行政院國家科學委員會專題研究計畫　成果報告

子計畫四:MPEG-4 與 MPEG-7 系統技術之研究(3/3)

計畫類別： 整合型計畫
計畫編號： NSC91-2219-E-009-041-
執行期間： 91 年 08 月 01 日至 92 年 07 月 31 日
執行單位： 國立交通大學電子工程學系

計畫主持人： 杭學鳴

計畫參與人員： 張峰誠 唐健霖 蔡尚軒 楊政偉 范振韋 曾建統

報告類型： 完整報告

處理方式： 本計畫可公開查詢

中　華　民　國 92 年 10 月 23 日

行政院國家科學委員會補助專題研究計畫 ■ 成 果 報 告
□期中進度報告

MPEG-4 多媒體通訊技術之研究—子計畫四：MPEG-4 與 MPEG-7 系統技術之研究(3/3)

A Study on MPEG-4 and MPEG-7 Systems (3/3)

計畫類別：□ 個別型計畫 ■ 整合型計畫
計畫編號： NSC 91-2219-E-009-041
執行期間： 91 年 8 月 1 日至 92 年 7 月 31 日

計畫主持人：杭學鳴
計畫參與人員：張峰誠 唐健霖 蔡尚軒 楊政偉 范振韋 曾建統

成果報告類型(依經費核定清單規定繳交)：□精簡報告 ■完整報告

本成果報告包括以下應繳交之附件：
□赴國外出差或研習心得報告一份
□赴大陸地區出差或研習心得報告一份
□出席國際學術會議心得報告及發表之論文各一份
□國際合作研究計畫國外研究報告書一份

處理方式：除產學合作研究計畫、提升產業技術及人才培育研究計畫、列管計畫及下列情形者外，得立即公開查詢
□涉及專利或其他智慧財產權，□一年□二年後可公開查詢

執行單位：國立交通大學電子工程學系

中 華 民 國 92 年 10 月 15 日

行政院國家科學委員會專題研究計畫成果報告

MPEG-4 與 MPEG-7 系統技術之研究(3/3)

A Study on MPEG-4 and MPEG-7 Systems(3/3)

計畫編號：NSC 91-2219-E-009-041

執行期限：91 年 8 月 1 日至 92 年 7 月 31 日

主持人：杭學鳴　　國立交通大學電子工程學系教授

計畫參與人員：張峰誠 唐健霖 蔡尚軒 楊政偉 范振韋 曾建統

國立交通大學電子研究所

中文摘要

過去數年間，由於多媒體通訊突飛猛進，網際網路迅速普及，以多媒體物件製作、處理、編輯及傳輸為目標的 MPEG-4 標準活動受到大家的關注。我們在此研究計畫的目的為(1)深入研究並模擬 MPEG-4 系統，主要為 MPEG-4 IPMP 延伸系統，以及(2)研讀模擬 MPEG-7 系統與製作 MPEG-7 發展平台。

關鍵詞：多媒體通訊, MPEG-4, IPMP, MPEG-7,多媒體資料庫

英文摘要

Due to tremendous advances in multimedia communication and the aggressive expansion of Internet in the past a few years, the MPEG-4 standards activity, which aims at establishing a comprehensive specifications for multimedia object construction, manipulation, editing and delivery, receives a lot of attentions. Our goals in this project are to (1) investigate and simulate the MPEG-4 Systems and its IPMP extension, and (2) study and simulate the MPEG-7 systems and construct an MPEG-7 platform for testing and research purpose.

Keywords: Multimedia communication, MPEG-4, IPMP, MPEG-7, Multimedia database

目錄 **Table of Contents**

第一部分 MPEG-4 系統的研究

A. 背景與目的

每種多媒體，例如書本、音樂、電影、電視節目等等，被數位化前或者後，都是智慧財產的一種。現今網際網路的蓬勃發展，涉及數位化多媒體財產的電子商務日漸普遍。因此，能保護智慧財產和使得智慧財產可以透過更方便的形式作商務交易的技術 – 智慧財產的數位化管理和保護(Intellectual Property Management & Protection，簡稱 IPMP)，將會成為電子商務系統之中不可或缺的一環。

由 1997 年開始，MPEG 組織開始嘗試為 IPMP 訂立標準的公用介面和相關協定，以解決數位化媒體在網路上交易所衍生的問題, 目標是把 IPMP 整合到現有的 MPEG 標準之中。其中利用密碼方法或數位浮水印的技術以有效的提高非法剽竊的困難度。目前，類似但簡化之技術已被應用在數位電視系統中。截至本論文撰寫時，MPEG-4 和 MPEG-2 標準中的 IPMP 訂定已經快將完成, 而 MPEG 組織也開始投入更多的資源訂定 MPEG-21 IPMP 標準。

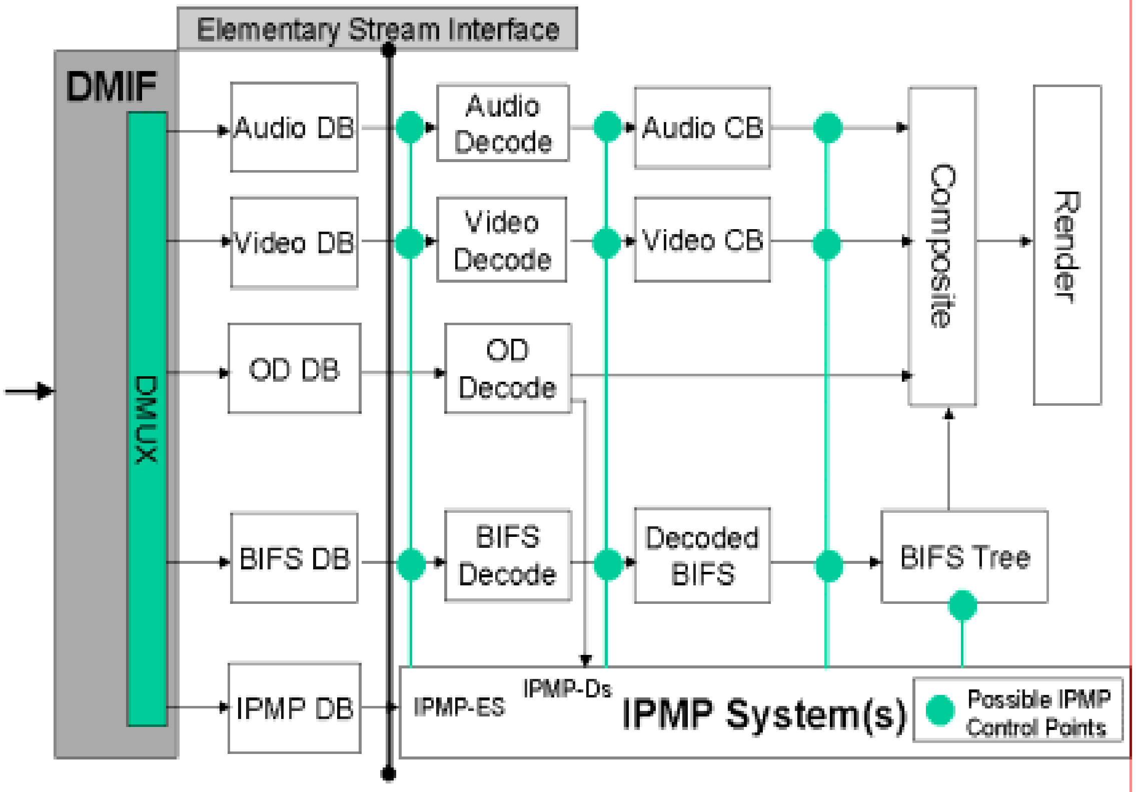
MPEG-4 IPMP 架構已經歷重大改進，由最初只能作有限範圍保護的 IPMP hook 介面，進展為更具彈性與痛通操作性的 IPMP Extension (IPMPX)介面。MPEG-21 IPMP 尚在起草階段，許多介面方式仍有待討論，不過可以預期將來與 MPEG-21 主架構整合，可以提供比 MPEG-4 IPMPX 更為完善的數位資訊保護。

B. 研究步驟 – IPMP 標準與架構

我們的研究承續之前對於 MPEG-4 系統[1]的了解，本期專注於 MPEG-4 標準 (ISO/IEC 14496-1) 中所提供的 IPMP 架構，並輔以部分 MPEG-21 IPMP，以期對 MPEG IPMP 有更前瞻的認識。

MPEG-4 Systems ver.1 中所定義的 IPMP 架構提供了一個標準化的介面讓 MPEG-4 players 可以使用不同的 IPMP System[2]，MPEG-4/AMD3[4]提出後，為了區別兩個版本中差異頗大的 IPMP 子系統, 前者稱 IPMP Hook, 而後者稱為 IPMP Extension (IPMPX)。IPMP Hook 運作主由 IPMP ES (Elementary Stream) 以及 IPMP 描述子 (descriptor) 所構成。IPMP ES 傳遞時間上變化較快的週期性的資訊給一個或是多個 IPMP 系統。IPMP 描述子由 object descriptor stream 所攜帶並傳送出去。圖一是 MPEG-4 中的 IPMP Hook 架構。

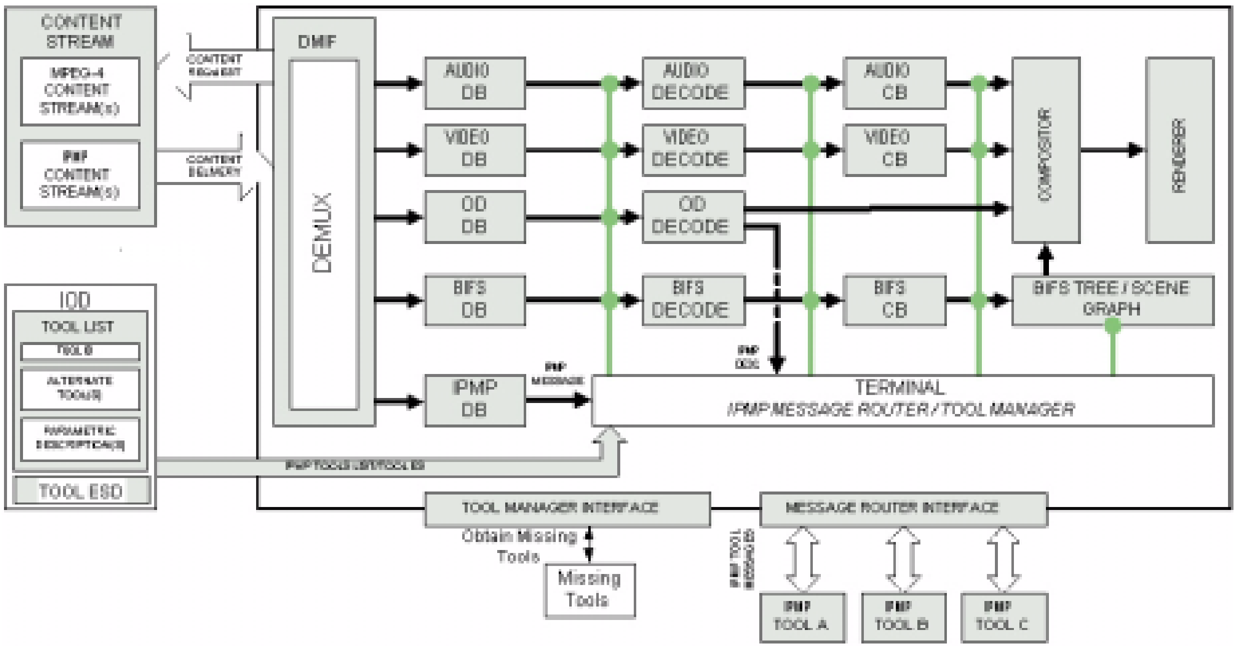
IPMP Hook 架構的最大不足之處在於工具間的溝通方式並沒有正式定義, 這使得各工具的實作者無法利用他人的實作成果。有鑑於此，IPMPX 架構觀念改為虛擬終端機 (Virtual Terminal)，與既有的 MPEG-4 系統以 Message 互相溝通。IPMP 虛擬終端主要由兩大概念合成，一為 Message Router (MR)，另一為 Tool Manager (TM)。Message Router 負責將所有的 IPMP Message 傳送至對應的 IPMP Tool，而各 Tool 則根據 Message 內容負責串流的處理，例如解碼或是權限控制。Tool Manager 負責 Tool 的建立、消滅、與關聯等功能，當需要時，可由 MPEG-4 系統或 Message Router 發出管理需求。



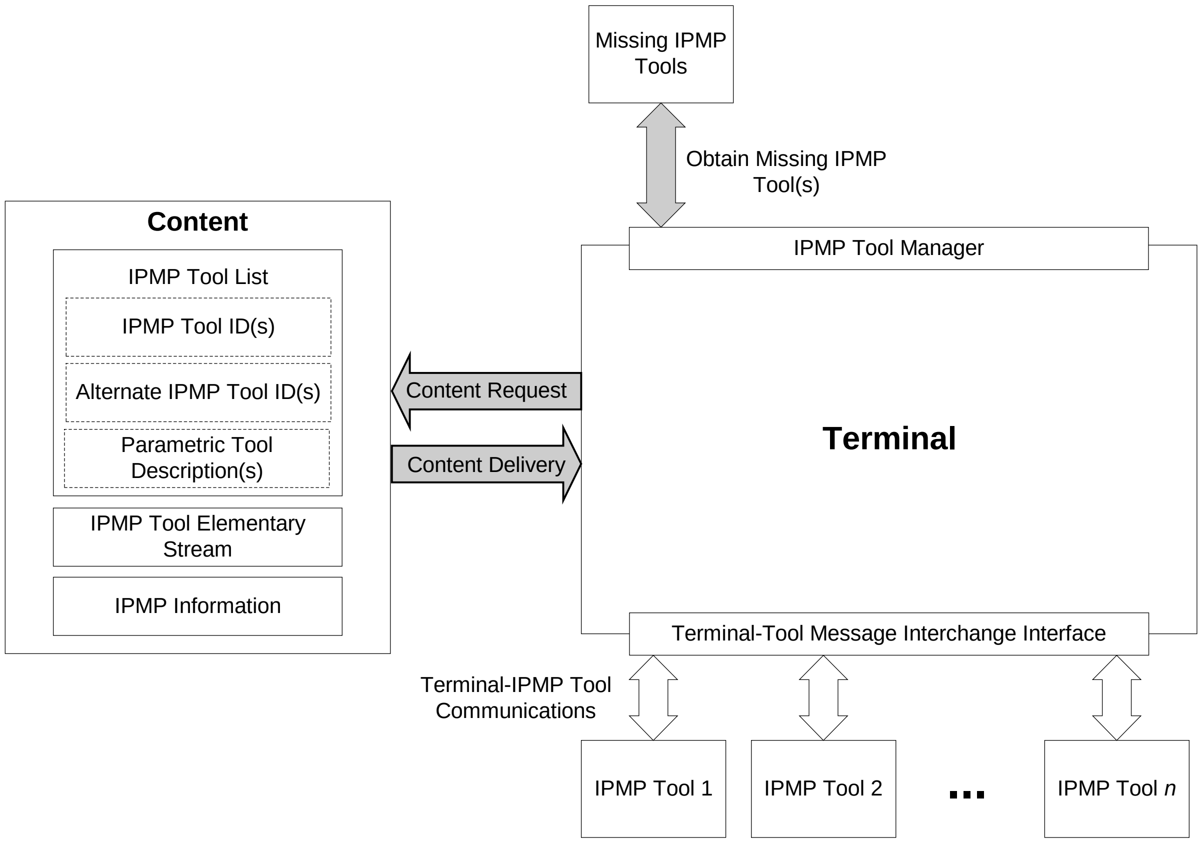
圖一 MPEG-4 的 IPMP Hook 架構[3]

