VIRTUAL

Connext®
2021

rti

October 26-27, 2021

Dave "Whitt" Whitten
Senior Field
Application Engineer

✉ dwhitten@rti.com

Vinh Du
Field Application
Engineer

✉ vinh@rti.com

Inside DDS

Dave “Whitt” Whitten and Vinh Du

Field Application Engineers

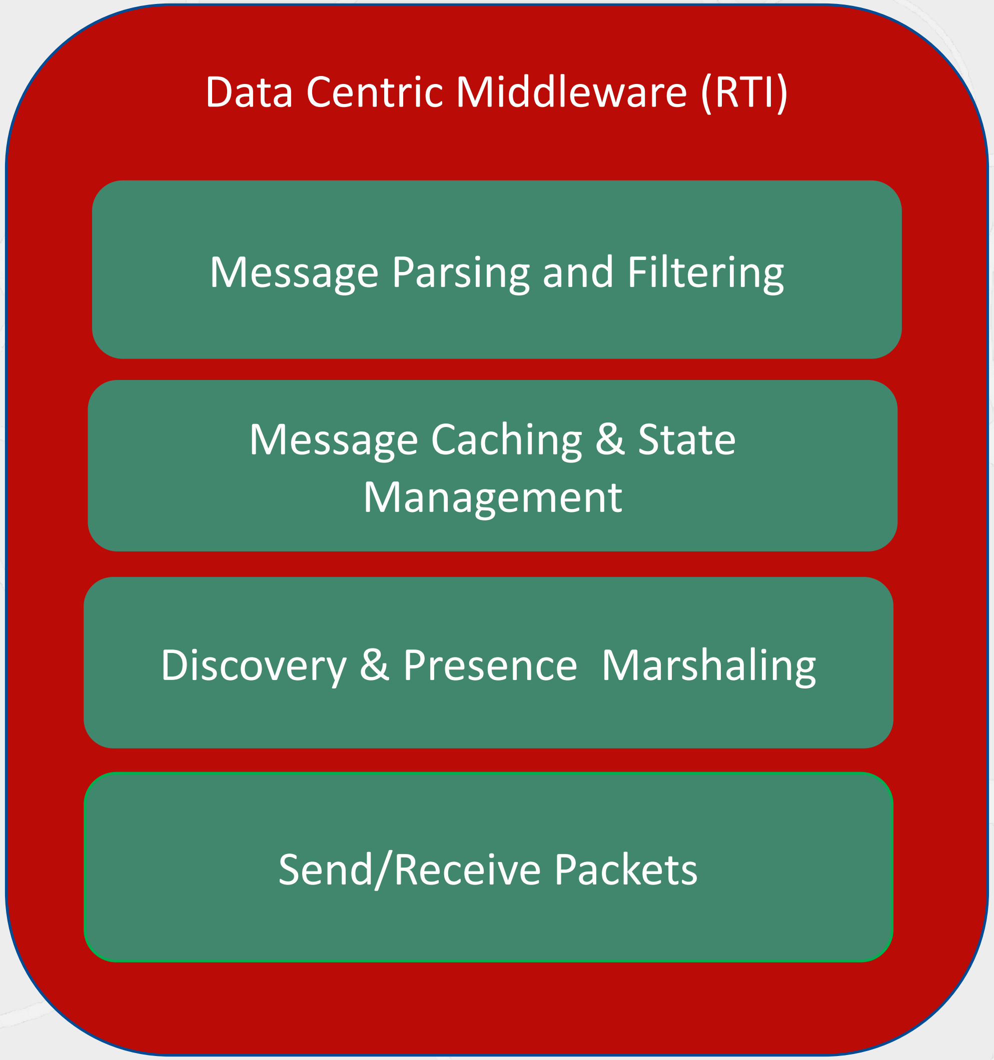
Agenda

- What is DDS?
- Introduction to DDS concepts
- Real-world Use Cases
- Connext DDS in action

What is DDS?

What is DDS?

Data-Distribution Service (DDS) is the first open international middleware standard directly addressing publish-subscribe communications for real-time and embedded systems.



Data Centric Middleware (RTI)

Message Parsing and Filtering

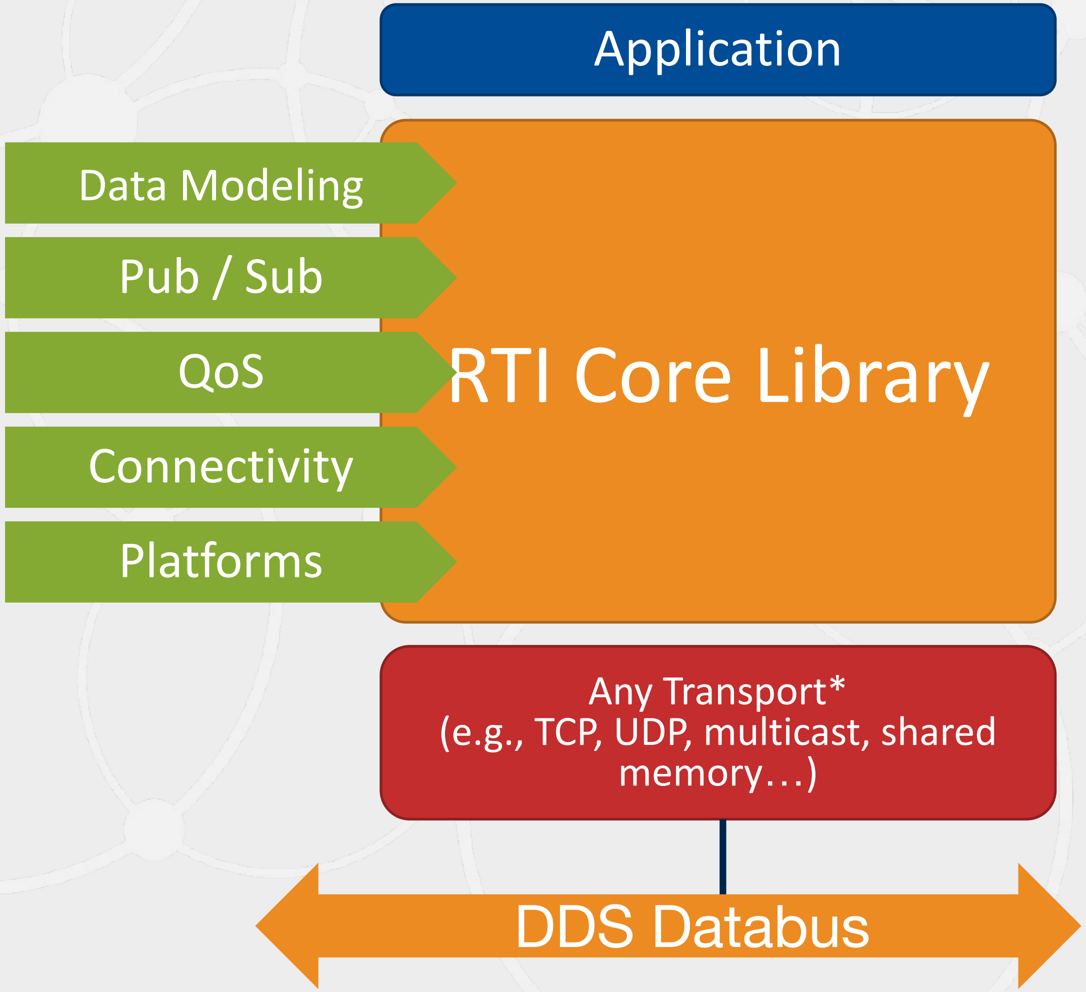
Message Caching & State Management

Discovery & Presence Marshaling

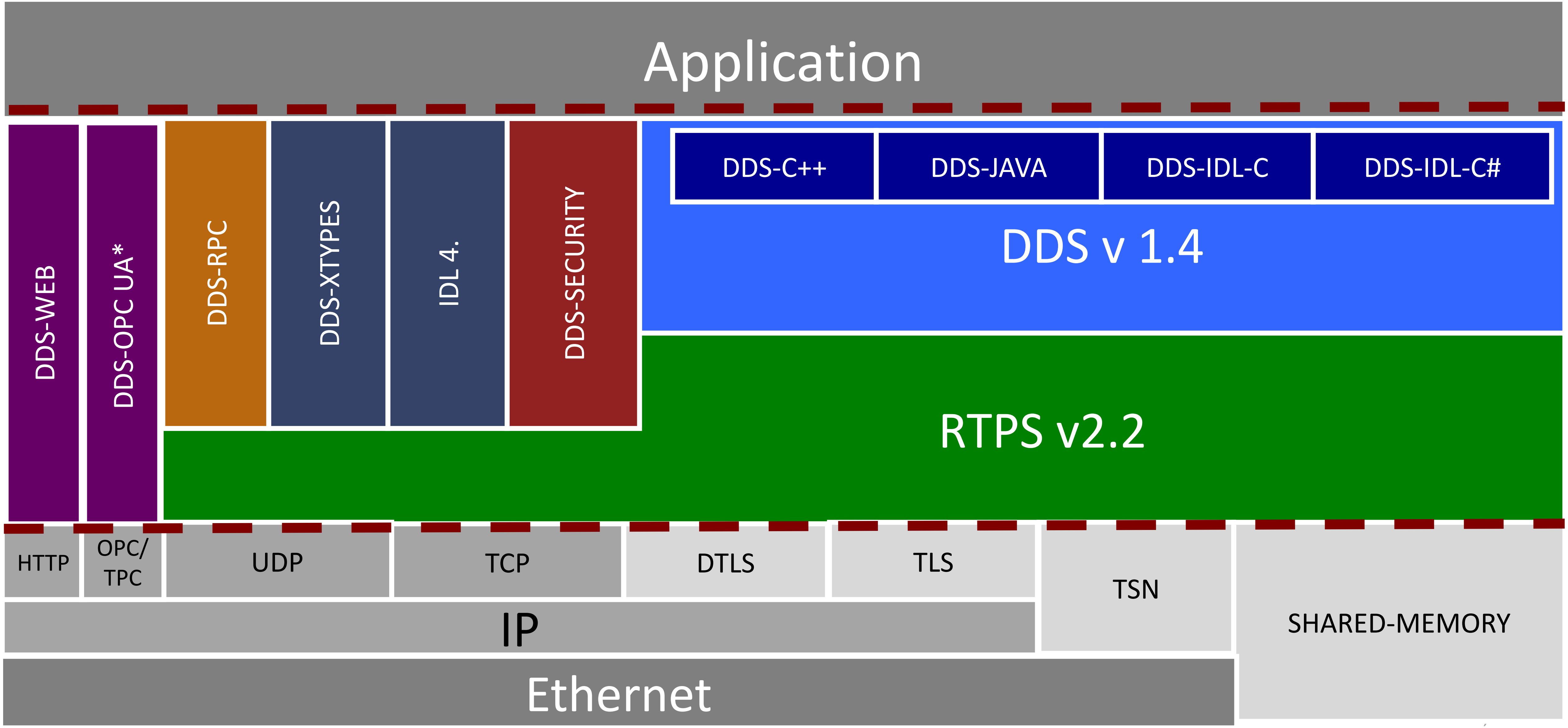
Send/Receive Packets

What are the benefits of using DDS?

