goto;

Adaptive Architecture

Reimagining API Federation & Integration

About Me

Marty Pitt
CTO - Notional Consulting

Taxi - taxilang.org
Orbital - orbitalhq.com
Spring Fox (Swagger Spring)

@marty_pitt

“If we’re building machine readable API specs,

why aren’t the machines building the glue?”

Building API integration layers that build and repair themselves.

shop-o-pie.com

Product API

Pies API

Orders API

Buy a Pie via API

Customers API

Girls ‘n’ Guys who buy pies via API

Reviews API

Let Girls ‘n’ Guys who buy pies via API tell lies about the pies they buy... via API

Product API

OAS

Orders API

OAS

Customers API

OAS

Reviews API

OAS

orbital

Product API

OAS

Orders API

gRPC / Proto

Customers API

RAML

Reviews API

SOAP

OAS

Product API

gRPC / Proto

Orders API

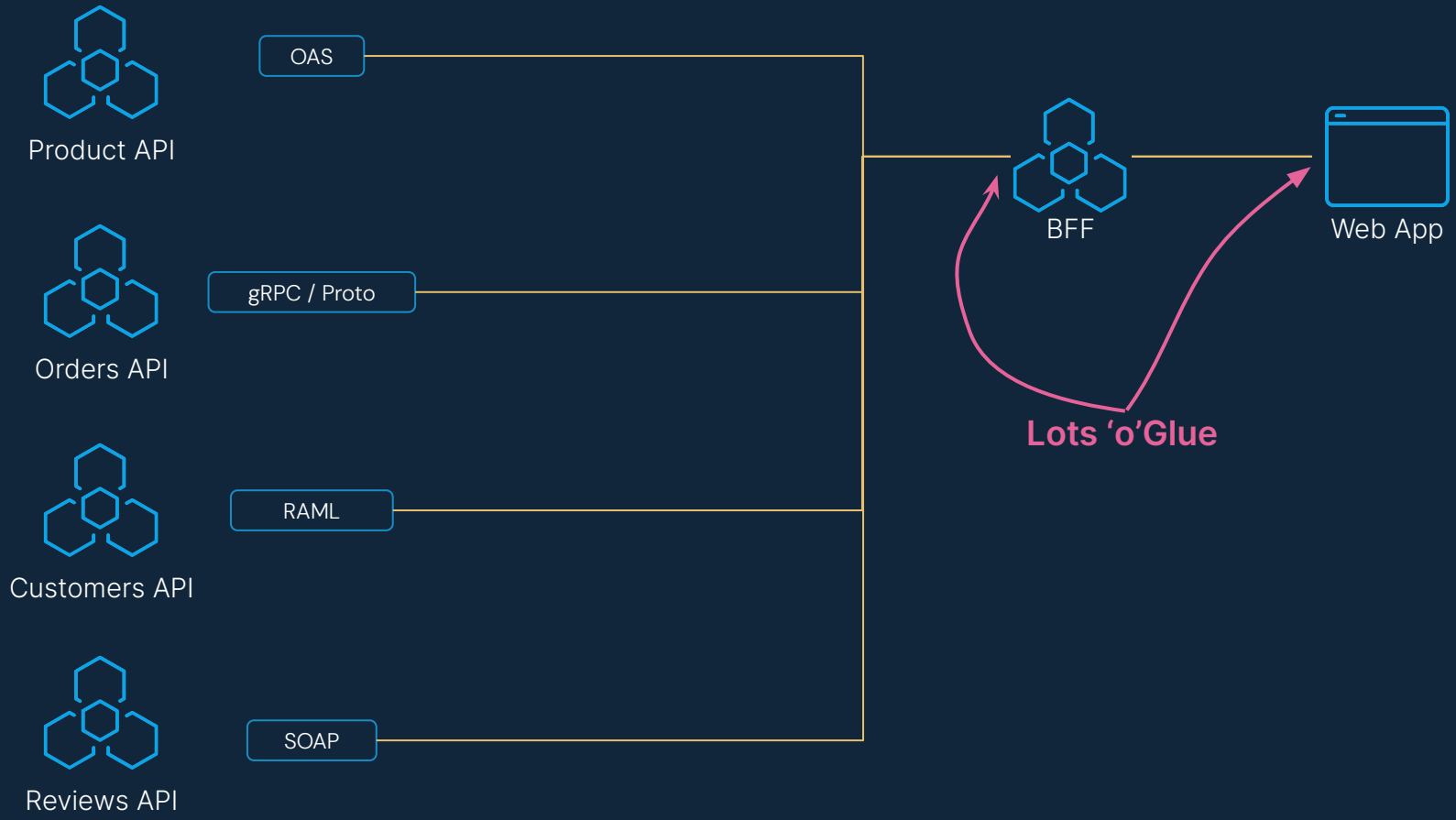
RAML

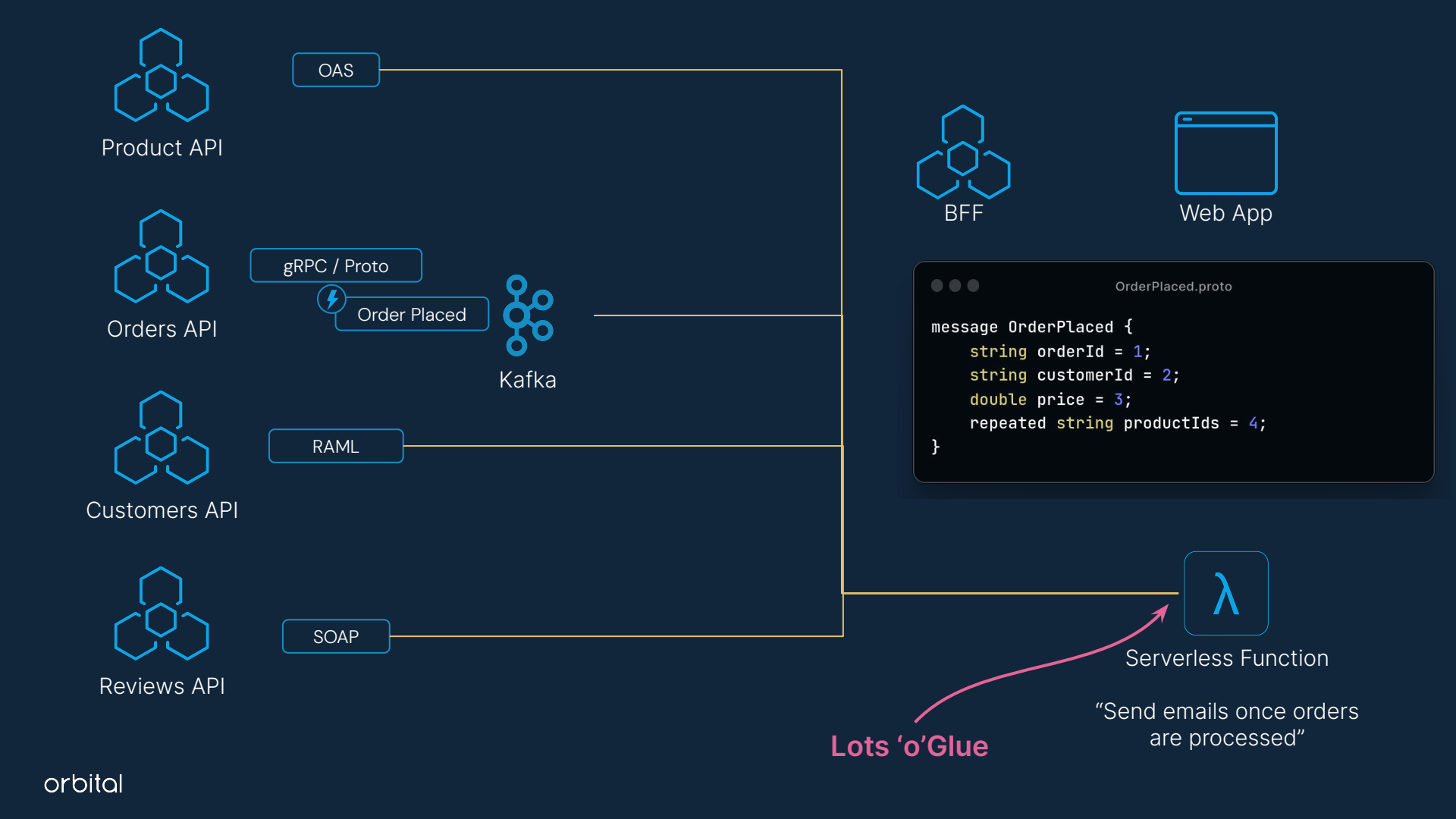
Customers API

SOAP

Reviews API

Gettin' stuff done requires Glue.


Product API
OAS
Orders API
gRPC / Proto
Customers API
RAML
Reviews API
SOAP
BFF
Web App
Lots ‘o’Glue


Product API
OAS
Orders API
gRPC / Proto
Order Placed
Kafka
Customers API
RAML
Reviews API
SOAP
BFF
Web App
OrderPlaced.proto
message OrderPlaced {
string orderId = 1;
string customerId = 2;
double price = 3;
repeated string productIds = 4;
}
λ
Serverless Function
"Send emails once orders are processed"
Lots 'o'Glue
orbital

Let's talk about Glue.

orbital

Gettin’ stuff done requires glue.

Time Consuming
Low Reuse
Breaks.

Writing Glue code takes time

Find & Grok the APIs
Generate the clients
Orchestrate & sequence

Observability
Testing

Parallelism
Caching

Retries
Blocking

And all the work to create it

Glue Code isn't often reused

BFF

Serverless Function

"Send emails once orders are processed"

Glue Code is tight coupling

Glue Code is tight coupling

Changing Producers
Breaks Consumers

“If we’re building machine readable API specs,

(and we agree that building glue sucks)

why aren’t the machines building the glue?”

Because they can't.

Because our APIs aren't rich enough.

APIs tell us...

- Where to find it
- Protocol (http, https, tcp) & Verbs
- How to decode (JSON / Proto)
- Field names in a map

Transport & Structure

APIs don't tell us...

- What stuff means
- How stuff relates

Semantics

