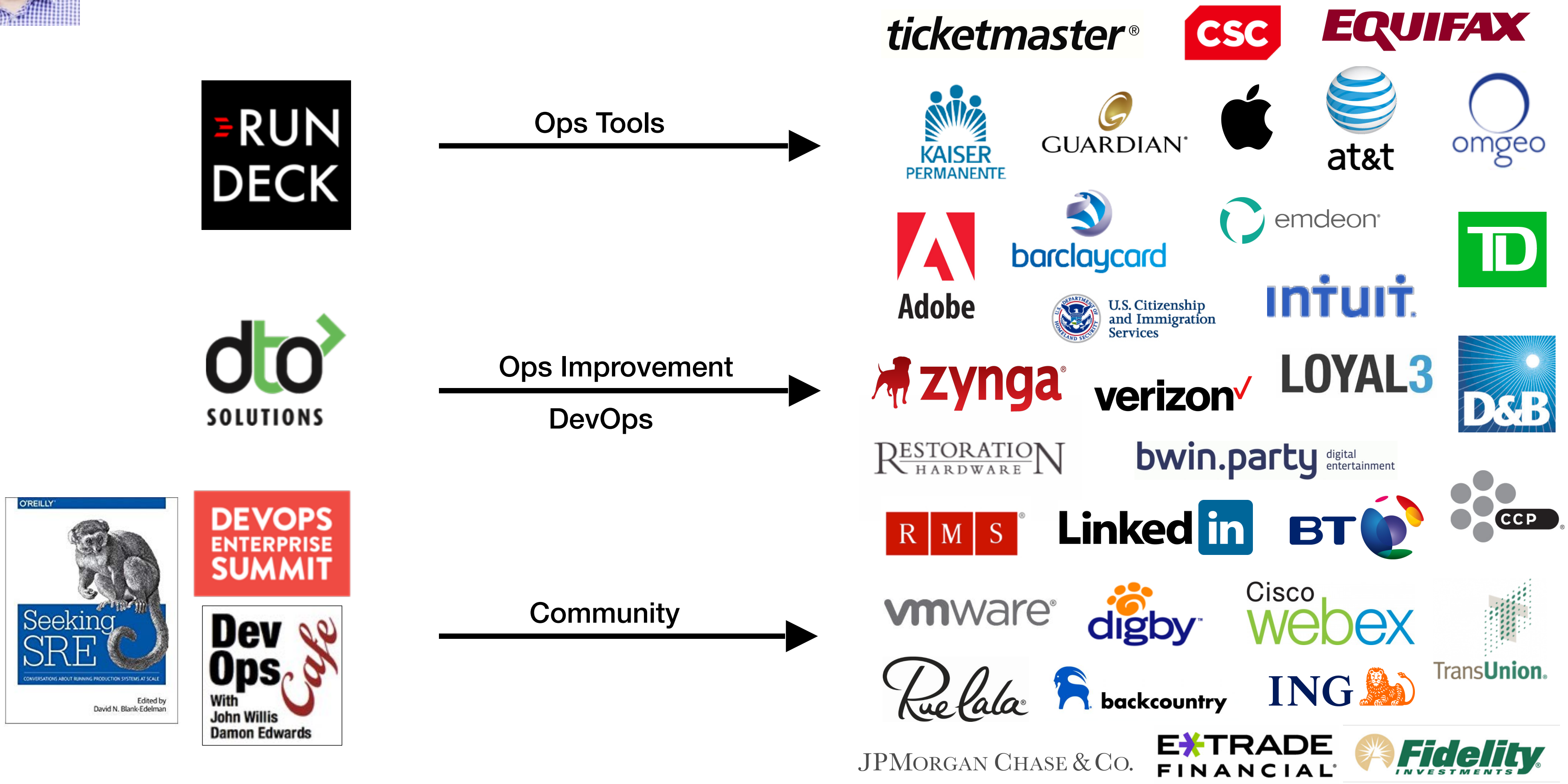
SRE CON EUROPE MIDDLE EAST AFRICA
AUGUST 29–31, 2018
DUSSELDORF, GERMANY

Clearing the Way For SRE in the Enterprise

Damon Edwards
@damonedwards

RUNDECK

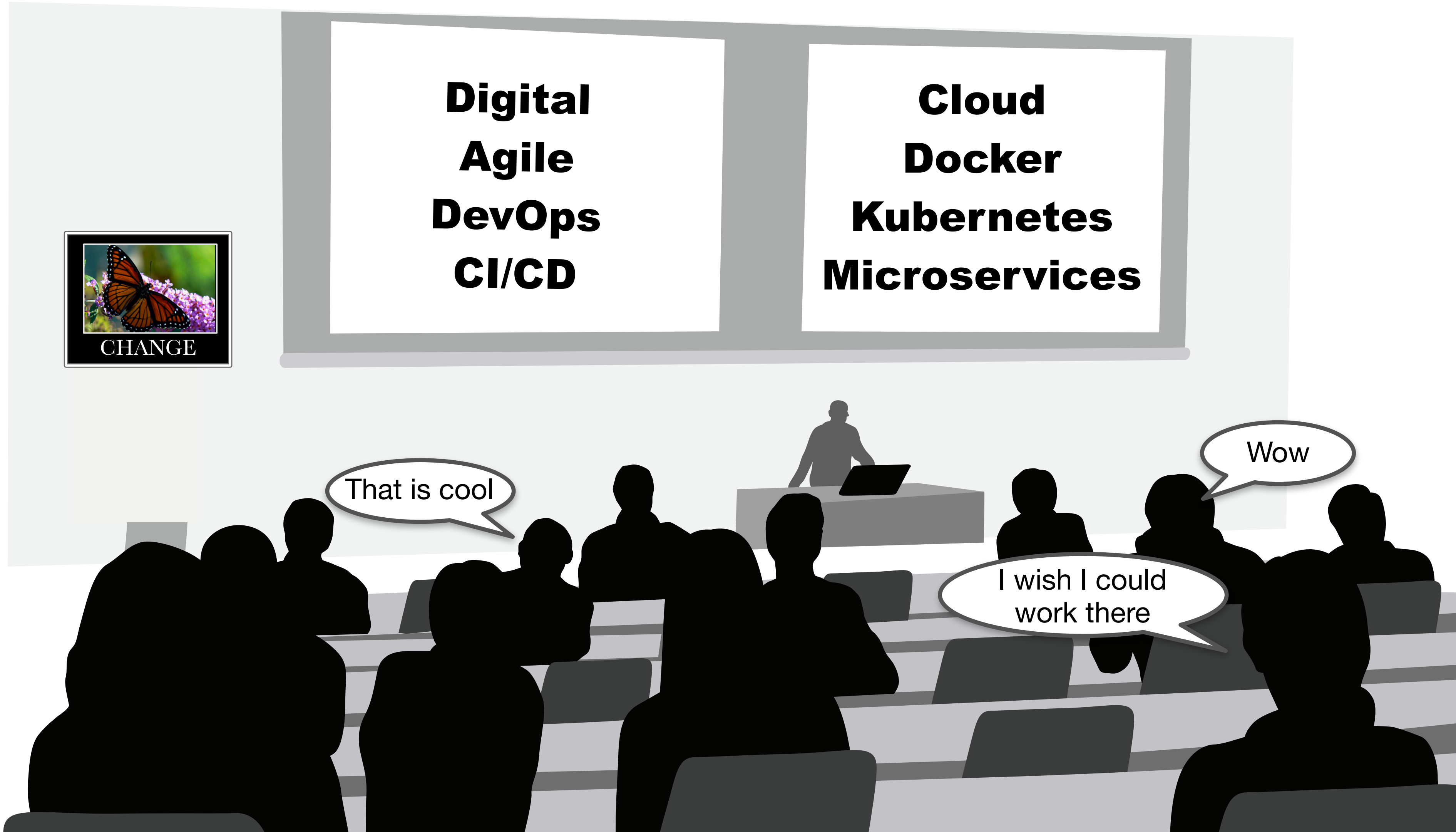
Damon Edwards



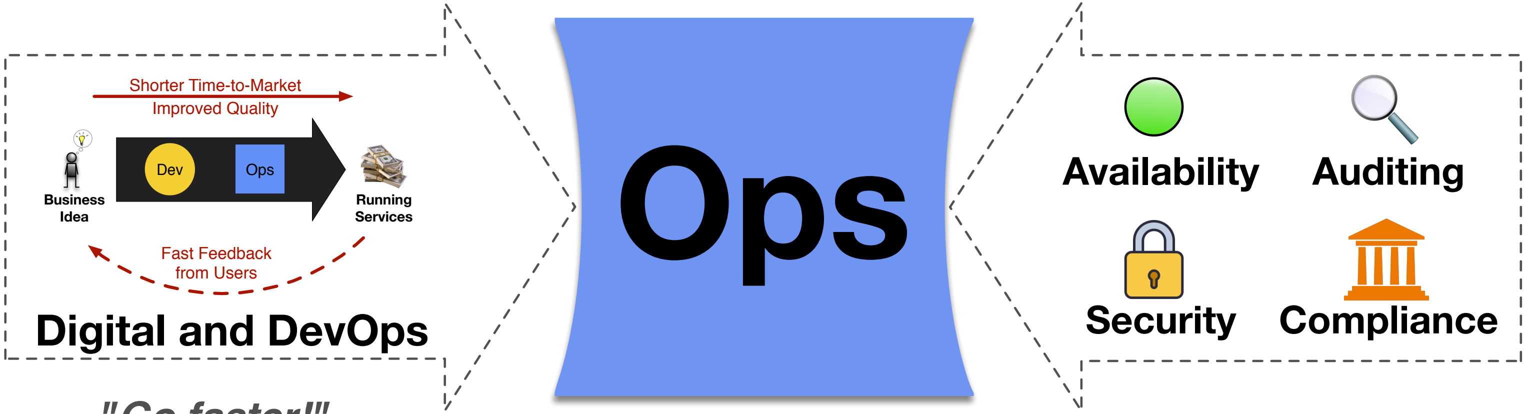
RUN DECK → Ops Tools

dto SOLUTIONS → Ops Improvement / DevOps

Seeking SRE · DEVOPS ENTERPRISE SUMMIT · DevOps Cafe → Community


Digital
Agile
DevOps
CI/CD
Cloud
Docker
Kubernetes
Microservices
CHANGE
That is cool
Wow
I wish I could work there

“Great for Dev, but what about Ops?”



Shorter Time-to-Market

Improved Quality

Business Idea

Dev

Ops

Running Services

Fast Feedback from Users

Digital and DevOps

Ops

Availability

Auditing

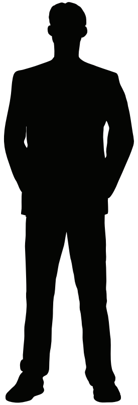
Security

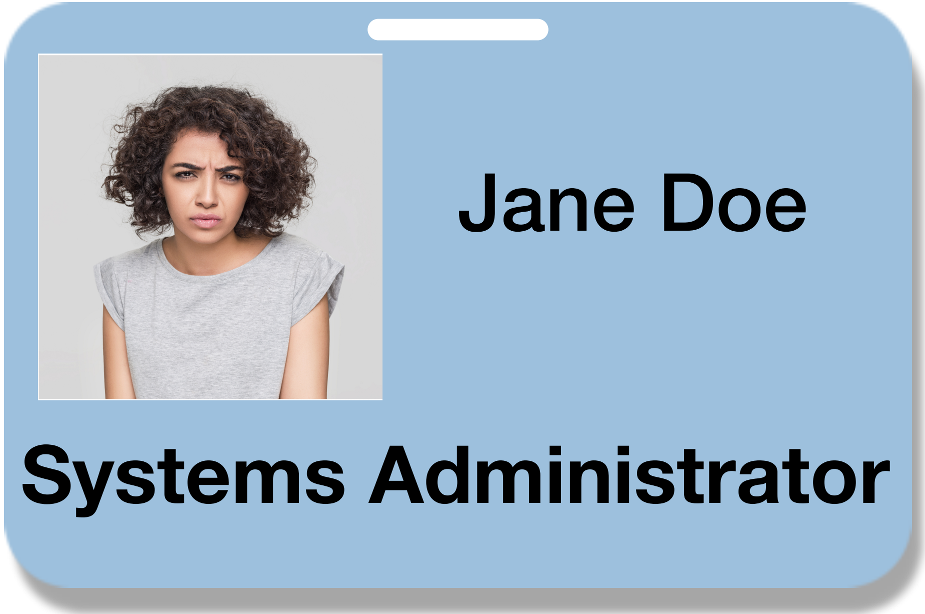
Compliance

"Go faster!"
“Open up!”

“Lock it down!”

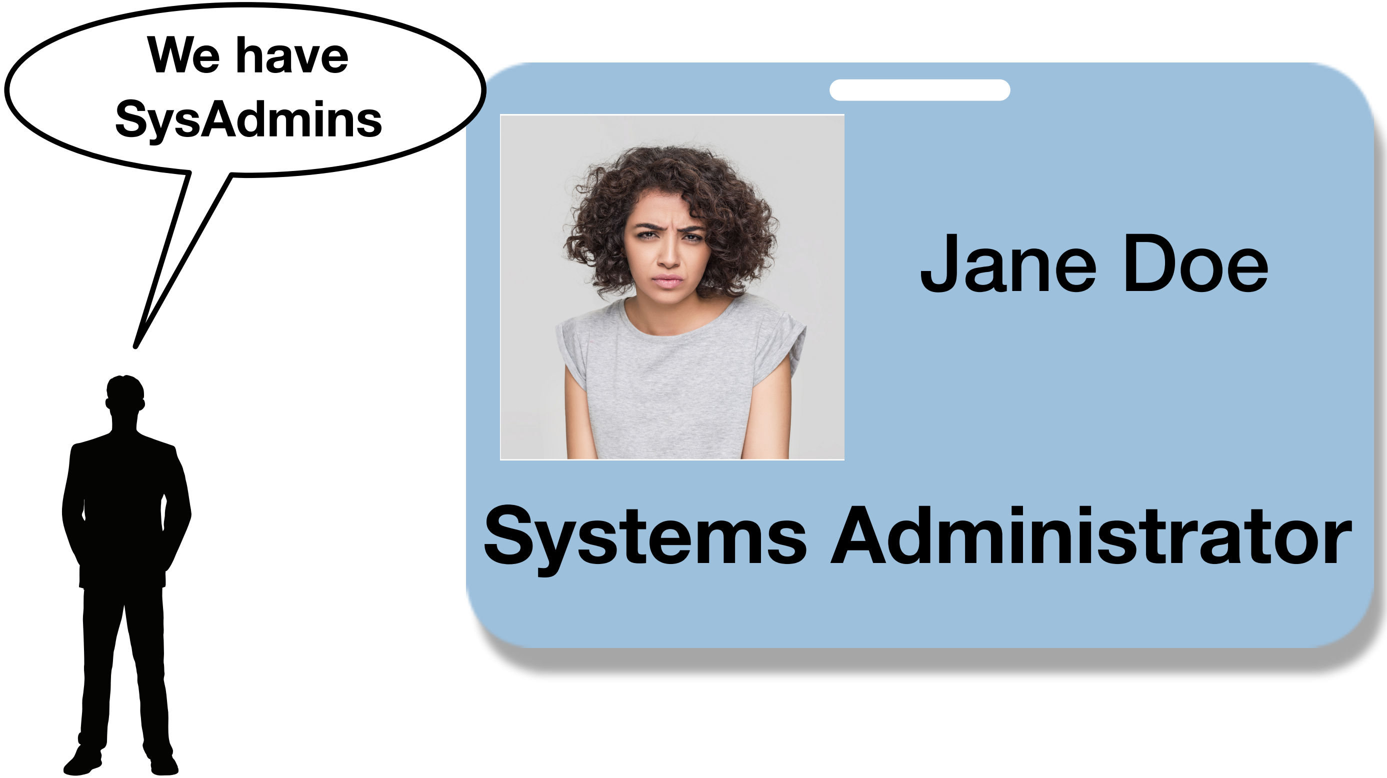
Our transformation has largely ignored Ops. Any ideas?
Have you heard of SRE? Google does it.

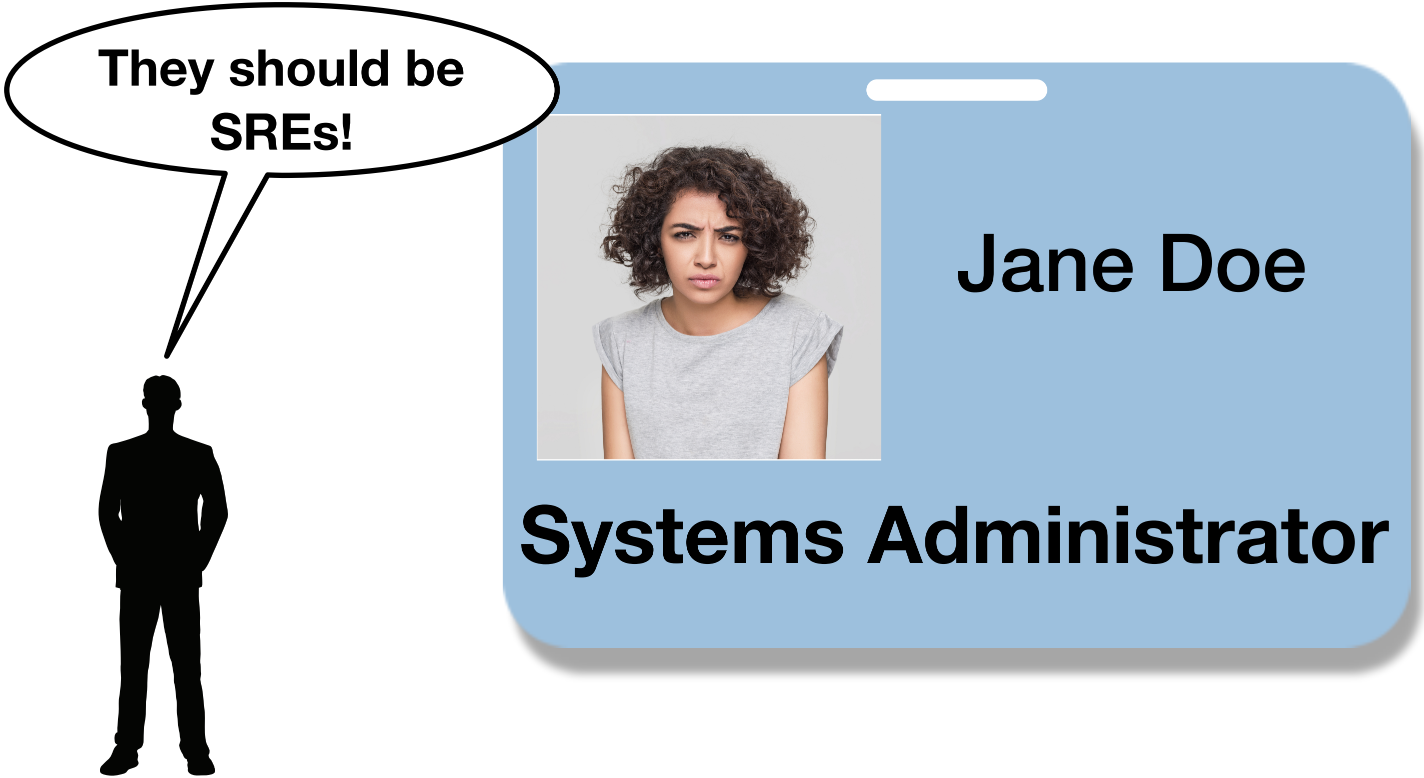


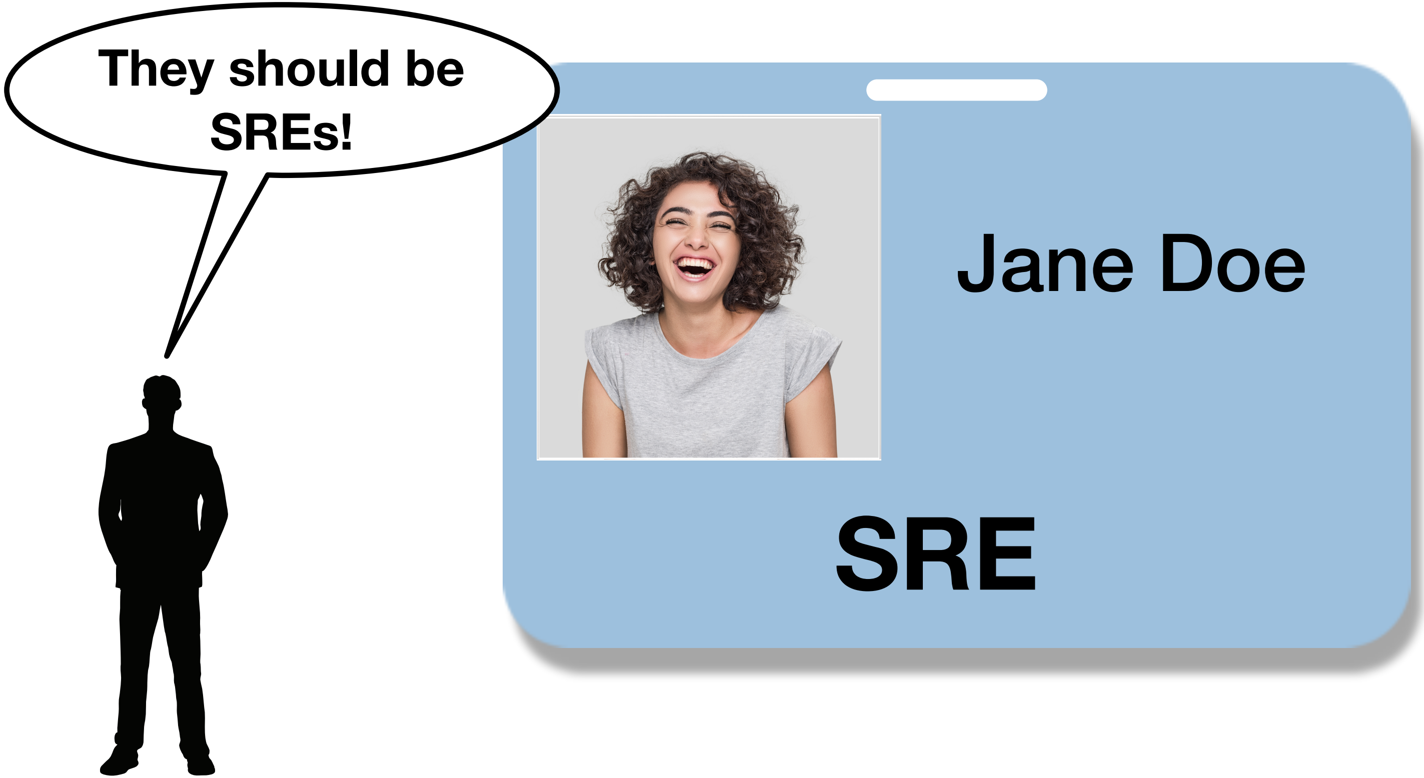


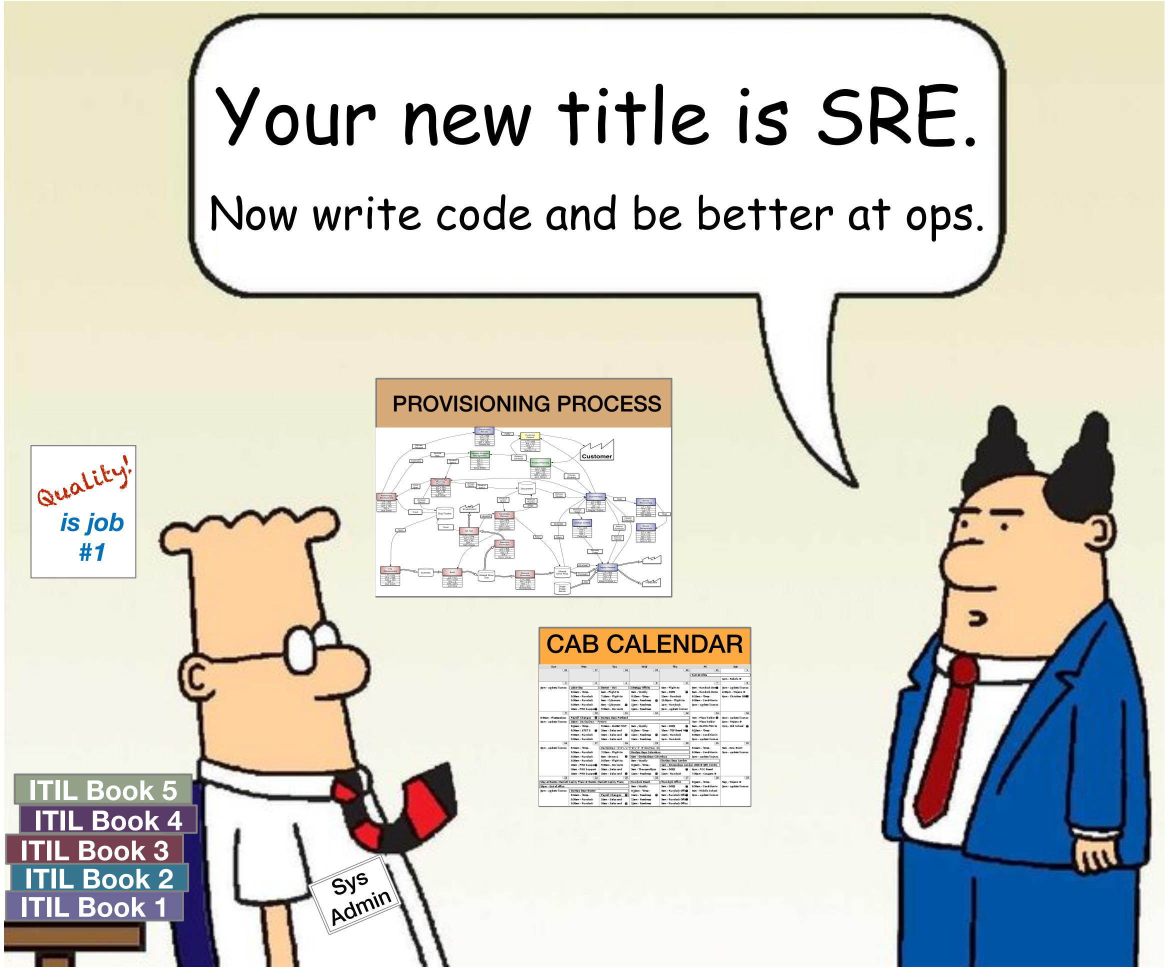
Jane Doe

Systems Administrator


We have SysAdmins
Jane Doe
Systems Administrator


They should be SREs!
Jane Doe
Systems Administrator


They should be SREs!
Jane Doe
SRE



Dilbert characters © Scott Adams www.dilbert.com

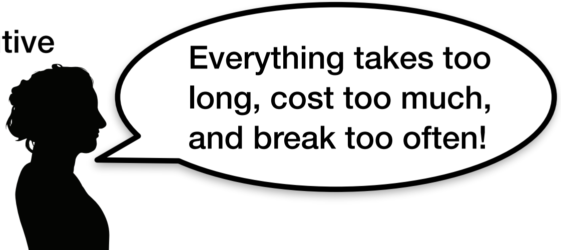
SysAdmins
Overloaded. Constant firefighting.
Waiting in ticket queues for everything.
Things break. Break again. And again.
Everyone is busy, but it doesn't get any better.

