國立交通大學

工學院聲音與音樂創意科技碩士學位學程

碩 士 論 文

基於 IEEE 1599 標準的自動作曲系統架構

An Architecture Of Automated Composition System Based On IEEE 1599

研 究 生：鄭中皓

指導教授：黃志方　教授

成維華　教授

中 華 民 國 一〇二 年 六 月

基於 IEEE 1599 標準的自動作曲系統架構

An Architecture Of Automated Composition System Based On IEEE 1599

研 究 生：鄭中皓　　Student：Chung-Hao Cheng

指導教授：黃志方　　Advisor：Chih- Fang Huang

成維華　　Wei-Hua Chieng

國 立 交 通 大 學

工學院聲音與音樂創意科技碩士學位學程

碩 士 論 文

A Thesis

Submitted to Master Program of Sound and Music Innovative Technologies

College of Engineering

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of Master

in

Engineering

June 2013

Hsinchu, Taiwan, Republic of China

中華民國一〇二年六月

基於 IEEE 1599 標準的自動作曲系統架構

學生：鄭中皓　　指導教授：黃志方

成維華

國立交通大學工學院聲音與音樂創意科技碩士學位學程

摘　　要

本論文之目的在於建構一套基於具有整合各種異質音樂資訊功能的自動作曲系統架構，進一步輔助作曲者完成符合音樂理論的音樂作曲。歌曲作品除了聲音訊號之外，也包含各種不同格式的異質媒體資料，例如 MP3 音樂檔案、MIDI 音樂合成檔、電子樂譜檔案、原始創作手稿、以及各種音樂描述註解檔。在資訊被蒐集越廣泛的情況下，一首作品的深層內涵及創作元素較容易被被萃取出來，其原始作曲規則更能被探勘量化，自動作曲的作品將至臻完善。目前傳統單一標準的媒體儲存格式已明顯不敷所求，因此本文提出一種基於 IEEE 1599 標準的電腦自動作曲系統架構，透過整合所有電腦自動作曲所需之元素，提供音樂研究者電腦自動作曲及其過程中所需之音樂內涵式分析一套完整的解決方案。

關鍵字：IEEE 1599、自動作曲、異質資訊整合

An Architecture Of Automated Composition System Based On IEEE 1599

Student：Chung-Hao Cheng　　Advisor：Chih- Fang Huang

Wei-Hua Chieng

Master Program of Sound and Music Innovative Technologies

National Chiao Tung University

Abstract

The purpose of this study is to build a basic framework of automatic composition system which integrates different forms of music information, and to further assist composers with the ability to accomplish works which meet the standard of musical theory. Besides wave files, the works also include multimedia files in different forms. For instance, MP3 files, MIDI files, Electronic score files, original manuscripts, and other kinds of content descriptors. With more musical elements collected, the deeper meanings and composing factors will be easily extracted, thus enable the system to engender better music. Nowadays, the single format of media storage obviously fails to fulfill the needs of the users. Therefore, a basic framework of automatic composition system based on the standard of IEEE 1599 has been proposed, providing researchers with a complete solution to computer based automatic composition and music information analysis.

Key words: IEEE 1599, automated composition, heterogeneous information Integration

誌謝

在碩士兩年期間當中，首先要感謝黃志方老師的教導，使我們具有電腦音樂以及自動作曲領域的專業知識，以及各方面所給我們的許多教導與建議。另外要感謝的是，大學指導我專題的劉志俊老師，使我在大學時期開始具有閱讀論文、撰寫論文、發表論文的經驗，使我的碩士生涯更為順遂。除此之外，也感謝所有教導過我的老師，您們所傳授的各項知識都使我獲益良多，使我在研究的路途當中更為順利。

感謝 JeffLab 與學程的每一位同學、學長姊與學弟妹們，特別謝謝可柔教導了我許多與音樂有關的知識，使我更理解樂理，讓我在系統實作上更有方向。謝謝育賢學長教導我有關自動作曲的知識，使我在系統設計上更為完整。謝謝耿萍、哲宜、捲捲、魏崴在口試前協助我將 MIDI 檔案套上好的鼓組及音源，使我在口試當天 DEMO 的自動作曲音樂更為動聽。謝謝偉剛、小麥、見臺在口試當天幫我出去領取口試餐點，讓我能夠更專注在準備簡報。謝謝蘇苑，在口試前跟我一起整理 339 教室的環境與擺放餐點，讓我能夠以比較輕鬆和從容的心情準備口試。感謝偉剛、沂俊、依玲、可柔、姿慧、小涼、婉珍，在研究所階段能夠碰到你們讓我感到非常幸運，無論是一起玩耍或是一起努力都非常開心。除此之外，也感謝所有我這兩年來認識的同學，無論是一起修課或是做研究，都讓我學習到很多。

感謝家人與朋友們的支持，因為有你們的體諒及幫忙，使我的碩士生活有很不一樣的光景。在推甄之前，家人告知我有這個學程，才使我今天有機會在交大聲音學程完成研究所的學業。感謝所有朋友在這段時間的陪伴，讓我有豐富的生活，對於所有幫助過我、關懷過我的人，致上由衷感謝。

最後，謹以此文獻給我摯愛的父母及所有關心我的人。

目錄

圖目錄

表目錄

第一章、緒論

隨著數位時代的興起，電腦輔助設計軟體(Computer-aided design)應用範圍越來越廣，各種與音樂相關的新興軟硬體推陳出新，過去作曲工作完全是由作曲家來進行，但是目前電腦已經有能力幫人處理部分作曲的工作，甚至是憑空創造出音樂。倘若電腦程式具有產生出與人類作曲家品質相似的成果，其方法無論在學術或是商業應用領域均具有極大價值。例如，當作曲家靈感枯竭時，能夠透過電腦程式創造出的大量音樂樣本中找尋創作靈感。電腦遊戲中的固定場景配樂亦能夠透過此項技術，依據當下場景產生出相同風格但是內容相異的音樂，以增加遊戲配樂的豐富性。因此，我們可以將隱含在音樂當中的特徵萃取出來成為規則，或是將已知的音樂創作原則設計成演算法供程式自動產生音樂，以達到使電腦程式能夠產生出與我們期望相符的音樂。

電腦自動作曲概念約西元前 500 年即被提出[1]，當時 Pythagoras 相信音樂與數學並非完全分離的知識。Hiller 和 Isaacson 可能是最早使用一個基於 Markov Chains 的亂數產生器的可計算模型(Computational model)來進行電腦自動作曲[2]。根據[1]所轉述於 1993 年於 ICMC 研討會之 Panel Discussion 當中，David Cope 將電腦自動作曲(Algorithm Composition)定義為「在有限步驟內使用一序列的規則解決(或完成)某一個音樂部份或整體的問題」。

電腦學習音樂規則並產生音樂的方式與亦與人類相似，而人類學習事物的方法可分為兩種，內顯學習法係指無意識獲得刺激環境知識的過程，而外顯學習為採取一定策略來完成學習活動。因此，電腦學習音樂而得到之規則由過去研究可以得知分為兩種：內顯規則(implicit rule)與外顯規則(explicit rule)。

圖 1　內顯式規則作曲法

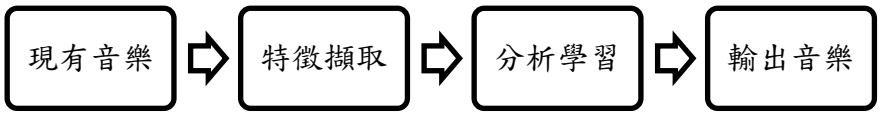


圖 2　內顯式規則作曲法

在自動作曲的過程中，內顯規則法需要其他音樂來學習規則，產生的音樂風格會與給定的音樂相似；外顯規則需要使用者指定各種音樂參數來產生與其參數相符的音樂。在自動作曲的過程中，系統可以透過分析音樂或由使用者指定規則以產生使用者期望的音樂。但是，目前在使用者與自動作曲系統的互動過程中，尚無一套系統來整合各種音樂異質資訊與自動作曲工作，並能夠滿足在特徵擷取、分析、學習、產生、評估、發佈、評價等各流程的完整解決方案。

由於各種異質數位多媒體資料特性的不同，且與傳統文字資料相異，較難以透過文字描述儲存在傳統資料庫中來達到有效查詢。MPEG 組織為解決此一問題，遂制定 MPEG-7 標準。針對多媒體的基本結構，提供了內容描述(Content Description)、導覽與存取(Content Management)、內容組織(Content Organization)、使用者互動(User Interaction)等功能。MPEG-7 以 XML 語言為基礎，並再加入其延伸格式，作為影音(AV)內容的描述結構(Multimedia Description Schemes, MDS)。用這種方式描述影音內容，其產生的格式能夠讓人們能夠了解所要呈現的內容，且使影音內容夠容易被搜尋、索引、過濾、使用、管理[3-7]。

在音樂內涵式分析與電腦自動作曲的過程當中，甚至是到未來自動編曲、自動伴奏等延伸程序，無論是在分析音樂、修正參數、產生音樂，若能夠整合大量的異質資訊供電腦分析與使用者參考之用，其作曲結果將更臻完美。惟 MPEG-7 係作為影音內容描述之用，並非針對紀錄音樂而專門設計，且音樂資料的豐富度遠大於一般聲音，若以 MPEG-7 為基礎建構一套自動作曲系統，其能記載之資訊於音樂豐富度上有所不足，故我們結合了 IEEE 1599[8-11]作為描述、整合、同步各種異質音樂資訊之用。

IEEE 1599 標準依照音樂的不同特性分成六層架構,其描述檔案以 XML 格式記載，具有容易透過網路交換、適合人類閱讀、無相容性問題、具有跨平台特性等特點。雖然 IEEE 1599 標準於邏輯層亦提供樂譜記載的功能，但目前就電子樂譜格式而言，最為通用的格式為 MusicXML，目前市面上超過 160 種商業製譜軟體均支援其讀取與寫入，故我們採用 MusicXML 記載自動作曲的輸出樂譜。

綜合以上所述,本文為整合自動作曲過程中各項研究所需,提出一種基於 IEEE 1599 標準的電腦自動作曲系統架構，使未來自動作曲演算法及其成果的研發與評估，以及音樂元素對於人類心理感知的影響，在流程所產生的所有描述資訊，能夠更有架構與有內涵意義的被記載。另一方面，自動作曲的成果亦能夠更廣泛的展開實驗性應用，品質能夠更快速的被各種方式評估，使自動作曲演算法及其產生能夠有效的被檢視與改善，最終能夠產生與人類作曲品質相近的音樂。

本篇論文的第二章將介紹相關背景知識以及相關的文獻回顧；在第三章中，我們將說明系統分析與設計方法；第四章將展示結果與系統功能範例分析；最後，在第五章總結成果，以及討論未來研究可能的方向。

第二章、相關研究

2.1 研究背景

作曲家將自身的各種體驗與情緒轉化成靈感，並利用自身的才能在作曲的過程將各種靈感轉化成元素置入，過程中歷經反覆修正，最後建構成完整的音樂，並期盼聆聽者能夠了解創作者的最真實與最深層的情緒。聆聽音樂的同時，人會因為音樂而產生幻想，進一步產生聯想，最後專屬於聆聽者的意境會被創造出來，並受到聆聽者自身之生活經驗與價值觀影響。由於電腦自動作曲具有商業潛力，應用範疇相當廣，故非相當新穎的概念，透過各種演算法模擬作曲家而產生音樂在過去已累積相當多的研究成果[1]，並成為一個獨立的研究領域。

近年來因為內涵式音樂檢索與分析(music information retrieval and analysis)在學術研究上成為相當重要的議題，主要是從音樂中找出各種特徵，例如旋律(melody)、節奏(rhythm)、和弦(chord)等資訊，並加以分析及歸類。完整的音樂除了最基本的訊號資訊以外，還包含各式各樣的異質音樂資訊，例如創作手稿、電子樂譜檔案(如 MusicXML、製譜軟體 Finale 與 Overture 格式的樂譜檔案)、歌詞、其他音樂內涵描述註解等(如 ID-3 tag)，在未來均有可能作為音樂資訊檢索分析之元素。

由於研究發展相當快速，從低階的訊號資訊到高階的語意資訊之檢索分析皆有發展，其分析結果更可做為音樂自動推薦、分類、辨識、分段、索引以及自動作曲之用。若各種音樂異質音樂資料整合的越完整，歌曲就能被分析的越透徹，所歸納出來的規則也會更為完整，對於創造出更多音樂片段、刺激作曲者靈感，對於自動作曲宗旨目標與發展更為有利。基於上述理由，我們設計一個基於 IEEE 1599 標準的自動作曲系統架構，以協助一般使用者或是研究者可以藉由整合廣泛的異質音樂資訊，分析後學習特徵並作為自動作曲的內顯規則法之用，或是藉由整合描述檔案內的結構描述資訊供外顯規則法作曲，使自動作曲之分析、產生、結果評估等流程較流暢，並達到一個無論是在學術或商業應用方面均極有價值之整體解決方案。

2.2 音樂結構分析

在 1992 年由 Hammer 與 Cole 所發表的音樂特徵分析著作[12]可得知，音樂特徵可分為四大類，分別為時值、音高、音質、強度。上述四個部分亦可再細分出其他分支，詳細部分如圖 3 所示。

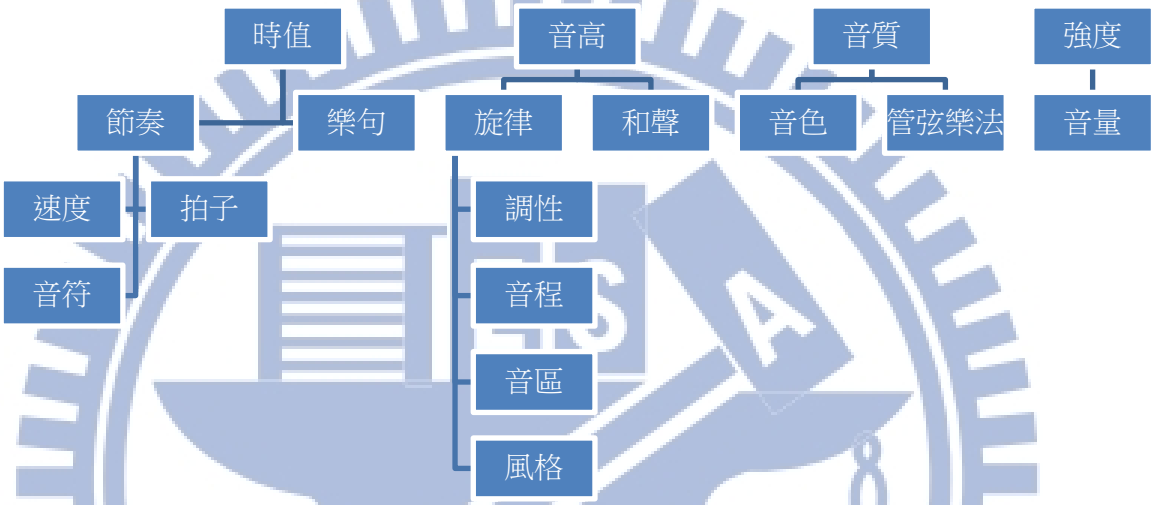


圖 3　音樂特徵分類

在眾多音樂特徵當中，樂句(phrases)是組成曲調的主要單位，如同文章中的一句話。由於能夠透過樂句分析出音樂中的結構，故樂句是音樂分析的基本單位。因此，偵測一首歌曲中的樂句分段位置，是音樂索引、音樂內涵查詢、音樂曲式自動分類等研究所需的基本核心技術。

根據 1995 年由 Smith 提出的論文[13]得知，作曲家在作曲的時候通常會先有動機(motif)，再由動機發展出主題及主旋律，最終發展成完整的音樂。同時，上述作曲過程中會使音樂具有結構性，故分析音樂結構及分段的相關研究則開始發展[14]，並透過分段來分析音樂的結構，目的為找出結構性的音樂單位[15-17]。除此之外，Rossing 於[18, 19]指出，當音樂中的動機改變的時候，若音樂結構的關係已被聆聽者記住，即可被辨識出來。

