The Design and Implementation of a Script Language and Playback System for Interactive Electronic Book

Chorng-Shiuh Koong
Department of Computer and Information Science
National Tai-Chung University
No.140, Min-Sheng Rd., Taichung, 403 Taiwan
Taiwan

Abstract: - Generally speaking, Electronic Book can be defined as a multimedia software product synthesizing multimedia such as text, voice, audio effect, image, animation and video into a scenario where interaction can be made with the readers during the acting process. An e-book is composed of 3 parts: (1) programming design; (2) script design; (3) art and audio effect design. Producing an e-book is quite labor-consuming since artists like script writer, art designer, and software programmer have to be taken into consideration. Hence, it is hard to manage the progress in developing an e-book, not to even mention supervising the software quality. For this reason, cultivating a supplementary software tool to help developing multimedia e-book is rather essential. This study has devised an e-book environment, specified an electronic book script language (EBSL) and a windows platform to implement an e-book playback system so that multimedia e-book software product can be made possible. The script language is a scene-based, parallel-executing language that can be applied to write e-book script. In addition, its functions in producing animation effect, controlling synchronization and providing interaction with users are prevailing. The system is designed under the idea of software reuse employing the object-oriented technology. To demonstrate the tangibility and capability, we have implemented a compact e-book based on the script language.

Key-Words: - Script Language, Interactive Electronic Book, Multimedia Presentation, E-Book, software component

1 Introduction

Massive labors and resources are needed in producing an e-book, which can be categorized into 3 parts: (1). programming design; (2). script design; (3). art and audio effect design.

(1). Programming design: It is generally done by Authoring Tools or Language. The current Authoring Tools are mostly designed for general purpose. It is rather complicated to be applied to develop multimedia e-book. For example, it is arduous to produce animation effect, and its program execution is less efficient. It also takes a professional programmer to operate the system even if it is controlled by program language.

(2). Script design: There is no fixed script format. Good script designers are needed to search for story topics and to design appealing story scenario, and then to write into script.

(3). Art and audio effect design: Art crews are responsible for designing casts, props, and background pictures for each scene, while the jobs of sound effect crew are to look for suitable sound effects or even to create one in order to meet the needs of the scenario.

Our goal is to create an environment where e-book can be developed through:

(1). playback system + script language: to solve the program problem
(2). script language + authoring system: to solve the script problem
(3). authoring system + resource management system: to reduce the work of art and sound effects

The rationale is that, by using script language to write e-book script, more interactive e-book software product can be produced, and more time and energy can be focused on the designing, art, and sound effect of the program rather on coding.

This study is intended to explore the characteristics of e-book integrating the object-based technology and software component design methodology under the idea of software reuse. We have outlined an environment for e-book, prescribed EBSL, and from which implemented the playback system.
2 The Characteristics of Electronic Book

Firstly, we have designed the general architecture of e-book based on its characteristics, designed the needed software component, and analyzed its control flow.

2.1 Electronic Book Architecture Analysis

Electronic Book is divided into the following categories according to the complexity of scenario and the capability to interact with users:

(1) Single Point: The system plays straight to the end once started. The parts readers can operate are buttons that can control the procession of the story, which are similar to those of the traditional video tape and VCD such as play, stop, pause, forward, and backward. The current CD/DA e-book is classified as the kind (Fig. 1).

(2) Single Line: It is played by SCENE as a unit. The SCENE changing from one to another is fixed, so is the sequence of the program (Fig. 2).

(3) Multiple Line: It is played by SCENE as a unit. There are alternatives between SCENES changing. Different SCENE can be selected according different selection paths. The program in procession can be varied and more flexible (Fig. 3).

(4) Conditional Line: It is played by SCENE as a unit. SCENE changing is varied according to scenario. Readers can interact and take part in the scenario. Different endings can make possible depending on different selections and paths. This is the most appealing e-book for the readers (Fig. 4).

Synthesizing from the above classifications, the characteristics of Electronic Book architecture can be concluded as:

(1) Scene-Based
(2) Interactive
(3) The program is preceded according to the changing of SCENE.

Therefore, the basic architecture of a popular and more flexible e-book is: e-book is composed by many Scenes. Each Scene is divided into Prelude, Interactive, and Finale stages. Each stage is subdivided into Cast List and Action Script (Fig. 5).
2.2 Electronic Book Basic Component Analysis

E-book is a product within which different multimedia such as text, voice, audio effect, image, animation and video are inter-playing. The software component used to demonstrate the multimedia data is called Cast Component. We have examined the functions needed to act on current e-book and categorized the Cast Component as follows (Table 1):

1. Static component: Static component is used merely to display multimedia data. It is not responsive to readers and is not capable of interactive capability, for example, decorated casts like tree and moon.

