Modeling Structure with Blocks - Block Definition Diagrams (Part 1 – SysML Concepts)

Content Developer

OBJECT MANAGEMENT GROUP • SM • CERTIFICATION PROGRAM

Section Objectives

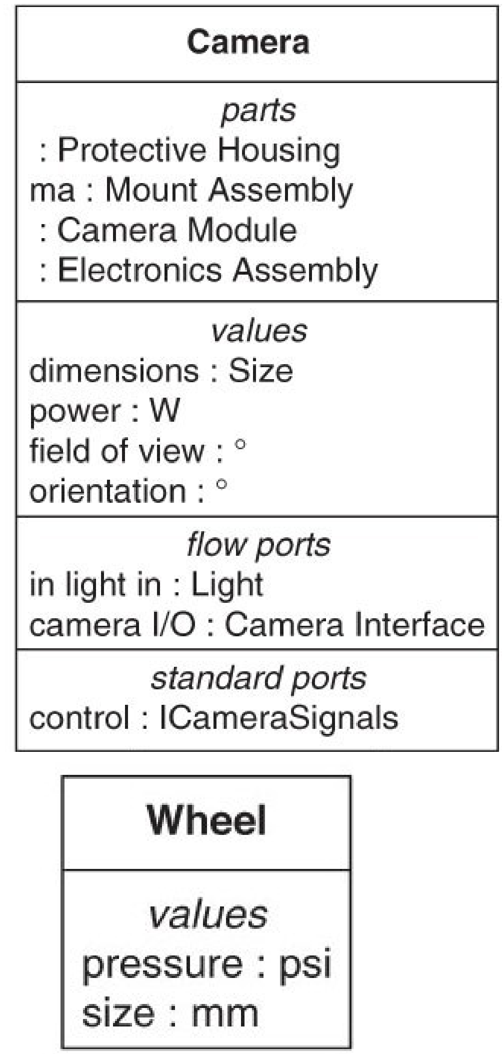
- In this Section, you will learn:
  - How to model Block Definition Diagrams in SysML

Overview

- This section will discuss:
  - Block Concepts
    - Defining Blocks
    - Why model Blocks?
    - Purpose of Block Definition Diagrams
      - Depicting Relationships between Blocks
      - Depicting and Defining Block Characteristics
    - How to model Blocks
  - Modeling Blocks for In-Class Project

Defining Blocks

- Blocks are the basic structural element used to model the structure of systems
- Blocks can be used to represent:
  - Systems
  - System components (Hardware and Software)
  - Items that flow
  - Conceptual entities and logical abstractions
- Blocks are depicted as a rectangle with compartments that contain the block characteristics, such as:
  - Name (mandatory)
  - Properties (e.g. parts, values, ports)
  - Operations
  - Requirements that the block satisfies
  - etc



Why Model Blocks?

- Used to define the domain model
  - Defines the glossary for the ‘things’ in the problem space
  - Graphically depicts how the ‘things’ relate to each other
- Said in Systems Engineering terms:
- Depicts the static structure of a system
  - What the system consists of
  - How those components are related
- Part of the Physical Definition phase of Systems Engineering Method
- Clarification, Elaboration, Communication
  - Communication with Users, Domain Experts, Stakeholders
  - Consistency in terminology among team members

Purpose of Block Definition Diagrams

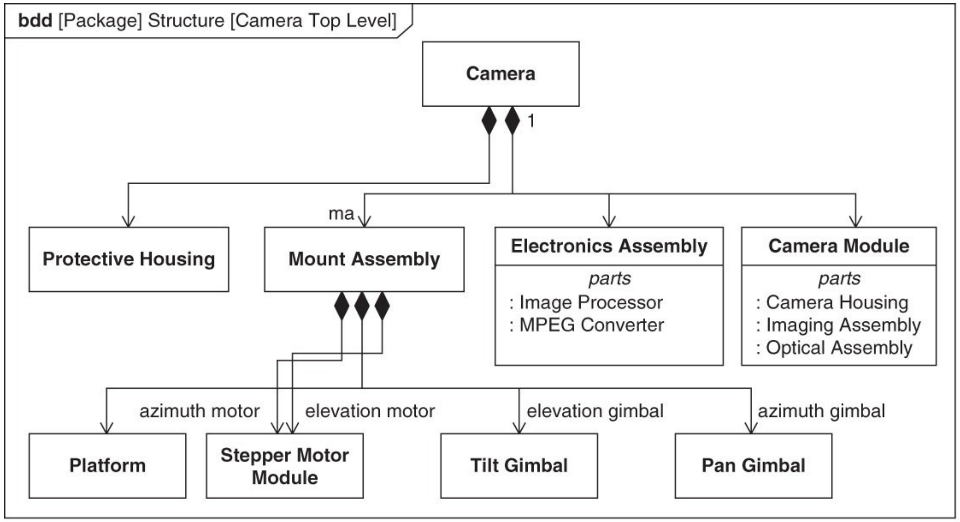
- Depicting Relationships between Blocks
  - Associations
  - Generalizations
- Depicting Block Characteristics
  - Structural Characteristics
    - Part Properties
    - Value Properties
    - Flow Ports
      - Atomic Ports
      - Non-atomic Ports and Flow Specifications
  - Behavioral Characteristics
    - Operations
    - Receptions
    - Interfaces

Depicting Relationships between Blocks

- Associations
  - Part Associations
  - Shared Associations
  - Reference Associations
- Generalizations
- Example: Top Level Block Diagrams

