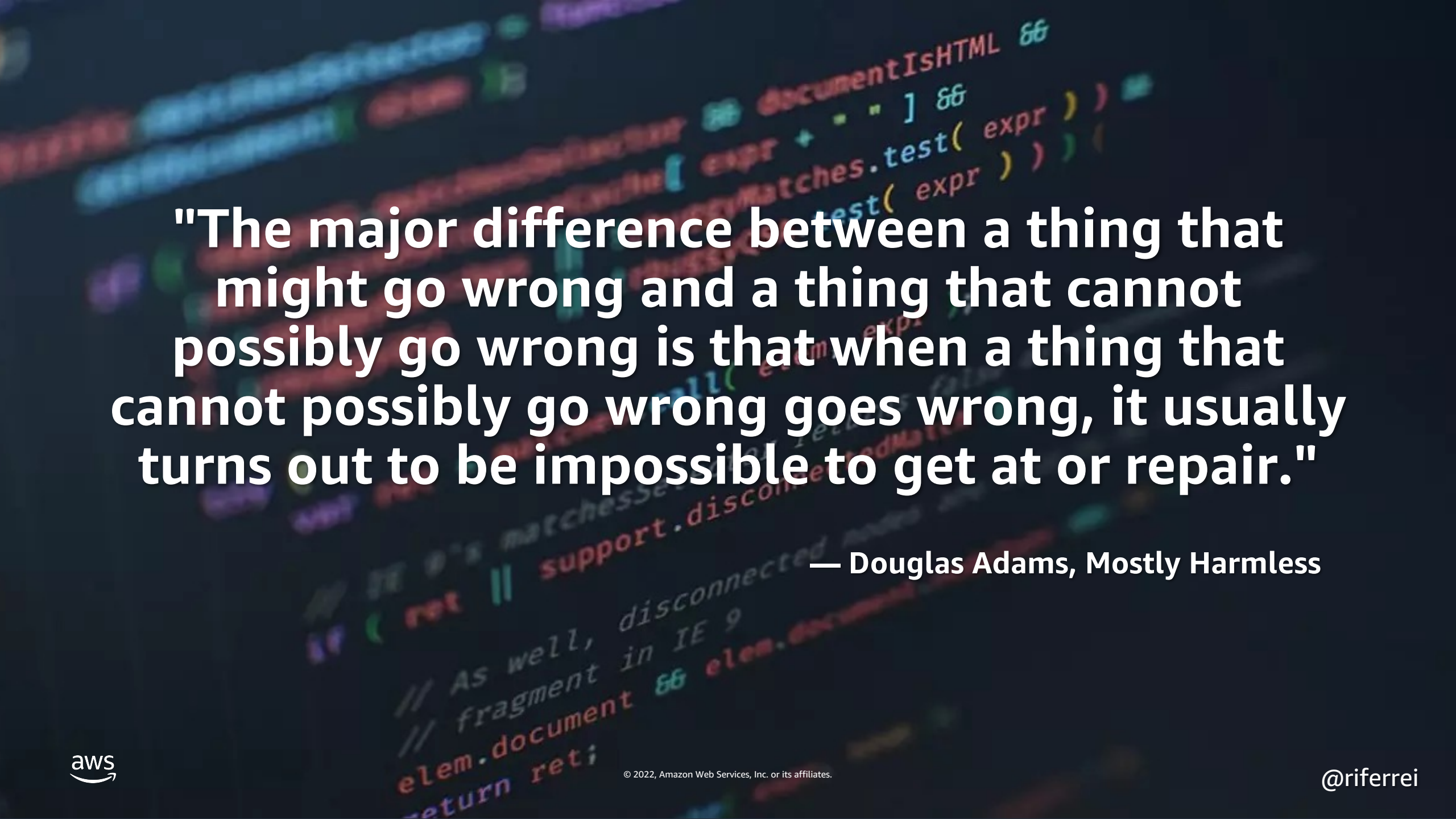
When Kafka is the Source of Truth;

Schemas Become your Source of Headaches

Ricardo Ferreira

Senior Developer Advocate
Amazon Web Services

@riferrei



👋 Hi, I'm Ricardo Ferreira

- Developer Advocate at AWS.
- Fanatic Marvel fan. My favorite characters are Daredevil, Venon, Deadpool, and the Punisher.
- Yes, it's my birthday today. 😅

Back to basics: data serialization and deserialization

@riferrei

Let's do some 👨🏻‍💻 coding

- Scan the bar code in the right for the GitHub repository.
- Each example has a unique number and a use case name.
- In the folder that starts with "00-" you can find a Docker Compose file to spin up a Kafka deployment for testing.

