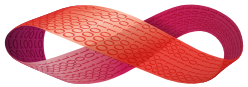
Fixing Erlang's Distribution Protocol

Peer Stritzinger, Erlang User Conference
Stockholm, June 2017



DIPL. PHYS. PEER STRITZINGER GMBH

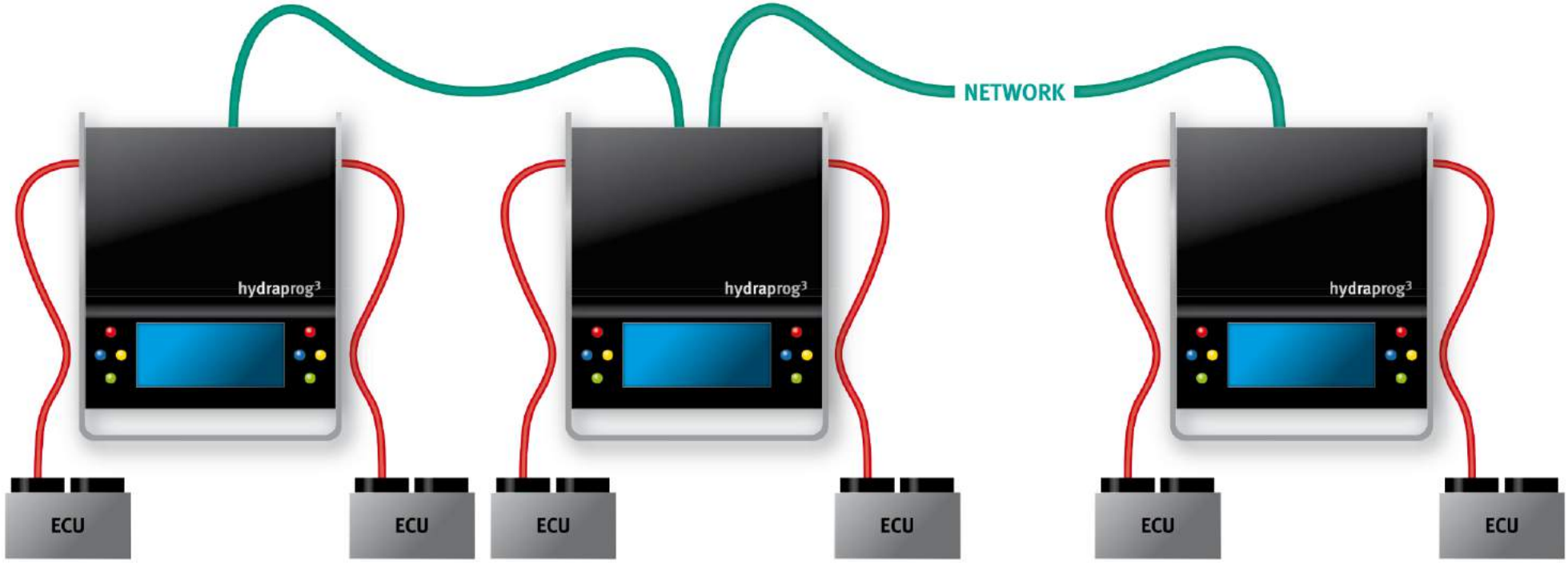
LIGHTKONE

Lightweight computation for networks at the edge


hydraprog3
hydraprog3


X1
X2
HP3 GW-LCFM
X3

Self-networked units
= multi channel virtual device



DIPL. PHYS. PEER STRITZINGER GMBH

Application

- Next Generation RFID System
- Industry 4.0, Smart Factory System
- Distributed data sharing and material routing in Erlang
- Programmability by PLC Programmers
- Research Project: Cyber-physical IT-Systems to handle the complexity of a new Generation of multi adaptive Factories

GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung
DLR
PT-SW des BMBF


ERLANG
GRiSP
Embedded With
RTEMS
www.rtems.com


24 Vout
24 Vin
ETH 3
ETH 2
ETH 1
PWR/SYS
Rexroth
Bosch Group
ATC/BI0-1HF
3 842 410 506
RFID
137
110


GeBIT
future talk
GeBIT

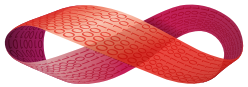

Rexroth

DIPL. PHYS. PEER STRITZINGER GMBH

DIPL. PHYS. PEER STRITZINGER GMBH

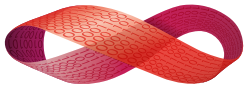
Scaling Solutions

- SD Erlang
- Hidden nodes
- Ditching Erlang Distribution
- Fixing Erlang Distribution



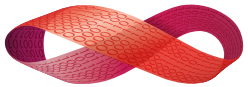
Hidden nodes

- Connect only when communicated to
- „Invisible“ to to normal nodes() so won’t get connected by global e.al.
- Start hidden node: **erl -hidden**
- Use **nodes/1** function to query
- Connected hidden nodes can be monitored



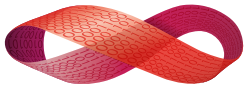
Hidden Nodes (cont.)

