EXTENDING THE UNIFIED MODELLING LANGUAGE TO MODEL COMPLEX INTERACTIVE TIMELINE-BASED SYSTEMS

Ian Smith
HCI Research Group
School of Computing
Napier University
10 Colinton Road
Edinburgh EH10 5SD
ia.smith@napier.ac.uk

Phil Turner
HCI Research Group
School of Computing
Napier University
10 Colinton Road
Edinburgh EH10 5SD
p.turner@napier.ac.uk

Iain McGregor
HCI Research Group
School of Computing
Napier University
10 Colinton Road
Edinburgh EH10 5SD
i.mcgregor@napier.ac.uk

ABSTRACT

The increased take-up of Interactive Television, 3G and broadband networks is effecting a growth in the use of interactive, timeline-based media. This paper will describe how to facilitate the design of linear narrative systems utilising the extensibility features of the unified modelling language (UML).

The UML provides a solution to model the integration of such diverse media as 3D animation and still imagery within a single design solution. Current technologies extend well beyond their established models, incorporating UML into the design process will bring differing media technologies into the realm of the software developer, facilitating a cohesive approach to interactive systems.

Keywords

Timeline, multimedia, design, composition, UML

1. INTRODUCTION

The increased take-up of Interactive Television, 3G and broadband networks is effecting a growth in the usage of interactive, timeline-based media. Interestingly while there are significant developments in the supporting technology the same cannot be said for the modelling or design tools. Indeed, there is a growing gap between envisagement of the target interactive application and its techno-centric implementation [1].

This paper is an attempt at bridging the gap between the current implementations of time line-based design and a formal modelling language. We propose the use of the Unified Modelling Language (UML) as the basis of a method for the effective design of interactive multimedia systems. We will demonstrate that complex compositions of software objects can effect a method of modelling diverse timeline-based media.

We begin by briefly introducing the elements of UML relevant to our argument before describing the extensions to the language in order to model timeline based design. We conclude with an example of its potential application.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

Proceedings Volume 2 of the
16th British HCI Conference
London, September 2002

© 2002 British HCI Group

2. UNIFIED MODELLING LANGUAGE

The UML is (or at least aspires to be) a standard modelling language that unified the dominant object-oriented analysis and design concepts developed in the 1980s and 1990s. In addition to its notation (syntax) and meta-model (diagrams) it is extensible via a number of mechanisms not least stereotypes, which are discussed below.

In recent years the UML has attracted the attention of the HCI community as it offers the promise of an integrated OO (Object Oriented) & HCI (Human Computer Interaction) approach to the development of interactive systems which is now reasonably well established with a range of proposed methods compatible with the UML [2]. Some of the pioneering research on establishing a role of UML in HCI and CSCW (computer supported cooperative working) has been carried out by Sommerville and his team at Lancaster. A perennial problem for HCI and CSCW has been the gulf between the qualitative data gathered using techniques drawn from the social sciences and the formal representations of computer science.

Sommerville has tackled this problem in a number of ways the most relevant of which to the current discussion has been in using the UML to model ethnographically acquired data [3]. While a full treatment of this work is beyond the scope of this paper it is worth noting that he successfully used constraints, tagged values and stereotypes to model awareness in a conference-booking centre.

More recent work in a similar vein was presented at a UML symposium [4] jointly hosted by the British HCI Group and the Scotland IS Usability Forum. Martin et al [4] presented work on using the UML to communicate ethnographic data to systems designers in a paper of

the same name. Their two key findings were that *patterns* proved to be an effective means in communicating ethnographic data to designers. Secondly, they were able to build upon lessons learned from these studies in, for example, standardising design artefacts. Similarly Johnson has extended the UML to model collaborative tasks. He and his colleagues have been able to model and account for the behaviour of an aircrew during an incident. To this diversity of applications of extended UML we now contribute our own.