@riferrei

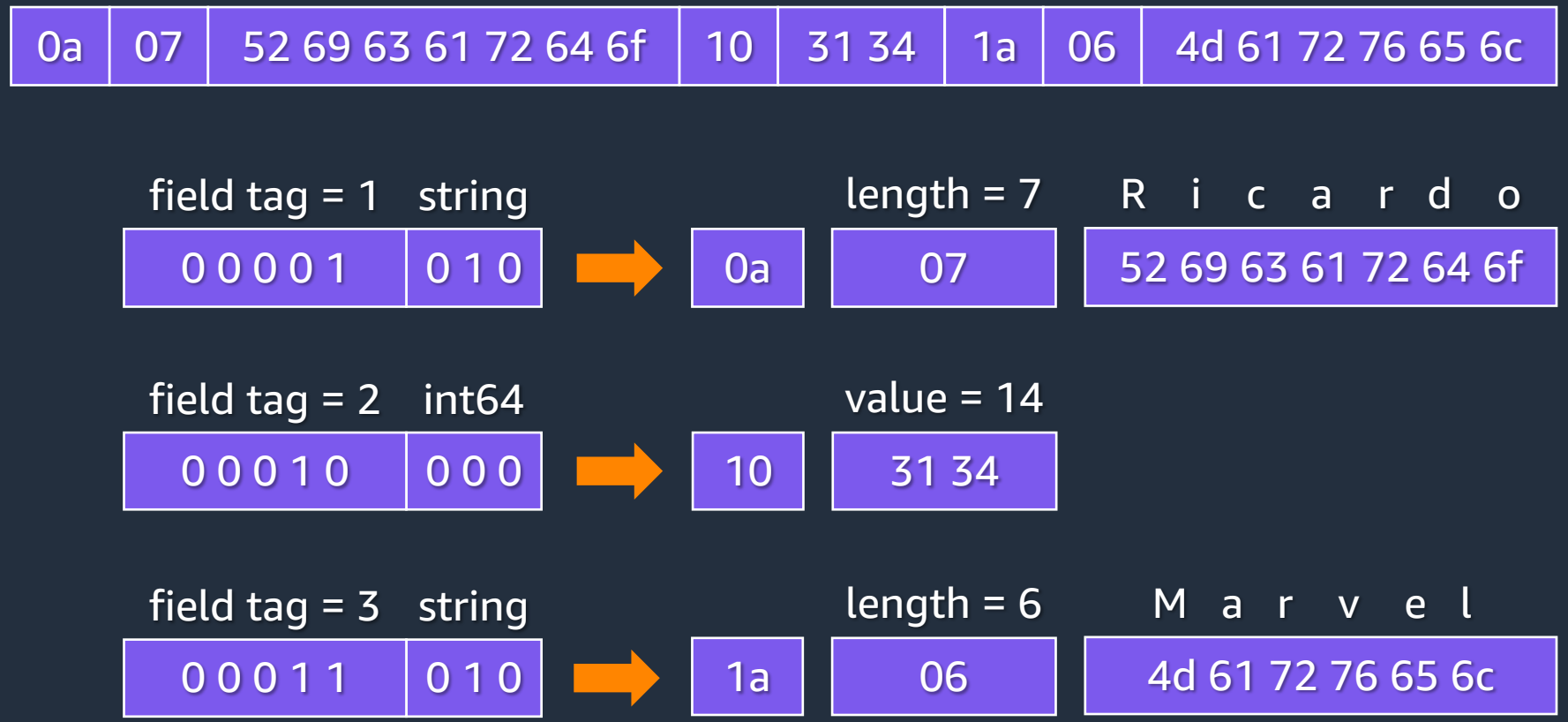
Binary Encoding with ProtoBuf, Thrift, and Avro

@riferrei

Schema using Protocol Buffers

```
message Person {
    string userName = 1;
    optional int64 favoriteNumber = 2;
    repeated string interests = 3;
}
```

