



 **@billwscott**

Sr. Director
User Interface Engineering
@paypal

Digital Garage/NEO
San Francisco, CA
Nov 6, 2013

transforming engineering into a lean partner
operational readiness at scale



netflix view of engineering

continuous customer feedback (GOOB)

customer metrics drive everything

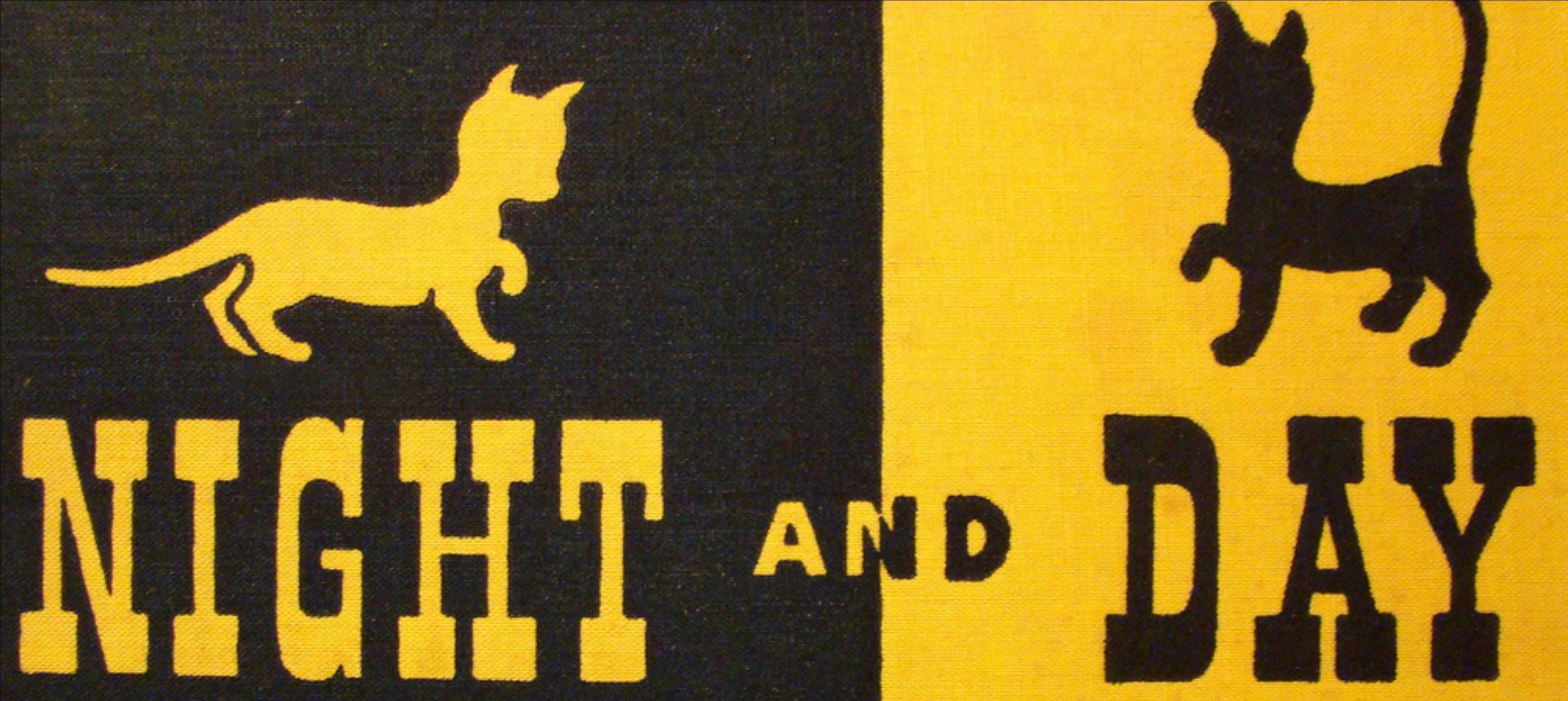
think it. build it. ship it. tweak it

fail fast. learn fast.

lots of experimentation... build/measure/learn

small teams with lots of collaboration

engineering for experimentation



paypal vs netflix

contrast this with a large enterprise like paypal (circa 2011)

Two Pizza Rule

**if a team can't be fed by two pizzas
then it is too big**

- Jeff Bezos



two pizza team?

paypal way of engineering (in 2011)

roll your own. disconnected delivery
experience. **culture of long shelf life.**
inward focus. risk averse.



Home

Individuals

Business

Partners

Get started

How it works

Buying safely

Selling safely

Donate to Charity

GET THE MOST OUT OF PayPal

Managing Your Account



Your account is very easy to manage.
Select a demo chapter to see how to:

 [Manage Your Account](#)

 [Update Your Email Address](#)

 [Link Your Credit Card or Bank Account](#)

LOG IN

New to PayPal? [Sign Up](#)



Home

Individuals

Business

Partners

Get started

How it works

Buying safely

Selling safely

Donate to Charity

GET THE MOST OUT OF PayPal

Managing Your Account



In 2011, even a simple content copy change could take as much as 6 weeks to get live to site

Your account is very easy to manage.
Select a demo chapter to see how to:

[▶ Manage Your Account](#)

[▶ Update Your Email Address](#)

[▶ Link Your Credit Card or Bank Account](#)

LOG IN

New to PayPal? [Sign Up](#)

new dna inserted

jan 2012

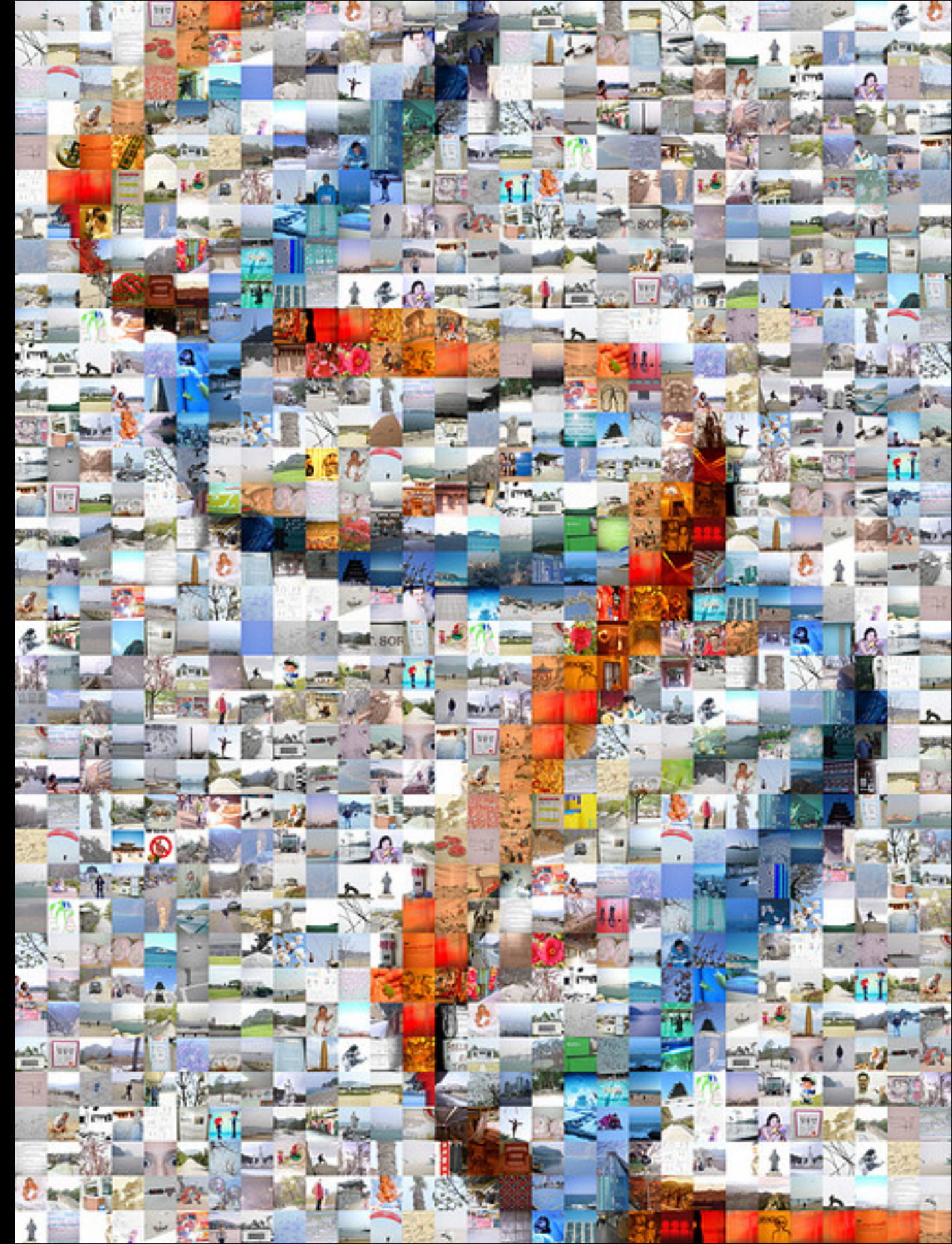
fleshed out ui layer that could support rapid experimentation

march 2012

david Marcus becomes president of PayPal

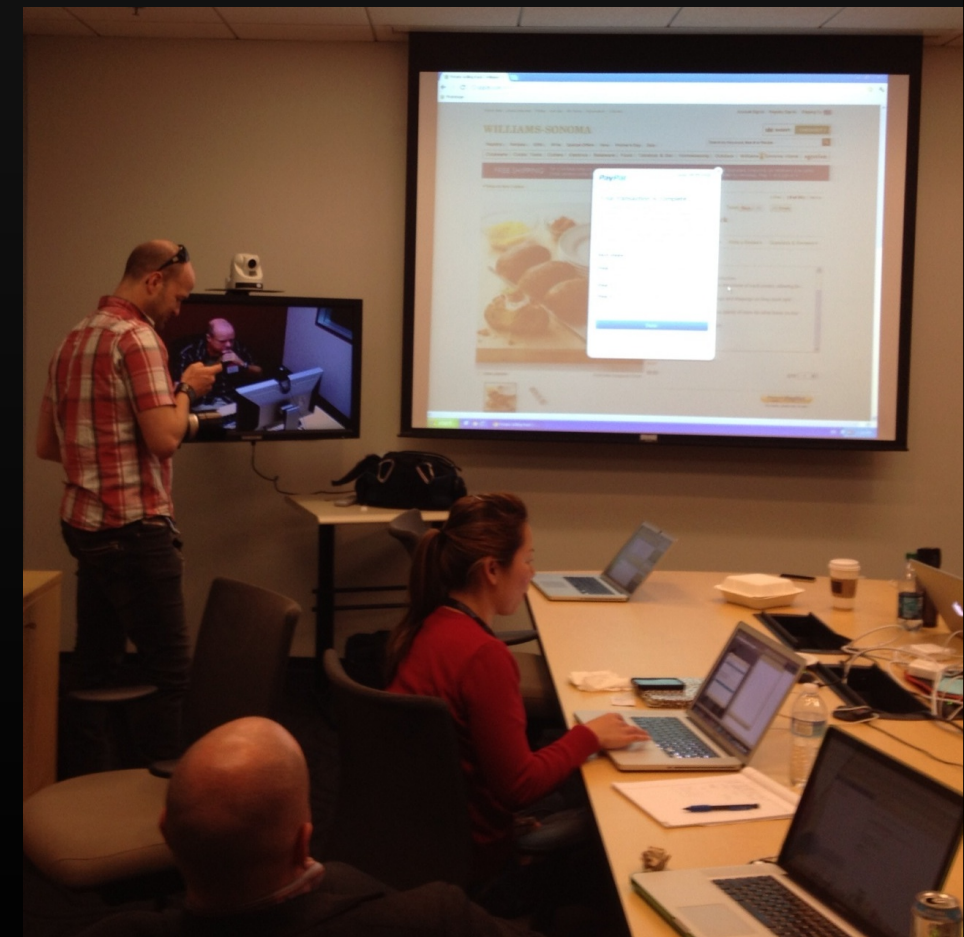
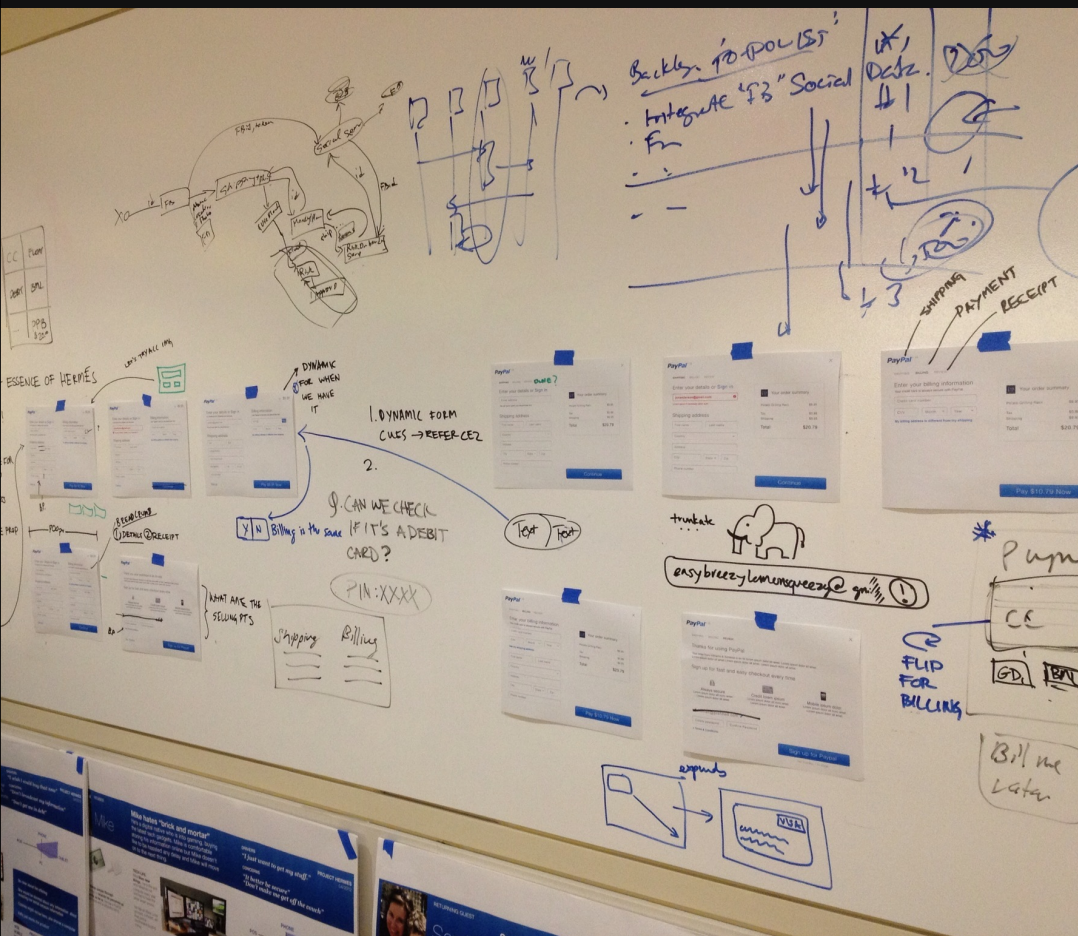
april 2012

