

MIT Sloan

Management Review

An interview with Marten Mickos

The Oh-So-Practical Magic of Open-Source Innovation

The Oh-So-Practical Magic of Open-Source Innovation

There are 12 million reasons why Marten Mickos isn't afraid that his rivals in the database software industry will ever overtake him. "Let them try," he says brazenly of his competitors. "Our secret is in the way we operate our culture,

and I'm convinced others cannot imitate that."

Culture is too vague a word. Mickos is referring to the fact that MySQL AB, the business he has built since 2001, has committed itself to "open-source" innovation since its founding in 1995 — with results successful enough that Sun Microsystems Inc. acquired what is the world's fastest-growing database vendor earlier this year for \$1 billion. Like such well-known proponents as Linux, the operating system, and Wikipedia, the online encyclopedia, MySQL shares its source code for free, giving programmers everywhere permission to debug, add features or otherwise modify the product before redistributing it. (MySQL, whose high-profile customers include Facebook and Google, makes money by selling commercial licenses and by offering support and services.) MySQL's collaborative community now consists of 12 million coders, who typically receive compensation that amounts to — in today's dollars — nothing at all. Mickos, a native of Finland, works in the company's offices in Cupertino, California. He first met the company's cofounders when they were all students (and avid poker players) at the Helsinki University of Technology.

Here Mickos, now a senior vice president at Sun, talks with Josh Hyatt for the *MIT Sloan*

Management Review, freely sharing his ideas about why this Internet-age version of a barn-raising produces superior innovation, what murky motivations keep all those developers devoted and why Leonardo da Vinci is the father of the open-source movement.

Why would a company like MySQL choose to give away the guts of its product — was it just a desperation move by its founders to differentiate it from its rivals?

MICKOS: Interestingly, the whole company was started by the founders writing the product code themselves, so it's not like there was a big smorgasbord of contributions from many people. They were thinking of a closed-source product. Then one of the founders saw a presentation about open source and convinced the others that this was the way the world was going to go. That was in the first year, 1995.

Did sharing the source code generate outside contributions right away?

MICKOS: You need a good user base before you start getting contributions. Sometime in the late 1990s, people sent in new features or patches and other stuff. We began to get contributions from the world. But that's not the only way we innovate. We also innovate in-house, we pay others to contribute to our product and we have customers who pay us to create the feature they are looking for. Already I've listed four different ways of innovating. You need to have many arrows in your quiver, and to use them appropriately.

Is there a limit to what outside innovation is good for? Is it safe to assume that people contributing that way are usually adapting existing products — rather than submitting plans for entirely new offerings?

MICKOS: The depth of the contributions varies by product and situation. The deeper you go into the

A conversation with MySQL chief Marten Mickos about the day-to-day realities of making open source work, the new outcomes of enlightened self-interest and why there's no risk that you could rip off MySQL no matter how much they let you see.

core of the database engine, the more difficult it is for somebody to contribute because it takes five years to learn. If you build something on the outskirts of the kernel — some tool or function that you add on top of it — then that is much easier because there's less risk that you will mess up the whole product. But something great can emerge out of many tiny-looking contributions. It's analogous to how, in economic development, microloans can have such a huge impact — each entry is minimal, but when you multiply it by the number of people who are involved, it grows massive. It starts getting a momentum of its own.

What role can MySQL play in focusing all those efforts? If you put a specific problem out there, would the community solve it for you?

MICKOS: If we go out and pose a problem, chances are somebody will react favorably to it. Statistically,

it's likely we'll find someone who has a genuine interest.

What do you do about ineffective contributors who send in ideas that waste your time — or damage a product? How can you prevent your competitors from doing that?

MICKOS: No [competitor] has ever tried. But of course we need to be protected against the risk of that anyway. The fact that you send us some code doesn't mean that it goes into the product. We put it through a quality review. We test it, we make sure the documentation is there and that the coding standards are right, we test it and make sure that it fits our road map. So if we found something that would be harmful to the product, or to the business, we would reject it. We *have* rejected code, but mostly on the grounds of low quality, not because someone was trying to put a Trojan horse into our product.