倘若將獨立音符以時間結構為關係組成起來，即成為音樂的分段，其主要作法是將音樂主旋律找到多組分段點後，即可把主旋律分成許多片段，以找出較高階的音樂結構，例如對比與和聲[15]。另一方面，若不納入高階的音樂資訊作為考量的話，亦可以使用音高、音長資訊來判定是否發生分段[17, 20]。

在[21, 22] 論文當中，Cambouropoulos 提出一種稱為區域邊界偵測模型 (LBDM, local boundary detection model) 的方法來偵測樂句的分段點。LBDM 方法係根據音高、起音點間距、以及休止符的變化評估樂句分段點發生的可能。

Cheng 與 Chew 在[23, 24]等論文中提出一種稱為區域最大樂句偵測 (LMPD, local maximum phrase detection) 的樂句分段方法。LMPD 方法的理論根據是以往在樂句特徵方面的許多研究結果指出，一個樂句的力度變化大致上會呈現出由漸強 (crescendo) 轉為漸弱 (decrescendo)的拱門結構，兩個拱門交界處即是樂句邊界。利用此特性，Cheng 等人定義了樂句強度 (phrase strength)、樂句穩定度 (phrase volatility) 與樂句典型性 (phrase typicality) 等衡量公式，來表示某一演奏家在樂句方面的詮釋特性。

傳統的樂句分段係以心理學角度來探討音樂群組(musical grouping)之關係。1920 年時在德國有心理學家們提出物體群組的相似理論，稱為完形學派。到了 1923 年，成為完形理論[25]。完形理論提出了三項基本原則，是有關於在心理層面對於音樂群組的認知：

1. 鄰近原則(principle of proximity): 傾向把時間(time)與空間(space)上彼此之間非常相近的事物組合成同一群組。
2. 相似原則(principle of similarity): 傾向於把相似的事物組結合為同一群組。
3. 連續原則(principle of continuation): 指傾向於把形成連續圖案的事物組合為一個群組，即便為中空的圖形排列，只要受到彼此連續的符號或圖形排列而成整體圖樣的刺激，很容易視為組合成的一種整體圖樣。

Temperley 在[26]一書中詳細探討有關樂音群組規則在感知心理學與音樂理論的根據與發展沿革，提出一套樂句結構傾向規則(PSPR, phrase structure preference rules)，藉由考慮樂音群組間隙(gap)、樂句長度、以及韻律對句(metrical parallelism)來計算樂句分

段點的可能性。

申克(Schenker)於 1935 所出版的 Der freie Satz(自由作曲)[27]中提出的音樂分析理論為本世紀最重要的音樂理論之一，而該本書於 1997 年被翻譯為簡體中文出版[28]。申克理論認為，由一首音樂作品的前景，找出隱藏在音樂表面背後的中景與背景結構。申克理論雖然受到廣泛的支持，但不容易轉換為一組明確的規則，並進而以一組程式來實作，是申克理論電腦程式化的主要困難之處。除此之外，完形學派的心理學家們雖然有提到可將完形理論應用在音樂層面，但並未真正將完形理論系統化，使之能夠套用在音樂理論上。

有鑑於此，Lerdahl 與 Jackendoff 在 1983 年提出調性音樂生成理論 (GTTM, Generative Theory of Tonal Music)[29]，提出的調性音樂生成理論中的群組分析(grouping analysis)，針對調性音樂於人類直覺上的意義做具體描述。GTTM 主要用於從音樂的表面形式，找出隱含於內的結構。GTTM 能夠分析的音樂為主音音樂，即由一個主旋律與外加伴奏所組成。

在眾多理論當中，GTTM 是目前較具體的音樂結構分析理論，具有相當程度的參考價值。它主要將音樂拆成以群組的方式來分析，即將主音音樂拆解成一群一群的小單位，並且具有階層的關係，所有的群組都必須符合被上層群組階層包含的關係。

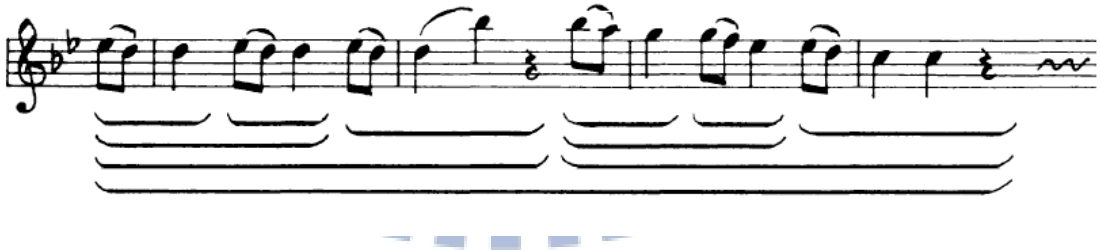


圖 4　　主音音樂群組關係

GTTM 主要利用主音音樂在演奏或演唱的時候，需要換氣的部分來作為分段的依據，而這些分界點通常都是音長較長、休止符出現或是音高起伏的部分。群組分析是由 7 個群組傾向法則(Grouping Preference Rules, GPRs)所構成，此 7 個法則整理如下表：

GPR1	樂句長度	避免分析非常短的群組。群組長度愈短則群組化傾向程度愈低。
GPR2	鄰近原則	假設有一個由 4 個音符 n1n2n3n4 所形成的序列，在其他條件均相同的狀況下，如果下列條件滿足，則在 n1-n2 之間可能為群組邊界：
GPR2a	圓滑線/休止符	音符 n2 的結束時間與音符 n3 的開始時間之間隔大於音符 n1 的結束時間與音符 n2 的開始時間之間隔；也大於音符 n3 的結束時間與音符 n4 的開始時間之間隔。
GPR2b	起音點	音符 n2 與音符 n3 的起音點間距大於音符 n1 與音符 n2 的起音點間距；亦大於音符 n3 與音符 n4 的起音點間距。
GPR3	變化	假設有一個由 4 個音符 n1n2n3n4 所形成的序列，在其他條件均相同的狀況下，如果下列條件滿足，則在 n1-n2 之間可能為群組邊界：
GPR3a	音程變化	n2-n3 音符轉換之間的音程距離變化大於 n1-n2 以及 n3-n4 之間的音程距離變化。
GPR3b	力度變化	n2-n3 音符轉換之間的力度變化大於 n1-n2 以及 n3-n4 之間的力度變化。
GPR3c	運音變化	n2-n3 音符轉換之間的運音方式變化大於 n1-n2 以及 n3-n4 之間的運音方式變化。
GPR3d	音長變化	n2 與 n3 的音長不同；且 n1 與 n2 的音長相同，n3 與 n4 的音長相同。
GPR4	階層性	當 GPR2 與 GPR3 出現相對而言更強的地方，可以建立更高一階層的群組邊界。
GPR5	對稱性	傾向使用將一個群組切分成兩個相同長度的子分割之群組分析方法。
GPR6	平行對句	如果有兩個或多個音樂片段可以構成平行對句時，傾向於將這些

		音樂片段建立平行的群組。
GPR7	時間展開與延長	傾向於使群組結構可以產生較穩定的時間展開與延長約化分析的結果。

表 1　　GTTM 的群組傾向法則

2.3 電腦自動作曲演算法

演算法作曲(Algorithmic Composition)，或稱自動作曲(Automated composition)指利用數學或邏輯方式產生音樂，使人在利用電腦進行音樂創作時介入程度最小的研究[30]。

最早在西元 11 世紀的時候，Guido d'Arezzo 的傳教士為了宗教目的，將 8 世紀的一首聖約翰讚美詩 (Hymn to St. John)的每行詞句第一音節作為音階的唱名以對應音名，以便記錄音高的模型[31]。此時便有了樂譜及記錄音樂的記號，開啟了人類研究音樂的先驅。20 世紀初期，自動作曲方式開始進入發展階段，Arnold Schoenberg 引入了 12 音列概念[32]，然後由 Anton Webern 與其後繼者進一步的發展。在此階段技術中，音樂被特徵化，提取出如音高、音長、音色等參數並加以發展控制，都對現今的電腦演算作曲造成了影響。

早期的自動作曲程式為由使用者輸入相關參數，即由電腦程式自動產生、修改、選擇三個程序以產生樂譜。1979 年的時候，Hiller 及 Isaacson 將部分十六世紀的對位法及現代的十二音列法理論輸入電腦中，並將不和諧的方式排除，從各種可能中任意排列組合，最後完成 Illiac 的組曲[33]。直到後期，在 1996 年開始 Cope 則開始使用自動化分析技術來重組給定的資料庫中的音樂片段，產生新的音樂，以及利用自動分析方法來模擬風格創作，並使用人工智慧演算法來加強音樂演算法[34, 35]。

電腦自動作曲方法可依據是否需要現有音樂來分析，將方法分為「內顯規則法」與「外顯規則法」兩種[36]。內顯規則法係參照樂理規則或是使用者提供的參數，由程式來產生樂譜的過程。外顯規則法為使用者提供音樂歌曲，程式利用機器學習技術分析其特徵並產生規則，並透過其結果完成作曲。內顯式與外顯式作曲法兩者最大差別在於有

無訓練資料。除了上述的二分法之外，Papadopoulos 及 Wiggins 在 1999 年所發表的論文將過去自動化產生音樂之方式進行分類，將電腦自動作曲方式分成六大類，分別為數學模型(Mathematical models)、知識庫系統(Knowledge based system)、語法(Grammars)、演化法(Evolutionary methods)、學習系統(System which learns)、混合系統(Hybrid System)等六種[1]，其說明如下列段落。

「數學模型」法包含使用隨機過程(Stochastic Process)[37]、馬可夫鏈(Markov Chain)[38, 39]或是其他計算性模型模擬音樂創作過程。2001 年時 Miranda 於[40]提出，可以利用機率分佈與迭代演算法來產生音符，完成公式化音樂。隨機過程指所有隨機變數(Random Variable)所形成之集合，定義 T 為一個非負整數之集合，並在一個有下標之隨機變數的集合$\{x_t\}$，其中 t 在給定之集合 T 之內演進，x_t代表在時間 t 時一種能夠衡量的特性。若在時間 t+1 時之狀態與 t 時狀態相關，並與過去時間 t-1, ... , 0 狀態無關，則稱為此隨機過程為馬可夫練。Schulze 等人於 2011 年透過讀取 MusicXML 檔案，移除拍號轉變並將音樂轉調(Transpose)為 C 大調後，透過分析以取得每個音符之間的狀態改變(音高及音長)，最後計算出和弦進行(Chord Progression)、旋律線條(Melodic Curve)、終止式(Cadence)之混合階層馬可夫模型(Mixed-order Markov Model)，依據其模型即可產生出新的音樂創作[39]。作者將同樣風格的人工創作與透過馬可夫模型而產生的電腦創作透過問卷調查，大部分受測者仍較偏好人工創作，但有 72%受測者無法正確區分人工創作與電腦創作。優點為時間複雜度低，可達到即時運算效能。

「知識庫系統」是將知識規則存成資料庫後，即可經由這些知識資料庫中導出某些知識，並可作為「推論」、「診斷」、「預測」、「控制」之用，綜合推理機制提供專業的自動化服務。將上述方式推廣至自動音樂創作上，即為將特定風格的音樂模型化並建立一套規則，依循其規則即可產生出對應風格的音樂。Ebcioğlu 於 1988 年建立的系統，包含約 350 條第一層敘述邏輯型式(First-order predicate calculus)的規則，其規則表示著許多不同的音樂觀點，例如和弦骨架(Chord skeleton)、每一個聲部的旋律線(Melodic lines)、高音與低音之間的宣克單一聲部水平移動(Schenkerian voice leading within descant and bass)，以建立四聲部的巴哈風格的音樂，優點為能夠明確解釋某項規則之合理性[41]。

「語法」法則是將音樂視為一種語言，對音樂的結構組成視為有如語言般的規則來分析與創作。Steedman 首先在 1984 年提出了用在藍調 12 小節之和弦級數的生成語法[42]。1996 年的時候，Steedman 亦提出「語法音樂的想法，可能跟語法本身的想法一樣古老」，並指出音樂形式文法的研究上已落後於描述性的充分性、心理的合理性、計算的實用性，特別是合聲的分析[43]，且透過 categorical grammars[44]改善，能夠在和弦進程上更進一步的模擬人類心理聽覺，達到更好的效果。另一方面，1992 年時 Cope 則是透過了解不同的音樂風格並利用語法方式複製各作曲家的作品風格[45]。

「演化計算」法則為根據生物演化觀念與模擬自然界演化過程所建構成的一種計算模式，演化式策略(evolution strategy)、演化式規劃(evolutionary programming)、基因演算法(genetic algorithms)等等均屬於演化計算範疇。最主要的概念為，在遺傳的過程當中，子代可能會繼承親代的各種特性，但亦可能發生變異而產生與親代不相同的新物種。在這三種方法當中，基因演算法是在各領域中應用最廣泛的。Wiggins 等人於 1998 年透過基因演算法針對使用者所指定旋律產生對應的四部調性主音和聲(four-part homophonic tonal harmony) [46]。作者認為若以審美角度檢視產出結果，缺乏明確的計畫及意象，在結構上無法做到像人類手法般的複雜與微妙。最主要的原因有二點，首先是基因演算法是一種隨機、啟發式的方法，無法保證能夠找到解。第二點為作曲家已發展了許多複雜與微妙解決音樂作曲問題的手法，其手段無法被基因演算法以同樣方式利用。因此，人類在審美角度上能夠接受的音樂，未必能夠透過基因演算法產生出來。

「學習系統」指系統無預先定義規則與限制，而是透過機器學習的方式(例如類神經網路、模糊系統等)，透過自動分析資料以求得規律，再用來對未知的資料進行預測。Todd 於 1989 年提出類神經網路方法透過跳選特定旋律來訓練模型，產生出新的單聲部旋律[47]。除了透過類神經網路學系之外，亦透過使用類似規則的一般化能力，產生出新的旋律且類似於一開始被選擇用來訓練模型之旋律，此方式能夠用來建立出各種不同的音樂特徵，但是結果可能會出現受到同一組音符不斷重複的現象。Mozer 於 1991 年則受到[47]的啟發，採用新式的方法(novel method)來描述音高與音長，並使用連結論(connectionist approach)來產生單聲部旋律，加強改善為一個更完善的系統[48]。

最後一個「混合系統」指混合以上兩個以上的方法學習及建構音樂，通常較單一方法的系統複雜，但可取得較複雜的音樂結果，Gibson 等人於 1991 年混合了基因演算法與類神經網路兩種方式來產生音樂[49]。

2.4 電腦自動作曲結果評估

電腦自動作曲成果產出後，必須有方法以區分出適當與不適當的旋律。但是，將美學編纂成冊或將其以公式描述有其困難之處。因此，評估音樂是否符合音樂美學最常見的做法為請專家進行評估。但是，音樂是時空藝術，被擷取出來被評估的音樂片段在其他的音樂片段當中可能會呈現出不同的感受，如何切割應呈現這種方法評價的合理性和與所顯示出的大小之間的權衡。

目前尚無法完全客觀評價音樂美學的最大的原因，肇因於無法了解審美觀如何表達成為方程式以評估旋律是否符合音樂上的美感。除此之外，每個人對於聆聽音樂所產生的感受極度主觀，縱使不同的聆聽者若聆聽同樣的音樂，亦不會產生同樣的感受[50, 51]。

Cross 於 1998 年在[52]中提到，透過使用不同的科學方法的相關性和實用性理解音樂現象，針對我們所了解的音樂信仰及看法，並對於有關應用科學和音樂的相關性方面，提出對於「音樂」可能的定義。Cross 亦認為，在目前的音樂學研究中，內在主義(immanentist)或解構主義(deconstructionist)的觀點是相當普遍的，在音樂的理解上拒絕了物理科學的可能性，作曲家在作曲過程中並非經度量得知該音樂片段必定會有何作用才使用，而是因為其心理感受及共鳴。除此之外，音樂沒有任何物理上的事實或是軌跡，僅能夠透過音樂學上的思考和寫曲，推斷出人類賦予意義的能力或是人類與物質世界互動的現象的意義，故否定物理科學和音樂方面的關聯。

