(12) 按照专利合作条约所公布的国际申请

(19) 世界知识产权组织
国 际 局

(43) 国际公布日
2008 年5 月29 日 (29.05.2008)

PCT

(10) 国际公布号
WO 2008/061457 A1

(51) 国际专利分类号:
G06F 17/21 (2006.01)

(21) 国际申请号: PCT/CN2007/070132

(22) 国际申请日: 2007 年6 月19 日 (19.06.2007)

(25) 申请语言: 中文

(26) 公布语言: 中文

(30) 优先权:
200610114649.1
2006 年11 月20 日 (20.11.2006) CN

(71) 申请人 (对除美国外的所有指定国): 北京书生国际信息技术有限公司(BEIJING SURSEN CO., LTD) [CN/CN]; 中国北京市海淀区紫竹院路81号北方地产大厦5层, Beijing 100089 (CN)。

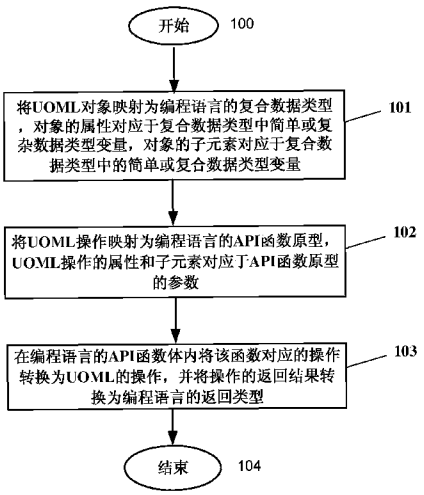
(72) 发明人; 及

(75) 发明人/申请人 (仅对美国): 王东临(WANG,

[见续页]

(54) **Title:** A METHOD FOR PACKAGING THE UOML TO APPLICAIONS PROGRAMMING INTERFACE

(54) **发明名称:** 一种将UOML封装成应用程序编程接口的方法



100 START
101 MAPPING AN UOML OBJECT TO A COMPLEX DATA TYPE OF A PROGRAMMING LANGUAGE, THE ATTRIBUTES OF THE OBJECT CORRESPONDING TO THE SIMPLE OR COMPLICATED DATA TYPE VARIANTS IN THE COMPLEX DATA TYPE, THE SUBELEMENTS CORRESPONDING TO THE SIMPLE OR COMPLEX DATA TYPE VARIANTS IN THE COMPLEX DATA TYPE
102 MAPPING AN UOML OPERATION TO AN API FUNCTION SCRIPT OF THE PROGRAMMING LANGUAGE, THE ATTRIBUTES AND SUBELEMENTS OF THE UOML OPERATION CORRESPONDING TO THE PARAMETERS OF THE API FUNCTION SCRIPT
103 IN THE API FUNCTION BLOCK OF THE PROGRAMMING LANGUAGE, CONVERTING THE OPERATION CORRESPONDING TO THE FUCTION INTO THE OPERATION OF UOML, AND CONVERTING THE RETURNED RESULT OF THE OPERATION INTO THE RETURNED TYPE OF THE PROGRAMMING LANGUAGE
104 END

(57) **Abstract:** A method for packaging an UOML into an API comprises the following steps: mapping an UOML object to a complex data type of a programming language; mapping an UOML operation to an API function script of the programming language and the object of the UOML operation corresponding to the parameters of the API function script; according to the mapping defined in the above steps, in the API function block of the programming language, converting the operation corresponding to the function into the UOML operation and converting the returned result of the operation into the returned type of the programming language.

(57) **摘要:**

一种将 UOML 封装成 API 的方法，包括如下步骤：将 UOML 对象映射为编程语言的复合数据类型；将 UOML 操作映射为编程语言的 API 函数原型，UOML 操作的对象对应于 API 函数原型的参数；根据上述步骤中定义的映射，在编程语言的 API 函数体内将函数对应的操作转换为 UOML 的操作，并将操作的返回结果转换为编程语言的返回类型。

WO 2008/061457 A1

Donglin) [CN/CN]; 中国北京市海淀区紫竹院路81号北方地产大厦5层, Beijing 100089 (CN)。 **邹开红 (ZOU, Kaihong)** [CN/CN]; 中国北京市海淀区紫竹院路81号北方地产大厦5层, Beijing 100089 (CN)。

(74) 代理人：北京德琦知识产权代理有限公司(DEQI INTELLECTUAL PROPERTY LAW CORPORATION); 中国北京市海淀区知春路1号学院国际大厦7层, Beijing 100083 (CN)。

(81) 指定国 (除另有指明，要求每一种可提供的国家保护)：AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW。

(84) 指定国 (除另有指明，要求每一种可提供的地区保护)：ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), 欧亚 (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), 欧洲 (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG)。

本国际公布：

— 包括国际检索报告。

一种将UOML封装成应用程序编程接口的方法

技术领域

本发明涉及电子文档处理技术，尤指一种将非结构化操作操作标记语言（UOML）封装成应用程序编程接口（API）的方法。

发明背景

UOML 规范是一系列以“动作+对象”格式定义的、通过可扩展标记语言（XML）描述的文档库系统命令。本申请人在中国专利申请号为CN200510131641.1 的专利申请说明书中，对其进行了详细的说明。由于XML 具有跨平台、跨语言的能力，所以，UOML 规范的提出，解决了文档库系统命令本身的跨平台、跨语言交换的问题。但是，在实际应用过程中，对文档库的操作一般通过编程语言实现的代码控制，因此，在代码中需要实现对 UOML XML 文本的解析和处理。如果每个应用开发人员在各自设计的代码中实现对 UOML XML 文本的解析和处理，会造成工作量大且开发效率低下等问题。