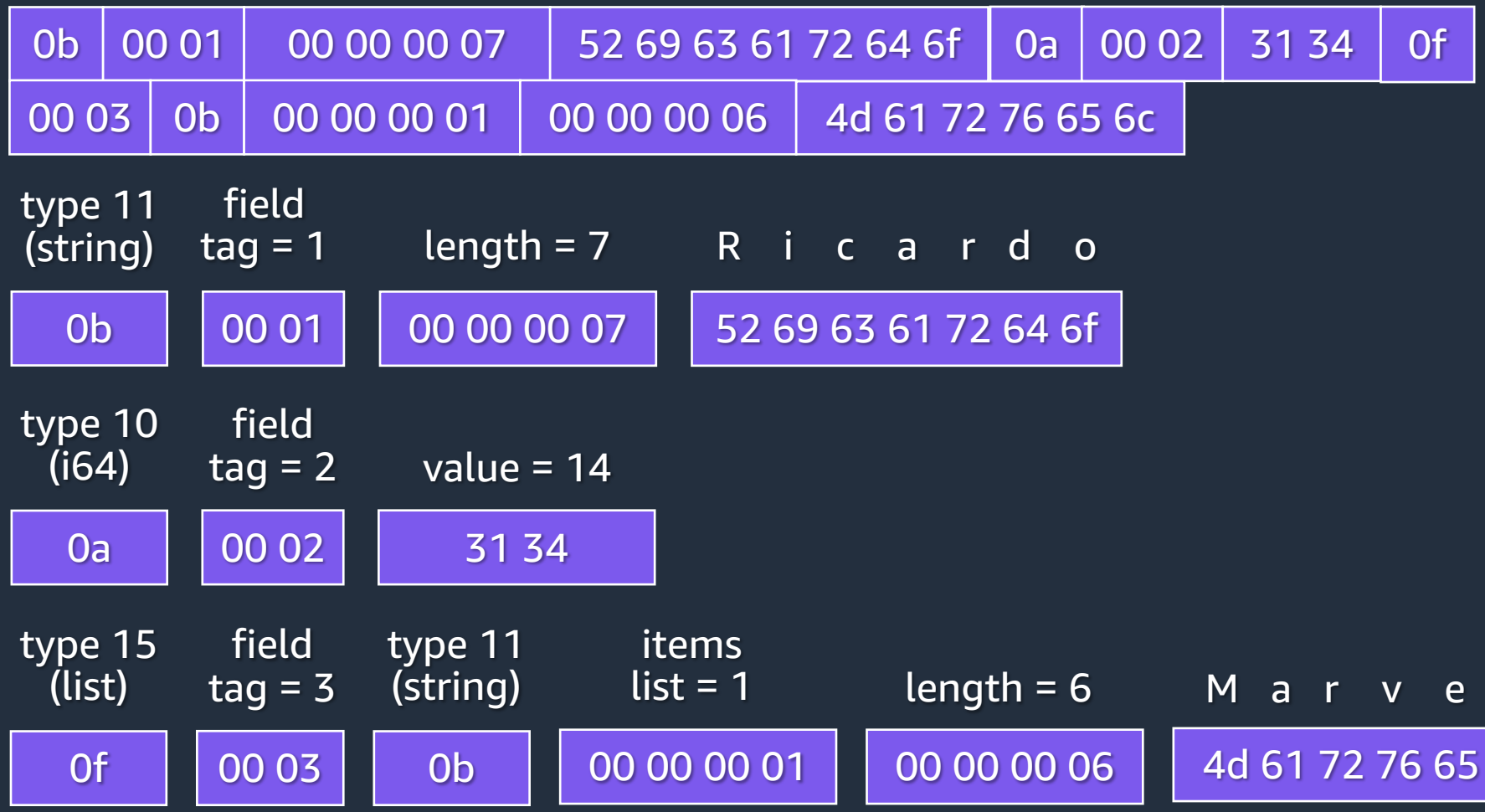
Schema using Protocol Buffers



Schema using Thrift

```
struct Person {
    1: required string userName,
    2: optional i64 favoriteNumber,
    3: optional list<string> interests
}
```

