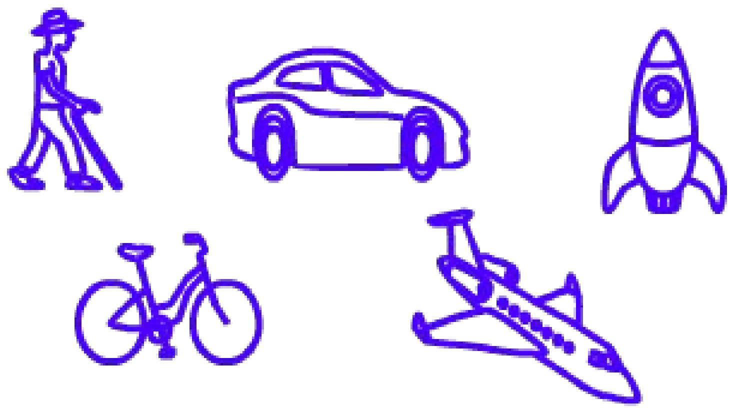
What does "speed" mean in software product delivery?



Jason Yip
Agile Coach, Spotify NYC
@jchyip
jyip@spotify.com
https://medium.com/@jchyip

"Speed" in software product delivery

“Speed” in software product delivery

What feels fast

(the human experience of speed)

?

"Speed" in software product delivery

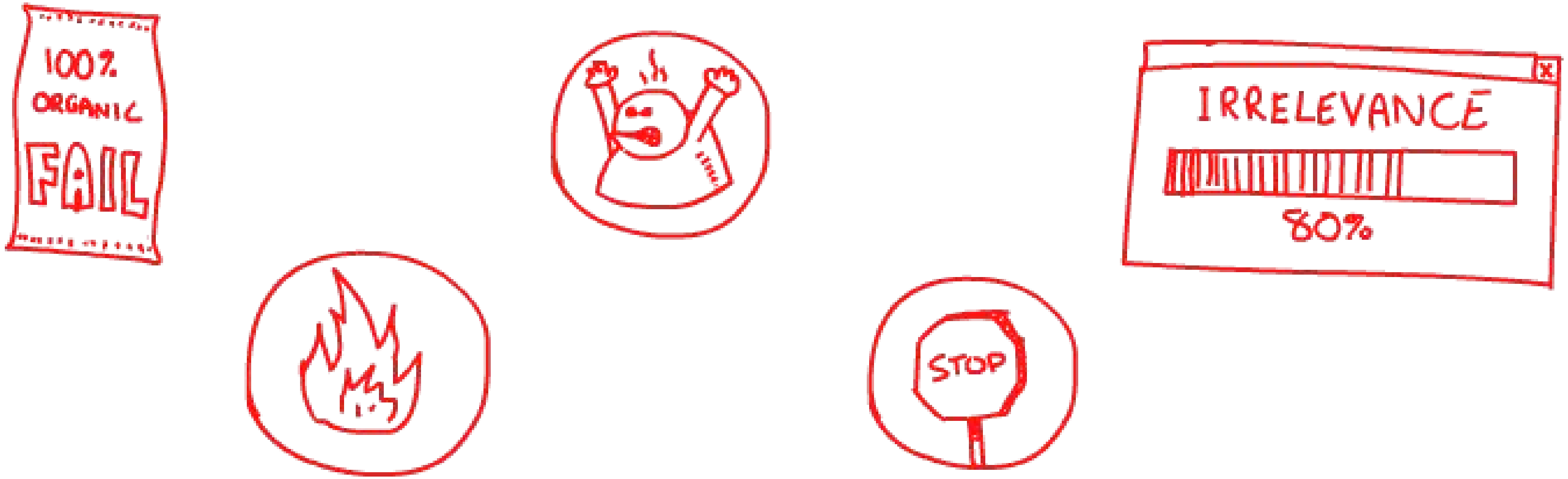
What feels fast

What is actually fast

(in terms of measurable outcomes)

Feeling fast is about removing friction.

Friction is when it feels like the work is fighting you.



"Make everything better."

"What does that mean?"

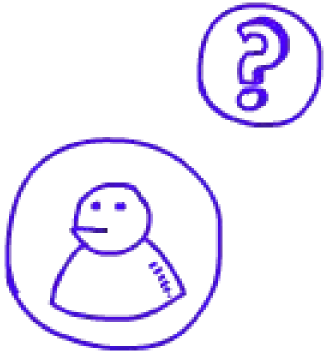
"Work it out."

"Are there any constraints?"

"Work it out."

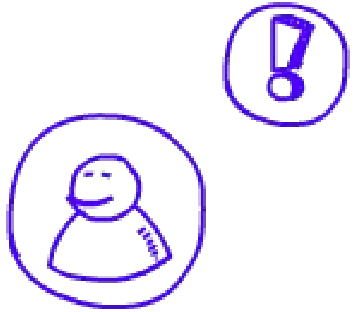
"What are the main things to do?"

It feels slow when goals are unclear.



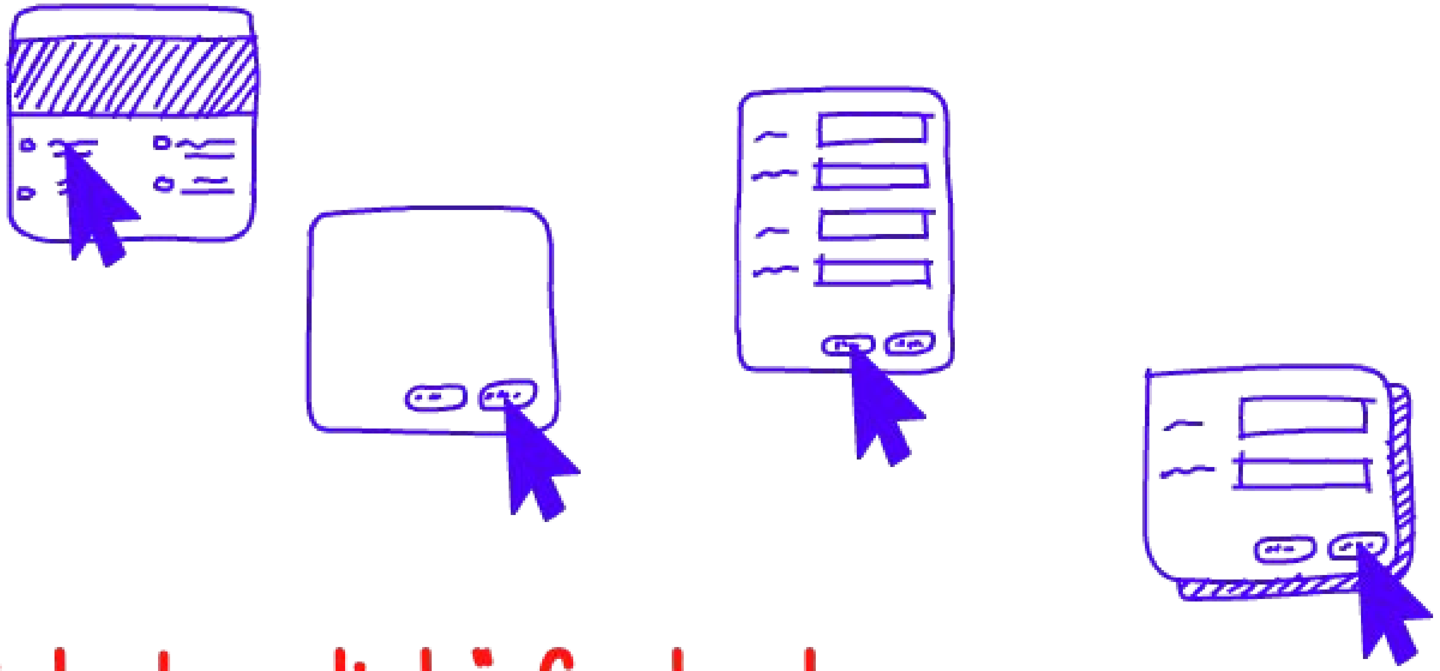
"Remove friction from day-to-day work in order to improve work experience."	Intent and rationale
"Stay within in-team activities."	Constraints
"Main opportunities are probably within: 1. How we describe work ..."	Main implied tasks

It feels fast when goals are clear.

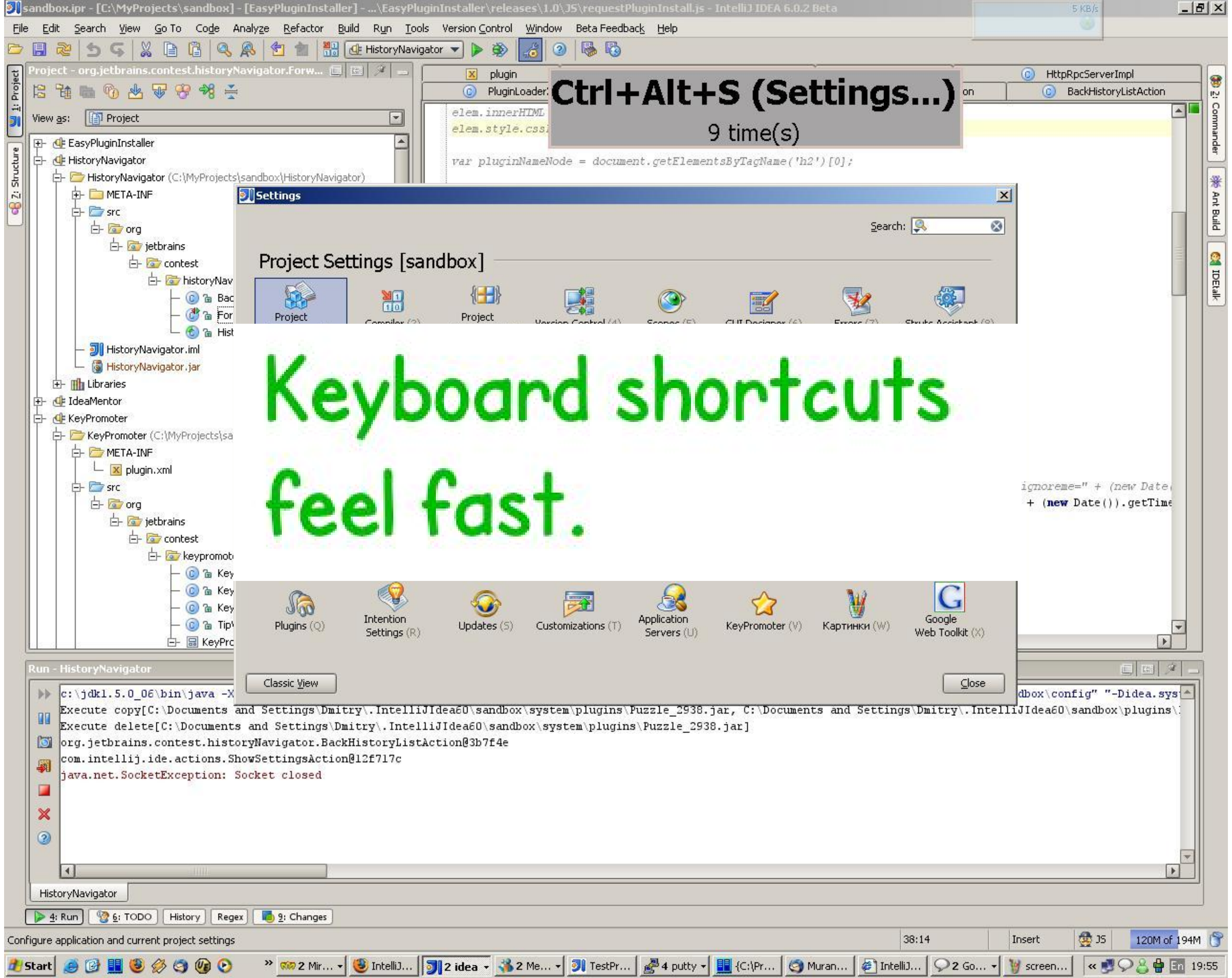


Email tag feels slow.