3. SOFTWARE OBJECTS

A software object comprises state, behaviour and identity information. All software objects can be described through an abstracted class, an object being an instance of that class. The class of an object is defined in terms of its identity, attributes, operations and relationships. A class diagram is utilised in the UML to describe the static structure of a system.

The values of the attributes define the state of an object. Within the UML, state diagrams are used to describe the dynamic behaviour of a system.

3.1 Extensibility

The UML has three key extensibility mechanisms: constraints, tagged values and stereotypes.

A constraint is a Boolean condition used to define the relationship between model elements. A constraint is expressed within braces ({ }). Predefined examples are {or} and {xor}.

A model element may have a tag defined with a related value, which can be used to associate additional data. These keyword-value pairs are expressed within braces.

Stereotypes permit the creation of a new model element derived from an existing one. A stereotype is expressed within guillemets (<< >>).

3.2 Timeline Object

A range of multimedia authoring environments have traditionally utilised the timeline paradigm. Objects have associated timelines that determine the current visibility, persistence, transition and functionality. The key components of a timeline are layers (or channels or tracks depending upon the medium), frames, and the playhead. The content of a movie is determined over time by the status of the layers and frames. A timeline, as shown in Figure 1, can be directly controlled or programmed through the application of key frames, scripting and markers.

The dynamic structure of a timeline object can be described with respect to its internal timeline. This is composed of individual units of time called frames. Each frame can incorporate a number of layers. A layer can incorporate a range of objects that may alter on a frame-by-frame basis. Consequently, the attributes and operations of the timeline object can change. This introduces the concept of *dynamic classification* [5] where the class of the object is dependent upon the state of the timeline.

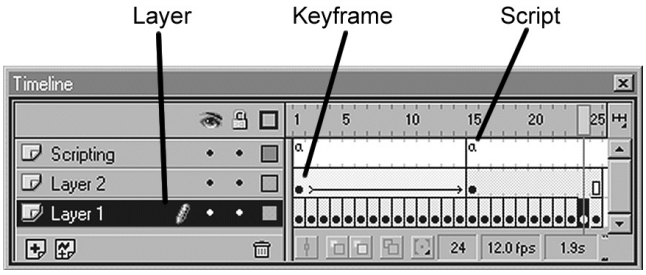

Figure 1: Timeline

3.3 Dynamic Structure

Currently the UML describes the static structure and dynamic behaviour of an interactive system. However, modelling a linear narrative requires the concept of *dynamic structure* that describes the relationship of classes through time.

Dynamic classification models a change of class during an object's lifetime. (e.g. an academic object instantiated as a research associate can become a professor over time). This is appropriate when describing a class model during the conceptual perspective. The term dynamic classification refers to a variation of generalisation in which an object can change type or role. It is modelled by means of a <<dynamic>> stereotype as shown in Figure 2.

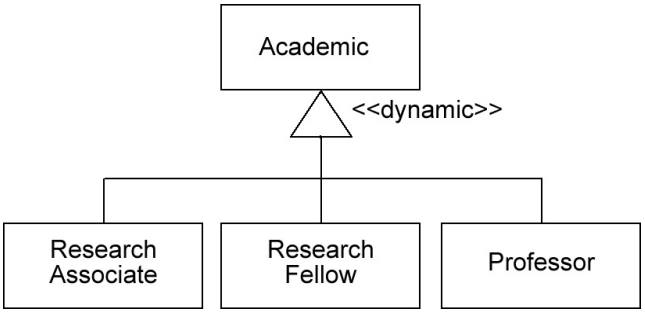

Figure 2: Dynamic Classification

When describing a timeline object from the implementation perspective, the apparent change of class is defined by the current state of the timeline. A timeline object can be considered as an encapsulated composition of a number of objects. The accessibility of these objects from the perspective of the system is dependent upon the content of the individual layers on particular frames.

Dynamic behaviour describes how an object changes state. The values of the attributes define the state of a

software object. The state of a timeline object is influenced by the status of the internal timeline.