- erlang:send/3 option noconnect
- Sending to {*name*, *node*}
- All of net_kernel
- Topology and routing on application level



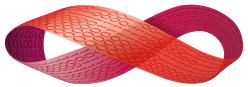
Fixing Erlang Distribution

- Fully connected network can't scale
- Head of line blocking of large messages
- Possible concurrency issues
- Hard Realtime networking
- Security in hostile networks



OTP Team ongoing

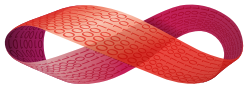
- Heterogenous
- Distributed Hash Table to replace
- Link management



epmd

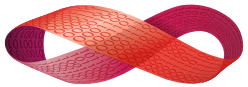
- Some epmd support or replacement is possibly needed for other protocols
- Pure Erlang epmd implementation eases extending
- Already used to ease small embedded systems integration like http://www.grisp.org/
- Adopted by OTP team

https://github.com/erlang/epmd

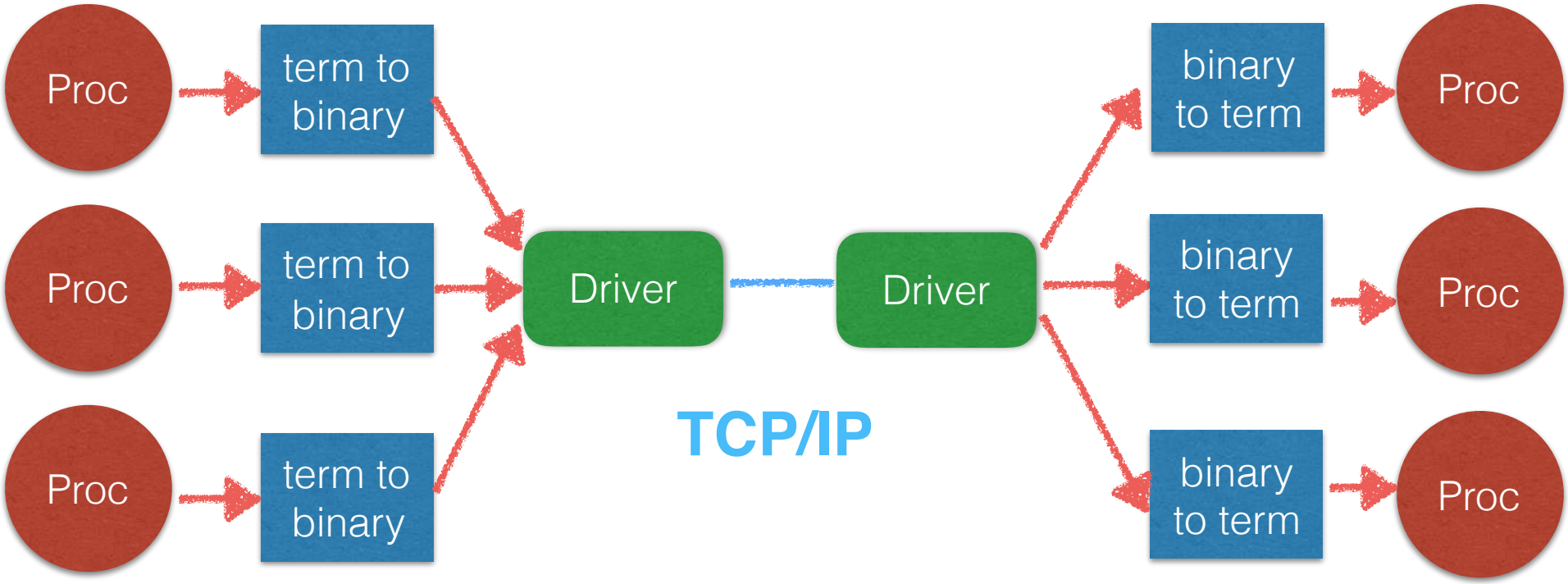


Plug in another distribution transport protocol

- Command line **-proto_dist** *mod* makes distribution call *mod_dist*
- Module implementing: childspecs/0, listen/1, accept/1, accept_connection/5, setup/4, close/1, select/1, is_node_name/1
- Port driver also needed, is called by erts_schedule_dist_command() see otp/erts/emulator/beam/dist.h
- More details http://www.erlang.org/doc/apps/erts/alt_dist.html



