zetta scale

Zenoh

A Crash Course in Building Cloud-to-Microcontroller Applications

Zenoh Team

SMART FACTORY

An interesting use case...

zetta scale

What is a smart factory?

An interconnected **network** of **machines**, **communication mechanisms**, and **computing power**

https://www.sap.com/insights/what-is-a-smart-factory.html

Data is key

Smart factories are built around data for **realtime operations**, **monitoring**, **data analytics**, etc.

Data in **motion** and data at **rest** are both crucial

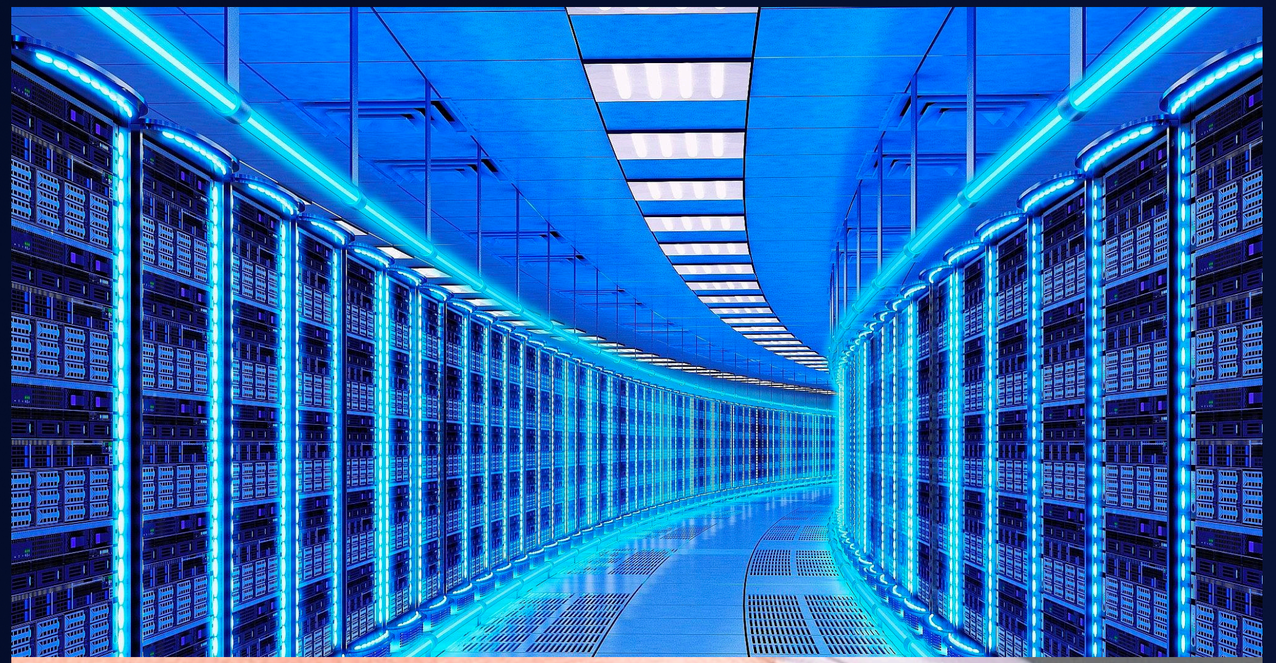
A variegated world

Different devices and systems live together

Sensors, robots, control systems, cloud, etc. need to cooperate and interact

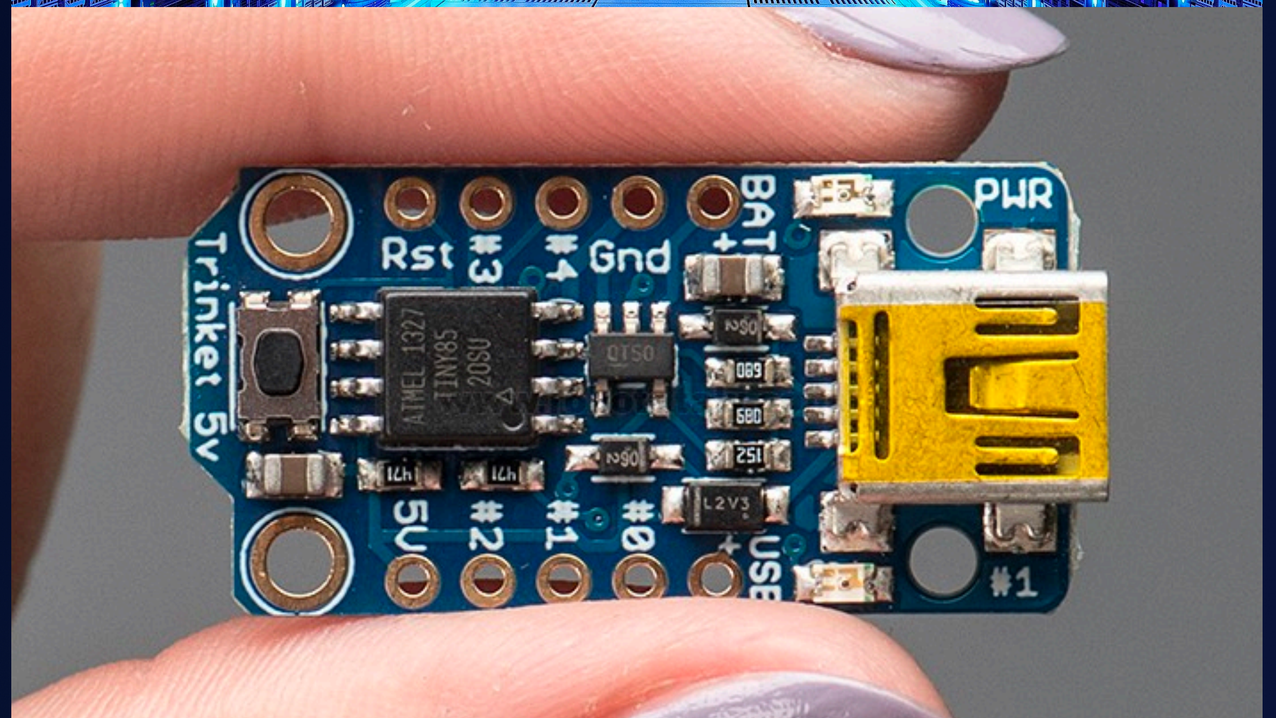
zetta scale

Making big the tiny world



Microcontrollers are as important as **data centres**

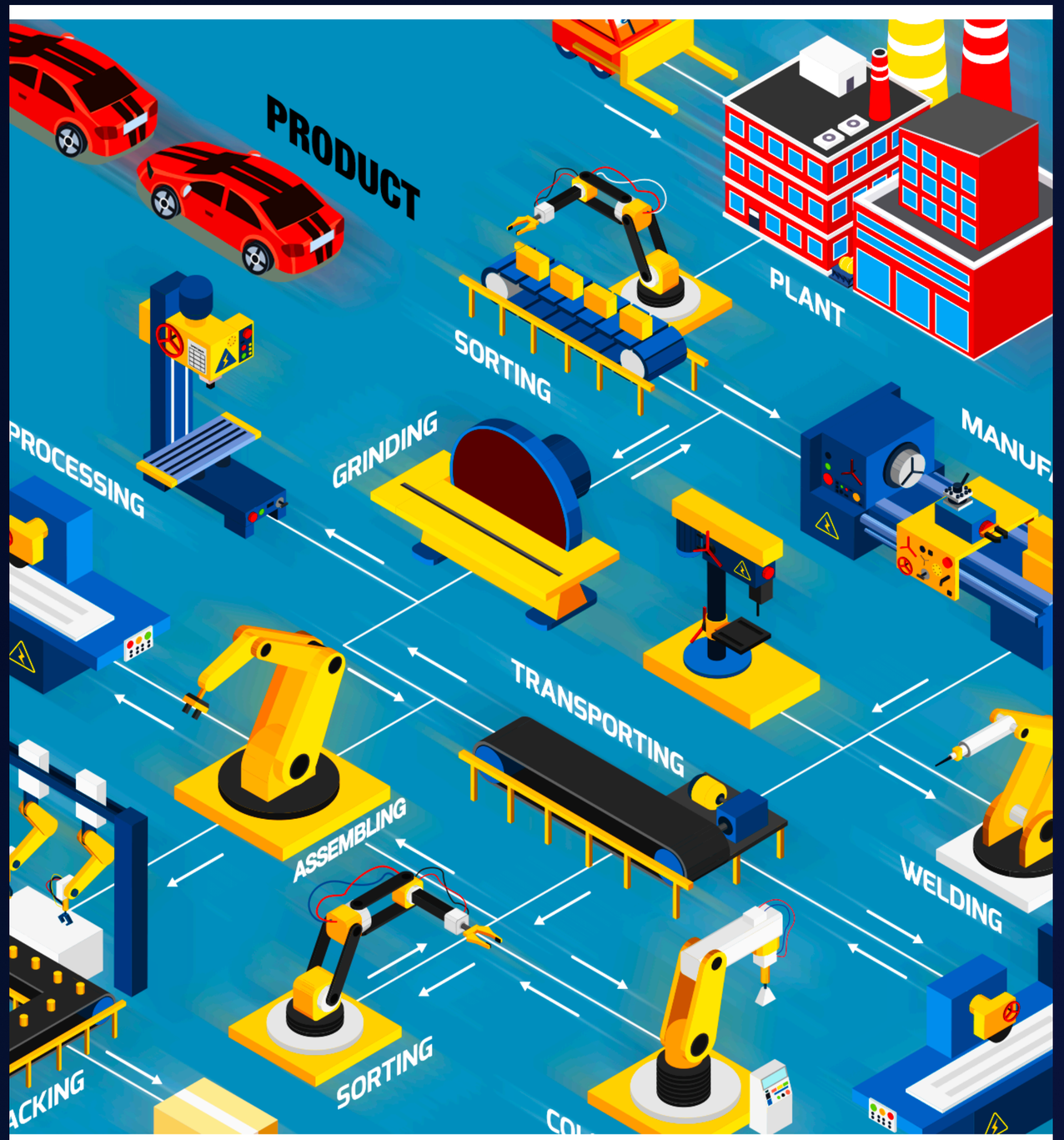
Data are **generated** and **consumed** by sensors, actuators, and apps in data centres



