Don Kiely interviewed Bill Storage Spring 2001 by telephone. They spoke about Bill's role as a VB activist and hacker, among other topics. During the interview, Bill provided an account of an early beta review group where those who participated in the discussion told Microsoft they valued powerful features more than backwards compatibility.

DK: So, why don't we start off with—if you could describe what your involvement has been with Visual Basic over the years from the very earliest basically through today.

BS: I was an early user of VB, version 2 maybe—yeah, version 2. And when version 3 came along, I got involved with beta testing and I guess I was involved in that to a sufficient degree that Microsoft asked me to participate more in focus groups and that kind of thing. They talked to me—they talked to a lot of us—and had me up there for a visit and at one point. I think this was during Eric Rudder's reign. Eric was a particularly important figure to VB, even though he was there only a short while.

But he asked me, "Bill, would you want to work for Microsoft?"

And I said, "Well, I'm not sure I would really. I'm too old for you guys, but one reason I would want to is the chance to influence where this product goes—its feature set, and so forth."

And Eric said, "Well, you don't have to be an employee to do that, you know. We'll treat you like an employee. You can come to meetings and probably have as much or more input toward where the product goes without being a Microsoft employee."

And I said, "Okay, great." That started a long relationship that still exists. I get to go up occasionally and hang around with the group, and have been I guess on the inside track. I've been making minor contributions toward the feature set. For example, for VB7, since about six months before VB6 was released.

DK: Cool.

BS: So that's how I got connected with them. There are various levels of beta groups, and there has been one VB7 beta group… Or actually, it was an alpha focus group—I don't know exactly what they call it—for about two years now consisting of maybe a dozen people, and I've been active in that group.

DK: Now it was interesting talking with some of the folks on Tuesday about the multiple versions that are active at a given time, where the next version is well into design well before the release of the previous version.

BS: Yeah.

DK: Cool. Now, you said that you started using VB2 as a user. What kind of applications were you using at that time—were you developing?

BS: I was an aerospace consultant primarily, and a lot of my work was concerned with probabilistic risk analysis. Probabilistic risk analysis involves extremely heavy calculations, and a typical problem that you run there might … There were, in the early 90s, a couple of PC-based products that did these calculations, and getting the answer, the

process would run for a day or two days, and the calculations were limited to 10,000 or 50,000 cut sets—and it's not even important what a cut set is—but that limitation was unacceptable for my work in aerospace, so I had been kicking around for a couple of years the notion of building my own software thinking "this calculation looks like it's large, but not impossibly large"—and I was asking people, friends of mine, "What tools would you use to build one of these things?"

And I was only thinking of C and Pascal, and I had written some of the core of the calculations in C and realized that the user interface that would be needed would have to be very sophisticated. In those days, "very sophisticated" meant an editable treeview with discrete editable columns. And John Ganter, a friend said, "Take a look at Visual Basic."

I chewed him out real good, and said, "No, this is serious programming I'm talking about."

And a month or two later, he said, "Really, you ought to take a look at Visual Basic."

And I'm going, "Damn it, I'm trying to tell you, this is like, serious calculations I'm talking about." I had never heard of Visual Basic, and I had seen the Basic language in its early days, and went, "Okay, that's cute for a desktop calculator, you know."
So finally a friend sent me a copy of Visual Basic, and I turned it on and went, "wow!" You know? "I can make Windows here. This thing is kind of neat." So I took a couple of months and I coded up probabilistic risk analysis calculations in VB, and damn, despite the crudeness of VB in version 2, I was able to handle it through the use of some add-on libraries.

So anyway, one company had a VB add-on that I think was a revision of a Basic language product that allowed you to use arrays with more than 32,000 or 64,000 elements—whatever the VB2/3 limitation was. And through using that product and a treeview control that they released right when VB3 came out, I was able to make this really sophisticated aerospace tool that did not have the limitations of the existing product and ran faster besides. I invented it for my work, but then a couple of my clients wanted to use it, and then I ended up selling it to them for—I don't remember the amount—but it was a lot of money in those days for a VB program. Three or six or eight thousand dollars a copy. And then I said, "Well, this is kind of cool."

