The Best of comp.lang.smalltalk

Alan Knight

Smalltalk Solutions

THIS MONTH I'M going to be discussing the Smalltalk Solutions conference that took place in New York at the beginning of March. This isn't my usual territory, since it hasn't got much to do with the Internet, so I'll break with tradition and discuss some of the topics I generally avoid: rumors, impressions, and products I haven't used. Obviously, you shouldn't be basing important decisions on my first looks at a product, or on unsubstantiated rumors. In fact, just to make things more interesting, I made up one of the rumors myself. See if you can spot it yourself before turning to the end for the solution.

In general, the thing that impressed me most about this conference was the maturing of the Smalltalk industry. The number and variety of different applications was remarkable, and they weren't confined to traditional business systems. For example, I was surprised to learn that the driver's license kiosks in Ontario (where I live) are programmed in Smalltalk.

It's no longer the case that everyone is talking about small pilot projects and introducing Smalltalk into the organization. An increasing number of organizations have delivered mission-critical systems in Smalltalk and realized significant gains from them. There's more concern with "business value" and how to make the transition to the "early majority" of users than there is with the latest cool features.

NEWS AND RUMORS

Speaking of cool features, Java is heavily in the news these days, and often cited as a threat to Smalltalk. In fact, many advocates of Java seem to believe that it instantly makes all existing languages and operating systems obsolete. I admit to knowing some people who feel that way about Smalltalk, but the Java zealots are doing a remarkable job of worrying people who really ought to know better. This leads to a couple of interesting Java rumors.

One, which appeared in comp.lang.smalltalk suggested that ParcPlace-Digitalk was in the process of making a Java VM which would be several times faster than Sun's. This is technically plausible, since current Java implementations are abysmally slow, and it shouldn't be that difficult to adapt a Smalltalk VM to run Java. On the other hand, I've seen no confirmation of this, especially not from PPD.

Microsoft is reacting to the Java hysteria, and although they have licensed it for use in their own web browser, they're also at work on a product to rival Java, leveraging their existing technology. In accordance with the emerging standards for naming conventions of such products (bad puns based on coffee) the new product will be named "au lait."

One of the strengths of Java is that implementations are quite cheap, and often free. There have been numerous complaints from the community that, in the pursuit of the corporate market, Smalltalk vendors have priced themselves out of range of individuals and small companies. Anyone who feels this way should be happy to hear the latest from Skip McGaughey (market manager for VisualAge). Responding to a question on his keynote speech, he said that we "absolutely need" a cheap Smalltalk that runs on an 8- to 16-MB machine and comes with multimedia instructional software so that users don't need expensive training. "Are we there today? No. Will we be there a year from now? We have to be."

Getting back into the factual and the present, the draft X3J20 report on the ANSI standard Smalltalk is now available for review. The initial informal review period ended April 30, but review and revisions continue. To get a copy, contact Lynn Barra at 202.626.5738 or lbarra@itic.nw.dc.us. There's also an X3 web page at http://www.x3.org. The ANSI committee has done some very interesting work, and obviously put a lot of thought into defining the language without overconstraining future implementations. They have decided not to define a standard for namespaces, not because they don't think they're important, but because they think standardization now would be premature. Although I'm a little disappointed that vendors won't be forced to implement namespaces, I have to agree with their reasons.

A big part of this column turned out to be about IBM and OTI. That's partly because there was interesting news on that front and partly for another reason. It's been quite a while since the merger and there's still very little infor-

Alan Knight is a Smalltalk guru with The Object People. He can be reached at 613.225.8812 or by email as knight@acm.org.

mation on PPD's future plans, which are of critical importance to anyone working in Smalltalk. I had prepared a modest diatribe on the subject and it had already been typeset when the information finally started to flow. It's still a trickle, but it's enough that I'm willing to hold back my wrath a little while, especially since the trickle contains encouraging words like "no runtime fees."

IBM BUYS OTI

When IBM became a Smalltalk vendor it marked a significant milestone in the evolution of the language, giving it legitimacy in the eyes of many major corporations. IBM had licensed their underlying Smalltalk technology from OTI, and now they have acquired that technology outright.

Given the close relationship between the two companies lately this wasn't a complete surprise, but it did worry a number of people. One worry, for fans of OTI, is that being part of IBM might destroy their unique corporate culture. The other worry, for fans of VisualWorks/Envy, is the long-term outlook for that product. Skip McGaughey tried to dispel these fears in his keynote address.

