

# The Open Source Way

Episode 15: Compiler History – The Open Source  
in Your Favorite 1990s First-Person Shooter



## Transcript

**Karsten:** Welcome to the Open Source Way. This is our podcast series, SAP's podcast series about the difference that Open Source can be. In each episode we'll talk with experts about Open Source and why they do it the Open Source Way. I'm your host Karsten Hohage, and in this episode I'm going to talk to Jack Schueler about Watcom and probably about many things from back in the days, because Jack has been in the industry for, let's say, a while. He was and is part of a very long running compiler project, the Watcom C and C++ compiler that is, that was made Open Source already back in 2002. You never heard of it? You'll find out why you should have maybe. Today Jack supports customers by investigating and resolving the tough issues, and he also supports the SQL Anywhere documentation team at SAP. His data say that he has been with SAP for 40 years. I said 40. Let's find out in the course of what's coming how that came about. And in case I may not be able to follow Jack into the depths of technology here, we're also joined by Sebastian Wolf. He's a developer, as opposed to me, and a member of the SAP Open Source Program Office. So welcome, Jack, and welcome Sebastian.

**Sebastian:** Thank you very much.

**Karsten:** Hello. Okay, let's start with Jack and Watcom. Jack, what exactly is Watcom?

**Jack:** So Watcom is a company that was started in 1981, basically a spinoff from the University of Waterloo. Watcom is a contraction for Waterloo Computing Systems Ltd., that was the official name of the company, but we always refer to it as Watcom and our product names use that term. You know, in 1981, our primary focus was on teaching computer languages, and later we became interested in databases and started a little project called Watcom SQL. And Watcom SQL eventually got the attention of an American company called PowerSoft, who merged with us, and, a year later, we joined Sybase.

**Karsten:** Before we go deeper into the history, let's maybe first figure out what exactly the part of all this that was open sourced in 2002 was.

**Jack:** Since our focus turned to SQL, we decided to divest ourselves of the languages, the compilers that we had been working on, so primarily that was the Watcom C, C and C++ compilers and also a Fortran compiler.

**Karsten:** The C++ compiler and the Fortran compilers are the ones that were open sourced in 2002 and are still around as an Open Source project. Right?

**Jack:** Yes, so that's the compilers, linkers, debuggers, editor; all kinds of tools associated with developing software.

**Karsten:** Now that we've got that down, we can come back to how this all came to life and how you have 40 years at SAP in your bio there.

**Jack:** In the late 70s, I was working for the computer systems group at the University of Waterloo, and we were doing research in microcomputers. And one of the focuses of the computer systems group was teaching computer languages to students.

**Karsten:** What kind of languages would that have been in the 1970s, as you just said? I mean, like Python wasn't around yet, I guess.

**Jack:** There are a few languages, so by 1980 we had developed interpreters for programming languages called APL or Basic, COBOL, Fortran, and Pascal. Those are, you know, say five of the popular languages back then.

**Karsten:** Okay, at least Basic and Pascal I still do remember; Sebastian, do you know all of them? I mean, I've heard of Fortran, but I've only ever seen Pascal and Basic, I think, in action.

**Sebastian:** Action would be too much, but at least I've heard of most of the programming languages. APL was completely unknown to me, to be honest. So, yeah, that's computer history.

**Karsten:** What was the specialty of APL, for instance, there?

**Jack:** APL was basically, I guess, a very mathematically oriented language that operated on vectors, it used a lot of Greek symbols. So, one of the interesting things was that you had a totally different keyboard. So back then you tended to have a plastic overlay that you would set over top of your keyboard that had all these different Greek symbols, Greek looking symbols that were actual operators that did things like, you know, inverted a matrix or, you know, or rotate or transpose the matrix in some way and, you know, add two things together. And that kind of thing.

**Karsten:** And for that, you use all kinds of Sigmas and Phis and Chis and whatever?

**Jack:** Yeah, yeah, basically.

**Karsten:** Okay. I would have had a hard time figuring out how to get that out of my keyboard.

**Jack:** The fun challenge with APL was to write an entire program in one line. So you would proceed from left to right, the whole the whole application was written in one line with a lot of symbols. And, I mean, it wasn't easy to understand or decipher what this program actually did.

