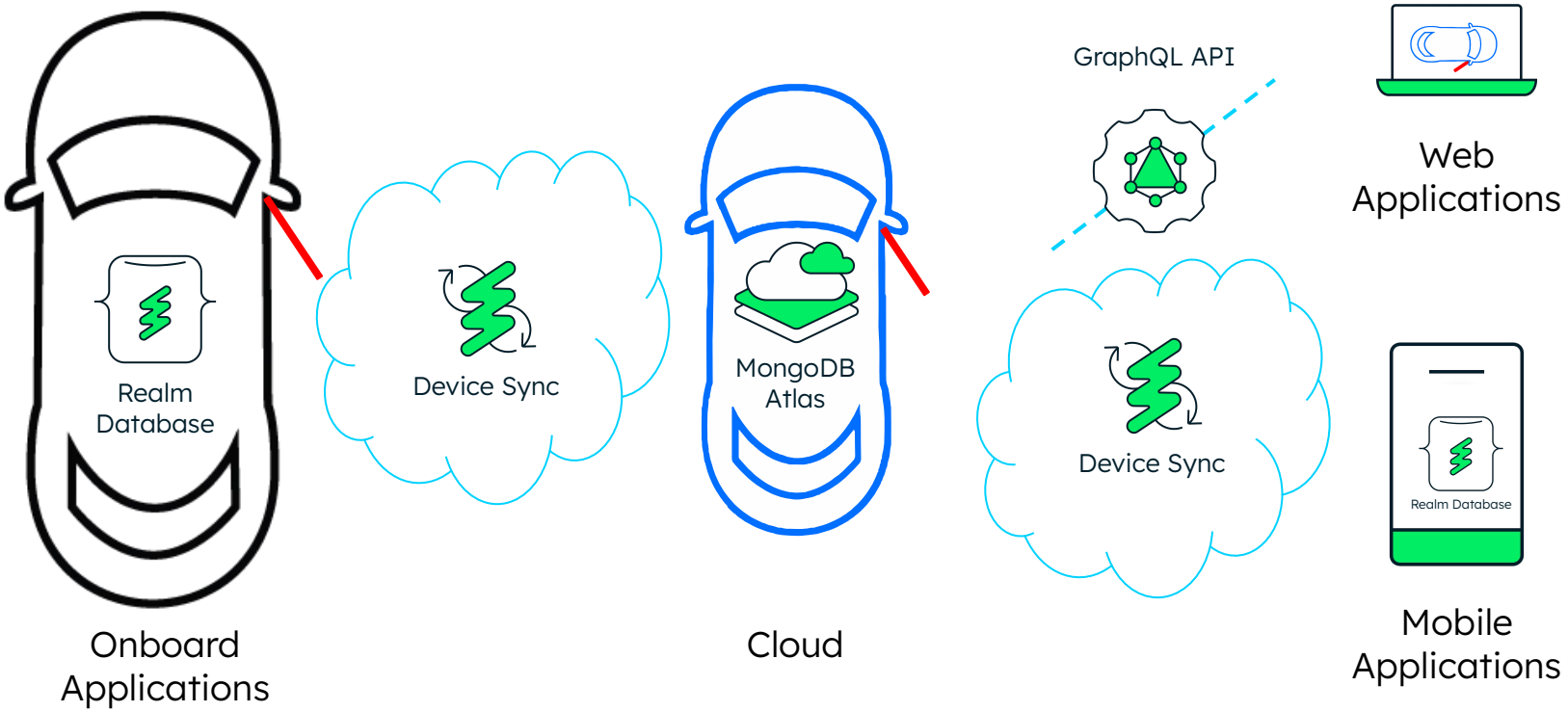
