Boiling the Frog

Implementing a modern message based architecture without anyone noticing

Hazel.Bohon@particular.net

Who Am I?

(How did I get here?)

- Software engineer @ Particular Software
- Survivor of multiple software modernization/migration projects

“all modern applications are alike, but all legacy applications are dysfunctional in their own unique ways.”

Tolstoy, Anna Karenina (kinda...)

Introduction

Overview of the Session

- Understand the outcomes you want to achieve
- Decide how to break the problem up into smaller parts
- Successfully deliver the parts
- Change the organization to allow this to happen on an ongoing basis

Change is hard

Software Modernization

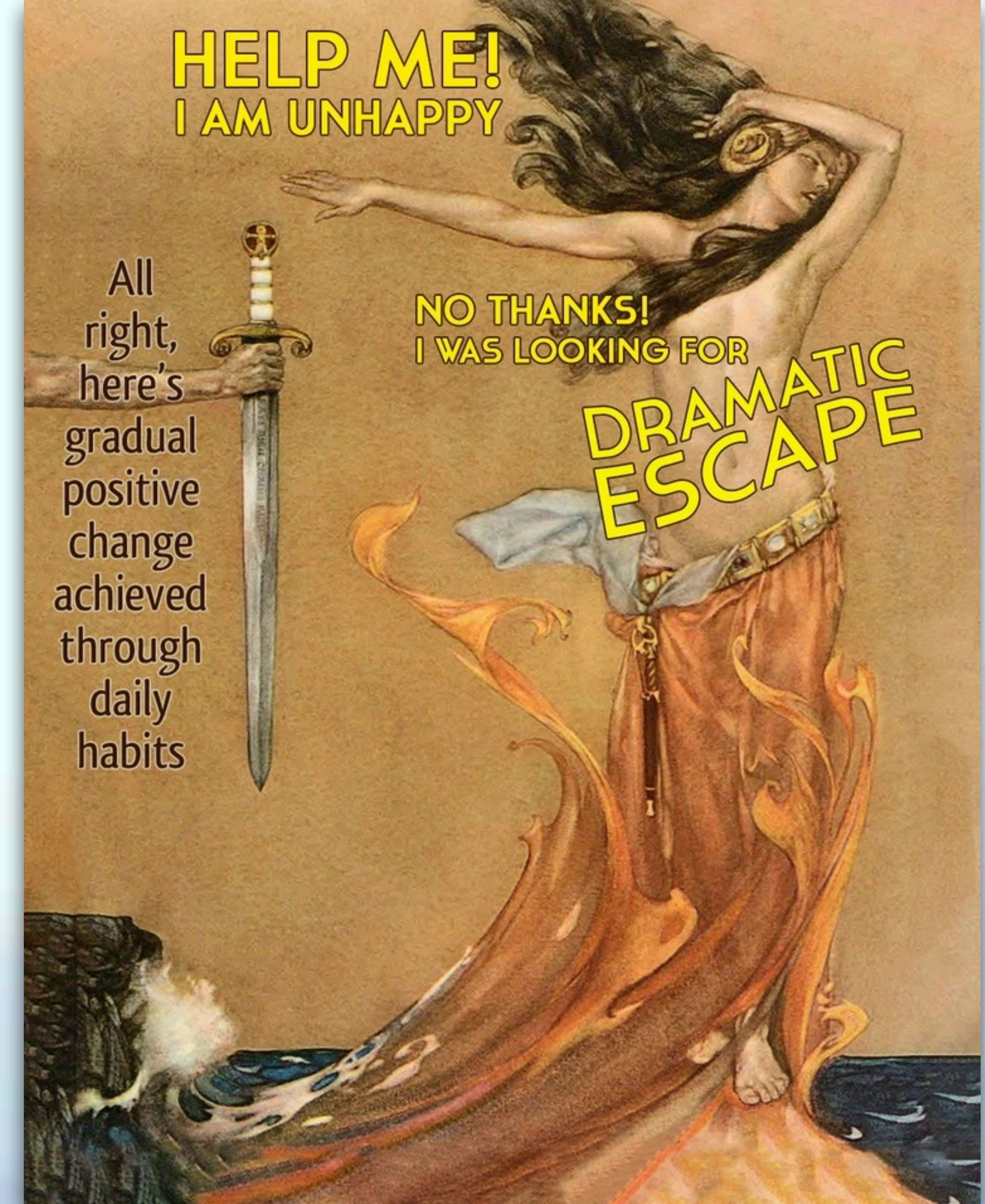
- Easier to maintain code
- Easier to onboard new people
- More reliable, more secure, more robust
- Easier to change and deliver value to the business as a whole