4. EXTENDING UML TO MANAGE COMPLEX COMPOSITION

A composition is a specialised form of aggregation, within which the lifetime of the part is bound within the lifetime of the composite class (whole). For reasons of clarity we shall describe this as *static composition*.

Developing classes through composition is generally regarded as being better practice than by inheritance, which encourages tight coupling [6]. We propose that with respect to timeline objects, there are three types of composition, namely: static, timeline and dynamic.

4.1 Timeline Composition

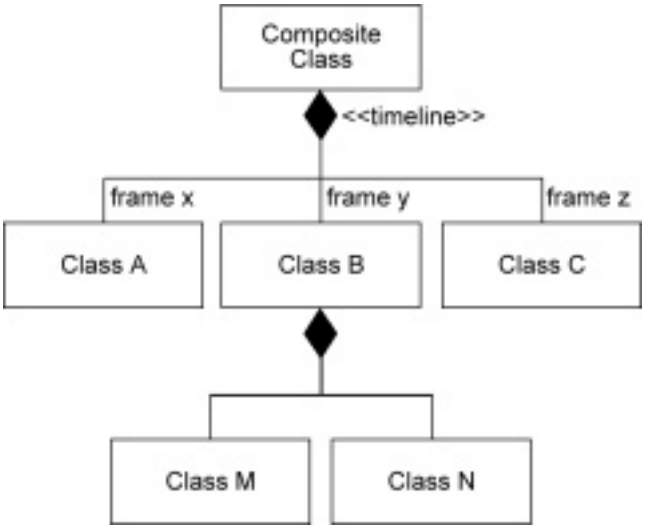

Figure 3: Timeline stereotype

The dynamic structure of a system incorporating timeline objects can be modelled using the extensibility of the UML. The introduction of a timeline stereotype, <<timeline>>, associated with a composite aggregation would define the permutations of compositions. Each permutation could be labelled with a discriminator to indicate the basis of the composition as shown in Figure 3. The Composite Class is composed of Class A, Class B (a composite of Class M and Class N) and Class C all of which are destroyed if the whole is destroyed. The apparent type of the Composite Class is dependent on the position of the timeline.

Though a discriminator may allude to a possible state of an object, only the dynamic structure is modelled. A state diagram would be required to model the transition from one state to another.

This technique was initially conceived to model Flash-based games, but is applicable to the development of any timeline-based form of digital media. Figure 4 shows the class model for an interactive movie that has a timeline composed of 1 or more scenes and zero or more commercials.

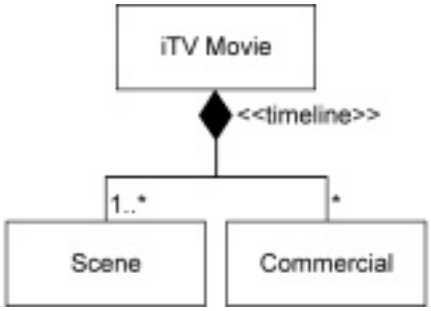

Figure 4: interactive Movie example

4.2 Dynamic Composition

The concept of *dynamic composition* [7] allows a change of aggregation in an existing structure. The notation is similar to that for a timeline aggregation, except that the dynamic stereotype, <<dynamic>>, is used as modelled in Figure 5.

The composite class can be a permutation of all or some of the aggregated classes all of which are destroyed if the whole is destroyed. With respect to timeline objects, the apparent type of the composite class is dependent on the content of individual layers that can be changed during run-time.

