

Features of Software Development for Robots

CSCI 420-02 Robotics



WILLIAM & MARY

CHARTERED 1693

Software Specification: Physical

- State Properties:

- Acceleration $< 4.9 \frac{m}{s^2}$
- Altitude $< 35,000$ ft
- Battery $> 80\%$ charge
- Calibrated $==$ True
- Located within Washington, D.C.

Properties can include
physical terms
and/or relate to the
physical world



Software Specification: Dependent

- Conditional State Properties

- If descending, landing gear out
- If turning, speed $< 3 \frac{m}{s}$
- If raining, headlights must be on
- If downhill, low gear must be engaged

Properties can depend on world and system state

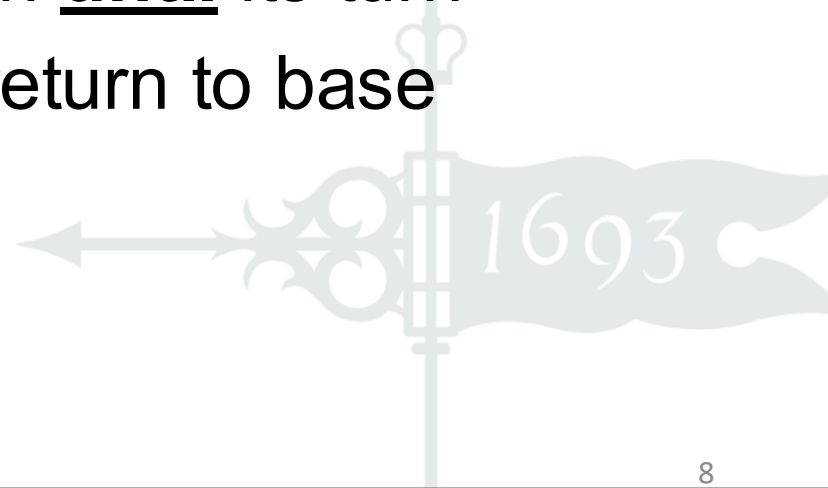
Software Specification: Timeliness

- Timeliness properties
 - (not robot) The cabin crew must be capable of evacuating all passengers in 90 seconds.
 - Brake must engage within 20ms of signal
 - Watchdog timer must run at 50Hz

Ability to respond may be
impacted by physical world

Software Specification: Temporal

- Temporal Properties
 - Engine must ignite **before** launch
 - Car must wait at stop sign **until** its turn
 - Drone must **eventually** return to base



Robot SW Architecture

- Asynchronous
- Loosely Coupled
- Abstracted
- Closed-loop

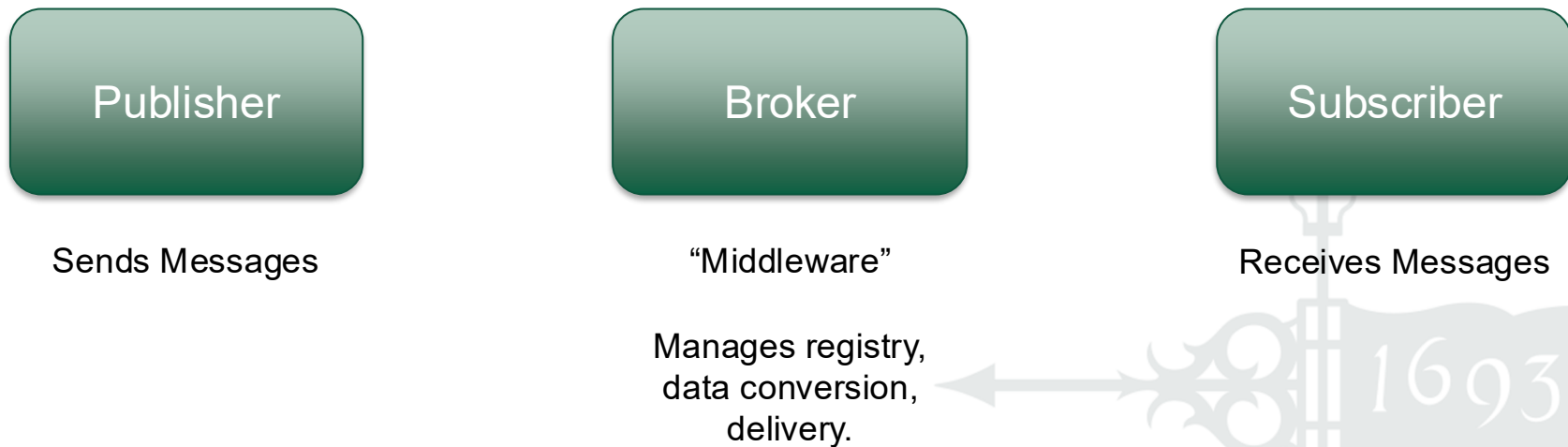


Robot SW Architecture

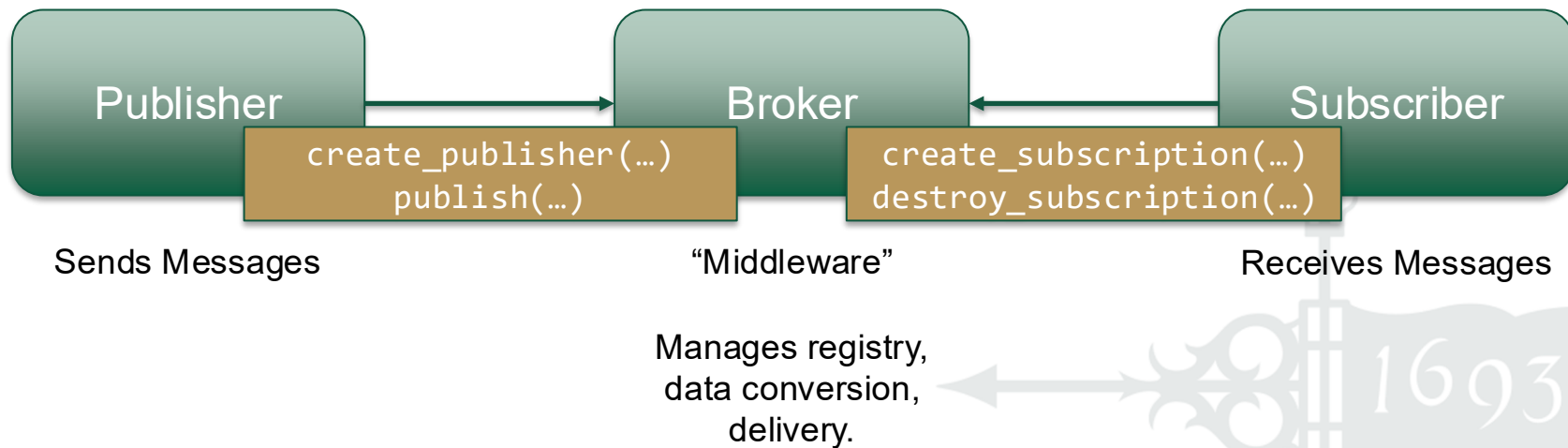
- Asynchronous – Event Driven
- Loosely Coupled – Parallelization, reuse
- Abstracted – Manage Complexity
- Closed-loop – Respond to Change



