1997年10月 西北大学学报(自然科学版) Oct. 1997
第27卷第5期 Journal of Northwest University(Natural Science Edition) Vol. 27 No. 5

用户界面模型与形式规格说明研究*

华庆一

TP311.1

(西北大学计算机科学系,710069,西安;41岁,男,副教授)

摘 要 在讨论人机交互方式和界面控制与通信的基础上,描述用户界面抽象模型及形式规格说明方法和技术方面的研究和进展。主要强调图形用户界面的模型和描述问题。

关键词 用户界面模型;规格说明;交互控制与通信

分类号 TP311

一般地,应用程序中完成各种具体任务的软件部分称为程序的计算部分或应用语义,而处理用户与系统交互的软件称为用户界面。

早期的用户界面基本上采用了顺序命令行形式。而现在大都使用了WIMP(窗口、图标、菜单、指示器)形式的图形用户界面,极大提高了人机交互的方便性和有效性而被一般用户所接受。

人机交互能力的提高意味着设计和开发难度的增加,使得过去那种直接程序设计的方式不再能保证界面的质量和满足维护的要求。因此,各种工具已经被开发用来支持用户界面的设计和实现。这些工具通常使用某种用户界面的模型定义所开发界面软件的控制和通信结构,并允许开发者使用某种界面表示技术描述人机交互的形式、内容和顺序以及界面的布局等。界面的开发者有必要了解不同的界面模型和相应的描述技术,以便选用适当的工具来开发所需的界面。

1 交互的方式

当今的人机交互方式至少可用两种隐喻(metaphor)来描述:会话隐喻和模型隐喻[1]。会话隐喻将界面看成是用户和整个应用系统,而不是和特殊任务相关的应用语义对象之间的交互。特征是用户用某种语言输入命令,然后系统回答,如此反复。交互的逻辑是可预见的、线性的,故属于顺序对话类型,如命令行和菜单系统。这种对话具有一种明显的语言特征,故其容易使用一种类似于编译器的模型和规格说明来描述。

图形界面一般采用模型隐喻。允许用户使用直接操作(direct-manipulation)[2]的交互方式。与任务相关的应用对象在界面上表示为两维图形,这样的图形可被用户直接地操作,如选择、移动或删除。这样的交互方式给用户一种印象,好像他直接参与了应用语义的操作,而不是通过系统进行对话。直接操作一般具有三个特征:①对象可视化,用户感兴趣对象的连续表示;②语法极小化,采用物理动作或按钮代替复杂的语法;③语义反馈,在对象上快速、增量式和可逆的操作立即带来可视的效果。用户参与感极大地增强,交互从语义上更为接近特殊的应用语义。这种对话具有一种明显的面向对象的特征,界面的表示涉及到大量的语义操作,所以如何描述这类风格的界面仍然是用户界面研究中的一个开放性问题。

直接操作式界面的对话过程通常是多线式(multi-thread)的。这意味着同时可有多个对话路径供用户随机访问。每个表示应用对象的图形代表了一个有关它自己的特殊子对话。一旦用户选择了其中某个对象的表示,则他可以开始一个有关该对象的对话。他也可以暂时中断这个对话,转移到并进入另一

* 国家自然科学基金和中国科学院软件研究所开放实验室基金资助课题
收稿日期:1997-06-19

个对话(如通过移动鼠标),尔后又可返回至被中断的对话继续进行,即用户与系统的交互可在多个线索中切换。

但需注意到虽然可能有多个正在活动的交互设备(如键盘和鼠标器),但每个应用通常只有一个输入/输出流。对话实际上是以交替方式异步进行的,所以称异步对话。

2 对话的控制与通信

概念上,对话的控制和通信涉及到如何管理和实现用户和系统之间交互的逻辑顺序,以及系统的界面部分与语义部分如何相互传递信息。对话的控制机制和通信方式与对话的方式直接相关,特别是控制的顺序结构,不同的对话方式对对话的控制和通信的要求不同。

