

Center for

eBusiness@MIT

<http://ebusiness.mit.edu>



A research and education initiative at the MIT Sloan School of Management

**Open Source Software:
Innovation By and For Users –
No Manufacturer Required!**

Paper 133

Eric von Hippel

March 2001

For more information,

please visit our website at <http://ebusiness.mit.edu>

or contact the Center directly at ebusiness@mit.edu or 617-253-7054



Open Source Software:**Innovation by and for Users – No Manufacturer Required!**

Forthcoming Sloan Management Review, Summer, 2001

Eric von Hippel

March, 2001

Open source software projects and products are an exciting example of complete innovation development and consumption communities run by and for users – no manufacturer required. User innovation communities have a great advantage over the manufacturer-centered innovation development systems that have been the mainstay of commerce for hundreds of years: They enable individual users (persons or firms) to develop exactly what they want when they want it - rather than relying on a manufacturer to act as their (often very imperfect) agent. At the same time, individual users do not have to develop everything they need on their own – they can benefit from innovations developed by others and freely shared within the user community.

User innovation communities have existed long before the emergence of open source software and apply far beyond it. For example, very similar communities can be found developing physical products like novel sports equipment. As we learn to understand such communities better, we will be in a position to improve them where they now exist and to systematically extend their reach and attendant advantages more widely across the economy.

Consider and compare the following two examples of user innovation communities in their early stages – the first in software, and the second in sports.

Apache Server Software

Apache open source web “server” software is used on web server computers connected to the Internet. A web server forms the essential backbone of the World Wide Web infrastructure. It hosts all the web pages and provides the appropriate content as requested by Internet browsers. The initial server software that evolved into Apache was developed by an undergraduate at the University of Illinois - Rob McCool, who developed it for and while working at the National Center for

2

Supercomputing Applications (NCSA). The source code as developed and periodically modified by McCool was posted on the web, so that users in other sites could download it, use it, and also modify and further develop it to suit their own needs.

When McCool left NCSA in the middle of 1994, a small group of web masters who had adopted NCSA server software for their own web sites decided to take on the task of continued development for themselves. A core group of eight users began the work by gathering all documentation and bug fixes that had been made for NCSA server software up to that point. They put this material together in the form of a consolidated patch. Over time, the name of this *patchy* web server software evolved into Apache. After extensive feedback and modification by users, Apache 1.0 was released on December 1, 1995.

In the space of four years and after many modifications and improvements contributed by many users, the Apache web server has become the most popular web server software on the Internet. It has also received many industry awards for

excellence and, in the face of strong competition from commercial competitors like Microsoft and Netscape, it is now used by about 60% of the millions of World Wide Web sites extant in 2001. (Lakhani and von Hippel, 1999)

High performance windsurfing.

The evolution of “High-performance” windsurfing was reported on by the MIT PhD student Sonali Shah in a recent working paper. It involves acrobatics such as jumps and complete turns in the air while on a windsurfing board. Prior to this development, the sport tended to focus on more traditional sailing techniques, with windsurfing boards being used essentially as small, agile sailboats. The development of high-performance windsurfing has involved innovations in windsurfing technique and also in the equipment, which has been co-evolved to support those new techniques. The fundamentals of the sport were developed in 1978 in Hawaii where a group of like-minded users were clustered. In an

3

interview with Shah, Larry Stanley, a high performance windsurfing pioneer, describes these events.

“In 1978 Jurgen Honscheid came over from West Germany for the first Hawaiian World Cup and discovered jumping, which was new to him, although Mike Horgan and I were jumping in 1974 and 1975. There was a new enthusiasm for jumping and we were all trying to outdo each other by jumping higher and higher. The problem was that, like in the past, the riders flew off in mid-air because there was no way to keep the board with you – and as a result you hurt your feet, your legs, and the board.

