Notes on Notes on Postmodern Programming: Radio Edit

James Noble
Victoria University of Wellington
Wellington, New Zealand
kjx@mcs.vuw.ac.nz

Robert Biddle
Carleton University
Ottawa, Canada
robert_biddle@carleton.ca

ABSTRACT

These notes have the status of letters written to ourselves: we wrote them down because, without doing so, we found ourselves making up new arguments over and over again. So began the abstract of our earlier paper *Notes on Postmodern Programming*. We now revisit the issue of postmodern programming, and attempt to address some of the questions raised by our exposition. To illustrate the nature of postmodernism we do not do this directly, but instead present a series of snapshots, parodies, and imagined conversations that we hope will help. What do you think of the abstract so far? Self-reference and a irreverent approach are part of this topic, so it's important to chill out and let things flow. We claim that computer science and software design grew up amid the unquestioned landscape of modernism, and that too often we cling to the otherwise ungrounded values, even as modernism itself is ever more compromised.

Categories and Subject Descriptors

D.2.2 [**Design Tools and Techniques**]: OO design methods

General Terms

Design

Keywords

Object-Oriented Design, Object-Oriented Programming

0. RADIO EDIT

This paper is a condensed version of *Notes on Notes on Postmodern Programming* [23].

1. MANIFESTO

> What are you reading?
>
> Some weird thing from OOPSLA [22]: I can't imagine where they find this junk.

OOPSLA'04, Oct. 24-28, 2004, Vancouver, British Columbia, Canada.
ACM 1-58113-833-4/04/0010.

0 Manifesto

The ultimate goal of all computer science is the program. The performance of programs was once the noblest function of computer science, and computer science was indispensable to great programs. Today, programming and computer science exist in complacent isolation, and can only be rescued by the conscious co-operation and collaboration of all programmers.

The universities were unable to produce this unity; and how indeed, should they have done so, since creativity cannot be taught? Designers, programmers and engineers must once again come to know and comprehend the composite character of a program, both as an entity and in terms of its various parts. Then their work will be filled with that true software spirit which, as "theory of computing", it has lost. Universities must return to programming. The worlds of the formal methods and algorithm analysis, consisting only of logic and mathematics, must become once again a world in which things are built. If the young person who rejoices in creative activity now begins his career as in the older days by learning to program, then the unproductive "scientist" will no longer be condemned to inadequate science, for their skills will be preserved for the programming in which they can achieve great things.

Designers, programmers, engineers, we must all return to programming! There is no essential difference between the computer scientist and the pro-

Figure 1: Notes on Postmodern Programming. Section 0: Manifesto.

> Let's see. Oh, it's the Bauhaus manifesto, cool.
>
> Bauhaus? Weren't they a punk band?
>
> No, the real Bauhaus, the architecture and design school in Germany during the Weimar republic. We studied it at Art School. I still have a book about it somewhere [31].
>
> Oh. That's weird. An architecture school had a communist manifesto? And about programming?
>
> Huh? No, not the communist manifesto. Let's see. Oh, it's not exactly the Bauhaus manifesto. They've substituted words like "programming" instead of "building".
>
> Really? You're kidding? Let me google it. Oh yeah. I guess they're making some point about programming being like building, and software engineering being like architecture. Hmm, only the manifesto says they're really the same.
>
> Well, the Bauhaus people were idealists, that's for sure.
>
> Hmm, some of the other wording seems to be different. Maybe it's a different translation. Where are these guys from? Maybe they don't speak American. But they are idealists. And look where it got the Bauhaus: total irrelevance!
>
> But lots of modern architecture came straight from the Bauhaus: all those big rectangular modern office blocks.

Well, it was the 60s, I guess everyone was doing the same thing.

Actually the Bauhaus was in the days of the Weimar republic: the 1920s. They designed their own modern building in 1925. It looked nice: not the kind of giant nightmare of Le Corbusier . . .

What? Modern architecure in the 1920s? No way. But about that Courvoisier?

Anyway, they were idealists. They thought that the academic world and the world of industry should work together.

Really? That's what I always say. Those university space-cadets could really learn something. So this thing is really saying that everybody should work together, is that all?