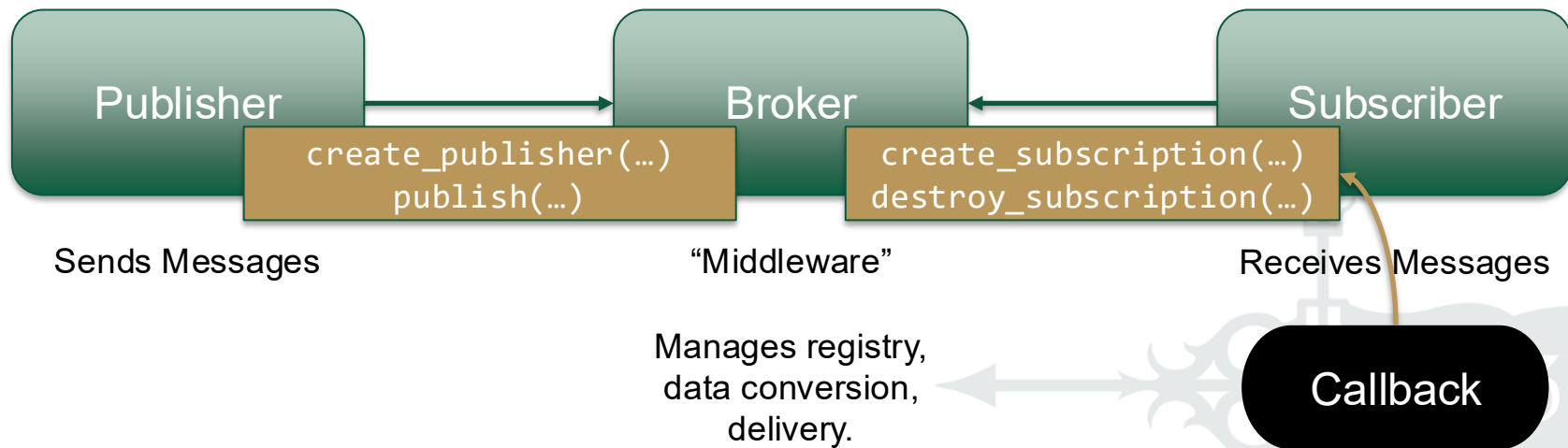
Event-Driven Comms: Pub/Sub



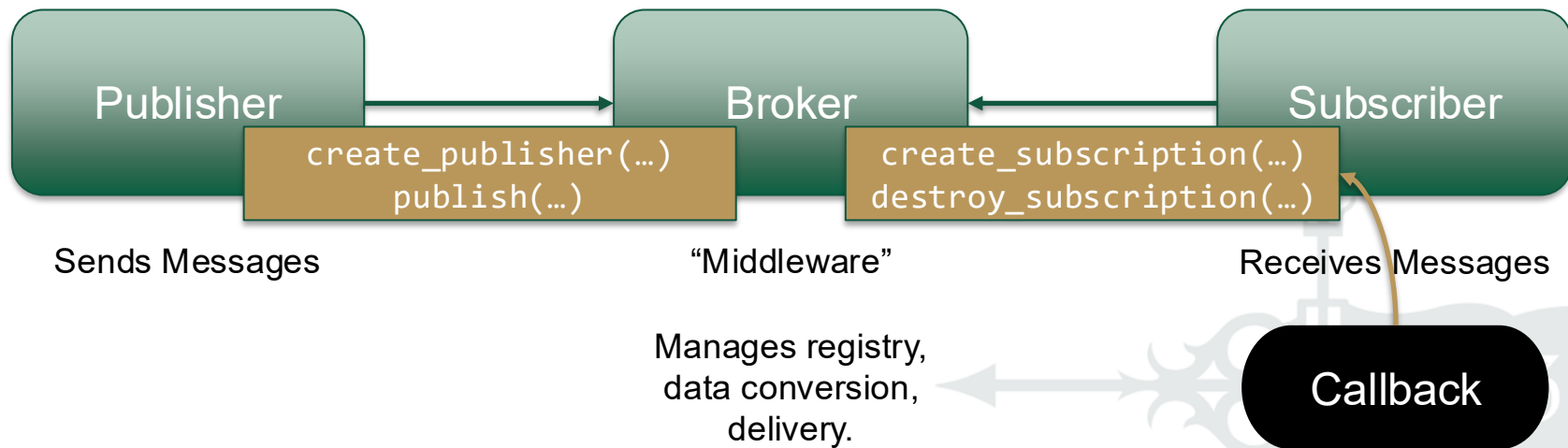
Event-Driven Comms: Pub/Sub



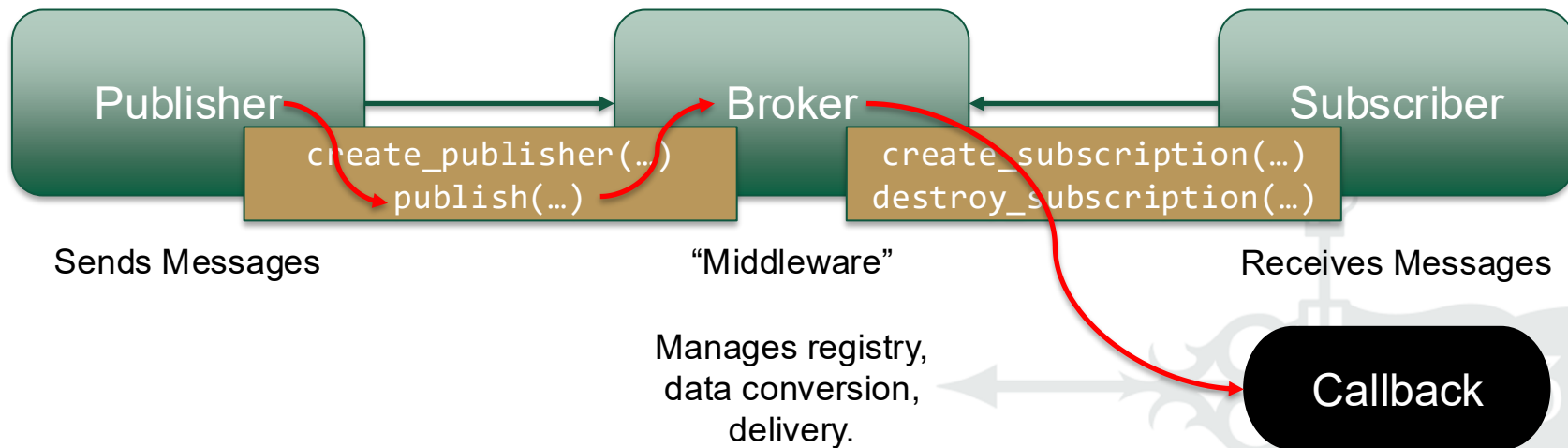
Event-Driven Comms: Pub/Sub



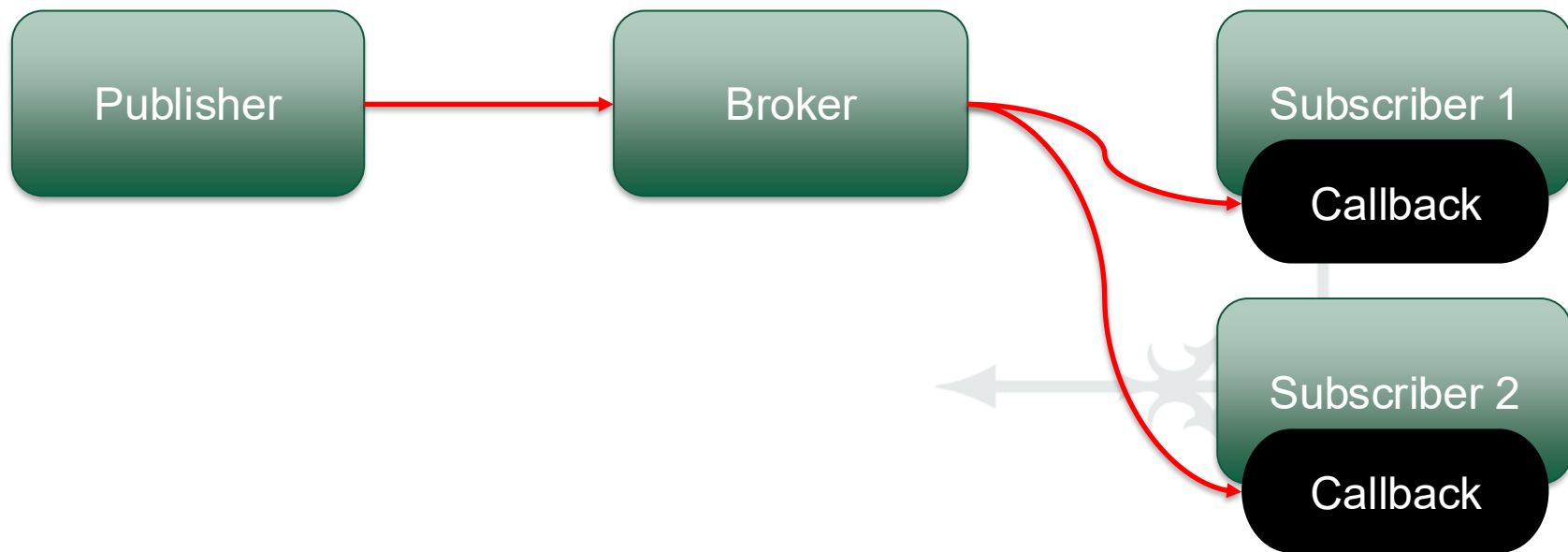
Event-Driven Comms: Pub/Sub



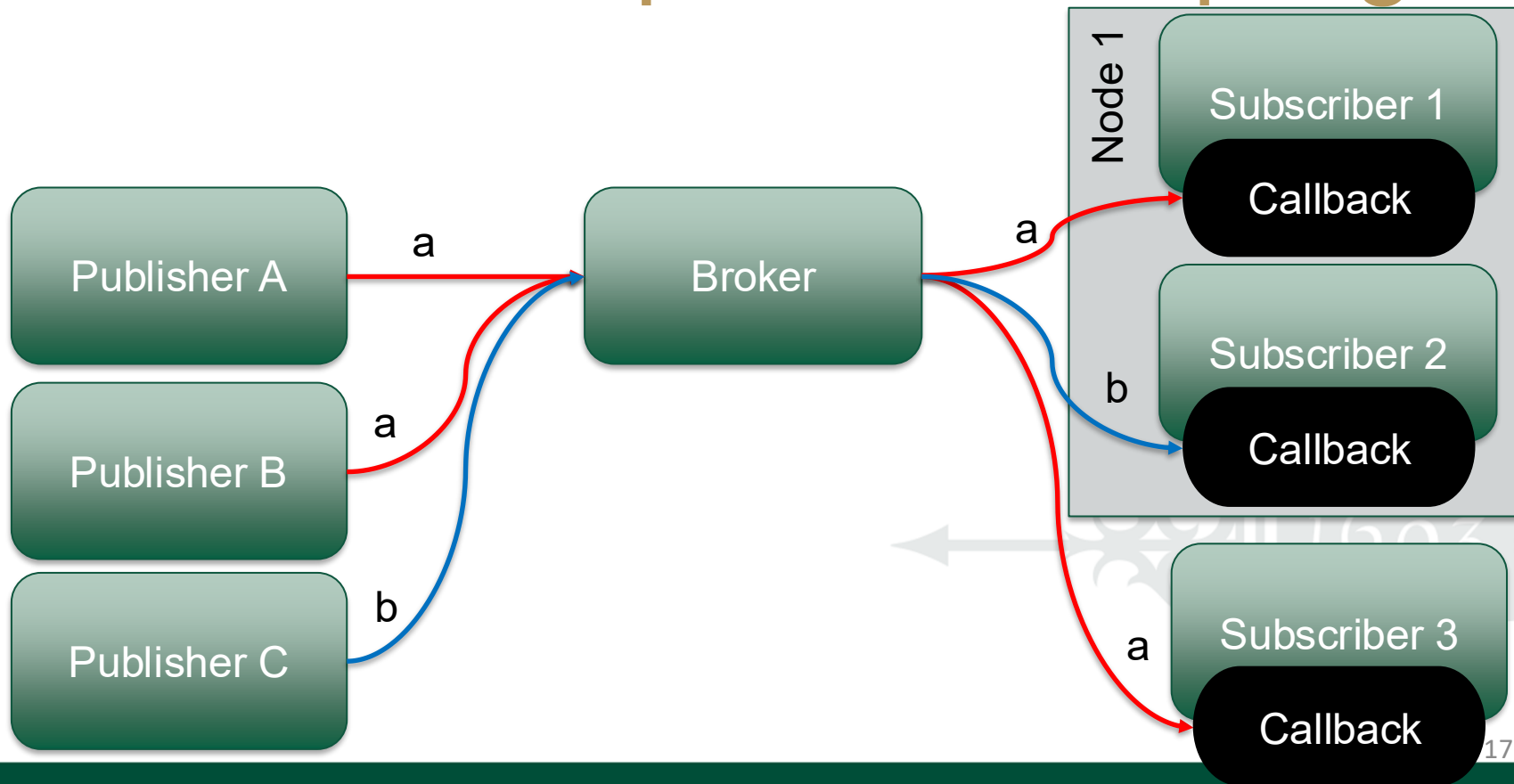