在 IPMPX 下，依據 IPMP Tool Identifier，系統可以使用的 Tool 可分為下列幾類：

- Unique Implementation -- 代表唯一的獨立實作
- Parametric Description
  - 根據給定參數，符合特定條件的實作
  - 實作所需的參數可透過 Parametric Configuration Information 傳入
- List of Alternative Tools
  - List 中指明的所有 Tool 皆視為等效實作
  - 可搭配 Parametric Description 描述所有符合特定條件的等效實作



圖二 IPMPX 與 MPEG-4 系統[4]


Missing IPMP Tools
Obtain Missing IPMP Tool(s)
Content
IPMP Tool List
IPMP Tool ID(s)
Alternate IPMP Tool ID(s)
Parametric Tool Description(s)
IPMP Tool Elementary Stream
IPMP Information
Content Request
Content Delivery
IPMP Tool Manager
Terminal
Terminal-Tool Message Interchange Interface
Terminal-IPMP Tool Communications
IPMP Tool 1
IPMP Tool 2
...
IPMP Tool n

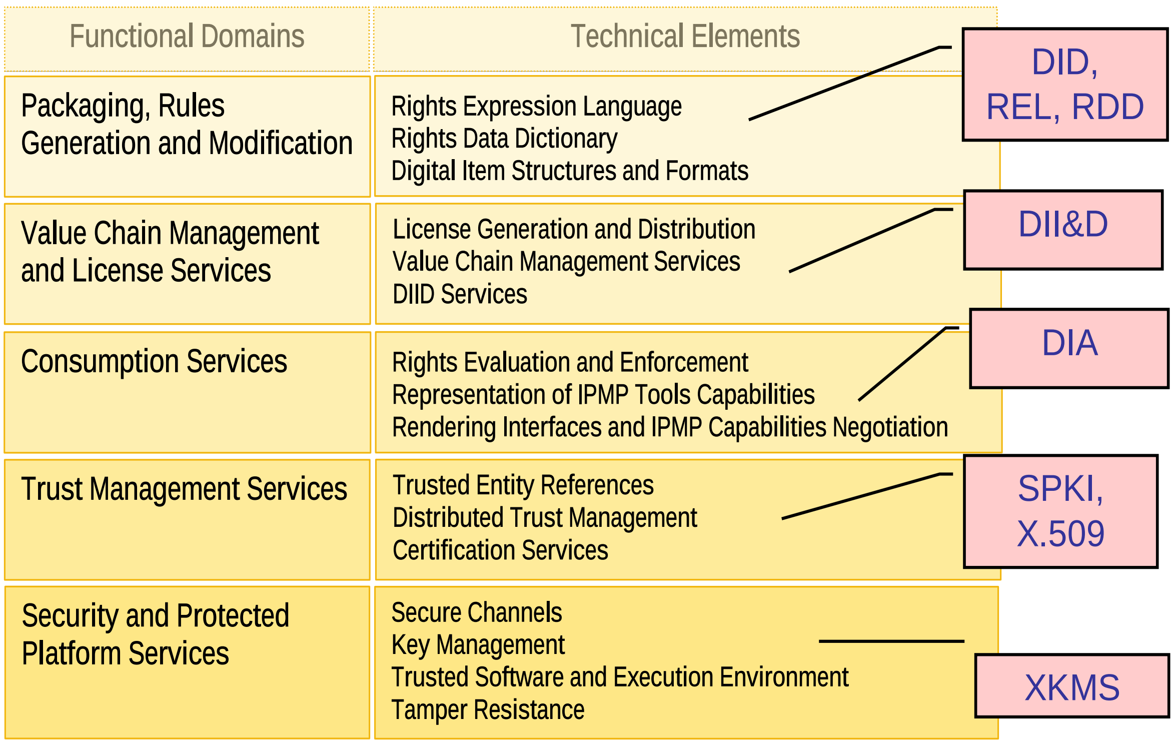
圖三 IPMP 使用流程示意圖[4]

IPMPX 的起始資訊也是透過 MPEG-4 系統串流傳遞至終端。接收端對於這部分的處理流程如下：

1. 使用者要求特定的數位內容
   - IPMP 需求先於內容需求
   - 在傳送內容前先完成 access control
2. 取得 IPMP Tools descriptoion
   - 從內容中取得所需的 Tool List
   - 根據 Tool List 搜尋可建立的 Tool
3. 取得 Tool
   - 找尋並下載無法在終端機找到的 Tool，這項功能不在標準定義內
4. 建立 Tool instance
   - 實際建立 Tool
   - 從內容中取得 IPMP information
5. 初始化 Tool 與更新 Tool
   - 給予各 Tool 起始 IPMP information
   - 終端開始消化允許處理的內容
   - 開始處理內容後，終端機可根據收到的 IPMP 命令，重新出始化或更新 IPMP Tool

MPEG-21[4] IPMP[6]需求文件尚在草稿階段，不過可從其中窺見 MPEG 委員會對於未來 IPMP 系統的構想。在 MPEG-21 的數位內容描述與散佈架構下，MPEG-21 IPMP 涵蓋的範圍非常廣，從某個特定數位內容的生命期來看，IPMP 作用的範圍包含所有觸及權限管理的軟硬體及技術。從技術範圍來看，IPMP 同時具有系統性與切割性。就系統性而言，IPMP 必須整合散佈在整個商業價值鏈的各個部分，包含各點之間信賴關係的建立與管理，商業模式的操作方式與架構等等。另一方面，切割性來自於 IPMP 與許多技術領域相關，例如應用整合、元件化軟體、行動程式碼、作業系統、分散式服務、數位內容管理、安全服務等等。

圖四為 IPMP 的抽象系統模型，這個模型從兩個維度來說明。垂直方向代表 IPMP 技術的層次，越上面的越高層。水平方向代表各功能與相對應的技術，可以看出 IPMP 的切割性。最右側的說明方塊標示在 MPEG-21 中對應的標準工具。以下將對圖中五大功能簡略描述：



Functional Domains	Technical Elements
Packaging, Rules Generation and Modification	Rights Expression Language Rights Data Dictionary Digital Item Structures and Formats
Value Chain Management and License Services	License Generation and Distribution Value Chain Management Services DIID Services
Consumption Services	Rights Evaluation and Enforcement Representation of IPMP Tools Capabilities Rendering Interfaces and IPMP Capabilities Negotiation
Trust Management Services	Trusted Entity References Distributed Trust Management Certification Services
Security and Protected Platform Services	Secure Channels Key Management Trusted Software and Execution Environment Tamper Resistance

圖四 Abstract IPMP System Model[6]

- Packaging, Rules Generation and Modification 的相關技術：
  - ■ 數位內容包裝技術，通常包含內容與 meta-data、智慧財產權資訊的整合包裝，最後產出為受到保護的數位內容。
  - ■ 著作權與相關資料表示法的標準規格。
  - ■ 數位內容的產生與修改。
- Value Chian Management and License Services 的相關技術：
  - ■ 在商業模式下數位內容的散佈與使用。
  - ■ 包裝好的內容除了可以直接實體散佈出去外，也可以採用類似 URL 的方式只將參考連結散佈到使用者手上。
- Consumption Services
  - ■ 內容的使用者，包含一般使用者及價值鏈中的內容處理者，皆是透過 Consuption Service 使用數位內容及服務。
- Trust Management Services
  - ■ 可用於管理 IPMP 各元件間的信賴關係，這些元件包含執行程序與結構。
  - ■ 這部分的功能可包含下列幾種形式：
    - ◆ 信賴模式，例如點對點模式、web 模式、階層模式
    - ◆ 價值鏈中的軟硬體認證

◆ 價值鏈中的軟硬體註冊

◆ 安全平台或服務的個人化設定

◆ 安全機制的生命期，包含更新與廢止等管理

◆ 使用者憑證與密碼管理

◆ 可信賴的對時服務

● Security and Protected Platform Services

■ 要與受到保護的系統或服務溝通，或是使用受保護的內容，本身必須是安全機制中的一環，也必須能夠向對方證明自己的安全能力。

■ 這些安全機制可能為：

◆ 對抗攻擊的能力

◆ 作業系統執行環境的安全管理

◆ 軟體個人化

◆ 對使用者存取遠端或近端資源的認證

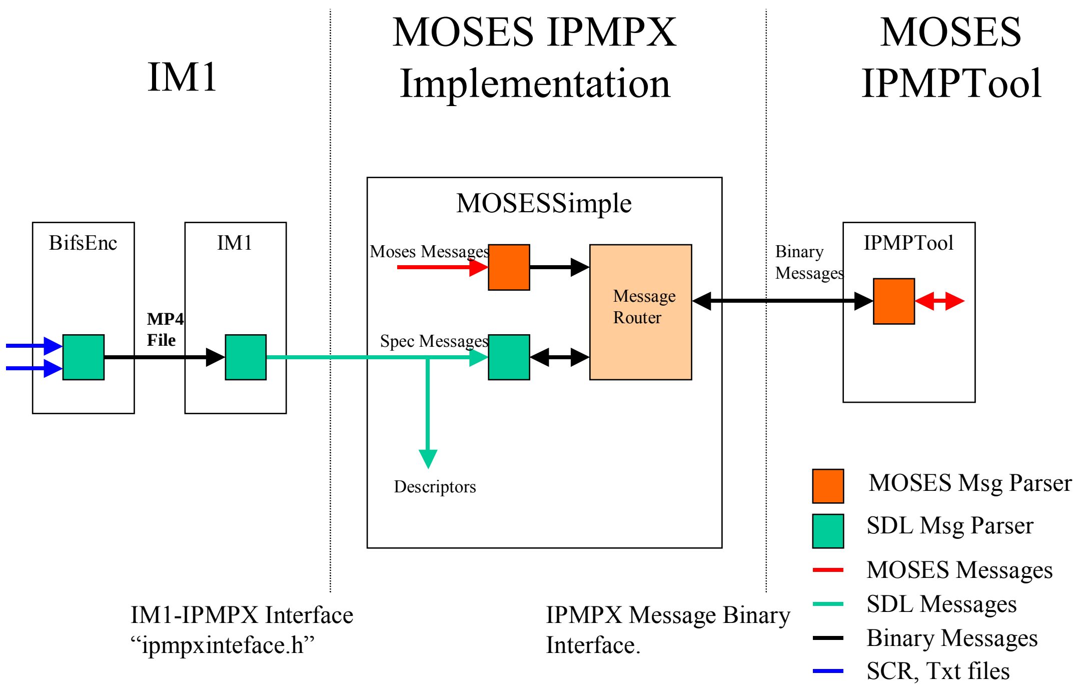
由以上說明，我們可以了解 MPEG IPMP 的目標是建立一個從數位內容產生到消失，整個過程都加以保護的散佈與使用機制。現階段接近完成的為 MPEG-4 IPMPX[9] 與 MPEG-2 IPMP[9]，這二者採用 Virtual Terminal 概念，在現有系統之外，加上 IPMP 介面。為了使 IPMP 工具之間溝通順暢，採用 Message 方式交換 IPMP 資訊及命令。與 MPEG-4 IPMPX 與 MPEG-21 IPMP 最大不同點在於，前者作用範圍在 Terminal 端，後者則散佈在整個傳遞架構各環節中。

C. 模擬與實驗

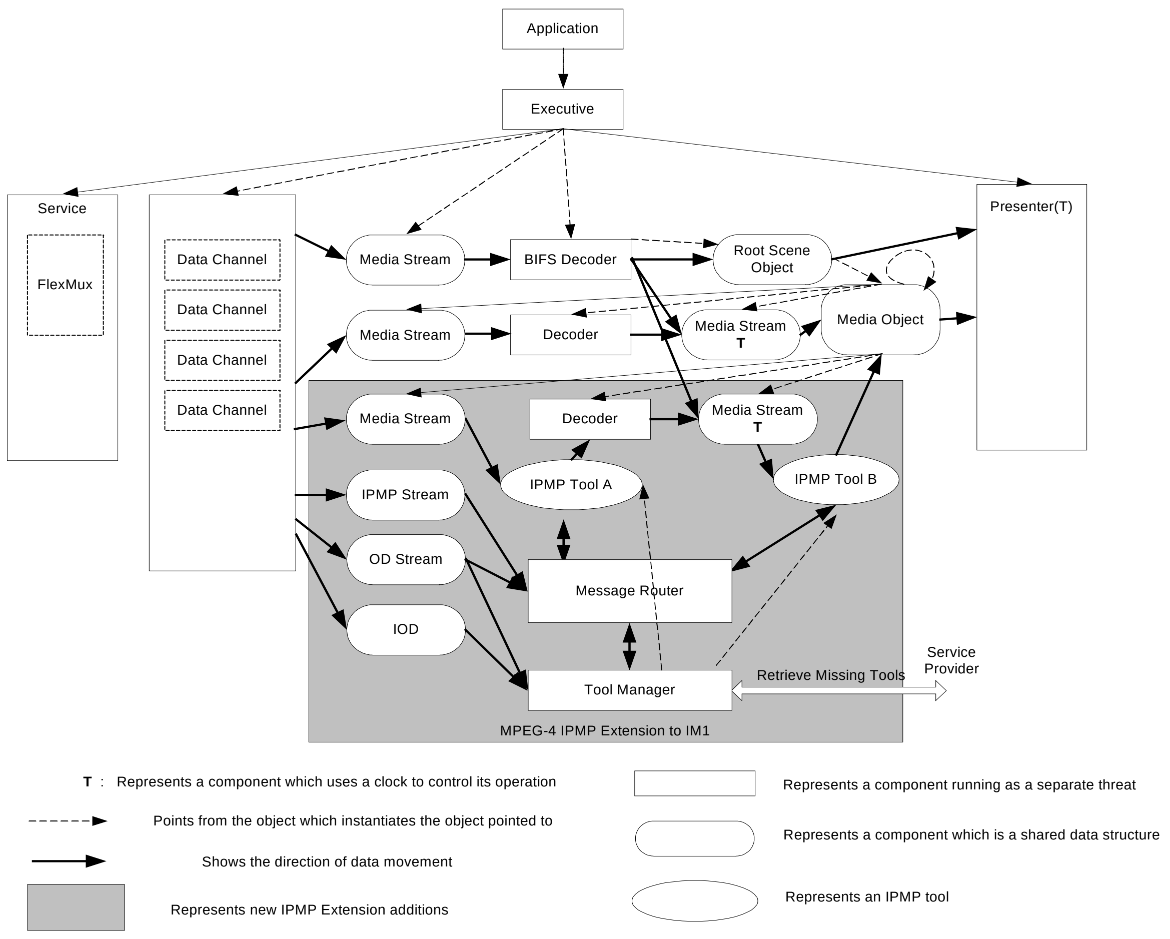
本模擬重點為 IPMPX[8][9]，相關的參考軟體不多，一為 Craig A. Schultz 所作，另一個為 MOSES，兩者皆利用 IM1 為 MPEG-4 player。IM1[7]，或是稱做 AHG (Ad-hoc group) on Systems Reference Software Implementation，是 MPEG 委員會中負責從事開發以及整合 MPEG-4 System 軟體的團體。IM1 軟體包括了所有 MPEG-4 System 中標準化的部分。MOSES 至今未有正式公開版本，但根據架構圖(圖五)顯示，設計上像是 IM1 的加強版，因此與 Craig 的 IPMPX 實作架構(圖六)有些許不同。

