Static Modeling

Chapter 8

Part of Requirements Modeling

Designing Concurrent, Distributed, and Real-Time Applications with UML
Hassan Gomaa (2001)

Static Modeling

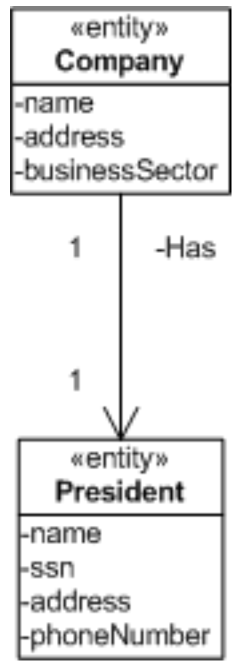
- Static Models describe the structural aspects of a problem.
  - The real-world entities involved.
  - These are less likely to change than functional requirements, are are hence called static.
- Static Models are depicted with UML's class diagrams in COMET.

Static Model Relationships (8.1)

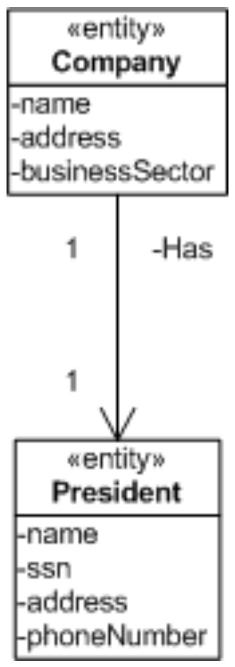
- Association
  - Denotes a static structural relationship.
- Composition and Aggregation
  - Denotes a *made-up-of* relationship.
  - Offers two strengths of relationship.
- Generalization and Specialization
  - Denotes an *is-a* relationship
  - Creates a hierarchy.

Association (8.1.1)

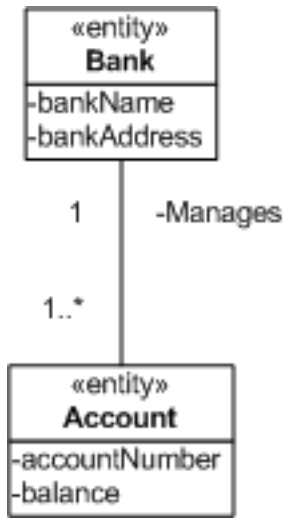
- Shown by an arc connecting two classes.
- Can demonstrate
  - Nature of the relationship.
  - Multiplicity at both ends.



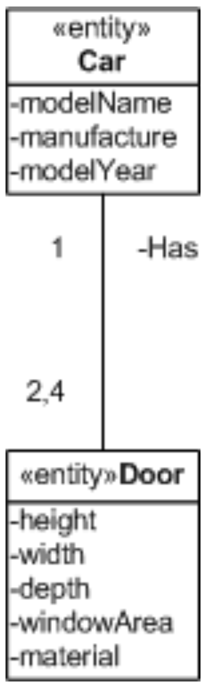
Multiplicity Examples (8.1.2)



