



Desenvolvimento de grandes aplicações com a programação orientada a objeto do LabVIEW

Alisson Kokot

Engenheiro de Vendas

Oswaldo Santos

Engenheiro de Sistemas

Agenda

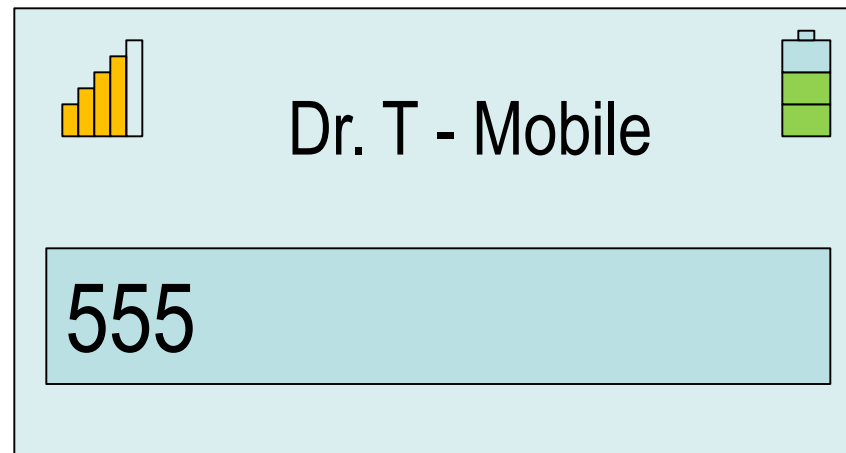
1. **Arquitetura da Aplicação**
2. Apresentação Separada
3. Inserção de Dependências
4. HAL – Camada de Abstração de Hardware

Desafios no Projeto de Grandes Sistemas

- Complexidade
- Construído por times
- Gerenciar Mudanças
- Múltiplas versões de produtos

DEMO

JKIphone



?

