Association, aggregation, and composition (I)

• The existence of a relationship between two classes of the same conceptual granularity
  – The existence of one without the other still makes sense
  – Example: a PC and a laser printer
Association, aggregation, and composition (II)

- Aggregation is a specialized association the emphasizes whole-part relationship
  - The existence of the part is meaningful only if the whole existed
  - The part maybe sharable by multiple wholes
  - Example: a room and its four walls. A wall is shared by two rooms. Destruction of one room should not destruct the shared wall.
Association, aggregation, and composition (III)

• Composition is a specialized aggregation in which the whole disposes the part
  – The whole creates and destroys the part
  – Although creation can be delayed, destruction is immediate upon destruction of the whole
  – The part is non-sharable to other whole
  – Example: a car and its four doors; a polygon and its vertices; a window and its frames
Implication of design on implementation (I)

- A is associated with B
  - A has a pointer or reference to B
- A aggregates B
  - A has pointer or reference to B
  - A creates B, or is settable on B
  - A may (B not shared by others) or may not delete B (B shared by others) in its destructor
  - Or: B should be ref counted
Implication of design on implementation (II)

• A composes B
  – A has a instance variable of B; A has pointer or reference to B
  – A creates B in constructor or is settable on B (if B a pointer)
  – A destructs B in destructor
Rules of engagement

- Use association for starters; advance to aggregation and composition only when doing so clarifies your design.
- Using association = leaving the design options pertaining to following open:
  - Object creation and destruction
  - Sharing of objects
Ref