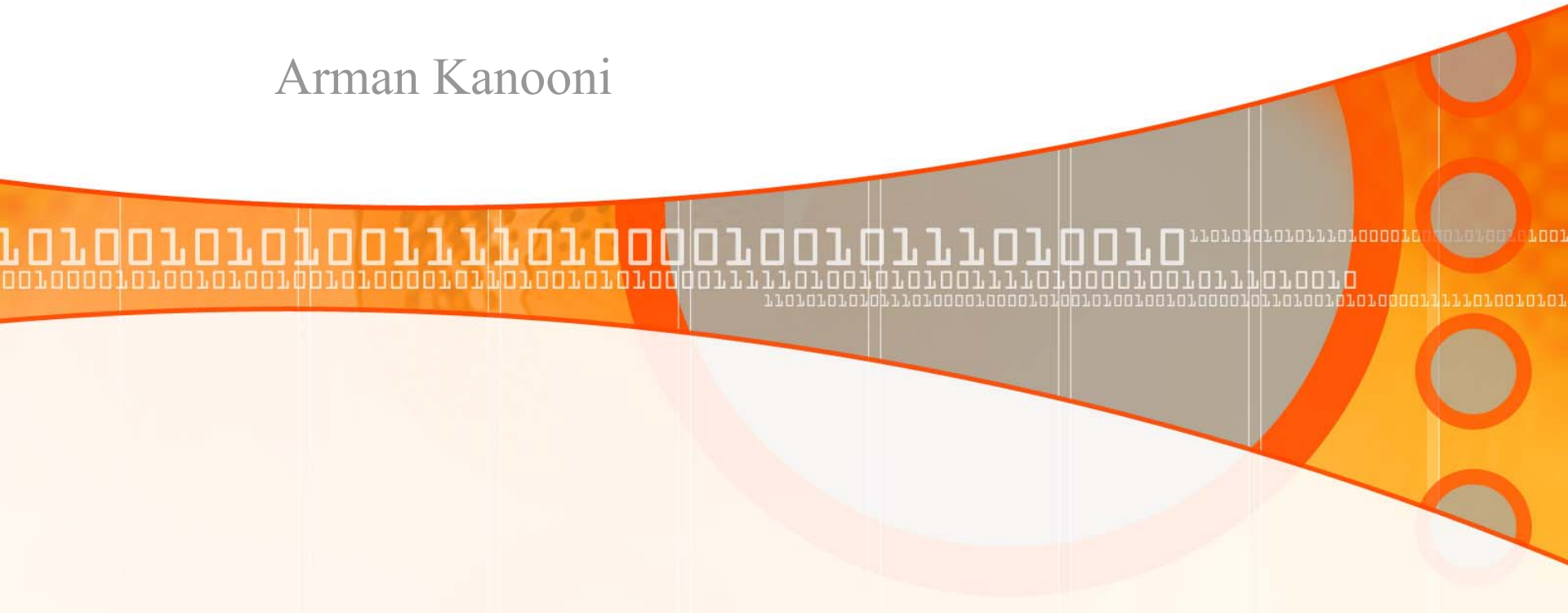


UML

Unified Modeling Language Overview

Arman Kanooni



Plan

1. *Class Diagrams: The Essentials*
 2. *Class Diagrams: Advanced Concepts*
 3. *Interaction Diagrams*
- 

1. Class Diagram: The Essentials

- 1.1. *Definition*
- 1.2. *Perspectives*
- 1.3. *Associations*
- 1.4. *Attributes*
- 1.5. *Operations*
- 1.6. *Generalization*
- 1.7. *Constraints*

1. Class Diagram: The Essentials (cont'd)

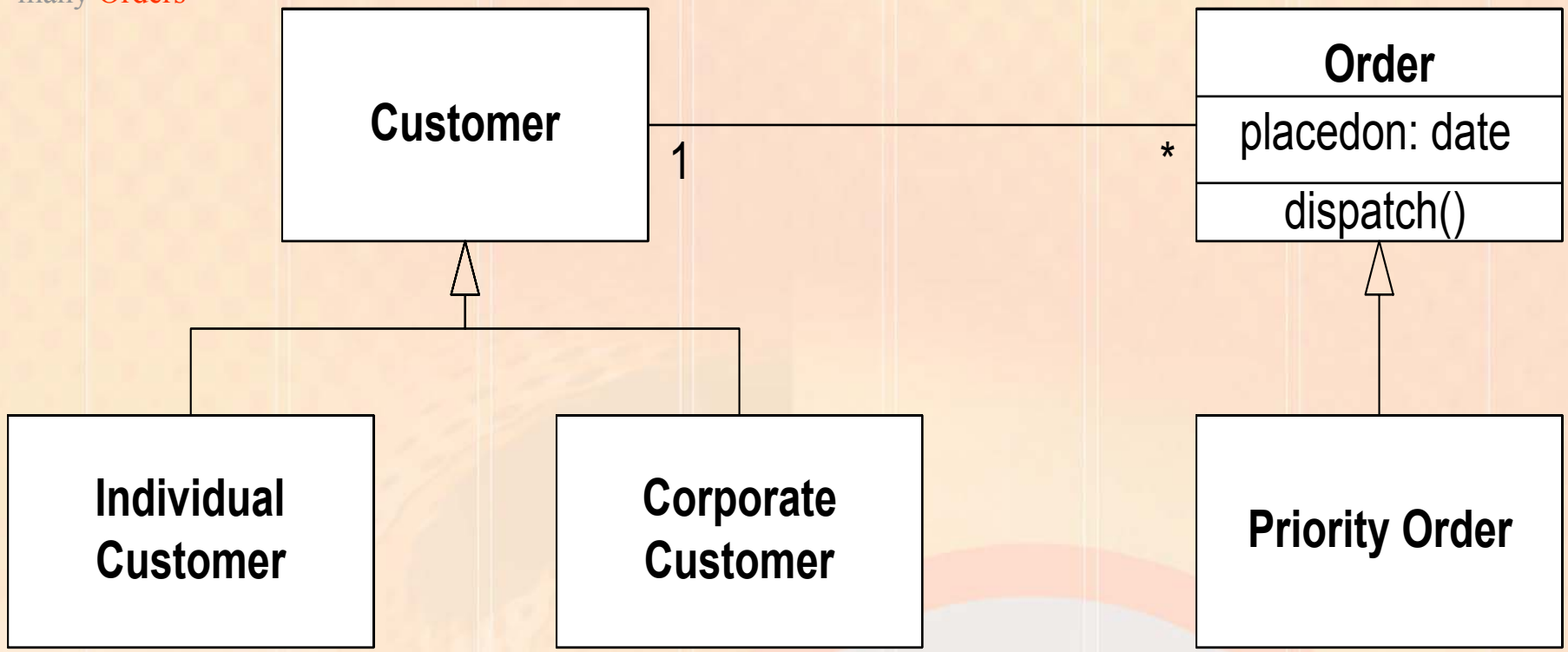
1.1. Definition:

- *A Class Diagram describes the types of objects in the system and the various kinds of Static Relationships that exist among them.*
- *2 principal kinds of Static Relationships:*
 - **Association** (A Customer may rent a number of videos.)
 - **Subtypes** (A Nurse is a kind of person.)