我們分析了 IM1 Core 中有關 IPMPX 的部分，了解 IPMPX 系統的操作模式，並據此設計三個程式，以循序漸進的方式，展示 IPMPX 的運作。

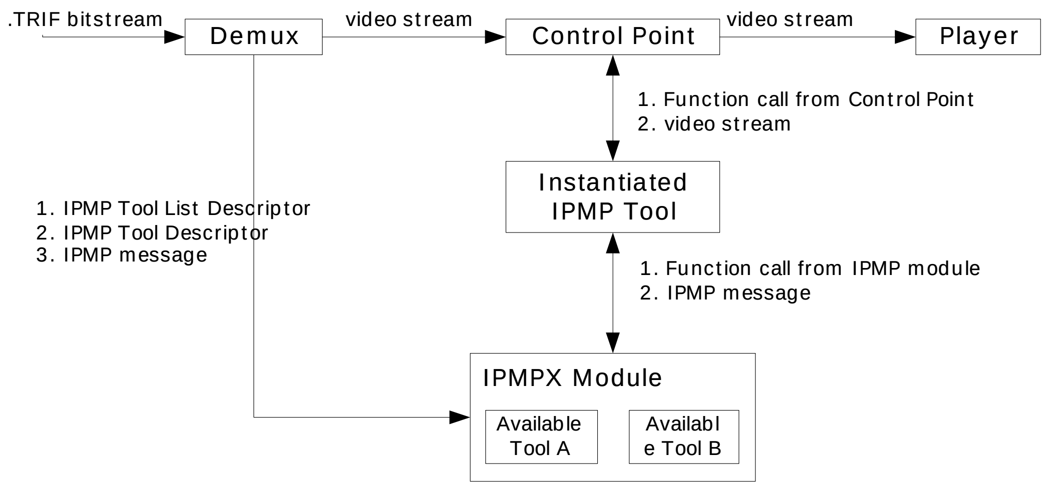
圖七為本展示程式的資料流程圖，包含 IPMP 資訊的 MPEG-4 串流到達後，先經過 Demultiplex 影像資料流往 player 的 control point。而 IPMP 資料則流向 IPMPX 模組，啟動對應的 IPMP Tool 並開始對流經 control point 的影像資料作處理。



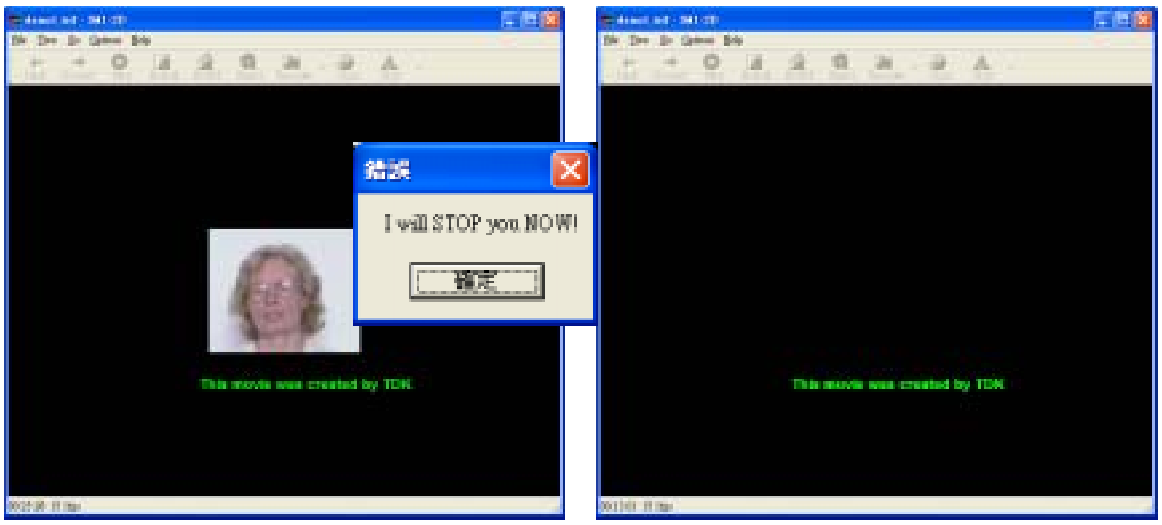
圖五 MOSES IPMPX



圖六 IM1 IPMPX (Craig's version)



圖七 展示程式 Dataflow



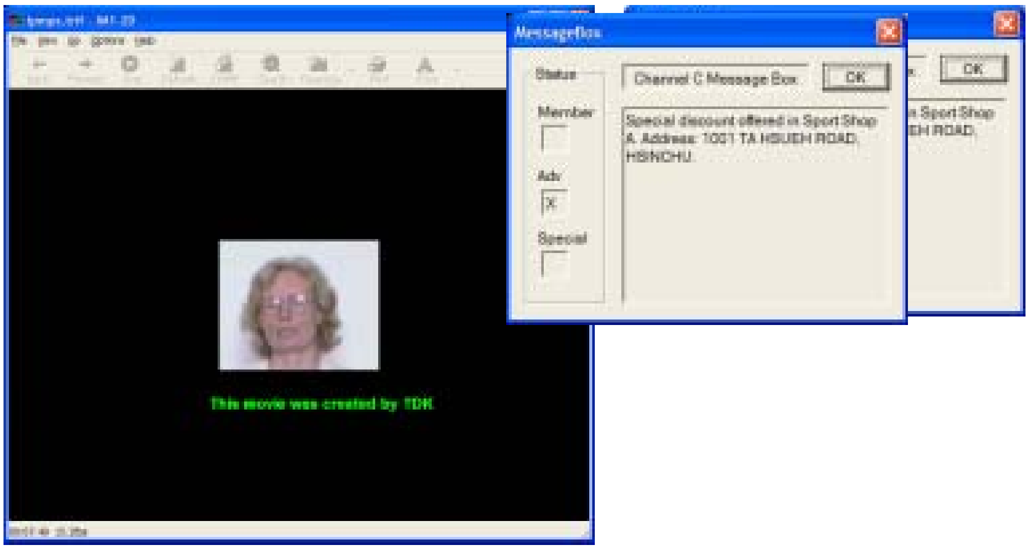
圖八 程式展示

第一個程式展示 IPMPX 模組啟動，當收到 IPMP 串流時，啟動 AES Tool。當收到後續的 IPMP 資料時，顯示資料內容。第二個程式主要用來展示 IPMP 資訊可以隨著時間變動，因此我們在特定時間送出 IPMP update 命令，使 Tool 失效(

圖八)。第三個程式展示權限控管的概念(

圖九)，在傳送出來的串流中，事先依據不同權限對不同部分加密，接收的使用者必須插入 ID 卡認證，並取得符合其身份的權限，就可對授權部分解密。這種應用可能出現在購物頻道，一般用戶只能看廣告，而物品的優惠訊息則需更高權限的使用者才能

看到。



Channel C Message Box

Special discount offered in Sport Shop A. Address: 1001 TA HSUEH ROAD, HSINCHU.

圖九程式展示

D. 結論

本研究的主要目的是探討 MPEG IPMP，除了以 MPEG-4 IPMPX 為主要研究對象外，我們也對 MPEG-21 IPMP 的設計進行了解。以現今的多媒體系統而言，要達到 MPEG-21 那種理想並不容易，因此我們以 MPEG-4 IPMPX 為主，追蹤 IM1 程式架構，並了解 IPMPX 運作原理，把標準文件中抽象的 Message Router 與 Tool Manager 對應到實際的系統中。

IPMPX 以 Message 為交換 IPMP 資訊的方法，使得各 Tool 間的互通性大增，也讓 IPMPX 比 IPMP Hook 介面更為可行、更具彈性。了解程式的過程中，我們發現 IPMPX 與 IM1 間的相關性仍然太高，如果想把 IPMPX 抽離 IM1，成為像概念圖上的 MPEG-4 Terminal 與 IPMP Virtual Terminal，仍有許多技術問題需要克服。或許未來 MOSES 正式版本公佈後，可以作些比較。

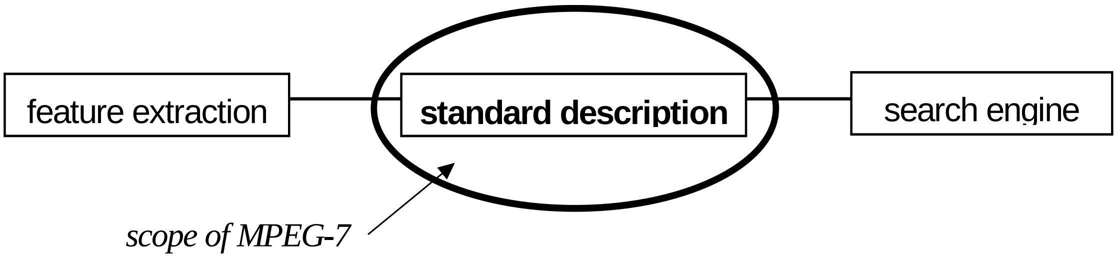
研究的最後，我們以三個程式展示 IPMP 模組的啟動、Message 的傳遞、IPMP 資訊的動態更新，以及透過 IPMP 進行權限控管。根據這幾個程式，我們初步驗證 IPMPX 與 MPEG-4 player 搭配，的確可以執行一些電子商務所需的功能。

第二部分 MPEG-7 系統的研究

A. 背景與目的

傳統的印象裡，影音資訊經由播放系統送出後，其接收者為人類。現在許多影音應用逐漸增加，其接收者不一定是人，而是特定的運算系統。舉例來說，機器人的視覺及聽覺辨識，或是媒體型態自動轉換（語音轉文字或圖片轉語音等）。也有些應用於搜尋特定條件的媒體，例如監視錄影機一邊錄下影像，一邊比對是否有危險事件發生，再據此決定是否要發出警報。又例如根據電視影片中的特定資訊啟動或關閉錄影機，只錄下自己要的片段。

在未來將會有更多的應用領域需要處理影音資訊，為了增加處理效能，將影音資訊以特定的格式描述，方便軟硬體處理，是必要的步驟。MPEG 系列的標準已訂出 MPEG-1 及 MPEG-2 這類 frame-base 的影音資料格式，或是 MPEG-4 這種 object-base 的影音資料格式。這些格式只表示出資料的值，卻無法表示資料在概念上的關係或意義，如果我們想從中找尋具有特定意義的影音片段，傳統上多靠人工搜尋。MPEG-7 就是基於上述的需求而生，如圖十所示，目的為定義一套多媒體的描述格式。其應用的範圍並無特定限制，原則上只要是與影音資訊搜尋比對相關的應用皆可使用。



圖十 MPEG-7 Scope

B. 研究步驟 -- MPEG-7 標準與工具

研究 MPEG-7 的過程中，我們先對 MPEG-7 標準的內容進行研讀，除了了解 MPEG-7 可以有哪些應用方式外，還對其提供的工具加以追蹤，最後我們提出一個軟體測試平台，並利用平台建立原型程式驗證架構的設計是否有足夠彈性，希望能利用這個架構輔助往後的 MPEG-7 研究。

為了能表達影音資料的意義，我們會產生另一種資料，這種資料並非原本影音資訊的一部份，可視為附加上去的，有些時候，我們稱這種描述資料的資料為 meta-data。MPEG-7 正是規範 meta-data 格式的標準，為了讓 MPEG-7 資料能在最多的系統之間交換及處理，MPEG-7 採用 XML 相關技術定義其提供的工具。也就是說，符合 MPEG-7 的 meta-data 格式都是 XML 的 Application。參與 MPEG-7 的專家們除了規範各式各樣的基本 descriptor（簡稱 D）及 description scheme（簡稱 DS），同時也提供一套參考軟體供大家驗證其可行性。Descriptor 用來描述媒體單一方面的特性，例如色彩分佈、紋理、物體輪廓、物體移動軌跡等等。Description scheme 可視為用來組合基礎特徵 (D or DS) 的樹狀資料結構。以下簡述 MPEG-7 提供的各項工具：

➢ **Systems**[10] 定義 MPEG-7 資料的傳送及使用方式，meta-data 除了可以用純文字 XML

格式傳送外，也可以採用 MPEG-7 內定義的 binary 格式(BiM)傳送。

- ➢ **Description Definition Language**[10] 用來定義新的 D 或 DS 的語言，是根據 XML Schema 標準而來，主要目的是讓使用者能根據不同需求，定義新的資料型態及結構。
- ➢ **Visual**[10] 與視訊相關的 D 及 DS。這部分包含顏色、紋路、形狀、以及移動等相關 meta-data。
- ➢ **Audio**[10] 與聲音相關的 D 及 DS。這部分包含音效、樂音、以及語音等相關 meta-data。
- ➢ **Generic Entities and Multimedia Description Schemes**[10] 基本的 D 與 DS。這部分除了包括通用於視訊與音訊的 meta-data 外，例如時間與解析度，還定義其他類型的 meta-data，例如基本資料類型、媒體使用資訊與管理、媒體瀏覽、使用者互動等等。
- ➢ **Reference Software**[10] 簡稱 XM，為 eXperimentation Model 的縮寫。主要提供一套 C++軟體，展示各 D 及 DS 的可行性與搜尋準確度。特徵抽取與比對演算法不在 MPEG-7 標準定義範圍內，但 XM 中所採用的演算法，雖然不是最好的，卻是研究 MPEG-7 的很好入門參考。
- ➢ **Conformance**[10] 規範如何測試是否為 MPEG-7 相容系統的原則。

MPEG-7 提供許多現成的 D 與 DS，並不代表這些 meta-data 有一定的使用方法，事實上，MPEG-7 的 formative 部份只提及格式及其代表的意義，至於參考的使用方法則列在 informative 部份。