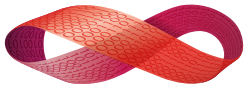
Node to Node Link



Demo

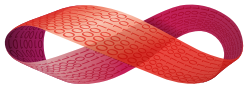
```
#define ERTS_DIST_MSG_DBG

#define ERTS_RAW_DIST_MSG_DBG
```



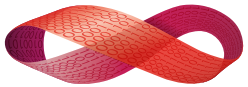
Message Routing

- Direct TCP connections to all nodes are a scalability hindrance
- Dynamic connection management can help some applications
- Forwarding received messages towards another connected node

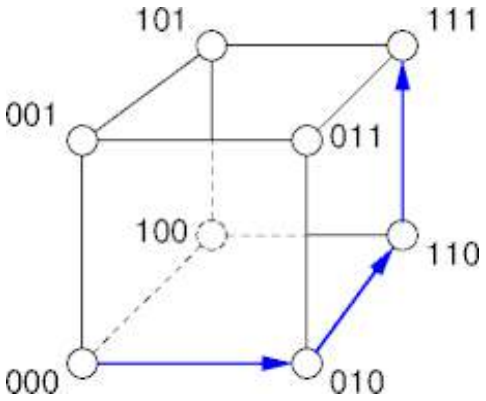


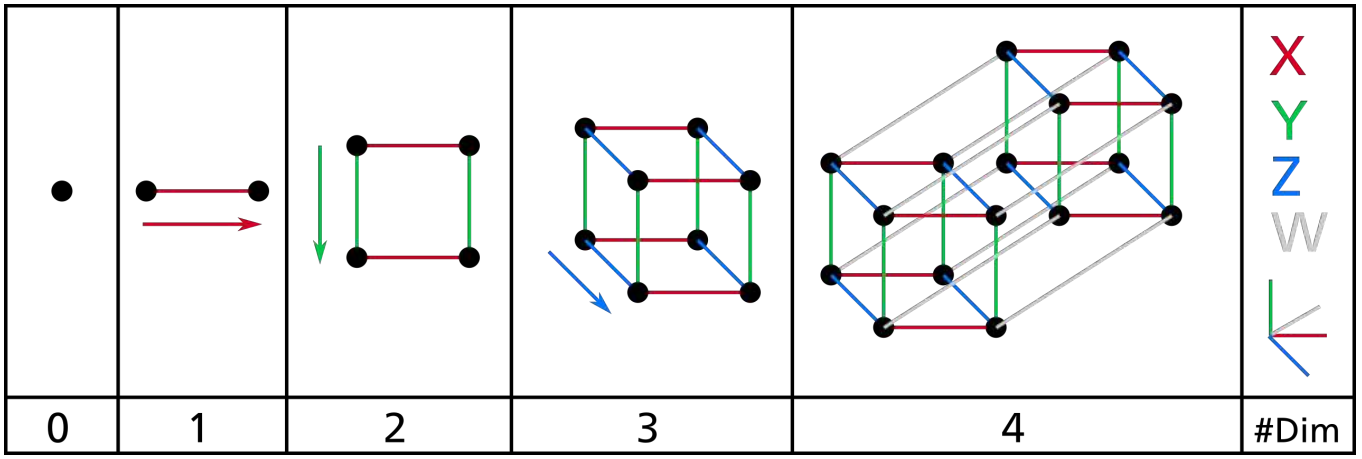
Message Forwarding

- Routing table lookup
- Filled and updated from the Erlang level
- Used for quick lookup inside ERTS
- Can be used for heterogenous link decision too