在內在主義的觀點，科學與音樂理論是無關的，而與心智的意向性以及文化有關。Cross 主張，在音樂的理論和實踐研究的重要性方面，音樂認知心理學目前還在非常初期的發展階段，且涉及到各方面的音樂思想和行為在許多層面上的解釋，必須透過執行理解音樂的認知科學的研究計劃以及理論探討，以跨學科的形式化建模和實證實驗的研

究。因此，Cross 認為在現階段尋找一個完全客觀的評估自動作曲之方法是無用的工作。

綜上所述，在過去的電腦自動作曲相關研究中，評估部分篇幅較少，多半著重於音樂產生之方法。在 Johanson 等人於 1998 年透過基因演算法產生音樂，在結果評估部分僅陳述「幾乎所有產出的音樂都很動聽」，流於主觀且缺乏公信力與度量標準[53]。Unehara 等人的評估方式為改善前述缺點，一共邀請 6 位 20~30 歲年齡層的人，以 1 分至 10 分的標準評斷系統所產生出的音樂，但亦無具體描述客觀評估方式[54]。為改善評估過程過於主觀之問題，即開始有論文採用蒐集演奏家及閱聽人的回饋作為度量之基準[55, 56]。但是 Pearce 等人認為，蒐集大量的主觀判斷以進行綜合判斷，雖能降低單一主觀判斷所造成之問題，但由於每個人之判斷基準與音樂認知標準不盡相同，故此方法仍然無法完全作為客觀評估方式[57]。若要發展一套完全科學客觀地評估標準，以評估旋律或是音樂是否符合藝術美感，則必須找出影響美感的音樂元素。

在 Freitas 等人於 2012 年提出一系列的理念，以提供一個根據多方觀點以進行綜合檢驗跟研究旋律的構想[58]。其中一些想法是基於音樂理論所發展出來的，在過去音樂分析的研究當中較常被列入考慮，但是在自動作曲領域的研究當中較常被忽略。此篇論文將在自動作曲研究過程中較常被忽略的觀點整理出來，以進一步幫助電腦輔助的創作者定義更複雜跟有用的方法來評估音樂.，盼能夠使自動作曲結果的評估更加成熟。作者一共提出了 10 個能夠在分析旋律上有所依據，如表 2 所示。

Pitch (音高)
Tonality and dissonance (調性與不和諧音)
Intervals (音程)
Melodic expectation (旋律期待性)
Contour (旋律線)
Rhythm (節奏)
Patterns (特徵)
Phrases (樂句)
Originality (獨創性)
Second order analyses (二級分析)

表 2　旋律分析觀點[58]

雖然 Freitas 等人於[58]提出了一系列的旋律分析觀點，但是較傾向於分析觀點的理念，無一套完整的分析架構。因此，Pearce 等人提出了一套自動作曲評估架構，盼能較客觀分析自動作曲產出之成果，並對音樂創作的評價進行了討論[57]。這個架構包含了四個階段：

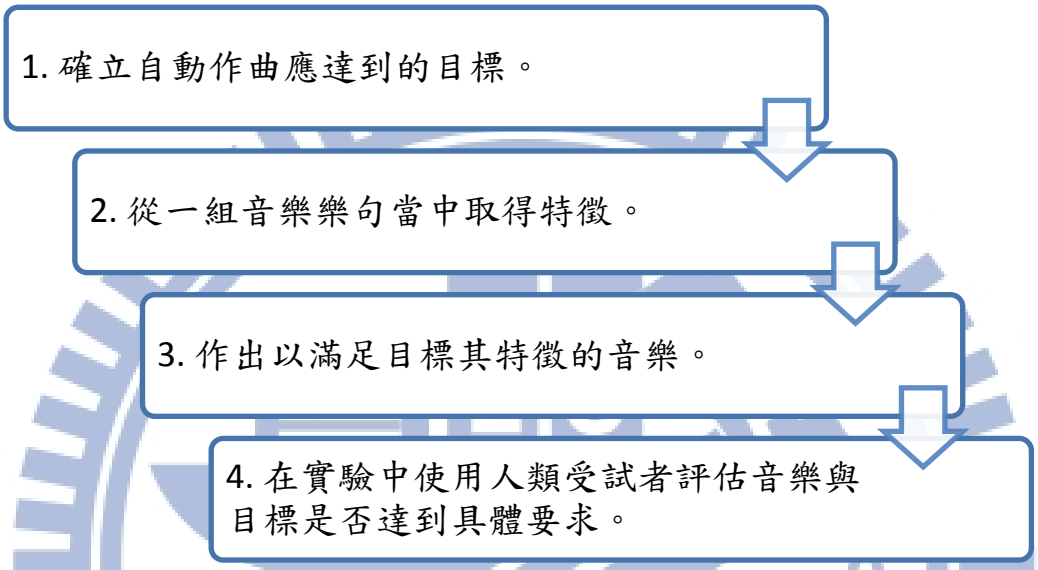


圖 5　自動作曲評估架構[57]

在 Pearce 等人所提出的論文[57]當中，提出了能夠滿足符合自動作曲評估需求的架構，包含確立目標、取得風格特徵、機器創作音樂、評估等四個階段，各階段目的分述如下：

在第一階段的時候，目的為確立目標。在進行自動作曲之前，就必須針對欲得到的結果清楚定義目標，在目標確立例如說產出之音樂必須為特定風格、或是近似於某位特定作曲家之風格，以及能夠容忍的風格差異程度。

在第二階段時，目的為取得特徵規則。必須蒐集與第一階段所確立之目標相符合的音樂，再使用機器學習技術將特徵萃取出來以建立規則。在此階段所使用的特徵擷取技術必須要能夠被量化與計算，否則其產生結果無法評估。採用機器分析音樂以取得規則的最大原因為，若採用知識工程師所建構出的規則，除了會可能會因為其本位主義而得

到較為偏頗且較不客觀的資訊，且會因為來源較少而得到較不多樣化的音樂規則。倘若採用機器分析方式，即可以透過大量蒐集音樂來源，透過各種不同的客觀方法分析得到較豐富的規則。

在第三階段則為依據第二階段產出的規則作曲，並且要能夠滿足其規則限制。最後階段則為評估，將自動化產生的音樂及人類作曲的音樂混合在一起請人以主觀方式判斷能否辨別出是人類或是機器所創作。此方式於該論文中稱為區別測試(Discrimination test)，測試過程近似於圖靈測試(Turing test)。但是，圖靈測試係用於檢測機器是否具有思考能力，在人與機器互動的情況之下，由人判斷出該機器是否展現出與人等價或是無法與人區別的思考能力。但是在此處，僅有機器已產生完成的音樂與人已製作完成的音樂，以隨機方式排序且在不揭露答案的情況之下，供人判斷該首音樂是否為機器所創作，並無涉及圖靈測試中的「互動」，僅單方向接收音樂作出判斷。

在此架構下具有許多特色，由於第二階段的特徵係由人為作曲的音樂當中以數學方式或是其他計算性方式萃取出來，而非由專家制定的規則。因此，整個作曲與評估階段都是在封閉系統(closed system)中運作，無人為的度量標準介入。在評估實驗階段時，會以「人們是否能夠區分出這首音樂是否為機器作曲或人作曲的音樂」取代「這首音樂是否好聽」。倘若受測者無法正確區分出該首音樂為機器所作或人所作，即可以客觀的方式宣稱「該首音樂具有與人為作出的音樂競爭的能力」。

在評估階段計有三項實驗，以各種不同觀點及考量檢定。作者從 Edinburgh 大學的人工智慧研究所找了共 19 名受測者，針對系統與人分別以 GS Roland 909 drum set，節拍 150 BPM 所產生的一小節 MIDI 鼓組音樂進行評估。系統所產生的音樂目標為「聽起來必須像是 drum and bass 風格」、「能夠與人在此種風格下作的音樂相比較」，及「每次系統產生出的音樂必須要有所差異」，各次實驗結果分述如下：

第一個實驗為「區別測試實驗」，受測者必須從訓練集當中區分出該特徵為機器或是人所產生。在測試資料當中，包含了 10 個透過基因演算法所產生的音樂，及 10 個人所產生的音樂。如果受測者無法成功區分出的話，則代表機器所產生的音樂是成功的。實驗結果顯示，在相同風格前提之下，駁斥了「該系統生成的音樂與人類產生的模式並

無區別」觀點。

第二個實驗為樣式評估，由受測者將系統所產生的音樂以樣式進行分類，若人所認定的樣式與原先系統產生時所設定的樣式相同，且分類正確率大於或等於受測者分類人所做的音樂時的正確率，則代表機器所產生的音樂是成功的。實驗結果顯示「該系統所生成的音樂確實具有 D&B 風格」，但受測者卻無法正確辨識並歸類在 D&B 風格，故駁斥了「該系統生成的音樂之樣式能夠被正確分類在 D&B 風格中」之假設。

在第三個實驗當中，為評估系統每一次所產生的音樂之多樣性。實驗結果顯示，系統生成模式未能達到標準水準的明顯變化，駁斥了系統生成的音樂之多樣性大於或等於人類生成音樂的多樣性的論點。

在上述三個實驗當中，結果均顯示所採用的機器創作方式若在同樣的條件與人相競爭，且在未揭露機器與人所創作的作品的情況下，受試者仍然能夠區分出創作者為機器或人，人創作的風格與多樣性亦較機器創作明確與較高品質。因此，透過[57]所提出的評估架構，具有效衡量評估自動作曲是否達到目標需求之能力。

2.5 音樂情緒與心理感知

音樂的調性、速度會使我們在聆聽音樂的時候，會有不同的情緒反應。一般來說，開朗活潑的音樂，速度上大都是稍快的，調性可能是屬於大調。大調較常給人一種快樂而歡騰的感覺，小調則大多為較悲傷或沉悶的情緒。速度輕快的音樂，則較不容易使我們產生沉悶的情緒。

在過去心理學已有針對音樂能夠帶給聆聽者的感受，例如快樂 (happy)、難過 (sad)、生氣 (angry)、平和 (peaceful)等。在眾多音樂元素當中，能夠影響聆聽者的感受的元素包含了速度與調性。在速度的部分，根據 1991 年潘智彪的譯著[59]指出，經由實驗發現，將事先準備好的音樂以每分鐘 60 拍到每分鐘 160 拍不等的速度演奏，受測者的便會因為音樂演奏速度而有截然不同的情緒感受。因此，實驗結果指出，一首緩慢的音樂，給

大部分人的感受不會是激動的，而一首快的音樂給大部分人的感受不會是嚴肅或哀傷的。在調性的部分，根據莊婕筠於 2004 年提出的著作[60]指出，十二的大調與十二個小調可以給人有不同的感覺，其對應的情緒如所示。

調號	大調特性	調號	小調特性
C	莊重的，開朗的	c	陽性的，但有缺點的
♭D	典雅的，澹然的	♯c	優雅悲傷的
D	熱鬧平凡的	d	莊嚴的
♭E	鬆弛的	♯d	鬆弛的，有點悲哀的
E	高貴的	e	平凡、平穩的
F	強力的	f	激烈，有陽剛性的
♭G	優美的	♯f	有銳利度的，但很完美
G	平穩的	g	憂愁卻完美
♭A	柔和高貴的	♯g	悲傷優雅的
A	歡欣的	a	高貴柔和的
♭B	優美但不明的	♯a	鬆弛陽氣的
B	光明高貴的	b	粗野激烈的

表 3　調性及其特性[60]

在情緒的表示，通常以二個維度表示情緒狀態，第一個維度是情緒的正向與負向，第二個維度是該情緒強烈或緩和的程度，分別以 valence 和 arousal 表示[61]。若 valence 為正的，代表是正向情緒，若為負則為負向情緒。arousal 越大則代表該情緒越活躍，反之則為越不活躍。valence 和 arousal 這兩個指標，可以組成一個由平行軸與垂直軸的四象限的二維座標圖，James 將一些音樂情緒種類標記在二維座標圖上，則能夠根據座標角度得知情緒種類[62]。在 Steven 及 Andrew 在 2005 年提出的論文[63]，則整理了過去

音樂情緒的研究結果，歸納出音樂情緒與音樂特徵的關聯性。作者將二維座標圖畫分為八個象限，分類出以下音樂情緒：happy、excited、angry、sad、depressed、dreamy，如圖 6 所示。

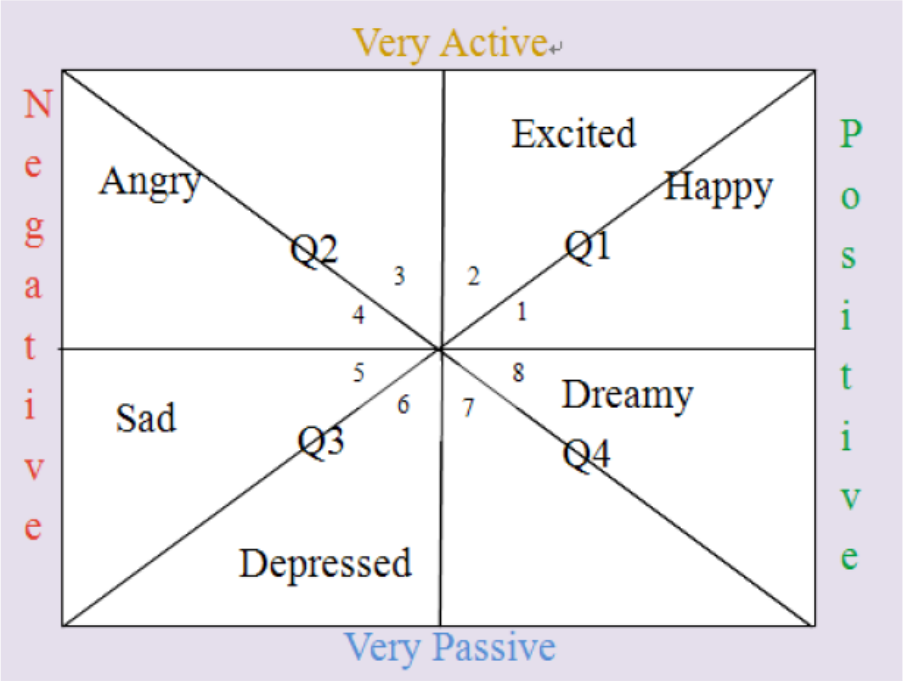


圖 6　二維情緒象限圖

2.6 音樂異質資訊整合

在西洋音樂理論中，樂譜的記載係依靠五線譜來表示，五線譜組成分為第一線、第二線、第三線、第四線、的五線，在線與線之間尚有第一間、第二間、第三間、第四間，倘有超過部分則可自行往上或往下加線或間。五線譜除了記載該樂譜的調號、拍號，其餘為分散在五線譜各處的音符與符號，如圖 7 所示。

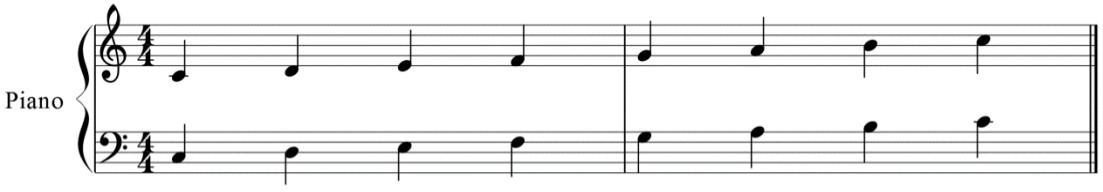


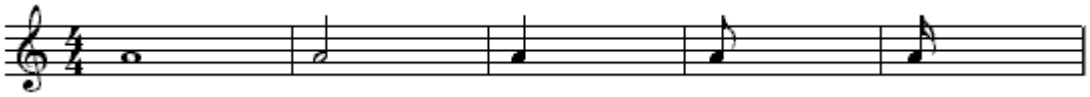
圖 7　五線譜圖

音高的表示方法則依照音的高低將音符置放於五線譜中的不同位置。音高具有「音名」、「唱名」兩種，為了便利看譜還有發展出「簡譜」來表示不同的音高，其對照表如表 4 所示。

