Scaling SRE Organizations
The journey from 1 to many teams
Wait! I don’t have an SRE team yet. Zero. Nada.

Please read *Do you have an SRE team yet?*

Let’s start a conversation via #sre_across_companies on the SRECon slack.
$ whoami

Customer Reliability Engineer (CRE) at Google

Gustavo Franco, aka stratus

11 years at Google
6 years on this subject
Scaling SRE Organizations

All things SRE but also changing how you do work, splitting, creating new teams, merging, ...

A set of two or more SRE teams.
Questions
(Don’t be shy! I won’t call you out)

Who’s been a member of a brand new SRE team?

Who’s participating in a new team, split, move or merge at the moment?
Why are we here?

Before an answer, think of a system

- Boundaries are unknown
- Workload is unbounded
- It’s a monolith
Why are we here?

Before an answer, think of a system

- Boundaries are unknown
- Workload is unbounded
- It’s a monolith

SLOs

o11y

Systems Design

Automation

Incident Response

....
Why are we here?

Think of an SRE team where

- Boundaries are unknown
- Workload is unbounded
- It’s a monolith

SLOs

e11y

Systems Design

Automation

Incident Response

....
Your SRE team is likely to fail if your implementation and scope aren’t written down and agreed upon.
SLO with consequences

Team charter with goalposts
Agenda

Assumptions
Do you need a new team?
  SRE Implementations
  Scope changes
  Load categories
Example
Talk with your leaders
Lessons I’ve learned
Assumptions

CxO/VP buy in

Funding

Healthy relationship with developers

Your organization is growing
Do you need a new team?

while True:
  ● Evaluate current team(s) implementation
  ● Look for recent or upcoming scope changes
  ● Assess current load
SRE Implementations

How teams are organized and approach their work
SRE Implementations

01 Consulting
02 Embedded
03 Tools or Infra
04 Slice
05 Kitchen Sink
SRE Implementations

01 Consulting
Self explanatory.
Usually not many code or config changes, no oncall.

02 Embedded

03 Tools or Infra

04 Slice

05 Kitchen Sink
SRE Implementations

01 Consulting

02 Embedded
Similar to consulting but with oncall, code and config changes

03 Tools or Infra

04 Slice

05 Kitchen Sink
SRE Implementations

01 Consulting
02 Embedded
03 Tools or Infra
  Service owners for tools or infrastructure
04 Slice
05 Kitchen Sink
SRE Implementations

01 Consulting
02 Embedded
03 Tools or Infra
04 Slice
05 Kitchen Sink

Slice
Service owners for a slice of the product(s) or workflows
SRE Implementations

01 Consulting
02 Embedded
03 Tools or Infra
04 Slice
05 Kitchen Sink

Scope is everything. First SRE team for many companies
# SRE Implementations

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>01</strong></td>
<td><strong>02</strong></td>
<td><strong>03</strong></td>
<td><strong>04</strong></td>
<td><strong>05</strong></td>
</tr>
<tr>
<td><strong>Consulting</strong></td>
<td><strong>Embedded</strong></td>
<td><strong>Tools or Infra</strong></td>
<td><strong>Slice</strong></td>
<td><strong>Kitchen Sink</strong></td>
</tr>
<tr>
<td>Self explanatory. Usually not many code or config changes, no oncall.</td>
<td>Similar to consulting but with oncall, code and config changes</td>
<td>Service owners for tools or infrastructure</td>
<td>Service owners for a slice of the product(s) or workflows</td>
<td>Scope is everything. First SRE team for many companies</td>
</tr>
<tr>
<td>Tier</td>
<td>Description</td>
<td>Details</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>------------------------------</td>
<td>----------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>01</td>
<td>Consulting</td>
<td>Usually not applicable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>Embedded</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>03</td>
<td>Tools or Infra</td>
<td>T1: Consulting</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>T2: Projects</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>T3: Oncall</td>
<td></td>
<td></td>
</tr>
<tr>
<td>04</td>
<td>Slice</td>
<td>Same as Tools or Infra</td>
<td></td>
<td></td>
</tr>
<tr>
<td>05</td>
<td>Kitchen Sink</td>
<td>Same as Tools or Infra</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SRE Implementations - Size and Oncall

