國立交通大學

多媒體工程研究所

碩士論文

遊戲式教材的可重用劇情之視覺化編輯器之設計與實作

The Design and Implementation of Visual Authoring Tool for Reusable Scenario Applied to Game-Based Courseware

研 究 生：陳品宏

指導教授：陳登吉　教授

中華民國 一百零一年 七月

遊戲式教材的可重用劇情之視覺化編輯器之設計與實作

The Design and Implementation of Visual Authoring Tool for Reusable Scenario Applied to Game-Based Courseware

研 究 生 ： 陳品宏　　Student : Pin-Hung Chen

指導教授 ： 陳登吉　　Advisor : Deng-Jyi Chen

國立交通大學

多媒體工程研究所

A Thesis

Submitted to Institute of Multimedia Engineering

College of Computer Science

National Chiao Tung University

In partial Fulfillment of the Requirements

For the Degree of

Master

in

Computer Science

July 2012

Hsinchu, Taiwan, Republic of China

中華民國一百零一年七月

遊戲式教材的可重用劇情之視覺化編輯器之設計與實作

研 究 生 ： 陳品宏　　指導教授 ： 陳登吉

國立交通大學多媒體工程研究所碩士班

摘要

隨著科技的進步，數位學習日漸蓬勃發展，一般傳統的紙本教材，已經不太能夠滿足學習者的需求。智慧型手機與平板電腦的發達，也使得越來越多學習者將學習的平台從紙本上轉往智慧型手機或是平板電腦。由文字、聲音、影像與圖片等組合而成的多媒體教材變得日趨流行。遊戲式的教材也屬於多媒體教材的一種，藉由遊戲的特性讓學習者在學習的過程獲得更多的學習動機與更好的學習效率，所以遊戲式的教學已經慢慢的倍受注目。

雖然遊戲式教學有更好的學習效果，但是在編輯的過程中，遊戲式教材卻是較困難於一般傳統紙本教材的編輯。遊戲式教材提供了良好的互動、絢爛的聲音與華麗影像的效果，藉由這些特色去提高學習者學習動機，但是就是因為這些迷人的特色讓遊戲式教材的編輯困難。編輯一遊戲式教材編輯者首先須克服寫程式的困難、整合多種媒體的困難與演員的穿插互動等問題。所以提供一視覺化的編輯環境讓編輯者可以在此環境用視覺化語言編輯遊戲式教材劇情可以使得遊戲式教材的編輯更有效率。

關鍵字 ： 遊戲式教材、遊戲式教學、視覺化語言、視覺化劇情 、劇情重用、視覺化編輯器

The Design and Implementation of Visual Authoring Tool for Reusable Scenario Applied to Game-Based Courseware

Student：Pin-Hung Chen **Advisor：Deng-Jyi Chen**

Department of Multimedia Engineering National Chiao Tung University

Abstract

With the progress of science and technology, e-learning is developed rapidly. Learners can't be satisfied with using traditional textual teaching materials only. More and more learners had transferred their learning platform from the paper to the smart phone or tablet PC. With the composition of text, image, audio and video, multimedia's teaching materials become more and more popular. Game-based teaching materials are a kind of multimedia teaching materials. Learners will have higher learning motivations and better learning efficiency with the characteristics of the game. Game-based teaching already gets much attention gradually.

Although game-based teaching materials have more learning efficiency, it is harder to be edited than that of traditional textual teaching material. Game-based teaching materials provide a good interaction, the effect of the gorgeous voice and the attraction of images, using these features to improve learners' learning motivations. But because of using these features, game-based teaching materials are more difficult to be edited. When editors edit the game-based teaching materials, they have to overcome the difficulties of writing program, integrating multimedia and interspersing interactive actors. Therefore, providing a visual programming environment makes editors get more efficiency when edit the game-based teaching materials by using visual programming language in this environment.

Keywords : game-based teaching material, game-based teaching, visual programming language, visual scenario, reusable component, visual programming editor

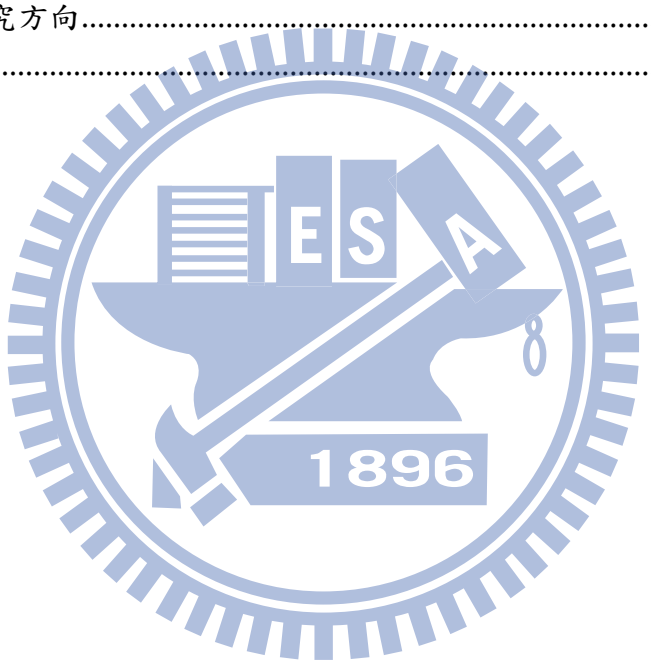
誌謝

感謝指導教授 陳登吉老師這兩年來的細心指導與教誨，不論是在研究上或是生活上都給了很大的鼓勵與指引。老師時常會在課餘之時分享他所領悟到的生活經驗讓我們學習到課本所學習不到的東西，使我的兩年研究所生活過得相當充實。感謝孔崇旭老師在畢業的最緊要關頭不厭其煩的從台中到新竹指導我的論文，感謝有兩位老師的幫助，讓我可以順利完成我的論文。老師，謝謝您。

此外，我要感謝一路上在交大陪伴我的同學、學長姊、學弟妹們，在研究的期間給予我許多課業上與生活上的幫助，很高興能夠認識這些優秀的朋友，讓我這兩年的研究生活過得更多采多姿。

最後，感謝我的家人，一路上在我背後默默的支持與鼓勵，使我可以專心於我的論文無後顧之憂，謝謝。

目錄



表目錄

圖目錄

一、緒論

1.1 研究背景

1.1.1 遊戲式學習

隨著數位科技的進步，傳統的紙本教學已經不足以滿足眾多學生與老師的需求。多媒體教材結合了眾多的媒體(聲音、影像、圖片、文字等)，由於這些媒體的組成，使得教學變得更加生動有趣。遊戲式的教材就是其中一種多媒體教材，遊戲式教材類似於問題式學習（Problem Based Learning，PBL），將特定的問題腳本放入一個遊戲的架構裡[1]。所謂的遊戲就是定義一套有規則可循，具有可以達到的目標或成果的系統，學習者可藉由此系統在遊戲中練習、互動得到學習的經驗，達到學習的目的[2]。遊戲式的教材也可稱為嚴肅遊戲（serious games），它被用來泛指所有不純粹以娛樂為目的之電腦遊戲，且通常指用在教育、訓練或模擬上的電腦遊戲[3]。遊戲式學習能讓學生不自覺地將時間投入在玩遊戲中，達到學習的效果[4]。且藉由遊戲本身具備的”Fun”與”Pleasure”使使用者加強學習意願[5]。傳統的教材較令人乏味，但遊戲卻可以引起學習動機，提供讓人專注與沈浸的經歷[6]。線上遊戲可讓人主動、合作、對話、反思等特點，而此正符合建構式網路學習環境的特點[7]。

遊戲式的學習可結合多種硬體例如手機、任天堂開發的掌上型遊戲機(NDSL)等，東京一所高中的老師用 NDSL 的英語學習軟體與課本，讓學生學習拼音，聽力等英語學習能力(東京路透社)。手機上面的遊戲式學習不僅僅可以較有趣、互動、較有彈性且更能使使用者更能適應不同類型的學習方式[8]。Bottino、Ferlino、Ott 和 Tavella 實驗發現長時間接觸邏輯遊戲，對兒童推理能力有正向影響[9]。Kebritchi, Hirumi, Bai 指出遊戲式教學對國三學生數學方面能提升學習成效與動機[10]。透過遊戲的方式可以增進學生的學習成效，其中有研究發現有玩手術遊戲的外科醫生與沒有的技術增強了(Courant , 2008)。由以上眾多研究結果顯示遊戲式的學習可以達到主動學習、提高學習興趣與成效，因此遊戲式學習在眾多學習方法中佔有更大的優勢以及學習成效。

1.1.2 劇情的描述方式

在上一個章節提到了遊戲式教材的特性與其重要性，在本章節中本研究會根據遊戲式教材的劇情做探討。製作一份遊戲式的教材最重要的就是其劇情，劇情包含了教材中演員的動作、場景的安排與關卡的難度等，所以編輯者能不能呈現完整的劇情現大大的影響了遊戲式教材的品質好壞。劇情表達的方式多樣，用文字、影片、聲音、圖片或表格皆可以表達一段或多段的劇情，以下再深入探討近幾年較常用的劇情表達方式，分別是一般傳統程式語言式、Multimedia、UML Modeling、Video 與 Audio。

1. 一般傳統程式語言

   用寫程式的方法來描述一段劇情，編輯者需要具備程式相關背景才有辦法用此方式描述，一般常見的 C++、Java 都是屬於文字式的劇情編輯器。

2. Video

   將使用者劇情需求以 video 方式呈現，所需的工具在硬體方面需要 V8 或其他可錄影之器材；軟體方面，則需要將拍攝後的影像利用影像編輯軟體後製編排。Jacob Buur 指出了以 Video Spec 來幫助開發過程，建立明確的限制與解決方案[11]。將使用者劇情需求拍成影片，並加以編排。影片就像電影一樣有劇情。圖 1 為一個以 video 呈現劇情需求的範例，將劇情需求以 video 方式呈現並且分類成 3 種不同劇情需求層級，讓開發者可以明確的了解劇情需求的內容與優先程度。

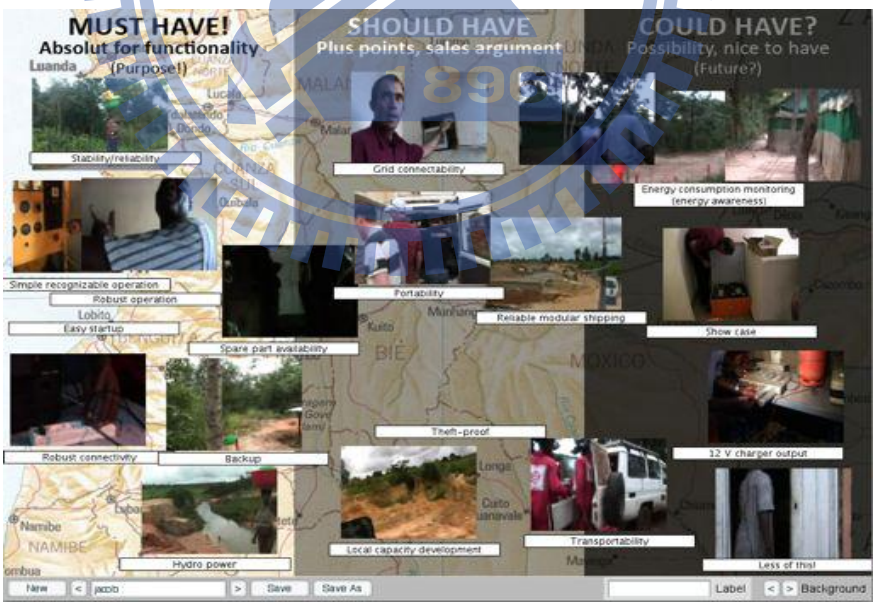


圖 1- Video Wall activity: Video clips are labeled and arranged as requirements according to their importance

3. Multimedia

以定義有意義的 icon 圖示(semantics of icons)來代表物件如圖 2，用線條來代表物件之間的關連[12]。使用者利用 icons 與建立關聯來完成需求的流程編輯，最後再以動畫方式呈現編輯結果如圖 3

Icon shape	semantics
	human type
	"Fax," device type
	"a message," data type

圖 2 Icons Definition

圖 3 Visual SRS

4. Audio

將劇情以口述的方式描述，並將此錄音。在播放的過程中劇情會以說故事的形式呈現。

5. UML Modeling

在業界利用”統一塑模語言”(UML)，”模型驅動開發”(MDD)與 ”模型驅動架構” (MDA)開發方式漸漸興起[13]。UML 中的”循序圖” (Sequence Diagram)或是”活動圖” (Activity Diagram)用來描述流程、”狀態圖” (State Diagram)用來描述流程中特定物件狀態的轉移與變化過程、”類別圖”(Class Diagram)用來描述物件與物件之間的關係。以下是使用 UML Modeling Tool 的一個範例。利用 UML 相關圖形的特性，建立一個流程(ReVU)，將”功能需求”(Functional Requirements)用視覺化方式 UML modeling 方式來呈現[13]。將使用者需求以 UML Modeling 方式呈現，所需工具需要 UML Modeling Tools 來方便編輯如圖 4 所示。現行已有許多 freeware 的 UML Modeling Tools 如” StarUML”、”JUDE”…等。

