## 引擎性能优化

由于原来cflow引擎在解析和调用方面很多xpdl对象都加载到内存中，随着更多的设计过程发起，内存的使用量也更大，对系统的压力也更大，就会造成系统运行效率降低，根据论证主要可以从下面几点进行优化：

1引擎流转

2应用服务器启动加载

3解析机制

4内存调用机制

5内存销毁机制

Cflow内存应用机制对比

|  |  |  |
| --- | --- | --- |
|  | 原来 | 优化后 |
| 引擎 | 解析转移线与事物对应关系 | 仅存储转移线，简化TRANSACTION对应关系 |
| 应用服务器启动加载 | 加载所有相关xpdl文件 | 不加载xpdl文件，使用时加载 |
| 解析机制 | 保存xpdl文件后，对xpdl文件进行解析比且放到内存  PeraXpdlGlobalVariable.*getXpdlHash*().put(  peraXpdl.getPackageId() + peraXpdl.getVersionNumber(),  peraXpdl); | 调用redis内存服务器，将解析好的xpdl对象放到redis缓存服务器中 |
| 调用机制 | 调用时通过version和pkgId从内存中读取xpdl对象 | 根据模版id从缓存服务器中调用 |
| 销毁机制 | 使用完以后不进行任何操作，仍然保留在内存中，内存随着流程实例的发起而增加 | 使用完以后手动销毁内存中的xpdl对象 |

优化调用方式图见EA

执行环境调度引擎接口优化方案：

1. 内部接口(作为提供方)

说明：提供给分布式Agent获得任务信息、更新任务执行状态、更新任务结束状态、更新agent节点的工作状态(空闲/负荷工作中)、 注册agent、更新工作节点的活动时间、更新节点并发工作任务数、更新任务失败原因、获得客户端默认配置信息

处理方案：采用模块服务化，在高并发处理情况下，DRM服务器动态横向扩展，减少接口调用时间消耗；同时针对该接口小数据量通信，接口协议采用NIO长链接，相对其他协议（webservice,rmi,http-json,hession）在调用性能上有较大提升。

2、外部接口(作为提供方)

说明：该接口主要提供给流程引擎，主要功能为，接受远程任务和中止远程任务；

处理方案：采用阻塞队列的方式实现接口，调用方发送异步消息完成调用过程，这样减少调用方的时间消耗，并且DRM服务器集群环境下能够使计算压力均衡化。