Conversations with a Prominent Propagator: Jens Mönig

David P. Bunde
Zack Butler
Christopher L. Hovey
Cynthia Taylor

Encouraging faculty to adopt new, high-impact teaching practices, tools, and curriculum in computer science (CS) undergraduate education requires intentional planning and sustained effort. This article is the next installment in the series of interviews [1–3] with prominent propagators: members of the CS education community who have successfully spread pedagogical or curricular innovations. The goal is to capture knowledge and experiences that others can use to propagate their own teaching projects.

For this article, we interviewed Jens Mönig. Jens is a researcher and designer at SAP and the architect and lead developer of Snap! [6], an extensible block coding language that is used in UC Berkeley's Beauty and Joy of Computing curriculum [4, 5] as well as the basis for many open-source projects. He previously helped develop Scratch with the MIT Media Lab. Jens also holds a law degree and was a lecturer at the University Hospital of Tübingen in medical law.

Below are highlights of the interview, which ran approximately 1.5 hours. The transcript has been edited for clarity and style.

Q: How did you come to develop Snap!?

JM: I'm not a computer scientist by training. I was educated as a lawyer, and I was practicing law in my own private practice until 2008, when there was an economic slump. I figured I'm not having fun anymore doing law, but I really loved programming Smalltalk for IBM back in the '90s. I started googling around for Smalltalk. I found that there was one project that was still using Smalltalk, and that was this funky thing at the MIT Media Lab called Scratch. I was blown away by the sheer beauty of it: not by what it was doing, but by how good it looked. Up to then everything I had written in and seen in Smalltalk looked bad, and I wanted to know how John Maloney [lead developer of Scratch] had done it so nicely.

I knew from my time at IBM that usually if you find some bug in a Smalltalk program, you get a debugger. Almost immediately, I got an error message and a debugger, and I could look into how Scratch was made. I dug into this system and I started adding little things to it. I added a little code into a hidden block and now you've got a Scratch that has lists. And I started blogging about it.

Then I got a call from Mitch Resnick [founder of Scratch] and he invited me to the first Scratch conference. Once there, I was in heaven. I was like, "Oh gosh, here's my heroes." So at the end of the conference, when Mitch asked me if I would like to be part of the Scratch team, I was like, "Would I like to play guitar with the Beatles? Of course!" This was an absolutely mind-boggling thing, to be this lawyer, kind of depressed about my job, contributing to Scratch.

And so my job with Mitch was the best job ever. Mitch was like, "Oh, Jens, why don't you invent cool new features for Scratch? One thing we've been thinking about is that everybody would like to build their own blocks." I made the first prototype of what I called BYOB (Build Your Own Blocks). Dan Garcia at UC Berkeley got in touch with me and said, "We really love this thing that you built, this prototype. Would you mind if we use this for a new course?" I said, "Great!"

2153-2184/2025/MonthOfPublication - ArticleNumber

http://dx.doi.org/10.1145/3773090

They wanted some more features. We added actual lists that John Maloney did that were much better than mine, and text functions like joining and spitting text. Then I got this email from this other professor at Berkeley, Brian Harvey, who said, “We need one more thing and that's lambda.” I was a lawyer; I'd never heard about lambda. I'd never heard of functional programming. I was a Smalltalker. Brian and Dan taught me all about it. So we started adding these features as they were building the first curriculum. Eventually, we realized we had started a new programming language. At that time, we were getting to a point where we couldn't fit it into this hacked up Scratch virtual machine anymore. And at that time the Scratch Team, Evelyn Eastmond and John, decided to rewrite Scratch.

Q: Is that when Snap! became separate from Scratch?

JM: In 2012, the BJC (Beauty and Joy of Computing) course really started taking off. That was when the first people started actually using not BYOB, but Snap!. Since then, Brian and I have added things that came up in BJC and things that we thought would be cool ideas. Then John's former boss, Alan Kay [lead developer of Smalltalk], invited us all over. He said he knew this guy from Germany who had a software company called SAP, and he sweet-talked him into getting the band back together and funding it for a couple of years. And would I like to join? That was another one of these moments. Everybody gets to do what they love. This is how I got to SAP, and parts of the Scratch team ended up here.

