I will make you a better C# programmer
Keeper of Code Edition

Kathleen Dollard

.NET Team

@kathleendollard

kdollard@microsoft.com

Keeper of Production Code

EVERYONE IS THE KEEPER OF A DREAM

OPRAH WINFREY


He did not know that a keeper is only a poacher turned outside in, and a poacher a keeper turned inside out.
QUOTEHD.COM
Charles Kingsley
British Clergyman


AFRICAN JOURNEY
Penguin
passion.

Being a goalkeeper is like being the guy in the military who makes the bombs - one mistake and everyone gets blown up.

Artur Boruc

For a goalkeeper, there is no hiding place

Brad Friedel

As a goalkeeper, you can't come off the bench for 10 minutes and prove your worth - it's either you're in or you're out.

Joe Hart

I couldn't have been a great goalkeeper without power, agility and quickness.

Hope Solo

Agenda

- Intro
- Production Code
- Assessment
- Microsoft Strategy
- Code Organization
- Tests
- Debugging n
- .NET Gotchas
- Summary

Legacy

Ginny Hendry characterized creation of code as a challenge to current coders to create code that is "like other legacies in our lives—like the antiques, heirlooms, and stories that are cherished and lovingly passed down from one generation to the next. What if legacy code was something we took pride in?".

Sumary: https://en.wikipedia.org/wiki/Legacy_code

Original: https://8thlight.com/blog/ginny-hendry/2014/07/11/take-pride-in-your-legacy-code.html

Trying to define legacy code

Trying to define legacy code

- Source code inherited from someone else (Wikipedia)
  - *No, because if you wrote it could still be legacy*
- Source code inherited from an older version of the software (Wikipedia)
  - *No, because there might be one version evolving*
- Code without tests (Michael Feathers)
  - *No, because there is code with tests that is legacy*

https://en.wikipedia.org/wiki/Legacy_code

Approaching code

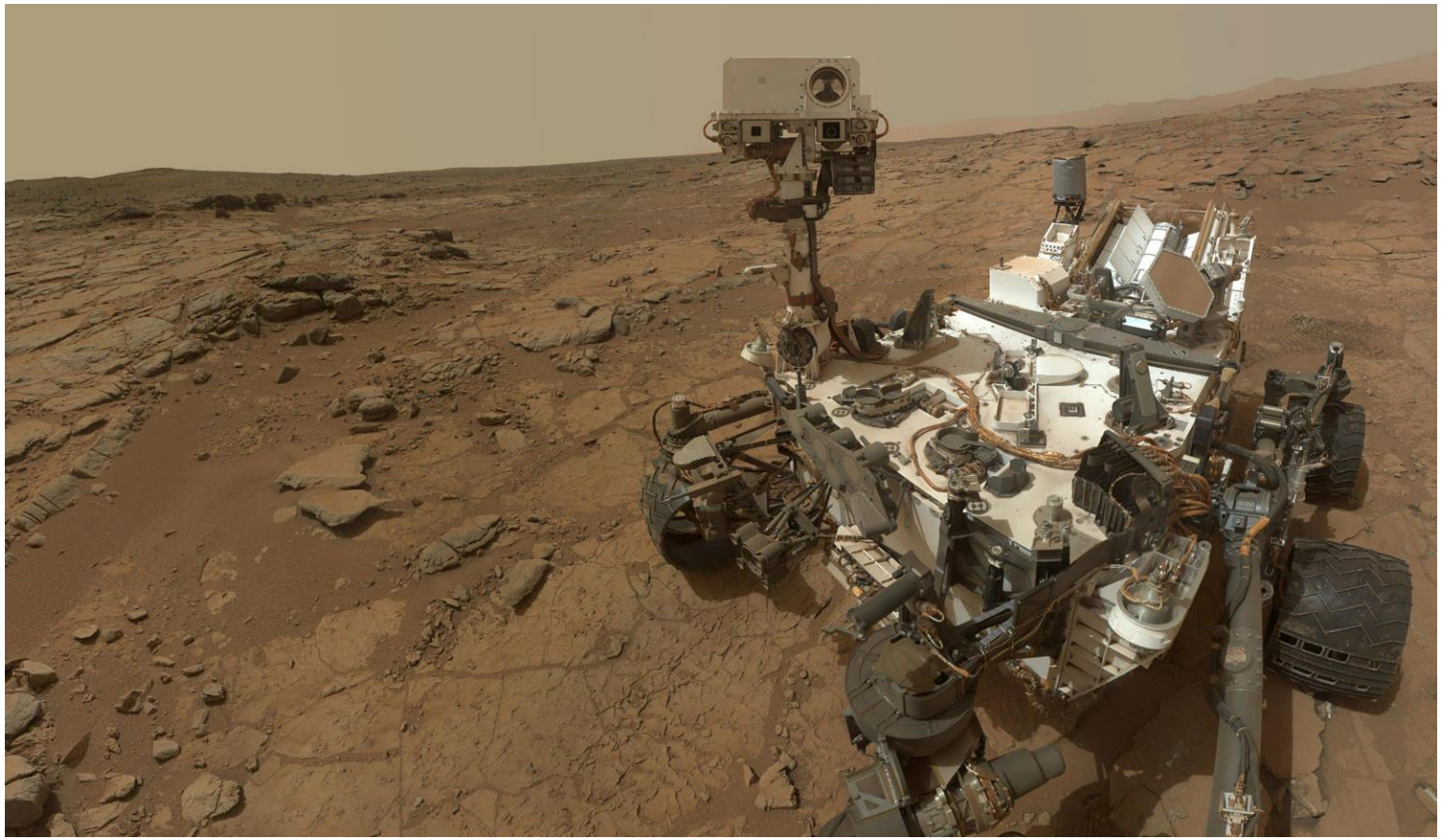
- Test first
- Latest greatest innovations
- Relentless consistency
- Fanatic best practices
- Confidence
- Often isolated systems
- Small data sets
- Minimal debugging
- Architectural discussions

- Fight for every test
- Strategic innovation
- Slow move toward consistency
- Patient practices
- Fear
- Live or near live systems
- Enormous or hidden data sets
- Much time spent debugging
- Accepting existing architecture

For this workshop, legacy code is

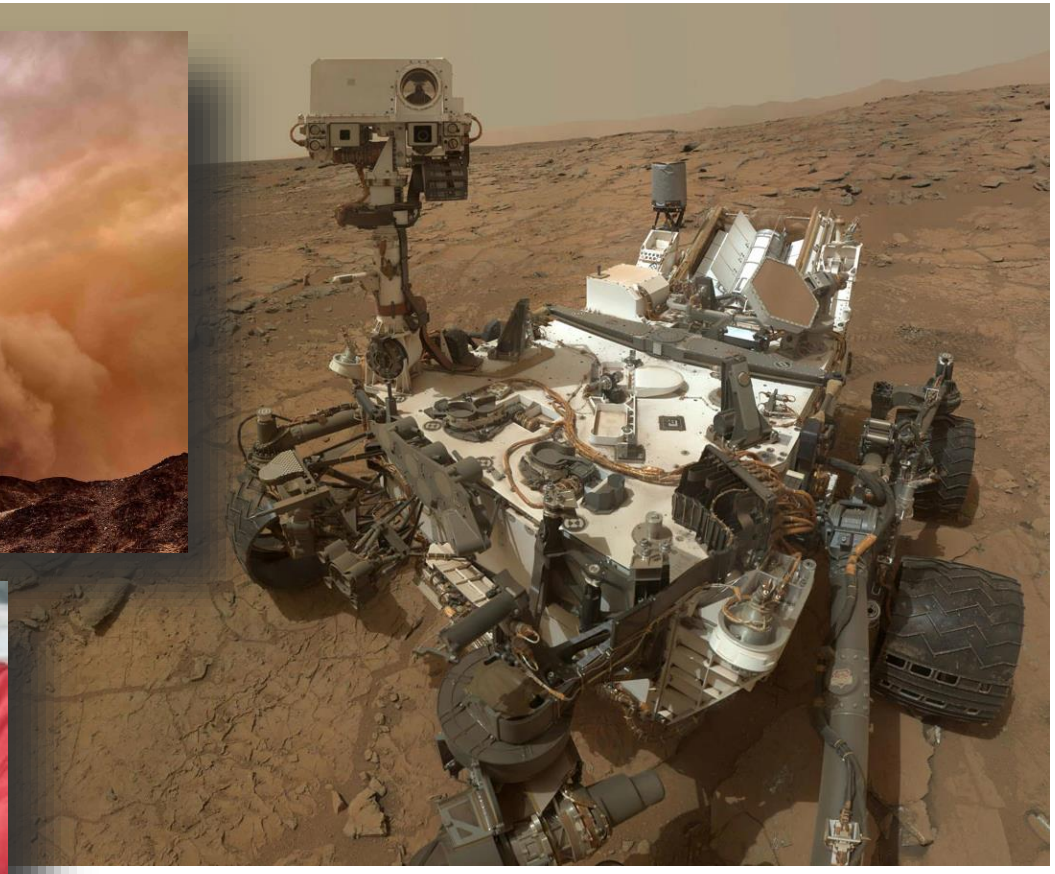
- Any code deployed in production
- Assumed to be useful and important to someone
- Includes unused applications in production environment
- Includes any dead code in production applications

Trying to define legacy code



Do you want this job?

- Old code base written for machines that are now obsolete
- Hardware is 15 years old
- Planned lifecycle - 3 months
- Very slow deployment cycle for any code changes
- When memory started failing due to age, someone added code to ignore memory faults
- No one plans to upgrade
- Even the bandwidth is tiny



Do you want this job?

- Old code base written for machines that are now obsolete
- Hardware is 15 years old
- Planned lifecycle - 3 months
- Very slow deployment cycle for any code changes
- When memory started failing due to age, someone added code to ignore memory faults
- No one plans to upgrade
- Even the bandwidth is tiny

What’s the one thing

we can say with confidence

about legacy code?

What's the one thing
we can say with confidence
about legacy code?

It works!!

Legacy code is…

Code you approach from the point of view
of making changes
instead of designing from a blank sheet

All code that is in production

So how much legacy code is there?

- Your bank
- Your social media
- Your payroll
- Your health records
- Your car

Demo

Picard Tips @PicardTips · Feb 7

Picard management tip: If you succeed too much and become cocky, recognize it, admit it, and stop it.

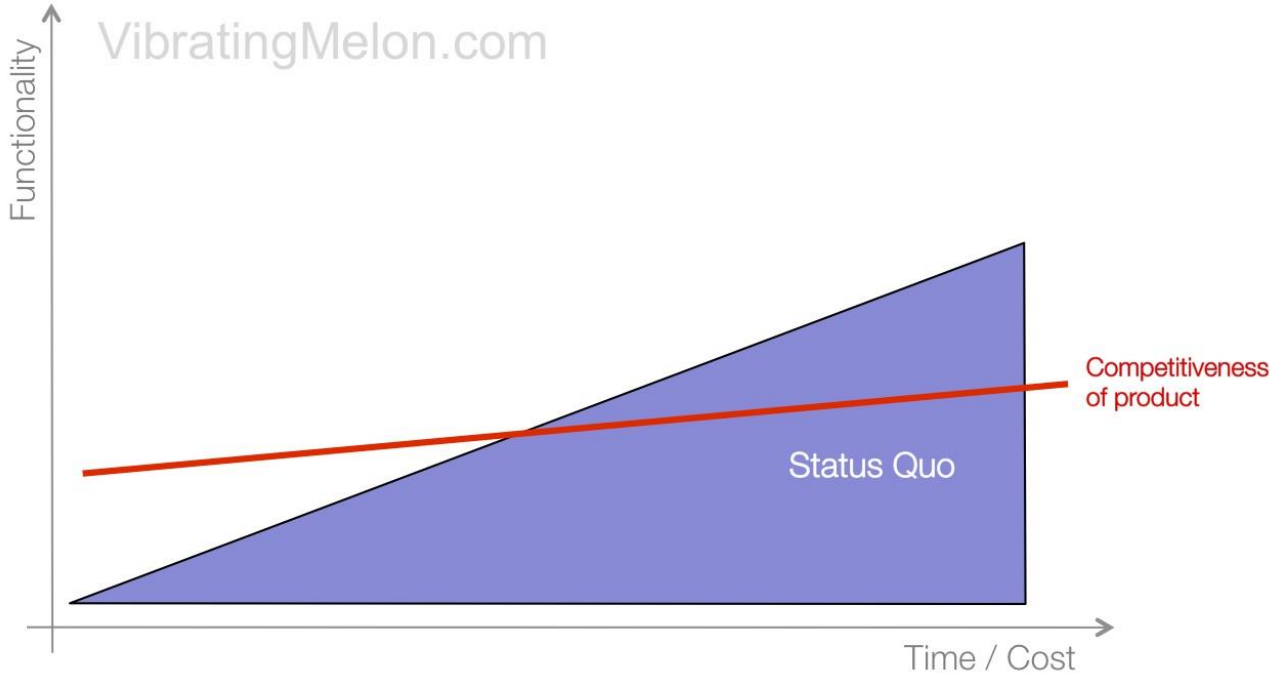
8 277 1.1K

The idea that new code is better than old is patently absurd.
Old code has been *used*.
It has been *tested*.
Lots of bugs have been found, and they've been *fixed*.
There's nothing wrong with it.
It doesn't acquire bugs just by sitting around on your hard drive. …
Back to that two page function.
Yes, I know, it's just a simple function to display a window,
but it has grown little hairs and stuff on it and nobody knows why.
Well, I'll tell you why: those are bug fixes.
One of them fixes that bug that Nancy had when she tried ....
Another one fixes that bug that occurs in low memory conditions.
Another one fixes a floppy disk and the user yanks out the disk in the middle.
That LoadLibrary call is ugly but it makes the code work on old versions of Windows 95.