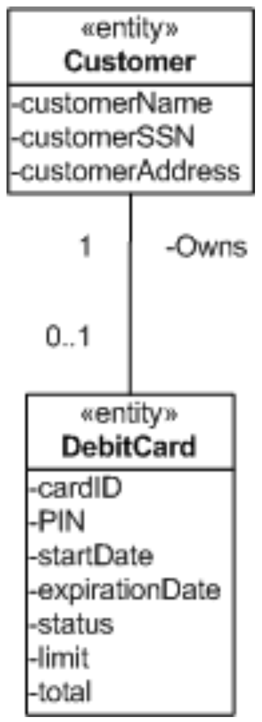
One-to-One



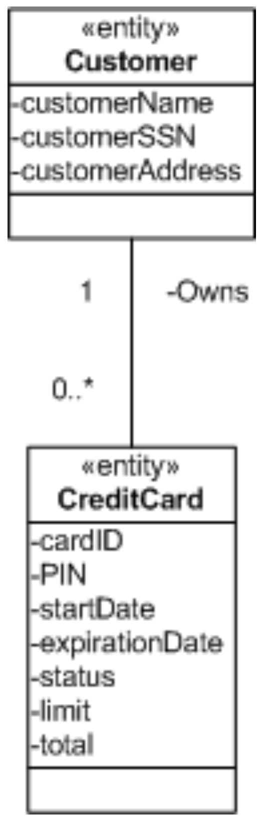
One-to-Many



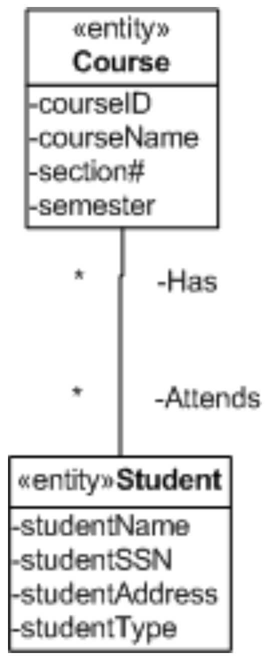
Numerically Specified



Optional - One



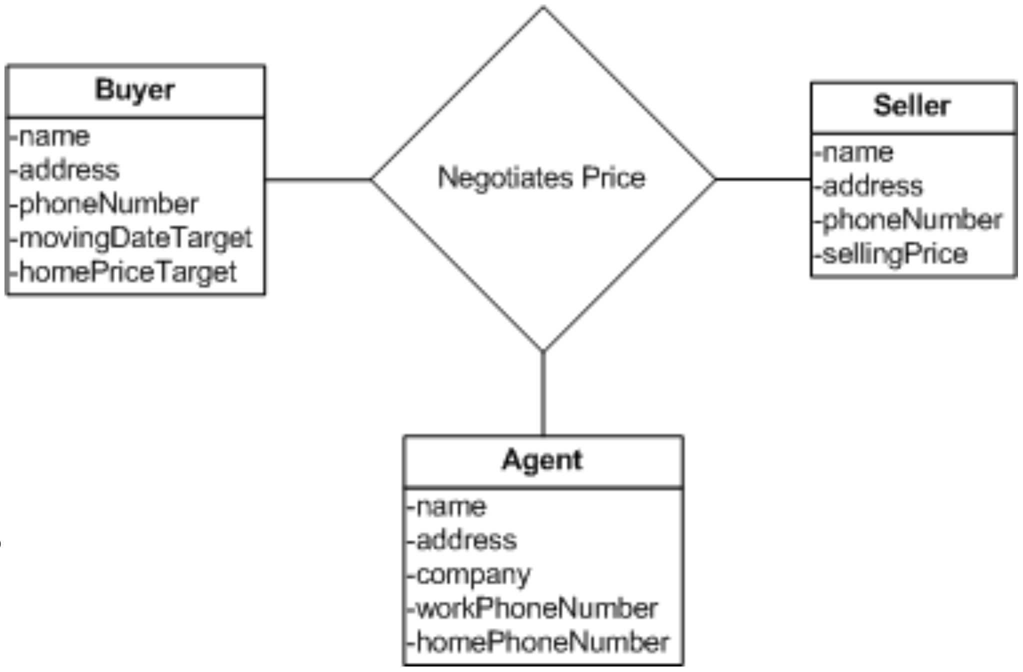
Optional - Many



Many-to-Many

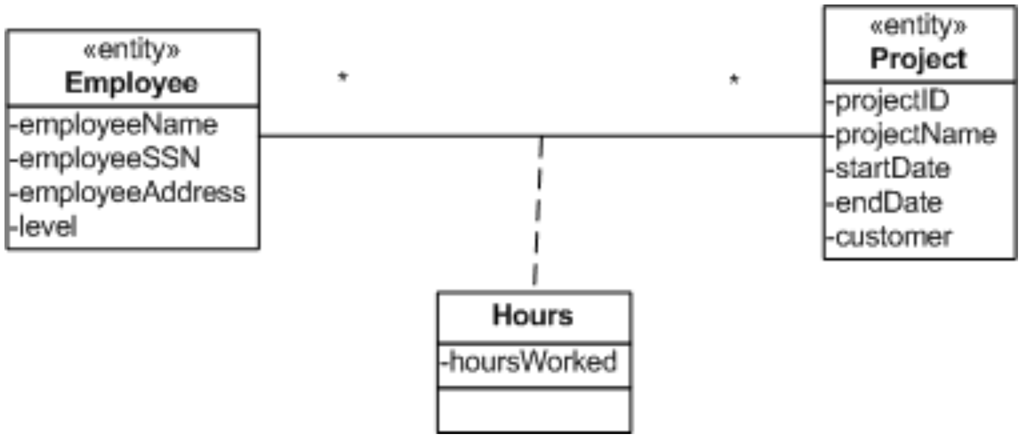
Link Attributes (8.1.4)

- Enable complex relationships
- Useful for many-to-many relationships
  - Allows attributes on the association itself.