Event-Driven Comms: Pub/Sub



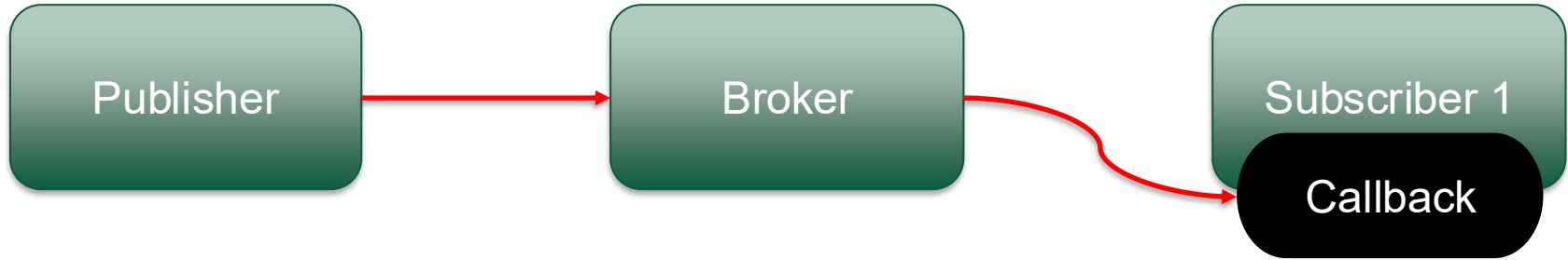
Pub/Sub: Space Decoupling



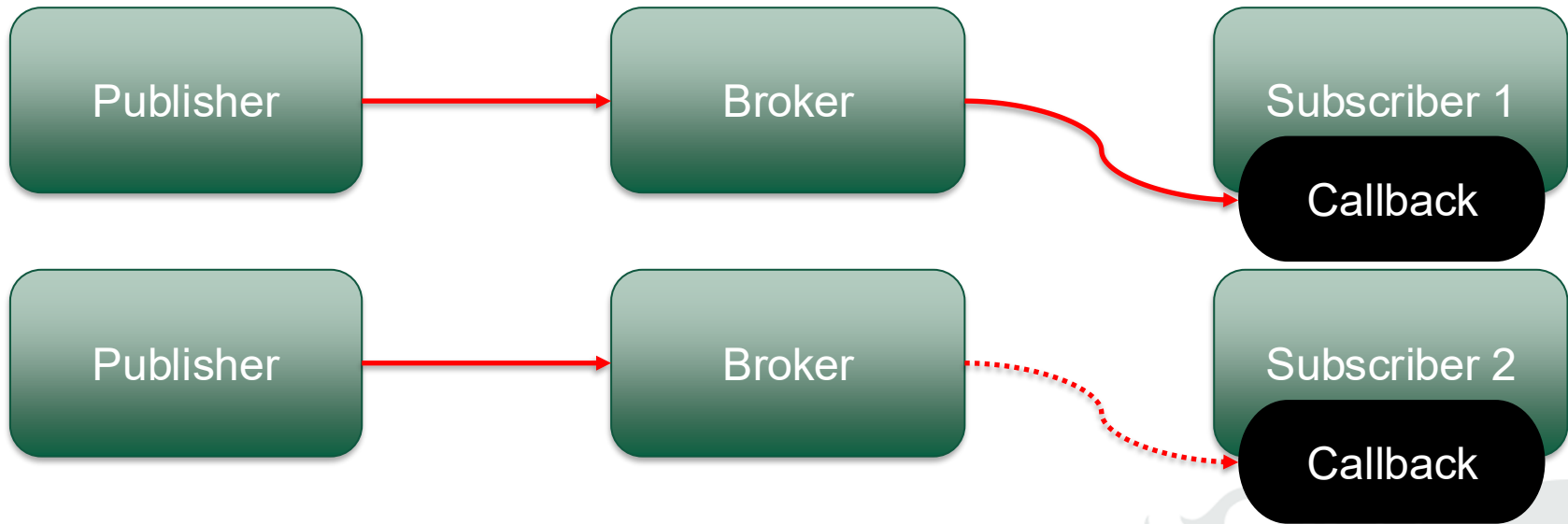
Pub/Sub: Space Decoupling



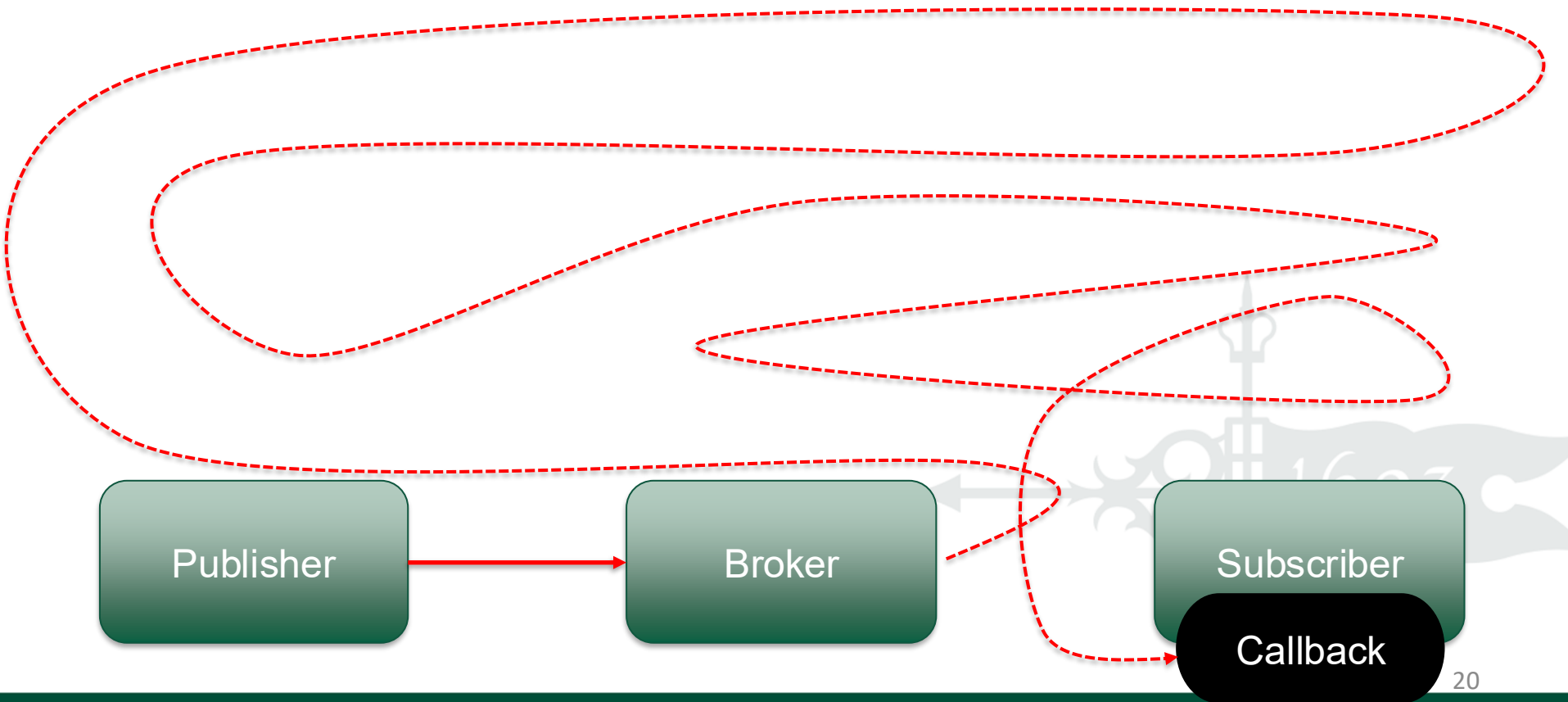
Pub/Sub: Time Decoupling



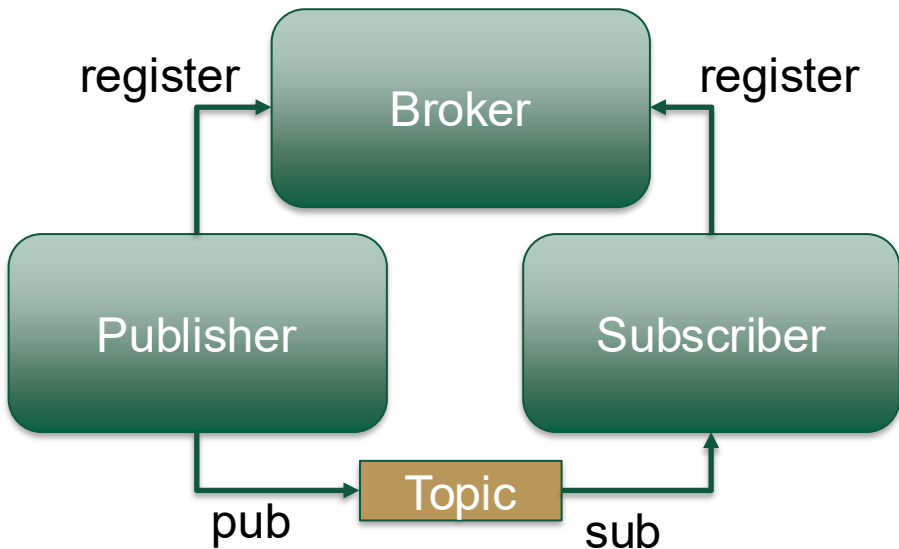
Pub/Sub: Time Decoupling



