An Interview with

GREGG M. TOWNSEND

OH 204

Conducted by David S. Cargo

on

26 July 1990

Flagstaff, AZ

Gregg M. Townsend Interview
26 July 1990

Abstract

Townsend describes how he became involved with the Icon project under the direction of Ralph Griswold as a graduate student. He discusses the work environment and staffing of the project and concludes the interview with a brief comparison of the C and Icon programming languages.

GREGG M. TOWNSEND INTERVIEW

DATE: 26 July 1990 INTERVIEWER: David S. Cargo

LOCATION: Flagstaff, AZ

CARGO: Gregg, I am going to start by asking some questions about your background and what you did before you got involved with Icon to help set the context. So could you talk about where you went to school, when you went to school, what kind of work you did before you got involved with Icon?

TOWNSEND: I actually first got hooked on computers when a guy in my high school printed out a Snoopy calendar and gave it to me. The next year I took the data processing class, as they called it - learned FORTRAN and a few things like that, and really got hooked. So I went to the University [of Arizona] I suppose because that seemed like the easiest default thing to do, living in Tucson, and signed up for a degree in systems engineering because there was no undergrad major in computer science, but the systems engineering department did teach a couple of computer classes. I got my bachelor's degree in 1974, and along the way I had been working as a system programmer at the Computer Center there on CDC equipment - CDC 6400. I went off to work for Control Data in Sunnyvale for three years and worked there on the loader project and then in the system design group. I left Control Data in 1977 and came back to the University of Arizona to be a systems programmer there again, and did some more work on 6400, which then turned into a Cyber 175. I also did a little bit of work on a DEC 10, which was there by then. Along the way I started taking some computer science courses because they looked interesting. After I had taken several of those I realized that I didn't have to take too many more uninteresting ones in order to get a master's degree. So I followed that up and finished that up. I got the master's degree in 1984 - master's in computer science. And at the same time a job opened up in the computer science department for a research programmer, and I applied for it. As it happened, Bill Mitchell got that one, and then I got another one a month later or so, and went to work for the computer science department.

My first connection with Icon actually began a couple years earlier than that when I took a course mostly from Ralph, although Dave Hanson taught a little bit of it, on string and list processing. I learned Icon in that class and then a year or two later took Ralph's course on Icon internals. That was the particular class that implemented the set data type in Icon. One of those implementations eventually made it into the released distribution of Icon. Since 1984, I have been working for the computer science department on a lot of projects. Several of those have involved Icon. Several of them have involved other things - the SR language probably being the biggest of those. I also did some early work on Chris Fraser's peephole optimizer and then several projects involving computer graphics.

CARGO: When you were getting the job in the computer science department, were you aware at the time that you might be working with Icon?

TOWNSEND: Yes, it certainly seemed like a good possibility. I don't remember thinking about Icon in particular, just in general. I guess I was feeling a bit burned out after all those years of systems programming and wanted to do something a little bit different. And the job in computer science involved more variety, moving around, working on different projects. And it certainly would be different from systems programming a CDC machine in COMPASS assembly language.

CARGO: So you are primarily involved with the Icon Project on a project by project basis rather than continuing.

TOWNSEND: That's right.

CARGO: Are you one of several resources they have to contend for in terms of programming help?

TOWNSEND: There are now two people doing the same sort of stuff that I am doing. Bill and I originally both started out with the same sort of job, but just the way our interests ran, it kind of ended up with I was doing more research programming, and Bill more or less evolved into doing pretty much solid system management on the VAX and Sun systems. Then I was the only research programmer for maybe a year or two after Bill left to go to Sunquest. A year ago we hired Sandy Miller, and she is also doing some research projects. She did some of the work in moving the large integer arithmetic into Icon -- porting in Chris Smith's core. Some of the projects I have worked on... I have worked on the Memmon project in several stages. The ones I can remember - the first stage was a memory monitor program that drove the AED 1024 graphics display directly. Then we got a Raster Tech One/80 and wanted to drive it. That's where, I think, we split out the Memmon data into writing a separate file and piping into a separate program. Later on I added color Postscript, and miscellaneous mods along the way as new data types needed small mods to that. Soon I am going to be working on an X-Windows driver for Memmon. Let's see, some of the other stuff I have done, I did a dynamic hashing implementation for Icon. This was based on some work that Bill Griswold did at Washington. I took a very good look at his code and used the ideas a lot. Most of the code was actually rewritten pretty much from scratch. At the moment I can't pull any other projects out. I know there were a couple, at least.

