CHAPTER 1

The Ubuntu Story

- The Vision
- Free Software, Open Source, and GNU/Linux
- How the Vision Became Ubuntu
- What Is Ubuntu?
- Ubuntu Promises and Goals
- Sustaining the Vision: Canonical and the Ubuntu Foundation
- Beyond the Vision: Ubuntu Subprojects, Derivatives, and Spin-Offs
- Summary

THIS CHAPTER brings the history of the Ubuntu Project to life and through its history and looks to its future. If you are looking to jump right in and get started with Ubuntu, you can skip this and proceed immediately to Chapter 2, Installing Ubuntu. If you are interested in first learning about where Ubuntu comes from and where it is going, this chapter provides a good introduction.

The Vision

In April 2004 Mark Shuttleworth brought together a dozen developers from the Debian, GNOME, and GNU Arch projects to brainstorm. Shuttleworth asked the developers if a better type of operating system (OS) was possible. Their answer was "Yes." He asked them what it would look like. He asked them to describe the community that would build such an OS. That group worked with Shuttleworth to come up with answers to these questions, and then they decided to try to make the answers a reality. The group named itself the Warthogs and gave itself a six-month deadline to build a proof-of-concept OS. The developers nicknamed their first release the Warty Warthog with the reasonable assumption that their first product would have its warts. Then they got down to business.

It's fulfilling, particularly for those of us who were privileged to be among those early Warthogs, to see the progress made by this project over the years. We had a strong beginning when, far from being warty, the Warty Warthog surpassed our most optimistic expectations and *everyone's* predictions. Within six months, Ubuntu was in the Number 1 spot on several popularity rankings of GNU/Linux distributions. Ubuntu has demonstrated the most explosive growth of any GNU/Linux distribution in recent memory and had one of the most impressive first years and continued growth of any free or open source software project in history.

It is staggering to think that after so few years, *millions* of individuals are using Ubuntu. As many thousands of these users give back to the Ubuntu community by developing documentation, translation, and code, Ubuntu improves every day. As many thousands of these users contribute to a thriving advocacy and support community—both online and in their

local communities—Ubuntu's growth remains unchecked. Ubuntu subprojects, a list of efforts that includes Kubuntu and Ubuntu Studio, along with other projects, are extending the reach and goals of the Ubuntu project into new realms.

Meanwhile, millions of pressed Ubuntu CDs have been shipped at no cost to universities, Internet cafés, computer shops, and community centers around the world. You can find Ubuntu almost anywhere people use computers. The authors have personally seen strangers running Ubuntu on trains in Spain, in libraries in Boston, in museums in Croatia, in high schools in Mexico, across North Africa in Morocco and Tunisia, and in many more places too numerous to list here.

Over the years, Ubuntu has continued to mature. The public took even more notice of Ubuntu beginning with the release of Ubuntu 6.06 LTS, the first polished release with long-term support for both desktops and servers, and followed by a new release every six months and a new LTS release every two years up to the current 12.04 LTS. With these releases, Ubuntu has proven it intends to stick around for the long term while also improving consistently and on a predictable schedule. Even with this maturation, the project maintains its youthful vigor, its ambitious attitude, its commitment to its principles, and its community-driven approach. As the project ages, it is proving that it can learn from its failures as well as its successes and that it can maintain growth without compromising stability.

We've come a long way—and we're still only getting started.

Free Software, Open Source, and GNU/Linux

While thousands of individuals have contributed in some form to Ubuntu, the project has succeeded only through the contributions of many thousands more who have indirectly laid the technical, social, and economic groundwork for Ubuntu's success. While introductions to free software, open source, and GNU/Linux can be found in many other places, no introduction of Ubuntu is complete without a brief discussion of these

concepts and the people and history behind them. It is around these concepts and within these communities that Ubuntu was motivated and born. Ultimately, it is through these ideas that it is sustained.

Free Software and GNU

In a series of events that have almost become legend through constant repetition, Richard M. Stallman created the concept of free software in 1983. Stallman grew up with computers in the 1960s and 1970s, when computer users purchased very large and extremely expensive mainframe computers, which were then shared among large numbers of programmers. Software was, for the most part, seen as an add-on to the hardware, and every user had the ability and the right to modify or rewrite the software on his or her computer and to freely share this software. As computers became cheaper and more numerous in the late 1970s, producers of software began to see value in the software itself. Producers of computers began to argue that their software could be copyrighted and was a form of intellectual property much like a music recording, a film, or a book's text. They began to distribute their software under licenses and in forms that restricted its users' abilities to use, redistribute, or modify the code. By the early 1980s, restrictive software licenses had become the norm.

Stallman, then a programmer at MIT's Artificial Intelligence Laboratory, became increasingly concerned with what he saw as a dangerous loss of the freedoms that software users and developers had previously enjoyed. He was concerned with computer users' ability to be good neighbors and members of what he thought was an ethical and efficient computer-user community. To fight against this negative tide, Stallman articulated a vision for a community that developed liberated code—in his words, "free software." He defined free software as software that had the following four characteristics—labeled as freedoms 0 through 3 instead of 1 through 4 as a nod to computer programming tradition and a bit of an inside joke:

- The freedom to run the program for any purpose (freedom 0)
- The freedom to study how the program works and adapt it to your needs (freedom 1)

- The freedom to redistribute copies so you can help your neighbor (freedom 2)
- The freedom to improve the program and release your improvements to the public so that the whole community benefits (freedom 3)

Access to source code—the human-readable and modifiable blueprints of any piece of software that can be distinguished from the computer-readable version of the code that most software is distributed as—is a prerequisite to freedoms 1 and 3. In addition to releasing this definition of free software, Stallman created a project with the goal of creating a completely free OS to replace the then-popular UNIX. In 1984, Stallman announced this project and called it GNU—also in the form of common programmer humor, a recursive acronym that stands for "GNU's Not UNIX."

Linux

By the early 1990s, Stallman and a collection of other programmers working on GNU had developed a near-complete OS that could be freely shared. They were, however, missing a final essential piece in the form of a kernel—a complex system command processor that lies at the center of any OS. In 1991, Linus Torvalds wrote an early version of just such a kernel, released it under a free license, and called it Linux. Linus's kernel was paired with the GNU project's development tools and OS and with the graphical windowing system called X. With this pairing, a completely free OS was born—free both in terms of price and in Stallman's terms of freedom.

