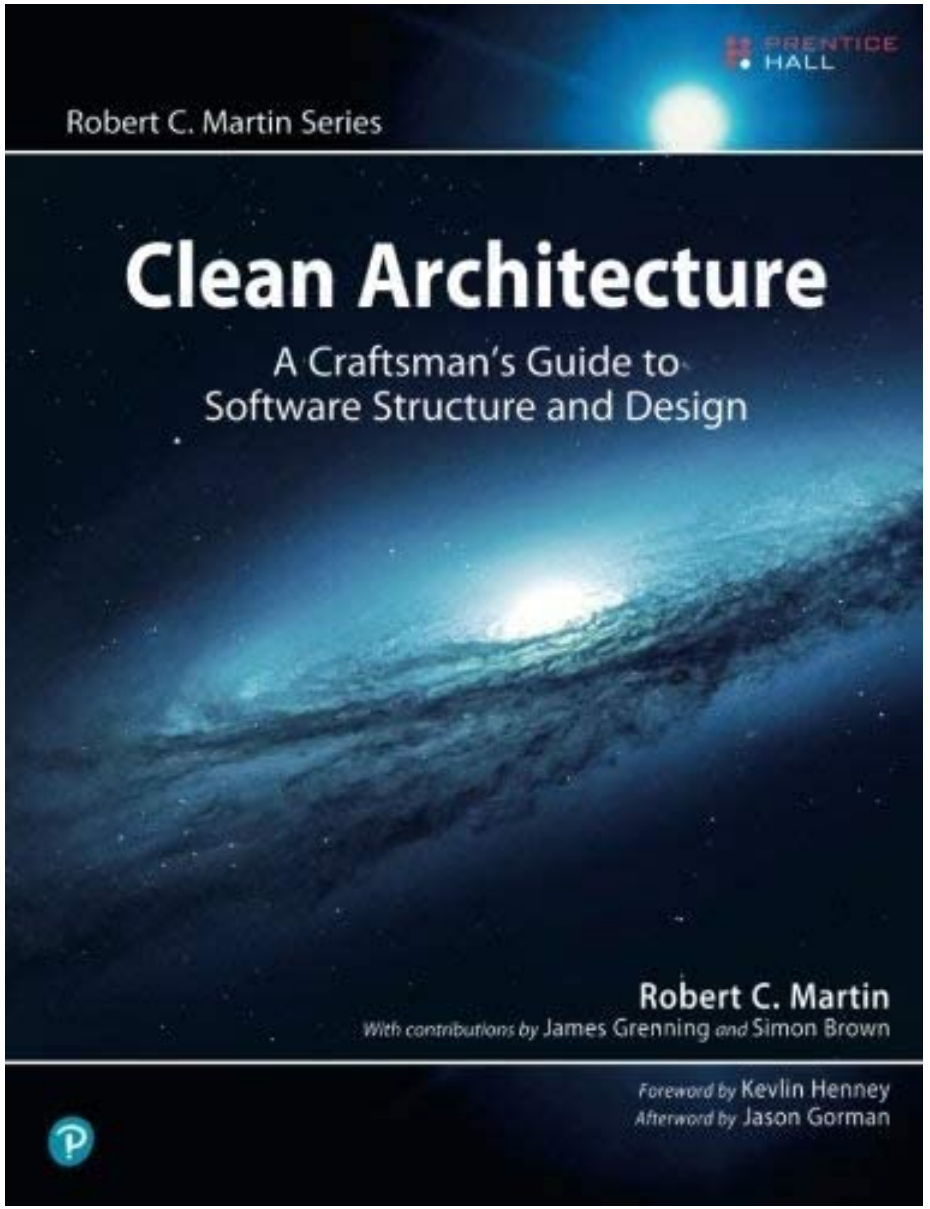
Mixing Clean Architecture

Мирослав Малкин
Руководитель направления экспериментальных разработок FunBox

О чем я расскажу

1. Clean Mixer github.com/funbox/clean_mixer.
2. Архитектура: компоненты, принципы, метрики.
3. Интерфейсы и Инверсия Зависимостей.
4. Clean (Hexagonal) Architecture.
5. Как за этим всем следить?

Robert Martin
Clean Architecture

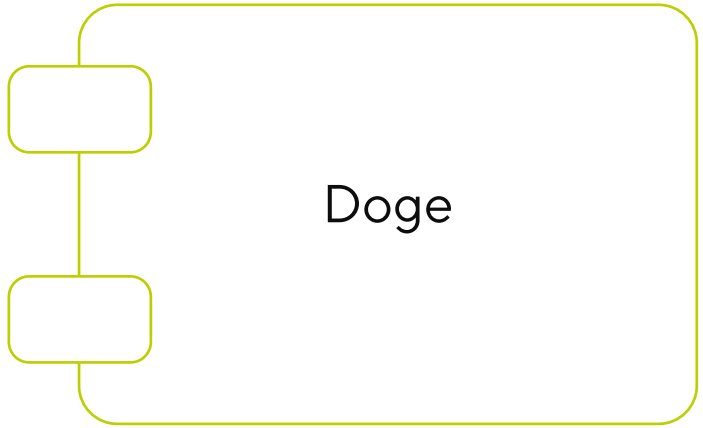


Особенно: часть IV
Component Principles

Компоненты и связи

Компонент

- Физически — любой набор исходных файлов:
  - Неймспейс в отдельной папке.
  - Umbrella app.
  - Hex пакет.
- Логически:
  - Бизнес:
    - Bounded Context (DDD).
  - Технически:
    - Слой.
    - Отдельный инфраструктурный элемент (utils).



Связь

- Упоминание модуля другого компонента в коде.

.clean_mixer.exs

```
[
  components: [
    {"chatbot_platform_api", "apps/chatbot_platform_api/lib"},
    ####
    {"app_server", "apps/app_server/lib"},
    ####
    {"im", "apps/im/lib"},
    {"im/business_chats", "apps/im/lib/im/business_chats"},
    {"im/app_services", "apps/im/lib/im/app_services"},
    ####
    {"rcs/app_services", "apps/rcs/lib/rcs/app_services"},
    {"rcs/messaging", "apps/rcs/lib/rcs/messaging"},
    {"rcs/cpim", "apps/rcs/lib/rcs/cpim"},
    {"rcs/session_management", "apps/rcs/lib/rcs/session_management"},
    {"rcs/ims_stack", "apps/rcs/lib/rcs/ims_stack"},
    ####
    {"rcs_chatbots/app_services", "apps/rcs_chatbots/lib/rcs_chatbots/app_services"},
    {"rcs_chatbots/messaging", "apps/rcs_chatbots/lib/rcs_chatbots/messaging"},
    ####
    {"sip", "apps/sip/lib"},
    ####
    {"msrp", "apps/msrp/lib"},
    ####
    {"network", "apps/network/lib"},
    ####
    {"data_io/interface", "apps/data_io/lib/data_io/interface"},
    {"data_io/app_services", "apps/data_io/lib/data_io/app_services"},
    ####
    {"utils", "apps/utils/lib"}
  ]
]
```

```
miros@miros-retina:~/pjs/rcs-messaging-server clean-mixer-demo * >mix clean_mixer.list
===> chatbot_platform_api -> im/app_services
 * apps/chatbot_platform_api/lib/controllers/messages_controller.ex
 |---> apps/im/lib/im/app_services/business_chat_manager.ex (runtime)
===> chatbot_platform_api -> im/business_chats
 * apps/chatbot_platform_api/lib/controllers/messages_controller.ex
 |---> apps/im/lib/im/business_chats/business_chat.ex (runtime,struct)
 * apps/chatbot_platform_api/lib/views/messages_view.ex
 |---> apps/im/lib/im/business_chats/business_chat.ex (runtime,struct)
 * apps/chatbot_platform_api/lib/webhook.ex
 |---> apps/im/lib/im/business_chats/chatbot_event_handler.ex (compile)
===> chatbot_platform_api -> utils
 * apps/chatbot_platform_api/lib/chatbot_platform_api.ex
 |---> apps/utils/lib/config_fetch.ex (compile)
 * apps/chatbot_platform_api/lib/webhook.ex
 |---> apps/utils/lib/nonstrict_logger.ex (compile)
===> chatbot_platform_api -> im
 * apps/chatbot_platform_api/lib/manager/use_cases/send_message.ex
 |---> apps/im/lib/im/event_listener.ex (runtime)
 |---> apps/im/lib/im/message.ex (runtime)
 |---> apps/im/lib/im/message_status.ex (runtime)
 * apps/chatbot_platform_api/lib/views/messages_view.ex
 |---> apps/im/lib/im/message_status.ex (runtime,struct)
 * apps/chatbot_platform_api/lib/webhook.ex
 |---> apps/im/lib/im/message.ex (runtime,struct)
 |---> apps/im/lib/im/message_status.ex (runtime,struct)
```

```
miros@miros-retina:~/pjs/rcs-messaging-server clean-mixer-demo >mix clean_mixer.list_-s im

miros@miros-retina:~/pjs/rcs-messaging-server clean-mixer-demo >mix clean_mixer.list -t im
===> chatbot_platform_api -> im
 * apps/chatbot_platform_api/lib/manager/use_cases/send_message.ex
 |---> apps/im/lib/im/event_listener.ex (runtime)
 |---> apps/im/lib/im/message.ex (runtime)
 |---> apps/im/lib/im/message_status.ex (runtime)
 * apps/chatbot_platform_api/lib/views/messages_view.ex
 |---> apps/im/lib/im/message_status.ex (runtime,struct)
 * apps/chatbot_platform_api/lib/webhook.ex
 |---> apps/im/lib/im/message.ex (runtime,struct)
 |---> apps/im/lib/im/message_status.ex (runtime,struct)

===> im/business_chats -> im
 * apps/im/lib/im/business_chats/business_chat.ex
 |---> apps/im/lib/im/abonent.ex (runtime)
 |---> apps/im/lib/im/chat.ex (runtime)
 |---> apps/im/lib/im/business_chats/chatbot.ex (runtime)
 |---> apps/im/lib/im/business_chats/chatbot_event_handler.ex (runtime)
 |---> apps/im/lib/im/event_listener.ex (runtime)
 |---> apps/im/lib/im/message.ex (runtime)
 |---> apps/im/lib/im/message_status.ex (runtime)
 |---> apps/im/lib/im/session.ex (runtime)
 * apps/im/lib/im/business_chats/chatbot_event_handler.ex
 |---> apps/im/lib/im/abonent.ex (runtime)
 |---> apps/im/lib/im/business_chats/chatbot.ex (runtime)
 |---> apps/im/lib/im/message.ex (runtime)
 |---> apps/im/lib/im/message_status.ex (runtime)
```