1. Class Diagram: The Essentials (cont'd)

Each **Customer** may have many **Orders**

Each **Order** has a single **Customer**



A **Customer** may be an **Individual Customer** or **Corporate Customer**



1. Class Diagram: The Essentials (cont'd)

1.2. Perspectives

- *The way you should interpret a diagram.*
- *There are 3 Perspectives:*
 - *Conceptual*
 - *Specification*
 - *Implementation*

1. Class Diagram: The Essentials (cont'd)

1.2. Perspectives (cont'd)

- *The lines between the perspectives are not sharp.*
- *When draw a diagram, draw it from a single clear perspective.*
- *Perspective is not part of formal UML!*

1. Class Diagram: The Essentials (cont'd)

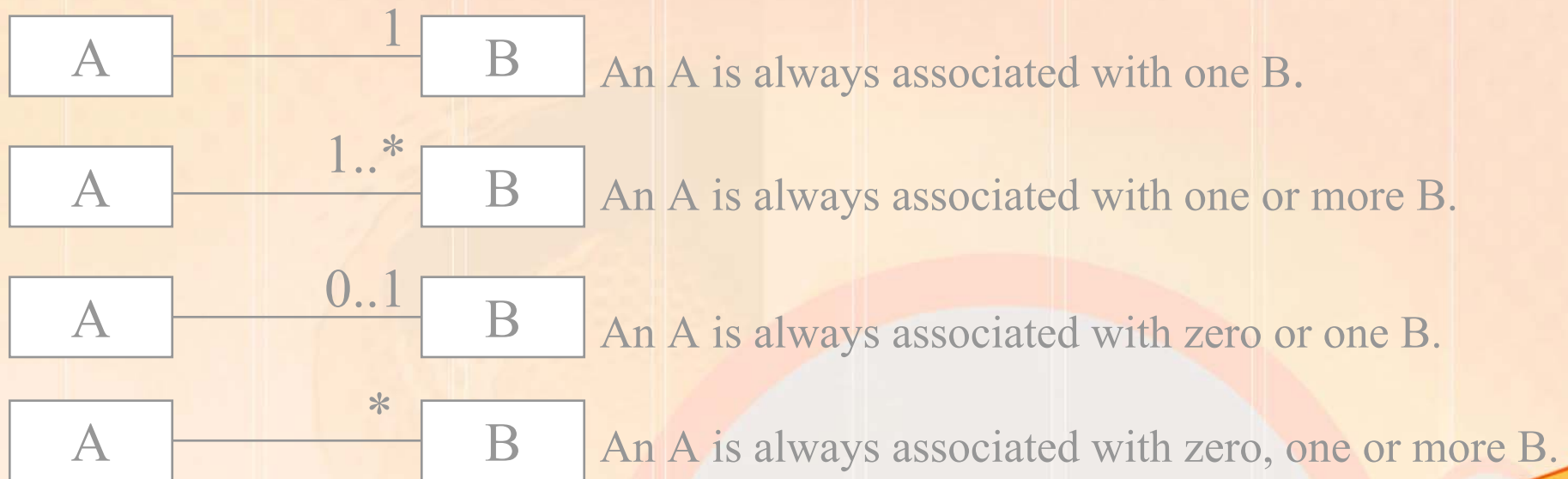
1.3. Associations

- *Represent relationship between instances of classes.*
- *Each Association has 2 **Roles**; each Role is a direction of the association.*
- *A Role can be explicitly named with a **Label**.*
- *If there is no Label, you name a Role after the Target Class.*

1. Class Diagram: The Essentials (cont'd)

1.3. Associations (Cont'd)

- A role has **Multiplicity**, which is an indication of how many objects participates in the given relationship.



1. Class Diagram: The Essentials (cont'd)

1.4. Attributes

Syntax:

visibility *name*: **type** = default value

Where **visibility** is :

- + (*public*)
- # (*protected*)
- - (*private*)

1. Class Diagram: The Essentials (cont'd)

1.5. Operations:

- *They are Processes that a Class knows to carry out.*

Syntax:

visibility name (parameter-list) : return-type-expression {property-string}

1. Class Diagram: The Essentials (cont'd)

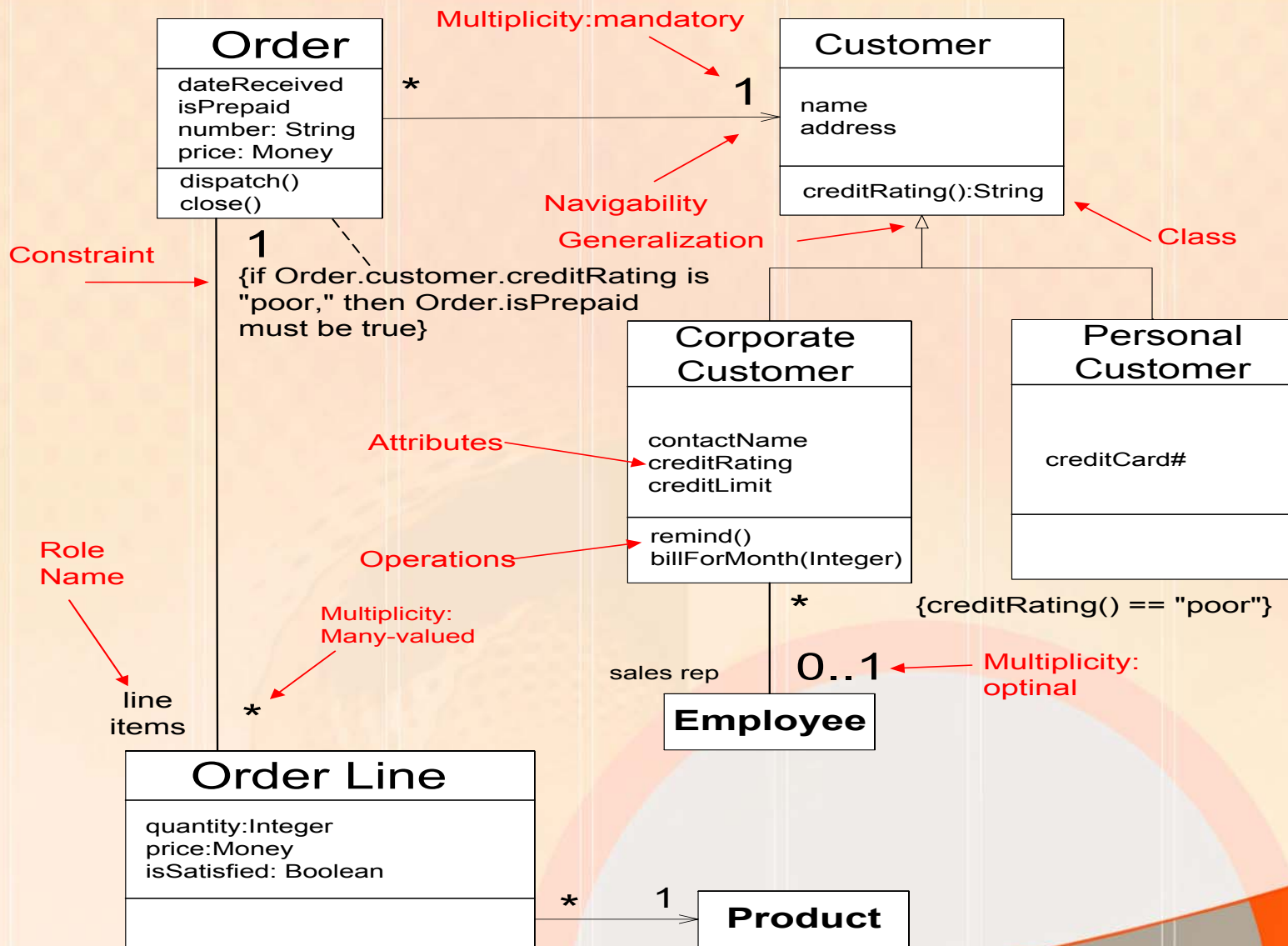
1.6. Generalization

- *Personal and Corporate Customers of a business. They have differences but also many similarities. The similarities can be placed in a general **Customer Class** (**Super Type**) with **Personal Customer** and **Corporate Customer** as **Subtype**.*

1. Class Diagram: The Essentials (cont'd)

1.7. Constraint Rules

- An **Order** can be placed by a single **Customer**.
- Each **Line Item** is thought of separately: you say 40 brown widgets, 40 blue widgets, and 40 red widgets, not 40 red, blue, and brown widgets.
- **Corporate Customers** have credit limits but **Personal Customer** do not.



2. Class Diagram: Advanced Concepts

- 2.1. *Stereotypes*
- 2.2. *Multiple and Dynamic Classification*
- 2.3. *Aggregation and Composition*
- 2.4. *Derived Associations and Attributes*
- 2.5. *Interfaces and Abstract Classes*
- 2.6. *Qualified Associations*
- 2.7. *Association Class*
- 2.8. *Parameterized Class*

2. Class Diagram: Advanced Concepts (cont'd)

2.1. Stereotypes

- *High-level classification of an object that gave you some indication of the kind of object it was.*
- *As subtypes of the meta-model types class, Association, and Generalization.*
- *Ex: “Controller” versus “Coordinator”*

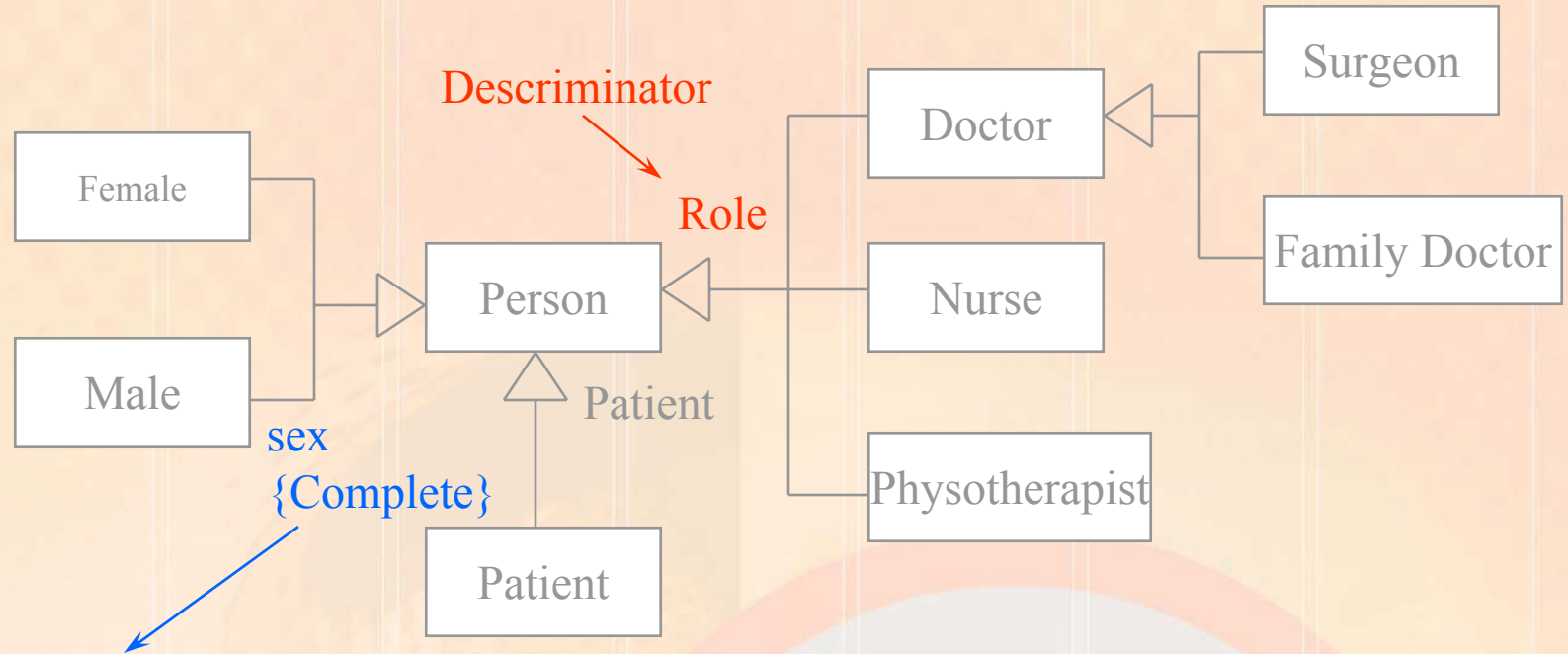
2. Class Diagram: Advanced Concepts (cont'd)

2.2. Multiple and Dynamic Classification

- Consider a person subtyped as either man or woman, doctor or nurse, patient or not. **Multiple Classification** allows an object to have any of these type assigned to it in any allowable combination without the need for types to be defined for all the legal combination.