- Decoupled applications
- Location transparency
- Future proofing
- Secure & Reliable
- Quality of Service



DDS Standard family



Application

DDS-WEB

DDS-OPC UA*

DDS-RPC

DDS-XTYPES

IDL 4.

DDS-SECURITY

DDS-C++

DDS-JAVA

DDS-IDL-C

DDS-IDL-C#

DDS v 1.4

RTPS v2.2

HTTP

OPC/
TPC

UDP

TCP

DTLS

TLS

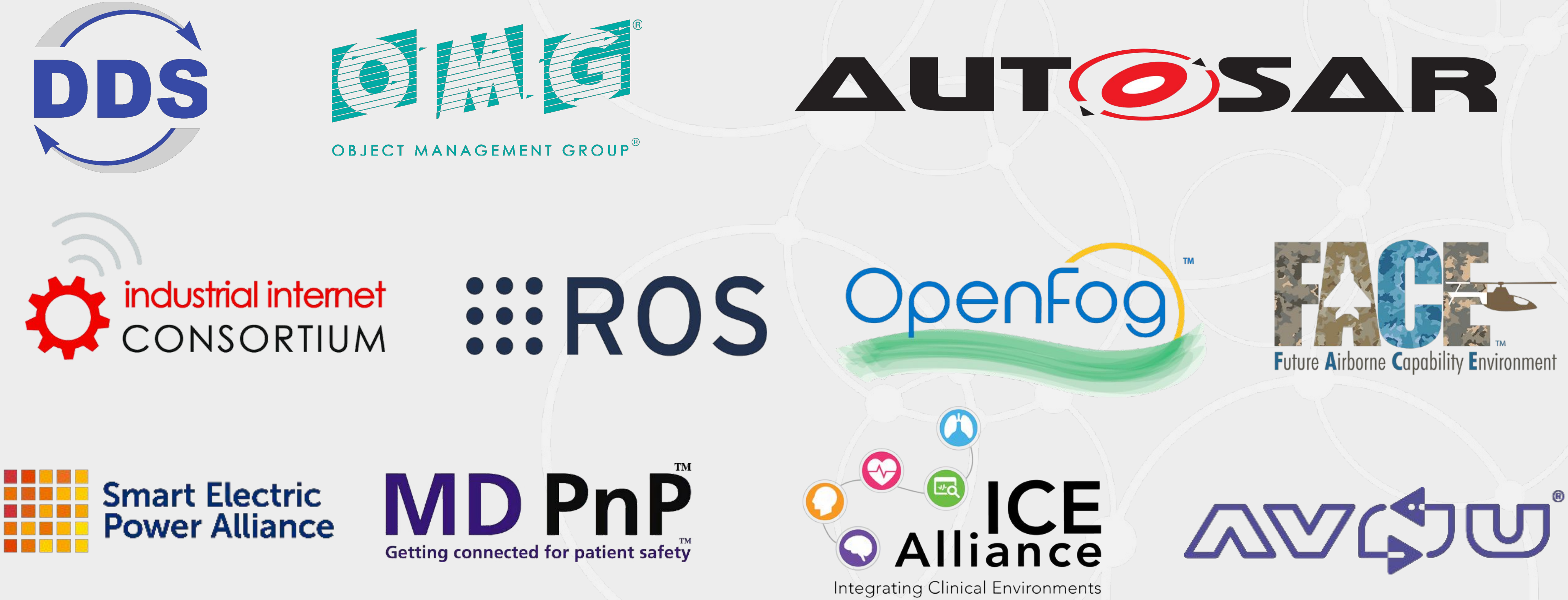
TSN

SHARED-MEMORY

IP

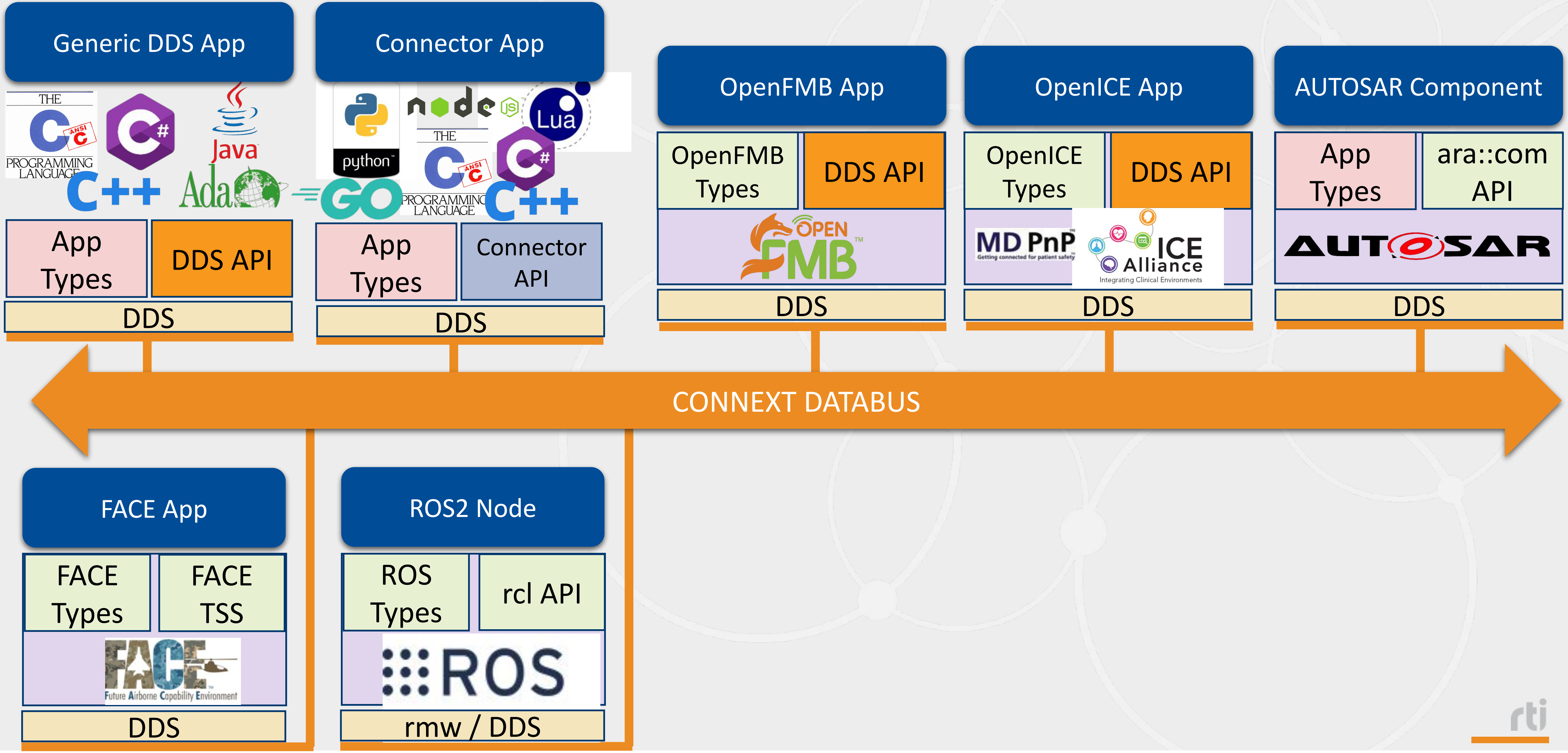
Ethernet

15+ Standards and Consortia Efforts


DDS
OMG
OBJECT MANAGEMENT GROUP®
AUTOSAR
industrial internet CONSORTIUM
ROS
OpenFog™
FACE™
Future Airborne Capability Environment
Smart Electric Power Alliance
MD PnP™
Getting connected for patient safety™
ICE Alliance
Integrating Clinical Environments
AVNU®

