Piazza

Piazza

Enabling Government Teams to Share and Access Data in the Cloud in 2016

Michael P. Gerlek
mgerlek@radiantblue.com

4 May 2016

PEE-AHT-ZA

*"A **piazza** is commonly found at the*
meeting of two or more streets...

Shops and other small businesses
are found on piazzas…

Metro stations and bus stops
are found on piazzas…

An ideal place to set up
a business…"

"An infrastructure that
enables geospatial services"

PART ONE

Hello, Piazza!

in which we attempt to justify our existence

A PROBLEM

Within the government, we see:

- Teams of analysts trying to solve problems
- Often limited by
  - Access to new data sources
  - Access to new tools, technologies
  - Access to scalable hardware, …
- And often only a handful of supporting devs
  - JavaScript devs, Python data scientists, …

➔ *Our target customers*

And they're all under orders to

Move To The Cloud!

PIAZZA'S ATTITUDE

Make the "easy" things easy

And let the hard thing be hard

We can do the heavy lifting

Because you shouldn't have to

(Because many of you
don't need all the hard bits)

Common Pains

- Data access
  - *Where is it? How can I access it? Which ones have kittens?*
- Workflows
  - *Do X, and then when Y happens, go do Z.*
- User services
  - *Go run my new, cool algorithm! At scale!*
- Security, Auditing, Logging
  - *Hands off my stuff.*

Piazza's Answer

- Build a set of web services
  - All cloud-native
  - All open source
- Providing essential functionality
  - Load, access, search
  - Orchestrate, execute
- All under one framework
  - Without exposing any of the complexity

An Example

in which we attempt, again, to justify our existence

I Have a Friend…

- The Shoreline Extraction Problem
  - Given a set of (coastal) images
  - Compute shoreline vectors

- *Every N years, do:*
  - Collect a big pile of imagery data
  - Hire a bunch of contractors
  - Draw coastlines, insert into database
  - (Discard all intermediate products, supporting scripts, and process documentation)

AND THEN…

- As the night follows the day…
  - Better shoreline detection algorithms are written
  - More and more imagery is collected
  - People start asking for up-to-date shoreline data in their AOIs

- So ***someone*** has to automate this process
  - *(And that **someone** is not a rock star hipster geospatial dev)*

Hello, Piazza!

And so ***someone*** needs to:

- Harvest metadata from large datasets
- Search for AOIs in all that data
- React when new imagery becomes available
- Run the detection algorithms

Oh, and:

- Do everything in the cloud
- Do everything at scale
- Do everything automatically

