行政院國家科學委員會專題研究計畫　成果報告

子計畫四:整合交互教學和直接教學的網路合作閱讀環境

(3/3)

計畫類別： 整合型計畫
計畫編號： NSC93-2520-S-003-005-
執行期間： 93 年 08 月 01 日至 94 年 07 月 31 日
執行單位： 國立臺灣師範大學資訊教育系(所)

計畫主持人： 黃冠寰
共同主持人： 宋曜廷
計畫參與人員： 黃冠寰 宋曜廷 游智皓 林哲生 張道顧 陳彥傑 張昇賀

報告類型： 完整報告

處理方式： 本計畫可公開查詢

中　華　民　國 94 年 10 月 14 日

行政院國家科學委員會專題研究計畫成果報告

整合交互教學和直接教學的網路合作閱讀環境

計劃編號：NSC 92-2520-S-003-005-
執行期限：93 年 8 月 1 日至 94 年 7 月 31 日

主持人：黃冠寰
執行機構及單位名稱：國立台灣師範大學資訊教育學系

一、 中文摘要

本計劃研究如何將直接教學、交互教學、或交互教學融入直接教學之策略於 Internet 上執行。第一部分是研究如何根據 Internet 及網路環境的特性來制定出直接教學、交互教學、或交互教學融入直接教學所需的教學流程及教學環境。第二部分是運用群組軟體（Groupware）及網際網路運算（Internet Computing）的技術於直接教學和交互教學來設計出資訊化教學模式，並套用在現有的教學上。第三部份根據系統的需求，增加程式設計的易更改性及可攜性。第四部份是設計系統架構，我們實作出一個系統，名為 JOO-WfMS (Java Oriented Object-Workflow Management System) 。其為利用 Java 物件導向程式語言之特性並針對線上交互教學系統所需提供的教學、測驗、檢視以及再學習功能所設計出來的工作流程系統 (Workflow Management System)。

二、 緣起及目的

在閱讀教學中，交互教學是少數經過長期的檢驗與修訂，並經過驗證提升學生的閱讀能力的有效果教學方法。近年來，則有不少學者發現，在交互教學當中，如果能融入直接教學（Direct instruction），教學效果往往要優於單純的交互教學。交互教學除了應用在閱讀外，亦有若干學者認為其強調後設認知和自我調整的特色，也適用於寫作的訓練上。

另外一方面，我們根據教學十大原則中的「熟練原則」，讓學生對於所學習的事物，做到真正的學習。從知識思考方面而言，就是指記憶熟練、理解透徹、判斷清楚、思考細密。而美國中等教育專家莫禮生根據其所創設的熟練原則，擬定了一個「熟練公式」，即：瞭解學生程度→教學→測驗教學結果→調整教學手續→再教學→再測驗，達到真正的學習點為止。由這項公式我們得知，每次教學要徹底達成教學目的，在學習的過程中，教師要注意學生困難和錯誤，並予以協助和矯正。在教學之後，還要進行測驗，倘若學生的成績還沒有達到十分純熟，就應該繼續教學，直到目的徹底實現為止。為了讓學生達成目的，我們使用了「自學輔導法」，也就是學生在教師的指導之下，而進行自學的方法。道爾頓制(The Dalton Plan)是美國柏克赫斯特女士 (Miss Helen Parkhurst)所創。因為柏氏不滿意教師講解的教學方法，在一九一一年為八歲至十二歲的兒童擬訂了一個教育實驗室計畫。其主要宗旨是破除上課時間和教室教學，每一學科先由教師規定各年級之作業項目，稱為「公約」，星期開始時，教師根據學生的測驗成

績，指定學生從某一公約開始學習，當每一學生學完了某一個公約，然後由教師加以測驗，各科及格後，即可進行下一個公約。

三、 資訊化教學模式

我們利用道爾頓制和自學輔導法設計出一套線上教務交互教學系統，每當教師在課堂上進行完課程教學後，教師根據課程內容和學習觀念做規劃，將測驗學生的題目放在教務教學系統上，學生便可在課餘時間透過網際網路進行自學測驗。如 **Fig. 1** 所示：

1. 透過網際網路，教師將學習流程安置在教務教學系統上。每個學生都遵循教師所擬訂的教學流程進行測驗學習。
2. 依照學生自我學習進度的不同，每個學生擁有自己的一套學習流程。如 Fig. 1，學生 A 在流程上學習到 C1，還在剛開始的階段；學生 B 的進度到了 C3，快要結束流程的學習；而學生 C 到達了 C2.1 的進度。
3. 學生有各自的進度，教師可以於適當的時間和學生進行互動。例如學生 A 進行到 C1 並測驗完後，教務教學系統在 T1 自動確認學生的答題情形，未達標準系統自動將學習流程送回 C1 要求學生再度學習測驗；如果達到既定標準即可過關然後將流程遞交給教師審核，教師即可利用檢視頁面觀察學生的答題情況並給與協助和互動。
4. 學生所有的答題情況都會記錄在資料庫中。例如學生 B 在 C3 的測驗中，假設題目共有 10 題，及格標準 80 分，但是 C 在第一次測驗只有達對 6 題，也就是只有得到 60 分沒有達到標準。教學系統會將這次的情形紀錄在資料庫中。例如：「學生 C，於觀念 C3，第一次測驗，60 分，不及格」。而 C 重新測驗後，達對了 9 題，得到 90 分超過標準，系統便會允許通過測驗並將此次結果記錄在資料庫中。例如：「學生 C，於觀念 C3，第二次測驗，90 分，及格」，然後送交教師審核。