All systems referred to as Linux today are, in fact, built on the work of this collaboration. Technically, the term *Linux* refers only to the kernel. Many programmers and contributors to GNU, including Stallman, argue emphatically that the full OS should be referred to as GNU/Linux in order to give credit not only to Linux but also to the GNU project and to highlight GNU's goals of spreading software freedom—goals not necessarily shared by Linus Torvalds. Many others find this name cumbersome and prefer calling the system simply Linux. Yet others, such as those working on the Ubuntu project, attempt to avoid the controversy altogether by referring to GNU/Linux only by using their own project's name.

Open Source

Disagreements over labeling did not end with discussions about the naming of the combination of GNU and Linux. In fact, as the list of contributors to GNU and Linux grew, a vibrant world of new free software projects sprouted up, facilitated in part by growing access to the Internet. As this community grew and diversified, a number of people began to notice an unintentional side effect of Stallman's free software. Because free software was built in an open way, *anyone* could contribute to software by looking through the code, finding bugs, and fixing them. Because software ended up being examined by larger numbers of programmers, free software was higher in quality, performed better, and offered more features than similar software developed through proprietary development mechanisms. It turned out that in many situations, the development model behind free software led to software that was *inherently better* than proprietary alternatives.

As the computer and information technology industry began to move into the dot-com boom, one group of free software developers and leaders, spearheaded by two free software developers and advocates—Eric S. Raymond and Bruce Perens—saw the important business proposition offered by a model that could harness volunteer labor or interbusiness collaboration and create intrinsically better software. However, they worried that the term *free software* was problematic for at least two reasons. First, it was highly ambiguous—the English word *free* means both gratis, or at no cost (e.g., free as in free beer), and liberated in the sense of freedom (e.g., free as in free speech). Second, there was a feeling, articulated most famously by Raymond, that all this talk of freedom was scaring off the very business executives and decision makers whom the free software movement needed to impress in order to succeed.

To tackle both of these problems, this group coined a new phrase—open source—and created a new organization called the Open Source Initiative. The group set at its core a definition of open source software that overlapped completely and exclusively with both Stallman's four-part definition of free software and with other community definitions that were also based on Stallman's.

One useful way to understand the split between the free software and open source movements is to think of it as the opposite of a schism. In religious

schisms, churches separate and do not work or worship together because of relatively small differences in belief, interpretation, or motivation. For example, most contemporary forms of Protestant Christianity agree on *almost* everything but have separated over some small but irreconcilable differences. However, in the case of the free software and open source movements, the two groups have fundamental disagreements about their motivation and beliefs. One group is focused on freedom, while the other is focused on pragmatics. Free software is most accurately described as a social movement, while open source is a development methodology. However, the two groups have no trouble working on projects hand in hand.

In terms of the motivations and goals, open source and free software diverge greatly. Yet in terms of the software, the projects, and the licenses they use, they are completely synonymous. While people who identify with either group see the two movements as being at odds, the Ubuntu project sees no conflict between the two ideologies. People in the Ubuntu project identify with either group and often with both. In this book, we may switch back and forth between the terms as different projects, and many people in the Ubuntu community identify more strongly with one term or the other. For the purposes of this book, though, either term should be read as implying the other unless it is stated otherwise.

How the Vision Became Ubuntu

There was a time when writing a history of Ubuntu may have seemed premature; however, that is no longer the case, as the last several years have been busy ones for Ubuntu. With its explosive growth, it is difficult even for those involved most closely with the project to track and record some of the high points. Importantly, there are some key figures whose own history must be given to fully understand Ubuntu. This brief summary provides some of the high points of Ubuntu's history to date and the necessary background knowledge to understand where Ubuntu comes from.

Mark Shuttleworth

No history of Ubuntu can call itself complete without a history of Mark Shuttleworth. Shuttleworth is, undeniably, the most visible and important person in Ubuntu. More important from the point of view of history,

Shuttleworth is also the originator and initiator of the project—he made the snowball that would eventually roll on and grow to become the Ubuntu project.

Shuttleworth was born in 1973 in Welkom, Free State, in South Africa. He attended Diocesan College and obtained a business science degree in finance and information systems at the University of Cape Town. During this period, he was an avid computer hobbyist and became involved with the free and open source software community. He was at least marginally involved in both the Apache project and the Debian project and was the first person to upload the Apache Web server, perhaps the single most important piece of server software on GNU/Linux platforms, into the Debian project's archives.

Seeing an opportunity in the early days of the Web, Shuttleworth founded a certificate authority and Internet security company called Thawte in his garage. Over the course of several years, he built Thawte into the second largest certificate authority on the Internet, trailing only the security behemoth VeriSign. Throughout this period, Thawte's products and services were built and served almost entirely from free and open source software. In December 1999, Shuttleworth sold Thawte to VeriSign for an undisclosed amount that reached into the hundreds of millions in U.S. dollars.

With his fortune made at a young age, Shuttleworth might have enjoyed a life of leisure—and probably considered it. Instead, he decided to pursue his lifelong dream of space travel. After paying approximately US$20 million to the Russian space program and devoting nearly a year to preparation, including learning Russian and spending seven months training in Star City, Russia, Shuttleworth realized his dream as a civilian cosmonaut aboard the Russian Soyuz TM-34 mission. On this mission, Shuttleworth spent two days on the Soyuz rocket and eight days on the International Space Station, where he participated in experiments related to AIDS and genome research. In early May 2002, Shuttleworth returned to Earth.

In addition to space exploration and an only slightly less impressive jaunt to Antarctica, Shuttleworth has played an active role as both a philanthropist and a venture capitalist. In 2001, Shuttleworth founded The Shuttleworth Foundation (TSF)—a nonprofit organization based in South Africa. The

foundation was chartered to fund, develop, and drive social innovation in the field of education. Of course, the means by which TSF attempts to achieve these goals frequently involves free software. Through these projects, the organization has been one of the most visible proponents of free and open source software in South Africa and even the world. In the venture capital area, Shuttleworth worked to foster research, development, and entrepreneurship in South Africa with strategic injections of cash into start-ups through a new venture capital firm called HBD, an acronym for "Here Be Dragons." During this period, Shuttleworth was busy brainstorming his next big project—the project that would eventually become Ubuntu.

The Warthogs

There has been no lack of projects attempting to wrap GNU, Linux, and other pieces of free and open source software into a neat, workable, and user-friendly package. Mark Shuttleworth, like many other people, believed that the philosophical and pragmatic benefits offered by free software put it on a course for widespread success. While each had its strengths, none of the offerings were particularly impressive as a whole. Something was missing from each of them. Shuttleworth saw this as an opportunity. If someone could build *the* great free software distribution that helped push GNU/Linux into the mainstream, he would come to occupy a position of strategic importance.