因为顺序对话的语言特征,使用户和对话设计者都可预计到任何对话的逻辑顺序,故其控制和通信的表示较简单。系统本身具有一个顶级控制循环让用户输入一个命令。在对话的过程中,控制逻辑由系统决定,系统根据当前输入部分的内容决定对话的下一部分。因此,这种控制方式是高度模式(mode)化的,其特征是顶级控制显式表示在应用程序中。这样的结构通常包含在命令串、菜单网或应答式会话的控制中。

直接操作式对话似乎是无模式的。通常许多对象同时对用户来说是可见的,用户可在对话的任意时刻将一个命令应用于任意对象(多线索),通常通过移动鼠标来选择(异步方式)。因此,用户输入的类型和顺序一般无法估计,在任意时刻系统几乎都处于顶级控制状态。故单线式控制对多线索对话是不适当的,这种对话需要异步控制。

在异步对话方式下,用户的输入通常被看作事件。事件具有名字和若干属性,分别表示交互的类型和特征。对话控制需根据其属性决定执行线索。因此,基于事件的机制是构造异步对话的基本控制和通信技术。

因用户处于全局控制,对话须按用户的动作激活或挂起,故与顺序对话控制机制不同,异步控制的顶级命令循环不能显式表示在应用程序的控制结构中。这样的顶级控制通常是由执行环境自动处理的。界面的开发者只需描述或实现有关个别对象的对话控制。多线式对话中的每个线索,即有关个别对象的内部对话一般仍采用了顺序控制,但是这些控制必须能够开始、中断和恢复有关自己的对话,并允许通过顶级控制来交换控制和信息。值得指出的是这种方式不一定反映了多线式对话的异步性质。

这样,异步或基于事件对话控制的基本结构可看成由两部分组成:其低级成分是由单个对话组成的并发(concurrent)例程集,而高级成分是一个对话控制器(或循环)。该控制器可在任意时刻根据事件激活或挂起了一个对话例程。一个并发例程描述了一组在对话中逻辑相关的操作,而对话控制器负责并发例程之间的通信和同步。这两个层次间通过事件通信。

Hartson[3]将界面与应用之间的通信划分为两种基本类型:宏通信和微通信。前者是指界面本身具有语义处理能力,能够对低级输入事件做出语义反馈,并将其翻译为相应命令或参数,而后者不具这种能力。直接操作界面因为需要应用语义能更为接近界面,所以采用微通信会极大地增加通信量,且应用需处理大量的图形操作。但把更多的语义带入界面部分将破坏"对话独立性"(界面和应用分离)的原则,使得一方的改变势必影响另一方。因此,较好的方法是提供好的界面模型和工具来支持图形对象操作,使得与应用的通信仅限于数据的计算。

3 用户界面的模型

界面开发工具可分为界面建造工具箱或用户界面管理系统(UIMS)。界面开发工具的一个主要特征是显式或隐式地利用了一个用户界面模型。界面模型不仅决定了所产生界面的控制和通信方式,而且也影响了工具本身的结构和对交互式软件设计和开发的支持程度。

最早的用户界面模型是 Seeheim 模型[4]。如图 1 所示,该模型将用户界面划分为 3 个部分:表示成

分涉及界面的外部表示，界面的其他成分不能与外部直接通信；对话控制部分指定用户和系统之间的对话结构；应用界面模型建立与应用语义之间的通信联系，描述界面可访问的数据结构和例程，并负责调用这些例程。在逻辑上这3个成分是相互独立的，它们之间通过发送单词来进行通信。

