

How do you know that your product works?

Lean Kanban Central Europe keynote
Nov 2014

Consultant



Henrik Kniberg

henrik.kniberg@crisp.se
@HenrikKniberg

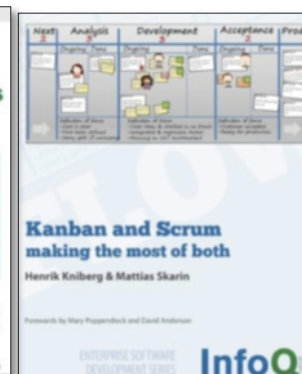
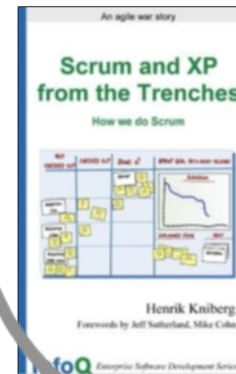
Parent

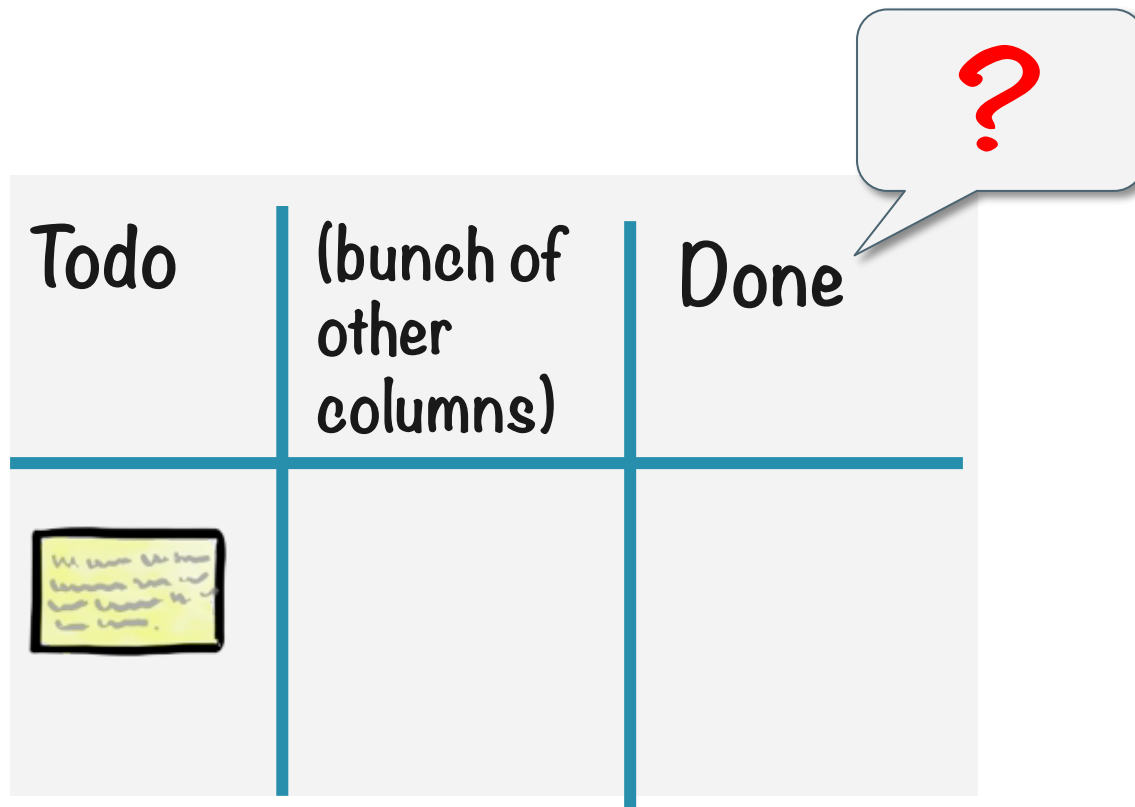


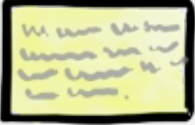
Agile & Lean coach



Author





Todo	(bunch of other columns)	Code committed
		



Developer

I'm done!

Where's my product?



User



Where's my product?



User

We're Done!



Developer



100% completed requirements
No reported defects



We're Done!



Wait... is anyone actually using it?

Code committed	Tested	In production	People actually use it
			

100% completed requirements
No reported defects

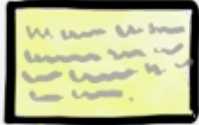


NOW we're
surely done!

Aren't we?

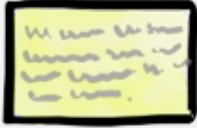
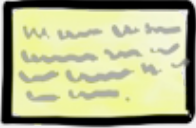
<http://www.youtube.com/watch?v=FWTtcOQjOXI>



Code committed	Tested	In production	People actually use it
			

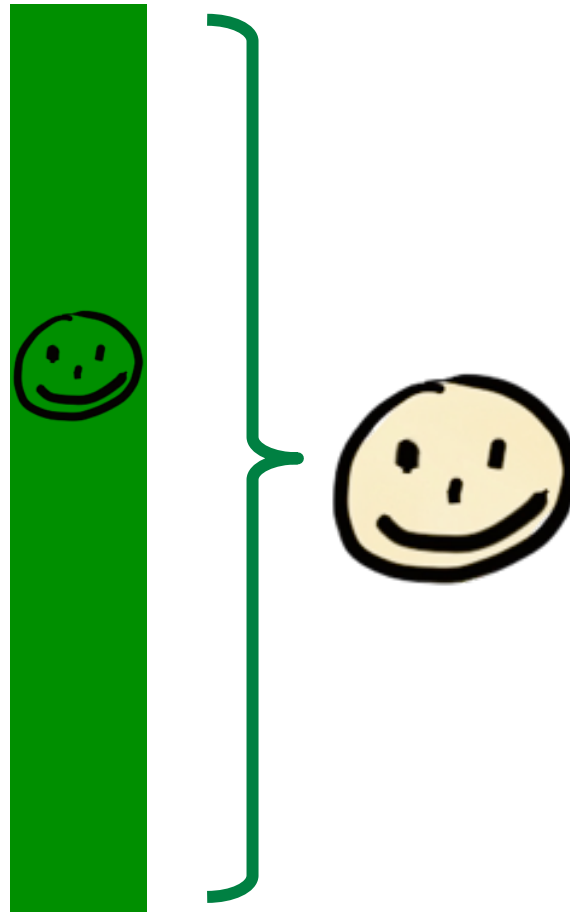
100% completed requirements
No reported defects



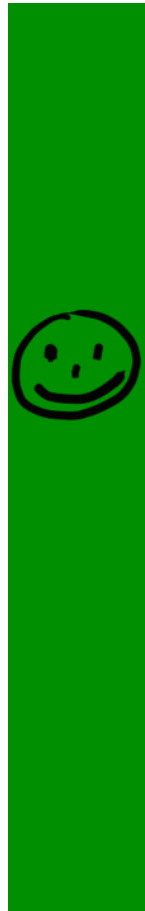
Code committed	Tested	In production	People actually use it	Solves the user's problem
				

