



NIWeek 2010 Session TS3037

LabVIEW Object-Oriented
Programming Design Patterns for
Large Systems

Tomi Maila

JKI

Aug. 5 2010

Content

- 1. Introduction**
2. Application Architecture
3. Model View Controller
4. Dependency Injection

Challenges in Large System Design

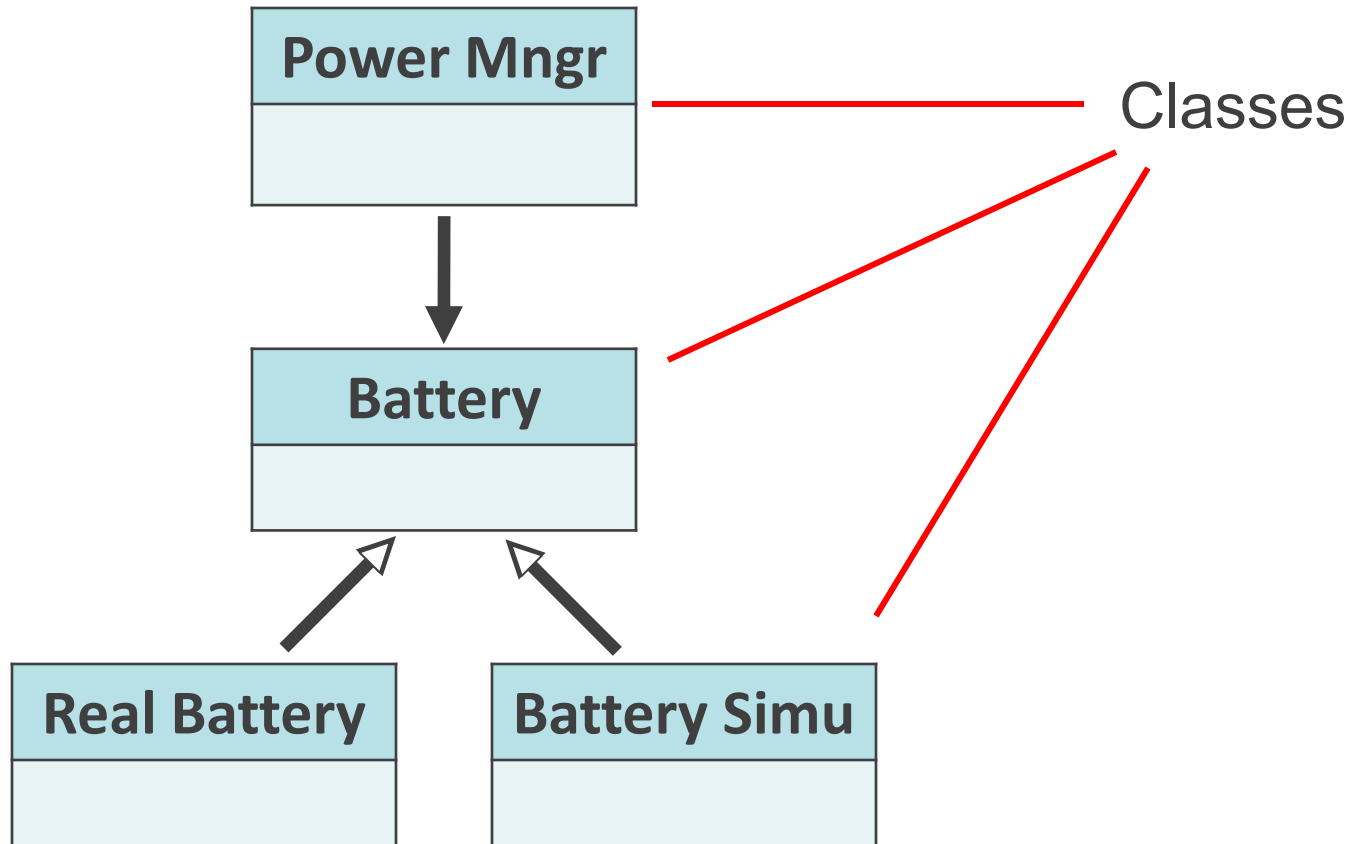
- Complex
- Built by Teams
- Undergo Changes
- Multiple Product Versions

What Is a Design Pattern?

A design pattern is a template for a design that solves a general, recurring problem in a particular context.

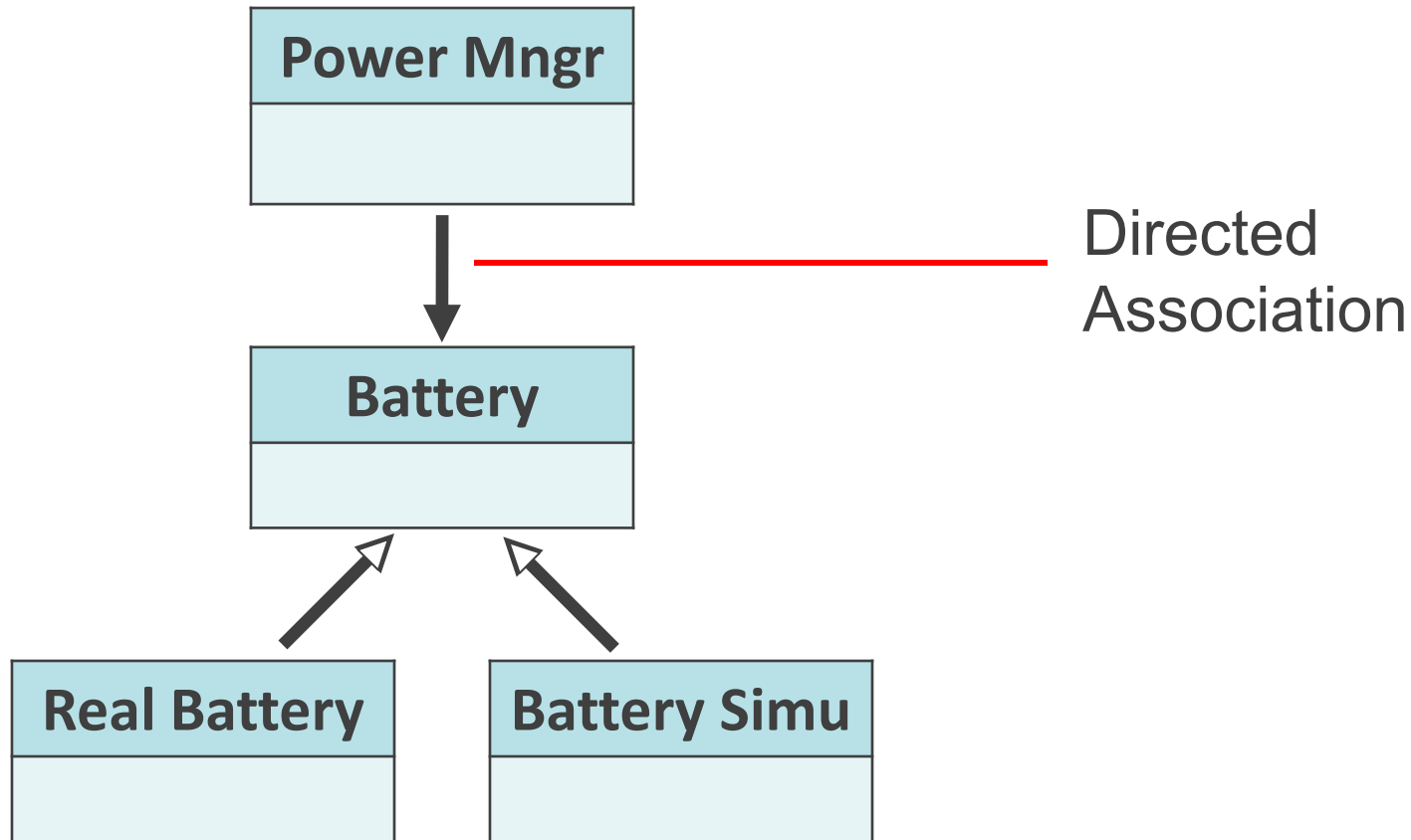
Basic Object Oriented Programming Concepts

