of the different types of clouds. For private clouds, you will need to operate the cloud itself more independently, so you need a deeper background in virtualization technologies. Public clouds tend to be large in scale, enabling independent software vendors (ISVs) and others to develop innovative services and solutions. To do this successfully, ISVs need to understand how to develop reusable cloud services. Interoperability is important in hybrid clouds, and you might find yourself focusing on standards. Likewise, collaboration is important in community clouds, so open source projects and collaboration techniques might be important.

Workloads

The term *workload* in the context of cloud computing is an abstraction of the use to which cloud consumers put their virtual machines on the cloud. For example, a desktop workload might be supporting a number of users logging on to interactive desktop sessions. An SAP workload might be a system of virtual machines working together to support an enterprise's SAP system. Workloads are a key characteristic differentiating the requirements for cloud computing. Different workloads have different characteristics in terms of computing capacity, variability of load, network needs, back-up services, security needs, network bandwidth needs, and other quality-of-service metrics. At a high level, cloud workloads are divided into three groups: server centric, client centric, and mobile centric. Table 1.1 summarizes the common types of cloud workloads.

Table 1.1 Common Workloads in Cloud Computing

Workload	Description and Examples	Key Quality-of-Service Metrics
Server Centric		
Web sites	Freely available web sites for social networking, informational web sites large number of users	Large amounts of storage, high network bandwidth,
Scientific computing	Bioinformatics, atmospheric modeling, other numerical computations	Computing capacity
Enterprise software	Email servers, SAP, enterprise content management	Security, high availability, customer support
Performance testing	Simulation of large workloads to test the performance characteristics of software under development	Computing capacity
Online financial services	Online banking, insurance	Security, high availability, Internet accessibility

Workload	Description and Examples	Key Quality-of-Service Metrics
E-commerce	Retail shopping	Variable computing load, especially at holiday times
Core financial services	Banking and insurance systems	Security, high availability
Storage and backup services	General data storage and backup	Large amounts of reliable storage
Client Centric		
Productivity applications	Users logging on interactively for email, word processing, and so on	Network bandwidth and latency, data backup, security
Development and testing	Software development of web applications with Rational Software Architect, Microsoft® Visual Studio, and so on	User self-service, flexibility, rich set of infrastructure services
Graphics intensive	Animation and visualization software applications	Network bandwidth and latency, data backup
Rich Internet applications	Web applications with a large amount of JavaScript	
Mobile Centric		
Mobile services	Servers to support rich mobile applications	High availability

It is apparent from Table 1.1 that different workloads are appropriate for different types of clouds. For example, free online social networking web sites need many virtual machines to support many users and save large numbers of media files. Public cloud computing is ideal for supporting online social networking sites. Security and high availability is a top consideration for core financial services that need to be isolated from the Internet. The data integrity provided by a relational database is important for financial applications, to ensure that financial transactions are accounted for accurately. However, social networking web sites often use NoSQL data stores that do not provide full relational integrity.

The workloads can be refined further. For example, desktop needs are different for a handful of developers than they are for a large number of general employees. The developers might use a Linux desktop and set up everything themselves. The general employees might use a standard desktop image maintained from a central point. Support is also important for the general employees, who do not have the expertise to troubleshoot and reinstall, if needed, as developers do.

The paper *MADMAC: Multiple Attribute Decision Methodology for Adoption of Clouds* [Saripalli and Pingali, 2011] discusses in detail cloud workloads and decision making for enterprise cloud adoption.

Use Cases

This section explores some of the use cases driving cloud computing. Cloud computing offers many advantages that are important for individual use cases. Infrastructure virtualization also opens up new possibilities and IT assets that traditional computing does not use. Finally, operating in a public Internet environment offers new collaboration possibilities while also introducing security challenges. See "Use Cases and Interactions for Managing Clouds" [Distributed Management Task Force, 2010] for more detail on use cases.

Actors

A number of actors collaborate together in cloud use cases. Consider this basic list.

Cloud service developer—Develops software and other assets for consumption on the cloud.

Cloud service consumer—Requests cloud resources and approves business expenditures. Cloud service consumers can include users, administrators, and business managers.

Cloud provider—Provides a cloud service to consumers.

Web Site Hosting

Operating a web site that requires database access, supports considerable traffic, and possibly connects to enterprise systems requires complete control of one or more servers, to guarantee responsiveness to user requests. Servers supporting the web site must be hosted in a data center with access from the public Internet. Traditionally, this has been achieved by renting space for physical servers in a hosting center operated by a network provider far from the enterprise's internal systems. With cloud computing, this can now be done by renting a virtual machine in a cloud hosting center. The web site can make use of open source software, such as Apache HTTP Server, MySQL, and PHP; the so-called LAMP stack; or a Java™ stack, all of which is readily available. Alternatively, enterprises might prefer to use commercially supported software, such as WebSphere® Application Server and DB2®, on either Linux® or Windows operating systems. All these options are possible in IaaS clouds and, in particular, in the IBM SmartCloud Enterprise.

Figure 1.1 shows a use case diagram for this scenario.

When building the web site, the developer needs to create a virtual machine instance that will host the web and application servers needed. The developer can save an instance to an image when the development of the site reaches a certain point or just for back-up purposes. Usually an administrator does not want to use an instance that a developer created. However, the administrator needs to know the hosting requirements in detail and might use an image that the developer saved or scripts that a developer created, as a starting point. In the process of maintaining the web site, an administrator might need to add storage and clone storage for back-up purposes. After cloning, the administrator might want to copy the data to some other location, so having it offline from the production web site would be an advantage. From the users' perspective, users will be unaware that the web site is hosted in the cloud.

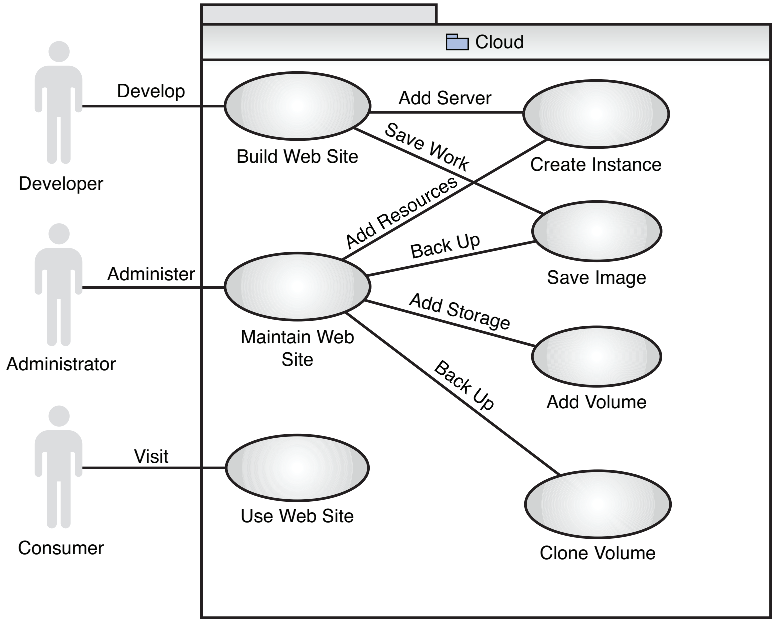

Figure 1.1 Use case diagram for hosting a web site on the cloud

The activities of the developer and administrator can be accomplished via a console with a graphical user interface, such as the one the IBM SmartCloud Enterprise provides. However, as time passes, many regular cloud users will automate with scripts. Command-line tools are ideal for these power users because they execute much faster than a user can click a mouse and navigate pages. Many power users have cheat sheets for common operations, such as installing software and patches, that they can retrieve and edit as needed. They can save scripts for creating instances, saving images, and performing other operations along with the rest of the script collection.

