PART I Groundwork

Welcome to the world of Linux and open source. This book assumes that you have a technology background, but does not assume prior knowledge of Linux and open source. Part 1 lays a foundation of basic knowledge that you need to navigate and understand an open source operating system, development model, and licensing process.

Chapter 1 will give you the business basics of why you need to care about Linux and open source. Chapter 2 will detail the core of the Linux kernel. Chapter 3 will walk you through the Open Source Definition and its associated licenses. Finally, Chapter 4 will give a view of just how big the open source movement really is.

01Fink.F 8/19/02 5:46 PM Page 2

CHAPTER 1

The Business of Linux and Open Source

Linux and open source present many new technologies, ideas, concepts, and paradigms. Linux is a major new operating system that was developed using open source methodologies. To delve into what Linux is and how open source development can affect your enterprise, a crash course in some of these new concepts is required. The goals for this chapter are for you to understand:

- Where Linux is used and how it is growing
- Terminology that will be useful in understanding the book
- The key business benefits of Linux and open source
- Major inhibitors limiting the growth of Linux
- Major players in the open source community

This chapter explains why Linux and open source are important to your business and gives you the base foundation that will enable you to progress through the book.

Linux Adoption

Normally we look at charts that show the revenue that a product or service is expected to generate in the marketplace. That picture cannot be accurate for Linux as it is impossible to accurately count the actual number of copies of Linux in use. This is very difficult to do for Linux because

Linux is open source. Yes, it is possible to purchase copies of Linux. But, most often, companies will purchase one copy, customize it, and reuse the customized copy throughout the enterprise. Linux can also be downloaded from numerous places for free, built from scratch, included with books, and bundled with any number of products. Some users also configure their systems in a dual-boot configuration to allow the use of multiple different operating systems on the same computer. This makes the task of counting the actual number of copies of Linux in use very difficult, although many industry analysts, such as what IDC published and represented in Figure 1–1, are attempting approximations. Figure 1–1 enumerates the number of copies (both free and purchased) of Linux operating environments in use.

As I mentioned, since Linux itself can be obtained for free, the market revenue of the Linux operating system itself tends to be meaningless. But, having an approximate view of the copies in use gives you a leading indicator of what associated product revenues, such as clients, software, storage, and others, will be. This chart gives you a view of how fast Linux has been growing and that its growth is not stopping anytime soon. Notice that you must consider free as well as purchased copies. You should logically infer from this that companies in many technology segments will shift future product investments to Linux to take advantage of this growth.

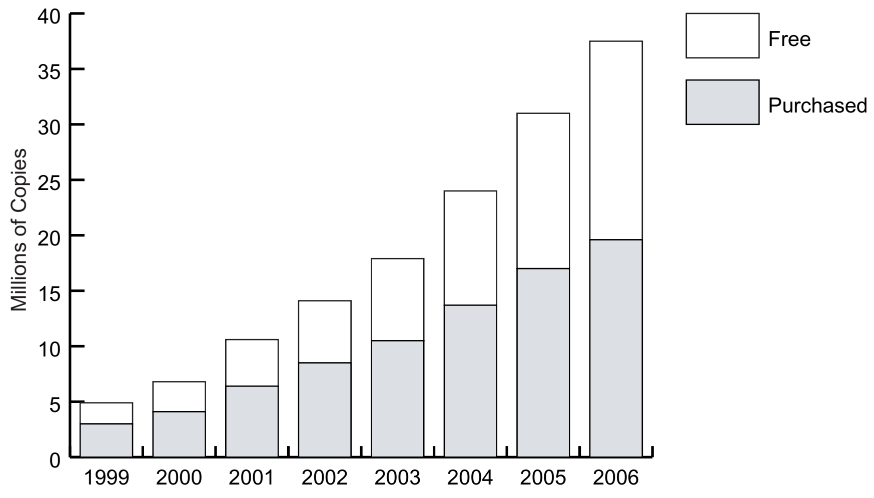

Figure 1–1 Linux adoption (Source: IDC, 2002).

Crash Course in Linux and Open Source Lingo

By the time you finish this book, you will have a solid understanding of how Linux and open source work and apply to your business. However, an early crash course on some key terminology will help us get through some of the early parts of this book. This set of explanations is only meant to give you enough understanding to make it through the book. Most of these concepts will be explained in detail as the book progresses.

