为了对一个对象的生命周期进行扩展，使得其生命周期超过调用它的客户，DCOM 提供了一个 moniker 作为这个对象的持久性引用。这个持久性引用 moniker 被保存在磁盘上，这个 moniker 包含了重新构造这个对象所需的必要信息，例如这个对象的最后一个客户退出时这个对象的状态。DCOM 提供了各种不同类型的 moniker，其中一个重要的类型是文件 moniker。对于文件 moniker，它包含了一个本地的文件名字，这个文件用于构造对应的对象。另外它还包含一个类对象的标识符 CLSID，这个类对象通过对应的类实际创建一个对象。

一个文件 moniker 提供了一个 BindToObject 操作，客户通过调用这个操作与这个 moniker 对应的对象结合。表 1 描述了客户通过文件 moniker 和一个对应的持久性对象进行结合的处理过程。

**表 1　客户通过文件 moniker 和持久性对象结合**

| 步骤 | 参与者 | 描述 |
|---|---|---|
| 1 | 客户 | 在一个 moniker 上调用 BindMoniker |
| 2 | Moniker | 查找对应的 CLSID 并指示 SCM 创建一个对象 |
| 3 | SCM | 装入类对象 |
| 4 | 类对象 | 创建一个对象并返回指向一个 moniker 的接口指针 |
| 5 | Moniker | 指示对象装载以前保存的状态 |
| 6 | 对象 | 从一个文件装载它的状态 |
| 7 | Moniker | 返回一个指向对象的接口指针给客户 |

从上面的论述中我们认识到，moniker 是持久性对象。在 DCOM 中，每个 moniker 提供了一个接口，这个接口包含了一些方法，这些方法将对象的内容保存在磁盘上。在许多情况下，应用程序告诉 moniker 将自己作为文件进行保存。以后当重新读这个文件时，这个 moniker 可以重新构造为一个真正的 DCOM 对象。

客户和一个 moniker 结合后，该 moniker 会将其对应的对象在客户机上的运行对象表 ROT（Running Object Table）中进行注册。当另外一个客户要使用这个 moniker 同这个对象结合时，这个 moniker 首先在 ROT 中查找这个对象，如果发现这个对象已经被创建，客户将直接同这个对象结合。这样，对象可以被同一个机器上的不同进程所共享。

除了文件 moniker 之外，DCOM 还支持其他类型的 moniker，如 URL moniker、类 moniker 等。Moniker 的实现细节可见文献。