笔式用户界面开发工具研究*

栗 阳[1+], 关志伟[2], 戴国忠[1]

[1](中国科学院 软件研究所 智能工程实验室,北京 100080)

[2](美国国家研究委员会 美国海军研究生院 软件工程自动化中心,美国)

Research on Development Tools for Pen-Based User Interfaces

LI Yang[1+], GUAN Zhi-Wei[2], DAI Guo-Zhong[1]

[1](Intelligence Engineering Laboratory, Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China)

[2](Software Engineering Automation Center, Naval Postgraduate School of National Research Council of U.S., America)

+ Corresponding author: Phn: 86-10-62552013, E-mail: yangli@ieee.org

Received 2002-04-28; Accepted 2002-06-19

Li Y, Guan ZW, Dai GZ. Research on development tools for pen-based user interfaces. *Journal of Software*, 2003,14(3):392~400.

Abstract: Pen-Based user interfaces provide more natural interactions for users. However, it is difficult to be constructed because a usable pen-based user interface generally involves multiple areas and interdisciplinary knowledge. In this paper, the design and implementation of a toolkit, named Penbuilder, is described systematically, supporting the development of pen-based user interfaces. It is designed based on the attributes of pen interaction and ubiquitous computing. It offers high-level supports to developers. Based on Penbuilder, several typical prototypes of pen interaction have been constructed, which demonstrates the capability of Penbuilder for easy construction and fast-prototyping of pen-based user interfaces.

Key words: pen-based user interface; toolkit; Penbuilder; SketchPoint

摘 要: 笔式用户界面提供给用户更为自然的交互方式,然而,笔式用户界面的构造是一项非常困难的工作,一个可用的笔式用户界面系统往往需要多领域、多学科的知识.系统地论述了一个支持笔式用户界面开发的工具系统 Penbuilder 的设计实现.它基于笔交互的特性以及无处不在的计算环境的要求而设计,为笔式用户界面的开发提供高级的支持.基于 Penbuilder 的支持,设计开发了一批典型的笔式用户界面原型系统.该研究为笔式用户界面的构造与快速原型提供了有力的支持.

关键词: 笔式用户界面;开发工具;Penbuilder;SketchPoint

中图法分类号: TP311 **文献标识码**: A

笔式用户界面(pen-based user interfaces,简称 PUI)是后 PC 时代的一个主要界面形式,它通过纸和笔(pen-paper)的交互隐喻给人以极大的自然性,通过自由勾画、手势等交互方式,用户可以实现自然而高效的交互.

* Supported by the National Natural Science Foundation of China under Grant No.60033020 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2001AA114170 (国家高技术研究发展计划)

第一作者简介: 栗阳(1974－),男,陕西榆林人,博士,主要研究领域为人机交互,用户界面设计,无处不在的计算.

然而,笔式用户界面的特点也使得传统的用户界面软件工具不能满足它的要求,因为它的交互不再基于一组既定的交互控件(如菜单和按钮),除了常用的点击(click)等离散交互事件以外,更多的交互是以连续的、隐式的方式完成的.因为笔式用户界面的构造往往涉及许多不同领域的知识,如软件工程、模式识别、计算机图形学和认知心理学等,所以它易于使用,但难以构造.笔式用户界面开发工具将会极大地提高设计和开发的效率[1,2].根据笔式用户界面的特点,我们设计并实现了一个支持笔式用户界面开发的工具箱系统 Penbuilder,并基于该工具箱系统构造出了一批典型的笔式用户界面应用程序.

在笔式用户界面工具的研究中出现了许多的商用产品或研究性原型系统[3~7],商用系统基本上是在 GUI 的界面中嵌入了笔交互,提供了非常低级的支持,而把大量的工作留给了界面开发者.在研究领域中,SATIN[2]是最为复杂的支持笔式用户界面开发的工具系统.SATIN 支持具有非正式界面风格的笔式用户界面开发,它提供了大量基于电子墨水(digital ink)的操作,使用了常用的图形组织技术 SceneGraph[8,9]来组织界面对象,如笔划.同时采用了 Kramer 的 Translusent Patch 的思想来设计用户界面[10].SATIN 中提供了多视图的机制,并引入了可缩放用户界面(zoomable user interface)[11]的思想来访问图形对象.SATIN 是较为成功的一个笔式用户界面工具系统,但 SATIN 的事件处理策略对复杂交互的设计略显不足.而且庞大的工具系统对于开发者而言,也需要较多的时间来理解和掌握.此外,它对于无处不在的计算所需要的分布式计算、界面的自适应性以及 Ink 的结构化和共享的特性支持不足.