发明内容

本发明要解决的一个技术问题是提供一种方法，通过该方法可以将UOML 封装成不同编程语言的 API，以提高文档库系统应用开发者的开发效率。

本发明提供的将 UOML 封装成应用程序编程接口 API 的方法，包括如下步骤:

A，将 UOML 对象映射为编程语言的复合数据类型;

B，将 UOML 操作映射为编程语言的 API 函数原型，UOML 操作的对象对应于 API 函数原型的参数;

C，根据步骤 A、B 中生成的映射，在编程语言的 API 函数体内将该函数对应的操作转换为 UOML 的操作，并将操作的返回结果转换为编程语言的返回类型。

所述步骤 A 具体包括:

所述 UOML 对象的属性对应于复合数据类型中简单数据类型变量或复杂数据类型变量;

所述 UOML 对象的子元素对应于复合数据类型中简单或复合数据类型变量。

所述编程语言为面向对象的编程语言;

步骤 A 中所述的编程语言中的复合数据类型为类，所述 UOML 对象的属性和子元素对应于类的成员变量;

步骤 B 中所述的编程语言的 API 函数原型为对应类中的成员函数。

所述的面向对象的编程语言包括 C++、Object-C、Delphi、Java、Python 或 Ruby。

所述编程语言为非面向对象的编程语言;

步骤 A 中所述的编程语言中的复合数据类型为结构;

步骤 B 中所述的编程语言的 API 函数原型为全局函数。

所述的非面向对象的编程语言包括 C、TCL、Pascal 或 Perl。

所述编程语言具有和 C 或者 C++的接口;

步骤 C 中所述转换的方法为:通过调用 C++或 C 语言封装的 UOML 的 API 函数来实现向 UOML 操作的转换。

所述的编程语言为 Java、TCL、Perl、Python 或 Ruby。

通过本发明提供的方法，可以实现将 UOML 封装成编程语言的 API。开发人员在一编程语言上开发对文档库系统的应用时，可以直接调用生成的封装好的该编程语言的 API，从而省去了大量的解析 UOML 的工作

量，提高了开发的效率。

附图简要说明

图 1 是本发明将 UOML 封装成编程语言的 API 的方法的流程图;

图 2 是本发明将 UOML 封装成面向对象语言的 API 的方法的流程图;

图 3 是本发明将 UOML 封装成非面向对象语言的 API 的方法的流程图;

图4是本发明编程语言的API封装层次的示意图。

实施本发明的方式

本发明利用 UOML 规范的“动作+对象”的格式特点，以及 UOML 对象和编程语言中的类或结构的内在对应关系，提供一种可以将 UOML 封装为不同的编程语言的 API 的方法，从而利用不同编程语言对文档库系统进行应用开发时可以直接调用该语言为 UOML 提供的 API，提高开发人员的开发效率。

图 1 是本发明将 UOML 封装成编程语言的 API 的方法的流程图，如图 1 所示，本发明方法包括如下步骤:

步骤 101，将 UOML 对象映射为编程语言的复合数据类型。其中，UOML 对象的属性对应于复合数据类型中简单数据类型变量或复杂数据类型变量，UOML 对象的子元素对应于复合数据类型中简单或复合数据类型变量。

首先在编程语言中提供 UOML 对象的对应表示。利用编程语言的复合数据类型的用户自定义功能，将每个 UOML 对象映射到一个定义的复合数据类型。复合数据类型可以是面向对象的语言中的类，或者是非面

向对象的语言中的结构。其中，UOML 对象中的属性在复合数据类型中可以通过简单数据类型的变量或者复杂数据类型的变量来表示，如整型（INT）、字符类型（CHAR）或者浮点类型（FLOAT）的变量，或者相应的数组等。UOML 对象的子元素通常属于 UOML 中的对象，在编程语言中定义了相应的复合数据类型，在这种情况下，在复合数据类型中，UOML 对象的子元素通过该子元素对应的复合数据类型的变量来表示。UOML 对象中一些不是 UOML 对象的子元素如描述位置的坐标值等，也可以通过简单数据类型的变量来表示。需要指出的是，上述的简单类型的变量或者复杂数据类型的变量以及复合数据类型的变量包含指针变量的情形。

步骤 102，将 UOML 操作映射为编程语言的 API 函数原型，UOML 操作的属性和子元素对应于 API 函数原型的参数。

由于 UOML 规范中的操作可能支持多个对象，在将 UOML 操作转换为编程语言的 API 函数原型时，需要考虑该编程语言是否支持函数的多态性定义。如果编程语言不支持函数的多态性定义，需要为每个操作支持的对象定义一个 API 函数模型，如果支持，则可以只定义一个 API 函数原型，通过函数原型中参数的类型来区分对不同对象的操作。UOML 操作的属性和子元素对应于 API 函数原型的参数。对于面向对象的语言，可以将 API 函数定义为类的成员函数，在这种情况下，如果操作的属性或者子元素同时是该类的成员变量，则在成员函数原型中可以不包含这些参数，而在函数体中直接引用这些变量。当然，也可以和非面向对象的语言一样，把操作所有的属性和子元素都作为函数原型的参数。