Shuttleworth, like many other technically inclined people, was a huge fan of the Debian project (discussed in depth later in this chapter). However, many things about Debian did not fit with Shuttleworth's vision of an ideal OS. For a period of time, Shuttleworth considered the possibility of running for Debian project leader as a means of reforming the Debian project from within. With time, though, it became clear that the best way to bring GNU/Linux to the mainstream would not be from within the Debian project—which in many situations had very good reasons for being the way it was. Instead, Shuttleworth would create a new project that worked in symbiosis with Debian to build a new, better GNU/Linux system.

To kick off this project, Shuttleworth invited a dozen or so free and open source software developers he knew and respected to his flat in London in April 2004. It was in this meeting (alluded to in the first paragraphs of this

introduction) that the groundwork for the Ubuntu project was laid. By that point, many of those involved were excited about the possibility of the project. During this meeting, the members of the team—which would in time grow into the core Ubuntu team—brainstormed a large list of the things that *they* would want to see in their ideal OS. The list is now a familiar list of features to most Ubuntu users. Many of these traits are covered in more depth later in this chapter. The group wanted

- Predictable and frequent release cycles
- A strong focus on localization and accessibility
- A strong focus on ease of use and user-friendliness on the desktop
- A strong focus on Python as the single programming language through which the entire system could be built and expanded
- A community-driven approach that worked with existing free software projects and a method by which the groups could give back as they went along—not just at the time of release
- A new set of tools designed around the process of building distributions that allowed developers to work within an ecosystem of different projects and that allowed users to give back in whatever way they could

There was consensus among the group that actions speak louder than words, so there were no public announcements or press releases. Instead, the group set a deadline for itself—six short months in the future. Shuttleworth agreed to finance the work and pay the developers full-time salaries to work on the project. After six months, they would both announce their project and reveal the first product of their work. They made a list of goals they wanted to achieve by the deadline, and the individuals present took on tasks. Collectively, they called themselves the Warthogs.

What Does *Ubuntu* Mean?

At this point, the Warthogs had a great team, a set of goals, and a decent idea of how to achieve most of them. The team did not, on the other hand, have a name for the project. Shuttleworth argued strongly that they should call the project Ubuntu.

Ubuntu is a concept and a term from several South African languages, including Zulu and Xhosa. It refers to a South African ideology or ethic that, while difficult to express in English, might roughly be translated as "humanity toward others," or "I am what I am because of who we all are." Others have described Ubuntu as "the belief in a universal bond of sharing that connects all humanity." The famous South African human rights champion Archbishop Desmond Tutu explained Ubuntu in this way:

> A person with Ubuntu is open and available to others, affirming of others, does not feel threatened that others are able and good, for he or she has a proper self-assurance that comes from knowing that he or she belongs in a greater whole and is diminished when others are humiliated or diminished, when others are tortured or oppressed.

Ubuntu played an important role as a founding principle in post-apartheid South Africa and remains a concept familiar to most South Africans today.

Shuttleworth liked *Ubuntu* as a name for the new project for several reasons. First, it is a South African concept. While the majority of the people who work on Ubuntu are not from South Africa, the roots of the project are, and Shuttleworth wanted to choose a name that represented this. Second, the project emphasizes relationships with others and provides a framework for a profound type of community and sharing—exactly the attitudes of sharing, community, and collaboration that are at the core of free software. The term represented the side of free software that the team wanted to share with the world. Third, the idea of personal relationships built on mutual respect and connections describes the fundamental ground rules for the highly functional community that the Ubuntu team wanted to build. *Ubuntu* was a term that encapsulated where the project came from, where the project was going, and how the project planned to get there. The name was perfect. It stuck.

Beyond the Vision

In order to pay developers to work on Ubuntu full time, Shuttleworth needed a company to employ them. He wanted to pick some of the best people for the jobs from within the global free and open source communities. These communities, inconveniently for Shuttleworth, know no national

and geographic boundaries. Rather than move everyone to a single locale and office, Shuttleworth made the decision to employ these developers through a virtual company. While this had obvious drawbacks in the form of high-latency and low-bandwidth connections, different time zones, and much more, it also introduced some major benefits in the particular context of the project. On one hand, the distributed nature of employees meant that the new company could hire individuals without requiring them to pack up their lives and move to a new country. More important, it meant that *everyone* in the company was dependent on IRC, mailing lists, and online communication mechanisms to do their work. This unintentionally and automatically solved the water-cooler problem that plagued many other corporately funded free software projects—namely, that developers would casually speak about their work in person, and cut the community and anyone else who didn't work in the office out of the conversation completely. For the first year, the closest thing that Canonical had to an office was Shuttleworth's flat in London. While the company has grown and now has several offices around the world, it remains distributed and a large number of the engineers work from home. The group remains highly dependent on Internet collaboration.

With time, the company was named Canonical. The name was a nod to the project's optimistic goals of becoming the canonical place for services and support for free and open source software and for Ubuntu in particular. *Canonical*, of course, refers to something that is accepted as authoritative. It is a common word in the computer programmer lexicon. It's important to note that being canonical is like being standard; it is not coercive. Unlike holding a monopoly, becoming the canonical location for something implies a similar sort of success—but *never* one that cannot be undone, and *never* one that is exclusive. Other companies will support Ubuntu and build operating systems based on it, but as long as Canonical is doing a good job, its role will remain central.

What Is Ubuntu?

The Warthogs' goal and Canonical's flagship project is Ubuntu. Ubuntu is a free and open source GNU/Linux distribution and operating system. More information on what a distribution is and how other distributions have played a role in vision that is now Ubuntu can be found in Chapter 7.

What Is a Distribution?

It's clear to most people that Ubuntu is an OS. The full story is a little more complex. Ubuntu is what is called a distribution of GNU/Linux—a *distro* for short. Understanding exactly what that means requires, once again, a little bit of history. In the early days of GNU and Linux, users needed a great deal of technical knowledge. Only geeks needed to apply. There were no Linux operating systems in the sense that we usually use the term—there was no single CD or set of disks that one could use to install. Instead, the software was dozens and even hundreds of individual programs, each built differently by a different individual, and each distributed separately. Installing each of the necessary applications would be incredibly time consuming at best. In many cases, incompatibilities and the technical trickery necessary to install software made getting a GNU/Linux system on a hard disk prohibitively difficult. A great deal of knowledge of configuration and programming was necessary just to get a system up and running. As a result, very few people who were not programmers used these early GNU/Linux systems.

