文章编号:1001 - 9081(2009)01 - 0306 - 03

基于 fisheye views 算法解决可视化程序设计语言问题的研究

沈夏炯[1,2],顾 军[2],王 戈[2],董新法[2]
(1. 河南大学 数据与知识工程研究所,河南 开封 475004; 2. 河南大学 计算机与信息工程学院,河南 开封 475001)
(goojune@hotmail.com)

摘 要:可视化程序设计语言已经引起了越来越多学者的研究兴趣,并产生了一系列研究成果。在对编程元素由二维图形对象构成的传统可视化程序设计语言中产生的两个显示问题进行了详细分析,提出引入一种改进的 fisheye views 算法来解决这些问题的方法。并通过设计的原型系统 VPMF 表征了该方法的效果和可行性。

关键词:可视化程序设计语言;二维可视化编程环境;fisheye views; small-screen; scaling-up
中图分类号: TP31 **文献标志码**:A

Research on problems in visual programming languages based on fisheye views algorithm

SHEN Xia-jiong[1,2], GU Jun[2], WANG Ge[2], DONG Xin-fa[2]
(1. *Institute of Data and Knowledge Engineering, Henan University, Kaifeng Henan* 475004, *China*;
2. *College of Computer and Information Engineering, Henan University, Kaifeng Henan* 475001, *China*)

Abstract: The visual problems in traditional 2D visual programming languages where programming elements are represented by 2D graphical objects were analyzed. An improved fisheye views algorithm was introduced to deal with these problems. A prototype system VPMF was developed to show the validity and effectiveness of the approach.

Key words: visual programming languages; 2D visual programming environment; fisheye views; small-screen; scaling-up

0 引言

目前,对于可视化程序设计语言的定义有很多,文献[1]将其定义为一种程序设计语言,这种程序设计语言能够允许用户操控图形化的编程元素来设计程序而非使用文本来实现编程。对符合这种定义的编程语言就称其为可视化的程序设计语言(Visual Programming Languages, VPLs)。另外,这种可视化的程序设计语言区别于我们通常所接触到的微软公司所开发的包括 VB、VC++、VC#以及 VJ++ 等在内的 Visual Studio 系列编程语言。事实上,上述语言还不能称为真正意义上的可视化编程语言,因为它们的基本程序设计方式是一维文本而非图形。

有研究表明,相对于文本来说,人们对于图形信息能够处理的更快、更好[2]。与文本相比,图形本身所具有的直观、象形以及高度信息浓缩等特性使得可视化程序设计为编程人员提供了一种比传统的文本程序设计更为直观的人机交互方式。事实上使用传统的文本编程语言实现编程,某种意义上来说也是一种可视化的编程方式,可以将其称之为一维的可视化编程[3]。这种编程语言由一系列代表一定意义的文本语言符号线性排列构成,用来传达它所代表的某一功能的涵义。而在这种线性的文本表达中,作为程序代码的字符本身是无法反应出其所要完成的功能意义的,因此即使一名经验丰富的程序设计人员如果不去仔细阅读这段代码的话,也无法马上识别出其所要完成的功能。相反,如果能将这段代码以一种恰当的二维或者三维的图形符号表示出来,那么只需略微浏览一下该程序框图,其所要实现的功能就会被清晰准确的识别。因此,人们很自然地对传统的一维文本编程方式进行了扩展,在编程过程中引入了图形对象,而这样的引入无疑是一个不小的进步,它使得非专业的编程人员也能够就自己感兴趣的某一问题,通过对可视化编程环境中提供的图形对象进行适当安排和连接来实现可视化编程。

然而,使用传统的二维图形环境进行可视化程序设计并非毫无缺陷[4-6]。在大规模编程中,含有成百上千个图形节点和将其连接的边可以说是司空见惯。而在编程中使用太多的二维图形对象使得显示屏幕拥塞不堪,从而造成无法清晰灵活的对这些图形表达式进行有效管理和操纵。另一个问题是,有时为了更好的对这些图形对象表达式进行理解,通常允许用户为其添加额外的文本注释信息。这些文本信息虽然帮助这些图形对象被更好的理解,但同时对它们的过分使用可能又会使得该图形对象的意义变得晦涩难懂,以至于影响到对整个程序框架正确理解。

