基于 VML 的复句关系层次树的可视化研究 *

胡金柱[a,b], 舒江波[a,b], 周 星[b]
(华中师范大学 a. 语言与语言教育研究中心; b. 计算机科学系, 武汉 430079)

摘 要: 在现代汉语复句教学和研究中,为了分析复句的逻辑语义关系,经常需要绘制复句关系层次结构图。传统的做法是利用手工方式绘制,以图片的形式存储,但这种方式具有存储容量大、绘制工作量大的缺点。采用 VML 技术,在标注复句的基础上,研究了复句关系层次树的自动生成方法以及复句关系层次树在网页中的可视化。通过对不同类型复句进行试验,结果表明各种类型的复句均能准确显示。这就表明该可视化方法能够有效地应用到复句信息工程的研究中。

关键词: 矢量可标记语言; 复句; 关系层次树; 可视化
中图分类号: TP391 **文献标志码**: A **文章编号**: 1001-3695(2010)01-0127-04
doi:10.3969/j.issn.1001-3695.2010.01.038

Visualization of relation hierarchical tree of compound sentence based on VML

HU Jin-zhu[a,b], SHU Jiang-bo[a,b], ZHOU Xing[b]
(*a. Center for Language & Language Education, b. Dept. of Computer Science, Huazhong Normal University, Wuhan* 430079, *China*)

Abstract: In the research and teaching of Chinese compound sentence, it often needs to draw compound sentence relation hierarchical tree (short for RHT) for the analyzing of logic semantic relation among clauses. In traditional way, the RHT is drawn in manual manner and stored as picture. This method has the shortage of needing huge storage and having a huge workload for getting RHTs of a large number of compound sentence. This paper proposed a method to automatically generate RHT on the basis of marked-up compound sentence by using VML, and displayed the RHT in Web pages. By testing the compound sentences with different structures, the result shows that it can display the RHTs correctly, and this also demonstrates the efficiency of the method.

Key words: vector markup language(VML); compound sentence; relation hierarchical tree; visualization

现代汉语复句信息工程包括以下五个方面的研究内容:a)汉语复句关系词的覆盖范围、配对情况、搭配强度和关系词的相似度;b)复句关联模式、内部结构特点与外部功能特点;c)关系词和复句复杂特征集的描述与合一运算;d)关系词的自动识别和标注、关联项功能的自动识别和标注、复句句式类别与层次关系的自动识别与标注;e)有标复句标注语料库的设计与建设以及面向复句领域的研究工具的开发[1]。

在现代汉语教学和科研的内容中,多重复句的层次划分是一个研究的难点。目前,较多的是采用人工划分的方法。当然,也有学者研究多重复句的自动层次划分方法,如文献[2,3]。不管是人工划分也好,还是自动划分也好,所得到的结果都是形如"S:1…,||SZZ2…,||||SBL3…,|||SDJ4…,|SBL5不…,||||SBL6…,|||SBL7…,||||SBL8…,||STJ9…。"的形式。显然,在教学的过程中,这种形式不太直观。用图形化的方式,如用树形图的方式就能比较形象、直观地反映复句的层次关系。但是,每一个划分的复句,都要靠手工绘制对应的关系层次树,这是非常烦琐且费时费力的。因此,研究复句关系层次树的自动生成以及关系层次树的可视化,是一个有意义的课题。

目前,在汉语言资源的可视化方面,国外的研究较少,国内在这方面做了较多工作的是哈尔滨工业大学和北京大学。哈工大信息检索研究室的 LTP:S 可视化系统使用了 VML 技术,将句子的词性标注、词义消歧、命名实体、句法分析、语义分析的结果在网页中显示出来。北京大学采用 VML 技术,实现现代汉语句法树库中句法结构树的自动生成,并将句法分析的结果在网页中进行显示,方便了人们对句法知识的学习、研究和交流。但是,北大的现代汉语树库,主要处理的是单句,而没有处理复句。虽然在句法树中有标记"fj"表示复句,但是,它是从语表的层次来标记的,反映的是多个分句构造成一个整句的情况,而不是标记的邢福义[4]提出的反映逻辑语义关系的复句。