步骤 103，根据函数原型对应的操作和函数原型的参数，在编程语言的 API 函数体内实现函数功能向 UOML 操作的转换和执行，并将操作的返回结果转换为编程语言的返回类型。

在编程语言的 API 函数体内，根据函数原型和 UOML 操作的映射关系，以及函数参数和 UOML 操作的属性和子元素的对应关系，生成 UOML 的操作命令。将生成的操作命令发送给文档库系统，文档库系统执行后会返回执行结果。由于文档库系统返回的执行结果是 UOML 格式，所以需要将其转换为编程语言中对应的返回类型。

通常编程语言可以分为面向对象的语言和非面向对象的语言。由于编程语言本身的特点不同，所以对不同的编程语言，将 UOML 封装成编程语言的 API 的具体实现也有一些不同。

下面以编程语言 C++为例，介绍 UOML 封装成面向对象的语言的 API 的方法。

图 2 是本发明将 UOML 封装成面向对象语言的 API 的方法的流程图，如图 2 所示，包括如下步骤:

步骤 201，将 UOML 对象映射为面向对象语言的类，对象的属性对应于类中简单数据类型变量或复杂数据类型变量，对象的子元素对应于类中的类变量或简单数据类型变量。

以 UOML 文档对象为例，首先在 C++语言中定义一个对应的类 UOML_Doc。该类可以从 C++的所有类的基类 CObject 派生，也可以没有派生的基类。还有一种实现是，首先为所有的 UOML 对象定义一个基类:

```
class   UOML_Obj
{
public:
   virtual ~UOML_Obj();
   void   Init(UOML_Obj_Type type);
   UOML_Obj Clone(bool bMaintainRef = true);
   …
```

};

然后在基类 UOML_Obj 上派生 UOML_Doc。

为 UOML 文档对象定义对应类 UOML_Doc 之后，将 UOML 文档对象的子元素映射为 UOML_Doc 类的对应的成员变量。UOML 文档对象包含元数据（metadata）、各页面（pageset）、嵌入字库（fontinfo）等子元素。这些子元素分别有对应的类如：metadata 用 UOML_Meta 类表示，pageset 用 UOML_Page 类表示，fontinfo 用 UOML_FontInfo 类表示。所以 UOML 文档对象的这些子元素在 UOML_Doc 类定义中有对应的成员变量:UOML_Meta metadata; UOML_Page *pageset; UOML_FontInfo fontinfo。其中，pageset 子元素还可以通过 UOML_Page 的数组来表示。由于 UOML 文档对象不包含属性，所以没有属性向成员变量的映射。

同理，对于 UOML 页对象，为其定义 UOML_Page 类。将 UOML 页对象的 resolution、size、rotation、log 等属性，在 UOML_Page 类定义中用整型或浮点类型的成员变量表示，而 UOML 页对象的 GS、metadata 等子元素在 UOML_Page 类中以相应类的成员变量表示。

通过上述的方式，可以为 UOML 的所有对象在面向对象的语言中定义相应的类，包括文档库（UOML_DOCBASE）、文档集（UOML_DOCSET）、文档（UOML_DOC）、页（UOML_PAGE）、层（UOML_LAYER）、对象组（UOML_OBJGROUP）、文字（UOML_TEXT）、图像（UOML_IMAGE）、直线（UOML_LINE）、曲线（UOML_BEIZER）、圆弧（UOML_ARC）、路径（UOML_PATH）、源文件（UOML_SRCFILE）、背景色（UOML_BACKCOLOR）、前景颜色（UOML_COLOR）、ROP（UOML_ROP）、字符尺寸(UOML_CHARSIZE)、字体(UOML_TYPEFACE)、角色（UOML_ROLE）、权限（UOML_PRIV）等对象。本技术领域的技术人员可以参照上面的

说明来具体实现，在此不一一赘述。

步骤 202，将 UOML 操作映射为面向对象语言的类的成员函数，UOML 操作的属性和子元素对应于类成员函数的参数。

以 UOML 文档对象为例，作用于 UOML 文档对象的操作，如打开（UOML_OPEN）、关闭（UOML_CLOSE）等，在 UOML_Doc 类内定义相应的成员函数：Open()和 Close()。操作 UOML_OPEN 的属性包括文档所在的位置信息，将作为成员函数 Open()的参数：Open(char *szDocPath)。而 UOML_CLOSE 的子元素包括文档对象的句柄，也将作为成员函数 Close()的参数。

对于 UOML 的其它操作，如获取（UOML_GET）、设置（UOML_SET）、插入（UOML_INSERT）、删除（UOML_DELETE）、检索查询（UOML_QUERY）等也可以通过上述的方式对应到类的成员函数。

