An Interview with

RALPH and MADGE GRISWOLD

OH 201

Conducted by David S. Cargo

on

25 July 1990

Flagstaff, AZ

Charles Babbage Institute
Center for the History of Information Processing
University of Minnesota, Minneapolis

Ralph and Madge Griswold Interview
25 July 1990

Abstract

Ralph and Madge Griswold recall the development of the Icon programming language. Ralph Griswold begins the interview with a description of the evolution of Icon from SNOBOL4 during his work at Bell Laboratories in the early 1960s. In this context he describes the difficulties of developing software in a corporate environment. Most of the interview concentrates on the development of Icon after Griswold took a faculty position at the University of Arizona. The Griswolds describe the creation of the Icon Project, the project's support from the National Science Foundation, the importance of the project in graduate education in computer science and the contributions of graduate students to the language's development. Also discussed is the dissemination of information regarding Icon through the *Icon Analyst* and the project's interaction with the commercial software industry through two small software firms, Catspaw and The Bright Forest Company.

RALPH and MADGE GRISWOLD INTERVIEW

DATE: 25 July 1990 INTERVIEWER: David S. Cargo

LOCATION: Flagstaff, AZ

CARGO: The best place to start is in the beginning. This is supposed to be the history of the Icon Project; you mentioned that it's impossible to disentangle it from the events that preceded it. What would you say was the immediate genesis of what we call the Icon Project?

RALPH GRISWOLD: We have been doing programming language design, development, implementation, and distribution for a long time - dating back to 1962 or 1963 in Bell Laboratories, and since 1971 at the University of Arizona. It's a somewhat but not completely gradual transition from one language to another. The language preceding Icon was a language called SL5, which was preceded by SNOBOL4. Icon was at that point just another language, and we didn't have any idea where it was going. So at the time you would perceive the Icon Project as actually starting, we were beginning development of a new programming language without any clear idea of where it was going to go. So my view of this is that that's just what you would put on a chart or something that said, "I can't start it here. It's just an event in a long history." But it did constitute a time in which we decided that we had enough interesting ideas for a new programming language, so that we would focus on that new programming language and essentially stop work, and for the most part stop support, of earlier work. That was probably the idea sometime in late 1975, and it was about 1980 before anything actually appeared on the scene that anybody else would recognize. So in that sense I could go back and say, "Yes, the idea for Icon occurred somewhere in late 1975, maybe early in 1976. But we didn't know it was going to be successful, and we certainly didn't know what we were getting into." So in that sense you could pick a time that says it was Icon, but from my point of view it's a segment in a longer period of research and development in programming languages.

CARGO: Just as an example of a point in time, when did you pick the name Icon for the new language?

RALPH GRISWOLD: Well, I am on record, I think, as saying that picking a name is the hardest part of any of these things. We actually had some other tentative names for a while. One of my colleagues, Dave Hanson, was fond of

the name "s," which I guess was based on the idea of the C programming language at Bell Laboratory - a short letter s would stand for something: I don't know - strings, or something. That didn't appeal to me a whole lot, although I think there are some documents around with that in it. It's typographically not very attractive, and it's a little hard to say and justify. The name Icon was picked, I would guess around Christmas 1976. And I am responsible for the name; it doesn't stand for anything. It's not an acronym; it's just a word. We regret it, of course, because we didn't anticipate that Xerox and Apple were going to come along and make a household word out of it, and that it meant something different has caused us a certain amount of difficulty, but we didn't have any way of anticipating that. It's a labeling, that's all. We didn't pick the name and then start the language. We had to find a publication name at some point and so we did.

CARGO: What was the first group of people that was working on this project?

RALPH GRISWOLD: The first group of people would be myself, and David Hanson, who was I guess at that time new on our faculty.

MADGE GRISWOLD: His previous work was for his dissertation. It had already been completed, and he had gone away and come back at that time.

RALPH GRISWOLD: His work was in a previous language, SL5, that I mentioned. And he had gotten his Ph.D., gone to Yale for a year, and come back to our faculty. But he was one of the main people; Tim Korb (John T. Korb), a graduate student at that time, working on our grants with another person, with somewhat lesser involvement, Walter J. Hansen, who was also a graduate student at that time. There were a lot of people involved in it from time to time, and it's hard for me to separate out the different languages, because there was a flow between one and the other, and sometimes there was more than one language going on. And if you were still working on SL5 and working on Icon, so I would have to go back and consult records to sort that out. But certainly Dave Hanson and Tim Korb would be the primary people initially involved with it. Later on, during the later part of the development the two main people involved were Cary Coutant and Steve Wampler, who were both at that time Ph.D. candidates and research assistants in the department. But a lot of others in little bits and pieces. I would have to reconstruct that to be fair, to credit

hundreds of people, but those were the ones who were mainly essentially employed to do it, in some sense.

CARGO: Did you have some specific goals for the Icon Project, as you constituted it?

RALPH GRISWOLD: Well, again, from my point of view it's a segment of a long sequence of events that are similar. Well before the Icon Project, in work done at Bell Telephone Laboratories originally and then at the University of Arizona, we had been involved in design and implementation of programming languages and their distribution, general distribution to the computing community. Icon was really our third or maybe fourth significant experience of this kind; there was a lot that went before it. We viewed Icon as an improvement over previous programming languages. It is important to understand that the research motivated it, although a lot of the things the Icon Project has done have not been strictly research. They have been in making this work available to the computing community and a variety of things that you really wouldn't call research so much as engineering and development. But my view of this is that we did research on language design over a period of time. Every so often we would have enough of what we thought were new and interesting ideas to put together to justify calling it language, fleshing out the language, filling in the features you need for real language as opposed to just academic ideas, implementing it, reducing it to practice, and distributing it. I think the thing that characterizes our work at that time at least, which is unusual in academic environments, is that we did bring this work to a useful product that people could use, distributed it, made it widely and freely available, and supported it. I think that when we started the Icon Project, we had every intention of doing that. We had done it before, and it wasn't the first time we have done it. We felt at that point we had many years of experience collectively. I thread through it from the beginning of 1962, so by, say, 1978 there was a lot of prior experience. This was viewed as an evolutionary development and the culmination of a lot of that work. I suppose I viewed it probably as the end product of all of this; I still do. So we or I decided that we would take these new ideas and fill in the pieces around them, build a complete programming language, a complete implementation, and do what we had done before, make it freely available. That was certainly in our minds at the time. I don't think we anticipated what the consequences would be.

MADGE GRISWOLD: No, I would like to interject something about this. Back in the time of SNOBOL4, when it was first being disseminated there was no real understanding of how popular it would be. It was made available and the

response was rather overwhelming. Distribution had gone on in a more informal way than it has with the Icon Project before. So this was a more organized repeat performance of something we had already done. When we first made Icon available we didn't know that it would be appealing to anyone for sure. We thought it might be, but we weren't sure at all. It was made available to see if anybody wanted it, not with the goal that it would be disseminated to so many people or whatever - but it was just made available, and there was feedback, and more things came out of it. So it wasn't that there was a goal to disseminate it to a specific number of people, or by a specific amount of time, or that there would be so many releases, or anything like that. It was just a, "Well, does anybody want this?" "Yes." And more copies went around, so does that say anything to make sense to you?

RALPH GRISWOLD: Well, I agree with that pretty much. I think that we were not, of course, certain how much interest there would be in this. Having your own self-interest involved, you don't know how good something is or interesting it is to somebody else. Of course we couldn't have any idea where it was going to go because that depends on a lot of things. It could have gone lots of different ways for various reasons.

Personally I hoped it would develop into a full-fledged programming language that people would be interested in and use and would achieve the same kind of acceptance and interest that previous work had. But it wasn't at all certain that that was going to happen. When you are deciding whether to make an investment in this kind of thing, which is a very substantial investment in an academic environment because there isn't any mechanism really for supporting it, you have to really do it outside of existing support mechanisms. If the initial reaction had been unfavorable we wouldn't have continued. On the other hand I certainly knew what we were getting into in terms of distributing and supporting software, because we distributed a lot of SNOBOL4 over a period of time, made a lot of implementations. So we knew pretty much what we were getting into, although computing is a moving target, and a lot of things are different now than they were when we started. It's been some fifteen years or approximately, and lots of things in the computing world have changed in those fifteen years. There weren't any PCs then, for example. We could not anticipate that. Well, that's had a big impact on us.

CARGO: We mentioned outside the interview that you had gotten a fair amount of funding over the years from NSF.

RALPH GRISWOLD: Yes, prior to coming to the University of Arizona I was at Bell Telephone Laboratories, which is an entirely different matter. I came to the University of Arizona in 1971. They were building a new computer science department, and I was the first faculty member in that department. Universities rely, to a large extent, for their support of research on grant funding and do not expect funds at a state university from taxpayer sources to support research. I didn't know anything about this, coming from Bell Telephone Laboratories. The administration at the University of Arizona was obviously hopeful that I would begin developing a funding program that would provide the basis for developing a department, and obviously I was going to need that to support my own work.

So I rather naively went out to the National Science Foundation with a proposal in 1971. It probably would look embarrassing in retrospect since I had no experience and no guidance from the university. It was funded, and the funding started in 1972. I went to the National Science Foundation because the university felt that it was the best source of funding for this kind of thing, because it is the only federal funding that really funds pure research, and it's free from so-called product development. It doesn't expect you to produce a tank or something like that. In many respects it's an ideal source of federal funding among the possible agencies.