The main advantage of using the cloud for this use case is that renting a virtual machine in a location where it is accessible from the Internet is considerably cheaper than placing physical machines in a data center accessible from the Internet. Other cloud advantages also apply to this use case, including the rapid ability to substitute in a new virtual machine for a server experiencing a hardware fault.

Short-Term Peak Workloads

In the retail industry, workloads come in short peaks at certain times of the year (notably, at Christmas) or coincide with advertising campaigns. Quickly adding capacity during these times is important. With their elastic ability to add servers as desired, clouds are ideal in this situation. Monitoring is important because user traffic varies from year to year based on economic conditions and other factors that make predicting the workload difficult. The IBM SmartCloud Enterprise includes an IBM Tivoli Monitoring image in the catalog that can be helpful. Along with other images in the catalog, it can be rented for as long as needed, and no installation is necessary. Figure 1.2 shows a use case diagram for this scenario.

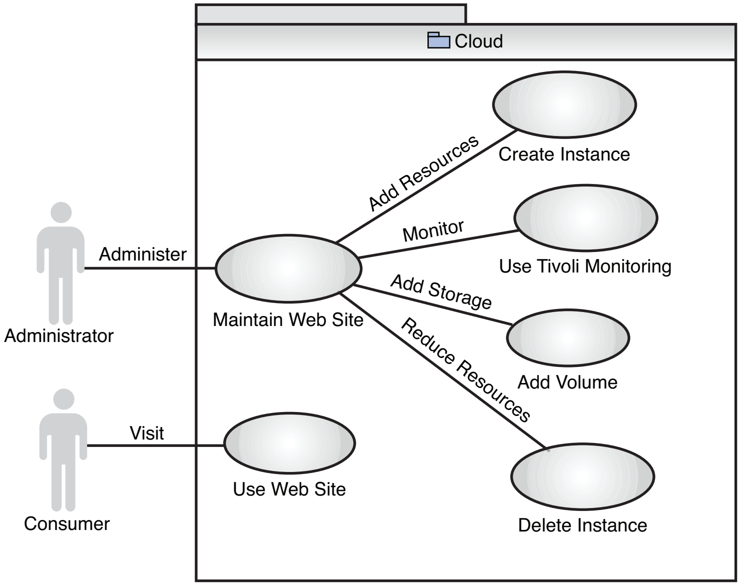

Figure 1.2 Use case diagram for monitoring peak workloads

As in the previous use case, all actions required to do this can be done in the console graphical user interface. However, scripts avoid repetitive work and save administrators time.

The main advantage of the cloud in this use case is its elastic scalability.

Proof-of-Concept

Enterprises usually do proof-of-concept or pilot studies of new technologies before committing to use them. External IT consultants are often invited to do these proof-of-concepts. The consultants are typically under a lot of pressure to deliver a large quantity of computing capacity in a short period of time. If they do not have prior experience in this area, they generally have little hope of succeeding. Assets that they can take from job to job are critical. The cloud can make this easier by allowing saved images to be reused directly and to allow consultants and enterprise users to easily share the same network space. This solution is a better one than requiring the consultant to transport physical machines, install everything on her or his laptop, or install all the software on-site at the enterprise in a short period of time.

Figure 1.3 shows a use case diagram for this scenario.

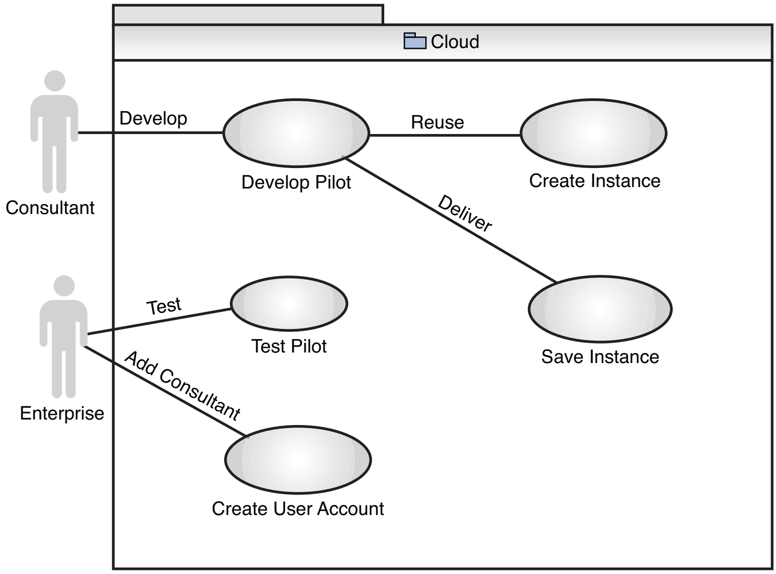

Figure 1.3 Use case diagram for a proof-of-concept on the cloud

Working in a public cloud environment with support for user administration is critical here, to allow the enterprise to add an account for the consultant. Alternatively, the consultant could use his or her account space and simply allow access via a network protocol such as HTTP. If the enterprise likes the proof-of-concept, it might want to use it long term. It can move it to the company's private network by saving an image and starting up an instance on its virtualization LAN. Table 1.2 compares a traditional proof-of-concept and a proof-of-concept on the cloud.

Table 1.2 Comparison of Traditional and Cloud Environments for a Proof-of-Concept

Traditional	Cloud
The consultant travels to the customer site.	The consultant works over the Internet.
The customer gives the consultant access to the enterprise network, subject to an approval workflow.	The customer gives the consultant access to the cloud with account or specific virtual machines with cryptographic keys.
Customer procures hardware for the pilot.	Customer creates an instance with the self-service interface.
The consultant works independently.	The consultant pulls in experts for high availability, performance, security, and so on for a few hours, as needed.

Table 1.2 Comparison of Traditional and Cloud Environments for a Proof-of-Concept (continued)

Traditional	Cloud
The consultant cannot connect his or her laptop to the enterprise network and instead must use only tools that the customer makes available.	The customer can use her or his favorite application lifecycle management tools on a laptop or available on the cloud.
The consultant installs everything from scratch.	The consultant starts up instances from prebuilt images.
The server is repurposed after completion.	Server instances are saved as images, and running instances are deleted.

The cloud enables a different set of deliverables for proof-of-concept, pilot, beta programs, and consulting projects. In traditional environments, enterprise network constraints (especially security issues) often require consultants to work with unfamiliar tools. This results in written reports documenting deployment steps and best practices that customers cannot easily consume. In other situations, consultants are left in a permanent support position long after the project has "finished." The cloud enables a different set of deliverables, including virtual machine images, deployment topology models, and software bundles, as shown Table 1.3.

Table 1.3 Comparison of Traditional and Cloud Project Artifacts

Traditional	Cloud
Software installation program (time consuming to develop)	Virtual machine image (capturing an instance with the click of a button)
Written reports summarizing deployment steps	Deployment topology models, automation scripts
User documentation written from scratch	Documentation reused from standard images
Configuration files in miscellaneous locations	Asset added to cloud catalog
Difficult support process	Support via remote access to cloud

The primary advantages of the cloud for this use case are elastic scalability, access from the Internet, and the capability to save and reuse projects assets.

Extra Capacity

In this scenario, the IT department runs out of computing resources, delaying in-house projects. The department rents resources on the cloud to meet the shortfall. A virtual private network is used to connect to a private virtual local area network (VLAN) in the cloud to the enterprise network.