Speaking many languages

Multiple **communication protocols** (e.g. DDS, OPC-UA, MQTT, PROFINET, REST, etc.)

Multiple **network technologies** (e.g. WiFi, 5G, Serial, Bluetooth, etc.)



Talking freely

Communication in smart factories is a mix of

Peer-to-peer (e.g. between robots for coordination)

Infrastructured (e.g. with the control room)

zetta scale

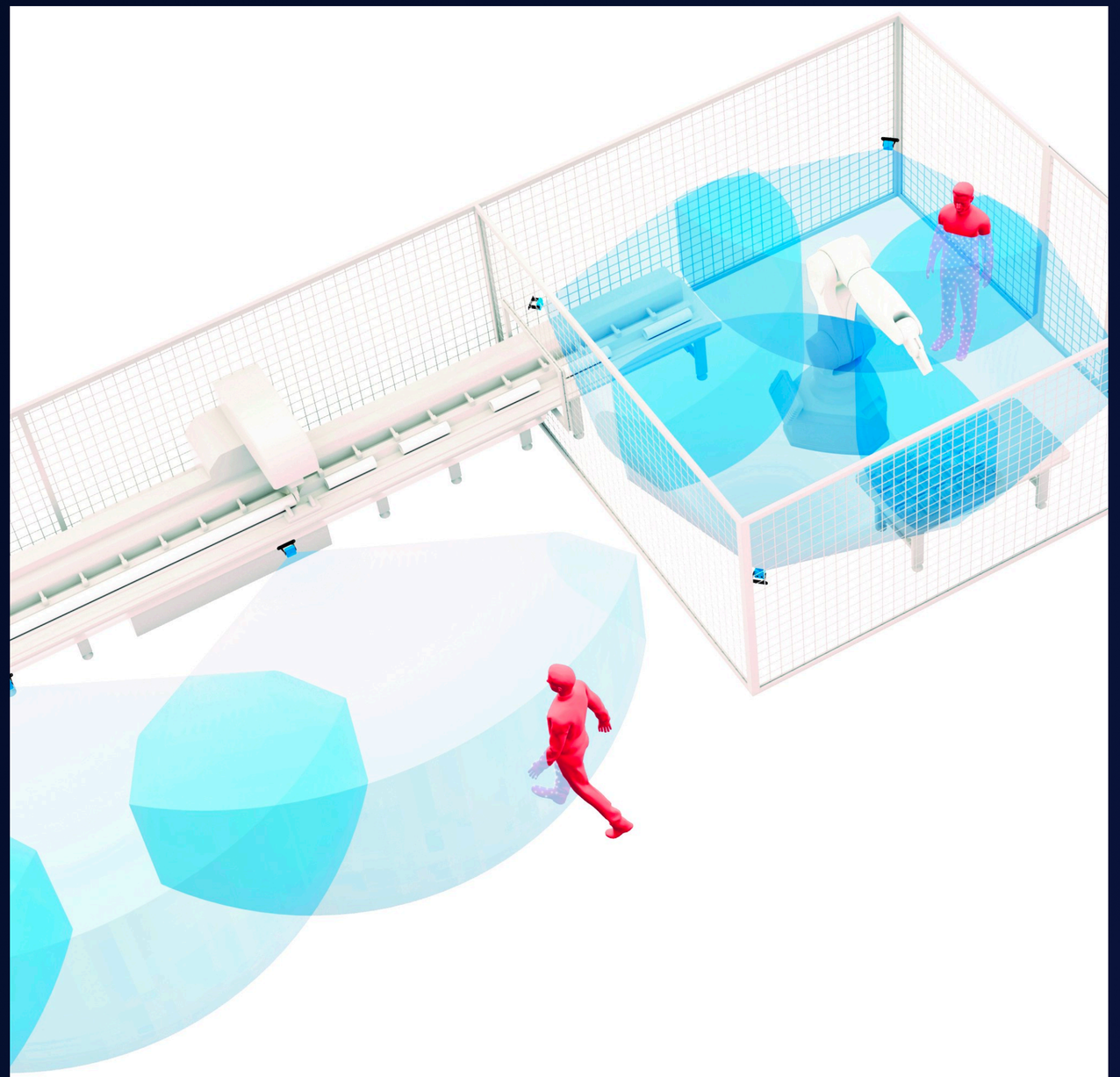
Let’s build a virtual fence for our smart factory!

zetta scale

Virtual fence and robots

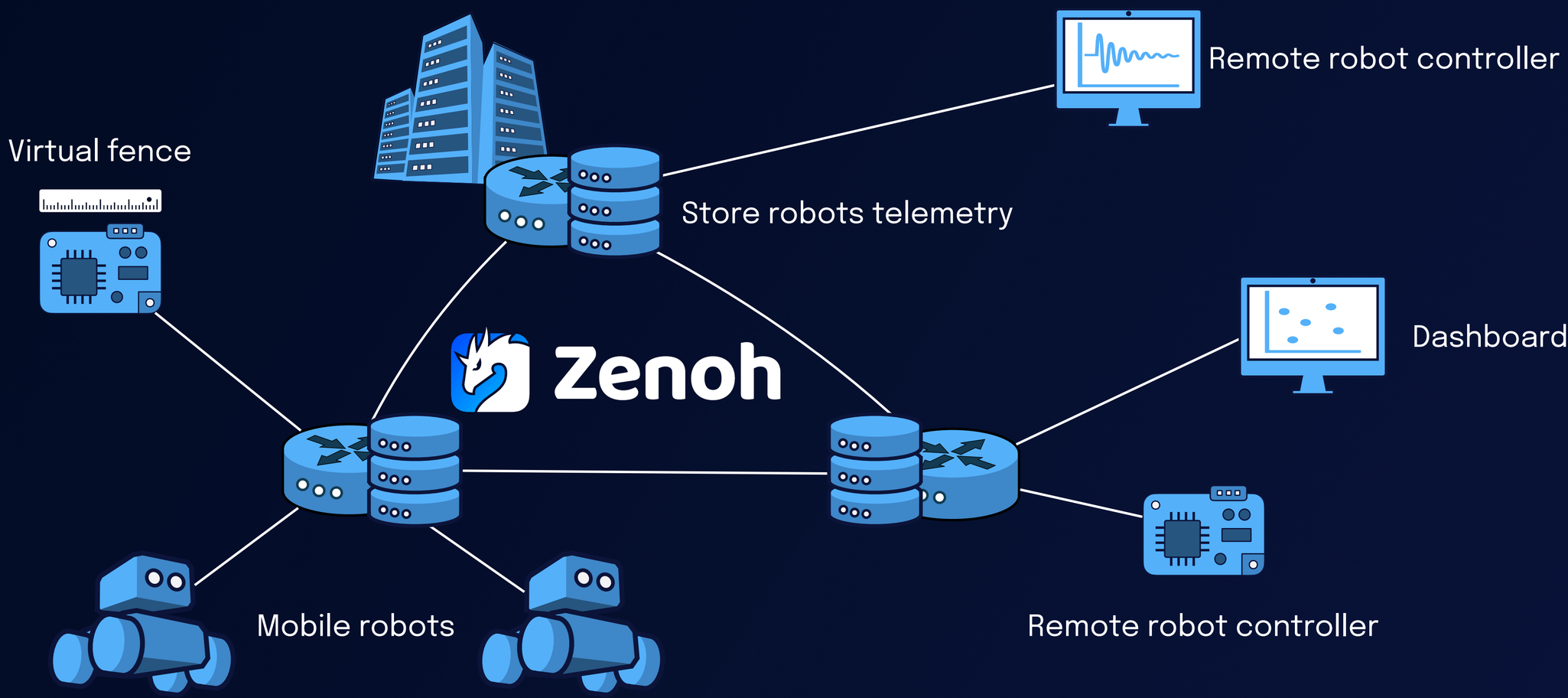
Sensors and **actuators** are ubiquitous: from the **building** to the **robots**

