Introduction to Cyphal

A modern protocol for intravehicular distributed computing and communication.

Alexander Entinger, MSc

LXRobotics

December 6, 2023

Who am I?

Ing. Alexander Entinger, MSc

Owner LXRobotics GmbH

What do I do?

I provide both expertise and hands-on development concerning **reliable real-time communication and control** for robotic systems.

Table of contents

1. Theoretical introduction
   - Main principles
   - Data structure description language
   - Transport layer
   - Minimum message & service set
   - Motivation
   - Adopters, users & use cases
2. Practical demonstration

OpenCyphal

Theoretical introduction

Main principles

Architecture

A Cyphal network is a decentralized peer network.[1]

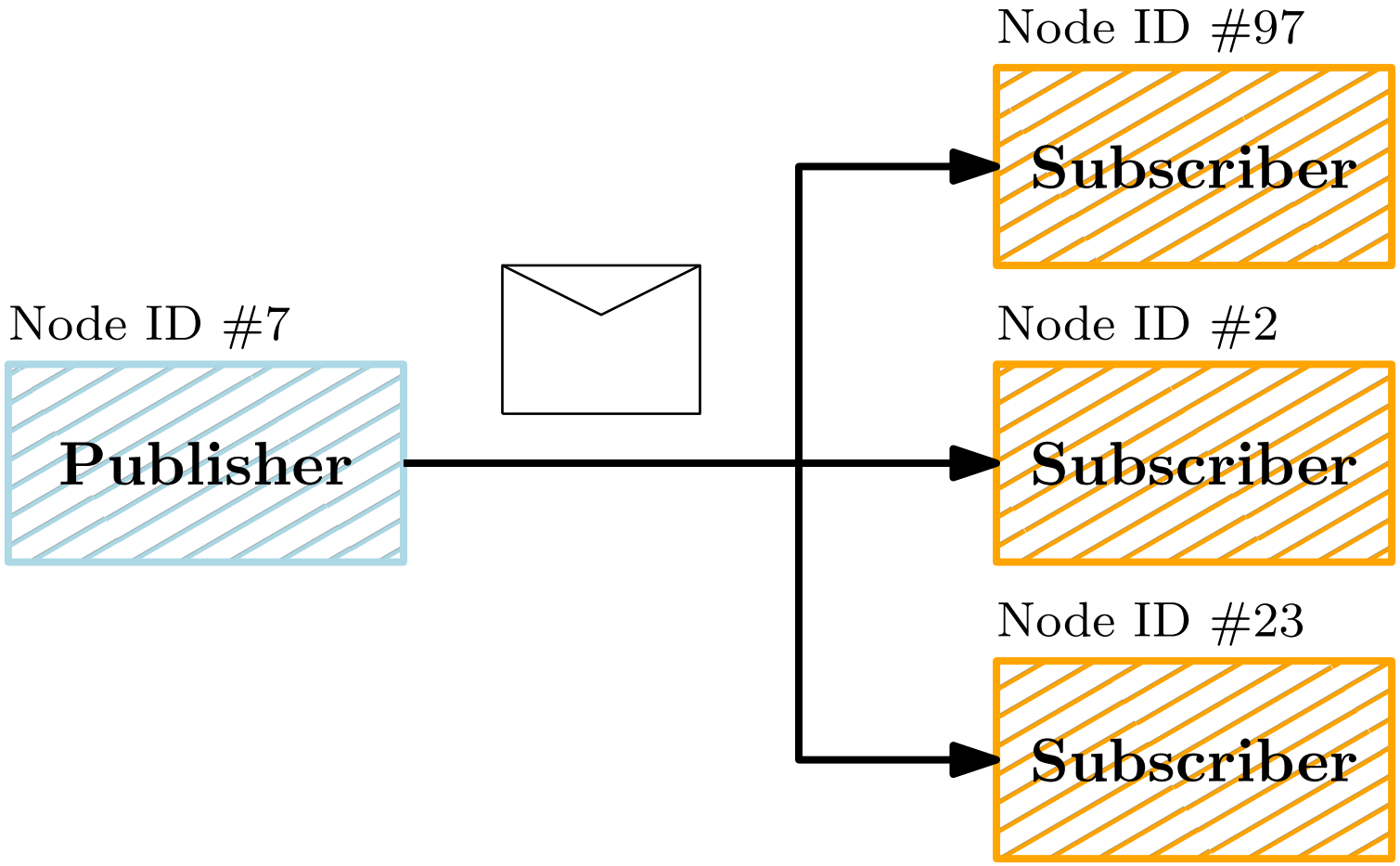
Architecture

Each Cyphal node has a unique numeric identifier.

