Shaping Reality to Shape Outcomes

Making SRE Work with Uber Growth

Tom Croucher
@sh1mmer
Uber Growth  
(a super scientific graph)

every time you think you are here
Uber Growth (a super scientific graph)

turns out you are still here
uber is complicated

a bit like my code
How does a rider from **Bangalore**, travelling in **New York**, use Facebook messenger to request a ride via *api.uber.com*?

How does an Uber Engineer build a reliable feature that supports this without a building feature specifically to support this?
every other engineer
in the company

engineers you know
functions of the company
the school of hard knocks

i have 99 problems, and reliability is 1
creating new realities

like Steve Jobs but for infrastructure
Amount of the time you want engineers to build for failover: 100%

Amount of the time you actually failover: 0%

Amount of the time engineers actually build for failover: 0%
failover “testing”
aka you just lost all cooling in your China datacenter
Amount of the time you want engineers to build for failover

Amount of the time you actually failover, including tests

Amount of the time engineers actually build for failover
1) Humans (Sebastian) inducing failure

SebDestroy

2) Baseline for repetition
- Reliable Core Services: RealTime Trip Replication (RTTR)
- Reliable Failover Tools: Zombie Apocalypse Recovery (ZAR)

3) Self-service
- Automated destruction
- Randomized tests
ask for what you want

like a pony
Amount of datacenters most services should work in: 100%
Amount of datacenters most services actually work in: 0%
Amount of the datacenters you actively use to serve traffic: 100%
all datacenter reality

all requirements all the time
Amount of datacenters most services should work in

Amount of datacenters most services actually work in

Amount of the datacenters you actively use to serve traffic
what even is productivity

it’s probably measured in $/%#! / minute
some code

0% of users

100% of users

productivity
Some code increases productivity for 5% of users. For 100% of users, there is a super effective rollback of bad code.
you don’t know anything

and you never will
every other thing you don’t know that might break in the company

what your know
Universal Guide to Stuff Just Works Now™

1) Make sure you have a place to move traffic that is not broken (most of the time)
2) Get good at knowing that users are having a bad time (why is less important)
3) Get really good at moving traffic to a safe place fast (ideally automatically)
3b) Fix whatever is broken, I guess
Questions?

tomc@uber.com
@sh1mmer

We’re hiring!
http://t.uber.com/srejobs