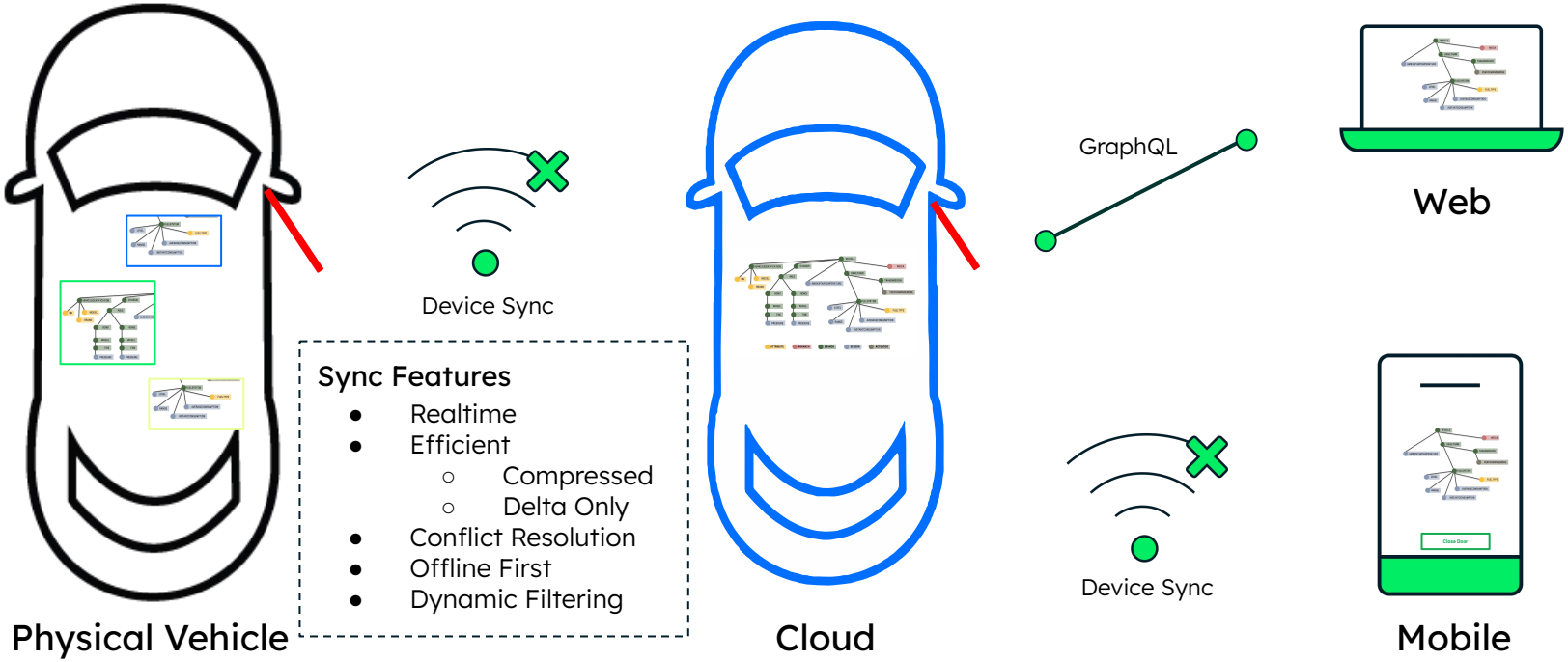
Distributed VSS Data Sets

