

# 一种 UML 模型到 XML 模型的转换方法

王建光,段 富

(太原理工大学 计算机软件学院,山西 太原 030024)

**摘 要:**为了把 UML 模型用 XML 描述出来,可以从模型转换的角度来考虑这个问题,把 XML 描述结果当作 XML 模型考虑。利用集合的概念定义了一个 UML 源模型,并用构造树的形式描述了 UML 基本模型元素的结构。根据构造树的描述定义目标 XML 模型的 XML Schema。提出了一种基于对模型元素实例集合的扫描和对模型元素构造树遍历的方法来进行 UML 模型到 XML 模型的自动转换。

**关键词:**UML 模型;XML;模型转换;构造树

**中图分类号:**TP311

**文献标识码:**A

**文章编号:**1673-629X(2007)07-0123-04

## A Method for Transformation from UML Model to XML Model

WANG Jian-guang, DUAN Fu

(College of Computer and Software, Taiyuan University of Technology, Taiyuan 030024, China)

**Abstract:** Consider the problem on how to describe UML model using XML from the angle of model transformation and regarding the result of XML description as XML model. In this paper, a UML source model is defined using the concept of set and the structure of element of UML basic model is described using the form of the construct tree. Afterward, a target XML Schema is defined basing the description of construct tree. In the end, a method is put forward to implement the automated transformation from UML model to XML model, which basing scan for the set of the instances of model element and traversing the construct tree.

**Key words:** UML model; XML; model transformation; construct tree

### 1 模型变换概述

对于构造一个目标系统,总是需要对将要构造的目标系统进行建模,开发目标系统的模型。模型描述了系统全部或部分的功能、结构和行为。这种描述总是以某种语言表达的。这种语言可以是图形符号语言,如 UML;也可以是字符类语言,如编程语言;也可以是自然语言甚至任何语言。但是,这种语言必须具有确定的语法和语义。

一般地,从模型的抽象程度的角度上来讨论,模型可以分为四个层次,主要包括元元模型(Meta-meta Model)层、元模型(Meta Model)层、模型(Model)层、用户对象(User Object)层。其中,元元模型是元模型的基础体系结构,定义了一种说明元模型的语言。元模型是元元模型的一个实例,定义了一种说明模型的语言,并说明该模型语言的抽象语法,严格定义其语义<sup>[1]</sup>。模型是元模型的一个实例,定义一种语言来描

述信息领域。用户对象是模型的一个实例,定义了一个特定的信息领域。对各级模型元素的定义方式一般是,首先给出它的抽象语法,然后给出其形式化规则,最后描述其语义。

一个系统可能存在很多不同的模型。既然这些模型是对同一系统所做的描述,那么同一系统的两个或多个模型之间必然存在某种联系。从模型转换的角度来说,模型之间存在一种自动变换生成关系,也即可以由一个模型变换生成另外一个模型。从源模型变换到目标模型,必须把源模型中的每个部分同目标模型中的一个或多个部分建立联系。要建立这种联系,可以考虑源模型与目标模型所使用的建模语言之间的联系,也即基本模型元素之间的联系。建立模型元素之间的联系,一种很好的方式是吧源建模语言中的基本模型元素进行适当的分解,然后把分解的每一部分映射到目标建模语言中的基本模型元素。

从源模型到目标模型的变换,需要定义一系列无歧义的变换规则<sup>[2]</sup>。这些变换规则表述了模型之间是如何变换的。变换是依据模型以及变换定义产生新模型的过程。把变换过程放进一个黑盒,对于所有的输

收稿日期:2006-09-20

作者简介:王建光(1981-),男,湖北人,硕士研究生,研究方向为分布式环境下的数据管理与安全;段 富,教授,研究方向为软件理论与算法、计算机图形学。







一个文本描述,是一个 XML 文件。

3 UML 模型到 XML 模型的转换

UML 中每一个基本的模型元素都有确定的语法和语义。从 UML 模型转换到 XML 模型,两者之间必须具备语义上的一致性。所以,可以根据 UML 模型元素的语法和语义来定义 XML Schema 文件。而对于 UML 中每一个模型元素根据其语法都可以从形式上表达成一棵构造树,更进一步地可以从模型元素的构造树来定义 XML Schema 文件。由此定义的 XML Schema 使得从 UML 模型转换到 XML 模型不会丢失语义信息。