Given that you might reject the work that they've done for free, how do you keep those outside innovators motivated to even try?

MICKOS: We sometimes use money. There's a part of our product called the JDBC driver, a sizable piece of code, and a guy in the open-source community had developed it. It was his, and everybody loved it. We went to him and said, "Hey, would you like to join us as an employee and sell your code to us?" He made money on it. But that warranted payment. For something small like a bug fix or a tweak here or there, we know what to do because we understand the motivation of the contributors.

And just what drives those contributors, as far as you can tell? Why do they do it?

MICKOS: Those who contribute to us are as selfish as anybody else. There's rarely any charitable aspect of this. Their No. 1 reason is that they are contributing so that they will get, in return, a better functioning product. They are looking after their own business interests. No. 2, and almost equally important, is their desire to build a reputation for themselves. We will publicly announce what they have done. So they get great recognition for being smart engineers and brilliant programmers. That gives them better job offers, and it gives them a feeling of usefulness in the world. They get a lot of emotional and practical benefits. And they have an incentive to make contributions

The New (Open Source) Deal

Right, so tell me again why programmers contribute to open-source products for free?

Contrary to certain folk-cultural wisdom, "Open-source people are not a bunch of Mahatma Gandhis trying to do good for everybody," says MySQL head Marten Mickos. "Those who contribute to us are as selfish as anybody else. There's rarely any charitable aspect of this." Many contributors do it at least partly for fun and to learn new things, but for the most part, claims Mickos, "These are people with practical needs who are coming up with practical solutions. They are trying to improve their own product or their own career."

The four biggest reasons, according to Mickos, that developers contribute to MySQL in open-source ways are:

- 1. To get the product they need.** "Their No. 1 reason is that they are contributing so that they will get, in return, a better functioning product. They are looking after their own business interests."
- 2. To build a reputation.** "No. 2, and almost equally important, is their desire to build a reputation for themselves. We will publicly announce what they have done. So they get great recognition for being smart engineers and brilliant programmers. That gives them better job offers, and it gives them a feeling of usefulness in the world. They get a lot of emotional and practical benefits."
- 3. To prove something to themselves.** "They are trying to prove to themselves how skilled they are. And they have an incentive to make contributions of high quality because they know that others will be examining it; it's a Darwinian system and only the best innovations survive."
- 4. To get satisfaction from seeing their work have effect.** "Our cycle time is very fast. We give them a functioning product back. They see the changed product quickly. If you go to one of the giants and give them a bug report, you typically won't hear back from them."

of high quality because they know that others will be examining it; it's a Darwinian system and only the best innovations survive.

How do you tell the world about an outsider's contribution — by naming a product after them?

MICKOS: In the product, they will be on a list of contributors. And we'll even write a blog posting or an article about them. Every year we announce the Community Person of the Year award. This year we did it at our big conference, during my keynote, with 2,000 people in the audience. In the audience, there were many people who wished that they had been selected. Open-source people are not a bunch of Mahatma Gandhis trying to do good for everybody. These are people with practical needs who are coming up with practical solutions. They are trying to improve their own product or their own career or they are trying to prove to themselves how skilled they are.

But only one person receives the award, and not everybody cares about having his or her name as part of a list that comes with a product; they may figure that their peers will already know. Are there any other tricks to keeping programmers engaged?

MICKOS: We give them a functioning product back, and our cycle time is very fast. They see the changed product quickly. If you go to one of the giants and give them a bug report, you typically won't hear back from them.

What about the in-house innovation you do? Have you found the secret for building that function into the structure of a business?