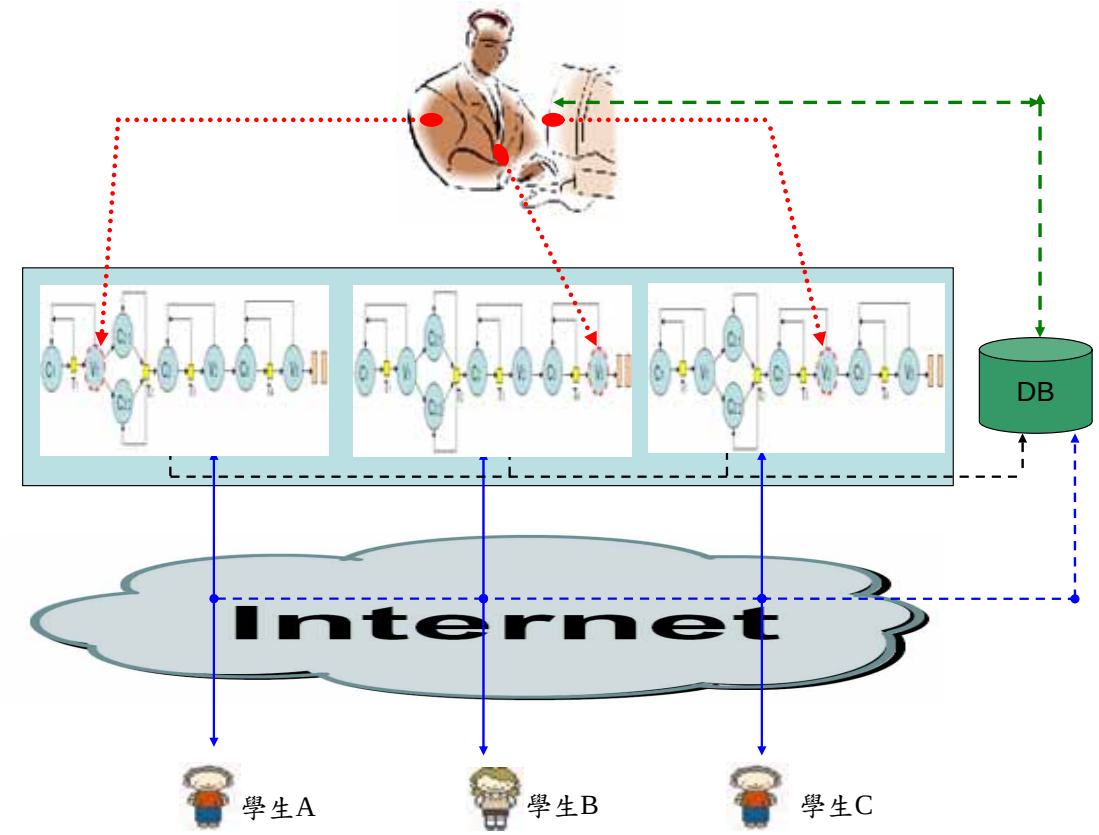

Fig. 1：網際網路交互教學模式

以九年一貫課程五年級下學期數學科的「整數四則」為例，主要觀念分為三個部份：「由左向右算」、「先乘除後加減」、「碰到括號要先計算」。在教師在課堂上進行講解後，利用我們所設計的學習測驗系統，教師可針對課程內容分配學生測驗的階段：因為在第二個觀念「先乘除後加減」，在此學習單元中佔有相當大的比例和重要性，為了讓學生更熟悉此部分的數學運算方法，所以我們將「先乘除後加減」又分為兩個部份，分別為「先算乘再算加減」和「先算除再算加減」。教師針對學生學習上的需求作完分析後，將此套課程測驗設計成五個部份，分別為：

1. 「觀念一(C1)：由左向右算」。
2. 「觀念二之一(C2.1)：先算乘再算加　減」。
3. 「觀念二之二(C2.2)：先算除再算加減」。
4. 「觀念二(C2)：先乘除後加減」。
5. 「觀念三(C3)：碰到括號要先計算」。

流程圖如下 **Fig. 2** 所示，學生登入後，每作答完一個觀念的題目，系統會將學生先暫停在此觀念，然後將學生的答題情形紀錄在資料庫中，送交給教師，讓教師審核學生是否可以繼續下一個觀念的學習。

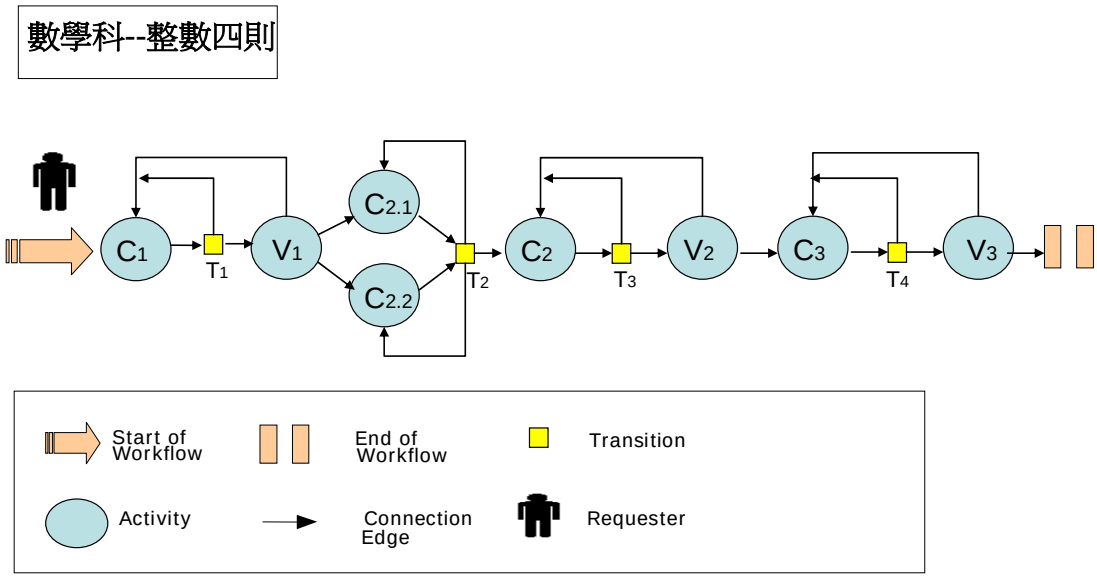

Fig. 2：整數四則學習流程

C1：「整數四則觀念一：由左向右計算」。

C2.1：「整數四則觀念二之一：先乘後加減」。

C2.2：「整數四則觀念二之二：先乘後加減」。

C2：「整數四則觀念二：先乘除後加減」。

C3：「整數四則觀念三：括號要先計算」。

T1：「教務教學系統自動審核學生測驗情況」，當學生完成 C1 的題目測驗後，此分支進行到下一個觀念之前，包含兩個步驟如下：第一步，系統會先根據教師先前所設定的標準來決定學生是否通過測驗，例如：在「C1：由左向右」觀念中一次由題庫抽考五題，而八十分是及格標準的話，也就是學生必須答對題目的數目要多於四題才能算通過然後達到標準；如果只有答對三到一題，或是根本沒有達對，教務教學系統會直接要求學生直接從 C1 重新測驗，直到學生符合標準為止。學生每次的答題結果和測驗的次數都會經由教務教學系統直接記錄在資料庫當中，所以學習測驗的情形教師可以透過資料庫的紀錄清楚學生的程度。

V1：「教師檢視學生測驗情況」，在學生通過系統的預設標準後，學習流程隨即先暫時停止，送到教師端進行審核。教師根據學生學習「由左向右」的情形，判斷是否讓學生繼續學習下個觀念，或是需要再重新測驗，讓學習流程回到 C1。

T2：「教務教學系統自動審核學生測驗情況」，因為教務教學系統的目的是要幫助學生更熟悉課堂上所學習到的知識然後加以測驗，一方面也是讓教師能更瞭解學生的程度加以額外的指導。所以在某些

簡單或是較小的觀念單元，為了減少教師每經一個步驟就必須確認的時間，我們也另外設計一個分支流程只需系統根據標準來判定學生是否夠格進行下一個觀念的測驗學習。如果學生已經成功滿足 C2.1 和 C2.2 的標準，即可自動進行「C2：先乘除後加減」的整合測驗學習。

T3：「教務教學系統審核學生測驗情況」，當學生完成 C2 的題目測驗後，第一步系統會先自動判定學生是否達到標準，如果合格便將把學習流程遞交給教師，不合格就再度從 C2 開始學習測驗。

V2：「教師檢視學生測驗情況」，教師根據學生學習 C2 的情形，判斷是否讓學生通過此階段性測驗，或是需要再回到 C2 重新測驗。

T4：「教師及教務教學系統審核學生測驗情況」，當學生完成 C3 的題目測驗後，第一步系統會先自動判定學生是否達到標準，如果合格將把學習流程遞交給教師，不合格再度從 C3 開始學習測驗。

