私立東海大學資訊工程與科學系研究所

碩士論文

指導教授：朱正忠　教授

以使用者介面之模型應用擷取需求樣板

UI Driven Pattern-Based Requirement Engineering

研究生：陳宏豪

中　華　民　國　九　十　八　年　七　月　二　日

摘要

在嵌入式系統開發初期，由於產品的開發時程(Time To Market)將會影響到市場的需求以及系統發展趨勢，快速的撰寫出一份完善的系統規格說明是件不容易的事情，而在產品價格上，減少系統開發時程將可提高產品價值。因此提出嵌入式系統開發方式來縮短開發時程。並且採用了雛型開發方法(Prototyping Development)，使用此種開發方法可快速產生完成系統，但是仍會遺漏系統中某些重要特性或是相關的文件，因此引入設計樣板(Design Pattern)來輔助需求的擷取。又因為傳統的設計樣板太晚導入到開發過程，因此以 UI 加上設計樣板，以輕量級的 Model Driven Architecture 幫助將系統中容易遺漏的資訊及文件給補齊。本研究中進行探討並且提出分鏡應用於 Prototyping 開發上，除了能縮短開發時程，對於開發者與客戶在溝通上，將可透過階段式開發雛型展示提供更準確的系統開發。

關鍵詞：嵌入式系統、雛型開發法、使用者介面、設計樣板。

Abstract

It is not easy to describe a complete System Requirement Specification in the first step of embedded System development as fast as possible. Due to products of Time Market will involve requirement and tendency of system development. By decreasing development time, we can raise the product value and price. So we use the prototyping development to develop system quickly, but we still leave some important attributes or documents of System information. In the research we discuss how to use Ui-Driven applying in prototyping development, besides to decrease development time, we make a communication in developers and customers that making a prototyping modeling correctly.

Keywords: Embedded System, Prototyping Development, Ui-Driven, Design Pattern

目錄

圖目錄

表目錄

第1章　導論

1.1　前言

近年來，由於嵌入式軟體平台產業及掌上型產品的盛行，中介軟體與應用軟體系統的需求量愈來愈大，複雜度與異質性也越來越高，中介軟體與應用軟體產業的問題也因為產業的迅速開發伴隨而來。嵌入式軟體由於開發方式特殊，根據英國電機工程師協會的定義，它是屬於一種電腦軟體與硬體的綜合體，並且特別強調「量身訂作」，因而導致開發流程的評估不易，往往需要開發者為客戶「客製化(Customization)」，但開發者所使用的方法及流程都是固定的，所以我們必須要開發一套專屬於嵌入式系統的軟體工程方法來處理可能伴隨而來的效應，例如；工時的增加導致時程延後、預算超支導致軟體品質下降，會產生這些問題最主要的原因就是目前 IT 產業針對嵌入式軟體的開發流程方法尚未純熟，所以開發一個軟體需要包含各個領域的專家(Domain Expert)才能確保這個軟體是符合需求的。若使用的軟體輔助開發系統工具互相整合有難處，彼此間缺乏協調整合機制，所帶來的不確定因素讓開發者之發展環境持續在改變而使其無所適從，軟體開發方法、軟體開發流程與軟體輔助開發工具無法相互整合，以致於不能發揮整體功效。

1.2 研究動機

軟體在開發階段時，往往在需求分析階段時，須投注相當大的時間及心力，在嵌入式軟體開發所注重的開發時程上，快速的分析使用者需求更為重要。並且對於軟體開發流程以及輔助工具之整合，仍有改進的空間。因此在需求分析階段後，我們會產出以文字為主的文件並做為設計階段的輸入，接著產生出平台獨立模式(Platform Independent Model, PIM)的文件，但將文字為主的需求文件進行分析並設計需求描述的過程中，因為缺乏系統化的分析方法，必須花費較多的時間來反覆檢查需求文件的正確性與完整性，除此之外還需仰賴分析師系統開發的經驗來設計出需求文件。

因此目前已經有前人提出強韌分析(Robustness Analysis)的方法，透過此方法分析使用個案(Use Case)之敘述內容，可初步辨識出參與使用個案的物件，並將物件分成介面物件(Boundary Object)、控制物件(Control Object)與實體物件(Entity Object)[18]。但是如何從需求文件透過強韌分析轉換出強韌圖(Robustness Diagram)以及利用強韌分析修飾需求塑模文件的相關研究是目前較為缺乏的部分。[17]

1.3 研究目的

為改善並解決上述之問題，我們提出一個以使用者介面之模型應用擷取需求樣板，可以將需求以視覺化的方式來表達，並且透過分鏡的角度加強需求的描述方式。有別於以往的嵌入式系統開發方式，我們導入需求樣板的概念以及塑模語言的表示方法將使用者需求快速分析，讓開發人員能夠清楚的了解使用的需求，進而開發出符合使用者的軟體。而經過領域專家（Domain Experts）所描述後的需求，將需求依照劇情法(Scenario)的描述，得到完整的故事劇情。而根據分鏡過程所產生之物件，最後依據物件導向的特性便可再利用此需求物件，或者以繼承、實現的方式，加上特有的需求屬性即可用以描述新的需求，減少在需求擷取階段到設計階段時所需時間，並且快速的描述出需求特性及關係。另外，透過本論文所提出之情境法則，可使每一項需求皆有對應之事件流，並由許多的事件流(包括基本事件，其他事件，例外事件)組合成一個情境，透過一連串的畫面呈現，並且藉由牛頓法的應用來解決使用者與開發者對於使用自然語言來描述需求所發生的不一致、不清楚以及不了解的問題，更不會因為每個人的看法及想法的不同，而有不同的解讀，也減少產生模稜兩可或是不精確需求的情況產生。我們也提出一套新的元件分類方式，將我們的元件依照分鏡的呈現方式作為分類的依據，日後在開發類似系統時，只需抽離及加入需要更換的元件，可達到元件之再利用性。

1.4 章節安排

論文的章節安排分述如下：

第二章　背景知識與相關研究，此章說明與本論文相關的研究，並介紹需求樣板的觀念，以及方法的基本特性及其相關的應用。

第三章　以使用者介面之模型應用擷取需求樣板，此章說明本方法提出定義之物件導向需求模型。

第四章　案例研究，本章說明利用使用者介面之模型應用擷取需求樣板來實際操作。

第五章　結論及未來方向，此章將本論文之研究成果做總結，並提出未來研究之方向。

第2章　背景知識與相關研究

在系統開發流程中，需求擷取是在開發過程時相當重要的一環，因此在需求擷取時，我們更需要投入更多的心力在上面，讓需求能夠完整表達。而不好的開發流程，已經被認定為是造成成本浪費及進度落後的主要原因，在任何新的專案的生命週期裡，最重要的一個部份就是要定義出新系統的需求，需要專業的投入以及和客戶之間的溝通。在實例中，我們觀察到近年來嵌入式軟體中的專案進度是工程師們最在乎的事項(如圖 1 所示)。專案進度的時程若是無法準時達成目標往往會影響到整個專案的成敗，完整的開發流程能夠幫助我們瞭解及建立系統在各階段所需之資料。因此良好的開發流程能夠有系統的將專案完成。造成不好的開發流程的原因有許多，其中在需求(Requirement)與設計(Design)的結果不盡相同，也就是客戶需要的東西不是開發者所設計東西，主要原因與目前並無一套標準，也沒有量化後的數據能做追蹤所造成，缺乏完整的流程容易導致專案失敗的可能性提高。

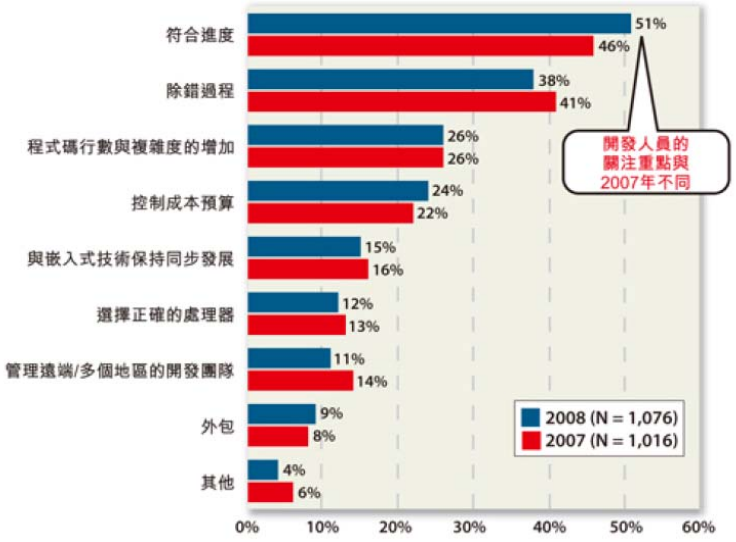


圖 1 2007 年與 2008 年嵌入式的挑戰

(Source: Global Source 電子工程專輯，2008 年 11 月)

以往在開發嵌入式系統時並無導入軟體工程，會使其在開發中未能注意到更多的層面，進而增加錯誤的產生以及人力和資源成本提高，造成開發時程的延誤。當增加了各方面的成本，在產品價格上會有嚴重的限制，即對於市場的需求與市場的趨勢無法契合。而開發時程的延誤，也將產生出功能不完善的產品，或是將無法確切的掌握搶佔市場的時機(Time to Market)。對於是否能夠有效的縮短時程是相當重要的議題。

2.1 雛型開發(Prototyping)

在嵌入式系統開發初期，想得到一個完整準確的規格說明不是一件容易的事。特別是對一些 Time To Market 的軟體系統開發。用戶往往對系統只有一個模糊的想法，很難完全準確的表達對系統的全面需求。開發者對於所要解決的問題或是需求更是模糊不清。隨著開發功能增加，用戶可能會產生新的需求，或因為系統環境有新的變化，要求系統也能隨之變化；開發者又可能在設計與實作階段的過程中遇到所沒有預料的問題，像是規格說明難以完善，需求的變更以及在溝通中的模糊及誤解，都會成為開發者的阻礙。

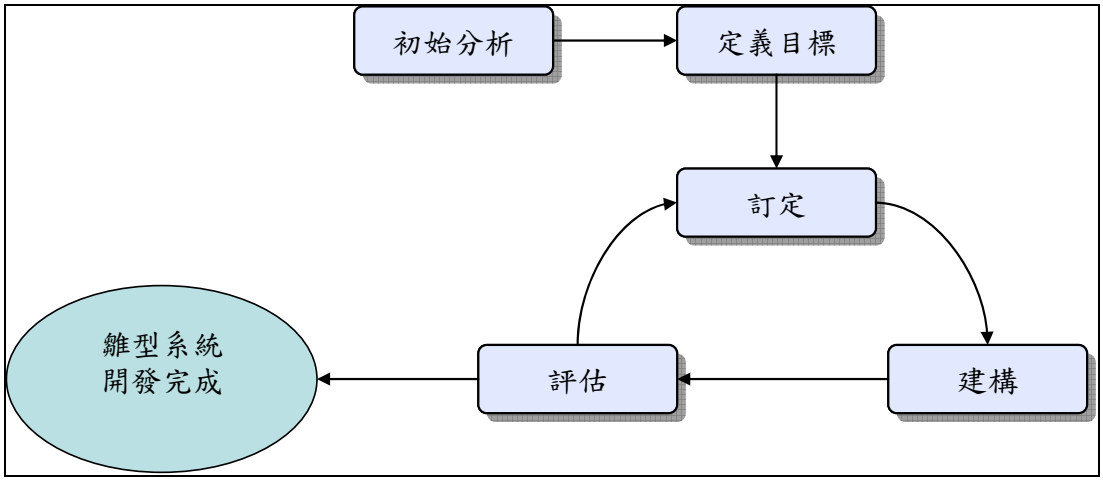


圖 2 雛型開發生命週期

雛型開發生命週期分為五大階段，如圖 2，包含有以下階段：

執行初始分析(Perform an initial analysis)

定義雛型系統目標(Define prototype objective)

訂定雛型系統(Specify prototype)

建構雛型系統(Construct prototype)

雖然快速軟體開發的方式有很多種，不過都有以下幾個共同的特色，如：規格制定、設計和實作程序是並行的（concurrent）。它沒有詳細的系統規格，而且設計文件也是盡量減少，或者是由程式設計環境來自動產生。使用者需求文件也只定義系統最重要的特性，並且雛型系統是以一連串的增量模組來開發。終端使用者和其他的系統關係人都會參與指定和評估每個增量模組，透過系統使用者介面經常都是使用互動式系統來開發，此時的介面設計只要在介面上畫出和放置圖示即可。雛型開發的概念在於能夠快速的分析、訂定、建構，開發等動作。在開發生命週期的初期時，便決定了此軟體開發的走向。

2.2 情境故事法(SCENARIO)

一般人所認知的情境故事法最早由英國 ID TWO 設計公司與美國設計公司

Richardson & Smith (目前已被英國 FITCH 設計公司所併購) 共同為全祿公司開發影印機面板設計所用的方法，最先的應用是從觀察使用者情境開始，然後才慢慢被廣為應用於各類產品設計上，所以情境故事法最早是應用於互動設計。

目前相關類似的方法有 SCENARIO-情境故事法[20]、劇本法、使用情境、場景描述；還有電影腳本(Screenplay)、劇情概要(Outlines of a play)、拍攝腳本(Shooting Script)…等。Campball [3]認為情境故事法有太多的用法，定義相當模糊，但他認為情境故事法具有兩個特性：它是有順序的描寫一個過程、一些動作及事件。 它是以敘述型式(narrative)對活動作有型的描述。情境故事法是依照時間順序將一些動作(action)及事件的片段(fragmentary)描述所串連而成。[14]

2.3 分鏡(UI Operation Sequence)

分鏡技術的概念取自電影，主要是指電影、動畫、電視劇、廣告、音樂錄影帶等各種影像媒體，在實際拍攝或繪製之前，以圖表的方式來說明影像的構成。利用連續畫面以一次運鏡為單位作分解，並且標註運鏡方式、時間長度、對白、特效等；而在本研究中，我們採用分鏡技術的概念，電影分鏡技術中使用影像來描述故事，正如同軟體使用圖片來述說使用者的需求一樣，“當我看到它我就知道我要甚麼”(I’ll know it when I see it , IKIWISI)這種現象在軟體開發中普遍存在，所以利用連貫關係的介面加上各種觸發事件加以描述，以視覺的方式來確定使用者需求的構成，讓使用者清楚瞭解系統流程及確定自己需求。

目前市售產品中，已有相關軟體需求塑模工具，如 SoftScore[9]為鼎新科技公司開發出的軟體需求確認及塑模工具，以協助使用者確認流程，並提供模擬功

能，類似播放電影的方式，讓使用者及開發人員確認系統的外觀及功能，比傳統文件化的表達方式更易理解。SoftScore 符合 TDD（Test-Driven Development）的精神，可註解測試方法及系統功能，匯出需求模型成為 PDF 檔案，可產出 UML 的 Use Case 及測試案例。

2.4 標準模式語言(Unified Modeling Language, UML)

統一模式語言(UML)是一種 Modeling Language，結合了 G. Booch，J. Rambaugh 與 I. Jacobson 三人的物件導向方法論所提出的一種物件導向語言(OO Language)，並被 OMG 納入為標準。UML 結合了 Booch 的 OO method 與 Rambaugh 的 OMT 與 Jacobson 的 OOSE,發展至今(OMG UML v2.0)已經包含了 13 種 diagram 分別是：Class Diagram, Component Diagram, Composite Structure Diagram, Deployment Diagram, Object Diagram, Package Diagram, Activity Diagram, State Machine Diagram, Use Case Diagram, Communication Diagram, Interaction Overview Diagram, Sequence Diagram, Timing Diagram 用來描述軟體開發過程不同階段的抽象概念，也因此 UML 已成為物件導向軟體開發的一項重要的技術。[21][2]目前 UML 已成了一種通用的物件導向語言(Object Oriented Language)，廣為被應用在各類系統的描述及設計上，成為一種溝通設計理念的語言，除了主導廠商 Rational 的推廣及擴展之外，其他為數眾多的廠商及研究單位也利用 UML 做為表達設計或研究成果的語言。在軟體工程中，UML 已被應用在相關的研究中，譬如：Software Architecture、Framework、Patterns、Software Process。應用在其他領域如：Real-Time system、Embedded System 以及 workflow。目前支援 UML 的廠商與團體紛紛推出許多的工具，除了最著名的 IBM Rational 之 ROSE(Rational Object Software Engineering)以外，TogetherSoft 與 VisualUML 也推出了 UML 的相關工具，而 Microsoft Visio 2007 也支援 UML 的 Diagram 繪製模組，並以 UML 所訂定的各種圖形為標準。

2.5 強韌圖(Robustness Diagram)

一開始是由 Jacobson [8]參考在他的書中所提出的分析模式(Analysis Model)概念，目的是分析出系統的三種型態物件，並且描述物件之間的互動溝通的連接為何；後來描述一種技術稱為強韌分析(Robustness Analysis)，主要是分析使用個案的步驟敘述、驗證使用案例中的事件邏輯，以及確認所用的敘述是與之前分析的其他使用個案是一致的，並確認哪些物件都會參與在這個使用個案。強韌分析是一種介於使用個案與軟體設計階段的中間分析，強化使用個案內容表達的正確性與完整性(What)並連接系統物件的互動設計(How)之間的落差，有效地成為使用個案圖與其他 UML 圖形(如循序圖及類別圖)的轉換橋樑。

我們在建立循序圖之前先做強韌分析，可得到更完整的使用者需求塑模文件，並節省更多繪製循序圖的時間。除此之外，使用個案是從使用者觀點去描述，缺少了系統觀點之描述，其原因是行為者本身可能對於系統的執行流程不夠瞭解，因此造成使用個案之描述並不夠完整與健全，因此當進入循序圖或類別圖的設計階段時，需要花費更多的時間去設計出其系統規格。為了能夠迅速與有效分析其使用個案，並更細節化的描述使用個案來幫助系統分析師建構循序圖或類別圖，可先針對需求文件來實施強韌分析，並將所得到的分析結果回饋至使用個案之描述，以確保使用個案描述之完整性與正確性。

圖 3 強韌圖的主要三種表示方式

強韌圖如圖 3 所負責工作內容如下所示：

1. 介面物件（Boundary Object)：是行為者和系統交談的媒介。
2. 實體物件（Entity Object）：用來表示儲存在資料庫中的資料，某些的實體物件可以是暫存的物件，例如搜尋結果，當使用個案執行結束時，這些資料也將消失。
3. 控制物件(Control Object）：包含許多的應用系統邏輯（例如企業規則)，是實體及介面物件之間的溝通者。

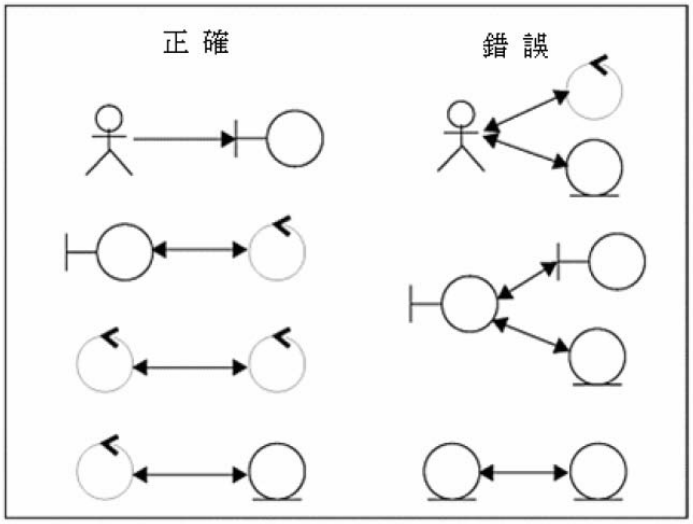


圖 4 強韌圖之連接規則

以下為圖 4 之連接規則說明：

1. 行為者只能跟介面物件互動。
2. 介面物件只能和控制物件與行為者互動。
3. 實體物件只能跟控制物件互動。
4. 控制物件可以與介面物件、實體物件和控制物件互動，但是不可和行為者互動。

強韌圖與物件互動規則：強韌圖在結構上是屬於 UML 的類別圖，但是依據 Jacobson 原本的概念中是比較接近合作圖，主要表達出哪些物件參與使用個案的情節，以及這些物件如何進行互動的架構。物件連接規則的目的是當瞭解個別的實體類別，但不清楚實體類別彼此之間的關係時，能有效建立關聯。例如知道實體類別彼此的關聯後，在兩個實體類別中間可以安插控制類別。繪製強韌圖連接規則之說明與圖形如圖 4 所示。

2.6 需求樣板(Requirement Pattern)

軟體最常見的問題是在於獲取、精煉以及分析需求。樣板是用來解決一直重複性上的問題，也可以幫助我們針對問題的解決溝通與分享[22]。簡單的說，樣板是讓我們能再使用我們已有的解決舊問題的知識應用到解決新的類似的問題上，而且樣板必須有一般性的特質。而需求樣板是一種用來幫助我們取得、建立模組並驗證系統的需求[13]。Eduardo B. Femandez 等人提出分析樣板是定義初始的物件導向模型[5]。Haitham Hamza 等人提出大部分的分析樣板都被當作範本使用，可被實例化而且再使用。在 Alexander 的著作中提出了樣板，以抽象的方法取代詳細的方法能精煉的獲取設計上的需求[1]。軟體再使用是從現存的軟體創造一個軟體系統的流程，由於軟體再使用是一個已被驗證過的產品，所以它可以被用來促進生產力、提昇品質。在軟體工程尤其是需求工程上，再使用之特性已被廣泛的建議用來改善軟體發展以節省時間並促進產品的品質。[12]

2.7 設計樣板(Design Patterns)

談到 Patterns，在整個軟體系統裡的 Design 階段存在風行已久且標準化的 23 個 Design Patterns [6]，當然 Patterns 不只會有 23 個，只不過這些是已被證實常被使用且提出標準化，Design Patterns 裡對於描述架構使用的是 Unified Modeling Language (UML)，並說明其欲解決的問題。在大部分的設計上就可以依據需要重複地使用這些 Patterns，替設計者省下許多工夫或者避免一些不必要的錯誤，並減少維護上的次數和困難[16]。

對於給軟體設計問題一個明確且精細的答案，一般而言，通常會在較大的

軟體系統裡廣泛地使用 Design Patterns。然而實際上，設計師可能會採取某個 Design Pattern 來當作這個系統的設計計畫，但是實作或維護程式的工程師有時會因為不瞭解這個設計樣板，或者為了某種方便而不嚴謹的遵照設計規格撰寫程式，這將會使架構失去原本的功能或受損。[23]因此，如何有效地使 Design Patterns 落實在實作上，是需要去做到 Design Patterns 的程式碼的自動化產生並驗證撰寫的程式碼是否符合設計樣板，目前 Integrated Development Environment(IDE)工具尚無提供這類完整的功能，以致一般沒有 Patterns 經驗的工程師沒有足夠的能力照著設計規格上撰寫程式碼，即使撰寫出來也無法驗證是否符合設計規格[15]。

第3章　研究方法

3.1 研究架構

在本論文中，我們提出一套流程來幫助我們快速產生系統雛型，並且分為若干個步驟：第一，利用電影在敘述情境時所使用的方法來完成描述情境，並且結合牛頓法作用力與反作用力的想法，將需求以一來一往的傳遞方式加以描述，降低需求的不一致性。第二、建立需求樣板時，透過上述的描述，在配合我們提出針對需求樣板中的細項，建立需求時會簡單許多，對於需求分析師或是開發人員在需求擷取時，能夠提供幫助。

第三步驟則是產生強韌圖(Robustness diagram)，將需求樣板所建立的流程依照強韌圖所建立的規則。可將需求部份做轉換。第四步驟時，根據第三步驟所建立的強韌圖轉換成循序圖(Sequence Diagram)，透過循序圖的特性來表示建立事件的先後順序。最後再根據前置條件與後置條件的建立將各個情境圖中的循序圖串聯起來，達到事件的完整性及流暢性。此方法可將一個系統的雛型快速產生並且清楚的描述各個情境。

在此架構中，我們藉由循序圖所產生分鏡畫面，將所對應之功能視為一組元件，將來若是有相關之系統開發時，可依照此元件所對應之畫面去做增加或是刪除。經由上述的步驟，我們將每個元件所組合而成之情境可組合成一個在利用元件(Reuse Component)，日後再開發其他相關系統時，可重複利用或是經由小幅度的修改，便可達到元件再利用性。圖 5 為本論文研究之架構圖：

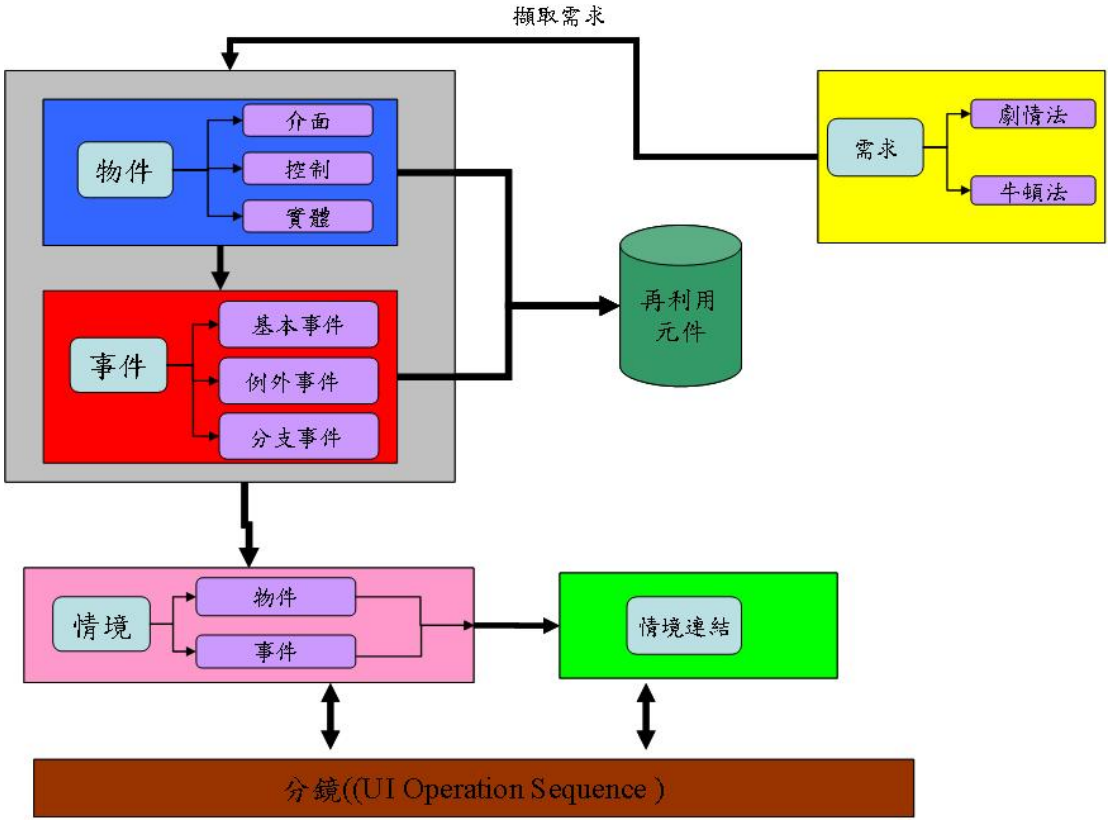


圖 5 研究架構圖

3.2 需求情境

3.2.1 故事建立

在故事的建立時，我們將以場景的描述方式來建立整個故事，將故事先分成若干個場景，如圖 6，故事的架構　人-境-物-活動(用)所以在規劃系統時應要以使用者的情境去模擬，而不是以使用的功能，步驟或是流程的角度去模擬。在取景的時候，透過『人-境-物』此三項資訊彼此互動可產生活動順序，敘述故事內容，描述事件脈絡。如荊軻的故事中，人有荊軻、秦王、境有宮殿、刺秦背景、物有魚腸劍、秦國地圖、活動因此可以展開來。將一個分鏡依時間順序，分成幾個部份描述，像連環圖一樣以圖與文的方式表達出來，協助深度細節的發展與意念表達。

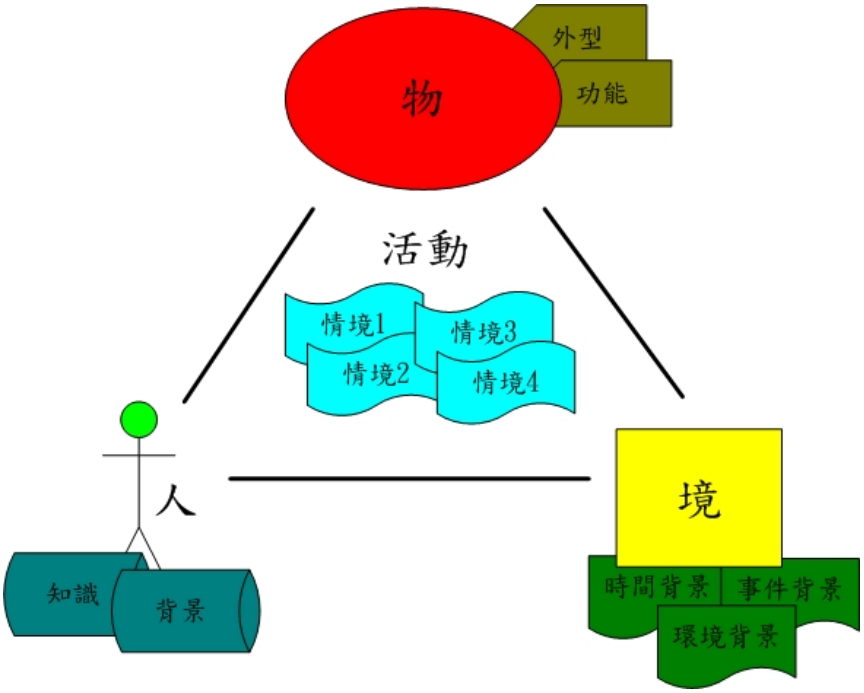


圖 6 情境互動圖

3.2.2 情境故事法與牛頓法

說故事與聽故事都是人類基本能力，透過說故事描述人、事、物，聽故事的人從中了解其背後的意義及訊息。情境故事法亦稱之為劇本導引(Scenarios)，其中概念的導引「起、承、轉、合」，是透過生活型態、技術、社會趨勢、市場定位、SWOT 分析，使用者的面貌、劇本地圖等

分析解讀與建購程序，引導設計者進入背景情況，再透過劇本撰寫閱讀體驗，將使用情境、互動模式、關鍵議題等抽象內隱知識帶出，並累積解決問題感張力。然後由靈感轉成應用概念、互動概念、產品模擬等具體型像。最後再透過各種劇本將概念轉換成更成熟具體的產品規格、產品型錄、行銷策略。除此「以使用者為導向」，於設計開發過程中，不斷以視覺化及實驗方式引導參與產品設計開發人員，以「人、時、地、事、物」做為劇本的基本撰寫模式敘述，也可幫助參與者自全無的開始狀況下發展出情境。本研究結合此方法與牛頓第三運動定律(作用力與反作用力)，如圖 7，將敘述給簡化，先由使用者做動作給系統，再由系統根據動作再回應給使用者。

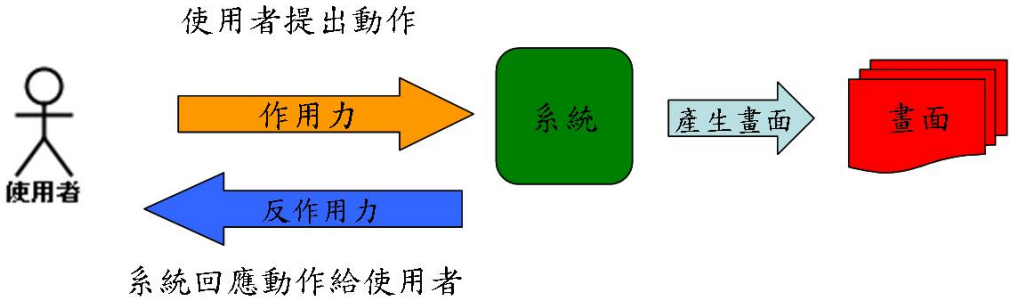


圖 7 畫面產生過程

在需求擷取時，我們以事件條例來描述表達事件之起始行為者或系統、動作及參與動作之物件等，其描述格式可表達如下：

主詞 ＋ 動詞 ＋ 受詞

通常主詞就是外部實體(行為者)或系統，是事件起始者或參與者。動詞是外部實體或系統之動作，描述事件要處理的工作。受詞表示事件完成之工作項目、表單或格式等。採用事件條列式的方式描述，此種結構化之寫作方式，有助於系統分析師閱讀與分析需求描述。[10] [19]

因此在需求擷取時，我們先將使用者需求的部份依照下列所提出之樣板，將所需之資訊擷取出來，但在與使用者做需求擷取時應注意上述之方式將需求描述出來，對於日後將使用者需求轉換成使用者案例時，將每個事件流的情境分別以使用者案例描述。

3.3 建立需求樣板

在需求樣板中，為了能夠快速開發及做情境的模擬，我們簡化一般需求規格書所需要的條件，而只取出有效率，對於系統有重大影響以及對於情境描述有幫助之部分，此部份定義如下：

表 1 需求樣板

項目	說明
名稱	此使用者案例名稱
編號	在此系統中的使用者案例編號。
描述	此使用者案例的功能介紹。
前置條件 (Pre-conditions)	進入此使用者案例時須先滿足之條件。
前置條件 (Post-conditions)	滿足此使用者案例才能繼續做的條件。
基本事件流	對於此使用者案例的步驟及過程說明，此事件流以主詞、動詞、受詞組成之句子為主。此事件流可細分為基本事件流、其他事件流、例外事件流。
其他事件流	使用者案例所發生之例外情形
例外事件流	使用者案例所發生之錯誤情形

3.4 需求分析

系統開發確認前，開發者如何與使用者進行完整的需求確認，減少系統開發後的修改，降低開發中的落差極為重要。因此透過圖型化的展示將系統雛型展示並與使用者進行討論，可以將系統需求更明確的與使用者進行溝通並確認。因此，我們將需求部份透過視覺化的方式進行輸入，利用點選式介面將使用者需求進行輸入，再經由系統自動辨識分析出元件、動作與操作事件。

根據先前我們所提的 主詞+動詞+受詞 方式先將觸發者(主詞)先找出來，再來就是尋找觸發事件(動詞)所影響之事件(受詞)，再依照我們根據強韌圖所訂定之延伸規則將需求轉換至強韌圖，其中分析之規則如圖 8：

1. 事件的觸發點為使用者或是介面。
2. 使用者在觸發事件會經由介面(一個動作的轉換, 由多個子畫面組合)將事件傳遞出去。
3. 事件傳遞至系統後，系統會回傳，則會產一個畫面。
4. 畫面為子事件的開端或是子畫面的回傳。
5. 系統根據使用者的要求，回覆事件(事件可能會有多個分支，即多個畫面產生)
6. 根據回覆事件回到第 1 點，即子畫面的開端。

基本流程圖如下所示:

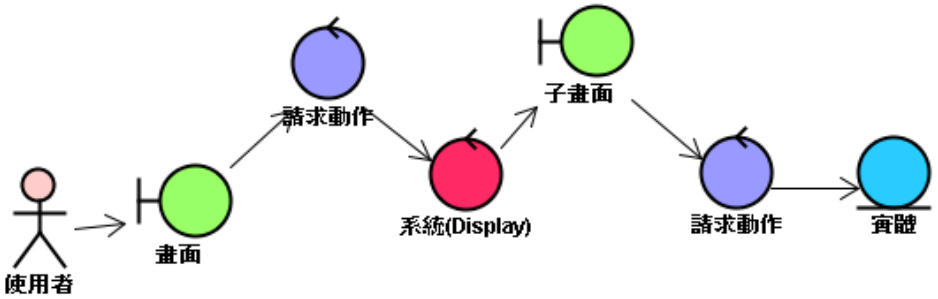


圖 8 需求強韌圖

分支流程圖如圖 9 所示：

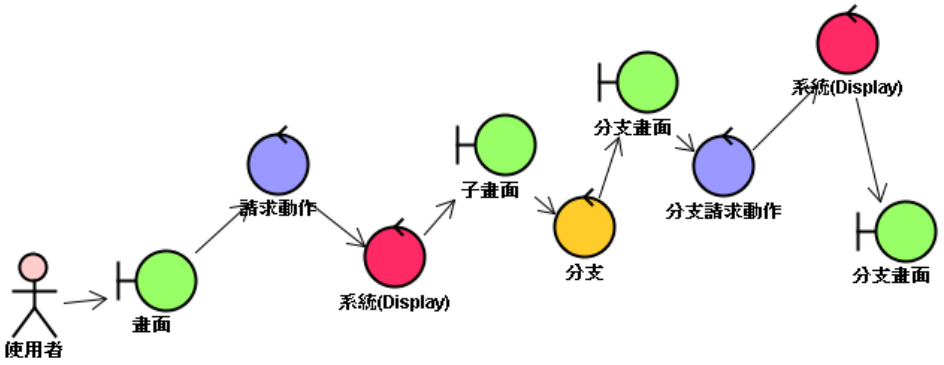
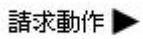


圖 9 需求強韌圖

在類別的分類上，我們將 Robustness 圖中的控制物件再細分為分支控制物件，一般控制物件以及系統控制物件，將事件的描述分解的更透徹，讓畫面的呈現能夠更清楚。表 2 為細分後各個物件所代表之涵義：

表 2 強韌圖表示方式

物件名稱	物件敘述	圖示
使用者	系統的操作者	
動作	使用者對系統所做出之動作	
介面	是行為者和系統交談的媒介	

分支(黃色)	若有其他事件發生時，可用分支物件來輔助。	
一般(紫色)	此物件為一個動作，因此可再細分為多個畫面	
系統(紅色)	系統的回應	
實體	用來表示儲存在資料庫中的資料	

3.5 強韌圖與循序圖

將需求擷取後，我們使用強韌圖來補足使用者案例(Use Case)與循序圖的之間的描述方式，得雛型開發→使用者案例→強韌圖→循序圖，變成一組故事系統，互相支援。而循序圖完成後，進入設計的階段將更為容易。

使用者案例在初始分析階段時，是採取使用者之觀點來描述系統提供之功能與定義系統內部之作業，通常是以事件條例式的方式加以詳述。使用個案描述可表達出行為者、行為者所輸入與動作之資訊以及行為者與系統的互動關係，利用分析模式可以找出系統的基本結構，透過此結構可以找出參與使用個案之三種型態物件，如圖 10 所示，可從使用個案功能描述中找出所屬的物件型態。例如使用個案條例式「客戶修改訂單資料，並顯示資料修改成功的訊息」，從這個事件條例式中可找出行為者是「客戶」，客戶要在一個修改訂單的「介面物件」中輸入資料，並且有處理修改的「控制物件」以及存放訂單資料的「實體物件」，最後系統將修改成功的訊息透過顯示的「控制物件」將訊息呈現在「介面物件」上。

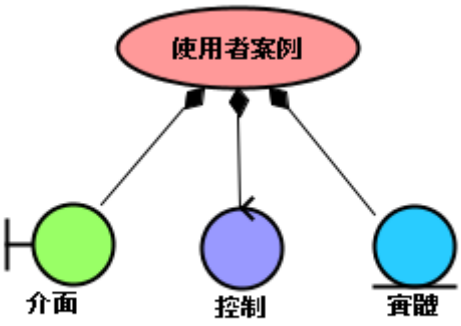


圖 10 使用者案例分割成三種型態的物件

我們利用這三種型態將分鏡開發時所需的元件以這三種物件來呈現系統的

關係,達到以分鏡的開發角度讓使用者可經由畫面呈現的方式將需求呈現給開發者及使用者觀看。其中如圖 11 所示，我們再將控制物件細分為分支物件、一般物件、系統物件將控制物件的描述更加詳盡，並且提供相對的功能敘述，讓需求的描述能夠更符合物件的要求。

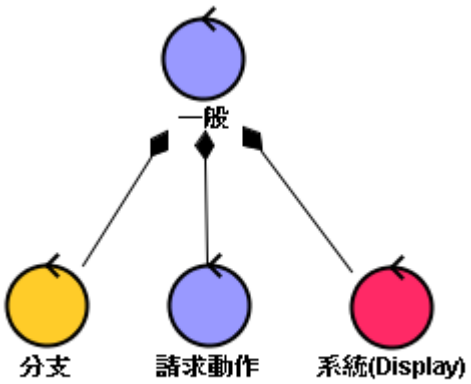


圖 11 控制物件細分圖

3.6 建立強韌圖

我們在描述需求時，利用建立事件流的方式以及牛頓運動定律之概念，進行以下的分析步驟：從基本事件之文句描述中，找出系統之行為者及介面、控制與實體等三種不同型態之物件與關係；進而依循強韌圖之物件連接規則建構強韌圖。這些步驟與方法描述如下：

3.6.1 找出行為者

基本上，強韌圖中之行為者即是使用個案圖中之行為者或活動圖中之外部實體。由於使用者必須與介面物件連接，不可與其他物件，因此在表示上，可由圖 12 表示。

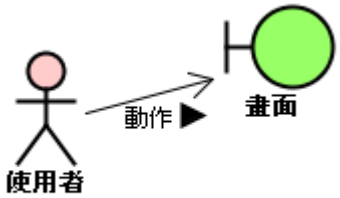


圖 12 使用者與介面物件之互動圖

3.6.2 找出物件、屬性與可能之操作

找尋物件、屬性與操作須從每一個使用個案之情節描述(或活動圖)中找出名詞、名詞片語與代名詞（以下通稱名詞）等，並依經驗法則判斷該名詞是否為物件；而從動詞可以找出物件可能之操作。因此在分析使用個案之描述時可以根據詞性來找尋，而分析活動圖時，可以依據活動圖元件的特性與詞性來幫助對物件、屬性與操作的找尋。如：“使用者按下檔案按鈕在空白視窗”這句話我們可以分析出使用者(主詞)按下(動詞)按鈕(受詞)在空白視窗(地方)，其相對應的強韌圖如圖 13 所示：

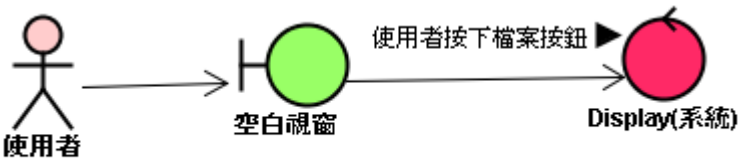


圖 13 使用者與介面、控制物件之互動圖

3.6.3 找出系統回應

依照先前兩個方法的描述，可將使用者及動作找出，此動作便有相對應的回應產生，系統物件因此會回應一個畫面或是動作給使用者，此畫面便可被捕捉到，而產生的動作也會有相對應的畫面產生，進而找到相關的實體物件。如：系統(主詞)丟下(動詞)選單的選項(受詞)，其相對應強韌圖如圖 14 所示：

圖 14 系統回應圖

若系統回應為其他事件，我們可以利用控制物件中的分支物件來表示事件的分歧點，而分歧點也會有相對應之事件或是畫面的產生。其相對應強韌圖如圖 15 所示：

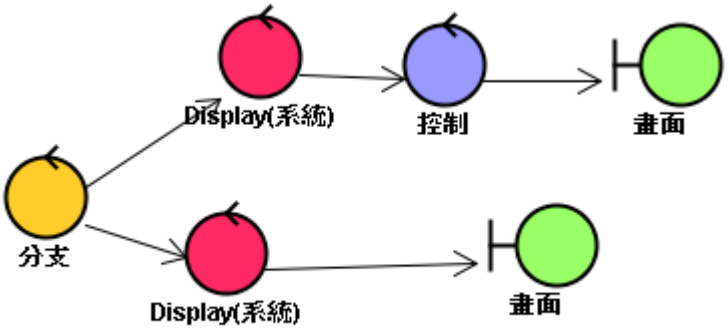


圖 15 分支畫面圖

3.7 建立循序圖

我們建立強韌圖後，便可將事件流的過程描述出，但是對於事件與事件之間的連結，無法清楚的描述，因此我們藉由循序圖的表示方法來幫助我們將事件之間的連結，即分鏡的連結。透過循序圖，我們可以表示控制物件中的一般控制物件所隱含的連續動作以及其他事件、例外事件所需表示的流程。在轉換中，我們將強韌圖中的介面元件、控制元件、實體元件以循序圖來表示，表示方法如圖16所示：

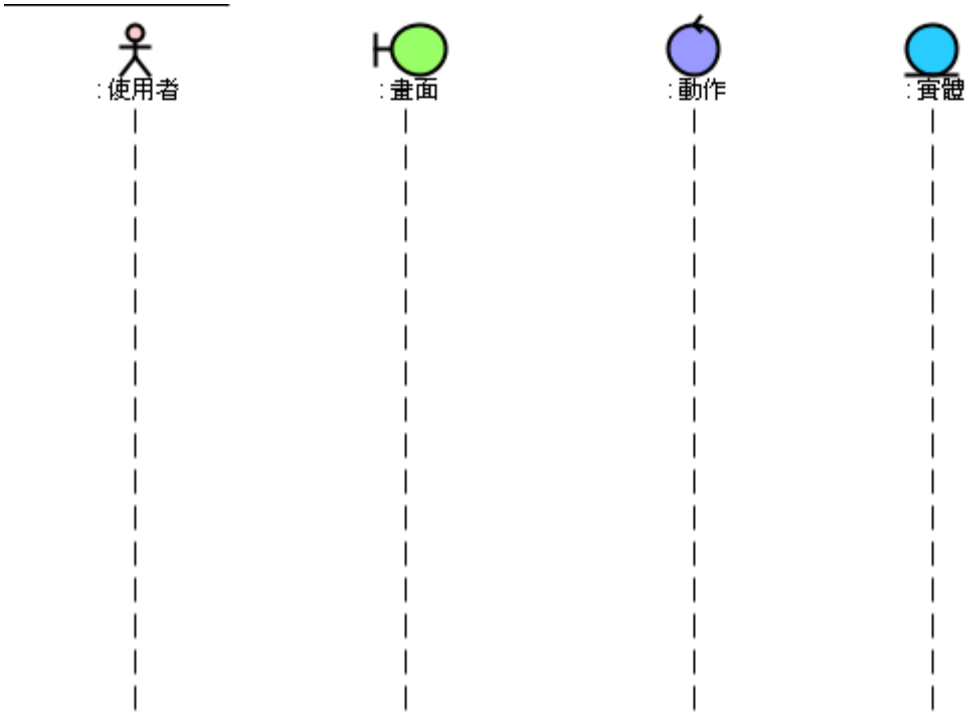


圖 16 元件表示圖

在循序圖中，各物件的功能分別有各自的表示方式，使用者透過畫面的連續性將傳遞訊息之功能呈現在此圖中，而此動作所產生之物件則為儲存資料庫之實體物件。每個物件之功能如下所示：

1. 使用者：此畫面之動作者。

2. 介面：此畫面為系統所產生之細節畫面，包括畫面的轉換過程，皆包含在裡面。
3. 動作：將相關畫面轉換的情境所對應之功能。
4. 實體：此功能所產生之實體物件。

我們將強韌圖轉換至循序圖的轉換方法分為以下兩種方式：

3.7.1 介面轉換

根據強韌圖所建立的圖形，我們可將圖形視為使用者與畫面的互動，並且會有實體物件的產生。其中的畫面轉換，可由強韌圖中的介面物件得到。其中的一般控制物件可由此圖展開來，進而有更細部的畫面呈現。圖 17 為強韌圖中物件的互動行為。

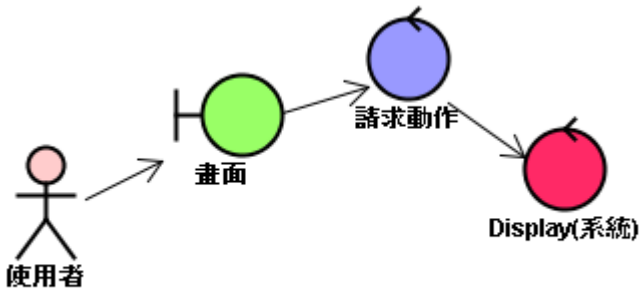


圖 17 強韌圖

透過圖形轉換後，我們可將物件轉換如圖 18 所示：

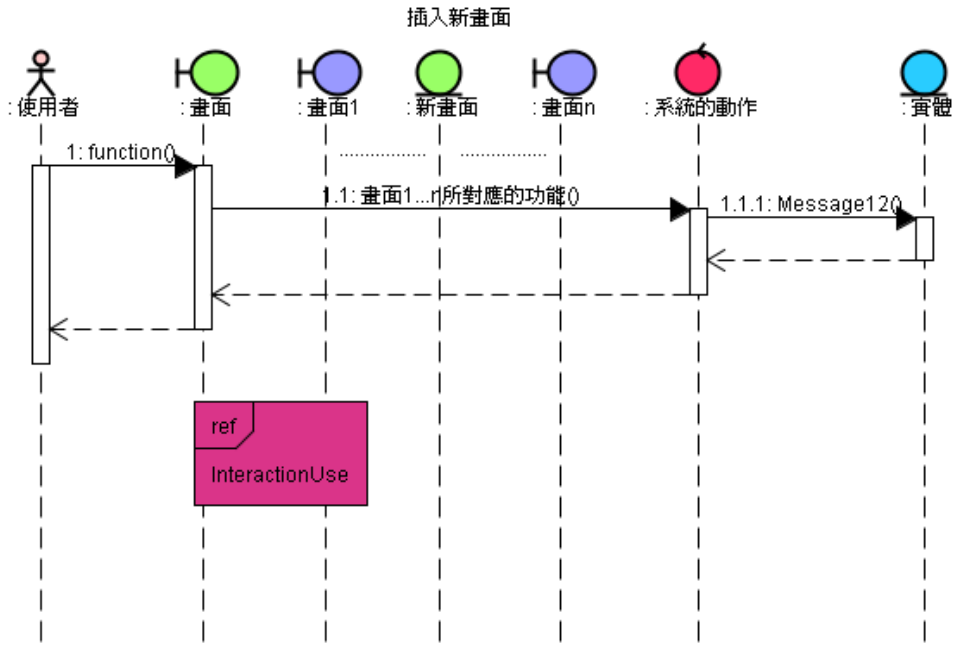


圖 18 循序圖

在此圖中，我們將強韌圖中的請求動作轉換成循序圖中連續的畫面，圖 18 中的新畫面為新的物件加入到此情境中或是替換原來情境中的某個子畫面，而此連續畫面 1…..畫面 n 所對應的功能為系統的動作，最後此圖形中的產出物為實體這個物件。

3.7.2 功能轉換

事件的轉換可視為強韌圖中描述物件之間的關係，將介面物件的連續畫面結合起來便是事件的功能，因此得到了事件流中所描述的場景，其相對應的功能。

3.7.3 產生實體物件

透過上述的兩種轉換方式，我們可以得到使用者與系統做互動時所產生的畫面以及動作，這些畫面以及動作的產出物便是實體物件。一般來說，此實體物件為儲存在資料庫中之資料。

3.8 分鏡連結

透過建立強韌圖及循序圖後，系統雛型畫面可由介面物件、控制物件與實體物件呈現，並且將相對應之功能對應到相關的畫面，進而產生相對應的情境。在情境的連結中，我們利用前置條件(Pre-Condition)與後置條件(Post-Condition)的描述當作分鏡與分鏡之間的連結。將分鏡之間的連結關係加至循序圖中，其表示方法如圖 19 所示：

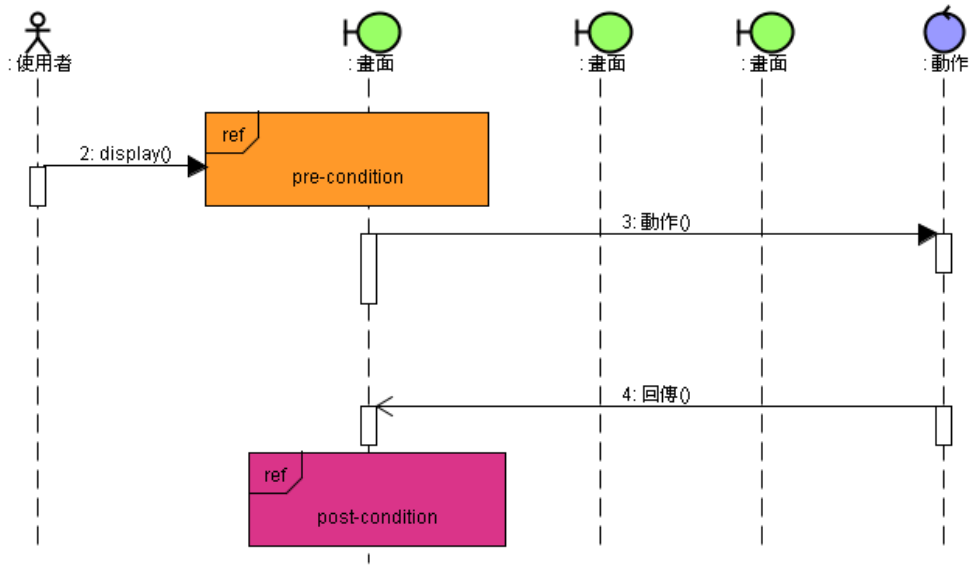


圖 19 Interaction use 表示方法圖

在此圖中，我們利用 Interaction Use 物件來幫助我們將各種不同的情境藉由此表示方法連接起來，在圖中，使用者要看顯示畫面物件時，須先參考前置條件(Pre-Condition)此外部連結，完成此外部連結之敘述後，才能繼續傳遞訊息。而動作物件完成動作後，便可滿足此後置條件(Post-Condition)之條件。使用此方法可快速的將分鏡的連結建立並且表示清楚。

在分鏡畫面與分鏡畫面的連結中，使用循序圖中的 Interaction use 物件，圖

20 中的 Interaction Use 物件加入到此分鏡畫面之前，用來表示與其它分鏡畫面的連結；而此分鏡也可做為其它分鏡之前置條件，對於畫面的銜接有很大的幫助。

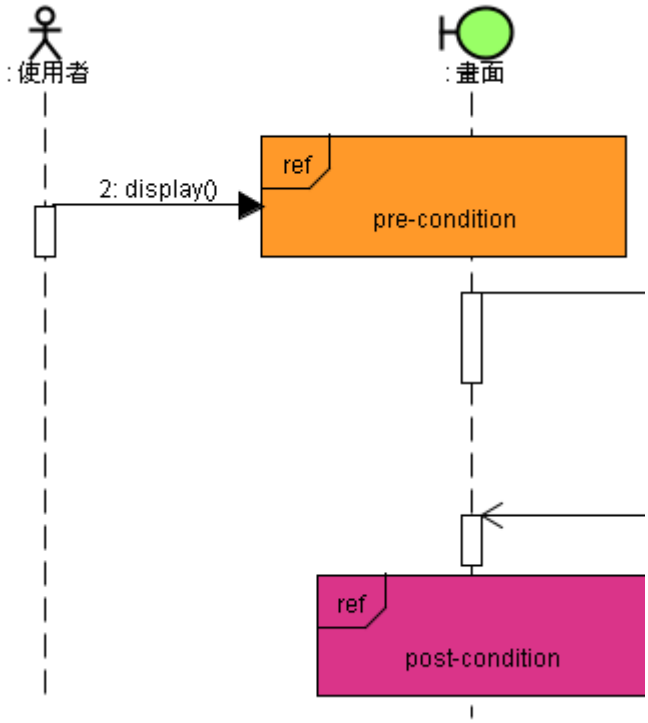


圖 20 分鏡畫面銜接圖

3.9 再利用元件

對於應用元件再利用的方法有許多種，像是直接在程式碼上做剪貼的動作(Copying And Pasting)、繼承類別、使用框架(Framework)或是應用程式介面(Application Programming Interface, API)等方式，但是對於大型的程式開發而言並非是好事，且對於日後軟體的維護以及相關的專案開發有極大的阻礙。在圖 21 中，我們針對此論文所提出之需求轉換成強韌圖以及強韌圖轉換成循序圖後，所得到的物件，將相關物件所組合之功能，視為一組元件，將來在畫面的抽離以及加入新的物件時，便可輕易的將所需要的畫面加入、修改或是刪除。

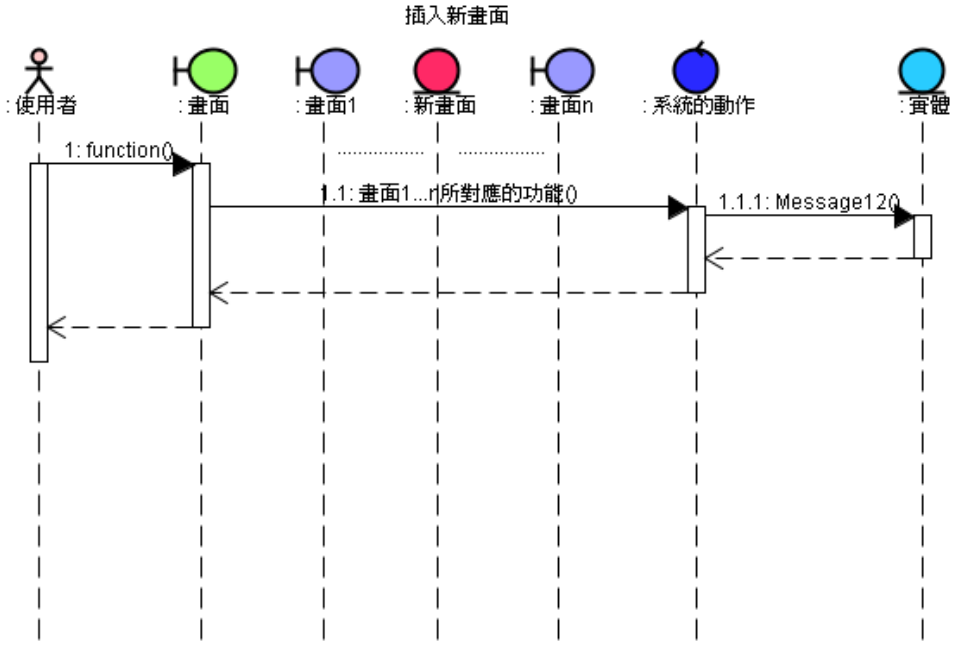


圖 21 再利用元件對應圖

在分鏡畫面中的事件所對應之元件，我們利用 package 的概念，將一組相關類別和介面的物件集合在一起，提供存取保護，可以讓其他類別使用套件中的類別和介面。簡單的說，套件是物件導向程式設計的零件庫，程式開發者可以直接選用套件中現成零件的各種物件，組合零件來建立物件集合。

我們將物件之間的連結利用繼承的方式串連起來，其中的介面為其他畫面的父類別，其中的子畫面則為子類別，透過此繼承關係，將各個元件之間的關係建立起來，將來在實作時，若是需要將物件作更改或是移除，將繼承的關係移除或是變更即可。圖 22 為介面物件轉換到類別物件之示意圖、圖 23 為控制物件轉換到類別物件之示意圖：

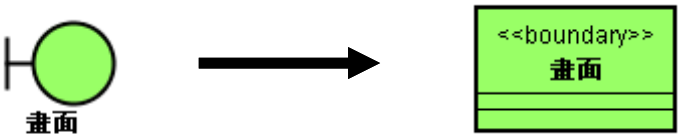


圖 22 介面元件與類別之對應圖

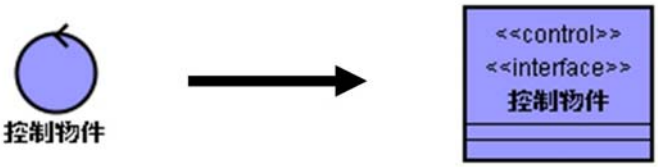


圖 23 事件元件與類別之對應圖

對於元件轉換至類別，則利用 UML 中的一般化關係、實現關係以及組成關係來表示，即物件導向中的繼承(Inheritance)、實作(Implements)及巢狀類別(Nested Classes)。在圖 24 中，我們將介面物件與控制物件中的請求動作關係宣告為繼承關係，此動作是用來表示一連串的動作是由主要介面去做延伸，讓控制物件繼承介面物件，控制物件則擁有此介面物件之操作及屬性。

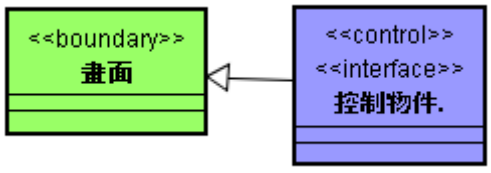


圖 24 介面物件與控制物件之關係

我們將控制物件中加入物件導向中的介面(Interface)，如圖 25 所示，此動作可替類別的物件提供共同介面，就算類別間沒有任何關係，一樣可以擁有共同介面。，藉由實作類別的介面，讓元件中的關係能夠透過介面去將功能實現，若是有新的需求要加入或是要更改，可直接實作此物件即可，不需要重新再建造一個新的物件。

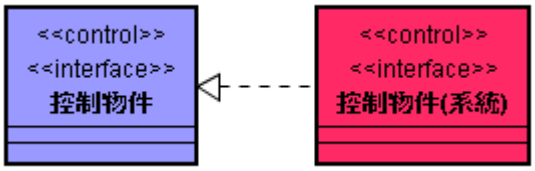


圖 25 控制物件與系統物件之關係

根據前面的動作，系統會回應子畫面給使用者，子畫面為一至多個畫面所組合而成，這些子畫面為系統回應所組合而成，利用巢狀類別強調類別之間的關聯性，強調內層類別一定需要外層類別，如果外層類別的物件不存在，內層類別物件也不會存在，即巢狀類別的內層類別是外層類別的零件，子畫面為其組合元件之一。其表示方式如圖 26 所示：

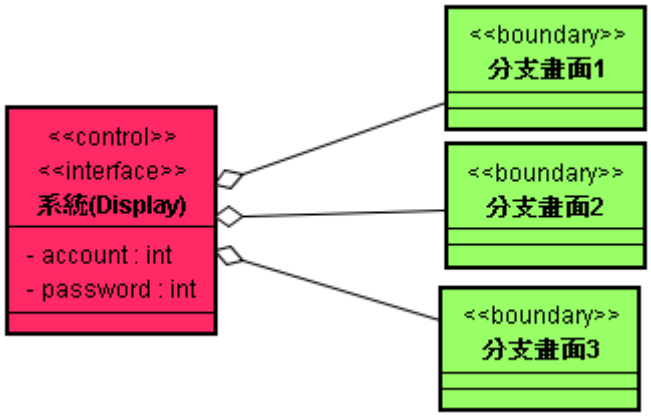


圖 26 系統物件與介面物件之關係

圖 27 為元件轉換至類別時，所進行轉換之對照圖：

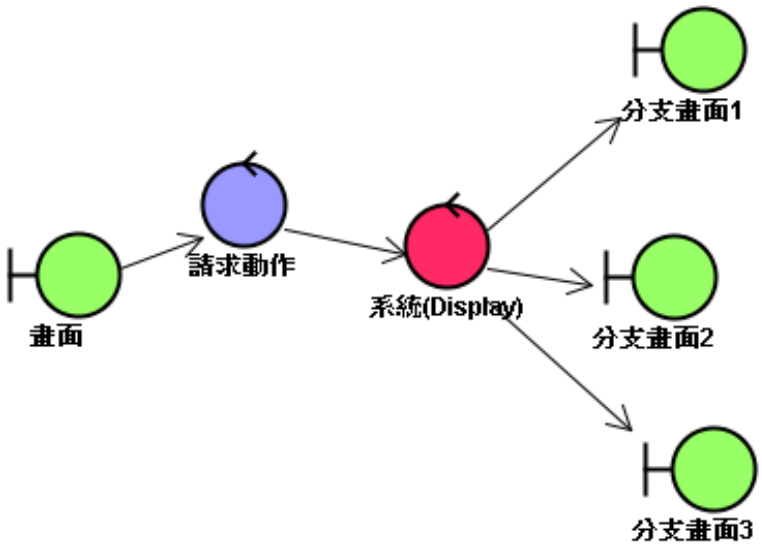


圖 27 元件對應類別圖

圖 28 為元件轉換至類別後，各個物件之間的關係表示方式：

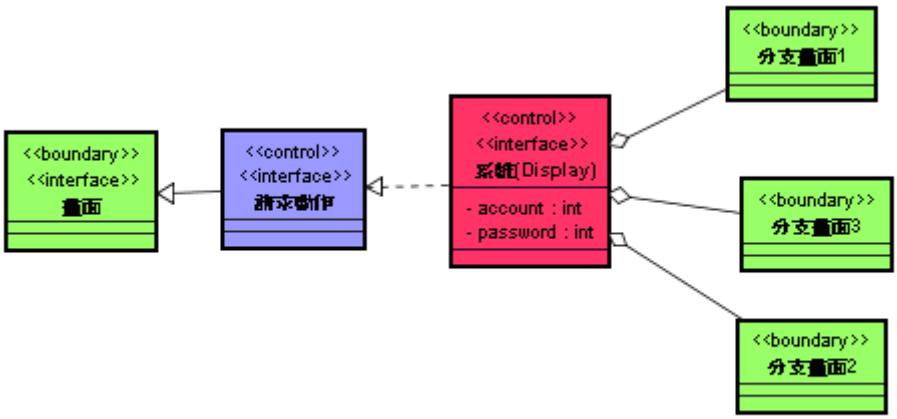


圖 28 元件對應類別圖

第4章　案例研究

我們將以 Digital Video Recoding (DVR) System 進行測試研究，運用本研究需求轉換編輯器(Requirement Editor)，將需求部份進行轉換，並透過分鏡開發法，將元件、屬性以及操作事件自動分解，運用元件再利用技術，將現有相關元件套件進行重新組合，產出系統雛型，並經由情境模擬，將系統操作流程進行雛型展示，並產出可交給使用者之系統需求確認雛型。DVR 系統可分為下列主要部份：

1. Video Streaming Server (VSS)
2. Remote Monitor Client (RMC)
3. Parallel Video Encoder(PVE)

其中，VSS 用來整合具可平行工作的特性的應用，其主要功能包含：提供多重使用者連線、提供多重資料查詢及多重即時或非即時影像傳輸。而 PVE 則是用來整合具可平行處理資料及可平行處理資料流的特性的應用。其主要功能包含：多重影像擷取及同步壓縮。RMC 則是遠端連線介面，除了查詢遠端伺服器的資料外，主要功能為：播放即時影像、重播錄影檔案。

根據一開始的模型樣板將需求規格書中最主要的部份給擷取下來，例如功能性需求、系統的描述等等。根據這些描述再依照操作者及開發者的需求分別各自擷取需求。針對操作者的使用習性及功能上的需求，以塑模語言 UML 來幫助我們了解操作者對於提出的需求是否符合開發者的期望，而開發者對於操作者所提出的需求能夠以視覺化的方式呈現出，以確認需求是否正確。

以下為本論文之研究案例，首先將依照需求樣板所需資訊將內容擷取下來，在針對所擷取之需求依照所分類方式選出合適的需求。以下是以 Digital Video

Recoding (DVR) System 進行各項操作之需求樣板，我們將 DVR 系統分為 5 個場景，依照各個場景的敘述，將需求樣板完成，表 3 為其中遠端登入之情境敘述：

表 3 遠端登入情境

項目	內容
名稱	遠端登入
編號	UC01-04
描述	使用者透過網路遠端登入 RMC
前置條件 (Pre-conditions)	使用者有帳號及密碼
後置條件 (Post-conditions)	使用者可查詢影像，更新影像
活動的基本事件流	1.在(User Login in interface)使用者登入介面下，使用者(user)在登入欄位鍵入(keyed in)帳號密碼在按確認按鈕(send button).系統顯示(display)登錄成功
其他事件流	1.使用者按送出按鈕前，若按取消按鈕，可將登入欄位的帳號密碼消除
例外事件流	1.系統登入失敗，使用者重新登入帳號

我們將遠端登入之使用者案例利用轉換的方式將表 3 轉換為強韌圖，其轉換圖如圖 29 下：

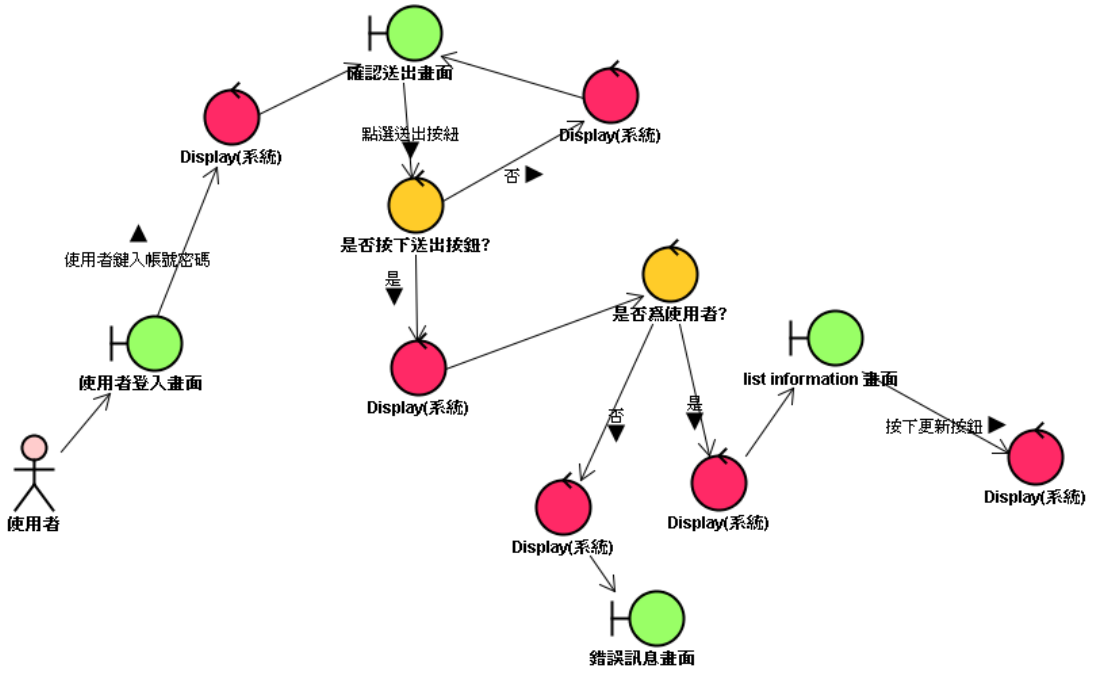


圖 29 遠端登入之強韌圖

系統初始轉換畫面如圖 30 所示：

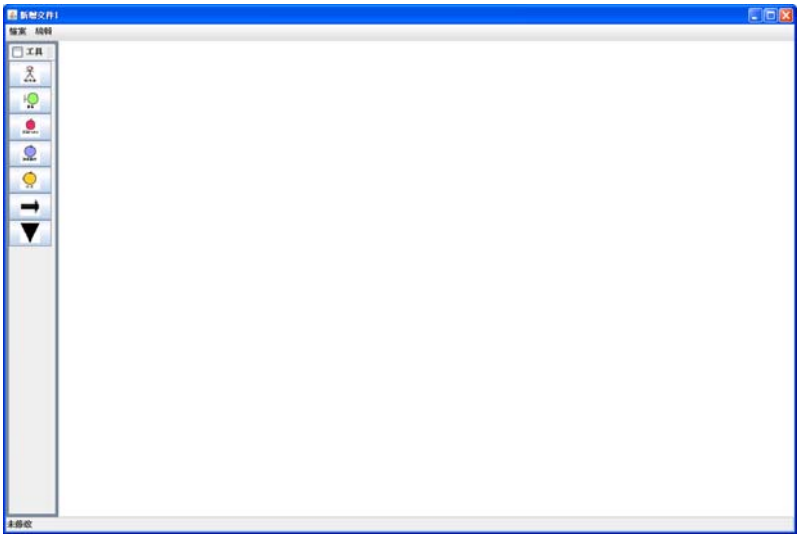


圖 30 系統畫面圖

我們先建立需求樣板，將所需的資訊填至欄位中，並且透過此樣板將資訊先轉換成基本的強韌圖，再依照輸入的資訊在強韌圖中做增修。如圖 31 所示：

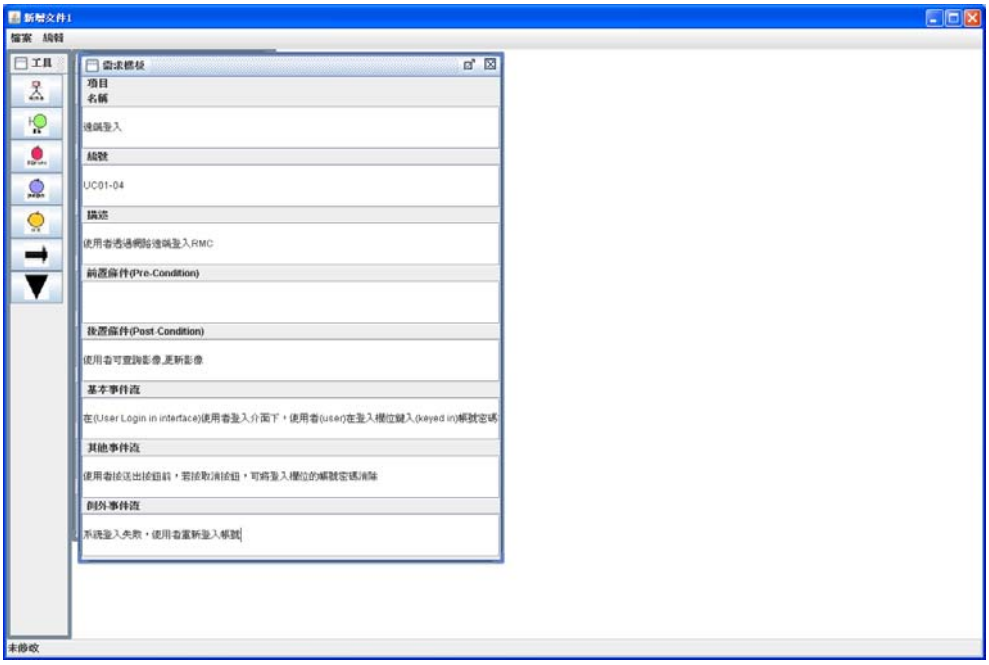


圖 31 系統畫面圖(需求樣板)

圖 32 為系統中，我們產生基本的強韌圖流程，使用者可依照最基本的流程去做新增、修改、或是刪除。圖 33 為將強韌圖轉換成圖片。

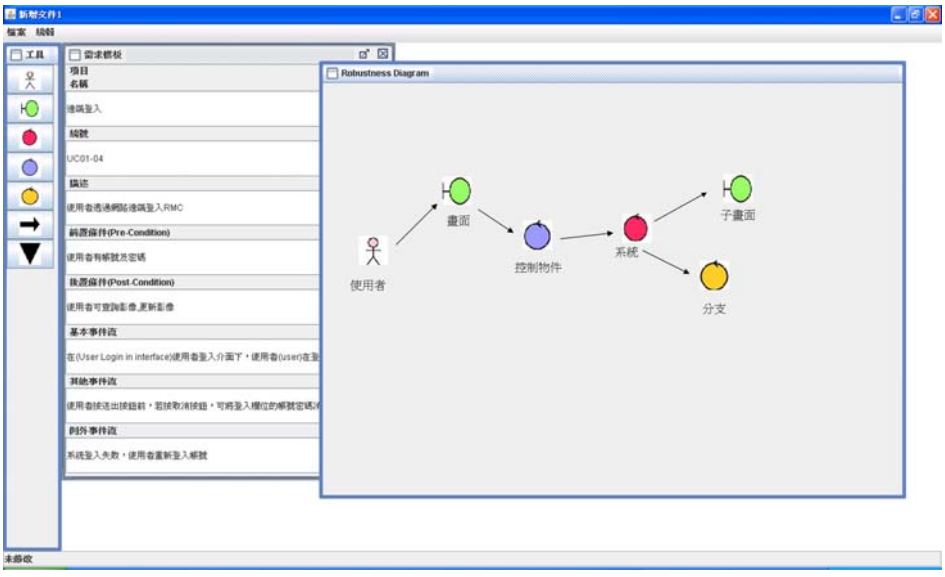


圖 32 基本強韌圖流程

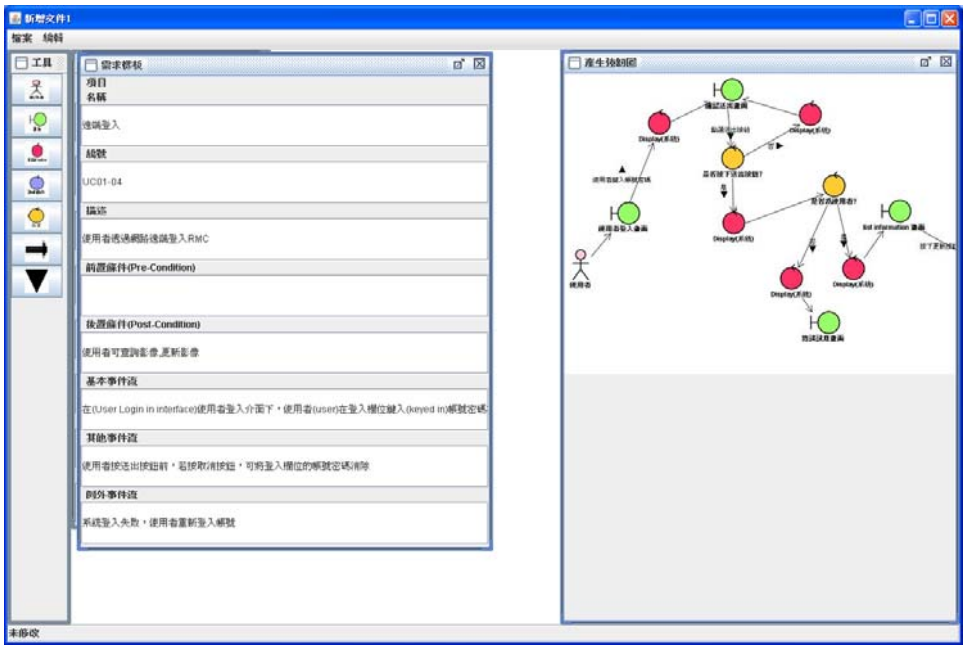


圖 33 系統畫面圖(產生強韌圖)

將強韌圖完成後，我們便可依照強韌圖的規則將循序圖建立，並且根據需求樣板的敘述,利用前置條件以及後置條件將情境之間做連結。其循序圖如圖 34 及圖 35 所示：

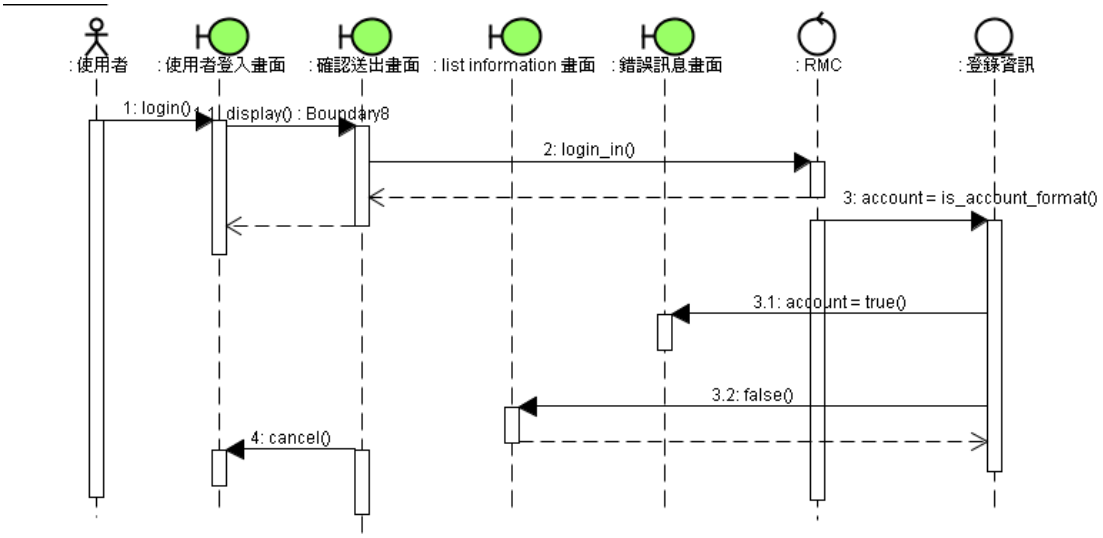


圖 34 遠端登入之循序圖

加入前置條件以及後置條件之敘述後的表示方法如圖 36 所示：

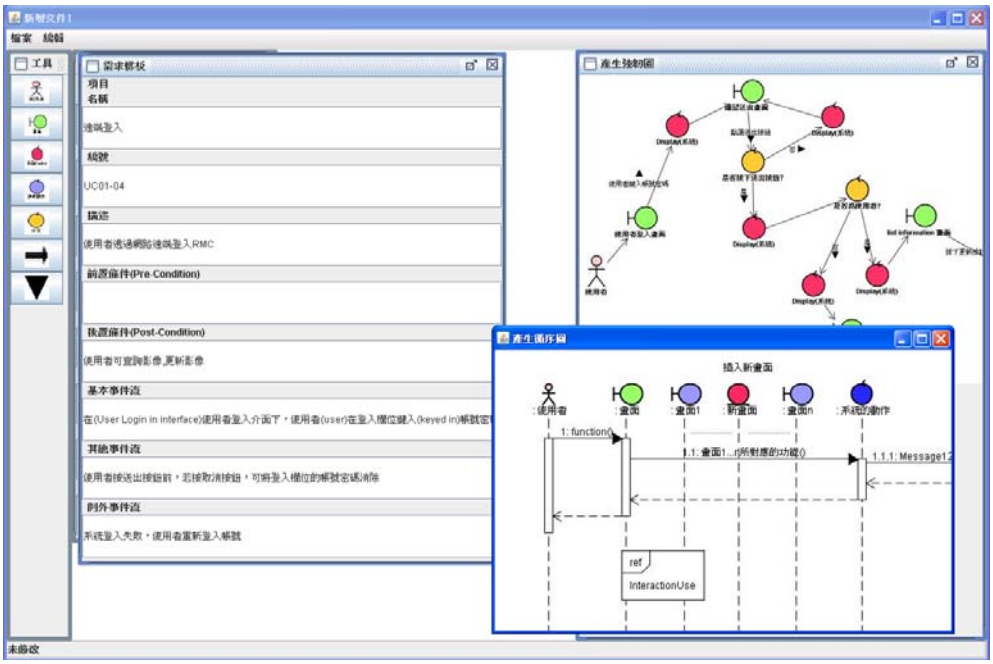


圖 35 系統畫面圖

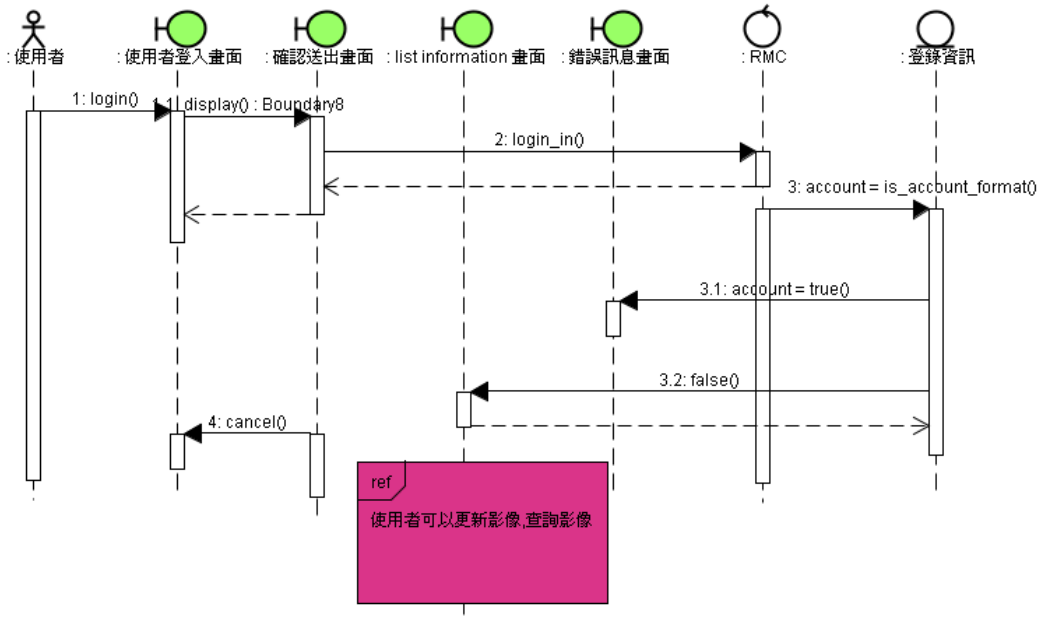


圖 36 遠端登入之循序圖

透過我們所提出之方法，我們依序將以下之情境，將強韌圖以及循序圖完成後，再將各個強韌圖連接在一起，讓故事情境能夠完整的呈現。

將循序圖以及強韌圖之結果透過元件化的處裡後，將畫面以及事件的類別描述出，使用者再登入畫面時，會有輸輸入帳號密碼以及按送出按鈕的動作，因此在使用者登入畫面介面的表示中會有 Button 元件 Textfiled 元件產生。而控制物件中的使用者鍵入帳號密碼在動作敘述上，則可發現有使用者的帳號及密碼屬性，如圖 37 所示。後續的子畫面則實作(Implements)其控制類別，元件只需實作其所需要的功能或是屬性，而不需要重新建構新的物件，因此可提升元件的再利用性。

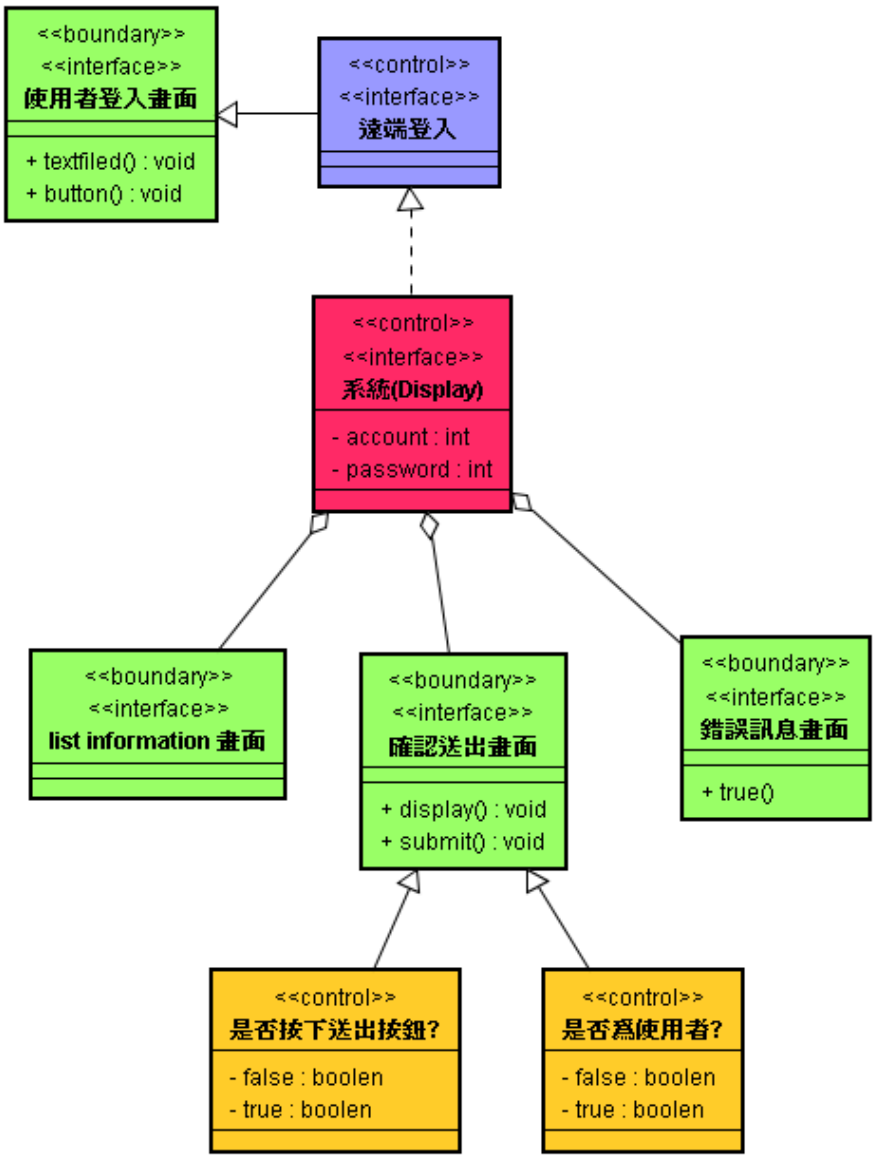


圖 37 元件類別圖

在此事件中，我們從需求截取出使用者的帳號以及密碼等屬性，並建立相關的操作至類別中，並且建立各個物件之間的關係。透過使用者介面擷取需求，讓情境能夠以物件的方式來描述及組合，並在需求擷取過程中將屬性、操作、及關係能夠快速建立及元件化。

第5章　結論與未來方向

由於市場的需求以及系統發展趨勢會受到開發時程(Time To Market)的影響，因此減少系統開發時程，將會減少開發成本，進而提高產品價值。因此為了縮短開發時程，我們使用了雛形開發方法，使用此種開發方法雖會遺漏系統中某些屬性或是相關的連結，但可快速了解系統全貌。本研究中進行需求文件之探討並且提出分鏡應用於雛型開發上，除了能縮短開發時程，對於開發者與客戶在溝通上，將可透過階段式開發雛型展示提供更準確的系統開發。對於軟體系統的快速開發及縮短時程提供一套不同以往的開發方法。一般而言，開發軟體時，都會先蒐集需求，再經由分析來得到系統規格。而現今大部分需求都是利用自然語言來描述，因此可能會產生不一致或是不明確的需求，進而增加開發成本。本論文利用牛頓法來描述需求，並加入情境法的描述，一來不會對於系統與使用者之間的傳遞有不明確的描述，而開發出與顧客的需求有所出入的系統；二來對於需求描述也更完整，減少需求資訊的短缺而開發出不完整的系統。另外利用強韌圖來規劃需求的使用狀態，再利用循序圖的描述，讓畫面的動作更流暢，這樣不但可以得到最主要的需求資訊，也不會有模稜兩可的問題產生。而以後如果有相同的需求時，便可以利用之前開發過的系統所產生的再利用元件，減少開發成本。而當中利用物件導向的概念結合物件與事件，只要當中的物件產生更動，那麼修改實作關係便可以重新使用原件。由於在需求階段便已經導入了物件導向的精神，將物件的關係連結起來，未來可以再進一步的將自動化的程序導入至圖形之間的轉換，進而整合至軟體開發階段，讓開發流程能夠更完善。

參考文獻

[1] Alexander, C. et al., A Pattern Language, Oxford Univ. Press, 1977.

[2] Ali. A.B.H., Boufares. F. & Abdellatif. A., "Checking Constraints Consistency in UML class diagrams" *Information and Communication Technologies*, 2006, Vol. 2,Page(s):3599-3604.

[3] Campbell, R.L., "Will the real scenario please stand up?" *ACM SIGCHI Bulletin*, Vol. 24 , Issue 2 ,April 1992, Page(s):6-8.

[4] Chu, C.W., Lu, C.W., Chang, C.H., & Chung, Y.C., "Pattern based software re-engineering," *the Proceedings of 1999 Asia Pacific Software Engineering Conference (APSEC'99)*, Dec. 1999, Page(s): 300-308.

[5] Femandez, E.B., "Building Systems Using Analysis Patterns," *Proceedings of the 3rd international Software architecture workshop, ACM 1998*, Page(s):37-40.

[6] Gamma, E., Helm R., Johnson, R., & Vlissides, J., *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.

[7] Hooper, J.W., & Hsia. P., "Scenario-based prototyping for requirements identification", *ACM SIGSOFT Software Engineering*, Vol.7, No. 5, Dec. 1982. Page(s):88-93.

[8] Jacobson, I. et al., "*Object Oriented Software Engineering: A Use Case Driven Approach*," Addison-Wesley Professional,1992.

[9] Jwo, J.S., & Cheng Y.C., "Pseudo Software: a New Concept for Iterative Requirement Development and Validation" 14th Asia-Pacific *Software Engineering Conference 2007*, Dec. 2007, Page(s):105-111.

[10] Kof, L., "Scenarios: Identifying Missing Objects and Actions by Means of Computational Linguistics" *Requirements Engineering Conference*, Oct. 2007 Page(s):121-130.

[11] Maiden, N., "User Requirements and System Requirements," *IEEE Software*, Vol. 25, Issue 2, March 2008 Page(s):90-91.

[12] Mavin, A., Novak, M., Wilkinson , Philip., Maiden, N., & Lynch, Perry., "Using Scenarios to Discover Requirements for Engine Control Systems" *International*

Requirements Engineering, 2008, 8-12 Sept. 2008 Page(s):235-240.

[13] Maiden, N., "CREWS validation frames: Patterns for validating systems requirements," *Understanding Patterns and Their Application to Systems Engineering*, Volume, Apr 1998, Page(s):212-217.

[14] Maiden, N., & Mavin, A., "WHERE: Scenario Modeling, Answering the Six Questions about Requirements, " *The Institution of Engineering and Technology Seminar*, 3 Oct. 2006 Page(s):35-44.

[15] Pelechano,V., Pastor, O., & Insfrán, E., "Automated code generation of dynamic specializations: an approach based on design patterns and formal techniques," *Data & Knowledge Engineering*, Vol. 40, No. 3, March 2002, Page(s):315-353

[16] Prechelt, L., Unger, B., Tichy , W.F., Brossler , P., & Votta , L.G., "A controlled experiment in maintenance: comparing design patterns to simpler solutions," *IEEE Transactions on Software Engineering*, Vol. 27, No. 12, Dec. 2001, Page(s):1134-1144.

[17] *Robustness diagram* (Online information: retrieved May 24,2009)
http://www.agilemodeling.com/artifacts/robustnessDiagram.htm

[18] Rosenberg, D., & Scott, K., *Use Case Driven Object Modeling with UML*, Addison Wesley,1999.

[19] Rolland, C., Souveyet, C., & Achour, C.B., "Guiding goal modeling using scenarios" *IEEE Transactions on Software Engineering*, Vol. 24, Issue 12, Dec. 1998 Page(s):1055-1071.

[20] *SCENARIO* retrieved (Online information: retrieved May 28,2009)
http://uoid.drhhtang.net/Home

[21] Szlenk. M., "Formal Semantics and Reasoning about UML Class Diagram" *International Conference on Dependability of Computer Systems* 2006,May 2006, Page(s):51-59.

[22] Wahono, R.S., & Cheng, J., "Extensible Requirements Patterns of Web Application for Efficient Web Application Development," *Proceedings of the First International Symposium on Cyber Worlds (CW.02)*, Nov. 2002, Page(s): 412- 418.

[23] Yuan, J., Miao, H., & Cai, L., "A design pattern verifier in two-tier programming environment," *the 4th International Computer and Information Technology*

Conference (CIT), Sept. 2004, Page(s):1081-1086.

附錄

表 4 更新影像情境

項目	內容
名稱	更新影像
編號	UC01-01
描述	使用者透過網路遠端遙控去更新符合的影像
前置條件 (Pre-conditions)	使用者先登錄 list information 畫面
後置條件 (Post-conditions)	使用者可查詢影像
活動的基本事件流	1. 在 List Information 介面下，使用者(user)按下(clicked)更新按鈕(update button). 系統顯示(display)更新影片日期及時間.
其他事件流	無
例外事件流	1. 系統更新失敗，使用者確認

表 5 查詢影像情境

項目	內容
名稱	查詢影像
編號	UC01-02
描述	使用者透過網路查詢符合的影像
前置條件 (Pre-conditions)	使用者先登錄 list information 畫面 使用者先更新影像
後置條件 (Post-conditions)	使用者可播放影片
活動的基本事件流	1.在查詢列上，使用者(user)鍵入(keyed in)影像日期/Camera 編號，在按下(clicked)查詢按鈕(search button).系統顯示(display)日期及影片
其他事件流	無
例外事件流	1.按下查詢按鈕後，系統查詢不到影片，請使用者確認

表 6 播放即時影像情境

項目	內容
名稱	播放即時影像
編號	UC01-03
描述	使用者透過網路遠端遙控去更新符合的影像
前置條件 (Pre-conditions)	使用者先登錄 list information 畫面 使用者先更新影像
後置條件 (Post-conditions)	無
活動的基本事件流	1.使用者(user)按下(clicked)影像/文字超連結在重播影像欄.系統(System)顯示(Display) Camera 影像在(播放介面)Video Displayer 上
其他事件流	1. 使用者(user)按下(clicked)影像/文字超連結.系統(System)顯示 (Display)等待影像在(播放介面)Video Displayer 上
例外事件流	1.系統連結失敗，請使用者在使用

表 7 播放重播影像情境

項目	內容
名稱	播放重播影像
編號	UC01-05
描述	使用者透過網路遠端遙控去重播影像
前置條件(Pre-conditions)	使用者先登錄 list information 畫面 使用者先更新影像確認是否有影像存在資料庫中
後置條件(Post-conditions)	無
活動的基本事件流	1.使用者(user)按下(clicked)影像/文字超連結.系統(System)顯示(Display) Camera 影像在(播放介面)Video Displayer 上.
其他事件流	1. 使用者(user)按下(clicked)影像/文字超連結.系統(System)顯示 (Display)等待影像在(播放介面)Video Displayer 上
例外事件流	1.系統連結失敗，請使用者在嘗試

圖 38 為 DVR 之情境繪製圖，由此圖，我們可清楚知道每個畫面與事件之間的連結，對於故事的畫面便有一套清楚的敘述，再透過循序圖的支援，將故事中的情境與情境之間的銜接點給連結起來，對於故事的敘述能夠更加的完整，透過連結各個情境圖，產生以下完整的故事全貌：。

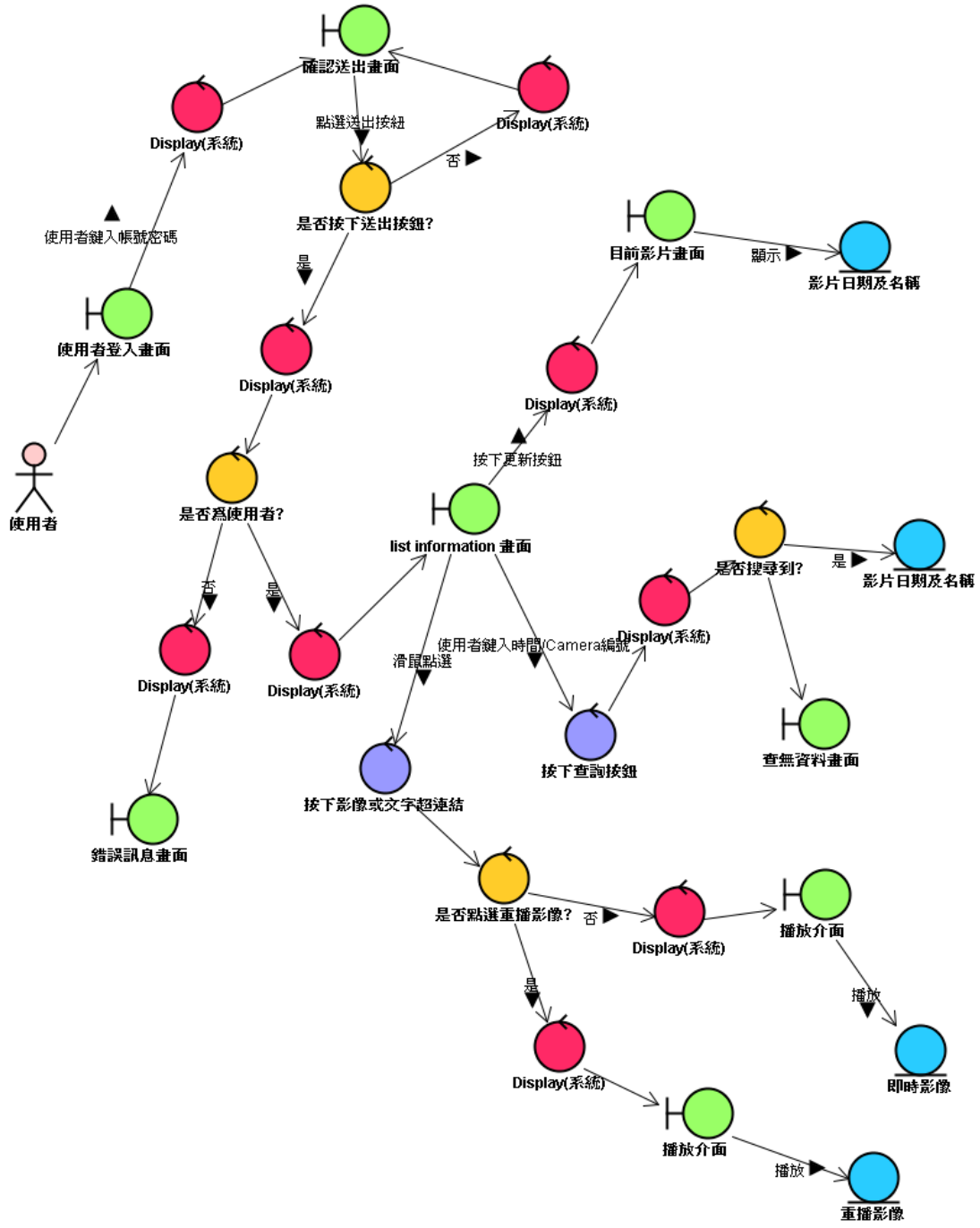


圖 38 DVR 之強韌圖