

# Research Issues for Continuous (Re)Design of Open Source Software

By Karim R. Lakhani

MIT Sloan School of Management | The Boston Consulting Group

October 2003

A recent dialog between the core developers of the Linux kernel illustrates the topic of our workshop<sup>1</sup>:

**Larry McVoy:** Linux isn't there yet and unless the development model changes somewhat, I'll stand behind my belief that it is unlikely to ever get there.

**Rik van Riel:** I'm very interested too, though I'll have to agree with Larry that Linux really isn't going anywhere in particular and seems to be making progress through sheer luck.

**Linus Torvalds:** Hey, that's not a bug, that's a FEATURE! You know what the most complex piece of engineering known to man in the whole solar system is?

Guess what - it's not Linux, it's not Solaris, and it's not your car.

It's you. And me. And think about how you and me actually came about - not through any complex design. Right. "sheer luck".

Well, sheer luck, AND:

- free availability and crosspollination through sharing of "source code", although biologists call it DNA.

- a rather unforgiving user environment, that happily replaces badversions of us with better working versions and thus culls the herd (biologists often call this "survival of the fittest")

- massive undirected parallel development ("trial and error")

..... And don't EVER make the mistake that you can design something better than what you get from ruthless massively parallel trial-and-error with a

feedback cycle.....Let's just be honest, and admit that it wasn't designed. Sure, there's design too - the design of UNIX made a scaffolding for the system, and more importantly it made it easier for people to communicate because people had a mental model for what the system was like, which means that it's much easier to discuss changes.

..... And I know better than most that what I envisioned 10 years ago has nothing in common with what Linux is today. There was certainly no premeditated design there. And I will claim that nobody else "designed" Linux any more than I did, and I doubt I'll have many people disagreeing. It grew. It grew with a lot of mutations - and because the mutations were less than random, they were faster and more directed than alpha-particles in DNA..... Have you ever heard of a project that actually started off with trying to figure out what it should do, a rigorous design phase, and a implementation phase? Dream on. Software evolves. It isn't designed. The only question is how strictly you control the evolution, and how open you are to external sources of mutations. And too much control of the evolution will kill you.

There is real tension between those that believe that "good" software development can best be accomplished via an emphasis on planning, control and coordination and those that believe in a more evolutionary perspective with an emphasis on rapid design-code-use cycles coupled with a very large and heterogeneous user/developer community. It is even more interesting that this divergent viewpoint is present within a single software development community.

A software design/development process that is continuous, evolutionary and open raises some very interesting research and pragmatic questions:

## *What kind of processes?*

- What are the variations in the types of design processes used by open source communities?
- What are the underlying mechanisms that enable a continuous design process?

---

<sup>1</sup> Linux Kernel mailing list archive (November – December 2001):  
<http://www.ussg.iu.edu/hypermail/linux/kernel/0111.3/1957.html>

- What are the implications for organization and software engineering theory given the presence of such practices and processes?

***What works?***

- What are the performance implications for such a design process?
- What are the appropriate metrics for performance?
- What are the contingent factors that affect choice of continuous design mechanisms?

***How to do it?***

- What happens at the intersection of an evolutionary and planned process?
- How can firms participate in a software development community that engages in a continuous design process?
- How can firms initiate a continuous design process within their own boundaries?

I am looking forward to the workshop!