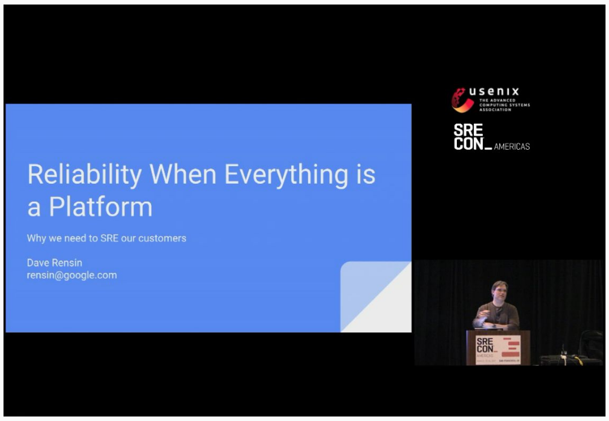
Building Successful SRE in Large Enterprises - One Year Later

Dave Rensin
Director, Google CRE
rensin@google.com

Google

One Year Ago...

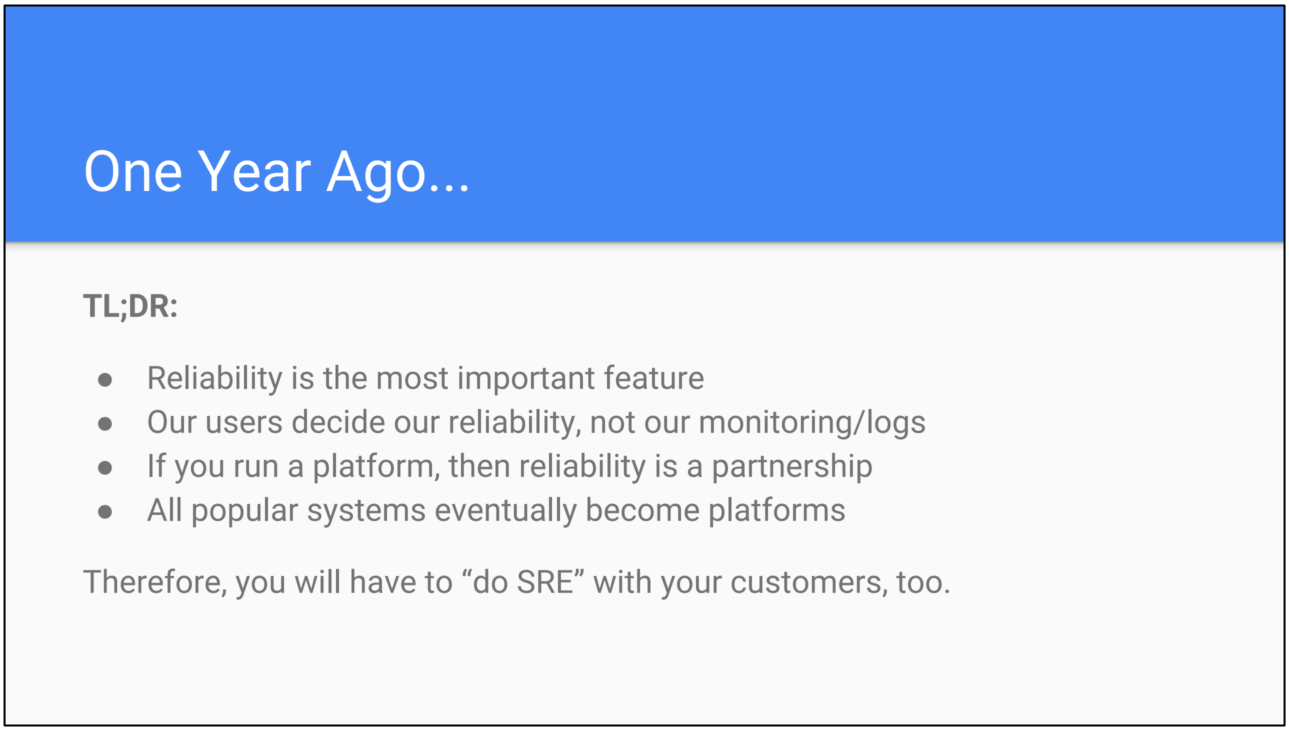

Reliability When Everything is a Platform

Why we need to SRE our customers

Dave Rensin
rensin@google.com

https://goo.gl/T83gcf

At SRECon last year I came to convince you that you would have to “do SRE” with your customers and to tell you about a new thing we were trying at Google called CRE -- Customer Reliability Engineering. If you weren’t at that talk (or don’t remember it), you can see it at that link (or use the QR code).



One Year Ago...

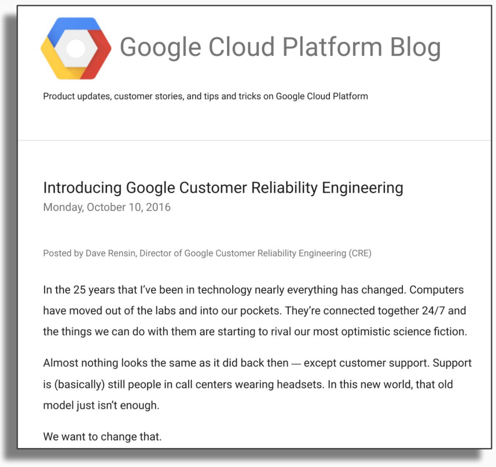
TL;DR:

- Reliability is the most important feature
- Our users decide our reliability, not our monitoring/logs
- If you run a platform, then reliability is a partnership
- All popular systems eventually become platforms

Therefore, you will have to “do SRE” with your customers, too.

A quick recap for everyone...

