AWS re:Invent

SVS308-R

Moving to event-driven architectures

Tim Bray

VP/Distinguished Engineer

Amazon Web Services

AWS re:Invent

aws

amazon

All

EN

Hello, Tim
Account & Lists

Orders

0
Cart

Deliver to TIMOTHY
Vancouver V5Y 2B3

Browsing History | Tim's Amazon.com | Today's Deals | Buy Again | Gift Cards | Help | Registry | Sell

Shop Today's Deals

You are on amazon.com. You can also shop on Amazon Canada for millions of products with fast local delivery. Click here to go to amazon.ca

Hi, Tim

Customer since 2003

Recommendations for you


Your Orders


Electronics


Computers & Accessories


Home & Kitchen

Recently viewed

Wed, Oct 16

See your browsing history

Deals & Promotions

Shop thousands of deals & promotions that ship around the world

Shop now

Shop by Category

Computers & Accessories


Video Games


Baby


Toys & Games

Shop now


AM DESK
AFE
AM DESK
AFE
INDUCT
62 - 61
2F
8

Amazon.com needs:

1. Scalable ingestion
2. Reliable storage
3. Fan-out

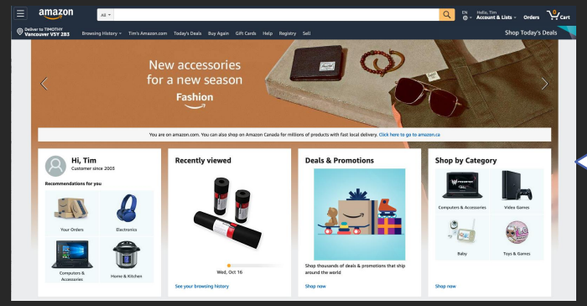
Amazon.com doesn't need:

1. Strong ordering
2. Deduplication
3. Push delivery
4. Ultralow latency

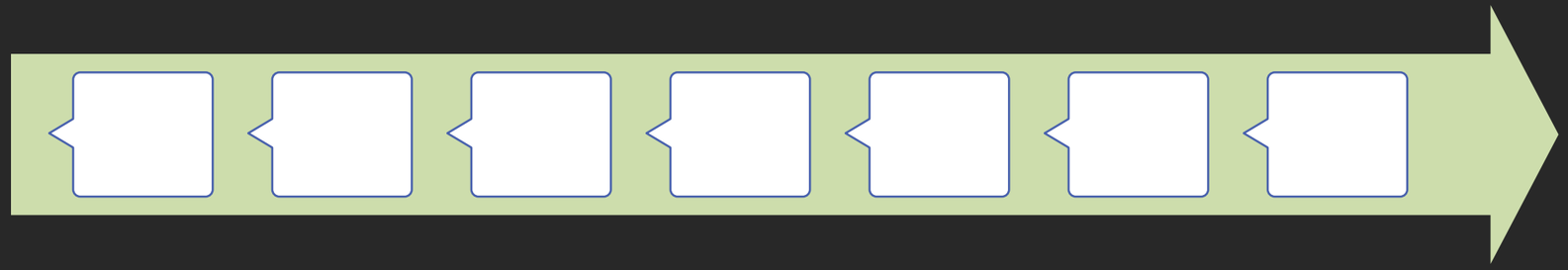
Is event-driven for you?

Is event-driven for you?