formed lean ux team to reinvent checkout experience



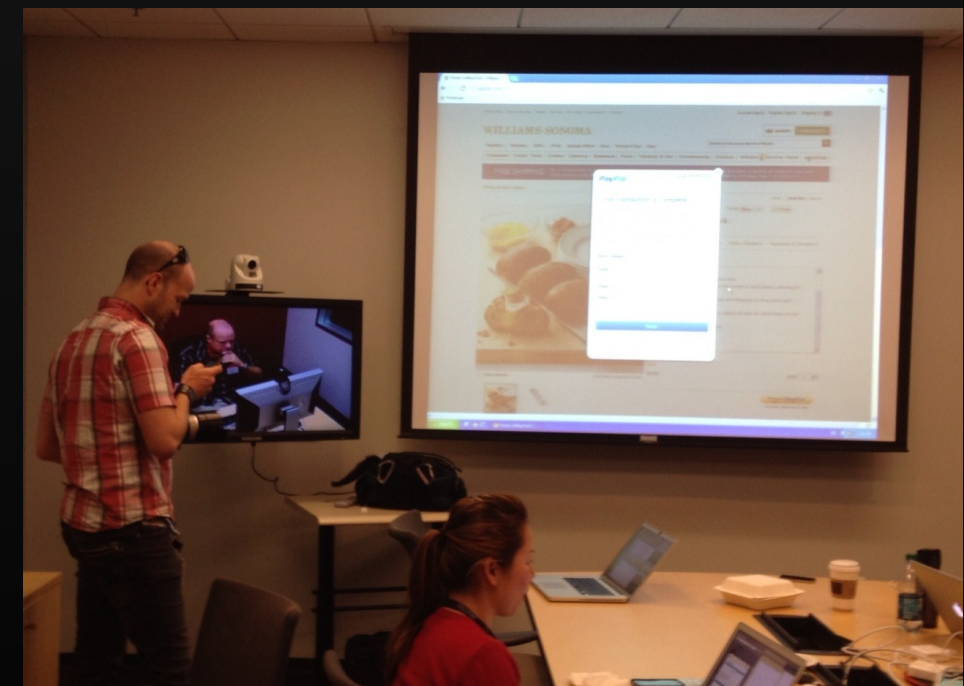
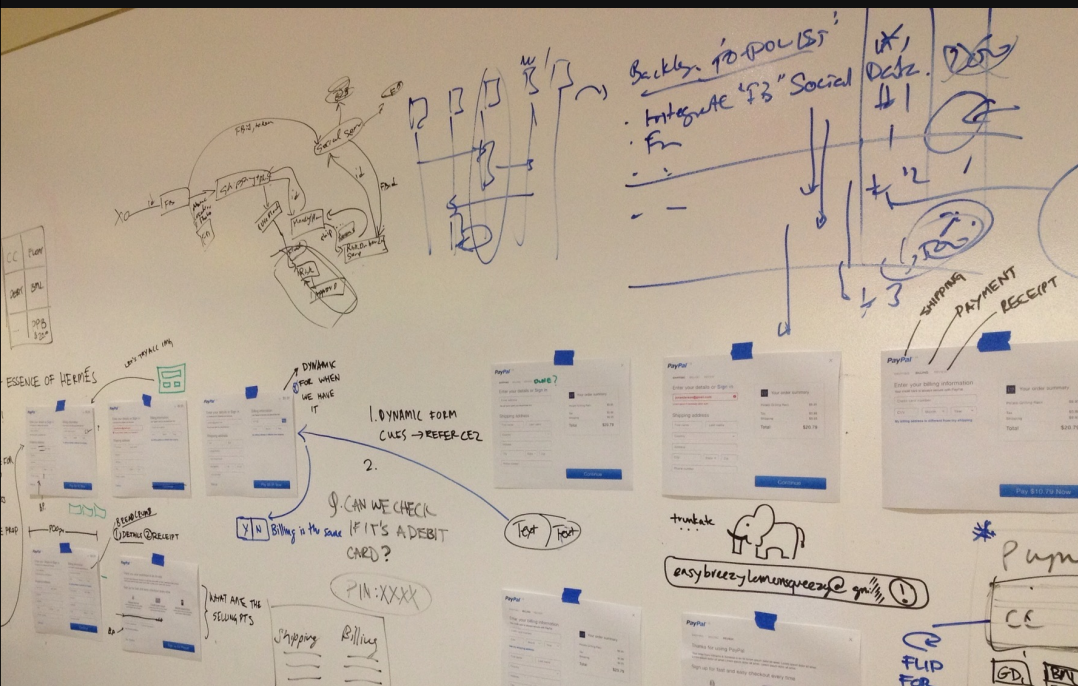
“hermes” project

lean ux
putting product, design, engineering together



“hermes” project

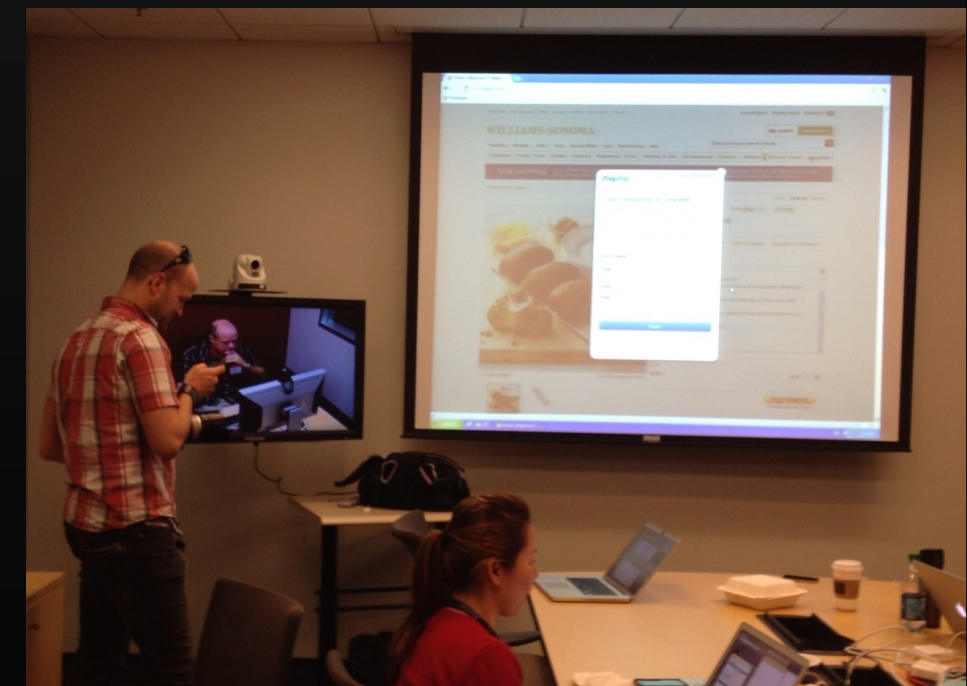
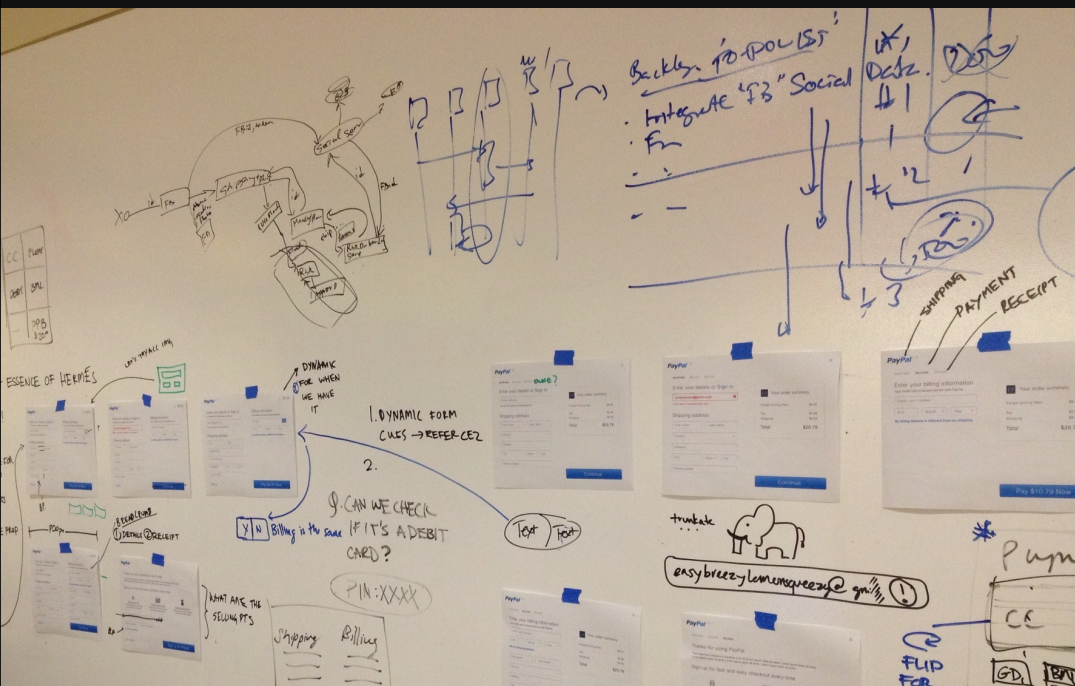
lean ux
putting product, design, engineering together



from whiteboard to code →

“hermes” project

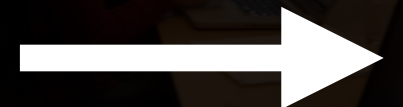
lean ux
putting product, design, engineering together



from whiteboard to code

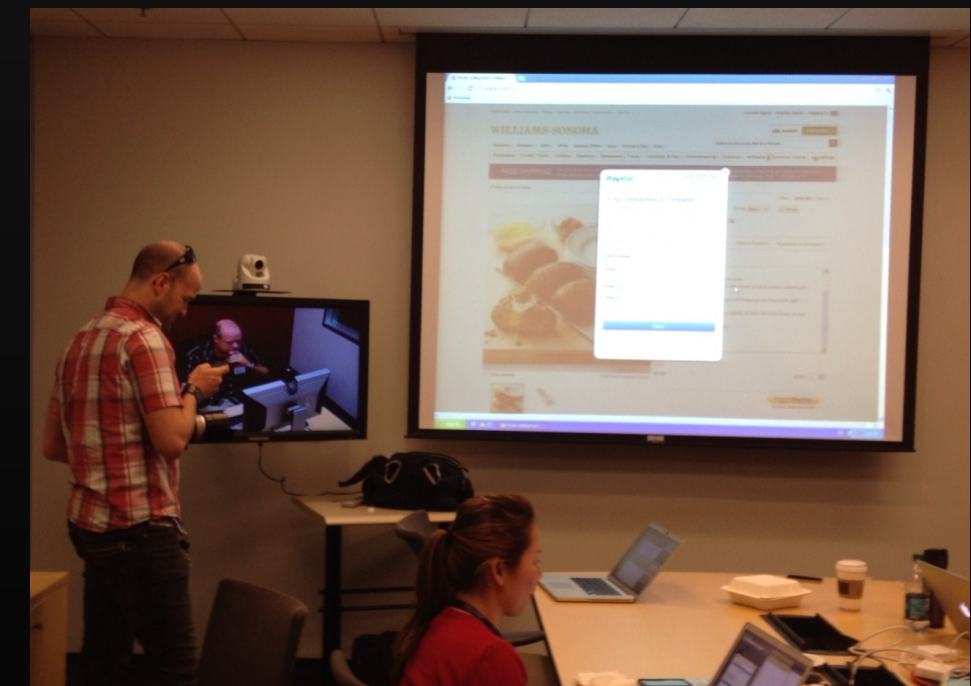
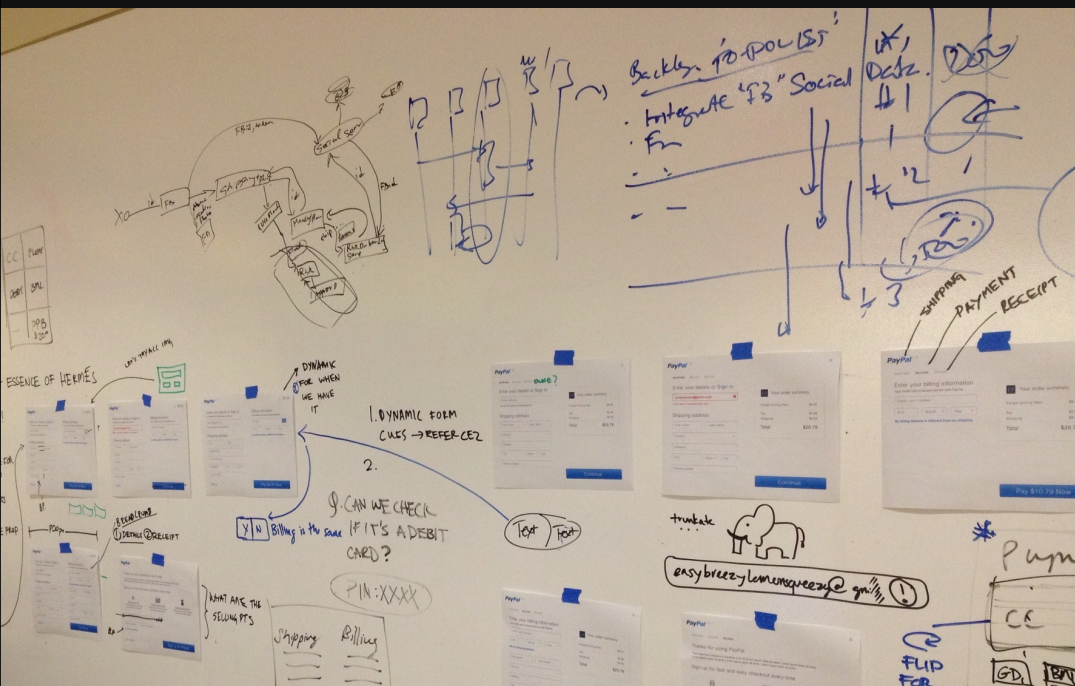