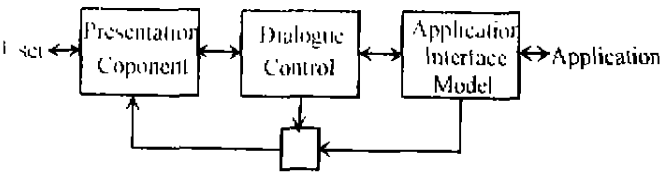

图1　Seeheim 模型的结构

Fig. 1　The architecture of Seehaim Model

Seeheim 模型本身是基于语言的模型。3个部分分别对应于词法、语法和语义3个层次。该模型的一个显著特征是强调对话控制部分的作用。然而，在直接操作对话中，用户是与个别应用语义对象的图形表示交互，而不是与整个应用系统对话。这意味着和个别对象相关的语法应当包含在各个图形表示对象之中，而不是作为一个统一的独立部分，即语法应极小化。另外，语义反馈对增加用户的参与感甚为重要。有时甚至认为是词法层次的操作也需要语义反馈，如拖动一个图形对象是一个词法操作，但倘若反馈该操作潜在的语义效果，则用户的参与感会极大地增加。这要求语义更加贴近于表示部分。显然 Seeheim 模型本身并不支持直接操作语法与语义的要求。

直接操作对话的特征建议使用面向对象的方法和技术。在面向对象范型(paradigm)中，应用系统被表示为一组对象，其属性和行为直接反映给用户，而不是通过系统。不同对象间的通信通过信息发送，所以对表示多线式对话是有效的。图2表示了一个面向对象的界面概念模型，称为 Multi-agent 模型。其中 Seeheim 模型的各个成分是在对象范围内定义的。每个对象，即 agent 包含自己状态与可视属性的表示，及输入操作定义。这样的 agent 可以是表示应用命令的菜单、按钮或应用对象的表示。然而，这个概念模型没有定义各个成分间的通信方式。

Multi-agent 的一个典型范例是 smalltalk-80 的 MVC 模型[5]。如图3所示，模型用于表示应用对象的状态属性，视图表示对象的可视属性，而控制器表示对象和用户的交互操作。交互过程开始于控制器，控制器要求模型改变其状态，然后模型要求视图和控制器改变它的显示和状态。视图可查询模型的状态和要求它的状态改变。

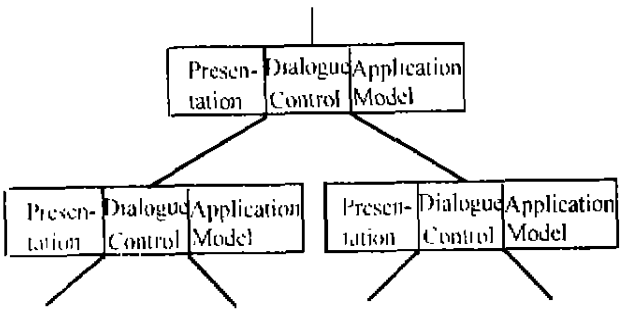

图2　Multi-agent 模型

Fig. 2　The Multi-agent Model

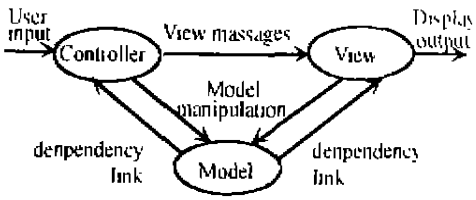

图3　Model-View-Controller 模型

Fig. 3　The Model-View-Controller Model

MVC 模型的一个特征是在对话独立性的前提下，允许语义和它的视图直接相互通信。另一个特征是将输入和交互处理与输出显示部分分离，以减少一方改变对另一方的影响。应当指出，MVC 模型仅是一个概念模型，如何分离和通信有赖于实现。

纯粹的分离对于直接操作界面是难于实现的。首先，控制器中通常包含显示功能以提供对输入操作的语义反馈能力。一方面，这使得控制器必须为不同的视图定做，影响代码可重用性。另一方面，一般较难确定哪些显示功能属于控制部分，哪些属于视图部分。再者，控制器和视图部分都需包含某些语义计算功能。控制器应对输入事件做出解释而不是发送低级事件，否则一旦词法或语法有所改变，模型部分必须做相应改变。同理，视图须含有图形操作能力。但这些语义计算功能的界限是难以判定的。

