Software Development as a Game

Earl T. Barr
e.barr@ucl.ac.uk
earlbarr.com

The One Less Traveled by

Image from CNN

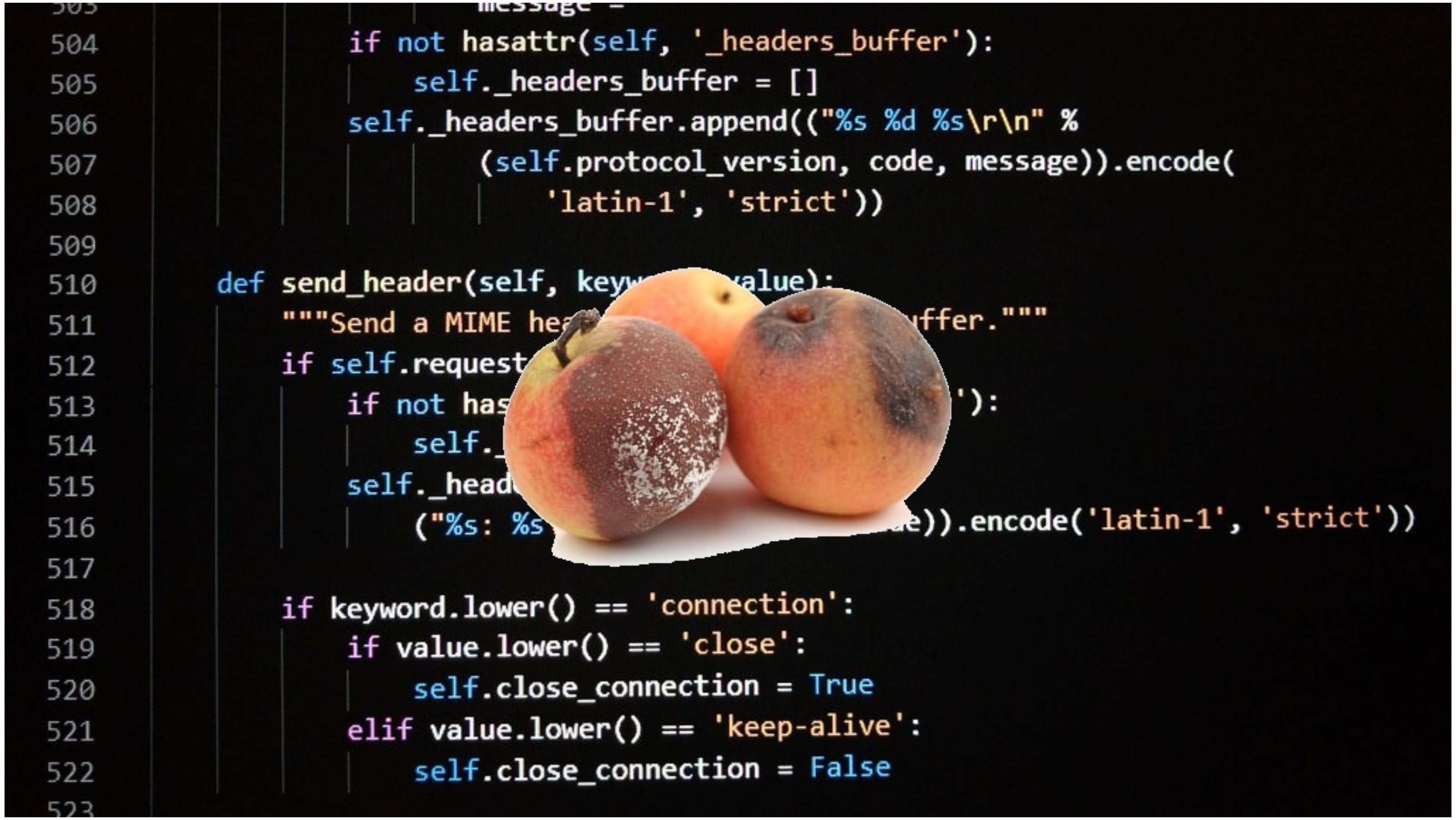
Image by Mark Lenihan

The Tyranny of “If it ain’t broke, don’t fix it.”

