# Version Management Tools: CVS to BK in the Linux Kernel

Maha Shaikh
*Department of Information Systems*
*London School of Economics*
m.i.shaikh@lse.ac.uk

Tony Cornford
*Department of Information Systems*
*London School of Economics*
t.cornford@lse.ac.uk

## Abstract

*Version management tools might be seen as a prerequisite for open source development today as projects become too large to be managed by maintainers alone. Yet the OS process depends on fluid coordination and collaboration, with the underlying qualities of this process based on firm trust and respect for fellow developers. This paper is a study of how debate over version tools reflects governance and decision making in an OS community. The paper is based on a study of the Linux kernel community as it first saw a partial acceptance of the CVS tool, and then later adopted BK. The paper explains the adoption processes in relation to governance concerns, licence issue, and questions of technical performance.*

## 1. Introduction

Open source projects, if successful, grow large quite rapidly. Under such conditions it may become necessary for developers to employ some version management tool. Thus, in the case of the Linux kernel, before version tools were introduced Linus Torvalds would make pre-patches and/or releases and coordination was achieved through individuals acting as maintainers. If maintainers were able to keep up with the patches, and not lose or confuse the various patches, then a version management system would not be needed. However, while this may be possible for a small band of developers working on a small application, it would not seem to be so for a project the size of the Linux kernel. In this case the maintainers for different applications and tools found, around 1996-97, that they badly needed some version tool to manage all the patches being sent to them and to maintain communications within the broader community. Linus Torvalds too was prepared to adopt a tool but was decidedly against the use of CVS [Concurrent Versions System] for the official kernel tree. CVS was (and is) the most common such tool, written by Dick Grune in 1986 as a collection of scripts to enhance RCS [Revision Control System], but later developed into a major Copy-Modify-Merge model[7]. According to Kilpi, a version management tool "manages and keeps track of the configuration items which are any documents created during a software development process, and which are found necessary to be placed under configuration control like requirements documents, data flow diagrams, design documents, source code, and test results" [12].

The focus in this paper is on how and why the Linux Kernel developer community came to adopt CVS, and then later moved, with some anguish, to a different, but non open source tool, BitKeeper [BK]. This was developed by Larry McVoy of the company BitMover in the late 1990s, more than 10 years after CVS, based to a large degree on meeting the Linux kernel community needs.

Through these transitions into CVS, and from CVS to BK, we expose a number of interesting issues of governance, judgements of technical quality and pursuit of ideology in OS projects. The principal data used are drawn from the Linux Kernel mailing list [LKML] archive. The paper focuses on the period of transition from CVS to BK, though the early phase of CVS-use is touched on to contextualize the discussion.

## 2. Brief history

Linus Torvalds' decision in 1995 to *not* adopt CVS for kernel development was backed by trusted lieutenants like Alan Cox and David Miller [31]. There is then little mention of CVS on LKML until February of 1996, and it not until April of 1997 that we see the first substantial mention of CVS for over a year; three messages on the archive for that month are devoted to CVS as a whisper of complaint about the lack of a complete CVS kernel repository. As Moody describes, at least Miller was not so happy with Linus disregarding the use of CVS [24] and, by late 1997, we see CVS widely used within the Linux community. The April messages proved to be the precursor to a heated debate over the decision of Linus to not use CVS and September 1997 saw a surprising reversal by David Miller, with an eager post to announce his decision to make '*full raw snapshots of my tree on vger available for ftp*' [20]. CVS was then acceptable within at least some parts of the kernel community, though Linus himself was never convinced of CVS-use, so the 'tree' kept up to date by Miller was only quasi-official.

The next few years proved troublesome for CVS and this debate sparked an interest in the use of BitKeeper [BK]. Larry McVoy, part-owner of BitMover, the company which develops BK, worked to adapt BK to suit the needs of both Linus and Linux. BK, however, was not universally seen as a remedy, indeed it was referred to as the 'Darkstar'

project and, not being open source (GPL), was anathema to many Linux developers.

Still, by 2002 BK was the official tool for Linux, thanks to the support of Linus Torvalds and a few of his lieutenants. This rosy period was not to last and later that year BK was labelled as '*evil* corporate software'. Predictably, Richard Stallman saw this as an opportunity to invoke his ideology of 'free' software, only to be not so subtly tamed by Linus announcing that he doesn't believe in ideology "*Quite frankly, I don't _want_ people using Linux for ideological reasons. I think ideology sucks. This world would be a much better place if people had less ideology, and a whole lot more "I do this because it's FUN and because others might find it useful, not because I got religion"*" [34].