本文在标注了层次关系复句的基础上,研究基于 VML 的复句关系层次树的自动生成以及复句关系层次树的可视化。其目标是实现汉语复句语料库的知识共享,将语料库中复句实例的句法语义信息详细准确地展现出来,方便现代汉语复句的教学和研究。

1 基本思路

复句关系层次树的可视化是复句信息工程的最后一个环节,其前期工作有关系词的识别和标注,复句关系层次的自动划分。复句关系层次树是复句的一种逻辑表示,以树状结构反映复句的层次关系。

在网页上显示复句关系层次树,其方法是利用浏览器支持

收稿日期: 2009-05-24; **修回日期**: 2009-06-28 **基金项目**: 国家教育部人文社会科学重点研究基地重大项目(07JJD740063);湖北省科技攻关资助项目(2007AA101C49)

作者简介:胡金柱(1947-),男,教授,博导,主要研究方向为中文信息处理、软件工程(jzhu@mail.ccnu.edu.cn);舒江波(1982-),男,博士研究生,主要研究方向为中文信息处理;周星(1985-),女,硕士研究生,主要研究方向为软件工程.

的语言,将复句的层次关系以及各个分句编码成对应的图形对象元素,然后在浏览器中解码显示。

VML 基于 XML 标准[5],支持高质量的矢量图形显示,支持广泛的矢量图形特征,它们基于由相连接的直线和曲线描述路径。在 VML 中使用两个基本的元素,即 shape 和 group。这两个元素定义了 VML 的全部结构;shape 描述一个矢量图形元素,而 group 用来将这些图形结合起来,这样它们可以作为一个整体进行处理。由于 VML 使用简单的文本来表示图像,这样就可用很少的字节来表示比较复杂的图像[6]。本文采用 VML 来表示复句关系层次树的图形描述。基于 VML 实现复句关系层次树的可视化的流程如图 1 所示。

图1　复合关系层次树的可视化处理流程

从图 1 可以看出,要在网页上显示复句关系层次树:首先将经过标注或未经标注的复句进行预处理,得到可索引形式;在这个形式的基础上,接着确定索引顺序,根据索引顺序扫描复句,生成复句结构树的逻辑节点;然后将逻辑节点转换成图形元素,并通过定位生成 VML 节点;最后通过浏览器解码,显示树形图。

预处理过程主要是去除标注复句中的词语标注信息,如关系词的类别标注、关系词邻接项的功能标注、省略成分标注、特殊标志词标注以及篇章关联词标注等,留下层次标注信息和分句标号。

2　可视化的算法实现

为方便描述,本文采用的复句实例为 S:"1 尽管苹果公司目前遇到了一些问题,||SZZ2 但它毕竟还是个大公司,||||SBL3 它的产品仍然过硬,|||SDJ4 尤其是技术力量雄厚,|SBL5 不管它将来是被别的公司兼并,||||SBL6 或是自己坚持走下去,|||SBL7 只要吸取经验教训,||||SBL8 调整策略,||STJ9 它仍应是"电脑社会"令人瞩目的家族。",后面的分析说明均针对该句子。

2.1　确定索引顺序

要将线性语流的复句表示成树状结构,首先要找出层次关系,然后根据层次关系层层扫描,生成树状逻辑结构。找出层次关系的过程,就是确定索引顺序的过程。例如,对于例句 S,先找出第一层所在的位置,然后找出第二层所在的位置,依次下去。

确定索引顺序有以下两种策略:

a)多次扫描的策略。它是按照关系层次的优先级对 S 进行多次扫描,每扫描一次,确定一个关系层次的索引顺序。对于句子 S,该方法得到的中间结果如下:

第一次扫描:{|SBL5}

第二次扫描:{||SZZ2,||STJ9}

第三次扫描:{|||SDJ4,|||SBL7}

第四次扫描:{||||SBL3,||||SBL6,||||SBL8}

索引顺序为:四个集合中的元素从左到右依次排序。

b)一次扫描的策略。它是从左往右扫描句子 S,每扫描到一个关系层次符号(记为一个节点),先计算其优先级,然后根据当前已处理的节点集合,确定当前节点的临时索引顺序;然后更新已处理节点集合中各个节点的索引顺序;最后将当前节点并入已处理节点集。

本文采用一次扫描的策略。具体方法如下:

节点定义:$H_i = \langle A, B, C \rangle$。其中:$A$ 为层次级别;B 为所在分句;C 为处理序号。

节点处理优先级:优先级的影响因素有两个,即层次级别和分句所在位置。在判断两个节点之间的优先级时,先比较层次,再比较位置。层次小的优先级高。若层次相同,则位置靠前的优先级高。$P_i = \alpha \times H_i.A + \beta \times H_i.B$。本文中,$\alpha = 1$,$\beta = 0.01$。例如,节点"|SBL5" $= \langle 1, 5, * \rangle$,其中 * 表示处理序号未知,其优先级 $P_1 = 1 \times 1 + 0.01 \times 5 = 1.05$;节点"||SZZ2" $= \langle 2, 2, * \rangle$,其优先级 $P_2 = 1 \times 2 + 0.01 \times 2 = 2.02$。$P_i$ 越小,表明该节点的优先级越高。

a)初始节点为 H_1,其处理序号 $H_1.C = 1$,初始历史节点集合 $H = \{H_1\}$。设历史节点集合中节点最大优先级为 $\max P$,最小优先级为 $\min P$。初始情况下,$\max P = \min P = H_1.A + 0.01 \times H_1.B$。

b)设历史节点集为 $H = \{H_1, H_2, \cdots, H_n\}$,对应的节点优先级集合 $P = \{P_1, P_2, \cdots, P_n\}$。对于当前节点 H_{n+1},首先,计算 H_{n+1} 的优先级 $P_{n+1} = H_{n+1}.A + H_{n+1}.B \times 0.01$;然后,分三种情况进行处理。

(a)如果 H_{n+1} 的优先级小于集合中所有节点的优先级,即 $P_{n+1} > \max P$,则 $H_{n+1}.C = \max\{H_i.C\}$ $(i = 1, 2, \cdots, n)$,$\max P = P_{n+1}$。

(b)如果 H_{n+1} 的优先级大于集合中所有节点的优先级,即 $P_{n+1} < \min P$,则 $H_{n+1}.C = \min\{H_i.C\}$ $(i = 1, 2, \cdots, n)$,$\min P = P_{n+1}$。

(c)如果 $\min P < P_{n+1} < \max P$,则将 $P_1, P_2, \cdots, P_n$ 按升序排列,得到 $Pa_1, Pa_2, \cdots, Pa_n$,找到 Pa_i, Pa_j,使 $Pa_i < P_{n+1} < Pa_j$。那么,$H_{n+1}.C = Ha_j.C$;然后,对于集合 $\{Ha_j, \cdots, Ha_n\}$,$Ha_k.C = Ha_k.C + 1 (k = j, j+1, \cdots, n)$。

c)更新历史节点集,将 H_{n+1} 并入 H,即 $H = H \cup \{H_{n+1}\}$,继续扫描下一个节点。如果下一节点存在,则跳转至 b),否则,执行完毕。

对例句 S 的一次扫描策略处理过程及中间结果如表 1 所示,最终的索引顺序为 $H_4 > H_1 > H_8 > H_3 > H_6 > H_2 > H_5 > H_7$。

表 1　一次扫描策略处理的过程及中间结果

	当前节点							
	H_1	H_2	H_3	H_4	H_5	H_6	H_7	H_8
优先级	2.02	4.03	3.04	1.05	4.06	3.07	4.08	2.09
$H_1.C$	1	1	1	2	2	2	2	2
$H_2.C$	*	2	3	4	4	5	5	6
$H_3.C$	*	*	2	3	3	3	3	4
$H_4.C$	*	*	*	1	1	1	1	1
$H_5.C$	*	*	*	*	5	6	6	7
$H_6.C$	*	*	*	*	*	4	4	5
$H_7.C$	*	*	*	*	*	*	7	8
$H_8.C$	*	*	*	*	*	*	*	3