I realized in working on that problem that I liked writing code. Writing code is something I'd done only peripherally in my aerospace work for the previous 15-20 years. But it occurred to me that I could write code full-time and would probably like it. So I started hiring myself out to write scientific and then, as probably happened to all of us, whatever app people were needing at the time. VB3 came out, and it had data-access capabilities, so all of a sudden around San Francisco, there were all kinds of financial, healthcare apps that were basically the beginning of client/server in VB, I guess.

So, as part of those early years of VB, we were trying to use it for large database, data-centric apps, or client/server apps. Again, through a couple of third-party controls and libraries, and we very quickly discovered extremely severe limitations to VB. Basically, these resulted from not having direct access to Windows messages and that kind of thing. We discovered SpyWorks, and told clients that we could get rid of VB's limitations. You know, we could trap the KillFocus message and much more thorough data validation than was permitted by VB's LostFocus event. We could also modify the look of Windows, and we could even set up callbacks so that we could add menu items to

the Windows… - whatever that menu is called at the top-left corner of a window. And that provided me with consulting work for about two years, you know, being the guy who could make VB jump through hoops.

As neat as SpyWorks was, our projects got messy because we had four or five SpyWorks controls on every form in the app, and the code in those controls was fairly foreign to the average VB coder. And that's when I started lobbying hard for a lot of the features that we got in VB4 and VB5—along with little bit of object-orientation. For example the VB dev team gave us the ability to hack into VB through the pointer functions and the AddressOf operator. I made a lot of trips to Microsoft lobbying for those features, and at one point, I sent a large amount of exotic foreign beers up to Microsoft. That was in the VB5 alpha I guess, and the joke that came out of that was that you could buy a feature in VB for the right price.

DK: [Laughs]

BS: Of course, that was only a joke. I had a number of long meetings with Eric Rudder and a bunch of people whom I just have phenomenal respect for. I don't know where Microsoft gets their recruiters, but it seems to me that they have hired some of the most talented people I've ever seen in the software industry. And somehow they manage to be 25 years old. And I don't mean talented just as these guys are phenomenal programmers, but you know, they're good project managers and business managers, and they listened to our cases.

For one, we needed this stuff. They kicked around the obvious tradeoffs. "Okay, we're going to expose VB to hacks. It's very likely that future versions won't allow these same hacks, and that we're going to break people's code when we rev the versions of VB." And a lot of the stuff that people are complaining about with the impending release of VB7. Yeah, these issues were identified three or four or five years ago; and they did the right thing for the time, which was to say, "Yeah, we agree you guys need this stuff. We believe you, you really are writing serious projects in VB." Up to that point, I don't think they realized that people were doing really large VB projects. I sent in a couple examples of 100,000-lined client/server apps written in VB that were pretty decent projects. And the guys up there said, "Wow, okay. People are rising to the occasion. Yeah, we agree, you do need this stuff."

DK: Cool. Wow, thanks for all your work on that.

BS: You're letting me ramble here. So should I continue along on this stream of consciousness?

DK: Yeah, that's fine. This is the kind of information that's good.

BS: Okay. So I kind of got to be known in the community as one of the chief hackers. When VB5 came out, Matt Curland of Microsoft and I partnered up and wrote a lot of articles that *VBPJ* carried in their Black Belt Programming column. And the charter of that column, by the way, was to intentionally lose some people. The *VBPJ* staff was really kind of fed up with Letters to the Editor that said, "This magazine is for sissies,"

you know, “You guys don’t write any really tough stuff.” And of course, their goal was to serve business needs and to make people better programmers, not to pacify the people who needed their egos stroked by difficult code.

Nevertheless, they said, “Well, you know, a lot of people will learn from hacks and advanced coding technique articles, so go ahead and write these articles and don’t hold back. Make them difficult. Lose a lot of people. At least we won’t have the reputation of not having any serious material.” So Matt and I wrote a lot of stuff on subclassing, using the newly released pointer—object pointer, variable pointer, function pointer, operators, and functions—that were in native VB 5. Then we went on to demonstrate runtime v-table modification, which was really necessary if you wanted to implement some VB-unfriendly interfaces. So COM was growing up and there we were trying to show them we could do anything in VB that you could do in C. We took the challenge of implementing the IEnumVariant interface in VB. And we discovered that, wow, there were a lot of obstacles. And while the API extensibility features of VB—the ability to call the API functions—was all set up for handling pointers to C structs or pointers to arrays, things like that, there were no similar provisions in implementing an interface in VB. So we had to do some pretty serious hacks, including swapping functions in and out of the v-table during run time. At which point Microsoft said, “Okay, we gave you guys the tools, and now you’re doing really dangerous stuff. You’re showing people these hacks that we absolutely know will not work in future versions of VB.”