CARGO: That's all right. From the sound of this, you were involved at least with implementing major aspects or significant aspects of the Icon runtime system. Were you involved at all with any of the stuff with language definition?

TOWNSEND: Not to any heavy extent. By being there and being in contact with Ralph, I certainly put in some of my ideas and some of my votes, and some of the little things have been reflected in the language - but nothing major.

CARGO: Could you explain more the decision-making process? You were referring to your vote. That's one of the interesting things about how the Icon Project works - how decisions are made, and so I am curious about that.

TOWNSEND: Okay. We don't really vote on things, or at least if we take a vote that's not the final answer. Ralph is the final arbiter. What usually happens is that we will sit around and talk about the various possibilities and eventually a consensus will emerge. Ralph and I actually seem to see a lot of things pretty much the same way, so I am usually pretty happy with the final results of these. Sometimes we don't and, like I say, Ralph makes the final decisions, of course. The only thing I really remember disagreeing with was the error recovery stuff. That's where you can turn any sort of a run-time error into failure. And I am still not convinced that it was and is justified on the whole, but on the other hand very shortly after it was put in I found some real good uses for it.

CARGO: Have you ever been a situation where there is some proposed change under discussion, and rather than waiting for a decision to be made, you will prototype it and then show the results to see what the reaction is?

TOWNSEND: That certainly sounds plausible, but I can't recall a particular instance of having done that. But, yes, that's the sort of stuff that could certainly go on.

CARGO: I was just wondering if that had happened to you personally.

TOWNSEND: Not on anything major enough that I would remember it, but that's not the sort of thing that would be so remarkable that I would necessarily remember it, because it's a very informal atmosphere. And of course, everybody has access to the source code and can do whatever they want with their private copies, and certainly that's where some things could come from.

CARGO: When you're going about making some changes to the system, do you folks use any configuration control?

TOWNSEND: No, we don't - nothing formal. That sounds surprising, but it has actually worked fairly well. What happens is that Ralph maintains the master copy of the source code, and what happens usually is that anyone else who is making a change will make a change on a private copy and then he will back those changes into the source code. Let's say it's me, for discussion, but this could be anyone. When I have all of my mods ready to go, I will tell Ralph and he will make a clean copy of the current source as of that time, and then I will back my mods into that copy. He will do his diffs and whatever else he does to check things out and to keep records, and then he will move the stuff in. And he won't actually have that much to do because there won't be any other outstanding mods at that point.

CARGO: When you say mods, do you mean actually scripts to make changes or simply replacement modules?

TOWNSEND: Generally replacement modules. I think, depending on the magnitude of the change, he will either replace a routine wholesale, or he will just edit in the two-line change, depending on the sort of modification it is.

CARGO: How do you guys go about testing once you have the changes backed in?

TOWNSEND: We certainly don't have any formal system for that. There's a standard set of Icon verification programs, and Ralph runs those, presumably after every set of major modifications. I think he has some additional tests beyond the standard ones that he also runs. All of us who don't want to be embarrassed, of course, run on all of those tests on our mods before we give them to Ralph to make sure we are not giving him any booby traps.

CARGO: Do you also develop tests that verify the particular feature that you have installed operates also?

TOWNSEND: Yes, I will do that. One part of the package of changes will be perhaps one or two new programs for the test suite or modifications to existing programs in a test suite.