rti

Connext DDS as Core Connectivity Framework



Prevalence of RTI Connext

Deployed in 1000s of Unique Systems

Market Leader

- Largest embedded middleware vendor

1000+ Designs

- Defense
- Aerospace
- Healthcare
- Transportation
- Energy

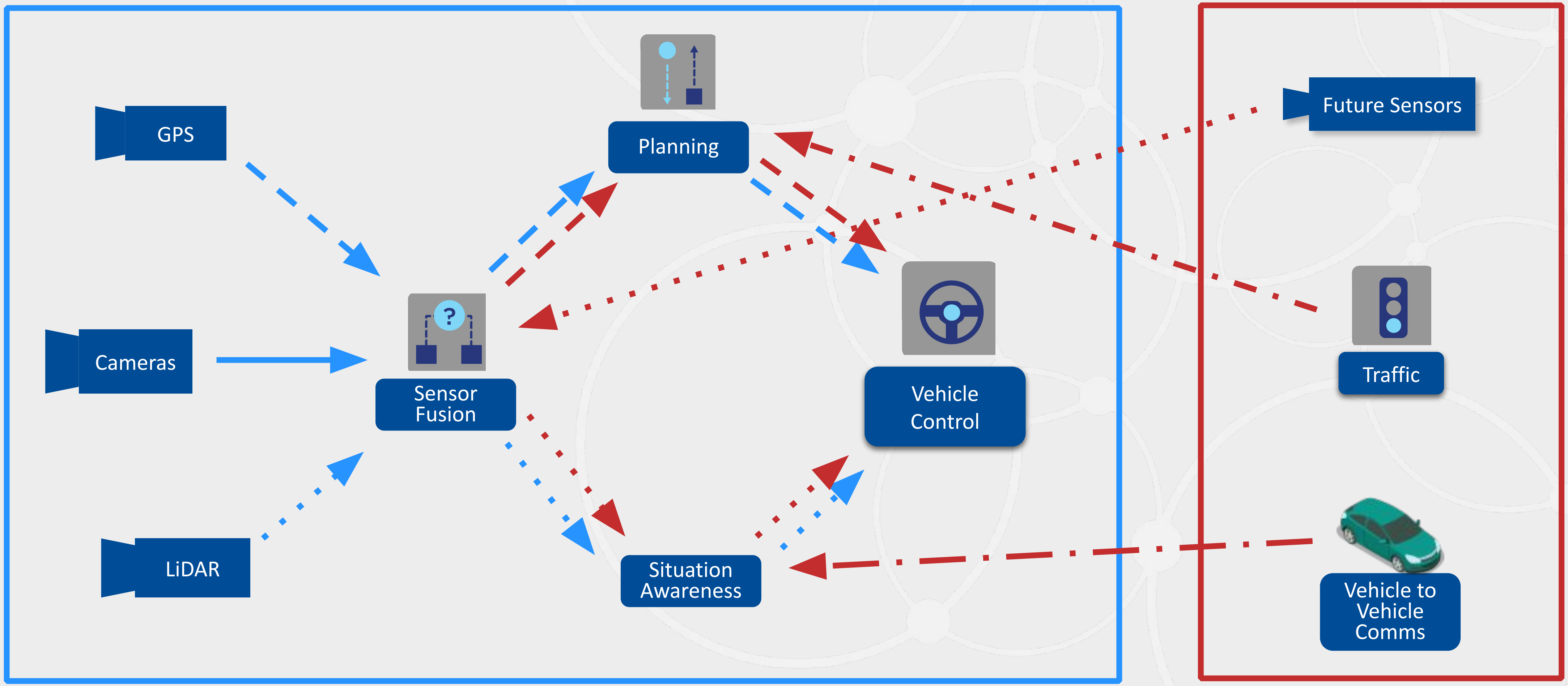
DO-178C Level A Safety Artifacts

Technology Readiness Level (TRL) 9

15+ Standards & Consortia Efforts

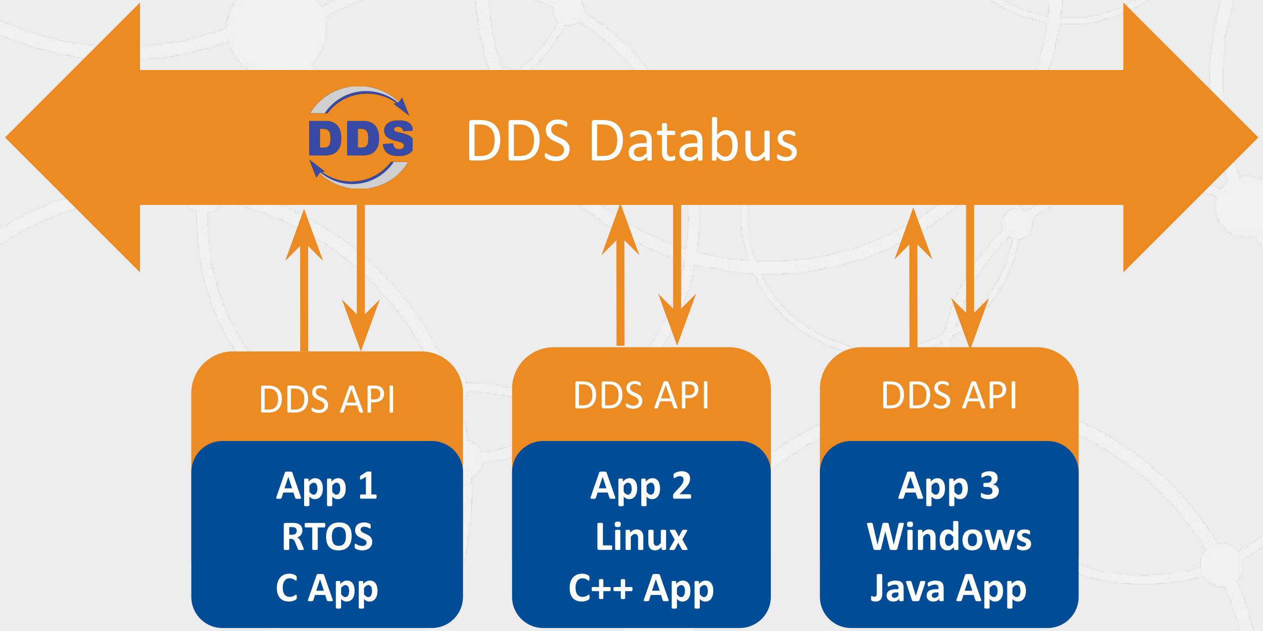
Introduction to DDS concepts

First an example without DDS

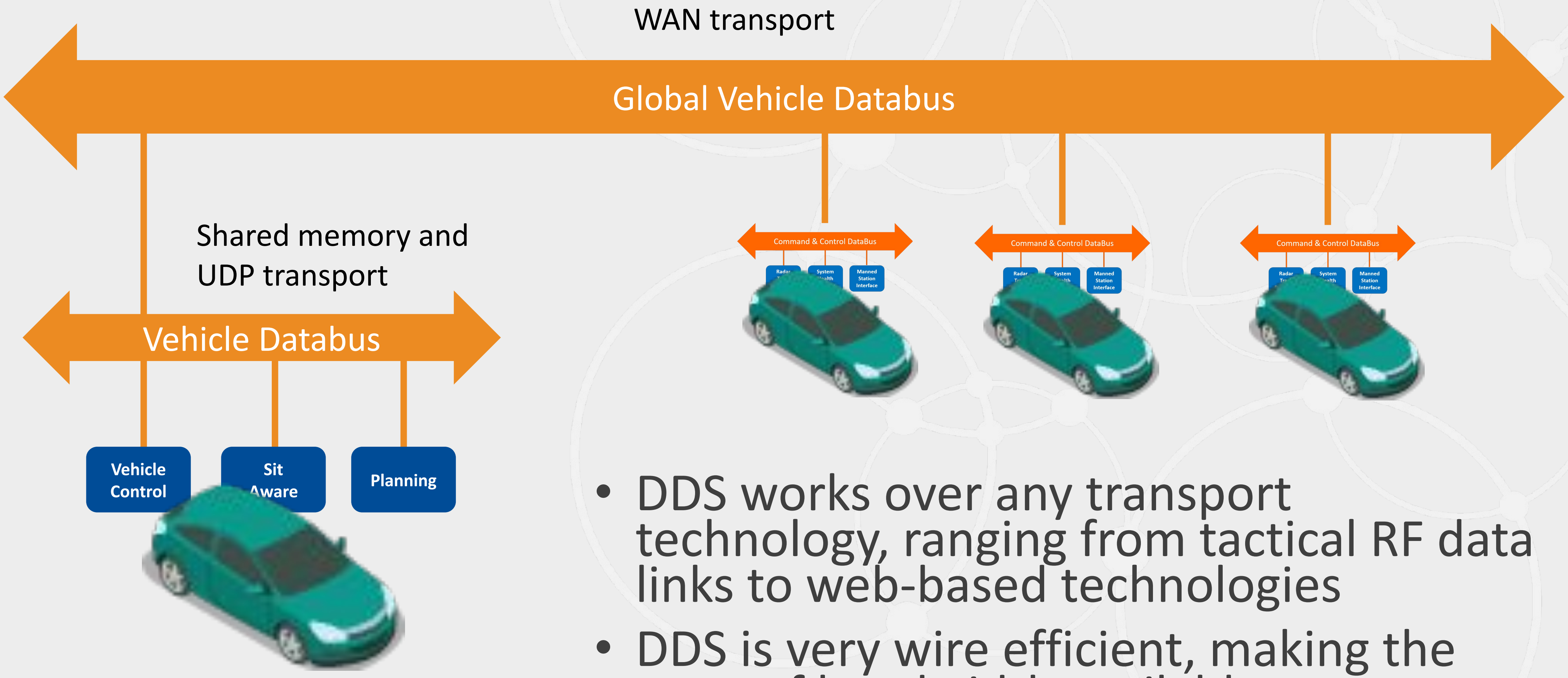


DDS helps cross programming language and OS boundaries

- Out of the box library support for 100+ platforms(operating system + CPU combination)
- Multiple language bindings
  - C
  - C++
  - Modern C++(03 and 11)
  - Java
  - Ada
  - Python
  - Lua
  - Javascript