When we write Glue code,
We create **mental maps** between
Structure & Semantics

machine readable
We just don't have good ways of
capturing this.


Prepare to be amazed

Contract

```
model Customer {
   id : Int
   name : String
   email : String
}
```

Data

```
{
  “id” : 123
  “name” : “Jimmy”
  “email” : “j@foo.com”
}
```

Contract

```
model Customer {
  id : Int
  name : String
  email : String
}
```

Encoding

Data

```
{
  “id” : 123
  “name” : “Jimmy”
  “email” : “j@foo.com”
}
```

Contract

```
model Customer {
   id : Int
   name : String
   email : String
}
```

Structure

Data

```
{
  “id” : 123
  “name” : “Jimmy”
  “email” : “j@foo.com”
}
```

Contract

```
model Customer {
   id : Int
   name : String
   email : String
}
```

Data

```
{
  “id” : 123
  “name” : “Jimmy”
  “email” : “j@foo.com”
}
```

“What do I call this person?”

Contract

```
model Customer {
   id : Int
firstName : String
   email : String
}
```

Data

```
{
  “id” : 123
  “firstName” : “Jimmy”
  “email” : “j@foo.com”
}
```

“What do I call this person?”

Contract

```
model Customer {
   id : Int
givenName : String
   email : String
}
```

Data

```
{
  “id” : 123
  “givenName” : “Jimmy”
  “email” : “j@foo.com”
}
```

“What do I call this person?”

Contract

```
model Customer {
   id : Int
ix_332 : String
   email : String
}
```

Data

```
{
  “id” : 123
  “ix_332” : “Jimmy”
  “email” : “j@foo.com”
}
```

“What do I call this person?”

Contract

```
model Customer {
   id : Int
givenName : String
   email : String
}
```

Data

```
{
  “id” : 123
  “givenName” : “Jimmy”
  “email” : “j@foo.com”
}
```

“What's their email address?”

Contract

```
model Customer {
   id : Int
givenName  : String
  contacts  : {
    email: String
   }
}
```

Data

```
{
  “id” : 123
  “givenName” : “Jimmy”
  “contacts” : {
    “email”: “j@foo.com”
   }
}
```

“What's their email address?”

orbital

When we write Glue code,
We create **mental maps** between
Structure & Semantics

Field names are Structural, not Semantic.

They help people understand meaning, but aren’t semantic contracts

Field names are Structural, not Semantic.

We can change a field name, without changing the meaning.

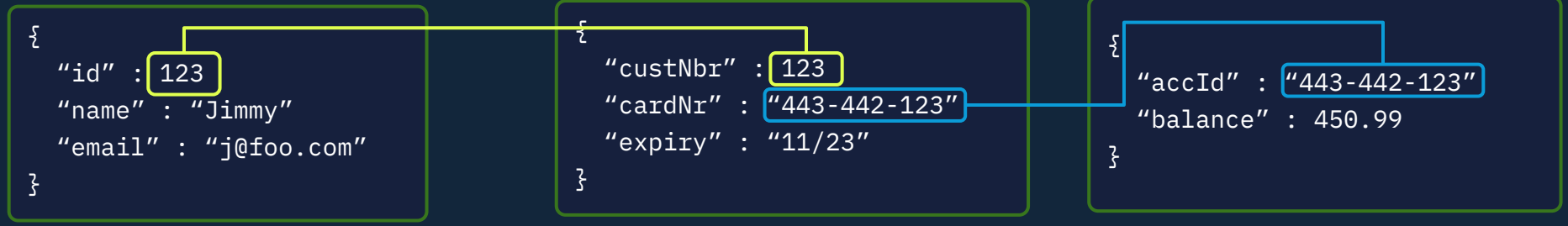
Field names are Structural, not Semantic.