Early distributions were projects that collected all of the necessary pieces of software from all of the different places and put them together in an easier-to-install form with the most basic configuration already done. These distributions aimed to make using GNU/Linux more convenient and to bring it to larger groups of users. Today, almost nobody uses GNU/Linux without using a distribution. As a result, distribution names are well known. Ubuntu is such a project. Other popular distros include Red Hat and Fedora, Novell's SUSE, Gentoo, and of course Debian.

Most distributions contain a similar collection of software. For example, they all contain most of the core pieces of GNU and a Linux kernel. Almost all contain the X Window System and a set of applications on top of it that may include a Web browser, a desktop environment, and an office suite. While distributions started out distributing only the core pieces of the OS, they have grown to include an increasingly wide array of applications as well. A modern distribution includes all of the software that "comes with an OS," that is, several CDs or DVDs containing anything that most users might want and that the distribution is legally allowed to distribute.

Ubuntu, like other contemporary distros, offers a custom installer, a framework including software and servers to install new software once the system has been installed, a standard configuration method through which many programs can be configured, a standard method through which users can report bugs in their software, and much more. Frequently, distributions also contain large repositories of software on servers accessible through the Internet. To get a sense of scale, Ubuntu includes more than 30,000 pieces of software on its central servers—each piece of software is customized slightly and tested to work well with all of the other software on the system. That number grows daily.

What's important to realize is that the creators of distributions do not, for the most part, write or create the applications you use. The Ubuntu team did not write Linux, and it did not write GNU—although individuals on the team have contributed to both projects. Instead, the Ubuntu team takes GNU, Linux, and many thousands of other applications and then tests and integrates them to be accessible under a single installer. Ubuntu is the glue that lets you take a single CD, install hundreds of separate pieces of software, and have them work together as a single, integrated desktop system. If you were to pick up a CD of another distribution such as Debian, Red Hat, or Novell, the software installed would be nearly identical to the software in Ubuntu. The difference would be in the way the software is installed, serviced, upgraded, and presented and the way it integrates with other pieces of software on the system.

An Ecosystem of Distributions

Many hundreds of GNU/Linux distributions are in active use today. A quick look at Distrowatch's database (distrowatch.com) demonstrates the staggering number and growth of distributions. One of the first GNU/Linux distributions was called Softlanding Linux System, or SLS. For a number of reasons, a programmer named Patrick Volkerding thought he could improve on SLS. Because SLS was free software, Volkerding had the freedom to make a derivative version of SLS and distribute it. Volkerding did just this when he took SLS's code and used it as the framework or model upon which to create his own variant called Slackware. Subsequently, Slackware became the first widely successful GNU/Linux distribution and is maintained to this day.

With time, the landscape of GNU/Linux distribution has changed. However, the important role of derivation that made Slackware possible has remained fully intact and is still shaping this landscape. Today, the hundreds of GNU/Linux distributions serve a multitude of users for a myriad of purposes: There are distributions specially designed for children, for dentists, and for speakers of many of the world's languages. There are distributions for science, for business, for servers, for PDAs, for nonprofit organizations, for musicians, and for countless other groups.

Despite this diversity, the vast majority of derivatives can be traced back to one of two parent distributions: Red Hat and Debian. While it is not necessary to understand the details of how these projects differ, it's useful to know that Red Hat and Debian offer two compelling, but frequently different, platforms. Each project has strengths and weaknesses. For almost every group making a Linux-based OS, one of these projects acts as square one (with a few notable exceptions, such as the Gentoo project).

However, while the process of deriving distributions has allowed for a proliferation of OS platforms serving a vast multiplicity of needs, the derivative process has, historically, been largely a one-way process. New distributions based on Red Hat—CentOS and SUSE, for example—begin with Red Hat or a subset of Red Hat technology and then customize and diverge. Very few of these changes ever make it back into Red Hat and, with time, distributions tend to diverge to the point of irreconcilable incompatibility. While the software that each system includes remains largely consistent across all distributions, the way that it is packaged, presented, installed, and configured becomes increasingly differentiated. During this process, interdistribution sharing and collaboration grow in difficulty.

This growing divergence indicates a more general problem faced by distribution teams in getting changes upstream. Frequently, the users of GNU/Linux distributions find and report problems in their software. Frequently, distribution teams fix the bugs in question. While sometimes these bugs are in changes introduced by the distribution, they often exist in the upstream version of the software and the fix applies to *every* distribution. What is not uncommon, but is unfortunately *much* less frequent, is for these bug fixes to be pushed upstream so that all distributions and users get to use them. This lack of collaboration is rarely due to malice,

incompetence, or any tactical or strategic decision made by developers or their employers. Instead, tracking and monitoring changes *across* distributions and in relation to upstream developers is complicated and difficult. It's a fact of life that sometimes changes fall on the floor. These failures are simply the product of distribution-building processes, policies, and tools that approach distributions as products in and of themselves—not processes within an ecosystem.

Like many other distributions, Ubuntu is a derivative of Debian. Unlike the creators of many derivatives, the Ubuntu community has made it one of its primary goals to explore the possibility of a better derivation process with Debian, with Debian and Ubuntu's common upstreams (e.g., projects such as Linux or GNU), and with Ubuntu's *own* derivatives. A more in-depth discussion of Debian can help explain how Ubuntu positions itself within the free software world.

The Debian Project and the Free Software Universe

Debian is a distribution backed by a volunteer project of many hundreds of official members and many more volunteers and contributors. It has expanded to encompass over 30,000 packages of free and open source applications and documentation. Debian's history and structure make it very good at certain things. For example, Debian has a well-deserved reputation for integrated package management and access to a large list of free software applications. However, as a voluntary and largely nonhierarchical organization, Debian had a challenging time providing frequent and reliable releases, corporate support and liability, and a top-down consistency.

Each new distribution exists for a reason. Creating a new distribution, even a derivative, is far from easy. In large part, Ubuntu exists to build off of the many successes of the Debian project while solving some of the problems it struggles with. The goal is to create a synthetic whole that appeals to users who had previously been unable or unwilling to use Debian.