https://www.joelonsoftware.com/2000/04/06/things-you-should-never-do-part-i/

Continue current development

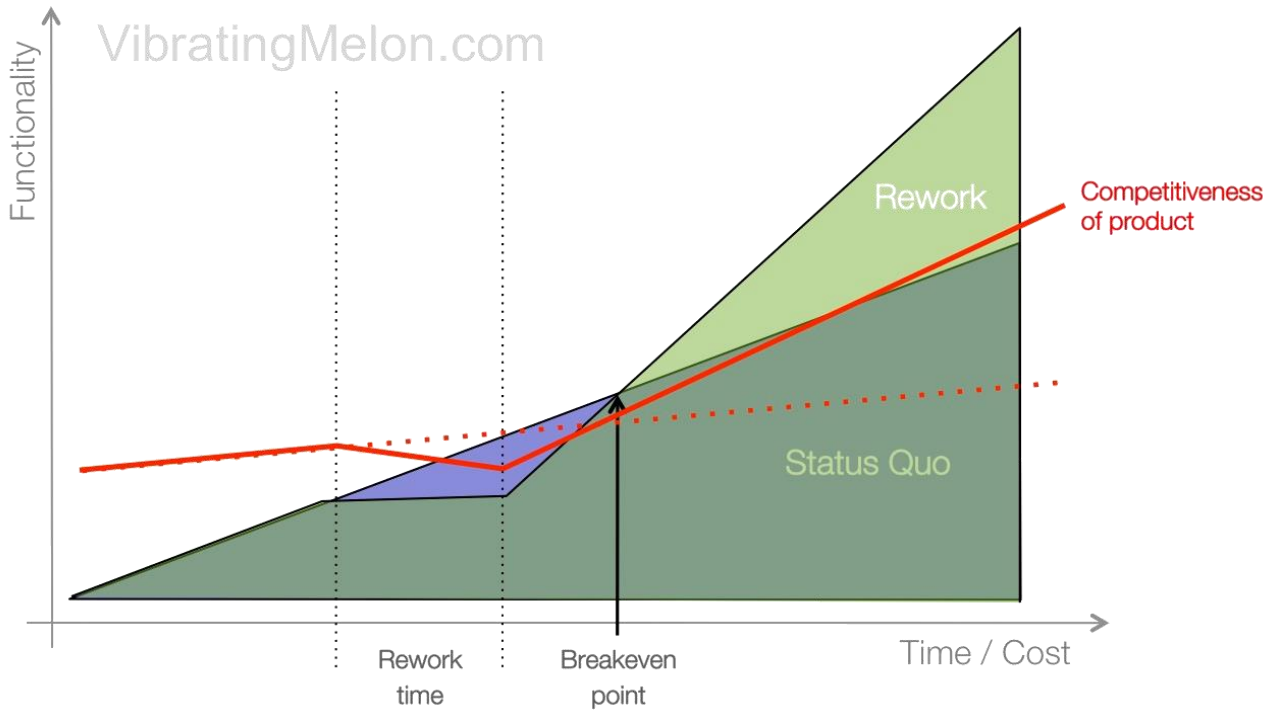
- Competitors are improving also so slopes don’t match



https://vibratingmelon.com/2011/06/10/why-you-should-almost-never-rewrite-code-a-graphical-guide/

Great new approach in theory

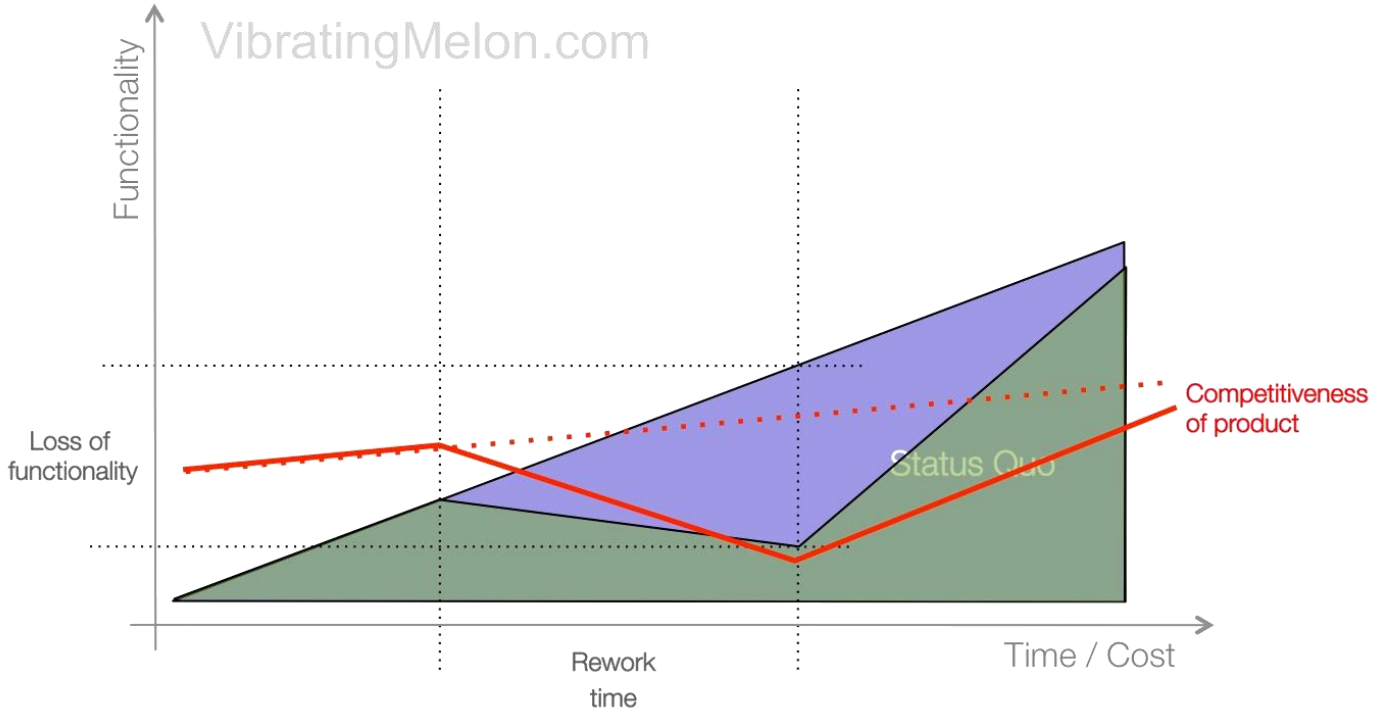
- Period of no new functionality
- Drop in competitiveness, because competitors keep moving
- But then you can move super fast
- So you catch up in competitiveness and functionality
- Then you are off to the races!!!



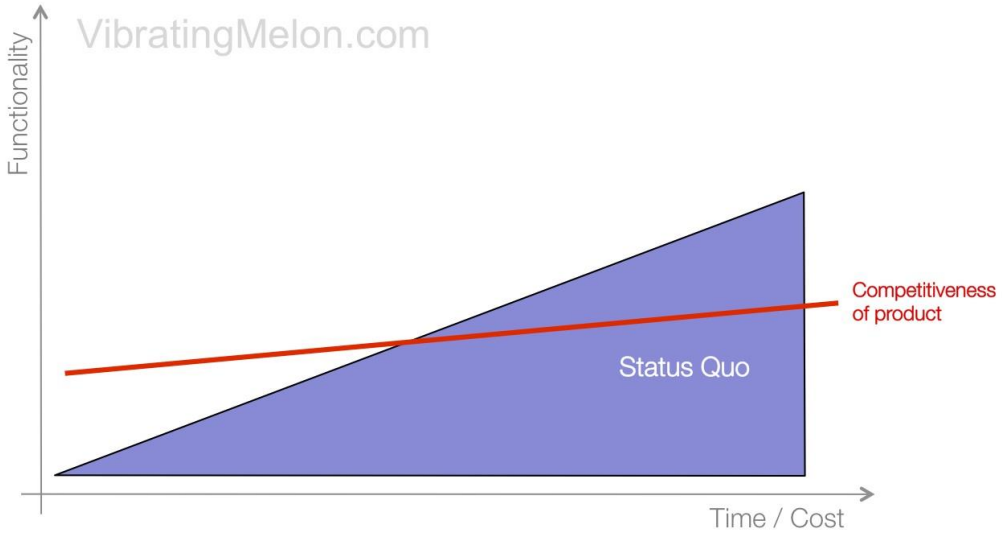
https://vibratingmelon.com/2011/06/10/why-you-should-almost-never-rewrite-code-a-graphical-guide/

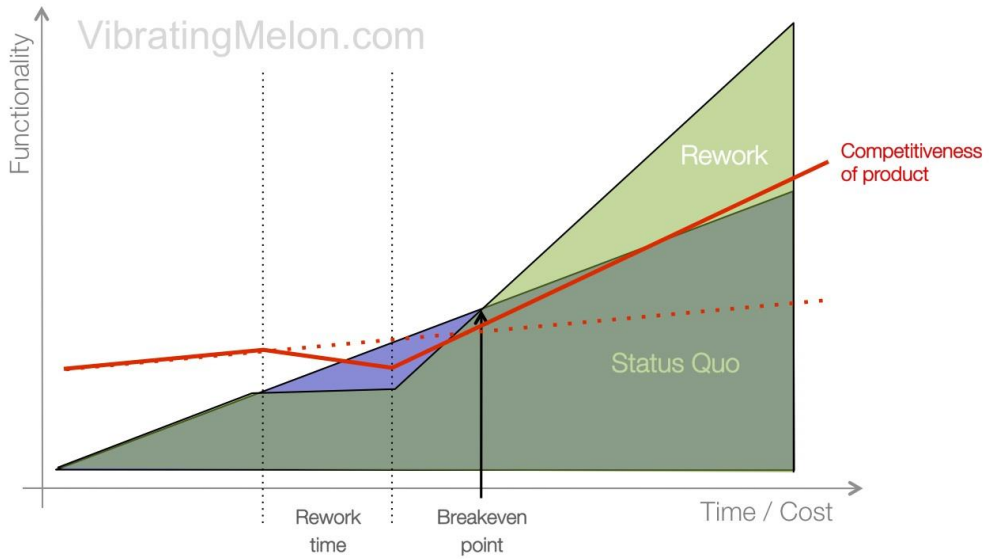
Great new approach reality

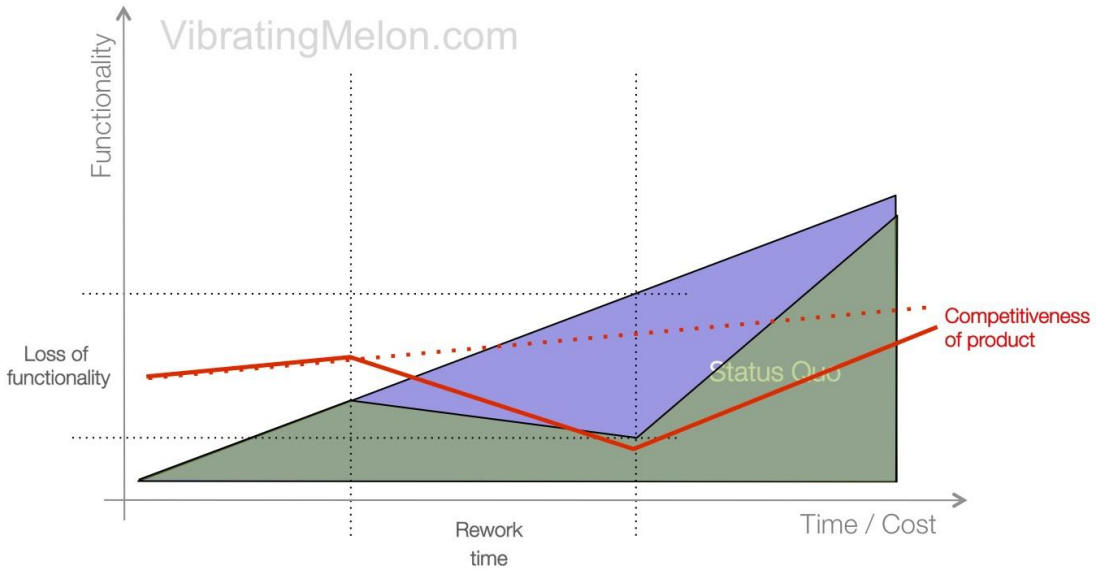
- You actually lose functionality
- It takes way longer than expected
- Competitors don’t wait
- It’s very hard to go actually go faster



https://vibratingmelon.com/2011/06/10/why-you-should-almost-never-rewrite-code-a-graphical-guide/







https://vibratingmelon.com/2011/06/10/why-you-should-almost-never-rewrite-code-a-graphical-guide/

Is it easier to fix or rewrite code?

But my code is so bad, I *have* to rewrite…

- Netscape 6
- Borland Arago and CA Clipper (dBase for Windows)
- Borland QuatroPro

…and so on

If your code is bad, fix it. One step at a time.

(from Joel Spolsky's article)

Netscape 6.0 is finally going into its first public beta. There never was a version 5.0. The last major release, version 4.0, was released almost three years ago. Three years is an *awfully* long time in the Internet world. During this time, Netscape sat by, helplessly, as their market share plummeted.

It's a bit smarmy of me to criticize them for waiting so long between releases. They didn't do it *on purpose*, now, did they?

https://www.joelonsoftware.com/2000/04/06/things-you-should-never-do-part-i/

Netscape 6.0 is finally going into its first public beta. There never was a version 5.0. The last major release, version 4.0, was released almost three years ago. Three years is an *awfully* long time in the Internet world. During this time, Netscape sat by, helplessly, as their market share plummeted.

It's a bit smarmy of me to criticize them for waiting so long between releases. They didn't do it *on purpose*, now, did they?

Well, yes. They did.

They did it by making the **single worst strategic mistake** that any software company can make:

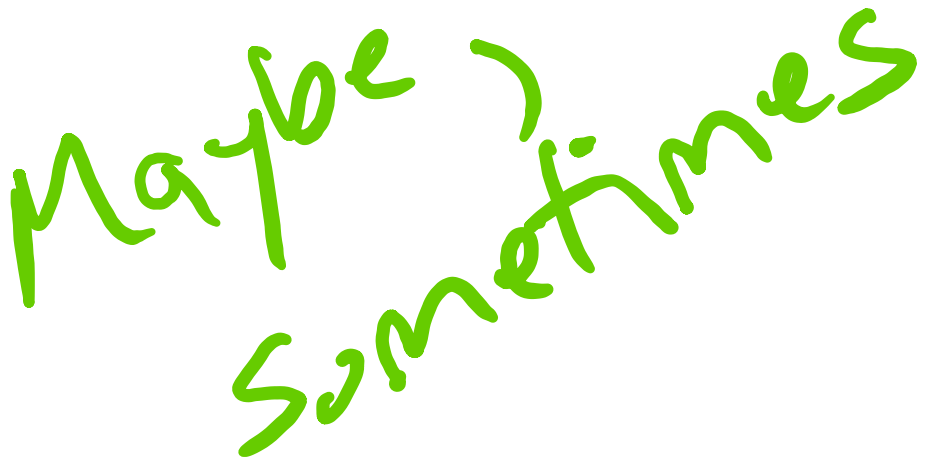
They decided to rewrite the code from scratch.

https://www.joelonsoftware.com/2000/04/06/things-you-should-never-do-part-i/

We’re programmers.

