An Intelligent Event-Driven Interface Agent for Interactive Digital Contents in Ubiquitous Environments

Sukhoon Kang[1] and Seokhoon Bae[2]

[1] Department of Computer Engineering, Daejeon University
96-3 Yongun-Dong, Dong-Gu, Daejeon, Korea 300-716
shkang@dju.ac.kr
[2] INUS Technology, Inc.
601-20 Yuksam-dong, Kangnam-ku, Seoul, Korea 135-080

Abstract. Interactive digital contents in the field of networked media and ubiquitous computing enable the consumers to test the features of the product from their handheld computers as if they were using it in real life, by simulating the actions and responses of the product. This new type of digital content can be used extensively to make sales personnel training manuals, sales tools, user manuals and user trouble shooting documents. In this paper, we present the enhanced characteristics of the event flow chart with which the events in an intelligent event-driven interface agent for interactive digital contents in ubiquitous environments, named PlayMo-based agent, are structured. A tree structure can be formed from the array of options or functions. PlayMo-based agent generates the events and gives action commands according to this tree structure, allowing user to perfectly simulate the features and functions of a product simply and directly. The solution to provide intelligent interaction between human and digital contents makes it possible to recreate digital contents by bringing interactivity and intelligence into it.

1 Introduction

What is an 'interactive digital content'? Simply put, it is digital content that enables the consumers to test the features of the product from their PCs as if they were using it in real life, by simulating the actions and responses of the product. Take the case of an interactive digital content for a mobile phone for example. The user will be able to open the flip panel of the mobile phone by clicking on it. Furthermore, the user will be able to see the numbers being displayed on the mobile phone screen when he/she clicks on the number pad of the phone in the catalog. In other words, an interactive digital content can simulate hundreds of different features to match the features of the real mobile phone to perfection.

An event-driven modeling for interactive digital content, named PlayMo [1, 2, 3], enables the authoring of new digital contents to simulate real world products such as cellular phones, various electronics, machinery and systems on PC. PlayMo-based agent's independently developed engine, which increases the fidelity of the interactive contents dramatically, provides professional approach in creating layered events and

M. Denko et al. (Eds.): EUC Workshops 2007, LNCS 4809, pp. 93–102, 2007.

actions in a visual environment, permitting novice to develop contents without coding or programming. The resulting contents could be compressed into less than a hundred kilobytes for uploading on the web. Also, compared to conventionally available expensive and complicated professional graphic and simulation programs, the PlayMo-based agent provides an inexpensive solution, which allows anyone to create interactive contents easily.

Interactive digital content by using the PlayMo-based agent makes easy, simple and effective design for e-learning, e-catalogue, e-marketing/sales, e-prototyping, customer support, etc. Through its application-ready 3D function visualization solution, engineers and designers can rapidly turn a CAD design model into a 3D interactive virtual product, and the effective function prototyping job can be also completed within a minute.

2 Why Intelligent Event-Driven Interface Agent

PlayMo offers a unique and intuitive modeling method, which received a patent for 'Visual Event Driven, Free Coding Modeling Method'[1] and continuously has refined with collaborative research of academy [2, 3]. PlayMo itself is a powerful developing environment that allows the creation of highly interactive rich media content from a wide range of source files such as still images, video clips, audio clips, 3D models (VRML), CAD models and more. PlayMo's intuitive WYSIWYG editing functions make it easy to embed complex interactivity into models to accurately recreate the functionality of real-world objects, requiring no coding. Publishing electronic products, mechanical components, and dynamic multimedia contents with PlayMo allows users to interact with objects on the internet as if they were real. It is easy to create computer-based training or online/offline education materials based on PlayMo models, and user manuals based on published models for complicated products can be created automatically. PlayMo's content, delivered online or offline, offers enticing product experiences, intuitive web-manuals, effective learning tools, and impressive presentations. The solution, PlayMo-based agent, with the goal to 'provide intelligent interaction between human and digital contents' makes it possible to recreate digital contents by bringing life and intelligence into it in ubiquitous computing environments.