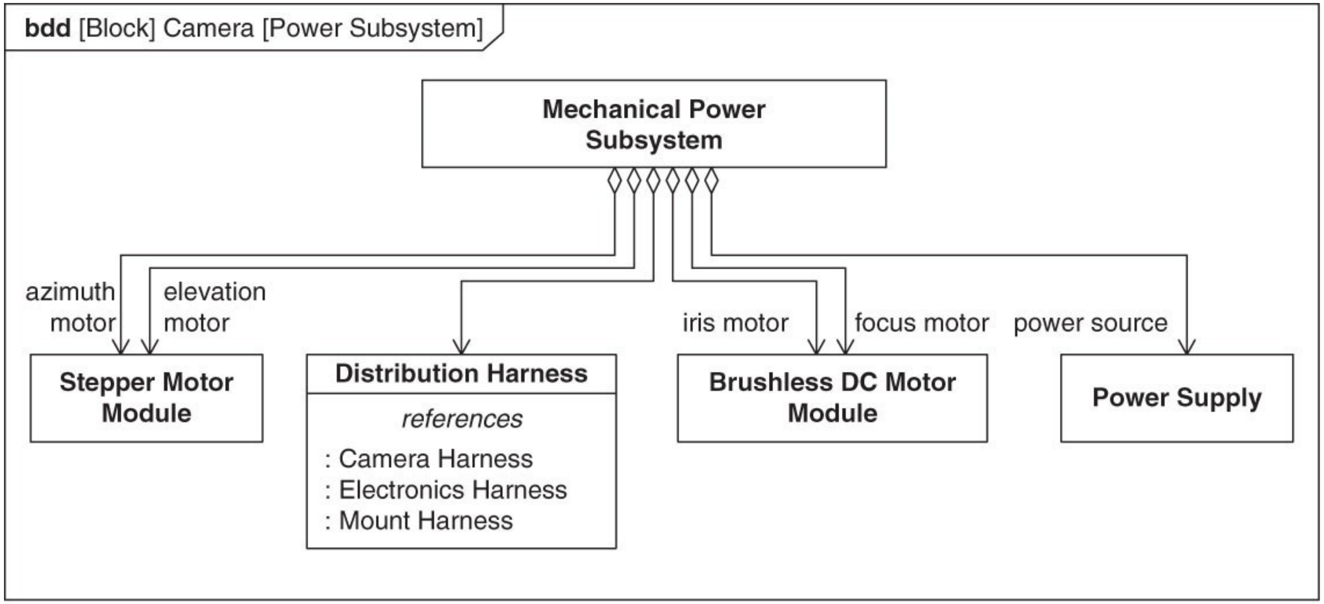
Part Associations

- Block composition can be depicted using Part Associations
- Represents the Parts that make up the Whole
  - Depicted with a black diamond on the Whole end
- Multiplicity on the Whole end:
  - Lower bound may be 0 or 1:
    - 0 means the Part can exist without the Whole
    - 1 means the Part always exists within the Whole
  - Upper bound is always 1
    - An instance of a Part may exist in only one instance of a Whole at a time
    - Depicts 'ownership'
  - Default is [0..1]
- Role names can appear on the part end of the association



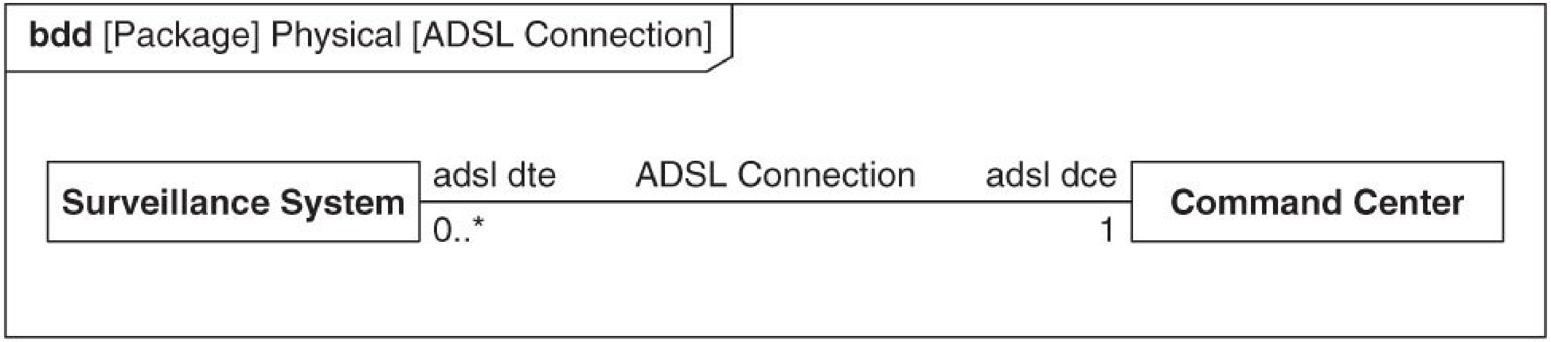
Shared Associations

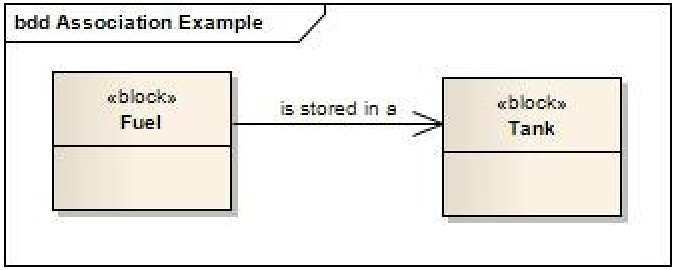
- Can be used to depict an aggregation of components into a logical subsystem
- Associated blocks are not ‘owned’ by the Whole
- Depicted with white diamond



Reference Associations

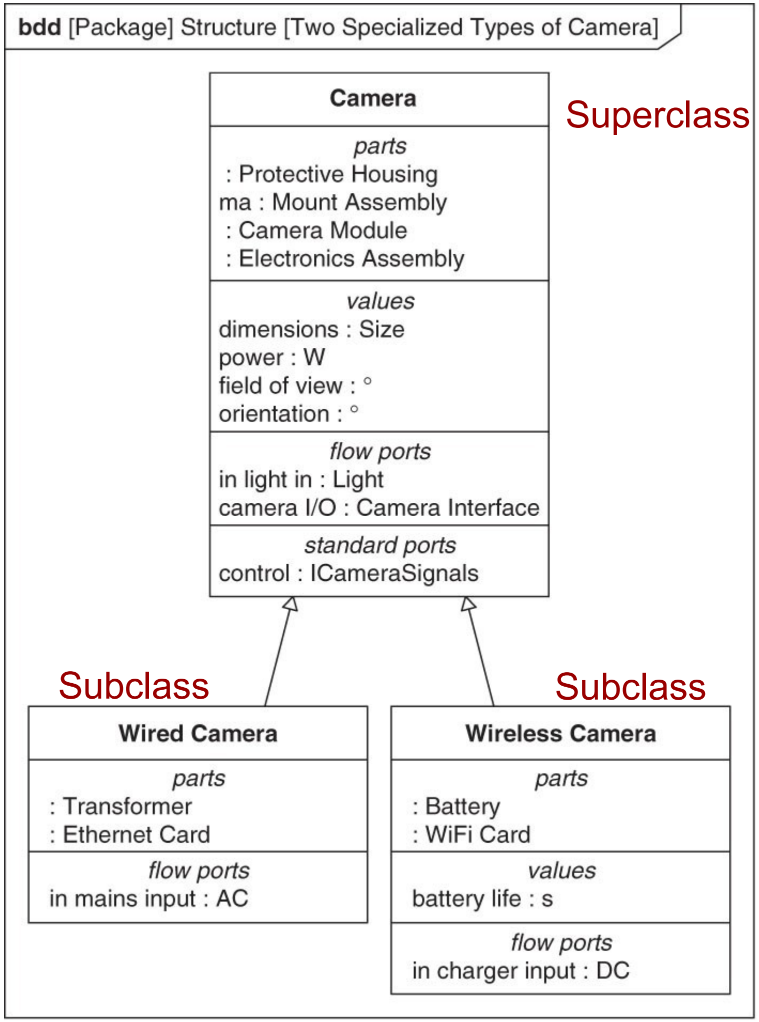
- Reference Associations can be used to specify a relationship between two blocks
- Can be used to depict a connection
- Can also be used to depict other relationships that exist between blocks