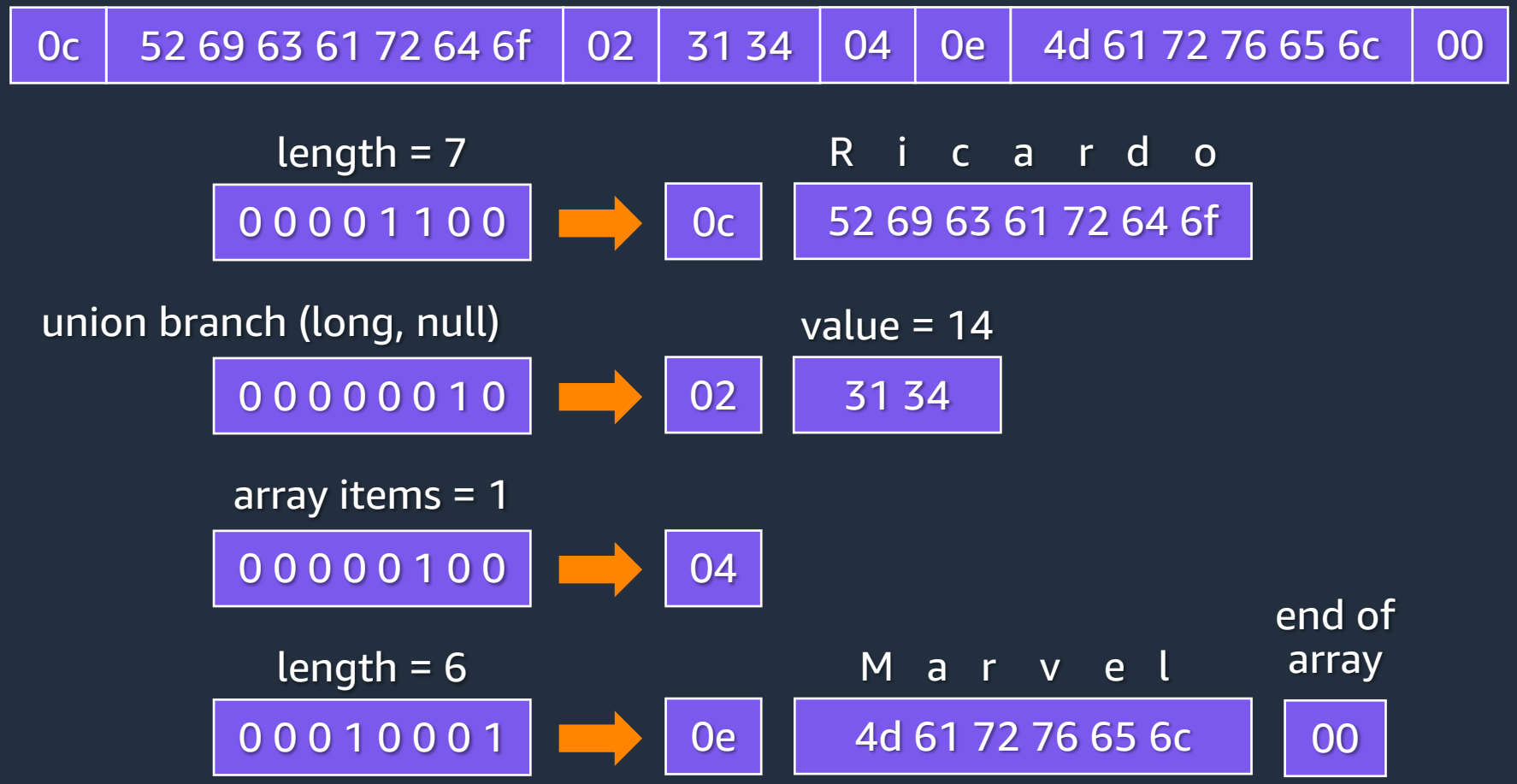
Schema using Thrift (BinaryProtocol)



