Process Oriented Microservice with Elixir

Outline

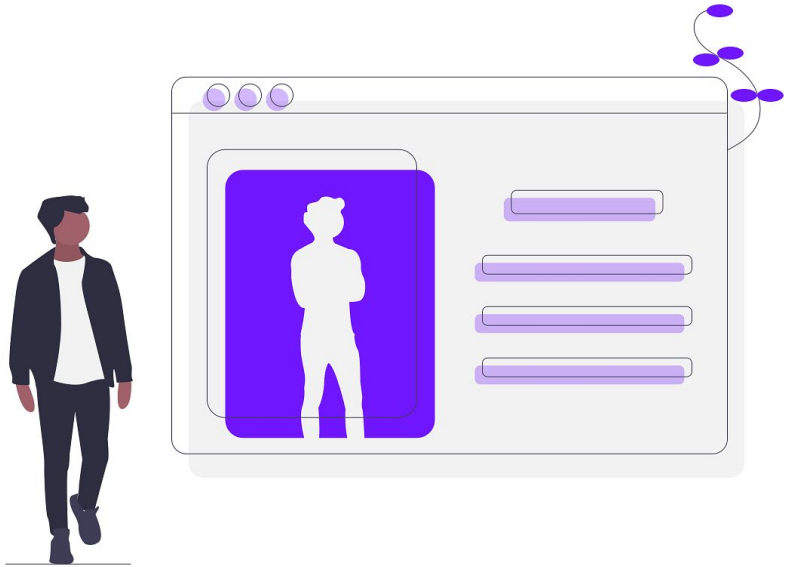
Outline

- A taste of elixir
- How distributed elixir works
- How elixir hold up compared to popular solution
- Discussing Cloud application from architectural pattern

About me

About me

- Software engineer at kumparan
- Helping young talent grow at Proclub
- Mentally exhausted at Telkom University



A Taste of Elixir

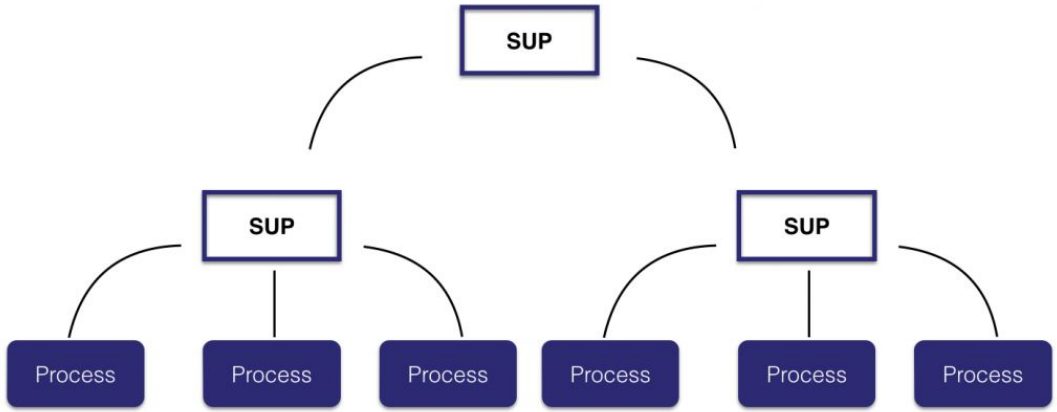
A taste of elixir

+

A taste of elixir

```
def handle_in("board:update", %{"board" => body}, socket) do
  IO.inspect(body)

  body
  |> Enum.map(&Cell.live/1)

  IO.inspect(Cell.Supervisor.children())

  broadcast!(socket, "update_view", %{"board" => body})
  {:reply, :ok, socket}
end
```

A taste of elixir



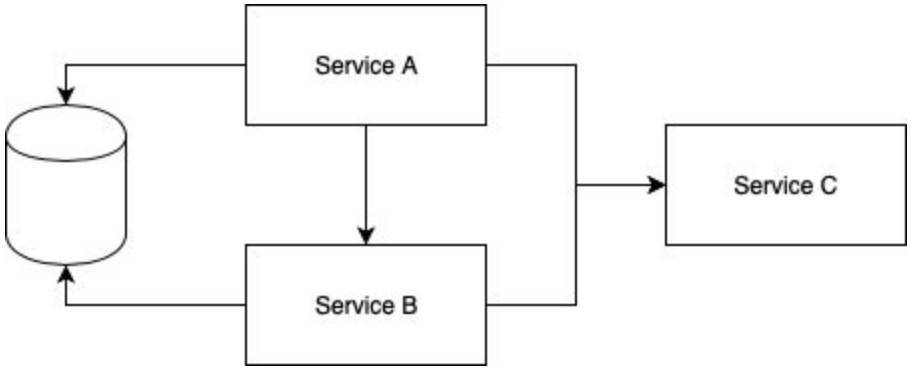
Supervisor

A supervisor is a process which supervises other processes, which we refer to as *child processes*. Supervisors are used to build a hierarchical process structure called a *supervision tree*. Supervision trees provide fault-tolerance and encapsulate how our applications start and shutdown.