He re-affirmed that OTI would act as an independent subsidiary of IBM, and that it would continue to be run by Dave Thomas. In fact, he said that all Smalltalk activity within IBM now reports to Dave, so that "in a very real sense, Dave Thomas acquired IBM." He also emphasized the idea of both competing and collaborating. One example of this collaboration was that OTI will continue to supply VisualWorks Envy, enabling their competition, but allowing everyone to win by growing the market. That takes care of one side of the equation, but it remains to be seen how ParcPlace-Digitalk feels about having such an important system component provided by a competitor. Whether or not we see a version of Envy for VisualWave will be a very strong signal of their future direction.

One area where OTI's culture is already being affected is in the relaxation of their vows of silence. OTI staff are known for never letting any information slip before something is officially announced. Skip McGaughee instead emphasized the need for clear communications, even with competitors. For example, he said that IBM will let anyone see their plans for the next version without signing a non-disclosure agreement.

EMBEDDED/SERVER SMALLTALK

OTI's tools for doing embedded and server programming in Smalltalk are one of the most interesting bits of technology I've seen in a while. Although it's an area which is unfamiliar to most of the Smalltalk community, I believe that they will be very important to the future of Smalltalk.

Smalltalk is generally considered to be very resource-intensive. Development environments typically suggest 16 to 32 MB RAM and executables have trouble running in 4 to 8 MB. You certainly wouldn't think of using Smalltalk for a real-time system with only 512 K, would you? OTI would, and they've been doing it for quite a long time. Now they've come out with their second generation of

INFO@SIGS

SIGS Publications, Inc., 71 West 23rd Street, 3rd Floor, New York, NY 10010; 212.242.7447; Fax: 212.242.7574

ARTICLE SUBMISSION

To submit articles for publication, please contact:
John Pugh & Paul White, Editors, 885 Meadowlands Dr. #509, Ottawa, Ontario, K2C 3N2 Canada; email: streport@objectpeople.on.ca

PRODUCT REVIEWS AND ANNOUNCEMENTS

To submit product reviews or product announcements, please contact the Editors at the address above.

CUSTOMER SERVICE

For customer service in the US, please contact PO Box 5050, Brentwood, TN 37024-5050; 800.361.1279; Fax: 615.370.4845; in the UK, please contact Subscriptions Department, Tower Publishing Services, Tower House, Sovereign Park, Market Harborough, Leicestershire, LE16 9EF, UK; +44.(0)1858.435302; Fax: +44.(0)1858.434958

SIGS BOOKS

For information on any SIGS book, contact: Don Jackson, Director of Books, SIGS Books, Inc., 71 West 23rd Street, New York, NY 10010; 212.242.7447; Fax: 212.242.7574; email: donald_jackson@sigs.com

SIGS CONFERENCES

For information on all SIGS Conferences, please contact: SIGS Conferences,71 West 23rd Street, 3rd Floor, New York, NY 10010; 212.242.7515; Fax: 212.242.7578; email: info@sigs.com

BACK ISSUES

To order back issues, please contact: Back Issue Order Department, SIGS Publications, 71 West 23rd Street, 3rd Floor, New York, NY 10010; Phone: 212.242.7447; Fax: 212.242.7574

REPRINTS

For information on ordering reprints, please contact:
Reprint Management Services, 505 East Airport Road, Box 5363, Lancaster, PA 17601; Phone: 717.560.2001; Fax: 717.560.2063

ADVERTISING

For ad information for any SIGS publication, please contact:

East Coast/Europe:	Gary Portie
Central US:	Elisa Marcus
Recruitment:	Michael Peck
Exhibit Sales, West Coast:	Kristin Viksnins
Exhibit Sales, East Coast:	Sarah Olszewski

Phone: 212.242.7447; Fax: 212.242.7574; email: sales@sigs.com

West Coast:	Diane Fuller

Phone: 408.255.2991; Fax: 408.255.2992; email: dhfsigs@hooked.net

INTERNATIONAL OFFICES

SIGS Conferences Ltd., Brocus House, Parkgate Road, Newdigate, Surrey RH5 5AH, United Kingdom. Phone: 011.44.1.306.631.331; Fax: 011.44.1.306.631.696; email: 100131,3500@compuserve.com.

SIGS France, 105 rue Jules Guesde, 92532 Levallois Perret Cedex, Paris, France. Phone: +33 (1) 41 06 18 00; Fax: +33 (1) 41 06 18 19; email: 100631,1050@compuserve.com.

SIGS Conferences GmbH, Odenthaler Strasses 47, D-51465 Bergisch Gladbach, Germany. Phone: 011.49(0).2202.936.810; Fax: 011.49(0).2202.936.812; email: 100634,2070@compuserve.com.