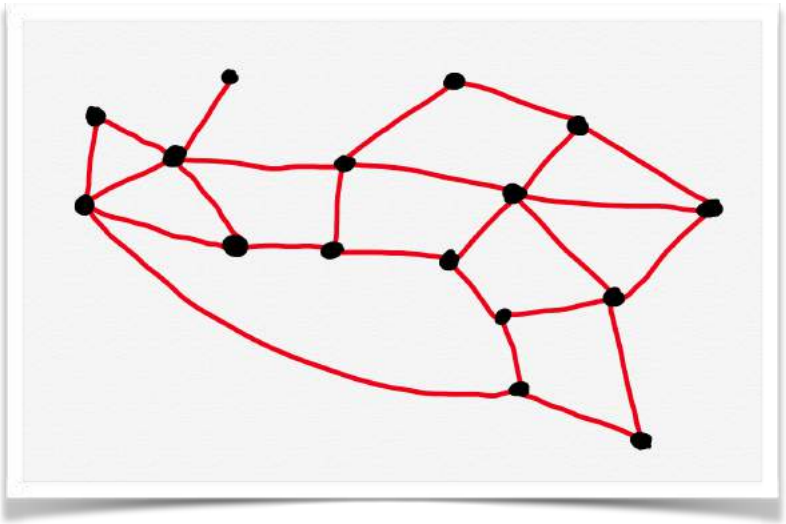
Static routing





DIPL. PHYS. PEER STRITZINGER GMBH

Routing protocols



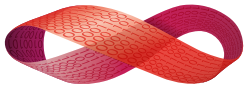
- Discover network graph

  topology

- Shortest path
- Minimal weighted paths
- Detect changes, update topology and distribute to nodes

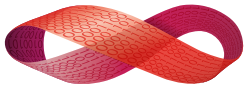
Link State Protocols

- Every node floods network with neighbor topology
  - Link State Packets (LSP)
- When converged every node knows the whole network connection graph
- Every Node calculates Minimum Distance Spanning tree = shortest Route to every other node
- Packet Forwarding according to this



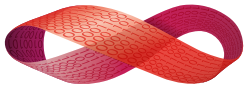
Variants in use

- Inter System to Inter System (IS-IS)
  - ISO/IEC 10589:2002
- Open Shortest Path First
  - RFC2328 (v2 for IPv4)
  - RFC5340 (vs for IPv6)



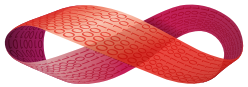
Routed Messages

- Can get out of order
- Can get lost
- Can go in circles



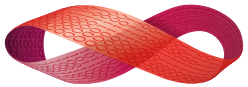
Invariants to be observed

- Any message from a node will be after nodeup and before nodedown
- Messages between any process pair need to stay ordered



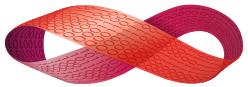
Requirements

- Reordering messages between two processes
- Detecting messages lost
- Drop messages that go in circles



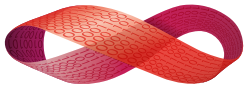
Sequence Numbers

- Separate counter per {From, To} pair
- Buffer message that are ahead
- Missing message is detected after a timeout
- Old messages that arrive after newer ones are consumed are dropped



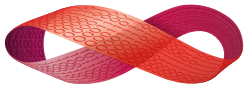
Message Loss?

- Backwards compatible semantics
  - Trigger a nodedown+nodeup on loss
- Optionally different semantics
  - Tell process there were lost messages



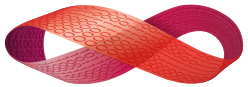
Message Sending Options

- Out of order messages
- Unreliable messages
- Hard realtime messages
- Options
  - `erlang:send` or per Process



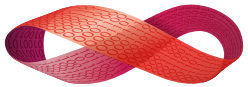
Head of Line Blocking

- Send large messages in chunks
  - Fragment numbers
  - Scheduling
- Use secondary TCP connections
  - But potential scalability issue here



Possible optimizations

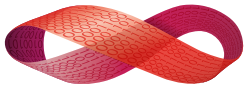
- Start sending the first chunks while encoding is ongoing
- Reassemble and decode in process context



LIGHTKONE

Lightweight computation for networks at the edge

- Hybrid Gossip Protocols
- CRDTs
- Computation at the edge



DIPL. PHYS. PEER STRITZINGER GMBH

16:00 **Mälarsalen**

Ditching the Data Center: How to Stop Worrying and Lov the Edge

Peter Van Roy

Professor at UCL and Coordinator of LightKone

Open for pre-order now
Delivery start end of June

Talk Adam Lindberg Fri 14:55

GRiSP

www.grisp.org
http://www.stritzinger.com

@peerstr @grisporg

https://github.com/grisp

DIPL. PHYS.
PEER STRITZINGER GMBH