The Features

in which we show what Piazza can do

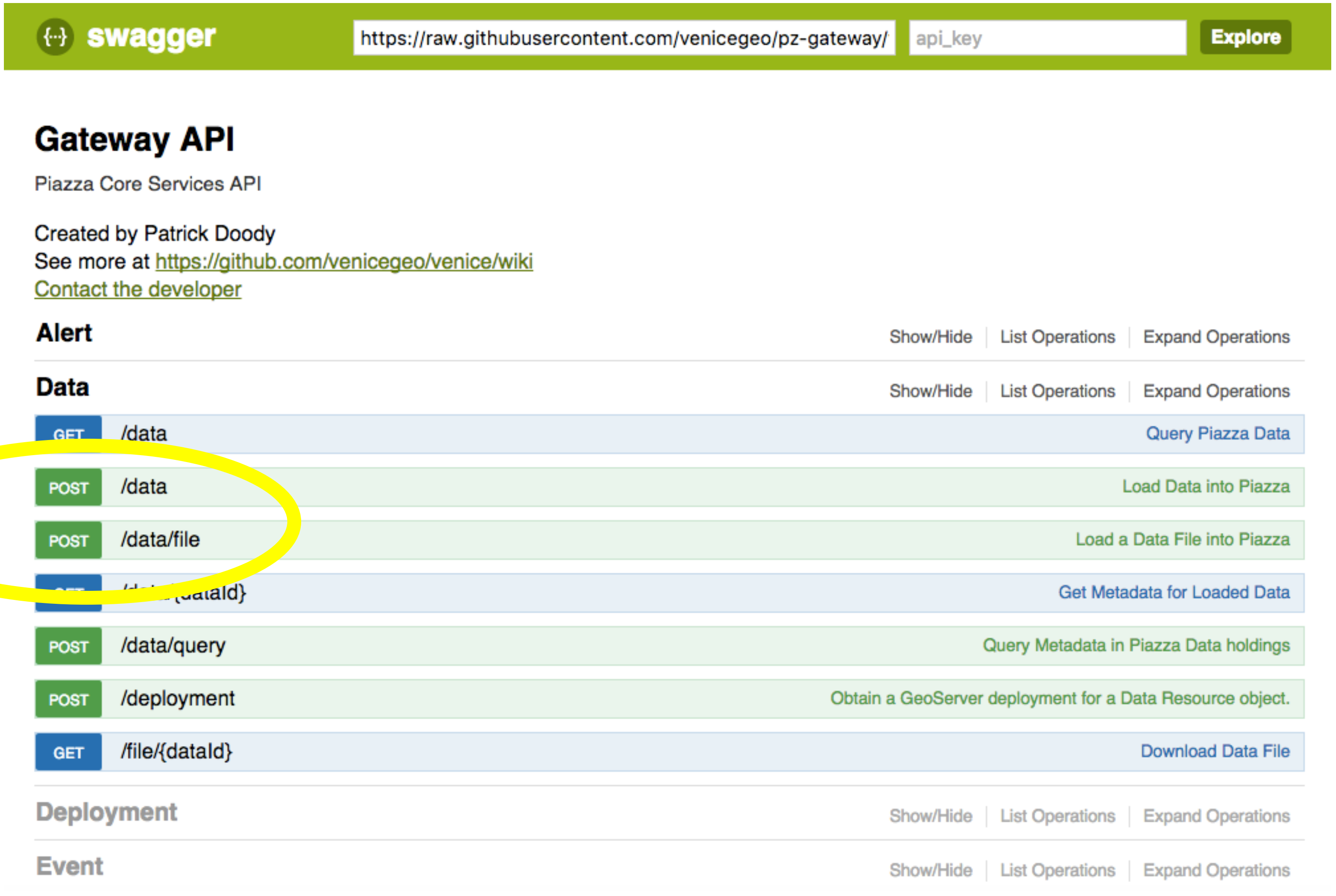
DATA MODEL

- The "No-Host" model
  - Piazza is not intended to be a data hub
  - Rather, Piazza is a proxy of sorts
- Registration and Metadata
  - URL of data source
  - Title, bounding box, fitness-for-use, …
  - Features, images, point clouds
- Resource IDs

N x M?
N + M!

RadiantBlue TECHNOLOGIES

Because REST.



swagger https://raw.githubusercontent.com/venicegeo/pz-gateway/ api_key Explore

Gateway API

Piazza Core Services API

Created by Patrick Doody
See more at https://github.com/venicegeo/venice/wiki
Contact the developer

Alert Show/Hide | List Operations | Expand Operations

Data Show/Hide | List Operations | Expand Operations

GET	/data	Query Piazza Data
POST	/data	Load Data into Piazza
POST	/data/file	Load a Data File into Piazza
GET	/data/{dataId}	Get Metadata for Loaded Data
POST	/data/query	Query Metadata in Piazza Data holdings
POST	/deployment	Obtain a GeoServer deployment for a Data Resource object.
GET	/file/{dataId}	Download Data File

Deployment Show/Hide | List Operations | Expand Operations

Event Show/Hide | List Operations | Expand Operations

A Data Load

```
{     type: ingest,
      host: true,
      data: {
            dataType: { type: raster },
            metadata: {
                  name: test.tif,
                  description: foss4g_test
            }
…}
```

```
curl -X POST …
      test.tif …
      https://pz-gateway/data/file
```

