THE ALCHEMIST’S CODE

BRINGING MORE VALUE WITH LESS MAGIC

The Pragmatic Programmer

from journeyman
to master

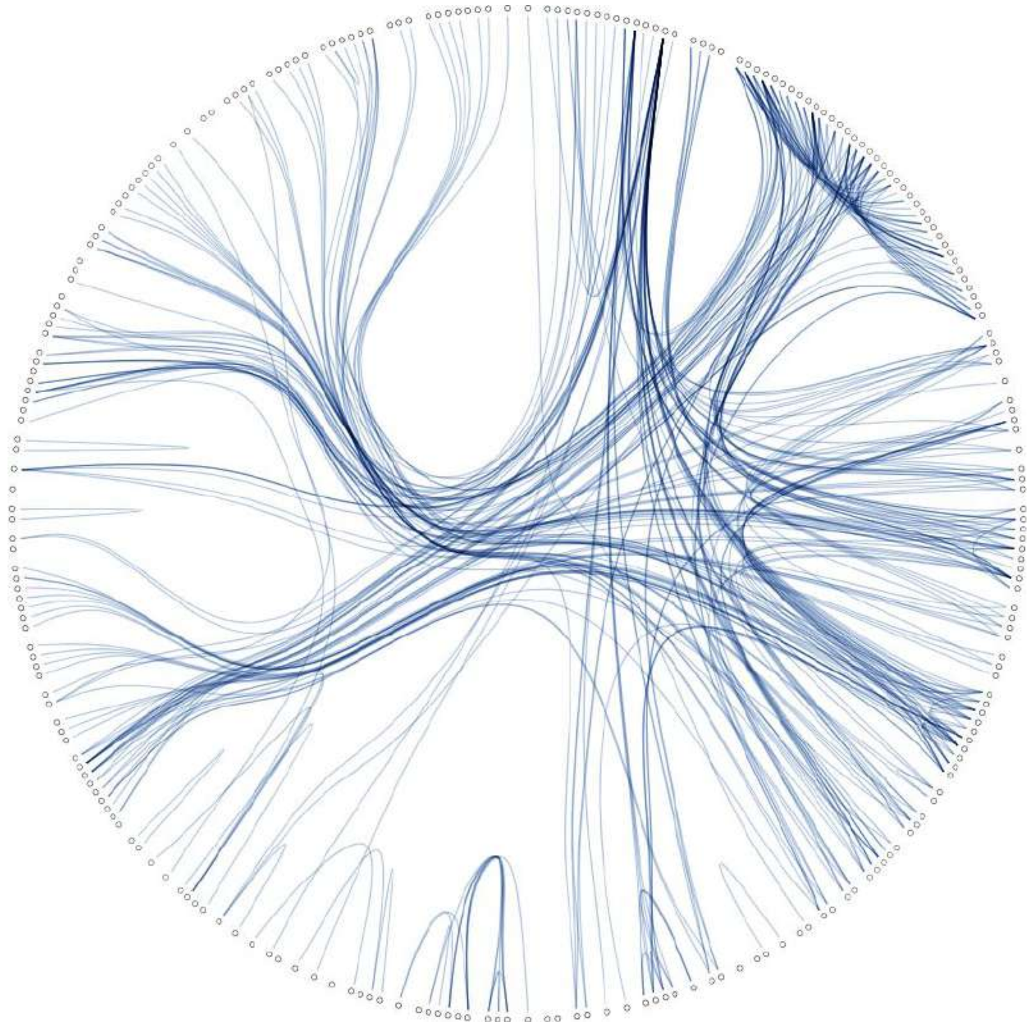
Andrew Hunt
David Thomas

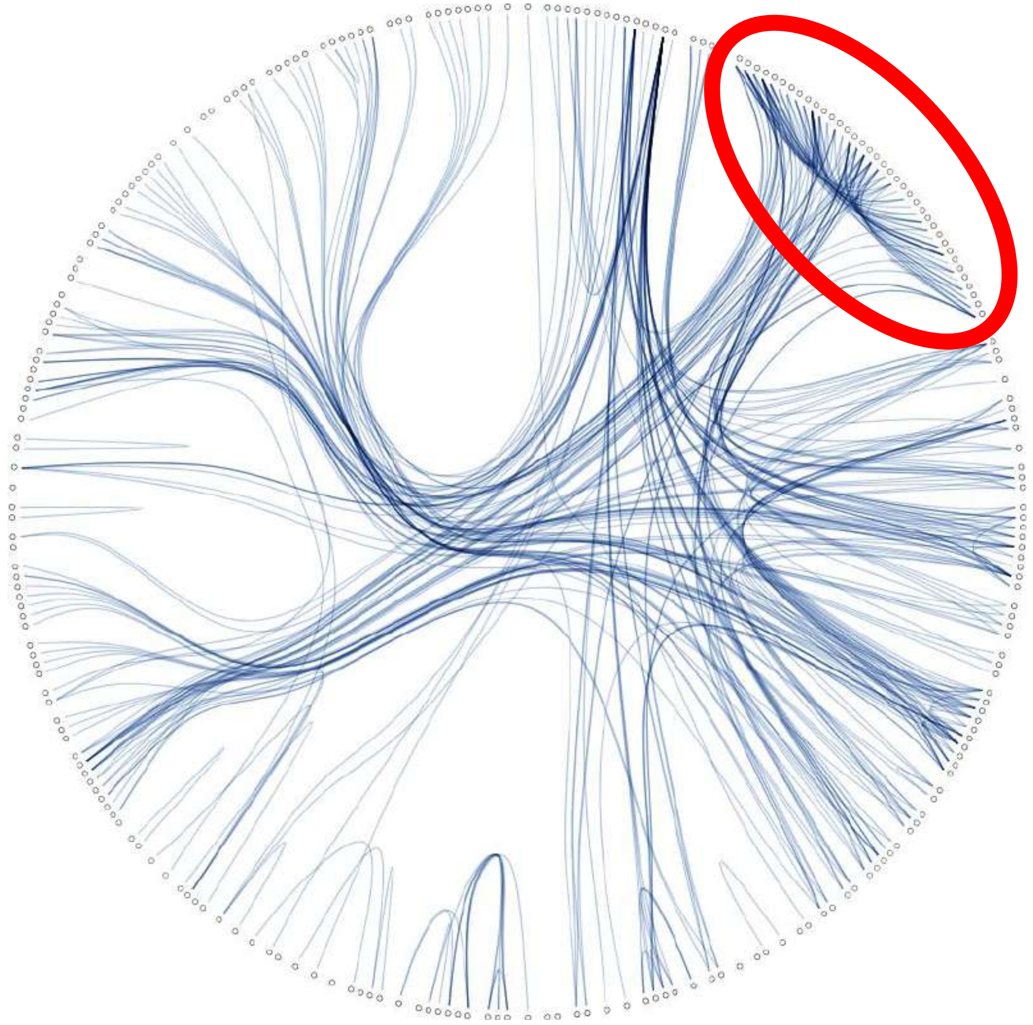
Foreword by Ward Cunningham