The National Science Foundation has supported this work under several grants since 1972. One grant, I think, has kept the same title pretty much since that time. There has been more than one grant on one time on occasions, but we have had continuous support. The support has been, of course, obviously essential for what we have done. It has been all pretty much on the basis of a single investigator grant as opposed to a coordinated or interdisciplinary grant. The funding has been primarily for personnel - support for graduate students, summer support for me as principal investigator, and modest amounts of things that go with that. But it has been primarily funding students and faculty to do this kind of work. Without that kind of support we certainly couldn't have done what we have done. We had to rely entirely on the National Science Foundation. The University of Arizona has been supportive in providing things that are not appropriate to buy with grant funds. It isn't appropriate to ask a grant to get the manufacturing of diskettes or something like that. That's not appropriate. I don't know what the total amount of that funding is, but it has been on the order of $100,000 a year, probably, for the last eighteen years or something like that. They would support more than just things like Icon support. We would work on program portability, which was related to some of the implementation work, and it supported work on instrumentation and analysis of software,

things like that. So that hasn't all been just funneled down to programming languages. It never has been to design programming languages; it has always been to investigate programming language facilities primarily for non-numerical computation. Programming languages have been by-products of this. Not that they haven't been pleased with this, but the thrust of it has been to produce new ideas, not to produce diskettes, or package products, or anything like that.

MADGE GRISWOLD: I concur. No, I do think that that's an important distinction to make, that we didn't say we're going out to get grants to design a programming language, or I shouldn't say *we* in this sense because I have never been supported on this grant; that's a separate thing. But we didn't go out to say we are going to design a programming language. That's why we're looking for funding. This is really pure research ideas, a much more diverse set of ideas perhaps than it looks.

RALPH GRISWOLD: Many of which are not embodied in any programming language, but rather in papers and things that have appeared over the years. There is one point here that might be amusing in terms of the change in attitude toward this. The first proposal that I submitted, which was to continue work on SNOBOL4 that I had been doing at Bell Laboratories, as I said was naive, because I had no experience with grant funding and sent it off into the blue according to their format without even any assistance from The University of Arizona. And there were no other faculty members in computer science at the time that could have provided any advice in the technical area. But one of the criticisms of the first proposal, and it was revised I believe to reflect this, was the National Science Foundation did not want to provide funding to support development of programs in the academic environment because their experience with this had been poor - that academics moved and left and the software came to nothing. And nothing good had ever been produced usefully in the academic environment. So they were willing to support research, but they were not willing to support development of software because this was not something one could do in an academic environment. I disagreed with them, and their attitude now is different; they do realize that software as a culmination of research work is an important contribution to the computing community. At the time that had to be taken out of the grant, that it would produce something for distribution. They didn't want to see that. Tom Keenan, who was the program monitor on my first grant and who still at the present time is my program monitor at the National Science Foundation who I understand is retiring in September, was the one who made that comment. I think it

reflected their collective experience at that time. The National Science Foundation was supporting software kinds of work in universities that faculty members just didn't stand behind their product essentially. When things got tough they moved, and the product became useless and so they weren't about to do that. They have what I would view as a more enlightened opinion at the present time. Also, the situation is much better. There is a history of people producing useful software in universities, and that's been worthwhile. There wasn't at that time.

I would say in this respect, and I think this applies to Icon, we have tried to be a model for software distribution from universities. I mean, that was a conscious intention on our part to prove that you could distribute software of sufficient quality to be useful and support it from a university environment. I think we, maybe not with Icon but maybe some of the other work, may have been one of the first to do that. There are lots of examples now - Berkeley UNIX being the main one. At least I was trying to prove something, that you could do this in a university environment, and people didn't believe me. And I guess we have been modestly successful, but it is not always clear (at a considerable expense, incidentally). I am actually only thirty years old; I just look like this. [laughter]

CARGO: Who decided what format the distribution mechanism should be in?

RALPH GRISWOLD: I did.

CARGO: Was that based on prior experience with SNOBOL?

RALPH GRISWOLD: We got a lot of experience. Of course, again, our experience developed as the computing facilities and software in general developed. Our first distribution software from Bell Laboratories would be in 1964 and 1965, and things were very primitive then. We distributed punched cards - binaries. Tapes were unreliable. People didn't understand software. There's a lot of information about that in the book on the history of programming languages that came out of the HOPL Conference under SIGPLAN. It is based on a lot of experience, but the other thing is that the environment around that changed all the time. What we did originally would be ludicrous now. We couldn't have begun to do then what we are able to do now. Nobody had invented a diskette; nobody had to decide that software might need to be licensed, or that it might have any value, or that it might need to be patented, or it

might not, or what public domain meant. Nobody ever heard of shareware - any of those kind of things.

But I would say, as far as Icon is concerned - our policy on distribution, there are lots of issues here. There's how you do it, and there is policy. It certainly was formed at Bell Laboratories from the initial distribution of software from Bell Laboratories, which was in the public domain, and at a time when Bell Laboratories did not understand what software was at all. It got into the public domain without their lawyers really being aware of the fact that the program had any value, or how one might do it. In fact, they were trying to patent software.

I had decided a long time ago, and still think, that unrestricted public domain distribution of software has many merits. Lots of people put restrictions on software that are vacuous but just annoying. Even little license fees bring in lawyers and university councils, and problems with state constitutions, and even a copyright that says, "This is copyright and you are allowed to make copies if you want to," introduces legal problems. We have felt for a long time that just simply having no restrictions whatsoever was much the cleaner way to go.

A lot of people in academic environments as well as business environments have seen the opportunity to make money at this. And so lots of universities have gotten into business selling software, licensing, and so forth. I felt, and still do, that that tends to inhibit distribution. Not that we give things away; we charge a nominal fee for it. We have to, although we didn't originally. Our view was to get software as widely available as possible you don't put restrictions on it, especially vacuous ones that are designed only to make someone feel comfortable about pseudo-legal mumbo-jumbo or something that looks professional. We are one of the few organizations that really does distribute public domain software. Almost all of the software that I know about comes with some kind of licensing requirement even if it doesn't cost anything. I am sure that I have a different opinion than many people do, but we did start out with that view and it was based on that past experience. Then the mechanics of it were responsive to need and to requests basically. We tried to do what was reasonable with respect to what people seemed to want. We can't do everything, but we have gone a lot farther than I think most people would be willing to in an attempt, I think, to make it as widely available as possible - and that's partly altruistic and partly self-interest, and I couldn't sort those two out. But I am not sure that's even the answer to the question you started me out, but it's an answer. If it's not responsive, remind me of what the question was.

CARGO: It sounds fine. One of the things that has come up as a thread in what you have been talking about is the development of software languages, in particular at Bell Labs and then in the university environment. I am curious about what other contrasts you might make between Bell Labs and the U of A in terms of being able to choose the direction of your work, the amount of administrative overhead you have had to fight, or the amount of bureaucracy that you have had to deal with in those environments.

RALPH GRISWOLD: Well, that's an big question; it covers a lot of ground. It's an interesting one, and there's a lot of collateral documentation on that. The situation in Bell Laboratories initially was, while we were developing programming languages, not having any idea that anybody wanted to use them and then discovering that there was an interest that we had not anticipated, Bell Laboratories had a policy on releasing information. I mean, it does, after all, have proprietary information. If you wanted to give a technical paper, a technical talk, or something like that, you had to get a release. It went through some kind of process to make sure this wasn't important proprietary secret information and that it was technically sound. Dave Farber, who was one of the initial SNOBOL people with Bell Laboratories working with us on that, was invited to give a talk at the University of Michigan on the work on SNOBOL, and he wanted to give them a copy of the program. He looked up into the air somewhere and decided that a permission to give a talk on software constituted permission to release the software. Nobody at Bell Laboratories knew any better, because nobody had any idea of what a program was, at least in the patent department and the release procedure. That sort of happened and then that was used as a precedent for everything that followed. A pyramid was built on this precedent.

At some point Bell Laboratories discovered the value of software, I guess, as their device work was becoming less and less a part of their budget and the people in the patent department and the lawyers were concerned about (A) the company's resources, and (B) their own jobs. They decided the programs really were valuable and they wanted to try to patent them; that was their initial view. They discovered that our work was already in the public domain; de facto in the public domain. There's a long, long story there, which I won't go into here. They wanted to protect things and ultimately patent them. They finally said, "Well, if you ever do anything like that again, don't release it without permission," or something like that, at which point I decided I would move somewhere else. There is no problem with

that in the university. There may become a problem over intellectual property, but there has never been any problem about release at the university. It's entirely our own business and nobody pays any attention to it. They would if we put licensing requirements on it, probably. But the environment is quite different. At a university you don't have to get or couldn't get a release for a paper or anything like that. It's entirely your own business and your own responsibility. So there isn't the mechanism in the university.

It's conceivable the universities at some point could become and are becoming interested in this. But from our point of view there has never been any problem about the release aspects of it, or administrative overhead - zero, essentially, at the university. However, at Bell Laboratories there was a problem at times, because sometimes it just took a long time to get a release. I remember one time it took a year to get a release on a paper, because they were trying to decide whether the ideas in it were patentable or not. It just sat there for a year. That's not a problem in a university; it's a different environment. Now, of course, Bell Laboratories is very involved in software distribution. They have been very generous in making things available to the public, but it's also a big business for them. And I don't have any first-hand experience with that, but that's easily discovered.

CARGO: What about selecting your research directions? I don't know anything about the kind of autonomy you might have had at Bell Labs, but that contrasts with, for example, your need to secure funding from NSF now.

RALPH GRISWOLD: The Bell Laboratory situation is easy to describe. It depends where you are and when you are in Bell Laboratories, but where I was nobody cared what I did and I spent nearly ten years there, never having an assignment, never giving a report, never having a budget, never having anybody care very much. My director of the center that I worked in had the philosophy that if you gave some intelligent people some tools they might come up with something interesting eventually - sort of the sandbox philosophy.

