facebook

The Production Engineering Lifecycle

How we build, run, and disband great reliability-focused teams

Andrew Ryan

Production Engineer
Facebook, Inc.

What is Production Engineering @FB?

And why should you listen to me?

- Production Engineering is Facebook's reliability*-focused engineering team
- Started in 2009 with a handful of engineers, in 1 office
- Now several hundred engineers supporting dozens of teams, in 6 offices and 4 countries

* and performance, and efficiency, and scalability, and...

Our lab for growing teams and orgs...

The FB Production Engineer

What kind of candidates do we hire as Production Engineers?

- **Good coders in at least one non-shell language**
- Linux systems
- TCP/IP networking
- Distributed systems design and debugging
- Usually some “reliability engineering” background
- Will be on call

The FB Software Engineer

What kind of candidates do we hire as Software Engineers?

- Strong in coding and software design
- Other varied specialized skills
- Usually no “reliability engineering” background
- Will be on call

The FB Operating Environment

What are we putting our engineers into?

- Large scale
- Rapid pace of code/infrastructure change
- High growth rates

Putting it together: PE's and Devs

Production engineers are a scarce, often misunderstood resource

- Developer::PE ratio approximately 10:1
- Not every team can get PE support, even if they want it
- Teams don't necessarily know what to do with PE's

Devs

PE

How do we decide where to start Production Engineering teams?

Early approaches were hit-or-miss

The “hottest fire” approach

The “who screams the loudest” approach

Yay, we’re getting Production Engineers!

You fix stuff when it’s broken, right?

Yay, we’re getting Production Engineers!

You’re going to take over monitoring, go on call nights and weekends, do all of our upgrades, and...

Yay, we're getting Production Engineers!

You'll be just like the team we had at $LAST_COMPANY_I_WORKED_AT, right?

Or that team I read about on Hacker News?