from code to usability



“hermes” project

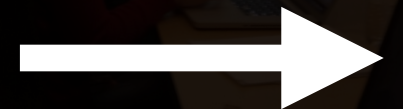
lean ux
putting product, design, engineering together



from whiteboard to code



from code to usability

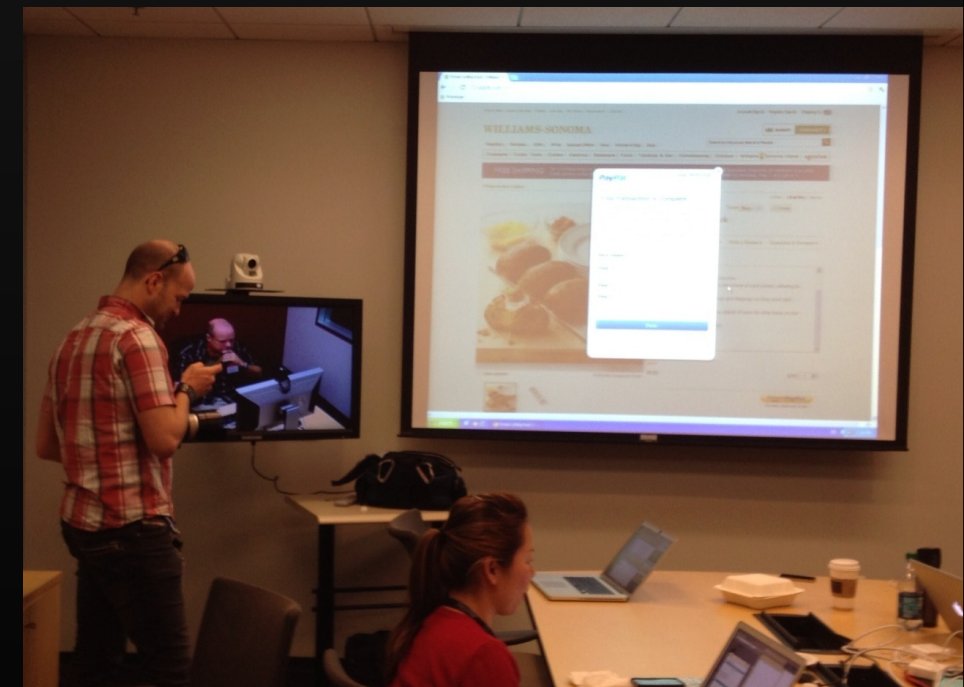
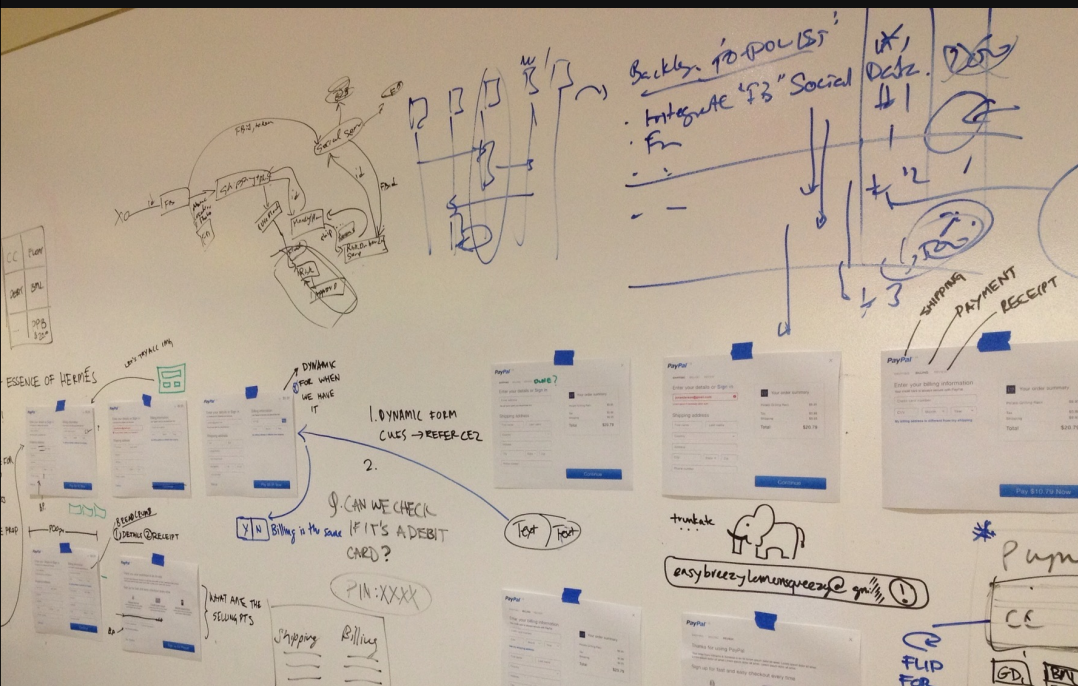


learnings



“hermes” project

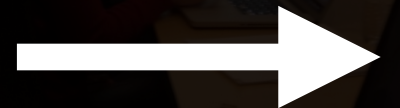
lean ux
putting product, design, engineering together



from whiteboard to code



from code to usability



start again



learnings



before

My Volusion Store

Your order summary

Descriptions	Amount
Order Sub-Total	\$60.00
Item price: \$60.00	
Quantity: 1	
<hr/>	
Item total	\$60.00
Tax	\$7.80
Shipping and handling:	\$12.71
<hr/>	
Total \$80.51 USD	

Review your information



Continue

Shipping address [Change](#)


Valued Customer
123 Street Dr.
City, ST 12345
United States

Note to seller: [Add](#)

Payment methods [Change](#)

Instant Transfer : Chase Manhattan Checking (Confirmed) x-2458 \$80.51 USD

PayPal will use MasterCard XXXX-XXXX-XXXX-4472 to fund this transaction if your bank does not have enough funds.

 PayPal gift card, certificate, reward, or other discount [Redeem](#)
View [PayPal policies](#) and your payment method rights.

Contact information

valued_customer@mail.com

Continue

You're almost done. You will confirm your payment on My Volusion Store.

[Cancel and return to My Volusion Store.](#)

[Site Feedback](#) (*)

PayPal. The safer, easier way to pay. For more information, read our [User Agreement](#) and [Privacy Policy](#).

after



[Sign Up](#) | [Log In](#) | [Help](#) | [Security Center](#)

MoneyMerchant ServicesAuction Tools

customers – and increase sales.

be updated automatically in late October.

the new ones. Log in, click the Merchant Services tab, I know you're a business that currently uses PayPal. This them coming back.

HTML changes required.

Old button	New button
	Buy Now
	Add to Cart
	Donate
	Subscribe
	Buy Gift Certificate

PersonalBusinessEmail addressforgot?Pa

PayPal™BuySellTransfer


Three little helpers...

- Price Matching
- Return Shipping on Us
- Give Now, Pay Later*

Finish your shopping with us


Learn About Holiday Exclusives

[See Terms & Conditions](#)
*Give Me Later is subject to credit approval.



Buy into being safer

No matter where you shop, we'll keep your financial information private and protected.



Sell in fewer steps

Gone are the days of waiting to get paid. Now you can request a secure payment in a few clicks.

change has started working its way out

Sandbox

[Home](#)[Test Accounts](#)[Test Email](#)[API Credentials](#)[Test Tools](#)

Additional resources

[Documentation](#)[PayPal Developer Network](#)[Customer Support](#)

PayPal Sandbox

API Credentials

You must have credentials to test API in the PayPal Sandbox. In most cases, you will use the test accounts identified below.

The test accounts identified below are for testing only.

Note: These credentials will not work on the live PayPal.com to go live.

Test Account

Test Account: test_131206115

API Username: test_131206115

API Password: 131206115

Signature: AFc

To download the certificate, log into the account associated with the account.

[Documentation](#)[Applications](#)[Dashboard](#)[Support](#)

// REST APIs

// Native SDKs

// Built for developers

[Get Started](#)

Looking for Sandbox? Import your test accounts to continue testing. Learn [what's new](#)

change has started working its way out

Mobile SDKs ^{BETA}

Easy, 100% native iOS SDKs that use our new REST APIs. Android coming soon.

[Download mobile SDKs](#)

REST APIs ^{BETA}

Great for simple payments using PayPal or credit cards. More REST APIs on the way.

[See our REST APIs](#)

LEAN ENGINEERING

Engineering for
Experimentation
with Lean Startup
Principles

**rethink engineering in the
light of lean**

**shift the lens of engineering to
embrace the build/measure/learn cycle**

engineer for experimentation

“bring design to life”

LEAN ENGINEERING

**6 principles for enabling
build/measure/learn**



1. engineer for learning, not delivery

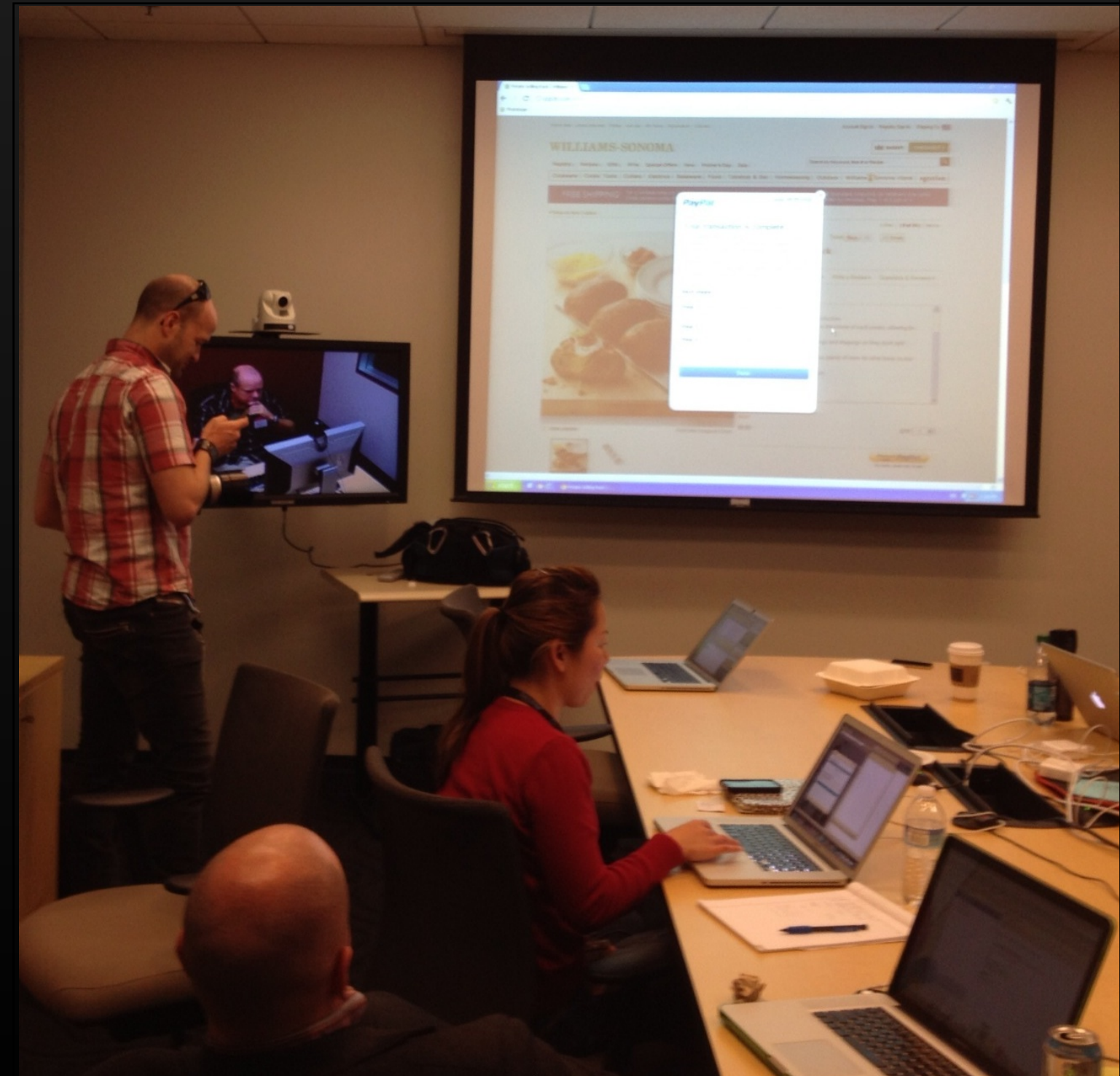
most organizations biggest challenge is moving
from a **culture of delivery** to a
culture of learning