And so I had zero supervision for nine years at Bell Laboratories. Every so often they would recognize we had got some approbation for something, and that allowed us to keep doing our own thing. But nobody cared in the least, nor did we ever have any goals except the ones we generated internally, which we certainly had. We had no supervision so that wasn't the problem. It could have become a problem at any time they decided they might want to

put us on a project. We were successful enough that they left us alone basically.

In the university environment you are entirely free to choose your own research directions, but you are responsible for the results, consequences of that, and for getting the resources to do it. Certainly over a period of time the attitude of funding agencies, and particularly in my case the National Science Foundation, has certainly affected what I have done. I mean, there are some things that they don't support. They don't support product development thoroughly. There are certain kinds of funding they give you and not others. In my area they are much more likely to give you support for students than they are equipment. So certainly I have, at least subconsciously, been influenced in the kinds of requests I have made by the kinds of things I knew they would fund and the kinds of things that were appropriate for them to fund, and that has undoubtedly had some effect on it. On the other hand, the National Science Foundation is very good about giving you freedom to do what you think is best, or what the best opportunities come along in research independent of the details of your initial proposal. It took me a while to figure this out. But they're open about this, too. So as long as I stayed in the general guidelines and the general area that I asked for, and as long as the work has been successful in their terms, which means publication and degrees, that has worked out fine. But there isn't any question that having to get funding does affect the kinds of things you do.

TAPE 1/SIDE 2

CARGO: More on the issue of the NSF funding and the way it limits your choices, given that you have had to work around the limitations they're putting on what you can spend money to do, have you had a consistent policy for how to bridge that gap?

RALPH GRISWOLD: Well, in the first place, I don't view these constraints as being major problems. There have been times that we've felt that we needed equipment and the National Science Foundation did not themselves have the budget for equipment. And there are policy questions, like does NSF funding go towards departmental infrastructure development or not? These policies have changed over the years. When I started out at the University of Arizona I was *the* only faculty member in computer science, and there was no funding. We have developed the department over a period of time. It has been very successful in terms of getting research funding,

and when that gets to a certain point then there are opportunities for getting resources from places other than your own individual grant. For example, a few years ago the department got a department-wide, then called CER grant for departmental development, equipment and personnel - about $3.6 million for five years, and that has recently been renewed under another program with a different name. But the point is that when the department has a lot of funding available to it then individual grants, especially ones that are successful, can expect to get some support for that. So I can get support from the general department grant that I could never hope to get on a single individual investigator grant. And this is perfectly appropriate, and the National Science Foundation knows and expect this, and in fact, the success of the individual grant is what leads the department to get funding at a broader level. They buy big pieces of equipment that no single person can get. So then you begin to share the resources with other people. There are never enough resources, especially personnel resources, which most NSF funding has been for, but it doesn't fund professional programming and things like that generally.

So, yes, the way around it, to the extent there is a way around it, is to be a part of a large, successful organization that has lots of resources where things can be allocated in different ways to cover those. We don't have the kind of funding obviously that a department like CMU or Stanford does, but we do have funding that does help. Department-wide funding has definitely been very helpful in my own research program in the Icon Project. For example, one of the things that that funding supports is software distribution, and that's a specific thing in the grant proposal and one of the things the National Science Foundation wanted to see continued. So there's money in the general department-wide grant for software distribution, which we draw on for our work. We'd find a way, but that's a nicer way than having to go out and beg on a street corner or rob a bank. But that's sort of my pragmatic answer to that. You don't look to the university for everything; moral support yes, but not major funding. They just don't have the money for that and they don't expect to provide it.

CARGO: Does the funding that comes in from the duplicating fees for Icon materials go back directly to the university general fund, or back to the Icon Project in effect?

RALPH GRISWOLD: No, it goes back into an account locally that I have control over. It's internally an accounting problem. It doesn't go back to the university as a whole; it goes back into a fund in our department that I can use for,

you know, buying more diskettes or equipment or compilers for software development.

MADGE GRISWOLD: Well, the printing bill for the Newsletter.

RALPH GRISWOLD: Printing is a big item.

MADGE GRISWOLD: Mailing the Newsletters.

RALPH GRISWOLD: Mailing - all of those things. The Newsletter, which is distributed freely, is subsidized by the extra revenue we make off of sales of diskettes and tapes and books. We are not allowed to discount the book because that would be unfair competition with local business. We do get the book at discount; we're a bookstore basically, so we do make some money on that. That funding is paying the expenses of several people here at the conference from the Icon Project. It's paying the travel and living expenses for students.

MADGE GRISWOLD: Nothing that's important.

RALPH GRISWOLD: Yes. And really, you know, there's some restrictions on that. But the main problem is that the university runs it so that it has to be on a positive cash flow basis. And it's accumulated some money that I can't spend [laugh] - money accumulated in the past, but on a yearly basis I can't eat into that. It's not a big project, but it's a little annoying sometimes. I would like to maybe buy a computer that I want to develop Icon on - a PC. I can't touch that money. It's a little annoying, but, yes, it does a lot of things. And it's under my discretion, my control. So for the most part it's no problem, but we don't buy cars from it - things like that. We can't eat off of it, but as long as it's related to the Icon Project and in my opinion it is a suitable expense, and as long as I manage to make up a budget a year in advance that covers it, which I have to do, then it's purely up to me. And that's a very nice thing. Grant overhead typically goes back to the university, and the university gives some of it back but keeps some of it. That varies from university to university. Some universities it goes back to the state legislature; the university doesn't see it. It's like we're in a little business, and it's our own as long as we don't do anything inappropriate.

MADGE GRISWOLD: Well, it's kind of interesting to know - this came out in the Newsletter a while back - that nobody is actually paid a salary to be a part of the Icon Project. In other words, their funding comes from somewhere else, if their funding comes at all, and so the Icon Project isn't actually paying a salary to anybody to administer the Icon Project. What funds come in from that are buying materials to keep the project itself going.

RALPH GRISWOLD: Yes, in that sense (and that's an important point, really) the department provides the personnel that handles Icon orders, distribution, duplication of diskettes - all those kinds of things. You know, replace the stock and all that. The Icon Project doesn't have to pay for that. So if it did, we probably would have gone broke a long time ago. It's just something that the university provides support for; the department does, or the department grant does, depending upon who's doing it. There are a lot of people involved in that, and the personnel expenses would be very high. It's not a big enough business to support the people, so it would be impossible to run privately that way, because you would have to have a certain volume in order to have a staff and handle all these things. So the personnel costs of the Icon Project are entirely supported outside of it except for research. It doesn't support any staff salaries at all. Well, it may support a little secretarial work for the research segment, but none for the Icon Project segment. The Icon Project is totally self-supporting, but it does have donated labor, including the people you are talking to. But no salaries are paid out of the Icon Project. If they had to be you could not manage it; it would not be an economically viable operation.

MADGE GRISWOLD: Well, the university does not provide anything specifically labelled the Icon Project. No one is funded to work on it specifically - maybe a secretary as a part-time thing.

RALPH GRISWOLD: Well, it's not an official thing.

MADGE GRISWOLD: It's not in the job description anyhow.

RALPH GRISWOLD: It's just part of the department's secretarial staff. It's both secretarial and laboratory staff. Laboratory staff handles diskette duplication, writing tapes, and things like that. It's just part of the general support for the department. The department has been supportive of this operation. It believes it's the right thing to do. It

believes it's good for the department, good for the computing community, good for the university. It's appropriate, and they support it just like they would support secretarial support for any faculty member - something like that. It's just part of the general support. Part of that is funded by the state; part of it is funded by grants of various kinds. But without that support, without that agreement to do that, it would not probably be possible to conduct the Icon Project.

CARGO: There have been a couple other names that have cropped up in reading about the Icon Project - one being The Bright Forest Company, and the other being Catspaw.

RALPH GRISWOLD: Catspaw has no connection to the Icon Project directly. The situation is this. Let me start with Catspaw, because that's one leg of the arrangement. Catspaw is a small software house. The president is Mark Emmer, who has been doing implementations, implementing and distributing of SNOBOL4 commercially for some time starting with implementations for MS-DOS on a PC. For whatever other reasons, as a labor of love, but it's a small commercial operation - very small - that does high quality work, but has been involved in SNOBOL4. I have known these people for a long time because I have been involved with it, and certainly, as we have for many other implementations, provide information freely a long time ago.

The question came up a few years ago. As a matter of fact, the president of the University of Arizona, Henry Koffler, in a department review that he attended and we had talked about Icon, made some remark about making it a commercial venture. The university likes this idea of technology transfer because they see money in it for themselves. I was opposed to it at the time. It didn't seem appropriate to me, but there is a feeling that at some point things developed in an academic environment ought to be supported commercially, or otherwise they are not plausible. The National Science Foundation still feels this. A few years ago we looked at this situation and thought we might explore the commercial possibilities of Icon, in particular for Macintosh. Madge and I as individuals contacted Catspaw because they had done the SNOBOL work, and they kept in touch with Icon, and were involved in Macintosh product development, had the necessary technical expertise to do this. Out of this contact, which has nothing to do with my university responsibilities, we decided to engage in a venture to develop this product for the Macintosh. And the structure of it is that there are two corporations involved in the joint venture. The joint venture

is called ProIcon, and the two corporate entities involved in it are Catspaw, Incorporated and The Bright Forest Company, which Madge and I formed as a small, privately-held corporation of our own to encapsulate this essentially, plus some other possible things that we might do in a business sense. That operation has nothing to do with the Icon Project. As a matter of fact we have to be careful not to do anything inappropriate or commingle things. It's perfectly legitimate and legal, and the whole thing is filed with The University of Arizona under the state laws about this kind of thing - lots of faculty members have entrepreneurial activities. This is not a very highly entrepreneurial activity, but it's of an entrepreneurial nature. So it's basically a private enterprise that is not officially part of the Icon Project. There's no monetary connection; there's no business connection or anything with the university. It's entirely through my entrepreneurial activities, which are encouraged by the university, and then Madge is involved with it.