1 传统二维可视化编程环境中的问题

能够提供图形对象来对可视化程序设计语言的图形语法进行创建和修改的工具就称为一个可视化编程环境(Visual Programming Environment, VPE)[7]。目前,包括 Spreadsheets、Visula 以及 NI 公司开发的 LabVIEW 在内的二维图形可视化开发工具都是比较成功的可视化编程环境。在这些二维(2D)VPE 中,图形对象常被用来表达那些用一维文本形式难以表达出来的信息,比如展现静态数据的图表、计算机编程中的数据流图等。然而,在这些传统的 2D VPE 中仍然存在一些问题。

首先,传统的二维可视化编程环境在进行规模较小的程序设计时能够完全胜任,但是当需要将其应用于具有现实意

收稿日期:2008 - 08 - 08。 **基金项目**:河南省教育厅自然科学研究指导计划项目(2008B520004)。
作者简介:沈夏炯(1963 -),男,河南开封人,教授,主要研究方向:软件工程、知识发现、分布式/并行处理、分布式存储、可视化程序设计;顾军(1979 -),男,河南郑州人,硕士研究生,主要研究方向:可视化程序设计。

义的较大规模的程序设计之中时,由于太多的 2D 图形对象阻塞了屏幕空间因而造成无法对其进行适当的管理。而且,即使对这些繁杂的图形对象进行合理的布局安排,仍然无法在可以接受的屏显范围内呈现出整个程序的框架结构。这也是就我们所说的 Small-screen 问题[8],或称为小屏幕空间问题。

此外,从另一个角度来讲,VPLs 允许编程人员对数据关系进行描述和展示而并不是把它们转变为有序的命令、指针或抽象的符号。这往往是一件极其复杂的工作,通常需要耗费一名经验丰富的程序员数月或者一个编程团队几周的时间来完成,而大型程序所需要的图形编程元素由于其数量庞大,背景复杂,上下文过长,往往造成难于理解,这就是传统的 2D 可视化编程环境中的 Scaling-up 问题[9]。

Small-screen 和 Scaling-up 问题的产生使得 VPLs 的发展一度受到的阻碍,但是研究人员很快发现在某些方面三维(3D)图形的表达方式比二维的要好,于是三维可视化编程的语法和其对应的可视化编程环境开始逐渐发展起来。三维可视化编程环境为放置其中的对象增加了额外的一维空间,在一定程度上缓解了这些问题。并且,三维视图特别适合于表达那些包含大量信息的编程情况,而同样的问题如果用二维可视化编程的方式来解决就会显得繁杂而臃肿。然而,在可视化编程环境中,3D 的视觉模式也并不能完全解决问题。因为,通过标准显示器所呈现的 3D 环境的视角有其局限性。例如,虽然通过对 3D 视角的旋转似乎可以包含更多的信息,但是这些信息常常是以一种扭曲的方式来呈现。同时,包含信息的 3D 图标对象也有可能位于观察视角之外,这样就会造成信息对象的不可见。同样,较远距离的图标对象之间也难以进行区分,而且还有可能被离视角较近的对象所遮蔽[10]。

2　fisheye views 算法及其改进

许多在传统 2D VPE 下设计的应用程序,由于 Small-screen 和 Scaling-up 问题的出现使得整个数据关系太过繁杂难于理解以及图形对象的屏幕显示过于拥堵而无法对其进行恰当有效的管理,而且这种现象随着可视化编程的应用已经变得非常普遍。为了解决这一问题,本文引入了 fisheye views 算法。

2.1　通用的 fisheye views 算法

Fisheye views 算法最开始被开发用来提供一种对由大量信息集合组成的概念化布局结构的全局信息和局部信息进行同时显示的方法。而采用 fisheye views 算法的图形工具通过一系列的变形(比如:重新计算图形大小和位置),在对焦点图形对象进行高度细节显示的同时也能够显示出其在整体框架结构中的上下文关系。这对解决传统 2D VPE 中的上述两个问题有很大的帮助。

对 fisheye views 算法进行实施,需要采取两个步骤。首先,需要计算每个图形节点相对于焦点位置的重要度即 DOI (Degree Of Interest)。Furnas 给出了其用于计算 *DOI* 的通用公式如下[11]:

$$DOI(x \mid fp = y) = API(x) - D(x,y) \tag{1}$$

计算出节点 x 相对于焦点 y 的 *DOI*,*API* 为节点 x 的先验重要性,这个值是指定给节点以表示出其在整体框架结构中的相对重要程度,$D(x,y)$ 用来计算节点 x 和当前焦点 y 之间距离。只要在上述必要元素具备的情况下,该方程可应用于任何框架结构,因此称之为通用的 fisheye views 算法。