Image by Mariakray from Pixabay

Image by M. Chen

Code Rot



```
                    message =
            if not hasattr(self, '_headers_buffer'):
                self._headers_buffer = []
            self._headers_buffer.append(("%s %d %s\r\n" %
                    (self.protocol_version, code, message)).encode(
                        'latin-1', 'strict'))

    def send_header(self, keyw          value):
        """Send a MIME he                    ffer."""
        if self.request
            if not has                          '):
                self._
            self._head
                ("%s: %s                        e)).encode('latin-1', 'strict'))

        if keyword.lower() == 'connection':
            if value.lower() == 'close':
                self.close_connection = True
            elif value.lower() == 'keep-alive':
                self.close_connection = False
```

Code image by Jonathan Cutrer and peaches by Erik Dietrich from NDepend

Refactoring = Spring Cleaning Code

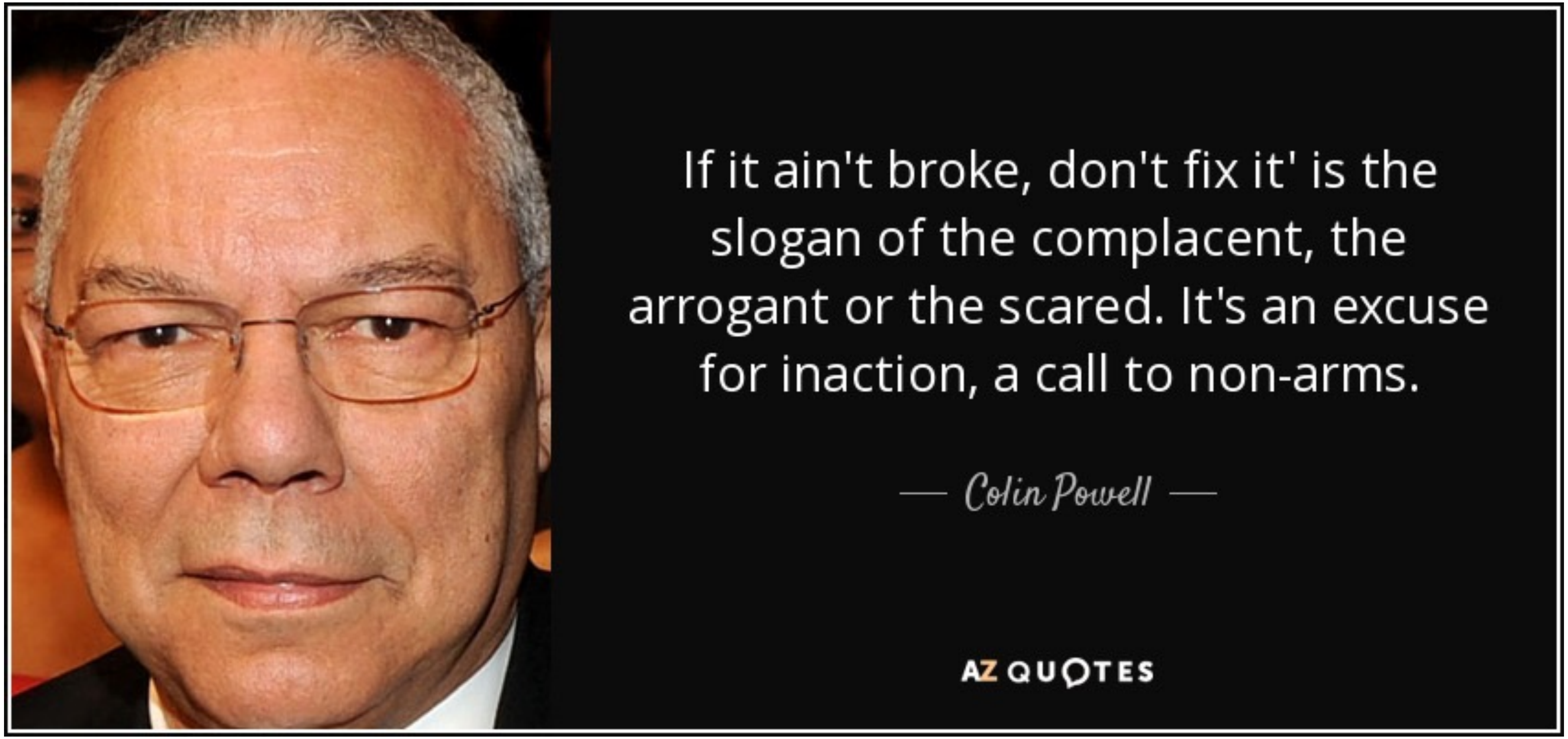
Spring cleaning by Nick Youngson CC BY-SA 3.0 Alpha Stock Images