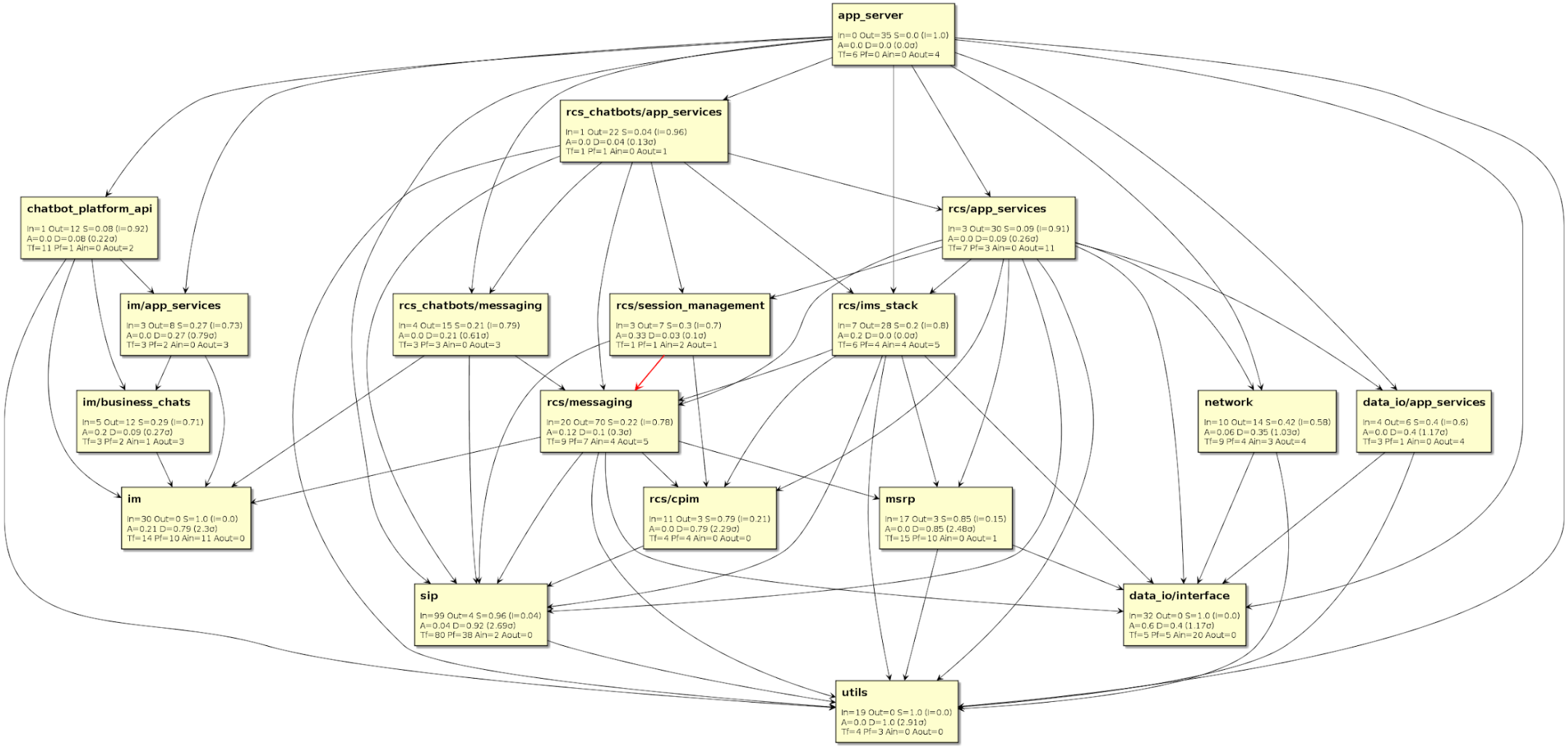
```
miros@miros-retina:~/pjs/rcs-messaging-server clean-mixer-demo >mix clean_mixer.list --file-sources "*/rcs_session.ex"
===> rcs/messaging -> data_io/interface
 * apps/rcs/lib/rcs/messaging/rcs_session.ex
 |---> apps/data_io/lib/data_io/interface/route.ex (runtime)
 |---> apps/data_io/lib/data_io/interface/endpoint.ex (runtime)
===> rcs/messaging -> rcs/cpim
 * apps/rcs/lib/rcs/messaging/rcs_session.ex
 |---> apps/rcs/lib/rcs/cpim/encoder.ex (runtime)
 |---> apps/rcs/lib/rcs/cpim/cpim_wrapper.ex (runtime)
===> rcs/messaging -> utils
 * apps/rcs/lib/rcs/messaging/rcs_session.ex
 |---> apps/utils/lib/nonstrict_logger.ex (compile)
===> rcs/messaging -> msrp
 * apps/rcs/lib/rcs/messaging/rcs_session.ex
 |---> apps/msrp/lib/msrp/sdp/decoder.ex (runtime)
===> rcs/messaging -> sip
 * apps/rcs/lib/rcs/messaging/rcs_session.ex
 |---> apps/sip/lib/sip/dialog/timer.ex (runtime)
 |---> apps/sip/lib/sip/uri.ex (runtime)
 |---> apps/sip/lib/sip/message/request.ex (runtime,struct)
 |---> apps/sip/lib/sip/message/header/content_type.ex (runtime,struct)
 |---> apps/sip/lib/sip/dialog/statuses.ex (runtime,struct)
 |---> apps/sip/lib/sip/transaction.ex (runtime)
 |---> apps/sip/lib/sip/message/request/method.ex (compile)
 |---> apps/sip/lib/sip/dialog.ex (compile)
 |---> apps/sip/lib/sip/message.ex (runtime)
 |---> apps/sip/lib/sip/message/response/status.ex (compile)
 |---> apps/sip/lib/sip_dialog_handler.ex (runtime)
 |---> apps/sip/lib/sip/message/response.ex (runtime,struct)
 |---> apps/sip/lib/sip/message/body/content.ex (runtime,struct)
 |---> apps/sip/lib/sip/transaction.ex (runtime,struct)
```

```
miros@miros-retina:~/pjs/rcs-messaging-server clean-mixer-demo >mix clean_mixer.list --file-sources "*/rcs_session.ex"
===> rcs/app_services -> rcs/messaging
 * apps/rcs/lib/rcs/app_services/rcs_session_service.ex
 |---> apps/rcs/lib/rcs/messaging/rcs_session.ex (runtime)
 * apps/rcs/lib/rcs/app_services/registry_session_manager.ex
 |---> apps/rcs/lib/rcs/messaging/rcs_session.ex (runtime)

===> rcs/session_management -> rcs/messaging
 * apps/rcs/lib/rcs/session_management/session_manager.ex
 |---> apps/rcs/lib/rcs/messaging/rcs_session.ex (runtime)
miros@miros-retina:~/pjs/rcs-messaging-server clean-mixer-demo >
```

```
miros@miros-retina:~/pjs/rcs-messaging-server clean-mixer-demo >mix clean_mixer.list_usages
===> chatbot_platform_api <- [app_server]
 * apps/chatbot_platform_api/lib/webhook.ex

===> im <- [chatbot_platform_api, im/business_chats, im/app_services, rcs/messaging, rcs_chatbots/messaging]
 * apps/im/lib/im/abonent.ex
 * apps/im/lib/im/app_services/business_chat_service.ex
 * apps/im/lib/im/chat.ex
 * apps/im/lib/im/business_chats/chatbot.ex
 * apps/im/lib/im/business_chats/chatbot_event_handler.ex
 * apps/im/lib/im/command_handler.ex
 * apps/im/lib/im/event_listener.ex
 * apps/im/lib/im/message.ex
 * apps/im/lib/im/message_status.ex
 * apps/im/lib/im/session.ex

===> im/business_chats <- [chatbot_platform_api, im/app_services]
 * apps/im/lib/im/business_chats/business_chat.ex
 * apps/im/lib/im/business_chats/chatbot_event_handler.ex

===> im/app_services <- [chatbot_platform_api, app_server]
 * apps/im/lib/im/app_services/business_chat_manager.ex
 * apps/im/lib/im/app_services/business_chat_supervisor.ex

===> rcs/app_services <- [app_server, rcs_chatbots/app_services]
 * apps/rcs/lib/rcs/app_services/msrp_server/supervisor.ex
 * apps/rcs/lib/rcs/app_services/registry_session_manager.ex
 * apps/rcs/lib/rcs/app_services/session_management_supervisor.ex
```

```
miros@miros-retina:~/pjs/rcs-messaging-server clean-mixer-demo >mix clean_mixer.list_usages -v
===> chatbot_platform_api <- [app_server]
 * apps/chatbot_platform_api/lib/webhook.ex
   | <--- apps/app_server/lib/app_server/endpoints_supervisor.ex

===> im <- [chatbot_platform_api, im/business_chats, im/app_services, rcs/messaging, rcs_chatbots/messaging]
 * apps/im/lib/im/abonent.ex
   | <--- apps/im/lib/im/business_chats/business_chat.ex
   | <--- apps/im/lib/im/business_chats/chatbot_event_handler.ex
 * apps/im/lib/im/app_services/business_chat_service.ex
   | <--- apps/im/lib/im/app_services/business_chat_manager.ex
 * apps/im/lib/im/chat.ex
   | <--- apps/im/lib/im/business_chats/business_chat.ex
   | <--- apps/rcs_chatbots/lib/rcs_chatbots/messaging/chat_id.ex
 * apps/im/lib/im/business_chats/chatbot.ex
   | <--- apps/im/lib/im/business_chats/business_chat.ex
   | <--- apps/im/lib/im/business_chats/chatbot_event_handler.ex
 * apps/im/lib/im/business_chats/chatbot_event_handler.ex
   | <--- apps/im/lib/im/business_chats/business_chat.ex
 * apps/im/lib/im/command_handler.ex
   | <--- apps/im/lib/im/app_services/business_chat_manager.ex
   | <--- apps/rcs_chatbots/lib/rcs_chatbots/messaging/rcs_chatbot_session.ex
 * apps/im/lib/im/event_listener.ex
   | <--- apps/chatbot_platform_api/lib/manager/use_cases/send_message.ex
   | <--- apps/im/lib/im/business_chats/business_chat.ex
   | <--- apps/im/lib/im/app_services/business_chat_manager.ex
   | <--- apps/rcs/lib/rcs/messaging/rcs_session.ex
   | <--- apps/rcs_chatbots/lib/rcs_chatbots/messaging/rcs_chatbot_session.ex
 * apps/im/lib/im/message.ex
   | <--- apps/chatbot_platform_api/lib/webhook.ex
   | <--- apps/chatbot_platform_api/lib/manager/use_cases/send_message.ex
   | <--- apps/im/lib/im/business_chats/business_chat.ex
   | <--- apps/im/lib/im/business_chats/chatbot_event_handler.ex
```

```
miros@miros-retina:~/pjs/rcs-messaging-server clean-mixer-demo >mix clean_mixer.plantuml
image file created at clean_mixer.png
miros@miros-retina:~/pjs/rcs-messaging-server clean-mixer-demo >open clean_mixer.png
miros@miros-retina:~/pjs/rcs-messaging-server clean-mixer-demo >
```



I = Instability = out / (in + out)
S = Stability = 1 - I
A = Abstractness = behaviours / total_modules
D = Distance = |A+I-1|
Tf = Total files
Pf = Public files
Ain = Abstract in
Aout = Abstract out

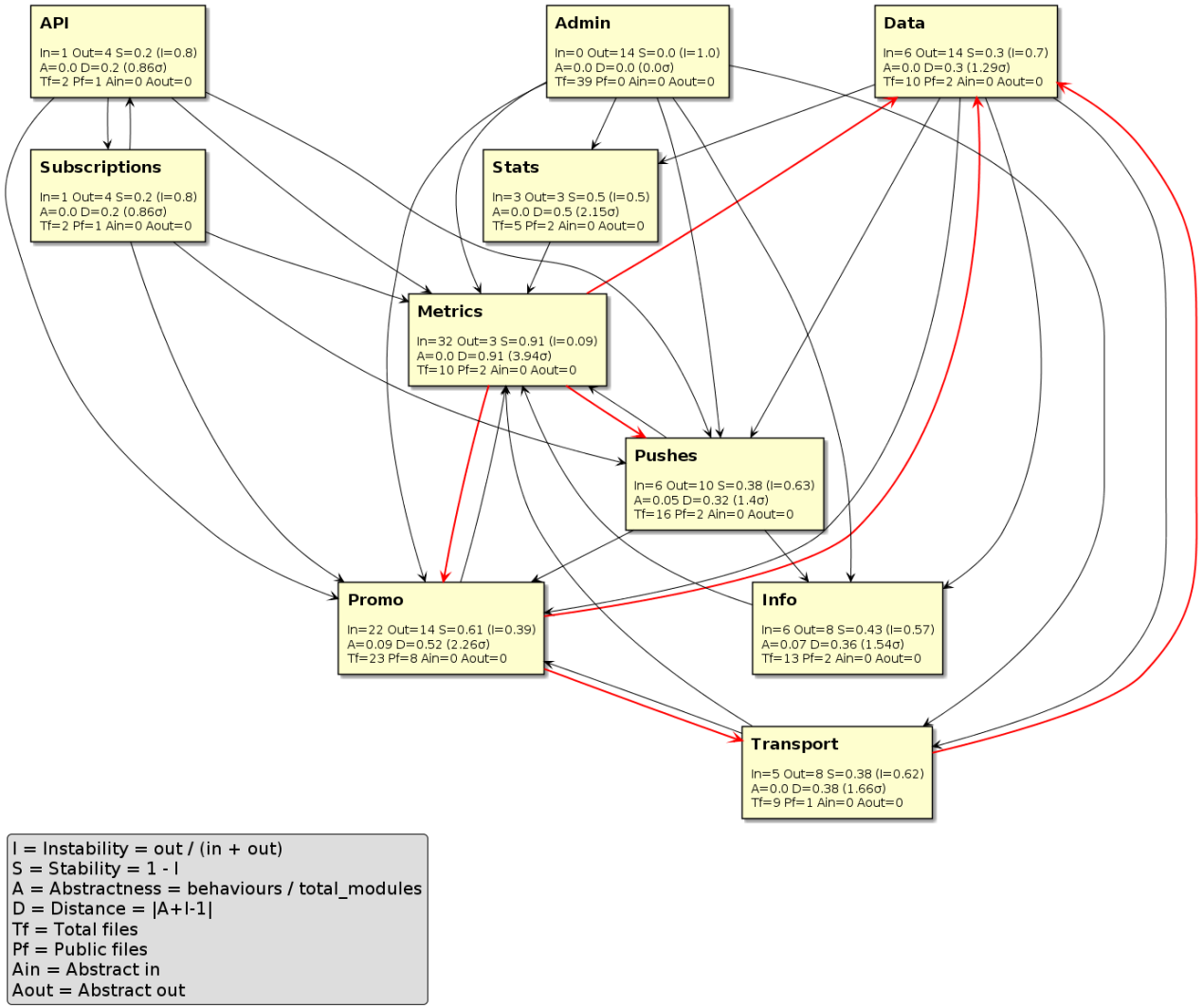
Когда принципы нарушаются

Проблемы:

- Не очевидно, где домен-ядро.
- Слои плохо просматриваются.
- Визуально выглядит «клубком».