Programmers are, in their hearts, architects, and the first thing they want to do when they get to a site is to bulldoze the place flat and build something grand.

We’re not excited by incremental renovation: tinkering, improving, planting flower beds.



https://www.joelonsoftware.com/2000/04/06/things-you-should-never-do-part-i/

There’s a subtle reason that programmers always want to throw away the code and start over.

The reason is that they think the old code is a mess.

And here is the interesting observation: *they are probably wrong.*

The reason that they think the old code is a mess is because of a cardinal, fundamental law of programming:

It’s harder to read code than to write it.

This is why code reuse is so hard.

https://www.joelonsoftware.com/2000/04/06/things-you-should-never-do-part-i/

Sure,

But mostly programmers think they can do a BETTER job.

Why?

Our project is different

It’s so messy and ugly!

It’s so awful that we can’t possibly fix it

No one understands it

We are smarter than we were last year

It is painful to work on

We can make it faster

There’s new tech/approach

We have a new team/consultant

Back to Joel

…there are architectural problems…These problems can be solved, one at a time, by carefully moving code, refactoring, changing interfaces...Even fairly major architectural changes can be done without *throwing away the code*.

What can we say with confidence about legacy code?

It works!!

What can we say with confidence about legacy code?

It works!!

No one knows everything it does

What’s the worst thing lurking in your code?

Zombie code

Kevlin Henny https://www.infoq.com/news/2017/02/dead-code

Re: Knight CapitalGroup

This bankruptcy-defining event arose from a perfect storm. In anticipation of a new NYSE system, to be launched on the 1st of August, they had deployed updates to their servers. They updated their servers manually and, unbeknown to them, one of the deployments failed, leaving the old version running. To take advantage of the new NYSE system, they recycled an old flag, a flag that was no longer used but had now been repurposed to mean something different. Although it hadn’t been used in eight years, the old version of the code still had a dependency on the old flag.

The code had been dead for years, but was awakened by a change to the flag’s value. The zombie apocalypse arrived and the rest is bankruptcy.

Zombie code

Kevlin Henny https://www.infoq.com/news/2017/02/dead-code

Re: Knight CapitalGroup – 450 million in 45 minutes

This bankruptcy-defining event arose from a perfect storm. In anticipation of a new NYSE system, to be launched on the 1st of August, they had deployed updates to their servers. They **updated their servers** manually and, unbeknown to them, **one of the deployments failed, leaving the old version running**. To take advantage of the new NYSE system, they recycled an old flag, a flag that was no longer used but had now been repurposed to mean something different. Although it hadn't been used in eight years, the old version of the code still had a dependency on the old flag.

The code had been dead for years, but was awakened by a change to the flag's value. The zombie apocalypse arrived and the rest is bankruptcy.

Zombie code

Kevlin Henny https://www.infoq.com/news/2017/02/dead-code

Re: Knight CapitalGroup – 450 million in 45 minutes

This bankruptcy-defining event arose from a perfect storm. In anticipation of a new NYSE system, to be launched on the 1st of August, they had deployed updates to their servers. They **updated their servers** manually and, unbeknown to them, **one of the deployments failed, leaving the old version running**. To take advantage of the new NYSE system, they **recycled an old flag**, a flag that was no longer used but had now been repurposed to mean something different. **Although it hadn't been used in eight years, the old version of the code still had a dependency on the old flag**.

The code had been dead for years, but was awakened by a change to the flag's value. The zombie apocalypse arrived and the rest is bankruptcy.

Zombie code

Kevlin Henny https://www.infoq.com/news/2017/02/dead-code

Re: Knight CapitalGroup – 450 million in 45 minutes

This bankruptcy-defining event arose from a perfect storm. In anticipation of a new NYSE system, to be launched on the 1st of August, they had deployed updates to their servers. They **updated their servers** manually and, unbeknown to them, **one of the deployments failed, leaving the old version running**. To take advantage of the new NYSE system, they **recycled an old flag**, a flag that was no longer used but had now been repurposed to mean something different. **Although it hadn't been used in eight years, the old version of the code still had a dependency on the old flag**.

The code had been dead for years, but was awakened by a change to the flag's value. The zombie apocalypse arrived and the rest is bankruptcy.

What else?

By Carol M. Highsmith - Library of CongressCatalog: http://lccn.loc.gov/2011630866Image download: https://cdn.loc.gov/master/pnp/highsm/12600/12672a.tifOriginal url: http://hdl.loc.gov/loc.pnp/highsm.12672, Public Domain, https://commons.wikimedia.org/w/index.php?curid=52220198

Contraption Code

James Gleick https://www.around.com/ariane.html

“But in this case, the programmers had decided that this particular velocity figure would never be large enough to cause trouble. After all, it never had been before. Unluckily, Ariane 5 was a faster rocket than Ariane 4. One extra absurdity: the calculation containing the bug, which shut down the guidance system, which confused the on-board computer, which forced the rocket off course, actually served no purpose once the rocket was in the air. Its only function was to align the system before launch. So it should have been turned off. But engineers chose long ago, in an earlier version of the Ariane, to leave this function running for the first 40 seconds of flight -- a "special feature" meant to make it easy to restart the system in the event of a brief hold in the countdown.”

Contraption Code

James Gleick https://www.around.com/ariane.html

“But in this case, the programmers had **decided that this particular velocity figure would never be large enough to cause trouble**. After all, it never had been before. Unluckily, Ariane 5 was a faster rocket than Ariane 4. One extra absurdity: the calculation containing the bug, which shut down the guidance system, which confused the on-board computer, which forced the rocket off course, actually served no purpose once the rocket was in the air. Its only function was to align the system before launch. So it should have been turned off. But engineers chose long ago, in an earlier version of the Ariane, to leave this function running for the first 40 seconds of flight -- a "special feature" meant to make it easy to restart the system in the event of a brief hold in the countdown.”

Contraption Code

James Gleick https://www.around.com/ariane.html

“But in this case, the programmers had **decided that this particular velocity figure would never be large enough to cause trouble**. After all, it never had been before. Unluckily, **Ariane 5 was a faster rocket** than Ariane 4. One extra absurdity: the calculation containing the bug, which shut down the guidance system, which confused the on-board computer, which forced the rocket off course, actually served no purpose once the rocket was in the air. Its only function was to align the system before launch. So it should have been turned off. But engineers chose long ago, in an earlier version of the Ariane, to leave this function running for the first 40 seconds of flight -- a "special feature" meant to make it easy to restart the system in the event of a brief hold in the countdown.”

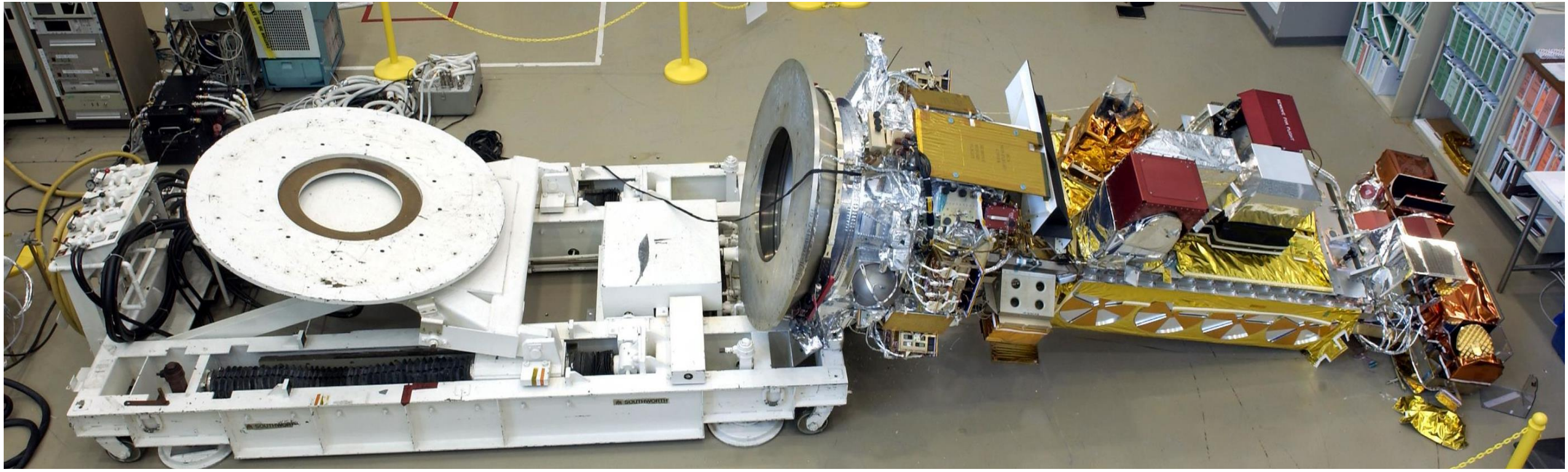
Contraption Code

James Gleick https://www.around.com/ariane.html

“But in this case, the programmers had **decided that this particular velocity figure would never be large enough to cause trouble**. After all, it never had been before. Unluckily, **Ariane 5 was a faster rocket** than Ariane 4. One extra absurdity: the calculation containing the bug, which shut down the guidance system, which confused the on-board computer, which forced the rocket off course, **actually served no purpose once the rocket was in the air**. Its only function was to align the system before launch. **So it should have been turned off**. But engineers chose long ago, in an earlier version of the Ariane, to leave this function running for the first 40 seconds of flight -- **a "special feature" meant to make it easy to restart the system in the event of a brief hold in the countdown.**”

Sloppiness

- $135M
- 24 bolts



Solution is not to rewrite the code

- Respect the code and the coders
- Add basics that support stability, like testing and tracing
- Fix bugs thoughtfully
- Be vigilant on dead, spurious, unused and redundant code

$135M
24 bolts

Should you ever rewrite code?

Keep current

- Constant evolution extends lifetime
- Move to modern tech makes sense
- Architecture separates concerns

Static support

- Little change now or expected
- Technology is stable, if old
- Focus refactoring on separating concerns

Plan to abandon

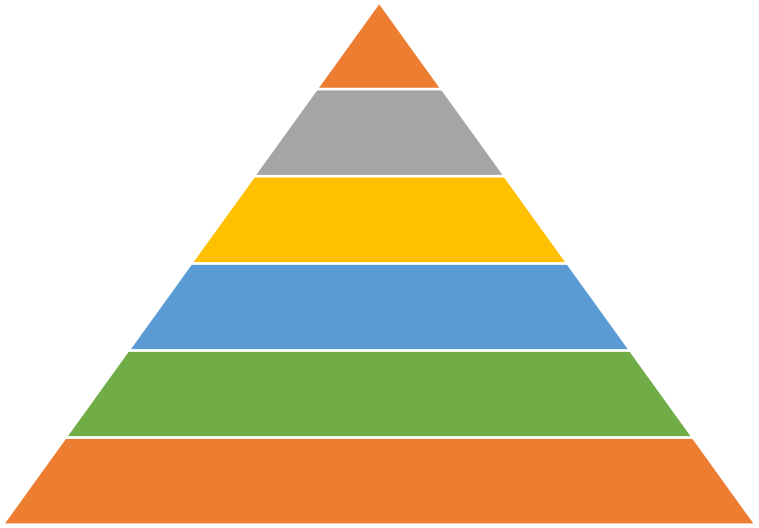
- Technology is out of support
- Large percent of app affected by business change (ie. GDPR)
- Try to save business logic

Rewriting code
and not shutting down the previous system
hides a ton of zombie code,
results in hybrid contraptions, and
risks pulling the bolts out

Working on production apps

- It’s real – issues and changes effect real people and companies
- Peeling an onion instead of planting a garden
  - You don’t have to wait to see outcome
  - It will take a long time to get to the heart of it
  - You might cry
- The wires are live – you’re working on a live system
- There’s generally not a net – tests do not deserve confidence

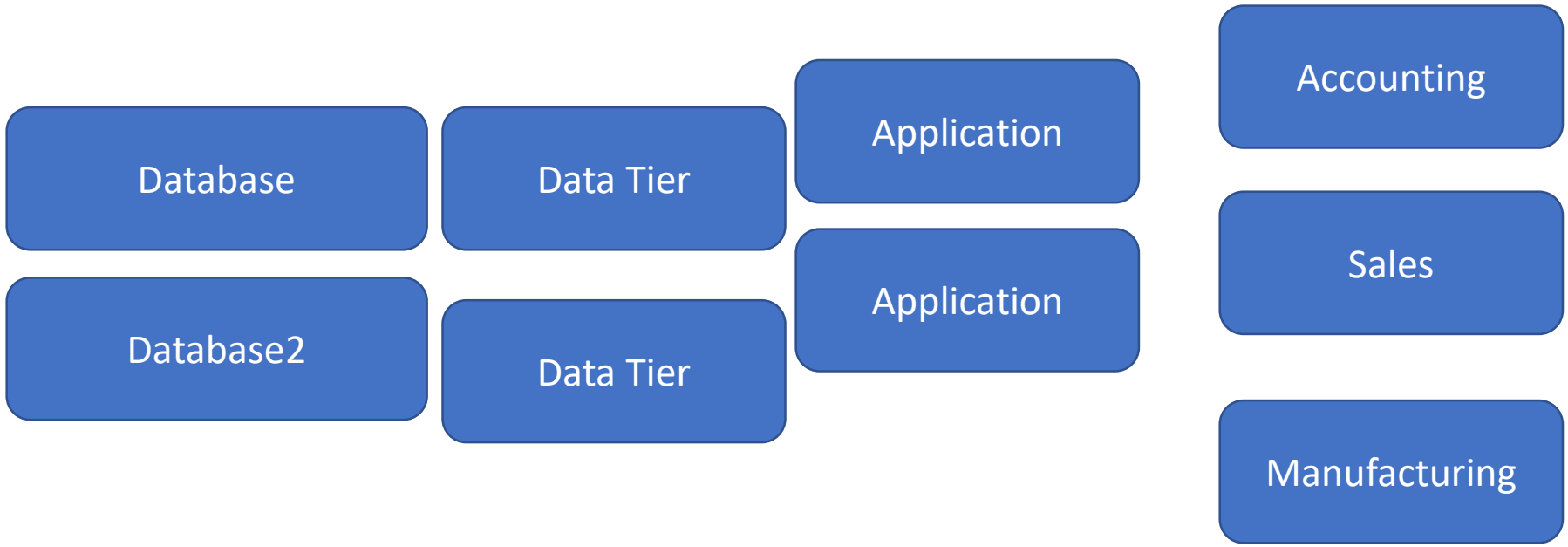
Order of need



1. Access to code and other artifacts
   - It will not get easier
2. Ability to build and deploy
   - Automated build
3. Source control
4. Regression test plan
   - Even specific manual tests
5. Health tracking/tracing/telemetry
6. Bug tracking
7. Cost assessment, long term planning

Identify the parts…

- What pieces deploy onto **different machin**es?
- What pieces deploy as **separate assemblies**?
- What communication **protocols** are used?
- What **exchange mechanisms** are used?


