real world lessons moving to lean ux

lessons learned @ paypal

bill scott (@billwscott)
sr. director, user interface engineering, paypal

webvisions, chicago, IL, sept 2013

the purpose

help you identify the common **problems** encountered

put before you how we **adopted** lean in our environment

talk about specific **lessons** we learned along the way

discuss some practical **tools/techniques** in adopting lean

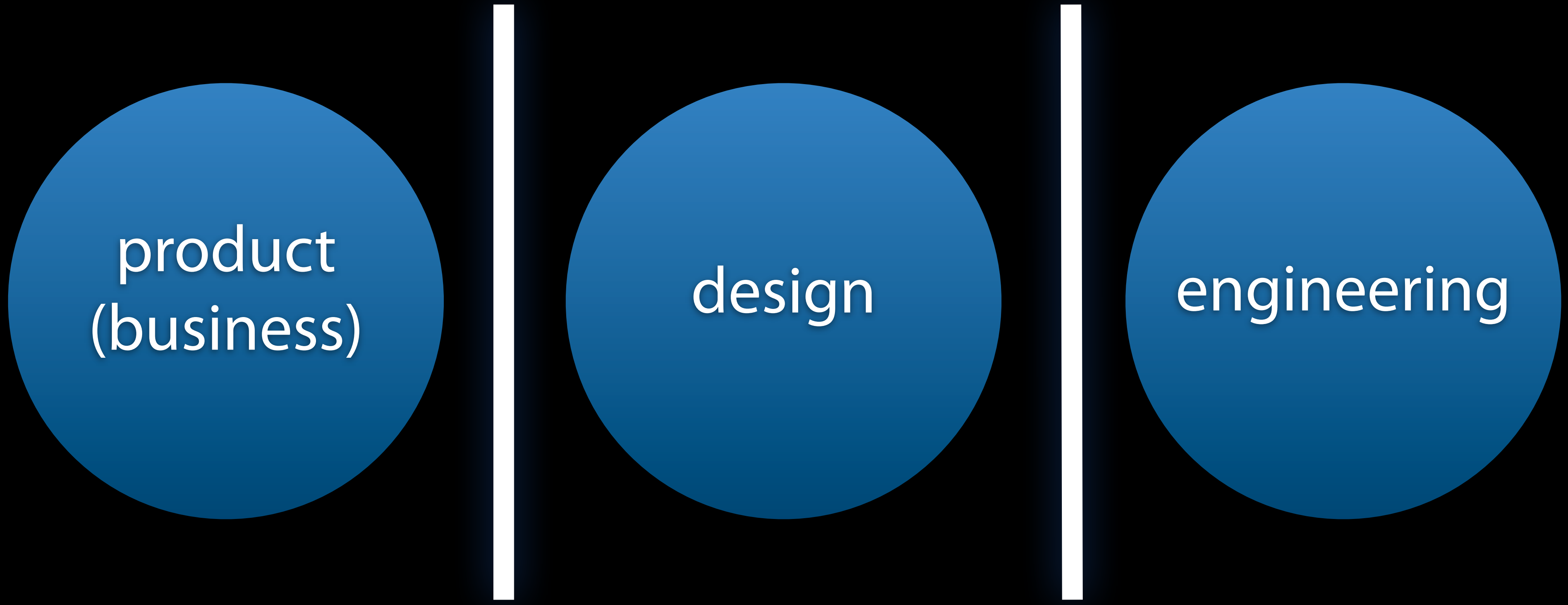
the schedule

2:00–2:15	introductions (15)
2:15–2:45	the **problems** (30)
2:45–3:15	**adopting** lean (30)
3:15–3:45	break (15)
3:45–4:45	the **lessons** (60)
4:45–5:15	the **tools** (30)
5:15–5:30	discussion and Q&A (15)

the problems

a look at where paypal has been. can you relate?

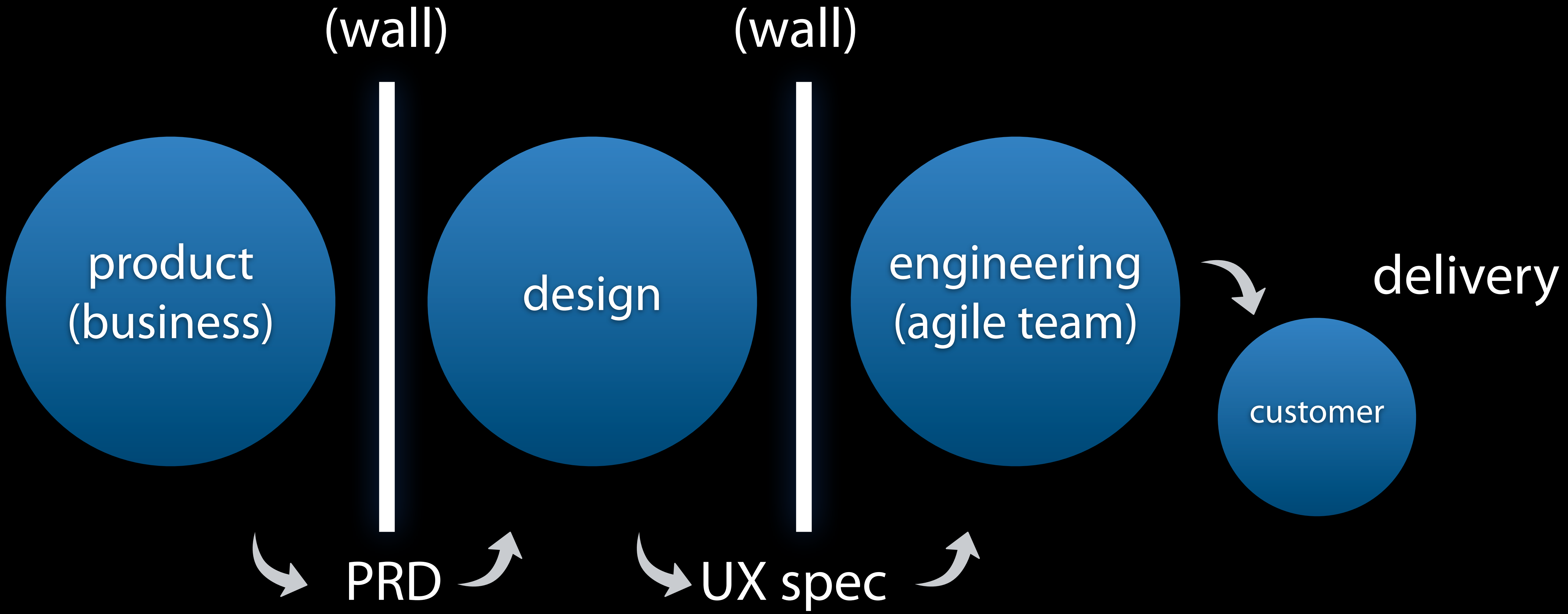
organizational model



standard process creates distinct work phases

boundaries are the hand-off points between roles

typical product life cycle



upon delivery, team disbands and forms into new teams

what was broken in design?

late 2011/early 2012

deep silos

iteration planning done by developers without designer’s involved

designers hand off specs without prior involvement of developers

developer days (“dev days”) valued over design time

frequent WFH days created low energy and less collaboration time

hyper-segmentation of products

broad team distribution

geographic distribution created wedges, duplications and blocked collaboration

lack of alignment with UED partners (not uncommon to have designers & engineers in same region to be working on different products)

lack of agile understanding

while UED interfaced with agile teams they did not participate directly in agile planning, retrospectives, etc.

agile machinery also did not account for experience design

no strong ownership

UED staff in a pooled/studio model instead of a dedicated model

once delivery happened the designers moved to another project

often engineers did not know who the designer was for a product to ask questions to

teams not empowered to make decisions as a gauntlet of other teams had to approve to go live

what was broken in product?

late 2011/early 2012

no measurement/learn culture

in several products there were no key performance indicators to measure & learn against

since a/b testing was hard to do, there was no concept of an MVP (minimal viable product)

feature-itus

since the organization rallied around projects instead of products, product tended to try to put as much on the train as possible

without kpis you guess more and more (F.O.G.)

without measurement you never get rid of features

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”

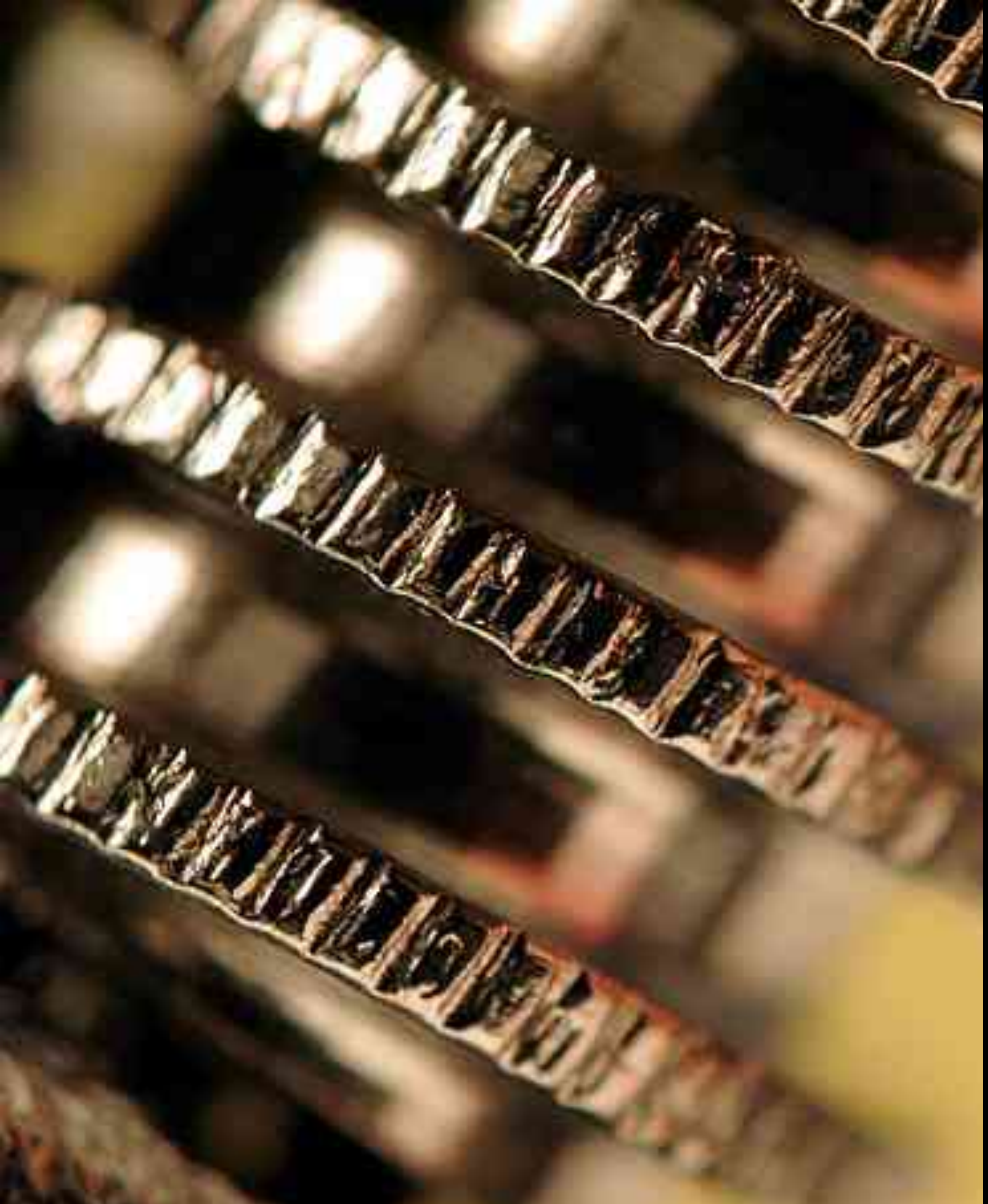
departs all the time

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

too many silos

product was divided over 9 different organizations!

mobile was also a separate business, product and engineering silo



what was broken in engineering?

late 2011/early 2012

too many silos

just like our counterparts, we were broken into many different organizations

mobile was a separate organization

too hard to go live

37 tickets just to get a bug fixed and pushed to live

every organization was set up to say “no” to anything that might be innovative for fear of failure, risk, security issues, etc.

no devops, no CI/CD

technology broken

no modern services architecture

all solutions were built as silos

ui logic and business logic intertwined

technology platform assumed developers were not to be trusted

agile way too granular

one product had 20+ agile streams. 12 of these were experience streams. each stream was responsible for one small part of the experience

created nightmares of integration

created a fractured experience for users

paypal circa 2011

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

Sign Up | Log In | Help | Safety Advice Search

PayPal

Home | Individuals | Business | Partners

Get started | How it works | Buying safely | Selling safely | Donate to Charity

GET THE MOST OUT OF PayPal

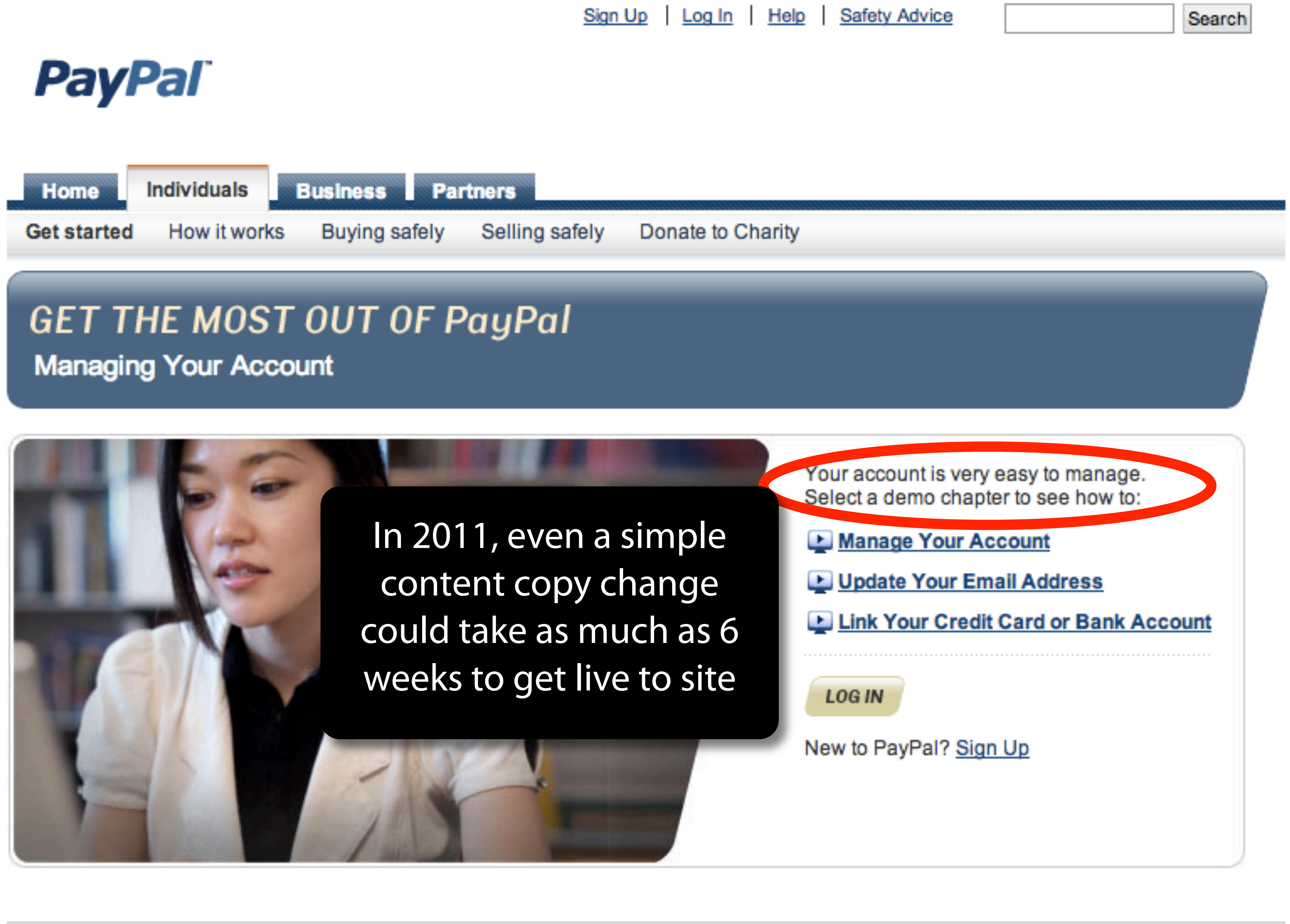
Managing Your Account

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


Sign Up | Log In | Help | Safety Advice

Search

PayPal

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

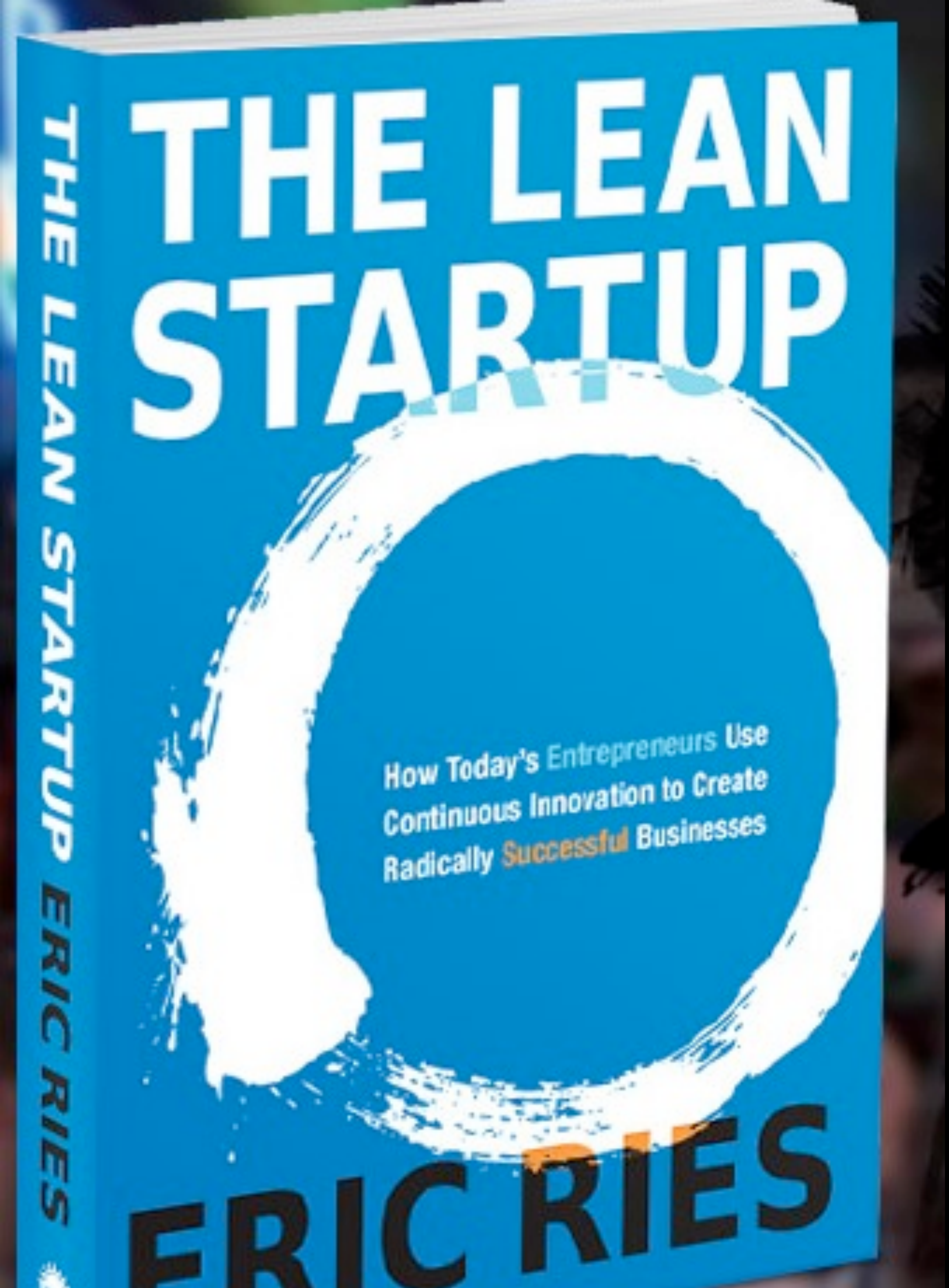
LOG IN

New to PayPal? Sign Up

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

adopting lean

following a build/measure/learn mindset


THE LEAN STARTUP
How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses
ERIC RIES
THE LEAN STARTUP ERIC RIES