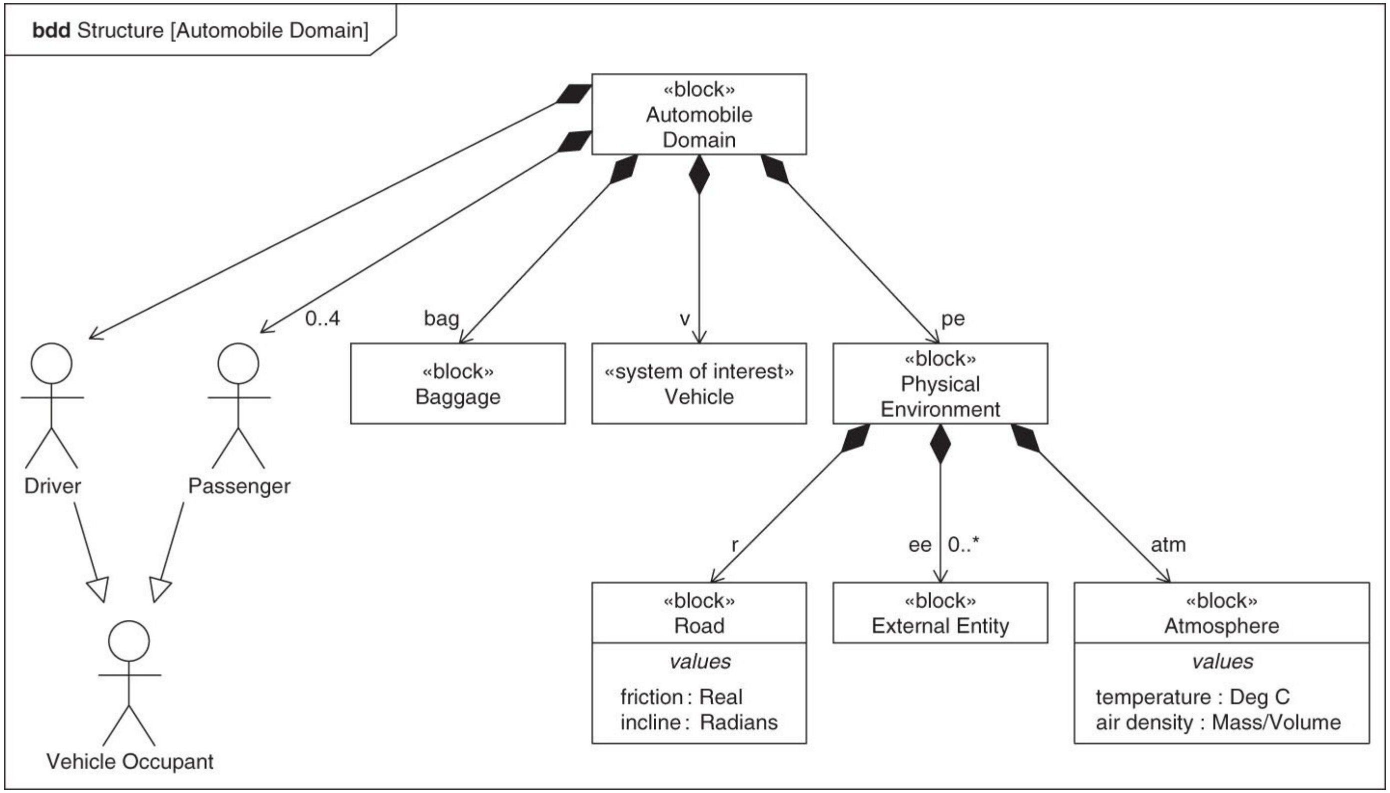
Generalizations

- Block Definition Diagrams can be used to depict generalization and specialization relationships
- Facilitates reuse
  - The specialized block (subclass) reuses (inherits) the features of a generalized block (superclass), and adds its own features
- Depicts an 'is-a' relationship
- Depicted with a closed arrowhead pointing toward the generalized block



Example: Top Level Block Definition Diagrams

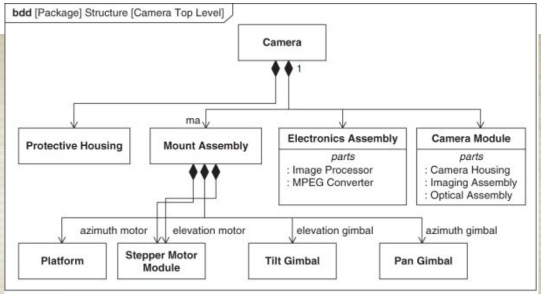
- Can be used to depict the problem domain and model scope
  - System of interest and the external systems that interact with it