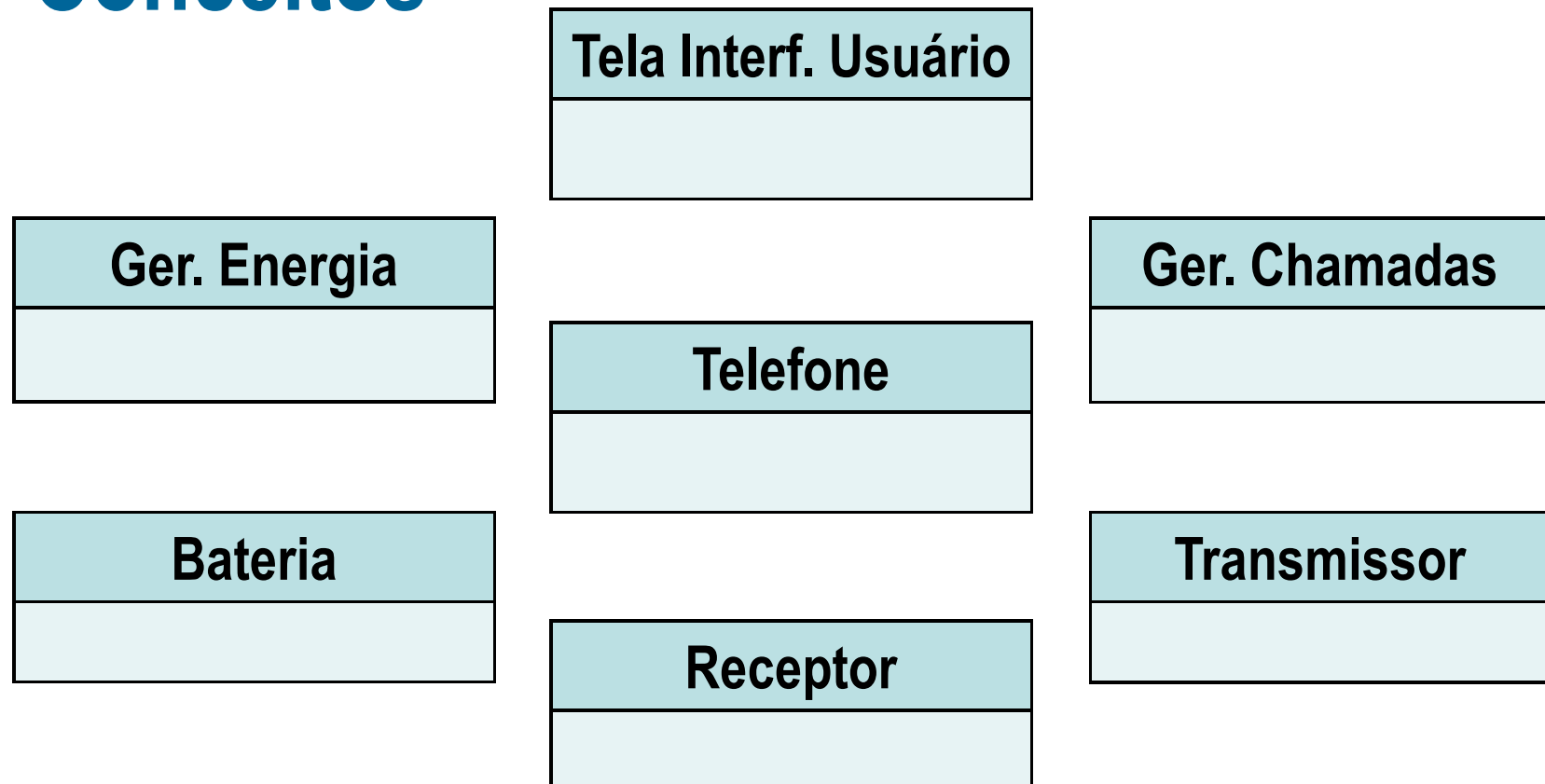
Componentes de Hardware

Bateria

Receptor

Transmissor

Modelo de Objeto – Separação de Conceitos



Projeto em camadas

Camada de
Apresentação

Apresentação
e lógica da apresentação

Camada de
Serviço

Coordenação de acesso
ao negócio / Dominio lógico

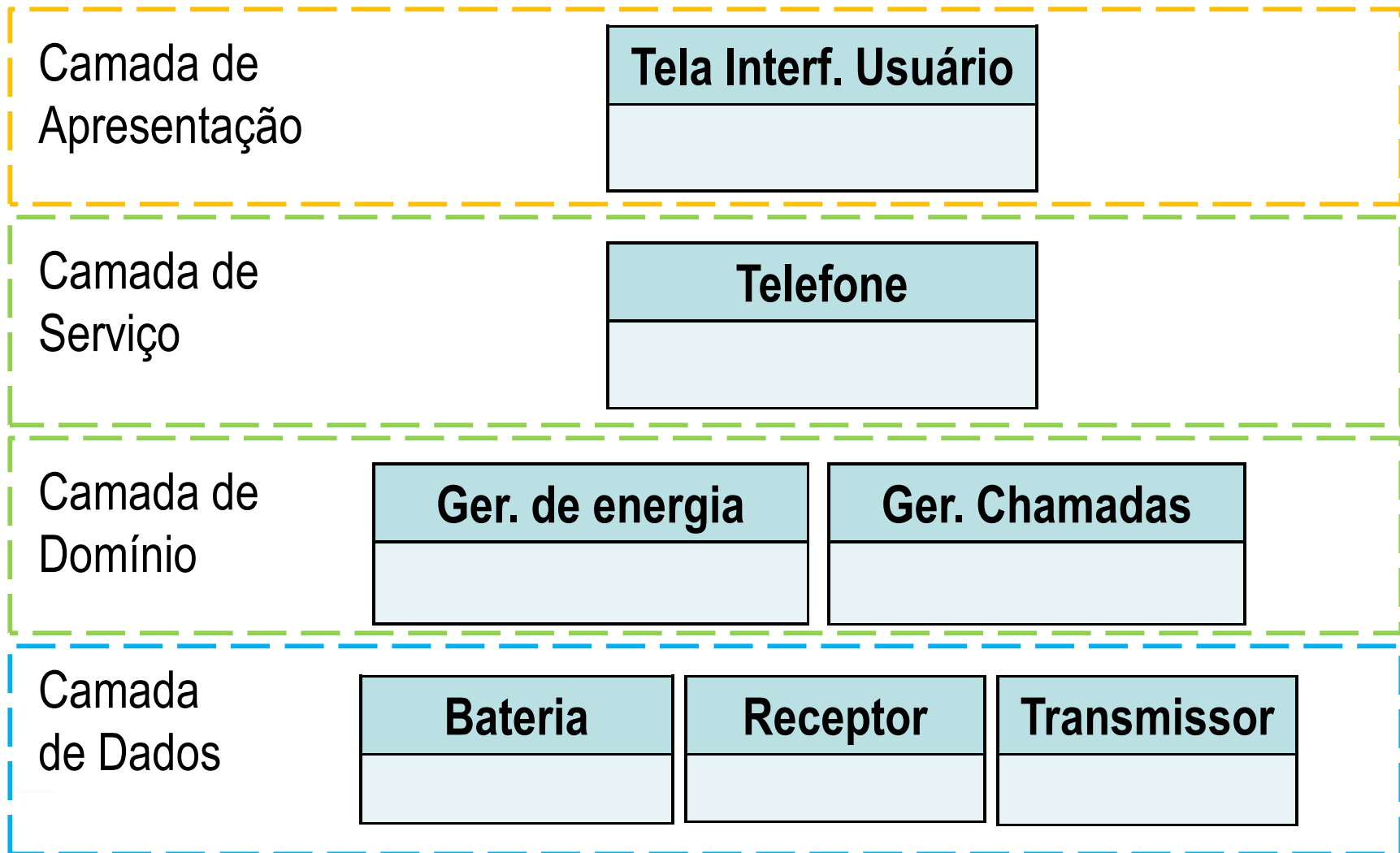
Camada de
Domínio

Negócio / Dominio lógico

Camada de
Dados

Dados e abstração de Hardware

Exemplo de projeto em camadas



Conteúdo

1. Arquitetura da Aplicação
- 2. Apresentação Separada**
3. Inserção de Dependências
4. HAL – Camada de Abstração de Hardware

