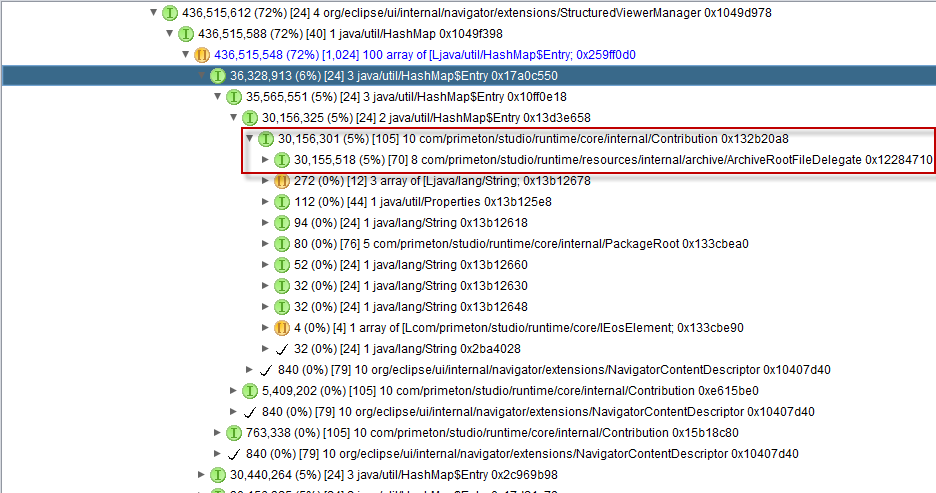
# 上海银行IBS产品编译问题

## DUMP文件分析

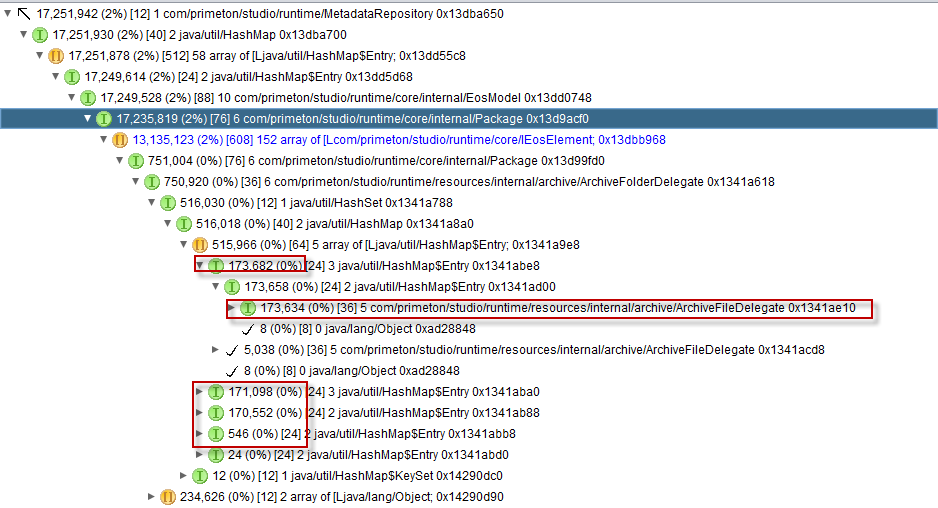
展开dump文件：



看了dump文件，内存主要被ArchiveRootFileDelegate文件占用了，大概有18个这样的文件，每个30M左右。

ArchiveRootFileDelegate是用来解析ecd、zip、jar等压缩文件的，说明此处有较大的文件被解析。

继续展开dump文件：



发现每个构件包都加载了多个压缩文件，每个大约170k左右，经过分析应该是图片资源。

### 问题一：项目引用库加载问题

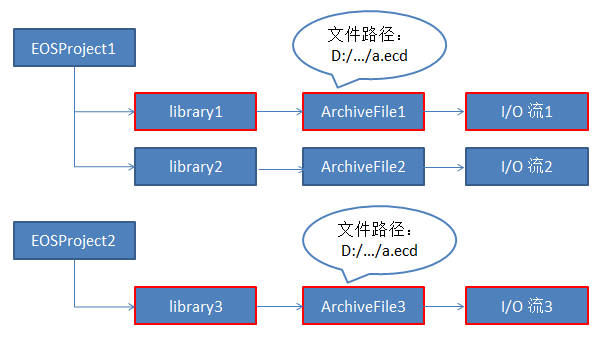
目前EOS引用库加载模式：一个EOS项目包含多个引用库配置，每个引用库包含若干个ecd资源，由引用库ILibrary对象控制ecd资源加载和缓存（用时间戳控制资源更新）。