100% completed requirements
No reported defects





Value of
your solution



Value of
your solution



Value of old
solution



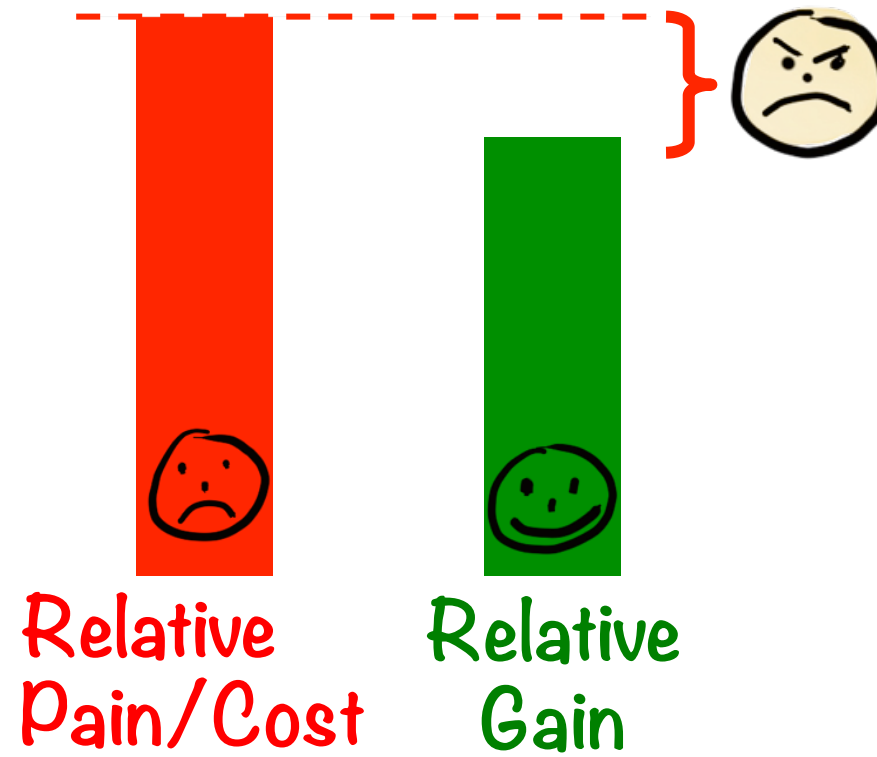


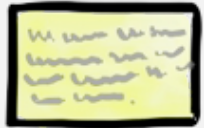
Relative gain

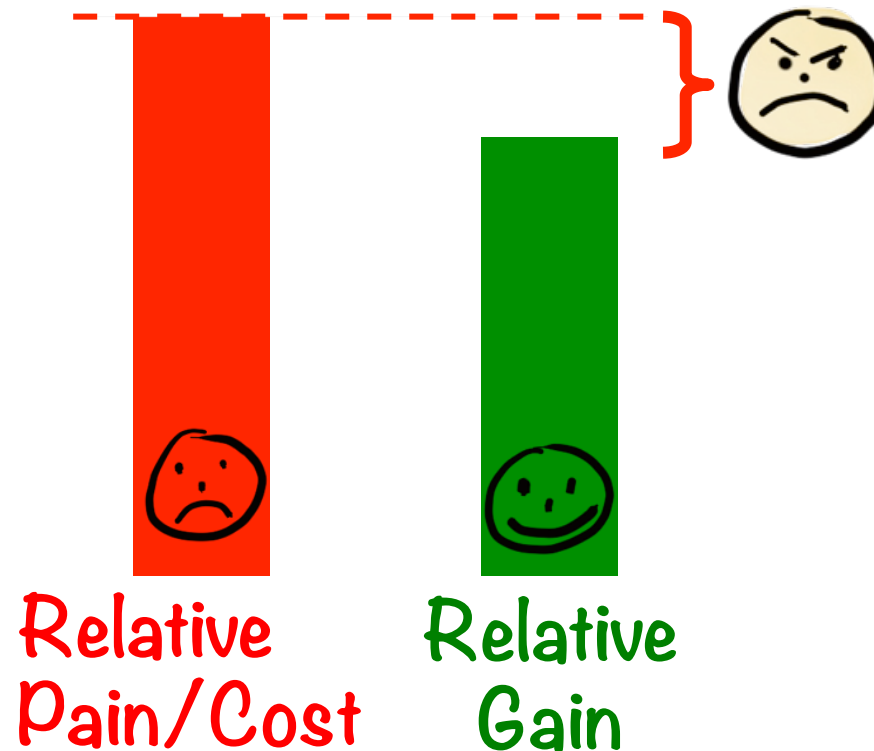


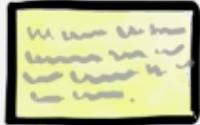
Value of
your solution

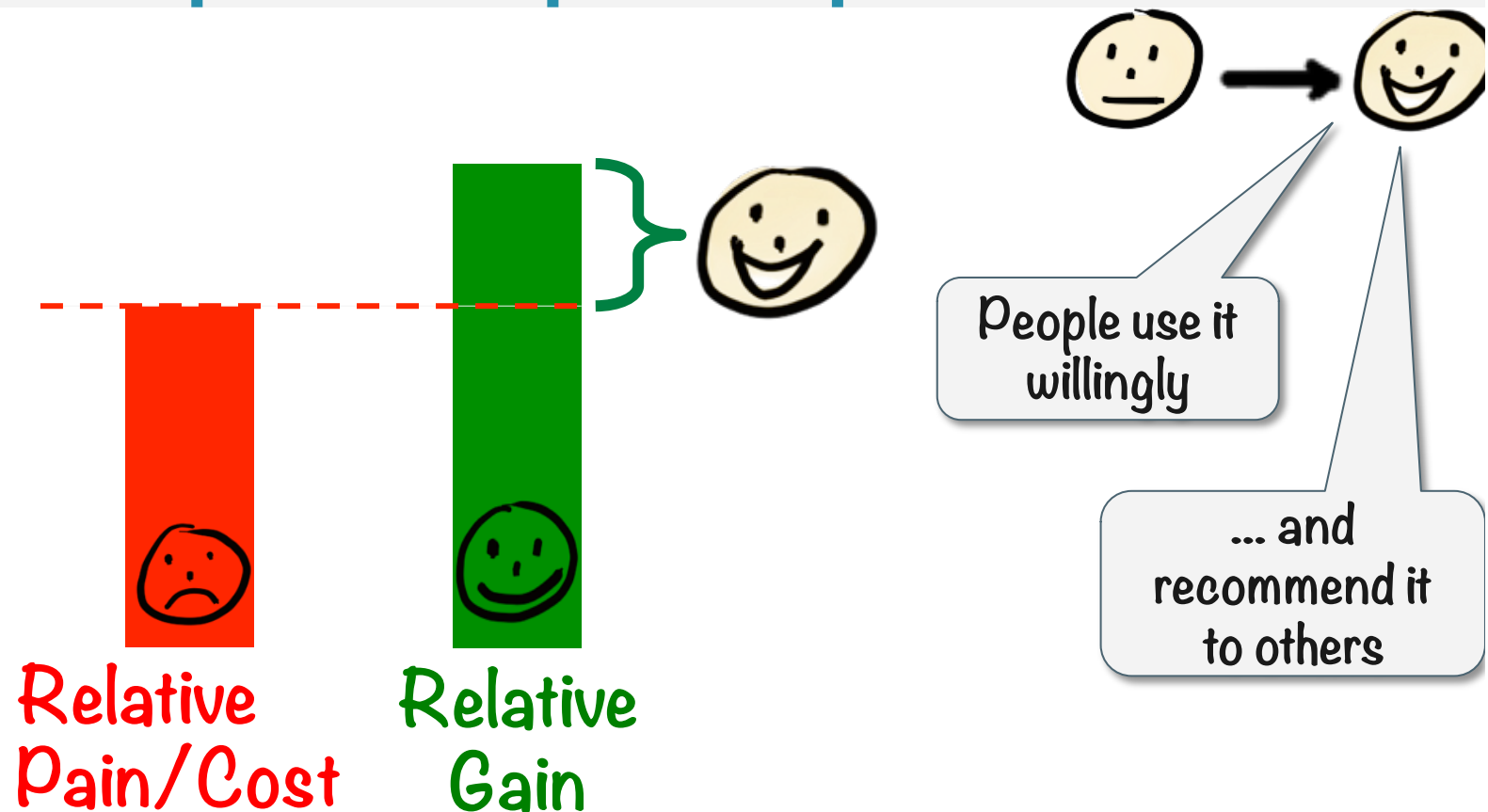
Value of old
solution



Code committed	Tested	In production	People actually use it	Solves the user's problem	... in a way that is better than before
					

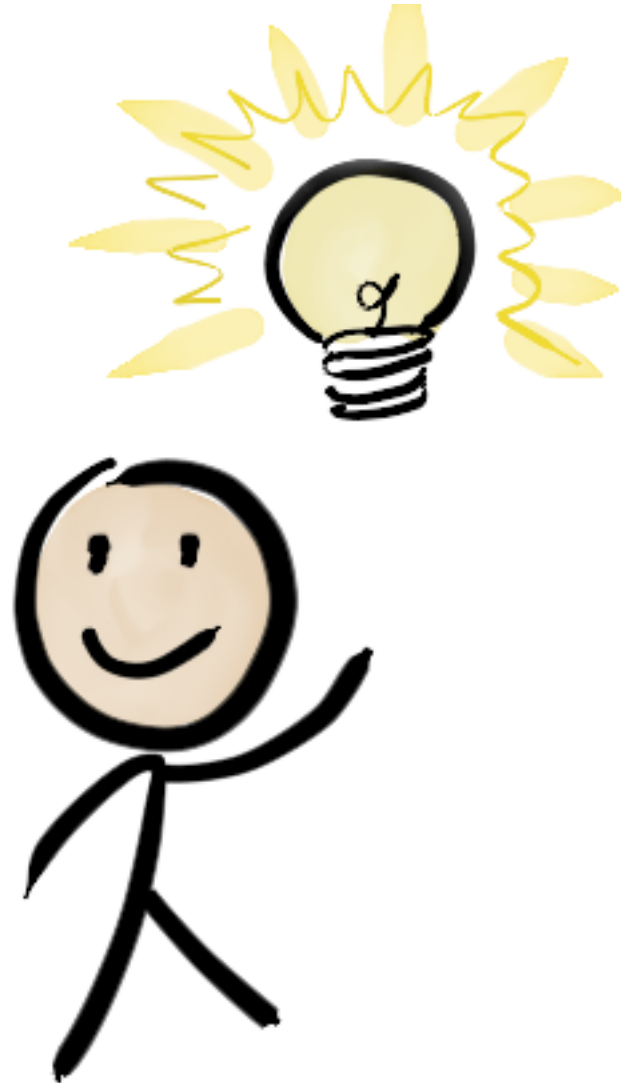


Code committed	Tested	In production	People actually use it	Solves the user's problem	... in a way that is better than before
					



Escaping the Big Bang

All products start with a Great Idea!



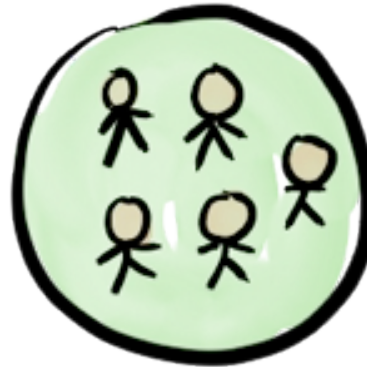
Risk



Business risk



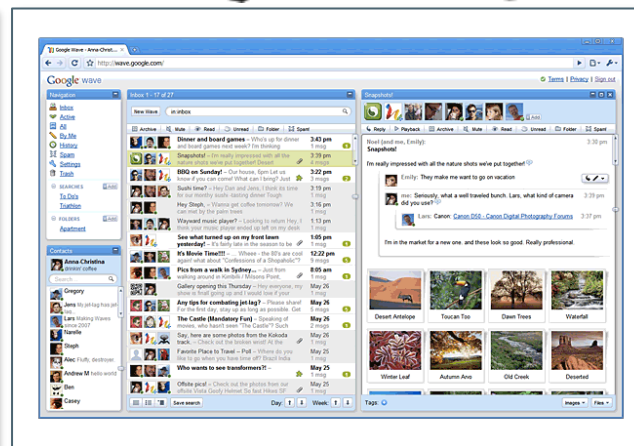
Technical risk



Social risk



Cost & schedule risk



Assume you are building the **WRONG** thing!

Suppose we're
building the wrong
thing....

How can we find out
as quickly as
possible?



- List your hypotheses
- Build an MVP to validate/invalidate it
 - "Minimum Viable Product" – the fastest & cheapest thing you can build to field-test your hypothesis

Example: Dropbox

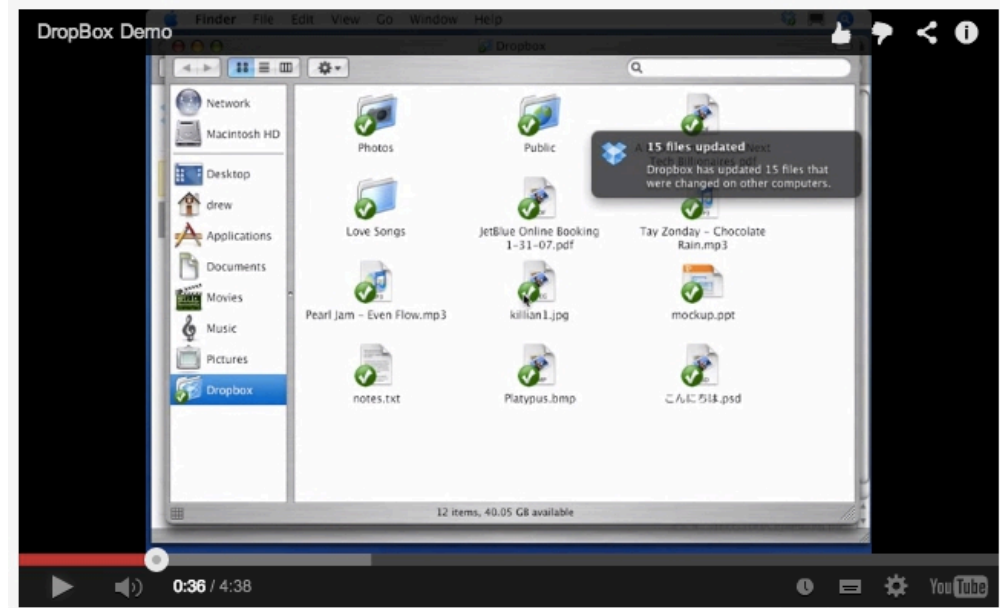
Hypotheses

1. File sync is a problem for people

2. Our product will solve the problem

3. People will want to use our product

Original dropbox demo





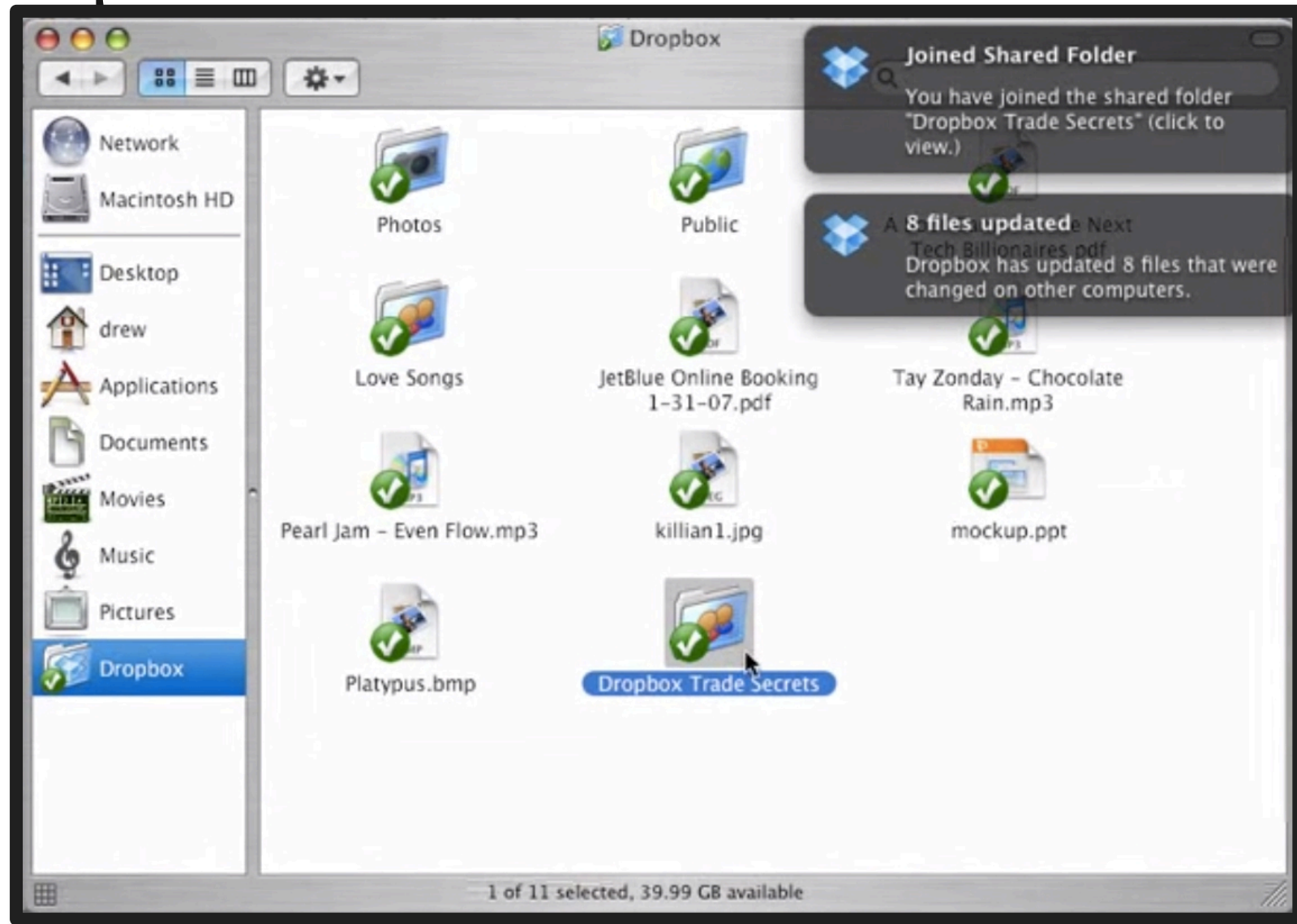
Dropbox

www.getdropbox.com

Drew Houston

beta@getdropbox.com

Dropbox MVP



Dropbox MVP



It drove hundreds of thousands of people to the website.