Architecture

Nodes exchange data via message publication and service invocation.

Message publication

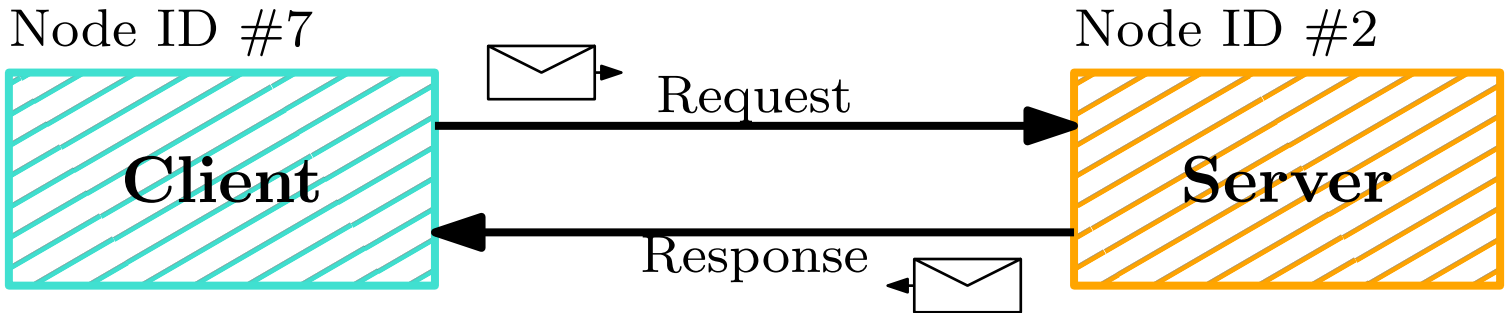


Subject-ID

A message is identified by a unique subject-ID.

$$(0 \leq \text{Subject-ID} \leq 2^{13} - 1)$$

Service invocation



Service-ID

Service request and response messages are identified by a unique service-ID.

$$(0 \leq \text{Service-ID} \leq 2^9 - 1)$$

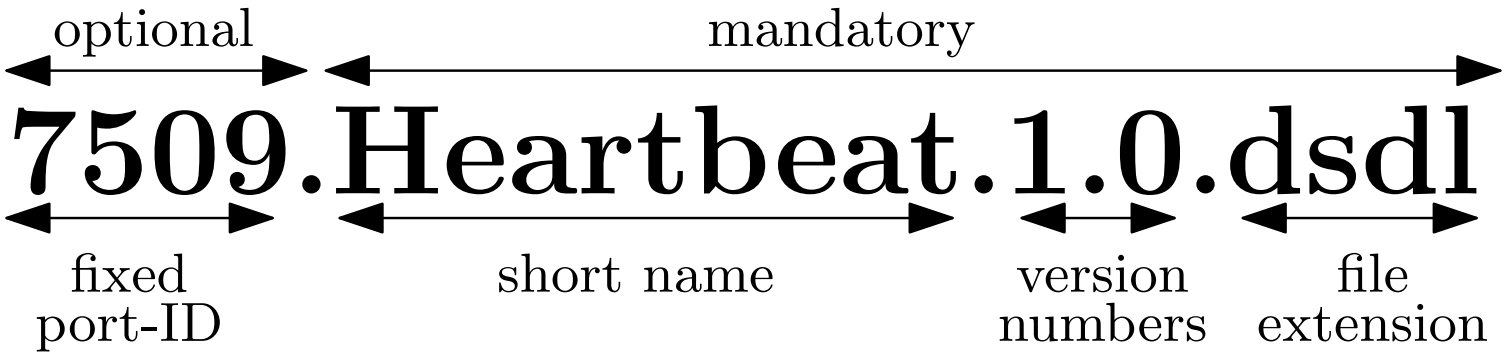
Data structure description language

Definition

The data structure description language is a domain specific description language.