- **Kernel**—The kernel is the heart of the operating system that manages the key resources (such as memory, processes, etc.) on your system. Linux is in fact a kernel, not an operating system.
- **GNU**—This is a recursive acronym for **G**NU's, **N**ot **U**NIX. As you can see the first letter of the acronym is the acronym itself. Whereas Linux is the kernel of the operating system, the GNU system represents many of the other parts (compilers, tools, editors, etc.) needed to have a useful system. Purists contend that referring to Linux as the operating system is erroneous and it should be referred to as the GNU/Linux system, an important distinction to keep in mind as you discuss Linux opportunities in your company.
- **Distribution**—Most companies that deploy or develop applications for Linux will usually do this by acquiring a distribution. Some of the more common Linux distributions are Red Hat, SuSE, Debian, Mandrake, Connectiva, and Red Flag. Distribution vendors combine the Linux kernel, many parts of the GNU system, other open source components, and enhancements of their own to distribute the whole as an integrated and tested system. In many instances, distributions are targeted at specific sectors or specific geographies of the market.
- **Package**—The individual components installed on a Linux system are distributed as packages. Generically, a package refers to any collection of files distributed together to serve a specific purpose. Since many packages can share the same set of files, complex interdependencies are managed by installation and deployment tools.
- **Free software**—Free is meant to signify freedom, as in free speech, and not to characterize "without cost." Free software originated with the Free Software Foundation under the premise that all software should be shared and everyone should have equal access to source code, or the blueprints of the system.

- **Open source**—This term will generally be used in two contexts: first, as the marketing phrase for free software; and second, as a development methodology that I will cover in great detail in Part 3. While the leaders of the free software movement tend to object to the association with open source, the term was coined to overcome the common misconception that free software implied software without cost. Open source is ***not*** a license. It provides a common set of specifications to which licenses must adhere to be considered open source.
- **Community**—Any collection of software developers working collaboratively on a software project. A community can represent students, hobbyists, corporate developers, competitors, and customers, among many others.
- **Maintainer**—The individual, committee, board of directors, or foundation that accepts or rejects code changes into an open source project. You can also think of the maintainer as the project manager. Probably the most well-known maintainer is Linus Torvalds (the creator of the Linux kernel).
- **GPL**—This stands for the GNU general public license. The GPL will be covered in depth in Chapter 3. For now, understanding that the GPL is the most frequently used open source license will suffice. The GPL is the license that governs the Linux kernel. The GPL requires that any modifications to the source base be returned to the community at large.

This list of terms should be enough to get you going. As you read through the book, refer back to these to make sure you have a solid grasp of these key terms. By the time you finish the book, most of these should be very familiar to you.

Linux Workloads

Figure 1–2 demonstrates the key places where Linux is being used and how usage is expanding over time. In the server realm, Linux is strongest at the edge of the network. This is largely due to the powerful combination created by the Linux operating system and Apache Web Server. These appliance-like devices draw on the strength of Linux as a network operating system.

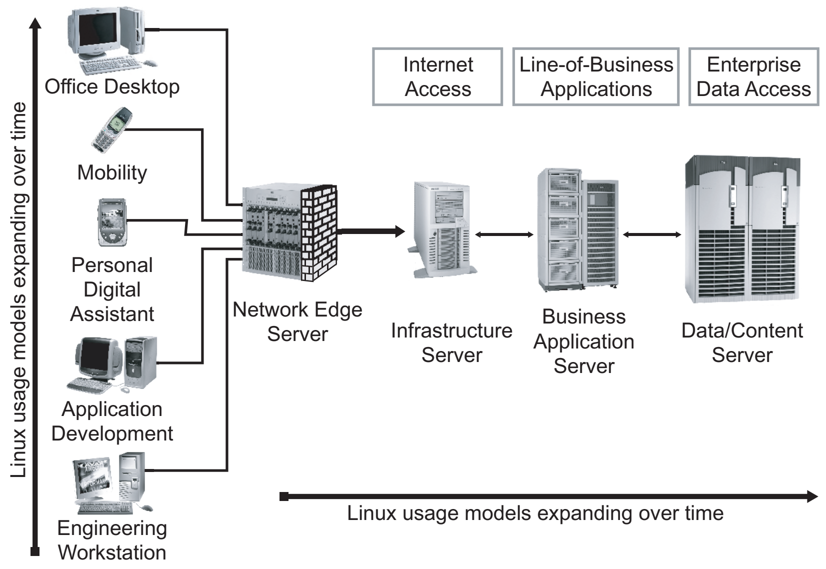

Figure 1–2 Linux workloads growing over time.