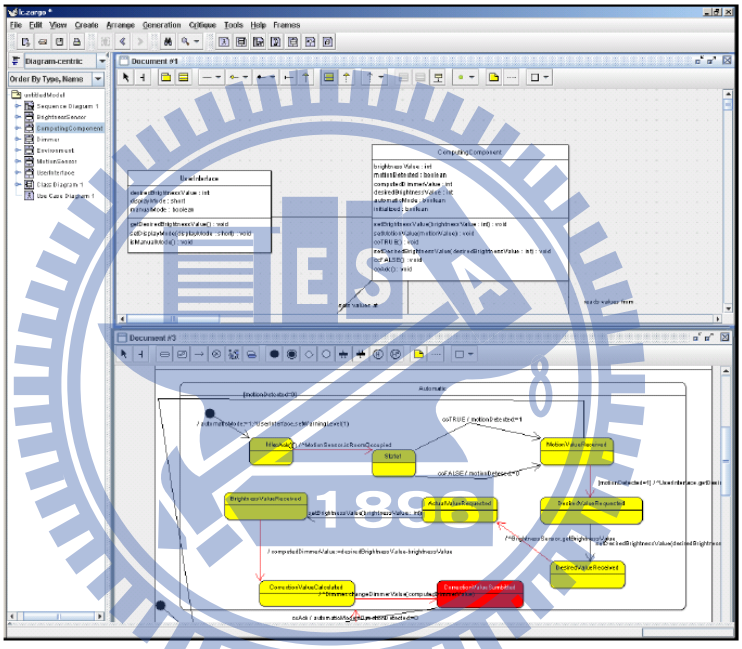


圖 4 UML Modeling

1.1.3 視覺化劇情的優點與缺點

由上一節的分析可以了解劇情可由多種方式來表達，但不管是由何種方式來表達劇情，這幾種表達方式是等價的[14]，但是根據研究的結果顯示，對於較複雜的事物，人們總是習慣用圖表或是動畫的方式，使事情變得較容易理解[15]。對於教材編輯器來說，由於軟體具有不可見性，編輯教材過程中編輯者是無法了解編輯過程是否出了錯誤，所以如何將編輯的過程視覺化，讓編輯者做視覺化的即時修改劇情就變得非常重要。以下探討了視覺化劇情表達的優點[14]。

1. 不具程式相關背景的編輯者也可輕易上手
2. 更接近自然的需求表達
3. 較容易了解需求內容
4. 對於軟體使用者及系統開發者來說是較友善的介面
5. 對使用者來說是較易於了解及易於溝通的方式

因為用上述的方式來表達劇情都是等價的，而視覺化的表達方式又具備了以上幾點優點，所以本研究接下來的研究內容將以視覺化的編輯方式為基礎去加以延伸。

1.2 研究動機

雖然遊戲式的教材較能提高學習者的學習動機與學習效率，但是編輯一遊戲式的教材卻是較一般傳統的教材要困難上很多。編輯一遊戲式教材的方式有兩種，一是用一般程式語言編輯，例如 C 或 Java 等語言，但是如果用一般程式語言的設計環境對於沒有程式相關背景的編輯者來說並不是那麼的友善，所以本研究提出了用第二種方法，也就是視覺化程式語言(Visual Programming Language)的方法編輯。用視覺化程式語言的方式編輯，編輯者無須具備程式相關背景即可以完成視覺化劇情的編輯，但是目前用視覺化程式語言的方式編輯的話，編輯者還須克服以下幾點困難處:

1. 編輯軟體底層架構複雜，沒有程式相關背景的的編輯者無法使用[16]:
   編輯軟體所提供的功能有限，並不一定能滿足所有編輯者的需求，當需求大於軟體所提供的功能時，編輯者則須想辦法去修改編輯軟體底層的語言才有辦法達到需求。
2. 編輯者不容易將劇情套入遊戲式的架構[17, 18]:
   沒有對應的指令將編輯者的創意直接轉成遊戲式的表達。
3. 遊戲式教材架構較複雜:
   遊戲式教材的的設計分成構想教材與實作教材兩部分，構想教材的學者沒有辦法獨力完成教材的實作，需藉由程式設計師幫忙。
4. 太過複雜的視覺化劇情設計困難[19]:
   對於遊戲式教材中每個演員、每個劇情與每個場景等，都需要有其對應的定義，當演員或劇情太過於複雜，則在編輯或者修改的過程中會非常困難。
5. 沒有統一的視覺化劇情編輯語法[16]:
   每個研究所定義的視覺化語法階不一定相同，編輯者不一定能夠馬上熟悉各個系統所定義的語法來編輯。
6. 大型的遊戲式教材沒有重用劇情元件套用，編輯困難:
   在視覺化程式語言當中，視覺化劇情重用的功能開發尚未成熟，所以當遇到重複性質高的劇情，可能無法直接呼叫重用的劇情元件直接套用。必須重頭編輯在編輯的效率較低落。

根據以上六點的探討，可以了解用視覺化程式語言的編輯的確是還有一些的不足尚須改進，所以接下來本研究會根據以上探討的幾點不足提出幾點改進方法，希望本研究所提出的方法可以改進目前視覺化程式語言在編輯劇情上面的不足之處。

1.3 研究目的

因為視覺化劇情的重用在視覺化程式語言是一個重要的議題，藉由重用元件的套用可以加快的編輯劇情使編輯更有效率，本研究先探討了目前已知的視覺化程式語言編輯器，找到了幾點關於重用上面稍為不足的部分，並設計一視覺化程式語言編輯器來輔助在劇情重用上面的不足。設計視覺化程式語言編輯器可分為四個步驟:

1. 定義視覺化劇情的語句
   編輯者可以用視覺化的方式來編輯劇情
   編輯者可以根據定義的語法與句子完成編輯
   編輯者不必藉由程式設計師幫助即可完成編輯
2. 定義重用劇情的元件以及套用的方式
   編輯者可將編輯完成的劇情以 Reusable Scenario Component 的方式儲存
   遇到重複性高的劇情時可直接利用 Reusable Scenario Component 加速編輯
3. 開發一個可編輯重用劇情的編輯器
   利用此編輯器中，編輯者可以編輯視覺化劇情，並可以對此劇情作修改、新增、刪除、儲存等功能，如果有重用的視覺化元件可以套用時，也可以在編輯的過程中直接呼叫套用。
4. 開發一個可預覽/重播劇情的播放器
   此播放器為將編輯器編輯好的劇情作為預覽與播放之功能，當編輯的視覺化劇情不滿意時，可以跳回編輯器，再加以對劇情作新增、修改、刪除等功能。

1.4 章節概要

本研究共分為五大章節，以下簡述個章節之內容：

第一章：緒論。敘述本研究的研究背景和動機，並根據研究動機延伸出研究目標。

第二章：文獻探討與相關研究。探討與本研究相關文獻，例如市面上較知名的幾個視覺化程式語言編輯器如 Scratch、Google Blockly 等，並分析這些編輯器在重用機制上面是如何套用。

第三章：系統分析與設計。分析及設計一具有重用編輯功能之視覺化程式語言編輯器。詳細定義在此系統中所使用的語法、重用元件的儲存以及套用方式。並定義一個以 XML 為基礎的重用元件等等相關內容。

第四章：系統設計與實作。依據第三章的系統分析與設計，使用 C Sharp 語言實作出一個具有劇情重用功能的視覺化程式語言編輯器，並剖析其設計元件組成方式。包括了 User interface 與 Scenario editor 等相關元件。
搭配在第三章定義的 XML 重用元件達到視覺化劇情重用的功能。在此章節會實際編輯一多媒體劇情，在編輯的過程中，使用者可以利用自行編輯的重用元件來

達到更有效率的編輯。

第五章：實作範例。實作一多媒體劇情。在編輯的過程中，搭配定義的 XML 重用元件，讓編輯者在編輯劇情的過程中，可以套用重用的元件來達到更有效率的編輯。

第六章：結論與未來研究方向。說明本研究的貢獻，以及未來的展望，對於本研究的系統分析其不足與未來能修改的地方。

二、文獻探討

在此章節中本研究會對目前市面或學術上較知名的幾個視覺化編輯器作進一步的分析。

2.1 視覺化的軟體需求樣板

一份視覺化軟體需求如圖 5 所示，如同一部電影，由四種主要元素所組成:場景(Scene)，場景佈局(Layout)，演員(Actor)，以及劇情(Scenario)。其中演員為最基本的重用單位 MRC(Multimedia Reusable Component)。

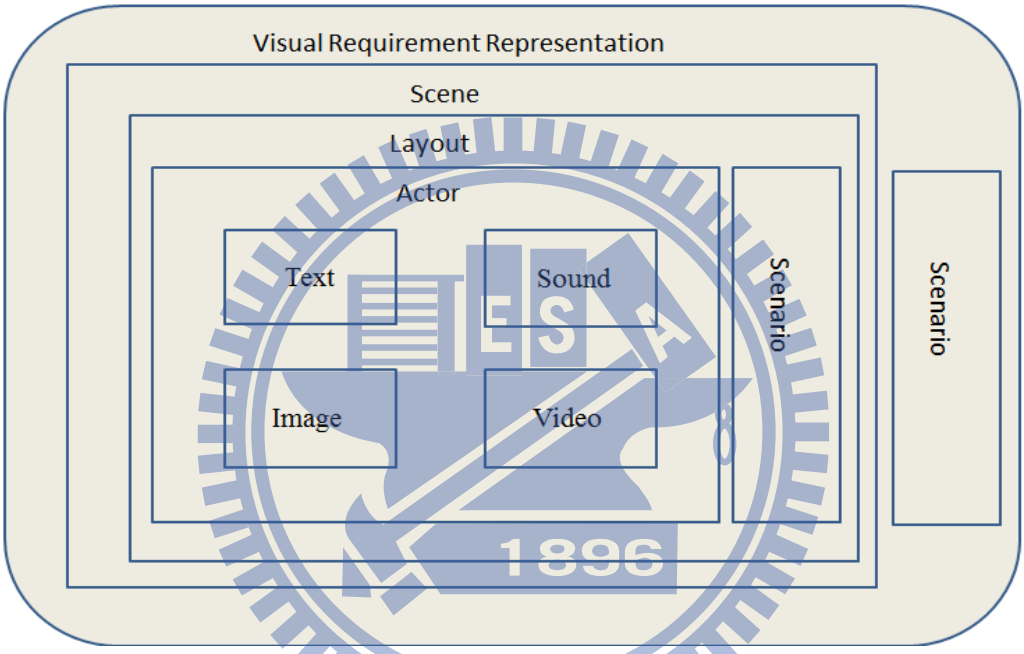


圖 5 視覺化軟體需求結構圖

在視覺化軟體需求樣板[20]中，重用的單位從最基本的演員提升到了整個視覺化軟體需求，此方法將複雜的需求架構以及實作內容封裝起來，只留下置換及設定參數的簡單介面供編輯者修改。這樣的重用元件稱為視覺化軟體需求樣板。因此，就算編輯者本身不具程式或軟體等相關背景，也能藉由不同的參數輸入達到製作自己需要的視覺化軟體需求。如圖 6 所示，藉由套用不同的資料到樣板中，產生不同的視覺化軟體需求。圖 6 為一多媒體試題的樣板，根據輸入的參數不同，可以產生不同的題目與不同的選項以供搭配。

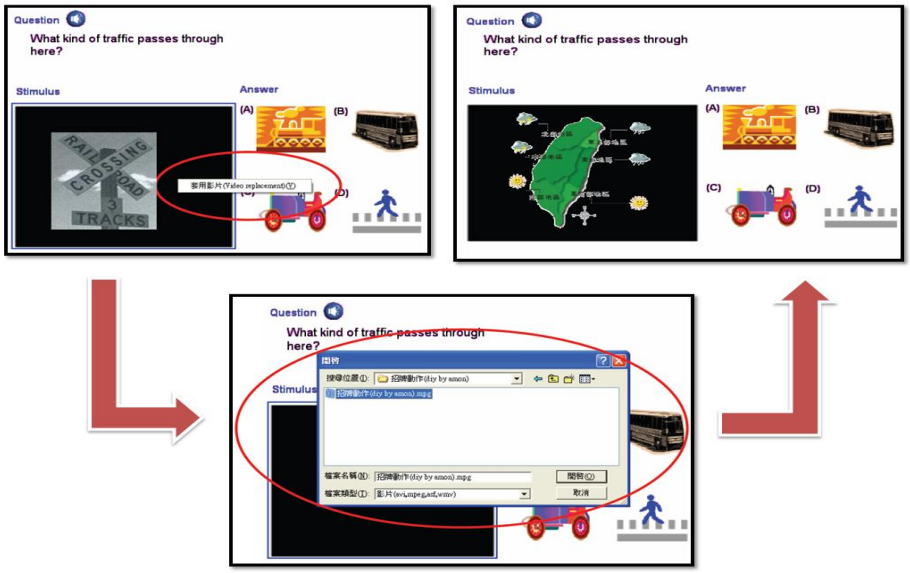


圖 6 多媒體試題套用劇情描述

2.2 Scratch

Scratch[21]是一套視覺化程式編輯軟體，適合作為學習程式設計的入門軟體，它可以讓編輯者輕鬆的規劃程式設計的劇情、動畫、遊戲、音樂等，並且可以透過網路上傳功能，將編輯者的作品分享至全世界。Scratch 是由美國麻省理工學院媒體實驗室(MIT Media Lab)的終身幼稚園團隊(Lifelong Kindergarten Group)所開發的一種新的程式語言，讓編輯者可以輕易的創造自己的數位作品，編輯者還可以透過 Scratch 官方網站[21] 與人分享編輯者的創作。圖 7 為 Scratch 的編輯畫面。