SATIN 体现了许多先进的设计思想,但是一个庞大的面向对象工具系统往往是难于学习和使用的,它需要界面开发者较多的学习努力.尤其是当界面所需的对话控制十分复杂时,对于界面场景的事件管理往往过于复杂,也使得界面开发者不能清楚地了解系统的运行机理,进而难以设计出复杂的笔式用户界面.此外,随着笔交互设备的发展,笔输入信息不再局限于两维的位置信息(x,y),而是可以提供如压力和方位等多维交互信息,SATIN 只支持等价于通常鼠标事件的信息类型,并不能充分利用笔输入设备的信息.本文给出了支持自由笔式用户界面开发工具箱系统 Penbuilder 的设计实现,它从底层到高层完全基于笔交互的特性进行设计.具体的设计目标主要体现在:易于掌握的系统架构、支持各种笔交互信息的处理、支持灵活的事件处理、支持跨平台的面向无处不在的计算的笔计算环境.下面,我们主要从 Penbuilder 的设计目标、设计思想、系统体系结构、事件模型、绘制模型和语义模型等方面进行论述.

1 Penbuilder 的设计目标

- 自由图形的管理.进行自由的勾画是笔式用户界面的一个重要特征,所以界面中常常有大量的、多层次的自由图形,这些图形通常没有规则的形状.而交互中涉及许多对于这些图形的操作,从而导致对这些图形的操作以及图形关系的维护显得尤为困难.工具系统是否能够提供高效的图形管理是其性能的一个重要标志.
- 对于基本数据类型 Ink 的支持.上述自由图形其实就是所谓的 Ink 的表象,Ink 是笔式用户界面的基本组成,它是界面中信息表示的主要手段.随着笔交互的日益流行,Ink 愈来愈成为人们所关注的焦点.如何提供对于 Ink 的高级支持,而不是在一个两维点的数组上进行操作是笔式用户界面的一个关键问题.
- 支持增量式意图提取.在文献[12,13]中讨论了增量式意图提取的思想和方法,它是对于用户非精确交互的一个有力支持.在 Penbuilder 的设计开发中,将会贯彻这些思想,使得界面开发者能够方便地受益于增量式交互意图的提取.
- 支持高级的笔交互事件.虽然目前的笔交互设备提供了远远多于鼠标的信息,但是许多工具箱系统仍然把笔当作鼠标来看待,没有给用户提供笔的专用的开发接口,从而未能充分地利用笔的交互自然性和丰富的交互意图.同时笔交互是一个以连续交互为主的交互方式[14],然而目前暴露给界面开发者的仍然是离散的、低级的事件.Penbuilder 将力求提供一个面向笔交互的、设备无关的高级事件模型.
- 自适应性.自适应性主要包含 3 个方面,首先是对交互环境的适应.笔交互环境是多种多样的[15],从手持 PDA、桌面机到白板系统,不同的交互环境往往具有不同的硬件机制和底层软件支持,如何在体系结构上支持多种多样的交互环境是非常重要的;其次是对于用户的适应性,笔交互中包含了大量的个人特性,如何提供机制使得界面开发者容易实现具有适应用户特性的笔交互界面也是非常重要的;最后是计算环境的适应,传统的用户

界面主要以单机计算为主,然而笔交互界面常常会在分布适的计算环境中进行,所以如何在体系结构上满足应用系统的分布计算需求是一个值得考虑的问题.

• 平台的可扩充性.笔式用户界面尚没有标准形成,研究人员从各个领域出发进行研究,提供了大量的新技术和交互设备,开发平台必须保证其可扩充性才有生命力,必须提供方便的机制,使得新的技术、算法或交互设备可以容易地加入,如多种多样的识别算法或图形处理技术,良好的设计可以获得新技术的即插即用(plug and play,简称 PNP).

• 灵活性.灵活性和工具的高级程度是一个两难的问题,在提高工具的高级程度的同时将意味着丧失灵活性,如何获得二者良好的折衷是设计中的一个重要议题.工具对于应用语义的表达是一个非常困难的问题,太多的支持会使得界面开发者有很多的束缚.在提供高级支持的同时,保留灵活性是 Penbuilder 的一个设计目标.

• 易学性.任何一个强大的工具,如果使得开发者很难掌握,那么它仍然是失败的[1].一个工具系统必须有一个简单易懂的理论模型,这样才会让开发者有整体的认识.目前,面向对象工具箱的一个主要问题是开发者很难快速地掌握,因为复杂的类继承关系使得开发者为了了解一个类而需要了解它的所有父类.Penbuilder 希望通过域模型和框架结构等技术降低界面开发者学习的负担.

