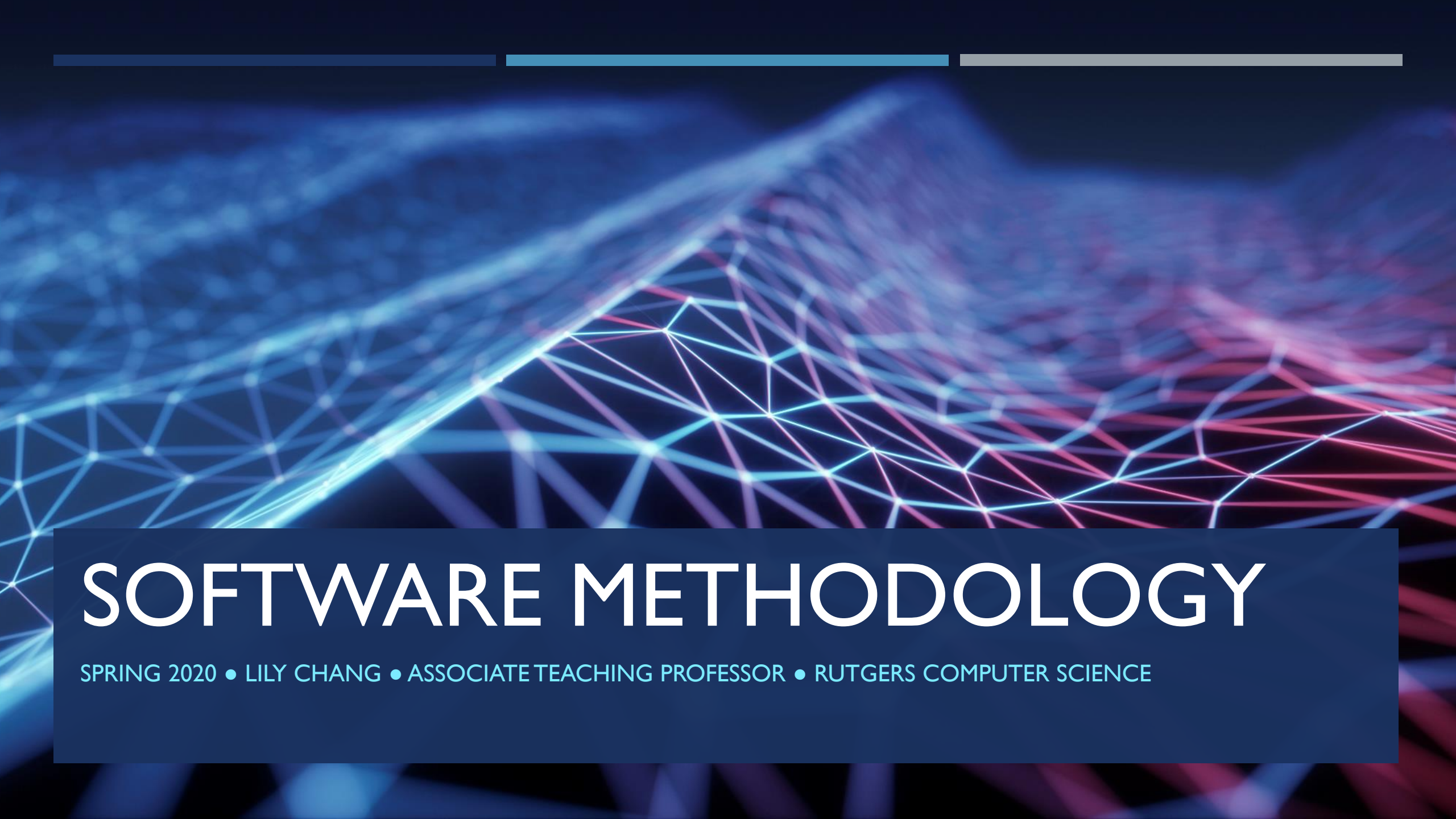


SOFTWARE METHODOLOGY

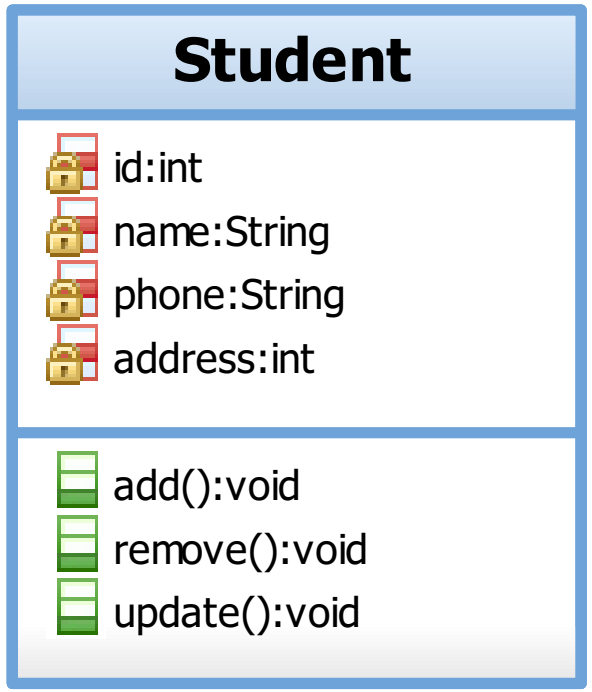
SPRING 2020 • LILY CHANG • ASSOCIATE TEACHING PROFESSOR • RUTGERS COMPUTER SCIENCE

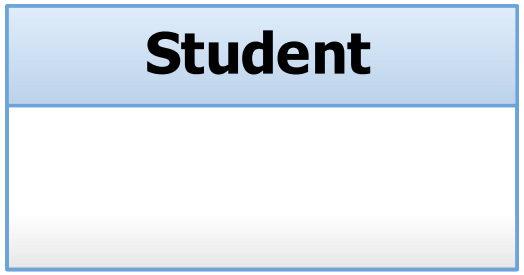
CLASS DIAGRAM

UML CLASSES

- A class is the definition of the attributes, the operations, and the semantics of a set of objects
- Classes are represented by rectangles which either bear only the name of the class (in bold), or show attributes and operations as well.