音名	C	D	E	F	G	A	B
唱名	Do	Re	Mi	Fa	Sol	La	Si
簡譜	1	2	3	4	5	7	7

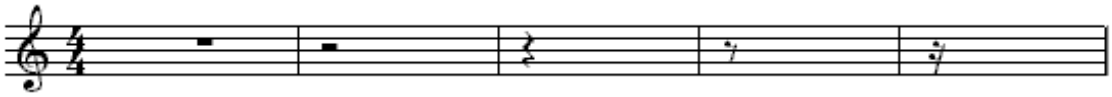
表 4　音名、唱名、簡譜對照表

音符除了音高(Pitch)之外，還具備時值(Duration)資訊。最常見的時值表示方式，為全音符、二分音符、四分音符、八分音符、十六分音符。全音符長度為二分音符的兩倍，二分音符長度為四分音符的兩倍，其餘依此類推。音符時值除了前述兩倍式的成長之外，還具有「附點」的表示方法，可以將時值增長為原本 1.5 倍的長度，亦為原時值加上原時值二分之一之長度。例如，二分附點音符長度則為，二分音符長度加上四分音符長度之總長。各式時值對照表如圖 8 所示。若無音高但有時值的則為休止符，各種休止符如圖 9 所示。



全音符　二分音符　四分音符　八分音符　十六分音符

圖 8　各式時值音符



全休止符　二分休止符　四分休止符　八分休止符　十六分休止符

圖 9　各式休止符

若要將上述五線譜與人交換或共用，則必須將樂譜數位化編碼轉。在 Beyond MIDI 一書[64]中，描述了超過 20 種以上不同的樂譜儲存方式，而數位化的樂譜檔案長久以來都呈現非常分歧的現象。例如，Finale 與 Sibelius 各有其專屬的檔案格式，彼此之間缺乏共通的轉換標準。隨著網際網路的發展，在網路上交換數位樂譜檔案的需求日益增加。目前數位樂譜的檔案以 PDF 為主，但其格式僅記錄了樂譜的印刷資訊，而沒有音樂上的描述涵義。另一方面，MIDI 格式雖然成功地用於電腦合成音樂的領域，但 MIDI 檔所能記載的紀錄與資訊與樂譜相比顯得缺乏許多。

MIDI (Musical Instrument Digital Interface) 為一種工業電子的通訊協定，定義了各種音符及彈奏控制訊號，訊號當中包含音調、強度、音量、顫音或相位等資訊，但不包含聲音資訊。在 MIDI 技術規格中，可分為硬體標準介面與軟體標準介面兩種，硬體標準介面部分可以使各種支援 MIDI 的硬體介面在輸出的時候為一致的規格，在軟體標準介面部分定義了儲存與傳輸數位資料編碼的結構，包含了音量、音高、力度、計時器訊號等，接收端收到訊號後即可依照其自身合成器，依據接收到的控制訊號合成出聲音。在 MIDI 檔案當中，以 0~127 代表音高資訊，例如中央 C 音高為 60。但是，MIDI 音樂卻無法反推出完整的樂譜。主要原因是，MIDI 檔案主要用於記載音樂合成資訊，無法一對一反推出原始樂譜資訊，亦無小節 (Measure)、重複 (Repeat)、連奏 (Slurs)等概念。雖然音樂撥放順序與原始樂譜相同，但是還原出來的樂譜卻與原始樂譜會有所差異。例如，若兩個同樣的音符之間有連結線連接的話，在 MIDI 檔案中所記載的資訊是完全一樣的。雖然在實際演奏中兩種樂譜可能是使用相同的彈奏方法演出，但是在其樂譜意義上並不完全相同。例如，圖 10 和圖 11 若轉換為 MIDI 檔案，兩者所聽起來是完全一致的。

圖 10　無連結線的樂譜

圖 11　有連結線的樂譜

為了解決 MIDI 無法完整記載樂譜的問題，在 1994 年的時候許多音樂軟體商共同發展了 NIFF (Notation Interchange File Format) 格式，能夠完整描述音樂樂譜中的各項元素及其結構性，但由於為二進位格式且複雜不易使用，最後未能成為流行的電子樂譜檔案格式。另一種 SMDL (Standard Music Description Language) 格式，亦試圖解決樂譜記載問題，但由於相容性差最終仍告失敗。在 2004 年，Recordare 公司發表了 MusicXML 格式(Music Extensible Markup Language)，利用人類及機器均能方便閱讀的 XML 技術為基礎，發展一套數位樂譜的格式。由於閱讀方便且結構清晰，對於音樂樂譜的各種符號定義的非常完整，並在網路上交換非常便利，故發展得非常快速。截至 2013 年 4 月為止，已有超過 160 種商用軟體支援 MusicXML 格式。MusicXML 格式能夠快速發展成功的主要原因為，在制定標準的過程中參考了的學術上的 MuseData 和 Humdrum 之設計，非常複雜的音樂均能被完整記載，在各種不同的軟體之間具有相當佳的相容性表現。圖 12 的 MusicXML 的樂譜如圖 13 所示。

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE score-partwise PUBLIC
    "-//Recordare//DTD MusicXML 2.0 Partwise//EN"
    "http://www.musicxml.org/dtds/partwise.dtd">
<score-partwise version="2.0">
  <part-list>
    <score-part id="P1">
      <part-name>Music</part-name>
    </score-part>
  </part-list>
  <part id="P1">
    <measure number="1">
      <attributes>
```

```
        <divisions>1</divisions>
        <key>
          <fifths>0</fifths>
        </key>
        <time>
          <beats>4</beats>
          <beat-type>4</beat-type>
        </time>
        <clef>
          <sign>G</sign>
          <line>2</line>
        </clef>
      </attributes>
      <note>
        <pitch>
          <step>C</step>
          <octave>4</octave>
        </pitch>
        <duration>4</duration>
        <type>whole</type>
      </note>
    </measure>
  </part>
</score-partwise>
```

圖 12　MusicXML 範例檔案

圖 13　MusicXML 範例樂譜

當談論到音樂的時候，不僅有聲音訊號與樂譜可以描述音樂，仍有許多不同觀點可用於描述音樂。例如，演奏者習慣透過樂譜描述音樂，音樂賞析者則偏好瀏覽原始樂譜

手稿，音樂 DJ 則較偏好錄製完成的音訊來表達音樂。在聆聽者的觀點，最習慣的音樂描述則是現場 LIVE 音樂或是錄音室錄製好的唱片。由上述的範例可得知，各種不同類型的音樂資訊在本質上即為異質的，因為音樂本身包含了各式各樣不同的媒體與描述資訊。以歌劇魅影 (Phantom of Opera) 作品為例，所包含的各式異質資訊可能包含但不限於樂譜、音樂、影片、海報、照片、傳單等多種不同格式的描述資訊，但卻沒有一套整合描述方式進行整合。

在數位的領域當中，若現實生活中的各種資訊若要轉換成電腦可以管理、組織的方法， 必須要先轉換成電腦可以處理的二進位格式。但是，在此轉換過程中，往往只將其資料內容與詮釋資料儲存成適當之格式，而未考慮到其資料真正的內涵資訊或與其相關的處理程序。例如，目前數位樂譜檔案以 PDF 或是 JPEG、TIFF 等圖像檔案為主，電腦合成音樂檔案以 MIDI 格式為主，音樂訊號檔以 MP3 為主。從上述格式可知，不論是 PDF、JPEF、 TIFF、MIDI、MP3 都欠缺完整的音樂內涵意義，其個別格式具有其本質上的限制，無法將所有與音樂有關的觀點進行整合描述。

MPEG-7 以 XML 語言為基礎，並再加入其延伸格式，作為影音(AV)內容的描述結構(Multimedia Description Schemes, MDS)。用這種方式描述影音內容，其產生的格式能夠 讓人們能夠了解所要呈現的內容，且使影音內容夠容易被搜尋、索引、過濾、使用、管理[3-6]。惟 MPEG-7 係作為影音內容描述之用，並非針對紀錄音樂而專門設計，且音樂資料的豐富度遠大於一般聲音。

有鑑於此，IEEE 組織便開始發展一套以 XML 為基礎的格式，能夠整合描述各種音樂觀點與異質資訊，成為 IEEE 1599 標準[11]，其架構圖如圖 14 所示。它依照音樂的不同特性分成六層音樂內涵描述架構，可以完全整合不同異質形式的音樂資訊，其描述檔以 XML 語言定義各層次音樂資訊的描述語法、含意及同步關係[9, 10]。最上層為一般音樂描述層 (general layer)、邏輯層 (logical layer) 描述各種形式音樂資訊間的同部與參照關係、結構層 (structural layer) 描述曲式與結構方面的資訊、譜號層 (notational layer) 描述樂譜相關的資訊、表演層 (performance layer) 描述音樂合成資訊、最底層為描述音樂訊號的音訊層 (audio layer)。

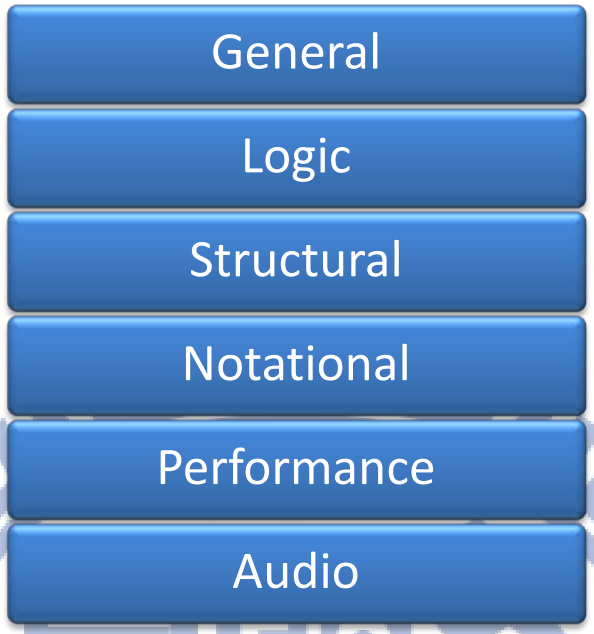


圖 14　IEEE 1599 六層架構圖

由於完整的音樂資料是由各種異質資訊所組成，例如一般常見的 MP3 檔案、MIDI 檔案、樂譜手稿、電子樂譜檔案等多種格式，IEEE 1599[11]將各種異質音樂資訊依其特性分為六層音樂內涵描述架構，可以完全整合不同異質形式的音樂資訊，其描述檔以 XML 語言定義各層次音樂資訊的描述語法及含意。最上層為一般音樂描述層(general layer)、邏輯層(logical layer)描述各種形式音樂資訊間的同部與參照關係、結構層(structural layer)描述曲式與結構方面的資訊、譜號層(notational layer)描述樂譜相關的資訊、表演層(performance layer)描述音樂合成資訊、最底層為描述音樂訊號的音訊層(audio layer)，其對應的 XML 描述如圖 15 所示。

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ieee1599 SYSTEM "http://www.mx.dico.unimi.it/ieee1599.dtd">
<ieee1599>
    <general>...</general>
    <logic>...</logic>
    <structural>...</structural>
    <notational>...</notational>
    <performance>...</performance>
    <audio>...</audio>
</ieee1599>
```

圖 15　IEEE 1599 音樂描述範例

我們可以在一般層中描述這首歌曲的標題(main_title)、歌手(voice)、作曲者(composer)、作詞者(lyricist)、曲目編號(number)、作品名稱(work_title)、發行日期(date)、以及曲風(genres)、等關於此樂曲的詮釋資料(meta data)，與此樂曲相關的音樂與影片等相關檔案的參照，則以<related_files>標記方式進行整合，如圖 16 所示。

```
<general>
        <description>
            <main_title>煎熬</main_title>
            <author type="作詞者">徐世珍</author>
            <author type="作曲者">饒善強</author>
            <work_title>感謝愛人</work_title>
            <date type="發行日期">2011-09</date>
          <genres>
            <genre name="POP" description="流行音樂"    weight="100" />
          </genres>
       </description>
       <related_files>
          <related_file file_name="煎熬.mp3"
                file_format="audio_mp3"
                encoding_format="audio_mp3"
                description="MP3" copyright="" notes="" />
          <related_file file_name="煎熬.mid"
                file_format="audio_x_midi"
                encoding_format="audio_x_midi"
                description="MID" copyright="" notes="" />
          <related_file file_name="感謝愛人.jpg"
                file_format="image_jpeg"
                encoding_format="image_jpeg"
                description="COVER" copyright="唱片公司"
                notes="" />
          <related_file file_name="煎熬_樂譜 1.xml"
                file_format="text_html"
                encoding_format="text_html"
                description="版本一樂譜" copyright=""
                notes="" />
          <related_file file_name="煎熬_樂譜 2.xml"
                file_format="text_html"
                encoding_format="text_html"
                description="版本二樂譜" copyright=""
                notes="" />
       </related_files>
</general>
```

圖 16　IEEE 1599 音樂一般描述層範例

除此之外，IEEE 1599 標準在結構層包含音樂物件 (music object) 從作曲與樂理觀點的因果關係之明確描述，例如音樂物件如何被描述成像之前的音樂物件的變化。但是，這不是時間上 (timed) 或是實際上 (physically) 順序的變化，而是因果關係的變化。在這層裡面的資訊不包含音樂事件的時間順序和絕對時間的明確描述，而是描述了在樂譜的音樂物件的在合成或分析過程中的轉換和定位之間的因果關係，但是目前沒有已經確切可以使用的標準來表示結構層的資訊，儘管如此，[65]指出仍有其他方法來表示其資訊。第一種為使用 SMDL[66]、第二種使用包含高階音樂程序語言(high-level music programming languages) 與正規形式工具進行模型化。最後一種為本系統採用之方法，將音樂結構資訊描述在 IEEE 1599 標準中的結構層，如圖 17 所示。

```
<structural>
    <analysis>
        <segmentation id=”A”>
            <segment id=”s1”>
                <segment_event   event_ref=”se6” />
            </segment>
        </segmentation>
    </analysis>
</structural>
```

圖 17　IEEE 1599 結構描述範例

第三章、系統分析與設計方法

3.1 系統流程

本研究所提出之系統流程如圖 18 所示，包含了七個流程與描述資訊，在流程當中所有產生的描述資訊均會依據 IEEE 1599 標準整合描述，再存入 IEEE 1599 描述檔資料庫中。

