1000-9825/2001/12(05)0633-12 ©2001 Journal of Software 软件学报 Vol. 12, No. 5

Modeling Post-WIMP User Interfaces Based on Hybrid Automata*

LI Yang, GUAN Zhi-wei, DAI Guo-zhong

(*Intelligent Engineering Laboratory, Institute of Software, The Chinese Academy of Sciences, Beijing* 100080, *China*)

E mail: ly@imd.cims.edu.cn

http://www.ios.ac.cn

Received June 13, 2000; accepted December 25, 2000

Abstract: Post-WIMP (i.e. Windows, Icons, Menus and Pointer) interface as the next generation user interface has substantial differences from WIMP interface, which has been the dominant paradigm of human-computer interaction for decades. Post-WIMP interface provides more natural and efficient interaction by utilizing the advanced interactive technology, such as virtual reality, voice technology, gesture interaction and so on. However, it is proved to be more difficult to be constructed. In order to build post-WIMP interface effectively, it is a better way to specify the interaction at an abstract level without concerning the details of the implementation before construction. In this paper, the fundamental attributes of post-WIMP interface are discussed. The most distinguished property of post WIMP interface is the hybrid interaction, which means the continuous interaction blending with discrete interaction. The post-WIMP interface is analyzed from the point of view of hybrid system, which can give more accurate and strict analysis for post WIMP interface. Post WIMP interface can be modeled as a set of cooperating hybrid automata which are special for hybrid system. Hybrid automata are used to model post-WIMP interface. A semi-formal specification language LEAFF is designed based on hybrid automaton for specifying post-WIMP interface by the combination of the text-based specification and graph-based specification, and it can clarify the control and temporal relationships in post-WIMP interface. Two typical instances, object manipulation in virtual reality and pen interaction, are specified by LEAFF. Some issues and the future work are also discussed, such as the specification of parallel issues in post-WIMP interaction, verification of interaction and specification based interface construction. The modeling technology presented in this paper ought to be useful to the research and development of post-WIMP interface.

Key words: HCI (human-computer interaction), post-WIMP interface, hybrid automaton

With the development of the interactive devices, the improvement of the hardware's performance and the evolution of the computing model, the traditional WIMP interface (i.e. Graphical User Interface, GUI), which is a

* Supported by the National Natural Science Foundation of China under Grant No. 60033020 (国家自然科学基金); the Key Project of China 863 Advanced Technology Plan under Grant No. 863-306-03-01 (国家863高技术研究发展计划)

LI Yang was born in 1974. He is a Ph.D. candidate at the Institute of Software, the Chinese Academy of Sciences. He received his M.S. degree in computer science from Northwest University in 1999. His research interests are human-computer interaction and software engineering. **GUAN Zhi-wei** was born 1974. She got her Ph.D. degree at the Institute of Software, the Chinese Academy of Sciences in 2000. Her research interests are human-computer interaction and recognition sciences. **DAI Guo-zhong** was born in 1944. He is a professor and doctoral supervisor of the Institute of Software, the Chinese Academy of Sciences. His current research areas include human-computer interaction, real-time system, computer graphics and CAD.

dominant interactive paradigm presently, is incapable of making a good use of them. The computing model shifts from centralized model to distributed model and the interactions in mobile computing environments have become more and more popular. More powerful and natural interactive devices carrying magnificent information have emerged, such as data gloves, data headpieces[1]. As a result, there will be two major directions for the evolution of user interface relative to desktop-based user interface: one is evolving towards "*larger*" interactive environment, such as virtual reality, and the other is towards "*smaller*" interactive environment, such as mobile computing. WIMP interface is a desktop-oriented interactive paradigm primarily based on mouse and keyboard which are obviously cumbersome for these future user interfaces. WIMP interface can not meet the requirement of real-time information processing for these two interactive environments and it is incompetent for achieving natural and effective interaction. In order to solve all above matters, it is necessary to introduce a new interactive paradigm.

The term post-WIMP interface refers to the user interfaces coming in the future, after the current generation of WIMP interfaces. The term doesn't give an unambiguous and strict definition for future user interface but just a catchall for the next generation user interface. Post-WIMP interfaces provide non-command, parallel, multimodal, real-time, continuous and discrete interaction, which have a higher degree of interactivity than any previous one. The design of the interactive behavior in post-WIMP interface is based on the users' mental model so that it can provide a natural, effective way for users to communicate with computer without much cognitive effort. So it is also a sort of intention-based interaction. It releases the users from the frustrating mode switching and gives a broader communication bandwidth between human and computer. There are several typical user interfaces which belong to post-WIMP interface, such as virtual reality, new types of games, intelligent agent interfaces, interactive entertainment media, pen-based interfaces, eye movement-based interfaces, speech interface, and ubiquitous computing[2].

