Modularização via BPL

Abordagem prática para DataSnap & Front-end

Mário Guedes

Embarcadero Conference

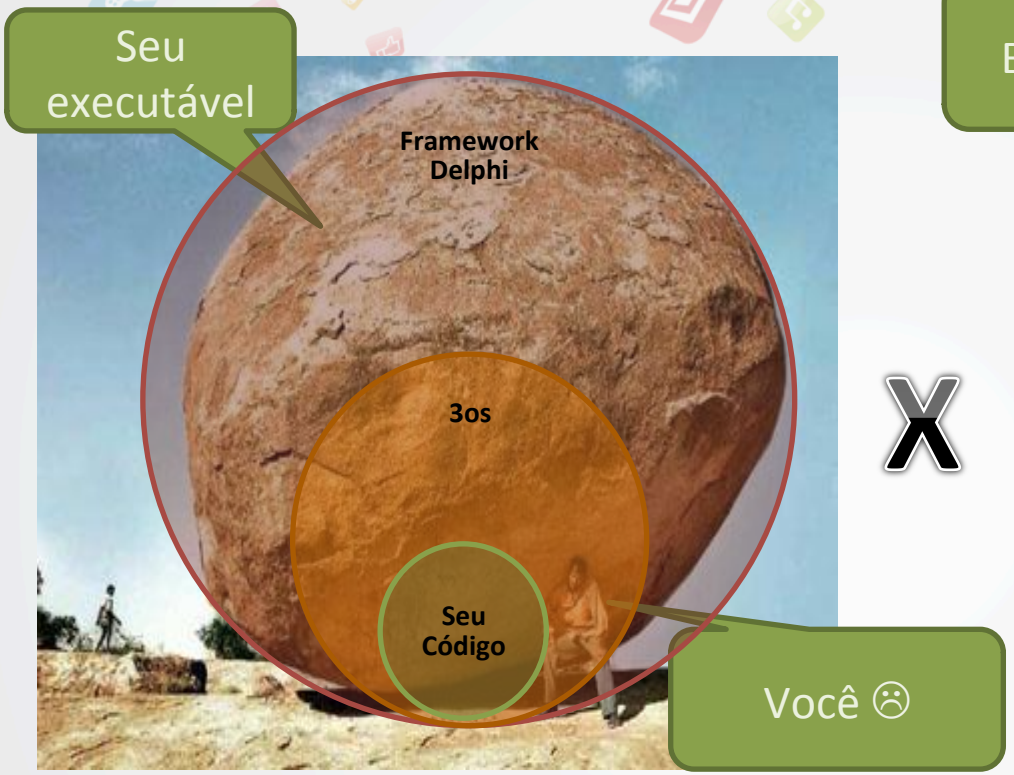
embarcadero

Do que estamos falando?

- O resultado final do nosso trabalho é um grande executável: **“.exe**”
- Mas podemos quebrá-lo em partes menores: **MÓDULOS**
- A **BPL** - *Borland Package Library*, nos dá esta possibilidade.

O Delphi é um ótimo exemplo de aplicação modular.

Qual a diferença?