Bring VSS to life

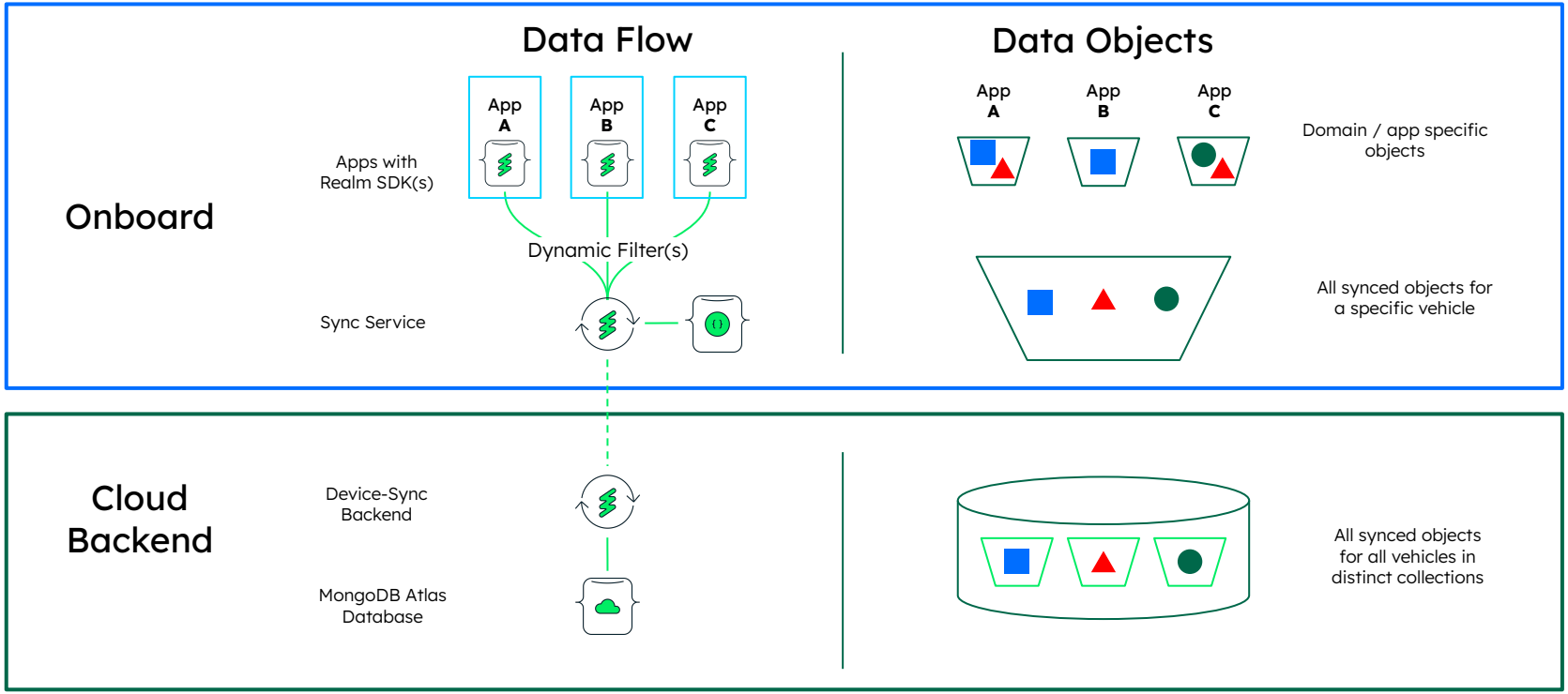
The Concept


Realm
Database
Onboard
Applications
Device Sync
MongoDB
Atlas
Cloud
GraphQL API
Device Sync
Web
Applications
Realm Database
Mobile
Applications

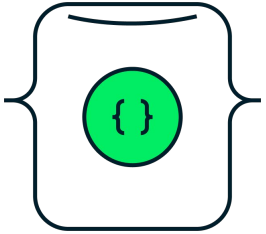
Keep the Data in Sync



Future Outlook / Roadmap



MongoDB®



Document Model or Object Oriented Database

- Flexible
- Scalable
- Always On
- Freedom to run anywhere
- …

Documents Are Objects

Related data contained in a single, rich document

```
{
  "_id" : ObjectId("5ad88534e3632e1a35a58d00"),
  "name" : {
    "first" : "John",
    "last" : "Doe" },
  "address" : [
    { "location" : "work",
      "address" : {
        "street" : "16 Hatfields",
        "city" : "London",
        "postal_code" : "SE1 8DJ"},
      "geo" : { "type" : "Point", "coord" : [
         -0.109081, 51.5065752]}},
+   {...}
  ],
  "dob" : ISODate("1977-04-01T05:00:00Z"),
  "retirement_fund" : NumberDecimal("1292815.75")
}
```

“Realm” - Embedded OSS Database

Offline first paradigm

- Usage: **100k+** developers; **65%** of Fortune 1000; **2B+** app installs
- **47k+** Github stars
- **Apache 2.0** license
- Active community involvement

Easy for developers

- Designed and built for **resource constrained environments**
- Just objects, with **native code paradigms**
- **Live objects** update automatically
- The class definitions ***are*** the database schema