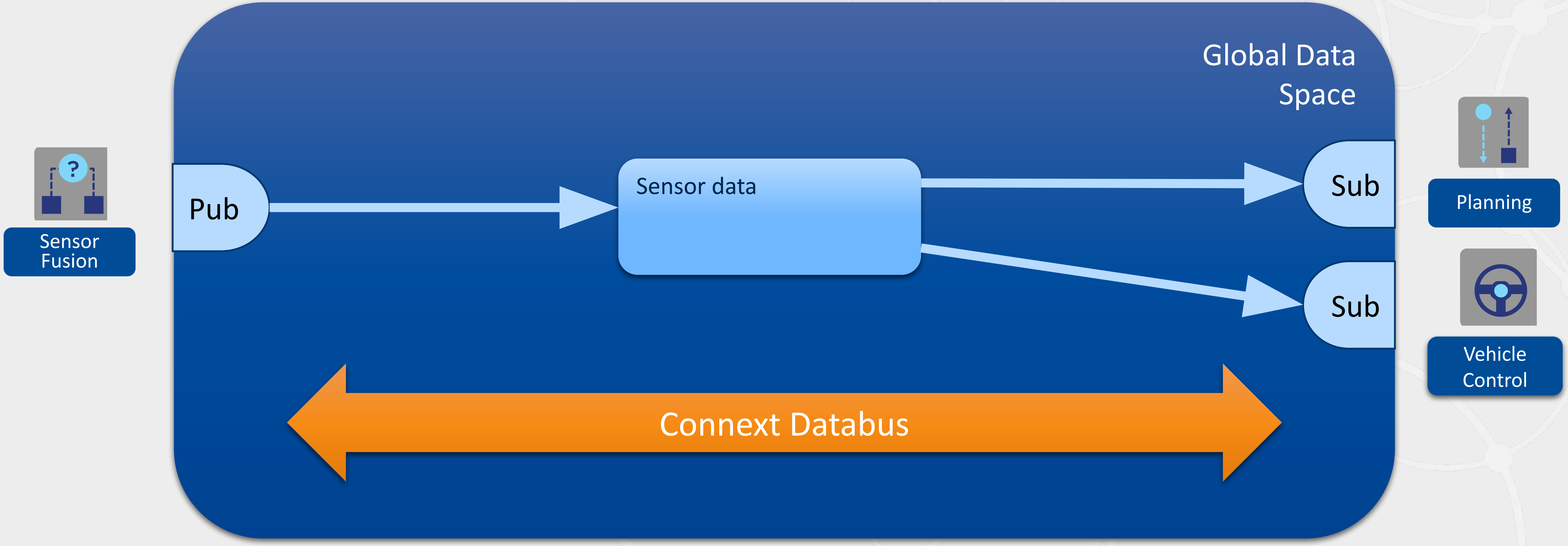


DDS is Transport Agnostic



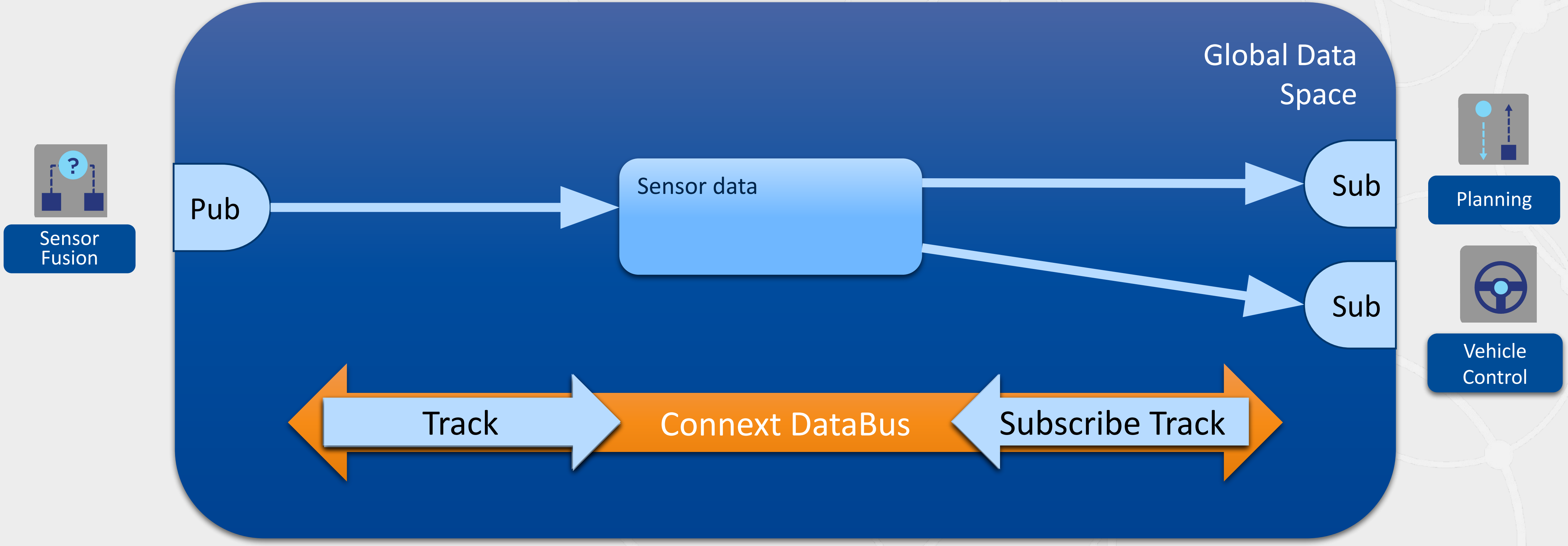
- DDS works over any transport technology, ranging from tactical RF data links to web-based technologies
- DDS is very wire efficient, making the most of bandwidth available.

DDS utilizes a Publish/Subscribe Pattern



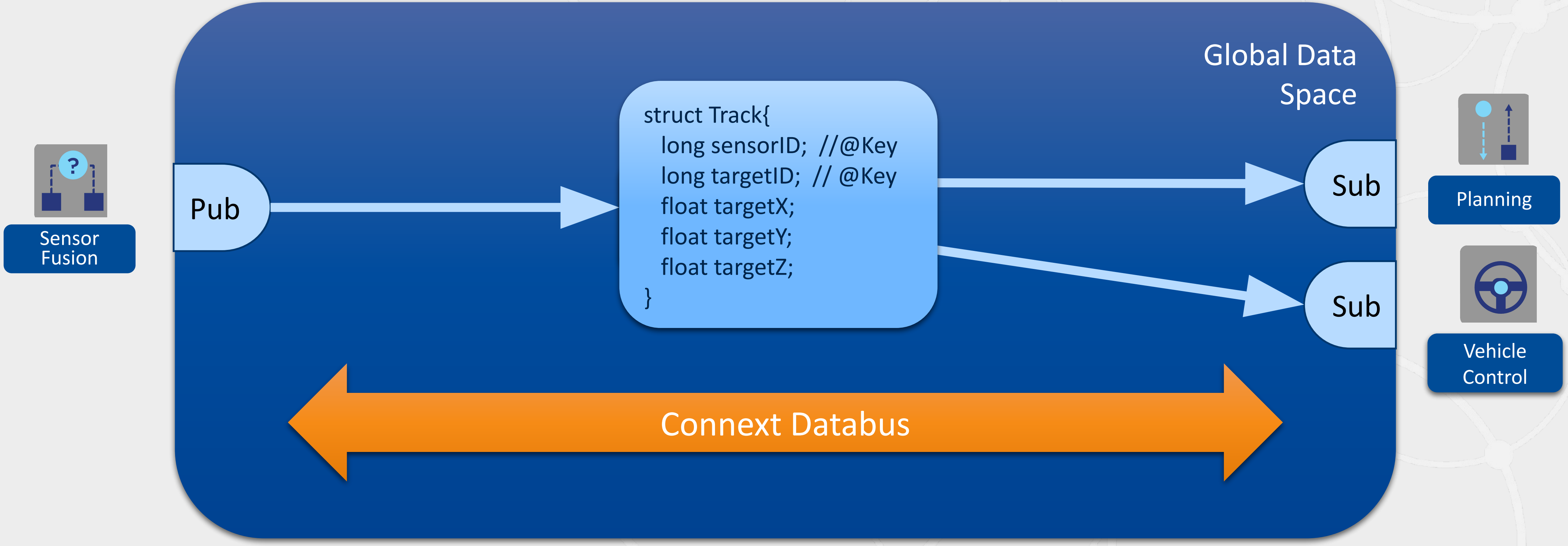
Basic Publish/Subscribe concept. Publishers offer data of a specific type, and subscribers subscribe to data of interest. A publisher can be the source of data for multiple subscribers

Peer Discovery matches Publishers and Subscribers



The process by which Publishers and Subscribers find each other is discovery. Generically think of it as a broadcast announcement by a publisher, and potential subscribers responding to it to determine interest. Note, the discovery process happens often meaning DDS provides **loosely coupled endpoints.**

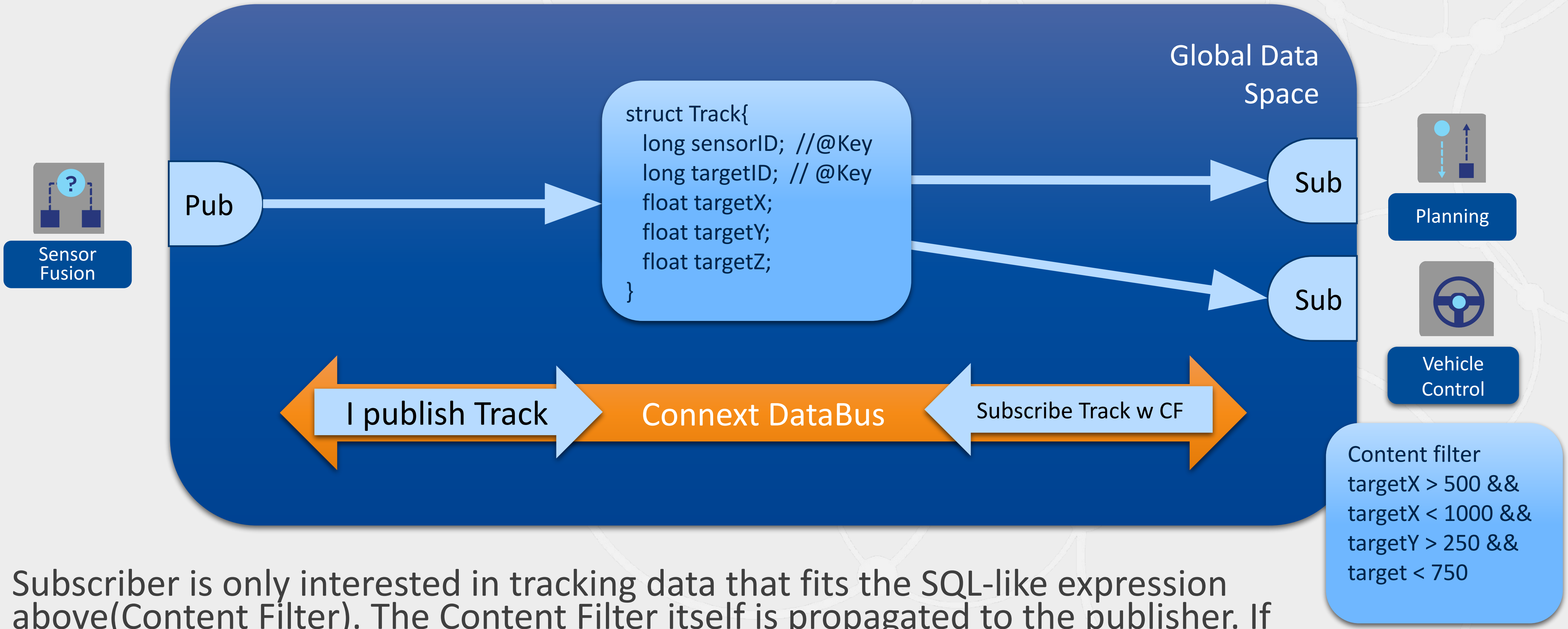
Publish/subscribe with data-centricity



```
struct Track{
  long sensorID;  //@Key
  long targetID;  // @Key
  float targetX;
  float targetY;
  float targetZ;
}
```

One of the strengths of DDS is that the data model is hard-typed and shared. This means DDS can apply intelligence to messages since it can understand the contents. This is what we call data-centricity.

Data Centricity in action



```
struct Track{
  long sensorID;  //@Key
  long targetID;  // @Key
  float targetX;
  float targetY;
  float targetZ;
}
```

Content filter
targetX > 500 &&
targetX < 1000 &&
targetY > 250 &&
target < 750

Subscriber is only interested in tracking data that fits the SQL-like expression above(Content Filter). The Content Filter itself is propagated to the publisher. If no one is interested in a particular sample based on that filter, that packet is NOT transmitted (vs sent and ignored by subscriber).

How is DDS data aware?

