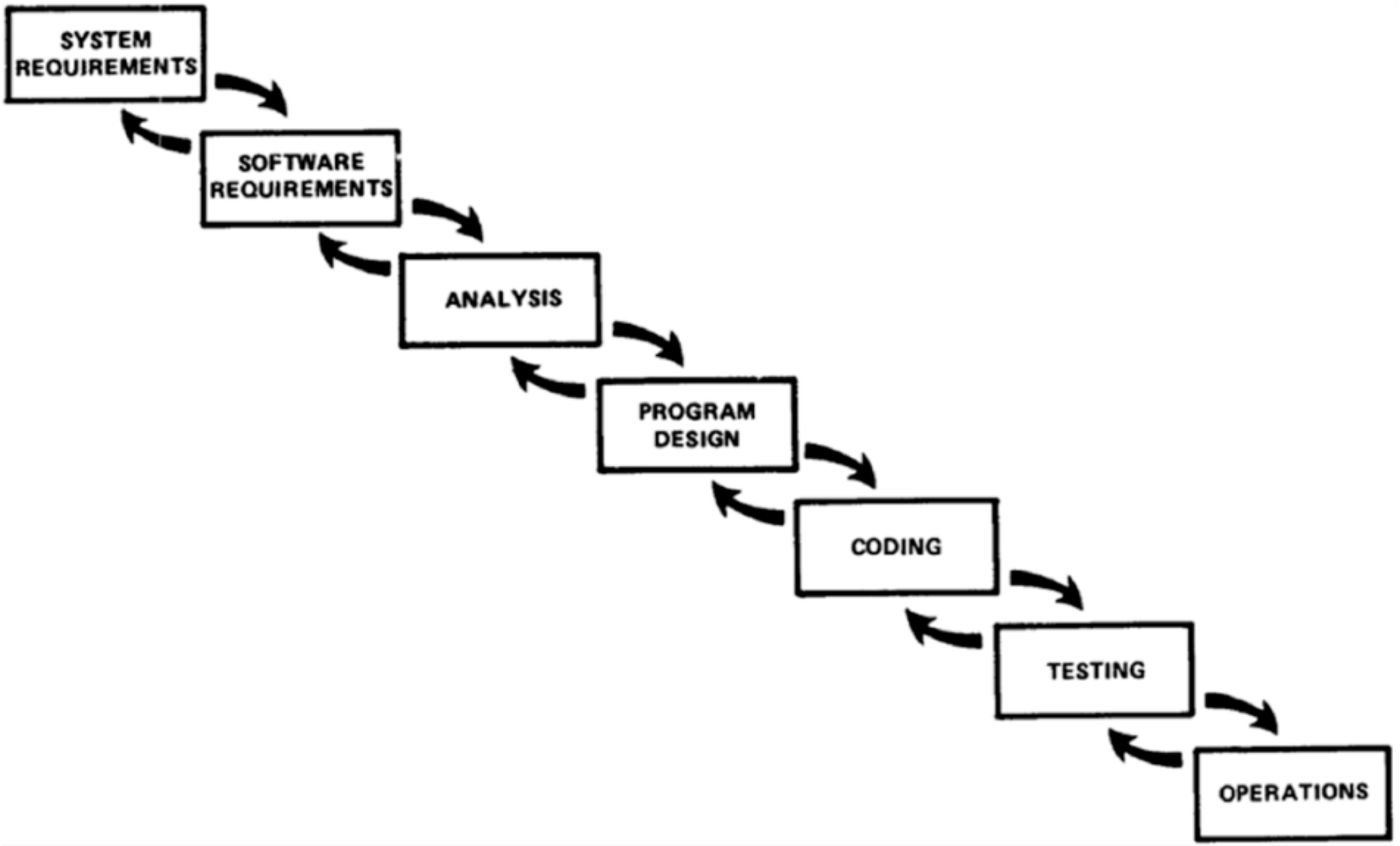
LESS is MORE

Seb Rose
Claysnow Limited
@sebrose

Last century?



http://leadinganswers.typepad.com/leading_answers/files/original_waterfall_paper_winston_royce.pdf

Agile manifesto signatories

Kent Beck	James Grenning	Robert C. Martin
Mike Beedle	Jim Highsmith	Steve Mellor
Arie van Bennekum	Andrew Hunt	Ken Schwaber
Alistair Cockburn	Ron Jeffries	Jeff Sutherland
Ward Cunningham	Jon Kern	Dave Thomas
Martin Fowler	Brian Marick	