MADGE GRISWOLD: Well, I am the president of The Bright Forest Company. I think it should also be pointed out that the name of the joint venture is The ProIcon Group, and the product is ProIcon, the current product name, just for the record. But no, it has no connection. The only thing that is, that as a matter of courtesy, I believe persons who purchase ProIcon are automatically put on the Icon Newsletter mailing list. But that's simply a matter of courtesy and we communicate those names over to the Icon Project.

RALPH GRISWOLD: Yes, we do that for a lot of other people, too.

MADGE GRISWOLD: We do that for a lot of other people too.

RALPH GRISWOLD: We get any name we get. [laugh] We'll take it any place we can get it. But there's no business connection between the two. No commingling of funds.

MADGE GRISWOLD: No funds going back and forth.

RALPH GRISWOLD: As a matter of fact the Icon Project purchases the MS-DOS 386 implementation of Icon from Catspaw totally independent of everything else, because they provide it.

MADGE GRISWOLD: Well, there is one simply cooperative relationship, and that's with the new Newsletter, the *Icon Analyst*, which is just coming out. And that is that it is jointly published.

RALPH GRISWOLD: Well, it started to be published by Bright Forest.

MADGE GRISWOLD: Bright Forest and The ProIcon Group. No, I am sorry, Bright Forest and the Icon Project.

RALPH GRISWOLD: Yes, it gets complicated, because The Bright Forest Company can engage in any kind of activity it wants to. It happens to have this joint venture for a particular product with Catspaw. The *Icon Analyst*, which is published by the Icon Project, is supported by The Bright Forest Company. I mean, here's one of the editors. But it doesn't get any money back out of it.

MADGE GRISWOLD: It's simply that its name has been contributed to this. And its editor is part of The Bright Forest Company.

RALPH GRISWOLD: And it's printed on a laserwriter that The Bright Forest Company owns. But there's no financial involvement there; it's really contribution. It does confuse the issue I suppose if you want to know the relationship. We kind of like that - kind of mysterious, you know. [laugh]

MADGE GRISWOLD: But since it actually comes out and says that in the *Analyst*, I thought it might be appropriate to clarify.

RALPH GRISWOLD: That's a good point, but that has nothing to do with The ProIcon Group or Catspaw. That's just one of Bright Forest's...

MADGE GRISWOLD: No, that's one of Bright Forest's...

RALPH GRISWOLD: The Bright Forest Company might engage in other things. It might very well decide to engage in a commercial implementation of Icon that has nothing to do with Catspaw. We have no binding agreement with Catspaw to prevent that. Or it might decide to publish a book, or something like that. It's a vehicle for doing things that cannot sometimes be conveniently done within a university environment, or that might sometimes be inappropriate. We sort of set it up in anticipation. For example, the assistance of the Icon Project is dependent on benign neglect. As I said, the department supports it. It thinks it's a great idea, gives it a lot of publicity. And it does all the right kinds of things that you should do in an academic environment. But we could get another department head that would decide that it ought to go like that.

MADGE GRISWOLD: We could get a new university president to decide that. We are about to get a new university president, and he or she might decide that's not appropriate.

RALPH GRISWOLD: Or somebody could pass a law that says this kind of activity is inappropriate, or somebody in the university might decide the way the funding is being handled is inappropriate. One of the things we've thought about is it might have to take over the distribution. I wouldn't like that particularly, but it would be a possible way of maintaining it if the university environment became inappropriate or hostile, or if I retired - something like that. So it's a little planning there. But you know, it's not your typical, tangled corporate web of organized crime or anything like that. [laugh] I am sure it looks a bit mysterious to people as to what the relationships are.

MADGE GRISWOLD: We like that; it's kind of fun.

RALPH GRISWOLD: And of course I wear two hats, and I have to remember which one I am wearing. But lots of academics these days do that, and it's encouraged generally by universities. If we made any money, I am sure they would like to have a piece of it, but that's problematical.

CARGO: I guess, to go delving into more into some of the mechanics of how you have been doing things at The University of Arizona. You have obviously had a large number of graduate students over the years - probably more doctoral students than anybody else working on Icon. Have you had much in the way of long-range planning about

what you want these people to work in, or has it been a matter of they have their own dissertation ideas and you recruit them on the basis of what their ideas are and how they apply to Icon, or how does that process work?

RALPH GRISWOLD: Well, it varies from time to time, and it's very pragmatic. It depends. They are just young professionals, and in many cases they are just as talented as the faculty members are; they just haven't gotten as far along in the system. Talent is the rarest commodity. And I try to take advantage of that when I find it. There is a tendency, I think, on the part of faculty members in general, and probably specifically in computer science and software, to view graduate students, supported graduate students as cheap labor. And it's not at all uncommon for them to have some kind of an internship where they sort of have to do whatever needs to be done until they earn the rights to do their own research.

My preference is to give my students much more opportunity and capacity than is typical. And in fact, that's, in my opinion, one of the reasons why some of this work has paid off, because it certainly could not have been done all from my own ideas and original effort. So I have given my own students, for the most part, a lot more opportunity for individual expression. However, there is a constraint that what they do has to fall within what's reasonable for the kind of funding I have, and sometimes within obligations or expectations that may have a larger context associated with it. So I don't attempt to take on students who want to work on something, or have ideas in areas that are outside those that I feel I can comfortably direct, or have the time to direct, or which I have funded. So it's a tension here.

Many of the things that have been done, specifically about the Icon language, have been purely a result of an individual at a particular time and place having certain interests and ideas. I have been an opportunist in this. That is, when there have been ideas and opportunities, and we have a student with a talent and ability, I have tried to take advantage of that. So certainly it has had an effect. Icon would be a very different language if other people had been involved with it with other ideas. They're not all my ideas by any means.

During a graduate student's time - in my case, it has been almost entirely doctoral students - there's a process that they're getting used to - working independently. Some are too independent, but most of them have trouble getting started independently and sort of expect another homework assignment that's just bigger. It takes them a while to

recognize that they have responsibility for initiating ideas and developing a research topic, as well as just solving a problem or executing a program of some variety. So the students typically go through a number of false starts on this, and very frequently things don't come out the way that I expect. Dissertation topics don't come out to be what I thought they might be. And I typically don't assign a topic and say, "Go do this." Nor do I typically expect a student to develop a topic in advance of doing any research and then carry it out. I expect that to develop as a process of learning how to do research and conducting it.

There is a factor here that probably would not be evident. At least it is a factor for me. There are times of feast and times of famine. There will be times when the faculty have a lot of funding and with very few students around, you're competing for those students. And then there will be times there's a lot of students around and not much funding, and students are competing to get it. I am certain there have been a couple of times in the time that I have been at The University of Arizona that I have run out of plausible students altogether. Even at one time I sort of advertised, which I regretted ultimately. I should have waited for someone more talented to come along than what I got by putting a sign on my door saying "Help!"

Right now I have one senior student and a whole flock of students that just started and don't quite know what they're doing. When they are just starting they tend to get more specific assignments than they would if they had more developed ideas and make suggestions of their own. Also at the present time, my work that I am looking at doing and am seeking grant funding for is more of a group effort than an individual effort, so that will have some effect in my providing more direction, if you like, or more channeling into certain areas than it might be if we just wandered around looking for a neat idea, and waiting for somebody to come up with one. So it varies a lot. It varies with individuals. It's very, very much an individual thing.

There are just plain timing problems. Sometimes a person that would be ideal and ideally would like to work on this, it's just the timing isn't right. The funding isn't there or something like that. But it's a very difficult, chaotic kind of situation; at least it has been my experience. The other thing is we need a lot of work to be done, and somebody has to do that work. It isn't always a directly related thrust toward the doctoral work. We get some support from laboratory staff - people like Gregg Townsend, who are supported by grants, we get a decent share of their time. But

some work just has to be done, and so the question is how you get that done. If you are committed to doing a big project you just plain have to get it done, and the people that you hire have to be willing to do it and they have to make dissertations out of it somehow - one way or another, which is quite different from truly inspirational things. You have an arbitrary amount of money and say, "Okay, here's the sandbox philosophy again. Let me know when you come up with something neat."

MADGE GRISWOLD: Well, you don't exactly leave them alone to come up with something neat. You stretch their minds when they come in. I worked as a research person under you at Bell Labs, and I know that you can sit for an hour with him and have him stretch your mind this way and then that and you are not feeling it's upside down, but you have done something different than you have ever done before. You're contributing to the process, but they're finding a way.

RALPH GRISWOLD: I suppose. It just varies a great deal. I think my present group of students would feel very uncomfortable because most of them don't have any ideas particularly of their own. They may not quite see where things are going, nor do I. But it's work being done more on a cooperative project - people doing different parts of things. I mean, I sort of have an idea where it's going. I think they probably at the moment if you asked them, even the ones that you may interview at about this time, would probably express some degree of discomfort about what it's like doing this kind of thing. But they're new at it, and they're uncomfortable with it. You know, it's very much a people thing, and people vary a great deal. And now you're talking about, in some cases, people with very considerable talents, but also in many cases, immature. These are very different and difficult interpersonal kinds of things. It's not just turning a crank and getting, you know, science or a project out of the other end. It's lots of individuals and the most talented ones are often the most difficult. If you are going to do research and get things accomplished you need talent, not just hard work. You need ability; you need ingenuity; you need creativity. And sometimes people don't work well together. It's very much an individual thing. I think I could do this for 200 years and not have any coherent idea of how to approach it, except just on an individual basis.

Incidentally, the success rate, in terms of getting degrees in this research that has been supported by the National Science Foundation has been quite high. I have lost perhaps four or five students over the last 18 years, failing to

complete. Something like that; maybe not that; I haven't actually counted. I could count.