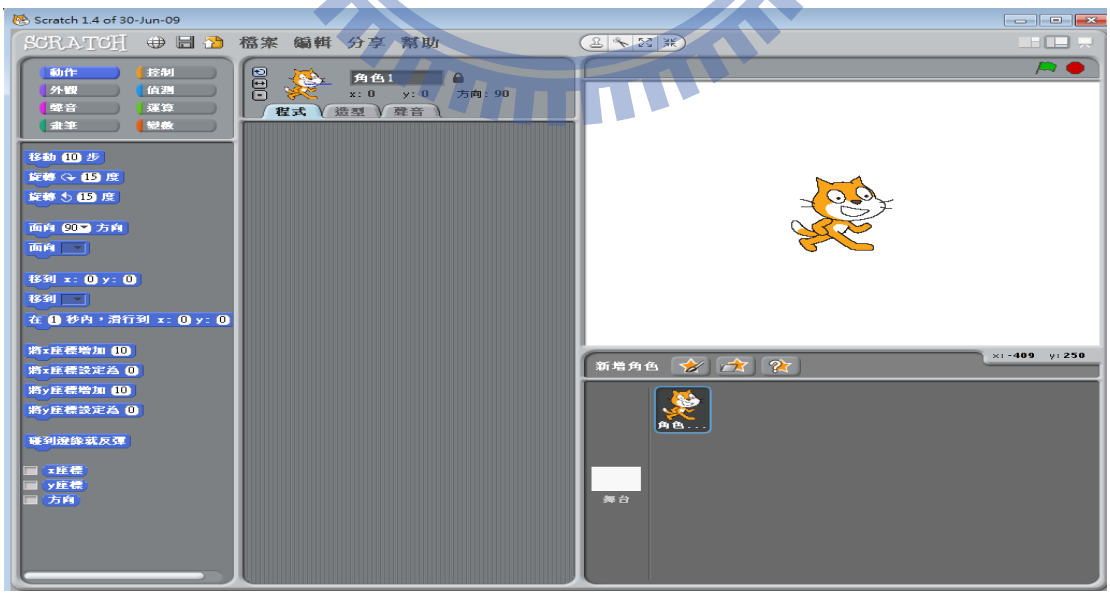


圖 7 Scratch 編輯畫面

Scratch 的編輯方式為組合不同的積木程式碼來創造出不同的視覺化劇情，編輯者在編輯過程完全不需要寫程式，只需要拖拉圖 7 左邊的積木程式即可完成編輯。Scratch 所支援的積木程式包括移動、旋轉、碰撞、設定變數、輸入聲音等功能。在視覺化編輯的功能非常全面。而在重用元件這方面，Scratch 也有這方面的功能，從最基本的演員重用到多個演員的場景重用皆可以儲存，如圖 8 所示。Scratch 所儲存的檔案為.sb 檔(Scratch Project)，要編輯或修改此檔案時，必須要開啟 Scratch 主程式才可修改，對於儲存檔案修改的彈性來講，不若 XML 檔案來的有彈性。

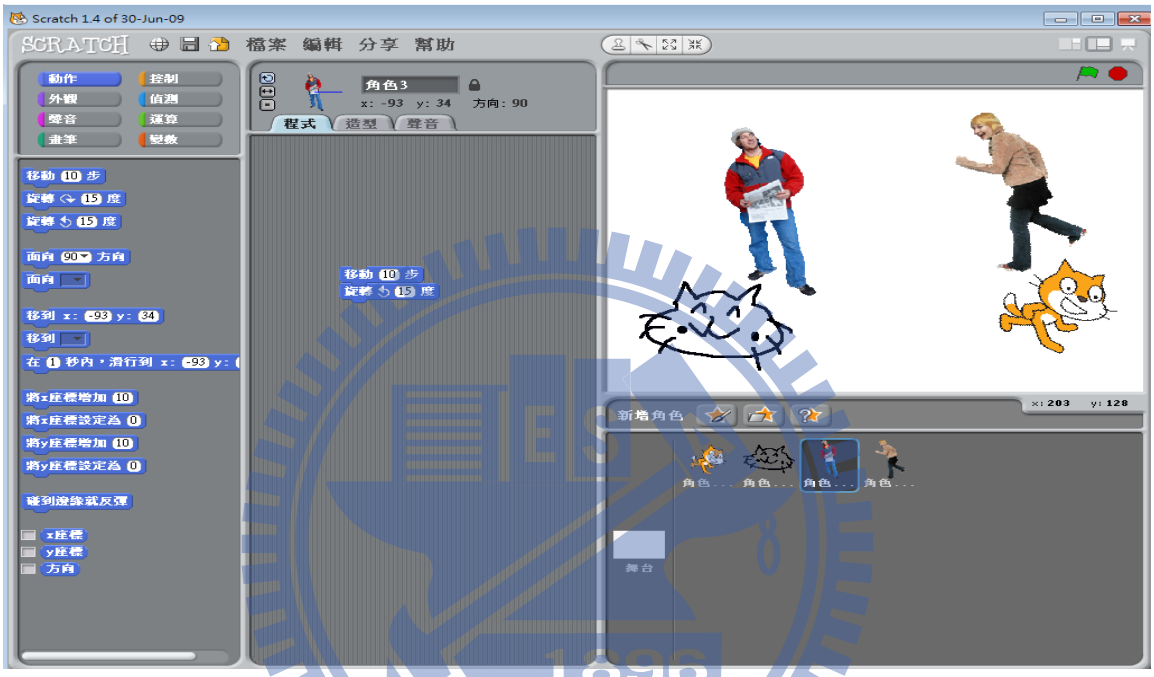


圖 8 Scratch 多個演員之場景儲存

2.3 Google-Blockly

Google-Blockly[22]為"Scratch-like"視覺化程式語言，Blockly 是以 HTML5，CSS3 和 Javascript 打造而成的，視覺化可拖拉拼貼式程式積木系統。目前 Google-Blockly 提供了幾個 Demo，第一種如圖 9 所示，為一個迷宮的程式，使用者必須用 Google-Blockly 所提供的拼圖式語法，將動作托拉到編輯區目標為脫離迷宮。圖 9 的右邊為本研究利用 Commands 一步一步將小人偶走到目標地。Google-Blockly 也提供了使用 Logic 的方式，Logic 的方式可以使用 While、If、Repeat 等條件指令，來加快完成編輯，如圖 10 所示。

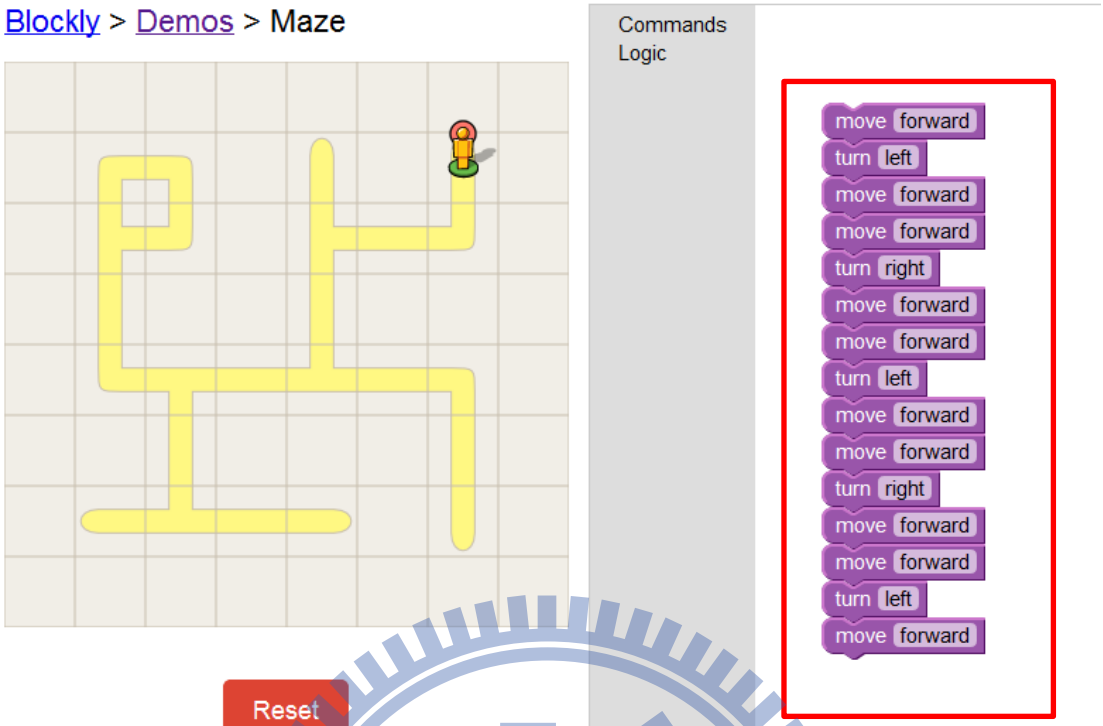


圖 9 Google-Blockly Demo1 法一

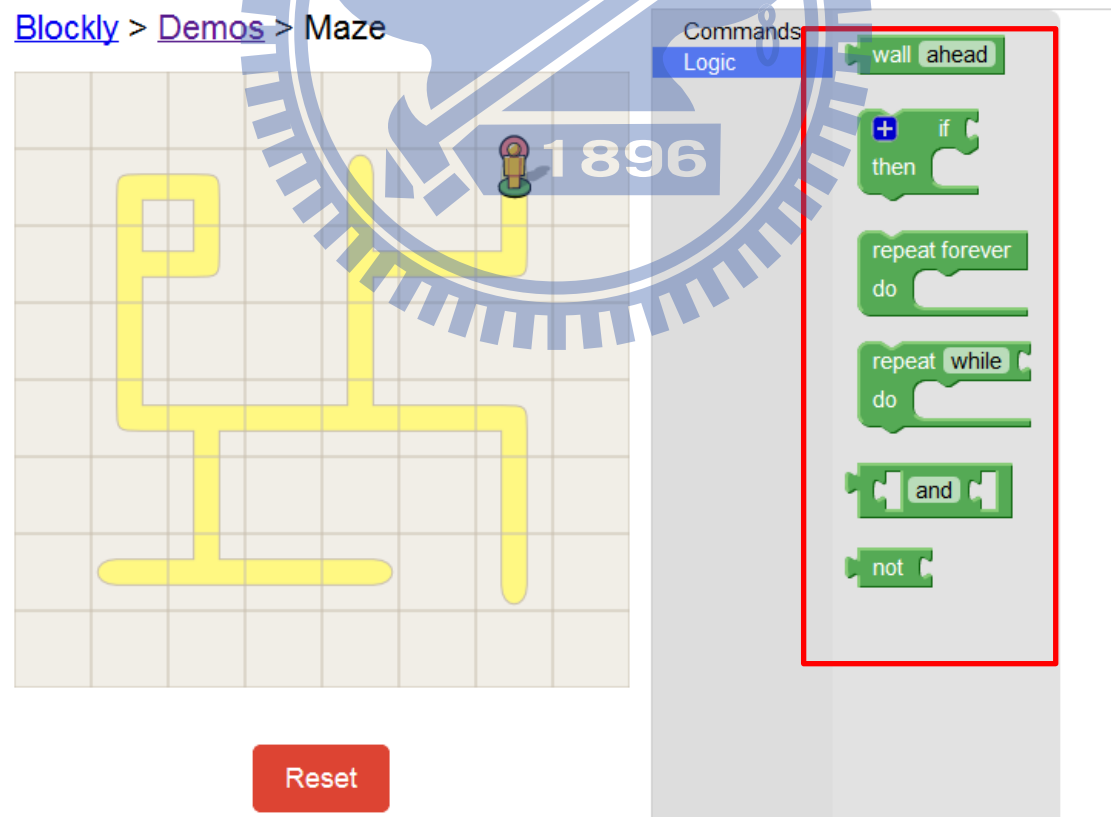


圖 10 Google-Blockly Demo1 法二

第二個 Demo，如圖 11 所示，編輯的積木程式是由程式語言的文字所描述而成，這樣的編輯方法可以避免掉一般撰寫程式語言中語法上面的錯誤，適合沒有程式背景的人初學使用。

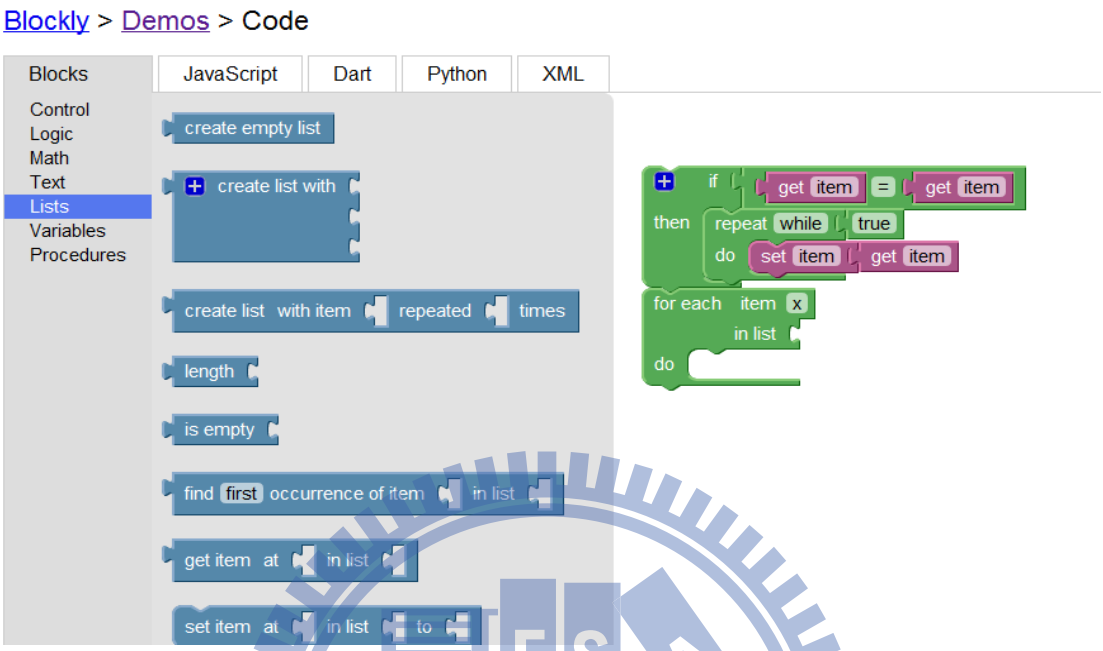


圖 11 Google-Blockly Demo2

雖然 Google-Blockly 也是在網頁上進行操作與編輯，但是 Google-Blockly 可以將編輯好的成果以 XML 的方式儲存到本機端，所以當下次遇到類似的視覺化時，可以直接將本機端已經儲存好的 XML 檔案呼叫出來。因為是 XML 檔案儲存，所以使用者不必藉由 Google-Blockly 才能開啟已經儲存的檔案，檔案可以直接開啟增進修改與保存的彈性。