APPRENTICESHIP

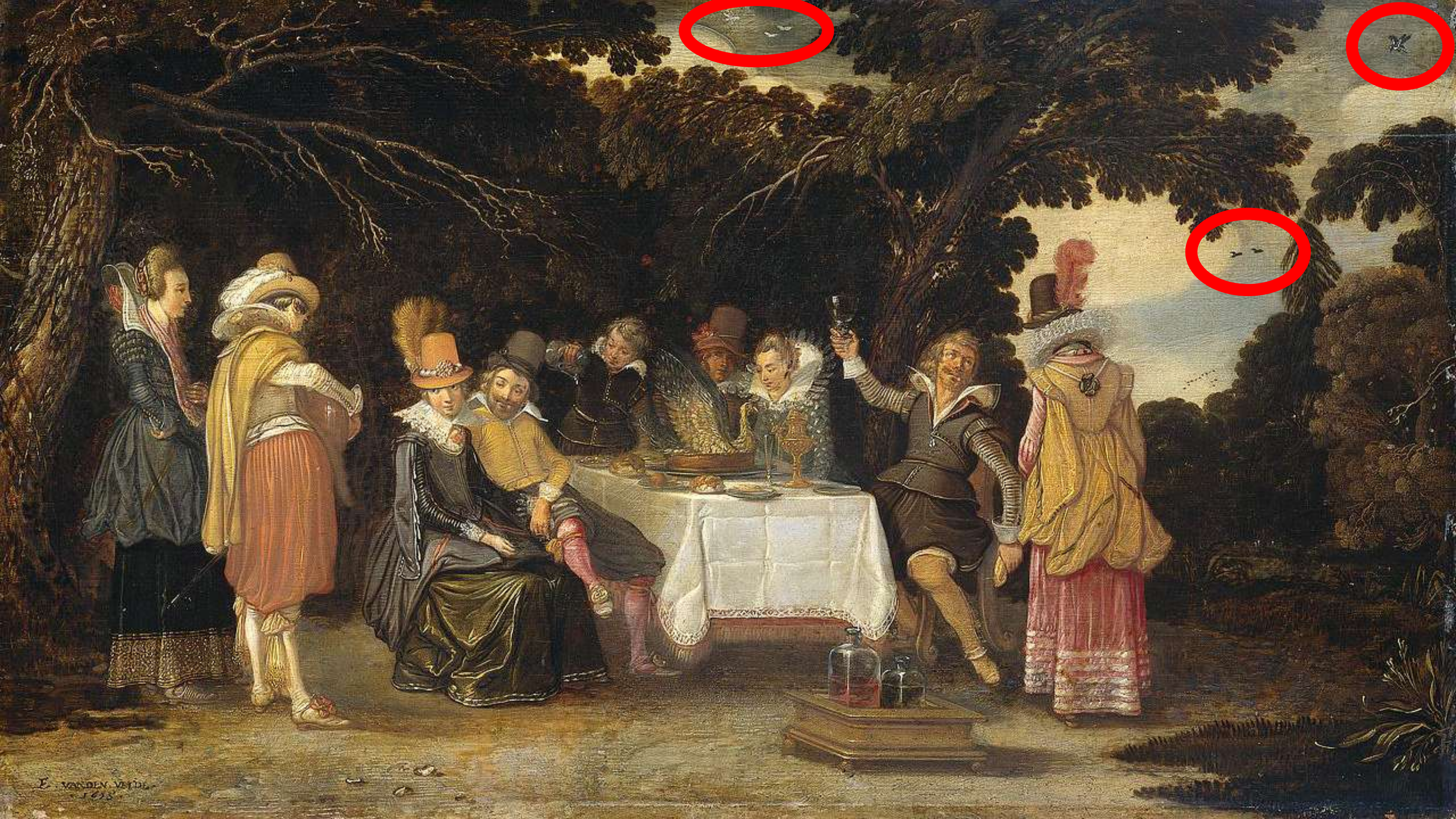
DIFFUSION

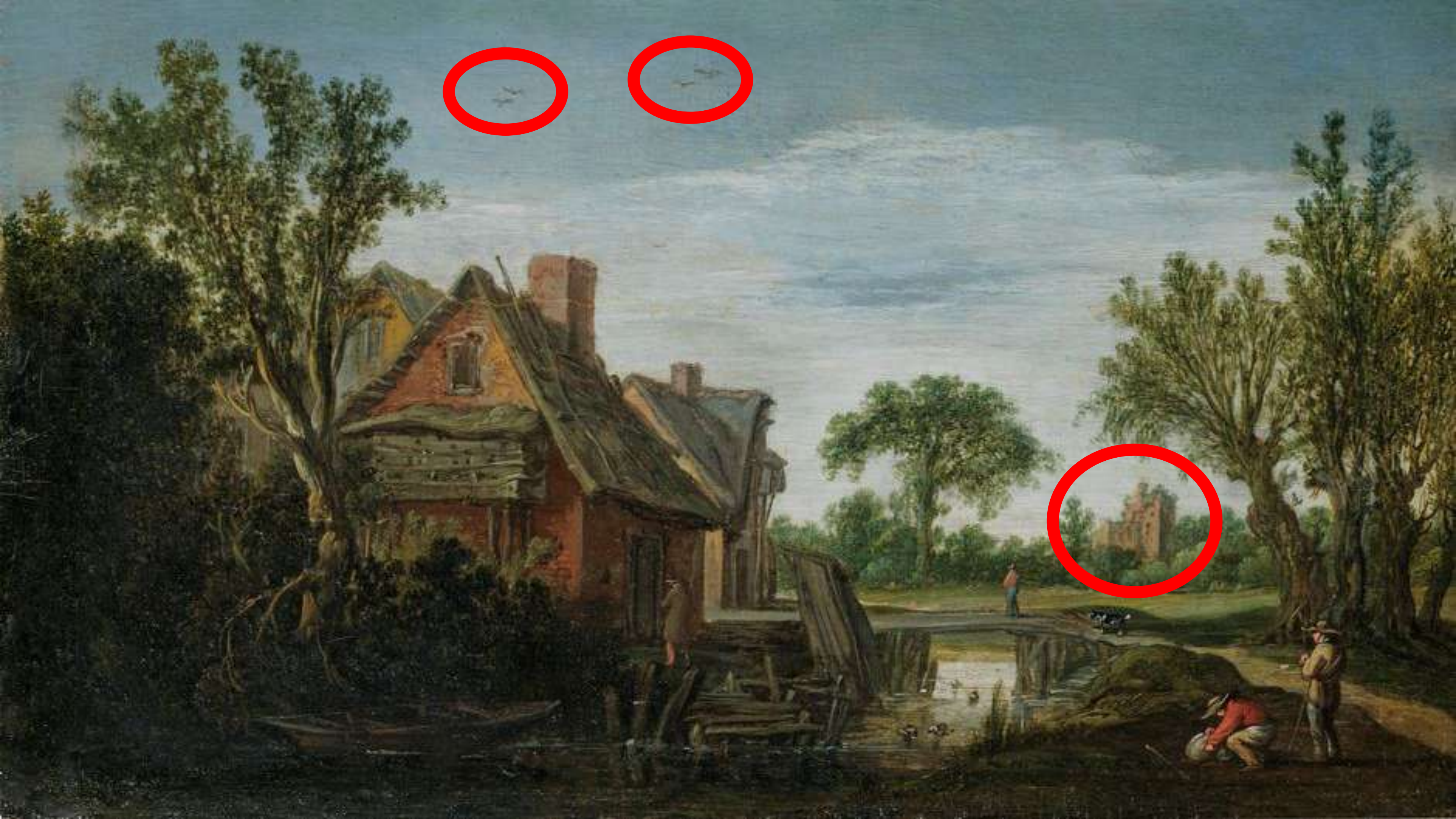
FRAMEWORKS

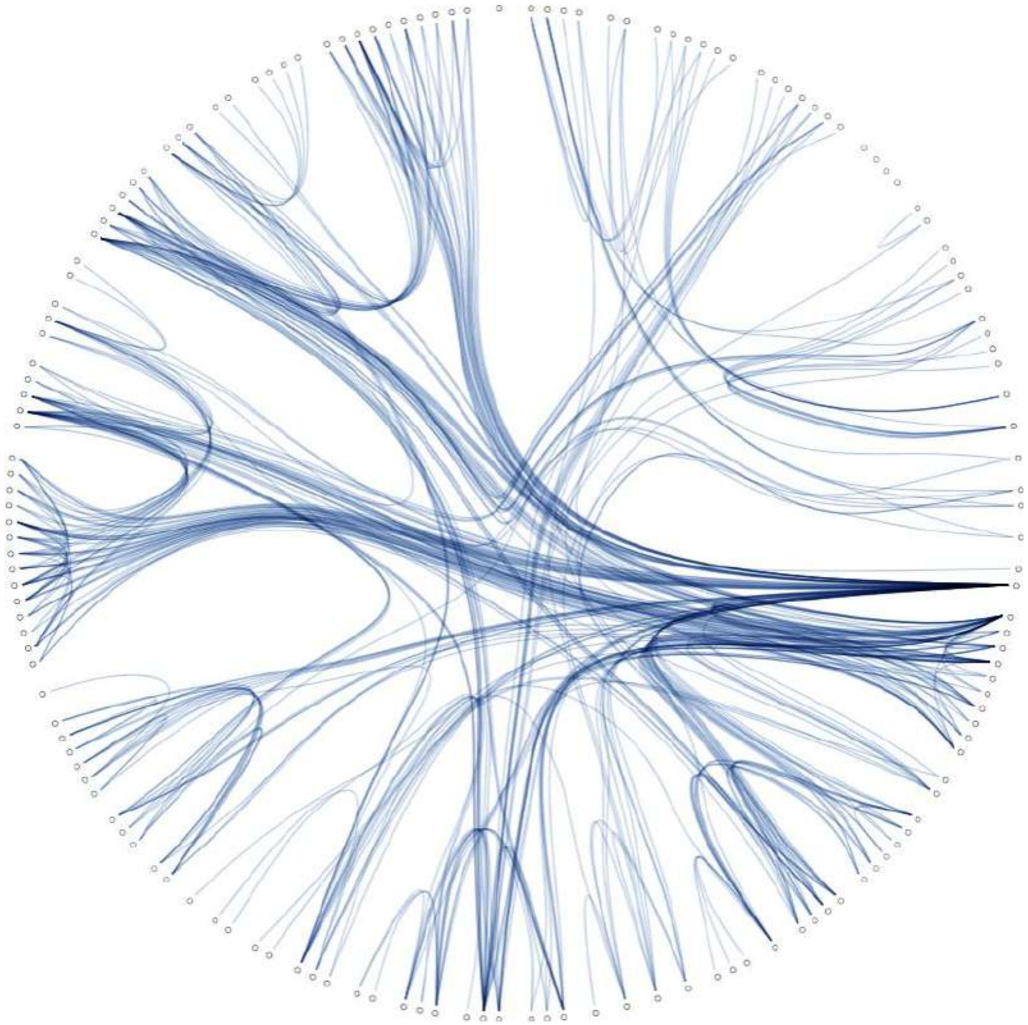


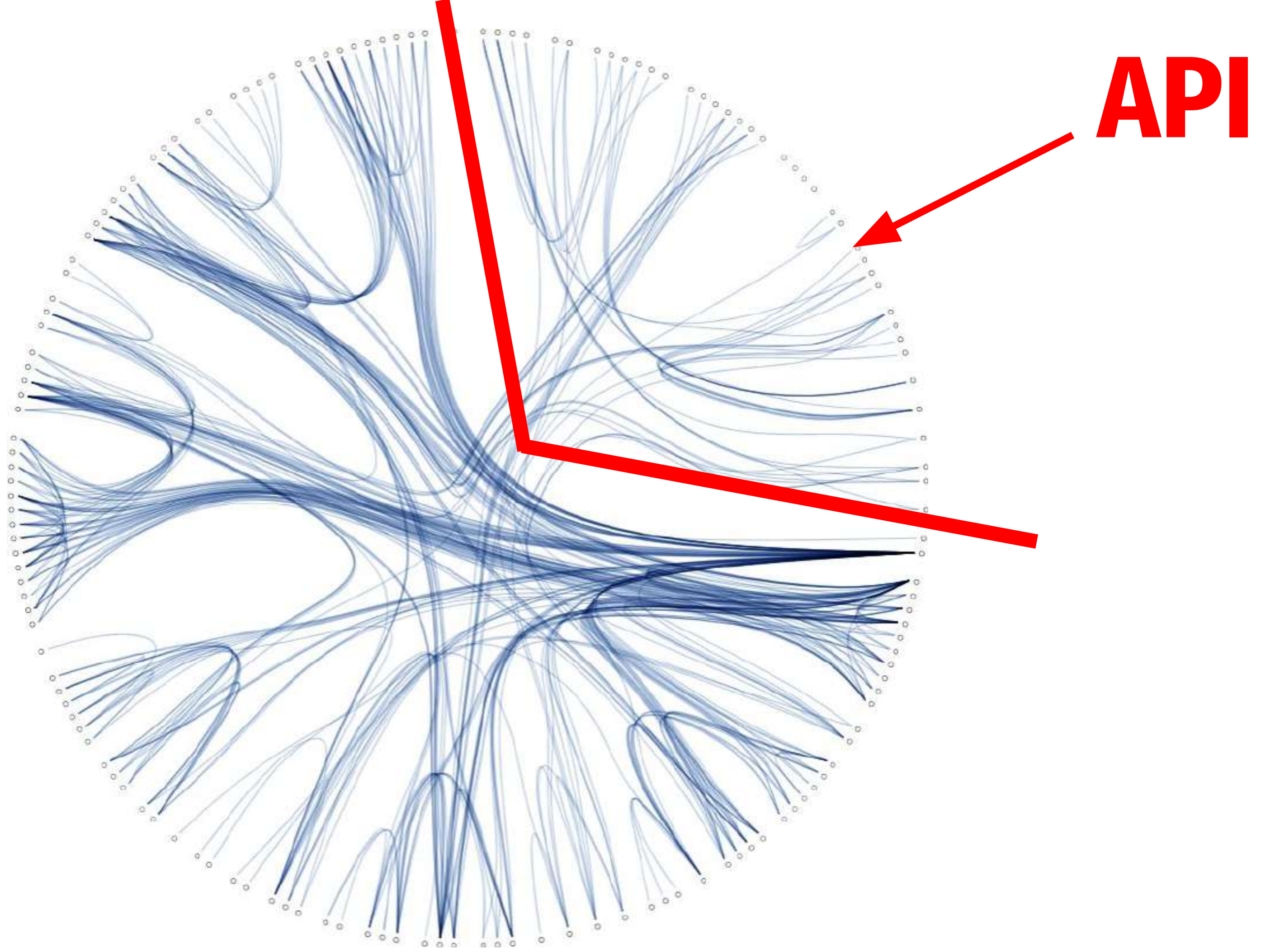


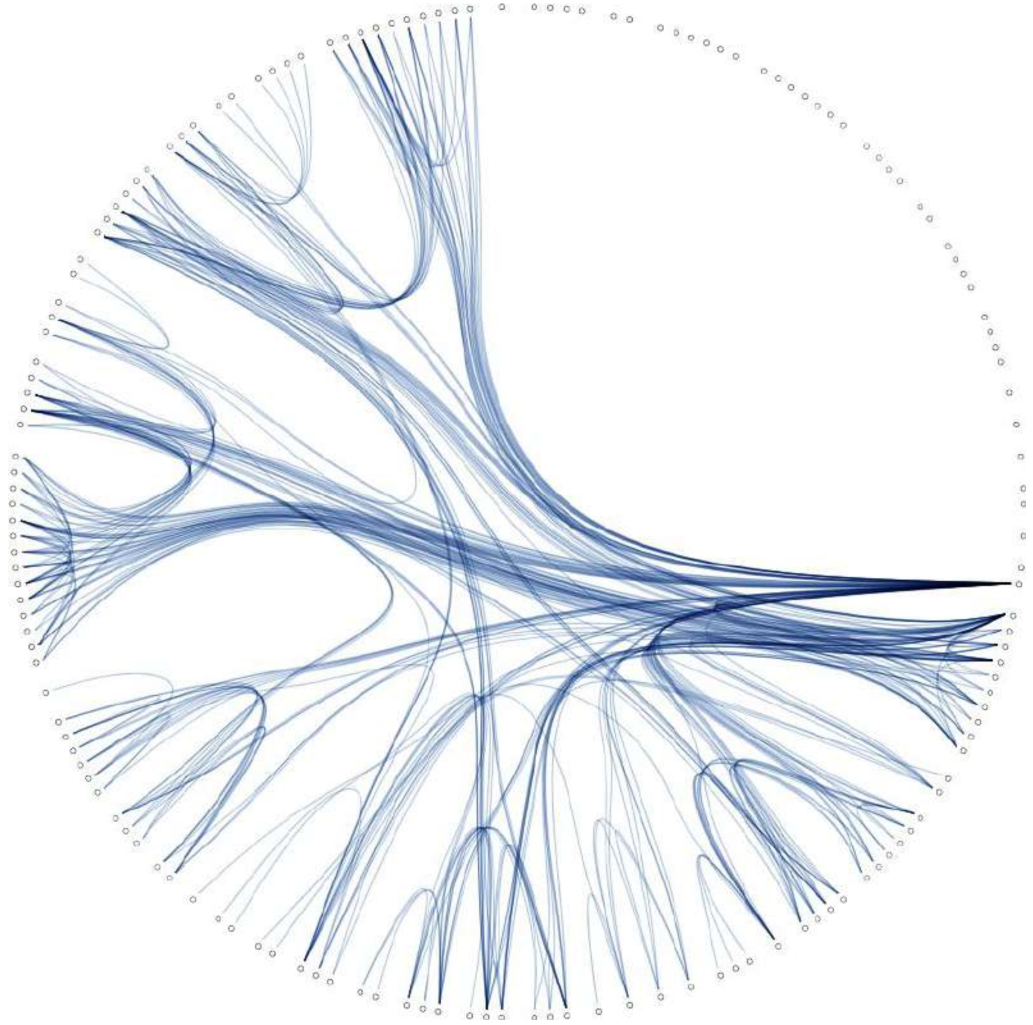