注:表 1 中 * 表示随机值,H_1 = ||SZZ2,H_2 = ||||SBL3,H_3 = |||SDJ4,H_4 = |SBL5,H_5 = ||||SBL6,H_6 = |||SBL7,H_7 = ||||SBL8,H_8 = ||STJ9。

2.2　逻辑结构树的生成及其 XML 文本表示

本文中需要生成节点的三种结构类型:逻辑类型、XML 文本类型和 VML 图形元素类型。其中,逻辑类型、XML 文本类型在本节生成,VML 图形元素类型在 2.3 节生成。

节点的逻辑结构定义:LogicNode $= \langle S, R, L, C \rangle$。其中:$S$ 是节点编号;R 是层次关系;L 是所属层次;C 是分句标号集。

XML 文本类型表示的节点,对于内部节点,其形式为〈PNode id = " * " relation = " * " layer = " * " clauseSet = " * " LChild = " * " RChild = " * " /〉;对于叶子节点,其形式为

〈LNode id = “ * ” clauseNo = “ * ” clauseStr = “ * ” /〉

对于每个索引节点,执行以下操作:

a)根据索引节点的划分位置,确定其所属的当前活跃分句标号集,进而确定划分的逻辑节点;然后,根据划分位置,将逻辑节点的分句标号集划分成两个子集,标记子集合为活跃状态,逻辑节点分句标号集标记为非活跃状态。

b)更新逻辑节点层次关系 R 和所属层次 L 的值。

c)生成两个子节点,初始化节点编号 S(记为 S_{index} 和 $S_{index+1}$)和分句标号集 C,然后 index = index + 2。其中 index 表示下一个节点的编号。若子节点的 C 中只有一个元素,则该节点为叶子节点,令该子节点的 R = NULL,L = -1。

d)生成 XML 文本节点。对当前的逻辑节点 LogicNode(i),其对应的 XML 文本节点为 PNode(i),且 PNode. id = LNode(i). S, PNode. relation = LNode(i). R, PNode. layer = LNode(i). L, PNode. clauseSet = LNode(i). C, PNode. LChild = S_{index}, PNode. RChild = $S_{index+1}$。

若当前逻辑节点生成的子节点中存在叶子节点,则生成该叶子节点对应的 XML 文本节点 LNode,且 LNode. id 为叶子节点的编号 S, LNode. clauseNo 为叶子节点的分句标号集 C, LNode. clauseStr 为编号对应分句的文字串。

对于例句,下面以第一次和第二次索引过程为例对方法进行阐释。

初始逻辑节点为 LogicNode(0) = 〈0, * , * ,{1,2,3,4,5,6,7,8,9}〉,index = 1,为方便描述,用 S_i 代表 LogicNode(i),当前活跃分句标号集为{1,2,3,4,5,6,7,8,9}。

第一次索引的是“ISBL5”,因为 5 ∈ {1,2,3,4,5,6,7,8,9},集合{1,2,3,4,5,6,7,8,9}以 5 为分界点划分为{1,2,3,4}和{5,6,7,8,9}。更新 R 和 L,有 LogicNode(0) = 〈0,SBL,1,{1,2,3,4,5,6,7,8,9}〉。生成两个子节点 S_1 和 S_2。S_1 = 〈1, * , * ,{1,2,3,4}〉,S_2 = 〈2, * , * ,{5,6,7,8,9}〉。接下来生成逻辑节点 S_0 的 XML 文本节点〈PNode id = 0 relation = SBL layer = 1 clauseSet = {1,2,3,4,5,6,7,8,9} LChild = S_1 RChild = S_2/〉;最后更新编号 index = index + 2 = 3。