Executive View

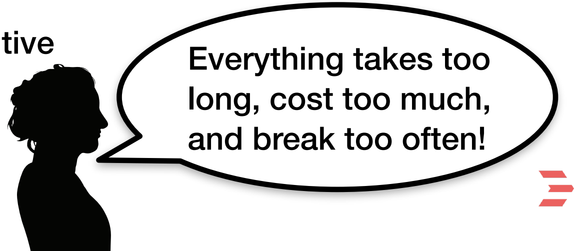
Everything takes too long, cost too much, and break too often!

SysAdmins	(False) SRE
Overloaded. Constant firefighting.	*Overloaded. Constant firefighting.*
Waiting in ticket queues for everything.	*Waiting in ticket queues for everything.*
Things break. Break again. And again.	*Things break. Break again. And again.*
Everyone is busy, but it doesn't get any better.	*Everyone is busy, but it doesn't get any better.*

Executive View


Everything takes too long, cost too much, and break too often!

Executive View


Everything takes too long, cost too much, and break too often!

Changing job titles or adding individual skills doesn’t make systems administrators SREs.

Principles of SRE are what set SRE apart

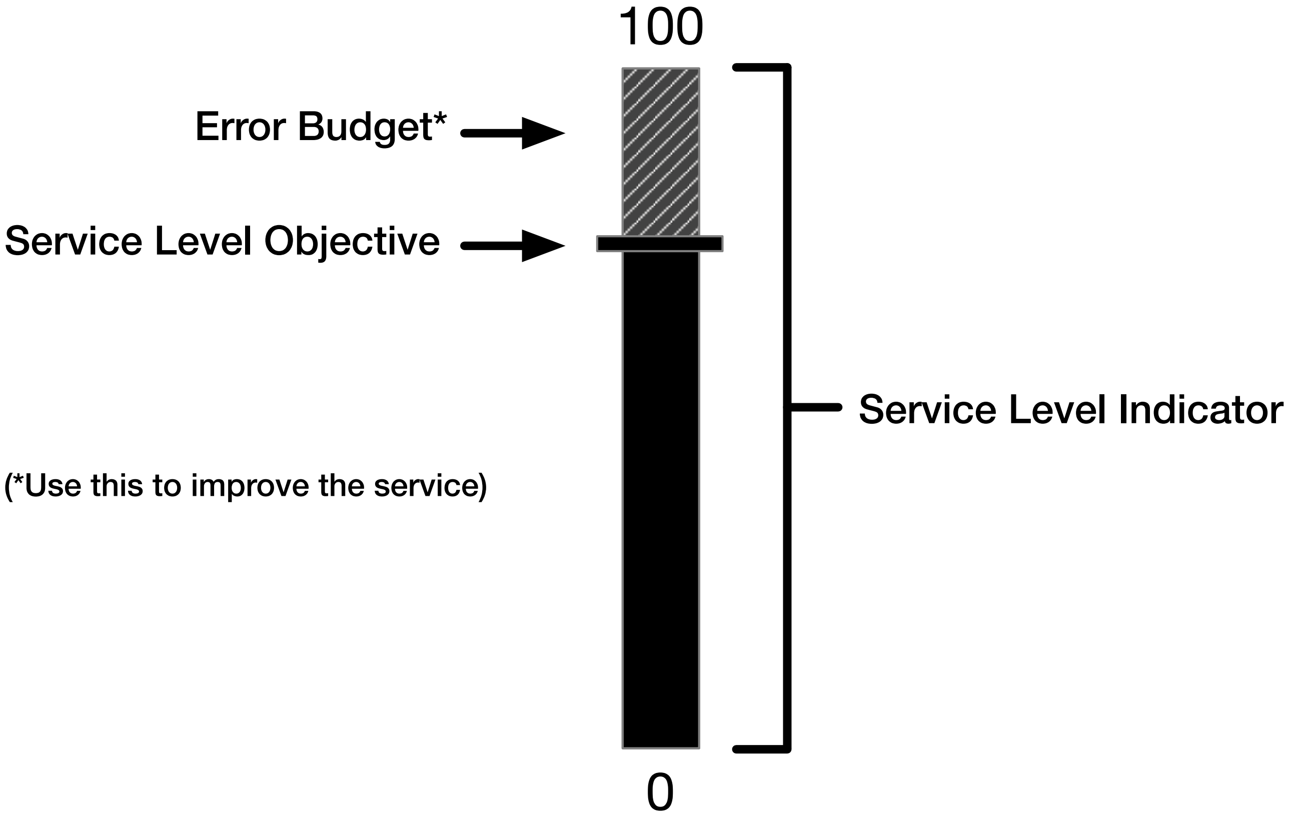
Principles of SRE are what set SRE apart

1. SRE needs Service Level Objectives, with consequences

Principles of SRE are what set SRE apart

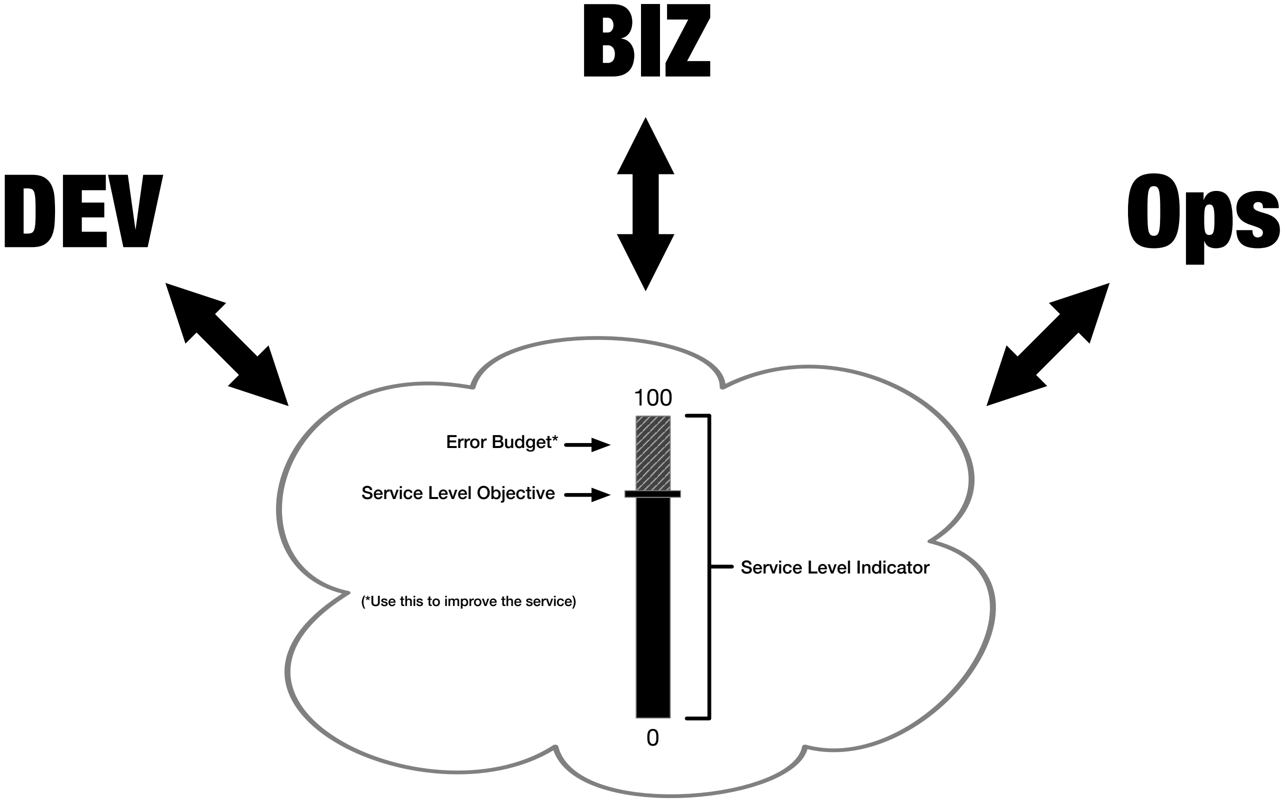
1. SRE needs Service Level Objectives, with consequences

SLO and Error Budgets: Tools for Shared Responsibility



(*Use this to improve the service)

SLO and Error Budgets: Tools for Shared Responsibility



RUNDECK

Principles of SRE are what set SRE apart

1. SRE needs Service Level Objectives, with consequences

Principles of SRE are what set SRE apart

1. SRE needs Service Level Objectives, with consequences
2. SREs have time to make tomorrow better than today

Principles of SRE are what set SRE apart

1. SRE needs Service Level Objectives, with consequences
2. SREs have time to make tomorrow better than today
3. SRE teams have the ability to regulate their workload

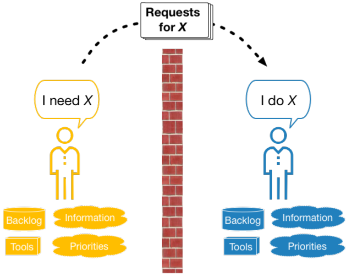
Principles of SRE are what set SRE apart

1. SRE needs Service Level Objectives, with consequences
2. SREs have time to make tomorrow better than today
3. SRE teams have the ability to regulate their workload

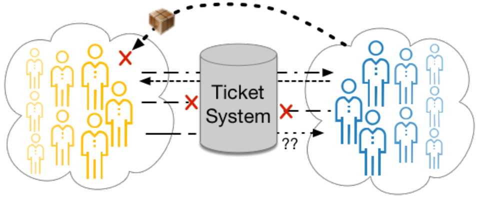
Stephen Thorne
At DevOps Enterprise Summit
London 2018 https://youtu.be/c-w_GYvi0eA

RUNDECK

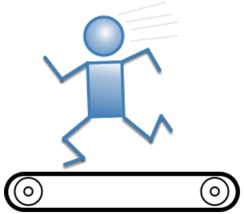
Forces That Undermine SRE Principles



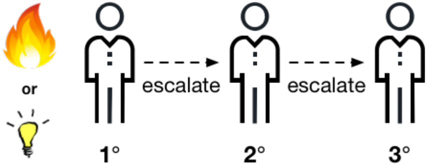
Silos



Queues

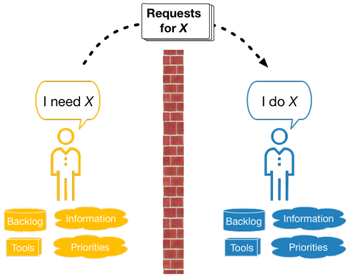


Excessive Toil

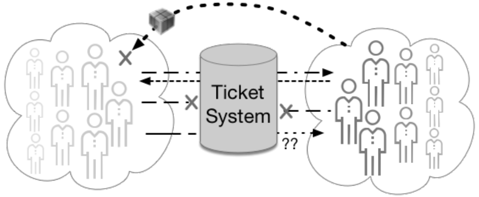


Low Trust

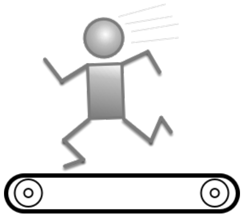
Forces That Undermine SRE Principles



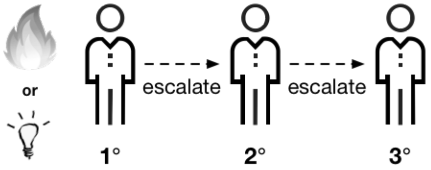
Silos



Queues

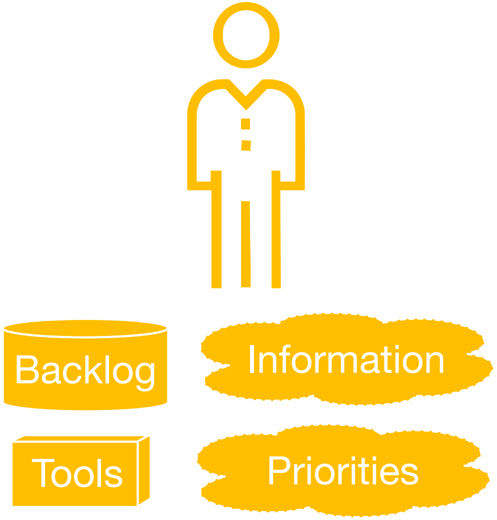


Excessive Toil

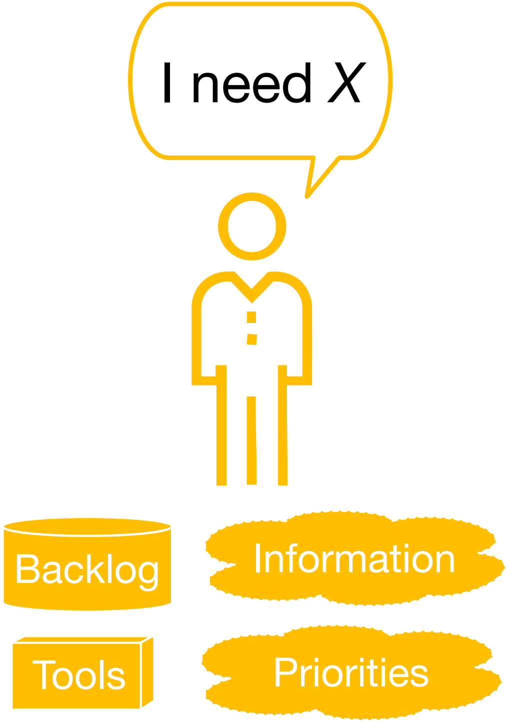


Low Trust

Silos

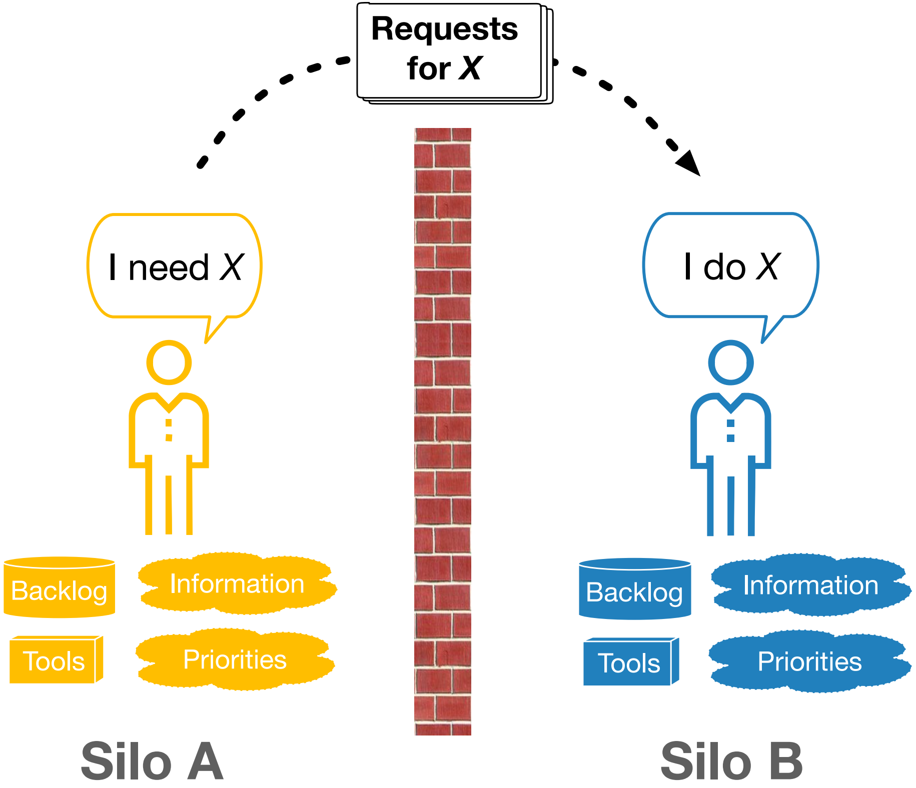


Silos



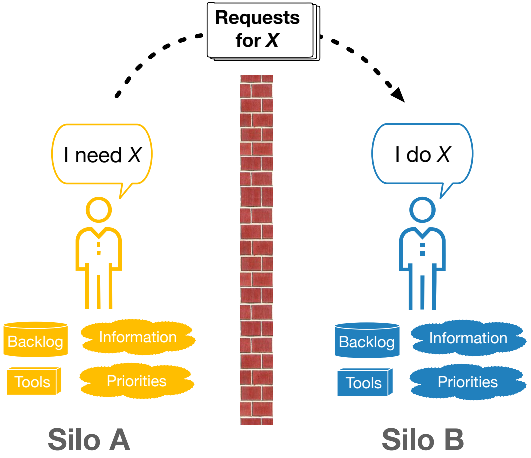
RUNDECK

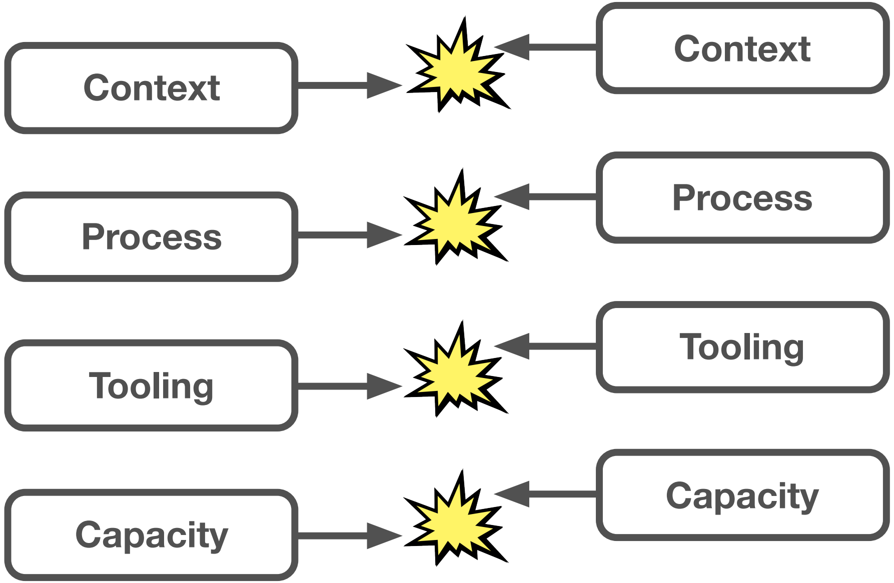
Silos



RUNDECK

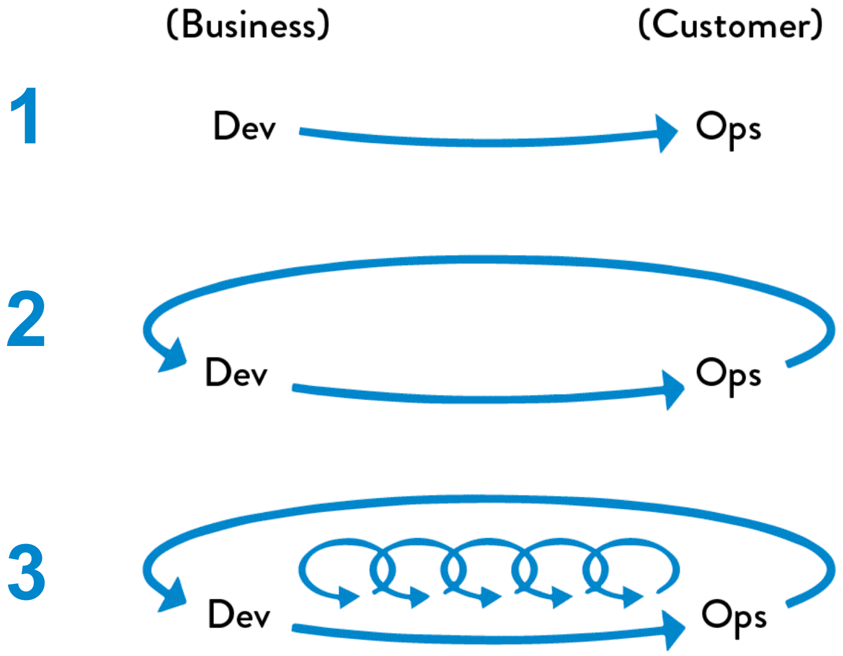
Silos cause disconnects and mismatches

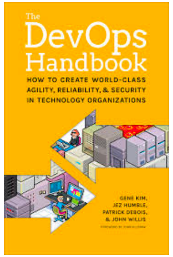




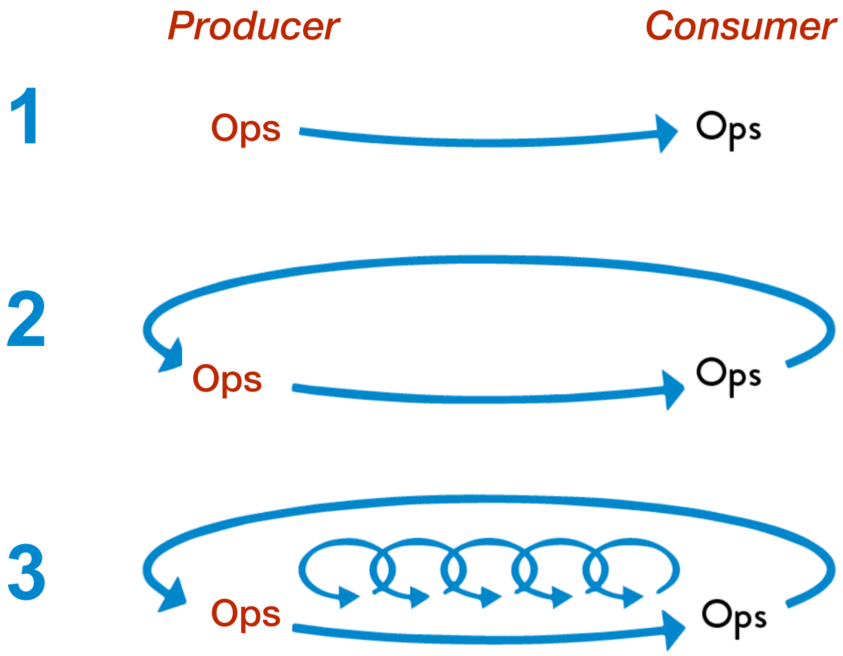
RUNDECK

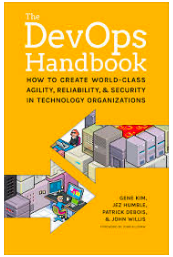
Silos Interfere with feedback loops



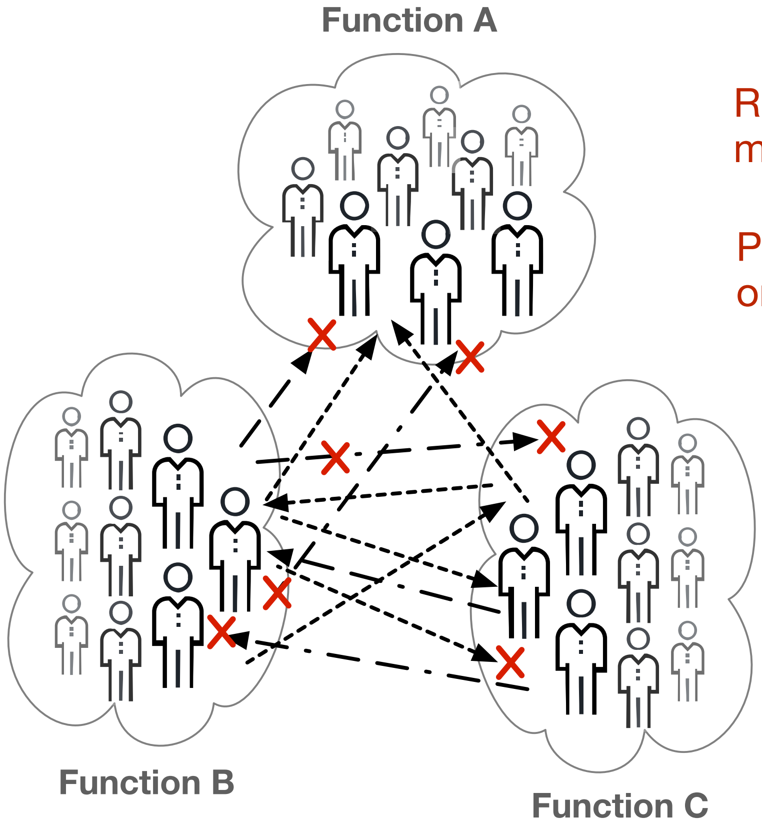

The DevOps Handbook
HOW TO CREATE WORLD-CLASS AGILITY, RELIABILITY, & SECURITY IN TECHNOLOGY ORGANIZATIONS
GENE KIM, JEZ HUMBLE, PATRICK DEBOIS, & JOHN WILLIS