2. Class Diagram: Advanced Concepts (cont'd)

2.2.1. Multiple Classification

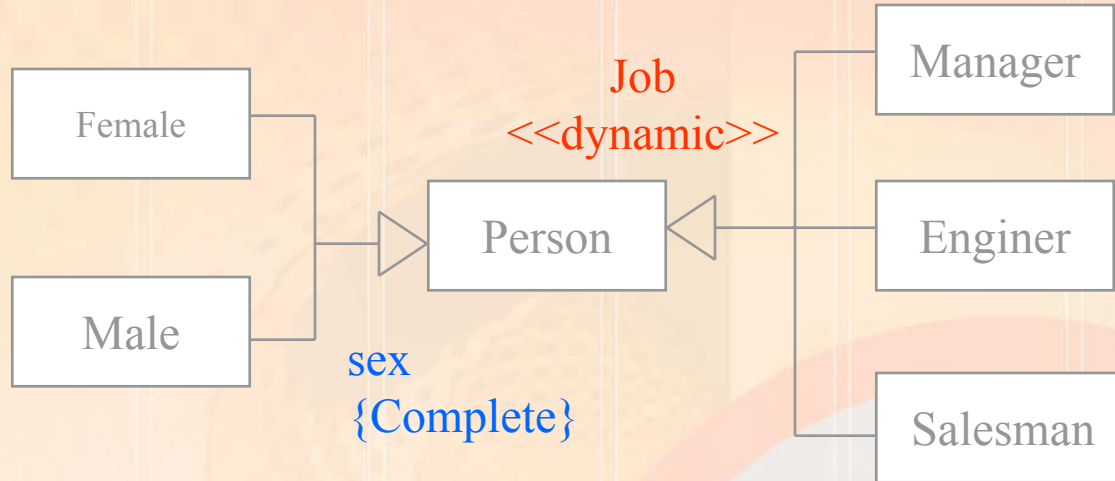


Any instance of the superclass must be an instance of one of the subclass in that group.

2. Class Diagram: Advanced Concepts (cont'd)

2.2.2. Dynamic Classification

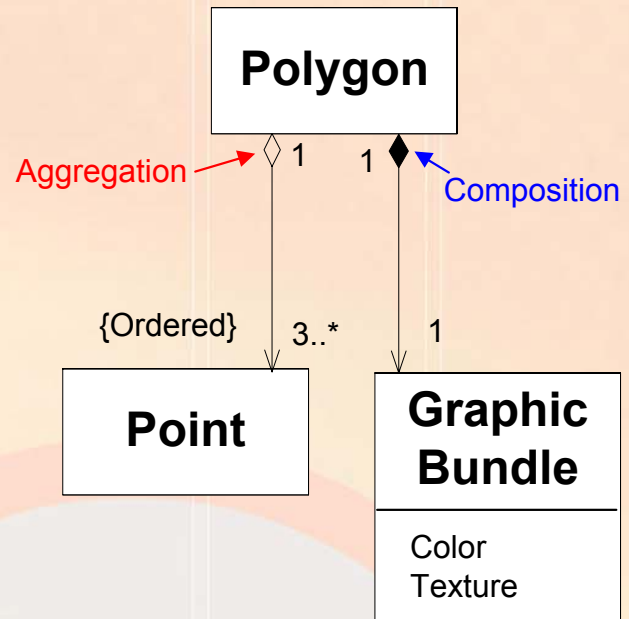
- Allows objects to change type within the subtyping structure.



2. Class Diagram: Advanced Concepts (cont'd)

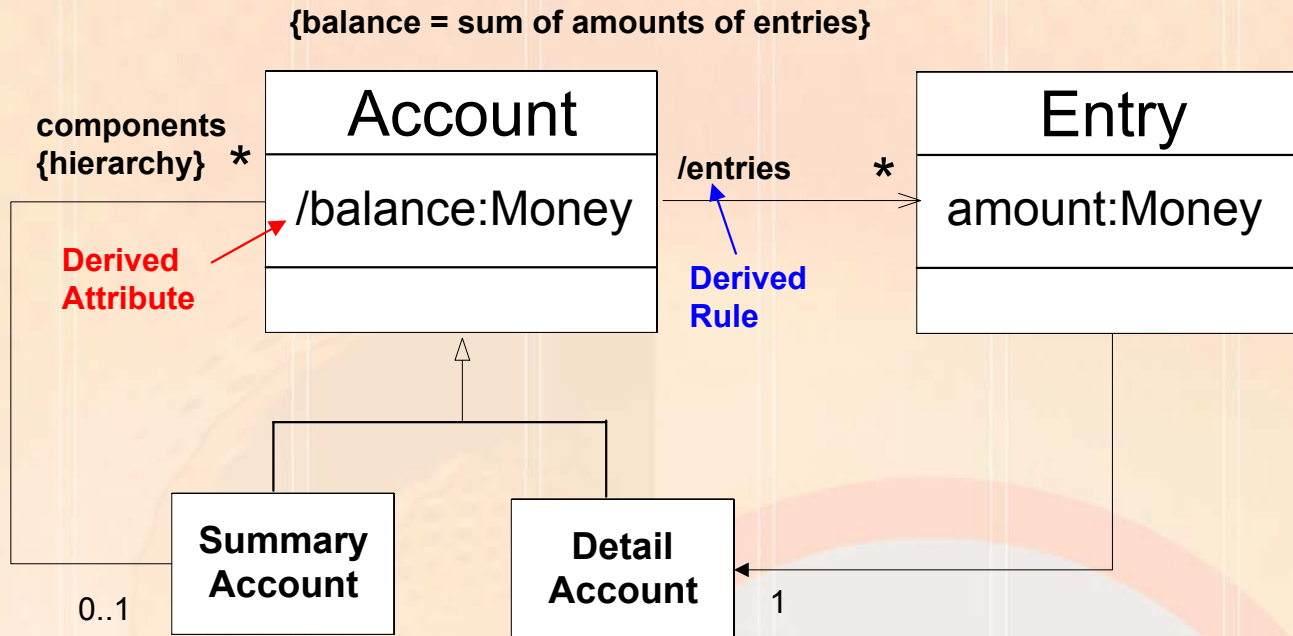
2.3. Aggregation and Composition

- **Aggregation** is Part-Of relationship.
- **Composition** is the part object may belong to only one whole.