CARGO: I have some questions that relate to scheduling your involvement with the Icon Project for implementations. How long do they have to wait for you to be available, and about how long are you working on their projects once you start, and are you working full-time?

TOWNSEND: As far as the waiting time, that interacts with what other projects are going on in the department. The department head is the one who schedules my time in response to requests from all the faculty. Usually, I guess I would say there is a one- to two-month delay before I get started on a particular Icon project. Ralph usually thinks fairly far in advance and tries to schedule these things, and often schedules other things to come to a point where it's a good point for me to jump in and get started on something, with no other big mods of his being in the middle at that point, for instance. As far as duration, I would guess anywhere from perhaps three to six weeks would be a typical project. Part of that is because these things are typically defined only to the level of one or two sentences until I get started, and then part of the process includes coming up with ideas and meeting and discussing them with Ralph, and refining to the point of exactly what it is that we are trying to do. And so even the estimates tend to be somewhat nebulous until I actually get started, and then, of course, lots of good ideas come up along the way, too.

CARGO: Who writes that one or two line description? Is that Ralph?

TOWNSEND: That's usually Ralph, yes. The way the process works, Ralph fills out a little form and gives it to the department head. And the reason for the form is just mainly to have a stack of papers somewhere so you can keep track of how many requests are waiting. And on that is a space for an estimate of how much of my time he is going to need. And where that usually comes from is first he comes and discusses the project with me and I come up with an estimate out of thin air and hope that it is going to be somewhat accurate and he puts that down on the form, and then it's there when it comes back to me.

CARGO: Now, I guess the question about whether you work on each particular thing full-time hadn't gotten answered yet.

TOWNSEND: Right. I work on it about as full-time as I can work full-time on anything. That is, there are typically on-going activities and occasional fires that need fighting. I am doing work for the SR language also, so every week I have a meeting with the Icon Project and a meeting with the SR Project, regardless of what my current so-called full-time project is. Also, administratively I am part of the laboratory support group, and there is a meeting for that every week. To that you can add a small amount of time for answering people's questions and whatever else. But discounting that, as far as serious programming projects, yes, I am working full-time.

CARGO: When you have these meetings with the projects, are you mainly hearing from them where they are, or are you trying to give your progress assessment to them, or is it both?

TOWNSEND: The Icon meetings are actually a fairly new thing, since Ralph all of a sudden picked up a large number of grad students to work on Icon, and I haven't been doing much on Icon myself since then, so I have mostly been sitting in and listening and throwing in my comments on what I thought about what people were doing. The SR meetings are typically pretty short. We just go around the table and everyone gives a progress report, and if I am working on something else, I mostly just say, "Nothing to report."

CARGO: Have you noticed a lot of fluctuation in the years you have been working for the laboratory staff and the amount of support that Ralph needs from you?

TOWNSEND: Well, sure. I guess it's more a question of availability than anything else. I think Ralph could use up about four full-time people if they were available. He has enough ideas that he could certainly keep many, many people busy. It's more a question of how much of my time can he get and how would he like to prioritize it. Of course, it makes me feel good to be in demand like that. There have been fluctuations, because there are good times and bad times to work on the Icon Project. A bad time would be, for instance, when the code for Version 8 has been frozen, and now the long mechanical process of producing implementations for all of these various platforms is under way. That's a bad time to add code to the Icon system, because then it will come out in some of those platforms and not others.

CARGO: Under those circumstances, do you find yourself trying to help make scheduling decisions about when they should be using you, or are you still pretty much just gotten on an "as available" basis?

TOWNSEND: It's pretty much on an "as available" basis. One of the long-term projects I am involved is the SR compiler. We are rewriting our compiler. This is going to be Version 2, which is really the third one since SR naught was the first one. And I guess my main input there is what is a good time to break away from that and do something else. And certainly I am consulted on that. It's actually, perhaps, a lot less formal than I have made it sound by these descriptions. All of this is just kind of a framework within which mostly everything is settled by informal discussion.