include engineering in customer learning

engineers should regularly be in usability studies & customer visits

feedback from “measure” phase should be regularly discussed in engineering

you want to create an engineering culture that focuses on real customer problems



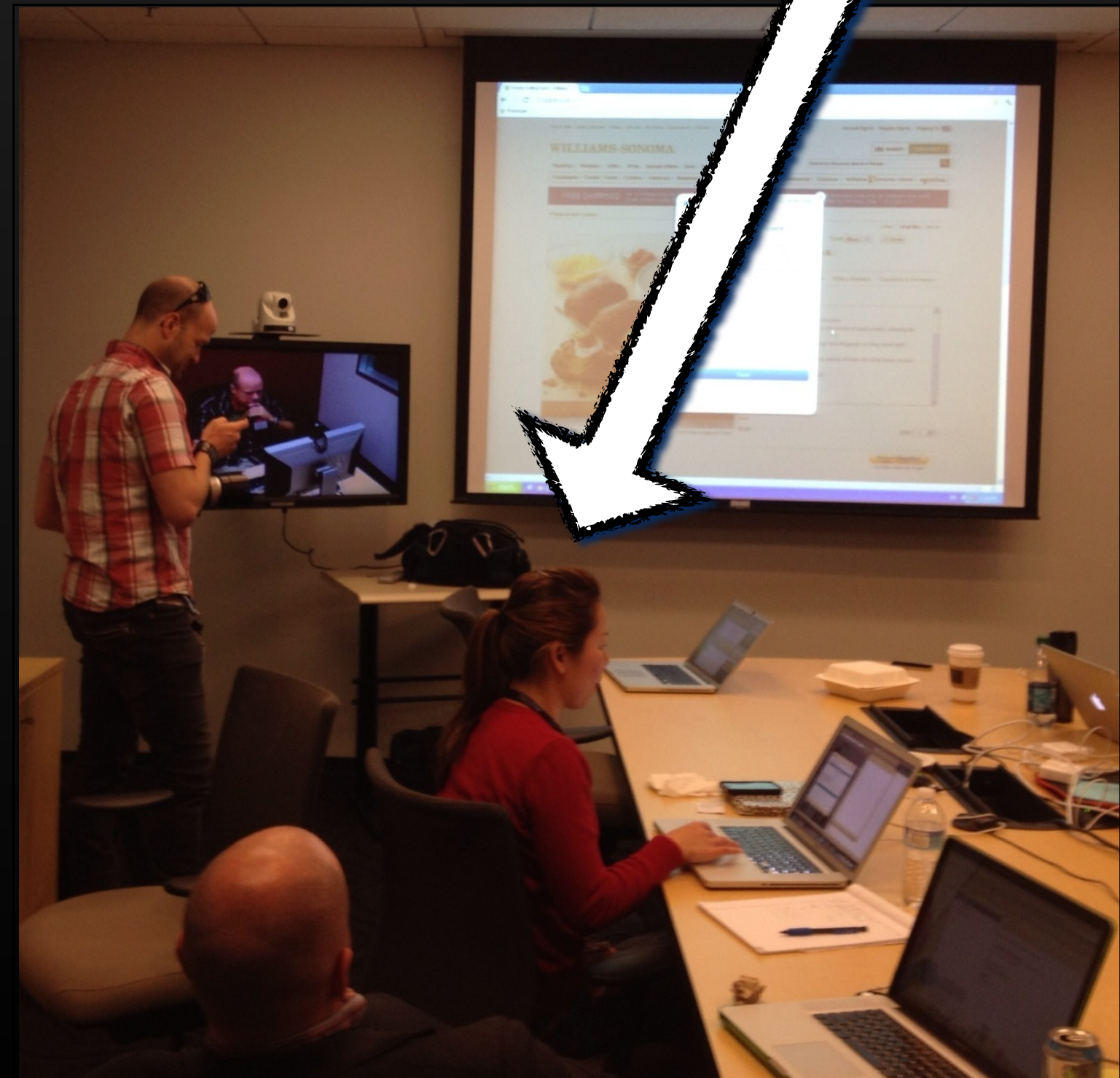
include engineering in customer learning

engineer

engineers should regularly be in usability studies & customer visits

feedback from “measure” phase should be regularly discussed in engineering

you want to create an engineering culture that focuses on real customer problems



enable prototyping in the engineering stack



because engineering teams are not trying to solve the learning problem, they see prototyping as outside the engineering discipline

make prototyping part of the engineering stack

engineer for the “living spec”

enables a single source of truth to iterate on as a team

stack circa 2011/early 2012

prototyping
was hard

“ui bits” could
only live here

restricted
capabilities*

client

server side
components**

server

jsp***

simple change could take minutes
to see

java

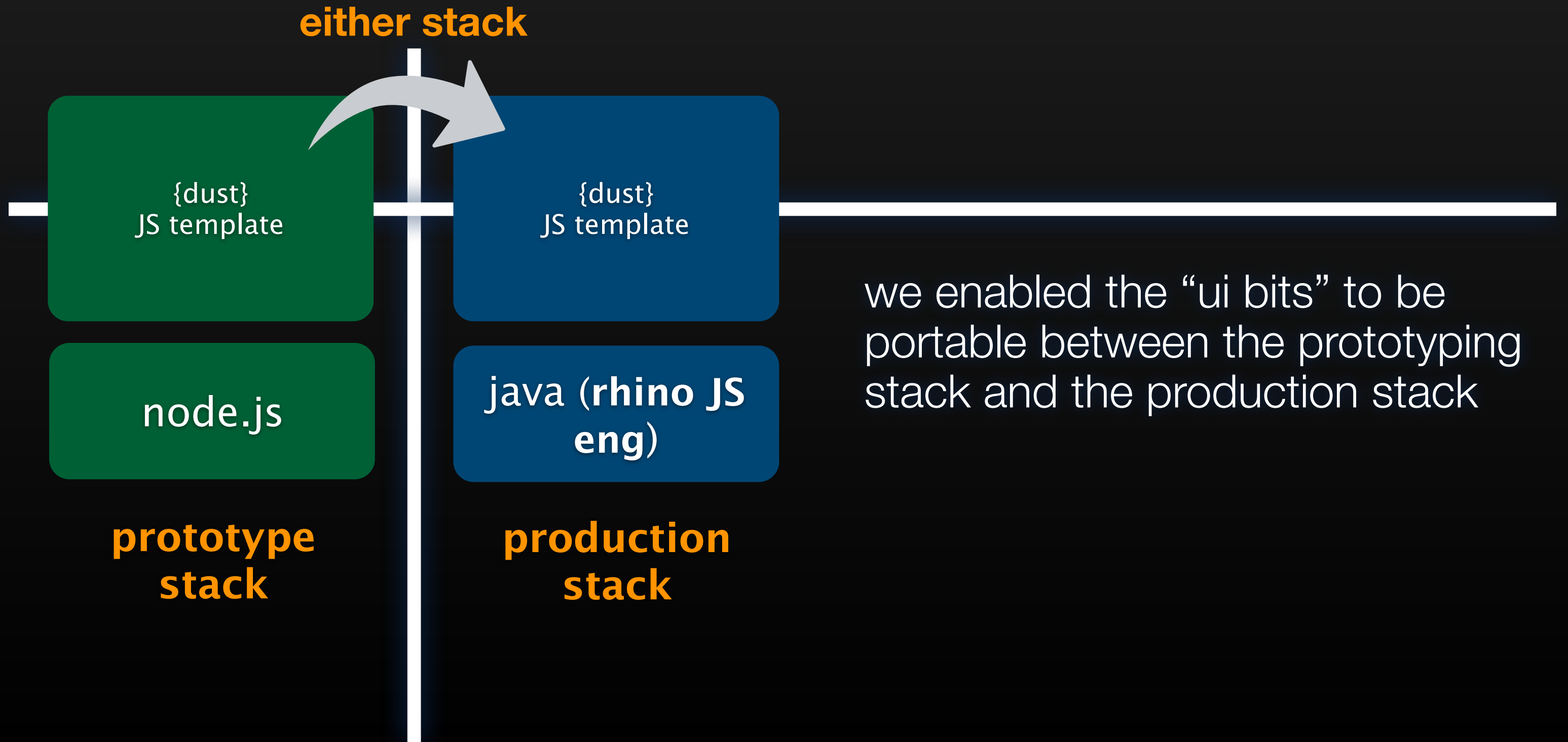
follows an “enterprise application”
model. ui gets built into the “app”

* assumed client developers were low-skill

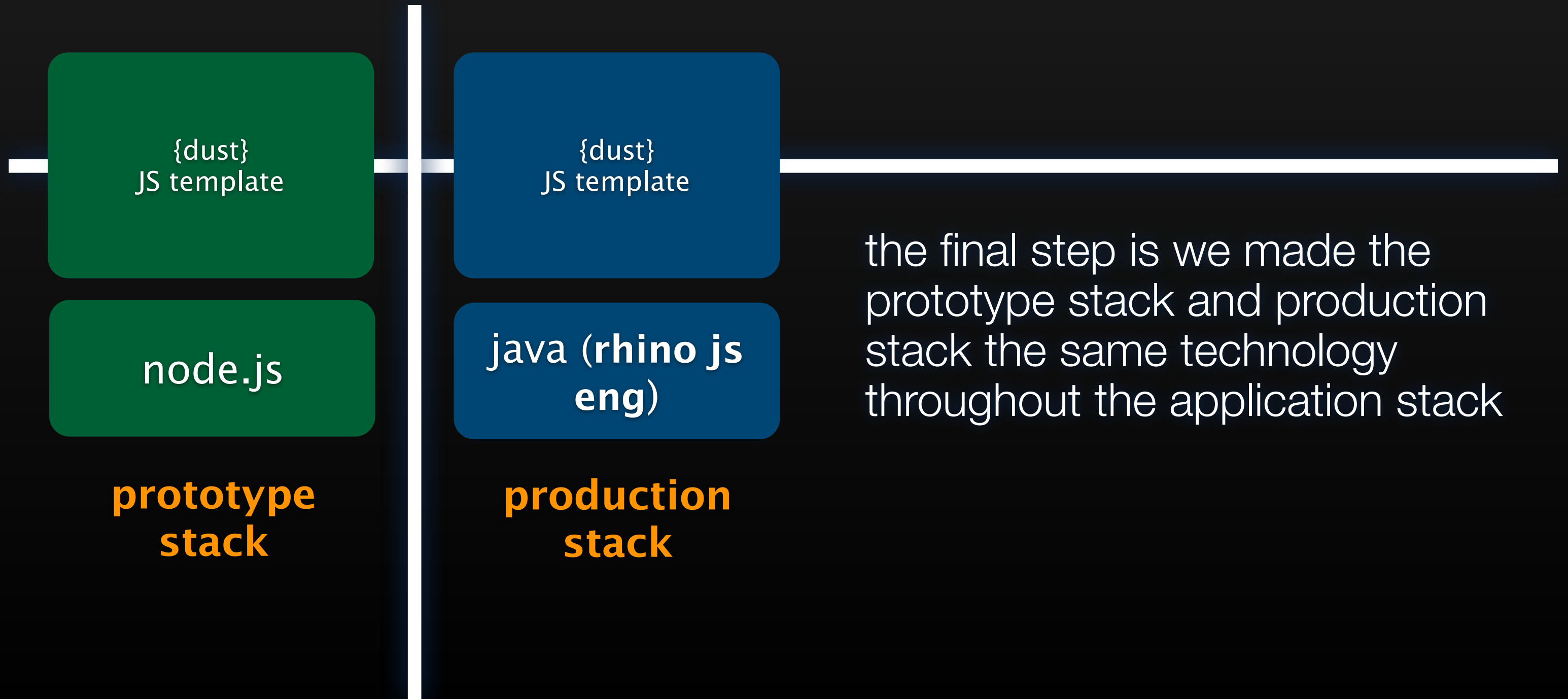
* required server side java eng for simple client changes

** java server pages. server-side java templating solution

we blended prototype & production



new single stack: prototype & production



new single stack: prototype & production



The diagram illustrates a 'new single stack' architecture. It features a vertical stack of two green rounded rectangles. The top rectangle is labeled '{dust} JS template' and the bottom one is labeled 'node.js'. A horizontal white line extends from the right side of the top rectangle across the slide. Below the stack, the text 'prototype stack' and 'production stack' are written in orange, indicating the two environments. To the right of the stack, a paragraph explains that the final step was making the prototype and production stacks use the same technology throughout the application stack.