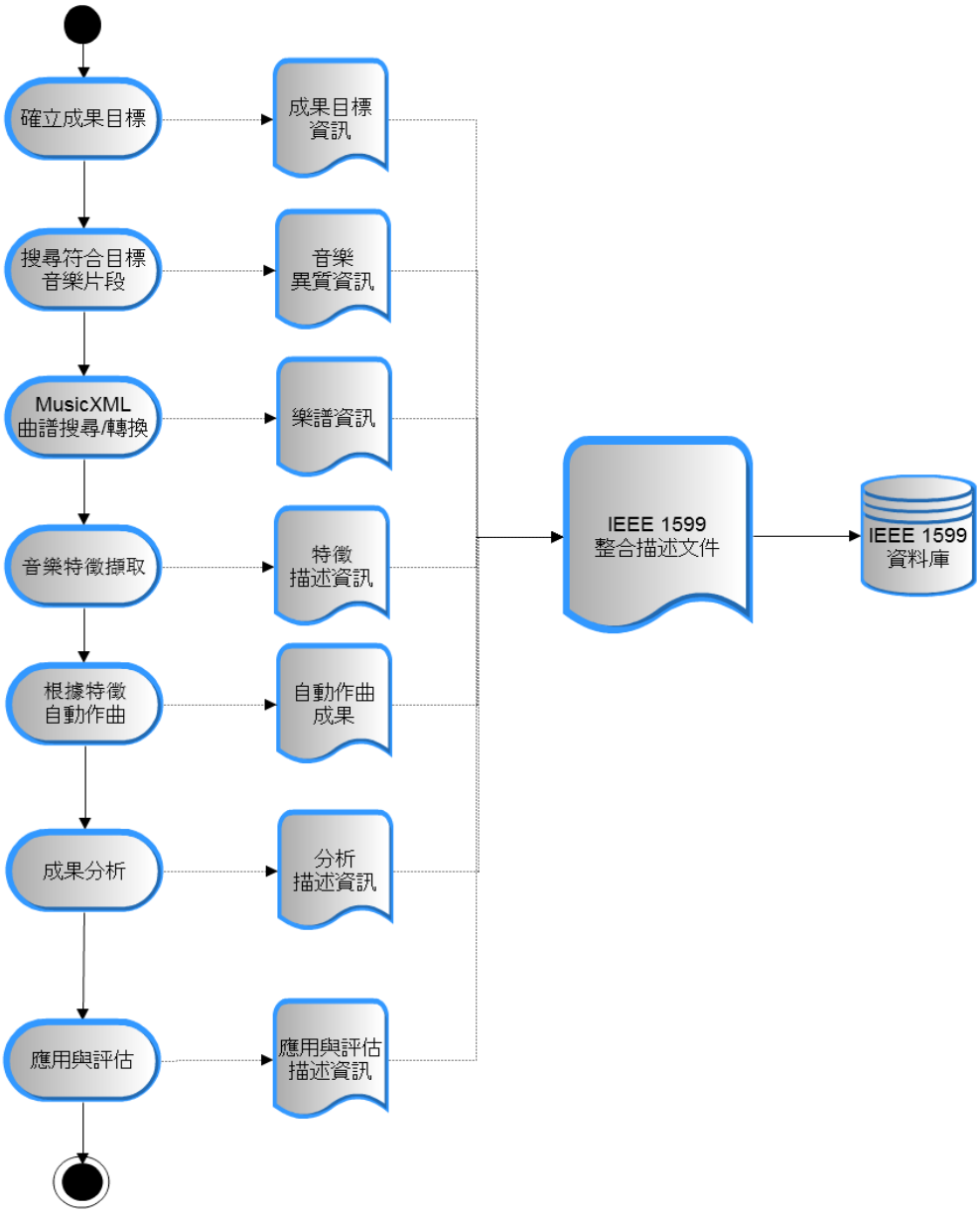


圖 18　系統流程圖

透過圖 18 所示，本系統有確立成果目標、搜尋符合目標的音樂片段、MusicXML 曲譜搜尋/轉換、音樂特徵擷取、根據特徵自動作曲、成果分析、應用與評估等七個階段，並在每一階段執行結束後皆會產生描述資訊，包含成果目標描述資訊、音樂異質描述資訊、樂譜描述資訊、特徵描述資訊、自動作曲成果描述資訊、成果分析描述資訊、應用與評估描述資訊等七種描述資訊。惟 IEEE 1599 標準為整合描述異質資訊之用，若要記載自動作曲相關資訊，必須擴充及修正其 XML 文件大綱。我們將標準中的文件大綱稍作修改，以符合本系統開發需求。我們透過修正式 IEEE 1599 標準記載流程中所產生的所有紀錄，再存放於 IEEE 1599 資料庫當中，系統流程於 3.1.1 至 3.1.7 中詳細說明。

3.1.1 確立成果目標

在本系統的目標確立階段，必須先確立產出成果必須滿足的成果目標，倘若沒有定義明確的成果目標，系統產出成果將無法有效地被客觀的評估。例如，我們可以定義成果目標為，產出的音樂必須符合特定曲風或情緒，或是特定統計方式下必須有特定比例的受測者能夠同意已定義的成果目標。我們可以將確立完成的成果目標定義在修正式 IEEE 1599 標準的一般描述層當中，其新增元素內容模型如圖 19 所示。

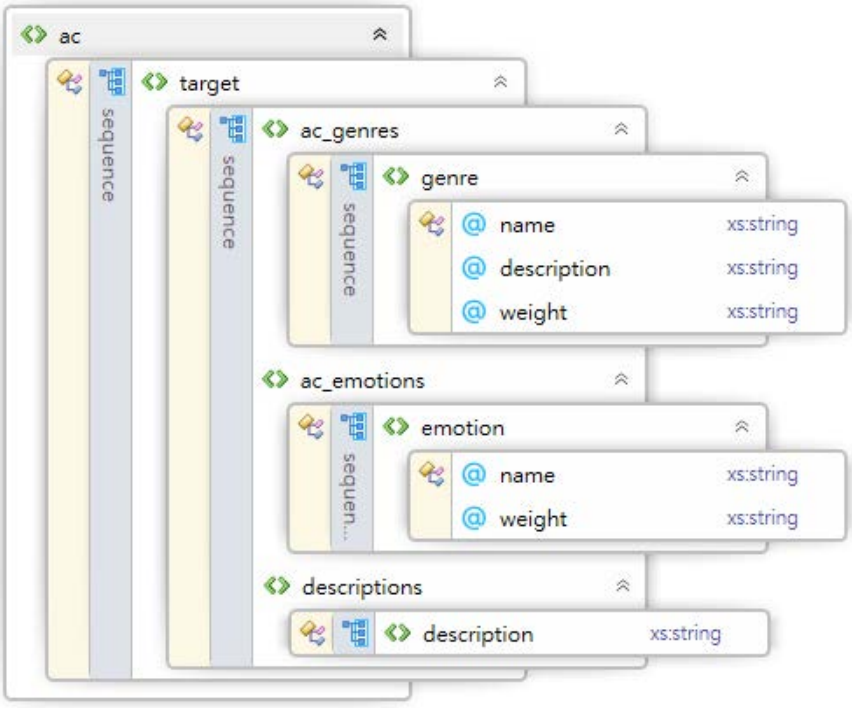


圖 19　成果目標描述內容模型

3.1.2 蒐集符合目標音樂片段

在確立明確的成果目標後，則必須廣泛蒐集符合訂立的目標音樂片段。此階段目標為，我們可以透過蒐集符合目標需求的音樂片段，並以能夠達到第一階段確立的曲風及情緒為要件，在不限制來源的情況下蒐集符合條件的音樂及其異質資訊，避免有所篩選而造成偏差。最主要的原因為，倘若蒐集的音樂片段過度集中在特定作曲家或是特定歌手，容易造成在特徵擷取階段，擷取出的特徵可能會呈現出較為偏頗的規則。我們蒐集到的音樂片段，可以全部紀錄在修正式 IEEE 1599 標準中的一般描述層。related_files 元素下可以包含多個related_file 元素，以屬性方式紀錄各異質資訊的資訊，包含檔案名稱、檔案格式、相關描述、相關備註及唯一識別碼，其內容模型如圖 20 所示。

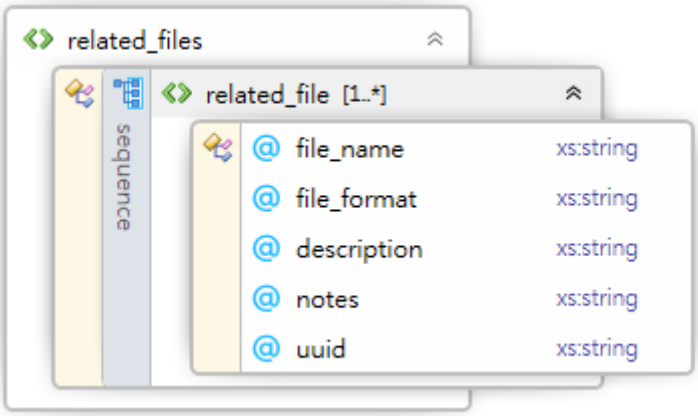


圖 20　相關檔案記載於一般描述層之內容模型

3.1.3 MusicXML 曲譜搜尋/轉換

在確立成果目標並蒐集完符合其目標的音樂素材之後，素材當中需要包含音樂的原始樂譜檔案進行分析，倘若僅蒐集到非數位樂譜檔案，則要透過人工或其他工具轉換為數位樂譜檔案。音樂素材需要包含樂譜的原因有二點：(1) 由於聲音訊號檔案相較於原始音樂樂譜檔案，相對欠缺音樂內涵意義。(2) 聲音訊號檔案直接轉換為電子樂譜易受到音樂中其他聲響干擾，轉換為樂譜相當不易。

故在此階段當中，倘若取得作曲者所提供的原始樂譜，或經其他具有公信力的專家或出版社所提供的樂譜，對於後續分析階段較能得到與原曲相貼近的特徵。除此之外，雖然 IEEE 1599 標準當中的邏輯層能夠記載樂譜資訊，但由於目前尚無軟體支援從 IEEE 1599 描述檔案當中讀取其樂譜。本系統採用目前大部分商業軟體均支援的 MusicXML 格式記載樂譜資訊，再將 MusicXML 檔案記載於修正式 IEEE 1599 標準中的一般描述層，其餘圖像檔案格式的樂譜則記錄在樂譜層。譜號層透過 graphic_instance_group 元素的子元素 graphic_instance 記載，position_in_group 屬性紀錄該譜物件於其 group 的相對位置，其他屬性則記錄檔案名稱、檔案格式、編碼格式、備註等資料，其資料模型如圖 21 所示。

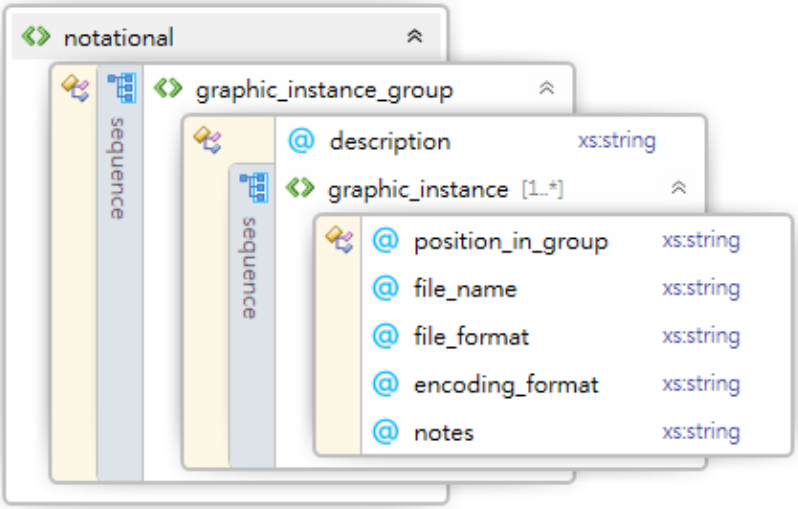


圖 21　相關樂譜記載於譜號層之內容模型

3.1.4 音樂特徵擷取

在音樂特徵擷取流程當中，目標為將流程中蒐集的 MusicXML 檔案，進行分析以得到音樂特徵。本系統在此階段僅定義抽象流程，而不定義任何實作方式，但擷取完成之特徵必須包含量化資訊，否則無法適用於本系統之架構。音樂特徵擷取方式繁多，針對不同需求可能必須依賴不同分析方法，而產生出不同分析觀點的結果。

例如，我們可以將 MusicXML 樂譜檔案剖析完畢後，紀錄所有和弦進行的狀態改變，再以和弦為單位統計小節中出現的旋律音高、音長之變化，再以 Markov Chain 為基礎建立為模型。除了以和弦為單位統計之外，亦能夠以樂句為單位切割，統計每一樂句的音高、音長資訊作為特徵，當作重新產生新音樂的規則。當音樂特徵擷取完成後，我們可以將產生的結構描述檔案儲存在結構層。在結構層當中，於 analysis 的當中如圖 22 所示。

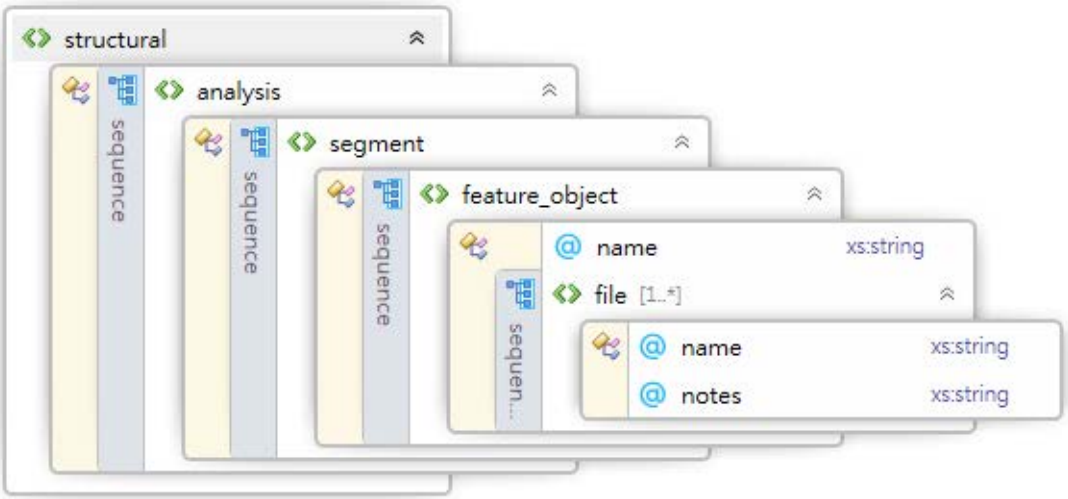


圖 22　音樂特徵記載於結構層之內容模型

3.1.5 根據特徵自動作曲

在根據特徵自動作曲流程中，目標為將量化分析得到的特徵，建立為模型供自動作曲，並產生出新的音樂。假設我們的音樂特徵為和弦進行的機率模型，以及各和弦下旋律的音高、音長機率模型，即可以依照該機率模型產生出新的音樂，並以 MusicXML 方式記載其音樂樂譜，再標示於修正式 IEEE 1599 標準中的一般描述層當中，如圖 20 所示。在自動作曲的過程當中，可以依據確立的目標調整原創性與蒐集的音樂片段之相似程度。除此之外，自動作曲可以依據特徵產生出無限首新的音樂，因此可以依據確立目標定義適應性函數，在自動作曲過程中就先行衡量其結果的好壞程度，避免將該音樂特徵自身缺陷所導致的較差結果。

3.1.6 成果分析

在自動作曲階段結束之後，必須有客觀標準評估該成果是否符合系統當中在第一階段所確立的目標。在成果分析階段，可以針對產生出的音高與音長進行分析，以得到量化特徵。由於音樂與悅耳程度的關係，目前尚難以建立一套描述公式，判定產生出來的音樂好聽與否。但是根據[58]的論文所提出的觀點，能夠藉由統計旋律的特徵，能夠協助我們在未來建立一套客觀的評估標準，我們可以透過我們將評估的成果以 XML 為格式編碼，再標示於修正式 IEEE 1599 標準的結構層，如圖 22 所示。

3.1.7 應用與評估

在評應用與評估階段當中，最重要的目標為大規模廣泛蒐集其他使用者的反饋，再由專家學者針對其反饋檢視，進而修正系統第二階段所蒐集的音樂、第三階段的自動作曲演算法。

將自動作曲成果儲存於資料庫當中，並以開放式介面供各種開發者或有興趣的人或團體進行應用。使用者所反饋的資料，我們以 XML 格式編碼，再標示於修正式 IEEE 1599 標準的結構層。

3.2 系統架構設計

為了使 IEEE 1599 標準能夠適用於自動作曲研究領域，因此本文針對 IEEE 1599 標準進行擴充，以符合本系統之需求，修正式 IEEE 1599 的文件大綱於附錄一所示。在本系統流程當中，會產生修正式 IEEE 1599 標準描述檔、音樂特徵描述檔、作曲結果描述檔，評估結果描述檔，使用者回饋描述檔等五個 XML 文件，而其對應的文件大綱如附錄二至附錄六所示。