Distributed control system using a **virtual fence** to stop **robots**



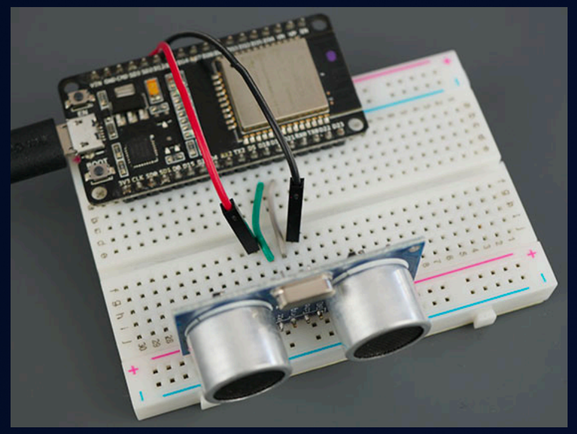
zetta scale

What we are going to build

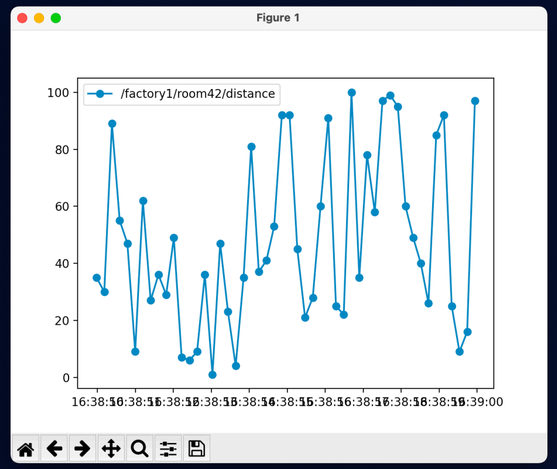


Virtual fence: let’s sense

ESP32 with ultrasonic sensor



Pub distance
factory1/room42/distance

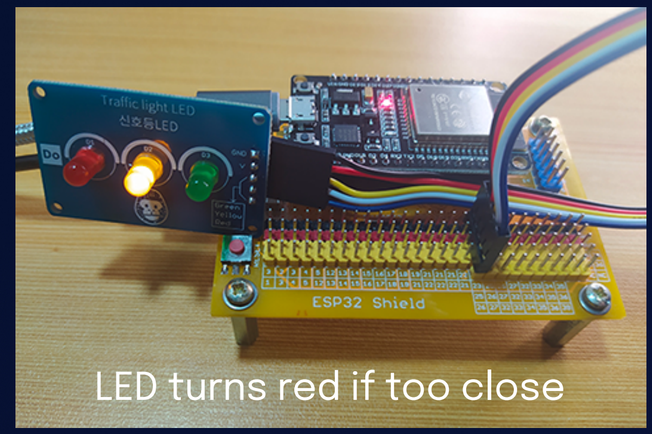


Sub distance

Zenoh

WiFi

ESP32 with LED



Sub distance

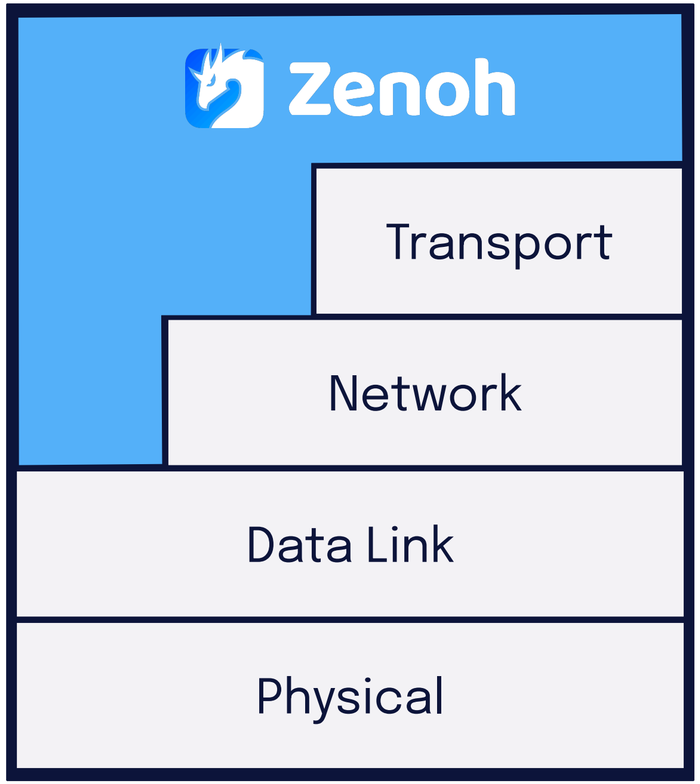
Pub alarm light
factory1/room42/led/green
factory1/room42/led/red

Zenoh runs everywhere

Native libraries and **API bindings** for many programming languages

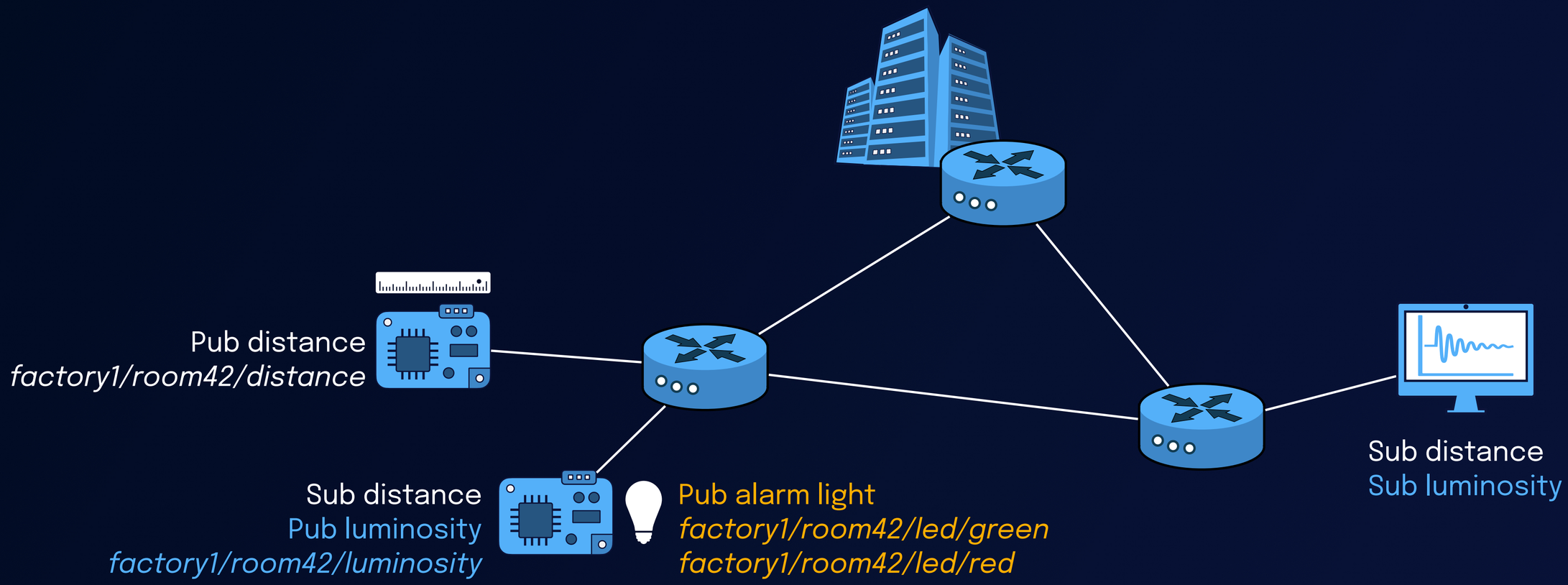
Over various **network technologies**: from **transport layer** to **data link**