1. Are you passing around self-contained transactions?

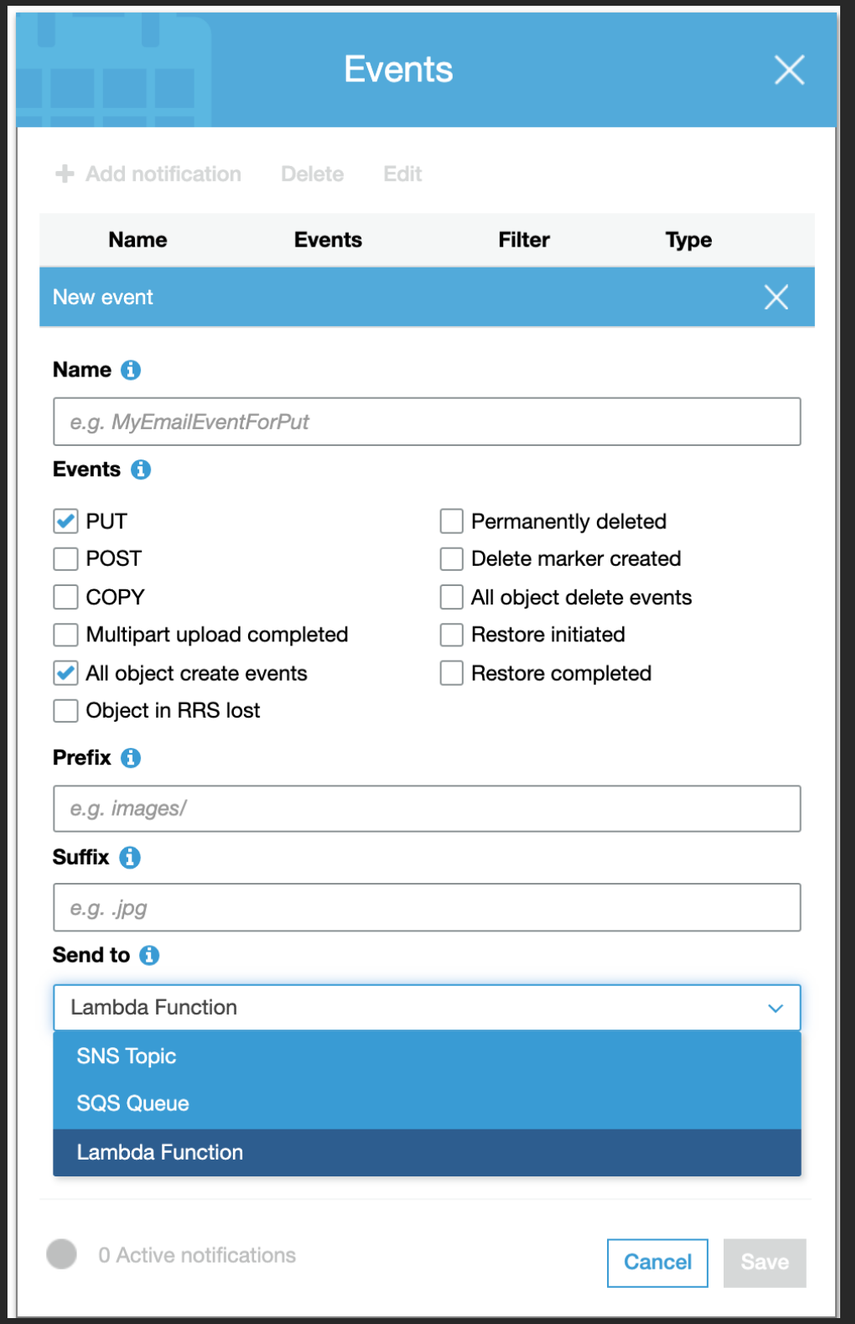


“Account 0973482 bought an Instapot ID 238479r8732 for $131.32, Visa confirmation 341513, in Prime, sending to 510 W Georgia St, Vancouver.”



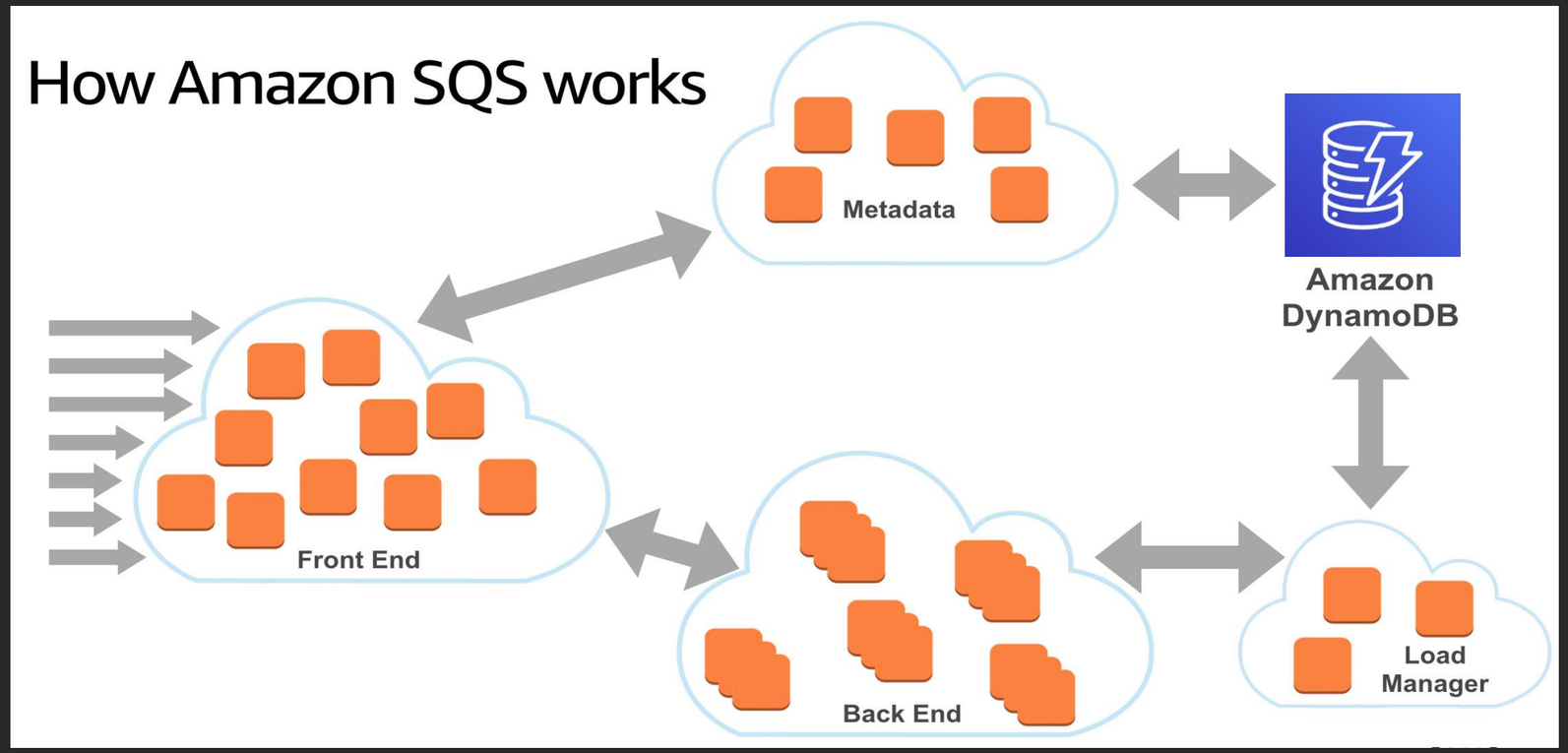
Is event-driven for you?