Introducing CRE



Google Cloud Platform Blog

Product updates, customer stories, and tips and tricks on Google Cloud Platform

Introducing Google Customer Reliability Engineering

Monday, October 10, 2016

Posted by Dave Rensin, Director of Google Customer Reliability Engineering (CRE)

In the 25 years that I've been in technology nearly everything has changed. Computers have moved out of the labs and into our pockets. They're connected together 24/7 and the things we can do with them are starting to rival our most optimistic science fiction.

Almost nothing looks the same as it did back then — except customer support. Support is (basically) still people in call centers wearing headsets. In this new world, that old model just isn't enough.

We want to change that.

COMPUTERWORLD FROM IDG

Home > Cloud Computing

NEWS ANALYSIS

Google's secret weapon in cloud computing: people

While the company touted technology and service at Google Cloud Next this week, customers had other reasons for switching

By David Needle

Computerworld | MAR 10, 2017 12:16 PM PT

InfoQ

DevOps and Cloud InfoQ T

Posted by Manuel Pais, Daniel Bryant, Richard Seroter

QCon Software Development Conference

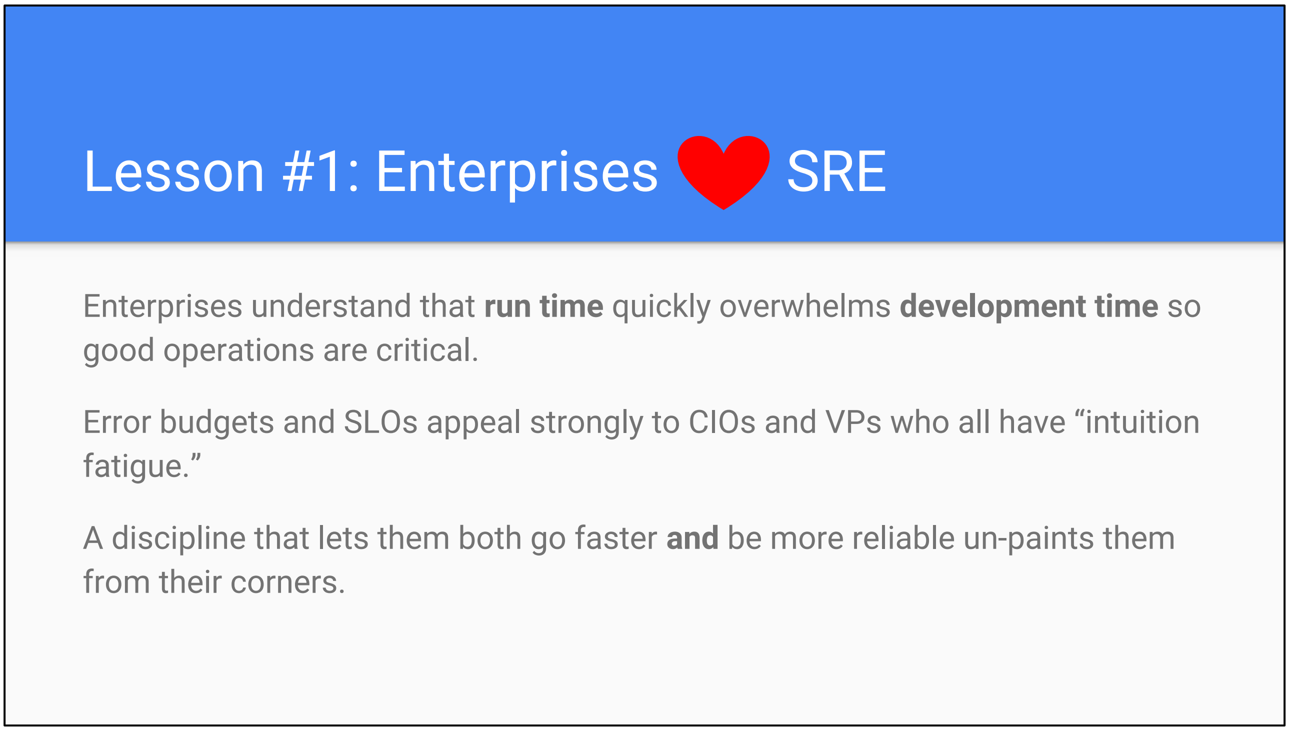
AI & ML SF Apr 9-11, 2018

New York Jun 25-29, 2018

San Francisco Nov 5-9, 2018

The Architects' Newsletter

Because we believe these things, we launched CRE -- a global team of highly experienced Google SREs who help customers learn SRE by actually "doing SRE" together. <click> People seemed to think it was a good idea and we were off and running.. <click> In this last year, we've learned some pretty interesting stuff…



Lesson #1: Enterprises ❤ SRE

Enterprises understand that **run time** quickly overwhelms **development time** so good operations are critical.