- Data that is shared on a DDS databus is defined via an OMG-defined specification called Interface Definition Language(IDL)
  - https://www.omg.org/spec/IDL/
- IDL defines the data member types for user defined data types
- IDL is a shared data model across DDS applications that are interested in a particular type of data
- This enables DDS to know what the data looks like so that it can handle serialization/deserialization, endianness, and allows distributed intelligence based on the contents

Simple IDL example:

```
struct Track{
    long sensorID;     //@key
    long targetID;     //@key
    float targetX;
    float targetY;
    float targetZ;
}
```

Data Model Extensibility

- Final Type
  - Type definitions are strictly defined

- Extensible Type
  - Newer applications can add fields to existing base types
  - This means you can use new message contents on new aspects of a system and not break or need to modify the legacy component

- Mutable Type
  - Type representations can differ from each other with Additions, Deletions and Transpositions
  - Support for Optional Fields

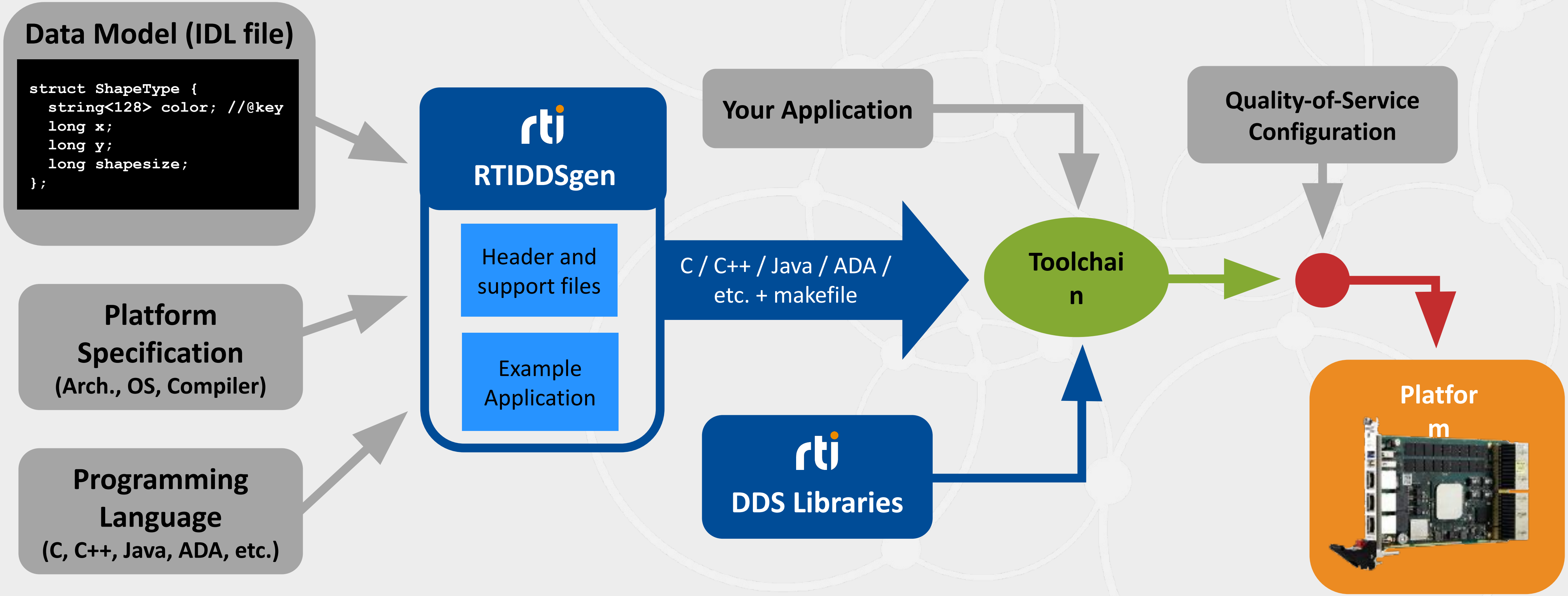
Base Type

```
struct Track{
    long sensorID;    //@key
    long targetID;    //@key
    float targetX;
    float targetY;
    float targetZ;
}
```

Extended Type

```
struct timedTrack{
    long sensorID;    //@key
    long targetID;    //@key
    float targetX;
    float targetY;
    float targetZ;
    double timestamp;
}
```

Typical Development Workflow for Connext



Data Model (IDL file)

```
struct ShapeType {
  string<128> color; //@key
  long x;
  long y;
  long shapesize;
};
```

Platform Specification
(Arch., OS, Compiler)

Programming Language
(C, C++, Java, ADA, etc.)

rti
RTIDDSgen

Header and support files

Example Application

C / C++ / Java / ADA / etc. + makefile

Your Application

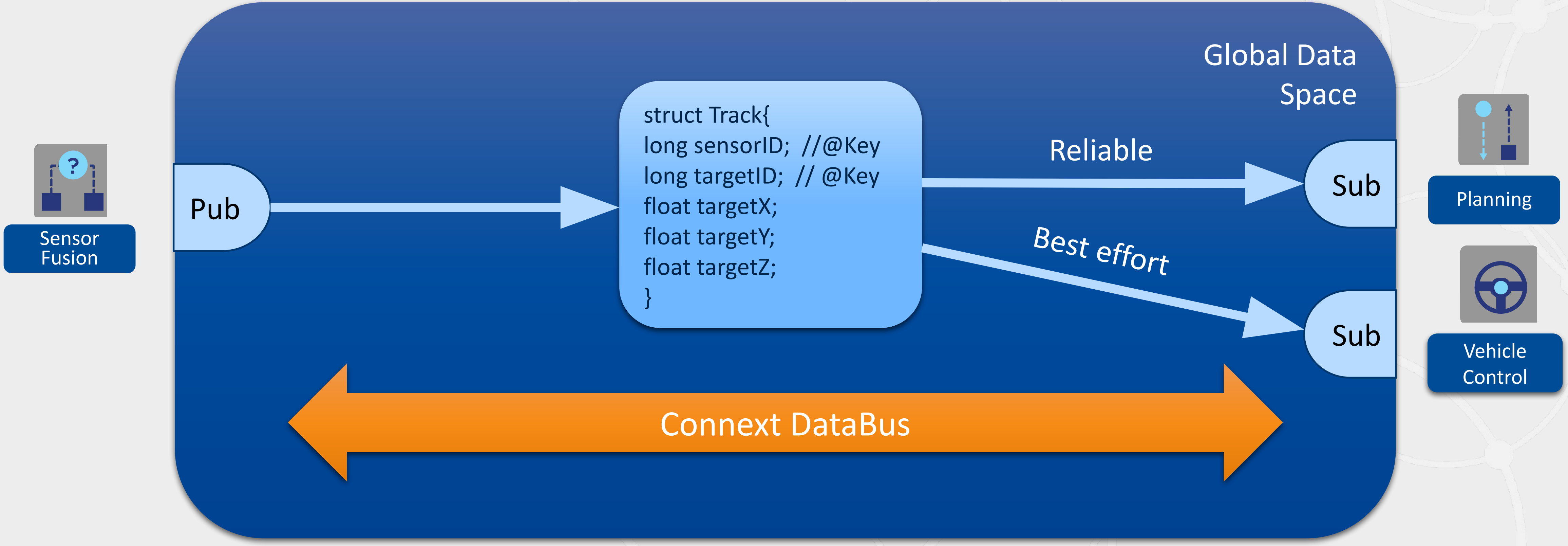
rti
DDS Libraries

Toolchai
n

Quality-of-Service Configuration

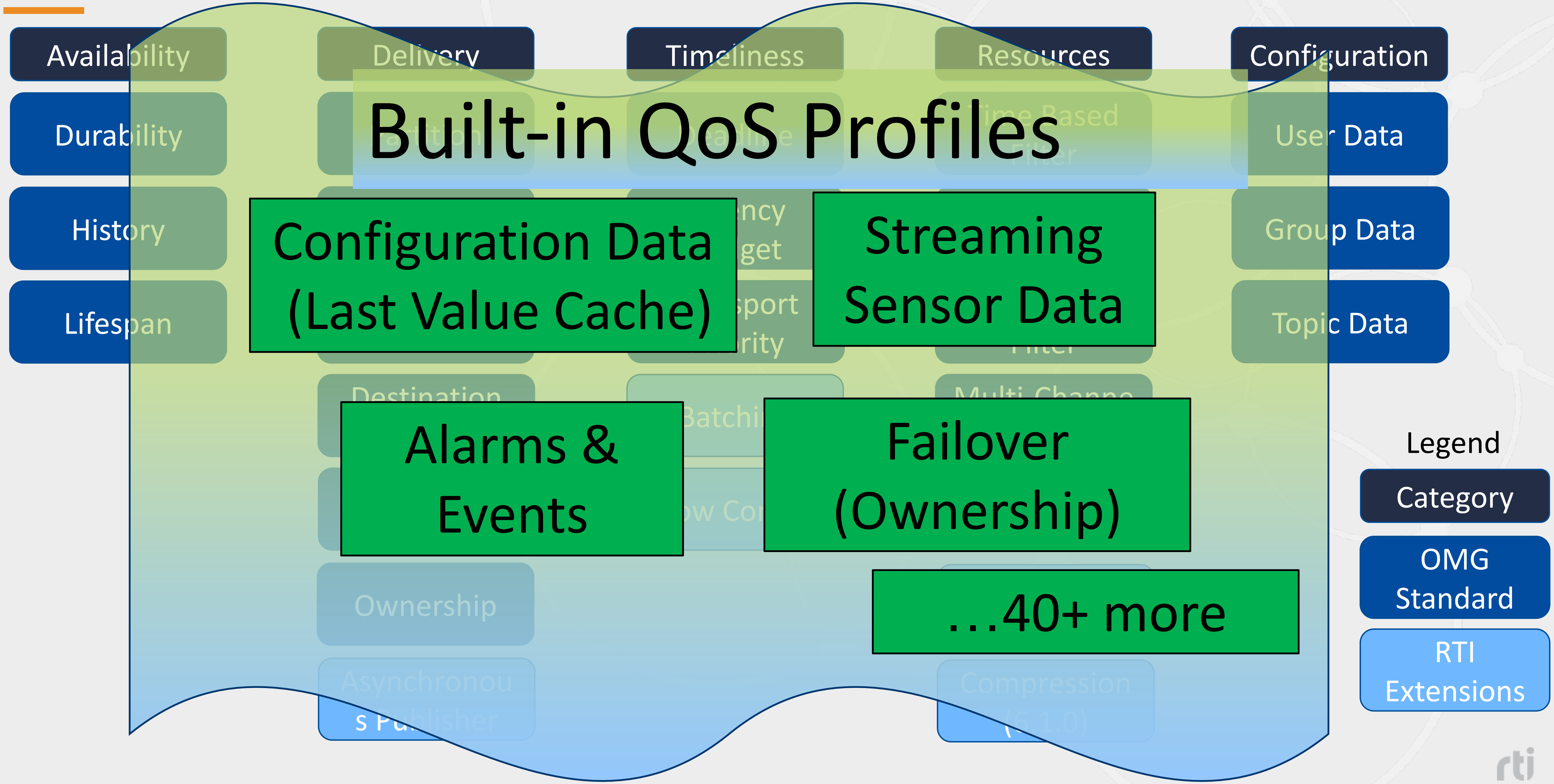
Platfor
m

Quality of Service - Not all data is treated the same



In the example above, one Publisher can provide data in two very different ways to two subscribers. Quality of Service is negotiated during the peer discovery process discussed earlier.