Classes



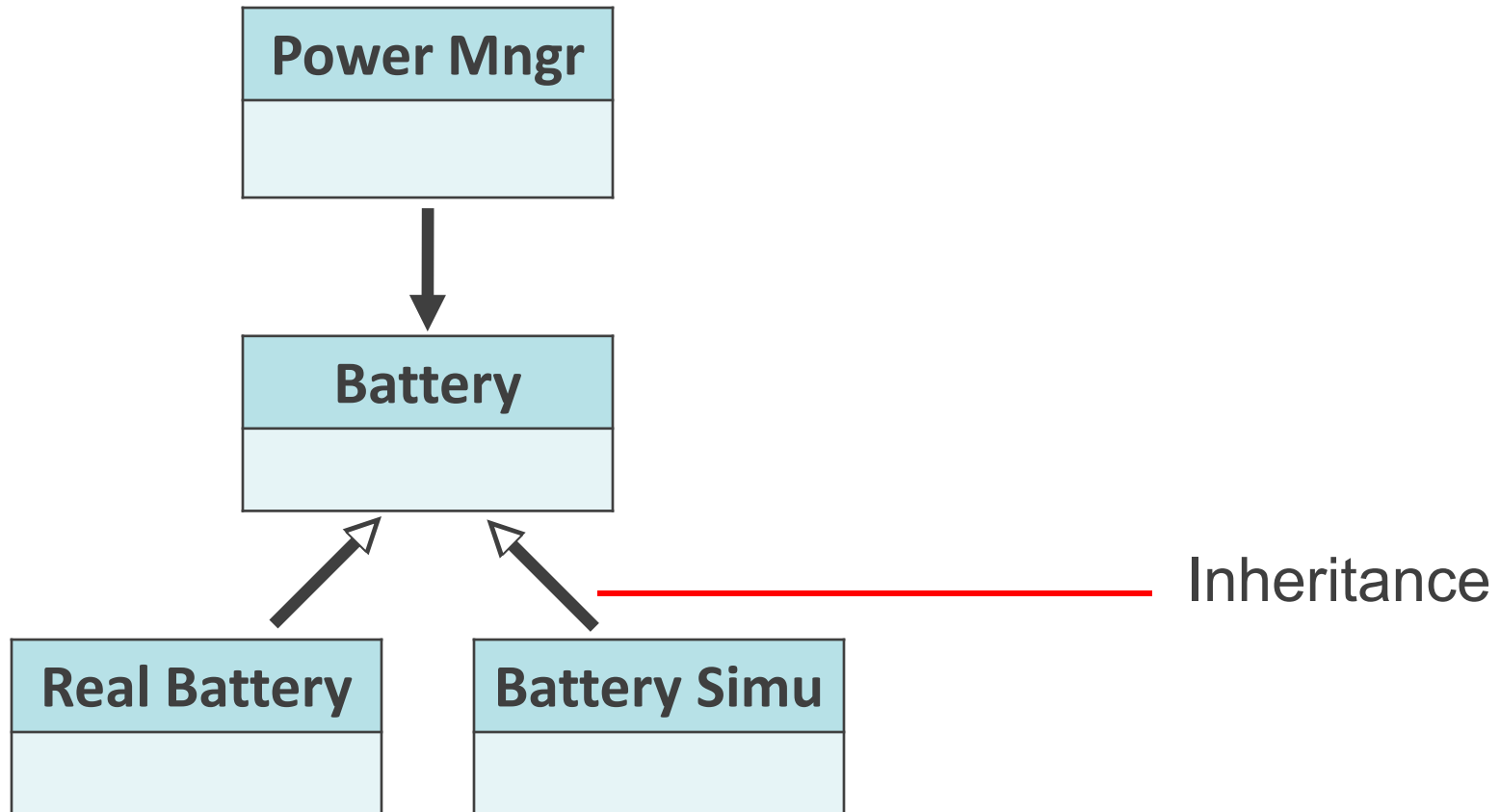
Basic Object Oriented Programming Concepts