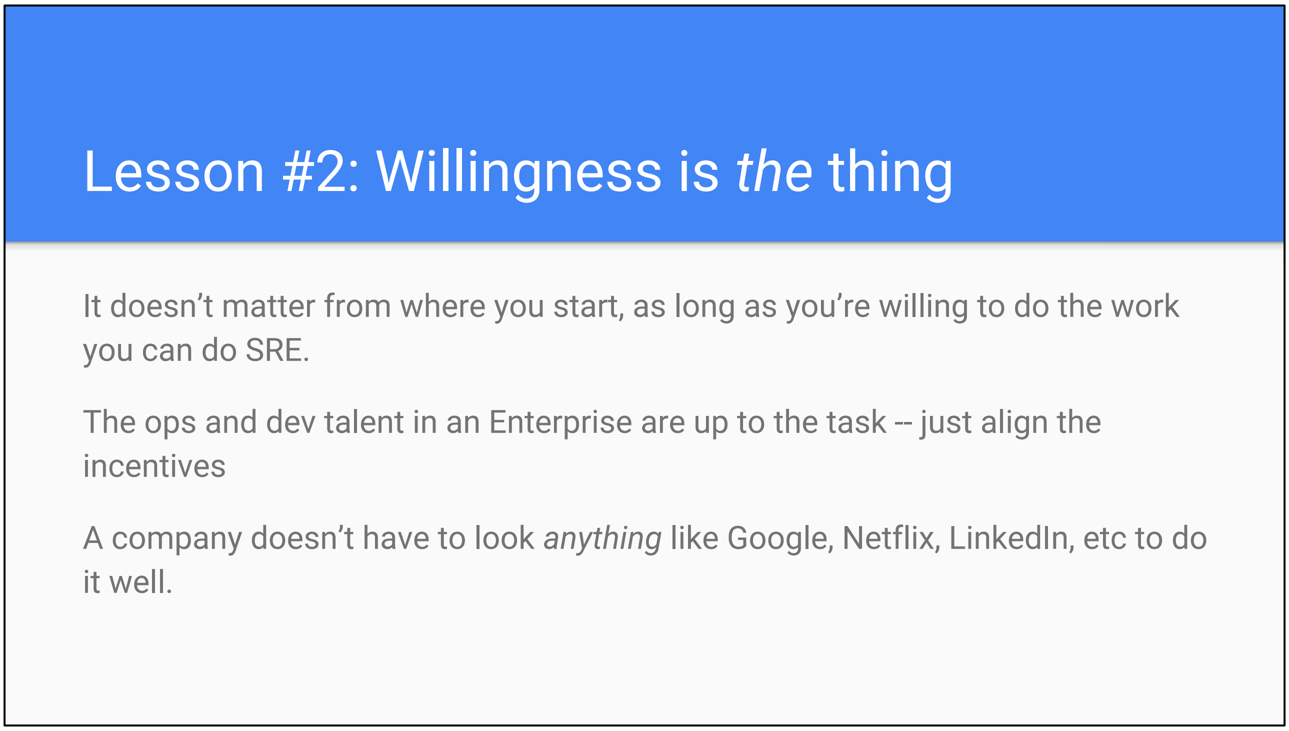
Error budgets and SLOs appeal strongly to CIOs and VPs who all have “intuition fatigue.”

A discipline that lets them both go faster **and** be more reliable un-paints them from their corners.

The older a company is, the more they have systems that have been running a long time. The understand that the long-term costs (and therefore returns) of a system are dominated by its operational burden. The lower the burden, the higher the ROI

Experienced CIOs and VPs understand the emotional fatigue that comes with making critical operational and feature decisions via intuition. It’s tiring, imprecise, and risky. They’re looking for a better way.

As these systems get older, the space for innovation shrinks because of operational growth. Any system that lets them recover that innovation space and keep reliability is a welcome godsend.



Lesson #2: Willingness is *the* thing

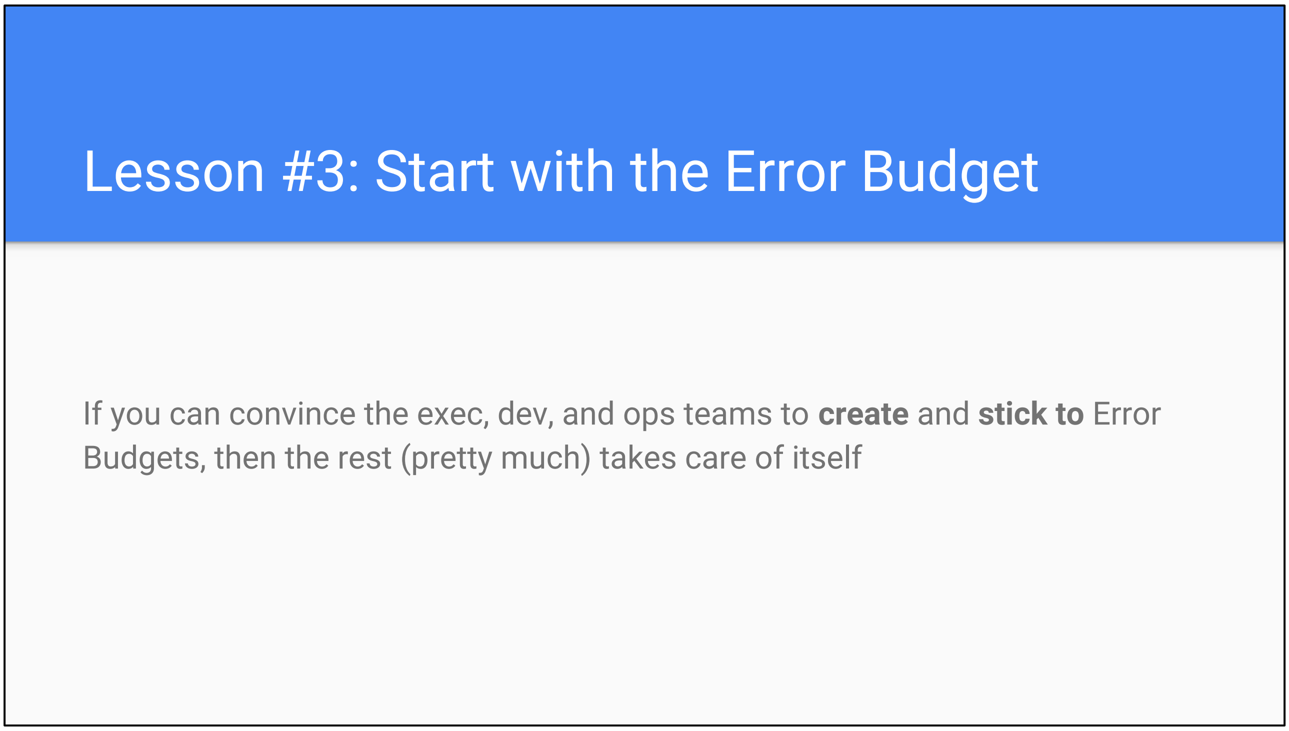
It doesn't matter from where you start, as long as you're willing to do the work you can do SRE.