Причины:

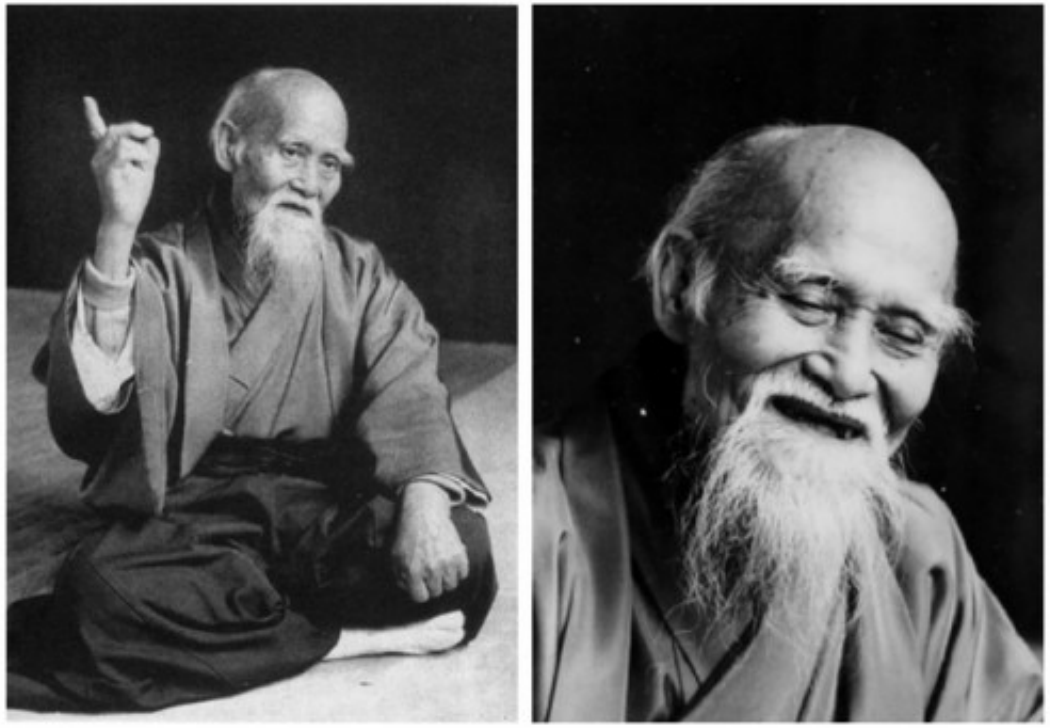
- Циклические зависимости.
- Нарушение Stable Dependency Principle.
- Все компоненты конкретные.



Cohesion

Как формируются компоненты?

1. Common Closure Principle.
2. Common Reuse Principle.



Common Closure Principle

1. Файлы, которые меняется по одной и той же причине и одновременно должны быть в одном компоненте.
2. Файлы, которые меняются по разным причинам и в разное время должны быть в разных компонентах.
3. Maintainability > Reusability.

Common Reuse Principle

1. Файлы, которые переиспользуются вместе, должны быть в одном компоненте.
2. Имеют много зависимостей между собой.
3. Сложно зависеть только от одного файла из такого компонента и не зависеть от остальных — файлы неразделимы.
4. Файлы, которые слабо связаны с остальными не должны быть в компоненте.
5. Не зависьте от того, что вам не нужно.

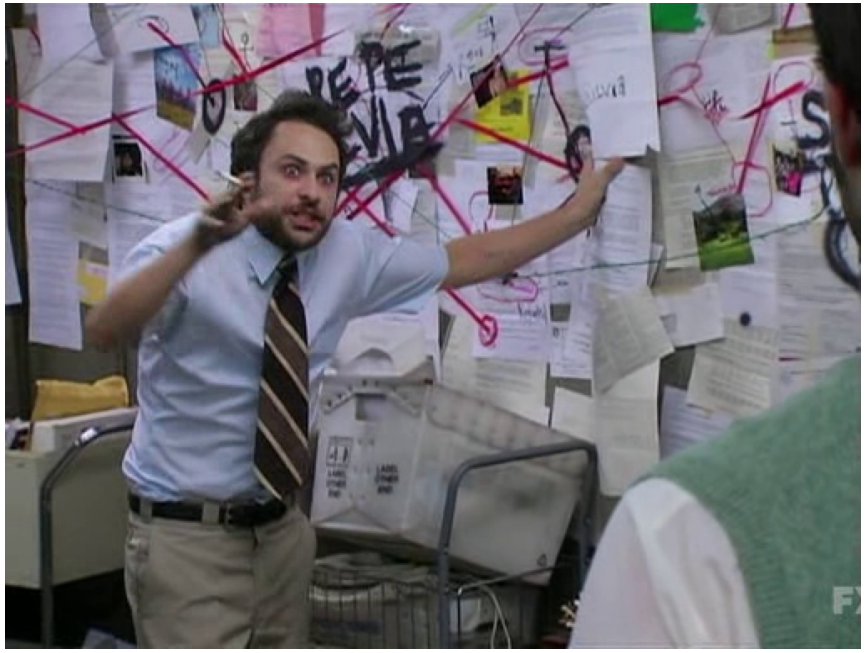
Coupling

Как компоненты связаны между собой?

- Acyclic dependencies principle.
- Stable dependency principle.
- Stable abstraction principle.

Связь:

- Зависимость на уровне исходных файлов (compile time dependency).

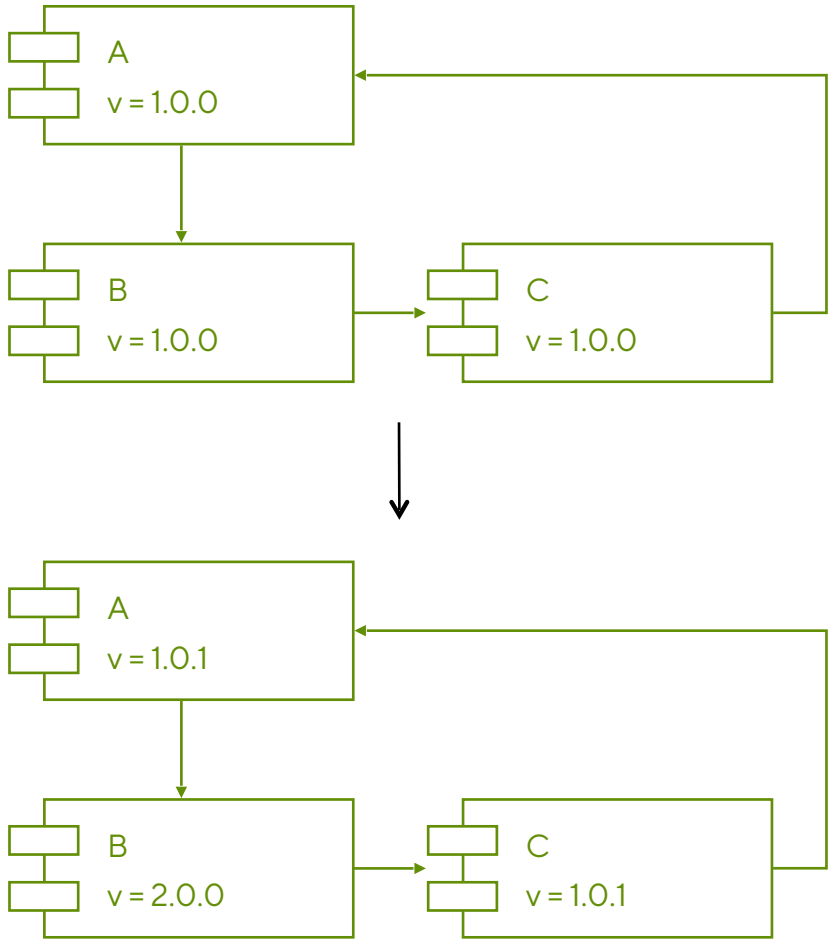


Acyclic Dependencies Principle

В графе зависимостей компонент
не должно быть циклов.

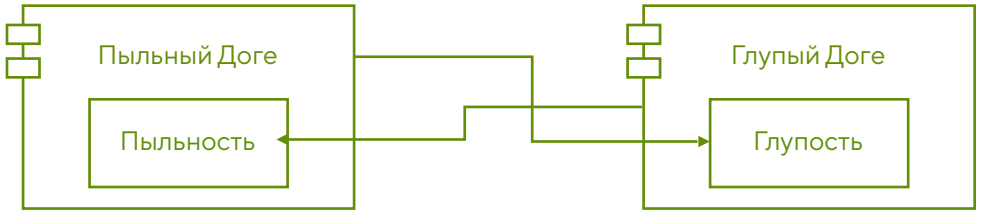
Причины

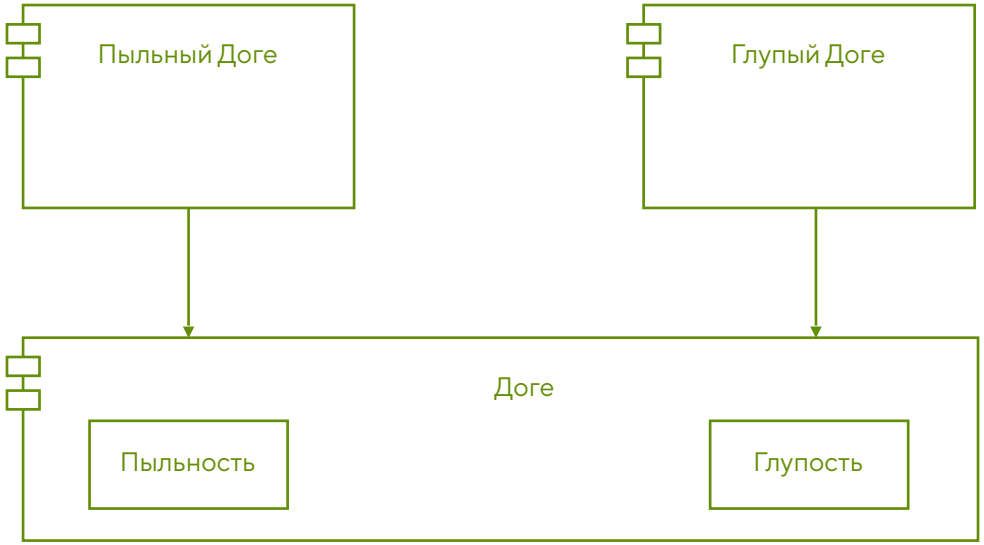
- При циклах в зависимостях нет независимых компонент.
- Независимая работа над разными частями системы затрудняется.
- Пример:
  - Компонент — hex пакет с версией.
  - Поднимаем версию одного — релизим все.
- В мире микросервисов — распределённый монолит.
- Весь цикл — один клубок — один большой компонент.