**Karsten:** It's not what you consider a well-structured code nowadays, right?

**Jack:** Not at all.

**Sebastian:** You know, there's a popular saying that if it was hard enough to write, it should be as hard to read.

**Jack:** Yes.

**Sebastian:** Some people still apply that these days in their programs, unfortunately.

**Karsten:** Okay, so that was out of the University of Waterloo back in the 70s. I did a little bit of reading there. I think the professor was Wes Graham, who was one of the first ones who kind of saw that it was important for the industry to invest money into University Institutes. Right?

**Jack:** Yeah, Wes was one of the pioneers in computer science and computer literacy, getting students access to computers and letting them use computers in their work and learn about computers. So, Wes was the head of the computer systems group for which I worked. It was primarily Wes and a few other professors that got us looking into microcomputers as a new way of introducing students to computing and a way of facilitating the learning of computer languages.

**Karsten:** And how did this lead to the Watcom project then?

**Jack:** By 1980, we had developed all this software, including a microcomputer system based on the Motorola 6809 processor. The whole works, basically a microcomputer with a bunch of programming languages that it could run and an operating system platform that it sat on. So, one of the nice things about the University of Waterloo is that they're pioneering things. I think at the university any intellectual property was owned by the professors, the students, staff who developed it. So, we were free to take all this hard work that we had done and try to commercialize it. So, in 1980, we decided to create this new company called Waterloo Computing Systems Ltd., or Watcom, and take all this work that we had done and try to sell it to the industry, and one of our first partners or first investors or our first contracts was with Commodore Business Systems. Commodore had developed a PET computer, so we showed them how we could take the computer, install a second motherboard into the computer, and with the flip of a switch, switch into a totally different platform that was based on a 6809 processor and have all these programming languages available. And this would be a wonderful thing to sell to universities and to High Schools even, because the machines were fairly inexpensive. So, it looked like it was a good business opportunity or at least we sold them on it as a good business opportunity for Commodore.

**Karsten:** That Commodore machine by generation or by size, how would that relate to what I had as a kid? I had a C-64.

**Jack:** The C-64 was quite small, the Commodore machine had an integrated keyboard and screen, so it was bigger, but it sat on a desktop. It was about the size of old CRT cathode ray tube monitors that people used to have on their desks, about that size. Lightweight enough that you could carry it around from one room to the next, but...

**Karsten:** But not much further.

**Jack:** Yeah, a little bit heavier than a Commodore 64.

**Karsten:** Well, what was your first computer, Sebastian? As I said, we had a C-64 with the floppy disk right in the beginning, even a cassette drive for data, I think.

**Sebastian:** Yeah, so my personal first computer was actually an AMD Athlon 500, so much later. So somewhere in the beginning of the 2000s, I suppose, but well beforehand, of course, I had some access to my father's computers. And he actually started with a Schneider CPC running with CPM, if I recall correctly. That was my first interaction with computers.

**Karsten:** Okay.

**Jack:** Before the Commodore 64, there was something called the VIC 20, and we had one of those said at home.

**Karsten:** Okay, so the business ran in the family, basically, is that it?

**Jack:** I don't remember, I think the VIC 20 came later. I probably convinced my family that all these, you know, microcomputers were really interesting and that they should try this thing, the VIC 20. It was very inexpensive and you could play games on it.

**Karsten:** I was just going to say that that was you, not your father like in Sebastian's case, because I think with us it was my brother who actually got the C-64 in the house and even our first gaming console also. You just said what you did for the flip over, switching off different main boards in these Commodore computers, taking your IP from Waterloo University into basically a university spin off, right. The company was the Waterloo computer system. So, this is where Watcom comes from. I hadn't figured that one out yet. Now I know. And now how does the C++ compiler story go?

**Jack:** The initial software that we had created convinced Commodore to invest in were all language interpreters. The software was all written in a language. It wasn't C, it was

called WSL – Waterloo Systems language. So, it was kind of like a blend of Pascal and C. And at some point, for the IBM PC, different companies started to come out with C compilers. So, I think there was one from Borland, from Microsoft, there were some other small companies that were developing C compilers, and we looked at them and looked at the kind of code that were they were generating, and we said, oh, I think we could do better than this. The code we are generating from our WSL compiler was much better. And so, we thought, well, why don't we write a C compiler using WSL. So, we did that, we basically got a C compiler going in WSL, and then we translated that into that WSL code into C code and basically bootstrapped our way into having a C compiler that could compile a C compiler.