优点：

每个引用库配置只加载一次

缺点：

项目较多时，每个项目下的引用库尽管只加载一次，占用内存也相当大。

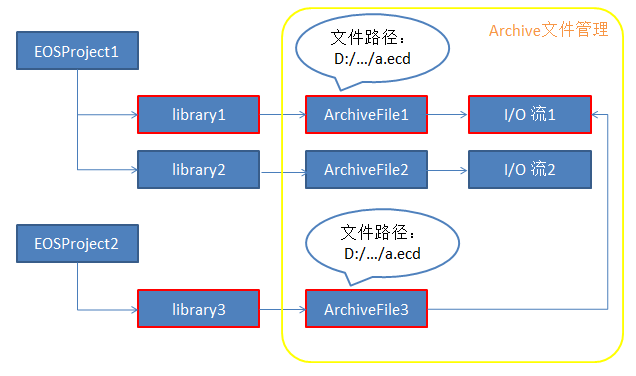


如上图所示：

Project1与Project2当中的library1对象和library3对象使用ecd文件物理路径完全相同，但是I/O操作执行了两次。也就是说，在我们的工作区当中尽管许多项目配置的引用库路径完全一样，但是在使用过程中依然会加载多次。

#### 【解决方案一】

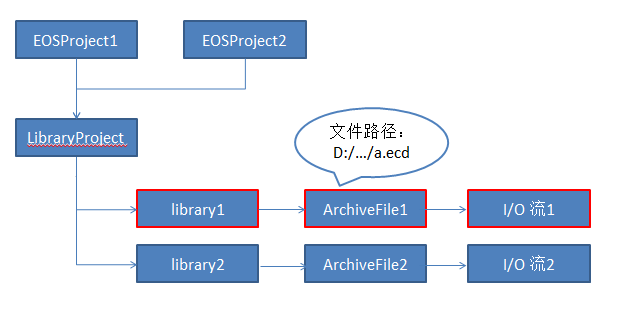
如下图所示：



增加一个压缩文件管理，保证I/O流操作唯一。

#### 【解决方案二】

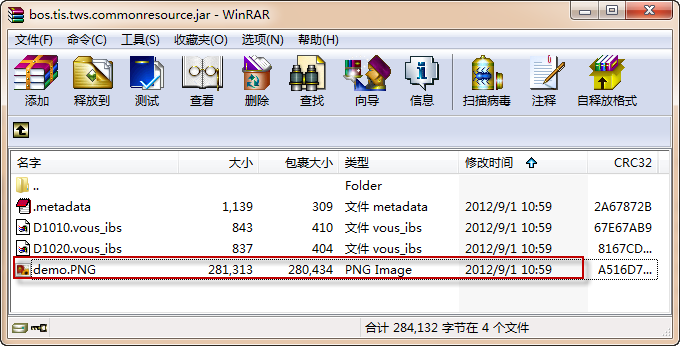
参考工行定制项目设计，增加组件库项目，项目不再独立包含library对象。具体内容参考工行定制项目设计文档。如下图：



#### 【解决方案三】

构件包项目(Bundle)方式管理，所有项目都以构件包形式出现（部分构件包为压缩文件放在TargetPlatform当中），所有bundle均唯一。

### 问题二：ECD中的构件包里包含较大的图片资源



上海银行提供的基础构件库ECD当中，所有构件包下都包含多个200K左右的图片文件，这次性能问题就处在这些大文件上，如果这样的文件太多，尽管解决了问题一，依然有可能导致性能瓶颈和堆栈溢出。