2.1 Notion of Event-Driven Object with Event Flow Chart

Event is an occurrence, which could trigger a change or reaction in PlayMo-powered contents. In other words, event defines user action, which cause specific change or act ion on the object. These events are assigned to each event node, and event nodes are i nterrelated with each other in a hierarchy.

PlayMo-based agent enables users to create intelligent 3D contents which react upon various inputs according to the predefined rules of external and internal conditions. The internal structure of this event-driven development environment is made of three independent concepts. The three axes, which constitute PlayMo-based agent are *Object*, *Event*, and *Action*. Object refers to a material such as pictures,

multi-media and text which forms the PlayMo-based agent contents externally. Event refers to an occurrence which could alter or trigger a reaction on interactive digital contents. Action is a result or alteration of interactive digital contents when the predefined event satisfying the condition occurs. PlayMo-powered contents could be defined as organically combined contents which react intelligently to dynamic user inputs.

For example, assume operating an actual cellular phone. It has an appearance of a cell phone as well as its flip, folder, function buttons, sound, LCD screens are its components. PlayMo-based agent handles these as images, GIF Animations, A/V clips and Sound files. These are defined as Objects, the external factors. Next, a user would open the flip or folder on the phone and would press many different buttons. Then, the flip or folder would open, and the LCD display would change with the appropriate sound. In that case, we may define the action opening the flip and pressing the button as external Event being applied to the phone. With these events applied, the cell phone's flip or folder would open, with the buttons pressed, sounds or display will change. These are defined as Action.

More specifically, PlayMo-based agent generates contents identical to actual product, or intriguing contents possible in the virtual world by creating Objects necessary in the product, and assigning Events such as a mouse click, keyboard input, timer pulse signals, and with an appropriate event, executes variety of Actions possible in the actual item. In PlayMo-based agent, we can organize all the events in a visual event tree using the Event Flow Chart technique. Compared with the traditional State Chart [4] technique, the Event Flow Chart, which controls the events according to the order of precedence and hierarchy, defines events more concretely, explicitly and intuitively. An 'event tree' is the hierarchy of functions or the order of interconnected event nodes. Fig. 1 shows this tree structure of events. Events are activated from the 'root' and there are three kinds of events in order of precedence: *parent event*, *sibling event* and *child event*.

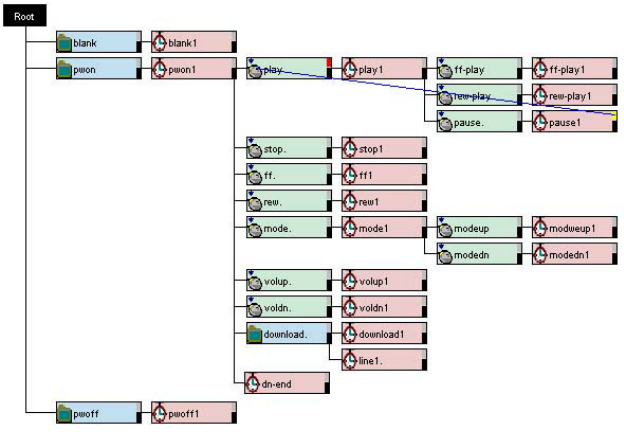

Fig. 1. An Event-driven Modeling Diagram for PlayMo-based Agent

In order to complement control flow more perfectly, we enhanced the Event Flow Chart by extending two characteristics of it:

(1) Unconditional Flow: An unconditional flow proceeds compulsorily. All events, except the events linked with unconditional flow, are ignored during an unconditional flow. This simplifies the Event Flow Chart by avoiding repetition.

(2) Synchronous Flow and Return Flow: Global stack is used for storing the states. When a synchronous event occurs, the current state is pushed into the stack. On the contrary, states in the stack will be popped up when a return event occurs. A synchronous event, as an interrupting event, is needed for the system to return to the original state.