On **embedded** and **constrained devices**



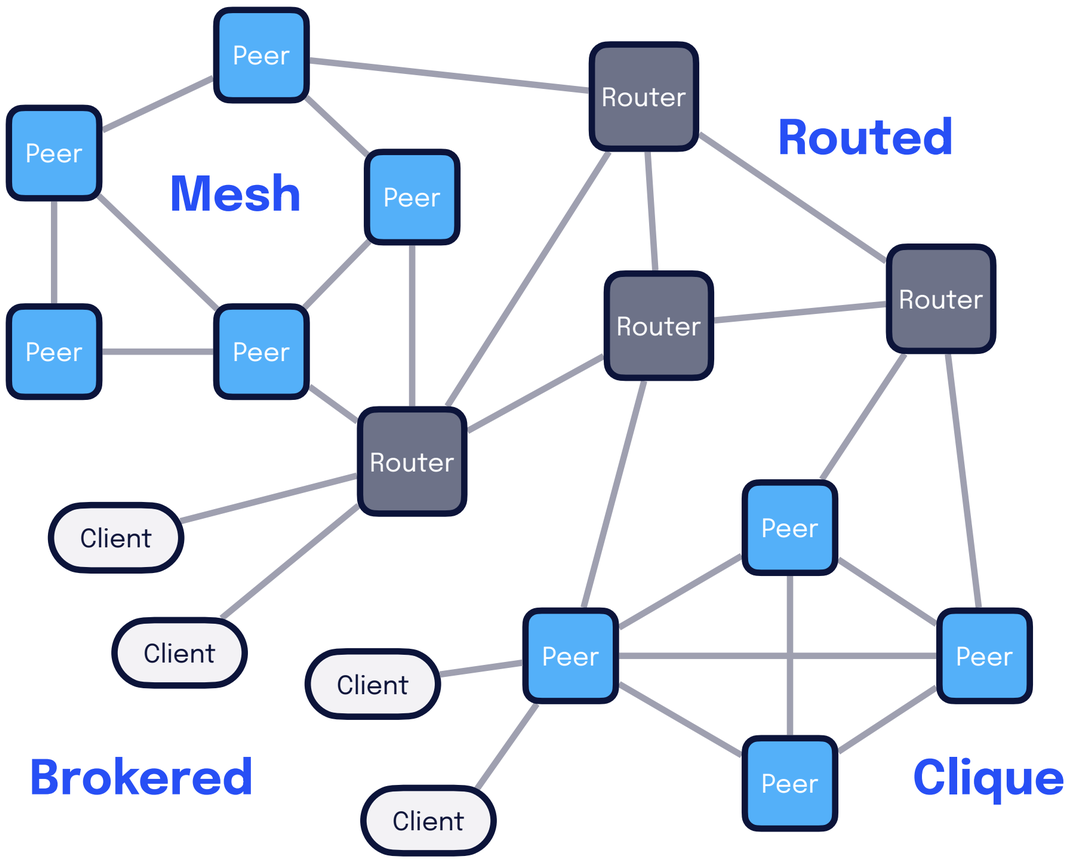
zetta scale

Extending beyond a LAN



zetta scale

Zenoh supports many topologies



Peer-to-peer

Clique and mesh topologies

Brokered

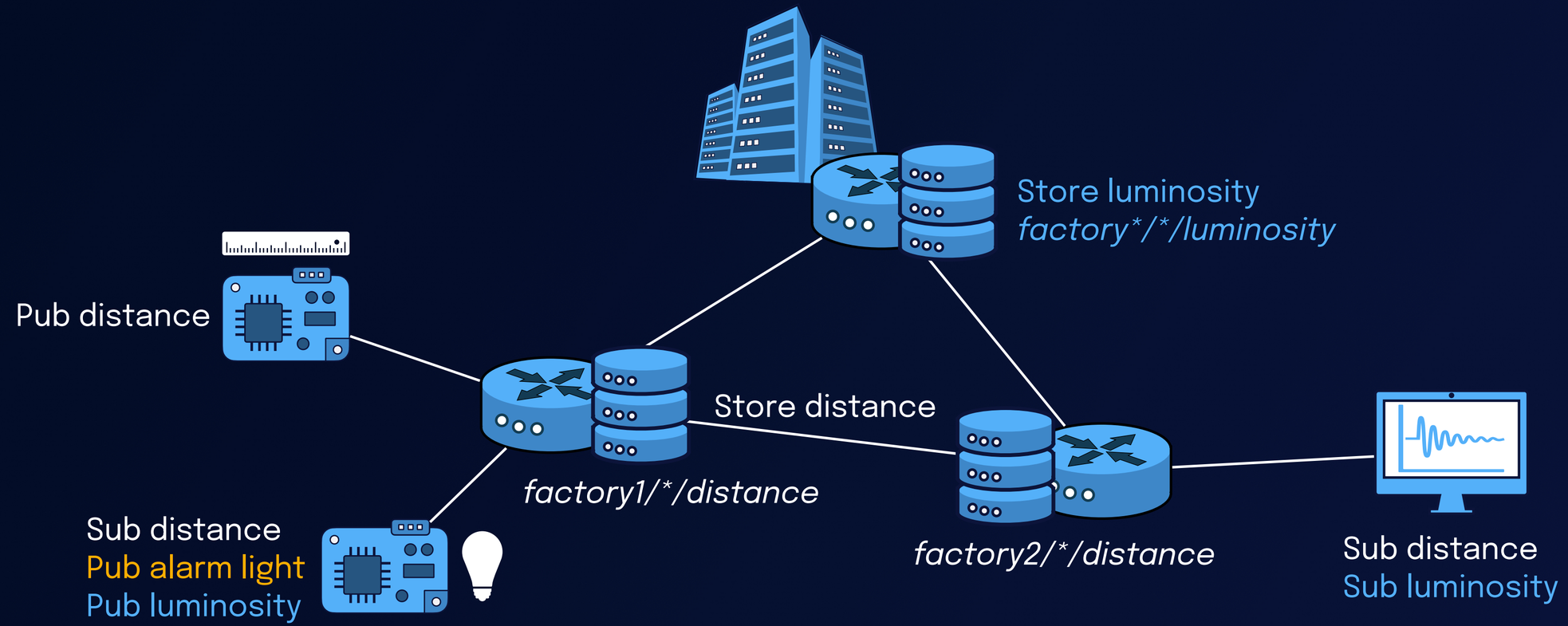
Clients communicate through a router or a peer