Association Classes (8.1.5)

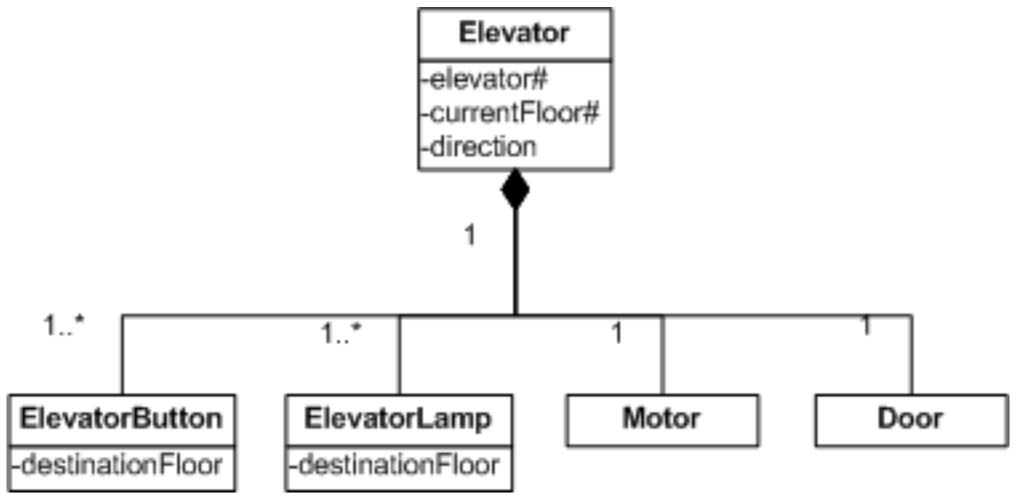
- An alternative to Link Attributes
- Allows a full class (not just attributes) to model the relationship between two other classes

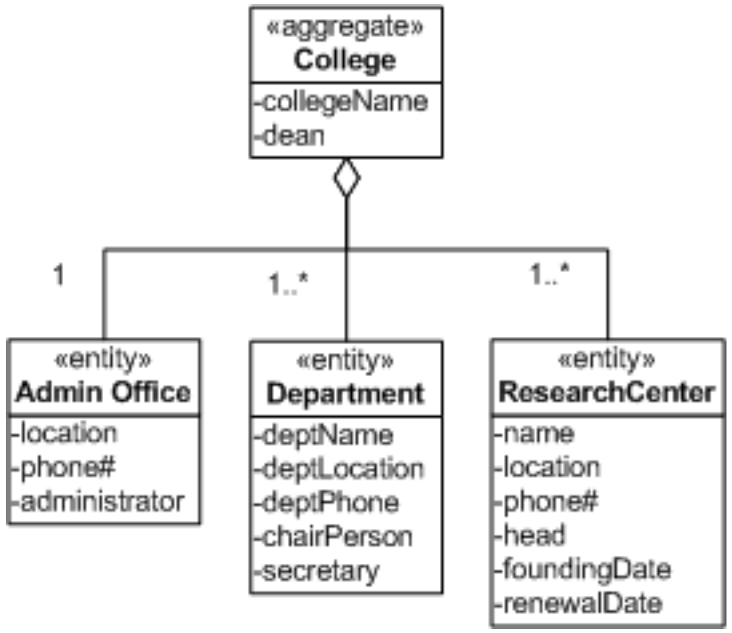


Composition and Aggregation (8.2)

- Composition
  - Parts of a composition can only belong to a whole.
  - All the parts live and die together, with the whole.
  - Marked by a shaded diamond on the connector.
- Aggregation
  - Parts can be added and removed.
  - Typically more conceptual than compositions.
  - Marked by a clear diamond on the connector.

Composition and Aggregation Examples (8.2)