定义 1 UML 模型的表示。

设  $E_i$  为 UML 中的任意一个基本模型元素,  $E_{ij}$  为基本模型元素  $E_i$  的任意一个实例;再设 UMLModel 为基于 UML 建模语言所构建的一个模型,则 UMLModel 可以表示为一个集合:

$$UMLModel = \{M_1, M_2, \dots, M_n\}$$

其中,

$$M_i = \{E_{i1}, E_{i2}, \dots, E_{ik}\}$$

例如,图 2 是一个 UML 模型,其中包括类 Supplier、类 Transaction、类 Buyer,另外还包括关联 Association1 (Supplier, Transaction)、关联 Association2 (Buyer, Transaction)。所以这个 UML 模型可以表示为集合  $UMLModel = \{\{Supplier, Transaction, Buyer\}, \{Association1, Association2\}\}$ 。在集合 UMLModel 中,元素 {Supplier, Transaction, Buyer} 是类模型元素的实例的集合,反映了模型中所有类的集合;元素 {Association1, Association2} 是关联模型元素的实例的集合,反映了模型中各个类之间的关联。

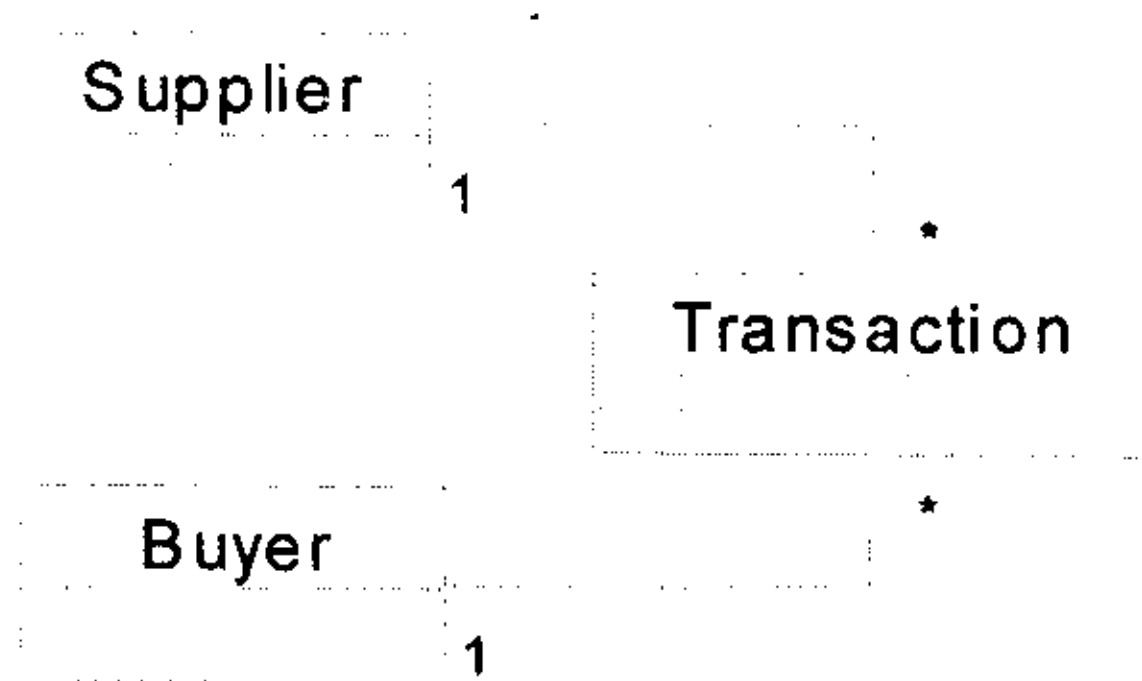


图 2 一个 UML 模型

下面以构建 UML 类图模型为例,说明从 UML 模型转换到 XML 模型的步骤。

(1)描述模型元素的构造树。

UML 类图是最常用的 UML 图,它用于描述系统的结构化设计,其中包括类关系以及与每个类关联的属性及行为。类图能出色地表示继承与合成关系。UML 类图中的基本模型元素主要包括类、关系、接口、包四种元素,其中关系包括关联、泛化、聚合、复合、依