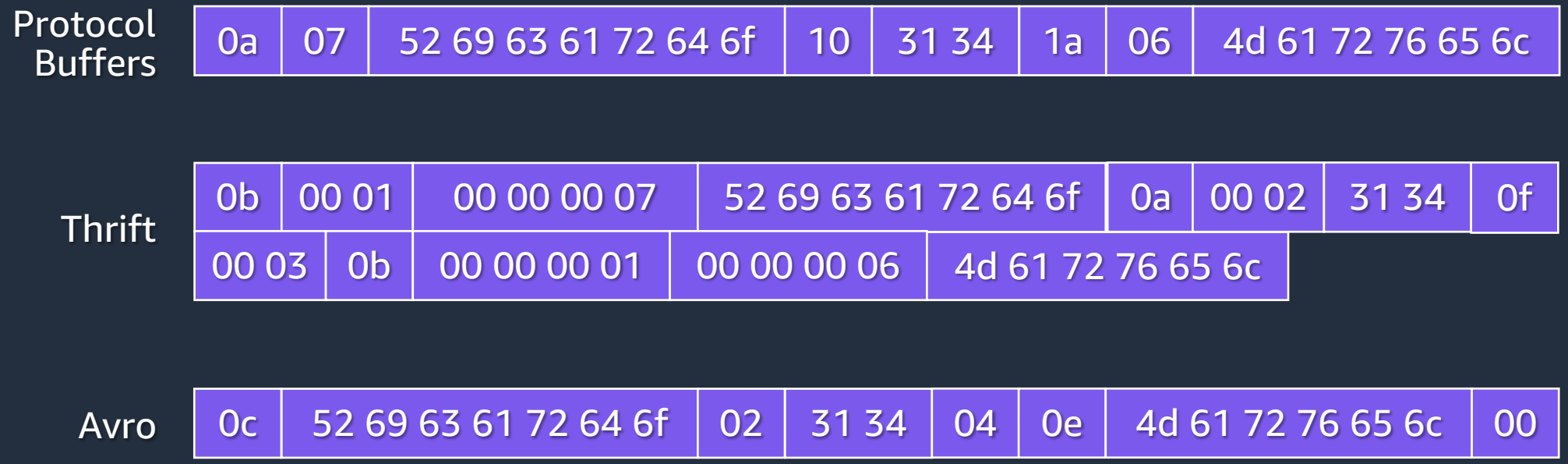
Schema using Avro

```
{
   "type": "record",
   "name": "Person",
   "fields": [
      {"name": "userName", "type": "string"},
      {"name": "favoriteNumber", "type": ["null", "long"], "default": null},
      {"name": "interests", "type": {"type": "array", "items": "string"}}
   ]
}
```