Silos Interfere with feedback loops




The DevOps Handbook
HOW TO CREATE WORLD-CLASS AGILITY, RELIABILITY, & SECURITY IN TECHNOLOGY ORGANIZATIONS
GENE KIM, JEZ HUMBLE, PATRICK DEBOIS, & JOHN WILLIS

Silos create labor pools of functional specialists



Requests fulfilled by semi-manual or manual effort

Primary management focus is on protecting team capacity

Silos Undermine SRE Principles

1. Org has Service Level Objectives, with consequences?

2. SREs have time to make tomorrow better than today?

3. SRE teams have the ability to regulate their workload?

Silos Undermine SRE Principles

1. Org has Service Level Objectives, with consequences?

    ⓧ Disjointed silos make meaningful SLOs and shared responsibility almost impossible

2. SREs have time to make tomorrow better than today?

3. SRE teams have the ability to regulate their workload?

Silos Undermine SRE Principles

1. Org has Service Level Objectives, with consequences?

   ⓧ Disjointed silos make meaningful SLOs and shared responsibility almost impossible

2. SREs have time to make tomorrow better than today?

   ⓧ Siloed labor pools, disconnected processes and tools, and slow feedback loops tend to consume all available capacity

3. SRE teams have the ability to regulate their workload?

Silos Undermine SRE Principles

1. Org has Service Level Objectives, with consequences?

   ⓧ Disjointed silos make meaningful SLOs and shared responsibility almost impossible

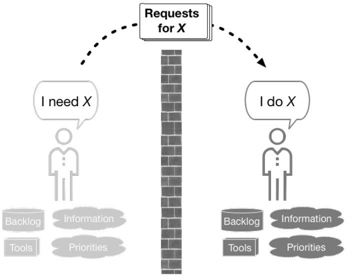
2. SREs have time to make tomorrow better than today?

   ⓧ Siloed labor pools, disconnected processes and tools, and slow feedback loops tend to consume all available capacity

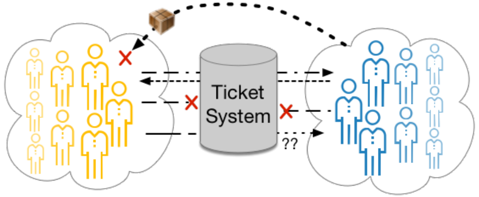
3. SRE teams have the ability to regulate their workload?

   ⓧ Struggling to keep up with demand and unable to protect capacity

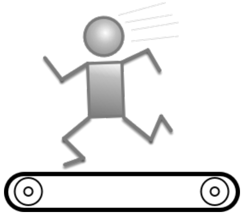
Forces That Undermine SRE Principles



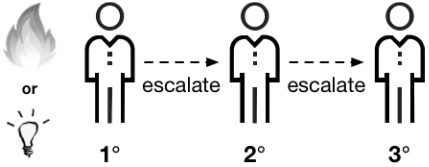
Silos



Queues



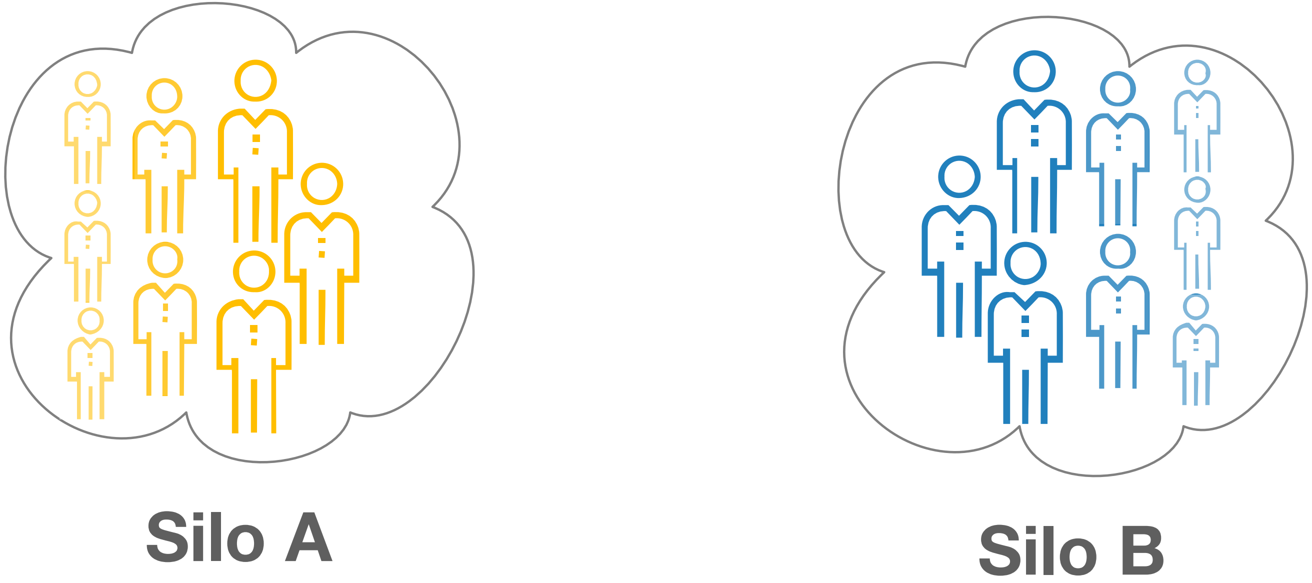
Toil



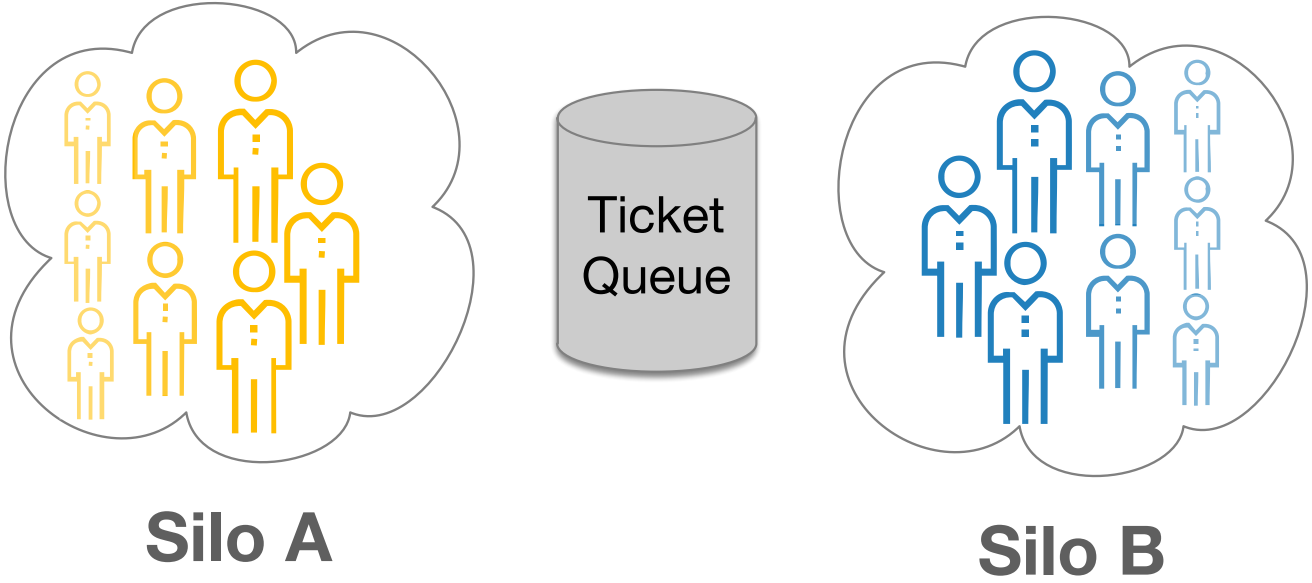
Low Trust

RUNDECK

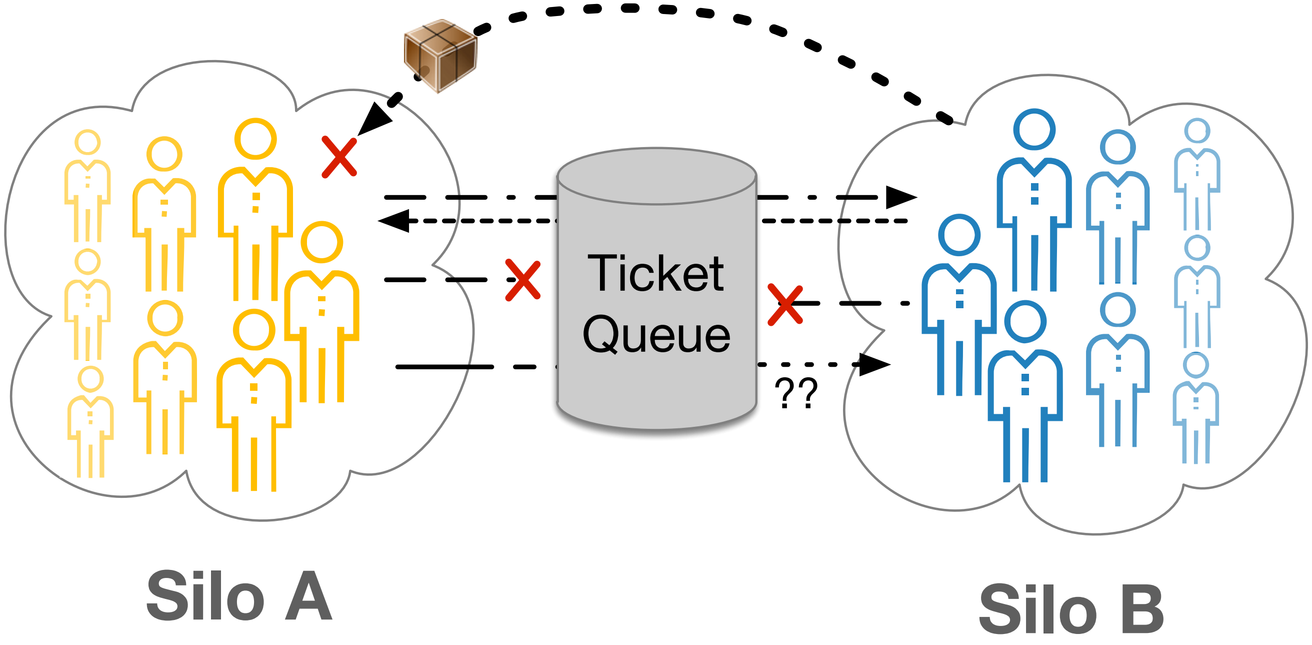
How do we cover for our cross-silo disconnects and mismatches?



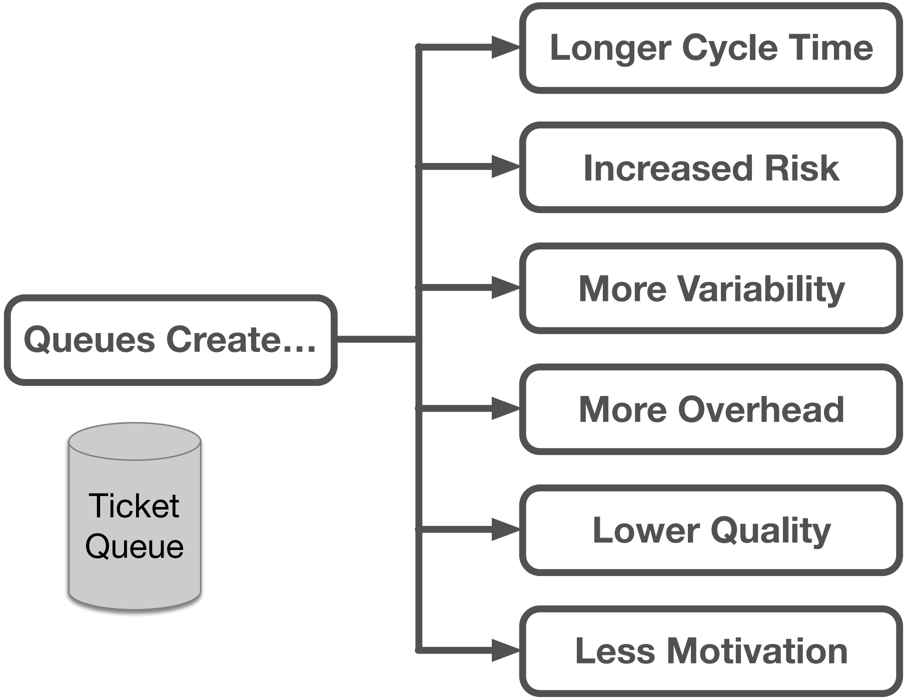
How do we cover for our cross-silo disconnects and mismatches?



We all know how well that works



Request queues are an expensive way to manage work



Queues Create...

Ticket Queue

Longer Cycle Time

Increased Risk

More Variability

More Overhead

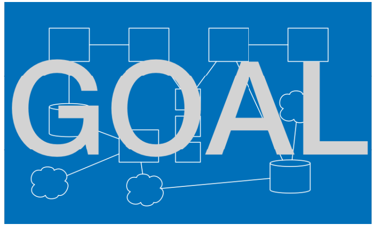
Lower Quality

Less Motivation

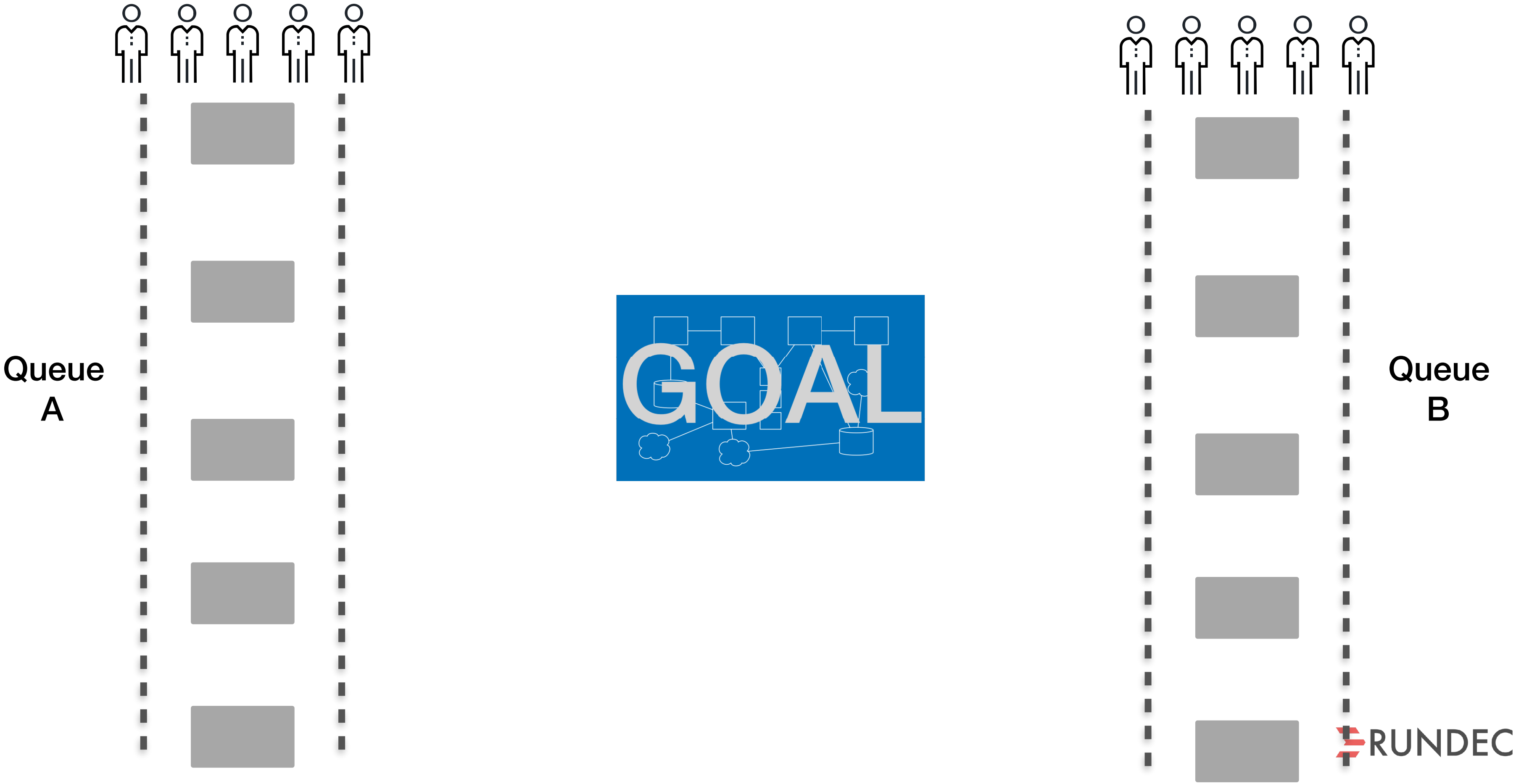
Adapted from Donald G. Reinertsen, The Principles of Product Development Flow: Second Generation Lean Product Development

RUNDECK

What do queues do to value streams?



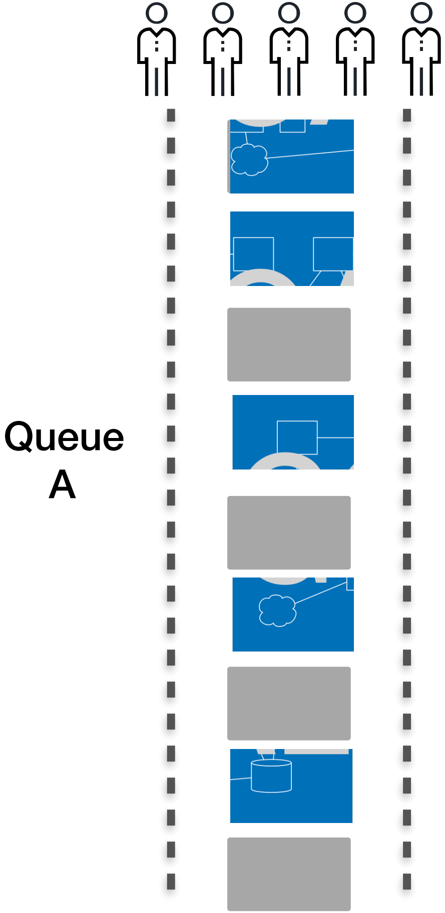
What do queues do to value streams?



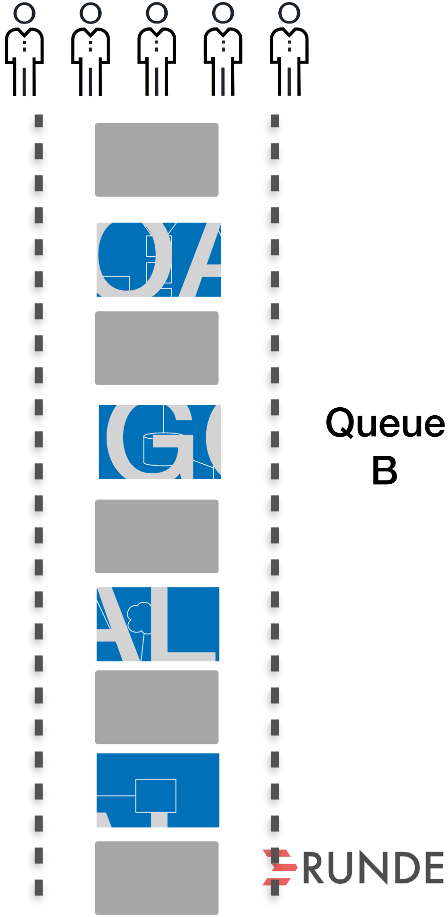
Queue A

Queue B

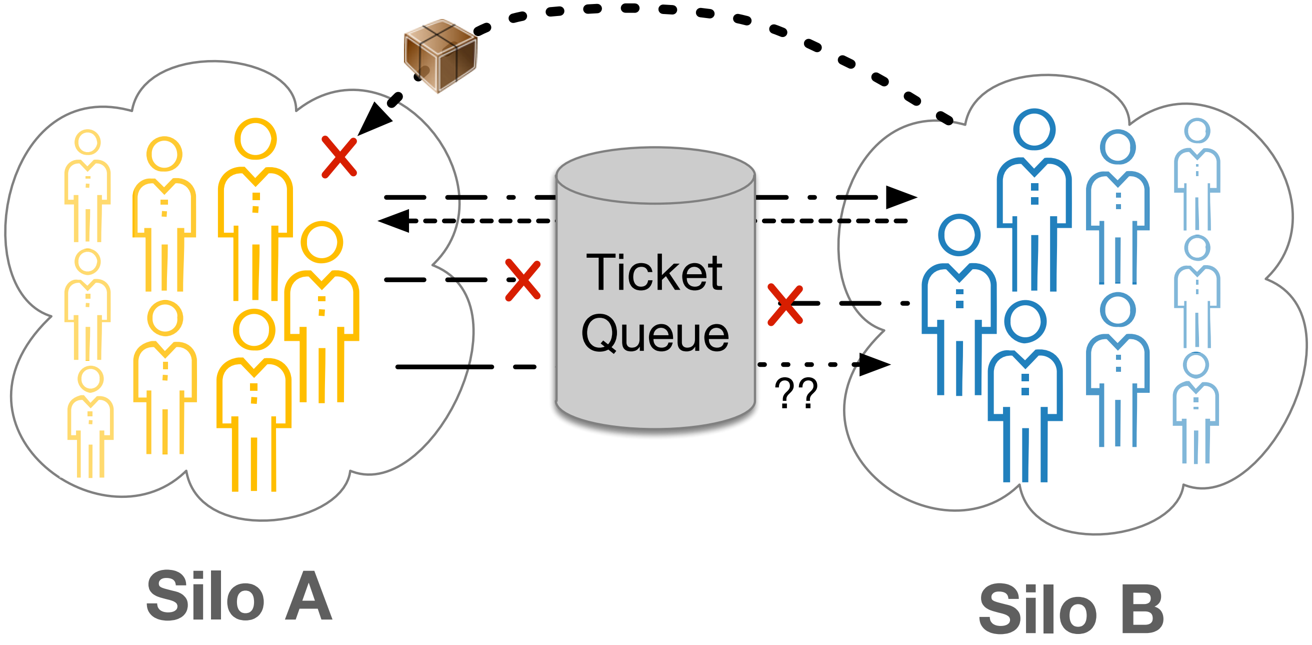
What do queues do to value streams?