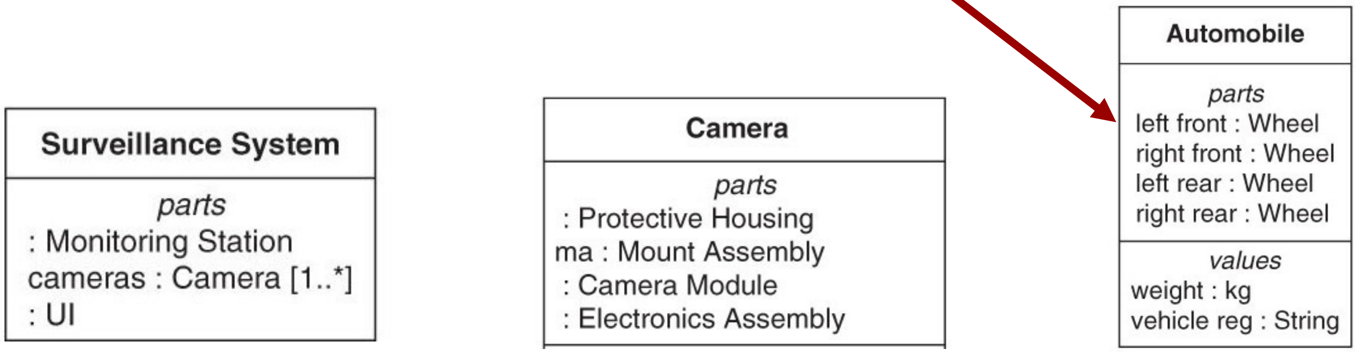
Structural Characteristics of Blocks

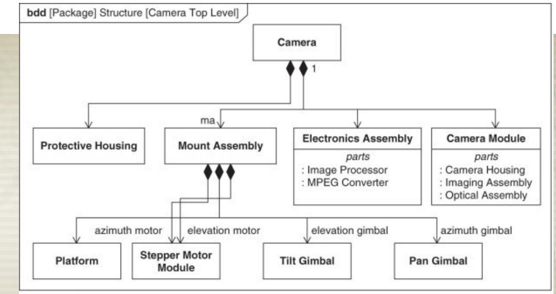
- Part Properties
  - Multiplicities
- Value Properties
  - Value Types, Dimensions, and Units
- Flow Ports
  - Atomic Ports
  - Non-atomic Ports and Flow Specifications

Part Properties



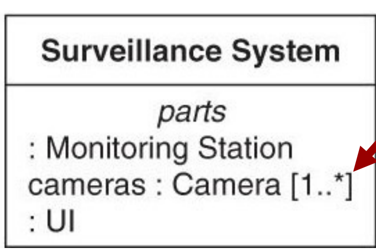
- Parts compartments list the Part blocks that make up the Whole block
- Same as Part Association relationship between blocks
- Parts are listed in the *parts* compartment in the following format:
  - part name: block name [multiplicity]
- Part Names are typically used to specify a "role"
  - Example below: "left front : Wheel"
    - "left front" is the "role" or part name
    - "Wheel" is the block name

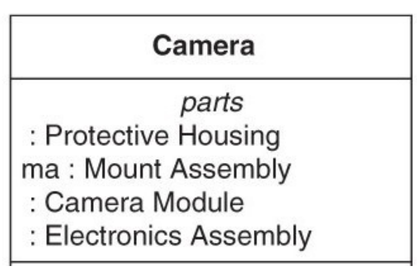


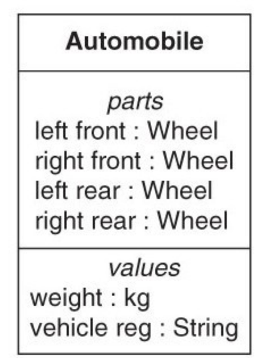


Multiplicities

- Multiplicity specifies the potential number of Parts that the Whole may include
  - Depicted as "lower bound..upper bound", where:
    - Lower bound is the minimum number of Parts that make up the Whole
      - 0 means the part is optional
    - Upper bound is the maximum number of Parts that make up the Whole
      - * means "many"
    - Default is [1..1], which means "exactly one"
    - Example below: "cameras : Camera[1…*]"
      - Denotes that the Surveillance System consists of 1 to many cameras

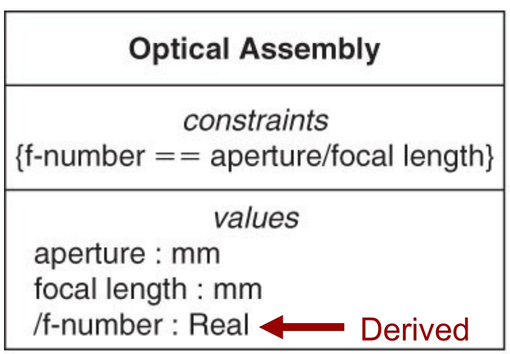


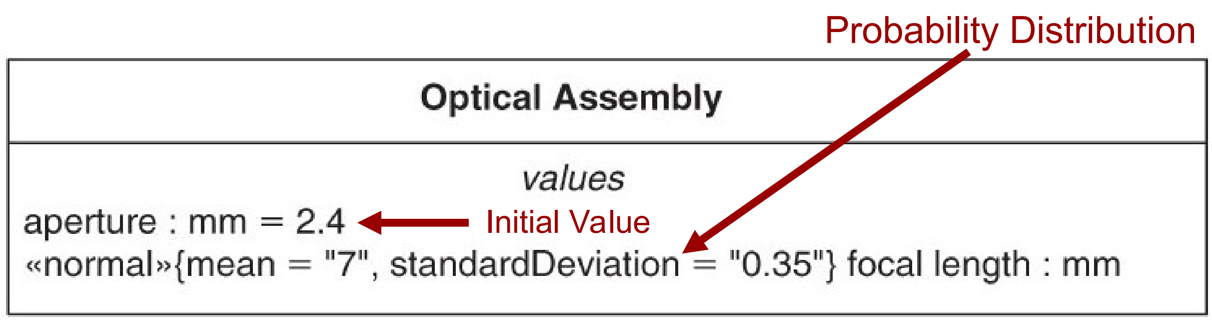




Value Properties

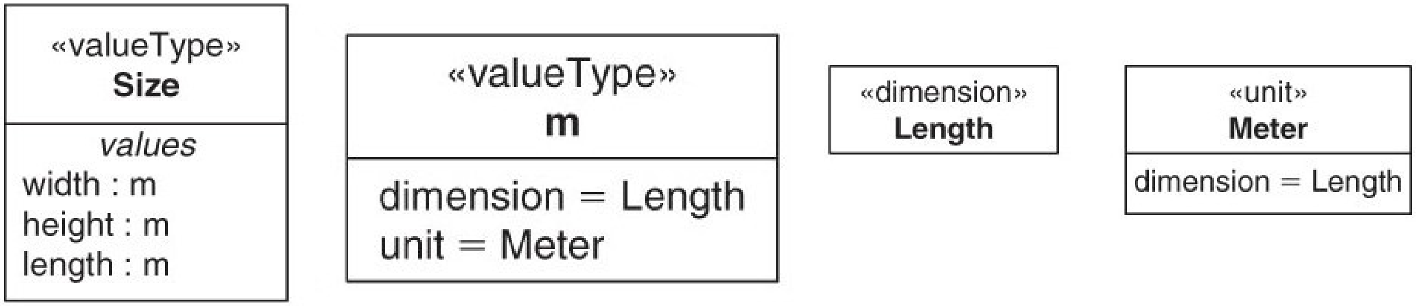
- Used to model quantifiable block characteristics or attributes
- Based on a Value Type, which describe the values for quantities
- Listed in compartments using the following syntax:
  - value property name: value type name
- Value Properties:
  - can be derived (depicted with a '/')
  - can have initial values
  - can also define a probability distribution for their values