Как разбить цикл

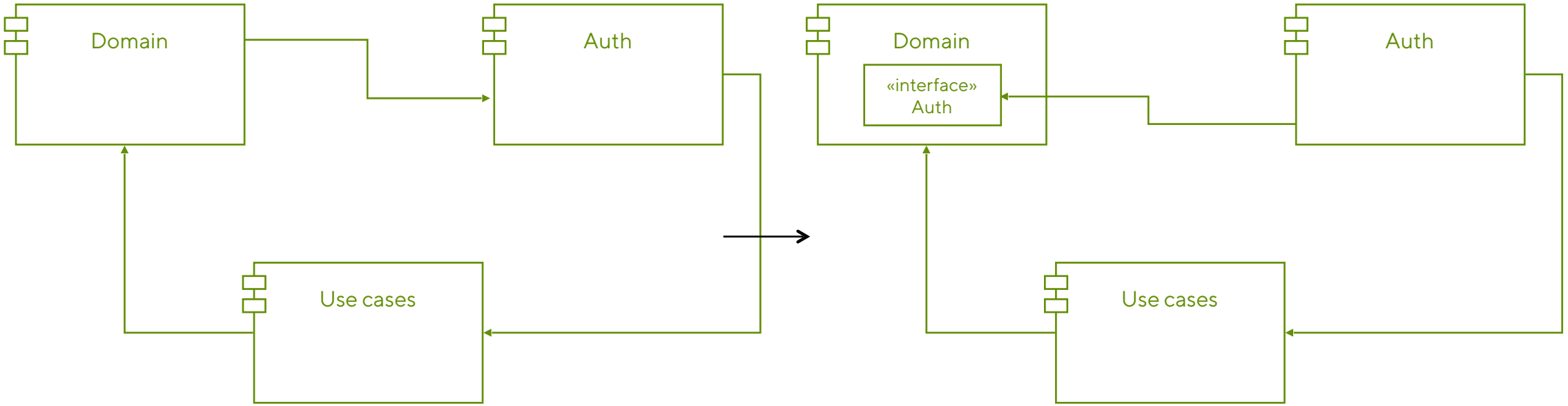
Вынести общий код в новый компонент





Как разбить цикл

Поменять направление зависимости —
инверсия зависимостей



Инверсия зависимостей

```elixir
defmodule Domain.UseCases.CreateUser do
  alias Domain.User

  @spec create_user(String.t, UserRegistry.t()) :: {:ok, User.t()} | {:error, term}
  def create_user(username, user_registry) do
    case user_registry.exists?(username) do
      {:ok, true} ->
        # ...
      other ->
        # ...
    end
  end

end

defmodule Domain.UserRegistry do
  @type t :: module
  @callback exists?(username) :: {:ok, boolean} | {:error, term}
end
```

```elixir
defmodule Auth.LDAP do
  alias Domain.UserRegistry
  @behaviour UserRegistry

  @impl UserRegistry
  def exists?(username) do
    # ...
    {:ok, true}
  end
end
```

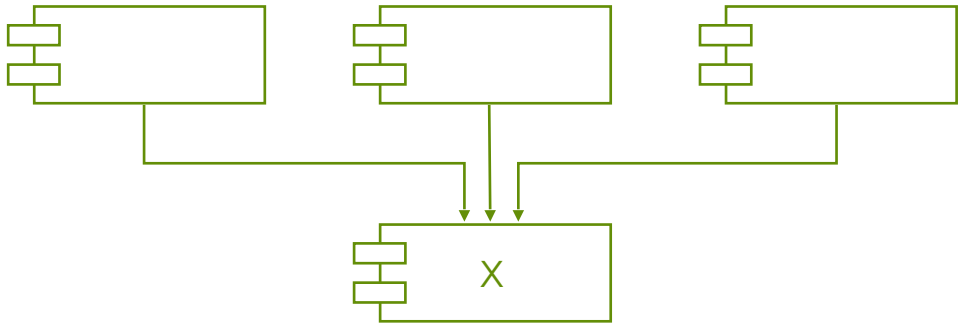
Интерфейсы

```
miros@miros-carbon:~/pjs/rcs-messaging-server master >mix clean_mixer.behaviours -b "IM.*"
===> IM.BusinessChatCommandHandler
 |---> IM.AppServices.BusinessChatManager

===> IM.GroupChatCommandHandler
 |---> IM.AppServices.GroupChatManager

===> IM.Session
 |---> RCSMessagingServer.RCSGroupChats.Messaging.RCSGroupChatSession
 |---> RCSMessagingServer.RCSChatbots.Messaging.RcsChatbotSession

===> IM.SubscriptionSession
 |---> RCSMessagingServer.RCSGroupChats.Messaging.RCSSubscriptionSession


miros@miros-carbon:~/pjs/rcs-messaging-server master >mix clean_mixer.behaviours -c "app_server"
===> Application
 |---> RCSMessagingServer.AppServer.Application

===> RCSMessagingServer.RCS.ImsStack.TransportRouting.MessageHandler
 |---> RCSMessagingServer.AppServer.RoutingHandlers.PingHandler
 |---> RCSMessagingServer.AppServer.RoutingHandlers.RegisterHandler
 |---> RCSMessagingServer.AppServer.RoutingHandlers.ProxyMessageHandler

===> Supervisor
 |---> RCSMessagingServer.AppServer.EndpointsSupervisor
```

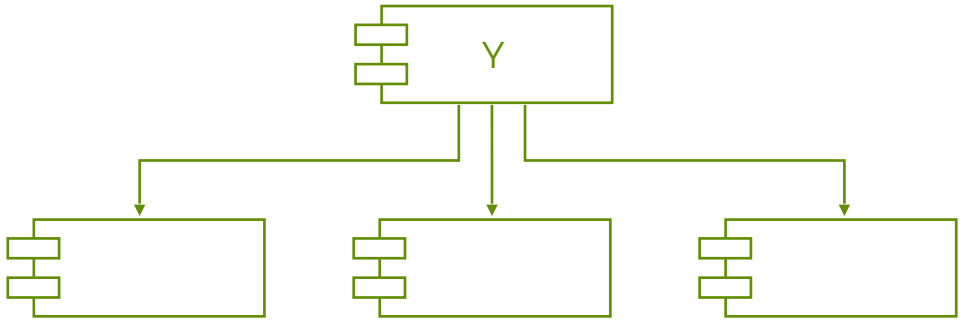
Stable dependency principle

Зависимости должны вести в направлении стабильности.

- Стабильность — оценивает сколько нужно работы, чтобы изменить компонент.
- Стабильность != изменчивость; стабильность — количество зависимостей.



Стабильный компонент — несет ответственность за компоненты, зависящие от него.



Нестабильный компонент — может меняться как хочет.

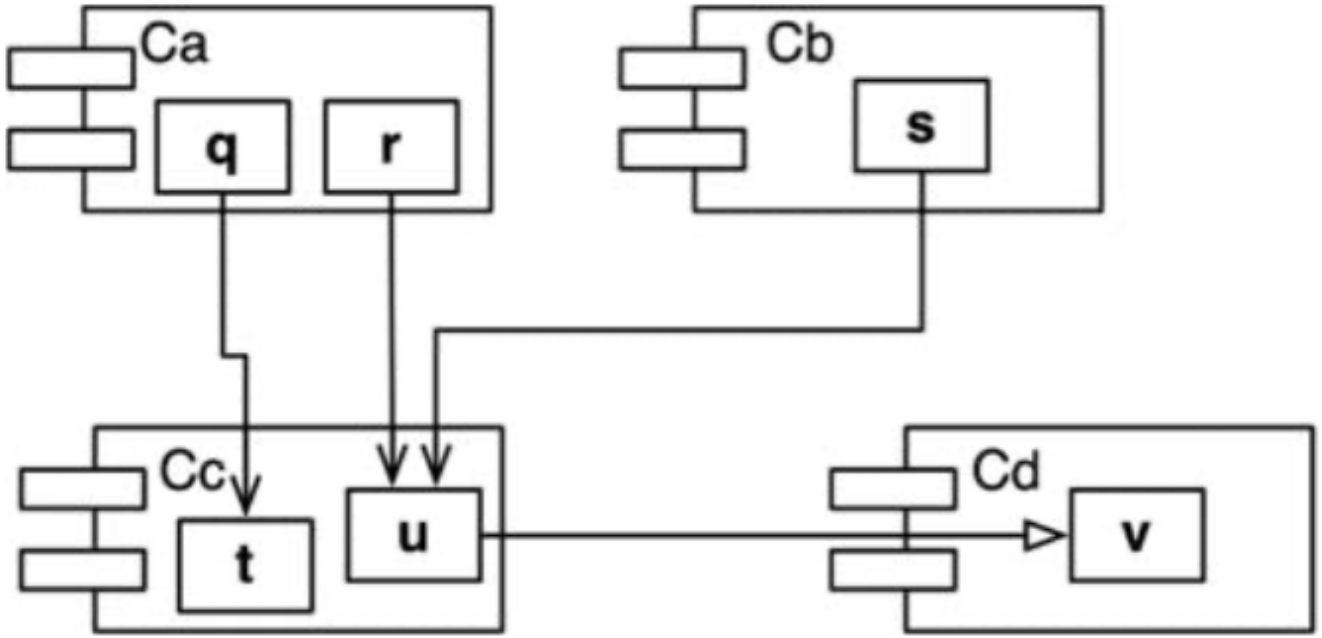
Stable dependency principle

1. Некоторые компоненты должны быть изменчивыми (как мы ожидаем).
2. Разделяйте, то что меняется часто и редко.
3. Компоненты часто меняются ⇒ Это должны быть нестабильные компоненты.
4. Компонент, который легко изменить (нестабильный) должен зависеть от компонентов, которые изменить сложнее, а не наоборот.
5. И наоборот: изменчивый компонент не должен быть зависимостью компонента, который сложно изменить.
6. ⇒ Зависимости должны вести в направлении стабильности.

Метрики IN и OUT

Fan-in = IN = количество входящих связей.

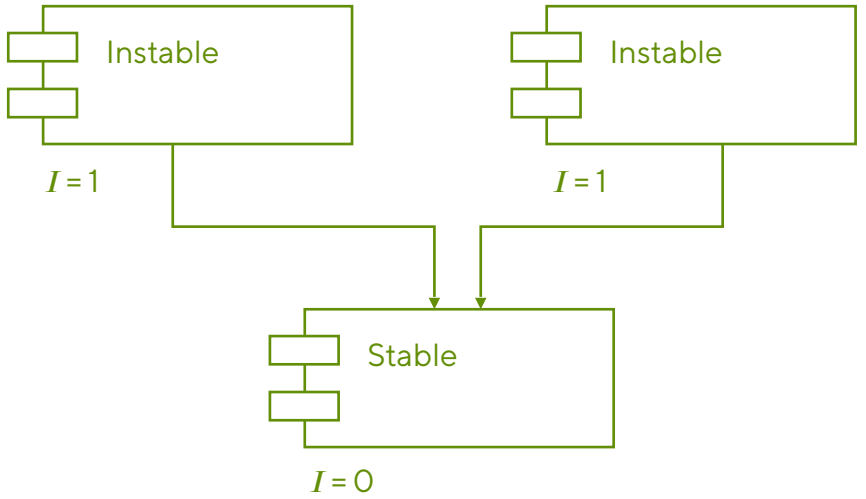
Fan-out = OUT = количество исходящих связей.



Метрика I (Instability)

I = OUT / (IN + OUT)

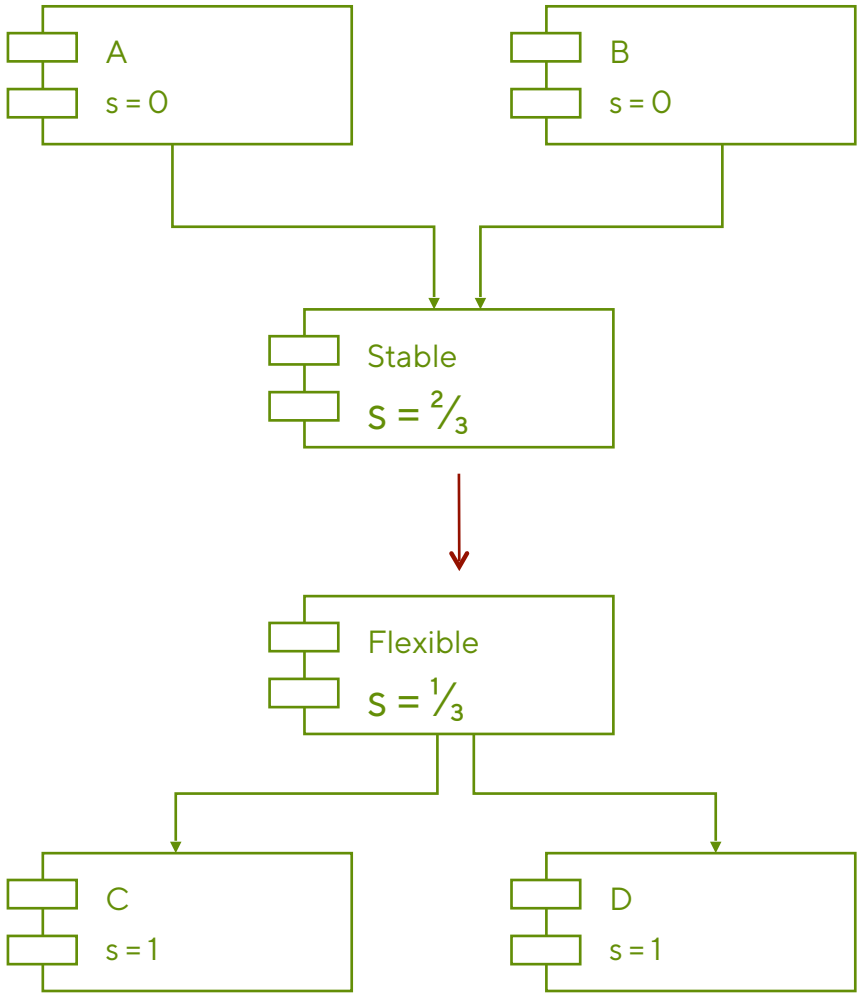
- Если I = 1, то от компонента никто не зависит, у него нет причин не меняться.
- Если I = 0, от компонента зависят другие, но сам он ни от кого не зависит. Ему сложно меняться, но с другой стороны его никто и не заставит измениться.



