The Past, Present, and Future of the Macintosh Desktop
Dan Smith Interview, Semaphore Signal March, 1986

To a first-time user, perhaps the most striking thing about the Macintosh is its use of the desktop metaphor: the folders and other icons intended to help make the Macintosh a user-friendly machine.

For a perspective on where those ideas came from, how they were further developed by Apple, and what they might lead to in the future,we interviewed Dan Smith, an Apple Principal Software Engineer.

Signal: Give us a brief history of your career at Apple.

Smith: I've been at Apple for a little over five years now. I initially signed on to the Lisa project to work on what we called the Desktop Manager, essentially the equivalent of the Finder on the Macintosh. I worked on that for about two years,until the whole Lisa project was near completion. Then I became User Interface Coordinator for the Lisa project, then switched to a consulting role for theMacintosh, since Mac picked up about halfway into the Lisa development stage. I took some time off from the Macintosh and Lisa to start working on some future projects, did that for about nine months, then got pressed back into service to do a program development environment for the Macintosh, which is what I'm working on right now.

Signal: Were you the desktop programmer? What was the organization responsible for the desktop and the other Lisa software?

Smith: The effort was split up into a couple of different groups. There was the desktop group. Two of us actually did the implementation. I programmed the user interface portion, and Frank Ludolph did a fair amount of the lower level implementation. Then there was the applications group, and that was split up into essentially the different applications that came out: LisaDraw, LisaWrite, and so on. There was also an operating system group, which did the much lower level software.

Signal: How did the ideas for the desktop originate, and how were they

incorporated into your design?

Smith: That's a pretty interesting story. When I started at Apple, the idea of the desktop hadn't really quite been born. In fact, it was thought we'd do something fairly simple, and it would be a one-person job for a couple of months and it would be over. It was a little later in the project that we realized the desktop was going to be a central part of the entire system. The idea of an iconic form didn't come along until quite late into the development of the product. We started off with something that was pretty Smalltalk-like. There was a notion of a thing called a browser, which is essentially a table you could flip through, listing the documents you had in a hierarchical fashion. But the whole initial desktop was essentially technically oriented. We went through iteration after iteration. I remember doing prototype after prototype, and trying them on several groups of people, getting it to be more and more useable. But a number of us were not happy with what we were getting, so fairly far into the project a couple of us took a radical departure. We took a fair amount of our own time developing an iconic model, then sprung it on the whole group. It met with some resistance, but the majority of people really liked it. Then it was a mad rush to incorporate it into the final product.

Signal: Exactly who did what? Was the design fluctuating with the whims of a few individuals?

Smith: The desktop had a pretty fiery history. We did have design reviews of all of the components of the system. We had teams: writers and marketing and engineering people who were all involved actively in the product. When we were actually doing the design and programming the application, the specification circulated amongst the teams. They all had a voice in the initial design of the product. As we circulated some of the actual design, and showed some of the prototypes, there was some frustration by some people that the desktop was not as easily useable as we would like. A couple of people, myself and Bill Atkinson primarily, and later Bruce Daniels, snuck off and worked on a prototype iconic model.

Signal: Where did your ideas for the iconic desktop come from?

Smith: It stemmed from a number of different places. From some of the work done

at Xerox on the Alto computer. There were some IBM research papers, believe it or not, that discussed some office models that were iconic based that we looked at. And somewhat after we had worked on this, the Xerox Star computer had come out, so we compared it to our model. There was also some work done at MIT that had some influence on us. We got together and kicked around a lot of ideas, and tried to keep it consistent with our user interface philosophy. After a short period of time, we put together a prototype that stayed pretty close to the final product.

Signal: Was the big breakthrough essentially just realizing how to use icons inside of windows to represent files? Were you already using windows and pull down menus?

Smith: The initial design already had windows and pull down menus, although even those had some history. But the actual desktop's primary function was a filing function, an organizational function. The iconic model was really the late development.

Signal: So various ideas would come, they'd be implemented and tried, and then approved by peer groups, as opposed to steadily working toward some predefined specification?