2. Passive component: In addition to displaying multimedia data, passive component can interact with reader, produce message notify system, and can lead to next event, for example, Image Button, Animation button.

3. Active component: The functions Active component can offer are similar to those of passive component. It can also respond to the results of dialogue and act accordingly. In addition, it can receive message to display a particular program, for example, Animation Cast is equipped with this function.

<table>
<thead>
<tr>
<th>Types</th>
<th>Interactive Capability</th>
<th>Delivering message</th>
<th>Receiving message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Passive</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
</tr>
<tr>
<td>Active</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

2.3 Electronic Book Control Flow Analysis

In e-book, the sequence of each cast is based on the pre-written script. The action control flow can be divided into the following types [1][2]:

1. Sequential: action is done by order
2. ParallelOr: The segment of the program is ended when one cast has finished his action.
3. ParallelAnd: The segment of the program is not ended until all casts finished their action.
4. Loop: Repeat the execution.
5. Delay: Delay the execution.
6. Branch: Separate the execution.
7. If ....Else: Conditional execution.
8. Appear
9. Disappear
10. Stop

3 Electronic Book Developing System (EBDS)

Considering the characteristics and developing flow of e-book, we have projected a developing environment—Electronic Book Developing System (EBDS) for e-book with the adoption of software reuse concept [7], object-oriented technology and designing software component methodology [8]. The reuse scale in the environment includes multimedia software component and multimedia data component. We have named the multimedia software component as Cast Component. The system is divided into 3 parts (Fig. 5):

1. Electronic Book Playback System (EBPS):
2. Electronic Book Authoring Editor (EBAE):
   - Adopting visualized icons to author the sequence, action, and the relative scenario of each cast software component, and then producing e-book scenario corresponds to the script file syntax.
   - Managing the storing multimedia data that can be reused. This makes ease the reuse of multimedia data and saves the labor, resource and time in producing art and sound effect.

This study is mainly focused on the investigation of e-book script language and playback system.

4 Introduction of EBSL

script language which can describe complex scenario.
In our research, we proposed an EBSL script language for interactive multimedia electronic books. The characteristics of EBSL are as follows:
1. Scene-based
2. The multimedia data in e-book is packed into a software component or Cast Component, which are basic control units for reuse, for example, Static Cast component, Image Cast component, Button Cast component, Text Cast component, Video Cast component, Animation Cast component and Audio Cast component.
3. Parallel control: It is capable of controlling cast components that can act simultaneously, for example, ParallelAnd, ParallelOr.
4. It has the synchronizing capability over text and audio.
5. Prevailing power over animation script language: It takes simple animation script language to present animation function.
6. Dominant interactive capability: Each cast component is capable of interaction, and can respond to relative scenario. It has great flexibility.
7. Providing controls e-book normally needs such as Loop, Branch, Appear, Disappear, Delay.

Following is the brief introduction of electronic book script language. The structure of e-book script file is mapped as Fig. 6 (the syntax of EBSL please refer to [6]).

4.1 Electronic Book
\[\text{Electronic_Book} ::= \text{Scene} \mid \text{Scene}\]

An e-book is composed of many Scenes. Each scene is an independent unit. The playing of an e-book is an inter-playing between one scene to another.

4.2 Scene
\[\text{Scene} ::= [\text{SCENE} : <number> ] \]
\[\text{Effect_Segment} \text{Cast_List_Segment} \text{Scenario_Segment}\]

Each scene is composed of 4 parts:
1. Scene number: [SCENE : <number> ]
2. Scene special effect segment: <Effect_Segment>
3. Cast List script segment: <Cast_List_Segment>
4. Scenario script segment: <Scenario_Segment>
The functions of each part is explained as follows:

1. Scene number: Identifying each scene.
2. Scene special effect segment: Defining the background picture, background music, prelude(for example the background picture is spreading from left to right), and finale(for example the background picture is closing from 2 sides). The definition format is as follows:

\[\text{[EFFECT]}\]
BEGIN
BACKGROUNDBMP=bmp_file_name
BACKGROUNDAUDIO=audio_file_name
SCENEENTER=mode block delay
SCENELEAVE=mode block delay
TOTALCAST=cast_number
END

3. Cast List script segment: Defining the type and action data of each cast, for example, the location and speed of action. Different types of cast have different characteristics. There are 8 types of casts that have been defined under current system: Static_Cast, Image_Cast, Audio_Cast, Text_Cast, Avi_Cast, Animation_Cast, MoveAnimation_Cast, and Button_Cast. We use the definition of Animation Cast to explain the concept. Please refer to [6] for others.

ANIMATION CAST: Animation Cast is a cast component composed of animation multimedia data, responding audio multimedia, and control data when playing. The definition format is as below.