Schema using Avro

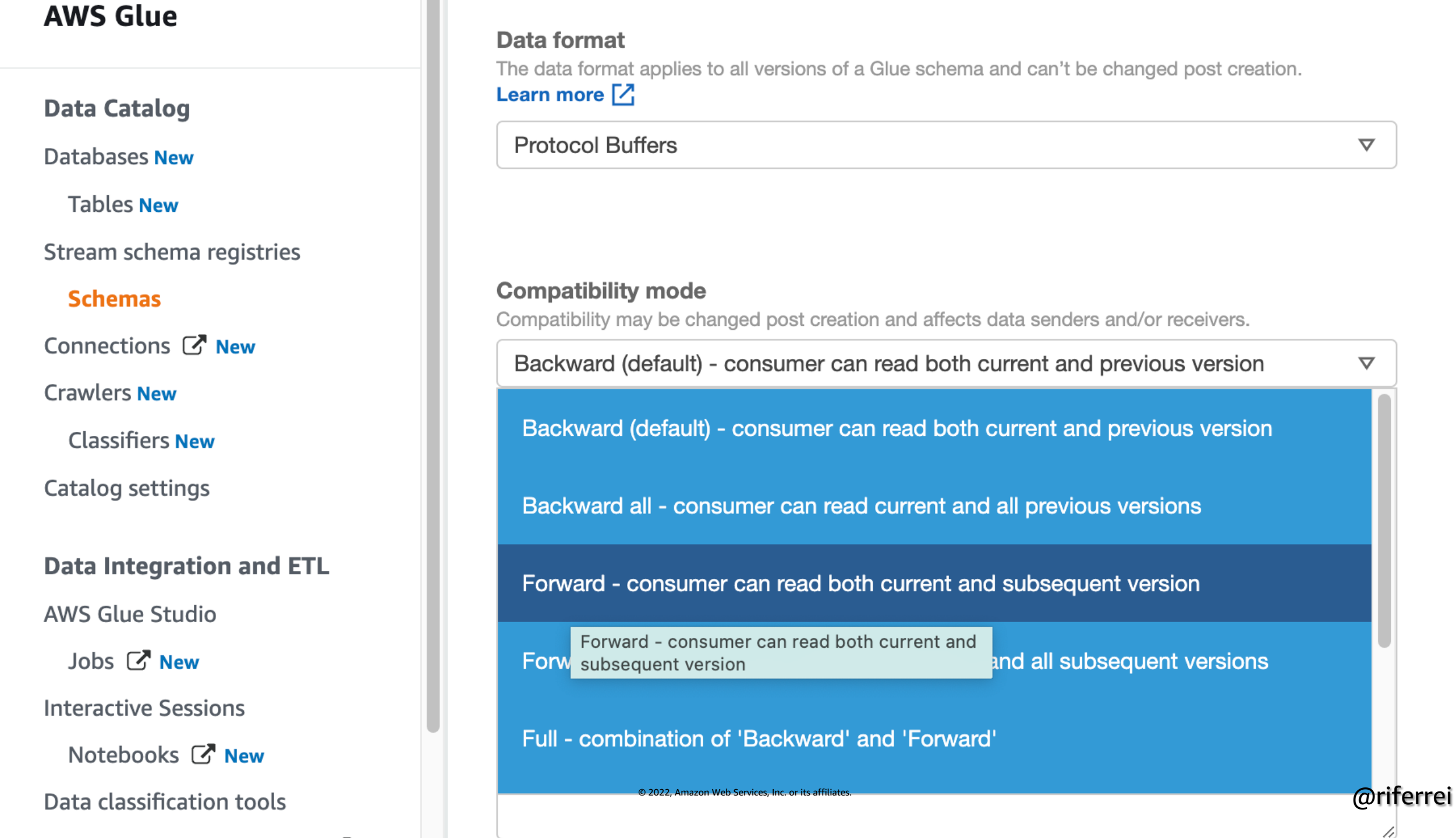


Same record; different binary encoding



Schema compatibility

- **Backward compatibility**: Newer code can read data from older code
- **Forward compatibility**: Older code can read data from newer code


AWS Glue
Data Catalog
Databases New
Tables New
Stream schema registries
Schemas
Connections New
Crawlers New
Classifiers New
Catalog settings
Data Integration and ETL
AWS Glue Studio
Jobs New
Interactive Sessions
Notebooks New
Data classification tools
Data format
The data format applies to all versions of a Glue schema and can't be changed post creation.
Learn more
Protocol Buffers
Compatibility mode
Compatibility may be changed post creation and affects data senders and/or receivers.
Backward (default) - consumer can read both current and previous version
Backward (default) - consumer can read both current and previous version
Backward all - consumer can read current and all previous versions
Forward - consumer can read both current and subsequent version
Forward - consumer can read both current and subsequent version
Full - combination of 'Backward' and 'Forward'
© 2022, Amazon Web Services, Inc. or its affiliates.
@riferrei

Required/Optional fields

- Not written into the binary format.
- Used to provide runtime checks.
- Useful to catching bugs in the code that writes data into the streams.

Schema evolution

- Change field names, not the tags.
- New fields must use new tags.
- Old just code ignore unknown tags.
- Datatype annotations help the parser to know how many bytes to skip when old code is reading new code.

Changing data types

- Possible, but it may lose precision or get data being truncated. Read the documentation.
- Change from single value to list depends on the binary format.
- Protocol Buffers is just repeated in the format. It can be skipped. But Thrift requires data to be there.

@riferrei

Can I switch the Schema Registry to another registry?

Time for 👨🏻‍💻 coding again

- Scan the bar code in the right for the GitHub repository.
- Each example has a unique number and a use case name.
- In the folder that starts with "00-" you can find a Docker Compose file to spin up a Kafka deployment for testing.

@riferrei

Summary

1. Each programming language handles serialization differently.
2. Schema Registry is a database for schemas. It's not Doctor Strange.
3. A record payload is not just the payload. Mind the Schema IDs.
4. Backward/forward compatibility is different in each binary format.
5. Migrating from one registry to another is possible. But not easy.

Chapter 4: Encoding and Evolution

- Many thanks to Martin Kleppmann.
- This chapter teaches all about code that can evolve using schemas.
- This session is just a small portion of the chapter. I recommend you read the chapter in its entirety.

@riferrei

Thank you!

Ricardo Ferreira
@riferrei