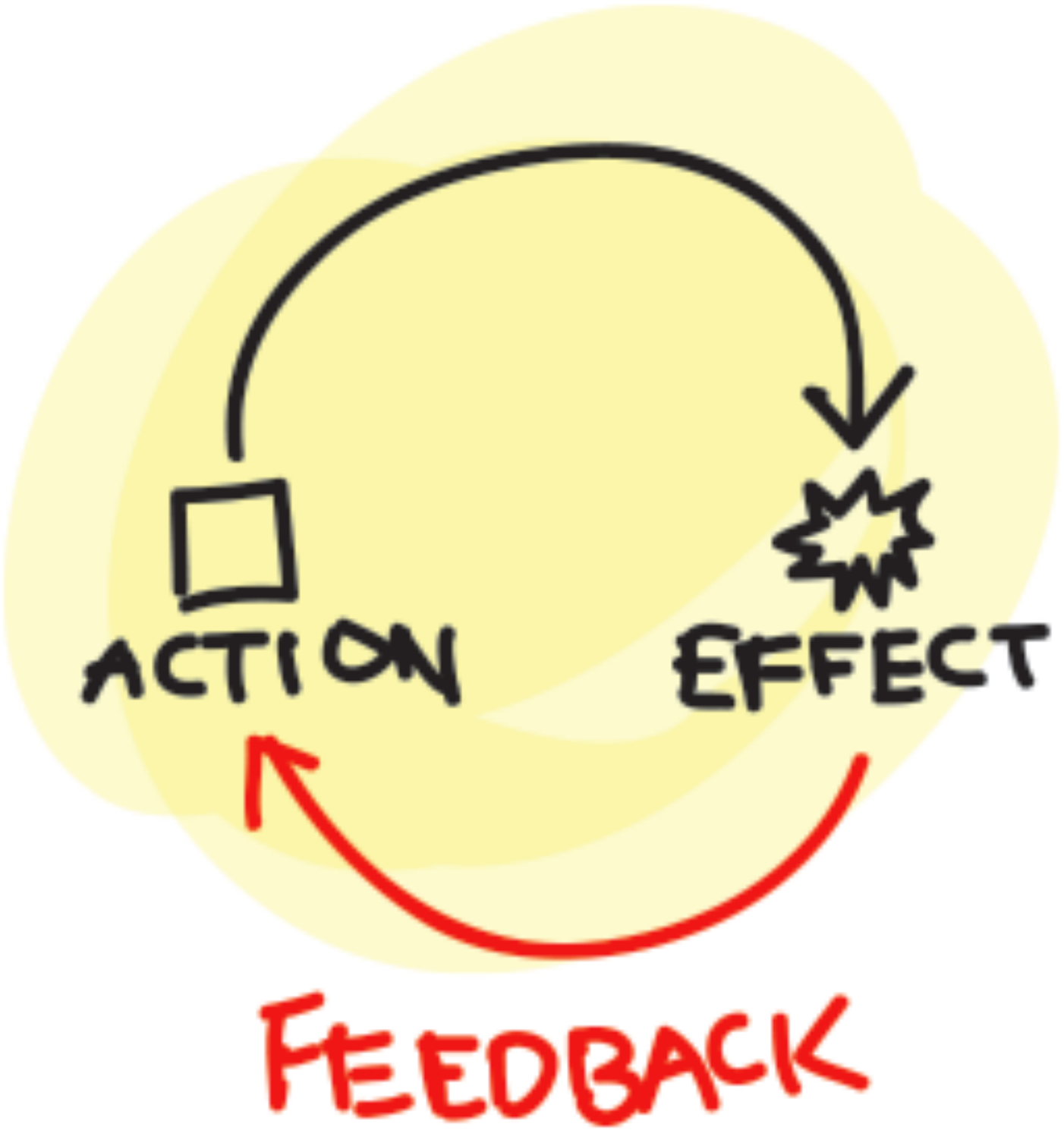
Overview

http://www.developersurvivalguide.com/tag/agile/

@sebrose

http://claysnow.co.uk

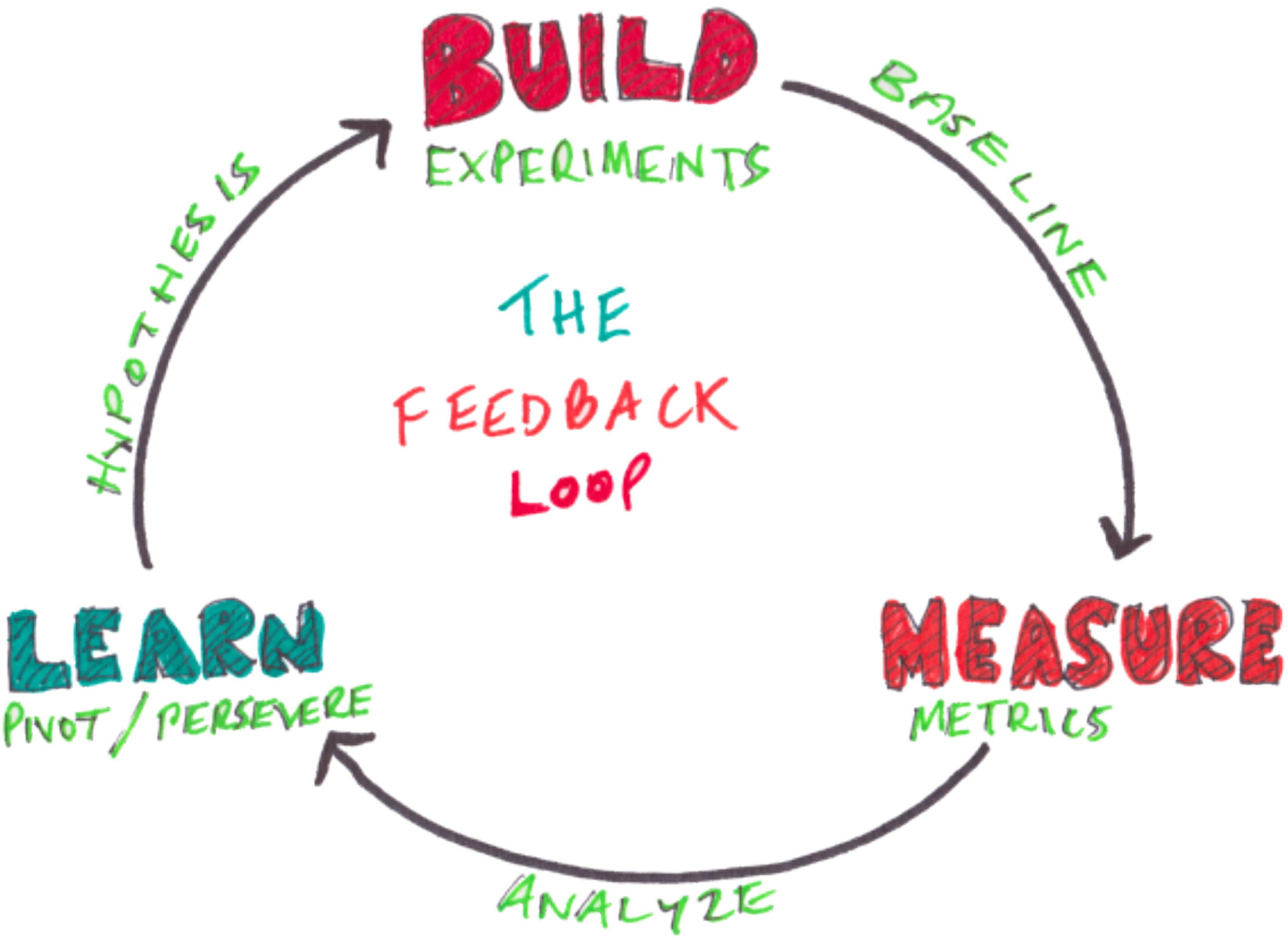
https://markkolier.files.wordpress.com/2012/12/feedback.png?w=300