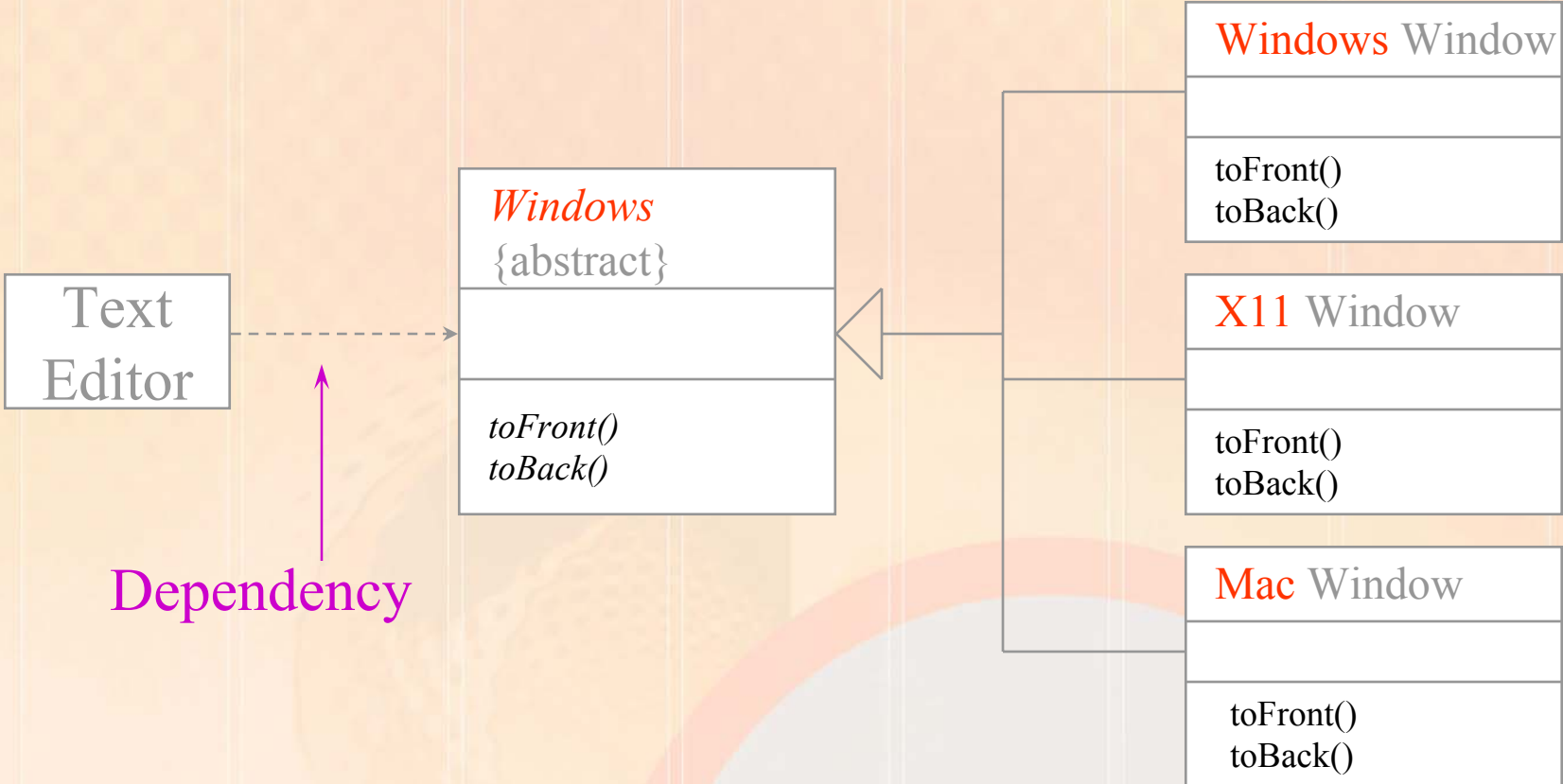
2. Class Diagram: Advanced Concepts (cont'd)