As Linux matures, it will gradually see its application workloads evolve to the infrastructure and application server tiers. The great debate is how long this evolution will take and how far Linux will push into the enterprise data center. On the client side, Linux, with its heritage of being a UNIX-like operating system, has naturally gained popularity with engineering workstations and application development systems. Linux has also started to penetrate the embedded device arena. The great debate here is whether or not Linux will become a viable operating system for the general office desktop. While these are general adoption trends, there are pockets of adoption already scattered in virtually all areas of the computing spectrum.

Business Benefits

Linux has rapidly evolved from a niche and hobbyist toy to a credible environment in use at many businesses. Much of Linux's initial transition into the business world was as a result of the Internet boom. Service providers saw Linux as a way to deploy systems using commodity components and not have any costs associated with licensing either the operating system or key open source applications such as the Apache Web Server and other programs that are essential to the Internet and that run well under Linux. Service providers tend to have highly technical personnel, thereby rendering vendor support a controllable issue. Mainstream businesses are now seeing that Linux is maturing as a credible alternative to other operating environments from the cost, resource, and control perspectives. Here are some of the key business benefits of Linux:

Cost

The fact that Linux can be freely copied, subject to reasonable license terms, without payment of royalties is clearly viewed by many as one of the main business advantages of Linux. Although it is possible for you to deploy Linux without paying license fees, this does not mean that Linux, or any operating system, is "free" in the sense that there are no costs associated with its installation, maintenance, support, and training. In Chapter 6, we will take a detailed look at some of the new and different cost considerations associated with Linux (and many other open source software).

Availability of Trained Resources

Linux is now more than 10 years old and so trained resources are available. The cost advantages of Linux have enticed many education and research institutions such as the University of Waterloo in Ontario, Canada, the University of Illinois at Urbana-Champaign/NCSA (National Center for Super Computing Applications), and many others to deploy Linux aggressively. This has now been happening for a few years. For example, the Groupe ESIEE in France, a center for advanced scientific and technical education, has been using Linux for three years because Linux is open source. They state their motivation very simply: ESIEE develops for the community, and the community develops for ESIEE. It is a simple return on investment for the center. The University of New South Wales converted all of their UNIX-based teaching labs to Linux in one year. The result is that many, many new graduates are fully trained and well-versed in the Linux operating system, even more so than in most flavors of UNIX.

It also follows that the newest generations of developers are all trained and have built their expertise on Linux and its associated application programming interfaces (APIs). Many of the most popular companion open source applications, such as the Apache Web Server, are also very familiar to this new generation.

Figure 1–3 shows the key Linux stakeholders that represent core competence in the Linux movement and resources you can draw on.

As your enterprise grows and you need more talented personnel to run your network infrastructure, application servers, and data center, Linux-capable talent will be one of the most readily available resources.

Support

One of the raging debates about Linux is the issue of where support comes from. One side of the debate argues that since Linux is maintained and enhanced by a community of loosely coupled developers, the ability to get guaranteed support is questionable. The other side of the debate argues that since the code is available to all, anyone can provide support and that self-support now becomes a lower cost and more viable option. Which side of the debate you take will depend largely on the type of information technology (IT) organization you have built and your previous experiences with various vendors. The benefit to you is that you have the choice of which support model to implement and anyone can provide support. Chapter 8 will go into the details of support options and the implications of the choices you make.

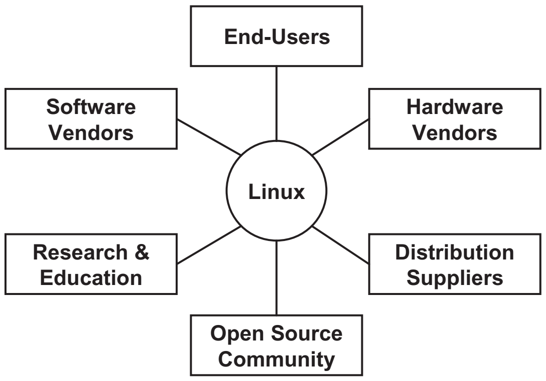

Figure 1–3 Linux stakeholders.