Rules of governing flow are as follows. Going from one event location to the next event, there is an event that should be executed and an event that should not be executed. In other words, it cannot be executed a close command when the lid hasn't even been opened. From these laws of physics, PlayMo-based agent gives a few rules governing the flow of events:

[Rule 1]: May flow to an event node that is one step below as shown in Fig. 2.
[Rule 2]: May flow to event nodes on the same level as shown in Fig. 3.
[Rule 3]: May flow to an event node that is many steps above.
[Rule 4]: May not flow to an event node located more than two steps below as shown in Fig. 4.
[Rule 5]: May not flow from an equal level or higher level event node to a lower level event node on a different branch as shown in Fig. 5.

Let's explore first of all, the relevant event node from the child node of a currently activated event node.

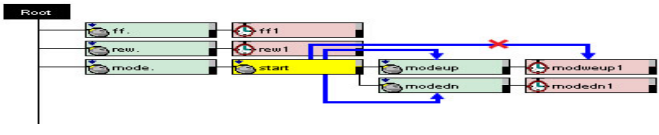

Fig. 2. [Rule 1]: May flow to an event node that is one step below

If nothing is found from the above search, explore sibling nodes connected to the same parent node.

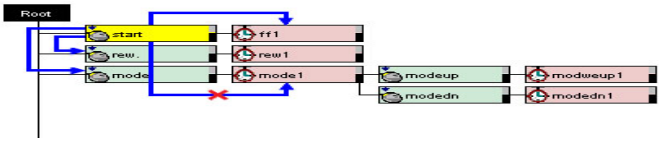

Fig. 3. [Rule 2, 3]: May flow to event nodes on the same level and to an event node that is many steps above

Search other levels of parents from the activated node, or its siblings. This only connects to the direct branch connected to the Root. In other words, you cannot go from a sibling parent that is not its direct parent to its offsprings.

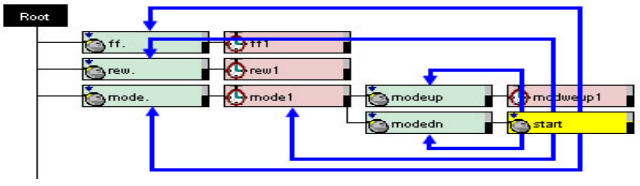

Fig. 4. [Rule 4]: May not flow to an event node located more than two steps below

There may be areas where the above rules cannot be used to connect, but a connection is necessary nevertheless. We use the Goto Line in these instances. Drag from the box located on the top right hand corner of the activated event node to the event node to be connected. This creates a blue line. This will move the flow to the Goto Line connected point, immediately when the flow reaches the activated event node. At this time, check GotoEventWithAction from the even panel options. As it is being moved, it will execute the action included in the event node that is located in the position it is being moved to. When you check the next GotoEventWithoutAction, it moves without executing the action in the event node.

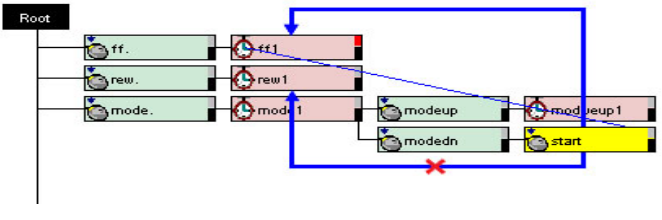

Fig. 5. [Rule 5]: May not flow from an equal level or higher level event node to a lower level event node on a different branch

2.2 Seamless Integration in Ubiquitous Computing Environments

It is reasonable to define ubiquitous computing as the attempt to break the pattern paradigm of the traditional relationships between users and computational services by extending the computational interface into the user's environment [5]. Given the large numbers of devices and services that users will encounter in ubiquitous computing environments, there are many situations that will require that users serendipitously compose available resources into configurations suitable for their current activities. A scaleable interface to interactive digital contents would provide a unified service with multiple device interfaces. The challenge in this case is to provide free-coding capability to enable radically different interaction methods with the same underlying data and service. An intelligent event-driven interface agent for interactive digital