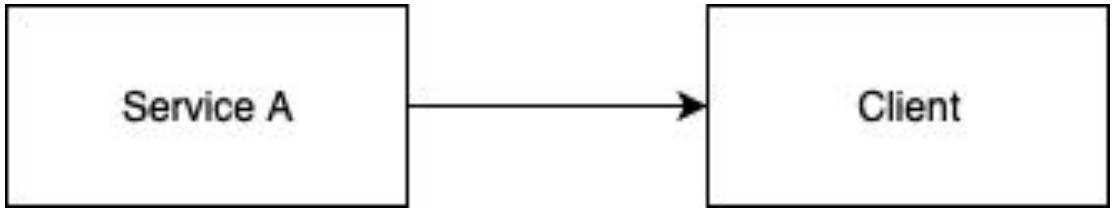
```elixir
defmodule Universe.Supervisor do
  use Supervisor

  def start(_type, _args) do
    start_link()
  end

  def start_link do
    Supervisor.start_link(__MODULE__, [], name: __MODULE__)
  end

  def init(_) do
    children = [
      worker(Universe, []),
      supervisor(Cell.Supervisor, []),
      supervisor(Registry, [:unique, Cell.Registry])
    ]

    supervise(children, strategy: :one_for_one)
  end
end
```

Mainstream Cloud App vs Elixir App

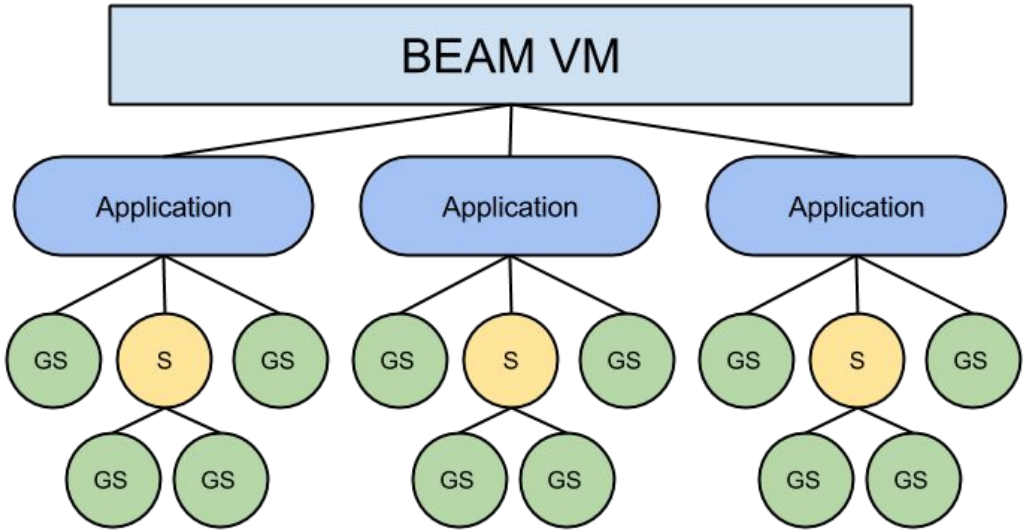
Microservice (Go)