Control and Vendor Independence

The Intel x86 architecture (also known as IA-32) brought commodity economics to the personal computer (PC). In many ways, Linux does the same thing for the operating system. Prior to Linux, your choice was primarily a proprietary operating system on proprietary hardware, or a proprietary operating system on commodity hardware. While there are other open source operating systems such as BSD, Linux is the first viable combination of a commodity operating system on commodity hardware (Linux is also available on a wide range of proprietary and embedded architectures). As you will discover later, the licensing model used by Linux is what has differentiated it from BSD and created a single operating system developed by many. With more and more independent software vendors (ISVs) making their applications available on Linux, IT managers can now deploy solutions on the hardware and operating environment that is the most cost-effective and delivers the highest return.

Enterprises that have deployed commodity hardware on the Intel IA-32 architecture have seen the many benefits of commodity economics. Companies are no longer tied to one vendor and the switching costs are very low. Companies also have the choice to mix and match hardware and peripherals since a large ecosystem develops around commodity products. Most, if not all, of these same advantages, which boil down to choice and control, are available with Linux. If one Linux operating system vendor does not provide the level of service or quality your company needs, then switch. If the support does not measure up, find someone else or do it yourself. The do-it-yourself option is one that is much more viable with open source software than with commodity hardware. With commodity hardware, you still need to work with one hardware vendor for the features or defect repairs you need. With software, you can implement changes on your own or seek the help of professionals to do it on your behalf.

Software Development

Linux has become the primary platform for Fluent's development team. Fluent is a leading vendor of computer-aided engineering (CAE) software solutions. Linux gives Fluent's developers an extremely fast and stable platform with a wealth of freeware tools. In addition, developers have complete access to all Windows applications via VMware, an Intel chip emulator that runs under Linux. This gives them access to both UNIX and Windows environments from a single machine.

By almost any measure, Windows with Visual Studio continues to grab the lion's share of the software development market. But, if you segment out the UNIX market, the software development activity is rapidly shifting to Linux. A significant part of this phenomenon is driven by the education and research market mentioned earlier.

Upgrades

Today, Linux is considered to be rock-solid for many application workloads. Every year, companies spend millions of dollars upgrading applications or environments that work just fine. Much of the motivation for the upgrades is due to hardware and licensing requirements by various vendors. With Linux, you are in control. You have the option to consider the costs of self-support, buying support, or doing upgrades and making the best set of tradeoffs for your business. It is not uncommon for users to skip many upgrade opportunities until there is a compelling business benefit and a return for the costs and risks of upgrading an environment. One of the beauties of open source is that you decide when to do this, not the vendor. Hardware is also not necessarily obsolete simply because you upgrade your software. Many developers in the community are making very productive use of systems as much as 10 years old, which are considered obsolete by most others.

It is well understood by most corporations that change implies risks and opportunities, both of which need to be measured and qualified just as any other operational activity within the corporation. If you deploy an environment, no license in the Linux and open source world will compel you to change, downgrade, or upgrade that environment. It is also always possible for you to obtain and use old versions of software. You are in complete control of your environment; therefore, you control the success/failure of your network, and ultimately, your business.

Inhibitors to Linux Growth

Linux is not a panacea to solve all the woes of the average IT department. It is clearly a new opportunity to lower costs, maintain or expand functionality, and partner with a huge community of developers that would be too expensive for anyone to hire. Let's explore some of the more significant inhibitors holding Linux back. As with most things in the high-tech world, there is a large contingent working diligently to eliminate any inhibitors to Linux's growth. Time will eliminate many of the inhibitors listed here.

Application Availability

Linux owes much of its success to one killer open source application: the Apache Web Server. According to the Netcraft Web server survey (*www.netcraft.com*), at the end of 2001, the Apache Web Server was being used by more than 50% of sites on the Internet. The combination of Linux and Apache creates a low-cost, high-performance, easily deployed environment for almost any service provider or IT department.

Beyond Apache, many applications have been developed by research and educational communities, or have been developed in-house by companies on the leading edge.

The classic chicken-and-egg conundrum occurs with any environment, and Linux is no different. While many companies wait for applications to be available before they deploy Linux in their environment, ISVs wait for customers to deploy Linux before they make their applications available.

