uProtocol: Connecting Automotive Apps and Services, Everywhere

Frank Ghenassia
Our challenge
Connected vehicles

A network of connected devices hosting communicating Apps and Services

(*) for illustration purpose only
Our approach

Internet services

Connected vehicles

IoT networks

Vehicles: services consumers and providers

Cloud-assisted vehicle features

- Adopt and Adapt
- No lock-in
- Scale
No off-the-shelf solution covers all these features. uProtocol glues existing solutions into a coherent architecture covering all necessary features.
Communication

- **Location-independent**
  - Routing
  - URI

- **SOA 2.0**
  - Event-based
  - Pub/sub & RPC & notification design patterns
  - Authentication
  - Permissions

- **Event Message Types**
  - Message
  - File
  - Video

- **Discovery**
uProtocol: a layered approach

Software Entity

Payload

Header

CloudEvent

Application
Layer (uP-L3)

Communication
Layer (uP-L2)

Transport & Session
Layer (uP-L1)

Software Entity

protobuf

serialize

de-serialize

protobuf

send(CloudEvent)

Binder IPC

MQTT

HTTP/2

receive(CloudEvent)
Fostering an eco-system

Apps
- Hello World
- Dog Mode
- Telemetry

Productivity
- VSCode
- Auto-complete
- IntelliJ
- Eclipse Studio

Cloud
- uP-Azure
- uP-GCP
- uP-CPPrivate
- uP-AMZ

Services
- AC Controls
- Remote Access
- Camera

Mobile OS
- uP-Android
- uP-iOS
- uP-Linux

Vehicle OS
- uP-CA
- uP-AA

Transports
- MQTT
- Binder
- SOME/IP
- HTTP

Foundation
- uProtocol SDK
- Protocol spec: uI/m for Java, C++, Python
- Protobuf/APis
- Github test files

A production-proven contribution