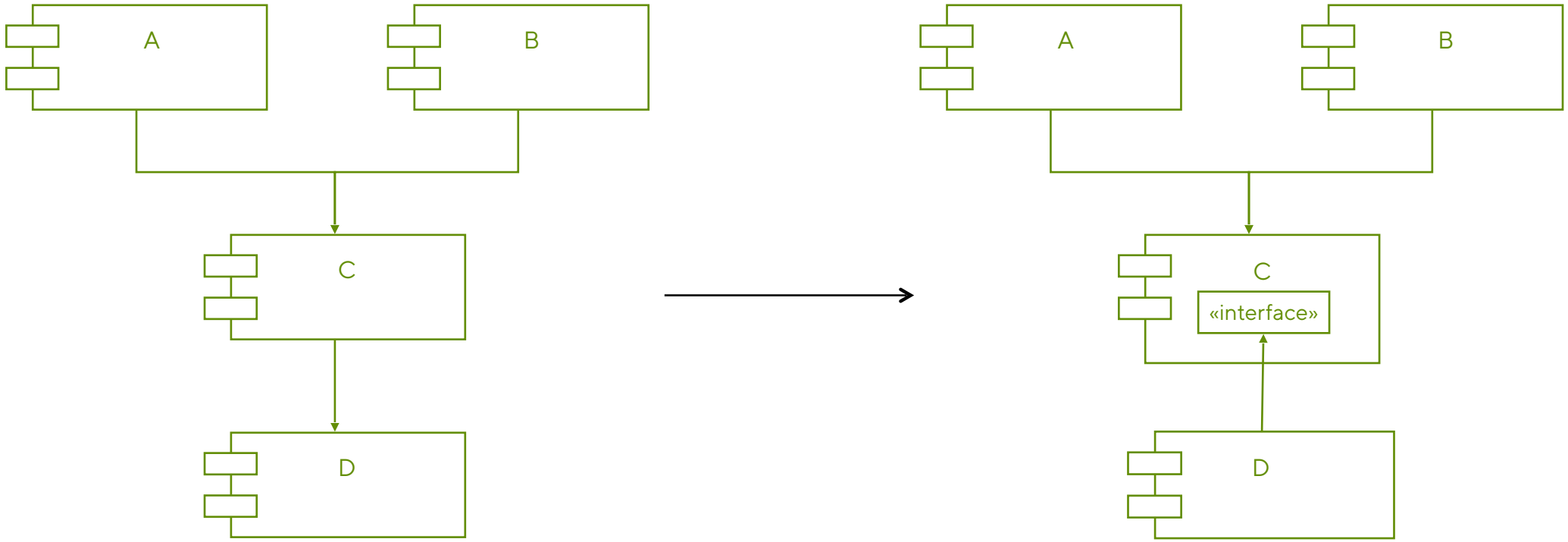
Нарушение Stable Dependency Principle

S = 1 – I = IN / (IN + OUT)

- Stable — изначально стабильный компонент.
- Flexible — первоначально изменчивый.
- ⇒ Кто-то модифицирует Stable, используя Flexible (было удобно).
- S stable > S flexible.
- Зависимости должны вести в направлении стабильности.
- ⇒ Нарушение SDP.



Решение — Инверсия зависимостей



Stable Abstraction Principle

- Стабильные компоненты должны быть абстрактными:
  - Чтобы ослабить связи зависимых от них компонент, определив явные интерфейсы-контракты.
- Нестабильные компоненты должны быть конкретными.

⇒ Зависимости должны вести в направлении стабильности.

⇒ Зависимости должны вести в направлении увеличения абстрактности.

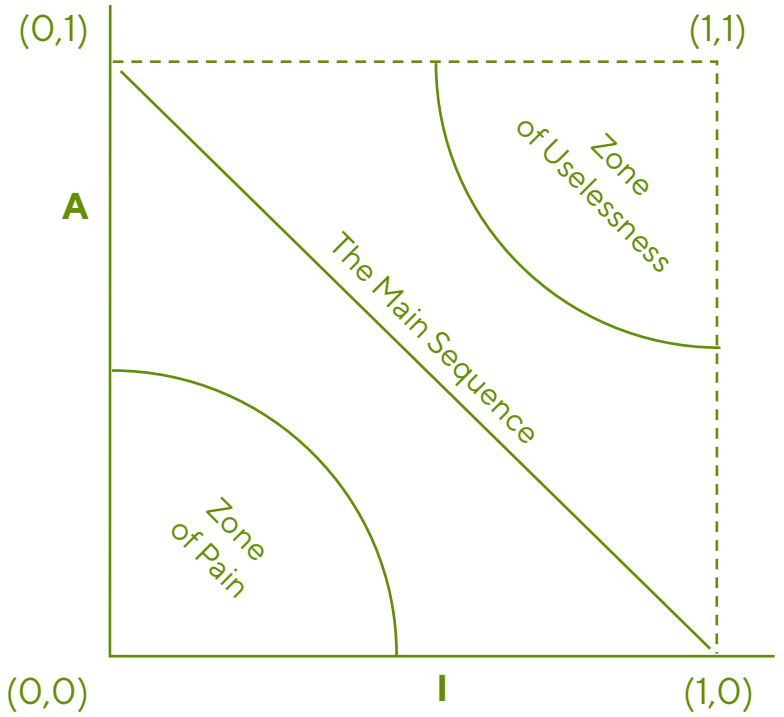
- Абстракция = Интерфейс = behaviour в elixir.

Метрика A

A = AbstractFiles / TotalFiles

Метрика D

- Главная последовательность.
- Зона боли — стабильные и конкретные компоненты.
- Зона бесполезности - нестабильные и абстрактные компоненты.
- Расстояние от главной последовательности
  D = |A + I - 1|

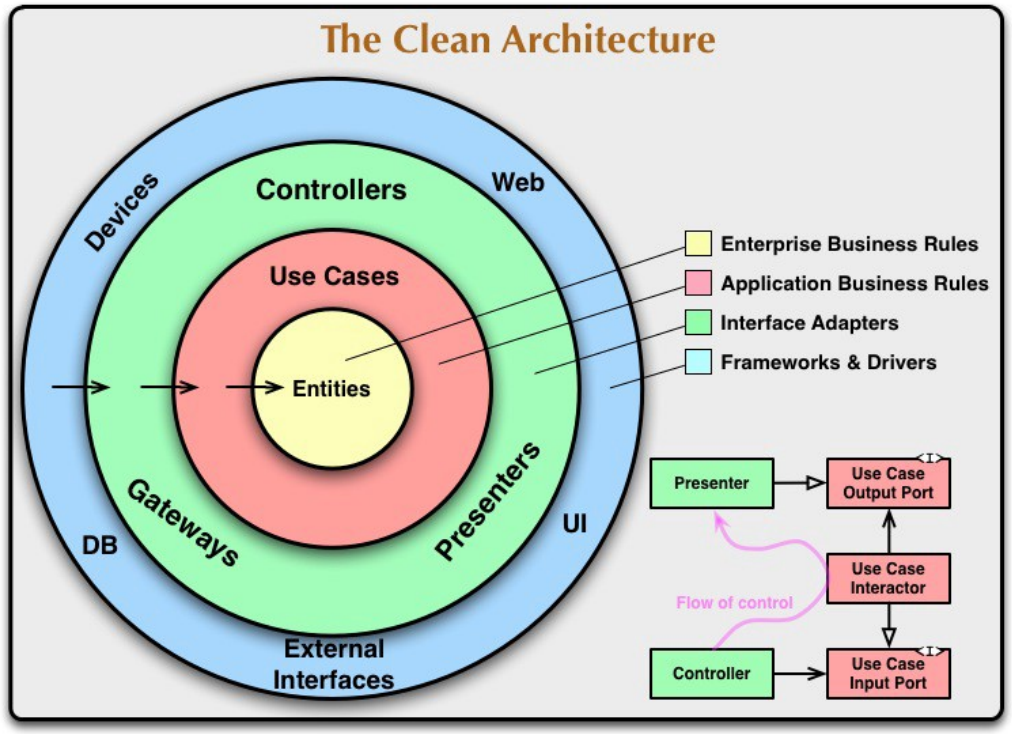


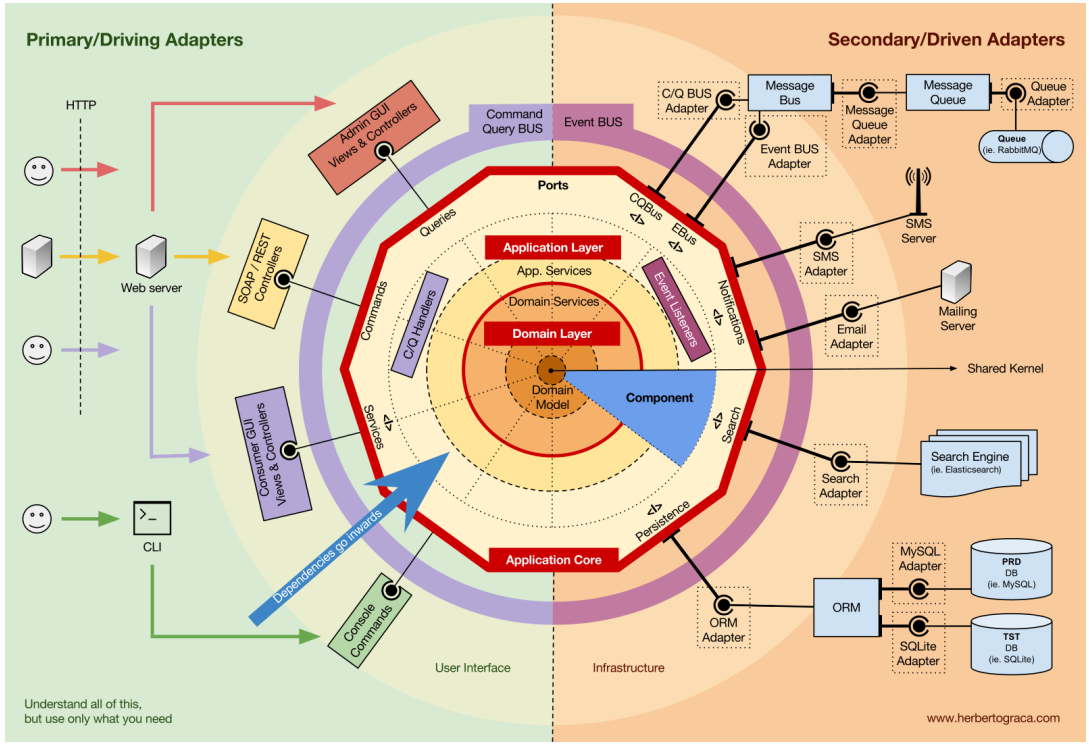
Неизменчивые компоненты

- Находятся в зоне боли, но не представляют опасности.
- Компоненты очень редко меняются.
- Примеры:
  - Стандартная библиотека языка.
  - Компоненты-утилиты (можно вынести в отдельную либу).