But it does break decoding.

```
model Customer {
  id : Int
  name : String
  email : String
}
```

```
model Cards {
  custNbr : Int
  cardNbr : String
  expiry : String
}
```

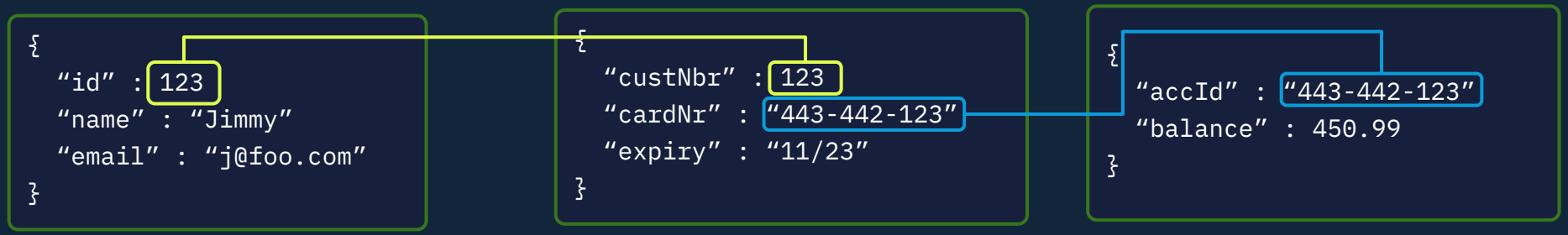
```
model Balance {
  accId : String
  balance : Decimal
}
```



```
model Customer {
  id : CustomerId
  name : String
  email : String
}
```

```
model Cards {
  custNbr : CustomerId
  cardNbr : CardNumber
  expiry : String
}
```

```
model Balance {
  accId : CardNumber
  balance : Decimal
}
```



```
model Customer {
  id : CustomerId
  name : String
  email : String
}
```

```
model Cards {
  custNbr : CustomerId
  cardNbr : CardNumber
  expiry : String
}
```

```
model Balance {
  accId : CardNumber
  balance : Decimal
}
```

```
// Taxi
type CustomerId inherits Int
type CardNumber inherits String
```

Taxi

taxilang.org

taxilang/taxilang

Semantic Metadata for APIs

orbital

Taxi

taxilang.org

Semantic Metadata

orbital

Taxi

taxilang.org

"This thing is the same as that thing."

Taxi

taxilang.org

API Specs:

Transport & Structure

(where & How)

Taxi:

Semantic

(what)

Taxi

taxilang.org

Embeddable in API Specs

(Proto / OAS / SOAP / JsonSchema / DDL)

Semantic Type System

Compiler & Build Tooling

Standalone API Specs

(OAS Alternative / CSV / XML / etc)

```
model Customer {
  id : CustomerId
  name : String
  email : String
}
```

```
model Cards {
  custNbr : CustomerId
  cardNbr : CardNumber
  expiry : String
}
```

```
model Balance {
  accId : CardNumber
  balance : Decimal
}
```

```
// Taxi
type CustomerId inherits Int
type CardNumber inherits String
```

```
// OpenAPI Spec:
Customer:
 properties:
   id:
     type: int32
     x-taxi-type: CustomerId
```

```
// Protobuf
message Card {
 int32 custNbr = 1 [(DataType)="CustomerId"];
 string cardNbr = 2 [(DataType)="CardNumber"];
}
```

```
// Kotlin
@DataType("CardNumber")
typealias CardNumber = String

data class Balance(
   val accId: CardNumber
)
```

taxilang.org

"This thing is the same as that thing."

Capture,
And embed in API specs

“If we’re building machine readable API specs,

(and we agree that building glue sucks)
(and APIs have semantic metadata)

why can’t the machines build the glue?”

They can.

“Given this **Email Address**,
What is the customer’s **Account Balance?**”

TaxiQL:

```
given { EmailAddress = "jimmy@foo.com" }
find { AccountBalance }
```

"Given this **Email Address**, What is the customer's **Account Balance?**"

REST API Customer System

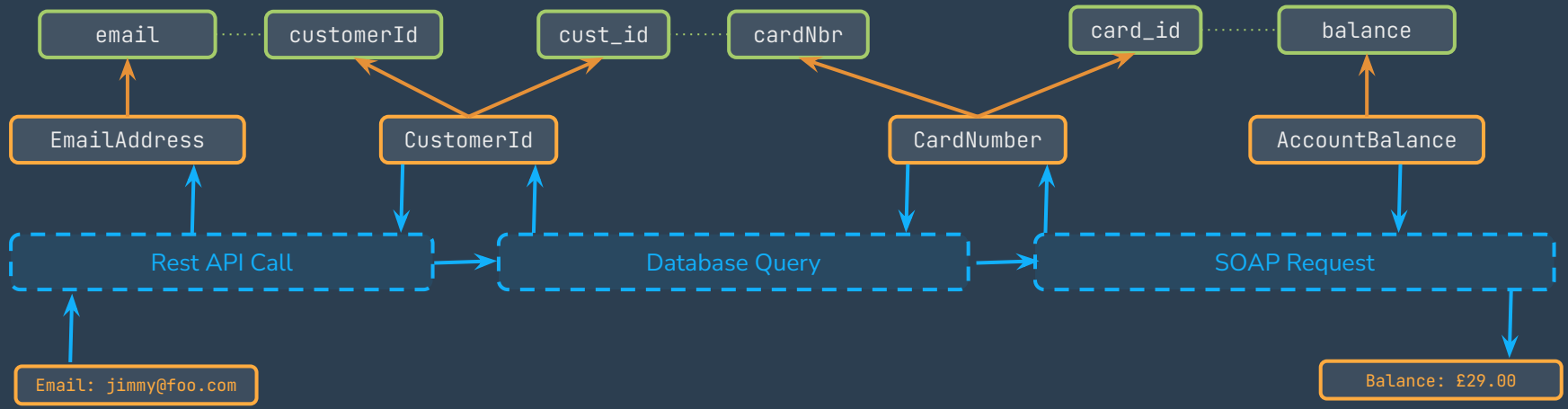
customerId	String	CustomerId
email	String	EmailAddress

SQL DB Cards Table

cust_id	String	CustomerId
cardNbr	String	CardNumber

SOAP API Transaction System

card_id	String	CardNumber
balance	Decimal	AccountBalance



TaxiQL:

Ask for data,
Generate the glue.

```
given { EmailAddress = “jimmy@foo.com” }
find { AccountBalance }
```

Declarative

As services change,
rebuild glue on-the-fly.

```
given { EmailAddress = “jimmy@foo.com” }
find { Customer } as {
  id: CustomerId
  name: upperCase(FirstName + ‘ ‘ + LastName)

  balance: AccountBalance
  last10Transactions: Transaction(
        Count = 10,
        OrderedBy = TransactionDate,
        Order = Desc )[]

  fave: PreferredPieFlavour
}
```

```
given { EmailAddress = “jimmy@foo.com” }
find { Customer } as {
  id: CustomerId
  name: upperCase(FirstName + ‘ ‘ + LastName)

  balance: AccountBalance
  last10Transactions: Transaction(
        Count = 10,
        OrderedBy = TransactionDate,
        Order = Desc )[]

  fave: PreferredPieFlavour
}
```

```
given { EmailAddress = “jimmy@f
find { Customer } as {
  id: CustomerId
  name: upperCase(FirstName + ‘

  balance: AccountBalance
  last10Transactions: Transacti
        Count = 10,
        OrderedBy = Transaction
        Order = Desc )[]

  fave: PreferredPieFlavour
}
```