In summary, the Linux kernel archive shows that over the last 10 years or so there have been large pockets of CVS users who gained in legitimacy and then gradually shifted to BK, but not without serious concerns being raised. To this day there is no single version tool being employed by all Linux developers. Torvalds is currently using BK but has shown himself receptive to other options if they are technically superior and secure, and fit the Linux governance structure, and he is on record as willing to try Subversion, Arch etc. if they can match the efficiency levels of BK.

## 3. Governance issues

Linux is usually understood as having a well-defined hierarchy and governance structure with a sequence of steps that must be followed in order to review and/or accept a developer's patch. Such leadership structures become explicit when examining who has write/commit privileges for the core repository [6]. In these terms it is understood that Torvalds, to some extent, is willing to share his authority with a few 'trusted lieutenants' and there is thus a role structure within Linux where each role 'both reflects and supports its [community's] activities' [25]. The two most important roles are that of the trusted lieutenant [also called the credited developers] and credited maintainer. Maintainers care for one module of the whole program assessing user contributions and keeping interfaces [4, 8, 25], while developers have a wider responsibility for architecture and stronger commit privileges.

Linus Torvalds, the "*benevolent dictator the whole community trusts"*, leads Linux [2]. Indeed comments such as "*Linus owns the Linux kernel. He is the dictator on what happens with it. As such he can do with it as he pleases. If anyone doesn't like his actions, they are free to fork the kernel*" are common on the LKML [1]. Landley explains further "*He's an architect. He steers the project, vetoing patches he doesn't like and suggesting changes in direction to the developers. And he's the final integrator, pulling together dispirate patches into one big system*" [14].

There are, however, two opinionated camps on the LKML – one which reveres Linus and the other which is more sceptical of his decisions. If Linus can inspire statements such as "*I nominate Linus for Beloved Benevolent Dictator*" [5], where beloved seems to be at odds with dictatorship, he also invokes "*I wish Linus would be more responsive…Linus likes the way he does things and doesn't care if others don't like it. I don't expect to see much change there*" [11]. Senior developers like Raymond say "*You're our chosen benevolent dictator and maybe the second coming of Ken*" [26], but also play a sobering influence if adulation and praise have, on occasion, gone to Linus's head and where he has attacked even Raymond's judgement; "*I'll give you a rule of thumb, and I can back it up with historical fact. You can back up yours with nada*". Raymond replies that "*Yes, if twenty-seven years of engineering experience with complex software in over fourteen languages and across a dozen operating systems at every level from kernel out to applications is nada :-). Now you listen to grandpa for a few minutes. He may be an old fart, but he was programming when you were in diapers and he's learned a few tricks..[I] can see that *you*, poor damn genius that you are, are cruising for a serious bruising.* " [26].

As noted above, from about September 1997 the resistance to CVS had been eroded by the growing perception of a need for a version tool, though security concerns with CVS did force restricting access to the CVS tree. This still required many developers to keep their personal CVS tree working alongside the one maintained by Miller, "*The CVS tree is not the official kernel tree. Linus controls what gets into the official kernel tree, a lot of different developers have write-access to the CVS tree*" [29]. As pointed out above, Linus, due to reservations over CVS, did not keep the official tree though he was aware that Miller was maintaining a tree.

More generally, there was growing resentment of, what was seen as mismanagement of patches being sent for appraisal. Rik van Riel criticized the maintainer abilities of Linus "*Silently dropping bugfixes on the floor is not maintainership*" [27], so too King, "*I envy Alan, Linus, and Marcelo for having the ability to silently drop patches and wait for resends.*" [13]. Still Linus was quite adamant about his refusal to use CVS officially and the public repository feature of CVS made him shudder, "*I don't like the idea of having developers do their own updates in my kernel source tree. I know that's how others do it, and maybe I'm paranoid, but there really aren't that many people that I trust enough to give write permissions to the kernel tree*" [31]. Trust is a key feature of the OS process, the deterioration of which could lead to a breakdown of the community and process. Linus' dropping patches was creating uncertainty [24]. It took a face-to-face

meeting with Torvalds, Miller, McVoy and some others to bridge the gap.