We Need to Refactor the Code

“If it ain’t broke, don’t fix it!”

Comic from Dilbert.com

Discussing the Boss' Decision



If it ain't broke, don't fix it' is the slogan of the complacent, the arrogant or the scared. It's an excuse for inaction, a call to non-arms.

— Colin Powell —

Image from AZ Quotes

Old Meta’s Old Motto

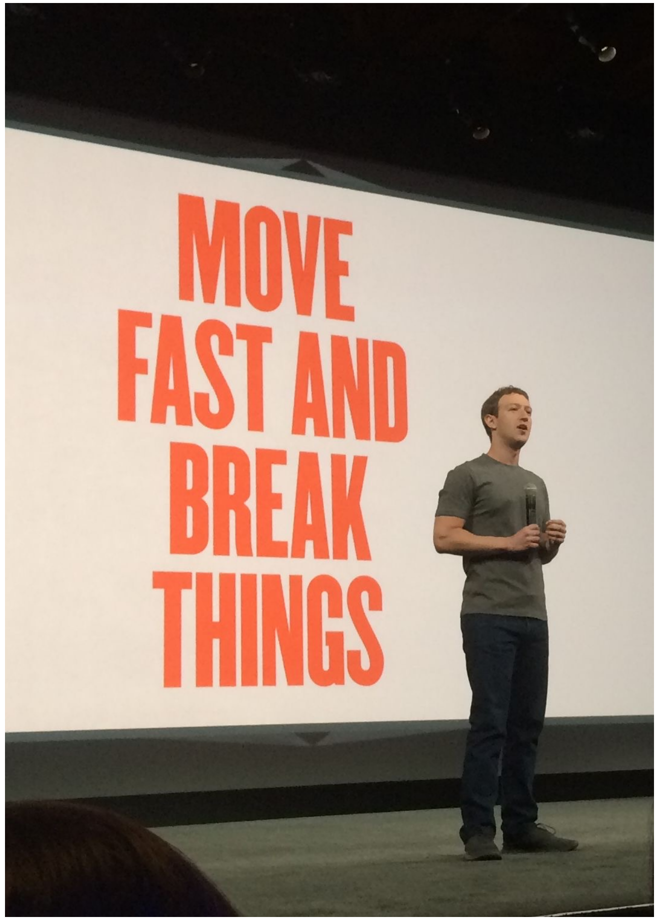


Image from Wikimedia Commons

Refactoring as Burglary

Image by yayayoyo from istockphoto

Sherlock Holmes on the Refactoring Case

Image by OpenClipart-Vectors from Pixabay

The Problem: Misaligned Incentives

Image by Nick Tune

Software Development Processes Based on Intuition or Emergent?