**Karsten:** Okay, where are we in time when you were parting from WSL to C?

**Jack:** 1983, something like that.

**Karsten:** Okay, so that's the very start of the Watcom compiler then.

**Jack:** So, we got the C compiler going, we also had a strong interest in Fortran back then, so we developed a Fortran compiler as well. And then C++ was coming on strong. So, we developed a C++ compiler. We liked the results that we were getting from our compiler. We were producing faster code and benchmarks than some of the other compilers that were out there. So, we decided to turn that into a commercial product, so we started selling it. We hit lots of companies like PC Magazine and Dr. DOBs, back then, saying that we were doing benchmarks and saying, oh, hey, folks, you should check this out, this compiler here is producing tighter, faster code and blowing away the competition. So that's what got a lot of the personal computing world interested in what Watcom was doing.

**Karsten:** You started with C, you continued with C++ and then you got the commercial world interested in Watcom, where are we in time now?

**Jack:** I think 1984, 1985. We were seeing reviews of the code being produced by the compiler. This is the 8088 platform, so this is 16-bit. Later in the 80s, Intel came out with the Pentium processors, much more powerful processors, and this is what we call 32-bit processors. So, they had a different instruction set that was a superset of the 16-bit

instructions set. So, we were convinced by a company called Novell Networks. They were very interested in us developing a 32-bit compiler. The incentive was there, provided by Novell, we did it under contract with them and started developing code generator for the 32-bit platform.

**Karsten:** That was, as far as I know, in the very late 80s, when the 32-bit compiler was to be released. All right, okay. And there we basically reach a point where things got really interesting to me because from the late 80s into the early and mid 90s, there was a special part of the industry that was very fond of the Watcom compiler. Jack, I hope you know what I mean.

**Jack:** Yeah, so the technology that was implemented to run 32-bit applications on the disk operating system, DOS, which was a 16-bit system, was called the DOS extenders, DOS extenders were basically implemented to allow you to run 32-bit applications on a 16-bit DOS platform. And we had companies like Rational that developed a DOS extender. That, combined with the fact that we were producing really good code, got the interest of game developers. And so, one of our very first customers were id software. If you don't know, id is responsible for games like Doom and Quake.

**Karsten:** It seems like Watcom was behind all of my most favorite games of the 90s. So, it was Doom, it was Quake, it was Descent, it was Tomb Raider; apparently they were all based on the Watcom compiler, with their core game engines, I found that really fascinating because I had been in contact with Watcom indirectly without even knowing back then.

**Jack:** One of the fun things that I did with my son when he was very young, I think maybe nine or ten years old, is he was interested in Doom. I showed him Doom and he was able to run it on his computer. And I said, dear, let me show you something interesting, and I ran what we call the strings tool on the binaries for the Doom game. And it comes out with the string copyright Watcom Systems.

**Karsten:** All right.



**Jack:** And it's the only copyright notice that's embedded in the game because the game, of course, embeds the C or C library. And our C library has a copyright message in it. So, I used to joke with him, see, that their dad developed this game, but...

**Karsten:** Yeah, although I'm not really sure if we can broadcast this, that you played Doom with your nine year old son, I think back in Europe it was indexed.

**Jack:** Oh, really? Yeah.

**Karsten:** Yeah.

**Sebastian:** I think it was only in Germany.

**Jack:** Maybe he was ten, I don't know.

**Karsten:** Are you even a gamer yourself, Jack?

**Jack:** Well, I would say not really, because I'm really busy, but I have to admit that the only game that I actually ever played seriously was Doom, Doom is a game where you have to progress through many levels to get to the end. And of course, the more levels you go through, the more challenging it becomes. So, I don't know if it was because my co-workers were managing to get all the way to the end, but I was determined that I could if they could do it, I could do it. And so, I did play Doom right to the end. And I think that was it for me. I fulfilled all my desire for gaming.

**Karsten:** So, you fulfilled the ambition to take it to the end and then that's it. I don't even know, Sebastian, are you a gamer?

**Sebastian:** I was a gamer, I would say, quite some time ago, and actually during the days when I got my first own computer, I was also very much into the first person shooter genre, so to say. Later on, I switched to strategy games. And these days, probably similar to Jack, very busy, other interests. I also develop in my spare time and so I'm not that much into gaming anymore.

**Karsten:** Same here, by the way, I actually worked for a small game's magazine in the second half of the 90s and at that time developed a severe addiction, I would say. And I totally, almost totally quit in the early 2000s. But we had just reached 1990 and the 32-bit compiler there; how did this all, in the end, or how did you, in the end, come to SAP?