Clean (hexagonal) architecture

Домен в центре зависимостей и определяет интерфейсы (инверсия зависимостей).





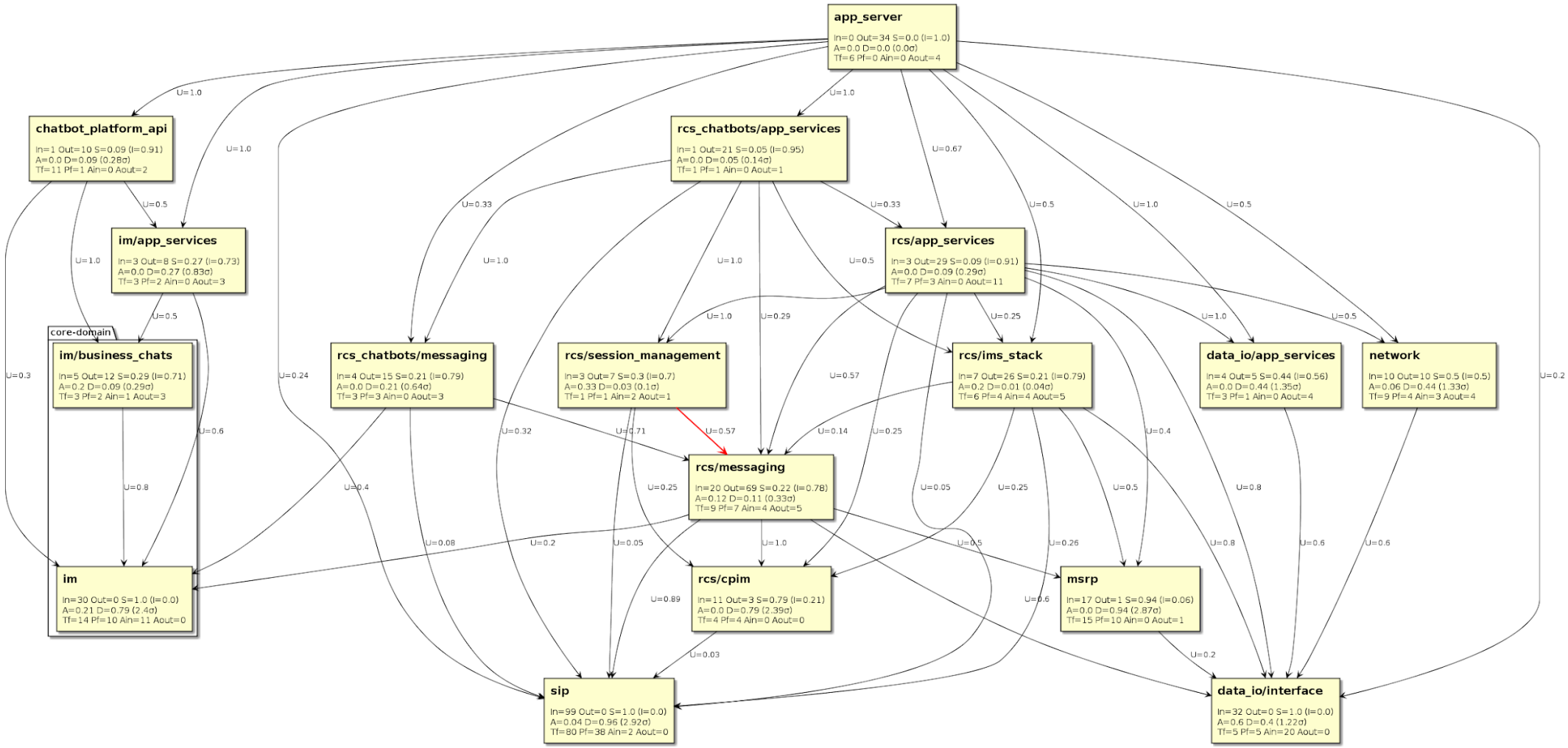
Стабильность домена

- Домен самая стабильная часть приложения (всё зависит от него).
- Но эти зависимости от домена должны быть от:
  - Стремящихся к неизменчивым структур данных (Domain Entities).
  - Абстрактных интерфейсов определяемых доменом.

Слои и Градиент стабильности

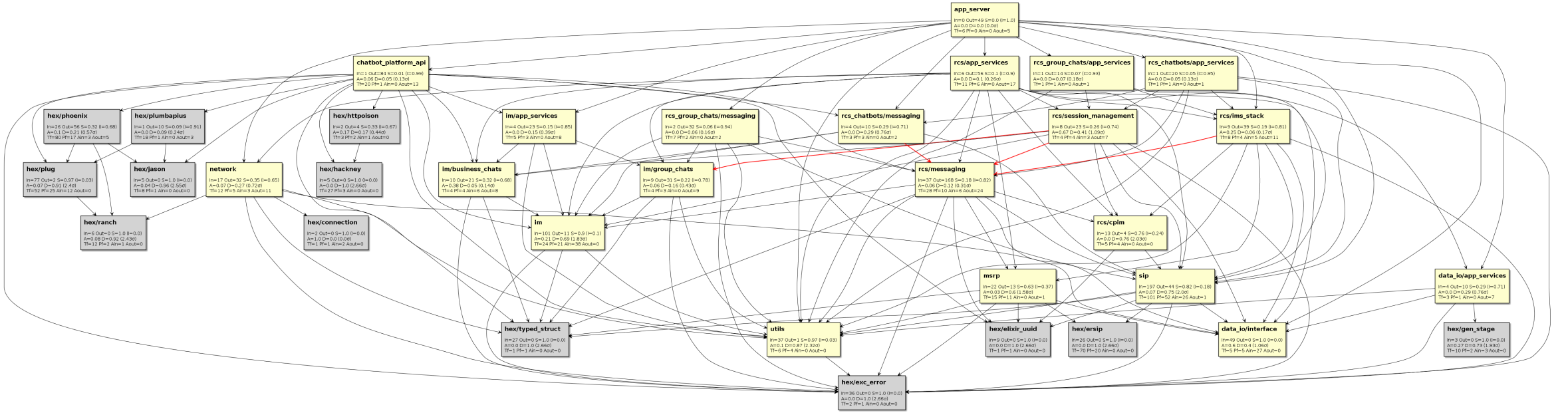
- Domain Entities = Structs.
- Domain Services = Use Cases.
- Application services = OTP behaviors, state management.
- Adapters = HTTP API, Kafka, IO.
- Reusability.
- Domain entities > Domain Services > Application Services > Adapters.
- Stability.
- Domain entities > Domain Services > Application Services > Adapters.

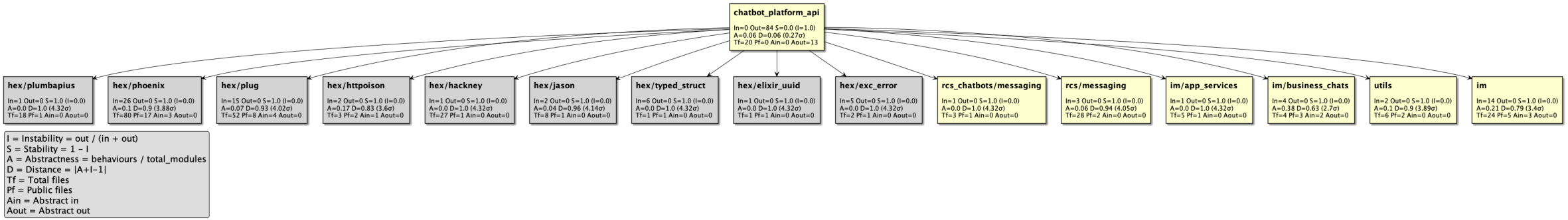
```
miros@localhost:~/pjs/rcs-messaging-server master >mix clean_mixer.plantuml -v --group --except=utils
image file created at clean_mixer.png
miros@localhost:~/pjs/rcs-messaging-server master >
```


app_server
In=0 Out=34 S=0.0 (I=1.0)
A=0.0 D=0.0 (0.0σ)
Tf=6 Pf=0 Ain=0 Aout=4
chatbot_platform_api
In=1 Out=10 S=0.09 (I=0.91)
A=0.0 D=0.09 (0.28σ)
Tf=11 Pf=1 Ain=0 Aout=2
rcs_chatbots/app_services
In=1 Out=21 S=0.05 (I=0.95)
A=0.0 D=0.05 (0.14σ)
Tf=1 Pf=1 Ain=0 Aout=1
im/app_services
In=3 Out=8 S=0.27 (I=0.73)
A=0.0 D=0.27 (0.83σ)
Tf=3 Pf=2 Ain=0 Aout=3
rcs/app_services
In=3 Out=29 S=0.09 (I=0.91)
A=0.0 D=0.09 (0.29σ)
Tf=7 Pf=3 Ain=0 Aout=11
core-domain
im/business_chats
In=5 Out=12 S=0.29 (I=0.71)
A=0.2 D=0.09 (0.29σ)
Tf=3 Pf=2 Ain=1 Aout=3
rcs_chatbots/messaging
In=4 Out=15 S=0.21 (I=0.79)
A=0.0 D=0.21 (0.64σ)
Tf=3 Pf=3 Ain=0 Aout=3
rcs/session_management
In=3 Out=7 S=0.3 (I=0.7)
A=0.33 D=0.03 (0.1σ)
Tf=1 Pf=1 Ain=2 Aout=1
rcs/ims_stack
In=7 Out=26 S=0.21 (I=0.79)
A=0.2 D=0.01 (0.04σ)
Tf=6 Pf=4 Ain=4 Aout=5
data_io/app_services
In=4 Out=5 S=0.44 (I=0.56)
A=0.0 D=0.44 (1.35σ)
Tf=3 Pf=1 Ain=0 Aout=4
network
In=10 Out=10 S=0.5 (I=0.5)
A=0.06 D=0.44 (1.33σ)
Tf=9 Pf=4 Ain=3 Aout=4
rcs/messaging
In=20 Out=69 S=0.22 (I=0.78)
A=0.12 D=0.11 (0.33σ)
Tf=9 Pf=7 Ain=4 Aout=5
im
In=30 Out=0 S=1.0 (I=0.0)
A=0.21 D=0.79 (2.4σ)
Tf=14 Pf=10 Ain=11 Aout=0
rcs/cpim
In=11 Out=3 S=0.79 (I=0.21)
A=0.0 D=0.79 (2.39σ)
Tf=4 Pf=4 Ain=0 Aout=0
msrp
In=17 Out=1 S=0.94 (I=0.06)
A=0.0 D=0.94 (2.87σ)
Tf=15 Pf=10 Ain=0 Aout=1
sip
In=99 Out=0 S=1.0 (I=0.0)
A=0.04 D=0.96 (2.92σ)
Tf=80 Pf=38 Ain=2 Aout=0
data_io/interface
In=32 Out=0 S=1.0 (I=0.0)
A=0.6 D=0.4 (1.22σ)
Tf=5 Pf=5 Ain=20 Aout=0
I = Instability = out / (in + out)
S = Stability = 1 - I
A = Abstractness = behaviours / total_modules
D = Distance = |A+I-1|
Tf = Total files
Pf = Public files
Ain = Abstract in
Aout = Abstract out
U = Usage = UsedFiles / Pf