Post-WIMP interface is more complicated than any previous user interface. Consequently, it is hard to be constructed. It has been turned out to be an effective way to specify user interface at an abstract level without concerning the details of the implementation before programming[3]. It is especially useful and crucial to clarify the relationship of the controls in post-WIMP interface. In the following sections of this paper, the essence of post-WIMP interface will be argued and a detailed analysis for post-WIMP interface will be presented from the point of view of hybrid system[4]. We model post-WIMP interface as a set of hybrid automata. A semi-formal specification language LEAFF based on hybrid automaton is also presented. Two examples of specification for post-WIMP interface are given and some discussions are preformed.

1 Applying Formal Techniques to User Interface

Traditionally, the envisaged role of a formal method is to describe the essence of a system function without premature commitment to implementation consideration[5]. However, our motivation for specifying user interface formally is to systematize and normalize the knowledge and idea about human-computer interaction and instruct the construction of the interactive system. The process of development is specification-centered but not specification-driven. Using formal method for user interface specification brings some advantages: it brings together contributions originating from different perspectives and allows the developer to check their consistency and, possibly, to identify issues which require further consideration[6]. It is helpful to designers for anticipating diversified possibilities of interaction without implementation.

A certain interaction can be equipped with various appearances. The specification should aim at the user-visible behavior of the interface but not the implementation of the input and output. User-visible behavior of the interface is fulfilled by a dialog process and supervised by dialog control which is the most important and sophisticated part of user interface. Dialog control is responsible for maintaining the states of the dialog process, coordinating the I/O of user interface and invoking the semantic action. This part is usually embodied as the interactive styles.

Traditional specification techniques for user interfaces are compiler-based, such as BNF (Backus-Naur Form), ATN (Augmented Transition Network) and AG (Attribute Grammar)[3,7]. These formal or semi-formal specification techniques work well for command-line-based user interface or GUI which are discrete-token-based. However, there are interactive behaviors which are inherent to be continuous from the user's point of view in post-WIMP user interface, though any user interface is ultimately executed in a discrete way in the computer[2]. UML (i.e. Unified Modeling Language) as a popular modeling technique models the behavior of objects with state transition diagrams. It is not suitable for modeling post-WIMP interface because its modeling is based on the analyses of the system instead of behaviors of user interaction, and it is designed for the universal systems but not special for the user interface. Consequently, the characteristics of the user interface are unable to be highlighted, and UML also seems corpulent to the design of user interface in its early period. A well-designed specification method should specify the user interface from the user's point of view but not the system and it is also supposed to be used in portable way. Summarily, the traditional specification techniques are no longer compliant for post-WIMP user interface.

2 Essence of Post-WIMP User Interface

The purpose we strive for with today's user interface is to minimize the mechanics of manipulation and the cognitive distance between intent and the execution of that intent. It's obvious that the user wants to focus on the task, not the technology for specifying the task[8]. The interaction in post-WIMP interface is based on the user's existing skills for interacting with the real world. We should identify the structure of post-WIMP interface as the user sees it. Post-WIMP interface has the following characteristics.

• Hybrid interaction (Continuous alteration blending with discrete events)

To post-WIMP interface, trying to describe the whole interface in purely continuous terms or purely discrete terms would be entirely possible, but inappropriate[2]. From the user's point of view, in post-WIMP interface, there are continuous behaviors, such as movement of the hands, and discrete behaviors, such as pressing buttons. It has essential difference from WIMP interface which treats all interactive information as discrete token. But in post-WIMP, it is difficult to tokenize all interactive information. This attribute is the most distinguished difference from any previous user interface.

• Parallel dialog threads

Generally, there are multiple dialog threads existing simultaneously in post-WIMP interface, which communicate with each other and they are executed in parallel and asynchronous way. These dialog threads coordinate to accomplish a task. Dialog threads can be suspended and resumed. This structure makes good use of the resource of the system[7].

• Multimodal interaction

User's intention of interaction is decomposed and allocated into different physiological modalities, such as hands, mouth and eyes, and transferred to interactive system by the artificial modalities of computer, such as pen and microphone. These modalities work simultaneously in a cooperant way and the information from different modalities is integrated for execution by modal integration. The interactive feedback is also achieved by the information fission to several output modalities, such as visual modality, audio modality and tactile modality. As a result, the user interface has multiple I/O streams. Multimodal interaction greatly improves the efficiency of the interaction.

• Real-time requirements

The processes of input and output in post-WIMP interface are based on deadline-based computations[2]. The integration of interactive modalities is time-related. Time is a significant factor in the extraction and execution of the interactive intention and most of relationships in post-WIMP interface are temporal.