步骤 203，在成员函数体内将该函数对应的操作转换为 UOML 的操作，并将操作的返回结果转换为面向对象语言的返回类型。

例如在 UOML_Doc 类的成员函数 Open(char *szDocPath)的实体内，把相应的调用转为对文档库调用的 UOML XML 字符串，并发送到文档库：

```
<?xml version="1.0" encoding="UTF-8"?>
<UOML_OPEN create="true">
    <path val=szDocPath/>
</UOML_OPEN>
```

文档库收到请求后即执行操作。如果打开文档成功，即返回 XML 字符串。在 Open()函数体内部，根据接到的字符串构建 UOML_Doc 对象，然后作为函数的返回值返回该对象。

上面以 C++语言为例对面向对象的语言进行 UOML 到 API 的封装方法，可以看出的是，上述方法对其它的一些面向对象的语言，如 Object-C、Delphi、Java、Python、Ruby 等同样适用。

下面以编程语言 C 为例，介绍 UOML 封装成非面向对象语言的 API 的方法。

图 3 是本发明将 UOML 封装成非面向对象语言的 API 的方法的流程图，如图 3 所示，包括如下步骤:

步骤 301，将 UOML 对象映射为非面向对象语言的结构，对象的属性对应于结构中简单数据类型变量或复杂数据类型变量，对象的子元素对应于结构中结构变量或简单数据类型变量。

以 UOML 文档对象为例，首先在 C 语言中定义一个对应的结构 struct_UOML_Doc。

为 UOML 文档对象定义对应结构 struct_UOML_Doc 之后，将 UOML 文档对象的子元素映射为结构 struct_UOML_Doc 的对应的成员变量。UOML 文档对象包含 metadata、pageset、fontinfo 等子元素。这些子元素分别有对应的结构定义如: metadata 用结构 struct_UOML_Meta 表示，pageset 用结构 struct_UOML_Page 表示，fontinfo 用结构 struct_UOML_FontInfo 表示。所以 UOML 文档对象的这些子元素在结构 struct_UOML_Doc 定义中有对应的成员变量: struct_UOML_Meta metadata; struct_UOML_Page *pageset; struct_UOML_FontInfo fontinfo。其中，pageset 子元素还可以通过 struct_UOML_Page 的数组来表示。由于 UOML 文档对象不包含属性，所以没有属性向成员变量的映射。

同理，对于 UOML 页对象，为其定义结构 struct_UOML_Page。将 UOML 页对象的 resolution、size、rotation、log 等属性，在结构 struct_UOML_Page 定义中用整型或浮点类型的成员变量表示，而 UOML

页对象的 GS，metadata 等子元素在结构 struct_UOML_Page 中以相应的成员变量表示。

通过上述的方式，可以为 UOML 的所有对象在非面向对象的语言中定义相应的结构，包括 UOML_DOCBASE、UOML_DOCSET、UOML_DOC、UOML_PAGE、UOML_LAYER、UOML_OBJGROUP、UOML_TEXT、UOML_IMAGE、UOML_LINE、UOML_BEIZER、UOML_ARC、UOML_PATH、UOML_SRCFILE、UOML_BACKCOLOR、UOML_COLOR、ROP、UOML_CHARSIZE、UOML_TYPEFACE、UOML_ROLE、UOML_PRIV 等对象。本技术领域的技术人员可以参照上面的说明来具体实现，在此不一一赘述。

步骤 302，将 UOML 操作映射为非面向对象语言的全局函数原型，UOML 操作的属性和子元素对应于函数原型的参数。

以 UOML 文档对象为例，对于作用于 UOML 文档对象的操作，如 UOML_OPEN、UOML_CLOSE 等，定义相应的 API 函数原型：

struct_UOML_Doc* UOML_Doc_Open (char *szDocPath)

和

void UOML_Doc_Close(HANDLE *hDocHandle)。

操作 UOML_OPEN 的属性包括文档所在的位置信息，将作为 API 函数 UOML_Doc_Open ()的参数：szDocPath。而 UOML_CLOSE 的子元素包括文档对象的句柄，也将作为 API 函数 UOML_Doc_Close 的参数：hDocHandle。

对于 UOML 的其它操作，如 UOML_GET、UOML_SET、UOML_INSERT、UOML_DELETE、UOML_QUERY 等也可以通过上述的方式对应到 API 函数原型。

步骤 303，在函数体内将该函数对应的操作转换为 UOML 的操作，

并将操作的返回结果转换为非面向对象语言的返回类型。

例如在 API 函数 UOML_Doc_Open (char *szDocPath)的实体内，把相应的调用转为对文档库调用的 UOML XML 字符串，并发送到文档库:

```
<?xml version="1.0" encoding="UTF-8"?>
<UOML_OPEN create="true">
    <path val=szDocPath/>
</UOML_OPEN>
```

文档库收到请求后即执行操作。如果打开文档成功，即返回 XML 字符串。在 UOML_Doc_Open()函数体内部，根据接到的字符串构建 UOML_Doc 对象，然后作为函数的返回值返回该对象。

上面以 C 语言为例对非面向对象的语言进行 UOML 到 API 的封装方法，可以看出的是，上述方法对其它的一些非面向对象的语言，如 TCL、Pascal、Perl 等同样适用。