Associations



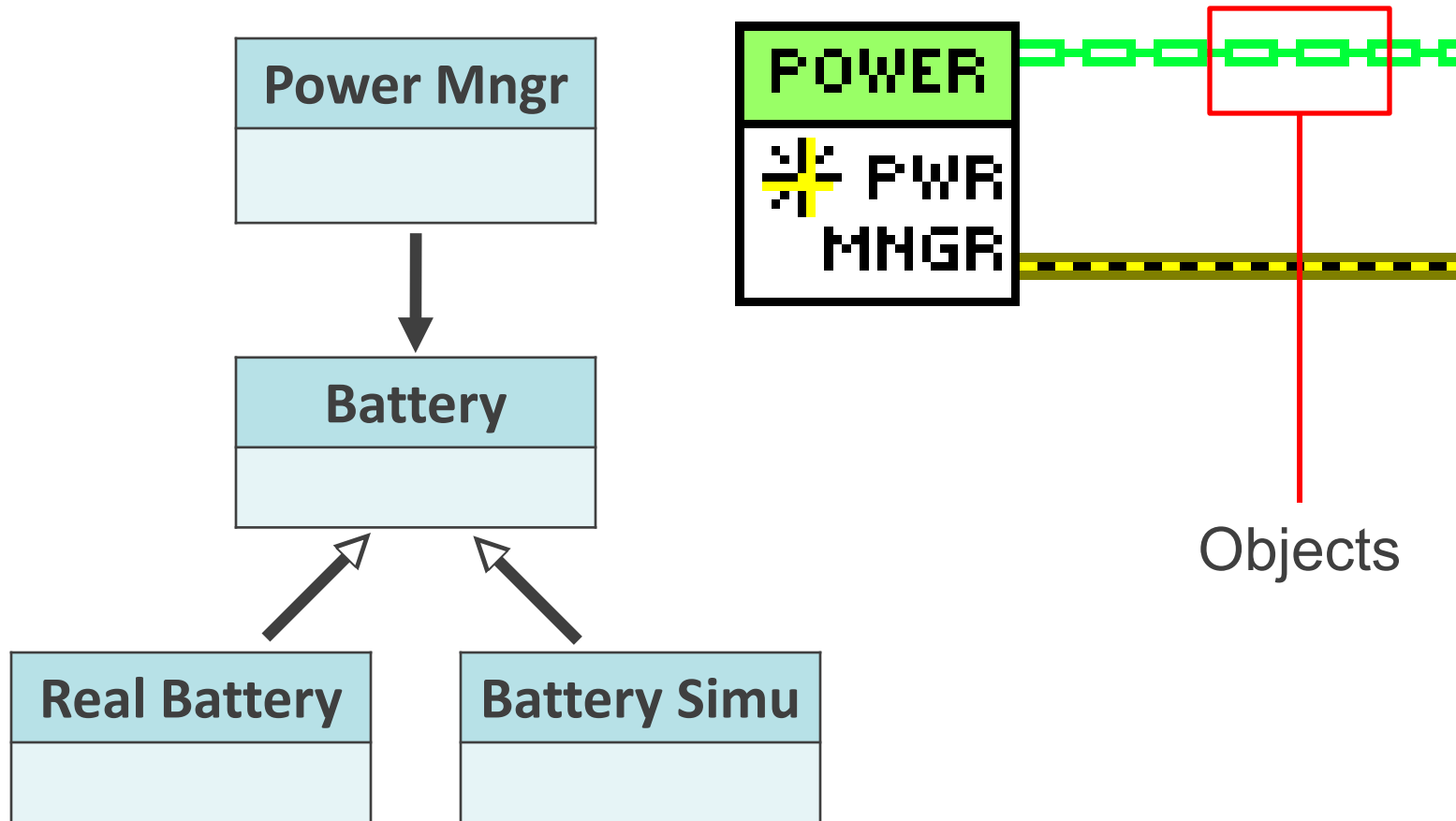
Basic Object Oriented Programming Concepts

Inheritance



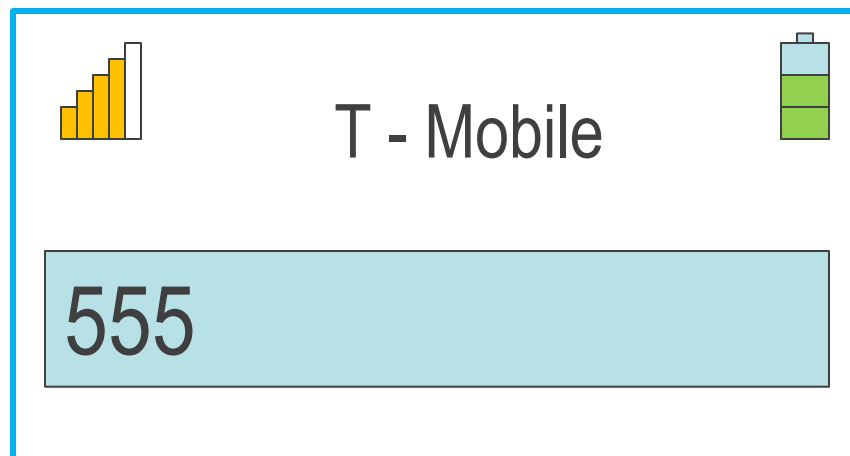
Basic Object Oriented Programming Concepts

Objects



Content

1. Introduction
- 2. Application Architecture**
3. Model View Controller
4. Dependency Injection



?