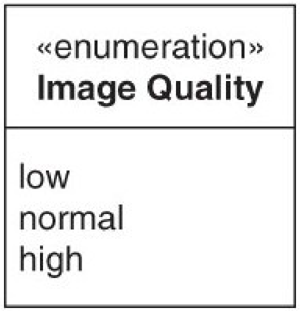
Value Types, Dimensions, and Units

- Value Type is a form of data type (e.g. integer, real) with units (e.g. m = meter)
- Value Types may include a dimension and a unit
  - Dimension – identifies a physical quantity (e.g. length)
  - Unit – identifies the unit of measure (e.g. meter)
- Value Type is used to type a value property



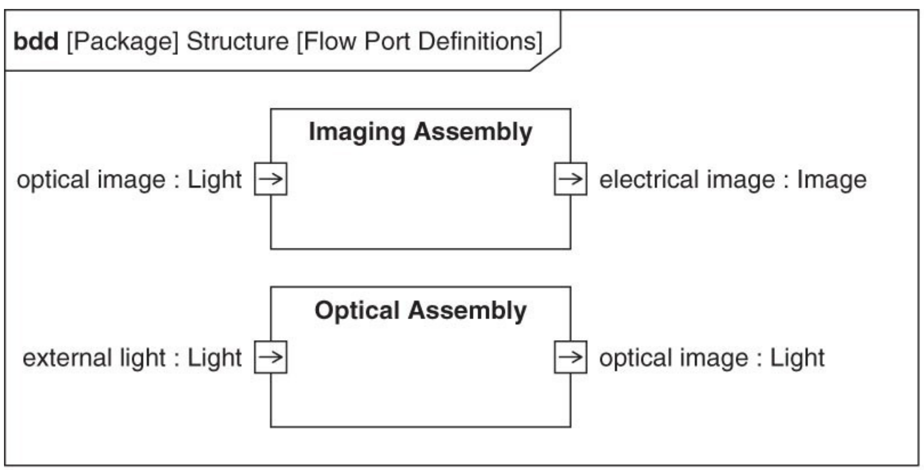
Enumerations

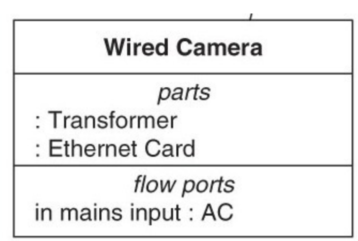
- A Value Type whose values are a set of literals (e.g. low, normal, high)



Flow Ports

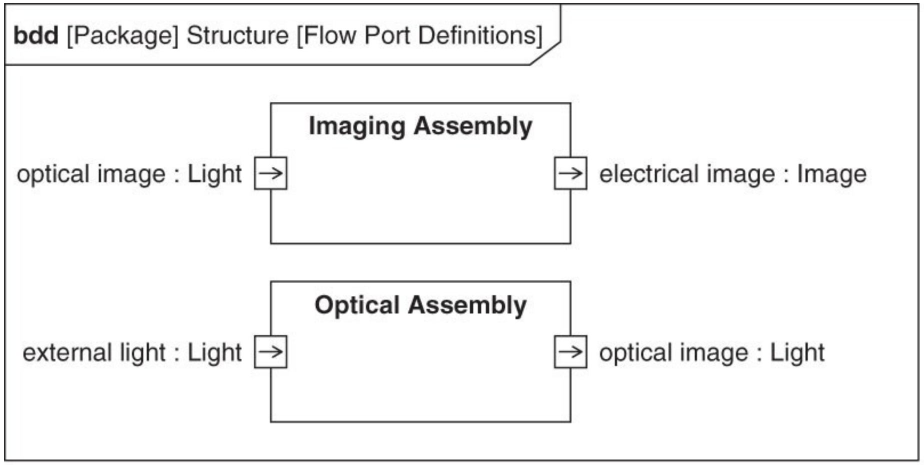
- Flow Ports – used to describe an interaction point for items flowing in or out of a block
- Two types:
  - Atomic Ports
  - Non-atomic Ports
- Can be depicted as a box on the block border or in a block compartment





Atomic Ports

- Atomic Flow Ports – specify a single type of input or output flow
- Flow direction can be in, out, or inout
- Described as follows:
  - port name: item name[multiplicity], where:
    - Item name is the name of the item flowing in or out of the block
    - Multiplicity is the number of instances of the port on an instance of the block

