a bare minimum runtime platform for ROS 2 nodes in Elixir

Hideki Takase, Shintaro Hosoai (The Univ. of Tokyo), Ryota Kinukawa (Tombo Works), Shunsuke Kikuchi (SAKURA internet, Inc.)

A part of this work is supported by JST CREST JPMJCR21D2 and the commissioned research (04001) by National Institute of Information and Communications Technology (NICT), Japan.
Hobby

Affiliation

Recent R&D

Mission

@takasehideki

Cutting-Edge Platform and Design Methodology for embedded/IoT Computing

Optimization  Methodology

a design infrastructure that connects all Things and Human

Toward a world where anyone can easily create awesome products

1,659 contributions in the last year
Topic: Rclex on Nerves

Functional Language

Client Library for Elixir

bare minimum and resilient runtime platform

1..1000
- Flow.from_enumerable()
- Flow.map(& foo(&1))
- Flow.map(& bar(&1))
- Enum.to_list
- Enum.sort

Lab#8, IPC, IST, UTokyo Computing System Laboratory
is awesome!!

- rclcpp
- rclpy
- rclrs

RCL (ROS Client Library)

RMW (ROS MiddleWare)

DDS A

DDS B

General Purpose OS

- common functions for ROS 2
- implemented by C
- any client library can be implemented by calling it

- Abstraction layer for DDS
- Able to select preferred DDS (and Zenoh??)

DDS (Data Distribution Service)

- OMG standard specification
- Pub/Sub communication functions
- Automatic search of nodes by RTPS
ROS 2 Client Library for

rclcpp
rclpy
rclrs
node.cpp
node.py
node.rs
node.ex

RCL (ROS Client Library)

RMW (ROS MiddleWare)

DDS A
DDS B
Zenoh??

General Purpose OS

github.com/rclex
We love **Elixir!!**

**Functional language** (appeared in 2012)

Operated on Erlang VM (BEAM)
- lightweight processes with robustness
- highly concurrency/parallelism
- soft real-time feature
- easy to realize distributed and fault tolerance system

- Similar to **Actor Model**
  - Actors (processes) send and receive messages
  - "Let it Crash": The problematic process should be promptly crashed and restored immediately
  - We can spawn a massive number of processes!!

* The copyrights of these logos belong to their respective creators
We love Elixir!!

Functional language (appeared in 2012)

Language Design based on Ruby
• easy to learn! familiar grammar and notation
• very highly productive despite functional
• high communication performance suitable for IoT

example Elixir code

```
1..1000
| > Flow.from_enumerable()
| > Flow.map(& foo(&1))
| > Flow.map(& bar(&1))
| > Enum.to_list
| > Enum.sort
```

Programming should be about transforming data

- Data flow and parallel processing can be described intuitively with
  `Enum Flow |>`
We love Elixir!!

• Awesome ecosystem!

* The copyrights of these logos belong to their respective creators
• Actors on BEAM send and receive messages
  – (I believe) Actor Model is suitable for robot systems
Simple Publisher by Rclex

```ex
defmodule RclexNode do
  def pub(num_node) do
    Rclex.rclexinit()
    |> Rclex.create_nodes('rclex_node', num_node)
    |> Rclex.create_publishers('chatter', :single)
    |> Rclex.Timer.timer_start(1000, &func/1)
  end

  def func(publisher_list) do
    n = length(publisher_list)
    msg_list = Rclex.initialize_msgs(n, :string)
    Enum.map(0..(n - 1), fn index ->
      data = "Hello World from rclex_node_" <> to_string(index)
      IO.puts("publish message: #{data}")
      Rclex.setdata(Enum.at(msg_list, index), data, :string)
    end)
    Rclex.Publisher.publish(publisher_list, msg_list)
  end
end
```

- create node by name and index
- number of nodes
- create topic
  - :single-> one topic
  - :multi-> multiple
- set callback func and timer interval
- create list of messages for each node
- publish message

setting node info. with pipe operator

processing a data message with Enum
Example of Rclex on Native Ubuntu
Developed by **Nerves Project**!

- **Dedicated IoT Platform for Elixir**
  - fully compatible with Elixir
  - useful libraries for IoT systems
  - **bare-minimum!!**
    - too small (one linked fw, ~>30 MB)
    - portable and easy to configure
  - **resilient!!**
    - of course, robust thanks to Erlang VM!!
    - file system is duplicated and read-only for reliability

[GitHub link: github.com/rclex]
**Supported Targets**

<table>
<thead>
<tr>
<th>ROS_DISTRO</th>
<th>arm64v8</th>
<th>arm32v7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foxy</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
How to try it out?

- Only 7 steps, just in 5 min (I hope,,,:D
  1. create & prepare project for target
     $ mix nerves.new rclex_on_nerves --target rpi4
  2. Install Rclex into Nerves project
     add {:rclex, "~> 0.9.2"}, into mix.exs and $ mix deps.get
  3. prepare ros2 resources to Nerves filesystem
     $ mix rclex.prep.ros2 --arch arm64v8
  4. generate resources for msg types of topic comm.
     $ mix rclex.gen.msgs
  5. set LIBRARY_PATH into Nerves firmware
  6. enjoy writing Rclex code!!
  7. finally, build firmware, and burn to microSD

github.com/rclex
• For ROS developers
  – will acquire performance, especially in communication
  – will want to employ Elixir/Erlang ecosystem
  – will get a **bare minimum and resilient environment!!**

• For Elixir developers
  – will get autonomous communication
  – will utilize OSS packages published for ROS!!

---

MAY THE **BEAM**
BE WITH YOUR ROBOT!!

---

https://aliexpress.com/item/
https://aliexpress.com/item/
https://aliexpress.com/item/32604221183.html

---

**github.com/rclex**
論よりRUN!! "ron yori run"
The RUN is mightier than the word

https://github.com/pojiro/rpi4_mouse

Raspberry Pi Mouse V3 by RT Corporation

Momo (WebRTC Native Client) by Shiguredo
論よりRUN!! "ron yori run"
The RUN is mightier than the word
One More Thing??

**Rclex on Nerves with Zenoh**

**Zenoh**

- Eloquent, Scalable, and Fast network protocol
- The most promising candidate for a new Tier-1 RMW??

**Our Trial**

- Rclex msgs on Nerves go beyond the Pacific Ocean!!
  - Code BEAM America 2023
    - [Slide] [YouTube] [demo(on X)]
  - (WiP) Zenohex [GitHub]

![Zenohex Diagram]
Wrapping Up

• **Rclex**: Robotics meets Elixir!
  – **ROS** will acquire **Elixir**’s performance
  – **Elixir** will utilize **ROS** packages

• Ongoing/upcoming works
  – Service, Action and Parameter
  – actively integration with Zenoh
  – documentation, academic paper,

**https://github.com/rclex/rclex**

**https://hex.pm/packages/rclex**

STAR our repo! try it out!!
and, Give your contributions!!!