E. VANDEN VELDE
1615

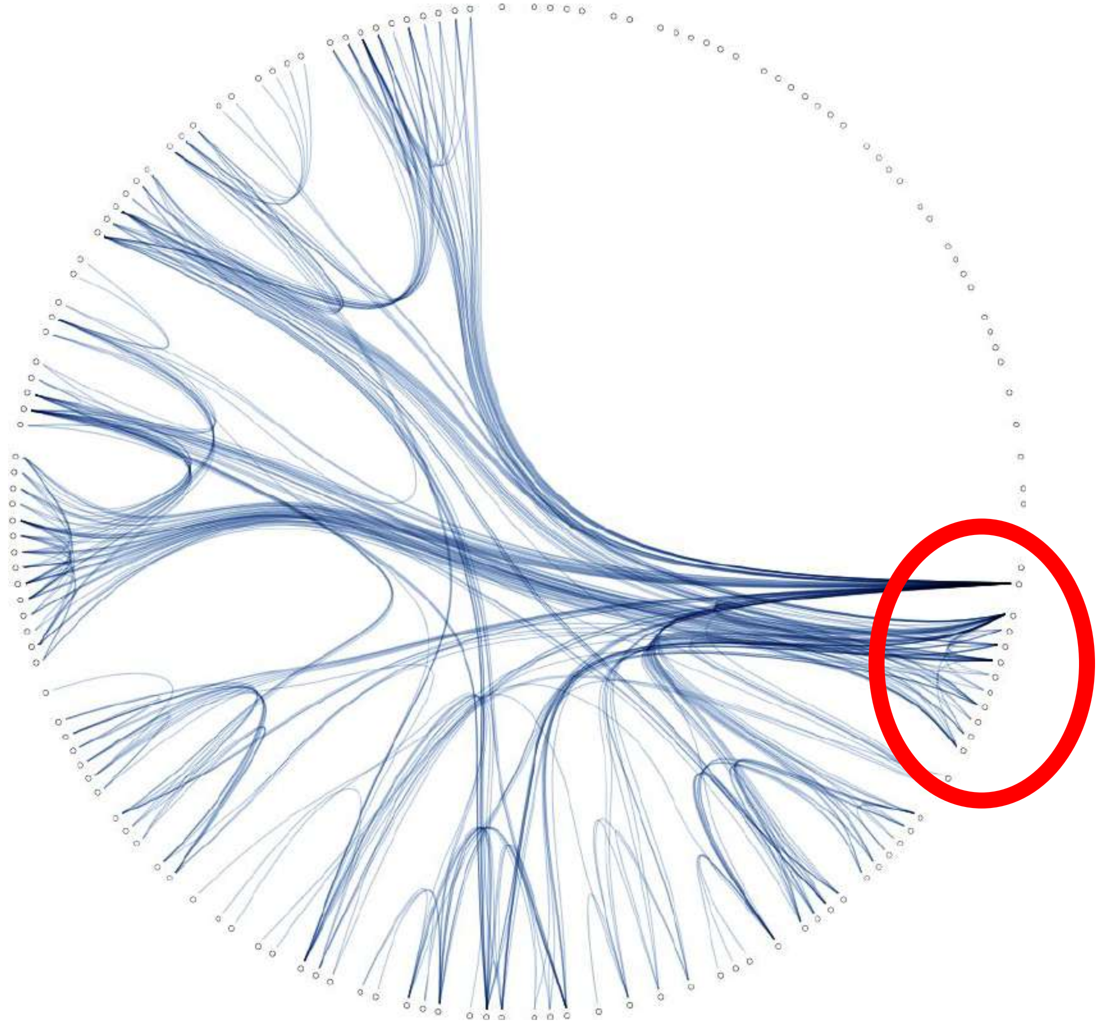

E. VANDEN VELDE
1615

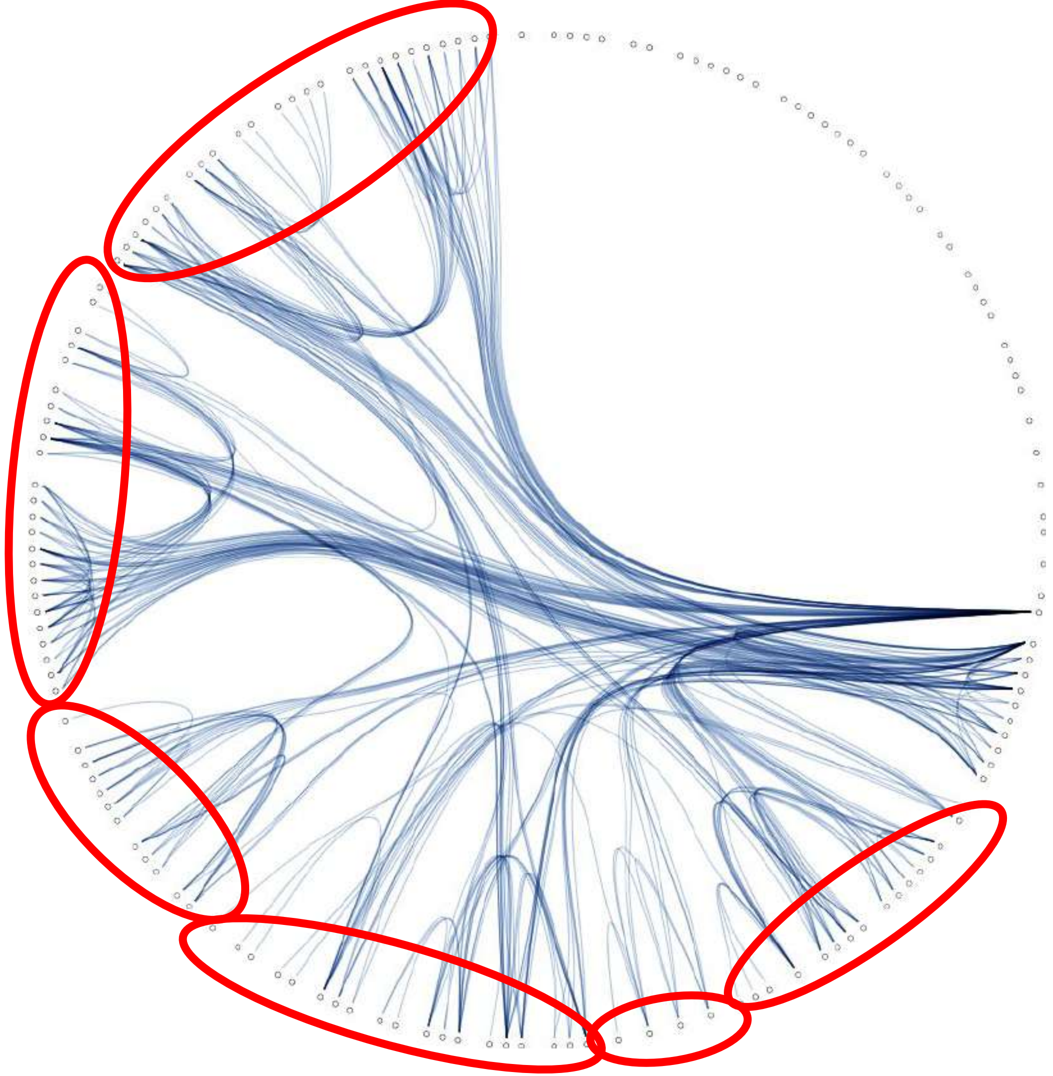





API





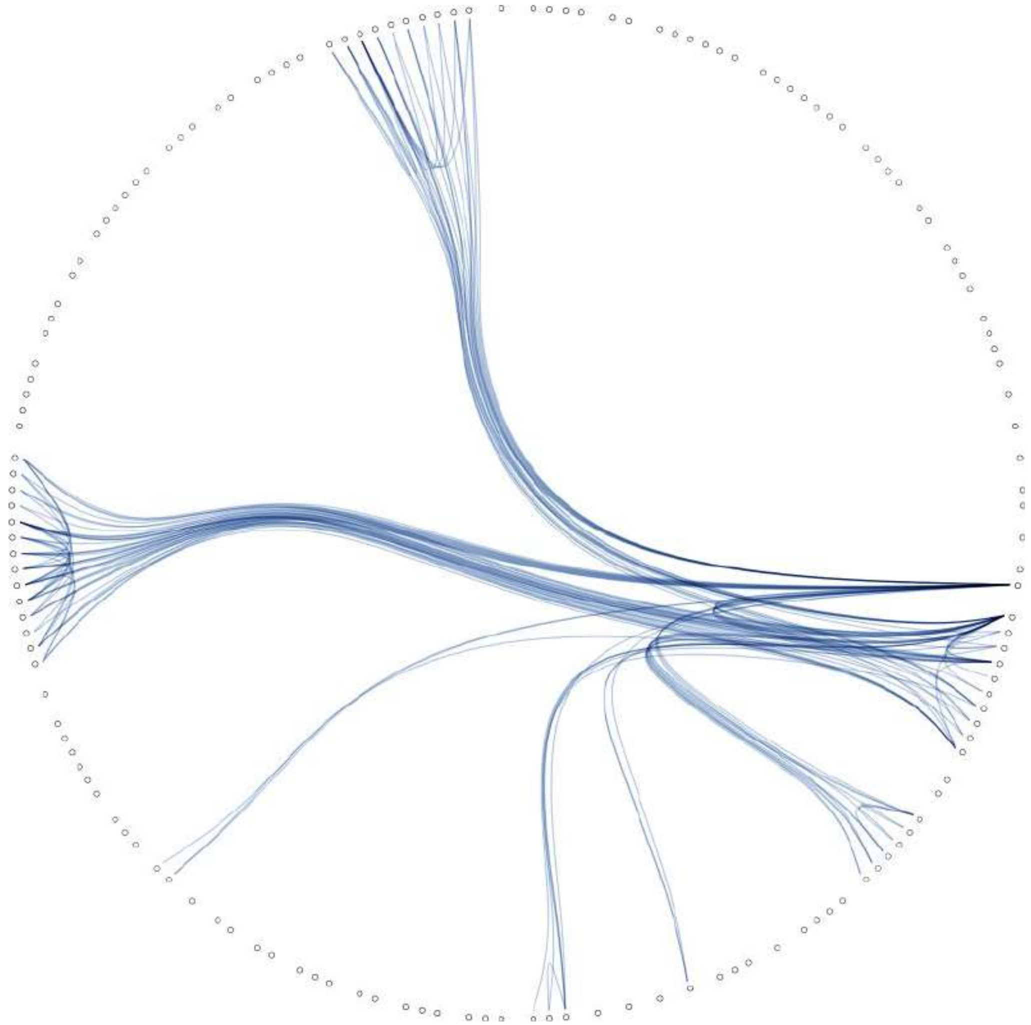


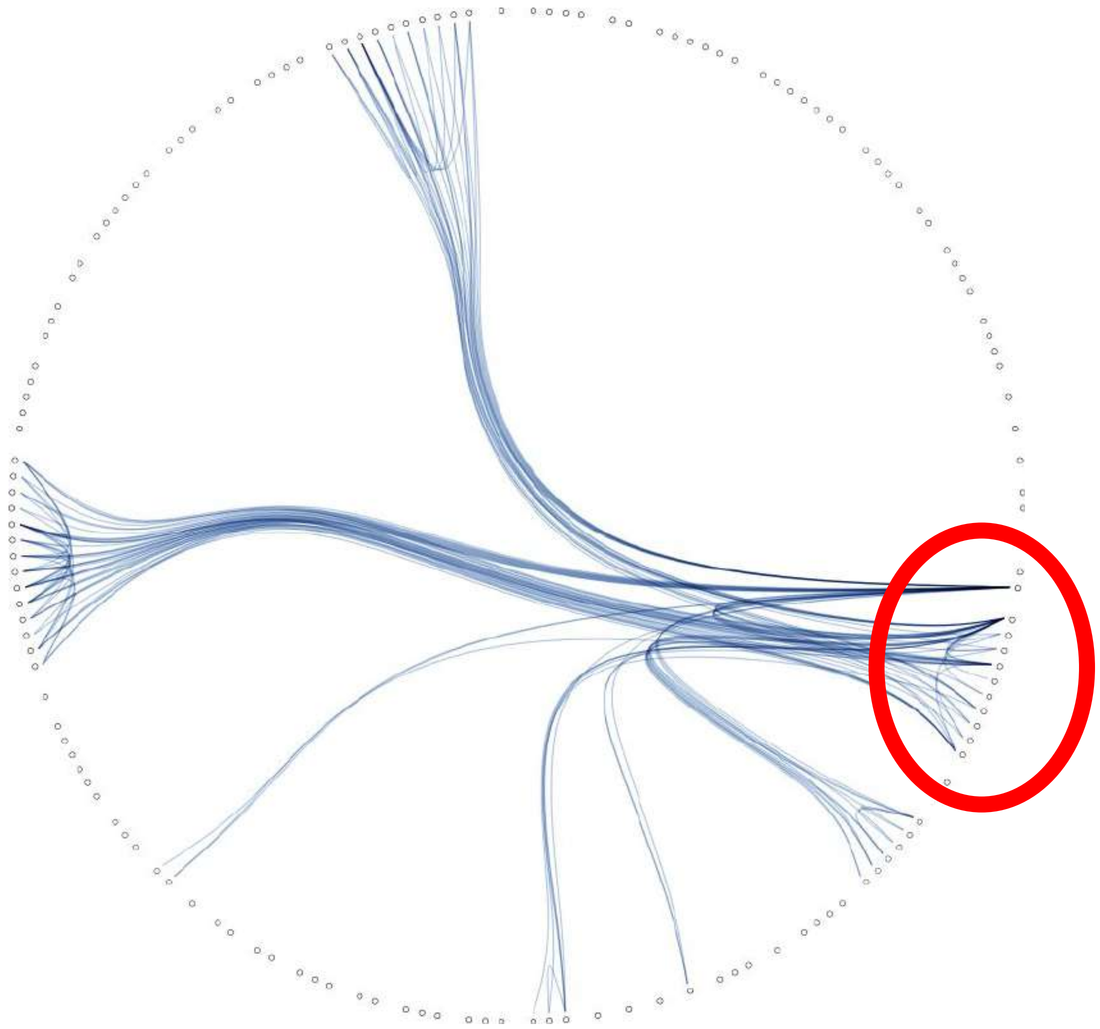
```elixir
defmodule Loyalty.Customers.Model.Customer do
 defstruct [:id, :name, :tier]

 alias Loyalty.Customers.Model.Tier

 @type id :: String.t()
 @opaque t :: %__MODULE__{
          id: id | nil,
          name: String.t(),
          tier: Tier.t()
         }
```