“Then I remembered the “Chip,” a small experimental board we had built with footstraps, and thought "it's dumb not to use this for jumping." And that's when we started jumping first with footstraps and discovering controlled flight. I could go so much faster than I ever thought and when you hit a wave it was like a motorcycle rider hitting a ramp – you just flew into the air. We had been doing that before but had been falling off in mid-air because you couldn't keep the board under you. All of a sudden not only could you fly into the air but you could land the thing and not only that but you could change direction in the air!

“The whole sport of high performance windsurfing really started from that. As soon as I did it, there were about 10 of us who sailed all the time together and within one or two days there were various boards out there that had footstraps of various kinds on them and we were all going fast and jumping waves and stuff. It just kind of snowballed from there.”

4

By 1998 there were over a million people engaged in windsurfing, with a large fraction of boards sold incorporating the user-developed innovations for highperformance windsurfing.

As matters progressed, these two user innovation communities evolved and became more complex. On the surface, being in different fields, they may look quite

different, but in fact they are very similar in fundamental ways. Both open source and sports user innovation communities may consist of from a few to many thousands of volunteer participants. In the case of open source software projects, participants interact on-line via the Internet. In the case of sports communities, participants physically interact via travels to favorite sports sites and sports contests. In each community there are many interwoven tasks that get carried out on a volunteer basis. In the case of OS project communities, most simply use the code the community has generated. Others may write new code, debug code others have written, ask or answer questions posted on Internet help sites, or help coordinate the project. In the case of sports communities – even those that are new or rapidly evolving - most simply “play the game.” Others may develop new techniques and equipment, debug innovations that have been developed, receive or voluntarily provide coaching or coordination of group activities. In both cases, commercial enterprises may attach to the user community or play complementary roles: Red Hat and VALinux are well-known examples in the case of Open source, and professional sports leagues and commercial producers of sports equipment are examples in the case of user sports communities.

2. Advantages of user innovation

In both of the cases just described, users were the developers, builders and consumers of the innovations they developed – no manufacturer required. Direct innovation by users has a great advantage over innovation by manufacturers from the users’ point of view – an advantage that is captured by the well-known adage: “If you want something done right, do it yourself!” This adage holds in the case of new product and service development for two reasons. First, a manufacturer cannot know what a user wants to the depth and detail that the user does. Second, even if a manufacturer *did* know

5

exactly what a user wants, it would not have an incentive to provide exactly that. New product developers clearly must have accurate information on users’ needs and context of use if they are to build a product accurately responsive to user need. This information is *generated* at user sites and is naturally accessible there – but it is typically very “sticky” - costly to move from the users site to outside developers. (For example, the conditions that cause software to crash are available “for free” at the site of a user with the problem, but can be very difficult to reproduce elsewhere.) Also, this information is not a static matter that can be transferred to manufacturer-based developers all at once. Rather, it evolves at the user site through “learning by doing” as the user experiments with prototype innovations. (Recall from the windsurfing example presented above that users only *discovered* that they could – and wanted to – control the direction of a board when it was in the air *after* they began experimenting with the prototype footstraps they had developed.)

Manufacturers are the agents of users with respect to new products and services. It is their job to develop and build what users want and need – they do not want the products for themselves. The trouble is that manufacturers’ incentives don’t match those of users – so users end up paying an “agency cost” when they delegate design to manufacturers. A major part of this cost comes in the form of being offered products that are not be the best possible fit to users’ needs – even assuming manufacturers know precisely what those needs are.

Manufacturers want to spread their development costs over as many users as

possible – which leads them to want to design products that are a close-enough fit to induce purchase from many users, rather than to design precisely what any particular user really wants. This incentive can be explicitly seen in the operation product “users’ groups” set up by manufacturers to give advice on desired product improvements. Commonly, manufacturer representatives in these groups urge user-members to make “really difficult compromises,” and to conclude their meetings with an agreed-upon common specification for a new desired new product. After all, as they point out, a manufacturer can not afford to design and build a product unless many users will want to buy.

