

The Open Road Map

For most of its decade-plus history, the Linux kernel has cycled between odd-numbered experimental revisions (eg. 2.3 and 2.5) and even-numbered production versions (like 2.2 and 2.4). Traditionally, forward-looking developers, interested community members, and other users eager for new features and capabilities have kept their eyes on the experimental lines. Linux distribution suppliers, their customers and many others usually built and deployed on production kernels.

This familiar odd-even alternation came to a quiet end after the release of the 2.6 Linux kernel in December of 2003. The 2.6 kernel maintainer Andrew Morton and also Linus Torvalds felt that the two year wait between major releases, while not extreme by proprietary standards, was not in keeping with the rhythm and spirit of the rest of Open Source.

Today, both development and experimentation occur on the 2.6 kernel source base. New capabilities, bug fixes, security patches, and other enhancements are applied to kernel trees maintained at kernel.org. Every 2-3 months a stable 2.6 family version is "blessed" for commercial use. Andrew Morton has pointed out that with 2.6, major subsystems within the kernel have relatively few interdependencies, making it easier to revise (or safer to break) any one of them. Linus and Andrew have not ruled out an eventual 2.7 branch - they're just not predicting when or how or if it will come into being.

With this new regime in place, 2.6 kernel adoption proceeds apace. Few hiccups have arisen from this shift in practice. I write this column on SuSE 9.2 Professional built on 2.6.5-7.111.5-default. I have just installed Fedora Core 3 based on 2.6.9 on my multimedia and hacking machine. I am very happy with 2.6. The community seems pretty content, and millions of end-users benefit from faster access to new capabilities and improved performance.

But is the market pleased? Yes and no.

In the first half of 2004, commercial interests and analysts fretted over perceived instability in 2.6, a lack of device drivers, uptake by distribution vendors, and agonized over the change in numbering. Today 2.6 is

Linux has no Road Map because it is a technology, not a product.

remarkably stable and widely deployed, the device driver gap has been bridged, and hysterical kernel-version innumeracy is in remission. However, the market still frets. Why? Because the new development model exposes a truth heretofore only clear to Open Source folk - there is no Linux Road Map.

Road Maps are forward-looking, laying out versions, features and technologies that vendors hope to introduce in months or years to come.

Road Maps are supposed to "point the way" for product teams and customers. As artifacts of the Cathedral, Road Maps more often serve to document, whimsically and ephemerally, wishful thinking by engineering and marketing.

The notion of a Road Map implies continuity from past to present to future. Reality intrudes as Road Maps are regularly defied by market necessity and corporate edict. Product development meanders across terra incognita, blazing trails later found littered with the tatters of discarded Road Maps.

So why does the market crave these relics from the Cathedral? Road Maps are product marketing catechism. Road Maps speak ex cathedra giving comfort with visions of order and progress and strategy. Road Maps give salespeople and journalists something to proffer and then to complain about when led astray.

Open Source developers revel in the shock value of having no Road Map. Coming out of the proprietary world some time ago, I was not shocked by this cartographic vacuum. I was delighted that FOSS had the good sense to admit a basic truth of software engineering - serving real-world needs of customers and communities seldom corresponds to the niceties of documented and approved straight paths. In its honest embrace of the vagaries of the marketplace and engineering realities, FOSS rejects the Cathedral view of development and embraces a simple fact of science - discovery cannot be planned.

Linux has no Road Map because it is a technology, not a product. Technologies are used to build products. Technologies are embodied in products, but are seldom products themselves.

Linux kernel technology is used to build distributions and platforms, like Red Hat, SuSE, Mandrake, Debian, Oracle Unbreakable Linux, embedded platforms, and roll-your-own distros used by business, academics and hobbyists.

It would be dangerously disingenuous to pretend that Linux and FOSS advanced by dint of inspiration and discovery alone. Of course project leaders plan for one or more releases into the future. Certainly there are guiding hands and guiding principles. For example, my organization, the OSDL, gathers requirements from the Linux ecosystem of platform suppliers, ISVs and end-users. OSDL members refine those requirements and use them as fuel for initiatives to foster development and to create specifications for distribution suppliers to follow. Project teams, companies, and other .orgs, like the FSF, FSG, SA-Forum, and CELF do likewise with specifications, standards, licenses, and reference platforms.

What FOSS development really lacks is the pretence of a Road Map.

It has been said that Open Source creates a chorus from cacophony - ordered technology from a chaotic universe of developer and market interests. Traditional Road Maps create chaos by repeatedly positing order and then dashing expectations of timely or functional delivery. Which do you prefer?

Bill Weinberg is Open Source Architecture specialist and Evangelist for the Open Source Development Lab (OSDL)



Bill Weinberg