Our beta waiting list went from 5,000 people to 75,000 people literally overnight.

It totally blew us away.



Drew

Pirate metrics



AARRRR!

Aquisition

Do people
come?

Activation

Do they use the
product?

Retention

Do they come
back?

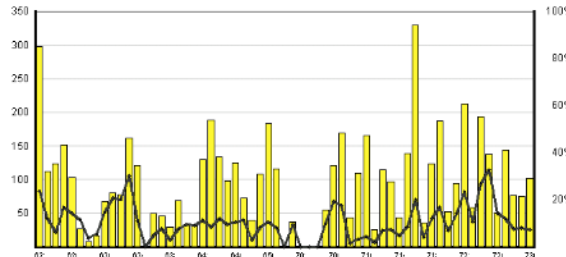
Referral

Do they recommend
it to others?

Revenue

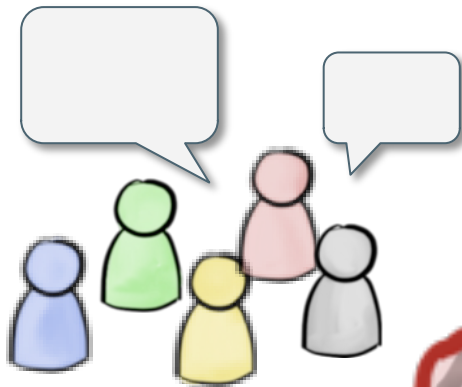
Do they pay?

Don't rely ONLY on hard data



Data & metrics

1 corner = maybe build it
2 corners = probably build it
3 corners = definitely build it

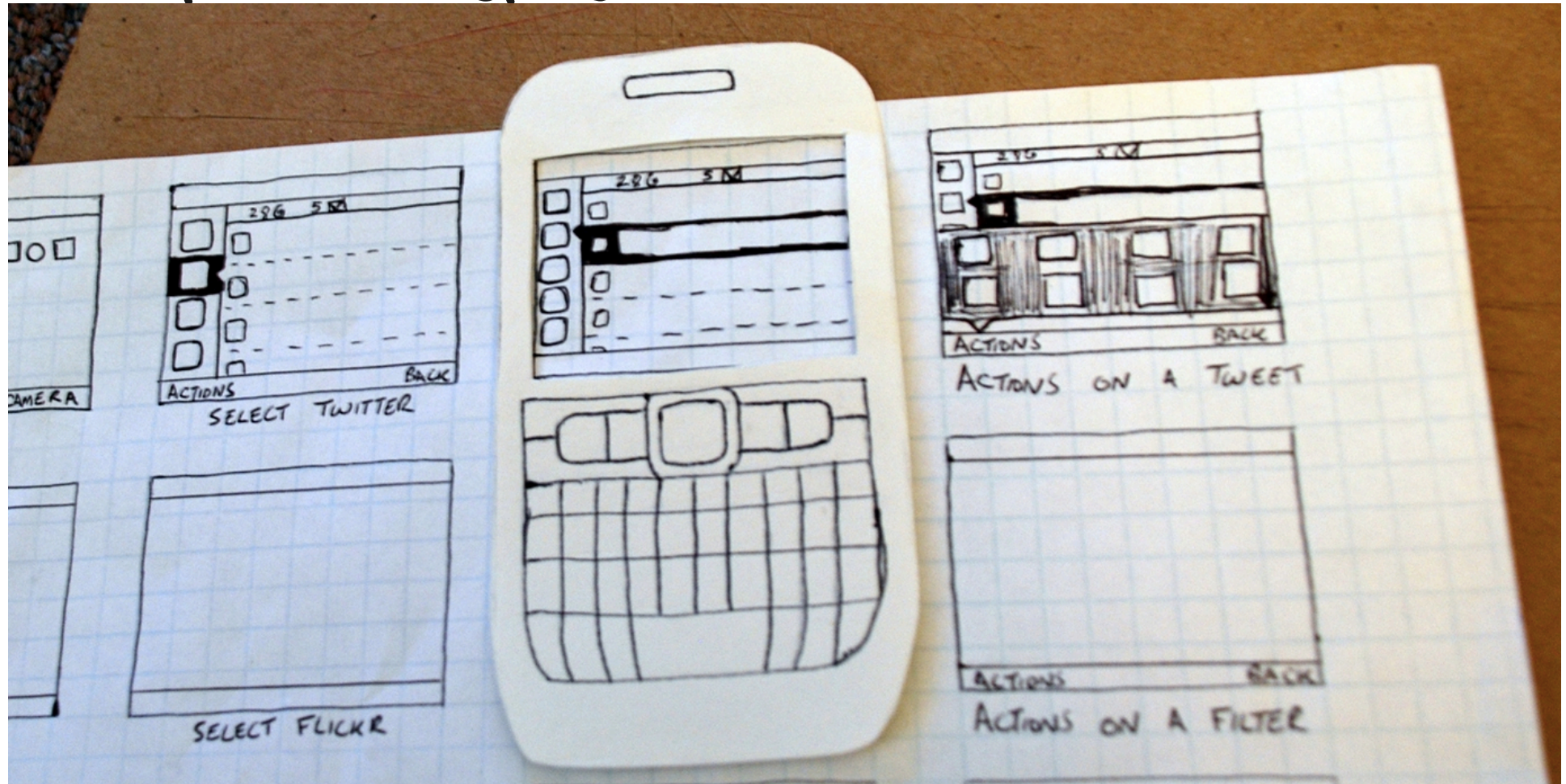


User forums
& feedback



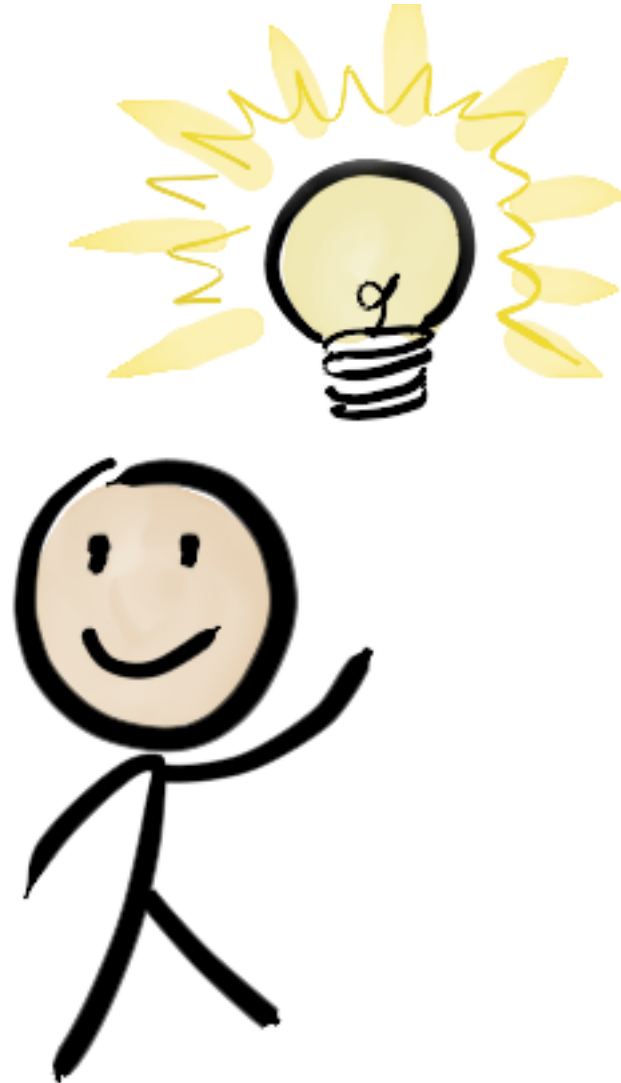
Gut feel & experience

Paper Prototyping = Lo-tech MVP

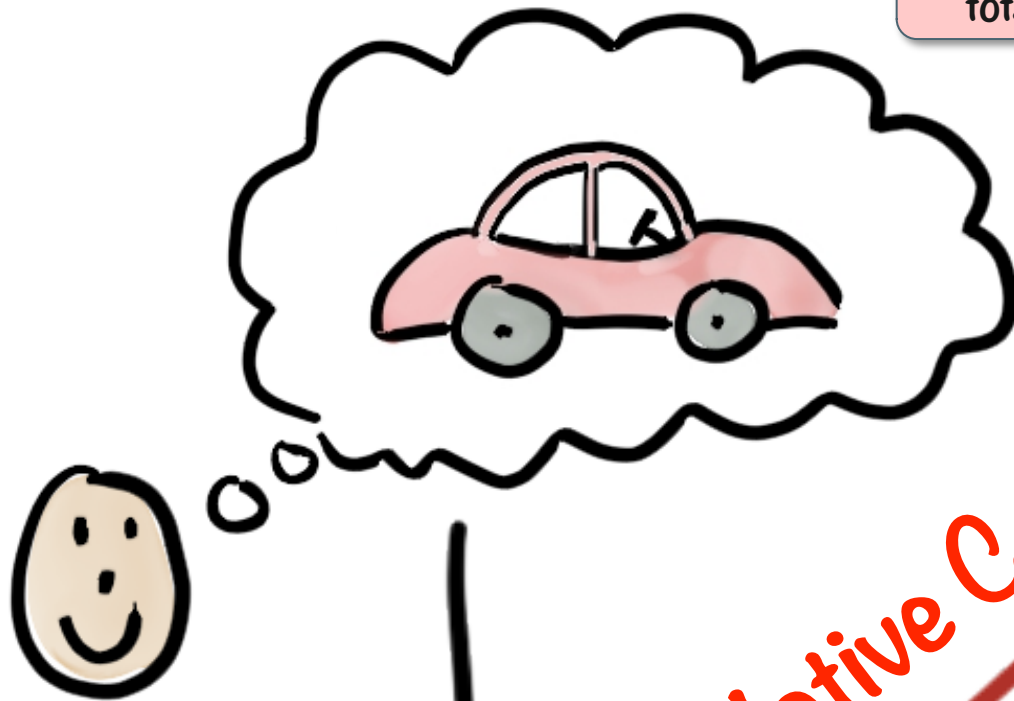


Credit: Ariel Waldman, on Interaction Design/ Rachel Ilan
<http://chiefdisruptionofficer.com/helpful-rapid-prototyping-methods-and-tools-to-bring-digital-ideas-to-life-fast/>

All products start with a Great Idea!