NOPE

Five steps to a successful PE engagement

1. **Ensure an understanding of the PE skillset**

Ensure understanding of the PE skillset

Collaboration and accountability are key

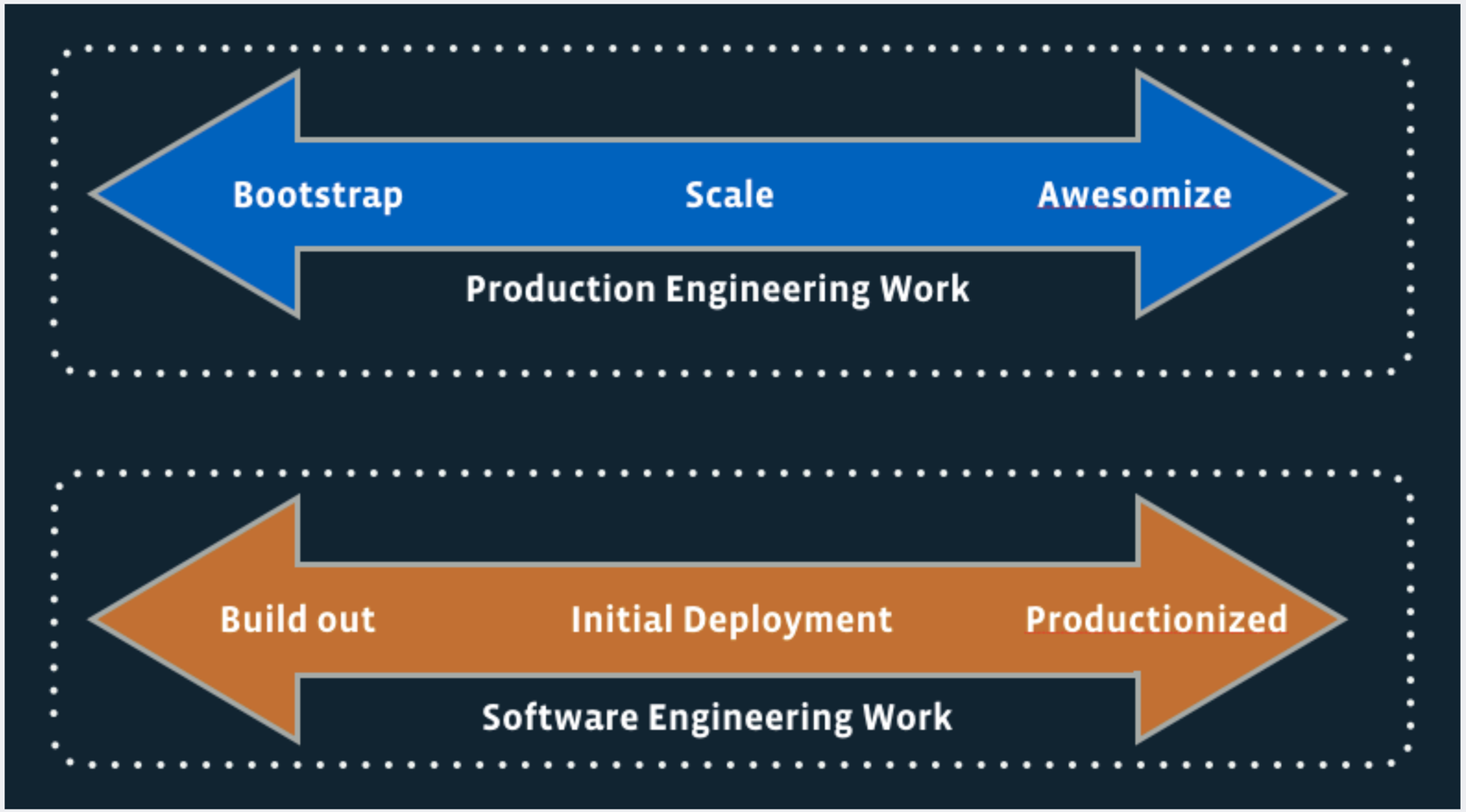
- Make sure Dev teams know PE's can code!
- Make Dev teams partners in our hiring process
  - Familiarize with PE hiring standards
  - Have them interview PE candidates
  - Make them attend hiring debriefs
- Share and publicize work done by PE's

Five steps to a successful PE engagement

1. Ensure an understanding of the PE skillset
2. **Understand the service/software we are supporting**

Understand what we are supporting

Use a “maturity model” framework



Five steps to a successful PE engagement

1. Ensure an understanding of the PE skillset
2. Understand the service/software we are supporting
3. **Decide on a level of engagement**

Decide on a level of engagement

- Advisory
  - 1-2 production engineers, not full-time
  - No formal deliverables
- Consulting
  - 1-3 production engineers, not full-time
  - Formal deliverables, negotiated on an ongoing basis
- Full
  - Full-time production engineers, 6+ if on call is needed
  - Dedicated team, usually on call

Five steps to a successful PE engagement

1. Ensure an understanding of the PE skillset
2. Understand the service/software we are supporting
3. Decide on a level of engagement
4. **Assign the most suitable engineers**