1. Are you passing around self-contained transactions?
2. Are useful events available for free?



Is event-driven for you?

1. Are you passing around self-contained transactions?
2. Are useful events available for free?
3. Do you need to strongly decouple your microservices?

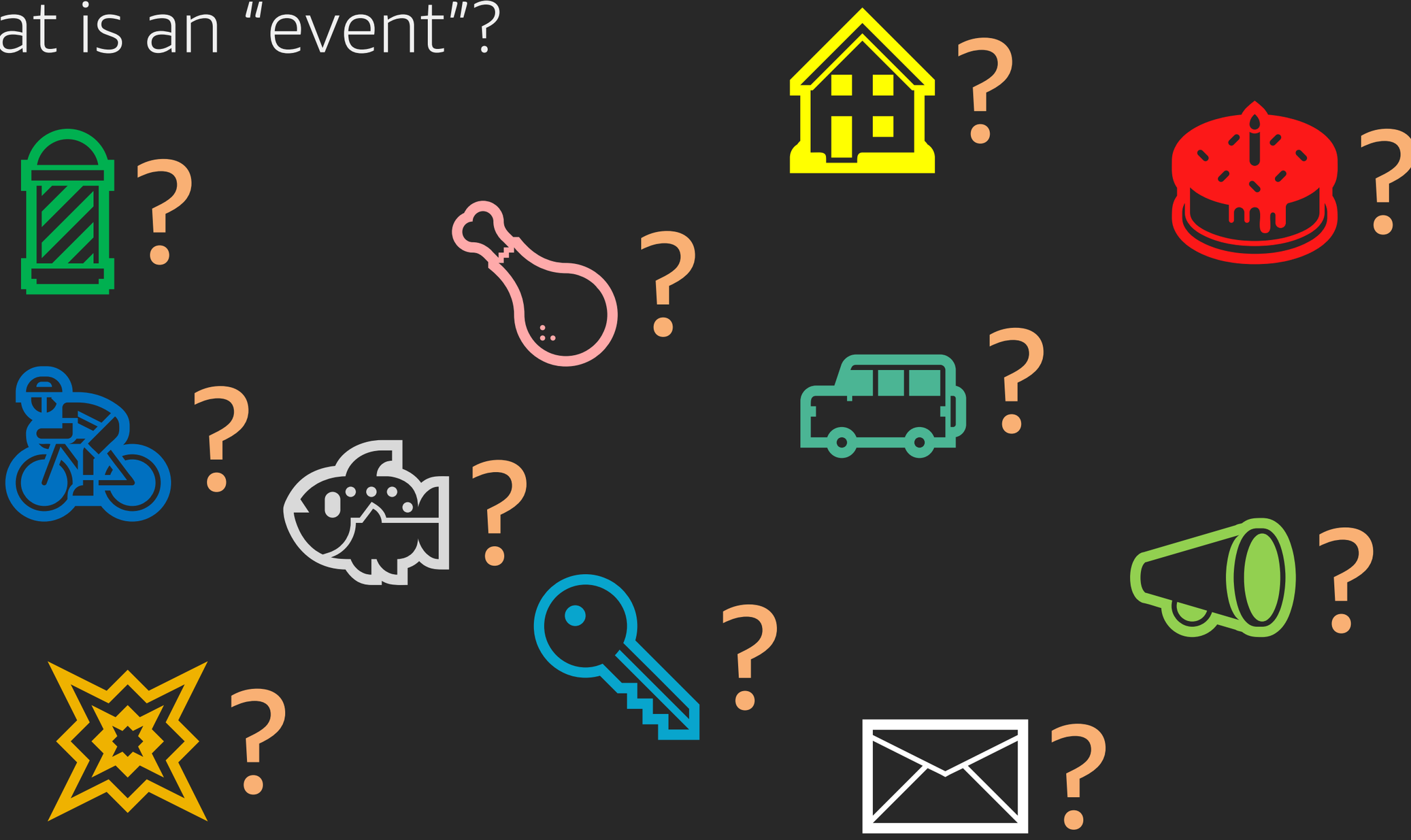


Is event-driven for you?

1. Are you passing around self-contained transactions?
2. Are useful events available for free?
3. Do you need to decouple your microservices?
4. Do you need publish-and-subscribe?

Photo: Terje Skjerdal

What is an “event”?