DSDL data type definitions

- are contained in a UTF-8 encoded **text file**
- have the file extension **.dsdl**
- are **versioned**
- may have a **fixed port-ID**
  - port-ID → subject-ID (message)
  - port-ID → service-ID (service)



Example for a message DSDL data type definition

7509.**Heartbeat.1.0**.dsdl

```
uint16 MAX_PUBLICATION_PERIOD = 1    # [second]
uint16 OFFLINE_TIMEOUT = 3           # [second]
uint32 uptime                        # [second]

Health.1.0 health
Mode.1.0 mode

uint8 vendor_specific_status_code
```

Mode.1.0.dsdl

```
uint3 value

uint3 OPERATIONAL = 0
uint3 INITIALIZATION = 1
uint3 MAINTENANCE = 2
uint3 SOFTWARE_UPDATE = 3
```

Example for a service DSDL data type definition

435.**ExecuteCommand**.1.1.dsdl

```
uint16 command
uint16 COMMAND_POWER_OFF = 65534
uint16 COMMAND_BEGIN_SOFTWARE_UPDATE = 65533
uint16 COMMAND_FACTORY_RESET = 65532
uint16 COMMAND_EMERGENCY_STOP = 65531
uint16 COMMAND_STORE_PERSISTENT_STATES = 65530
uint8[<=uavcan.file.Path.2.0.MAX_LENGTH] parameter
---
uint8 STATUS_SUCCESS          = 0
uint8 STATUS_FAILURE          = 1
uint8 STATUS_NOT_AUTHORIZED = 2
uint8 STATUS_BAD_COMMAND      = 3
uint8 STATUS_BAD_PARAMETER    = 4
uint8 STATUS_BAD_STATE        = 5
uint8 STATUS_INTERNAL_ERROR = 6
uint8 status
```

Regulated DSDL definitions

Regulated data types include **standard** and **domain-specific** public definitions.

Regulated DSDL definitions

All regulated DSDL definitions can be found at `https://github.com/OpenCyphal/public_regulated_data_types`.

DSDL → C/C++/Python/HTML

Using **nnvg**[1] DSDL can be transpiled into C/C++/Python and HTML.

```
struct Heartbeat_1_0 final
{
  inline nunavut::support::SerializeResult serialize(
    const Heartbeat_1_0& obj,
    nunavut::support::bitspan out_buffer)
    { /* ... */ }
  inline nunavut::support::SerializeResult deserialize(
    Heartbeat_1_0& obj,
    nunavut::support::const_bitspan in_buffer) {
    { /* ... */ }
```