Just turning around to ask feels fast.



"Clickety-click" feels slow.



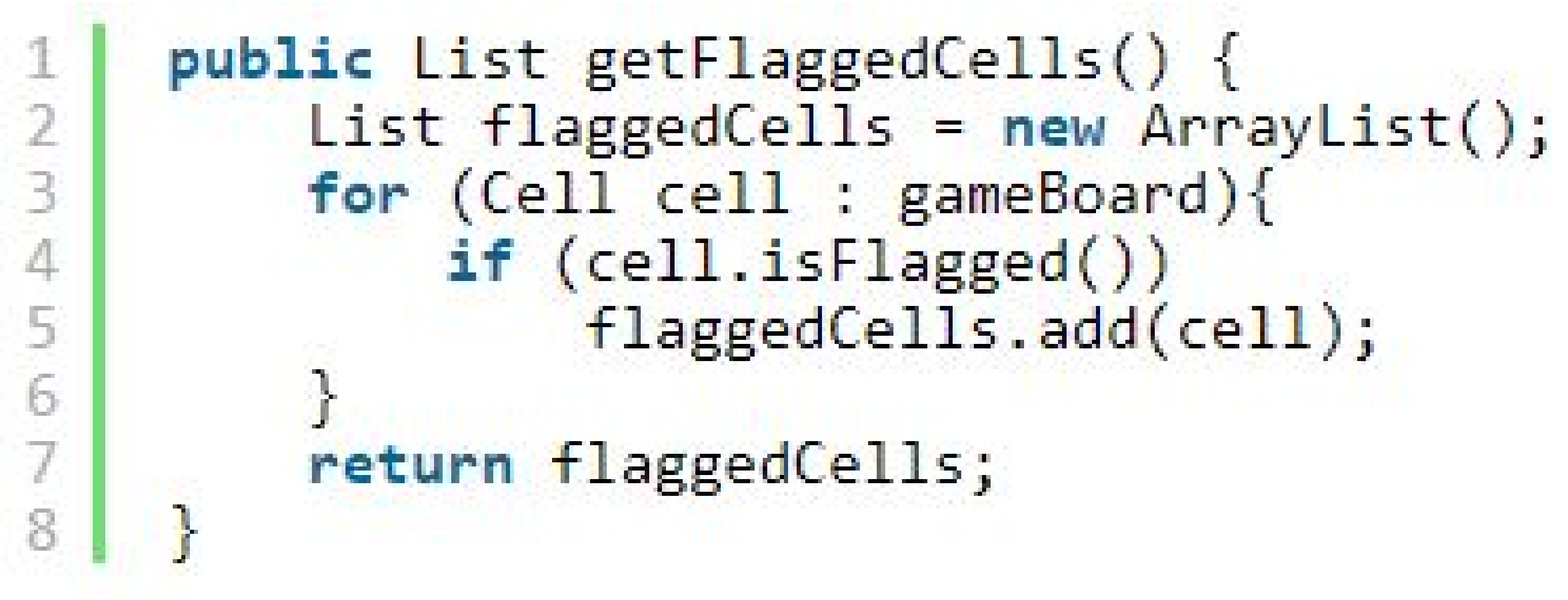
```
c:\jdk1.5.0_06\bin\java -X
Execute copy[C:\Documents and Settings\Dmitry\.IntelliJIdea60\sandbox\system\plugins\Puzzle_2938.jar, C:\Documents and Settings\Dmitry\.IntelliJIdea60\sandbox\plugins\
Execute delete[C:\Documents and Settings\Dmitry\.IntelliJIdea60\sandbox\system\plugins\Puzzle_2938.jar]
org.jetbrains.contest.historyNavigator.BackHistoryListAction@3b7f4e
com.intellij.ide.actions.ShowSettingsAction@12f717c
java.net.SocketException: Socket closed
```

Keyboard shortcuts feel fast.

```c
                             O S[]="syntax_error!"
                        "M@K~|JOEF\\^~_NHI]"; L*N,*K,*
                   B,*E,*T,*A,*x,D; Q(*k)(),v; V z(P),j,_,*
                 o,b,f,u,s,c,a,t,e,d; J l; Q _k(P){ R*K?*K++:
               ~-d; } V r(L a){ R a&&putchar(a); } L n(){ R*T=j=
             k(),++j; } O G(P){ *o=d,longjmp(l,b); } Z g(Y a){ R a
          >>s|(a&~-e)<<s;} C p(L*T) { W(r(*T++)); *--T-c&&r(c);} L m
        (P){ W(!((v=A[*T++])-f)); R v; } P q(L**N) { O*q; b=!b; f=-~b;
       u=f|b; s=b<<u; c=s|f;                         a=s<<f; t=-~u; e=a
     <<u; D=v=u<<t;q=S                                    +c-~t; q[~s]=a;
    q--[f]+=a;q--[c                                           ]+=a; B=(L*)
    N+~e*e;x=B+e                                                 ; A=B+e/f;
  o=(V*)(x+a                                                          ); A[-
  ~s]=f; T                                                             =K =A-
 a*f;A[*--                                                              q]=c+
 c; *q=!                                                                 c; W
(++j&&*                                                                   ++q)
A[*q-a                                                                    ]=j;
W(v<D                                                                      +c)
x[v-                                                                       D]=
v,A[                                                                       v++
]=j;                                                                       ++j
 ,v=                                                                      D=e/
  t;                                                                     W(++
   v<=                                                                  D+f*
    u)x                                                              [v-D+~
    -c]=                                                            v|a,A[
      v]=A                                                        [v|a]=j;
        W((                                                    A[v]=A[v|a
          ]=j                                                  ,++v<a*u+~t));
             for(                                              ; E=*++N;T[~d]
                =a)W                                           (*T++=*E++);k=
                  T
```

Confusing code feels slow.

http://www.ioccc.org/2015/burton/prog.c

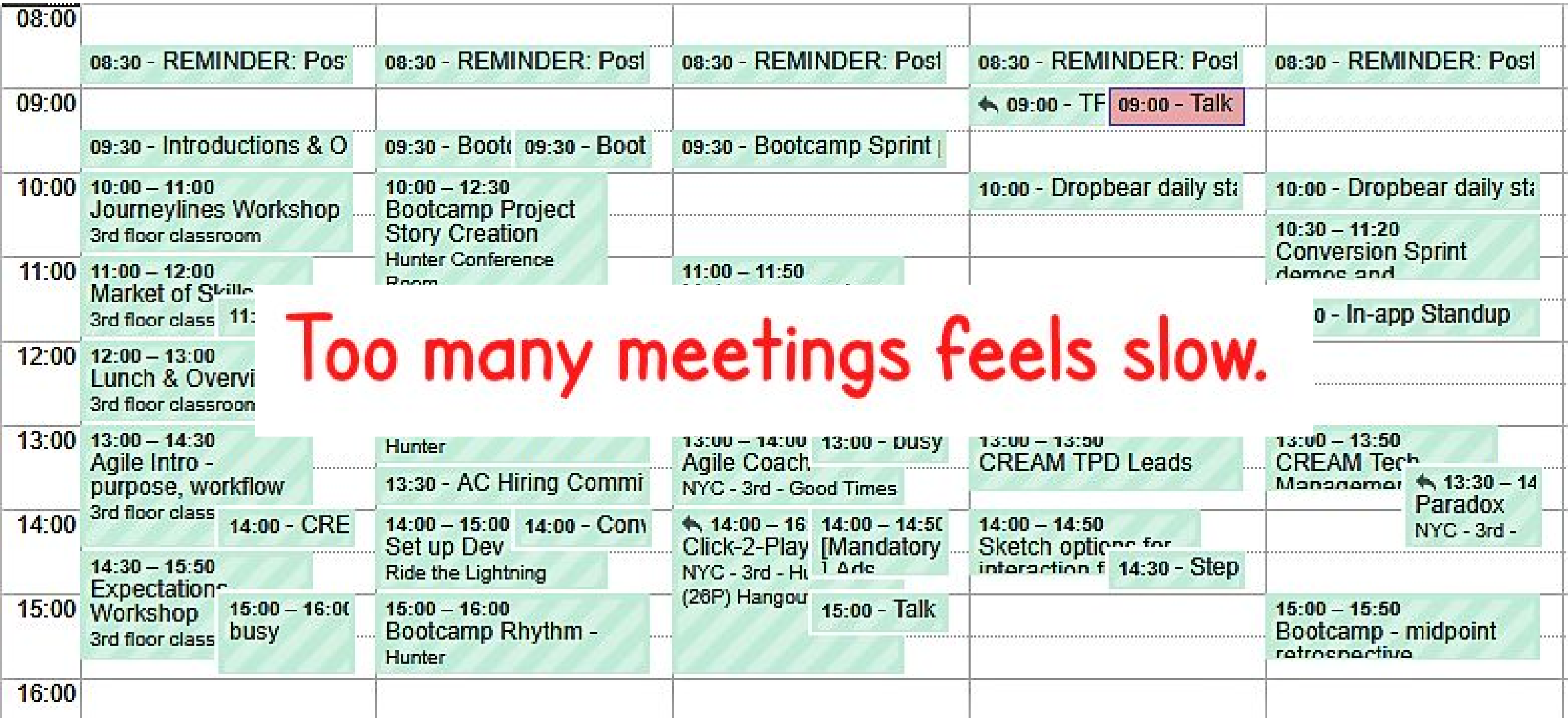


```
public List getFlaggedCells() {
	List flaggedCells = new ArrayList();
	for (Cell cell : gameBoard){
		if (cell.isFlagged())
			flaggedCells.add(cell);
	}
	return flaggedCells;
}
```

Clean code feels fast.

http://www.itiseezee.com/?p=83

Too many meetings feels slow.



