
Palestra #32
embarcadero
Um só código!
Como escrever
aplicações realmente
flexíveis?
Francisco Thiago
de Almeida
Da sala de reunião à vida real
Embarcadero Conference 2019

Quem sou?

ID #32

- Desenvolvedor Delphi há 15 anos
- Setor Público e Privado
- Comércio, Transporte, Trabalho e Justiça
- Coordenador do meetup Delphi Ingá
- Comitê técnico DB1 Group
- Colaborador DB1 Global Software
- 2x palestrante na ECon

Agenda

ID #32

- Por que eu preciso de uma arquitetura flexível?
- O que é Arquitetura hexagonal?
- É possível em Delphi? Como?
- Código!
- Perguntas

Palestra #32

embarcadero

Por que uma arquitetura flexível?

Da sala de reunião à vida real

Embarcadero Conference 2019

Por que Arq. Flexível?

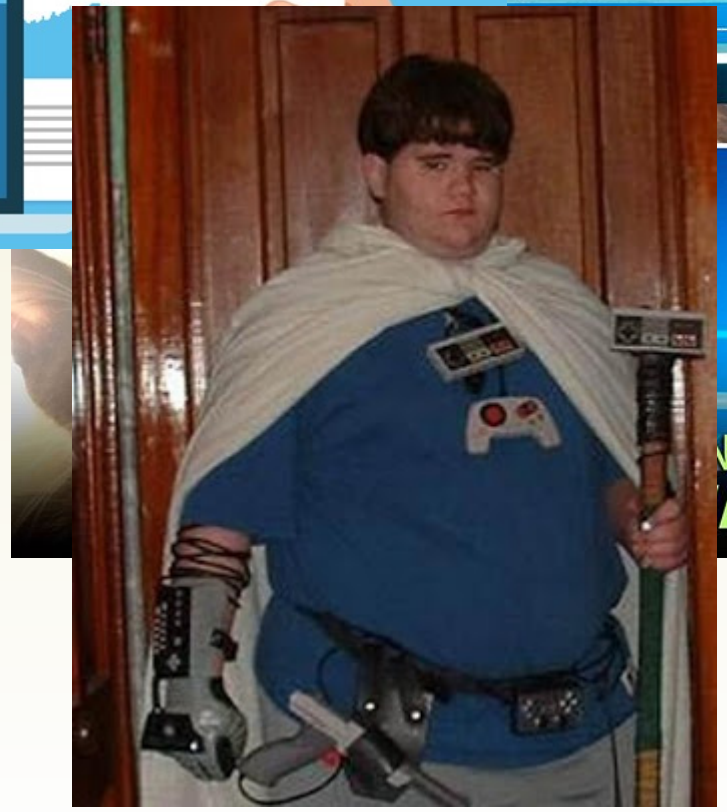
ID #32

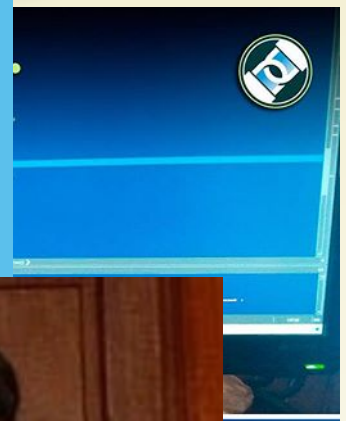
Por que Arq. Flexível?

ID #32

Por que Arq. Flexível?

ID #32

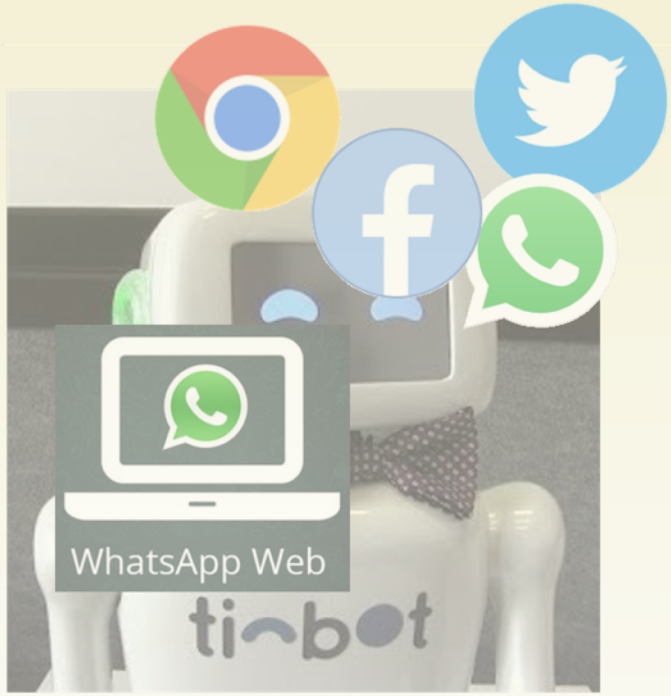




embarcadero

Por que Arq. Flexível?

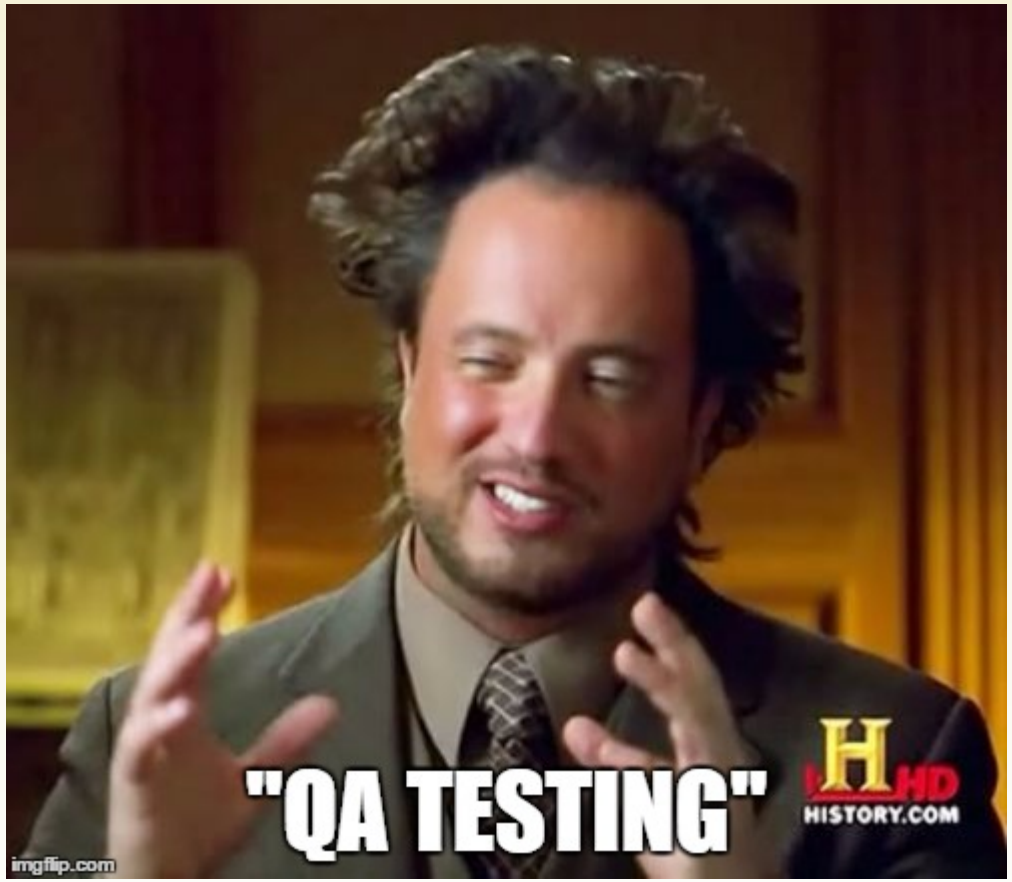
ID #32



Quem vai **testar** tudo isso?

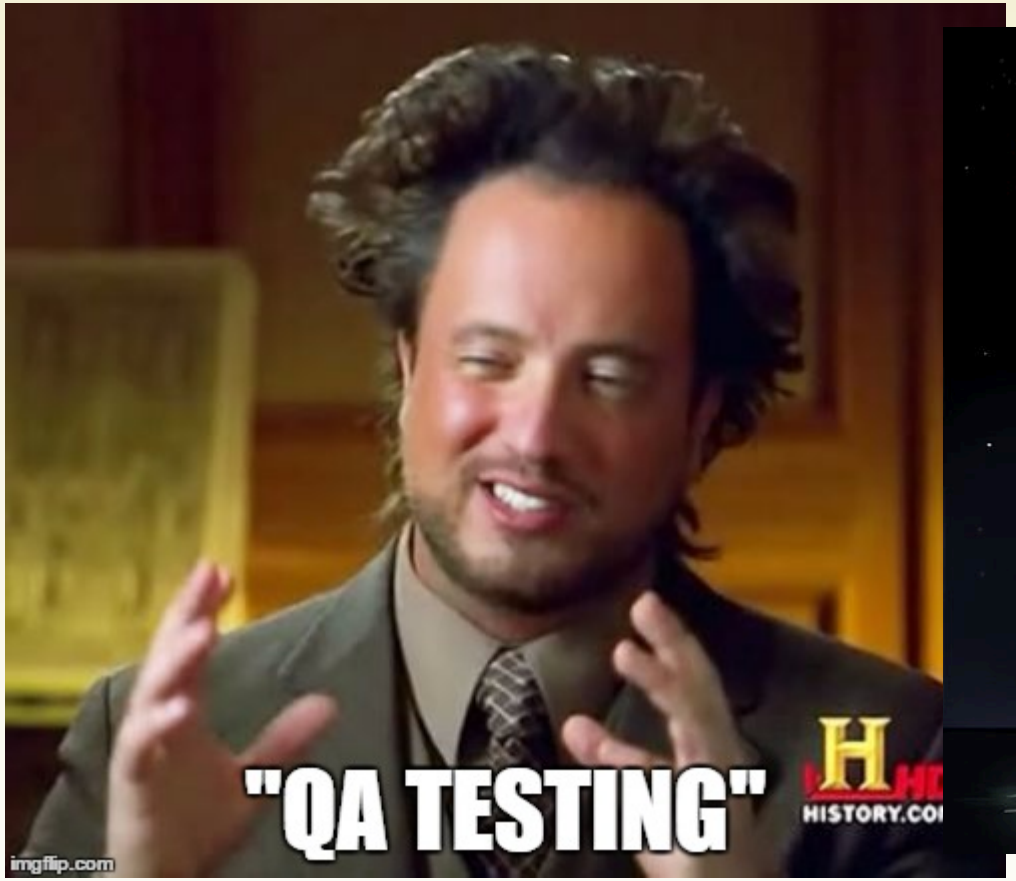
Por que Arq. Flexível?

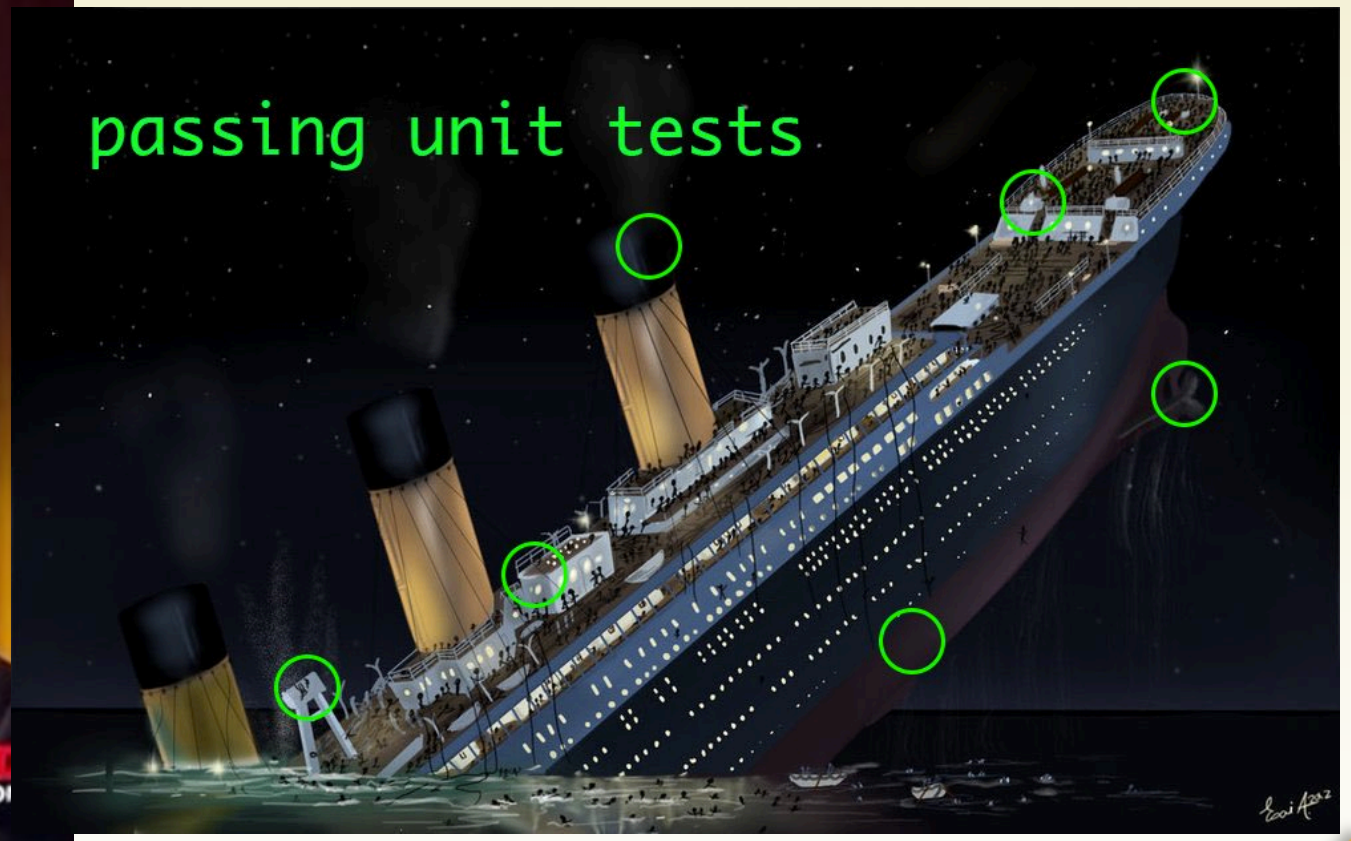
ID #32



Por que Arq. Flexível?

ID #32





Por que Arq. Flexível?

ID #32

Quanto **tempo**
você quer durar
no *mercado*?

Por que Arq. Flexível?

ID #32

Quanto tempo até o seu cliente demandar o software em todas essas interfaces?

Você não pensou a mesma coisa dos Palm? Dos Smartphones?

Por que Arq. Flexível?

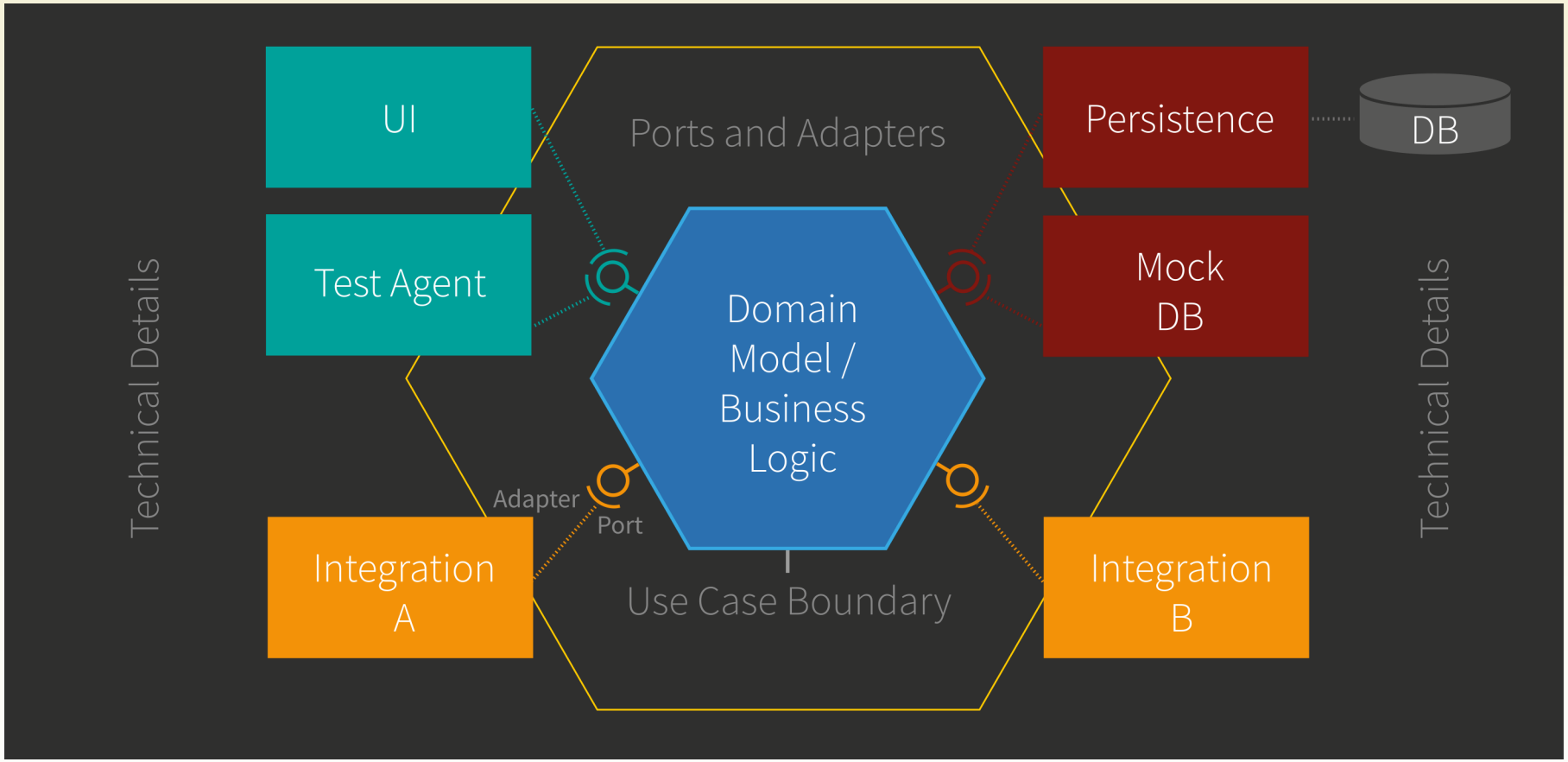
ID #32

{ REST }

embarcadero

Por que Arq. Flexível?

ID #32



Palestra #32

embarcadero

O que é a arquitetura hexagonal?

Da sala de reunião à vida real

Embarcadero Conference 2019

O que é? - Disclaimer

ID #32

- Na internet... Tá tudo misturado
- DDD, Onion, Clean Arch, Patterns
- Não acaba aqui, mas continua nas redes!

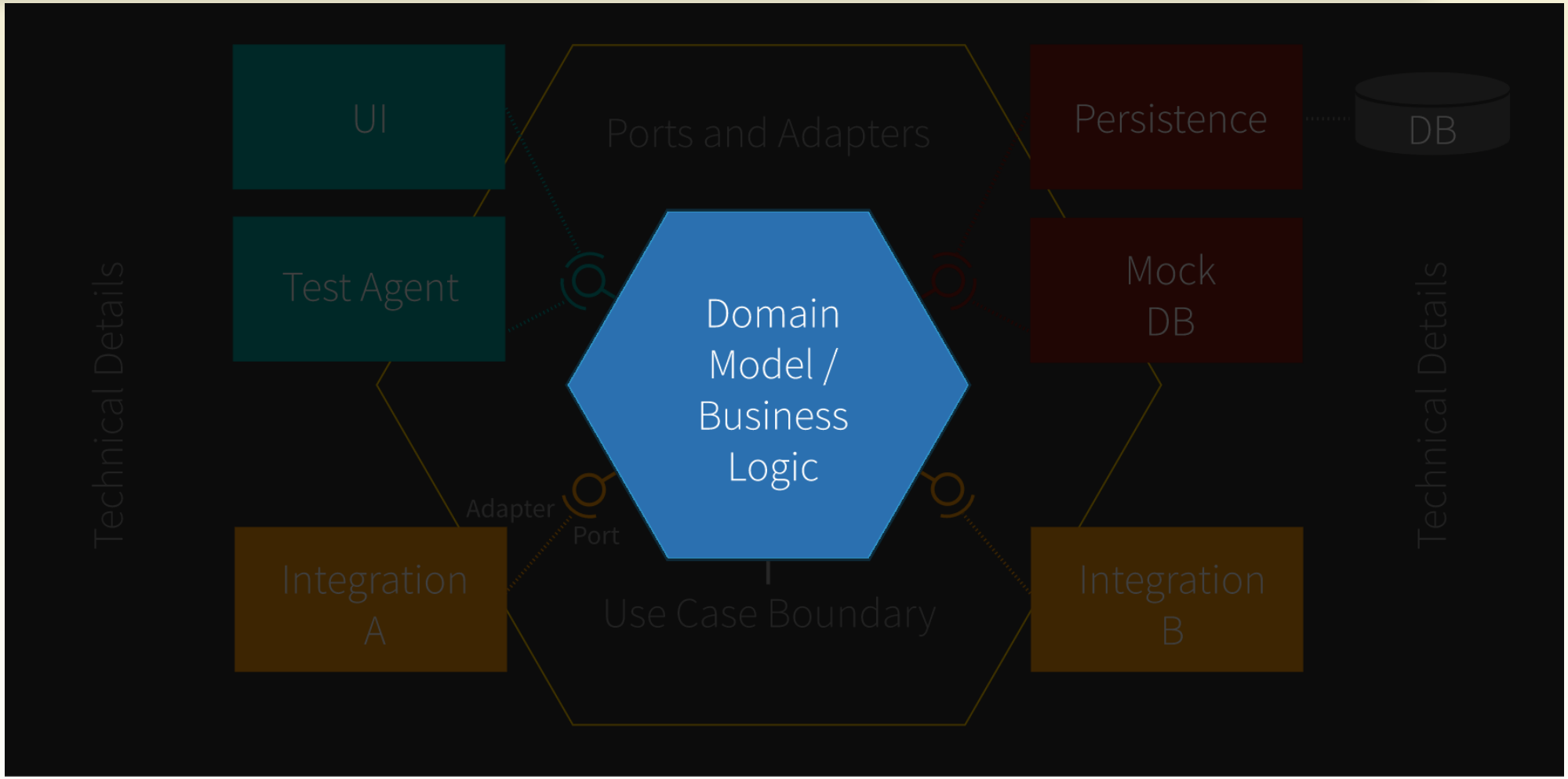
O que é?

ID #32

- Quer resolver
  - Mistura => Separação de interesses
  - Duplicação de código => Reaproveitamento
  - Acoplamento => Desacoplamento
  - Inflexível => Flexível

O que é?

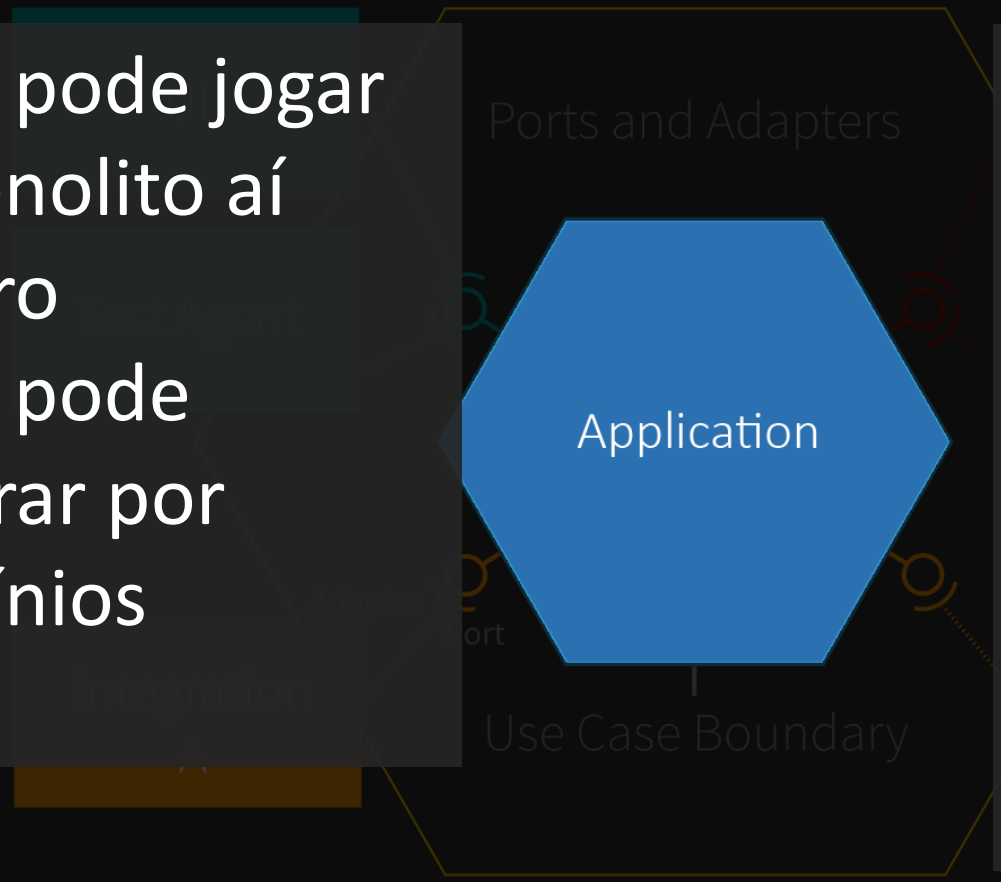
ID #32



O que é?

ID #32

- Você pode jogar o monolito aí dentro
- Você pode separar por domínios

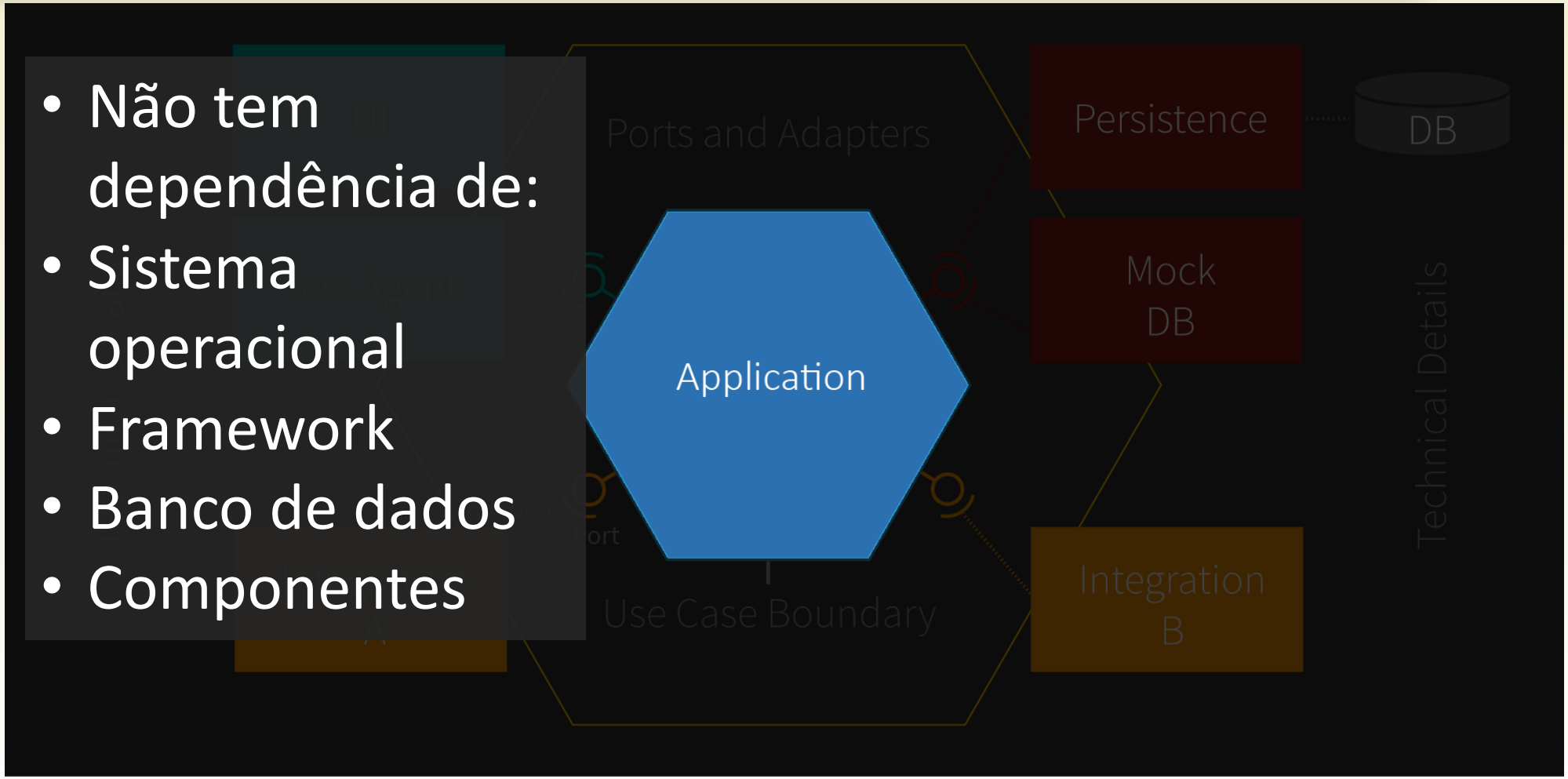


- Monolito: camada de Aplicação coordena as chamadas
- Domínios trabalha com eventos

O que é?

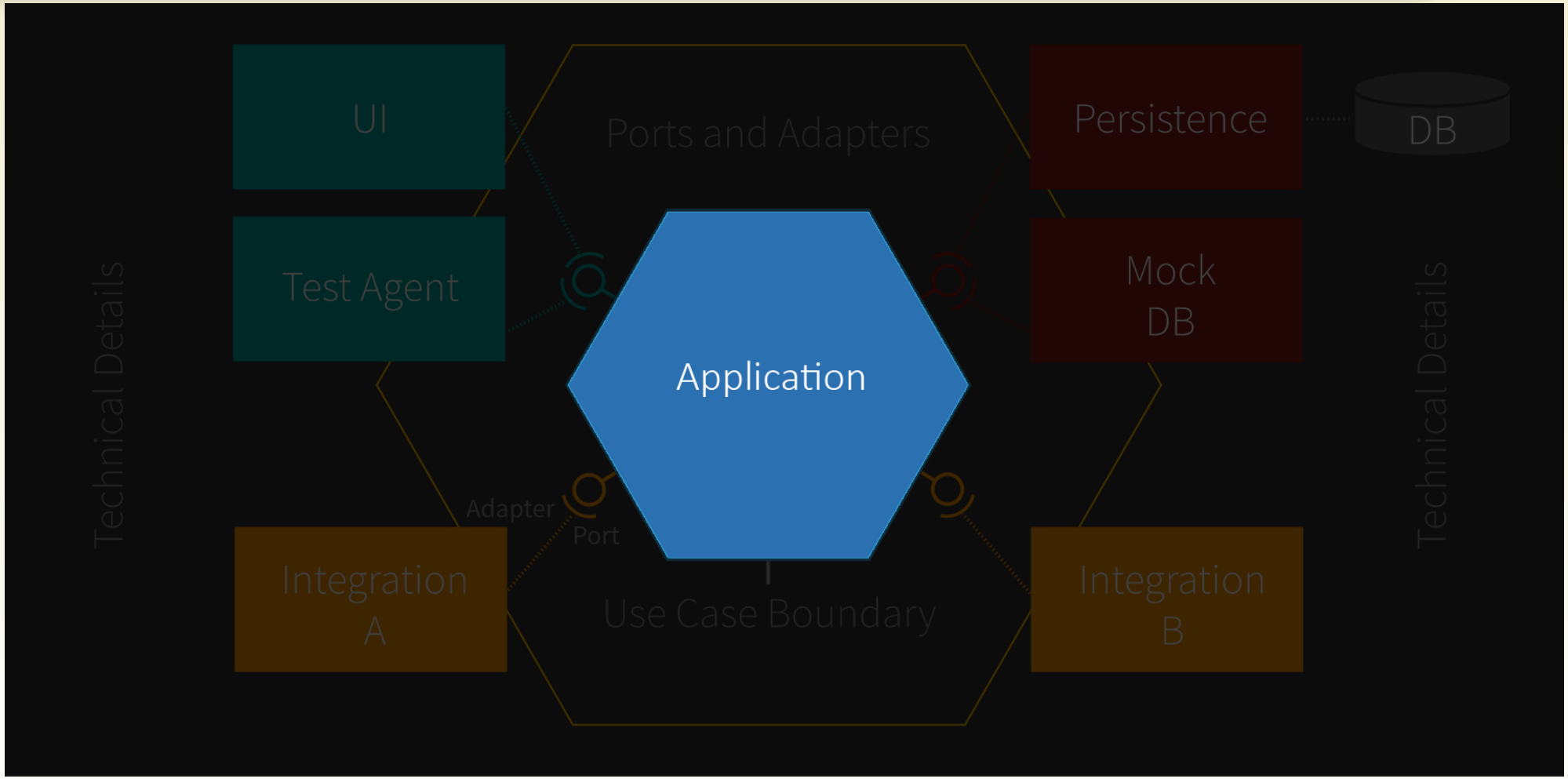
ID #32

- Não tem dependência de:
- Sistema operacional
- Framework
- Banco de dados
- Componentes



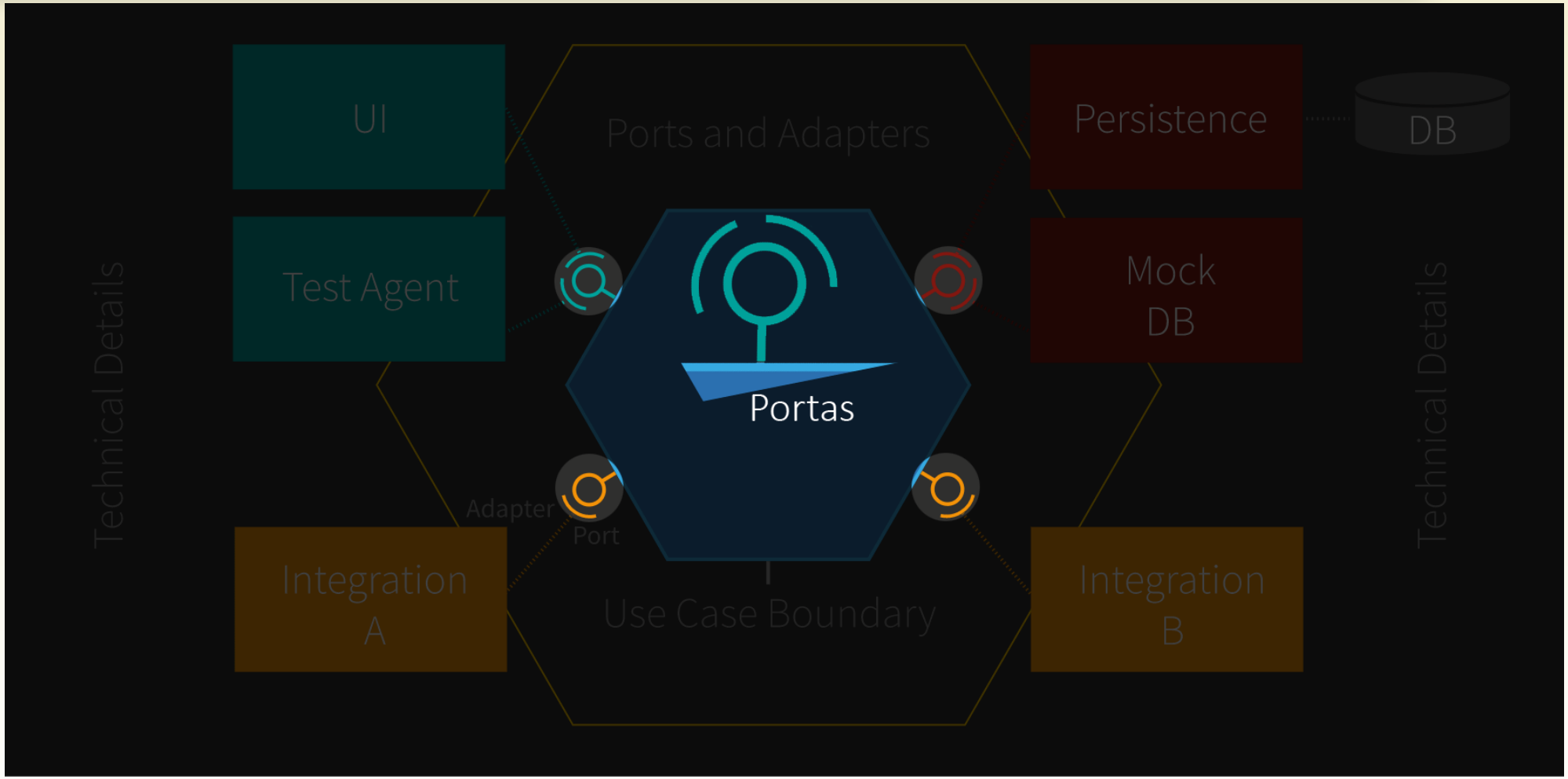
O que é?

ID #32



O que é?

ID #32



O que é?

ID #32

Você sabe o **que é** e **para quê serve** uma **interface**?

O que é?

ID #32

```
1  unit Unit2;
2
3  interface
4
5  implementation
6
7  end.
8
```

O que é?

ID #32

```
type
  IEhMuitoChatoInterface= Interface(IInterface)
    ['{6FE0CB75-DD22-44B0-8757-F1BDA9932238}']
    procedure MasEhSoVantagem;
  end;
```

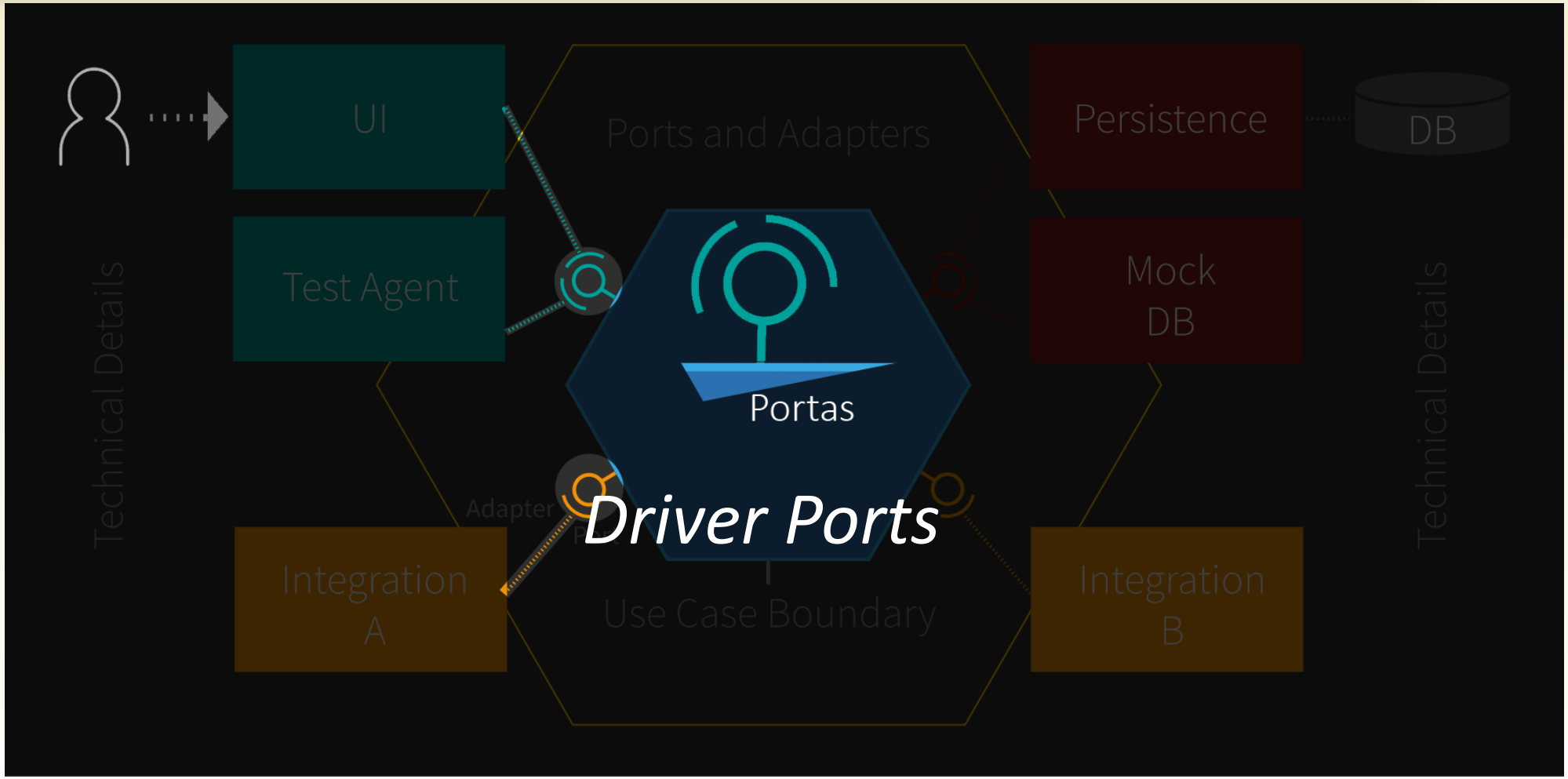
O que é?

ID #32

Interface: elemento que proporciona uma **ligação** física ou lógica **entre dois sistemas** ou partes de um sistema que *não poderiam* ser conectados *diretamente*.

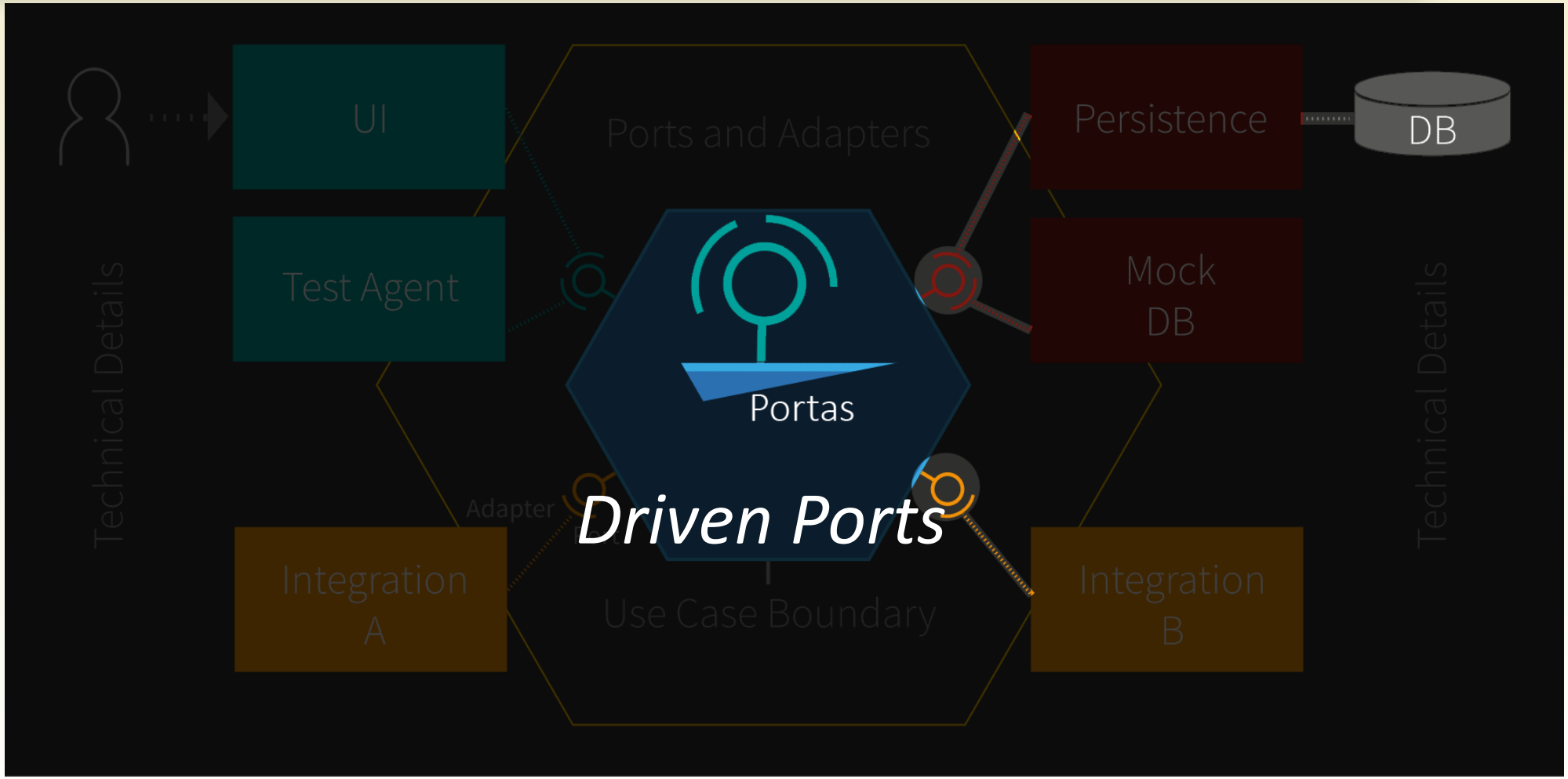
O que é?

ID #32



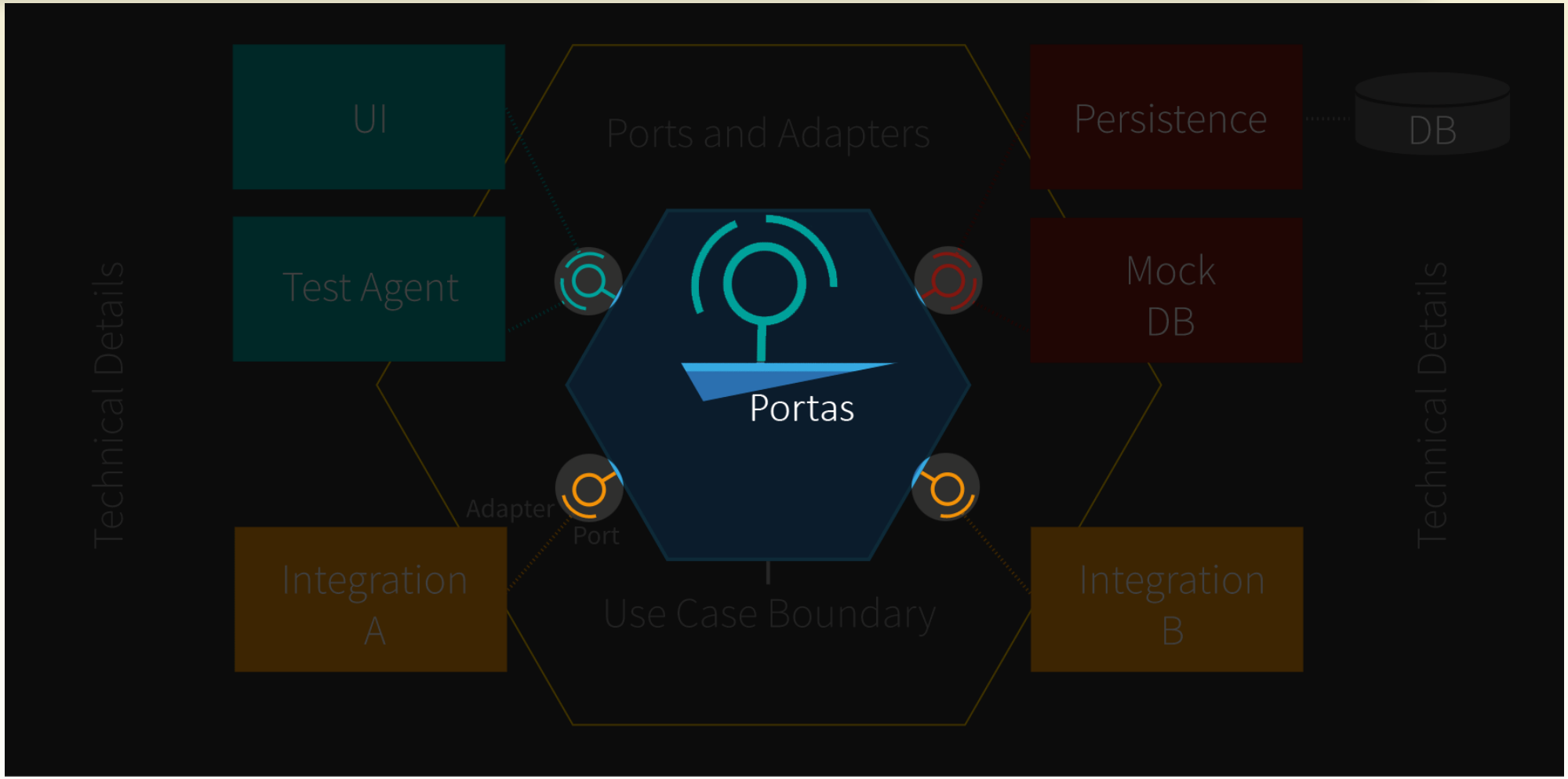
O que é?

ID #32



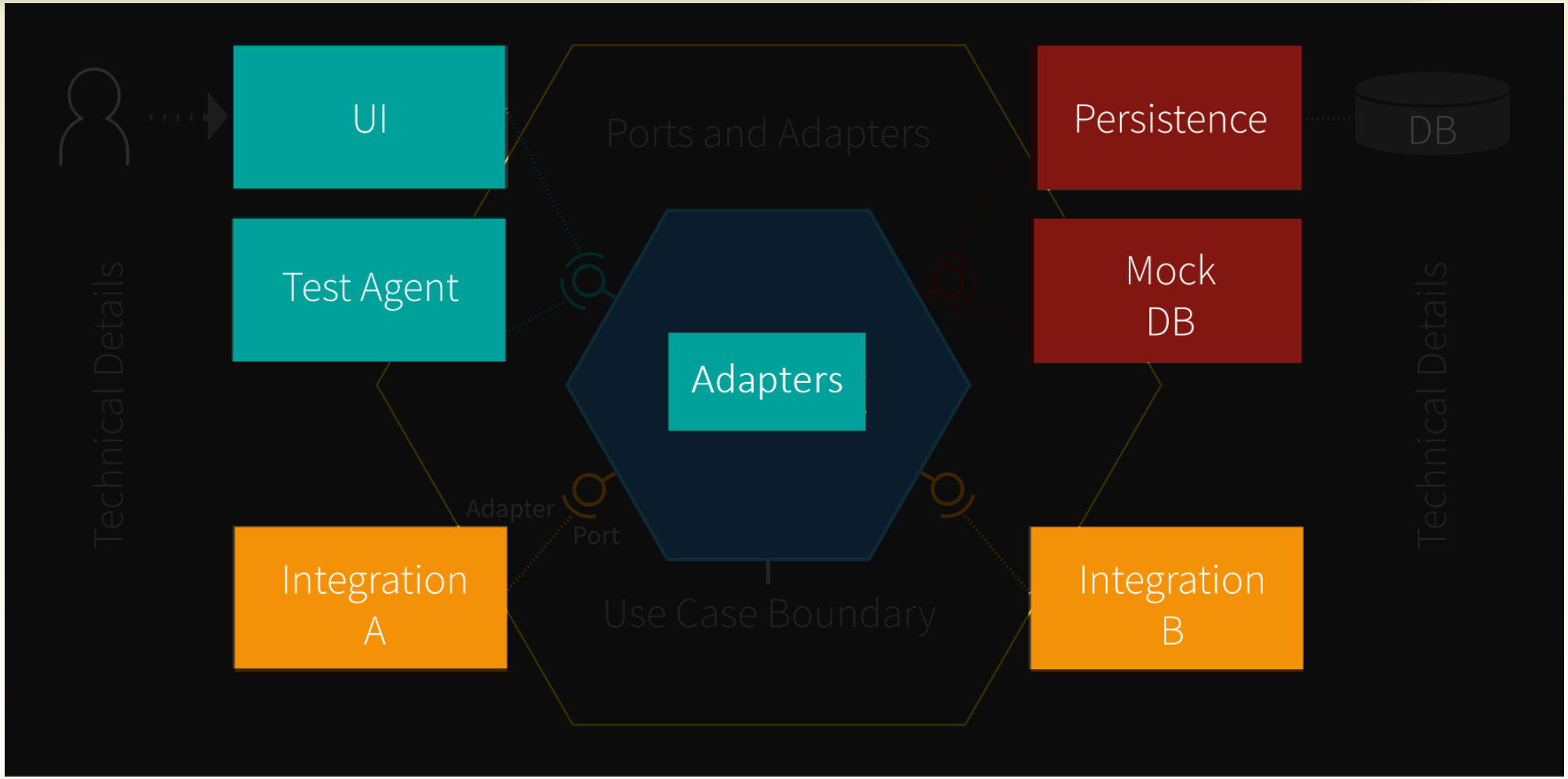
O que é?

ID #32



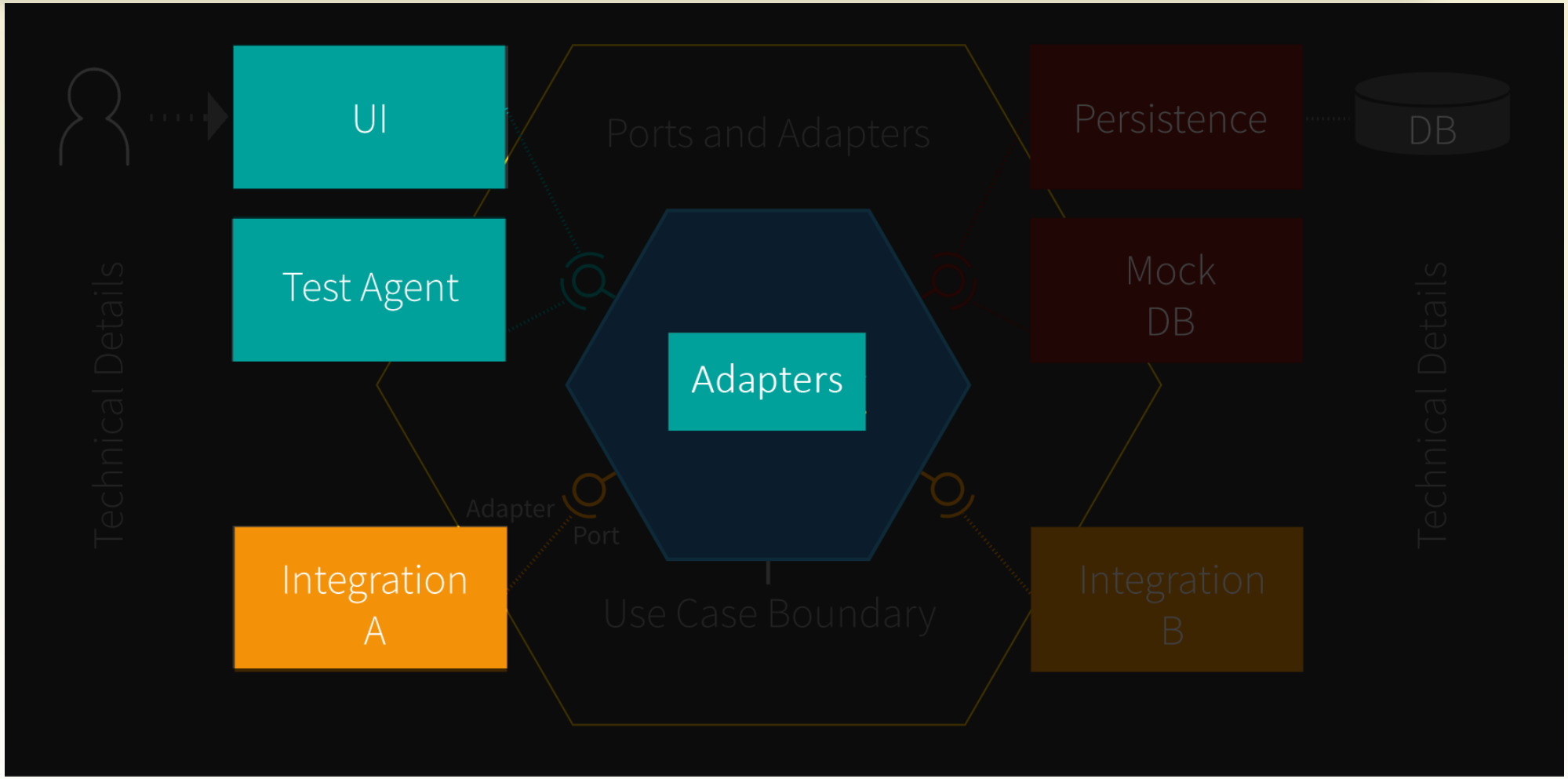
O que é?

ID #32



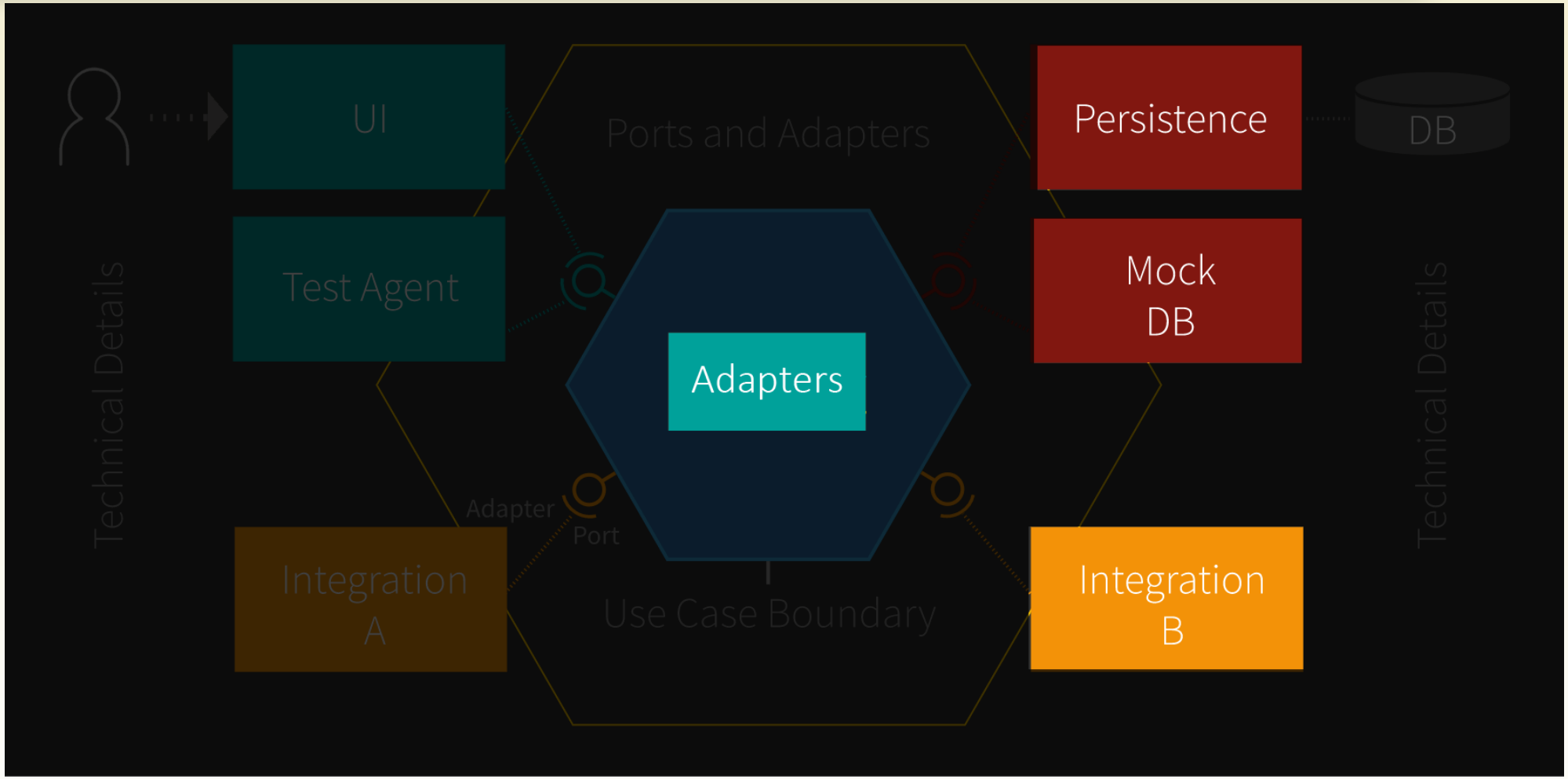
O que é?

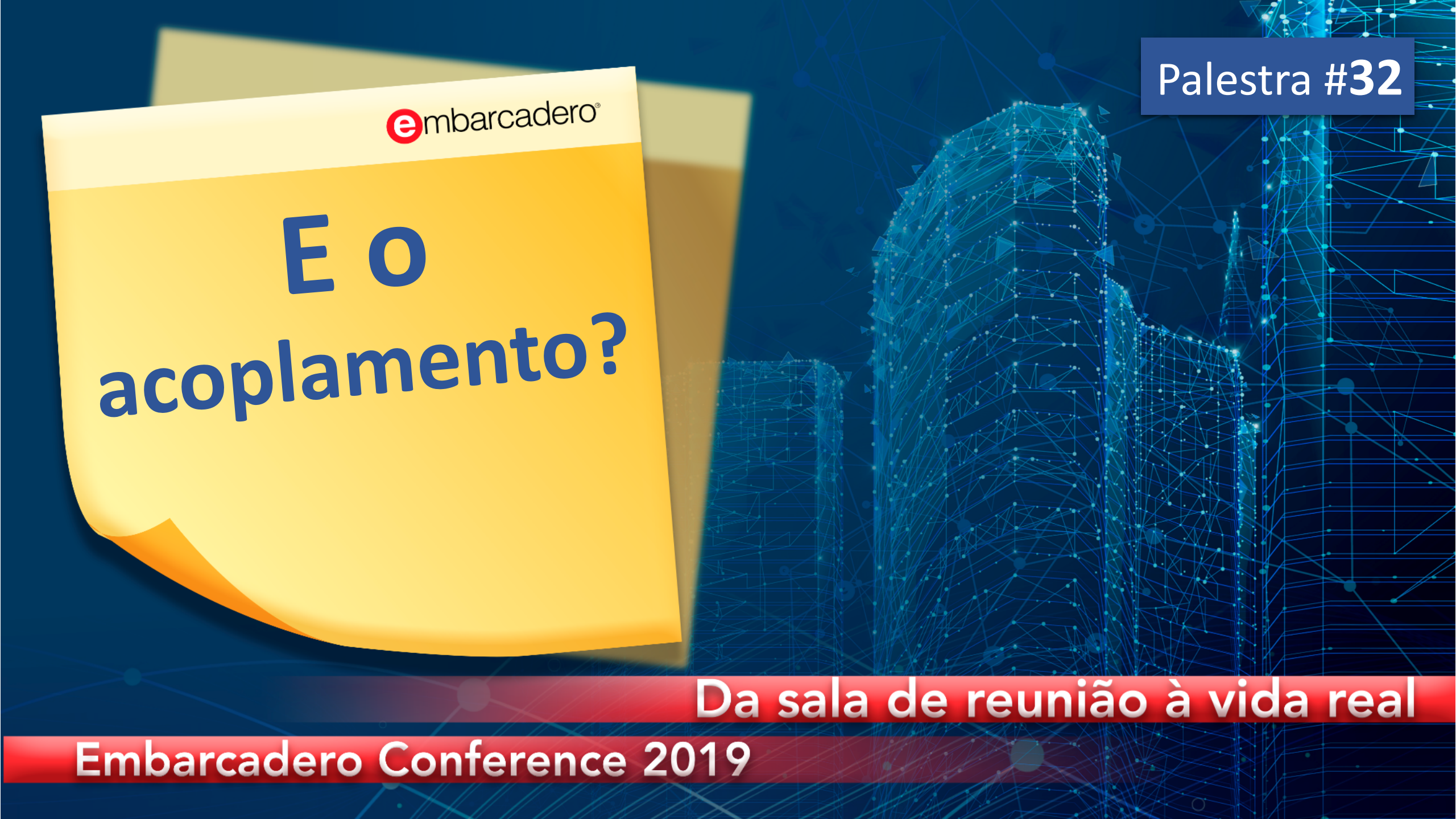
ID #32



O que é?

ID #32




Palestra #32
embarcadero
E o acoplamento?
Da sala de reunião à vida real
Embarcadero Conference 2019

E o acoplamento?

ID #32

```
procedure TController.ProcessarVenda(const Venda: TVenda);
var
  _vendaService: IVendaServico;
  _vendaEfetuadaComSucesso: boolean;
begin
  _vendaService := TVendaService.Create;
  _vendaEfetuadaComSucesso := _vendaService.ProcessarVenda(Venda);
  if not _vendaEfetuadaComSucesso then
    raise Exception.Create('Deu ruim');
end;
```

E o acoplamento?

ID #32

```
procedure TController.ProcessarVenda(const Venda: TVenda);
var
  _vendaService: IVendaServico;
  _vendaEfetuadaComSucesso: boolean;
begin
  _vendaService := TVendaService.Create;
  _vendaEfetuadaComSucesso := _vendaService.ProcessarVenda(Venda);
  if not _vendaEfetuadaComSucesso then
    raise Exception.Create('Deu ruim');
end;
```

E o acoplamento?

ID #32

```
function TVendaService.ProcessarVenda(const Venda: TVenda): boolean;
var
  _vendaRepositorio: IVendasRepositorio;
begin
  if not Venda.ehValida then
    Exit(false);

  _vendaRepositorio := TVendaRepositorioInterbase.Create;
  Result := _vendaRepositorio.AdicionarVenda(Venda);
end;
```

E o acoplamento?

ID #32

```
function TVendaService.ProcessarVenda(const Venda: TVenda): boolean;
var
  _vendaRepositorio: IVendasRepositorio;
begin
  if not Venda.ehValida then
    Exit(false);

  _vendaRepositorio := TVendaRepositorioInterbase.Create;
  Result := _vendaRepositorio.AdicionarVenda(Venda);
end;
```

E o acoplamento?

ID #32

Como resolver as interfaces sem acoplá-las?

E o acoplamento?

ID #32

- Observer/Event bus
  - Mensageria
  - Event Handlers no monolito?
  - Camada de aplicação orquestrando chamadas


Palestra #32
embarcadero
É possível no Delphi?
Da sala de reunião à vida real
Embarcadero Conference 2019

É possível no Delphi?

ID #32

- Implementa DIP

Spring4D

```
procedure TInfraDI.Build;
begin
  CDI.RegisterType<TBuildRepositorio>.Implements<IBuildRepositorio>;

procedure TControllerDI.Build;
begin
  CDI.RegisterType<TBuildDomainEvent>.Implements<IBuildEvent>;
```

É possível no Delphi?

ID #32

- Implementa DIP

```
TBuildService = class(TInterfacedObject, IBuildService)
private
  FBuildRepositorio: IBuildRepositorio;
  FBuildEvent: IBuildEvent;
  function TratarURL(const AURL: string): string;
protected
  function PegarRepositorio: IBuildRepositorio;
  procedure LancarEvento(const ABuildModel: IBuildModel);
public
  constructor Create(const BuildRepositorio: IBuildRepositorio;
    const BuildEvent: IBuildEvent);
```

É possível no Delphi?

ID #32

- Implementa Service Locator


```
begin
  CDI.RegisterType<TBuildService>.Implements<IBuildService>;
end;
```

```
var
  _service: IBuildService;
begin
  _service := CDI.Resolve<IBuildService>;
```

É possível no Delphi?

ID #32

- Como controlar o DIP?
  - Cada camada controla o seu registro de DIP
    - Vantagem: camadas exteriores não precisam conhecer as classes;
    - Desvantagens: contamina o hexágono com a dependência de um framework (Spring4D)

É possível no Delphi?

ID #32

- Como controlar o DIP?
  - Todas as dependências são resolvidas na camada mais externa da aplicação
    - Vantagem: não há contaminação do núcleo;
    - Desvantagens: A camada externa precisa conhecer (e expor) as classes das camadas internas;

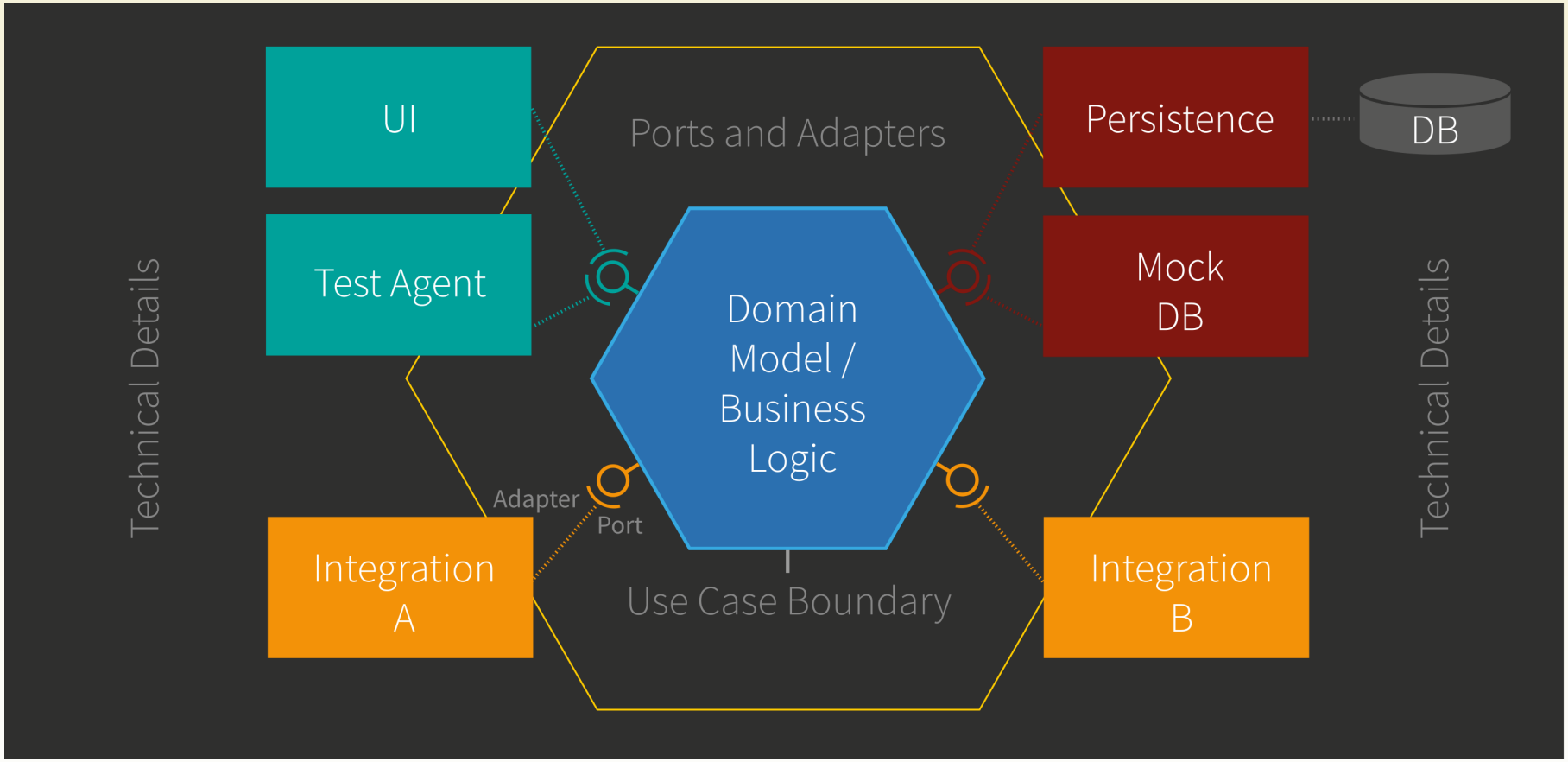
É possível no Delphi?

ID #32

- Packages
  - Modularização da aplicação em
    - Infra: Adaptadores para as *driven ports*
    - App/Drive: Responsável por estabelecer contato com o mundo exterior. Não precisa ser um pacote, podendo ser a própria aplicação

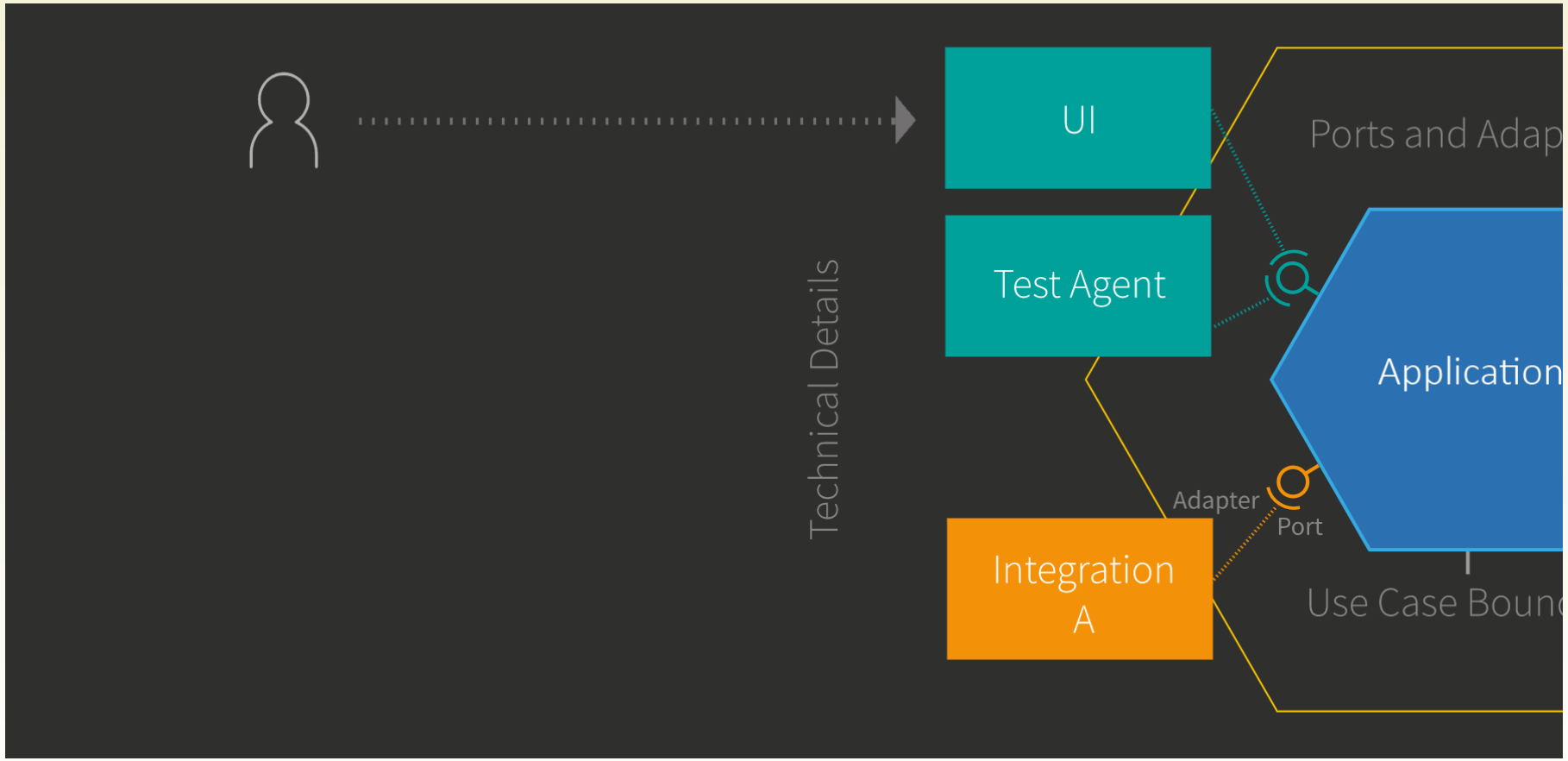
É possível no Delphi?

ID #32



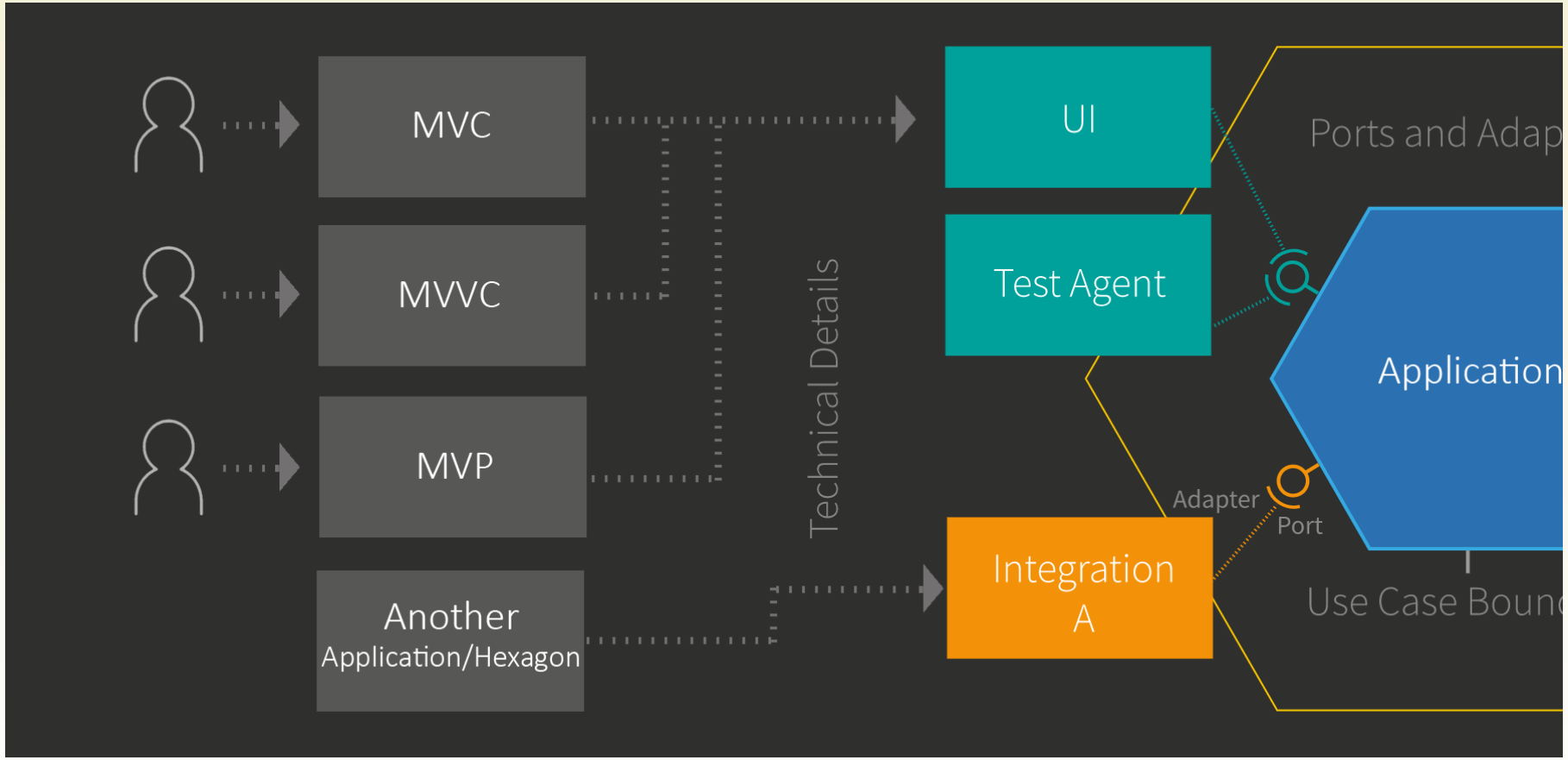
Por que Arq. Flexível?

ID #32



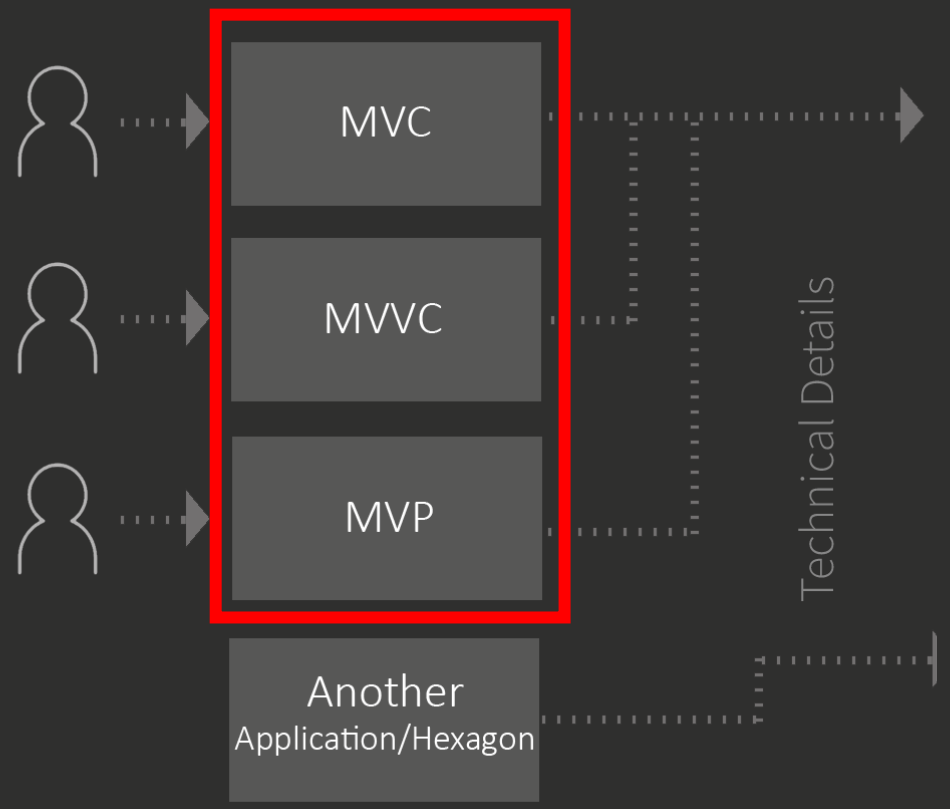
É possível no Delphi?

ID #32



É possível no Delphi?