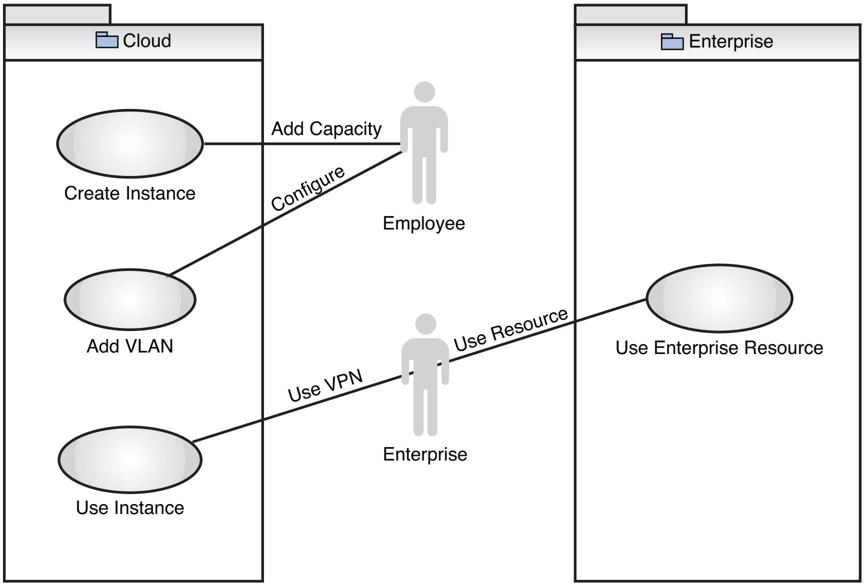

Figure 1.4 Use case diagram for adding extra capacity for enterprise IT infrastructure

Open Source/Enterprise Collaboration

Recently, enterprises have embraced the idea of open source. However, this is best done in a controlled way. An organization might be unwilling to host an open source project on SourceForge or Apache but might want to use open source in a more controlled way. By hosting the project itself on the cloud, the enterprise maintains complete control over the project while still gaining the advantages of an open source model.

Outside contributors can make use of these advantages:

- Be given user accounts without granting access to the enterprise's internal IT systems
- Use a common set of development tools hosted on the cloud

Storage System for Security Videos

Some application domains consume huge amounts of data. Video files are one example. In addition to the files themselves, a management application must allow the videos to be accessed and store additional metadata about them. Hadoop, a freely available open source distributed file system capable of storing huge amounts of data, might fulfill the storage needs of such a security video management and access system. IaaS clouds are an ideal platform for hosting Hadoop and being able to add nodes to the cluster on demand.

Business Scenario: IoT Data Hosting Provider

To tie together the information presented in this book, this section describes how it can be used in a business scenario. In this situation, the company IoT Data provides a hosting service for Internet-connected devices to store data. IoT Data's business services include the following:

- Registering devices
- Storing data from a device using a REST web service
- Conducting HTML and programmatic searches of the data
- Sharing the data in public, community, and commercial modes

IoT Data charges customers by gibibytes (GiB) of data stored and 10% of any data sold. For large customers, IoT Data also provide the entire suite of software for private hosting on the cloud itself. In this case, the changes are per virtual machine hour and depend on the size of the virtual machine (in addition to the per-GiB charge). A diagram showing the main actors and use cases for IoT Data is shown next.

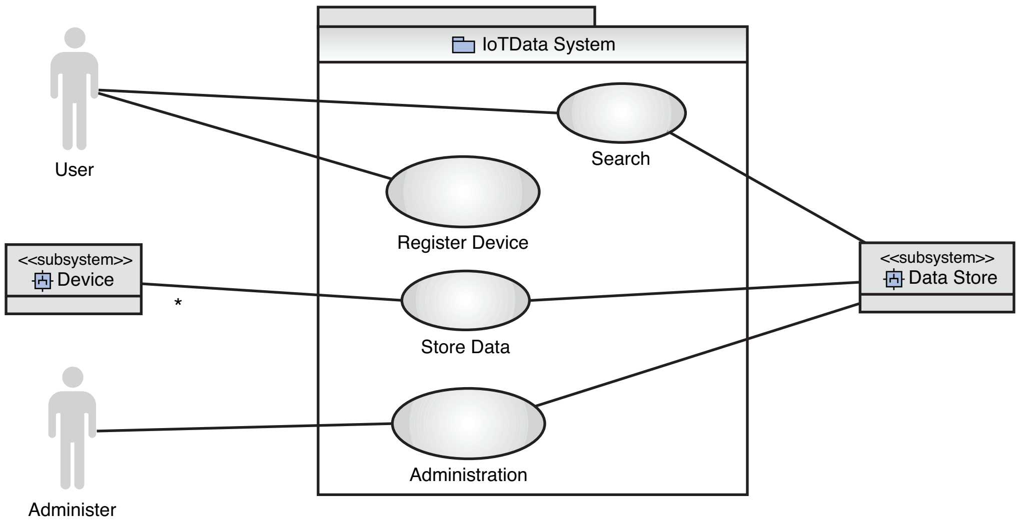

Figure 1.5 IoT Data use case diagram

IoT Data does not have a large budget to hire employees, so as much work as possible has to be automated. IoT Data also cannot afford to buy servers, so it needs a pay-as-you-go model, such as a public cloud provides. In addition, the company has few resources to develop its own software and thus must leverage as much as possible from the cloud provider. This book explains how different technologies can meet IoT Data's business needs (however, we do not actually write the code for doing so).

Virtualization

We briefly discuss virtualization, with the goal of providing a foundation for discussing IaaS clouds and the resource model. The term *virtualization* can apply to a computer (a virtual machine and the resources it uses), storage, network resources, desktops, or other entities. Virtualization of hardware resources and operating systems dates back the 1960s, with IBM mainframes, and was later used on AIX® and other UNIX® platforms. It has been a powerful tool for these platforms for many years. In 1999, VMWare introduced virtualization for low-cost Intel® x-series hardware, based on the research of its founders at Stanford University. This made the practice of virtualization more widespread.

A hypervisor, or virtual machine manager, is a software module that manages virtual machines. The hypervisor resides on the host system on which the virtual machines run. The relationship of the hypervisor to the host operating system and to the virtual machine is one of the key distinguishing characteristics of the different virtualization systems.

Major virtualization systems for x86 hardware include these:

- VMWare, a broad range of virtualization products for x86
- Xen, an open source virtualization system with commercial support from Citrix
- Windows Hyper-V, introduced by Microsoft in Windows Server 2008
- Kernel Virtualization Machine (KVM), a part of the Linux kernel since version 2.6.2

Virtualization became widespread in the early 2000s, several years before the rise of cloud computing. Virtualization offers many practical benefits, including the following:

- The ease of setting up new systems. New systems do not need to installed using installation media.
- No need to buy new hardware to simulate various system environments for debugging and support.
- The capability to recover quickly from system corruption.
- The ease of relocating and migrating systems. For example, a move to a more powerful machine can simply be a matter of taking a snapshot of a virtual machine and starting up a new virtual machine based on that snapshot.
- The ease of remote management. Physical access to data centers is tightly controlled these days. The use of virtual machines greatly reduces the need for physical access.
- The capability to run multiple operating systems simultaneously on one server.

In virtualization of hardware and operating systems, we refer to the *guest* system as the system being virtualized. The system the guest runs on is called the *host*, which uses a *hypervisor* to managing scheduling and system resources, such as memory. Several types of virtualization exist: full virtualization, partial virtualization, and paravirtualization.