CARGO: Do you find yourself working with the graduate students and the other research programmer, either coaching or helping them get familiar with the Icon source in other capacities aside from actually doing implementation?

TOWNSEND: Sometimes. There's always the usual amount of going around and asking questions to who's the expert, and certainly I answer some questions, and certainly I ask some. I haven't done a lot of extensive coaching of anybody.

CARGO: Is that something that has been roughly the same in the past as it is now, or is it something that looks like there might be more of since it sounds like there's a recent influx of graduate students?

TOWNSEND: Well, I don't know about that. Generally the people who are working on the Icon system for Ralph have all gone through Ralph's Icon internals course, and in fact that includes the other research programmer. So all of them are fairly well-versed on the internals and don't need a lot of help in getting started.

CARGO: Have you thought about going back and getting a Ph.D. yourself?

TOWNSEND: Not very seriously. I suppose it would be a nice thing to hang on the wall, but other than that I am not sure what I would do with it and of course it is a lot of work. I enjoy the sort of work that I am doing. The few times that I have been closer to pure research I haven't enjoyed it as much as just sitting down and programming, and I don't particularly enjoy teaching, so I think I have found my niche, if you will.

CARGO: Do you expect to be continuing to work for the Icon Project off and on for the indefinite future? As long as Ralph continues to have an opportunity for you to do things?

TOWNSEND: I can certainly see that proceeding or continuing indefinitely. I don't know if two years from now it will be the Icon Project anymore, but certainly I will be doing projects for Ralph.

CARGO: Have there been other aspects of the Icon Project aside from implementing specific features that you have worked on?

TOWNSEND: Well, I was involved in several of the VMS ports - sometimes just in verifying that the system still worked and a couple of times in figuring out why not when it didn't. I think there were two VMS ports actually. I don't remember who did the first one off-hand. Chris Janton did the second one that I am aware of. He is a programmer with the Computer Center, and I worked kind of as a liaison on that project. At one point the memory region expansion code stopped working under VMS, and I went in and did some code there that basically involved using some VMS specific facilities to reserve a portion of the 32-bit address space that was out of everybody else's way for Icon's regions. And that's probably the biggest piece of code that I have done for VMS. The only tricky thing about that, or the nonobvious part of that, was it turned out that there was at that time, anyway, some cost at process termination depending on just how far into the address space you went for this private portion. So you couldn't just set it way far away from everything else, because that incurred an overhead. So there's a tradeoff in how far away from the standard data segment you want to place it.

Other than that, I did some work with the context switch and overflow code for one of the MIPS machine - I think it was the DEC 3100. I did a little bit of work helping test out the large integer code. I have done a few things like that are not particularly large and significant, so I probably can't remember them. Every once in a while I will get in and just do little stuff like that trying out the large integers just to see if I can calculate the world's largest prime, and things like that. I guess maybe it's kind of in conjunction with this; it's worth mentioning that my job is fairly loosely structured, so there's no problem when I take on these little side projects on my own initiative. And it's fairly often that I will be doing some little thing, like playing with the large integer package, or poking up a panel display of lights for the Icon Virtual Machine, or something like that. I have done some playing with that; it doesn't work yet. I have a display that flashes a real impressive display, but the data is all made up by the random number generator.

CARGO: Yes, I was going to ask if these non-implementation tasks were ones that you took under your own initiative or whether they were also ones that Ralph filled out a form and formally requested your presence.

TOWNSEND: No, the way it works is we only even approach the formal scheduling with the big projects. I am always doing little things, for Icon and for SR and for the lab and for anything else that comes by.

CARGO: What's the operational definition of "big?"

TOWNSEND: I guess my operational definition of "big" would be something that's going to last for more than a week, say.

CARGO: Can you tell me some more about the other people you have worked with in conjunction with supporting the Icon Project?