This chicken-and-egg conundrum is gradually breaking down, however. Part of the breakdown is attributed to the normal maturation of Linux. Also, mainstream application vendors are seeing part of their business being eroded by open source alternatives. One of the first and most significant of these breakdowns was the effect of Apache on the Netscape Web server family.

Another factor driving the change is one already mentioned: the development environment. ISVs have had to maintain multiple development environments for each of their UNIX platforms. Many are shifting to Linux as their core development environment and using it as the base port for all other platforms. In fact, in conversations with ISVs, many of them were surprised to find that a significant portion of their engineers were already using Linux both at work and at home. Some of them expected huge efforts to port to Linux, only to discover that it was already done (or at least, large parts were finished).

Large hardware and software companies are fully supporting Linux. Now that BEA, Dell, Hewlett-Packard (HP), International Business Machines (IBM), Oracle, SAP, and many others are making products available on Linux, the ecosystems around each of these (and many other) companies are also planning to support their products and services on Linux. The snowball is well on its way down the mountain and steadily gaining momentum.

Of course, paying customers have the final say. As more and more companies request, and even demand Linux, vendors will have no choice but to respond.

Maturity

Maturity is a relative term. But, 10 years is actually very young for an operating system, especially when compared to UNIX, which has a heritage that goes back to the 1960s. Linux has a solid and proven track record in the Internet infrastructure. As we saw from Figure 1–2, it is also starting to make inroads in the application server tier. Many of the critical infrastructure applications (databases, application servers, etc.) needed for core business applications are now becoming available.

Another area of maturity can be classified as “enterprise-readiness.” IT managers count on a broad range of tools and capabilities to manage large environments. However, many of the system management, administration, deployment, and update tools that most IT managers take for granted are either not available or very early in their stages of development. Hardware vendors, software vendors, and the open source community at large are working diligently to fill these gaps. This obstacle will be overcome faster as companies put their own engineers to work part-time or full-time on improving this state of Linux. By opting to use this potential control point, companies of any size can influence the direction and development speed of Linux—a luxury not available to many in the past with other operating systems.

Many software vendors are also looking for additional features in the Linux kernel to be able to tune the performance, availability, and scalability of their applications. Today, many of them make compromises to overcome some of these limitations. Again, the open source community provides a proactive forum to solve these limitations in ways not tied to company budgets, roadmaps, financials, or legal constraints.

Scalability

Vertical scalability allows Linux to scale its performance by adding processors within the same system. Horizontal scalability increases scale by distributing tasks among many systems. Scalability is a relative characteristic. For many, many applications, Linux scales very well. With the release of the Linux 2.4 kernel, multi-processor scalability to 16 processors is usually possible, provided the application can take advantage of it. Today’s UNIX systems will often scale to 128 processors and beyond.

However, scalability can take many other forms. Many of those who need scalability beyond what Linux can deliver and also want the cost advantages have turned to clusters to solve their performance needs. Horizontal scalability, or combining a large number of low-cost systems and making them work in tandem, has been and continues to be one of the great strengths of Linux. However, applications are usually developed and tuned to scale well in either multiprocessor or clustered environments. Rarely will applications be optimized to scale well in both configurations. Linux also has many opportunities to improve scalability in input/output (I/O) capabilities, availability, deployment, and management.

The scalability of the Linux kernel continues to be an area of debate. The changes to the kernel involve difficult tradeoffs between complexity and capability. Linus Torvalds, the creator of the Linux kernel (discussed in detail in the next chapter), has tended to err on the side of keeping the kernel small, reducing complexity, and optimizing for smaller systems. This may make it more difficult to find ways to vertically scale the kernel.

Business Risk

Linux (and most open source software) presents a new class of business risks, usually involving the management of intellectual property. Implementing any new technology within an enterprise involves risk that needs to be managed. Linux and open source are no different. Business processes that may not have existed before must be established. Many companies are discovering, however, that managing these risks responsibly are well worth the economic returns that Linux brings. Part 3 of this book will examine, in significant detail, many of the risks associated with open source and how to manage them. In fact, open source opens a wealth of new opportunities to increase productivity and refocus your energy to the core value you bring to your customers.

Who's Who in Open Source?

The community is a collection of individuals around the world working on open source projects. Some do this work as part of their normal duties assigned by their employers; others do it in their spare time because they love the work. Some are driven by simple ego, some by their belief in open source, and others by their desire for more creative answers to their problems. Many of these developers are part of the education and research community. Figure 1–4 shows a basic timeline of significant events in the life of Linux and open source.