6

This view by the manufacturers is reasonable from their perspective but can retard the innovation process in the absence of innovation by users. Research shows that innovations wanted by only a few “lead users” today will often turn out to represent general demand tomorrow – *if* lead users have a chance to innovate, learn by doing and perhaps reveal the general utility of their innovations.

3. User innovation communities “shouldn’t exist” – but they do

Manufacturers rather than users themselves have traditionally been considered the most logical developers of the innovations they sell. There are two major reasons for this. First, manufacturers seem to have higher financial incentives to innovate than do individual or corporate users of a product or service. After all, a manufacturer has the opportunity to sell what it develops to an entire marketplace of users. Each userinnovator, on the other hand, can typically expect financial benefit only from its own internal use of its innovations: Benefiting from diffusion of an innovation to the other users in a marketplace would require some form of intellectual property protection followed by licensing – costly matters to attempt, with very uncertain outcomes. Second, invention and development must be followed by production and distribution and field support if an innovation is to get into general use. And with respect to these matters, the manufacturer appears to have major cost advantages over individual users or communities of users. After all, production and distribution and field support of a physical product have traditionally involved large economies of scale. How could users possibly accomplish these tasks in as cost-effective a manner as a manufacturer? Perhaps one might imagine users effectively uniting in a temporary fit of passion, such as the passion felt by many computer hackers today to “beat Microsoft” – but as a stable part of an ordinary economic landscape? – never!

Yet, as we have seen – impossible or not – user innovation development and consumption communities clearly do exist. And, when products they develop, such as Apache server software, compete against head-to-head against products developed by manufacturer-centric methods such as Microsoft and Netscape server software, the former seems to be handily beating the latter in the marketplace. So, they shouldn’t exist but they do - and they even triumph! (As Galileo is said to have murmured after

7

recanting his statement that the earth moves around the sun “And yet it moves!”) What *is* going on here?

4. Conditions favorable to user innovation communities

Complete user-centric innovation development and consumption communities can flourish when (1) at least some users have sufficient incentive to innovate; (2) at least

some users have an incentive to voluntarily reveal their innovations *and the means to do so*; and (3) self-manufacture and/or distribution of innovative products directly by users can compete with commercial production and distribution. A pattern of user innovation followed by commercial manufacture and distribution will occur when only the first two conditions hold.

4.1 User incentives to innovate

Users have a sufficient incentive to innovate when they expect a high-enough benefit from the innovation to offset their costs. Judging from their actions, many users consider these conditions met in both the case of Open source Software code-writing and in the case of sports equipment development. Indeed, the costs as experienced by innovating users can be extremely low or even negative – because users typically report enjoying the development work itself as well as deriving benefit from the innovations that they create. Proof of the pudding is in empirical research that documents the presence of user innovation in many fields – often carried out by significant fractions of the user community (table).

Table: Extent of user innovation in four product categories

Innovation Area No.

Users

Sampled

% Innovating for

Own Use

Were the innovating

users “lead users”?

Library Information

Systems (a)

102 26% Yes

Printed Circuit CAD

Software (b)

136 24.3% Yes

Pipe Hanger

Hardware (c)

74 36% NA

8

Outdoor Consumer

Products (d)

153 9.8% Yes

Sources of Data: (a) Morrison, Roberts and von Hippel (2000),

(b) Urban and von Hippel (1988),

(c) Herstatt and von Hippel (1992), and

(d) Luthje (2000).

Of course, users might not innovate if manufacturers preempted them and supplied what they wanted as soon as they wanted it. So it is reasonable that, as the table shows, many user innovations are found among “lead users” who are at the leading edge of markets and are expecting high benefits from the innovations they desire. Under these conditions, from a manufacturer’s point of view, markets are small and also can be very uncertain. (To appreciate this, recall again the rapid evolution of user need described in Shah’s performance windsurfing example. “I could go so much faster than I ever thought and when you hit a wave it was like a motorcycle rider hitting a ramp – you just flew into the air... All of a sudden not only could you fly into the air but you could land the thing and not only that but you could

change direction in the air!” These learning-by-doing discoveries had a strong impact on the kind of equipment the user wanted.)