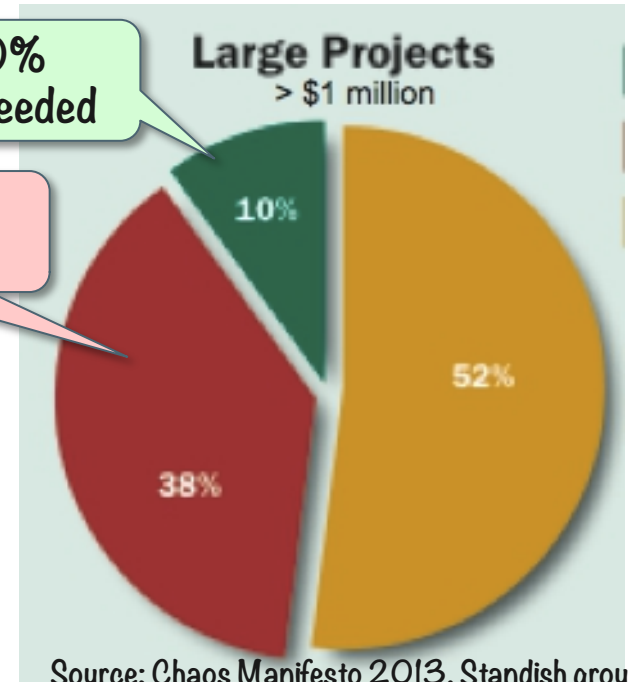


Big Bang = Big Risk



38%
totally failed!

10%
succeeded



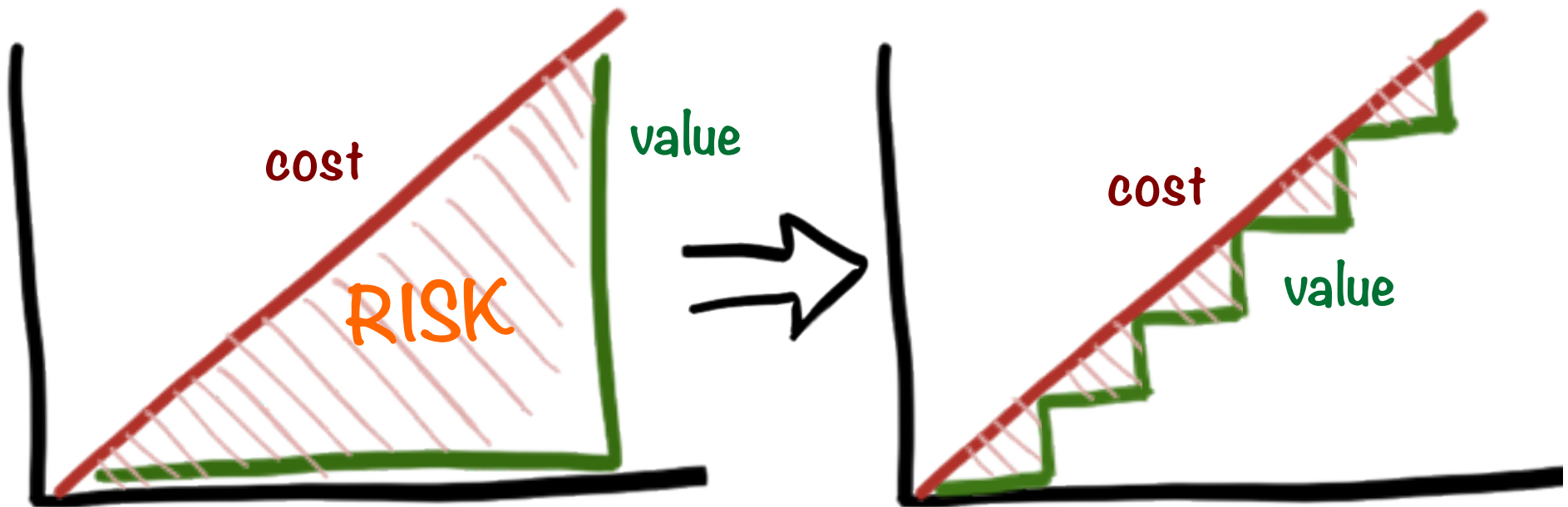
Source: Chaos Manifesto 2013, Standish group



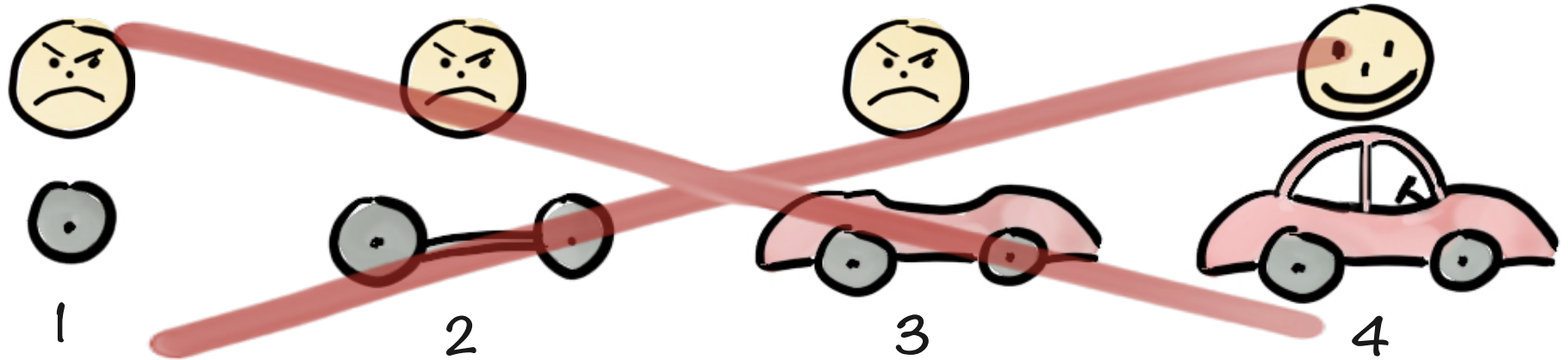
Agile/Lean = Iterative + Incremental

Don't try to get it all right
from the beginning

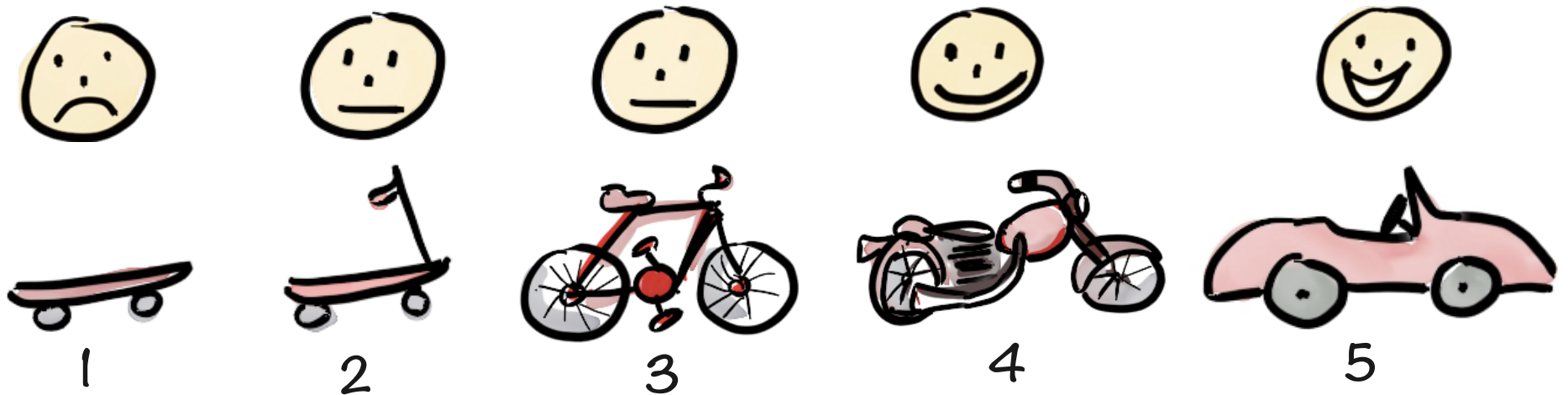
Don't build it all at once



Not like this....

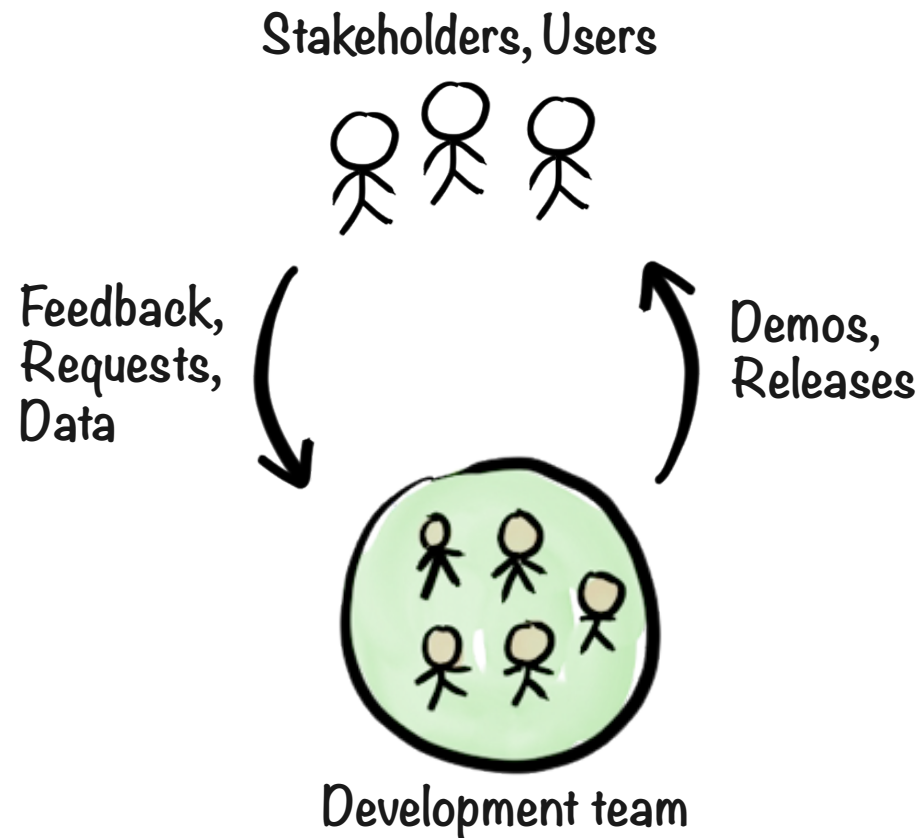


Like this!



Fastest learner wins!

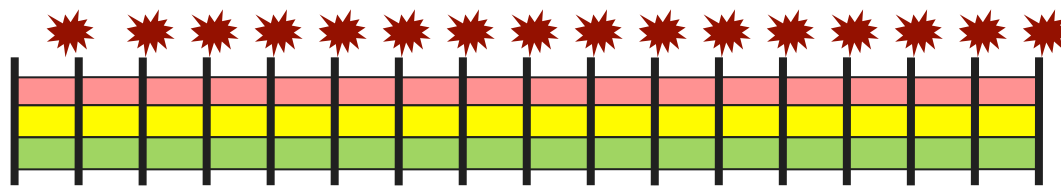
Delivery frequency = Speed of learning



Release must be REALLY easy!

Release = Drama!

Release!