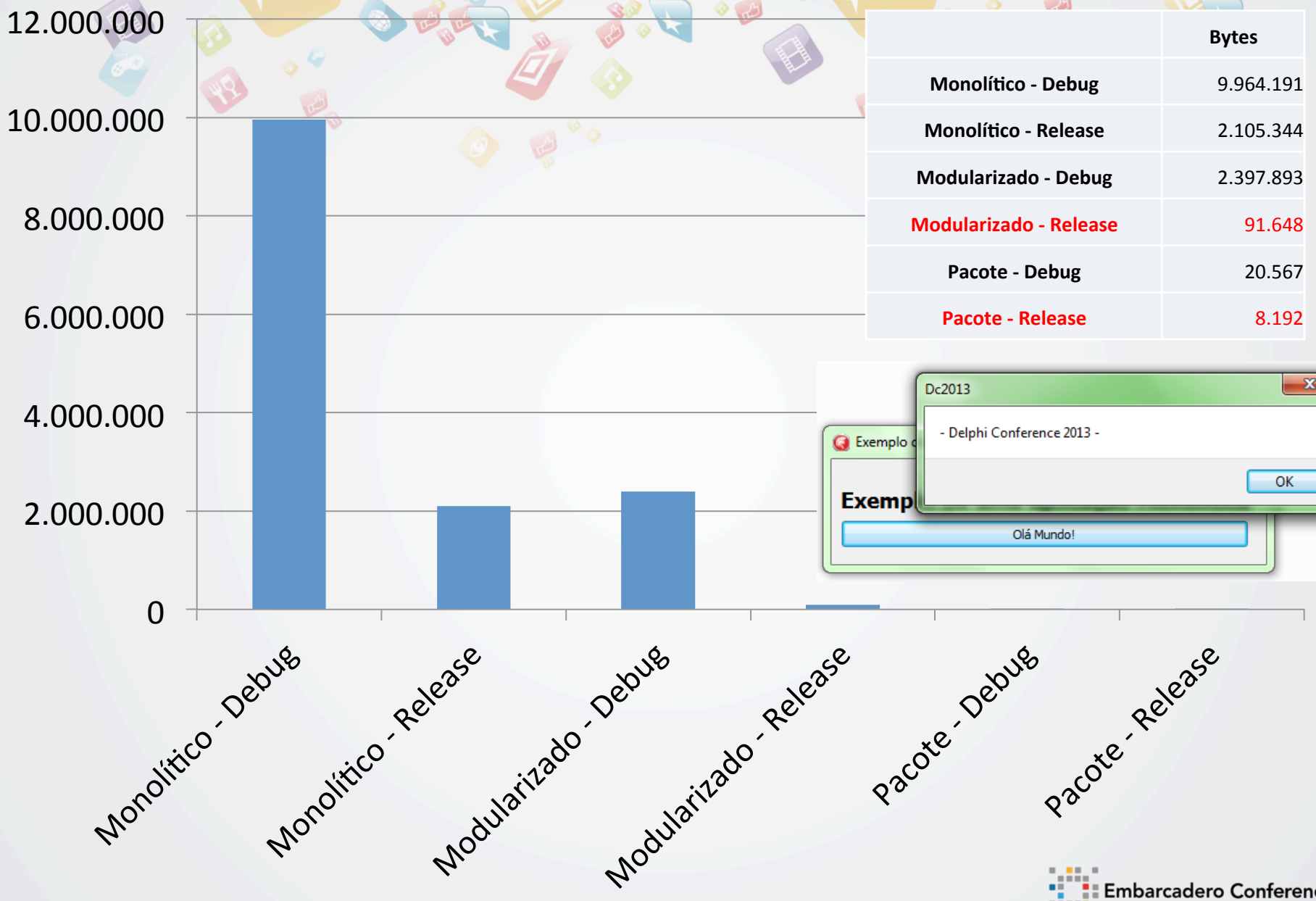


X

Um grande projeto com dezenas de dependências

Vários pequenos projetos com poucas dependências

Um simples “Olá Mundo”



	Bytes
Monolítico - Debug	9.964.191
Monolítico - Release	2.105.344
Modularizado - Debug	2.397.893
Modularizado - Release	91.648
Pacote - Debug	20.567
Pacote - Release	8.192

Embarcadero Conference

Motivação

- Melhor divisão de responsabilidades entre a equipe.
- Melhor aderência de um novato por não precisar lidar com algo complexo imediatamente.
- Menor escopo favorece maior domínio.
- Permite que várias pessoas trabalhem em uma mesma solução.
- Permite a adoção de novas tecnologias **sem refazer** o sistema: Exemplo: BDE → dbExpress → FireDAC → ?
- Aderência às metodologias ágeis.

E mais vantagens!

- Facilita a *personalização* da solução para os clientes finais (plug-ins).
- Diminuição do tamanho dos artefatos (exe, dll e bpl).
- Facilita a atualização do aplicativo, ficando quase tão instantâneo quanto um aplicativo web.
- Possibilidade de armazenar formulários, imagens, resources strings para internacionalização, procedimentos, enfim: ***uma biblioteca*!**
- Interação com tipos do Delphi: *classes, records, enumerados, string e etc*.

Quem ganha com a modularidade?

- ✓Você
  - Passa a focar no real problema a ser resolvido e não na tecnologia em volta. E volta mais cedo para casa.
- ✓Sua equipe
  - Ninguém fica sobrecarregado. Ninguém fica ocioso. Temos efetivamente uma *equipe*.
- ✓Sua empresa
  - Custa caro manter um software “vivo”.
- ✓Seu cliente
  - Obtêm respostas rápidas às demandas. Com qualidade e flexibilidade.
- ✓Seu consultor
  - e-mail no penúltimo slide ☺

Ressalvas

- Um sistema desenvolvido em Delphi XE5 só carregará BPLs compiladas em Delphi XE5. Conforme o Delphi avança cria-se incompatibilidades de mapeamentos de tipos com o Delphi anterior.
- Migração de Delphi exige um trabalho extra para as compatibilizações.
- A carga da aplicação fica ligeiramente mais lenta, em especial se os módulos estiverem “longe”. Em geral é um problema irrelevante.
- Duas units com o mesmo nome não poderão ser carregadas pela mesma aplicação. Utilize **namespace** para evitar ambiguidades.
- Na primeira instalação será descarregado muito mais conteúdo do que se espera (BPLs do Delphi e de componentes de terceiros).