2 Penbuilder 的设计思想

纸笔(pen-paper)的界面隐喻是 Penbuilder 设计思想的主要基础,设计中结合了面向对象技术、域模型的思想和 SceneGraph 的图形管理技术.Penbuilder 将界面中的信息分为两种,一个是域,另一个是域中的内容.一个域可以理解为是一张纸,它具有自己的特性,即外观、行为和语义,每个域可以有自己独特的抽象交互设备和空间管理,域与域之间的关系是通过 SceneGraph 技术来组织的,而域内的内容是以 Ink 为主(它也是 Penbuilder 中的 light-weight component,轻量级对象),可以理解为“纸上的笔迹”,这些 Ink 可以依据线性的(其实为 SceneGraph 的一个特例)或 SceneGraph 的方式组织,以此来适应不同的情况.

通过域模型的方式对笔式用户界面进行组织,即整个用户界面是一个域,而这个域又是由一些子域构成,依此类推继续细分,每个子域往往针对于一个具体的问题,这样的组织方式自然而然地支持了软件工程的问题逐步求精过程(如图 1 所示).

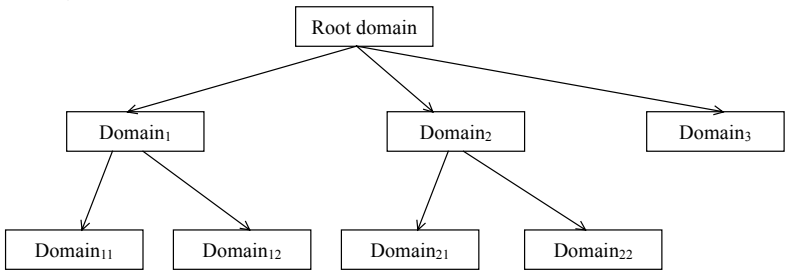

Fig.1 Domain model based design

图 1 基于域模型的设计思想

这里,我们将一个域定义为如下五元组:Domain=(device,style,area,behavior,ink).

其中 device 是指这个域中使用的交互设备抽象,如鼠标(可以产生轨迹信息)、Tablet 或白板环境中的笔,它们可以基于各种底层技术实现,如电磁、Camera(computer vision)或无线定位等方式.Penbuilder 通过这个成员来实现在一个笔交互环境中同时支持多个笔交互设备.Style 是对一个域对象中交互风格的描述,如颜色、透明等信息.它表现了一个域外观上的风格,它是交互反馈相关的描述.Area 确定了这个域在可视界面上的区域,因为一个域往往表现为屏幕上的一块区域,它可以是任意封闭图形区域,可以抽象地将其理解为“一片纸”,它也确定了这个域中所有对象的坐标系.Behavior 是一个域的行为描述,针对于每个特殊的问题都有相关的行为描述,它实现了交互动作在特定上下文中的解释,是域对象最为重要的一个动态属性.Ink 是一个域的数据存储对象,它实现了对这个域的数据管理,其中包括对子域对象的管理.通过这个机制,Penbuilder 实现了域对象树的嵌套层次关系.Ink 本身由前边所说的轻量级对象构成.这些域的属性影响了整个笔式用户界面的设计.

3 Penbuilder 的基本组成

依据上述设计思想,Penbuilder 以面向对象的方法进行构建.目前它主要由 5 个部分构成:基本对象库、事件库、动作库、意图分析库和交互环境库.基本对象库主要支持域模型和域内 Ink 的表示和操作.事件库和动作库是对笔交互事件的高级支持,实现各种复杂的动作支持,如拖动、连续点击等.意图分析库主要提供了识别和知识推理的支持.交互环境库用于提供一个虚拟的笔交互环境,使得应用程序独立于交互设施细节.本节主要介绍基本对象库,其对象模型如图 2 所示.