Elixir App



Elixir App - OTP



Go vs Elixir

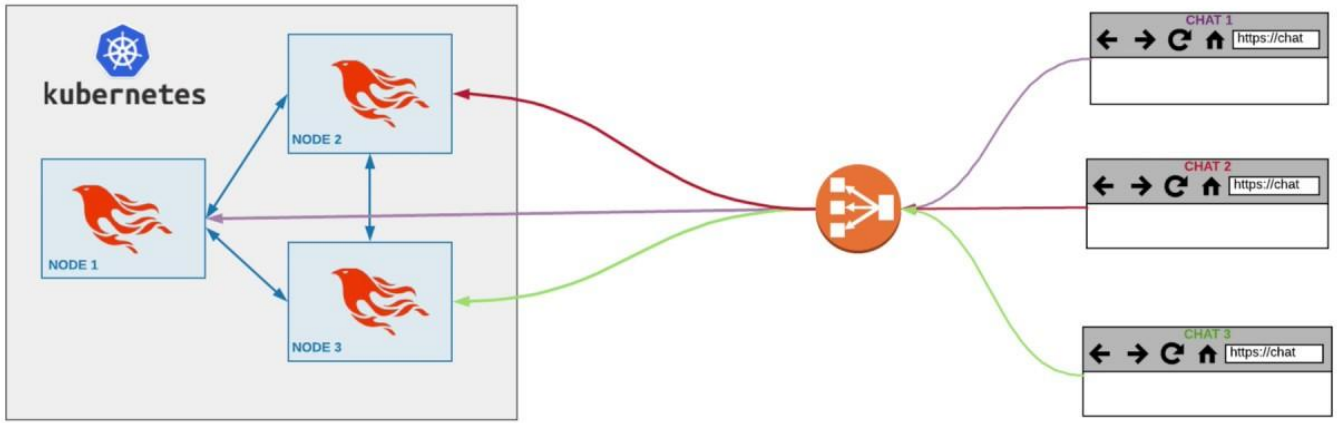
Go

- Built binaries = ridiculously easy deployment
- CPU and memory usage are generally very small (we build small services with Go)
- Excellent parallel processing performance
- Dependencies were originally challenging to manage (due to simplicity of model) but now everything is vendored, so it is not as much of an issue
- Standard libs cover a huge amount of use cases, meaning external dependencies can be minimized
- Ideal for small services or applications that run and exit often

Elixir/Erlang

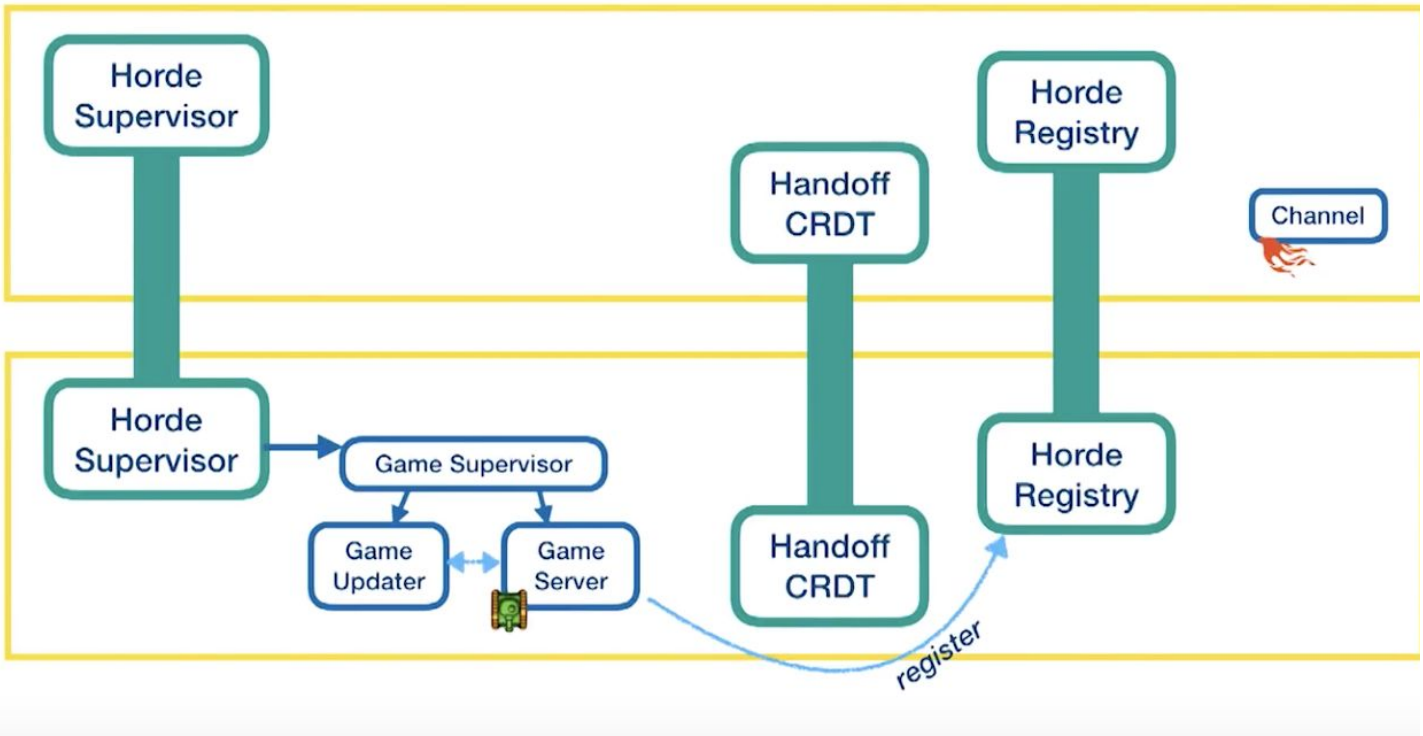
- Excellent dev tools (mix for example)
- Excellent runtime inspection (bind to a running service and start calling functions)
- Excellent parallel processing performance
- OTP is a powerful system for building long running services combined of multiple small parts (apps and libs)
- Fail fast design seems to result in more robust systems that are easier to reason about when problems occur
- Ideal for services that run for long periods of time

Extra: Clustered elixir



{poeticoding}

Extra: Clustered elixir



My Contact

- github.com/blinfoldking
- linkedin.com/in/ganeshad/

Futher Refrence

https://youtu.be/nLApFANtkHs

Thank you