C. MPEG-7 軟體平台

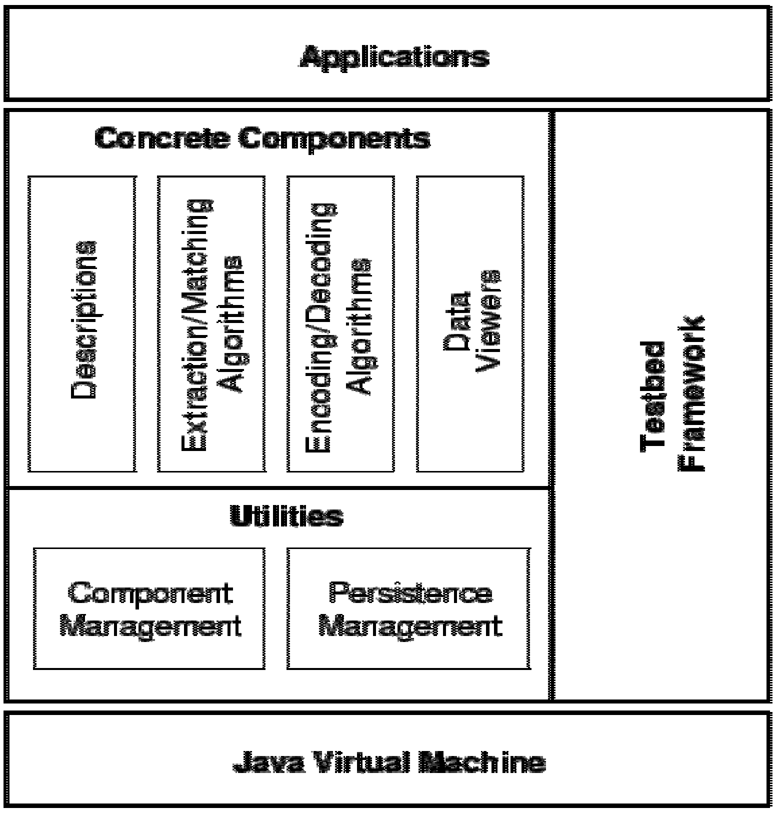
儘管 MPEG-7 XM 已經提供 C++軟體供我們研究，但其目的是驗證各個 D 與 DS 是否可行，以及參考演算法是否正確無誤。對於深入研究的需求，例如以圖表方式顯示找出的 D 或 DS 內容，或是資料庫運作的設計，XM 的功能就不符所需。我們根據 MPEG-7 的通用使用狀況，配合研究上可能的需求（資料庫、網路、圖形化展現等等），規劃一個軟體平台。如圖十一所示，這個平台以 Java 語言為基礎，其上建構一個我們設計的 framework，這個 framework 主要由三類抽象元件構成：

- ● Data：觀察或處理的主體，可能是媒體資料或 meta-data。
- ● DataAlg：與 Data 關聯的演算法，就媒體資料而言，為編碼與解碼。若作用主體為 meta-data，則除了編解碼之外，還包括抽取與比對演算法。
- ● Viewer：用來觀察 Data 的圖形化介面，或是轉換 Data 型式的演算法。

在這個測試平台中，以 Data 為中心，允許有零至多個 DataAlg 或 Viewer 與其相關，因此設計 Component Management Unit 用來維護元件間的關聯性。有了 framework 與 Component Management Unit 後，我們就可依此架構實作不同的媒體、D、DS、以及相對應的顯示元件。目前在此平台中，我們完成了 Scalable Color、Color Layout、Dominant Color、Edge Histogram 等 meta-data 相關元件，也完成 image 相關元件。

實際應用上，如何搭配資料庫搜尋以達到最佳效能也是研究重點之一，因此我們設計資料庫單元，允許研究者設計專屬特定型態資料儲存媒介的存取方式。例如儲存媒體為關聯資料庫時，可以利用 JDBC 完成。我們特別實作一個檔案系統的 Persistence

Manager 來展現這個設計不只可以與實際的資料庫相連，其彈性足以應付其他種類的儲存媒體。此外，為了讓程式架構維持元件化的形式，我們將資料庫單元的介面設計成以物件為處理單位，其功能簡述如下：

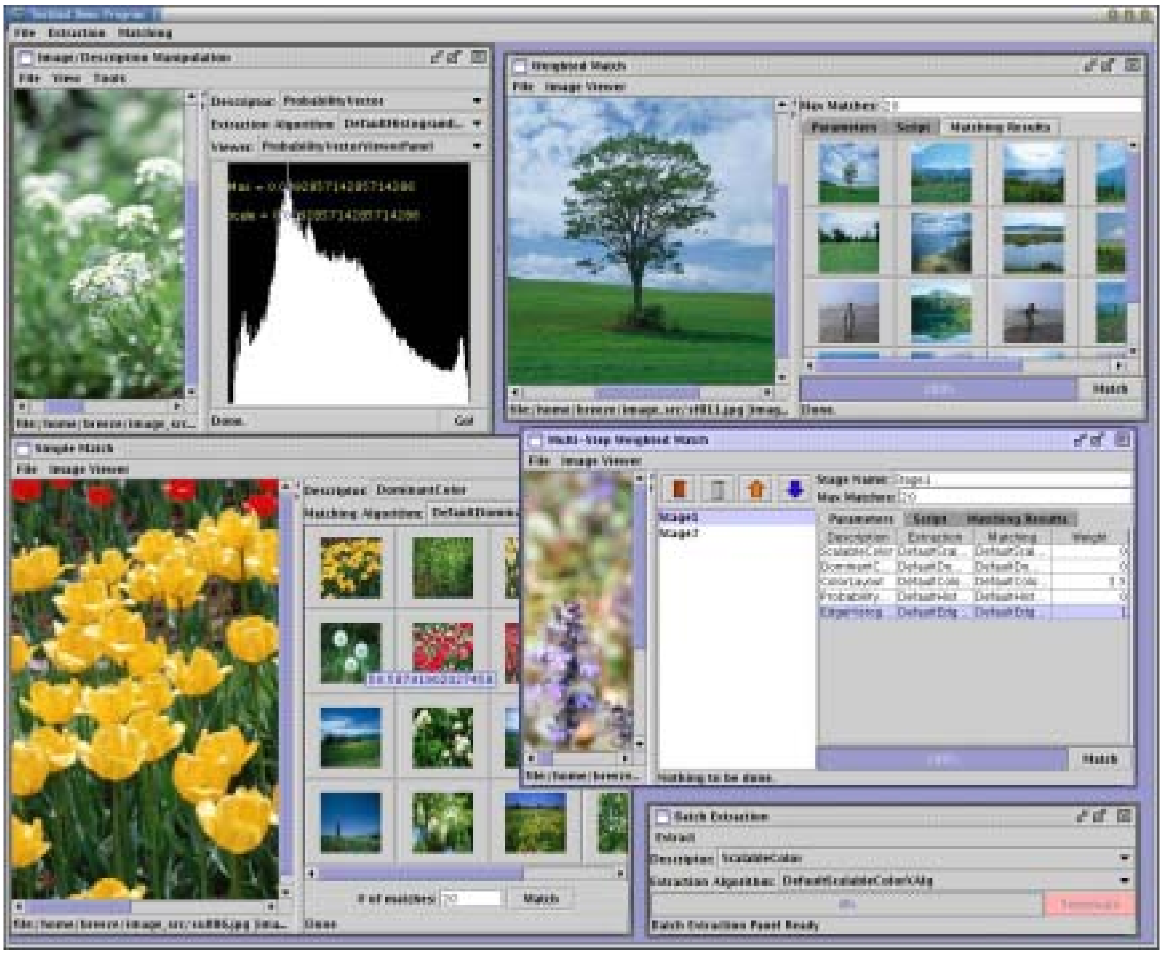


圖十一 軟體平台架構

- 物件存取
  - key(obj)：取得物件的 key，必須是此物件型別下的 unique key
  - listKeys(type)：列出資料庫中某物件型態的有效 key
  - get(key)：按照 key 取出物件
  - insert(key,obj)：將物件加入資料庫
  - remove(key)：將物件移出資料庫
  - match(obj)：依照定物件作相似性比對，並將結果傳回
  - match(obj,alg)：依照指定的演算法對指定物件作相似性比對
- 資料庫單元維護
  - getInstance(config)：依照組態建立資料庫單元
  - configure(config)：依照組態設定資料庫單元
  - register(type, cmds)：將物件存取命令與物件型別關聯起來
  - deregister(type)：取消某物件型別的存取命令

■ 命令搜尋的方式採相容式比對，當有已註冊的命令時，就以此為準，如果沒有，則根據物件型別的繼承關係，先比 interface 再比 super class，如果有相符的命令，就以此相容命令為準，如果沒有，繼續向上搜尋，直到到達繼承關係頂點為止

圖十二所示為利用此軟體平台開發出來的原型程式，在這個程式中，我們除了展示了 Histogram、Color Layout、Edge Histogram、Scalable Color 相關的 feature extraction、feature matching、以及 image/feature viewer，並利用設計的 Component Management 單元展示如何在不更動程式下，新增或移除 Concrete Components。此外，我們在應用程式層次加上額外兩種 matching 方式，稱為 weighted matching 及 multi-step matching，前者可用 weighting factor 方式組合多個 D 或 DS 作為搜尋條件，後者則是進一步將搜尋分成多級，每級皆為 weighted matching，且每級的搜尋範圍限制在前一級的搜尋結果中。實驗結果發現，適當安排各級搜尋條件不僅可以得到較令人滿意的結果，而且搜尋速度與 weighted matching 相差不會太多。除了 weighted matching 類型外，我們也實作了以集合運算結合多種特徵的比對方式，實驗結果顯示在某些狀況下的確可以得到比較好的結果，但缺點是如果集合不夠大，可能無法取到共同交集。


Image/Description Manipulation
Weighted Match
Simple Match
Multi-Step Weighted Match
Batch Extraction
Descriptor: DominantColor
Descriptor: ScalableColor
Parameters
Script
Matching Results
Match
Done

圖十二 軟體平台程式展示

D. 結論

在 MPEG-7 系統的研究中，我們先研讀 MPEG-7 標準及相關參考資料，了解 MPEG-7 的目的、內涵、使用的技術、與提供的工具。接著針對 MPEG-7 XM 不足之處，我們根據 MPEG-7 精神加上日後研究所需，設計軟體測試平台。這個平台以 Java 實作而成，主要是由 framework (Viewer-Data-DataAlg 關聯)、Component Management、以及 Persistence Management 組合所形成的基礎架構。

在此架構之上，我們以數個 MPEG-7 中定義的 Descriptor 為例，在此平台上實作出來，以確認作為 meta-data 測試平台的可行性。最後，設計一個影像特徵抓取與搜尋原型程式，除了展示基本的 MPEG-7 應用方式 (特徵抓取、儲存、比對、整合)，並提供使用者檢視 descriptor 內容的介面，最重要的，此程式驗證了測試平台的設計具有足夠彈性，可用來開發 MPEG-7 相關的研究或應用程式，且在程式測試期間，可以充分利用平台上提供的動態關聯，在不需更動程式碼的狀況下，試驗不同組態下的執行速度與比對準確度。當決定組態後，亦可從平台中將各元件抽離，以程式碼固定關聯性，可減少耗費在元件管理機制中的執行時間。

參考文獻

[1]ISO/IEC JTC1/SC29/WG11 N3747. *MPEG-4 Overview* - (V.16 – La BauleVersion), Contribution for La Baule, October 2000.

[2]ISO/IEC JTC1/SC29/WG11 N3850. *ISO/IEC 14996-1 ,COR1, AMD1.*

[3]ISO/IEC JTC1/SC29/WG11 N2614 *MPEG-4 Intellectual Property Management & Protection (IPMP) Overview & Applications.*

[4]ISO/IEC JTC1/SC29/WG11 N5068, *Study of FPDAM ISO/IEC 14496-1:2001/AMD3, Jul. 2002.*

[5]ISO/IEC JTC1/SC29/WG11 N5333, *MPEG-21 Requirements v.14, Dec. 2002.*

[6]ISO/IEC JTC1/SC29/WG11, N5535, *Requirement for MPEG-21 Intellectual Property Management and Protection, Pattaya, Mar. 2003.*

[7]ISO/IEC JTC1/SC29/WG11, Part 5 – *Reference Software – Systems (ISO/IEC 14496-5 Systems)*

[8]ISO/IEC JTC1/SC29/WG11 N4702, *MPEG-4 IPMP Extension Reference Software Architecture based on IM1, Jeju, Mar. 2002.*

[9]ISO/IEC JTC1/SC29/WG11 N4850, *MPEG-2 and MPEG-4 IPMP Extension Reference Software Architecture based on IM1, May. 2002.*

[10] ISO/IEC FCD 15938-1 *Information Technology - Multimedia Content Description Interface - Parts 1 to 7.*

計畫成果自評

本計畫有以下幾類成果。第一類為研究 MPEG-4 Systems (IPMP)與 MPEG-7 多媒體系統所發展出的技術、經驗及成品極具實用價值，可促進國內工業研發技術開發。第二類為計畫執行過程所獲得之研究成果論文兩篇，已發表於國際學術會議。其三，參與計畫之同學可獲得國際多媒體最先進的 MPEG-4 與 MPEG-7 相關技術及多媒體系統設計經驗，畢業後進入產業，直接有助於產業界開發新產品，提昇我國工業技術能力。達到人才培育之目的。

綜合評估：本計畫產出相當多具有學術與應用價值的成果，並培育高科技人才培育，整體成效良好。已發表學術論文兩篇以及碩士學位論文一冊如下表。

Publications:

(1) C.-C. Huang, H.-M. Hang, and H.-C. Huang, "MPEG IPMP Concepts and Implementation," *The 3rd IEEE Pacific-Rim Conference on Multimedia*, Hsinchu, Taiwan, Dec. 2002.

(2) F.-C. Chang, H.-M. Hang, and H.-C. Huang, "Research Friendly MPEG-7 Software Testbed," *Image and* Video *Communications and Processing 2003 Conference*, USA, Jan. 2003.

(3) Kin-Lam Tong 唐健霖，*An Implementation of MPEG IPMP System,* MS Thesis, NCTU, June 2003.

附錄

1. C.-C. Huang, H.-M. Hang, and H.-C. Huang, "MPEG IPMP Concepts and Implementation," *The 3rd IEEE Pacific-Rim Conference on Multimedia*, Hsinchu, Taiwan, Dec. 2002.
2. F.-C. Chang, H.-M. Hang, and H.-C. Huang, "Research Friendly MPEG-7 Software Testbed," *Image and* Video *Communications and Processing 2003 Conference*, USA, Jan. 2003.

<附件一>

Advances in Multimedia Information Processing – PCM 2002, pp. 344-352,
Springer-Verlag, Dec. 2002.