Routed

Routers forward data to and from peers and clients

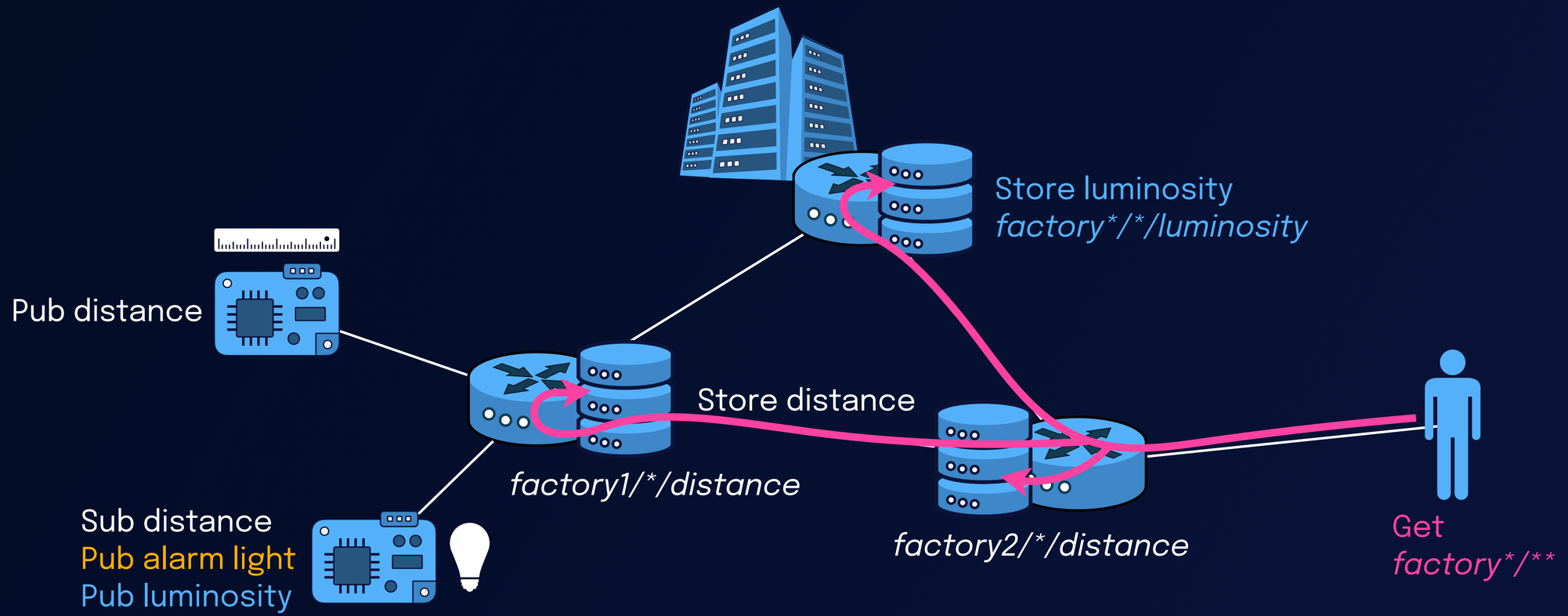
zetta scale

Storing data

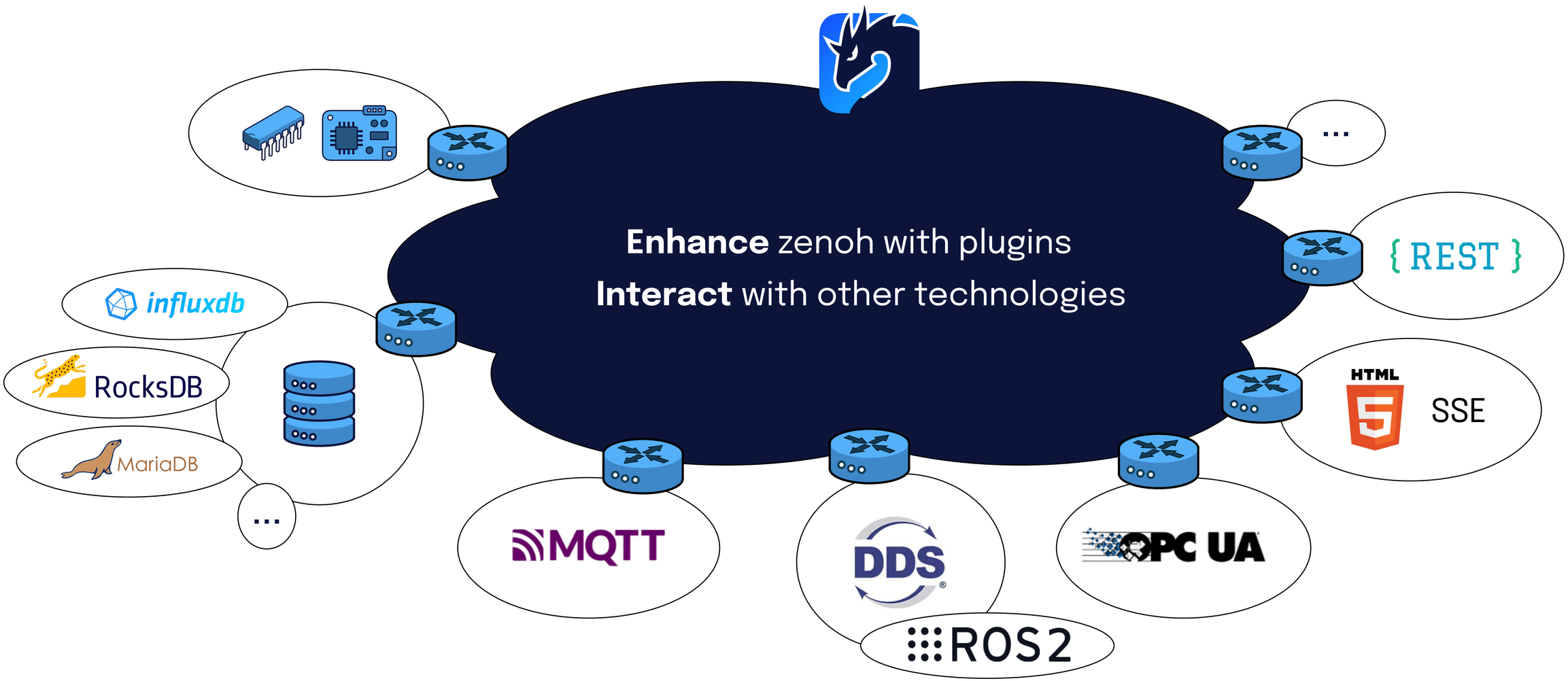


zetta scale

Retrieving data



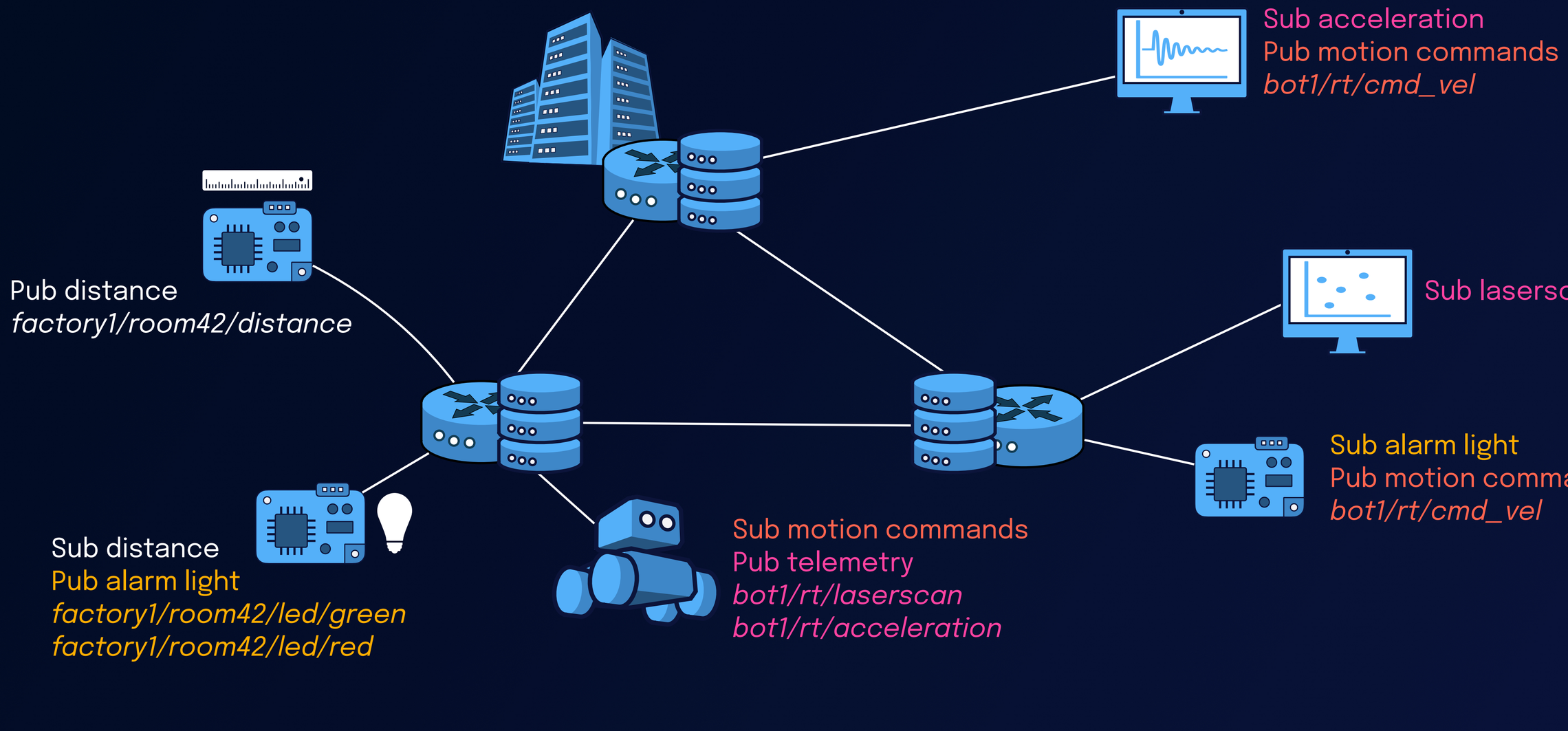
Zenoh is extensible



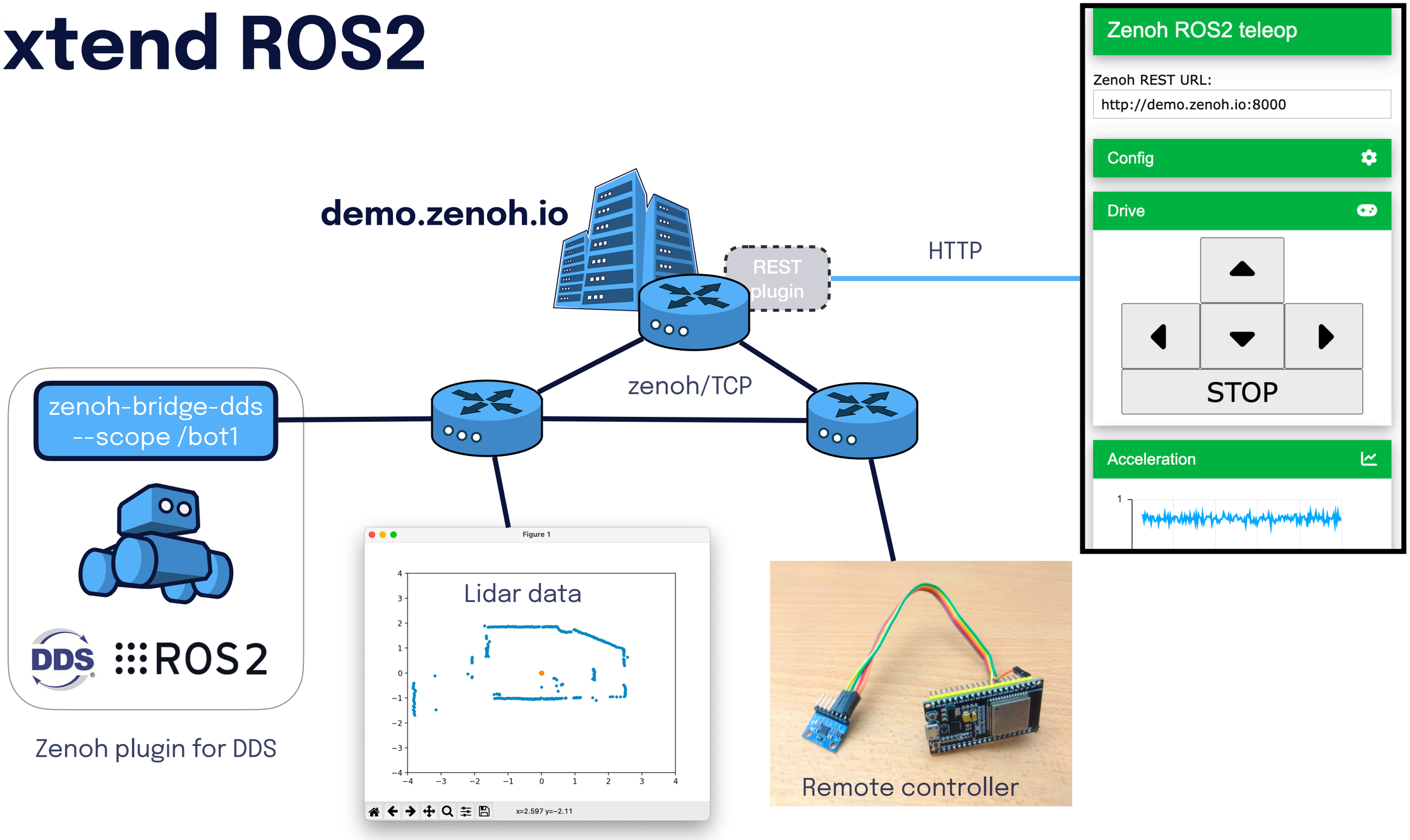
"Coming together is a beginning; keeping together is progress; working together is success." - Edward Everett Hale