{dust}
JS template

node.js

**prototype
stack**

**production
stack**

the final step is we made the
prototype stack and production
stack the same technology
throughout the application stack

teams must connect delivery to learning

in 1985 I delivered software on a 3.5" diskette

little or no feedback loop

everything was focused on getting it
the one right experience on the disk

no user in the loop. experience
happened somewhere down the
supply chain





2. engineer for experimentation

the netflix way

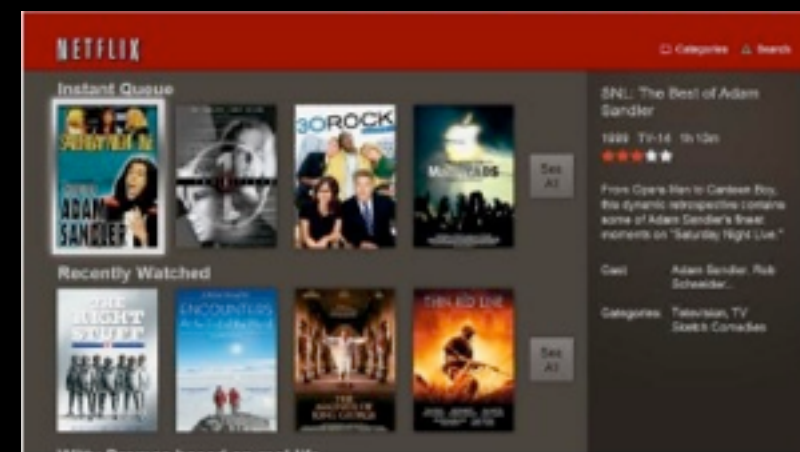
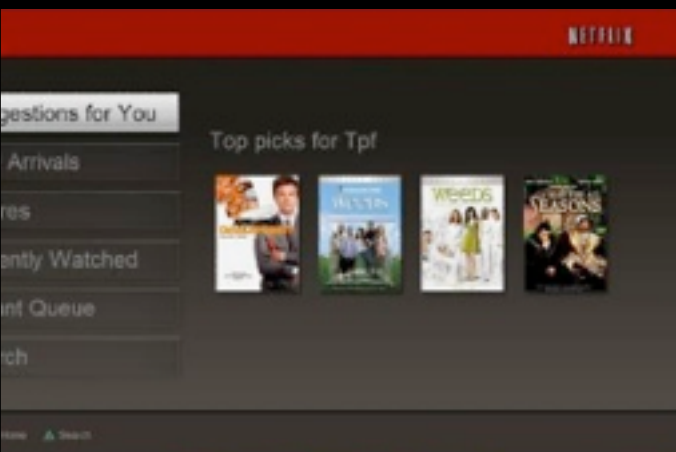
16 different test cells in the initial PS3 Launch (2010)

focus is on **experimentation**

the netflix way

16 different test cells in the initial PS3 Launch (2010)

focus is on **experimentation**



four distinct PS3 experiences launched on same day

the etsy way. Kellan Elliott-McCrea, CTO etsy

build

embrace
continuous delivery

make mistakes fast

measure

use metrics driven
development

know that you made a mistake

learn

blameless post mortems

learn from your mistakes

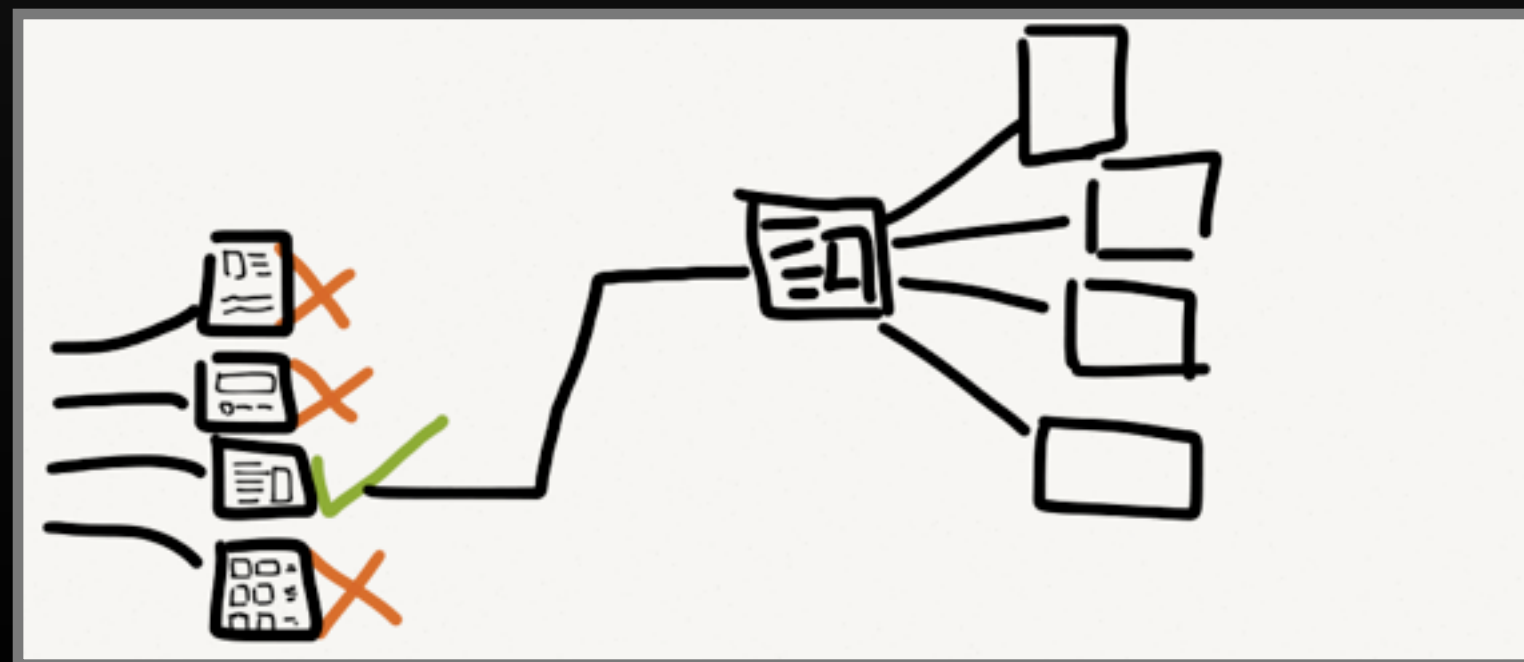


ramping vs experimenting



the big bet. **ramping** model results in one experience (with some tweaks along the way) after a long ramp up time

VS



lots of little bets. **experimentation** model results in many experiences being tested all along the way



long shelf life kills experimentation

engineering has to make delivery a non-event

result

delivery dates drive the experience

feature-itus becomes prevalent

BDUF & waterfall prevail

little to no learning

a tale of two trains



departs infrequently

“gotta get on the train or I will have to wait a long time”

a tale of two trains



departs infrequently

“gotta get on the train or I will have to wait a long time”



departs all the time

“if I miss this train another one comes in a few minutes”

using git for continuous deployment

starting to use git repo model for continuous deployment

- marketing pages

- product pages

- content updates & triggers into i18n, l10n, adaptation components

works well with cloud deployment (devops model)

enables the train to be leaving all the time



html5 is critical to learning strategy



netflix gambled on html5 for mobile (iOS, android) and for game consoles, bluray players, hdtvs, etc.

why? build/measure/learn. network delivery.



new users will see your html5 experience

the onramp to onboarding is the lowly link

network delivery makes a/b testing straightforward



3. refactor your way out of debt

technical debt

rarely do you have a clean slate

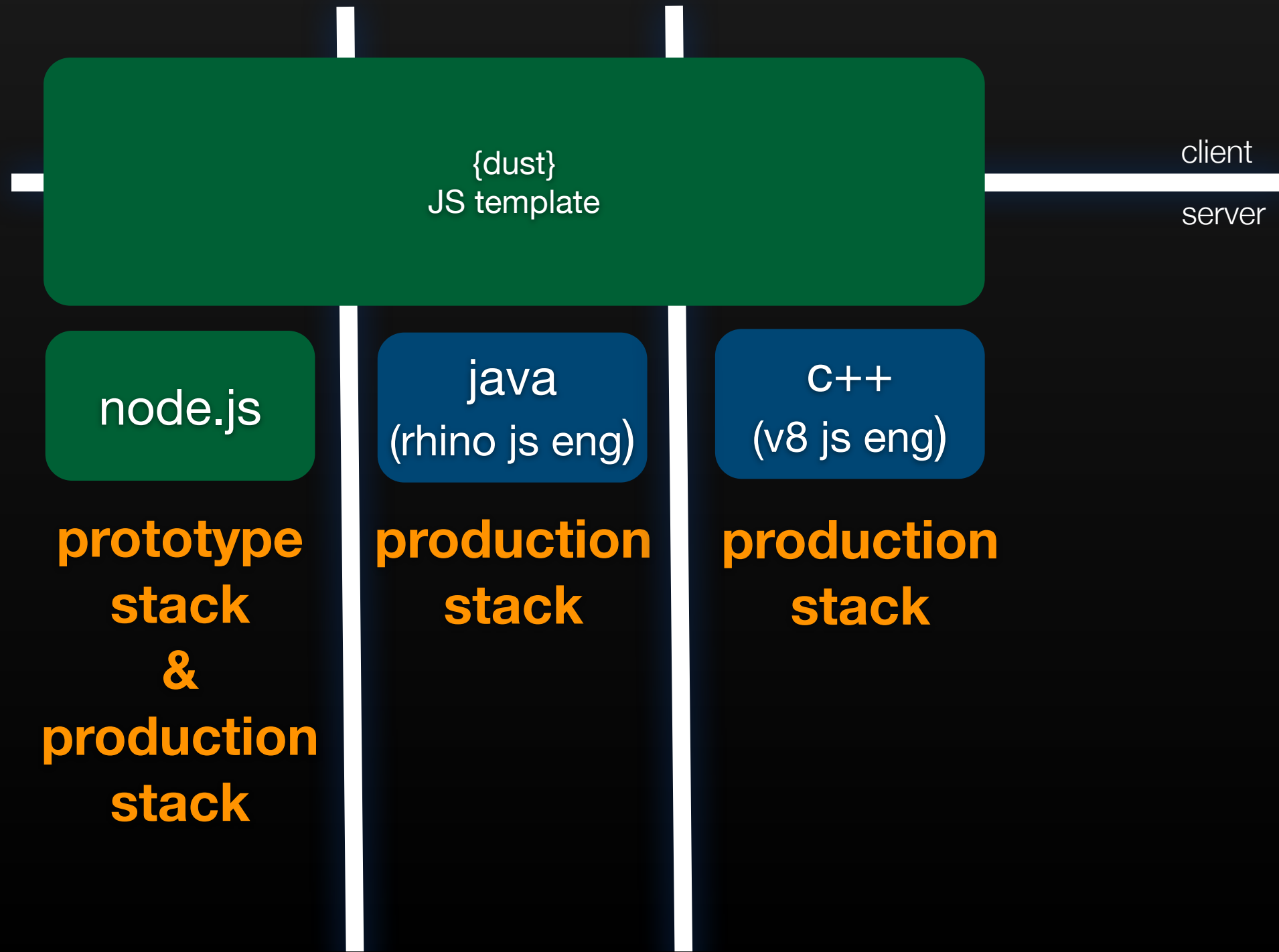
generally you will have to refactor your way to a nimble framework

you need to be clever to move from a legacy stack to a lean engineering stack



ensured we could run “ui bits” on new & legacy

any stack



we chose to make our “ui bits” as close to regular HTML/CSS as possible

we enabled the same “ui bits” (templates) to run on any of our stacks



ZZ
bad design kills

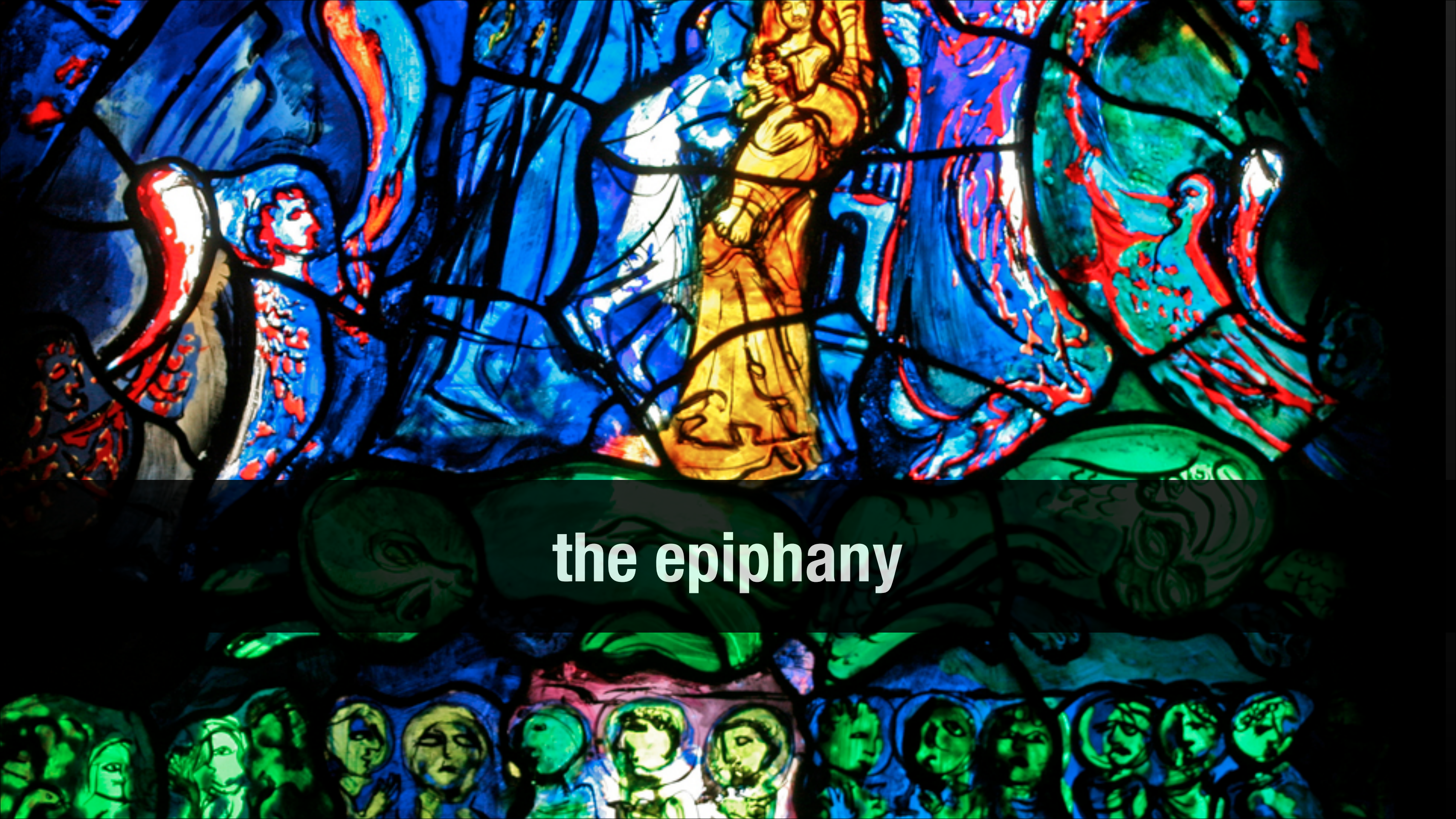
experience debt

don't just think about our technical debt
consider the "experience debt"
cripples our ability to capture market and
inhibits learning

key that engineering sees a chance to
improve the experience whenever they
are cleaning up technical debt