XML 是一組可以用來建造標記語言的規則，透過將標記加入文件當中，即可標示出所有標記之間的關係，我們可以按照自己所定義的文件結構，在文件當中儲存任何類型的資訊。XML 文件僅用於記錄資料，但並不限制語意(semantics)，在擴充程度上具有相當程度的彈性。雖然文件本身並不記錄文件樣式，但可透過 XSLT(Extensible Stylesheet Language Transformations) 將來源 XML 文件的內容轉換成另一種不同格式或結構的文件。在文件品質檢驗方面，在語法規則或是文件模型觀點支援文件大綱以檢查是否完善(well-formed)。在多個 XML 文件之間，可以透過 XLink(XML Linking Language)在不同 XML 文件之間建立超連結的語言，能夠連結一系列 XML 文件的資源。除此之外，XML 為開放式標準，在各種網路服務間具有良好的互通性，能夠被輕易整合，可以有效解決目前分散式系統使用各自採用不同機制造成整合困難的問題。

基於上述採用 XML 編碼的優點，IEEE 1599 標準與 MusicXML 檔案均採用 XML 格式編碼，為了使本自動作曲系統流程及其結果所產生的文件能夠妥善儲存及應用，本系統亦採用 XML 格式編碼儲存各流程所產生之描述。在修正式 IEEE 1599 標準的描述檔案中，所有物件都會被賦予唯一識別碼識以識別其描述資訊，作為該筆紀錄存放於資料庫的主鍵，以便檢索之用。本系統共有六個資料庫，分別儲存 IEEE 1599 描述資訊，異質音樂資訊描述資訊，音樂特徵描述資訊、自動作曲成果描述資訊、評估結果描述資訊、使用者反饋描述資訊，各描述資訊與資料庫之關係圖如

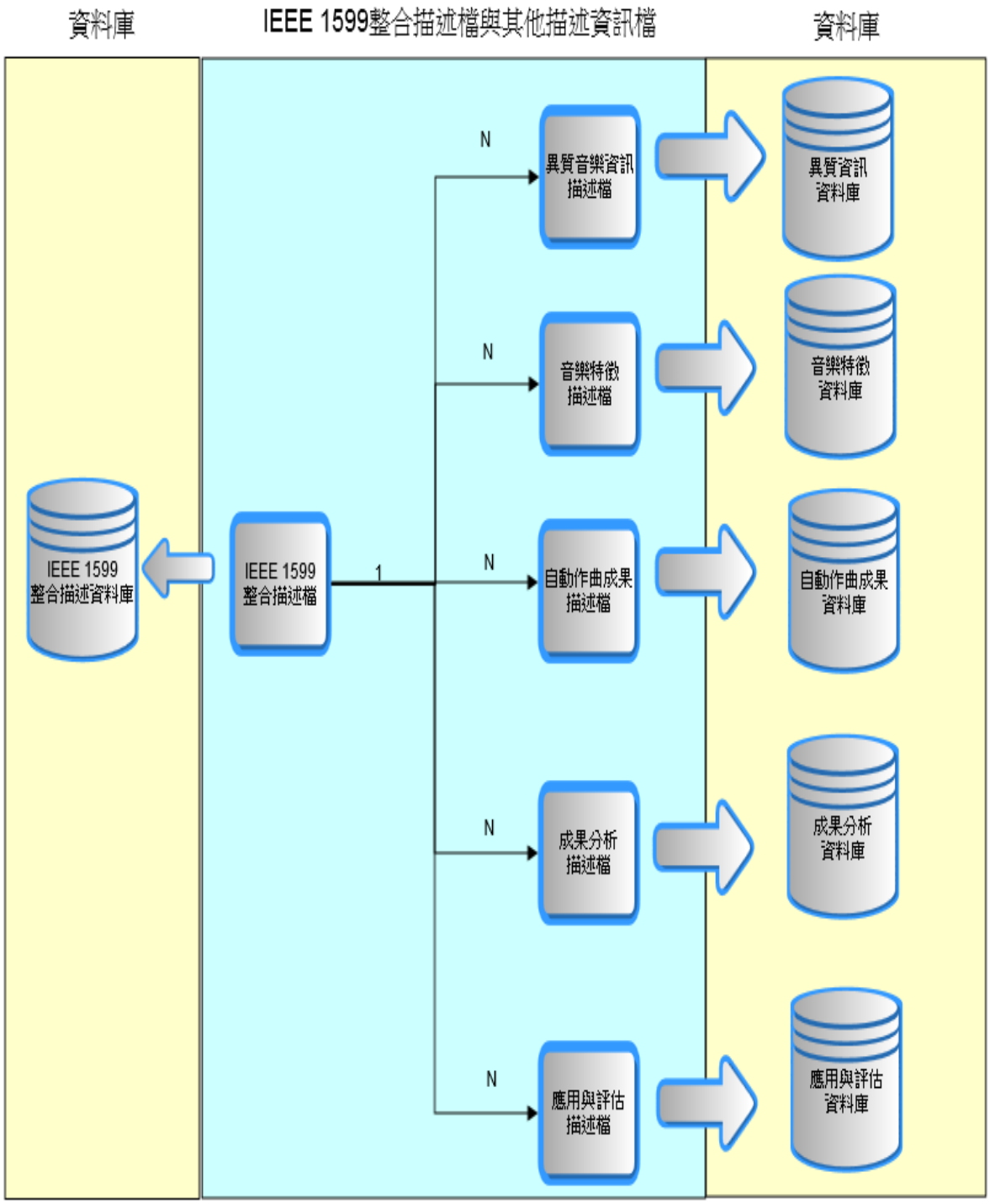

資料庫
IEEE 1599整合描述檔與其他描述資訊檔
資料庫
IEEE 1599
整合描述資料庫
IEEE 1599
整合描述檔
1
N
異質音樂資訊
描述檔
異質資訊
資料庫
N
音樂特徵
描述檔
音樂特徵
資料庫
N
自動作曲成果
描述檔
自動作曲成果
資料庫
N
成果分析
描述檔
成果分析
資料庫
N
應用與評估
描述檔
應用與評估
資料庫

圖 23 所示。

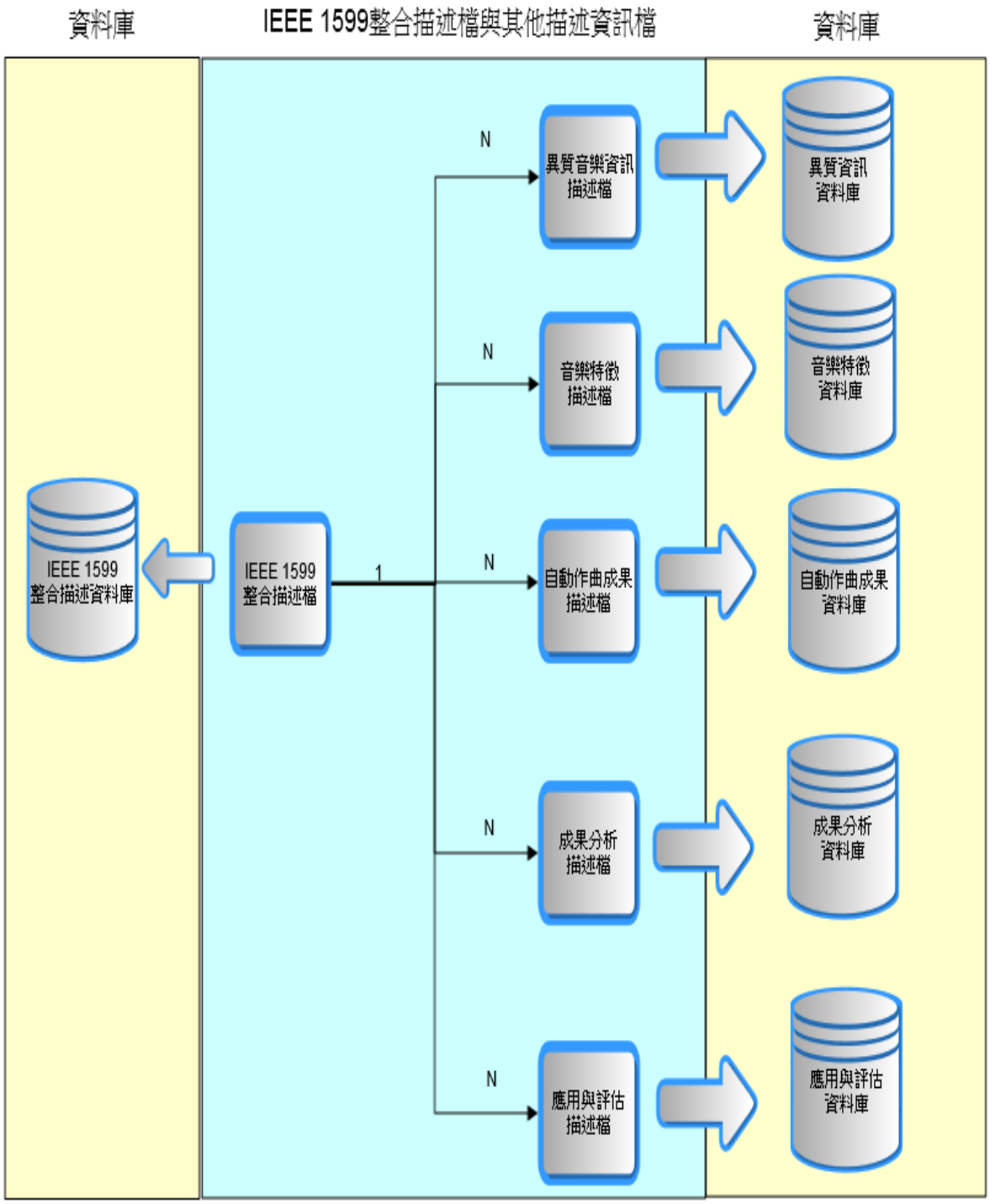


圖 23　IEEE1599 整合描述檔及其他描述檔與資料庫之關係

由

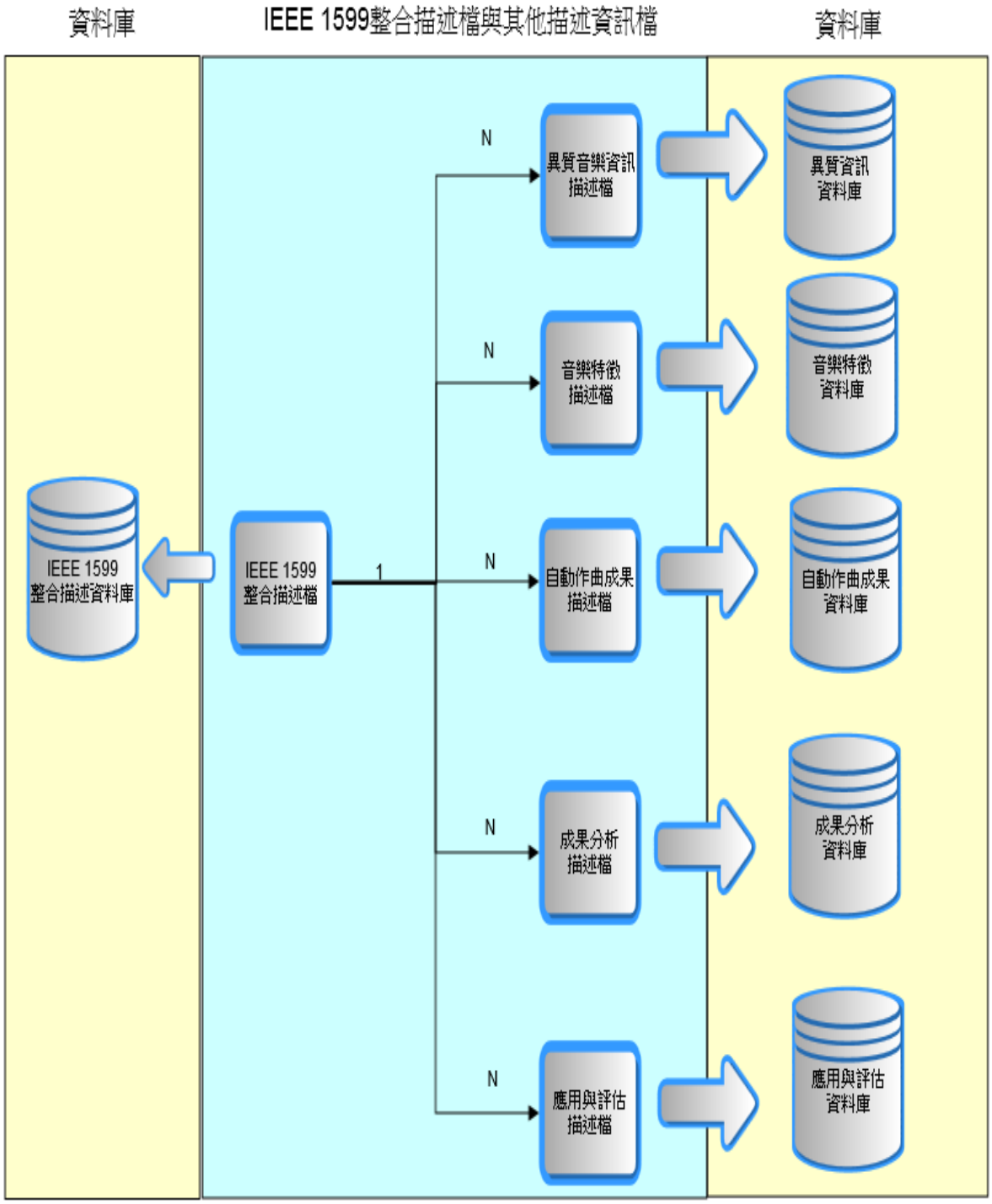


圖 23 所示，單一個 IEEE 1599 整合描述檔，可以連結多個異質音樂資訊描述檔、一個音樂特徵描述檔、多個自動作曲成果描述檔、多個評估結果描述檔、多個使用者反饋描述檔。在異質音樂資訊描述檔案當中所記載的所有異質資訊，均會被賦予唯一識別

碼以供檢索。其餘的描述檔案，亦會被賦予唯一識別碼以茲識別。

在存放各式異質資訊的資料庫當中，即會以該唯一識別碼做為主鍵。在 IEEE 1599 所有整合描述檔當中以唯一識別碼以代表識別到該描述檔案，再以該唯一識別碼為關鍵字即可至對應資料庫檢索並存取其資訊。倘若有在資料庫檢索所需描述資訊的需求，即可以透過 XQuery 語言檢索。XQuery 是被設計用來查詢 XML 資料，且不僅限於 XML 文件，還包含任何以 XML 形態呈現的數據或資料庫。相較於直接將 XML 文件存放在關聯式資料庫的欄位當中再以 SQL 語言查詢，無論在檢索、更新、刪除等指令的效率上，以原生的 XML 資料庫來處理 XML 文件會比以關聯式資料庫處理 XML 文件的方式來得更有效率。

3.3 系統目標

本研究設計之基於 IEEE 1599 標準的自動作曲系統，係以異質音樂資訊整合為中心，在參考[57]所提出之評估架構，再整合音樂分析、特徵擷取、自動作曲、成果分析等概念，及使學術研究流程更為流暢與便利實際應用為前提之下進行改良，使其具有重用性、延展性、擴充性特性之自動作曲系統架構。此系統架構目的為將自動作曲中各流程緊密整合，為確保系統的彈性程度，各流程元件均透過開放式標準連接、儲存、存取，並依據 IEEE 1599 標準整合所有流程中產生的紀錄與成果，最終目標為使自動作曲技術能夠有效發展、評估與應用，且能夠量產出各式音樂，依據不同用途與要求產生出對應之音樂，使不具有音樂創作能力的使用者，也能夠透過本系統得到所想要的音樂，而不須另外尋覓音樂創作專業人才，使人為介入創作的程度能夠最小化。