They wrote a Knowledge Base article naming us - naming me personally - and said, “These techniques are not supported. Do not call Microsoft Product Support and say that you’re doing v-table manipulations.” Drew Fletcher called me up and said, “I’m going to invent a version of VB that makes your silly hacks unnecessary.”

I said, “Great. You give me that version of VB and I won’t want to hack. I am not a hacker by nature, and I’m getting a reputation here for being a guy who likes to muck around in low-level VB.” I write client/server apps for a living, and I needed those hacks in my client/server work. An example of that, by the way, is that many tables and databases that have non-numeric primary keys. When you dump a list table into a list control in VB, you want to put the primary keys in the ItemData area. Well, you can’t do that if it’s a non-numeric primary key, so the easy thing to do there is to take a pointer to that string and put the pointer in the ItemData, and then you can get the string out by dereferencing that pointer. Really cool.

So that was one of my examples that I used with Drew, and he said, “Okay, I know, we’re going to fix this so you don’t need any of those hacks.” Oh, you know, six months later, they call and invite me up to Microsoft and say, “Take a look at VB7. Look at where we’re going here. You’re not going to like it, Bill, because we’ve broken all your hacks.”

And I went up there and they took me through the show and I said, “That is absolutely fantastic. I don’t need many of those hacks anymore.” Maybe they thought that my response was going to be similar to the other hackers, but in reality, there are a lot of people who like hacking VB because it is fun, and it’s gratifying to be able to get at the internals of VB and muck around with it. Needless to say, there were some people when the VB7 betas rolled out that were unhappy with where it was going. “By god, you took pointers away from us! You took deterministic finalization away from us.” I don’t think those folks are people who do client/server programming for a living though.

DK: [Laughs] Right.

BS: A few years before, I advised Microsoft to think hard about whom they were trying to satisfy. Basically, that companies, big VB users—and I named Fidelity, John Hancock, Prudential, Wells Fargo, Schwab—these guys are big VB users, and I suggested you maybe ought to get those kind of people in your focus groups rather than the most expert VB people. These experts have a very valuable perspective, but I don't think you want to tailor the product to their needs.

DK: [Laughs]

BS: I guess they agreed. They went out and got a bunch of people from those kinds of industries that I mentioned, and I think VB7 had a lot more influence—a lot more of their influence—than the people who had been influential in previous versions. Needless to say, those who were more influential in previous versions are complaining loudly about where the VB feature set has gone. And as you can probably tell from my discussion to this point, I disagree strongly with them and think they're doing a tremendous disservice to the VB community by getting their followers up in arms. You know, some of their followers regard these guys as community leaders, so they're vulnerable to what I think is a narrow, unrealistic viewpoint. Those guys also feel like I have betrayed them. You know, I used to be the lead hacker and look where I am now [laughs].

DK: [Laughs] Right.

BS: They're expressing similar regrets really about, for example, that vtable modification stuff. I'm extremely happy with the VB7 feature set. I think this language is incredible. When I reviewed my code for deterministic finalization - my VB5/6 code - I find that in every single place I was using deterministic finalization, the need goes away with VB7. I know there are people who use the Terminate event to get rid of GDI objects, and I also know that the number of people in that boat are very few. And I can't see defending the overhead associated with reference counting of all objects just for destruction of GDI objects needs. And, by the way, if you do your coding using .NET properly—that is, if you use the intrinsic features, you're a whole lot less likely to need the Windows API, for example with GDI, than you used to be.

DK: You said you got involved originally with VB2. So you had not used VB1?

BS: I saw VB1. No, I never used it in production. It was VB1 that the guy sent me that got me interested. But at that point, VB2 was available, so I ran out and bought VB2.

DK: Okay. All right. Did you ever get involved with VB DOS?