MADGE GRISWOLD: That includes master's and doctoral.

RALPH GRISWOLD: We don't do much research at the master's level any more; we used to.

CARGO: That's four or five out of how many?

RALPH GRISWOLD: Four or five out of probably 30 or 40, all told. That's counting master's degrees. The number of Ph.D.'s is in the order of 10 or 12, I guess at this point, with maybe more. I don't know. I don't count. I don't put notches on the handle of my revolver. I attribute that to a good part in trying to be flexible about matching people and their abilities with opportunities. It's sometimes chaotic though. Sometimes things don't go at all the way I expect. I don't think I have ever had a grant do what it thought it was going to do in any coherent sense. They accomplished lots of things - sometimes much more than we anticipated, but it's not like saying you are going to build a tank and delivering a tank at the end. It's more like saying you're going to build a tank and delivering a toothpaste tube or something like this. And that may not be apparent because the Icon Project looks like a very well-organized monolithic effort to do something very specific. But a lot of things have gone on in connection with it that are not part of Icon, or never were realized in an existing language. Not everything is in a language. There are lots of things that have been published and done that are interesting intellectual contributions, we hope, that fall outside of what you would see if you really looked right down the line at Icon.

CARGO: I think I am going to stop this tape...

TAPE 2/SIDE 1

CARGO: When you have more things to do than you have people to do them, how do you wind up making decisions about what things are to be done?

RALPH GRISWOLD: I should put a parenthetical remark in front of the answer to that. If you look at the Icon Project, what it's actually done, especially all of the implementations for various machines, and the cooperation with a large number of people outside of The University of Arizona who contributed to this, most of that work doesn't resemble research. And it isn't something that I feel I can reasonably ask a graduate student to do, and so I do it myself. That's the answer to your other question. If there is more work to do than I can do, how do I decide what to do? I usually don't decline to do things. They may take a little longer, or I may put more time into it when I do it myself. I'm rather focused in that respect. I don't think there's anything that we really wanted to do that we haven't done. We have found a way to do it. Now, sometimes I have had to work very long hours, and sometimes I have had to put other things out of my life. And we have even done some things I didn't want to do that way.

So that's some kind of an answer with respect to the Icon Project - that it's just displaced other things in my life. But probably the effect has been more subconsciously to restrict adventures when there wasn't anyone evident that could carry them out. There are some things that we haven't attempted, because I didn't see where we would have the personnel at the time to do them. These include things like environments (programming environments, in particular), platforms, and things like this. Debuggers - things like that. I just felt that I couldn't see that we could do it on the basis of the resources we had. It probably had some effect in picking our directions. And sometimes things get postponed a bit, or they take longer than you expect. Also, I would say, to some extent, it's a question of priorities - what can wait and what can't wait. For example, we do get, when it's available, some support, professional programming support from the laboratory staff in our department. But it's resources contended for, and sometimes, unlike most things we do, we have to wait six months, or maybe a year to get it. You just adjust your priorities, but I don't think it has adjusted my overall perception of where we are going. I don't know that that's really an answer, but I don't think that, except for undertaking major things that we might have and just didn't at all, that we have ever had an availability of personnel resources cause us to make a major change in what we were doing. I could be wrong about that, but I don't recall any cases.

CARGO: Your remark brought up two questions, the first of which is you talked about work being done outside The University of Arizona. Could you comment about the kind of work that's being done and how people are staying organized with the Icon Project at the U of A?

RALPH GRISWOLD: Well, again, this kind of thing goes back before the Icon Project, but it was probably more significant with the Icon Project and earlier work. By making our work freely available, and by I hope its intrinsic interest, we have had a number of people volunteer to make various kinds of contributions to our work - people that we don't provide any financial support for and they do it simply because it's interesting, or because they're inspired to do it, or because they maybe share our feeling about the importance of making software products available to the computing community. It varies somewhat. But we have had a great deal of help of this kind.

Many of the implementations of Icon have been done by individuals that have only an informal relationship with us. There's no business relationship. We exchange information freely, which makes it possible, and we try to support help with technical advice when we can. But the Macintosh MPW implementation was done by Bob Alexander when he was working for a software company - purely out of his interest. And, of course, people that do this, we give them something else back. We give them recognition. And their name appears on things, and they get appreciation in addition to the intrinsic merits of what they do. The MS-DOS implementation of Icon was done by Cheyenne Wills, who is a systems programmer. He was just doing it in his spare time, because he was interested and wanted to do it. And the 370 implementations were done by Alan Beale at SAS Institute because he thought it was a neat thing. Lots of this kind of thing.

I would suppose by this point hundreds of people have been involved in one way or another. And they have done it because they were interested in Icon; they had a machine they wanted it on; they thought it would be interesting; they thought it would be fun. In very, very few cases have they gotten any kind of support from their employers. And very frequently they have gotten the opposite. The problem sometimes is not in the commercial environment; it's getting it out again. And this is sort of an informal collection of people who are participating, becoming part of something. They obviously get some gratification from it. It maybe comes in various ways, like they wanted the implementation for this. But most of them would like to see their implementations be made available to others. The people that do this, obviously, have the personality and value systems that go along with making this kind of an investment. The surprising thing is that the vast majority of these, not all of them, but the majority of these efforts actually produce useful work. They don't just produce half-done implementations that don't work and are an

embarrassment. We also have had to bring all that code back in and centralize it and institutionalize it, which is sometimes pretty painful. I mean, it's not uniformly high quality, but much of it is. And there's certainly lots and lots of things that we have done that we couldn't have done otherwise - and resources that were just given to us freely. Not entirely freely, because we have had to put some effort back into supporting these people and encouraging them, and actually making use of their work. Large integer arithmetic is an example done by the Chris Smith at Convex Corporation - we didn't even know he was doing it until he sent it to us. We never would have had the resources to do that.

And it doesn't all come to good ends. Lots of people have, you know, great ideas and ambitions that don't come to fruition. But a good half of our implementations - probably more - come from this source. And it's a communal kind of thing. It varies from individual to individual. But the Icon Project would look entirely different from the outside without this freely contributed help. There's very little we can do in return. Sometimes we can do little things. On occasion we can pay small consulting fees, or provide somebody with a piece of equipment or a compiler - something like that that they need. But we can't really employ them, and we couldn't begin to pay them for the market value of their work.

Not very many people are willing to accept this kind of help. It's a mixed bag, and we can get in a lot of trouble along with it. Sometimes you get terrible bugs, for example. But we have been willing to do that, and I don't regret it, in retrospect, although there are times that I have regretted individual adventures. I would have to say, I think in a case like this, of software like this that is in the public domain, contrary to the way that most people are dealing with software these days, you develop a certain mystique associated with it - esprit de corps - why people feel a certain way about software distribution. I think that has gone a long way to providing the energy and motivation for the help we have gotten.

But if graduate students are hard to manage, somebody out there freely donating their time, which you have no holds on at all, and they're doing it in their spare time while they have got a regular job, it's really hard to get performance sometimes, although we have been pretty successful. But that has been a very encouraging thing. As a matter of fact, it's that kind of thing that provides the encouragement that keeps these things going sometimes, because

sometimes there's a large amount of clerical work involved with something like this - an enormous amount of clerical work, and high-quality clerical work. It takes a lot of talent, ability, and experience to do some of this, but it is still clerical work. In many occasions the freely contributed work that other people have done has provided the compensation for doing all that clerical work. People just pop up. It's one of the aspects, of course, of our better electronic communication system now - that with electronic communication and the ability to send informal communications in addition to programs back and forth easily and inexpensively, that kind of communication facility is essential to, oh, certainly the last ten years of Icon. Lots of things have happened that wouldn't even have happened if you had to send letters and tapes back and forth. So that's a very evident change that I have witnessed myself. It's an example of the value for that communication to technology transfer.

CARGO: It sounds like you're talking specifically about Internet mail and FTPs for people to get hold of the service, to some extent.

RALPH GRISWOLD: Well, all I am saying is it makes it very much easier. It makes it easier for them to communicate with us and express an interest. It makes it easier for them to get technical advice; it makes it easier for us to give the technical advice. It makes it easier for them in many cases to get the material back and forth, although the actual transfer of large files is not the real issue. The real issue is ready communication. You know, we don't reach everybody this way, but it's a great facilitator, because if you think of somebody doing an implementation of Icon as a labor of love in the evenings after they're working in a professional job, it makes a lot of difference whether things are easy or hard. And it's a little bit of difference that can make a difference between success and failure. And over the period of years when we have had this kind of thing going on for a long time in the current situation it's very clear that the facility of communication has been instrumental, I think, in some of these things that otherwise just might have been just a little too hard to do.

But a lot of that communication, of course, is underground communication. People use electronic mail for things that their employers have no idea they're using it for. Lots of times employers might not really appreciate the fact that people are spending their resources and their time and their computer to work on little corners of Icon when they're supposed to be doing a database or something, but that's an entirely different topic, and only an edge or something

much bigger than that. In many cases, they benefit, of course. They may not know they going to benefit, but they frequently benefit. The employers benefit by running software that's then used. We have seen that in a number of commercial organizations including Microsoft.

CARGO: Going back a bit to the degrees that have been granted to the people working on Icon or the Icon Project, where would you say that most of the students go after they have gotten their degrees and finished working on the Icon Project?