Consumers define contracts

- Decoupled from producer contracts
- Producers can change, without breaking consumers
- Data sourced from multiple services, glue generated from API specs

```
given { EmailAddress = “jimmy@f
find { Customer } as {
  id: CustomerId
  name: upperCase(FirstName + ‘
  balance: AccountBalance
  last10Transactions: Transacti
        Count = 10,
        OrderedBy = Transaction
        Order = Desc )[]
  fave: PreferredPieFlavour
}
```

Publish Models

Consume Semantics

```
given { EmailAddress = “jimmy@foo.com” }
find { Customer } as {
  id: CustomerId
  name: upperCase(FirstName + ‘ ‘ + LastName)
```

Functions in query layer

- Taxi stdlib with common functions out-the-box
- Extensible function support - Write functions in Java / Kotlin
- Functions run in-process on the query node

```
type UpperCaseName = upperCase(FirstName + ' ' + LastName)
id: CustomerId
name: UpperCaseName
```

Functions in query layer

- Taxi stdlib with common functions out-the-box
- Extensible function support - Write functions in Java / Kotlin
- Functions run in-process on the query node

```
given { EmailAddress = “jimmy@foo.com” }
find { Customer } as {
  id: CustomerId
  name: upperCase(FirstName + ‘ ‘ + LastName)

  balance: AccountBalance
  last10Transactions: Transaction(
        Count = 10,
        OrderedBy = TransactionDate,
        Order = Desc )[]

  fave: PreferredPieFlavour
}
```

Constraints inform API calls

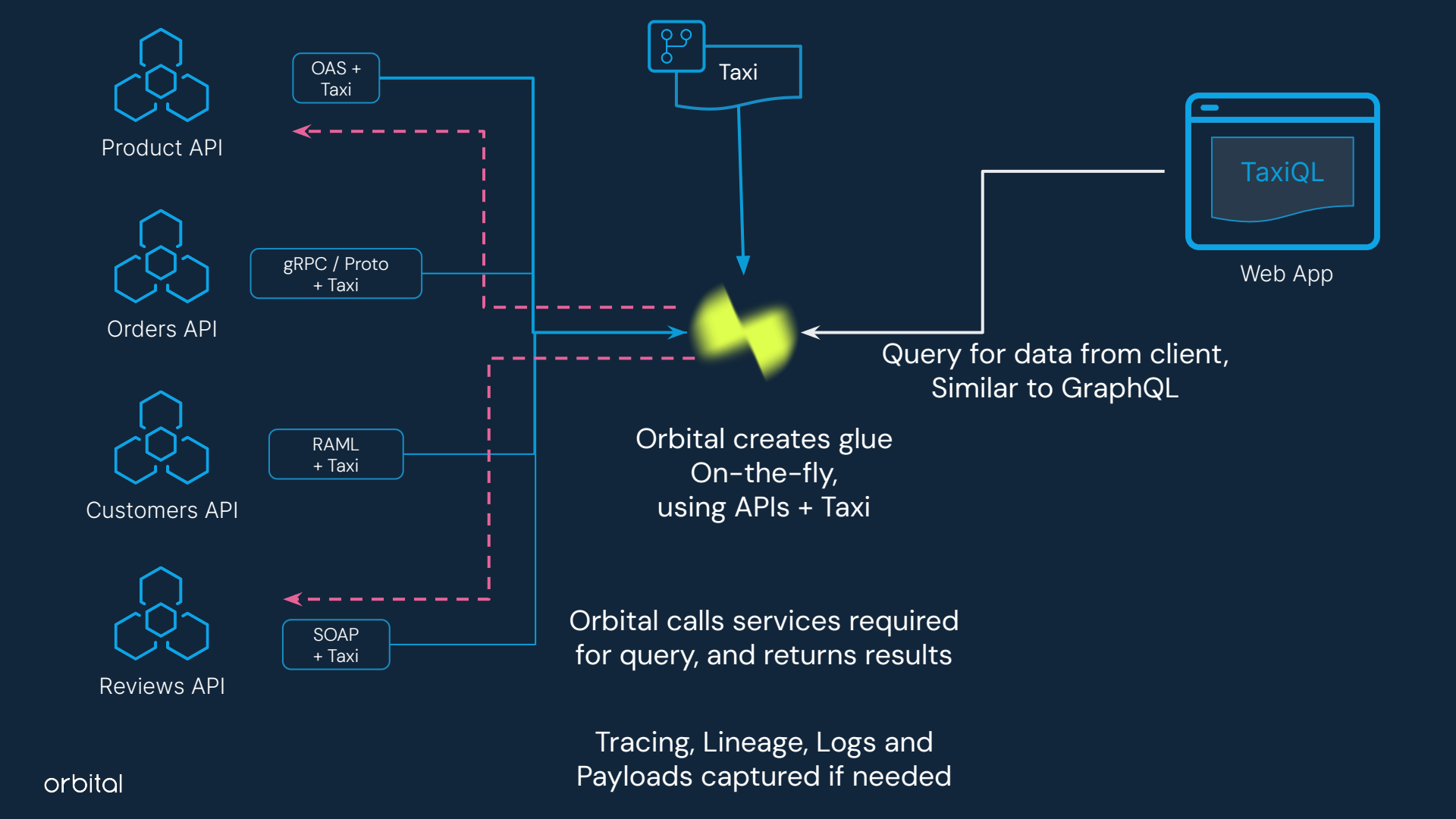
- Tooling to detect insufficient context

