How to Improve Your Service by Roasting It

Jake Welch
<jawelch@microsoft.com>

Caskey L. Dickson
<caskey@microsoft,twitter,gmail,github,etc...>

Microsoft Azure SRE
Brief History of SRE @ Azure

• 2008 Azure Public Launch
• 2008 - 2014 Divergence of DevOps approaches
• 2014 SRE Pilot started
• 2015 Dedicated SRE organization formed

Enter the challenge of adapting SRE in an established organization
Engaging with Product Teams

• How do you get a product team to open up and work with you?
• Only they know where debt lies, what it looks like, where their service fails
• Is there a common understanding of SRE, agreement on goals?

*We can't help you if you won't tell us where it hurts*
Service Roast

Pronunciation: "sər-vəs" "rōst"

n.

A series of meetings at which a service is subjected to good-natured but frank discussions to uncover design/process flaws, scale limits or other shortcomings
What is a Service Roast?

Goal: Expose and understand the warts, wrinkles, design flaws, shortcomings and problems everyone knows a service has but doesn’t want to talk about

Covers the entire service lifecycle from Development to Disaster Recovery

Output: List of flaws, issues, opportunities for improvement, and understanding

You can and should do this for SRE services!
Why Do This?

• Builds relationships and trust between the teams
• SRE learns about the service
• Dramatically speeds up ‘newbie to expert’ process
• Exposes details that otherwise would be difficult (or painful) to learn of
• Creates a shared backlog of improvements
Guidelines: Working Together

Requires investment from SRE and product teams: everyone is there to participate

- Get real contributors in the room (go away managers)
- Put away phones, laptops
- End to end process requires ~10 hours
- Meet over several weeks, not a single day
- More than 1 hour is too long; 45 minutes works well
Guidelines: Tone

Successful engagement requires clarity of purpose and tone

• Not an attack on the service
• Not a judgment of past choices
• Focus on ‘How’ questions not ‘Why’ questions
  
  Why’s can be seen as judgmental

• Every participant must understand this
Example Questions

❌ Why did/didn’t you… ?

❌ Why don’t you instead… ?

❌ Why can’t you just… ?

❌ Why aren’t you simply… ?

✅ How does ${feature} work?

✅ When do these two pieces communicate?

✅ What part of the system handles ${feature}?

✅ Where are user requests routed?
Roles

- **Service Owners**
  SME experts on service providing insights

- **Roast Participants**
  Ask questions, gain clarity on service

- **Scribe**
  Keeps track of interesting tidbits, actions, learnings

- **Moderator**
  Impartial observer not otherwise involved in the engagement
The Moderator

• A designated impartial observer
• Focuses on tone and body language of participants
• Monitor language to avoid attacks
• De-escalate conversations as necessary
• Decides when to call the meeting off

Strongly recommend implementing this role
Meeting Agenda

Choose a single area or subsystem to drill into

• Moderator provides overview of guidelines and sets tone

• Area SME kicks off with an overview using whiteboards, diagrams as needed

• Sessions are interactive: ask questions, clarify, dispel misinformation

• Moderator keeps conversation on topic

• Scribe keeps track of off-shoots for future meeting topics
Service Roast Sample Topics

• Service overview: What is it, who uses it, where does it fit in overall
• Architectural overview - confirm up and downstream dependencies
• Development process - tools, source control, library dependencies, build, test
• Capacity planning - how do you scale, how do you load test?
• Deployment & configuration management practices
• Monitoring, Logging, Diagnostics
• SLAs, SLOs, KPIs, etc.
• Production playbook, disaster recovery/high availability, backup/restore
Meeting Closure

• At the end of the meeting, the next topic is chosen and adjustments are discussed for future sessions (new topics, participants, etc.)

• After each meeting, the scribe summarizes key learnings and opportunities identified in a centralized doc

• At the end of the series, the identified items should be jointly prioritized for bugs/tasks opened
Gotchas

- Things can be said in the room that don’t leave (except the fix)
- Don’t do this if you think it will degrade relationships between the teams
- Each service will be at a different maturity point in each area - that’s ok!
- Don’t compare one service to another
- When the product team is talking to each other, don’t stop them - listen harder
Summary

- A Service Roast can be a great tool to safely gain E2E service understanding
- Expectations and tone are critical success components
- Managing emotions is critical to a safe discussion environment
- Multiple, 45 minute meetings are best to cover all areas
- The moderator role helps smooth over bumps in the process
Questions?