4. Architecture

Artifact: Analysis Class

- Abstraction of one or several classes or subsystems
  - Focuses on handling functional requirements
  - Postpones non-functional requirements
  - Larger granularity and more conceptual
  - Seldom defines interfaces - operations, signature
  - High level description of cohesive subset of behavior
  - Defines attributes - higher level
  - Defines relationships - conceptual - navigation not important
  - Fits one of the stereotypes: `boundary`, `control`, `entity`
Boundary Classes

- Interaction between system and its actors
- Receiving / presenting information and requests
  - Translates actor’s input into events in system
  - Translates interesting events into presentable form
- Models part that depends on the actors
- Collect requirements on system’s boundaries
- Windows, forms, UI, printer, sensor, APIs
- Fairly kept at higher level
- Describes what it achieves not how
- Related to at least one actor and vice versa
  - actors communicate with system through these objects

Entity Classes

- Models information that is long lived
- Often derived directly from business entity class
- May involve complex behavior related to information it represents
- Presents the logical data structure
- Tells what information the system depends upon
Control Classes

- Sequencing, coordination of other objects
  - Behavior that is not naturally placed in either Entity or Boundary classes
  - These behavior belong neither to interface nor to specific information

- May represent complex calculations not specific to any one entity class

- Encapsulate and isolate changes to control, sequencing, transactions, and calculations

Architecture

- System must be layered
  - Each layer taking care of different concerns

- N-tier architecture contains n such layers

- Typically you should at least follow a 3-tier architecture
3-Tier Architecture

Presentation Tier <-> Business Tier <-> Entity Tier