Redistribuição

- Um problema inicial é a redistribuição das BPLs das quais o seu sistema depende.
- Cuidado com componentes de terceiros mal projetados: o correto é ter uma BPL de "*design time*" e outra de "*run time*", esta última será redistribuída.
- Sugere-se **não** colocá-las no system32, mas sim criar um diretório específico e colocar este diretório na variável de ambiente PATH.
- Os arquivos do Delphi permitidos para redistribuição estão em: \Embarcadero\RAD Studio\12.0\Redist
- Para gerar uma lista das dependências estáticas (BPLs e DLLs) sugere-se o aplicativo **Dependency Walker**:

  http://www.dependencywalker.com/

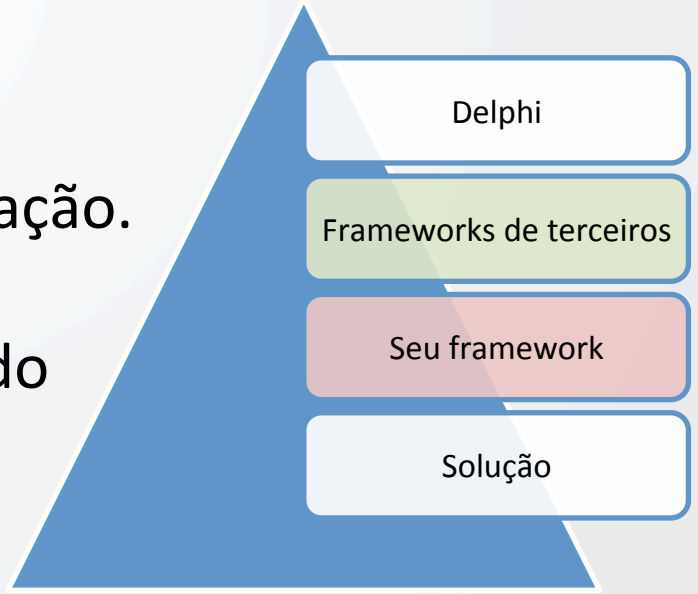
Ortogonalidade

Ortogonalidade, em sistemas, refere-se ao nível de independência que um módulo tem em relação a outro módulo.

Quanto mais independente mais ortogonal.

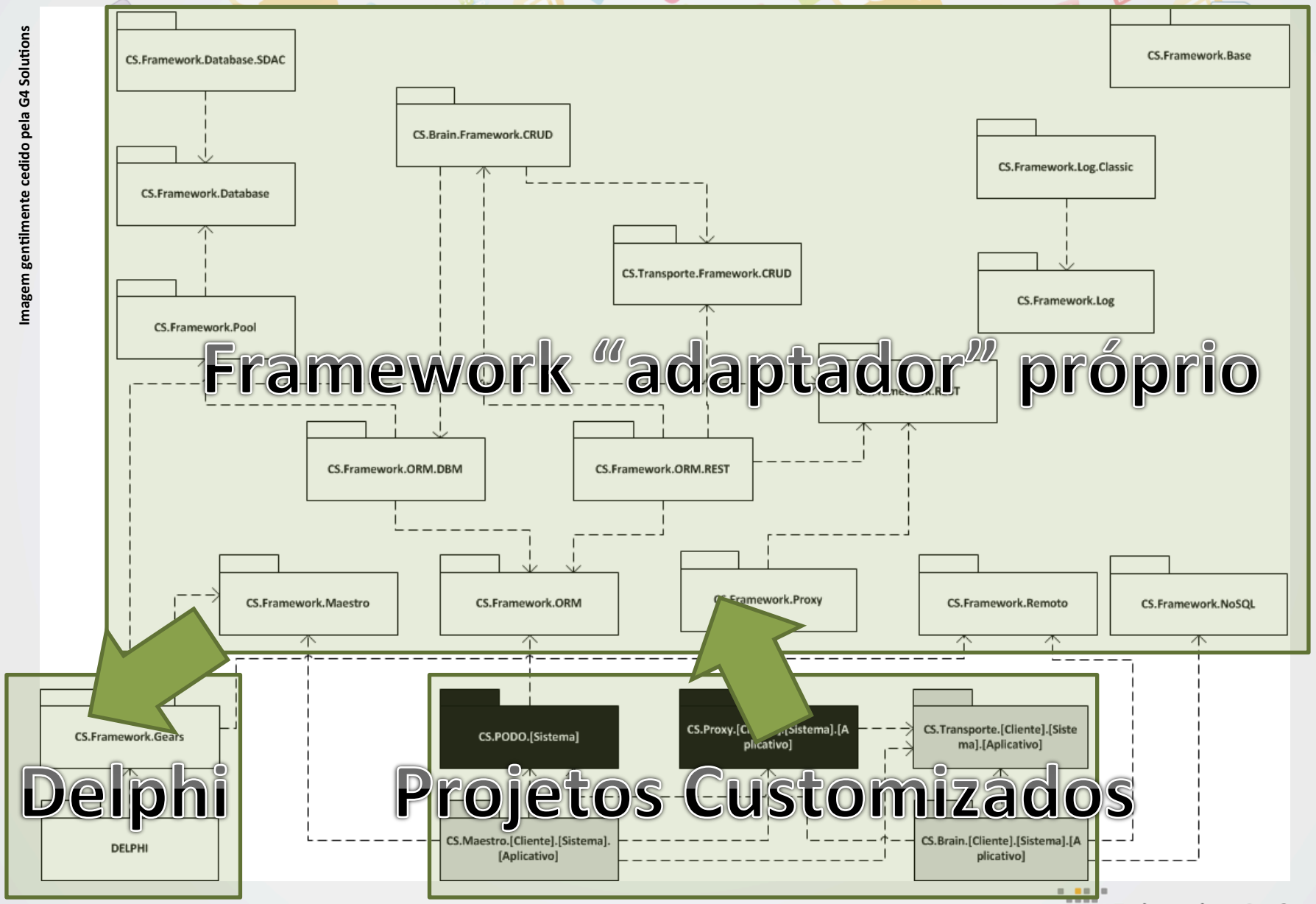
Exemplo: Devemos conseguir mudar de gerenciador de banco de dados sem afetar as telas do front-end e vice versa.