Blockly > Demos > Code

Blocks | JavaScript | Dart | Python | XML

```
<xml>
  <block type="controls_if" inline="false" x="350" y="31">
    <value name="IF0">
      <block type="logic_compare" inline="true">
        <title name="OP">EQ</title>
        <value name="A">
          <block type="variables_get">
            <title name="VAR">item</title>
          </block>
        </value>
        <value name="B">
          <block type="variables_get">
            <title name="VAR">item</title>
          </block>
        </value>
      </block>
    </value>
    <statement name="DO0">
      <block type="controls_whileUntil" inline="false">
        <title name="MODE">WHILE</title>
        <value name="BOOL">
          <block type="logic_boolean">
            <title name="BOOL">TRUE</title>
          </block>
        </value>
        <statement name="DO">
          <block type="variables_set" inline="false">
            <title name="VAR">item</title>
            <value name="VALUE">
              <block type="variables_get">
                <title name="VAR">item</title>
              </block>
            </value>
          </block>
        </statement>
      </block>
    </statement>
    <next>
      <block type="controls_forEach" inline="false">
        <variable data="x" name="VAR"></variable>
```

圖 12 Google-Blockly Demo2(轉成 XML 格式)

圖 12 為將圖 11 描述的視覺化 Block 語言轉成 XML 的表示方法，Google-Blockly 也還提供了將視覺化 Block 語言轉成 JavaScript、Dart、Python 的語法。

2.4 ScenEdit

ScenEdit[23]為一視覺話的編輯工具，他提供了讓編輯者以視覺話的方式編輯教學環境，為一以 ISiS(Intensions-Strategies-interactional Situation)為基礎的 goal-oriented framework，它定義了四個 workspaces:

1. Context workspace
   定義學習物件的關聯，當學習物件要被呼叫執行時。
2. Components workspace
   負責管理ISiS模型的組成。
3. Scenario Editor workspace
   負責建立與定義元素的工作空間
4. Exporting workspace
   允許可以以不同型態的輸出產生

圖13為ScenEdit的主編輯畫面，劇情均可由視覺化的方式編輯而成。

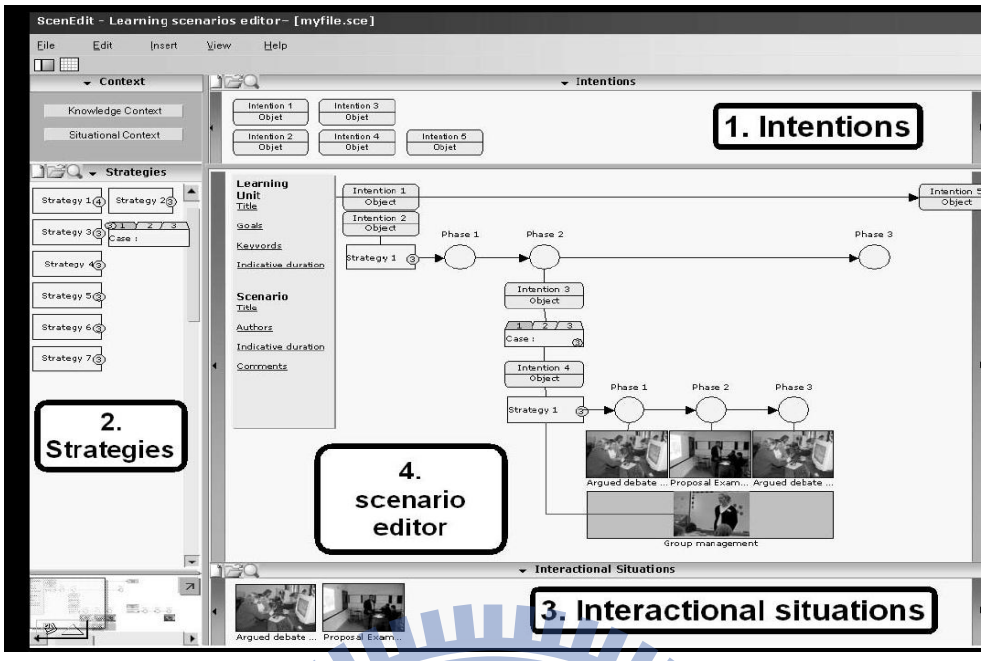


圖 13 ScenEdit 的主編輯視窗

2.5 MACHS

MACHS[19]為一個設計 Course 的遊戲，由多位學者 A. Mujika, D. Oyarzun, I. Iparragirre, 等人開發出來的一套視覺化的方式編輯嚴肅遊戲(Serious games)的系統，此系統的最主要目的為，讓使用 MACHS 系統開發嚴肅遊戲的每一個人可以獨立完成編輯，而不須程式設計師的幫忙。在 MACHS 系統中主要分成兩個功能，第一個功能為一個可以讓編輯者編輯教材的視覺化編輯器(Authoring Tool)叫做 Course Editor，第二個功能為一接受視覺化編輯器 output 的 3D 模擬環境稱 3D Simulator，以下對兩個功能作進一步介紹。

1. Course Editor

   藉由此功能，即使編輯者不是使用 3D 的編輯器，也可以達到模擬 3D Course 的效果。此功能中使用者可對編輯的 Course 作進一步的描述，例如 Name、Description of simulation、Description of exercise 等。編輯 Course 的方式如圖 14 所示，在圖 14 中可以看出來是由多個方形與連接箭頭所組成。每個方形或物件代表可以走的 Step，連接箭頭為 Step 接下來往哪個 Step 前進。

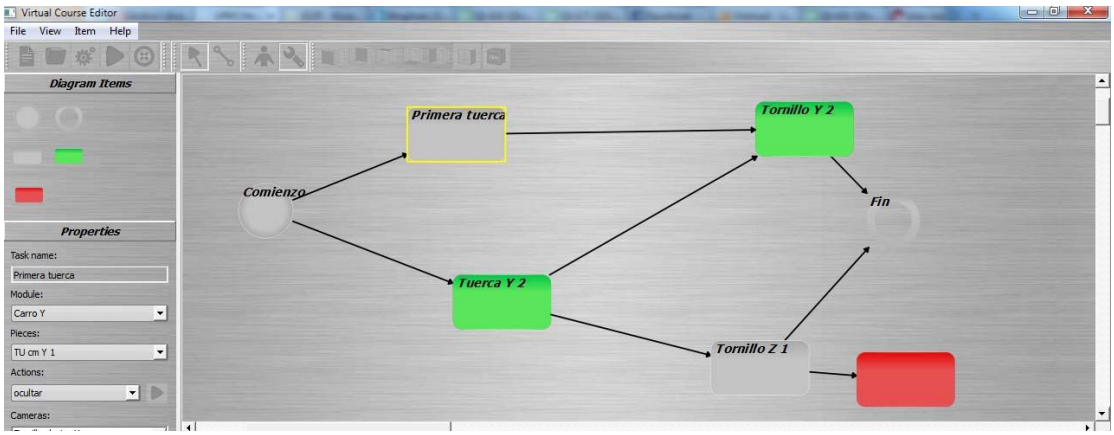


圖 14 The diagram and the machine in the Course Editor

2. 3D Simulator

3D Simulator 功能為接收 Course Editor 的 Output，並將此 Output 自動的產生 3D 環境的 Course，在 3D Simulator 中不允許使用者作任何的修改，只允許對 3D Simulator 作互動。例如使用者可以選擇系統顯示出下一 Step 為何，以加快完成遊戲。或者是使用者也可以選擇系統完全不需要提示。圖 15 為接受圖 14 的資訊所顯現的 3D 模擬。

圖 15 Simulator 可選擇的圖示

除了 Course Editor 與 3D Simulator 之外還需要有個 Editor/Simulator CONNECTION 將編輯的 Course 與要模擬的 3D 環境連接。為了達到同時性（synchronization），MACHS 使用了以 XML 為基礎的語法稱為 Virtual Course Simulation (VCS)，圖 16 為 MACHS 所定義的 VCS。

```
<vcs>
   <exercise>
      <action>
         <transition/>
         <transition/>
         …
      </action>
      …
   </exercise>
   <simulation>
      <action>
         <transition/>
      </action>
      …
   </simulation>
</vcs>
```

圖 16 VCS

2.6 SeGAE

SeGAE(Serious GAME Authoring Environment)[18]是一個 author-friendly 編輯環境，提供了教材編輯者藉由重新定義物件去作遊戲的修改。SeGAE 定義了 XSD 格式去描述腳色的特性、動作與訊息等資料。SeGAE 提供了方法把遊戲編輯者編輯的教學式物件轉換成遊戲的物件。SeGAE 的編輯架構是根據 MVC 架構而成(model-view-controller)，視覺者所看到的編輯畫面是由 MXML(XML-based user interface markup language)架構而成。SeGAE 創造了:1. 一個可以創造、儲存與載入物件的 proxy，2. 一個可透過 GUI 並利用其指令創造物件的 mediator。圖 17 為 SeGAE 的架構

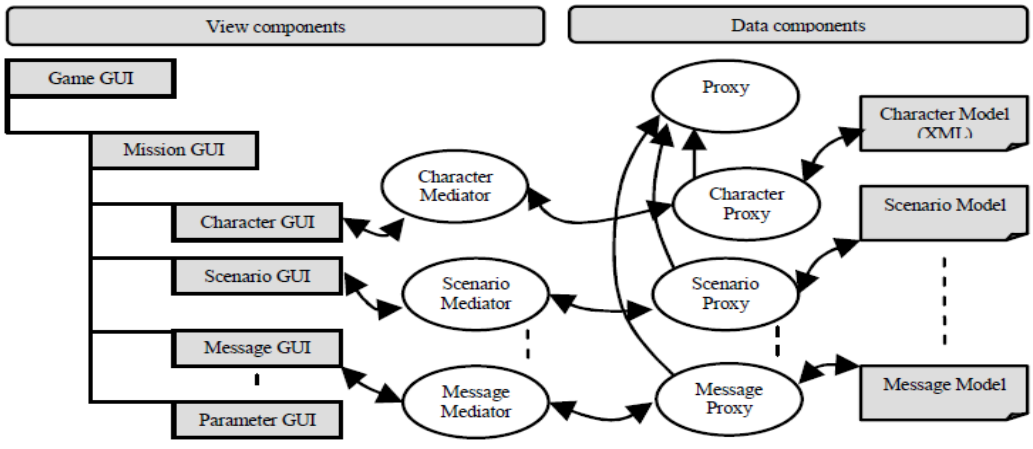


圖 17 SeGAE 架構

SeGAE 的編輯環境提供了四種視覺化的編輯:

1. Character editor
   提供用視覺化編輯演員與不同特性的關卡。
2. Temporal scenario editor
   是由連續的動作序列的有限狀態機組成，編輯者可以加入或修改任務來改變有限狀態機的組成。
3. Message and dialogue editor
   在進行遊戲任務的時候，SeGAE 可以允許加入訊息或者是對話視窗。
4. Game internationalization
   可以選擇使用多個國家的語言去加入訊息或是對話視窗。並將訊息與對話視窗的資料用 XML 檔案儲存。

不管是 Scratch、MACHS 或 SeGAE 等編輯器，它們雖然具備了豐富的劇情供編輯者編輯，但是對於視覺化的劇情重用甚少提到，所以本研究將根據重用劇情的部分作進一步的探討。

三、視覺化劇情介紹

在此章節中，主要分成四個部分，視覺化劇情的組成、重用元件的設計、提供的視覺化功能與視覺化重用元件套用四個部分，本研究主要在目標為克服目前市面上視覺化劇情重用的不足。在此章中會根據第二章所分析的結果加以延伸並設計一視覺化系統希望可以彌補市面上多個視覺化編輯軟體在重用劇情上面的不足。

3.1 視覺化劇情的組成

因為不是每個教材編輯者都具有程式相關背景，所以一般的程式語言語法對每個編輯者來說並不是那麼的熟悉。本研究系統定義了幾個視覺化的元件來幫助編輯者作視覺化的編輯，設計的元件有以下幾類:

1. 演員劇情(Actor Scenario)元件:
   演員劇情的元件記錄著編輯者對該演員設定的劇情，例如小狗演員的劇情為，上下左右移動，則該演員元件的資料即為上下左右。
2. 演員(Actor)元件:
   演員元件包含了演員劇情與設定的演員參數，演員參數包括了演員的圖片、演員移動的距離、演員旋轉的角度、演員每個動作的間隔與演員的初始位置等資訊。
3. 句子(Sentence)元件
   句子劇情包含了演員劇情元件、演員元件與句子的關聯。一個句子可由多個演員所組成，句子的關聯包括了循序的句子關聯、平行的句子關聯與互動的句子關聯。
4. 場景(Scene)元件
   場景元件為上述三個元件的組合，額外包括了背景與條件是設定的相關資訊。

3.2 重用元件的設計

視覺化語言的編輯過程中，提供一可以令編輯者儲存的格式可以讓編輯者在編輯時更有效率。在本小節中會根據 3.1 節提供的元件設計一儲存格式，本系統設計的重用元件格式是以 XML 語法撰寫，XML (Extensible Markup Language) 是一套資料儲存的規格，XML 具有以下的特徵:

1. XML 可以按照編輯者需求，以編輯者設計的格式儲存。
2. XML 為開放標準，任何編輯者皆可按照此標準設計專屬格式。
3. XML 的結構較容易讓人或者程式讀取與閱讀。

以演員重用元件為例，如圖 18 所示，演員重用元件裡儲存的資料包括有:

1. 演員的名稱
   如圖 11 所示，演員的名稱叫做小狗，圖片的名稱也叫做小狗，以供系統分辨用。
2. 圖片的路徑
   每個演員由一個 PictureBox 組成，路徑代表著存放在 PictureBox 裡面圖片的位置，位置表位於本機端的位置。
3. 演員移動距離
   演員每次移動的距離，如圖 18 所示，move_step 這個成員即是所記錄的移動距離，每次固定為 50 Pixel。
4. 演員旋轉角度
   每一個演員皆可以旋轉，例如輸入 90 的話，每次輸入旋轉按鈕，演員將會呈 90 度的旋轉。
5. 演員放大縮小比例
   系統可以提供演員做放大與縮小的動作，圖 18 的 big_step 與 small_step 皆為 20，代表放大或縮小時，長寬各增加或縮小 20。
6. 動作間格時間
   因為考慮到現在電腦運算很快的原因，每一個演員的動作皆由各自的 timer 去處理，當每一次的 timer_tick 演員才會往下去動作，timer_tick 的時間就是演員動作的間格時間。如沒有用 timer 去做運算，則演員的動作只會停留在最後的結果，不會一步一步的動作。例如圖 18 演員的動作間格時間，wait_time 為 500 毫秒。
7. 演員大小
   演員出現在場景中的大小，由使用者自行輸入，預設值為長 50 pixel 寬 50 pixel，圖 18 小狗演員的儲存資料中，演員長度為 50 pixel，演員寬度也為 50 pixel。

8. 演員位置

   儲存的演員位置為編輯者按下演員儲存按鈕時的最後位置，而非一開始輸入演員參數時所記載的資料，圖 18 中的演員位置.x 代表演員再 x 座標軸的 x 座標，演員位置.y 代表演員在 y 座標軸的 y 座標。

9. 演員動作序列

   演員的動作序列由演員的多個動作(action)組合而成，每個演員有各自獨立的動作序列，其中演員可以是用的動作為，往上、往下、往左、往右、放大、縮小、隱藏、出現、延遲、旋轉十個動作，藉由這十個基本的動作讓演員去組合不同的劇情。圖 18 中的 action 序列就是小狗儲存的小狗演員劇情。

```
<actor>
  <actor name 演員名稱="小狗" 圖片名稱="小狗" 圖片路徑="C:\Users\popo\Desktop\重用元件\casa-icon.jpg">
    <information move step="20" big step="20" small step="20" rotate step="90" wait time="500" />
    <information location 演員位置.x="182" 演員位置.y="112" 演員長度="50" 演員寬度="50" />
    <action 演員向上="1" />
    <action 演員向下="2" />
    <action 演員往左="3" />
    <action 演員往右="4" />
    <action 演員放大="5" />
    <action 演員縮小="6" />
  </actor_name>
</actor>
```

圖 18　actor 的 XML 檔案

3.3 提供的視覺化功能

在本小節中，會根據本研究設計的系統將提供的視覺化功能分成三類:

1. Atomic scenario(基本劇情)
2. Composite scenario(組合劇情)
3. Interactive scenario(互動式劇情)

根據提供的這三種劇情，基本劇情、組合劇情、互動式劇情，讓使用者用視覺化的方式編輯所需要的視覺化劇情。

3.3.1 Atomic scenario(基本劇情)

在基本劇情中，本研究提供多種視覺化劇情中常用的指令，當使用者按下了這些指令，系統便會呼叫對應的程式碼播放輸入的指令。重用元件中的演員動作序列即是儲存了多個基本指令。提供的基本劇情有下面幾個:

1. 演員往上
   控制的演員會根據使用者輸入的參數移動，當輸入的移動距離是 50 時，演員則會往上 50 pixel。
2. 演員往下
   控制的演員會根據使用者輸入的參數移動，當輸入的移動距離是 50 時，演員則會往下 50 pixel。
3. 演員往左
   控制的演員會根據使用者輸入的參數移動，當輸入的移動距離是 50 時，演員則會往左 50 pixel。
4. 演員往右
   控制的演員會根據使用者輸入的參數移動，當輸入的移動距離是 50 時，演員則會往右 50 pixel。
5. 演員放大
   控制的演員會根據使用者輸入的參數放大，當輸入的放大倍數為 20 時，演員的長寬會各自放大 20 pixel。
6. 演員縮小
   控制的演員會根據使用者輸入的參數縮小，當輸入的縮小倍數為 20 時，演員的長寬會各自縮小 20 pixel。
7. 演員隱藏
   當遇到了演員必須暫時消失的劇情時，則可用這基本指令，演員隱藏的時間會根據使用者本身的劇情，除非遇到了演員出現的指令演員才會再次的出現在場景中。
8. 演員出現
   配合演員隱藏指令使用，連續的演員隱藏與演員出現則可以達到演員閃爍

的劇情演出。如果沒有搭配演員隱藏，則此指令效果不明顯。

9. 演員停滯

   此指令為讓演員播放過程中停滯，停滯的時間為使用者設定的 timer_tick 時間，假設輸入的 wait_time 為 500 毫秒，演員停滯指令輸入三次，則在播放過程中演員會停滯 1500 毫秒的時間。

10. 演員旋轉

    此指令會根據使用者輸入的旋轉角度進行旋轉的動作，系統本身提供的旋轉角度有 90 度、180 度、270 度，假設使用者輸入了 90 度，演員會一直以 90 度的角度進行旋轉，直到下一次的修改。

以上十個為本系統所提供的十個基本劇情，透過基本劇情的組合，使用者可以編輯出視覺化語言中最基本的元件，演員。圖 19 為各個控制演員的參數輸入。

圖 19 編輯者可以控制的演員參數

如圖 19 所示，編輯者可以自行設定演員的參數，圖 19 右方列為目前所設定的演員序列，用滑鼠點選演員列中的演員就可以即時的對選取的演員做修改或刪除的動作。對於載入的演員檔案亦是如此，按下圖 19 中的重新整理即可以顯示出載入的演員並可以做修改的動作。

3.3.2 Composite scenario(組合劇情)

在 Composite scenario(組合劇情)中，組合劇情結合了多個 Atomic scenario(基本劇情)與控制項，並根據組合多個 Atomic scenario(基本劇情)加以延伸，演員是由 Atomic scenario(基本劇情)編輯而成，場景則是由多個演員與 Composite scenario(組合劇情)組合而成。本研究系統提供了以下幾種組合劇情:

1. 循序播放(Sequential)

   多個演員的循序播放，假設循序序列中儲存的是，演員:小狗、演員:小貓、演員:小鳥、演員:小星星這樣的序列，則在播放過程中最先播出的演員為小狗演員，其他的演員暫時不能行動，當小狗演員所有的劇情播放完畢時，小貓演員才會接下去播放，以此類推最後播放的演員為小星星演員，當小星星演員劇情播放完畢，循序播放劇情才完畢。圖 20 為編輯循序演員的設定。

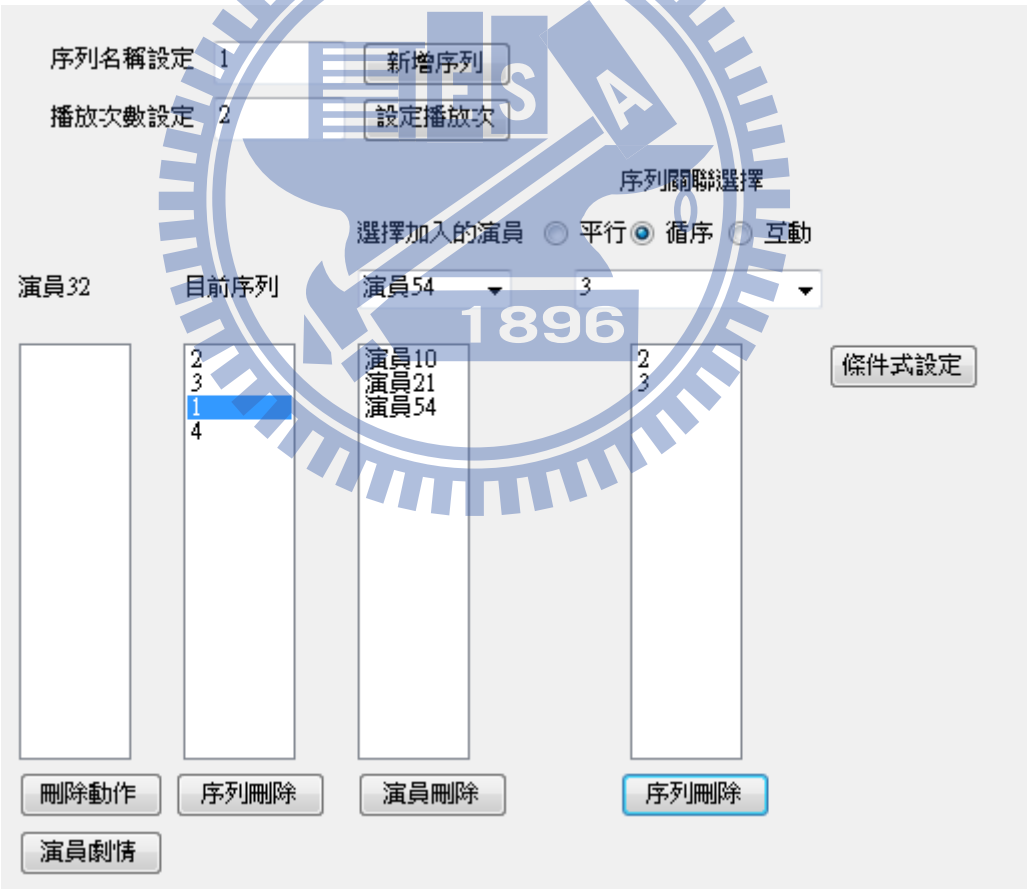


圖 20 循序劇情的設定

如圖 20 所示，上圖的循序序列有:序列 2 與序列 3，所以必須要等到序列 2 播放完畢序列 3 才會接著播放。

2. 循序迴圈

   循序迴圈劇情為循序播放劇情的延伸，循序播放劇情只播放了一次循序劇情，循序迴圈劇情則是可以根據編輯者本身輸入迴圈參數來控制循序迴圈播放次數。假設循序序列中儲存的是，演員:小狗、演員:小貓、演員:小鳥、演員:小星星這樣的序列，輸入的循序迴圈為 3 次，最先播放的演員為小狗演員之後為小貓演員，以此類推當小星星演員播放完畢時，迴圈又跳回小狗演員依序播放，當播放此循序序列到達三次時，此循序迴圈的劇情才算播放完畢。

3. 序列間的循序播放

   多個序列的循序播放，序列循序關聯為序列 1，序列 2 與序列 3，則播放的順序為序列 1 所有的演員播放完畢之後才會播放序列 2，最後才會播放序列 3。並分別可以對每個序列設定序列迴圈的次數。

4. 序列間的平行播放

   多個序列的平行播放，序列平行關聯為序列 1，序列 2 與序列 3，則序列 1、序列 2 與序列 3 會同時播放直到播放次數達到所設定的迴圈數。

5. 條件式播放

   當某序列播放時，設定的序列才會播放。假設今天播放的序列有序列 1，序列 2，序列 3，條件式序列為序列 4 並設定啟動條件式序列的為序列 2，當序列 1、序列 3 在播放的時候序列 4 並不會起動，當序列 2 一啟動時序列 4 才會跟著一起動。

以上五種組合劇情為本研究系統的組合劇情，藉由基本劇情編輯出來的演員檔，加上組合劇情與控制項即可以編輯出場景(scene)。在組合劇情中，演員檔只能選擇使用循序或者是平行序列中的一個，選擇循序劇情的話，則平行劇情的功能皆不能使用，以避免同時間演員播放錯亂的問題。

3.3.3 Interactive scenario(互動式劇情)

在互動式的劇情中，提供了使用者與系統之間的互動，讓編輯教材不再是那麼的乏味。編輯者可以利用鍵盤與滑鼠操作本系統所提供的互動式劇情，互動式劇情包括有以下幾種:

1. 點選

   用滑鼠點選了在場景中的各個演員，演員會在場景中顯示目前所點選的演員名稱，也會顯示出目前演員所播放的基本劇情。使用者可以針對點選的演員做進一步的劇情修改、新增或刪除等動作。

2. 拖拉

   在播放過程中，若使用者想即時的修改演員的位置，則可以藉由滑鼠點選演員並拖曳到使用者想要的位置，而不必再進入演員編輯的介面就可以直接做即時的修改。

3. 滑鼠右鍵與左鍵

滑鼠的控制有分成，按下右鍵、按下左鍵、滑鼠進入、滑鼠移動等控制，按下右鍵即為第一點所提到的點選功能，滑鼠移動則為第二項所說的拖曳功能。按下滑鼠右鍵則會觸發儲存演員、載入演員、儲存演員劇情、載入演員劇情、演員修改、載入演員參數、儲存句子、載入句子八個功能。