2.4. Derived Associations and Attributes



2. Class Diagram: Advanced Concepts (cont'd)

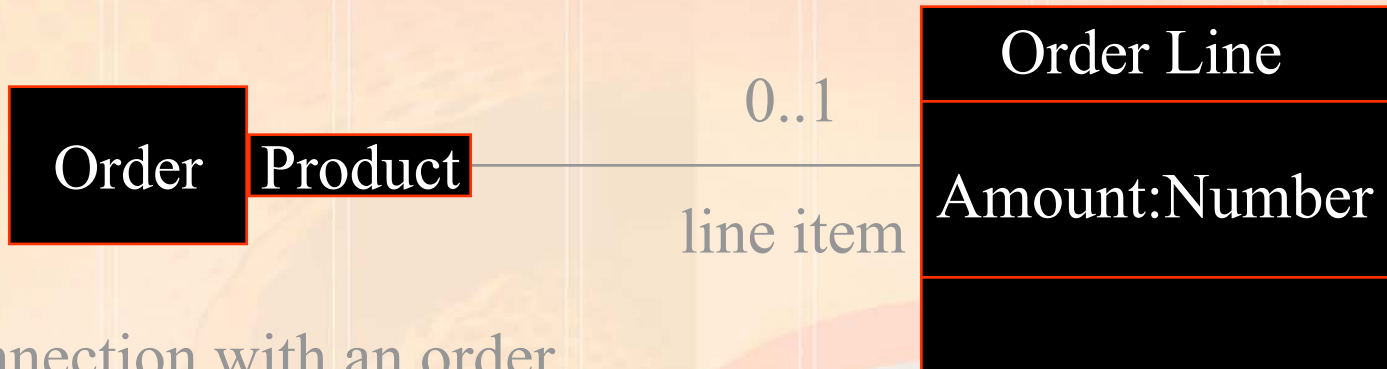
2.5. Interfaces and Abstract Classes



2. Class Diagram: Advanced Concepts (cont'd)

2.6. Qualified Associations

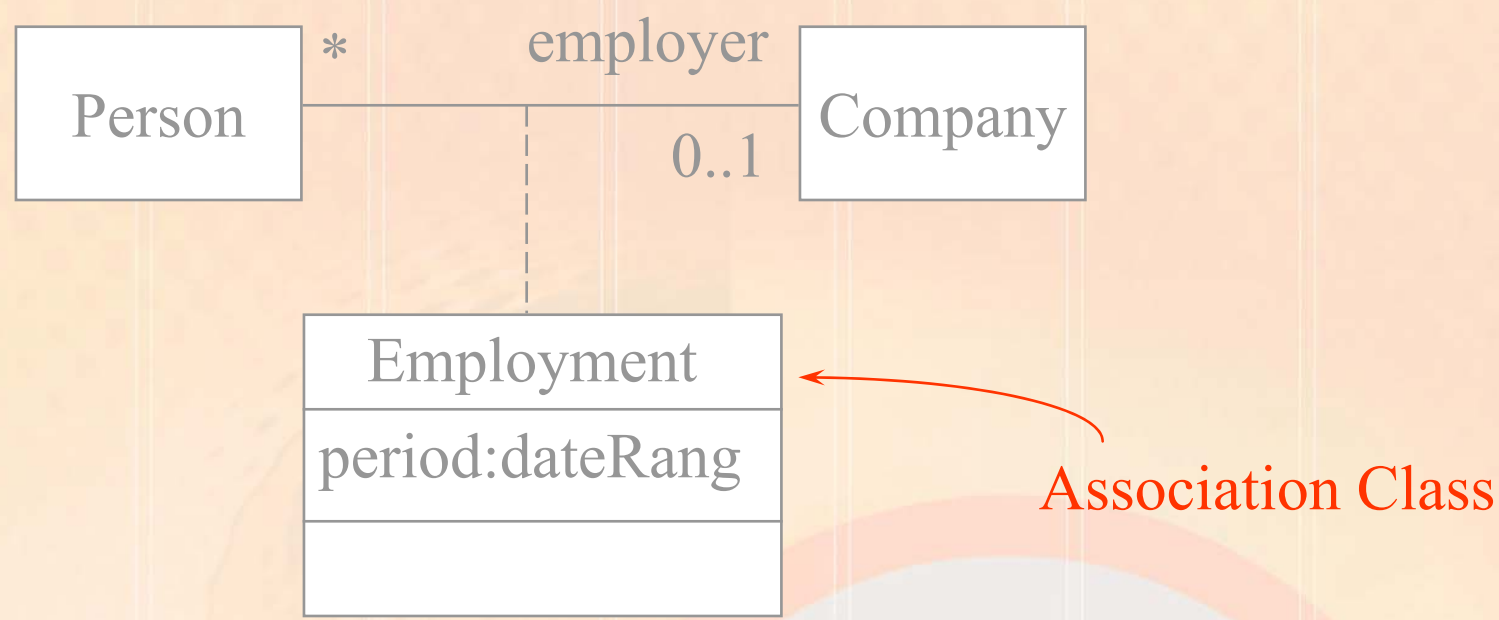
- *A qualified association is the UML equivalent of a programming concept known as Associative Arrays, Maps, and Dictionaries.*



In connection with an order, there may be one Order Line for each instance of Product.

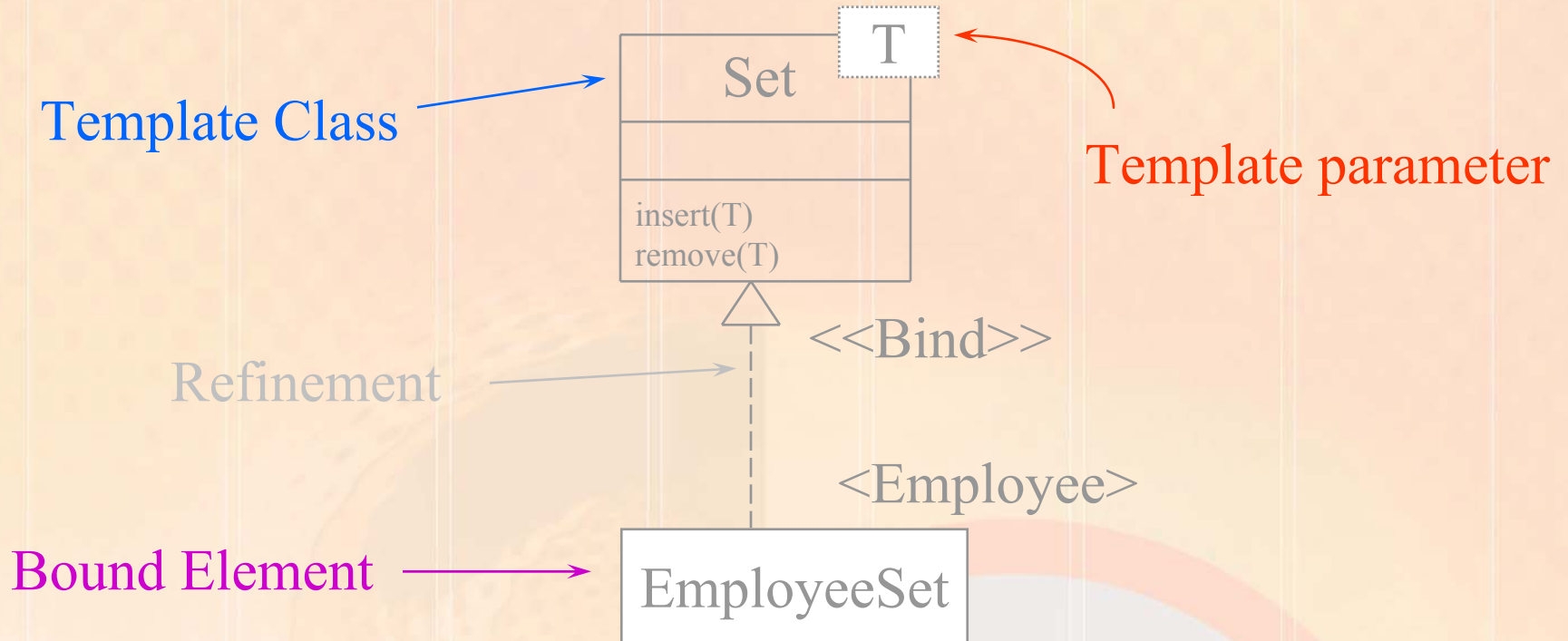
2. Class Diagram: Advanced Concepts (cont'd)

2.7. Association Class



2. Class Diagram: Advanced Concepts (cont'd)

2.8. Parameterized Class



3. Interaction Diagrams

- 3.1. *Sequence Diagrams*
- 3.2. *Collaboration Diagrams*
- 3.3. *Comparing Sequence and Collaboration Diagrams*