RALPH GRISWOLD: I think that my students are split about 50/50 between academic positions and commercial positions of one kind or another. The last time I looked it was pretty close to 50/50. The typical academic position is a faculty member in a research-oriented computer science department, although some have gone to teaching positions in academe. The typical commercial organization is a large software company like DEC or something like that; in some cases, smaller organizations. Most of them have gone on to professional computing jobs of some kind or another, which seems plausible if they have got the Ph.D. in computer science. Some have been successful professionally by conventional standards, and some haven't. But I haven't seen any particular trend. Most of them, I think evidently because of the nature of the kind of research they have done, to have been involved with my work, have a lot of pragmatic experience in software design and development. Some are more implementation-oriented than others, but they have all worked on large software systems and they have all had major responsibilities for software design development and creation and innovation. So you would expect them to go into positions that correspond to that part of computer science. I have never had a student that you would call a student theory of comp utation, so that's probably the reason that, say, 50% of my students go into industry in one form or another. If it were theoretical work, they would all be going into academic environments. But a surprising percentage of them, despite the strong pragmatic orientation of much of the work, have gone into academic situations. And none of them has set the world on fire yet in that sense, but many of them are successful professionally.

CARGO: Have they tended to take Icon with them; try to use it where they go?

RALPH GRISWOLD: They take Icon with them, and they try to use it where they go and sometimes they proselytize

for it, although not as extensively as one might imagine. I would suppose that of the students that I have had in the Icon Project over the period of years, probably half of them have made Icon a major part of their professional work in one way or another, or a noticeable part of their professional work. But that's also true of students that take courses - students that get master's degrees but never get involved in research. They show up sort of spontaneously and say, "Do you remember me?" I don't remember all the students, of course, I've ever had in classes. And they will say, "Well, I've gotten..." They are using Icon in a big way here - something like that.

MADGE GRISWOLD: Well, one of the big examples of that is Rick Fonorow, who was your student, and he came and went. And then all of a sudden he bloomed at... Was it AT&T, or Bell Labs or...?

RALPH GRISWOLD: It was at AT&T; not Bell Labs. It was in one of the parts of AT&T whose name changes every six months - communications, I guess; or information systems. He was an evangelistic sort of person, and he really believed in it. And he was very effective. He managed to get AT&T to use Icon as a prototyping tool in their big software systems, despite a company policy to use C. And they have been very happy with it, but you know, he's definitely evangelical and has had a major influence on fairly widespread use of Icon in industry. Most of my students are not that evangelical. I don't tend to produce disciples, but they come in sometimes, you know. He was a master's student.

CARGO: Have there been any other major proponents of Icon who have gone out there and tried to convert the world to the Icon point of view?

RALPH GRISWOLD: Well, not very many active promoters. There are certainly some proponents. My experience is that many of them are rather low-key. Jerry Nowlin, who is at AT&T, contracted to AT&T, has certainly used it extensively and encouraged other people to use it. And Kelvin Nilsen, who is at Iowa State, has certainly been promoting its use in their curriculum and things like that. I think probably part of the answer to your question is that sometimes I don't know about this until after something has happened. All of a sudden I will discover that something like this is going on. Programming languages tend to develop followings. There are jokes about, you know, LISP religion and especially APL. And how a kind of a following of a language develops depends to a certain extent, in

fact to a large extent, on the nature of the language. For example, I don't think that PL/I produces a very strong adherence on the basis of emotional attachment. Icon does not produce the kind of fanatic dedication that, say, APL does, but it does something of that nature. So for my part I try to discourage that, because I don't think that, you know, zealous evangelicalism with respect to a programming language for emotional reasons is justified. So I tend to try to discourage that kind of thing. I don't strongly discourage, but I certainly don't encourage it. I don't try to build a cult of Icon. So it's more of an underground movement, I guess, you might say. I don't know. It's very hard for me to tell. And of course, people ask us how many people are using Icon. Who is using it? We have no way of knowing those kinds of things. We know how many copies we have sent out, but we don't know what happens to them.

CARGO: You know who is receiving the Newsletter.

RALPH GRISWOLD: We know who receives the Newsletter at any one time. Right now there's about 4800 people on our mailing list. But we don't know. It's free, and people take things that are free. Okay.

MADGE GRISWOLD: Well, also, it's fairly copyable. For example, we know there are many people inside IBM who have copied it off a bulletin board inside IBM. But only one copy has been registered, so within the one copy, a lot of people have it. You don't know how many people have it.

RALPH GRISWOLD: Well, it's IBM's official policy that their employees are not supposed to do that kind of thing.

MADGE GRISWOLD: They're not supposed to do that sort of thing, so nobody is going to tell you.

RALPH GRISWOLD: No, you can't send a package, a program, to an IBM employee with any assurance that they're going to get it, because IBM is concerned about Trojan horses of a commercial variety, that somebody will send their software into the company and then later on claim that IBM used it in their products and sue them. So they're very legally conscious about this. So these people will not even register their Icon copies or subscribe to the Newsletter. I don't know; it's hard to say. We know how many people are on the Icon Newsletter mailing list. We know who they

are, and we know their geographic and demographic distributions, and to a certain extent their business orientations, or what kind of organizations they work for, by inference. The only thing I can say is the last time we sent out a questionnaire asking people to comment on things we got back a quite high percentage and with lots of detailed answers. So people don't do that unless they're interested. But I am sure we have a fairly high percentage of people who get the Newsletter who have a very marginal interest. I have no way of knowing. The way to find out is to make them return something to stay on the subscription list, but then you lose a lot of people who would like to be on the subscription list but aren't very good at keeping track of their business. I don't know.

CARGO: What do you think are the main obstacles to the wide-spread use of Icon?

RALPH GRISWOLD: There are technical obstacles and there are pragmatic commercial, social, or political obstacles. Let me address the latter group. The situation now with Icon is different from what it was when SNOBOL4 was first distributed from Bell Laboratories in the early 1960s. There weren't any software companies. There wasn't any commercial software then. Now there are a lot of commercial companies. There's a lot of software; there's a lot of advertising; there's a lot of hype. There's a big industry. There's a lot of magazines about software, and everybody is aware of it. It's very difficult now to get anybody to pay attention to you if you are not commercial and have a big advertising budget and you're competing with a lot of organizations. They're selling products that are really bombarding the potential market.

So it's much more difficult now for public domain software to get treated seriously and to be used. This lack of commercial support means it doesn't have any substance so far as a commercial organization is concerned. There's no support, no guarantees, nobody to sue. Sometimes these things are ridiculous; they don't mean anything anyway, because all that's there is an advertisement. The company's gone in a year anyway. But the appearance is different, so it's very hard to get taken seriously if you don't have a commercial organization behind you, or the government behind you, or in a slightly different matter, a computer vendor behind you, if you are not an official language of the computer, or if you are not being sold by a software company, or not being mandated by the government. You are relying basically on people that have an interest that isn't interdicted by any of these things. That sort of automatically excludes most commercial organizations, so it's mostly individuals. And that, I think, is

one thing. The other one is just plain getting the information around. There's a lot more people in computing now, and short of doing something spectacular like getting the *National Enquirer* or something, it's hard to get pretty well known. There are lots of other things around that are not of comparable quality or importance but sort of look the same on the bulletin board.

There are some technical aspects. One is the speed of the implementation, the fact that it is not tailored to run for a particular platform. Somebody running Windows or OS/2 wants an application that fits that environment. We don't have the capacity to support that; it's probably not appropriate to try to. And there is also a trend away from programming toward so-called fourth, fifth, or sixth or seventh-generation languages, depending on who you talk to where the people are using computers these days; they are using applications; they're not programming. So in lots of environments people don't want a programming language anymore; they want something that will solve their problems. That may be an illusion, but from a sales point of view, Icon's technical characteristics being an imperative programming language and, you know, not cosmetically looking like, say, Prolog, has a harder time getting accepted, especially by non-technical people. Lots of things, but I think those are the main ones.

MADGE GRISWOLD: I would concur that I see a lot of people buying things because they see large ads. That's a very simplistic explanation, but it's true. Celebrity value being widely recognized, or seen in a magazine is a convincing point in our society right now. There are some very big advertising budgets behind some software that's causing it to be sold.

RALPH GRISWOLD: There's also a trend toward faddism, which is promoted by the popular press in computing: object-oriented programming, fourth-generation language, labels that are attached to things. And the user interest tends to go with these things, I mean, whether it's C++ right now or something else. We're not trying to hype Icon in any respect like this. We're not advertising it at all, but it's not ever going to look like it fits into one of these things that is all of a sudden cosmically important. People will ask us, "Is it in AI language?" You know, those kind of things, because that's where their focus is.

MADGE GRISWOLD: Or, "Is a fourth-generation language?"

RALPH GRISWOLD: Yes, or, "Is it object-oriented?" All these kinds of things. And those really are not probably the important questions, but that's where a large part of the naive market is driven by this. It's very evident, if you watch the magazines or the popular books that are published on computing. Icon is never going to fall into one of those kind of categories. So it's always going to suffer a disadvantage of not having a popular label currently attached to it.

CARGO: In fact, it has popular confusion attached to it because of the name.

RALPH GRISWOLD: That doesn't help; I am not sure it hurts as much as we thought it might. But occasionally we get somebody who thinks they're getting something else. But it doesn't help, certainly.

CARGO: How much have you seen Icon being used in academic environments, either as a language for teaching people about programming, or any other way?

RALPH GRISWOLD: We see it used in academic environments quite a bit. There are two or three sides to this concerning things that go on in academic environments. There's the instructional side of it. Icon is frequently taught as one of the languages in a comparative programming language course where students are being introduced to different kinds of concepts and programming languages. It has displaced SNOBOL4 there, which has been one of the languages that's most frequently taught. They'll be taught in a course with Prolog, APL, Icon, something like this. We know what the adoption list is from the publication of the book, but that really doesn't tell the whole story. It's not taught as much as it might be because of the lack of people who are competent to teach it. We've even put it on our own curriculum. We have such a course and we have an instructor who teaches the course - one of our own students originally. Icon is in that course at his request, but he doesn't teach it well. He teaches it as C with strings. He misses the point. And we don't have a large enough number of people who understand Icon well enough to teach it. So if you don't know how to teach something, you tend not to put it in the course, especially when it's optional. But it still is used quite a bit.