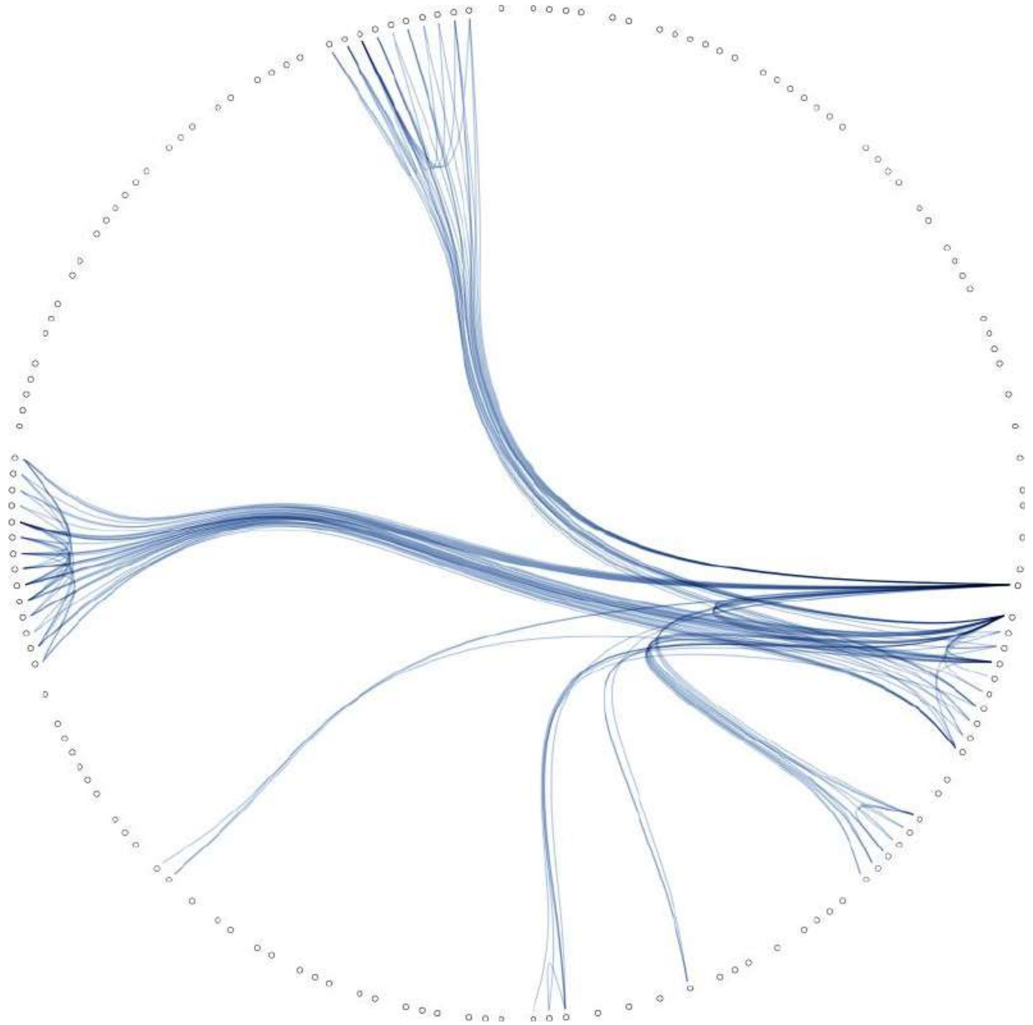
```elixir
@spec new(map()) :: {:ok, t()} | {:error, any()}
def new(params) do
   case cast(params) do
     {:ok, params} ->
       {:ok, struct(__MODULE__, params)}


     {:error, _} = error ->
       error
   end
 end
```

```elixir
@spec upgrade(t(), Tier.t()) :: {:ok, t()} | {:error, any()}
def upgrade(%__MODULE__{tier: current_tier} = customer, new_tier) do
   case Tier.compare(new_tier, current_tier) do
     :gt ->
       {:ok, %__MODULE__{customer | tier: new_tier}}


     _ ->
       {:error, "can't upgrade to lower tier"}
   end
 end
```

```elixir
@spec to_map(t()) :: map()
def to_map(%__MODULE__{} = customer) do
   customer
   |> Map.from_struct()
   |> Map.update!(:tier, &Tier.to_atom/1)
 end
```

