

# Administrative Stuff

- ✦ Visio Professional on its way
  - ✦ One per team - install on a personal machine
- ✦ Books

# Contract Summary

- ✦ Overall, very good—some excellent
- ✦ “What” versus “how” (“how” now...)
- ✦ Diagram/section numbering schemes
  - ✦ Labels as persistent identifiers, tokens, or “icons” for an item
- ✦ Schedules
  - ✦ Milestones
  - ✦ Risk Analysis (Assessment)

# Schedule

## ★ Due on Tuesday

- Design Specification Document (w/ Schedule III)

## ★ Suggested schedule

### ● Before Thursday

- DIR: sketch out design w/ diagrams

### ● Thursday in place of class (no class)

- Team: hold a preliminary design review

### ● Friday-Sunday

- Team: work on Design Specification Document

### ● Monday

- Team: meet for final review of document

The background is a dark blue field filled with various shades of blue gears of different sizes, some overlapping. On the left side, there is a vertical strip with a colorful, textured pattern of gears in shades of orange, yellow, and brown.

# Design I

Design in general

The Design Specification Document



# Architectural Design Breakdown

*Note: this is tailored to COMP 145*

- ★ High-Level Design

- System structuring
- Control modeling

- ★ Detailed Design

- Modular decomposition

# Graphical Models (General)

- ✦ Structural model (static)
  - ✦ Subsystems and modules
- ✦ Run-time process model (dynamic)
  - ✦ Run-time process/thread organization
  - ✦ E.g., stand-alone server processes, spawned processes, etc.
- ✦ Interface model
  - ✦ Public interface specification



# High-Level Design

System structuring  
Control modeling



# System Structuring

- ☀ Overall static architecture

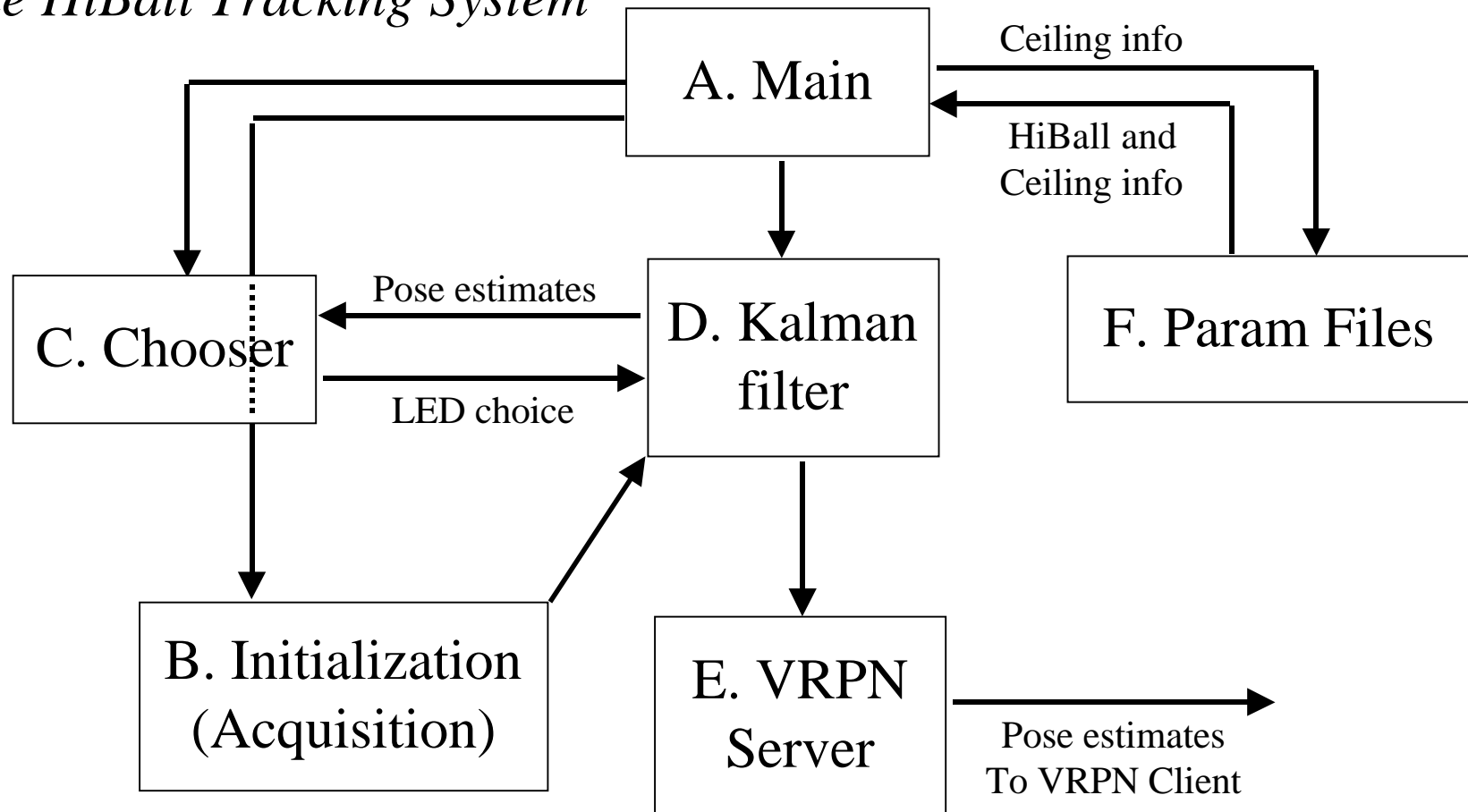
- Provide context—show all of the pieces
- (~User-Level Requirements)