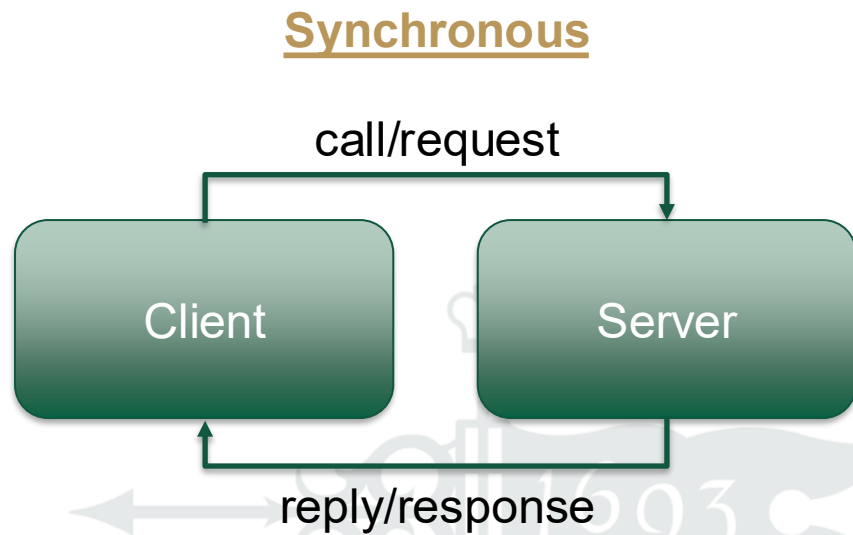
Pub/Sub: Time Decoupling



Pub/Sub vs Client Server



Asynchronous



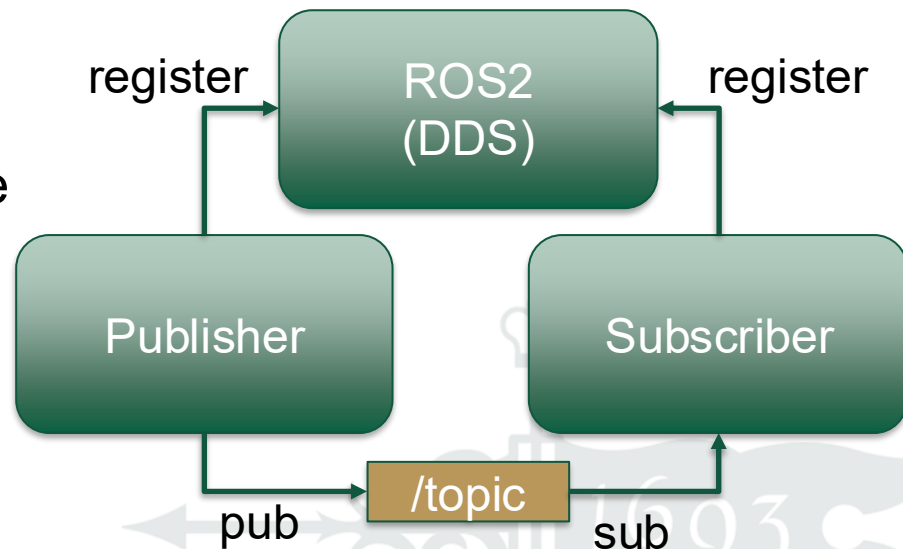
ROS also allows non-blocking synchronous calls

Pub/Sub Functionality

- Filtering
 - Topic-based
 - Content-based
- Message Delivery
 - How to physically transmit the message
 - Unicast, Multicast, push/pull