Database
Database2
Data Tier
Data Tier
Application
Application
Accounting
Sales
Manufacturing

Look for key areas…

- Where is the **user**?
- Where is the **data**?
- Where is there data **input or imported data**?
- Is data **exported**?
- Where are the **business rules**?

Per identifiable part of the system

- Is the **hardware** supported/available/practical?
- Is the **platform** supported/available/practical?
- Once in your user's hands, is it's **web/desktop** approach reasonable?
- Does it have **legal issues** like GDPR or HIPAA?
- Can it be **secured**?

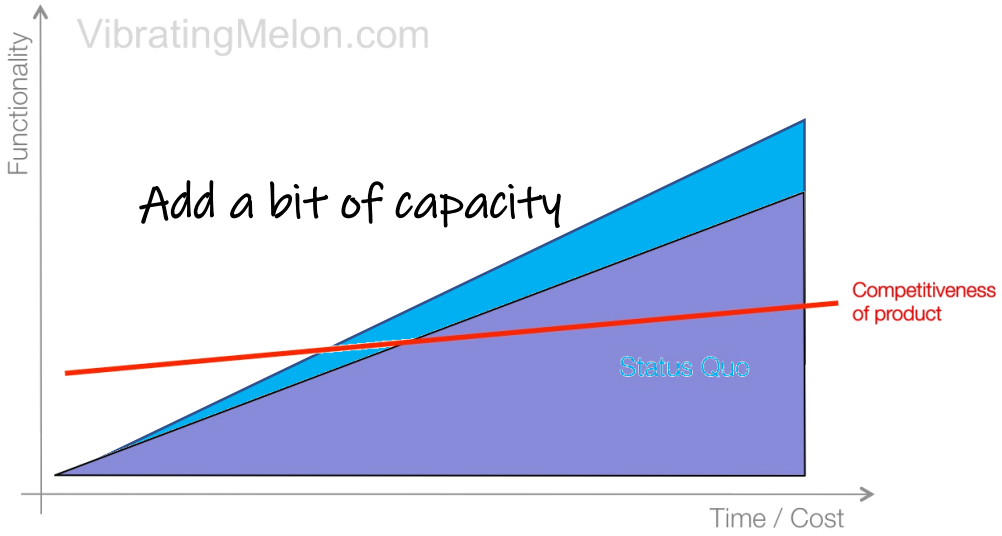
The Big Stuff

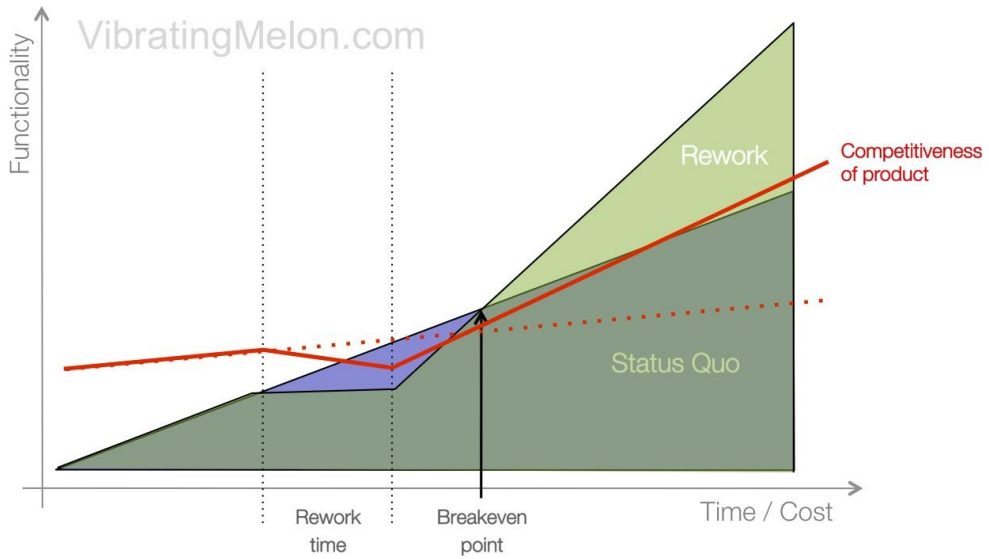
- Security
- GDPR
- True obsolescence

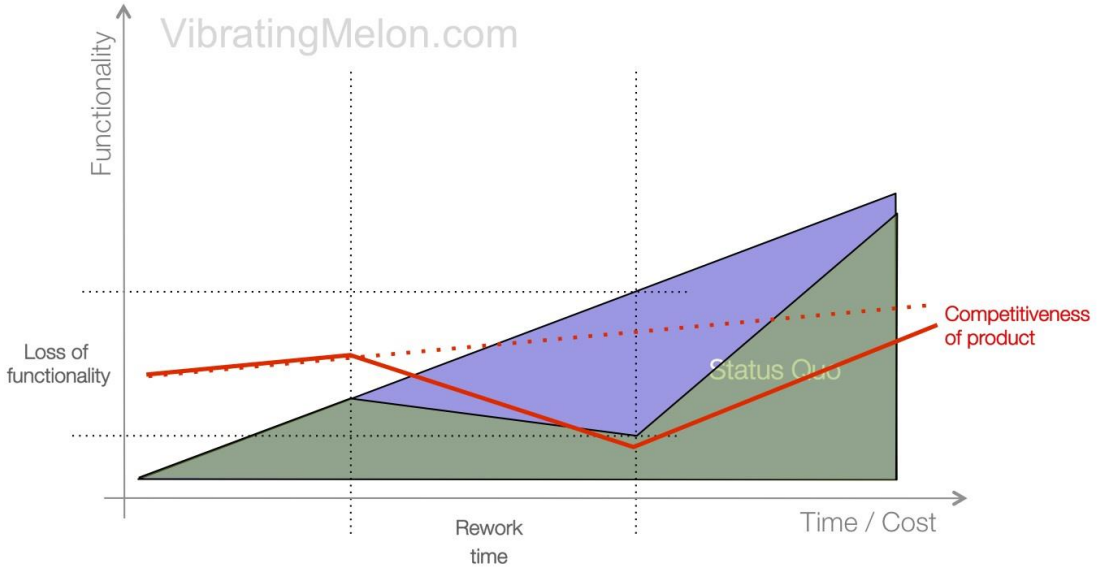
- Maybe big stuff
  - License or maintenance cost
  - Really can’t get programmers
  - Real changes in business

Look back at the your system diagram

- Update diagram due to deeper assessment
- Identify areas with problems
- Split out (real) problem code
- Identify most valuable changes architecturally (6-12)
- Pick one







https://vibratingmelon.com/2011/06/10/why-you-should-almost-never-rewrite-code-a-graphical-guide/

Break

What can we say with confidence about legacy code?

It works!!

No one knows everything it does

What can we say with confidence about legacy code?

It works!!

No one knows everything it does

You can make it better

Skills

- Test
- Debug
- Refactor

Demo

Movie database

Demo

John’s Social Club

Michael Feathers

- ***Working Effectively with Legacy Code, 2004***
- "Legacy Code is code without tests"
  - From Kathleen: FWIW, I disagree with this. I've seen plenty of systems where tests did not remove the "legacy-ness" of the system

r7k

Orange Code

What humble citrus fruit can tell us about software

Michael Feathers

August 28, 2018

Whenever I work on unfamiliar code I start extracting methods. When I do this, I look for chunks of code that I can name - then I extract. Even if I end up inlining the methods I've extracted later, at least I have a way of temporarily hiding detail so that I can see the overall structure.

When I'm working with someone, often they point out thatI'm adding more code when I do these extractions. In terms of line count, they are right. An expression with 10 tokens on one line becomes three lines in most languages. There's a line for the declaration, a line for the expression, and a line for the syntax that terminates a function definition.

```
public void run() {
    adapter.tier().run();
}
```

A few languages spare us that extra line at the end to complete functions. Python is one of them.

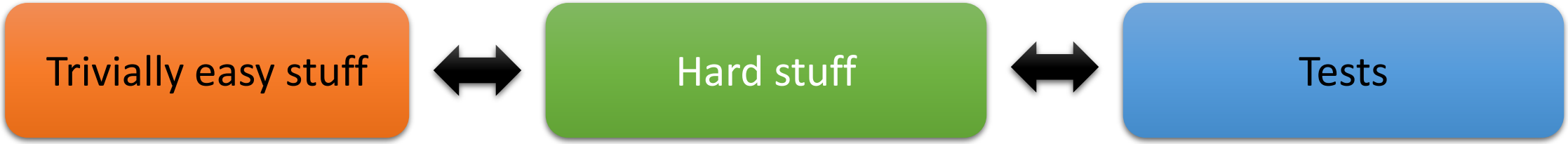
```
@given(st.lists(st.integers()))
def test_reversing(self, xs):
    ys = list(xs)
    ys.reverse()
    ys.reverse()
    assert xs == ys
```

So, by extracting methods, we add more code. Is that bad?

Puzzles!

Testing

Architecture for testing



Test Driven Debugging

Test Driven Debugging

- A bug in the wild is always an indication of two bugs
  1. The bug itself
  2. The bug in testing that allowed it to get into the wild
- Fix the second bug first
  - Write a failing test to reproduce the problem
- Fix the bug, prove it with a test

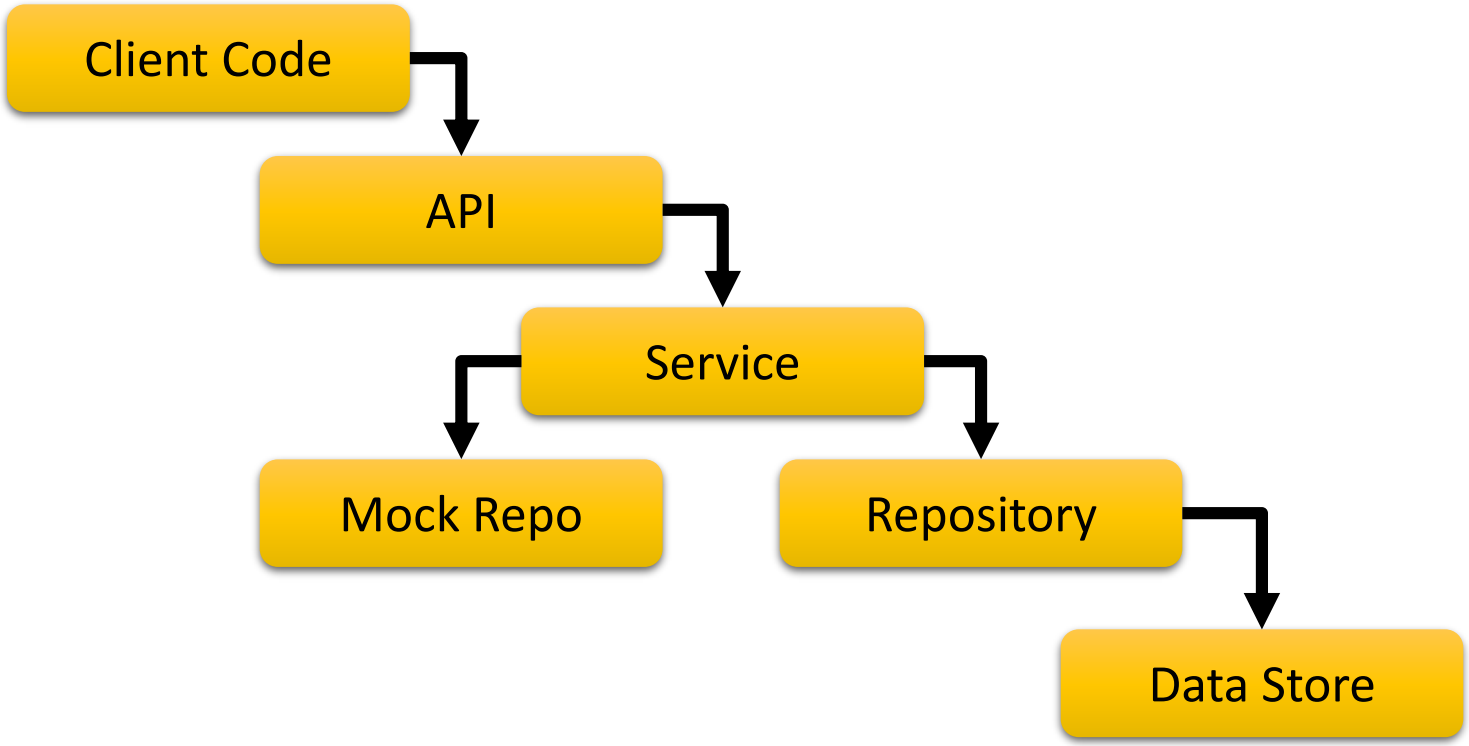
This is hard in an increasingly common set of problems that stem from configuration, bad dependencies, and other environment issues

- Unit tests provide value as…
  - Regression (debugging)
  - Developers to know they are done (lots of short loop debugging during development)
  - Confirmation that everything's right in the dev workflow (build servers)
  - Quality, maybe
- Seams must be found or created to allow unit testing