**Jack:** In the early 90s, we switched focus from compilers to the SQL database and we developed something called Watcom SQL. I think it was somewhere, perhaps around the fourth release of the product. There was a company called PowerSoft that had a tool called PowerBuilder that was basically like a tool for developing a client application, and PowerBuilder interface to databases like SQL Server. And I think what was then the Sybase SQL Server became Sybase Adaptor Server and Enterprise. So, they had this tool that worked with databases and they thought, "Wouldn't it be cool if we could not only provide the tool for developing the database, the front end, but also include a database in the product as well, so that you have a complete package, a complete solution?" And that's, I guess, one of the motivations for them acquiring Watcom and for us merging with PowerSoft. I think that was like February '94, and by February '95, we were acquired by Sybase. Sybase was interested in our database expertise as well. So, SQL became part of the product offerings of Sybase.

**Karsten:** I think SAP acquired Sybase in 2010. I noticed one thing: I mean, some companies are like gone because of chains of acquisitions, like, for instance, PowerSoft doesn't exist anymore. But the name PowerBuilder, that's still around, right.

**Jack:** Yes, by virtue of the acquisition, Sybase owned PowerBuilder, and in fact, PowerBuilder was part of the product offerings for quite a few years. It was sold to a third party company, so someone else now looks after power builder.

**Jack:** We actually still support the use of PowerBuilder in our products.

**Karsten:** In the SAP products, that is. And SAP, it seems, gave you credit for, like almost your entire career, when, you know, when all your address book entry says your 40 years at SAP...

**Jack:** Yes. Fortunately, SAP has been around quite a bit longer than 40 years, so it's not inconceivable that I started with SAP. But in truth, I have only been working for SAP for since, I guess, 2010.

**Karsten:** And as we found out, some products that you may have been involved in have been inherited all through this entire sequence, so that's cool. And I think when we talked before, you said that despite these changes of company names and everything, you never had to move office, or only once.

**Jack:** Unlike most people who move from one company to the next, they generally move offices and maybe even move to a different city. And my case, I have only ever moved offices basically once from one street in and near the University of Waterloo into the research park at the university.

**Karsten:** Most of us at SAP don't even have to change companies to be moved around offices every one or two years, I can't count how many different offices I've had.

**Sebastian:** So, it's maximum one to three years, I would say, absolutely. And by the way, the PowerBuilder companies, Appeon, they basically signed an agreement with SAP about five years ago.

**Karsten:** And they have taken over responsibility for PowerBuilder?

**Sebastian:** Support and maintenance.

**Karsten:** So how did the Watcom project continue up to today?

**Jack:** As a result of the acquisition by Sybase and our focus on database software, we decided it was time to let the compiler development business go. It was becoming less profitable. Some of the practices of our competitors were somewhat cutthroat, like giving away compilers for free. It's hard to compete with free, no matter how good the quality of your code is. We decided it was time to get out of the compiler business, but we had a lot of fans and the fans didn't want to see the project die. So they offered to take it over and said, why don't you take it Open Source. You know, that was an interesting prospect. And so we went ahead and did that.

**Karsten:** Was that after the SAP acquisition or still before?

**Jack:** It was while we were part of Sybase.

**Karsten:** Because I seem to remember, like in the early 2000s, that the Watcom compiler became Open Source, right.

**Sebastian:** The announcement was done already back in 2000, the day when I got my first computer, by the way, and the actual open sourcing was then done a few years later, early 2003.

**Sebastian:** The first public Open Source release under the special license.

**Karsten:** That's what our official encyclopedia says, right.

**Sebastian:** I also looked it up in the Perforce repository, so the 1.01 release was early in 2003, which was basically the first public open Watcom release.

**Karsten:** And as the Open Source project, it is still around, right?

**Sebastian:** As the Open Source project it's still around, yes, and that's also quite an interesting thing about this one, there was the latest, let's say, official open Watcom release 1.9. And there's even a community fork, which is Open Watcom 2.0, which is still actively developed these days. And yeah, that's also one of the reasons why we are talking, because the community reached out to us to discuss a potential license change, to simplify the use of open Watcom and other Open Source projects. And yeah, we are, of course, not promising anything right now, but we are actively looking into that with the community. It's still around and many projects are still using it even for areas which are pretty vital to certain projects and certain areas in the industry.

**Karsten:** Is it still involved in gaming engines?

**Sebastian:** At least when you compile it in DOS and FreeDOS, so to basically make the classic games, I wouldn't say old games, the classic games run in a FreeDOS environment and FreeDOS itself, if I recall correctly, is still there and still vital.