Внешние зависимости

```
miros@miros-retina:~/pjs/rcs-messaging-server master * >mix clean_mixer.plantuml --except=utils --include-hex
image file created at clean_mixer.png
miros@miros-retina:~/pjs/rcs-messaging-server master * >
```


app_server
In=0 Out=49 S=0.0 (I=1.0)
A=0.0 D=0.0 (0.0σ)
Tf=6 Pf=0 Ain=0 Aout=5
chatbot_platform_api
In=1 Out=84 S=0.01 (I=0.99)
A=0.06 D=0.05 (0.13σ)
Tf=20 Pf=1 Ain=0 Aout=13
rcs/app_services
In=6 Out=56 S=0.1 (I=0.9)
A=0.0 D=0.1 (0.26σ)
Tf=11 Pf=6 Ain=0 Aout=17
rcs_group_chats/app_services
In=1 Out=14 S=0.07 (I=0.93)
A=0.0 D=0.07 (0.18σ)
Tf=1 Pf=1 Ain=0 Aout=1
rcs_chatbots/app_services
In=1 Out=20 S=0.05 (I=0.95)
A=0.0 D=0.05 (0.13σ)
Tf=1 Pf=1 Ain=0 Aout=1
hex/phoenix
In=26 Out=56 S=0.32 (I=0.68)
A=0.1 D=0.21 (0.57σ)
Tf=80 Pf=17 Ain=3 Aout=5
hex/plumbapius
In=1 Out=10 S=0.09 (I=0.91)
A=0.0 D=0.09 (0.24σ)
Tf=18 Pf=1 Ain=0 Aout=3
hex/httpoison
In=2 Out=4 S=0.33 (I=0.67)
A=0.17 D=0.17 (0.44σ)
Tf=3 Pf=2 Ain=1 Aout=0
im/app_services
In=4 Out=23 S=0.15 (I=0.85)
A=0.0 D=0.15 (0.39σ)
Tf=5 Pf=3 Ain=0 Aout=8
rcs_group_chats/messaging
In=2 Out=32 S=0.06 (I=0.94)
A=0.0 D=0.06 (0.16σ)
Tf=7 Pf=2 Ain=0 Aout=2
rcs_chatbots/messaging
In=4 Out=10 S=0.29 (I=0.71)
A=0.0 D=0.29 (0.76σ)
Tf=3 Pf=3 Ain=0 Aout=2
rcs/session_management
In=8 Out=23 S=0.26 (I=0.74)
A=0.67 D=0.41 (1.09σ)
Tf=4 Pf=4 Ain=3 Aout=7
rcs/ims_stack
In=9 Out=39 S=0.19 (I=0.81)
A=0.25 D=0.06 (0.17σ)
Tf=8 Pf=4 Ain=5 Aout=11
hex/plug
In=77 Out=2 S=0.97 (I=0.03)
A=0.07 D=0.91 (2.4σ)
Tf=52 Pf=25 Ain=12 Aout=0
hex/jason
In=5 Out=0 S=1.0 (I=0.0)
A=0.04 D=0.96 (2.55σ)
Tf=8 Pf=1 Ain=0 Aout=0
network
In=17 Out=32 S=0.35 (I=0.65)
A=0.07 D=0.27 (0.72σ)
Tf=12 Pf=5 Ain=3 Aout=11
hex/hackney
In=5 Out=0 S=1.0 (I=0.0)
A=0.0 D=1.0 (2.66σ)
Tf=27 Pf=3 Ain=0 Aout=0
im/business_chats
In=10 Out=21 S=0.32 (I=0.68)
A=0.38 D=0.05 (0.14σ)
Tf=4 Pf=4 Ain=6 Aout=8
im/group_chats
In=9 Out=31 S=0.22 (I=0.78)
A=0.06 D=0.16 (0.43σ)
Tf=4 Pf=3 Ain=0 Aout=9
rcs/messaging
In=37 Out=168 S=0.18 (I=0.82)
A=0.06 D=0.12 (0.31σ)
Tf=28 Pf=10 Ain=6 Aout=24
hex/ranch
In=6 Out=0 S=1.0 (I=0.0)
A=0.08 D=0.92 (2.43σ)
Tf=12 Pf=2 Ain=1 Aout=0
hex/connection
In=2 Out=0 S=1.0 (I=0.0)
A=1.0 D=0.0 (0.0σ)
Tf=1 Pf=1 Ain=2 Aout=0
im
In=101 Out=11 S=0.9 (I=0.1)
A=0.21 D=0.69 (1.83σ)
Tf=24 Pf=21 Ain=38 Aout=0
rcs/cpim
In=13 Out=4 S=0.76 (I=0.24)
A=0.0 D=0.76 (2.03σ)
Tf=5 Pf=4 Ain=0 Aout=0
msrp
In=22 Out=13 S=0.63 (I=0.37)
A=0.03 D=0.6 (1.58σ)
Tf=15 Pf=11 Ain=0 Aout=1
sip
In=197 Out=44 S=0.82 (I=0.18)
A=0.07 D=0.75 (2.0σ)
Tf=101 Pf=52 Ain=26 Aout=1
data_io/app_services
In=4 Out=10 S=0.29 (I=0.71)
A=0.0 D=0.29 (0.76σ)
Tf=3 Pf=1 Ain=0 Aout=7
hex/typed_struct
In=27 Out=0 S=1.0 (I=0.0)
A=0.0 D=1.0 (2.66σ)
Tf=1 Pf=1 Ain=0 Aout=0
utils
In=37 Out=1 S=0.97 (I=0.03)
A=0.1 D=0.87 (2.32σ)
Tf=6 Pf=4 Ain=0 Aout=0
hex/elixir_uuid
In=9 Out=0 S=1.0 (I=0.0)
A=0.0 D=1.0 (2.66σ)
Tf=1 Pf=1 Ain=0 Aout=0
hex/ersip
In=26 Out=0 S=1.0 (I=0.0)
A=0.0 D=1.0 (2.66σ)
Tf=70 Pf=20 Ain=0 Aout=0
data_io/interface
In=49 Out=0 S=1.0 (I=0.0)
A=0.6 D=0.4 (1.06σ)
Tf=5 Pf=5 Ain=27 Aout=0
hex/gen_stage
In=3 Out=0 S=1.0 (I=0.0)
A=0.27 D=0.73 (1.93σ)
Tf=10 Pf=2 Ain=3 Aout=0
hex/exc_error
In=36 Out=0 S=1.0 (I=0.0)
A=0.0 D=1.0 (2.66σ)
Tf=2 Pf=1 Ain=0 Aout=0
I = Instability = out / (in + out)
S = Stability = 1 - I
A = Abstractness = behaviours / total_modules
D = Distance = |A+I-1|
Tf = Total files
Pf = Public files
Ain = Abstract in
Aout = Abstract out


chatbot_platform_api
In=0 Out=84 S=0.0 (I=1.0)
A=0.06 D=0.06 (0.27σ)
Tf=20 Pf=0 Ain=0 Aout=13
hex/plumbapius
In=1 Out=0 S=1.0 (I=0.0)
A=0.0 D=1.0 (4.32σ)
Tf=18 Pf=1 Ain=0 Aout=0
hex/phoenix
In=26 Out=0 S=1.0 (I=0.0)
A=0.1 D=0.9 (3.88σ)
Tf=80 Pf=17 Ain=3 Aout=0
hex/plug
In=15 Out=0 S=1.0 (I=0.0)
A=0.07 D=0.93 (4.02σ)
Tf=52 Pf=8 Ain=4 Aout=0
hex/httpoison
In=2 Out=0 S=1.0 (I=0.0)
A=0.17 D=0.83 (3.6σ)
Tf=3 Pf=2 Ain=1 Aout=0
hex/hackney
In=1 Out=0 S=1.0 (I=0.0)
A=0.0 D=1.0 (4.32σ)
Tf=27 Pf=1 Ain=0 Aout=0
hex/jason
In=2 Out=0 S=1.0 (I=0.0)
A=0.04 D=0.96 (4.14σ)
Tf=8 Pf=1 Ain=0 Aout=0
hex/typed_struct
In=6 Out=0 S=1.0 (I=0.0)
A=0.0 D=1.0 (4.32σ)
Tf=1 Pf=1 Ain=0 Aout=0
hex/elixir_uuid
In=1 Out=0 S=1.0 (I=0.0)
A=0.0 D=1.0 (4.32σ)
Tf=1 Pf=1 Ain=0 Aout=0
hex/exc_error
In=5 Out=0 S=1.0 (I=0.0)
A=0.0 D=1.0 (4.32σ)
Tf=2 Pf=1 Ain=0 Aout=0
rcs_chatbots/messaging
In=1 Out=0 S=1.0 (I=0.0)
A=0.0 D=1.0 (4.32σ)
Tf=3 Pf=1 Ain=0 Aout=0
rcs/messaging
In=3 Out=0 S=1.0 (I=0.0)
A=0.06 D=0.94 (4.05σ)
Tf=28 Pf=2 Ain=0 Aout=0
im/app_services
In=1 Out=0 S=1.0 (I=0.0)
A=0.0 D=1.0 (4.32σ)
Tf=5 Pf=1 Ain=0 Aout=0
im/business_chats
In=4 Out=0 S=1.0 (I=0.0)
A=0.38 D=0.63 (2.7σ)
Tf=4 Pf=3 Ain=2 Aout=0
utils
In=2 Out=0 S=1.0 (I=0.0)
A=0.1 D=0.9 (3.89σ)
Tf=6 Pf=2 Ain=0 Aout=0
im
In=14 Out=0 S=1.0 (I=0.0)
A=0.21 D=0.79 (3.4σ)
Tf=24 Pf=5 Ain=3 Aout=0
I = Instability = out / (in + out)
S = Stability = 1 – I
A = Abstractness = behaviours / total_modules
D = Distance = |A+I–1|
Tf = Total files
Pf = Public files
Ain = Abstract in
Aout = Abstract out

Контроль за регрессиями архитектуры

- Автоматизированные тесты.
- Проверяют допустимые связи между компонентами.
- Java: ArchUnit.
- C#: NetArchTest.
- Elixir: CleanMixer.

test > arch_test.exs > ...

```elixir
ExUnit.start(capture_log: true, trace: true)

defmodule RCSMessagingServer.Architecture.ArchTest do
  use ExUnit.Case

  alias CleanMixer.Workspace

  @domain [
    "im",
    "im/business_chats"
  ]

  @core_infrastructure [
    "rcs/messaging",
    "rcs/session_management",
    "rcs/ims_stack",
    "rcs/cpim",
    "rcs_chatbots/messaging",
    "sip",
    "msrp",
    "data_io/interface"
  ]

  @application_services [
    "rcs/app_services",
    "rcs_chatbots/app_services",
    "im/app_services",
    "data_io/app_services"
  ]

@adapters [
  "chatbot_platform_api",
  "app_server",
  "network"
]

# special cases

@interface_components [
  "data_io/interface"
]

@protocol_components [
  "sip",
  "msrp"
]

setup_all do
  workspace = Workspace.new()
  %{ws: workspace}
end
```