第二次索引的“IISZZ2”,当前活跃分句标号集为{1,2,3,4}和{5,6,7,8,9}。因为 2 ∈ {1,2,3,4},确定划分的逻辑节点为 S_1。集合{1,2,3,4}以 2 为界划分为{1}和{2,3,4}。S_1 = 〈1,SZZ,2,{1,2,3,4}〉。生成两个子节点 S_3 和 S_4。S_3 = 〈3,NULL,-1,{1}〉,S_4 = 〈4, * , * ,{2,3,4}〉。逻辑节点 S_1 的 XML 文本节点的生成这里不赘述。因为 S_3 的分句标号集中只包含一个元素,所以是叶子节点,要生成 S_3 的 XML 文本节点〈LNode id = “3” clauseNo = “1” clauseStr = “尽管苹果公司目前遇到了一些问题”/〉。

2.3　图形元素定位及 VML 节点生成

复句结构树在页面显示时主要由矩形、直线和文本三种图形元素构成,其对应的 VML 标签分别为 v:Rect、v:line 和 v:textbox。所以,复句结构树在页面显示的主要工作就是根据 XML 文本节点,生成页面的 VML 标签节点。

对于内部节点〈PNode id = “ * ” relation = “ * ” layer = “ * ” clauseSet = “ * ” LChild = “ * ”RChild = “ * ” /〉,生成的 VML 标签节点的格式如下:

〈v:Rect style = “top:#rx;left:#ry;width:#w;height:#h;”〉〈v:textbox id = “#id” style = “ * ” inset = “ * , * , * , * ”〉#relation〈br〉#layer〈br〉#clauseSet〈/v:textbox〉

〈v:line from = “x1,y1” to = “x2,y2” style = “ * ” /〉

〈v:line from = “x1,y1” to = “x3,y3” style = “ * ” /〉

〈/v:Rect〉

对于叶子节点〈LNode id = “ * ”clauseNo = “ * ” clauseStr = “ * ”/〉,生成的 VML 标签节点的格式如下:

〈v:Rect title = “#clauseStr”

style = “top:#rx;left:#ry;width:#w;height:#h;”〉

〈v:textbox id = “#styleid” 〉#clauseNo〈/v:textbox〉

〈/v:Rect〉

从 VML 标签元素可以看出,构成复句结构树的各个节点和节点之间的连线要完整地组合在一起,关键是各个页面元素坐标属性值的确定,如 top:#rx;left:#ry;(x_1,y_1),(x_2,y_2),(x_3,y_3)。图 2 给出了复句关系层次树的坐标系以及图形节点坐标。

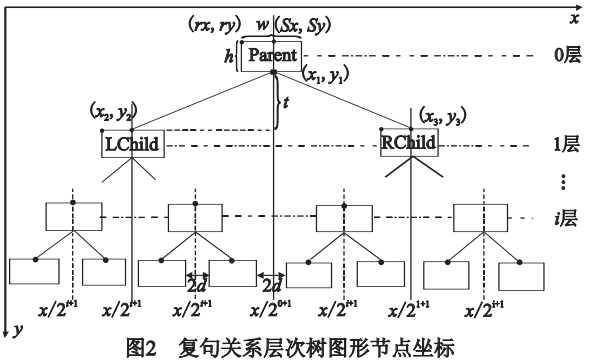

图2　复句关系层次树图形节点坐标

设复句结构树有 n 层,结构树中节点的矩形宽为 w,高为 h,节点间水平间隔为 d,垂直间隔为 t。坐标系原点坐标 coordorig = “0,0”,坐标系大小 coordsize = “x,y”。其中:$x = (2^{n+1} - 2) \times d + 2^n \times w$, $y = (n+1) \times h + n \times (w/2 + d)$。

根节点定位坐标记为(Sx,Sy),有 $Sx = x/2$, $Sy = 0$。

对根节点:

$$\begin{cases} rx = Sx - w/2 \\ ry = Sy \end{cases} \tag{1}$$

$$\begin{cases} x_1 = Sx \\ y_1 = Sy + h \end{cases} \tag{2}$$

$$\begin{cases} x_2 = Sx - x/2^{i+2} \\ y_2 = Sy + h + d + w/2 \end{cases} \tag{3}$$

$$\begin{cases} x_3 = Sx + x/2^{i+2} \\ y_3 = Sy + h + d + w/2 \end{cases} \tag{4}$$