Full virtualization is complete simulation of the hardware. Full virtualization is simulating to emulate. In emulation, an emulated system is completely independent of the hardware. The Android smart phone emulator and QEMU in unaccelerated mode are examples of system emulation. Full virtualization differs from emulation in that the virtual system is designed to run on the same hardware architecture as the host system. This enables the instructions of the virtual machine to run directly on the hardware, greatly increasing performance. In full virtualization, no software is needed to simulate the hardware architecture. Figure 1.6 gives a schematic diagram of full virtualization.

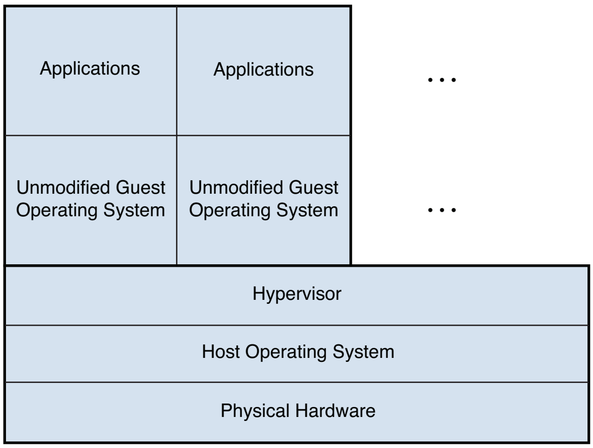

Figure 1.6 Schematic diagram of full virtualization

One of the key characteristics of full virtualization is that an unmodified guest operating system can run on a virtual machine. However, for performance reasons, some modifications are often made. Intel and AMD introduced enhancements to CPUs to allow this: the Intel VT (Virtual Technology) and AMD-V features introduced in 2005. These features support modifications of the guest operating system instructions through variations in their translation to run on the hardware. The Intel VT-x (32-bit processors) and VT-i (IA64 architecture) introduced two new operation levels for the processor, to be used by hypervisors to allow the guest operating systems to run unmodified. Intel also developed a VT-d feature for direct IO, to enable devices to be safely assigned to guest operating systems. VT-d also supports direct memory access (DMA) remapping, which prevents a direct memory access from escaping the bounds of a virtual machine. AMD has a similar set of modifications, although implemented somewhat differently.

Figure 1.6 shows the hypervisor running on top of the host operating system. However, this is not necessary for some hypervisors, which can run in "bare-metal" mode, installed directly on the hardware. Performance increases by eliminating the need for a host operating system.

VMWare Workstation and the IBM System z® Virtual Machine are examples of full virtualization products. VMWare has a wide range of virtualization products for x86 systems. The ESX Server can run in bare-metal mode. VMWare Player is a hosted hypervisor that can be freely downloaded and can run virtual machines created by VMWare Workstation or Server. Xen can run as a full virtualization system for basic architectures with the CPU virtualization features present.

In paravirtualization, the hardware is not simulated; instead, the guest runs in its own isolated domain. In this paradigm, the hypervisor exports a modified version of the physical hardware to the guest operating system. Some changes are needed at the operating system level. Figure 1.7 shows a schematic diagram of paravirtualization.

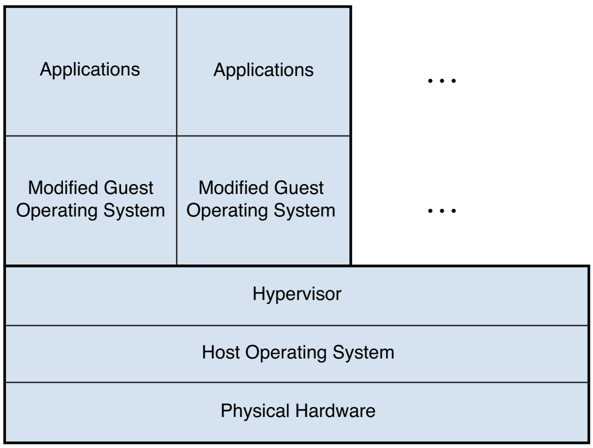

Figure 1.7 Schematic diagram of paravirtualization

Xen is an example of a paravirtualization implementation. VMWare and Windows Hyper-V can also run in paravirtualization mode.

In operating system–level virtualization, the hypervisor is integrated into the operating system. The different guest operating systems still see their own file systems and system resources, but they have less isolation between them. The operating system itself provides resource management. Figure 1.8 shows a schematic diagram of operating system–level virtualization.

One of the advantages of operating system–level virtualization is that it requires less duplication of resources. Logical partitions on the IBM AIX operating system serves as an example of operating system–level virtualization.

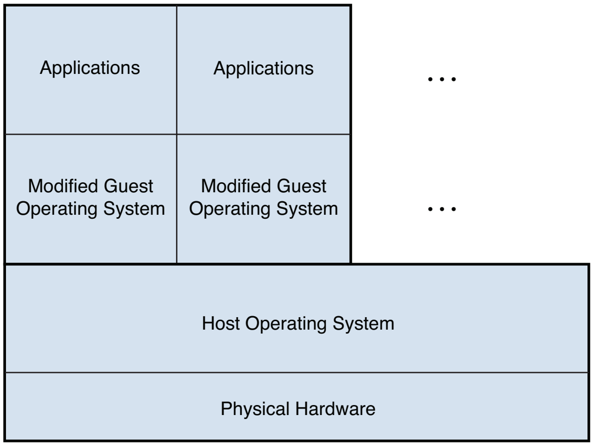

Figure 1.8 Schematic diagram of operating system–level virtualization

KVM can be considered an example of operating system–level virtualization. KVM is a Linux kernel module and relies on other parts of the Linux kernel for managing the guest systems. It was added to the Linux kernel in version 2.6. KVM exports the device `/dev/kvm`, which enables guest operating systems to have their own address spaces, to support isolation of the virtual machines. Figure 1.9 shows the basic concept of virtualization with KVM.

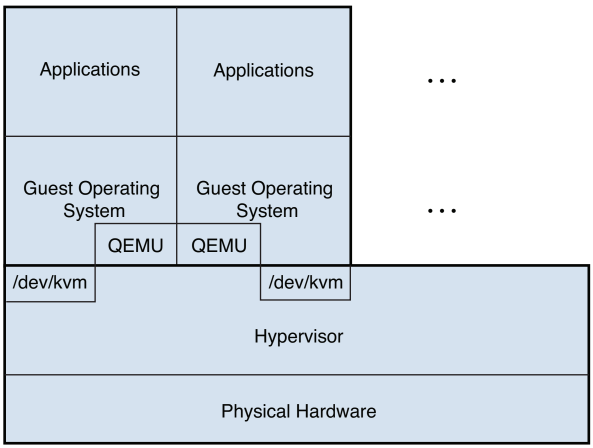

Figure 1.9 Virtualization with KVM

KVM depends on libraries from the open source QEMU for emulation of some devices. KVM also introduces a new process mode, called *guest,* for executing the guest operating systems. It is a privilege mode sufficient to run the guest operating systems but not sufficient to see or interfere with other guest systems or the hypervisor. KVM adds a set of shadow page tables to map memory from guest operating systems to physical memory. The `/dev/kvm` device node enables a userspace process to create and run virtual machines via a set of `ioctl()` operations, including these:

- Creating a new virtual machine
- Allocating memory to a virtual machine
- Reading and writing virtual CPU registers
- Injecting an interrupt into a CPU
- Running a virtual CPU

In addition, guest memory can be used to support DMA-capable devices, such as graphic displays. Guest execution is performed in the loop:

- A userspace process calls the kernel to execute guest code.
- The kernel causes the processor to enter guest mode.
- The processor executes guest code until it encounters an IO instruction or is interrupted by an external event.