DDS Quality of Service (QoS)



Built-in QoS Profiles

Configuration Data (Last Value Cache)

Streaming Sensor Data

Alarms & Events

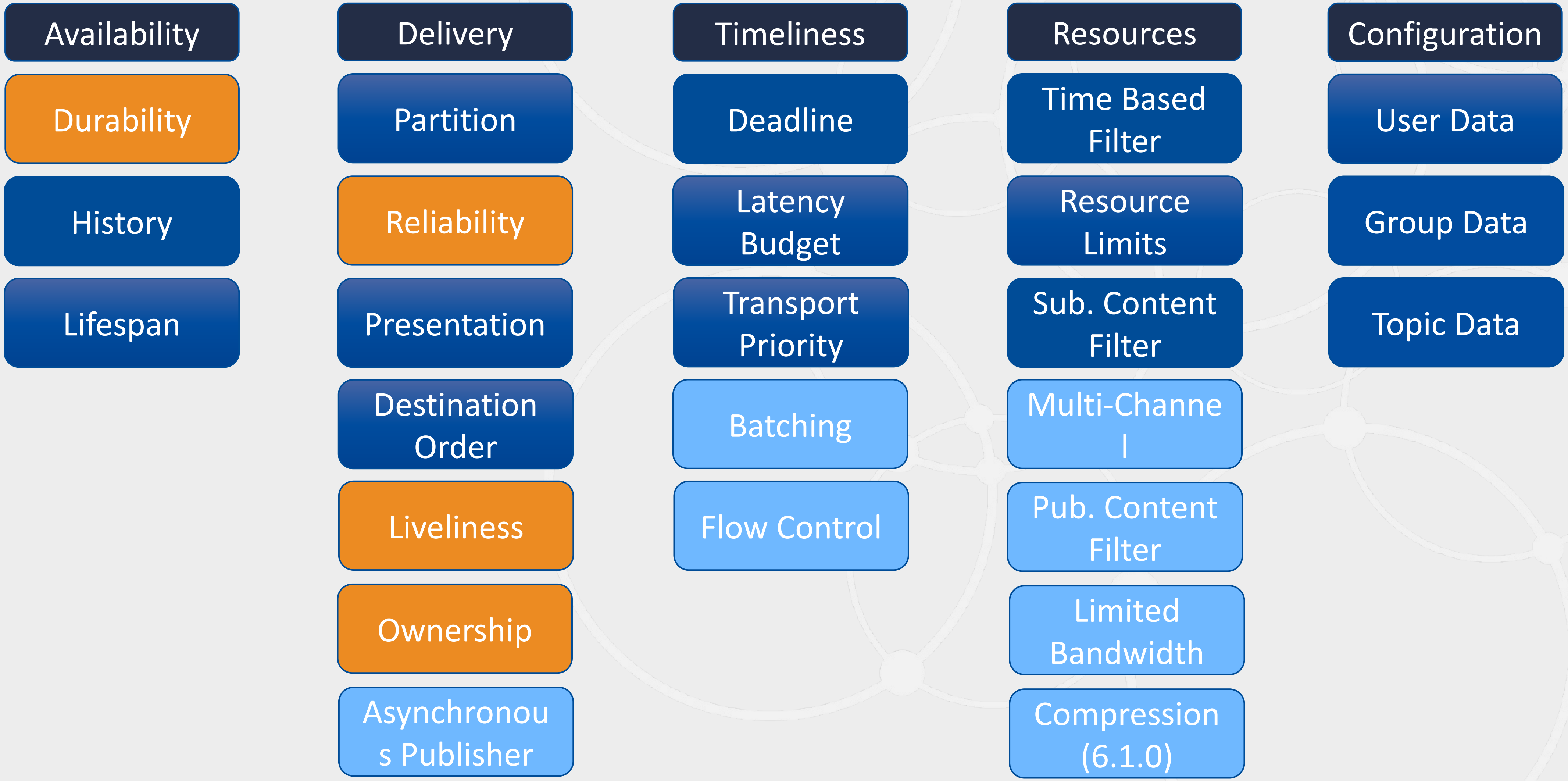
Failover (Ownership)

…40+ more

QoS Use Case: Alarms / Events

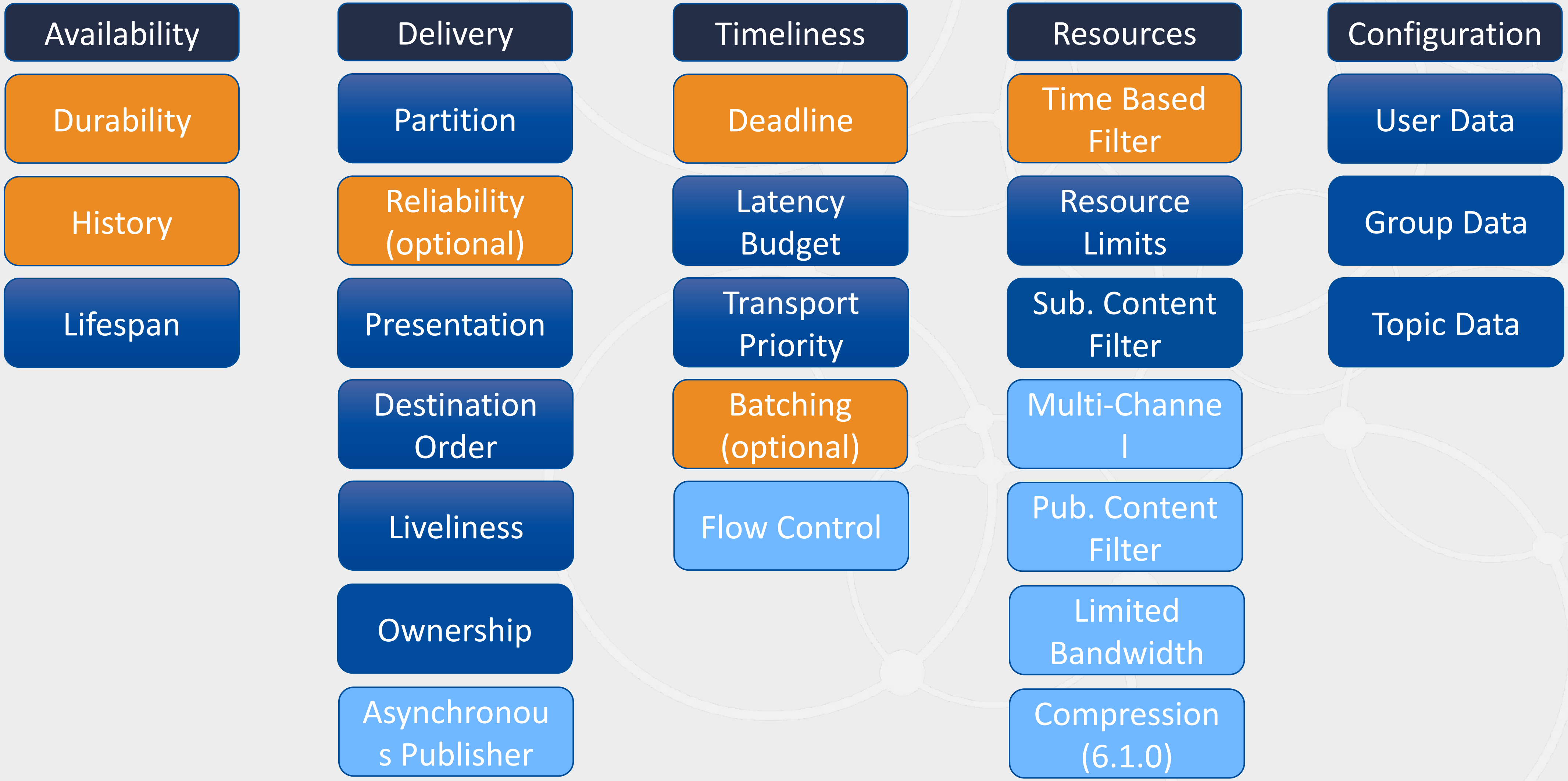
Availability	Delivery	Timeliness	Resources	Configuration
Durability	Partition	Deadline	Time Based Filter	User Data
History	Reliability	Latency Budget	Resource Limits	Group Data
Lifespan	Presentation	Transport Priority	Sub. Content Filter	Topic Data
	Destination Order	Batching	Multi-Channel	
	Liveliness	Flow Control	Pub. Content Filter	
	Ownership		Limited Bandwidth	
	Asynchronous Publisher		Compression (6.1.0)	

QoS Use Case: Failover



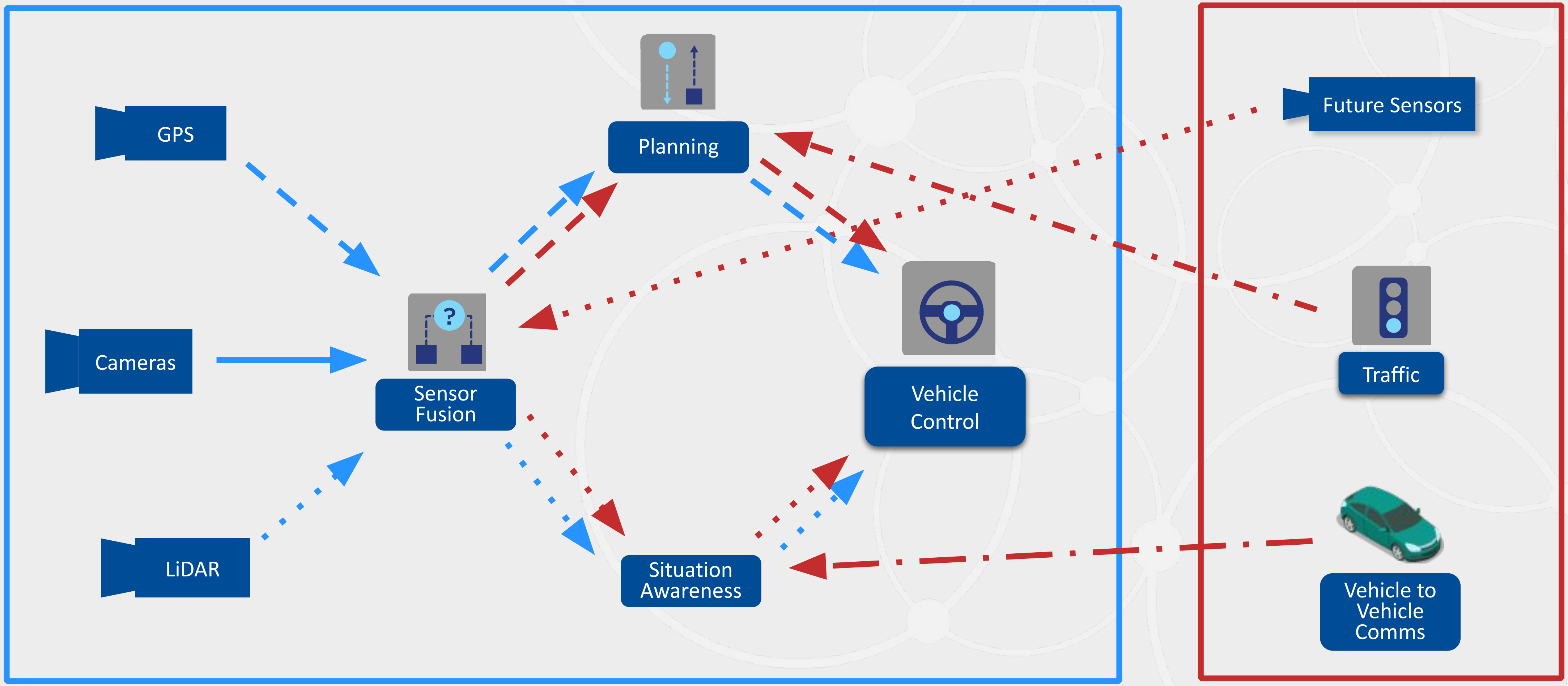
Availability	Delivery	Timeliness	Resources	Configuration
Durability	Partition	Deadline	Time Based Filter	User Data
History	Reliability	Latency Budget	Resource Limits	Group Data
Lifespan	Presentation	Transport Priority	Sub. Content Filter	Topic Data
	Destination Order	Batching	Multi-Channe l	
	Liveliness	Flow Control	Pub. Content Filter	
	Ownership		Limited Bandwidth	
	Asynchronou s Publisher		Compression (6.1.0)	

