DDS Connector

Sept, 2016

Gianpiero Napoli, Senior Software Engineer

Agenda

- Goals
- Concept
- Architecture
- Status
- Demo
- Next Steps

Connector Goals

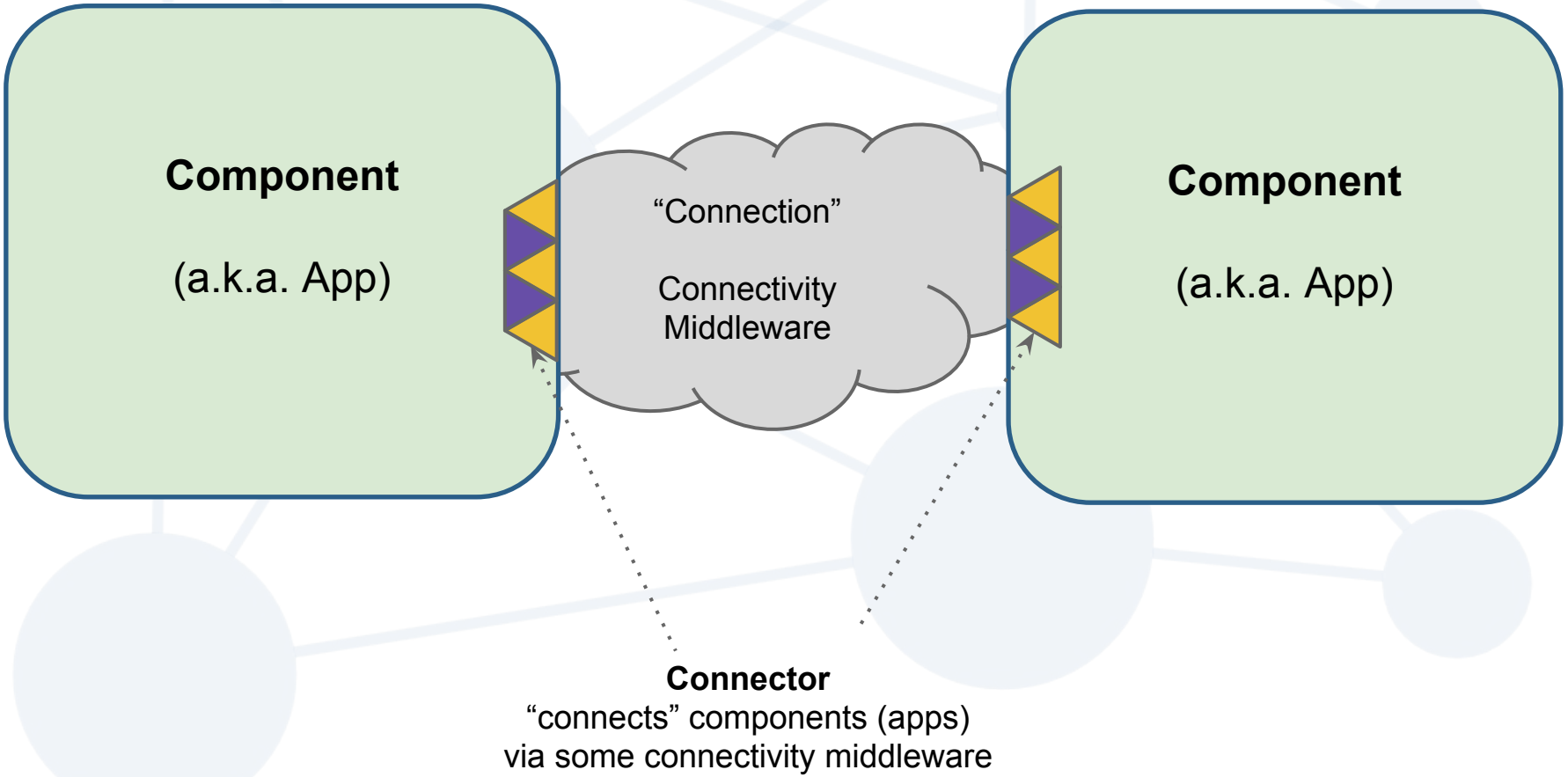
- Faster ("Rapid")
  - Development
  - Integration
  - Troubleshooting
- Integrate
  - Connectivity technologies
  - Scripting
- Simplify
  - API, Configuration, Options
- Preserve

  Data-Oriented Design|Architecture

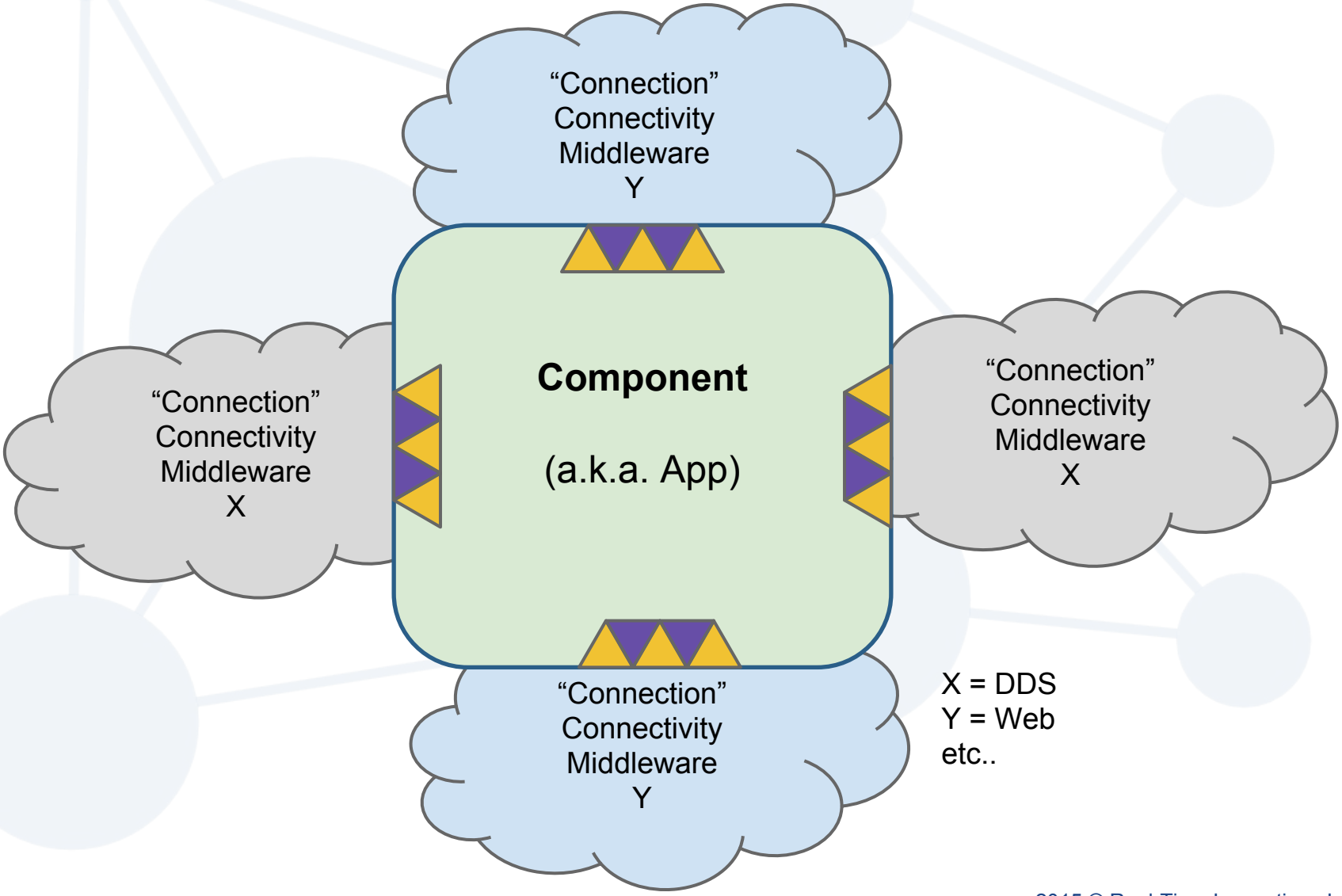
The Connector Model

What?

