What would it take for us to move from “technical debt” to “technical health”

Lessons learnt at Agile 2017 by Leon Maritz

Intro

We are going to talk about where the term technical debt came from.

Then what impact it has had.

What unforeseen problems the term created.

Why we should change to “Technical Health”.

What could help.

What is technical debt

10 min

On your experience with Technical Debt

Definition

Ward Cunningham wanted to explain why he was refactoring the code for WyCash - 25 years ago.

Debt

Speed

Burden

Agility

Ward on technical debt

Debt

I coined the debt metaphor to explain the refactoring that we were doing on the WyCash product. This was an early product done in Digitalk Smalltalk, and it was important to me that we accumulate the learnings we did about the application over time by modifying the program to look as if we had known what we were doing all along and to look as if it had been easy to do in Smalltalk.

The explanation I gave to my boss, and this was financial software, was a financial analogy I called "the debt metaphor". And that said that if we failed to make our program align with what we then understood to be the proper way to think about our financial objects, then we were gonna continually stumble over that disagreement and that would slow us down which was like paying interest on a loan.

Ward on technical debt

Speed

With borrowed money you can do something sooner than you might otherwise, but then until you pay back that money you'll be paying interest.

I thought borrowing money was a good idea, I thought that rushing software out the door to get some experience with it was a good idea, but that of course, you would eventually go back and as you learned things about that software you would repay that loan by refactoring the program to reflect your experience as you acquired it.

Ward on technical debt

Burden

I think that there were plenty of cases where people would rush software out the door and learn things but never put that learning back into the program, and that by analogy was borrowing money thinking that you never had to pay it back.

Of course, if you do that, you know, say with your credit card, eventually all your income goes to interest and your purchasing power goes to zero.By the same token, if you develop a program for a long period of time by only adding features and never reorganizing it to reflect your understanding of those features, then eventually that program simply does not contain any understanding and all efforts to work on it take longer and longer. In other words, the interest is total -- you'll make zero progress.

Ward on technical debt

Agility

A lot of bloggers at least have explained the debt metaphor and confused it, I think, with the idea that you could write code poorly with the intention of doing a good job later and thinking that that was the primary source of debt.

I'm never in favor of writing code poorly, but I am in favor of writing code to reflect your current understanding of a problem even if that understanding is partial.

You know, if you want to be able to go into debt that way by developing software that you don't completely understand, you are wise to make that software reflect your understanding as best as you can, so that when it does come time to refactor, it's clear what you were thinking when you wrote it, making it easier to refactor it into what your current thinking is now.

In other words, the whole debt metaphor, let's say, the ability to pay back debt, and make the debt metaphor work for your advantage depends upon your writing code that is clean enough to be able to refactor as you come to understand your problem.

I think that's a good methodology. It's at the heart of Extreme Programming. The debt metaphor is an explanation, one of many explanations why Extreme Programming works.

60%-90%

Development is on maintenance.

Banker & Software Technology Support Center [Banker 1991, 1993; STSC 2003]

What does it cause?

- Slow releases
- Slow to market
- Code Re-writes
- Code translations
- Business distrust
- Project overruns
- Long onboarding
- Generally bad for developers

How is it seen from a financial point?

It is seen as an expense.

What impact does finance have on it?

When speaking to stakeholders and people responsible to the investors. They would see the problem of technical debt as a “bottom line problem”.

But it’s really a “ top line problem”

10 min

On what we as a community can do about Technical Debt

What can we do.

Change the term in everyday use.

Let us to move from using “technical debt” to “technical health”

Change it in financial impact, log is as a Capital expenditure.

There is a new updates to Financial Accounting Standards Board (FASB)

Final point

Let’s change the perception by changing the term.

“Fear is the path to the dark side. Fear leads to anger. Anger leads to hate. Hate leads to suffering.”

- Yoda

Let’s work on our “Technical health.”

Thanks!

Contact us:

Leon Maritz
leon@leonmaritz.com