zetta scale

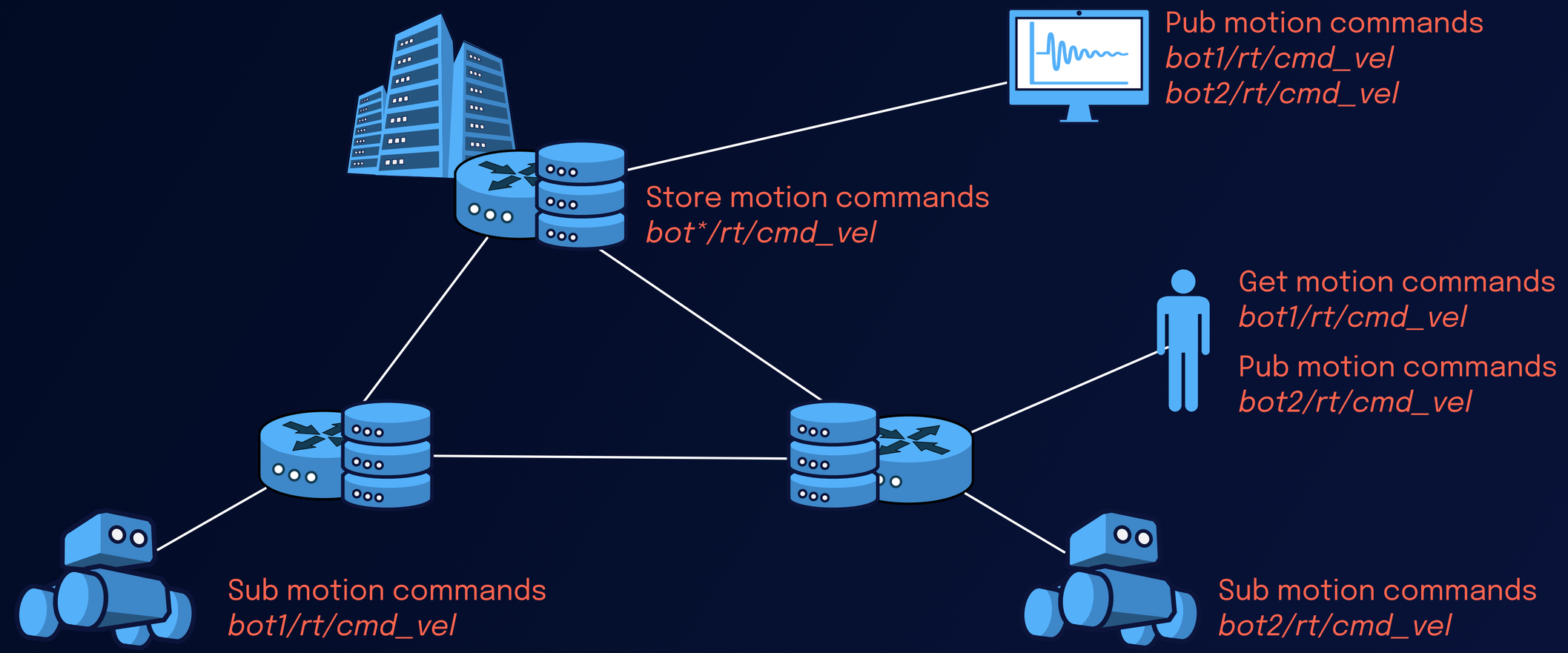
Robots are coming



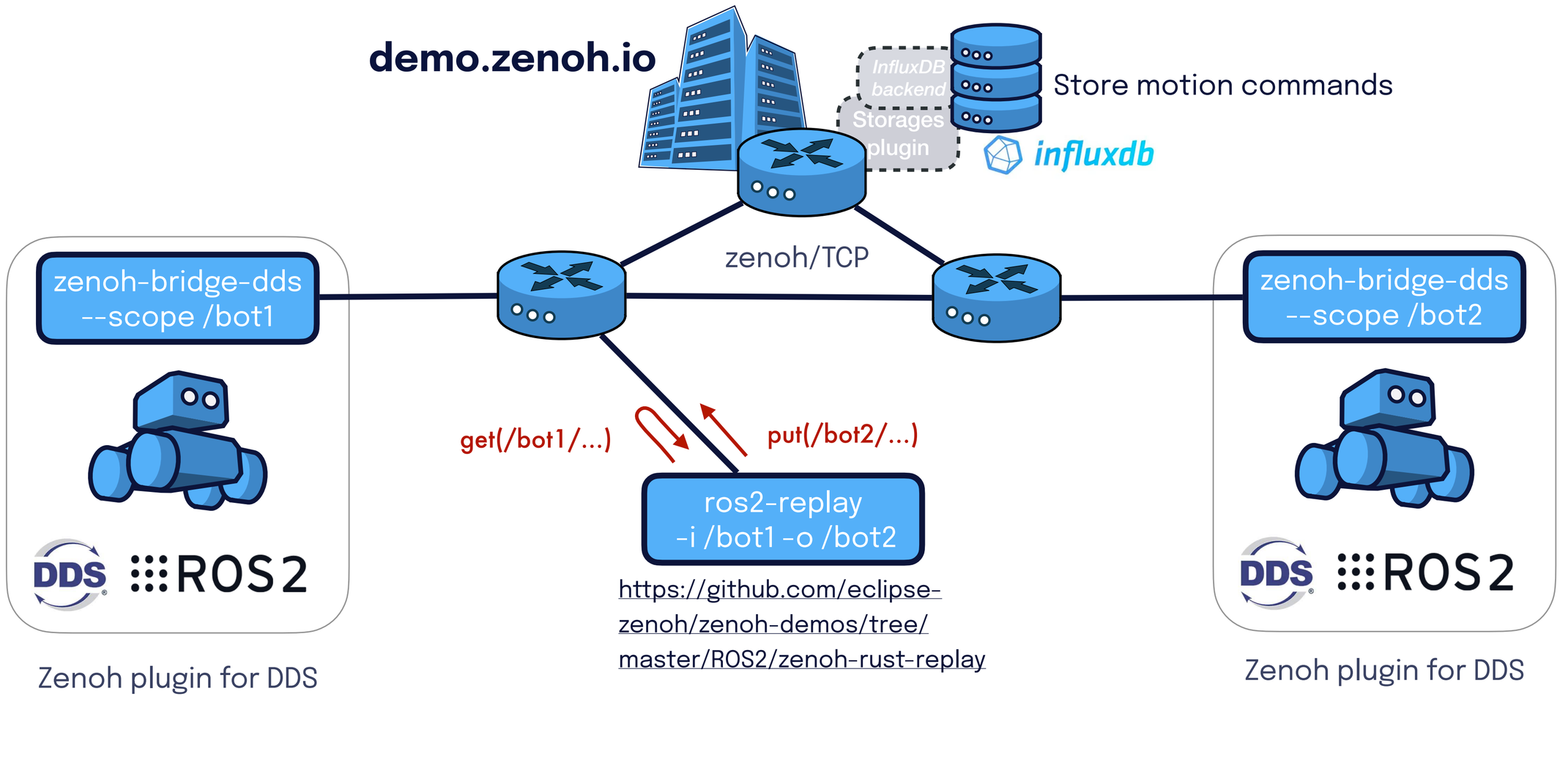
Extend ROS2


demo.zenoh.io
REST plugin
HTTP
zenoh/TCP
zenoh-bridge-dds --scope /bot1
DDS
ROS2
Zenoh plugin for DDS
Figure 1
Lidar data
x=2.597 y=−2.11
Remote controller
Zenoh ROS2 teleop
Zenoh REST URL:
http://demo.zenoh.io:8000
Config
Drive
STOP
Acceleration
1

Getting a twin



Record and replay ROS2



https://github.com/eclipse-zenoh/zenoh-demos/tree/master/ROS2/zenoh-rust-replay

Zenoh plugin for DDS

Zenoh plugin for DDS

Key takeaways

zetta scale

Zenoh and the smart factory

Seamless mixing of **real time** and **stored** data

Different **devices** and **networks:** from **micro-controllers** to the **cloud**

Integration with third-party **technologies**

zetta scale

Zenoh and robotics

Out-of-the box integration with **ROS2** robots

Intra-robot and **inter-robot** communication

High **throughput**, low **latency**, low **wire overhead**, and **simple API**

zetta scale

Zenoh

Unifies data in **motion**, data **in-use**, data at **rest** and **computations**
from **embedded microcontrollers** up to powerful **data centres**

Provides a **location-transparent API** for high performance **pub/sub**
and **distributed queries** across **heterogeneous** systems

Facilitates data representation **transcoding**, **geo-distributed storage**
and **distributed computed values** in a plug-and-play fashion

“Some people want it to happen, some wish it would happen, others make it happen.” – Michael Jordan

zetta scale

Zenoh-Flow: A data flow programming framework for the Cloud-to-Thing

Motivation

Zenoh holds a **promise**: becoming the *backbone of many (new) distributed applications*.