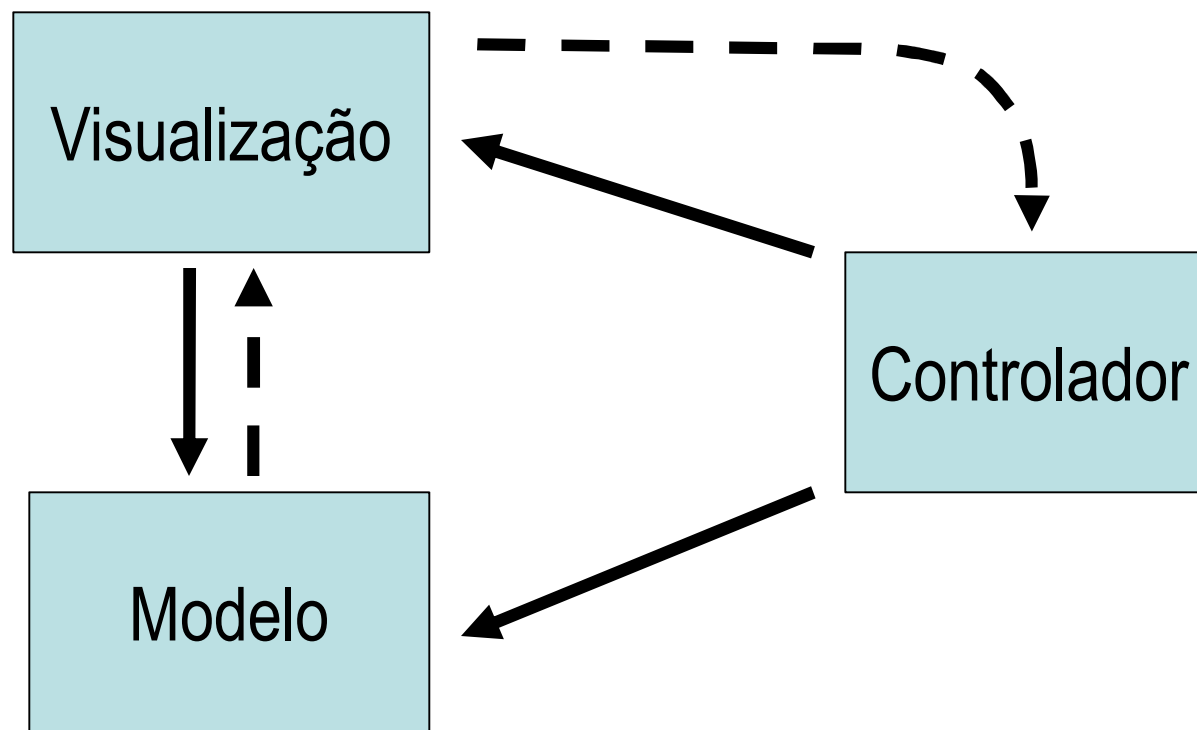
Apresentação Separada

Código que manipula apresentação

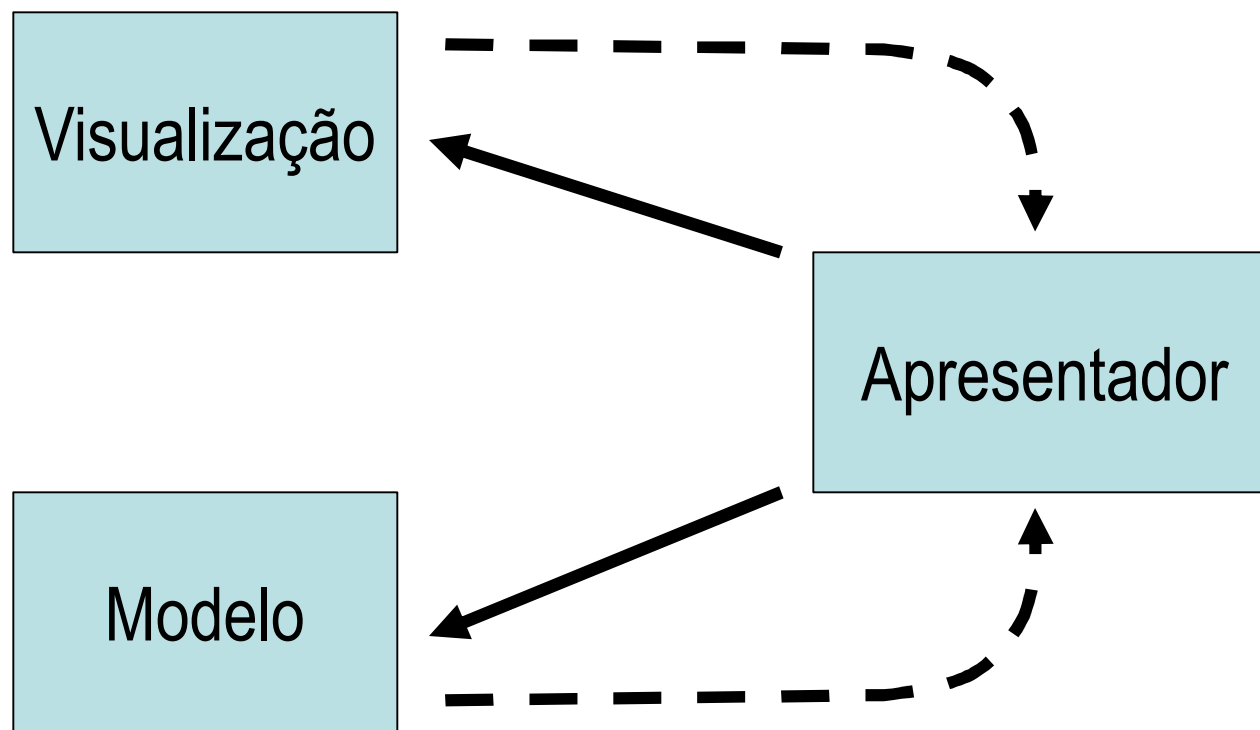
Somente manipula apresentação

Todo o domínio e fonte de dados lógicos
em partes claramente separadas do programa

Padrão de Projetos Modelo-Visualização-Control