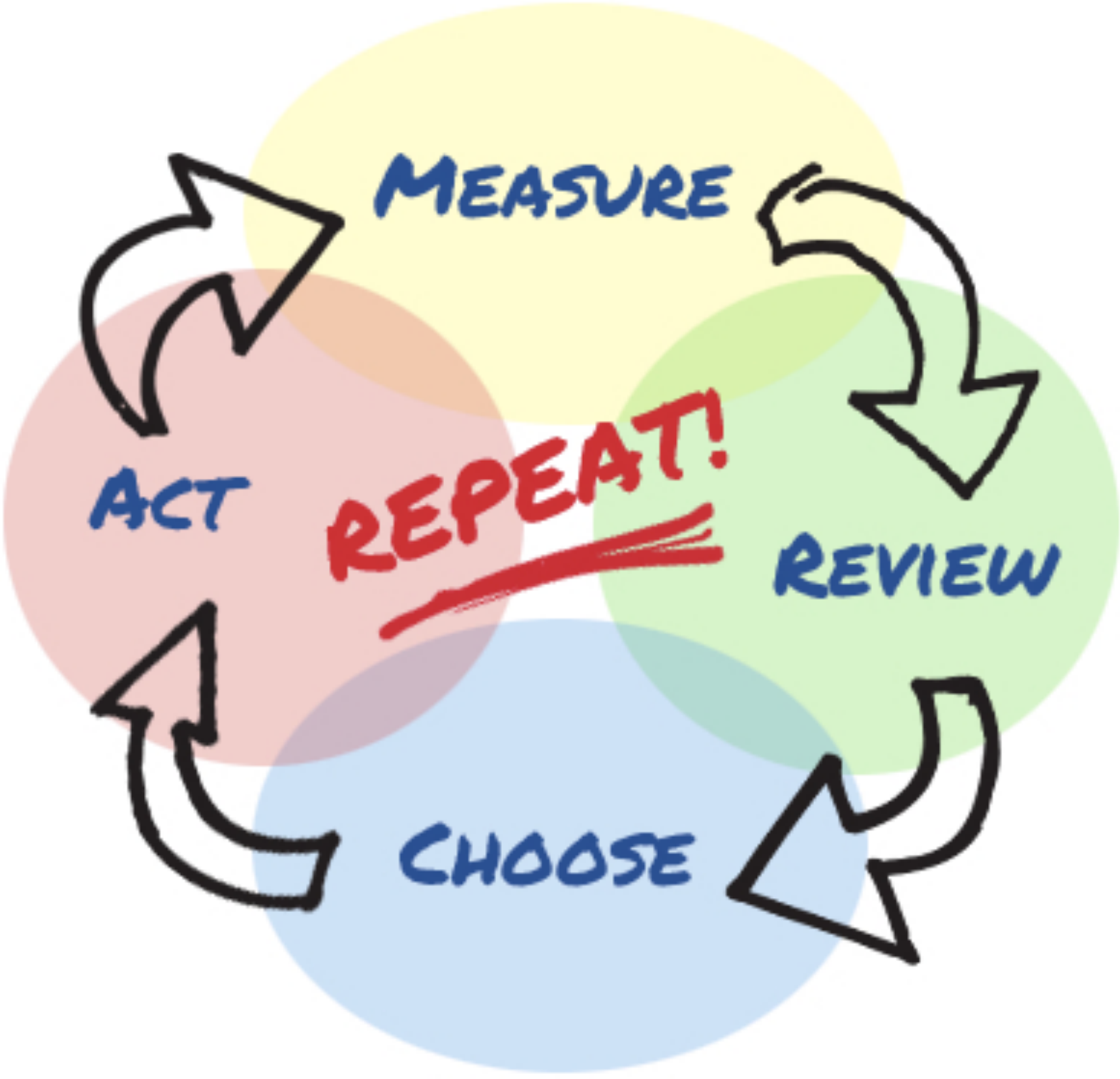
http://media.mediatemple.netdna-cdn.com/wp-content/uploads/2013/01/floops_loops.png

@sebrose

http://claysnow.co.uk



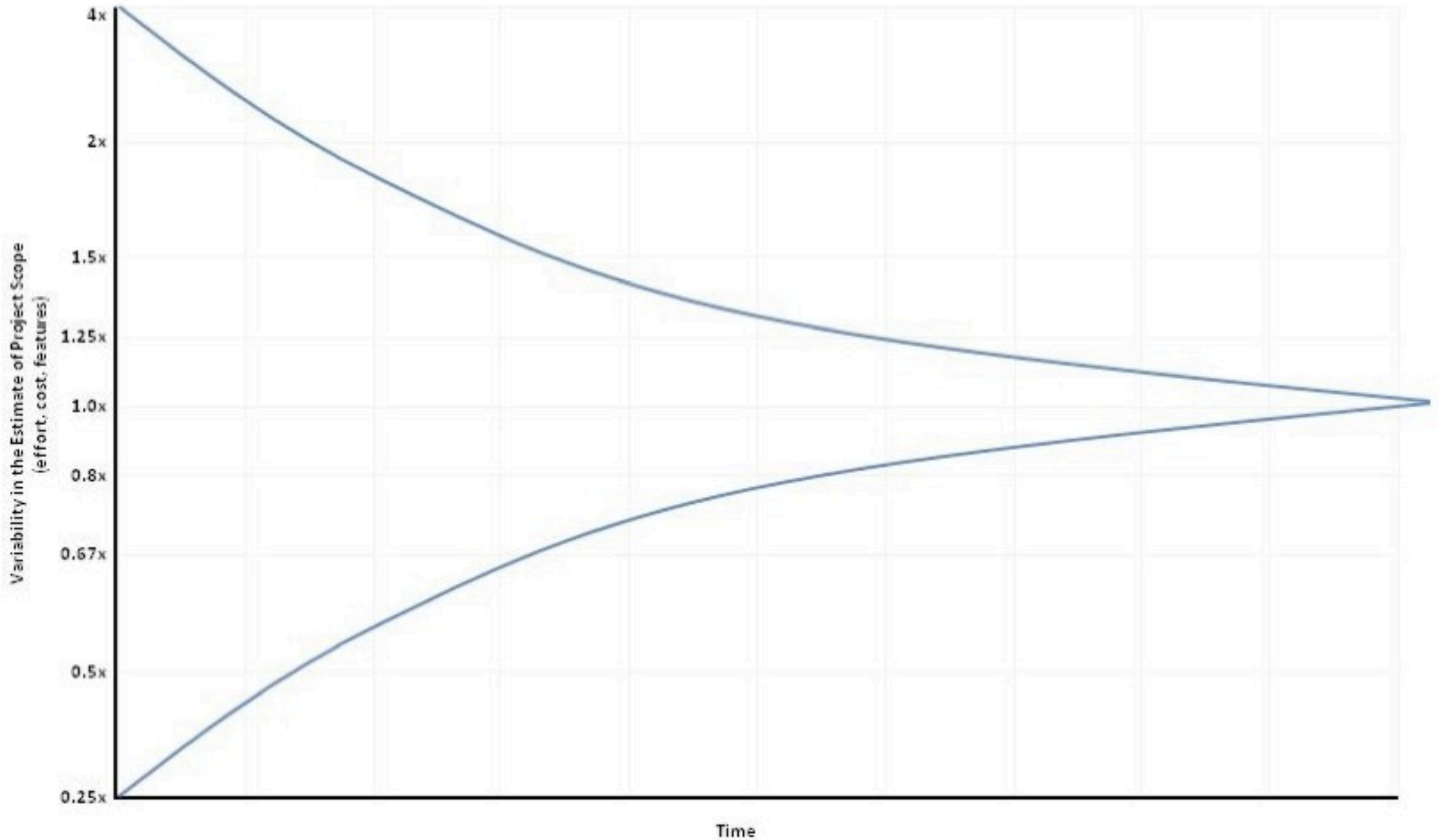
http://conversionxl.com/wp-content/uploads/2014/05/3.png



http://blog.rescuetime.com/wp-content/uploads/2012/08/feedback-loop-graphic-1.jpg

Is it small,
or just far away?