3. Interaction Diagrams (cont'd)

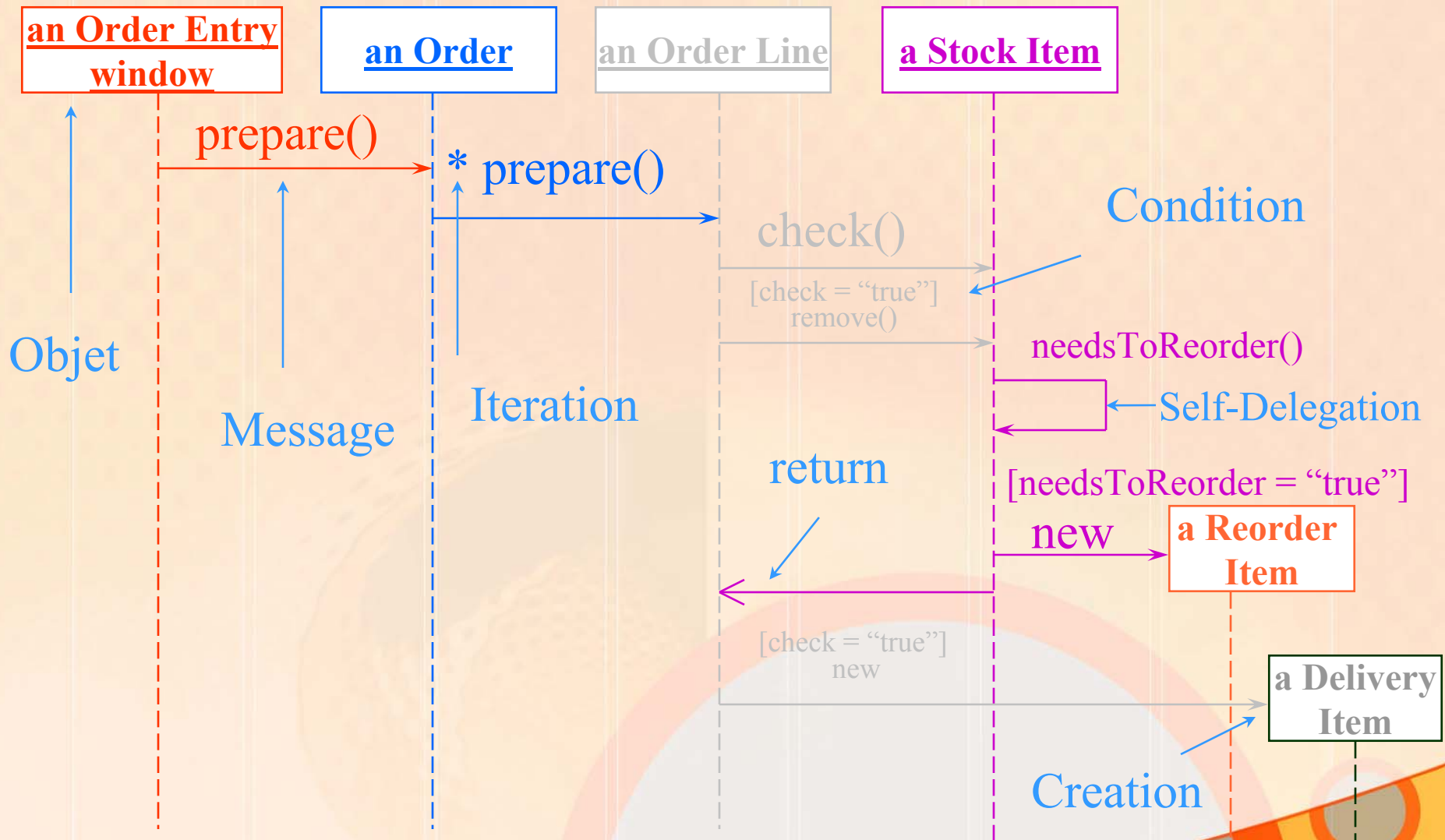
- *Definition: A pattern of interaction among objects is shown on an interaction diagram. Interaction diagrams come in two forms based on the same underlying information but each emphasizing a particular aspect of it:*
 - *Sequence Diagrams,*
 - *Collaboration Diagrams.*