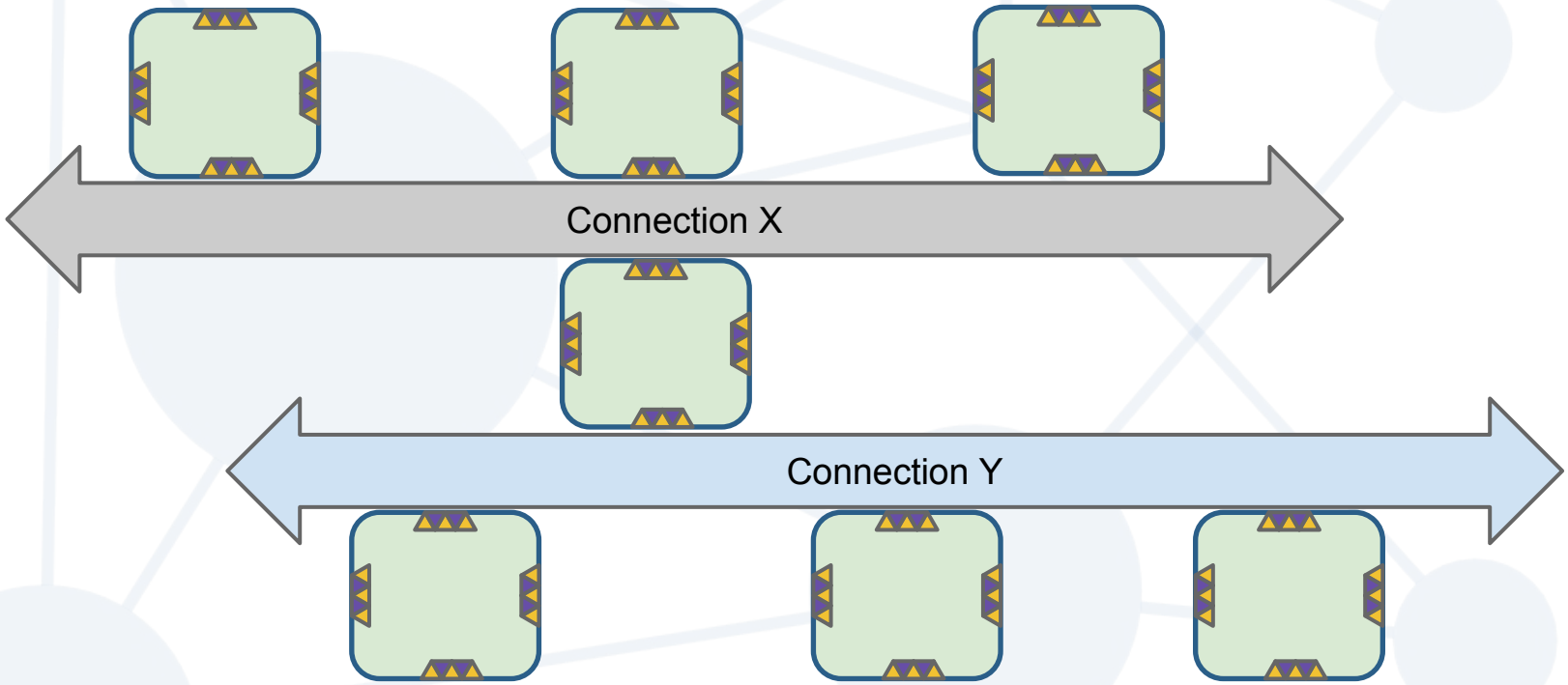
Connector Concept



Connector Facilitates Integration



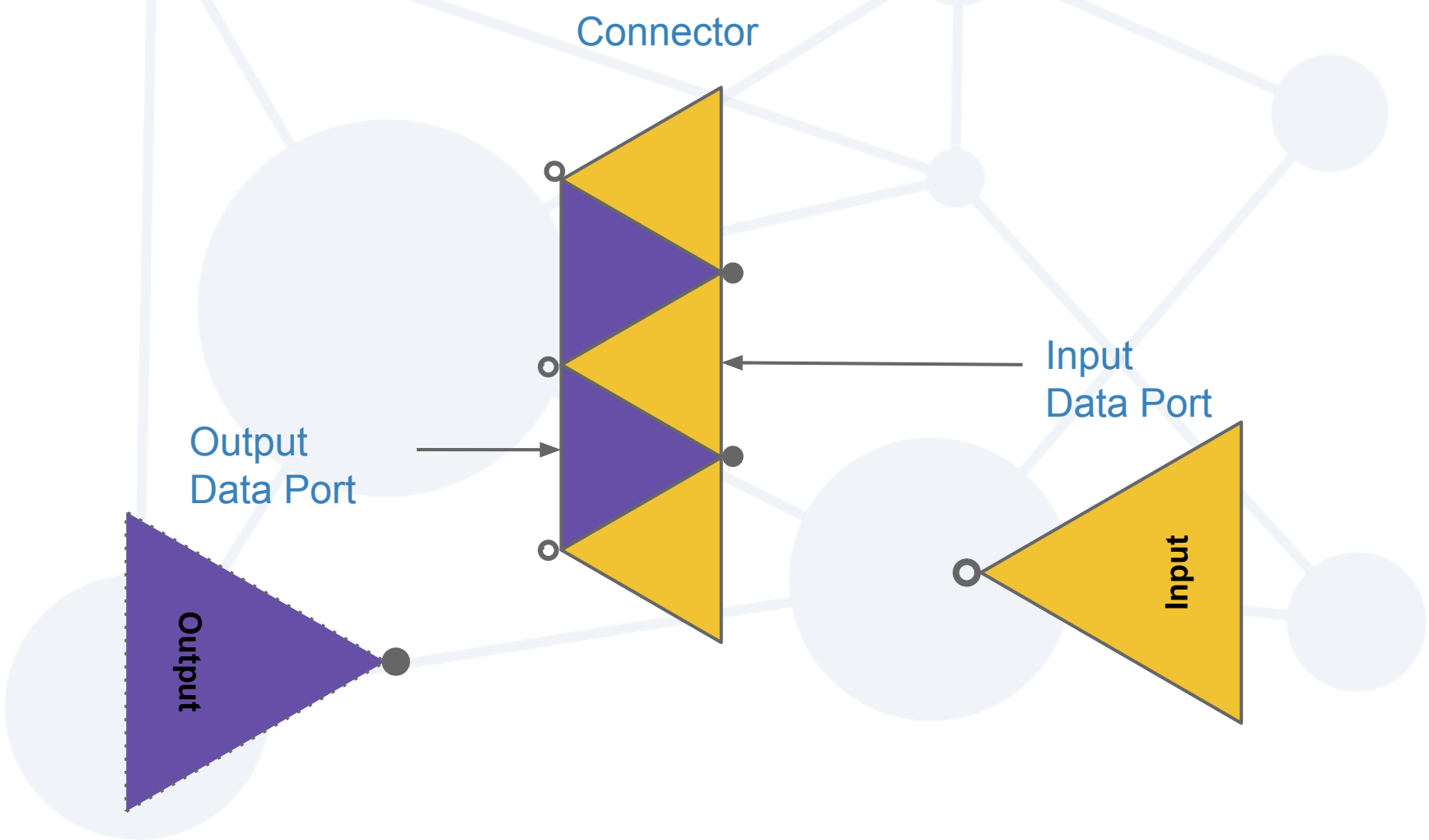
Connectors Speed Up Integration