• Inaccurate interaction

WIMP user interfaces employ well-defined digital inputs, but post-WIMP ones often have fuzzier ranges. Probabilistic inputs take great percentage of the interactions in post-WIMP interface. Unlike traditional mice and keyboards, the input/output devices used in these interfaces, such as speech or gesture recognizers, may produce a probability vector rather than a single token for a user input[9].

Post-WIMP interactions convey a sense of hybrid interaction to the user. Here we analyze a pen-based interaction based on tablet. When we submit a multi-stroke pen gesture, feedback of pen's ink track is related to the position of the pen's cursor. This relationship is operative when pen gives a pressure to the tablet and ceased when the pressures disappear or the inter-stroke's interval delay is out. So this continuous relationship is temporal. But there is also a relationship defined between the pen's position and cursor's position on the screen that is permanent because the cursor makes an immediate movement whenever pen is moved. Once the gesture is submitted, it will be recognized and a discrete token will be generated, which will trigger the transition from one interactive mode to another.

By above analysis, the post-WIMP interface can be abstracted as a set of continuous relationships, some of which are permanent and some of which are engaged and disengaged from time to time. These relationships accept continuous input from the user and typically produce continuous responses or inputs to the system. The actions that engage or disengage them are typically discrete[2]. Based on this analysis, a model for combining constraint-like continuous relationships and token-based event handlers is proposed for the specification of post-WIMP interface[2]. This method is innovatory to the research of post-WIMP interface. Our research is based on this analysis and a further research is done by using the theory of hybrid automaton for reference. The hybrid property of Post-WIMP interface is similar to a hybrid system from the point of view of formal techniques. A run of a hybrid system is assumed as a sequence of steps. Within each step the system state evolves continuously according to a dynamical law until a transition occurs[4]. By employing the framework of hybrid automaton, we intend to inherit some formal techniques, such as specification and verification, from the theory of hybrid automaton in order to analyze and investigate post-WIMP interfaces in the formal way. In Section 3, analysis based on hybrid automaton will be performed and modeling post-WIMP interface based on hybrid automata is described.

3 Analysis for Post-WIMP User Interface

3.1 Post-WIMP user interface and hybrid system

The structure of post-WIMP interface accords with multi-agent model where each agent acts for a dialog thread. Traditional user interface has single input/output stream but post-WIMP interface has multiple modalities for interaction as mentioned in Section 2. The result of the modal integration will be submitted to the related agents according to the user's intention of interaction and the outcome of the agents' processing will be dispatched to multiple modalities. So post-WIMP interface has the many-to-many structure, which means that multiple modalities serve for multiple agents.

In WIMP interface, the execution of a dialog thread can be simulated by PDA (Push Down Automaton) which is special for CFG (Context Free Grammar)[10]. It is unnatural to model post-WIMP interface because PDA can only accept and process token-based stream and it's difficult to reflect the continuous interaction aspect of post-WIMP interface. A suitable model for post-WIMP is required.

As the result analyzed previously, post-WIMP interface can be abstracted as hybrid system, which can be modeled as a set of finite automata where each is equipped with a set of variables, known as hybrid automaton.

The control locations of the automaton are labeled with evolution laws. At a location the values of the variables change continuously with time according to the associated law. The transitions of the automaton are labeled with guarded sets of assignments. A transition is enabled when the associated guard is true, and its execution modifies the values of the variables according to the assignments. Each location is also labeled with an invariant condition that must hold when the control resides at the location[4].

We give a simple exposition for hybrid automaton here. A hybrid automaton is specified as a vector with six elements[4]. $H=(\mathrm{Loc},\mathrm{Var},\mathrm{Lab},\mathrm{Edg},\mathrm{Act},\mathrm{Inv})$.

• A finite set Loc of vertices called locations.

• A finite set Var of real-valued variables. A valuation v for the variables is a function that assigns a real-time $v(x)\in R$ to each variable $x\in \mathrm{Var}$. We write V for the set of valuations.

• A finite set Lab of synchronization labels that contains the stutter label $\tau\in \mathrm{Lab}$.

• A finite set Edg of edges called transitions. Each transition $e=(l,a,\mu,l')$ consists of a source location $l\in$ Loc, a target location $l'\in$ Lcc, a synchronization label $a\subset$ Lab, and a transition relation $\mu\subseteq V^2$. The transition is enabled in a state (l,v) if for some valuation $v'\in V$, $(v,v')\in\mu$. The state (l',v'), then, is a transition successor of the state (l,v).

• A labeling function Act that assigns to each location $l\in$ Loc a set of activities. Each activity is a function from the nonnegative reals $R\geqslant 0$ to V.

• A labeling function Inv that assigns to each location $l\in Loc$ an invariant $Inv(l)\subseteq V$.