赖、实现等基本模型元素。

类是对象的蓝图,是具有相同属性和操作的对象集合<sup>[5]</sup>。其中包含 3 个组成部分:第一个标识类的类名,它用来标识信息系统中的某个实体,在一个系统中,类名是唯一的;第二个是类的属性,它体现了某个实体的自身属性,类可以有零个或多个属性;第三个是该类提供的操作方法,它主要表达对实体属性允许的操作和功能。类也可以有零个或多个操作。属性和操作都有可见性。一般通过利用可见性把实体属性封装在类中,对外部实体来说是不可见的;另外,类的操作一般对外部实体是可见的,外部实体通过调用某个类的操作来改变其所表达的实体的状态。可见性一般包括公共(Public)、私有(Private)和保护(Protected)。对于类,可以对其进行形式化描述如下:

```
<Class> ::= <ClassName> [ <Attribute> * ] [ <Method> * ]
<Attribute> ::= <Visibility> <AttributeType> <AttributeName>
<Method> ::= <Visibility> <ReturnType> <MethodName> [ <Param> * ]
<Param> ::= <ParamType> <ParamName>
<Visibility> ::= public | private | protected
```

类之间的关系是类图中比较复杂的内容。类之间的关系表达了信息系统中各个实体相互之间错综复杂的联系。类之间的关系主要包括关联、泛化、聚合、复合、依赖、实现等。每一种关系都有两个或多个关系端,一个关系端是对实体的一个引用。

a. 关联关系表达信息系统实体间的一种语义联系,是类之间一种较弱的联系。关联可以有方向,可以是单向关联,也可以是双向关联,并且可以给关联加上关联名来描述关联的作用。关联两端的类也可以以某种角色参与关联,角色可以具有多重性,表示可以有多个对象参与关联。另外,可以进一步描述关联的属性、操作以及其它属性。

b. 泛化关系定义了一般元素和特殊元素之间的分类关系。泛化关系表达了一种继承关系,是“a - kind - of”关系。

c. 聚合关系是一种特殊的关联,它表示整体与部分之间的关系。聚合暗示着整体在概念上处于比局部更高的一个级别,而关联暗示两个实体在概念上处于同一个级别。关联和聚合的区别纯粹是概念上的,而严格反映在语义上。

d. 复合关系也是一种特殊的聚合,但在复合中整体与部分具有相同的生存期。聚合关系是“has - a”关系,复合关系是“contains - a”关系。

e. 依赖关系体现了这样一个事实:实体之间的“使



用”关系暗示一个实体的规范发生变化后,可能影响依赖它的其它实体;也表达了一组实体与另外一组实体之间的相互影响关系。

f. 实现关系指定了两个实体之间的一个合同。也就是说,一个实体定义一个合同,而另一个实体保证履行该合同。

类图中的关系可以形式化描述如下:

$\langle \text{Relation} \rangle ::= \langle \text{RelationType} \rangle \langle \text{RelationEnd} \rangle *$

$\langle \text{RelationType} \rangle ::= \text{association} | \text{generalization} | \text{aggregate} | \text{composition} | \text{dependence} | \text{realization}$

$\langle \text{RelationEnd} \rangle ::= \langle \text{RelationEndRef} \rangle \langle \text{Multiplicity} \rangle \langle \text{Role} \rangle$

$\langle \text{RelationEndRef} \rangle ::= \langle \text{ClassName} \rangle$

$\langle \text{Role} \rangle ::= \langle \text{RoleName} \rangle$

$\langle \text{Multiplicity} \rangle ::= \langle \text{Upper} \rangle \langle \text{Lower} \rangle$