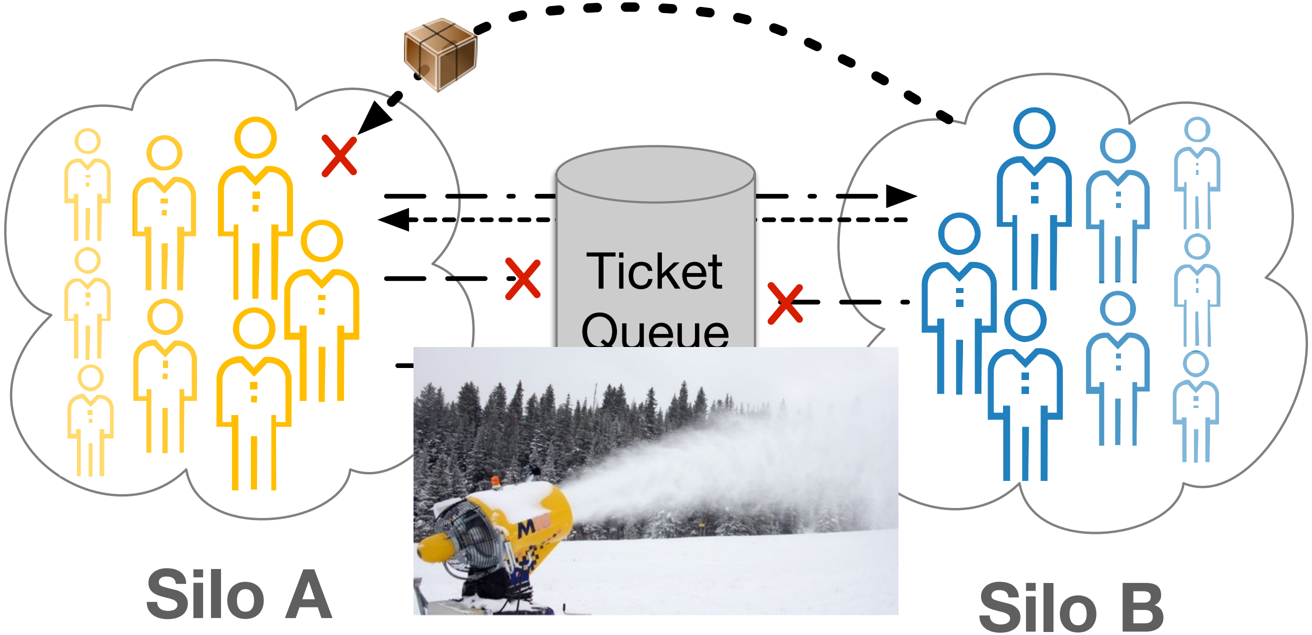
Queues disintegrate and obfuscate value streams



Tickets queues become "snowflake makers"



Tickets queues become “snowflake makers”



Snowflakes

(each unique, technically acceptable but unreproducible and brittle)

Ticket Queues Undermine SRE Principles

1. Org has Service Level Objectives, with consequences?

2. SREs have time to make tomorrow better than today?

3. SRE teams have the ability to regulate their workload?

Ticket Queues Undermine SRE Principles

1. Org has Service Level Objectives, with consequences?

   ⓧ Tickets reinforce siloed behaviors and obfuscate the value stream

2. SREs have time to make tomorrow better than today?

3. SRE teams have the ability to regulate their workload?

Ticket Queues Undermine SRE Principles

1. Org has Service Level Objectives, with consequences?

   ⓧ Tickets reinforce siloed behaviors and obfuscate the value stream

2. SREs have time to make tomorrow better than today?

   ⓧ Longer cycle time, more variability, more overhead, lower quality, and more snowflakes consume available capacity

3. SRE teams have the ability to regulate their workload?

Ticket Queues Undermine SRE Principles

1. Org has Service Level Objectives, with consequences?

   Ⓧ Tickets reinforce siloed behaviors and obfuscate the value stream

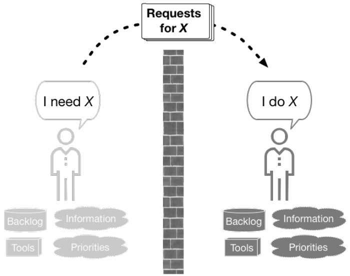
2. SREs have time to make tomorrow better than today?

   Ⓧ Longer cycle time, more variability, more overhead, lower quality, and more snowflakes consume available capacity

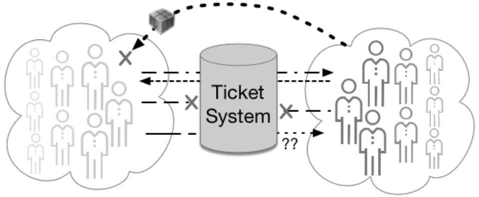
3. SRE teams have the ability to regulate their workload?

   Ⓧ Queues obfuscate the pressure being put on request fulfillers

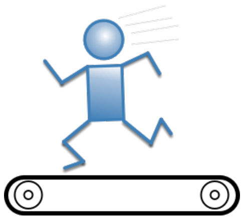
Forces That Undermine Operations



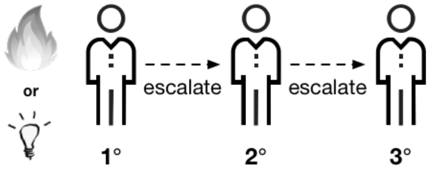
Silos



Queues

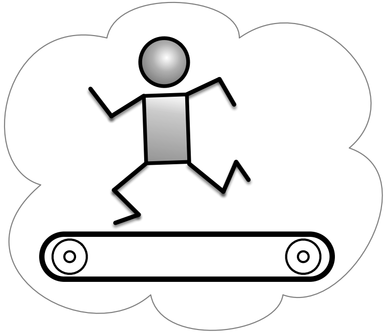


Toil

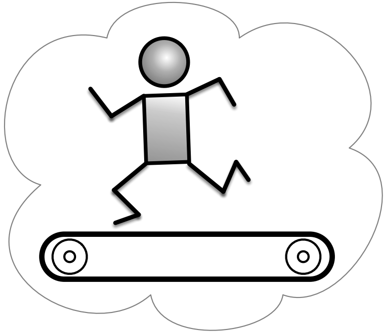


Low Trust

Toil is the enemy of SRE



Toil is the enemy of SRE



“Toil is the kind of work tied to running a production service that tends to be **manual, repetitive, automatable, tactical, devoid of enduring value, and that scales linearly** as a service grows.”

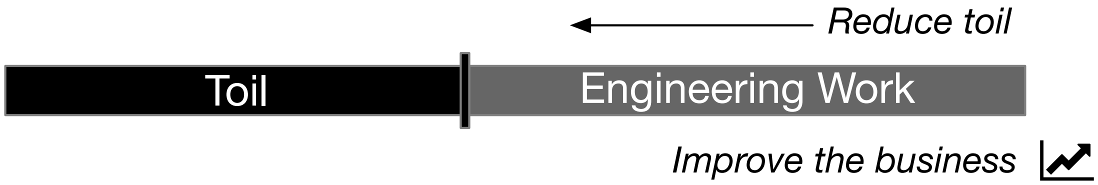
-Vivek Rau
Google

Toil vs. Engineering Work

Toil	Engineering Work
Lacks Enduring Value	Builds Enduring Value
Rote, Repetitive	Creative, Iterative
Tactical	Strategic
Increases With Scale	Enables Scaling
Can Be Automated	Requires Human Creativity

Excessive toil prevents fixing the system

Toil at manageable percentage of capacity



Reduce toil

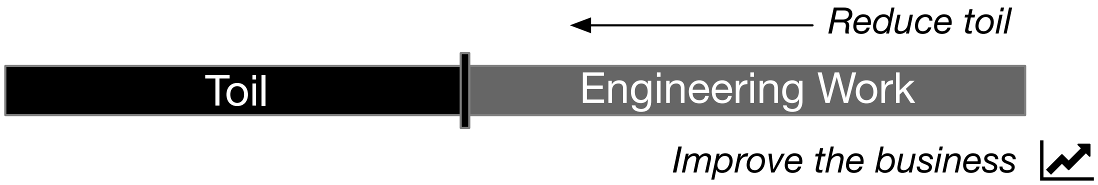
Toil

Engineering Work

Improve the business

Excessive toil prevents fixing the system

Toil at manageable percentage of capacity



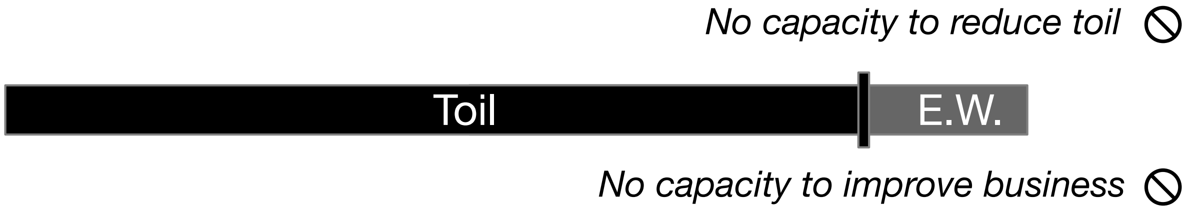
Reduce toil

Toil

Engineering Work

Improve the business 📈

Toil at unmanageable percentage of capacity (“Engineering Bankruptcy”)



No capacity to reduce toil 🛇

Toil

E.W.

No capacity to improve business 🛇

Excessive Toil Undermines SRE Principles

1. Org has Service Level Objectives, with consequences?

2. SREs have time to make tomorrow better than today?

3. SRE teams have the ability to regulate their workload?

Excessive Toil Undermines SRE Principles

1. Org has Service Level Objectives, with consequences?

   ⓧ Buried in toil keeps team from contributing engineering work to uphold their end of the shared responsibility deal

2. SREs have time to make tomorrow better than today?

3. SRE teams have the ability to regulate their workload?

RUNDECK

Excessive Toil Undermines SRE Principles

1. Org has Service Level Objectives, with consequences?

   ⓧ Buried in toil keeps team from contributing engineering work to uphold their end of the shared responsibility deal

2. SREs have time to make tomorrow better than today?

   ⓧ Buried in toil… no capacity for engineering work to reduce toil.

3. SRE teams have the ability to regulate their workload?

Excessive Toil Undermines SRE Principles

1. Org has Service Level Objectives, with consequences?

   ⓧ Buried in toil keeps team from contributing engineering work to uphold their end of the shared responsibility deal

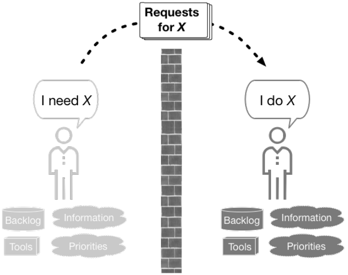
2. SREs have time to make tomorrow better than today?

   ⓧ Buried in toil… no capacity for engineering work to reduce toil.

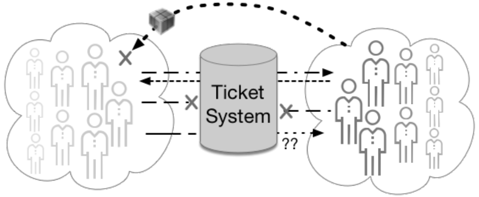
3. SRE teams have the ability to regulate their workload?

   ⓧ Buried in toil… no capacity for engineering work to reduce toil.

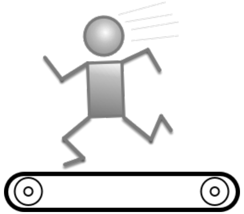
Forces That Undermine Operations



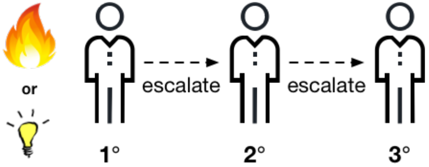
Silos



Queues



Toil

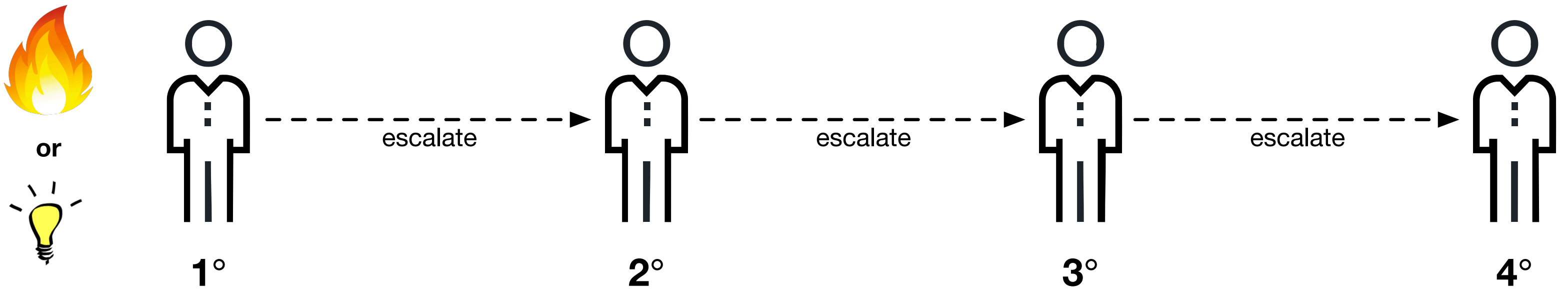


Low Trust

RUNDECK

Where are decisions made? Who can take action?

Decisions made here



RUNDECK

All work is contextual

John
Allspaw

RUNDECK

All work is contextual

```
rm -rf $PATHNAME
```

John
Allspaw

All work is contextual

```
rm -rf $PATHNAME
```

Is this dangerous?

John
Allspaw

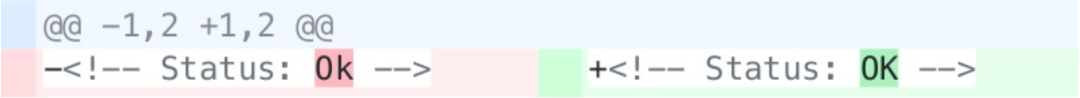
All work is contextual

```
rm -rf $PATHNAME
```

John
Allspaw

All work is contextual

```
rm -rf $PATHNAME
```



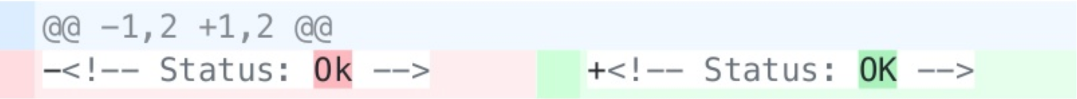
```
@@ -1,2 +1,2 @@
-<!-- Status: Ok -->          +<!-- Status: OK -->
```

John
Allspaw

RUNDECK

All work is contextual

```
rm -rf $PATHNAME
```



```
@@ -1,2 +1,2 @@
-<!-- Status: Ok -->          +<!-- Status: OK -->
```

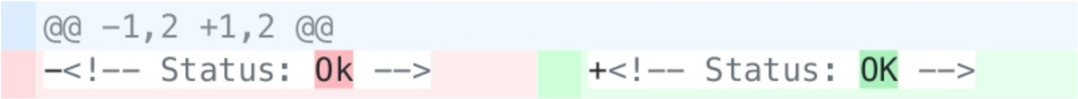
Is this dangerous?

John
Allspaw

RUNDECK

All work is contextual

```
rm -rf $PATHNAME
```



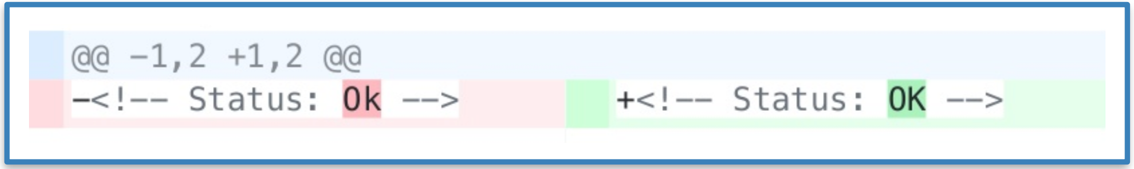
```
@@ -1,2 +1,2 @@
-<!-- Status: Ok -->          +<!-- Status: OK -->
```

John
Allspaw

RUNDECK

All work is contextual

```
rm -rf $PATHNAME
```

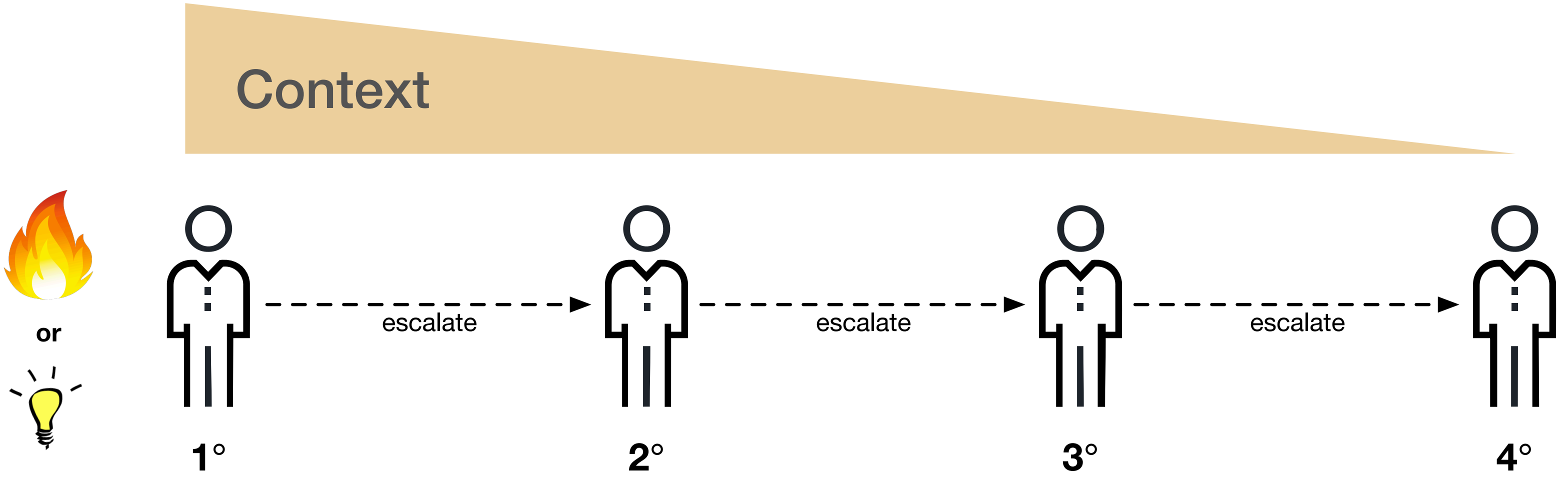


```
@@ -1,2 +1,2 @@
-<!-- Status: Ok -->          +<!-- Status: OK -->
```

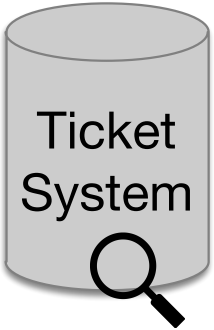
Answer is always
“it depends”

John
Allspaw

Where are decisions made? Who can take action?

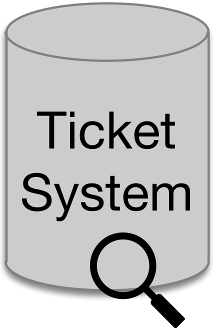


Low trust + approvals = illusion of control



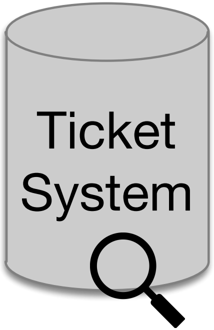
Low trust + approvals = illusion of control

Add up the total number of approval requests and



Ticket System

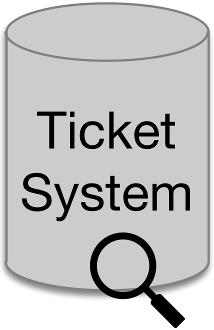
Low trust + approvals = illusion of control



Add up the total number of approval requests and

…subtract the info radiators ("I need to be in the loop")

Low trust + approvals = illusion of control



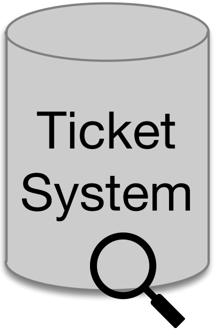
Ticket System

Add up the total number of approval requests and

…subtract the info radiators (“I need to be in the loop”)

…subtract the CYAs (“Prove you followed the process”)

Low trust + approvals = illusion of control



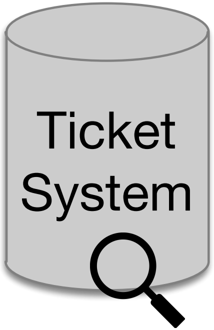
Add up the total number of approval requests and

…subtract the info radiators ("I need to be in the loop")

…subtract the CYAs ("Prove you followed the process")

…subtract the too removed to judge ("mostly guessing")

Low trust + approvals = illusion of control



Ticket System

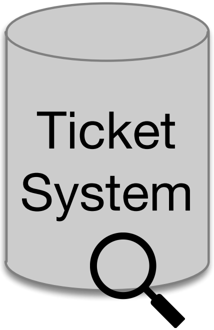
Add up the total number of approval requests and

…subtract the info radiators ("I need to be in the loop")

…subtract the CYAs ("Prove you followed the process")

…subtract the too removed to judge ("mostly guessing")

Low trust + approvals = illusion of control



Add up the total number of approval requests and

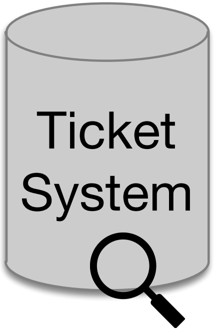
…subtract the info radiators (“I need to be in the loop”)

…subtract the CYAs (“Prove you followed the process”)

…subtract the too removed to judge (“mostly guessing”)

RUNDECK

Low trust + approvals = illusion of control



Add up the total number of approval requests and

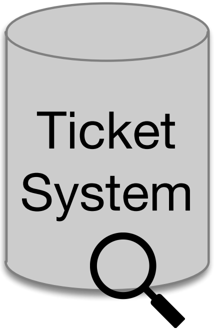
…subtract the info radiators ("I need to be in the loop")

…subtract the CYAs ("Prove you followed the process")

…subtract the too removed to judge ("mostly guessing")

How many are you left with?

Low trust + approvals = illusion of control



Add up the total number of approval requests and

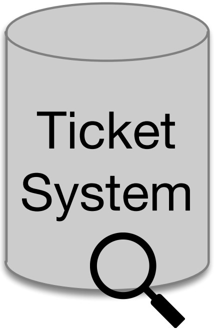
…subtract the info radiators ("I need to be in the loop")

…subtract the CYAs ("Prove you followed the process")

…subtract the too removed to judge ("mostly guessing")

How many are you left with?
How many were the right call?

Low trust + approvals = illusion of control



Add up the total number of approval requests and

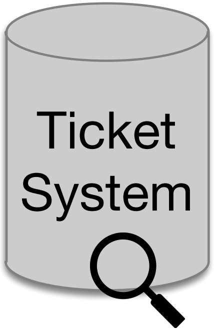
…subtract the info radiators ("I need to be in the loop")

…subtract the CYAs ("Prove you followed the process")

…subtract the too removed to judge ("mostly guessing")

How many are you left with?
How many were the right call?
How many got rejected?

Low trust + approvals = illusion of control



Add up the total number of approval requests and

…subtract the info radiators ("I need to be in the loop")

…subtract the CYAs ("Prove you followed the process")

…subtract the too removed to judge ("mostly guessing")

How many are you left with?
How many were the right call?
How many got rejected?