An example of hybrid automaton is shown in Fig. 1. The skeleton of a hybrid automaton is characterized by a finite automaton which is composed by a set of locations (e. g. l_0) and a set of transitions labeled by a guard (e. g. $y=w_1$), which means discrete events. The evolution laws, in the form of functions (e. g. f and g), establish the relationship between the variables with continuous changes. The real-time property can be easily specified in hybrid automaton because any variable can be time-based, which is usually presented by derivative of time.

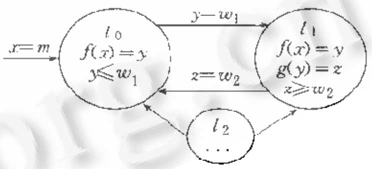


Fig. 1 An example of hybrid automation

The above formal specification gives strict definition for hybrid automaton. A post-WIMP interface is often too complex to be directly modeled by hybrid automaton. Some extensions for hybrid automaton have to be made. Furthermore, we are not going to use purely formal method to model post-WIMP user interface. It should be more practical to partially adopt formal techniques into specification than to provide a pure formal system, and as a matter of fact, it is quite difficult to model a complicated system by pure formal system entirely. It is evident that the specification language presented in Section 4 is a combination of formal techniques and object-oriented ideas. In the procedure of the modeling, what we concern about are the continuous aspects of the interactive behaviors but not the continuous aspects of the internal representation. There are often various kinds of variables used in post-WIMP interface, such as "Array", "Boolean" and "String". It would be better not to confine the types of the variables. A post-WIMP interface can be modeled as a set of hybrid automata which cooperate with each other by the external variables in the locations and labels on the transitions. The situation of post-WIMP interface can be quantified as a state vector. The continuous interaction behaviors can be specified in the locations and discrete behaviors can be specified by the transition between the locations.

3.2 Temporal model

In this subsection, post-WIMP interface in analyzed from temporal point of view. The temporal model of post-WIMP interface is given in Fig. 2. There are three variables in Fig. 2, related to each other by the evolution laws and related to the factors of time indirectly. The changes of all variables in the same location (or step) are executed simultaneously. At each point for transition, the curve is not continuous. An abrupt change occurs when the curve enters another step. The solid circles represent discrete events. In each step, the change of the variable is continuous. A location of hybrid automaton can be mapped into several steps here, which means that there will be a set of similar curves emerging in different steps. The hybrid automaton reflects the characteristic of computation in interactive model of post-WIMP. The interactive process of post-WIMP interface can be simulated by hybrid automaton accurately. In this figure, the state of the whole post-WIMP interface can be quantified as the vector (variable$_1$,variable$_2$,variable$_3$). In an actual post-WIMP interface, the elements of a vector are used to specify the information of the input/output modalities, control and semantics. The continuous changes in a location accord to the continuous interactive behaviors. Once the continuous interactive behaviors are executed to some extent, when some conditions are met, a discrete event will be triggered and it will cause the transition between locations, and accordingly, the movement from one step to another step will happen.