lean startup

founded on build/measure/learn

get out of the building (GOOB)

invalidate your risky assumptions

go for the minimal viable product (MVP)

fail fast, learn fast

get to the pivot

lean ux

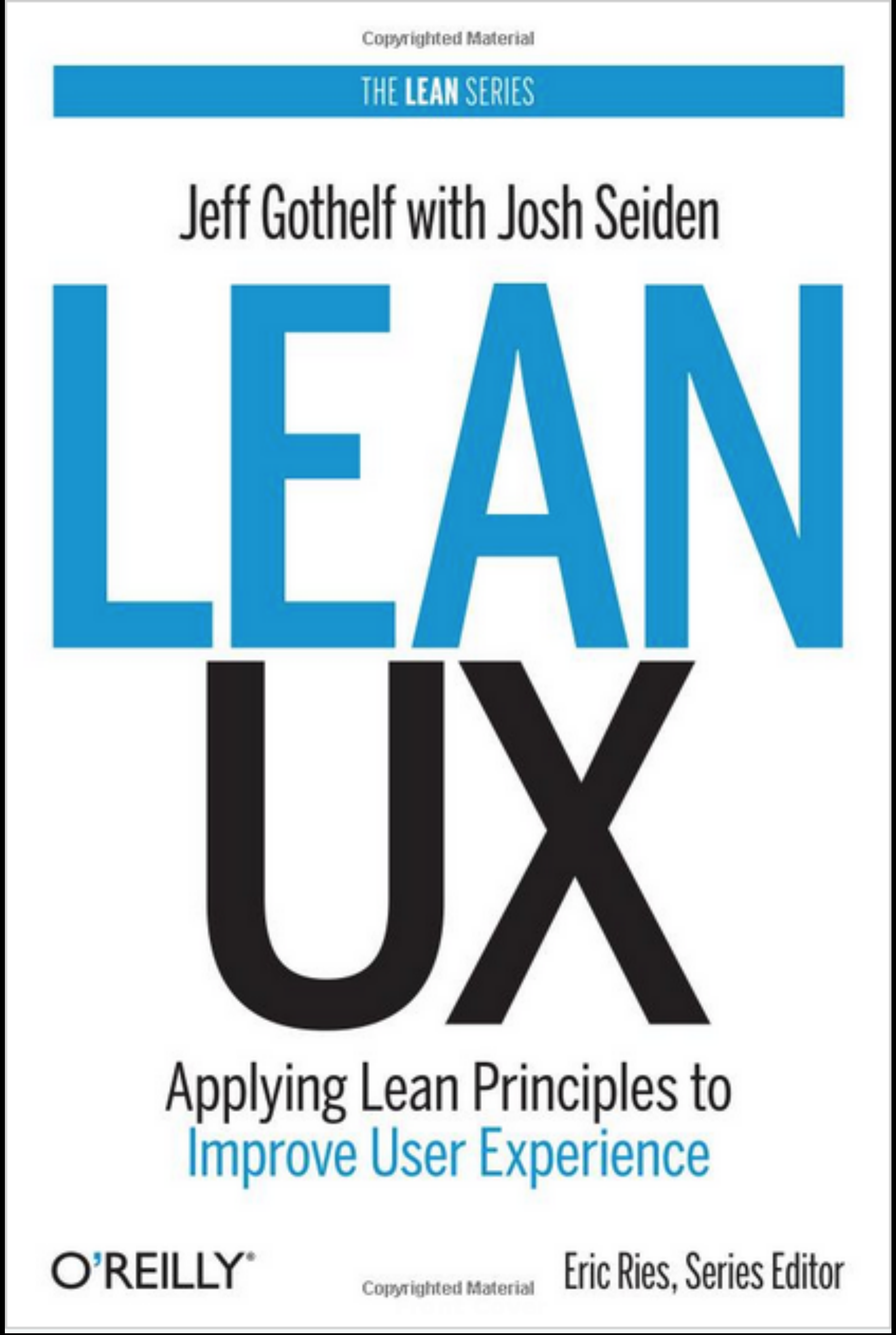
designing products for build/measure/learn **(lean startup)**

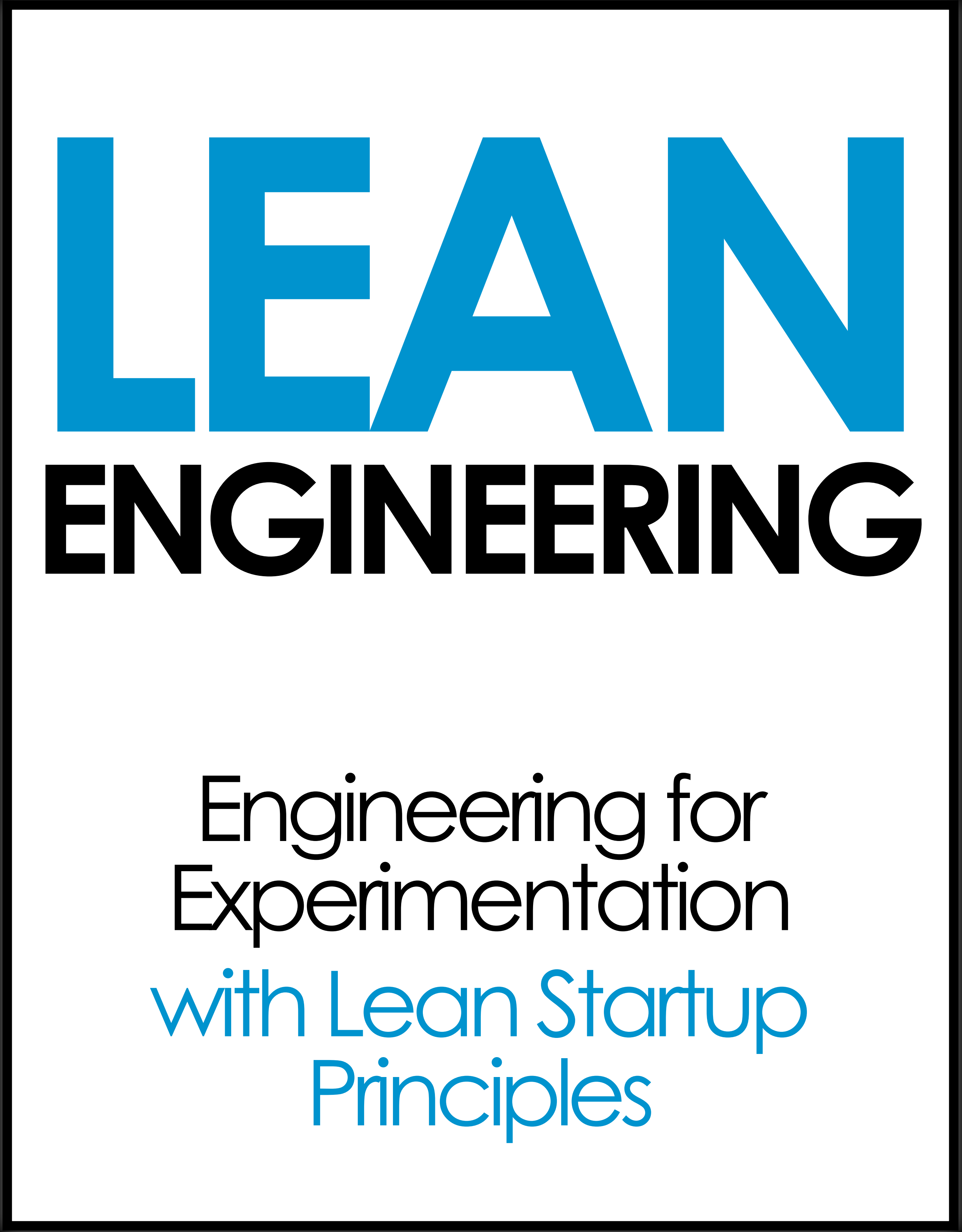
requires 3 rules to be followed at all times

get to & maintain a **shared understanding**

form **deep collaboration** across disciplines

keep **continuous customer feedback** flowing




LEAN ENGINEERING

Engineering for Experimentation with Lean Startup Principles

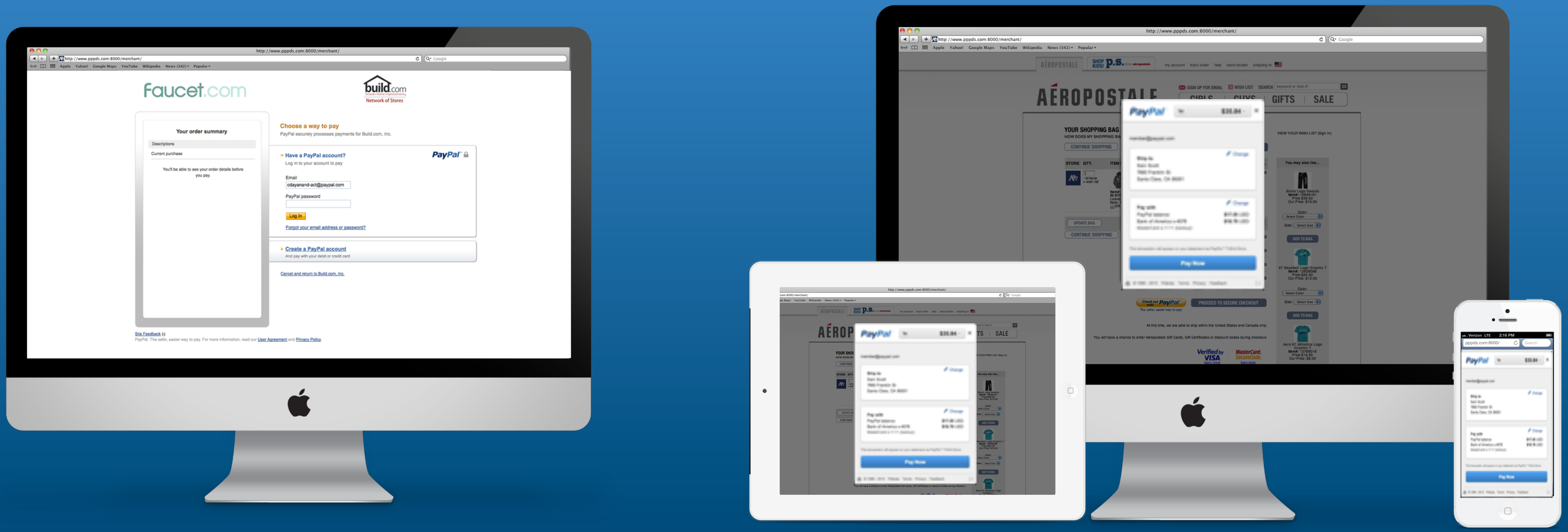
rethink engineering in the light of lean

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

engineer for experimentation

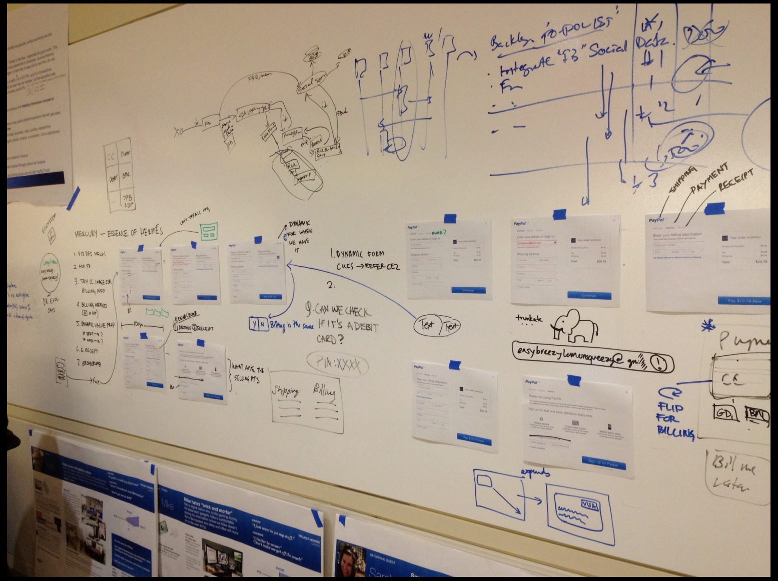
hermes project

re-inventing checkout with lean ux



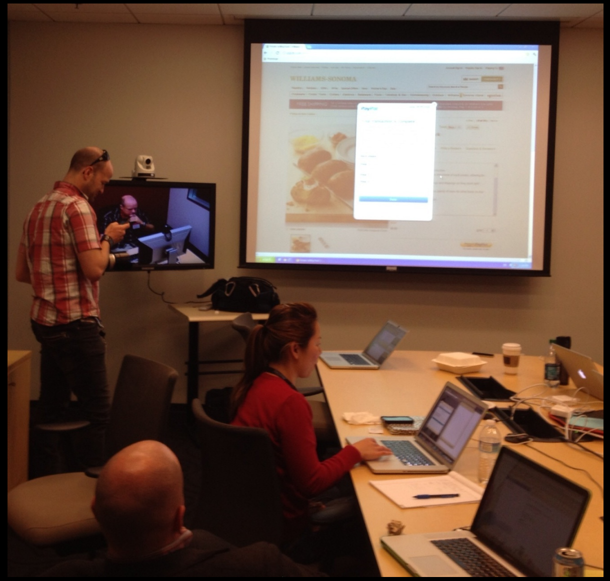
hermes project

lean ux in action



whiteboard to code

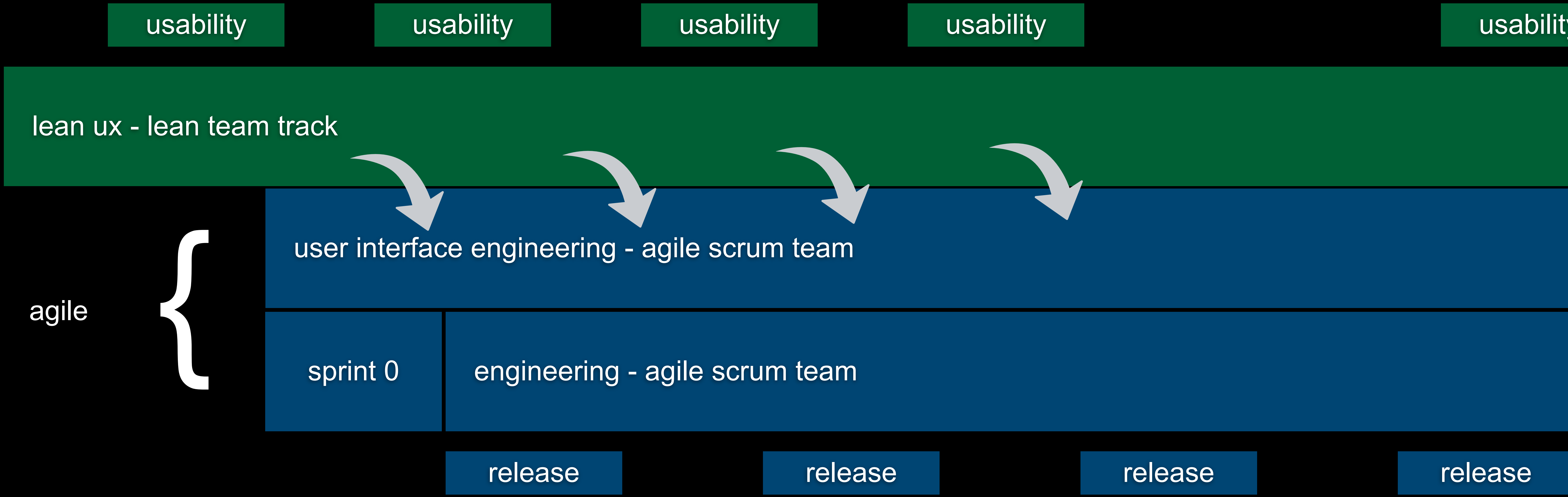
code to usability



product/design/engineering teams

usability/customers

lean ux scrum team + agile scrum teams


usability
usability
usability
usability
usabilit
lean ux - lean team track
agile
user interface engineering - agile scrum team
sprint 0
engineering - agile scrum team
release
release
release
release

