PART 2

PRACTICAL SOFTWARE-DEFINED UNDERWATER NETWORKS

Prasad, Chinmay, Shiraz
Subnero Pte. Ltd.
Singapore

Global Oceans 2020 - Singapore U.S. Gulf Coast

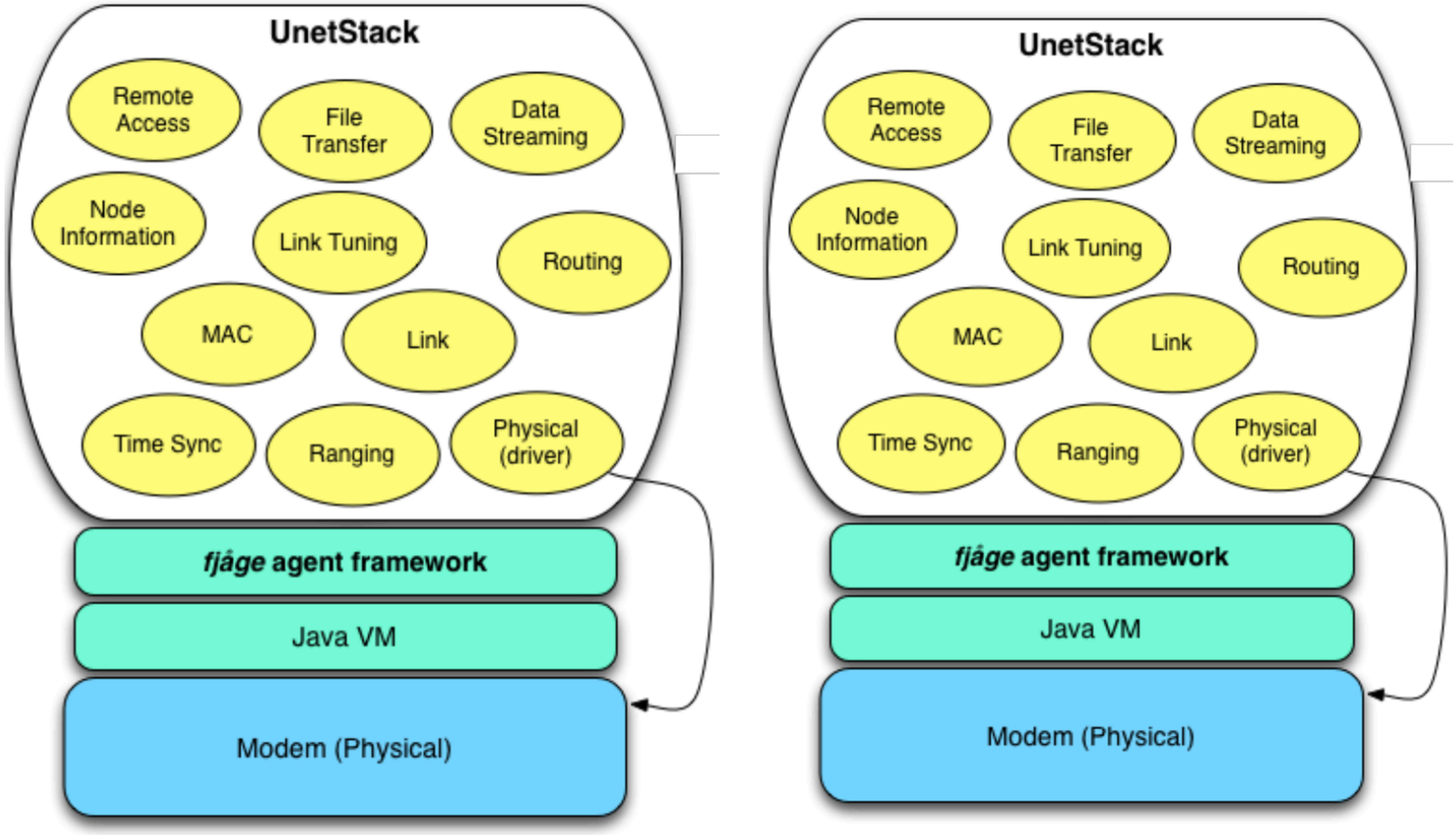
2 NODE NETWORK - DEMO

```
import org.arl.fjage.*

////////////////////////////////////////////////////////////////////////////
// display documentation

println '''
2-node network
--------------

Node A: tcp://localhost:1101, http://localhost:8081/
Node B: tcp://localhost:1102, http://localhost:8082/
'''

////////////////////////////////////////////////////////////////////////////
// simulator configuration

platform = RealTimePlatform    // use real-time mode

// run the simulation forever
simulate {
  node 'A', location: [ 0.km, 0.km, -15.m], web: 8081, api: 1101, stack: "$home/etc/setup"
  node 'B', location: [ 1.km, 0.km, -15.m], web: 8082, api: 1102, stack: "$home/etc/setup"
}
```

AGENTS AND LAYERS

Traditional stacks are based on layers

UnetStack is based on an agent architecture



2 NODE NETWORK - DEMO

- Connectivity - ping
- Transmissions
- Modulation parameters

```
> ping host('B')
PING 31
Response from 31: seq=0 rthops=2 time=2507 ms
Response from 31: seq=1 rthops=2 time=2852 ms
Response from 31: seq=2 rthops=2 time=2852 ms
3 packets transmitted, 3 packets received, 0% packet loss
> ping host('C')
PING 74
Response from 74: seq=0 rthops=2 time=2600 ms
Response from 74: seq=1 rthops=2 time=2634 ms
Response from 74: seq=2 rthops=2 time=2737 ms
3 packets transmitted, 3 packets received, 0% packet loss
```

A UNIVERSAL SDM

- The variety of languages for accessing Unetstack includes C, Java, Python, Julia etc.
- Makes it easily integrated to existing software systems
- Users can replace all existing agents for different layers too
- Thus we feel overall, it is a good platform to discuss our proposed topic of Practical Underwater networks without loss of generality

A SIMPLE AGENT

- Echo Daemon

```
class EchoDaemon extends UnetAgent {

  @Override
  void startup() {
    // subscribe to all agents that provide the datagram service
    subscribeForService(Services.DATAGRAM)
  }

  @Override
  void processMessage(Message msg) {
    if (msg instanceof DatagramNtf && msg.protocol == 35) {
      // respond to protocol USER datagram
      send new DatagramReq(
        recipient: msg.sender,
        to: msg.from,
        data: msg.data
      )
    }
  }
}
```

ADDRESSING

- Node Addressing
- ports
- protocol numbers
  - Protocol.USER (32) to Protocol.MAX (63) for user applications
  - In later sections on sockets concept, this will be elaborated

USING AUDIO

- Using PC sound

UPCOMING SESSIONS

- Multihop Routing - Prasad
- Sensors and the Internet - Chinmay
- Localization - Prasad

AN ILLUSTRATIVE UNET