接口是一系列操作的集合,它指定了一个类所提供的服务。接口并不揭示其内在的结构或具体实施,通常用来定义或限制实体对外的服务。形式描述如下:

$\langle \text{Interface} \rangle ::= \langle \text{InterfaceName} \rangle [ \langle \text{FinalAttribute} \rangle * ] [ \langle \text{Method} \rangle * ]$

$\langle \text{FinalAttribute} \rangle ::= \langle \text{FinalType} \rangle \langle \text{FinalName} \rangle \langle \text{FinalValue} \rangle$

包是组织和管理实体或接口服务的容器。一个包里的各个实体相互合作,共同给外部使用者提供一组完整的服务。包与包之间可以相互引用,形成依赖关系。形式描述如下:

$\langle \text{Package} \rangle ::= \langle \text{PackageName} \rangle \langle \text{PackageBody} \rangle$

$\langle \text{PackageBody} \rangle ::= ( \langle \text{Class} \rangle | \langle \text{Relation} \rangle | \langle \text{Interface} \rangle | \langle \text{PackageBody} \rangle ) *$

上述各个类图模型元素的形式化描述表达了模型元素的形式化语法,对于 UML 模型的转换提供理论上的依据。根据形式化描述,可以对每个模型元素定义转换,创建转换单元;然后,把每个转换单元集中一起就可以构造自动转换机。从上述形式化描述可以看出,其中每一个模型元素都可以被表达成一个构造树,其中根结点为模型元素的标识符,非叶子结点又是一个构造单位,叶子结点为模型元素构造部分的极小单位(对于建模过程中所产生的模型元素的实例来说,是作为最小输入单位体现的)。如图 3 所示为一个类的构造树。