Smith: That's not quite true, although the desktop was the exception to the rule. All of the major software products were designed by a group responsible for that component, and were pretty much spec'd out completely before any implementation was done, although there would be little mock-ups to check out some of the ideas. We tried to do the same thing with the desktop, but it's one of those products everybody is forced to use, and everybody has an idea on how it should be, so there was a lot of feedback about that particular component. It took a fair amount of iterating to come up with something that appeared to be not only satisfying to the groups working there, but also something that went over fairly well in our user testing, which we did quite a bit of. We tested mainly on people who had very little computer experience. We tried initially to make it mainly those who were in Apple, who had very limited experience, maybe new hires or people whose job didn't involve working directly with a computer. Further on into the project, we actually hired some people for testing purposes and we had a set of people in-house whose primary job was to gather testing feedback. We even set up

a couple of testing rooms, and carefully recorded everything that happened, and tried to draw whatever conclusions we could. It took a fair amount of that to get us headed in the right direction.

Signal: What are some examples of things considered for the desktop, but that ended up being changed or left out?

Smith: The initial version had icons which were three dimensional and much more cartoon-like and somewhat entertaining. In fact, the initial trashcan was this beat up old trashcan you'd expect to see in an alley, with the lid half open and flies buzzing around it. We had actually talked about putting in some sound effects for when somebody put something into the trashcan. That met with some resistance from some of the stodgier people on the team, so that was dropped. One thing in particular we had a heated debate about was the notion of whether a document had to be saved or not. A number of people on the project wanted the system to be as far removed from typical computer interactions, and be as concrete as possible. The argument was that when you write something down on paper, it's always there. You don't have to say "save" to prevent it from mysteriously disappearing, as opposed to a typical computer model where people think "I know I have something in the computer's memory, and it's temporary, and I have to make sure I get that information from memory onto the disk otherwise I'm going to lose it". So there was quite a heated debate on that one, and unfortunately we tended to battle more toward the computer model, for a couple of different reasons. One reason was familiarity, the idea that users who had computer experience were more used to saves. There were some technical considerations too, having to do with how much information we had to keep around all the time when somebody stopped working on a particular document. Another desktop feature was the problem you always have anytime you go to a filing system with folders buried in folders. This was something Frank Ludolph was very concerned about. How do I find my documents, how do I find my folder? With folders nested in folders, and having to double click and open them and search around, it became quite tedious to find something if you had misplaced it. Rightly so, he thought we were duplicating some of the frustrations a person would find in a normal office, having to manually dig through their filing cabinet to find something. A computer should be able to take care of that job for them, essentially doing an electronic search of the filing cabinet. That was a feature we wanted to implement on the Lisa, but didn't quite

have time to do.

Signal: How would automatic searching have been accomplished?

Smith: That feature wasn't fully specified, but it would be primarily by specifying certain attributes about the item you were looking for, whether it was simply the title of the document you were looking for, or some key words in the document, almost like a database query. Something to make a search a lot less painful.

Signal: What other desktop capabilities never made it into your final model?

Smith: One was the ability to allow new applications to be added, which could operate on documents not of their type. Suppose somebody wanted to add a new compare program that compared two text files for example, and gave you the differences. With the user interface model we had, simply clicking on a document opened that document, or clicking on an application opened that tool. There was no real clear way of specifying documents of different types as essentially parameters to another program. Probably the biggest single feature missing in both the Lisa and Macintosh is the ability to automatically remember a series of actions and play those over again. One of the things the desktop did was to make most operations very simple to do, which is a great benefit to first-time computer users. But computer users expect the ability to give the computer a series of commands and have the computer record those, and save them from the drudgery of having to repeat those over and over again. Primarily because of not having quite the right user interface for it, and the technical problems involved, that particular feature was never fully implemented in the Lisa or the Macintosh.

Signal: Were there any schemes that would allow repeatable commands?

Smith: There were a couple. One that comes to mind that was actually done with limited success was to simply record all the actions of the user on a real low level: the user moved the mouse from here to here, clicked the button, typed keys at this point. That allowed us to do little demos where the machine appeared to be running under automatic control. The problem with that approach is it requires you be in the exact same state every time you run this little script, even down to the exact position that icons are on the screen or within their folders. That situation turns out

to be very rare, so that approach is flawed. It's also not something you can easily parameterize. You pretty much specify an absolute action.

Signal: Programmers were disappointed to find that this beautiful, iconic, straightforward, user-friendly, state-of-the-art desktop was built with a fairly old programming language and tools. The Macintosh didn't come with any new, powerful systems to make programming a Mac as easy as using one. Is there a lack of tools for developers to easily implement new Macintosh programs?