Why do orgs resist this?

- People are skeptical of the unknown
- Rewriting the core business logic of an org can take years
- It's difficult to justify the investment when there are no results to show for the work.
- "I've been burned before."
- "If it Ain't broke, don't fix it."

“Big problems get big because systems amplify problems and make them sticky. It’s tempting to decide that a chronic problem is big enough that we ought to declare war on it, drop everything, hyperinvest, hold our breath, and do nothing until it is solved. But that’s not how the problem got here, and it might not be the best way for the problem to be solved.”

Seth Godin


HELP ME!
I AM UNHAPPY
All right, here's gradual positive change achieved through daily habits
NO THANKS!
I WAS LOOKING FOR
DRAMATIC ESCAPE

Migrating the ball of mud

- Incremental process
- You cannot do it all at once
- Organizing functionality into discrete services is a wicked problem

Wicked Problems

1. They do not have a definitive formulation.
2. **They do not have a "stopping rule." In other words, these problems lack an inherent logic that signals when they are solved.**
3. Their solutions are not true or false, only good or bad.
4. There is no way to test the solution to a wicked problem.
5. They cannot be studied through trial and error. Their solutions are irreversible so, as Rittel and Webber put it, "every trial counts."
6. There is no end to the number of solutions or approaches to a wicked problem.
7. **All wicked problems are essentially unique.**
8. Wicked problems can always be described as the symptom of other problems.
9. **The way a wicked problem is described determines its possible solutions.**
10. Planners, that is those who present solutions to these problems, have no right to be wrong. Unlike mathematicians, "planners are liable for the consequences of the solutions they generate; the effects can matter a great deal to the people who are touched by those actions."

Conway’s Law

“[O]rganizations which design systems (in the broad sense used here) are constrained to produce designs which are copies of the communication structures of these organizations.”

Melvin E. Conway, How Do Committees Invent?

Let’s cook a frog

Step Zero

Code Hygeine & Best Engineering Practices

- Trainings
  - Brown Bag Lunches
- Unit tests
- Code reviews
- CI/CD
- We haven’t even turned on the stove

Get a hype man

- A vocal advocate
- Can trumpet your successes

Find a Guide

- Personal connections outside of the org chart
- Water cooler chats
- Affinity groups inside the organization
- Establish unofficial touch points across the org
- Authenticity is key

Step One

Groundwork

- Start publishing events
- Emit messages in places at the end of method calls
- Do not worry too much about picking these up
- You can start off using SQL db tables to store the queue
- The water is only at the temperature of a Jacuzzi.