- ☀ Decompose **system** into set of interacting **subsystems**

- How are the subsystems distributed?
- How do they manage/share data?
- High-level interfaces

# Structural Model (Static)

*The HiBall Tracking System*



# Control Models (Typical)

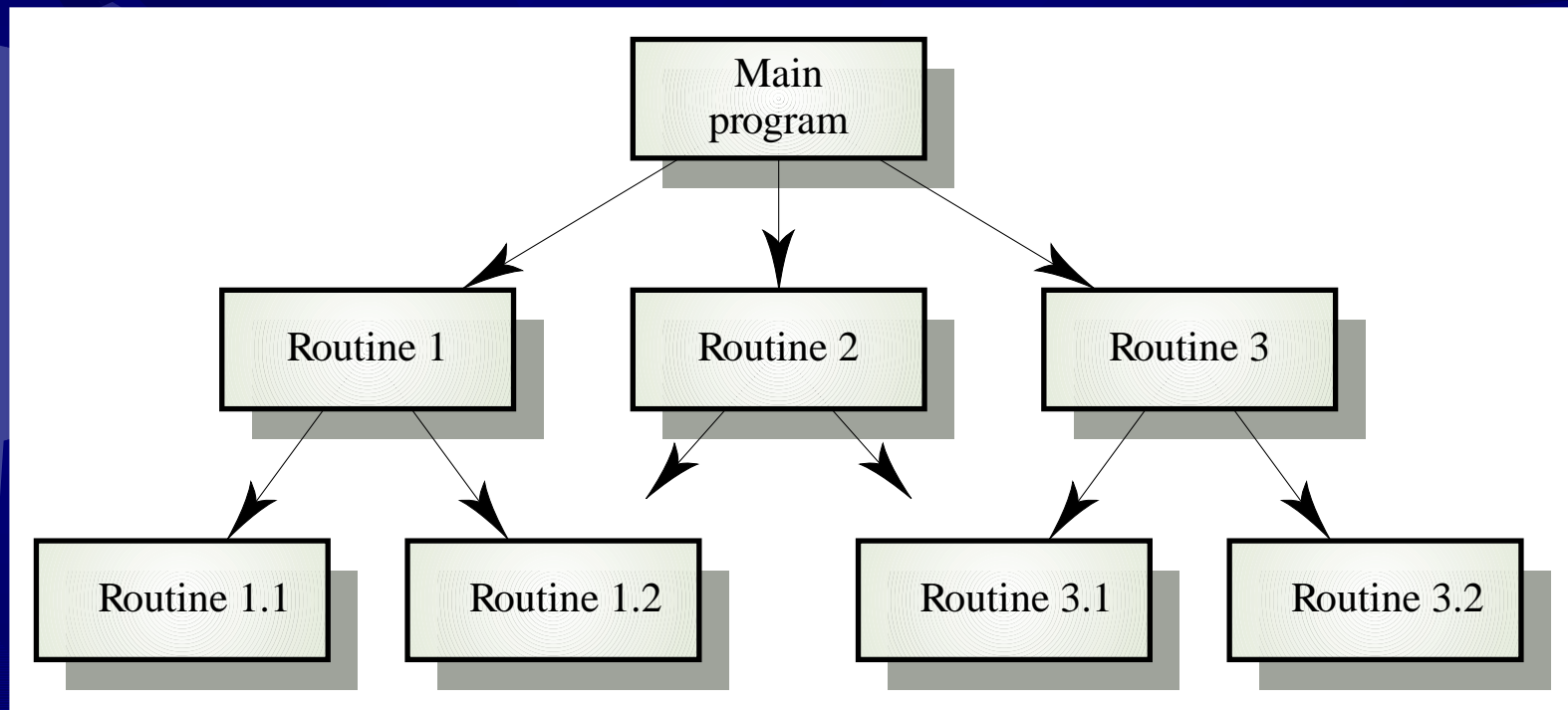
## ☀ Centralized

- One subsystem is the master controller
- **Call-return** (sequential) vs. **manager** (concurrent)