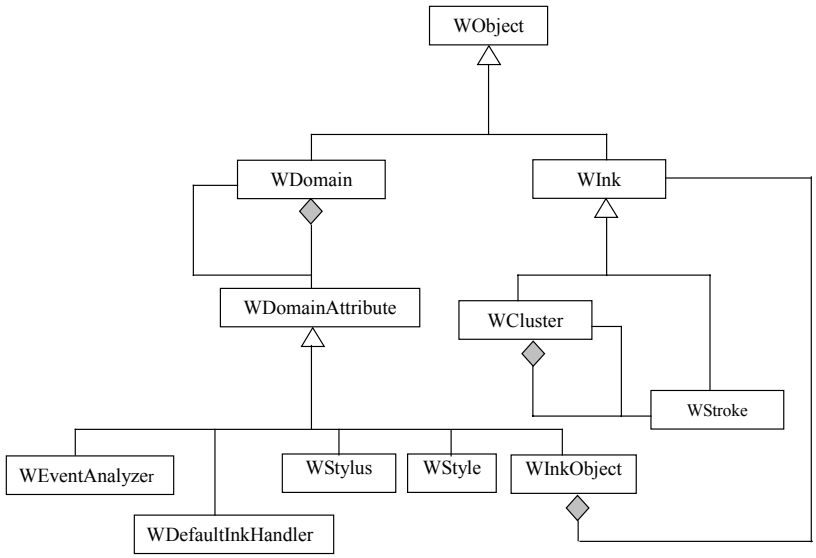

Fig.2 Object model of fundamental object library

图 2 基本对象库对象模型

WObject 是一个抽象的类,由它派生出了轻量级对象(域内对象)和重量级对象(域级的对象).WDomain 表示一个域,依据上节中域的五元组思想,它包含了一组域属性(WDomainAttribute),如行为(WEventAnalyzer)、交互设备(WStylus)、交互风格(WStyle)和 Ink(WdefaultInkHandler,WInkObject)等.界面开发者可以通过配置或扩展这些属性来定制自己需要的域对象,以满足应用领域和交互环境的具体需求.除此之外,WDomain 的对象也可以包含子域对象(WDomain),从而实现了对于应用问题求解中基于域的递归表示.界面中的轻量级对象以 WInk 为基类,派生了 WCluster 和 WStroke 两个类,而 WCluster 是由 WStroke 聚合而成的,即一个笔划簇中包含多个笔划.这里,WCluster 的对象也可以包含 WCluster,即支持笔划簇的递归表示,从而形成了 Ink 的层次性结构(这里也称为 Cluster-Stroke 的结构).

4 运行时结构

在 Penbuilder 1.0 中,主要关注了 Domain 之间的关系[5].基于 Penbuilder 1.0 的工作,Penbuilder 2.0 关注于域内部的信息组织和处理,一个域的运行时结构如图 3 所示.事件通过 Virtual Pen 之后形成了抽象的笔交互事件,Virtual Pen 是域的 device 属性中重要的一部分,它封装了笔交互装置的细节,使得系统独立于具体设备,而 Virtual Paper 是一个虚拟的反馈设备,它负责将抽象的反馈请求转化为显示设备相关的作图原语.这使得系统可以使用虚拟的作图原语进行反馈,而不需要了解反馈设备的细节,这都为界面的自适应性、面向无处不在的计算环境创造了条件.事件分析器(event analyzer)是一个非常重要的部件,它负责把低级的笔交互事件分析转化为高级的交互意图,它主要依据后边将要介绍的事件模型和意图推理机制来工作.系统运行时有些交互事件可以直接进行反馈而不需要分析,例如,基于 Tablet 的笔设备(输入板与显示器分离),当笔移动时屏幕中的光标也随之移动,这其实是词法级的反馈.事件分析器有时也会将交互事件向其子域传播.

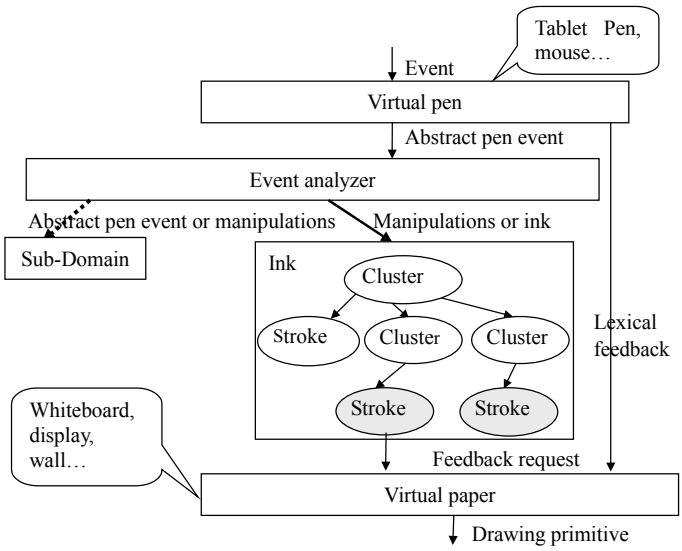

Fig.3 Runtime architecture

图 3 运行时结构

5 笔交互的事件模型

这里所说的笔交互的事件模型并非是指面向对象技术中的事件模型,而是对笔交互事件及其处理的抽象.笔交互作为一种以连续交互为特征的交互方式,与传统的基于鼠标和键盘的交互有着截然不同的区别.鼠标的离散事件模型不再适用,我们需要另外构造一个适合笔交互的高级事件模型.