The other sides are in the routine programming that goes on in academic departments. I can't remember whether your question was directed toward computer science departments or not, but let me assume that it was. It's used quite commonly for utility programming and UNIX environments. That's very common, in VMS environments. It's pretty well-known in those environments. And it's used as a tool in research in many cases - as a prototyping tool, as a software design tool, or something like that. And I would say probably a third of the people we know that are actively using Icon are in academic environments, using it in one capacity or another. Our own laboratory staff uses it as a tool in systems programming of various kinds. But that's most noticeable in UNIX environments. We have not, incidentally, penetrated the IBM 370 mainframe environment very far. We haven't reached those people.

MADGE GRISWOLD: Well, we haven't had an implementation for them for very long actually.

RALPH GRISWOLD: Well, we have had implementation for a while now and we haven't reached the market, as it were. I don't mean the commercial market, but the user market.

TAPE 2/SIDE 2

CARGO: What other faculty at The University of Arizona have contributed support to Icon or the Icon Project?

RALPH GRISWOLD: Not very many. The only one really directly would be David Hanson. After he came back and got his degree he was there until a few years ago. For the most part my collaborations have been with students, not with faculty. We have a fairly small faculty - 10 or 12 depending on how you count them. They all have their own research interests. So in recent years I haven't had much collaboration with other faculty members. In the past I have had collaboration - pre-Icon, I guess. But right now my work is sort of an island in the department, and that's an island with one person sitting on it.

CARGO: What sort of a relationship would you say that you had with Tim Budd?

RALPH GRISWOLD: Well, let's see. Tim joined our faculty after he got his degree at Yale. I guess he got it at Yale; I

can't remember where he actually got it. And he was on our faculty for several years. He was one of the people we've had on faculty who was interested in programming language design. We never actually collaborated on anything, although we talked a lot about different things - common interests. I think he modelled (as junior faculty tend to do) his implementation of his version of Smalltalk - A Little Smalltalk - after what we had done, and modelled his distribution of it after that. As a matter of fact, there are several faculty members with other software projects who sort of imitated the kind of thing we have tried to do. I have never been directly involved with any of those. They come and ask for advice, or want to know what's going to happen to them. When I tell them, they usually decide they don't want to do it. Junior faculty really can't afford the kind of thing we're doing at Icon. It's probably professionally deadly if you are trying to get tenure to put that kind of effort into one long-range project, because your tenure at the university prior to a decision doesn't survive that long. Tim is another person who is interested in programming language design, and he is probably one of the ones most interested in that area, but we never had a close collaboration or working effort.

CARGO: You were talking earlier about the direction you saw that Icon was going. Can you elaborate a bit more about what you see that direction being?

RALPH GRISWOLD: There was a period of time in which Icon was an idea, and we were developing a programming language that was a complete useful whole, and implementing that. After we distributed that we had some new ideas and saw new directions, and we got some feedback and ideas from other people. We have gone through a series of versions that have added new things to the language or made it more practically useful to certain parts of the computing community. But we also in the process have developed a large clientele. Every incremental effort has become more manual labor and less intellectual labor. What we are supposed to be doing is research and these other things are cantilevered off of it for good reasons. But much of the work we have done in the last few years has not been research by normal measure. The research that we have done has been in other areas, and not so visible directly in Icon.

I did decide about three or four years ago that Icon has been widely enough accepted; there's enough interest in it that I would devote a certain amount of time to bringing it to a higher degree of professional maturity, providing more

support for it in terms of the Newsletter, documentation, the Icon program library; things that I never really did with SNOBOL4. I decided that Icon had had enough positive acceptance that that was justified and overall worthwhile. But I also realized that that was a finite amount of time; that I couldn't continue to do this in a research and academic environment. It was not the kind of thing we were supposed to be doing averaged over a long period of time. It was certainly not the thing that gets students degrees and produces research and those kinds of things. So in some sense we wrapped up a phase in the sense of producing the last version - Version 8, and the program library and all the implementations and the books and the documentation that goes with it. Despite what other people would like, it just isn't practical in this kind of an environment to continue to go on incrementally improving a product. If it was Borland with another version of their C compiler, that's one thing, because there's income to support it, and that's what you're supposed to be doing. And so, somebody has decided it's a vital product. But it's not reasonable in an academic environment.

So from my point of view, which is not a point of view that's appreciated by many people, we are wrapping something up and looking toward other things. For the first time since Icon started we are not working on a new version of the language; there is no intention of producing a new one. That will disappoint some people, but it's a fact of life. Where we do see it going is continuing to support it, continuing to improve things, possibly continuing to provide new implementations as a minor part of what we are doing. In the short range we are working on a programming environment that is more research-oriented - not specifically intended for distribution - certainly not over a wide community. And an independent issue, and that is a highly efficient implementation of Icon and an Icon compiler. I don't know at this point where resources are going to come to support the distribution of that piece of software or to maintain it. That's one of our concerns at the moment.

Basically we are finishing up something. Things stop changing in various ways. They stop changing as you lose interest in them (or other people lose interest and nothing happens), or they stop changing because you conspicuously decided that they are going to stop changing. I wouldn't want to categorically say that there's never going to be a major change in Icon, but we are not working on one, not planning one, and we don't have the resources to do one at the moment. So I view the development as sort of winding down in that area.

We are interested in implementation issues, and in issues related to how people think and program in Icon and similar languages. We are moving more toward that area, especially in terms of visualization tools, both for the performance of the behavior of the language, and the performance of the behavior of the implementation. I guess another way of putting it is that our interests or my research interests are moving more toward things that have spun off of Icon.

CARGO: Have you been working with the people with the object-oriented layers on top of Icon, or has that been pretty much independent?

RALPH GRISWOLD: Clint Joffery did an object-oriented, at least several object-oriented, versions of Icon. They are all pasted on top of Icon as additional features on top of it. I have never been as enthusiastic about object-oriented programming as some people have been. And I certainly don't think that it deserves all the hype it has gotten in the popular press. I think there are some very important, good things there. If there were an interest in this area I would like to see a language designed from the ground up that had object-oriented features - possibly a combination of some of the features that we perceive as being the most significant and important features of Icon - not as just pasting in a layer on top of Icon as another level. Clint wouldn't agree with me probably. He feels very strongly about his object-oriented version of Icon. My interest in doing this is not very high, I guess.

Let me make a comment here on language design. I have made it a professional policy to work orthogonally to the mainstream of computing - and deliberately to a certain extent; partly by accident, because we started out being totally ignorant. We weren't any computer scientists, really, when we started out. Partly because I felt that the opportunities for doing significantly interesting and different things for me were more in trying to do work in areas that other people weren't working in rather than to try to jump on the bandwagon. I have a personal philosophy of not trying to synthesize everything that's in existence that other people thought up into something else. I am not a synthesizer. I am not sure that I would find working on object-oriented features with respect to Icon very interesting. They might be for somebody else, but it wouldn't be for me. In order to do research you really have to be interested; you have to be passionate, and it's not just a routine kind of thing. So if I were going to do something it would probably be something bizarre, not something like an object-oriented version of Icon.

CARGO: What do you think are the major open issues to finishing off Icon as you see it? There still is a section of the Icon program library that needs to be issued, as I recall.

RALPH GRISWOLD: [laugh] Well, the Icon program library support... there is a section. There is a large database of contributed programs. It's a very painful and difficult thing to do, because people contribute programs in various states of repair and disrepair. But in order for us to be willing to even to distribute them as, you know, somebody else's contributions, we have to feel that they are reasonable programs, that they actually work. People submit programs that don't compile; the documentation is lacking. It's an enormous amount of work of a very unoriginal and unrewarding nature to put any part of the Icon program library together. I have got megabytes of contributed material, but it's very, very difficult to get that together and distribute it, and it's very unrewarding from a personal point of view. So, yes, we'll probably get some of that out. The material that we are working on for the *Icon Analyst* idea is a lot more important because that will be more highly technical material concerning programming in Icon and things related to Icon than is available in any other way. I think we will provide a much more important resource to people who are really interested in Icon, and so actually we are concentrating a lot of effort in that area. We, meaning Madge and myself. In other words, I guess I think documentation is probably one of the most important things - especially about the really interesting, in some cases, esoteric or very different parts of Icon.

I would probably be pleased to see implementations that fit naturally into the environments for some other platforms. I am not an OS/2 fan, but I can well imagine a Windows version that would be appealing to people. I am not sure we would want to or could undertake doing that. There are a few other implementations that I think would be useful to have, but perhaps not very many. And perhaps tools relevant to programming - understanding Icon - that would not necessarily take the form of an integrated programming environment, but support facilities; things like that. Then other things come up, as will come up in the workshop, I think. For certain kinds of applications, a macro preprocessor is almost an essential. And we do not have one of those - not a very intellectual endeavor though. Not something one builds a degree program around for some graduate student.

That becomes a constant problem, because as Icon matures, more and more of the things relating to it that seem appropriate to do have less and less to do with original research contributions, and there's no support for them

basically. So it becomes an asymptotic problem of just how far do you want to push to get that little bit of extra out. Sometimes we get surprises. For example, the Icon compiler could be quite important, but it's still problematical. And I don't know; that's something that's very uncertain at this point.

I guess I would have to say (and I am going back to something about the question you asked earlier about the impediments to our use of Icon). Performance of the implementation is one. I personally am more interested in implementation than language design at this point in my life. And I am more interested in that kind of thing than, for example, object-oriented Icon or features for Icon. So I don't see a clear end. These things sort of stop abruptly, or they sort of dwindle, or stop moving. Nobody notices they stop moving, because nobody's looking, or something, but I really don't know. I really couldn't give you a good feeling for where I think things would be five years from now. I don't know, and I don't have a clear idea of where I would like them to be, except maybe wider acceptance, wider use.

