Olé, olé, olé

Adrian Kosmaczewski

2009-02-15

I just stumbled into this amazing TED talk by Elizabeth Gilbert via James Duncan Davidson (@duncan in Twitter) and I want to share it with you with some very personal thoughts below.

I've been fortunate enough to earn a pretty decent living doing basically what I consider a hobby for the past 13 years, which is typing code on a computer and see if it works. Which most of the time doesn't, but that's part of the game.

I strongly believe in what Elizabeth says in this talk, and I have believed in this for years. I deeply believe that we, software developers, software engineers, both self-taught and those coming out of college, are just creators, just as Elizabeth describes them. Simple creators, being able to provide new ways to information to be shown, to flow, to entertain, to move. Simple channels through which ideas are transformed into tools, behaviours, images and sound.

There's been a huge debate in this matter. Knuth named his masterpiece "The Art of Computer Programming", and the single choice of this title has sparkled a longlasting debate in the software community, one that this essay is unfortunately going to feed, too.

Interestingly enough, Knuth developed his own typesetting system for his book, TeX, which is named after the Greek word meaning "art or craft". His work not only had to be the most important book ever written on programming, but also, it had to be beautiful.

It had to be an object of art.

And I think that programming itself is art. And I think that programmers are artists. And this is maybe the single reason why so much has been written about programmer productivity, why software project management is so hard, why discussions around programming languages distort into trolls and heated arguments, and why you feel this anger against this words on my blog and you call me names.

This is why writing opinionated software is key to success, that's why the best software companies take time into creating great software development environments that stand out, that's why Peopleware is so important, even 20 years after being published for the first time.

It is all about letting the flow of art come through the person whose hands are on the keyboard. It's all about letting this happen. It is not us who write, it is the writing that comes to us.

Software is art, and as such, it needs time, patience, iterations, silence, passion, coffee, naps, pizza, books, compilers, laughs, Nintendo Wiis and unit testing suites.

The creation of good software is embodied in the creative process itself. The best engineers I know suffer from this process as much as they enjoy it, using an iterative process of trial and error which, even after all these years, still applies. The best software developers release sometimes early, sometimes late, sometimes with quality, sometimes not, but they release. They refactor. They document. They teach others about all of this. They always think that they can do better.

They always think, as Elizabeth says in her talk, that their current work is the worst in the history of programming: "Not just bad, but the worst". They suffer about it. But they release, and they fight against the fear of being critisized because of their choice of programming languages, operating systems, tools, processes, insufficient testing or design patterns.

Real artists ship. The making of this industry is full of examples of why software is an art: the first Macintosh, Smalltalk, NeXTstep, the Internet, Erlang, Apache, Ruby on Rails, UNIX & C, Lisp; just glimpses of wisdom, brilliantly crafted, that struck as obvious yet incredible, and which prompt a huge crowd to cheer up and applause.

Anyway, I'm not as famous or well-known as Elizabeth or Duncan. I have not yet done anything such as Ant or Tomcat (originally written by Duncan, by the way), even if I release software and projects with an increasingly high rate lately, and with many projects in the pipeline these days. I hope that my best successes are still ahead of me. And I hope you'll enjoy them one day, too.

Do I think that a little "genie" is besides me? Yes I do. And given the extremely rational background of most engineers out there, stating such an argument will raise more chuckles than anything else. Heck, who cares.

If you ask me, there is something magic out there.

PS: there's this quote attributed to Jorge Luis Borges which says that "publishing is a way to stop editing"… and I thought about it just after publishing this post. I don't know if he really said that, but in any case I agree. The difference being that, in our case, we refer to publishing as "releasing". But the feeling is the same.

PS (2): I've already written about the importance of delivering working software. I just forget about all I've written or linked to.