Assign the most suitable engineers

Balance between personal growth, existing skills, and business needs

Troubleshooting

TCP/IP networking

Relationship building

Efficiency

Security

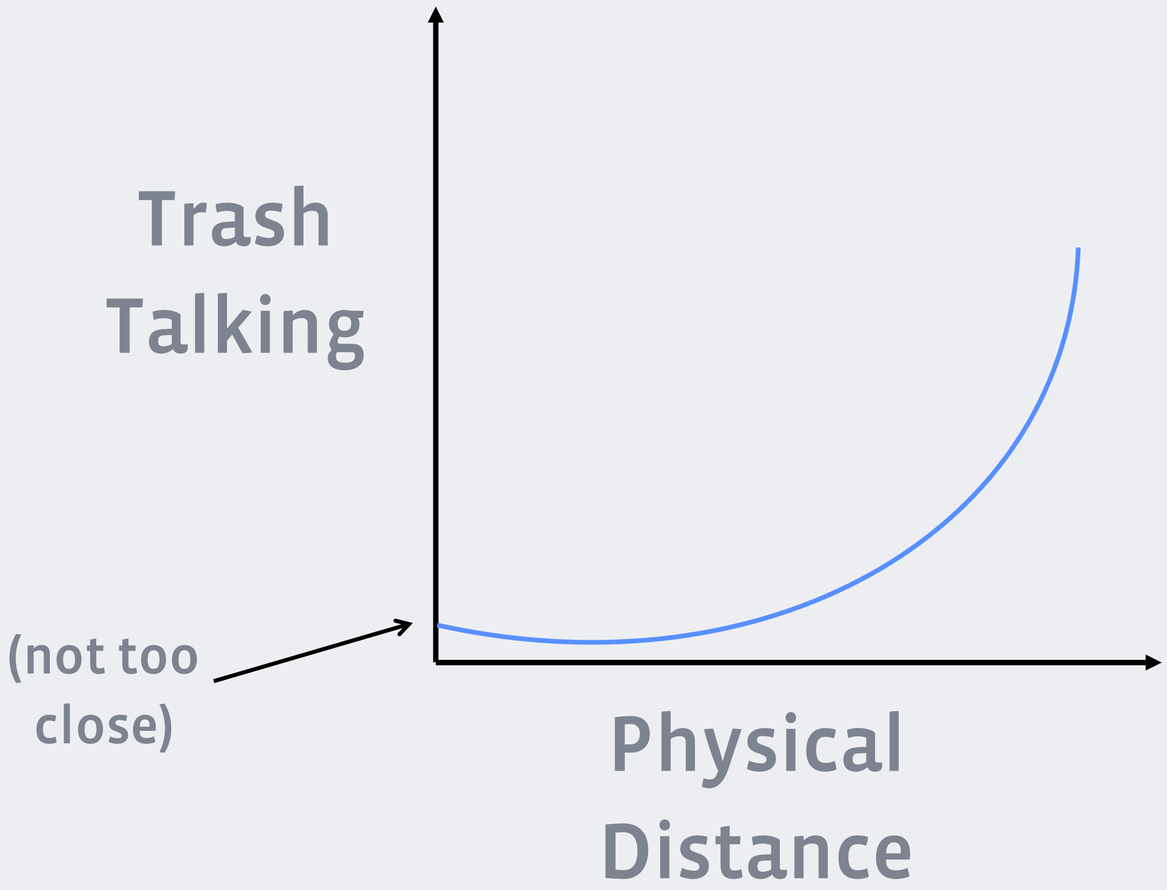
Storage

Coding

Monitoring & alerting

Sit PE's and devs together

Just say no to the "Ops Pit"



Five steps to a successful PE engagement

1. Ensure an understanding of the PE skillset
2. Understand the service/software we are supporting
3. Decide on a level of engagement
4. Assign the most suitable engineers
5. **Actually split up tasks and on call work**