- Class names begin with an uppercase letter and are singular nouns
- Attributes and operations are listed at least with their names

NOTATION – UML CLASSES



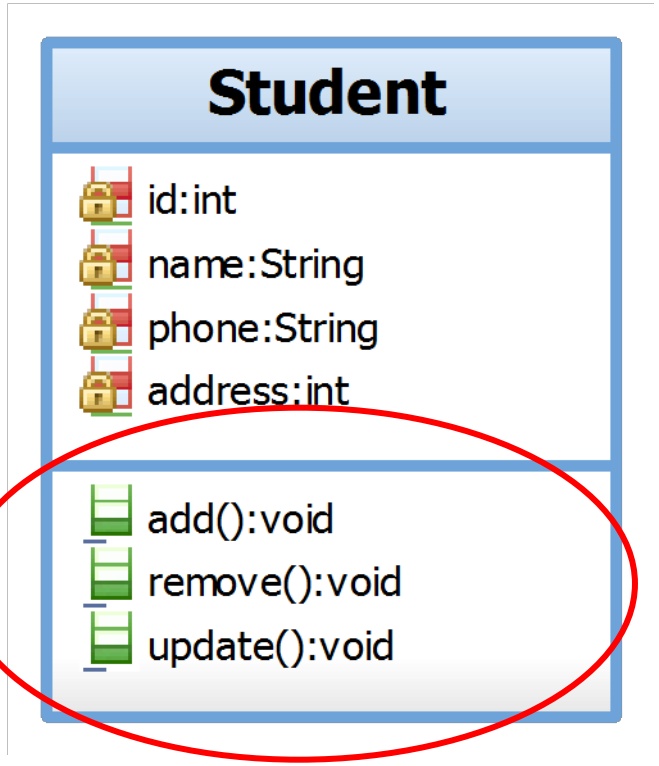


ATTRIBUTES

- An attribute is a data element which is contained in each object of a class
- Each attribute is at least described by its name; a data type or a class, plus an initial value and constraints may be defined.
- Attributes names begin with lowercase characters
- Visibility modifiers
  - + public, # protected, - private