But we sort of were in a lurch at SAP. The guys who hired us left the company, and we were an orphaned group. I talked to the CEO, Bill McDermott, and he said, “Jens why don't you just keep doing what you were doing? It's good for SAP.” And so this has been an incredible streak of luck. I've been doing this at SAP now for 10 years after our group kind of got disbanded. And they hired two more people who are working full-time on Snap!. We can keep Snap! unbranded and we don't have to use any SAP software. We get to hang out with our friends at Berkeley and we get to hang out with all the teachers and to help spread the beauty and joy of computing, which is dear to our heart.

Q: How do you convince SAP to keep funding the project?

JM: It's a constant struggle, resisting being managed by SAP people and insisting that we need the freedom and we are going to deliver. What really helps is measures of success and one thing that helps is authority. Back when they were close to firing me, one thing that helped was a letter from the University of California at Berkeley using an official letterhead. Brian Harvey agreed to sign this as Professor Dr. Brian Harvey. We do a lot of name dropping, “Oh, it's Ken Kahn, University of Oxford, using it.”

A lot of it is also numbers. Because we have this cloud and the community website, we have the numbers of projects and signups and activity. In a good year, we have a hundred thousand new users and a million new projects. Working at an enterprise software company, these numbers are fantastic because enterprise software doesn’t have that many unique customers. So our product has more users than most SAP products have customers.

Q: If you have no revenue, how do you put that into corporate language?

JM: Totally no revenue. Every two years they ask me to write PowerPoint slides outlining the benefit to SAP. Currently the story is that Snap! is SAP's gift to computing education, and SAP flourishes in a climate that is appreciative of computing. Being Europe's biggest software company, we need the environment of people understanding and appreciating computing.

This is a message that resonates with them. We also try to do a lot with teachers and government groups. A professor from Göttingen wrote a book about AI for German schools, grade 10 and up, about everything from perceptrons to multilayer neural networks. She'd written it so they're going to use Snap! for everything. Stuff like that helps us tremendously when I can show this to my boss and say someone's writing books about this stuff that we do. They're using Snap! in schools, which is why it's great that it's free.

Q: What role do you think industry in general should play in funding educational innovations?

JM: I know Neil Fraser, who's worked on Blockly. I know the gang who's doing MakeCode at Microsoft. Same for Google. Michael Kölling and Greenfoot are being funded by Oracle. Greenfoot at least is in a university context. But there is not enough of that kind of software. And we are always having these discussions about how to sustain it. I really feel that the industry has a duty to invest — as they used to do. I think they're getting their money's worth out of it.

Why is Michael Kölling getting a hard time building the most successful Java teaching environment in the freaking world? Isn't that enough? Why isn't that an actual accomplishment? But he has to apply every year.

I don't understand why funding isn't discussed more openly. I'm very disappointed by the role of the industry in this, because they're only doing it if they're gonna sell it or use it to bait-and-switch people to their products. But I'm also really disappointed by universities. Bootstrap is nice, but that's like, what, Rice University and Brown University? They should all do it.

Q: Do you feel like there's a pressure to constantly implement new things to continue being funded?

JM: I'm constantly being asked, "what are the new features you're planning to release?" Not that anybody would understand when I tell them about things like first-class colors. I actually think we're the only ones that I know who are long-term consistently funded.

These business people are going to thank us later when their kids say, "I learned something today in school using this cool thing from SAP." Then business people notice, "Oh wow, we're having an impact in society." So you have to force them into doing this because their first impulse will be to cut funds, but everybody loves education because it opens minds. You just need to give them time.

Q: You hold a conference called Snap!Con [7] every year. How does that contribute to building community and spreading your ideas?