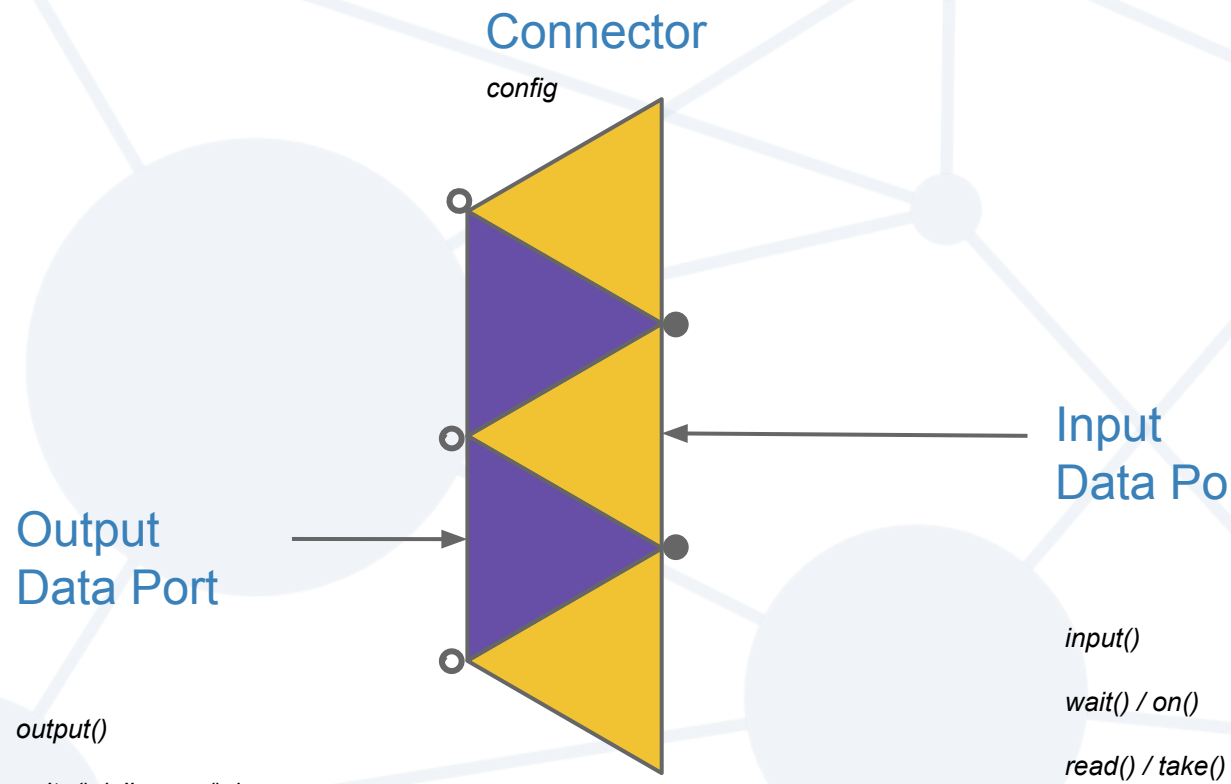
Connector

- connects your app to a data space (DDS)
- exposes a data-centric API
- lets your app read/write structured data