A small AWS event

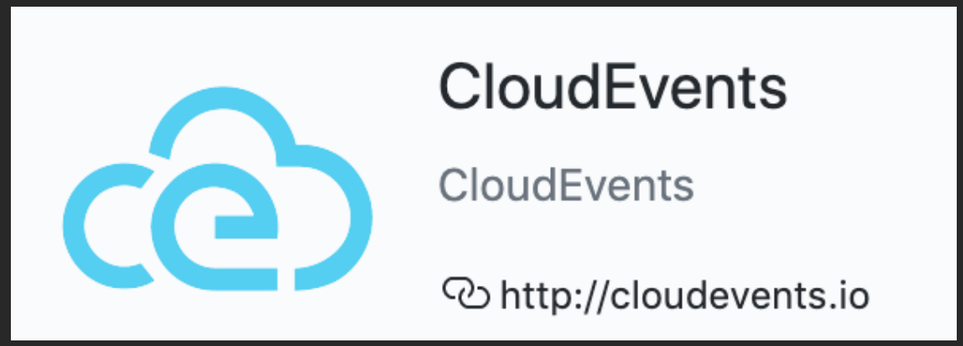

```
{
  "version": "0",
  "id": "f7a39f75-eff9-a823-5534-1075b196edd3",
  "detail-type": "EC2 Instance State-change Notification",
  "source": "aws.ec2",
  "account": "************",
  "time": "2018-08-21T20:55:26Z",
  "region": "us-east-1",
  "resources": [ ],
  "detail": {
    "instance-id": "i-00b414b880501ae45",
    "state": "running"
  }
}
```

A larger AWS event


```
{
  "version": "0",
  "id": "5af0d99b-0841-2766-e5d5-06a865895fdf",
  "detail-type": "Support Ticket: Status Changed",
  "source": "aws.partner/zendesk.com/9242270/default",
  "time": "2019-05-25T01:23:45Z",
  "region": "us-east-1",
  "resources": [ ],
  "detail": {
    "ticket_event": {
      "type": "Status Changed",
      "previous": "open",
      "current": "solved",
      "ticket": {
        "id": 35436,
        "created_at": "2019-05-20T22:55:29Z",
        "updated_at": “2019-05-25T01:23:45Z",
        . . .
```

Example CloudEvent

```
{
    "specversion" : "1.0",
    "type" : "com.example.someevent",
    "source" : "/mycontext",
    "id" : "C234-1234-1234",
    "time" : "2018-04-05T17:31:00Z",
    "comexampleextension1" : "value",
    "comexampleothervalue" : 5,
    "datacontenttype" : "application/json",
    "data" : {
        "appinfoA" : "abc",
        "appinfoB" : 123,
        "appinfoC" : true
    }
}
```



AWS event ⟶ CloudEvent

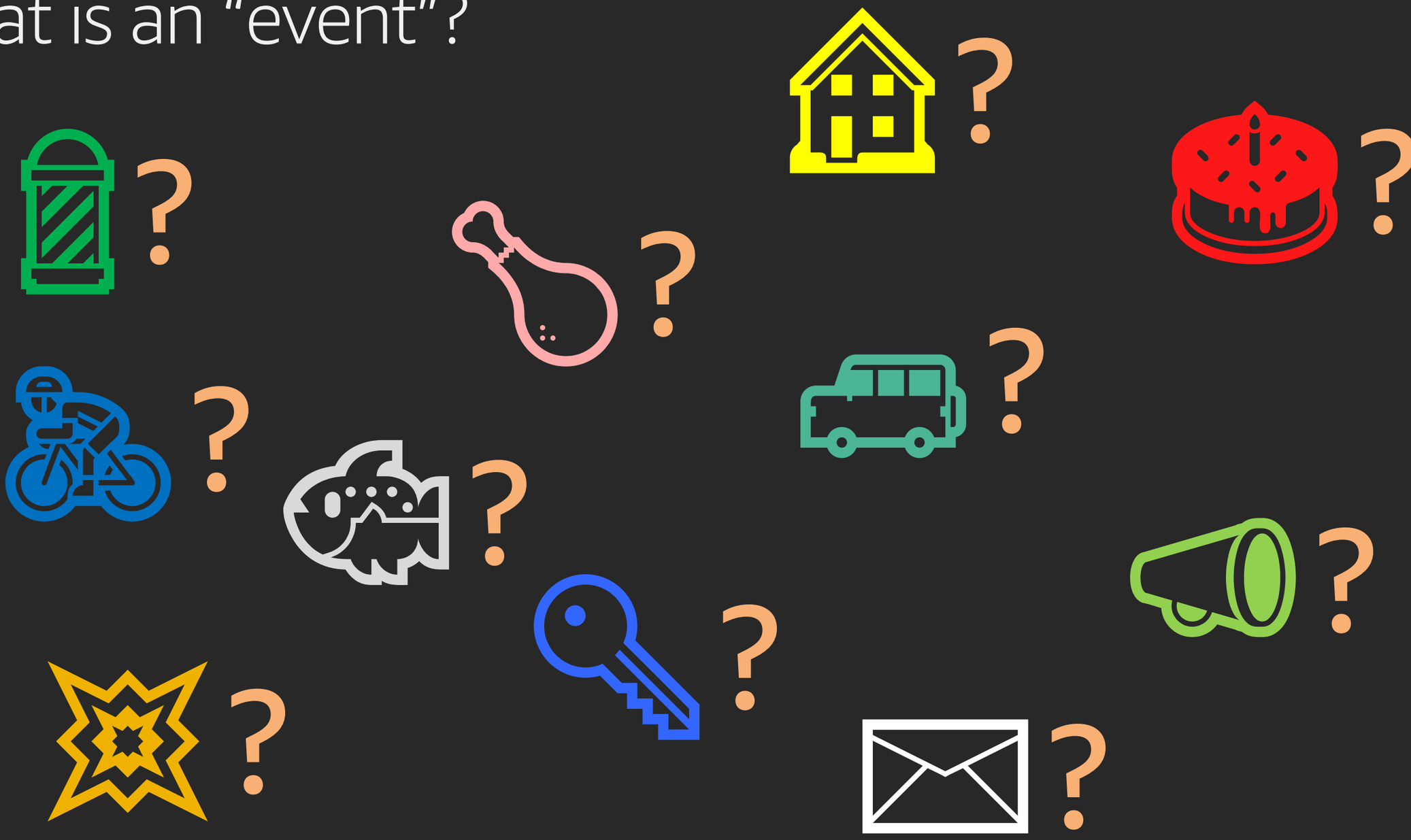
```
func toCloudEvent(ae *awsevents.AWSEvent) (*cloudevents.Event, error) {
	ce := cloudevents.NewEvent();

	ce.SetType(ae.GetDetailType())
	awsSource := ae.GetSource()
	if (strings.HasPrefix(awsSource, "aws.")) {
		awsSource = awsSource[4:]
	}
	ce.SetSource("com.amazonaws/"  + string(ae.GetRegion()) + "/" + awsSource)
	ce.SetID(ae.GetId())
	ce.SetTime(ae.GetTime());
	ce.SetExtension("awsregion", ae.GetRegion())
	ce.SetExtension("awsaccount", ae.GetAccount())
	ce.SetDataContentType("application/json")
	err := ce.SetData(ae.GetDetail())
	if err != nil {
		return nil, err
	}
	return &ce, nil
}
```