(2) 定义 XML Schema。

现在已经描述好模型元素的构造树,接下来就可以根据构造树的深度把其中的结点定义相应的 XML Schema 中的元素或属性。把构造树中的非叶子结点

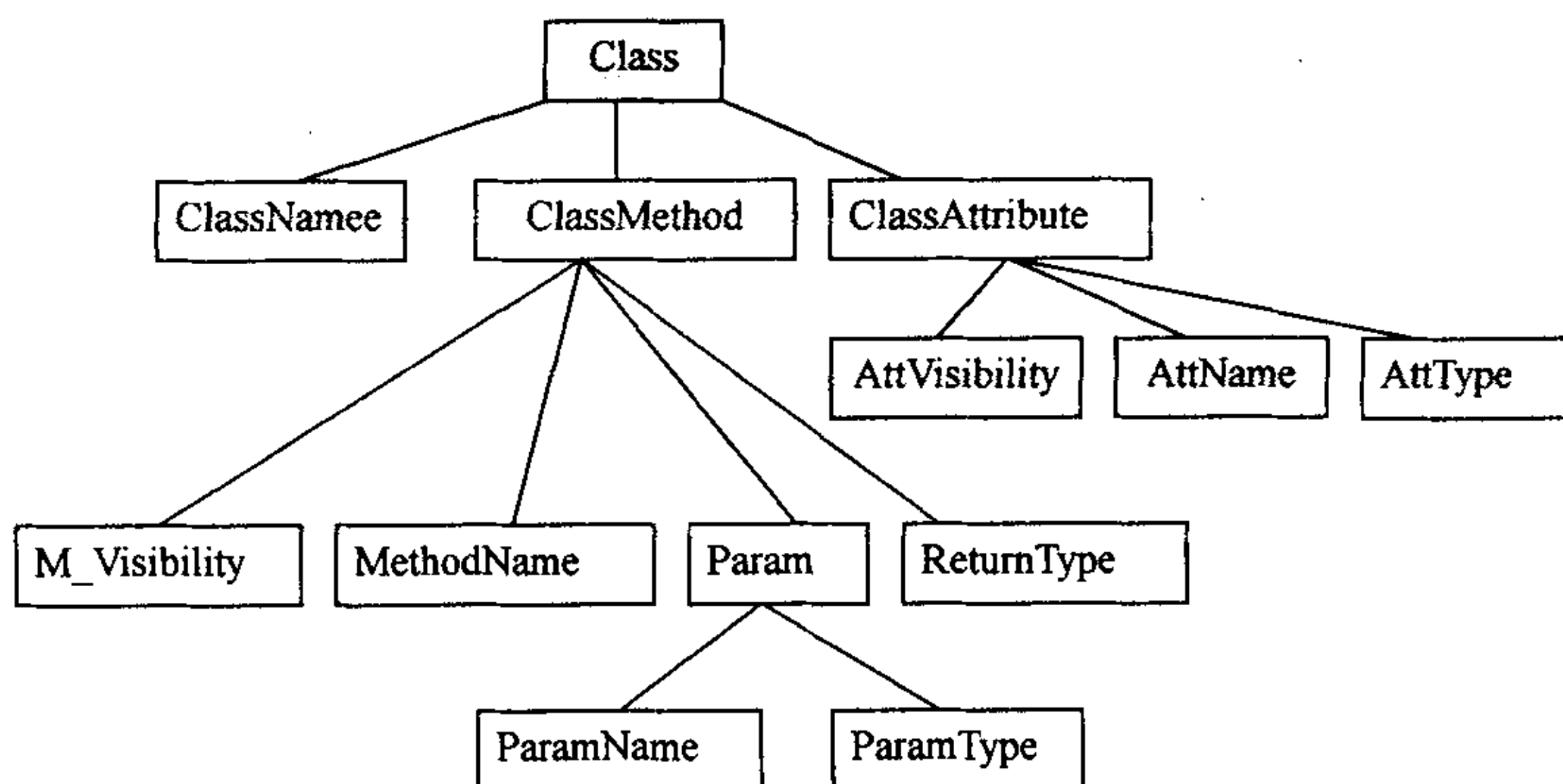


图 3 类的构造树

定义成 XML 中的元素 (element), 其中每个元素都是复杂类型的元素, 其子元素为空或者为其父元素在构造树中相应结点的非叶子类型的子结点所对应的元素序列。把叶子结点定义成其父结点在 XML Schema 中对应元素的属性 (attribute)。如图 4 所示为 XML Schema 片段。

```

<xsd:element name="Class">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="ClassAttributes" minOccurs="1" maxOccurs="1"/>
      <xsd:element ref="ClassMethods" minOccurs="1" maxOccurs="1"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
    
```

图 4 XML Schema 片段

(3) 构造 UML 基本模型元素到 XML 的转换规则。

有了 UML 模型元素的结构树的描述和相应的 XML Schema 定义, 构造转换规则就比较简单了。可以根据定义 XML Schema 的原则来为每种模型元素构造转换规则函数: 采取先根遍历算法扫描模型元素的构造树, 如果扫描到非叶子结点, 那么把结点转换为 XML 中的元素 (element), 元素名为结点名; 如果扫描到叶子结点, 那么把结点转换为叶子结点的父结点所对应 XML 中元素 (element) 的属性 (attribute), 属性名为叶子结点的名称, 属性值为叶子结点的值。

(4) 从 UML 源模型转换到 XML 目标模型的转换方法。

根据定义 1, 扫描 UML 源模型, 把 UML 源模型中同类模型实例装载到同一个集合, 这些不同类型的实例集合的集合构成了 UML 源模型。然后扫描 UML 源模型集合, 对其中的每个集合性元素 (因为源模型集合中的元素是集合, 为了便于理解, 把这些元素

(下转第 174 页)



和  $t_{data-flight-max}$ 。

本系统中使用 CADENCE 公司的板级设计工具 ALLEGRO 进行设计和仿真,布局之后首先布好时钟线,用 33 欧姆的电阻源端端节以保证源-负载阻抗匹配,然后进行仿真,时钟信号波形如图 6 所示。