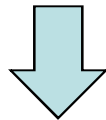


Padrão de projetos Modelo-Visualização-Apresentador

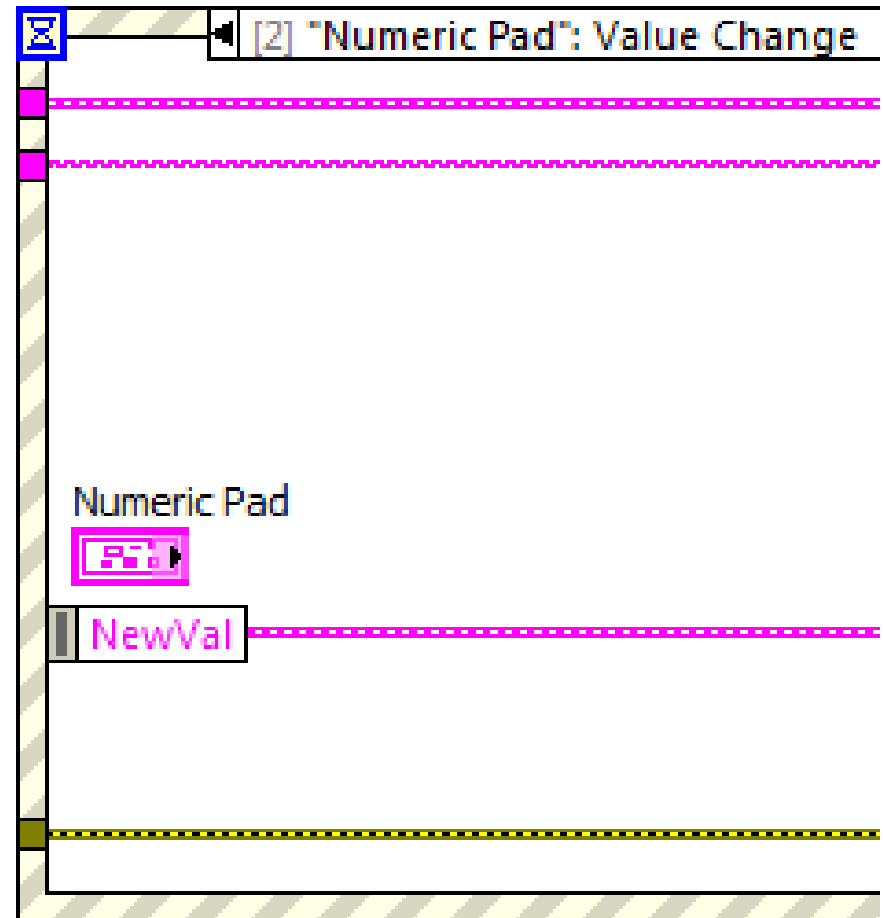


Natureza Especial da Interface de Usuário no LV

- Eventos de usuário são convenientemente gerenciados somente no VI que contém a interface de usuário