three key principles that drive lean ux

remember these to keep your teams on track

shared understanding

the more understanding the less documentation

but this doesn't mean NO documentation

you need whatever is needed to gain a shared understanding

deep collaboration

strong belief that ideas come from many different voices

trust is essential

all efforts never stray far from collaborative efforts

continuous customer feedback

this is the lifeblood of the team

gets rid of politics

turns a team outside-in

the lessons learned

things we have discovered in the journey

1

create a sandbox

our lean ux teams created a sandbox for learning

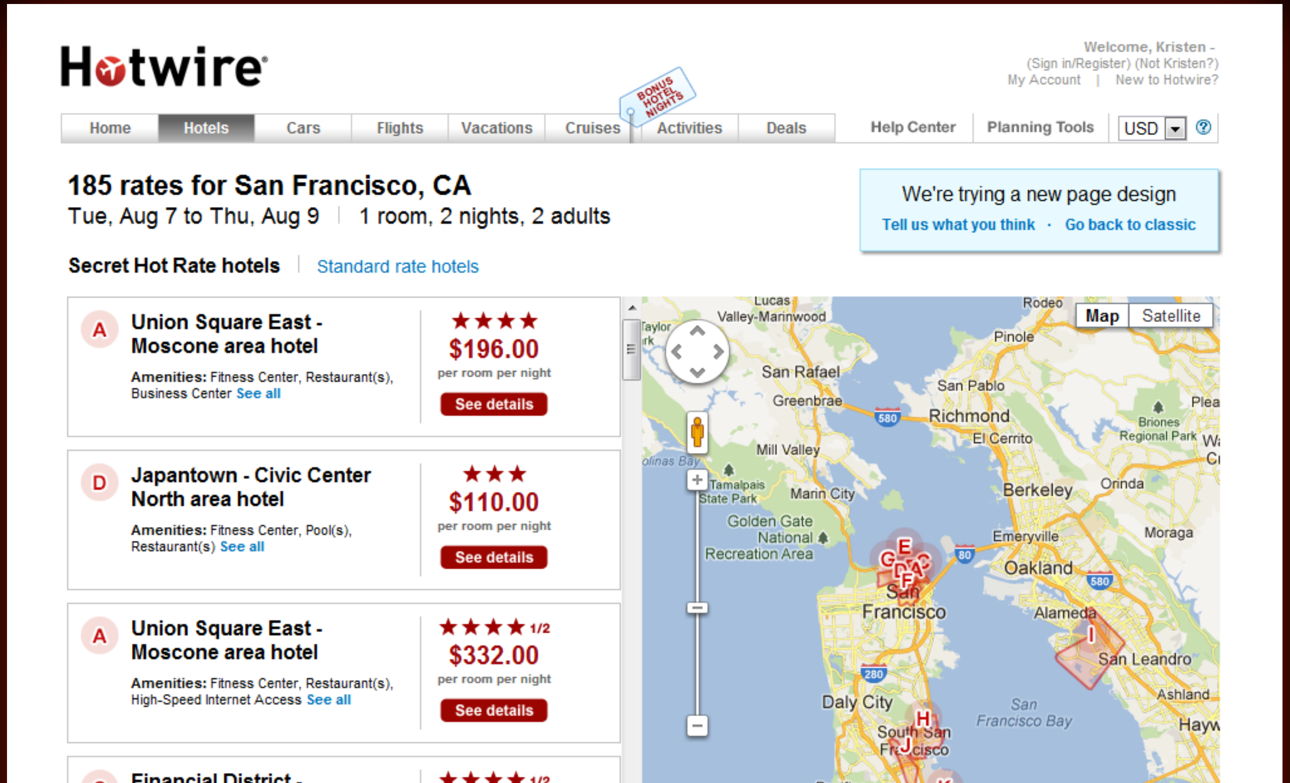
IMVU allows every engineer to put a test out to 1% of users

at netflix we often created additional tests that designers or engineers independently wanted to try as a solution to a hypothesis

hotwire case study

Source: “Lean Startup in the Hotwire Enterprise” by Kristen Mirenda & Karl Shultz

how do you protect the parent organization from the internal startup?
create a sandbox



185 rates for San Francisco, CA
Tue, Aug 7 to Thu, Aug 9 | 1 room, 2 nights, 2 adults

We're trying a new page design
Tell us what you think · Go back to classic

Secret Hot Rate hotels | Standard rate hotels

A Union Square East - Moscone area hotel
Amenities: Fitness Center, Restaurant(s), Business Center See all
★★★★ $196.00 per room per night
See details

D Japantown - Civic Center North area hotel
Amenities: Fitness Center, Pool(s), Restaurant(s) See all
★★★ $110.00 per room per night
See details

A Union Square East - Moscone area hotel
Amenities: Fitness Center, Restaurant(s), High-Speed Internet Access See all
★★★★ 1/2 $332.00 per room per night
See details

Financial District -

hotwire case study: feedback

Source: “Lean Startup in the Hotwire Enterprise” by Kristen Mirenda & Karl Shultz

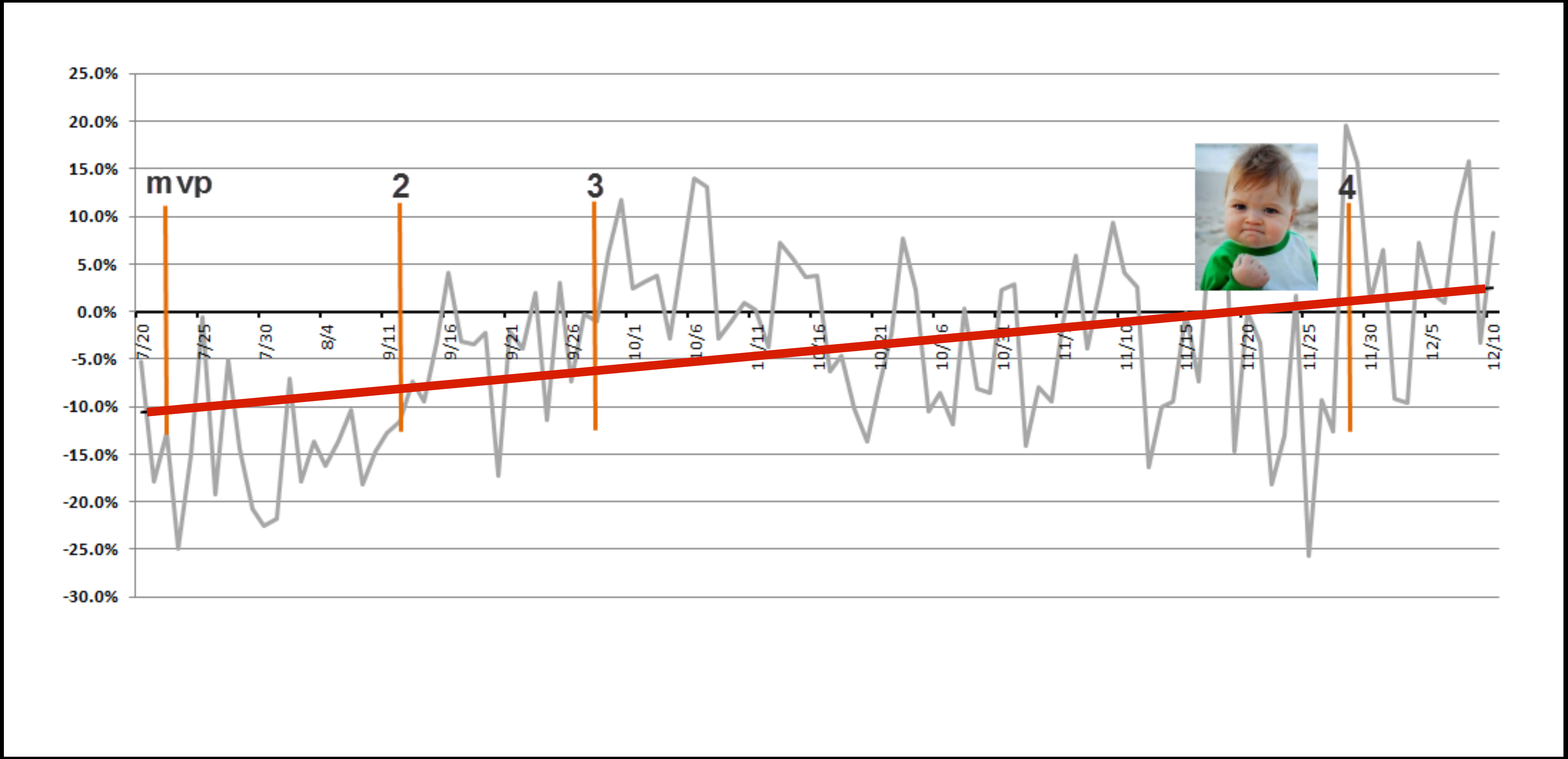
“I don't like it because you cannot filter the results or even sort them.. What were you thinking?”

“hate it - can't even sort anymore”

“absolutely blows...pure garbage. need to be able to sort asap. i'll come work for you and help you figure it out. wtf.”

hotwire case study: data

Source: “Lean Startup in the Hotwire Enterprise” by Kristen Mirenda & Karl Shultz

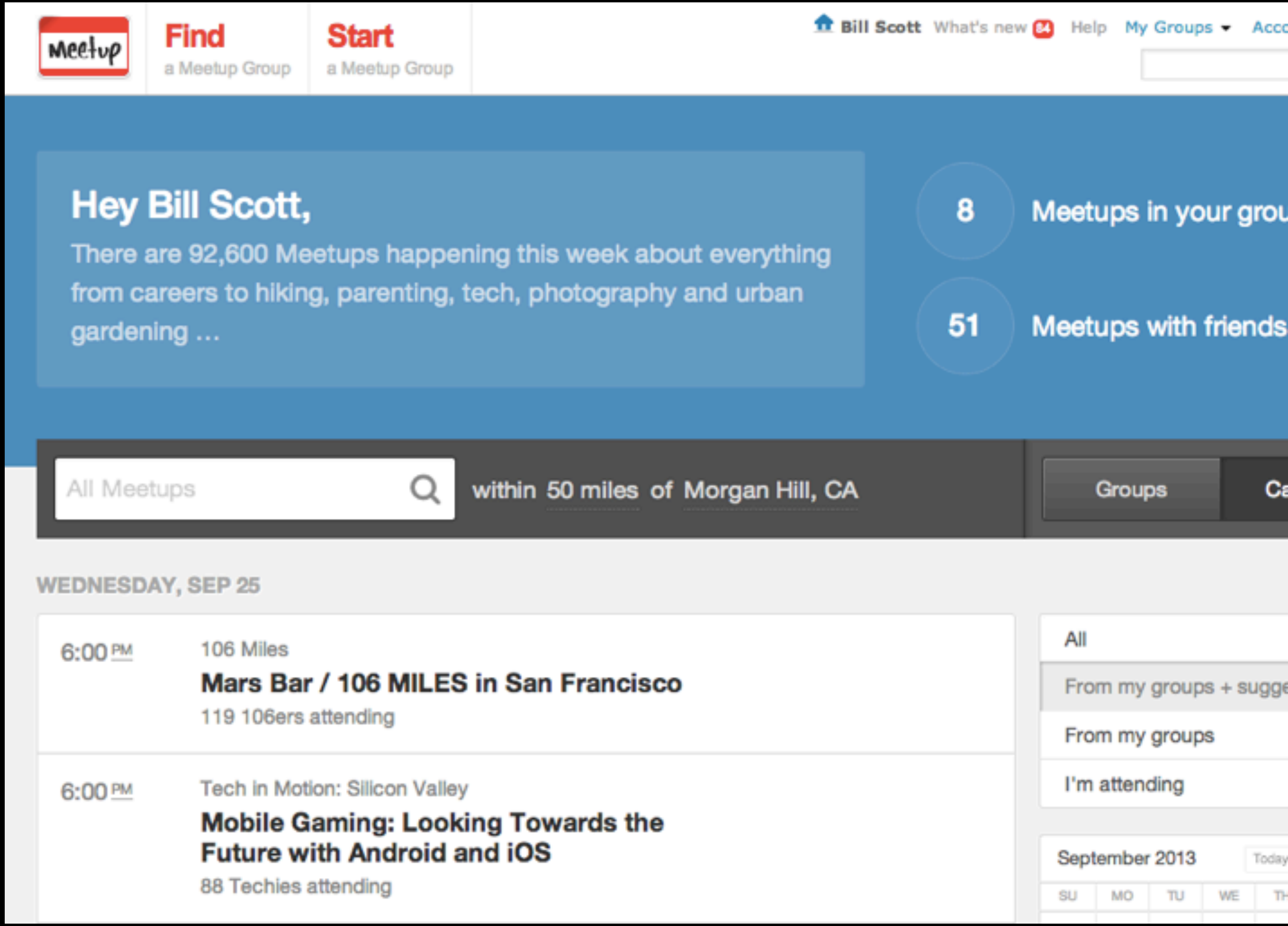


meetup.com

hold 400-500 usability sessions a year

anyone can request a study

- self-service



Find a Meetup Group

Start a Meetup Group

Bill Scott What's new 84 Help My Groups Acc

Hey Bill Scott,

There are 92,600 Meetups happening this week about everything from careers to hiking, parenting, tech, photography and urban gardening ...

8 Meetups in your grou

51 Meetups with friends

All Meetups within 50 miles of Morgan Hill, CA

Groups Ca

WEDNESDAY, SEP 25

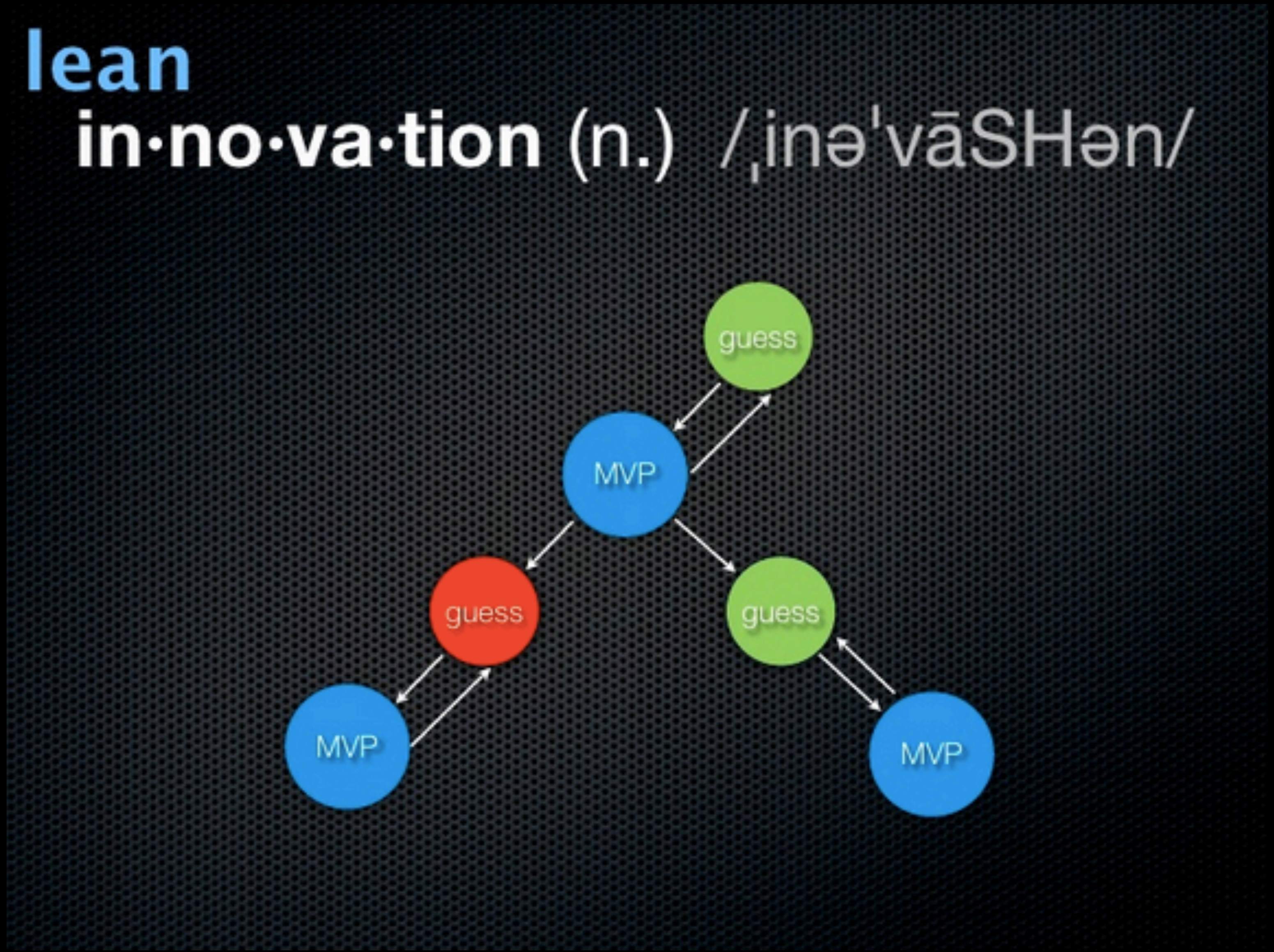
6:00 PM 106 Miles
Mars Bar / 106 MILES in San Francisco
119 106ers attending

6:00 PM Tech in Motion: Silicon Valley
Mobile Gaming: Looking Towards the Future with Android and iOS
88 Techies attending