In building off the great work of the Debian project, as well as GNU, Linux, and other projects that Debian is built on, the Ubuntu team wanted to explore a new style of derivation that focused on a tighter interproject relationship within an ecosystem of different developers. While Ubuntu

tries to improve and build on Debian's success, the project is in no way trying to replace Debian. On the contrary, Ubuntu couldn't exist without the Debian project and its large volunteer and software base, as well as the high degree of quality that Debian consistently provides. This symbiotic relationship between Ubuntu and Debian is mirrored in the way that both Ubuntu and Debian depend heavily on projects such as GNU and Linux to produce great software, which they can each package and distribute. The Ubuntu project sets out explicitly to build a symbiotic relationship with both Debian and their common "upstream."

The relationship between Ubuntu and Debian has not been simple, straightforward, or painless and has involved patience and learning on both sides. While the relationship has yet to be perfected, with time it has improved consistently, and both groups have found ways to work together that seem to offer major benefits over the traditional derive-and-forget model. It is through a complex series of technological, social, and even political processes—many of which are described in the rest of this chapter—that Ubuntu tries to create a better way to build a free software distribution.

The Ubuntu Community

Ubuntu would not be what it is today without the Ubuntu community. Even the definition of *ubuntu* is one that revolves around people interacting in a community. The Ubuntu community and how it is organized as well as how those who choose can get involved will be discussed in detail in Chapter 10.

Ubuntu Promises and Goals

So far, this book has been about the prehistory, history, and context of the Ubuntu project. After this chapter, the book focuses on the distribution itself. Before proceeding, it's important to understand the goals that motivated the project.

Philosophical Goals

The most important goals of the Ubuntu project are philosophical in nature. The Ubuntu project lays out its philosophy in a series of documents

on its Web site. In the most central of these documents, the team summarizes the charter and the major philosophical goals and underpinnings.

- **Our philosophy**

  Our work is driven by a philosophy of software freedom that aims to spread and bring the benefits of software to all parts of the world. At the core of the Ubuntu Philosophy are these core ideals:

  1. Every computer user should have the freedom to download, run, copy, distribute, study, share, change and improve their software for any purpose, without paying licensing fees.
  2. Every computer user should be able to use their software in the language of their choice.
  3. Every computer user should be given every opportunity to use software, even if they work under a disability.

  Our philosophy is reflected in the software we produce and included in our distribution. As a result, the licensing terms of the software we distribute are measured against our philosophy, using the Ubuntu License Policy.

  When you install Ubuntu, almost all of the software installed already meets these ideals, and we are working to ensure that every single piece of software you need is available under a license that gives you those freedoms.

  Currently, we make a specific exception for some "drivers" that are available only in binary form, without which many computers will not complete the Ubuntu installation. We place these in a restricted section of your system, which makes them easy to remove if you do not need them.

- **Free software**

  For Ubuntu, the "free" in free software is used primarily in reference to freedom and not to price—although we are committed to not charging for Ubuntu. The most important thing about Ubuntu is that it confers rights of software freedom on the people who install and use it. These freedoms enable the Ubuntu community to grow and to continue to share its collective experience and expertise to improve Ubuntu and make it suitable for use in new countries and new industries.

  Quoting the Free Software Foundation's "What Is Free Software?" the freedoms at the core of free software are defined as

  - The freedom to run the program for any purpose
  - The freedom to study how the program works and adapt it to your needs

- The freedom to redistribute copies so you can help others
- The freedom to improve the program and release your improvements to the public so that everyone benefits

- **Open source**

  Open source is a term coined in 1998 to remove the ambiguity in the English word *free*. The Open Source Initiative described open source software in the Open Source Definition. Open source continues to enjoy growing success and wide recognition.

  Ubuntu is happy to call itself open source. While some refer to free and open source as competing movements with different ends, we do not see free and open source software as either distinct or incompatible. Ubuntu proudly includes members who identify with both movements.

Here, the Ubuntu project makes explicit its goals that every user of software should have the freedoms required by free software. This is important for a number of reasons. First, it offers users all of the practical benefits of software that runs better, faster, and more flexibly. More important, it gives every user the capability to transcend his or her role as a user and a consumer of software. Ubuntu wants software to be empowering and to work in the ways that users want it to work. Ubuntu wants all users to have the ability to make sure it works for them. To do this, software *must* be free, so Ubuntu makes this a requirement and a philosophical promise.

Of course, the core goals of Ubuntu do not end with the free software definition. Instead, the project articulates two new but equally important goals. The first of these, that all computer users should be able to use their computers in their chosen languages, is a nod to the fact that the majority of the world's population does not speak English, while the vast majority of software interacts only in that language. To be useful, source code comments, programming languages, documentation, and the texts and menus in computer programs must be written in *some* language. Arguably, the world's most international language is a reasonably good choice. However, there is no language that everyone speaks, and English is not useful to the majority of the world's population that does not speak it. A computer can be a great tool for empowerment and education, but *only* if the user can understand the words in the computer's interface. As a result, Ubuntu believes that it is the project's—and community's—responsibility to

ensure that *every* user can easily use Ubuntu to read and write in the language with which he or she is most comfortable.

The ability to make modifications—a requirement of free software and of Ubuntu's first philosophical point—makes this type of translation possible. This book is a case in point. While it helps explain Ubuntu only to the relatively small subset of the world that already speaks English, the choice to write this book in English was made to enable it to have the widest impact. More important, it is distributed under a Creative Commons license that allows for translation, modification, and redistribution. The authors of this book cannot write this book in all of the world's languages—or even more than one of them. Instead, we have attempted to eliminate unnecessary legal restrictions and other barriers that might keep the community from taking on the translation work. As a result, the complete text of the several editions have been translated into other languages like German, Japanese, Polish, and Spanish.

Finally, just as no person should be blocked from using a computer simply because he or she does not know a particular language, no user should be blocked from using a computer because of a disability. Ubuntu must be accessible to users with motor disabilities, vision disabilities, and hearing disabilities. It should provide input and output in a variety of forms to account for each of these situations and for others. A significant percentage of the world's most intelligent and creative individuals also have disabilities. Ubuntu's usefulness should not be limited when it can be inclusive. More important, Ubuntu wants to welcome and to be able to harness the ability of these individuals as community members to build a better and more effective community.

Conduct Goals and Code of Conduct

If Ubuntu's philosophical commitments describe the *why* of the Ubuntu project, the Code of Conduct (CoC) describes Ubuntu's *how*. Ubuntu's CoC is, arguably, the most important document in the day-to-day operation of the Ubuntu community and sets the ground rules for work and cooperation within the project. Explicit agreement to the document is the only criterion for becoming an officially recognized Ubuntu activist—an Ubuntero—and is an essential step toward membership in the project.