V3：「教師檢視學生測驗情況」，教師根據學生在測驗 C3 的情形，判斷是否讓學生通過此階段性測驗，或是需要再回到 C3 重新測驗。

四、 教學內容的描述語言

電腦程式語言設計是一門專業的知識和技能，而我們發展教務交互教學的基礎是建立在工作流程管理系統上。工作流程管理系統是「一個定義、產生並管理工作流程運作的系統。此系統藉由流程引擎上運作之軟體，具有解析程式定義並與流程參與者互動之能力，同時更須搭配資訊科技工具與應用程式的使用。」 除了以上的特徵與特性之外，工作流程管理系統[1-5]至少需具備以下幾項基本能力：

1. 作業流程式控制管能力
2. 文件管理能力
3. 資料安全控管能力
4. 資訊搜尋能力
5. 資訊傳遞與分析能力
6. 決策支援能力

並且提供下列幾項系統功能：

1. 完整、簡單與友善的圖形化流程設計功能。
2. 針對流程設計各個步驟的特殊需求，提供人性化表單設計的輔助工具。
3. 可根據工作性質，使用者名稱或是職務關係來決定執行步驟的順序及關聯性。
4. 監督各項工作流程執行狀態的功能。
5. 評估各項工作流程執行效能的功能。

所以我們在致力於教學系統的設計過程中，有幾點重要的訴求：

1. 為了方便教師更簡單使用教學系統，教師完全不用接觸到底層程式設計的部份。
2. 因為學習觀念必須依照學生程度和教學上進度的需求，隨時會有所大量或是些微的調整和變動，所以我們必須讓流程的元件可以經由簡單的步驟就可以修改。
3. 為適應網際網路的變動性，我們希望這套描述語言可以提供異質系統環境的可攜性。

基於以上訴求，我們運用網路運算中之重要技術 XML 來描述直接及交互教學所需流程並定義一個用於直接教學及交互教學之標準語言格式。XML 為目前於網際網路運算或網路運算上作為資料交換及描述語言最為普及的工具，我們希望根據直接教學及交互教學所建置出的教學系統能很容易：（一）移植於各式不同平臺；（二）能易於修改且讓此 XML 標準語言格式能成為一個運用於其他教學模式之流程制定標準語言格式。如此一來如果在想要建至其他種類的教學模式時，便能以最少的時間及花費就可以完成。

設計分為兩個部份，第一個部份為教師定義學習流程，利用 XPDL[6] 技術來描述包含學習單元中的觀念節點(在工作流程中是一個 Activity)還有學習完此觀念後，下一個會銜接到的觀念點，而所傳送的方法會經由教師所訂定的標準來進行分支的動作，分支的動作可以是很基本的循序流程，如下圖 **Fig. 3** 所示：

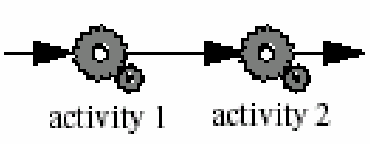

Fig. 3：The sequence structure

用整數四則的流程圖來看，由 C1 到 T1 即是一個循序流程。或是像 T1 到 C2.1 和 C2.2 的 OR-Split，如下圖 **Fig. 4** 所示：

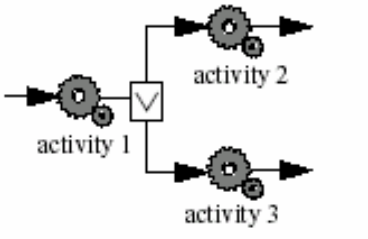

Fig. 4：The OR-Split structure

代表在「T1：教師審核學生測驗情況」核准後，學習流程會自動分支到 C2.1 和 C2.2，學生可以選擇先測驗 C2.1 或是 C2.2，如果測驗通過後，才會進行到 C2，所以在這之間，是透過 AND-Joint 方法來協助，如下圖 **Fig. 5** 所示：

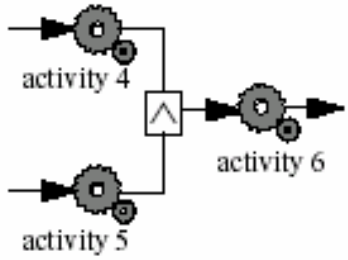

Fig. 5：The AND-Join structure

在教學流程中，是代表學生必須通過 C2.1 和 C2.2 教師所預設的標準，才能進行到 C2 做一個「先乘除後加減」的大型測驗。

在XML的設計上，我們利用XPDL技術來支援。此工作流程標準XML 程式定義語言（XML Process Definition Language）－XPDL 1.0版，是由非營利性組織－工作流程管理聯盟（Workflow Management Coalition，WfMC)於2002年十月公佈。WfMC 為工作流系統制定了五類功能介面。XPDL 1.0規範主要定義介面（程式定

義輸入/ 輸出介面），這個介面包含了一個用於描述程式定義的公共詮釋模型（meta-model），以及程式定義間進行相互轉換的XMLSchema 。我們可以用各種不同的工具來分析、建模、以及描述業務過程。而使用工作流程式定義介面所定義的公共交換格式，可以實現兩個不用系統間工作流程式定義的相互轉化。

參照 **Fig. 1**：整數四則學習流程，我們用 XML 描述如 **Fig. 6**、**Fig. 7**、**Fig. 8**：

```
<?xml version="1.0" encoding="Big5"?>
<xpdl:Package xmlns:xpdl="http://www.wfmc.org/standards/docs/xpdl" Id="math-05-01-03"
Name="整數四則">
 <xpdl:PackageHeader>
  <xpdl:XPDLVersion>0.2</xpdl:XPDLVersion>
  <xpdl:Created>2005-02-04</xpdl:Created>
  <xpdl:Description>九年一貫-數學科-五上-整數四則學習流程</xpdl:Description>
 </xpdl:PackageHeader>
 <xpdl:WorkflowProcesses>
  <xpdl:WorkflowProcess Id="math-05-01-03-01" Name="整數四則">
   <xpdl:ProcessHeader>
    <xpdl:Created>2005-02-04</xpdl:Created>
    <xpdl:Description>九年一貫-數學科-五上-整數四則學習流程</xpdl:Description>
   </xpdl:ProcessHeader>
   <xpdl:Activities>
    <xpdl:Activity Id="由左向右計算" Name="由左向右計算">
     <xpdl:Description>由左向右計算</xpdl:Description>
     <xpdl:Implementation/>
     <xpdl:StartMode>AUTOMATIC</xpdl:StartMode>
     <xpdl:FinishMode>AUTOMATIC</xpdl:FinishMode>
    </xpdl:Activity>
    <xpdl:Activity Id="先算乘再算加減" Name="先算乘再算加減">
     <xpdl:Description>先算乘再算加減</xpdl:Description>
     <xpdl:Implementation>
      <xpdl:SubFlow Id="math-05-01-03-subflow-1" Execution="SYNCHRONOUS"/>
     </xpdl:Implementation>
     <xpdl:StartMode>AUTOMATIC</xpdl:StartMode>
     <xpdl:FinishMode>AUTOMATIC</xpdl:FinishMode>
    </xpdl:Activity>
    <xpdl:Activity Id="先算除再算加減" Name="先算除再算加減">
     <xpdl:Description>先算除再算加減</xpdl:Description>
     <xpdl:Implementation>
      <xpdl:SubFlow Id=" math-05-01-03-subflow-2" Execution="SYNCHRONOUS"/>
     </xpdl:Implementation>
     <xpdl:StartMode>AUTOMATIC</xpdl:StartMode>
     <xpdl:FinishMode>AUTOMATIC</xpdl:FinishMode>
    </xpdl:Activity>
    <xpdl:Activity Id="先乘除後加減" Name="先乘除後加減">
     <xpdl:Description>先乘除後加減</xpdl:Description>
     <xpdl:Implementation>
      <xpdl:Loop Kind="WHILE">
       <xpdl:Condition></xpdl:Condition>
      </xpdl:Loop>
     </xpdl:Implementation>
     <xpdl:StartMode>AUTOMATIC</xpdl:StartMode>
     <xpdl:FinishMode>AUTOMATIC</xpdl:FinishMode>
    </xpdl:Activity>
```