对 MVC 模型的一个改进是采用所谓的交互式对象取代控制器(如 Mayers 的 Ganet[6])。交互式对象封装了各种交互操作行为，并通过限制(constraint)机制自动维护交互与语义反馈间的动态不变关系。交互式对象捕捉了几类交互操作的共同特征(如鼠标移动、选择等)且是高度参数化的，所以可应用

于不同的视图。另一个改进是使用控制器、视图和应用模型的一个共享数据结构组织交互对象和图形对象。

面向对象范型的一个缺点是难以表示对话过程中的时序关系，特别是语法层次上的对话控制逻辑。再者，它使用原型或其他非形式的规格说明技术指定和实现用户界面，导致其结果难以预测，且界面不易维护和修改。因此，它通常用于提供对交互式图形对象的支持，而对对话控制逻辑需要形式描述技术的支持。

4 用户界面的描述方法与技术

用户界面变得愈复杂，形式规格说明的作用就愈加重要。它们形成了界面软件自动生成的基础。UIMS 的优点是通过某种形式描述技术来支持用户界面对话控制结构的自动生成。

早期基于 seeheim 模型的 UIMS 一般采用了形式语言记号作为描述工具，如状态转移网(TN)和上下文无关文法(CFG)。它们的描述能力较弱，相当于某种类型的状态或下推自动机。TN 的优点是直观，直接反映了控制的时序关系。而 CFG 不具有这样的优点，但 CFG 可从词法和语法的角度精确地反映对话的层次，这对于对话的设计是重要的。这些形式工具的主要问题是不支持多线式对话所要求的异步控制的描述，因为不允许根据语义信息(事件属性)与顶级控制交换控制和通信。另外，它们也不能对直接操作所要求的语义反馈提供有效的支持，因为它们本质上是文法工具。

```
Event-handler-line:
var state, first, second;
event Button do {
    if state = 0 then
        first := current position;
    else  second  = current position:
        deactive(self): }
event Move do {
    if state = 1 then
        draw a line from first
        to current position: }
init
    state := 0;
```

图 4 橡筋线的事件处理器

Fig. 4 Event-handler for Rubber Band

为了满足异步控制的需要，通常的方法是在传统的顺序描述工具中引入基于事件的机制。其基本技术是用这些工具描述个别对话，并使得每个对话的控制可由语义信息来制导：根据输入事件开始执行自己的对话序列，或保存状态并返回控制至执行环境，或从返回点恢复状态继续对话序列。虽然这可能未能反映多线式对话的实质，但直至今日仍是一种行之有效的技术。

一些 UIMS 使用了事件语言或类似的技术描述多线式对话。在事件语言中，每个对话由一个事件处理器描述。事件处理器主要由它处理的事件和处理这些事件的过程组成，如一个橡筋线做图的事件处理器可表示为图 4[8]。事件语言是基于 seeheim 模型的，所以难以支持语义反馈。

Jacob 将 TN 与事件机制结合，以描述多线式对话[9]。如图 5 所示，界面的控制结构被划分为两个 TN 子网。包含在 UIMS 运行时支持环境中的顶级控制是一个简单的分支结构，而每个分支用于表示个别对话。这样，顶级分支可根据不同事件激活或挂起一个对话。为了避免复杂的子网结构和提高描述的可重用性，Jacob 利用面向对象的技术允许一个子网可继承其他子网的结构。TN(以及 CFG)的一个缺点是无法表示控制过程中的语义信息和所需的计算，必须使用其他非形式工具，有时会造成理解上的困难，且结果难以预计。