MICKOS: We don't have an innovation team or a group or a chief innovation officer. Innovation is so central to us that everybody does it. Our innovations are not meeting-centric. We prefer trial and error, getting scrutiny and commentary from everyone. We do have five people whose only role is to engage with the community and activate them, listen to them and get them going. This team has put up Web sites where they congregate and share their code with each other. If you go there [<http://forge.mysql.com>], it will look so geeky you might not really see the value of it.

When you say that this team engages the community — do they ever do it face to face, or is it strictly on a virtual basis?

MICKOS: Virtual, mostly. We are dealing with such a huge number of people. We have over 12 million people in our community. To reach them, we have to use electronic means. We send our people places physically, too. They'll go to big open-source events to get the human touch.

Innovation is so central to us that everybody does it. Our innovations are not meeting-centric. We prefer trial and error, getting scrutiny and commentary from everyone.

What do your MySQL representatives do when they go to these places — do they just walk in and announce that they are willing to pay nothing for any good ideas folks might like to give?

MICKOS: The popular thing now is to arrange an un-conference, and the idea is that it doesn't have a predefined agenda. You look in, and they are dressed in T-shirts, babbling about technical details. It looks unproductive, but it's amazingly stimulating to these guys. We have no budget, so you have to pay for your own travel and accommodations.

Does that mean that you are actually getting people to pay to give MySQL their ideas?

MICKOS: They find the meetings very energizing.

Does innovation always have to be that casual? Is that the only way it can happen?

MICKOS: I think so. I think that innovation happens in encounters with other people and also when you step over some boundary and you combine ideas that haven't been combined before. The original innovation moment always has the aspect of being chaotic.

MySQL is a virtual company on a grand scale — with hundreds of employees working from their homes in more than two dozen countries. Doesn't that hinder the innovation process?

MICKOS: I think it adds the spice to our discussions. When we get together, there are so many dialects and accents that there isn't a norm, so you don't become normalized. We don't all look the same, or dress the same. Lots of good ideas come out of the

fact that we are each so different. We have had to learn how to be innovative across distances, to get things going across multiple time zones.

What was the toughest challenge in learning how to do that?

MICKOS: We didn't hire an innovation professor who would guide us through any of it. We just couldn't afford to travel to each other, so we had to

I'm so confident in our agility, our speed of action and the strength of our culture that even if someone studied us in tiny detail, I don't think they could really match us.

come up with our own rules. At first it was a trial-and-error process. We figured out that we needed to make sure that any one of us could draw something up instantly, and send it to everybody else. It's as if we are sitting in one room together, with somebody taking notes on the wall while we are discussing them. We each get a copy of minutes from the meeting while it's still fresh in everybody's memory. Physical meetings can be very boring and stale. We have this electricity in our virtual meetings. People can get heard even if they don't speak English as well as everybody else; they can comment in writing and get seen instantly by everyone else.

Does collegiality improve innovation?

MICKOS: It's good for all of us if we can connect on a personal level. When we get together — we used to do it once a year, but it has become harder now that we are over 400 employees — we need to do a full day of being social. It's as if we've piled up those 15-minute breaks that people in offices use to be social. We leave knowing each other's body language and feeling more connected. That can only be good.

But you also rely on contributors who are disconnected in that sense, and who may feel free to behave any way they want. How do you control their online conduct?

MICKOS: It is true that people are ruder when they are anonymous on e-mail. We had to develop a culture of behaving well online, internally. External people can be overly critical and very negative. Anybody can speak up, even those who shouldn't

get their voices heard. We tell ourselves never to get defensive and never to get offended. You have to make sure you don't feed those conversations, or you just get more of it.

So what do you say — “thank you and good riddance?”

MICKOS: We say, “Thanks for your input.” Then we remind ourselves that those who are saying they hate us are actually asking for attention. They are saying, “I would love to love you, but currently I cannot.”

That seems like an overly generous interpretation.

MICKOS: Yes, but when you do it, it works. You thank them for their feedback and apologize for not

living up to their expectations. Then you ask them if they have a concrete proposal. When you turn it around like that, they either go silent or they start talking. When they feel that someone is finally listening to them, they can say, “I love you.” In fact, they can become your most passionate supporters. Never underestimate the hidden positive value in somebody who is upset.