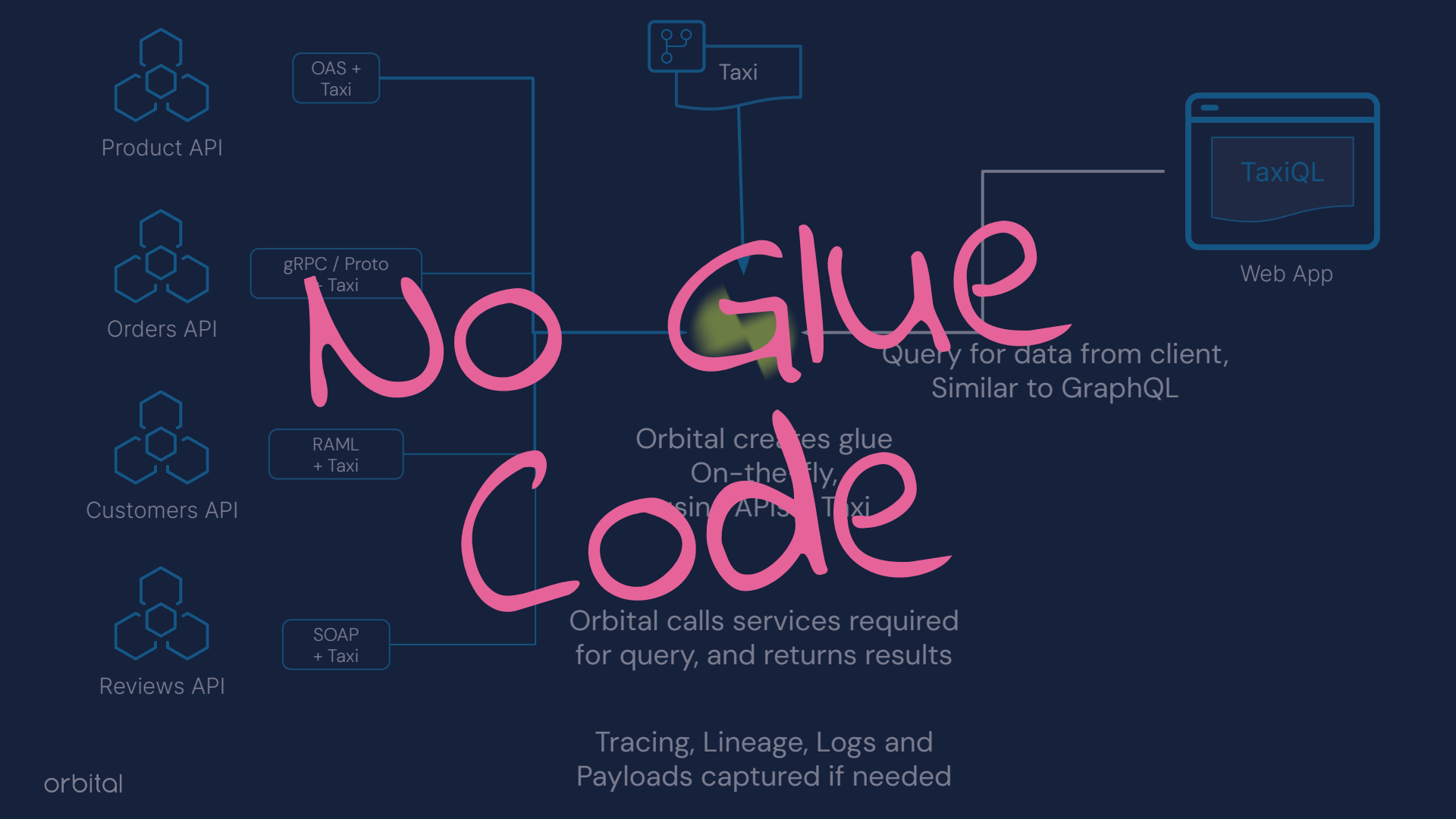
Query Execution.