QoS Use Case: Streaming Data

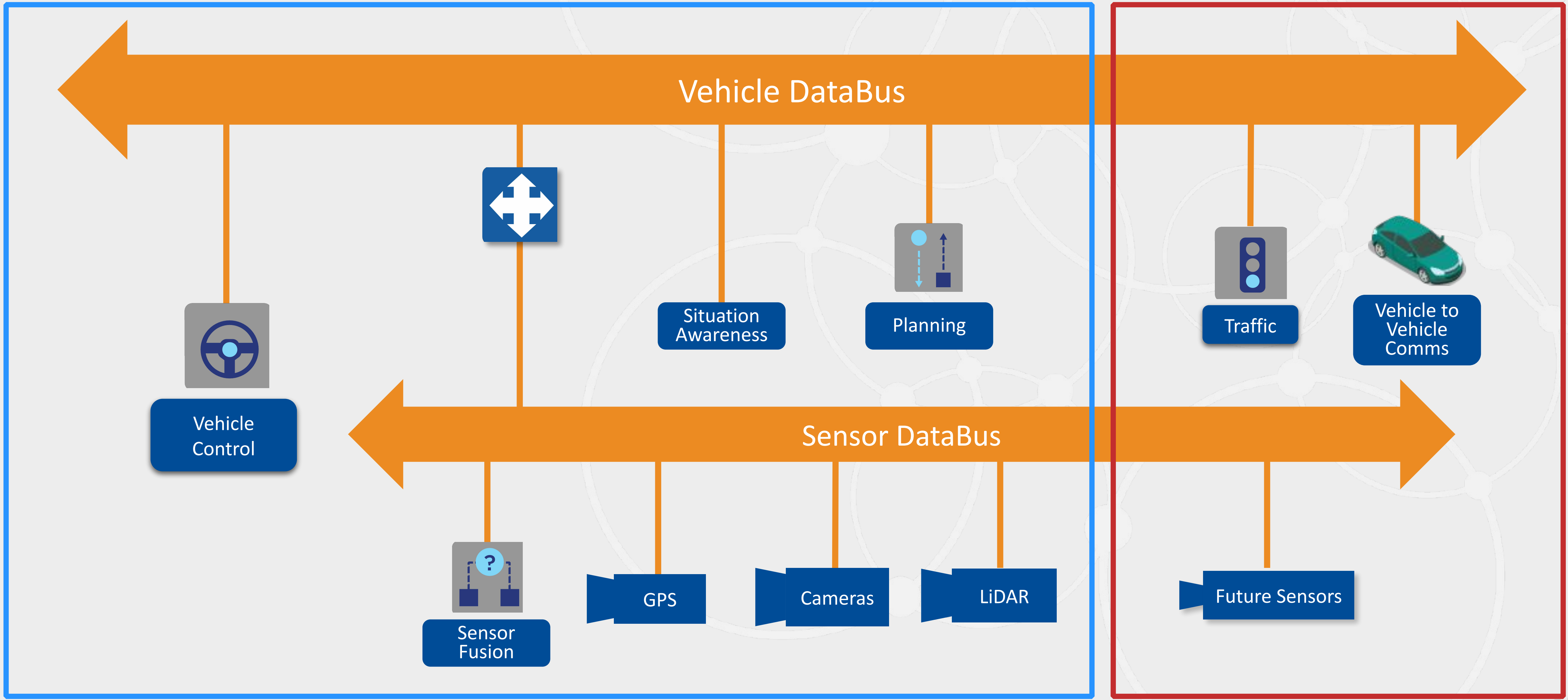


Availability	Delivery	Timeliness	Resources	Configuration
Durability	Partition	Deadline	Time Based Filter	User Data
History	Reliability (optional)	Latency Budget	Resource Limits	Group Data
Lifespan	Presentation	Transport Priority	Sub. Content Filter	Topic Data
	Destination Order	Batching (optional)	Multi-Channel	
	Liveliness	Flow Control	Pub. Content Filter	
	Ownership		Limited Bandwidth	
	Asynchronous Publisher		Compression (6.1.0)	

Revisiting the Example



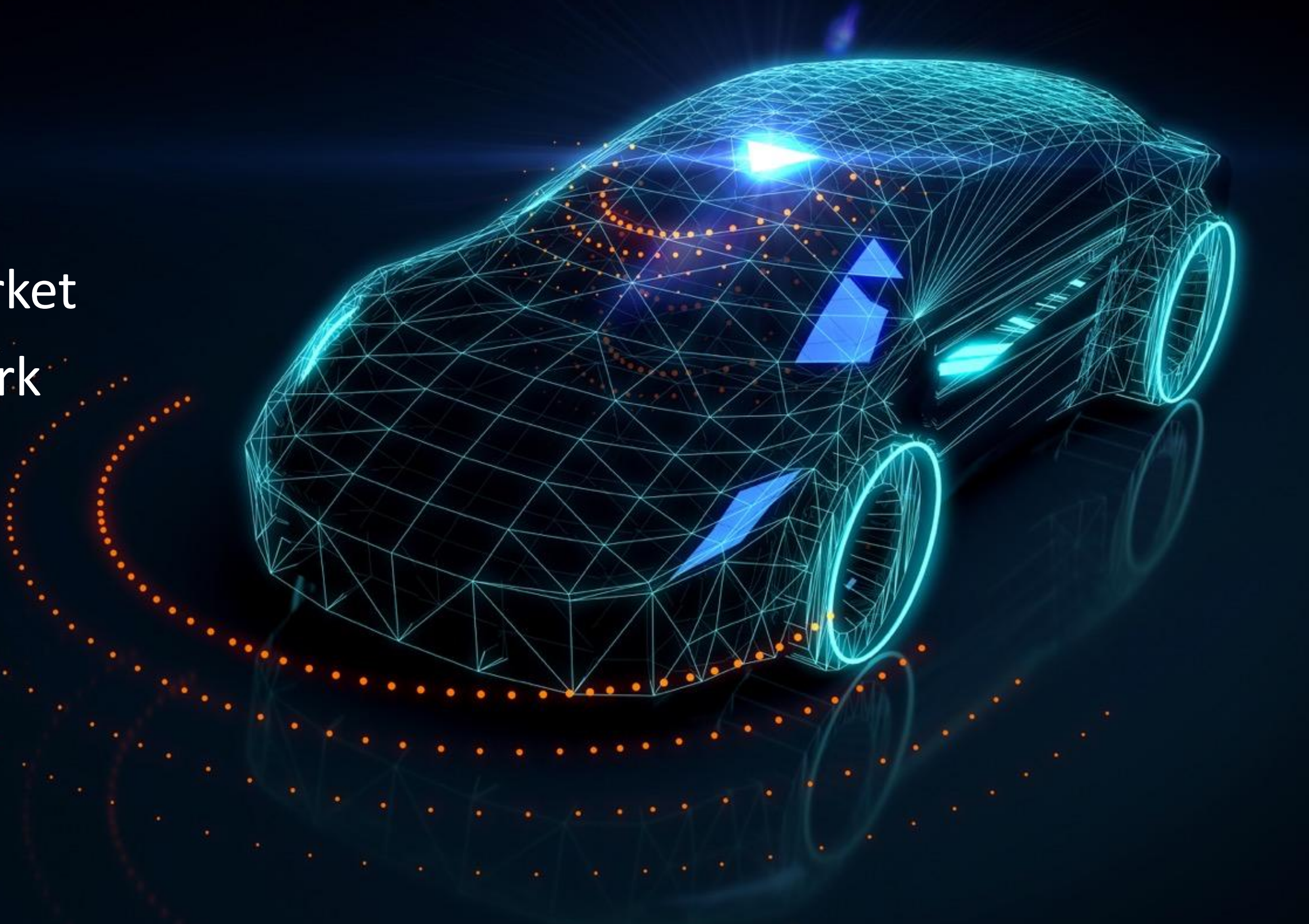
Revisiting the Example with DDS



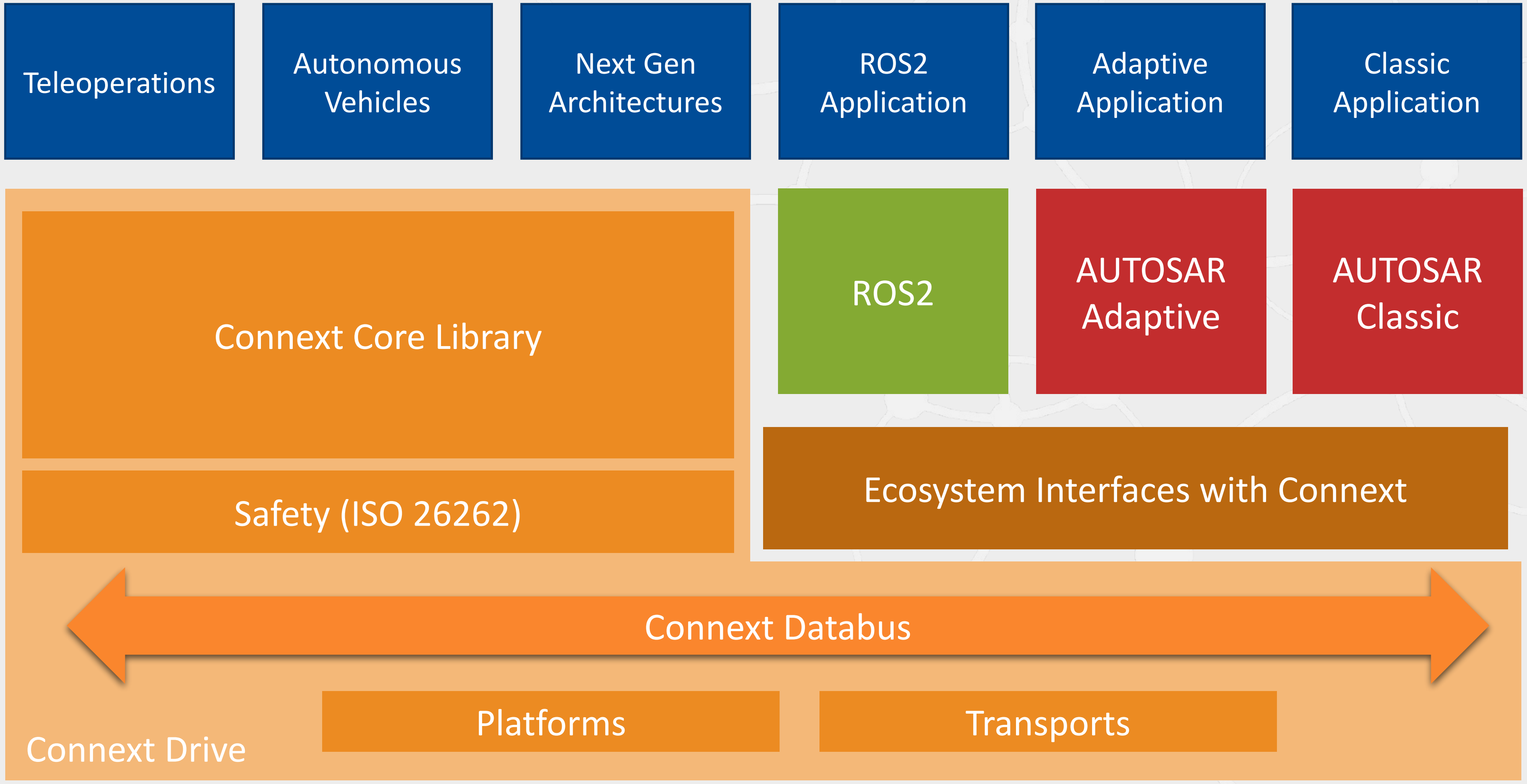
Real-world DDS Use Cases

Automotive Use Case

- Simple solution for Automotive customers
- Designed for the Automotive Market
- Complete ECU to Cloud Framework
- Only proven-in-use framework that will meet all Autonomous use cases
- Future Proof, data-centric architecture will support industry evolution