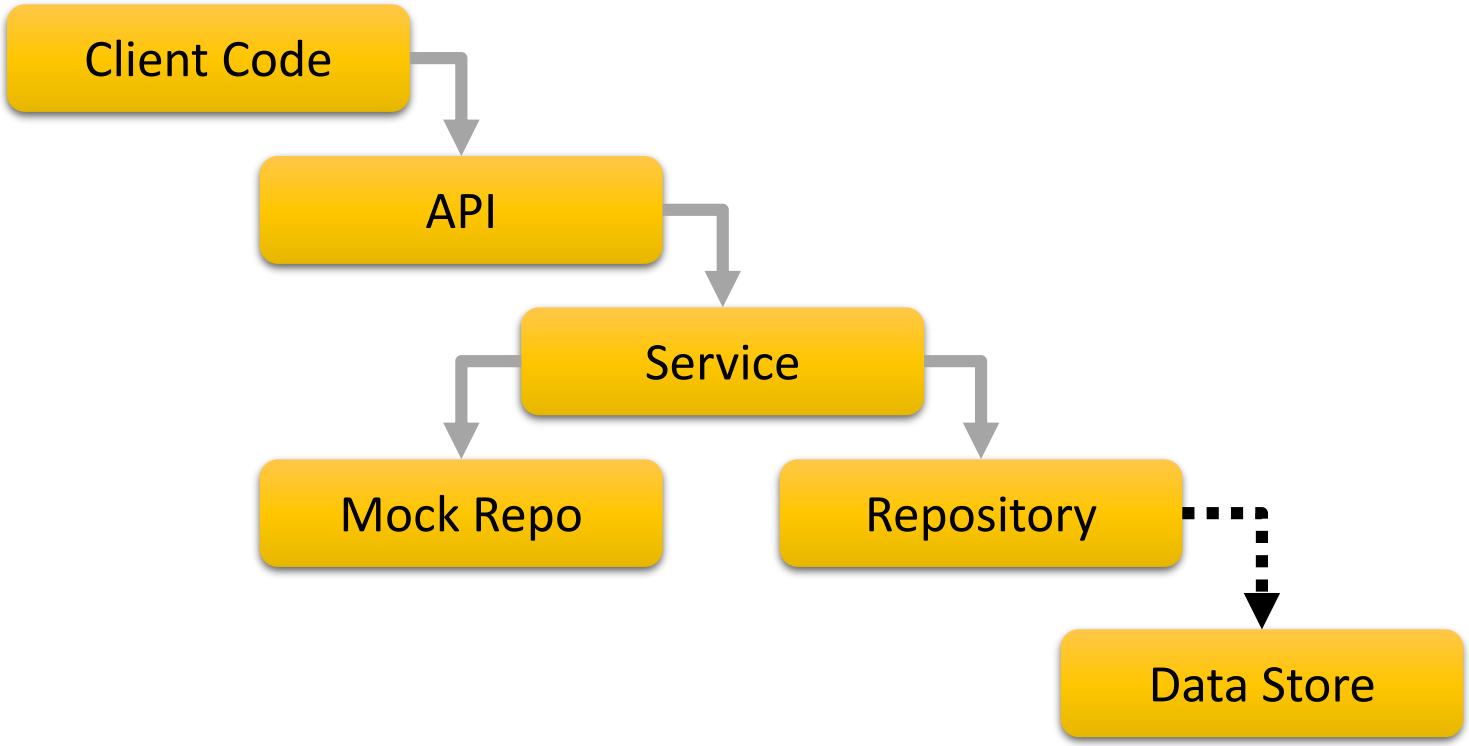
Seam

A seam is a place where you can alter behavior in your program without editing in that place.

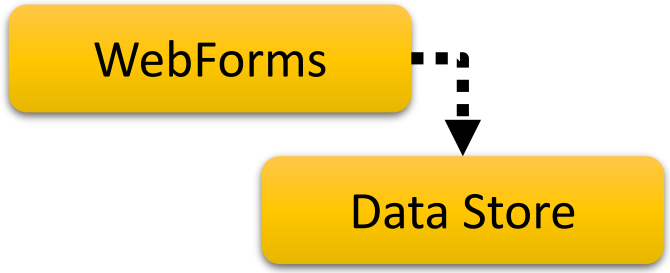
Seams in a Service Driven App



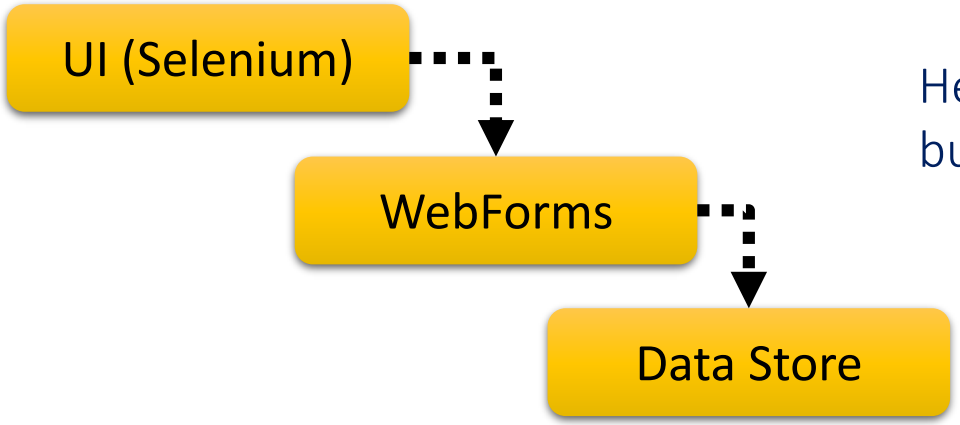
Seams in a Service Driven App



Legacy WebForms App

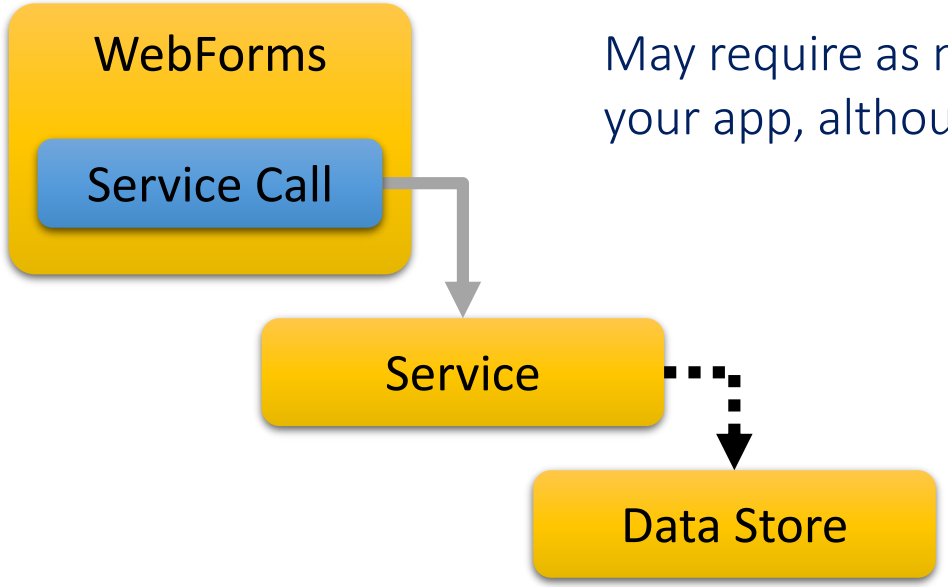


Legacy WebForms App



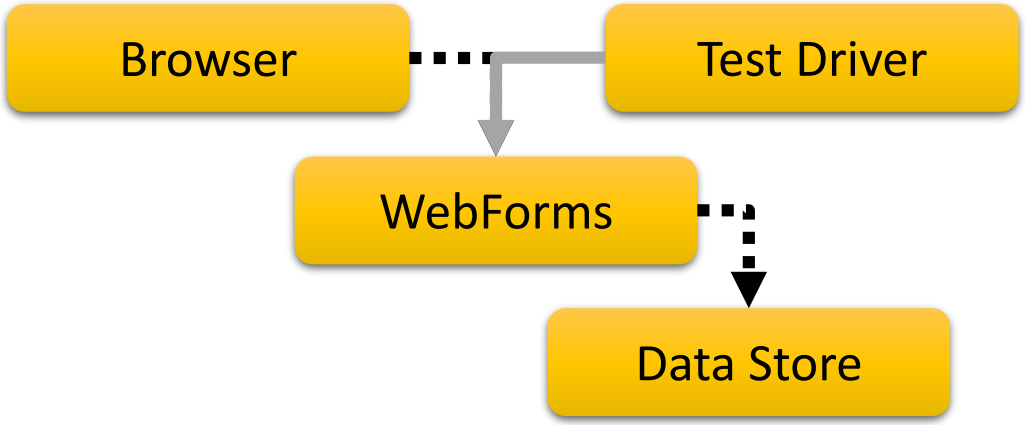
Helps quality, particular smoke,
but too slow to help developers