OPERATIONS OR METHODS

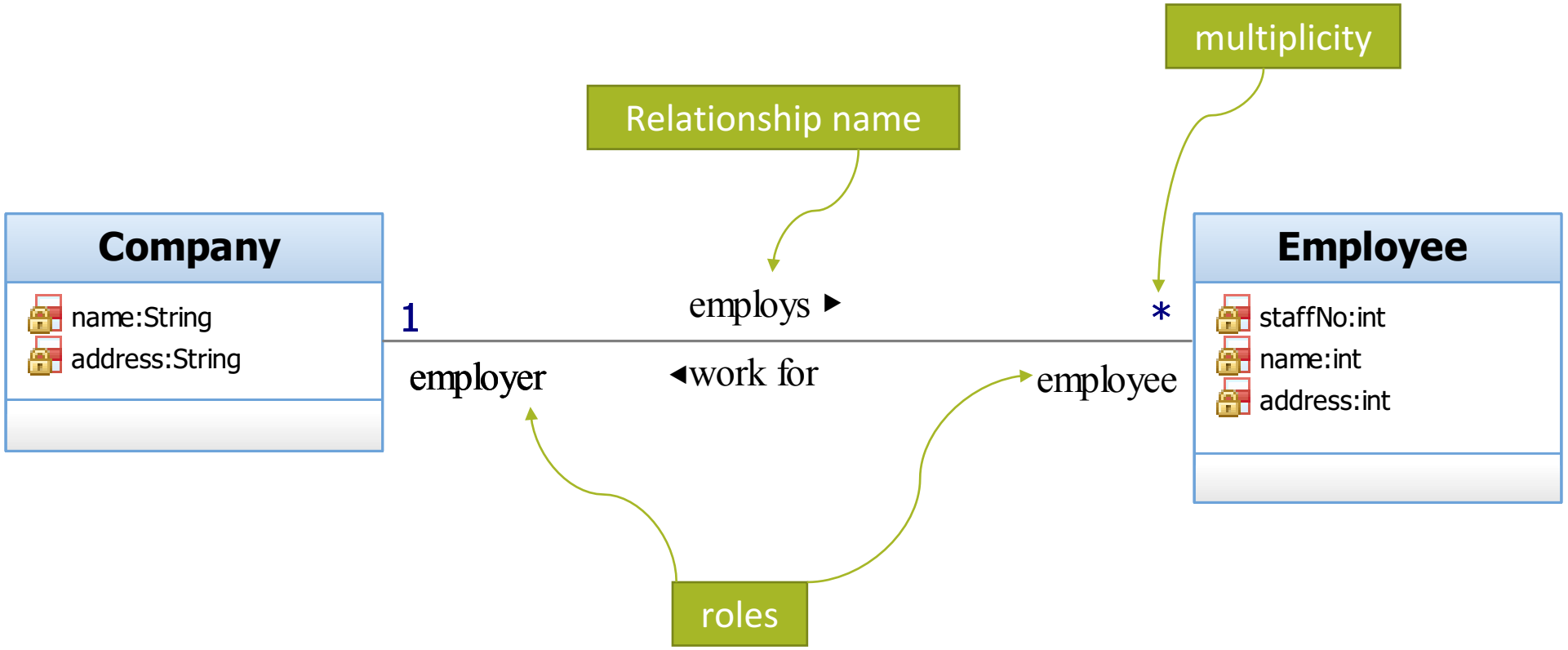
- Operations are services which may be required from an object, and are described by their signatures (operation name, parameters and return type)
- A method implements an operation
- A message passes an object the information on the activity it is expected to carry out
- A message consists of a name and a list of arguments



ASSOCIATIONS

- Dependencies and Generalizations represent difference level of importance or difference level of abstraction
- Association is a structural relationship that specifies that objects of one thing are connected to objects of another (objects are peers)
- You can navigate from an object of one class to an object of the other class, and vice versa. That is, objects rely on each other for services and data.
- Associations may be annotated with information that describes the association.
- These relationships identify public methods that are used to promote the relationship.

