Relations Between Objects

In addition to *inheritance* and *implementation* that we've already seen, there are other types of relations between objects that we haven't talked about yet.

UML Association. Professor communicates with students.

Association is a type of relationship in which one object uses or interacts with another. In UML diagrams the association relationship is shown by a simple arrow drawn from an object and pointing to the object it uses. By the way, having a bi-directional association is a completely normal thing. In this case, the arrow has a point at each end.

In general, you use an association to represent something like a field in a class. The link is always there, in that you can always ask an order for its customer. It need not actually be a field, if you are modeling from a more interface perspective, it can just indicate the presence of a method that will return the order's customer.

UML Dependency. Professor depends on salary.

Dependency is a weaker variant of association that usually implies that there's no permanent link between objects. Dependency typically (but not always) implies that an object accepts another object as a method parameter, instantiates, or uses another object. Here's how you can spot a dependency between classes: a dependency exists between two classes if changes to the definition of one class result in modifications in another class.

UML Composition. University consists of departments.

Composition is a "whole-part" relationship between two objects, one of which is composed of one or more instances of the other. The distinction between this relation and others is that the component can only exist as a part of the container. In UML the composition relationship is shown by a line with a filled diamond at the container end and an arrow at the end pointing toward the component.

While we talk about relations between objects, keep in mind that UML represents relations between *classes*. It means that a university object might consist of multiple departments even though you see just one “block” for each entity in the diagram. UML notation can represent quantities on both sides of relationships, but it’s okay to omit them if the quantities are clear from the context.

UML Aggregation. Department contains professors.

Aggregation is a less strict variant of composition, where one object merely contains a reference to another. The container doesn’t control the life cycle of the component. The component can exist without the container and can be linked to several containers at the same time. In UML the aggregation relationship is drawn the same as for composition, but with an empty diamond at the arrow’s base.