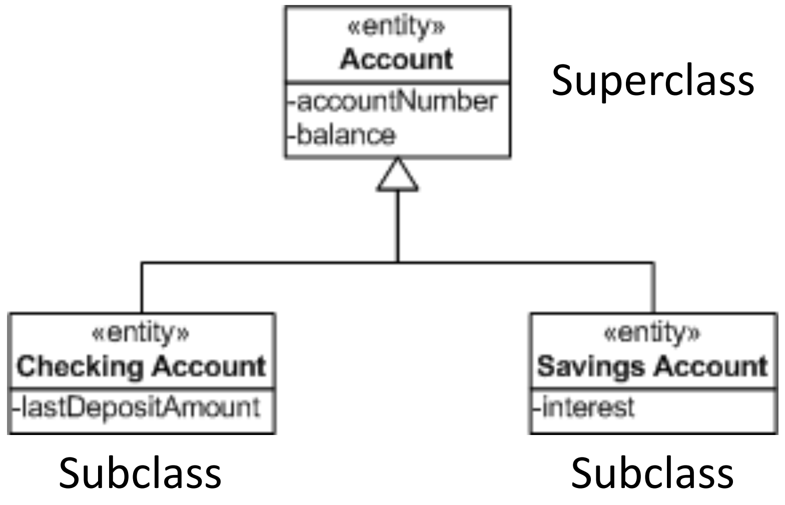


Composition

Aggregation

Generalization and Specialization (8.3)

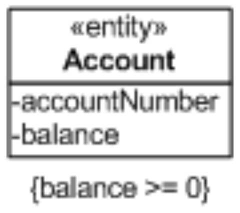
- Used to show similarities between classes
  - The similarities are abstracted into a generalization class.
  - Depicts an *is-a* relationship.

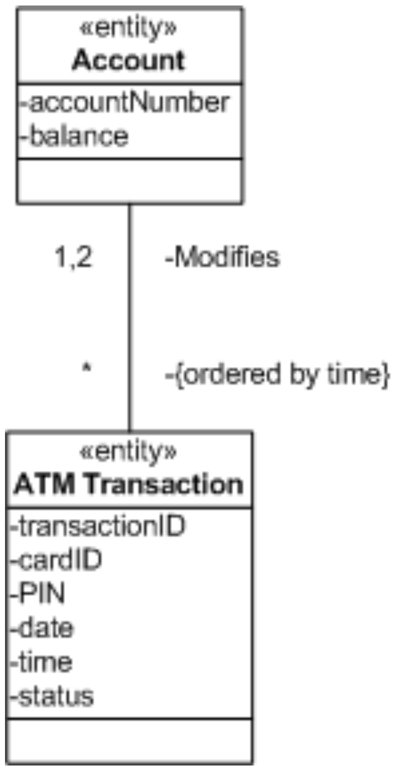


Constraints (8.4)

- Specify conditions or restrictions that must be true.

- Can constrain attributes or associations.

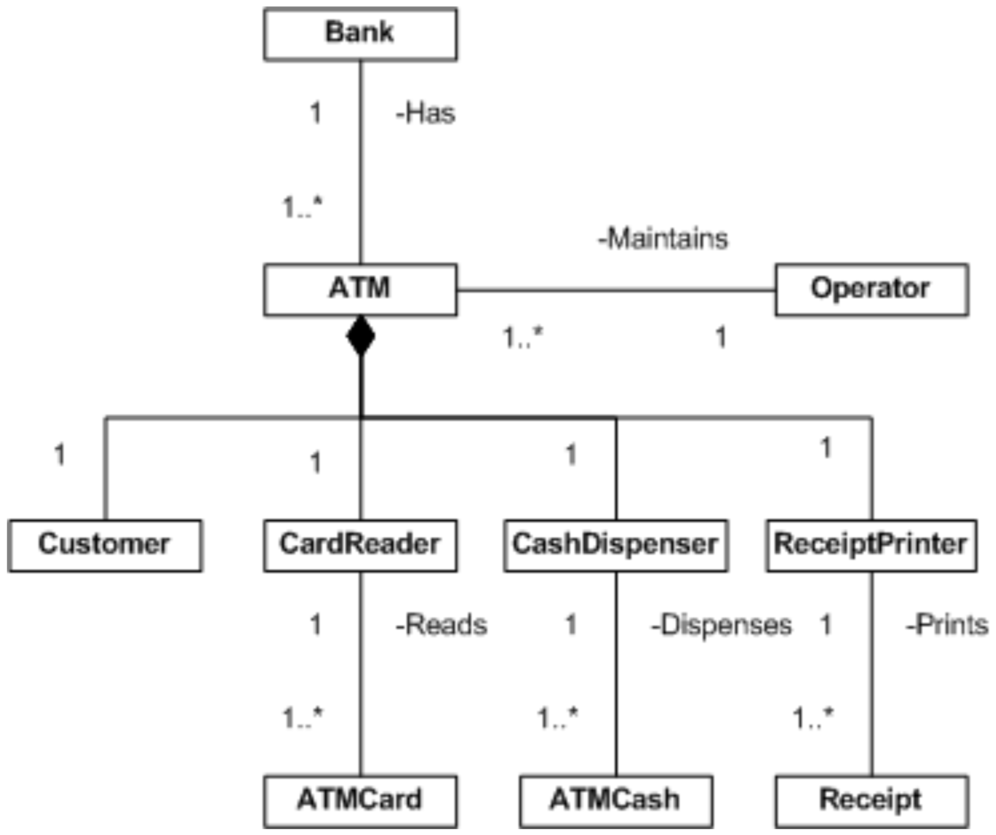




Static Modeling – Problem Domain (8.5)