01 Consulting
Generally not a concern for both

02 Embedded
Generally not a concern in terms of size
May participate in the dev rotation

03 Tools or Infra
Very company specific

04 Slice
-

05 Kitchen Sink
-
SRE Implementations - Main Risks

01 Consulting
May be perceived as hands-off

02 Embedded
May suffer from lack of knowledge sharing

03 Tools or Infra
Scope may be too narrow

04 Slice
Same as tools, also it’s costly to run many slices

05 Kitchen Sink
Tends to move from deep engagement to shallow as scope and load grows
SRE Implementations

We are Company SRE (Kitchen Sink)
We have a team charter [and SLOs]
We are happy with our implementation (no tiers)
Do you need a new team?

while True:

- Evaluate current team(s) implementation
- Look for recent or upcoming scope changes
- Assess current load
Scope Changes

How the subject matter may change for any given team
<table>
<thead>
<tr>
<th>Infrastructure</th>
<th>Services</th>
<th>Products</th>
</tr>
</thead>
</table>

Scope Changes (Examples)
Scope Changes (Examples)

**Infrastructure**
- Migrations
- Running legacy and new infra post migration

**Services**

**Products**
Scope Changes (Examples)

Infrastructure
- Migrations
- Running legacy and new infra post migration

Services
- Brand new services or simply new to SRE

Products
# Scope Changes (Examples)

<table>
<thead>
<tr>
<th>Infrastructure</th>
<th>Services</th>
<th>Products</th>
</tr>
</thead>
<tbody>
<tr>
<td>Migrations</td>
<td>Brand new services or simply new to SRE</td>
<td>Brand new products or new to SRE</td>
</tr>
<tr>
<td>Running legacy and new infra post migration</td>
<td>Tends to require establishing new inter team relationships</td>
<td></td>
</tr>
</tbody>
</table>

@stratus
## Scope Changes (Examples)

<table>
<thead>
<tr>
<th>Infrastructure</th>
<th>Services</th>
<th>Products</th>
</tr>
</thead>
<tbody>
<tr>
<td>Migrations</td>
<td>Brand new services or simply new to SRE</td>
<td>Brand new products or new to SRE</td>
</tr>
<tr>
<td>Running legacy and new infra post migration</td>
<td></td>
<td>Tends to require establishing new inter team relationships</td>
</tr>
</tbody>
</table>
Scope Changes

We are Company SRE
We have a team charter [and SLOs]
We are happy with our implementation (no tiers)
Yeah, we’ve had some major changes in scope recently (partial cloud migration and a new product)
Do you need a new team?

while True:

- Evaluate current team(s) implementation
- Look for recent or upcoming scope changes
- Assess current load
Starting a new SRE team due to growth of tickets or pages may be counter-productive
Load Categories

Implementation and scope enable work to get to your team.

Load is what you are carrying at any given moment.
Load Categories

Project  Toil  Cognitive
Load Categories

**Project**

Engineering reliability into services

**Toil**

It can be indirectly (consulting)

**Cognitive**
## Load Categories

<table>
<thead>
<tr>
<th>Project</th>
<th>Toil</th>
<th>Cognitive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engineering reliability</td>
<td>Some kinds of operational work</td>
<td></td>
</tr>
<tr>
<td>into services</td>
<td></td>
<td></td>
</tr>
<tr>
<td>It can be indirectly</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(consulting)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Image: Google Cloud*
Load Categories

- **Project**: Engineering reliability into services
- **Toil**: Some kinds of operational work
- **Cognitive**: The cost to SRE grows faster as systems and problem spaces diverge