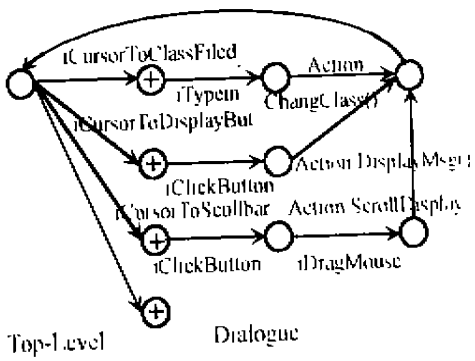

图 5 扩充 TN 规格说明

Fig. 5 The Extended TN Specificatiom

作者使用了一种基于属性文法(AG)，并扩充了事件机制的规格说明方法[10]。该方法分别使用 AG 的语法和语义记号指定每个对话内部的控制序列、对话间的通信，以及语义表示和计算。为了表示多线式对话，该方法对 AG 的规则扩充了一个布尔前提条件(称为 guard)。该 guard 利用规则的(语义)属性值决定对话是否进行或挂起，因而控制可由语义制导。因为所用的 AG 族是可以从顶向下、从左到右单遍扫描计算的，故控制的实现可利用一个简单、高效、扩充了

guard 机制的 LL(1)分析器。与上述方法相比，其优点是分别用语法和语义规则表示对话控制和语义计算。图 6 说明了一个这样的例子。

```
Grammar-line:
  parameter:   Position,                  // coordinates of drawing region
               Presentation;              // display object in the presentation component
  attr-var:    first, second,             // line start- and end-point
               position;                  // current mouse point
  token:       BUTTON,                    // mouse button clicked
               MOVE;                      // mouse moved
  rule:
    guard      BUTTON[position] is in Position                                   (1)
    syntax     <line> ::= BUTTON <end-point>
    semantic   <end-point>    first := BUTTON[position];
                   output(Presentation, LINE, BUTTON[position], <end-point>    second).
    guard      MOVE[position] is in Position                                     (2)
    syntax     <end-point>^0 ::= MOVE <end-point>^1.
    semantic   <end-point>^1    first := <end-point>^0    first;
                   apply draw-a-line, <end-point>^0    first, MOVE[position]);
               <end-point>^0    second := <end-point>^1    second;
    guard      BUTTON[position] is in Position                                   (3)
    syntax     <end-point> ::= BUTTON
    semantic   <end-point>^1    second := BUTTON[position];
```

图 6　橡筋线的 AG 规格说明

Fig. 6　The AG Specification for Rubber Band

上述的种种描述方法均采用了可执行的规格说明技术，其抽象层次较低。这要求开发者显式地定义控制顺序、提供词法和语法细节，显然是十分复杂和费时的。

某些 UIMS（如 MIKE[11]，UofA*[12]，HUMANOID[13]，UIDE[14]）采用了面向任务的方法。这样的方法从应用的任务描述出发，从应用提供的语义命令中产生一个界面控制的高级描述。一个应用命令通常包括命令名和所需的参数。系统从这样的高级描述中自动生成和实现界面的词法和语法层次。例如 MIKE 从一种类 Pascal 的命令描述中产生界面控制的 CFG 描述，UofA* 采用了类似的方法，但它生成基于事件语言的事件处理器，UIDE 还使用了布尔前提条件来综合出高级控制顺序。