Another key difference between virtualization systems is between client-based and server-based virtualization systems. In a client-based virtualization system, such as VMWare Workstation, the hypervisor and virtual machine both run on the client that uses the virtual machine. Server products, such as VMWare ESX, and remote management libraries, such as libvirt, enable you to remotely manage the hypervisor. This has the key advantage of freeing the virtual machine from the client that consumes it. One more step in virtualization is needed in cloud computing, which is to be able to manage a cluster of hypervisors.

Computing capacity is not the only resource needed in cloud computing. Cloud consumers also need storage and network resources. Those storage and network resources can be shared in some cases, but in other cases, they must be isolated. Software based on strong cryptography, such as secure shell (SSH), can be used safely in a multitenant environment. Similarly, some software stores data in encrypted format, but most does not. Thus, storage and network virtualization and tenant isolation are needed in clouds as well.

Storage virtualization provides logical storage, abstracting the details of the storage technology from users and application software. This is often implemented in network-attached storage devices, which can provide multiple interfaces to a large array of hard disks. See the “Storage” section later in this chapter for more details.

Network resources can also be virtualized. This book is most concerned with virtualization at the IP level. In the 1990s, local area networks (LANs) were created by stringing Ethernet cable between machines. In the 2000s, physical network transport was incorporated directly into cabinets that blade servers fit into, to keep the back of the cabinet from looking like a bird’s nest of

Ethernet cable. Today we can do the virtual equivalent of that with virtual network management devices in a VLAN, which can be managed conveniently and also provides network isolation for security purposes. See the “Network Virtualization” section later in this chapter for more details.

These virtualization platforms provide great convenience, but management comes at the cost of learning them and developing efficient skills. Some other limitations exist as well:

- The different virtual hosts must be managed separately, and only a limited number of guest machines can be placed on one host. Today 16 dual-core CPU machines are affordable, to support around 32 capable virtual machines, but we need a way to scale to larger numbers.
- End users still need to contact a system administrator when they want a new virtual machine. The administrator then must track these requests and charge for use.
- Virtualization itself does not provide a library of images that can be readily used. A feature of organizations that use a lot of direct virtualization is image sprawl, consisting of a large number of unmanaged virtual machine images.
- The system administrator still must manage the various pieces of the infrastructure. Some small companies cannot afford system administrators, and many large organizations would like to reduce the number of system administrators they currently have.
- Hardware still must be bought. Most enterprises would like to minimize their capital investments in hardware.

Infrastructure as a Service Clouds

An IaaS cloud provides abstractions beyond virtualization so that you do not need to learn how to manage them yourself. In fact, when using an IaaS cloud, you normally are not even aware of what virtualization platform is being used. In this case, the cloud service provider is concerned about the virtualization platform; you do not need to worry about it. Figure 1.10 shows some of the main components of an IaaS cloud.

The user logs into a self-service user interface that is part of a system called the business support services (BSS). The BSS knows how to charge a user for the resources used, and the self-service user interface enables the user to create and manage resources such as virtual machines, storage, and IP addresses. This gives the user a central location for viewing and managing all resources instead of being left to manage a collection of independent virtualization technologies. A programmatic API also is often provided, to enable automation of resource management for a similar set of capabilities as those of the self-service user interface. The operational support system (OSS) manages a collection of hypervisors and federates other virtualized resources. The end result is that the user can have access to the virtual machine without having to know how it was created.

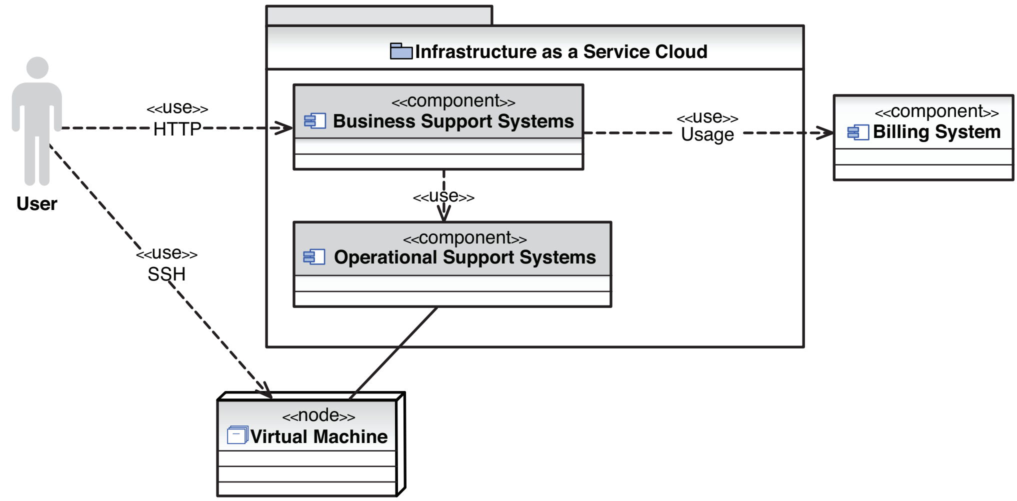

Figure 1.10 Basic concepts of an Infrastructure as a Service cloud

Additional features of IaaS clouds, such as the IBM SmartCloud Enterprise, are convenient for enterprises. Importantly, the relationship is between the cloud provider and the enterprise. An enterprise contact can thus manage the users, who can create and use resources that the enterprise is paying for. In addition, the work products created by people using the cloud should belong to the enterprise, and the cloud infrastructure should support this.

One of the most interesting aspects of cloud computing is that it enables a new level of tooling and collaboration. It enables reuse of work products, especially images, by teams. For example, an operating system expert can set up a base operating system image, a software developer can add an installation of a software product on top of it, and an enterprise user can make use of the image by taking snapshots suitable for his or her enterprise's needs. Figure 1.11 shows how a developer can interact with cloud tools to provide assets that an end user can consume.

Business support systems (BSS) are a critical part of the cloud and might be important to your applications if you sell services to customers. Most online systems need a BSS. BSS includes subscriber management, customer management, contract management, catalog management, business partner enablement, metering, and billing. Clearly, BSS is a wider concept than just IaaS. The Apple iPhone AppStore and the Android AppStore are examples of platforms that include a BSS.

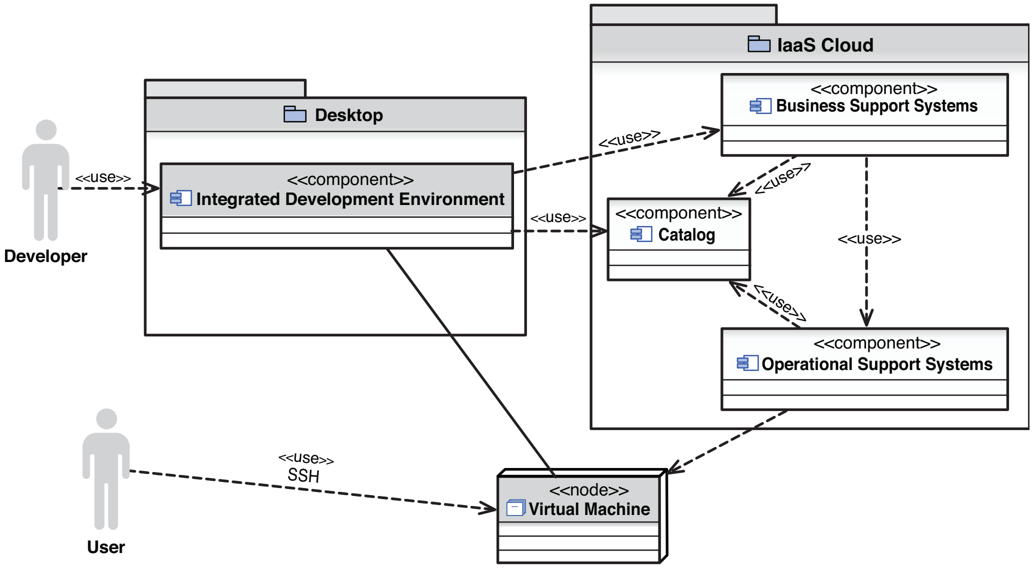

Figure 1.11 Use of development tools in a cloud environment

Other Cloud Layers

Cloud layers operate at a higher level of abstraction than IaaS, including Platform as a Service (PaaS), Software as a Service (SaaS), and Business as a Service (BaaS). In its definition of cloud computing, NIST recognizes three of these layers: IaaS, PaaS, and SaaS. Figure 1.12 illustrates this concept of different layers of cloud computing.

