**不良信息过滤模块任务具体设计和内核API调用说明**

——胡杨 2014.6.23

**1．操作简述：**

在原始情报库全文索引建立完成的情况下，数据库表tbl\_index中会存有全部内容的索引信息。

准备分词器，分词器通过事先调用produceDiyAnalyzer()【1】获得，传入参数：stopWordList：停用词List列表（查询数据库表tbl\_stopWords表获得，ps：注意转化成List数据类型），从数据表tbl\_illegalWords查询得到所有的敏感词，调用内核方法phraseQuerySearcher()【2】，调用查询方法后，返回值：List<Map<String, Object>>结果键值对，这里记为firstSearchResMap。

每次分批检索任务结束，获得结果键值对后，对键值对进行过滤处理，首先要对结果键值对存储结果映射缓存（持久化），mongodb中数据库searchMapPool存储这部分内容，对于每个检索结果，调用handleNewSearchRes()【3】处理查询结果和关键词的映射对。调用结束后，mongodb对应表中会存有查询关键词和结果对应的映射对。集合名称为对应corpus表中的集合名称。

检索任务全部完成后，mongodb.searchMapPool表中存放了所有检索结果和对应关键词的映射对，关键词集合用“；”隔开，获得该键值对的postUrlMD5和keyWords以及所在的collectionName可以对原是情报的内容以关键词为准进行过滤，调用execSensitiveContentFilter()【4】方法开展过滤操作，返回结果为过滤后的结果键值对List<Map<String, Object>>，这里将此键值对称作为sensitiveContentNodeMap。

将sensitiveContentNodeMaps键值对中的信息取出，形成文本文件，文件名为nodeId + collectionName，文件内容为content。并在数据库表tbl\_corpusText表中存入相应的元信息。

模块完成后，调用dropOldSearchMapPool()【5】方法删除过期的检索结果映射对信息。

**2．容器说明：**

firstSearchResMap：｛key1：postUrlMD5，value1：对应原始语料corpus的MD5唯一标识码｝；｛key2：collectionName，value2：对应原始语料所属corpus的集合名称｝

sensitiveContentNodeMap：｛key1：nodeId，value1：过滤语料唯一标识的复合MD5编码｝；｛key2：content，value2：过滤语料的内容｝；｛key3：collectionName，value2：对应原始语料corpus所属的集合名称｝

**3．内核方法说明：**

【1】：分词器工厂方法：package ims.nlp.lucene.analyzer；class AnalyzerFactory；传入：List<String> stopWordList，停用词列表，返回：兼容lucene的分词器。

【2】：原是情报全体内容检索方法：package ims.nlp.lucene.search；class SearchInLocalContent；传入：value，关键词字符串（可多个关键词同时传入），num，查询结果上限，analyzer，兼容lucene的分词器。

【3】：初次检索结果处理方法（形成结果mongodb映射对）：package ims.nlp.lucene.search；class SearchMapBufferManagment；传入：postUrlMD5，collectionName，strKeywords，关键字列表字符串（格式要求：key1；key2；key3；…）。

其他选择：1.同名方法；传入：直接传入firstSearchResMap键值对，strKeywords（格式要求同上）2.同级目录下：class SmartSearchMpBufManagment（继承class SearchMapBufferManagment）中的两个同名方法，strKeyWords此时可以传入任何形式的中文字符串，系统会自动分词组装成符合要求的strKeyWords。

【4】：package ims.nlp.lucene.search；class SearchResContentFilter；传入：firstSearchResMap（mongodb，searchMapPool数据库中的键值对）返回sensitiveContentNodeMap。

【5】：package ims.nlp.lucene.search；class SearchMapBufferManagment；删除searchMapPool中所有的集合。

**4．其他：**

在低依次检索得到结果，向searchMapPool映射对数据库中写入的时候，可以考虑使用多线程，后期从searchMapPool取出数据，进行二次检索，写成文本文档和元数据组合的时候，可以考虑使用多线程。

线程的调度自行设计，内核已经考虑了线程的安全，但是为了万无一失，在多线程编写的过程中请谨慎编写。

任务执行者：刘卓然