**1．Ppt11:**

[耦合性](http://baike.baidu.com/view/1731634.htm)是程序结构中各个模块之间相互关联的度量。它取决于各个模块之间的接口的复杂程度、调用模块的方式以及哪些信息通过接口。耦合可以分为以下几种，它们之间的[耦合度](http://baike.baidu.com/view/1599212.htm)由高到低排列如下：

（1） 内容耦合。当一个模块直接修改或操作另一个模块的数据时，或一个模块不通过正常入口而转入另一个模块时，这样的耦合被称为内容耦合。内容耦合是最高程度的耦合，应该避免使用之。

（2） 公共耦合。两个或两个以上的模块共同引用一个全局数据项，这种耦合被称为公共耦合。在具有大量公共耦合的结构中，确定究竟是哪个模块给全局变量赋了一个特定的值是十分困难的。

（3） 外部耦合 。一组模块都访问同一全局简单变量而不是同一全局数据结构，而且不是通过[参数](http://baike.baidu.com/view/327406.htm)表传递该全局变量的信息，则称之为外部耦合。

（4） 控制耦合 。一个模块通过接口向另一个模块传递一个控制[信号](http://baike.baidu.com/view/54338.htm)，接受信号的模块根据信号值而进行适当的动作，这种耦合被称为控制耦合。

（5） 标记耦合 。若一个模块A通过接口向两个模块B和C传递一个公共参数，那么称模块B和C之间存在一个标记耦合。

（6） 数据耦合。模块之间通过参数来传递数据，那么被称为数据耦合。数据耦合和最低的一种耦合形式，系统中一般都存在这种类型的耦合，因为为了完成一些有意义的功能，往往需要将某些模块的输出数据作为另一些模块的输入数据。

（7） 非直接耦合 。两个模块之间没有直接关系，它们之间的联系完全是通过主模块的控制和调用来实现的。

**总结**：耦合是影响软件复杂程度和设计质量的一个重要因素，在设计上我们应采用以下原则：如果模块间必须存在耦合，就尽量使用数据耦合，少用控制耦合，限制公共耦合的范围，尽量避免使用内容耦合。

内聚（Cohesion）是一个模块内部各成分之间相关联程度的度量。

内聚的概念是Constantine、Yourdon、Stevens等人提出的。按他们的观点，把内聚按紧密程度从低到高排列次序为偶然内聚、[逻辑内聚](http://baike.baidu.com/view/431329.htm)、时间内聚、[过程内聚](http://baike.baidu.com/view/1482386.htm)、通信内聚、信息内聚、[功能内聚](http://baike.baidu.com/view/431335.htm)。但是紧密程度的增长是非线性的。偶然内聚和[逻辑内聚](http://baike.baidu.com/view/431329.htm)的模块联系松散，后面几种内聚相差不多，[功能内聚](http://baike.baidu.com/view/431335.htm)一个功能、独立性强、内部结构紧密，是最理想的内聚。

内聚按强度从低到高有以下几种类型：

偶然内聚：如果一个模块的各成分之间毫无关系，则称为偶然内聚，也就是说模块完成一组任务，这些任务之间的关系松散，实际上没有什么联系。

逻辑内聚：几个逻辑上相关的功能被放在同一模块中，则称为逻辑内聚。如一个模块读取各种不同类型外设的输入。尽管逻辑内聚比偶然内聚合理一些，但逻辑内聚的模块各成分在功能上并无关系，即使局部功能的修改有时也会影响全局，因此这[类模块](http://baike.baidu.com/view/675607.htm)的修改也比较困难。

时间内聚：如果一个模块完成的功能必须在同一时间内执行（如[系统初始化](http://baike.baidu.com/view/811790.htm)），但这些功能只是因为时间因素关联在一起，则称为时间内聚。

通信内聚：如果一个模块的所有成分都操作同一数据集或生成同一数据集，则称为通信内聚。

顺序内聚：如果一个模块的各个成分和同一个功能密切相关，而且一个成分的输出作为另一个成分的输入，则称为顺序内聚。

功能内聚：模块的所有成分对于完成单一的功能都是必须的，则称为功能内聚。

信息内聚：模块完成多个功能，各个功能都在同一数据结构上操作，每一项功能有一个唯一的入口点。这个模块将根据不同的要求，确定该模块执行哪一个功能。由于这个模块的所有功能都是基于同一个数据结构（[符号表](http://baike.baidu.com/view/5044878.htm)），因此，它是一个信息内聚的模块。

内聚与耦合

[内聚](http://baike.baidu.com/view/553710.htm)标志一个模块内各个元素彼此结合的紧密程度，它是信息隐蔽和局部化概念的自然扩展。[内聚](http://baike.baidu.com/view/553710.htm)是从功能角度来度量模块内的联系，一个好的内聚模块应当恰好做一件事。它描述的是模块内的功能联系。耦合是软件结构中各模块之间相互连接的一种度量，耦合强弱取决于模块间接口的复杂程度、进入或访问一个模块的点以及通过接口的数据。 程序讲究的是低耦合，高内聚。就是同一个模块内的各个元素之间要高度紧密，但是各个模块之间的相互依存度却要不那么紧密。

内聚和耦合是密切相关的，同其他模块存在高耦合的模块意味着低内聚，而高内聚的模块意味着该模块同其他模块之间是低耦合。在进行软件设计时，应力争做到高内聚，低耦合。

由于结构化方法将过程和数据分离为相互独立的实体, 程序员在编程时必须时刻考虑到所要处理的数据的格式。对于不同的数据格式做相同的处理或对于相同的数据格式做不同的处理都需要编写不同的程序, 而且往往不能对数据的安全性进行有效的控制。如果程序进行扩充或升级, 也需要大量修改函数, 因此结构化程序的可重用性不好。

但它仍是一种[面向过程](http://wenwen.soso.com/z/Search.e?sp=S%E9%9D%A2%E5%90%91%E8%BF%87%E7%A8%8B&ch=w.search.yjjlink&cid=w.search.yjjlink)的程序设计方法，它把数据和处理数据的过程分离为相互独立的实体。当数据结构改变时，所有相关的处理过程都要进行相应的修改，每一种相对于老问题的新方法都要带来额外的开销，程序的可重用性差。

由于[图形用户界面](http://wenwen.soso.com/z/Search.e?sp=S%E5%9B%BE%E5%BD%A2%E7%94%A8%E6%88%B7%E7%95%8C%E9%9D%A2&ch=w.search.yjjlink&cid=w.search.yjjlink)的应用，程序运行由顺序运行演变为[事件驱动](http://wenwen.soso.com/z/Search.e?sp=S%E4%BA%8B%E4%BB%B6%E9%A9%B1%E5%8A%A8&ch=w.search.yjjlink&cid=w.search.yjjlink)，使得软件使用起来越来越方便，但开发起来却越来越困难，对这种软件的功能很难用过程来描述和实现，使用面向过程的方法来开发和维护都将非常困难