Google Cloud

@stratus
Load Categories

**Project**
Engineering reliability into services

**Toil**
Some kinds of operational work

**Cognitive**
The cost to SRE grows faster as systems and problem spaces diverge
Cognitive Overload Outcomes

- Informal oncall split
- Always one SRE per project*
- High attrition rate
- Insecurities about the unknown
- Reactive projects
- Long recruiting pitch
Cognitive overload is usually a positive indicator for a team split
Do you need a new team?

while True:

- Evaluate current team(s) implementation
- Look for recent or upcoming **scope** changes
- Assess current **load**
Example

We are Company SRE
We have a team charter [and SLOs]
We are happy with our implementation
Yeah, we’ve had some major changes in scope recently

We see all signs of cognitive overload
Example

We are Company SRE
We have a team charter [and SLOs]
We are happy with our implementation
Yeah, we’ve had some major changes in scope recently
We see all signs of cognitive overload

**Company SRE will split in Slice SRE and Infra SRE...**
Example: Company SRE Split

Comms  Staffing  Team
Example: Company SRE Split

**Comms**
Inform team and key partners ASAP

**Staffing**
Recruit as needed

**Team**
Draft team charters
Compile a longer version of this list
Get team charters reviewed

@stratus
Example: Company SRE Split

**Comms**
- Inform team and key partners ASAP
- Establish recurring comms about split

**Staffing**
- Recruit as needed
- Get additional leadership in place

**Team**
- Draft team charters
- Compile a longer version of this list
- Get team charters reviewed
- Shard project work

@stratus
## Example: Company SRE Split

### Comms
- Inform team and key partners ASAP
- Establish recurring comms about split
- ...

### Staffing
- Start recruiting
- Get additional leadership in place
- ...

### Team
- Draft team charters
- Compile a longer version of this list
- Get team charters reviewed
- Shard project work
- @stratus
Example: Company SRE Split

Company SRE is now an organization. It has Slice SRE and Infra SRE as teams.
“Slice and Infra SRE teams are doing so well. We need SRE support for even more services!”
“Slice and Infra SRE teams are doing so well. We need SRE support for even more services!”

**Maybe**

Evaluate impact of scope changes on existing teams load

Can we establish tiers?

**No**

It is out of bounds based in our teams charters
Talk with your leaders

Debug
Debug your team with them as you go

Charters

Peers buy-in
Talk with your leaders

<table>
<thead>
<tr>
<th>Debug</th>
<th>Charters</th>
<th>Peers buy-in</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debug your team with them as you go</td>
<td>Changes to implementation and scope should require their approval</td>
<td></td>
</tr>
</tbody>
</table>
Talk with your leaders

**Debug**
Debug your team with them as you go

**Charters**
Changes to implementation and scope should require their approval

**Peers buy-in**
Get peer leaders to buy-in on any changes as well
Talk with your leaders

Debug
Debug your team with them as you go

Charters
Changes to implementation and scope should require their approval

Peers buy-in
Get peer leaders to buy-in on any changes as well
Lessons I’ve learned

Consider a project manager
Lessons I’ve learned

Consider a project manager

Evaluate change coupling
Lessons I’ve learned

Consider a project manager

Evaluate change coupling

Avoid Flip Flop

@stratus
Lessons I’ve learned

Consider a project manager
Evaluate change coupling
Avoid Flip Flop

Private meetings before team-wide announcements
Lessons I’ve learned

Consider a project manager

Evaluate change coupling

Avoid Flip Flop

Private meetings before team-wide announcements

Organization and teams identity may diverge
Recap

**Set goalposts**
Team charter with goalposts

**Self Eval**
Evaluate implementation details and impact of scope changes
Evaluate implementing tiers of service

**Safety first**
Establish Psychological safety, e.g. formalize project sharding first
THANKS!

@stratus

Contributors
Adrian Hilton, Cheryl Kang, David Huska, Dmitry Nefedkin, Gwendolyn Stockman, Henry Robertson, Jesus Climent, Kent Kawahara, Liz Fong-Jones, Marilia Melo, Meg McNally, Myk Taylor, Stephanie Yan, Tanya Reilly, Todd Underwood, Vivek Rau