Fig. 6：**XPDL 描述整數四則學習流程(A)**

```
<xpdl:Activity Id="括號要先計算" Name="括號要先計算">
        <xpdl:Description>括號要先計算</xpdl:Description>
        <xpdl:Implementation/>
        <xpdl:StartMode>AUTOMATIC</xpdl:StartMode>
        <xpdl:FinishMode>AUTOMATIC</xpdl:FinishMode>
      </xpdl:Activity>
      <xpdl:Activity Id="系統審核" Name="系統審核">
        <xpdl:Description>系統審核</xpdl:Description>
        <xpdl:Implementation/>
        <xpdl:StartMode>AUTOMATIC</xpdl:StartMode>
        <xpdl:FinishMode>AUTOMATIC</xpdl:FinishMode>
      </xpdl:Activity>
      <xpdl:Activity Id="教師審核學生學習情況" Name="教師審核學生學習情況1">
        <xpdl:Description>教師審核學生學習情況</xpdl:Description>
        <xpdl:Implementation/>
        <xpdl:StartMode>AUTOMATIC</xpdl:StartMode>
        <xpdl:FinishMode>AUTOMATIC</xpdl:FinishMode>
      </xpdl:Activity>
      <xpdl:Activity Id="教師審核學生學習情況2" Name="教師審核學生學習情況2">
        <xpdl:Description>教師審核學生學習情況2</xpdl:Description>
        <xpdl:Implementation/>
        <xpdl:StartMode>AUTOMATIC</xpdl:StartMode>
        <xpdl:FinishMode>AUTOMATIC</xpdl:FinishMode>
      </xpdl:Activity>
      <xpdl:Activity Id="教師審核學生學習情況3" Name="教師審核學生學習情況3">
        <xpdl:Description>教師審核學生學習情況3</xpdl:Description>
        <xpdl:Implementation/>
        <xpdl:StartMode>AUTOMATIC</xpdl:StartMode>
        <xpdl:FinishMode>AUTOMATIC</xpdl:FinishMode>
      </xpdl:Activity>
    </xpdl:Activities>
 <xpdl:Transitions>
      <xpdl:Transition Id="math-transition-05-03-01-01" From="由左向右計算" To="教師審核學生學
習情況1" Name=""/>
      <xpdl:Transition Id=" math-transition-05-03-01-02" From="教師審核學生學習情況1" To="先算
乘再算加減" Name=""/>
      <xpdl:Transition Id=" math-transition-05-03-01-03" From="教師審核學生學習情況1" To="先算
除再算加減" Name=""/>
      <xpdl:Transition Id=" math-transition-05-03-01-04" From="先算乘再算加減" To="系統審核"
Name=""/>
      <xpdl:Transition Id=" math-transition-05-03-01-05" From="先算除再算加減" To="系統審核"
Name=""/>
<xpdl:Transition Id=" math-transition-05-03-01-06" From="先乘除後加減" To="教師審核學生學習情
況2" Name=""/>
      <xpdl:Transition Id=" math-transition-05-03-01-07" From="系統審核" To="先算乘再算加減"
Name=""/>
      <xpdl:Transition Id=" math-transition-05-03-01-08" From="系統審核" To="先算除再算加減"
Name=""/>
<xpdl:Transition Id=" math-transition-05-03-01-09" From="教師審核學生學習情況2" To="括號要先
計算" Name=""/>
      <xpdl:Transition Id=" math-transition-05-03-01-10" From="括號要先計算" To="教師審核學生學
習情況3" Name=""/>
```

Fig. 7：XPDL 描述整數四則學習流程(B)

```
      <xpdl:Transition Id=“ math-transition-05-03-01-11” From=“教師審核學生學習情況3” To=“括號要先計算" Name=""/>
      <xpdl:Transition Id=" math-transition-05-03-01-12" From="教師審核學生學習情況2" To="先乘除後加減" Name=""/>
<xpdl:Transition Id=" math-transition-05-03-01-13" From="教師審核學生學習情況1" To="由左向右計算" Name=""/>
    </xpdl:Transitions>
  </xpdl:WorkflowProcess>
  <xpdl:WorkflowProcess Id=" math-05-01-03-02" Name="先算乘再算加減">
   <xpdl:ProcessHeader>
    <xpdl:Created>2005-02-04</xpdl:Created>
    <xpdl:Description>先算乘再算加減</xpdl:Description>
   </xpdl:ProcessHeader>
  </xpdl:WorkflowProcess>
  <xpdl:WorkflowProcess Id=" math-05-01-03-03" Name="先算除再算加減">
   <xpdl:ProcessHeader>
    <xpdl:Created>2004-02-04</xpdl:Created>
    <xpdl:Description>先算除再算加減</xpdl:Description>
   </xpdl:ProcessHeader>
  </xpdl:WorkflowProcess>
 </xpdl:WorkflowProcesses>
</xpdl:Package>
```

Fig. 8：XPDL 描述整數四則學習流程(C)

<xpdl:PackageHeader>是此 XML 的標頭，其中用<xpdl:Description>說明這個流程的名稱為「九年一貫-數學科-五上-整數四則學習流程」。<xpdl:Activities>代表這個流程的元件有哪些，在這裡每個元件代表是一個數學科的學習觀念，例如：<xpdl:Activity Id=" 先乘除後加減" Name=" 先乘除後加減">代表是整數四則中的第二個觀念「先乘除後加減」。<xpdl:Transition>用來說明流程圖元件間的走向，例如<xpdl:Transition Id="math-transition-05-03-01-01" From=" 由左向右計算" To=" 教師審核學生學習情況 1" Name=" "/>代表一個 ID 名稱為「math-transition-05-03-01-01」，從觀念「由左向右計算」遞送到「教師審核學生學習情況 1」的 transition。除此之外，在這個 XML 流程的 ID 命名方法為「科目-型態-年級-單元-觀念-編號」。