圖 24　系統目標示意圖

第四章、結果與系統功能範例分析

我們透過本文所提出的系統架構進行自動作曲作為示例，以進一步驗證本系統的可用性。在資料庫建置方面，採用 eXist 資料庫系統。主要的原因是，eXist 在 XML 方面係完全依照 W3C 的標準開發，亦支援透過 REST 介面與 AJAX 型態的網頁表單互動及 WebDev 介面直接將 XML 檔案直接存入資料庫當中。因此，我們決定採用 eXist 系統來建置本自動作曲系統之資料庫，無論在資料儲存的、介面的標準化、未來發展的延展性，應用程式連接的開放程度，都提供了一套完整的解決方案。綜觀過去自動作曲相關研究，方式能夠分為內顯式與外顯式兩種，為了能夠與過去已發展之自動作曲系統相容，故系統實作部分亦參考過去自動作曲方法之分。在系統實作部分，我們以此架構進行內顯方式自動作曲範例，並以修正式 IEEE 1599 標準整合描述各式資訊。

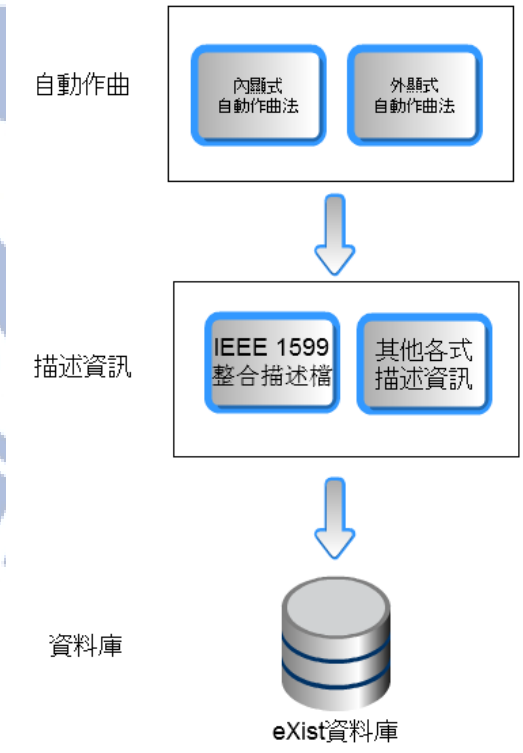


圖 25　系統設計概念示意圖

本系統在自動作曲的第一步驟，必須先確立成果目標方面。倘若我們希望能夠以自動作曲方式，產生出有具有華語流行音樂風格的音樂，製做出的音樂情緒分別為正向與負向情緒兩種版本。因此，成果目標在修正式 IEEE 1599 標準中的一般描述層如圖 26 所示。

```
<general>
…
    <ac>
        <target>
            <genres>
                <genre name=”pop” description=”pop” weight=”100%” />
            </genres>
            <emotions>
                <emotion name=”positive” weight=”100%” />
                <emotion name=”negative” weight=”100%” />
            </emotions>
            <descriptions>
                <description>華語流行音樂</description>
            </descriptions>
        </target>
    </ac>
…
</general>
```

圖 26　自動作曲成果範例：目標確立

成果目標確立完成後，我們即可從現有的華語流行音樂當中，選擇我們欲分析的音樂，在此階段我們挑選了以下十首的音樂來擷取其音樂特徵，供自動作曲階段使用。

伍思凱：愛的鋼琴手	梁靜茹：如果有一天
羅志祥：好朋友	梁靜茹：寧夏
陳奕迅：K 歌之王	劉若英：後來
陳奕迅：十年	光良：童話
周杰倫：千里之外	張棟樑：當你孤單你會想起誰

表 5　自動作曲成果範例：欲分析的音樂

在本系統的第三步驟為 MusicXML 曲譜搜尋/轉換階段，在我們向以出版流行音樂歷史悠久的卓著出版社購買上述十首歌之樂譜，並將和弦以及主旋律以打譜軟體輸入，除了輸出為 PDF 檔案供人讀譜之用，亦轉存為 MusicXML 檔案以供分析。因此，在修正式 IEEE 1599 標準一般描述層當中則記載該五首歌之異質資訊，包含 MP3 聲音訊號檔、MIDI 檔案、MusicXML 檔案、PDF 樂譜檔案，其中的 MusicXML 檔案可作為本系統分析之用。本範例在修正式 IEEE 1599 標準的一般描述層所記載的資訊如圖 27 所示。

```
<related_files>
…
    <related_file file_name=" 愛的鋼琴手.mp3" file_format="audio_x-mp3" description=" MP3 File" notes=" M1" uuid=" 0d5cd8a1-d895-463d-8a3c-0853f73993dc" />
    <related_file file_name=" 愛的鋼琴手.mid" file_format="audio_x_midi" description=" MIDI File" notes=" M1" uuid=" 47484120-4876-4c41-bcb3-c7d26154a8e7" />
    <related_file file_name=" 愛的鋼琴手.xml" file_format="text_xml" description=" XML Score File" notes=" M1" uuid=" 68a4c275-3312-4a5b-b4e9-237395def01e" />
    <related_file file_name=" 愛的鋼琴手_1599.xml" file_format="text_xml" description="IEEE 1599 file" notes=" M1" uuid=" 7d878ee7-4e5a-4cda-9c2a-ac2f7d54f2df"/>
    <related_file file_name=" 好朋友.mp3" file_format="audio_x-mp3" description=" MP3 File" notes=" M3" uuid=" 97a2fb77-e992-4d6f-a889-ed2888dc1a9e" />
    <related_file file_name=" 好朋友.mid" file_format="audio_x_midi" description=" MIDI File" notes=" M3" uuid=" 151c2fa5-e8d4-4dbc-b683-b0308d90b995" />
    <related_file file_name=" 好朋友.xml" file_format="text_xml" description=" XML Score File" notes=" M3" uuid=" ab9fb22d-5505-4b1d-8c64-54c66c883ed2" />
    <related_file file_name=" 好朋友._1599.xml" file_format=" text_xml" description=" IEEE 1599 file" notes=" M3" uuid=" 4ddce2c5-d427-49f8-8c18-77cd1efd93ee"/>
    <related_file file_name=" K 歌之王.mp3" file_format="audio_x-mp3" description=" MP3 File" notes=" M4" uuid=" 6849f025-cfac-4034-a8b1-c313f877efe9" />
    <related_file file_name=" K 歌之王.mid" file_format="audio_x_midi" description=" MIDI File" notes=" M4" uuid=" 9ebee785-f0d1-4982-b5b5-1f52b608f8c7" />
    <related_file file_name=" K 歌之王.xml" file_format="text_xml" description=" XML Score File" notes=" M4" uuid=" f2e1284d-8722-4c0e-bfed-f346c0ffe10a" />
    <related_file file_name=" K 歌之王.xml" file_format="text_xml" description=" IEEE 1599 file" notes=" M4" uuid=" 790d907f-5dca-46ae-af5c-98c33713cf64"/>
….
</related_files>
```

圖 27　自動作曲成果範例：異質資訊記載

我們可以透過分析記載在整合描述檔裡面的 MusicXML 樂譜以取得特徵，而音樂的基本元素有和聲、旋律、節奏。合聲是整個音樂的基底，旋律是在合聲當中發展，旋律之間的快慢程度則可以構成節奏，故在此階段則將上述三元素進行分析，以取得基本的音樂元素特徵。因此在本系統範例的特徵擷取階段，參照[40]所提出的方法，將樂譜當中的音符視為多個序列(sequence)，包含和弦、主旋律音高與音長。在本階段共有兩個部分，並將其多個序列透過一階馬可夫過程(First-Order Markov Process)處理。

第一個部分為和絃進行的特徵，而第二個部分為主旋律。在進行特徵擷取之前，我們先將所有蒐集的音樂調性轉到 C 大調，包含和弦進行及其主旋律。在和絃進行的部份，我們依據樂譜上所記載的和絃作為馬可夫鍊的狀態(States)，若和弦發生轉位則視為同個和弦，忽略其改變的根音。我們即可計算不同和弦之間的變化以求得轉移機率(Transition Probabilities)，最後即可得到一個轉移矩陣(Transition Matrix)。在主旋律部分則分為音高與音長，以和弦為單位進行統計。音高記載方式採 MIDI 音高方式記載，音長則以該音符在該小節所佔拍數為記載方式，再以前述方式求得轉移矩陣。

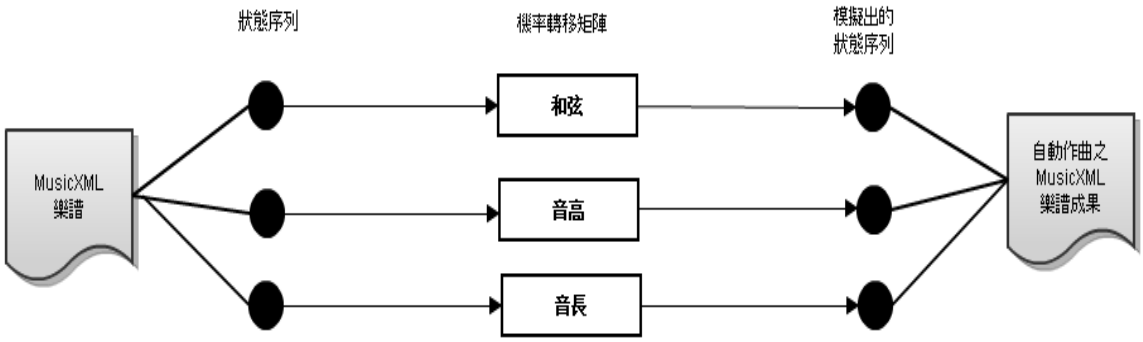


圖 28　本系統採用之一階馬可夫鍊示意圖

經過分析過後，會得到兩個轉移機率描述資訊，包含和絃進行與主旋律(音高及音長)，即作為本系統的特徵擷取結果，並於在修正式 IEEE 1599 標準當中的一般描述層當中記載描述檔案資訊，供後續自動作曲之用，如圖 29 所示。

```
<structural>
    <analysis>
        <segment>
            <feature_object   name=”AC_result”>
                <file name=”M1_result.xml” notes=” M1”   uuid=”
902658e7-e73d-4811-8eb9-f55750bdb816”/>
                <file name=”M2_result.xml” notes=” M2”   uuid=”
8dfe70d5-7d15-439f-8cea-b3caaa5ae50b”/>
                <file name=”M3_result.xml” notes=” M3” uuid=”
142a6549-a87b-48f4-9f24-8c28046ad202” />
                ……
            </feature_object>
        </segment>
    </analysis>
</structural>
```

圖 29　自動作曲成果範例：特徵擷取結果描述資訊

在經過分析該三首音樂後，其和弦與主旋律的特徵描述檔透過 XML 格式記載(M1_result.xml~ M3_result.xml)，再將其 XML 檔案資訊記載在修正式 IEEE 1599 標準的整合描述檔的一般描述層，其內容模型如圖 30 所示。我們以其中一首音樂的分析結果為範例說明，其和絃轉換機率如表 6 所示，對應的 XML 描述檔如圖 31 所示。主旋律音高轉換如表 7 至表 12 所示，其對應的 XML 描述檔如圖 32 所示，音長轉換如表 13 所示，其對應的 XML 描述檔如圖 33 所示。

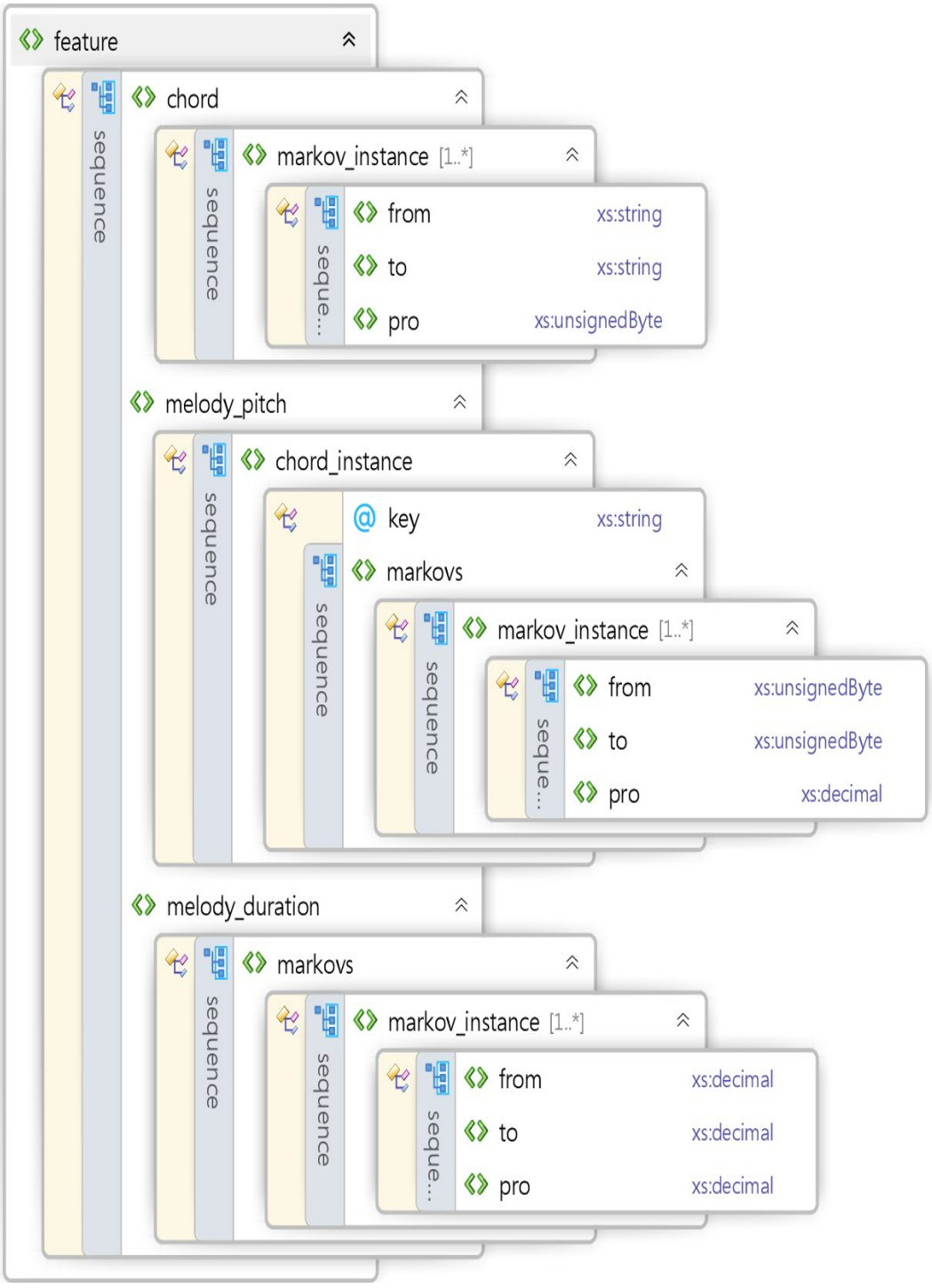


圖 30　特徵描述檔內容模型

單位：和弦

To / From	C	G	Am	Em	Dm	F
C	0	0.5	0	0	0.5	0
G	0	0	1	0	0	0
Am	0	0	0	1	0	0
Em	0	0	0	0	0	1
Dm	0	1	0	0	0	0
F	1	0	0	0	0	0

表 6　伍思凱「愛的鋼琴手」之和弦轉換機率

```
<feature>
    <chord>
        <markov_instance>
            <from>C</from>
            <to>C</to>
            <pro>0</pro>
        </markov_instance>
        <markov_instance>
            <from>C</from>
            <to>G</to>
            <pro>0</pro>
        </markov_instance>
        ….
    </chord>
</feature>
```

圖 31　伍思凱「愛的鋼琴手」之和弦轉換機率對應 XML 描述檔案(節錄部分)