2010

Development started by two former Nokia engineers

2016

Official announcement of Realm mobile platform

realm

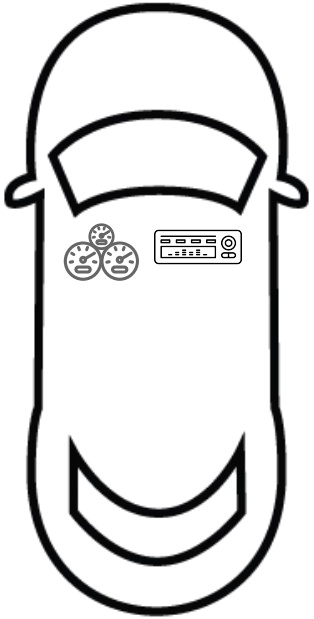
>2019

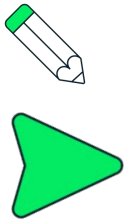
Acquisition by MongoDB

>2022

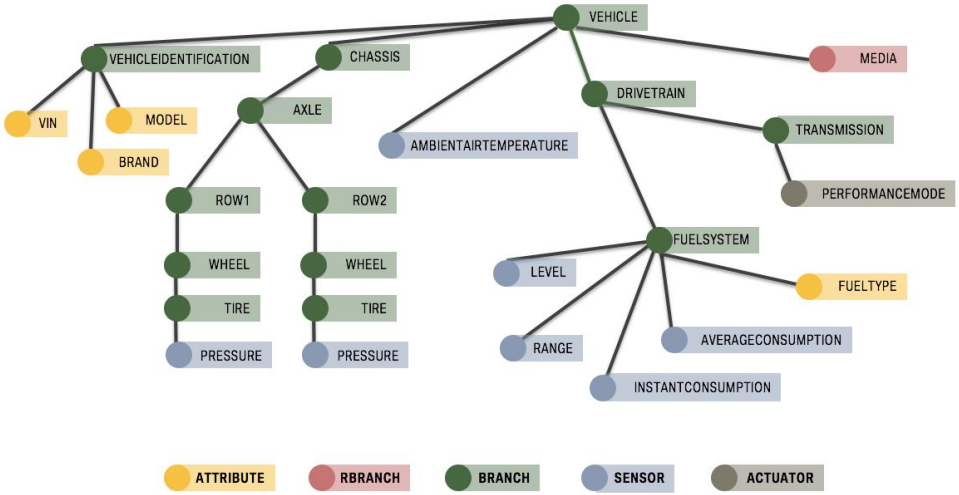
Device Sync integration into Atlas

Vehicle Signal Specification



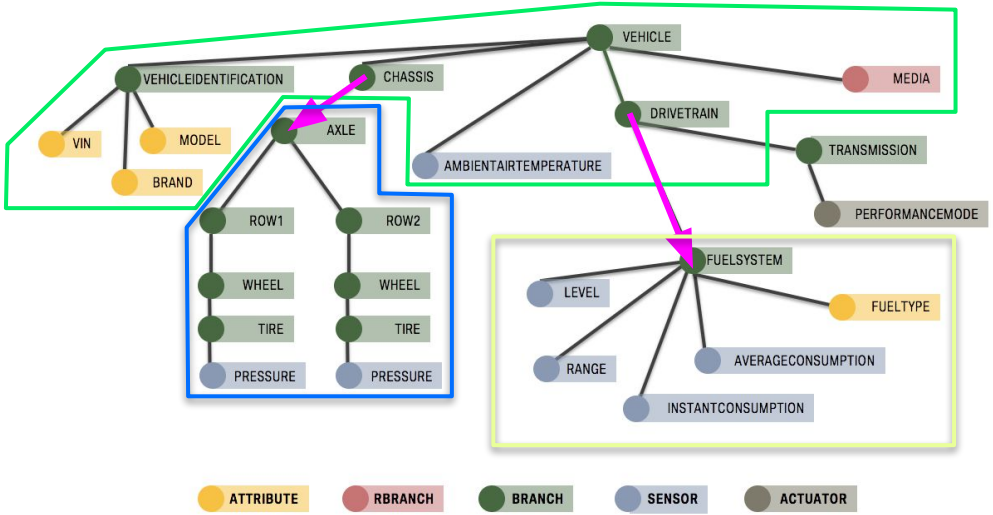


VSS Tree



Official VSS Documentation:
https://covesa.github.io/vehicle_signal_specification/