Smith: There's no question that implementing a Macintosh type of user interface is difficult, and the initial implementation on the Lisa and the Macintosh was done in pretty traditional form. What we've tried to move towards is more of an object-oriented implementation that involves extensions to existing languages. In the long run, that's really going to be the way to develop software for this type of a system. The problem you typically have with object-oriented software is performance and size. Since it's a general system, you tend to get a lot more software than you actually want because everything is built in, and you may be only using a subset. The size of any application tends to be quite a bit bigger, and the performance often can be quite a bit sluggish.

Signal: Has object-oriented programming really been proven? Both the Lisa and Macintosh have demonstrated excellent user software, but only using traditional, non object-oriented programming. Apple's object-oriented systems always seem to arrive a day late and a dollar short and have never really been used to implement anything of commercial significance.

Smith: That's certainly been true up to this point. Whether you can use that approach or not depends on how much horsepower you have behind you. Smalltalk is heavily object oriented. Depending on what kind of machine you're running on, Smalltalk is a wonderful system to use, or it's just horribly slow and painful. So it's a combination of the hardware you have, and the software implementation. Programmers who use an object-oriented system are really pleased with it, because of the amount of work that's already done for them. Developers will gravitate towards that type of a programming solution once it's implemented in the most efficient possible way, and there's sufficient hardware behind it to make the performance acceptable.

Signal: Why were the tools that Apple used to implement its first desktop applications never released to developers?

Smith: Great question. The reason is we didn't have the object-oriented extensions made to any of our languages at the time we were developing the product.

Signal: If the Apple programmers didn't need finished object-oriented extensions to complete their applications, why did you think third party developers would have to wait for the extensions to be completed before they could get tools for their products? Smith: The primary reason was because of the difficulty we all faced while developing the applications. We were all working at Apple on a daily basis, and had access to everyone on the project on a face to face basis, and we were still having difficulties in the amount of time it took to implement just a standard user interface. Each of the applications had to implement scrolling, and the menus, and all of the appropriate things, according to the standard user interface and specification. It turned out to be just incredibly difficult to do, and incredibly time consuming.

Signal: Couldn't Apple programmers borrow each others code and algorithms?

Smith: Well, we tried to some degree, but there would be these slight little differences. It was very difficult at times to trade some of that stuff, because of the way an application was structured. We were essentially pioneering all of those techniques, even the basic design of an application being event driven, as opposed to reading in a little command line and a carriage return and spitting out the answer. It was all quite a bit different from the programming experience of most people there. We just realized way too much effort went into developing desktop applications, and something better had to be done for outside developers.

Signal: How did you get involved with the Macintosh group?

Smith: Towards the end of the Lisa project, Macintosh development was getting much more established. I spent a little bit of time with Bruce Horn and Steve Capps, the people who worked on the Finder, because they were doing the same thing I'd done on the Lisa. We were able to share some ideas. That was useful for

all of us. I had already conquered a number of the problems they were facing and, in their development, they had found a couple of interesting solutions to problems I had not found a solution for.

Signal: What's an example of something they found a solution for?

Smith: They had a user interface solution for the problem of how to get a tool to operate on documents not of its kind. The solution was to simply select all the documents and the application involved, and click on one of those as a set.

Signal: Wasn't the Macintosh group reinventing the wheel by not letting you write the Finder?

Smith: It was to some degree, but the Macintosh group was faced with constraints which were pretty extreme compared to those on the Lisa. The amount of memory available and not having any real hardware memory management made some things quite a bit more difficult on the Macintosh.

Signal: Was the Macintosh a step backwards from the Lisa? Was the machine sorely
underpowered for what it was trying to do?

Smith: Not with respect to the original design goal. The original goal had envisioned the Macintosh as being a personal computer that would run a single application at a time. It would have superior graphics and the Lisa-style user interface. But it was going to be restricted in terms of exactly how much it could do. It turns out that in terms of computational power, in terms of raw speed, the Macintosh actually exceeded the Lisa to some degree.

Signal: Do you feel the Macintosh group did a good job of capitalizing on what the Lisa group had pioneered? Or were not all the lessons learned?