Legacy WebForms App



May require as much effort as rewriting your app, although lower risk

Legacy WebForms App

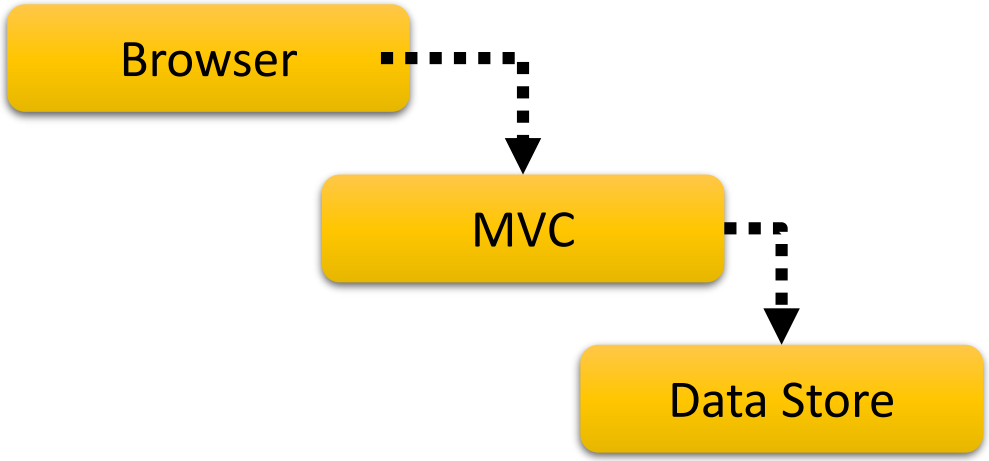


Sets up the WebForms environment and issues calls on the test behalf

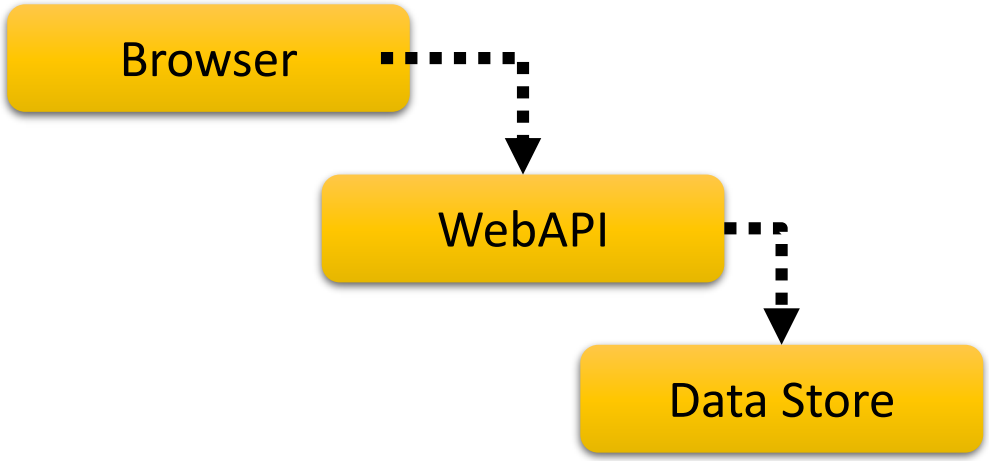
WebFormsTest

https://github.com/csharpfritz/WebFormsTest/issues

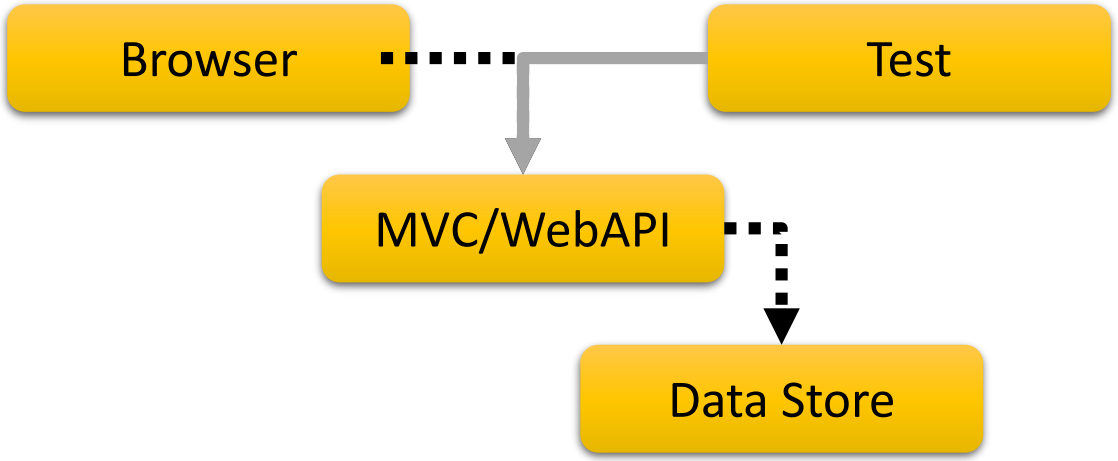
Legacy MVC App



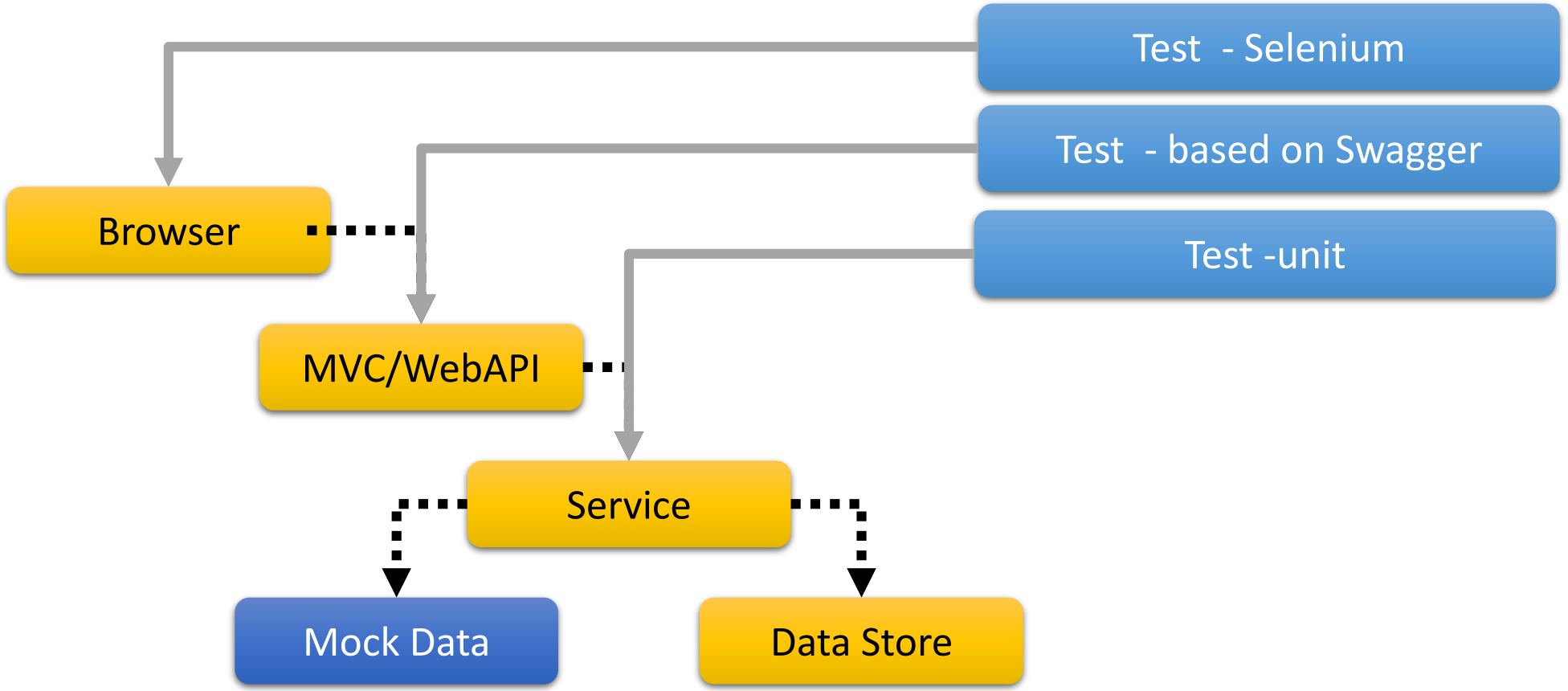
Legacy MVC App



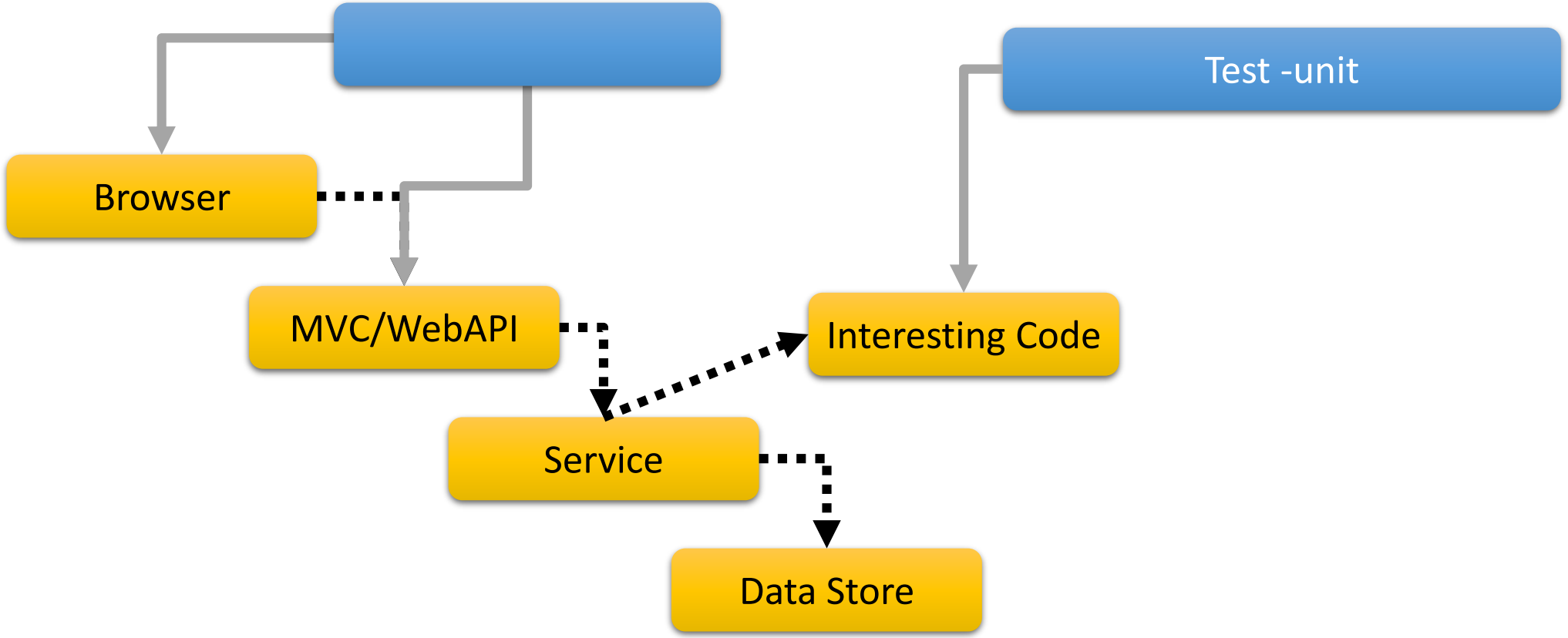
Legacy MVC/WebAPI App



Testable MVC/WebAPI App



Isolate interesting parts for testing



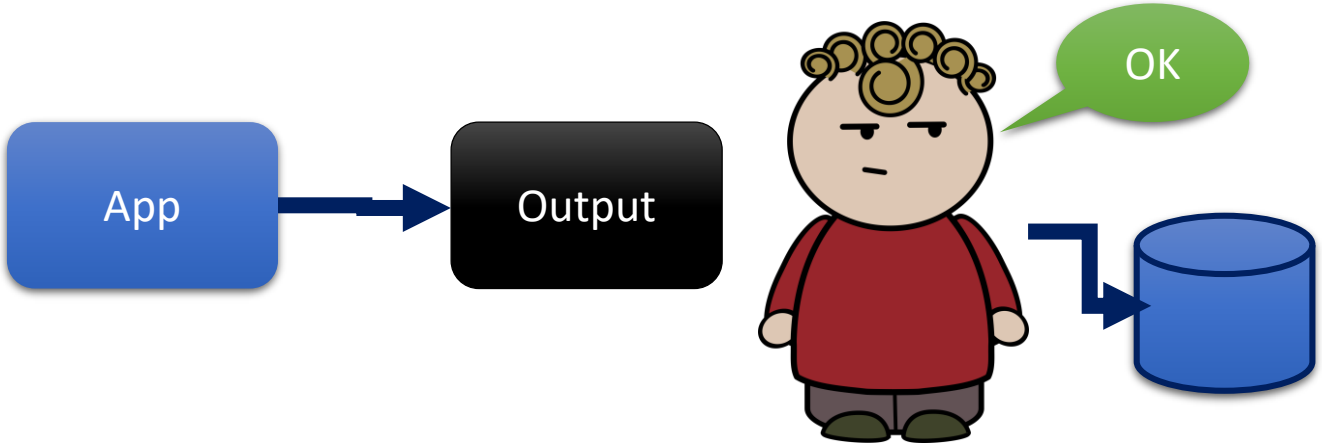
Approval Tests

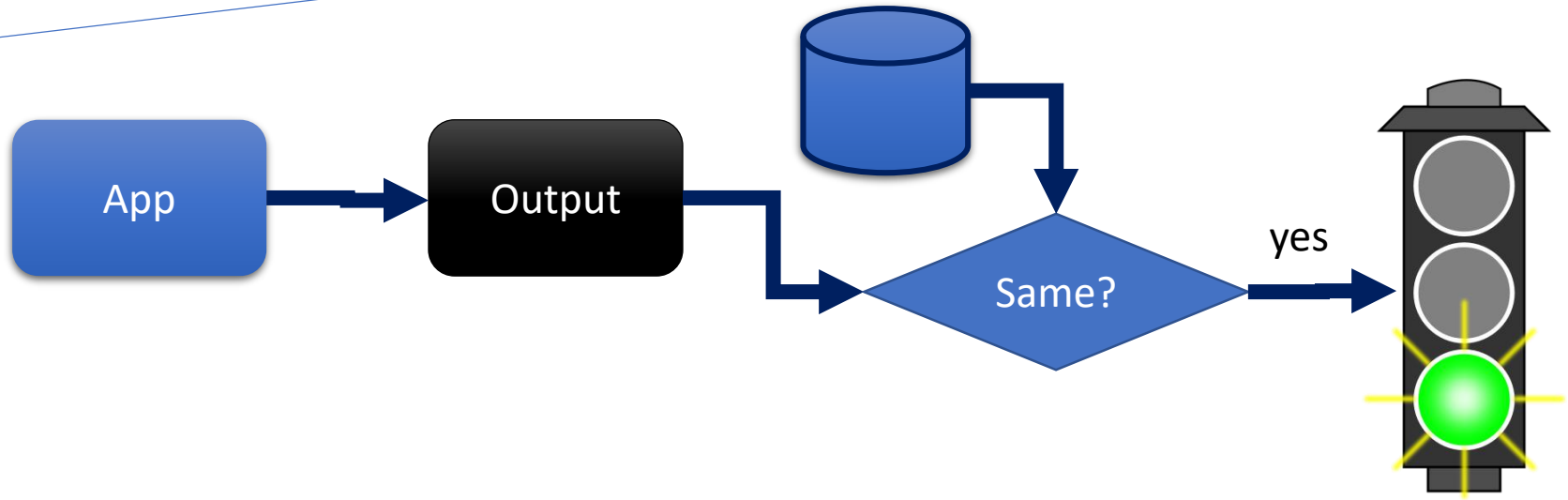
http://approvaltests.com/

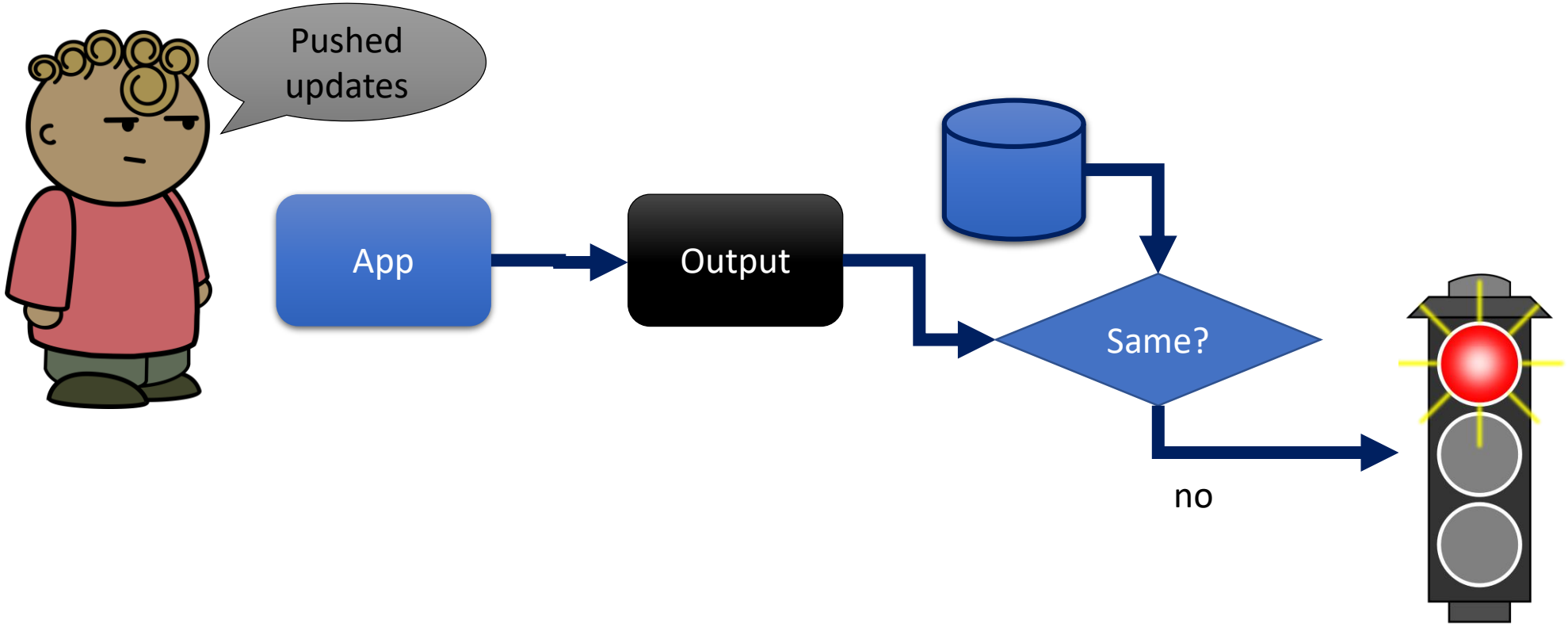
Approval tests

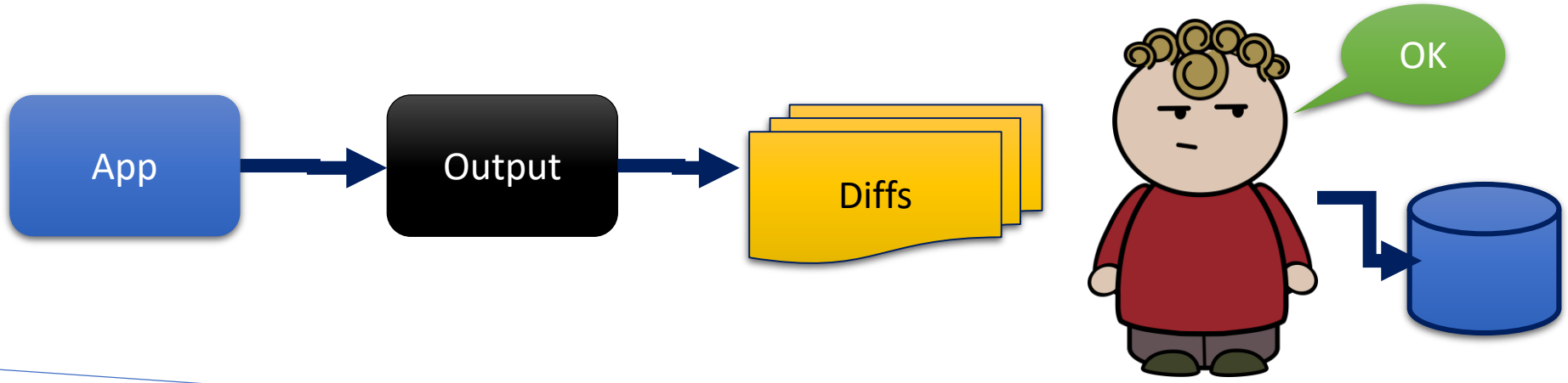
Llewellyn Falco, http://approvaltests.com