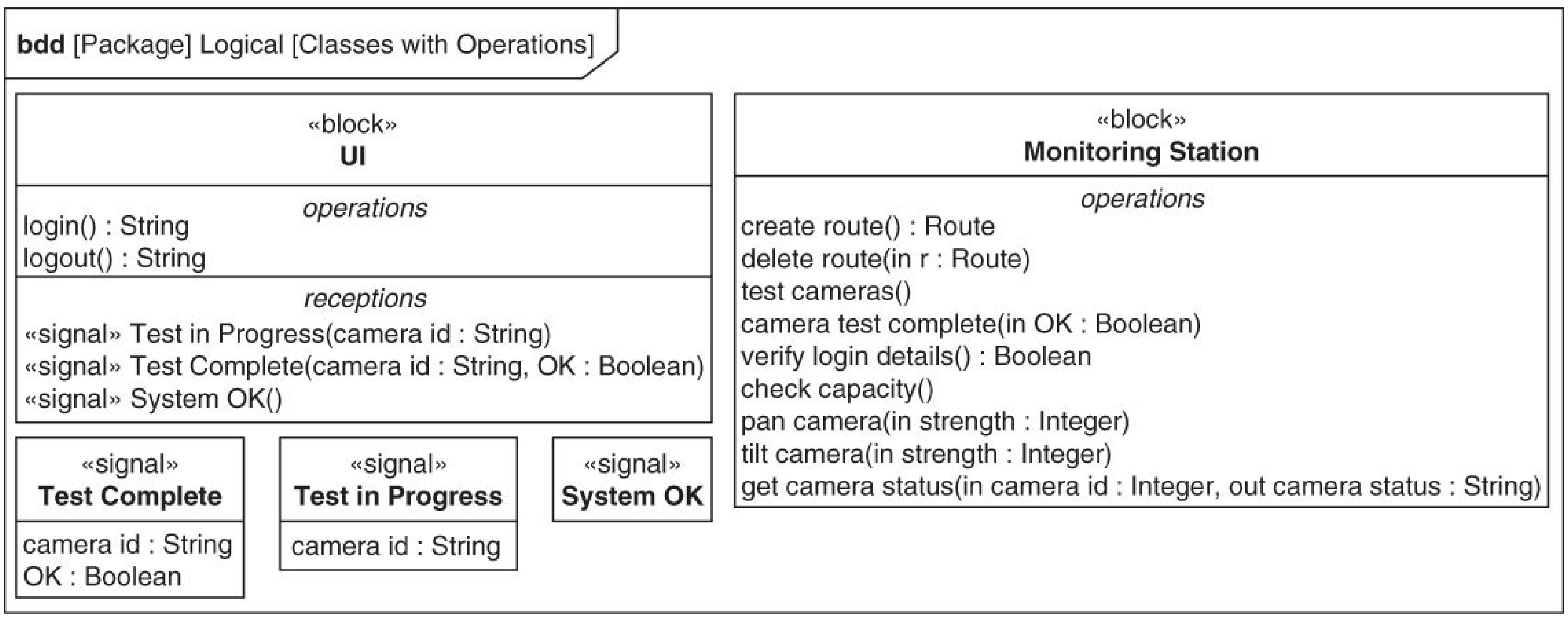


Behavioral Characteristics of Blocks

- Operations
- Receptions
- Interfaces

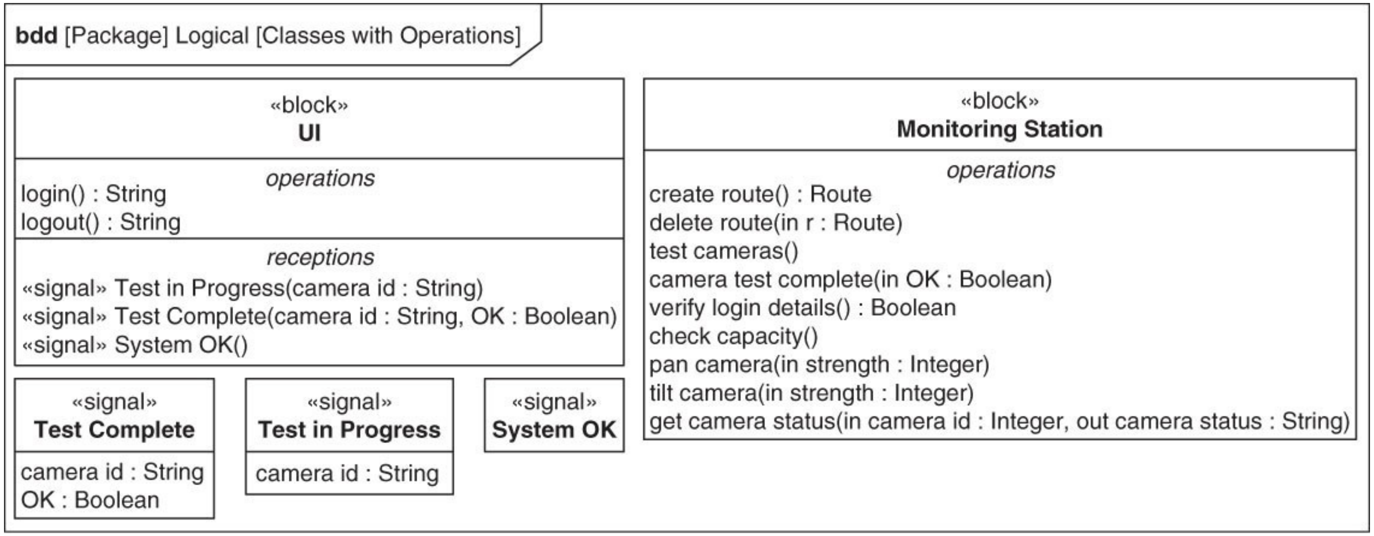
Operations

- Operations describe something that a block can do
- Operations can have parameters that are passed into or out of the operation
- Operations are typically synchronous, (i.e. requestor waits for a response)
- Operations are listed in the 'operations' compartment of a block, as follows:
  - operation name (parameter list): return type



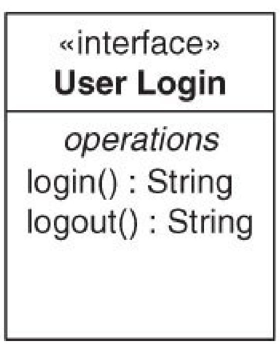
Receptions

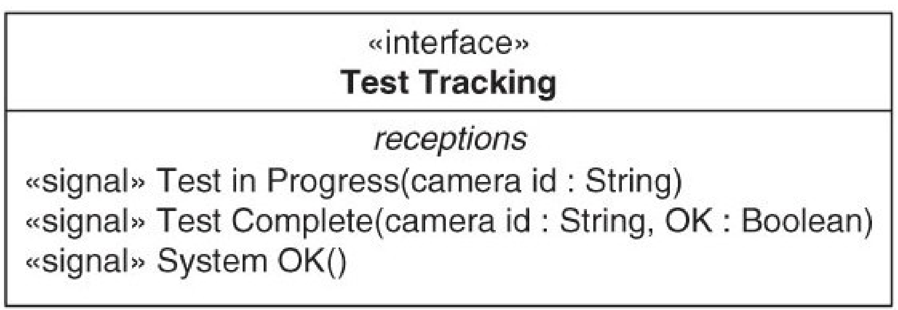
- Receptions are asynchronous (i.e. the requestor does not wait for a response)
- Each reception is associated with a signal
  - Example: TV receives a signal from a remote
- Signals define a message with attributes that represent the content of the message
- Receptions are listed in the 'receptions' compartment of a block, as follows:
  - <<signal>> reception name (attribute list)