Though Linus never used CVS he did in time take up BitKeeper [BK] and even encouraged its use within the community "*The long-range plan, and the real payoff, comes if main developers start using bk too, which should make syncing a lot easier*" [33] and thus it became official that the Linux kernel would use BK with Linus's initiative and blessing. But many in the community due to its non-GPL licence did not like BK. Pragmatically, Ts'o suggested "*If Linus were willing to dictate from high that we were going to use bitkeeper, and that all patches had to come in as bitkeeper changelogs, then that might get us critical mass*" [36], and he was right. The indignant protests in the community at the adoption of BK didn't then stop its use, but it did force Linus to at least try and appease the kernel developers. Some like Alan Cox were still against the use of BK and perhaps that is why Linus emailed the LKML [35] in an attempt to meet them halfway, plus he felt strongly that BK was technically more efficient than other tools.

## 4. Licence disparity

Agreed that CVS was not considered the perfect tool for the Linux kernel but there was one consideration in its favour, it was GPLed software. BitKeeper, was and is not, an open source tool. Ts'o thus also noted "*given its non-OSS license, it's not clear it will get that critical mass.. there are enough other people who are license fanatics*" [36]. Linus, however, didn't think the licence should pose such an issue and in February 2002 he declared it the official version tool for the kernel. A volley of dissenting voices came screaming across the LKML, "*Linux ceases to be free software when you require nonfree software to contribute it*" and this from the trusted Alan Cox [3].

The situation was exacerbated when in early October 2002 some developers noticed that the BitKeeper licence [BKL] had been changed to include a new clause, '..this License is not available to You if You…develop, produce, sell, and/or resell a product which contains substantially similar capabilities of the BitKeeper Software, or, in the reasonable opinion of BitMover, competes with the BitKeeper Software'. "*This would seem to be a change which is not Open Source developer friendly",* said Gall [9]. McVoy replied clarifying that BitMover [BM] had left out the word 'distribute' in the clause in order to protect open source developers [19]. Again Richard Stallman took the opportunity to echo his 'freedom speech', "*The spirit of the Bitkeeper license is the spirit of the whip hand. It is the spirit that says, "You have no right to use Bitkeeper, only temporary privileges that we can revoke…. Outrage at this spirit is the reason for the free software movement*" [30]. There were some who would rather that *"your [Larry McVoy's] company dies ASAP and bitkeeper stops poisoning air here"* [15].

The substance of concern about BK not being GPLed is well expressed by Molnar, "*Today the 'Linux kernel' is not the source code anymore, it's the source code plus the BK metadata… The BKL.txt license currently says: 'By transmitting the Metadata to an Open Logging server, You hereby grant BitMover, or any other operator of an Open Logging server, permission to republish the Metadata sent by the Bit- Keeper Software to the Open Logging server'. i'm worried about the following issue: by default the data and the MetaData is owned by whoever created it. You, me, other kernel developers. We GPL the code, but the metadata is not automatically GPL-ed, just like writing a book about the Linux kernel is is not necesserily GPL-ed*" [23]. Molnar stressed that a change in the BKL is needed which 'ensures that metadata attached to GPL-ed code is also licensed under the GPL, and creates a clearly GPL-ed repository'.

In reply Larry McVoy stresses over and over again that "*The BKL says that you get source, you can wack the source and redistribute it, that subsections of the system such as the memory checker, the mmap based DMB library, and the installer, are all also licensed under the GPL. It is our intent, and the license reflects that, that if you are doing work out in the open, on open source or anything else, that you can use it free of any monetary payment*" [17]. McVoy is supported by David Miller who reiterates that BM does not charge the Linux community anything for their use of BK, *"nobody need[s] to give Larry one dime to use BK for kernel work. And to be honest, I get better support from Larry for *free* than you'll most often get when paying some company for software support*" [22]. Such use of BK within the Linux community might be explained, using the language of actor network theory, as BK becoming an obligatory passage point (OPP), something through which all other interests must pass, and in contrast to the other acknowledged OPP, Torvalds himself.

The licence topic generated a heated discussion which lasted for more than 200 email messages in October 2002. Midway through this repertoire, Linus, sensing real annoyance and fear on the part of his community, reacted with, "*If this is a concern, it actually appears that BK has the capability to "enforce" a license, in that I could make BK aware of the GPL and that would cause BK to pop up a window saying "Do you agree to this license" before the first check-in by a person (the same way it asked you whether you wanted to allow openlogging)*" [35]. Thus it must have some superior qualities, technical or otherwise, to goad Linus to continue using it in spite of all the opposition. [38]

## 5. Technical superiority

A year earlier the debate had been on a more functional basis. In September, 2000 there were more than a 100 messages posted in less than a week

concerning problems developers had with source code management tools, including CVS, BK, Subversion, and Arch. Out of these four tools 3 were open source and only BK was closed source.