- Separação de interface de usuário da lógica pesada



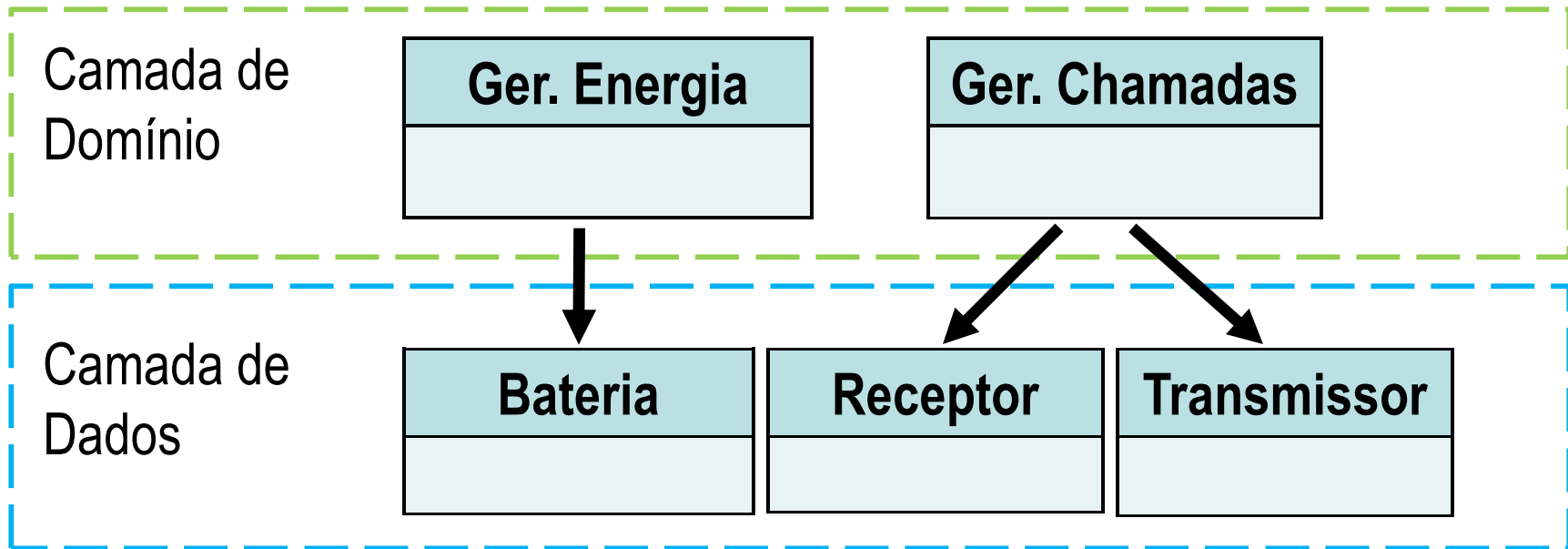
Garantindo uma apresentação testável

- Realizar Unit Testing de uma máquina de estados é um desafio
- Mova a apresentação lógica da máquina de estados para VIs
- A máquina de estados deve atuar apenas como um chamador

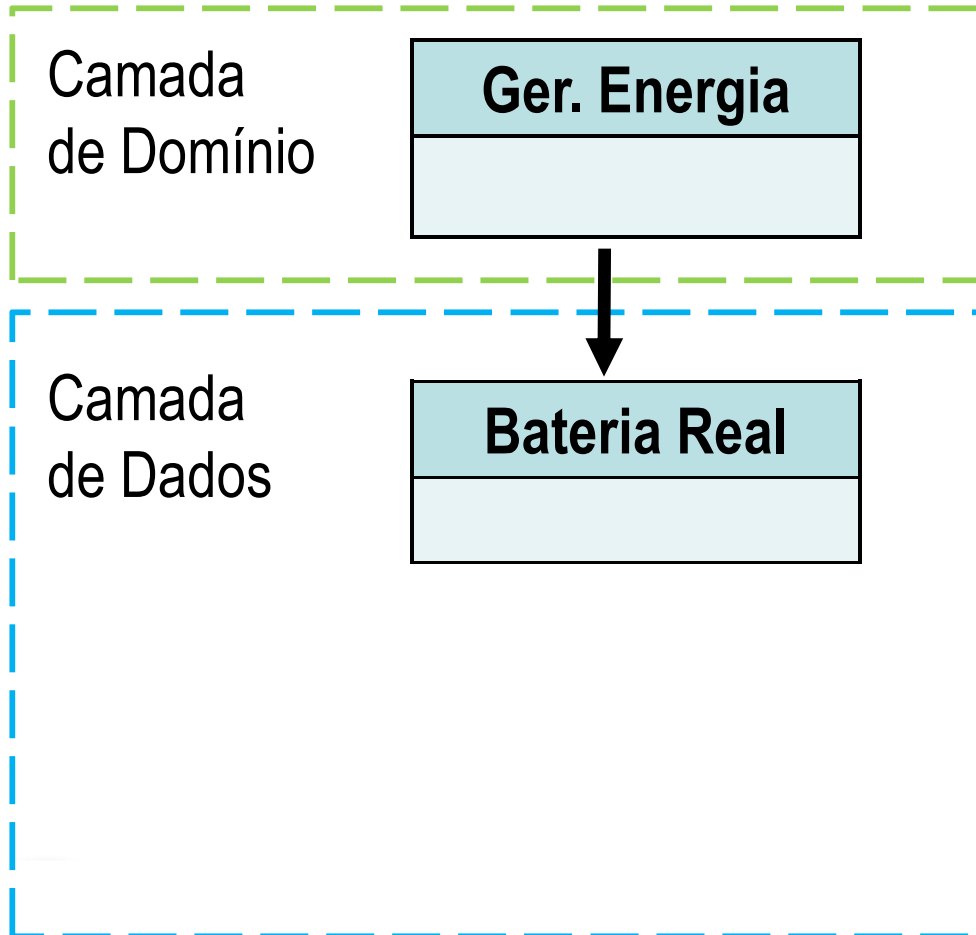
Conteúdo

1. Arquitetura da Aplicação
2. Apresentação Separada
- 3. Inserção de Dependências**
4. HAL – Camada de Abstração de Hardware