ASSOCIATION EXAMPLE

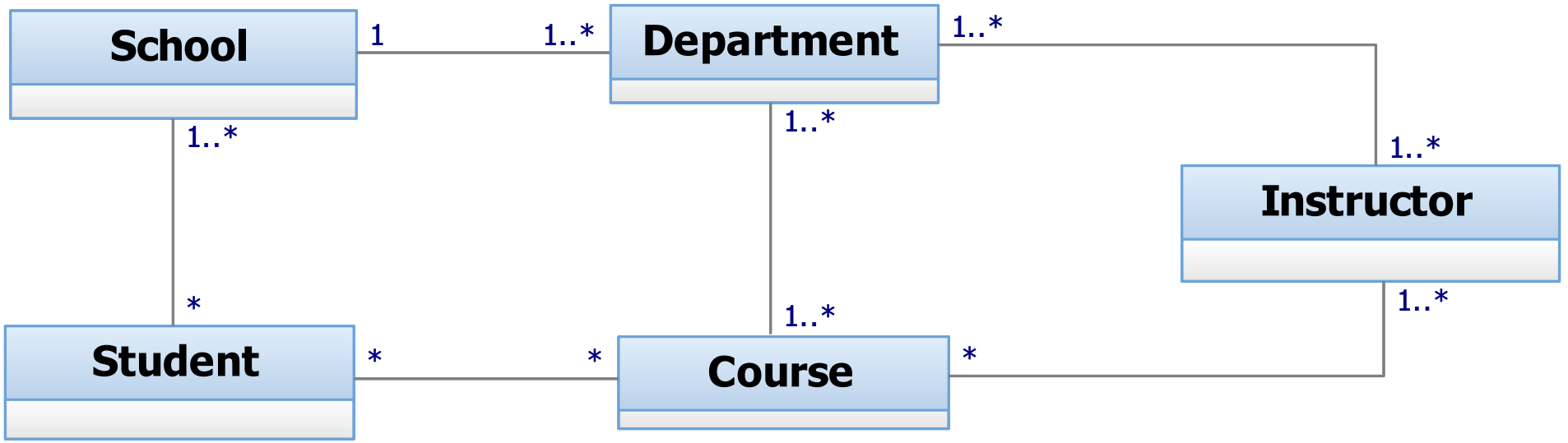


MULTIPLICITY

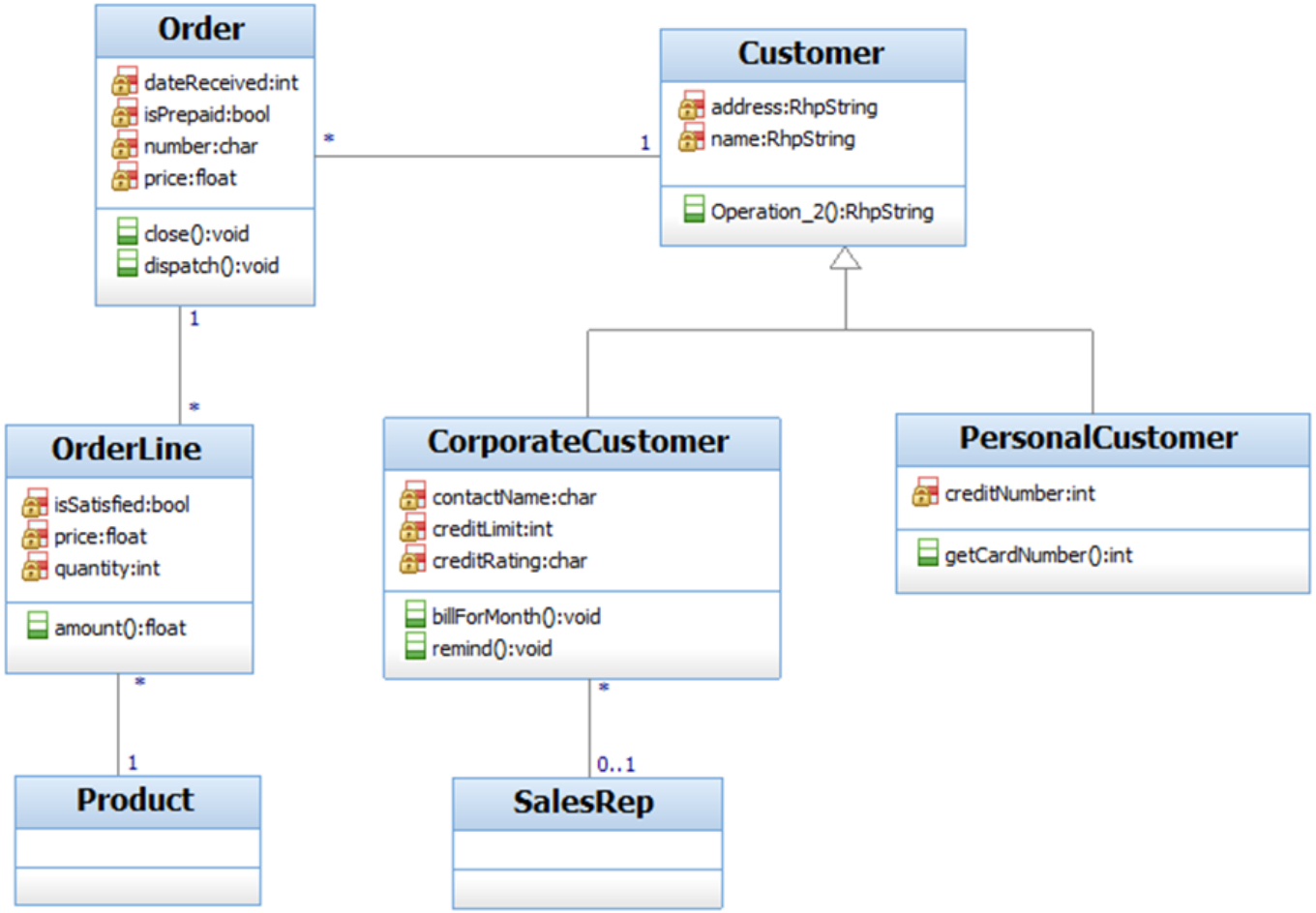
- Specify the number of objects of the opposite class to which an object can be associated
  - Cardinality—number of elements
  - Multiplicity—range of allowed cardinalities