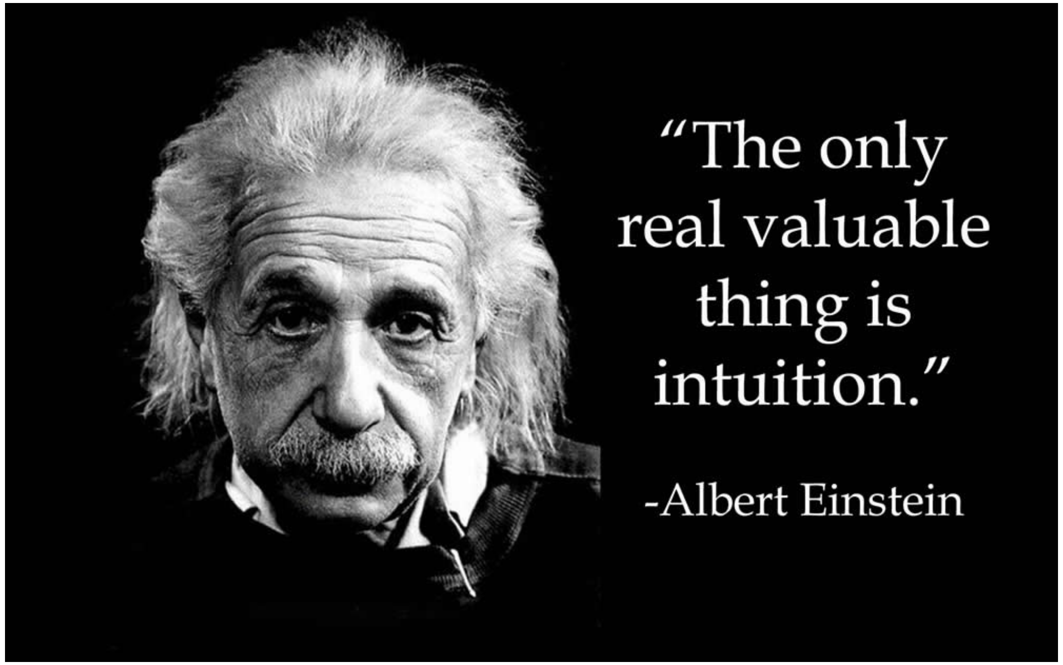


Image by Lorenz Terita from Medium

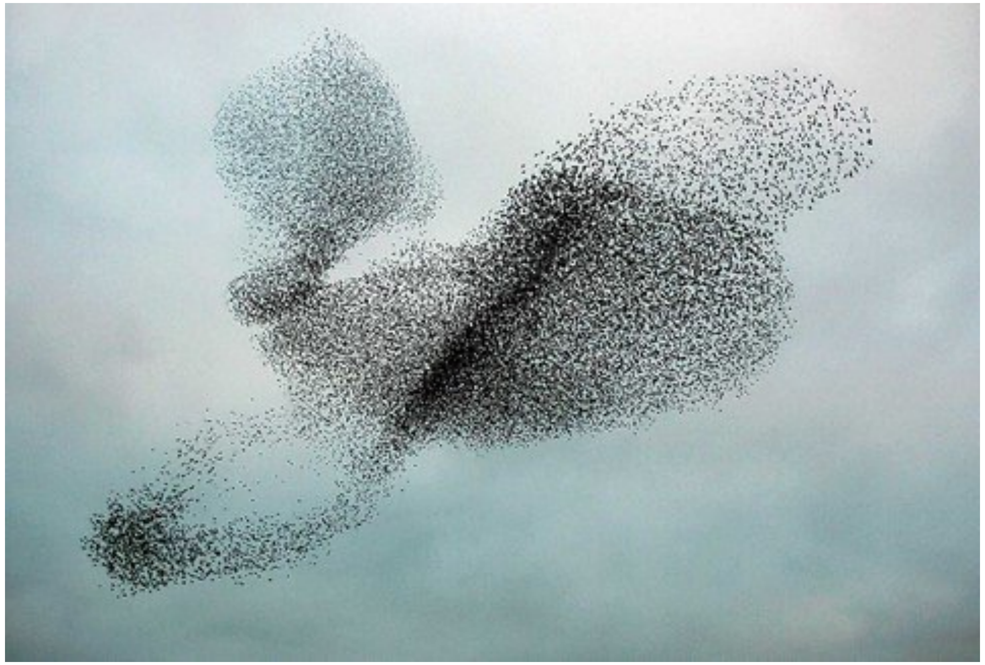


Image by Neil Gottel from Quora

Software Development Processes Based on Intuition or Emergent?