当获得所有必要的取值后,进入实施该算法的第二个步骤,也就是对 *DOI* 进行描绘与显示。目前有许多可视化技术能够根据计算出的 *DOI* 值进行显示,但需要注意的是在 fisheye views 中定义的距离并不一定指两点之间的实际空间距离,它也可能表示包括概念和语义在内的抽象距离。

2.2　改进的 fisheye views 算法

由于在 2D VPE 中连接 2D 图形节点之间的管道都是水平或垂直的线段,因此需要采用改进的 fisheye views 算法以适应这一特点。在 fisheye views 中的图形节点的大小与其在正常坐标下的大小和它的 *API* 相关, fisheye views 中图形节点的位置又由正常坐标下该节点和焦点之间的距离函数求得,所以我们需要计算出每个节点分别在正常坐标和 fisheye views 模式下的对应取值。对于上述方法,可以使用式(2)计算[12]:

$$P_{feye} = \left(\frac{(d+1)D_{norm_x}}{d \times D_{norm_x} + 1} D_{\max_x} + P_{focus_x}, \frac{(d+1)D_{norm_y}}{d \times D_{norm_y} + 1} D_{\max_y} + P_{focus_y} \right) \tag{2}$$

该公式用来计算在 fisheye views 中图形节点的位置,即 P_{feye},其中 $D_{\max_x}$ 是正常坐标下焦点和边界的水平距离,D_{norm_x} 是正常坐标下待变形节点与焦点的水平距离。同样,对应于纵坐标轴,D_{norm_y} 和 D_{norm_x} 有相似的含义。常数 d 为扭曲因子,不同的赋值使得图标的扭曲程度不同。对应于本系统中取 $d = 0$,表示所有图形节点无扭曲。

$$\begin{cases} S_{feye} = S_{geom}(a \times API)^b \\ S_{geom} = 2\min(\mid Q_{feye_x} - P_{feye_x} \mid, \mid Q_{feye_y} - P_{feye_x} \mid) \end{cases} \tag{3}$$

该公式用来计算 fisheye views 中图形节点的大小,它由其在正常坐标下的大小、位置、*API* 及焦点的位置来构成。Q_{feye} 是在离开焦点方向上远离节点中心的那个点,由公式 1 求得。系数 a 和指数 b 是常量。

$$DTL_{feye}(v,f) = \min(DTL_{\max}(v), \alpha \times S_{feye}(v,f)) \tag{4}$$

一个图形节点能够显示多少细节依赖于它在 fisheye views 中的大小和能被显示的最大细节量。其中 α 为常数。

$$VW(v,f) = \beta S_{feye}(v,f) + \gamma \tag{5}$$

最后,每个图形节点都要分配一个可视化值 *VW*,它由正常坐标下的节点和焦点之间的距离和它的 *API* 构成。其中 β 和 γ 都是常数。

此外,正常坐标下的直连接线段,在改进 fisheye views 算法下当其连接的图形节点发生变形时仍然不会发生扭曲。

3　原型设计

本文采用改进的 fisheye views 算法来解决传统 2D VPE 中存在的 Small-screen 和 Scaling-up 问题。同时,我们设计了一个原型系统 VPMF (Visual Programming Models with Fisheye views)来验证其可行性。

3.1　关键值的计算

计算每个节点的 *DOI* 首先要知道其 *API*。按照每个节点在整个全局程序框架结构中的函数重要性对 *API* 进行计算或赋值。表 1 中列出了部分节点函数的 *API* 作为参考。全局(global)和局部(local)构成了属性(attribute)分类。不同的图形对象或可视化表达式 Vexp (Visual Expression) 按照其重要性被赋予不同的取值。此外,距离函数 $D(x,y)$ 由节点对象

之间的空间距离计算而得。

表1 VPMF系统中对象的*API*值

对象	属性	
	global	local
dataVExp	10	8
variableVExp	10	8
funcVExp	10	8
judgeVExp	10	8
pipeVExp	8	5
pointVExp	8	5
commentVExp	8	3