AWS event → CloudEvent


```
{
 "awsaccount": "123456789012",
 "awsregion": "us-east-2",
 "contenttype": "application/json",
 "data": {
  "instance-id": "i-00b414b880501ae45",
  "state": "running"
 },
 "id": "B977EE77-75E0-7A64-DD73-73A159BB4FF2",
 "source": "com.amazonaws/us-east-2/ec2",
 "specversion": "0.2",
 "time": "2019-10-28T19:56:14.195503Z",
 "type": "EC2 Instance State-change Notification"
}
```

What is an “event”?



Event processing facets

Event facet: Strict ordering

May vary

FIFO

Photo: Gabby Canonizado

Photo: A.Savin (WikiPhotoSpace)

Event facet: Deduplication

Eventing service

Event facet: Point-to-point vs. publish/subscribe

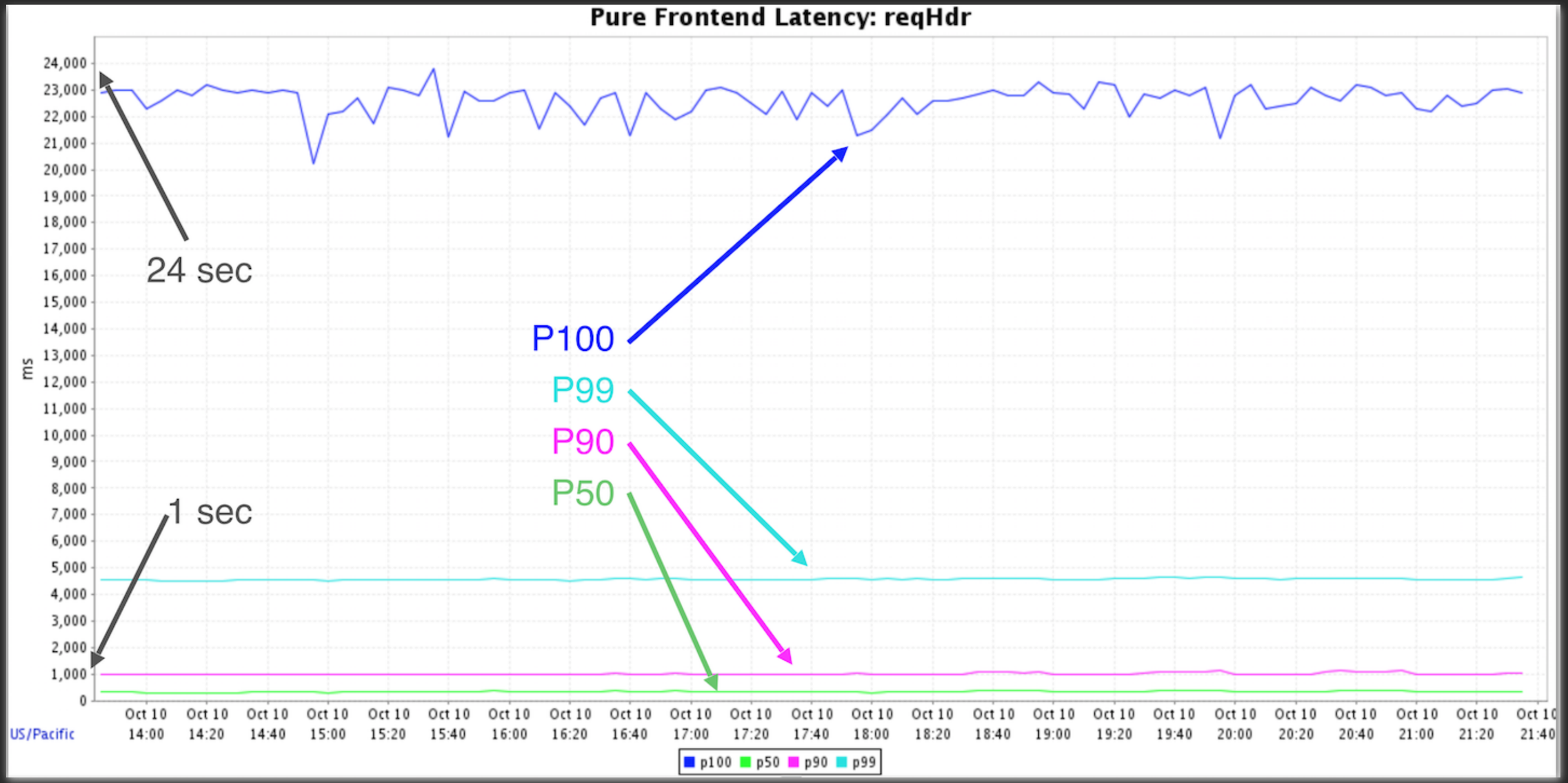
Photo: Terje Skjerdal

Event facet: Push vs. pull

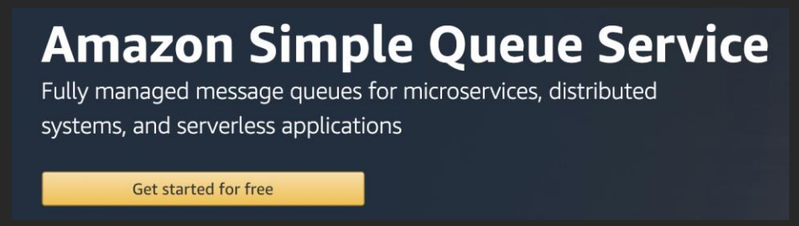
Photo: Petar Milošević (edited)

Photo: Btindall45

Event facet: Latency profile



Event facet: Serverless vs. broker/cluster

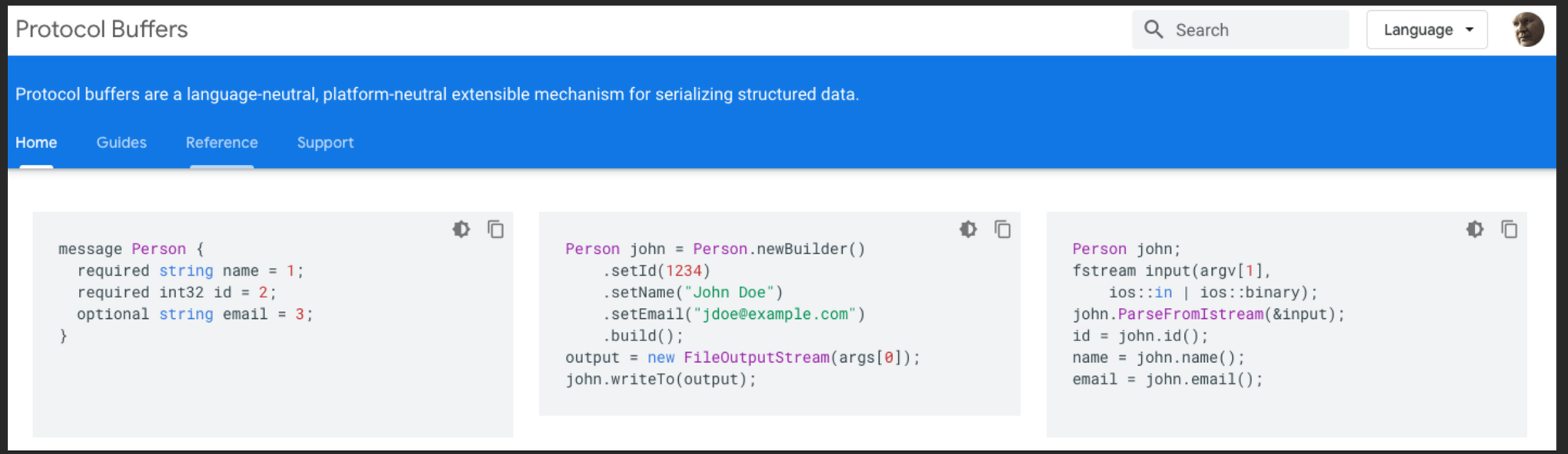


Event facet: Filtering vs. firehose

```
{
  "vendor": [ "vitamix", "instapot" ],
  "product_detail": {
    "price_usd": [
      { "numeric": ["<=", 150] }
    ]
  }
}
```