TOWNSEND: Well, let's see. I haven't really worked with them too much in supporting the Icon Project. That is, I can describe the people but I have not really worked with them closely in anything involving Icon. For instance, I was an office mate with Bill Mitchell for I think two years, but we never worked on an Icon Project together. He worked on a lot of projects during that time and I worked on some, but we didn't do anything jointly. Ralph would be the only one that I have really worked with on that basis. He gives me a pretty free hand. Usually we sit down and talk about what something should look like. He will have a general idea of what he wants and then I will come up with some specific proposals and he will tell me what is wrong with those, and we will go back and forth until we arrive at something. And, in fact, usually what happens is we will start off with some initial ideas, and then getting into it we will raise more questions and more ideas. We'll go back and forth and eventually end up with a finished project.

TAPE 1/SIDE 2

CARGO: It sounds like the schedule for the people who work for the lab is pretty much taking it in person-size chunks rather than having to have people working as a team. You mentioned you hadn't worked, for example, with Bill Mitchell on a particular project. Had you ever both been working on different pieces of some Icon implementation at the same, or near the same time?

TOWNSEND: I don't remember that happening. It might have, but I certainly don't remember it.

CARGO: What about with the most recent research programmer you mentioned?

TOWNSEND: Yes, that would be Sandy Miller. I did help her out a little bit with the large integer code, mostly by answering a few questions here and there. She, I think, prefers to dig in and really understand something herself rather than take the easy route of just asking a question every time it turns up like some people. So I didn't really have a lot of influence there on that one either.

CARGO: I was just trying to get a feel for whether Ralph sometimes had lab people working on two different things independently, simultaneously, as opposed to working on the same thing together, or never overlapping at all.

TOWNSEND: I don't think there would be any particular problem in having people working on two different projects simultaneously. In fact, if you go beyond the lab people to include his R.A.s, it probably has happened, because, as I said, everyone has their own private copy, and the only interaction comes at the point of installing the final code in the master source, and he coordinates that.

CARGO: Do you know if he has ever had the R.A.s work as teams on particular features, or if they also have been pretty much doing their own pieces of some change?

TOWNSEND: Well, of course, I am not the best person to be answering that, and the idea of having this many R.A.s at once is fairly novel to the Icon Project. But they're all working together on this general visualization problem now. And in particular, Beth and Nick are making heavy use of Clint's X-Windows interface. And there has been a lot of feedback there from "I need this feature" and "How do I do this?" and that sort of thing, and I think some code sharing, at least in the routines they have using, to drive the X-Windows package.

CARGO: You talked about an unusually large number of R.A.s recently. Have you noticed what is a more typical number of R.A.s in the years you have been involved?

TOWNSEND: I would say typical would be two or three, and now I think the number is something like five. If I had to pick one typical number over the years I have been watching I would say two, though.

CARGO: Has that pretty much been people in different phases of their degree programs? I realize you might not have intimate knowledge of this, but you certainly have been involved.

TOWNSEND: Well, I think more typically it has been two people working on almost completely unrelated projects, as distinct from the current situation where now I think everybody is looking at various aspects of the visualization question.

CARGO: Let me rephrase my question. Have you seen whether they tend to start in, you know, new R.A.s starting simultaneously, whether one has got a couple years of residence before another one starts and so they sort of have a junior and senior R.A.?

TOWNSEND: I guess I would say they have tended to be staggered. I am thinking mostly on the basis of when they have gotten out, because that is more obvious than when they have gotten interested and gotten started. But Ken Walker is getting close to finishing up. Janalee O'Bagy and Kelvin Nilsen were the two previous ones, and now I don't remember whether they left at the same time or not.

CARGO: Do you know anything about how Ralph decides to recruit people to work on the Icon Project as R.A.s?

TOWNSEND: No, I don't.

CARGO: That might be all the questions I have. Do you have anything you would like to add? Any high points of working with the Icon Project?