Smith: Yes and yes. They did a tremendous job of leveraging off what Lisa did. All the graphics routines carried over, almost to the byte, thanks to Bill Atkinson's foresight. The window and menu management routines were essentially brought over as is, at least as far as the interface. The major changes that had to be done for

the Macintosh were that things had to be recoded in asssembly language in order to get the program sizes down. But most of the interfaces and stuff like that were pretty much the same, but with improvements along the way. In that sense, the Macintosh was really a second generation Lisa, so they had an opportunity to improve on a lot of things. In some cases, a step was made backwards.

Signal: What's an example of Macintosh taking a step backward?

Smith: A step backward was the "run one program at a time" model. For example, you clicked on MacPaint, the entire screen would be cleared, and up would come something that was totally different than what you had been looking at before. The desktop model disappeared on you, and now you were running this entirely new program. If you wanted to do any type of filing operation, you either had to quit that program, go back to the Finder and manipulate your files, or there was this little dialogue box that allowed you to search through a list of names as opposed to seeing them in their iconic format. On Lisa, you stayed in the desktop the entire time and would just click on another icon to open it. That would open up a window, but you would stay in the desktop. You wouldn't get this dramatic state change, and you also wouldn't have to go from at one point dealing with icons to, at another point, dealing with a list of names.

Signal: How much of that step backwards was forced by the hardware limitations of the Macintosh? Could the programmers have pulled off a multi-application environment like the Lisa?

Smith: That's a tough one. With the 128K of memory available on the original Macintosh, they really couldn't have pulled it off. But with the 512K Macintosh, it was definitely more possible. Now with the Mac Plus, it could be done quite easily.

Signal: There are a lot of little differences between the Macintosh Finder and its predecessor, the Lisa desktop. The Lisa desktop underwent cycles of consumer testing and design reviews. Is it true the Finder programmers were influenced by a lot of Lisa features, but ended up doing many things their own way just because they wanted to?

Smith: That's fairly accurate. The Macintosh group had much more liberty as far

as the user interface design. They had the opportunity to make little tweaks to try to improve on the overall design. It wasn't that the Finder was totally unreviewed by anyone other than the programmers, but there was definitely more liberty. That's not necessarily a bad thing. The programming team on the Macintosh was quite a bit smaller. With a smaller group, they had an opportunity to have the product be a little bit more consistent, because you get more of a single-mindedness to the product. It didn't look like it was hatched up by twenty-five different people.

Signal: It's interesting how the Macintosh was able to successfully provide a standard set of programming routines for developers, even though they aren't object-oriented, compared to the Lisa group's inability to do that. Why?

Smith: It has to do with the initial orientation of the project. When I first started on the Lisa project, the goal was to produce a machine essentially not programmable by outside developers. When we later decided it would be appropriate to have outside software developed, we were in a little bit of trouble, because we had created quite a complex system. It was very difficult to program. Whereas the Macintosh software architecture tended to be open at the very outset, and all the routines were carefully documented and carefully designed with respect to outside people using them.

Signal: What was your influence on MacPaint?

Smith: While I was working for awhile on future products, I was working with Bill Atkinson and others out at Bill's home. Bill was still working on MacPaint at the time. We would get together in his workroom and kick around various ideas about the MacPaint user interface. He would cook up a new version and hand it to us, and we would play with it and give him feedback. We got to be firsthand users with that product. That was a real treat.

Signal: MacPaint, for all its greatness, violates a lot of Apple's user interface guidelines. Was there ever any concern about that?

Smith: Yes, there was, as a matter of fact. There were a number of people who were opposed to some of the things Bill did. Bill's feeling was that we needed to pioneer some new ideas, and there were just certain things in some of the user

interface guidelines which weren't appropriate for his application. He decided to push for some of those things, and hope for the best. It turns out he was able to move more towards the user interface guidelines down the line.

Signal: Why did it take so long for LisaDraw and LisaProject to migrate to the Macintosh and become MacDraw and MacProject?

Smith: The primary reason was the memory constraints on the Macintosh. Most of the programs on the Lisa were huge by comparison to other software products. Significant recoding had to be done in order to get the products moved over to the Macintosh. It takes a long time to program in assembly language, or to reorganize your entire program with space a primary consideration.

Signal: Didn't anyone realize Apple was trying to get the Macintosh to do Lisa-like things? Couldn't you have just put in more memory and saved many programmer-years of effort?