(1) 儲存演員

將目前滑鼠所指的演員儲存，儲存的項目包括演員的劇情、演員的圖片、演員大小、演員位置、所輸入的參數等資料。

(2) 載入演員

將儲存的演員檔案呼叫到場景中，系統根據使用者選取的演員檔案產生一個演員並將儲存的資料一一匯入到新演員中。

(3) 儲存演員劇情

將目前選取的演員設定的劇情儲存，如果編輯到劇情重複的演員只要設定演員參數即可，播放的劇情可由儲存的劇情載入。

(4) 載入演員劇情

將儲存好的演員劇情載入到新演員或舊演員中，以加快編輯的速度與效率。

(5) 演員修改

直接在播放過程中，載入新的演員圖片完成新演員的劇情。

(6) 載入演員參數

直接載入已經儲存好的演員參數。

(7) 儲存句子

將序列整個儲存，包含了序列中包含的各個演員、演員的動作與序列迴圈次數等資料。

(8) 載入句子

將序列整個重新載入，載入的序列可以再做修改新增演員的動作。

如圖 21 所示，在場景中按下滑鼠右鍵即會觸發上述四點所描述的儲存演員、載入演員、載入演員劇情、儲存演員劇情等劇情。

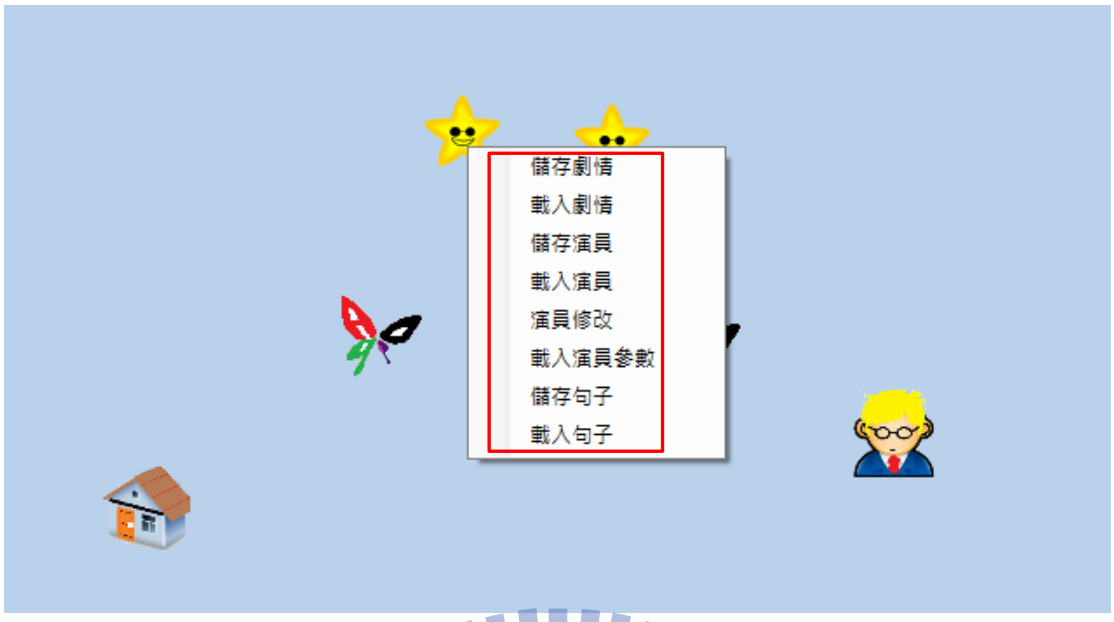


圖 21 滑鼠控制之劇情

4. 鍵盤控制

   編輯者可以利用鍵盤來控制演員的劇情，例如鍵盤的 QAWSEDRFTG 分別對應到了演員往上、往下、往左、往右、放大、縮小、隱藏、出現、停滯、旋轉的動作。

5. 條件式互動劇情

   對某個演員做滑鼠雙擊或按下鍵盤，則會有互相對應的劇情發生

6. 碰撞偵測

   當編輯者對演員拖拉碰撞到其他演員或者編輯者對演員做滑鼠移動碰撞到其他演員則會偵測到碰撞，則會產生其他對應劇情。

3.4 視覺化重用元件套用方式

在 3.3 節中，提到了本研究系統所提供的視覺化劇情。而在本節中會進一步地介紹要如何把系統提供的視覺化劇情加以儲存並套用，套用的方法有四種:

1. 演員劇情的套用
2. 演員的套用
3. 多個演員的套用
4. 句子(Sentence)劇情的套用
5. 場景劇情的套用

3.4.1 演員劇情套用

演員劇情的套用為本系統中最小單位的重用，將儲存好的演員劇情搭配 3.3.3 節中的滑鼠控制劇情將劇情套用到所選取的演員。舉例來說，當類似遇到了滿天小星星之類的劇情時，因為小星星的劇情只有一閃一閃即為基本劇情中的演員隱藏與演員出現，當類似遇到這種劇情即可以此種演員劇情儲存，之後可以套用到多個星星演員上，但是實際上使用者只編輯了一個演員劇情而已，其他的演員劇情都只是套用一開始的演員劇情而已，系統設定的演員劇情重用元件副檔名為.scn 檔案。

3.4.2 演員的套用

儲存的演員可以藉由 3.3.3 節中的滑鼠控制劇情將演員呼叫到場景裡面。編輯者可以對儲存的演員做修改參數的動作，修改的參數包括，圖片、大小而產生新的演員，但是實際上編輯者所編輯的演員只有一個，其他的只是藉由載入儲存的演員修改參數而產生。所以演員的套用這方法適合在編輯多個類似劇情的演員，藉由不斷的載入，不斷地修改參數達到快速編輯的效果，系統設定的演員重用元件副檔名為.act 檔案。

3.4.3 多個演員的套用

將多個設定好的演員儲存成一檔案，儲存的檔案包括了多個演員，演員各自的參數與劇情等資料。編輯者不需一次載入一個演員，可以一次載入多個演員。系統會將多個演員的重用元件副檔名設定為.allact 檔案。

3.4.4 句子劇情的套用

句子劇情即為演員所加入的序列，編輯者可將整個序列儲存，例如序列 1 裡面記錄著演員 1、演員 2、演員 3，則如將序列 1 儲存則會將此關聯紀錄。編輯者可以儲存多個序列，當要選擇序列關聯時則可以即時呼叫。系統會將設定的句子重用元件副檔名設定為.sen 檔。

3.4.5 場景劇情的套用

場景劇情包含了上述三種，演員劇情、演員、句子劇情在加上序列的關聯，將編輯好的場景儲存，如果需要修改時只需要再入場景劇情即可修改或新增，不必從單一演員到句子劇情都重新編輯，編輯的效率會大大的提高。圖 22 為場景重用元件所儲存的內容，包含了演員，演員劇情與句子劇情。系統會將設定好的場景劇情的副檔名設定為.play 檔案。

```
<panel_location 背景路徑="">
  <list_seq>
    <seq_name 序列名稱="2" 序列播放次數="3">
      <actor>
        <actor_name 演員名稱="演員43" 圖片名稱="演員43" 圖片路徑="C:\Users\popo\Desktop\
          <action 演員縮小="6" />
          <action 演員放大="5" />
          <action 演員往左="3" />
          <action 演員向上="1" />
          <action 演員向下="2" />
        </actor_name>
      </actor>
      <actor>
        <actor_name 演員名稱="演員54" 圖片名稱="演員54" 圖片路徑="C:\Users\popo\Desktop\
          <action 演員向上="1" />
          <action 演員向下="2" />
          <action 演員往左="3" />
        </actor_name>
      </actor>
    </seq_name>
    <seq_name 序列名稱="3" 序列播放次數="2">
      <actor>
        <actor_name 演員名稱="演員65" 圖片名稱="演員65" 圖片路徑="C:\Users\popo\Desktop\
          <action 演員往右="4" />
          <action 演員往左="3" />
          <action 演員向上="1" />
          <action 演員向上="1" />
        </actor_name>
      </actor>
```

圖 22 場景重用元件儲存的資料

四、系統設計與實作

本系統根據第三章所設計的重用元件、語法以及視覺化功能設計一視覺化劇情編輯系統。

4.1 系統架構

系統主要分成兩大部分，分別是 User Interface 與 Backend Controller，如圖 23 所示。

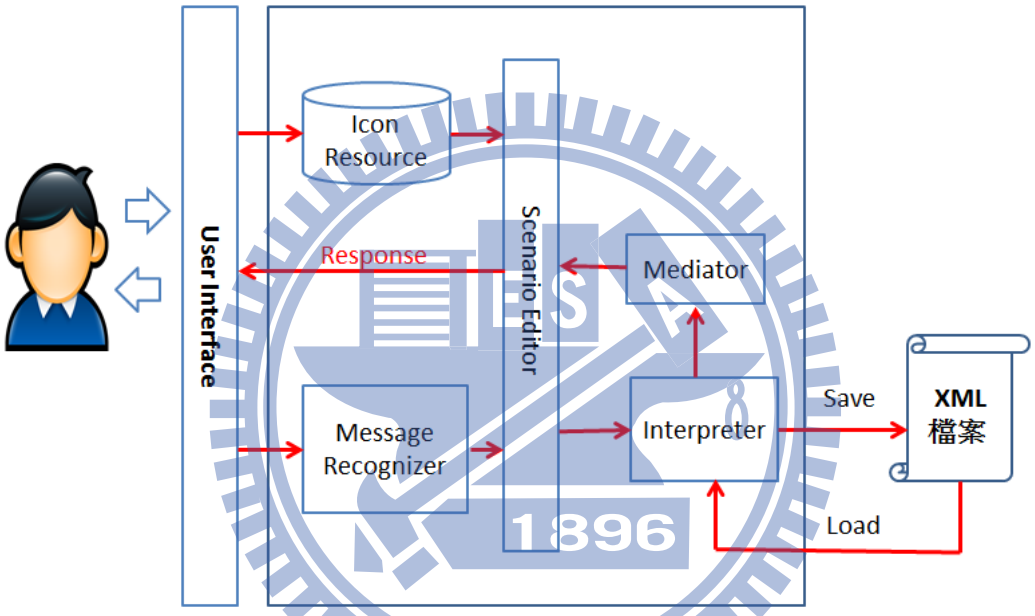


圖 23 系統架構

1. User Interface :
   使用者操作的介面，提供使用者以視覺化方式操作表單、演員與劇情等相關的資訊。
2. Icon Resource:
   儲存 Icon 的中心，提供編輯者在編輯時所需要的演員 Icon。
3. Message Recognizer
   當編輯者輸入了任何鍵盤或者是滑鼠的點擊，此元件會負責辨識使用者所傳遞的訊息。並根據鍵入的按鈕作出對應的反應，例如演員的辨識，演員的拖曳、載入演員、載入的劇情與劇情的播放等等相關訊息。根據鍵入的訊息產生出對應的事件。

4. Scenario Editor

   負責控制輸入演員劇情的編輯以及演員的參數輸入，可輸入的劇情包括第三章所提到的基本劇情、組合劇情與互動式劇情。並根據編輯者輸入的劇情回傳給編輯者。

5. Interpreter

   描述劇情的直譯器，Interpreter 會根據使用者所輸入的劇情解譯成為互相對應的演員與演員劇情。也可將所儲存的 XML 重用元件解譯成新的演員與新的演員劇情。

6. Mediator

   暫時儲存 Interpreter 解譯出來的演員以及劇情。當編輯者按下播放或者預覽的時候。系統才由 Mediator 取出對應的演員以及劇情播放並回傳給編輯者。

7. XML 檔案

   編輯者可將編輯好的劇情轉換成 XML 語法儲存，當下次遇到重複的類似劇情時可以呼叫此 XML 檔案以達到重用的效果。

4.2 系統設計

在此章節中，主要探討的本系統的程式架構，首先介紹本系統中一重要的資料結構 actor，如圖 24 所示，本系統所有的劇情全部圍繞在 actor 的資料結構，actor 資料結構裡面的 PictureBox 則是接受自 Icon Resourse 所選擇的演員圖片，pic_id 與 actor_name 則是編輯者所輸入的演員代號與演員名稱，move_step、big_step、small_step、wait_time、rotate_step 皆是編輯者可以輸入的參數，分別代表著移動距離、放大倍率、縮小倍率、動作間格時間、旋轉角度的資訊。而 action_list 則記錄編輯者所輸入的演員劇情，在此特別提到在 actor 資料結構中還包括了一個 Timer 的資料結構，演員劇情的播放順序由 action_list 記錄，如何播放的方法則是由 Timer 控制項裡的 Timer_tick 事件(Event)控制，如圖 25 所示。

```
public struct actor
{
    public PictureBox pic;
    public int pic_id;
    public string actor_name;
    public int move_step;
    public int big_step;
    public int small_step;
    public int wait_time;
    public int rotate_step;
    public List<int> action_list;
    public Timer actor_timer;
}
```

圖 24 actor 的資料結構設定

```
//=============================================================演員預覽的TIMER播放
private void timer1_Tick(object sender, EventArgs e)
{
    int temp;                                   //   紀錄滑鼠所指到演員在actor_list的index
    temp = find_index(actor_list, choose_pic);
    timer1.Interval = actor_list[temp].wait_time;
    switch (actor_list[temp].action_list[timer_start])
    {
        case 1: actor_action.pic_up(actor_list[temp].pic ,actor_list[temp].move_step);
            break;
        case 2: actor_action.pic_down(actor_list[temp].pic, actor_list[temp].move_step);
            break;
        case 3: actor_action.pic_left(actor_list[temp].pic, actor_list[temp].move_step);
            break;
        case 4: actor_action.pic_right(actor_list[temp].pic, actor_list[temp].move_step);
            break;
        case 5: actor_action.pic_big(actor_list[temp].pic, actor_list[temp].big_step);
            break;
        case 6: actor_action.pic_small(actor_list[temp].pic, actor_list[temp].small_step);
            break;
        case 7: actor_action.pic_hide(actor_list[temp].pic);
            break;
        case 8: actor_action.pic_show(actor_list[temp].pic);
            break;
        case 9: actor_action.pic_stop(actor_list[temp].pic);
            break;
        case 0: actor_action.pic_rotate(actor_list[temp].pic, actor_list[temp].rotate_step);
            break;

    }
    timer_start++;
```