- COMET initially emphasizes modeling:
  - Physical Classes
    - Have physical characteristics (i.e. real-world objects)
    - Include devices, users, external systems, and timers
  - Entity Classes
    - Long-lived, conceptual, data-intensive classes.
    - In a banking example, this would include things like accounts and transactions.

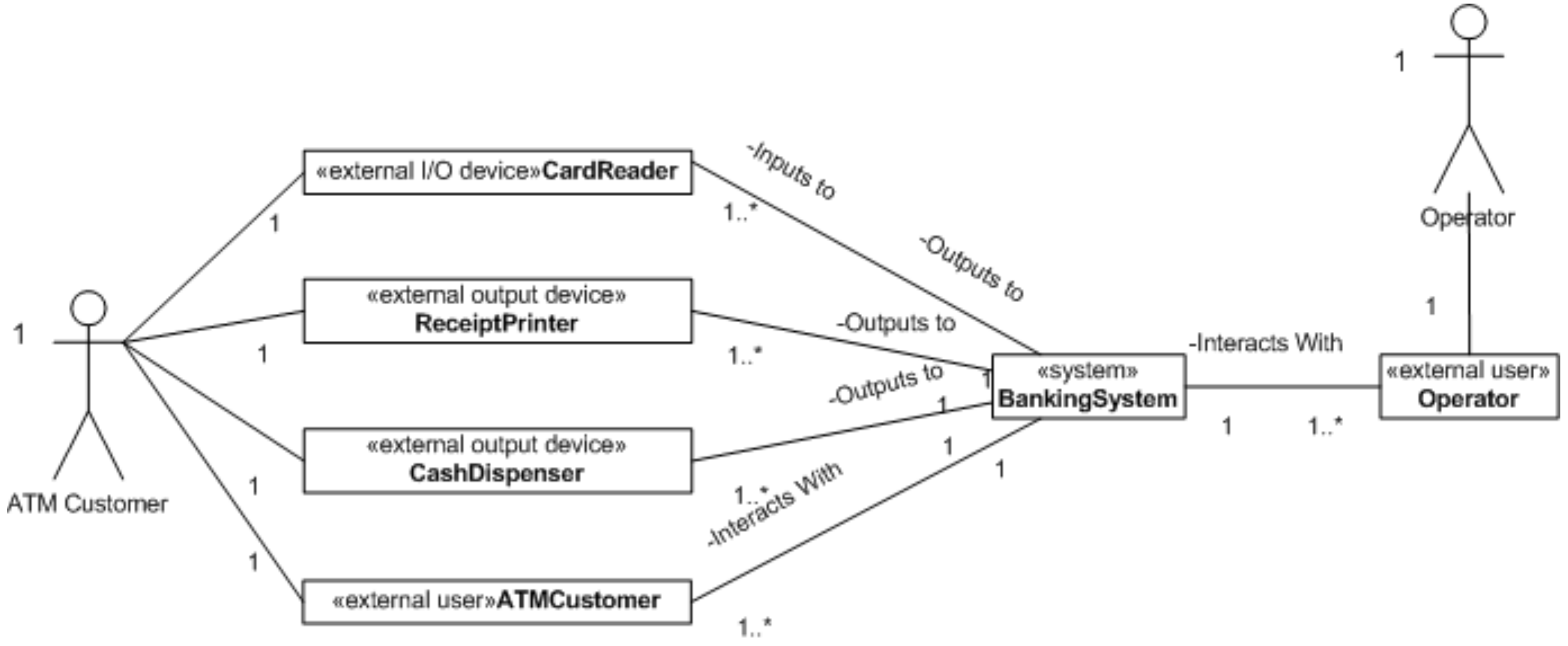
Physical Classes Example (8.5)



Static Modeling – System Context (8.6)

- Depicts the relationship between the system and it’s environment.
- Can be developed using actors or inputs and outputs.
- Use UML Stereotypes to model:
  - <<system>> for the system.
  - <<external…>> for classes in the context.
    - IO
    - Timer
    - System
    - User