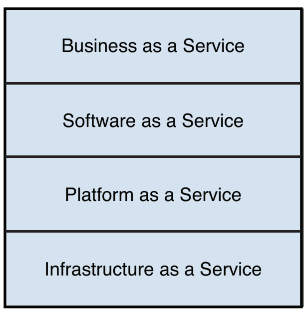

Figure 1.12 Cloud platform layers

Infrastructure as a Service is a term for services that provide a model for dynamically allocating infrastructure and software, starting with an OS, on that infrastructure. Platform as a Service describes concrete services used in the execution of an application or higher-level service. The services provide some generalized and reusable capability to their software consumer and are thus a "platform" service being consumed. They bring their own interface and programming model for the consumer to use, along with their own API, data, messaging, queueing, and so on.

If a platform service is hosted by an infrastructure service provider, the IaaS API is likely used as part of the process to instantiate and access the platform service, but it is a separate concept.

To complete the terminology, Software as a Service is typically a self-sufficient software solution to a consumer need. Typically, this is a tool or business application with on-demand, turn-key characteristics.

If a software service leverages a platform service, it normally does so transparently. If the software service is hosted by an infrastructure service provider, the IaaS application programming interface can be used as part of the process to instantiate and access the software service.

If you look at the service stack from the top down, you can see some of the value the other layers provide. At the very top are business services such as Dunn and Bradstreet, which provides analysis and insight into companies that you might potentially do business with. Other examples of business services are credit reporting and banking. Providing business services such as these requires data stores for storing data. However, a relational database by itself is not sufficient: The data retrieval and storage methods must be integrated into programs that can provide user interfaces for people can use. Relational databases also need to be maintained by database administrators who archive and back up data. This is where Platform as a Service comes in. Platform as a Service provides all the services that enable systems to run by themselves, including scaling, failover, performance tuning, and data retrieval. For example, the Salesforce Force.com platform provides a data store where your programs can store and retrieve data without you ever needing to worry about database or system administration tasks. It also provides a web site with graphical tools for defining and customizing data objects. IBM Workload Deployer is another Platform as a Service that runs on an infrastructure as a service cloud but is aware of the different software running on individual virtual machines; it can perform functions such as elastic scaling of application server clusters.

With Platform as a Service, you still need to write a program that enables a user to interact with it via a graphical user interface. If you do not like that idea, you can use Software as a Service. Salesforce.com enables enterprises to use a customer relationship management (CRM) system without having to do any programming or software installation. Its web site also supports graphical tools for customizing menus, data entry forms, and reports. It works great if you all you need to do is create, retrieve, update, and delete data or use a predefined service, such as email. If you need to do more than that, you need to drop down to the Platform as a Service level.

Virtual Machine Instances

An instance is a running virtual machine, in addition to some data the cloud maintains to help track ownership and status. The cloud manages a large pool of hardware that can be used to create running instances from images. The virtual machine includes a copy of the image that it instantiates and the changes that it saves while it runs. The instance also includes virtualizations of the different hardware that it needs to run, including CPUs, memory, disk, and network interfaces. The cloud manages a pool of hypervisors that can manage the virtual machine instances. However, as a user of the cloud, you do not need to worry about the hypervisors. In fact, the hypervisor you are using—KVM, Xen, VMWare, or any other—makes no difference.

When you delete an instance, that hardware can be reused. The cloud scrubs your hard disk before doing so, to make sure that the next user of the hardware finds no traces of previous data.

Virtual Machine Images

A virtual machine image is a template for creating new instances. You can choose images from a catalog to create images or save your own images from running instances. Specialists in those platforms often create catalog images, making sure that they are created with the proper patches and that any software is installed and configured with good default settings. The images can be plain operating systems or can have software installed on them, such as databases, application servers, or other applications. Images usually remove some data related to runtime operations, such as swap data and configuration files with embedded IP addresses or host names.

Image development is becoming a larger and more specialized area. One of the outstanding features of the IBM SmartCloud Enterprise is the image asset catalog. The asset catalog stores a set of additional data about images, including a "Getting Started" page, a parameters file that specifies additional parameters needed when creating an instance, and additional files to inject into the instance at startup. It also hosts forums related to assets, to enable feedback and questions from users of images to the people who created those images. Saving your own images from running instances is easy, but making images that other people use requires more effort; the IBM SmartCloud Enterprise asset catalog provides you with tools to do this.

Because many users share clouds, the cloud helps you track information about images, such as ownership, history, and so on. The IBM SmartCloud Enterprise knows what organization you belong to when you log in. You can choose whether to keep images private, exclusively for your own use, or to share with other users in your organization. If you are an independent software vendor, you can also add your images to the public catalog.

Some differences between Linux and Windows exist. The filelike description of the Linux operating system makes it easy to prepare for virtualization. An image can be manipulated as a file system even when the instance is not running. Different files, such as a user's public SSH key and runtime parameters, can be injected into the image before booting it. Cloud operators take advantage of this for ease of development and to make optimizations. The same method of manipulating files systems without booting the OS cannot be done in Windows.

Storage

Virtualization of storage can be done in different ways to make physical storage transparent to consumers of I/O services. Block storage is storage handled as a sequence of bytes. In file-based storage systems, the block storage is formatted with a file system so that programs can make use of file-based I/O services to create and manage files. Virtualization can be done at both levels.

Block Storage

Usually, the person installing an operating system partitions physical hard disks in a physical computer. A disk partition is a logical segment of a hard disk. Partitioning a disk can have several advantages, including separating the operating system from user files and providing a storage area for swapping. The disadvantages of partitioning include the need to reorganize or resize if you run out of space on one partition. The classical example is running out of space on your operating system partition (C:) when you still have plenty of space on the other partitions. One advantage of partitions in virtual systems is that you can plan for a large amount of storage space but do not have to actually allocate that space until you need to use it.

Clouds can make use of partitions as well. In the IBM SmartCloud Enterprise, when you provision a virtual machine, you have an option to create only the root file partition. This optimizes startup time. If you have a large amount of storage associated with the image, the time savings can be considerable. Later, when you use the storage, it is then allocated.

A Linux *logical volume manager* (LVM) provides a level of abstraction above block devices, such as hard disks, to allow for flexibility in managing storage devices. This can make it easier to resize physical partitions, among other tasks. The LVM manages *physical volumes*, which can be combined to form a *volume group*. *Logical volumes* can then be created from the volume groups. The logical volumes can span multiple physical volumes, allowing them to be any size up to the total size of the volume group.