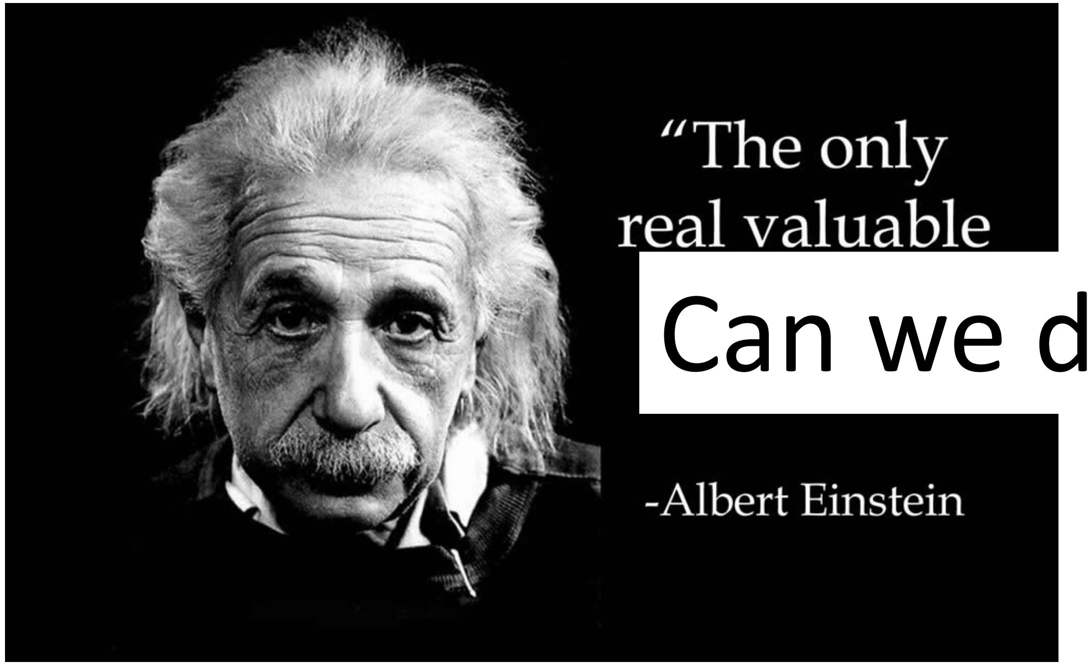


Image by Lorenz Terita from Medium

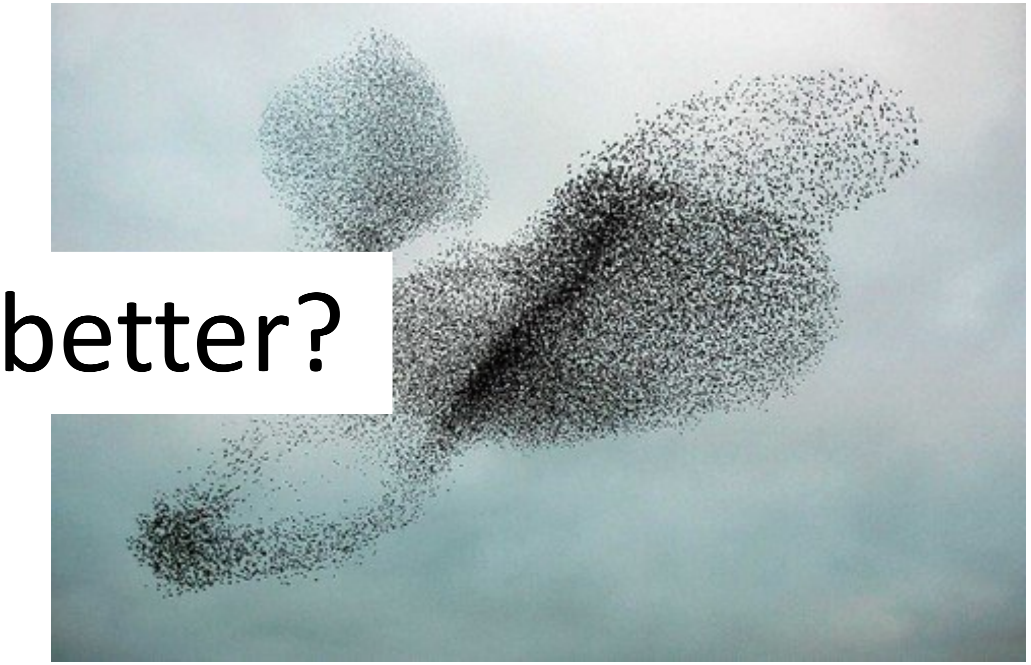


Image by Neil Gottel from Quora

Can we do better?

Game Theory to the Rescue

Édouard Detaille, Public domain, via Wikimedia Commons

Priority Inflation

Many software development processes use issue trackers. Issues are prioritised.

Trouble is, these priorities tend to be inflated.

My co-authors and I used empirical game theory to model priority inflation and to design a new, deployable process to combat it.

Hyperinflation in Germany, 1923

Image from Schoolhistory.org.uk

To Kludge or Not to Kludge

A *kludge* is a shortcut that a developer takes when partially fixing a bug or partially realising a feature.