contents in ubiquitous environments is only one part of what we call a ubiquitous software service, a service that actively searches out the user at convenient and salient times. Users interact with a variety of services and there are some obvious connections between the information that each service manipulates. Furthermore, this set of services is constantly subject to change. How can we provide intelligent ways to integrate the behavior of these different services without requiring additional programming effort by both the designer of a service and the end user? With intelligent event-driven interface agent, developers and end-users will better understand a concept via rich media representation. By way of designing an intelligent event-driven interface agent, the look-and-feel aspect will provide a life-like image immediately. PlayMo itself is designed by way of a user-friendly approach without coding. As such, the trouble of programming and algorithms are unnecessary: Simply, click, label, paste, and scroll.

2.3 Comparison with Other Works

Behavior representation models are classified as event model and state model. In event model, object action is constant in response to an event. Therefore, the characteristics of easy-to-learn and intuitive event flow design are definitely strong points, and exponentially increasing complexity, e.g., Web3D (Viewpoint, Cult3D, etc.), is weak point.

In state model, object action is varying according to its state in response to an event. Therefore, effectiveness to represent highly complex action relations seems to be strong point. The characteristics of hard-to-learn and initial design, e.g., system development tools (Rapid+, ROSE, OpenGL) are weak points.

Our interface agent model has the following characteristics of state and event conjugated model:

1. State-of-the-art concept to take full advantage of each model.
2. Automated state model conversion from designed hierarchical event tree.
3. Minimizing authoring cost while maximizing operation reality.
4. The most effective way to represent complicated operation relations.

Visual representation quality or richness of all kinds of simulation object such as bitmap image, movie clip, 3D model, sound clip, etc are the important factors. In case of 3D model, texture mapped rendering quality is the most crucial factor.

Regarding level of interactivity, in lowest level, sequential simulation is based on limited number of pre-defined scenarios. In highest level, arbitrary simulation is in response to user's random input (event).

Operational reality is the ratio of actual model's number of actions (functions) to simulation model's. Perfect operation reality is not always required to achieve a simulation purpose.

Regarding application-specific features, simulation system's capability to support application specific functionalities can be operation history logging, printable document generation, etc. It makes a direct effect on overall authoring cost.

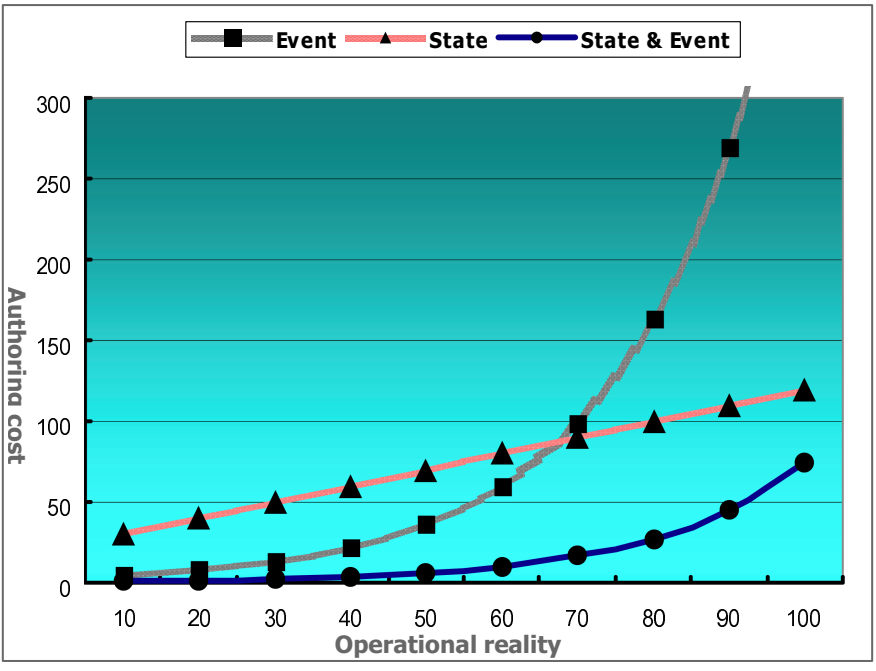

Fig. 6. Operation reality in behavior representation models

Table 1 shows that our interface agent model offers better solutions than similar models in the following fields.