TOWNSEND: Well, I always seem to enjoy all these projects I do in connection with Icon. I mentioned that Ralph and I seem to have a lot of the same views on several things, so it's always fun to talk with someone who tends to agree with you a lot and who comes up with a lot of ideas that you can agree with. And, of course, Ralph is a very easy person to talk to. I have found him very easy to work for. I haven't had as much contact with the other people on the project, but so far I have liked all of them. I have enjoyed what contact there has been. There is a certain amount of pleasure in working on something like the Icon Project - seeing your code and seeing your ideas get incorporated into something that then gets spread all over the world. You see people using it and appreciating it; I enjoy that. And that's all that comes to mind.

CARGO: I guess backing up a ways, asking about the relationship of the Icon Project with Ralph Griswold, it's hard to ask a question that doesn't appear to be a leading question, but what do you think the relationship is between Ralph's longevity at the U of A and perhaps steady hand at the helm of the Icon Project? Do you think that has been critical to the direction the project has taken?

TOWNSEND: I would say certainly it has. You could consider two alternatives. You could consider Ralph going elsewhere and taking the Icon Project with him, or you could consider Ralph going elsewhere and the Icon Project staying here, staying at Arizona. I can't really imagine that having any success at all without some driving force, and there is really no one on the faculty who would pick up the torch, if you would. Now if Ralph was to go elsewhere, the question of timing gets into it. If Ralph was to go elsewhere right now, probably Icon would go into a maintenance mode of some sort. If Ralph had gone somewhere else, say five years ago and taken Icon with him, quite possibly a similar set of interesting things would still have happened with Icon. They undoubtedly would have been different with different influences and different people, but certainly he has been the central driving force.

CARGO: I have to ask a few more questions pertaining specifically to how you wind up using Icon on a day-to-day basis. That's not something I have been able to ask the other people, and you certainly, in your multifaceted role at the U of A, probably have a lot of opportunity to use it for a lot of different things. Do you find yourself using those opportunities?

TOWNSEND: I certainly use it a lot when I have the chance; that is, when I need a one-shot program. There's a hierarchy of complexity. If I am doing something really simple that I can do with a three-line shell script, I will do it that way, and if I can do it with a five-line AWK program I am still more likely to do it in AWK than I am in Icon at that level. If it gets to where it would be, say, a twenty-line AWK program, that's where I see Icon as being competitive, and I am most likely to write it in Icon at that point. I guess that would carry forward, carry all the way up into things up to hundreds of lines. If it gets into a really large project, something involving thousands of lines, that's usually something that has lots of other constraints imposed on it, such as, for instance, being part of the SR compiler, which constrains it to being written in C, or having to interface with some other existing package. Of course, all of Icon itself is written in C. And so whenever I am doing anything that's part of the Icon Project I am writing it in C. I like to use Icon for small programs. I don't really want to qualify it that way. It does tend to be my language of choice more often than not, but of necessity I remain fluent in both C and Icon, and so for me that's not really a constraint as to which one I choose.

CARGO: What would you say the major strengths are of Icon versus C? Or what features of Icon do you use that push you in that direction instead of using C?

TOWNSEND: The things I really appreciate are the data structures - lists, associative tables, and things like that. The string facilities would actually have to come second to that for the sorts of stuff I seem to end up doing. Beyond that is less of a superstructure you need to create before you are actually writing the meat of your program. The same sort of ideas as fast prototyping, only I tend not to do prototypes so much as just little one-shot programs, which then I save because I know they're not going to be one-shot programs.

CARGO: So the things you see that give Icon the advantage over C has to do with the data structuring and presumably not having to do your own memory management?

TOWNSEND: Yes, right. The data structures and the memory management, and the string facilities are useful certainly. The main reasons I would have for writing something in C instead of in Icon would be when I need access to systems facilities like being able to read directories or something like that that's not easily accomplished from Icon.

CARGO: Since you also program in AWK, maybe you have had a chance to ponder this, which is a very small technical point, but sort of comparison between the two: In AWK uninitialized variables are either empty strings or zero, depending on the context, whereas in Icon uninitialized variables have the distinguished value of null, and then you can test whether or not they have a value or not. Do you find wishing things were more one way than the other, or are you pretty much used to the fact that they just treat uninitialized variables differently?

