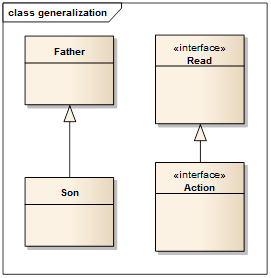
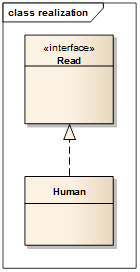
[**UML中常见关系详解（泛化、实现、依赖、关联、组合、聚合）**](https://www.cnblogs.com/jiyuqi/p/4571543.html)

UML中类与类，已经类与接口，接口与接口的关系有：泛化（generalization），关联（association），依赖（dependency），实现（realization）这几种。

泛化（generalization）关系时指一个类（子类、子接口）继承另外一个类（称为父类、父接口）的功能，并可以增加它自己新功能的能力，继承是类与类或者接口与接口最常见的关系，在Java中通过关键字extends来表示。

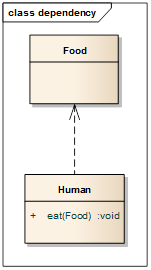


实现（realization）是指一个class实现interface接口（一个或者多个），表示类具备了某种能力，实现是类与接口中最常见的关系，在Java中通过implements关键字来表示。



依赖（dependency）关系也是表示类与类之间的连接，表示一个类依赖于另外一个类的定义，依赖关系时是单向的。简单理解就是类A使用到了类B，这种依赖具有偶然性、临时性，是非常弱的关系。但是类B的变化会影响到类A。举个例子，如某人要过河，则人与船的关系就是依赖，人过河之后，与船的关系就解除了，因此是一种弱的连接。在代码层面，为类B作为参数被类A在某个方法中使用。

在java中，依赖表现为：局部变量，方法中的参数和对静态方法的调用。



关联（association）关系表示类与类之间的连接，它使得一个类知道另外一个类的属性和方法。

关联可以使用单箭头表示单向关联，使用双箭头或者不适用箭头表示双向关联，不建议使用双向关联，关联有两个端点，每个端点可以有一个基数，表示这个关联的类可以有几个实例。

0..1 表示可以有0个或者1个实例

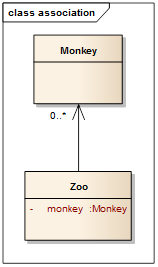
0..\* 表示对实例的数目没有限制

1     表示只能有一个实例

1..\* 表示至少有一个实例

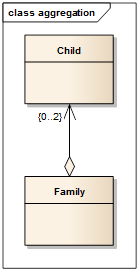
关联关系体现的是两个类，或者类与接口之间的强依赖关系，这种关系很强烈，比依赖更强，不是偶然性的，也不是临时性的，而是一种长期性，相对平等的关系，表现在代码层面，为被关联的类B以类属性的形式出现在类A中，也可能是关联类A引用了被关联类B的全局变量。

在Java中，关联关系是使用实例变量来实现的



聚合（aggregation）是关联关系的特例，是强的关联关系，聚合是整个与个体的关系，即has-a关系，此时整体和部分是可以分离的，他们具有各自的生命周期，部分可以属于多个对象，也可以被多个对象共享；比如计算机和CPU，公司与员工的关系；在代码层面聚合与关联是一致的，只能从语义上来区分。

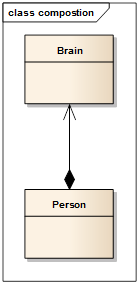
聚合关系也是使用实例变量来实现的，在java语法上区分不出关联和聚合，关联关系中类出于一个层次，而聚合则明显的在两个不同的层次。



组合（compostion）也是关联关系的一种特例，体现的是一种contain-a关系，比聚合更强，是一种强聚合关系。它同样体现整体与部分的关系，但此时整体与部分是不可分的，整体生命周期的结束也意味着部分生命周期的结束，反之亦然。如大脑和人类。

体现在代码层面与关联时一致的，只能从语义来区分。

组合与聚合几乎完全相同，唯一区别就是对于组合，“部分”不同脱离“整体”单独存在，其生命周期应该是一致的。



总结：

主要是关联关系的细化需要注意强弱，由若到强分别是 依赖 < 关联 < 聚合 < 组合