Low Trust Undermines SRE Principles

1. Org has Service Level Objectives, with consequences?

2. SREs have time to make tomorrow better than today?

3. SRE teams have the ability to regulate their workload?

Low Trust Undermines SRE Principles

1. Org has Service Level Objectives, with consequences?

    ⓧ Cultures of low trust have a really difficult time with shared responsibility

2. SREs have time to make tomorrow better than today?

3. SRE teams have the ability to regulate their workload?

Low Trust Undermines SRE Principles

1. Org has Service Level Objectives, with consequences?

   ⓧ Cultures of low trust have a really difficult time with shared responsibility

2. SREs have time to make tomorrow better than today?

   ⓧ People closest to problems know what to fix but tasking, priorities, and decisions are largely out of their control

3. SRE teams have the ability to regulate their workload?

Low Trust Undermines SRE Principles

1. Org has Service Level Objectives, with consequences?

   ⓧ Cultures of low trust have a really difficult time with shared responsibility

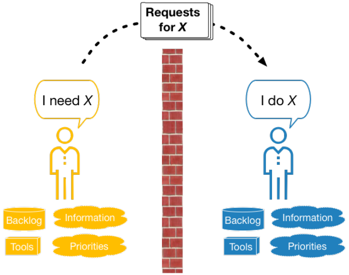
2. SREs have time to make tomorrow better than today?

   ⓧ People closest to problems know what to fix but tasking, priorities, and decisions are largely out of their control

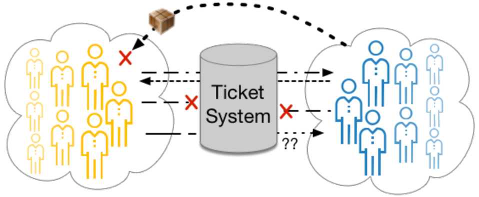
3. SRE teams have the ability to regulate their workload?

   ⓧ People aren’t trusted to plan or design their own work

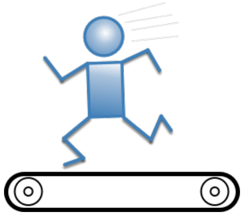
Forces That Undermine Operations



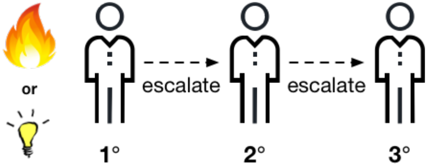
Silos



Queues



Toil



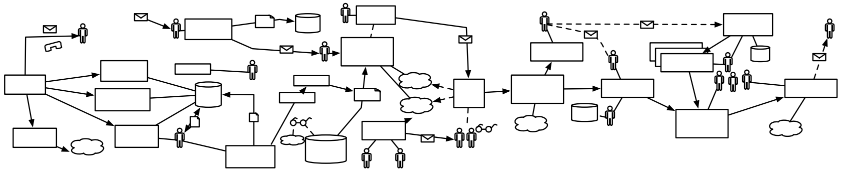
Low Trust

RUNDECK

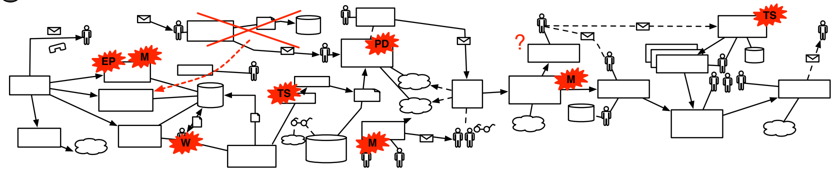
So what can we do differently?

Lean on Lean to find what to fix

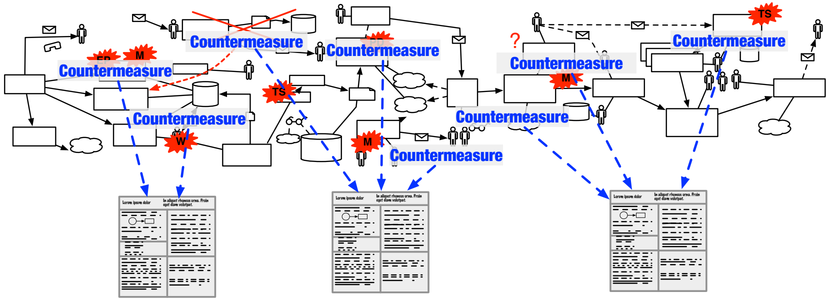
❶ **Map the end-to-end flow of information and artifacts (using a recent delivery or event)**



❷ **Identify what slows lead times, undermines quality, and impacts flow**

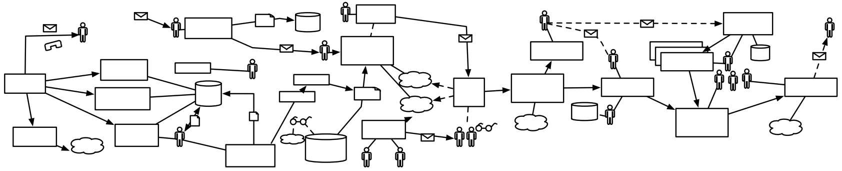


❸ **Identify countermeasures and create improvement storyboards (justification/plan)**

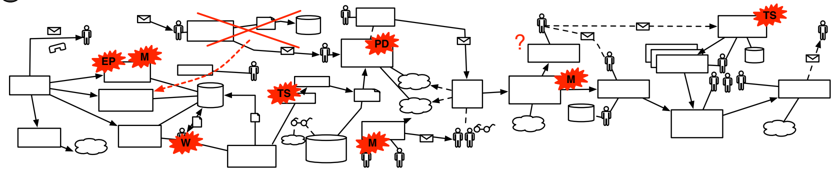


Lean on Lean to find what to fix

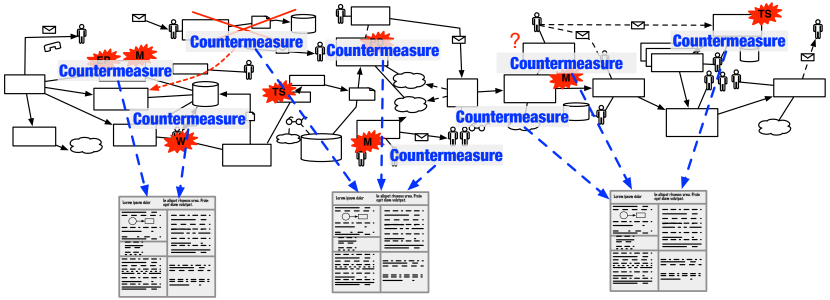
1 Map the end-to-end flow of information and artifacts (using a recent delivery or event)



2 Identify what slows lead times, undermines quality, and impacts flow



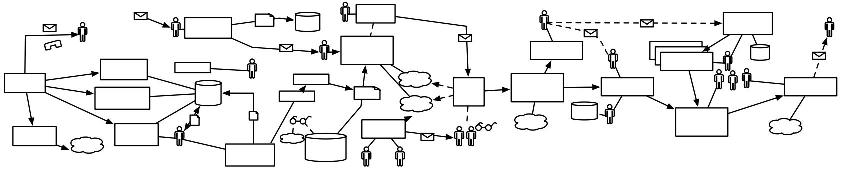
3 Identify countermeasures and create improvement storyboards (justification/plan)



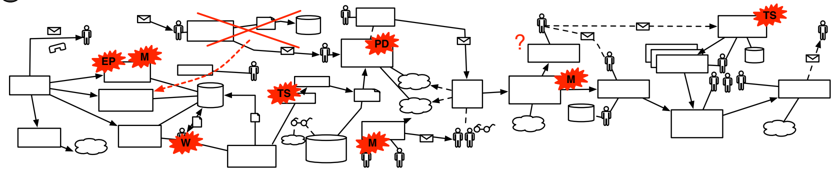
☑ All processes should be studied with an improvement disciple

Lean on Lean to find what to fix

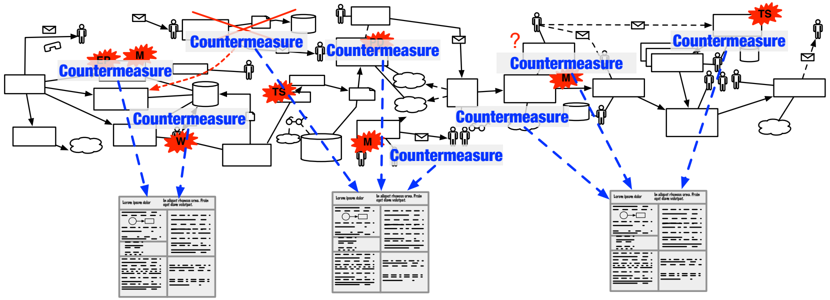
1 Map the end-to-end flow of information and artifacts (using a recent delivery or event)



2 Identify what slows lead times, undermines quality, and impacts flow



3 Identify countermeasures and create improvement storyboards (justification/plan)

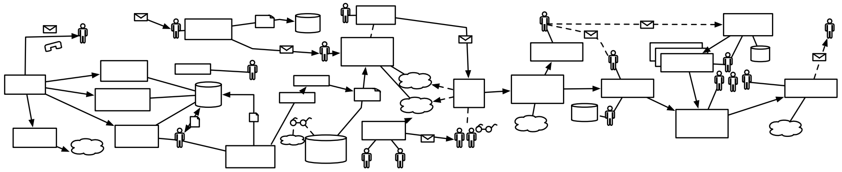


☑ <u>All</u> processes should be studied with an improvement disciple

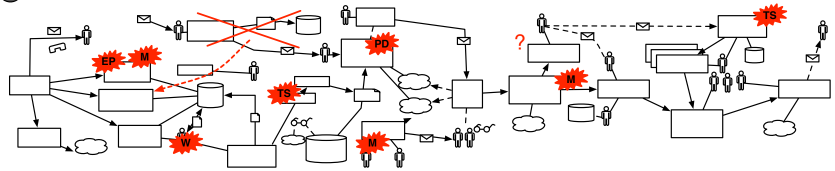
☑ Incidents are just as much a “process” as delivery

Lean on Lean to find what to fix

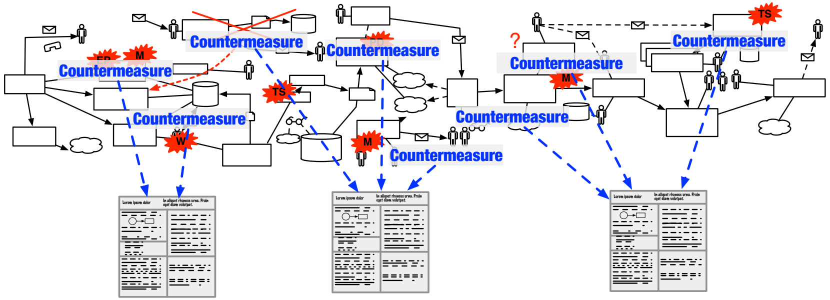
1 Map the end-to-end flow of information and artifacts (using a recent delivery or event)



2 Identify what slows lead times, undermines quality, and impacts flow



3 Identify countermeasures and create improvement storyboards (justification/plan)



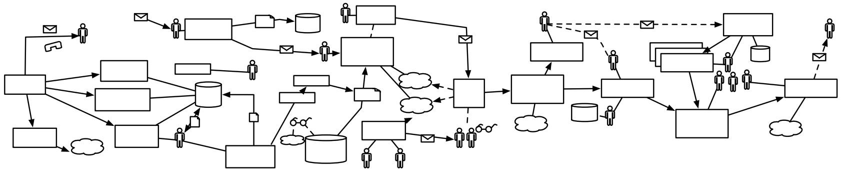
☑ All processes should be studied with an improvement disciple

☑ Incidents are just as much a “process” as delivery

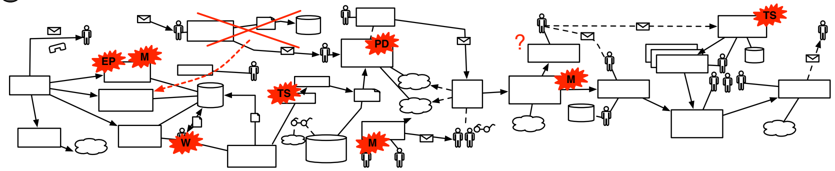
☑ Look to Lean for proven improvement techniques (value stream mapping, waste analysis, improvement kata)

Lean on Lean to find what to fix

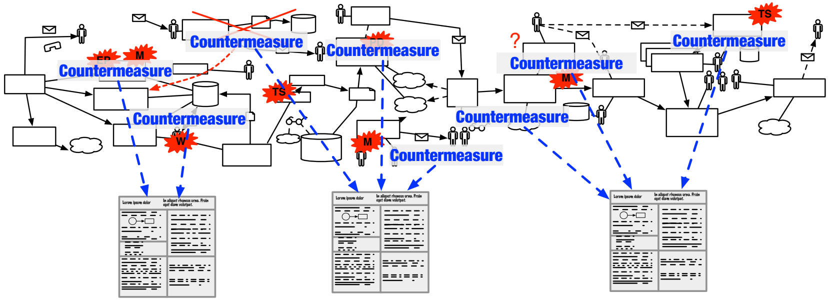
1 Map the end-to-end flow of information and artifacts (using a recent delivery or event)



2 Identify what slows lead times, undermines quality, and impacts flow



3 Identify countermeasures and create improvement storyboards (justification/plan)



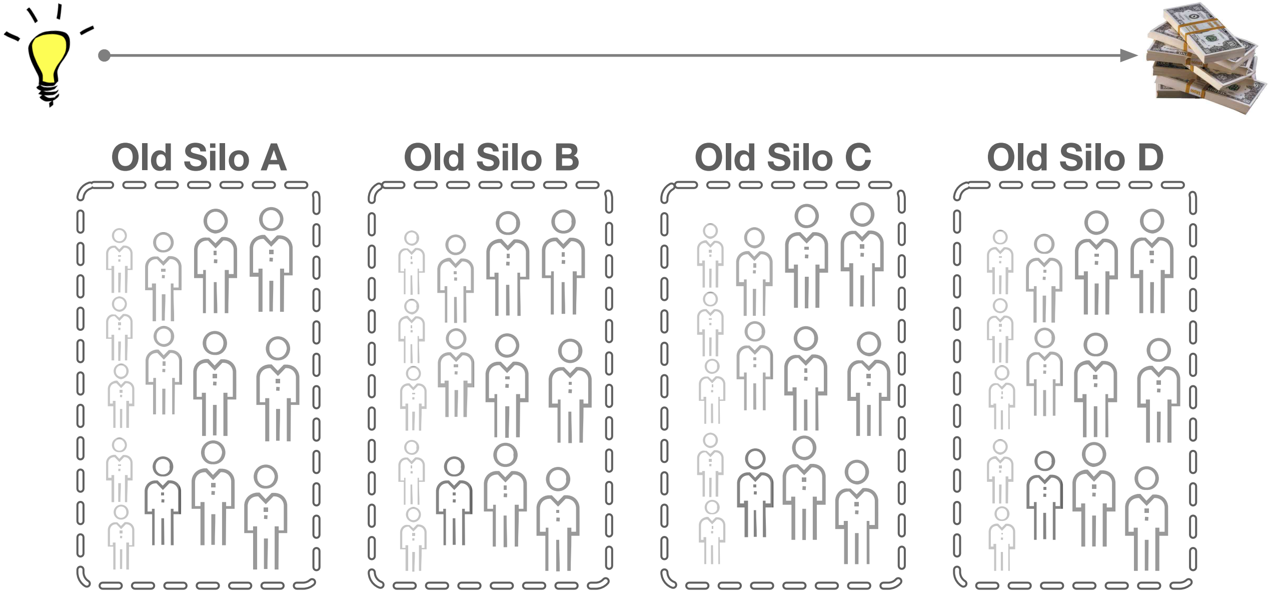
☑ All processes should be studied with an improvement disciple

☑ Incidents are just as much a "process" as delivery

☑ Look to Lean for proven improvement techniques (value stream mapping, waste analysis, improvement kata)

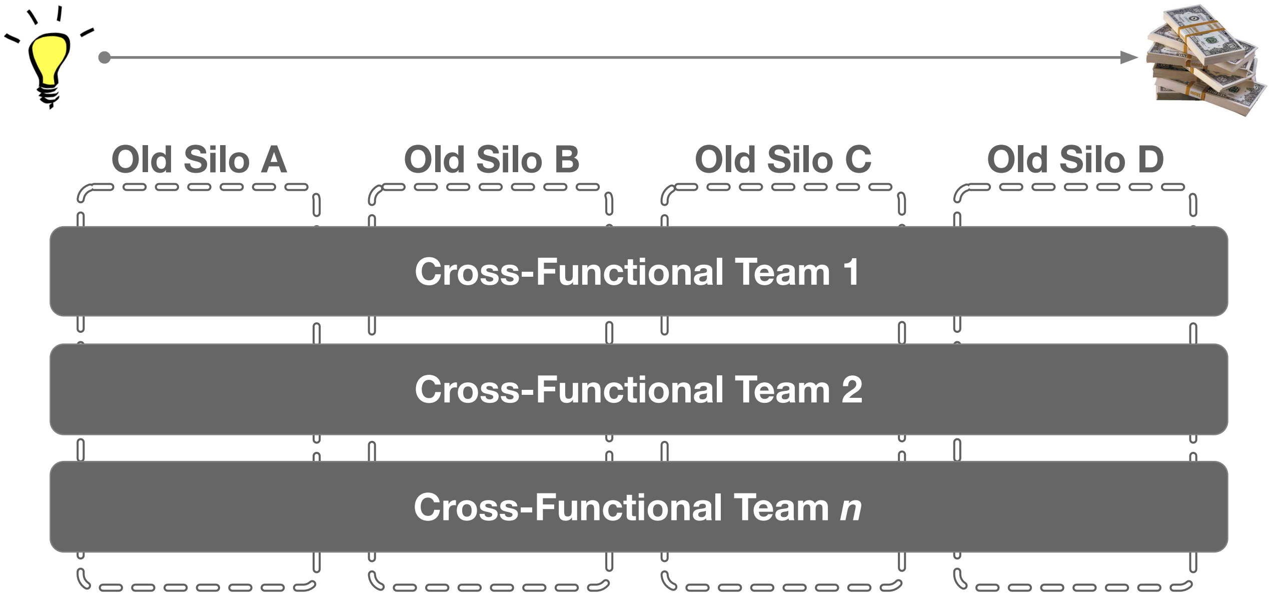
☑ Make it a part of your organization's discipline

Get rid of as many silos as possible



RUNDECK

Get rid of as many silos as possible



Old Silo A

Old Silo B

Old Silo C

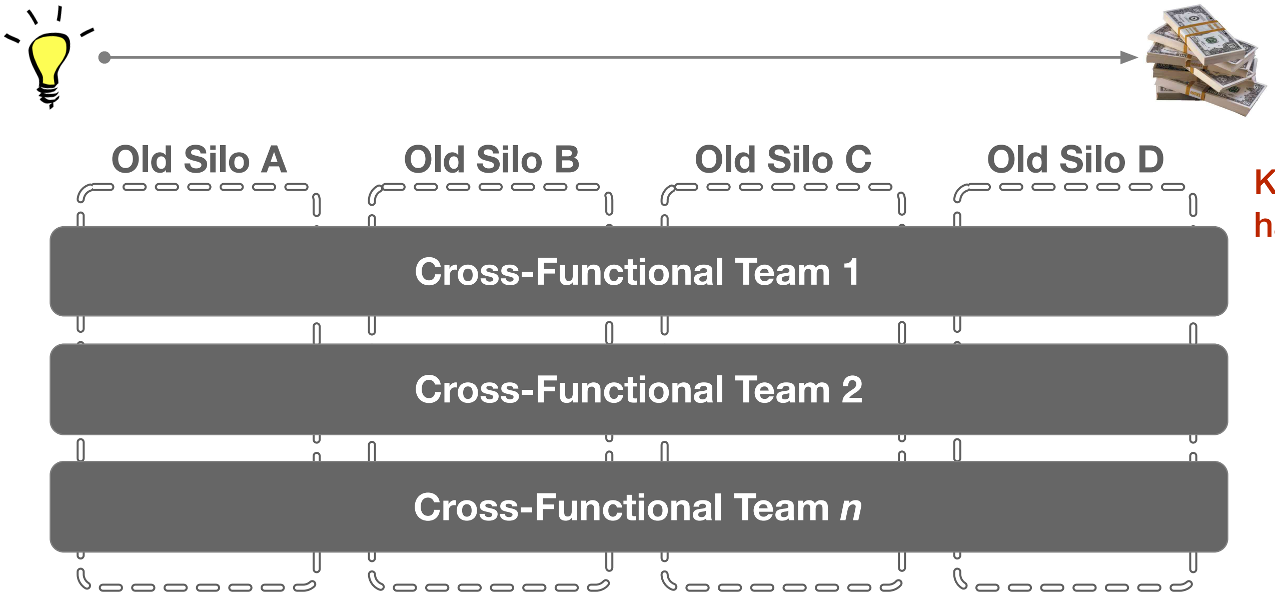
Old Silo D

Cross-Functional Team 1

Cross-Functional Team 2

Cross-Functional Team *n*

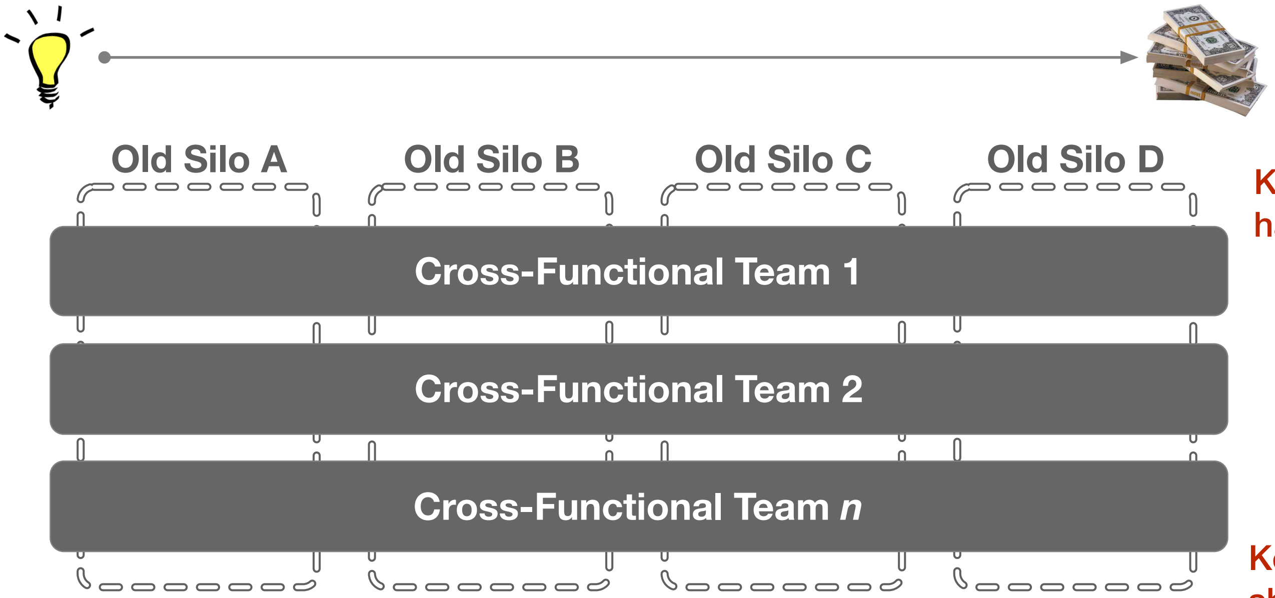
Get rid of as many silos as possible



Key 1: get rid of as many handoffs as possible

RUNDECK

Get rid of as many silos as possible



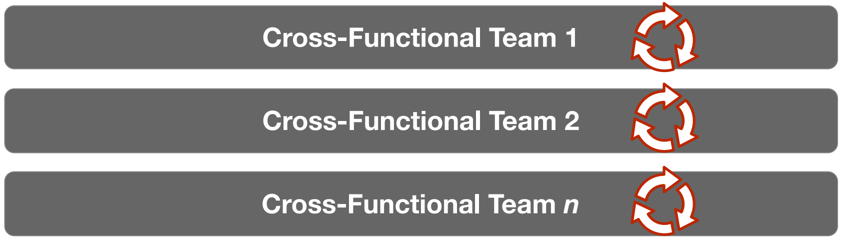
Key 1: get rid of as many handoffs as possible

Key 2: "Horizontal" shared responsibility, *not* everyone do everything!