→ *cd504e20-cb90-4ff3-bd4c-f755239f2bfd*

Data Access

- Get Metadata
  - System-extracted, user-supplied
- Request download link
  - *"export as"*
- Request WMS, WFS layers
  - On the fly, via GeoServer, with leasing

```
curl -X GET …
   https://pz-gateway/data/cd504e…
```

SEARCH

- Indexing
  - Metadata extracted during data load
  - File format parsing
  - User-supplied fields
- Querying
  - Elasticsearch DSL (for now)

```
POST /data/query
```

USER SERVICES

Users want to call their own algorithms
- With job management, with scaling

- Registration
  - URL to web API
  - Description of parameters
  - Metadata
  - `POST /service`

    → service ID

USER SERVICES

- Invocation/execution
  - Service ID
  - Parameters
  - `POST /job`

    → job ID

- Status
  - Progress, results
  - `GET /job`

    → resource ID of result data

WORKFLOW SUPPORT

Remember that shoreline example?

- Watch for new data
- Run an algorithm
- On analysis results:
  - If good, insert to database
  - If bad, push to queue for manual checks

Events

"Something happened!"

- Name, ID
- List of parameters

- *System-level*
  - e.g. "image was loaded"
  - Issued by Piazza's internal services

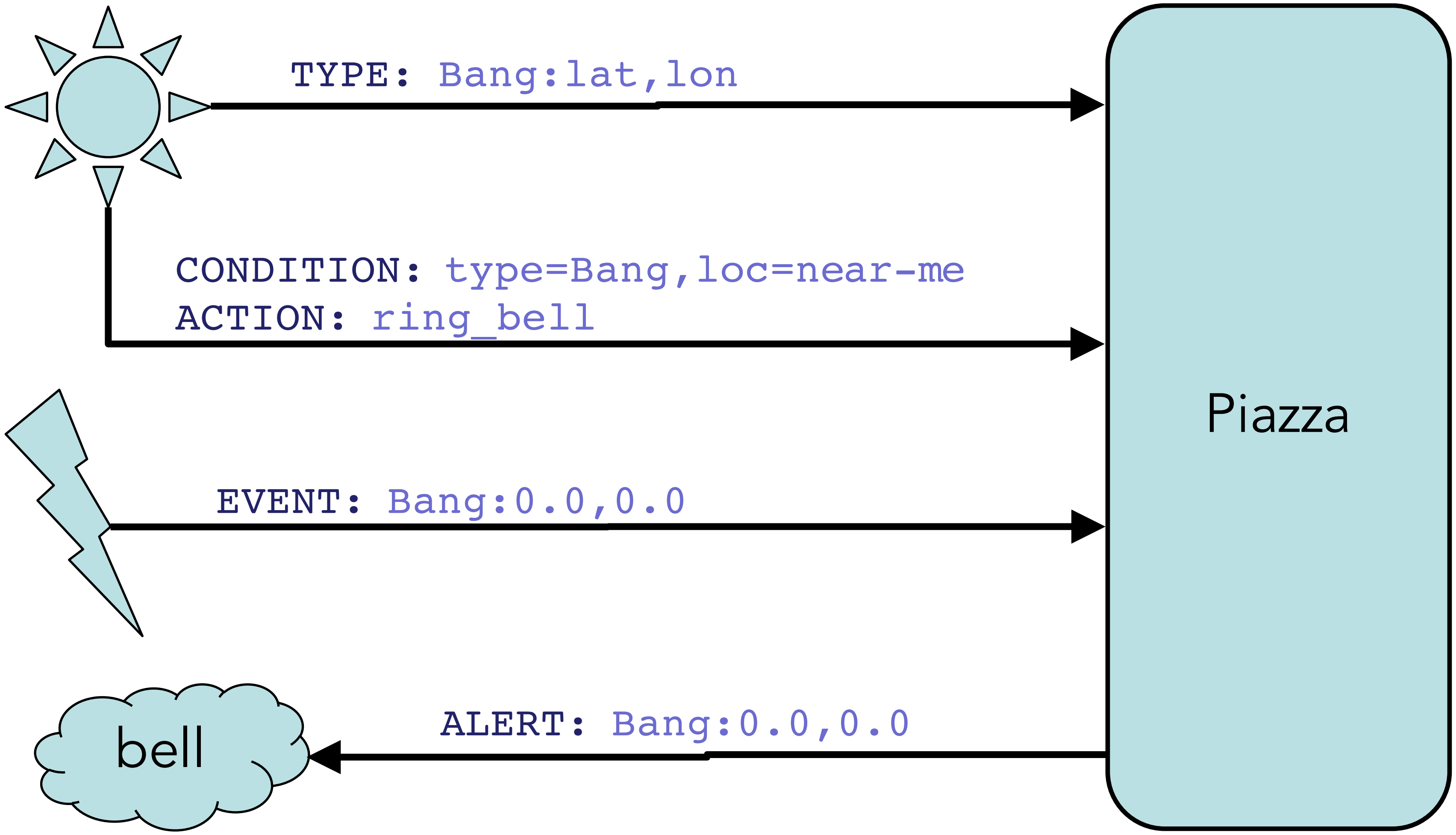
- *User-level*
  - e.g. "interesting new data from my sensor"
  - User-defined parameter list
    - `POST /eventType`
  - Issued by some external entity, i.e. client-side
    - `POST /event`

