# 需求分析

## 功能需求概述

* **xml文件解析**

需要解析doxygen工具产生的xml文件，从中解析出复合数据结构：struct、class、union以及简单语法单元，包括宏定义、类型重命名、函数原型、枚举类型

* **建立数据类型全文索引**

默认输入c++内建对象（<signed/unsigned> bool, char, short, int, long, long long <\*> | void \*），需要为复合数据结构建立索引，索引的目标是建立对象的名称及所有别名的关联包括指针别名。并枚举文件中所有的数据类型。构建数据类型索引数据库。

判定字符串是否是内建类型正则表达式：((((signed|unsigned)( )+)?((char)|(int)|(short)|(long)|(long long))|( )\*(\\*)))|(bool)|(void( )\*\\*)

* **建立函数全文索引**

建立函数的索引，需要对函数的返回类型，参数列表建模。

## xml文件解析功能

逆向建模系统主要需要解析的xml文件类型分为三类：

* 全局索引文件（index.xml）
* 复合数据类型描述文件（struct\_n\_e\_t\_\_\_d\_v\_r\_\_\_e\_t\_h\_e\_r\_n\_e\_t.xml）
* 文件语法元素描述文件（test\_8h.xml）

其中全局索引文件旨在枚举全局语法元素（含复合及单一语法单元）。对于复合语法单元，标注其类型，如struct；按定义顺序列举其成员（变量+函数）。

对于其他语法单元，则按照宏、枚举、枚举值、重命名、全局函数顺序列举在文件语法元素描述文件中

## 建立数据类型及函数全文索引

数据类型的全文索引，包括对全局复合数据类型struct、class、union及其成员变量的类型，以及内建对象bool, char, short, int, long, long long建立。

建立数据类型全文索引，主要需要解析：

* **全局索引文件（index.xml）：**

<compound>节点

> refid属性为详细描述信息的文件索引名，可依赖这个名字查询文件

> 当kind 取值属性为struct、class、union，节点表示一个复合数据类型，此时<compound>节点的子节点

|<name>表示数据类型首选名称，

|<member>描述一个复合数据结构的成员。此时<member>节点的kind属性可取{variable, function}，表示成员的类型，其子节点<name>取值则表示成员名称。

|> refid属性为成员索引名，可依赖这个找到详细描述文件中<memberdef> id属性相同的详细描述节点

> 当kind值为file时，除复合数据类型外的语法单元定义均列举在此节点

|<name>为所属文件名

|<member>为文件申明的语法类型，此时kind属性可取{define, {enum, enumvalue}, typedef, function}，其子节点<name>为语法单元名

在全局索引文件中，复合数据类型对成员的描述简略，需要检索对应复合数据类型描述文件。

* **复合数据类型描述文件（struct\_xxx.xml）:**

文件的主体节点为一个<compounddef>

> kind属性为复合数据结构的类型，对应全局索引文件<compound>的kind属性

> id复合类型的索引号

|<compoundname>子节点为复合数据类型名

|<includes>子节点为类型来源文件

|<sectiondef>子节点列举同访问权限的成员

|> kind属性取值{public-attrib, protected-attrib, private-attrib}，struct类型的话默认为public-attrib

||<memberdef>成员描述节点

||> kind对应全局索引文件中<member>节点的kind属性

||> id成员变量索引名，程序可根据这个id找到对应对象的详细定义信息

||> prot访问控制{public, protected, private}

||> static标记是否为静态成员{yes, no}

||| <type>若kind==”variable”则为数据类型，若数据类型是非内建类型名，则使用<ref>节点包裹类型名，可通过其refid属性检索到对应描述信息节点

|||> kindref属性标识节点类型，若数据类型为复合类型则取值compound，若数据类型为内建对象的别名则取值member

||| <definition>节点为成员声明

||| <argsstring>当成员为数组对象，此节点定义数据长度

||| <name>成员名

* **文件语法元素描述文件（test\_8h.xml）：**

在文件语法元素描述文件中，复合数据对象包括，复合数据对象内部定义的其他复合对象都是以<innerclass>节点枚举，其中，

> refid属性为索引号

> prot访问权限

除复合数据的其他语法元素，按照语法类型分别枚举在<sectiondef>节点

> kind取值{ define, enum, typedef, func }

case define:

|其中每一个语法单元表示为一个<memberdef>

|> kind 同父节点kind

|> id 索引名

|| <name> 宏名

|| <initializer> 初始化的值，若检测到宏

case enum:

| <memberdef>一个语法单元

|| <type> 略

|| <name>枚举类型名

||| <enumvalue>枚举元素

|||> id索引名

|||| <name>枚举元素名

|||| <initializer>初始值，可为空

case typedef:

| <memberdef>一个语法单元

|| <type>原类型，若原类型为已经定义的数据类型，则用<ref>描述

|| <definition>定义原型

|| <argsstring>略

|| <name>新类型名

case func:

| <memberdef>一个语法单元

|> static静态函数标识

|> const 常函数标记

|> explicit

|> inline内联

|> virt虚函数

|| <type>返回值类型，已定义的类型<ref >

|| <definition>函数原型（除参数）

|| <argsstring>参数列表

|| <name>函数名

|| <param>一个参数（多个）

||| <type>参数类型，已定义时<ref>

||| <declname>参数名

可以看出，凡是具有id属性的节点均为已经建立索引的语法元素。所以第一步工作是搜索全部xml文件，找出所有定义的带id的语法元素。实际上就是要搜索*复合数据类型描述文件*的<compounddef>节点及其子节点<memberdef>的id，以及*文件语法元素描述文件*的<sectiondef kind ={ define, enum, typedef, func }>的<memberdef>及<enumvalue>的id，并关联语法元素名。这一步的目的是追本溯源，要为所有数据类型建模。非复合类型一定要追溯到内见对象，或者未知对象为止。

然后，搜索全局索引文件将其中复合数据结构按照struct，union，class分类。将非复合对象按照define，enum，typedef，func分类，分别建立索引。