RUNDECK

Shared responsibility matters more than org model

“Netflix" Model

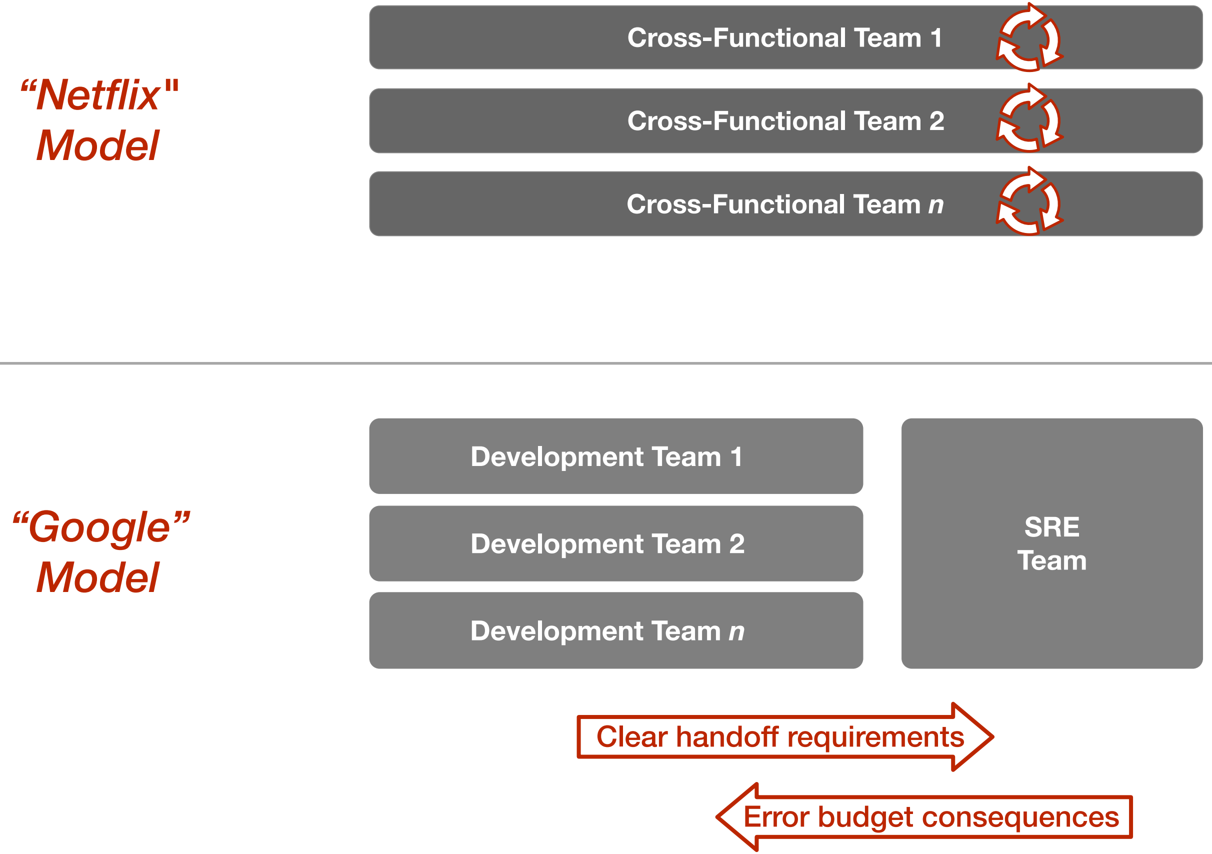


Cross-Functional Team 1

Cross-Functional Team 2

Cross-Functional Team *n*

Shared responsibility matters more than org model



“Netflix" Model

Cross-Functional Team 1

Cross-Functional Team 2

Cross-Functional Team *n*

“Google” Model

Development Team 1

Development Team 2

Development Team *n*

SRE Team

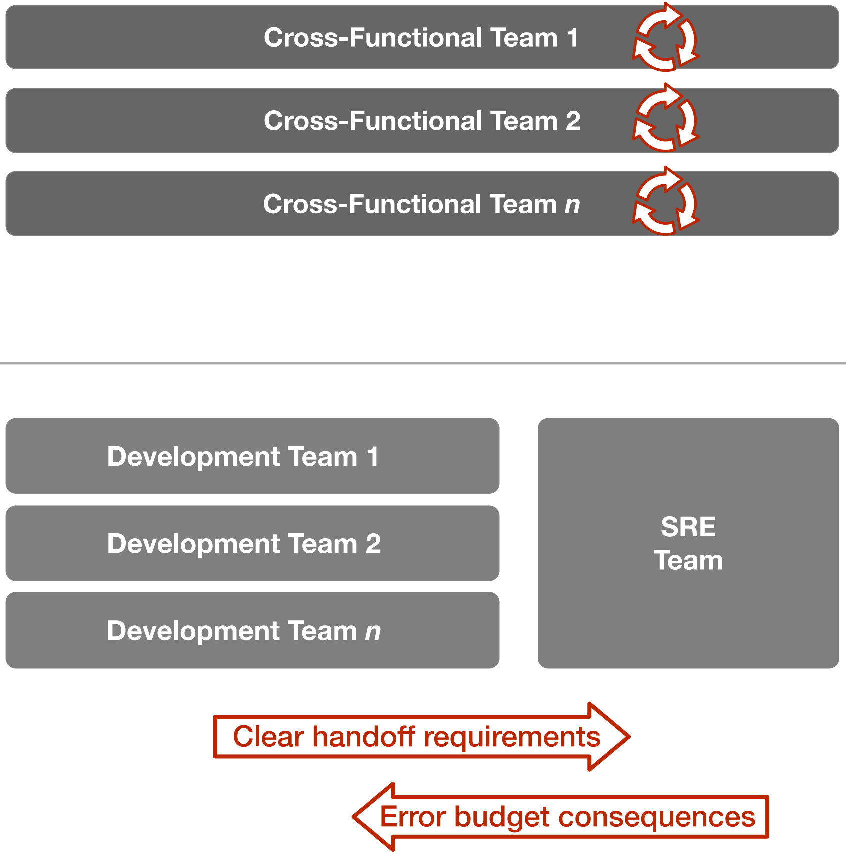
Clear handoff requirements

Error budget consequences

Shared responsibility matters more than org model

“Netflix" Model

“Google” Model



Cross-Functional Team 1

Cross-Functional Team 2

Cross-Functional Team *n*

Development Team 1

Development Team 2

Development Team *n*

SRE Team

Clear handoff requirements

Error budget consequences

Same high-quality, high-velocity results!

Why focus on getting rid of handoffs?

RUNDECK

Why focus on getting rid of handoffs?

1. Your people are your most valuable assets

Why focus on getting rid of handoffs?

1. Your people are your most valuable assets

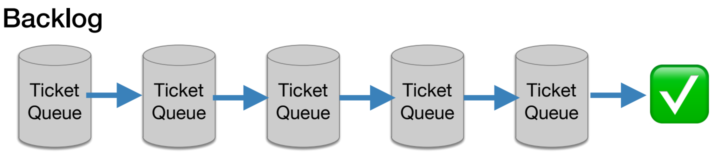
2. The SRE skillset is expensive

Why focus on getting rid of handoffs?

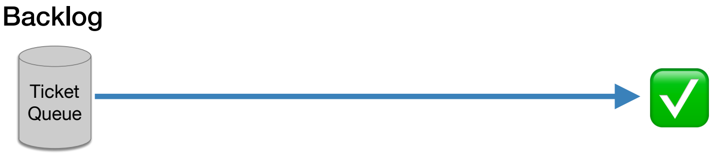
1. Your people are your most valuable assets
2. The SRE skillset is expensive
3. Stay out of their way!

SREs are expensive, stay out of their way!

Not this:

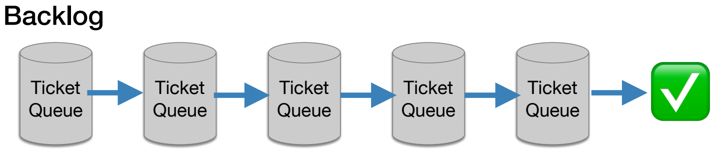


This:

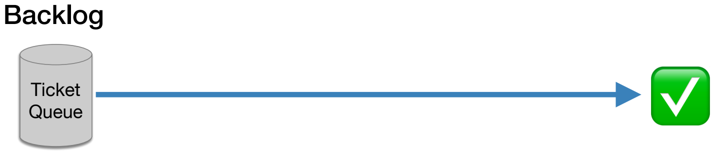


SREs are expensive, stay out of their way!

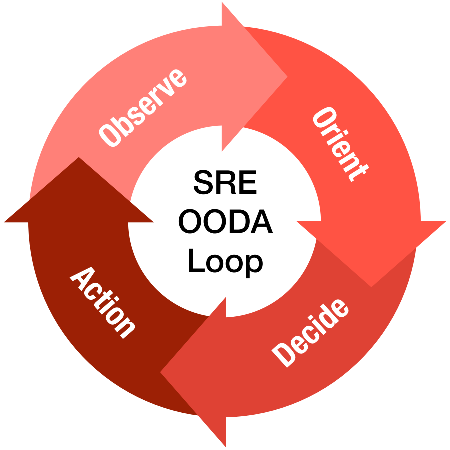
Not this:



This:



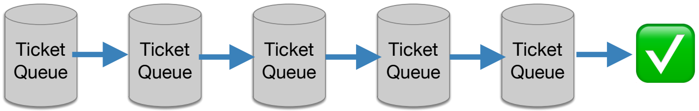
Reduce friction:



SREs are expensive, stay out of their way!

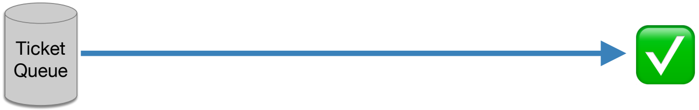
Not this:

Backlog



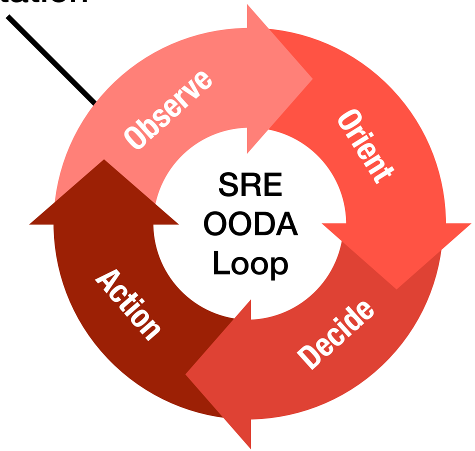
This:

Backlog



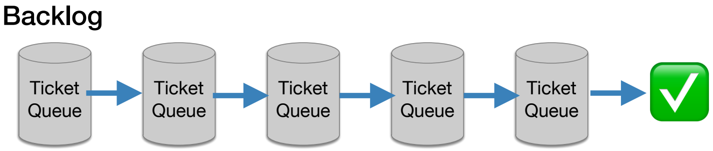
Reduce friction:

Invest in the right instrumentation

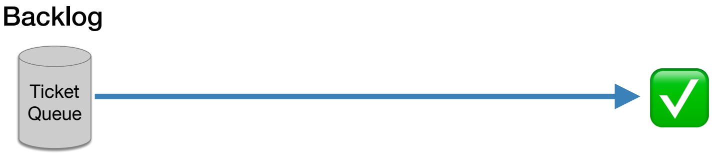


SREs are expensive, stay out of their way!

Not this:



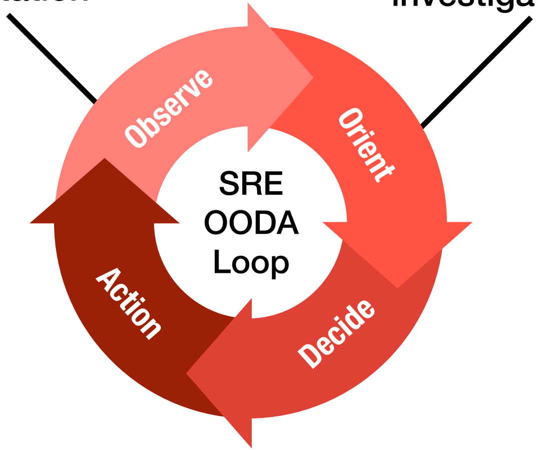
This:



Reduce friction:

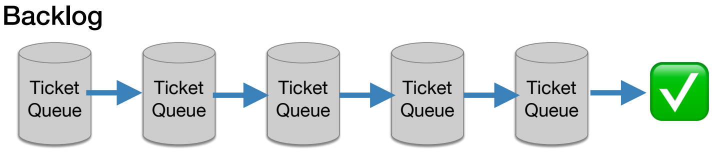
Invest in the right instrumentation

Invest in collaboration, checklists, investigatory tools

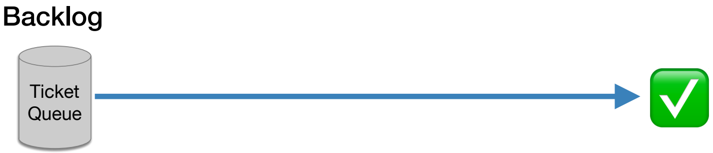


SREs are expensive, stay out of their way!

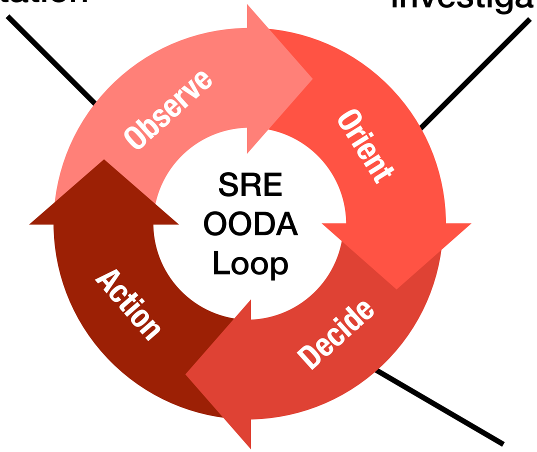
Not this:



This:



Reduce friction:



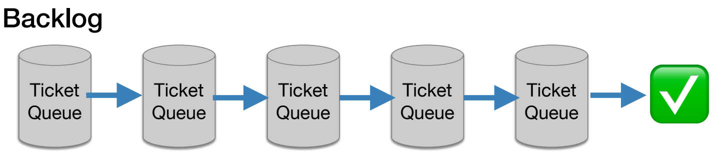
Invest in the right instrumentation

Invest in collaboration, checklists, investigatory tools

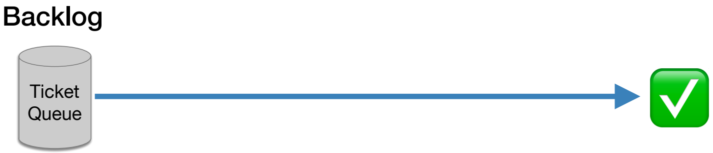
Empower them to make decisions!

SREs are expensive, stay out of their way!

Not this:



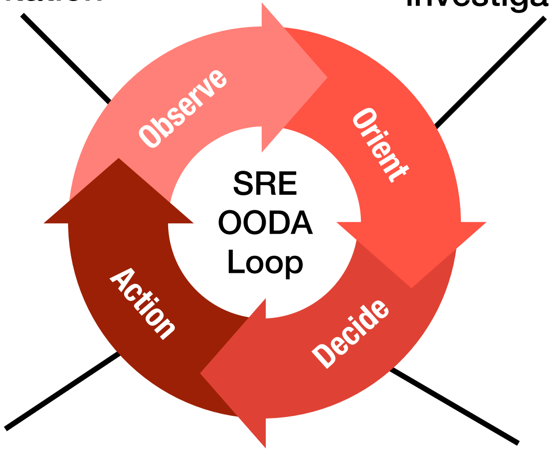
This:



Reduce friction:

Invest in the right instrumentation

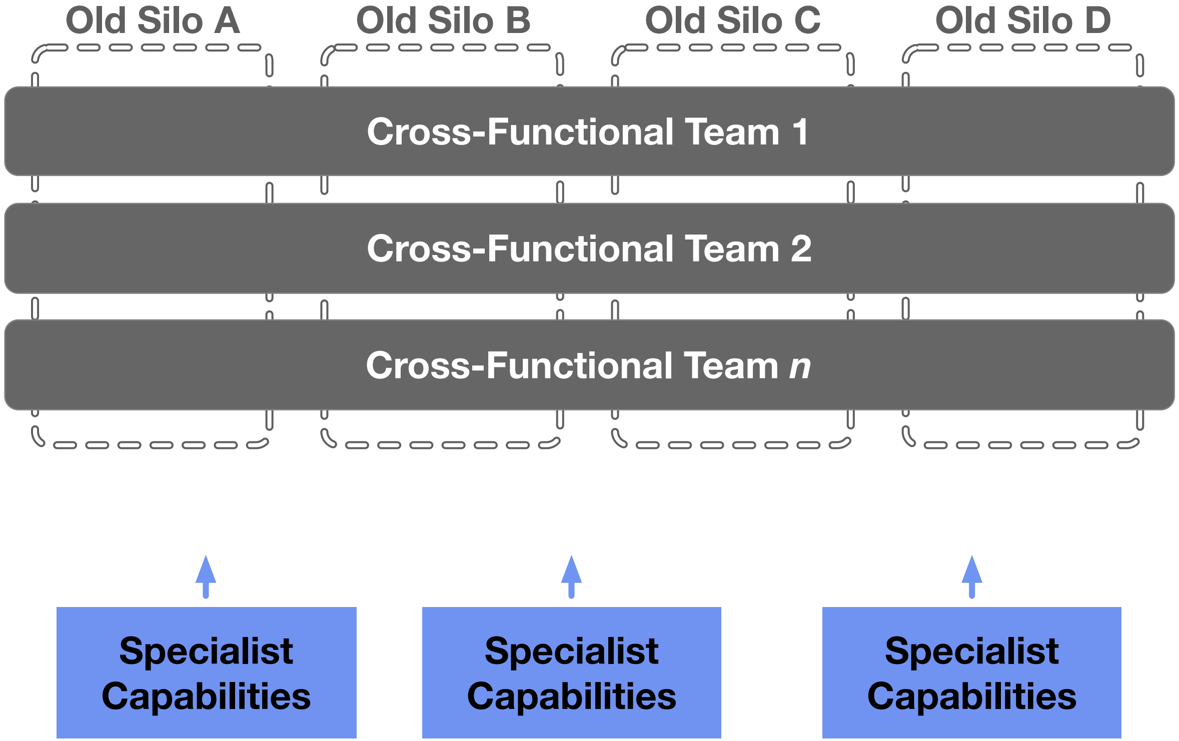
Invest in collaboration, checklists, investigatory tools



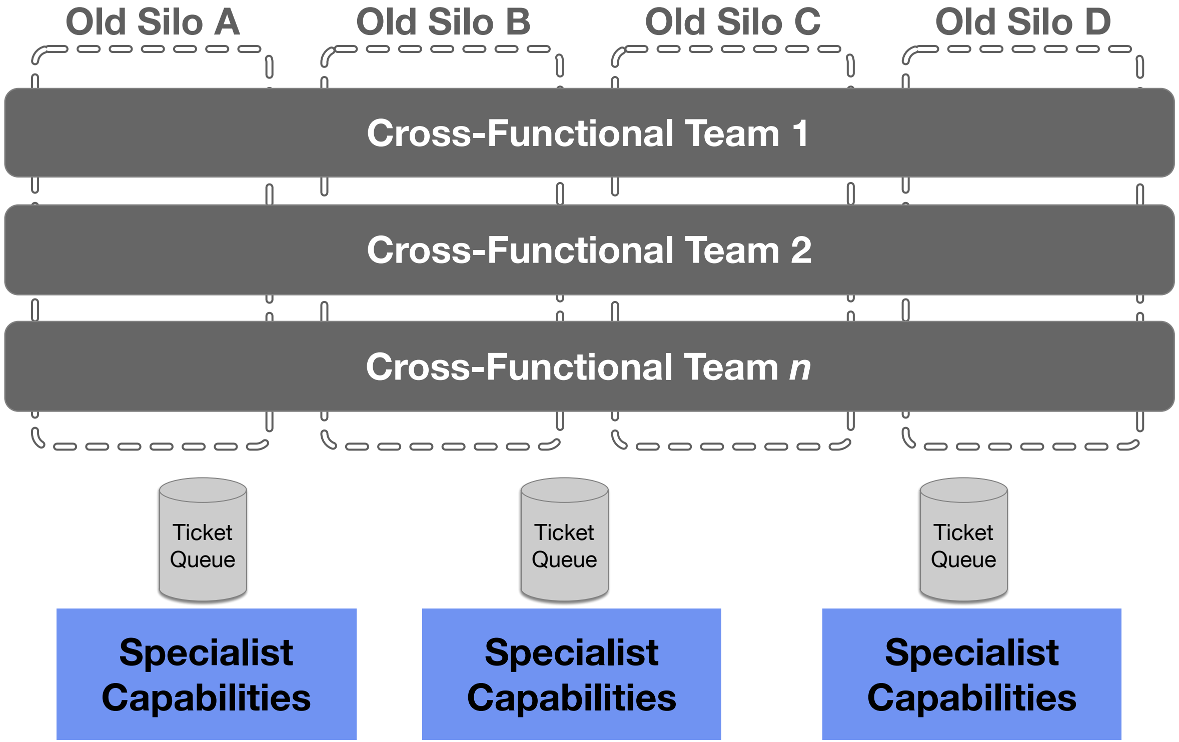
Empower them to take action!

Empower them to make decisions!

What about the handoffs you can’t get rid of?

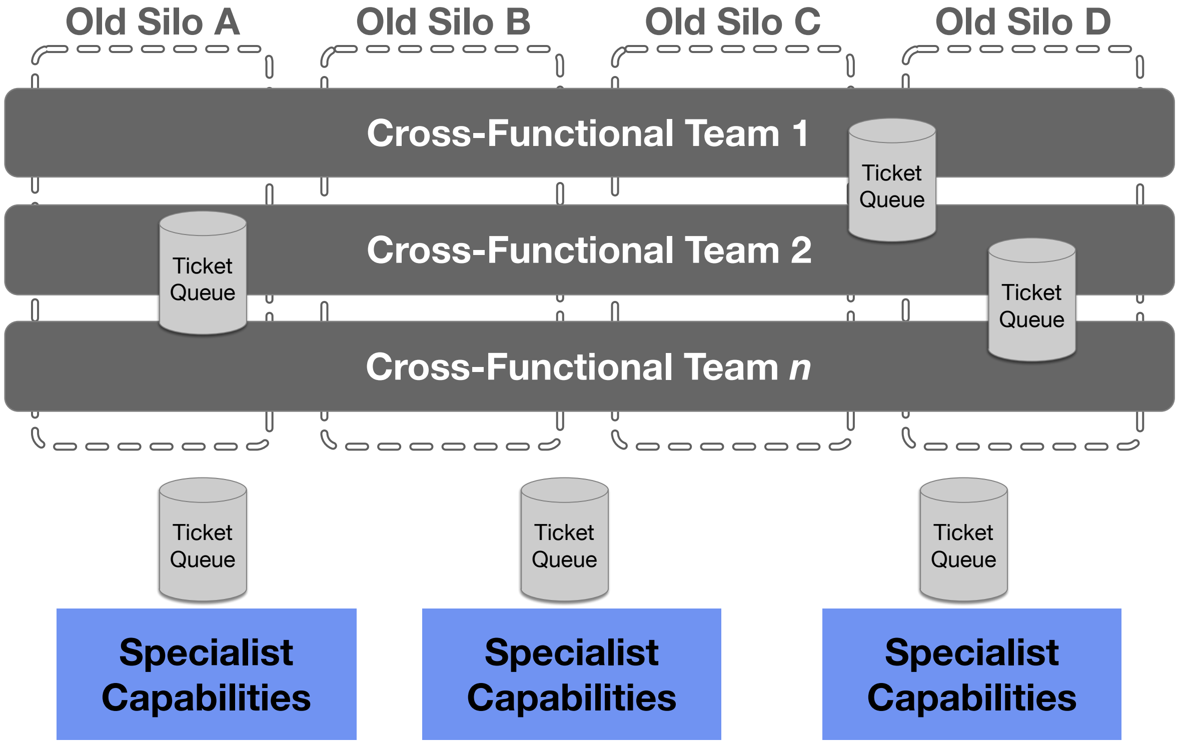


What about the handoffs you can't get rid of?