Copy on write is a technique for efficiently sharing large objects between two or more clients. Each client appears to have its own writable copy of the object, but each client actually has only a read-only copy of the shared object. When a client tries to write to the object, a copy of the block is made and the client is given its own copy. This is efficient when the object is only rarely changed by client programs, such as an operating system when a virtual machine loads and runs it. This technique can make starting the virtual machine much faster than first copying the operating system to a separate storage area before booting the virtual machine. In this context, copy on write is often used with a network-based file system.

The term *direct attached storage* is usually used to contrast local block-based storage with *network attached storage*. Direct attached storage is simple, cheap, and high performance. Its high-performance characteristics are due to the fact that it is directly attached. Its disadvantages include that its lifetime is usually tied to the lifetime of the virtual machine. In addition, it might not be scalable if you do not have physical access to the machine. In a cloud environment, you often have no way of increasing direct attached storage, so be sure to start with enough.

In an Infrastructure as a Service cloud, you do not need to be concerned with the different storage implementations the cloud provider uses. Instead, you should be concerned with the amount of storage and the level of performance the storage service provides. Cloud consumers need a basic understanding of the concepts to do informed planning. Generally, local storage comes and goes with virtual machines, and remote storage can be managed as an independent entity that can be attached to or detached from a virtual machine. In general, local and remote storage have a large difference in performance. Remote storage is not suitable for some applications, such as relational databases.

File-Based Storage

File systems provide a level of abstraction over block storage, to allow software to more easily use and manage files. As with block-based storage, a fundamental difference exists between local and remote file systems. Common local file systems in clouds are ext3 and ext4 on Linux and NTFS on Windows. Common remote file systems are NFS on Linux and CIFS on Windows. One huge difference between remote files systems and network attached storage, such as AoE and iSCSI, is that remote file systems are designed for multiple clients with simultaneous write access. This is not possible with remote block devices provided by network attached storage.

Some distributed file systems can span many servers. Apache Hadoop is an example of such a distribute file system used by many large web sites with huge storage requirements. Hadoop is discussed in the upcoming "Hadoop" section in Chapter 5, "Open Source Projects."

Table 1.4 compares different basic storage options.

Table 1.4 Comparison of Different Storage Options

Storage Option	Advantages	Disadvantages
Local block based	High performance	Lifetime tied to a virtual machine
Remote block based	Can be managed independently, with a lifetime not tied to a virtual machine	Cannot be shared among multiple virtual machines
Local file based	High performance	Lifetime tied to a virtual machine
Remote file based	Can be shared among different clients	Relatively lower performance

The persistence of virtual machines and their local storage can vary with different virtualization methods. Some virtual machines' local storage disappears if the virtual machine is deleted. In other implementations, the local storage is kept until the owner deletes the virtual machine. The IBM SmartCloud Enterprise uses this model. Some virtualization implementations support the concept of a persistent virtual machine. In a third model, some implementations boot the operating system from network attached storage and do not have any local storage. Be sure to understand the storage model your cloud provider uses so that you do not lose data.

Network Virtualization

Networking is one of the fundamental elements of cloud computing and also one of the hazards to users of cloud computing. Network performance degradation and instability can greatly affect the consumption of cloud resources. Applications that are relatively isolated or are specially designed to deal with network disruptions have an advantage running in the cloud.

From a different perspective, network resources can be virtualized and used in cloud computing just as other resources are. In this section, we first discuss basic use of IP addresses in a cloud context and then cover virtual networks.

Delivery of cloud services takes place over networks at different levels using different protocols. This is one of the key differences in cloud models. In PaaS and SaaS clouds, delivery of services is via an application protocol, typically HTTP. In IaaS, cloud services can be delivered over multiple layers and protocols—for example, IPSec for VPN access and SSH for command-line access.

Management of the different layers of the network system also is the responsibility of either the cloud provider or the cloud consumer, depending on the type of cloud. In a SaaS model, the cloud provider manages all the network layers. In an IaaS model, the cloud consumer manages the network levels, except for the physical and data link layers. However, this is a simplification because, in some cases, the network services relate to the cloud infrastructure and some services relate to the images. The PaaS model is intermediate between IaaS and SaaS.

Table 1.5 summarizes the management of network layers in different cloud scenarios.

Table 1.5 Management for Network Layers

OSI Layer	Example Protocols	IaaS	PaaS	SaaS
7 Application	HTTP, FTP, NFS, SMTP, SSH	Consumer	Consumer	Provider
6 Presentation	SSL, TLS	Consumer	Provider	Provider
5 Session	TCP	Consumer	Provider	Provider
4 Transport	TCP	Consumer	Provider	Provider
3 Network	IP, IPSec	Consumer	Provider	Provider
2 Data link	Ethernet, Fibre Channel	Provider	Provider	Provider
1 Physical	Copper, optical fiber	Provider	Provider	Provider

This table is a simplification of the many models on the market. However, it shows that an IaaS gives cloud consumers considerably more flexibility in network topology and services than PaaS and SaaS clouds (but at the expense of managing the tools that provide the flexibility).

IP Addresses

One of the first tasks in cloud computing is determining how to connect to the virtual machine. Several options exist when creating a virtual machine: system generated, reserved, and VLAN IP address solutions. System-generated IP addresses are analogous to Dynamic Host Control Protocol (DHCP)–assigned addresses. They are actually static IP addresses, but the IaaS cloud assigns them. This is the easiest option if all you need is a virtual machine that you can log into and use.

Reserved IP addresses are addresses that can be provisioned and managed independently of a virtual machine. Reserved IP addresses are useful if you want to assign multiple IP addresses to a virtual machine.

IPv6 is an Internet protocol intended to supersede IPv4. The Internet needs more IP addresses than IP v4 can support, which is one of the primary motivations for IPv6. The last top-level block of IPv4 addresses was assigned in February 2011. The Internet Engineering Task Force (IETF) published *Request for Comments: 2460 Internet Protocol, Version 6 (IPv6)*, which was the specification for IPv6 in 1998. IPv6 also provides other features not present in IPv4. Network security is integrated into the design of IPv6, which makes IPSec a mandatory part of the implementation. IPv6 does not specify interoperability with IPv4 and essentially creates an independent network. Today usage rates of IPv6 are very low, and most providers operate in compatibility/tolerance mode. However, that could change.

Network Virtualization

When dealing with systems of virtual machines and considering network security, you need to manage networks. Network resources can be virtualized just like other cloud resources. To do this, a cloud uses virtual switches to separates a physical network into logical partitions. Figure 1.13 shows this concept.

VLANs can act as an extension of your enterprise's private network. You can connect to it via an encrypted VPN connection.

A hypervisor can share a single physical network interface with multiple virtual machines. Each virtual machine has one or more virtual network interfaces. The hypervisor can provide networking services to virtual machines in three ways:

- Bridging
- Routing
- Network address translation (NAT)

Bridging is usually the default mode. In this mode, the hypervisor works at the data link layer and makes the virtual network interface externally visible at the Ethernet level. In routing mode, the hypervisor works at the network layer and makes the virtual network interface externally visible at the IP level.