All
From my groups + sugge
From my groups
I'm attending

September 2013 Today
SU MO TU WE TH

edmunds



http://ismailelshareef.com/2012/06/13/lean-innovation-how-to-become-an-effective-innovator/

wanted to try facebook connect

sandbox -> created a POC with small amount of traffic

significant uplift

became the new experience

move to a “living spec”

break down barriers between prototyping and production

use developers for prototyping as forcing function

embrace RITE

avoid tools/process that get away from collaboration

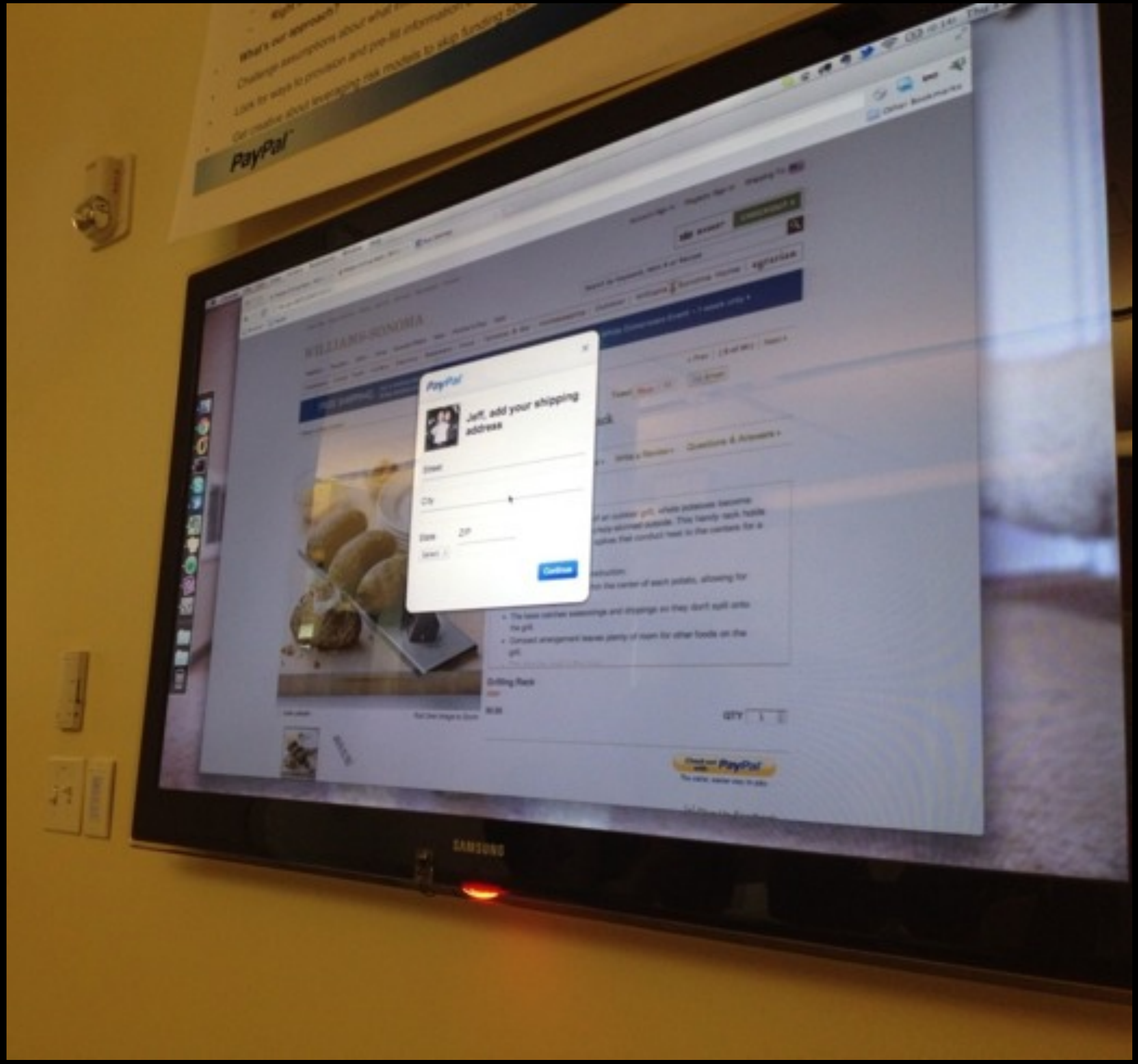
make the spec real

there are many, many prototyping tools available now

you can create a living
spec with these

however the fidelity
is never the same as
real code

recommend HTML
prototyping as get closer to agile
(more on this later)



but what about docs?

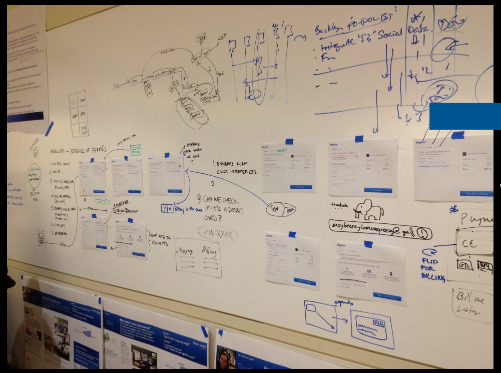
watch out for “predictive documentation”

watch out for documentation that replaces collaboration or is a band-aid for bad process

good documentation will enhance collaboration, shared understanding and disseminate learnings

use a prototype stack

to enable learning



whiteboard
to code

code to usability

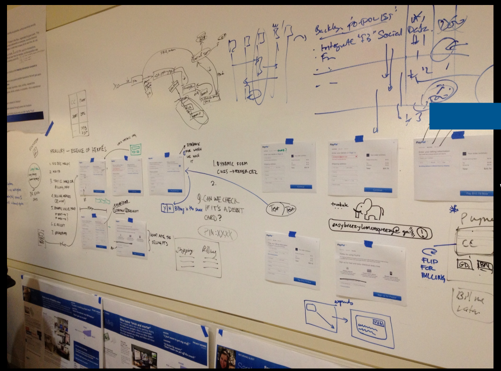
product/design
team

user interface
engineers

usability/customers

use a prototype stack

to enable learning



whiteboard
to code

code to usability

product/design
team

user interface
engineers

usability/customers

JS Templating
(dustjs)

JS libraries

less -> CSS

images

nodejs

enables sketch to code

forcing function

- it brings about a close collaboration between engineering and design
- it creates a bridge for shared understanding

requires a lot of confidence and transparency

3

engineer for experimentation

long shelf life to rapid experimentation

focus on learning not on delivery

design for volatility

refactor the tech stack with learning in mind

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

the netflix way

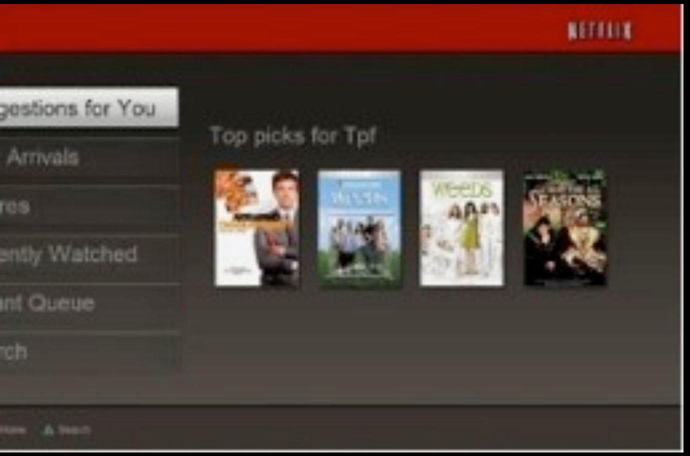
16 different test cells in the initial PS3 Launch (2010)

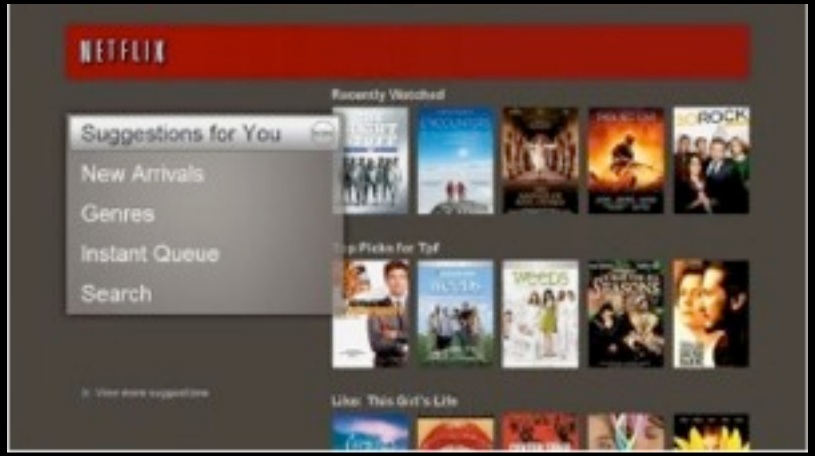
focus is on **experimentation**

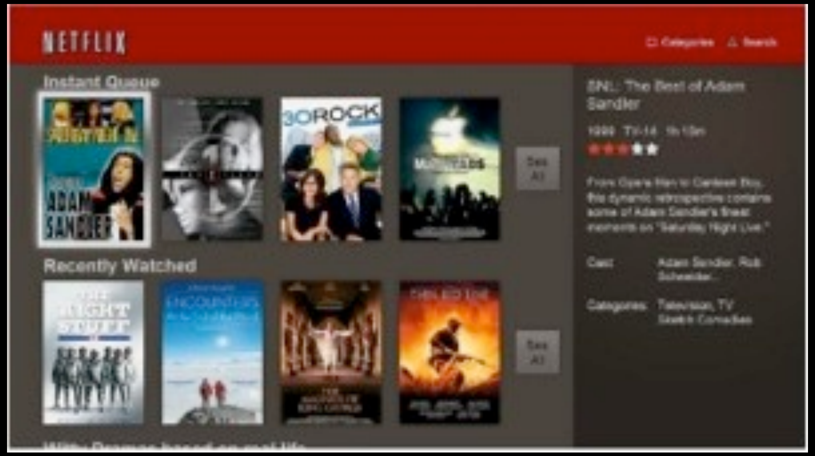
the netflix way

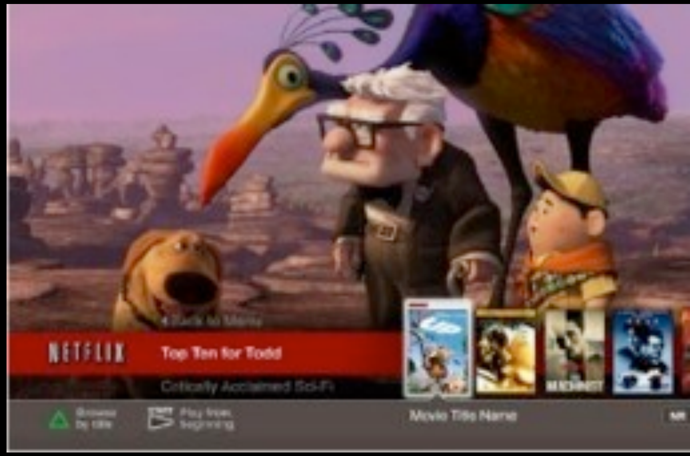
16 different test cells in the initial PS3 Launch (2010)

focus is on **experimentation**



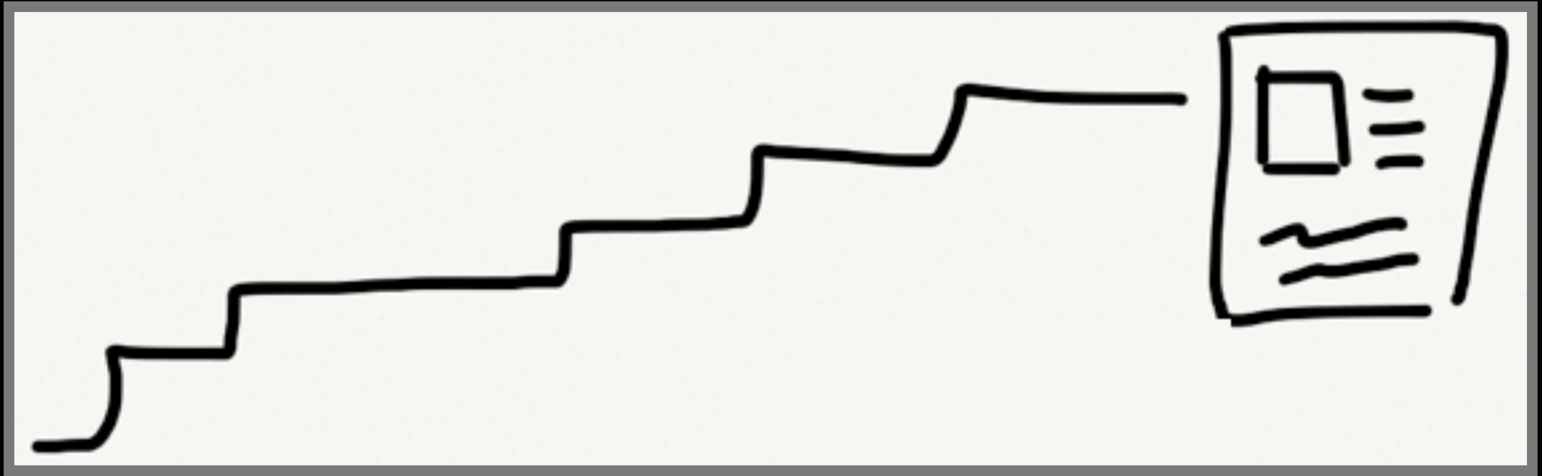






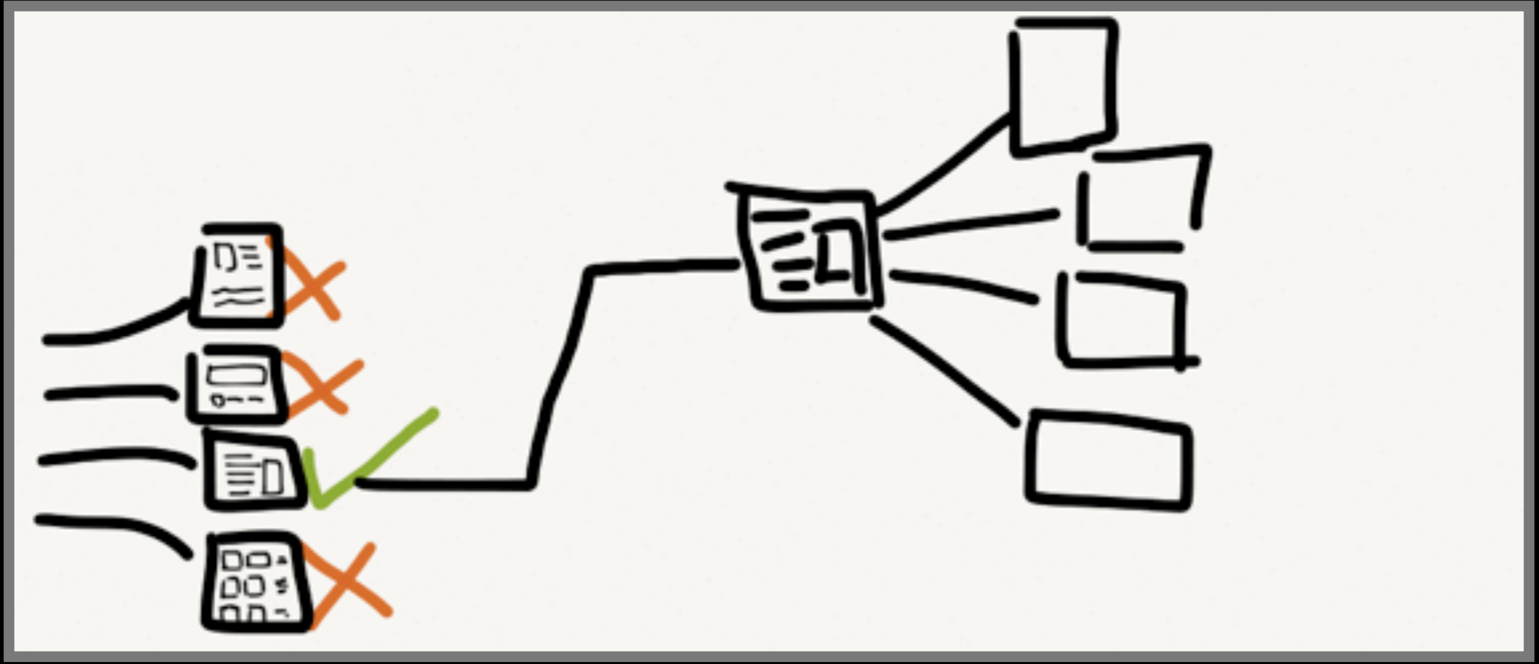
four distinct PS3 experiences launched on same day

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

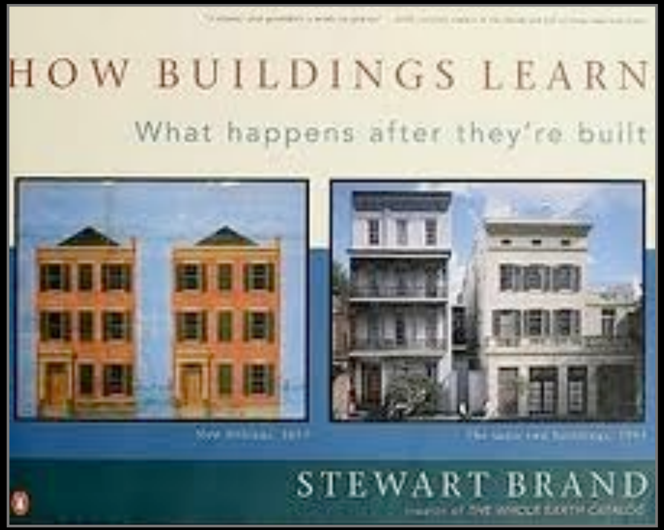
starting to use git repo model for continuous deployment

- 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

experiences must learn

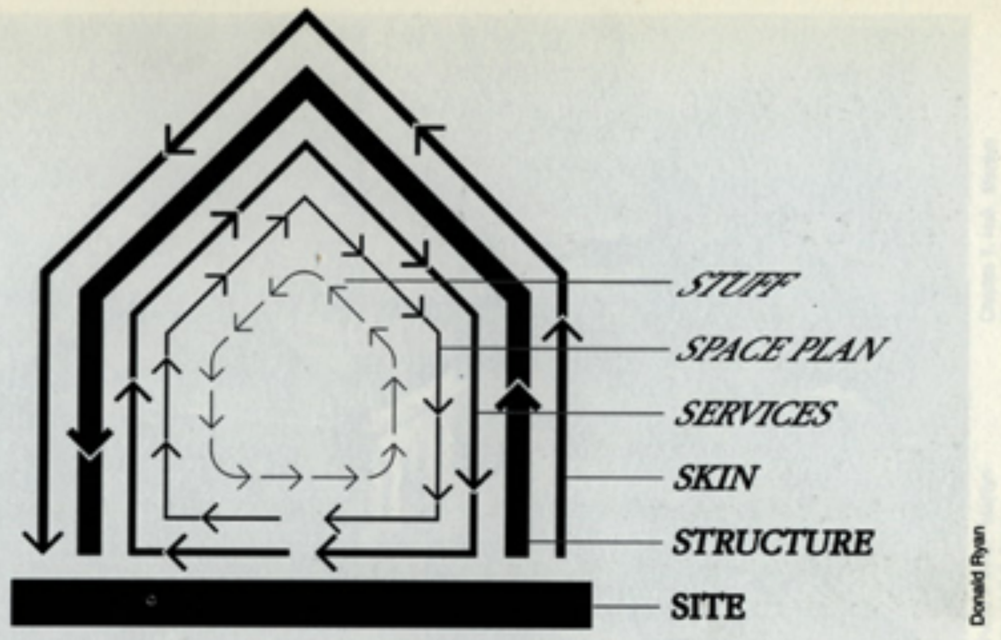


All buildings are predictions.
All predictions are wrong.

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

Stewart Brand

13



SHEARING LAYERS OF CHANGE. Because of the different rates of change of its components, a building is always tearing itself apart.