TOWNSEND: I guess I am pretty much used to that one. I probably haven't thought about it in a long time until you asked that question. I guess the main things I like about AWK, depending on what you are trying to do, is that certain things can be written much more concisely in AWK, especially if you're doing stuff involving regular expressions and the sort of pattern matching that can be handled by simple substitutions using those regular expressions. It's also handy the way AWK breaks things into fields. Depending on what particular tasks you are trying to accomplish, it's nice to have that done at the start. Probably the biggest AWK program I have written is a program that checks C programs for their coding style. The particular coding style it is checking is the one that's standard for the Icon system. And so I use an AWK program to check the C code that I am writing for Icon. Ralph, of course, thought that was a very strange idea, but I have found it useful. And that's one program that I don't think I could write nearly as easily in Icon because it makes extensive use of string substitutions.

CARGO: Out of curiosity, who set the coding style, or how long has the coding style been established?

TOWNSEND: It's the sort of thing that has evolved over many years and I don't really know who set it up originally. I think it was probably about two or three years ago that Ralph actually wrote it down and formalized it in a document and that was the point where I wrote this AWK script program. Before then it had just always been kind of informal, "Well, look and see how the rest of the code is done and make your code look the same."

CARGO: When you are coding stuff for the Icon source, is there particular emphasis given to performance or portability or readability?

TOWNSEND: Well, at the level of the coding style, performance is almost never a consideration. That's important when you are designing the algorithms and so on, but at the micro level we generally don't worry about it. Readability and portability are the two important things. And generally we will trade off performance for either of those. As to where they would rank with regards to each other, in one sense the portability requirement has to be taken as an absolute, and so we have a lot of ugly conditionals throughout the code that make it somewhat hard to read. So I guess portability has to be the prime factor.

CARGO: Do you use any of the personalized interpreters, or do you just find you use the standard available Icons for the system you use?

TOWNSEND: Well, I want my own programs to be portable among various Icon implementations, and so I stay away from using personal interpreters. The most I will do is try and agitate to get something incorporated into the standard system. And that's a very long, drawn-out process, but I have occasionally been successful at that, and then I feel I can finally start calling a square root routine in my Icon programs.

CARGO: Is that an example of one you agitated for?

TOWNSEND: Math functions, in general, yes. And I am certainly not the only one that was interested in those.

CARGO: What were some of the others?

TOWNSEND: What I meant was that many people were interested in math functions. But some of the other math functions would be things like logarithms, which I have used for logarithmic scaling on graphs, for instance. And trig functions - sine and cosine, and so forth. What we ended up putting in is pretty much the standard UNIX set of math functions.

CARGO: I meant, aside from math functions, were there features that you wanted put into the language that you succeeded in?

TOWNSEND: For a long time I kept getting bitten by writing ".25" and having this taken not as one quarter but as the dereferenced integer constant 25. So every time the discussion came up, which was about once a year, I would bring that one up and that one has been addressed at least to a point I can live with it now; the Icon interpreter gives you a compile time warning when you try and do something like that. So I still write that occasionally, but I always find out a lot less painfully now. I don't remember any others. There were probably a couple of other minor functions. I was in favor of an entab and a detab function and was able to get those in by promising to write them myself, which I did. A long time back when I was a student in Ralph's string and list processing class, I remember I suggested an escape sequence so that you could put a control T, let's say, in a string literal without having to go look up the octal. That was at a time when we used a lot of those control characters. I don't remember any others, but they would have been similarly minor.

CARGO: In those sorts of instances, was it only Ralph you had to convince or were you in fact trying to convince the other people working on the Icon translator at the same time?

TOWNSEND: Ralph was the one I had to convince.

CARGO: Now, for sure, was there anything else you might want to add?

TOWNSEND: I can't think of anything.

END OF INTERVIEW