除了學習流程的 XML 定義外，另外一個部份乃是支援題庫所設計出來的架構。我們繼續以「數學科整數四則」單元為例，範例如 **Fig.9**：

```
<?xml version=“1.0” encoding=”Big5” ?>
<範圍 年級=”五上” 科目”數學” 單元=”整數四則”>
 <觀念 觀念名稱=”由左向右”>
  <問題 敘述=”15+25-13=”>
   <答案>27</答案>
   <選項>3,14,27,64</選項>
  </問題>
  <問題 敘述=” 8*7-50=”>
   <答案>6</答案>
   <選項>5,6,7,8</選項>
  </問題>
 </觀念>
 <觀念 觀念名稱=”先算乘再算加減”>
  <問題 敘述=”13*13-13*3=”>
   <答案>130</答案>
   <選項>112,130,120,140</選項>
  </問題>
  <問題 敘述=”21+45*2=”>
   <答案>111</答案>
   <選項>111,123,145,102</選項>
  </問題>
 </觀念>
    <觀念 觀念名稱=”先算除再算加減”>
     <問題 敘述=”15/3 + 15/5=”>
```

```
        <答案>8</答案>
        <選項>5,6,7,8</選項>
    </問題>
    <問題 敘述="57 + 9/3=">
        <答案>60</答案>
        <選項>60,61,62,69</選項>
    </問題>
    </觀念>
  <觀念 觀念名稱="先乘除後加減">
    <問題 敘述="56/4 - 2*3 =">
      <答案>10</答案>
      <選項>8,12,10,20</選項>
    </問題>
    <問題 敘述="12*3 + 42/2 =">
      <答案>57</答案>
      <選項>57,98,43,23</選項>
    </問題>
   </觀念>
   <觀念 觀念名稱="括號先算">
    <問題 敘述="(3+27)/3=">
      <答案>10</答案>
      <選項>4,6,10,11</選項>
    </問題>
    <問題 敘述="7*(54-34)+60=">
      <答案>200</答案>
      <選項>200,211,199,190</選項>
    </問題>
   </觀念>
  </範圍>
```

Fig. 9：XML 題庫設計

此範例說明這個 XML 的測驗範圍是九年一貫教育中的「數學科」，適用於「五年級上學期」的小學生，單元名稱是「整數四則」。其中包含了五個觀念的學習步驟，分別是「由左向右算」、「先算乘再算加減」、「先算除再算加減」、「先乘除後加減」、「括號先計算」。每個觀念中目前只有包含兩個問題，每個問題有其敘述、答案、選項。因為 XML 的 Well-form 特性支援，教師可以簡單並輕鬆的直接修改 XML 檔案，來新增、修改、刪除題目。除此之外，此套 XML 設計的題庫，可以達到跨平臺或是透過任何的工作流程系統或是學習系統來使用。

五、系統架構

因應九年一貫教學內容主要採用「論理組織法(螺旋狀組織法)」，也就是每學習一個新的章節的內容一定會架構在前面學習的基礎上，所以就單一學科來看，學習內容的主體架構其實都是大同小異，而細部內容有所差異。舉例來說，在網路上閱讀理解策略的教學上，我們設計了如下的步驟（流程）:【發問】→【摘要】→【澄清】→【預測】，當遇見教學的情境改變；老師要求改變步驟的順序（例如【摘要】和【發問】兩個步驟要求對調）、或是在原流程中加入新的步驟（在流程的最後加入一個【自我解釋】的步驟）時，傳統的結構化軟體架構必須更改許多部份的程式（例如負責流程的程式、負責每個步驟的程式等）；因此使用傳統的結構化架構使流程的開發及維護變得困難，也不利於多變的網路環境。

針對這樣的學習情況，我們設計了一套物件導向的工作流程系統－JOO-WfMS（Java Object-Oriented Workflow Management System），在 JOO-WfMS 中，我們使用物件導向的設計將工作流程分散成數個種類的介面（Interface），使用者可以繼承這些介面自己實作工作流程的各部份元件（例如流程的元件、步驟的元件、連結資料庫元件等），然後用這些元件組裝出自己的工作流程；當改變的需求提出時，只需將元件拆開再組裝，大大的提高軟體的彈性（flexibility）及再使用性（reusability），同時也提高軟體開發的效率；而這套系統我們研究

報告在 The International Workshop on Groupware CRIWG2003) [16] 、International Journal of Cooperative Information Systems(2005) [15] 還有 The International Workshop on Groupware CRIWG2005) [7] 所發表。

JOO-WfMS 的軟體架構

JOO-WfMS 提供給使用者自製元件的介面、以及客製程式執行的環境；不過即便是繼承 JOO-WfMS 所提供的介面，寫作工作流程的各部份元件然後再將之組裝依然是件浩大的工程，因為一個能運作的工作流程往往牽涉許多種類的元件，例如：與使用者互動介面的元件、連結資料庫的元件、流程的定義以及每個 Activity 的程式，可以想見，要完成一個工作流程可能必須由不同領域的人員共同參與，例如：流程設計人員、workflow 元件設計人員和資料庫方面的程式設計人員；所以我們將 JOO-WfMS 的介面及元件再作進一步的分層，同時也作為開發人員分工的依循；提供概略的角色參考，如 **Fig. 10**：

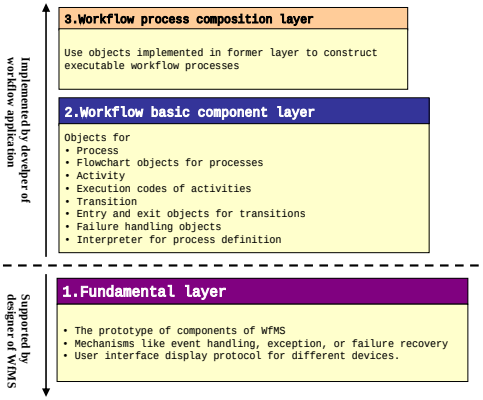

Fig. 10：**Software architecture of WfMS**

上圖說明 JOO-WfMS 的程式堆疊元件，從軟體工程的觀點上來看，將各個元件放到堆疊中至少滿足兩個優點：(1)提升所有元件在系統中的可重複使用性, (2)增加軟體開發的效率。我們將堆疊切割成三層，詳述如下：

1. 基本層：這一層提供系統所需基礎的元件還有機制，經由工作流程系統的開發人員實作還有維護，而工作流程應用程式的發展者是不被允許修改變動這一層的部份。此層包含所有元件的 Java 介面(也就是標準)，如 **Fig. 11**，工作流程引擎的設計也是跟隨這一層的介面定義去實作，除此之外，還有一些特殊機制，如事件處理、錯誤處理、錯誤回復、不同裝置設備上的使用者介面展示的協定也都是實作再這一層上面。
2. 工作流程基本元件層：在這一層中，程式人員根據在基本層中所定義出的 Java 介面實作出所有必要的類別檔，包含流程分支(transitions)、activities、execution codes of activities、錯誤處理物件(failure-handling objects)、process-definition interpreter objects。
3. 工作流程組合層：在這一層，程式人員利用已經實作於第一和第二層中的元件，來建構出自己所需要的工作流程系統。