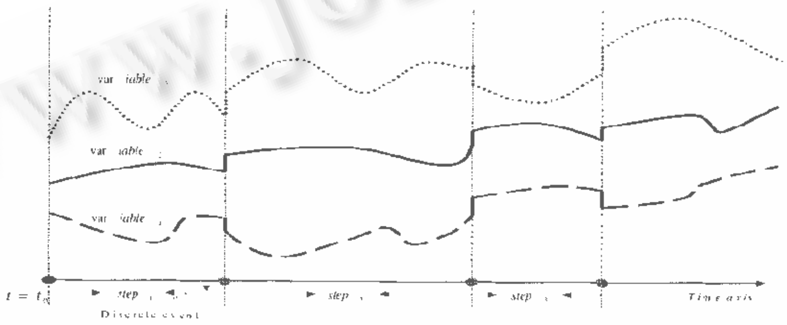


Fig. 2 Temporal model of post-WIMP interface

4 Specification Language LEAFF

A specification language LEAFF (LEAFF is an abbreviation of variable, Event, Action, Flow, Function) for post-WIMP interface is introduced here based on the analysis of previous sections. A specification template is shown in Fig. 3. Each agent in a post-WIMP interface can be specified by an **"Interactor"** module named as **"interactor—name"**. Actually, it is a text-style hybrid automaton. A module is composed of the definitions of the variables, events, actions and flows. The state of an interactive agent can be specified by a set of variables. There are four kinds of variables: **"InputVariable"** serves for input modalities which is directly related to input devices, **"FeedbackVariable"** stands for output modalities which controls the interactive feedback, **"ControlVariable"** is employed for interactive control and **"SemanticVariable"** is used to characterize the application semantics. Each variable has its own data type, such as "Integer", "String" and so on. Specification for events is accomplished in the section **"Event"** in the form of "event_name {premise→conclusion}". **"Flow"** section is used for the specification of the continuous interactions. Each flow is identified as a conjunction of a set of continuous functions of a location. The specifications for these functions are completed in the **"Function"** section. **"Action"** section is used for the specifications of semantic actions. The rules for specifying functions and actions are not strictly defined here and the designers can even specify them by natural language. In the text based specification, events, flows, actions and functions are explicitly defined.

```
Interactor interactor_name
    {
        InputVariable variable11[datatype],variable12[datatype]...
        Feedbackvariable variable21[datatype],variable22[datatype]...
        ControlVariable variable31[datatype],variable32[datatype]...
        SemanticVariable variable41[datatype],variable42[datatype]...
        Event
        {
            event1{...}
            event2{...}
        }
        Action
        {    ...    }
        Flow
        {
            flow1{...}
            flow2{...}
        }
        Function
        {    ...    }
    }
```

Fig. 3 Specification template

The communication between interactors is very important. In the specification, one way of the communication is by accessing the variables defined in other interactor scope, which is in the form of " interactorName$_i$. variable-Name$_j$". In order to make the specification more intelligible, LEAFF employs graph-based specification for presenting specification in a straightforward style and it will be shown in Section 5.

5 Examples of Specification

5.1 Grabbing and manipulating in VR (virtual reality)

Grabbing and manipulating are the most general behavior in VR. The users usually manipulate the visual 3D (three-dimension) objects in the VR environment by 3D-mouse or Data gloves[11]. There used to be a cursor representing user's hands, which gives the user timely optical feedback of the movement of hands. Grabbing and manipulating are often started from "grabbing" and ended by "releasing", and in the procedure, the user can change the orientation of the object freely. The example presented here is based on 3D-mouse with which an interactive device generates 6-freedom vector and button information. We employ events "button-down" and "button-up" for the action "grabbing" and "releasing" respectively. The process of manipulating is continuous. As shown in Fig. 4, a cube is grabbed and dragged for a distance and then dropped. The two crossed circles represent the cursor for use's hand. The user can perform six-freedom movement for her/his hand and the circles will rotate and shift according to the user's movements. When the circles embrace an object, the user can push down a button to grab it and then manipulate it. The text-based specification is presented in Fig. 5. The state of the interface is characterized as a vector (3D_mouse_ori, 3D_mouse_button, hand_cursor_ori, object_ori). "3D_mouse_ori" is for the state of the 3D mouse with data type "Movement" denoted as 6-element vector $(\Delta x, \Delta y, \Delta z, \Delta\alpha, \Delta\beta, \Delta\gamma)$ for reflecting the changes of 6 freedoms in 3D space. "hand_cursor_ori" and "object_ori" act for cursor's orientation and object' orientation respectively with date type "Orientation" denoted as $(x, y, z, \alpha, \beta, \gamma)$. "3D_mouse_button" is for button information from 3D-mouse with "ButtonEnum" type (ButtonDown, ButtonUp).

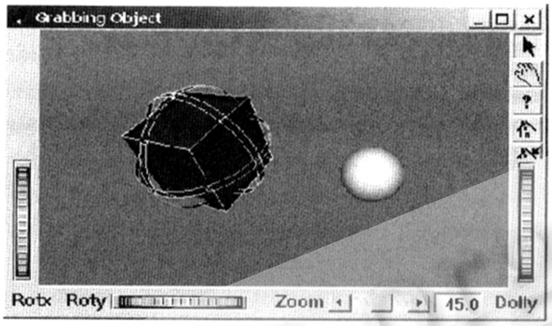


Fig. 4 3D-Mouse-Based grabbing and manipulating

```
Interactor 3D_object
   {
         InputVariable 3D_mouse_ori[Movement], 3D_mouse_button[ButtonEnum]
         FeedbackVariable hand_cursor_ori[Orientation]
         SemanticVariable object_ori[Orientation]
         Event
         {
              grabbing{3D_mouse-button=ButtonUp→3D_mouse-button=ButtonDown}
              releasing{3D_mouse_button=ButtonDown→3D_mouse_button=ButtonUp}
         }
         Flow
         {
              idle{l1: 3D_mouse_button=ButtonUp→f}
              manipulating{l2: 3D_mouse_button=ButtonDown→f∧g}
         }
         Function
         {
              f: modify the cursor's orientation according to the movement of user's hand and the
                 present cursor's orientation
              g: modify the objects' orientation according to cursor's orientation
         }
   }
```

Fig. 5 Specification of 3D manipulation

The graph-based specification is shown in Fig. 6.