Table 1. Comparison with other tools

Level 1 (no-interactivity)	Level 2 (simple triggering)	Level 3 (low-interactivity)	Level 4 (high-interactivity)
Single action time-line control	Multiple actions unconditional play & stop control	Multiple actions Non-systematic & conditional play & stop control	Multiple actions Systematic & conditional play & stop control
User controls only playing or stopping a set of sequentially defined actions in a certain time-frame.	Multiple sets of actions are defined and user triggers one or more actions independently. No relations are supported between multiple actions.	Multiple sets of actions are related with each other under the user defined rules described in script or program code. However, simulation engine itself does not support such relations systematically.	Multiple sets of actions are related with each other under the simulation engine based on vendor's own methodology.
Animation	PowerPoint, Viewpoint, Cult3D, Flash		PlayMo, Rapid+

3 Work Flow in Intelligent Event-Driven Interface Agent

A brief description of the process for interactive digital contents involved in working with *PlayMo based agent* is as follows.

Step 1 (Analysis for Product Function): Function to analyze is concrete action such as button. It would be better imagine the structure of a product and consider each reaction of each event in right or wrong action.

Step 2 (Preparatory to Image Source): It is necessary to get image source of product to design include sounds and motion images. Crop the image into object images to each function then save them.

Step 3 (Object Import and Property Set Up): Create new project and import saved image source in OBJECT mode. Arrange each image at proper position. Create other multimedia sources such as html, motion images, sound, timer and the like. Then, set up the property type to each object.

Step 4 (Creation of Events): Create events on each image source depend on user command like mouse click, over, keyboard hit and so on.

Step 5 (Creation of Action): It is necessary to create action on each event. An action means real interactivity such as 'show or hide of object' and 'play on and off music'.

Step 6 (Arrangement of an Event Tree): Event has the order system as we can play music after turn on the power. This order structure is almost similar to that of real product. A button could do another action at certain situation, however, it could be treated all the complete problems related the function structure with event tree.

Step 7 (Simulation): It is possible to see the interactive digital contents like real products with simulator.

Step 8 (Debugging): Start debugger, it is possible to see which event is acted at real time and also see list of action activated on output window. With Debugging tool and Watch Window, it is possible to check out special action that users want. After go through these kind of whole things, users could got perfect product simulation.

4 Implementation of Intelligent Event-Driven Interface

One of the main interface functions in PlayMo-based agent for interactive digital contents is as follows.

1 **Title Bar:** Default title bar displayed when PlayMo is launched.

2 **Menu Bar:** When PlayMo is opened initially, menus are listed in the order of File, View, Tools, Help. As the picture displayed above, when a new content is being created, or existing contents file is opened(.at3), the menu will appear in the order of File, Edit, View, Object, Event, Debug, Tools, Window and Help.

3 **Icon Bar Set:** Icon Bar Set is a set of icons including a series of commands frequently used in the menu to provide assistance. Title of the icon is displayed when the mouse pointer is on top of the icon. Icon will change into pressed form when clicked.

4 **Project Tab:** Displays name of the project and may change work area.

5 **Object Window:** Window aligning Objects such as 3D image of the target product, A/V files according to the aim of the user. The Object panel is used to open a file.

6 **Event Window:** Window creating an Event node to assign Event and Action command to the opened 3D Object. Event node identical to the picture above is created and interconnected. Event and Action commands are executed in each event and action panel.

7 **Panel:** On the Panel, users can define the attributes of active objects and event nodes. The Panel is divided into Object, Material, Event and Action panel. When each panel is selected with the mouse, that particular panel is activated enabling the user to define the attributes of each of them.

8 **Status:** Shows the current status of PlayMo. Located on the bottom of the GUI.