- Delphi *forever.*
- Utilize frameworks de boa reputação.
- Crie seu próprio framework.
- Crie as soluções finais em cima do seu framework.



Um exemplo real

Imagem gentilmente cedido pela G4 Solutions



Conceitos que irão ajudar

- OOP
- Padrões de Projetos
- Interfaces
- RTTI e Generics
- ORM
- DDD
- Geradores automáticos de código
- OTA
- Integração contínua

OOP e Padrões de Projeto

- Um maior domínio sobre a **OOP** é importante para termos um real proveito de todo o poder que o Delphi oferece.
- Um tema que deve ser continuamente estudado é **Padrões de Projeto**.
- Alguns padrões que são aderentes:
  - **Singleton**: *Instância única de uma classe.*
  - **Factory:** *Classes abstratas e classes concretas.*
  - **Façade:** *Fachada de facilidades.*
- A utilização de **Interfaces** é uma forma de abstrair as classes concretas.
- A RTTI dá mais poder à abstração.

Aplique o MVC

- ***Model View Control*** é um padrão de projeto que propõe separar a **regra de negócio** da **regra de visualização**.

- É de extrema importância para dar vida longa ao seu aplicativo, pois o mundo é cada vez mais "poliglota": desktop, web, mobile, SQL, noSQL etc, etc e etc.

- **Isole as regras de negócio**:

  - A interface muda constantemente e coexistem por um longo tempo:

    WinXP, Aero, Ribbon, Metropolis UI, Web, Mobile e etc.

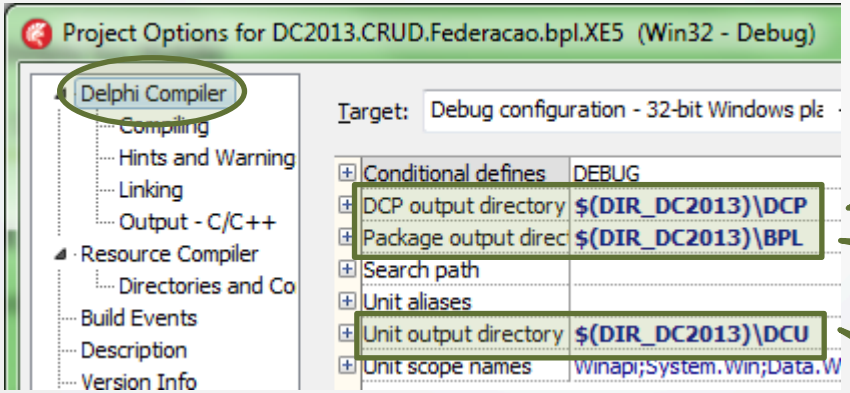
  - As regras de negócio são perenes.