The Cone of Uncertainty in Project Management



@sebrose

http://claysnow.co.uk

"Ignorance is the single greatest impediment to throughput."

http://dannorth.net/2010/08/30/introducing-deliberate-discovery/

ASSUME YOU'RE IGNORANT

@sebrose

http://claysnow.co.uk

CHANGE OF PLAN

Outside-in: user stories, then TDD

@sebrose

http://claysnow.co.uk

User Stories

Overview

http://www.developersurvivalguide.com/tag/agile/

@sebrose

http://claysnow.co.uk

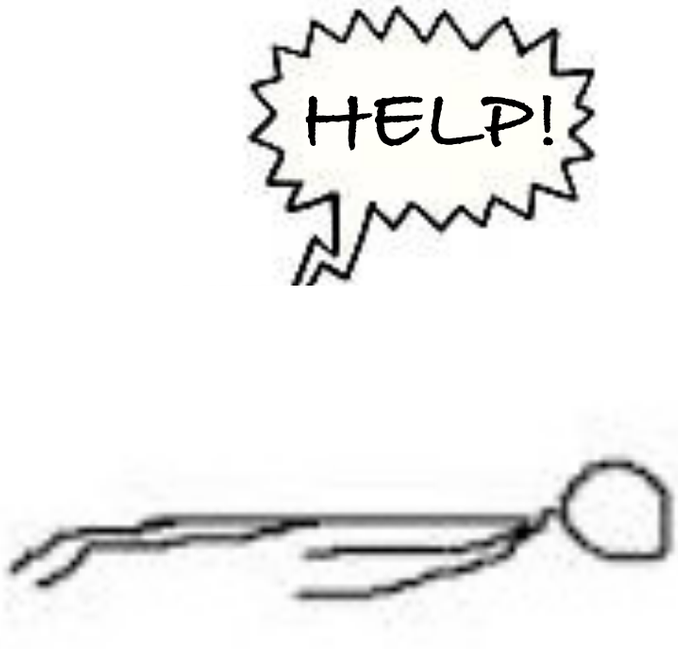
User Story Coupon

JBRAINS.CA

This coupon entitles the team to talk to

about

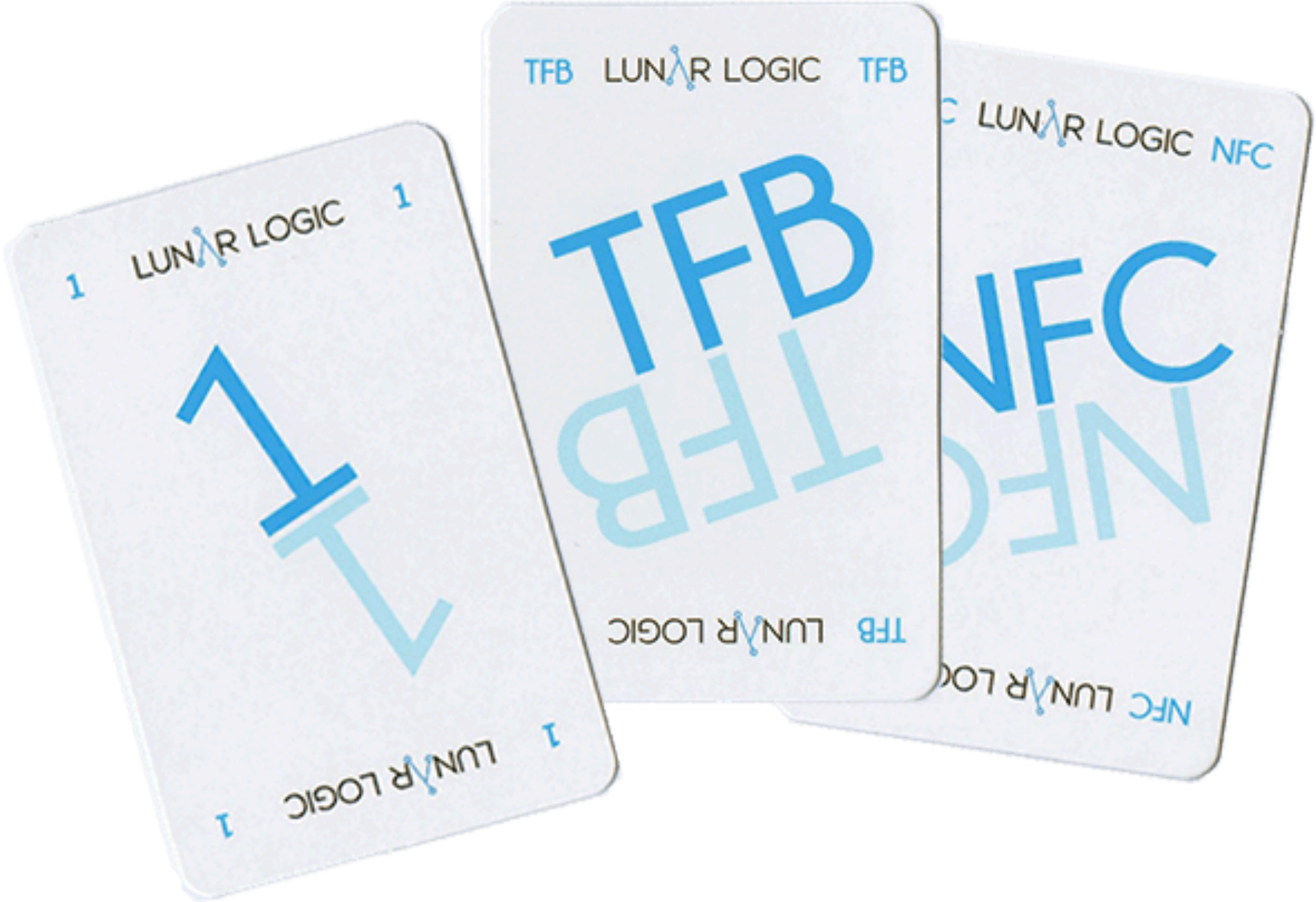
Already have some examples in mind? Turn this over.