MPEG IPMP Concepts and Implementation

Cheng-Ching Huang[1], Hsueh-Ming Hang[2] , and Hsiang-Cheh Huang[2]

Department of Electronics Engineering, National Chiao-Tung University,
Hsinchu, Taiwan.
[1]cchuang.ee89g@nctu.edu.tw
[2]{hmhang, huangh}@cc.nctu.edu.tw

Abstract. Intellectual Property (IP) protection is a critical element in a multimedia transmission system. Therefore, ISO/IEC MPEG started the IP protection standardization project on MPEG-4 a few years ago. A basic IPMP (Intellectual Property Protection and Management) structure and interface was first defined in its System part. In this paper, we will first outline the MPEG-4 basic IP protection mechanism and then describe our simulation of an MPEG-4 IPMP system. An IP protection application is constructed using the MPEG-4 system software – IM1 (Implementation Model one). This application includes a client-server program, in which a client can request the keys from a server in a secure way using a hierarchical key distribution structure.

1 Introduction

With the rapid development in computer industry and the swift growth of Internet, there is a widespread use of the digital multimedia contents in our daily life. The progress in data compression techniques also makes transmission of multimedia data stream possible. However, Internet is an open environment, therefore, if the user data and information are not protected, it might be illegally used and altered by hackers. To protect privacy and intellectual property (IP) right, people often use cryptographic techniques to encrypt data, and thus the contents protected by encryption are expected to be securely transmitted over the Internet.

One requirement of typical multimedia applications is the demand for real-time transmission. In contrast, conventional security methods are often designed to protect digital data files, which might not be suitable and efficient for real-time applications. To fulfill the demands for both real-time distribution and data security, including the IP protection mechanism into the multimedia standard might be a feasible and effective way to achieve an unambiguous communication environment.

MPEG (Moving Picture Expert Group) is the ISO committee to set up the international standards for multimedia data exchange. MPEG-2 has been applied to digital video broadcasting with some access control specifications [1][2]. IPMP (Intellectual Property Management and Protection), proposed for MPEG-4 standard, aims at protecting the compressed multimedia. In this paper, we will describe and implement a multimedia transmission system using the MPEG-4 IPMP concepts.

This paper is organized as follows. Sec. 2 is an overview of the MPEG-4 System and IPMP standards. Sec. 3 describes the IPMP plug-ins in the MEPG-4 System reference software "IM1." Sec. 4 describes the procedure of constructing the MPEG-4 IP plug-ins and an application example is included. Sec. 5 concludes this paper.

2 MPEG-4 Standard Overview and IPMP Framework

MPEG-4 is an international standard defined by the ISO/IEC committee. Compared to it predecessors, MPEG-4 pays more attention on the following three subjects: (i) real-time streaming, (ii) object-based coding, and (iii) enriched user interaction.

MPEG-4 standards contain 10 parts. The portion related to IP protection is in the first part, Systems. The IPMP framework in ISO/IEC 14496 consists of a normative "interface" that permits an ISO/IEC 14496 terminal to host one or more IPMP subsystems. An IPMP sub-system is a non-normative component of terminal, which provides several intellectual property management and protection functions. At the moment, MPEG committee is refining and extending the MPEG-4 IPMP specifications. A Message Router mechanism is to be added into the third Amendment of 14496-1.

In the MPEG-4 standards, the IPMP interface consists of IPMP elementary streams and IPMP descriptors. The IPMP elementary streams usually convey time-variant information such as keys associated with the encryption algorithm, which may change very rapidly. IPMP descriptors often convey time-invariant information associated with a given elementary stream or a set of elementary streams. IPMP elementary streams are treated as regular media elementary streams. And the IPMP descriptors are transmitted as part of an object descriptor stream.

Fig.1 shows how an IPMP sub-system works in an MPEG-4 terminal. Almost all the streams may be controlled or accessed by the IPMP sub-system but the Object Descriptor streams shall not be affected by the IPMP sub-systems.

Stream flow controller is a conceptual element that accompanies with every elementary stream. Stream flow controller can take place between the SyncLayer decoder and the decoder buffer. As Fig. 1 indicates, elements of IPMP control can take place at other points in the terminal. For example, they can appear after decoding (as in the case with watermark extractors).

3 IPMP in IM1

IM1 is an MPEG-4 Systems software developed by the MPEG committee. It may be used to verify and demonstrate the functionalities of MPEG-4 [4].

The Systems Core module in IM1 defines the infrastructure to implement MPEG-4 players. It provides the functionality of MediaObject, the base class for all specific node types. The API for Decoder, DMIF and IPMP plug-ins is also supported by IM1. Moreover, the code is written in C++, which is fairly platform-independent [5].

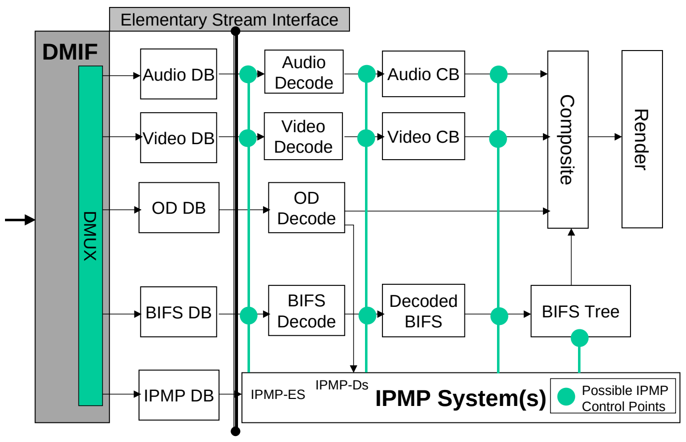


Fig. 1. IPMP sub-system in the ISO/IEC 14496 terminal architecture [3]

3.1 IPMPManager

In IM1, IPMP sub-systems are implemented by extending the IPMPManager class. IPMPManager is an interface between MPEG-4 player and the IPMP sub-system. Each media content access unit goes through the sub-system before it is stored in the decoding buffer. An implementation of IPMPManager can decrypt the encrypted content and thus block the unauthorized access to the media content.

IPMPManagerImp extends the IPMPManager interface, and it provides the major functionality of an IPMP sub-system. Simple implementations need to overload a few setup functions and the *Decrypt()* function, which decrypts one access unit using one IPMP stream. More complex implementations, for instance, when multiple IPMP streams are used to decrypt a single elementary stream, may overload the *Run()* function and implement different data flows by directly accessing the MediaStreams.

IPMP plug-ins interact with the core codes of the player through a special kind of buffer, known as MediaStreams. An IPMPManager object fetches an access unit, which is a kind of media, from one MediaStream object. After decrypting an access unit, it will dispatch one decrypted access unit into an output MediaStream object, which usually is a decoding buffer [6].

3.2 IPMPManagerImp

IPMPManagerImp extends the IPMPManager interface. It is the base class of all the IPMP sub-systems. IPMPManagerImp provides all the needed functions of a regular IPMP sub-system.

Each IPMP sub-system runs on its own thread. An IPMP sub-system is usually attached to three MediaStream objects – the encrypted input stream, the decrypted output stream, and the IPMP stream. According to the SDK [6], the workflow of a typical IPMP sub-system is shown in Fig.2. Our design procedure is modified from that in [6] and is outlined below.

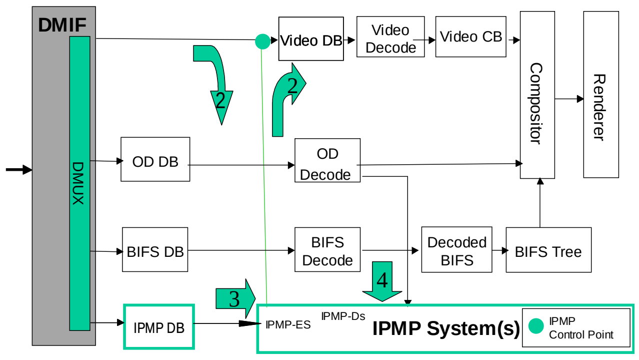


Fig. 2. A typical IPMP sub-system workflow

1. An object derived from IPMPManagerImp is instantiated by the IPMP sub-system module (usually a Dynamic Link Library, or DLL).
2. The application calls *IPMPManager::SetInputStream()* and *IPMPManager::Set OutputStream()* to attach input and output MediaStreams to the IPMP sub-system.
3. The application calls *IPMPManager::SetIPMPStream()* to attach an IPMP stream to the IPMP sub-system. This function may be called more than once if the elementary stream is protected by multiple IPMP streams.
4. The application calls *IPMPManager::SetDescriptor()* for each IPMP descriptor assigned to the elementary stream.
5. The application calls *IPMPManager::Init()* to initialize the IPMP sub-system and to confirm that the user has access to the protected elementary stream.
6. The application calls *IPMPManager::Start()*, which spawns the IPMP sub-system thread.
7. The IPMP sub-system thread fetches an access unit from the input stream and the corresponding access unit from the IPMP stream. Note that one IPMP access unit can control multiple content access units.
8. The IPMP sub-system calls a private virtual function, *Decrypt()*. This function is overloaded by specific IPMP sub-systems and performs the actual decryption.
9. The output of *Decrypt()* is stored in the output MediaStream.
10. Steps 7-9 are repeated until *IPMPManager::Stop()* is called by the application, or until reaching the end of the input stream.

Some of these steps have been implemented in IPMPManagerImp class, but in some special cases, we need to re-implement them again.

3.3 MediaStream

MediaStream class handles the buffering and synchronization of an elementary stream. It manages the memory buffer and fetch/disfetch access units from the buffer. The stored access unit maybe has time stamp on it. The current solution is to fetch the access unit immediately and ignore the time stamp, fetch the matured unit only, or otherwise suspend.

4 Constructing an MPEG-4 IPMP Application Example

We will implement and demonstrate a multimedia transmission system with MPEG-4 IPMP by incorporating modern cryptographic techniques [7]. In designing the system, we adopt the Conditional Access (CA) concept by using a hierarchical key distribution structure as shown in Fig. 3.

In this system, we encrypt only the bitstreams in the TRIF files. The server generates and embeds the keys into the bitstream. When the keys are correctly retrieved, the decoded and decrypted video sequence can be played properly. Otherwise, the bitstreams cannot be decoded successfully.

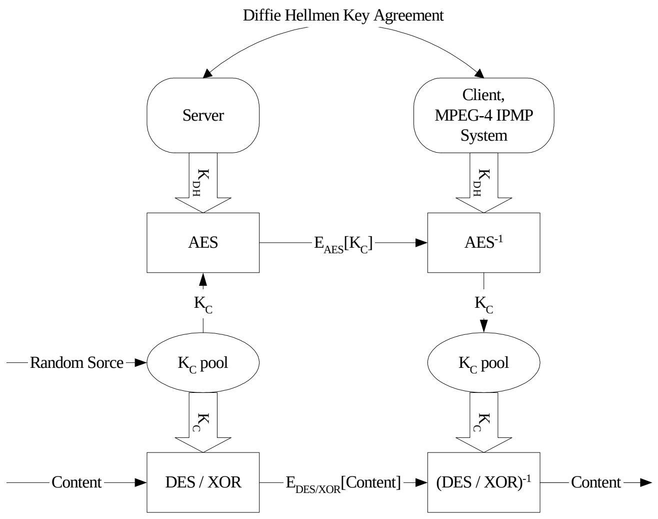


Fig. 3. The hierarchical key distribution structure

4.1 System structure and handshaking protocol

Our hierarchical key distribution system is illustrated by Fig. 3. At the upper level, we use the Diffie-Hellmen Key Agreement [8] that enables both the client-end and the server-end to securely retrieve the Session Key, K_{DH}, over the Internet. By applying the Advanced Encryption Standard (AES) [9], K_{DH} can serve as a secret key to encrypt K_C, and the encrypted K_C are then transmitted. The use of K_C is to serve as the key for the bottom layer encryptor. In our example, the contents to be encrypted are the compressed video, audio, or image bitstreams. Similar to the CA system in DVB, we achieve the security requirement by changing K_C frequently. The throughput of K_C is so high that we need a K_C pool to generate the keys constantly.

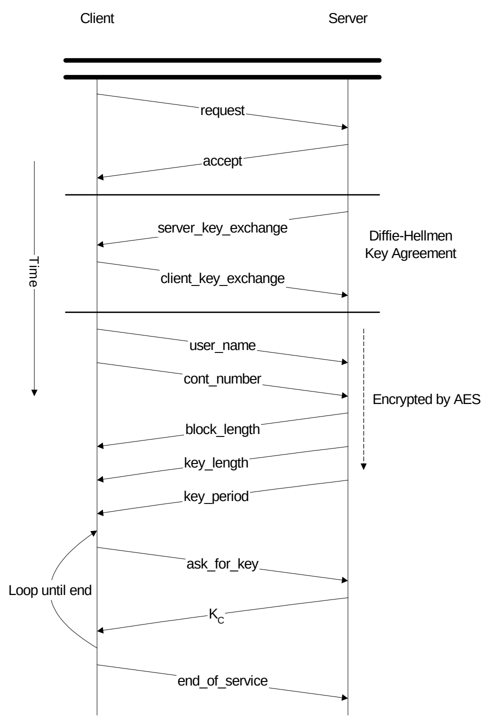


Fig. 4. The handshaking protocol

One of the most important elements of our system is the handshaking protocol. Fig. 4 shows the basic steps in establishing a connection between the client-end and the server-end. The procedure is stated as follows.

1. Client sends request = 0x31403 (4 bytes).
2. Server sends accept = 0x31403 (4 bytes).
3. Client and server proceed with the Diffie-Hellmen key agreement; all the forthcoming information will be encrypted with AES by this key.
4. Client sends user_name (44 bytes) and cont_number (4 bytes), representing the user name and content number, respectively.
5. Server sends block_length (2 bytes) and key_lengh (2 bytes) to initialize the encryptor, and key_period (1 byte) to tell the bottom layer encryptor the lifetime of K_C.
6. Client sends ask_for_key = 0x5327 (4 bytes) to ask for a new key from the server. Server sends a new K_C to client after receiving ask_for_key.
7. Client sends end_of_service = 0x0 (32 bytes) to terminate the handshaking.

4.2 The client-end IPMP plug-in

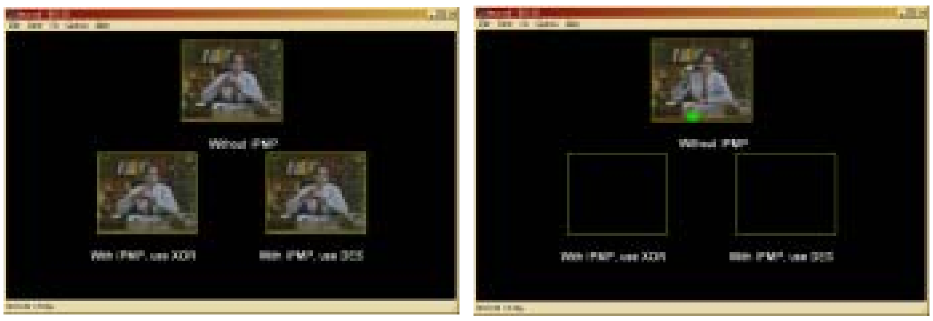
The player is the “IM1 2D player” executed under the Windows environment. Hence, the IPMP plug-in can be implemented with the DLL file in Windows.