4. design for volatility



the epiphany

you have to engineer for volatility

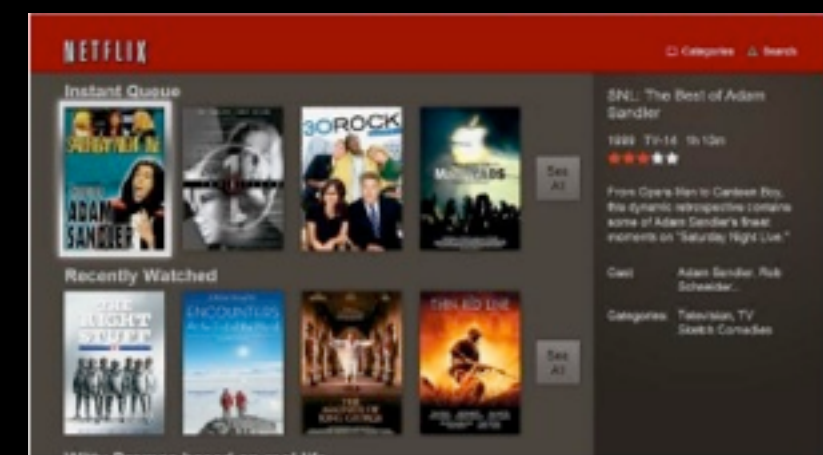
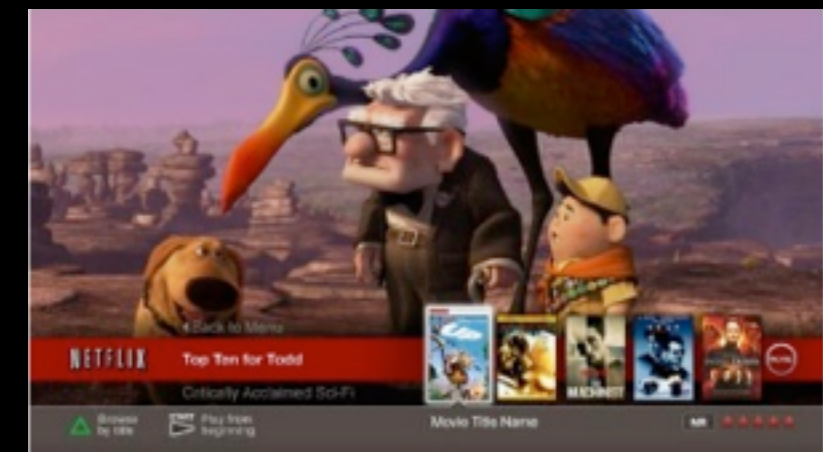
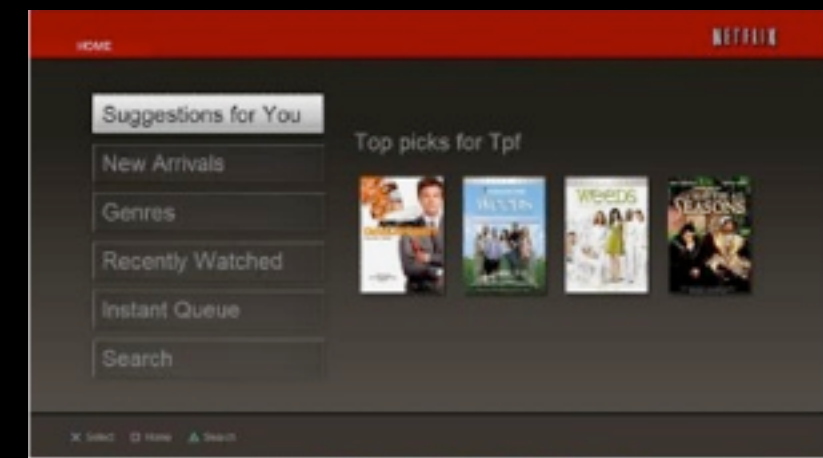
change is the norm

experimentation is not a one time event

launching a product is giving birth to the
product. the product's life just begins.

design for throwaway-ability

*majority of the
experience code
written is thrown
away in a year*



you have to engineer for volatility

change is the norm

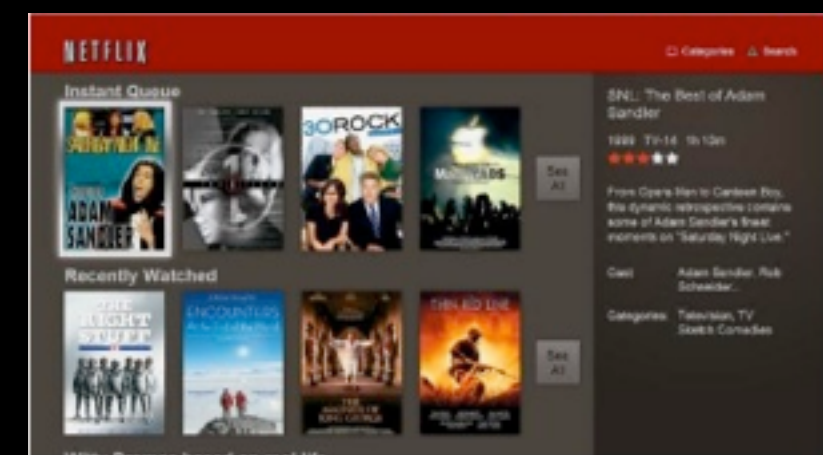
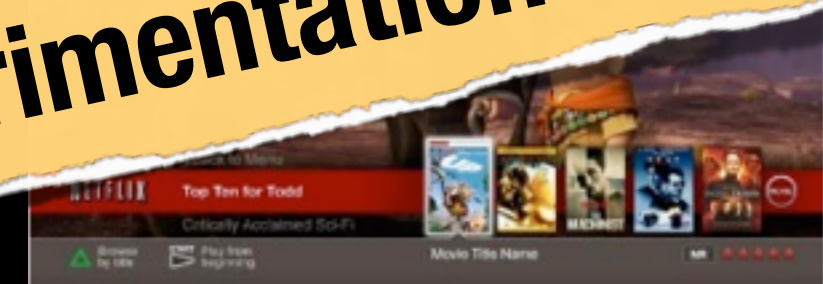
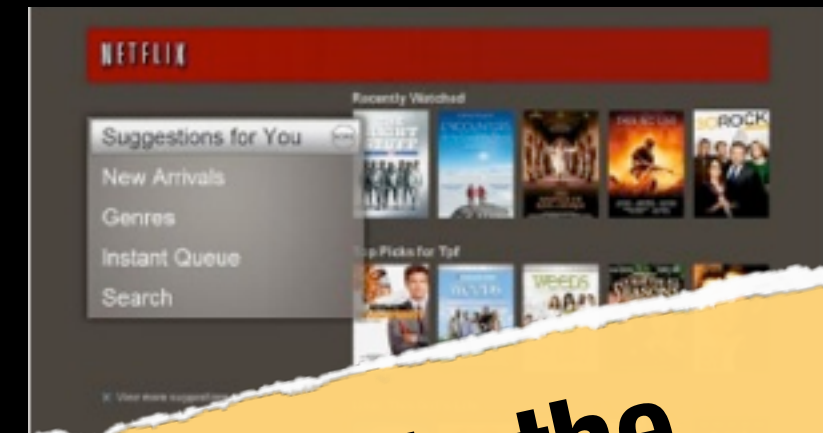
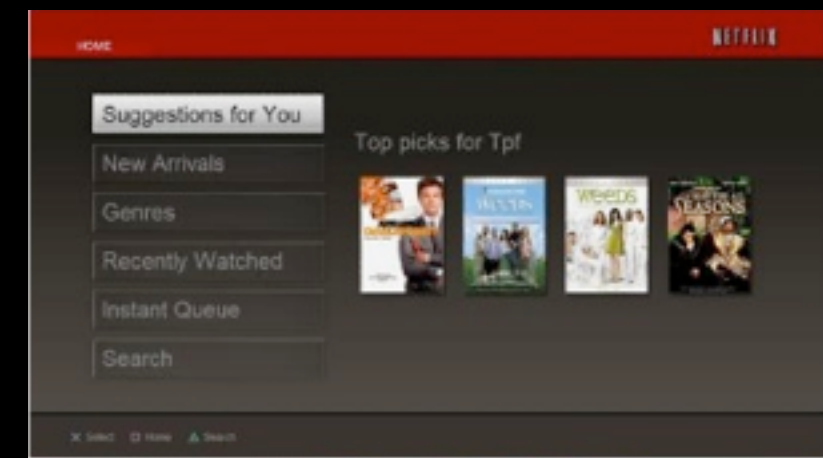
experimentation is not a one time event

launching a product is giving birth to the product. the product's life just begins.

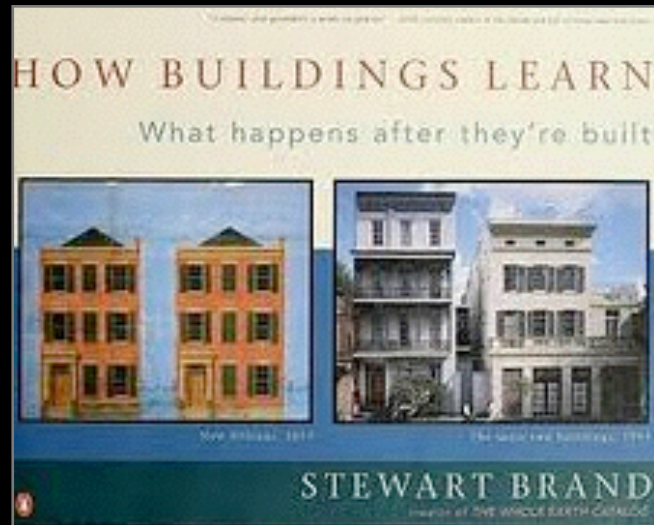
design for throwaway-ability

*majority of the
experience code
written is thrown
away in a year*

**the ui layer is the
experimentation layer**



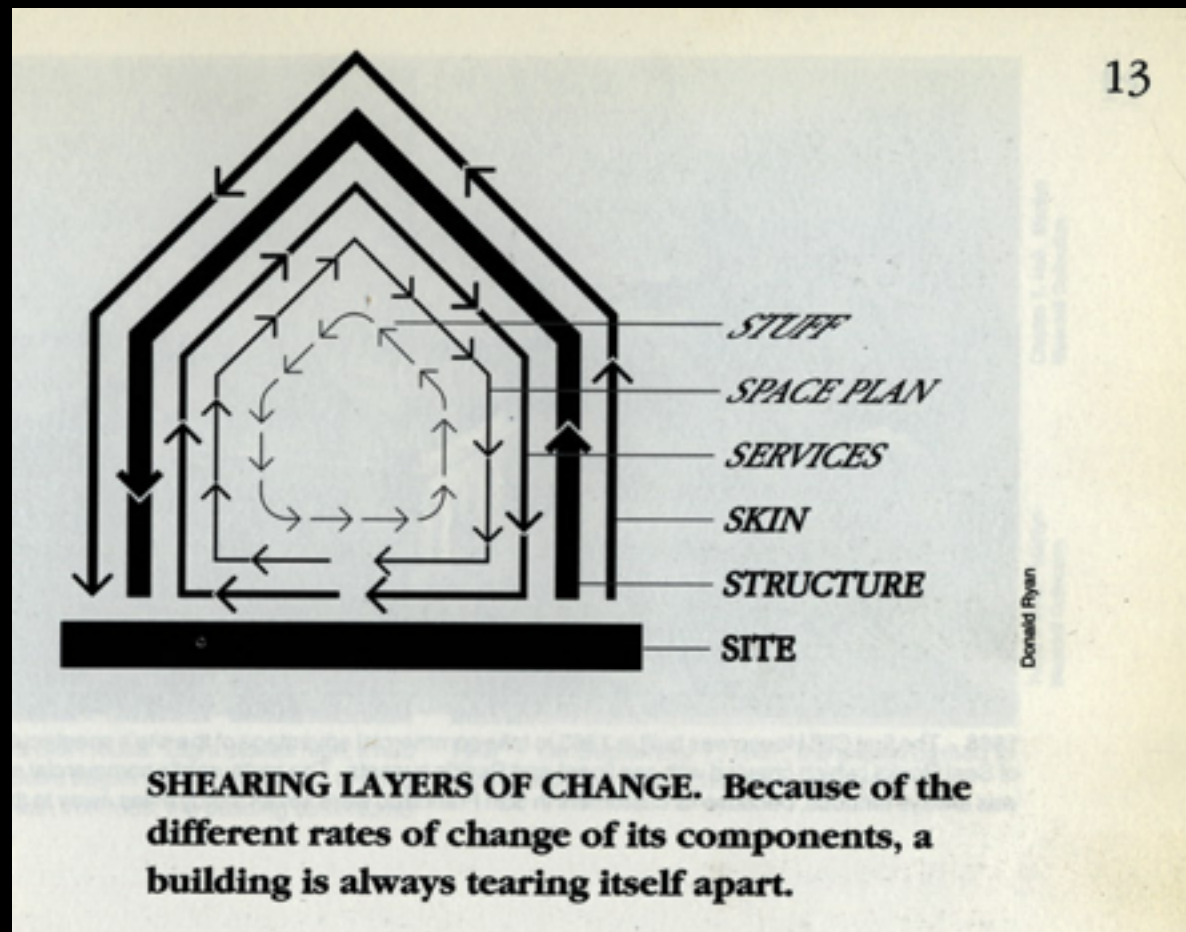
experiences must adapt



All buildings are predictions.
All predictions are wrong.