08:00
09:00
10:00
11:00
12:00
13:00
14:00
15:00
16:00

08:30 - REMINDER: Pos
09:30 - Introductions & O
10:00 – 11:00 Journeylines Workshop 3rd floor classroom
11:00 – 12:00 Market of Skills 3rd floor class
11:
12:00 – 13:00 Lunch & Overvi 3rd floor classroom
13:00 – 14:30 Agile Intro - purpose, workflow 3rd floor class
14:00 - CRE
14:30 – 15:50 Expectations Workshop 3rd floor class
15:00 – 16:0(busy

08:30 - REMINDER: Post
09:30 - Boot(
09:30 - Boot
10:00 – 12:30 Bootcamp Project Story Creation Hunter Conference Room
Hunter
13:30 - AC Hiring Commi
14:00 – 15:00 Set up Dev Ride the Lightning
14:00 - Conv
15:00 – 16:00 Bootcamp Rhythm - Hunter

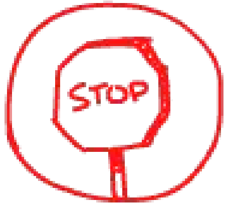
08:30 - REMINDER: Post
09:30 - Bootcamp Sprint
11:00 – 11:50
13:00 – 14:00 Agile Coach NYC - 3rd - Good Times
13:00 - busy
14:00 – 16 Click-2-Play NYC - 3rd - Hu (26P) Hangou
14:00 – 14:5([Mandatory] Ads
15:00 - Talk

08:30 - REMINDER: Post
09:00 - TF
09:00 - Talk
10:00 - Dropbear daily sta
13:00 – 13:50 CREAM TPD Leads
14:00 – 14:50 Sketch options for interaction f
14:30 - Step

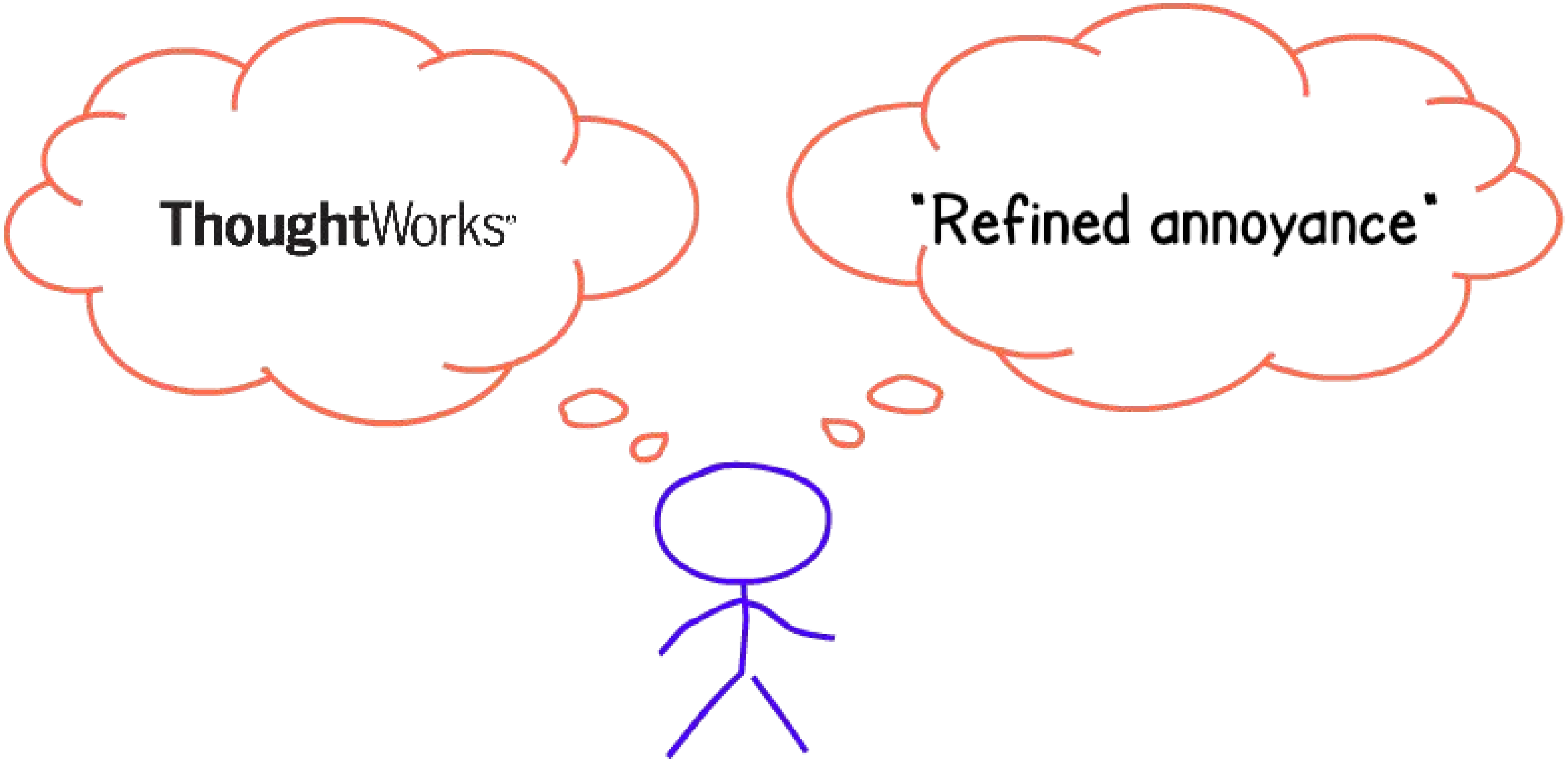
08:30 - REMINDER: Post
10:00 - Dropbear daily sta
10:30 – 11:20 Conversion Sprint demos and
0 - In-app Standup
13:00 – 13:50 CREAM Tech Managemer
13:30 – 14 Paradox NYC - 3rd -
15:00 – 15:50 Bootcamp - midpoint retrospective

Meeting-free days feel fast.

Friction is when it feels like the work is fighting you.



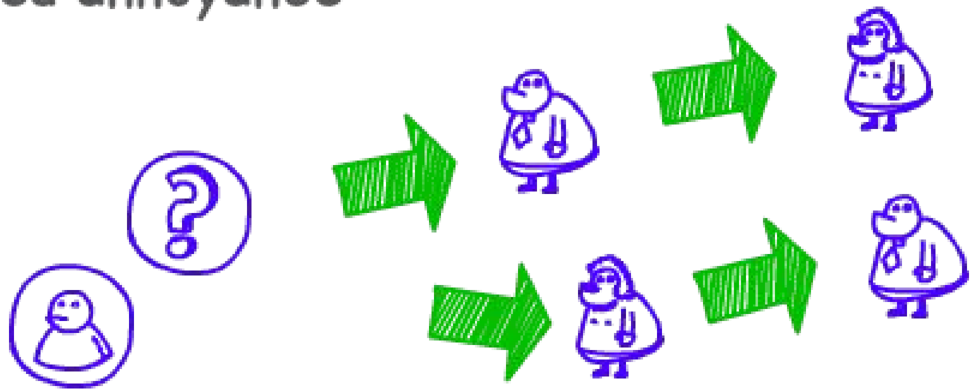
Feeling fast is about removing friction.


ThoughtWorks®
"Refined annoyance"

"Refined annoyance":

Refusal to accept friction of any kind

Refined annoyance

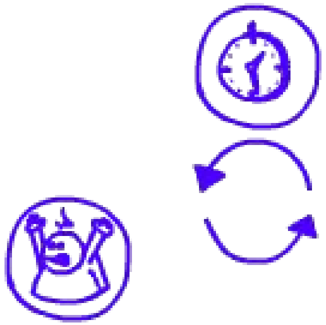


If you don't know
what to work on next

chase people, explore yourself,
whatever to find out... immediately

wait for someone to tell you

Refined annoyance

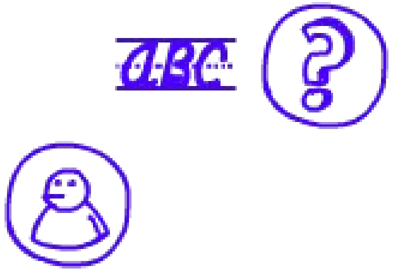


If anything takes too long / is too repetitive

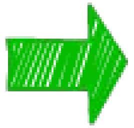
automate, build tools, whatever is necessary

~~just grin and bear it~~

Refined annoyance



"The names we're using are slightly confusing."



Clear naming is a high priority issue. Find better names.

"That's too trivial to worry about."

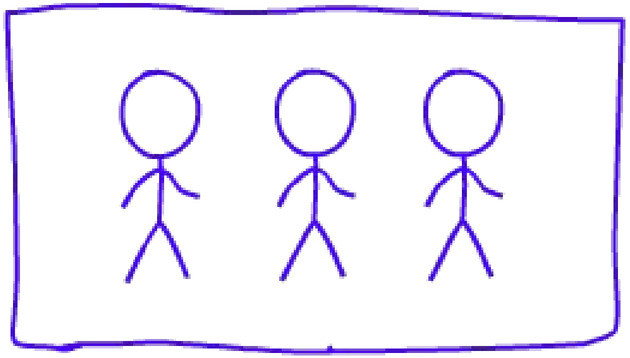
Friction goes away primarily due to a systematic habit of "refined annoyance"

My claim

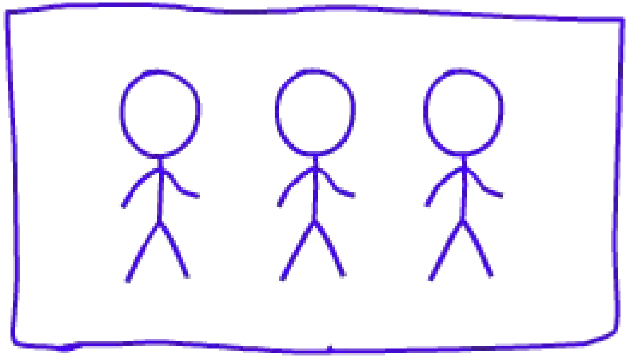
Friction goes away primarily due to a systematic habit of "refined annoyance"

My claim

How does one cultivate this?