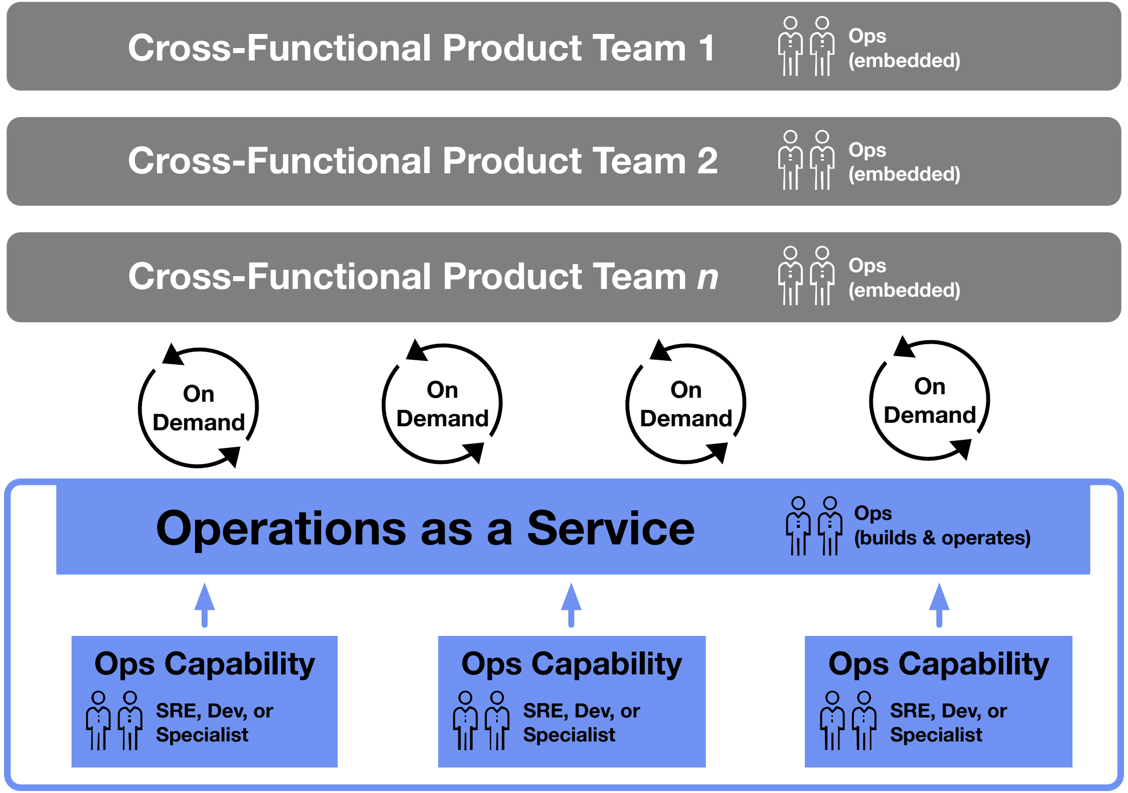


RUNDECK

What about the handoffs you can't get rid of?

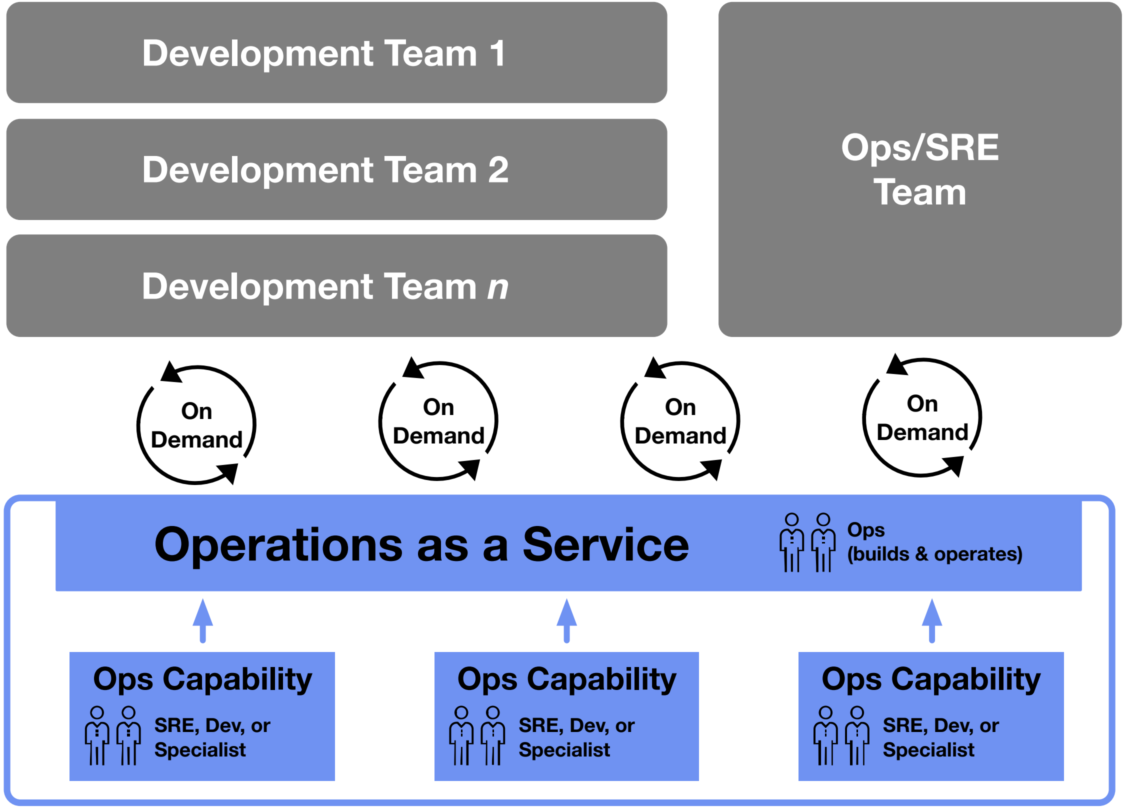


Operations as a Service: Turn handoffs into self-service



RUNDECK

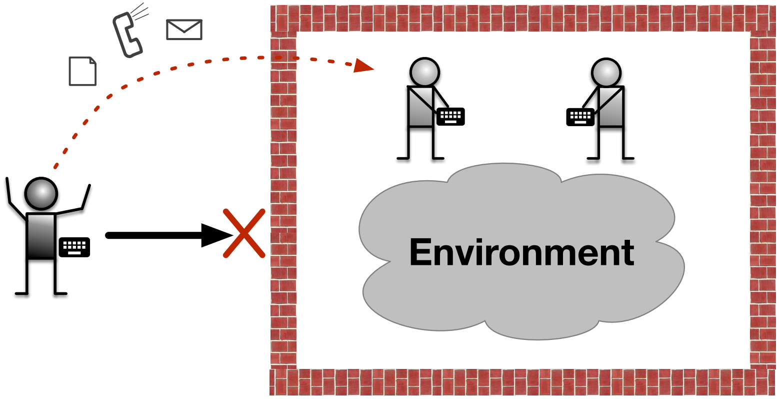
Operations as a Service: Works with any org model



RUNDECK

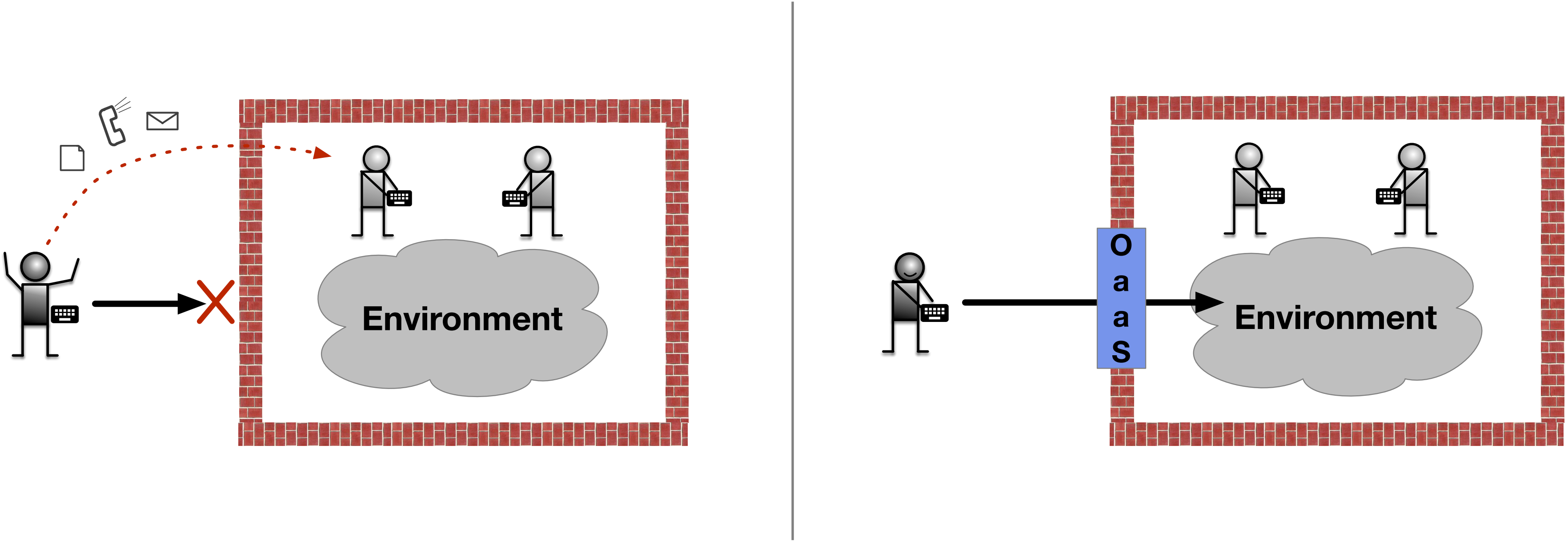
Operations as a Service: Popular Uses for SRE

"I could fix it, if I could get to it"



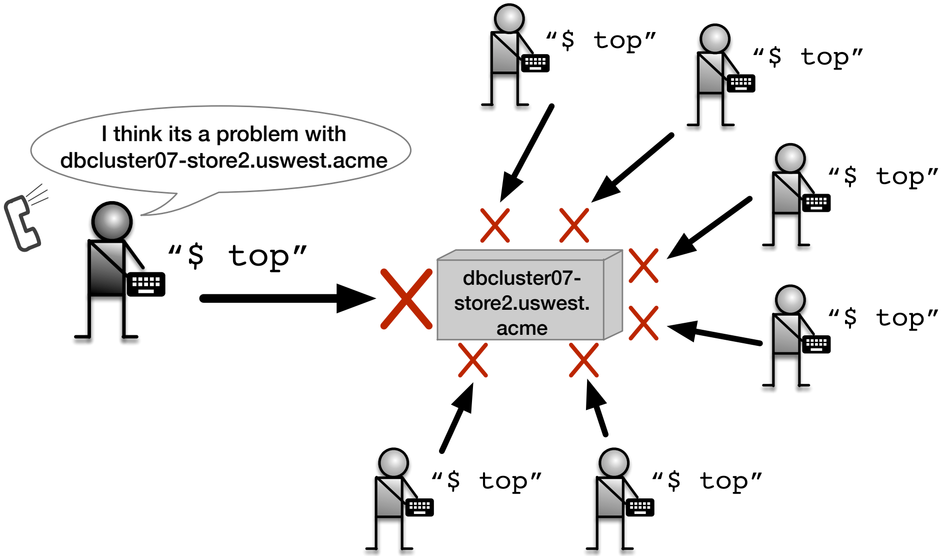
Operations as a Service: Popular Uses for SRE

"I could fix it, if I could get to it"



Operations as a Service: Popular Uses for SRE

“Avoiding the dogpile”

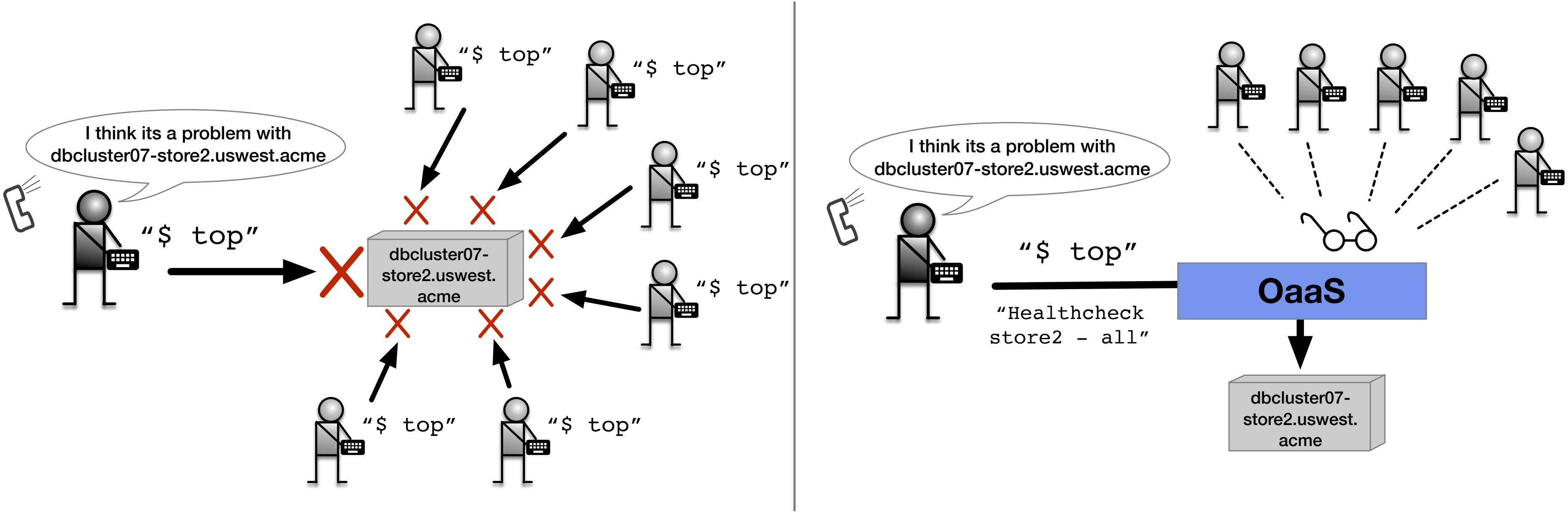


I think its a problem with dbcluster07-store2.uswest.acme

RUNDECK

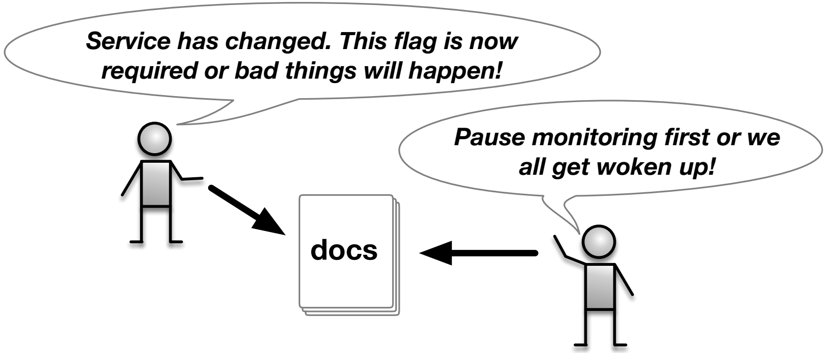
Operations as a Service: Popular Uses for SRE

“Avoiding the dogpile”


I think its a problem with dbcluster07-store2.uswest.acme
“$ top”
“$ top”
“$ top”
“$ top”
“$ top”
“$ top”
“$ top”
dbcluster07-store2.uswest.acme
I think its a problem with dbcluster07-store2.uswest.acme
“$ top”
“Healthcheck store2 - all”
OaaS
dbcluster07-store2.uswest.acme

Operations as a Service: Popular Uses for SRE

“I don’t read wikis. I’m an expert.”

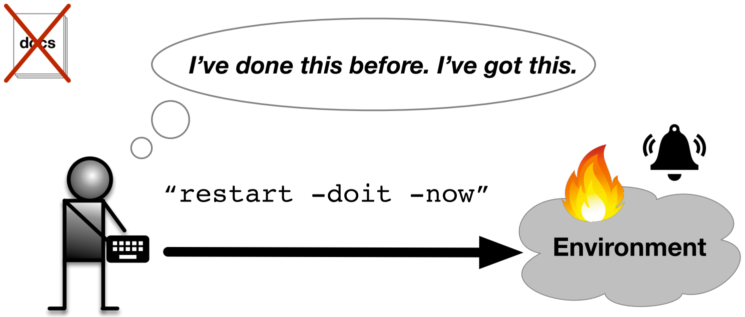


Service has changed. This flag is now required or bad things will happen!

Pause monitoring first or we all get woken up!

docs

Later…



docs

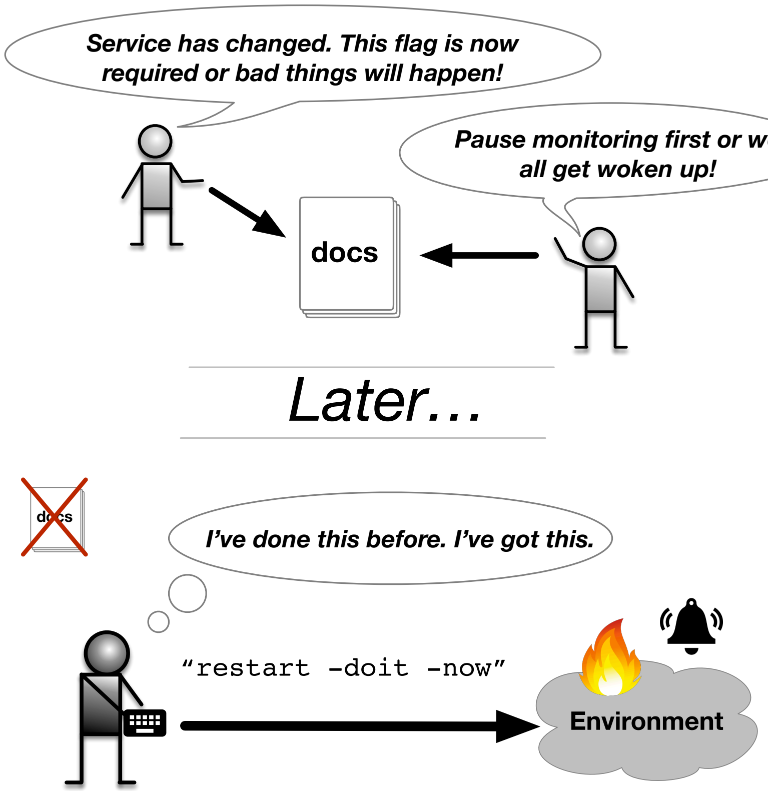
I’ve done this before. I’ve got this.

```
“restart -doit -now”
```

Environment

Operations as a Service: Popular Uses for SRE

"I don't read wikis. I'm an expert."



Service has changed. This flag is now required or bad things will happen!

Pause monitoring first or we all get woken up!

docs

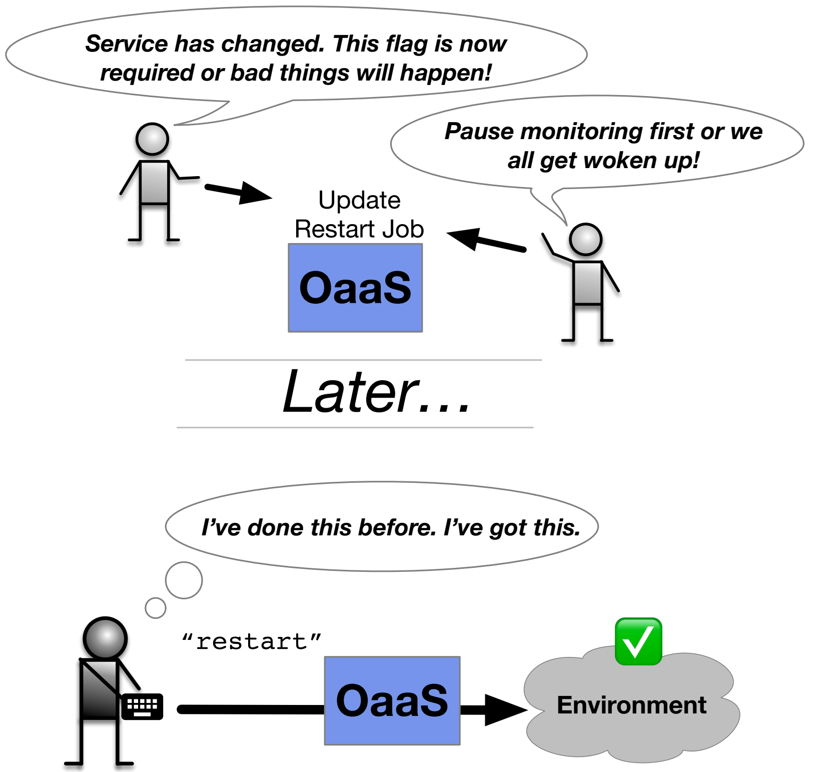
Later…

docs

I've done this before. I've got this.

`"restart -doit -now"`

Environment



Service has changed. This flag is now required or bad things will happen!

Pause monitoring first or we all get woken up!

Update Restart Job

OaaS

Later…

I've done this before. I've got this.

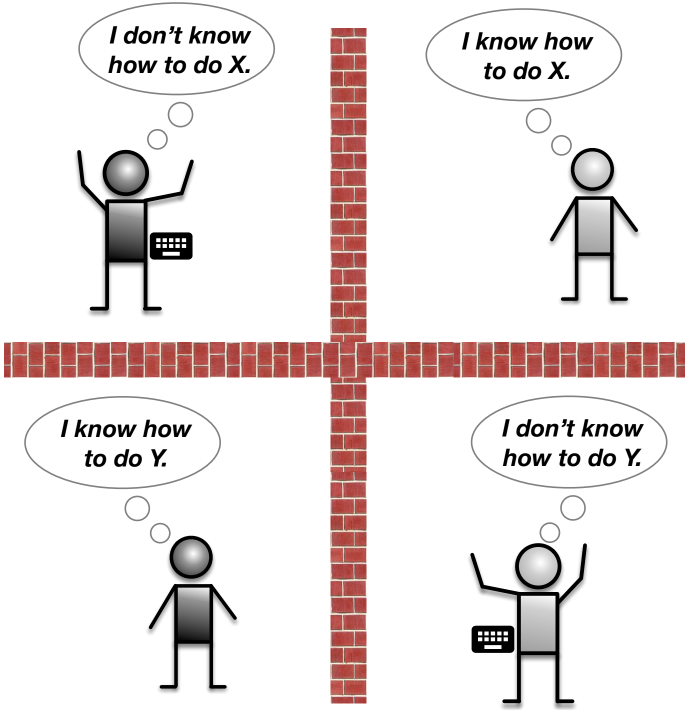
`"restart"`

OaaS

Environment

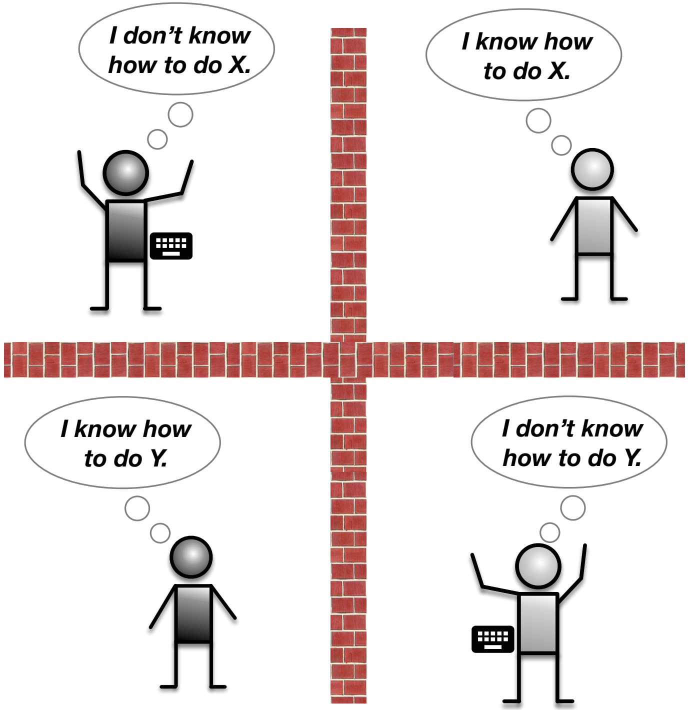
Operations as a Service: Popular Uses for SRE

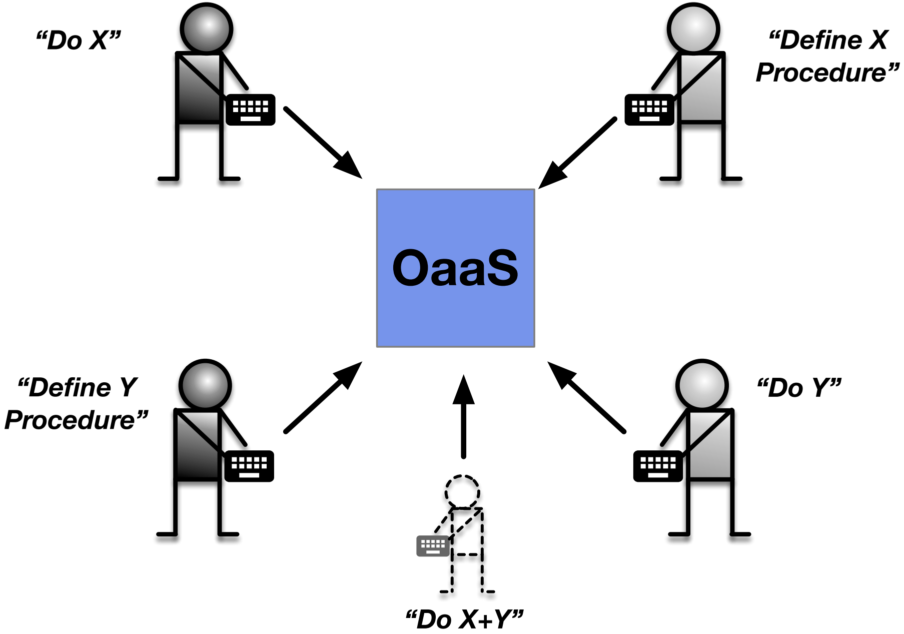
“Uneven and hidden skills”



Operations as a Service: Popular Uses for SRE

“Uneven and hidden skills”

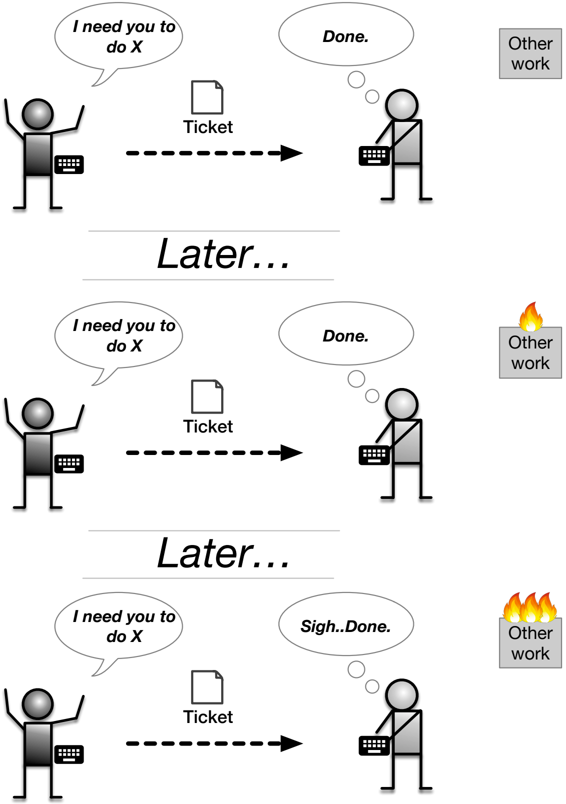




RUNDECK

Operations as a Service: Popular Uses for SRE

“Let me do that for you again… and again”



I need you to do X

Ticket

Done.