4.2 Users’ incentives to freely reveal their innovations

A user innovation community would not get far unless at least some users freely give away their innovations to others. After all, absent free revealing, each user would have to redevelop the same innovation in order to use it – resulting in a huge systemlevel cost – or users would have to protect their innovations and license them and collect revenues from other users – something that would burden user innovation communities with a huge overhead.

It has been shown that users do often freely reveal details of their innovations to other users and to manufacturers in a number of fields (Harhoff et al 2000). This can be seen clearly in the case of Open source software, where users can be seen to post improvements and code that they have written on a OS project websites where anyone – ranging from rival users to manufacturers – can view it and download it for free. Free

9

revealing is also clearly present in the sports innovation example presented earlier. Innovating users would gather together on the beach, inspect each others’ creations and imitate or develop additional modifications that were freely revealed in their turn. But how do we understand such behavior? Free revealing does not make sense from the point of view of conventional economic wisdom. In that view, it is logical that innovating users would attempt to keep their innovation-related information secret. After all, innovating users spend money and time to create their innovations, and revealing of their developments without compensation to non-innovating users – either directly or via a manufacturer - should represent a loss that they would seek to avoid if at all possible. Users will reveal innovations when the costs of revealing are outweighed by the benefits. In the case of user innovation communities, the costs of revealing are generally low, and are adequately offset by even low levels of related benefits. There are two kinds of costs associated with revealing an innovation: costs associated with the loss of proprietary intellectual property, and costs of diffusion. With respect to intellectual property losses, users who have innovated will expect low losses if they have low rivalry with potential adopters. (For example, town libraries have low rivalry – they each serve a different population and do not seek to gain market share from each other.) Even those who would prefer not to reveal due to rivalry considerations will do so if they expect that others will reveal if they do not – a belief that appears to be commonly-held by Open source project participants (Lakhani and von Hippel 1999). Also, of course, users will freely reveal if they cannot hide their innovations. For example, performance windsurfers experimenting on the open beach had no plausible way to hide their technique and hardware innovations from fellow users even if they had wanted to. In both the OS and windsurfing examples, the costs of innovation diffusion are low. In software widespread diffusion of an innovation can be accomplished essentially costlessly online. Code can be posted on a public website with a few clicks on a computer keyboard. In the case of windsurfing, information on an innovation can be widely diffused at *zero* cost to the innovator. Some potential adopters will visit the beach where the innovators are working – perhaps because they have traveled to that site to participate in a contest. They will take the initiative to walk over and study the new

10

technique and equipment and then, when they go home, will transfer what they have learned to their own local user communities.

Benefits from free revealing need not be large under the conditions just described, and are reported to come in many forms, ranging from reputational gains, reciprocity expectations, or incentives to help build a community.

4.3 Innovation diffusion

As was mentioned earlier, user innovation communities can flourish when at least some users have an incentive to (1) innovate and (2) freely reveal, *and* when (3) selfmanufacture

and/or distribution of innovative products directly by users can compete with commercial production and distribution. We also noted that a pattern of user innovation followed by commercial manufacture and distribution will occur when only the first two conditions hold.

In the case of Open source software, “full function” user innovation and consumption communities – no manufacturer required - are possible because innovations can be “produced” and distributed essentially for free on the web – software is an information product rather than a physical one. In the case of our sports innovation example, however, equipment (but not technique) innovations are embodied in a physical products which must be produced and physically distributed in order to achieve general diffusion. Production and physical distribution does, as was mentioned earlier, involve significant economies of scale. The result in the case of our windsurfing example and also in the case of physical products generally, is that innovation may be done by users but production and diffusion of products incorporating the innovations is later carried out by manufacturing firms (Shah 1999).