To Do	In Progress	Done
Story A As a ..... I want ..... So that ..... 13 points		
Story B As a ..... I want ..... So that ..... 8 points		

https://intranet.5amsolutions.com/display/process/Playing+Planning+Poker

@sebrose

http://claysnow.co.uk



https://estimation.lunarlogic.io/assets/cards-range-8fc41b2e3fd282125f4602a712020204.png

@sebrose

http://claysnow.co.uk

Why split them?

Organize Email

Manage Email

Manage Calendar

Manage Contacts

Search Email | File Emails | Compose Email | Read Email | Delete Email | View Calendar | Create Appt | Update Appt | View Appt | Create Contact | Update Contact | Delete Contact

Searc by Keyword (WIP) | Move Emails | Create and send basic email (Done) | Open basic email (Done) | Delete email | View list of appts (Done) | Create basic appt (Done) | Update contents /location | View Appt (Done) | Create basic contact (Done) | Upda contact info (WIP)

Create sub folders (Done) | Send RTF e-mail | Open RTF e-mail | View Monthly formats | Create RTF appt | Accept/ Reject/T entative

Highest Priority Slices go here!

Release 1

Limit Search to one field | Send HTML e-mail | Open HTML e-mail | Empty Deleted Items | View Daily Format | Create HTML appt | Propose new time | Add address data | Update Address Info | Delete Contact

Limit Search to 1+ fields | Set email priority | Open Attachments | Mandatory/Optional

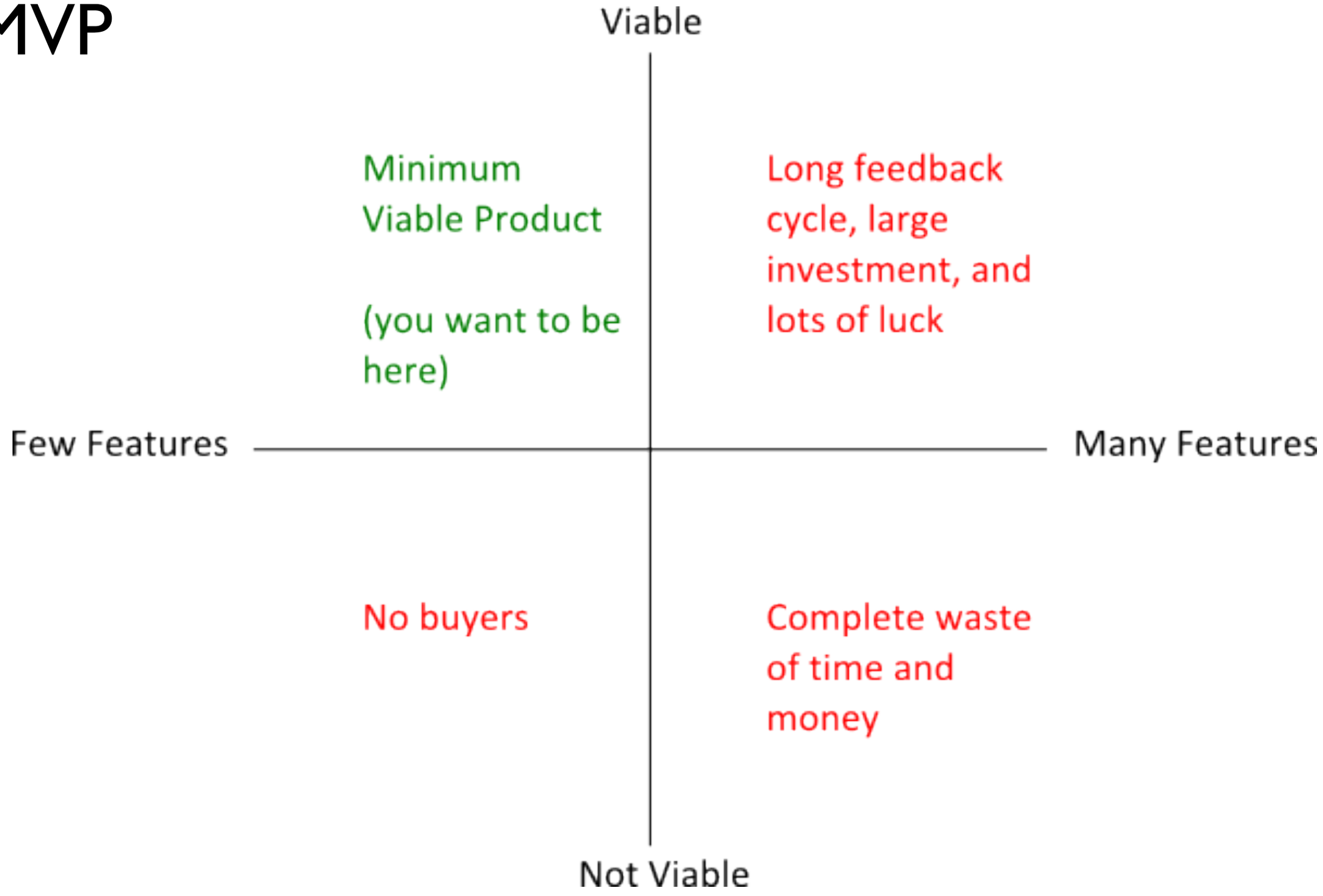
Release 2

Search attachments | Get address from contacts | View Weekly Formats | Get address from contacts | View Attachments | Import Contacts

Search sub folders | Send Attachments | Search Calendar | Add Attachments | Export Contacts

Release 3

MVP



http://toolsforagile.com/blog/wp-content/uploads/2011/03/mvp.png

I ndependent

N egotiable

V aluable

E stimable

S mall (Sized appropriately)

T estable

@sebrose

http://claysnow.co.uk

1. Just about everyone in the world has done this.
2. Lots of people have done this, including someone on our team.
3. Someone in our company has done this, or we have access to expertise.
4. Someone in the world did this, but not in our organisation (and probably at a competitor).
5. Nobody in the world has ever done this before.

http://lizkeogh.com/2013/07/21/estimating-complexity/

Naked Planning

MMFs

1.

2.

3.

4.

5.

6.

7.