Arquivos DPK, DCP & BPL

- Um projeto de pacote Delphi é encabeçado por um arquivo **DPK** – *Delphi Package* - e não por um arquivo DPR.
- Um **DCP** - *Delphi Compiled Package* - é um mapeamento de uma BPL e só tem serventia para a IDE do Delphi.
- Não é o diretório da BPL que deve ir para a Library Path do Delphi, e sim o diretório do DCP.
- Separe as suas BPLs das do Delphi e componentes de terceiros.
- Sugere-se criar um diretório específico para as saídas do DCP e outro específico para a BPL.
- O diretório de saída das BPLs podem ir para a variável de ambiente PATH.

Configurações de um Projeto DPK

- É muito importante configurar consistentemente as saídas dos artefatos produzidos pela compilação.
- Evite conflitos com os seus colegas usando variáveis de ambiente.

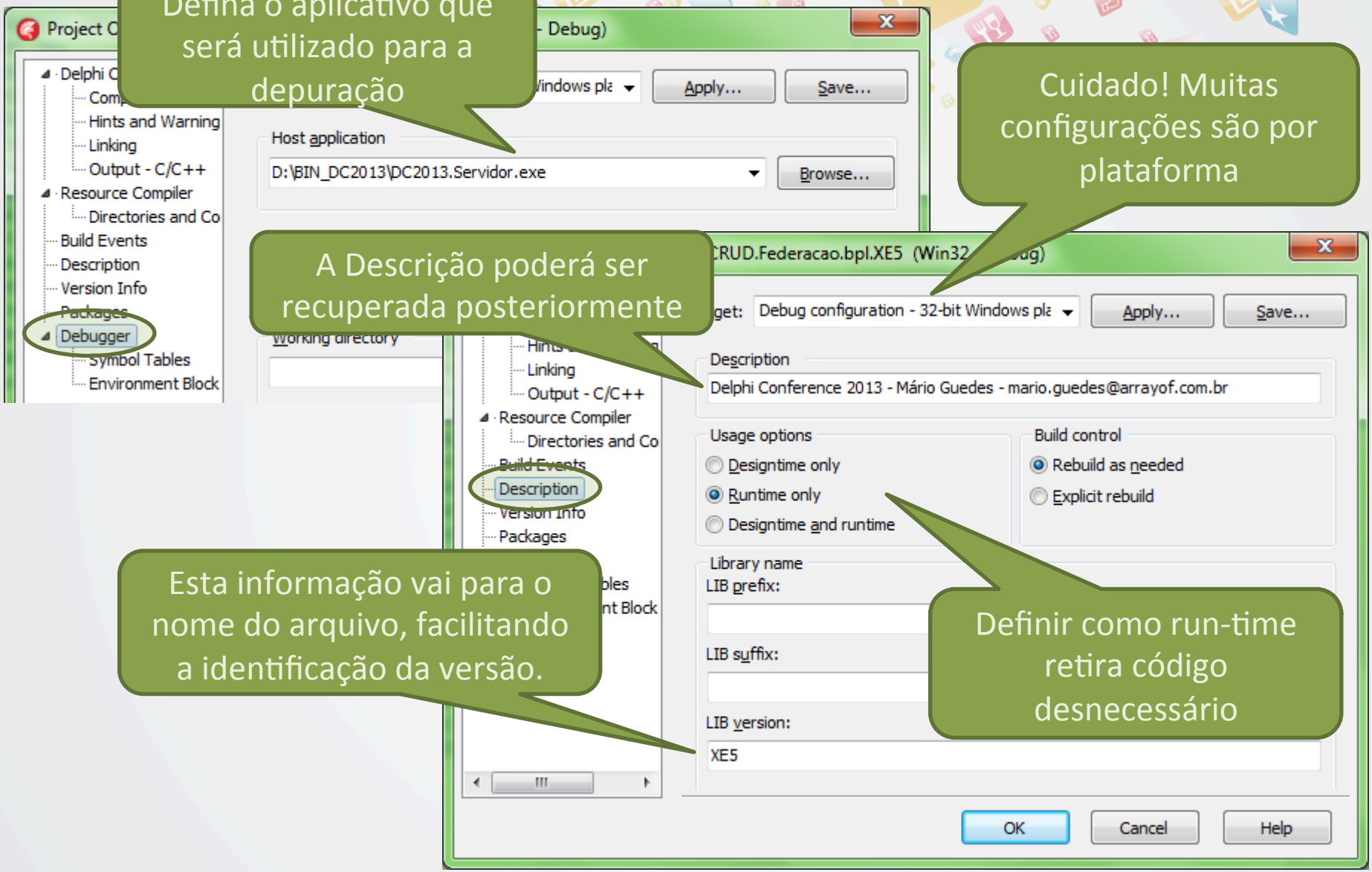


Diretório onde será gerado o DCP

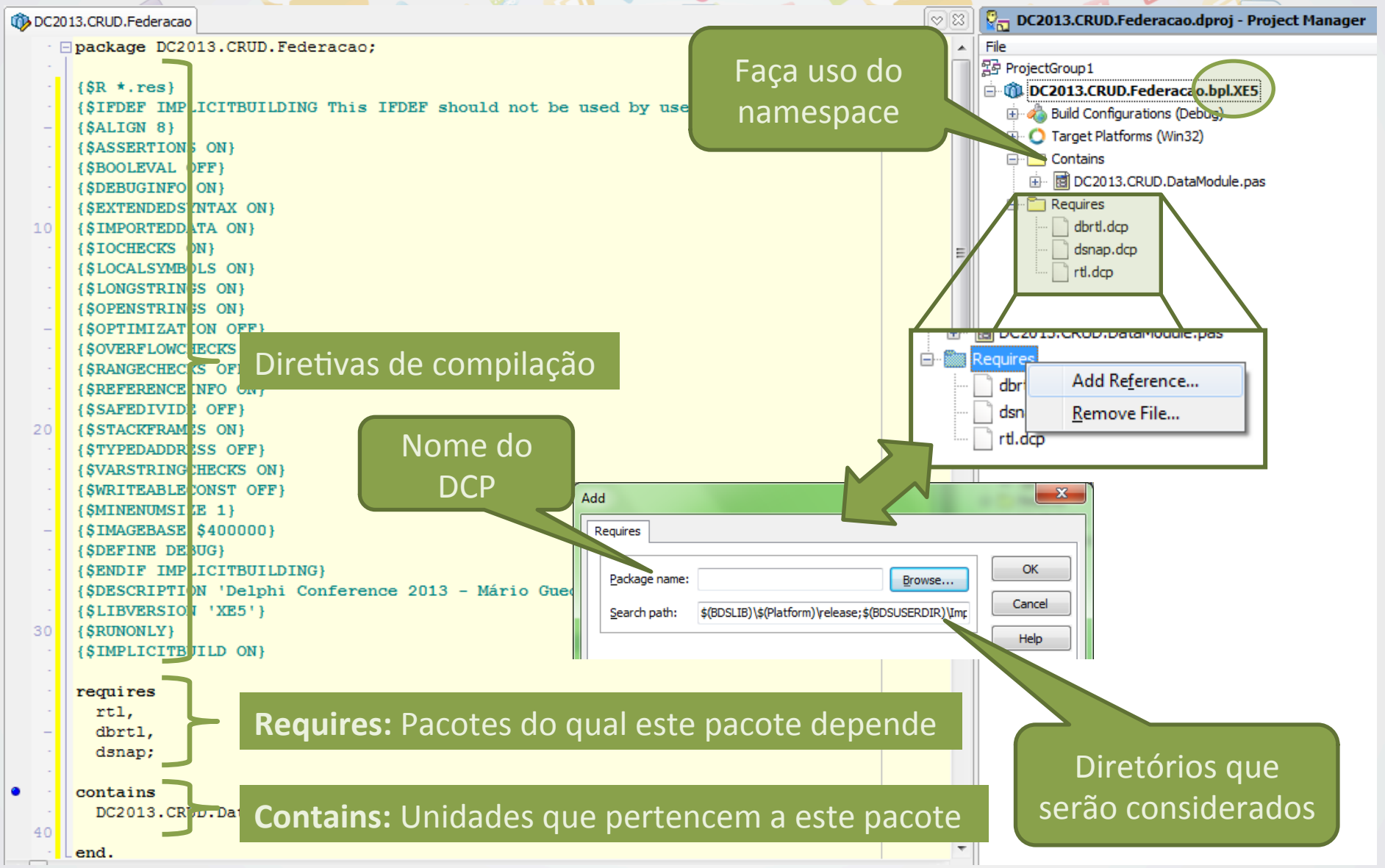
Diretório onde será gerado a BPL

Diretório onde serão gerados os DCUs

Configurações de um Projeto DPK