In Part 3 of this book, we will cover the reasons why companies encourage and sponsor their employees to work on open source projects.

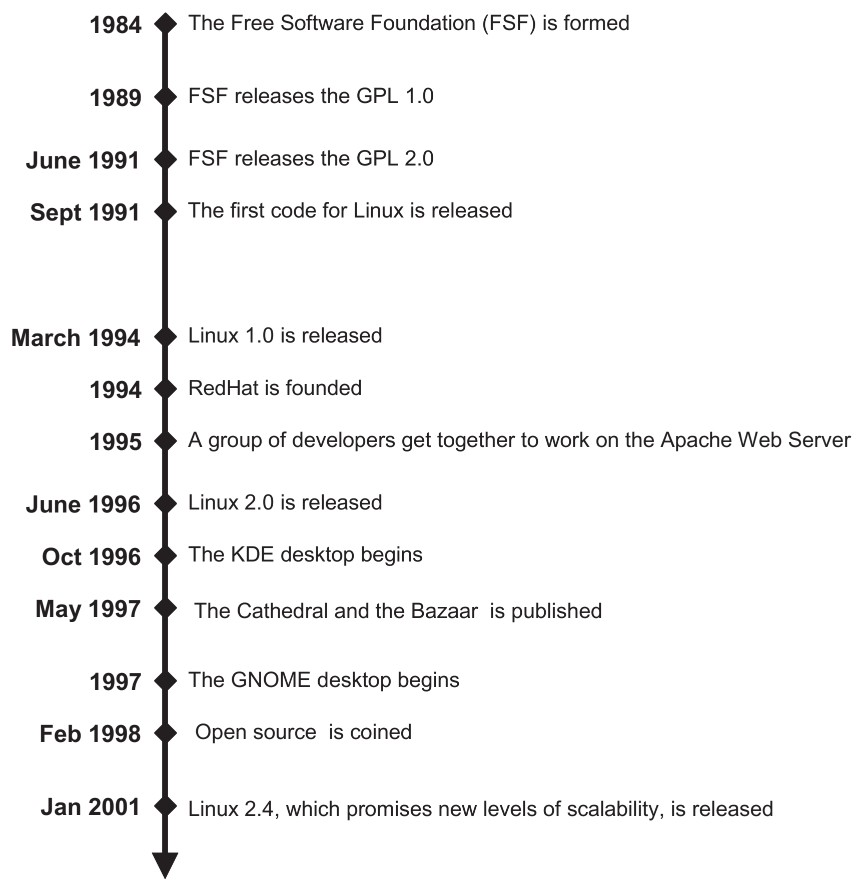
1984 The Free Software Foundation (FSF) is formed
1989 FSF releases the GPL 1.0
June 1991 FSF releases the GPL 2.0
Sept 1991 The first code for Linux is released
March 1994 Linux 1.0 is released
1994 RedHat is founded
1995 A group of developers get together to work on the Apache Web Server
June 1996 Linux 2.0 is released
Oct 1996 The KDE desktop begins
May 1997 The Cathedral and the Bazaar is published
1997 The GNOME desktop begins
Feb 1998 Open source is coined
Jan 2001 Linux 2.4, which promises new levels of scalability, is released

Figure 1–4 A timeline of significant Linux and open source events.

For now, there are a few key names and a few significant events that are useful to know and understand their role and impact within the community.

I can't list everyone who has an important role, and I am sure I will bruise a few egos by not including them in this list. The fact is that there are many I have not listed here who play critical roles in many communities. You will learn later that open source is largely a game of influence and relationships. Once you understand the communities that are important to your business, you will be well-served to invest time and energy in building relationships with the leaders and participants of that community.