[1] https://github.com/OpenCyphal/nunavut

Transport layer

CAN, UDP, Serial

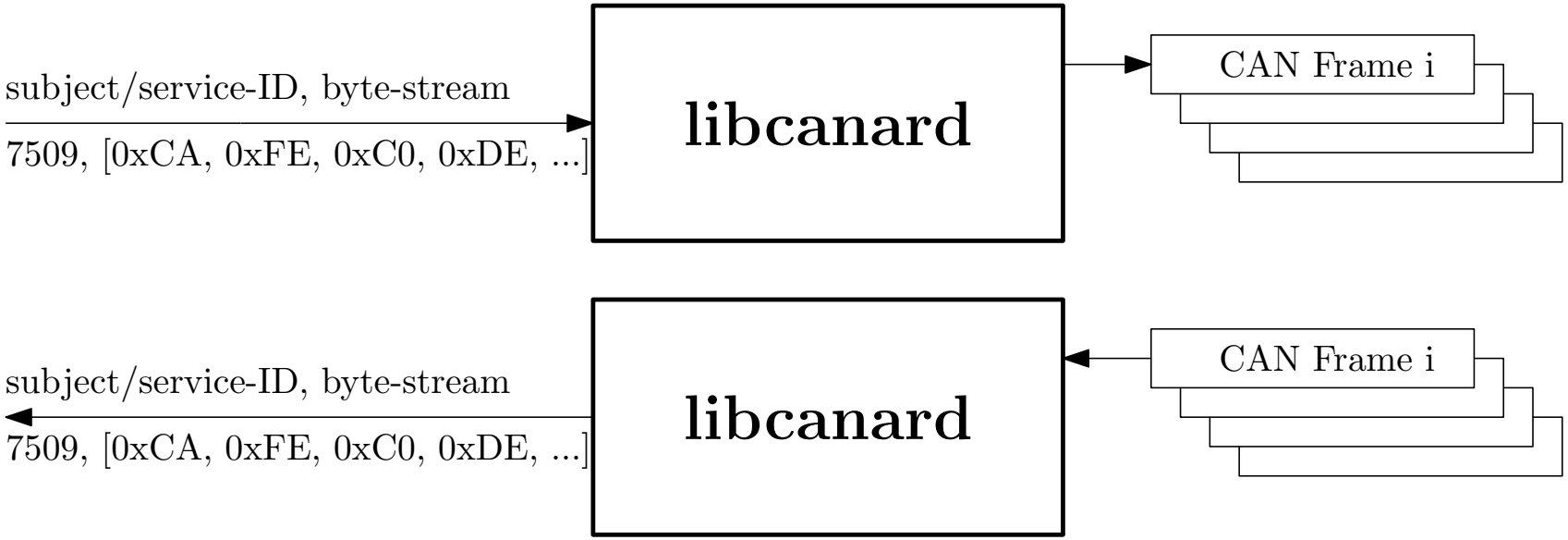
Transport layer	Library	MISRA C	License
CAN, CAN FD	libcanard[2]	Yes	MIT
UDP Ethernet: 10BASE-T1S, ...	libudpard[3]	Yes	MIT
Serial UART, RS422/232, TCP/IP, ...	libserard[4]	Yes	MIT

- *libudpard* works on any abstraction similar to **Berkely sockets**.

[2] https://github.com/OpenCyphal/libcanard
[3] https://github.com/OpenCyphal/libudpard
[4] https://github.com/OpenCyphal-garage/libserard, experimental

lib[can/udp/ser]ard

Convert **serialized byte-stream** into appropriate **transport layer** representation.



Convert **transport layer** representation into **byte-stream for deserialization**.

Minimum message & service set

Heartbeat

7509.**Heartbeat**.1.0.dsdl

- needs to be sent at least **once / second**
- contains information on node
  - up time
  - mode
  - health

Node information service

430.**GetInfo**.1.0.dsdl

- provides **run-time static** node-specific information on request:
  - protocol version
  - hardware version
  - software version
  - software revision id
  - unique id
  - name
  - ...
- queried by tools i.e. **yakut** and **yukon**

```
NodID Mode Health VSSC Uptime         VProtcl VHardwr VSoftware(major.minor.vcs.crc)          Unique-ID                          Name
    0 oper nomina    0     0d00:04:10   1.0     1.0     0.1.7b41c1b93afc3120                   e660c06213516529000000000000000 107-systems.open-cyphal-pico-base
   93 oper nomina   11     0d00:00:09   1.0     0.0     0.13                                   3f87b3ec67f0ead1cdea6e01cac88c23 org.opencyphal.yakut.monitor
Legend: pub/cln|sub/srv|(pub+sub)/(cln+srv)|activity|uavcan.node.port.List is published/not|
```

Register interface

The register interface allows the parametrization and configuration of a Cyphal node.

Register

- is identified by a **unique name**
- has a scalar or array **value**
  - String
  - Bit
  - Integer8/16/32/64
  - Natural8/16/32/64
  - Real16/32/64
- can be **read-only** or **read-write**
- **persistent** or **volatile** storage

Register list/access service

385.**List**.1.0.dsdl

- obtain a list of all registers

```
y rl 10
  [cyphal.node.description,
   cyphal.node.id,
   cyphal.pub.analoginput0.id,
   cyphal.pub.analoginput0.type,
   ...
```

384.**Access**.1.0.dsdl

```
y r 10 cyphal.node.id
  10
```

- read from a register
- write to a register

```
y r 10 cyphal.node.id 47
  47
```

Subject/Service List

7510.**List**.1.0.dsdl

- needs to be sent at least **once / 10 seconds**
- contains information on
  - subscribed & published messages
  - consumed & provided services
- used for network introspection and diagnostics

Motivation

Cyphal is a DDS[5] for **deeply embedded** targets first, **evolving up** to integrate w/ modern robotic systems.