BS: No.

DK: Can you tell me a little bit about a slightly later period, when VB4 was underway?

BS: VB has almost been brought down a couple of times, particularly VB4, by Microsoft Office—I think the Office team was trying to turn VB into the Office macro language. I think that VB4 was headed for a total disaster and Eric Rudder came along and was an incredibly obstinate, tenacious young guy who said, "Bullshit, I won't have it" and just tore the department apart.

And VB4 did have some defects, but I did know that it actually incorporated some of the features that were slated for VB5. Which is part of why the VB4 beta was a very long beta cycle. And I know that Eric was instrumental in getting the product back on the right track as a programming language rather than as an Office macro tool.

Eric and company – as I recall, Drew Fletcher, Steven Lees, Craig Symonds, and Geoff Shilling, Rob Copeland, Chris Dias and others I can't recall – worked the beta groups pretty hard, took a lot of grief, and worked hard to get something real back in the product. For example: During the VB4 or 5 beta—I can't remember which now—it was the one where they introduced classes, I guess that was VB4. This was late in the beta, I said, "Look guys, a lot of people are going want to write a class called Things, you know, Authors, Customers, whatever, a class that has a plural in its name. They're going to need to implement default properties—, an Item property for example—so it'll behave just like a VB intrinsic collection, and they're going to need a NewEnum, which boils down to a method with a dispatch ID of –4, that allows you to say For Each x in Things." And I showed Craig Symonds some business cases for this in client/server apps and pleaded with him, "You absolutely cannot release this product without that in there. You're going to get murdered. People are going to expect this behavior and their need is legitimate."

This was after the final beta was released and Craig said, "Okay, I will get that feature in this product one way or another. You may have to do some real weird-looking coding to get it in there. You may have to edit the FRM file manually, but I guarantee you that feature will be in there because I believe the need is legitimate." And two weeks later, they released a minor last-minute rev and by god, the feature was in there.

DK: Cool.

BS: I know that sticking a feature in that late in the project is risky… because I write code, and you think about the feature, and you go, "Yep, that'll take me one hour to code that." Meaning, it'll take two weeks to do all the administrative garbage to get it in. So, you know Craig's a man of his word and a real practical kind of guy in touch with real users' needs.

DK: Okay. Well, let's talk a little bit more about COM VB4. Microsoft said uniformly that the reason why the beta was so long was because of performance. But, you suggest there may have been other reasons too?

BS: I think it was really just getting some of the testers calmed down.

I'm trying to remember what else in VB4 was - what were the key issues. Well, of course, that was the release where we got evil type coercion, about which there was a huge amount of protesting—"this is going to ruin everything". And in reality, it didn't cause a lot of trouble, but it caused some. And I think everybody in hindsight realized

that that was the wrong thing to do. I think—again, I'm reading between the lines—that it was a consequence of the influence of Office, where they were trying to dumb down the language.

DK: And boy, we're set in VB7.

BS: Boy, have we ever reversed course.

DK: Thank heavens. I fairly uniformly thanked everyone on Tuesday for breaking all the code. I think it's one of the best things to happen to VB. That's a simplistic view of it, but …

BS: I have an interesting story about the first focus group conducted for VB7, perhaps two years ago. We were in a room, a group of us from the industry. I want to make it clear that it was a focus group, and that Microsoft was looking for our feedback—we weren't there to help develop the product.

The goal at that time was to make VB7 backward-compatible; that is, even the forms were to be backward-compatible. Yuval Neeman came into the group and said, "Guys, this is looking a lot harder than we thought, so we're thinking of having two compatibility options: non-compatible and kind-of compatible." Then he stated all the issues associated with this and asked us what we thought.

After some discussion, I got up on my soapbox and suggested, "Drop backward compatibility altogether. Do us a favor; haven't you guys learned the lesson of DOS? Backward compatibility cost us so much money over the years. Break my code. Force me into getting rid of my old code if I want to add VB7 features into my product. But this means that you've got to make VB6 usable for the next five years alongside VB7 because, if I want to fix a bug, I've got to be able to open the old version and fix it. I cannot do that right now. VB6, VB5, and VB4 are incompatible."