Tree of Objects



Device (Vehicle) Class

Component A Class

Component B Class

Relationships / References

Objects are Instances of Classes

```
export class Device {

   public _id = new ObjectId;
   public name = "";
   public owner_id = "";
   public isOn = false;
   public flexibleData?: Realm.Dictionary<string>;
   public components: Array<Component> = [];

   public static schema = {
       name: 'Device',
       primaryKey: '_id',
       properties: {
           _id: 'objectId',
           name: 'string',
           owner_id: 'string',
           isOn: 'bool',
           components: 'Component[]',
           flexibleData: 'string{}'
       }
   }
}
```

```
export class Component {

   public _id = new ObjectId;
   public name = "";
   public owner_id = "";

   public static schema = {
       name: 'Component',
       primaryKey: '_id',
       properties: {
           _id: 'objectId',
           name: 'string?',
           owner_id: 'string'
       }
   }
}
```

Backend JSON Schema

Device

```
{
  "title": "Device",
  "bsonType": "object",
  "required": [
    "_id",
    "name",
    "owner_id",
    "isOn",
    "sensor"
  ],
  "properties": {
    "_id": {
      "bsonType": "objectId"
    },
    "name": {
      "bsonType": "string"
    },
    "owner_id": {
      "bsonType": "string"
    },
    "components": {
      "bsonType": "array",
      "items": {
        "bsonType": "objectId"
      }
    },
    "isOn": {
      "bsonType": "bool"
    },
    "flexibleData": {
      "bsonType": "object",
      "additionalProperties": {
        "bsonType": "mixed"
      }
    },
    "mixedTypes": {
      "bsonType": "mixed"
    },
    "sensor": {
      "bsonType": "long"
    }
  }
}
```

Relationship

```
{
  "components": {
    "ref": "#/relationship/mongodb-atlas/Digital-Twins/Component",
    "foreignKey": "_id",
    "isList": true
  }
}
```

Component

```
{
  "title": "Component",
  "bsonType": "object",
  "required": [
    "_id",
    "owner_id"
  ],
  "properties": {
    "_id": {
      "bsonType": "objectId"
    },
    "name": {
      "bsonType": "string"
    },
    "owner_id": {
      "bsonType": "string"
    }
  }
}
```

GraphQL Schema

Device

```
{
  "title": "Device",
  "bsonType": "object",
  "required": [
    "_id",
    "name",
    "owner_id",
    "isOn",
    "sensor"
  ],
  "properties": {
    "_id": {
      "bsonType": "objectId"
    },
    "name": {
      "bsonType": "string"
    },
    "owner_id": {
      "bsonType": "string"
    },
    "components": {
      "bsonType": "array",
      "items": {
        "bsonType": "objectId"
      }
    },
    "isOn": {
      "bsonType": "bool"
    },
    "flexibleData": {
      "bsonType": "object",
      "additionalProperties": {
        "bsonType": "mixed"
      }
    },
    "mixedTypes": {
      "bsonType": "mixed"
    },
    "sensor": {
      "bsonType": "long"
    }
  }
}
```

Relationship

```
{
  "components": {
    "ref": "#/relationship/mongodb-atlas/Digital-Twins/Component",
    "foreignKey": "_id",
    "isList": true
  }
}
```

Component

```
{
  "title": "Component",
  "bsonType": "object",
  "required": [
    "_id",
    "owner_id"
  ],
  "properties": {
    "_id": {
      "bsonType": "objectId"
    },
    "name": {
      "bsonType": "string"
    },
    "owner_id": {
      "bsonType": "string"
    }
  }
}
```