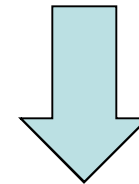
Dependências Inter-Classes



Testando Gerenciamento de Energia

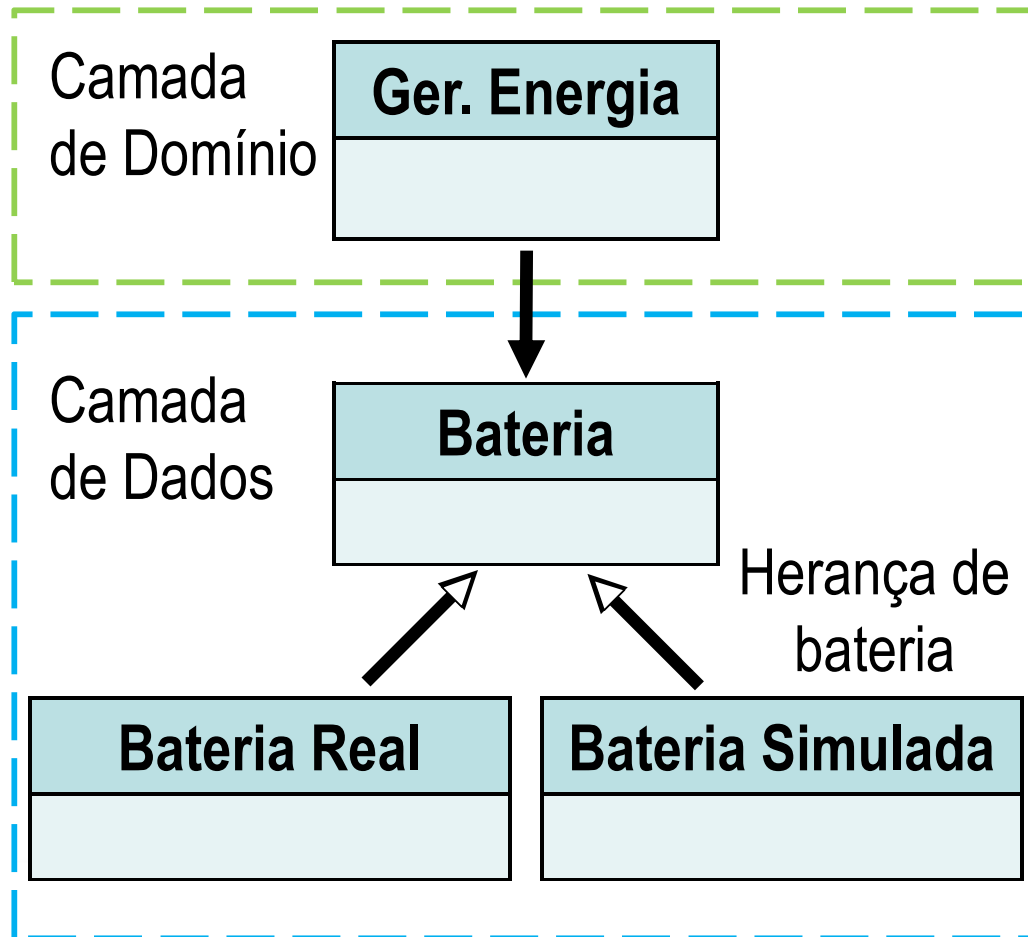


- O Gerenciamento de energia é acoplado a bateria real



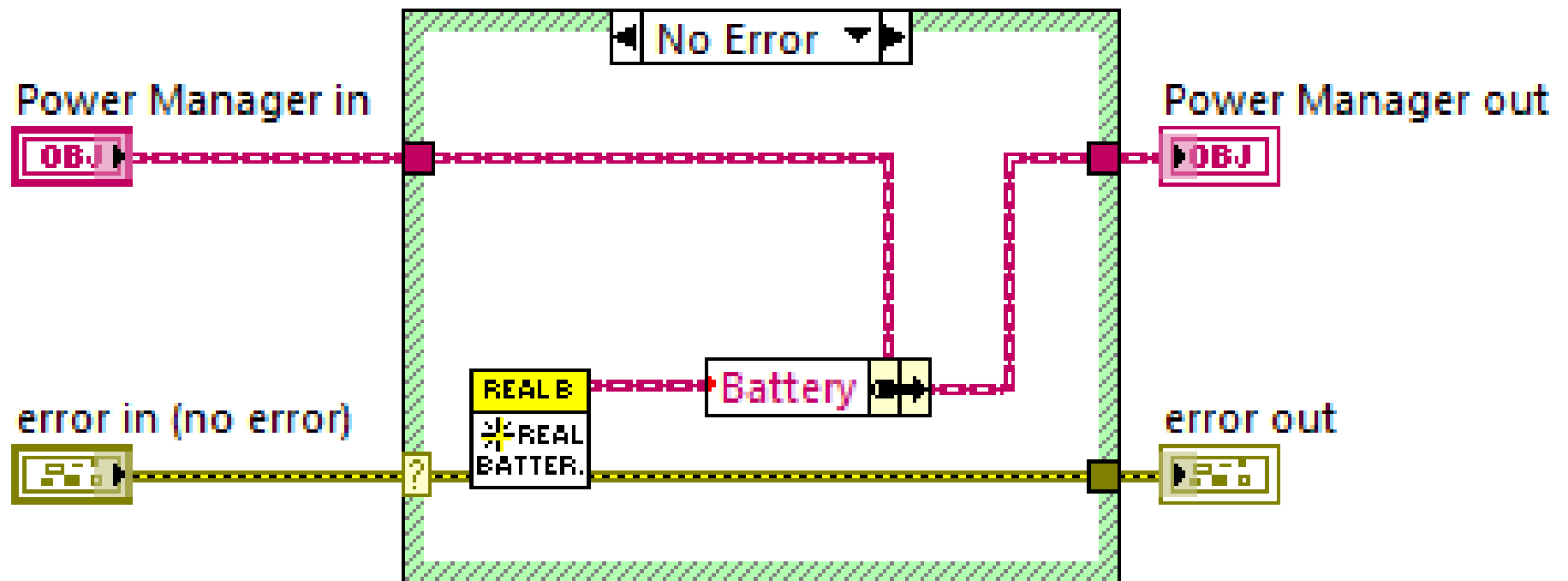
- O teste automático é difícil

Simulando Hardware



- Como podemos alternar entre o hardware real e simulado?

Construindo Gerenciamento de Energia sem Inserção de dependências



Conteúdo

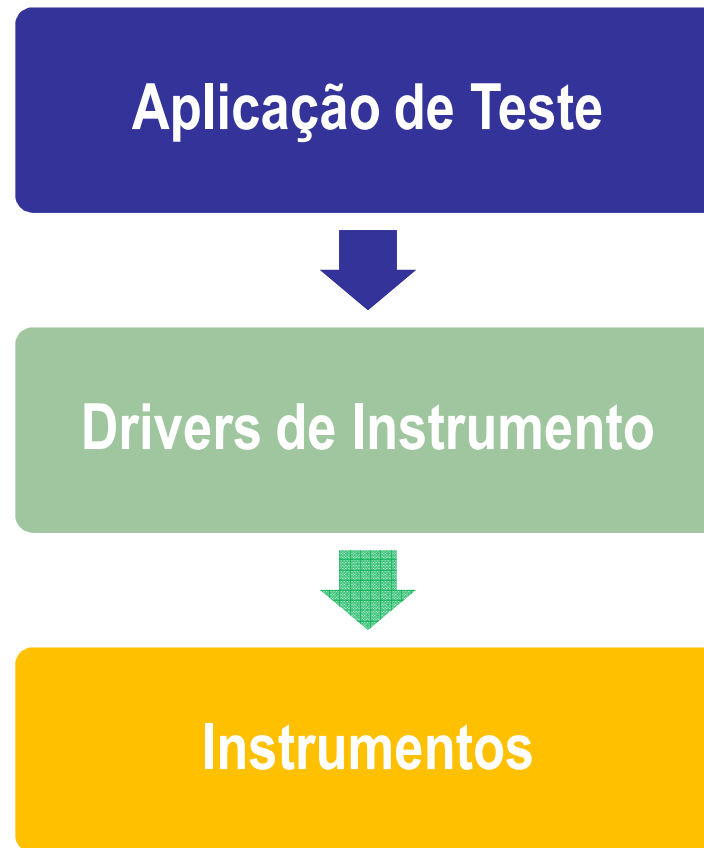
1. Arquitetura da Aplicação
2. Apresentação Separada
3. Inserção de Dependências
4. **HAL – Camada de Abstração de Hardware**

Ciclos de Vida Diferentes

- Produtos com ciclos de vida de décadas ou de meses
- Ciclo de Vida Longo
 - O dispositivo em teste não muda
 - Os instrumentos usados no teste se tornam obsoletos
- Ciclo de Vida Curto
 - O dispositivo em teste não muda
 - Os instrumentos usados para o teste continuam o mesmo
 - A aplicação do teste muda