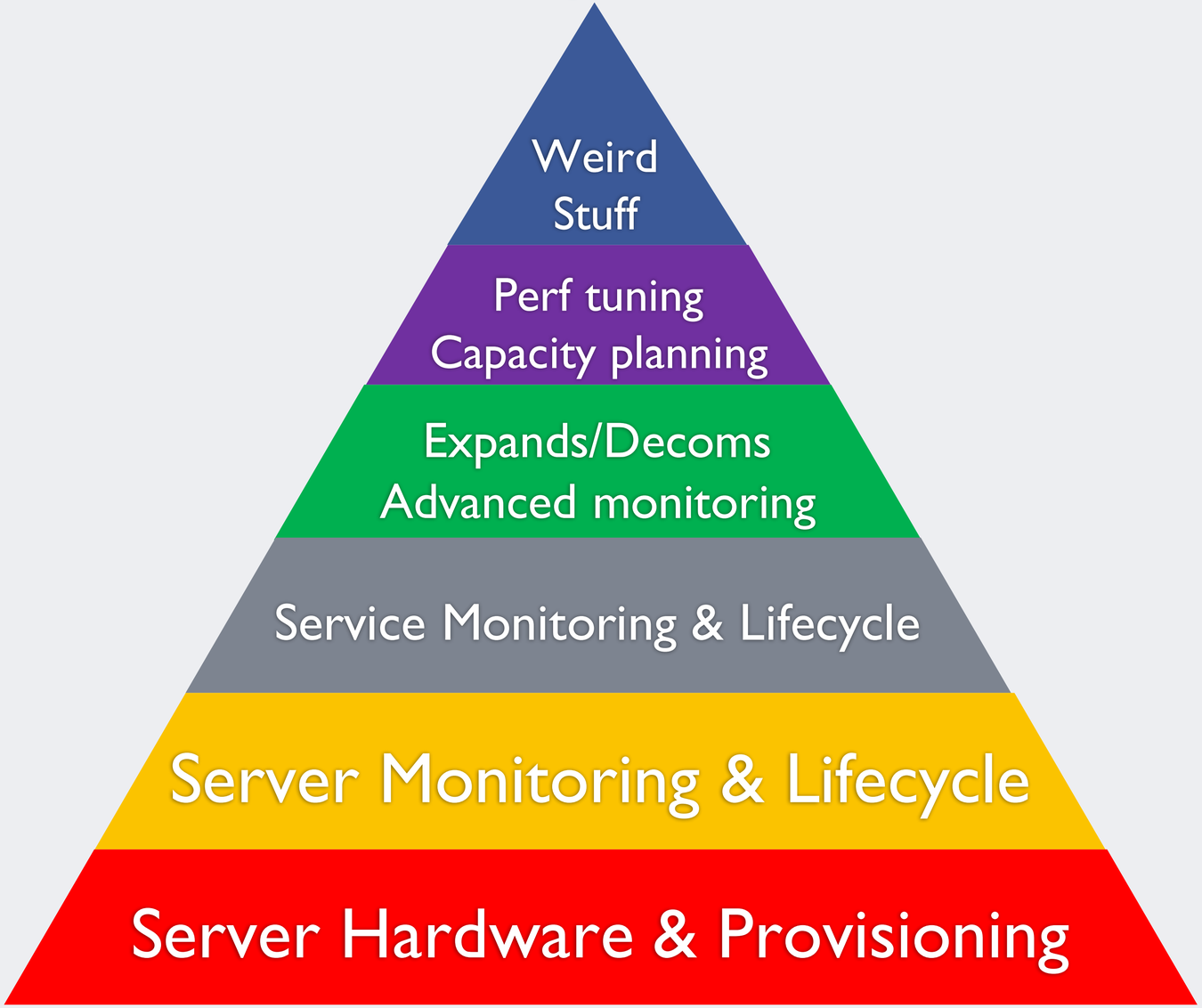
Actually splitting up tasks and on call

Where the rubber meets the road

Prioritizing task work: Hierarchy of needs