圖 25 Timer 控制項的事件

在 Timer_tick 的事件中，首先記錄目前滑鼠所指演員在 actor_list 中的 index，actor_list 為記錄目前在場景中的所有演員的陣列，當抓到目前的演員的 index 時，則根據目前 index 所記錄的 move_step、big_step 與 wait_time 等資訊播放，上圖的 switch 條件控制項則是根據此 index 的 action_list 進行播放，當 action_list 目前的值為 1 則演員往上移動，action_list 目前的值為 2 則演員往下移動直到此 index 的 action_list 從頭到尾 parse 結束則此演員結束播放。

本研究所設計的系統還另外提供了重用元件的套用，重用元件的儲存兩個方法，以下為儲存演員的 class 因為此部分程式較複雜，所以只擷取了部分的程式

碼，如圖 26 所示。

```
private void 儲存演員ToolStripMenuItem_Click(object sender, EventArgs e)                    //儲存演員
{
    for (int i = 0; i < actor_list.Count(); i++)
    {

        if (actor_list[i].actor_name == choose_pic)
        {
            if (saveFileDialog1.ShowDialog() == DialogResult.OK)
            {
                XmlDocument doc = new XmlDocument();
                XmlElement actor = doc.CreateElement("actor");
                doc.AppendChild(actor);

                //actor 屬性
                 XmlElement actor_name = doc.CreateElement("actor_name");
                 actor_name.SetAttribute("演員名稱",actor_list[i].actor_name);
                 actor_name.SetAttribute("圖片名稱",actor_list[i].pic.Name);
                 actor_name.SetAttribute("圖片路徑",actor_list[i].pic.Tag.ToString());
                 actor.AppendChild(actor_name);

                //參數屬性

                 XmlElement information = doc.CreateElement("information");
                 information.SetAttribute("move_step",actor_list[i].move_step.ToString());
                 information.SetAttribute("big_step", actor_list[i].big_step.ToString());
                 information.SetAttribute("small_step", actor_list[i].small_step.ToString());
                 information.SetAttribute("rotate_step", actor_list[i].rotate_step.ToString());
                 information.SetAttribute("wait_time", actor_list[i].wait_time.ToString());
                 actor_name.AppendChild(information);
                //演員位置
```

圖 26 檔案儲存的程式碼

首先利用 XmlDocument 先 new 出一 XML 檔案，並設定 actor 為其父節點，actor_name 為 actor 的子節點，actor_name 裡面記錄著演員名稱與演員代號等資訊。之後再設定一 information 節點，此節點為 actor_name 的子節點，此節點記錄著此演員所設定的參數，包括移動距離、旋轉角度、放大與縮小倍率等資訊，此 class 還有包含著記錄演員位置與演員大小的 information_location 的節點與記錄演員動作的 action 節點，因為程式碼太過冗長這邊不一一列出。

4.3 系統編輯流程

以下為本系統的編輯流程，首先編輯時需要先有文字式教材的構想，因為這部分屬於編輯者的設計巧思，本研究不加以探討。編輯者可以將所所設計的文字式教材藉由多媒體編輯器轉成視覺化的教材，如圖 27。

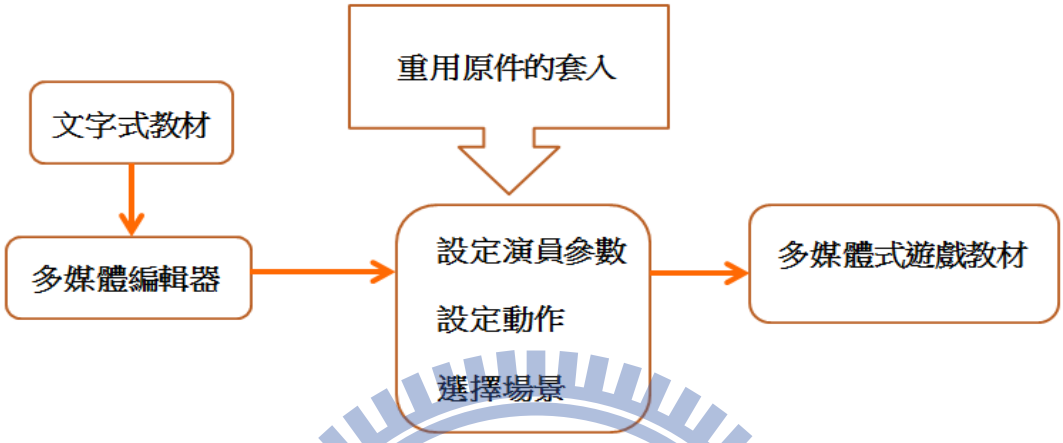


圖 27 文字式教材轉換成多媒體教材的流程

4.3.1 重用元件的編輯

本系統所提供的重用元件總共有四類，分別是演員、演員劇情、句子劇情與場景劇情。在編輯的過程可以直接儲存想要儲存的資料如圖 28 所示。

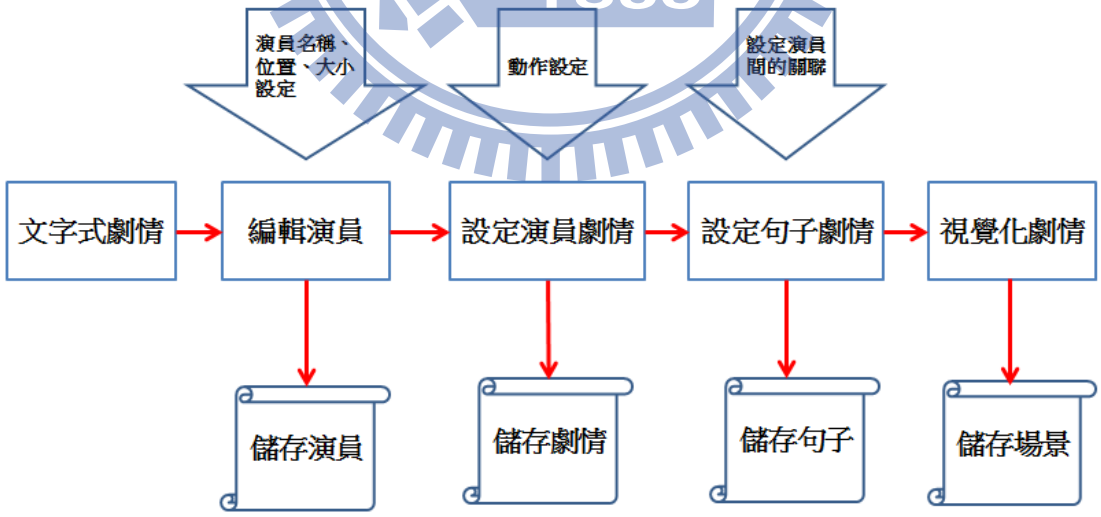


圖 28 重用元件的編輯

在編輯演員的過程中，首先使用者須先從主視窗進入演員設定視窗，如圖 29 所示，進入演員參數設定視窗之後，編輯者便可以設定演員大小，演員出現位置演員移動距離與演員的 icon 等參數。圖 30 右邊的 listbox 記載著目前所新增的演員有哪些只要在 listbox 中點選演員即可對此演員修改 icon。

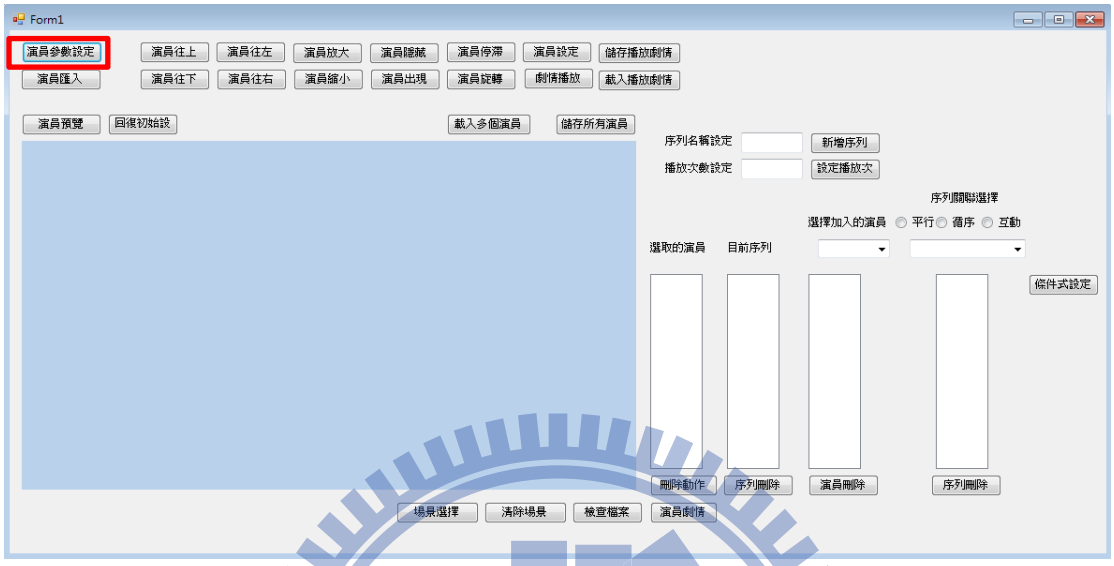


圖 29 系統的主畫面

圖 30 演員設定的編輯畫面

當編輯演員完成之後，即可匯入場景，並在場景中按下滑鼠右鍵，則可進行演員儲存的動作。如圖 31 所示，根據目前滑鼠位置所指到的演員並按下右鍵即會儲

存此演員，下圖儲存的為名稱為蝴蝶的演員。

圖 31 系統的儲存演員畫面

除了儲存演員之外，也可以將演員設定的劇情儲存，儲存目前滑鼠所指到演員劇情，並在演員 icon 上按下滑鼠右鍵，就會出現儲存劇情的選項，如上圖所示，按下儲存劇情時則會儲存蝴蝶演員的劇情，上圖儲存的演員劇情為演員放大、演員放大、演員縮小、演員縮小、演員往左、演員往左、演員往右、演員往右、演員往上、演員往上、演員往下、演員往下。

系統可以儲存演員之間的句子劇情，編輯者可以新增新的序列並將設定好的演員加入序列中，如圖 32 所示。編輯者可以點選下拉式選單點選欲加入序列的演員。系統會自動匯入目前場景中所有的演員到下拉式選單中。之後在選擇序列之間的關聯，有平行、循序、互動三個關聯可以選擇。

圖 32 序列的設定

當整個劇情皆編輯完畢時，編輯者可將此場景劇情儲存，此劇情包括，演員劇情元件、演員元件、句子劇情元件與序列關聯，如圖 33 所示。L3 序列裡面包含的演員有演員 3、演員 4 與演員 5，加入平行關聯的序列有序列 L1、序列 L2、序列 L3 與序列 L4。

圖 33 系統的播放畫面

4.3.2 重用元件套用與修改

根據上一小節所儲存的重用元件，編輯者可以將其載入到系統中作進一步的修改與新增，修改的流程如圖 34 所示。

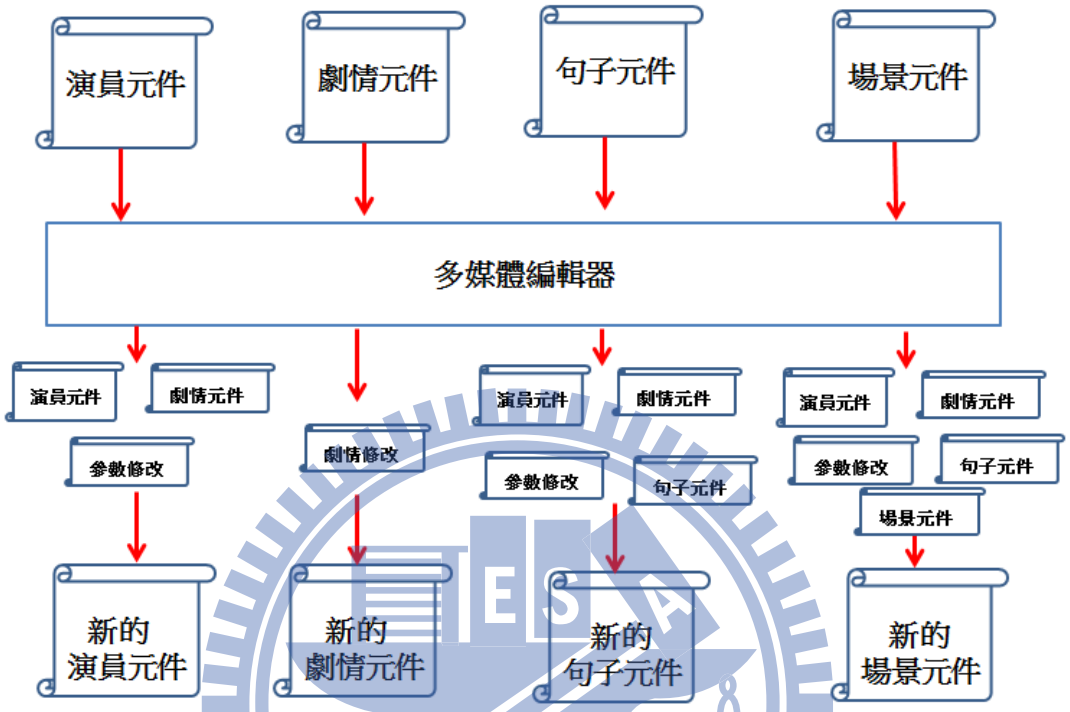


圖 34 重用元件的套用與修改流程

如圖 34 所示，每一種重用元件可以結合產生新的重用元件，例如演員元件加上劇情元件的載入可以產生新的演員元件。新的場景元件也可以藉由載入演員元件與句子元件產生新的場景元件。

每個演員的參數與劇情皆能夠載入並修改。修改的方法有:

1. 直接開啟 XML 檔案修改

   如圖 35 所示，XML 記錄著各演員的資訊，直接對 XML 檔案修改之後，載入系統的結果也會修改。

```
<actor>
  <actor_name 演員名稱="小喵咪" 圖片名稱="小喵咪" 圖片路徑="C:\Users\popo\Desktop\重用元件\cat-icon.jpg">
    <information move_step="20" big_step="20" small_step="20" rotate_step="0" wait_time="500" />
    <information_location 演員位置.x="224" 演員位置.y="109" 演員長度="50" 演員寬度="50" />
    <action 演員向上="1" />
    <action 演員向下="2" />
    <action 演員往左="3" />
    <action 演員往右="4" />
    <action 演員放大="5" />
    <action 演員縮小="6" />
  </actor_name>
</actor>
```

圖 35 XML 修改結果

例如將圖 35 最後兩個動作改成演員隱藏與演員出現則系統載入的結果則會不同，如圖 36 所示，圖 36 為上述修改檔案載入系統的結果。

圖 36 XML 檔案修改結果

2. 將重用檔案載入到系統修改

將上圖的重用資料，載入到系統直接修改如圖 37 所示，進入參數設定畫面，輸入重新整理，則會顯示出目前所有的演員。之後按下設定演員即可顯現出目前該演員所設定的資料，修改完畢之後再按下演員修改即完成修改。

圖 37 演員修改

4.3.3 系統比較

根據本研究所設計的系統與第二章相關研究部分所探討的各個研究系統比較，主要是根據各系統重用方式作探討，比較的重用套用方式有演員的套用、演員劇情的套用、句子劇情的套用、播放場景劇情的套用。下表為比較的結果。(下表中 N/A 表示不確定)

表 1 系統功能比較

	演員套用	劇情套用	句子套用	場景套用
視覺化樣板				V
Scratch	V			V
ScenEdit	N/A	N/A	N/A	N/A
Google Blockly	N/A	N/A	N/A	N/A
SeGAE	N/A	V		V
MACHS	N/A	V		V
本研究系統	V	V	V	V

五、實作範例

在此章節中，會以一實際視覺畫劇情編輯，此劇情為:「夏夜星空的場景」，在此劇情中，包含的演員有多個星星演員與蝴蝶演員，星星的劇情為在天空中一閃一閃的發亮，蝴蝶在底下輕快的飛舞，如圖 38 所示。

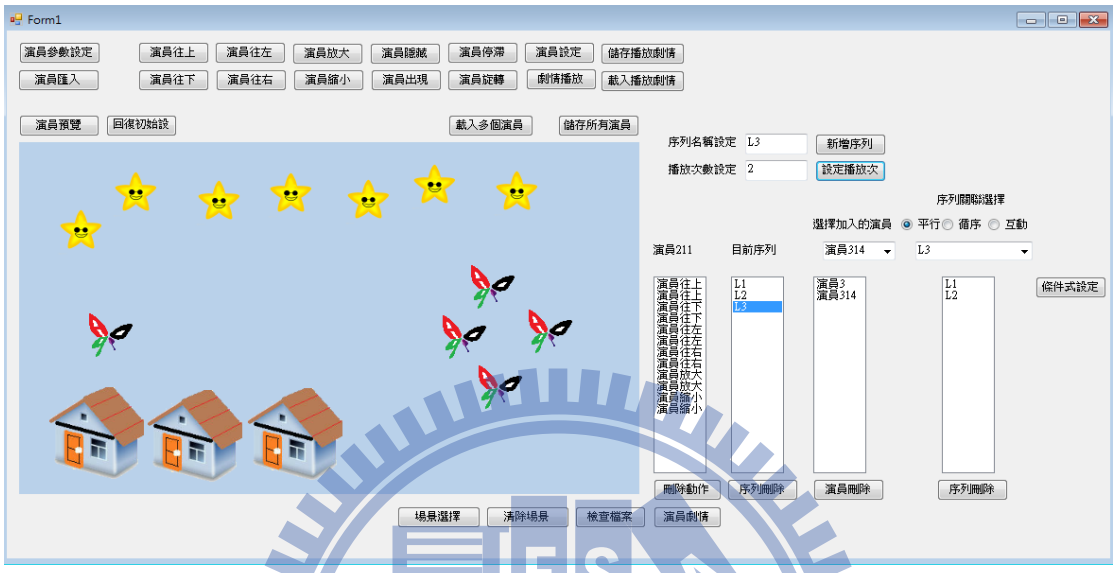


圖 38 夏夜星空的場景的劇情呈現

上圖場景中，總共出現的有三種演員:

1. 星星演員

   星星演員共有七個，演員名字分別是星星 1 到星星 7，播放的星星演員劇情分別有兩種，不斷的旋轉與閃爍，旋轉劇情的設計為設定旋轉角度為 90 度，然後播放四次即會回到原來的角度。閃爍劇情的設計為藉由連續的演員出現與演員隱藏的劇情達到閃爍的效果。因為星星閃爍的劇情都大同小異，所以在設計此夏夜星空場景時，編輯了一次閃爍星星劇情即可，剩下利用劇情或是演員的重用就可以完成編輯。

2. 蝴蝶演員

   蝴蝶演員共有四個，分別是蝴蝶 1 到蝴蝶 5，其中的劇情為基本劇情的組合。。

3. 房子演員

   房子演員本身沒有劇情，當作背景使用，在此場景中既沒有加入平行劇情關聯也沒有加入循序劇情的關聯，只當作一個靜態的演員背景。

當編輯好劇情之後，按下演員設定之後按下劇情播放，則場景內的所有演員皆會按照上述設定的劇情播放。編輯者在編輯過程可以分別儲存演員、演員劇情、句子劇情或者市場景劇情，所以當下次編輯者有需要類似的劇情時，直接呼叫已經儲存的重用元件套用即可。

六、結論與未來研究方向

在本研究中，實作出幾種重用元件的套用分別是演員重用元件、演員劇情重用元件、句子重用元件與場景重用元件。但是這樣的重用元件與套用方式對編輯者來說可能是不足夠的，所以本研究在這邊提出了幾點的改善方法與未來可研究的一些方向。

6.1 結論

遊戲式教材的重要性越來越重要，遊戲式教材不僅僅可以增加學習者的學習動機也可以增加學習者的學習成就感。以下為本研究系統所研究的成果有以下幾點:

1. 設計視覺化的語法:
   根據本研究設計的視覺化語法，編輯者可以在本研究設計的系統設計視覺化的劇情。
2. 重用元件的設計與重用元件的套用:
   本系統針對編輯者在編輯遊戲式教材的過程中可能會想重複使用到的元件作整合設計。在編輯過程中，可以將編輯的演員、場景等儲存為重用元件，當編輯者在編輯過程中遇到重複性質高的劇情，可以直接載入重用元件。並根據重用元件的不同，提供不同的套用方式。
3. 重用元件的修改與結合
   重用元件的設計為 XML 檔案，編輯者可以直接開啟並修改。編輯者可以針對不同的重用元件產生新的重用元件。

6.2 未來研究方向

本研究提出了以下的幾點，未來可以改進的方法:

1. 跨平台式的重用:
   本系統的重用元件是以 XML 為基礎，重用元件的設計只能套用在本系統中。所以如果把本系統設計的 XML 重用元件當作一 Intermediate language 再藉由 translator 將此 Intermediate language 轉成各種不同的程式語言，例如 JavaScript 或 ActionScript，這樣就可以達到跨平台式的重用，如圖 39 所示，利用一 language translator 把 Intermediate language 轉成各種平台的語言。

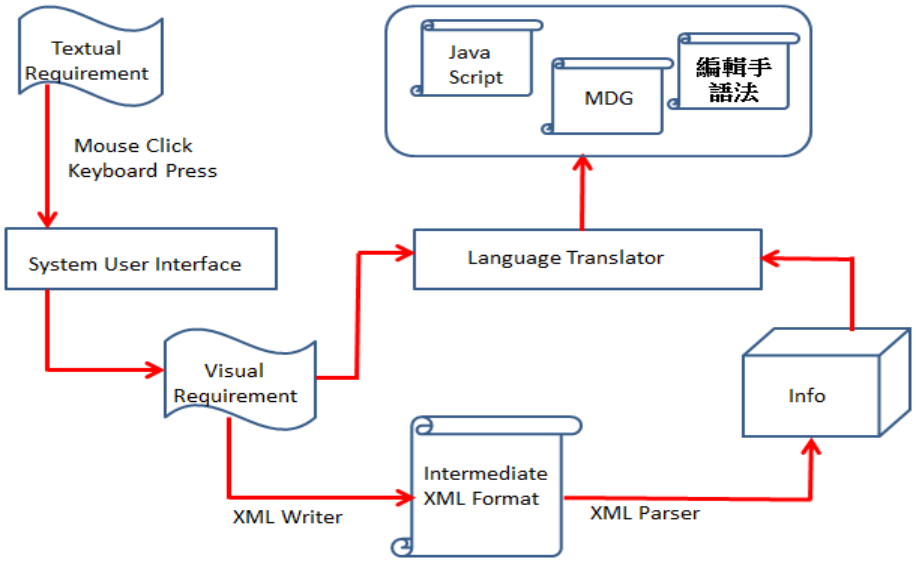


圖 39 語言的轉換-Meta model

2. 更複雜重用元件套用:
   因為本系統只提供了四項重用元件套用方式，不一定能夠滿足編輯者的需求，未來可能可以加入更複雜的句子套用方式，例如某些時段演員式進行平行播放另一個時刻進行循序播放。或者多加入一些條件式的組合劇情，例如 If then else 條件劇情元件、Switch 劇情元件、Branch 劇情元件或者是 goto 劇情元件，加入這些劇情應該可以使使用者在編輯上有更多的便利性。
3. 劇情的編輯更多樣化:
   這邊針對的是可編輯的基本劇情多樣性，因為目前系統提供的劇情對於編輯者可能還不太足夠，新增路徑移動、插入聲音與插入文字等的基本劇情可以使編輯的劇情更加多樣化。

參考文獻

[1] H. S. Barrow and R. M. Tamblyn, "Problem-based learning: An approach to medical education," ed, 1980.

[2] K. Salen and Z. E., *Rule of play: Game design fundamentals*. Cambridge MA: MIT Press, 2004.

[3] D. Michael and S. Chen, *Serious games: Games that educate, train and inform*. Boston, MA: Thomson Course Technology, 2006.

[4] S. D. Freitas and M. Griffiths, "Online gaming as an education tool in learning and training," *British Journal of Educational Technology*, vol. 38, pp. 535-537, 2007.

[5] J. Bourgonion, R. S. M. Valcke, and T. Schellens, "Exploring the Acceptance of Video Games in the Classroom by Secondary School Students " presented at the Proc. ICCE2009, 2009.

[6] M. Pivec, "Editorial: Play and learning: potentials of game-based learning," *British Journal of Educational Technology*, vol. 38, pp. 387-393, 2007.

[7] F.-H. Tsai, K.-C. Yu, and H.-S. Hsiao, "Designing Constructivist Learning Environment in Online Game," in *Digital Game and Intelligent Toy Enhanced Learning, 2007. DIGITEL '07. The First IEEE International Workshop on*, 2007, pp. 212-214.

[8] C. Ching-Chiu, "An Investigation of Learning Style Differences and Attitudes toward Digital Game-based Learning among Mobile Users," in *Wireless, Mobile and Ubiquitous Technology in Education, 2006. WMUTE '06. Fourth IEEE International Workshop on*, 2006, pp. 29-31.

[9] R. M. Bottino, L. Ferlino, M. Ott, and M. Tavella, "Developing strategic and reasoning abilities with computer games at primary school level," *Computers & Education*, vol. 49, pp. 1272-1286, Dec 2007.

[10] M. Kebritchi, A. Hirumi, and H. Y. Bai, "The effects of modern mathematics computer games on mathematics achievement and class motivation," *Computers & Education*, vol. 55, pp. 427-443, Sep 2010.

[11] J. Buur, "Ethnographic Video as Design Specs," ed, 2010.

[12] A. Ohnishi and N. Tokuda, "Visual software requirements definition environment," in *Computer Software and Applications Conference, 1997. COMPSAC '97. Proceedings., The Twenty-First Annual International*, 1997, pp. 624-629.

[13] S. Konrad, H. Goldsby, K. Lopez, and B. H. C. Cheng, "Visualizing Requirements in UML Models," in *Requirements Engineering Visualization, 2006. REV '06. First International Workshop on*, 2006, pp. 1-1.

[14] D. J. Chen, W. C. Chen, and K. M. Kavi, "Visual requirement representation," *The Journal Of System and Software*, vol. 61, pp. 129-143, 2002.

[15] 許光軒, "Design and Implementation of ultimate basic visualization metaphors for composing arbitrary visualization," 2010.

[16] D. Q. Zhang and K.Zhang, "One the design of generic visual programming environment," 1998.

[17] K. Zhang, D. Q. Zhang, and J. N. Cao, "Design, construction, and application of a generic visual language generation environment," *Ieee Transactions on Software Engineering*, vol. 27, pp. 289-307, Apr 2001.

[18] A. Yessad, J.-M. Labat, and F. Kermorvant, "SeGAE: A Serious Game Authoring Environment," pp. 538-540, 2010.

[19] A. Mujika, D. Oyarzun, I. Iparragirre, J. Dominguez, and J. Arambarri, "MACHS: An authoring tool to create serious games for machine-tool operator training," *2011 IEEE 9th International Conference on Emerging eLearning Technologies and Applications*, 01 2011.

[20] C.-Y. Chiu, "The design and implementation of the visual requirement representation template and its customization web-based system – using multimedia yearbook as example," 2004.

[21] *Scratch* Available: http://scratch.mit.edu/

[22] *Google Blockly*. Available: http://code.google.com/p/blockly/?redir=1

[23] V. Emin, "ScenEdit: an authoring environment to design learning scenarios," presented at the Eighth IEEE International Conference on Advanced Learning Technologies, 2008.