Other work

Later…

I need you to do X

Ticket

Done.

Other work

Later…

I need you to do X

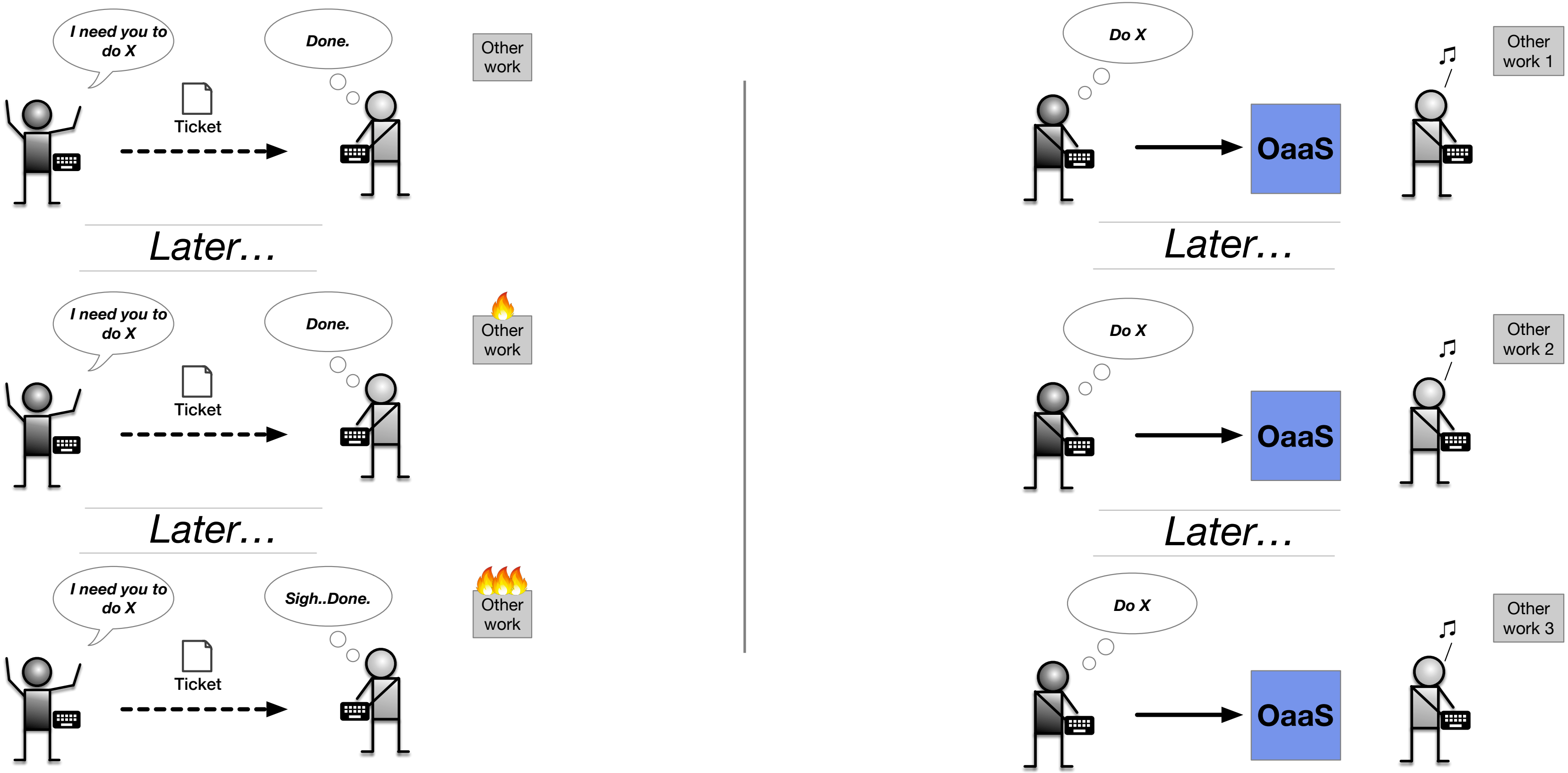
Ticket

Sigh..Done.

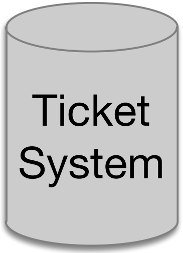
Other work

Operations as a Service: Popular Uses for SRE

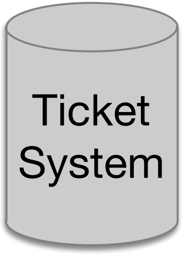
“Let me do that for you again… and again”



Use tickets only for what they are good for



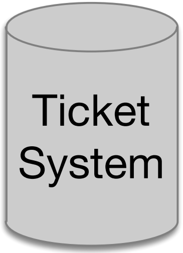
Use tickets only for what they are good for



Ticket System

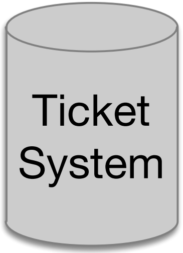
1. Documenting true problems/issues/exceptions

Use tickets only for what they are good for



1. Documenting true problems/issues/exceptions
2. Routing for necessary approvals

Use tickets only for what they are good for

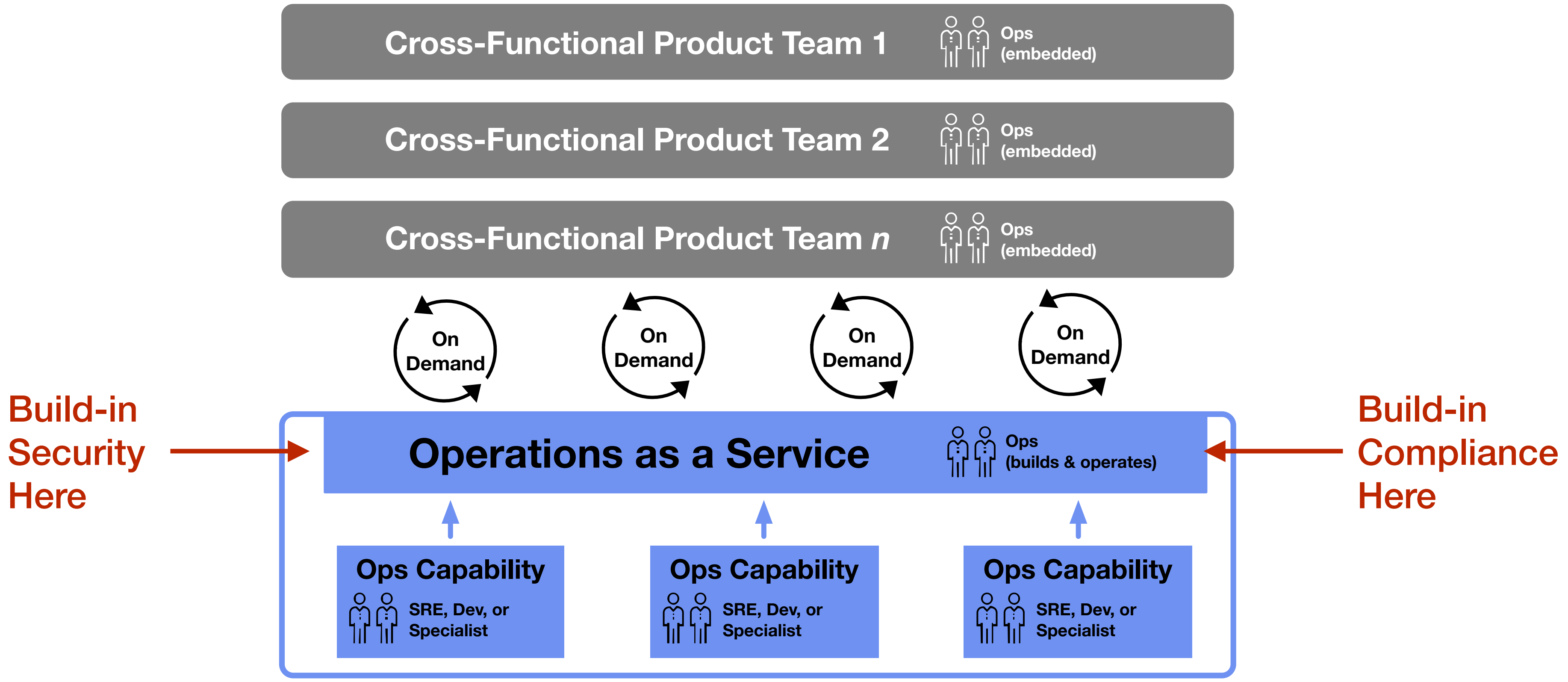


Ticket System

1. Documenting true problems/issues/exceptions
2. Routing for necessary approvals

Not **as a general purpose work management system!**

But won't Security or Compliance stop you?



RUNDECK

But what about ITIL® ?

But what about ITIL®?

- Ask ITIL people and they say SRE is ITIL compatible

But what about ITIL®?

- Ask ITIL people and they say SRE is ITIL compatible
- Ask people who have seen ITIL implemented and they say “how?”

But what about ITIL®?

- Ask ITIL people and they say SRE is ITIL compatible
- Ask people who have seen ITIL implemented and they say “how?”
- Agile+DevOps+SRE have self-regulation and shared responsibility features that seem to undermine ITIL command and control nature

But what about ITIL®?

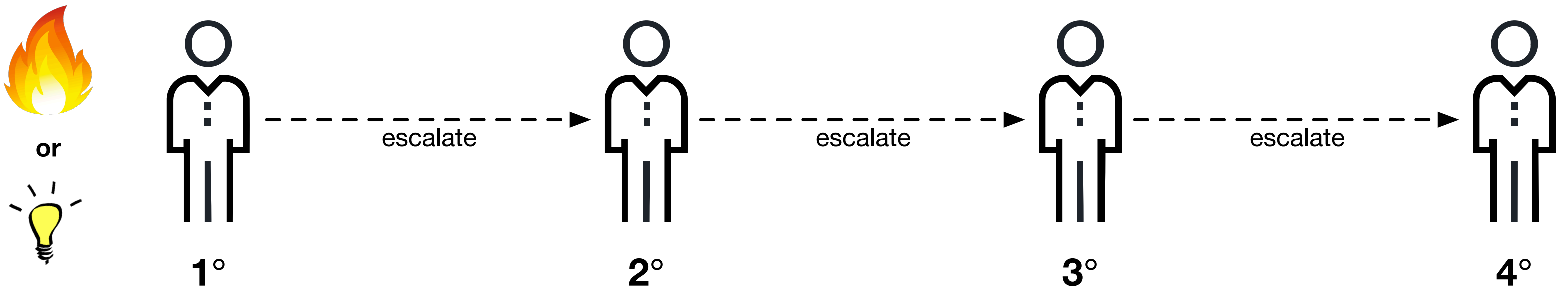
- Ask ITIL people and they say SRE is ITIL compatible
- Ask people who have seen ITIL implemented and they say “how?”
- Agile+DevOps+SRE have self-regulation and shared responsibility features that seem to undermine ITIL command and control nature
- ITIL “Standard Change” is often focus of discussion, but it still implies approval model

RUNDECK

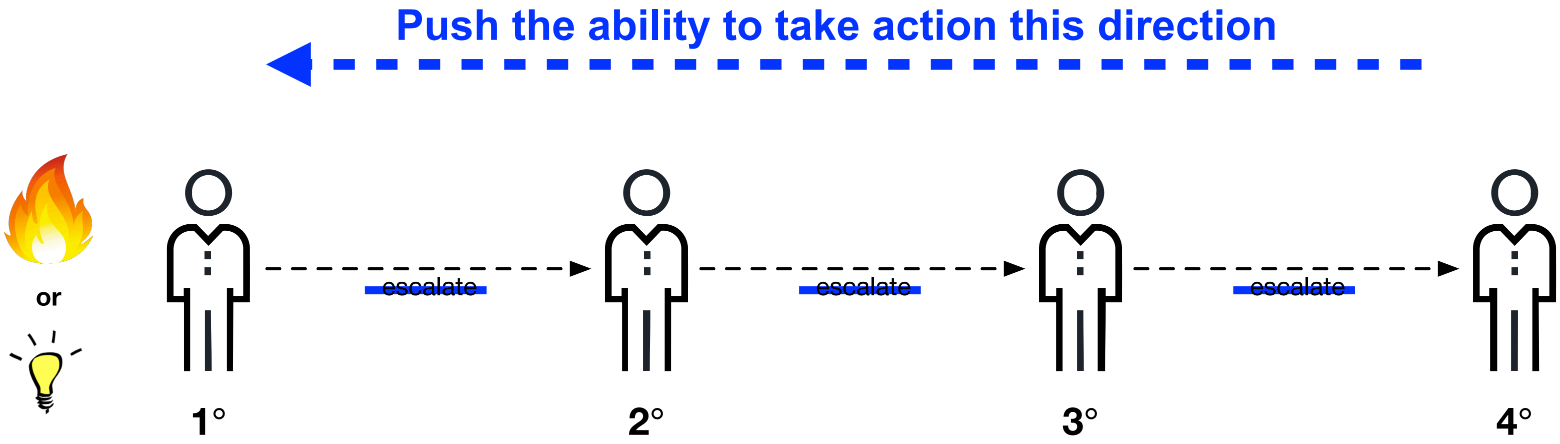
But what about ITIL®?

- Ask ITIL people and they say SRE is ITIL compatible
- Ask people who have seen ITIL implemented and they say “how?”
- Agile+DevOps+SRE have self-regulation and shared responsibility features that seem to undermine ITIL command and control nature
- ITIL “Standard Change” is often focus of discussion, but it still implies approval model
- Straight talk: are we doing contortions to defend a sunk cost?

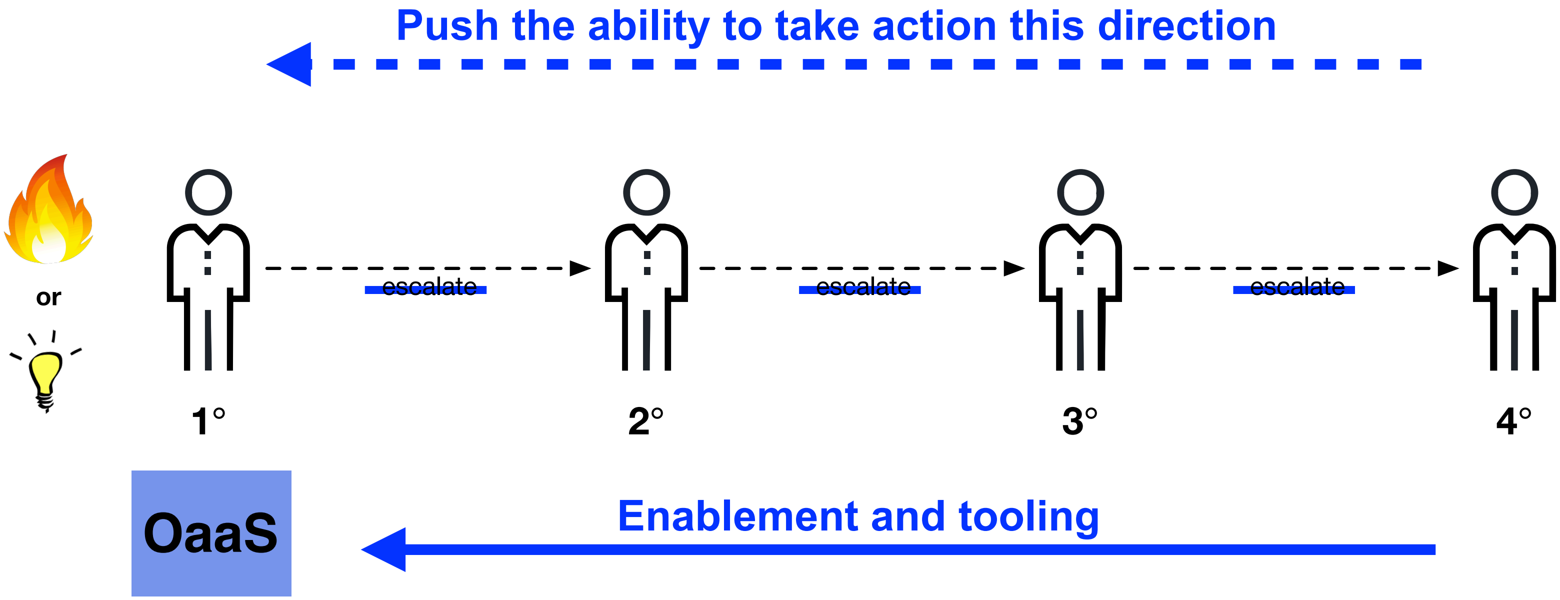
“Shift Left” the ability to take action



“Shift Left” the ability to take action



“Shift Left” the ability to take action



Reduce Toil

Reduce Toil

1. Track toil levels for each team

Reduce Toil

1. Track toil levels for each team

2. Set toil limits for each team

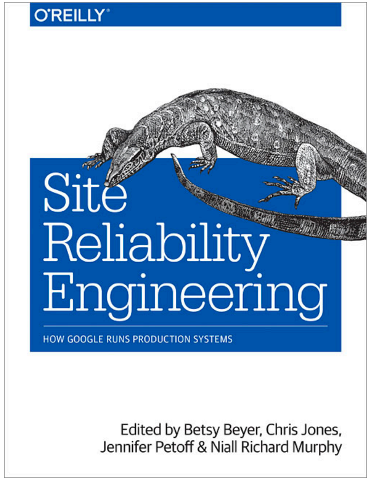
Reduce Toil

1. Track toil levels for each team

2. Set toil limits for each team

3. Fund efforts to reduce toil (with emphasis on teams over toil limits)

Start a book club

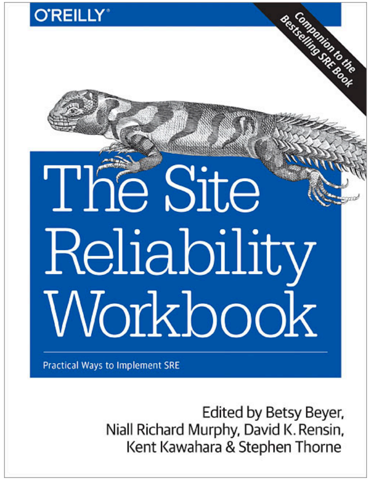


O'REILLY

Site Reliability Engineering

HOW GOOGLE RUNS PRODUCTION SYSTEMS

Edited by Betsy Beyer, Chris Jones,
Jennifer Petoff & Niall Richard Murphy



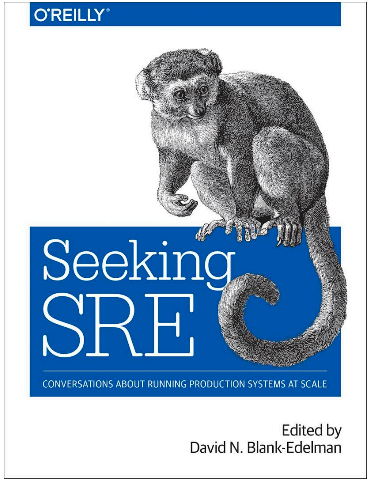
O'REILLY

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



O'REILLY

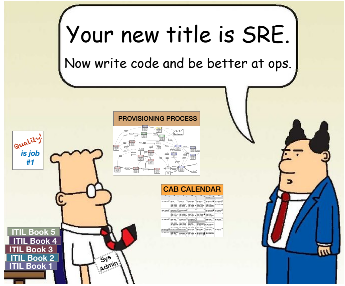
Seeking SRE

CONVERSATIONS ABOUT RUNNING PRODUCTION SYSTEMS AT SCALE

Edited by
David N. Blank-Edelman

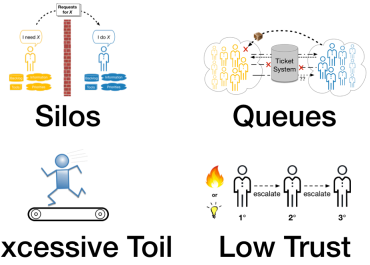
Recap

SRE is more than a title

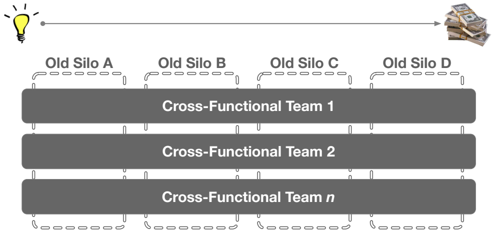


Dilbert characters © Scott Adams www.dilbert.com

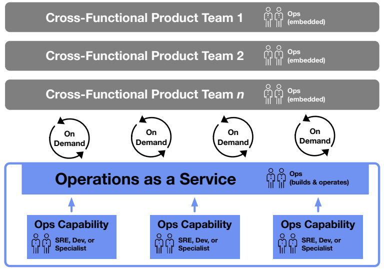
Understand the forces undermining SRE



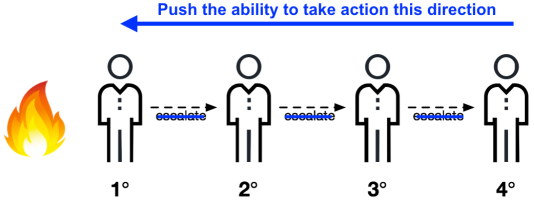
Focus on removing silos and queues



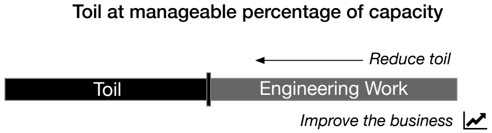
Leverage the Operations as a Service design pattern



"Shift-Left" control and decision making.



Reduce toil to create capacity to change



Let’s talk…

✉ damon@rundeck.com

@damonedwards

Dive Deeper Into Operations as a Service:

https://www.rundeck.com/oaas

RUNDECK