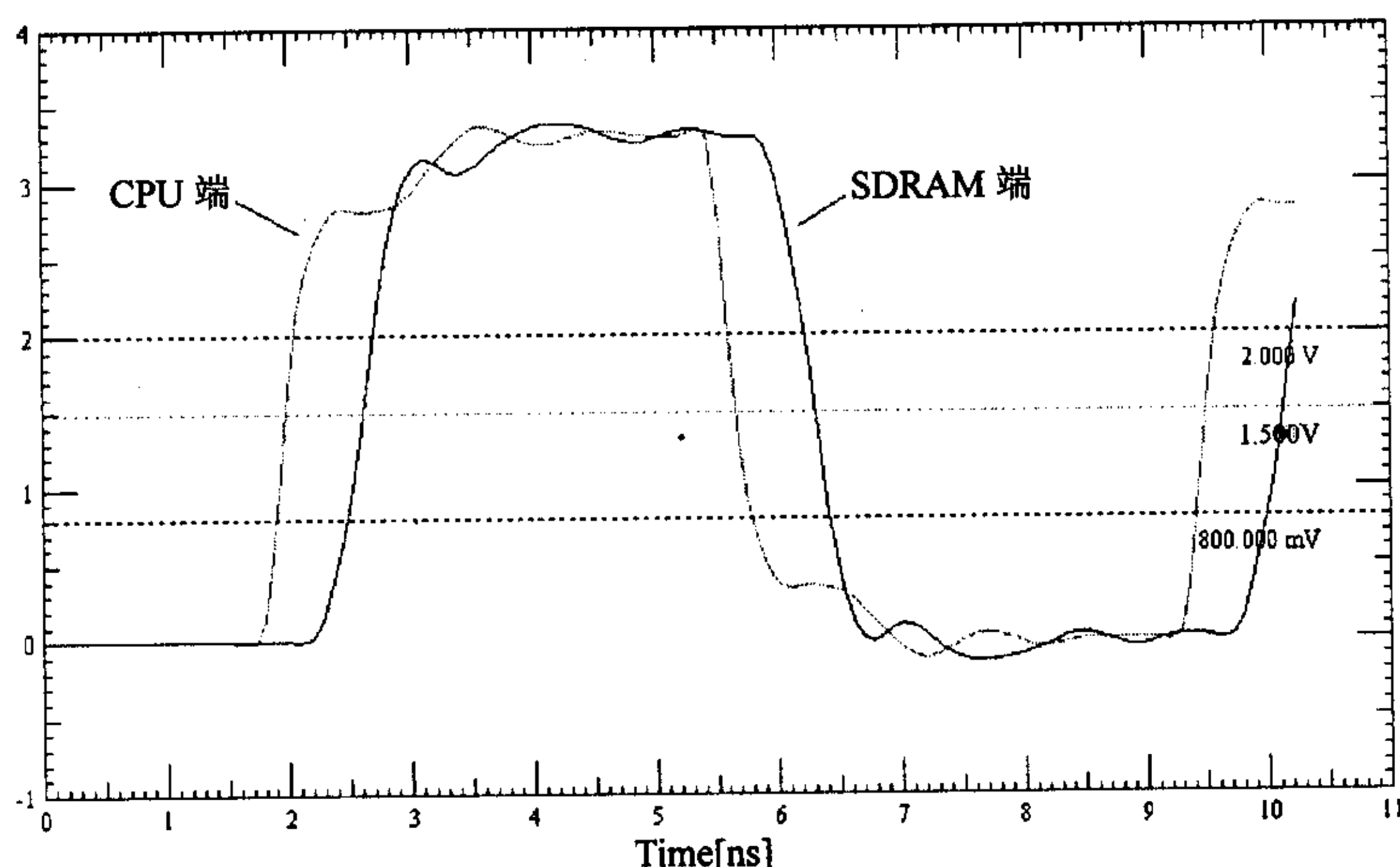


图 6 时钟信号仿真波形

从振铃、上下过冲、单调性等方面分析仿真波形,可以看出无论是源端还是负载端信号质量都较好,在此基础上再看仿真给出的  $t_{clk-flight-min}$  和  $t_{clk-flight-max}$ ,这两个值分别为 0.463ns 和 0.757ns。