A kludge can be a necessary expedient to meet a deadline. Or a kludge can be simple laziness.

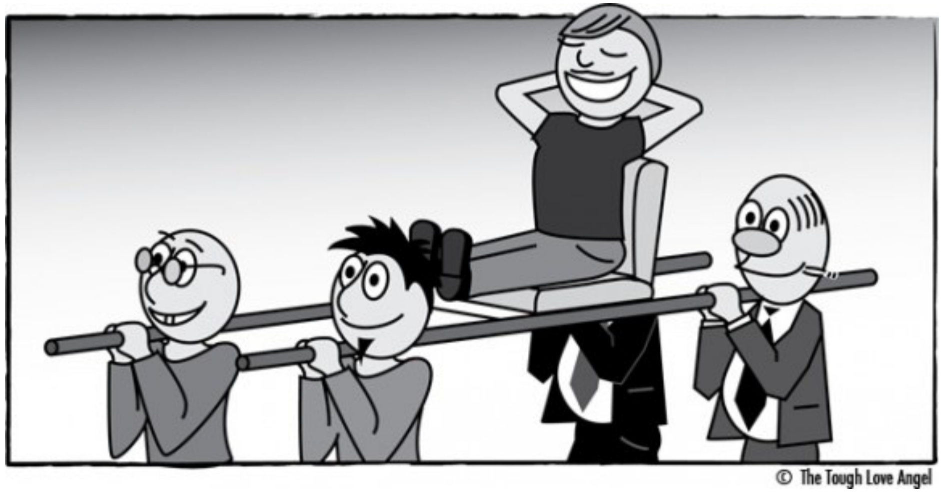


Image by Mika Marjalaakso from The Tough Love Angel blog

I am currently working with collaborators to use game theory to understand and discourage lazy kludging.

The Software Engineering Game

Interesting Characteristics of Software Development Processes

- Semi-automated

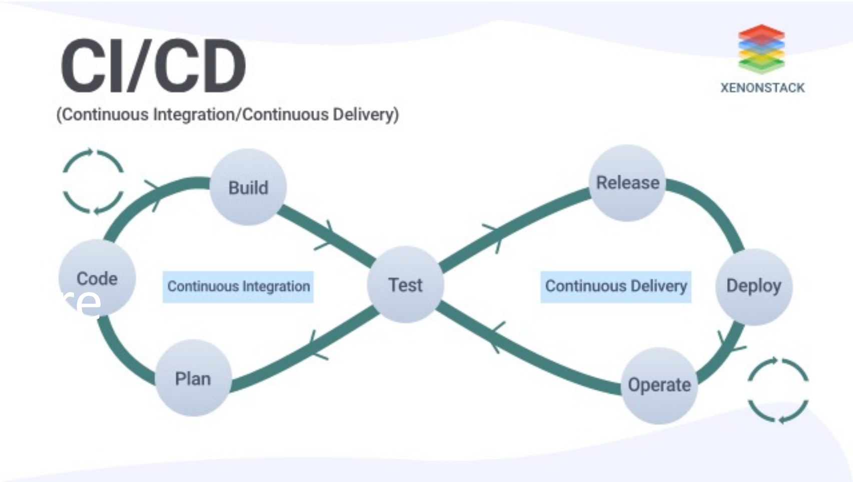


Image by Xenonstack

The Software Engineering Game

Interesting Characteristics of Software Development Processes

- Semi-automated
- Reflective: code review is a common development practice and a promising point to introduce a game mechanism