But not everybody out there is a secret admirer — or just waiting for the right moment to express their affections. What do you do about those types?

MICKOS: It's true that some of them never get over it. We leave them. Sometimes we don't get anything positive out of it. In that case, we must trust that the world will see what is happening and that it's the same guy complaining again.

Aren't you worried that one of those anonymous people will piggyback on your code and start a business that is aimed at crushing you?

MICKOS: Let them try. I'm so confident in our agility, our speed of action and the strength of our culture that even if someone studied us in tiny detail, I don't think they could really match us. They wouldn't really get it. Even if I showed you my DNA, you wouldn't know how to become me. We've shown them our code, and nobody has been able to measure up against what we have.

Why is that?

MICKOS: If 60-year-olds go to a teenage disco to learn how to dance, they won't be able to dress like

the youth do, or move like they do or enjoy it like they do. What we are doing requires a deep and profound understanding. I think you should always be open to learning from everybody, but you must build your own DNA.

Why is it that a growing number of companies are enamored with building their own DNA out in the open?

MICKOS: Some people believe that open sourcing is a panacea. They say, “We’ll open-source this and something will magically happen on its own.” That’s just completely wrong. The only way to get engagement from a community is to engage in that community yourself. It’s not like there is some saint out there waiting for you to open-source so he or she can contribute. A lot of software created by open source has no commercial value. Someone just created it for fun.

Is that why open sourcing seems so often linked with non-profits like Apache, Linux and Mozilla?

MICKOS: I wouldn’t be surprised if the proprietary companies are trying to portray it that way to reduce the competitive pressure on them. They tell

their customers, “Those guys are nonprofit communists, so don’t touch them.” The activity level of the community can help a small player become a threat to established companies without having to invest millions and millions of dollars to get brand recognition. People start writing blogs or they write books about it and they help others get used to your product.

Are we all getting too used to open sourcing? Are you concerned about it becoming a fad that everybody gets sick of?

MICKOS: This idea is not new. I don’t think innovation has changed in thousands of years. Leonardo da Vinci may very well have innovated just like we do. The difference between what he did and what we do isn’t in the openness, it’s in the reach. He didn’t have the Internet. So he sat down with his buddies and drank wine to get the juices flowing. That interaction is the essence of innovation.

Reprint 50109.

Copyright © Massachusetts Institute of Technology, 2008.

All rights reserved.

MIT Sloan

Management Review

PDFs ■ Reprints ■ Permission to Copy ■ Back Issues

Articles published in MIT Sloan Management Review are copyrighted by the Massachusetts Institute of Technology unless otherwise specified at the end of an article.

Electronic copies of MIT Sloan Management Review articles as well as traditional reprints and back issues can be purchased on our Web site: sloanreview.mit.edu or you may order through our Business Service Center (9 a.m.-5 p.m. ET) at the phone numbers listed below.

To reproduce or transmit one or more MIT Sloan Management Review articles by electronic or mechanical means (including photocopying or archiving in any information storage or retrieval system) **requires written permission**. To request permission, use our Web site (sloanreview.mit.edu), call or e-mail:

Toll-free in U.S. and Canada: 877-727-7170

International: 617-253-7170

Fax: 617-258-9739

e-mail: smrpermissions@mit.edu

Posting of full-text SMR articles on publicly accessible Internet sites is prohibited. To obtain permission to post articles on secure and/or password-protected intranet sites, e-mail your request to smrpermissions@mit.edu.

Hyperlinking to SMR content: SMR posts abstracts of articles and selected free content at www.sloanreview.mit.edu. Hyperlinking to article abstracts or free content does not require written permission.

MIT Sloan Management Review
77 Massachusetts Ave., E60-100
Cambridge, MA 02139-4307
e-mail: smrorders@mit.edu