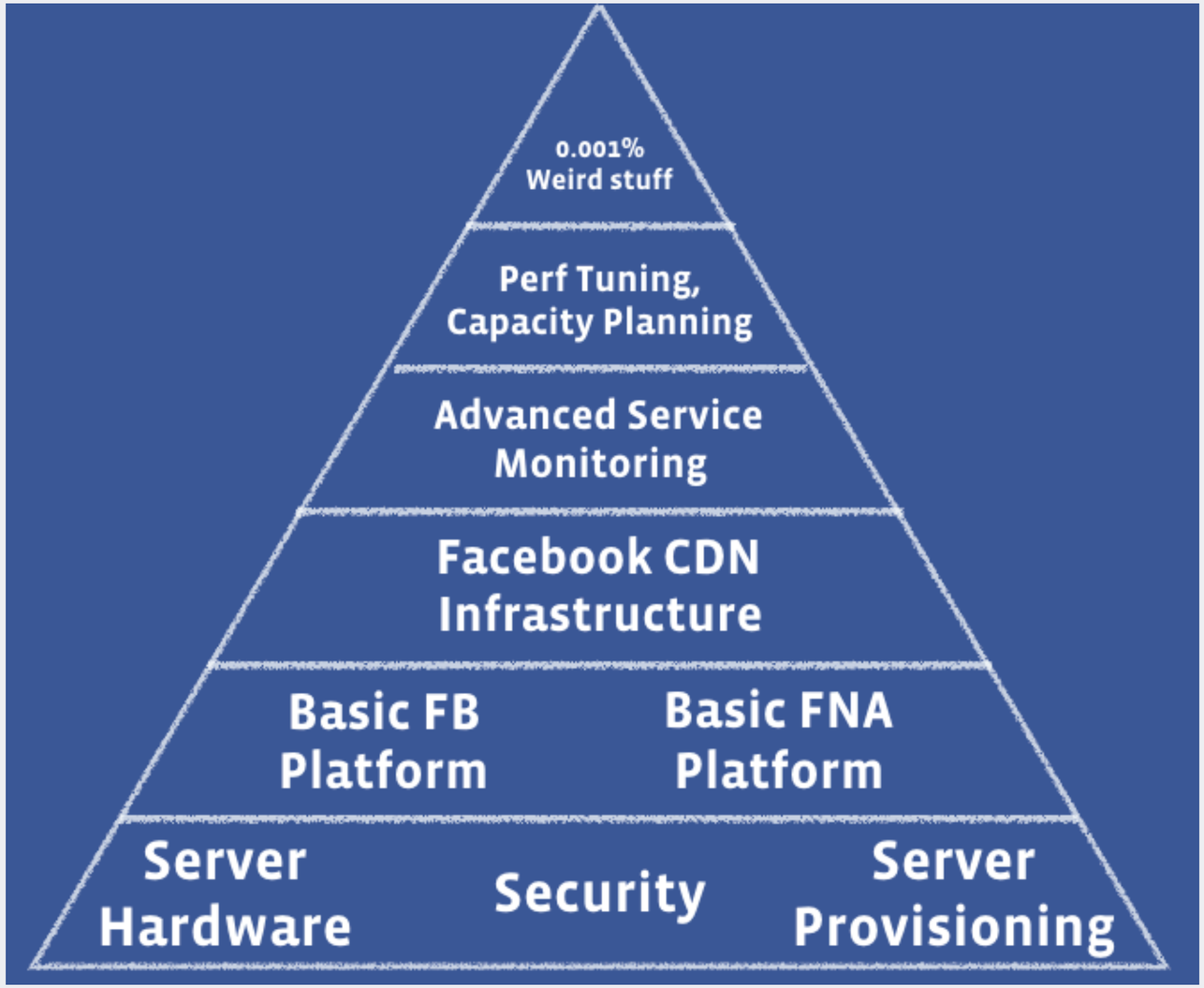
Maslow's hierarchy of needs

PE “Basic” Hierarchy of needs



PE Hierarchy of needs adapted for teams

Facebook Network Appliance (FNA)