The ops and dev talent in an Enterprise are up to the task -- just align the incentives

A company doesn't have to look *anything* like Google, Netflix, LinkedIn, etc to do it well.

Here in Silicon Valley, Enterprises get a bad rap. They're "slow". They're "outdated". They don't have the "best engineers".. Etc.. Let's bust a few myths, shall we?

1) The software engineering and operational talent in any large Enterprise is up to the task of adopting SRE. There might be skills gaps, but as long as they're willing to close those, they can get it done. We've been consistently impressed with the level of talent in these companies. LOTS of talented people live and work outside of Silicon Valley. We should get over ourselves!
2) The key trait that predicts success is the *willingness* to go on the journey. Willingness of the execs to give air cover. Willingness of the dev and ops teams to lay down their arms.
3) SRE works in companies in all kinds of shapes and sizes. You don't have to look like Google/Netflix/LinkedIn/etc to do this. Just ask The Home Depot. They don't look anything like Google and they are awesome SREs!



Lesson #3: Start with the Error Budget

If you can convince the exec, dev, and ops teams to **create** and **stick to** Error Budgets, then the rest (pretty much) takes care of itself

the hardest thing we've found for other companies is to develop principled responses to "out of error budget": OK, you go to feature freeze. What now? Have to distinguish between tactical and strategic fixes e.g. is it OK to live with 2-4 months of expected outages if we have a strategic plan to remove our primary outage trigger after that?

Lesson #3: Example -- Evernote

“...it is important to start the conversation from the point of view of your customers: **what promises are you trying to uphold?** Evernote's goal was to ensure the availability of the service for users to access and sync their content. Our SLO journey started from that goal. Starting from that goal, **we kept our first pass simple by focusing on uptime.** Using this simple first approach, we could clearly articulate **what we were measuring, and how**.”

“...’Perfect is the enemy of good.’ **Even when SLOs aren't perfect, they're good enough to guide improvements over time.** A "perfect" SLO would be one that measures every possible user interaction with our service and accounts for all edge cases. [...] Instead, we selected an initial SLO that covered most, but not all, user interactions, which was a good proxy for quality of service.”

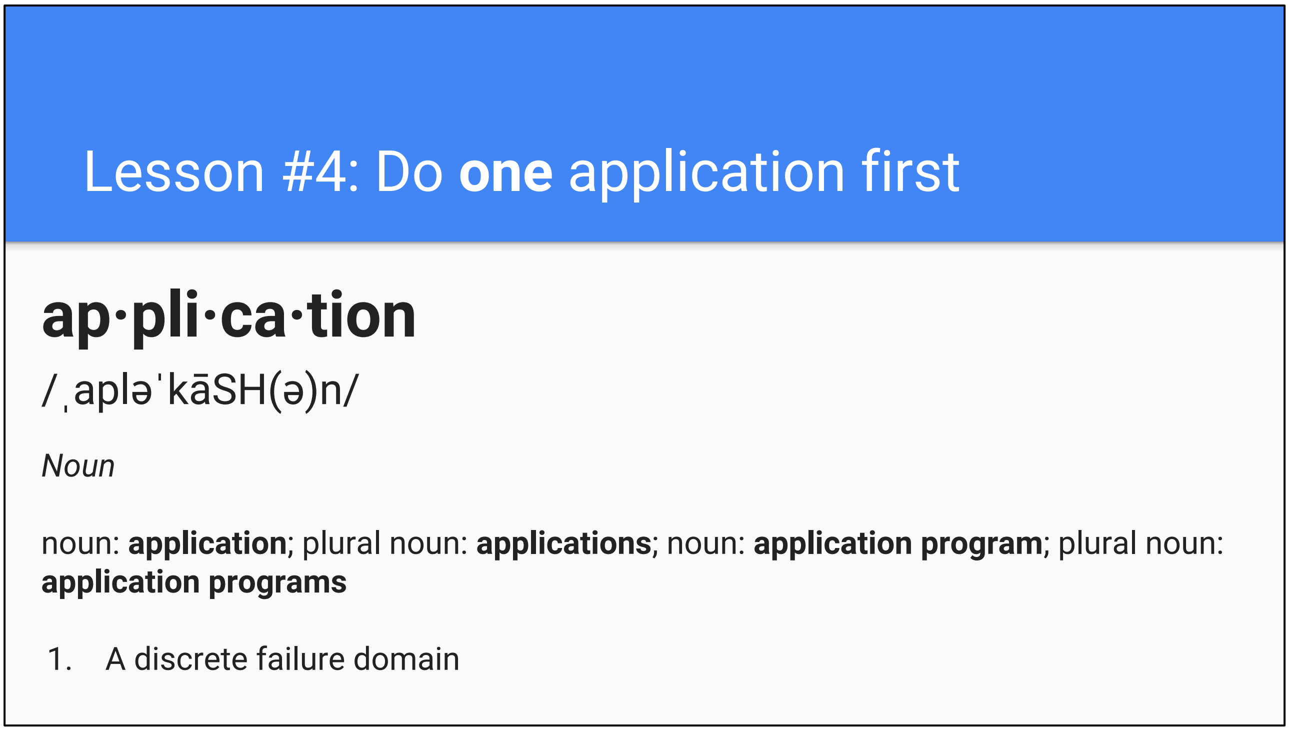
-- Ben McCormack (VP Operations / Chief of Staff -- Evernote)

Lesson #3: Example -- The Home Depot

“**The Home Depot didn’t have a culture of SLOs**. Monitoring tools and dashboards were plentiful, but were scattered everywhere and didn’t track data over time. We weren’t always able to pinpoint the service at the root of a given outage. Often, we began troubleshooting at the user-facing service and worked backwards until we found the problem, **wasting countless hours**. If a service required planned downtime, its **dependent services were surprised**. If a team needed to build a 3.5 nines service, **they wouldn’t know if the service they had a hard dependency** on could support them with even better uptime (4 nines). These disconnects caused **confusion and mistrust** between our software development teams.”