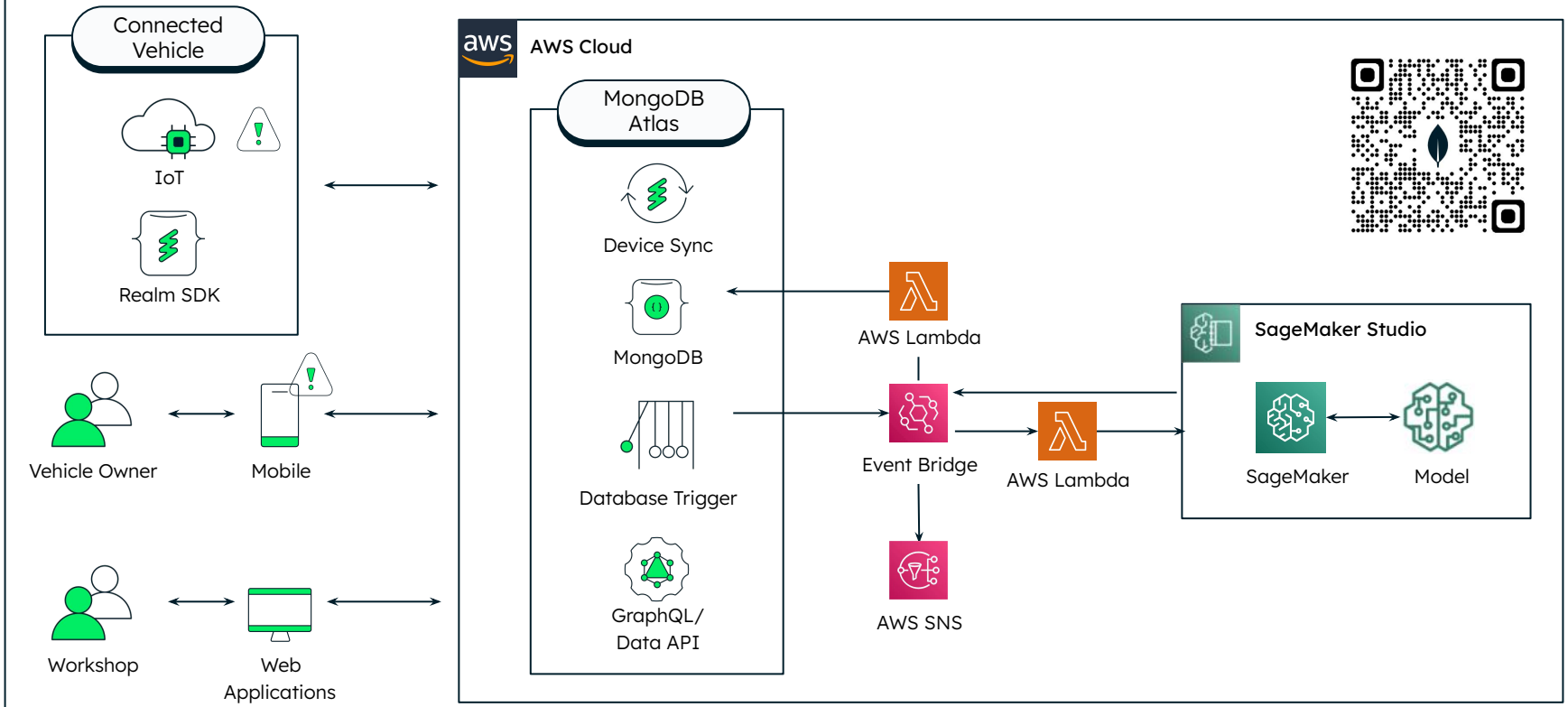
JSON schema is automatically converted into GraphQL schema

GraphiQL | Prettify | Merge | Copy | History

```
query {
  device {
    _id
    name
    owner_id
    isOn
    components {
      name
    }
  }
}
```