[5] Data Distribution Service: a middleware for real-time and scalable communication between distributed systems

Adopters, users & use cases

Cyphal adopters and users

Zubax

LXRobotics

FOUCAULT DYNAMICS

revolve NTNU

107-SYSTEMS

RACCOON LAB

AIR ENERGY

Figure: Amazon Prime Air Drone MK30[2]

LXRobotics L3X-Z

- a electric/hydraulic hexapod robot for Search&Rescue:
  - 1 x servo controller for 12 hydraulic valves
  - 2 x hydraulic pressure sensors
  - 1 x hydraulic pump
  - 1 x radiation sensor
  - 1 x general IO controller
- and a demonstrator of **a fully Cyphal networked system**!

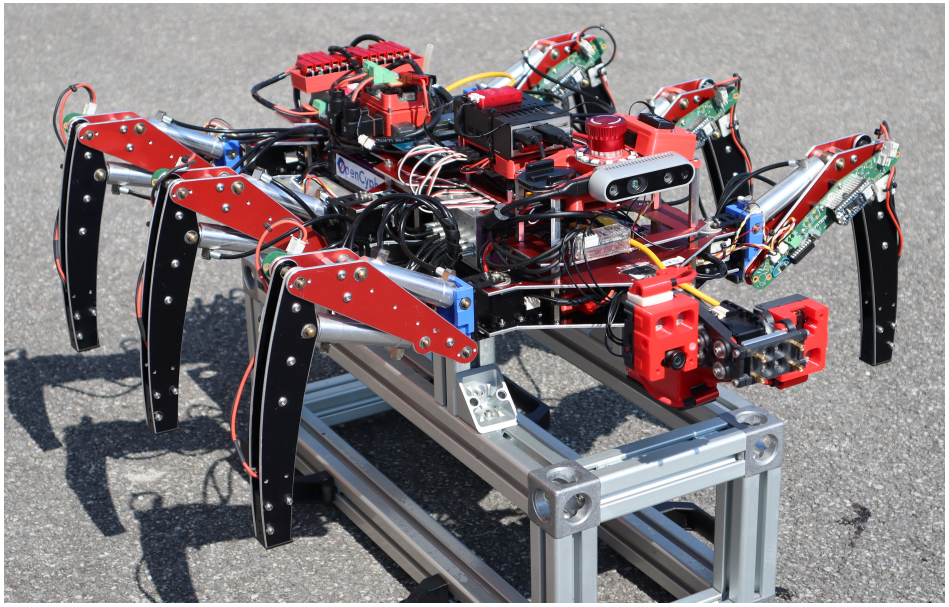
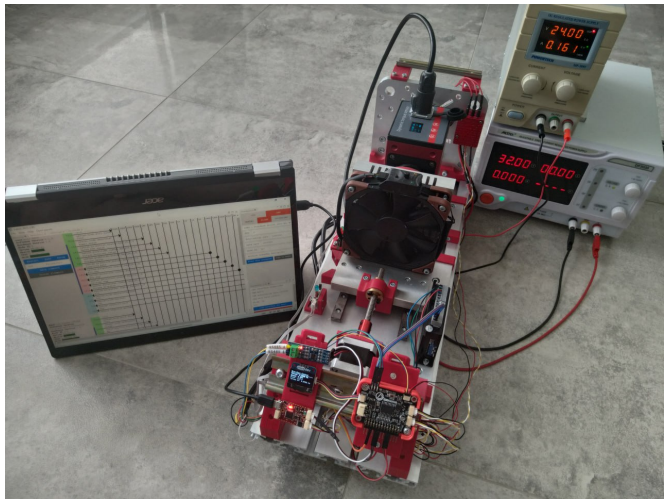
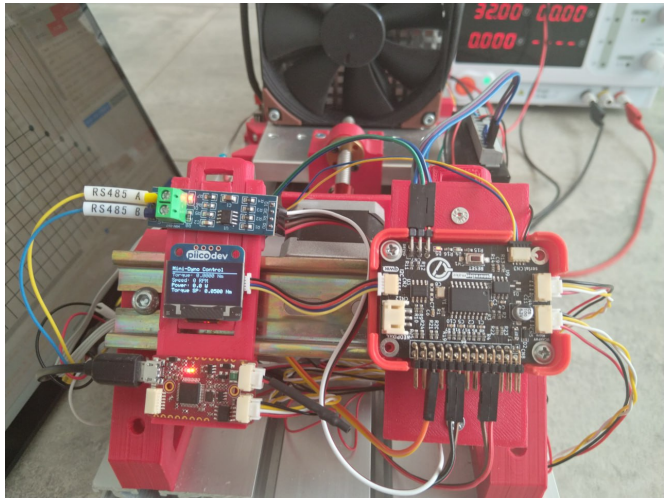