`1`	Exactly one
`0, 1`	zero or one
`0..4`	Between zero and four
`3, 7`	Either three or seven
`*`	ditto
`1..*`	Greater than or equal to one
`0..3, 7, 9..*`	Between zero and three, or exactly seven, or greater than or equal to nine

MULTIPLICITY EXAMPLE

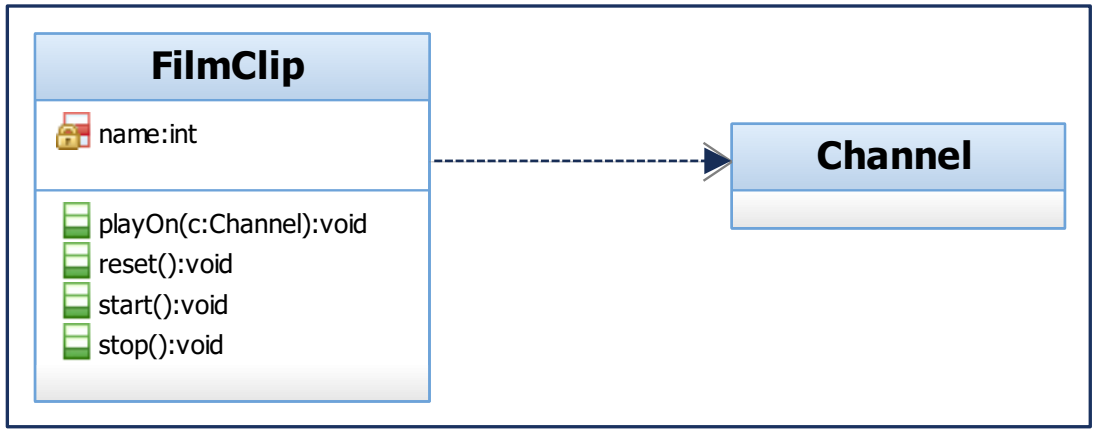


CLASS DIAGRAM EXAMPLE



DEPENDENCIES

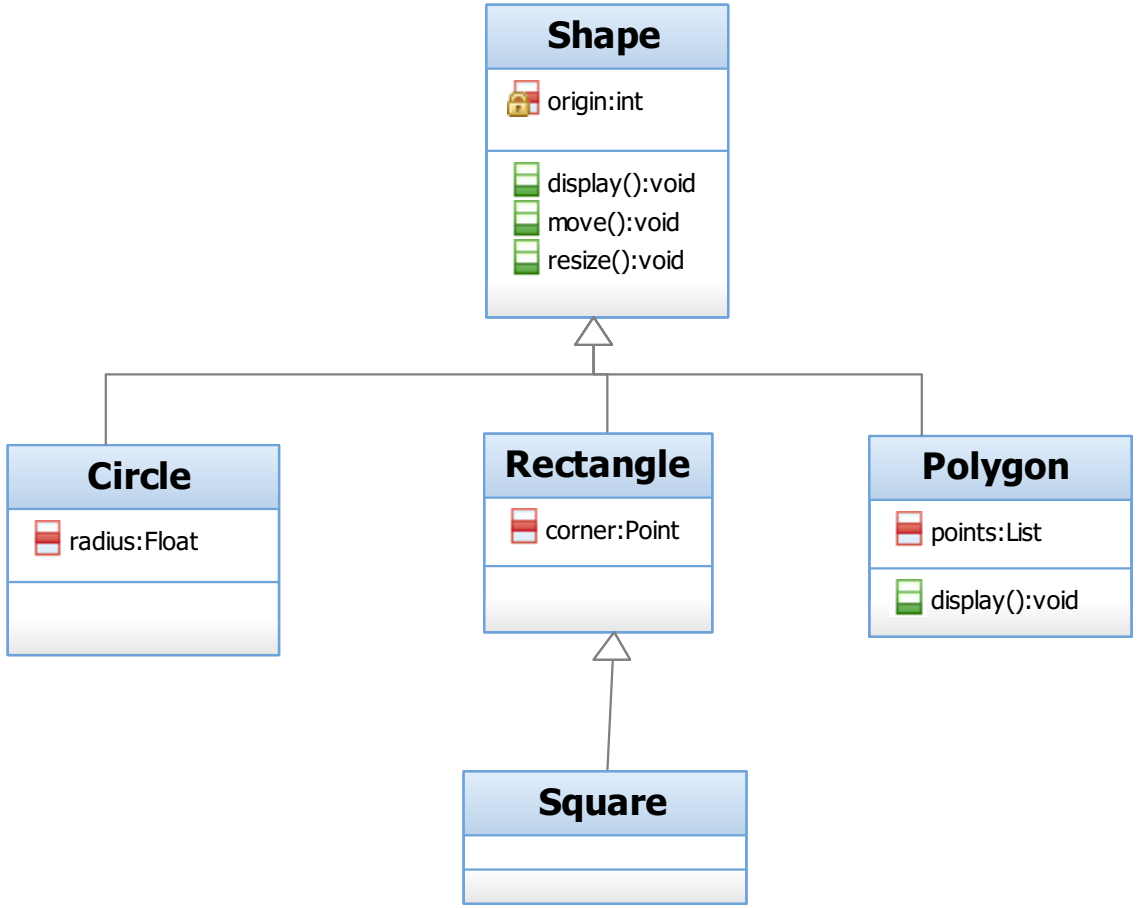
- Dependency is a "using" relationship
- A change in specification of one thing may affect another thing that uses it
- For example, one class uses another class as an argument in the signature of an operation



GENERALIZATIONS

- A relationship between a general thing (superclass) and a more specific kind of that thing (subclass)