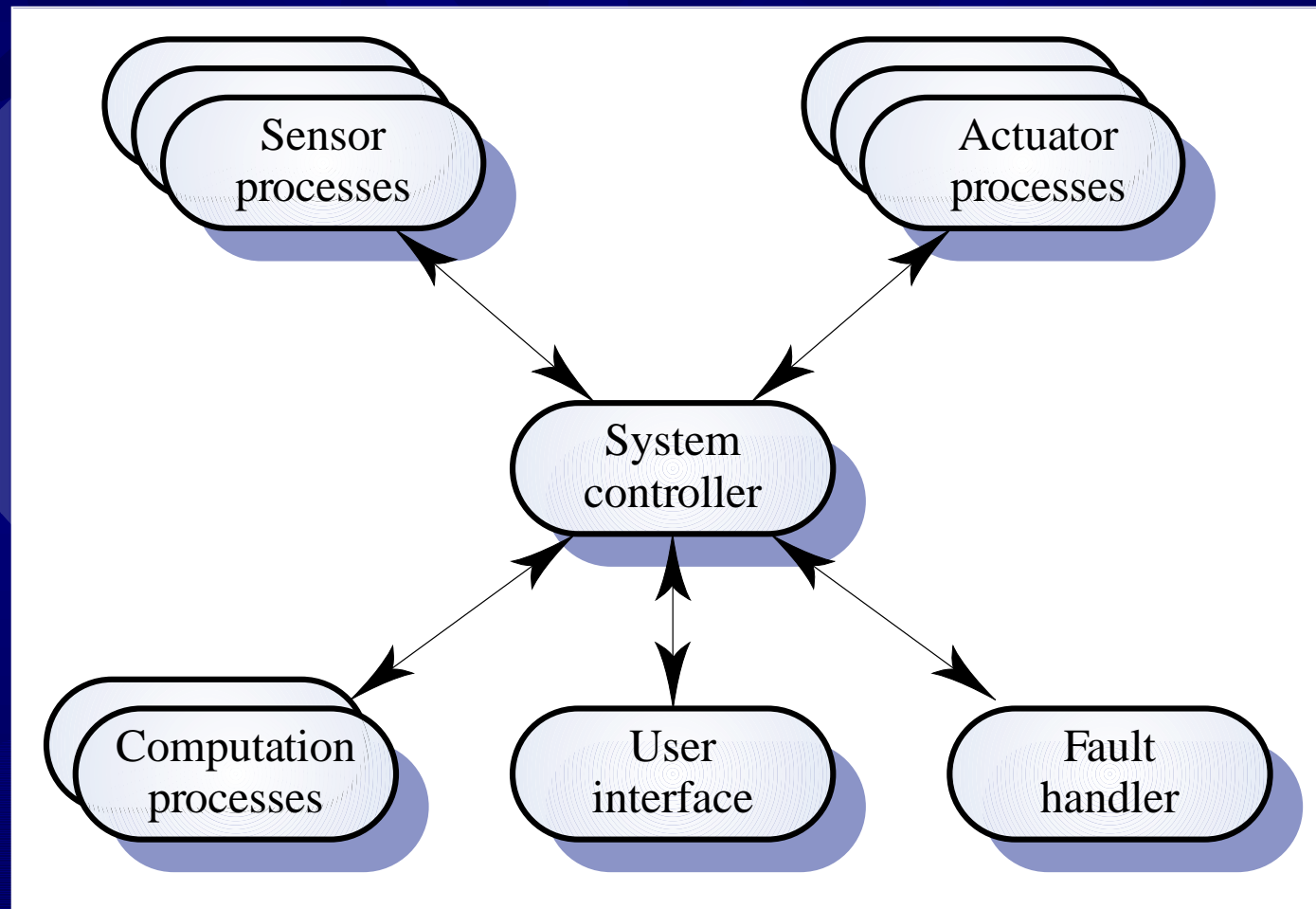
## ☀ Event-driven

- Subsystems respond to external events or triggers
- **Broadcast** (message) vs. **interrupt-driven**