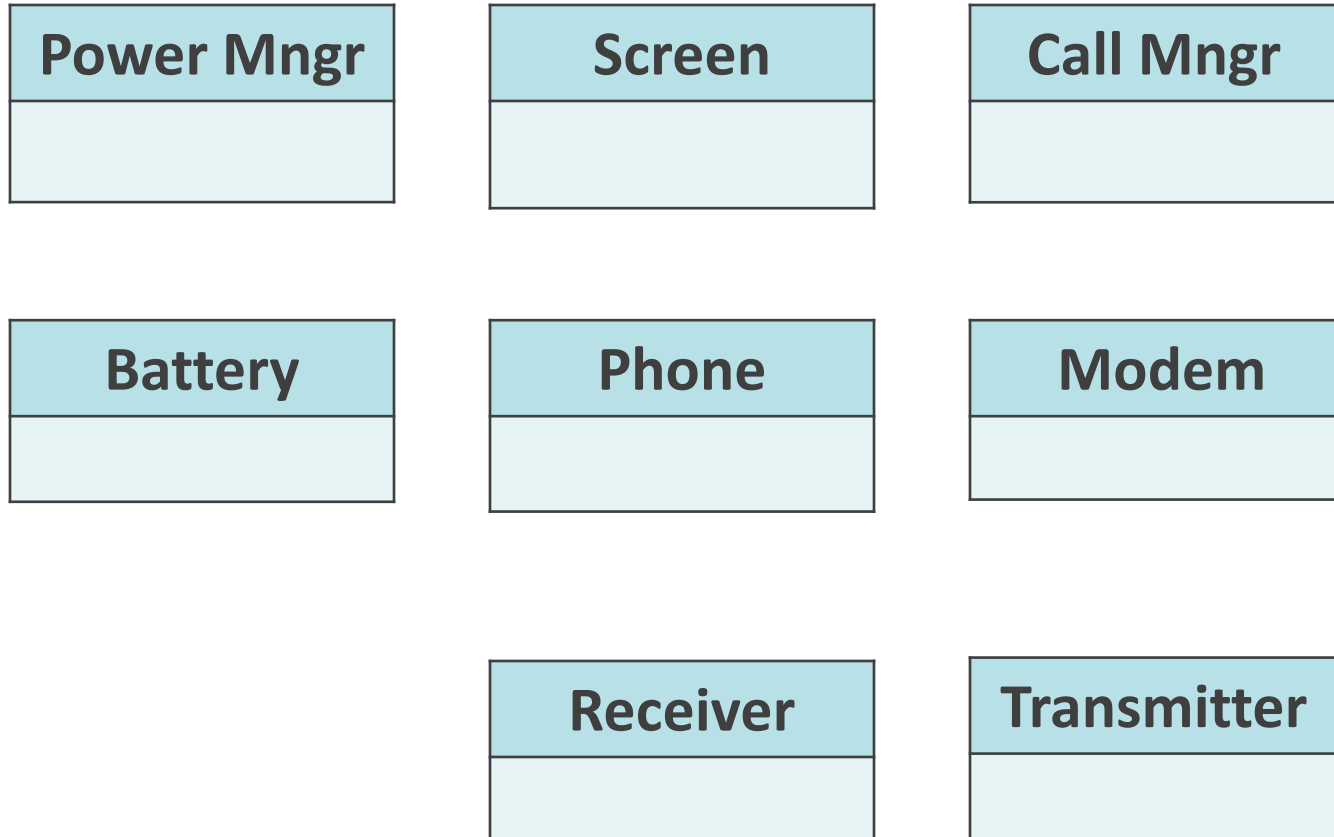
Hardware components

Battery

Receiver

Transmitter

Object Model - Separation of Concerns



Layered Design

Presentation Layer

Domain Layer

Data & Hardware Layer

Layered Design

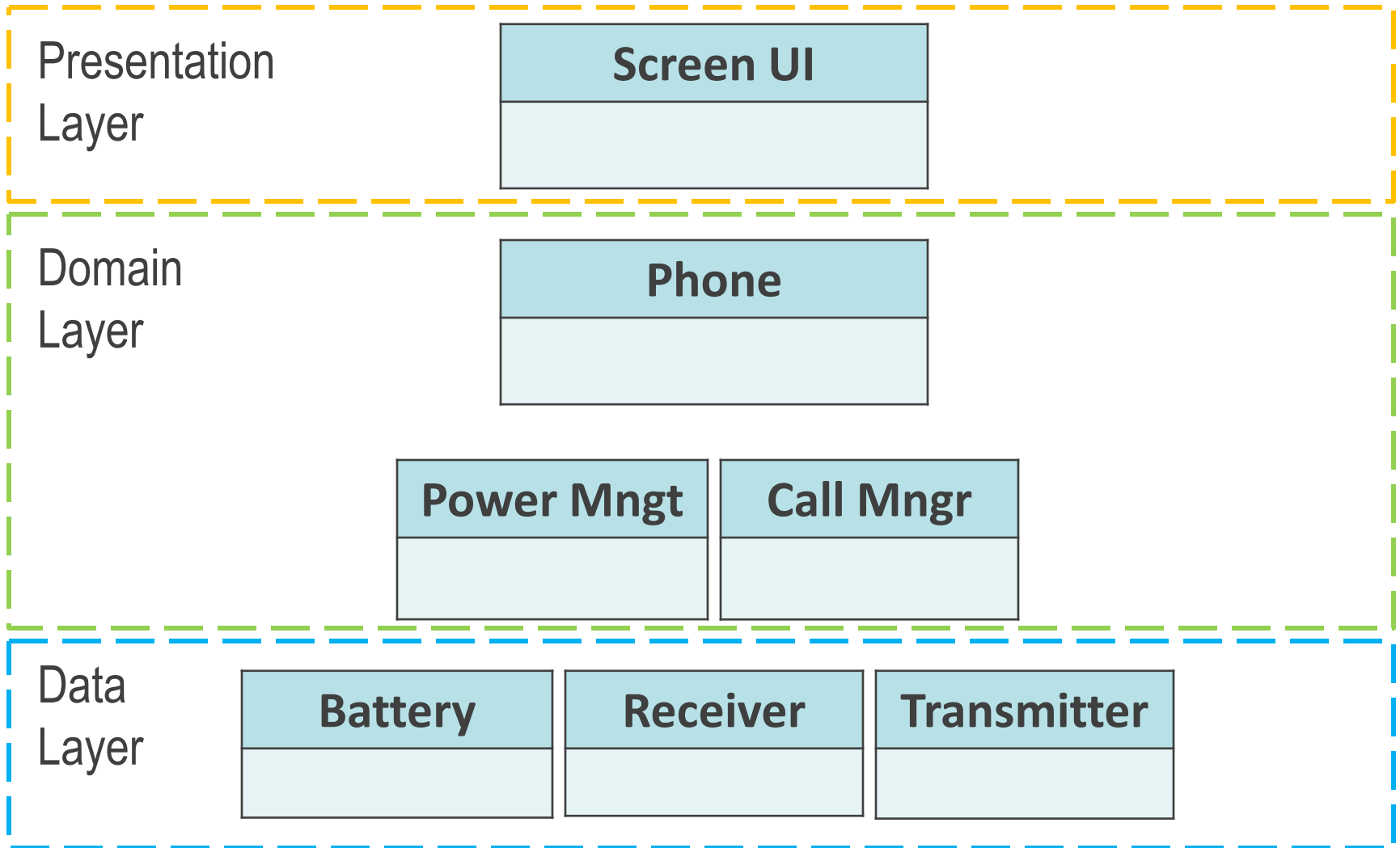
User Interface Components

User Interface Logics

Domain Layer

Data & Hardware Layer

Example of Layered Design



Content

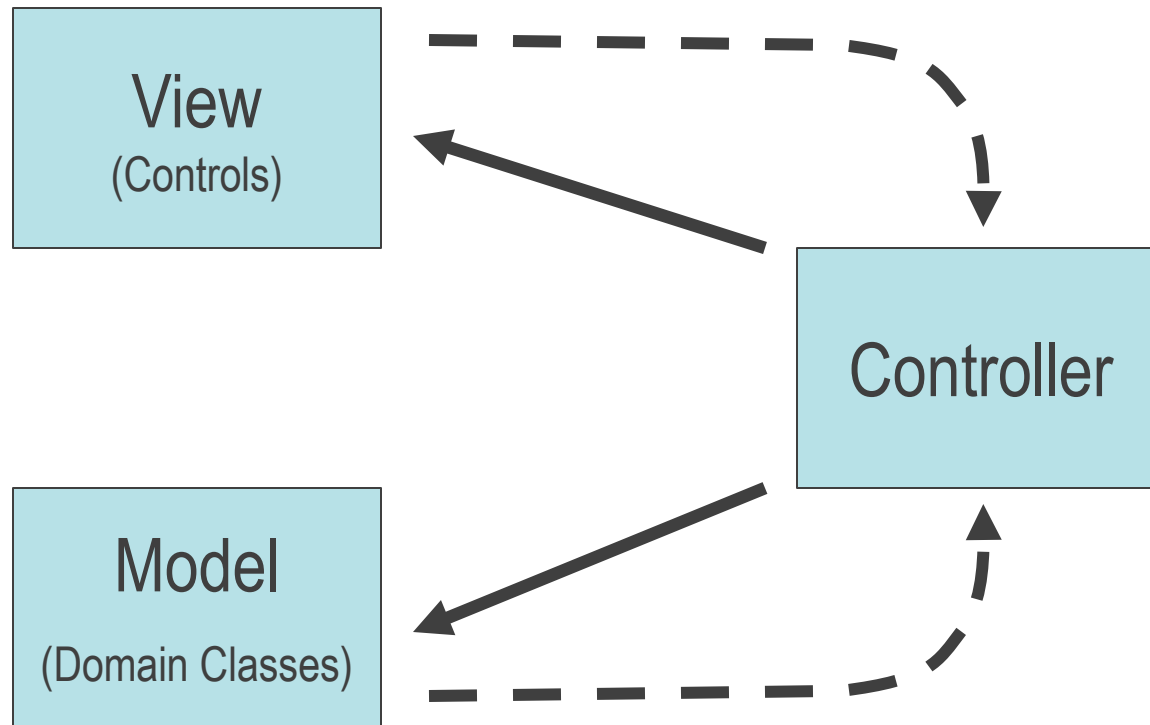
1. Introduction