Zenoh-Flow builds on that promise: providing a **Cloud-to-Thing framework** that **eases** the “making” of complex applications.

zetta scale

Ambition

Simpler & Resilient foundations

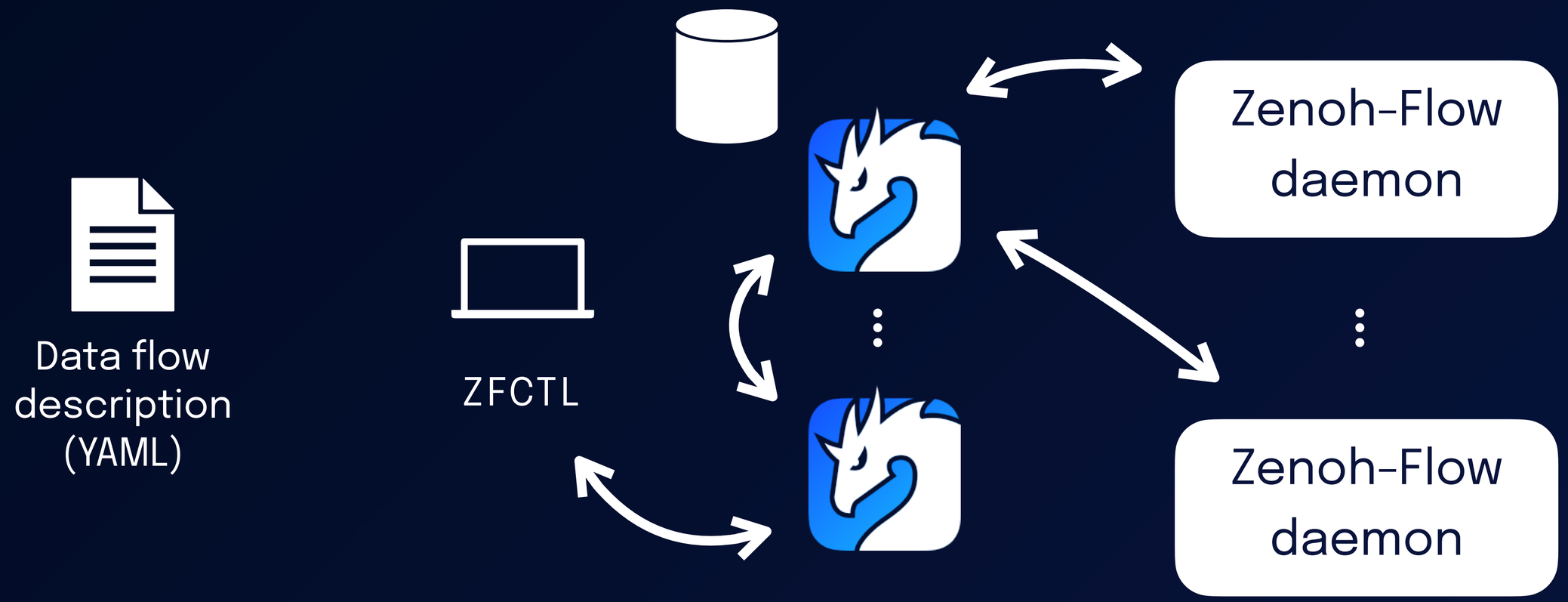
Declarative

Unified abstraction

Automatic deployment

Rust-based

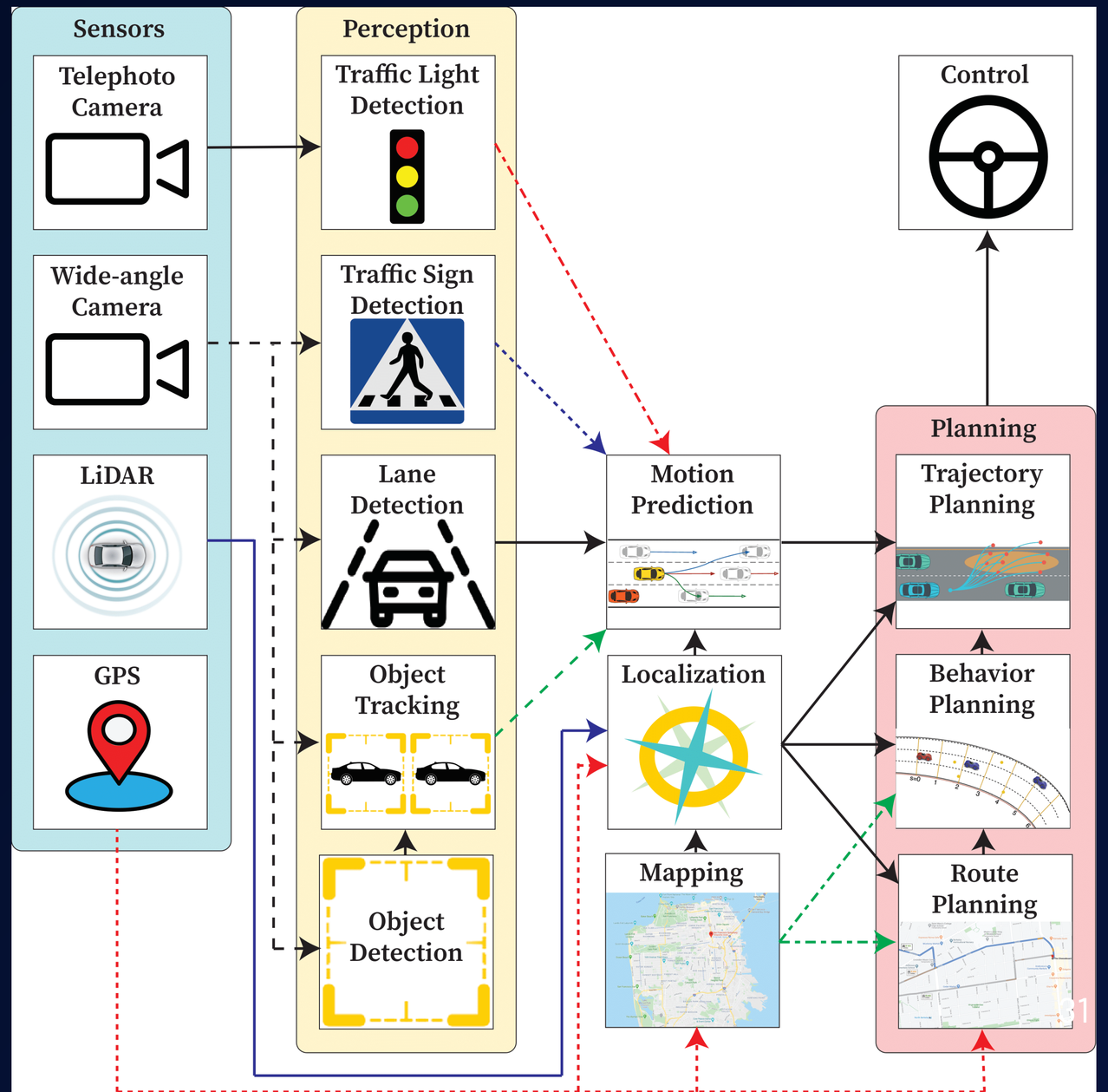
What it looks like



Example?