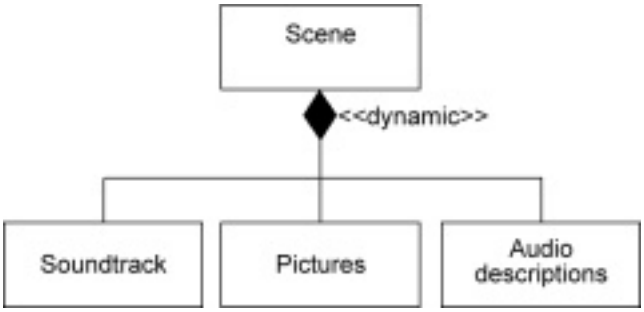

Figure 5: Example of Dynamic Composition

This technique is particularly applicable to Interactive Television and similar media, as demonstrated in Figure 5. This simplified class diagram of an interactive movie scene, models a dynamic composition of pictures, soundtrack and audio descriptions. To minimise bandwidth only the media required would be streamed but the options could be revised on demand.

5. KEY APPLICATIONS

With this innovation, interactive movies can be modelled as a complex composition of static, timeline and dynamic aggregates. This is illustrated in the scenario: a user wishes to watch the original Star Wars™ movie in its new interactive form. Their first choice is whether to subscribe without advertisements, or to view it for free but with

advertisements. We can think of these adverts as being either regional or tailored to the user's profile.

The user is then presented with a further raft of options, just as with current DVD packaged movies, namely, a choice of languages or sub-titling and so forth. We can also envisage this process being automated if they have already viewed an interactive movie through the same service provider. The remote server is easily made aware of the user's hardware configuration and scales the movie accordingly, utilising aspect ratio and number of audio tracks, and an advanced version of "pan and scan".

Should the user stop the movie at any point, then upon restart a recap is offered, based upon where the user left off, the user can also change devices in between. As the user views the movie, a number of functions become apparent, these are accessed in a similar manner to Apple's hidden dock. But all of the major interactive elements are accessible through hotspots within the movie itself. All of these elements appear within pop-up windows, which in turn can control the underlying movie.

6. FUTURE WORK

We have currently developed a conceptual UML model, utilising the aforementioned extensibility features and are currently implementing a working prototype. The intention is to evaluate the suitability of the UML when developing media such as interactive movies rather than test our concept for interactive movies. The implementation should bring to light any missing elements or contractions within the modelling, as the interactive movie is being developed by a large team each having different expectations and needs.

Though the UML is considered robust [8] the range of notation and meta-models are not extensive enough to successfully model traditional linear narrative techniques. It is hoped that the authors' work will address this issue, in addition to the research currently being pursued.

7. CONCLUSIONS

There is currently no standard method of integrating diverse forms of media such as complex 3D games through to simple animated GIFs. It is hoped that this work will help to bridge the gap between traditional media developers and interactive systems developers.

The authors have demonstrated that through extensibility, the UML, currently an industry standard is able to model the dynamic structure of a timeline object. We have also proposed that a timeline can be considered a complex composition of static, timeline and dynamic aggregates.

The UML is compatible with a range of integrated OO&HCI methods that can be used in the design of interactive time-line based applications [2]. This approach would be ideally suited to the design of emerging interactive media.

8. REFERENCES

[1] Chapman N. *Flash 5 Interactivity and Scripting* Wiley (2001)

[2] van Harmelen M. (ed), Object Modelling and User Interface Design. Addison-Wesley (2002)

[3] Viller S. and Sommerville I. Coherence: an approach to representing ethnographic analyses in systems design, *HCI journal* special issue on Representations in Interactive Software Development, vol 14, pp 9-41 (1997)

[4] ScotlandIS Usability Forum One-day Symposium: Usability and UML Web Site http://www.dcs.napier.ac.uk/~mm/uu2002/

[5] Fowler M., *UML Distilled, Second Edition.* Addison-Wesley (2000)

[6] Stevens P., *Using UML: Software Engineering with Objects and Components*. Addison-Wesley (2001)

[7] Neumann G. and Zdun U. Towards the Usage of Dynamic Object Aggregations as a Foundation for Composition *Proceedings 2000 ACM Symposium on Applied Computing* http://www.acm.org/conferences/sac/sac00/Proceed/FinalPapers/PL-18/

[8] Ambler, S. W. *Agile Model: Effective Practices for eXtreme Programming and the Unified Process.* Wiley (2002).