图 4 是本发明编程语言的 API 封装层次的示意图，如图 4 所示，过程 401 表示的是将 C/C++语言封装的 API 在对应函数体内直接转换为 UOML XML 操作命令，过程 402 表示的是将其它非 C/C++的编程语言封装的 API 在对应函数体内直接转换为 UOML XML 操作命令。对于其它非 C/C++的编程语言的 API 封装，除了在对应的 API 函数体内直接转换为 UOML XML 操作命令，还可以利用这些编程语言跟 C++或 C 语言的接口，在该编程语言的函数体内，直接调用 C++，C 语言封装成的对应 API 函数，正如过程 403 所示。而实际向 UOML XML 操作命令的转换在 C/C++封装的 API 函数体内完成。以编程语言 Java 为例，可以利用 Java 语言的本地接口规范（Java Native Interface，简称 JNI），在 Java 封装的 API 函数体内，绑定（binding）C/C++封装的对应 API 函数，而不需要进行直接的 UOML XML 转换。大多数脚本语言都提供了利用 C++

或 C 语言作扩展的机制，利用这种机制，就可以实现脚本语言的 API 对本地应用程序 API 函数的绑定。上述的脚本语言，除了 Java，还可以包括 TCL、Perl、Python 和 Ruby 等语言。

通过上述提供的方法，可以实现将 UOML 封装成编程语言的 API。开发人员在一编程语言上开发对文档库系统的应用时，可以直接调用生成的该编程语言封装的的 API，从而省去了大量的解析 UOML 的工作量，提高了开发的效率。

以上所述，仅为本发明的较佳实施例而已，并非用于限定本发明的保护范围，凡在本发明的精神和原则之内所做的任何修改、等同替换、改进等，均应包含在本发明的保护范围之内。

权利要求书

1、一种将 UOML 封装成应用程序编程接口 API 的方法，其特征在于，包括如下步骤:

A，将 UOML 对象映射为编程语言的复合数据类型;

B，将 UOML 操作映射为编程语言的 API 函数原型，UOML 操作的对象对应于 API 函数原型的参数;

C，根据步骤 A、B 中生成的映射，在编程语言的 API 函数体内将该函数对应的操作转换为 UOML 的操作，并将操作的返回结果转换为编程语言的返回类型。

2、根据权利要求 1 所述的方法，其特征在于，所述步骤 A 具体包括:

所述 UOML 对象的属性对应于复合数据类型中简单数据类型变量或复杂数据类型变量;

所述 UOML 对象的子元素对应于复合数据类型中简单或复合数据类型变量。

3、根据权利要求 2 所述的方法，其特征在于，所述编程语言为面向对象的编程语言;

步骤 A 中所述的编程语言中的复合数据类型为类，所述 UOML 对象的属性和子元素对应于类的成员变量;

步骤 B 中所述的编程语言的 API 函数原型为对应类中的成员函数。

4、根据权利要求 3 所述的方法，其特征在于，所述的面向对象的编程语言包括 C++、Object-C、Delphi、Java、Python 或 Ruby。

5、根据权利要求 2 所述的方法，其特征在于，所述编程语言为非面向对象的编程语言;

步骤 A 中所述的编程语言中的复合数据类型为结构;

步骤 B 中所述的编程语言的 API 函数原型为全局函数。

6、根据权利要求 5 所述的方法，其特征在于，所述的非面向对象的编程语言包括 C、TCL、Pascal 或 Perl。

7. 根据权利要求 3 或 5 所述的方法，其特征在于，所述编程语言具有和 C 或者 C++的接口;