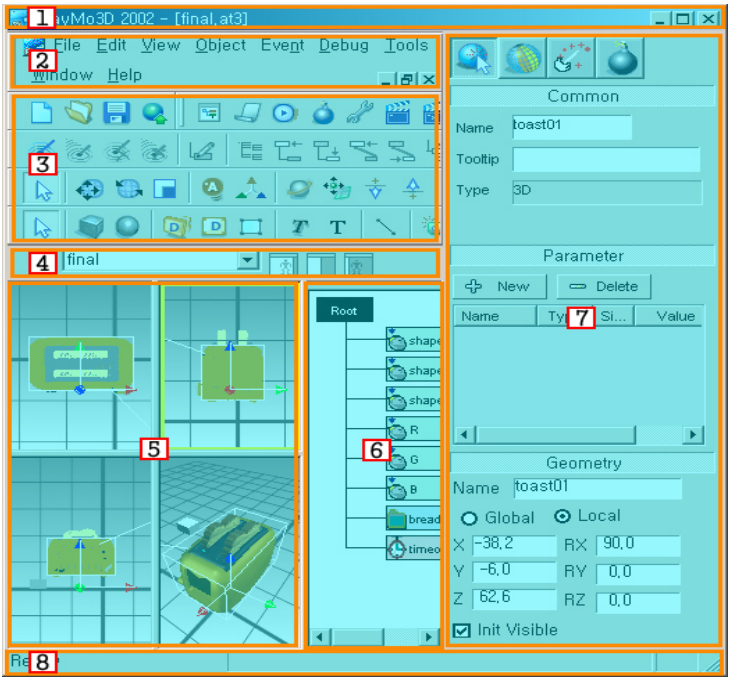

Fig. 7. Example of Intelligent User Interface for PlayMo-based Agent

5 Conclusions

Ubiquitous computing focuses on user interaction in real operating environments. Both ubiquitous computing and interactive digital contents are strongly linked together. We think one important point in ubiquitous computing is that the devices provide smart support to user without forcing the user to change behavior. The access to devices should not be complicated due to the fact that they offer more functionality. Often the opposite is the case - the devices become too complex and people just use

the very basic functions. We emphasize that the way people will interact with their environment is not changed in general.

Intelligent event-driven interface agent is dedicated to deliver smart and simple environments to enrich the contents on websites, education, advertising, or industrial design and modeling. By way of designing an intelligent event-driven interface agent, the look-and-feel aspect will provide a life-like image immediately. PlayMo itself is designed by way of a user-friendly approach without coding. As such, the trouble of programming and algorithms are unnecessary. Simply, click, label, paste, and scroll.

Content powered by intelligent event-driven interface agent improves management decisions, perks marketing content, simplifies training materials, and intensifies learning resources. All data is digital, and can be retrieved and stored anywhere at anytime.

Integrating intelligent event-driven interface agent with product data management, customer relationship management, supply chain management or other IT technologies in terms of product or component realistic visualization, fully integrative function simulation and real time product customization etc., users can get the priceless value and take advantage of using the virtual work environment on products through internet or mobile platform over the whole life cycle of the product. It is important to note the fact that our interface agents can will be deployed in real applications in the short term because they are simple, operate in limited domains and do not require cooperation with other agents.

References

1. INUS Technology, Inc., Visual Event Driven, Free Coding Modeling Method, Patent (2001), http://www.playmo.com
2. Kang, S., Hur, C.J.: PlayMo: An Event-Driven Modeling Tool for Active Catalog. In: Proceedings of 29th EUROMICRO Conference, Belek, Turkey (2003)
3. Kang, S., Bae, S., Hur, C.J.: A Development Environment for Interactive Digital Contents: PlayMo3D. In: Proceedings of 30th EUROMICRO Conference, Rennes, France (2004)
4. Harel, D.: STATEMATE: A Working Environment for Development of Complex Reactive System. IEEE Transaction on Software Engineering , 403–414 (1997)
5. Lindenberg, J., et al.: Improving Service Matching and Selection in Ubiquitous Computing Environments: A User Study. Personal and Ubiquitous Computing, 59–68 (2007)