在 Penbuilder 1.0 中已经定义了抽象的事件格式,并给出了交互事件逐层抽象的机制[5].通过笔交互设备的丰富信息,可以表达人的多个效应和感知通道,这其实也可以看做是一种多通道的交互[16].笔交互设备主要有以下几种输入信息:位置、姿势、压力、按键:“位置”用 *x*,*y* 来刻画;“姿势”用两个分量表示(如图 4 所示):azimuth 表示笔绕输入板法向量的值,altitude 表示笔与输入板的夹角(笔尖向下为正值,否则为负);“压力”由两个量表示:对输入板的法向压力和切向压力;“按键”通过一个枚举类型表示通常的鼠标按键操作.

基于 Penbuilder 1.0 的事件格式,Penbuilder 2.0 对事件进一步处理,即在域属性 behavior 中对事件进行分析.这里,我们给出域对象所使用的缺省事件处理器,即一个事件分析树.如图 5 所示.

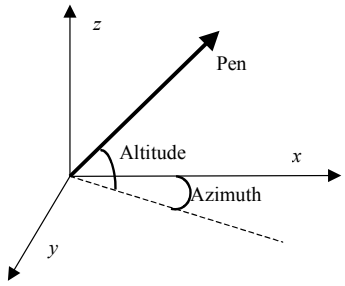

Fig.4 Orientation of pen

图 4 笔的姿势示意图

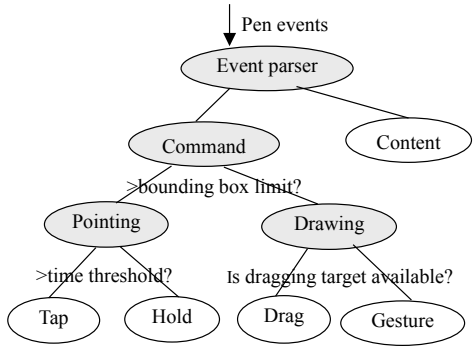

Fig.5 Event analyzing tree of Penbuilder 2.0

图 5 Penbuilder 2.0 中的事件分析树

分析时,对该分析树进行从顶向下、由左向右的遍历,该分析树支持一组通用的笔交互方式,如手势

(gesture)、点击(tap)、hold(按住笔并停留一段时间)、拖动(drag)等,使得界面开发者可以方便地重用这些特征.一个笔输入事件可能是交互命令(command),如手势;也可能是被操作的内容(content),如文字图形.对于大多数应用程序而言,命令是可枚举的小模式空间(有限的几种操作方式),而内容通常是不可枚举的.所以分析树缺省先将事件交由命令解释器进行处理,之后,交由内容分析器处理.该分析树具有动态可配置性,即遍历的策略和条件可以在应用系统运行时确定.图 5 中的灰色节点(Event Parser,Command 和 Pointing)提供了可配置的接口,使得应用系统可以在任何时候改变遍历的策略.例如,可以通过改变 Command 中边界框的限制,来改变对点事件和绘制交互的分析,也可以通过改变 Pointing 中 Time Threshold 的大小调整对于 Hold 事件和 Tap 事件的区分,以适应不同的用户习惯.整体上来讲,这棵分析树是一个笔交互事件分析的骨架,即分析过程中的一个控制结构,它可以与识别器或其他处理引擎相连,从而实现领域化、应用化.

这里我们给出笔式用户界面中几个典型的交互动作的形式化描述.首先对相关符号进行解释,penup 是指笔抬起,pendown 是指笔按下,pendrag 是指笔在压力大于零时的移动;*box*(*x*)是指 *x* 的边界框,*duration*(*x*)是指 *x* 所占用的时间,*elapse*(*x*,*y*)是指 *x* 与 *y* 之间的时间间隔;MAX_BOX 是最大边界框常量,MAX_DUR 是点击事件的最大可占用时间,MAX_INTERVAL 是“连续点击”动作中“点击”间的最大可允许时间间隔(双击为一个特殊的“连续点击”动作).

笔划(stroke):笔划是笔式用户界面中的基本单位,它可以被定义为:笔输入设备依照时序来采样,在笔按下和抬起之间的一组在多维信息空间中的点(包含轨迹、压力和姿势)的序列称为一个笔划.

$$\text{Stroke: } \{s \mid \forall s \in \text{pendown} \cdot \text{pendrag}^* \cdot \text{penup}\}.$$

点击(Tap): $\{s \mid \forall s \in \text{Stroke}, box(s) < \text{MAX\_BOX} \wedge duration(s) < \text{MAX\_DUR}\}$.

$$\text{Hold: } \{s \mid \forall s \in \text{Stroke}, box(s) < \text{MAX\_BOX} \wedge duration(s) \geq \text{MAX\_DUR}\}.$$

连续点击: $\{c \mid \forall c \in t \cdot t^+ \wedge t \in Tap \wedge elapse(t_i, t_{i+1}) < \text{MAX\_INTERVAL}\}$.