步骤 C 中所述转换的方法为:通过调用 C++或 C 语言封装的 UOML 的 API 函数来实现向 UOML 操作的转换。

8、根据权利要求 7 所述的方法，其特征在于，所述的编程语言为 Java、TCL、Perl、Python 或 Ruby。

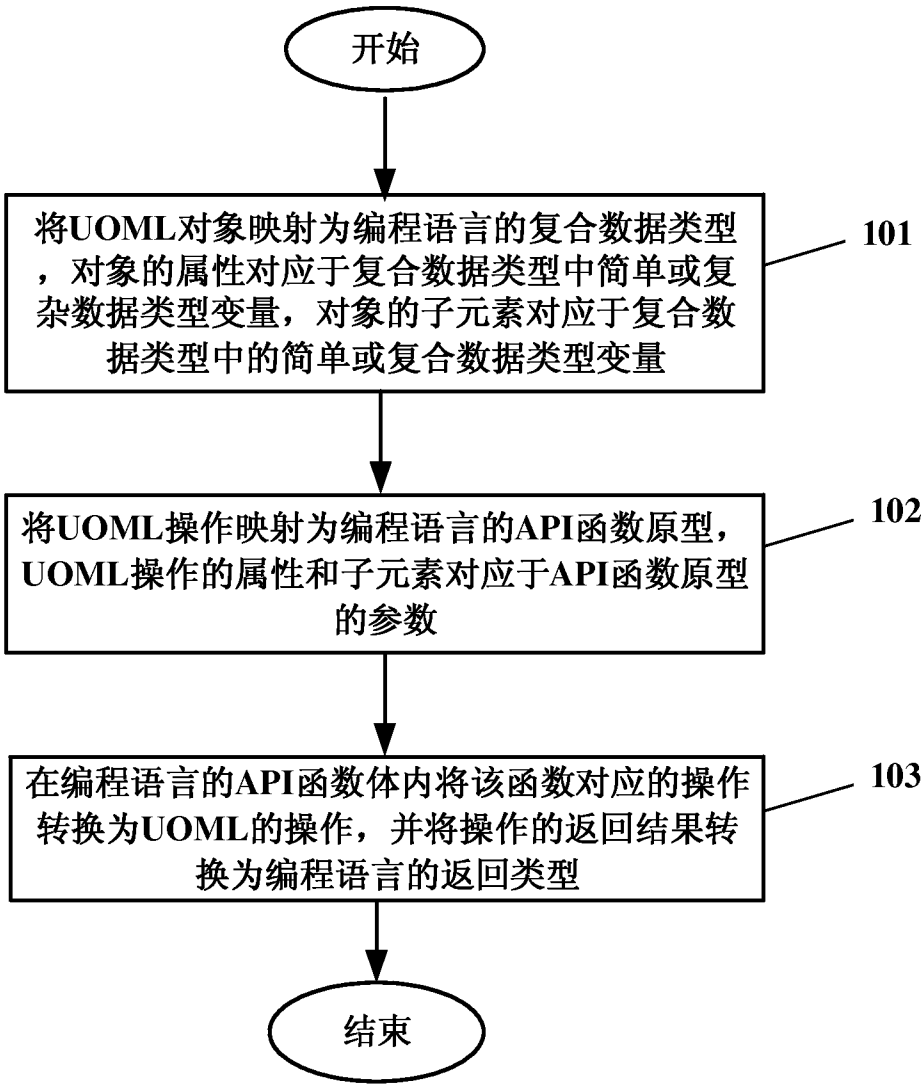

开始
将UOML对象映射为编程语言的复合数据类型，对象的属性对应于复合数据类型中简单或复杂数据类型变量，对象的子元素对应于复合数据类型中的简单或复合数据类型变量
101
将UOML操作映射为编程语言的API函数原型，UOML操作的属性和子元素对应于API函数原型的参数
102
在编程语言的API函数体内将该函数对应的操作转换为UOML的操作，并将操作的返回结果转换为编程语言的返回类型
103
结束

图 1

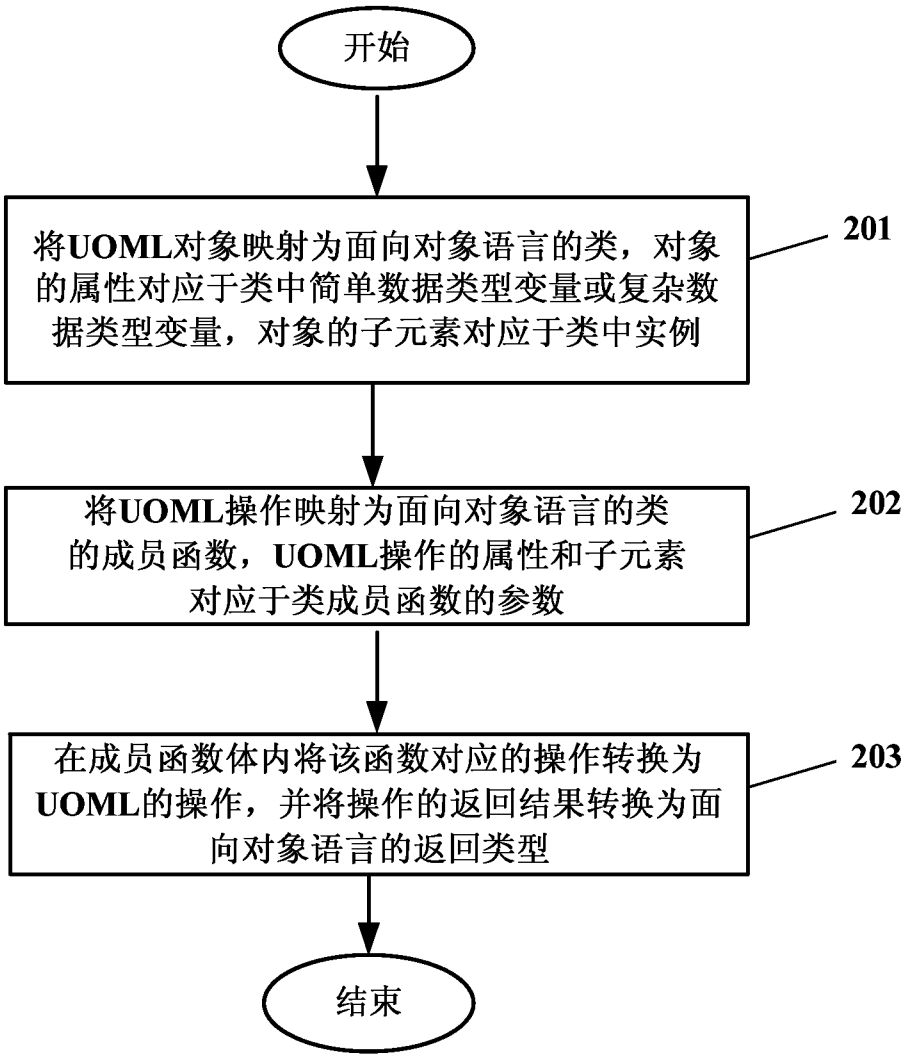

开始
将UOML对象映射为面向对象语言的类，对象的属性对应于类中简单数据类型变量或复杂数据类型变量，对象的子元素对应于类中实例
201
将UOML操作映射为面向对象语言的类的成员函数，UOML操作的属性和子元素对应于类成员函数的参数
202
在成员函数体内将该函数对应的操作转换为UOML的操作，并将操作的返回结果转换为面向对象语言的返回类型
203
结束