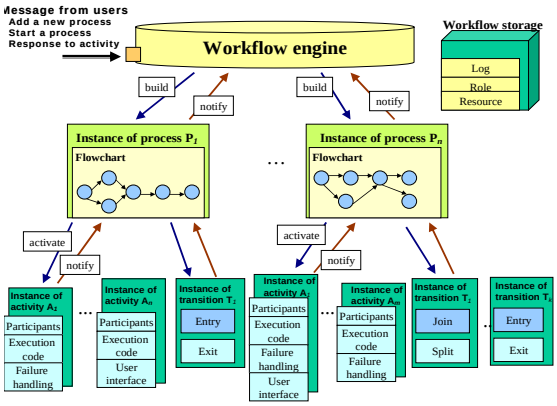

Fig. 11：Operation Model of JOO-WfMS

接下來是工作流程系統的細節：對於 Workflow Engine 的設計，必須滿足下列工作的訴求：

1. 接收來自工作流程參予者們的訊息。
2. 初始化(initiating)行程(process)和依據定義開始(starting)其工作流程。
3. 接受和處理來自行程(process)的事件。
4. 回復和繼續行程在系統發生當機或是行程錯誤產生的時候。
5. 管理人員透過引擎可以去控管和監視行程的運作情形。

Fig. 11 是 JOO-WfMS 的運作模型，在 JOO-WfMS 中由 Workflow Management Engine 來負責 flow 的生成，當 Engine 接收到 Process Definition 時，Engine 就會 Create 一個 Process 的執行實體（往後我們會簡稱 Process 的執行實體為 Process），接下來 flow 會依照 Flow Definition 建立一個 Flowchart 物件，Flowchart 物件中包含流程中所有 Activity 執行所需的資訊，flow 會依照 Flowchart 中的資訊來依序的執行 Activity，所有物件的執行狀態均透過 Event 的形式來回報。透過 Flowchart 及 Transition 的搭配，系統可以處理任何的流程分支狀況。同時，開發人員可以在 Transition 中撰寫程式來決定流程的走向。

Event Model 是 JOO-WfMS 傳遞訊息的重要機制，因為在透過 Internet 來進行交互教學的情境中，有許多 Activity 是必需和使用者溝通、等待使用者回傳某些結果（例如按下某個按鈕或上傳一份檔），系統如果使用迴圈的方式不斷的去偵測使用者是否回傳結果將會嚴重的損耗系統的資源，所以我們採用 Event 的機制以避免系統出現忙碌等待（Busy waiting）的狀況。

為了提升可重複使用性，我們採用合成(Compositional)來設計工作流程引擎。我們將 Workflow_Engine 這個類別函式的實作拆細分為訊息處理、系統管理和監控、流程定義說明和事件處理。以上這些功能都分別實作在不同的類別函式，當工作流程引擎需要使用部分功能時只需要另外去呼叫其類別。所以，我們可以依據工作流程的使用狀況便可很容易的新增或是修改某些功能，去支援工作流程引擎的擴充性。

Workflow Process

在 JOO-WfMS 中，工作流程系統的行程(Process)必須要有下述幾項能力：

1. 創造和開始 activity 和 transition 的實體(instance)。
2. 接收和處理來自 activity 和 transition 的事件。
3. 更新流程圖(Flowchart)內物件的資

訊。

流程的控制在工作流程中有相當重要的地位，包括下列兩種工作：(1)擷取下一個(或多個)即將要執行的activities，(2)決定流程在行程中的分支行為走向，判斷下一步要執行哪一個 activities。然而，物件導向系統缺乏程式性(procedural)的行程控制[10]因為流程控制被分散寫在不同的類別中。所以，相較於程式性的程式來看，物件導向程式在全域的流程控制還有行為的處理度的確比較薄弱[11]。因此有一個問題需要去挑戰的是，要如何在不違反高度物件導向原則的情況下建構出一套可以抽象化維護的工作流程。在 JOO-WfMS，程式設計人員將如何決策流程走向的規則先定義在 Transition 物件當中，而行程的實體便可以根據 Flowchart 知道下一個會走到的 activities，和在 transitions 中流程走向的判斷來獲得所需要的資訊。

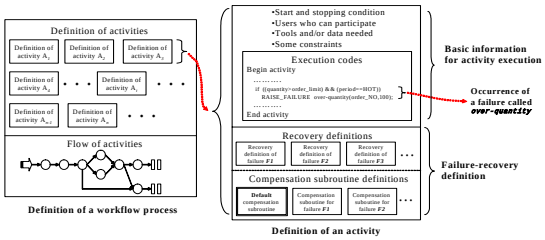

Fig. 12：Skeleton of an activity object

Workflow Activity

Fig. 12 描述一個 activity 物件在 JOO-WfMS 的骨架，其包含執行此 activity 的執行碼還有額外錯誤處理回復的物件的基本資訊。Activity 的執行碼至少擁有開始和終止流程的條件、哪些人可以參予流程、所須的工具和資料、這個流程的限制(例如說：必須在哪個時間內完成)、執行碼等等。額外的錯誤回復功能，結合了回復定義和補償子程式。

使用者介面運作模式

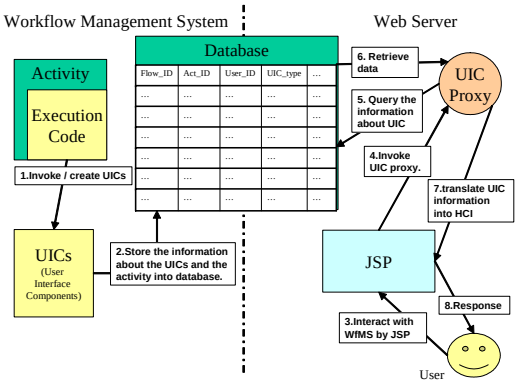

Fig. 13：Operation Model of User Interaction in JOO-WfMS

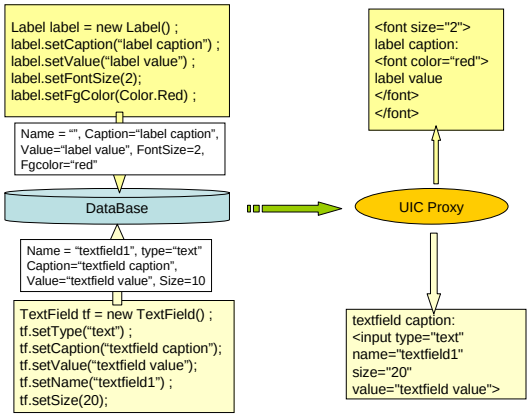

Fig. 14：The translation example of Label and TextField

我們在使用者介面上的研究，其目標對於不同作業系統環境(例如：Windows、Linux)或是不同的裝置設備(例如：PDA、PC)下多樣化的使用者介面，我們可以透過工作流程引擎對

這些不同的裝置進行溝通的動作，以求即使在不同的使用者介面下，仍然可以進行工作流程的檢視檔和控管流程。工作流程應用程式的發展者，呼叫已經設計好內嵌在 activity 上的 ExecutionCode 物件，來建造使用者端的程式。類似在 Window 的平臺上撰寫 Java 程式，我們可直接呼叫 JOO-WfMS 所提供的 UIC 的 Window 元件 (JOO-WfMS 的函式和相關 Window 元件細節收錄在[11])。