JM: The community idea is something that we are actually still struggling with. Our natural community is the cohort of BJC teachers, but they have their own mailing lists and they're in the US. We've got the internet, but there's time zones and there's language barriers, and there's also barriers in terms of curriculum. What is being taught in American schools is very different from what is being taught in German schools. So the idea was to find communities, to find people to actually meet in person. For several years, we had a website where people could share and submit. We had forums, but that wasn't really meeting in person. We used to sort of piggyback on the Scratch conference. Every kind of blocks enthusiast used to meet at the MIT Media Lab when they still had these conferences in person.

There was this Dutch math teacher, Joek van Montfort, who organized a set of European Scratch conferences. Joek was always very adamant to be as inclusive as can be. It was educators, enthusiasts, researchers just getting to show their stuff; more like demo day, more like, “Let me show you this wacky thing I'm building.” Much of that was physical stuff, almost a crossover with Maker Fair. And so we had three of these European Scratch conferences all organized by Joek that were official Scratch conferences.

It was really a close-knit European Scratchers community, the same community that is now coming to the Snap! Conferences. Most of them are either active or retired math teachers, computing teachers or professors, lecturers. Some are just people who love tinkering with hardware. It's not a terribly big group. Then there's all these dialects that happen where people, you know, fork Scratch or fork Snap!. And there are so many Snap! forks, which is one of the parts that I really enjoy very much. People use it and can do things.

One of my favorite Snap! forks is Turtle Stitch [turtle-like code for electronic embroidering machines] by Andrea Mayr-Stalder and Michael Aschauer from Vienna [8]. I met Andrea when she presented her first prototype at one of these European Scratch conferences. I was like, “This is the coolest thing I've ever seen!” She was afraid because she “ripped off” Snap!, and I said “No, it's fantastic!” We've been hanging out together ever since. I use it when I'm working with kids because everyone gets something to take home. You can embroider something on your bag and you can have your name on it or your pattern. It's stitched quite quickly. I love it.

Q: How did European ScratchCon become Snap!Con?

Joek couldn't do any more European Scratch conferences. We thought, maybe SAP should organize something. There is a teacher's college near SAP that had been enthusiastically using Snap!. They agreed to collaborate with hosting it, and so we had the first Snap! Conference here in 2019. It was really wonderful because all these people who used to come to these European Scratch conferences came.

Then Covid struck. We couldn't have it in person anymore, so we decided to do it online and hybrid, which was great because then the Americans could also participate better. So now we're always kind of shooting for something where in the morning we do some onsite workshops, but we're doing the broadcast part about keynotes in the afternoon, and it goes into the night. Then we found out that there's also people in India and Indonesia and Africa and Australia. It's amazing, but make no mistake, we're a small group.

Q: Has Snap! being open source been important for getting different groups to use it?

JM: Absolutely. Free and open source is the key to everything. It's sort of open anyway because it's written in JavaScript. Anybody could see it. I mean, I could go and try to obfuscate it, but that's not the point. I'm actually trying very hard to build in things that make it explorable the same way as Smalltalk used to be. You can drill down, you can see the code, you can play with it live and change it and mod it and do things that you want.

What I've always loved about Smalltalk is that the world could be a different place. There could be another way we use computers, where the ideas behind how we design a system aren't just in some document, but the system itself is explorable and can teach me about how it is built, and how I can turn it into something different. I'm very much trying to build this idea into Snap! itself. So open source is very important, but also trying to make it possible for people to reuse the system and to make it theirs.

Q: What are some of the challenges of making it open source?

JM: To me, the most courageous thing this old Smalltalk group did, and also John Maloney did with Scratch, is sharing their code, because people are going to pore over it and they're going to ask questions. I often can't answer questions because I've forgotten how it works, or I've just fidgeted around until it worked. Maybe there once was a plan, but I've forgotten about it. Building a system and sharing it, the criticism you get from people is sometimes crazy.

There are parts of open source culture that are toxic and can be extremely discouraging. But I think it's really important to share things, to be open and to be honest about it. I'm often discouraged by people who are impatient with me not learning stuff fast enough. I respect everybody who puts their code out there, because wonderful things are going to happen.

Q: How does your work with Snap! include curriculum development?