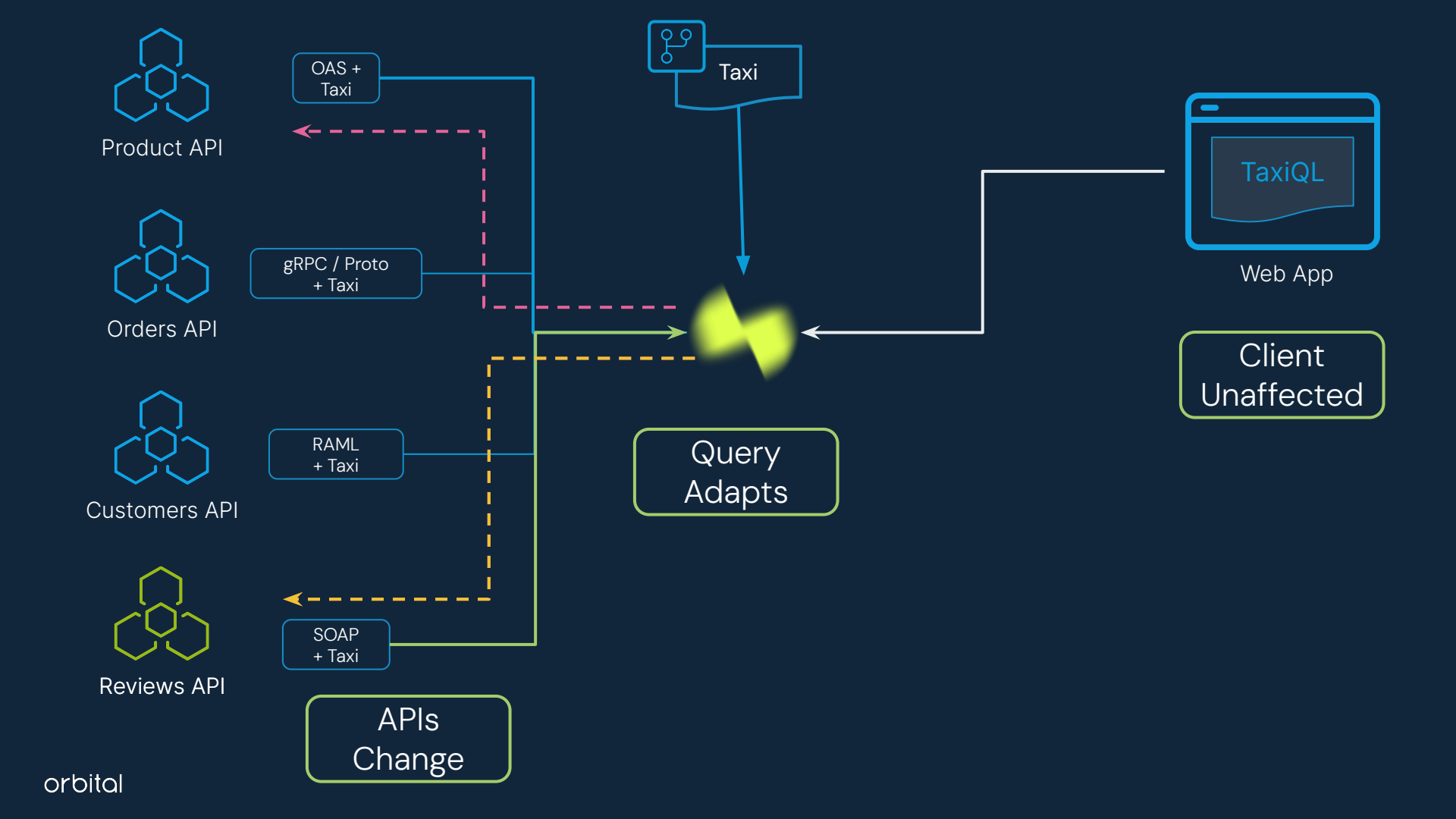
orbitalhq.com

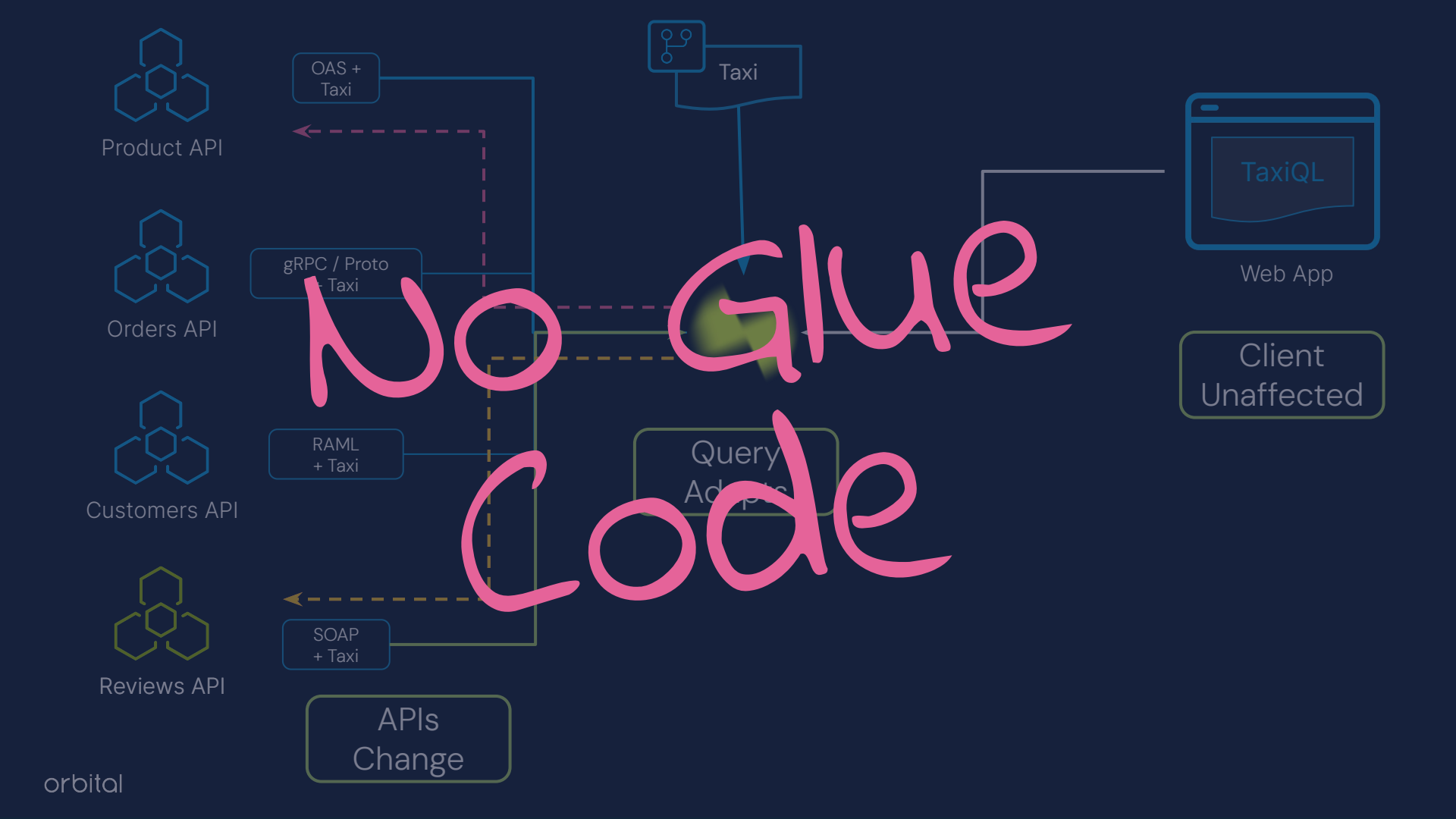
orbitalapi/orbital

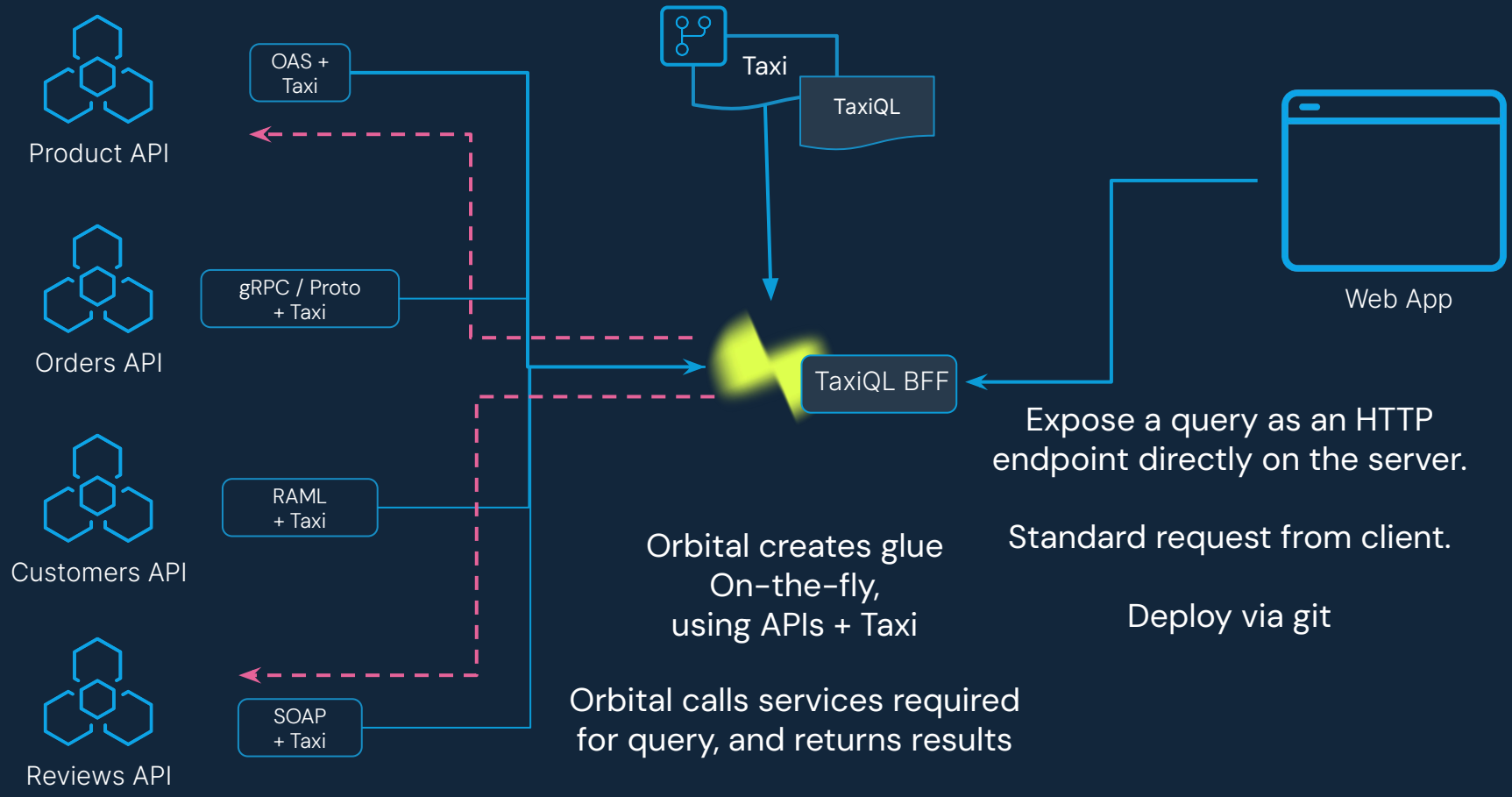
TaxiQL Query Engine & Observability Platform


Product API
OAS + Taxi
Taxi
TaxiQL
Web App
Orders API
gRPC / Proto + Taxi
Query for data from client,
Similar to GraphQL
Customers API
RAML + Taxi
Orbital creates glue
On-the-fly,
using APIs + Taxi
Reviews API
SOAP + Taxi
Orbital calls services required
for query, and returns results
Tracing, Lineage, Logs and
Payloads captured if needed
orbital