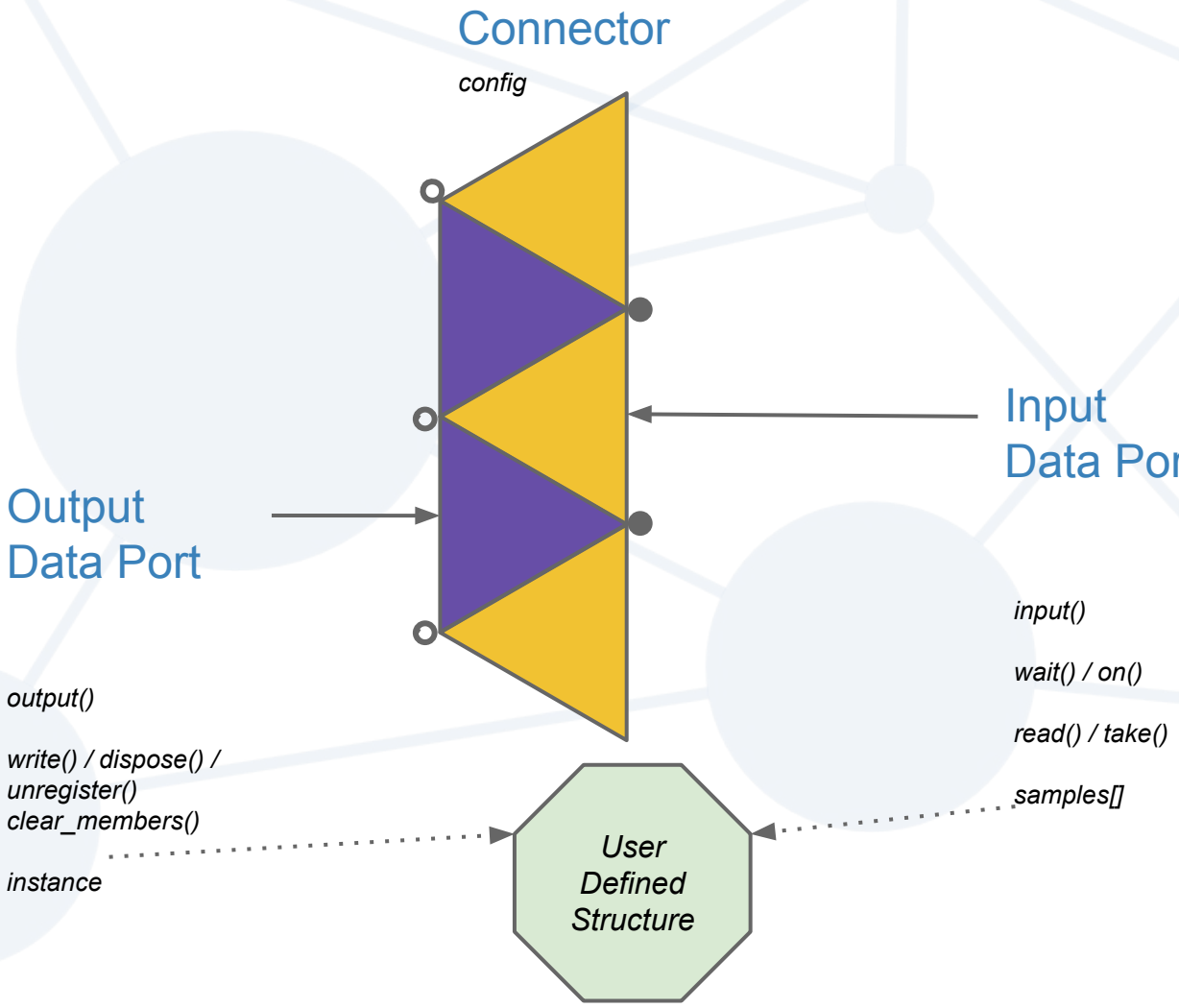
A Connector has Input & Output ports



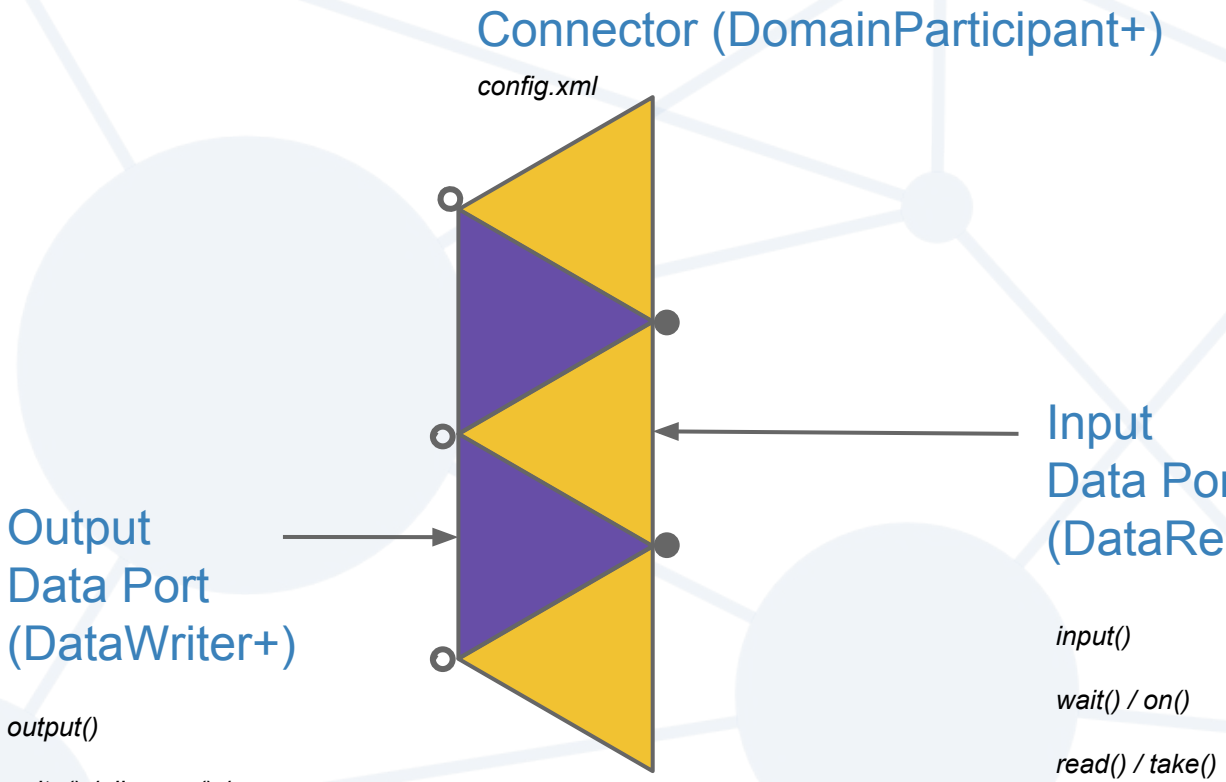
Connectors have a data-centric API



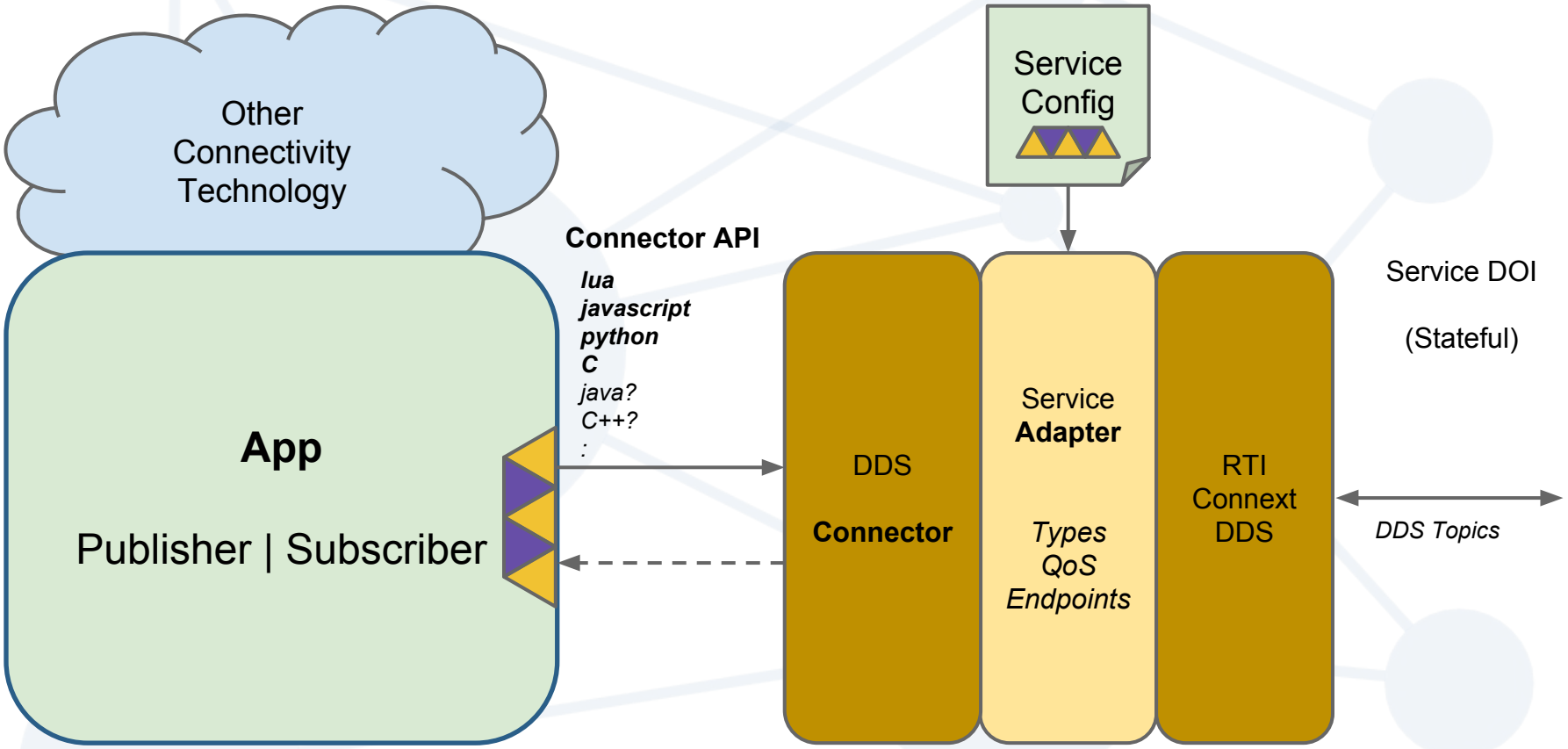
Connectors have a data-centric API