Step Two

Pub/Sub

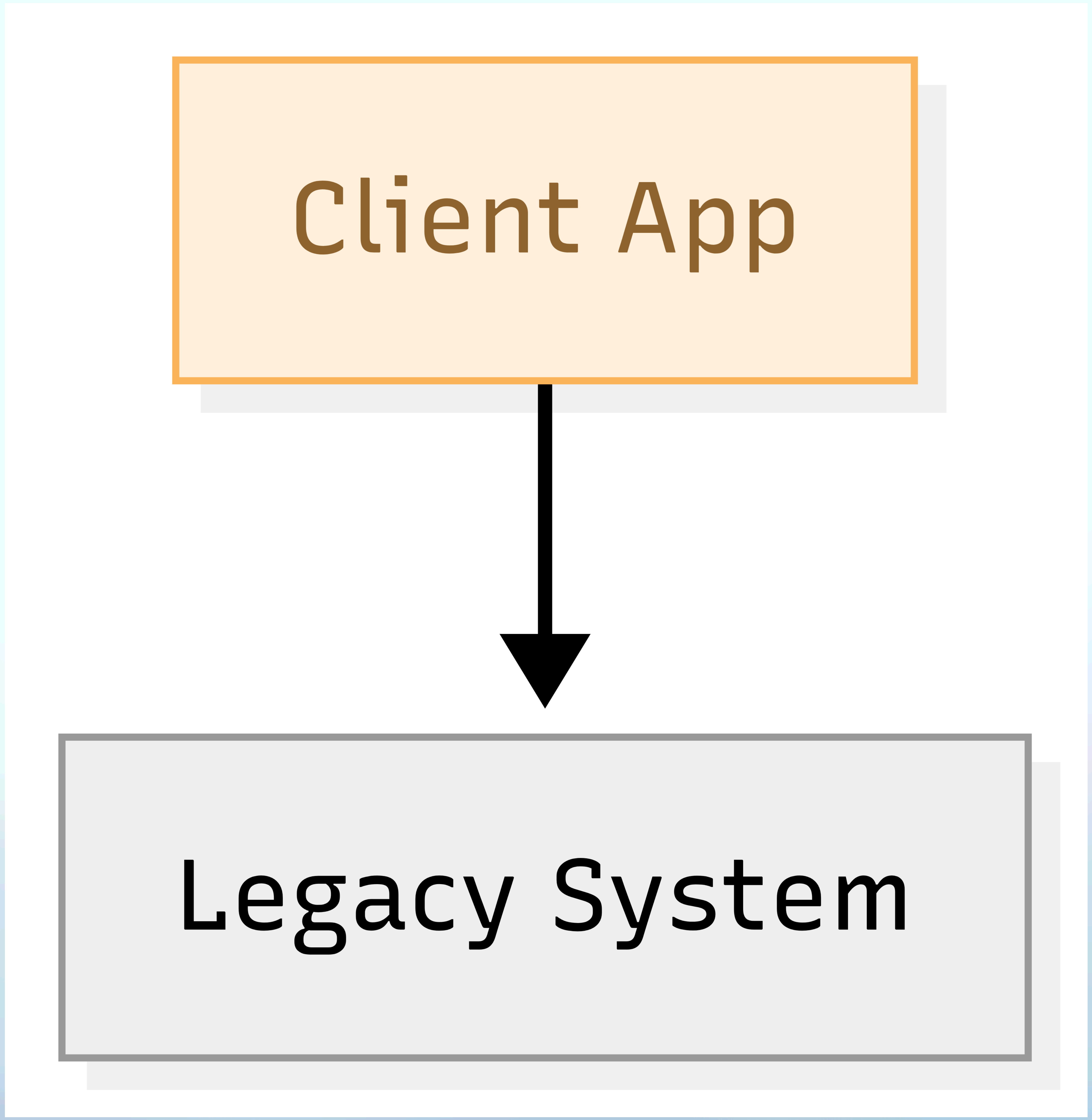
- Create subscribers to the events from Step One.
- Pick Pub/Sub services opportunistically
- Trumpet your successes
- You want the first time they hear about the migration to be when they notice a concrete benefit.
- Investment in resources is still relatively low.
- Establish a common message base.
- The pot has been brought to a low boil.

Step Three

Carve Out Pieces

- There will be cycles of destabilization / restabilization, pace yourself
- “As slow as possible, as fast as necessary”
- Be prepared to redraw service boundaries.
- Set up a facade, strangler fig pattern

Strangler Fig Pattern



Step Four

Divide the database

- Properly isolated services have governance over their data
- Once you understand the partitioning of your services, you can think effectively about splitting up the data.
- Moving things into separate tables > schemas > databases > technologies
- This has been put off to last because this is often the most tricky and time-consuming part

Wrapping Up

* Change is hard

* It is never over

* No frogs were harmed in the making of this presentation.