Connext Drive - Developer Framework & SDK



Teleoperations

Autonomous Vehicles

Next Gen Architectures

ROS2 Application

Adaptive Application

Classic Application

Connext Core Library

Safety (ISO 26262)

ROS2

AUTOSAR Adaptive

AUTOSAR Classic

Ecosystem Interfaces with Connext

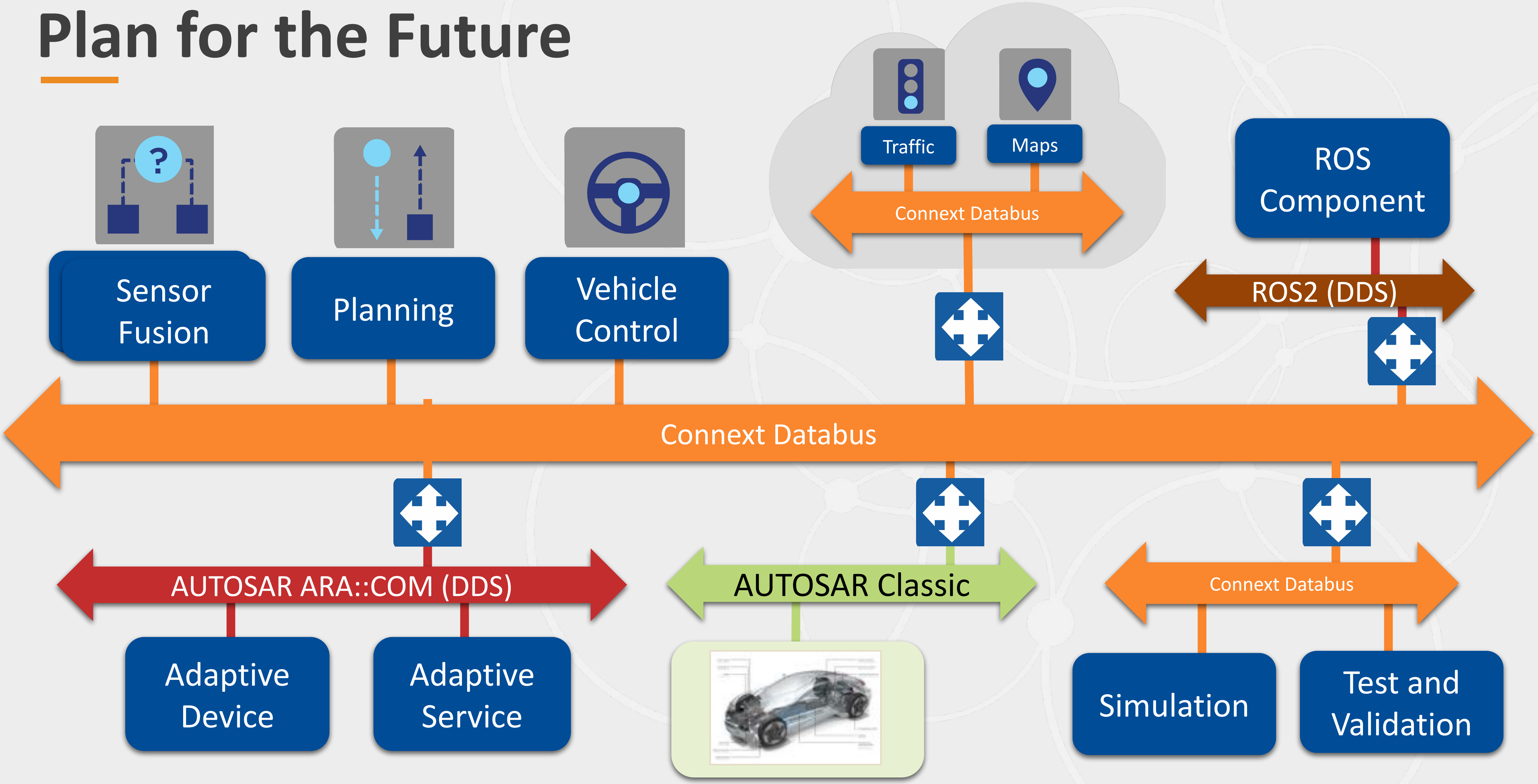
Connext Databus

Platforms

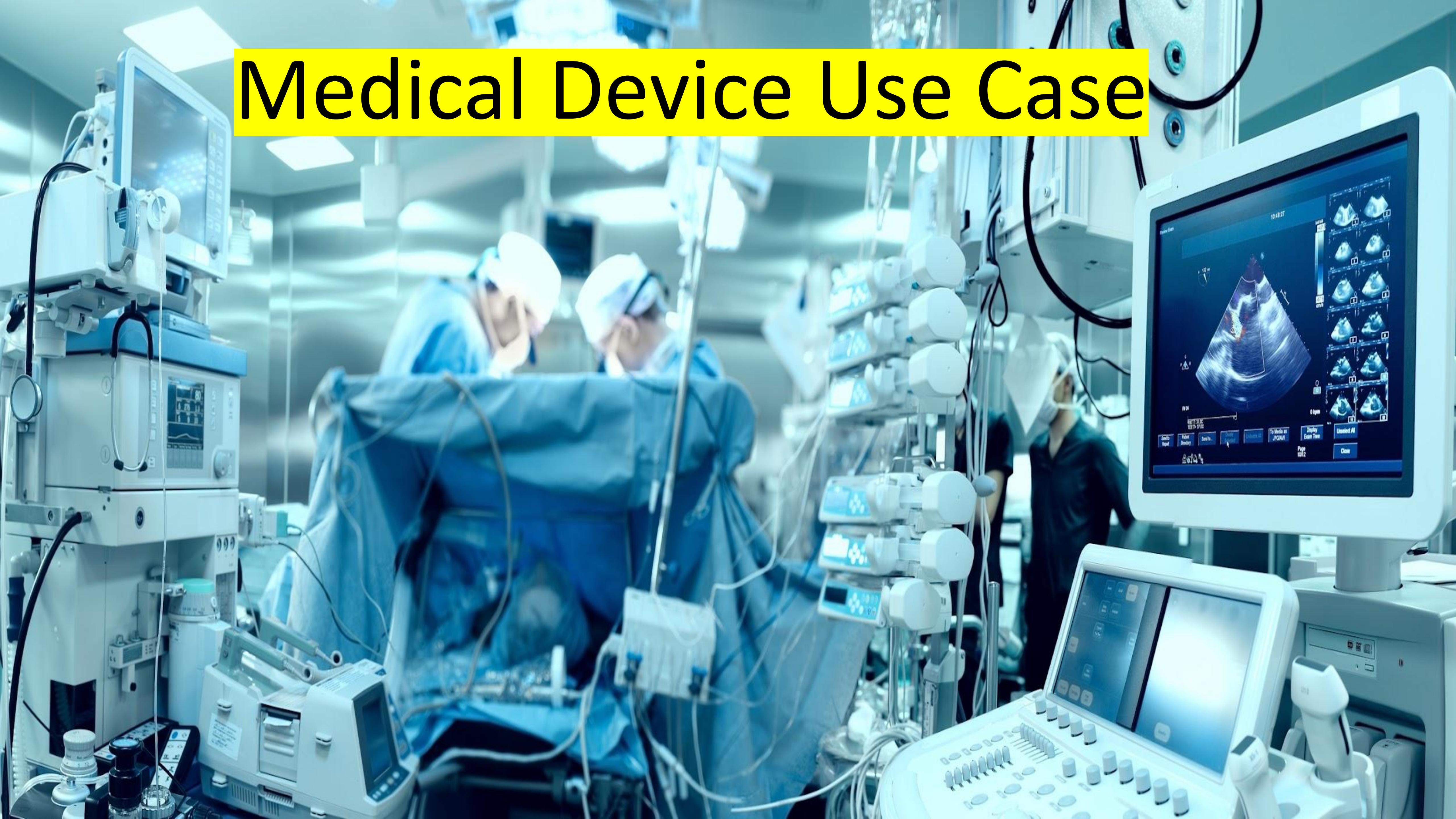
Transports

Connext Drive

Plan for the Future



Medical Device Use Case



Benefits of Robotically-Assisted Procedures

- Smaller incisions
- More precise and effective treatments
- Faster patient recovery

Copyright © Corindus, Inc. All rights reserved

©2012 Intuitive Surgical, Inc.

Robot-assisted surgery system

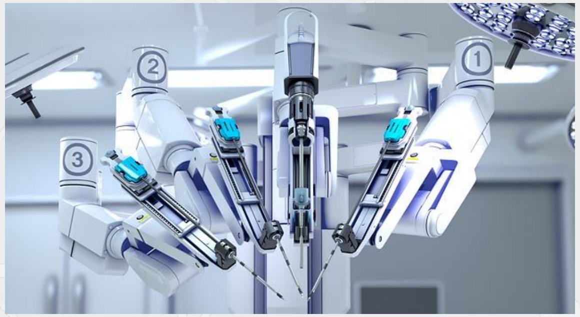
Controller – Nodes:

- Safety, visualization
- Power Control, foot switch emulator
- Gateway

DDS Command and Control

Robot Arms

- Real-time OS
- Low Latency High Reliability



Surgeon Console

- Realtime and HMI systems

Control Data

Safety Data , Alarms & Notifications

<u>Data Flows</u>

- Alarms and Notifications
- Safety Data
- Realtime Control Data
- Process and Deployment
- Configuration Management
- Logging

Demonstration