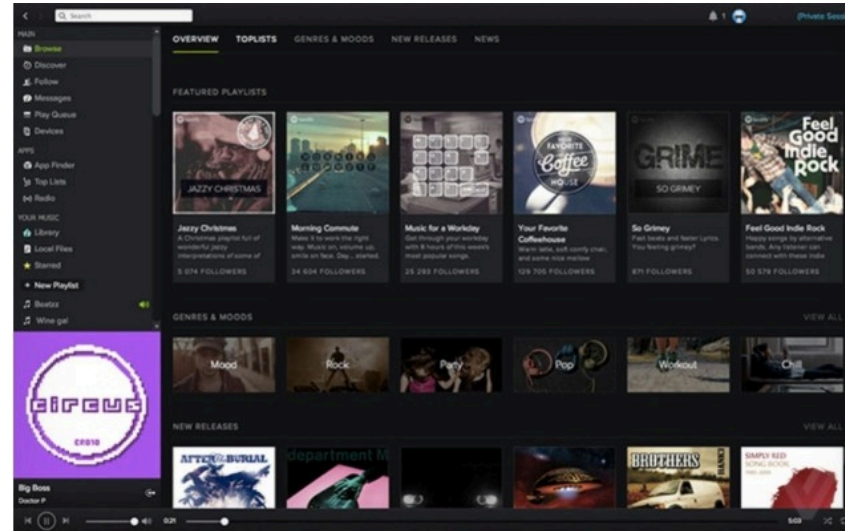
Release = Routine



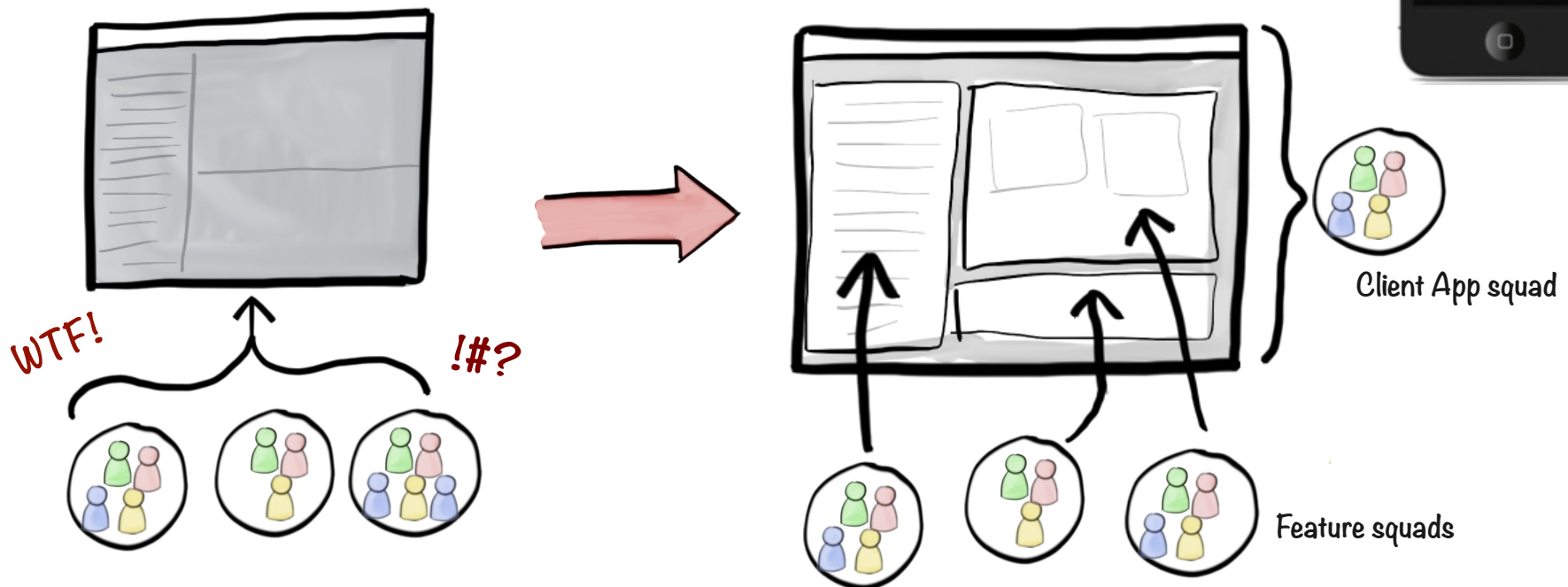
Planning gets easier with frequent releases



Decoupling to enable frequent releases



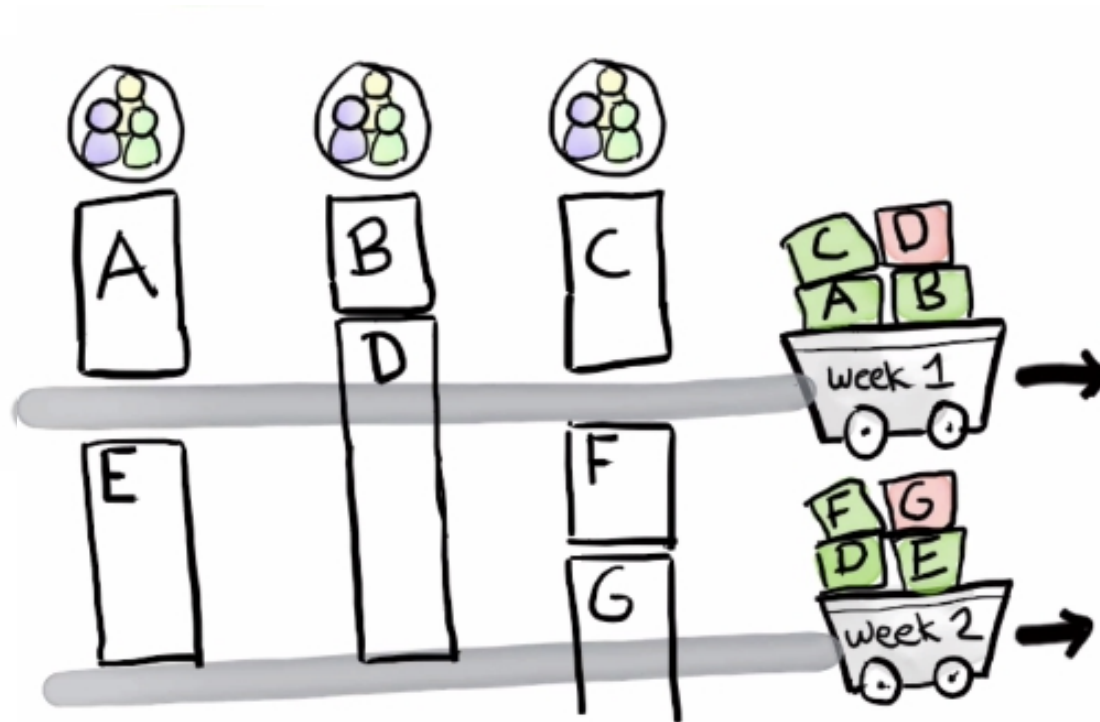
Spotify®



Continuous Delivery = Deployment is so easy
that even an Agile Coach can do it!

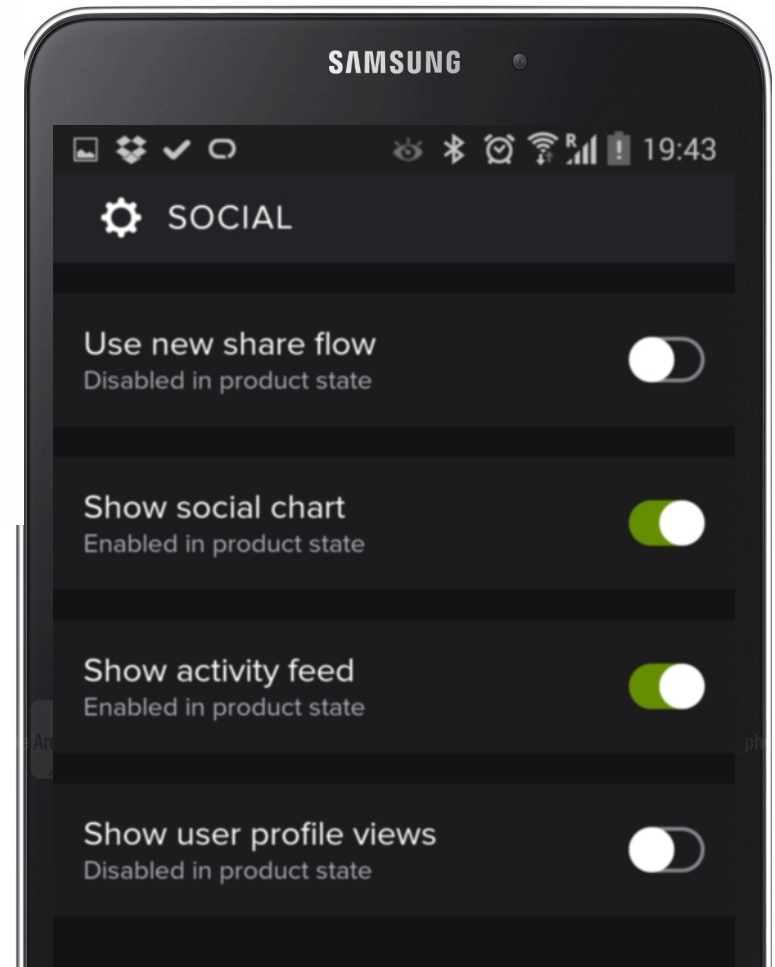


Release trains & Feature toggles



Visibility

Feature A ☒
Feature B ☒
Feature C ☒
Feature D ☐



SAMSUNG



19:42



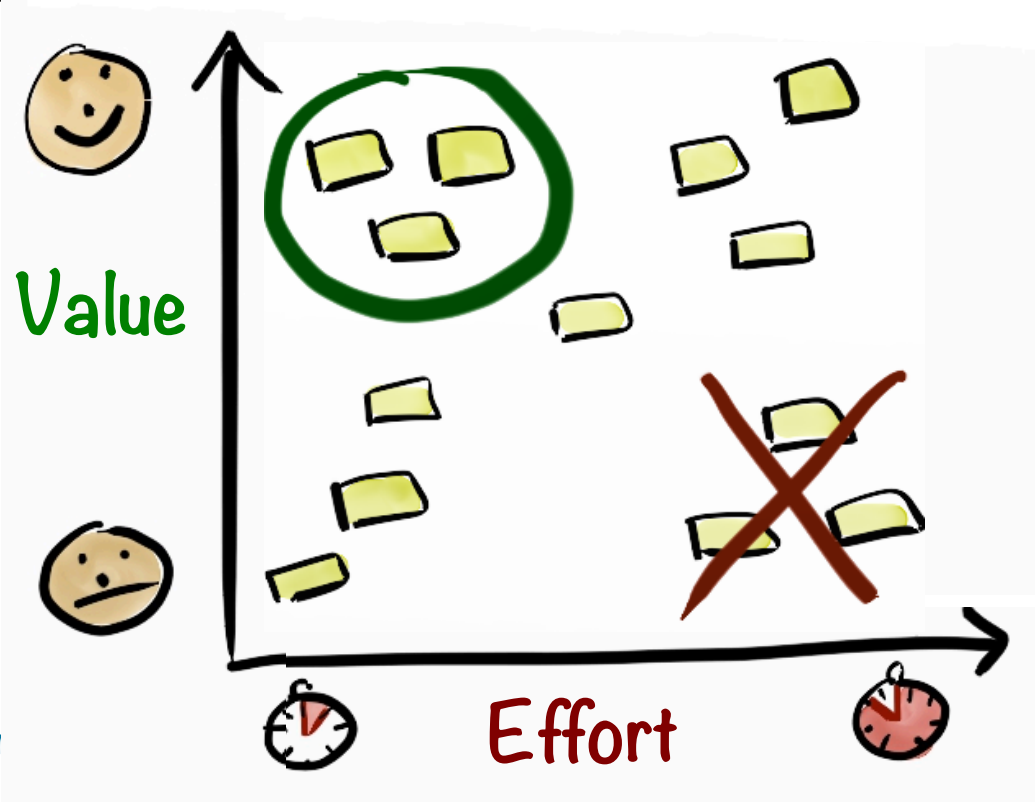
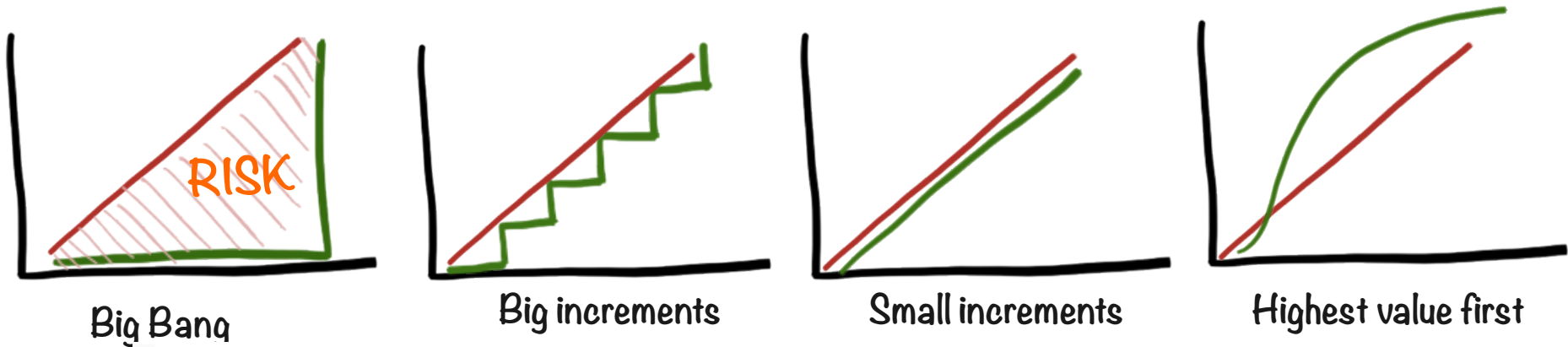
Spotify
Internal