3.2 原型的实施

该系统采用事件驱动(event-driven)方法来进行实施,用来显示出2D全局编程框架图,同时允许用户用鼠标定位到某一焦点时对其刷新并采用fisheye views进行显示。图1显示了一个用于计算5的阶乘的可视化程序体。图1中,每一个图形对象都在正常模式下显示。数据通过管线从一个可视化表达式流向另一个可视化表达式。

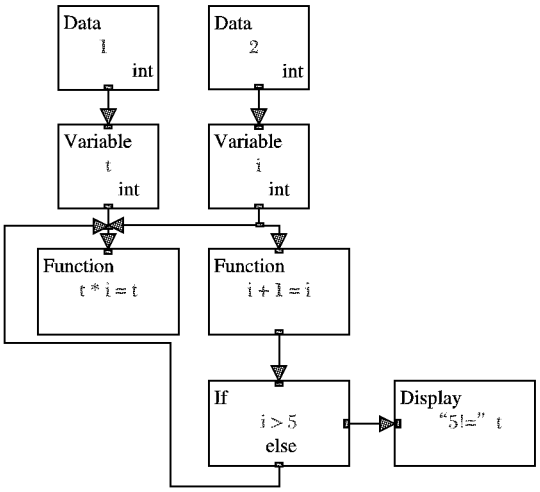

图1 VPMF原型系统中以正常模式显示的计算5!的可视化程序体

图2和图3分别展示了当焦点集中与"Function"和"If…else"可视化表达式图形对象时的fisheye views模式显示状态。可以看到,聚焦处的可视化表达式以较大面积显示出其所包含的细节信息,而旁边的节点均按照不同比例受到了压缩。

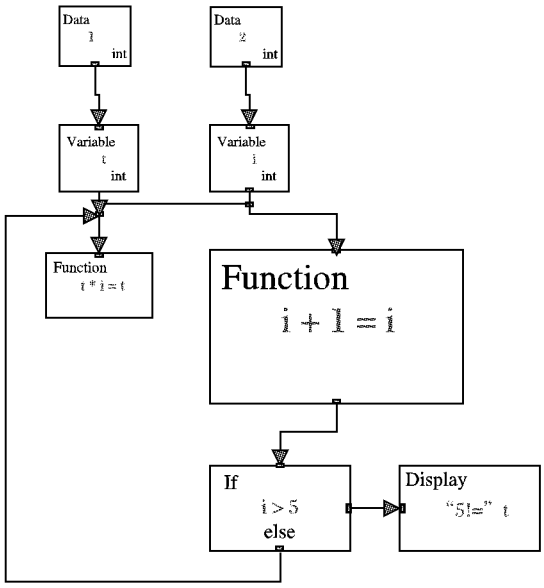

图2 焦点为"Function"的fisheye views显示图

可以看到,当一个可视化的程序体随着规模的不断增大以至于单个屏幕无法完全显示时,该原型系统可以提供一种有效的视觉模式来同时显示出该程序框图的整体结构和单位细节,采用这种fisheye views视觉模式可以有效地解决传统2D可视化程序设计语言中的small-screen和scaling-up问题。

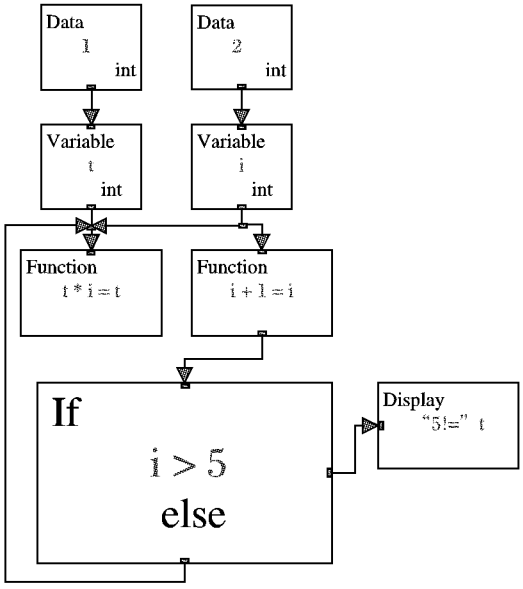

图3 焦点为"If…else"的fisheye views显示图

4 结语

本文引入的这种改进的fisheye views算法通过在单个窗口中同时显示出2D图形对象的细节信息及其在整体结构中的上下文关系,从而提供了一种有效的机制来解决Small-screen和Scaling-up问题。通过实验和分析,该算法能够解决传统二维可视化语言设计环境中存在的这些问题。然而,一些潜在的问题仍然有待解决,包括如何在单个框架中对多个焦点进行同时显示等。另外,*API*的取值也可能会随着程序体的规模而随之变化[13-15]。这些问题仍需要进一步探讨。