Product API
Orders API
Customers API
Reviews API
OAS + Taxi
gRPC / Proto + Taxi
RAML + Taxi
SOAP + Taxi
Taxi
TaxiQL
Web App
No Glue Code
Query for data from client,
Similar to GraphQL
Orbital creates glue
On-the-fly,
using APIs + Taxi
Orbital calls services required
for query, and returns results
Tracing, Lineage, Logs and
Payloads captured if needed
orbital


Product API
OAS + Taxi
Taxi
TaxiQL
Web App
Orders API
gRPC / Proto + Taxi
Client Unaffected
Query Adapts
Customers API
RAML + Taxi
Reviews API
SOAP + Taxi
APIs Change
orbital


Product API
Orders API
Customers API
Reviews API
OAS + Taxi
gRPC / Proto + Taxi
RAML + Taxi
SOAP + Taxi
APIs Change
Taxi
Query
TaxiQL
Web App
Client Unaffected
No Glue Code
orbital



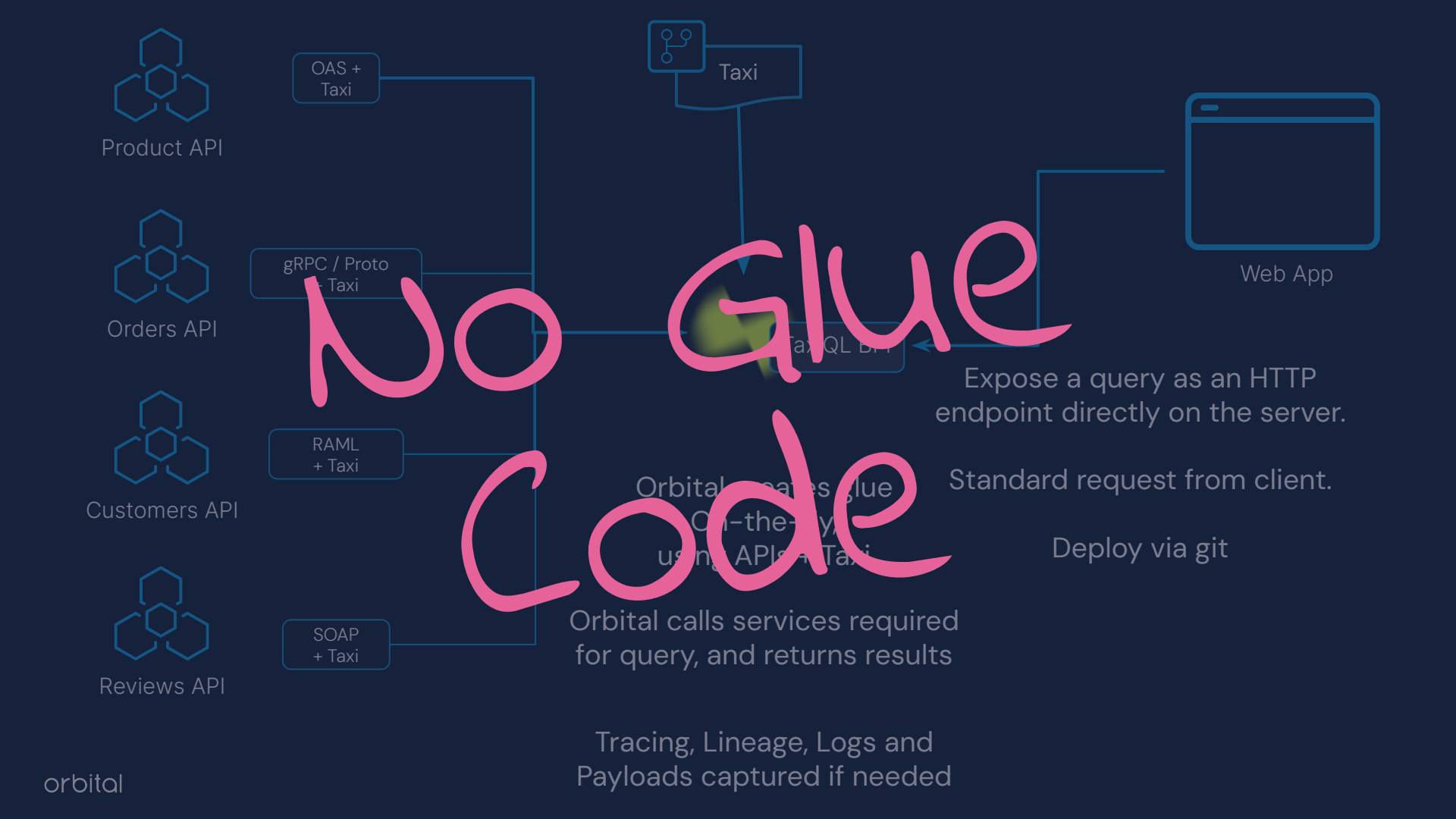
Orbital creates glue
On-the-fly,
using APIs + Taxi

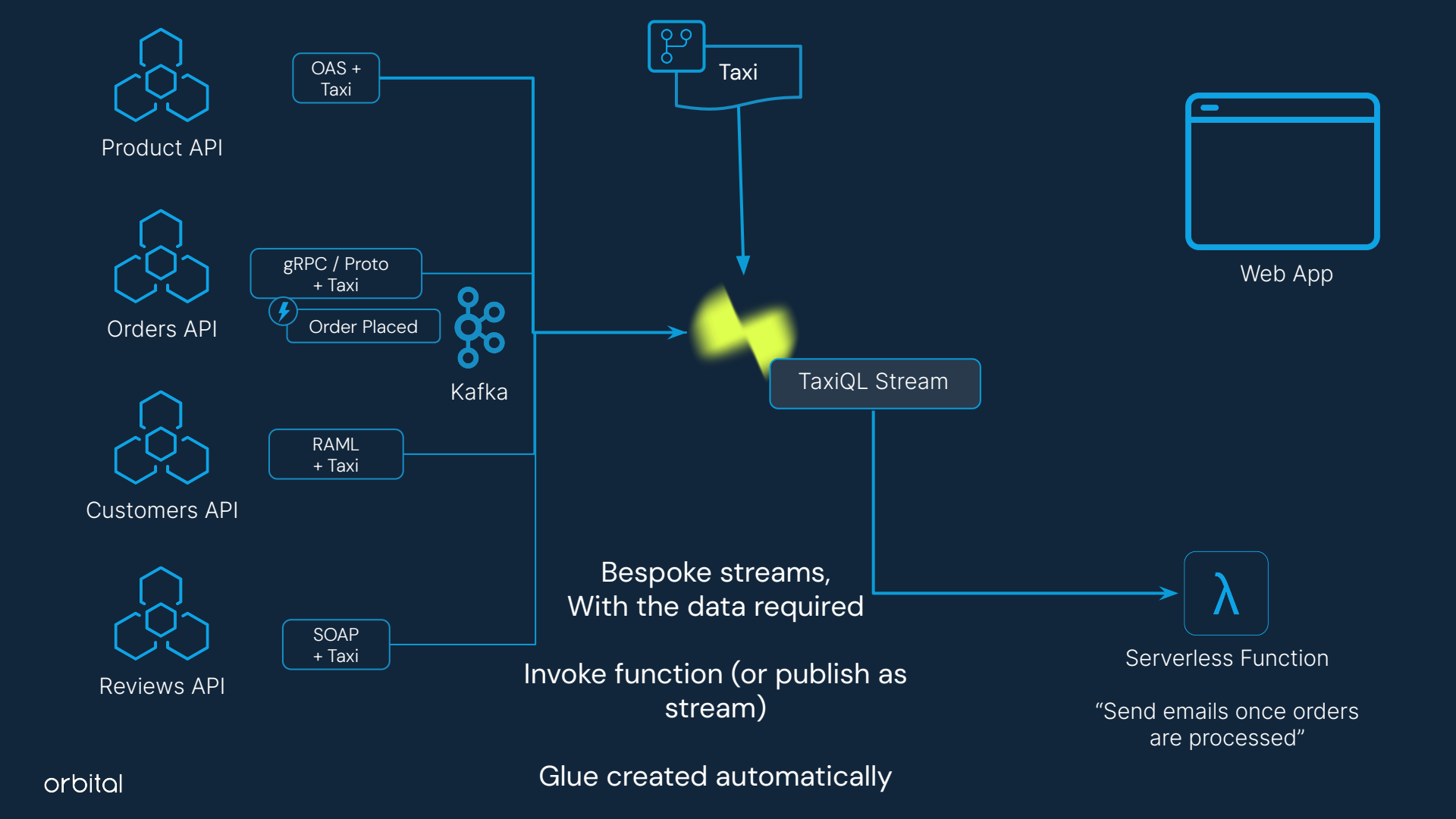
Orbital calls services required
for query, and returns results