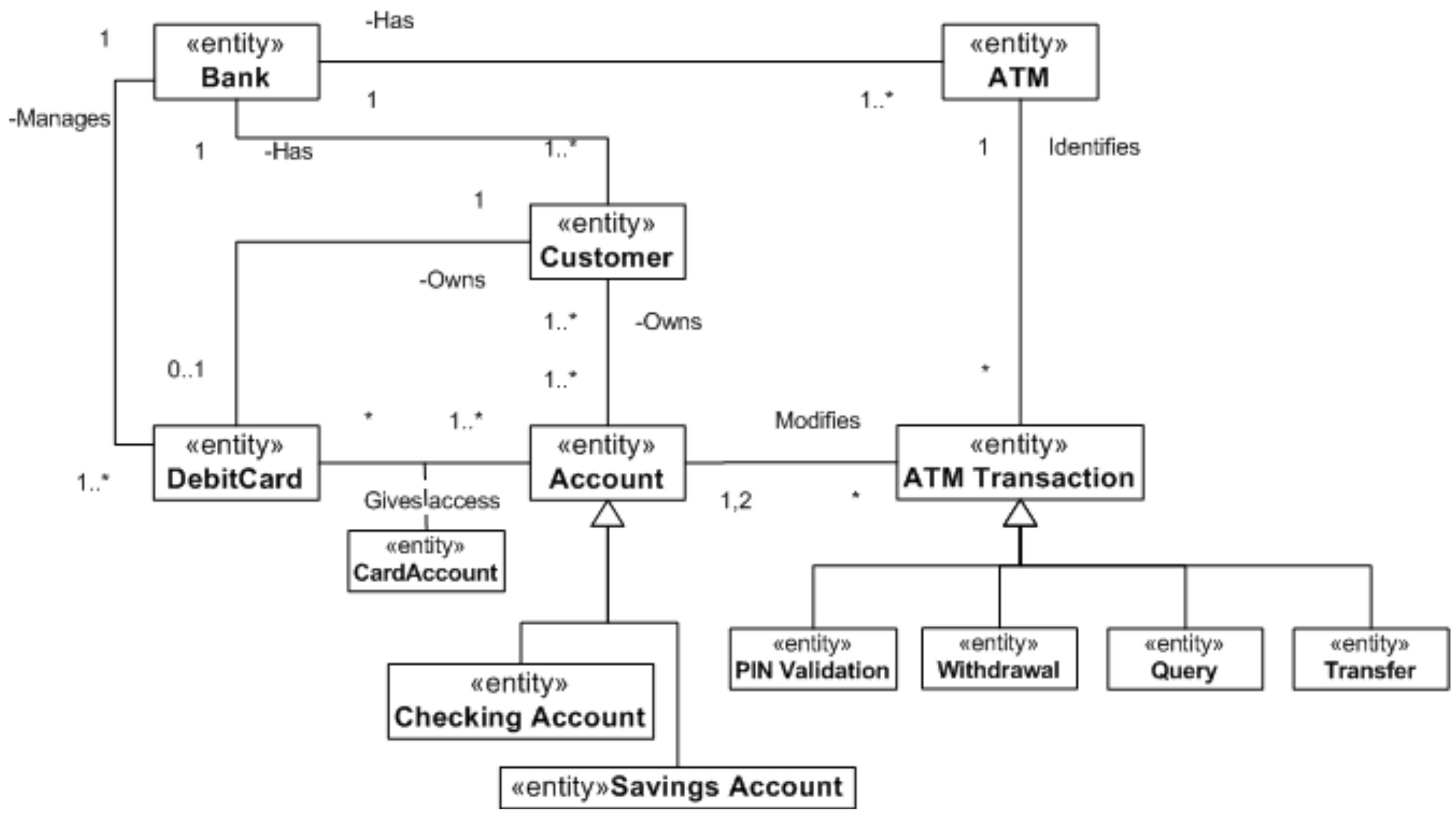
System Context Example (8.6)



Static Modeling – Entity Classes (8.7)

- Entity Classes are long-lived, data-intensive classes.
- Store persistent data that is used by multiple use-cases.
- Often mapped to database and storage mechanisms in the design phase.

Entity Classes Example (8.7)



Summary (8.8)

- Static Models are depicted on class diagrams.
- Classes may be related by:
  - Association
  - Aggregation/Composition
  - Generalization/Specialization
- COMET Emphasizes modeling:
  - Physical classes
  - System context
  - Entity classes