Signing the Ubuntu Code of Conduct and becoming an Ubuntu member is described in more depth in Chapter 10.

The CoC covers "behavior as a member of the Ubuntu community, in any forum, mailing list, wiki, Web site, IRC channel, install-fest, public meeting, or private correspondence." The CoC goes into some degree of depth on a series of points that fall under the following headings.

- Be considerate.
- Be respectful.
- Be collaborative.
- When you disagree, consult others.
- When you are unsure, ask for help.
- Step down considerately.

Many of these headings seem like common sense or common courtesy to many, and that is by design. Nothing in the CoC is controversial or radical, and it was never designed to be.

More difficult is that nothing is easy to enforce or decide because acting considerately, respectfully, and collaboratively is often very subjective. There is room for honest disagreements and occasional hurt feelings. These are accepted shortcomings. The CoC was not designed to be a law with explicit prohibitions on phrases, language, or actions. Instead, it aims to provide a constitution and a reminder that considerate and respectful discussion is *essential* to the health and vitality of the project. In situations where there is a serious disagreement on whether a community member has violated or is violating the code, the Community Council—a body that is discussed in depth in Chapter 10—is available to arbitrate disputes and decide what action, if any, is appropriate.

Nobody involved in the Ubuntu project, including Mark Shuttleworth and the other members of the Community Council, is above the CoC. The CoC is *never* optional and *never* waived. In fact, the Ubuntu community has also created a Leadership Code of Conduct (LCoC), which extends and expands on the CoC and describes additional requirements and expectations for

those in leadership positions in the community. Of course, in no way was either code designed to eliminate conflict or disagreement. Arguments are at least as common in Ubuntu as they are in other projects and online communities. However, there is a common understanding within the project that arguments should happen in an environment of collaboration and mutual respect. This allows for *better* arguments with *better* results—and with less hurt feelings and fewer bruised egos.

While they have been sometimes incorrectly used as such, the CoC and LCoC are not sticks to be wielded against an opponent in an argument. Instead, they are useful points of reference upon which we can assume consensus within the Ubuntu community. Much more frequently, if a group in the community feels a member is acting in a way that is out of line with the code, the group will gently remind the community member, often privately, that the CoC is in effect. In almost all situations, this is enough to avoid any further action or conflict. Very few CoC violations are ever brought before the Community Council.

Technical Goals

While a respectful community and adherence to a set of philosophical goals provide an important frame in which the Ubuntu project works, Ubuntu is, at the end of the day, a technical project. As a result, it only makes sense that in addition to philosophical goals and a project constitution, Ubuntu also has a set of technical goals.

The first technical goal of the project, and perhaps the most important one, is the coordination of regular and predictable releases. In April 2004, at the Warthogs meeting, the project set a goal for its initial proof-of-concept release six months out. In part due to the resounding success of that project, and in larger part due to the GNOME release schedule, the team has stuck to a regular and predictable six-month release cycle and has only once chosen to extend the release schedule—by six weeks for the first LTS release to ensure it was done right—and only then after obtaining community consensus on the decision. The team then doubled its efforts and made the next release in a mere four and a half months, putting its release schedule back on track. Frequent releases are important because users can then use the latest and greatest free software available—something that is

essential in a development environment as vibrant and rapidly changing and improving as the free software community. Predictable releases are important, especially to businesses, because it means that they can organize their business plans around Ubuntu. Through consistent releases, Ubuntu can provide a platform that businesses and derivative distributions can rely upon to grow and build.

While releasing frequently and reliably is important, the released software must then be supported. Ubuntu, like all distributions, must deal with the fact that all software has bugs. Most bugs are minor, but fixing them may introduce even worse issues. Therefore, fixing bugs after a release must be done carefully or not at all. The Ubuntu project engages in major changes, including bug fixes, between releases only when the changes can be extensively tested. However, some bugs risk the loss of users' information or pose a serious security vulnerability. These bugs are fixed immediately and made available as updates for the released distribution. The Ubuntu community works hard to find and minimize all types of bugs before releases and is largely successful in squashing the worst. However, because there is always the possibility that more of these bugs will be found, Ubuntu commits to supporting every release for 18 months after it is released. In the case of LTS releases such as the original LTS, Ubuntu 6.06 LTS, released in 2006, the project went well beyond even this and committed to support the release for three full years on desktop computers and for five years in a server configuration. This proved so popular with businesses, institutions, and the users of Ubuntu servers that in 2008 and 2010, Ubuntu 8.04 LTS and 10.04 LTS were released with similar three- and five-year desktop and server extended support commitments. The most recent release, Ubuntu 12.04 LTS, continues the pattern.

This bipartite approach to servers and desktops implies the third major technical commitment of the Ubuntu project: support for both servers and desktop computers in separate but equally emphasized modes. While Ubuntu continues to be more well known, and perhaps more popular, in desktop configurations, there exist teams of Ubuntu developers focused both on server and desktop users. The Ubuntu project believes that both desktops and servers are essential and provides installation methods on every CD for both types of systems. Ubuntu provides tested and supported software appropriate to the most common actions in both environments

and documentation for each. This book contains information on running Ubuntu both on the desktop and on a server. The release of 6.06 LTS with long-term support successfully helped pave the way for reliable long-term server support for Ubuntu and helped grow the now-vibrant Ubuntu server community. The 8.04 LTS release repeated this success with a more up-to-date platform, then 10.04 LTS, and now 12.04 LTS.

Finally, the Ubuntu project is committed to making it as easy as possible for users to transcend their role as consumers and users of software and to take advantage of each of the freedoms central to our philosophy. As a result, Ubuntu has tried to focus its development around the use and promotion of a single programming language, Python. The project has worked to ensure that Python is widely used throughout the system. By ensuring that desktop applications, text-based or console applications, and many of the "guts" of the system are written in or extensible in Python, Ubuntu is working to ensure that users need learn only one language in order to take advantage of, automate, and tweak many parts of their computer systems.

Bug #1

Of course, Ubuntu's goals are not only to build an OS that lives up to our philosophy or technical goals and to do it on our terms—although we probably would be happy if we achieved only that. Our *ultimate goal*, the one that supersedes and influences all others, is to spread our great software, our frequent releases, and the freedoms enshrined in our philosophy to as many computer users in as many countries as possible. Ubuntu's ultimate goal is not to become the most used *GNU/Linux distribution* in the world; it is to become the most widely used *OS* in the world.