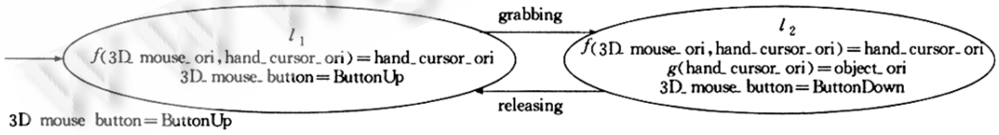
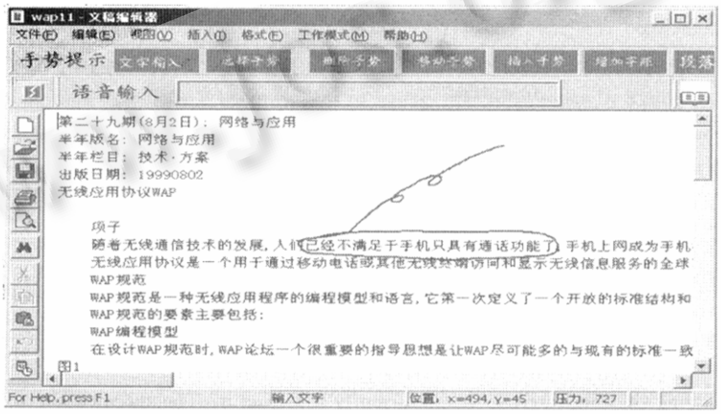

grabbing
l_1
f(3D_mouse_ori, hand_cursor_ori) = hand_cursor_ori
3D_mouse_button = ButtonUp
l_2
f(3D_mouse_ori, hand_cursor_ori) = hand_cursor_ori
g(hand_cursor_ori) = object_ori
3D_mouse_button = ButtonDown
releasing
3D mouse button = ButtonUp

Fig. 6 Graph specification for manipulation

5.2 Gesture interaction in PUI (pen-based user interface)

PUI is applied in mobile computing environment. Pen-gesture is an efficient way for expressing interactive intention in PUI, which provides information of pressure, orientation and button[12]. Alterations of pressure and orientation are continuous from the user's point of view, but button is discrete that the user only cares about whether button is down or up. Here we only use pressure and position information. A tool for draft editing (named

EasyEditor) has been implemented based on tablet-based pen (we use WACOM Intuos pen) which is magnetic inductive and equipped with double buttons[12,13]. EasyEditor employs pen-gesture as the major interactive way. User can use pen-gesture to fulfill deleting, inserting, selecting and so on. The pen is activated as soon as it enters the proximity of the tablet. When user moves the pen without touching the pen tip to tablet, the cursor is positioned according to the pen, and if the user moves the pen with pressure to the tablet, a track of red ink will occur. A stroke is produced in a continuous process which starts with pen tip touching to the tablet and ends with pen tip leaving the tablet. In this example, a gesture may consist of multiple strokes and intervals among these strokes which are limited to 0.5 seconds, which means that once the time is out the gesture will be recognized and some actions will be executed. The gesture in Fig. 7 means deleting the circled words and it consists of multiple strokes. The interaction of this example is time-based and reflects the real-time aspect of post-WIMP interface.



第二十九期(8月2日): 网络与应用
半年版名: 网络与应用
半年栏目: 技术·方案
出版日期: 19990802
无线应用协议WAP

项子
随着无线通信技术的发展，人们已经不满足于手机只具有通话功能了，手机上网成为手机
无线应用协议是一个用于通过移动电话或其他无线终端访问和显示无线信息服务的全球
WAP规范
WAP规范是一种无线应用程序的编程模型和语言，它第一次定义了一个开放的标准结构和
WAP规范的要素主要包括:
WAP编程模型
在设计WAP规范时，WAP论坛一个很重要的指导思想是让WAP尽可能多的与现有的标准一致
图1