JOOWfMS 的使用者介面運作模式如 **Fig. 13** 所示，雖然我們的架構不限制任何型態的使用者端的裝置和環境，但在這裡我們使用 Web 瀏覽器透過 JavaServerPages(JSP)[8]技術和工作流程系統溝通：

步驟 1：程式設計人員先呼叫在 activity 的 execution code 的 UIC 元件(**Fig. 13 錯誤！找不到參照來源。**)。

步驟 2：JOO-WfMS 將在步驟 1 所呼叫的多個 UICs 元件儲存在資料庫中。

步驟 3：使用者透過 Web 瀏覽器傳送 request 給 JSP 伺服器端。

步驟 4：在接收 request 之後，JSP 伺服器可透過 UIC proxy 來重新獲得相關資料和在使用者介面中的 UICs 所轉換後的資料。

步驟 5：UIC proxy 去搜尋資料庫中 UICs 的資料。

步驟 6：資料庫傳回搜尋的結果。

步驟 7：UIC proxy 將 UICs 的資料轉換成特定的格式，以供使用者介面呈現，例如 HTML 標籤。

步驟 8：JSP 伺服器透過 HTTP 傳送結果到使用者端。

Workflow Transition

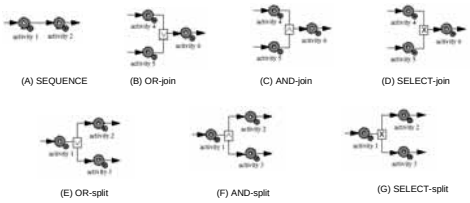

Fig. 15：Basic control flows in the WfMS

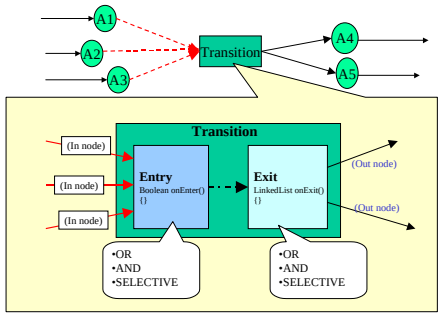

Fig. 16：Structure of the transition component in the JOO-WfMS

大部分的工作流程語言都支援基本的流程順序(sequence)的建構、iteration、splits (AND and OR), and joins (AND and OR)等物件[12-14]。

Fig. 15 表示基本的工作流程系統的控制流程，JOO-WfMS 所設計的元件，允許程式設計人員針對工作內容所須的功能，自行設計流程分支的決策行為。在 JOO-WfMS 我們將其封裝程 Transition 元件，**Fig. 16** 描述 Transition 元件的結構，每個 Tansition 包含了 Entry 和 Exit 兩個物件，程式設計人員利用 onEnter 和 onExit 兩個方法分別來實作 Entry 和 Exit 物件。Entry 和 Exit 物件在設計條件上的不同，

transition 便可以處理任何複雜的 join 和 split 情況(詳見[9])。這樣的控制流程設計，第一個優點是 JOO-WfMS 裡所有相關物件可以容易的被重複利用，包括 Entry、Exit、Transition 物件。其他的優點，它可以支援 real-time 的控制流程，如果當某個 activity 突然執行中止，可以依照系統原先設計的事件處理來做即時應對的動作。

Workflow Persistence

大部分商業化工作流程系統產品使用關聯式資料庫(relational database)管理系統來儲存歷史記錄，以某些技術(例如：the data access object introduced in J2EE)和資料庫連結標準(例如：ODBC、JDBC)輔助對資料庫系統的操作。除了使用資料庫之外，一些工作流程系統會定義其自己的資料結構來儲存執行結果。因此，persistence 這個元件可以允許程式發展者使用不同的資料儲存體來支援資料儲存的功能，工作流程系統引擎或是行程再發生事件處理的時候會呼叫 Persistence interface 的實作來儲存執行結果的資料，其中定義了兩個方法為：saveToStorage 和 retrieveFromStorage。程式發展者可以利用 saveToStorage 這個方法將資料儲存在資料儲存體中，另外再利用 retrieveFromStorage 這個方法從資料儲存體中重新取得資料。

六、實作結果

首先在伺服器端啟動 JSP Container-Tomcat 來支援 Servlet 的服務，另外啟動 JOO-WfMS 的執行，如此遠端電腦就可以連線到我們的伺服器，和 JOO-WfMS 內的教學流程(Teach flow)連線進行線上測驗。

如 **Fig. 17** 所示，教務教學系統的首頁，點選「學習流程開始」便會透過 Servlet 和 JOO-WfMS 的 socket 連線入教務教學系統首頁的登入畫面，如 **Fig. 18**，輸入使用者代號和密碼，和資料庫進行認證的動作後，此時系統會根據登入的身分做不同的動作：

如果身分是學生，教務教學系統便會去讀取儲存在資料庫中，該位學生上次的學習情況，然後接續學習或是測驗，如 **Fig. 19**，「觀念一(由左向右算)的測驗」為例，教務教學系統會自動讀取以 XML 撰寫的題庫，讀寫 XML 再轉換成使用者畫面，學生便可根據題目進行回答。作答完之後，按下「Submit Bottom」後，答題的資料立即會由 Servlet 傳回 JOO-WfMS，並儲存在資料庫中，而且根據預設在系統中 Transition 的判斷是否讓該位學生通過測驗或是重新測驗。

身份如果是教師，便會連結到 **Fig. 20** 的檢視畫面，當某位學生進行測驗後，需要經過教師的許可，都會存放在這個工作清單中，教師點選表格「工作內容」下的「View」便可以連結至 **Fig. 21** 的查閱學生學習情形的畫面。

教師檢視畫面分成三個部份組成，首先是列出被檢視學生的學習情況，其中包含學生資料、答題單元、答對題數、作答次數等等，如此教師便可以清楚透過資訊來瞭解學生的學習情形。接下來是「是」和「否」組成的 RadioItem。點選「是」則代表教師讓該學生通過此階段測驗可以繼續下一個觀念的學習；點選「否」則學生必須重新此階段的測驗。最後一個

部份是教師給學生的建議和回覆，填寫於 TextArea 欄位中，按下「Submit Bottom」，資訊會立即回傳並儲存回 JOO-WfMS 的資料庫中，當學生再次登入後，可以獲取教師給與的建議和回覆。