DDS Connector Input & Output ports

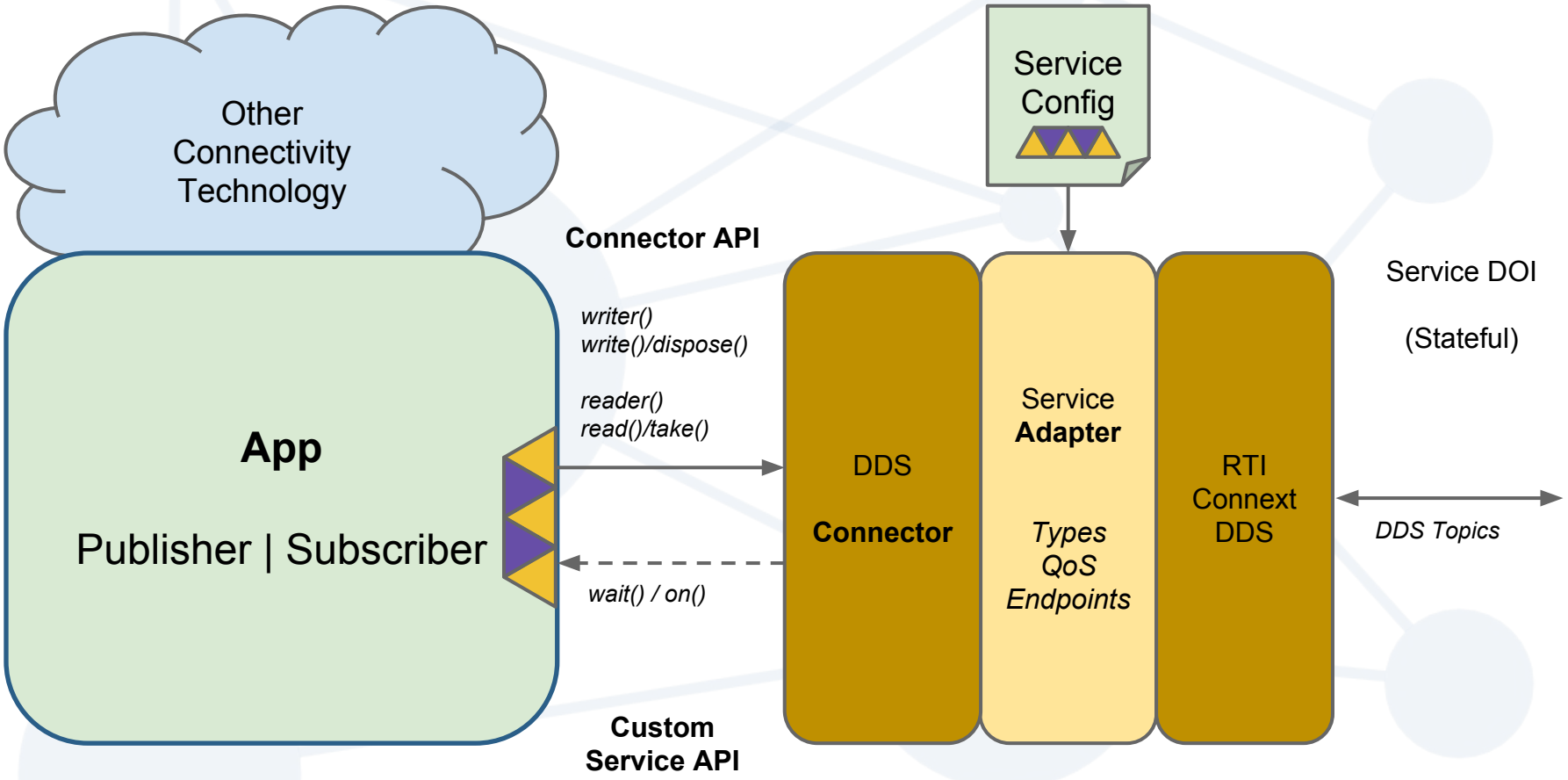


DDS Connector



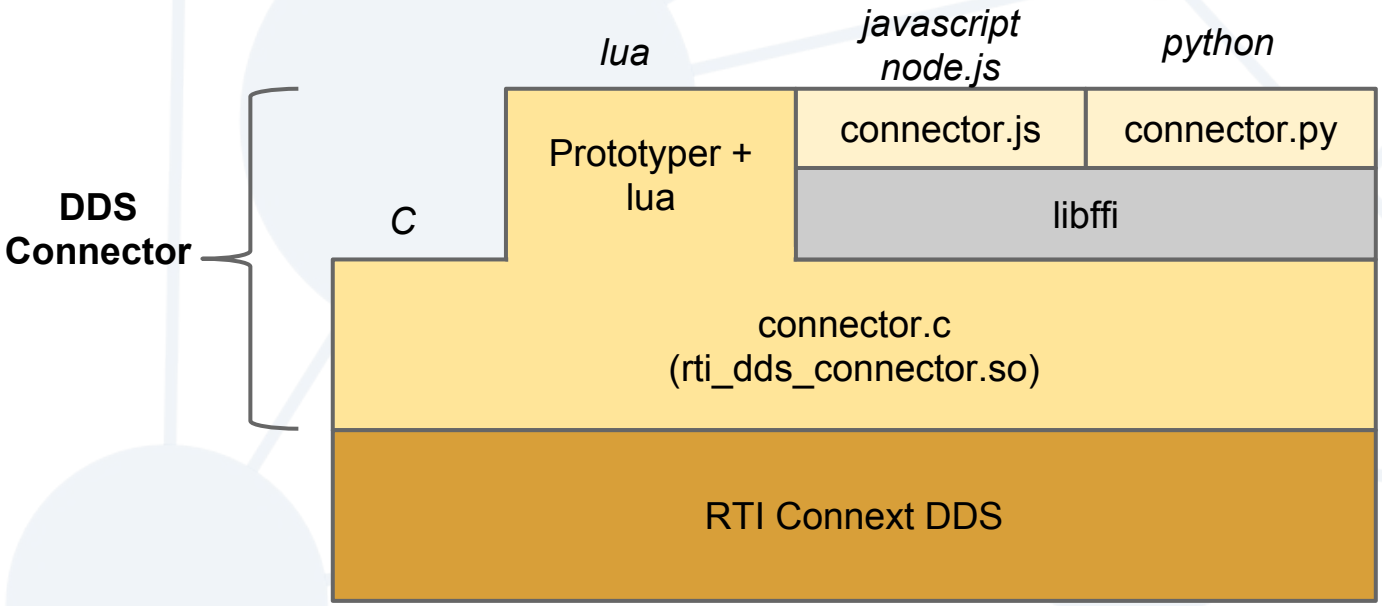
API = Application Programming Interface
DOI = Data-Oriented Interface