Pub/Sub in ROS2

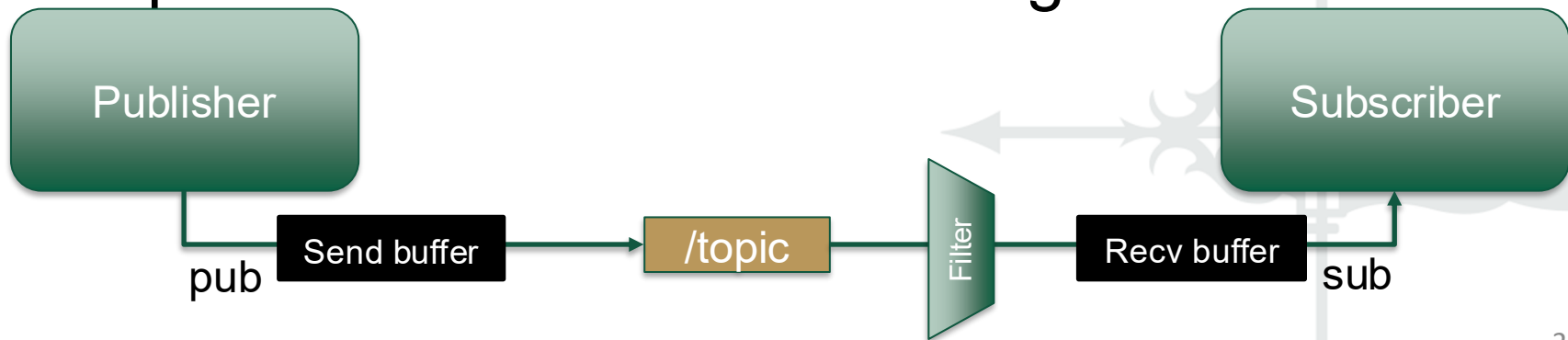
- ROS2 connects publishers and subscribers
 - Uses Data Distribution Service (DDS) middleware
- Nodes can be publishers and/or subscribers to unlimited topics



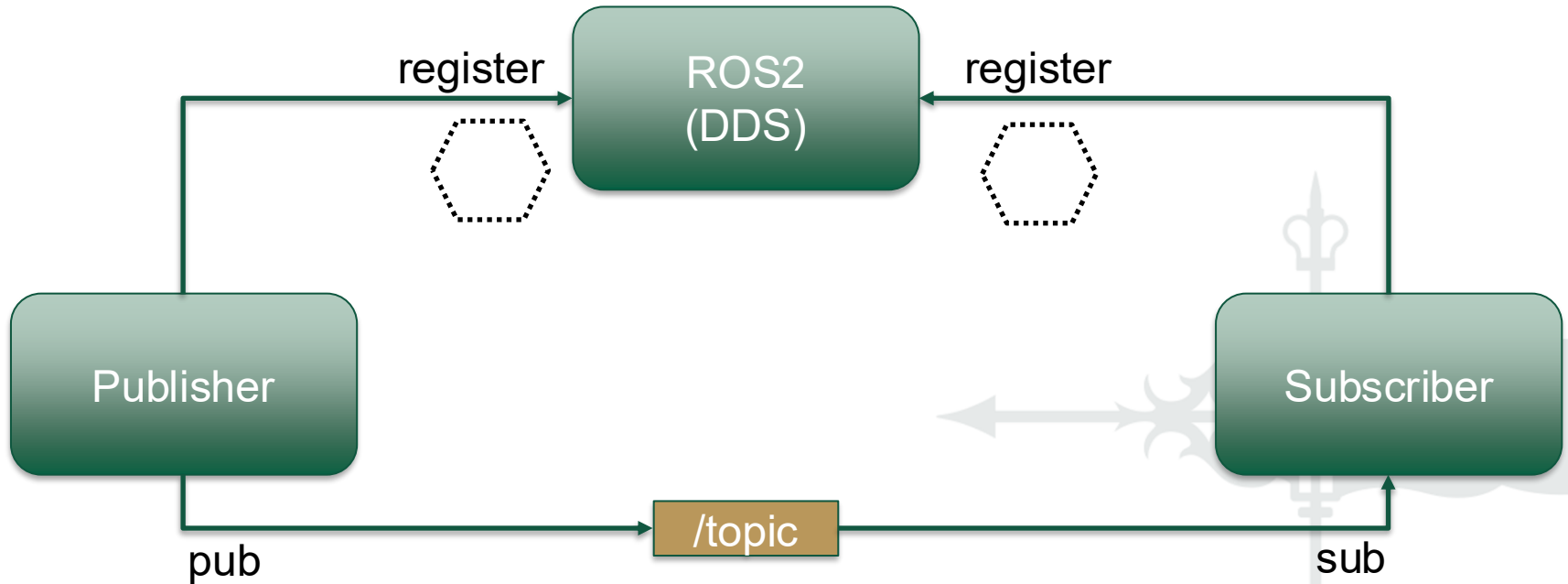
ROS also allows client-server and action paradigms