Figure: L3X-Z hexapod robot[3]

Foucault Systems Mini-Dyno

- a Cyphal-enabled **precision laboratory dynamometer** for testing of electric motors and electronic speed controllers.
    - Variable Air-Gap Permanent Magnet Eddy-Current Mechanical Resistance Unit



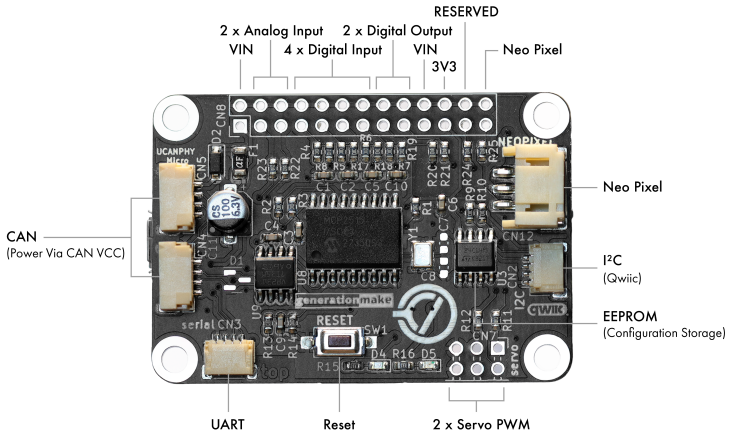


OpenCyphal

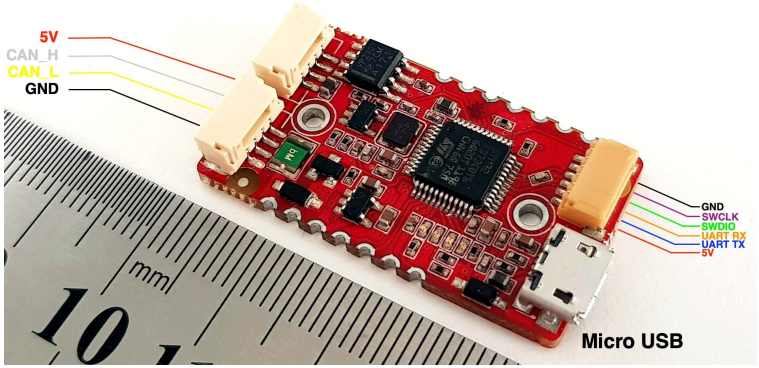
Practical demonstration

Hardware setup

- **CyphalPicoBase/CAN**
  - RP2040 based open hardware reference design (CERN-OHL-W-2.0)
  - Fully Cyphal-compliant open source firmware (MIT)



- **Zubax Babel**
  - USB-CAN Adapter via SLCAN
  - Power supply for CyphalPicoBase/CAN



Questions

Bibliography I

S. D. Pavel Kirienko *et al.*, "Opencyphal: an open technology for robust real-time intravehicular distributed computing and communication based on modern networking standards," 6 2023.

C. Chen, "11 photos of amazon's new prime air drone that can fly in light rain and deliver packages up to 5 pounds in under an hour," 2023. `https://www.aboutamazon.com/news/transportation/amazon-prime-air-drone-delivery-mk30-photos`.

A. Entinger, "L3x-z: a mixed electric/hydraulic hexapod robot," 2023. `https://github.com/107-systems/l3xz`.