5. Ongoing exploration of user innovation communities

The advent of the web and the consequent very public proliferation of Open source software development projects has focussed new and intense academic attention on the general phenomenon of user innovation communities in general, and Open source software in particular. The thousands of open source software projects extant represent natural experiments that academics and others can study to better understand the

11

phenomenon. Among the issues being explored are conditions under which Open source projects can be expected to succeed, how they can be most successfully managed, what attracts the interest of volunteers etc. We can now expect very rapid progress on this front.

Of course, even as we study the phenomenon, it is changing. The rationale for user innovation followed by manufacturer production in the case of physical products is compelling and compound user-manufacturer innovation models exist and are evolving. In the field of custom integrated circuits, for example, users are given access to “user toolkits for innovation” that allow them to develop design circuits that are both precisely suited to their needs and also producible in manufacturers’ “silicon foundries.” In the case of Open source software, it is likely that effort volunteered by users will sometimes be usefully supplemented by commercial suppliers like Red Hat and VALinux. But what is very important is that innovation communities exclusively by and for users work well enough to create and sustain complex innovation products without *any* manufacturer involvement. This means that, in at least some and probably many

important fields, users can build and consume and support innovations on their own independent of manufacturer incentives to participate. The user innovation community phenomenon is growing – and, as I have explained, appears to be on solid economic foundations. We shall all see where it leads!

NOTE: Recent readings on Open source Software and on user innovation communities can be downloaded from the website opensource.mit.edu. This resource is intended for all who are interested in keeping updated on - and perhaps contributing to - our understanding of these phenomena.

Sidebar: What is Open Source Software?

Open Source software is software that everyone is free to download and use and modify without payment. The legal mechanism that makes this possible is copyleft and similar legal agreements. The technical mechanism that makes it possible is free access to the “source code” used to create the software (thus, “open source” software). Well-known

12

examples of open-source software are the GNU/Linux computer operating system, the Perl programming language and the Internet e-mail engine called SendMail.

Open source software has its roots in the “free software” movement started by Richard Stallman in the early 1980s. Stallman founded the Free Software Foundation (FSF) as a means to counter the trend towards proprietary development of software packages, and the release of software without the underlying source code. The purpose of the foundation was to encourage development of software that would come with source code and be available to users for their own modification. A key feature of FSF based development is a licensing scheme called GNU General Public License (GPL), commonly referred to as 'Copyleft.' Under GPL, the author of the program has the traditional and legal entitlements of copyright protection along with a license for users to redistribute and change software. The GPL provides unique distribution terms that gives all users the rights to use, modify and redistribute the programs code or any program derived from it but only if the distribution terms are unchanged. Thus the code and the freedoms become legally inseparable. The Copyleft concept prevents private hoarding of free software that would be possible if it were just released under a public domain release .

The philosophy of the FSF movement has been recently extended by a number of individuals who are promoting the 'Open source' concept. These individuals are less concerned about the freeness of "free software" and are instead interested in encouraging software companies to release source code for their products. These individuals believe that companies that release source code, with liberal user rights and licensing, are inherently preferential to closed and proprietary firms (Raymond 1999)

The open and free sharing of information in an Open source project supports the emergence of a community of users with a range of interests related to extending and supporting the initial innovation. For example, some members may find errors that are unique to their situation, while others may write code that allows them to use the product in ways and environments that were unanticipated by the initial innovator. Still other

13

users may be interested in providing support in the mundane areas of trouble shooting and novice user help that may not be of interest to the original innovator.

Many thousands of Open source projects exist today and the number is growing rapidly.

A repository of Open source projects, Sourceforge.net, lists over 10,000 projects and over registered 100, 000 users. Implementing new OS projects is getting progressively easier as effective project design become better understood, and as prepackaged infrastructural support for such projects, such as is provided by SourceForge, becomes available on the Web.

References

Sue, I understand that you don't use references in an SMR essay, but that it is possible to work key ones into the text. I would like to work with the editor to get a (very) few inserted in this way.