Separate "developer productivity" team



Separate "developer productivity" team

✓

Deliver useful tools and platforms

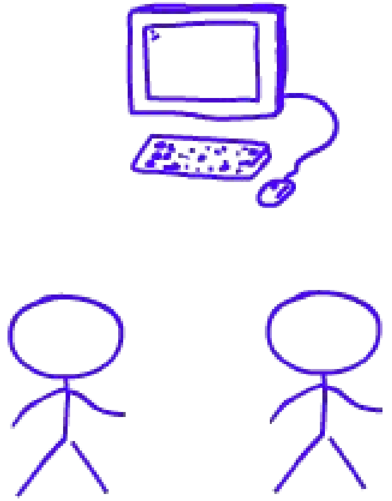
✗

Cultivate habits (typically not)

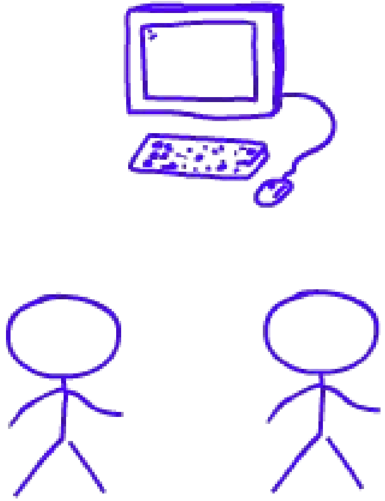
The only tactics
I've seen work...

Filter for "refined annoyance" when hiring

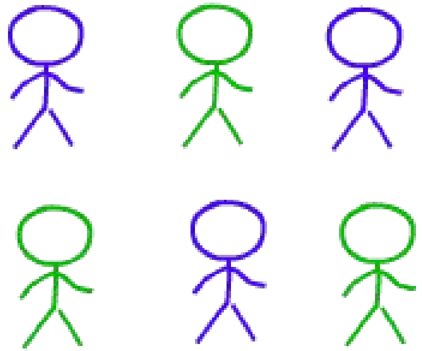
"Could you share an example of...?"



Pairing, embedding, role modeling

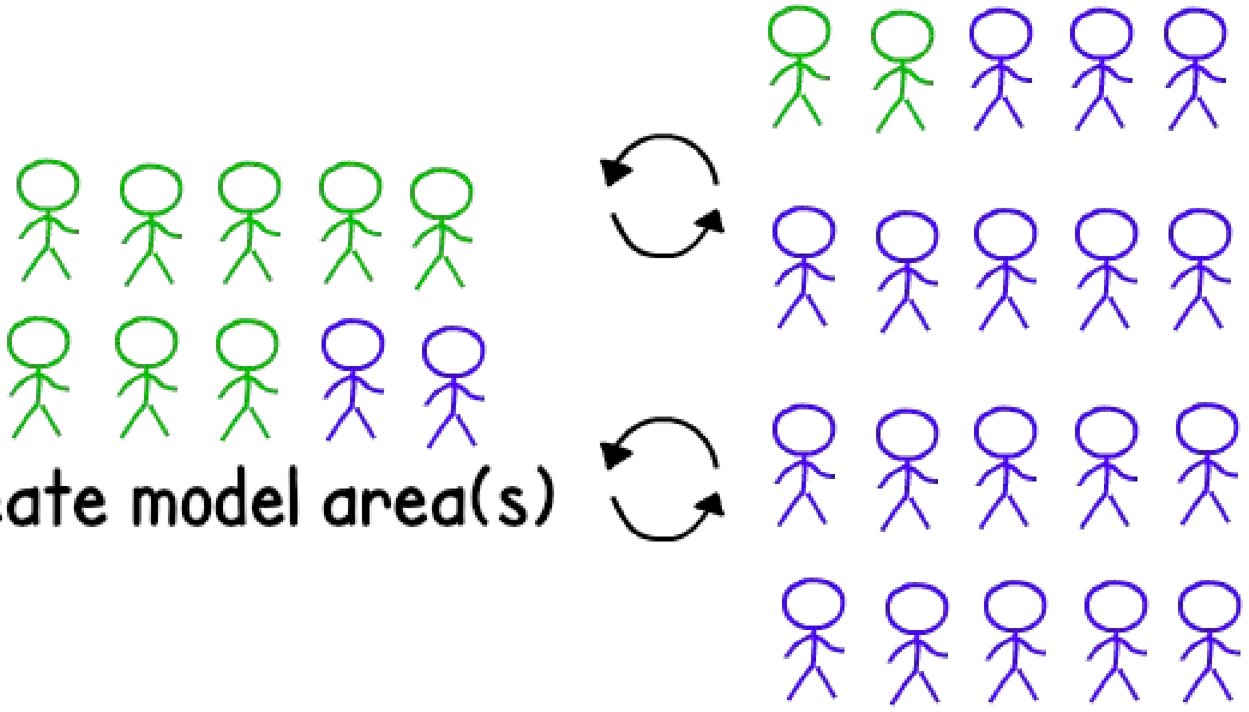


Role model team / area



Usually close to 50%

Incubator



Create model area(s)

Rotate people through

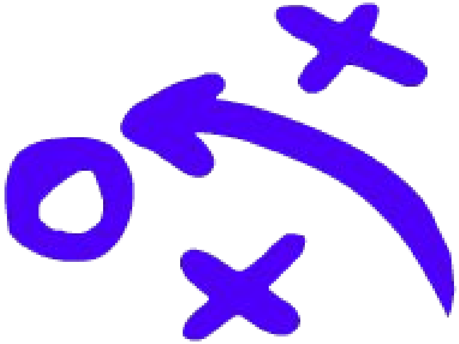
"Speed" in software product delivery

What feels fast

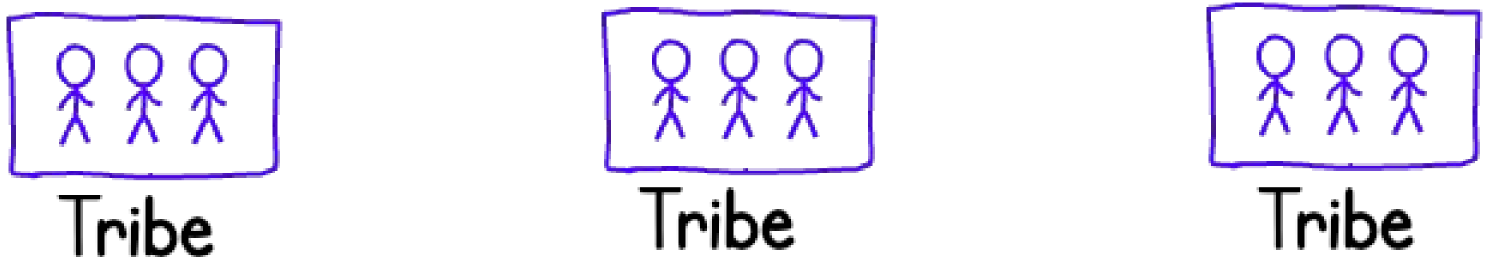
What is actually fast

(in terms of measurable outcomes)

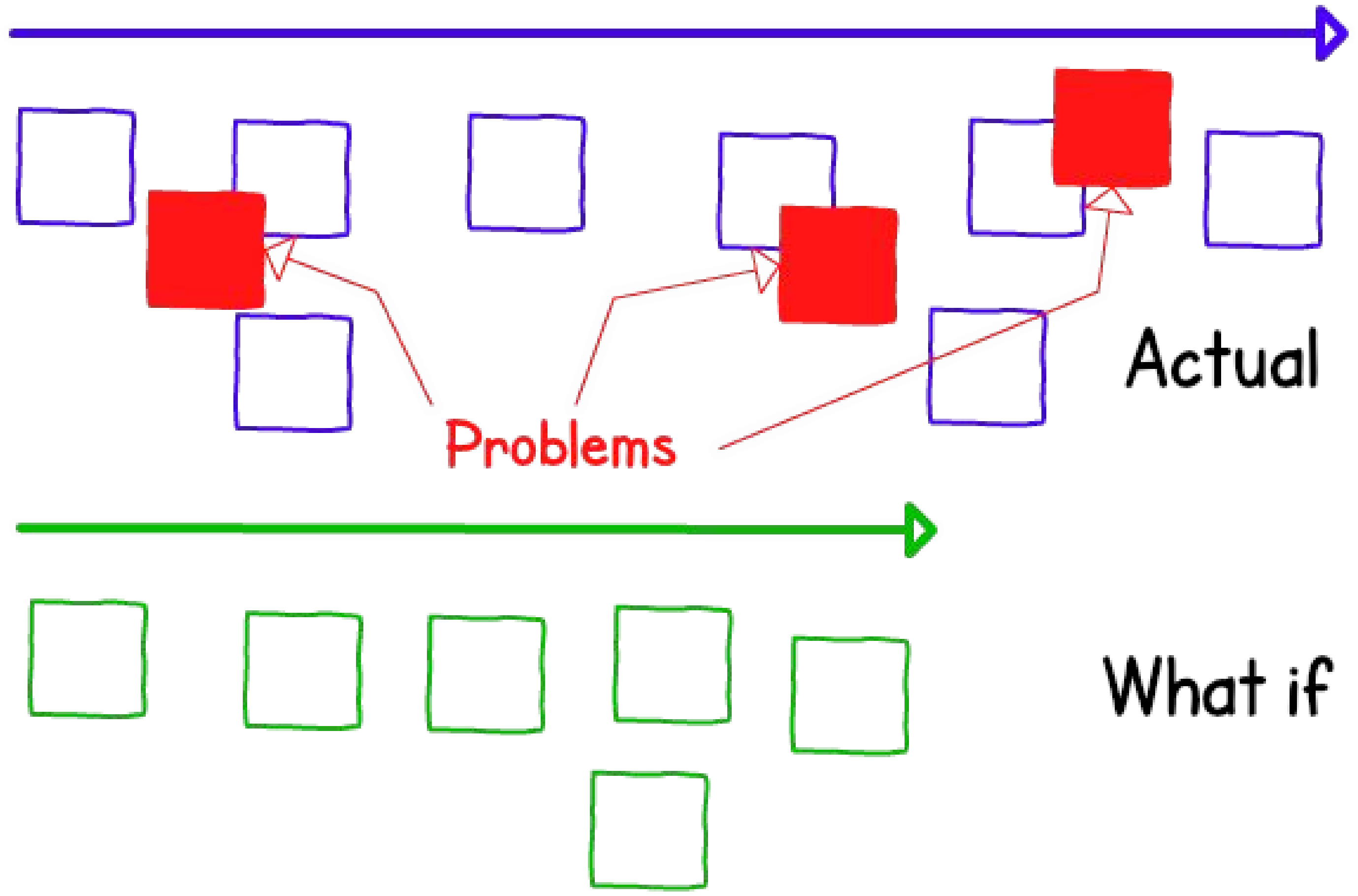
Actually delivering fast is about designing how you deliver



How might we improve the speed of discovery and delivery at Spotify?