然后寄存两个孩子节点的定位坐标:

$$\begin{cases} SLx = x_2 \\ SLy = y_2 \end{cases} \tag{5}$$

$$\begin{cases} SRx = x_3 \\ SRy = y_3 \end{cases} \tag{6}$$

对于每个内部节点(根节点除外),首先判断其节点类别。若为左孩子节点,则 $Sx = SLx$, $Sy = SLy$。若为右孩子节点,则 $Sx = SRx$, $Sy = SRy$;然后利用式(1)~(4)计算所需的坐标,最后利用式(5)(6)寄存该节点的孩子节点的定位坐标。

对于叶子节点,首先判断其节点类别。若为左孩子节点,则 $Sx = SLx$, $Sy = SLy$;若为右孩子节点,则 $Sx = SRx$, $Sy = SRy$;然后利用式(1)计算所需坐标。说明:(SLx,SLy)(SRx,SRy)分别为某节点对应的父节点寄存的左孩子和右孩子节点的定位坐标。

3　实验结果与分析

为了验证方法的有效性,本文选取了 27 个不同类别的复句进行实验,复句结构树在网页上的显示效果由人工判定。用

准确显示率对可视化方法进行评价,定义如下:

$$准确显示率=\frac{准确显示的个数}{实例个数}\times 100\%$$

其中:准确显示的结果由三个指标决定,分别为层次分析准确、节点定位准确、节点连接准确。对于生成的一个复句层次结构图,如果上述三个指标都满足,则该图被判定为准确显示;如果其中任意一个指标不满足,则该图被判定为不准确显示。实验结果如表 2 所示。

表 2　复句类别及效果

复句重数	实例个数	层次均匀分布个数	层次不均匀分布个数	准确显示率/%
1	2	2	0	100
2	6	4	2	100
3	10	4	6	100
4	6	2	4	100
5	3	0	3	100

从表 2 可以看出,本文的可视化方法对不同重数的复句可以准确地定位和显示,而且,对于一个多重复句中甲乙两个大部分层次分布不均匀的情况,也可以准确地定位和显示。图 3 给出了一个四重复句(文中例句 S)的关系层次树的效果图。

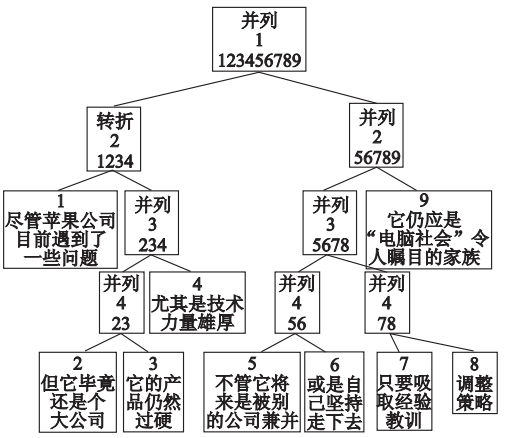

图3　例句S的关系层次树

该结构树是一棵二叉树,由内部节点、叶子节点和节点之间的连线组成。其中:内部节点显示了节点所在的层次级别、节点的层次关系以及节点包含的分句标号,叶子节点显示该分句的标号和分句的内容。当鼠标移至叶子节点区域时,显示叶子节点所代表的分句内容。如图 3 中叶子节点 1 显示的分句为“尽管苹果公司目前遇到了一些问题”。从图中可以看出,通过可视化算法自动生成的关系层次树所体现出来的层次和关系与例句的层次结构和分句间的关系是相同的,这说明了算法的有效性。

4　结束语

虽然本文着重探讨的是复句关系层次树的可视化实现策略,但是对于其他具有层次结构的实体而言(如本体),要实现实体在网页中的可视化,可以在本文讨论方法的基础上作相应的改动即可。