参考文献:

[1] BURNETT M M. Visual programming[M]. Encyclopedia of Electrical and Electronics Engineering. New York: Wiley, 1999.

[2] RAEDER G. A survey of current graphical programming techniques [J]. IEEE Computer, 1985, 18(8):11-25.

[3] COX P T, GILES F R, PIETRZYKOWSKI T. Prograph: A step towards liberating programming from textual conditioning[C]// IEEE Workshop on Visual Languages. [S. l.]: IEEE Press, 1989:150-156.

[4] NAJORK M A. Programming in three dimensions[J]. Journal of Visual Languages and Computing, 1996(7):219-242.

[5] GLINERT E P. Out of Flatland: towards 3-D visual programming [C]// Proceedings of the 1987 Fall Joint Computer Conference on Exploring Technology: Today and Tomorrow. Los Alamitos, USA: IEEE Computer Society Press, 1987:292-299.

[6] WARE C, FRANCK G. Viewing a graph in a virtual reality display is three times as good as a 2D diagram [C]// Proceedings of 1994 IEEE Conference on Visual Languages. [S. l.]: IEEE Computer Society Press, 1994:182-183.

[7] GOLDBERG A, BURNETT M, LEWIS T. What is visual object-oriented programming [M]. Greenwich, CT, USA: Manning Publications, 1995:3-20.

[8] HENDERSON D A, CARD S K. Rooms: The use of multiple virtual workspaces to reduce space contention in a window-based graphical user interface[J]. ACM Transactions on Graphics, 1986, 5(3):210-243.

(下转第336页)

验证。

这交给 XML 解析器去执行即可,只要严格按照 Schema 文档的规范来定义 XML,就可以通过 Well formed 和 Valid 验证。

2)约束条件的验证。

这部分必须通过 DOM 接口,结合编程语言(VBScript、JavaScript、Java、C++等)来编程实现,又可分为以下几个部分:

a)引用约束。

指 UserRoleAssignment.xml,RolePermissionAssignment.xml 中引用的 user、role 和 permission 在 User.xml、Role.xml 和 Permission.xml 中要先定义。

b)前提权限约束。

角色继承树的根元素(角色)权限在给它的子孙角色分配权限前必须已指派。

c)唯一性约束。

每个角色只可进行一次用户分配。指 UserRoleAssignment.xml 中属性 role 的值唯一。

每个角色只可进行一次权限分配。类同上一个约束。

给角色指派的权限必须唯一。指 Permission.xml 中同属一个 permissionGroup 的权限,只能选其中的一个指派给角色。

多重继承中如果继承了多个同名权限,则必须在子类角色中覆盖该权限,使得该权限唯一。通过重新指派一个 permissionGroupName 相同的权限来实现。

d)互斥角色。

通过 Role.xsd 中的 ssd 和 dsd 属性来限定用户不能同时分配给哪些角色。

e)基数约束。

通过 User.xsd 中 cardinality 属性来限定某用户所能扮演的角色数目,Role.xsd 中 cardinality 属性来限定扮演某一角色的用户数目。

通过 DOM 接口加载 XML 文档,就可得到一个 DOM 对象,它根据 XML 文档中元素、属性和值的组织方式形成一棵 DOM 树,而通过对 DOM 树的遍历就可实现约束条件的验证。

4 新模型在系统访问控制中的应用

基于面向对象特性和 XML 的 RBAC 模型在系统访问控制中的应用如图3所示。

管理员通过应用服务器向中心服务器发出 RBAC 服务请求,中心服务器加载 RBAC 模型,将相应的用户、角色和权限信息传回应用服务器,然后应用服务器通过RBAC信息对应用系统进行权限控制。