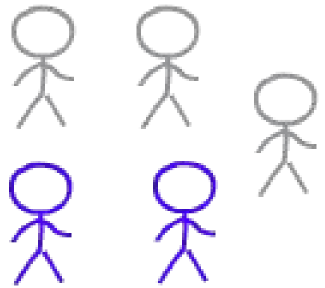
Nominations of deliveries that were particularly fast or slow (mostly the latter)



Key point: focus on impact to calendar time

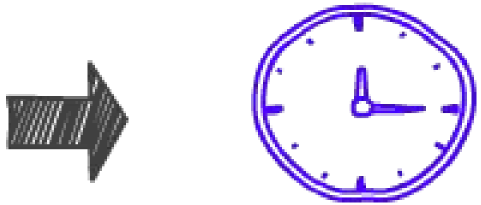
1. "Zombie projects"
2. Delivery accountability / decision making
3. Large launches

"Zombie projects"



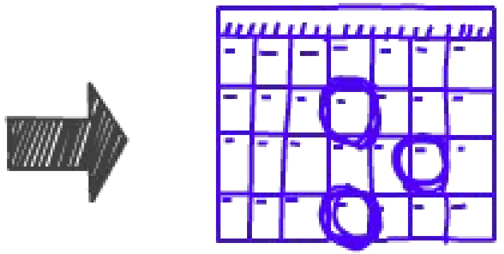
Insufficient staffing

Not really alive
(no real progress)

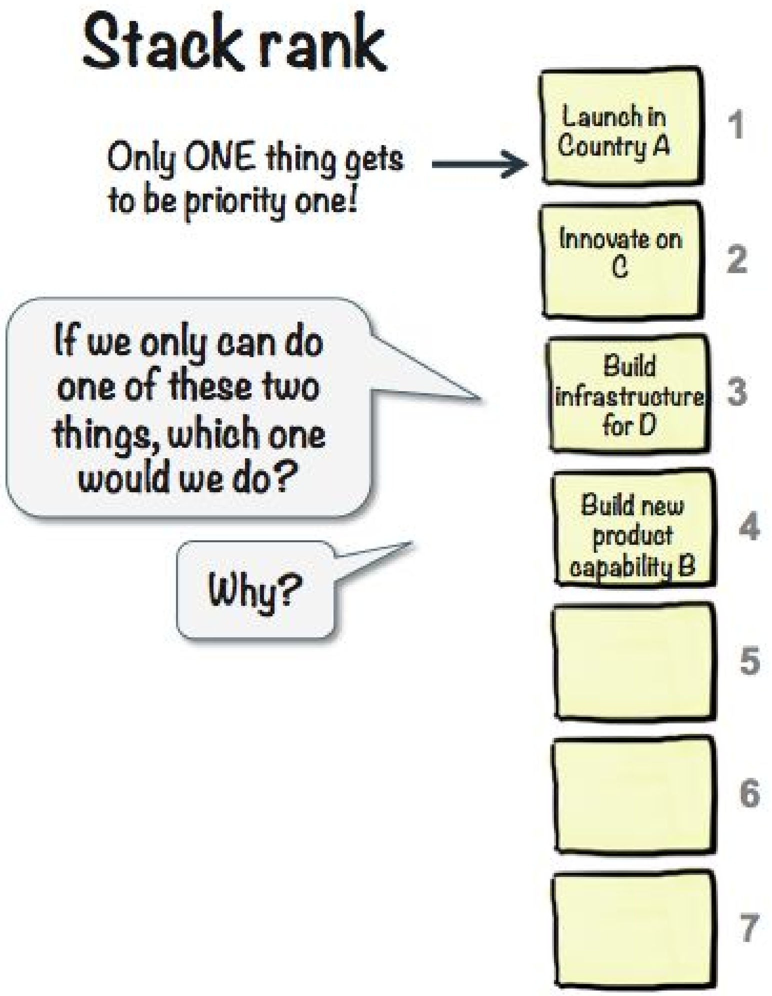


Time filling activities

Not really dead
(unavailable for other work)



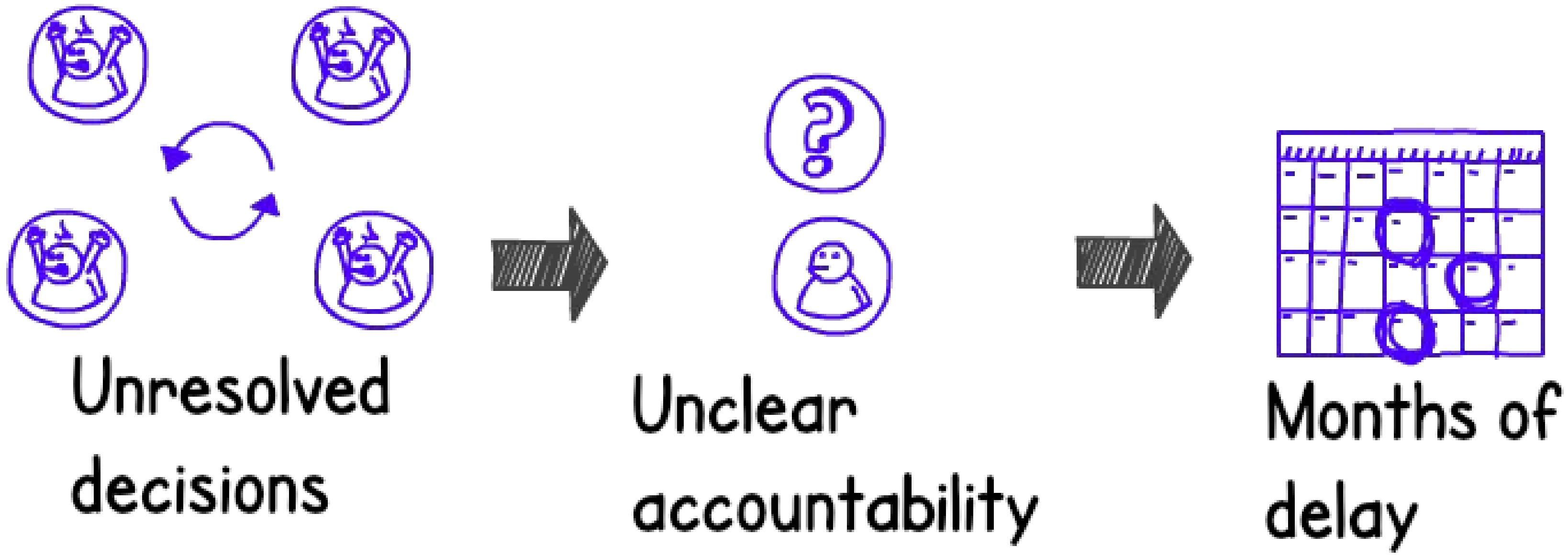
Months of delay


Stack rank
Only ONE thing gets to be priority one!
Launch in Country A
1
Innovate on C
2
Build infrastructure for D
3
Build new product capability B
4
5
6
7
If we only can do one of these two things, which one would we do?
Why?

North Star goals
Company Beliefs
X active users
World class employer

https://blog.crisp.se/2016/06/08/henrikkniberg/spotify-rhythm

"Delivery accountability"