TRIGGERS

- The Condition
  - If *`eventType = …`*
  - And `parameter >= ...`
- The Action
  - Invoke some service
  - With parameter substitution!
  - Simplest: post to Alert Queue

Think IFTTT

IN PICTURES


TYPE: Bang:lat,lon
CONDITION: type=Bang,loc=near-me
ACTION: ring_bell
Piazza
EVENT: Bang:0.0,0.0
bell
ALERT: Bang:0.0,0.0

Architecture

in which we provide the obligatory boxes-and-lines diagrams

ASIDE

Have you had to work with a complex library controlled by an equally complicated API?

ASIDE

Have you had to work with a complex library controlled by an equally complicated API?

And found yourself writing a simpler API layer just to hide all that complexity?

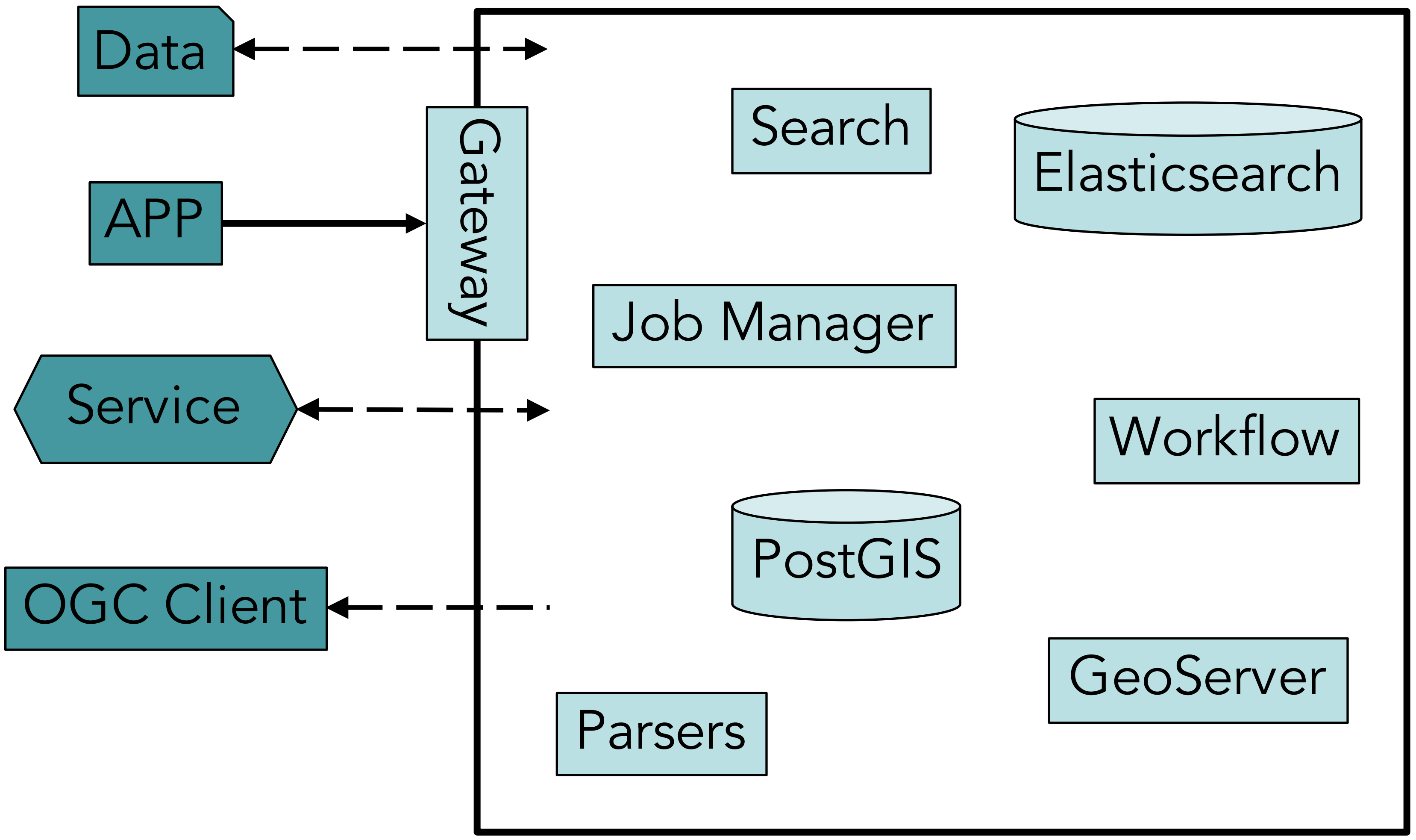
ASIDE

Have you had to work with a complex library controlled by an equally complicated API?

And found yourself writing a simpler API layer just to hide all that complexity?

Hi. Me too.