CARGO: Turning that around, where did you think things were going five years ago?

RALPH GRISWOLD: About where they are now. There were some times in the development of the Icon project, if you choose to take the Icon project as entity, when there were considerable uncertainties - for example, whether it was good enough to go, to carry on. That has to be measured by user acceptance; it can't be measured by our own ego, you know, self-esteem about it. I wasn't at all sure when we started that it was going to work in that respect. So initially I would not have anticipated where we are now. But in the last few years things are about where I expected they would be, and about where I hoped they would be, and maybe got there a little bit faster than I thought. For example, I thought it was going to take a lot longer to get the implementations of Version 8 done than it actually did. But we are still looking at like, a two-year period, and especially right now since we don't have anything in the works except the compiler and the Icon programming environment. Anything significant that might happen five years from now hasn't been thought of yet, so I can't project that. But for the last few years we have been working on a program, or at least in my own head, working on a program that puts us about where we are, about where I expected to be.

CARGO: Would you say that the standardization of C, both as an ANSI standard and sort of a de facto standard for systems implementation languages, has facilitated the work you have been doing with Icon?

RALPH GRISWOLD: That's kind of hard to say. It's made a lot of work for us, because Icon was written long before there was even a proposed standard and, of course, couldn't conform to it. I think that probably the answer to your question is yes, in the sense that the ANSI C standard effort has caused compiler writers and vendors to improve their products in a coherent fashion, so that the C compilers we are now dealing with are much better and more robust than they were a few years ago. I think the ANSI C effort contributed to that, although the market pressures were anyway - at least kept them from getting more diverse.

So you might hypothesize if there had not been an effort to standardize C it would now be much more divergent than it is, and that would have been a disadvantage to us. We still support an awful lot of C compilers and an awful lot of idiosyncracies, and some ancient ones that are abominations. And we have learned a lot about C, about portability from a pragmatic point of view, and about how to build portable software. The trouble is that producing a C standard does not convert all the existing compilers to that standard, and the main one that's used by most people, PCC, is a long way from that, but the UNIX environment intends to use that. It will probably for quite a while.

It would have been a big help to us if we had started later, or they had started earlier, or something. To make a comment on that, we first implemented Icon, when we implemented it in FORTRAN, largely because Dave Hanson wanted to prove it was possible to do something like that. Granted, we used RATFOR as a preprocessor to write in. That was a very portable implementation, but there weren't good FORTRAN compilers for many of the new machines coming out, especially PCs and things like that. We had to make a decision about implementation language at some point around Version 3 of Icon. We chose C, not out of many possibilities, and not with any expectation that things were going to be the way they are now, because at that time C was only available on UNIX systems, and we had no way of knowing whether it would be available on personal computers, or how widely available it is. We have benefitted a great deal from the fact that it is ubiquitous and widely available at the high quality C compilers for almost all computers, but that was sheer luck. We had no way of anticipating that.

MADGE GRISWOLD: That's true. I distinctly remember standing in the departmental computer center when it was first established, and they were making a decision, or you were making a decision about what to implement the next version of Icon in. And I think we were going from two to five, or... I don't know, what happened to three and four?

RALPH GRISWOLD: Well, three and three was there.

MADGE GRISWOLD: And saying, "Well, C is a good choice and that's what we have available for this machine, but it's not going to ever be widely available, or it doesn't look like it's going to be widely available for a number of machines." And it was simply a wonderful serendipity that it was accepted and made available for so many different configurations later.

RALPH GRISWOLD: In fact, when we went to Version 3 of Icon, an implementation away from FORTRAN, we felt we were abandoning portability, and we thought we were probably implementing Icon only for our own use, or possibly a small number of UNIX systems. We thought we were giving up the portability at that point. The fact that we didn't was just because of the development of C. So we did not anticipate that. We might have hoped for it, but we certainly didn't have any expectations.

MADGE GRISWOLD: There was no way of knowing at the time that that would happen.

RALPH GRISWOLD: Somebody might have, but we didn't see it.

CARGO: As I recall, the portability mechanism for SNOBOL was considerably different.

RALPH GRISWOLD: Yes, SNOBOL was written in a nonexistent virtual machine language all of its own, sort of like an assembly language. And you implemented SNOBOL4 by writing macros to these instructions on the virtual machine and then assembling it. We did think about writing our own implementation language for implementing Icon. However, we had had the experience of doing that for SL5 and we didn't like it very much, because it meant that we had two languages and somebody had to port both of them. It might be academically interesting, but rolling your

own there didn't seem worthwhile.

SNOBOL4 was not very well structured; it was written at the time when the only viable implementation language was assembly language, and it's a very different kind of thing. It is, in fact, quite portable in certain kinds of computer architecture, although it doesn't fit the current ones very well. But no, this is an entirely different thing. That was much more of an academic exercise, and an original exercise in implementation technique. Icon is just simply written in a programming language like anybody else would write one. It's not very unusual in that respect. But really, the C implementation of Icon was done because we wanted to be able to run Icon on our own computer. For the first time we had our own departmental computer; it ran UNIX. We wanted to be able to run it on our machine instead of the computer center's machine. What we had in mind with that implementation was to forget putting all this effort into portability. We were just going to write a version that we could use, that we could have access to to give us better resources.

CARGO: That was a PDP-11?

RALPH GRISWOLD: PDP-11/70, yes. A lot of the structure of the implementation was significantly affected by that. For one thing, we had to worry about space. With PDP-11/70, even with separate I and D spaces you only have 128 K of addressable memory, and we were really worried whether it would fit. If we had been starting on a machine that's a typical platform now, our concerns would be quite different. There are still some things there that show that - those origins that affect the current implementation. The implementation has changed a very great deal, but it still consists of a succession of modifications to the PDP-11. As a matter of fact, a lot of the hard part of the problem is that we took an implementation that was very UNIX-oriented and non-portable and had to sort of mash it into a portable implementation incrementally over many years, over many C compilers. We could do a lot better job starting over now.

CARGO: But that's not research.

RALPH GRISWOLD: That's not research, and it's probably not cost-effective in the sense that you can better afford

to distribute all of that non-research work incrementally over a large period of time than you can to take out a year or two and start over. It's just not feasible to spend your time in large pieces like that. But you're right; it's not research. And no, I don't love C. [laugh]

CARGO: Have you seen very many comparisons between Icon and AWK, or Icon and PERL? I am assuming you know what PERL is.

RALPH GRISWOLD: Only slightly. I know AWK. There have been comparisons with both. I don't feel competent to discuss PERL. The interesting thing about AWK is that AWK really is based largely on features of SNOBOL4, and they're generally not recognized. They're not, well, not recognized, but they also are not acknowledged until publications came out, in places where referees required acknowledgement. I view those as very different kinds of things. There are lots of other things around that are similar that are probably less known. I think that the comparisons tend to look at things in a certain way. They tend to say, you know, "Suppose I was doing this on AWK, you know, how would I do it on Icon? What is the comparison in performance and program complexity and length and ease?" When things aren't comparable at all people don't compare them. The impression that's left is that they're comparable but these are the only differences. You don't write for prototypes really in AWK or complicated interactive systems and things like that. It's the best thing for writing utilities. I get kind of bored by programming language comparisons. But most people see only a small portion of a programming language in my experience. They just look at it. They see a certain kind of thing. They don't see what other people see in it. Icon is a big language compared to many of these other languages. AWK did not influence Icon. AWK was influenced by Icon's predecessors. That we know. But I don't find them very comparable languages - certainly in small group of tasks for which they are comparable, and certainly in most cases AWK looks better than Icon. But of course, the other thing, from my view, is that things that interest me are the intellectually interesting aspects of Icon, not just entirely how good a programming language it is, or is it good for this application, or is it fast or slow for this application? Some of the ideas, such as generators and goal-directed evaluation, the concept of failure, which dates back to SNOBOL. Tables date back to SNOBOL. AWK acquired them in a limited form. Those are conceptual ideas that had been viewed as being important from a research point of view. The idea of expression evaluation, handling exceptional conditions through not producing results as opposed to dealing with Boolean values, is something that some people

think is an important, significant difference in contribution. But it's probably just an annoyance to most people programming in Icon. So it makes a lot of difference whether I am talking to academics, or whether I am talking to Icon programmers, or programmers in general, how they feel about these things. But we have been motivated, for the most part initially, and entirely funded by programming language design, and in some cases, implementation techniques. But nobody has ever funded this to design a complete programming language. You know, you would make a new product. That's something you sort of cobbled out of the pieces, I guess out of the perception that it wouldn't be funded - such a proposal, which I think is probably correct.

CARGO: Do you have any idea, at this point, how long you can continue to work with Icon? I mean, you talked about five years, but I don't know if you personally have got an idea of what you would do after that, whether you would ever suggest that Icon go before a standards body for standardization.

RALPH GRISWOLD: [laugh] Well, that's come up with the IEEE actually. I don't think there is enough interest in it to have that be a threat. I don't see any reason why I should ever particularly not want to continue working with Icon, whether I want to be actively involved with new things associated with it or not, I don't know. From my own point of view, there's a great number of things associated with programming language with semantic characteristics that have not been explored very fully, which I think are interesting and potentially valuable. It's just that I don't think that incrementally changing the language is something that I can do or want to do. Unless something that's clearly evidently better comes along, I can't say that I would ever refuse to be involved with it. You know, as a matter of fact, the last time Icon came up for standardization I was willing to be on a committee, and it turned out that IEEE wasn't all that interested in it, I guess. I don't know; I'm not in the politics of that, but I have no desire to get rid of my involvement.

CARGO: I think that's it.

RALPH GRISWOLD: Okay.

END OF INTERVIEW