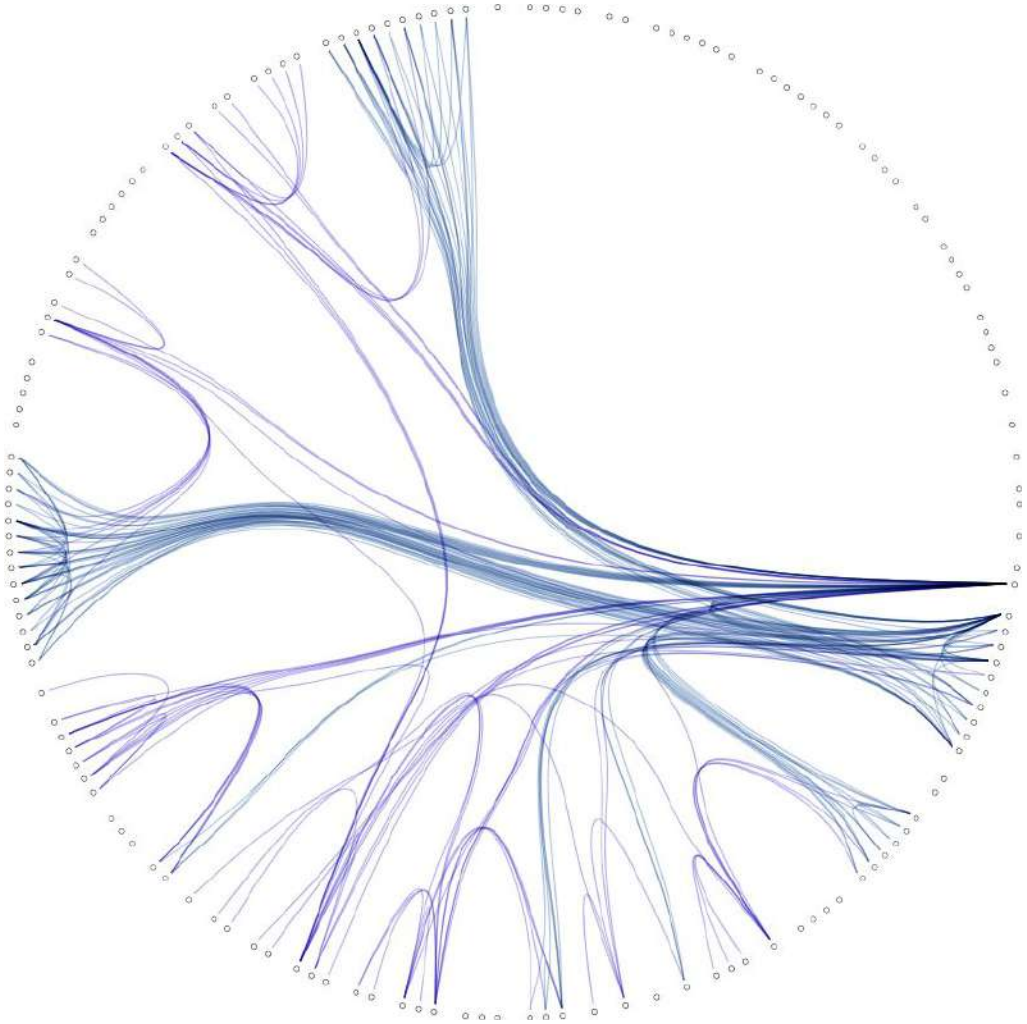


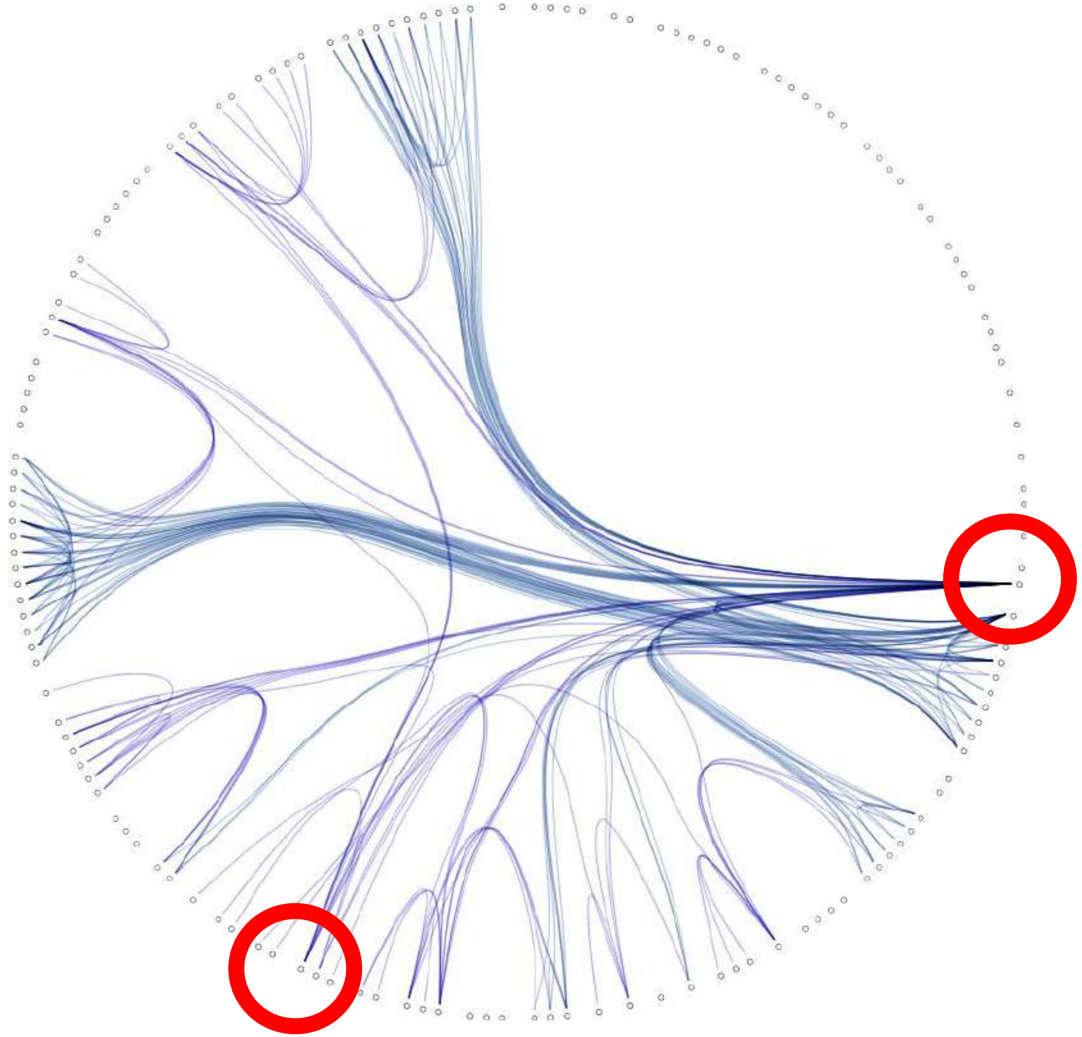


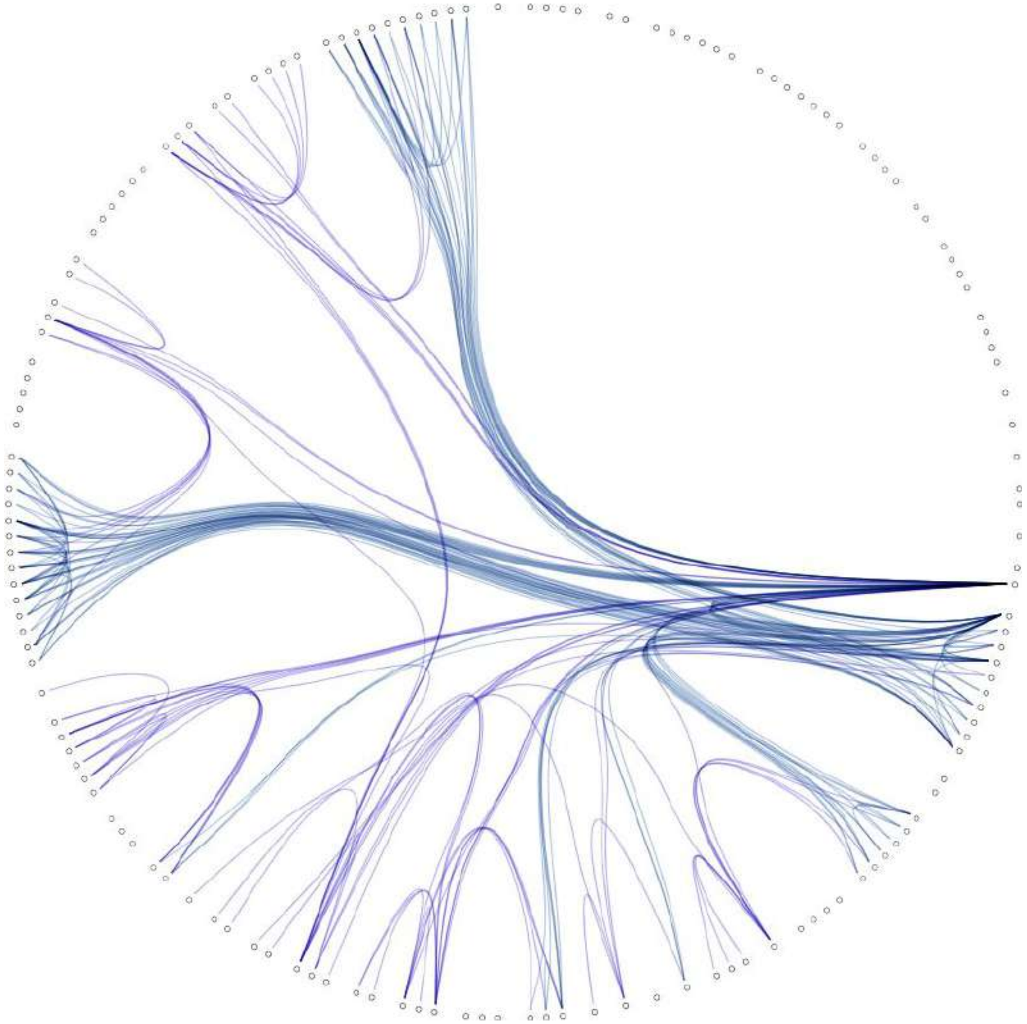


```elixir
defmodule Loyalty.Customers.IO.Customers do
 alias Loyalty.Customers.IO.Schema
 alias Loyalty.Customers.Model


 @spec get(Model.Customer.id()) :: {:ok, Model.Customer.t()}
                                | {:error, any()}
 def get(id) do
   # Repo logic lives here
     ...
 end
```

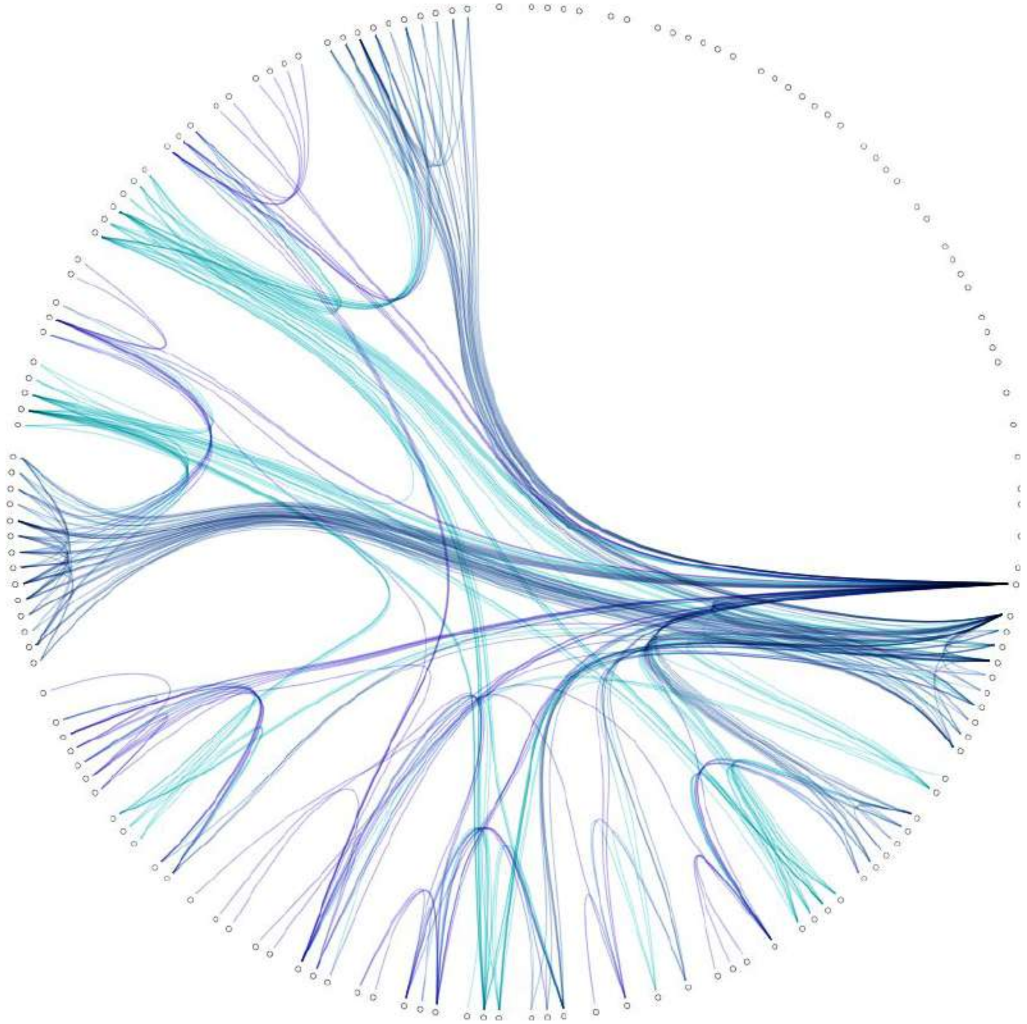
```elixir
@spec update(Model.Customer.t(), Model.Customer.t()) ::
         {:ok, Model.Customer.t()} | {:error, any()}

 def update(old_customer, new_customer) do
   # Repo logic lives here
     ...
 end
end
```