ROS2 Topic Communication

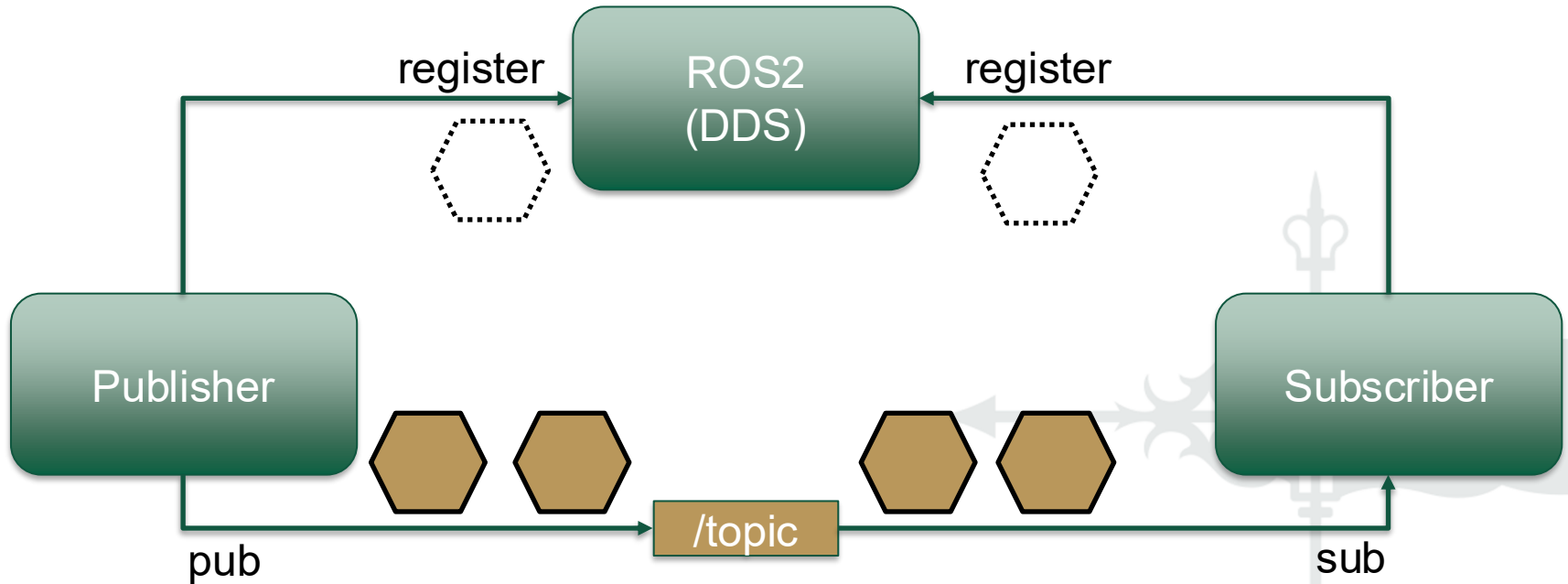
- All pub/sub uses topic-based filtering
- Buffering
- Peer-to-peer
- Optional: content-based filtering



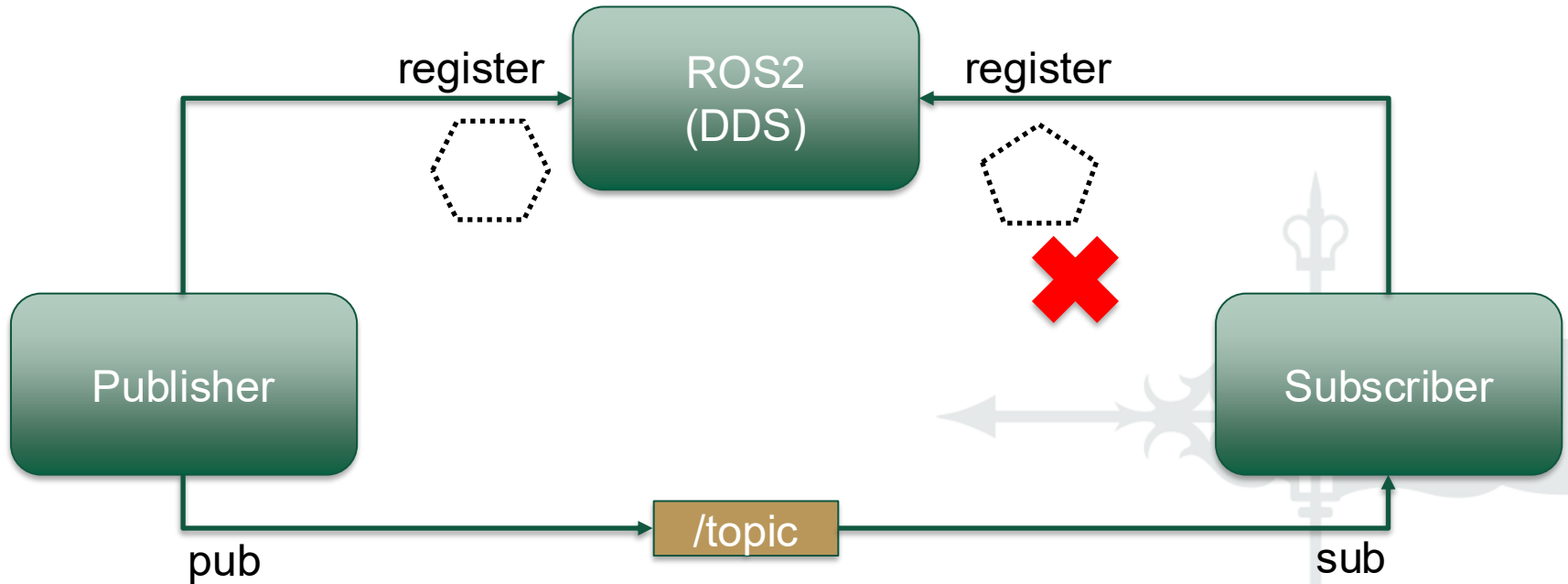
ROS2 Topic Message Types



ROS2 Topic Message Types



ROS2 Topic Message Types



ROS2 Topic Message Types

Geometry Msgs

[Accel](#)
[AccelStamped](#)
[AccelWithCovariance](#)
[AccelWithCovarianceStamped](#)
[Inertia](#)
[InertiaStamped](#)

Point

[Point32](#)
[PointStamped](#)
[Polygon](#)
[PolygonInstance](#)
[PolygonInstanceStamped](#)
[PolygonStamped](#)

[Pose](#)
[Pose2D](#)
[PoseArray](#)

PoseStamped

[PoseWithCovariance](#)
[PoseWithCovarianceStamped](#)
[Quaternion](#)
[QuaternionStamped](#)
[Transform](#)
[TransformStamped](#)

[Twist](#)
[TwistStamped](#)
[TwistWithCovariance](#)
[TwistWithCovarianceStamped](#)

Vector3

[Vector3Stamped](#)
[VelocityStamped](#)
[Wrench](#)
[WrenchStamped](#)

Standard Msgs

Bool

[Byte](#)
[Char](#)
[Float32](#)

Float64

[Int8](#)
[Int16](#)
[Int32](#)
[Int64](#)

String

[UInt8](#)
[UInt16](#)
[UInt32](#)
[UInt64](#)
[ColorRGBA](#)
[Empty](#)

