Arrested Development
The awkward adolescence of a microservices-based application

Europython 2015
Scott Triglia
yelp

The Company
77M reviews
142M monthly unique users
Your Speaker

Scott Triglia
@scott_triglia

4 years with Yelp
Search, ML, Services
Yelp Transaction Platform → The Product
Yelp Transaction Platform (or just “Platform”)
Nick's Crispy Tacos
Delivery $5.00 • Min $9.00
Estimated Delivery Time 45-75 min

ORDERING BY EAT24

Full Menu
Menu hours: 10:30 AM to 9:00 PM

What's Good

- Pescado Taco $4.95
  Baja style fried fish in a corn tortilla with salsa, lime mayonnaise, cilantro, cabbage and onions.

- Carne Asada Taco $3.95
  Grilled steak in corn tortilla with pico de gallo and pinto beans.

- Chips and Salsa $1.25

- Nachos $7.95
  Our house made chips topped with a traditional cheese sauce, black beans, and jalapeños.
Make Your Selection

SHOPTQUIRES.COM

CATHEDRAL MAXI DRESS, $149.00

COLOR

SIZE

DESCRIPTION
Lose fitting midi length scarf maxi dress with adjustable straps. Belt it or leave it loose for a great country vibe. Size small measures 51" at longest length.

ADD TO BAG

Le Meridien San Francisco
Fri, 8/7 - Sat, 8/8 (1 night) | 2 guests

1 Choose Room

1K bed: classic non-smoking: 55 inch led tv :::
Non-refundable
view details

$346

1K bed: classic accessible: visual fire alarm :: visual ...
Non-refundable
view details

$346

1K bed: superior non-smoking: 55 inch led tv :: wi-fi i...
Non-refundable
view details

$355

view all 58 rooms

2 Guest Information
Same as billing information

First Name
Last Name
Summary

Pickup
Nick’s Crispy Tacos

Carne Asada Taco
$3.95 x 2 $7.90

Subtotal $7.90
Tax $0.69
SF Healthy Mandate $0.32

Total $8.91

Your Contact Information

Phone Number 5554379125

Payment Information

Use VISA ****-6444

By proceeding, you agree to our Terms of Service and Privacy Policy. We’ll send your transaction to a partner company to fulfill.

Complete Purchase
Microservices

That Hot Trend
“...an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms...”

http://martinfowler.com/articles/microservices.html
Monolithic python code resisted decoupling
Monolithic python code catered to the lowest common denominator
Monolithic python code was anti-agile
API complexity increases
coupling rises
interactions get murky
process does not scale
So what’s an engineer to do?
• Decoupling
• Defining
• Understanding Production
• Staying Agile
• Decoupling
• Defining
• Understanding Production
• Staying Agile
Old boring problem

Monolithic spaghetti code
Solution: microservices!
New exciting problem
how to share concepts across services
New exciting problem
distributed tech debt
service_type
What product does your business provide and how do they provide it?
service_type

pickup
delivery
service_type

booking_at_customer

pickup

booking_at_business

delivery
Confusing
Pervasive
Convenient, but not designed
Draw boundaries, introduce domain-specific concepts tied to functionality
Lessons
Interfaces are the sum of APIs, shared libraries, and the data that flows through them.
Sacrificing DRYness can be the best choice for overall design
Service interfaces are a great opportunity to intentionally decouple systems.
• Decoupling
• Defining
• Understanding Production
• Staying Agile
Have you ever needed to understand a system and been told "go read the source?"
What about a system which only validates half its interface?
Coming from a python monolith, strong interfaces were quite rare
def checkout(order, price, **kwargs):
    """Process an order."""

    validate_order(order)
    charge_credit_card(order.user, price)
    notify_user(order, **kwargs)
SWAGGER
The World's Most Popular Framework for APIs.
# will be prefixed to all paths
basePath: /v1

produces:
- application/json

paths:
/products:
  get:
    summary: Product Types
    description: |
    The Products endpoint returns information about the *Uber* products offered at a given location. The response includes the display name and other details about each product, and lists the products in the proper display order.
    parameters:
    - name: latitude
      in: query
      description: Latitude component of location.
      required: true
      type: number
      format: double
    - name: longitude
      in: query
      description: Longitude component of location.
      required: true
      type: number
      format: double
    tags:
    - Products
**Swagger Petstore**

This is a sample server Petstore server. You can find out more about Swagger at [http://swagger.io](http://swagger.io) or on irc.freenode.net, #swagger. For this sample, you can use the api key `special-key` to test the authorization filters.