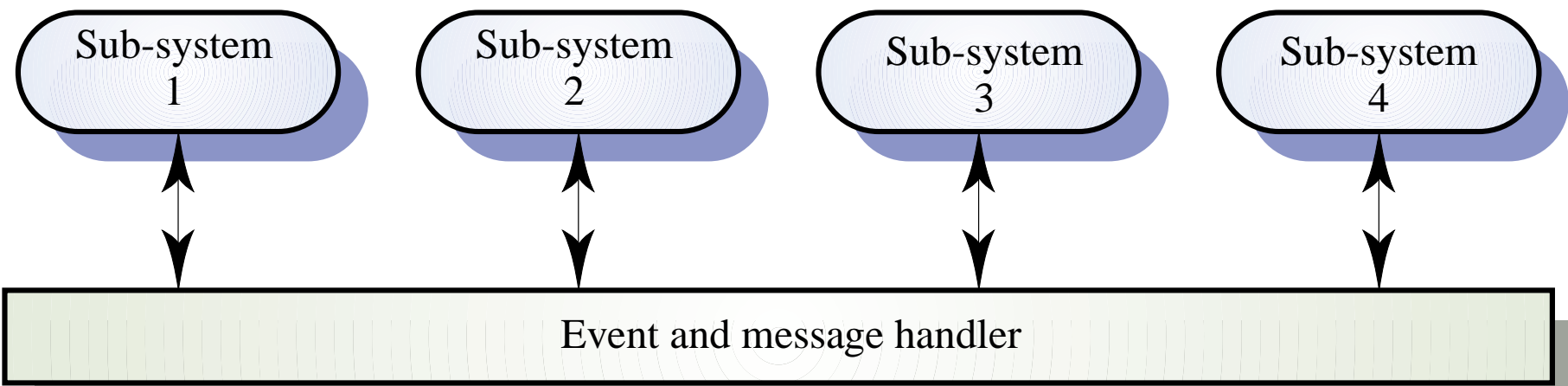
# Centralized Control: Call-Return



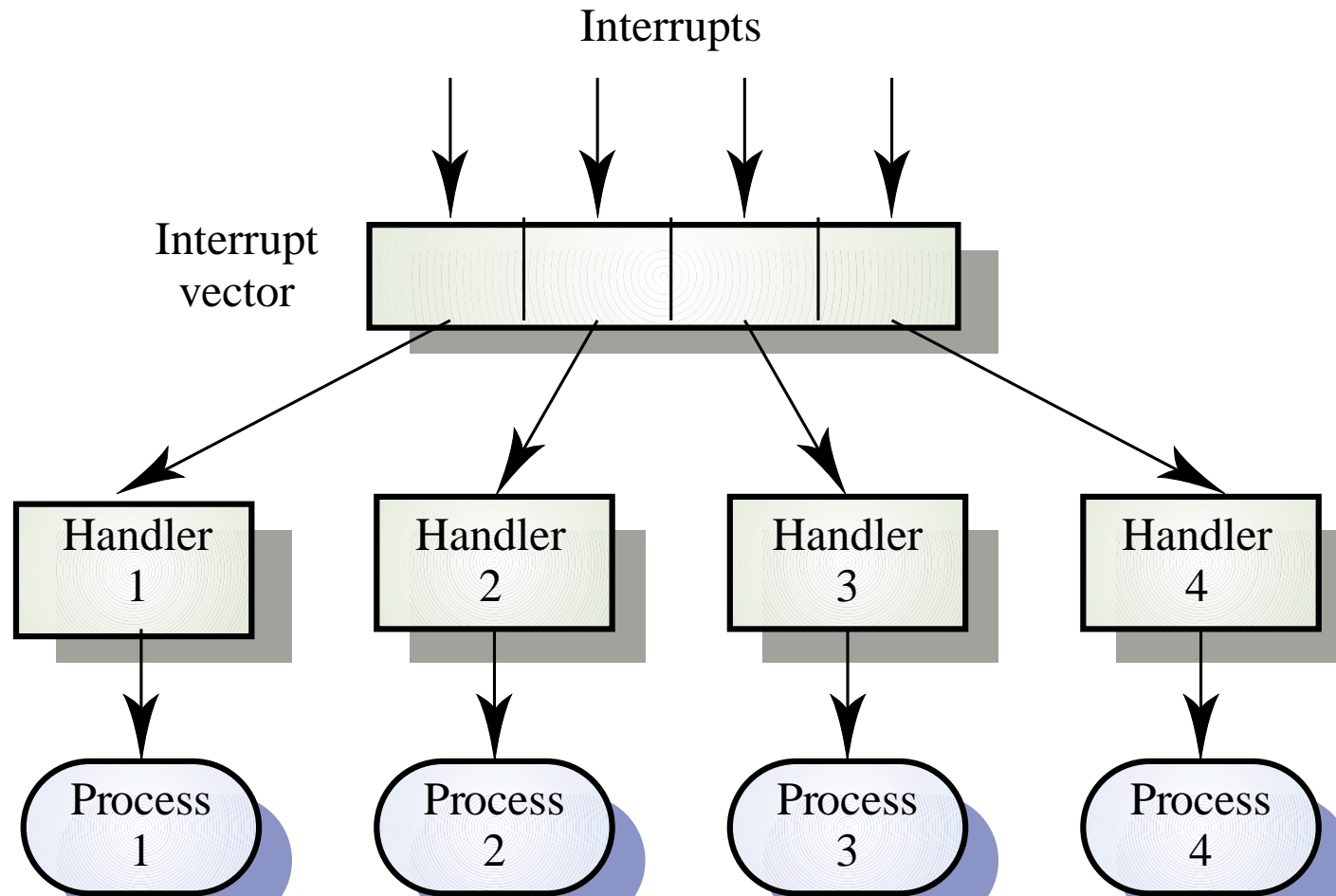
# Centralized Control: Manager



# Event-Driven Control: Broadcast



# Event-Driven Control: Interrupts





# Detailed Design

Modular decomposition  
(Object-Oriented)

# Modular Decomposition (DD)

- ✦ No hard/fast rules about granularity
  - ✦ Lowest levels would be modules
  - ✦ Top level is the system
  - ✦ In between are subsystems
  - ✦ All diagrams could apply at all levels
- ✦ For us
  - ✦ About same level as Contracts
  - ✦ Suggest OO decomposition/design
  - ✦ Read Coggins/Brooks Chapter 7