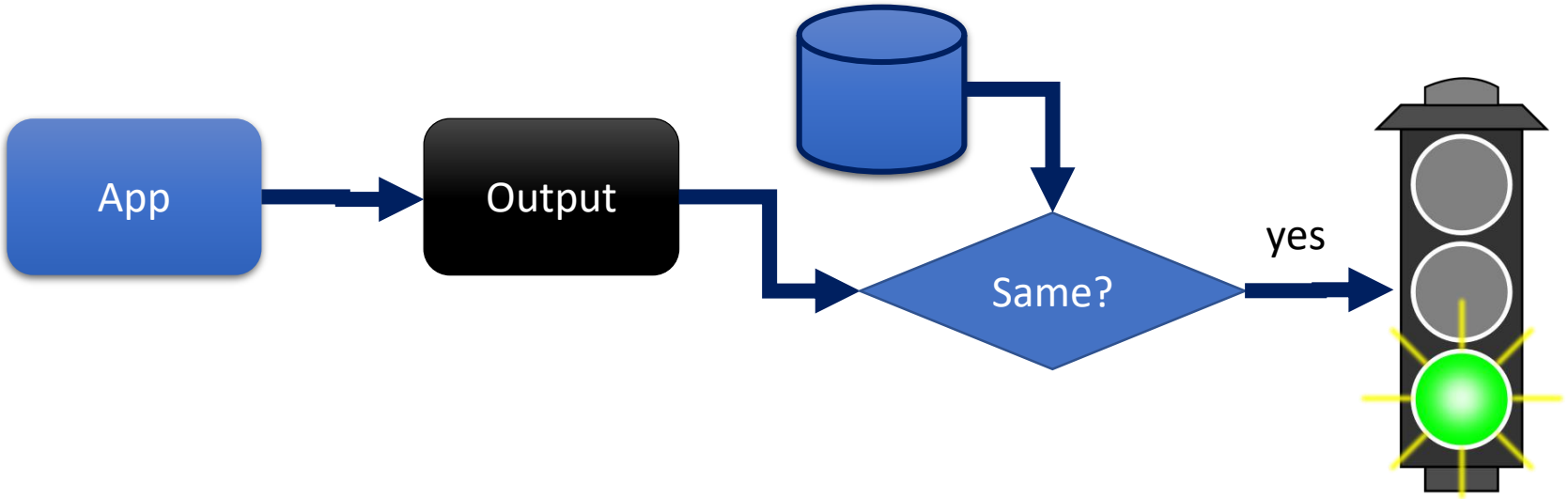
- Write a scenario with output
- Run and have a human (you) approve output
- Use as regression test


App
Output
OK


App
Output
Same?
yes

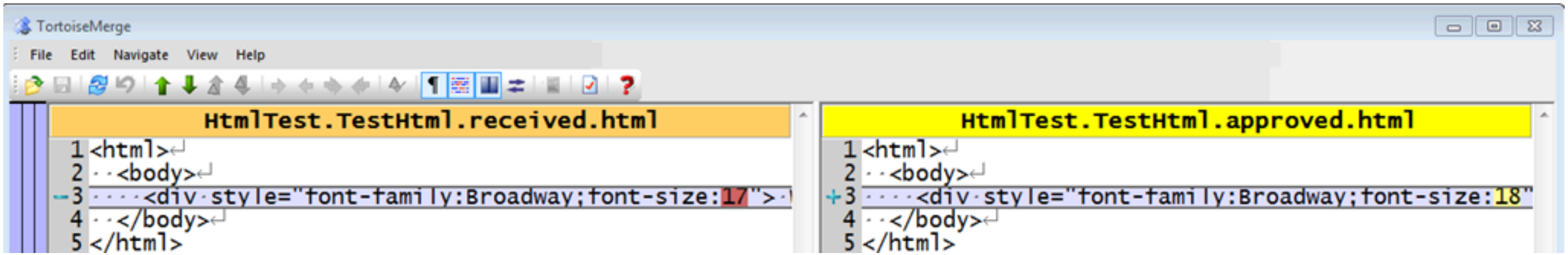

Pushed updates
App
Output
Same?
no


App
Output
Diffs
OK


App
Output
Same?
yes

ApprovalTest tests

```
[UseReporter(typeof(DiffReporter))]
[TestFixture]
public class SampleTest
{
    [Test] public void TestList()
    {
        var names = new[] {"Llewellyn", "James", "Dan", "Jason", "Katrina"};
        Array.Sort(names);
        Approvals.VerifyAll(names, "");
    }
}
```



http://blog.approvaltests.com/2011/12/using-reporters-in-approval-tests.html

Microsoft Strategy

Debugging

Thank You

Kathleen Dollard
.NET Team
@kathleendollard
kdollard@microsoft.com

- https://twitter.com/RichRogersIoT/status/998288346286768128
- Size you can handle
- https://twitter.com/IJohnson_TNF/status/998566264288305154
  - **Source control**

You are the Keeper of Your Code

Make it incrementally better

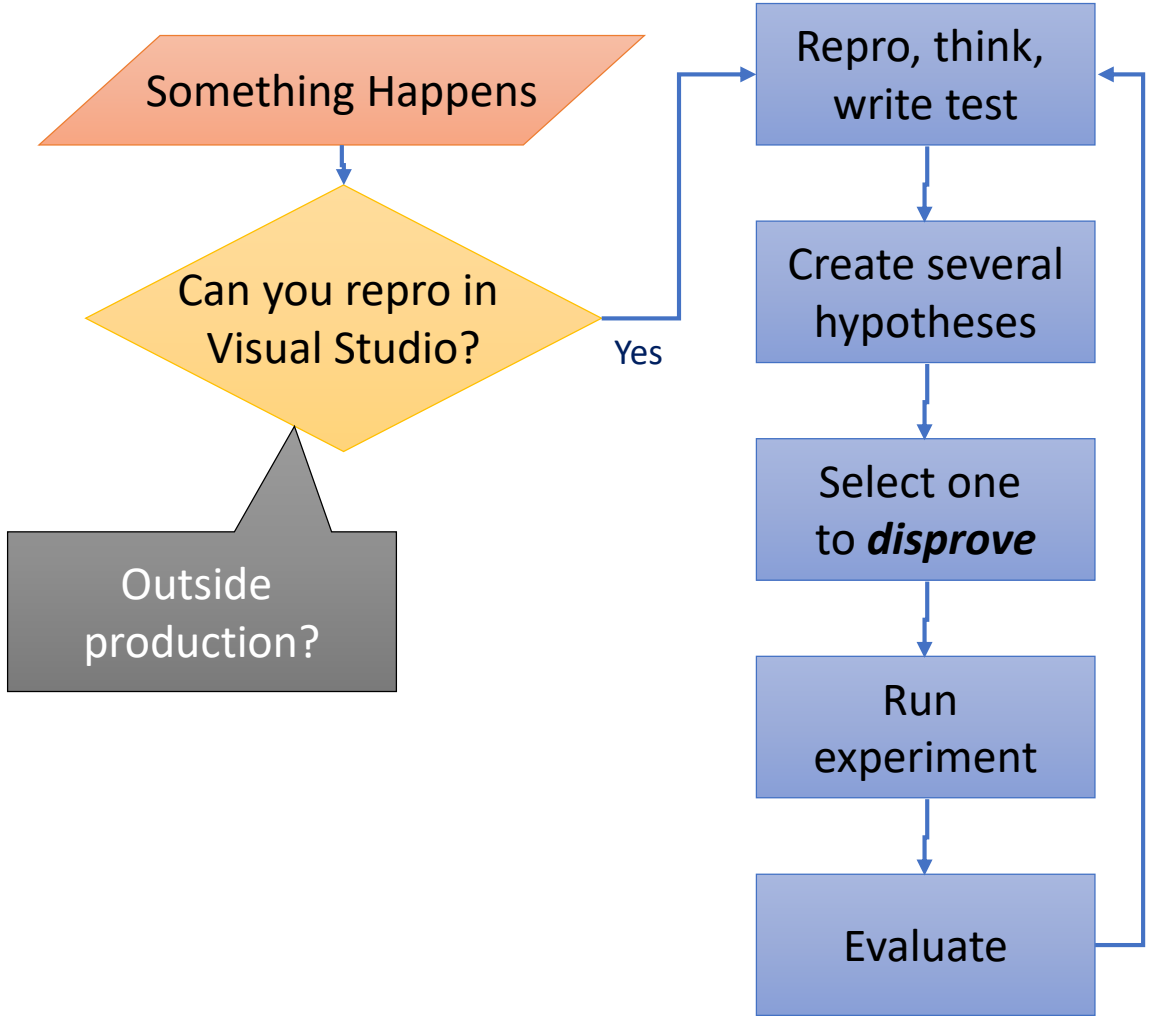
Picard Tips @PicardTips · Feb 27

Picard strategy tip: Pick your battles. Fight injustices before annoyances.

4 392 1.1K

Debugging

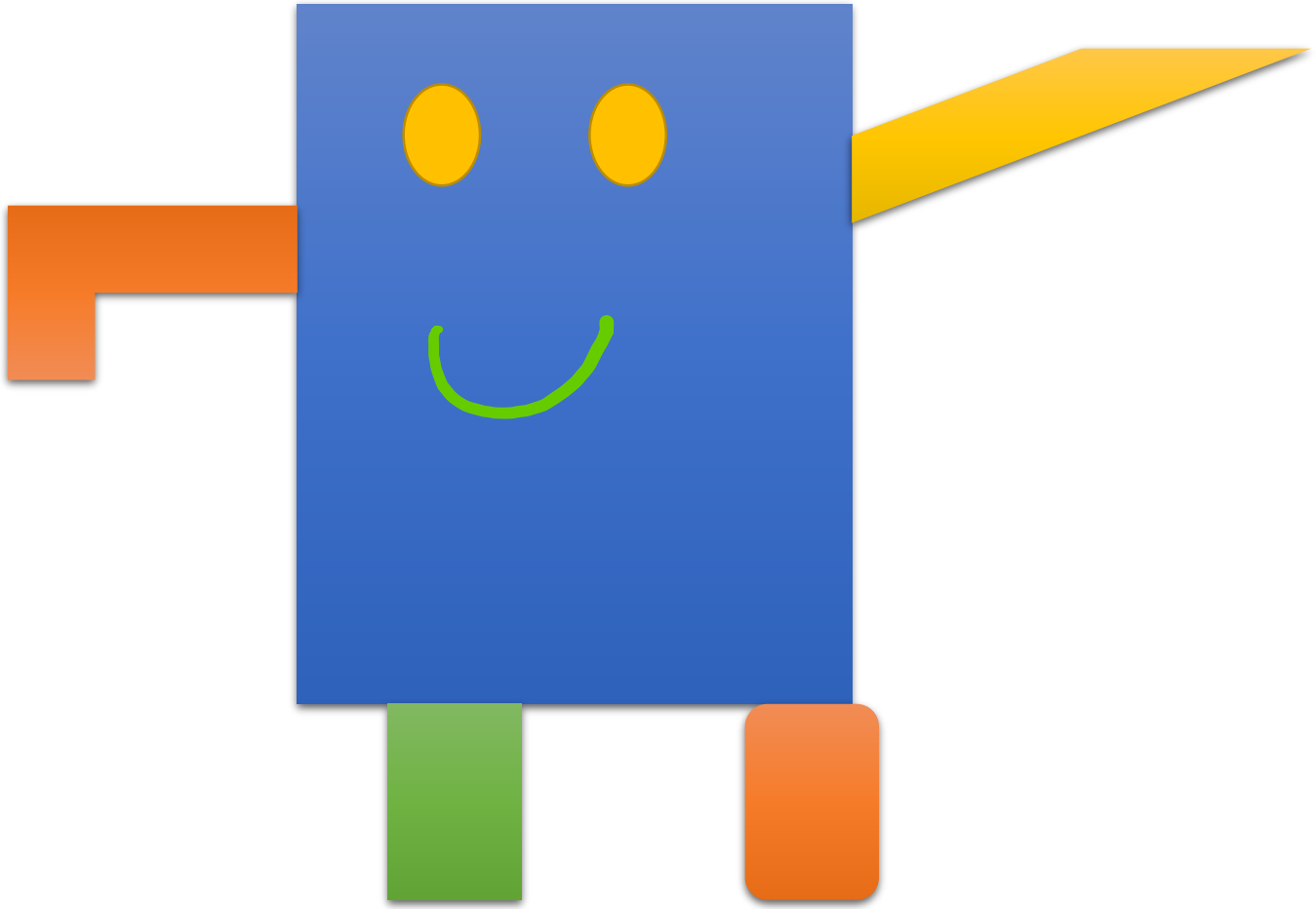
A Debugging Strategy - *Scientific Method*



Links on legacy code

- https://www.lighthousesoftware.com/best-practices
  - This website has good stuff on legacy – but its older legacy. Manifesto good
- https://builttoadapt.io/deal-with-legacy-before-it-deals-with-you-cc907c800845
  - Deeper article
- https://8thlight.com/blog/ginny-hendry/2014/07/11/take-pride-in-your-legacy-code.html
  - Ginny Hendry article that has parallels antiques

Build applications to bolt on new parts



Picard Tips @PicardTips · Feb 9

Picard schooling tip: Don't be too tentative. Have experiences. Be dastardly. Get into trouble.