In progress

Urgent

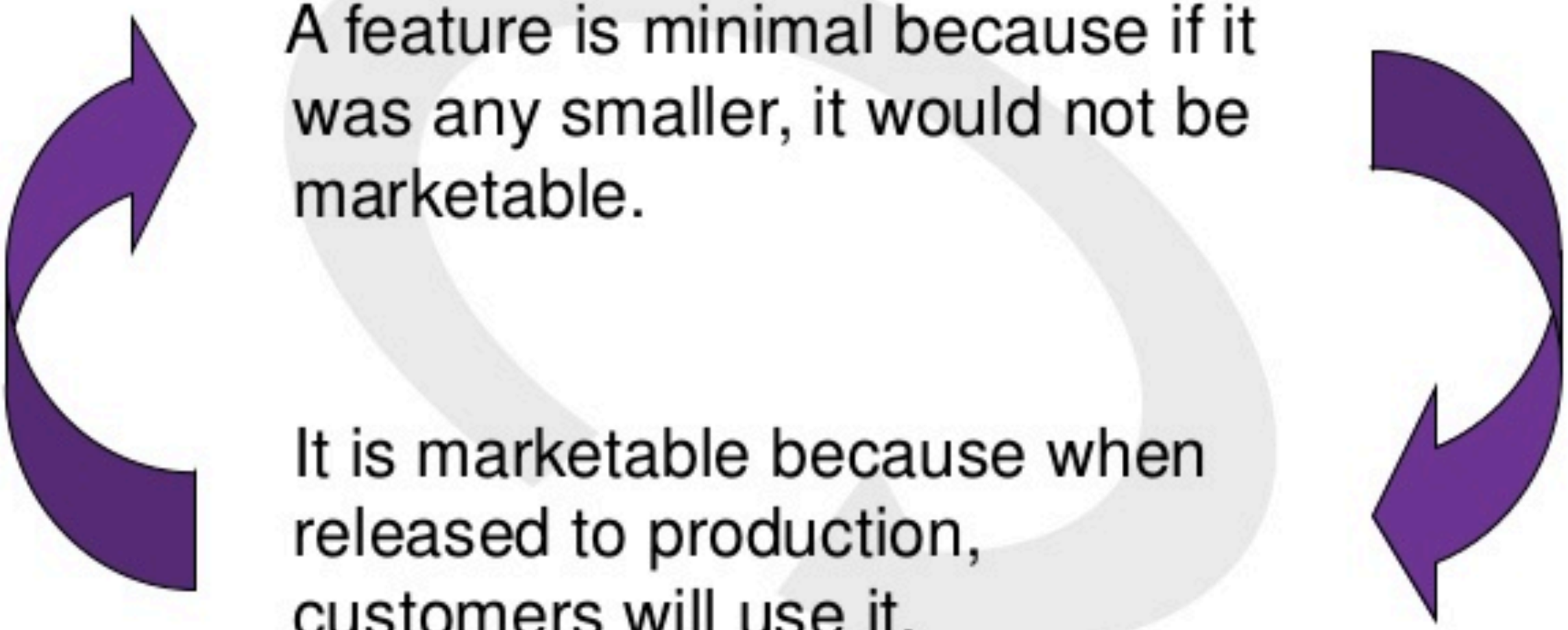
Average delivery time from here: ?? days

via @arlobelshee

@sebrose

http://claysnow.co.uk

Minimal Marketable Feature



http://image.slidesharecdn.com/agilepm-140409183015-phpapp01/95/agile-product-owner-51-638.jpg

©

How to split them?

HOW TO SPLIT A USER STORY

1 PREPARE THE INPUT STORY

Does the big story satisfy INVEST* (except, perhaps, small)?

NO → Combine it with another story or otherwise reformulate it to get a good, if large, starting story.

YES → Is the story size 1/10 to 1/4 of your velocity?

- You're done.
- Continue. You need to split it.

2 APPLY THE SPLITTING PATTERNS

start here

WORKFLOW STEPS

Does the story describe a workflow?

- Can you take a thin slice through the workflow first and enhance it with more stories later?
- Can you split the story so you do the beginning and end of the workflow first and enhance with stories from the middle of the workflow?

OPERATIONS

Does the story include multiple operations? (e.g. is it about "managing" or "configuring" something?)

- Can you split the operations into separate stories?

BUSINESS RULE VARIATIONS

Does the story have a variety of business rules? (e.g. is there a domain term in the story like "flexible dates" that suggests several variations?)

- Can you split the story so you do a subset of the rules first and enhance with additional rules later?

DEFER PERFORMANCE

Does the story get much of its complexity from satisfying non-functional requirements like performance?

- Could you split the story to just make it work first and then enhance it to satisfy the non-functional requirement?

VARIATIONS IN DATA

Does the story do the same thing to different kinds of data?

- Can you split the story to process one kind of data first and enhance with the other kinds later?

SIMPLE/COMPLEX

Does the story have a simple core that provides most of the value and/or learning?

- Could you split the story to do that simple core first and enhance it with later stories?

MAJOR EFFORT

When you apply the obvious split, is whichever story you do first the most difficult?

- Could you group the later stories and defer the decision about which story comes first?

INTERFACE VARIATIONS

Does the story have a complex interface?

- Is there a simple version you could do first?

Does the story get the same kind of data via multiple interfaces?

- Can you split the story to handle data from one interface first and enhance with the others later?

last resort

BREAK OUT A SPIKE

Are you still baffled about how to split the story?

- Can you find a small piece you understand well enough to start?
  - Write that story first, build it, and start again at the top of this process.
- Can you define the 1-3 questions most holding you back?
  - Write a spike with those questions, do the minimum to answer them, and start again at the top of this process
- Take a break and try again.

3 EVALUATE THE SPLIT

Are the new stories roughly equal in size?

- NO → Try another pattern on the original story or the larger post-split stories.
- YES → Is each story about 1/10 to 1/4 of your velocity?
  - Try another pattern on the original story or the larger post-split stories.
  - Do each of the stories satisfy INVEST?
    - Try another pattern.
    - Are there stories you can deprioritize or delete?
      - Try another pattern. You probably have waste in each of your stories.
      - Is there an obvious story to start with that gets you early value, learning, risk mitigation, etc.?
        - Try another pattern to see if you can get this.
        - You're done, though you could try another pattern to see if it works better.

* INVEST - Stories should be:
Independent
Negotiable
Valuable
Estimable
Small
Testable

AGILE FOR ALL
www.agileforall.com

Visit http://www.richardlawrence.info/splitting-user-stories/ for more info on the story splitting patterns

Copyright © 2011-2013 Agile For All. All rights reserved.