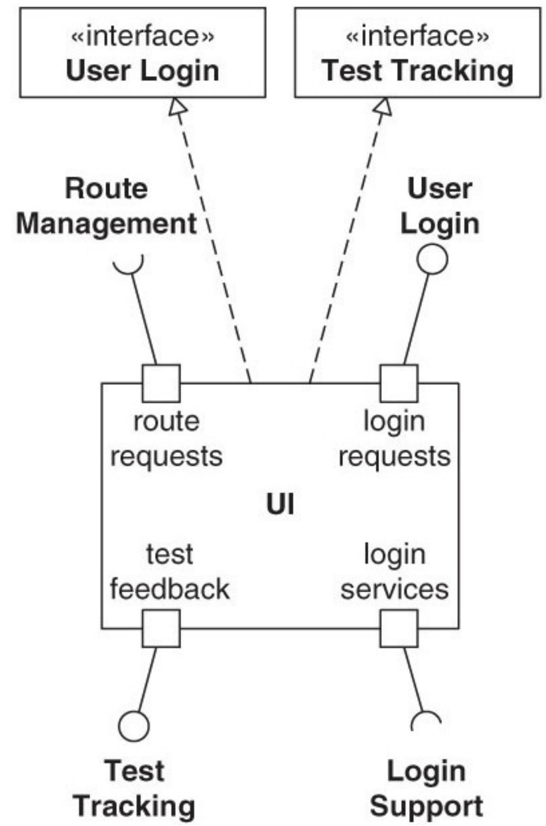


Interfaces (and Standard Ports)

- Standard Ports – depict interfaces that specify the behavioral features (services) that a block either provides or requires
- Interface symbols have operation and reception compartments like block symbols
- Provided Interface – specifies operations that a block provides
  - Depicted by a 'ball' or a realization dependency
- Required Interface – specifies operations required by the block
  - Depicted by a 'socket' or a uses dependency (not shown)







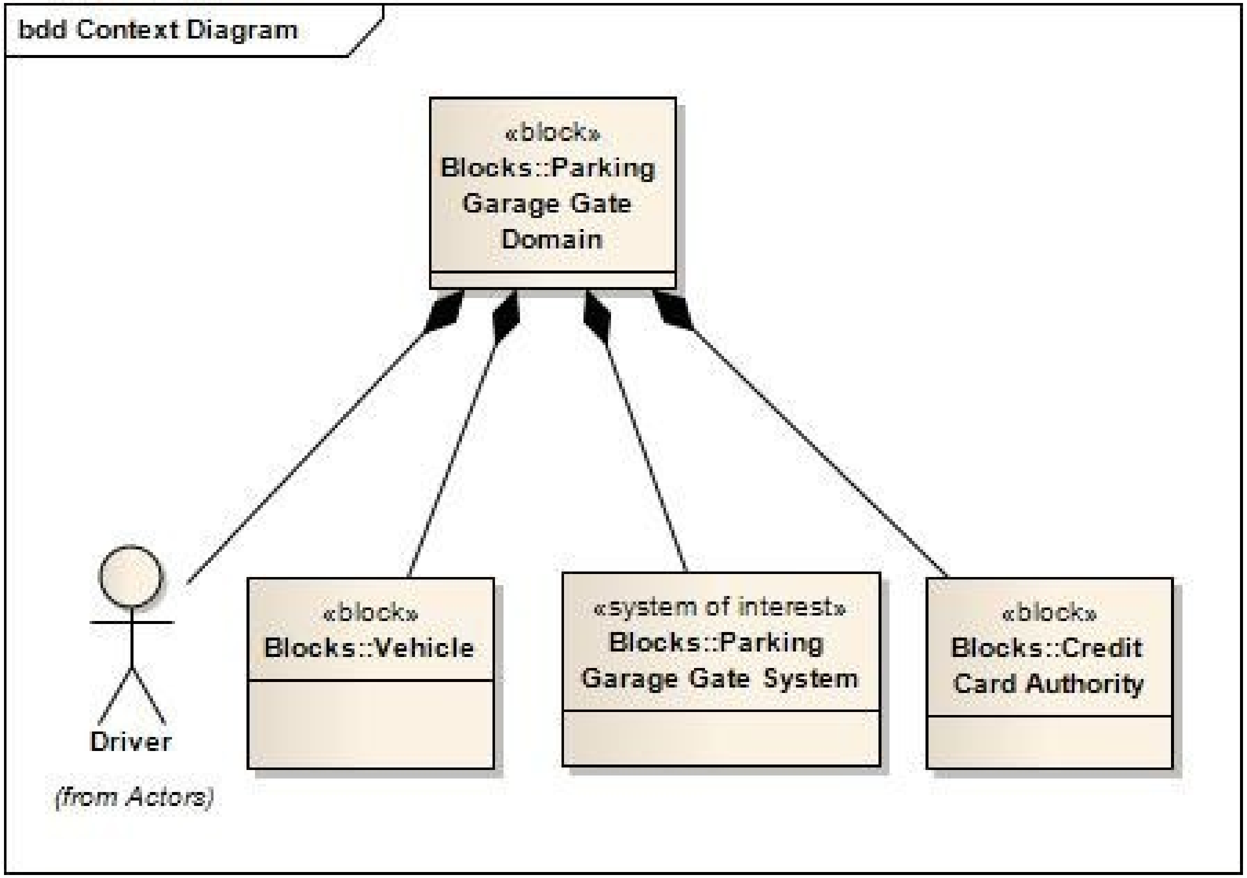
How to Model Blocks

- Define the 'real-world' blocks within the problem space
  - What are the 'nouns' of your system?
- Depict Block relationships
  - How are the blocks related?
    - Part (ownership associations)
    - Reference ('shared' or some other relationship between blocks)
    - Generalization ('is-a')
- Identify the multiplicity of the relationships
  - How many blocks of one type are related to one block of another?
- Identify the Value Properties for each block
  - What are the quantifiable attributes of each block?
- Identify the Flow Ports for each block
  - What flows in or out of the block?
- Identify the Behaviors for each block
  - What functions does each block perform?
- Iterate, as required