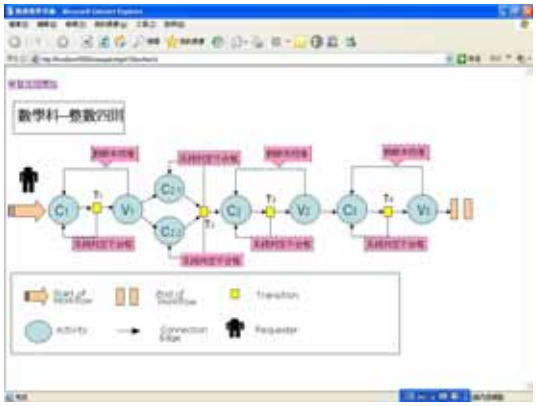

Fig. 17：教務教學系統首頁

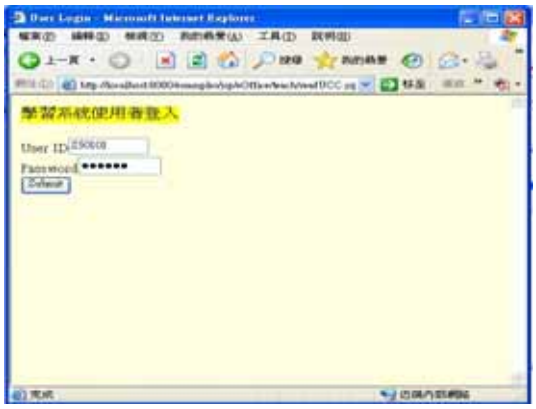

Fig. 18：學習使用者登入畫面

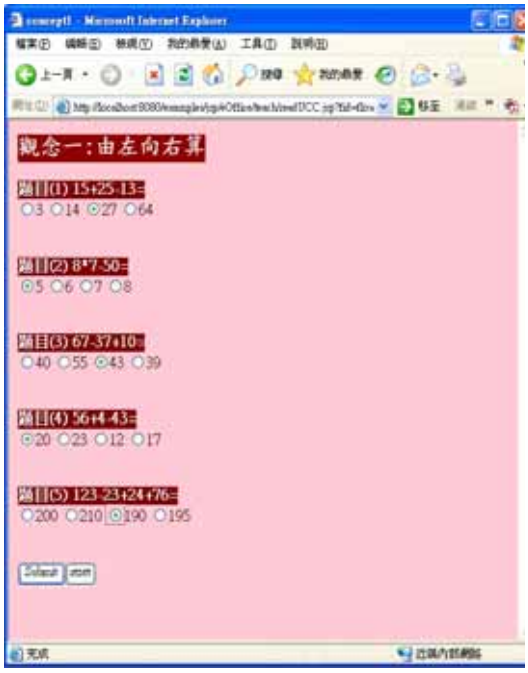

Fig. 19：觀念一(由左向右算)的測驗畫面

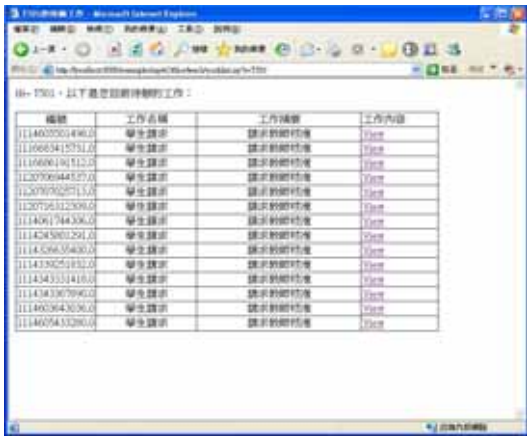

Fig. 20 教師處理學生的工作清單列表畫面

Fig. 21 教師查閱學生學習狀況畫面

七、計畫成果

本計劃主要的目的是研究如何將交互教學的模式，能以網路運算的方法線上運作。由於其型態和工作流程系統的運作有類似之處，所以我們整合工作流程的觀念和應用群組軟體的技術來支援，制定出了一個完整的線上交互教學的模型，並提出了數項要配合的機制。其一是教材的描述檔，利用網際網路運算中的 XML 作為資料交換及描述語言的標準。其二是軟體之系統架構，因為教學內容大多是以「螺旋式」並且科目種類繁多，我們以物件導向語言 Java 技術為基礎設計出可重複利用元件的軟體設計，方便程式設計人員開發新的課程內容。研究報告在 The 8th International Workshop on Groupware (CRIWG2003) [16] 、International Journal of Cooperative Information Systems 2005) [15] 還有 The 10th International Workshop on Groupware CRIWG2005) [7]所發表。另外在實作的成果證明在相關技術的配合下，線上交互教學是可行的。除了學生可以享受線上學習的方便外，教師們亦可依照自我課程需要，很容易的修改或編輯教材內容及流程。

八、參考文獻

1. D. Georgakopoulos, M. Hornick, and A. Shet. Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure. Distributed and Parallel Databases, Vol. 3, No. 2, 1995, Pages 119–153
2. Shi Meilin, Yang Guangxin, Xiang Yong, and Wu Shangguang. Workflow Management Systems: A Survery. International Conference on Communication Technology, 1998.
3. A. Elmagarmid, and W. Du. Workflow Management: State of the Art vs. State of the Market. Proceedings of NATO Advanced Study Institute on Workflow Management Systems, 1997.
4. Workflow Management Coalition. Workflow Reference Model. Workflow Management Coalition Standard, WfMC-TC-1003, 1995.
5. Workflow Management Coalition. Workflow: An Introduction. Workflow Handbook, 2002.
6. WFMC. Workflow Management Coalition Workflow Standard: Workflow Process Definition Interface -- XML Process Definition Language (XPDL) (WFMC-TC-1025). Technical report, Workflow Management

Coalition, Lighthouse Point, Florida, USA, 2002.

7. Gwan-Hwan Hwang, Yung-Chuan Lee, and Sheng-He Chang. Design of an Object-Oriented Workflow Management System with Reusable and Fine-Grained Components. H. Fuks, S. Lukosch, and A.C. Salgado (Eds.): CRIWG 2005, Lecture Notes in Computer Science 3706, pp. 192 – 207, 2005.
8. Sun Microsystem (2002): Inc. JSR-000053 Java™ Servlet 2.3 and JavaServer Pages™ 1.2 Specifications.
9. Yung-Chuan Lee, “Towards the Reusability of Object-Oriented Workflow Management Systems.” Master Thesis, Advisor: Gwan-Hwan Hwang, Dept. Information and Computer Education, National Taiwan Normal University, 2003.
10. Rex Hartson. User-interface management control and communication. IEEE Software, pages 62–70, January 1989.
11. Rex Hartson. User-interface management control and communication. IEEE Software, pages 62–70, January 1989.
12. B. Kiepuszewski, A.H.M. ter Hofstede, W.M.P. van der Aalst, Fundamentals of Control Flow in Workflows. QUT Technical report, FIT-TR-2002-03, Queensland University of Technology, Brisbane, 2002.
13. J. L. Peterson, Petri net theory and the modelling of systems, Prentice Hall, 1981.
14. van der Aalst, W., and van Hee, K., Workflow Management: Models, Methods, and Systems. The MIT Press. 368 pp., 2002. ISBN 0-262-01189-1.
15. Gwan-Hwan Hwang, Yung-Chuan Lee, and Bor-Yih Wu. A Flexible Failure-recovery Model for Workflow Management Systems. International Journal of Cooperative Information Systems, Vol. 14, No. 1 (2005) , pp. 1-24.
16. Gwan-Hwan Hwang, Yung-Chuan Lee, and Bor-Yih Wu. A New Language to Support Flexible Failure Recovery for Workflow Management Systems. Jesus Favela and Dominique Decouchant (Eds.): Groupware: Design, Implementation and Use, Lecture Notes in Computer Science 2806, ISBN 3-540-20117-3, 2003, pp. 135-150.