ID #32



- Framework
- Autenticação
- Autorização
- Detalhes de ambiente

Palestra #32

embarcadero

Projeto Vigilante

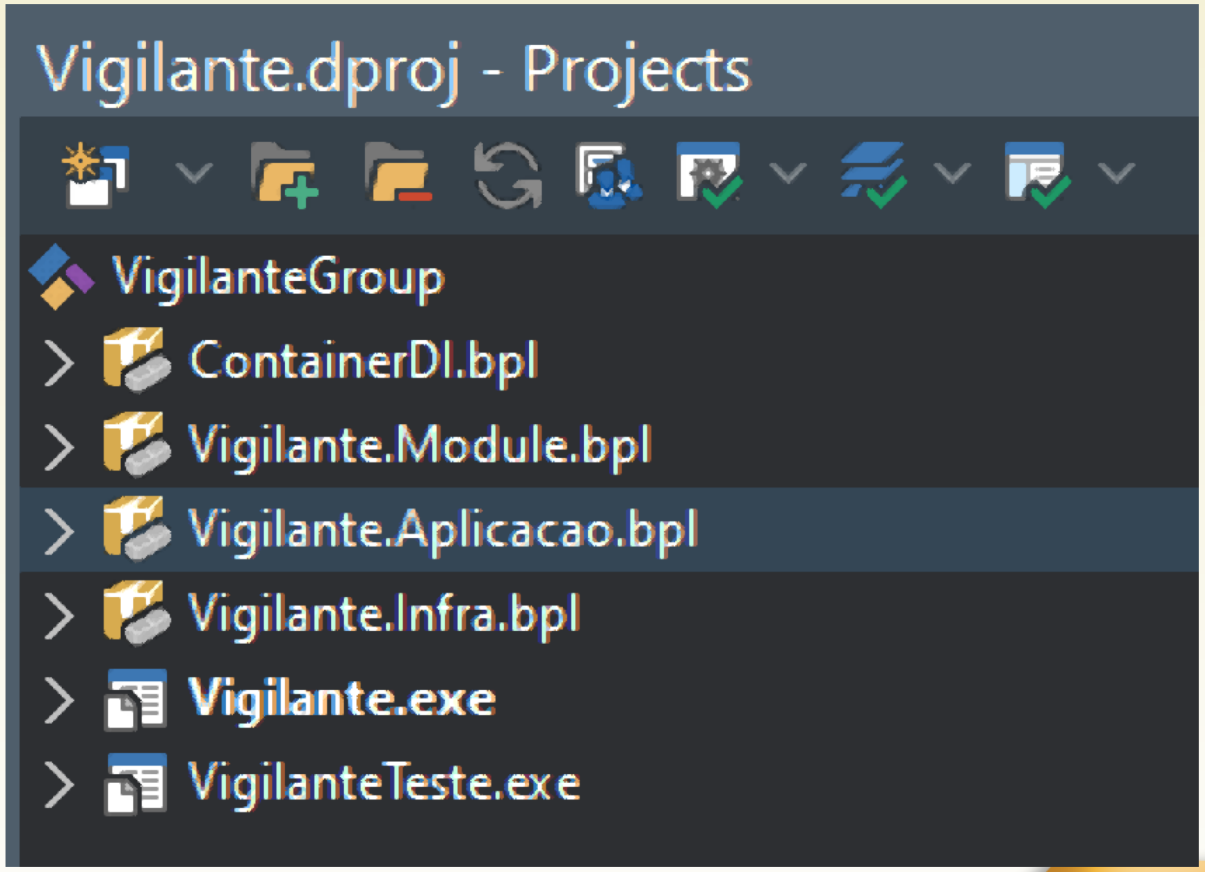
Da sala de reunião à vida real

Embarcadero Conference 2019

Projeto Vigilante

ID #32

- Integra com Jenkins
- Notifica mudanças de estado dos builds/compilações;



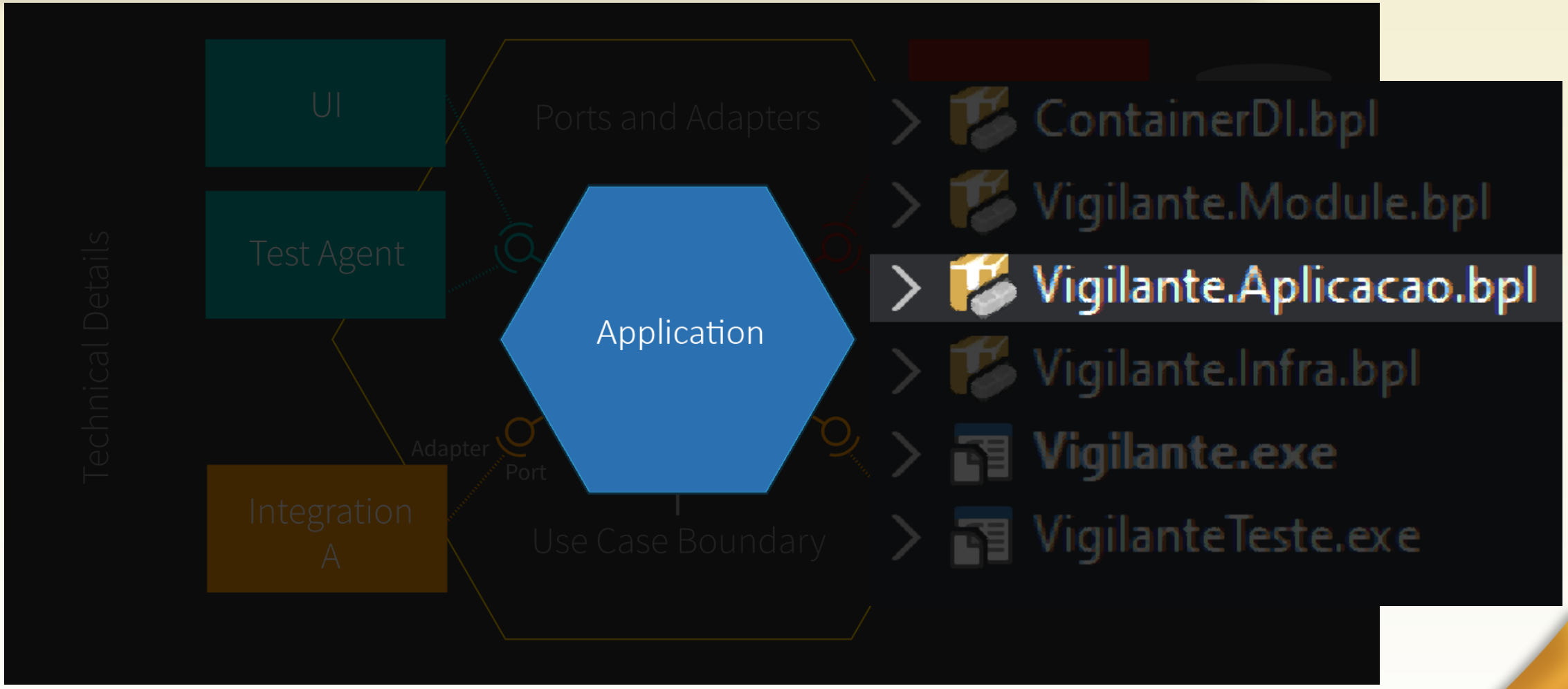
Projeto Vigilante

ID #32

embarcadero

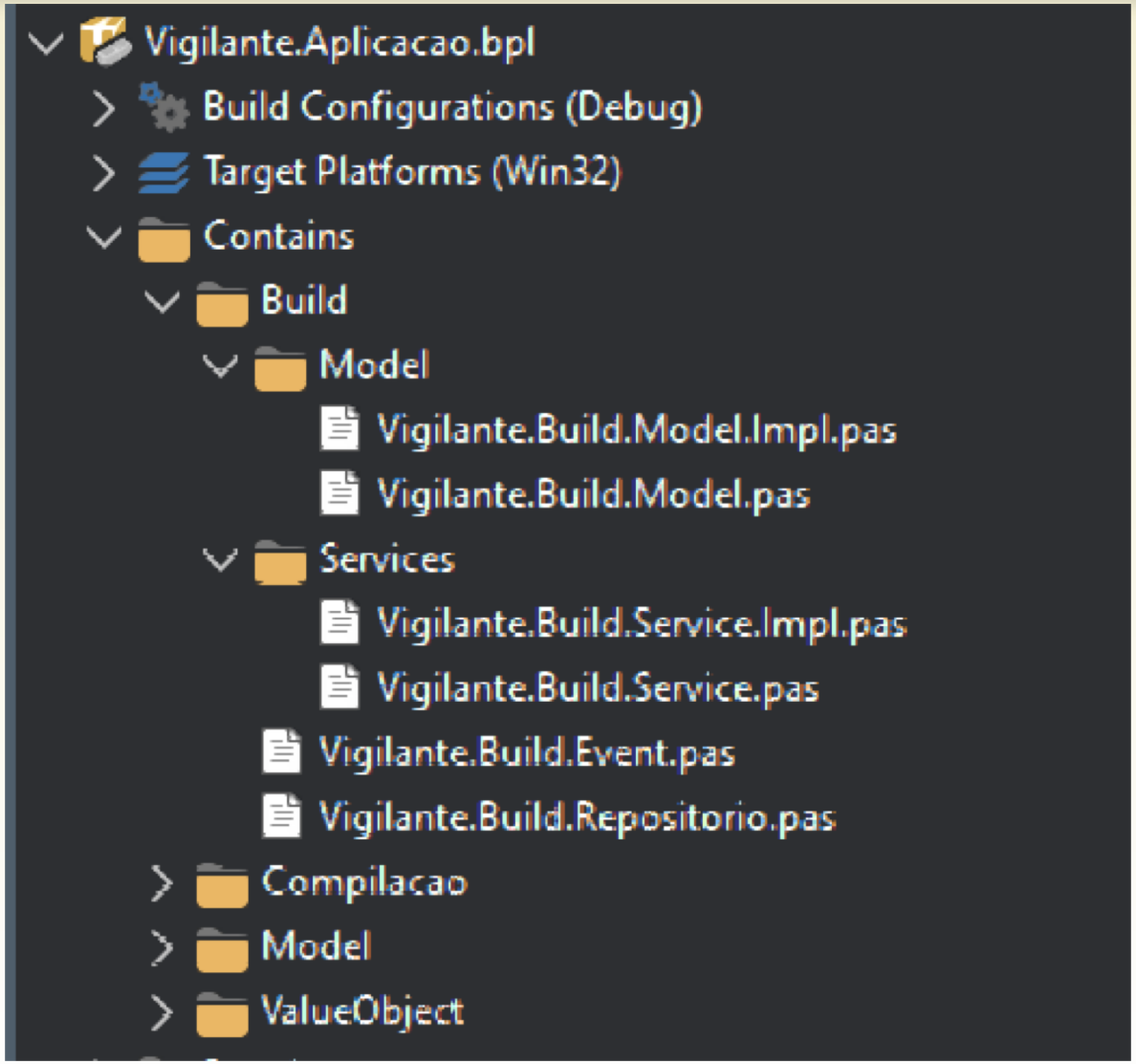
Projeto Vigilante

ID #32



Projeto Vigilante

ID #32



```
TBuildModel = class(TModelBase, IBuildModel)
private
  FNome: string;
  FURL: string;
  FSituacao: TSituacaoBuild;
  FBuilding: Boolean;
  FBuildAtual: integer;
  FUltimoBuildFalha: integer;
  FUltimoBuildSucesso: integer;
  FURLUltimoBuild: IURL;
protected
  function GetBuilding: Boolean;
  function GetNome: string;
  function GetSituacao: TSituacaoBuild;
  function GetURL: string;
  function GetBuildAtual: integer;
  function GetUltimoBuildFalha: integer;
  function GetUltimoBuildSucesso: integer;
  function GetURLUltimoBuild: IURL;
public
  constructor Create(const ANome: string; const AURL: string;
    const ASituacao: TSituacaoBuild; const ABuilding: Boolean;
    const ABuildAtual, AUltimoBuildFalha, AUltimoBuildSucesso: integer;
    const AURLUltimoBuild: IURL);
  function Equals(const BuildModel: IBuildModel): Boolean; reintroduce;
end;
```

ID #32

Projeto Vigilante

ID #32

```
type
  IBuildService = interface(IInterface)
    ['{938B103A-E58B-438C-A0EB-D2E295E25FAA}']
    function AtualizarBuild(const BuildModel: IBuildModel): IBuildModel;
  end;
```

Projeto Vigilante

ID #32

```
type
  TBuildService = class(TInterfacedObject, IBuildService)
  private
    FBuildRepositorio: IBuildRepositorio;
    FBuildEvent: IBuildEvent;
    function TratarURL(const AURL: string): string;
  protected
    procedure LancarEvento(const ABuildModel: IBuildModel);
  public
    constructor Create(const BuildRepositorio: IBuildRepositorio;
      const BuildEvent: IBuildEvent);
    function AtualizarBuild(const BuildModel: IBuildModel): IBuildModel;
  end;
```

ID #32

```
function TBuildService.AtualizarBuild(const BuildModel: IBuildModel)
  : IBuildModel;
var
  _url: string;
  _novoBuild: IBuildModel;
begin
  Result := nil;
  _url := BuildModel.URL;
  _url := TratarURL(_url);
  _novoBuild := FBuildRepositorio.BuscarBuild(_url);

  if not Assigned(_novoBuild) then
    Exit;

  _novoBuild.DefinirID(BuildModel.Id);

  if not _novoBuild.Equals(BuildModel) then
    LancarEvento(_novoBuild);

  Result := _novoBuild;
end;
```

Projeto Vigilante

ID #32

```
type
  IBuildRepositorio = interface(IInterface)
    ['{63C86F1E-E246-465F-A67C-2E6D8792E407}']
    function BuscarBuild(const AURL: string): IBuildModel;
  end;
```

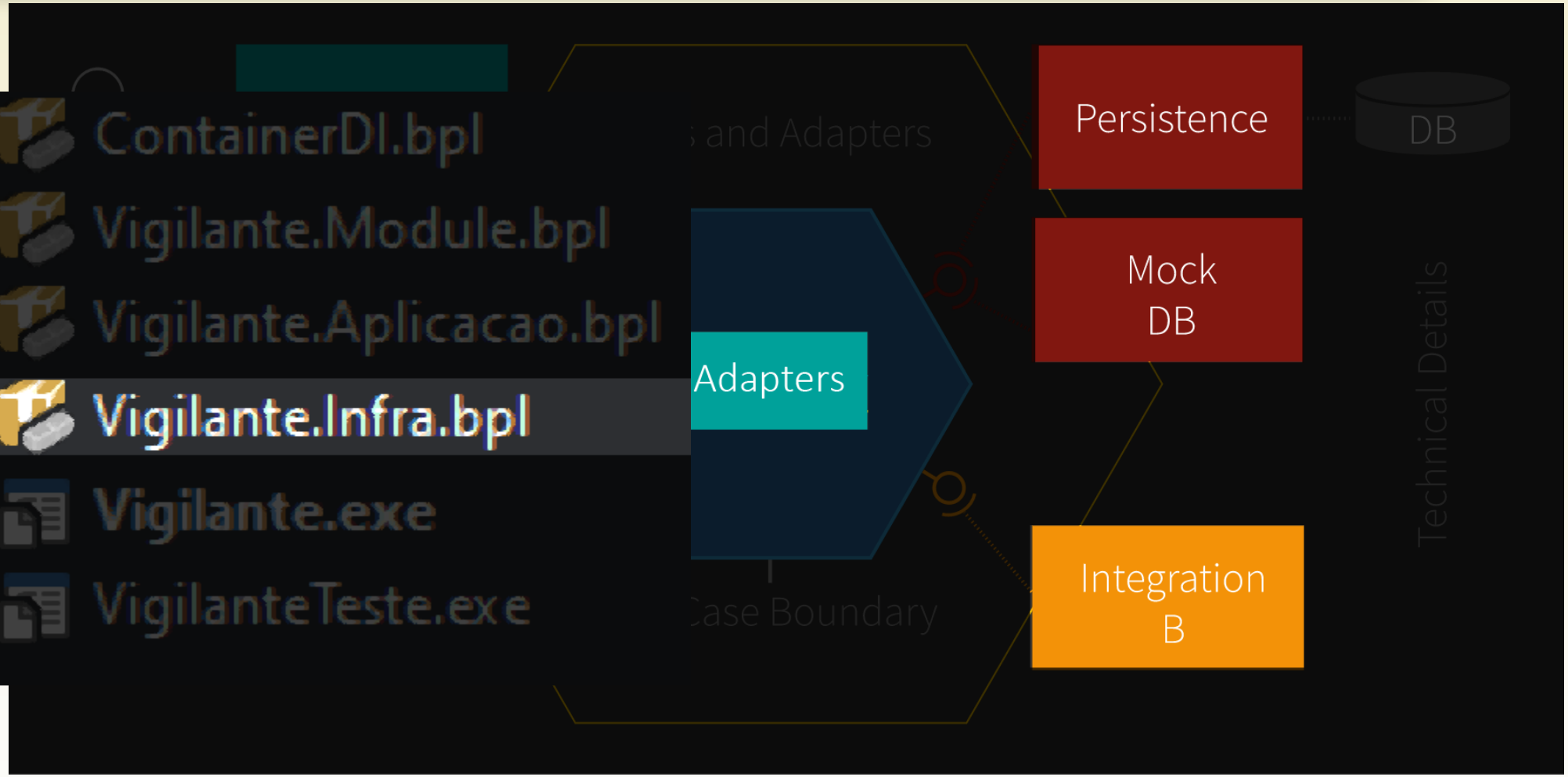
Projeto Vigilante

ID #32

```
type
  IBuildEvent = interface(IInterface)
    ['{51417C97-A33E-4EF8-A6B7-B98B0748B258}']
    procedure Notificar(const ABuild: IBuildModel);
  end;
```