Stability

S domain

>

S core infrastructure

>

S application services

>

S adapters

```elixir
defp aliases do
  [
    arch_test: ["run --no-start test/arch_test.exs"]
  ]
end

defp dependencies_of(workspace, name) do
  Workspace.dependencies_of(workspace, name) |> Enum.map(& &1.target)
end

defp dependency?(workspace, source_name, target_name) do
  Workspace.dependency?(workspace, source_name, target_name)
end

defp component(workspace, name) do
  Workspace.component(workspace, name)
end

defp format_cycle(cycle) do
  cycle |> Enum.map(& &1.name) |> Enum.join(" -> ")
end
```

```elixir
test "there are shall be no cyclical dependencies between components", %{ws: ws} do
  assert Workspace.component_cycles(ws) |> Enum.map(&format_cycle/1) == []
end

for comp <- @domain, app_service <- @application_services do
  test "domain component #{comp} shall not depend on application service #{app_service}", %{ws: ws} do
    refute dependency?(ws, unquote(comp), unquote(app_service))
  end
end

for comp <- @domain, adapter <- @adapters do
  test "domain component #{comp} shall not depend on adapter #{adapter}", %{ws: ws} do
    refute dependency?(ws, unquote(comp), unquote(adapter))
  end
end

for comp <- @core_infrastructure, app_service <- @application_services do
  test "core infrastracture component #{comp} shall not depend on application service #{app_service}", %{ws: ws} do
    refute dependency?(ws, unquote(comp), unquote(app_service))
  end
end

for comp <- @core_infrastructure, adapter <- @adapters do
  test "core infrastracture component #{comp} shall not depend on adapter #{adapter}", %{ws: ws} do
    refute dependency?(ws, unquote(comp), unquote(adapter))
  end
end
```

```elixir
for comp <- @core_infrastructure, app_service <- @application_services do
  test "core infrastracture component #{comp} shall not depend on application service #{app_service}", %{ws: ws} do
    refute dependency?(ws, unquote(comp), unquote(app_service))
  end
end

for comp <- @core_infrastructure, adapter <- @adapters do
  test "core infrastracture component #{comp} shall not depend on adapter #{adapter}", %{ws: ws} do
    refute dependency?(ws, unquote(comp), unquote(adapter))
  end
end

for comp <- @interface_components do
  test "interface component #{comp} shall have no dependencies", %{ws: ws} do
    assert dependencies_of(ws, unquote(comp)) == []
  end
end

for comp <- @protocol_components do
  test "protocol component #{comp} shall depend only on utils or interface components", %{ws: ws} do
    allowed_components = [component(ws, "utils")] ++ Enum.map(@interface_components, &component(ws, &1))

    assert dependencies_of(ws, unquote(comp)) -- allowed_components == []
  end
end

test "`utils` shall have no dependencies", %{ws: ws} do
  assert dependencies_of(ws, "utils") == []
end
```

```
miros@miros-retina:~/pjs/rcs-messaging-server clean-mixer-demo >mix arch_test

RCSMessagingServer.Architecture.ArchTest
  * test core infrastracture component sip shall not depend on adapter chatbot_platform_api (12.8ms)
  * test core infrastracture component rcs/session_management shall not depend on adapter app_server (6.8ms)
  * test core infrastracture component rcs/ims_stack shall not depend on adapter network (7.1ms)
  * test core infrastracture component msrp shall not depend on application service data_io/app_services (0.8ms)
  * test domain component im/business_chats shall not depend on adapter network (0.5ms)
  * test core infrastracture component sip shall not depend on application service rcs_chatbots/app_services (10.1ms)
  * test core infrastracture component rcs/ims_stack shall not depend on application service data_io/app_services (5.3
)
  * test core infrastracture component sip shall not depend on application service im/app_services (33.1ms)
  * test core infrastracture component rcs/session_management shall not depend on application service rcs/app_services
4.5ms)
  * test core infrastracture component data_io/interface shall not depend on application service rcs_chatbots/app_service
  * test core infrastracture component data_io/interface shall not depend on application service rcs_chatbots/app_service
es (0.3ms)
  * test core infrastracture component msrp shall not depend on adapter network (1.0ms)
  * test there are shall be no cyclical dependencies between components (660.5ms)
  * test core infrastracture component data_io/interface shall not depend on adapter app_server (0.3ms)
  * test domain component im shall not depend on application service rcs/app_services (0.6ms)
  * test domain component im/business_chats shall not depend on adapter app_server (0.6ms)
  * test core infrastracture component data_io/interface shall not depend on application service data_io/app_services
.1ms)
  * test core infrastracture component rcs/cpim shall not depend on application service rcs_chatbots/app_services (4.9
)
  * test core infrastracture component rcs/session_management shall not depend on adapter network (4.4ms)
  * test core infrastracture component rcs_chatbots/messaging shall not depend on adapter network (6.5ms)
  * test core infrastracture component rcs/messaging shall not depend on application service data_io/app_services (18.
s)
  * test core infrastracture component rcs/cpim shall not depend on application service im/app_services (3.5ms)
  * test core infrastracture component rcs/messaging shall not depend on application service im/app_services (11.1ms)
  * test domain component im shall not depend on application service rcs_chatbots/app_services (0.6ms)
```

```
  * test core infrastructure component rcs/ims_stack shall not depend on application service rcs/app_services (7.3ms)
  * test core infrastructure component rcs_chatbots/messaging shall not depend on adapter chatbot_platform_api (4.5ms)
  * test interface component data_io/interface shall have no dependencies (0.1ms)
  * test core infrastructure component rcs/messaging shall not depend on adapter network (12.7ms)
  * test core infrastructure component msrp shall not depend on application service rcs/app_services (0.7ms)
  * test core infrastructure component rcs/ims_stack shall not depend on application service im/app_services (4.6ms)
  * test core infrastructure component rcs/cpim shall not depend on adapter chatbot_platform_api (3.0ms)
  * test core infrastructure component msrp shall not depend on adapter app_server (0.8ms)
  * test core infrastructure component msrp shall not depend on adapter chatbot_platform_api (0.7ms)
  * test core infrastructure component rcs/messaging shall not depend on application service rcs/app_services (15.0ms)
  * test domain component im shall not depend on application service data_io/app_services (0.2ms)
  * test core infrastructure component rcs_chatbots/messaging shall not depend on application service rcs/app_services (
4.0ms)
  * test core infrastructure component rcs_chatbots/messaging shall not depend on application service im/app_services (4
.7ms)
  * test `utils` shall have no dependencies (0.03ms)
  * test core infrastructure component rcs/cpim shall not depend on application service data_io/app_services (2.8ms)
  * test core infrastructure component sip shall not depend on adapter app_server (7.3ms)
  * test core infrastructure component msrp shall not depend on application service rcs_chatbots/app_services (0.8ms)
  * test domain component im/business_chats shall not depend on application service im/app_services (0.3ms)
  * test core infrastructure component data_io/interface shall not depend on adapter chatbot_platform_api (0.2ms)
  * test domain component im shall not depend on application service im/app_services (0.2ms)
  * test domain component im shall not depend on adapter network (0.3ms)
  * test core infrastructure component msrp shall not depend on application service im/app_services (0.8ms)
  * test core infrastructure component rcs/session_management shall not depend on application service im/app_services (3
.7ms)


Finished in 1.5 seconds (0.2s on load, 1.2s on tests)
75 tests, 0 failures

Randomized with seed 432117
```

```
pipeline {

    ...

    post {
        always {
            archiveArtifacts artifacts: 'clean_mixer_*', fingerprint: true
        }
    }
}
```

Build #166 (Apr 7, 2020 2:16:41 PM)

Build Artifacts

- clean_mixer_62e3c9f.plantuml 4.46 KB view
- clean_mixer_62e3c9f.png 283.24 KB view
- clean_mixer_62e3c9f.txt 24.04 KB view

Changes

1. [RCSD-1011] Реализована отправка сообщений в неустановленную сессию. (details / bitbucketweb)
2. [RCSD-1011] Добавлен тест RcsSession. (details / bitbucketweb)
3. [RCSD-1011] Добавлен тест RcsChatbotSession. (details / bitbucketweb)
4. [RCSD-1011] Добавлен тест BusinessChat. (details / bitbucketweb)
5. [RCSD-1011] Сделано явное отображение создания новой сессии. (details / bitbucketweb)
6. [RCSD-1011] Убран шумный матчинг. (details / bitbucketweb)
7. [RCSD-1011] Изменен колбэк и заменен = на <- в with. (details / bitbucketweb)
8. [RCSD-1011] Возврат ошибки для всех состояний кроме Established. (details / bitbucketweb)

Branch indexing

This run spent:

- 4 ms waiting;
- 5 min 31 sec build duration;
- 5 min 31 sec total from scheduled to completion.

Revision: 62e3c9f10c6e028a203d2921da0b2c9a866de457

- master

Small Success Story

test core infrastracture component rcs_chatbots/messaging shall not depend
on adapter chatbot_platform_api (RCSMessagingServer.Architecture.ArchTest)

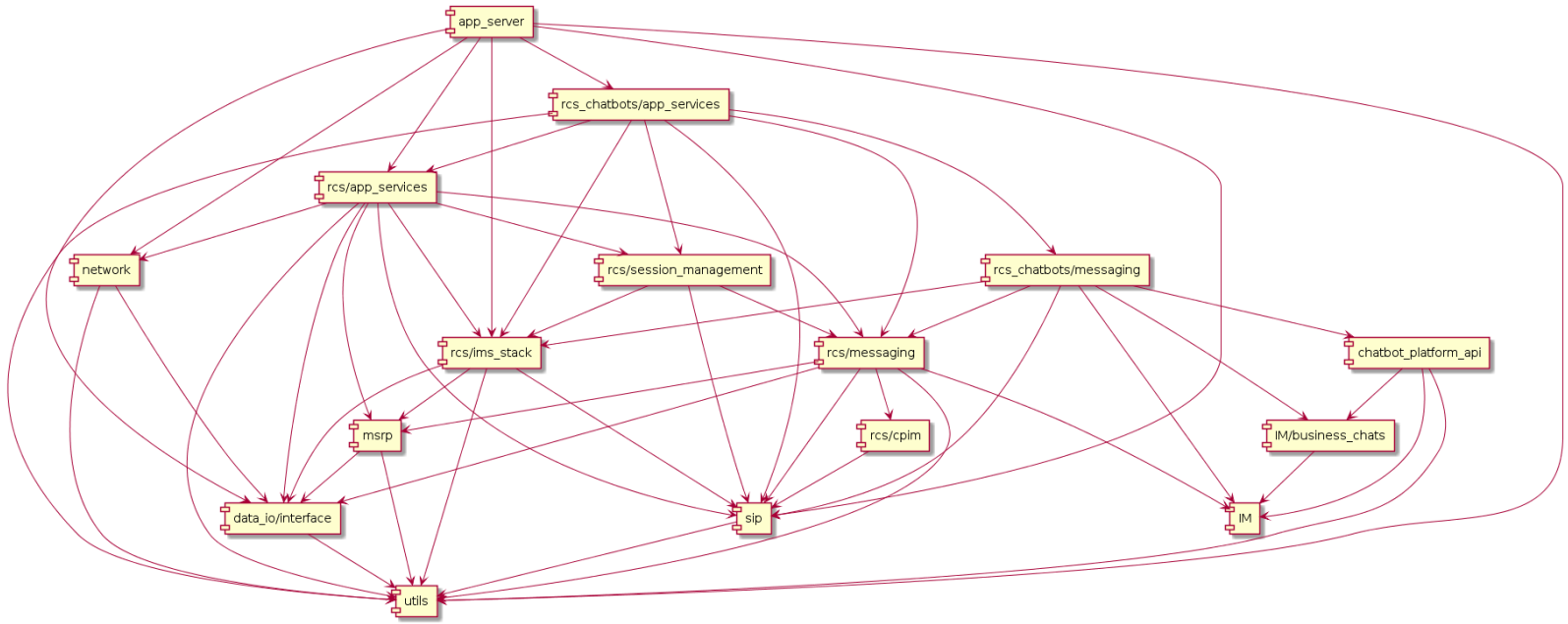
test/arch_test.exs:59

Expected false or nil, got true

code: refute dependency?("rcs_chatbots/messaging", "chatbot_platform_api")

stacktrace:

test/arch_test.exs:60: (test)



Микросервисы против монолита

Достоинства микросервисов

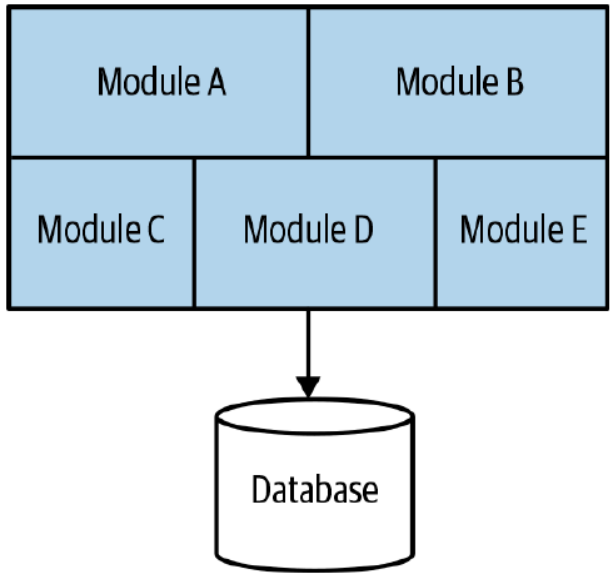
- Маленькие понятные кусочки по Bounded Contexts.
- Автономность (деплоя и разработки).
- Нефункциональная гибкость (масштабирование, отказоустойчивость, выбор технологий).

Недостатки микросервисов

- Распределенный монолит (неудачная декомпозиция на сервисы).
- Сложность эксплуатации (логи, трейсинг запросов).
- Проблемы распределенных систем (транзакции, синхронизация, недоступность компонент).

Модульный монолит

Отдельные компоненты (Bounded Contexts) в рамках единого процесса ОС и одной единицы деплоя



Достоинства

- Простота: если предметная область сложная, то зачем выбирать еще и сложную архитектуру?
- Understandability и параллельность разработки сильно выше чем у big ball of mud.
- Дает больше гибкости на первых этапах (легче изменить решения).

Недостатки

- Нужен автоматизированный контроль (Clean Mixer).
- Очень сложно избежать зависимостей компонент по БД (Решение: свое хранилище на каждый модуль).

Резюме

1. Запрещайте циклические зависимости.
2. Разделяйте домен и инфраструктуру.
3. Используйте абстракции: явно определяя интерфейсы через behaviour.
4. Управляйте направлениями зависимостей (Dependency Inversion), строя зависимости от интерфейсов, а не от реализаций.
5. Визуализируйте.
6. Автоматизируйте контроль.
7. Усложняйте архитектуру эволюционно (но будьте готовы к изменениям).