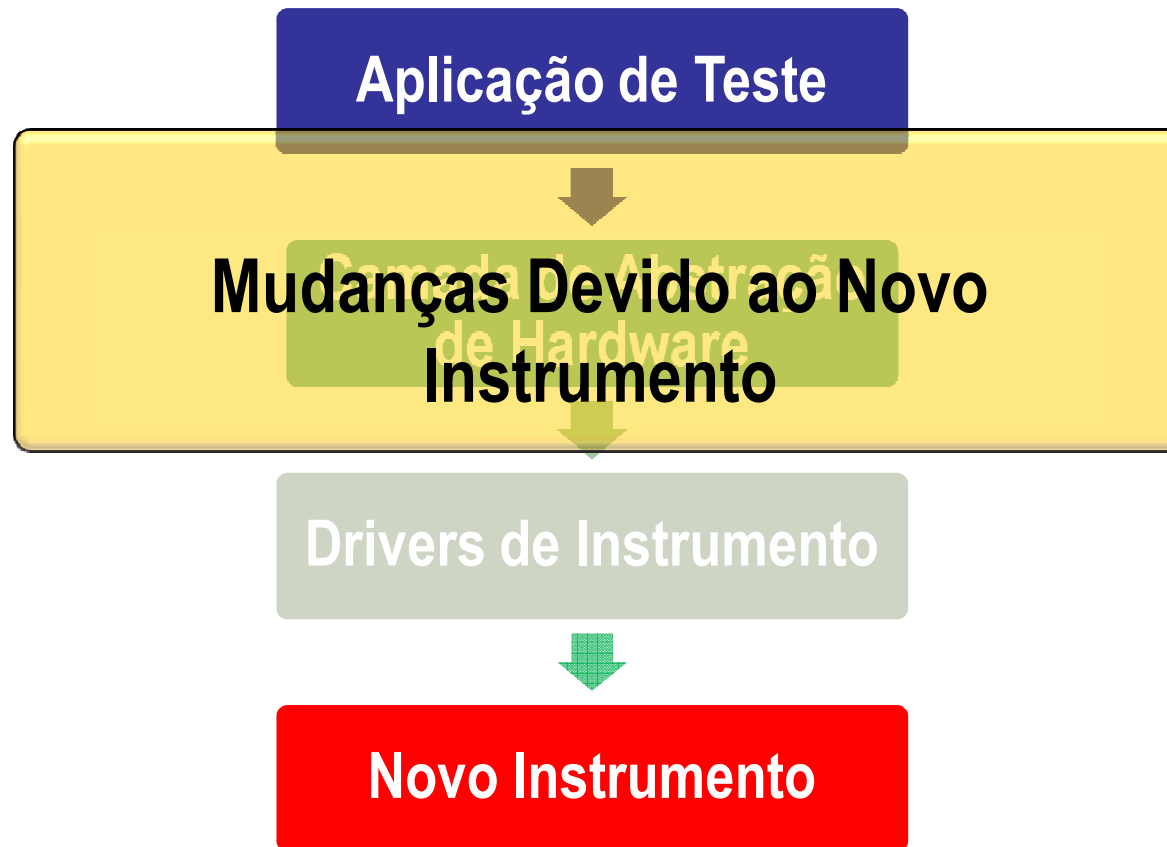
Aplicação Típica de Teste



Aplicação Típica de Teste: Migração



Aplicação Típica de Teste: Migração



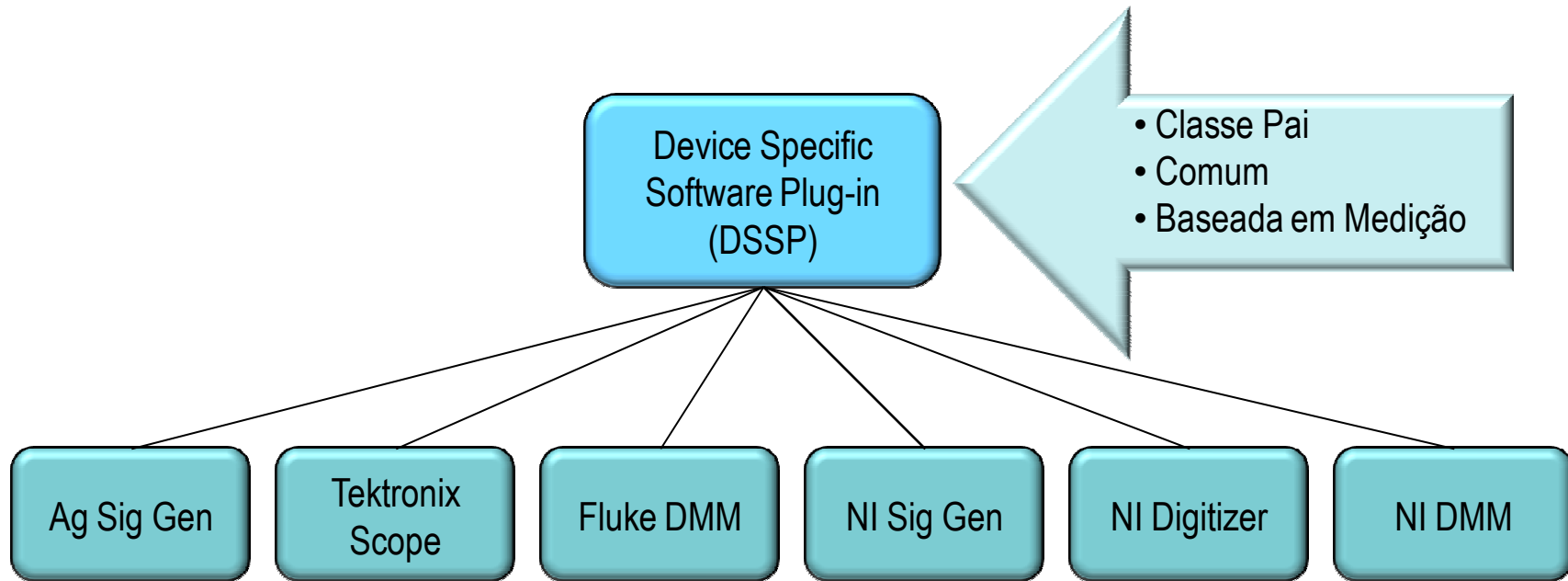
Benefícios do HAL

- Gerenciamento de Equipamentos Obsoletos
- Inserção de Tecnologia
- Menor Custo de Migração
- Menor Tempo de Migração
- Flexibilidade e Reuso
- Manutenção Simplificada

Desafios

- Desenvolvimento mais caro
- Convencer Gerenciamento
- Projeto Não Trivial

Implementação da Hierarquia de Classes



Implementação da Classe Pai