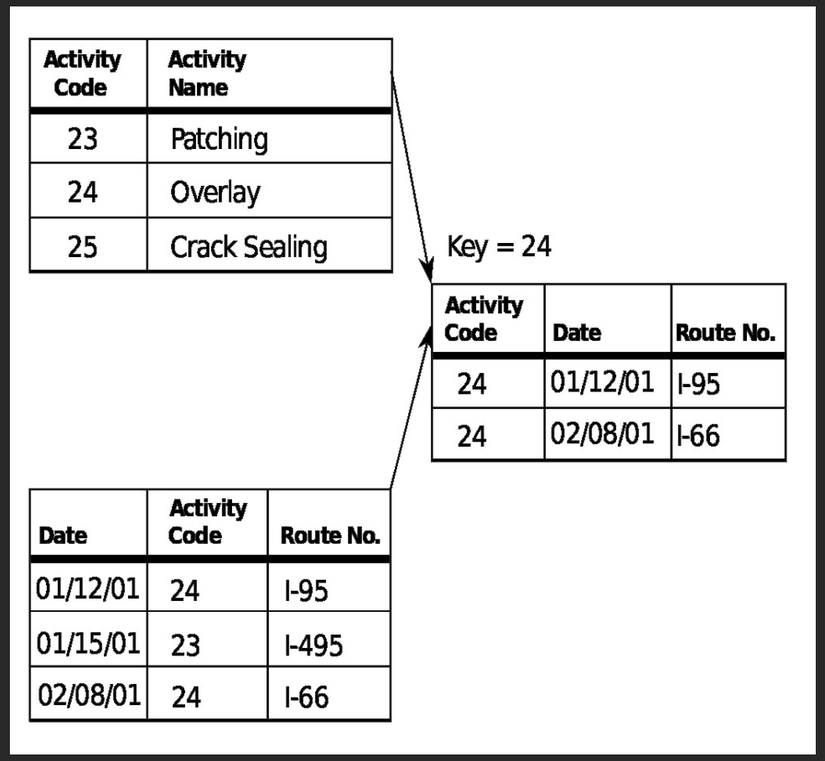
Event facet: Data blobs vs. structured objects

```
{
   "name":   "John Doe",
   "id":     1234,
   "email":  "johndoe@example.com"
}
```



(Also Avro, Thrift, Parquet, Cap'n Proto, Ion, etc.)

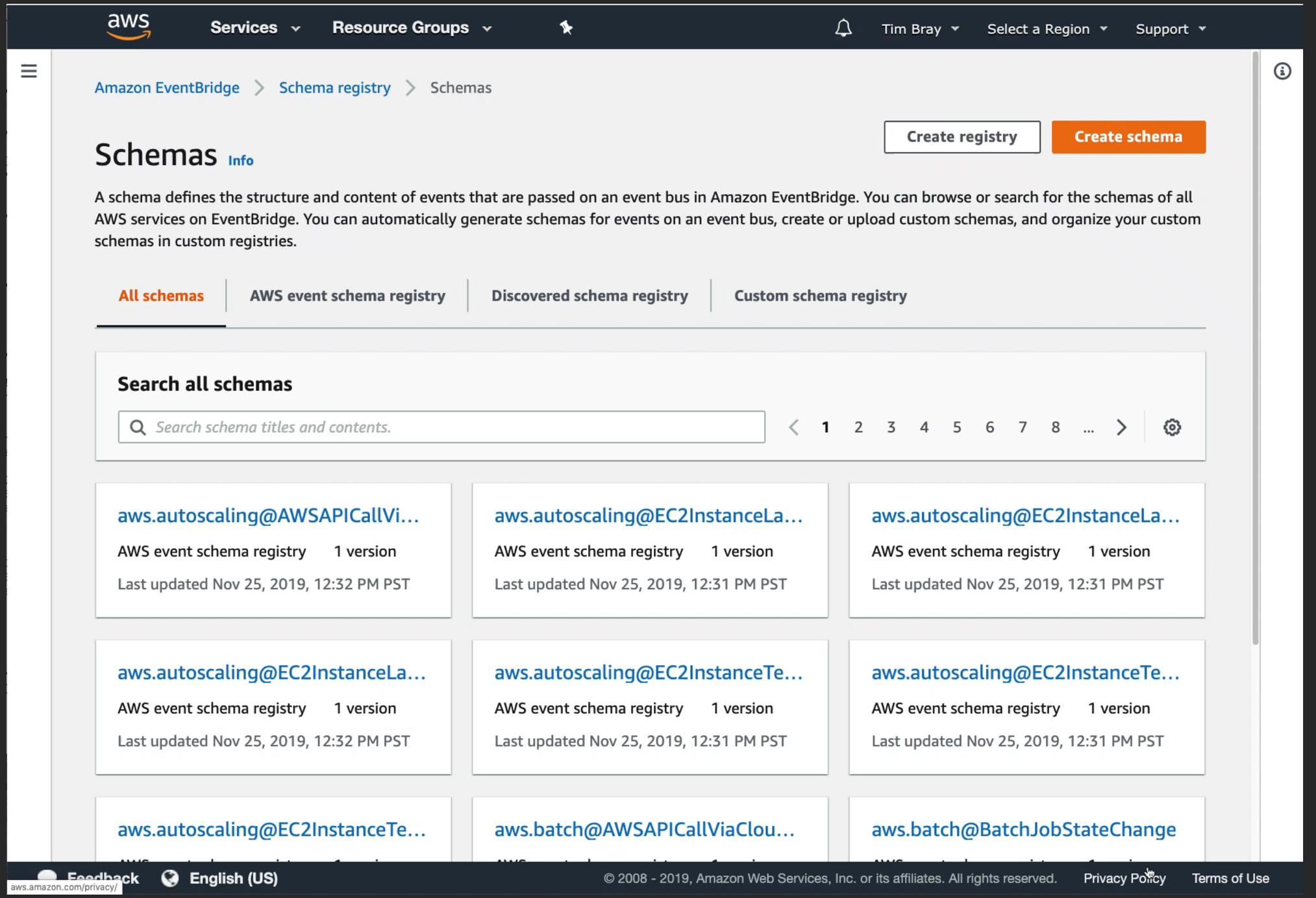
Event facet: Records vs. documents




```
{
  "version": "0",
  "id": "5af0d99b-0841-2766-e5d5-06a865895fdf",
  "detail-type": "Support Ticket: Status Changed",
  "source": "aws.partner/zendesk.com/9242270/default",
  "time": "2019-05-25T01:23:45Z",
  "region": "us-east-1",
  "resources": [ ],
  "detail": {
    "ticket_event": {
      "type": "Status Changed",
      "previous": "open",
      "current": "solved",
      "ticket": {
        "id": 35436,
        "created_at": "2019-05-20T22:55:29Z",
        "updated_at": "2019-05-25T01:23:45Z",
        . . .
```