CVS, it was noted, had caused security breakdowns within FreeBSD and OpenBSD and Miller, of all people in May 1997, had claimed that CVS use was the *"worst feature of the *BSD projects and is one of the numerous reasons behind the personal and political problems they have. This strategy leads to power struggles and political problems"* [21]. So not only were there security concerns but CVS was seen as leading to political struggles because it allowed many people to submit patches to the official software tree and thus questions of whose patch will get accepted became prolific. Wilson [37] suggested that CVS is '10 years behind equivalent commercial offerings' in which case Linus's decision to not use CVS maybe understandable. Linus wanted to avoid more pressures plus he believed that CVS "*allow*[ed] *automatic acceptance of patches, and positively _encourage_ people to "dump" their patches on other people, and not act as real maintainers*" [32]. McVoy was more than eager to second this, "*CVS can't do what you want, BK can. People can't have write access in CVS for the obvious reasons, the tree becomes a chaotic mess of stuff that hasn't been filtered. But in BK, because each workspace is a repository, people inherently have write access to *their* repository*" [18].

Besides these concerns, CVS was also not considered as technically efficient as BK. McVoy gave a comparison of how both CVS and BK work, pointing out just why BK shows more efficiency, "*BK only moves the data it needs to move. That means if you have a 100GB file in which you have changed one byte, BK will move on the order of 1 byte to update that file. And that's it. CVS moves whole files just to discover there is nothing to do*" [16]. David Gatwood came up in defence of CVS with, *"Faster, yes, but it still does the same thing, just in a different way that uses less bandwidth"* [10]. McVoy hit back with" *BitKeeper is faster in practice because it isn't trying to talk to the CVS server for all of the operations*" and showed comparative figures for time taken to carry out a null update "*CVS: 139.5 seconds* and *BK: 1.6 seconds*" [16].

One way to account for these technical debates is to see CVS as playing the role of stepping stone, as McVoy puts it, "*all* [of the developers had] *used and learned from CVS. But just because CVS is useful doesn't mean it is the best answer*" [16]. However, why wasn't Subversion or Arch, OS themselves, used instead of BK? As one posting puts it, "*Subversion isn't it, we can't work from the same repository with tens of thousands of people, any BK replacement would have to be a distributed system. PRCS2 might become a suitable system, if somebody gets around to picking up its development. Arch might work too, but I remember talking to some Arch fans a while back who "were about to" import the whole kernel history into an Arch repository ... the fact that I never heard from them again makes it look like maybe Arch couldn't yet handle a repository the size of the kernel. In short, until somebody builds a free (as in RMS-free) source control system that's as good as bitkeeper for what the kernel needs, bitkeeper is the only available tool for the job.*" [28]

So in spite of the fact that Torvalds suffered some initial hassles using BK, he had to " *spend about a week trying to change my working habits and scripting bitkeeper… I expect to be a bit slower to react to patches for a while yet, until the scripts are better. However, some of it pays off already*" [33]. It seems that BitKeeper is here to stay, at least for a little while longer anyway.

## 6. Conclusion

Three main issues, albeit intimately related, become apparent in this study of embedded technical infrastructure within an OS project; the relationship to and reflection of governance, the technological rationale, and the role of ideology in the form of licence considerations. This OS community is not above using closed source tools and products if it suits them. We need to understand this better and see it against the policy (or inclination) of open source developers, which has always been that, if you are not happy with any application, then set about creating your own ("scratch your itch"). In these terms Torvalds being questioned on his use of BK becomes understandable, but we are still left with the question why no appropriate OS tool has emerged? More generally we see here how tools come to play a key role in the OS process, technically, organisationally and as a discourse within the community, and how embedded and enmeshed they are with the human.

On this basis we propose as issues for discussion in the workshop the following:

- Why has no OS tool emerged to meet this need?
- To what extent is OS ideology implicated in the process of tool choice and development, or is it rather invoked to justify positions based on other factors?
- What theoretical and methodological tools are available to us as researchers to study such aspects of OS? Our own work draws on Actor Network Theory and social constructivism, but what else is available and of potential use?

## 8. References

[**Note**: To access the url references from the LKML archive add http://www.uwsg.indiana.edu/hypermail/linux/kernel. Abbreviated urls are used to save space in a 5 page paper.]