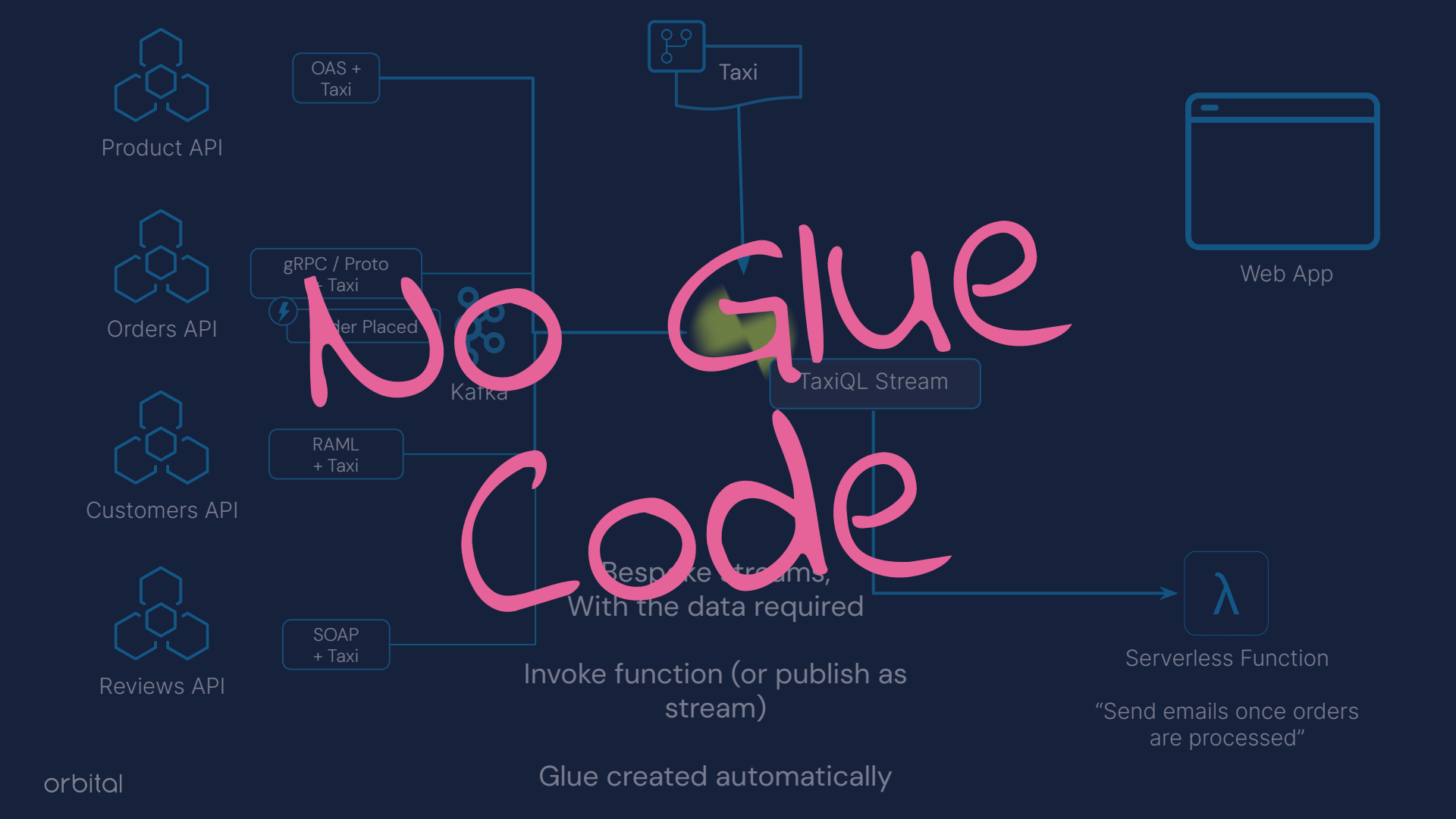
Expose a query as an HTTP
endpoint directly on the server.

Standard request from client.

Deploy via git


Product API
Orders API
Customers API
Reviews API
OAS + Taxi
gRPC / Proto + Taxi
RAML + Taxi
SOAP + Taxi
Taxi
Web App
No Glue Code
Expose a query as an HTTP endpoint directly on the server.
Standard request from client.
Deploy via git
Orbital calls services required for query, and returns results
Tracing, Lineage, Logs and Payloads captured if needed
orbital


Product API
OAS + Taxi
Taxi
Web App
Orders API
gRPC / Proto + Taxi
Order Placed
Kafka
TaxiQL Stream
Customers API
RAML + Taxi
Bespoke streams,
With the data required
Reviews API
SOAP + Taxi
Invoke function (or publish as stream)
Serverless Function
"Send emails once orders are processed"
Glue created automatically
orbital


Product API
Orders API
Customers API
Reviews API
OAS + Taxi
gRPC / Proto + Taxi
RAML + Taxi
SOAP + Taxi
Taxi
Kafka
TaxiQL Stream
Web App
λ
Serverless Function
"Send emails once orders are processed"
With the data required
Invoke function (or publish as stream)
Glue created automatically
No Glue Code
orbital

Taxi

Semantic Metadata for APIs

TaxiQL

Declarative Query Language

Orbital

TaxiQL Engine + Observability

Summary.

Glue Code 👎.

Time Consuming

Low Reuse

Breaks.

Make the machines write it.

API Specs:

Transport & Structure

(where & How)

+

Taxi:

Semantic

(what)

Taxi.

This thing is the same as that thing.

TaxiQL.

Declarative query language.

Orbital

Automated Glue.

Thank You.