2. Application Architecture
- 3. Model View Controller**
4. Dependency Injection

Separated Presentation

Code that manipulates presentation
only manipulates presentation

All domain and data source logic
in clearly separated areas of the program

Model-View-Controller

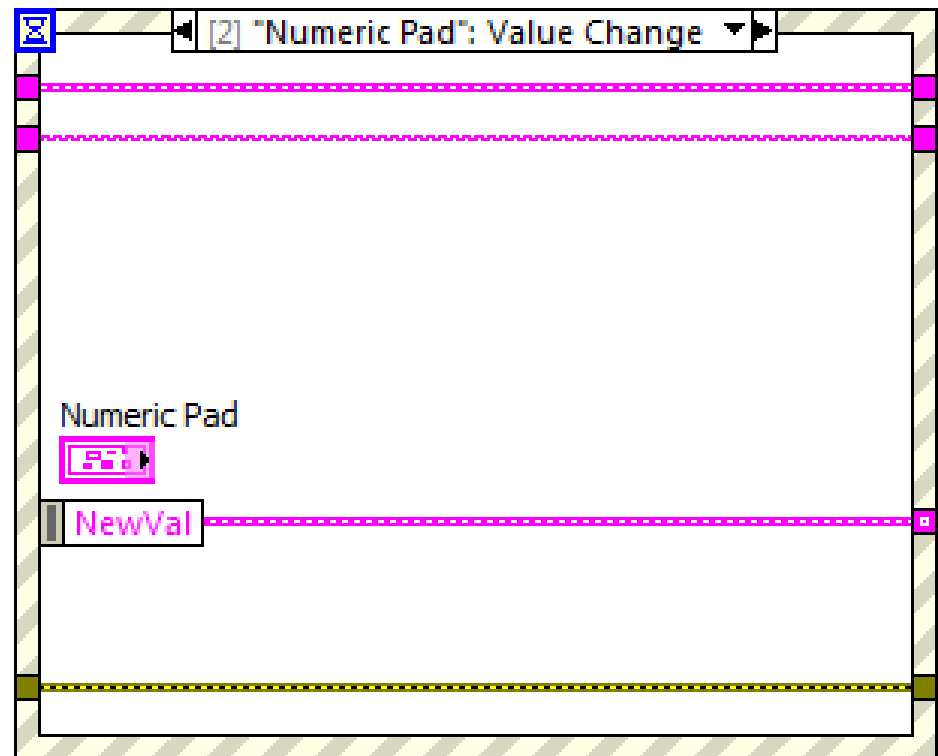


Special Nature of User Interface in LV

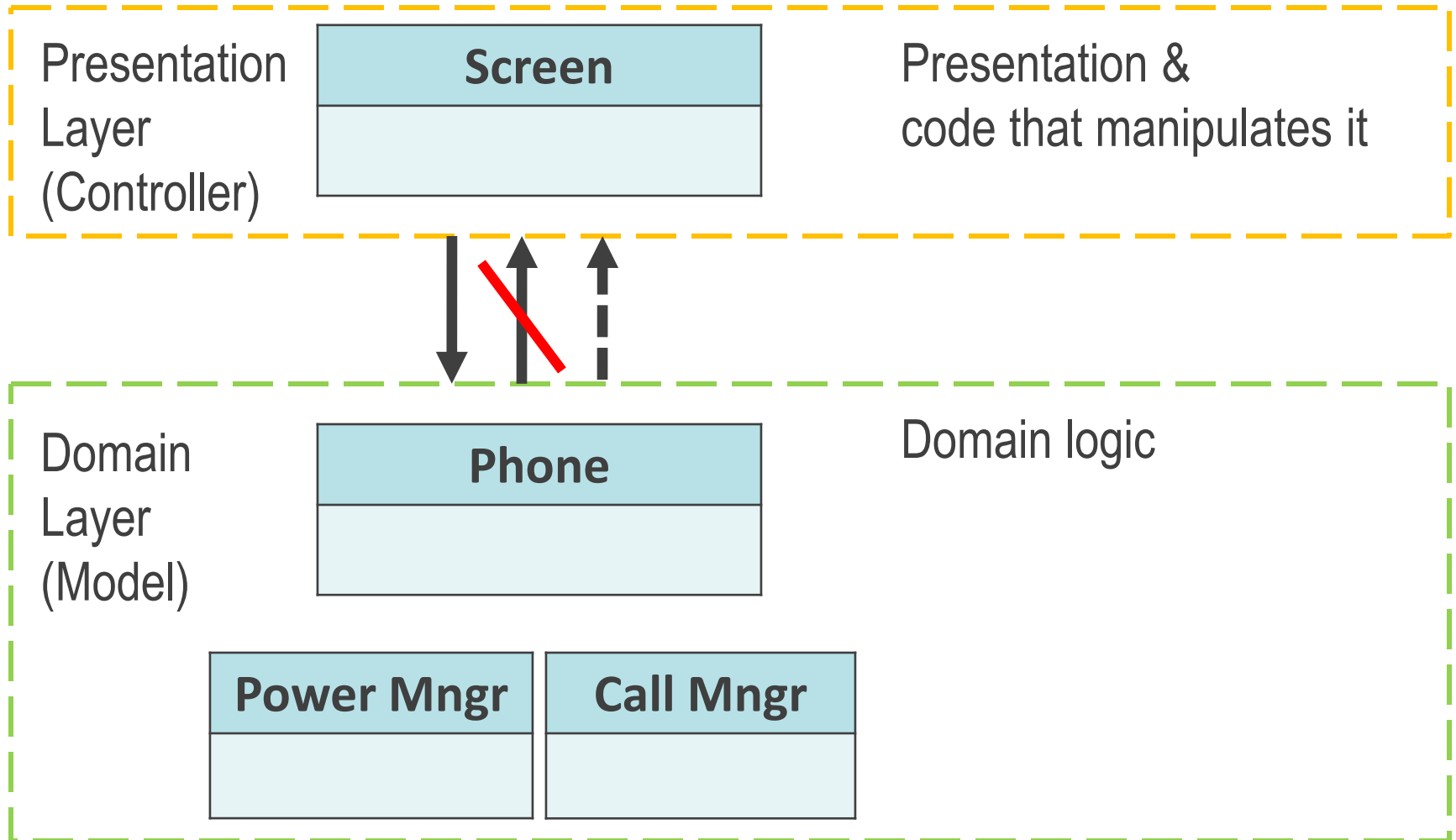
- UI events convenient to handle only in the VI owning the front panel