本文中对复句关系层次树采取的是二叉树的表示形式。对于并列结构,当存在两个以上的并列分句时,分句之间应该是平行关系;但本文中显示的是层级结构,虽然关系层次树中对层次关系有详细的标志,但平行关系用层级结构的形式来显示,不太符合人们的思维习惯。例如,一个并列关系复句包含三个并列成分,本文该复句的结构树处理成二层的二叉树形式,而符合习惯的形式是一层的三叉树。下一步的研究工作是采用 m 叉树的形式来显示复句关系层次树,使之显示的结果更加符合人们的逻辑思维习惯。

参考文献:

[1] 姚双云. 小句中枢理论的应用与复句信息工程[J]. 汉语学报, 2005(4):71-96.

[2] 鲁松,白硕,李素建,等. 汉语多重关系复句的关系层次分析[J]. 软件学报, 2001, 12(7): 987-995.

[3] 李晋霞, 刘云. 面向计算机的二重复句层次划分研究[C]//语言计算与基于内容的文本处理. 2003:147-153.

[4] 邢福义. 汉语语法学[M]. 长春:东北师范大学出版社, 2000: 308-311.

[5] 夏立民,王华. 基于 VML 的矢量图形动态生成过程的研究[J]. 计算机技术与发展, 2006,16(11):208-221.

[6] 闫旋,宋瀚涛. 使用 XSL 实现 XML 到 VML 的转换[J]. 计算机应用, 2004,24(s1):321-322.

(上接第 126 页)

[2] MAXWELL T, COSTANZA R. An open geographic modeling environment[J]. Simulation, 1997,68(3): 265-267.

[3] SYDELKO P J, HLOHOWSKYJ I, MAJERUS K, *et al*. An object-oriented framework for dynamic ecosystem modeling: application for integrated risk assessment[J]. Science of the Total Environment, 2001,274(1-3): 271-281.

[4] 陈崇成,王钦敏,汪小钦,等. 空间决策支持系统中模型库的生成及与 GIS 的紧密集成——以厦门市环境管理空间决策支持系统为例[J]. 遥感学报,2002,6(3):168-172.

[5] VOINOV A, FITZ C, BOUMANS R, *et al*. Modular ecosystem modeling[J]. Environmental Modelling & Software, 2004,19(3): 285-304.

[6] WATSON F G R, RAHMAN J M. Tarsier: a practical software framework for model development, testing and deployment[J]. Environmental Modelling & Software, 2004,19(3): 245-260.

[7] 任建武,闾国年,王桥. 多层体系 GIS 与模型集成研究[J]. 测绘学报,2003,32(2):178-182.

[8] HILL C, DELUCA C, BALAJI V, *et al*. The architecture of the earth system modeling framework[J]. Computing in Science & Engineering, 2004,6(1):18-28.

[9] MOORE R V, TINDALL C I. An overview of the open modelling interface and environment(the OpenMI)[J]. Environmental Science & Policy, 2005,8(3): 279-286.

[10] VALCKE S, GUILYARDI E, LARSSON C. PRISM and ENES: a European approach to earth system modeling[J]. Concurrency and Computation: Practice and Experience, 2006,18(2): 247-262.

[11] 王方雄,边馥苓. 基于网格的空间信息原子服务互操作与集成框架[J]. 武汉大学学报:信息科学版,2005,30(2):182-186.

[12] 温永宁,闾国年,杨慧,等. 面向服务的分布式地学模型集成框架研究[J]. 遥感学报,2006,10(2):160-168.

[13] 于海龙,刘丽萍,邬伦. 基于 Web Services 的模型复用研究[J]. 系统仿真学报,2007,19(18):4139-4145.

[14] 王健,王翀,何克清. RGPS 元建模框架在城市交通需求中的应用[J]. 武汉大学学报:信息科学版,2008,33(4):413-417.

[15] 王桥,吴纪桃. 空间决策支持系统中的模型标准化问题研究[J]. 测绘学报,1999,28(2):172-176.