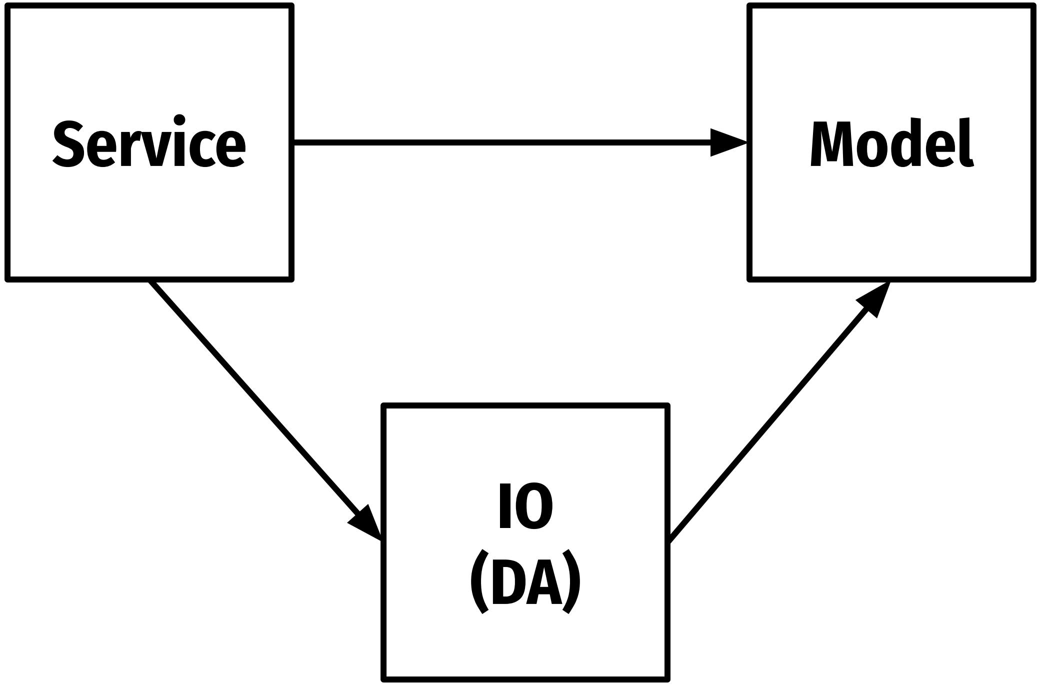


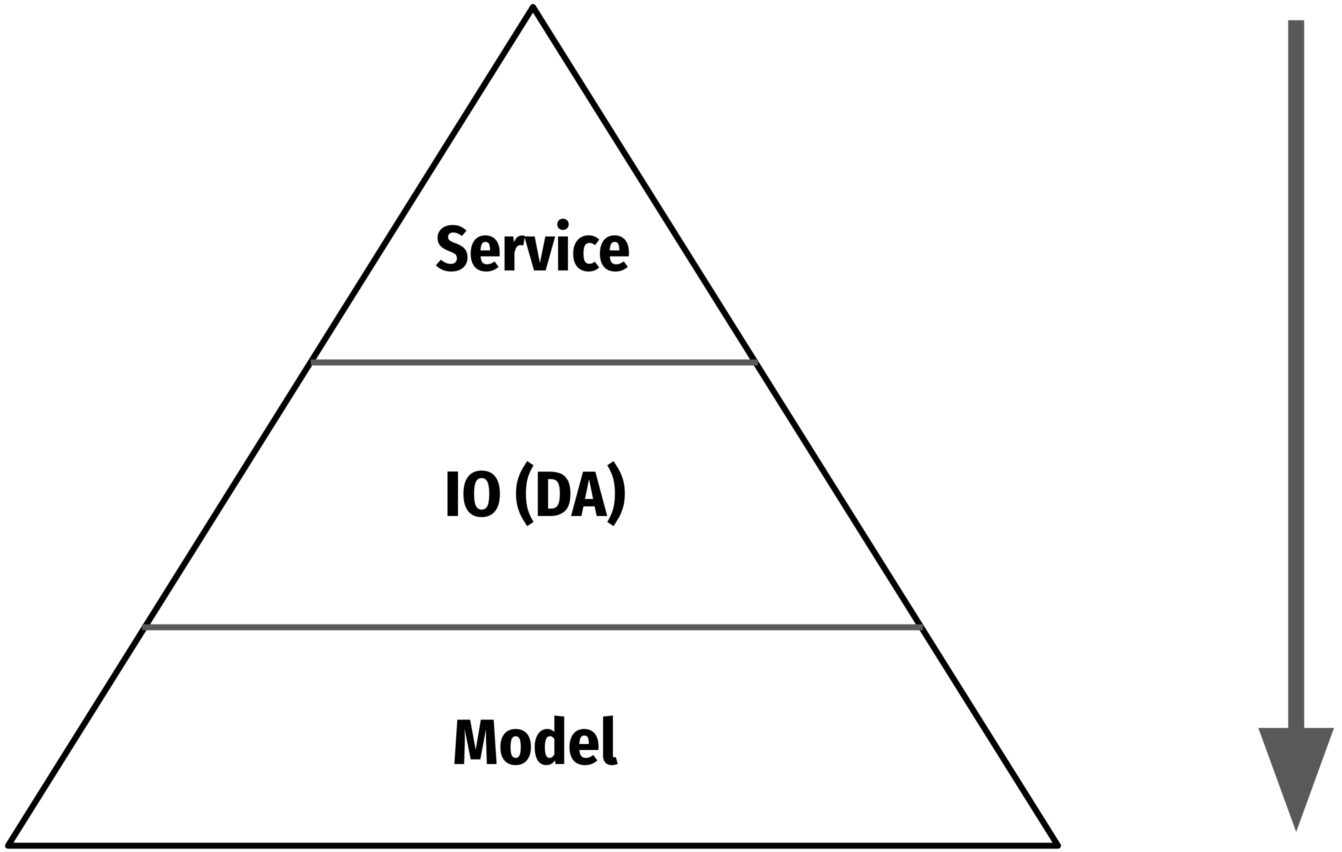


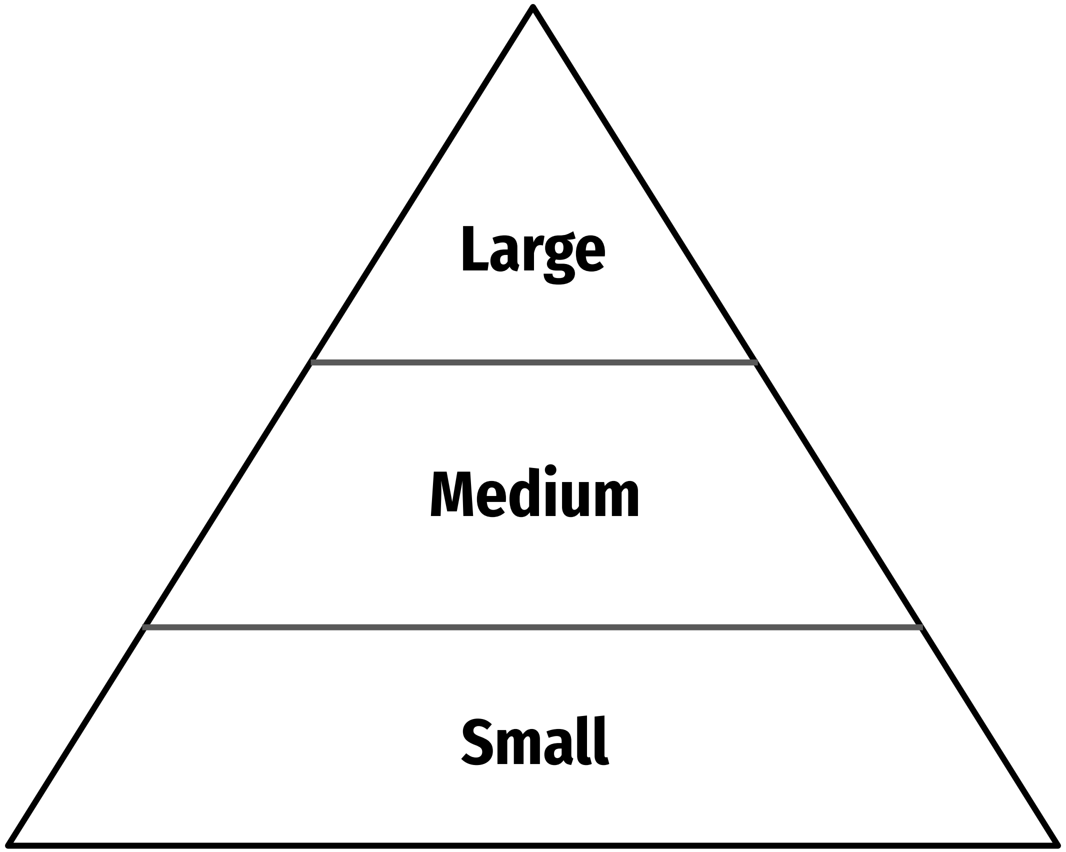


```elixir
defmodule Loyalty.Customers.Service.Customers do
 alias Loyalty.Customers.{IO, Model}


 @spec upgrade(String.t(), String.t()) :: {:ok, map()} | {:error, any()}
 def upgrade(customer_id, tier) do
   with {:ok, tier} <- Model.Tier.new(tier),
        {:ok, old_customer} <- IO.Customers.get(customer_id),
        {:ok, new_customer} <- Model.Customer.upgrade(new_customer, tier),
        {:ok, updated_customer} <- IO.Customers.update(old_customer,
new_customer) do
      {:ok, Model.Customer.to_map(updated_customer)}
   end
 end
end
```



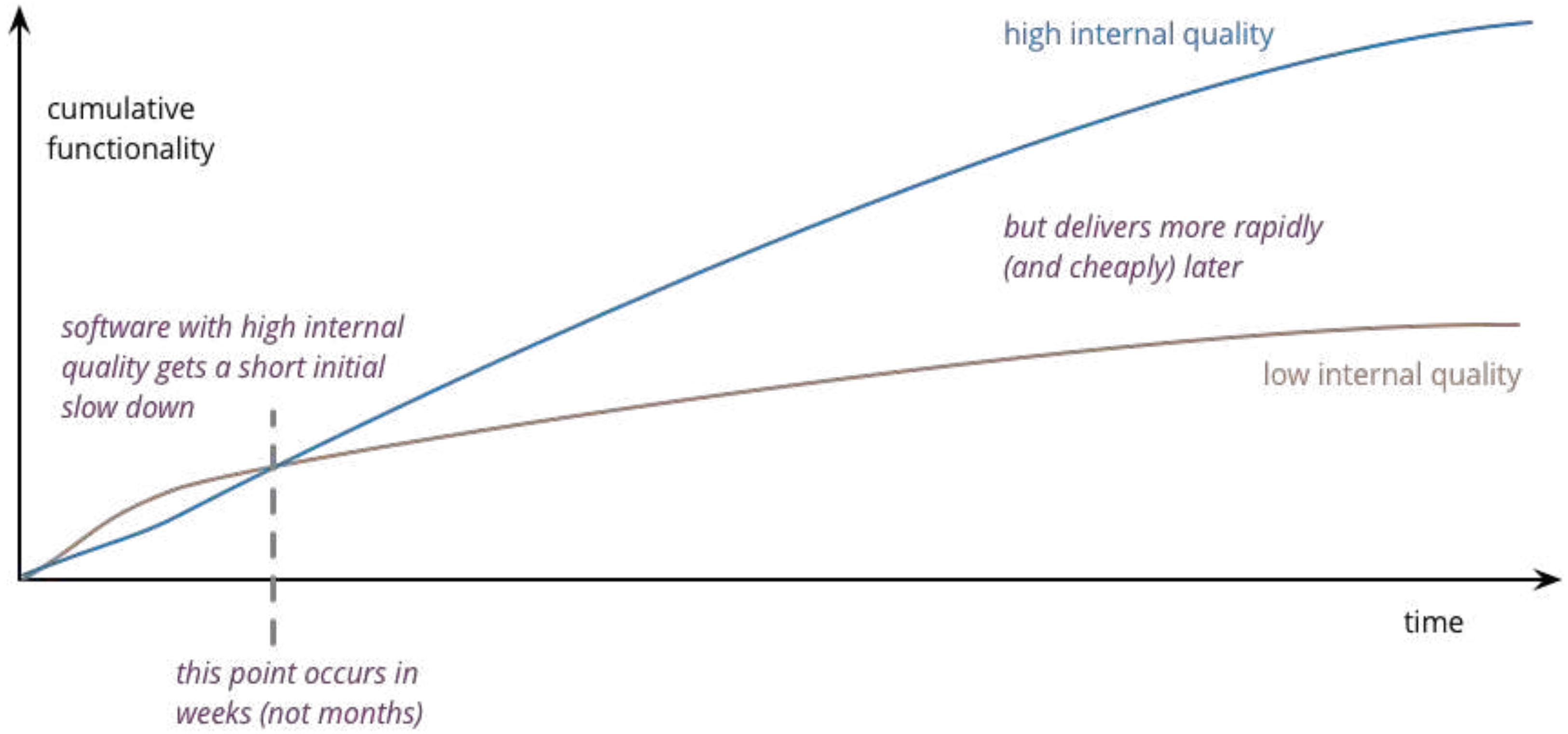

Service
Model
IO
(DA)


Service
IO (DA)
Model



https://testing.googleblog.com/2010/12/test-sizes.html

CONCERNS



https://martinfowler.com/articles/is-quality-worth-cost.html

WHAT DOES IT DO?

```
mix test test/large --trace
```

```
ls lib/*/*/service/*.ex
```

EXTERNAL DEPENDENCIES?