Event facet: Uniform vs. heterogeneous events

```
boolean isPassNode(final JsonNode node) {
  if (node.isObject()) {
    final JsonNode child = node.get(Constants.TYPE_FIELD);
    if (child != null) {
      if (child.isTextual()) {
        return Constants.PASS_TYPE.equals(child.asText());
      }
    }
  }
  return false;
}
```


aws
Services
Resource Groups
Tim Bray
Select a Region
Support
Amazon EventBridge
Schema registry
Schemas
Create registry
Create schema
Schemas
Info
A schema defines the structure and content of events that are passed on an event bus in Amazon EventBridge. You can browse or search for the schemas of all AWS services on EventBridge. You can automatically generate schemas for events on an event bus, create or upload custom schemas, and organize your custom schemas in custom registries.
All schemas
AWS event schema registry
Discovered schema registry
Custom schema registry
Search all schemas
Search schema titles and contents.
1 2 3 4 5 6 7 8 ...
aws.autoscaling@AWSAPICallVi...
AWS event schema registry 1 version
Last updated Nov 25, 2019, 12:32 PM PST
aws.autoscaling@EC2InstanceLa...
AWS event schema registry 1 version
Last updated Nov 25, 2019, 12:31 PM PST
aws.autoscaling@EC2InstanceLa...
AWS event schema registry 1 version
Last updated Nov 25, 2019, 12:31 PM PST
aws.autoscaling@EC2InstanceLa...
AWS event schema registry 1 version
Last updated Nov 25, 2019, 12:32 PM PST
aws.autoscaling@EC2InstanceTe...
AWS event schema registry 1 version
Last updated Nov 25, 2019, 12:31 PM PST
aws.autoscaling@EC2InstanceTe...
AWS event schema registry 1 version
Last updated Nov 25, 2019, 12:31 PM PST
aws.autoscaling@EC2InstanceTe...
aws.batch@AWSAPICallViaClou...
aws.batch@BatchJobStateChange
Feedback
English (US)
© 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
aws.amazon.com/privacy/

Event facets

	ActiveMQ	Artemis	Event Bridge	Kafka	Amazon Kinesis	RabbitMQ	Amazon SNS	Amazon SQS	Amazon SQS FIFO
Ordering	Yes	Yes	No	Yes	Yes	Yes	No	No	Yes
Dedupe	*	Yes	No	Yes	No	No	No	No	Yes
P2p vs. Pub/sub	P/S	P/S	P/S	P/S	P/S	P/S	P/S	P2p	P2p
Push vs. Pull	Both	Both	Push	Pull	Pull	Both	Push	Pull	Pull
Serverless	No	No	Yes	No	Yes	No	Yes	Yes	Yes
Filtering	Yes	Yes	Yes	*	No	No	Yes	No	No

*: “It’s complicated”

Events vs. . . .

APIs

Events vs. . . .

Service mesh

Eventing things that still need work:

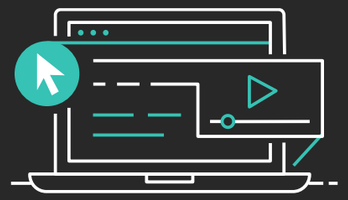
1. Loops!
2. Too much pipefitting.
3. Easier and safer access control.
4. Blobby events, discovery, and autocomplete needed.

Eventing things that still need work:

1. Loops!
2. Too much pipefitting.
3. Easier and safer access control.
4. ~~Blobby events, discovery, and autocomplete needed~~.

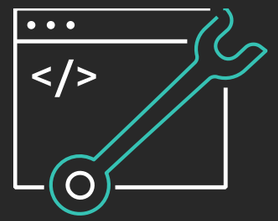
Learn serverless with AWS Training and Certification

Resources created by the experts at AWS to help you learn modern application development



Free, on-demand courses on serverless, including

- Introduction to Serverless Development
- Getting into the Serverless Mindset
- AWS Lambda Foundations
- Amazon API Gateway for Serverless Applications
- Amazon DynamoDB for Serverless Architectures



Additional digital and classroom trainings cover modern application development and computing

Visit the Learning Library at https://aws.training

aws training and certification

Thank you!

Tim Bray

timbray@amazon.com
@timbray
https://tbray.org

AWS re:Invent

aws

Please complete the session survey in the mobile app.