JM: There used to be this part of BJC about data, so I tried to come up with examples. Of course the interesting data is all personal data, but that's problematic. You can use dead people's data, so we love using the Titanic passenger data. We do a frequency analysis of the first names. [We asked] what are the top 10 male names? And it turns out, the number one favorite name was William, number two was John. Then we said, let's do the top 10 female names on the Titanic. And that blew my mind: the top number one female name of the Titanic is William. Back in 1912, married women were not able to enter a contract in their own name. So they were traveling under the name of the husband. But the real fun is to look at the names of the people and ask students, "What's wrong?" It turns out there is actually a bug in society, not a bug in the data. Data is a time capsule, and even if it's "objective" data, it still captures stuff.

We sometimes had an assignment idea up and running that we could demo very quickly, but then we took the liberty—and SAP gave us the liberty—to keep iterating over it, even for years. I can't stress enough how happy I am with that because I can come up with something right away over the weekend that is going to have all the important learning goals in there. But in order for it to actually work with a room full of kids, or with teachers, or with teachers who aren't programmers yet, we found that it often takes literally years to iterate and sometimes to come up with a totally different idea.

Q: What does success look like for you?

JM: That's a very good question. I constantly need to report success to people because I'm working at a corporation. All the success metrics that academics have, I don't have. Success to me is when somebody like Mark [Guzdial] loves an idea so much that he uses it in his course. This to me is something that I can show my boss and say, "You think we're just having fun, but here's University of Michigan, which is on the list of rankings that you always look at." I'm really sad to say, but this kind of authority helps.

Once I was visiting UC Berkeley, and I was strolling around the campus with my boss and Dan, and there was this guy coming up to me, taking my hands, "You must be Jens. You've changed my life." And I couldn't hold back the tears. To me it was the most awesome success. To me, success is when people are taking or feel inspired to share their ideas, and are able to express them using the language I make.

References

[1] Bunde, D.P., Butler, Z., Hovey, C.L. and Taylor, C. 2024. CONVERSATIONS: Conversation with a Prominent Propagator: Susan H. Rodger. *ACM Inroads*. 15, 4 (2024), 9–13. https://doi.org/10.1145/3699720.

[2] Bunde, D.P., Butler, Z., Hovey, C.L. and Taylor, C. 2025. CONVERSATIONS: Conversations with a Prominent Propagator: Amy Ko. *ACM Inroads*. 16, 3 (2025), 26–30. https://doi.org/10.1145/3756325.
[3] Bunde, D.P., Butler, Z., Hovey, C.L. and Taylor, C. 2025. CONVERSATIONS: Conversations with a Prominent Propagator: Philip Guo. *ACM Inroads*. 16, 1 (2025), 17–21. https://doi.org/10.1145/3706588.
[4] Garcia, D., Harvey, B. and Barnes, T. 2015. The beauty and joy of computing. *ACM Inroads*. 6, 4 (Nov. 2015), 71–79. https://doi.org/10.1145/2835184.
[5] Goldenberg, P., Mark, J., Harvey, B., Cuoco, A. and Fries, M. 2020. Design Principles behind Beauty and Joy of Computing. *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (New York, NY, USA, 2020), 220–226.
[6] Snap! Build Your Own Blocks: *https://snap.berkeley.edu/*. Accessed: 2025-09-17.
[7] Snap!Con: *https://www.snapcon.org/conferences/*. Accessed: 2025-09-17.
[8] TurtleStitch - Coded Embroidery: *https://www.turtlestitch.org/*. Accessed: 2025-09-17.

David P. Bunde
Knox College
2 E. South St
Galesburg, Illinois 61401 USA
dbunde@knox.edu

Zack Butler
Rochester Institute of Technology
Rochester, NY 14623 USA
zjb@cs.rit.edu

Christopher L. Hovey
University of Colorado Boulder
1045 18th Street, UCB 315
Boulder, CO 80309 USA
hoveyc@colorado.edu

Cynthia Taylor
Oberlin College
10 N Professor St
Oberlin OH, 44074 USA
ctaylor@oberlin.edu