3. Interaction Diagrams (cont'd)

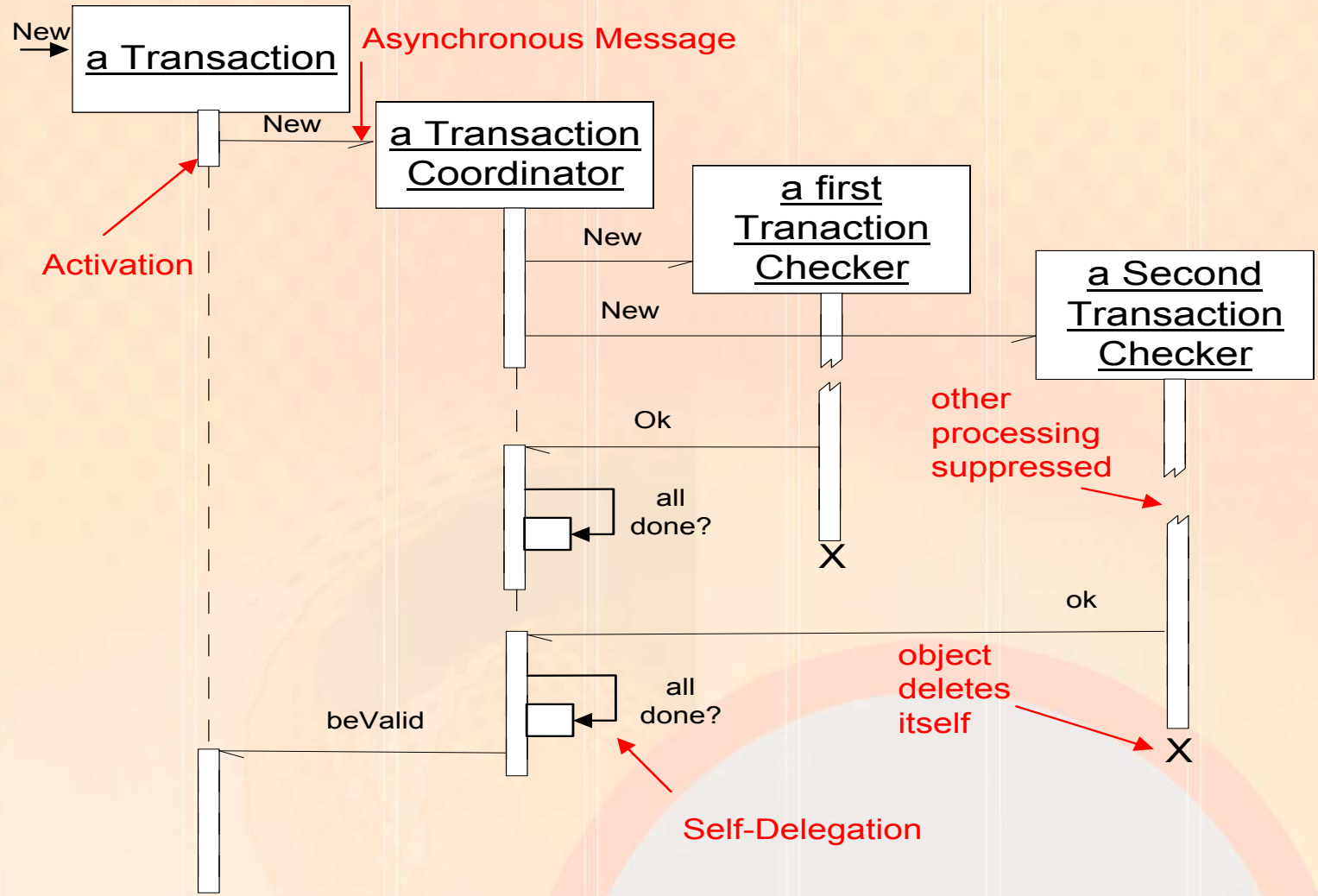
3.1. Sequence Diagrams

- *Semantics*
 - *A Sequence Diagram represents an Interaction, which is a set of messages exchanged among objects within a collaboration to effect a desired operation or result.*
- *Notation*
 - *2 dimensions:*
 - *Vertical dimension represents time.*
 - *Horizontal dimension represents different objects.*

3. Interaction Diagrams (cont'd)



3. Interaction Diagrams (cont'd)

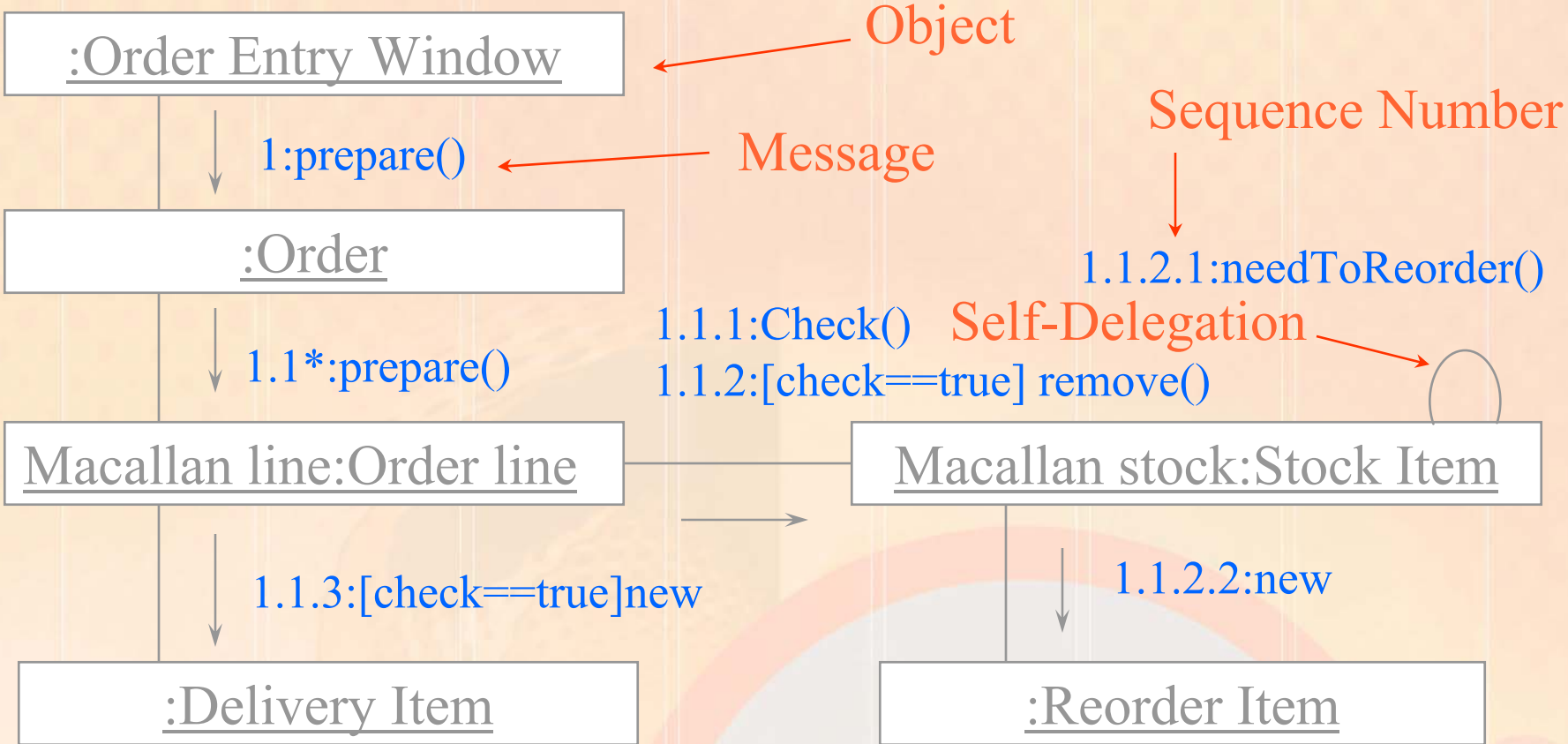


Concurrent Processes and Activations



3. Interaction Diagrams (Cont'd)

3.2. Collaboration Diagrams



3. Interaction Diagrams (cont'd)

3.3. Comparing Sequence and Collaboration Diagrams

- *Sequence Diagram puts its emphasis on **sequence**; it is easy to see the order in which things occur.*
- *Collaboration Diagrams indicate how objects are **statically connected**.*
- *One of the principal features of either form of an interaction diagram is its simplicity.*

Reference

For more information about UML visit:

- *<http://www.rational.com>
(link to creator of UML)*
- *<http://www.infosys.tuwien.ac.at/cetus/software.html>
(link to more 8000 Object Oriented resources on the net!)*