“Once SLOs were firmly cemented in the organization’s collective mind and effective automation and reporting were in place, **new SLOs proliferated quickly.** After tracking SLOs for **about 50 services at the beginning of the year, by the end of the year we were tracking SLOs for 800 services, with about 50 new services being registered [...] per month.**”

-- William Bonnell (Sr. Director, SRE -- The Home Depot)



Lesson #4: Do **one** application first

ap·pli·ca·tion

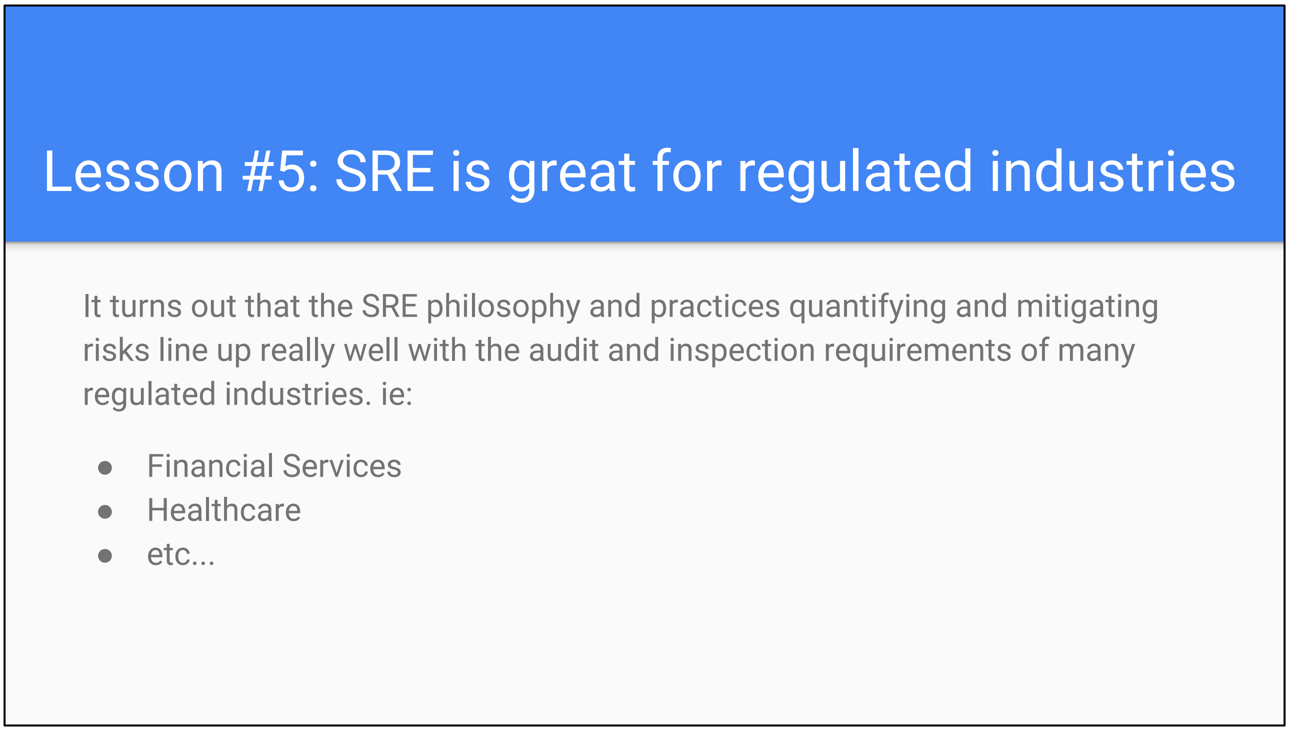
/ˌapləˈkāSH(ə)n/

Noun

noun: **application**; plural noun: **applications**; noun: **application program**; plural noun: **application programs**

1. A discrete failure domain

Start with one discrete failure domain. Circumscribe the execs, devs, and ops folks who own it and get buy-in and success. Once everyone sees it working, they'll want to do it, too! **Caveat:** Be sure to pick an app that is meaningful to the business. Non-business-important things (toys, tests, etc) won't convey the lessons you want to teach.