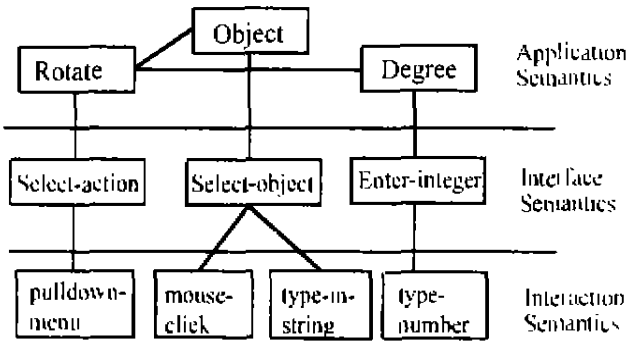

图 7　UIDE 应用模型的规格说明层次

Fig. 7　The Layered Specification Model of UIDE

今天的一种趋势是将面向对象的方法和规格说明技术相结合，利用面向对象的方法在低级层次上提供对交互技术、图形显示技术和语义反馈的支持，而使用 UIMS 的方法在高级层次上提供对控制顺序和应用模型的描述。将面向对象的内容尽量控制在系统内部，而在外部允许开发者使用规格说明的方法进行描述，支持界面的自动或半自动生成。例如第二代 UIDE[15]使用面向对象的技术在 3 个不同层次上定义交互式对象（类似于 Garnet 的 Interactor）、界面对象（应用对象的图形表示）和应用对象（应用数据表示）。与 MIKE，UofA*，以及 HUMANOID 等面向任务的 UIMS 类似，UIDE 从应用命令（任务）的高级描述中自动综合对话控制顺序。它提供了一个由应用语义、界面语义和交互语义 3 个层次组成的应用模型，允许开发者用从顶向下的方式定义应用动作（应用命令及参数）、选择预定义界面动作（交互操作命令）和交互式技术。例如图 7 表示了一个 Rotate 命令（将选中的 Object 旋转 Degree 度）的应用模型的描述层次。

5 结束语

综上所述,人机交互方式的特征直接影响了用户界面的表示技术与控制机制,要求有相应的界面模型、规格说明方法与技术,要求有好的工具帮助开发有效、可用的用户界面。虽然高度交互的直接操作方式界面被广泛地采用,但相应的界面模型、规格说明方法与技术仍是用户界面研究的一个开放性问题。

参 考 文 献

1 Hutchins E L, Hollan J D, Nonrman D A. Direct manipulation interfaces. In: Norman D A. ed. User Created System Design. Hillsdale, N J: Lawrence Erlbaum Assoc. 1986,87～124

2 Schneiderman B. Direct manipulation: a step beyond programming languages. IEEE Computer, 1983(10):57～69

3 Hartson R H. User-interface management control and communication. IEEE Software, 1989(1):62～70

4 Green M. Report on dialogue specification tools. In: Pfaff G. ed. User Interface Management Systems. Berlin: Springer-verlag, 1985, 9～20

5 Krasnar G E, Pops S A. Cookbook for using the model-view-controller user interface paradigm in smalltalk-80. J. of Object-Oriented Programming, 1989, 1(3):26～49

6 Mayers B A, Giuse D, Dannenberg R, et al. Garnet: comprehensive support for graphical, highly-interactive user interfaces. IEEE Computer 1990,23(11):71～85

7 Green M A. Survey of three dialog models. ACM Trans. on Graphics, 1986, 5(3):244～275

8 Jacob R J K. A Specification language for direct-manipulation user interfaces. ACM Trans. on Graphics, 1986, 5(4):283～317

9 华庆一. 一个基于属性文法的用户界面规格说明. 计算机学报,1996,5(19):351～357

10 Olsen Jr D R. MIKE: the menu interaction Kontrol environment. ACM Trans. on Graphics. 1986,5(4):318～344

11 Singh G, Green M. Automating the lexical and syntactic design of graphical user interfaces: the UofA* UIMS. ACM Trans. on Graphics, 1991,10(3):213～254

12 Luo P, Szekely P, Neches R. Management of interface design in HUMANOID. In: the Proceedings of INTERCHI'93. New York: Assoc for Computing Machinery, 1993,107～114

13 Foley J, Gibbs C, Kim W, et al. Defining interfaces at a high level of abstraction. IEEE Software, 1987(6):25～32

14 Sukaviriya P N. A Second Generation User Interface Design Environment: The Model and The Runtime Architecture. In: the Proceedings of INTERCHI'93. New York: Assoc for Computing Machinery. 1993, 375～382

责任编辑 曹大刚

Research on User Interface Models and Specifications

Hua Qingyi

(Department of Computer Science, Northwest University, 710069, Xi'an)

Abstract Research and progress on user interface models and formal specification techniques are described, based on the discussion of interactive models, control and communication within user interfaces. The main emphasis refers to those issues on the model and description of graphical user interfaces.

Key words user interface model; specification; interactive control and communication