For Help, press F1　输入文字　位置: x=494, y=45　压力: 727

Fig. 7　Pen gesture interaction in word processing

This example employs five variables: variable "pen_pos" refers to pen's position on the tablet, "cursor_pos" refers cursor's position on the screen, "pen_pres" is for pen's pressure to the tablet, "track" is a buffer for strokes and "t" is for the interval of strokes. "pen_pos" and "cursor_pos" are variables with type "POS" represented as (x,y), "pen_pres" and "t" are real-valued variables. "t" complies with the law that its derivative of the time equals 1 (i.e. a clock-compliant variable). The specification is given in Figs. 8 and 9.

```
Interactor paper
  {
      InputVariable pen_pos[POS],pen_pres[REAL]
      FeedbackVariable cursor_[POS],track[ARRAY]
      ControlVariable t[REAL]
      Event
      {
          GestureStart{pen_pres=0→Pen_pres>0}
          GestureEnd{pen_pres=0∧t<0.5→t≥0.5}
          StrokeEnd{pen_pres>0→pen_pres=0∧t<0.5}
          StrokeStart{pen_pres=0∧t<0.5→pen_pres>0}
      }
      Action
      {
          recognize&execute: recognize the gesture and execute actions
      }
      Flow
```

```
{
    idle{$l_0$: pen-pres$=0\rightarrow f$}
    stroke{$l_1$: pen-pres$>0\rightarrow f\wedge g$}
    Interval{$l_2$: pen_pres$=0\wedge t<0.5\rightarrow f\wedge t=1$}
}
Function
{
    $f$: modify the cursor's position according to pen's position
    $g$: add a point to the track buffer
}
```

Fig. 8 Specification of pen-gesture interaction

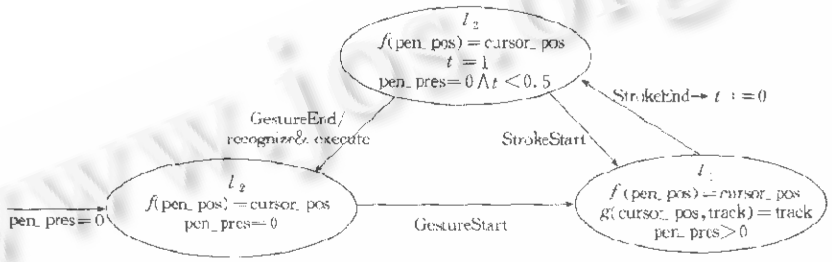


Fig. 9 Graph specification for pen-gesture interaction

6 Discussion

6.1 Constructing post-WIMP user interface

The aim of specifying the user interface is to efficiently construct it. The specification of LEAFF can instruct the designers to construct user interface step by step. The functions and actions of LEAFF can be implemented as "functions" of programming languages and the variables can be mapped to "variables" of programming languages. At present, the transformation from specification to interactive system is manually achieved and it will be furthered for automatic conversion. The specification of LEAFF is interactive-behavior-oriented and has nothing to do with the details of implementation. A specification is presented as a module encapsulated as an object of "Interactor". A specification based on LEAFF can provide great reusability. In order to make it more easily reused, the facility of object-oriented will be extended into the specification, such as inheritance and aggregation. Consequently, it might improve the productivity of the development of post-WIMP interface.

6.2 Parallel issues in post-WIMP user interface

In post-WIMP interface, there are two parallel situations. One situation happens among dialog threads executed by agents. An interactive task is usually executed by the cooperation of several agents. Each agent is specified as an "Interactor" which equals a hybrid automaton. So this situation can be mapped into a set of parallel hybrid automata. The interaction among these automata is quite complicated. We are not going to discuss it in detail and it's our future work. However, some issues of inter-automata interaction are mentioned here. Synchronization is a major problem in parallelism. We use the method of hybrid automaton to meet the requirement by the labels binding with the transition edges in hybrid automata. The transitions with the same label in different hybrid automata should be executed at the same time. So the transition of one location will affect the transition of another location. Another way for the interaction among these automata is through accessing the variables of each other, which is mentioned in Section 4.

The other parallel situation is multimodal interaction. The states of modalities can be specified by a set of variables. The alterations of these states in the same location take place simultaneously. So it embodies the parallel aspect of modalities, which brings multiple I/O streams to post-WIMP interface. Some functions based on these variables are defined for multimodal integration and the values of the functions are results of integration. These results can be classified into different layers based on the ways used by the integration[14]. The multimodal integration in LEAFF will be furthered in our future work.

6.3 Property verification

One of our purposes to use hybrid automaton is to utilize the existing theories and methods of it to do some verification. Although verification is actually a hard work, we can also benefit from hybrid automaton. Here a brief introduction to the verification of real-time property is presented. It is usually testified by shrinking the problem into a well-defined domain. Real-time interaction is one of the most significant attributes of post-WIMP interface. The traditional automaton is based on discrete token. The discrete time variables have to be imported in order to specify the real time attribute. Consequently, it is difficult to specify complex, high-requirement of real-time task system due to the time continuity. Hybrid automaton combines discrete events with continuous alterations and any variable of it can be time-based. So it is convenient to specify real-time issues by hybrid automaton. Due to the complexity of the practical hybrid system, timed automaton is used more often, which is a special case of a linear hybrid automaton (Linear hybrid automaton is a hybrid automaton with special property that for each time-related variable the rate of change is constant)[4]. The real-time problems can be testified in timed automaton by the solution of the reachability problem.