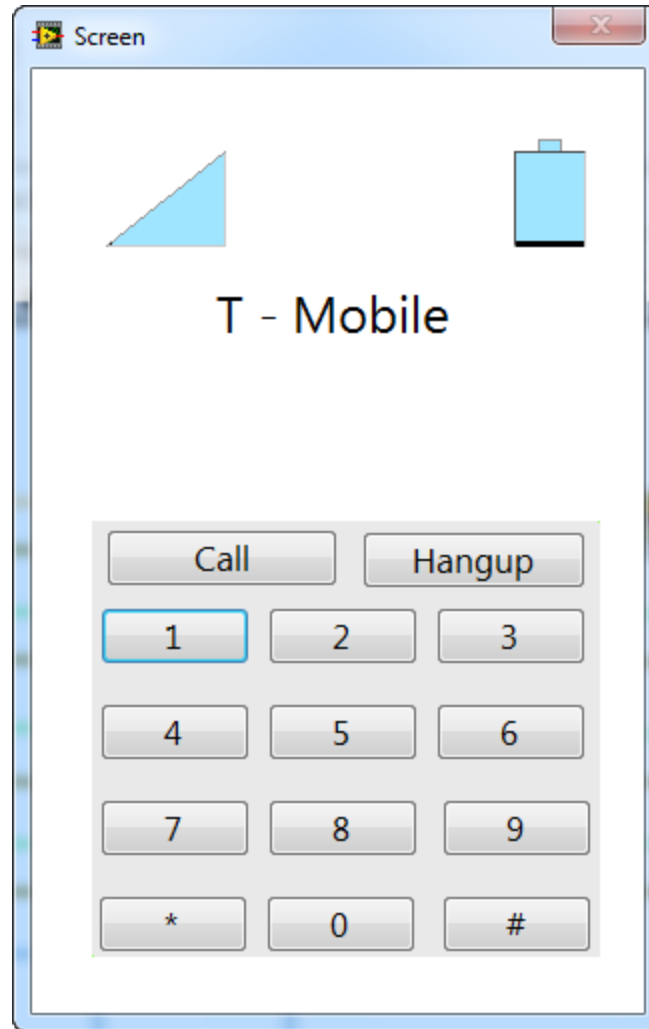
- Separation of user interface from its logic laborous



Model – Controller Interface



Demo



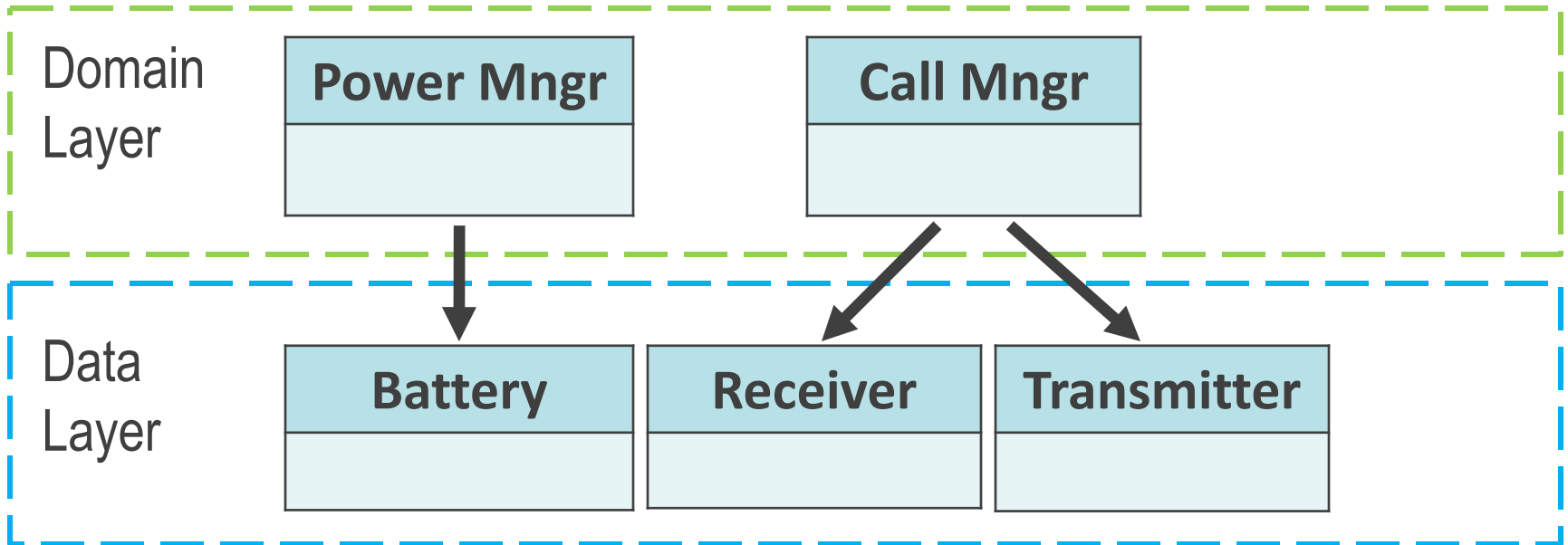
Ensuring Presenter Testability

- Unit testing of state machine code is a challenge
- Move presentation logic from state machine to method Vis
- State machine should act as simple dispatcher