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

Recognize that different layers change at different velocities

you have to engineer for volatility

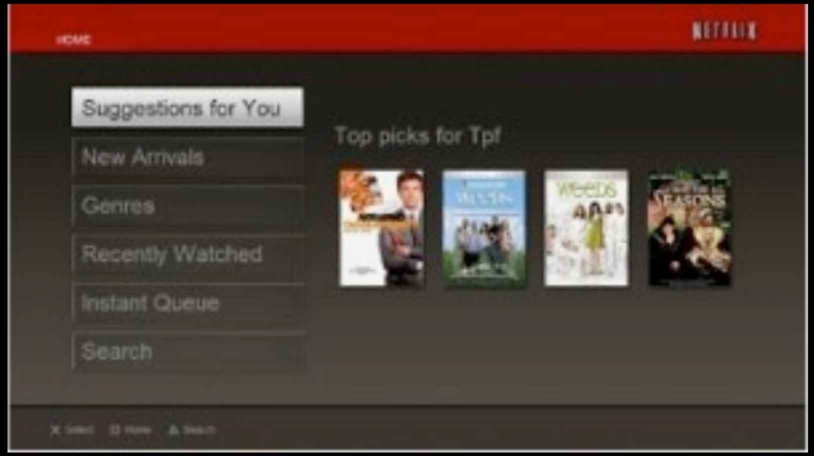
majority of the experience code written is thrown away in a year

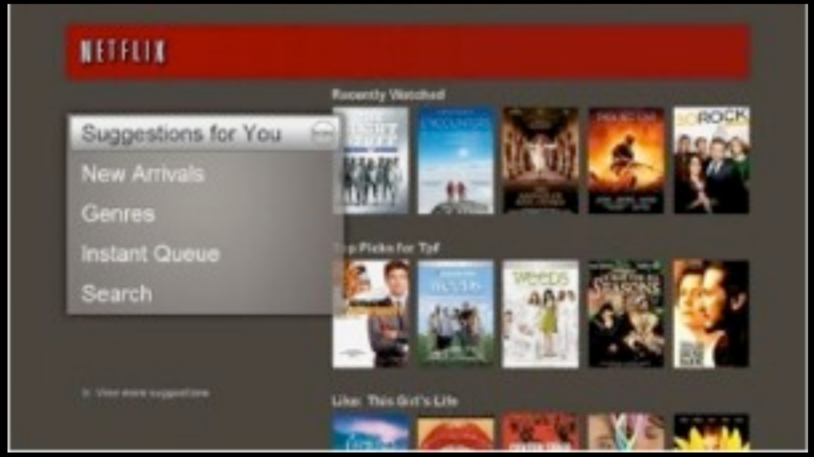
change is the norm

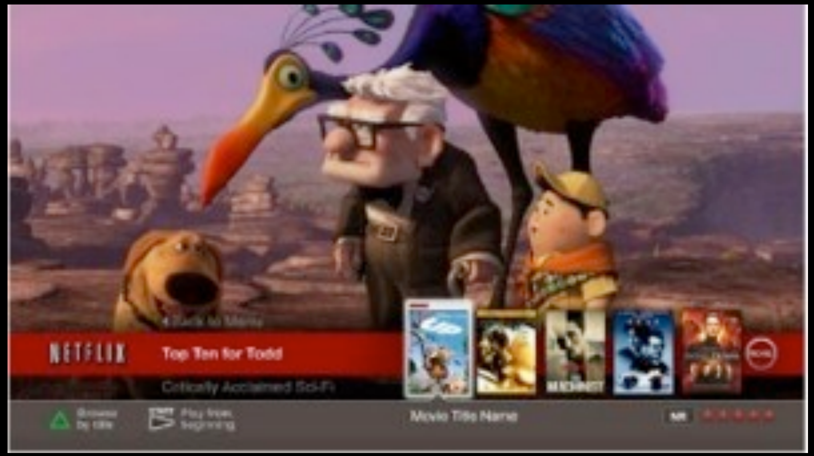
experimentation is not a one time event

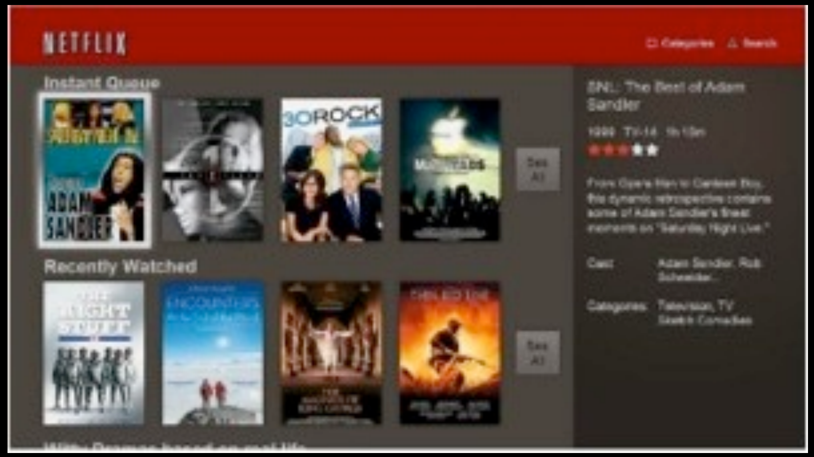
launching a product is giving birth to the product. the product's life just begins.

design for throwaway-ability









you have to engineer for volatility

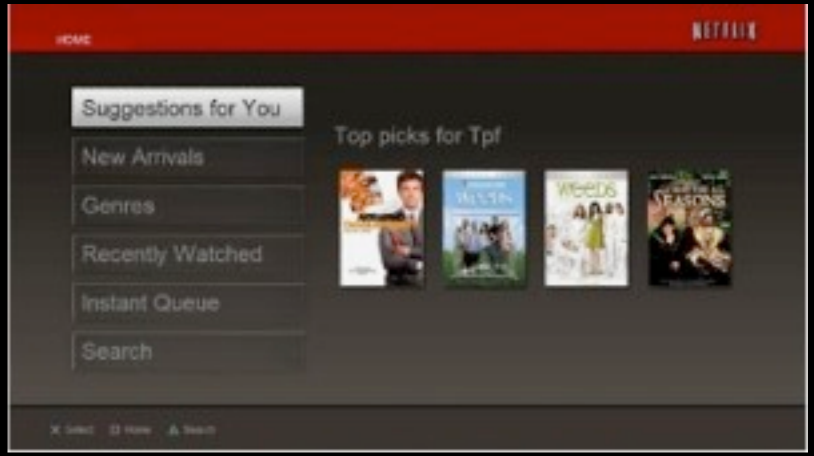
majority of the experience code written is thrown away in a year

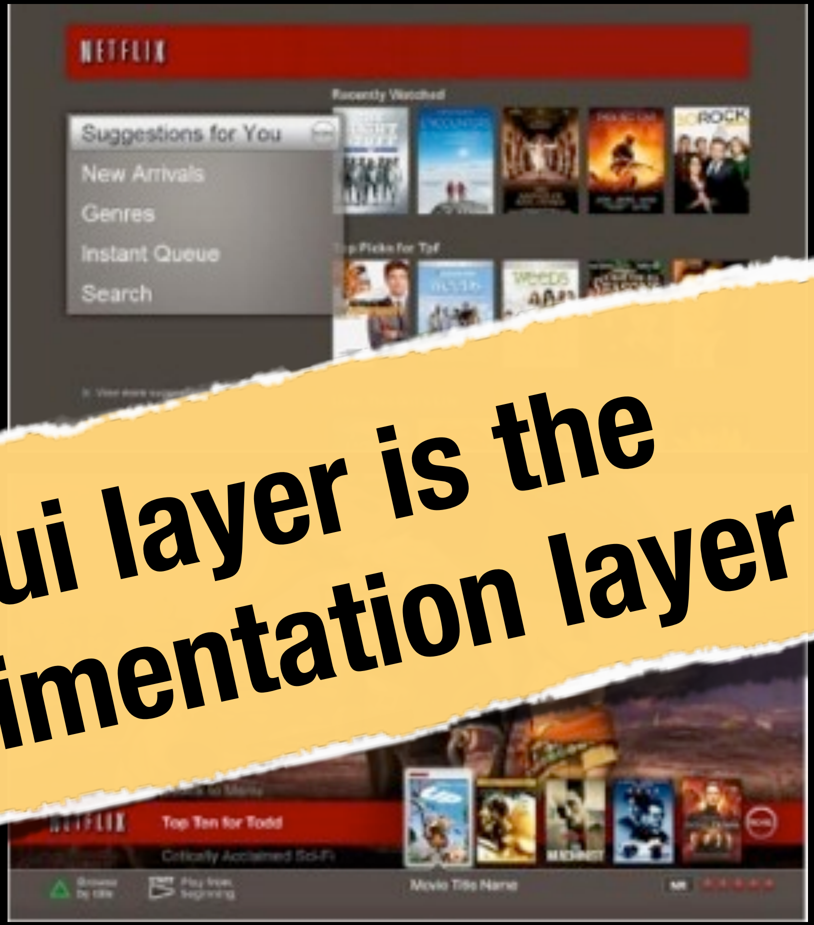
change is the norm

experimentation is not a one time event

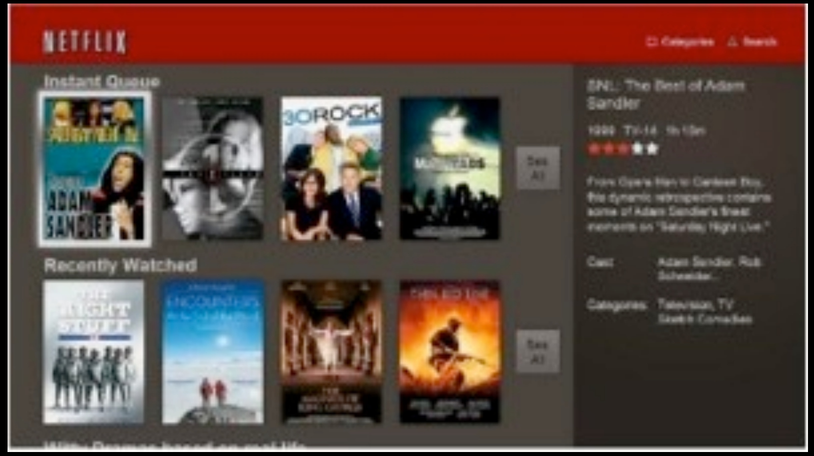
launching a product is giving birth to the product. the product's life just begins.

design for throwaway-ability





the ui layer is the experimentation layer



you have to engineer the tech stack

independent of the backend language

flexible enough to run in either the **server or in the client**

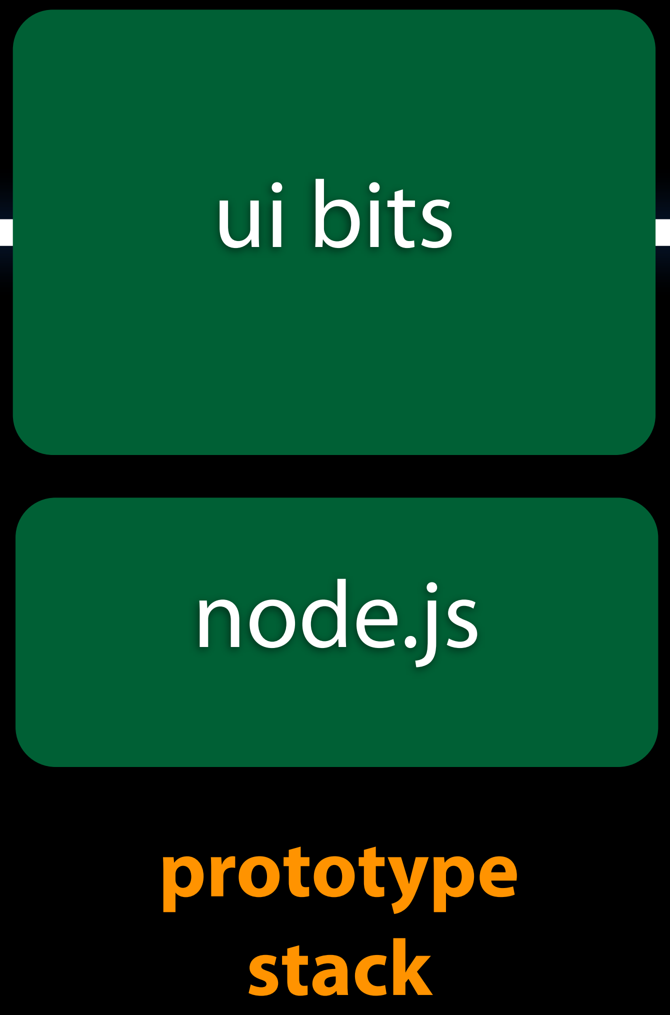
equally good at building web **sites** as it is building web **applications**

pushable outside of the application stack (publish model)

utilize what is **common to developers**

components built with **change** in mind

1st step: fire up a prototype stack (nodejs)