7 Conclusion

We employ the formal method to model and specify post-WIMP interface. A specification model and a specification language LEAFF based on hybrid automaton have been presented in this paper. It is based on the analysis for the essence of post-WIMP interface. We have modeled post-WIMP interface as a set of hybrid automata and the temporal model of interaction in post-WIMP interface has been described. Two typical instances of post-WIMP interface have been specified based on LEAFF. LEAFF specifies post-WIMP interface by the combination of the text-based specification and graph-based specification. Some open issues and future work have been discussed. The construction of the post-WIMP interface can be fulfilled by the instruction of the specification.

Acknowledgement We thank Prof. Robert Jacob for discussing some issues about post-WIMP interface and Doctor LI Guanyuan for the assistance on formal techniques.

References:

[1] Jacob, R.J.K. Human-computer interaction: input devices. ACM Computing Surveys, 1996,28(1):177~179.

[2] Jacob, R.J.K., Leonidas, D., Stephen, M. A software model and specification language for non-WIMP user interface. ACM Transactions on Computer-Human Interaction, 1999,6(1):1~46.

[3] Jacob, R.J.K. A specification language for direct-manipulation user interfaces. ACM Transactions on Graphics, 1986,5(4):283~317.

[4] Alur, R., Courcoubetis, C., Halbwachs, N., *et al.* The algorithmic analysis of hybrid systems. Journal of Theoretical Computer Science, 1995,138:3~34.

[5] Markopoulos, P., Johnson, P., Rowson, J. Formal architectural abstractions for interactive software. International Journal of Human-Computer Studies, 1998,49(5):675~715.

[6] Duke, D.J., Harrison, M.D. From formal models to formal methods. In: Taylor, R.N., Coutaz, J., eds. Proceedings of the ICSE'94 Workshop on Software Engineering and Human-Computer Interaction. Berlin: Springer, 1994. 159~173.

[7] LI, Yang, GUAN, Zhi-wei, WANG, Hong-an, *et al.* Design and implementation of a UIMS for component-based GUI development. In: Proceedings of the International Conference of Software, IFIP's 16th World Computer Conference. Beijing: Electronic Industry, 2000. 955~955.

[8] Andries, Van Dam. Post-WIMP user interfaces. Communications of the ACM, 1997,40(2):63~67.

[9] Green, M., Jacob, R. Software architectures and metaphors for non-WIMP user interfaces. ACM Transactions on Computer Graphics, 1990,25(3):229~235.

[10] Lewis, H.R., Papadimitriou, C.H. Elements of the Theory of Computation. Beijing: Tsinghua University Press, 1999.

[11] FANG, Zhi-gang. 3D space controller and its application in 3D space interaction technology. Journal of CAD & CG, 1998, 10(2):105~111.

[12] LI, Yang, GUAN, Zhi-wei, CHEN, You-di, *et al.* Research on gesture-based human-computer interaction. Journal of System Simulation, 2000,12(5):528~533.

[13] Andre, Meyer. PEN COMPUTING: a technology overview and a vision. ACM SIGCHI Bulletin, 1995,27(3):46~90.

[14] Dong, Shi-hai, Wang, Jian, Dai, Guo-zhong. Human-Computer Interaction and Multimodal User Interface. Science Publisher, 1999.

基于混合自动机的Post-WIMP界面的建模

栗 阳， 关志伟， 戴国忠

(中国科学院 软件研究所 智能工程实验室,北京 100080)

摘要：Post WIMP界面作为继当前的主流界面范式——WIMP界面后的下一代界面范式,它和WIMP界面有着很大的不同,通过使用虚拟现实、语音交互、手势交互等技术,它能够提供更加自然高效的交互方式.然而,它却难以构造.为了有效地构造Post-WIMP界面,在构造之前不考虑实现细节,而在一个抽象的层次上描述它是一个较好的方法.首先,分析了Post-WIMP界面的交互本质,交互混合性是Post-WIMP界面一个最为重要的特点.从形式化系统的角度分析Post-WIMP界面,通过将Post-WIMP界面抽象为混合系统能够更为准确和严格地分析Post-WIMP界面的特性.混合自动机是用于描述混合系统的形式化工具,将Post-WIMP界面建模为一组相互协作的混合自动机.设计了一基于混合自动机理论的半形式化语言LEAFF作为Post-WIMP界面的描述工具.LEAFF通过结合文本描述和图形描述描述Post-WIMP界面中的交互行为,能够准确地反映交互中的控制关系、时序关系.给出了对两个典型Post-WIMP界面——虚拟现实交互和笔式交互的描述实例,同时讨论了Post-WIMP界面中交互并行性的描述、交互实时性的验证和从描述到实际交互系统构造的转换.

关键词：人机交互;Post-WIMP界面;混合自动机

中图法分类号：TP311 **文献标识码**：A