There is one implementation of IPMP plug-in in IM1 core called IPMPNull. It works like a buffer to send the input MediaStream directly to the output side. Based on the existing IPMPNull program, we implement two new IPMP plug-ins in our system: IPMPXOR.dll and IPMPDES.dll. They are essentially two encryption methods. The first plug-in conducts the XOR operation between the received bitstreams and the key at the decryptor, a very simple encryption technique; the second one uses the DES [7] scheme for decryption.

In the MPEG-4 design, the IPMP stream is used to transmit keys. In our example, we transmit the key using TCP/IP, not DMIF, to avoid the incompatibility between our example system and the standard system.

The first step to implement IPMPDES is to create an IPMPDES class, and it inherits a class called "IPMPManagerImp." Then, we implement the *SetDescriptor()* function. The IPMPDescriptor within the TRIF file contains the information of the server location and the content identification number that is to be played. The *SetDescriptor()* function uses the above information to make a connection to the server and to initialize the decryptor locally. Next, we implement the *Decryptor()* function, which can decrypt the received MediaStream, and count the number of times K_C is used.

Figs. 5(a) and 5(b) are the demonstrations of two IM1 2D players. The video sequence is coded in the H.263 format. The bottom-left bitstreams in both figures are decrypted by IPMPXOR, and the ones on the bottom-right are decrypted by IPMPDES. The sequences on the upper side are not protected in both Figs. 5(a) and 5(b). In Fig. 5(a), we assume that the key can be reliably transmitted and received. Hence, the two encrypted bitstreams can be decrypted and displayed successfully. In Fig. 5(b), the keys are not retrieved. Thus, the encrypted bitstream cannot be decoded and displayed.



(a) (b)

Fig. 5. Demonstration of the proposed system: the unprotected bitstreams (*upper*) and the protected bitstreams (*lower*). (a) Correctly retrieved keys, and (b) keys not retrieved

4.3 The server-end

The server can be divided into two parts, one is the encryptor and the other part is responsible for sending keys. Fig.6 is a screenshot of the server-end application. We write it in C++ and it is a DOS command-line program. But the GUI is done in Java using the pipes stdin and stdout.

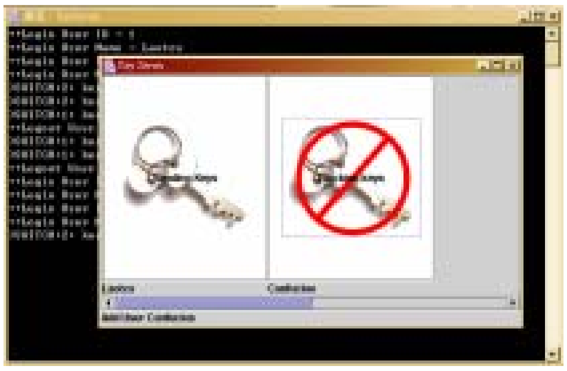


Fig. 6. The server that can turn on/off keys

5 Conclusions

In this paper, we first briefly describe the MPEG-4 IPMP system concepts. We then analyze the IPMP API in the reference software of the MPEG-4 Systems – IM1. After studying the IM1 Core and its IPMP API, we implement a functional IPMP sub-system by modifying IPMPNull – a prototype of the IPMP sub-system.

We use the hierarchical key architecture to construct an application example, following the MPEG-4 IPMP concepts. Our example simulates the functionalities suggested by the standard. We demonstrate that the MPEG-4 IPMP is a practical way for protecting the multimedia content.

6 Acknowledgement

This work was supported by National Science Council (Taiwan, ROC) under Grant NSC 90-2213-E-009-137.

References

1. ISO/IEC 13818-1 *Generic Coding of Moving Pictures and Associated Audio Information*: *Part 1 Systems* (ISO/IEC JTC1/SC29/WG11 N0801rev), April 1995.
2. H. Benoit, *Digital Television, MPEG-1, MPEG-2 and Principles of The DVB system*, Arnold, 1997.
3. ISO/IEC 14496-1:2000(E) *Coding of Audio-visual Objects*: *Part 1 Systems* (ISO/IEC JTC1/SC29/WG11 N3850), October 2000.
4. ISO/IEC JTC1/SC29/WG11 N4291, *MPEG Systems (1-2-4-7) FAQ*, Jul. 2001.
5. ISO/IEC JTC1/SC29/WG11 N4709, *MPEG-4 Systems Software Status and Implementation Workplan*, March 2002.
6. ISO/IEC JTC1/SC29/WG11 M3860, *IPMP Development Kit*, Aug. 1998.
7. B. Schneier, *Applied Cryptography, 2nd edition*, John Wiley & Sons, 1996.
8. *PKCS #3: Diffie-Hellman Key-Agreement Standard*, An RSA Laboratories Technical Note, ftp://ftp.rsa.com/pub/pkcs/ascii/pkcs-3.asc.
9. J. Daemen and V. Rijmen, *AES Proposal: Rijndael* (corrected version), http://www.esat.kuleuven.ac.be/~rijmen/rijndael/rijndaeldocV2.zip.

<附件二>

Research Friendly MPEG-7 Software Testbed

F.-C. Chang, H.-M. Hang, and H.-C. Huang[a]

[a]Department of Electronics Engineering, National Chiao Tung University (NCTU), 1001 Ta-Hsueh Road, Hsinchu, Taiwan 30010, R.O.C.

ABSTRACT

We design and implement a research friendly software platform, which aims at the flexibility and the abstraction of MPEG-7 application prototyping. We studied and analyzed the MPEG-7 standard, including a typical scenario of using MPEG-7. In order to fulfill to needs of researches, in addition to the normative parts of MPEG-7, additional requirements are included. By examining these requirements, we propose a research friendly software platform. The architecture consists of a framework, utility units, and the descriptors. Because this system is implemented using Java, it also incorporates the features of the Java environment, and thus it is flexible for developing new components and prototyping applications. We demonstrate the flexibility of this testbed by constructing an example program which allows users to manipulate image related descriptors.

Keywords: MPEG-7, testbed, platform, feature extraction, matching

1. INTRODUCTION

With the sweeping development and deployment of computers, the widespread use of the internet, and the advances of the audio-visual technology, it is easy to produce, distribute, and consume numbers of multimedia contents.[1,2] The enormous amount of digital contents hence make searching and retrieval a difficult task. The emergence of MPEG-7 aims at solving the problems for multimedia searching and retrieval.

MPEG-7,[3,4] also called *Multimedia Content Description Interface*, is an efficient tool for searching and retrieval of multimedia data. MPEG-7 specifies a standard way of describing various types of audio-visual information irrespective of its representation format and storage support.

The objective of MPEG-7 is to standardize content-based description for various types of audio-visual information. It enables fast and efficient content searching, filtering, and identification. Therefore, MPEG-7 provides a rich set of standardized tools to describe multimedia content[5] with a minimum set of tools, called the normative components, essential for interworking, including *Descriptor* (D), *Description Scheme* (DS), and *Description Definition Language* (DDL).[6] A Descriptor is a representation of a distinctive characteristic of the data. A DS specifies the structure and semantics of the relationships between its components, which may be Descriptors or Description Schemes. The DDL is a language that allows the creation of new DSs and Ds, and the extension and modification of existing DSs.

In addition to these standardized tools, the MPEG-7 eXperimentation Model (XM)[7] is a reference software used to evaluate the standard Ds and DSs. It is a collection of programs contributed by the members of MPEG standard committee. Originally, each piece in it implements a straightforward MPEG-7 application type, which generates a single feature and lists the matching results.

For real world applications, both the description generation, such as feature extraction, and consumption, such as search engine, are essential but they are non-normative parts of MPEG-7 (Fig. 1). We thus are

Further author information: (Send correspondence to H.-M. Hang)
H.-M. Hang: E-mail: hmhang@cc.nctu.edu.tw, Telephone: +(886)3-5731861, FAX: +(886)3-5723283/5731791
F.-C. Chang: E-mail: u8811833@cc.nctu.edu.tw
H.-C. Huang: E-mail: huangh@cc.nctu.edu.tw

motivated to design a platform that allows MPEG-7 researchers to develop various features and algorithms on this platform. Users can manipulate the MPEG-7 descriptors in the abstraction layer without concerning of the low-level functions, such as system calls and database connections. They can dynamically assemble different algorithms together, execute them, and view the generated data in various formats by visual viewer tools. Also, they can compose a new searching scheme (search engine) and test its feasibility on this testbed.

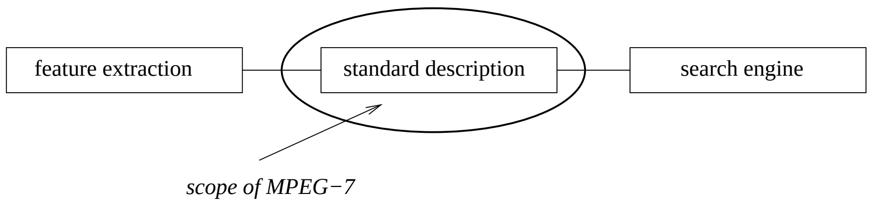


Figure 1. Scope of MPEG-7

This paper is organized as follows. Sect. 2 describes the motivation behind this project. Sect. 3 proposes the software architecture of this research friendly platform. The detailed system design and architecture implementation are described in Sect. 4. An application example is presented in Sect. 5. Sect. 6 concludes this paper.

2. MOTIVATIONS

Many different types of applications can be developed based on the MPEG-7 specifications. For example, a multimedia search engine typically has its own scheme in organizing and processing the meta-data. This scheme may comprise several standard Ds and DSs, but it is likely that we need to create application-specific descriptions and operations, and then to evaluate their performance. As discussed earlier, the MPEG-7 XM is a collective software set. Although it includes a few basic tools that can incorporate new Ds and DSs, it is mainly organized to demonstrate the extraction/matching results of using individual Ds (and/or DSs) and it has no database support. The database support is essential for a typical multimedia database application.[8] For research purposes, we need a platform that not only produces algorithm outputs, but also helps us analyzing the performance of an algorithm. We group the MPEG-7 related research topics into two categories. One category is developing a new description and its companion extraction and matching algorithms; the other is developing a new application, which is a composition of existing descriptions and algorithms.

For new description development, what we care is how useful the description is, and how well the extraction/matching algorithm performs. It is helpful if we have an environment that allows us to add and remove description data structures and algorithms easily. We evaluate the newly designed structures or algorithms by their experimental results. And the results need to be examined by various data visualization viewers.

For a new application development, we usually pick up several existing descriptions from a library, organize them into a scheme, and evaluate the new scheme to check its performance. A simple combination of the components, each designed originally for a single objective, may not perform well enough to achieve our goal. A carefully arranged processing sequence together with dedicated database schema design may greatly enhance the overall performance. After designing the application-specific descriptions and algorithms, we also need a versatile interface for handling the inputs and outputs of MPEG-7 streams.

3. ARCHITECTURE

Motivated by the discussions in Sect. 2, we start to design a research software platform. The design process to be elaborated below includes defining the requirements, analyzing the use-cases, and finally developing an architecture of the testbed. The unified modeling language (UML),[9,10] developed by the Object Management

Group (OMG),[11] is a graphical language for visualizing and documenting a software intensive system. We will use UML to design our platform throughout this paper.

3.1. Requirements of the Testbed

The platform should provide an environment for both developing descriptors and feature-based applications; correspondingly, the users of the system are classified into two categories. We call the descriptor developers the "designers", and call the application developers the "developers". The common requirement for both groups of users is a flexible and extensible environment for conducting MPEG-7 related researches that involve algorithm simulation and data analysis.

The designers develop descriptors and their companion algorithms, such as extraction, matching, encoding, and decoding. We need a well-organized architecture that enables the designers to design the structure of descriptions, the algorithms to extract features from media contents, the algorithms to compute the similarity measure between two objects, and the encoding/decoding algorithms. In some cases, designers may want to compare different algorithm implementations for the same descriptor, *e.g.*, distance calculation based on the quantized values or the reconstructed values. Using this platform, designers can test their descriptors and algorithms by the quick prototyping mechanism.

In addition to evaluating the correctness of algorithms, the designers may want to evaluate the efficiency or the fitness of the algorithms. Sometimes we can rate the efficiency of a descriptor by examining the query results. For some descriptors, subjective examination of the extracted feature values may be helpful. To facilitate examining various types of descriptors, feature visualization components are necessary in our platform.

A designer concentrates on a single descriptor design, while a developer focuses on a larger scale of software organization. The typical scenario for a developer is picking up desired descriptors and algorithms from a library, preparing the input/output interface, organizing the operations, attaching the visualizers, and building the main program. The scenario suggests that the developers need an architecture that enables easy composition of the ready-to-use components developed by the designers.

In developing and prototyping MPEG-7 applications, the platform should be flexible for integrating various components into a complete system. This requires not only a structure that holds all the components, but also a management mechanism that gets/puts components in an organized fashion. Because a large percentage of applications are search-engine-like, the condition of multi-user accesses should be considered. This implies that the platform should be able to evaluate the performance when a new scheme is involved in concurrent processes. The database support, or the persistence mechanism, is critical in developing an application. It may not be a critical element for a designer, because a designer usually deals with a small set of test data. For a real application, a developer should concern the organization of the database; otherwise, the execution performance may degrade drastically with a large data set.

3.2. Use-Cases

A use-case diagram specifies the behaviors of a system and also it captures the expected interactions that occur at run-time. It is important in that it serves to validate the software architecture (which will be discussed in Sect. 3.3) during project development.[9] The requirements of the testbed suggest many use-cases between the system (the testbed) and the actors (the users). According to the classification of users discussed in Sect. 3.1, we identify three types of actors in the use-case diagram: the user, the designer, and the developer. A user, no matter he/she is a designer or a developer, may want to decode the input media data, extract some descriptors, and calculate the distances among descriptors. Since our system views the data structures and algorithms as components, the user needs to execute the component fetching operation to get the desired components before he/she can really perform the other operations.

In the following paragraphs, we will discuss the designer/developer specific use-cases. Fig. 2 shows the use-cases for designers. A designer performs two types of specific operations: one is to register the developed components, and the other is to view a descriptor by a visualizer. Examining the requirements, we know that the designer produces three types of components. The first is the descriptor. It is a pure data structure that holds the extracted multimedia features. The second type of component is the (data) associated algorithms, such as extraction, matching, encoding, and decoding algorithms. The last one is the descriptor visualizer (we call it "viewer" in the following text for abbreviation). When the designer registers a component to the system, the system also records the associations related to a component. For example, we register a vector viewer with associations connected to the histogram descriptor. Then, the system records that the vector viewer is an available viewer of the histogram. Without the association records, the system is nothing but simply a component holder.