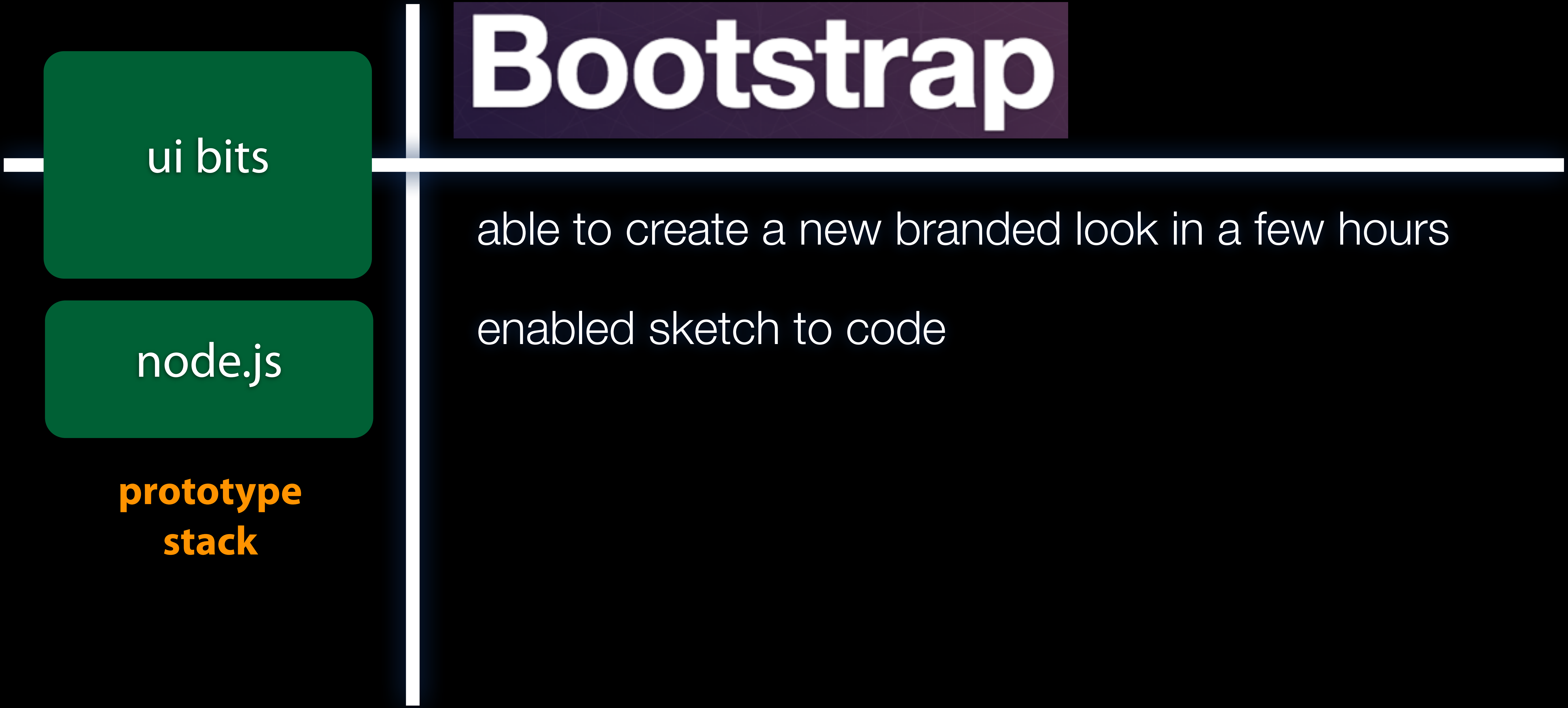


utilize opens source stack

express, connect, require.js

bring in javascript templating and other open source ui goodness

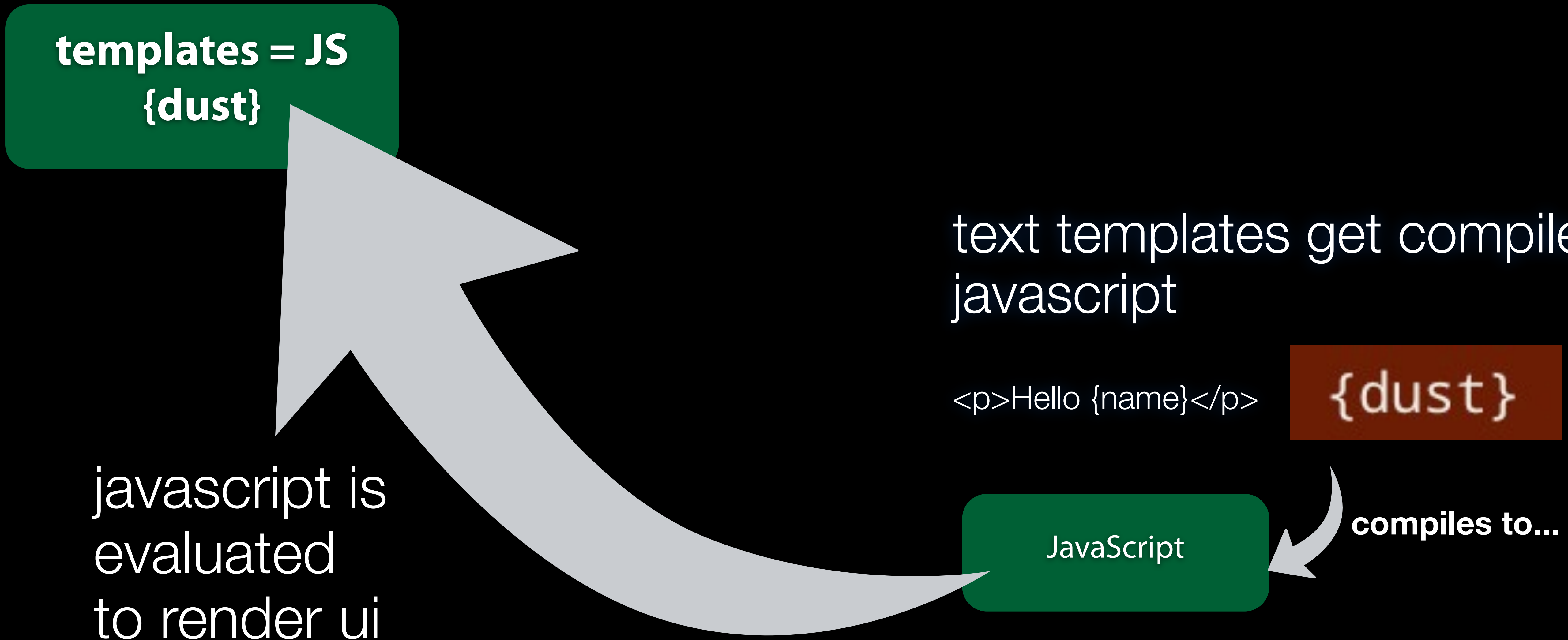
2nd step: bootstrap with bootstrap



able to create a new branded look in a few hours

enabled sketch to code

3rd step: use javascript templating



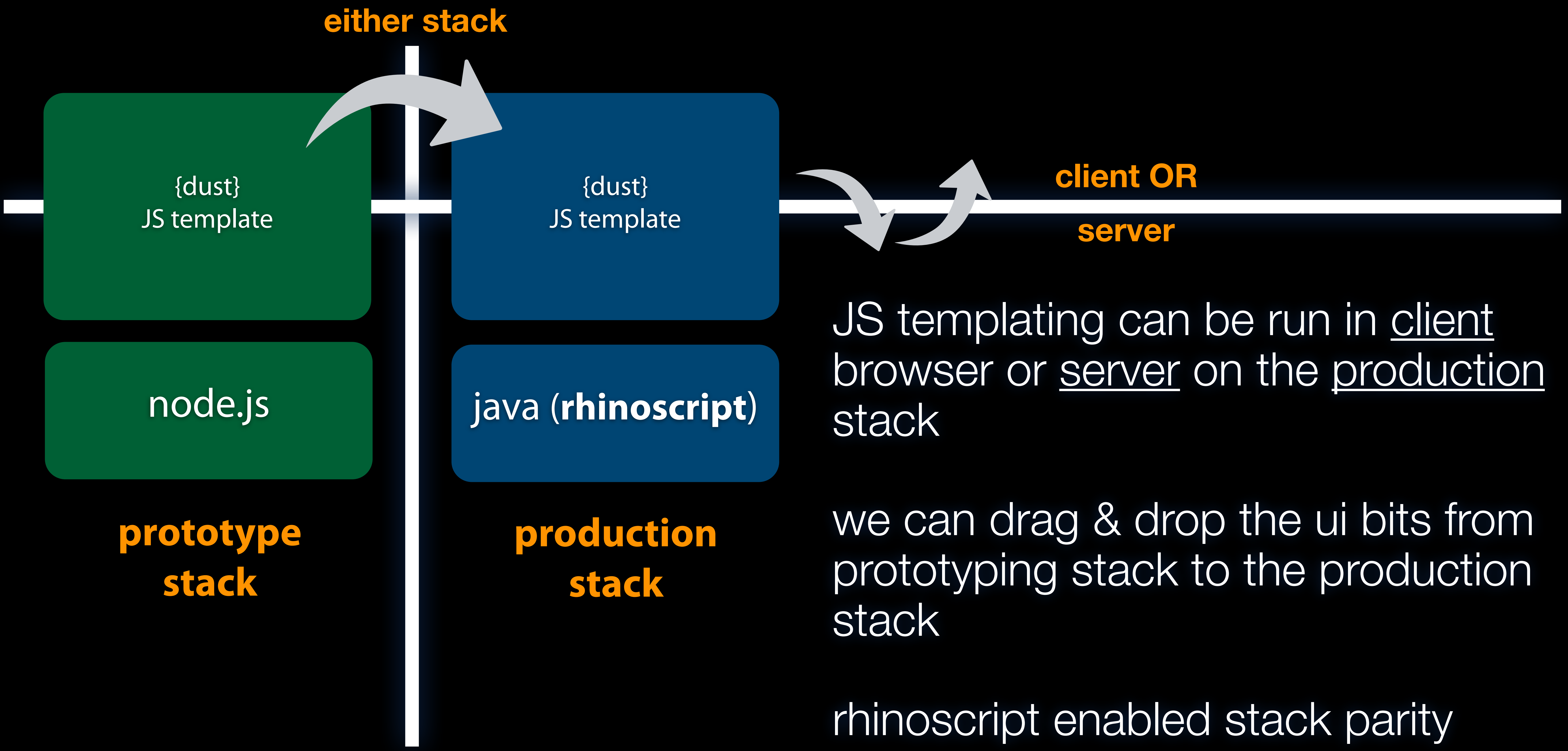
text templates get compiled to javascript

```
<p>Hello {name}</p>
```

javascript is evaluated to render ui

dustjs templates execute wherever there is javascript

4th step: make ui bits portable to legacy



JS templating can be run in <u>client</u> browser or <u>server</u> on the <u>production</u> stack

we can drag & drop the ui bits from prototyping stack to the production stack

rhinoscript enabled stack parity

5th step: build on open source

Bootstrap

less

UNDERSCORE.JS

jQuery mobile framework

BACKBONE.JS

REQUIRE.JS

{dust}

Passport

nconf

q

async

supertest

node JS

express

kraken

contains "webcore" for scaffolding and providing a lightweight framework for dev & production

Chai Assertion Library

mocha

prototype & production stack

6th step: bring node to production

project kraken

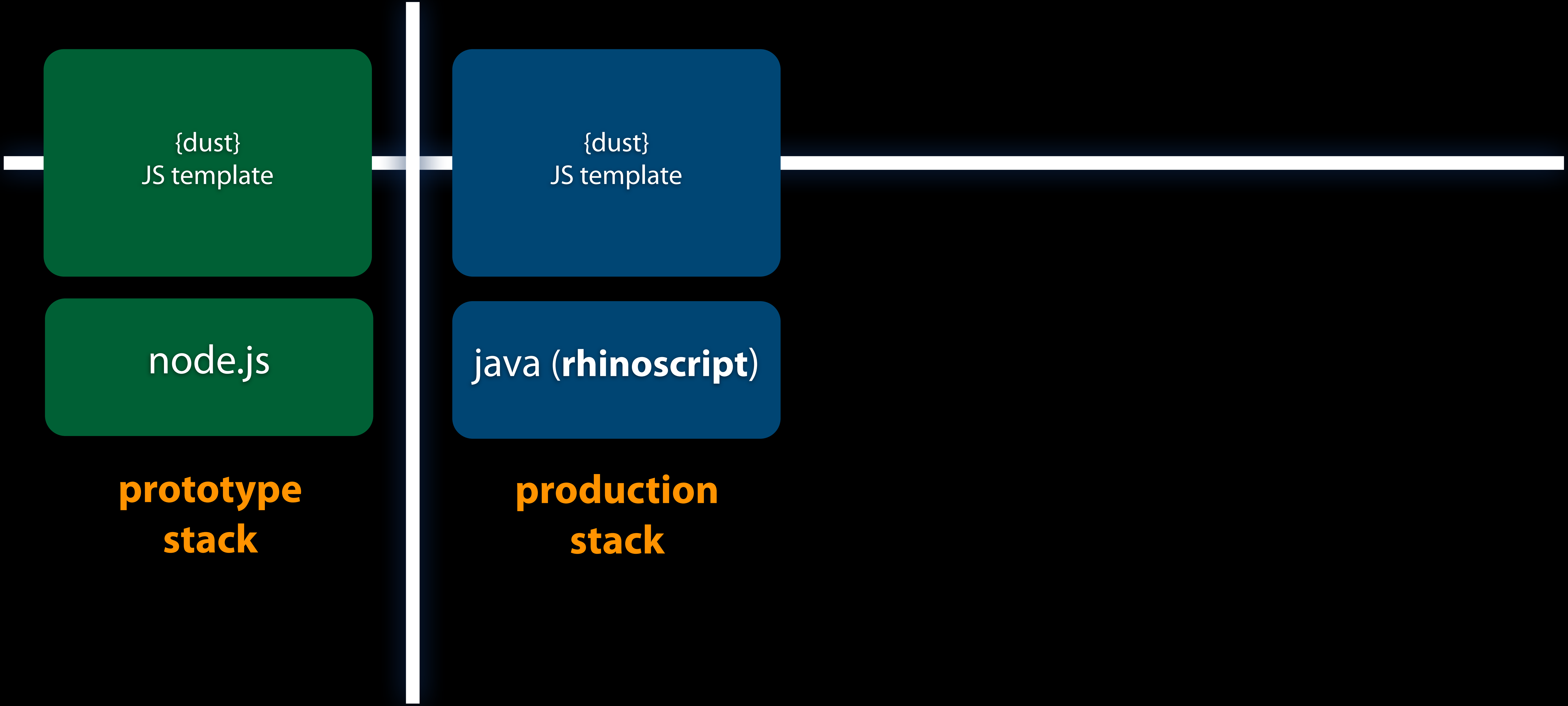
enable all of the standard paypal services WITHOUT looking like PayPal

but do it in a friendly npm way

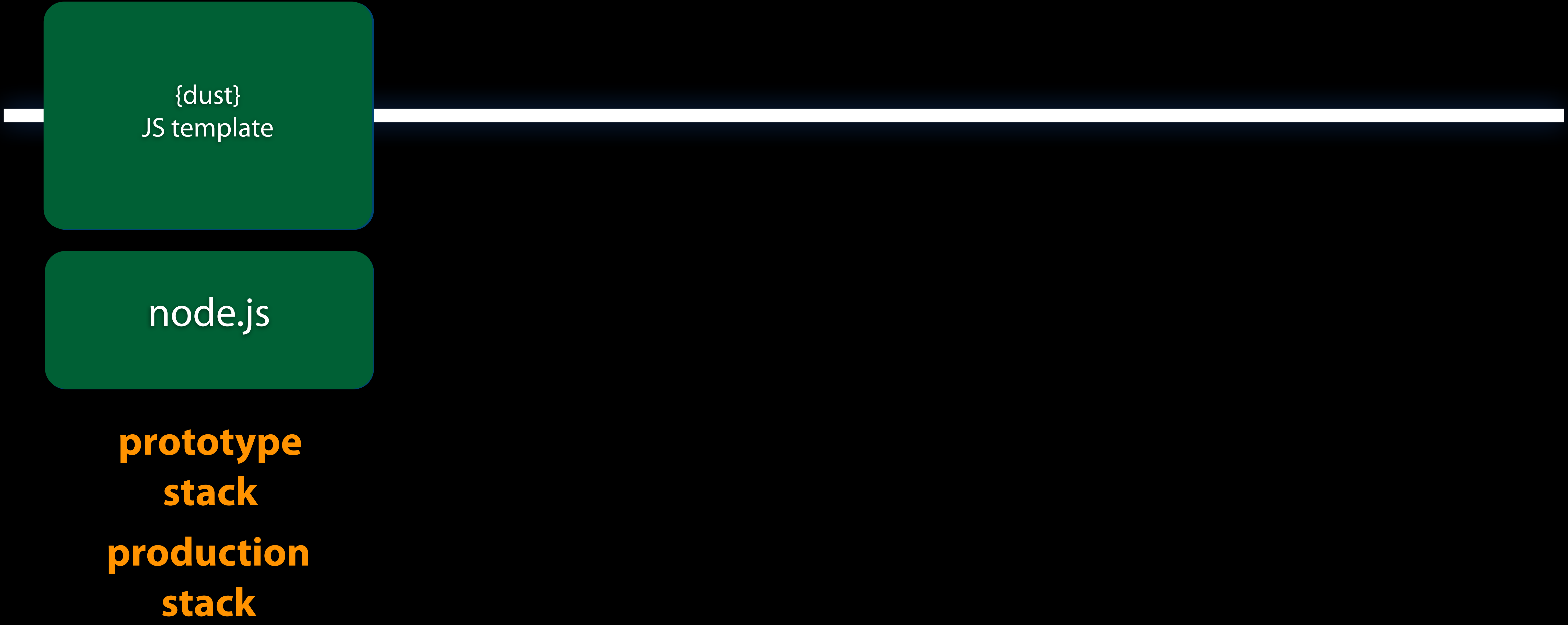
> monitoring, logging, security, content, locale resolution, analytics, authentication, template rendering, experimentation, packaging, application framework, deployment, session management, service access, etc.

simplifies creating an app in a few minutes with all paypal services

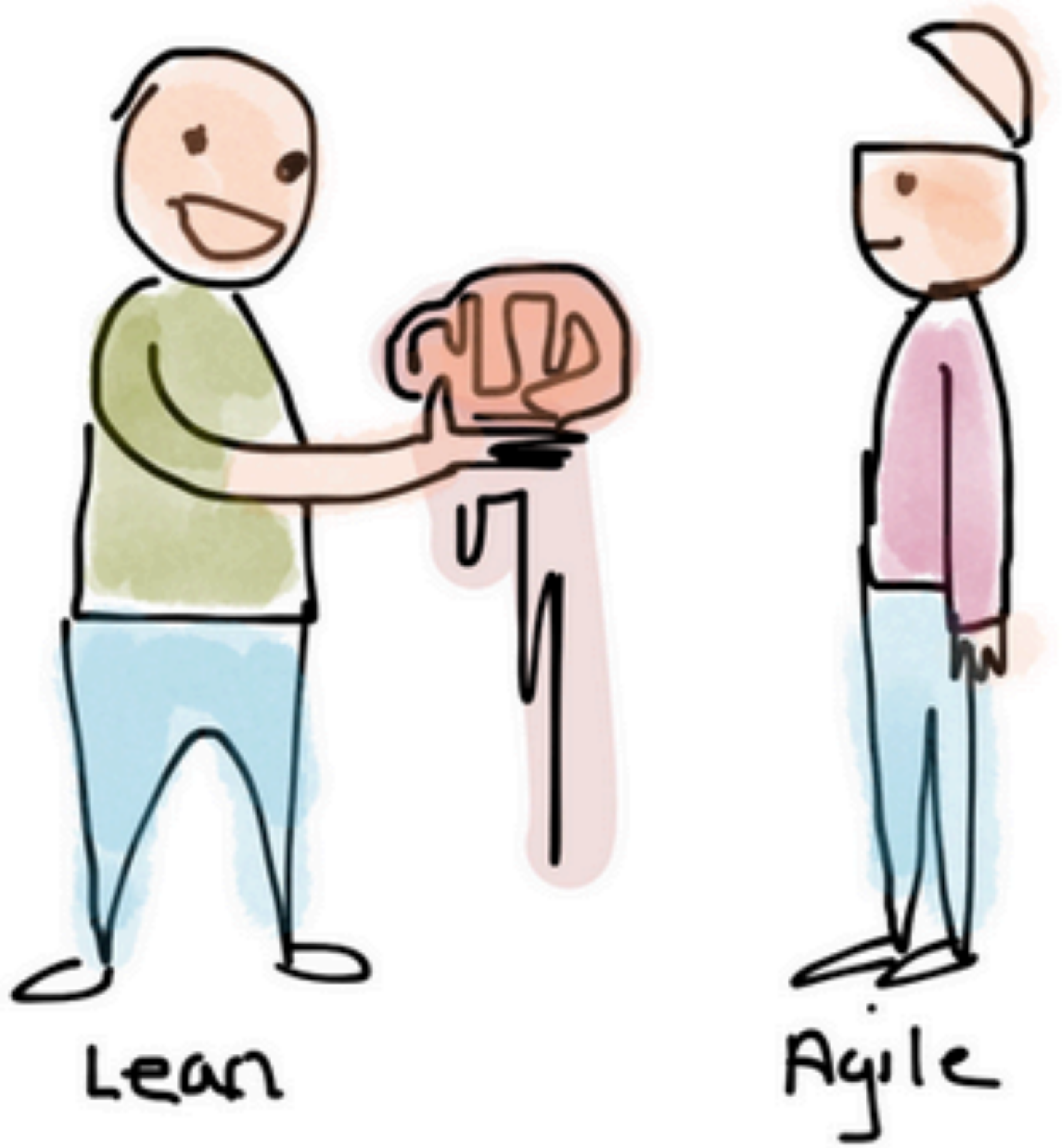
7th step: one stack to rule them all



7th step: one stack to rule them all



4



give agile a brain

use lean ux as the brain for agile

develop a lean cadence

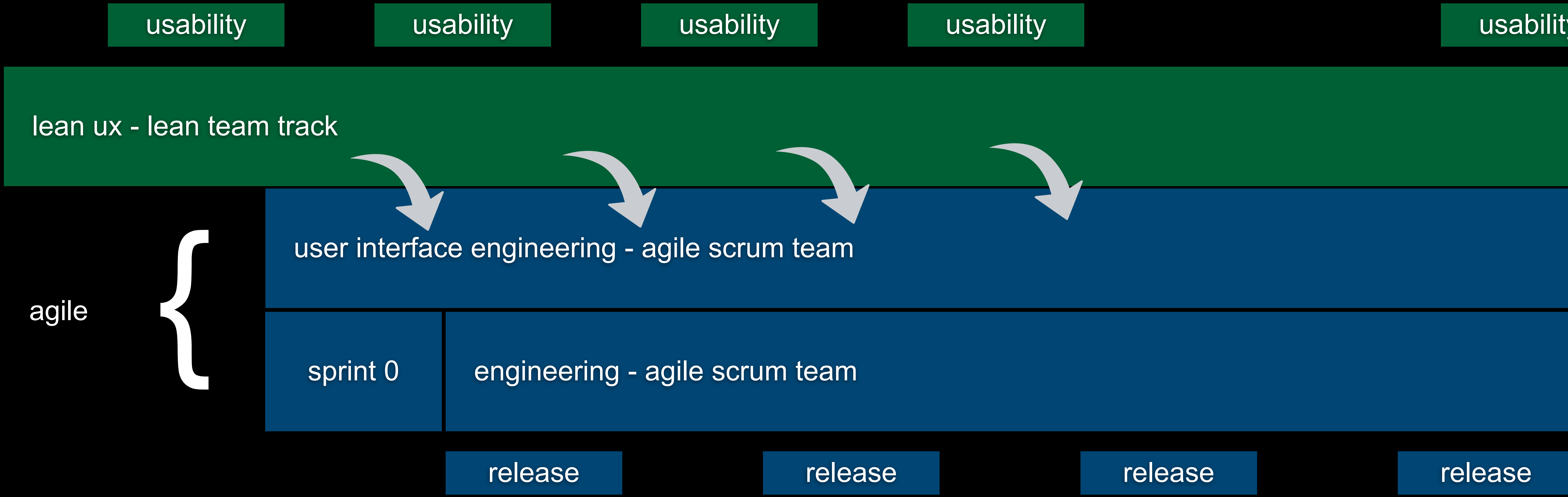
involve all members in lean ux (balanced teams)