Header

[ByteMultiArray](#)
[Float32MultiArray](#)
[Float64MultiArray](#)
[Int8MultiArray](#)
[Int16MultiArray](#)
[Int32MultiArray](#)
[Int64MultiArray](#)
[MultiArrayDimension](#)
[MultiArrayLayout](#)
[UInt16MultiArray](#)
[UInt32MultiArray](#)
[UInt64MultiArray](#)
[UInt8MultiArray](#)

Many Msg Packages

Sensor Msgs

- [BatteryState](#)
- [Image](#)
- [LaserScan](#)
- [PointCloud](#)
- [Temperature](#)

Simple Navigation

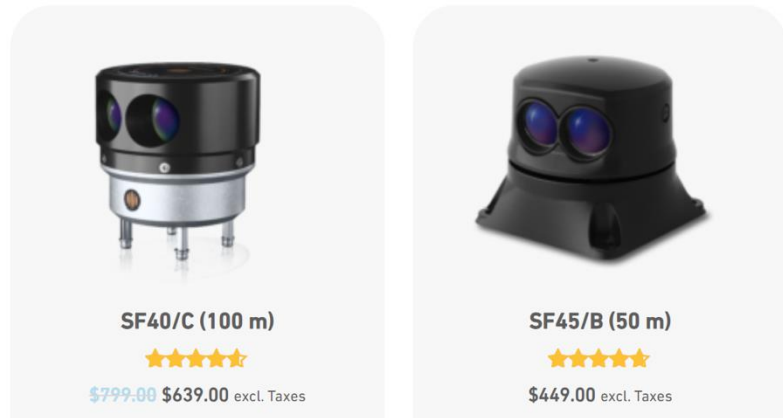
- [OccupancyGrid](#)

Advanced Navigation

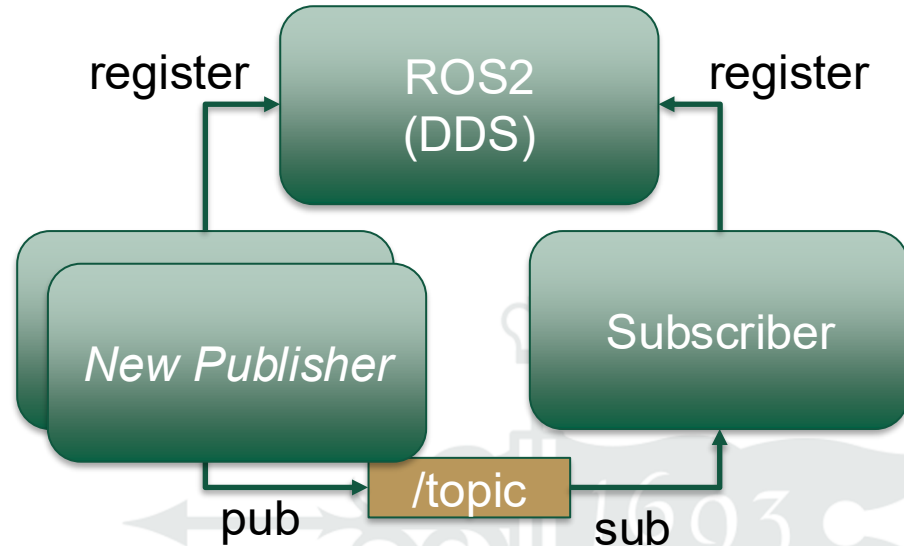
- [VoxelGrid](#)

Support custom types

Pub/Sub Modularity



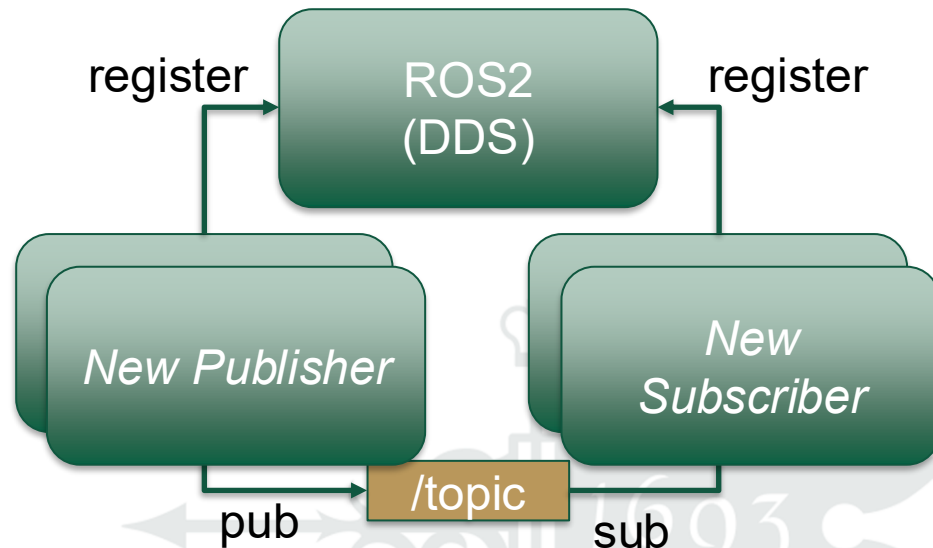
<https://lightwarelidar.com/product-tag/scanning>



Can replace pub/sub as new modules are available

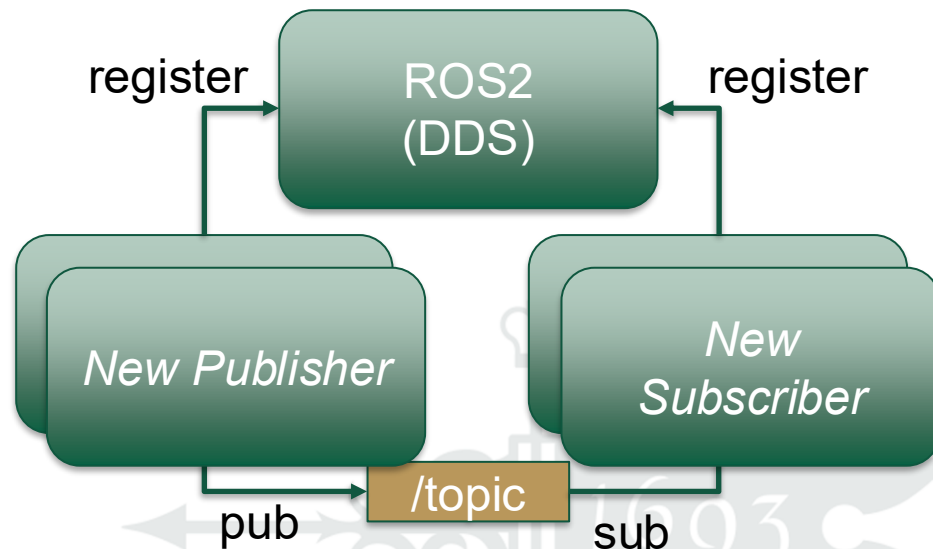
Leaky Abstractions

- What could leak?