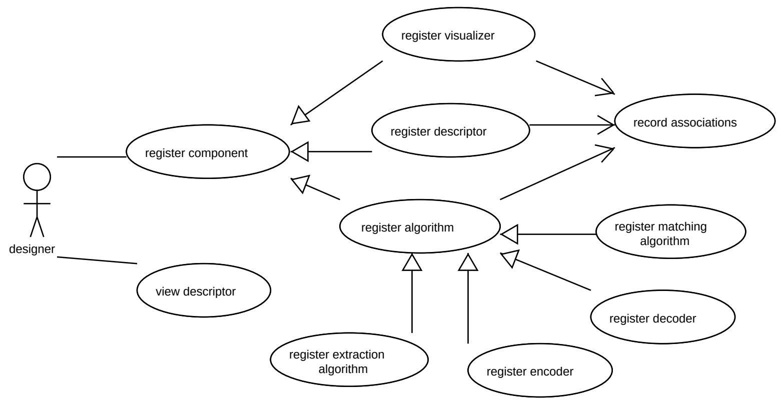


Figure 2. Use-Case diagram for designers

Fig. 3 illustrates the use-cases for developers. A developer performs three specific types of operations. The developer may want to extract application-specific features, calculate the similarity between two features, and export the feature values according to an external format. For example, a developer extracts the color histogram of a query image, transforms the internal feature representation to MPEG-7 format, dispatches the MPEG-7 feature to a search engine, and searches the best matches by comparing features in a local database. It is worth noting that the extraction and matching operations are different from those of general users in two aspects. First, the "features" that a developer requests is usually a composition of descriptors residing on the platform. Second, the extraction/matching algorithms are application-specific, because the MPEG-7 standard does not specify any method of combining multiple-descriptor algorithms.

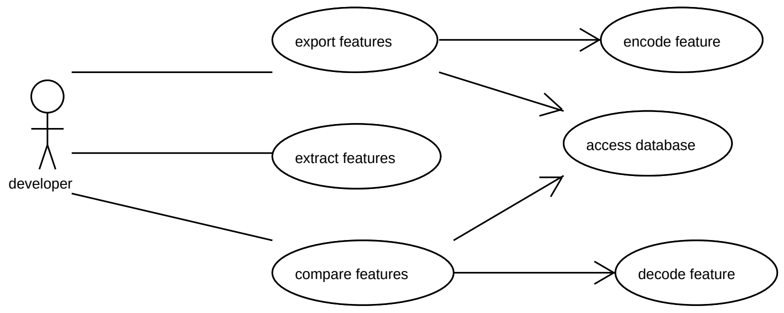


Figure 3. Use-Case diagram for developers

3.3. System Architecture

Before we design and implement the detailed structure of the system, we need an architectural view of the platform. The requirement and use-case analysis is focused on the behavioral view of the system. They explicitly describe how the system works, and implicitly hint what the system should have. We develop the main structure from those analysis results, especially the nouns appearing on the descriptions and the diagrams.

The system contains a number of concrete components, including descriptors, algorithms, and viewers, as shown in Fig. 4. To manage those components, the platform has a component management unit that deals with the registration and association of the components. The database support is handled by the persistence management unit. These two units are not directly related to the descriptor design, but are necessary to the application and prototype development. Hence, they are utility units of the testbed. The most important part, though not explicitly mentioned in the requirements, is the framework. The framework is the programming styles and rules when we use the platform to develop new applications. For instance, the common programming interface of using the components and the concurrent execution algorithms are defined in this part. It glues the scattered components and utilities by programming interfaces and class libraries.

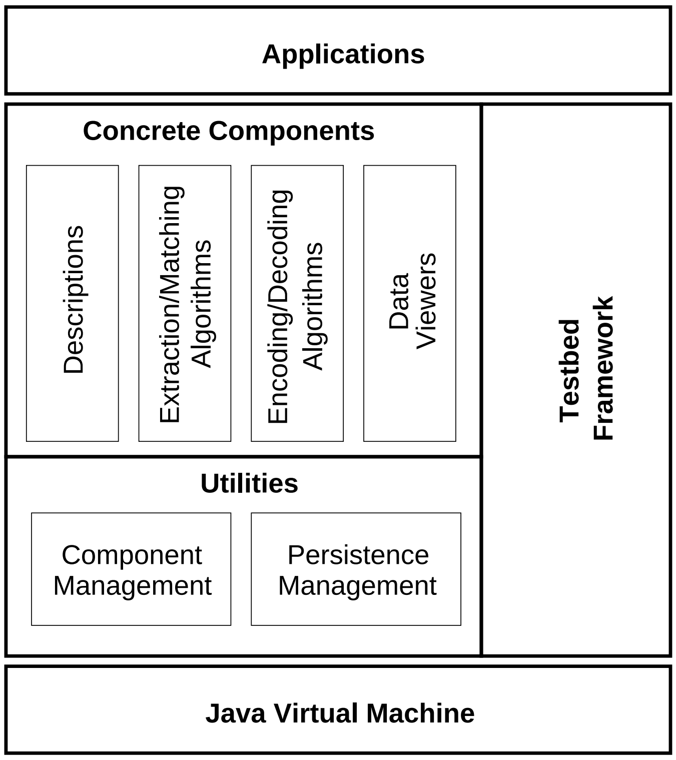


Figure 4. Architecture of the software platform

In general, database access and data input/output are the less portable part of an application, because they differ from system to system. We choose the Java platform as the base of our testbed, because it hides most platform dependent issues under the Java virtual machine. The Java platform tries to unify the database accesses through the JDBC drivers.[12,13] In our experiences, almost all JDBC driver implementations conform to the SQL standard, and handle text-based data very well. As to the binary data access, it is not always well-implemented in the drivers. Since binary object access is inevitable in a normal MPEG-7 application, this issue is handled by the persistence management unit rather than JDBC.

The interfaces exposed to the designer and the developer are the framework and the implemented components. Since these two parts are designed to be system independent, this framework enables users to create or use the system components without knowing the underlying system configuration. Another feature of this architecture is component re-use. As projects (applications) are gradually developed on this platform, more and more components are registered on the system. It is beneficial to be able to re-use the previously developed modules.

4. IMPLEMENTATION

We will describe the implementation of our proposed testbed in this section. Based on the proposed architecture in Sect. 3, the core of the platform is the framework and the utilities units. The concrete components are the extensions to the core. Our design will proceed from the use-case and the architecture level to the class level.

4.1. Framework Implementation

The framework is the most important part of our platform. For users, it is the interface that connects their applications to the testbed. For the system itself, it is the internal structure that holds the relationships among components. We identify the functionalities of the system by use-case diagrams in the last section, and we now identify the basic classes in our system. These classes are called "domain classes." Fig. 5 shows the classes we identified.

First of all, the Viewer/Data/DataAlg classes are the root of all subsequent components. Data, as the name implies, is a collection of bytes that represent the multimedia content. Specifically, it can be a picture, a video clip, a duration of sound, or any meaningful media data. The DataAlg is the algorithms associated with the data. For example, JPEG encoder and decoder are two DataAlgs bounded to JPEG data. Viewer classes are the renderers for Data classes, such as an image viewer, a sound player, and etc.

There are two special branches of DataAlg: one is for media data and the other is for meta-data (descriptions). The reason we separate these two types of algorithms is that the operations for media data and meta-data are significantly different. The operations applied to media data are encoding and decoding, while the meta-data are usually connected to feature extraction and feature matching. It is often reasonable to group encoder/decoder and extraction/matching algorithms in pairs. Note that we treat an MPEG-7 descriptor as a kind of media data, and hence the MPEG-7 bit-stream is the encoded result of a plain data structure. The associations among viewers and the data are not hard-coded, because a viewer can display more than one kind of data. For example, a vector viewer can display the histogram or the zig-zag scanned DCT coefficients of an image block. The association between data and algorithm is not one-to-one, because several algorithms may be associated with the same data. All the association management tasks are delegated to the ComponentMgr which will be discussed in Sec 4.2.

The domain classes are the conceptual building structures of the platform. In constructing the testbed, we need to define more detailed behaviors of the classes. The Data interface is a wrapper of the real data, and it is the most generic type in the testbed. It acts as a tag interface indicating if the wrapped object is acceptable by this system. Programmers can retrieve MIME-type information through this interface if the system can recognize the object when it is constructed.

The DataAlg is also a tag interface, which represents the algorithms associated with the Data object. It is the root of all algorithm classes. It intends to serve as a common interface of the system to manage algorithms. For the developers (sometimes the designers), it is not directly used as often as its inherited classes. For example, the encoding/decoding algorithms consume MediaData. We use MediaCodec to group them for management, and we use the inherited MediaEncodeAlg class to implement the encoder.

Here we briefly describe how a designer can use our platform to prototype a simple evaluation application. He or she first implements the feature extraction and matching algorithms. Then, he/she specifies a designed ExtractionAlg to consume the MediaData and produce a Description; the MatchingAlg computes the similarity

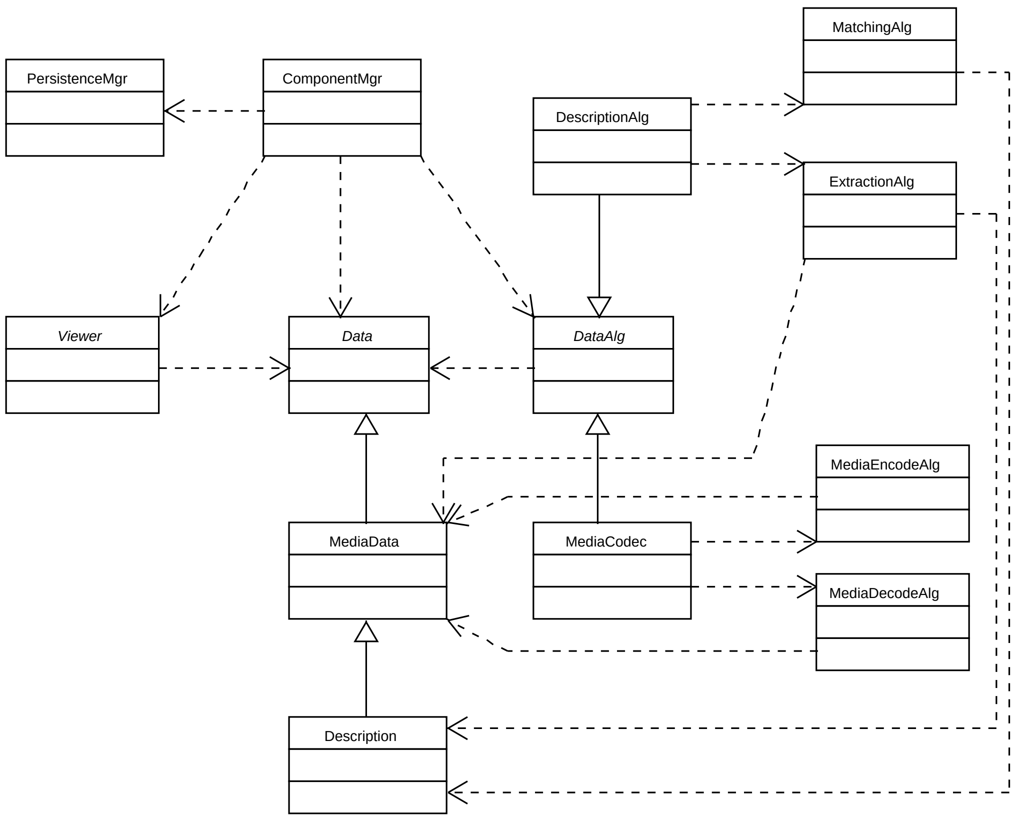


Figure 5. MPEG-7 testbed domain classes

of two Descriptions and gives the distance as the measure of the similarity. To generate a list of matching scores under this framework, the designer may conduct one-by-one comparison. Obviously, this may not be the best solution to a database search, but we view this an application specific issue and we let the developer handle it.

A Viewer is used to view the data object. Following the notations of the model-view-controller (MVC) pattern,[14] the Viewer interface merely indicates that it would co-operate with the model object (the Data) and it provides the general application programming interface (API) for feeding the data into the viewer. Note that we do not define the controller interface in our system, because different applications demand different interactions between a view object and a model object.

The multi-user concurrent execution requirement can be solved by the threading mechanism in Java. Algorithms are a collection of executable codes and related objects. Their properties are similar to those of a thread. Hence, we can implement the algorithms by using the Java Runnable interface, which extends the usability of the class in both single-thread and multi-thread design.

4.2. Utilities

The ComponentMgr manages the associations among viewers, data, and algorithms. One of the requirements of our testbed is to integrate a vast variety of components with their implementations. As more and more applications added into the system, the number of associations grows at a very fast pace. This unit copes with the complexity of the relationships among concrete classes.

The associations of the components are described by the Data-Viewer-DataAlg relationships. They are complicated because they are multi-to-multi associations. To make them simpler, we choose Data as the main entity and break the triple-relation into two double-relations. As illustrated in Fig. 6, one of the association is the relationship between the Viewer and the Data, and the other is the relationship between the Data and the DataAlg. The ComponentMgr holds all the valid associations. The manager provides the interface that retrieves/adds/removes the corresponding Viewers and DataAlgs for a given Data.

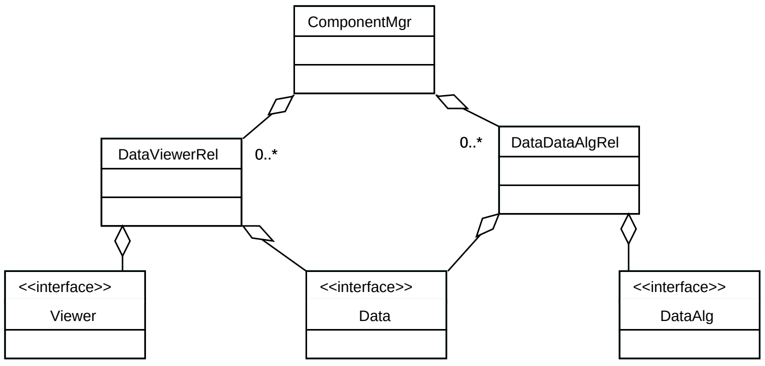


Figure 6. Relationships managed by the component management

The relationships should be stored for future use; however, it is not a good solution to record them as raw objects. The object references should be in an "indirect" form, such as string representation of the class names. We recover a real object from its indirect form. The Java platform provides the class-class, which loads the class by its name and instantiates an object from its class.

The system architecture allows that the users to store objects without knowing detailed database access commands. To satisfy this requirement, we design the persistence management unit, PersistenceMgr, which is a singleton[14] class, the run-time object it points to can be configured in the global configuration.

The PersistenceMgr also configures the JDBC driver automatically, and allows users to obtain Connection objects easily to handle the SQL commands. The binary data persistence is handled by other methods. Currently, our platform supports the binary object persistence of the PostgreSQL database. The implementation uses the LargeObject API provided by PostgreSQL to manipulate bit-stream type data.