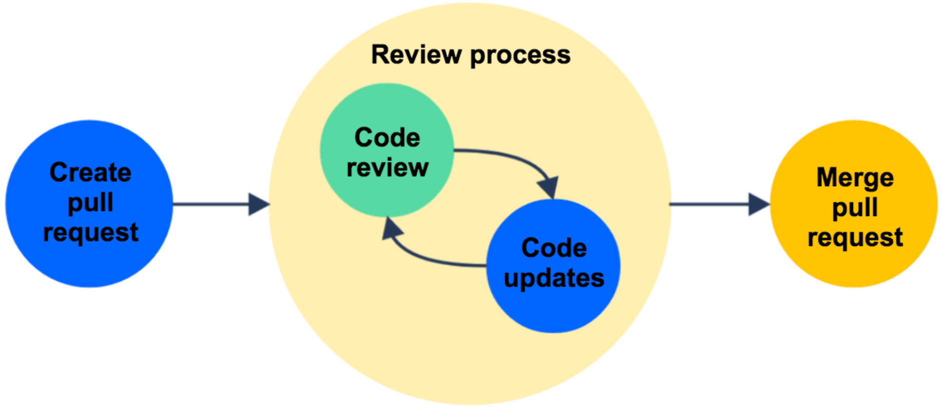


Image from Atlassian

The Software Engineering Game

Interesting Characteristics of Software Development Processes

- Semi-automated
- Reflective: code review is a common development practice and a promising point to introduce a game mechanism
- Structured, interdependent outputs, some of which are executable and amenable to static or dynamic analysis

```
//Load values from database store with channel select
nClient = new bl.desktop
ationClient.Insert();
cationClient.LastRequest = DateTim
cationClient.RequestCount = Notific
ficationClient.Update();
ionClient.Deny == false)
equest NotificationRe
```

Image from Institute of Information Security

The Software Engineering Game

Interesting Characteristics of Software Development Processes

- Semi-automated
- Reflective: code review is a common development practice and a promising point to introduce a game mechanism
- Structured, interdependent outputs, some of which are executable and amenable to static or dynamic analysis
- Fast development cycles will speed evaluating game mechanisms