The first bug recorded for Ubuntu illustrates this fact. The bug, filed by Shuttleworth and marked as severity critical, remains open today and can be viewed online at https://launchpad.net/distros/ubuntu/+bug/1. The text of the bug reads as follows.

> Microsoft has a majority market share | Non-free software is holding back innovation in the IT industry, restricting access to IT to a small part of the world's population and limiting the ability of software developers to reach their full potential, globally. This bug is widely evident in the PC industry.

Steps to repeat:

1. Visit a local PC store.

What happens:

2. Observe that a majority of PCs for sale have non-free software preinstalled.
3. Observe very few PCs with Ubuntu and free software preinstalled.

What should happen:

1. A majority of the PCs for sale should include only free software such as Ubuntu.
2. Ubuntu should be marketed in a way such that its amazing features and benefits would be apparent and known by all.
3. The system shall become more and more user friendly as time passes.

Many have described Ubuntu's success in the last several years as amazing. For a new GNU/Linux distribution, the level and speed of success have been unprecedented. During this period, Ubuntu has lived up to both its philosophical and technical commitments, achieved many of its goals, and built a vibrant community of users and contributors who have accomplished monumental amounts while collaborating in a culture of respect and understanding fully in line with the Ubuntu Code of Conduct. However, Bug #1 demonstrates that the Ubuntu project will be declared a complete success only when Ubuntu's standards of freedom, technical excellence, and conduct are the norm *everywhere* in the software world.

Sustaining the Vision: Canonical and the Ubuntu Foundation

While Ubuntu is driven by a community, several groups play an important role in its structure and organization. Foremost among these are Canonical, Ltd., a for-profit company introduced as part of the Ubuntu history description, and the Ubuntu Foundation, which is introduced later in this section.

Canonical, Ltd.

Canonical, Ltd. is a company founded by Mark Shuttleworth with the primary goal of developing and supporting the Ubuntu distribution. Many of the core developers on Ubuntu—although no longer a majority of

them—work full time or part time as employees of Canonical, Ltd. This funding by Canonical allows Ubuntu to make the type of support commitments that it does. Ubuntu can claim that it will release in six months because releasing, in one form or another, is something that the paid workers at Canonical can ensure. As an all-volunteer organization, Debian suffered from an inability to set and meet deadlines—volunteers become busy or have other deadlines in their paying jobs that take precedence. By offering paying jobs to a subset of developers, Canonical can set support and release deadlines and ensure that they are met.

In this way, Canonical ensures that Ubuntu's bottom-line commitments are kept. Of course, Canonical does not fund all Ubuntu work, nor could it. Canonical can release *a distribution* every six months, but that distribution will be made *much* better and more usable through contributions from the community of users. Most features, most new pieces of software, almost all translations, almost all documentation, and much more are created outside of Canonical. Instead, Canonical ensures that deadlines are met and that the essential work, regardless of whether it's fun, gets done.

Canonical, Ltd. was incorporated on the Isle of Man—a tiny island nation between Wales and Ireland that is mostly well known as a haven for international businesses. Since Canonical's staff is sprinkled across the globe and no proper office is necessary, the Isle of Man seemed like as good a place as any for the company to hang its sign.

In early 2010, Mark Shuttleworth, Canonical's first CEO, stepped down, and longtime Chief Operating Officer, Jane Silber, became the new CEO. Shuttleworth retains his position as the head of the Ubuntu Community Council and Ubuntu Technical boards. He focuses his energy on product design and working with enterprise customers and partners, and leaves the day-to-day running of Canonical to Silber. Silber has been with Canonical since before the first release, and the company is expected to continue expanding and operating on its current path.

Canonical's Service and Support

While it is surprising to many users, fewer than half of Canonical's employees work on the Ubuntu project. The rest of the employees fall into several categories: business development, support and administration,

and development of other projects such as Bazaar and Launchpad, which are discussed a bit later in this chapter.

Individuals involved in business development help create strategic deals and certification programs with other companies—primarily around Ubuntu. In large part, these are things that the community is either ill suited for or uninterested in as a whole. One example of business development work is the process of working with companies to ensure that their software (usually proprietary) is built and certified to run on Ubuntu. For example, Canonical worked with IBM to ensure that its popular DB2 database would run on Ubuntu and, when this was achieved, worked to have Ubuntu certified as a platform that would run DB2. Similarly, Canonical worked with Dell to ensure that Ubuntu could be installed and supported on Dell laptops and desktops as an option for its customers. A third example is the production of this book, which, published by Pearson Education's Prentice Hall imprint, was a product of work with Canonical.

Canonical also plays an important support role in the Ubuntu project in three ways. First, Canonical supports the development of the Ubuntu project. For example, Canonical system administrators keep servers up that support development and distribution of Ubuntu. Second, Canonical helps Ubuntu users and businesses directly by offering phone and e-mail support. Additionally, Canonical has helped build a large commercial Ubuntu support operation by arranging for support contracts with larger companies and organizations. This support is over and above the free (i.e., gratis) support offered by the community—this commercial support is offered at a fee and is either part of a longer-term flat-fee support contract or is pay-per-instance. By offering commercial support for Ubuntu in a variety of ways, Canonical has made a business for itself and helps make Ubuntu a more palatable option for the businesses, large and small, that are looking for an enterprise or enterprise-class GNU/Linux product with support contracts like those offered by other commercial GNU/Linux distributions.

Finally, Ubuntu supports other support organizations. Canonical does not seek or try to enforce a monopoly on Ubuntu support; it proudly lists *hundreds* of other organizations offering support for Ubuntu on the Ubuntu Web pages. Instead, Canonical offers what is called second-tier support to these organizations. Because Canonical employs many of the core Ubuntu

developers, the company is very well suited to taking action on many of the tougher problems that these support organizations may run into. With its concentrated expertise, Canonical can offer this type of backup, or secondary support, to these organizations.

Bazaar and Launchpad

In addition to support and development on Ubuntu, Canonical, Ltd. funds the development of Bazaar, a distributed version control tool, and the Launchpad project. Bazaar is a tool for developing software that is used heavily in Ubuntu and plays an important role in the technical processes through which Ubuntu is forged. However, the software, which is similar in functionality to other version control systems such as CVS, Subversion, or BitKeeper, is useful in a variety of other projects as well. More important, Bazaar acts as the workhorse behind Launchpad.