**Karsten:** Okay, but probably not for the newer games anymore, which all rely on the levels above Microsoft Windows graphic engines and so on, right?

**Sebastian:** That's certainly what I know.

**Karsten:** Why do you have to change the license? Is it on some kind of older license or what's the problem?

**Sebastian:** It is an Open Source license. It's actually a Sybase public Open Source license. And it has quite some similarities with popular copyleft licenses. That means once you change the source code, once you do anything with a source code, you need to republish it under the same license. But there are also some downsides to this license, which pretty much make it incompatible with other projects, and which also lead to the fact that popular distribution, such as DBN or Fedora, don't allow packages and projects that are compiled or that are using tools with this particular license.

**Karsten:** That's because Jack put this line in there that enables him to demonstrate to his son that he's made this, that you can always reengineer, how has this been compiled? I get it. So, it's still around. But you're not involved anymore, Jack.

**Jack:** That's correct, my son has shown me that the interest in Doom continues to this day. He told me once, so there's something I don't know, that there are maybe 60 or so variations of the game now that were been published about a year ago, he purchased the entire Doom suite of games, and he still likes to play it.

**Jack:** It's quite amazing that this particular game, Doom, has survived in popularity all these years, despite the really poor resolution graphics that it exhibits.

**Karsten:** I think it more or less founded the genre of the first person shooter. I don't know if there was anything like it before.

**Jack:** No, I think you're right.

**Karsten:** There may have been feeble attempts at something like it, but I think it was the first that totally consequently went in this "you look over the barrel of your gun" prospective. Jack, you're out of compilers. You're now a database man, is that right?

**Jack:** That's true.

**Karsten:** Okay, and what does it mean when your address book entry says that you help people with the tough problems? What's an example for a tough problem?

**Jack:** Oh, wow. One of my favorites goes back a few years, but a customer had an application. And they actually had a reproducible, which is always great. It's always great to have something that will reproduce the problem. So, the only issue with this reproducible was that it had to run up to three weeks before the bug might have shown. And believe it or not, trying to get an application to keep running on your computer, when IT wants to reboot your computer and you come back in the morning after two weeks of this application running, you find what happened here, so somebody rebooted my computer. After two or three weeks, this bug would show up. Right, and say, oh, okay, well, that's interesting. I can confirm that this bug is there now to figure out what causes it. And it was very difficult, you had to be standing under the right tree and have lightning strike at the right exact time in order to figure this problem out. There are bugs like that that are just, you know, baffling, totally, really difficult to decipher.

**Karsten:** And you did solve the three week bug?

**Jack:** Oh, yes, yes, I did.

**Karsten:** Did you get an exception from it that they wouldn't touch your machine for at least four weeks or something?

**Jack:** No, I don't know how, I might have disconnected my network cable or something to make sure that I was stand alone. I don't remember how I averted that problem.

**Karsten:** Sounds like a solution to avoid remote access.

**Jack:** Yeah, yeah.

**Karsten:** As we've been talking for a while now, I guess we could probably continue like this for at least another hour. And that's why I have a final question. Having been around the industry for 40 plus years, what are your main things you want to tell people that enter the industry right now?

**Jack:** For me, the computing industry has always been fun, first of all, and a very interesting occupation. As long as it's fun, you work at it. For someone who is just, say, starting their own company, it requires a lot of hours. I remember in the early days of Watcom, an 80 hour work week was typical. We would work from sunup to sundown and maybe head off to the bar at midnight to close it out and have a drink and talk about the events of the day and then back at it the next morning. Hard work.

**Jack:** Great, great people to work with. Working with a really fantastic team of individuals is, I think, another really important aspect.

**Karsten:** I think we can all also see that at SAP. If I'd have to summarize what you said, it's all about people and as long as you're having fun, it doesn't matter if you don't have the time to sleep. Right.

**Jack:** Exactly.

**Karsten:** All right, then. Thanks for allowing me to summarize it that way. And thank you for being here today, Jack. And also, Sebastian, thank you for being here today. It was great to have you two here. Thank you to all of you out there for listening to the Open Source Way. I hope you enjoyed this episode. And if you did, please share it and don't miss the next one. We usually publish on every last Wednesday of the month. And you'll find us on openSAP and in our regular podcast distributions like Apple podcast, Spotify, and the like. Thank you again for listening. And Sebastian, Jack, that's all. Say goodbye together. Bye bye.

**Jack:** Ok, bye.

**Sebastian:** Thanks, bye.