图 2

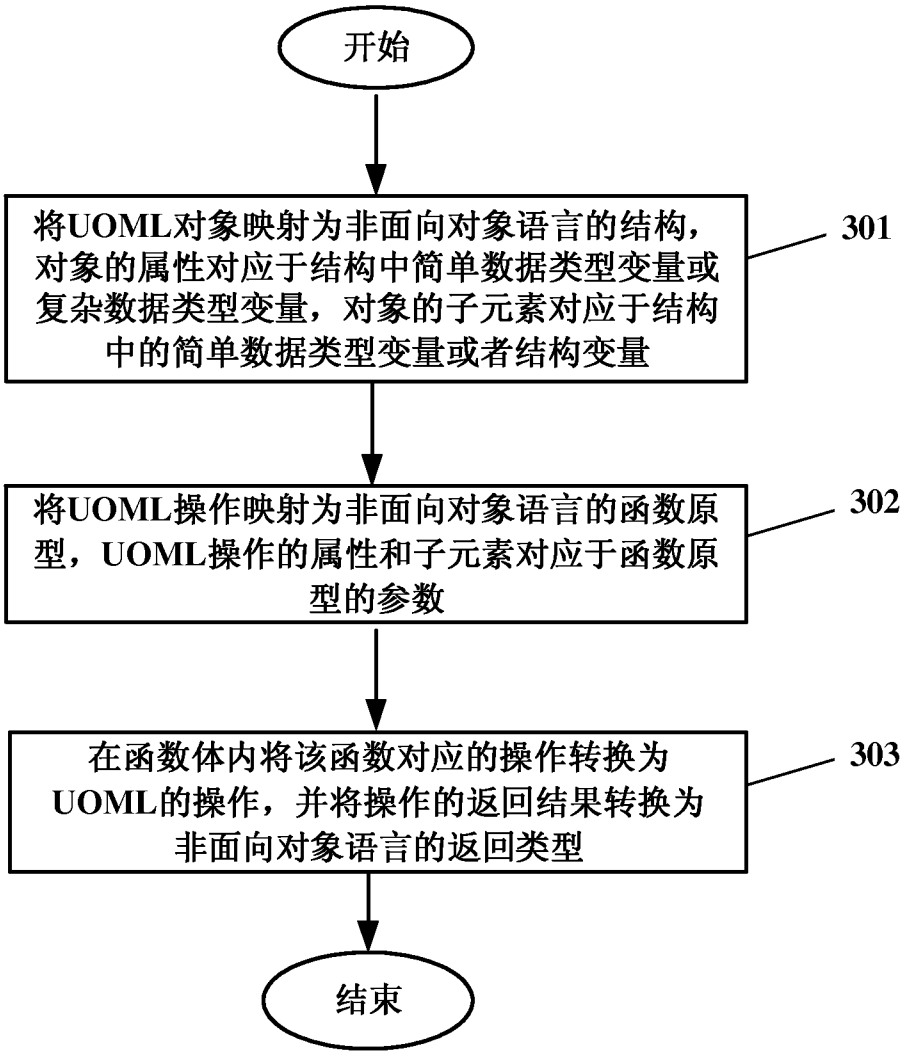

开始
将UOML对象映射为非面向对象语言的结构，对象的属性对应于结构中简单数据类型变量或复杂数据类型变量，对象的子元素对应于结构中的简单数据类型变量或者结构变量
301
将UOML操作映射为非面向对象语言的函数原型，UOML操作的属性和子元素对应于函数原型的参数
302
在函数体内将该函数对应的操作转换为UOML的操作，并将操作的返回结果转换为非面向对象语言的返回类型
303
结束

图 3

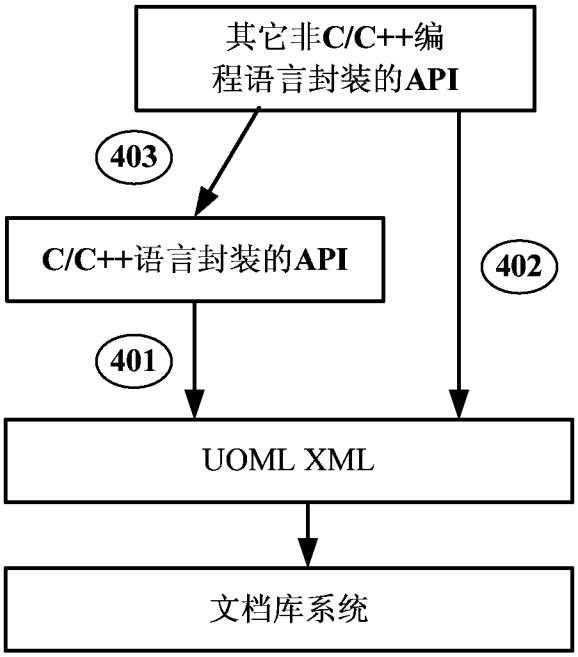


图 4

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2007/070132

A. CLASSIFICATION OF SUBJECT MATTER

G06F17/21 (2006.01) i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC: G06F17/-, G06F15/-, G06F9/-

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

WPI,EPODOC,PAJ,CNPAT,CNKI

JAVA, DELPHI, XML, HTML, NON STRUCTUR, EXTENSI+ MARKUP, OBJECT ORIENT+, FUNCTION?, CLASS??, OBJECT?