- **Linus Torvalds**—Probably the most known personality in the Linux and open source world, Linus is the creator and maintainer of the Linux kernel. He maintains the platform-independent parts of the kernel, and the components of the core kernel that are specific to the Intel IA-32 architecture. Ports of Linux to other architectures are each maintained by separate groups and individuals.
- **Alan Cox**—Until the fall of 2001, Alan was considered Linus' top lieutenant. As you will learn in Chapter 2, Linus maintained the Linux kernel currently in development, while Alan maintained the currently released kernel. As is typical within the community, when Alan decided he wanted to work on other things, he chose a successor to take on his present task. Although Alan is not longer the maintainer of the released Linux kernel, he still commands tremendous influence and respect within the community.
- **Marcelo Tosatti**—Linus and Alan transferred the maintenance torch of the stable Linux kernel (the kernel considered to be in production) to Marcelo. Since Linus must work closely with this individual, he obviously needed to be part of the selection process. Marcelo is a very capable kernel developer who has demonstrated considerable aptitude at working with the community of kernel developers.
- **Dave Miller**—If you submit changes to the networking stacks in the Linux kernel, David is the developer who will likely check your submission and decide if it should be accepted. Think of David as the keeper of Linux networking.
- **Richard Stallman**—The founder of the FSF. Richard is also the creator of the GNU GPL. Richard continues to play an influential role in the free software movement. Many will seek his guidance when clarity is needed with certain areas of the GPL.
- **Bruce Perens**—As the project leader for the Debian project (we will cover Debian in depth later), Bruce authored a "Social Contract" to instill a standard code of conduct when working with free software on the project. When the term "open source" was coined, this "Social Contract" became the Open Source Definition (OSD). Bruce is therefore known as the primary author of the OSD. Chapter 3 outlines this definition in detail.
- **Eric Raymond**—Known as an anthropologist and thought leader within the open source community, Eric is now well-known for his groundbreaking paper, "The Cathedral and the Bazaar," which outlined the open source development process, why it works, and how it can be replicated. Part 3 of this book extends Eric's work to the commercial enterprise.

- **Larry Augustin**—Larry is the CEO of VA Software Corporation. VA Software was originally incorporated as VA Linux Systems. Larry was one of the first to recognize the value of Linux and open source. He created the first hardware company dedicated to supporting Linux. As mainstream hardware vendors such as Dell, IBM, HP, and others began to support Linux, Larry changed his business model to sell software that supports the open source collaborative development process within corporations.
- **Tim O'Reilly**—As the President of O'Reilly Books, Tim continues to have a tremendous influence on the open source movement. The O'Reilly network encompasses key open source technologies such as Perl and others in wide use throughout the Internet. Tim provides a voice for the community by publishing many of their ideas.
- **Brian Behlendorf**—Brian is the co-founder of the Apache Software Foundation. Since Apache has been the open source killer application, Brian has had an influential role not only within the Apache movement, but also the community at large.
- **Michael Tiemann**—Michael founded Cygnus Solutions, the first commercial company based on an open source business model. Cygnus provided the infrastructure for a key component of Linux: compilers. Cygnus pioneered many of the business models based on providing support, services, and other enhancements to free software. Cygnus was acquired by Red Hat in 2000.
- **Bob Young**—Bob is credited with the success of today's most popular Linux distribution, Red Hat. Bob used his marketing savvy to create a powerful brand around the Linux operating system, and in many ways, brought Linux to the masses. Today, Bob is on the board of directors of Red Hat and speaks frequently at Linux and open source events.
- **Jeremy Allison**—Jeremy is the leader of the SAMBA project. As you will discover, SAMBA is key interoperability technology used between the UNIX/Linux and Windows worlds. SAMBA is core technology that is included with most Linux distributions.
- **Miguel de Icaza**—Miguel is a founder and currently CTO of Ximian (formerly Helix Code) Corporation. Miguel led the development of the GNOME desktop environment for many years. Today, Miguel is leading another ground breaking open source project called Mono (Spanish for monkey, the company's mascot). Mono is an open source implementation of a subset of the Microsoft .NET framework.

Summary

You now have a core foundation of knowledge we can build on. You also see that Linux brings many opportunities for you to reduce your IT costs, but that open source presents new business risks that must be managed. With a core understanding of key terms and knowing the major players, we can now start to explore the specifics of how you can take advantage of these benefits and manage the risks within your enterprise.

The rest of Part 1 will expand your understanding of the Linux operating system, and go into the details of open source and the associated licenses. Part 1 ends with an overview of the more significant open source communities that you will likely encounter and need to work with. Part 2 will take a look at the operational side of Linux, encouraging you to consider what you need to deploy in your enterprise, along with providing information about the support, migration, and cost implications of Linux. Part 3 will dig into managing the business risks and opportunities of open source. It will outline not only how to work with open source projects, but also how to implement the development methodology in your environment.