There's no escape from this grim
syllogism, but it can be softened.

Stewart Brand



Our software is always tearing itself apart
(or should be)

Recognize that different layers change at
different velocities

github
SOCIAL CODING

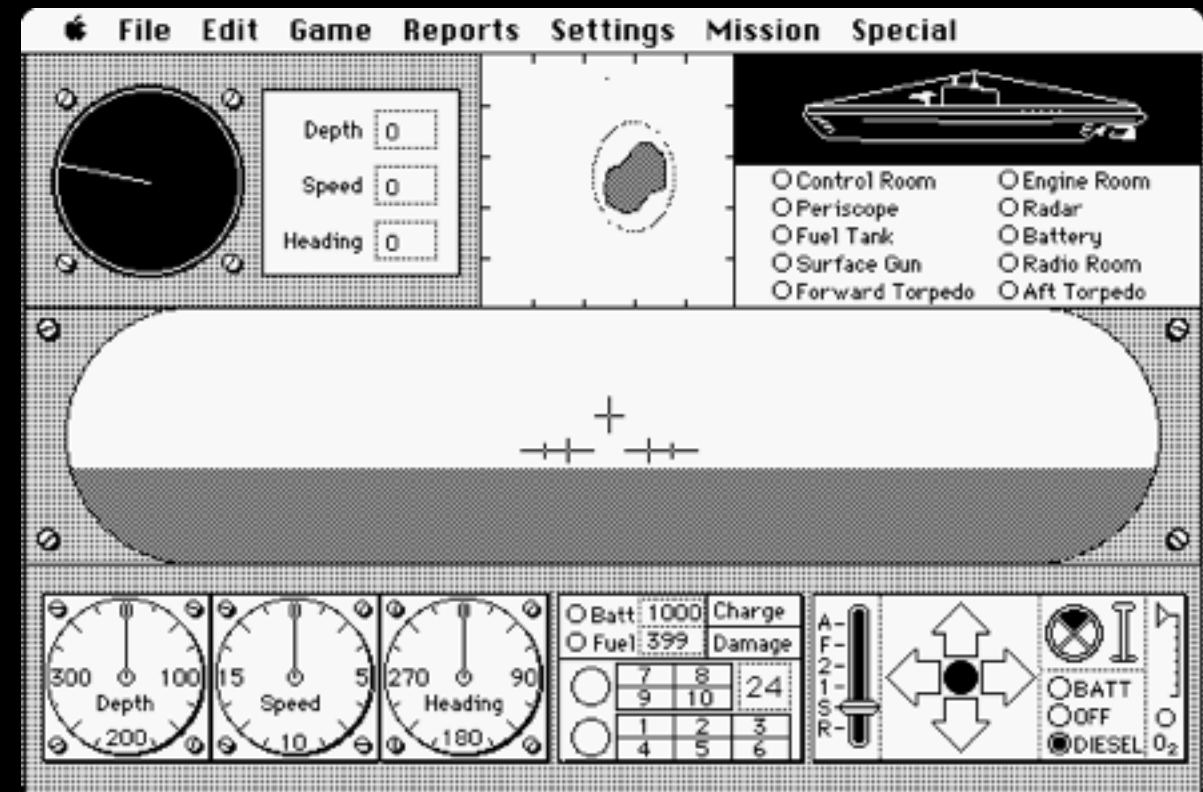
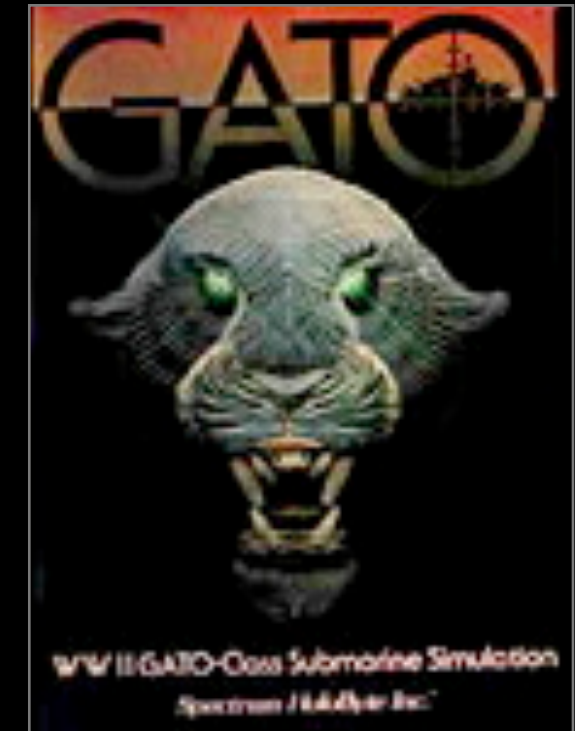


5. decentralize engineering

building experiences circa 1985

merry band of three. dropped out of
college for semester. it was nirvana.

however...





roll your own “everything”

(close your eyes & imagine)

no internet. no google. no blogs. no email. no
blogs. no stackoverflow. no github. no twitter.

much of the software era has been about
building from scratch.

of course open source was gaining momentum.
unix. gnu. linux. perl. mozilla.

work in open source model

internal github revolutionizing
our internal development

rapidly replacing centralized
platform teams

innovation democratized

every developer encouraged
to experiment and generate repos
to share as well as to fork/pull request



give back to open source

we have projects that we will open source

- node webcore (similar to yeoman)

we are contributing back to open source

- contributions to bootstrap (for accessibility)

- contributions to bootstrap (for internationalization)

- core committer on dustjs project

use open source religiously

Bootstrap, from Twitter

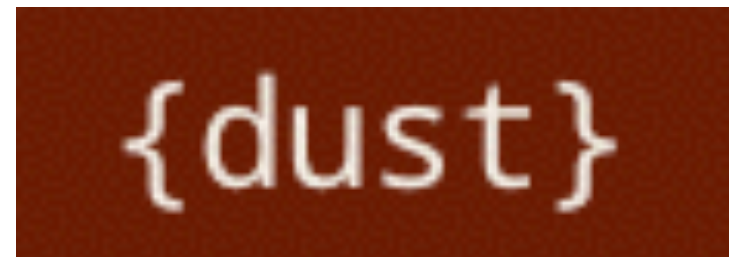


BACKBONE.JS

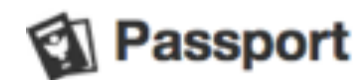
UNDERSCORE.JS

express

q



async

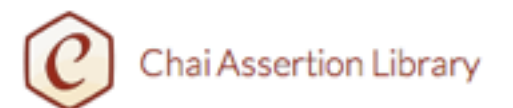


nconf



BOWER

supertest





Lean



Agile

Use LEAN
to give
AGILE a
BRAIN
AGILE is a MACHINE

6. put a brain on agile

agile doesn't have a brain...

agile is a hungry machine. it will crank out garbage or brilliance. and it will do it iteratively.

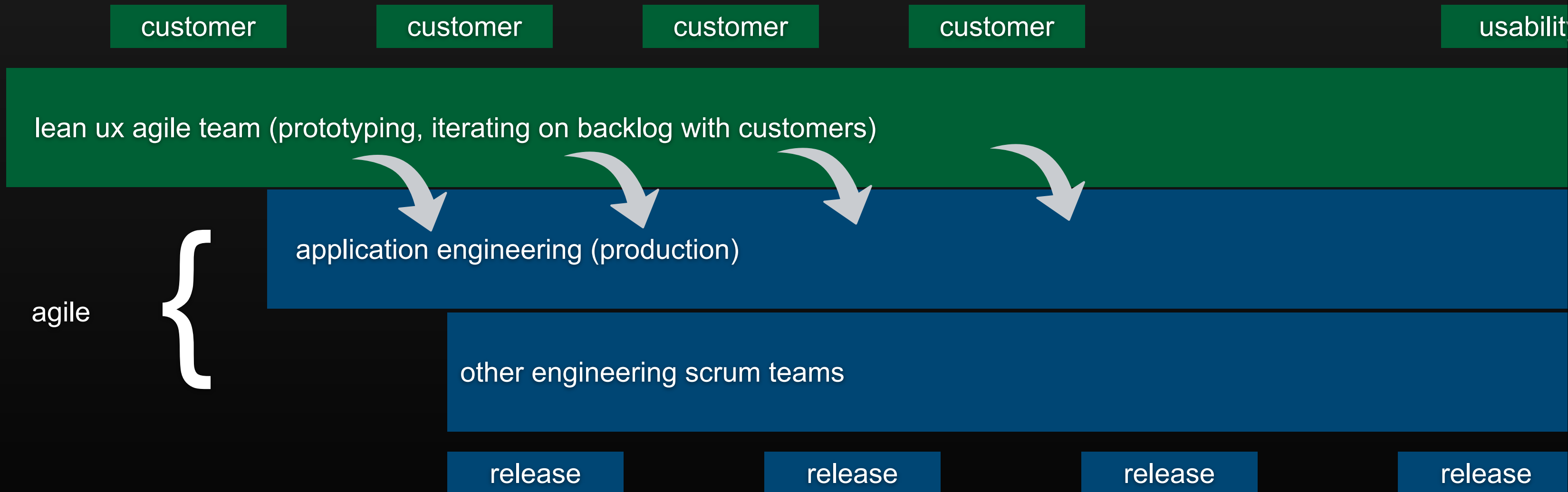
it is a travesty to waste this machine

you have to get the experience “in the ballpark” to best use the machine

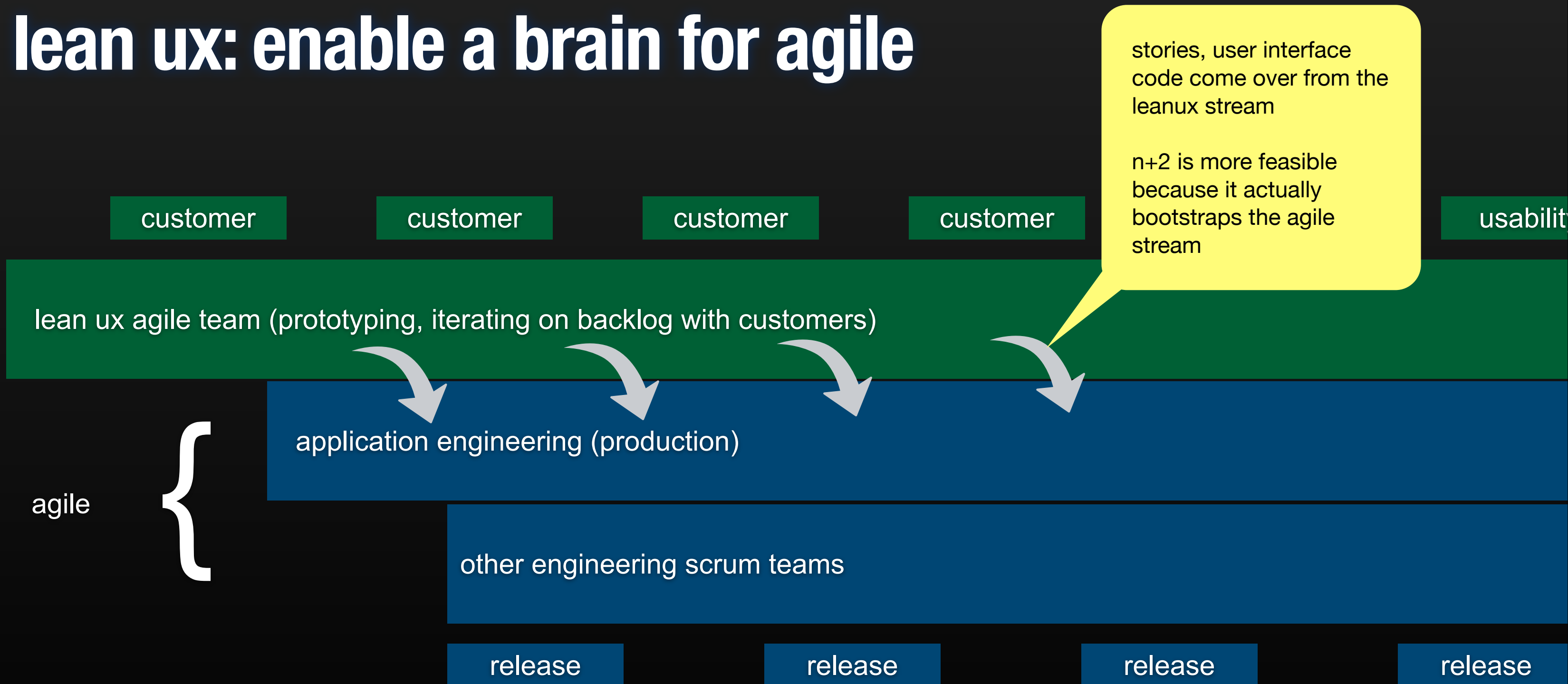
it is imperative to make it easy to iterate designs ahead of the agile sprints

leanux in the form of a “leanux scrum team” is one way to do this

lean ux: enable a brain for agile



lean ux: enable a brain for agile



center yourself around a “customer day”