Yuval asked, "What do you guys think of this? Agree with Bill? Disagree?"

I was expecting a huge fight from the other people in the focus group. However, all but one person said they agreed. We had a discussion for about half an hour, then the last guy said that he more or less agreed.

Then we broke for lunch, and afterward Yuval came in and said, "Do you guys realize what you're telling us? I mean, we're flabbergasted. We can't believe what we're hearing."

We kicked it around for another hour, then we said, "Yeah. This really is the right thing to do, from the perspective of those of us who have very large code bases."

We left that first focus group, and I said, "Yeah. Yeah. We sold the dev team on that, but when the marketing team gets hold of it, it's never going to fly."

Six months later, or something like that, they invited us back and said, "Take a look," and it was exactly what we had agreed upon six months earlier. All of us looked at each other with disbelief: Microsoft did the right thing rather than the popular thing.

I still feel that way. I think most of us in this community do. Understand, Microsoft went on and did quite a lot of market research testing and so on, and the result of that was consistent with what we were saying in that group. Who knows, maybe if I had not gotten on my soapbox, it might have gone a different way [laughs].

But please don't take this to mean that Bill Storage, lobbying on his soapbox, caused a lack of backward compatibility. The truth is that this was thought out by a huge number of people at Microsoft for a very long period of time.

It really was amazing to me that all the people in that group, all agreed that's the right thing to do.

DK: That is really amazing. I assume there was a pretty diverse set of people?

BS: Yeah, they were. But they were heavily biased toward large-industry players. But, you would expect those guysto be the primary objectors. They're the ones with the big code bases.

DK: Yeah, but with a dozen people like that. Great, interesting, thanks for doing that. Keeping with COM and back to VB4, what was the story with 16 bit?

BS: I don't know what the story was, but I know that it caught all of us … I was doing some writing at the time. Zane Thomas, Constance Peterson, Karl Peterson, and a few others and I were writing a couple of the Waite Group how-to books at the time, and 32-bit VB caught us with our pants down because we all took assignments and what chapters to write for the book, and we all went out and wrote 16-bit code, and Waite Group—can't think of the guy's name, but Mr. Waite.

DK: Mitch?

BS: Mitch, yeah. He got a hold of us and said, "What the hell are you guys doing? 32 bit's a big deal. Go rewrite your stuff in 32-bit." And when we did—of course this product was in beta—all of a sudden, we realized that everything was different in 32-bit. Cross-process subclassing doesn't work anymore, and allocating a chunk of memory and hitting it from multiple EXEs doesn't work. So I learned a lot in trying to rewrite my chapters for that book about 32-bit, but it was clear that none of us were anticipating that there was going to be a big demand for 32-bit apps.

DK: I understand that from Microsoft that it was originally a 16-bit product and that 32-bit was added later.

BS: I wasn't aware of that. It seems to me that I was thinking the first beta had both versions in it, but now that you mention it, maybe the first beta—or it could have been an alpha they sent us—maybe it was 16-bit only.

DK: And, as Microsoft explained it on Tuesday—I forgot who was talking with me about this—but that originally, the Windows 95 timeframe wasn't going to work for 32-bit, but then as the beta got a little bit longer, still in the early feature stage but not …maybe that's the difference: In the feature stage, it was 16-bit, but then before the first beta it might have been 32. I'll clarify that. As VB4 got delayed, Windows 95 … it was closer to the time that was being released, so they went with the 32-bit. But that could have been even before the alpha.

All right. So, you weren't directly involved with discussions whether that should include 16 or 32?

BS: No, no.

DK: All right. What do you see, briefly, as the major impacts of each of the versions? Another kind of wide-open question.

BS: VB3 brought data of course, and that was the biggest, probably the most influential change. And then VB4—believe it or not, it's interesting to think about this based on where we are now—VB4 introduced the concept of classes and public and private methods. Now we got our first hint at polymorphism and better encapsulation. And I believe VB5 fleshed that out pretty well, and VB5 brought the native code compiler. Correct, yes?

DK:Yep.