12 210 758

Not Rewriting. Tweaking tech.

ASP.NET MVC Music Store Tutorial

Version 3.0b

Jon Galloway - Microsoft

4/28/2011

Moving MusicStore Forward

- App last updated in 2011
- Find *smallest* step you think reasonable

- ***Open and run in Visual Studio 2017 Update 8 Preview***

Moving MusicStore Forward

- Requested migration, assuring it would leave a log file
- Dire warnings about WebPages no longer being cool

- ***Sure***
- ***What could possibly go wrong?***

Moving MusicStore Forward

- Tried to build, received errors on
  - System.Web.Helpers
  - System.Web.MVC
  - System.Web.WebPages
- Warning icons on several things in the Dependency node of Solution Explorer

- ***Searched MVC 3 migration***
- ***Found docs page and went through steps***
  - ***(this was to MVC 4)***
- ***To update NuGet, I chose MVC 5***
- ***What could possibly go wrong?***

Moving MusicStore Forward

- Could not install Razor 3.2.6 because the app targeted .NET 4.0

- ***Targeted .NET 4.6.1***
  - ***Might as well jump forward on that***
- ***Reinstalled MVC 5 via NuGet***
- ***Closed and reopened VS because of a warning message***

Moving MusicStore Forward

- Four compiler errors about an ambiguity in Compare and ErrorMessage

- ***Removed using statement for System.Web.MVC from the model class (what was it doing there anyway?)***
- ***YEAH!!! Compile works***

Moving MusicStore Forward

- Error on run
- WebPages 2.0 was specified and 3.0 found
- OK, I knew that MVC 5 decision would cause troubles

- ***Changed WebPages 2.0 to 3.0***

Moving MusicStore Forward

- Code ran to point of data access
- "…Provider not installed…"

- ***Panic sets in***
- ***Read Readme.txt***
  - ***(Finally)***
- ***Glance through PDF***
  - ***Great, it's a CE database***
- ***Tried to create a new connection string***
- ***Tried to create a new connection string***
- ***Tried to create a new connection string***
- ***Tried to create a new connection string***
- ***Tried to create a new connection string***

Moving MusicStore Forward

- There is a broken connection in Server Explorer
- Could not create a new SQL Client database
  - Is this a Community problem, can't be. Dunno.
- ***Finally modified the broken connection to work***
- ***Copied the connection string***

Moving MusicStore Forward

- “Database does not contain model metadata…Ensure that IncludeMdatataCon…”

- ***Searched for message***
- ***Apparently this ancient EF didn’t do modern migrations***
- ***One suggestion is to delete and recreate database***
- ***Panic sets in***
- ***Decide to commit***

Moving MusicStore Forward

- db_lock fail on a deep name
  - .git/…/…/…/…/…/…

- ***Search for error***
- ***Add .vs to .gitignore***
  - ***Just `.vs\*` didn't work***
  - ***Panic***
  - ***`.vs` and `.vs\*` worked***

------ back to database

- ***Instead of deleting, just go to package manager and type***

```
> update-database
```

Moving MusicStore Forward

- “…cmdlet not found”

- ***Check NuGet and find 5 packages that need updates***
- ***Update***

------ back to database

- ***Try again***

```
> update-database
```

Moving MusicStore Forward

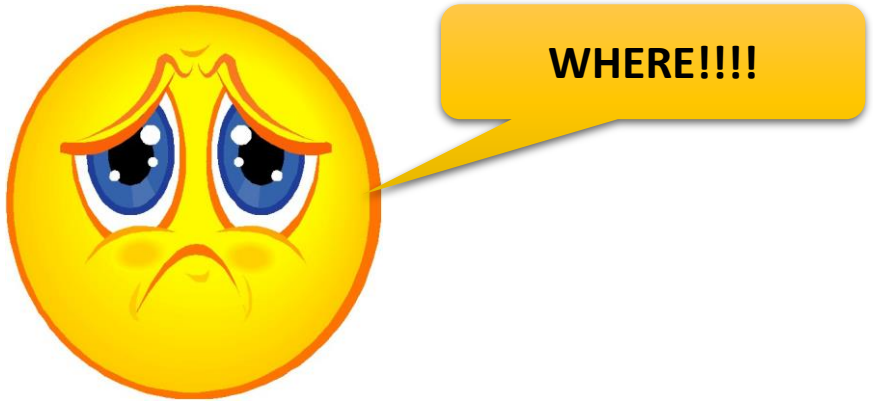
- “No migration configuration type found in the assembly…In Visual Studio you can use Enable-Migrations from Package Manager console…”

- ***OK, yeah, I can do that***

```
> enable-migrations
> update-database
```

Moving MusicStore Forward

- "…Unable to update…pending changes….or enable auto migrations. Set DbMigrationConfiguration.AutomaticMigrationsEnabled to true.

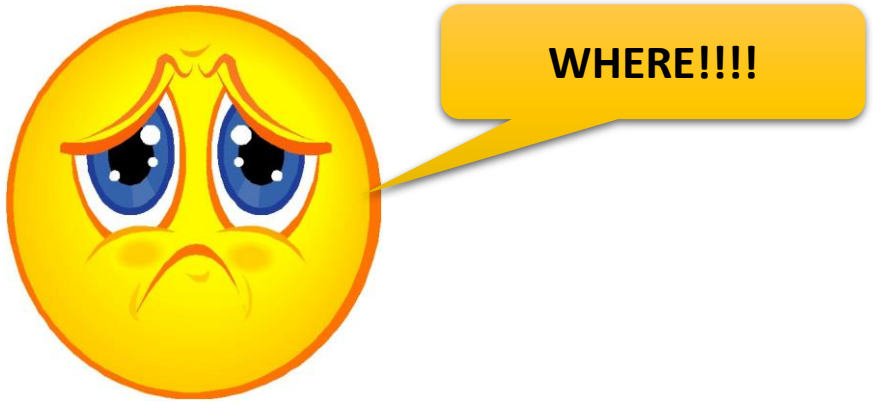


- ***Something about migrations?***

```
> add-migration
    Name:
```

Moving MusicStore Forward

- "…Unable to update…pending changes….or enable auto migrations. Set DbMigrationConfiguration.AutomaticMigrationsEnabled to true.



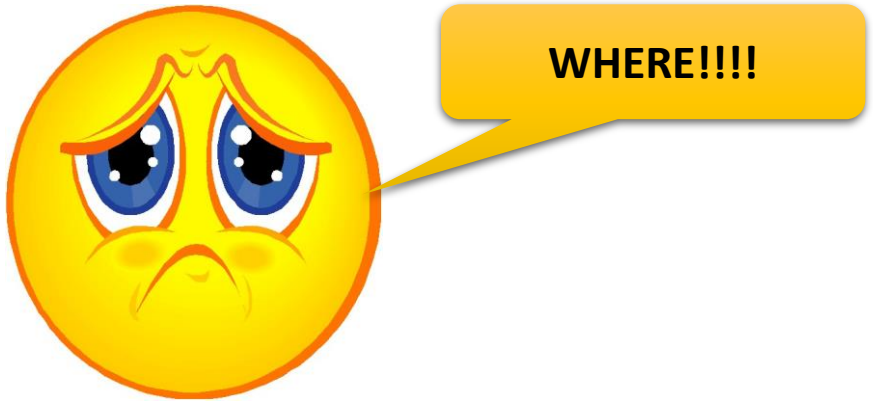
- ***Something about migrations?***

```
> add-migration

    Name: (this could have been a very bad word)
```

Moving MusicStore Forward

- "…Unable to update…pending changes….or enable auto migrations. Set DbMigrationConfiguration.AutomaticMigrationsEnabled to true.



- ***Something about migrations?***

```
> add-migration
    Name: MvcMusicStoreini
```

- ***Try to run again…***

Can't reach this page | Can't reach this page | ASP.NET MVC Music St

localhost:26641/

Home | Store | Cart (0) | Admin

ASP.NET MVC MUSIC STORE

***Fresh** off the grill*

built with ASP.NET MVC 3

Dave McComb

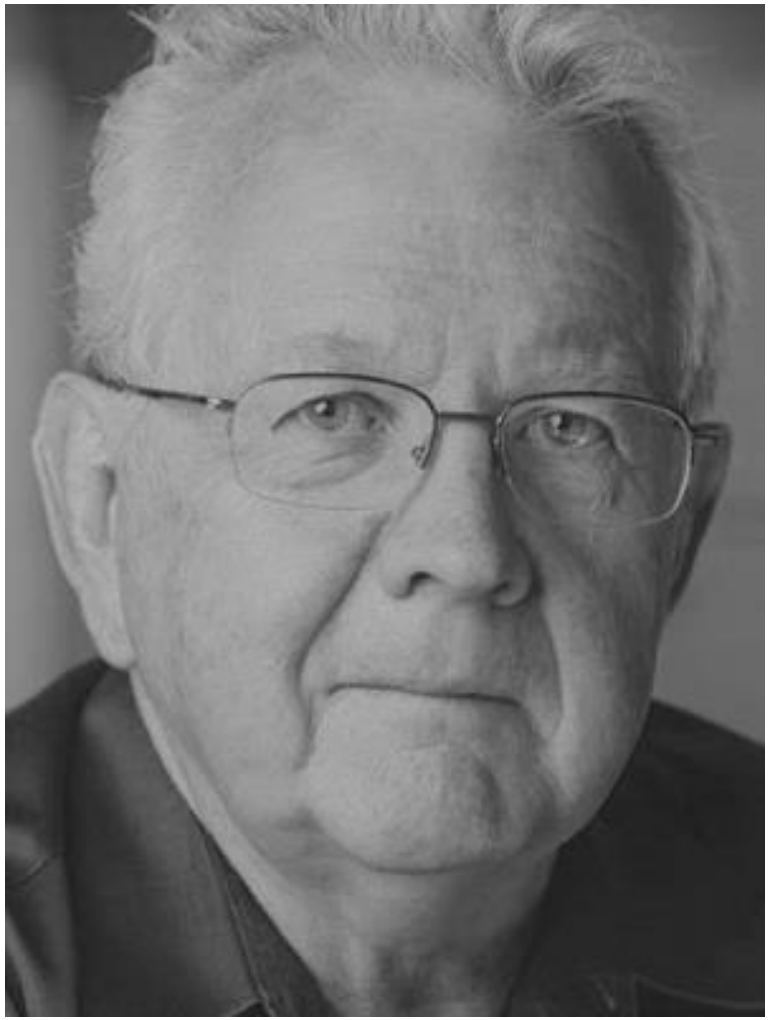
Dave McComb

Software Wasteland, February 2018

From his bio: “Prior to that, he was a part of the problem.”

From Kathleen: I haven’t had a chance to read this book yet, but I know and deeply respect Dave. Takeaways from the buzz:

- We spend way too much on software
  - You and I are invested in that
- Each company has a few hundred key concepts and relationship
- Each app may have hundreds of thousands fo distinctions
- I am deeply uncomfortable with the conclusion of new way



https://www.strategy-business.com/article/Are-You-Spending-Way-Too-Much-on-Software?gko=921cd

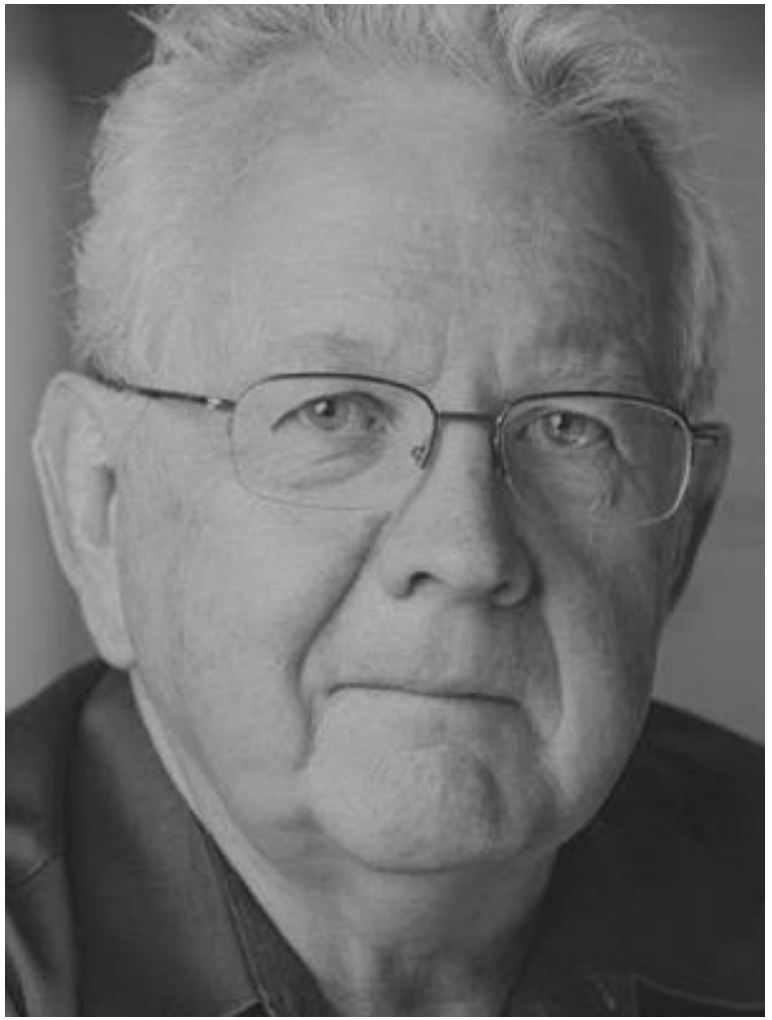
Dave McComb

Software Wasteland, February 2018

From his bio: “Prior to that, he was a part of the problem.”

From Kathleen: I haven’t had a chance to read this book yet, but I know and deeply respect Dave. Takeaways from the buzz:

- We spend way too much on software
  - You and I are invested in that
- Each company has a few hundred key concepts and relationship
- Each app may have hundreds of thousands fo distinctions
- I am deeply uncomfortable with the conclusion of new way



https://www.strategy-business.com/article/Are-You-Spending-Way-Too-Much-on-Software?gko=921cd