**解析器程序设计说明书**

**开发人员:**杜源峰，罗建辉，李昕祈

**开发日期:**2012年7月-9月

1. **程序架构及功能：**

解析器的架构如下图所示，包含编码查询、智能缓存、通用查询、配置、日志五大模块，各个模块的功能如下所示：



1. **编码查询模块：**负责编码的转换和查询，会调用其他三个模块的功能。编码转换过程包括：
2. 把SID转换成SID域名；
3. 通过DNS查询SID域名，获得转换RID的正则表达式；
4. 用正则表达式转换RID，把RID转换成RID域名；
5. 通过DNS查询RID域名，得到资源的地址信息。

通过SID可以根据不同的编码标准转换各种编码，实现兼容多编码。

1. **智能缓存模块：**分为普通缓存和预测缓存。
2. 普通缓存：解析器在客户端内存中开辟一段空间，保存应用程序每次请求查询的结果，当应用程序再次请求相同的编码解析时直接从缓存中返回结果，不用重复原来的转换域名操作和DNS查询。
3. 智能预测缓存：采用数据挖掘算法，对解析器生成的日志进行挖掘，分析应用程序使用该解析器的习惯，得到应用程序使用解析器的行为模式，预测应用程序下一个可能查询的物品编码，提前解析该编码，把结果放入缓存，当应用程序发来该编码的解析请求时可直接从缓存中取得结果。
4. **通用查询模块：**负责与DNS解析器交互的通用查询。
5. **配置模块：**记录解析器一些必要的配置，解析器启动时可以从该模块中获取配置信息，如转换SID需要的正则表达式等。
6. **发现查询模块：**用于发现查询，与编码查询的主要区别是将所有DS类型的NAPTR记录全部返回，供发现功能使用。
7. **日志模块：**记录应用程序使用解析器的解析历史数据，预测缓存通过挖掘日志得到用户的使用习惯。

**已完成功能模块：**编码查询模块，通用查询模块，配置模块，日志模块，发现查询模块

**待完成功能：**智能缓存模块

**开发环境: MyEclipse8.5**

**二、主要函数功能说明：**

**编码查询：**

函数原型String resolver\_getInfoByCode (String SID , String RID)

函数功能：根据物品编码的SID和RID对编码进行解析返回解析的结果（URL或IP）

**配置：**

函数原型：void SysConfig (String path)

函数功能：构造函数，设置配置文件的路径，在缺省路径的状况下采用默认配置

设置配置文件路径，默认的配置文件路径位于程序所在目录的jar文件夹中（采用默认路径时只需要写”文件名. properties就行”），配置文件的内容如下：

DELIMETOR=.

SIDName=oid

projectDomainName=tnsroot.cn

regexForRegex=!\^([^!]+)!([^!]+)! 用来匹配regex域的正则表达式

regexForDomain=\$([\d]+)\. 用来匹配domain域的正则表达式

urlRegex=!\^\.\\*\$!([^!]+)! 用来匹配url的正则表达式

SID\_SERVER=218.241.108.59

RID\_SERVER=218.241.108.63

SERVICE\_TYPE=ONS

DOT=00101110

codecount=4

**发现查询：**

函数原型：Vector<String> getAllRecordSContent (String SID , String RID)

函数功能：返回所有的NAPTR记录

**三、程序运行实例**

**通用查询**

SysConfig conf = new SysConfig("SysConfig.properties") ; //设置配置文件路径

SimpleResolver resolver = new SimpleResolver(conf); //实例化通用查询类

String url = resolver.resolver\_getInfoByCode(SID, RID) //调用通用查询接口实现查询

**发现查询**

SysConfig conf = new SysConfig("SysConfig.properties"); //设置配置文件路径FindResolver resolver = new FindResolver(conf); //实例化通用查询类

resolver.setFindServerAddr("218.241.108.63"); //设置用于RID查询的服务器地址

resolver.setFindServiceType("DS"); //设置发现服务所需查询的记录类型

Vector<String> vec=resolver.getAllRecordSContent(SID, RID)

//调用发现查询接口查询

**界面实现实例**

初始界面



示例ID

SID：100010111001000000101110100111000010111000010000000000001000

RID：0001001000111001000110000110000110010011100010111001100111101101

点击“Search”进行通用查询，结果如下图



点击“Finder Search”进行发现查询，结果如下图：



**四、普通缓存模块实现**

普通缓存替换策略LRU算法和TTL值：如果缓存在已无效数据，则选择无效数据替换；否则，选择最久未被访问的数据进行替换.

具体实现采用了三个结构：HashMap、双向链表和堆。

缓存的存储存单元封装了key、value和到期时间，HashMap存放key与存储单元引用的映射关系，双向链表按访问顺序组织缓存单元，堆则根据到期时间排列缓存单元的引用。

替换算法:

输入：key、expiredTime

将新元素插入HashMap、双链表、堆中

将替换元素从HashMap、双链表、堆中册除待替换元素

待替换元素设定为查找到的结点

待替换元素设定该过期元素

待替换元素设定链表尾节点

Y

N

查找key是否存在于缓存中

查看堆顶元素是否过期