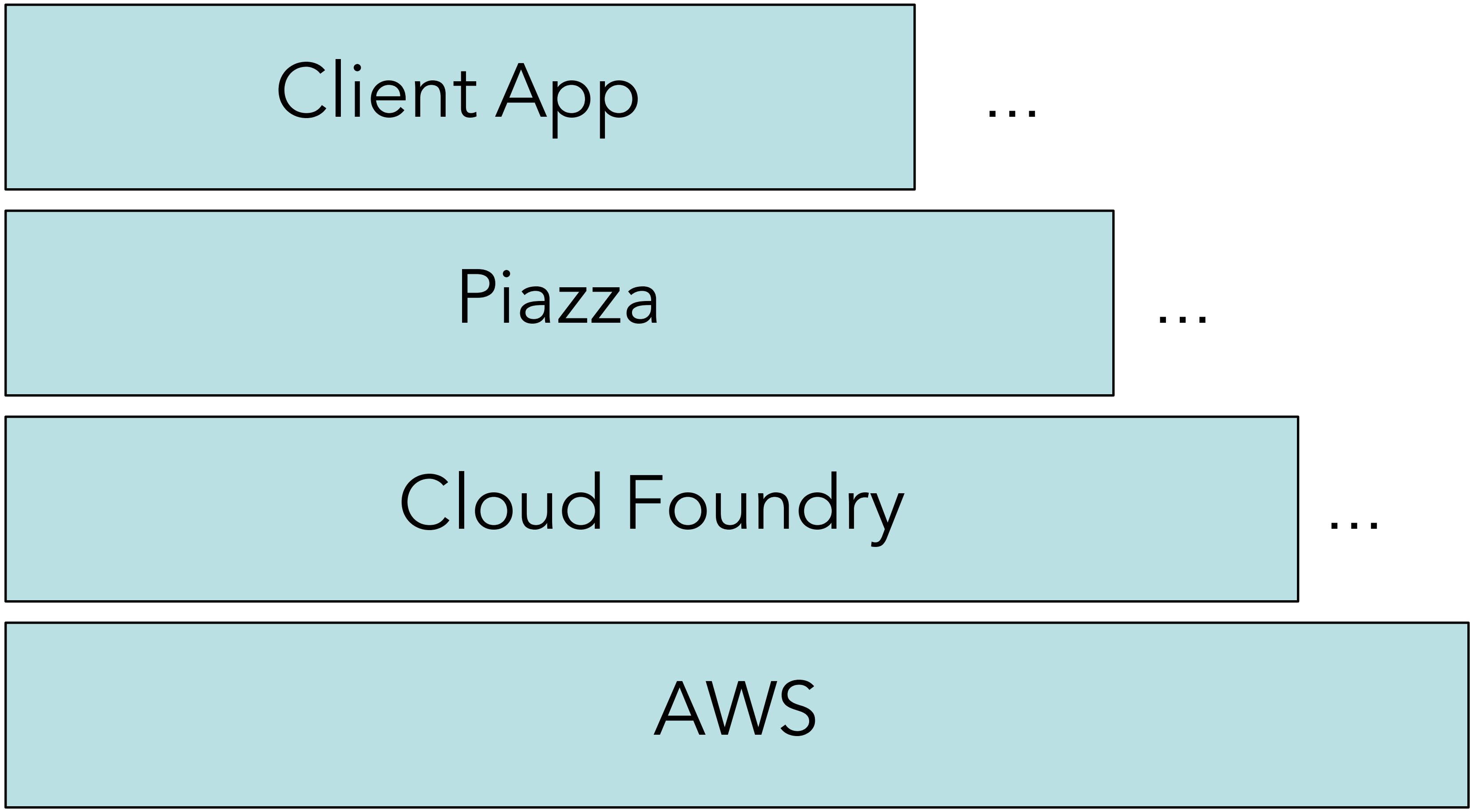
The Black Box



WHAT PIAZZA IS NOT

- An application
  - It’s a platform/architecture/framework/whatever
  - (Indeed, it’s pretty useless by itself)
- A universal solution to everyone’s problems
  - Will always need custom systems
- A “replacement” for anything
  - Focus is on new needs, new workflows
- Any new technology or rocket science
  - Use existing technologies and best practices

The Layers



CLOUD STACK

- We build on top of Cloud Foundry
  - *For now, anyway*
- Standing up Piazza is nontrivial
  - GeoServer, PostGIS, Elasticsearch, …
  - 20+ microservices
- Which is okay(ish) if Piazza is hosted for you
  - But will make you very sad otherwise

We have a lot of work to do here.

The Next Steps

in which we preview next year's FOSS4G talk

STILL TO COME

- User services
  - URLs should really be deployable objects
  - It's Piazza's job to stand up and scale
- Load-time services
  - User-supplied file formats, feature filters, …
- Platform deployability
  - OpenShift, et al
- Enterprise-level cataloging
  - Harvesting metadata in bulk
  - Standards #makeitstop #justshootmenow

And Also

- Security
  - *Umm, yeah. That.*
- Developer docs
- *And of course*
  - *Better use cases*
  - *Better users*
  - *A community*

<a href/>

pz-docs.venicegeo.io
/userguide

pz-swagger.venicegeo.io

RadiantBlue TECHNOLOGIES

CREDITS

REINVENTING GEOSPATIAL

Prominent Edge

RadiantBlue TECHNOLOGIES
A Company of The Radiant Group

imagematters

(AND MANY MORE)

Thank you.

Questions?

Piazza