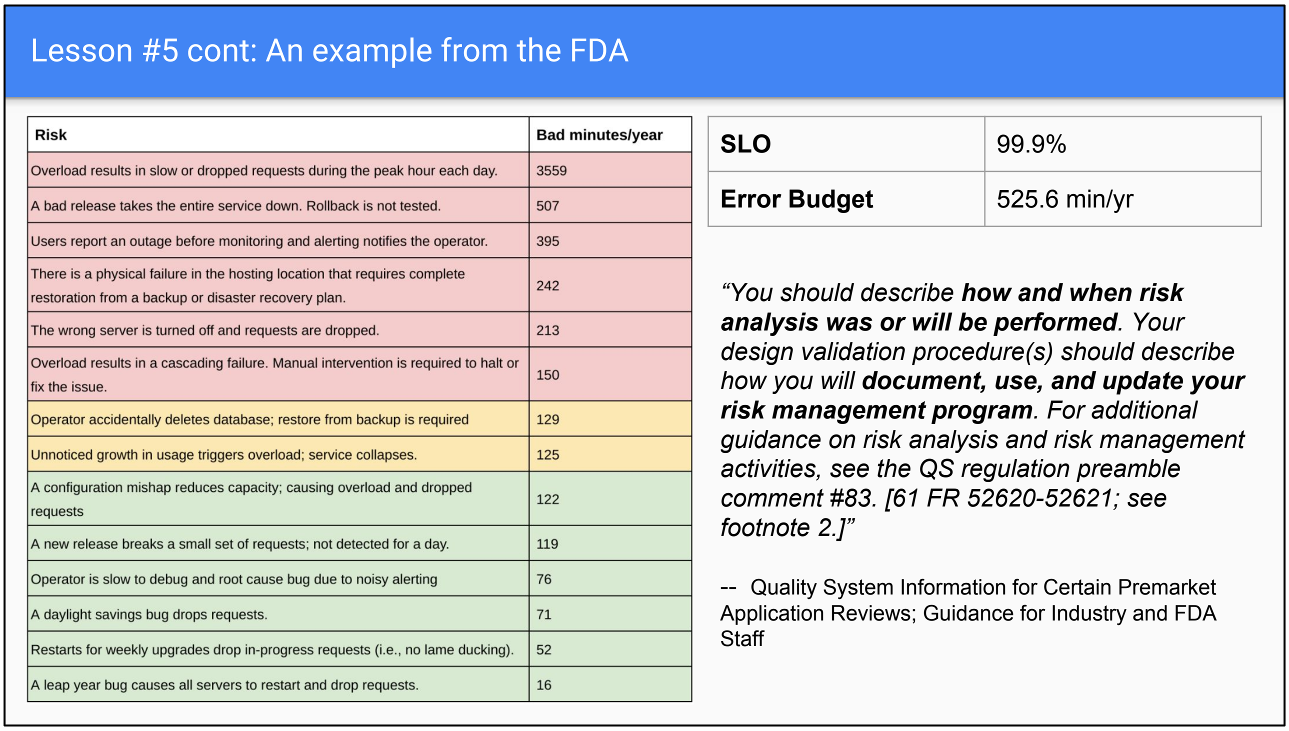


Lesson #5: SRE is great for regulated industries

It turns out that the SRE philosophy and practices quantifying and mitigating risks line up really well with the audit and inspection requirements of many regulated industries. ie:

- Financial Services
- Healthcare
- etc...

Regulators understand that 100% reliability or zero errors is unreasonable. What they want is a way to connect regulatory policy and risk regimes to operational principles -- which SRE happens to do really nicely.



Lesson #5 cont: An example from the FDA

Risk	Bad minutes/year
Overload results in slow or dropped requests during the peak hour each day.	3559
A bad release takes the entire service down. Rollback is not tested.	507
Users report an outage before monitoring and alerting notifies the operator.	395
There is a physical failure in the hosting location that requires complete restoration from a backup or disaster recovery plan.	242
The wrong server is turned off and requests are dropped.	213
Overload results in a cascading failure. Manual intervention is required to halt or fix the issue.	150
Operator accidentally deletes database; restore from backup is required	129
Unnoticed growth in usage triggers overload; service collapses.	125
A configuration mishap reduces capacity; causing overload and dropped requests	122
A new release breaks a small set of requests; not detected for a day.	119
Operator is slow to debug and root cause bug due to noisy alerting	76
A daylight savings bug drops requests.	71
Restarts for weekly upgrades drop in-progress requests (i.e., no lame ducking).	52
A leap year bug causes all servers to restart and drop requests.	16

SLO	99.9%
Error Budget	525.6 min/yr

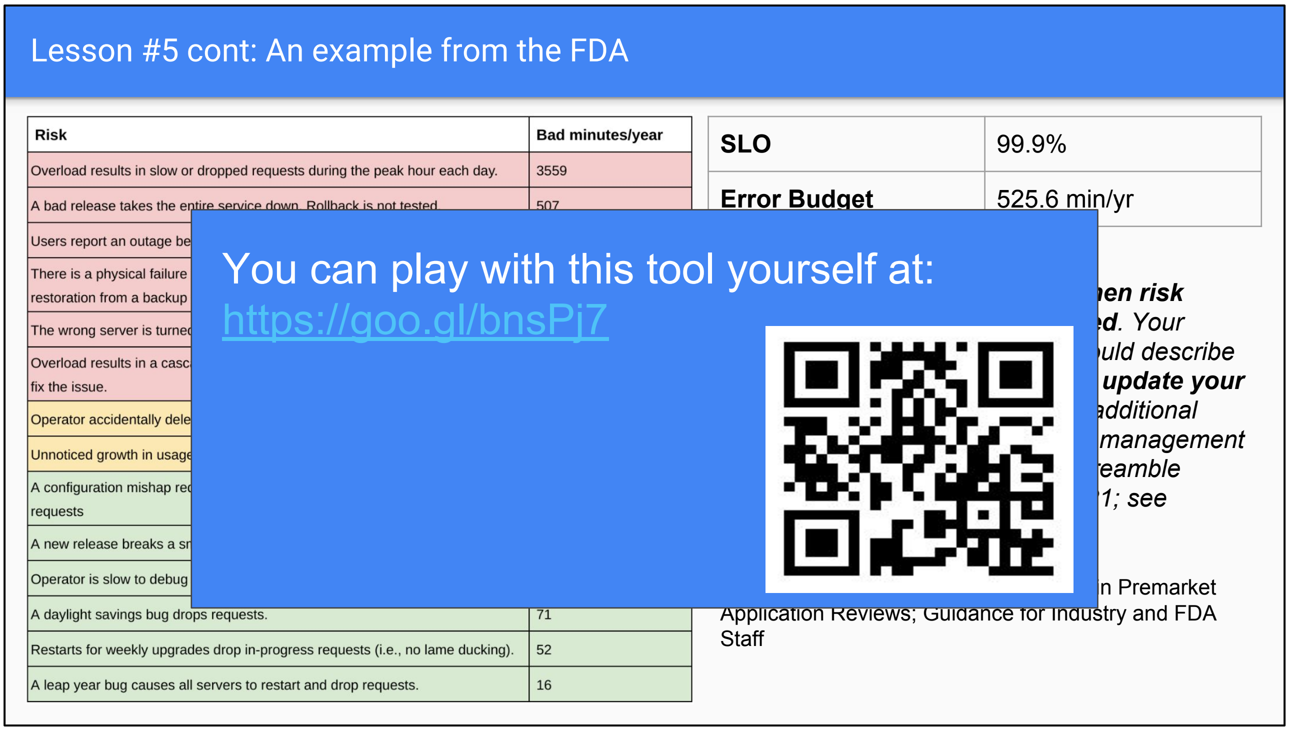
*"You should describe **how and when risk analysis was or will be performed**. Your design validation procedure(s) should describe how you will **document, use, and update your risk management program**. For additional guidance on risk analysis and risk management activities, see the QS regulation preamble comment #83. [61 FR 52620-52621; see footnote 2.]"*