DDS Connector

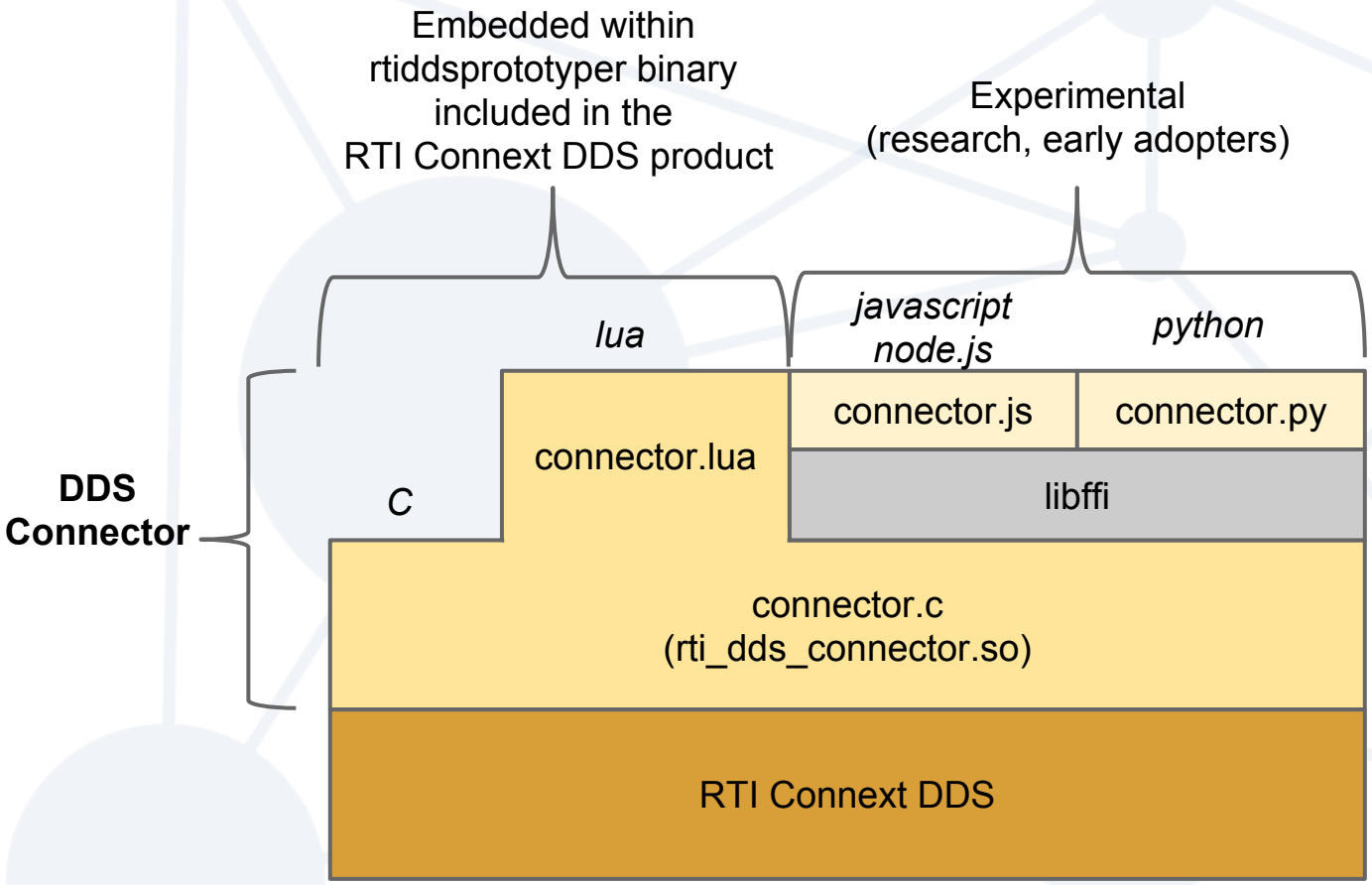


API = Application Programming Interface
DOI = Data-Oriented Interface

DDS Connector Software Stack

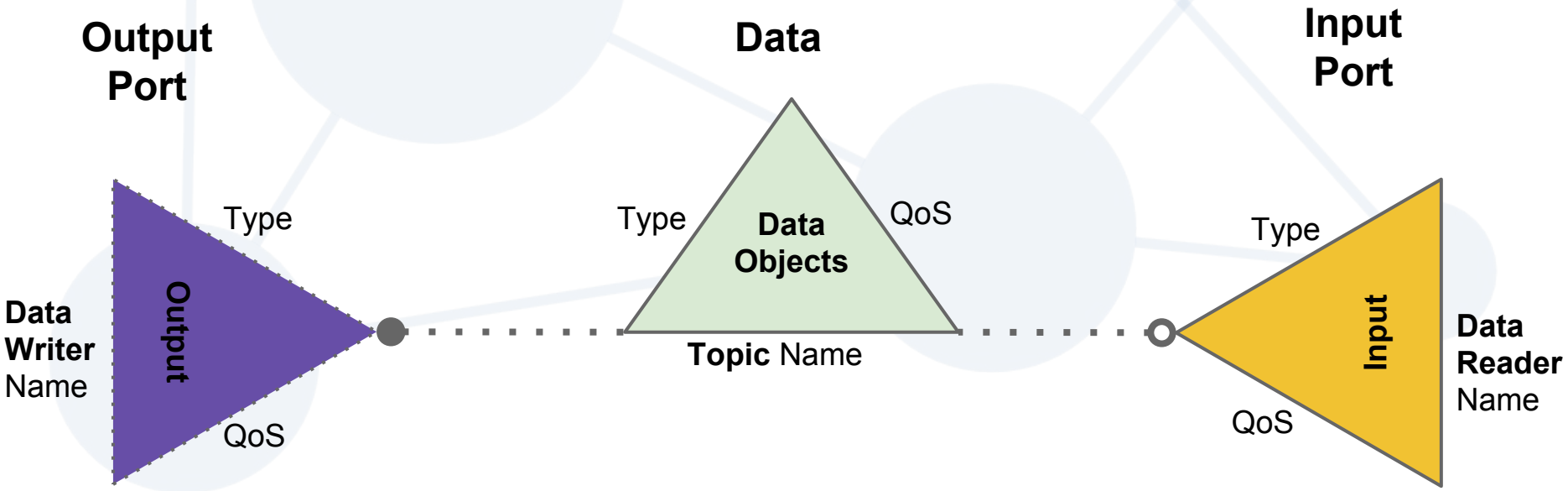


Current Status



DDS Connector Configuration

- XML Based App Creation
- Participant
  - Types
  - QoS
  - Endpoints



Using a Connector: Pseudo Code

Creating a Connector

```
connector = new Connector("participant_name", "config.xml")
```