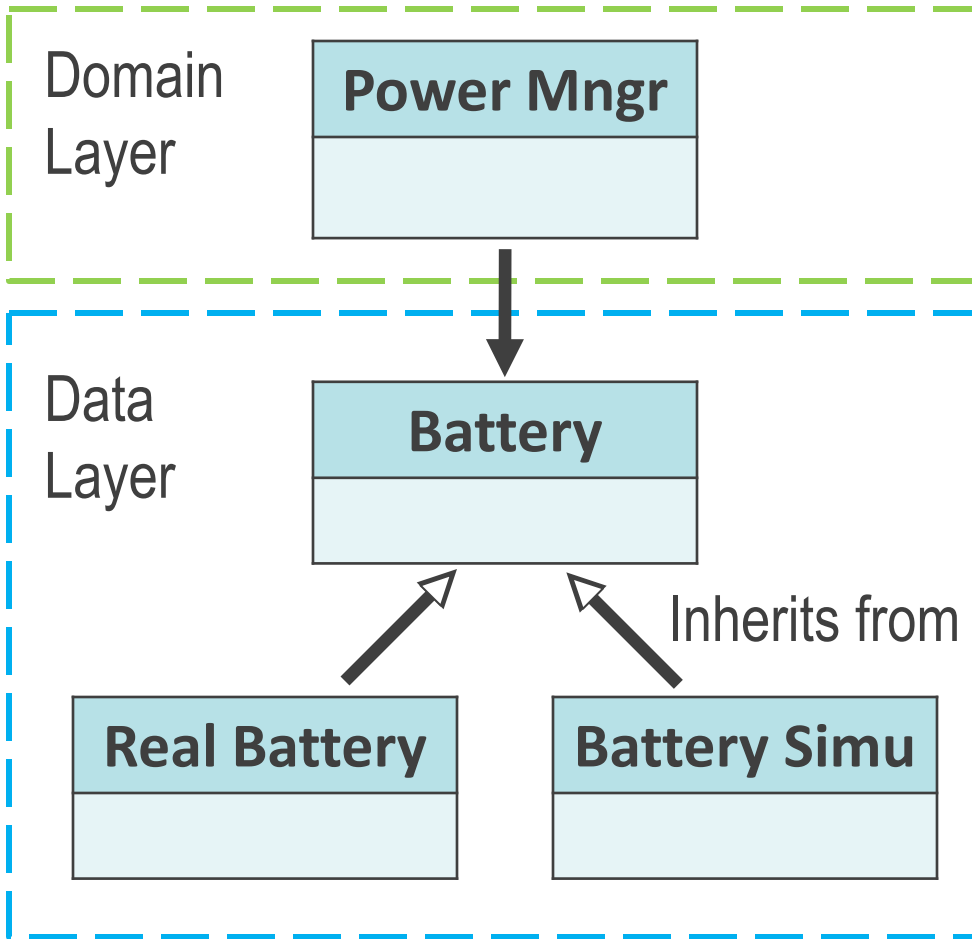
Content

1. Introduction
2. Application Architecture
3. Model View Controller
- 4. Dependency Injection**

Inter-Class Dependencies

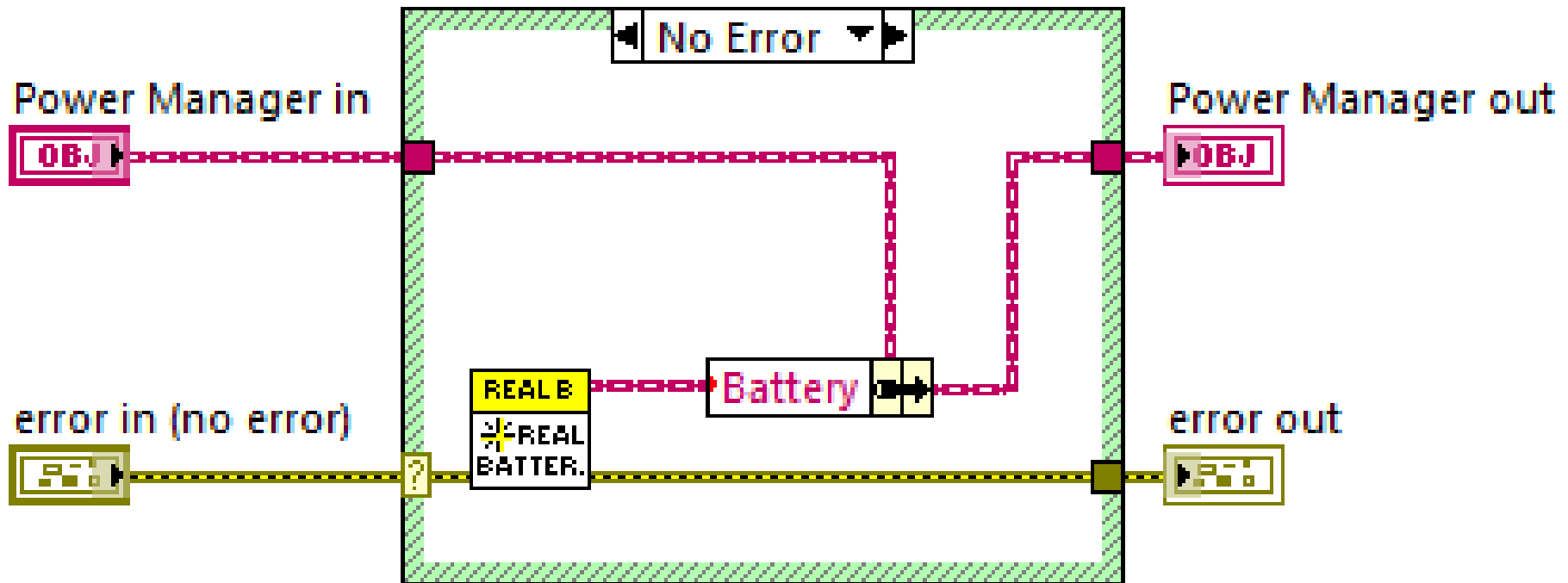


Simulating Hardware

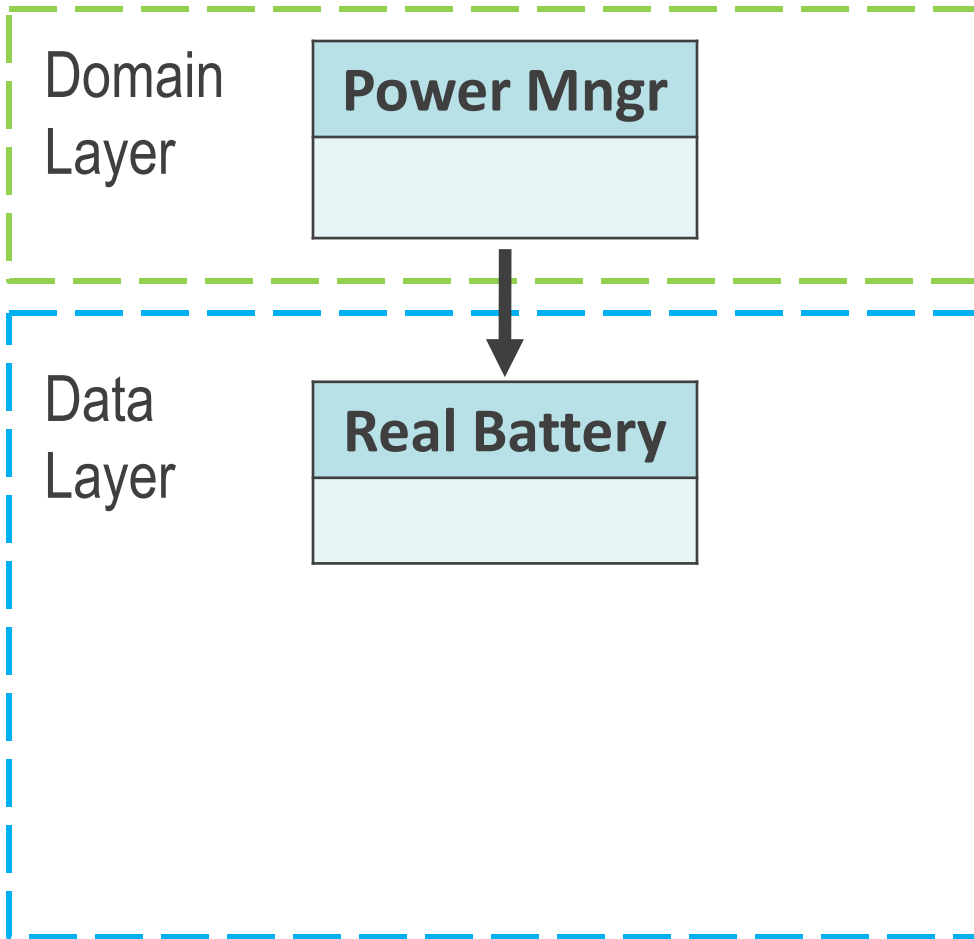


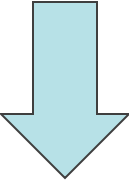
- How do we switch between real and simulated hardware?