Smith: A lot of people said, myself included, before the Macintosh even came out, that the thing was designed with memory that was way too low. The primary motivation there, of course, was to keep the price down. They wanted the Macintosh to be affordable to the common person. This was going to be a computer most people could go out and buy. As time went on in the project, more and more features were added. They decided to go with a different disk drive, with different memory, and slightly larger display, so the price ended up creeping up out of the reach of the common person, to the $2500 figure. At that point the memory limitation perhaps wasn't as good of an argument as it was earlier, because at that point the memory wasn't the dominating cost of the product. So I'd have to agree with you on that one. If the Macintosh had initially come out with a larger amount of memory, it would have saved significant amounts of time in development.

Signal: Is there any hope for a true, Lisa-like, multi-application desktop environment on the Macintosh?

Smith: There certainly is.

Signal: You say that like you're hinting.

Smith: It's sort of like a hint. I know of some outside developments in that area, outside of Apple. There are a couple of efforts going on there. I wouldn't be surprised to see something like that fairly shortly.

Signal: Switcher and the desk accessories have always seemed to be somewhat desperate attempts to give the Macintosh a Lisa-like environment. Couldn't Andy Hertzfeld have made Switcher juggle multiple applications on the screen at once? It's already executing any application on demand, but idle applications are kept off-screen. What if the Switcher's multiple screens were shrunk down to icons and saved on a visible background, or occupied windows that could be scaled and overlap? The Macintosh could still jump between applications, and yet the user could see all of them on the screen simultaneously.

Smith: I'm not sure if that particular idea was around at the time, but Andy's initial goal was to just do something simple, and that had the highest chance of success. Switcher seemed a fairly simple idea, although it turned out, because of the Macintosh architecture and the system design, to be quite difficult to do. It really took someone with Andy's intimate knowledge of the low level aspects of the system to pull that off. However, now that Switcher has been done, and it's been shown it's possible, we will see more Lisa-like models, where the visual continuity is retained.

Signal: Where do you see the user interface going? What will future products look like?

Smith: What we'll see in the long run, in terms of software products anyway, is a much tighter integration. You'll be able to buy small tools that fit into the rest of the system and work in combination with all other tools. Say, for example, you're trying to compose a memo to someone. In front of you is your piece of paper in electronic form, and you're typing or writing on it. You want to do something like check the spelling of a word. Rather than necessarily invoking the spelling checker built into the word processing program you're using, you simply use the spelling checker you purchased the other day. It would be tightly coordinated with the overall system, so you could take that tool or any other tool and use it to operate on the document you're currently working on. What it's going to take to do something

like this is a real solid low-level system design that provides a common data structure format everyone can plug into and that provides very easy access to all data on the system, whether it's a picture or a document or a database or whatever. Some sort of uniform style, such that any tool that's around can be used on any document. Some of that has been done to some degree with systems like Smalltalk, but nothing in any commercial sense.

Signal: That's a major hurdle for developers to overcome, but it probably won't seem like much of a breakthrough to users. How will products change in an obvious visual way that would be apparent from, say, looking at an ad for a computer of the future?

Smith: I don't think mice will be around forever. We'll move more towards something that allows you to bring the desktop metaphor closer to home, closer to reality. You could imagine your office desk actually being a display, and you actually manipulate information right on the desk.

Signal: What gets rid of the mice? Voice?

Smith: Not necessarily. You may be writing on the desk as you do now, with a pen of some sort, or using a lot of the tools you would use today. They might be touch sensitive to some degree, such that you could move things around by touching them. We're looking more and more towards making the abstraction more and more concrete. Voice has its place, but not as big a role as some people think it does. People will find they don't really want to talk to their computer. Not only that, but it's not very efficient. Imagine a voice driven car and telling it to take a left turn, but not specifying quite enough detail.

Signal: If you had the super deluxe model, all you'd have to say is "take me to the airport". Can you somehow characterize any of the work you've actually done on future products?

Smith: The only thing I can really say is that it's been looking down the line five years or so, with respect to what hardware will be available. A lot of the ideas people have had, Alan Kay in particular, who is now an Apple Fellow, require a significant amount of hardware. Some of those ideas are probably going to see the

light of day in the not too distant future, as the hardware developers catch up with the minds of the software developers.