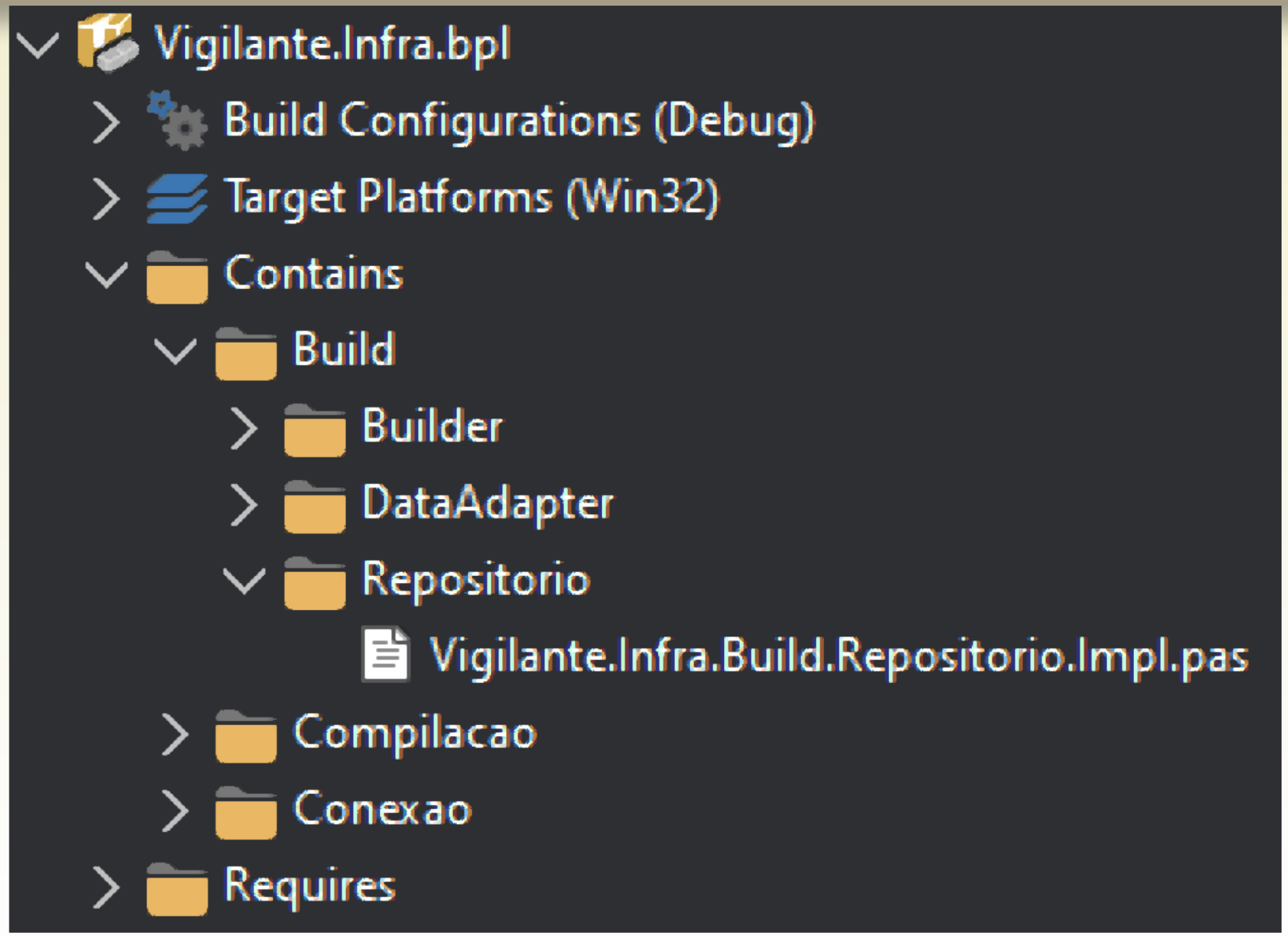
Projeto Vigilante

ID #32



Projeto Vigilante

ID #32



Projeto Vigilante

ID #32

```
type
  TBuildRepositorio = class(TInterfacedObject, IBuildRepositorio)
  private
    function BuscarJSON(const AURL: string): TJSONObject;
    function TratarURL(const AURL: string): string;
  public
    function BuscarBuild(const AURL: string): IBuildModel;
  end;
```

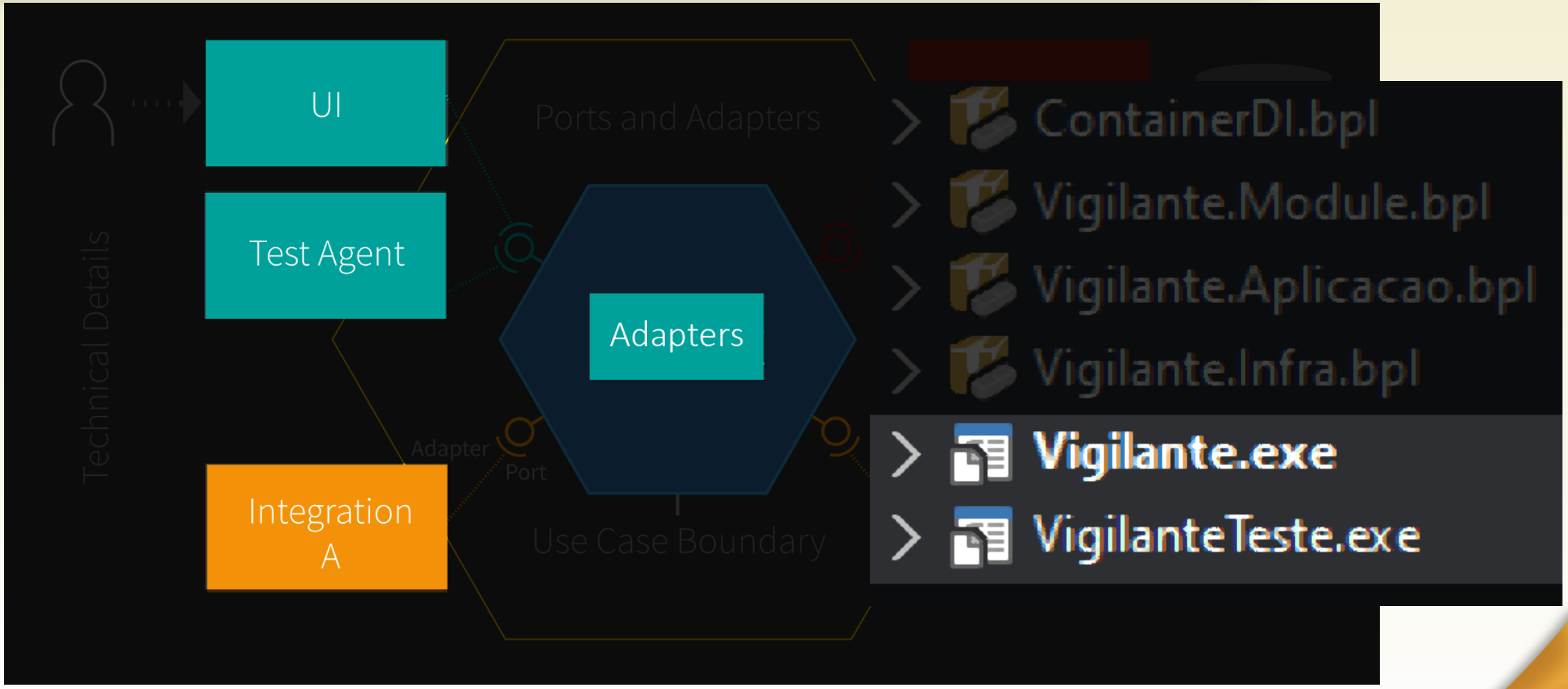
Projeto Vigilante

ID #32

```
function TBuildRepositorio.BuscarBuild(const AURL: string): IBuildModel;
var
  _json: TJSONObject;
  _buildBuilder: IBuildBuilder;
begin
  Result := nil;
  _json := BuscarJSON(AURL);
  if not Assigned(_json) then
    Exit;
  try
    _buildBuilder := CDI.Resolve<IBuildBuilder>([_json]);
    Result := _buildBuilder.PegarBuild;
  finally
    FreeAndNil(_json);
  end;
end;
```

Projeto Vigilante

ID #32



Projeto Vigilante

ID #32

```
type
  TBuildController = class(TControllerBase<IBuildModel>, IBuildController)
  private
    FDataSet: TBuildDataSet;
    FMediator: IControllerMediator;
    procedure SalvarDadosBuild;
    procedure CarregarDadosDoBuild;
  protected
    function GetDataSet: TBuildDataSet;
    procedure SetDataSet(const ADataSet: TBuildDataSet);
    function GetDataSetInterno: TVigilanteDataSetBase<IBuildModel>; override;
  public
    constructor Create(const AMediator: IControllerMediator);
    destructor Destroy; override;
    procedure AdicionarOuAtualizar(const ABuild: IBuildModel);
    procedure BuscarAtualizacoes; override;
    procedure TransformarEmCompilacao(const AID: TGUID);
    property DataSet: TBuildDataSet read GetDataSet;
  end;
```

```
procedure TBuildController.BuscarAtualizacoes;
var
  _service: IBuildService;
  _dataTemp: TBuildDataSet;
  _buildOriginal: IBuildModel;
  _build: IBuildModel;
begin
  inherited;
  if DataSet.IsEmpty then
    Exit;
  if DataSet.State in dsEditModes then
    Exit;
  _dataTemp := TBuildDataSet.Create(nil);
  try
    _dataTemp.CloneCursor(DataSet);
    _dataTemp.ApenasAtualizaveis();
    _dataTemp.First;
    while not _dataTemp.Eof do
    begin
      _service := CDI.Resolve<IBuil      ice>;
      _buildOriginal := _dataTemp.E      tarRegistro;
      _build := _service.AtualizarBuild(_buildOriginal);
      if not Assigned(_build) then
        Exit;
      if not _buildOriginal.Equals(_build) then
        _dataTemp.Importar(_build);
      _dataTemp.Next;
    end;
  finally
    FreeAndNil(_dataTemp);
```

ID #32

embarcadero

Projeto Vigilante

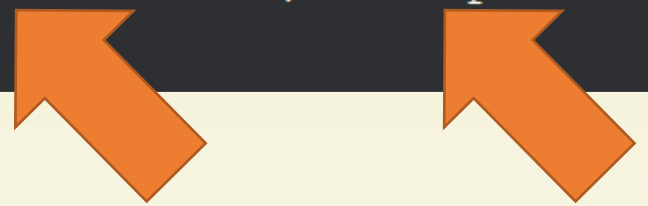
ID #32

```
type
  TBuildDomainEvent = class(TInterfacedObject, IBuildEvent)
  public
    procedure Notificar(const ABuild: IBuildModel);
  end;

implementation

uses
  ContainerDI, Vigilante.Build.Observer;

procedure TBuildDomainEvent.Notificar(const ABuild: IBuildModel);
var
  _observer: IBuildSubject;
begin
  _observer := CDI.Resolve<IBuildSubject>;
  _observer.Notificar(ABuild);
end;
```

Projeto Vigilante

ID #32

```
TfrmPrincipal = class(TFormBase, IBuildObserver, ICompilacaoObserver,
  IConfiguracaoObserver)
```



```
  procedure MostrarNotificacaoBuild(const ABuild: IBuildModel);
  procedure MostrarNotificacaoCompilacao(const ACompilacao: ICompilacaoModel);
public
  procedure IBuildObserver.NovaAtualizacao = MostrarNotificacaoBuild;
  procedure ICompilacaoObserver.NovaAtualizacao =
    MostrarNotificacaoCompilacao;
```

Projeto Vigilante

ID #32

```
procedure TfrmPrincipal.MostrarNotificacaoBuild(const ABuild: IBuildModel);
var
  _notificacao: TNotification;
begin
  if not(ABuild.Situacao in SITUACOES_NOTIFICAVEIS) then
    Exit;
  _notificacao := TNotification.Create;
  try
    _notificacao.ChannelId := NOTIFICATION_BUILD;
    _notificacao.Name := ABuild.Id.ToString;
    _notificacao.Number := ABuild.BuildAtual;
    _notificacao.Title := ABuild.Nome;
    _notificacao.AlertBody := ABuild.Situacao.AsString;
    NotificationCenter.PresentNotification(_notificacao);
  finally
    FreeAndNil(_notificacao);
  end;
end;
```

embarcadero

Projeto Vigilante

ID #32

Projeto Vigilante

ID #32

Perguntas?


DB1
global software
embarcadero
Obrigado
db1group.com/carreira/
ftathiago@gmail.com
#32
github.com/ftathiago
@fta_thiago
/ftathiago
Da sala de reunião à vida real
Embarcadero Conference 2019