手势(gesture):手势即可以触发一个命令执行的一组笔划.它的形式化定义如下,这里描述的是一个多笔划手势,*linkage*(*x*,*y*)表示 *x* 与 *y* 之间是否存在的语义约束,即是否属于同一个手势,而 *semantics*(*x*)为手势 *x* 是否能够触发一个命令(如通过模式识别,是否是一个可接受的手势,或是否符合上下文).

$$\{g \mid \forall g \in s^+ \wedge s \in \text{Stroke} \wedge box(s) \geq \text{MAX\_BOX} \wedge linkage(s_i, s_{i+1}) = \text{TRUE} \wedge semantics(g) = \text{TRUE}\}.$$

6 绘制模型

界面的反馈是人机界面的重要环节,它使得人在交互过程中的输入和输出形成一个回路,界面反馈的效果直接影响着用户界面的可用性.对于笔式用户界面,视觉反馈是交互的主要反馈形式,如交互中 Ink 的反馈(如在屏幕和输入板分离的情况下,还需要有笔位置变化的反馈——笔的光标).反馈是用户交互的结果的反映,有词法级的、语法级的和语义级的,其中涉及了大量的自由图形变换和绘制.复杂的笔式用户界面应用系统往往需要复杂的图形处理,通过使用 SceneGraph 技术对界面对象进行组织,可以对各种自由图形元素进行有效的操作.在进入每个域的绘制时,需要先将这个域的属性(如局部坐标空间、颜色等)压入绘制属性堆栈,在绘制完成之后弹出堆栈.在每个域内部的绘制可以是界面开发者自己定制的,也可以是工具系统提供的对于笔划簇和笔划的基本支持.

Penbuilder 的绘制机制提供了高级的绘制原语,如针对于笔划的绘制和笔划簇的绘制,同时提供了对绘制性能进行控制的方法,如是否反走样或是否平滑处理,以及是否需要通过使用双缓冲技术(double buffering)对界面绘制的闪烁进行控制等.为了做到绘制引擎的无关性(在无处不在的计算环境中,这个性质尤其重要),Penbuilder 使用了分布式窗口系统 XWindow 和 OOIGT[8,9]的思想,设计了一个虚拟绘制服务器,将抽象的绘制原语转化为具体显示环境中的绘制操作,如 Windows GDI,Java AWT 和 XWindow Xlib 等.同时,Penbuilder 2.0 也使用了 OOIGT 和 OpenInventor 中的绘制属性的动态继承机制,使得界面开发者可以对界面绘制进行有效的控制,还节约了系统资源.

7 语义模型

我们将应用程序的内部特征,如数据管理和应用语义处理的相关内容在语义模型中加以实现,语义模型是应用语义相关的.在 Penbuilder 2.0 中,界面开发者可以通过在事件分析树的各个节点上设置语义处理函数,或者通过动态配置事件分析树来体现应用相关的特征.对于应用系统内部语义的管理通过域模型动态地分布在各个域节点中,每个域的 Ink 属性正是为这个目的而设置的,通过对它的配置可以实现应用领域相关的语义特征.目前,Penbuilder 2.0 的研究正在进行以 XML 为主的应用语义管理.这就需要对 Ink 这种特殊的数据类型进行基于 XML 的建模,它也将是 Penbuilder 在无处不在的计算环境中的底层支持,因为,基于 XML 的数据建模方式将数据的语义和显示进行了有效的分离,使得数据的表示和操作能够获得较好的可伸缩性.下面,我们主要从意图提取和 Ink 的结构这两个方面进行阐述.

7.1 意图提取

Penbuilder 提供了识别所需的框架和标准接口,使得系统可以方便地加入各种识别器.这里采用了 SATIN 的思想,通过工具系统的识别管理器对各个识别器进行管理和调度.这些识别器可以支持不同的可识别集合,整个系统的可识别集合是它们的并集.而它们在流程中的顺序说明了它们在交互上下文中的优先权.通过调整优先权、禁止或开启某一识别器,可以做到表示与解释(语义)的动态绑定.Penbuilder 中对于 Ink 或手势的分析目前主要分为 4 类:

- 通过提供一些基本的分析函数来实现:如求笔划中的折点(corner)、笔划的封闭性(closure)、笔划的邻近性(adjacence)、包含性(containment)等.开发者可以通过这些函数进行灵活的分析以满足相应的需要,这是一种白盒式分析.
- 通过专用的手势识别器:开发者可以使用 Penbuilder 所提供的识别器进行识别,获得的结果是识别的类型和准确度(概率值).
- 专用的文字识别器:系统纳入了文字识别引擎,界面开发者可以通过系统提供的接口方便地使用手写文字识别的特征.
- 对于高级的分析可以采用 Bayes 网络进行推理:对底层的、局部的分析(如识别)所获得的概率值和类型信息可作进一步推理,从而将这些信息带入到更高级的、全局的认知分析中.