Last updated 3/26/2013

@sebrose

http://claysnow.co.uk

1. Write out any, and I mean any, meaningful end-to-end scenario in detail with concrete values at every step.

2. Now that you've chosen one real scenario, go to each step in that scenario and ask the question, "What would I need to assume to eliminate this step?" If you find those assumptions make for a reasonable scenario, then use that assumption to simplify the scenario.

3. Repeat step 2 until exhausted or unable to come up with a simplifying assumption with five minutes' thought.

http://www.jbrains.ca/permalink/how-youll-probably-learn-to-split-features

1. Write out any, and I mean any, meaningful end-to-end scenario in detail with concrete values at every step.

2. Now that you've chosen one real scenario, go to each step in that scenario and ask the question, "What would I need to assume to eliminate this step?" If you find those assumptions make for a reasonable scenario, then use that assumption to simplify the scenario.

3. Repeat step 2 until exhausted or unable to come up with a simplifying assumption with five minutes' thought.

http://www.jbrains.ca/permalink/how-youll-probably-learn-to-split-features

1. Write out any, and I mean any, meaningful end-to-end scenario in detail with concrete values at every step.

2. Now that you've chosen one real scenario, go to each step in that scenario and ask the question, "What would I need to assume to eliminate this step?" If you find those assumptions make for a reasonable scenario, then use that assumption to simplify the scenario.

3. Repeat step 2 until exhausted or unable to come up with a simplifying assumption with five minutes' thought.

http://www.jbrains.ca/permalink/how-youll-probably-learn-to-split-features

Shouty!

@sebrose

http://claysnow.co.uk

Ruby

Shouty

Shouty is a social networking application for short physical distances. When someone shouts, only people within 500m kan hear it.

Shouty doesn't exist yet - you will implement it yourself!

That is, if you're attending a BDD/Cucumber course.

https://github.com/cucumber-ltd/shouty.rb

C#

Shouty

Shouty is a social networking application for short physical distances. When someone shouts, only people within 500m kan hear it.

Shouty doesn't exist yet - you will implement it yourself!

That is, if you're attending a BDD/Cucumber course.

https://github.com/cucumber-ltd/shouty.net

Java

Shouty

Shouty is a social networking application for short physical distances. When someone shouts, only people within 1000m can hear it.

Shouty doesn't exist yet - you will implement it yourself!

That is, if you're attending a BDD/Cucumber course.

https://github.com/cucumber-ltd/shouty.java

Brainstorm capabilities

- Who are the main stakeholders?
- What can people do with the app?
- What are the main differentiators from other apps?

https://github.com/cucumber-ltd/shouty.java

Let's apply @jbrains approach:

1. Write out any, and I mean any, meaningful end-to-end scenario in detail with concrete values at every step.

2. Now that you've chosen one real scenario, go to each step in that scenario and ask the question, "What would I need to assume to eliminate this step?" If you find those assumptions make for a reasonable scenario, then use that assumption to simplify the scenario.

3. Repeat step 2 until exhausted or unable to come up with a simplifying assumption with five minutes' thought.

http://www.jbrains.ca/permalink/how-youll-probably-learn-to-split-features

What do you think of our story?

TDD

@sebrose

http://claysnow.co.uk

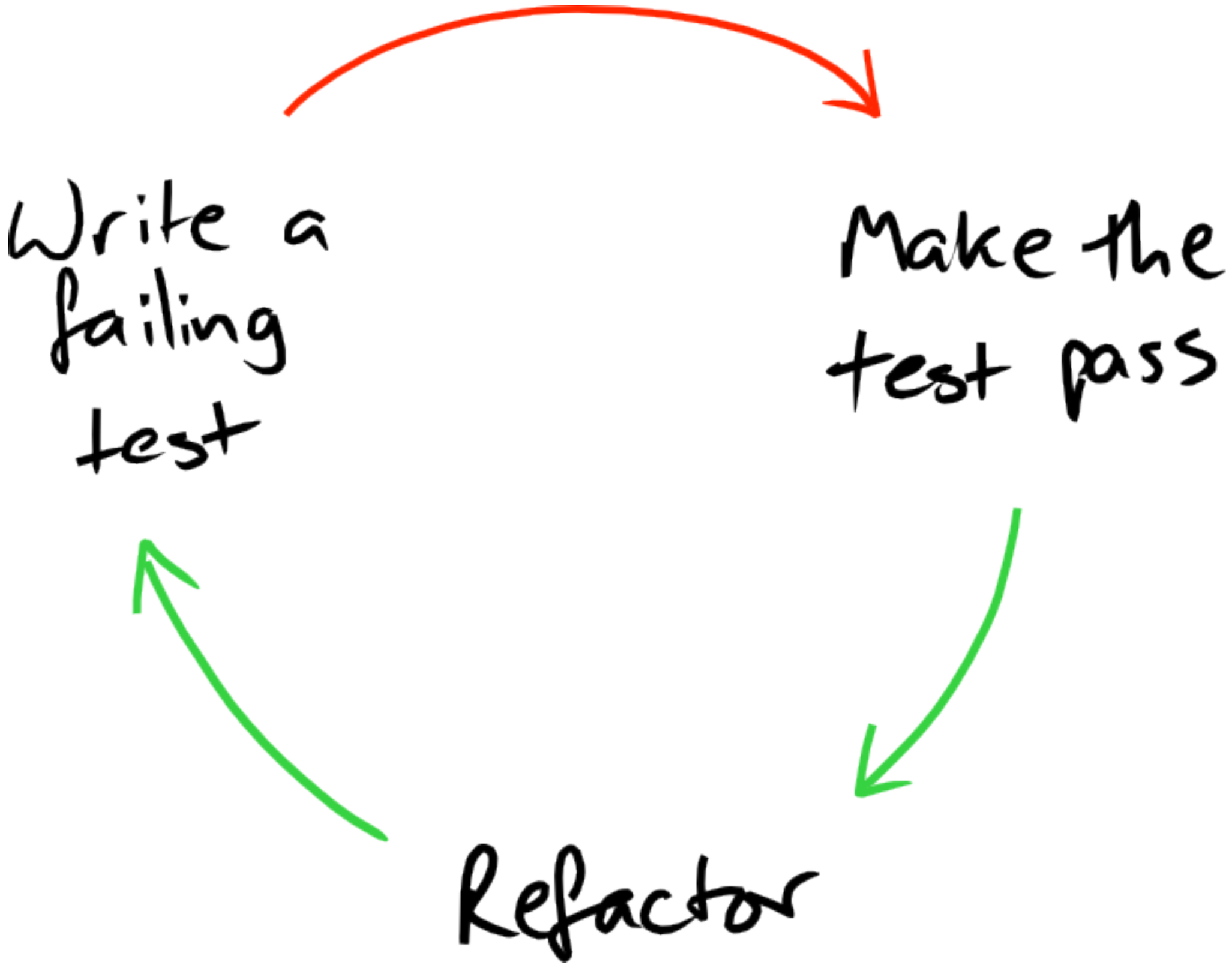
Print Diamond

Given a letter, print a diamond starting with ‘A’ with the supplied letter at the widest point.