5. AN APPLICATION EXAMPLE: A GENERIC IMAGE FEATURE MANIPULATION PROGRAM

In this section, we will demonstrate the flexibility of our testbed by constructing an example program. This application provides a user interface that manipulates image-related descriptors. The testbed framework makes it possible to operate different descriptors through a unified interface. Newly created descriptors can be easily included into the program, because the component manager has the ability to incorporate different kinds of descriptors.

Similar to the testbed design, we start developing this application program from its requirements. We will not describe programming details, such as the descriptor implementations, but we will show the framework and component re-use by an "evolutionary" design. The first requirement is that it shows an image on a window, and it extracts image features. The second requirement is to extract and store image features in a batch. The rest of the functionalities are needed for descriptor matching. We implement three matching schemes. They range from the simple single-descriptor matching to the more sophisticated multi-step matching.

5.1. Extraction-related Operations

Feature extraction is often the first step in a typical MPEG-7 scenario. In addition to the extracting operations, a research friendly program should provide the following extra functionalities:

- A user can select a desired image through a file chooser or an URL input dialog.
- A user can display an image using the specified image viewer.
- A user can extract a specified descriptor from an image.
- A user can display a descriptor by a specified viewer.
- A user can save an image and its associated descriptor for future use.

According to the requirements, we design the classes needed in our program to perform the extraction related operations. Many of these classes can be found in the testbed framework and in the Java class library; therefore, we only need to implement the graphical user interface (GUI) related and the I/O-related classes.

Sometimes we want to process a number of images by a single command, the so-called batch processing. It is convenient to have such a functionality to help users extract features of images. These extracted features can be saved for future processing. In this example program, we assume all images reside in the local file-system. Hence, additional requirements are listed below:

- Allow user to select image files for processing.
- Allow user to select descriptors for extraction.
- Allow user to (re-)build or update (according to the timestamp) the existing descriptors.

To implement the above operations, we need a few directory and file manipulation functions. Fortunately, Java provides them in the java.io.File class. We may create, check existence, and check timestamp by using these functions. Fig. 7 shows the result of our implementation. In Fig. 7(a), an image is loaded and its histogram is extracted. A two-file batch extraction process is shown in Fig. 7(b).

5.2. A Single-descriptor Match

The single-descriptor matching is a very simple matching method. We build a user interface to demonstrate this kind of search. The use of this functionality is similar to that in a typical MPEG-7 scenario. Since search techniques are not closely related to the framework flexibility, we implement a brute-force search algorithm. Reusing the CmdBase and CmdStatusListener introduced in Sect. 5.1, we design and implement the GUI and the command objects. The command extracts the feature from the query image, searches for similar images in the storage, and returns a list of matched results. The image files are displayed as 64×64 icons on the user interface. Fig. 8(a) is the implementation of the single-descriptor match function. The screenshot shows that a user selects the color layout descriptor as the feature, and chooses the default extraction and matching algorithms to perform the operations. The maximum number of returned (matched) images is set to 20. The sample image database contains 797 images. The returned image icons are displayed in the window from left to right and from top to bottom with the most similar one on the top-left corner. When an icon is clicked, its original image is displayed in a temporary image viewer. To speedup icon rendering, we implement a simple icon cache. The icon is generated and stored in the cache when the image icon is first-time listed on the query result. The same icon is loaded directly from the cache, if it is called again.

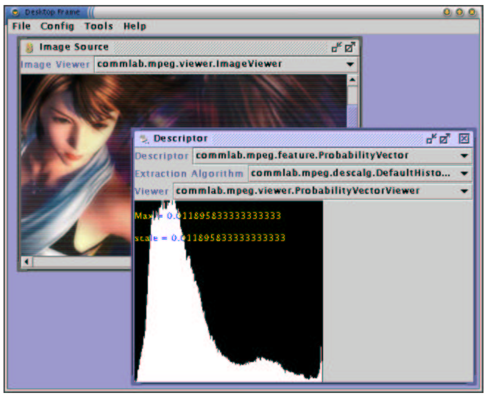
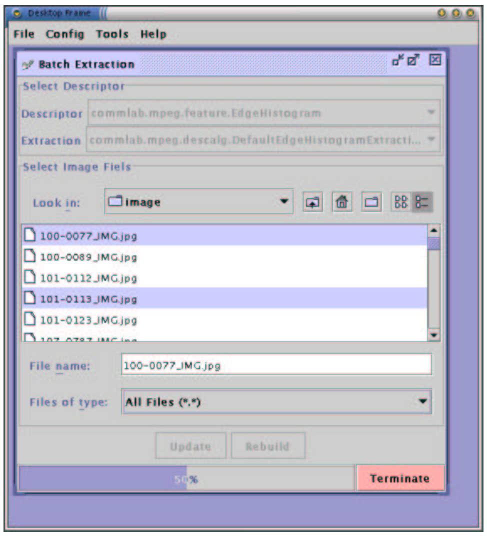


Image Viewer: commlab.mpeg.viewer.ImageViewer
Descriptor: commlab.mpeg.feature.ProbabilityVector
Extraction Algorithm: commlab.mpeg.descalg.DefaultHisto...
Viewer: commlab.mpeg.viewer.ProbabilityVectorViewer
Max = 0.011895833333333333
scale = 0.011895833333333333

(a) Loading and extraction



Descriptor: commlab.mpeg.feature.EdgeHistogram
Extraction: commlab.mpeg.descalg.DefaultEdgeHistogramExtracti...
Look in: image
100-0077_JMG.jpg
100-0089_JMG.jpg
101-0112_JMG.jpg
101-0113_JMG.jpg
101-0123_JMG.jpg
File name: 100-0077_JMG.jpg
Files of type: All Files (*.*)
50%

(b) Batch extraction

Figure 7. User interface for extraction-related operations

5.3. Weighted Match

A second match, called weighted matching, is implemented and discussed in this section. The single-descriptor matching often does not produce satisfactory results. The returned images are not subjectively similar to the query item from time to time. A descriptor represents one specific aspect of an image. The single-descriptor matching is not sufficient to imitate the human semantic matching. One improvement is considering a weighted distance sum of several descriptors in matching.

The basic requirements of this functionality are similar to those of the single-descriptor matching. The descriptor selections and matching commands are now extended to multiple-descriptor selections. We also need to include the weighting factors that are associated with the descriptors. We use a table to select descriptors and set their properties, such as algorithms and the weighting factors. Fig. 8(b) shows the user interface of the weighted match parameter input table.

5.4. Multi-step Match

The matching methods described in Sect. 5.2 and 5.3 are one-step matching; that is, the similarity metrics are calculated in one-shot. We obtain the searching results right after we scan through the entire image database. Although the weighted matching method often performs better than the single-descriptor matching method, one-shot scheme has a rather limited performance. This can be further improved by the multi-step schemes.

We can set the priorities of multiple descriptors using sequential steps. Each step produces a subset from the previous searching results. For example, we may first use the edge histogram to find, say, 100 similar images from the database. Then, we search by color layout in this list to produce the final 20 matches. This method can be considered as a chain of weighted matching processes.

One of the implementation issue is that we do not restrict the re-use of the same query descriptor in different steps. We have to modify the WeightedMatchCmd to prevent re-extract an extracted descriptor. This can be easily solved by adding a collection to hold all extracted descriptors. Before we extract the query descriptor, we fetch it in the collection. If there is an extracted version in the cache, we skip the extraction and continue the remaining process. If the descriptor has not been extracted, we perform the extraction and put the descriptor into the cache. Fig. 8(c) is a screenshot of the multi-step search. A user can input the step parameters in the left and the upper-right panel. For each step, the central part of the right panel is the same as the weighted matching parameter panel. The bottom part of the right panel displays the progress of both the global and the current processing.

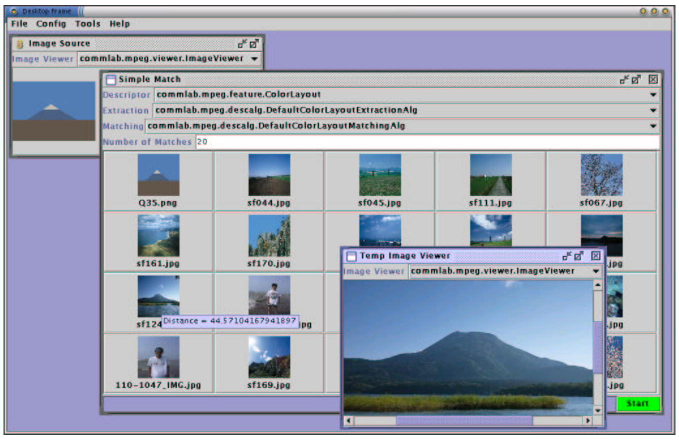


Image Viewer: commlab.mpeg.viewer.ImageViewer
Descriptor: commlab.mpeg.feature.ColorLayout
Extraction: commlab.mpeg.descalg.DefaultColorLayoutExtractionAlg
Matching: commlab.mpeg.descalg.DefaultColorLayoutMatchingAlg
Number of Matches: 20
Distance = 44.57104167941897

(a) Single-descriptor matching

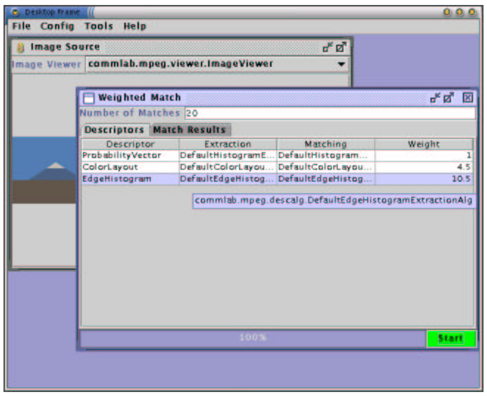


Image Viewer: commlab.mpeg.viewer.ImageViewer
Number of Matches: 20

Descriptor	Extraction	Matching	Weight
ProbabilityVector	DefaultHistogramE...	DefaultHistogram...	1
ColorLayout	DefaultColorLayou...	DefaultColorLayou...	4.5
EdgeHistogram	DefaultEdgeHistog...	DefaultEdgeHistog...	10.5

commlab.mpeg.descalg.DefaultEdgeHistogramExtractionAlg

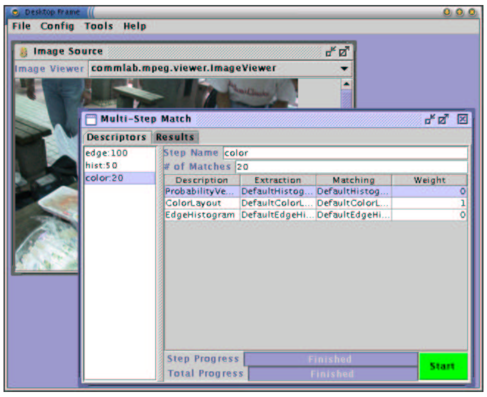


Image Viewer: commlab.mpeg.viewer.ImageViewer

edge:100
hist:50
color:20

Step Name: color
# of Matches: 20

Description	Extraction	Matching	Weight
ProbabilityVe...	DefaultHistog...	DefaultHistog...	0
ColorLayout	DefaultColorL...	DefaultColorL...	1
EdgeHistogram	DefaultEdgeHi...	DefaultEdgeHi...	0

Step Progress: Finished
Total Progress: Finished

(b) Weighted matching (c) Multi-step matching

Figure 8. Screenshots of matching related operations

6. CONCLUSIONS

In this paper, we designed a research friendly software testbed based on the concepts of MPEG-7. The system requirements come from an MPEG-7 typical scenario, plus the functionalities a researcher may need. The testbed has a framework that provides user programming interfaces and an abstraction of all MPEG-7 defined operations. We believe the viewer-data-algorithm relationship covers most of the research requirements. Though the testbed is designed for researchers, it can also be used as the base for developing a real MPEG-7 application. The application designer may benefit from the additional functionalities offered by the framework in this testbed.

For example, an image search engine can use the viewers as the renderers for browsing, and a distributed search engine may take the advantage of the Java platform to eliminate the cross-platform issue.

Treating the descriptors as a type of media data makes it possible to re-use the codec interfaces without messing up the architecture with redundant classes. The persistence and component management units hide the tedious database and configuration jobs, thus the users can focus on the descriptor design and application prototyping. With a few implemented concrete components, we show the flexibility of this testbed by an example, which contains three types of matching schemes. This example not only demonstrates the feasibility of developing applications on our platform, but also shows the extensibility and re-usability when the functionalities are incrementally added.

There are a few issues for further improving this testbed. Our system is built on the Java platform and it uses JDBC as the database connection method. Our experiences indicate that the binary object accesses are relatively slow comparing to the text-based accesses. The performance may be bounded by the JDBC driver. If we want to speed up the search, a search algorithm targeting at a particular application should be carefully designed. Java provides many network related functionalities such as socket, URL, and RMI.[15] It is an interesting topic to extend our testbed across the network. Java has several media extensions, such as JMF[16] and JAI.[17] They are useful extensions for handling media streams and images. We may enrich our testbed implementation by incorporating these extensions. Solutions to these issues are expected to be added into the next version.

REFERENCES

1. J. D. Gibson, *Multimedia Communications — Directions and Innovations*, Academic Press, San Diego, CA, 2001.
2. A. Puri and T. Chen, *Multimedia Systems, Standards, and Networks*, Marcel Dekker, New York, NY, 2000.
3. *Overview of the MPEG-7 Standard (version 5.0)*, *ISO/IEC JTC1/SC29/WG11 N4031*, MPEG7 Committee, Mar. 2001.
4. F.-C. Chang and H.-M. Hang, “An introduction to mpeg-7,” in *Computer Vision, Graphics, and Image Processing Conference*, pp. 50–57, (Taipei), Aug. 2000.
5. http://www.cselt.it/mpeg/, “Mpeg homepage.”
6. *Multimedia Content Description Interface - Part 2: Description Definition Language*, *ISO/IEC JTC1/SC29/WG11, FDIS N4288*, MPEG Committee, Jul. 2001.
7. *Multimedia Content Description Interface - Part 6: Reference Software*, *ISO/IEC JTC1/SC29/WG11, FCD N4206*, MPEG Committee, Jul. 2001.
8. Q. Huang, A. Puri, and Z. Liu, “Multimedia search and retrieval: New concepts, system implementation, and application,” *IEEE Trans. Circuits Syst. Video Technol.* **10**, pp. 679–692, Aug. 2000.
9. G. Booch, J. Rumbaugh, and I. Jacobson, *The Unified Modeling Language User Guide*, Addison Wesley Longman, Inc., 1999.
10. http://www.uml.org/, “Uml home page.”
11. http://www.omg.org/, “Omg home.”
12. http://java.sun.com/products/jdbc/, “Jdbc.”
13. G. O'Sullivan, *Java 2 Complete*, SYBEX Inc., 1999.
14. E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns*, Addison Wesley Longman, Inc., 1995.
15. http://java.sun.com/products/jdk/rmi/, “Remote method invocation.”
16. http://java.sun.com/products/java media/jmf/, “Java media framework.”
17. http://java.sun.com/products/java media/jai/, “Java advanced imaging.”