-- Quality System Information for Certain Premarket Application Reviews; Guidance for Industry and FDA Staff

Let's look really quickly at an error budget analysis of some fictitious software system. It has an SLO of three nines, which means it has a yearly error budget of ~526min (43.2min per 30 days). The red items are risks that burn big chunks of the budget -- say >- 25%. You **must** fix those. The yellow, you **should** fix, but are survivable. The green don't threaten your budget enough to worry about now. You can trade greens for yellows, maybe.

Now let's look at what the US Food and Drug Administration requires for companies manufacturing health devices (emphasis, mine).

They don't require a certain risk target, but they **do** require a systematic process for identifying, quantifying, and mitigating risks -- just like any SRE would!



Lesson #5 cont: An example from the FDA

Risk	Bad minutes/year
Overload results in slow or dropped requests during the peak hour each day.	3559
A bad release takes the entire service down. Rollback is not tested.	507
Users report an outage be	
There is a physical failure restoration from a backup	
The wrong server is turned	
Overload results in a casc fix the issue.	
Operator accidentally dele	
Unnoticed growth in usage	
A configuration mishap red requests	
A new release breaks a sr	
Operator is slow to debug	
A daylight savings bug drops requests.	71
Restarts for weekly upgrades drop in-progress requests (i.e., no lame ducking).	52
A leap year bug causes all servers to restart and drop requests.	16

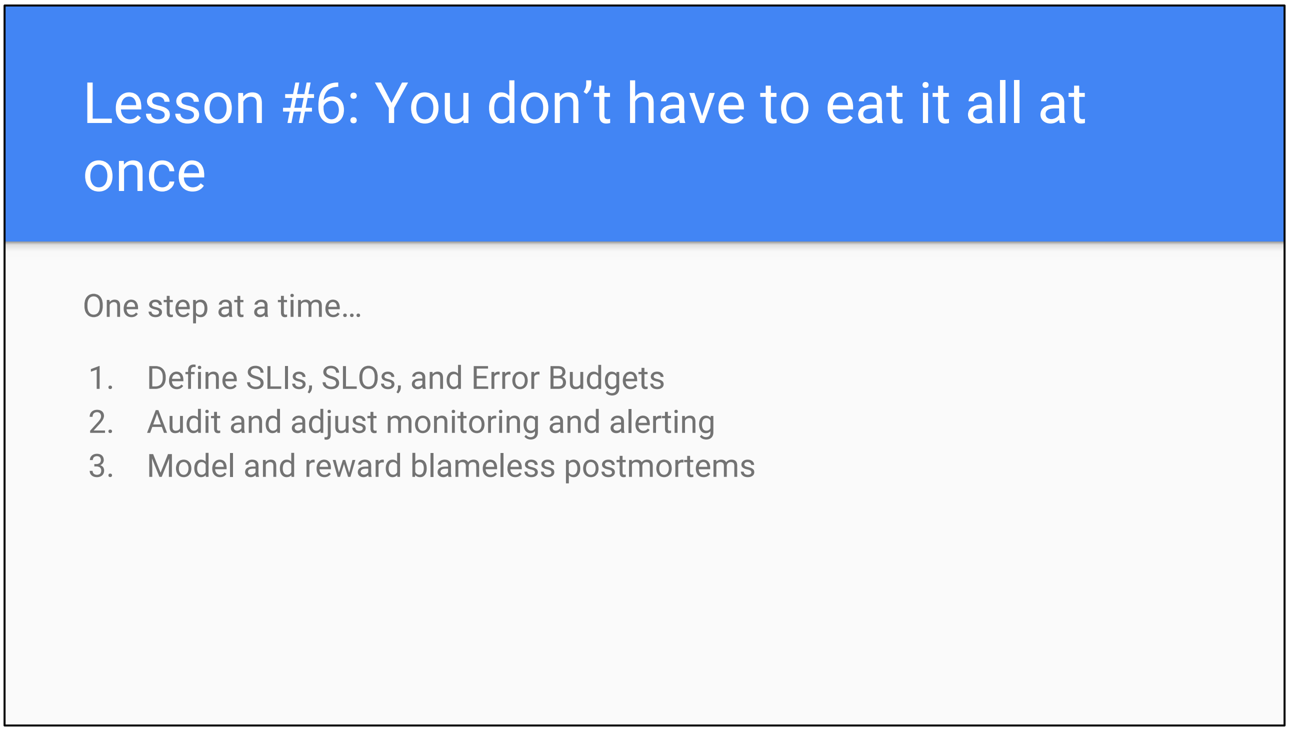
SLO	99.9%
Error Budget	525.6 min/yr

hen risk
ed*. Your*
uld describe
update your
additional
management
eamble
1; see

n Premarket Application Reviews; Guidance for Industry and FDA Staff

You can play with this tool yourself at:
https://goo.gl/bnsPj7

Learn more online and use this tool for yourself!