For example: `print-diamond 'C'` prints:

```
  A
 B B
C   C
 B B
  A
```

http://claysnow.co.uk/recycling-tests-in-tdd/



http://www.amazon.co.uk/Growing-Object-Oriented-Software-Addison-Wesley-Signature-ebook/dp/B002TIOYVW

@sebrose

http://claysnow.co.uk

Write a failing test

.... or make an existing test fail

What do you think of these two approaches?

Transformation Priority Premise

- *Transformations* are simple operations that change the behaviour of code.
- Transformations can be used as the sole means for passing the currently failing test in the `red/green/refactor` cycle.
- *Transformations* have a priority, or a preferred ordering,
  - prevent impasses, or long outages in the `red/green/refactor` cycle.

http://blog.8thlight.com/uncle-bob/2013/05/27/TheTransformationPriorityPremise.html

- **({}–>nil)** no code at all->code that employs nil
- **(nil->constant)**
- **(constant->constant+)** a simple constant to a more complex constant
- **(constant->scalar)** replacing a constant with a variable or an argument
- **(statement->statements)** adding more unconditional statements.
- **(unconditional->if)** splitting the execution path
- **(scalar->array)**
- **(array->container)**
- **(statement->recursion)**
- **(if->while)**
- **(expression->function)** replacing an expression with a function or algorithm
- **(variable->assignment)** replacing the value of a variable.

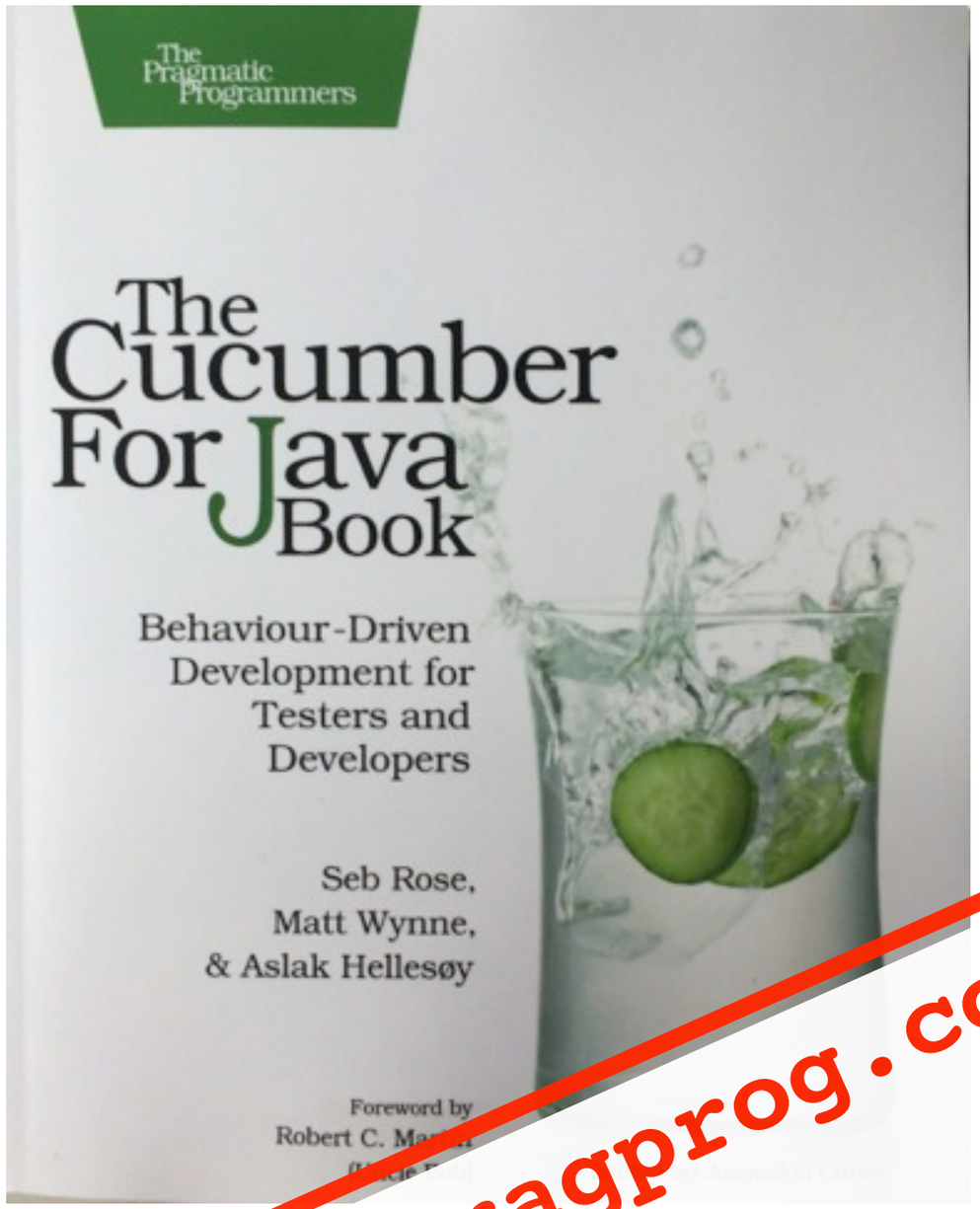
http://blog.8thlight.com/uncle-bob/2013/05/27/TheTransformationPriorityPremise.html

Deliver early to get
FAST FEEDBACK

Deliver increments to
MANAGE RISK

Deliver less to
LEARN MORE

CLAY SNOW



Seb Rose

Twitter: **@sebrose**

Blog: **www.claysnow.co.uk**

E-mail: **seb@claysnow.co.uk**

Discount code at pragprog.com
ACCU_Bristol_25
Valid for 1 month

"The size of a work item or it's complexity has ***no correlation*** *on [sic] the time it takes to deliver the work."*

http://scrumprouk.tumblr.com/post/74824285728/kanban-work-items-to-slice-or-not-to-slice