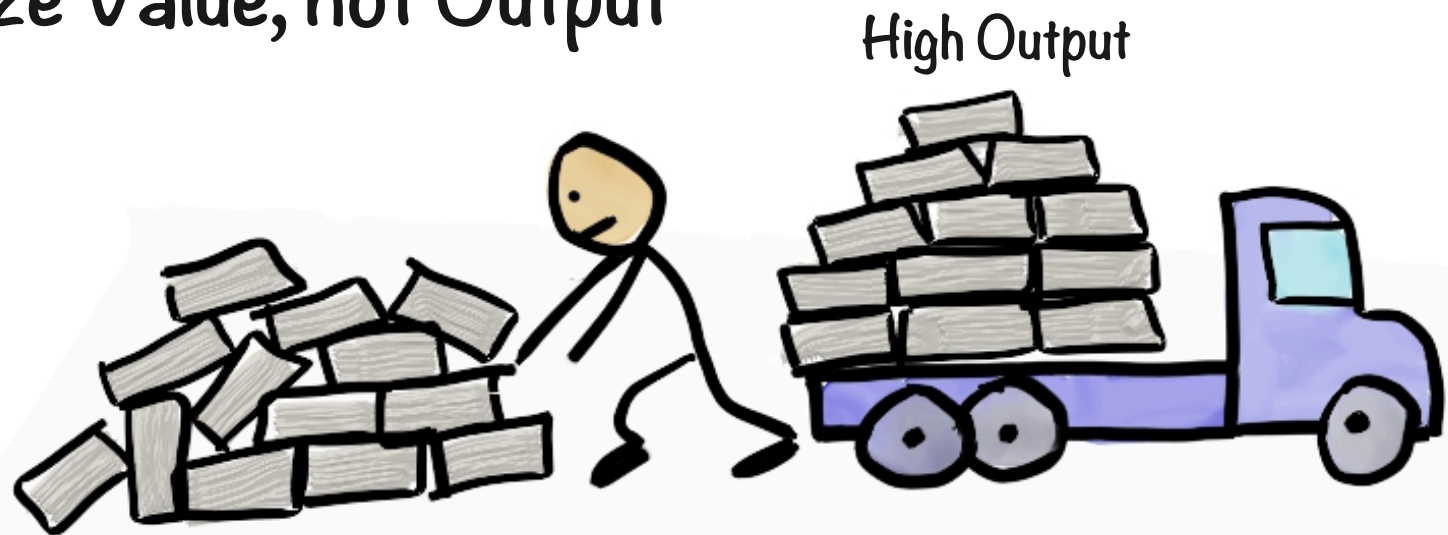
Spotify
Nightly

Hen

Improving the Value Curve



Maximize Value, not Output



Value

What is value?



How are you improving people's lives?



People can easily find the info they need



People enjoy more music



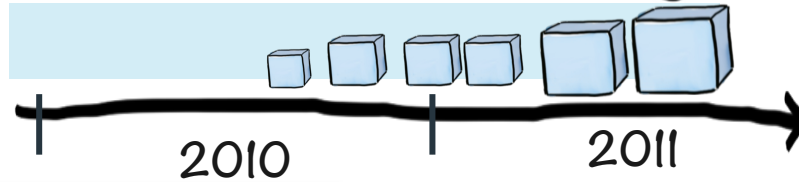
Patrolling police can do their job more effectively

Henrik Kniberg



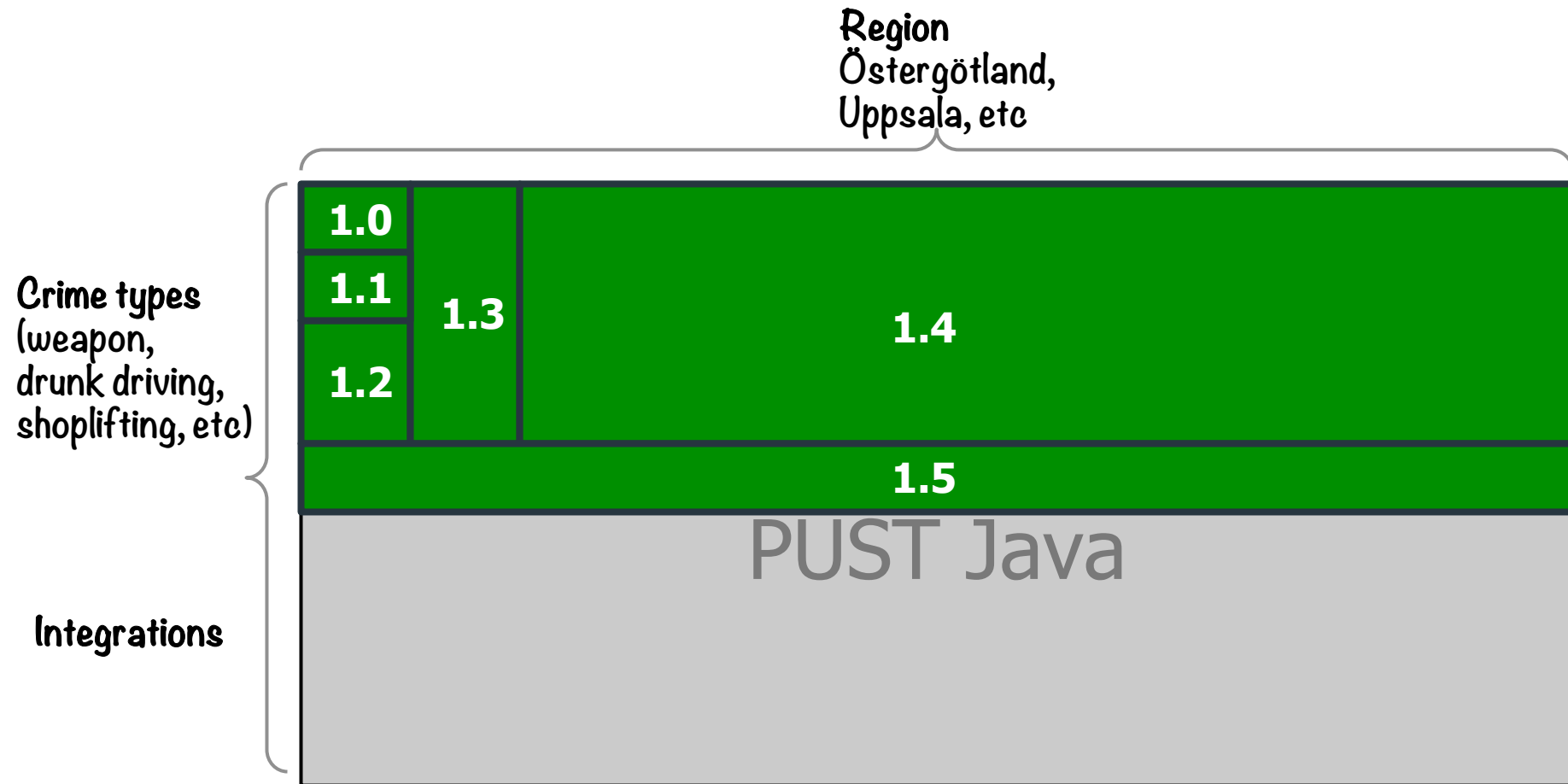
People have fun

Pust Java



Henrik K

Slice the elephant!



Pust Java



Focus on value & impact

Iterative delivery

2010

2011

Pust Siebel



Focus on feature-completeness

Big Bang delivery

2012

2013



Henrik K

How are you improving people's lives?



People can easily find the info they need



People enjoy more music



Patrolling police can do their job more effectively



People have fun

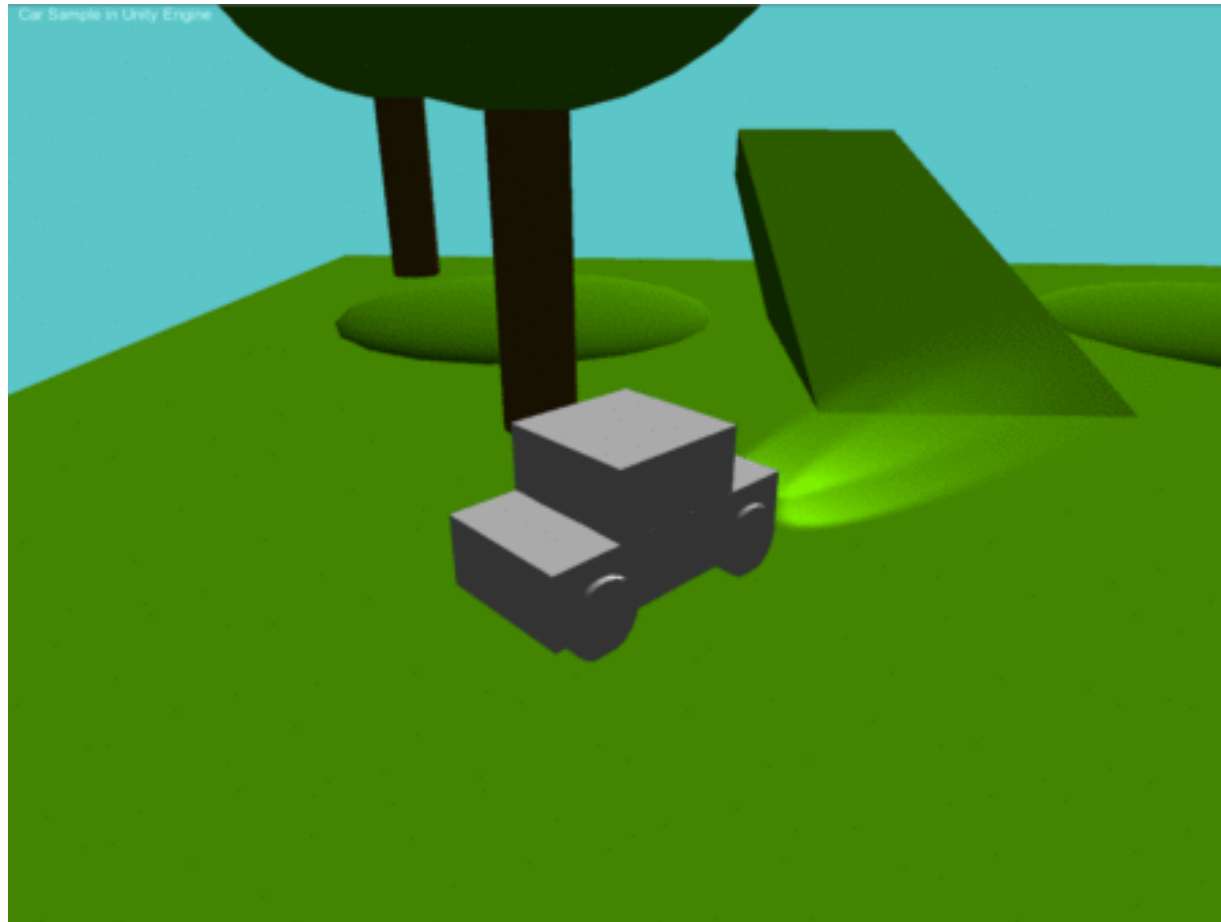


Henrik Kniberg

Done level 1: "sufficient for decision"



Done level 2: "sufficient for playtesting"



Done level 3: "sufficient for alpha/beta release"



Done level 4: "sufficient for final release"

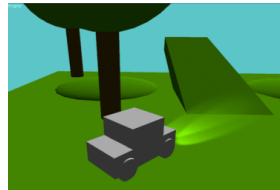


Henrik Kniberg

Finding the fun – step by step



Sufficient
for decision



Sufficient
for playtest



Sufficient for
alpha/beta



Sufficient
for release

	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	Feature 6	Feature 7	Feature 8
Sufficient for decision			1			3		6
Sufficient for playtest		2		3		5		
Sufficient for alpha/beta		4			6			
Sufficient for release			7					



TOCA BOCA

Find out what's fun -
BEFORE building the product!