Technology

Delivery of goals / milestones

Tech health

Product

Mission / Strategy / Metrics / Goals

Org health

Design

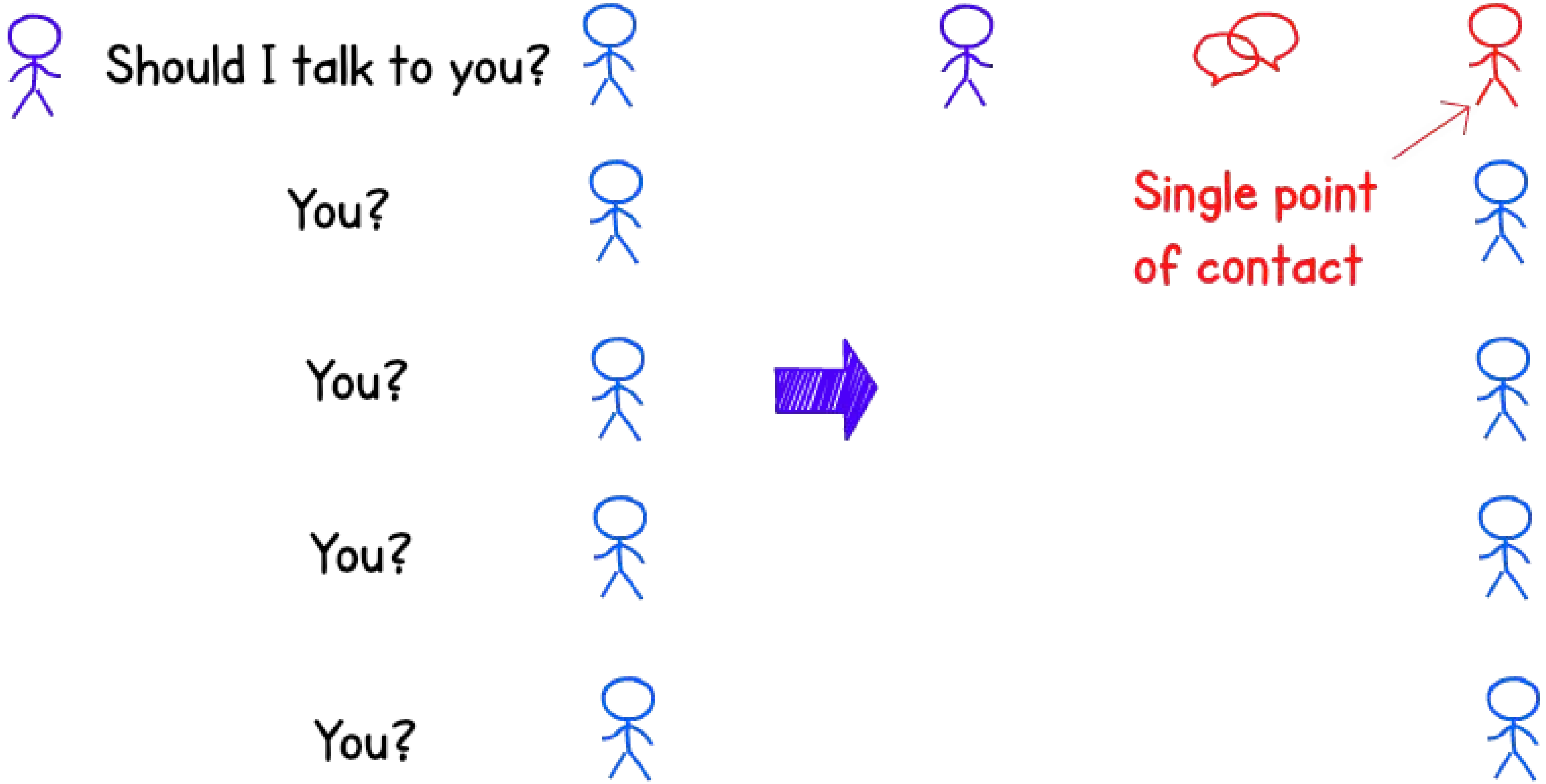
Holistic user experience

Technical Owner

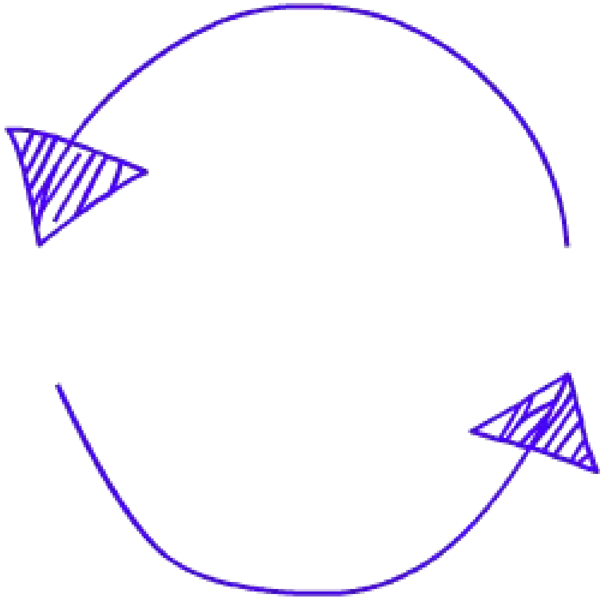
Primary point of contact for Squad delivery

Product Owner

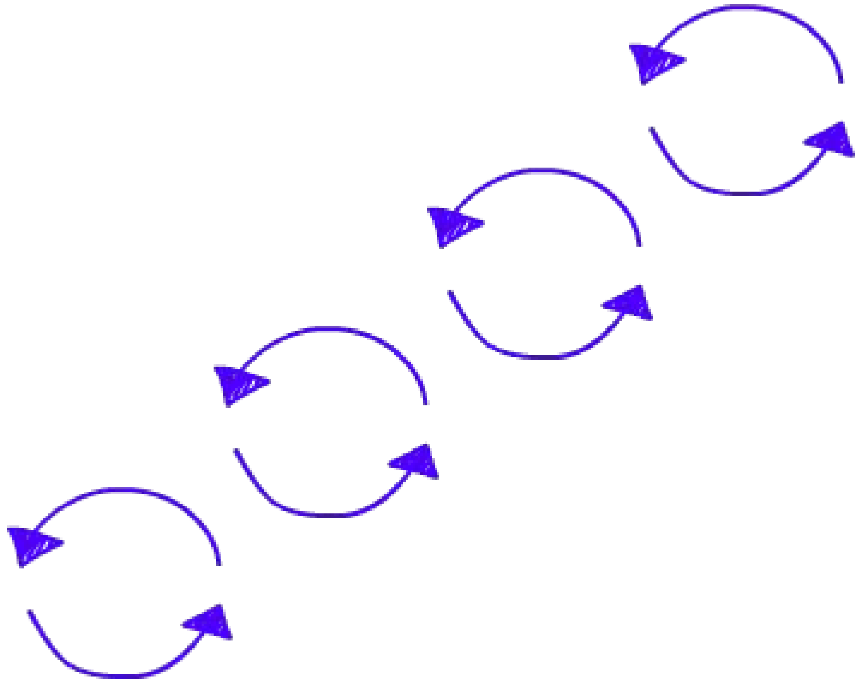
Designer


Should I talk to you?
You?
You?
You?
You?
Single point of contact

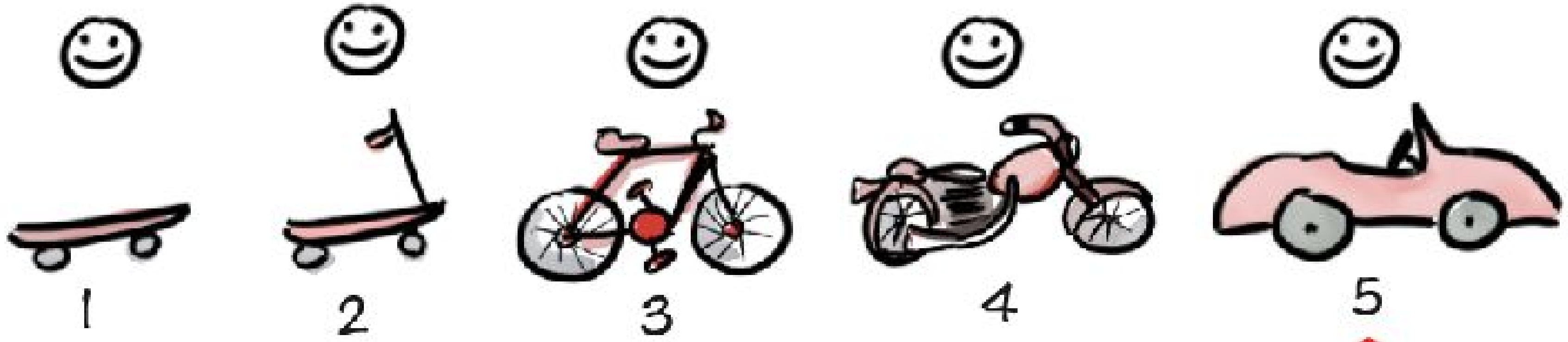
The allure of large changes



vs



Inevitable questions about iterative-incremental delivery

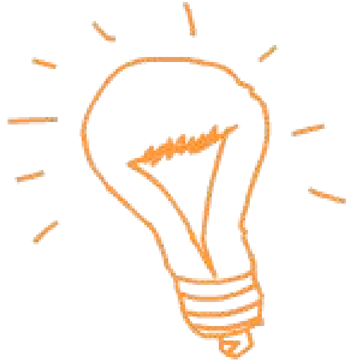


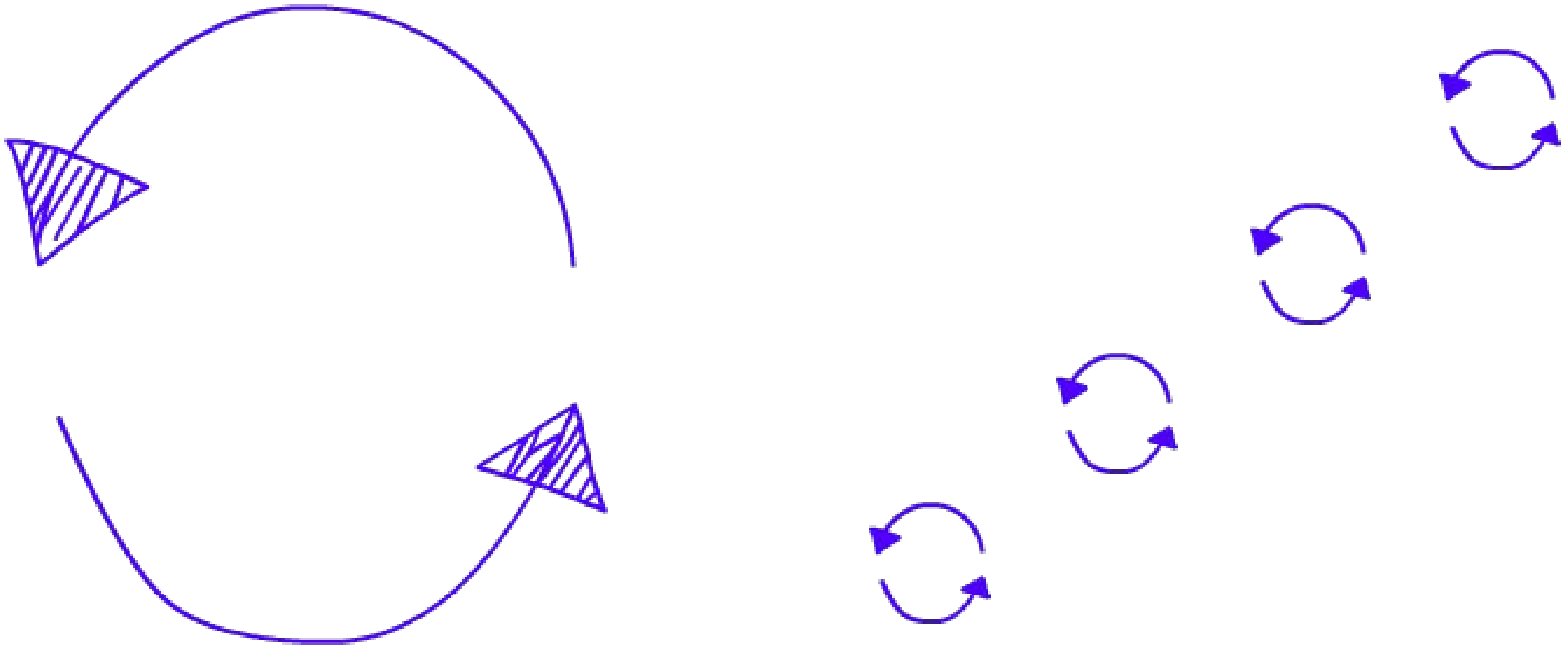
Do we actually learn anything from this?

Why aren't we more amibitious?

Do we really need so many iterations?