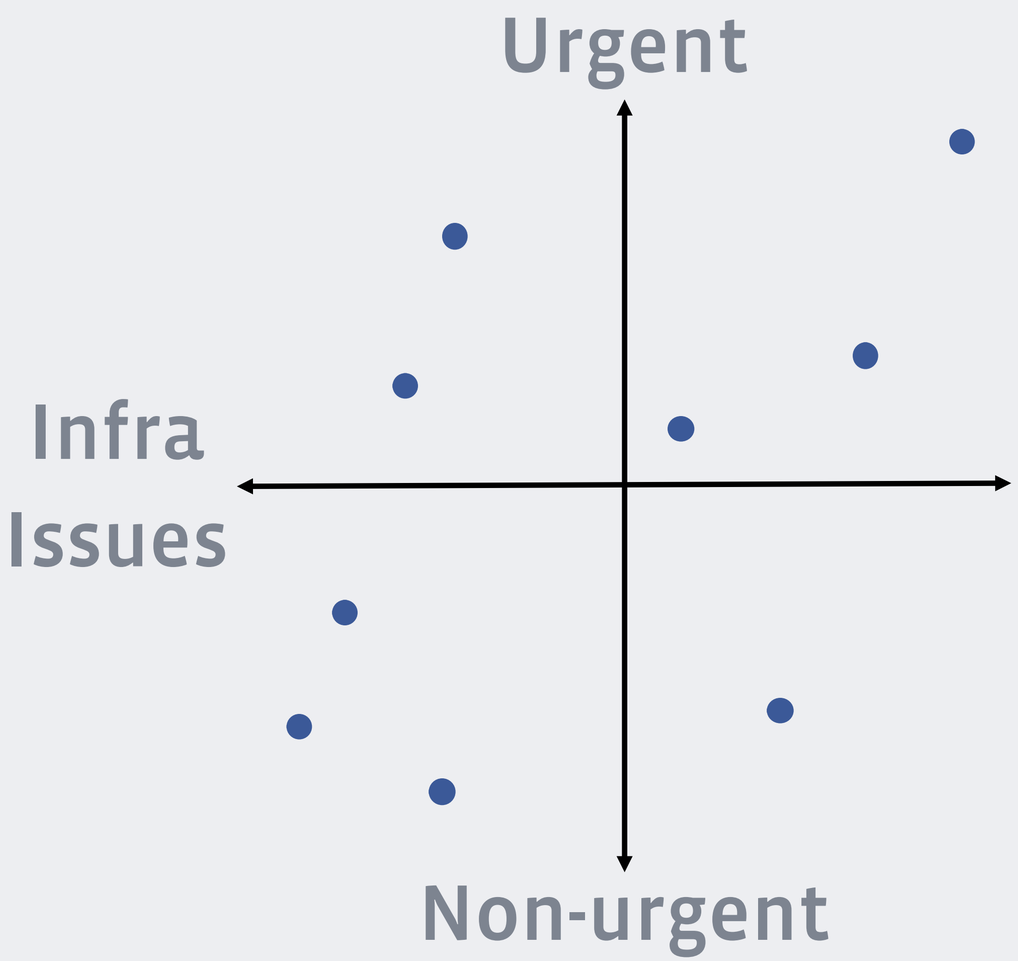
Integrating on call: guiding principles

Stop and take a deep breath before throwing bodies in the mix

1. Avoid burning out your engineers
2. Give your engineers enough time on call to stay sharp
3. Let your engineers do the work they are best at

What comprises your actual on call load?

Drive your decisions with facts, not emotions



Integrating on call: Mixed rotation

PE's and Devs mix together in a single on call rotation

- Use it when you have:
  - Low/medium software/infrastructure complexity
  - Moderate volume of issues
- Disadvantages:
  - Only scales to team size of ~12
  - Engineers often not working at what they are best at

Integrating on call: Separate rotations

Devs and PE's are in separate, concurrent rotations

- Use it when you have:
  - High software/infrastructure complexity
  - High issue volume
  - >12 engineers to go oncall
- Disadvantages
  - Contact points between oncalls increase
  - Some issues get dropped and misrouted
  - Engineers usually working at what they are best at

Separate rotations in action

Facebook's Core Data Cache team: 5 concurrent rotations

- Each oncall has its own service-level alarms
- Umbrella "Something is really wrong" alarms go to >1 oncall
- On calls constantly coordinate with each other during shifts