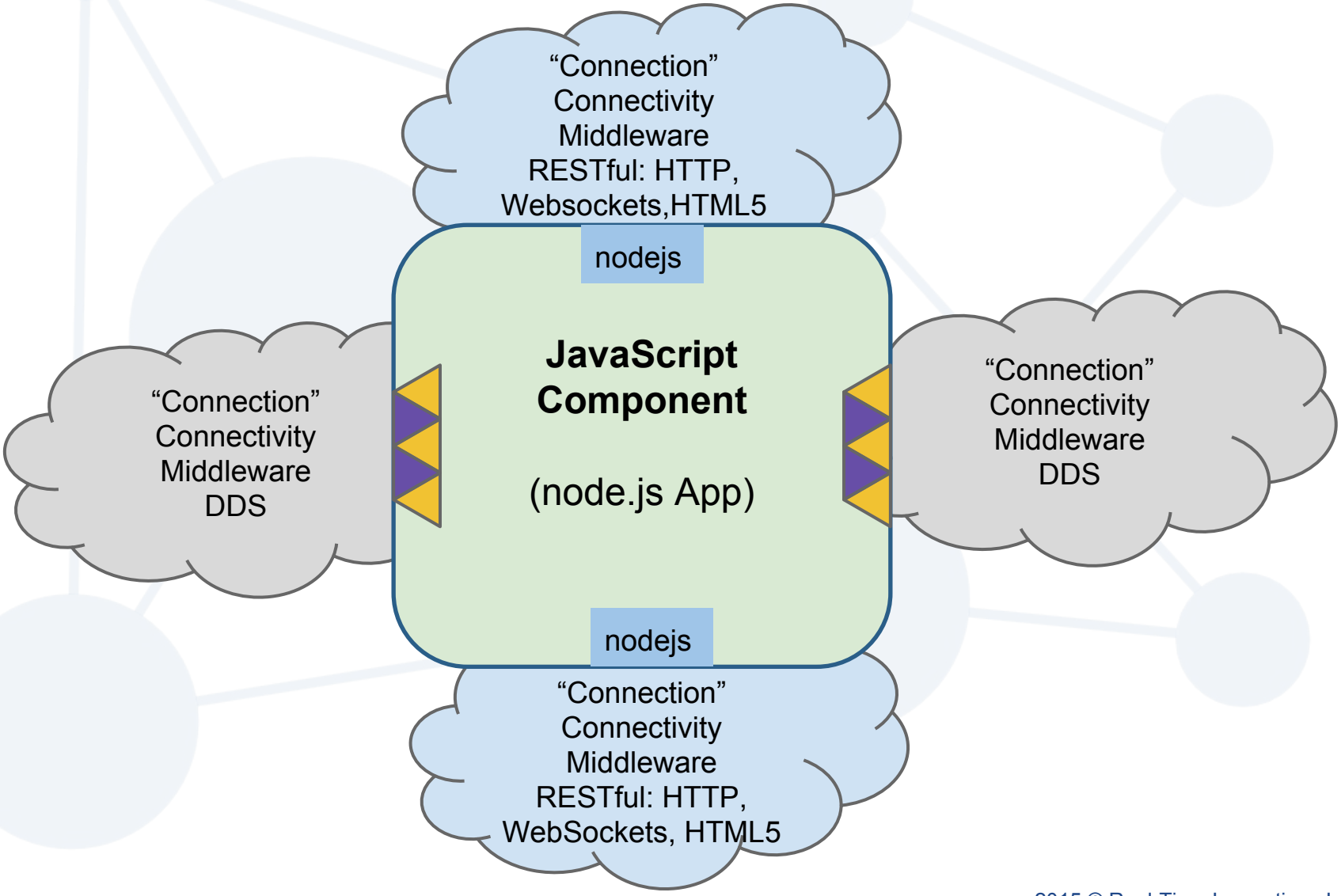
Reading Data

```
input = connector.getInput("in_name")

input.take()
length = input.samples.getLength(input)
for (i = 1; i <= length; ++i) {
        if connector.infos(i).isValid then {
                print(connector.samples(i))
        }
}
```

Writing Data

```
output = connector.getOutput("out_name")

output.instance.setNumber("x", 10.0)
output.write()
```

Integrating DDS & Web Technologies



Demo

Common Use Cases

- Integration
  - Middleware technologies
  - DDS Domain Bridging
- I/O
  - Device / Hardware
  - Web APIs and Services
- UIs
  - GUI Toolkits
  - Web Based
- Information Processing
  - Use various processors, eg Rx?
  - Integrate with CEP engines?
- ...

Use Case: Integrating DDS and Web

- Standard Thin Client (without DDS)
  - Real-Time Web
    - push using websockets (using socket.io)
  - Browser, Mobile, Handheld
    - no special software installation
- Custom Thin Client (with DDS)
  - HTML5 GUI using nw.js
  - +Plus the capabilities of Real-Time Web
  - +Plus the capabilities of DDS (state management)

Connector is Experimental Product

- Download connector here: https://github.com/rticommunity/rticonnextdds-connector
- More info on connector here: https://community.rti.com/downloads/experimental/rticonnextdds-connector

Thank You!

Examples and code digging!