credit: Krystal Higgins

http://bit.ly/18uP7N1

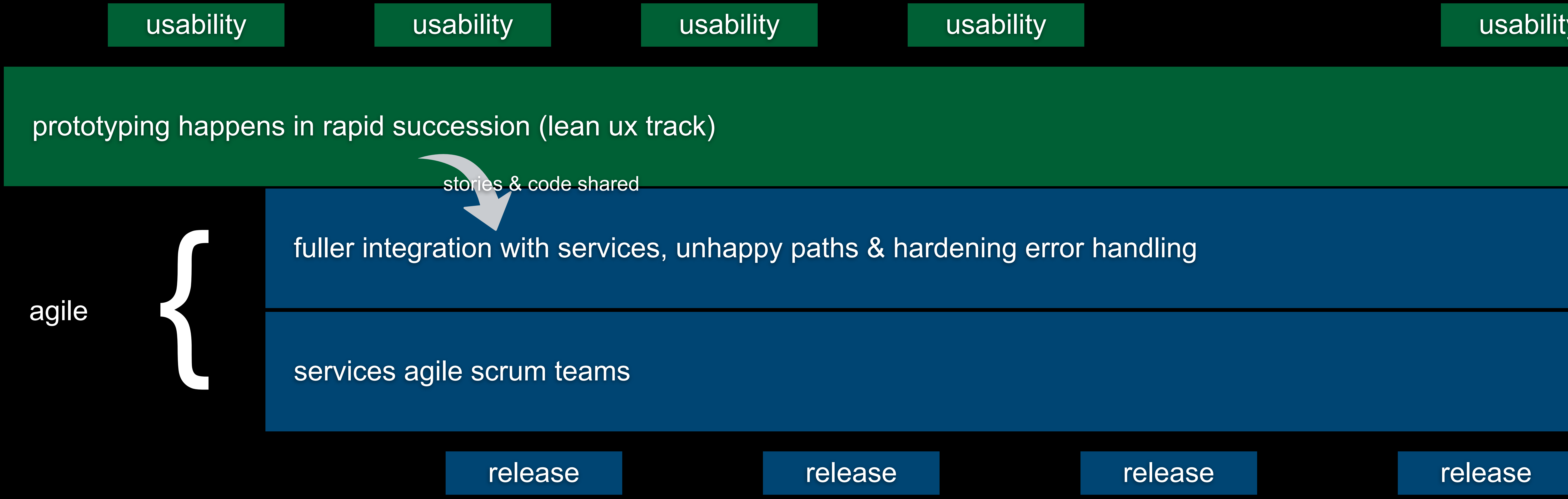
free to iterate independent of agile

lean ux can provide a brain for agile


usability
usability
usability
usability
usabilit
lean ux - lean team track
agile
user interface engineering - agile scrum team
sprint 0
engineering - agile scrum team
release
release
release
release

how lean & agile can play together

lean ux can provide a brain for agile


usability
usability
usability
usability
usabilit
prototyping happens in rapid succession (lean ux track)
stories & code shared
agile
fuller integration with services, unhappy paths & hardening error handling
services agile scrum teams
release
release
release
release

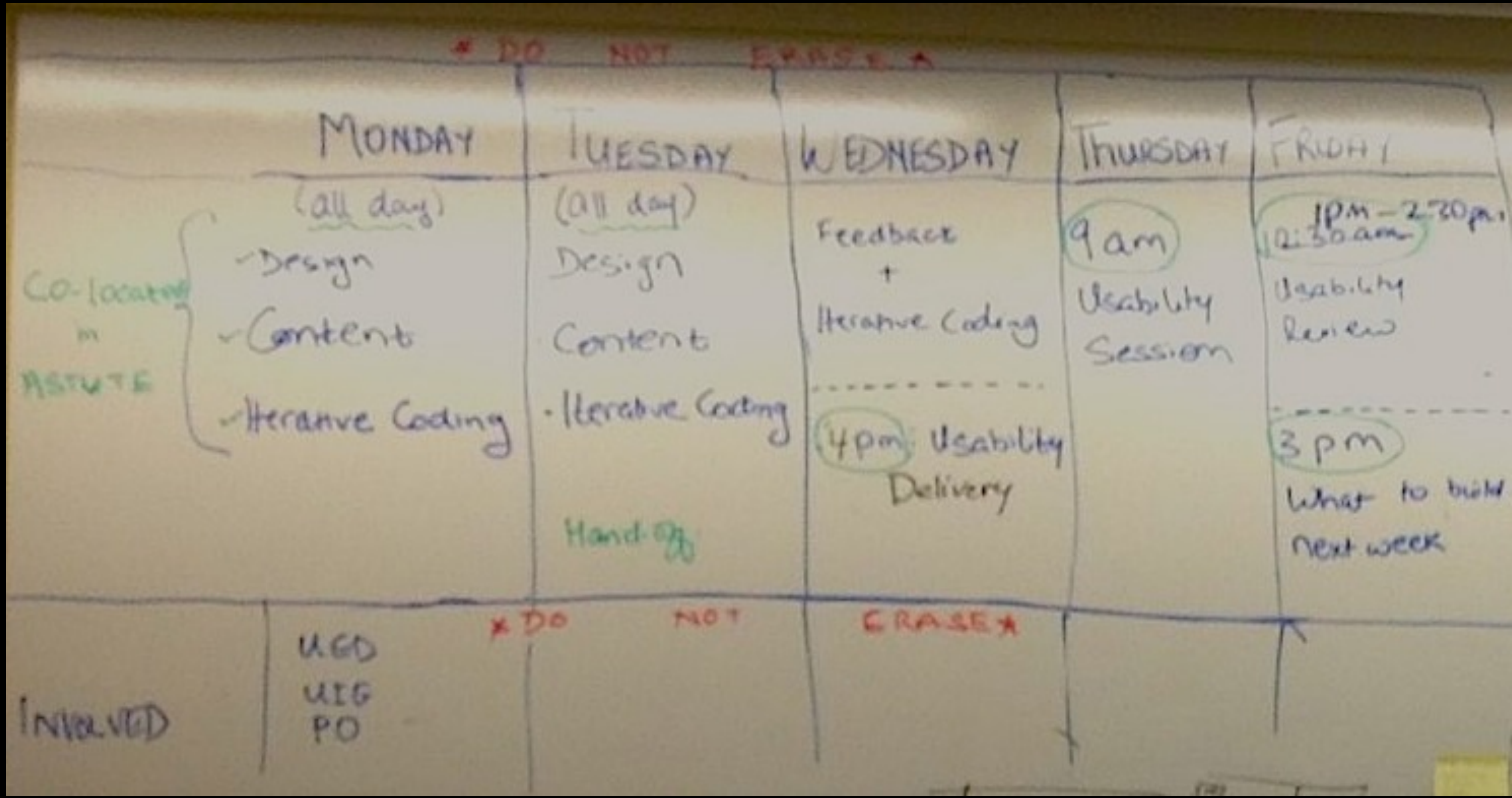
lean & agile teams should blend together

lean scrum team sprints

focus on getting to customer as early and as often as possible

removes the politics in the team as this becomes the arbiter

you can slow down this cadence after you converge on key hypotheses and potential solutions



* DO NOT ERASE *

		MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
Co-located in ASTUTE		(all day) ✓ Design ✓ Content ✓ Iterative Coding	(all day) Design Content · Iterative Coding Hand-off	Feedback + Iterative Coding - - - 4pm Usability Delivery	9am Usability Session	1pm – 2:30pm 12:30am Usability Review - - - 3pm What to build next week
INVOLVED	UGD UIG PO					

* DO NOT ERASE *

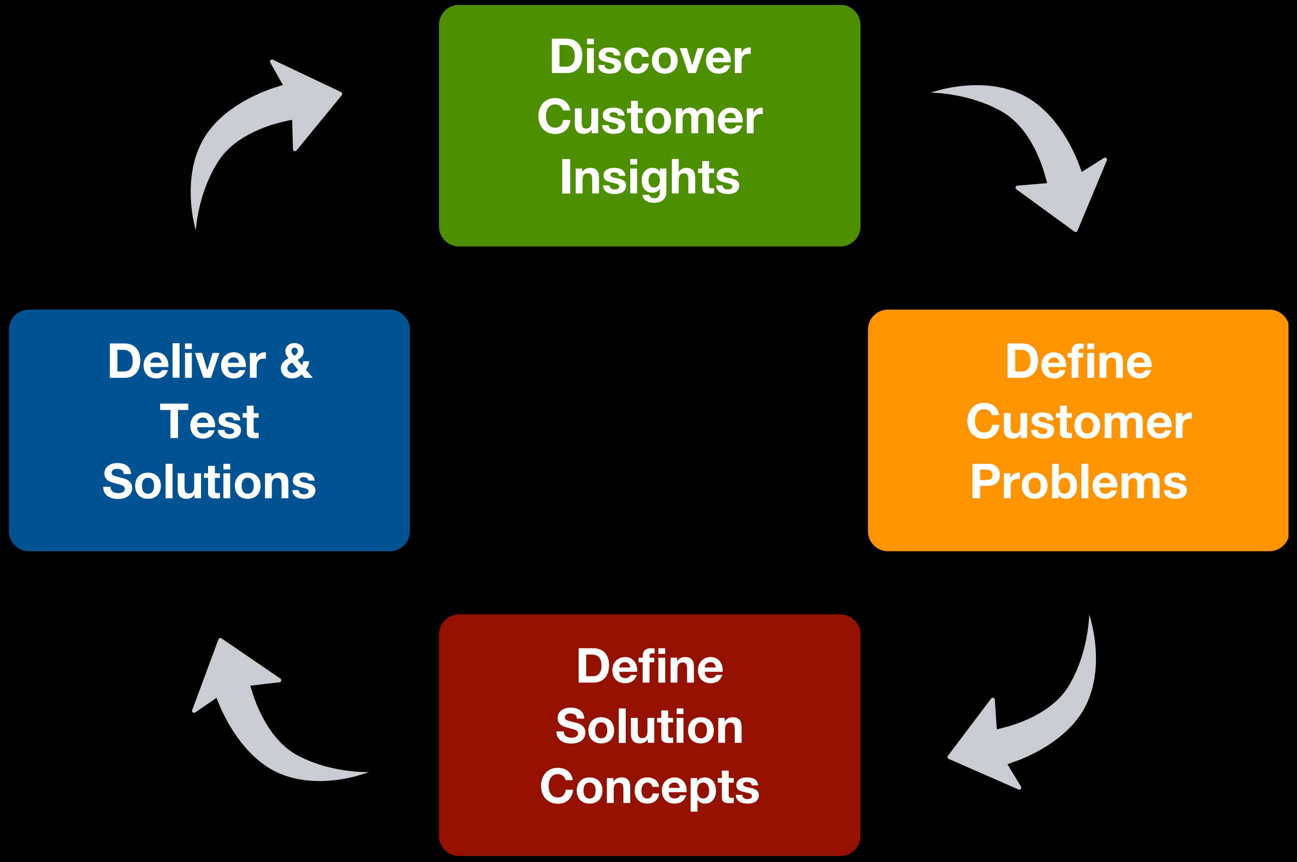
lean scrum teams

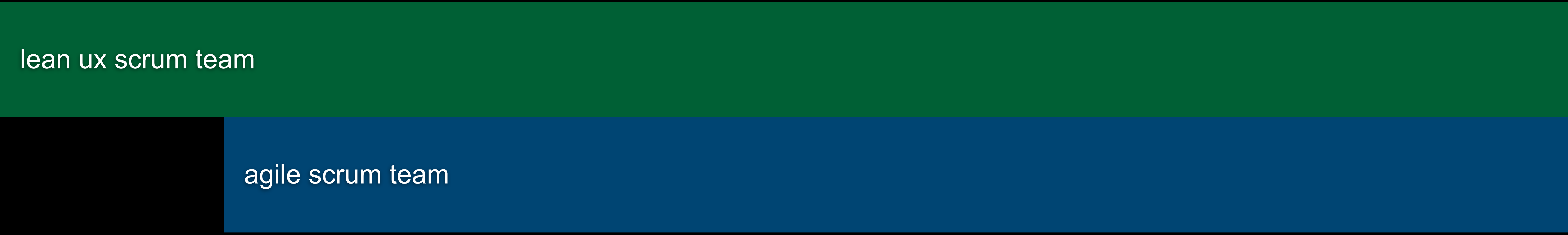
dedicate a lean ux “scrum master”

stay 2 sprints ahead of agile
(compare with dual scrum team model)

feed the agile backlog from lean scrum team

make it part of the overall larger lean ux process





become hypothesis driven

learn to state design goals in terms of solving specific customer problems

don’t be afraid to invalidate hypotheses

look for the pivot

hypothesis thinking

really hard for designers to think in this way

applies the scientific method

hypotheses form strategies that you can design against

apply F.O.G.

MVPs can be used to invalidate assumptions (or validate)

6

embrace the problem not the solution

engineering: don’t start with technology, start with experience

design: get your ideas out early

together: get in front of customers so problem is the focus, not our current solution

co-locate if at all possible

high bandwidth “meatspace” facilitates **shared understanding** and **deep collaboration**

also facilitates shared time with the **customer**

suggestions

at a minimum teams should come together for the first few weeks to build shared understanding, deep collaboration and getting feedback from customers

for distributed members use high bandwidth communication where possible (skype, tele-presence)

high bandwidth communication necessary.

github counterpoint

electronic: discussion, planning and operations process should be in high fidelity electronics.

available: work should be visible and expose process. work should have a URL. single source truth.

asynchronous: almost nothing should require direct interruption.

lock-free: avoid synchronization points.

cooperation without coordination

http://tomayko.com/writings/adopt-an-open-source-process-constraints

tools that can help

yammer
The Enterprise Social Network

asana

POSTMAN

collabedit
simple collaborative text

Trello

box

sococo Social Communications Company

Google+

Hangouts!

g+

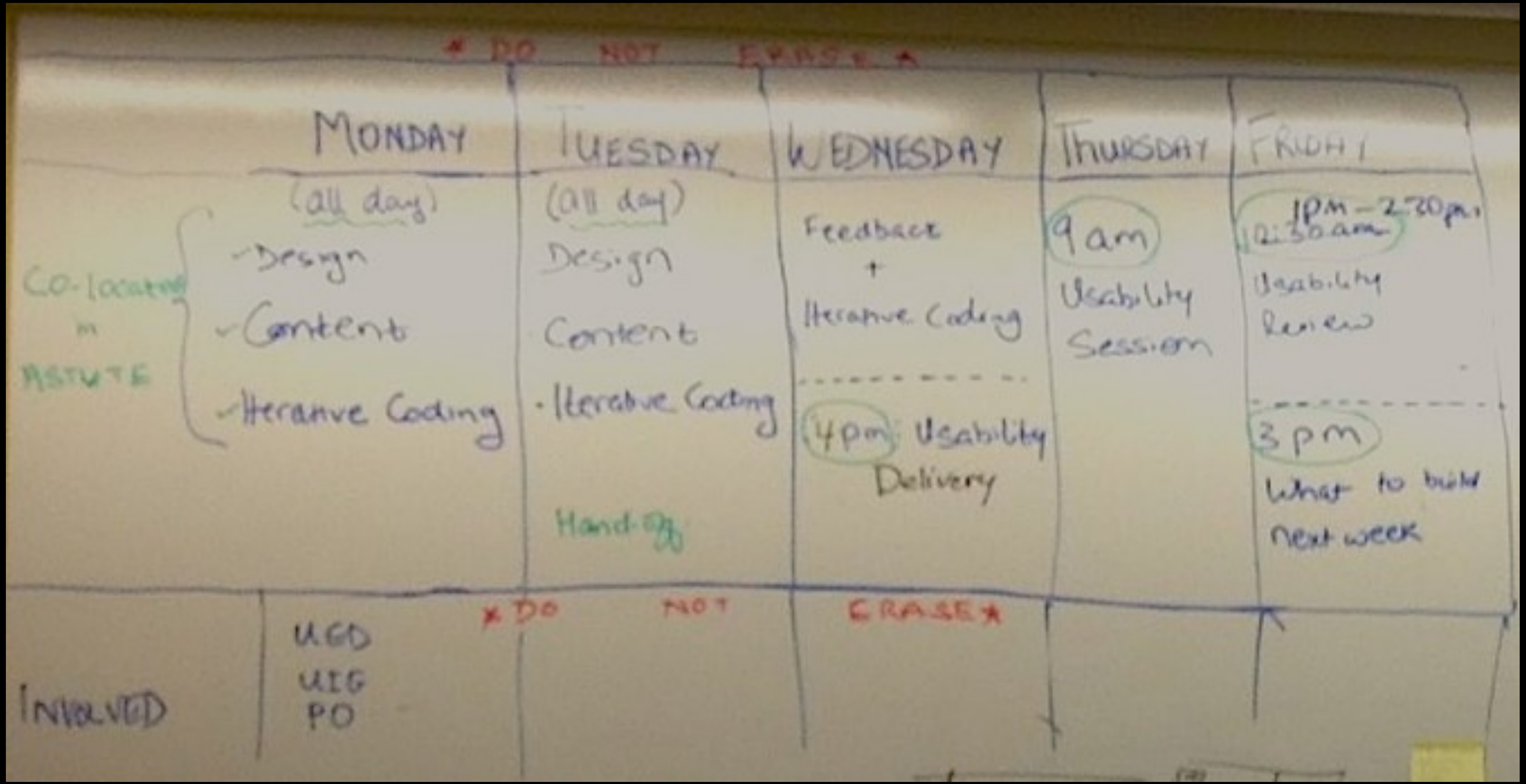
team working agreement

decide who is the decision maker

define your cadence

define how you will work together

define your hypotheses



* DO NOT ERASE *

	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
CO-locating in ASTUTE	(all day) -Design -Content -Iterative Coding	(all day) Design Content ·Iterative Coding Hand-off	Feedback + Iterative Coding / 4pm Usability Delivery	9am Usability Session	1PM – 2:30pm 12:30am Usability Review / 3pm What to build next week
INVOLVED	UGD UIG PO				

* DO NOT ERASE *

the tools

leveraging tools to be more effective

tools

sketching/whiteboard

paper prototyping

prototyping software

prototyping

patterns & visual language

sketching and whiteboarding

stop talking, start drawing!