- "is-a-kind-of" relationship, for example, "rectangle" is a kind of "shape"
- A subclass inherits the attributes and operations from its superclass and may add new methods or attributes of its own.
- Generalization in the UML is implemented as inheritance in OO programming languages.

GENERALIZATION EXAMPLE



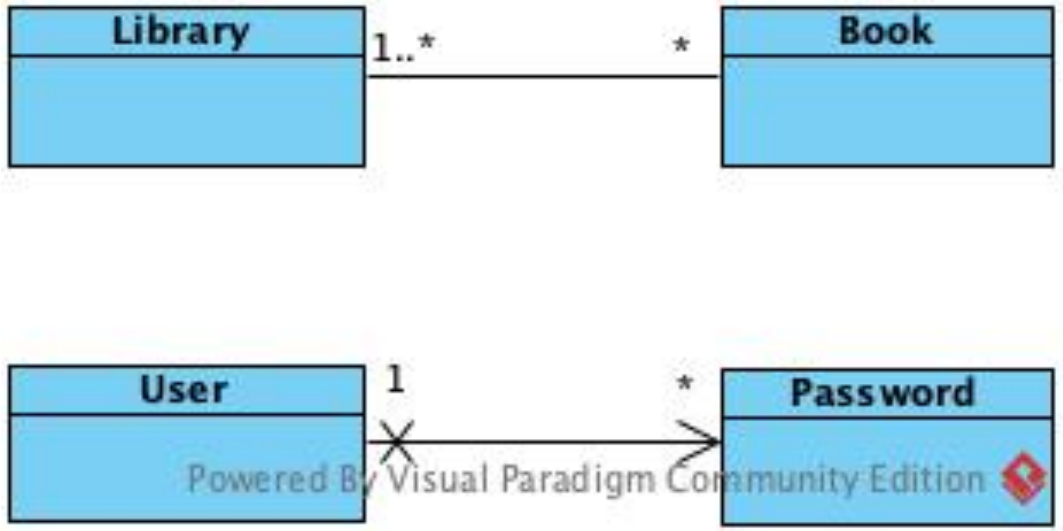
REFINING THE ASSOCIATIONS

- Navigation
- Aggregation
- Composition

NAVIGATION

- Directed associations
- For example,

Given a User, you'll be able to find the corresponding Password objects, but given a Password you don't want to be able to identify the corresponding User.

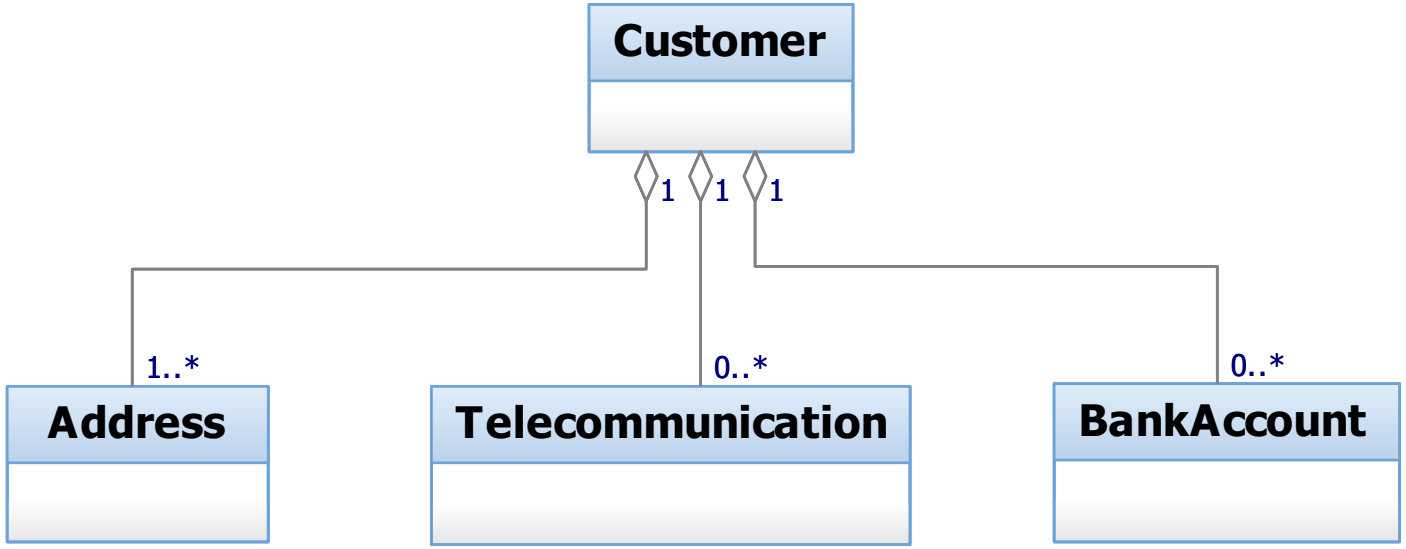


AGGREGATION

- Aggregations are special associations that represent "part-whole" relationships
- A "has-a" relationship, meaning that an object of the whole has objects of the part.
- Exactly one end of the relation must be the aggregate (whole), and the other stand for the individual parts.
- An aggregation is represented by a line drawn between two classes, and it is marked with a small empty diamond on the side of the aggregate

AGGREGATION EXAMPLE

- Customer class has the attributes of Address, Telecommunication and BankAccount
- Address, Telecommunication and BankAccount could be a part of other classes