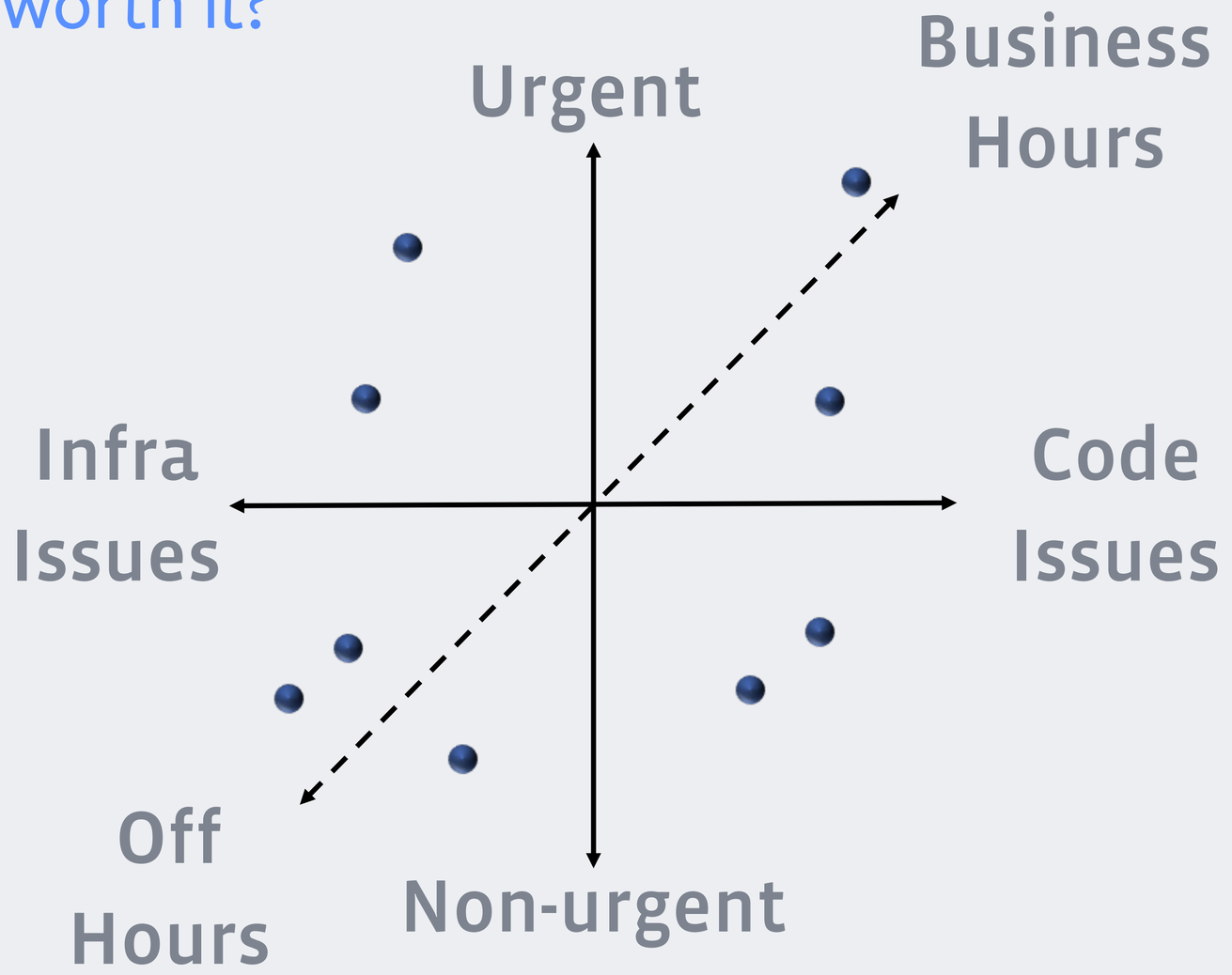
Integrating on call: None

No PE's are on call!

- Software/service not supportable
- Use on call integration as a carrot/stick for Dev team
- Get on call as soon as possible: reliability is a muscle, use it or lose it!

Integrating multiple timezones to on call

Is the price worth it?



Disbanding/merging reliability teams

Think “fertilizer” not “failure”

- Re-orgs
- Unhealthy team dynamic
- Team has “run its course”
- Move down the engagement ladder

Conclusions

Learn from our failures

We messed this up lots of times so you won't have to

1. "Field of Dreams" approach to PE/Dev integration
2. Separate PE/Dev teams by time and space
3. Leave people in teams too long
4. Leave unhealthy/stagnant teams alive too long
5. Put new senior hires in as tech leads right away

Learn from our successes

Team mobility and cross-pollination are key

1. Hire strong generalists
2. Maintain identical hiring standards across global offices
3. Invest in common reliability infrastructure technology
4. Have a consistent approach to starting, running, and disbanding reliability-focused teams
5. Co-locate PE and Dev teams whenever possible

facebook