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US2004/0015840A1 (AVAY-N) AVAYA INC 22 Jan. 2004 (22.01.2004) Fig.5, Para.22-35, 94-98, 112-118	1-4, 7
Y		5, 6, 8
Y	US6480865B1 (IBMC) INT BUSINESS MACHINES CORP 12 Nov. 2002 (12.11.2002) Fig. 2, Col. 5 line 35-Col. 11 line 25	5, 6, 8
A	WO2005008484A1 (COMP-N) COMPUTER ASSOC THINK INC et al 27 Jan. 2005 (27.01.2005) the whole document	1-8

☐ Further documents are listed in the continuation of Box C. ☒ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim (S) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&"document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
30 Aug. 2007 (30.08.2007)	**20 Sep. 2007 (20.09.2007)**
Name and mailing address of the ISA/CN The State Intellectual Property Office, the P.R.China 6 Xitucheng Rd., Jimen Bridge, Haidian District, Beijing, China 100088 Facsimile No. 86-10-62019451	Authorized officer GUO, Shumei Telephone No. (86-10)62084989

Form PCT/ISA/210 (second sheet) (April 2007)

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2007/070132

Patent Documents referred in the Report	Publication Date	Patent Family	Publication Date
US2004/0015840A1	22 Jan. 2004	none	
US6480865B1	12 Nov. 2002	none	
WO2005008484A1	27 Jan. 2005	US2005234924A1	20 Oct. 2005
		EP1644827A1	12 Apr. 2006

国际检索报告

国际申请号
PCT/CN2007/070132

A. 主题的分类

G06F17/21 (2006.01) i

按照国际专利分类表(IPC)或者同时按照国家分类和 IPC 两种分类

B. 检索领域

检索的最低限度文献(标明分类系统和分类号)

IPC: G06F17/-, G06F15/-, G06F9/-

包含在检索领域中的除最低限度文献以外的检索文献

在国际检索时查阅的电子数据库(数据库的名称，和使用的检索词（如使用）)

WPI,EPODOC,PAJ,CNPAT,CNKI

非结构化，扩展标记，面向对象，函数，对象，类，JAVA，DELPHI，XML，HTML，NON STRUCTUR，EXTENSI+ MARKUP，OBJECT ORIENT+，FUNCTION?，CLASS??，OBJECT?

C. 相关文件

类　型*	引用文件，必要时，指明相关段落	相关的权利要求
X	US2004/0015840A1 (AVAY-N) AVAYA INC 22.1 月 2004（22.01.2004） 图 5，第 22-35 段，第 94-98 段，第 112-118 段	1-4，7
Y		5，6，8
Y	US6480865B1 国际商业机器公司 12.11 月 2002（12.11.2002） 图 2，说明书第 5 栏第 35 行-第 11 栏第 25 行	5，6，8
A	WO2005008484A1 计算机联合思想公司 等 27.1 月 2005（27.01.2005） 全文	1-8

☐ 其余文件在 C 栏的续页中列出。　　☒ 见同族专利附件。

* 引用文件的具体类型：

"A" 认为不特别相关的表示了现有技术一般状态的文件

"E" 在国际申请日的当天或之后公布的在先申请或专利

"L" 可能对优先权要求构成怀疑的文件，或为确定另一篇引用文件的公布日而引用的或者因其他特殊理由而引用的文件

"O" 涉及口头公开、使用、展览或其他方式公开的文件

"P" 公布日先于国际申请日但迟于所要求的优先权日的文件

"T" 在申请日或优先权日之后公布，与申请不相抵触，但为了理解发明之理论或原理的在后文件

"X" 特别相关的文件，单独考虑该文件，认定要求保护的发明不是新颖的或不具有创造性

"Y" 特别相关的文件，当该文件与另一篇或者多篇该类文件结合并且这种结合对于本领域技术人员为显而易见时，要求保护的发明不具有创造性

"&" 同族专利的文件

国际检索实际完成的日期	国际检索报告邮寄日期
30.8 月 2007（30.08.2007）	**20.9 月 2007 (20.09.2007)**
中华人民共和国国家知识产权局(ISA/CN) 中国北京市海淀区蓟门桥西土城路 6 号 100088 传真号：(86-10)62019451	受权官员 郭姝梅 电话号码：(86-10) **62084989**

国际检索报告
关于同族专利的信息

国际申请号
PCT/CN2007/070132

检索报告中引用的专利文件	公布日期	同族专利	公布日期
US2004/0015840A1	22.1 月 2004	无	
US6480865B1	12.11 月 2002	无	
WO2005008484A1	27.1 月 2005	US2005234924A1	20.10 月 2005
		EP1644827A1	12.4 月 2006