Well you're reading it. What's the title again?

Um, Notes on Postmodern Programming.

Really? Postmodern? Because the Bauhaus was all about modernism.

Oh right. So why is this here at all?

Maybe they're making fun of it?

Really? But it seems so serious. How can they be advocating moderism and postmodernism at the same time?

2. ADVENTURE
The Brain of Alan Turing!

Paragraph 5

JIM: Vodka lime tonic in a tall glass.

BOB: Martini, Tanqueray if you have it.

Suddenly you realise what you've been missing — you look over the back of the booth to catch the eye of the waitress and order an OOPSLA Sunrise (Tequila, Orange juice, and Lingonberry juice). When you turn around, you realise Alan Turing is sitting across the booth from you. He is balancing an apple in his left hand.

ALAN: Excuse me, my machine seems to have run out of tape.

YOU: Tape?

ALAN: Yes. Tape. It's supposed to be infinite, you see.

YOU: Infinite?

ALAN: The Tape. Infinite. Yes.

He holds the apple in his hand.

ALAN: Consider this apple. Almost too good to be true, really. What does it represent? Any apple. The apple from the tree of knowledge of good and evil. The apple from the tree of life.

He puts the apple on the table and opens it. Its logo glows across the dim bar. Due to very expensive product design, the logo is the right way up when the screen is open. Alan Turing presses the power button while holding down the "Command-V" key combination.

Figure 2: Photograph of Wellington, New Zealand, showing lack of order, and wilfulness.

ALAN: If you hold down "Command-V" you get all the bootup details instead of the stupid grey screen with the little twizzy spinning thing. All the cool stuff about ethernet addresses and all. What's odd is that this seems to make the machine boot up quicker, although it doesn't of course. How could it? You could set "nvram boot-args = -v" but this is more tactile. I wish I'd had one of these back at Bletchley Park.

You drink more tequila.

ALAN: The apple isn't really here — I'm not really here — I'm just a talking point, a puppet created by some underachieving professor who's never written a program and couldn't write a theorem to save his life. If you stood up for a moment you'd see that I'm just a collection of latex and pneumatics. Like Stephen Hawking or Davros — same thing really.

He gestures behind him. As he does so, you see a "Weta Digital" trademark tattooed on the inside of his forearm.

You drink more tequila.

When you look up, Alan Turing has vanished. The stereo seems to be playing a recording of Kylie Minogue singing Alvin Lucier's "I am sitting in a room" remixed with John Cage's "Four Minutes Thirty-Three Seconds."

Go to Paragraph 14.

3. LE CORBUSIER

The Plan is the generator.

Without a plan, you have lack of order, and wilfulness.

The Plan holds in itself the essence of sensation.

The great problems of to-morrow, dictated by collective necessities, put the question of "plan" in a new form.

Modern life demands, and is waiting for, a new kind of plan, both for the house and for the city.

— Le Corbusier [28]

4. MORE ADVENTURE

Paragraph 9

BOB: So, if there were three key points they would be that:

- Modernism is ultimately the idea of a big story of progress, that liberty, socialism, etc. will eventually lift humankind from the mud to the stars: all will be equal, educated, happy ...
- Postmodernism: both fulfills modernism: in postmodernity everyone is equal, educated, liberated, free; and replaces it, because this equality, education, liberation, salvation, freedom turns out not to be like we thought.
- Computer Science / Software Engineering / Programming is deeply implicated in all of this as a participant: programs and programmes are part of the infrastructure of society, and have been greatly affected by the these developments.

JIM: Too fricken weak: this needs to be rephrased properly.

BOB: Ahh, who cares?

BOB: As I was saying:

- Computer Science is equally an instigator, because it provides much of the underlying technologies accelerating the shift into postmodernity.

JIM: Three points!?!

BOB: So? Who's counting?

Turn to Paragraph 10.

5. NO BIG PICTURE

This postmodern programming stuff just seems to be an excuse to create software by sticking stuff together.

Yeah, there doesn't really seem to be any theory at all. Losers.

Adam was the Average Joe all of America was hoping for.

Or there's a whole bunch of theories. How could that ever work?