[1] A. Altaparmakov, Re: [PATCH] Remove Bitkeeper documentation from Linux tree, 2002. ../0204.2/1107.html

[2] L. Augustin, Review of JUST FOR FUN: The Story of an Accidental Revolution, 2001. http://www.everythinglinux.com.au/item/TE0806?elinux=c5728616ca109cdcb6d6acdbe4ad6f32

[3] A. Cox, Re: Proposal: Linux Kernel Patch Management System, 2000. ../0009.1/1076.html

[4] G. Dafermos, "Management and Virtual Decentralized Networks: The Linux Project," *First Monday*, vol. 11, no. 6, 2001.

[5] A. Esh, Re: signal (SIGFPE,SIG_IGN), possible solution?, 1996. …/9604.3/0007.html

[6] J. Feller and B. Fitzgerald, *Understanding Open Source Software Development*, London, UK: Addison-Wesley, 2002.

[7] K. Fogel, *Open Source Development with CVS*, Scottsdale, AZ: Coriolis Open Press, 1999.

[8] E. Franck and C. Jungwirth, Reconciling investors and donators - The governance structure of open source, Working Paper Series, University of Zurich., 2002. http://www.unizh.ch/ifbf/webFuehrung/Dokumente/WorkingPaper/FranckJungwirthInvestorsAndDonatorsWP8.pdf

[9] T. Gall, New BK License Problem?, 2002. …/0210.0/1496.html

[10] D. Gatwood, Re: [ANNOUNCE] Darkstar Development Project, 2000. …/0009.1/0560.html

[11] R. Gooch, Re: The direction linux is taking, 2001. …/0112.3/0467.html

[12] T. Kilpi, "New Challenges for Version Control and Configuration Management: a Framework and Evaluation," *IEEE Computer*, 1997, pp. 33-41.

[13] R. King, Re: The direction linux is taking, 2001. …/0112.3/0447.html

[14] R. Landley, A modest proposal -- We need a patch penguin, 2002. …/0201.3/1000.html

[15] P. Machek, BK is *evil* corporate software [was Re: New BK License Problem?], 2002. …/0210.0/2398.html

[16] L. McVoy, Re: [ANNOUNCE] Darkstar Development Project, 2000. …/0009.1/0549.html

[17] L. McVoy, Re: Proposal: Linux Kernel Patch Management System, 2000. …/0009.1/1264.html

[18] L. McVoy, Re: The direction linux is taking, 2001. …/0112.3/0530.html

[19] L. McVoy, Re: New BK License Problem?, 2002. …/0210.0/1500.html

[20] D. Miller, First vger CVS 2.1.x kernel source repository snapshot, 1997. …/9709.1/0432.html

[21] D. Miller, Re: naive question: kernel CVS repository, 1997. …/9705.3/0292.html

[22] D. Miller, Re: BK is *evil* corporate software, 2002. …/0210.0/2410.html

[23] I. Molnar, BK MetaData License Problem?, 2002. …/0210.0/1918.html

[24] G. Moody, *Rebel Code: Linux and the Open Source Revolution*, London: Penguin, 2001.

[25] J.Y. Moon and L. Sproull, "Essence of Distributed Work: The Case of the Linux Kernel," *First Monday*, vol. 5, no. 11, 2000.

[26] E. Raymond, Re: [PATCH] Re: Move of input drivers, some word needed from you, 2000. …/0008.2/0240.html

[27] R.van Riel, Re: The direction linux is taking, 2001. …/0112.3/0498.html

[28] R.van Riel, Re: New BK License Problem?, 2002. …/0210.0/2082.html

[29] B. Rosenkraenzer, Re: CVS repository of the development Kernel, 1998. …/9807.3/0161.html

[30] R. Stallman, Bitkeeper outragem, old and new, 2002. …/0210.1/1767.html

[31] L. Torvalds, Re: CVS, Linus, and us, 1995. .../9602/0800.html

[32] L. Torvalds, Re: The direction linux is taking, 2001. …/0112.3/0474.html

[33] L. Torvalds, linux-2.5.4-pre1 - bitkeeper testing, 2002. …/0202.0/0989.html

[34] L. Torvalds, Re: [PATCH] Remove Bitkeeper documentation from Linux tree, 2002. /0204.2/1018.html

[35] L. Torvalds, Re: BK MetaData License Problem?, 2002. .../0210.0/2048.html

[36] T. Ts'o, Re: Proposal: Linux Kernel Patch Management System, 2000. …/0009.1/1022.html

[37] G. Wilson, "Is the Open-Source Community Setting a Bad Example?," *IEEE Software*, vol. 16, no. 1, 1999, pp. 23-25.

[38] R. Zippel, Re: [PATCH] Remove Bitkeeper documentation from Linux tree, 2002. /0204.2/1388.html