Fig. 6 The Framework of Electronic Book Script File
[CAST] ANIMATION
BEGIN
   ALIAS=alias_name
   TYPE=3
   ID=id_number
   SIZERGN=cast_position and cast_size
   ACTIVERGN=active_region
   NOTIFYPARENT=boolean
   CURSORID=cursor_id
   TRANSPARENT=boolean
   BGCOLOR=color_number
   STARTVISIBLE=boolean
   BMPFILE=bmp_file_name
   WAVFILE=audio_file_name
   ACTIONSPED=action_speed
   DELAYTIMES=delay_time
   LOOP=boolean
END

(4). Scenario script segment: Describing all scenarios within each scene, which is composed of 3 parts: <Prelude_Scenario>, <Finale_Scenario>, and <Interactive_Scenario>. The definition format is as below.

[PRELUDE]
BEGIN { Scenario_Block }END

[INTERACTIVE]
BEGIN
   ANCHOR 1:
      BEGIN { Scenario_Block }END
   ANCHOR 2:
      BEGIN { Scenario_Block }END
   
   END

[FINALE]
BEGIN { Scenario_Block }END

The bracketed part of BEGIN{ Scenario_Block }END is a Scenario Block, which is the genuine scenario was written. The Scenario_Block following each “ANCHOR number :” is the responding scenario when interaction with the reader is taking place. Each cast component can define a responding Scenario Block. The Statements provided by current system are as below.

(1) Single_Statement
(2) And_Statement
(3) Or_Statement
(4) Loop_Statement
(5) Appear_Statement

(6) Disappear_Statement
(7) SoundOn_Statement
(8) SoundOff_Statement
(9) Stop_Statement

< And_Statement > ::= AND (Actor_List )

The description of AND refers to several cast components are acting at the same time. The next description will not take place until all cast components have finished acting.

< Or_Statement > ::= OR (Actor_List )

The description of AND refers to several cast components are acting at the same time. The next description will continue if one cast component has finished acting.

< Loop_Statement > ::= LOOP ( < Actor > , < number > )

The description of LOOP refers to repeated action is specified to a certain cast component, that repeated action will be followed; “number” specifies the frequency of repetition. If “-1” is specified, it means repeat continuously.

(Example) The walking action of a man is a repeated movement, which is described as follows:

   LOOP ( man_walk , 5)

< Appear_Statement > ::= APPEAR (Castname_List )
< Disappear_Statement > ::= DISAPPEAR (Castname_List )

The description of APPEAR and DISAPPEAR refer to the appearance (or disappearance) of a certain cast component on the stage. These 2 descriptions are needed when a certain event happens or ends during the program.

(Example)
   APPEAR( bird , dog )
   DISAPPEAR (turtoise , bear )

4.3 Actor

< Actor > ::= < Castname > | DELAY < number > < Castname > | TIMES < number > < Castname > | DELAY < number > TIMES < number > < Castname >
Cast component is the basic control unit of EBSL. Each cast can be specified to repeat the action (TIMES < number >) or to delay for a certain time before action (DELAY < number >).

(Example) DELAY 1000 TIMES 5 man_walk
This means the cast component man_walk delays for 1 second (1000ms) before acting, and will repeat this action for 5 times.

5 EBPS

We have implemented an EBPS on the Windows platform based on EBSL. The main function of EBPS is to play e-book program according to the content of e-book scenario script file and the relevant multimedia data. The characteristics are as follows:

(1). EBPS is the Interpreter of EB Script file: EBPS is responsible for translating EB Script file into play order of e-book.
(2). EBPS is a binder binding multimedia data to cast component.
(3). EBPS is a cast component controller, which controls the synchronization between cast components, which means the acting behavior during the progress of the story[9].
(4). The cast component is the basic control unit. Detailed media controls such as synchronization of animation and sound, text and audio, playing and stopping of the video are all done by the internal cast component control.

6 Conclusion and Future Works

6.1 Conclusion

The system this study has developed, rather on investing enormous time on understanding software component (cast component), but to manage the syntax of EBSL. Once the syntax of EBSL is grasped, cast component that is corresponding to EBDS cast component interface can directly be applied without writing program code to control the component. Hence, with the multimedia data at hand, developing an e-book can be very accessible and convenient. For this reason, the time saved from writing program code can be utilized to fasten the speed to cultivate new software.

From the above mentioned, adopting the concept of software reuse to develop software component corresponding to the environment interface, and further incorporating it into new software component will make possible producing fast and economic software products.

6.2 Future Works

Based on the current e-book development system, we recommend the following directions for further exploration:

1. Strengthening and simplifying the EBSL syntax
2. Expanding different types of software component
3. Visualized supplementary authoring tools
4. Management of multimedia data

References: