

# 國立交通大學

資訊學院 資訊學程

碩士論文

以 MDG (MLCs Directed Graph) 為基礎來尋求

一多媒體教材素材的品質管制方法

A Study of a Quality Control Method -  
Based on MDG (MLCs Directed Graph)

研究生：李仲智

指導教授：陳登吉 教授

中華民國九十九年八月

以 MDG (MLCs Directed Graph) 為基礎來尋求  
一多媒體教材內容的品質管制方法  
A Study of a Quality Control Method -  
Based on MDG (MLCs Directed Graph)

研究生：李仲智

Student : Chung-Chih Lee

指導教授：陳登吉

Advisor : Deng-Jyi Chen



Hsinchu, Taiwan, Republic of China

中華民國九十九年八月

# 以 MDG (MLCs Directed Graph) 為基礎來尋求

## 一多媒體教材內容的品質管制方法

學生：李仲智

指導教授：陳登吉 博士

國立交通大學

資訊學院

資訊學程碩士班

### 摘要

多媒體教材日漸普及，其品質也益受重視，而目前大多以其內容、創意為品管的目標，但製作過程就比較少被討論。本論文希望能在製作過程中，以軟體工程中的 Graph 為基礎，利用其現有的理論及演算法，希望能幫助多媒體教材在製作過程的錯誤減少，進而使的品質能夠提昇。

藉由定義 MLCs Directed Graph (MDG)，來達成將教材及素材間的關連建立。它將多媒體教材的組成素材視為節點(nodes)，代表多個素材節點組成的邏輯群組也是一個節點；節點與節點間關連以邊(Directed edges)串連形成 Directed Graph (Digraph)，而此定義此關連的規範，是來自於依其自然語言需求而產生的「需求表」，進而依其需求表，將教材與素材的關連轉換成 Digraph。一份教材也視為一個節點，以它為根節點 (root node)，依其邊 (directed edges)的關連可以找出所有相關連的節點，這就是一個 MDG。

有此 MDG 之後，便可以尋找多媒體教材製作過程可能發生的錯誤或品質不良事件，找出這些事件發生的模式，利用 Graph 中的理論及方法，可在系統內自動化檢驗出來，冀以將之排除，使其在製作過程的錯誤減少，以達品質提昇的目的。本研究歸納出以下九大點的品質不良事件：一、利用 MDG 的關連結構，可將教材內散亂無章的素材串連起來，並方便素材的管理。二、參考 MDG 的關連，比較容易對照相關素材的正確性。三、教材內或邏輯群組內的素材，可有系統地做整備檢查，防止素材的缺漏。四、當素材面臨修改變更時，由 MDG 可以找出與之相關的節點，以利週邊效應的評估。五、可利用 MDG 來標記出沒有被使用到的素材節點。六、設計階段的元件缺漏檢查。七、基本的語意檢查。八、MDG 可定義群組關係與節點關連，藉此可做到語法外的元件群組關連檢查。九、可以發現元件間的遞迴。

以 MDG 為基礎，冀望未來有更多可能的發展及研究方向；例如歸納出更多的品質不良事件、進階的語法語意檢查、無限迴圈的檢查、Valid & Validation 及反向工程等等。

# A Study of a Quality Control Method - Based on MDG (MLCs Directed Graph)

Student : Chung-Chih Lee

Advisor : Dr. Deng-Jyi Chen

Degree Program of Computer Science  
National Chiao Tung University

## ABSTRACT

Multimedia Learning Contents (MLCs) become popular and quality of it is also becoming important. But most of them focus on content quality, few taking care on its production process quality issues. This study wishes to focus on process control issues of MLCs. We want to use Graph theories to help reducing errors happening during MLCs producing.

This study defines a new term ‘MLCs Directed Graph (MDG)’, and takes care on those materials of MLCs and builds relations on materials to MLCs or materials. The build-up materials of MLCs are seen as nodes, and a logical organization of material nodes could also be seen as a node. Nodes and nodes are linked by directed edges and which forms a directed graph (Digraph). It comes from demands of natural language, and translates it to a digraph. A MLC itself could be seen as a root which is a node, too. By searching those edges related to a root we could find a Digraph, and this is an MDG.

With this MDG, we could find and define MLCs quality defection issues, finding samples of those issues, and prevent form happening of these issues by using graph and graph mythologies. And this will improve the quality of MLCs by reducing defection issues. This study concludes 9 defection issues: (1) Materials in chaos and without management. MDG could build a structure of materials and easy to management. (2) With edge relations and MDG, it is easier to check the correction of materials. (3) Integrity of materials of MLCs or logical organizations could be checked easily and systematically. (4) When changes happen, side affections could be evaluated easily by reviewing nodes related. (5) Unused nodes could be noted by MDG. (6) Lacking material nodes check by MDG at design time. (7) Basic syntax error check. (8) Extra group relation defined between nodes helps node group relation examinations beyond syntax. (9) Looping could be marked.

Basing on MDG, there are lots things could be dig in. Maybe more quality deflection issues could be found. Further syntax error find out. Infinite loop find out at design. Valid & Validation and reversing engineer may work out.



## 誌謝

本論文得以完成，感謝陳登吉教授的大力協助及孜孜不倦的指導，花費了許多的時間及精神，開發新的靈感、修正論文的方向，並提供實驗室的資源使用。感謝孔崇旭教授為了論文指導，付出的許多時間及精神，及台中新竹每週的舟車勞頓。也感謝同學之間的相互鼓勵及學習。

感謝家人的支持。寫論文的過程中，無數個週末沒辦法出去走走，無數個晚上要熬夜寫論文及程式，家人精神上的鼓勵，是支持我努力下去的最大動力。

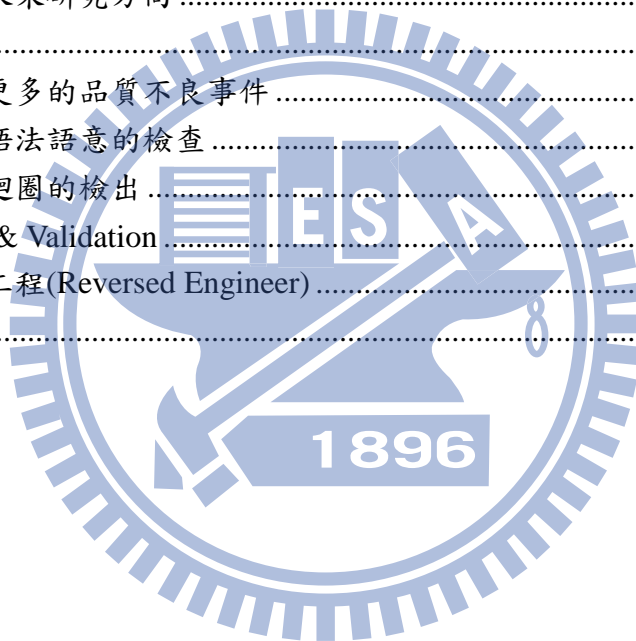
特別感謝妻子如玉在這段時間的支持、家務的辛勞及體諒，還為我們生下了一隻可愛的兒子，希望兒子們健康、聰明地長大。



# 目錄

摘要 .....	i
ABSTRACT .....	ii
誌謝 .....	iv
目錄 .....	v
表目錄 .....	vii
圖目錄 .....	viii
一、 緒論 .....	1
1.1 研究背景與動機 .....	1
1.2 研究目的 .....	2
1.3 研究方法 .....	3
1.4 章節概要 .....	3
二、 文獻探討 .....	5
2.1 多媒體教材的品質認證 .....	5
2.2 SCORM .....	7
2.2.1 Asset .....	8
2.2.2 Sharable Content Object (SCO) .....	9
2.2.3 Activities .....	9
2.2.4 Content Organization .....	10
2.3 多媒體教材品質管制模型設計 .....	11
2.3.1 Multimedia Learning Contents Process Life Cycle .....	11
2.3.2 MLCs Directed Acyclic Graph (MDAG) .....	13
2.3.3 MLCs Bidirectional Traceability Matrix .....	15
三、 多媒體教材製作過程品質管制方法 .....	17
3.1 多媒體教材與素材的關連：MDG .....	17
3.1.1 Content Organization .....	18
3.1.2 多媒體教材及其自然語言需求表 .....	19
3.1.3 素材邏輯群組及 Digraph .....	21
3.1.4 Content Organization + 素材 Digraph .....	22
3.1.5 MLCs Directed Graph (MDG) .....	23
3.1.6 reversed MDG (rMDG) .....	24
3.2 MDG 檢查法及可排除之品質不良事件 .....	25
3.2.1 MDG 可將散亂素材加以管理 .....	25
3.2.2 容易對照相關素材的正確性 .....	26
3.2.3 素材整備檢查 .....	28
3.2.4 週邊效應評估 .....	29

3.2.5	標記沒有被使用到的素材 .....	32
3.2.6	設計階段元件缺漏檢查 .....	33
3.2.7	初步的語意檢查 .....	34
3.2.8	語法外的元件群組關連 .....	35
3.2.9	元件的遞迴 .....	36
3.3	小結 .....	37
四、	系統架構及實作 .....	38
4.1	系統架構 .....	38
4.2	系統實作 .....	39
4.2.1	點(Node)及邊(Directed Edge)類別設計 .....	40
4.2.2	MDG 的展開法 .....	41
4.2.3	各種不良事件的檢驗 .....	43
五、	結論與未來研究方向 .....	45
5.1	結論 .....	45
5.2	歸納更多的品質不良事件 .....	46
5.3	進階語法語意的檢查 .....	46
5.4	無限迴圈的檢出 .....	47
5.5	Valid & Validation .....	48
5.6	反向工程(Reversed Engineer) .....	48
參考文獻	.....	50





## 表目錄

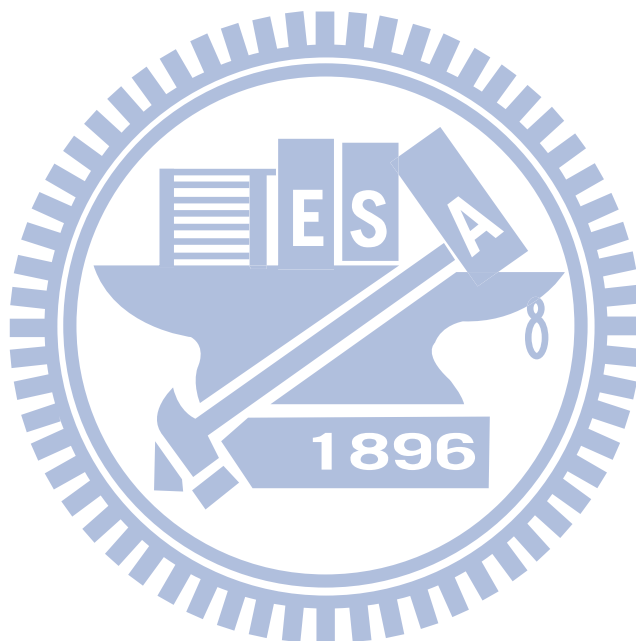
表 1 : 自然語言需求(Requirement in Natural Language)- 以編輯手製作多媒體教材的分鏡腳本(Storyboard).....	13
表 2 : MLCs Forward Dependency Matrix (MFDM).....	15
表 3 : MLCs Backward Dependency Matrix (MBDM) .....	16
表 4 : 自然語言需求範例 .....	20
表 5 : 自然語言需求 .....	34



## 圖目錄

圖 1 · 八大面向檢核圖 .....	5
圖 2 · Examples of Assets .....	8
圖 3 : Conceptual Makeup of a SCO (SCORM® 2004 4th Edition Content Aggregation Model (CAM) Version 1.0) .....	9
圖 4 : Conceptual Representation of Activities (SCORM® 2004 4th Edition Content Aggregation Model (CAM) Version 1.0) .....	10
圖 5 : Conceptual Illustration of a Content Organization and a Content Aggregation (SCORM® 2004 4th Edition Content Aggregation Model (CAM) Version 1.0).....	10
圖 6 : 多媒體教材開發流程(MLCs process flow chart).....	12
圖 7 : 以編輯手製作多媒體教材的分鏡腳本(Storyboard)展開的 MDAG.....	14
圖 8 : 汽車零件關聯圖與 MDG.....	18
圖 9 : 傳統教材章節關連 .....	18
圖 10 : 傳統教材章節對應到 DAG .....	19
圖 11 : 以編輯手製作多媒體教材的分鏡(Scene)所代表之 MDG.....	22
圖 12 : MLCs Directed Graph (MDG) .....	23
圖 13 : MDG.....	23
圖 14 : reversed MLCs Directed Graph( rMDG). .....	24
圖 15 : MDG 及素材檔案對應示意圖.....	25
圖 16 : 某個劇本 Script 2 裡的圖片素材演員.....	26
圖 17 : 劇本 Script 2 相關的前後兩劇本裡的圖片素材演員.....	27
圖 18 : 與設計時的 MDG 比較.....	27
圖 19 : MDG 素材整備檢查.....	29
圖 20 : MDG 週邊效應的發生.....	29
圖 21 : 週邊效應需評估的節點 .....	30
圖 22 : 週邊效應—無共用關連節點的修改 .....	31
圖 23 : 週邊效應—有共用關連節點的修改 .....	32
圖 24 : 以 rMDG 幫助標記未被使用的教材 .....	33
圖 25 : 元件缺漏示意圖 .....	33
圖 26 : 語意檢查示意圖 .....	34
圖 27 : 劇本示意圖及其所代表 MDG.....	35
圖 28 : MDG 中的「群組」關係圖.....	36
圖 29 : 元件遞迴示意圖 .....	36
圖 30 : (1)欲把 Script 1 加入 Script 2 下。(2)檢查 rMDG, 發現 Script 1 已在 Script 2 的 rMDG 中, 表示遞迴的存在。 .....	37
圖 31 : MLC Quality Control System。 .....	38

圖 32	: MQCS on Java EE。	39
圖 33	: 點(Node)及邊(Directed Edge)的 class diagram。	40
圖 34	: 點(Node)及邊(Edge)的 ER diagram。	41
圖 35	: MDG 的類別圖(Class Diagram)。	42
圖 36	: 序列圖(Sequence Diagram)－檢驗節點下之子節點是否整備齊全。	44
圖 37	: 現有的 MDG 關連圖	46
圖 38	: 加入語法、屬性的 MDG 示意圖	47
圖 39	: MDG 的無限迴圈檢核示意圖	48
圖 40	: 反向工程示意圖	49



# 一、緒論

## 1.1 研究背景與動機

今日個人電腦的普及帶動了數位學習的風潮，數位學習網站林立，數位學習教材也日益常見。掌上型行動裝置，如智慧型手機、電子書、PDA…等，提供了數位學習更多的可能性。

因此，數位學習教材的「品質」，日漸成為關注的對象。品質不好的教材，會使學習的效果大打折扣；相反地，有創意、高品質的數位教材，可以提升學員學習的成果。數位教材的內容及製作可能良莠不齊，而造成品質不佳的因素，可以分成兩個面向來討論：

### 1. 多媒體教材內容的設計

教材內容的設計，大多仰賴於教案編撰者的經驗、巧思及創意。好的編撰者可以抓出教材困難處，並在教案中多所著墨，或者有其獨門生動有趣的方法，可以引起學員注意、容易理解等等。好的內容設計可能會帶來較好的學習成果，其「品質」較佳；不好的內容設計，可能導致學習效果不佳，事倍而功半，其「品質」較差。

### 2. 多媒體教材的製作

教材的製作過程，如果發生了問題，也會導致教材的品質低落。如果在教材的製作過程中發生了問題，例如圖片、聲音的誤植，或互動元素發生錯誤，都會使學員發生混淆及誤解；即使有了一份優秀的教案設計，因為製作上的問題，同樣會使教材的「品質」下降。

本文希望以「軟體工程」的概念，來幫助多媒體教材的「品質管理」。第一項多媒體教材內容的設計，因為涉及個人創意，所以不在本文的目標中；但第二項多媒體教材的「製作過程」，是可以有方法去檢驗及規範的。於是希望能在軟體工程中，找出一個「品質控管的方法」，可以規範、自動化檢查的項目，使教材在製作過程中，自動偵測及預防發生品質不良事件的發生，冀以降低品質不良事件的發生。

## 1.2 研究目的

多媒體教材的製作與傳統紙本教材的製作大大的不同，並不像一般紙本教案，大抵就是排版及內容的校訂。多媒體教材則複雜的多，所以需要一個開發的流程，將多媒體教材從設計到製作，完整的納入管理。多媒體教材可以看成是一群素材的集合，在素材上賦予一些參數及行為，這樣便組成多媒體教材。開發過程中，必定會面對許多的錯誤需要更正，或者素材共享在需要做變更時的週邊效應，如果沒有加以管理，可以預期品質不良事件發生的可能性很高。依照製造多媒體教材的自然語言需求表，可以將一份教材轉換成素材及素材組成的邏輯群組，且可以對應到一個Directed Graph[1]，利用這個Graph及其特性，素材可以加以管理。

再將Graph的理論應用在開發流程系統中，找出「品質不良事件」的範本，配合Graph現有的演算法，並且更進一步地，可以自動化地檢查錯誤，依其不良事件範本及Graph的理論和演算法，規範出尋找這些不良事件的方法，讓教材在設計時，就可以發現品質不良事件；好像目前大家常用的程式編輯器，如果發現語法錯誤便會發出Syntax Error的訊息一樣，可以將一些常見的品質不良事件樣本整理歸納出來。用graph的理論及方法把它找出來，並套用在工具內，就形成了一套機制，可以協助偵錯、減少錯誤發生，以提升它的品質。

多媒體教材製作最常見的錯誤，便是素材的引用發生錯誤，而造成不一致(inconsistency)的現象，而導致此不一致現象的原因很多，因此需要整理歸納後加以自動化管理。傳統作法上的品質管制方法，只是將需檢核的項目一項項列表，再以人工檢核，或憑其經驗找出相關錯誤，難免會有疏漏而發生品質不良。

本研究的主要目的有下列三點：

1. 在現有的製作流程加入分析檢核

根據交通大學軟體工程實驗室提出的多媒體教材品質管制系統(MQCS: MLCs Quality Control System)[1]中，分析品質不良事件的發生原因，加入檢核，以冀在設計階段找出錯誤。

2. 以DAG現有的演算法尋求品質管制方法

在MQCS中的MDG (MLCs Directed Graph)表示出的素材清單，因為它是一種

Digraph，所以我們希望以它graph的特性，利用graph已知的演算法，用在系統的偵錯上，配合不良事件發生原因的分析，尋求品質管制的方法。

### 3. 可與多媒體開發工具或其它開發系統做整合

本文提出的工具，可以與其它外部的系統或開發工具做整合，將系統內排除已知品質不良事件的結果，對外輸出，以比對其開發工具或開發系統的素材清單，是否有品質不良事件發生。

## 1.3 研究方法

### 1. 相關研究及文獻探討

多媒體教材的製作過程若要被管制，就要先能夠分解教材中的各個素材，一份多媒體教材，實際上就可以視為「素材」的集合體；而從過去的研究及文獻，提供了我們一些基礎，可以將「素材」和「教材」的關連分析出來，進而形成一個directed graph，我們稱之為MLCs Directed Graph (MDG)。

### 2. 品質管制方法及設計

教材拆解成 DAG 後，就有點(Node)和關連(Directed Edge)的關係產生。本研究以 DAG Graph 為基礎，所以必需保證系統中的的所有點和關連維持沒有遞迴(Acyclic)的發生，並列出可能發生的品質不良因子，借以 DAG 的演算法來排除。在舊有 MDAG 不足處，再加上其它點(Node)與點目的關連種類，借此來排除更多的品質不良事件。

## 1.4 章節概要

本論文共分四個部份，其內容簡述如下：第一章「緒論」，說明本論文研究機動的描述與希望達成之研究目的，並對論文章節做簡單的介紹。第二章「文獻回顧」，探討與本研究相關的背景知識與文獻，如多媒體教材品質、SCORM的標準及MDAG定義，做一完整的定義及探討。第三章「多媒體教材品質管制模型設計」，探討分析多媒體教

材製作的過程中，常發生的品質不良事件，以作為後續系統提供品質不良偵測及自動化品管的建立。第四章「系統設計與實作」，介紹系統以Java為基礎，利用物件導向(Object Oriented)的技術，架構在Java EE的web平台上。第五章「結論與未來研究方向」，說明本研究的貢獻及結論，並對未來研究方向提出建議。



## 二、文獻探討

本文的目標是針對多媒體教材製作過程中，條列其常見的品質不良事件，並將教材中使用的素材有條理的集合、關連起來，將其關連以Directed Graph的方式表示，記錄在系統中管理及共享；冀望以Graph現有的演算法，加上其它輔助的關連，可以達成自動化的品質不良事件排除。

### 2.1 多媒體教材的品質認證

本研究的主題在於多媒體教材品質，因此先了解一下國內對於多媒體教材的品質認證。國內關於數位教材、多媒體教材的品質控管，目前是由「經濟部工業局」的「數位學習品質認證中心」[2]所主導。

其目的主要針對教學者支援、學習者支援、學程發展、課程設計、教學歷程、機構性支援、評鑑及科技八大面向檢核(圖 1)，以確保企業員工或個人所接受之數位學習服務具備一定的品質。



圖 1 · 八大面向檢核圖

八大面向的檢核中，品質認證中心有八大規範，來確保其品質，其規範如下：  
規範一、學習者支援:須能及時提供課程訊息、線上支援與協助。



- \* 檢核重點1.1：學習者可以在線上獲得關於課程的訊息。
- \* 檢核重點1.2：提供學習者有關線上學習系統的訓練課程。
- \* 檢核重點1.3：提供學習者支援與協助。

規範二、教學者支援:須能提供線上教材製作與課程實施的協助。

- \* 檢核重點2.1：提供教學者如何使用線上學習系統的訓練課程，並隨時提供線上協助。
- \* 檢核重點2.2：協助教學者將補充教材轉成線上教材。
- \* 檢核重點2.3：協助教學者處理學習者的問題。
- \* 檢核重點2.4：提供教學者間的互動機制。

規範三、學程發展:需具系統性、完整性、一致性。

- \* 檢核重點3.1：學程發展須對學程目標與結構有系統化的分析、設計與檢討。
- \* 檢核重點3.2：學程中各課程內容需符合數位教材品質規範。
- \* 檢核重點3.3：學程需具備的架構，且導入數位化之課程份量需合理。

規範四:課程設計、需對教學目標、教材內容、活動設計、與評量有系統化的分析、設計、與檢討。

- \* 檢核重點4.1：課程內容符合數位教材品質規範。
- \* 檢核重點4.2：課程內容具備完整的架構，且導入數位化之課程份量須合理。
- \* 檢核重點4.3：課程設計應檢討與更新。

規範五、教學歷程:需對學生的學習歷程與表現有適切的記錄與回應。

- \* 檢核重點5.1：協助學習者與教學者，學習者與學習者間之互動。
- \* 檢核重點5.2：對於學習者的問題或作業(練習)須適時給予回饋。
- \* 檢核重點5.3：能記錄與分析學習者的線上學習歷程。

規範六、機構性支援:需具備建全的發展計畫、人力資源、與管理機制。

- \* 檢核重點6.1：有數位學習發展之願景與計畫。
- \* 檢核重點6.2：數位學習之營運須有人力資源。

\* 檢核重點6.3：數位學習之營運須有管理機制。

規範七、科技:需提供完善的使用者端、伺服器端、與網路的建置。

\* 檢核重點7.1：提供網路連線與頻寬。

\* 檢核重點7.2：訂定使用者電腦規格。

\* 檢核重點7.3：學習平台具有相容性。

\* 檢核重點7.4：有軟硬體設備的維護支援。

規範八、評鑑:對人員、課程、與系統需有完整的評估。

\* 檢核重點8.1：數位學習相關人員或部門服務的評鑑。

\* 檢核重點8.2：課程評鑑。

\* 檢核重點8.3：系統與支援工具的評鑑。

其認證法分為，【單元認證】申請公司應符合數位學習服務品質以認證之規範一、二、七、八等規範，即可提出數位學習服務(單元)認證之申請。【課程認證】申請公司應符合數位學習服務品質認證之一、二、四、五、七、八等規範，即可提出數位學習服務(課程)認證之申請。【學服認證】申請公司符合數位學習服務品質之八項規範，即可提出數位學習服務(學程)認證之申請。

其規範的認證等級共分為三個等級：A級認證，滿足這個認證等級將能符合數位學習服務使用者之基本需求。AA級認證，滿足這個認證等級將有助於數位學習服務使用者提升學習效率。AAA級認證，滿足這個認證等級將非常有助於數位學習服務使用者學習效率的提升，並足以做為典範[3]。

## 2.2 SCORM

要將多媒體教材的素材有條理的模組化並關連起來，才能將之轉換成一個Directed Graph。在此介紹SCORM不是要製造符合SCORM標準的教材，而是取其長處；因為SCORM是以教材的溝通介面標準化，在不同平台之間傳遞教材單元，同樣也必需面對模組化的課題，而它的做法將會是本研究要將教材模組化，很有價值的參考目標。

Sharable Content Object Reference Model (SCORM) [4]是由美國Advanced Distributed Learning (ADL) 所提出來的標準，其目的是定義一種界面，可以在符合SCORM標準的 Learning Management Systems (LMSs) 中可以交換分享，而Sharable Content Objects (SCOs) [10,11]為其分享的最小單位。

多媒體教材的特色是，可以與學員互動，因此它有著和寫程式類似的特性，有什麼刺激就會有什麼反應，只是它的反應都是連結著一個或多個素材(如：圖片、動畫、聲音…等等)。記錄刺激與反應的，免不了就是一段小程式，最終都必需在一個有運算力的裝置(如：PC、筆記型電腦、PDM…等等)上執行；在同一份教案中，在不同執行介面上(如：Flash[7]和編輯手[8])記錄刺激與反應的小程式或許會因為執行界面的不同而有差異，但教材擁有的素材關係則大同小異。

### 2.2.1 Asset

Asset，素材，它是多媒體教材的基本建構元件；它是一個電子檔，例如：文字檔、圖片檔、聲音檔、影像檔…等等(如圖 2)。

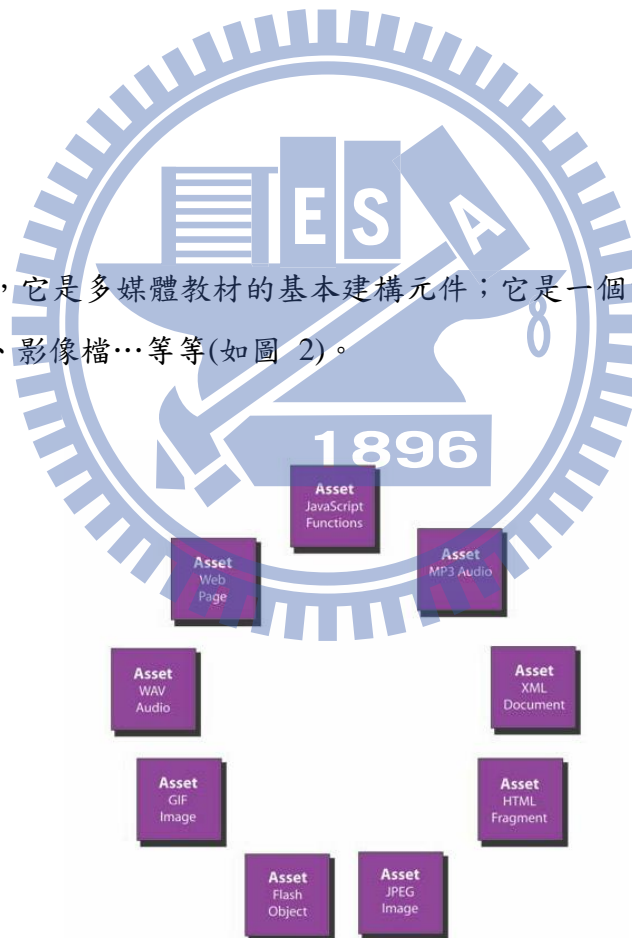


圖 2 · Examples of Assets

(SCORM® 2004 4th Edition Content Aggregation Model (CAM) Version 1.0)

## 2.2.2 Sharable Content Object (SCO)

一個SCO可以是一件或多件素材(Asset)的集合，同時也是一個可以被執行的，最小單位的學習資源，而這個學習資源可以在SCORM執行環境(SCORM RTE [11])下，在學習平台系統(LMS)中溝通共享(參考圖 3)。它和一般素材，即電子檔，最大的不同點是，它可以在不同學習平台(LMS)中共享，而且它符合Institute for Electrical and Electronics Engineers (IEEE) ECMAScript Application Programming Interface for Content to Runtime Services Communication standard [9]的規範。

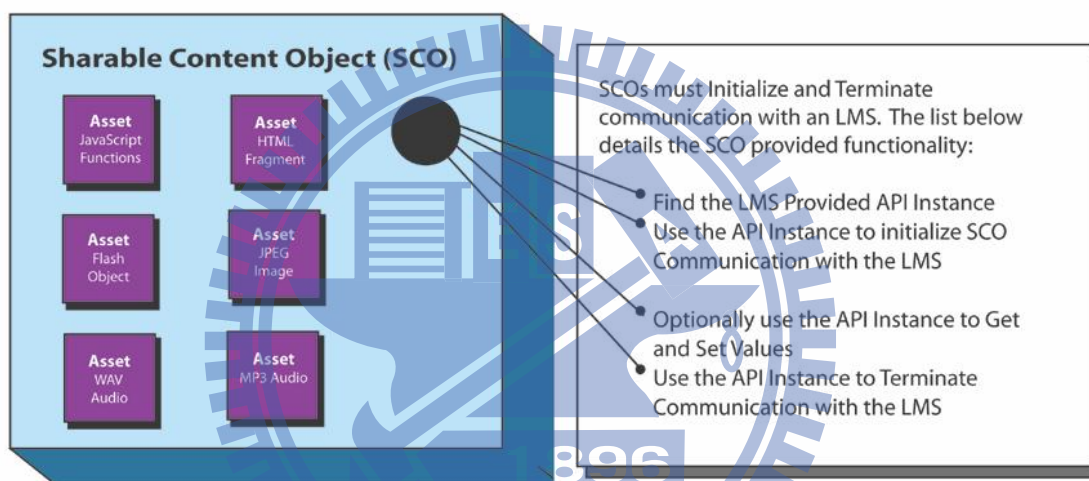


圖 3 : Conceptual Makeup of a SCO

(SCORM® 2004 4<sup>th</sup> Edition Content Aggregation Model (CAM) Version 1.0)

## 2.2.3 Activities

一個學習活動(Learning Activity)可以視為多媒體教材中(圖 4)，一個有意義的建構單元，但敘述的比較不嚴緊；概念上來說，就是學員可以在這個單元中，經歷一段學習的流程。一個學習活動可能只是單純提供一個學習資源(一個SCO或只有一個素材)給學員，也有可能是幾個子學習活動(sub-activities)的組成。

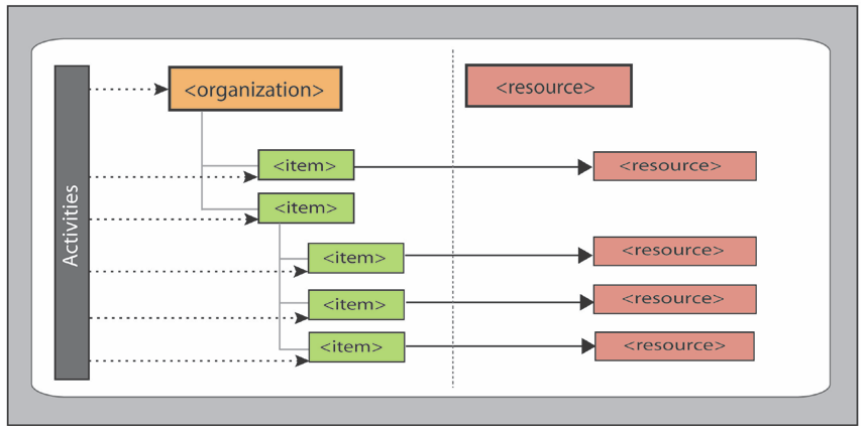


圖 4 : Conceptual Representation of Activities  
(SCORM® 2004 4th Edition Content Aggregation Model (CAM) Version 1.0)

## 2.2.4 Content Organization

內文結構 (Content Organization) 它表示這份教案的學習地圖，這份學習地圖顯示了學習活動(Activities)和學習活動之間的關係。下圖是內文結構的示意(圖 5)：

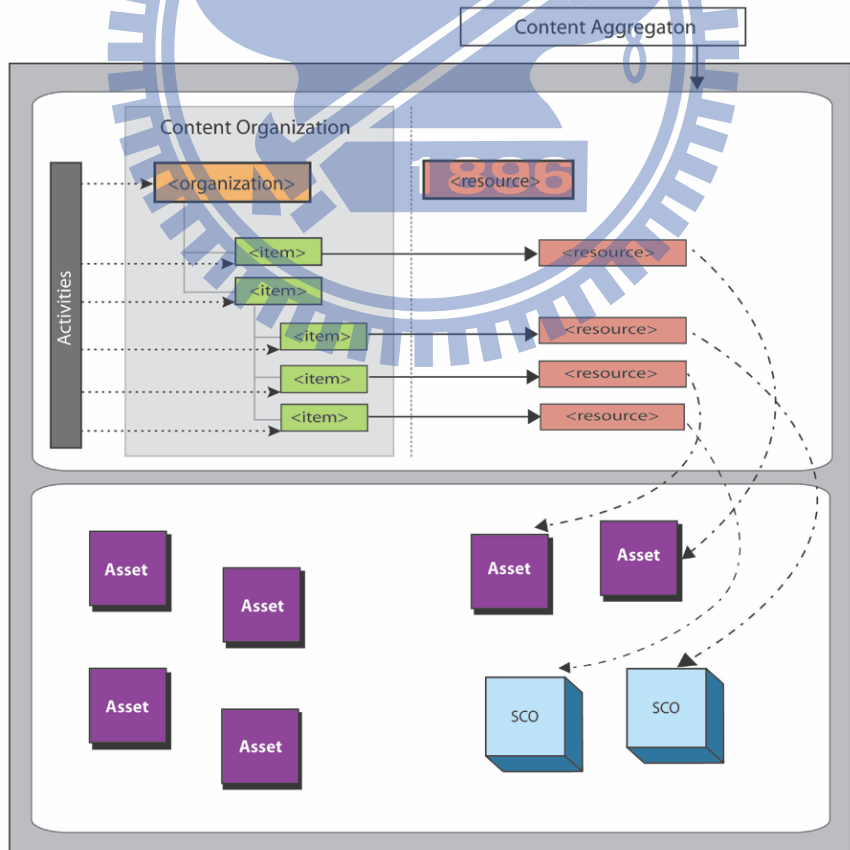


圖 5 : Conceptual Illustration of a Content Organization and a Content Aggregation  
(SCORM® 2004 4th Edition Content Aggregation Model (CAM) Version 1.0)

## 2.3 多媒體教材品質管制模型設計

軟體品質評量與製作流程的規範，最具代表性的是由美國卡內基美隆大學 (Carnegie-Mellon University, CMU) 軟體工程研究所 (Software Engineering Institute, SEI) 所提出的國際性標準能力成熟度整合模式 (Capability Maturity Model Integrated, CMMI) [10]。根據 CMMI 的規範，在多人協同的開發環境中，如果缺乏穩定的開發流程及良好的作業規範，會造成使用者所設計出來的元件或素材，在整合測試階段時，才發現元件不一致的衝突，或甚至是互斥現象，以致無法順利整合，而造成開發時程延誤、開發成本上升、顧客抱怨及開發成員士氣低落 [11,12]。

這些現象也會發生在多媒體教材的製造過程上，而此篇論文希望能學習 CMMI 的長處，建立一套多媒體教材品質管制的模型。

### 2.3.1 Multimedia Learning Contents Process Life Cycle

分析多媒體教材 (Multimedia Learning Contents, MLCs) 的開發及製作過程，可以歸納出一開發生命週期 (參考圖 6)；在其各個階段 (Phase) 中，需要規範各式文件規格，每一份文件用來表達其所對應的每一個主要工作結果；茲將其分成下列五個主要的階段：

1. 課程導入期 (RD)
2. 課程規劃期/單元腳本設計 (DM)
3. 課程規劃期/場景 UI 設計 (UI)
4. 課程製作期/素材製作 (MM)
5. 課程製作期/教材製作 (CM)

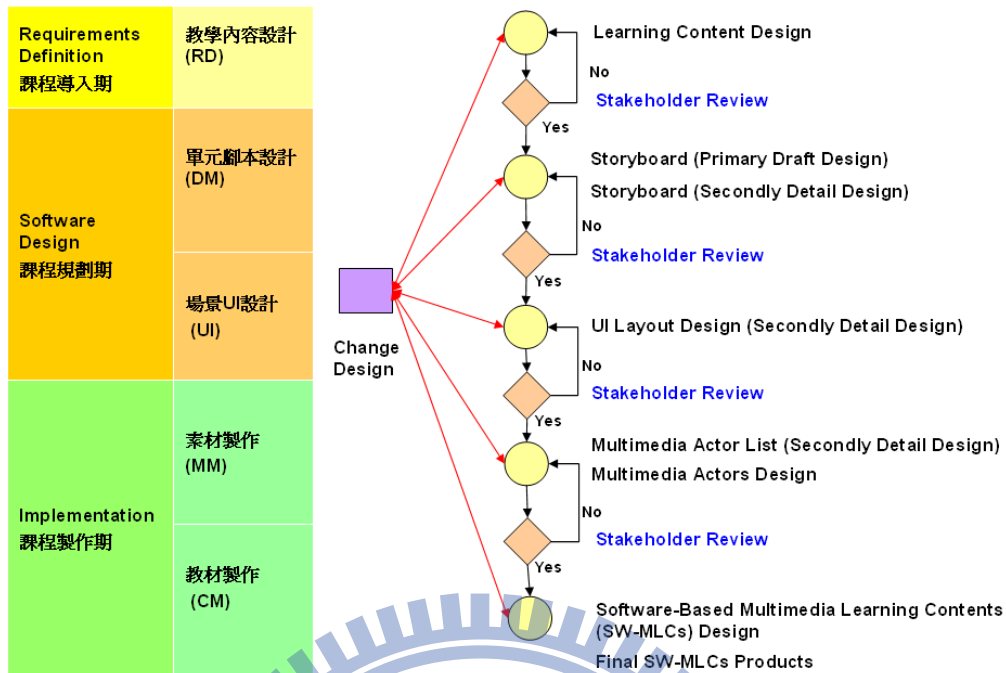


圖 6：多媒體教材開發流程(MLCs process flow chart)

在課程導入期(RD)主要的工作是根據教學目的、教學對象與教學環境的需求，可使用原創設計、規劃或是選定現有適當的學習教材或教案，作為開發多媒體教材的原始教學內容與依據。

在課程規劃期/單元腳本設計階段 (DM) 主要的工作是設計多媒體教材的分鏡腳本，此分鏡腳本是用來說明多媒體教材內容所要表達意境或觀念。腳本內的劇情同時需要描述這些多媒體素材之間與場景的空間及時間的關係。

在課程規劃期/場景UI設計階段(UI)主要的工作是設計多媒體教材的使用者界面，美術編輯者需要根據多媒體教材的分鏡腳本每一幕場景的需要，設計多媒體教材內容所需要使用者界面。主要工作包括主瀏覽畫面的使用者界面設計與美工設計，和每一幕之間的瀏覽方式及美工設計。

在課程製作期/素材製作階段 (MM)主要的工作是多媒體教材內容所需要的多媒體素材製作。多媒體教材開發人員需要根據多媒體教材的分鏡腳本劇情的需要，設計腳本中每段劇情所需要使用不同資料格式的多媒體素材，來協助了解教材內容的意境表達或觀念。常見的多媒體資料格式種類包括聲音、影片、圖片、動畫及文字。

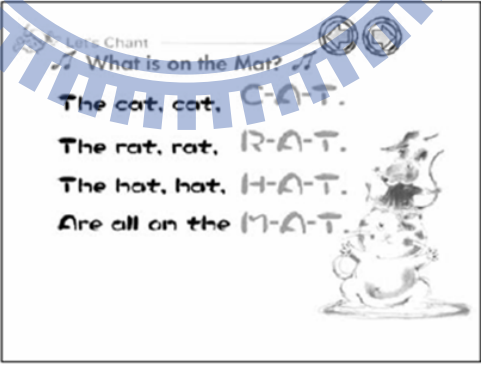
在課程製作期/教材製作階段 (CM)主要的工作是多媒體教材內容的編輯及製作，多媒體教材開發人員需要使用多媒體編輯軟體或程式開發工具，根據分鏡腳本的劇情描述，將多媒體素材及多媒體素材、多媒體素材及場景之間的空間及時間的關係，製作成一幕幕的多媒體教材內容。而此步驟所使用的多媒體編輯軟體或程式開發工具，對於多媒體教材的最終成品有關鍵性的影響。

### 2.3.2 MLCs Directed Acyclic Graph (MDAG)

由於需求變更(Requirement change)在多媒體教材的設計、製作過程中都避免不了。多媒體教材可視為被「賦予行為」的「素材」集合體；為了共享這些設計好的素材、有效管理這些素材及其行為，可以依其需求，將其邏輯群組建立，下表是一份自然語言需求的範例，對應的多媒體教材的編輯工具為智勝國際編輯手[8]。

參考本實驗室之前的研究成果[13]，可以了解到，一個多媒體教材的最小可執行單位(以編輯手為例)，可以自然語言敘述其需求，直接將之整理成如下(表 1)：

表 1：自然語言需求(Requirement in Natural Language)-  
以編輯手製作多媒體教材的分鏡腳本(Storyboard)

Scene: Scn1	
User Interface: UI1	
	
Natural Language Script	Multimedia Actor
<b>Script 1:</b> 播放鈴聲，Let's Chat. What is on the Mat? 前後方向按鈕的圖片，隨開場場景出現。	<b>Act 1 [Audio 1]</b> <b>Act 2 [Pic 1]</b>
<b>Script 2:</b> 播放鈴聲，女生說 "Page sixteen. Let's chat. What is on the Mat?"	<b>Act 2 [Pic 1]</b> <b>Act 3 [Audio 2]</b>
<b>Script 3:</b> 畫面出現對話 男生說 "The cat, cat, C-A-T." 女生說 "The rat, rat, R-A-T." 男生說 "The hat, hat, H-A-T." 男生和女生同時說 "Are all on the M-A-T."	<b>Act 1 [Audio 1]</b> <b>Act 2 [Pic 1]</b> <b>Act 4 [Audio 3]</b>
<b>Final Product: CM1</b>	



而由以上的分鏡腳本，可以找出其相依關係，例如：場景1(Scn1)裡有3個腳本 (Script1/2/3)；腳本1(Script 1)裡，有兩個演員(Act 1/2)。依此關連類推，可以得到下列的 MLCs Directed Acyclic Graph (MDAG) [13] (圖 7：以編輯手製作多媒體教材的分鏡腳本 (Storyboard)展開的MDAG)：

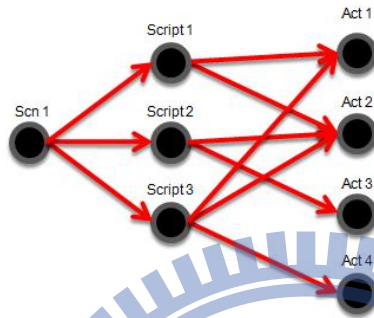


圖 7：以編輯手製作多媒體教材的分鏡腳本(Storyboard)展開的 MDAG

在MDAG中圖形內所使用節點的類型如下：

- 分鏡節點(Scene nodes)：  
此節點代表在分鏡腳本(Storyboard)中，多媒體教材內的一個獨立分鏡畫面作業單位。在此節點必須完成紀錄相關劇情及多媒體素材的關係性。
- 劇情節點(Script nodes)：  
此節點代表在分鏡腳本(Storyboard)中，多媒體教材內的一個獨立分鏡中，多媒體素材所需要演出的劇情。在此節點必須以自然語言說明多媒體素材依照劇情的時間與空間的關係性進行表演行為。
- 演員節點(Actor nodes)：  
此節點代表在分鏡腳本(Storyboard)中，多媒體教材內的一個獨立分鏡中，依照劇情的情節需要，所使用的多媒體素材演員。在此節點必須完成多媒體素材的設計工作。

其中的節點(nodes)和節點之間的關連就是有方向性的邊(directed edges)，它代表「有」關連，例如，Scn 1「有」Script 1, Script 2和Script 3三個子節點；自然語言中的表述，就是場景1(Scn1)裡有腳本1/2/3(Script1/2/3)三個腳本。

用MDAG的方法，就可以將多媒體教材中，分鏡腳本以下的素材，以腳本分鏡表的邏輯群組拆解，用節點及邊的方式描述。

### 2.3.3 MLCs Bidirectional Traceability Matrix

在本參考文獻的原始設計中，並沒有利用DAG的特性來做相依性關係的偵測或判斷，而是利用節點和節點之間的關連，形成的相依矩陣(Dependency Matrix)[14]，整理出一個多媒體教材正向相依矩陣(MLCs Forward Dependency Matrix)，簡稱MFDM。

由於需求變更(Requirements Change)會發生在多媒體教材開發生命週期(MLCs Process Life Cycle, MPLC)的任一階段，為了避免不一致性(Inconsistency)品質問題的產生，必須掌控在多媒體開發過程中，因為修改劇情、場景或是素材時，對於最終成品的影響。透過前一節的分鏡腳本及其所代表的MDAG，可以建立一個相依矩陣(Dependency Matrix)，來記錄素材及其邏輯元件之間的相依性關係(Dependency Relationship)。用前一節的例子(表 1及圖 7)來說，可以定義一個多媒體正向相依矩陣(MFDM)，如表 2。

表 2：MLCs Forward Dependency Matrix (MFDM)

Source	Target							
D.Value	Script 1	Script 2	Script 3	Act 1	Act 2	Act 3	Act 4	DOD
Scn 1	1	1	1	0	0	0	0	3
Script 1	0	0	0	1	1	0	0	2
Script 2	0	0	0	0	1	1	0	2
Script 3	0	0	0	1	1	0	1	3

若直行的節點存在橫列節點的相依關係，也就是有一個邊(directed edge)由直行的節點指向橫列的節點，則於正向相依矩陣對應的相關欄位設定值為1，若不存在相依關係，則相關欄位值為0。其主要意義可由上表中的Dependency-Out-Degree (DOD)[15,16]來說明其特性，DOD的數值大小代表當多媒體教材開發生命週期的任一階段的需求設計改變時，所影響下一階段的設計工作的範圍大小。將上面的多媒體教材正向相依矩陣(MFDM)作轉置(transpose)，可以得到多媒體教材反向相依矩陣(MLCs Backward Dependency

Matrix, MBDM), 如表 3。

表 3 : MLCs Backward Dependency Matrix (MBDM)

Target	Source				
D.Value	Scn 1	Script 1	Script 2	Script 3	DID
Script 1	1	0	0	0	1
Script 2	1	0	0	0	1
Script 3	1	0	0	0	1
Act 1	0	1	0	1	2
Act 2	0	1	1	1	3
Act 3	0	0	1	0	1
Act 4	0	0	0	1	1

反向相依矩陣其主要意義可由上表中的Dependency-In-Degree (DID)[15,16]來說明其特性，DID的數值大小代表當多媒體教材開發生命週期(MPLC)的任一階段的設計需求，受到來自上其它節點影響的數量多寡。

由此二矩陣所記錄的相依性關係(Dependency Relationship)的紀錄，可以知道當多媒體教材開發生命週期的任一階段的需求設計改變時，所影響下一階段的設計工作的範圍大小，以及任一階段的設計需求，受到上一階段的設計工作影響的範圍大小。除此之外導向非循環圖形(Directed Acyclic Graphs, DAG)具有方向性以及不會形成循環的特性[17]，可以說明在多媒體教材開發生命週期(MPLC)的過程中，需要按照各工作流程階段先後順序設計與製作的特性，同時又可表現出每一工作階段之間的相依性關係(Dependency Relationship)。因此MDAG適合作為設計具有雙向可追蹤性(Bidirectional Traceability)以及不一制性偵測(Inconsistent Detection)機制的多媒體教材的基礎模型。

### 三、多媒體教材製作過程品質管制方法

根據第二章的相關文獻研究，本文希望以Graph為基礎，利用現有的Graph理論及方法，尋求一套品質管制的方法。能夠在教材設計階段，於系統內自動化檢核已知的錯誤模式，以避免此種類別的錯誤發生；而這樣的素材群組結構，也可用於製作階段，匯出其素材結構以供比對，避免素材誤植等品質不良事件的發生；未來更可能與其它系統、開發工具做橋接，而對其成品做最終的檢核及比對，做成報表，複查其品質不良事件的存在。

多媒體教材的最終成品，可視為兩個部份：一是素材的組合，另一是素材間的邏輯行為。而本文的重點是在於素材的組合上。有了這樣多媒體教材與素材間的邏輯群組關連，才有辦法將這些素材群組，投射到Directed Graph (Digraph)[18]上，並以Graph現有的理論及方法，來檢示其可能的錯誤；錯誤一旦可以自動化地被標示出來，就代表我們可以避免該品質不良事件的發生；減少品質不良事件的發生，就是品質的提升。

本章將分成兩大部份，第一部份在說明素材與多媒體教材的邏輯群組對應關係以組成Digraph，本文稱之為Multimedia Directed Graph (MDG)；第二部份則是找出已知品質不良事件的樣本，應用Digraph的理論及方法，來排除這些品質不良事件。

#### 3.1 多媒體教材與素材的關連：MDG

傳統的多媒體製作過程中，並沒有有效的方式來管理其素材，而素材與成品或半成品之間的關連，也都靠人工的方式來做檢核及維護，但這樣的方法效率不佳且易出錯。所以本文擷取了ADL提出的SCORM中Content Organization做教材章節的邏輯關連，加上素材與教材執行單元的邏輯群組關連的Digraph，再佐以一些新加入的群組關係(為了達成其品質不良事件排除而需要的邏輯關係)，本研究將它稱做：MLCs Directed Graph (MDG)。

如果素材散亂沒有管理，很容易在製作時發生誤植，所以，要管理多媒體教材中的元件，首要是把元件和多媒體教材之間的關聯找出來，就像汽車藍圖一樣，也需建立教材元件關聯圖(如圖 8，汽車零件關連圖及教材元件關聯圖)。

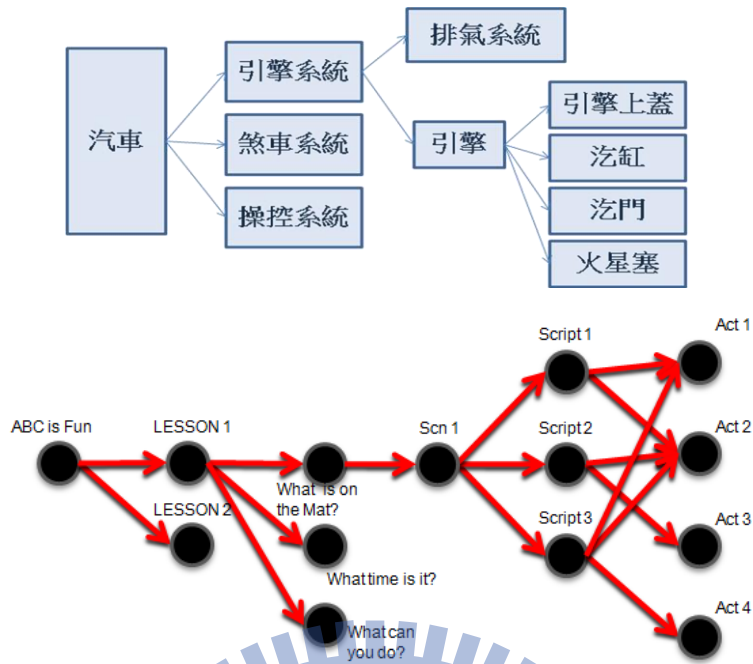


圖 8：汽車零件關聯圖與 MDG

### 3.1.1 Content Organization

教材的章節部份，可以依ADL的SCORM中的Content Organization，將該教材依章節拆到其可執行的最小單元(即SCO)。以下面這份教材(圖 9)中的章節：

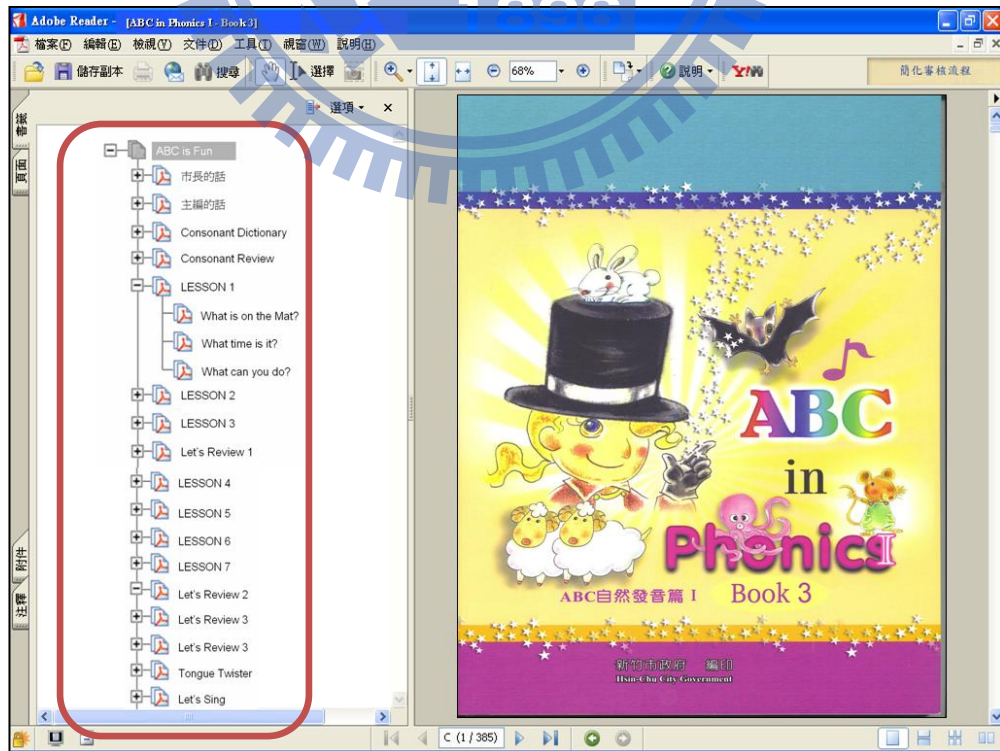


圖 9：傳統教材章節關連

依其章節的階層關係，可以直覺地，依SCORM的定義製作關連圖。以此例子來說，本教案名稱為「ABC is Fun」，所以根節點(root node)就是「ABC is Fun」。它底下有類似前言的四個段落，分別是「市長的話」、「主編的話」、「Consonant Dictionary」及「Consonant Review」，還有正式課程內文的部份從「LESSON 1」、「LESSON 2」…等等，所以根節點下有著上面敘述對應的節點(本文稱之為子節點)，並以Directed Edge來連接，方向是由根節點至其子節點。再以「LESSON 1」為例子，其下有「What is on the Mat?」、「What time is it?」及「What can you do?」三個小節，依前述方法，用Directed Edge關連起來，方向由「LESSON 1」節點到「What is on the Mat?」、「What time is it?」及「What can you do?」三個子節點。

整份教材都依前述方法繼續對「LESSON 2」、「LESSON 3」…等等，逐一分析，即可對應出一個Digraph的結構(圖 10，圖中藍色三角型代表系統中的一個sub Digraph)。

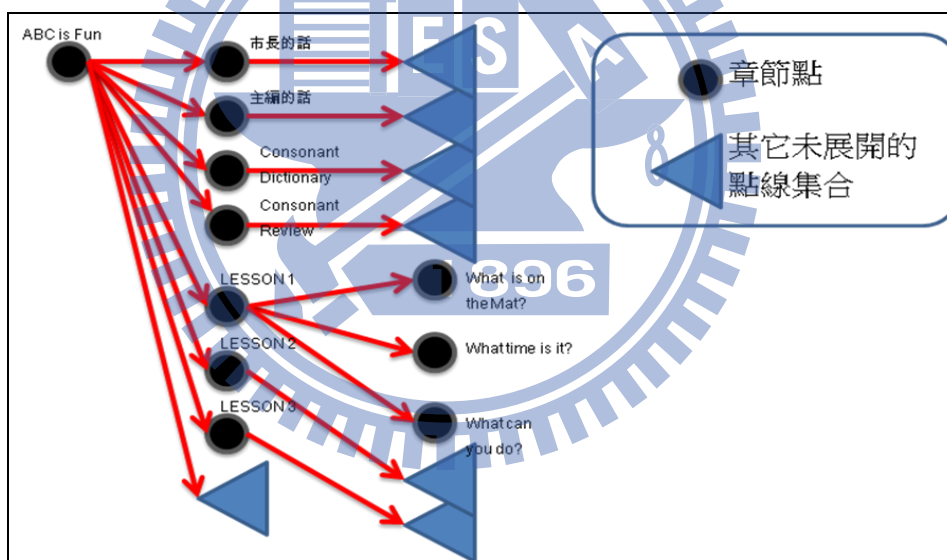


圖 10：傳統教材章節對應到 DAG

### 3.1.2 多媒體教材及其自然語言需求表


我們必需設法將多媒體教材轉換成Digraph，要達成這個目標，我們就必需先分析多媒體教材；而多媒體教材的開發過程中[19]，在設計階段即可分析其需求，進而建立其需求表。根據自然語言的需求表述，我們希望可以分析得出分鏡腳本，以「What is on the

Mat?」為例。這一小節的自然語言需求如下：

播放鈴聲及人聲說「Let's chat. What is on the mat?」之後，前後方按鈕圖片及相關圖片場景出現。圖片及場景出現完畢後，再次播放鈴聲，女聲說「Page sixteen. Let's chat. What is on the mat?」。此後畫面要出現對話，男聲說「The cat, cat, C-A-T.」，女聲接著說「The rat, rat, R-A-T.」，男聲接著說「The hat, hat, R-A-T.」最後男女聲同時說「Are all on the M-A-T.」。

依上面的需求，可以將之分成三段，製成自然語言需求表，如表 4。(註：一如寫程式一般，這不是唯一的表列法，不同的作法也可以製做出一樣的多媒體教材呈現。)

表 4：自然語言需求範例

Scene: Scn1	
User Interface: UI1	
	
Natural Language Script	Multimedia Actor
<b>Script 1:</b> 播放鈴聲，Let's Chat. What is on the Mat? 前後方向按鈕的圖片，隨開場場景出現。	<b>Act 1 [Audio 1]</b> <b>Act 2 [Pic 1]</b>
<b>Script 2:</b> 播放鈴聲，女生說 "Page sixteen. Let's chat. What is on the Mat?"	<b>Act 2 [Pic 1]</b> <b>Act 3 [Audio 2]</b>
<b>Script 3:</b> 畫面出現對話 男生說 "The cat, cat, C-A-T." 女生說 "The rat, rat, R-A-T." 男生說 "The hat, hat, H-A-T." 男生和女生同時說 "Are all on the M-A-T."	<b>Act 1 [Audio 1]</b> <b>Act 2 [Pic 1]</b> <b>Act 4 [Audio 3]</b>
<b>Final Product: CM1</b>	

### 3.1.3 素材邏輯群組及 Digraph

本節將說明邏輯群組的定義及素材Digraph如何型成。由於多媒體教材它也有類似程式性質的行為，例如按鈕後觸發一連串的動作，又或者滿足某一種條件後，觸發一組動作；這些行為，都是需要編撰簡單程式來滿足。多媒體教材中的邏輯群組可為成三階層：

(1) 分鏡(Scene)：

它像是一個容器，裡面裝著一組以上的劇本及如何觸發它們。

(2) 劇本(Script)：

它像是一組程式的巨集，記錄著一個或一個以上被觸發者的行為。被觸發者可以是演員或別的劇本。

(3) 演員(Actor)：

通常是個素材檔，可能是圖片、聲音、動畫...等等。

以上的三個階層，是來自於需求表，配合媒體教材的分析設計而獲得。例如，由表4可得知，【分鏡一】中有3個劇本；而【劇本一】中，先是播放鈴聲，後而場景圖片出現，所以有兩個演員：鈴聲(聲音演員)及場景圖片(圖片演員)，而其觸發方式為【分鏡一】一開始就觸發【劇本一】，而劇本中的兩個演員一個為「播放」(鈴聲)及「出現」(場景圖片)。依據以上的分析方法，我們可以得到下面的資訊：

(1) 【分鏡一】下有【劇本一】、【劇本二】及【劇本三】三個劇本。

(2) 【劇本一】有兩個演員分別是【演員一】聲音演員及【演員二】圖片演員。

(3) 【劇本二】有兩個演員分別是【演員二】圖片演員及【演員三】聲音演員。

(4) 【劇本三】有三個演員分別是【演員一】聲音演員、【演員二】圖片演員及【演員三】聲音演員。

綜合以上資訊，給予代號：【分鏡一】為Scn1；【劇本一】、【劇本二】及【劇本三】分別為Script1、Script2及Script3；【演員一】到【演員四】分別為Act1、Act2、Act3及Act4。所以我們可以拆解分析得到下面的Digraph(圖 11)。



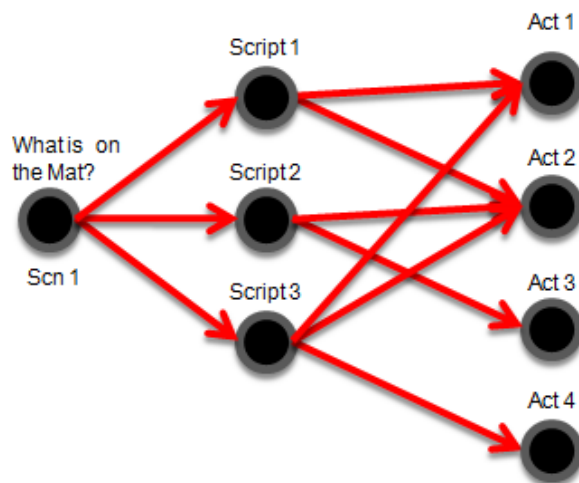
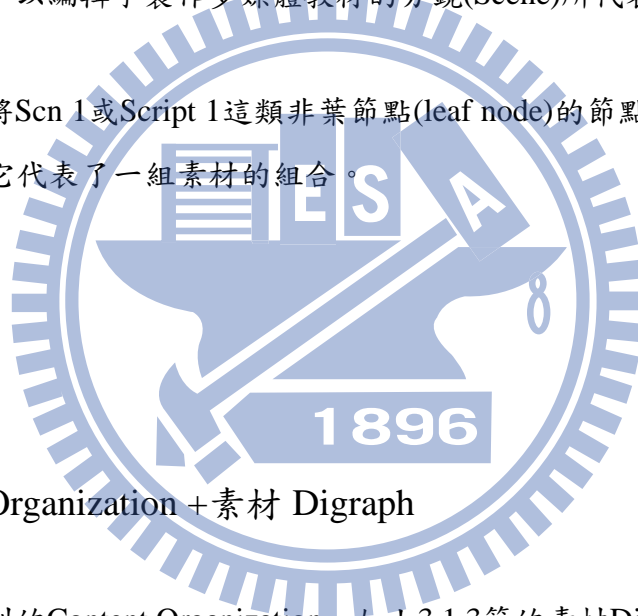


圖 11：以編輯手製作多媒體教材的分鏡(Scene)所代表之 MDG

其中本文會將Scn 1或Script 1這類非葉節點(leaf node)的節點，稱之為教材的「邏輯單元」或單元，它代表了一組素材的組合。



### 3.1.4 Content Organization + 素材 Digraph

將3.1.1節提到的Content Organization，加上3.1.3節的素材Digraph，可以將它們再加以串連；以此例來說，「What is on the Mat?」只有一個場景(Scene 1)，以「What is on the Mat?」為父節點，「Scn 1」為子節點，用Directed Edge關連起來。

如果依據上述方法一一串連，就可以形成整份教案完整的邏輯群組及素材關連Digraph，無論是系統內全部的Digraph，或是一份教材，或只是一個分鏡、一個劇本，本文都將之稱為MLCs Directed Graph (MDG)。以「ABC is fun」的教樣來看，其組成MDG的示意圖如圖 12。

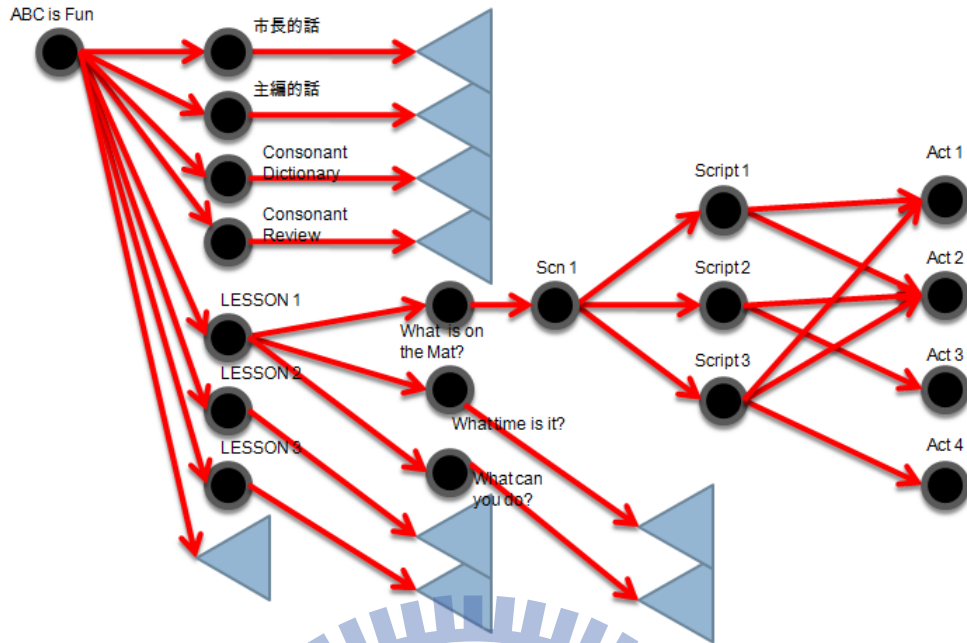


圖 12 : MLCs Directed Graph (MDG)

### 3.1.5 MLCs Directed Graph (MDG)

本研究目的是要利用Graph[20]的特性，以其現有的演算法，來排除可能的品質不良事件。MDG的邊(directed edge)，是由「擁有」的上對下關連所建立，依照前面的方法，我們可以得到一個多媒體教材所屬素材的關連graph，叫做MLCs Directed Graph (MDG)，如圖 13。

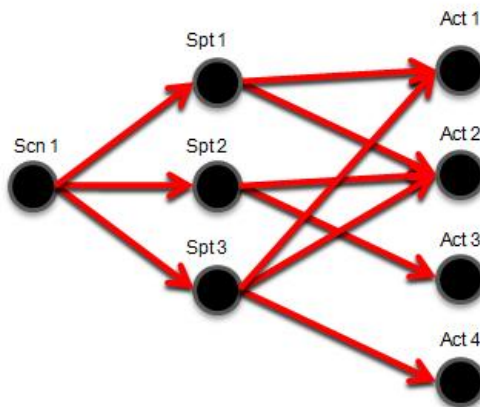


圖 13 : MDG

無論是一整份的教材所形成的digraph(如圖 12)，或者是場景以下的digraph (如圖 13，算是圖 12中digraph的sub digraph)、任何digraph的sub digraph，亦或者全系統中所有的digraph 集合，這一類的digraph都算是MDG，所以本文都會以MDG來稱之。

比較常出現的幾個節點，本研究有習慣的縮寫字頭加上一個數字，代表系統中的唯一示別鍵值，如此看到鍵值就可以知道它的種類，如此將有助於後面的討論，但在系統實作時，不一定需要和此規定一致，只要唯一就可以。例如分鏡(scene)的鍵值通常為Scn字頭後面加上一個數字，劇本(script)的縮寫為Spt，演員(actor)的字頭為Act。

### 3.1.6 reversed MDG (rMDG)

在某些狀況下，通常是為了追溯某節點的父節點，我們就需要將所有邊(directed edge)都反轉，才可以達成這些目的，原來MDG中的所有邊都做反轉後的digraph，本研究稱之為reversed MLCs Directed Graph (rMDG)，下圖 14為上圖 13的rMDG。

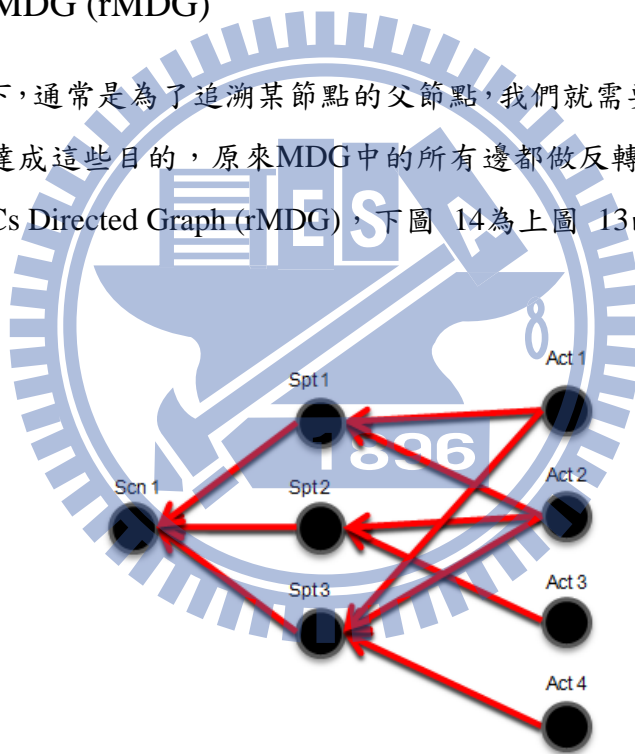


圖 14 : reversed MLCs Directed Graph( rMDG).

如同MDG，系統內任何rMDG的sub digraph或sub digraph的集合，都可以稱之為rMDG。

## 3.2 MDG 檢查法及可排除之品質不良事件

有了MDG可以記錄多媒體教材及其素材的關連，現在我們來看看，怎樣利用graph中現有的理論，來排除可能的品質不良事件，藉此提昇多媒體教材的品質。

### 3.2.1 MDG 可將散亂素材加以管理

傳統作法中，並沒有對素材加以管理，通常就是放在資料匣中，用人力去管理。但一份普通的多媒體教材，其素材非常多，而且因為不同的人製作，不同的檔案可能取相同的名字，或者檔名類似，這都會造成混亂而難以管理，很容易造成素材檔案被覆蓋、誤刪或遺失，使得教材在製作時可能會發生誤植或缺漏。

MDG可以記錄多媒體教材及其素材的關連，如圖 15，我們可以得知Scn1有用到4個演員，而它們分別對應到4個素材檔案。再透過系統來管理這些素材檔，無論存取修改都必需透過系統來操作，這樣作將比沒有任何管理來說好的多。

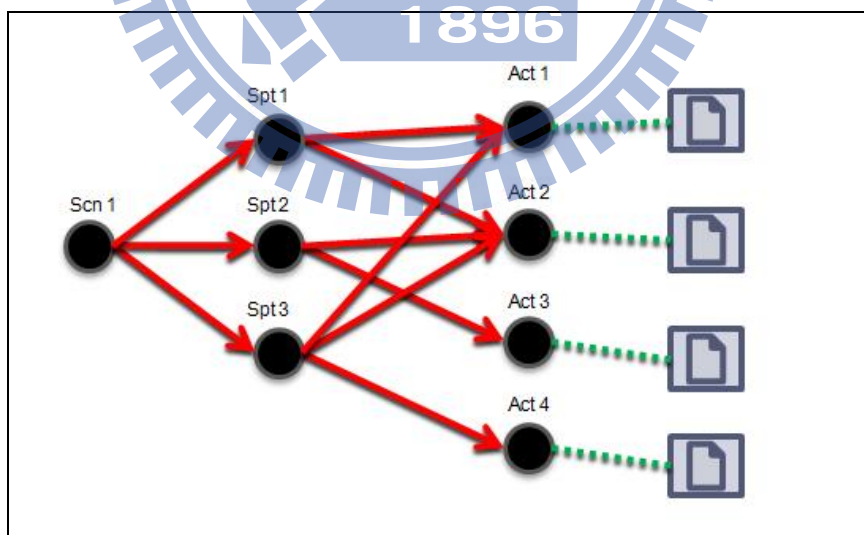


圖 15：MDG 及素材檔案對應示意圖。

### 3.2.2 容易對照相關素材的正確性

首先，我們先來看一下某個劇本裡所使用的圖片素材演員，如圖 16：

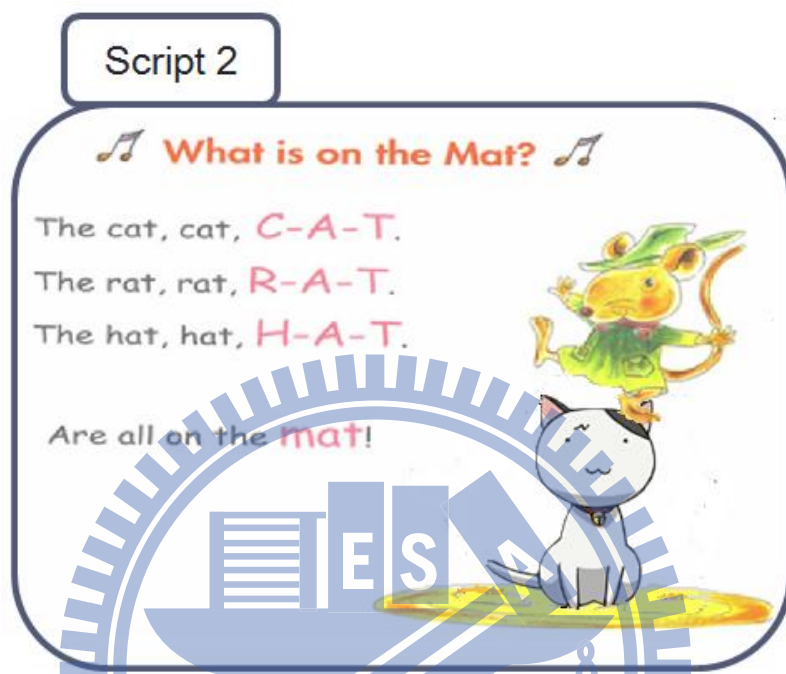


圖 16：某個劇本 Script 2 裡的圖片素材演員。

先注意一下Script 2所使用圖片中的貓，對照一下文字的部份有提到"What is on the Mat?"，而第一句便回答"The cat, cat, C-A-T."，由此可見，有一隻貓在墊子上，所以那隻白色黑耳朵的貓看起來也就合理。

但如果我們對照一下和它前後相關連的劇情：Script 1及Script 3，就會發現其中有不一致之處(見圖 17)。由圖 17可以發現，貓長的不一樣了!雖然都是貓，但貓變胖了；如果沒有特別的劇情引導下(例如說：貓減肥後又復胖)，這樣的狀況看起來就不太合理。因此在要製作成品時，我們可以比對設計時的MDG，來看看其中有沒有錯誤。

製作階段時，依其從系統中擷取素材的節點，可以得到一個製作的MDG；而設計階段也會產生一個設計時的MDG，兩相比較一下，就可以比對出節點有無不同之處(圖 18)。

【檢驗法】

步驟1 · 以製作成品節點為根節點(root)，以BFS找出所有可以找出所有與此節點相臨的節點，並存成一清單(list)。

步驟2 · 以同一節點的設計階段為根節點(root)，以BFS找出所有與此節點相臨的節點，並存成另一清單(list)。

步驟3 · 比較兩個清單內容是否一致，若一致則再以此清單內的節點為根節點，重覆此檢驗法，至到發現不一致為止。若直到所有節點檢查完，都沒有發現不一致則此根節點的製作有符合設計，即素材引用都正確。

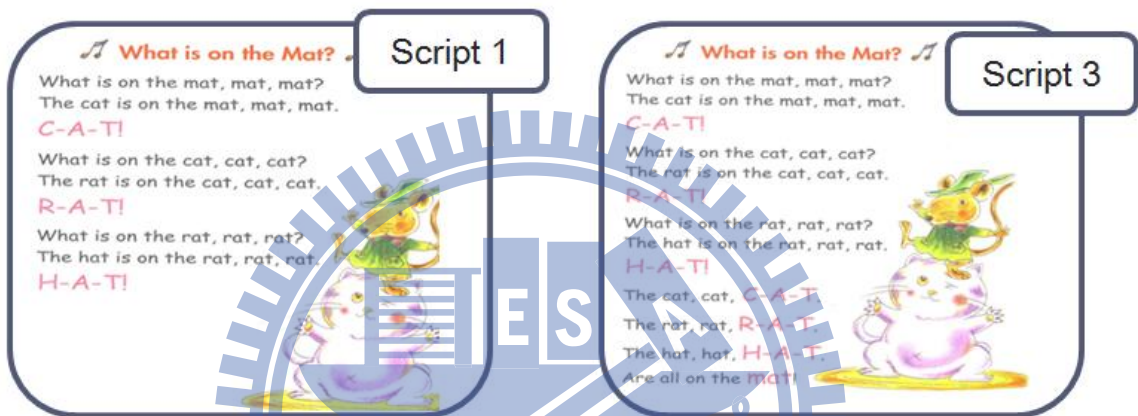


圖 17：劇本 Script 2 相關的前後兩劇本裡的圖片素材演員

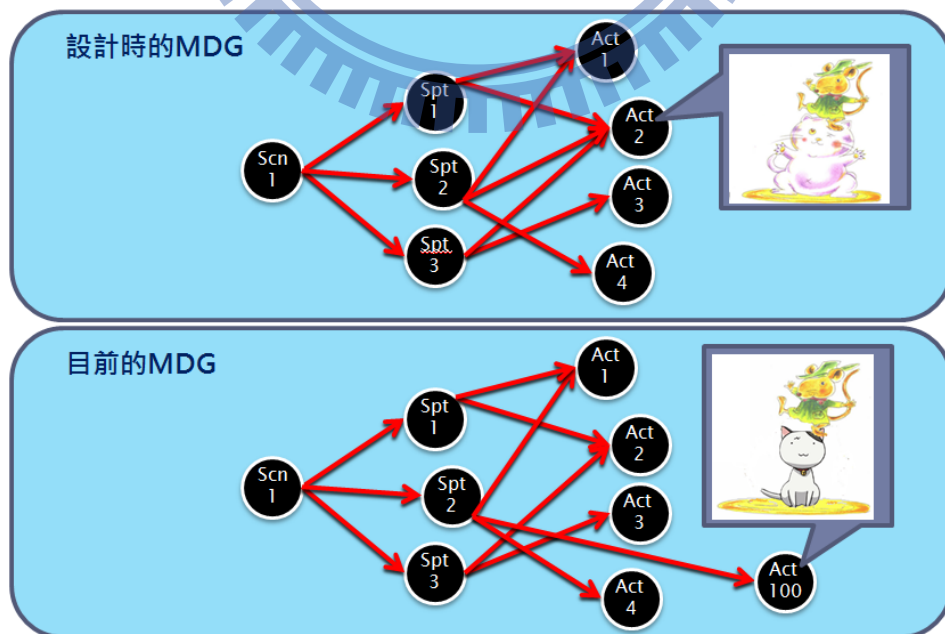


圖 18：與設計時的 MDG 比較

參考原設計時的MDG來看，三個劇本裡的貓應該都是同一個圖片演員(Act 2)，但顯然現在Spt 2所使用的圖片演員換了一個；這個圖用肉眼很容易就可以找出錯誤，但如果是要交給系統自動化執行，就必需要有一個可以讓電腦程式化執行的方法，這也是我們要利用Graph中現在演算法的原因。

本研究利用Graph中常見的演算法：Breadth-First Search(BFS)[21](BFS的演算法詳見[21]參考書目的22.2小節，p.531)來做逐層掃描的工具。系統在設計階段時，就會建立一組MDG，而依此MDG在系統內找出所需的素材來製作。而在製作時將其引用的素材匯整，也可得到一個階段製作成品的MDG；因為製作者的失誤，可能會引用錯誤的素材，但依此成品的MDG與設計時的MDG比對，就可以找出引用錯誤的素材。以上面的例子來說，想檢查Scn 1現在是否合於原始設計，我們便以Scn 1為起始節點(node)，向下做BFS的逐點掃描，便可以發現Spt 2目前所使用的 Act 100和當初設計時不同。因此，本研究用此方法，就可以自動化排除此類的品質不良事件。

### 3.2.3 素材整備檢查

將素材組成多媒體教材之前，必需檢查相關素材是否備齊，如果沒有，則有可能發生成品缺漏素材的狀況，而造成教材品質不良。

在系統中，除了會記錄著節點(node)與邊(directed edge)的關係，還可以記錄該節點(素材演員、劇本、分鏡…等等)是否已完成整備。已完成整備的節點，則其下層的所有子節點都需要是「完成」整備；例如圖 19中，如果要知道Scn 1是否已完成整備，可以它為起始節點，向下使用BFS逐點掃描，如圖所示，目前Act 2尚未完成整備(可能是還沒開始製作…等等任何原因)，所以Scn 1也不算完成整備。

#### 【檢驗法】

步驟1. 以欲檢查的素材做為根節點(root)，以BFS找出所有可以找到的路徑(path)及其經過的點。

步驟2. 檢查其所有葉節點(leaf)是否完成整備有素材檔案，若全部都有，則本節點已完成整備；反之則否。

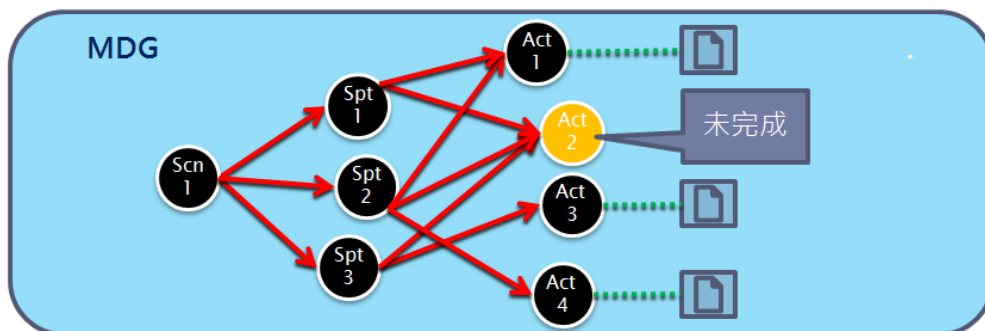


圖 19：MDG 素材整備檢查

傳統的作法只能靠人力一一檢核，不過有了MDG的幫助，將整份教材切分成比較小的邏輯群組，配合BFS就可以自動化、有系統地一一檢查，冀排除素材整備不全的品質不良事件。

### 3.2.4 週邊效應評估

教材有可能因錯誤要修正或有需要做改進時，就必需面臨修改所帶來的週邊效應。因為在系統內的素材或其它的邏輯單元節點，都是可以共用的，也就是說，可能有別的節點使用到它們，所以在修改時，需要一併通知這些有共用的父節點，來做週邊效應的評估，看看是不是需要一起改成新的單元，或者保留原來的狀況即可。如果週邊效應沒有評估，就有可能發生不一致的品質不良事件。

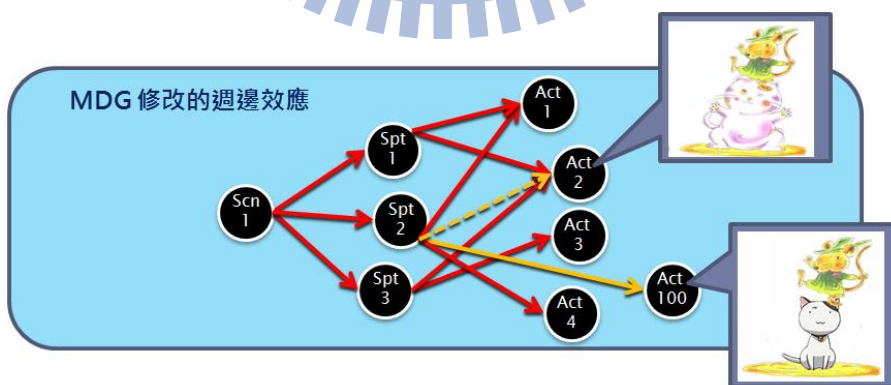


圖 20：MDG 週邊效應的發生

假設現在有一個需求，要將Spt 2下的Act 2換成Act 100(如圖 20)，而我們現在要找出需要做評估的節點。



### 【檢驗法】

步驟1·將原來的MDG做reversed，得到rMDG。

步驟2·再以欲修改的素材做為根節點(root)·以BFS找出所有可以找到的路徑(path)及其經過的點(node)，這些節點就是需要評估的點。

圖 21便是需要評估的節點。

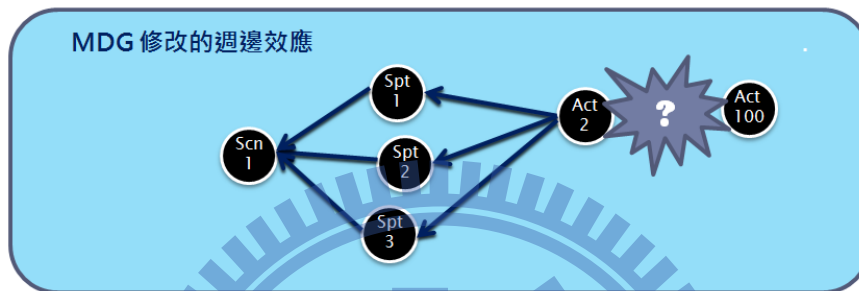


圖 21：週邊效應需評估的節點

而評估後的處置法就會因狀況不同而異。例如說，胖貓要換成瘦貓，而其劇情都沒有改變，則看來就是要一起把Spt 1, Spt 2和Spt 3的Act 2都一起換成Act 100才行。但如果因為劇情改成了有別隻瘦貓出現在Spt 2，後來胖貓又在Spt 3跑回來，則只要改Spt 2就可以。此處因為有自然語言需求的變更可能，所以需要人工來評估及判斷，但系統可以做到自動化找出需評估的節點並發出通知。

如果只到Scn 1的範圍(scope)不夠大的話，也可以向上多提升幾級，甚至做到全系統內所有節點檢查；上面範例只是方便說明而取了一個比較小的範圍。

在MDG中面臨變更的需求時，需考慮其它節點的共用關連；也就是說，欲變更的節點，可能不只一處被使用。為維持系統內共用MDG的完整性，變更需求所發生的基本的行為可分為下列兩種：

#### 1. 變更節點只有一個父節點：

由於欲變更的節點只有一個父節點，即代表沒有被任何其它節點共用，而其變更的節點是在它的直屬子節點時，只需單純考慮新增、修改或刪除(刪除節點只是從父節點下移除，而此節點還是存在於系統中)，也無需發出任何通知。

### 【演算法】

- 步驟1. 以欲修改的素材做為根節點(root)，在rMDG只有一個父節點。
- 步驟2. 修改其MDG中的直屬子節點。

以圖 22的例子來說，要將Spt2下的Act4換成Act14，因為Spt2在系統中只有一個父節點，即在系統內沒有被其它節點所使用，故可以直接將其子節點更換但不會發生週邊效應。

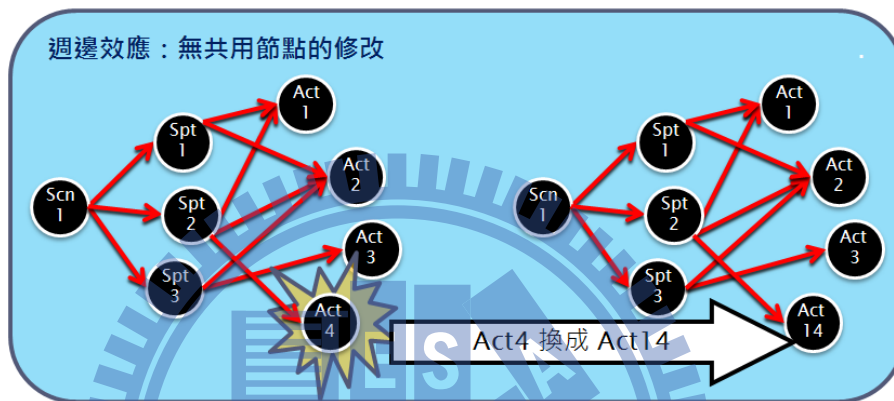


圖 22：週邊效應—無共用關連節點的修改

### 2. 變更節點有超過一個以上的父節點：

如果遇到這種情況為了要維持系統內其它共用此節點的狀態，不可以像前一個例子直接抽換某些節點，這樣將會造成其它共用此節點的元件發生問題，這也是週邊效應必需加以考量的原因之一。此時就需要Graph中的Branch-and-Cut [22]的方法來幫助系統完成此工作。使用Branch-and-Cut的原因是，唯有如此，才可以保留其它共用此節點的父節點的設計；週邊效應評估的部份，即通知有使用到此節點的父節點，看是要做同樣的處理或保持原狀，若要做同樣的處理就是引用因Branch而產生的新節點。

### 【演算法】

- 步驟1. 以欲修改的素材做為根節點(root)，在rMDG找到多個父節點。
- 步驟2. 複製此節點及其它所有節點在MDG中的關連(Branch)。
- 步驟3. 切斷此節點与其它父節點的關連，只保留此次變更的直屬父節點關連(Cut)。

步驟4·變更其子節點的關連。

以圖 23為例，欲變更Spt2下的某直屬節點，先依步驟1檢查，發現Spt2還有被Scn101引用。步驟2，將Spt2複製一份出來，叫Spt201，並複製其與其直屬子節點的所有關連；步驟3，Spt201只保留與Scn1的父類別的關連，其它的都移除；再來做欲做的修改變更。

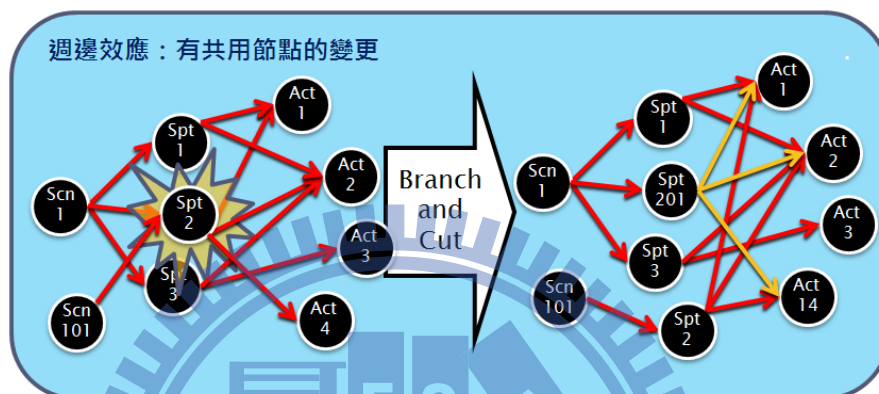


圖 23：週邊效應—有共用關連節點的修改

### 3.2.5 標記沒有被使用到的素材

系統中常常有可能有沒有被使用到的素材，而rMDG可以幫助我們找出沒有被使用到的素材。

【檢驗法】沒有被任何「教材」使用到的素材(圖 24)

步驟1·將系統內的邊全都反轉(reverse)，得到rMDG。

步驟2·以欲檢查的素材做為根節點(root)，以DFS(Depth-First Search)找出所有可以找到的路徑(path)及其經過的點。

步驟3·如果有發現節點是「教材」階層的就停止，代表此節點有被使用到。

步驟4·如果沒有新的路徑可以被找到，且沒有經果任何「教材」節點，就代表本教材在系統內沒有被教材使用到。

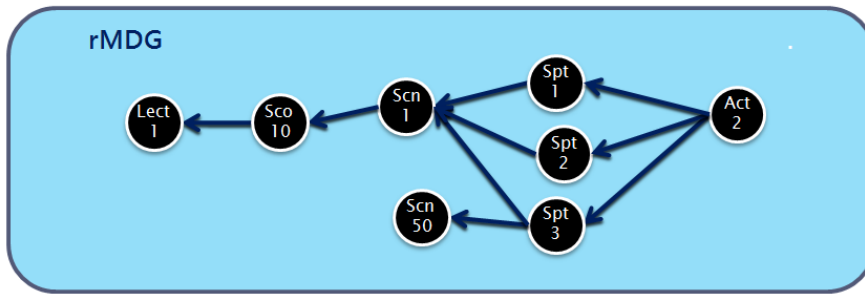


圖 24：以 rMDG 幫助標記未被使用的教材

如果想要知道在某個層級內有沒有使用到該素材，只要把「教材」換成別的(例如：劇本)，就可以得到想要的結果。

另外值得一提的是，在這裡使用Depth-First Search (DFS)[21](22.3小節，p.541)而不用BFS的原因是，系統內的素材大多應該有被使用到，因此用DFS直接走到底比較容易先找到目標的層階(此例來說為「教材」)。

### 3.2.6 設計階段元件缺漏檢查

在設計階段，可能發生缺件的錯誤，如果是在系統內建製，則可以利用系統自動化的檢查而發出提醒，以避免此類品質不良事件的發生(如圖 25所示)。

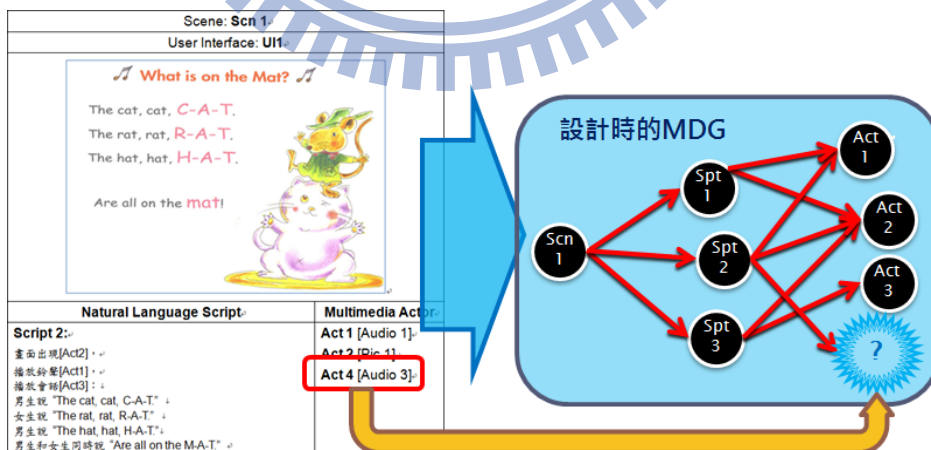


圖 25：元件缺漏示意圖

#### 【檢驗法】

步驟1· 以欲檢查的素材做為根節點(root)，以BFS找出所有可以找到的路徑(path)

及其經過的點。

步驟2· 如果有發現有邊(edge)無點(node)的狀況，就是有缺漏。

### 3.2.7 初步的語意檢查

透過初步基本的規範素材關連中的語意文法(syntax)，便可以做到類似程式Syntax Directed Editor的文法檢查，使得在設計階段，就能自動化找出「語意錯誤」的品質不良事件。以下面這個簡化過的自然語言需求來說(表 5)：

表 5：自然語言需求

Natural Language Script	
Script 2:	
畫面出現[Act2]	↙
播放鈴聲[Act1]	↘
播放會話[Act3]	↓
男生說 "The cat, cat, C-A-T."	↓
女生說 "The rat, rat, R-A-T."	↓
男生說 "The hat, hat, H-A-T."	↓
男生和女生同時說 "Are all on the M-A-T."	↘

以編輯手來製作的例子來看(圖 26)，它的素材演員有圖片、聲音兩種，聲音演員可以用「播放(play)」的，但圖片不行，圖片演員必需用「出現(show)」。而上面這個就是簡單的語意定義(syntax definition)，配合MDG就可以自動化檢查出「語意不合」的品質不良事件。

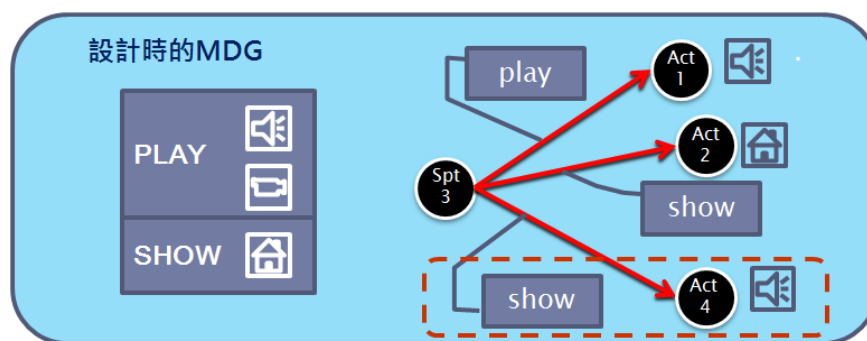


圖 26：語意檢查示意圖

### 【檢驗法】

步驟1· 以欲檢查的素材做為根節點(root)，以BFS找出所有可以找到的路徑(path)及其經過的點。

步驟2· 檢查節點及相關行為定義是否合於語法定義(註)。

(註) 語意文法(syntax)的定義不在本研究的探討範圍，未來展望的章節中有更多說明。

### 3.2.8 語法外的元件群組關連

有些不明顯的元件關連，也會造成品質不良事件的發生。例如有某個劇本(示意如圖 27)中有三個演員：草地背景圖片演員(Act 11)、小狗圖片演員(Act 12)及聲音演員(Act 13)。劇情內容是小狗跑過來，叫兩聲後，跑回原位。



圖 27：劇本示意圖及其所代表 MDG

這樣的一個劇情中的Act 12及Act 13，看似沒有關連但卻有；大家都知道小狗的叫聲應該是「汪汪」，但如果製作時誤植為「喵喵」，就有可能沒被發現。在設計階段時，我們就可以預知小狗圖片應該配上小狗叫聲，所以加上「群組」關連(Grp 1)，如下圖所示，其MDG的檢查法如下(MDG示意如圖 28)：

### 【檢驗法】

步驟1· 以欲檢查的素材做為根節點(root)，以BFS找出所有可以找到的路徑(path)及其經過的點。

步驟2· 檢查其所有子節點的群組關係是否正確。

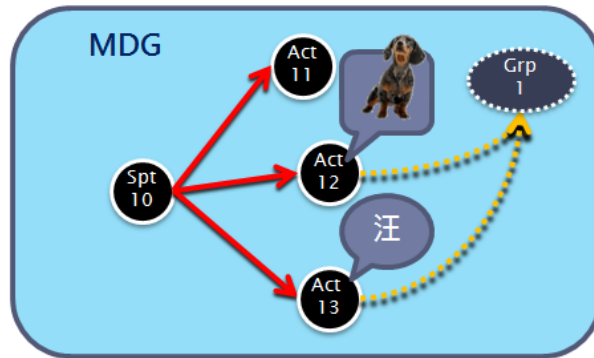


圖 28 : MDG 中的「群組」關係圖

群組關連在下一個版本或元件更新時，可以發揮更大的作用，因為沒有記錄的話，元件和素材之間的特殊關連就會隨著時間淡忘，需要此群組關連，在下次被修改時，保留這一方面的資訊加以控管。

### 3.2.9 元件的遞迴

元件關連發生的遞迴，有可能會發生無窮迴圈而使程式出錯，但它又是必要的存在；因為在某些情況下，MDG中的元件關連遞迴，並不一定表示無限迴圈的發生。以圖 29為例子來說，Script 1按下箭頭後，將會觸發動作跳到Script 2，按下Script 2的箭頭後，又會跳回Script 1。

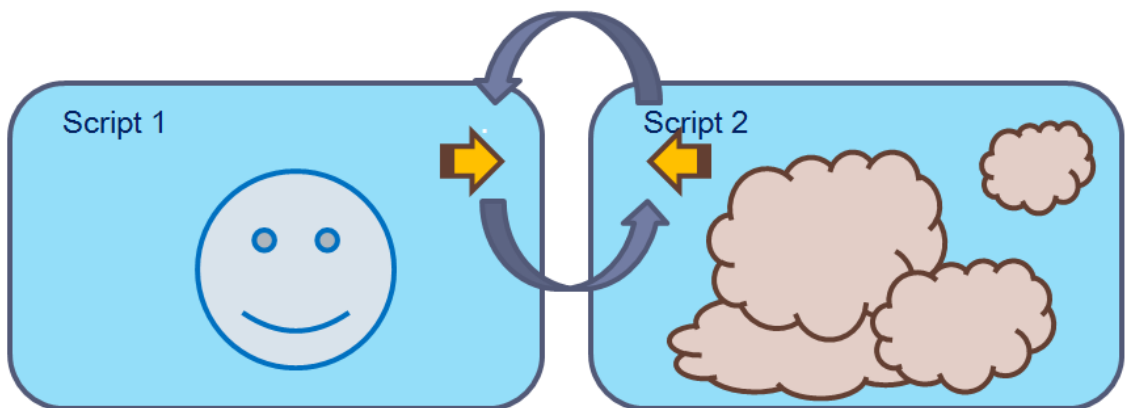


圖 29 : 元件遞迴示意圖

像上述的例子，就是可以視為正常的遞迴參照的情況，而本研究的目的是標示出遞迴的可能，但是否發生無限遞迴則需要人工進一步的判斷。標示遞迴的方式如下：

#### 【檢驗法】(MDG參考圖 30)

步驟1．在rMDG中，以欲檢查的素材做為根節點(root)，以BFS找出所有可以找到的路徑(path)及其經過的點。

步驟2．檢查其所有子節點是否有欲加入節點的存在。

步驟3．若存在則表示遞迴存在，但是否為無限遞迴則需進一步人工判斷。

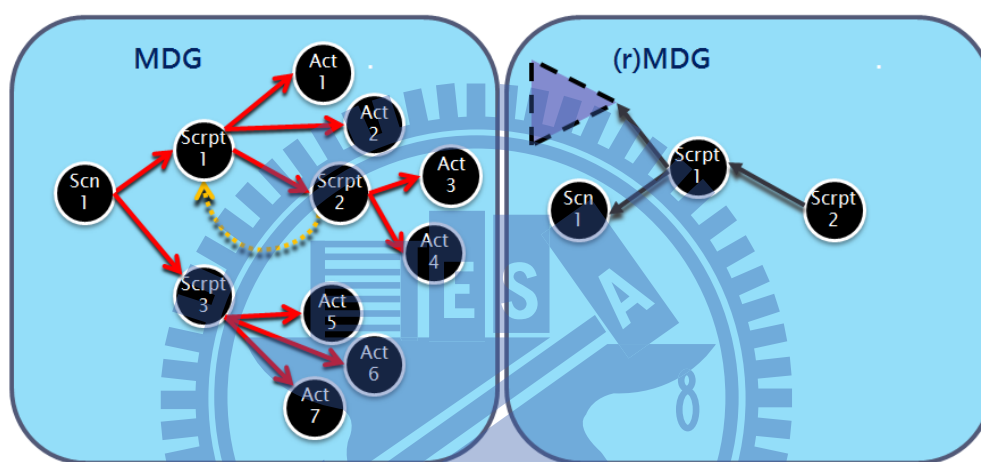


圖 30：(1)欲把 Script 1 加入 Script 2 下。(2)檢查 rMDG，發現 Script 1 已在 Script 2 的 rMDG 中，表示遞迴的存在。

### 3.3 小結

在MDG的幫助下，本章節所敘述的品質不良事件將可以「自動化」地被檢查出來，如果將MDG在多媒體教材的設計到製作都納入品質管制系統內，以現有的Graph理論及演算法，就可以達到上述品質不良事件之排除，而達到減少品質不良事件發生，進而達到提昇多媒體教材品質的目標。



## 四、系統架構及實作

MDG(Multimedia Directed Graph)是一個教材與素材關連的Digraph，配合定義出可能發生品質不良事件的範本，藉著Graph上的演算法，來排除這些已知的不良事件；在教材的製作流程中需要的不只是MDG，還需要製作流程的控管，例如需求管理、變更需求管理…等等，但這些流程上的控管並不在本文的研究範圍，但仍會顧及未來的發展性，本文聚焦在MDG邊及節點的組成，配合演算法的自動化檢查的可能性。本研究以Java[23]搭配MySQL資料庫[24]，實作一組核心程式來完成MDG邊及節點的Digraph架構，及其相關檢驗的實作；未來可在Java EE[25]上以此核心，發展Web上的開發流程系統，做為多媒體教材製作時，流程上的品質管制系統。

### 4.1 系統架構

配合未來的設計的可能性，所以在系統設計時，預留了未來擴充的可能性，本系統將規劃成可以被Java EE專案所引用的核心引擎，也就是圖 31中的「MDG Generation Factory」引擎，整個系統將規劃為多媒體教材的品質管制系統(MLC Quality Control System, MQCS)，MQCS系統將提供流程來管理多媒體教材、其素材及其製作流程，這些管理將基於MDG的結構之上；而本研究的重心就是提供MDG的引擎及管理。

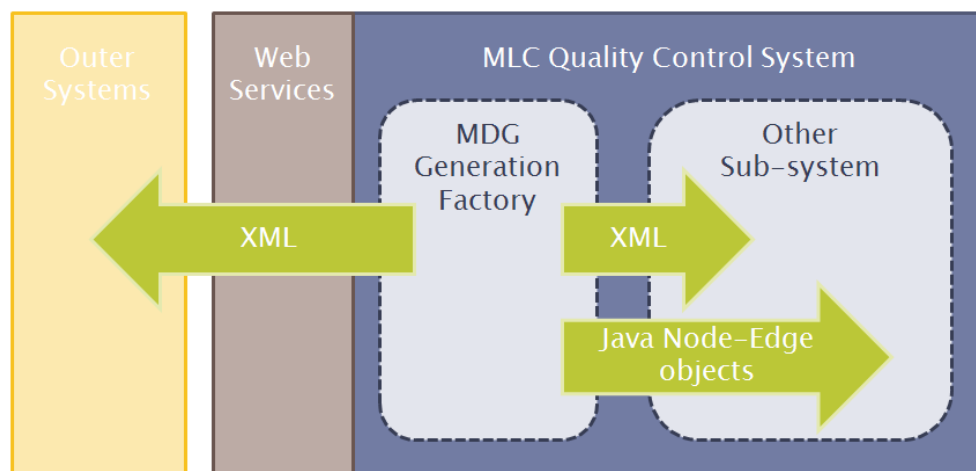


圖 31 : MLC Quality Control System。

MQCS系統中將切割成數個子系統，但其都需要透過MDG引擎來存取及管理元件，透過其它的子系統，也可以將MDG轉成XML給其它子系統利用，或者直接提供內含MDG的Java物件給其它子系統，或者透過web service來讓外部系統存取MDG轉出的XML。

MQCS系統在Java EE上的設計如圖 32所示，在web container下架構系統，使用者可以透過瀏覽器(web browser)來存取系統；後台使用的資料庫為MySQL。而本研究所製作的部份是在虛線框框內的MDG引擎(MDG Engine)，提供點(Node)及邊(Edge)的關係、資訊及結構的完整，並提供本文提到的不良事件的檢核，冀以排除這些不良事件，提升多媒體教材的品質。

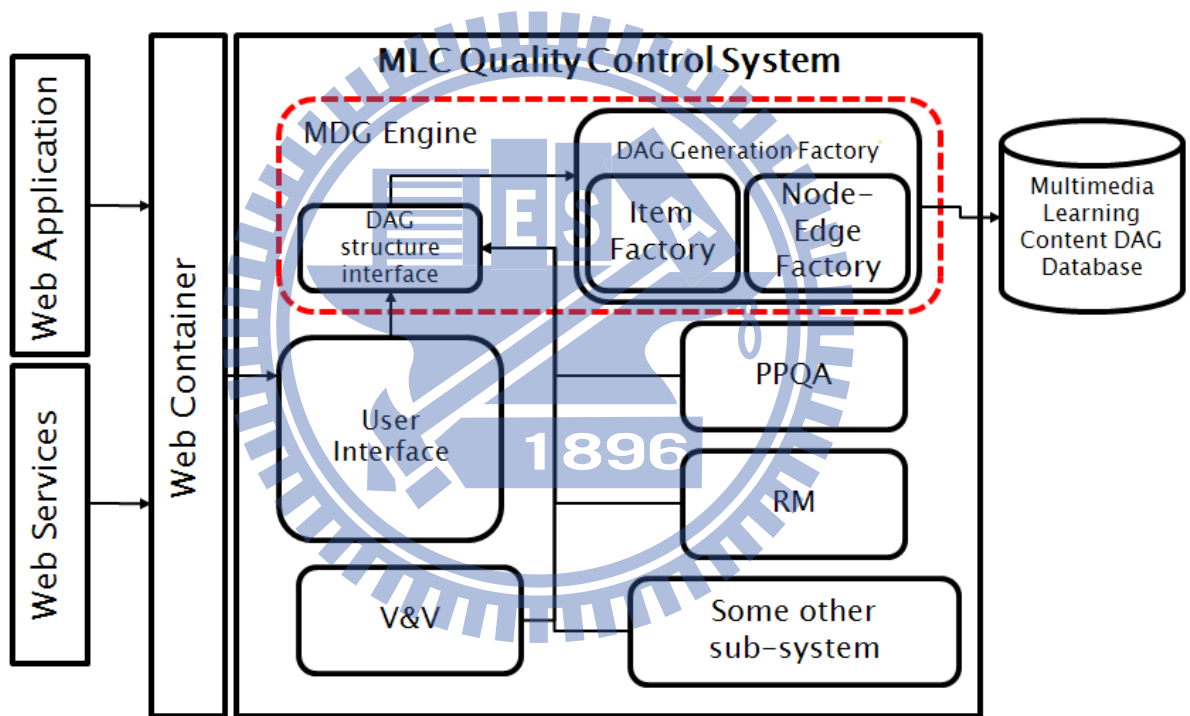


圖 32 : MQCS on Java EE。

## 4.2 系統實作

以Java及物件導向(Object Oriented) 的技術開發，背端存取的資料庫為MySQL。本章節將透過UML[26]及文字上敘述，介紹MDG引擎以Java實作的概念及方法。

## 4.2.1 點(Node)及邊(Directed Edge)類別設計

Digraph的部份，本研究使用Java中已有的類別來實現。作法是在Java中宣告新的Node類別，在其中以Java常用的Vector作為容器(Container)，來記錄其子節點，程式宣告的片段如下：

```
public class Node {  
  
    public Item item;  
    public Vector<Node> childNodes;  
    public Vector<Node> parentNodes;  
    public Vector<Edge> childEdges;  
    public Vector<Edge> parentEdges;  
    public Vector<Group> groups;  
  
}
```

childNodes便是Vector，其中裝的是此節點的子節點，以此結構一層一層下去，就可以展開一整個Digraph。為了方便rMDG的實作，也在這裡宣告了parentNodes，在展開rMDG時來使用。

圖 33是此設計的class diagram，由此圖可以清楚看出類別間的關連。為了節省系統資源parentNotes...等等這些容器不會在run time物件一建立起來就存放實體資料，會等到需要用到時，再呼叫方法(method)來將所屬物件實體化。

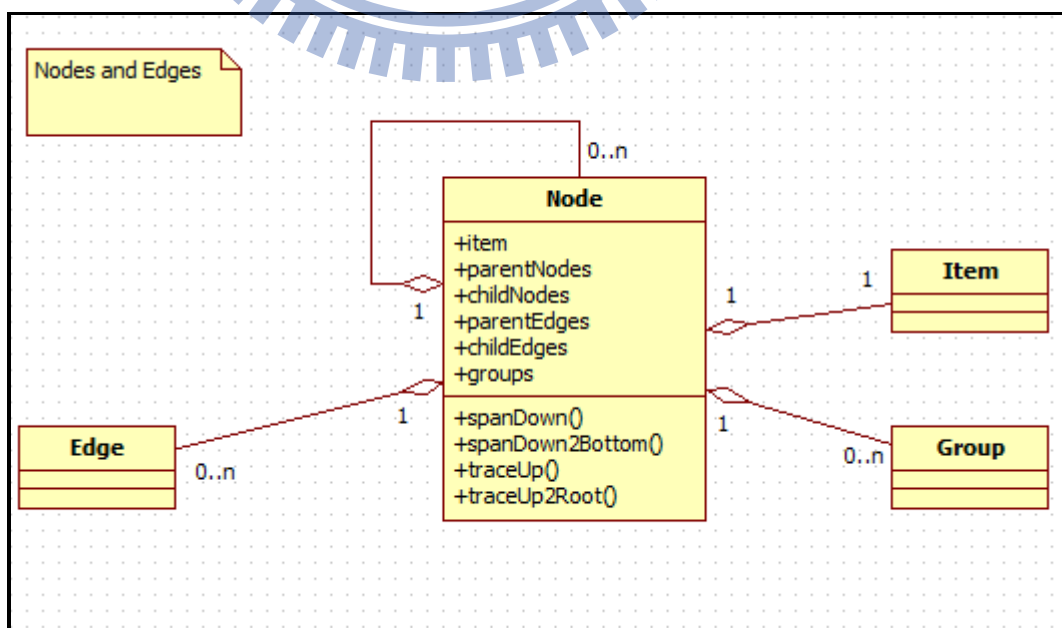


圖 33：點(Node)及邊(Directed Edge)的 class diagram。

至於存入DB的資料中，由於節點和節點之間的關係為多對多(many to many)，所以使用ER Modeling[27]中典型的M:N relationship來解決(圖 34)；透過Edge這個entity中記錄著父節點(parent\_uid)及子節點(child\_uid)，將其節點之間的關連串接起來，形成MDG。

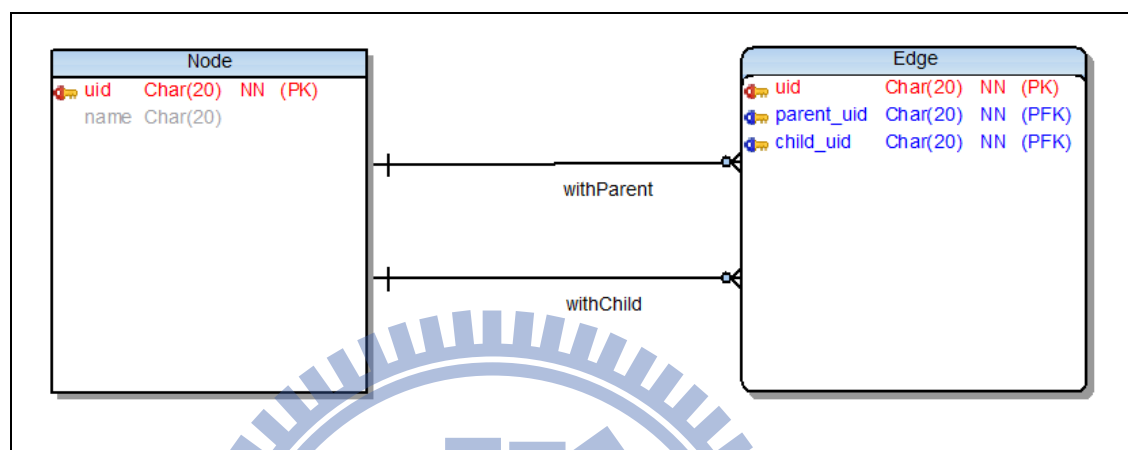


圖 34 節點(Node)及邊(Edge)的 ER diagram。

#### 4.2.2 MDG 的展開法

為了節省效能，並不會每次建構一個Node物件時，就把其底下的Edge物件及子節點或父節點Node物件也一併建構，會在需要使用之前使用Node類別中的spanDown()方法(method)來將相關物件實作(MDG系統設計的Class Diagram請參考圖 35)，來填滿該Node物件的子節點物件(即childNodes)；同樣地，使用Node類別中的traceUp()方法，可以填滿parentNodes。如果要一次就展開整個MDG到leaf為止的話，可以使用spanDown2Bottom()；同樣地要一次就展開整個rMDG到root的話，用traceUp2Root()可以做的到。

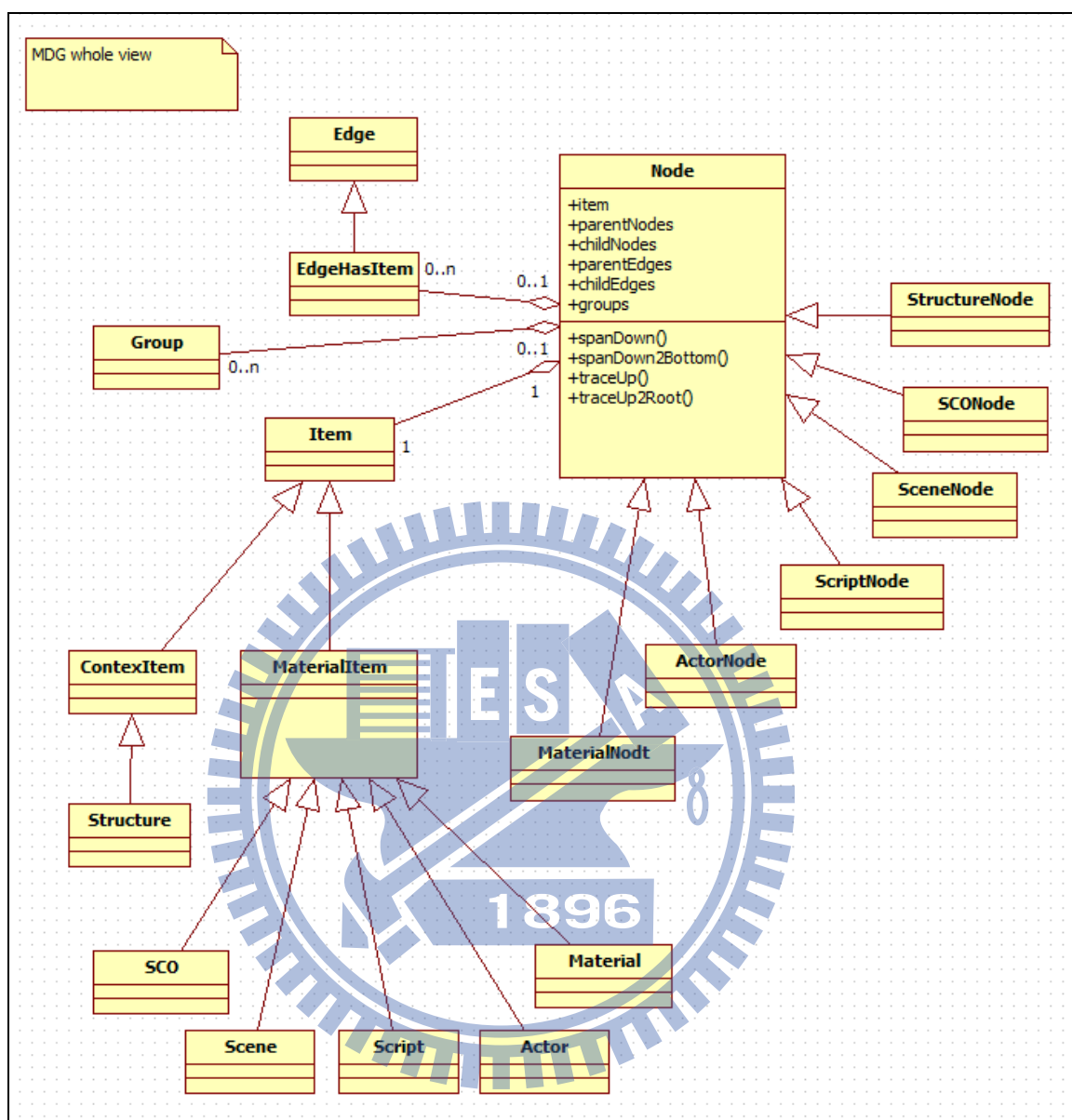


圖 35 : MDG 的類別圖(Class Diagram)。

其展開MDG的spanDown2Bottom()方法，使用的是遞迴(recursive)的方式，程式片段如下：

```

public void span2Bottom() throws SQLException{
    this.spanDown();
    if(childItems == null){
        childNodes = null;
        this.isSpanDown = true;
        return;
    }
    for(Node node : childNodes){
        node.span2Bottom();
    }
    this.isSpanDown = true;
}

```

而Node類別的設計，是使用物件導向(Object Oriented)中的進階的繼承概念：多型(Polymorphism)來設計。因為不同種類的節點，其屬性(attributes)及行為都會有些微的差距，不同的節點，如SCO及劇本(Script)其存放資料的table也不同，因此要下的SQL也不同，故以Node為父類別，並宣告spanDown2Bottom()為抽象方法(abstract method)，其它種類的節點繼承後再實作此方法，實際使用時，便可以利用多型的特性，如上面程式所示，呼叫node.span2Bottom()但透過多型的機制，它會找到其所屬正確的方法。

### 4.2.3 各種不良事件的檢驗

各種不良事件的檢驗方式，在程式的流程上都大同小異，因此本節只舉「檢驗節點下之子節點是否整備齊全」為例，其程式流程如下(圖 36為搭配此流程的Sequence Diagram)：

- (1)要先將欲檢查的節點物件建立，
- (2)-(8)將其MDG展開來，
- (9)以BFS逐節點檢驗是否備齊，
- (10)回傳檢驗結果。

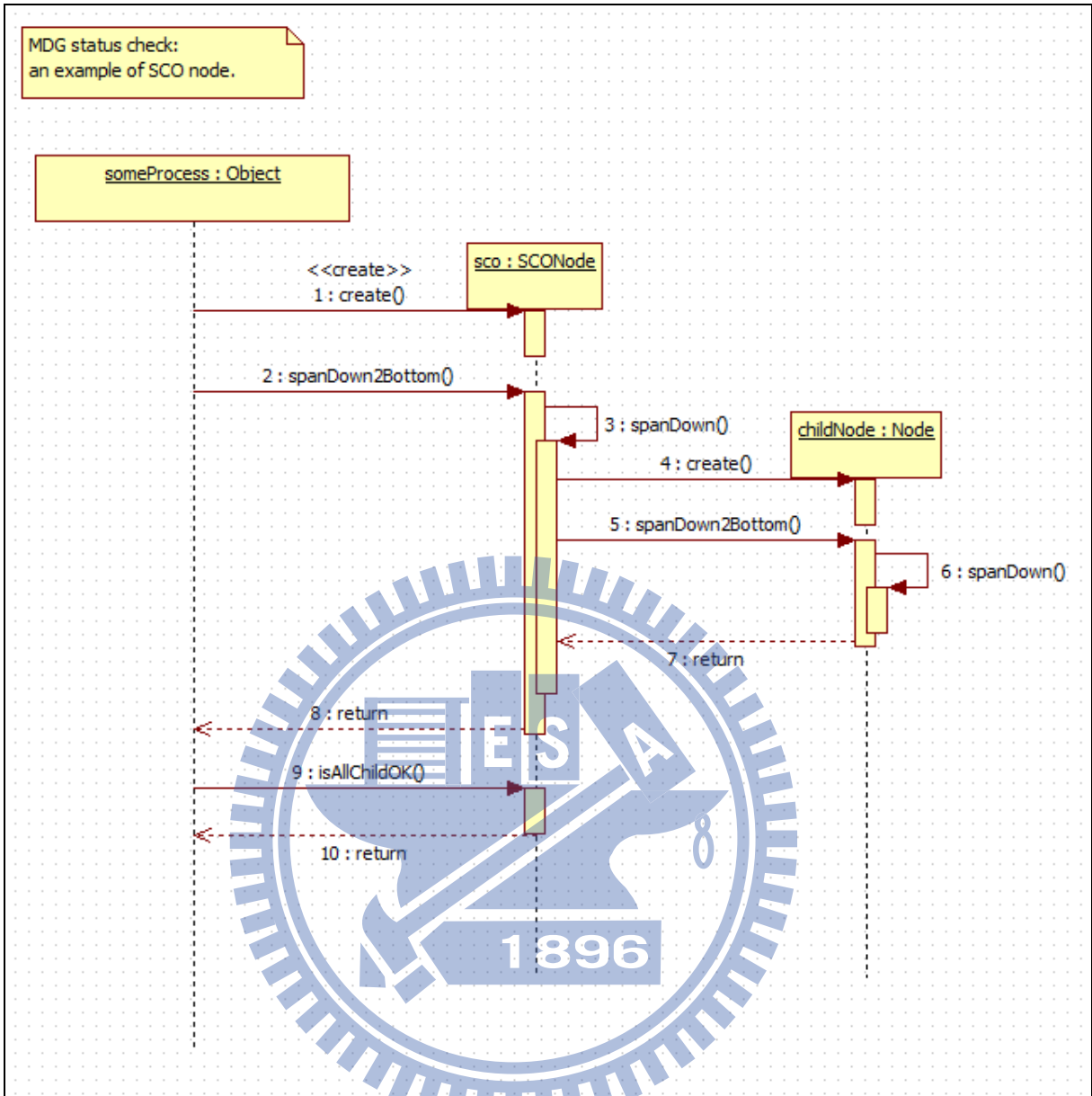


圖 36 : 序列圖(Sequence Diagram)－檢驗節點下之子節點是否整備齊全。

## 五、結論與未來研究方向

有MDG之後，佐以Graph現有的理論及演算法，找出品質不良事件的模式，並加以利用，避免或降低已歸納的品質不良狀況發生，冀以提昇其品質。而本研究仍有許多未逮之處，或有許多其它的可能性，藉由MDG的存在，開發新的途徑，可以做到更多的功能。本文將利用此章節，做一些未來研究方向的發想。

### 5.1 結論

多媒體教材的品質概念，大概可分成兩大面向，一種是build-in的品質，像是教材的設計規劃、內容創意等。另一方面是製作過程(process)的品質控管，此問題就比較少被注意到。而本研究就是以製作過程的品質控管為目標，利用Graph的特性及其現有的演算法，排除已知的品質不良事件。

本研究運用軟體工程的概念，將多媒體教材與素材之間的關連及其邏輯群組依其需求而建立，型成Digraph稱之為MLCs Directed Graph (MDG)；找出製作過程中可能發生的品質不良事件樣本，用常見的BFS和DFS等演算法，對MDG實行檢驗，排除樣本中的品質問題。進而發展Web上的系統：MLCs Quality Control System，利用系統的自動化檢驗，將這些品質不良事件在系統流程中被檢查出來，並加以排除。本研究歸納的品質不良事件如下：

- (1) 素材檔散亂無管理，易發生檔案缺漏或誤植，利用MDG可加以管理。
- (2) 素材檔的引用與設計時不符，利用MDG與設計階段對照其正確性。
- (3) 製作時素材引用錯誤版本或缺漏，可利用MDG做整備檢查。
- (4) 教材需變更時，利用MDG做週邊效應評估。
- (5) 利用MDG標記沒有被使用到的素材。
- (6) 設計階段時就可能發生元件缺漏，利用MDG做缺漏檢查。
- (7) 設計階段明顯的語法不合，可利用MDG做初步的語意檢查。
- (8) 元件素材存在不明顯的群組關連，而發生不一致的引用錯誤，利用MDG中的語法外的元件群組關連檢查可排除。



(9) 元件的遞迴可能發生無限迴圈的錯誤，利用MDG可標示遞迴的發生處，進而檢視是否合理。

而未來更希望能更豐富MDG的其它面向，希望能使MDG在多媒體教材的品管上，提供更好更多的功能。

## 5.2 歸納更多的品質不良事件

除了本研究所歸納的事件，應該存在著更多的品質不良事件，可以在MDG的結構下，利用現有甚至未知的演算法，來將問題自動化標示出來。以目前的MDG來說，它的基本關連就如圖 37中所表示的。紅色箭頭是代表父節點使用到該子節點，而黃色虛線則是語意外的群組關係，關連到群組的節點。

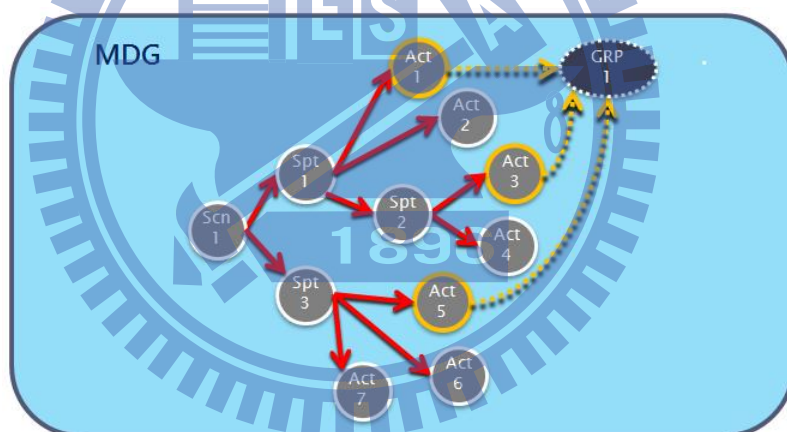


圖 37 · 現有的 MDG 關連圖

或許可以加強、延伸MDG中關連節點的種類，找出更多樣的群組關連；又或者可以借此，定義出更多的不良事件樣版，便可排除更多種類的品質不良事件。

## 5.3 進階語法語意的檢查

有了MDG的點-邊(Node-Edge)關連，若依照各個不同的多媒體教材的製作媒體，例

如Flash、編輯手、Power Point…等等，可以將它們的語法(Syntax)、屬性(attributes)整理出來，並加以定義及統整，建立其語料庫。多媒體教材一般來說，同時有內容(Content)和程式(Code)的特質，但又不像一般程式，有那麼複雜的語法，所以未來可以整理出教材製作媒體的語法，納入系統的規範，在設計到製作的各個階段，都可以反復檢示以避免錯誤。

因此，如果將語法及屬性加入MDG的設計中，如圖 38之示意圖，便有可能同樣使用BFS在掃描子節點，語法不合的品質不良事件可以參考語料庫而檢查出；其屬性參數則可留待製作階段參考或檢查。

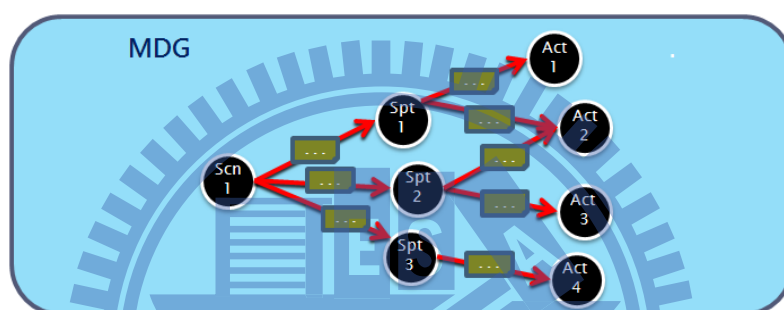


圖 38：加入語法、屬性的 MDG 示意圖

此工作的困難點在，需要對製作媒體語法做深入的了解，並在設計階段，就要將某些所需的變數考慮進去。

## 5.4 無限迴圈的檢出

有了前一節在MDG加上語意語法定義後，還可以帶來另一項好處，那就是「無限迴圈」可能可以檢驗出來，或許沒辦法做到百分之百檢出，但透過「無限迴圈樣版 (Pattern)」的尋找，至少可以排出掉已知的樣版，或者是正向表列出「一定不會發生無限迴圈」的樣版，而減少此類的品質不良事件的發生。

本文在此可以提出一個看似可行的樣版。例如圖 39左圖，有一組互相參照的劇本，依照前面第三章的規範，只能得知此處有交互參照的狀況發生，但不確定是否有遞迴；但此例子的劇本是，在Script 1中，人為按下黃色箭頭才會導向Script 2，反之亦然，所以

「有人為動作介入才可觸發」做為其中一端交互參照的觸發點時，此處就不會發生無限遞迴，因為到了「等待人為動作介入」的時機點時，程式就會停下來等待。但這種判斷如果沒有語意庫，是沒有辦法做到的。

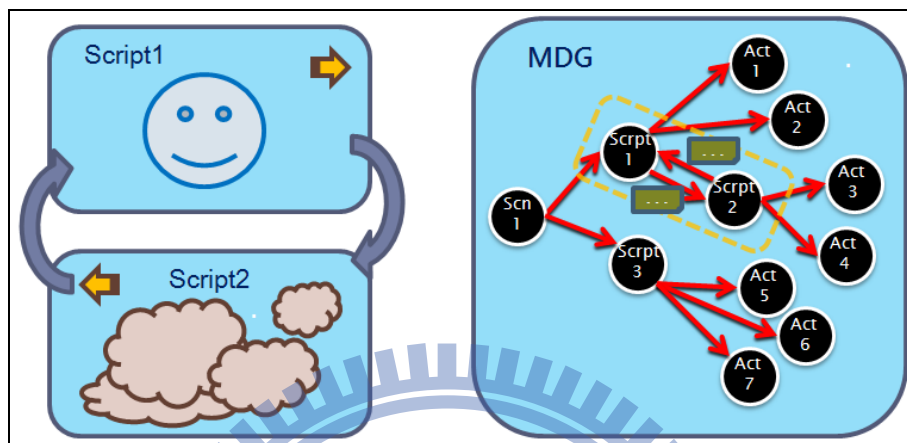


圖 39：MDG 的無限迴圈檢核示意圖

## 5.5 Valid & Validation

有了語意檢查，且MDG中加入了語意的記錄，就有可能更進一步地，產生中間媒介檔，來與其它系統或多媒體的製作媒介來做V&V。最簡單而且通俗的中間檔，就是使用XML，把MDG的節點及邊用標籤語言匯出，而其行為(即程式的部份)，用屬性(attribute)的方式來記錄，如此便可以提供一個V&V的橋樑，而且可以做到是雙向的。

## 5.6 反向工程(Reversed Engineer)

而MDG最終級的挑戰，就是反向工程(Reversed Engineer)。都可以做到前一小節的V&V及產生中間媒介(如XML檔)來做比對，如果系統內的參數記錄完整(當然要針對各種不同的製作媒介做不同的設定及調教)，而且語料庫的資料也完整正確，理論上，把系統內的MDG含其屬性，輸出成一份多媒體教材是有可能的；或者有些部份在沒有正

式編輯前，可能無法很精確(如圖片座標位置)，至少也可以輸出個大概，而後續再來調整，可以減少一點人工的工作。

當然，把完成的多媒體教材直接轉回來，變成邊(directed edge)與點(node)的MDG，可以此為目標(示意圖如圖 40)。

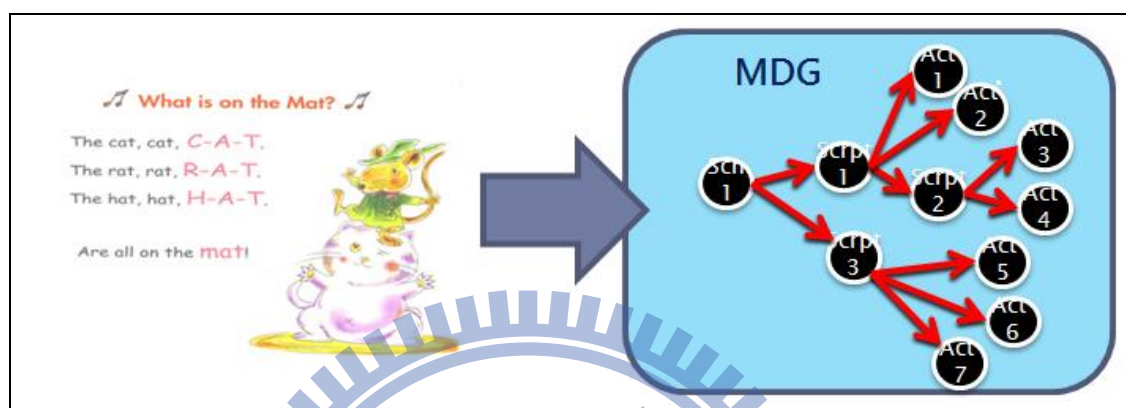


圖 40：反向工程示意圖

## 參考文獻

- [1] W. D. Wallis, A Beginner's Guide to Graph Theory, 2nd edition, Birkhauser, Boston, June 8, 2007.
- [2] 數位學習品質認證中心。2010年，取自經濟部工業局，數位學習品質認證中心網址 <http://www.elq.org.tw/index.aspx>。
- [3] 數位學習品質認證中心，數位學習服務品質規範，數位學習品質認證中心，96年6月1日。
- [4] Advanced Distribute Learning (ADL) Initiative, Sharable Content Object Reference Model (SCORM) 2004 4th Edition: Overview, Version 1.0, March 31, 2009.
- [5] Advanced Distribute Learning (ADL) Initiative, SCORM 2004 4th Edition Content Aggregation Model (CAM) Version 1.0, March 31, 2009.
- [6] Advanced Distribute Learning (ADL) Initiative, SCORM 2004 4th Edition Run-Time Environment (RTE) Version 1.0, March 31, 2009.
- [7] Acrobat, Acrobat Flash, [On-line]. Available: <http://www.flash.com/>
- [8] 智勝國際，編輯手 2004, [On-line]. Available: <http://www.caidiy.com/>
- [9] IEEE, IEEE 1484.11.2 Standard for Learning Technology – ECMAScript Application Programming Interface for Content to Runtime Services Communication. November 10, 2003, Available at: <http://www.ieee.org/>
- [10] SEI, CMMI for Development, Version 1.2, SEI, 2006. [On-line]. Available: <http://www.sei.cmu.edu/cmmi/>
- [11] Margaret Kulpa, Kent A. Johnson, Interpreting the CMMI: A Process Improvement Approach, April 2003.
- [12] Mary Beth Chrissis, Mike Konrad, Sandy Shrum, CMMI: Guidelines for Process Integration and Product Improvement, Addison-Wesley, Inc., Feb 2003.
- [13] 連瑞斌，以 CMMI 為基礎的多媒體學習內容開發流程品質管制方法，國立交通大學，碩士論文，民國九十六年。
- [14] Kalyanmoy Deb, et al., Genetic and Evolutionary Computation - GECCO 2004, Part II, Springer, Seattle, November 23, 2004.
- [15] Hans-Dieter Burkhard, et al., Multi-Agent Systems and Applications V, Springer, Leipzig, Germany, October 23, 2007.

- [16] Qing Li, et al., Conceptual Modeling - ER 2008, Part II, Barcelona, Spain, November 17, 2008.
- [17] Cormen, Thomas H. Leiserson, Charles Eric. Rivest, Ronald L., Introduction to algorithms, Cambridge, Mass.: MIT Press, 2001.
- [18] Narsingh Deo, Graph Theory with Applications to Engineering and Computer Science, Prentice-Hall of India Pvt.Ltd , October 15, 2004.
- [19] 鍾貴榮, 數位教材內容品質控制及管理系統平台設計與開發, 國立交通大學, 2006 年.
- [20] Reinhard Diestel, Graph Theory (Graduate Texts in Mathematics), 3rd edition, Springer, February 10, 2006.
- [21] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, Introduction to Algorithms, Second Edition, The MIT Press, September 1, 2001.
- [22] Patrick Healy, Nikola S. Nikolov, A Branch-and-Cut Approach to the Directed Acyclic Graph Layering Problem, Springer Berlin/Heidelberg, January 1<sup>st</sup>, 2002.
- [23] Ken Arnold, et al., Java(TM) Programming Language, 4th Edition, Prentice Hall, 2005.
- [24] Paul DuBois, MySQL, 4<sup>th</sup> Edition, Addison-Wesley Professional, September, 2008.
- [25] Eric Jendrock, et al., The Java EE 6 Tutorial: Basic Concepts, 4th Edition, Prentice Hall, September, 2010.
- [26] Thomas Baar, UML 2004: the unified modeling language, Springer, Berlin Heidelberg, 2004.
- [27] Peter Rob, Carlos Coronel, Database Systems: Design, Implementation, and Management, 8th Edition, Course Technology, December 20, 2007