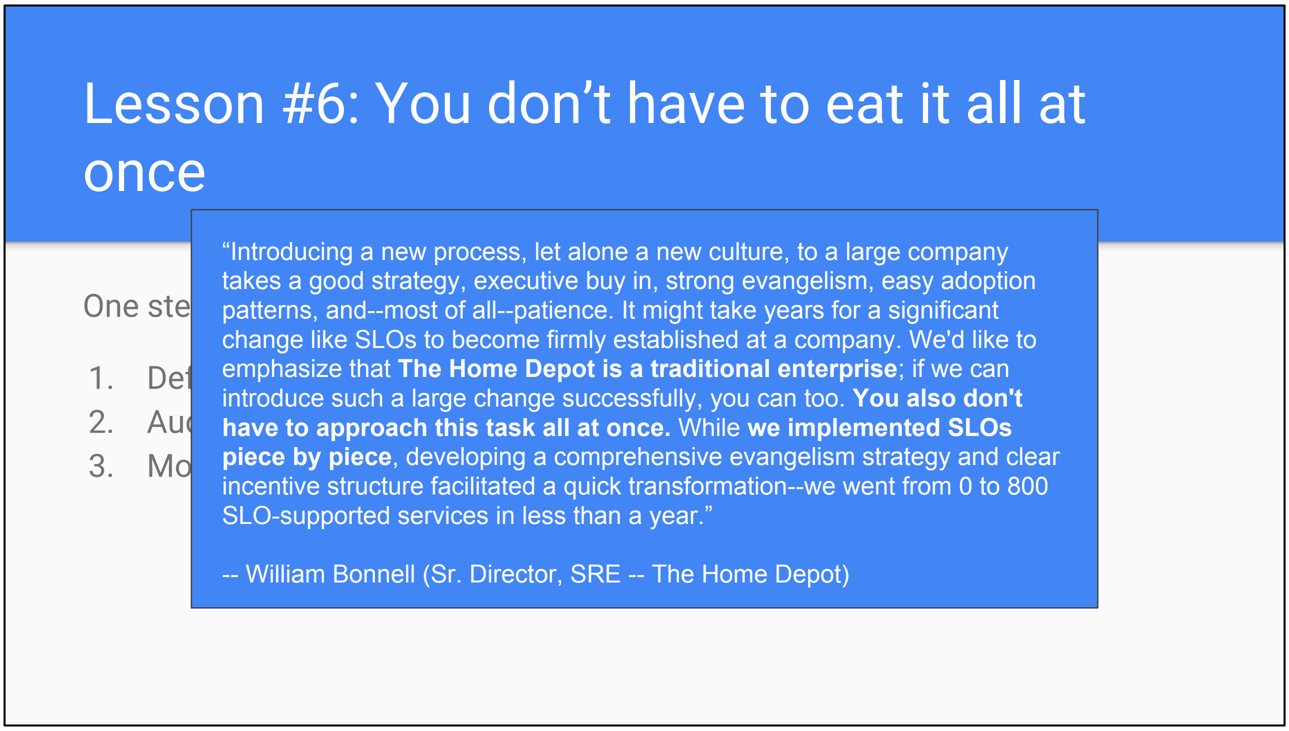


Lesson #6: You don’t have to eat it all at once

One step at a time…

1. Define SLIs, SLOs, and Error Budgets
2. Audit and adjust monitoring and alerting
3. Model and reward blameless postmortems

1) SLOs and Error Budgets are the currency of SRE. You have to start there. If that’s as far as you get, you’re still much better off.
2) Restricting your alerting to just the things that threaten the SLOs will reduce a huge amount of toil and, in turn, make your teams more productive and happier.
3) SRE is a culture as much as it is a set of principles and practices. The notion of a truly blameless postmortem is a big shift for large organizations, but it’s one they really want to take. You just need to model it for them!



Lesson #6: You don’t have to eat it all at once

One ste

1. Def
2. Au
3. Mo

“Introducing a new process, let alone a new culture, to a large company takes a good strategy, executive buy in, strong evangelism, easy adoption patterns, and--most of all--patience. It might take years for a significant change like SLOs to become firmly established at a company. We'd like to emphasize that **The Home Depot is a traditional enterprise**; if we can introduce such a large change successfully, you can too. **You also don't have to approach this task all at once.** While **we implemented SLOs piece by piece**, developing a comprehensive evangelism strategy and clear incentive structure facilitated a quick transformation--we went from 0 to 800 SLO-supported services in less than a year.”

-- William Bonnell (Sr. Director, SRE -- The Home Depot)

Again, a real customer speaks from authority.

Lesson #7: Not everyone makes it the whole way -- and that's OK!

- Any progress you make is time well spent.
- Leave lots of on/off-ramps for folks.
- It's OK to stop for a while and come back when they're ready.

One Last Thing

You can do this. They can do this. Everyone is better off.

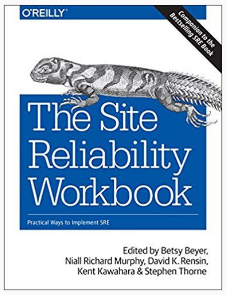
Learn More...

Know Thy Enemy: How to Prioritize and Communicate Risks

Thursday, 2:05 pm–2:45 pm -- Matt Brown, Google

Resolving Outages Faster with Better Debugging Strategies

Thursday, 9:15 am–9:55 am -- Liz Fong-Jones and Adam Mckaig, Google


O'REILLY
Companion to the Bestselling SRE Book
The Site Reliability Workbook
Practical Ways to Implement SRE
Edited by Betsy Beyer, Niall Richard Murphy, David K. Rensin, Kent Kawahara & Stephen Thorne

Q&A