Find out more about Swagger

- [http://swagger.io](http://swagger.io)
- Contact the developer
- Apache 2.0

### pet: Everything about your Pets

<table>
<thead>
<tr>
<th>Method</th>
<th>Endpoint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST</td>
<td>/pet</td>
<td>Add a new pet to the store</td>
</tr>
<tr>
<td>PUT</td>
<td>/pet</td>
<td>Update an existing pet</td>
</tr>
<tr>
<td>GET</td>
<td>/pet/findByStatus</td>
<td>Finds Pets by status</td>
</tr>
<tr>
<td>GET</td>
<td>/pet/findByTags</td>
<td>Finds Pets by tags</td>
</tr>
<tr>
<td>DELETE</td>
<td>/pet/{petId}</td>
<td>Deletes a pet</td>
</tr>
<tr>
<td>GET</td>
<td>/pet/{petId}</td>
<td>Find pet by ID</td>
</tr>
</tbody>
</table>
from bravado.client import SwaggerClient

client = SwaggerClient.from_url(
    "www.myservice.com/swagger.json"
)

pet = client.pet.getPetById(petId=42).result()
Server side - striglia/pyramid_swagger

# In your Pyramid webapp.py
config.include('pyramid_swagger')
Lessons
Interfaces should be intentional
Interfaces should be explicit
Find the **mechanical** things which don’t scale and **automate** them mercilessly.
• Decoupling
• Defining
• Understanding Production
• Staying Agile
Real customer bug report:
“We’re seeing 504s talking to the /user_info API”
Ancient times:

Use logic and whatever logs happen to exist
Better:

Log all incoming API requests to any service
Best:

Every service has a detailed access/error log and tooling to examine them.
errors over time split by exception class

@timestamp per 5 minutes
So what about that customer with the mystery 504?
## YTP: Checkout Fulfillment Dashboard

### YTP: API calls by partner

**Legend**
- ○: EAT24

### YTP: All CF requests

<table>
<thead>
<tr>
<th>Time</th>
<th>request.method_name</th>
<th>request.params.yelp_order_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>July 12th 2015, 12:02:23.821</td>
<td>checkout.v2.orders.fulfillment_user_info</td>
<td>eb8b228a026960981bc05f903f60139e</td>
</tr>
</tbody>
</table>
Realistically:

Don’t require the customer to report issues in the first place
Easy & Flexible Alerting With ElasticSearch [https://elastalert.readthedocs.org](https://elastalert.readthedocs.org)

- 300 commits
- 17 branches
- 34 releases
- 15 contributors

branch: master

Merge pull request #158 from sophicware/feature/smtp_ssl

Qmando authored 6 days ago

latest commit e74ee9e6f
es_host: elasticsearch-hostname
es_port: 14900
index: logstash-errors-%G.%V

type: frequency
num_events: 20
timeframe:
  minutes: 2

alert:
  - "modules.sensu_alert.SensuAlerter"
sensu:
  team: platform
tip: "This alert indicates a large number of errors across the Platform product. See <link to Kibana> for details."
page: true
status: 2  # CRITICAL
es_host: elasticsearch-hostname
es_port: 14900
index: logstash-errors-%G.%V

type: frequency
num_events: 20
timeframe:
  minutes: 2

alert:
  - "modules.sensu_alert.SensuAlerter"
sensu:
  team: platform
tip: "This alert indicates a large number of errors across the Platform product. See <link to Kibana> for details."
  page: true
  status: 2  # CRITICAL
es_host: elasticsearch-hostname
es_port: 14900
index: logstash-errors-%G.%V

type: frequency
num_events: 20
timeframe:
  minutes: 2

alert:
  - "modules.sensu_alert.SensuAlerter"
sensu:
  team: platform
tip: "This alert indicates a large number of errors across the Platform product. See <link to Kibana> for details."
page: true
status: 2  # CRITICAL
Lessons
Logging is a superpower. Use it wisely constantly.
But raw data is not enough! **Visualize and monitor actively.**
These approaches make a world of difference:

• Incident response from days to minutes
• Investigations from $\infty$ to minutes
• Decoupling
• Defining
• Understanding Production
• Staying Agile
Uncomfortable conversation:

“Customers had their orders interrupted. How are you preventing it going forward?”
Understandable response: “Deploy more carefully”
Understandable response: “Expand on call”
How do we ensure the team stays agile as our services grow in complexity?
Pain point: The testing environment is {broken, flaky, not like prod}
Pain point: Tests passed but production broke
Production monitoring is the natural extension of excellent pre-deploy testing.
Easy & Flexible Alerting With ElasticSearch [https://elastalert.readthedocs.org](https://elastalert.readthedocs.org)

- 300 commits
- 17 branches
- 34 releases
- 15 contributors

Branch: master

Merge pull request #158 from sophicware/feature/smtp_ssl

Qmando authored 6 days ago

Latest commit e74ea9e6f
Pain point:
No clue how much time we spend fixing production issues
Pain point:
Tough to argue what changes will make things more robust
Outage Started On *
mm/dd/yyyy, --:-- --
Example: 03/05/2013 11:30 AM

Outage Finished On *
mm/dd/yyyy, --:-- --
Example: 03/05/2013 11:30 AM

Did this impact a specific partner, or all of them? *

Was this a planned outage? *
○ Unplanned
○ Planned

What is the nature of the outage? *
And as with everything else, this must eventually be automated
Lessons
Networks of services are fundamentally harder to test. Prepare accordingly.
Failure will happen. Focus on both identifying and recovering quickly.
Staying agile is easy if your application rarely fails and recovers automatically.
Know your roots
Be explicit
Measure everything
Scale via automation
Yelp/bravado
striglia/pyramid_swagger
Yelp/elastalert
Our accumulated wisdom
Yelp/service-principles
Questions?

@scott_triglia

scott.triglia@gmail.com