Yeah, it would be an a band where everyone played different instruments and they all just made it all up as they went along. Losers.

But ...

Yeah, well, you know what I mean. Losers.

This week, Survivor is on Wednesday.

Well, the human mind can only understand one thing at a time, that's for sure.

Yeah, who ever heard of, like, a movie being split up with lots of other movies? Losers.

Get the door. It's a cheesy good time from Dominos.

Nobody would ever get used to it.

Yeah. And they want to force it on us. No way, loosers.

Right.

Yeah.

Tonight, on Viewer's Choice Pay-Per-View. Wrestlemania. Check channels 300 and up.

Yep.

Losers.

I'm going to make a collect call, by dialing down the center.

You could never learn to deal with it.

Yeah.

And you can imagine how it would creep in. All of a sudden it could get jumbled up. You wouldn't know what was what. It would be like getting some other drink when you were expecting a cold refreshing Coca-Cola.

6. YET MORE ADVENTURE

Paragraph 23

We claim that computer science and software design grew up amid the unquestioned landscape of modernism, and that too often we cling to the otherwise ungrounded values, even as modernism itself is ever more compromised.

There are implications for practical programming. When we work within modernism, this affects our programs, systems, languages, and applications. When we work within modernism, this affects the way we organise ourselves. The ways in which our work is affected may be useful, or may not, we may like it, or we may not, but the effects will be there nonetheless.

This affects we way we do things, even, and actually especially, when we try to do them *right*. With our technology: all those languages, all those toolsets, all those environments. And with ourselves, and how we organise how we work, how we succeed, how we fail. From the beginning of codification in Garmisch [19], this is how we think about what we do. Even when we don't.

Others have raised similar issues, and before we did. Wall's remarks [30] show how the success of Perl is related to his new approach; the exposition of Hall et al. [24] shows how pervasive the impact of the traditional approach is on our organisations.

All this has an impact on our past and our future, because it concerns reuse and reusability. Whether and how much these are possible relates to our view of how things must be done. Our discipline has a long history with reuse, but it is history itself affected by the stories we tell [7].

The idea of modernism points the way, and the style it engenders, progress, functionality, efficiency, all help us to follow. It brings hope, and soon comes confidence, but too often arrogance. It becomes easy to overlook the questions that arise. Is this the right way? Why is this so difficult? Couldn't we do this another way? What are those people doing over there? Where is the coffee? But if we repeat our own big stories to ourselves, if we repeat them louder and longer, the questions can soon be forgotten.

7. REFERENCES

[1] Alfred V. Aho, Ravi Sethi, and Jeffrey D. Ullman. *Compilers, Principles, Techniques, and Tools*. Addison-Wesley, 1986.

[2] Alfred V. Aho and Jeffrey D. Ullman. *Principles of Compiler Design*. Addison-Wesley, 1977.

[3] American National Standards Institute, 1430 Broadway, New York, NY 10018, USA. *Military Standard Ada Programming Language*, February 17 1983. Also MIL-STD-1815A.

[4] Ronald M. Baecker and Aaron Marcus. *Human Factors and Typography for More Readable Programs*. Addison-Wesley, Reading, MA, USA, 1989.

[5] Kent Beck. *Extreme Programming: Embrace Change*. Addison-Wesley, 2000.

[6] Samuel Beckett. *En attendant Godot. English translation by the author.* Grove Press, 1954.

[7] Robert Biddle, Angela Martin, and Robert Biddle. No name: Just notes on software reuse. *SigPlan Notices: Proceedings of the OOPSLA Onward Track*, 38(2):76–96, December 2003.

[8] Lewis Carroll. *Alice in Wonderland*, volume 11 of *Project Gutenberg*. Project Gutenberg, P.O. Box 2782, Champaign, IL 61825-2782, USA, 1991.

[9] O.-J. Dahl, E. W. Dijkstra, and C. A. R. Hoare. *Structured Programming*. Academic Press, 1972.

[10] Edsger W. Dijkstra. My hopes of computing science. In *Proc. 4th Int. Conf. on Software Engineering, Munich*, pages 442–448. IEEE and `http://www.cs.utexas.edu/users/EWD/ewd07xx/`, September 1979.