“**Pylot** is an autonomous vehicle platform for **developing** and **testing autonomous vehicle components** (e.g., perception, prediction, planning) on the **CARLA simulator** and real-world cars.”

https://github.com/erdos-project/pylot



Activities Terminator Apr 22 16:21 fr

abderahmane@goku: ~/work/ZF_pylot/dependencies/scenario_runner-0.9.11

abderahmane@goku: ~/work/ZF_pylot/dependencies/scenario_runner-0.9.11 65x43

```
│ Duration (Game Time)         │ 11.96s       │
│ Ratio (System Time / Game Time) │ 0.999s    │

 > Criteria Information

│ Actor                  │ Criterion                │ Result
│ Actual Value           │ Expected Value │
│ lincoln.mkz2017 (id=259) │ CheckMaximumVelocity (Opt.) │ SUCCES
S │ 0.012203460226641363 │ 20 │
│ lincoln.mkz2017 (id=259) │ CollisionTest (Req.) │ SUCCES
S │ 0 │ 0 │
│ lincoln.mkz2017 (id=259) │ CheckDrivenDistance (Req.) │ FAILUR
E │ 1.15 │ 150 │
│ tesla.model3 (id=260) │ CollisionTest (Req.) │ SUCCES
S │ 0 │ 0 │
│ │ Timeout (Req.) │ SUCCES
S │ 11.96 │ 180 │
│ │ GLOBAL RESULT │ FAILUR
E │ │ │

 ================================================================
================

Not all scenario tests were successful
No more scenarios .... Exiting
abderahmane@goku:~/work/ZF_pylot/dependencies/scenario_runner-0.9
.11$
```

abderahmane@goku: ~/work/zenoh/zenoh-flow/zenoh-flow-python 55x43


```
ss":null,"interface":null,"autoconnect":null},"gossip":
{"autoconnect":null,"peers_autoconnect":null},"add_tim
estamp":true,"local_routing":null,"queries_default_time
out":null,"transport":{"unicast":{"accept_timeout":1000
0,"accept_pending":100,"max_sessions":1000,"max_links":
1},"multicast":{"join_interval":2500,"max_sessions":100
0},"qos":{"enabled":true},"link":{"tx":{"sequence_numbe
r_resolution":268435456,"lease":10000,"keep_alive":2500
,"batch_size":65535,"queue":{"size":{"control":1,"real_
time":1,"interactive_high":1,"interactive_low":1,"data_
high":2,"data":4,"data_low":4,"background":4},"backoff"
:100},"threads":4},"rx":{"buffer_size":65535,"max_messa
ge_size":1073741824},"tls":{"root_ca_certificate":null,
"server_private_key":null,"server_certificate":null,"cl
ient_auth":null,"client_private_key":null,"client_certi
ficate":null}},"shared_memory":{"enabled":true},"auth":
{"usrpwd":{"user":null,"password":null,"dictionary_file
":null},"pubkey":{"public_key_pem":null,"private_key_pe
m":null,"public_key_file":null,"private_key_file":null,
"key_size":null,"known_keys_file":null}}},"plugins_sear
ch_dirs":["/usr/lib/"],"plugins":{"storage_manager":{"s
torages":{"zf":{"key_expr":"/zenoh-flow/**","volume":"m
emory"},"zfrpc":{"key_expr":"/zf/runtime/**","volume":"
memory"}}}}}
```


```
[2022-04-22T14:20:41Z INFO  zenoh::net::runtime] Using
PID: 91017AB0999648E7BC74147910B460EA
[2022-04-22T14:20:41Z INFO  zenoh::net::runtime::orches
trator] zenohd can be reached on tcp/192.168.16.147:788
7
[2022-04-22T14:20:41Z INFO  zenoh::net::runtime::orches
trator] zenohd can be reached on tcp/192.168.21.247:788
7
[2022-04-22T14:20:41Z INFO  zenoh::net::runtime::orches
trator] zenohd can be reached on tcp/172.17.0.1:7887
[2022-04-22T14:20:41Z INFO  zenoh::net::runtime::orches
trator] zenohd listening scout messages on 224.0.0.224:
7447
[2022-04-22T14:20:41Z INFO  zenohd] Successfully starte
d plugin storage_manager from "/usr/lib/libzplugin_stor
age_manager.so"
[2022-04-22T14:20:41Z INFO  zenohd] Finished loading pl
ugins
```

abderahmane@goku: ~/work/ZF_pylot 80x22

```
[2022-04-22T14:20:19Z INFO  zenoh_flow::runtime::dataflow::instance::runners] No
deRunner controller received stop trigger
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ RUST_LOG=zenoh_flow=info zenoh-flow-daemon -c
/etc/zenoh-flow/runtime.yaml
[2022-04-22T14:20:44Z INFO  zenoh_flow_daemon::daemon] Loaded extension /etc/zen
oh-flow/extensions.d/01-python.zfext
[2022-04-22T14:20:44Z WARN  zenoh_flow_daemon::daemon] Skipping /etc/zenoh-flow/
extensions.d/placeholder as it as no extension
[2022-04-22T14:20:45Z INFO  zenoh_flow_daemon::daemon] Runtime main loop startin
g
[2022-04-22T14:20:47Z INFO  zenoh_flow_daemon::daemon] Instantiating: ZF_AV
[2022-04-22T14:20:47Z INFO  zenoh_flow_daemon::daemon] Creating Flow ZF_AV - Ins
tance UUID: 50f9f6e8-a6b3-4cf1-a9e3-1a533afab8aa
[2022-04-22T14:20:47Z INFO  zenoh_flow_daemon::daemon] Preparing for Instance UU
ID: 50f9f6e8-a6b3-4cf1-a9e3-1a533afab8aa
[2022-04-22T14:21:14Z INFO  zenoh_flow_daemon::daemon] Instantiating: ZF_AV
[2022-04-22T14:21:14Z INFO  zenoh_flow_daemon::daemon] Creating Flow ZF_AV - Ins
tance UUID: 90076972-7c36-4866-bf0a-bc26a0918d84
[2022-04-22T14:21:14Z INFO  zenoh_flow_daemon::daemon] Preparing for Instance UU
ID: 90076972-7c36-4866-bf0a-bc26a0918d84
```

abderahmane@goku: ~/work/ZF_pylot 80x19

```
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ zfctl launch ZF_AV.yml
92450e8c-52db-465c-9709-076a74926dfd
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ zfctl launch ZF_AV.yml
thread 'main' panicked at 'called `Result::unwrap()` on an `Err` value: Empty',
zfctl/src/main.rs:550:64
note: run with `RUST_BACKTRACE=1` environment variable to display a backtrace
Aborted (core dumped)
abderahmane@goku:~/work/ZF_pylot$ zfctl launch ZF_AV.yml
```

abderahmane@goku: ~/work/ZF_pylot 100x17

```
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ sc
4.24.3-0+++UE4+Release-4.24 518 0
Disabling core dumps.
^CFUnixPlatformMisc::RequestExitWithStatus
FUnixPlatformMisc::RequestExit
Exiting abnormally (error code: 130)

abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ ^C
abderahmane@goku:~/work/ZF_pylot$ sc
4.24.3-0+++UE4+Release-4.24 518 0
Disabling core dumps.
```

abderahmane@goku: ~/work/pylot 101x17

```
abderahmane@goku:~/work/ZF_pylot$ cd ..
abderahmane@goku:~/work$ cd pylot/
abderahmane@goku:~/work/pylot$ code .
abderahmane@goku:~/work/pylot$ ^C
abderahmane@goku:~/work/pylot$ ^C
abderahmane@goku:~/work/pylot$ ^C
abderahmane@goku:~/work/pylot$ ^C
abderahmane@goku:~/work/pylot$ ^C
abderahmane@goku:~/work/pylot$ ^C
abderahmane@goku:~/work/pylot$ ^C
abderahmane@goku:~/work/pylot$
```

zetta scale

...and the blog

zenoh.io

Home Documentation Community Adopters Media Blog

Zenoh

Zero Overhead Pub/sub, Store/Query and Compute.

zenoh /zeno/ unifies data in motion, data at rest and computations. It elegantly blends traditional pub/sub with geo distributed storage, queries and computations, while retaining a level of time and space efficiency that is well beyond any of the mainstream stacks.

Get started

Don't forget to visit Zenoh's website...

https://zenoh.io/

zenoh.io

Documentation Community Adopters Media **Blog**

...ve and

...valuation ...onous

... and ...

DragonBotOne Egg Hatching with Zenoh and Zenoh-Pico

ROS 2 and microcontrollers integration via Zenoh-pico

Zenoh goes embedded with zenoh-pico

Indy Autonomous Challenge (IAC): Experiences from the

Zenoh-Pico: Above and Beyond

09 June 2022 -- Paris.

In a previous blog post, we introduced Zenoh-Pico, an implementation of Zenoh for microcontrollers and embedded devices, along with a preliminary performance results and its integration on off-the-shelf robots (by bridging both legacy ROS2+DDS and Zenoh systems or by making it a full-fledged Zenoh system).

In this post, we will dive deeper on Zenoh-Pico, show, how Zenoh-Pico is capable of:

- exchanging close to **2.5M msg/s** for small payloads, and over **25 Gbps** for larger messages,
- achieving end-to-end latency (i.e., one way delay) as small as **45 µsec** and **15 µsec** for unicast and multicast transports, respectively,
- minimizing the overhead in the wire down to **5 bytes** per data transmission,
- fitting all its capabilities in less than **50KB** footprint, which can be quickly reduced to **~15KB** in tailored compilation setups, and
- provides **simple** to use and yet **powerful APIs**.