More than half of Canonical's technical employees work on the Launchpad project. Launchpad is an ambitious Web-based superstructure application that consists of several highly integrated tools. The software plays a central role in Ubuntu development but is also used for the development of other distributions—especially those based on Ubuntu. Launchpad consists of the following major pieces.

- **Rosetta:**
  A Web-based system for easily translating almost any piece of free software from English into almost any language. Rosetta is named after the Rosetta Stone, which helped linguists finally crack the code of Egyptian hieroglyphics.

- **Malone:**
  The bug-tracking system that Ubuntu uses to manage and track bugs. It both tracks bugs across different versions of Ubuntu and allows the Ubuntu community to see the status of that bug in other places, including other distributions and potentially upstream. Malone is a reference to the gangster movie musical *Bugsy Malone*.

- **Blueprint:**
  The specification writing and tracking software that Ubuntu and a small number of other projects use to track desired features and their status and to help manage and report on release processes.

- **Answers:**
  A simple support tracker built into Launchpad that provides one venue where users can make support requests and the community can help answer them in ways that are documented and connected to the other related functionality in Launchpad.

- **Soyuz:**
  The distribution management part of Launchpad that now controls the processes by which Ubuntu packages are built, tested, and migrated between different parts of the distribution. Soyuz is a reference to the type of Russian rocket that took Mark Shuttleworth to space. The word *soyuz*, in Russian, means "union."

- **Code:**
  Code allows Launchpad users to publish Bazaar branches of their code and should they choose, can associate them with projects. Code also allows users to mirror or watch Bazaar branches that are hosted elsewhere and even import Subversion and CVS repositories into Bazaar branches.

Launchpad and its components are discussed in more depth in Chapter 9. The importance of Launchpad in the Ubuntu project cannot be overstated. In addition to handling bugs, translations, and distribution building, Launchpad also handles Web site authentication and codifies team membership in the Ubuntu project. It is the place where all work in Ubuntu is tracked and recorded. Any member of the Ubuntu community and any person who contributes to Ubuntu in almost any way will, in due course, create an account in Launchpad.

The Ubuntu Foundation

Finally, in addition to Canonical and the full Ubuntu community, the Ubuntu project is supported by the Ubuntu Foundation, which was announced by Shuttleworth with an initial funding commitment of $10 million. The foundation, like Canonical, is based on the Isle of Man. The organization is advised by the Ubuntu Community Council.

Unlike Canonical, the Foundation does not play an active role in the day-to-day life of Ubuntu. At the moment, the Foundation is little more than a

pile of money that exists to endow and ensure Ubuntu's future. Because Canonical is a young company, some companies and individuals found it difficult early on to trust that Canonical would be able to provide support for Ubuntu for the time frames (e.g., three to five years) that it claims it can. The Ubuntu Foundation exists to allay those fears. Time and consistency has also contributed greatly to the confidence of companies and individuals in Ubuntu, and the foundation will remain to ensure that consistency in the future.

If something unexpected were to happen to Shuttleworth or to Canonical that caused either to be unable to support Ubuntu development and maintain the distribution, the Ubuntu Foundation exists to carry on many of Canonical's core activities well into the future. Through the existence of the Foundation, the Ubuntu project can make the types of long-term commitments and promises it does.

The one activity that the Foundation can and does engage in is receiving donations on behalf of the Ubuntu project. These donations, and only these donations, are then put into action on behalf of Ubuntu in accordance with the wishes of the development team and the Technical Board. For the most part, these contributions are spent on "bounties" given to community members who have achieved important feature goals for the Ubuntu project.

Beyond the Vision: Ubuntu Subprojects, Derivatives, and Spin-Offs

No introduction to Ubuntu is complete without an introduction to a growing list of Ubuntu subprojects and derivatives. While Ubuntu was derived from Debian, the project has also developed a number of derivatives of its own.

First among these is Kubuntu—a version of Ubuntu that uses KDE instead of GNOME as the default desktop environment. The relationship between Kubuntu and Ubuntu is different from the relationship between Ubuntu and Debian. From a technical perspective, Kubuntu is *fully* within the Ubuntu distribution. Organizationally, the Kubuntu team works fully within Ubuntu as well.

A similar organization exists with the Edubuntu project, which aims to help develop Ubuntu so that a configuration of the distribution can be easily and effectively put into use in schools. Although the project has undergone a few changes in recent years, it remains focused on both educational and school-related software and on a Linux Terminal Server Project (LTSP) setup that allows schools to run many students' computers using one or more powerful servers and many "dumb" terminals that connect to the server and run software off it. This relatively simple technical trick translates into huge cost savings in educational settings.

The Xubuntu project is based on the lightweight window manager Xfce. Xubuntu is designed to be appropriate on older or less powerful computers with less memory or slower processors—or just for people who prefer a more responsive environment and a slimmer set of features. While started as an unofficial project, Xubuntu has enjoyed great popularity and has become integrated as an official part of the core distribution.

These and other derivatives are discussed in Chapter 9.

In a way, it is through these derivatives that the work and goals of the Ubuntu project come together and are crystallized. It is only through the free and open source software movements' commitment to freely accessible source code that Ubuntu could be built at all. Similarly, it is only through Ubuntu's continued commitment to these ideals that derivatives can spring from Ubuntu. As a derivative with a view of distributions within an ecosystem, Ubuntu does not see the process of derivation as an insult or criticism. Far from it—Ubuntu thinks derivation is the highest form of compliment.

Outside of Ubuntu, Canonical's work is largely based around software projects such as Launchpad and Bazaar that are designed to facilitate precisely this sort of derivative process. This process, when practiced right, is one that describes an ecosystem of development in which *everyone* benefits—the derivative, Ubuntu, and Ubuntu's upstreams. Only through this derivative process does everyone get what they want.

Derivation, done correctly, allows groups to diverge where necessary while working together where possible. Ultimately, it leads to more work done,

more happy users, and more overall collaboration. Through this enhanced collaboration, Ubuntu's philosophical and technical goals will be achieved. Through this profound community involvement, Bug #1 will be closed. Through this type of meaningful cooperation, internal and external to the project itself, the incredible growth of Ubuntu in its first four years will be sustained into the next four and the next forty.

Summary

This chapter looks at how the vision of a better distribution and OS became the phenomenon that is Ubuntu. It moves through the relationship Ubuntu has with Canonical, Ltd. and the Ubuntu Foundation and finishes with some discussion of the various Ubuntu subprojects, derivatives, and spin-offs that go beyond its original vision.