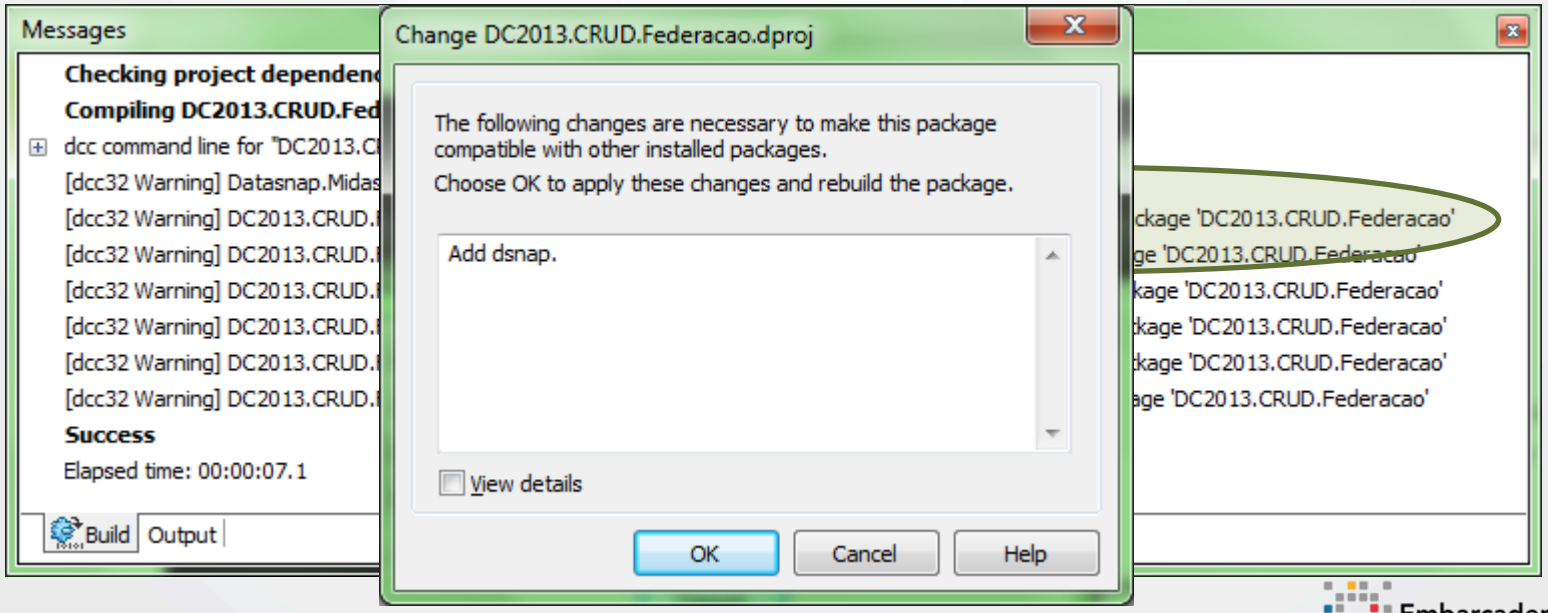
Estrutura e gerenciamento de um DPK



Adicionando dependências

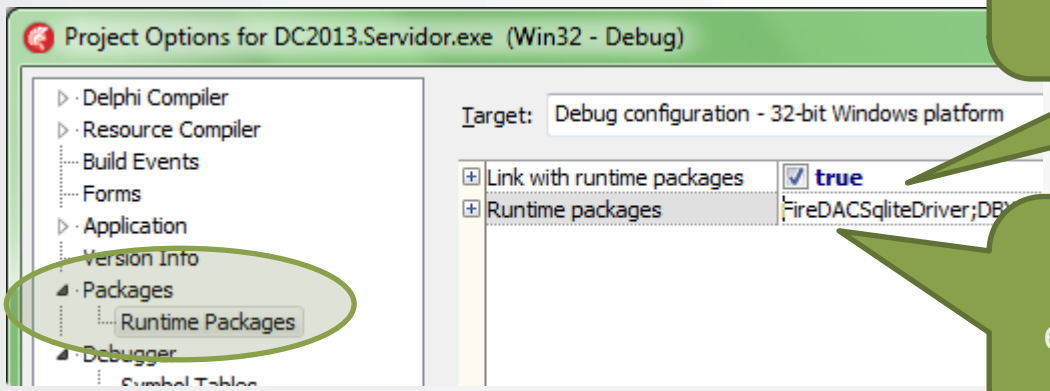
- Deve-se ficar muito atento às dependências e evitar a **importação implícita**.

- Em geral o Delphi detecta e sugere a adição das dependências. Mas isso não acontece para os pacotes que ele não conhece.



Executável fachada

- Deve-se criar um executável preparado para lidar com as BPLs.
- Temos dois modos de dependência: estática e dinâmica.