[11] Umberto Eco. Reflections on the name of the rose. *ENCOUNTER*, 64, April 1985.

[12] Richard P. Gabriel. LISP: Good news, bad news, how to win big. *AI Expert*, 6(6):30–39, 1991.

[13] Douglas R. Hofstadter. *Godel, Escher, Bach: An Eternal Golden Braid*. Basic Books, Inc., 1979.

[14] Richard Horn. *Memphis: Objects, Furniture and Patterns*. Simon & Schuster, 1986.

[15] Guy L. Steele Jr. Both ways, now. `http://www.poppyfields.net/filks/00041.html`.

[16] Bill Manhire and Gregory O'Brien. *The Brain of Katherine Mansfield*. Auckland University Press, 1988. `http://www.het.brown.edu/people/easther/brain/index.html`.

[17] Paul Morley. *Words and Music: A History of Pop in the Shape of a City*. Bloomsbury, 2003.

[18] Robin Murray. *Fordism and Post-Fordism*, pages 167–276. Academy Editions, 1992.

[19] P. Naur and B. Randell, editors. *Software Engineering: Report of a conference sponsored by the NATO Science Committee*. NATO Scientific Affairs Division, Brussels, 1969.

[20] Greg Nelson. *Systems programming with Modula-3*. Prentice-Hall, Inc., 1991.

[21] Theodor Holm Nelson. *Computer Lib / Dream Machines*. None listed, 1974.

[22] James Noble and Robert Biddle. Notes on postmodern programming. In Richard Gabriel, editor, *Proceedings of the Onward Track at OOPSLA 02, the ACM conference on Object-Oriented Programming, Systems, Languages and Applications*, pages 49–71, Seattle, USA, 2002. http://www.dreamsongs.org/.

[23] James Noble and Robert Biddle. Notes on notes on postmodern programming. *SIGPlan Notices: Proceedings of the OOPSLA 2004 Onward Track, To Appear*, 2004.

[24] Hugh Robinson, Fiona Hovenden, Pat Hall, and Janet Rachel. Postmodern software development. *Computer Journal*, 41(6), 1998.

[25] Ravi Sethi. *Programming languages (2nd ed.): concepts and constructs*. Addison Wesley Longman Publishing Co., Inc., 1996.

[26] A. Silberschatz, J. Peterson, and P. Galvin. *Operating System Concepts*. Addison-Wesley, 3 edition, 1991.

[27] John Simpson, editor. *Oxford English Dictionary*. Oxford University Press, OED online `http://oed.com/` edition, 2004.

[28] Le Corbusier Translated by Frederick Etchells. *Towards a New Architecture*. Dover Publications republication of 1931, 1986.

[29] Adriaan van Wijngaarden, B. J. Mailloux, J. E. L. Peck, C. H. A. Koster, M. Sintzoff, C. H. Lindsey, L. G. L. T. Meertens, and R. G. Fisker. Revised report on the algorithmic language ALGOL 68. *Acta Informatica*, 5(1–3):1–236, December 1975.

[30] Larry Wall. Perl, the first postmodern computer language. `http://www.wall.org/~larry/pm.html`, Spring 1999.

[31] Frank Whitford. *Bauhaus (World of Art)*. Thames & Hudson, 1984.

8. M-X DOCTOR

I am the psychotherapist. Please, describe your problems. Each time you are finished talking, type RET twice.
I don't know if postmodern programming is meant to be serious or a joke.
Why do you think it is meant to be serious?
Well, there are some good points about the way things really are.
Are you afraid of that?
A bit, because it means that we have been deluding ourselves about the nature of computer science. But maybe it is meant to be a joke.
Why do you think maybe it is meant to be a joke?
Well, they include lots of stuff from books and cartoons and movies, and they make fun of computer science. Can they be serious and joking at the same time?
I could ask the same thing myself.
I guess they could be making serious points in a postmodern way.
Why do you say that?
I suppose a postmodern point of view would involve a number of different influences stuck together, some serious, some funny, just like modern life. I mean postmodern life. Does that make sense?
Please allow me to do the questioning.