Henrik Kniberg

Definition of Fun



★★★★★ (90)

Rating: 4+

Made For Ages 6-8

How are you improving people's lives?



People can easily find the info they need



Patrolling police can do their job more effectively



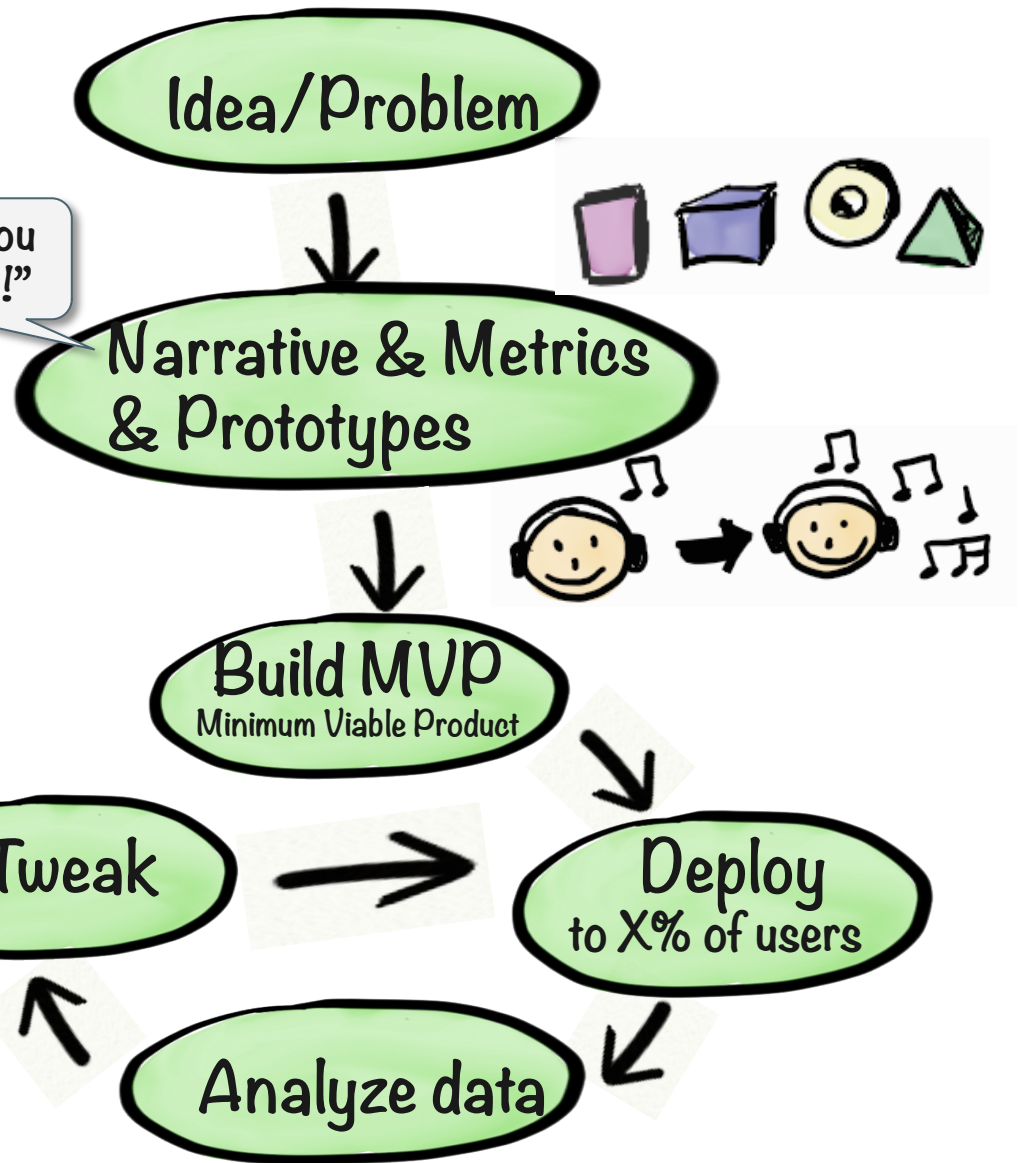
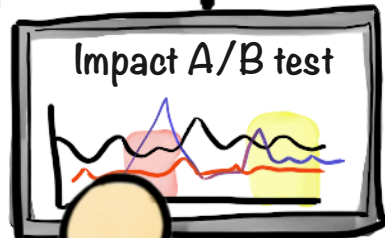
People enjoy more music



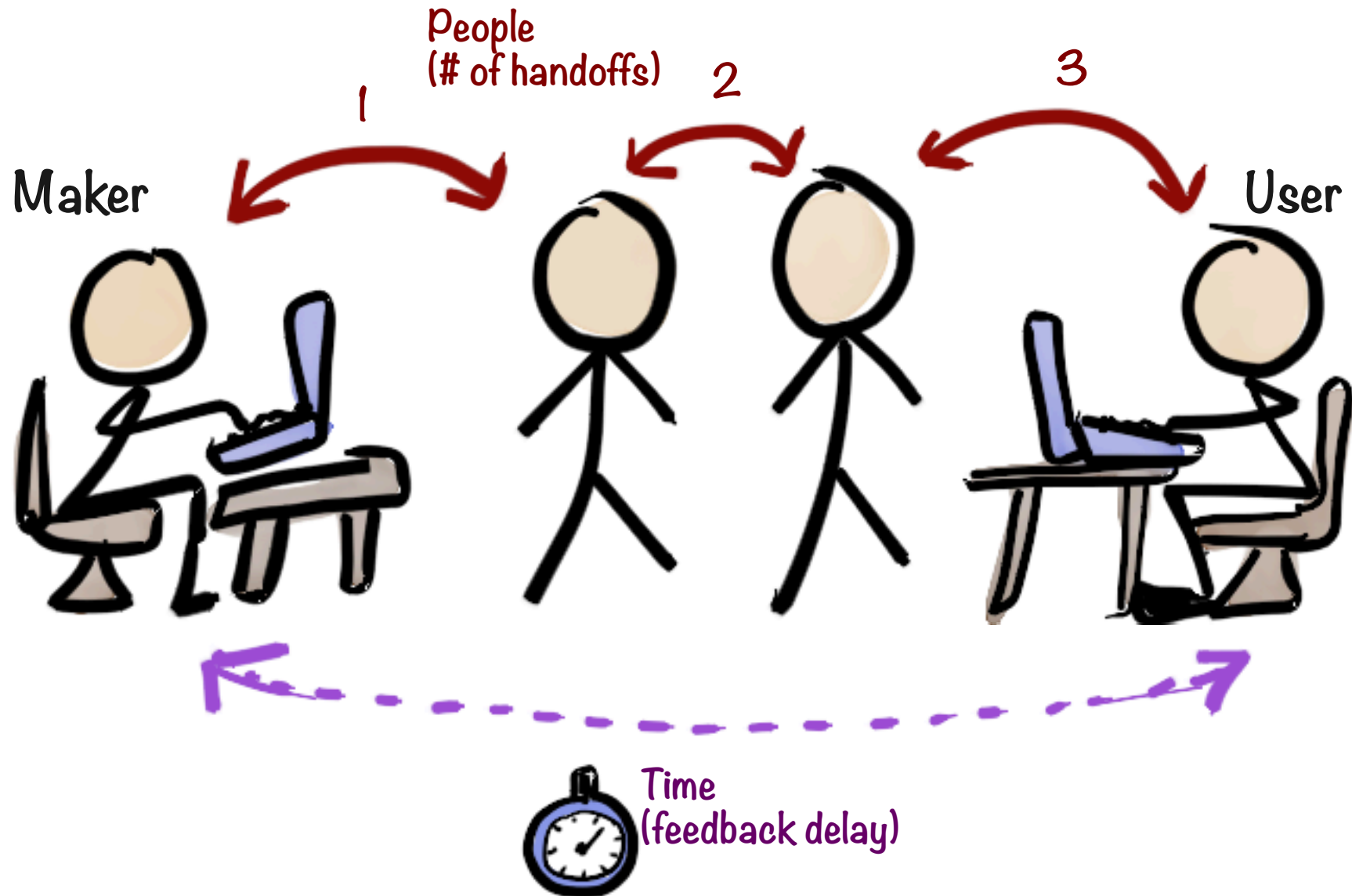
People have fun



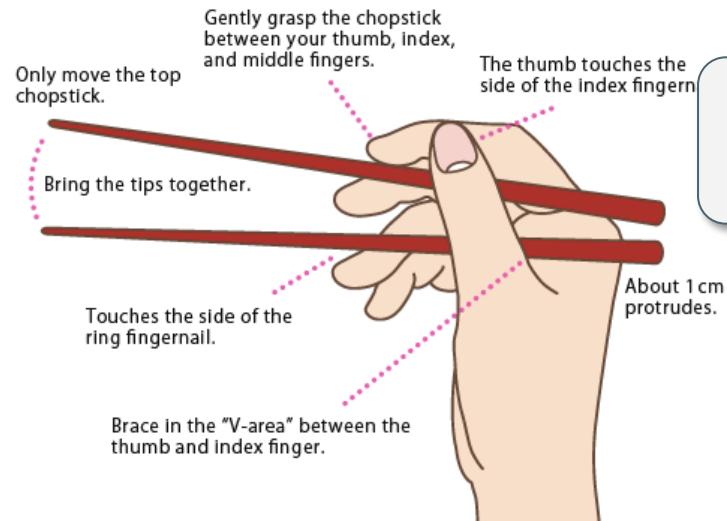
"Radio you can save!"



Minimize distance between Maker and User



Wrapup



There are no hard problems.
Just hard solutions.