BS: So, VB5 is when we started doing all of our cool hacks. So the ability to write a true collection class must havecome in VB5. VB6 introduced—for me, the big deal about VB6 was the morphing of the UserConnection into a DataEnvironment. People have very mixed feelings about DataEnvironments. I believe that Bill Vaughn was very unhappy with them. I don't use the DataEnvironment for coding or binding; I use it as a device to encapsulate stored procedure calls. And it's wonderful. I did a VB5 and VB6 project where the two projects were almost identical, and the VB6 version eliminated about 35 percent of the code, based on capabilities of the data environment. So for me, I think that thing was the unsung hero of VB6. It had some flaws, but it was far better than the user connection or code. And VB6 also introduced WebClasses, which was a pretty cool technology, but it was too difficult for people to use.

Those two things came out, then—by the way, VB5 introduced the Document object—Active Document. It's kind of interesting to look at some of the stuff that failed. And you really can't say that they failed because they were lame or they were bad ideas. They just weren't things that people wanted. Probably the thing that killed Active Document was it boils down to a container and that container performance isn't very good.

What killed the WebClass I think was merely the fact that it was difficult to program. It was quite a departure from the paradigm of VB coding. I used it. It was very bug-free, but there were some limitations in it, and of course, the WebForm is a hugely improved version of it. You might equate the WebClass/WebForm progression to the data access capabilities of VB2-which you would call rudimentary- and VB3: drop a data control on a form, set some properties. Cool!

DK: Yeah, I was really sorry about WebClasses. But some of the best, better features are getting into WebForms I guess.

What do you see as the roles for the various extensibility features in the various version, starting with VBXs and OCXs, and the extensibility object model for add-ins for VB's popularity?

BS: That has to be the thing that made VB successful. If you remember at the time that VB rolled out, there was a CA Realizer and a couple of other products that were in most ways, superior to VB - in terms of syntax and completeness. But they didn't have VCXs. VCXs allowed a bunch of little guys to make some cool tools and make money on them. Zane Thomas, of course, would be the absolute best guy in the industry to comment on this, because he's a good C coder guy who writes controls. He's also a VB coder who uses controls. He has kind of a novel perspective. Gustavo Eydelsteyn would probably be another person with a similar perspective at ComponentOne.

DK: Let's see. How about the pain and benefits of moving from VBXs to OCXs? Do you have any comments about that?

BS: Horribly overrated. The way it was didn't bother me at all.

DK: Yeah, the benefits were so good.

BS: Yeah. And there were a a few hacks in VB3 that were lost in VB4, such as the ability to destroy and re-create a control to set its pre-HWnd properties. The VB3 DLL used to have functions that allowed you to destroy controls and re-create controls, and we lost that with OCXs. So there was a brief time there where we were really stuck if you really wanted to set one of those pre-Hwnd properties. Then along came VB5 and you could—you could do this in VB4 if you knew what you were doing—but in VB5, it was pretty easy to use the CreateWindow function and create your own control windows from scratch. But that's pretty esoteric stuff—not many people were doing that.

DK: How about the COM Control Creation Edition?

BS: The Control Creation Edition is what they called the version of VB5 that Bill Gates demanded. Apparently, one day, he went down to the dev team and said, "Let me see VB5," and they showed it to him, and he said, "Cool. Give that away to attract people." I don't know, it was something like that, and it caused an incredible amount of concern to the dev team, a lot of which was very valid concern because there were problems. Basically, you put the VB community in Beta hell by having various beta versions of everything on people's machines. But as I recall, that was an entire complete set of VB5 minus the ability to compile an EXE.

DK: Yeah, it was pretty amazing. And of course, that was around the time that they were trying to get into the Web, so they wanted people to be able to create these components.

BS: It was a boon for me because it allowed me to write a bunch of articles that were really about VB5 beta, and it essentially excused me from my nondisclosure agreement.

DK: That's right!

BS: I wrote a couple of articles.

DK: I wrote a couple of those too.

BS: If you look at the first three or four Black Belt articles that Matt and I wrote, each one of them at least one place in the article contains the phrase “Control Creation Edition.”

DK: Yeah, I forgot about that. All right. Okay. Well, that’s all the specific questions I have. Is there anything else that would be interesting to know about development in VB over the years?

BS: Well, I think you let me do my long rants at the beginning.

DK: Thanks a lot again. I really appreciate it.

BS: It was great talking with you.

DK: Take care. Bye.