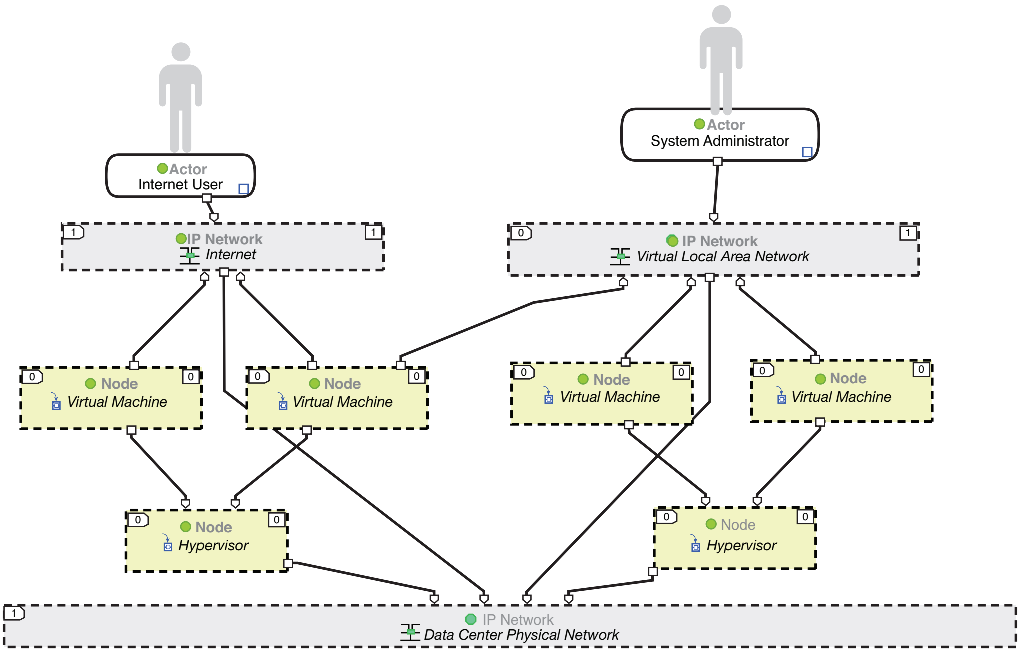

Figure 1.13 Physical and virtual networks in a cloud

In network address translation, the virtual network interface is not visible externally. Instead, it enables the virtual machine to send network data out to the Internet, but the virtual machine is not visible on the Internet. Network address translation is typically used to hide virtualization network interfaces with private IP addresses behind a public IP address used by a host or router. The NAT software changes the IP address information in the network packets based on information in a routing table. The checksum values in the packet must be changed as well.

NAT can be used to put more servers on the network than the number of virtual machines you have. It does this by port translation. This is one reason IPv6 is still not in wide use: Even though the number of computers exceeds the number of IP addresses, you can do some tricks to share them. For example, suppose that you have a router and three servers handling HTTP, FTP, and mail, respectively. You can assign a public IP address to the router and private IP addresses to the HTTP, FTP, and mail servers, and forward incoming traffic (see Table 1.6).

Table 1.6 Example of Network Address Translation

Public IP	Port	Private IP
9.0.0.1 (router)	80, 443	192.168.0.1 (HTTP server)
	21	192.168.0.2 (FTP server)
	25	192.168.0.3 (mail server)

Desktop Virtualization

Desktops are another computing resource that can be virtualized. Desktop virtualization is enabled by several architectures that allow remote desktop use, including the X Window System and Microsoft Remote Desktop Services. The X Window System, also known as X Windows, X, and X11, is an architecture commonly used on Linux, UNIX, and Mac OS X that abstracts graphical devices to allow device independence and remote use of a graphical user interface, including display, keyboard, and mouse. X does not include a windowing system—that is delegated to a window manager, such as KDE or Gnome. X is based on an MIT project and is now managed by the X.Org Foundation. It is available as open source software based on the MIT license. X client applications exist on Linux, UNIX, Mac OS X, and Windows. The X server is a native part of most Linux and UNIX systems and Mac OS X and can be added to Windows with the Cygwin platform. The X system was designed to separate server and client using the X protocol and lends itself well to cloud computing. X Windows is complex and can involve some troubleshooting, but because it supports many varied scenarios for its use, it has enjoyed a long life since it was first developed in 1984.

The Remote Desktop Service is a utility that enables users to use a Microsoft Windows graphical user interface on a remote computer. Remote Desktop Service makes use of the Remote Desktop Protocol (RDP), a protocol that Microsoft developed to support it. Client implementations exist for Windows (including most variants, such as Windows 7, XP, and Mobile), Linux, UNIX, Mac OS X, and others. Remote Desktop Service was formerly known as Terminal Services. The Remote Desktop Service implementation of RDP is highly optimized and efficient over remote network connections.

In addition to X and RDP, two other remote graphical user interface platforms worth mentioning are Virtual Network Computing (VNC) and the NX Client. VNC is a system that uses a remote control paradigm that uses Remote Framebuffer Protocol (RBP). Because it is based at the framebuffer level, it can operate on all Windows systems, including Linux/UNIX and Windows. VNC is open source software available under the GNU license. Setup of VNC on a Linux system is described in the section “Linux, Apache, MySQL, and PHP” in Chapter 2, “Developing on the Cloud.”

NX is commercial/open source developed by NoMachine. NX Server and Client are the components of the platform, which operates over an SSH connection. The big advantage of the MoMachine NX system is that it works over a secure channel. NX also compresses the display data and uses a client proxy to make optimal use of network bandwidth. Future versions of the tool from NoMachine might be commercial only, with the FreeNX project producing the open source version. Figure 1.14 shows the basic concepts of VNC and NX.

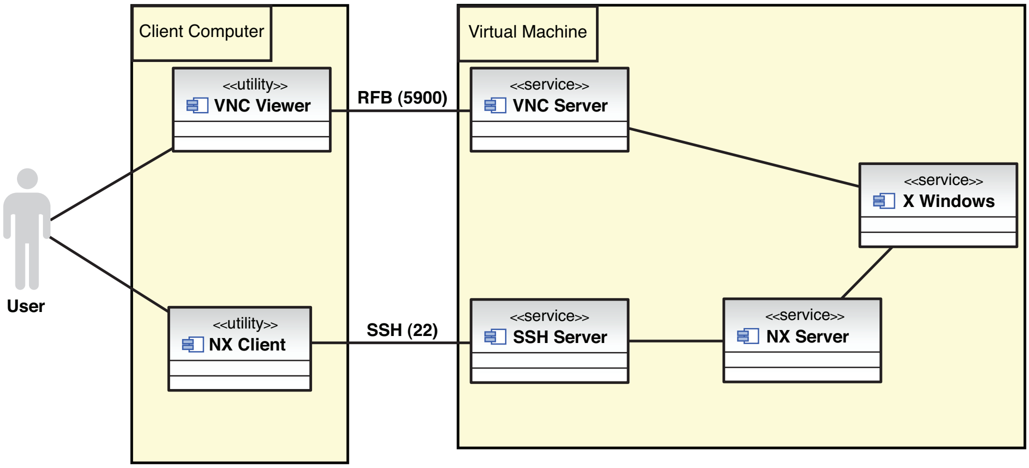

Figure 1.14 Remote Desktop Management with VNC and NX Client

Commercial distributions of NX can support many desktops centrally. Linux desktops, such as KDE and Gnome, work over the top of X to enable users to manage Windows in a multitasking environment and personalize their desktop settings. You can use the desktop environment of your choice with either VNC or NX.

In addition to VNC and NX, several open source and commercial implementations of X are available for Microsoft Windows, including Cygwin X server, Hummingbird Exceed, Reflection X, and Xming.

See the sections "Linux, Apache, MySQL, and PHP" in Chapter 2 for basic use of VNC and the section "Remote Desktop Management" in Chapter 6, "Cloud Services and Applications," for more details on using virtual desktops.

This page intentionally left blank

Index

E

F

G–H

I

M

N

O

S

W

X–Y–Z