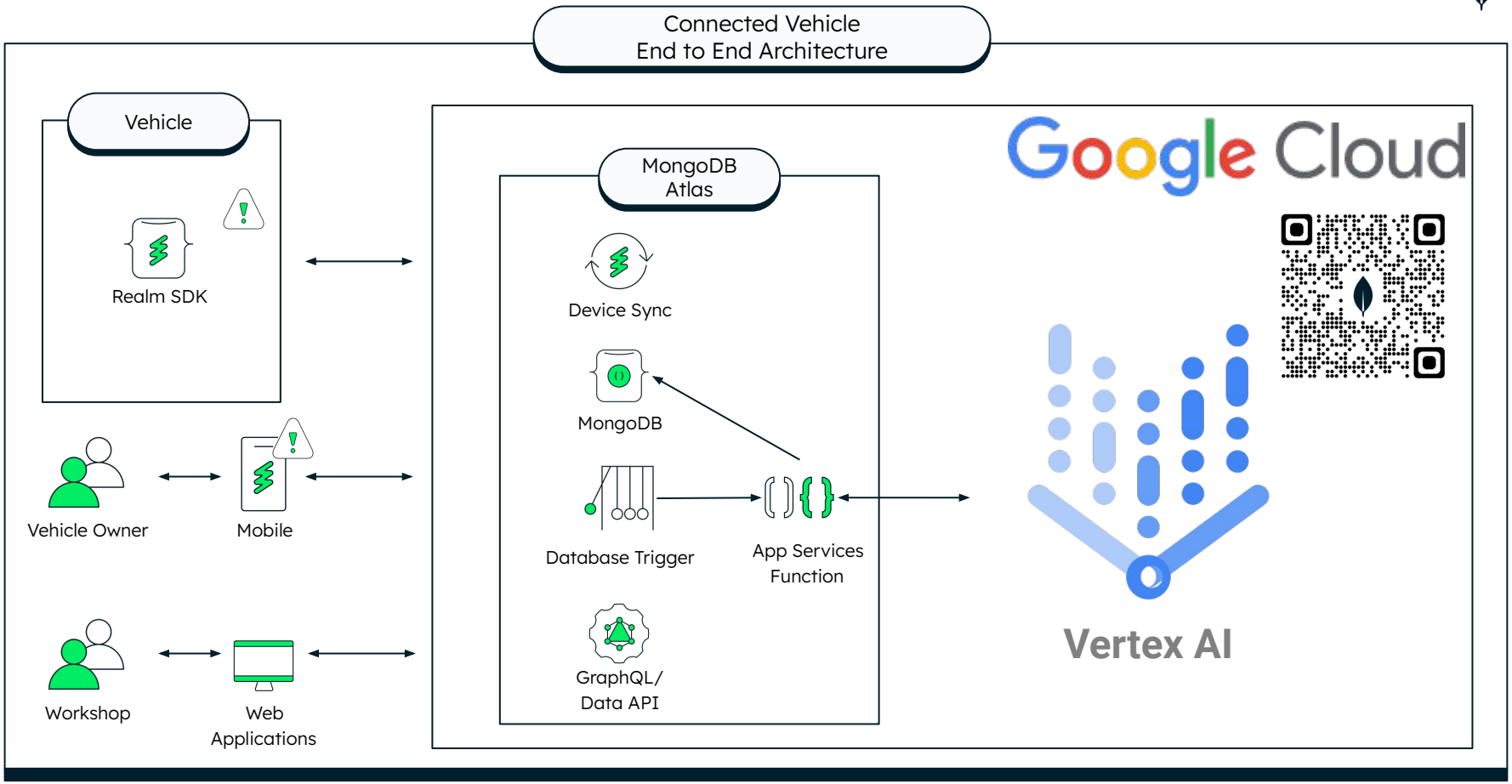
```
{
  "data": {
    "device": {
      "_id": "6299da19f8f54b2e29681cbb",
      "components": [
        {
          "name": "My First Component"
        },
        {
          "name": "My Second Component"
        }
      ],
      "isOn": false,
      "name": "My Device",
      "owner_id": "6270f5c2dbc08e4014fe0f48"
    }
  }
}
```

Connected Vehicle Demo Architecture



https://aws.amazon.com/blogs/industries/digital-twin-data-middleware-with-aws-and-mongodb/
https://aws.amazon.com/blogs/industries/how-to-solve-the-digital-twin-challenge-using-building-blocks-from-mongodb-on-aws/

Work in Progress



https://github.com/mongodb-industry-solutions/Vehicle-Digital-Twin-Feedback-Loop

Topics for Discussion

Unsolved Challenge(s)

- Conversion of hierarchical structures into classes leads to very long class names beyond class name length limits

Collaboration / Contribution

- Currently there is no VSS tooling for conversion to JSON schema

Curious? -> Reach out

industry.solutions@mongodb.coxm