7.2 Ink的结构和First-Class Data Type

无处不在的计算环境将彻底改变目前主流的基于 Desktop 的计算模式,它将使得计算不只局限于桌面,而是无处不在地渗透到人们的生活中.笔式用户界面是未来无处不在的计算环境的主要形式,而笔迹作为笔交互产生的主要信息形式已成为研究界和产业界努力开拓的方向.随着笔交互设备的成熟,使得人们越来越渴望以自然的笔迹信息进行交流.笔迹信息本身具有丰富的表达能力,将笔输入信息转化为格式化的信息,既不便于制作(识别工具易于出错),同时又丢失了大量的信息.然而,将笔迹保存为位图的传统方法需要大量的存储空间,尤其是在分布式网络环境及网络带宽不足的情况下,做到实时的笔迹交流是非常困难的.同时保存为位图也丢失了许多笔迹的结构信息,即使使用通常的数据压缩也并不能获得良好的效果.

目前,对于笔迹的基础支持还很不够,缺乏对于笔迹信息的有效存储和检索,即没有一个有效的、通用的、专门用于笔迹信息的表示格式.研究界和产业界都希望笔迹能同其他基本数据类型(如字符串)一样,能够被方便地操作和存取.即让笔迹成为计算机世界中的“一等公民”(first-class data type)[17].

要实现对于笔迹的有效操作,就必须获得笔迹的结构信息,而笔迹的结构通常是领域相关的,为了支持笔迹的结构和相关操作,同时保持工具系统的应用无关性,Penbuilder 提供基本的应用无关的 Ink 表示机制,领域相关的信息可以在此基础上由界面开发者扩充.基于这个基本表示,可以定义一些相应的操作,即 Ink 的“加、减、乘、除”等操作.

结构化的笔迹信息有许多优点,首先可以进行高效的操作,如文献[12].其次是存储和检索,对于减小笔迹的数据量(即压缩问题)是非常有帮助的.过去的研究多集中于笔迹的数据性压缩,而忽略了领域信息(笔迹固有的

宏观特性)和人的因素(即笔迹信息所反映的人的心理学特征).在 Penbuilder 的基于结构化的笔迹压缩中,将保持笔迹信息固有的逻辑结构,并将其调整映射到笔迹的可视结构中.根据具体应用的要求,依据笔迹结构,选择性地丢弃局部细节信息,依据人的心理学依据,保留大量的特征信息,而减少非特征信息.同时,通过分析笔迹信息的个性分析(作为一种新的信息载体,笔迹信息往往表现了人的个性特征),依据显著特征进行压缩,可做到有的放矢,获得低数据量、高品质的压缩笔迹.

8 目前状况和未来工作

目前,Penbuilder 已发展到其第 2 个版本,该工具箱具有 C++和 Java 两个版.Penbuilder 1.0 已在实验室内获得了广泛的使用,几乎所有的实验室笔交互原型系统均采用了该工具箱作为底层支持,如基于手势的字处理器、基于草图的概念设计和基于笔交互的 GIS 系统.基于 Penbuilder 2.0 开发的系统主要有 SketchPoint.Penbuilder 2.0 的一个重要特征是它在设计时考虑了如何同时支持多个笔交互设备以及各种交互环境之间的通信,使得它能够支持无处不在的计算环境中的界面开发.由于使用了域模型来构建系统,同时结合了基于框架的思想,如属性和值,使得工具系统使用者所需了解的信息大大减少,从而减少了使用者所需的学习负担.

可视化的编程为界面开发者提供了一种更为直观的开发方式,但由于笔式用户界面的复杂性,使得完全的可视化设计环境难以构造,在未来的研究中,我们将进一步完善 Penbuilder 的功能,并进行可视化设计的研究,力求为界面开发者提供易于使用的界面构造工具.

9 结 论

笔式用户界面提供给用户更为自然的交互方式,然而,笔式用户界面的构造是一项非常困难的工作.一个完整的笔式用户界面系统往往需要多领域、多学科的知识.笔式用户界面开发平台是构造笔式用户界面的有力工具,是提高界面构造效率的基础,也是笔式用户界面朝着标准化发展的保证.正如 GUI 刚成熟时缺乏好的界面构造工具一样,我们正处于类似的一个时代.Penbuilder 作为一个笔式用户界面的开发平台,总结了各种开发工具的优点、缺点,并结合了笔交互的特点设计的.它包含了构造一个笔式用户界面所需要的方方面面,尝试了各种新的技术,力求为界面开发者提供一个有力的支持.笔式用户界面开发平台的研究是一个非常艰巨的工作,它也是笔式用户界面最终成功的基础.