再以  $t_{clk-flight-min}$  和  $t_{clk-flight-max}$  两个值为基准,由式(9)至(12)算出数据总线上信号的飞行时间  $t_{data-flight-min}$  和  $t_{data-flight-max}$  满足:

$$t_{data-flight-min} > 0.657ns \text{ 且 } t_{data-flight-max} < 1.563ns \quad (13)$$

式(13)即为最终的对本系统的数据总线的时序要求,只有当数据总线上的信号满足这个式子,整个时序才能被满足。在 ALLEGRO 工具中给数据总线加上

这两个电气约束以指导布线,ALLEGRO 会对那些经过仿真不能满足上式要求的网络加上警告,便于检查和修改<sup>[5]</sup>,一般来说,最大飞行时间,即  $t_{data-flight-max}$  容易满足条件,而  $t_{data-flight-min}$  则可以添加延时线以符合要求。布线完毕之后,再对数据总线上的所有网络进行仿真,从波形中观察信号质量并查看最小/最大飞行时间是否满足式(13)的要求。若数据总线中每个网络的信号质量良好,且满足式(13),则表明该系统数据总线部分的时序得到了有效的控制。

## 4 总 结

在数字系统的运行速度越来越高,时序要求越来越严格的背景下,给出了如何保证系统关键网络时序正常的一种方法。该方法从满足器件建立/保持时间入手,并以可靠的电路仿真工具为重要手段,根据系统工作时序,推导出最小/最大飞行时间的约束,并以此来指导系统设计。

### 参考文献:

- [1] Bogatin E. 信号完整性分析[M]. 李玉山,李丽平译. 北京:电子工业出版社,2005.
- [2] 刘 斌,李仲阳. ARM/DSP 双核系统的通信接口设计[J]. 单片机与嵌入式设计,2005(5):22-24.
- [3] Brooks D. 信号完整性问题和印制电路板设计[M]. 刘雷波,赵 岩译. 北京:机械工业出版社,2005.
- [4] Johnson H, Graham M. 高速数字设计[M]. 沈 立,朱来文,陈宏伟译. 北京:电子工业出版社,2004.
- [5] Jon P, Mazur D. IBIS evolves: Keeping pace with signal integrity issues[M]. [s.l.]: Printed Circuit Fabrication, 1998.

(上接第 126 页)

称作集合性元素)进行处理。按照 UML 基本模型元素所建立的转换规则对集合性元素的子元素进行转换。当源模型集合中的所有元素都扫描完毕后,就自动生成了完整的 XML 模型。

## 4 结 论

对模型以及模型转换进行了论述,阐述了模型的概念以及模型转换的一般方法与原则。从模型的角度,把 XML 作为一种模型描述语言来描述 UML 模型,通过自动转换生成 XML 模型。最后,提出了一种从 UML 模型自动转换到 XML 模型的方法。该方法不完善的地方就是,XML 模型仅记录了 UML 模型的语义信息,没有记录 UML 图形模型的位置信息,所以

还不能从 XML 模型转换到 UML 模型。下一步的工作就是要解决这个问题。

### 参考文献:

- [1] 马于涛,陈建勋. 基于 UML 的软件体系结构建模技术的研究[J]. 武汉科技大学学报,2003,26(3):308-311.
- [2] Kleppe A, Warner J, Bast W. 解析 MDA[M]. 鲍志云译. 北京:人民邮电出版社,2003.
- [3] 颜玉兰,何克清,刘 进. 一种基于有限状态机的模型转换方法[J]. 计算机工程,2006,32(1):93-95.
- [4] 刘升平,林作铨,梅 婧,等. 一种 XML 模型论语义[J]. 软件学报,2006,17(5):1089-1097.
- [5] 韦银星,张申生,曹 键. UML 类图的形式化及分析[J]. 计算机工程与应用,2002(10):5-7.