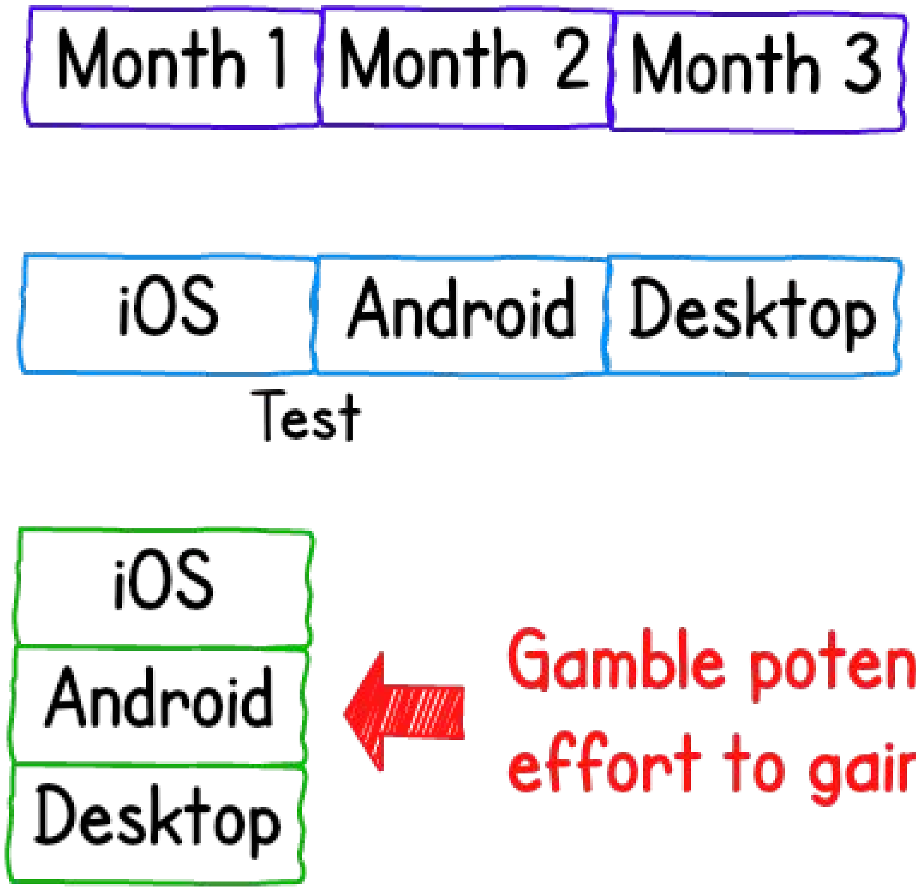
The more you know, the easier it is to assume you know more than you know





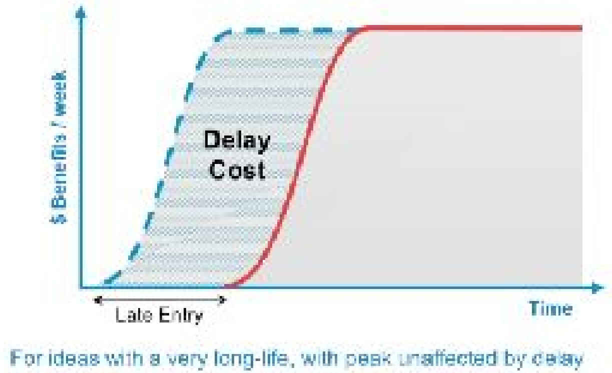
Measure the difference?

1. Parallel efforts
2. Conquer and divide



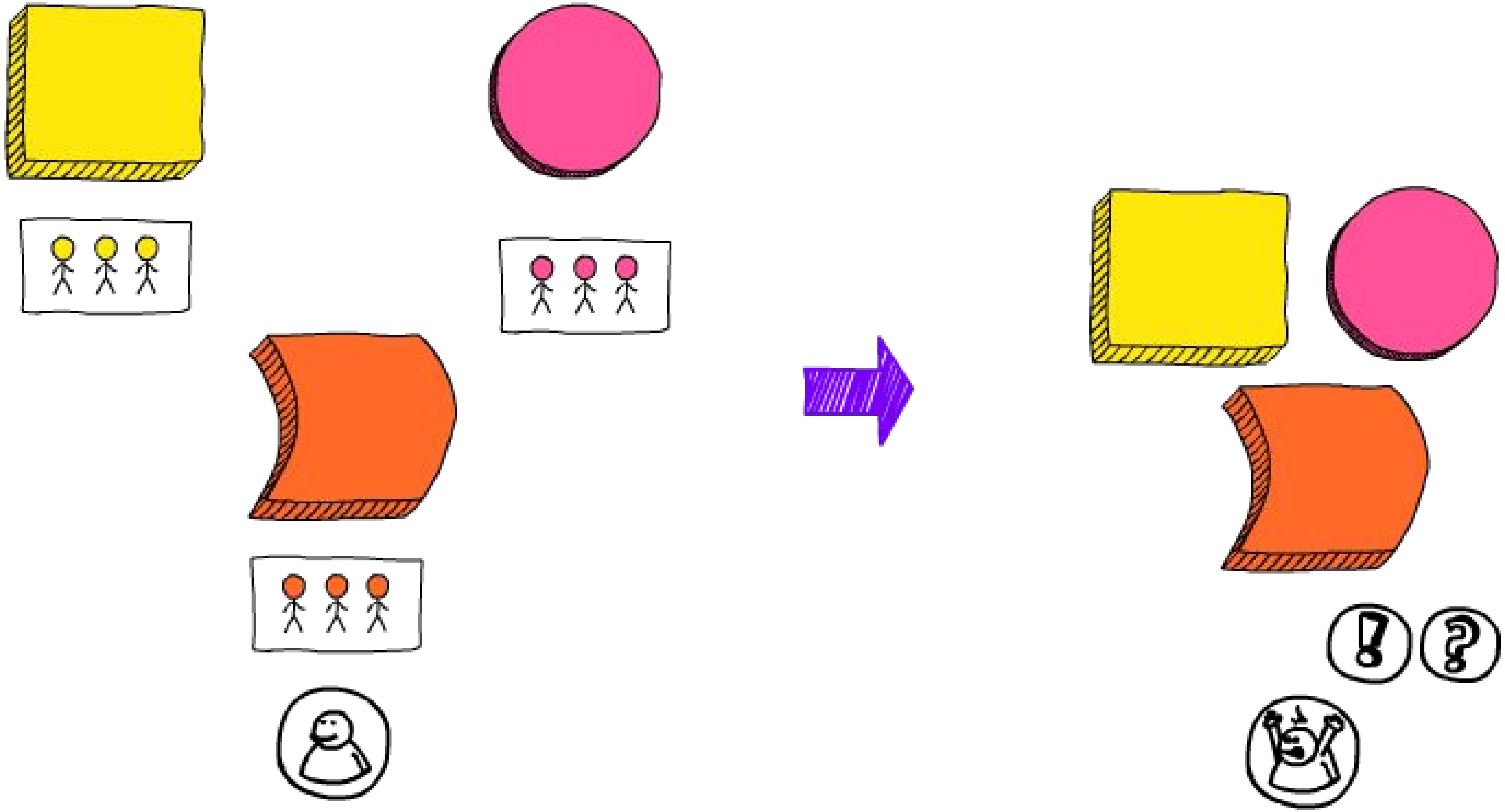
Gamble potentially wasted effort to gain calendar time

Education

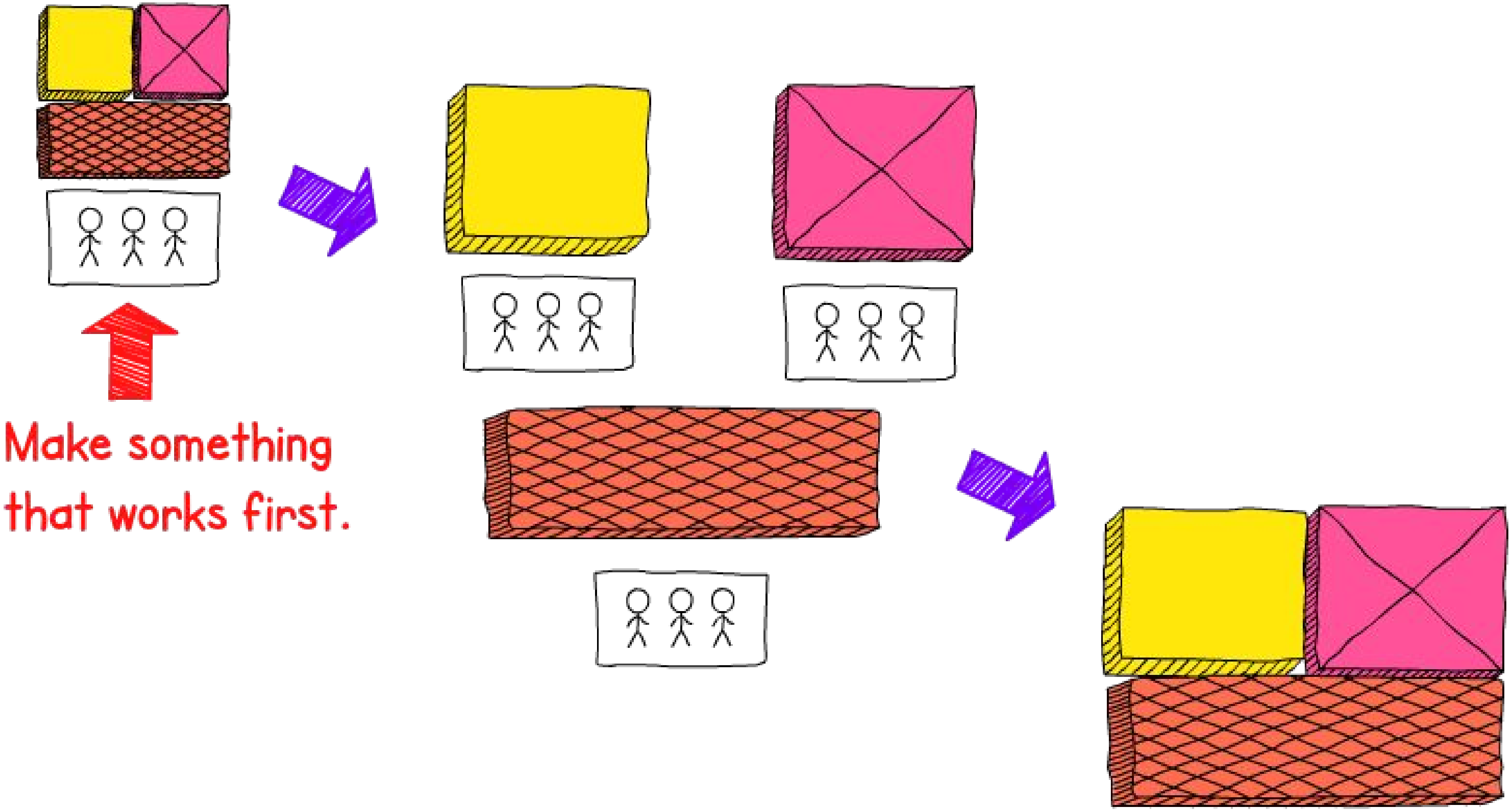


Estimate cost of delay?