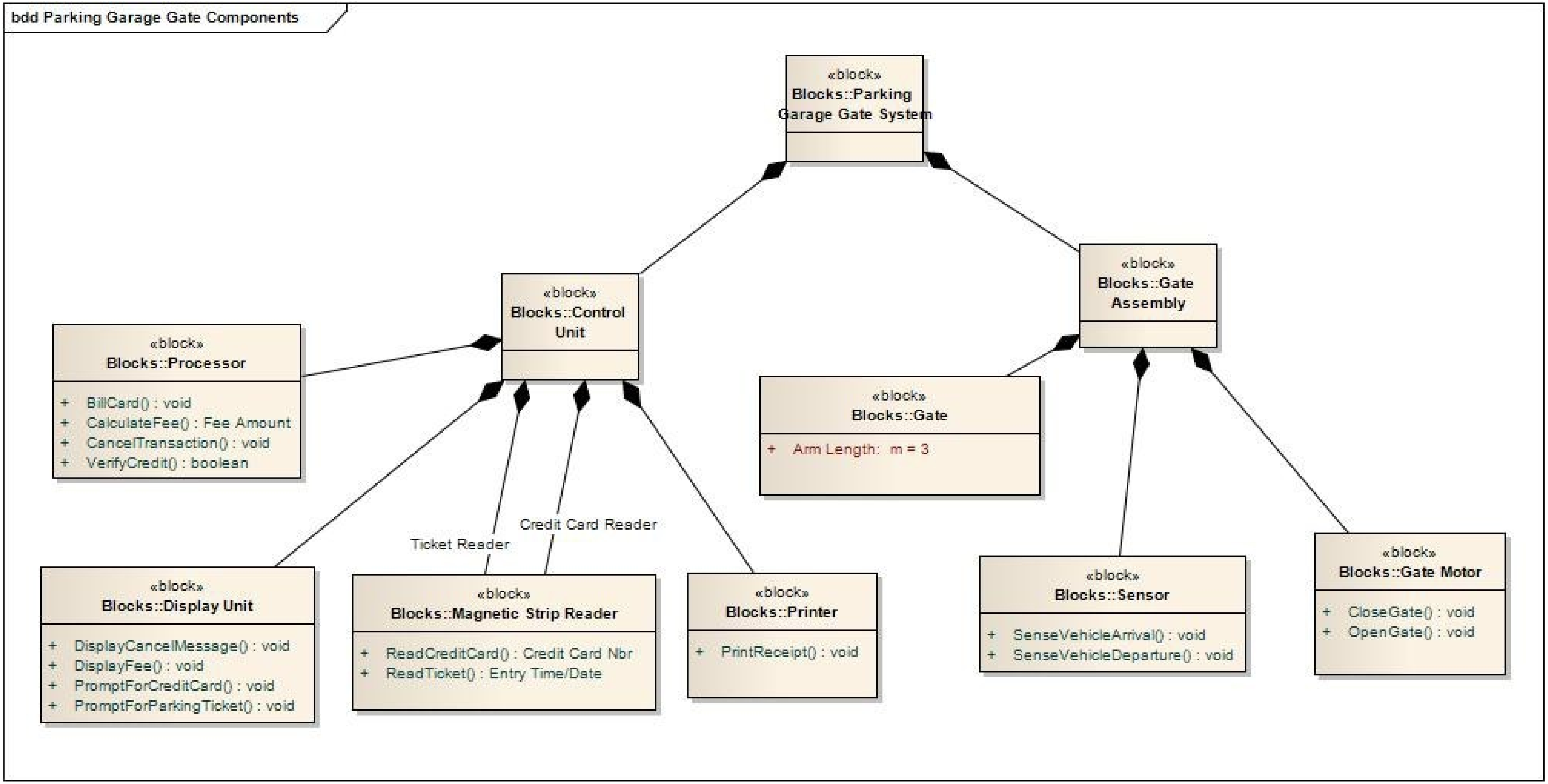
Modeling Structure for In-Class Project

- Create a Top Level BDD Diagram in EA for the Parking Garage Gate Domain
- Create a Block Definition Diagram in EA for the Parking Garage Gate System
  - Depict Part associations
  - Include Operations for each Block
  - Show a Generalization relationship

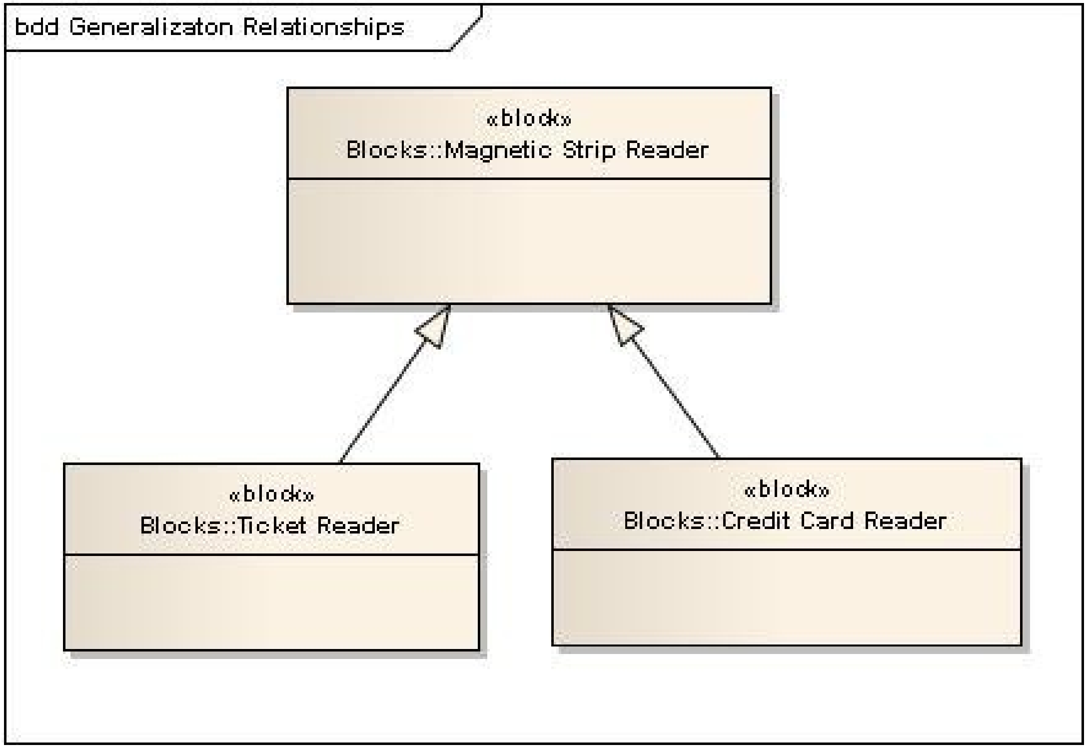
Top Level Block Definition Diagram for Parking Garage Gate Domain



Block Definition Diagram for Gate System



Generalization/Specialization Relationship



Summary

- Block Definition Diagrams are used to depict the static structure of a system
  - What the system consists of
  - How those components are related to each other
- A Block is the basic structural element used to model the system's structure
- Block Definition Diagrams can depict Block structural characteristics, functional characteristics, and relationships
- Block structural characteristics include: part properties, value properties, and flow ports
- Block functional characteristics include: operations, receptions, and interfaces
- Block relationships include: associations and generalizations