References:

[1] Myers BA. User interface software technology. ACM Computer Survey, 1996,28(1):189~191.

[2] Hong JI, Landay JA. SATIN: a toolkit for informal ink-based applications. In: Ackerman M, Edwards K, eds. ACM Symposium on User Interface Software and Technology (UIST 2000). New York: ACM Press, 2000. 63~72.

[3] Henry TR, Hudson SE, Newell GL. Integrating gesture and snapping into a user interface toolkit. In: Hudson SE, ed. ACM Symposium on User Interface Software and Technology (UIST'90). New York: ACM Press, 1990. 112~122.

[4] Landay JA, Myers BA. Extending an existing user interface toolkit to support gesture recognition. In: Ashlund S, *et al*, eds. ACM Conference on Human Factors in Computing Systems (INTERCHI'93). New York: ACM Press, 1993. 91~92.

[5] Li Y, Guan ZW, Chen YD, Dai GZ. Penbuilder: platform for the development of PUI (pen-based user interface). In: Tan T, Shi Y, Gao W, eds. Proceedings of the 3rd International Conference on Multimodal User Interfaces (ICMI 2000). Springer-Verlag, 2000. 534~541.

[6] Mankoff J, Hudson SE, Abowd GD. Providing integrated toolkit-level support for ambiguity in recognition-based interfaces. In: Turner T, Szwillus G, eds. ACM Conference on Human Factors in Computing Systems (CHI 2000). New York: ACM Press, 2000. 368~375.

[7] Myers BA, Myers BA, McDaniel R, Miller R, Ferrency A, Faulring A, Borison E, Kyle B, Mickish A, limovitski A, Doane P. The amulet environment: new models for effective user Interface software development. IEEE Transactions on Software Engineering, 1997,23(6):347~365.

[8] Li Y. Attribute grammar based user interface specification technology and supporting environment [MS. Thesis]. Xi'an: Northwest University , 1999 (in Chinese with English Abstract).

[9] Li Y, Guan ZW, Wang HA, Hua QY, Dai GZ. Design and implementation of a UIMS for component-based GUI development, In: Feng Y, *et al*, eds. Proceedings of the 16th IFIP World Conference of Computer: Software Theory and Practice (WCC 2000). Beijing: Electronic Industry Press, 2000. 955~955.

[10] Kramer A. Translucent patches: dissolving windows. In: Szekely P, ed. ACM Symposium on User Interface Software and Technology (UIST'94). New York: ACM Press, 1994. 121~130.

[11] Bederson BB, Hollan JD. Pad++: a zooming graphical interface for exploring alternate interface physics. In: Szekely P, ed. ACM Symposium on User Interface Software and Technology (UIST'94). New York: ACM Press, 1994. 17~26.

[12] Li Y, Guan ZW, Wang HA, Dai GZ, Ren XS. Structuralizing freeform notes by implicit sketch understanding, In: Davis R, Landay JA, Stahovich T, eds. AAAI Symposium on Sketch Understanding. Menlo Park, CA: AAAI Press, 2002. 91~98.

[13] Li Y. Research on pen-based user interfaces theory, technique and implementation [Ph.D. Thesis]. Beijing: Institute of Software, The Chinese Academy of Sciences, 2002 (in Chinese with English Abstract).

[14] Li Y, Guan ZW, Dai GZ. Modeling post-WIMP user interfaces based on hybrid automata. Journal of Software, 2001, 12(5):633~644 (in Chinese with English Abstract).

[15] Weiser M. Some computer science issues in ubiquitous computing. Communications of the ACM, 1993,36(7):75~84.

[16] Dong SH, Wang J, Dai GZ. Human-Computer Interaction and Multimodal User Interfaces. Beijing: Science Press, 1999 (in Chinese).

[17] Abowd GD, Mynatt ED. Charting past, present, and future research in ubiquitous computing. ACM Transactions on Computer-Human Interaction, 2000,7(1):29~58.

附中文参考文献:

[8] 栗阳.基于属性文法的用户界面描述技术及支持环境[硕士学位论文].西安:西北大学,1999.

[13] 栗阳.笔式用户界面研究:理论,方法和实现[博士学位论文].北京:中国科学院软件研究所,2002.

[14] 栗阳,关志伟,戴国忠.基于混合自动机的 Post-WIMP 界面的建模.软件学报,2001,12(5):633~644.

[16] 董士海,王坚,戴国忠.人机交互和多通道用户界面.北京:科学出版社,1999.