The Software Engineering Game

Interesting Characteristics of Software Development Processes

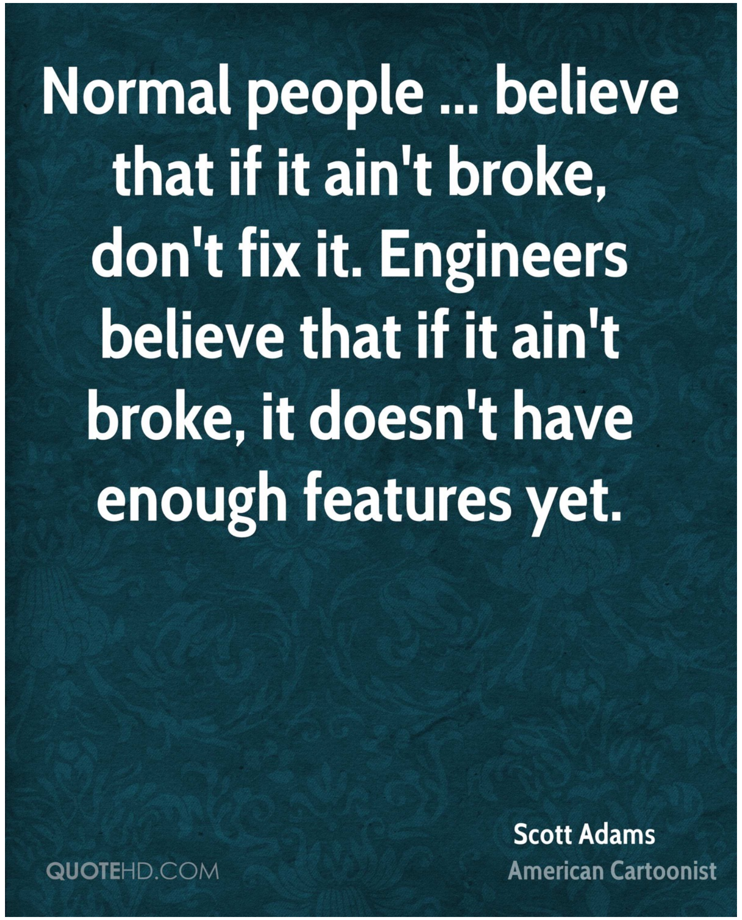
- Semi-automated
- Reflective: code review is a common development practice and a promising point to introduce a game mechanism
- Structured, interdependent outputs, some of which are executable and amenable to static or dynamic analysis
- Fast development cycles will speed evaluating game mechanisms
- ¿More likely to be able persuade a team of developers to adopt a new, designed process than, I imagine, in many other domains?

Software Engineering Games Calling for Mechanism Design

- When to refactor? (aka Overthrowing the “If it ain’t…” Tyrant)
- Lack of documentation
- Poor commenting or commit messaging practice
- Requirements traceability is the seminal problem of connecting requirements to code, yet it is often neglected.
- The tragedy of the test suite
- Program analysis false positives
- Encouraging developers to adopt design by contract

Anticipated Questions Slides

Ok, So, Maybe Some Bosses Have a Point



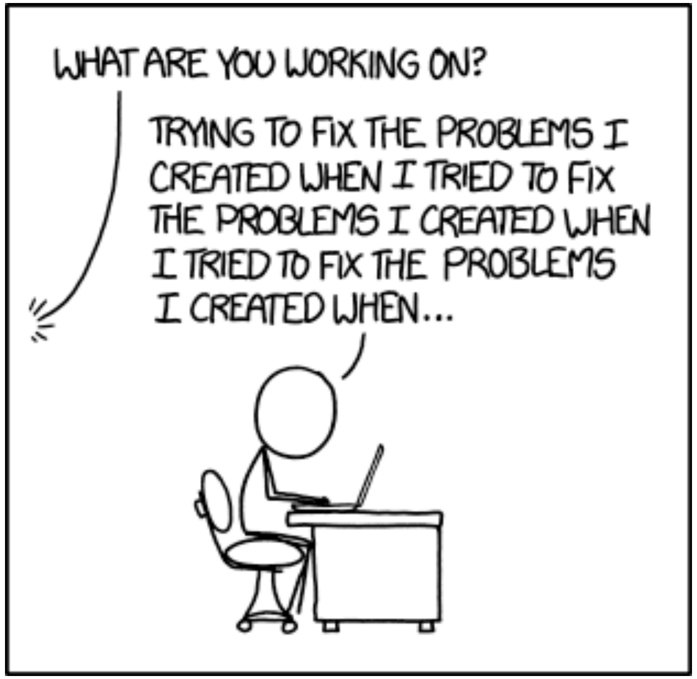
Normal people ... believe that if it ain't broke, don't fix it. Engineers believe that if it ain't broke, it doesn't have enough features yet.

Scott Adams
American Cartoonist

QUOTEHD.COM

Image from QuoteHD.com

Fixing Problems


WHAT ARE YOU WORKING ON?
TRYING TO FIX THE PROBLEMS I CREATED WHEN I TRIED TO FIX THE PROBLEMS I CREATED WHEN I TRIED TO FIX THE PROBLEMS I CREATED WHEN...

Title text: 'What was the original problem you were trying to fix?' 'Well, I noticed one of the tools I was using had an inefficiency that was wasting my time.'