

By David HM Spector, [Bittersweet Labs](#)

Talk, as they say, is cheap. Running a profitable business is not. In the war of words between traditional OS and application vendors and open source advocates, perhaps the most contentious debate is how to calculate the total cost of ownership/operation (TCO) and determine the return on investment (ROI) for these two very different software models.

Calculating the TCO and ROI for Windows or Solaris is pretty straightforward, involving annual licensing costs, per-user charges, application licenses, hardware depreciations, admin costs--and not a lot else. For Linux and other open source solutions, it's not nearly as clear.

Before we examine how we might calculate both of these values, let's clarify what we mean by TCO and ROI. [Wikipedia](#) defines these terms as follows:

- **Total cost of ownership:** "Total cost of ownership (TCO)' is a financial estimate designed to help consumers and enterprise managers assess direct and indirect costs related to the purchase of any capital investment, such as (but not limited to) computer software or hardware."
- **Return on investment:** "In finance, the return on investment (ROI) or just return is a calculation used to determine whether a proposed investment is wise, and how well it will repay the investor. It is calculated as the ratio of the amount gained (taken as positive), or lost (taken as negative), relative to the basis."

Of course, Wikipedia is not the be-all and end-all of definitions, but these simple descriptions will work well for the purposes of exploring how we might shed a little light on this debate.

TCO addresses a pretty complex question: How much does it cost to run something (in this case, a Linux server that provides some kind of service) from the time that machine/service is set up until the time it's decommissioned? We'll look at why this is a complex question in a moment.

ROI addresses a relatively simple question: Are we making or losing money running something. Again, this "something" is usually a server but also gets to the larger question of the profitability of whatever activity that server supports. ROI must also factor into the TCO; if the TCO eats up all the revenues, there is by definition, no ROI.

## Calculating TCO

Cost of ownership can comprise a lot of different things, but generally includes:

- Initial investment in hardware or lease startup costs
- Monthly lease payments
- Electricity
- Network bandwidth charges
- Portion of data center or server room charges (exclusive of actual bandwidth)
- Portion of your overall infrastructure costs (routers ports, switch ports, cabling)
- OS annual license cost plus periodic upgrades
- Per-user license cost for the OS
- Base cost per application (e.g., basic Sybase license)
- Additional annual per-user application costs
- System admin costs, including staff retraining or new hires
- Cost of backups and offsite storage
- Portion of business insurance

Some have argued that because open source software is "free," its TCO must be zero. Well, that's an interesting theory but wrong in a couple of profound ways.

First, the “free” part of “free software” is less about its actual cost and more about the options it gives the users in terms using the software freely, as they see fit. Open source advocates like to say that free software is “free as in speech” not “free as in beer,” and this is an important point to bear in mind.

Second, running Linux doesn’t mean that you have zero costs, it means you have *lowered* costs and more options. TCO for open source projects can never be \$0! Even if you were to download Linux and support it yourself, that support is still a cost, the server still has to be powered, it consumes bandwidth, etc. You have more options because Linux (and most open source software) is highly portable. So if you need to expand your capabilities by moving to higher performance hardware--even onto a new platform architecture (say from x86 to PowerPC)--you can bring up the new system, reload your data, and go.

## A minimalist TCO worksheet

Here's a good starting point to help determine the TCO for open source projects compared to traditional deployments.

	Linux	Proprietary
Hardware Cost (initial)		
Monthly Hardware Lease		
Base Software Cost		
Additional per user		
Application Base Cost		
Additional per user		
Data Center cost (space)		
Data Center cost (bandwidth)		
Staff Retraining		
System Admin Costs		
Router/Port/Cabling charges		
Insurance		
<b>Annual TCO:</b>		

Once you have identified these costs, they have to be multiplied out over the expected lifetime of either the hardware or the life of the service to calculate the TCO for that service.

The factors in calculating TCO can go beyond the concrete and into the abstract. Some factors can be difficult to quantify or fall outside the typical three- to five-year business plan or a seven-year amortization table, and open source platforms can make such calculations a little less cut and dried.

For example, what would it cost to upgrade to a bigger, faster, or more capable system with a traditional OS? Or conversely, with Linux, if you needed to hold off on upgrading some piece of hardware, how well would the current OS hold up under increased load? The answer is that with Linux, you can get better performance out of older equipment to stretch your hardware budget. This can help stave off being trapped in an endless hardware upgrade cycle. These kinds of analyses often fall outside of traditional TCO calculations.

## How to measure ROI

As you calculate these costs, the interesting savings is that with open source software, there typically isn't a per-user or per-CPU charge. Linux will happily run on a 1 CPU system and a 32 CPU system. Even if you were to buy a copy of Linux from RedHat, which does charge an annual fee for upgrades and support (and which will equate to some portion of the annual cost of, say, Microsoft Windows), it doesn't charge on a per-user or per-CPU basis.

So unlike with Windows, you could run 1,000 users on your 32 CPU system or have 10,000 connections to your databases for the same cost as a single user/single CPU machine.

TCO is a guideline for operating costs, but it also informs something more important: the ability to achieve a return on your investment. A simple way to look at ROI is that if you make more money than a) the basic purchase or lease price of the hardware and b) the cost to run it (the operating cost added up from inception until you decommission something), you have a positive ROI. If you are running at a loss, you have a negative ROI.

As with TCO, there are intangibles in calculating ROI. For example, what is the opportunity cost of going with a closed system versus an open one that gives your company more options to scale your services?

As noted above, with open source systems, you can usually squeeze more life out of older hardware instead of succumbing to the seemingly endless appetite for more and better performance you get with most OSES (such as Windows). Unless your existing (or available older) hardware is costing more in hardware support contracts than new hardware would cost, this is a big win from a ROI perspective.

Bear in mind that hardware vendors typically change almost their entire product lines every 12 to 14 months (even faster for some vendors like Dell). One of the decisions you have to make is whether it makes good business sense to keep running that older system, whose TCO costs pile up at the far end of the serviceable lifetime of the hardware, or set some sort of accelerated swap-out policy that allows you to remain reasonably current but still try to maximize that the value discussed above. Anytime you can make do with an older piece of equipment instead of buying new (which comes right off your bottom line) you're ahead of the game.

Open source systems give your business a lot more flexibility in terms of what software you can deploy and where to allocate your IT dollars, both in terms of an initial investment and as an ongoing service offering by helping you keep new investments to a minimum. Understanding the detailed variables of the operation of your IT infrastructure will allow you to track and manage the operational costs of your systems, and at the end of the day, the total cost of ownership for open source will generally be lower for both the initial deployment and for ongoing administration and operations.

## Additional resources

- TechRepublic's [Downloads RSS Feed](#) **XML**
- Sign up for TechRepublic's [Downloads Weekly Update](#) newsletter
- Sign up for our [Linux NetNote](#)
- Check out all of TechRepublic's [free newsletters](#)
- "[Linux 101: A comprehensive list of Linux services available for all distributions](#)" (TechRepublic download)
- "[Linux 101: Configuring and managing iptables for better network security](#)" (TechRepublic download)
- "[10 things you should know about every Linux installation and distro](#)" (TechRepublic download)

## Version history

**Version:** 1.0

**Published:** April 6, 2006

## Tell us what you think

TechRepublic downloads are designed to help you get your job done as painlessly and effectively as possible. Because we're continually looking for ways to improve the usefulness of these tools, we need your feedback. Please take a minute to [drop us a line](#) and tell us how well this download worked for you and offer your suggestions for improvement.

Thanks!

—The TechRepublic Downloads Team