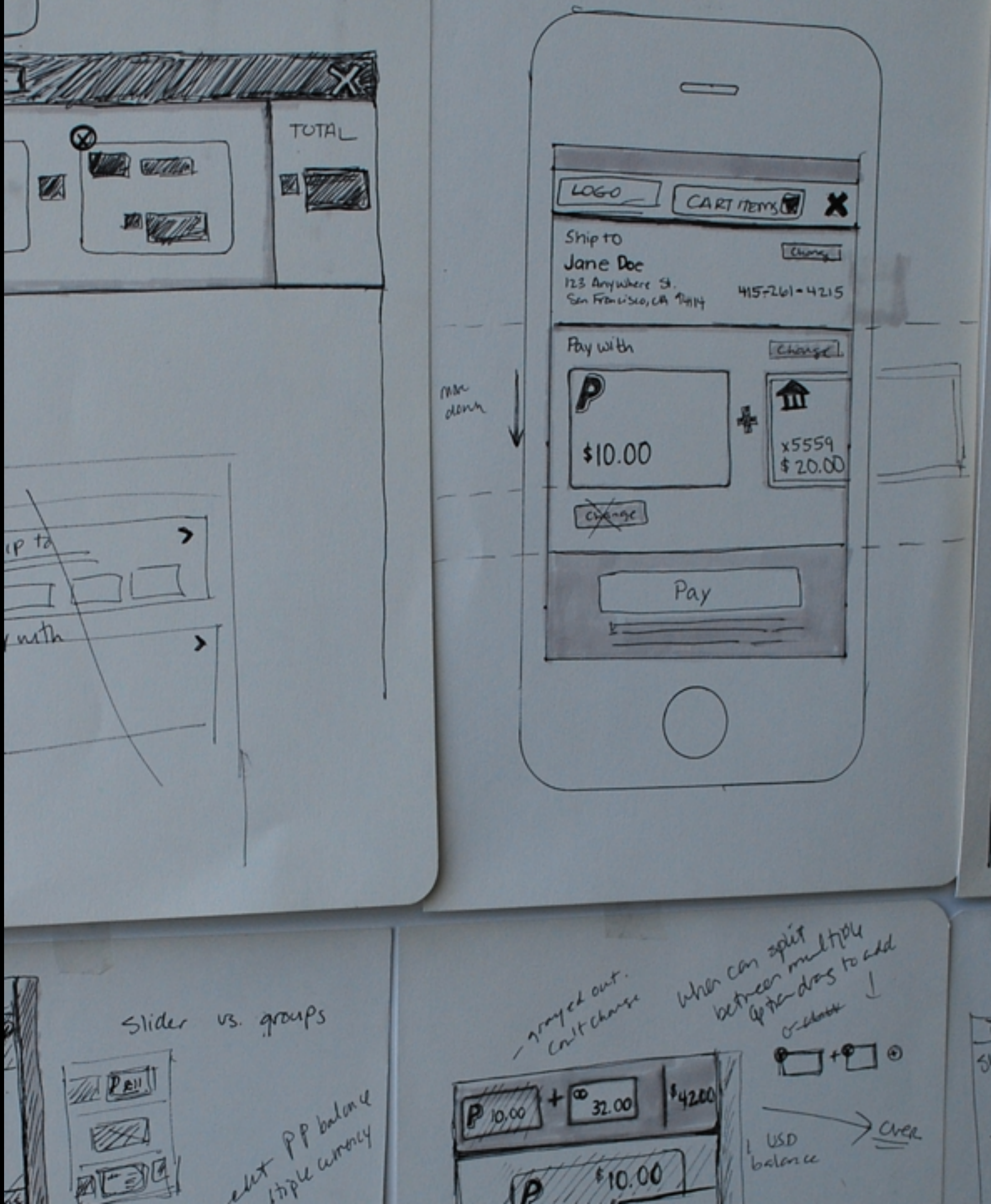
rapid ideation

throw away

validation

shared understanding

used as part of deliverable



paper prototyping

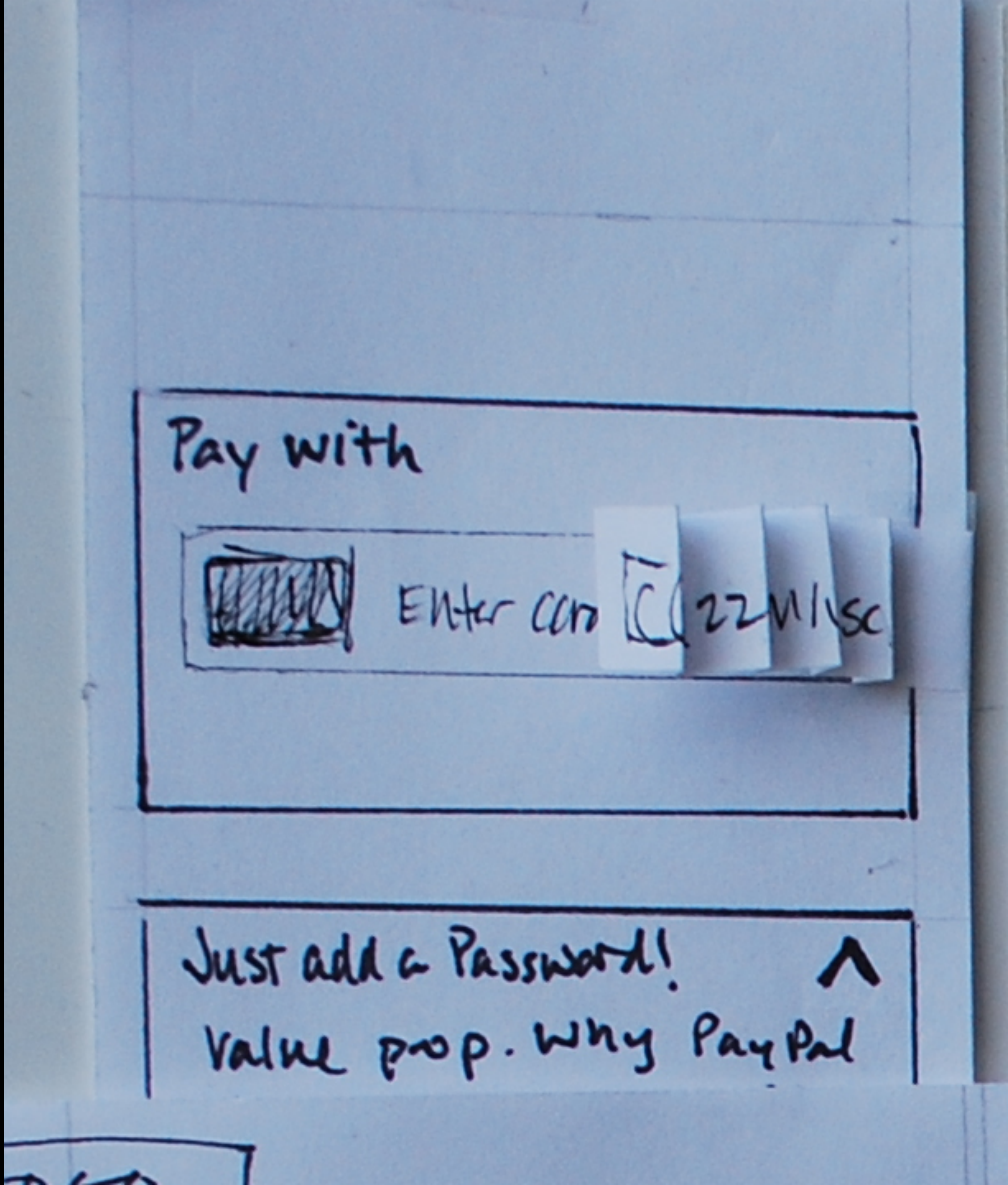
easy to go from paper to production

validate interactions

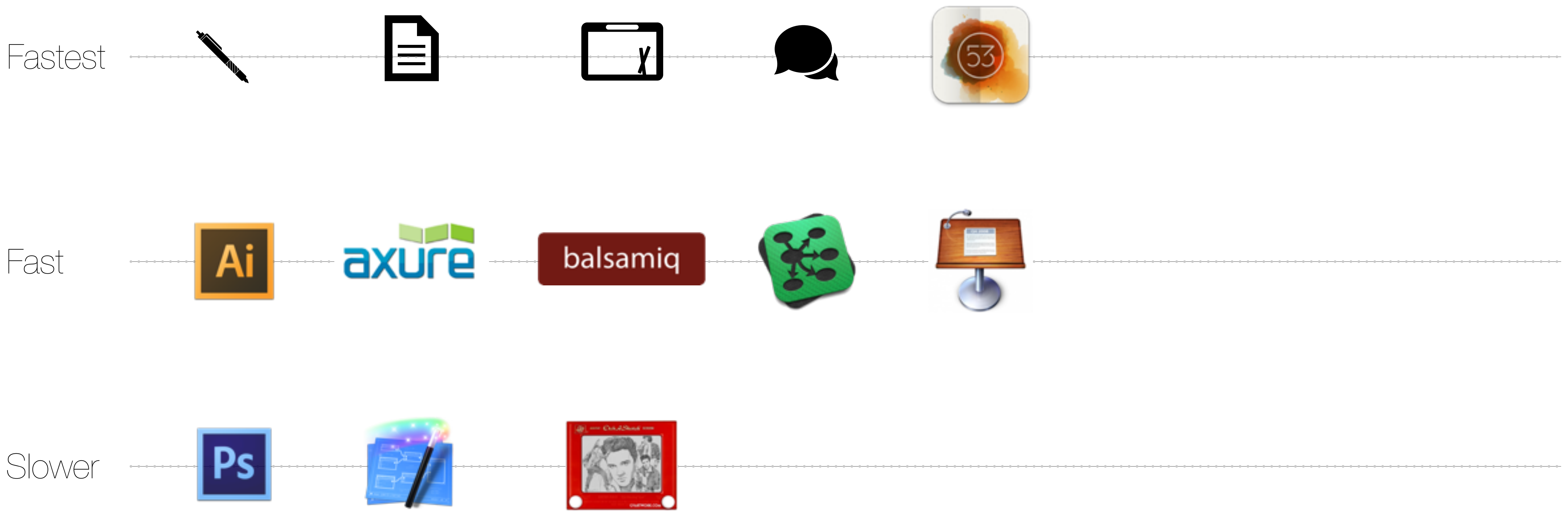
makes it easer and faster for developers to understand.

paints a clearer picture to business partners.

super fast



prototyping software (FIX)



prototyping tools

see:

- list of prototyping tools on my blog: http://bit.ly/SfWygk

few that we also use:

- Axure RP
- InVision
- POP

code prototypes vs tools?

- use the right tool at the right time
  - as you get closer to agile
- axure, proto.io, POP and a host of other prototyping tools are amazing -- especially early in the learning cycle
- code prototypes
  - important once you get close into the actual agile sprints
  - provide high fidelity to the user testing
  - faster cycle from “learning to live”

suggestions for code prototyping

bootstrap is one of the quickest to get going with

we use it on our production stack as well

jetstrap allows you to drag and drop a bootstrap page to get a quick start

nodejs is really powerful for prototyping your full application (web, tablet, desktop)

Bootstrap, from Twitter

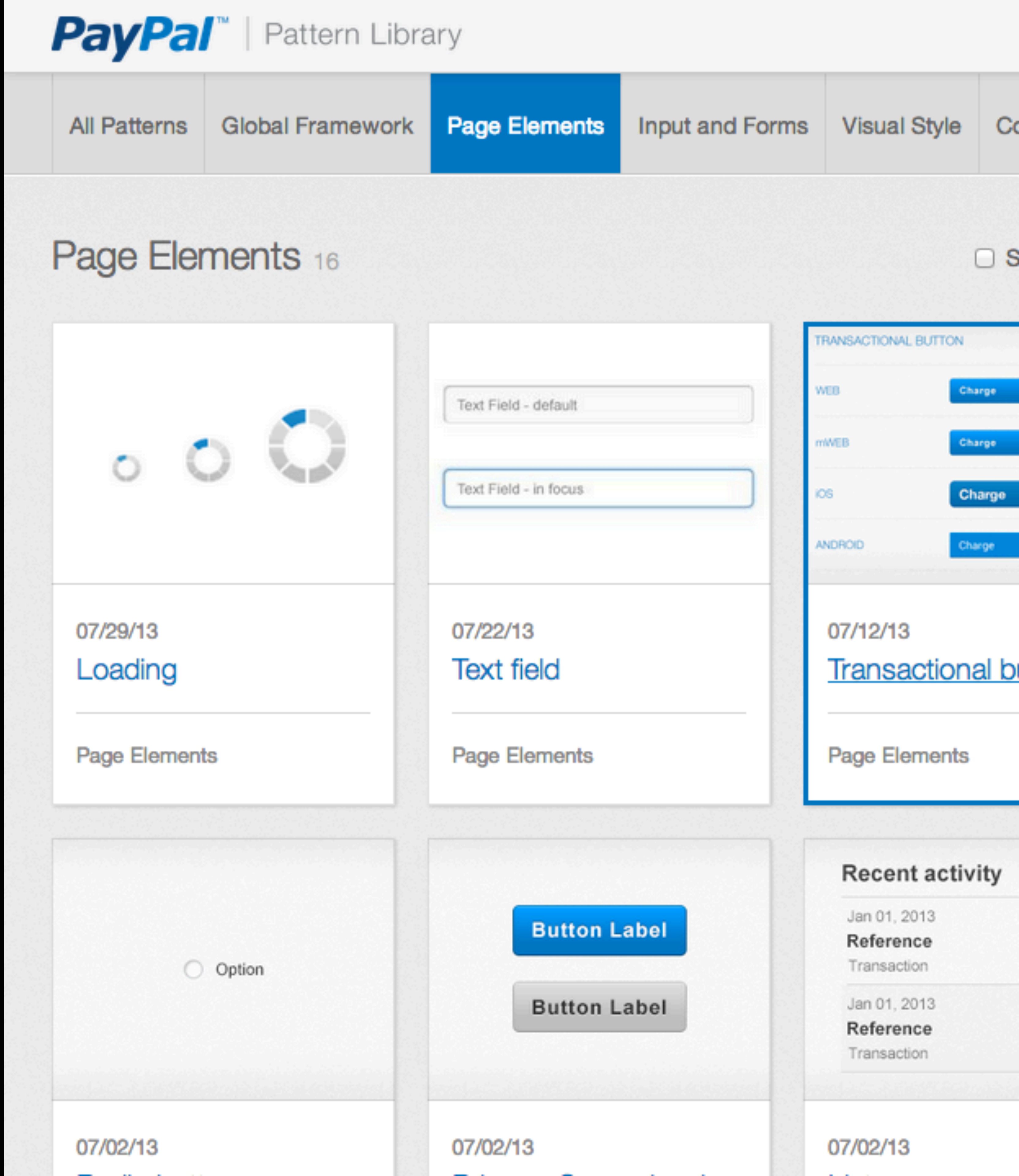
patterns & visual language

patterns enable rapid development

refine over time

ensure consistency

speed up design



picture credits

http://www.flickr.com/photos/wuschl2202/531914709/sizes/o/in/photostream/
http://www.flickr.com/photos/a_ninjamonkey/3565672226/sizes/z/in/photostream/
http://www.flickr.com/photos/funky64/4367871917/sizes/z/in/photostream/
http://www.flickr.com/photos/emdot/9938521/sizes/o/in/photostream/
http://www.flickr.com/photos/gregory_bastien/2565132371/sizes/z/in/photostream/
http://www.flickr.com/photos/trvr3307/3703648270/sizes/z/in/photostream/
http://www.flickr.com/photos/legofenris/5426012042/sizes/l/in/photostream/
http://www.flickr.com/photos/cleaneugene/6866436746/sizes/c/in/photostream/
http://www.flickr.com/photos/66309414@N04/6172219058/sizes/l/in/photostream/
http://www.flickr.com/photos/nicmcphee/2954167050/sizes/l/in/photostream/
http://www.flickr.com/photos/pasukaru76/6151366656/sizes/l/in/photostream/
http://www.flickr.com/photos/brianmitchell/2113553867/sizes/o/in/photostream/
http://www.flickr.com/photos/ciscel/422253425/sizes/z/in/photostream/
http://www.flickr.com/photos/zebble/6817861/sizes/l/in/photostream/
http://www.flickr.com/photos/nicasaurusrex/3069602246/sizes/l/in/photostream/
http://www.flickr.com/photos/nathangibbs/98592171/sizes/z/in/photostream/
http://www.flickr.com/photos/neilsingapore/4047105116/sizes/l/
http://www.flickr.com/photos/smb_flickr/439040132/
http://www.flickr.com/photos/therevsteve/3104267109/sizes/o/
http://www.flickr.com/photos/st3f4n/4193370268/sizes/l/
http://www.flickr.com/photos/eole/380316678/sizes/z/
http://www.flickr.com/photos/cobalt/3035453914/sizes/z/
http://www.flickr.com/photos/mbiskoping/6075387388/
http://www.flickr.com/photos/fragglerawker/2370316759/sizes/z/
http://www.flickr.com/photos/soldiersmediacenter/4685688778/sizes/z/

picture credits (continued)

http://www.flickr.com/photos/dahlstroms/4083220012/sizes/l/
http://www.flickr.com/photos/don2/53874580/sizes/z/
http://www.flickr.com/photos/hao_nguyen/3634552812/sizes/z/
http://www.flickr.com/photos/42573918@N00/8194636033/
http://www.flickr.com/photos/pagedooley/2420194539/sizes/z/
http://www.flickr.com/photos/neilsingapore/4047105116/sizes/l/
http://www.flickr.com/photos/smb_flickr/439040132/
http://www.flickr.com/photos/therevsteve/3104267109/sizes/o/
http://www.flickr.com/photos/st3f4n/4193370268/sizes/l/
http://www.flickr.com/photos/eole/380316678/sizes/z/
http://www.flickr.com/photos/cobalt/3035453914/sizes/z/
http://www.flickr.com/photos/mbiskoping/6075387388/
http://www.flickr.com/photos/fragglerawker/2370316759/sizes/z/
http://www.flickr.com/photos/soldiersmediacenter/4685688778/sizes/z/
http://www.flickr.com/photos/janed42/5033842895/sizes/z/
http://www.flickr.com/photos/9619972@N08/1350940605/
http://www.flickr.com/photos/alanenglish/483251259/sizes/z/

thanks flickr!