新 RBAC 模型主要应用于分布式网络环境下的系统访问控制,如在线考试系统、电子商务购物系统、大型企业的企业信息门户[7](Enterprise Information Portal,EIP)等。

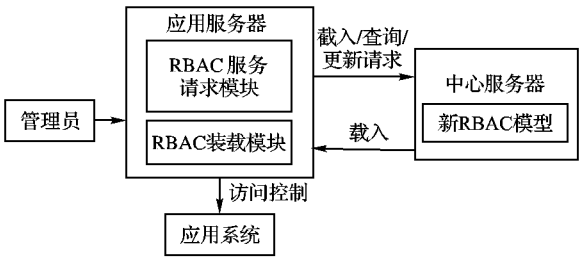

图3 基于面向对象特性和 XML 的 RBAC 模型在系统访问控制中的应用

5 结语

本文通过结合 XML 的数据表示结构化和面向对象的继承、封装特性,提出了一种基于面向对象特性和 XML 的 RBAC 模型,与传统的 RBAC 模型相比,新模型克服了私有角色的问题,通过 DOM 接口对模型的操作,进一步减轻了权限分配的负担,且适用于分布式网络环境下的系统访问控制。

下一步的研究:1)尝试通过线索二叉树来重新生成角色树,弥补 DOM 树效率不高,内存耗费较大的缺点;2)根据实际需要,进一步完善模型的有效性验证。

参考文献:

[1] SANDHU R, COYNE E. Role-based access control models [J]. IEEE Computer, 1996, 29(2): 38 - 47.

[2] FERRAIOLO D, SANDHU R, GAVRILA S, *et al*. Proposed NIST standard for role-based access control [J]. ACM Transaction on Information and System Security(TISSEC), 2001, 4(3): 224 - 274.

[3] 杜萍,刘弘. 协同设计系统中基于 XML 的访问控制实现 [J]. 计算机应用研究, 2007, 24(1): 174 - 176.

[4] WU DI, LIN JIAN, DONG YABO, *et al*. Using semantic Web technologies to specify constraints of RBAC [C]// PDCAT 2005: 6th International Conference on Parallel and Distributed Computing, Applications and Technologies. [S.l.]: IEEE Press, 2005: 543 - 545.

[5] 钟华,冯玉琳,姜洪安. 扩充角色层次关系模型及其应用[J]. 软件学报, 2000, 11(6): 779 - 784.

[6] 陈志炜,金远平,刘莹莹. 面向对象技术在基于角色的访问控制中的应用 [J]. 计算机应用, 2003, 23(12): 121 - 123.

[7] 耿晖,王海波. 基于 XML 的角色访问控制(RBAC) [J]. 计算机应用研究, 2002, 19(12): 14 - 16.

(上接第308页)

[9] BURNETT M M, BAKER M J, BOHUS C, *et al*. Scaling up visual programming languages[J]. IEEE Computer Society, 1995, 28(3): 45 - 54.

[10] CHUNG V W H. 3D Composer - A visual builder for 3D notations [EB/OL]. [2008 - 05-01]. http://www.cs.auckland.ac.nz/research/theses/1999/chung_vincent_thesis1999.pdf.

[11] FURNAS G W. Generalized fisheye views [C]// Proceedings of ACM CHI '86 conference on Human Factors in Computing Systems. [S.l.]: ACM Press, 1986: 16 - 23.

[12] SARKAR M, BROWN M H. Graphical fisheye views of graphs [C]// Proceedings of the ACM SIGCHI '92 Conference on Human Factors in Computing Systems. [S.l.]: ACM Press, 1992: 6 - 11.

[13] BARTRAM L, HO A, DILL J, *et al*. The continuous zoom: A constrained fisheye technique for viewing and navigating large information spaces [C]// Proceedings of the ACM Symposium on User Interface Software and Technology. [S.l.]: ACM Press, 1995: 207 - 215.

[14] LOKUGE I, ISHIZAKI S. GeoSpace: An interactive visualization system for exploring complex information spaces[C]// Proceedings of ACM CHI '95 Conference on Human Factors in Computing Systems. [S.l.]: ACM Press, 1995: 409-414.

[15] RUGER M, PREIM B, RITTER A. Zoom Navigation [M]// STROTHOTTE T. Computational Visualization: Graphics, Abstraction, and Interactivity. Berlin: Springer-Verlag, 1998: 161 - 174.