# Object Models (OO-Centric)

- ✦ Encapsulation and inheritance

- ✦ Useful object-oriented traits

- ✦ Class  $\leftrightarrow$  module

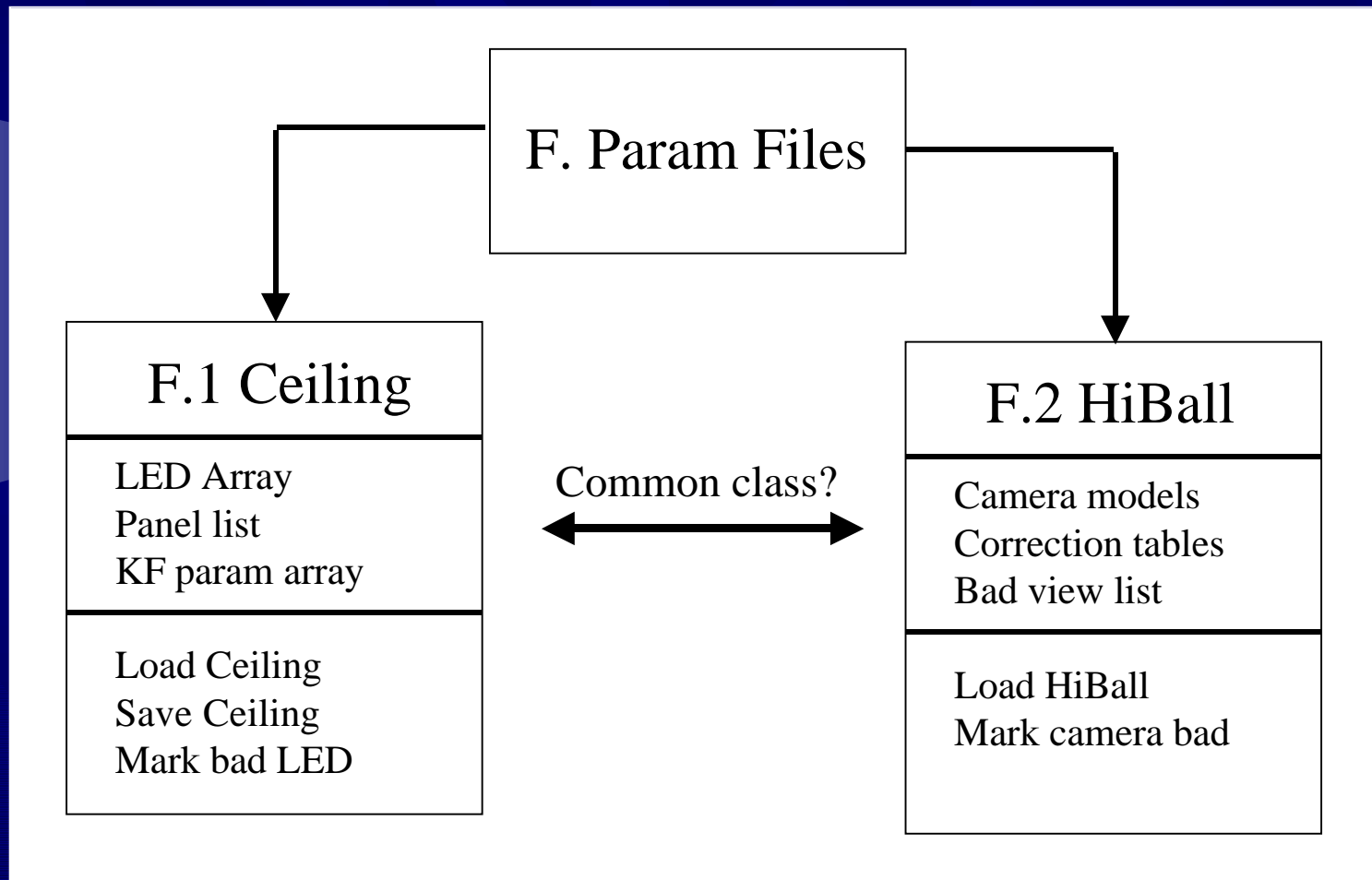
- ✦ Task decomposition

- ✦ Minimize communication and dependencies among modules

- ✦ Separate architecture from implementation

- ✦ Data and process hiding

# Object Model Diagrams



## Other DD

- ☀ Intra-module data-flow diagrams

- Same as with Contract, but lower level and specific to objects/classes

- ☀ Inter-module control diagrams

# Design Specification Document

- ★ Introduction
- ★ High-Level Design
  - Structure and control diagrams
- ★ Detailed Design
  - Object model diagrams (equivalent)
- ★ Requirements Traceability
  - Matrix w/ requirements vs. design
- ★ Schedule
  - New and improved milestones
- ★ (See example document on-line)