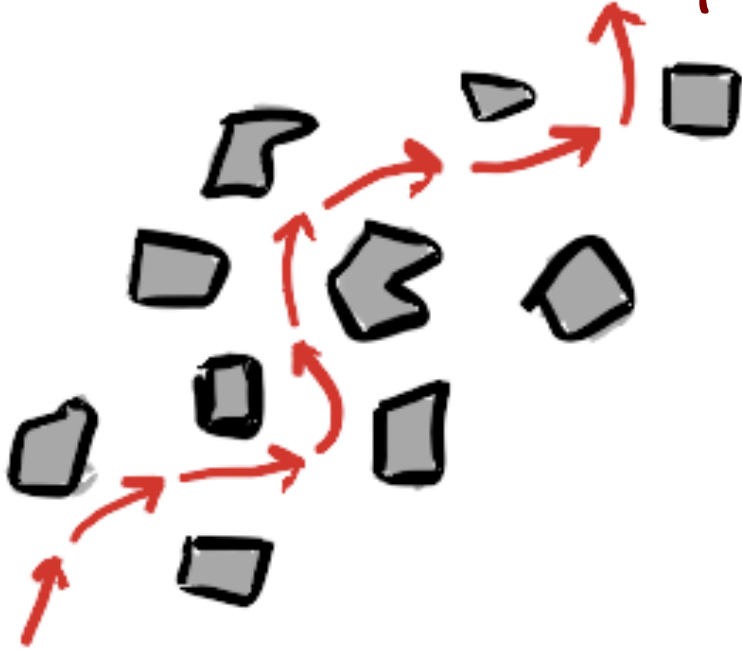
Jerry Weinberg



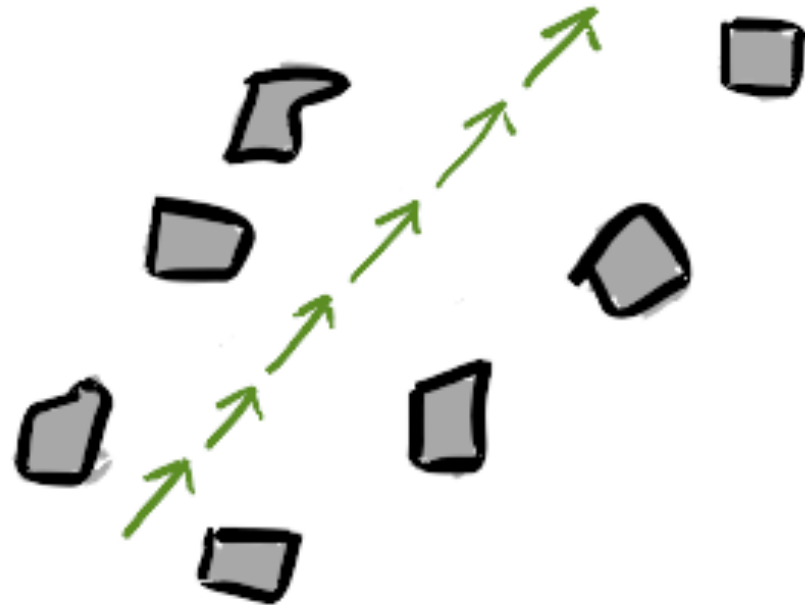
He

Lean = get fast by reducing waste

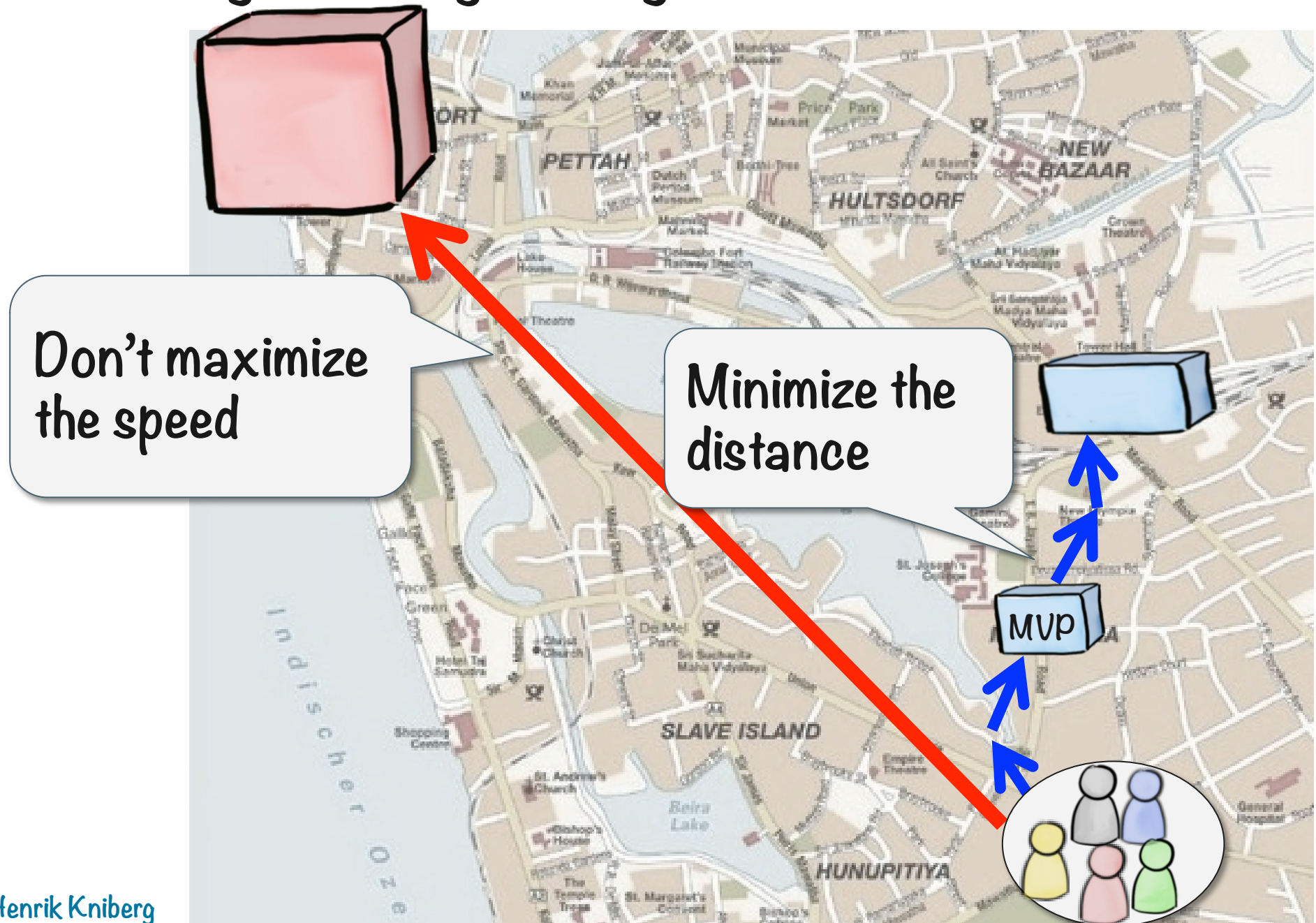
Don't increase the speed



Reduce the distance!



Lean = get fast by building less stuff



What you measure is what you get



Focusing on
Effort



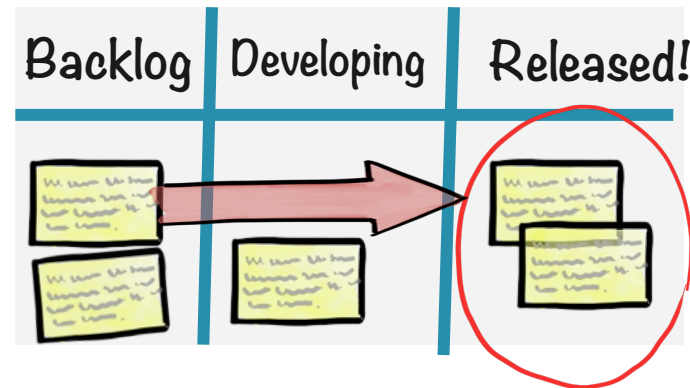
Hours

Time reports

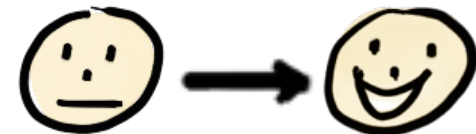
Resource utilization



Focusing on
Output



Focusing on
Outcome/Impact/Value



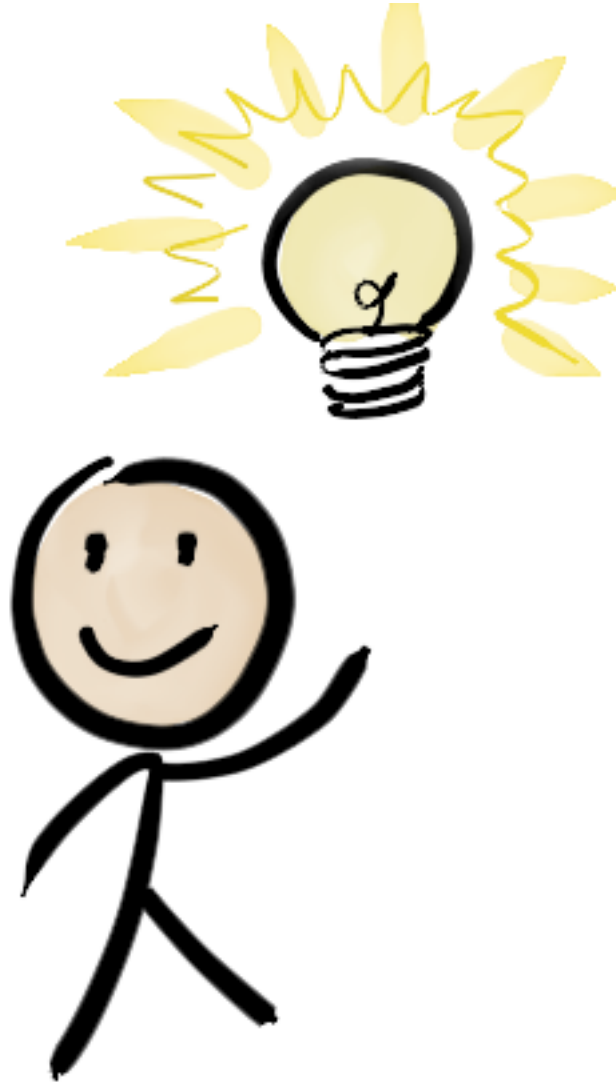
Manifesto for Agile **Product** Development

We are uncovering better ways of developing software by doing it and helping others do it.
Through this work we have come to value:

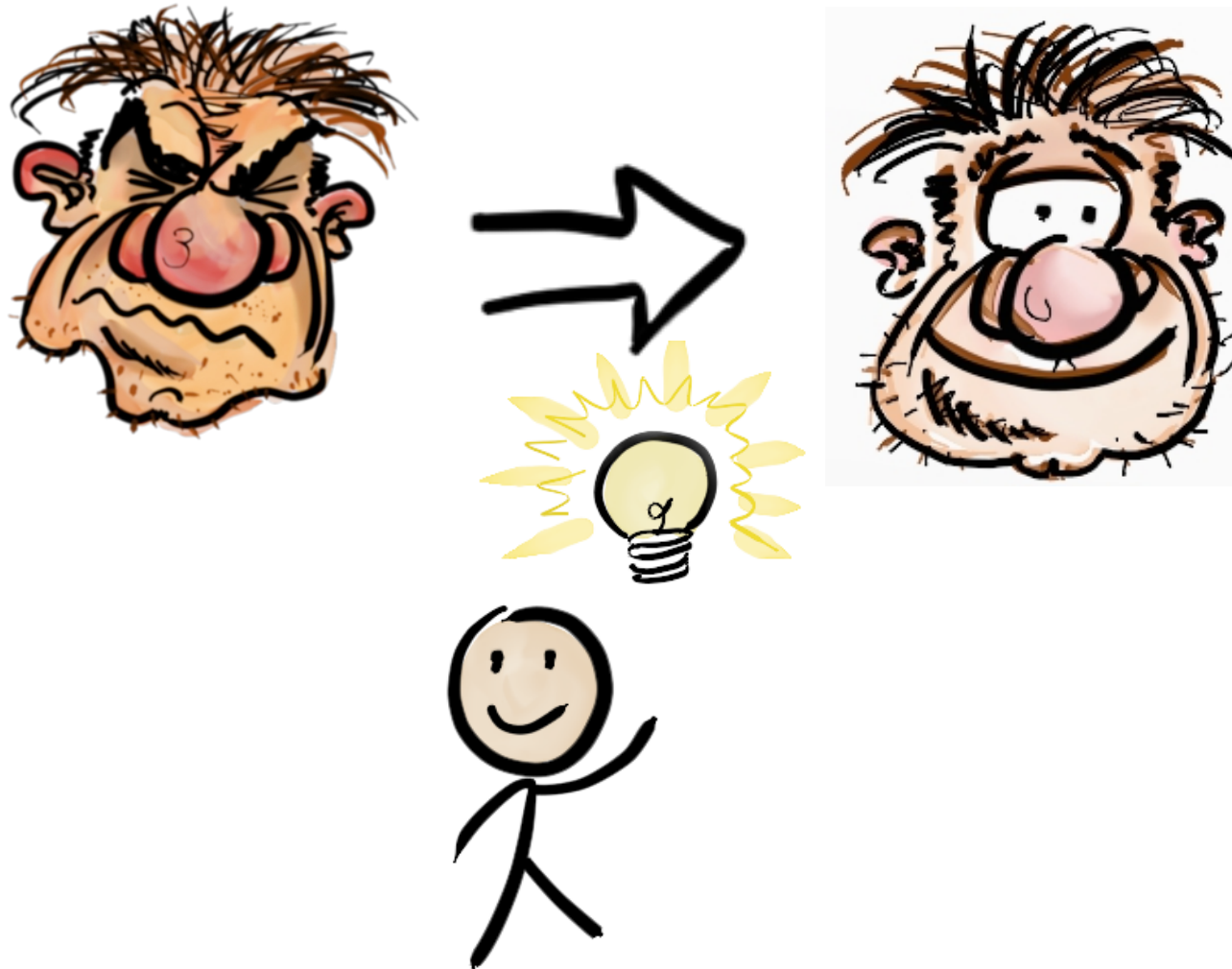
Individuals and interactions over processes and tools
Working **Product** over comprehensive documentation
User collaboration over contract negotiation
Responding to **Feedback** over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

All products start with a Great Idea!



All products start with a ~~Great Idea!~~
an Unsolved Problem!



How do you ^{ensure}~~know~~ that your product works?

1. Understand the problem

Who are the stakeholders?

What need do they have, that we want to solve?

How will we know when we've solved it?

How will we know if we're moving in the right direction?

2. Iterate until you've solved it

Minimize the distance to MVP

Deliver, measure, adjust continuously

PDCA....
Build Measure Learn....
Inspect Adapt...
Probe Sense Respond...