UML Class Diagrams

Josh Wilcox (jw14g24@soton.ac.uk)

February 18, 2025

Contents

1 Class Diagrams

1.1 Format

1.1.1 Class Name

- Literally just the name of the class

1.1.2 Attributes

- Visibility is included in front of each attribute of the class
  - private → -
    - Inaccessible from outside its class
  - public → +
    - Accessible from outside the class
  - protected → #
    - Only itself and classes that inherit from it
- Attribute syntax
  - `visibility attribute_name[<multiplicity> <ordering>] : type = initial value`
  - **Type**: Boolean, Integer, Real, String, etc
  - **Ordering**: Ordered/Unordered
  - **Multiplicity**: Integer intervals `[lower bound .. upper bound]`
    - Specifies the range of allowable instances for the attribute.
    - **Examples**:
      - `[1..1]`: Exactly one instance (mandatory).
      - `[0..1]`: Zero or one instance (optional).
      - `[0..*]`: Zero or more instances (unbounded).
      - `[1..*]`: One or more instances (at least one).

1.1.3 Operations

- **Operation Syntax**
  - `visibility operation_name(<list of parameters>): return_type`
- **Parameter Syntax**
  - `kind name : type = default_value`
  - **Kind**:
    - **in** - Input parameter
    - **out** - Output parameter
    - **inout** - Both input and output

1.2 Associations

1.2.1 Binary Associations

- Represents a relationship between two classes.
- Typically depicted as a line connecting the two classes.
- Can have multiplicity at each end to indicate the number of instances involved.

- Example: A `Student` class associated with a `Course` class.

1.2.2 N-ary Associations

- Represents a relationship between more than two classes.
- Depicted as a diamond shape connected to the involved classes.
- Example: A `Project` class associated with multiple `Employee` classes.

1.2.3 Composition

- Represents a strong "whole-part" relationship.
- Depicted as a filled diamond at the "whole" end.
- The "part" cannot exist independently of the "whole".
- Example: A `House` class composed of `Room` classes.

1.2.4 Aggregation

- Represents a weak "whole-part" relationship.
- Depicted as an empty diamond at the "whole" end.
- The "part" can exist independently of the "whole".
- Example: A `Library` class aggregating `Book` classes.

1.2.5 Generalization

- Represents an inheritance relationship.
- Depicted as a line with a hollow triangle pointing to the superclass.
- Example: A `Vehicle` class generalized by `Car` and `Bike` classes.