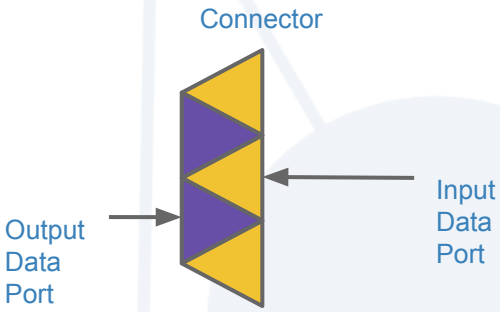
Getting Started

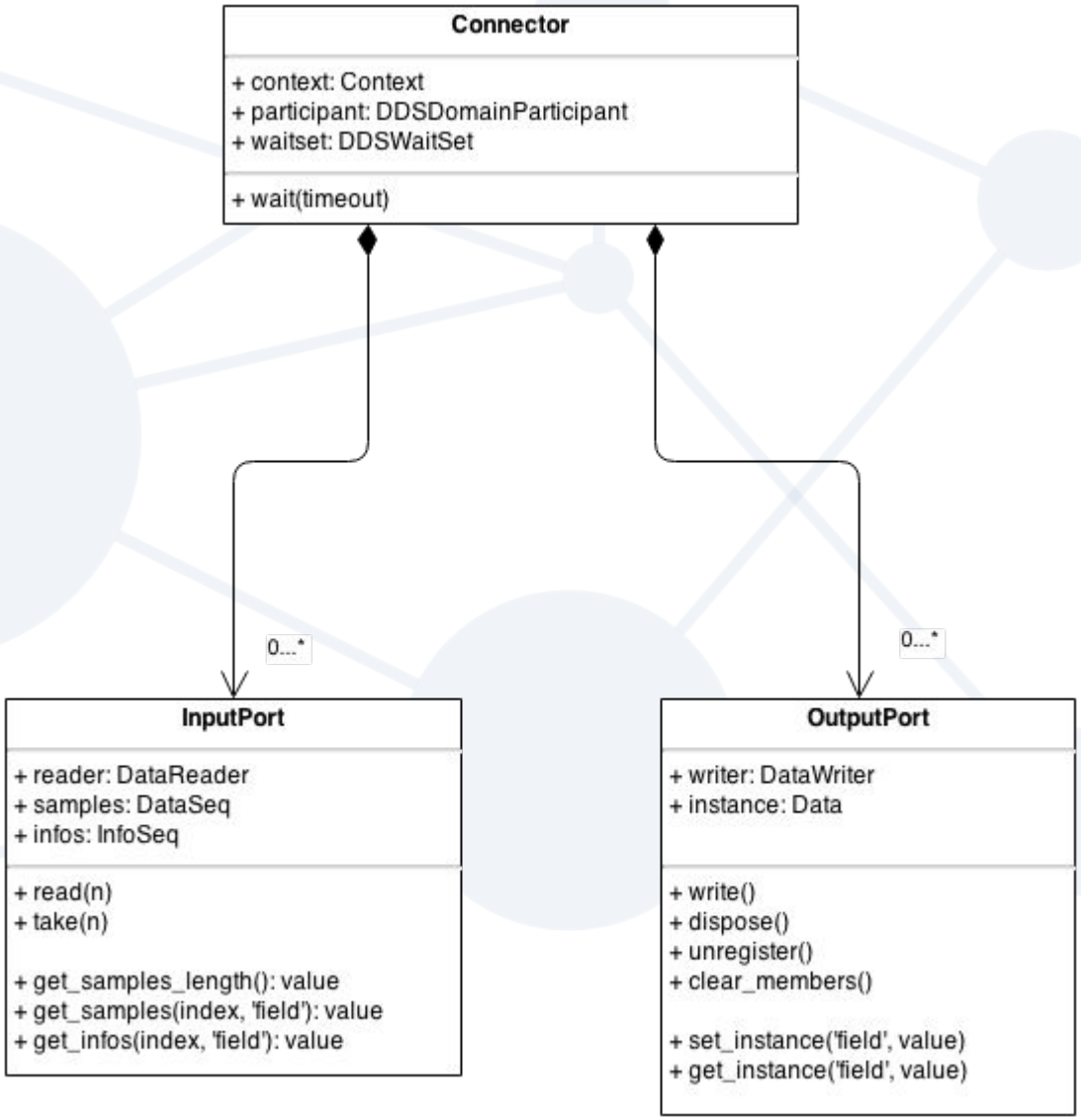
- simple
  - DDS ←→ DDS
    - poll
    - event
- web_http
  - DDS ←→ HTTP
    - poll
- web_socket
  - DDS ←→ WebSockets
  - HTML5: table, chart, plugin (Google Earth)

Data Indexing

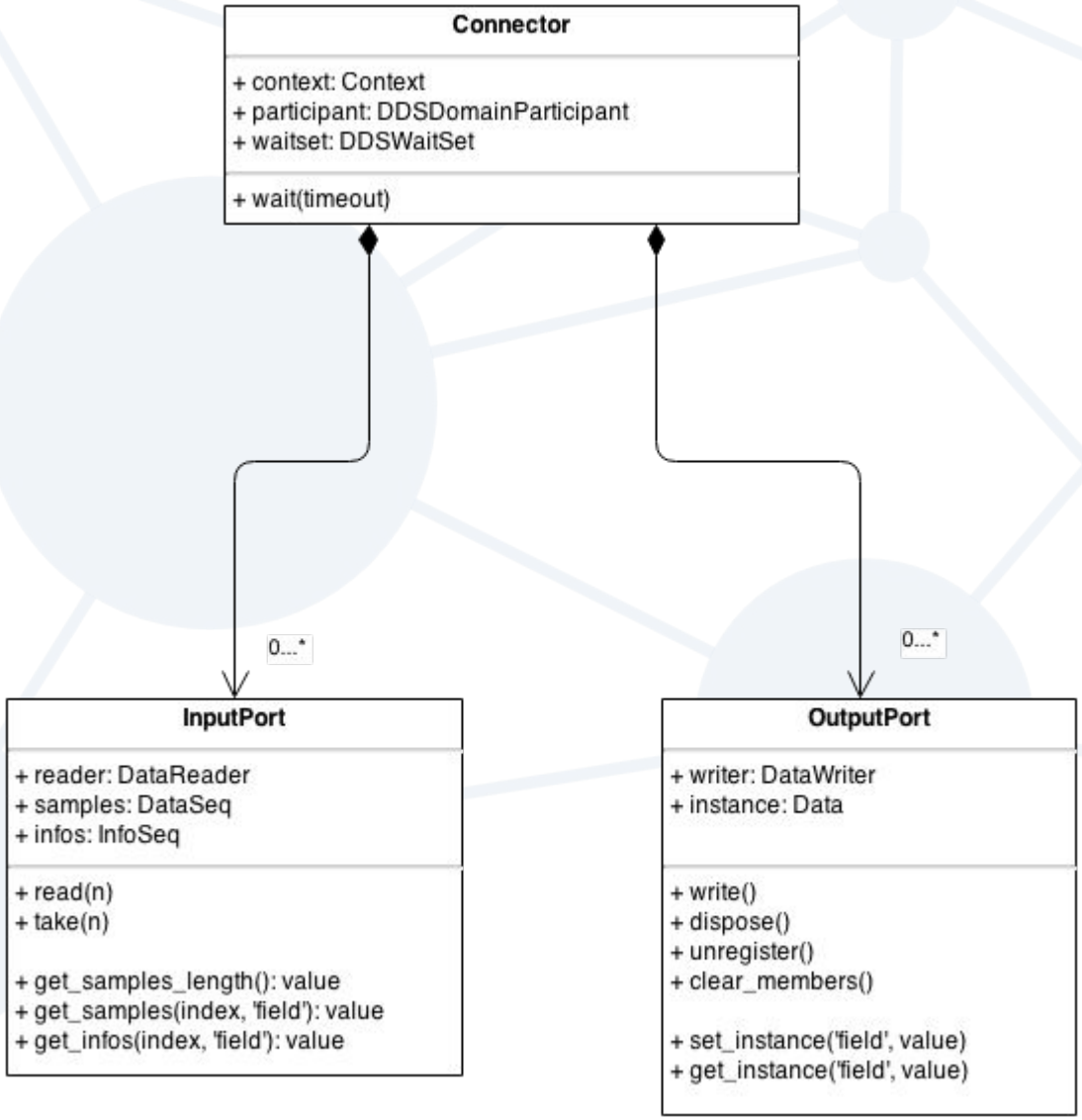
- Prototyper with Lua Getting Started Guide
  - Section 6.4
- NOTE
  - Prototyper = Singleton Connector + Execution Loop

DDS Connector API





DDS Connector API



DDS Connector for Lua - connector.lua

lua connector

DDS Connector for C - connector.c

C connector

DDS Connector for Python - connector.py

python connector