SIGS HOME PAGE AND ONLINE MAGAZINES

Access the SIGS Home Page at http://www.sigs.com;
Object Currents at http://www.sigs.com/objectcurrents; and
The X Spot at http://www.sigs.com/xspot.

embedded tools and they're starting to promote them more aggressively.

In getting such small footprints, they have an advantage over most of us because they're typically writing for systems that don't have screens, mice, keyboards, or disk drives. All these things need code to control them, which takes spaces. On the other hand, these absences make developing and debugging with these machines extremely difficult. OTI's toolset tries to make it more like regular development. Here's a quick summary of the features I found most interesting.

You develop on a workstation, but with a difference. In regular Smalltalk your development and execution environments are the same. You write code, then execute it in the same environment that runs your browsers, compilers, and so forth. When you're ready to deliver, you strip out what you're not using. In Envy/Embedded you create a specification of an image to run on the embedded system, and you write code to execute in that environment. The class libraries you use can be entirely different than what's on your workstation.

When you're ready to run, your code is transferred (through a serial port or network interface) to the real machine and run. You have full interactive debugging facilities, it's just that the debugger is on your workstation and the code being run is on the target system. You have remote inspectors and workspaces, single-stepping, and the ability to save code and continue.

For packaging, there are a number of very interesting features. The entire virtual machine is re-entrant, so it can be put in ROM and shared between multiple images. The same thing can be done with large parts of the image. On some real-time operating systems Smalltalk can use the operating system threads instead of the normal Smalltalk processes.

It's these last two that seem to me to have the most significant implications for desktop environments. Better separation of development and delivery environments is important, but the ability to share most of the environment in read-only mode could easily lead to truly shareable Smalltalk DLL's. This would let me run many fine-grained Smalltalk applications simultaneously without the memory overhead of starting a separate VM for each one. I don't think that the use of real operating system threads is critical if you have a non-blocking API capability, but it's something for which Smalltalk is often criticized, so it's nice to see a real implementation.

A lot of these features are very similar to IBM's forthcoming MVS Smalltalk, and this is no coincidence. Brian Barry of OTI, in presenting the embedded product, described MVS as a really, really large embedded system. Many of the characteristics of embedded systems are shared with servers, and many of the same features are important.

PROGRAMMING EPISODES

Ward Cunningham gave a talk on a model of the development process, subtitled "Finding and Exploiting Great Objects When You Barely Have Time to Think." Ward works in the financial world, with very demanding customers and very tight deadlines. This is his model of how to develop in that environment and still produce great code. I guess it's a development method, but it's a lot looser and willing to rely on people's competency than most of the methods I've seen. That makes it appealing to me as a programmer, but it still has enough structure that I can believe it would help meet deadlines. That's quite an accomplishment.

First, a bit of background, in case you're unfamiliar with Smalltalk theology. Ward Cunningham is a very long-time Smalltalker, who worked at Tektronix in close collaboration with Kent Beck. They did a lot of cool stuff together, like designing the HotDraw graphical editing framework and inventing CRC cards. They were also among the first to look at applying patterns to software, and although this talk was not described in terms of patterns, the influence was clear.

He started with a very simple structure for software development, which was successively elaborated with more detailed ideas, applicable in particular situations. It's hard to describe, so I'll just give a bit of flavor by paraphrasing a couple of the ideas.

Spike Solution

You've got an informal labor plan and you want to move towards implementation. You need to do some preliminary coding to make sure you understand the requirement and its implications, but you don't want to get bogged down in dealing with the complexities of existing code. So, write the smallest possible code to perform that requirement, independent of the existing mechanisms.

For example, take a clean image and implement the absolute basics of that requirement, as fast as possible. This is called a "Spike Solution" because it's like driving a spike into a wall. You do it to find out how thick the wall is and where you'll come out. You stop driving the spike as soon as the tip comes out the other side. Later on you'll drive the nails for real.

Motivated Consolidation

Consolidation is important, but your consolidation will be better the longer you put it off. Also, in the normal course of things you will never consolidate, because this is an environment where you barely have time to think. Therefore, you consolidate when, and only when, it's the shortest route to getting something out the door. Fixing the code and adding the new feature will take less time than just hacking in the new feature, and you only do it for regions of the code where it's motivated (i.e., funded). One of the essential elements for consolidation is regression tests. They're incredibly liberating, because they let you change something radically and still know if it works or not.

If this looks interesting, you can find more information, including the "Episodes" pattern language on which this is based, on Ward's web site at http://c2.com.

The solution to the rumor puzzle is, read from right to left: "romur tial ua eht pu edam I"