Leaky Abstractions

- What could leak?
 - Improper Msg Use
 - No time guarantees
 - Lost messages
 - Buffer size matters?
- New Module:
 - Hyrum's Law

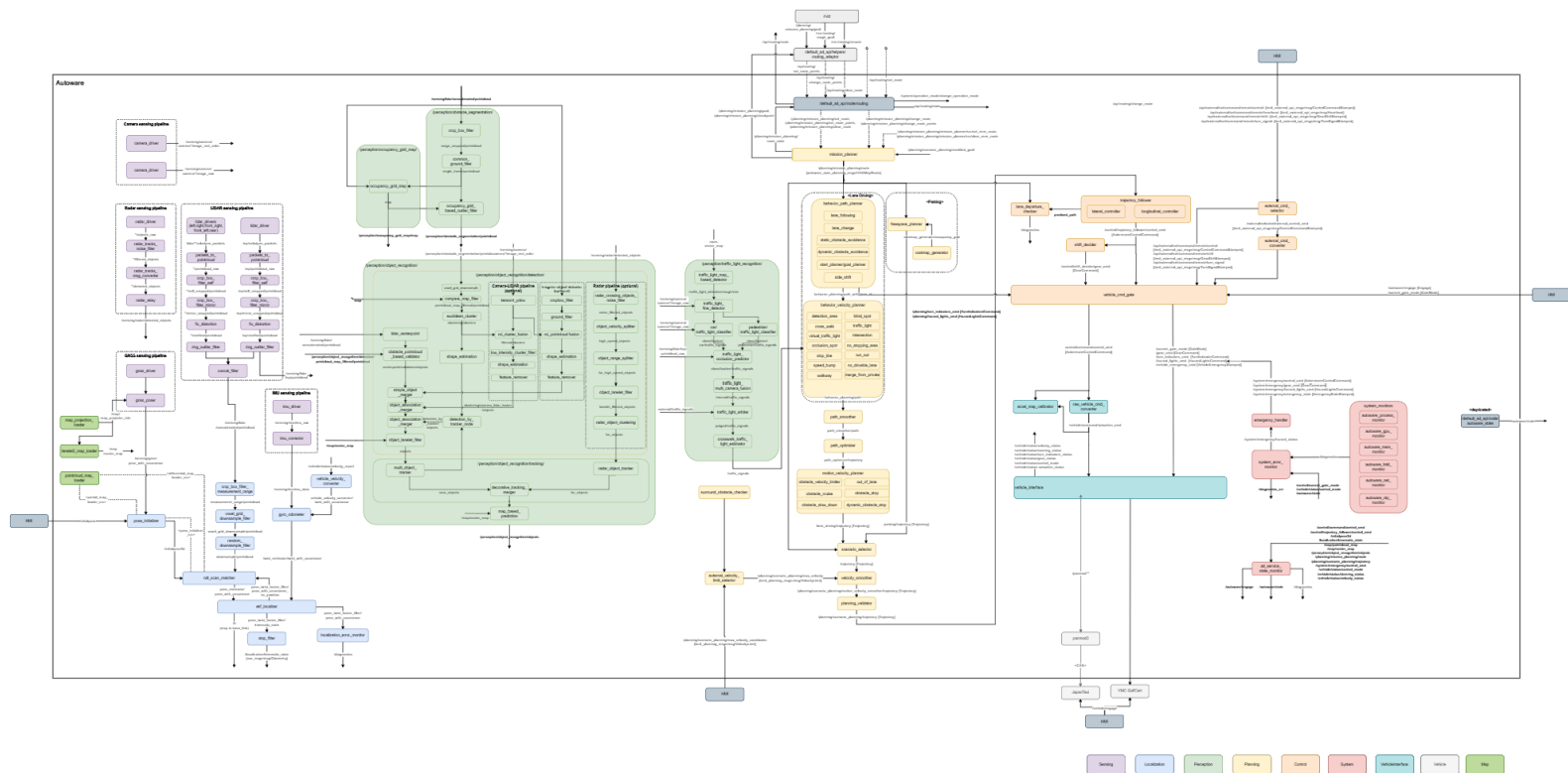


ROS2 – Real Systems!

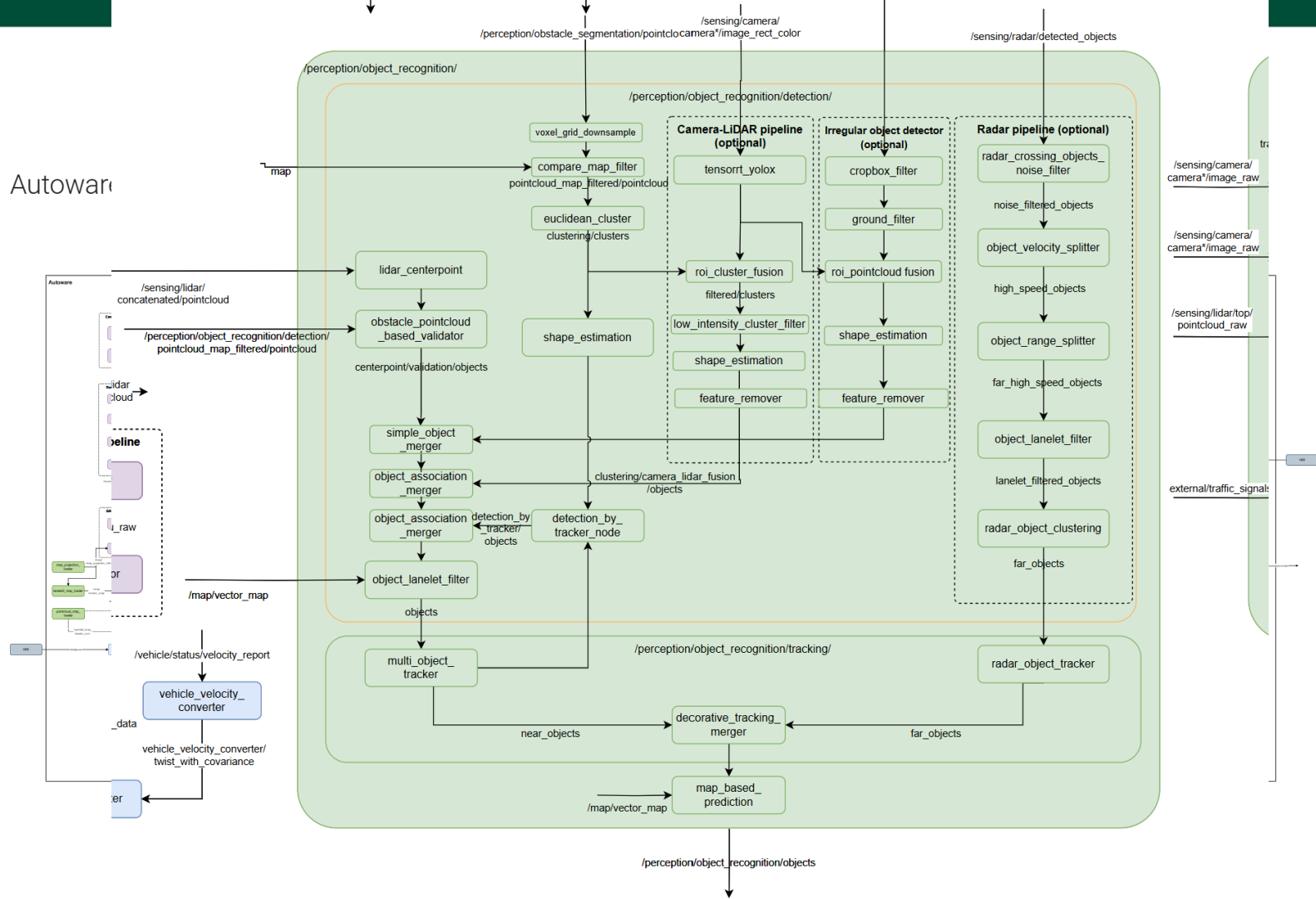
Autoware is the world's leading open-source project dedicated to **autonomous driving** technology. Built on the Robot Operating System (**ROS 2**), Autoware facilitates the **commercial deployment** of autonomous vehicles across various platforms and applications.

ROS2 – Real Systems!

Autoware Universe



Autoware



Core Robot SW Implications

- Development is complex and modular
 - Integration is key!
 - Abstractions are a must, but leaky!
- Asynchronous, event-driven, loosely coupled
 - Publish-Subscribe
 - Decentralized
 - Message Types