“In XP, we don't divide and conquer. We conquer and divide. First we make something that works, then we bust that up and solve the little parts.”
Kent Beck

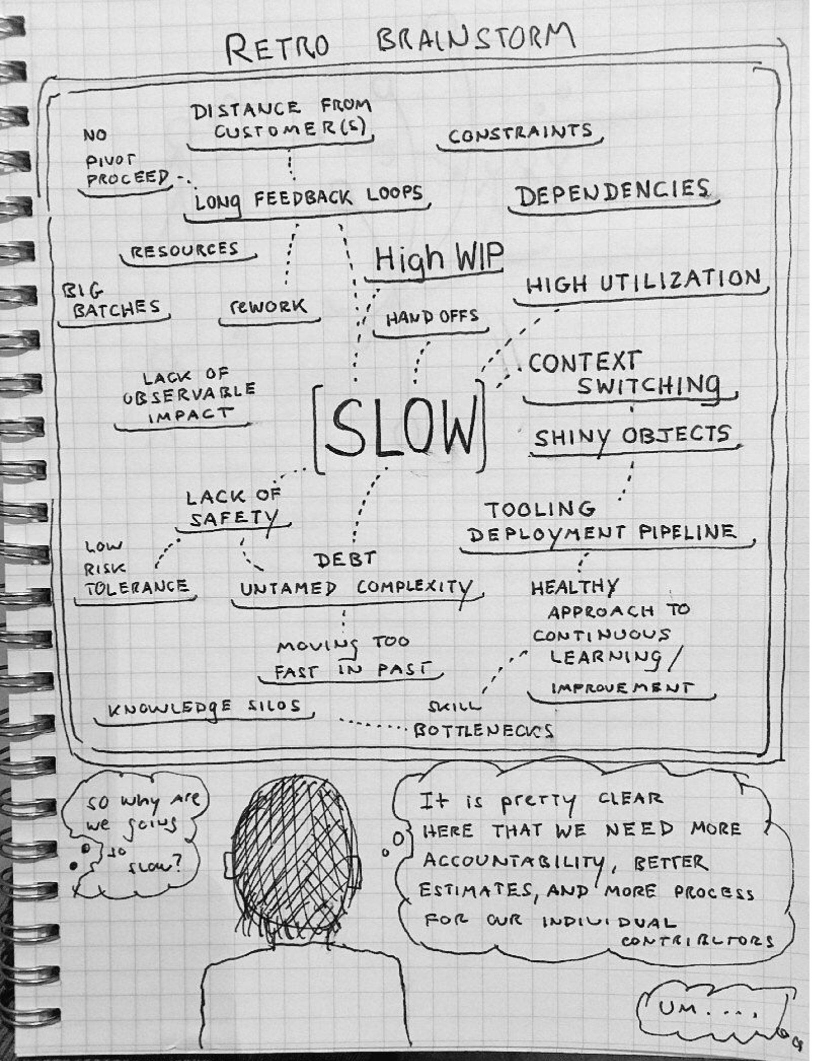


Divide and conquer leads to integration hell.


Make something
that works first.

These problems aren't new.

Why are we still solving them?



System-level problems are hard to see. There's a tendency to default to "It must be a problem with individual accountability."

All the poor approaches defer pain.

Zombie projects defer having to say no.

Not making a decision means not choosing wrong.

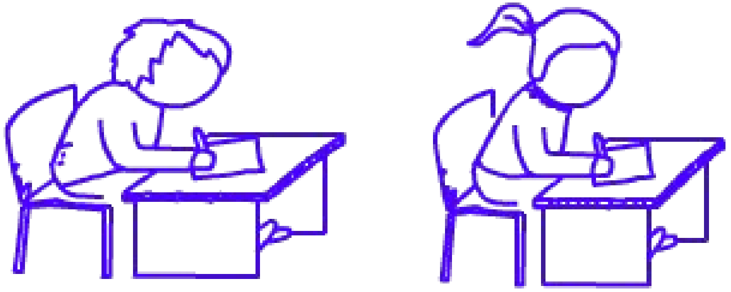
Large launches defer feedback.

Integrating later feels good now.

"This approach feels ~~better.~~"
faster

Given that... what tactics work?

These concepts are non-obvious.

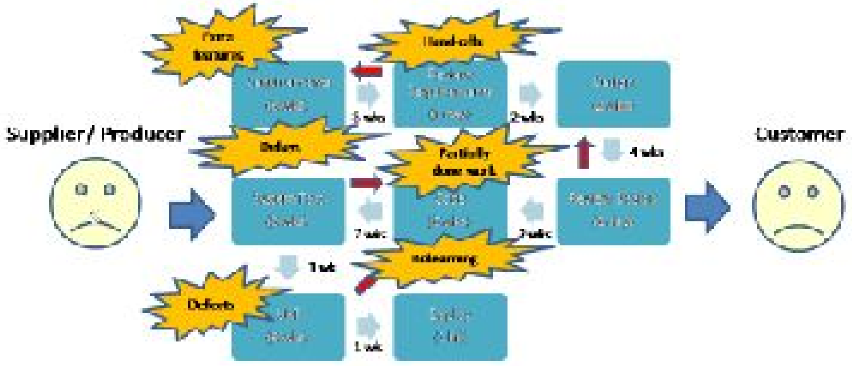


Catchy phrases (e.g., "zombie project", "conquer and divide")

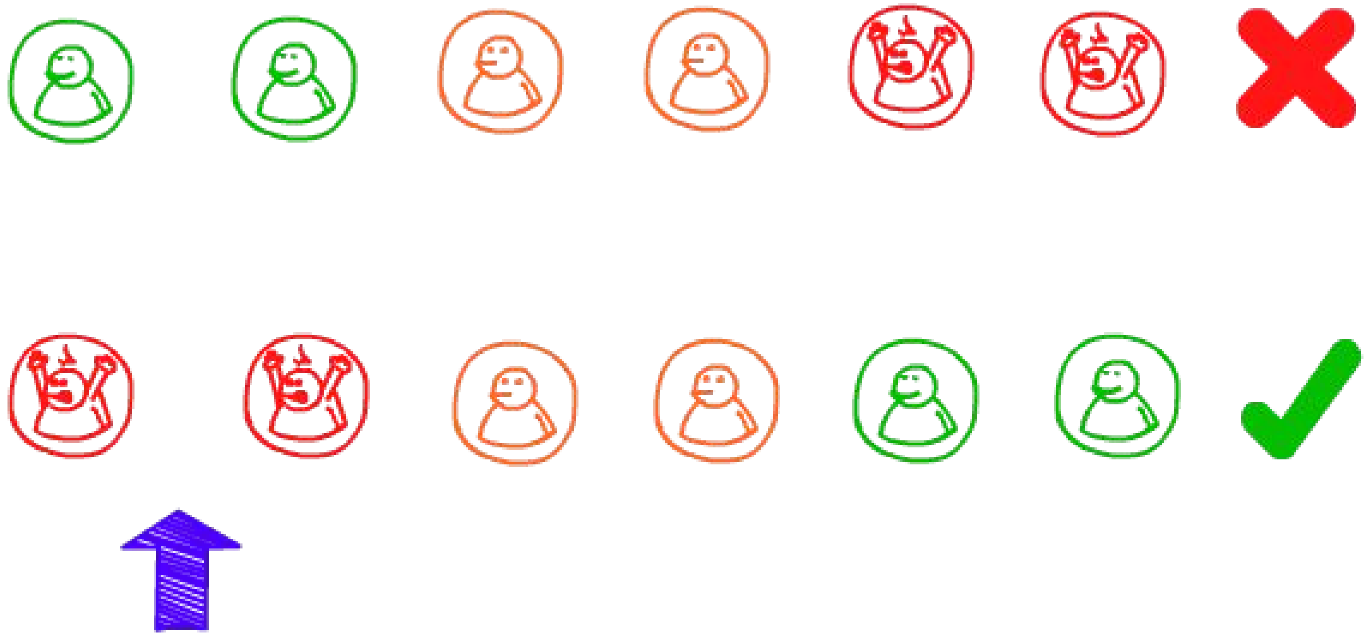
Visuals

Simulations

Value stream mapping is old school and works.



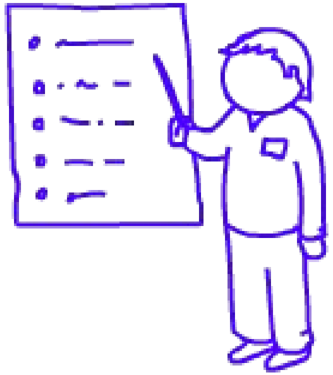
"How do we feel?" → "Does this model accurately reflect the situation?"



Warn people about "Early Pain".

Don't give up!

I don't want to end with the impression that the human experience of friction doesn't matter.



"...therefore your individual suffering doesn't matter."

"Speed" in software product delivery

“Speed” in software product delivery

What feels fast

(the human experience of speed)

?

"Speed" in software product delivery

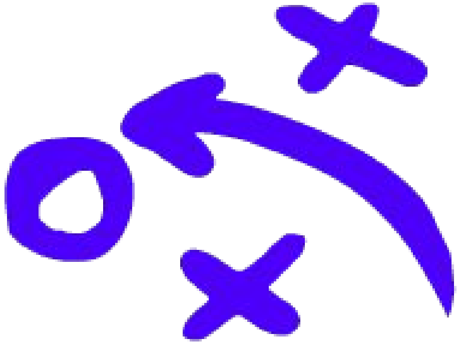
What feels fast

What is actually fast

(in terms of measurable outcomes)

Feeling fast is about removing friction.

Actually delivering fast is about designing how you deliver



First encountered Extreme Programming in 1999.

Joined ThoughtWorks in Feb 2001 (Buildmaster, Java developer, Agile / Lean consultant) (mostly Australia)

CruiseControl committer (retired)

Joined Spotify in Feb 2015 (Agile Coach)

@jchyip (twitter, medium, slideshare)
https://www.linkedin.com/in/jasonyip/
https://jchyip.blogspot.com (old blog)
jyip@spotify.com
https://www.spotifyjobs.com/