* DO NOT ERASE *				
MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
<u>(all day)</u> - Design - Content - Iterative Coding	<u>(all day)</u> Design Content Iterative Coding Hand-off	Feedback + Iterative Coding ----- 4pm Usability Delivery	9am Usability Session	1pm - 2:30pm Usability Review ----- 3pm What to build next week
CO-located in ASTUTE				
INVOLVED	UED UIG PO	* DO NOT ERASE *		

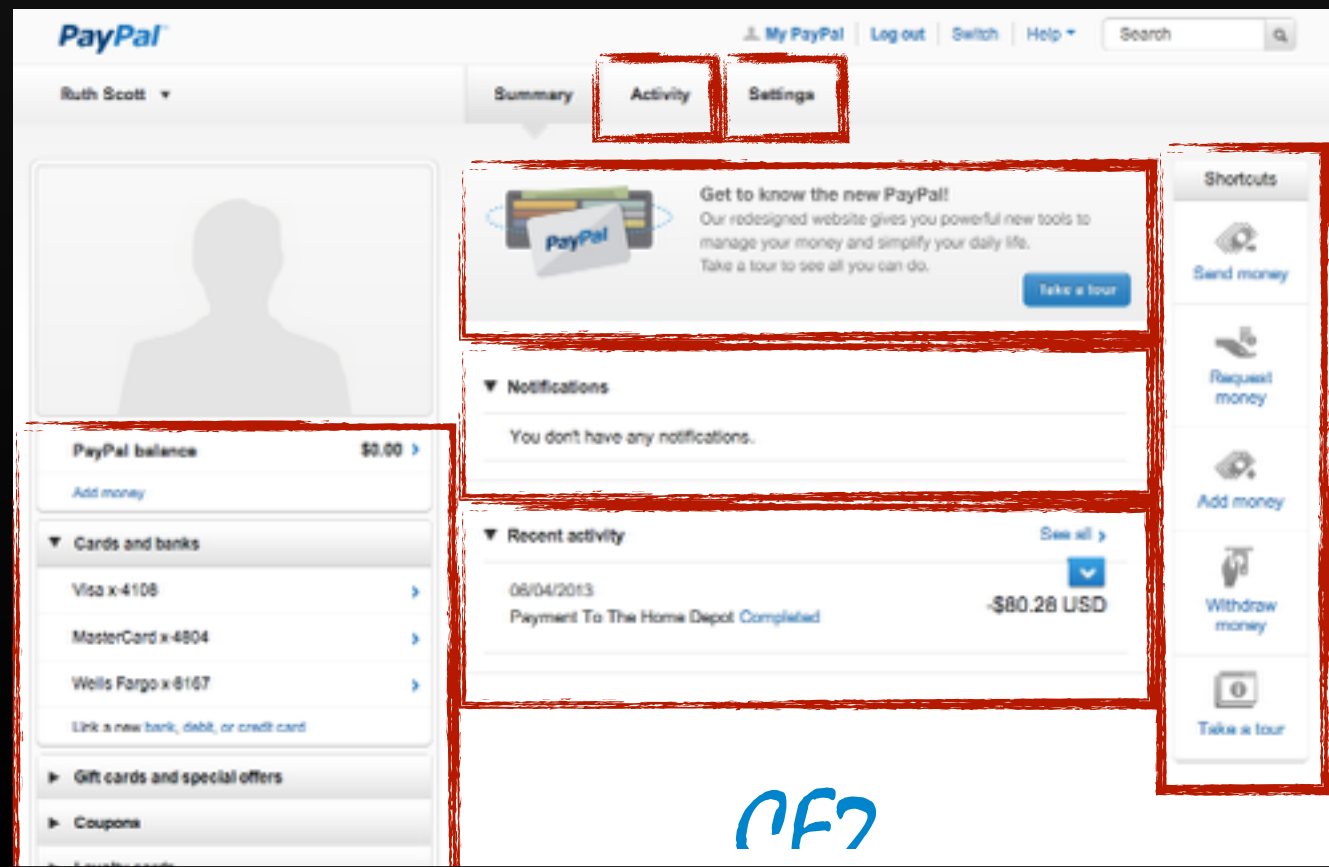
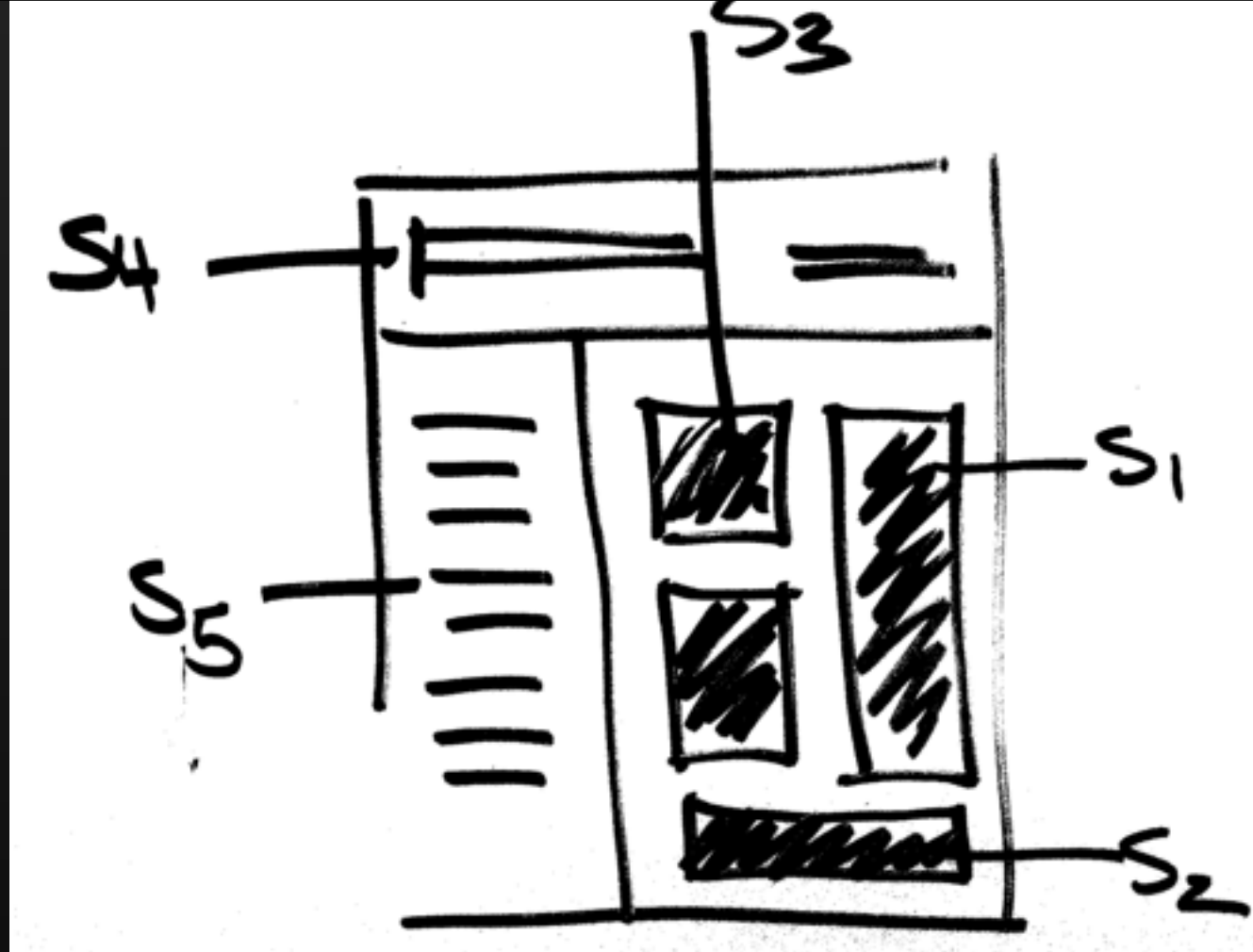
customer day

watch out for fracturing the experience

too many teams can create silos within the experience

what can fracture the experience

- number of scrum teams
- specialization of skills
- device channels
- regional adaptations



watch out for mismatch between teams

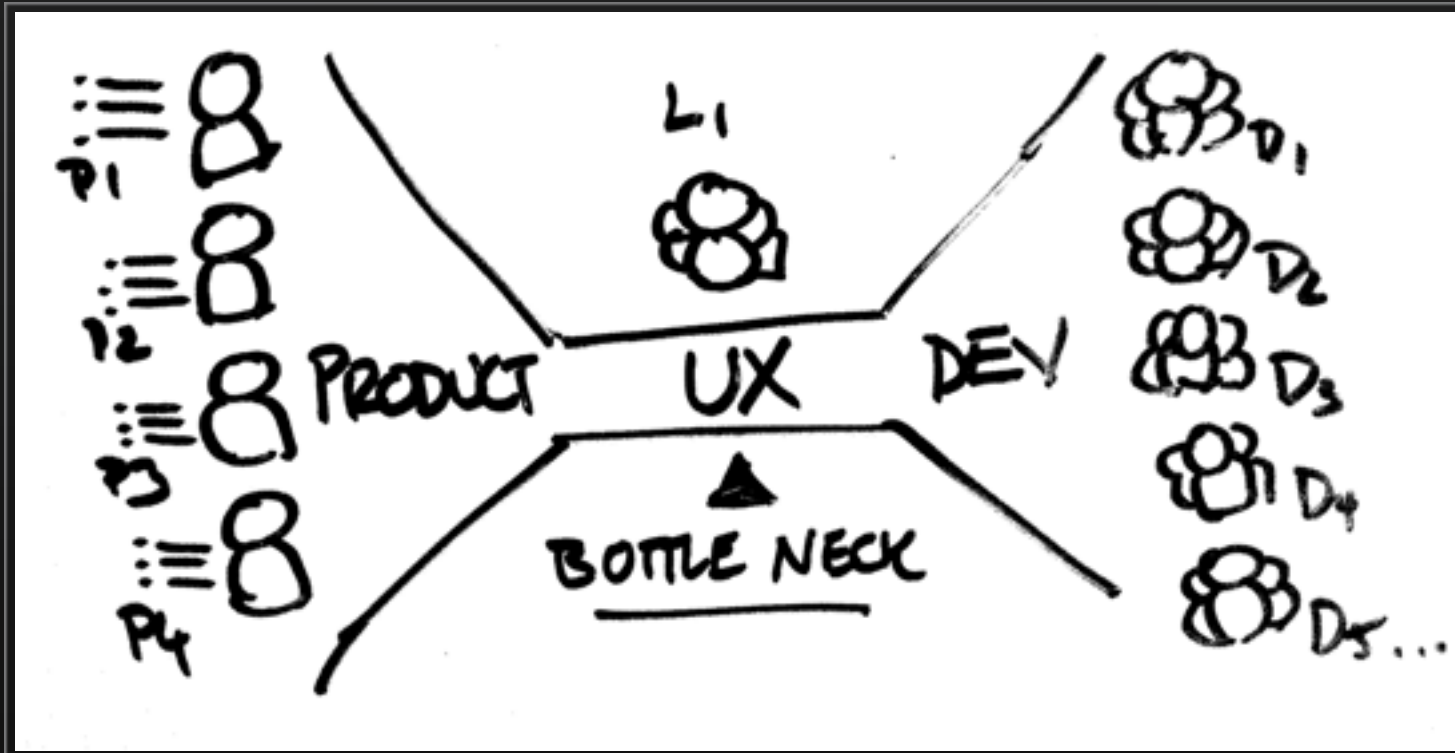
as we have mapped lean onto agile it has exposed mismatches between the way we work

you run the risk:

- driving the experience based on eng teams
- becoming a bottleneck

in almost all cases

- either product is not aligned to the biz
- or there are just too many people



specializations happen at scale

specialization of skills can be orthogonal to execution

but execution is dependent on the skills that come from specialization



our next step is refactoring our specializations

start thinking of “engineering as design” and “designers as engineers”

blend the UI engineering & backend app engineering into one engineering team

blend QA org into engineering team as test engineers

continue to break down walls between design & engineering

refine lean ux “scrum” teams to seamlessly blend with the agile scrum team

LEAN ENGINEERING

Engineering for
Experimentation
with Lean Startup
Principles

rethink engineering in the
light of lean

shift the lens of engineering to
embrace the build/measure/learn cycle

engineer for experimentation

“bring design to life”



picture credits

http://www.flickr.com/photos/decade_null/2053134780/

http://www.flickr.com/photos/not_wise/182849352/

<http://www.flickr.com/photos/37217398@N02/3442676067/>

<http://www.flickr.com/photos/hongiiv/4151964823/>

Photo by Kim White: <http://readwrite.com/2013/09/05/paypal-app-update-in-store-payments#awesm=~ohHUpP9dhMmMG>

<http://www.flickr.com/photos/matthewpaulson/6176787688/>

<http://www.flickr.com/photos/olvrbrown/4542851399/>

<http://www.flickr.com/photos/juanpol/16287486/>

<http://www.flickr.com/photos/olvrbrown/4542851399/>

<http://www.flickr.com/photos/mbiskoping/6075387388/>

<http://www.flickr.com/photos/giesenbauer/4092794246/>

<http://www.flickr.com/photos/kowani/5565778790/>

<http://www.flickr.com/photos/ahockley/2657296577/>

<http://www.flickr.com/photos/90585146@N08/8222922317/>

<http://www.flickr.com/photos/therevsteve/3104267109/>

Stewart Brand: How Buildings Learn (illustrations)

http://www.flickr.com/photos/light_seeker/7444052000/

Krystal Higgins:

<http://www.kryshiggins.com/sketchnotes-of-bringing-design-to-life-with-lean-ux-lean-engineering/>

<http://www.flickr.com/photos/epsos/8463683689/>

<http://www.flickr.com/photos/proimos/3473264448/>

<http://www.flickr.com/photos/stuckincustoms/2380543038/>

follow me on twitter
@billwscott