Constructing Power Manager w/o Dependency Injection



Testing Power Manager

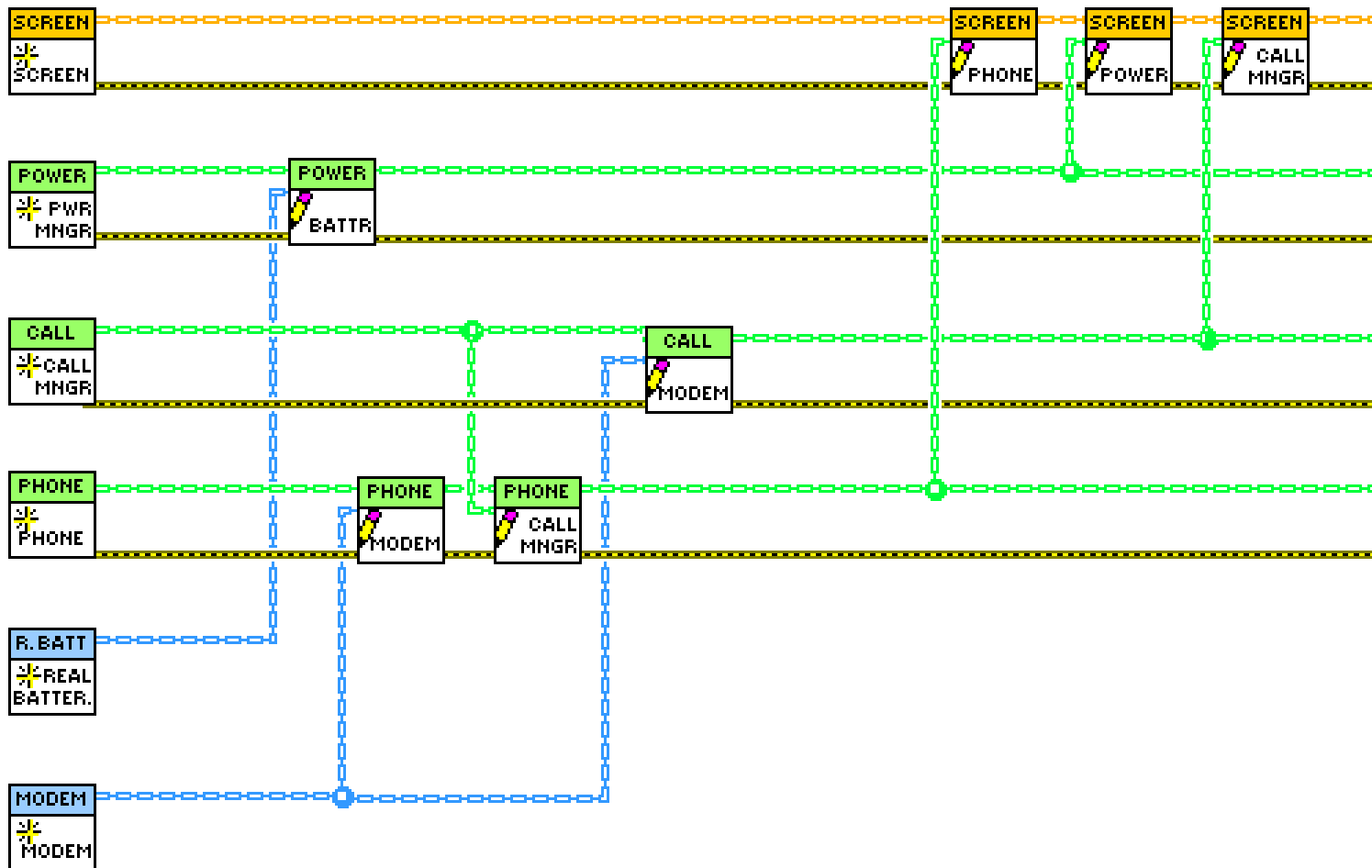


- Power Manager is coupled to the Real Battery
- 
- Automatic testing difficult

Dependency Injection

- Do not construct and fix dependencies inside the class
- Pass the dependencies to the object as arguments

Dependency Injection Demo





QUESTIONS?

Visit Us

NIWeek 2010: booth #841

Online: jki.net