Esta opção indica ao Delphi para gerar o executável dependente das BPLs ao invés de incluir estaticamente as units/dcu no exe final

Aqui vai a lista de BPLs que o executável carregará estaticamente. *O que não estiver na lista fará parte do executável*

Carga estática

- As BPLs são associadas estaticamente ao executável.
- Na prática o compilador irá escrever no cabeçalho do executável o nome das dependências.
- O Windows, ao carregar este executável, tomará conhecimento das dependências e carregará os módulos acoplando ao processo.
- Isso implica que o executável não irá carregar se as dependências não forem atendidas.

PATH

- O Windows procura os módulos estáticos nos seguintes locais:

  http://msdn.microsoft.com/pt-br/library/7d83bc18(v=vs.90).aspx

  - Diretório do aplicativo.
  - Diretório corrente.
  - Diretório de sistema: \Windows\System32.
  - Diretório do Windows: \Windows.
  - Diretórios listados na variável de ambiente PATH (da esquerda para direita).

- É aqui que a BPL Hell se manifesta!

Carga dinâmica

- A carga dinâmica lhe dá maior flexibilidade de distribuição das BPLs podendo-se efetivamente adotar o conceito de plug-ins.
- Mas tira a fluidez na hora de codificar, pois o Delphi não conhece a BPL em questão e portanto não consegue oferecer os métodos, parâmetros, tipos e etc que estão contidos na BPL.
- O requisito mínimo é **exportar** um ou mais procedimentos. Perceba que é *case sensitive.*
- Sugere-se então que exporte uma função que retorne uma *interface* ou *objeto* que consiga ser manipulado pelo executável fachada.
- Aqui também entra o Padrão de Projeto *"Factory"*, pois uma BPL poderá conter a classe abstrata e outras conterão as classes concretas. Do ponto de vista do executável só se conhecerá a classe abstrata.

Carga dinâmica - Comandos

- Em System.SysUtils temos algumas rotinas para lidar com as BPLs:
  - **LoadPackage:** *Carrega a BPL.*
  - **UnloadPackage:** *Descarrega a BPL.*
  - **GetPackageDescription:** *Retorna a descrição de uma BPL.*
  - **GetPackageInfo:** *Retorna diversas informações sobre a BPL, como o número da versão.*

Comandos para interagir com a BPL

- Para lidar com os recursos oferecidos pela BPL podemos usar alguns comandos (Winapi.Windows):
  - **GetProcAddress**: *Retorna o ponteiro de um método.*
  - **GetClass:** *Retorna uma referência de classe, permitindo instanciar objetos.*

Juntando tudo

EXEMPLOS PRÁTICOS

Proposta dos exemplos

- Evoluiremos o exemplo entre os seguintes cenários:
  - Aplicação Win32 monolítica
  - Aplicação Win32 modular
  - Aplicação DataSnap\REST modular
- Será utilizado um exemplo didático de “Olá Mundo”.
- Baixe os exemplos de: https://github.com/jmarioguedes/DC2013_BPL

Links

O formato PE
Vovó Vicki
http://www.numaboa.com.br/informatica/oraculo/230-formatos/1096-formato-pe

Advanced Dynamic Packages
Vino Rodrigues
http://delphi.cz/img/packages/advanced.pdf

Modularização de Aplicações
Álvaro Esteves Alves
http://www.activedelphi.com.br/modules.php?op=modload&name=News&file=article&sid=719

BPLs no Delphi
Paulo Quicoli
http://www.devmedia.com.br/websys.5/webreader.asp?cat=3&artigo=4709&revista=clubedelphi_142#a-4709

Pacotes em Delphi (Modulos)
Rafael Ferreira
http://www.portaldaprogramacao.com/artigos2.asp?n=882

Biblioteca (Computação)
Wikipédia
http://technet.microsoft.com/en-us/sysinternals/bb842062

Muito obrigado!

```
{
  “nome”    : “Mário Guedes” ,
  “e-mail” : “mario.guedes@arrayof.com.br” ,
  “blog”    : “http://eugostododelphi.blogspot.com.br”
  “perfis” : [
               {“twitter”     : “eugostododelphi”} ,
               {“facebook”    : “eugostododelphi”} ,
               {“linkedin”    : “jmarioguedes”} ,
               {“slideshare” : “jmarioguedes”} ,
               {“github”      : “jmarioguedes”} ,
               {“prezi”       : “jmarioguedes”}
             ]
}
```

Canais Embarcadero

Canais Embarcadero

http://edn.embarcadero.com

http://www.embarcadero.com/br

http://www.facebook.com/DelphiBrasil

http://www.facebook.com/EmbarcaderoBrasil

http://www.embarcadero.com/mvp-directory

http://www.embarcaderobr.com.br/treinamentos/