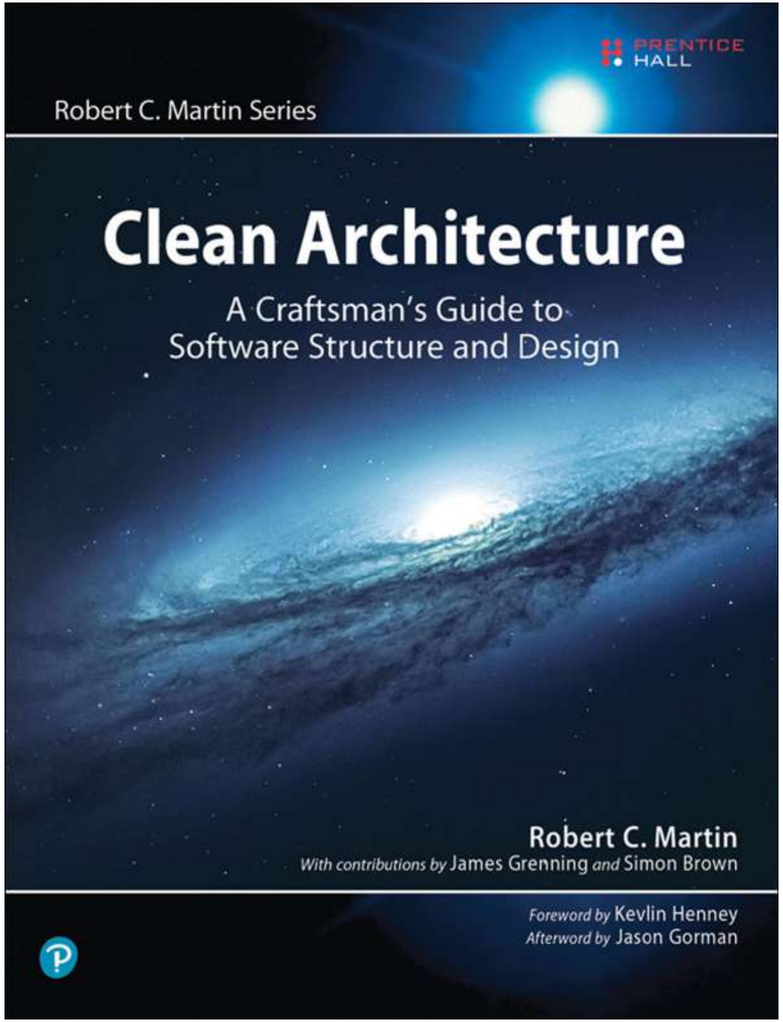
```
mix test test/medium --trace
```

```
ls lib/*/*/io/*.ex
```

DOMAIN MODEL?

```
mix test test/small --trace
```

```
ls lib/*/*/model/*.ex
```


PRENTICE HALL

Robert C. Martin Series

Clean Architecture

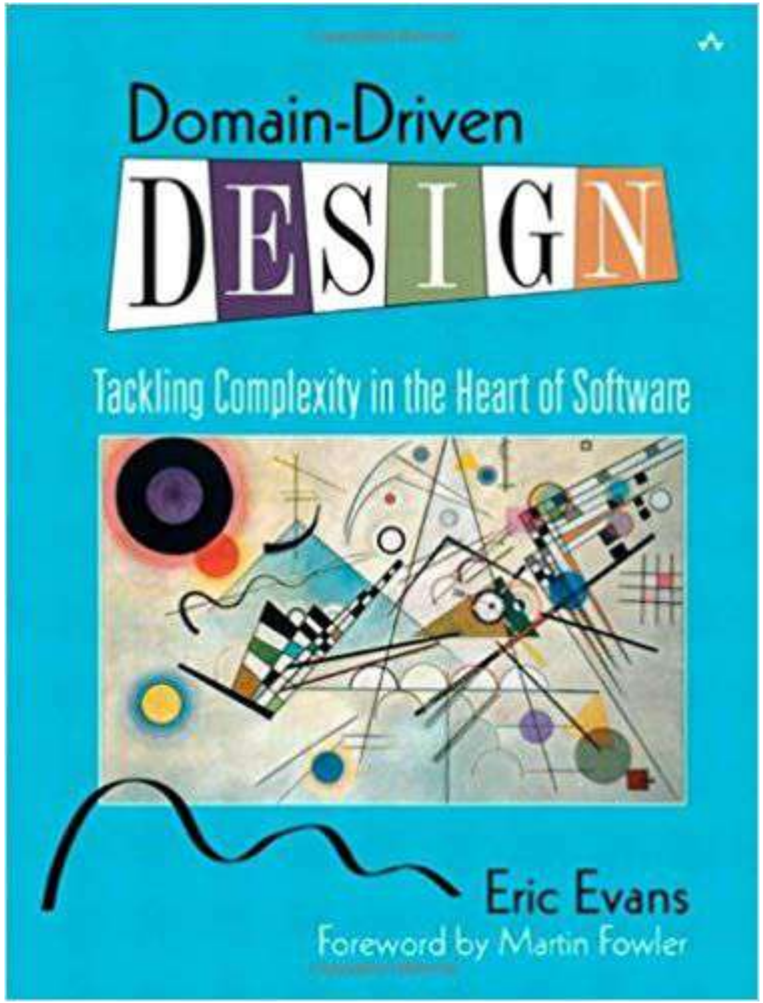
A Craftsman's Guide to Software Structure and Design

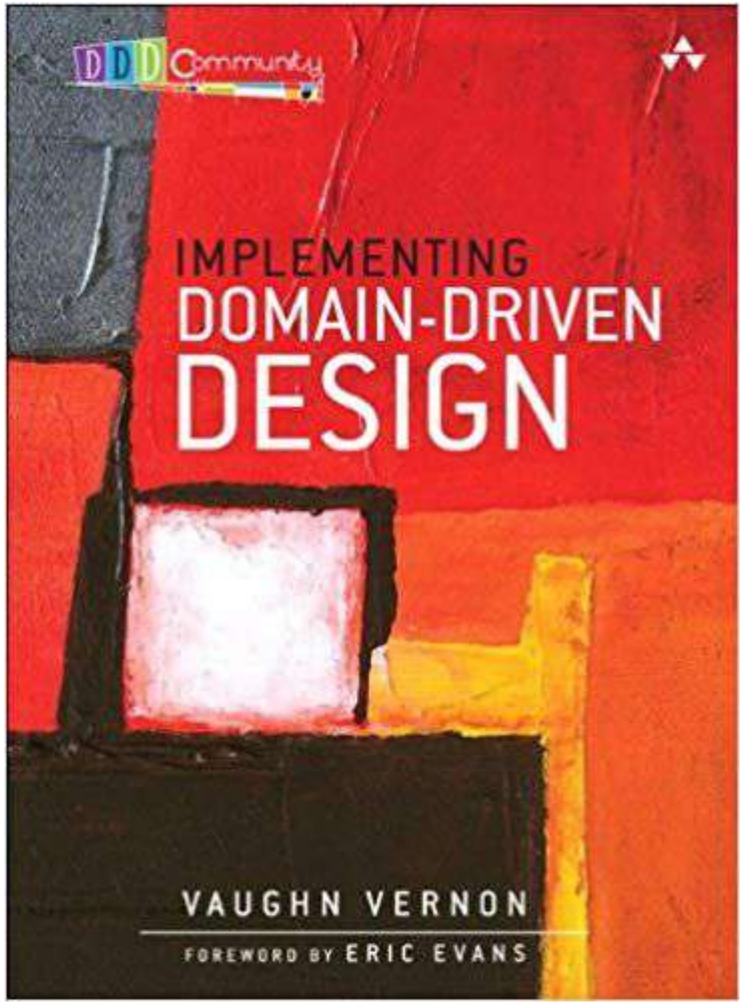
Robert C. Martin

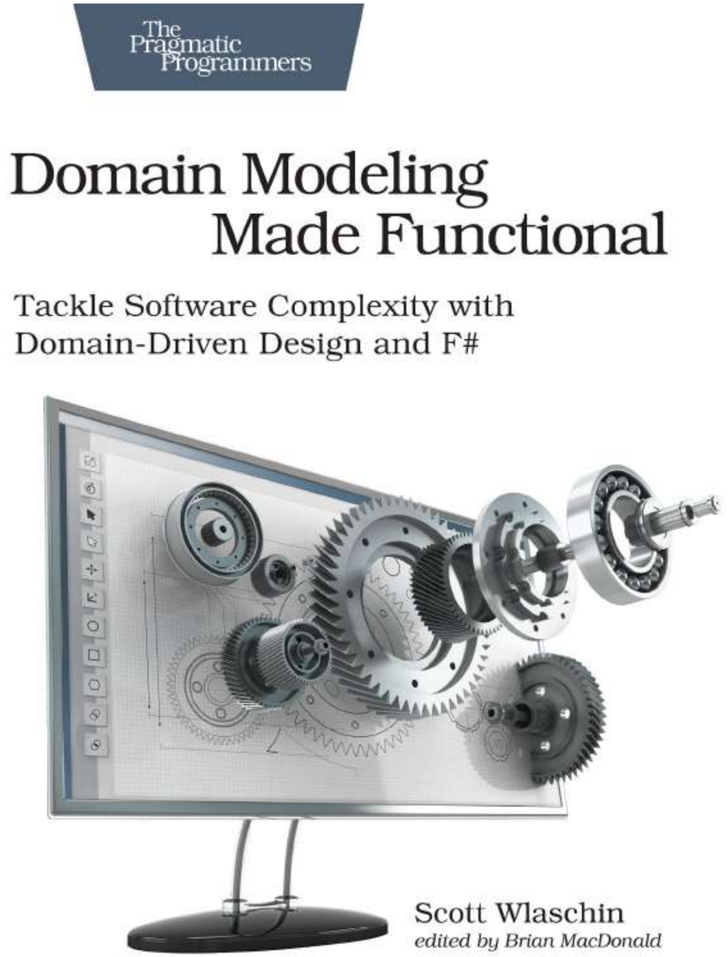
With contributions by James Grenning *and* Simon Brown

Foreword by Kevlin Henney

Afterword by Jason Gorman


Domain-Driven
DESIGN
Tackling Complexity in the Heart of Software
Eric Evans
Foreword by Martin Fowler


DDD Community
IMPLEMENTING
DOMAIN-DRIVEN
DESIGN
VAUGHN VERNON
FOREWORD BY ERIC EVANS



The Pragmatic Programmers

Domain Modeling Made Functional

Tackle Software Complexity with Domain-Driven Design and F#

Scott Wlaschin

edited by Brian MacDonald

https://fsharpforfunandprofit.com/

Mattheus van Hellemont - The Alchemist (17th century)

Louis Emile Adane - Apprentice. Man and boy making shoes (1914)

The Snow Shoe Tramp by Torchlight - on the mountain, Canadian Illustrated News (1873)

Bartholomeus van Bassen, Esaias van de Velde - Interior of a Catholic Church (1626)

Zarco and Zito Students - Levantate! Arise! Mural

Esaias van de Velde - De buitenpartij (1615)

Esaias van de Velde - A Wooded River Landscape With Figures on a Path on a River Bank Beside a Village (1624)

Jan Steen - Argument over a Card Game (17th century)