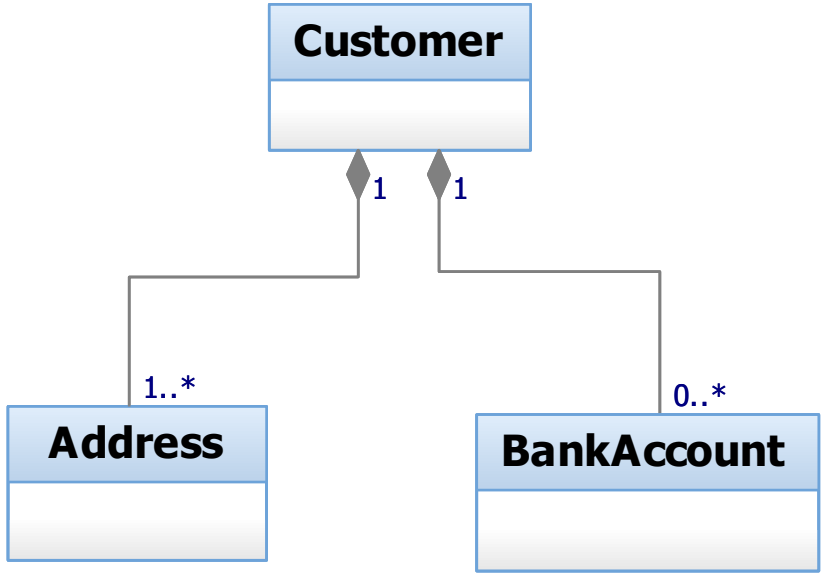


COMPOSITION

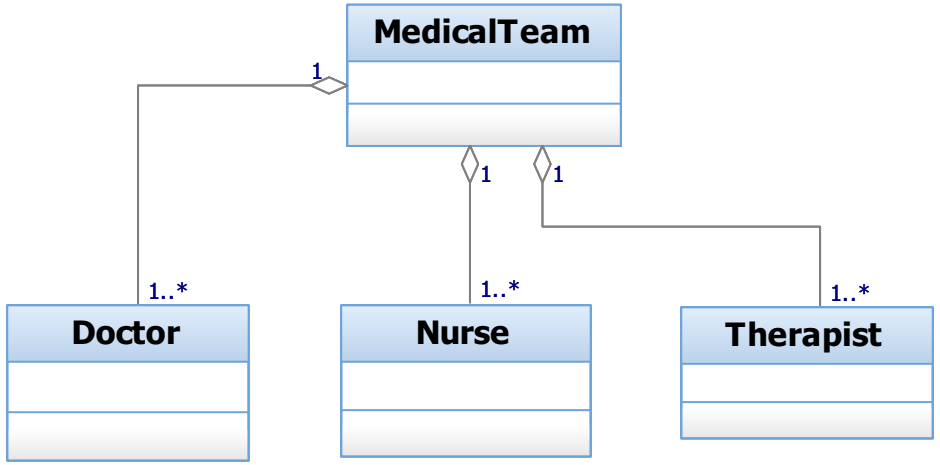
- A composition is a strict form of aggregation, in which the parts are existence dependent on the entirety
- If a part is deleted, the aggregate survives.
- If the aggregate is deleted, all parts are deleted with it
- An object may be a part of only one composite at a time
- The composite must manage the creation and destruction of its parts

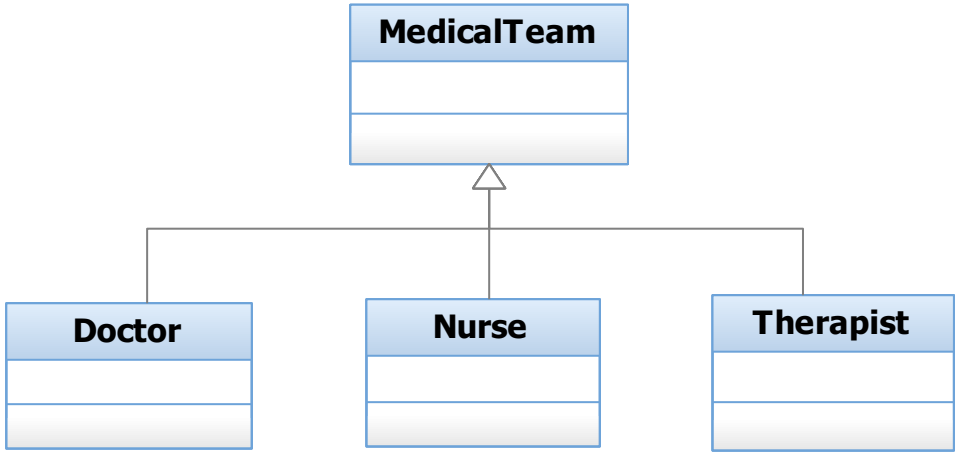
COMPOSITION EXAMPLE

- Customer class has the attributes of Address and BankAccount
- Address and BankAccount CANNOT be a part of other classes
  - Address and BankAccount live and die with the Customer



HOW WOULD YOU INTERPRET THE DIAGRAMS?





RULES FOR CLASS DIAGRAM

- Class names: singular nouns (occasionally, noun phrases) , begin with an uppercase letter
- Attributes: nouns; usually begin with a lowercase letter
- Operations: verbs/verb phrases, or nouns if it is a value-returning operation
- Use legal identifiers in target language, no reserved words, no illegal characters

THANK YOU