單位：MIDI 音高

To / From	67	69	71	72
67	0.5	0.5	0	0
69	0	0	1	0
71	0	0	0	1
72	0	0	1	0

表 7　伍思凱「愛的鋼琴手」：C 和弦下主旋律音高轉換機率

單位：MIDI 音高

To / From	67	69	71	72
67	0.5	0.5	0	0
69	0	0	1	0
71	0	0	0	1
72	0	0	1	0

表 8　伍思凱「愛的鋼琴手」：G 和弦下主旋律音高轉換機率

單位：MIDI 音高

To / From	55	57	59	60
55	0	1	0	0
57	1	0	0	0
59	0	1	0	0
60	0	0	0.25	0.75

表 9　伍思凱「愛的鋼琴手」：F 和弦下主旋律音高轉換機率

單位：MIDI 音高

To / From	59	60	64
59	0	0	1
60	0.75	0.25	0
64	0	1	0

表 10　伍思凱「愛的鋼琴手」：Am 和弦下主旋律音高轉換機率

單位：MIDI 音高

To / From	62	69
62	0.75	0.25
69	1	0

表 11　伍思凱「愛的鋼琴手」：Dm 和弦下主旋律音高轉換機率

單位：MIDI 音高

To / From	64	65
64	0	1
65	1	0

表 12　伍思凱「愛的鋼琴手」：Em 和弦下主旋律音高轉換機率

```
<feature>
    <chord>
    ...
    </chord>
  <melody_pitch>
    <chord_instance key="C">
      <markovs>
        <markov_instance>
          <from>67</from>
          <to>67</to>
          <pro>0.5</pro>
        </markov_instance>
        <markov_instance>
          <from>67</from>
          <to>69</to>
          <pro>0.5</pro>
        </markov_instance>
        <markov_instance>
          <from>69</from>
          <to>71</to>
          <pro>1</pro>
        </markov_instance>
        <markov_instance>
          <from>71</from>
          <to>72</to>
          <pro>1</pro>
        </markov_instance>
        <markov_instance>
          <from>72</from>
          <to>71</to>
          <pro>1</pro>
        </markov_instance>
      </markovs>
    </chord_instance>
    ….
  </melody_pitch>
</feature>
```

圖 32　伍思凱「愛的鋼琴手」之主旋律音高轉換機率之 XML 描述檔案(部分)

單位：拍數

To / From	0.5	2	3
0.5	0.93	0.05	0.02
2	1	0	0
3	1	0	0

表 13　伍思凱「愛的鋼琴手」之主旋律音長轉換機率

```
<feature>
    <chord>
    …
    </chord>
  <melody_duration>
      <markovs>
        <markov_instance>
          <from>0.5</from>
          <to>0.5</to>
          <pro>0.93</pro>
        </markov_instance>
        <markov_instance>
          <from>0.5</from>
          <to>2</to>
          <pro>0.05</pro>
        </markov_instance>
        <markov_instance>
          <from>0.5</from>
          <to>3</to>
          <pro>0.02</pro>
        </markov_instance>
    ….
  </melody_duration>
</feature>
```

圖 33　伍思凱「愛的鋼琴手」主旋律音長轉換機率之 XML 描述檔案(部分)

在透過前述的和絃轉換機率、音高轉換機率、音長轉換機率，我們即可透過其機率模型自動作曲，產生出自動作曲結果，並描述在修正式 IEEE 1599 標準中的一般描述層當中。自動作曲成果必須再經過調性與速度的調整，以符合原先所確立的目標。

我們一共產生了四首歌曲，拍號為 4/4 拍子、長度為 8 小節的歌曲。但是，為了使歌曲有結束感，我們在第九小節會使和弦回到一級和弦，旋律部分回到主音，且音長會較長。在正向情緒的部分，調性分別為 C 大調與 D 大調，速度為 100 及 120。負向情緒部分則為升 C 小調與升 D 小調，速度為 72，而和弦則以分解和弦的形式呈現。

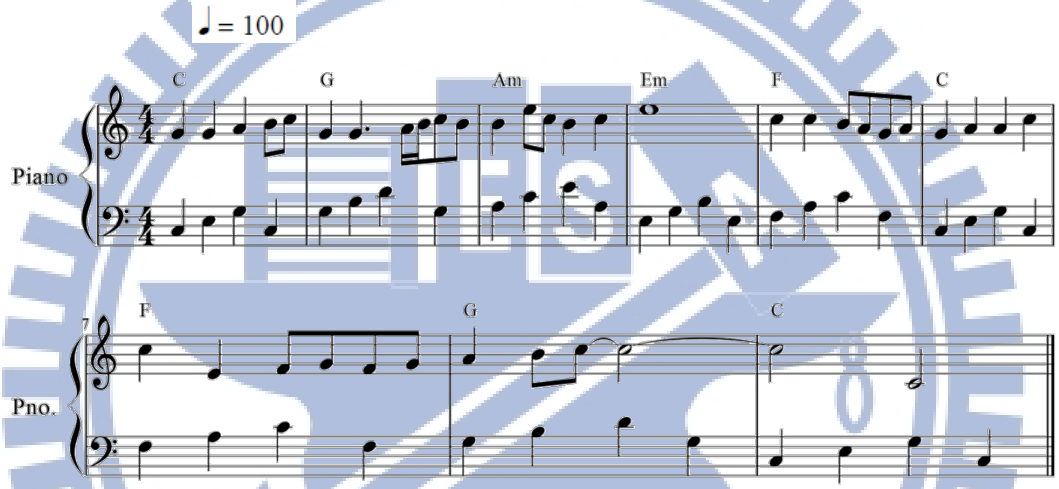


圖 34　自動作曲成果：正向情緒(1)

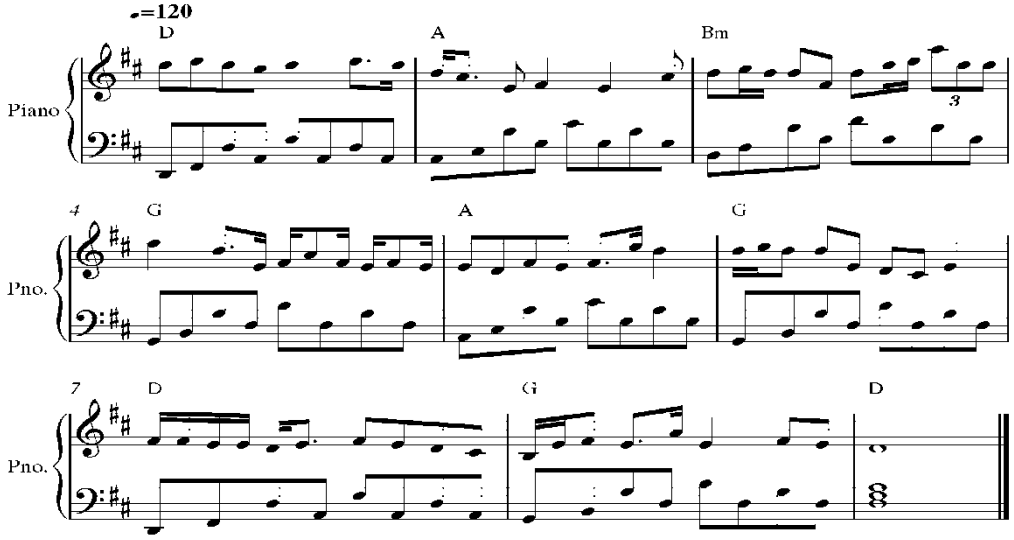


圖 35　自動作曲成果：正向情緒(2)

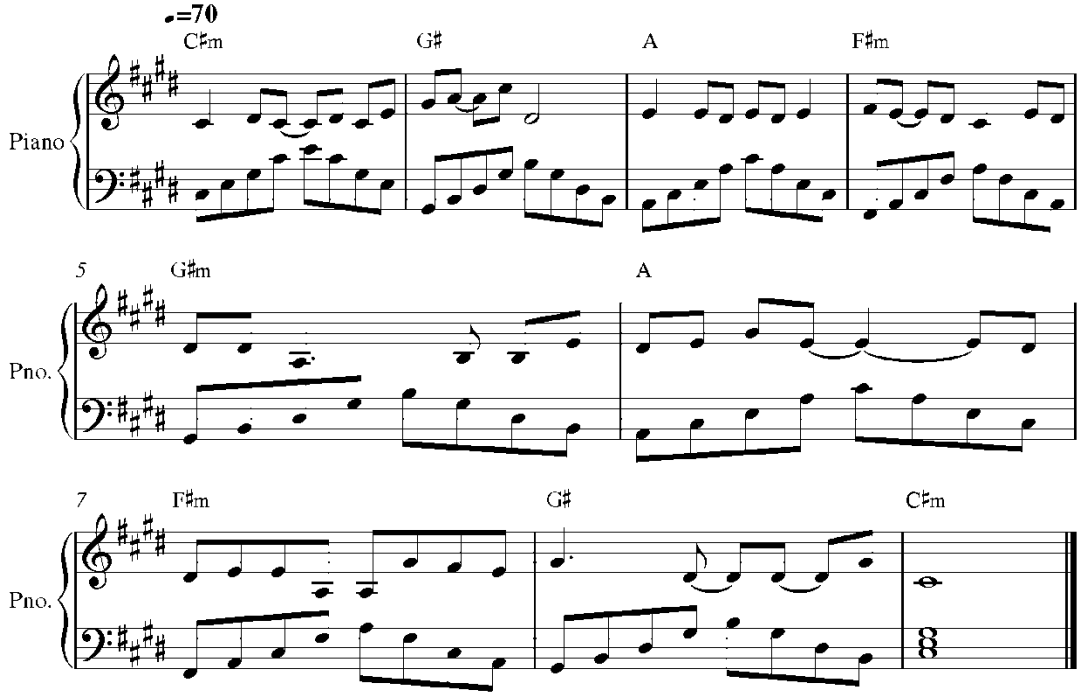


圖 36　自動作曲成果：負向情緒(1)

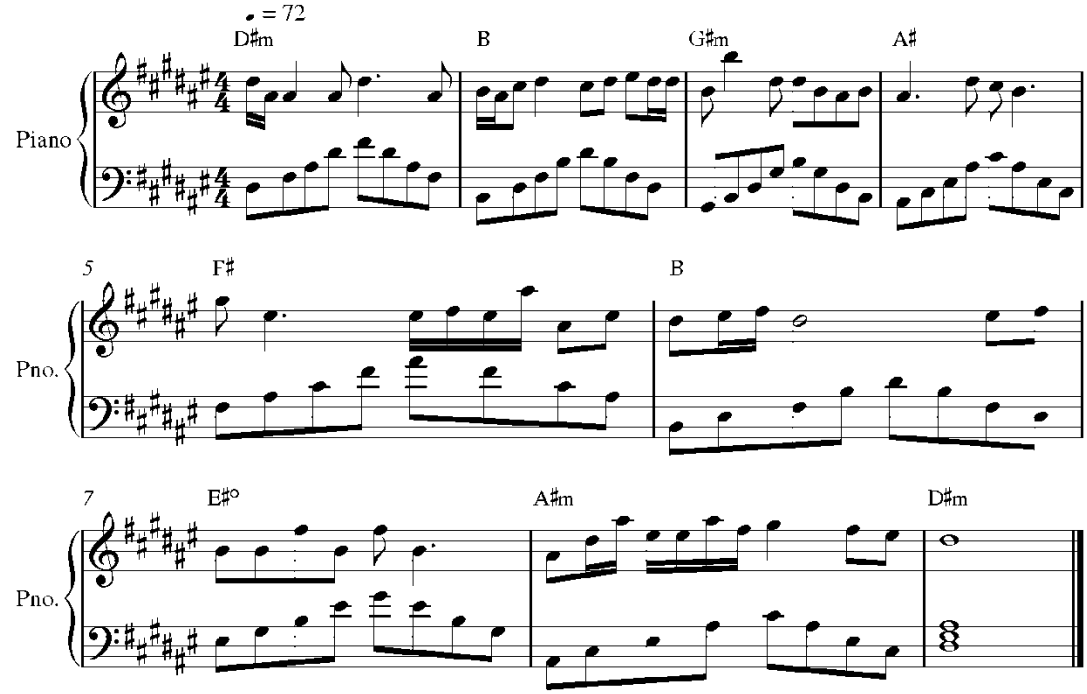


圖 37　自動作曲成果：負向情緒(2)

在描述資訊當中僅標示自動作曲完成的 MusicXML 檔案，倘若有其他連帶產生的合成音樂或是其他產物，亦可依照其特性標示在其他不同層當中。

```
<related_files>
  <related_file file_name=”AC_result_1.xml” file_format=” text_xml”
description=”Result_1” notes=”result” uuid=” 041ef406-8f3d-4796-a3da-e6c42ff3677a” />
  <related_file file_name=” AC_result_2.xml” file_format=” text_xml”
description=”Result_2” notes=”result” uuid=” d5b1be08-97ff-427f-ac93-eaa8574a78bb” />
  <related_file file_name=” AC_result_3.xml” file_format=” text_xml”
description=”Result_3” notes=”result” uuid=” 404dd2f8-1858-4c20-b535-6ca099afe4d9” />
    ….
</related_files>
```

圖 38　系統範例：自動作曲成果描述

在自動作曲完成後，即進入成果分析階段，透過建立音高比例圖、音長比例圖、和弦比例圖，並將其比例描述資訊記載於修正式 IEEE 1599 標準中的結構層。由於透過一階馬可夫鍊針對音樂素材其中的和弦、音高、音長分析成為機率模型後，並依據其模型產生出新的音樂。但是根據[39]的實驗及其統計結果，仍有約 30%的受測者能夠透過同時聆聽電腦所產生的音樂與專家所做曲的音樂，能夠成功分辨出該首音樂是否為電腦所產生的音樂特徵。

因此在音樂特徵當中，仍有其他尚未被探勘出來能夠影響人類心理聽學感知的因素。因此在成果分析階段當中，我們會針對音高與節奏，歸納出具有音樂意義的量化數據，以便在後續應用階段當中能夠更容易探討出影響人類對於音樂感知的因素。

本系統針對自動作曲成果主旋律音高的部分，計算每一個音之間所相差的半音數及其平均音高值，以及最高音跟最低音相差的半音數，並針對主旋律當中最高音出現的次

數以及出現的位置進行統計。主要原因為，樂曲與文章有起承轉合的結構，而最高音則連結了樂曲段的承與轉兩部份，使樂曲具有方向性與整體感。在旋律當中前後音高所相差的半音數能夠得知主旋律級近與跳進的現象。因此，統計出主旋律音高相差的半音數與頂點音的位置具有指標性意義，於後續應用階段當中藉由蒐集回饋內容再與其統計內容進行資料探勘，以便找出其隱含的規則。除了主旋律音高之外，亦針對主旋律音長進行分析，透過統計重拍是否為整數拍，以了解節奏特性。

我們以圖 37 的自動作曲成果為例，進行音高分析後得知最低音高為 64，最高音高為 77，相差 13 個半音，約使用了 1.1 個八度。在比較第 N 個旋律音與第 N+1 個旋律音的音高差異，其結果如圖 40 所示，而其對應的 XML 描述文件如圖 41 所示。經過統計後得知，每個音之間平均相差了 2.14 個半音，標準差為 1.83 個半音。完整的主旋律使用了 38 個音，其中頂點音出現在第 17 個，僅出現一次，出現位置約在整體音樂的 44.7% 長度的地方。在節奏的部分，統計整首音樂每一個音之間相差的拍數及平均拍數，以及每一小節重拍位置的平均拍數進行比較。在成果分析階段所產生的 XML 描述對應之資料模型如圖 39 所示。

透過前述統計過程，音長則能夠依據拍數差異及每小節的重拍是否為整數拍以檢視節奏特性。在音長的部分經過分析後，在比較第 N 個旋律音與第 N+1 個旋律音的拍數差異，其結果如圖 42 所示，其對應 XML 描述如圖 43 所示。經統計後得知，最長的拍數為 2 拍，最短的拍數為 0.5 拍，每個音之間平均相差了 0.3 拍，標準差為 0.42 拍。除此之外，8 個小節中有 16 個重拍，僅有 2 個重拍為非整數拍數，故在節奏織度上較為正規與保守，較無特殊變化。

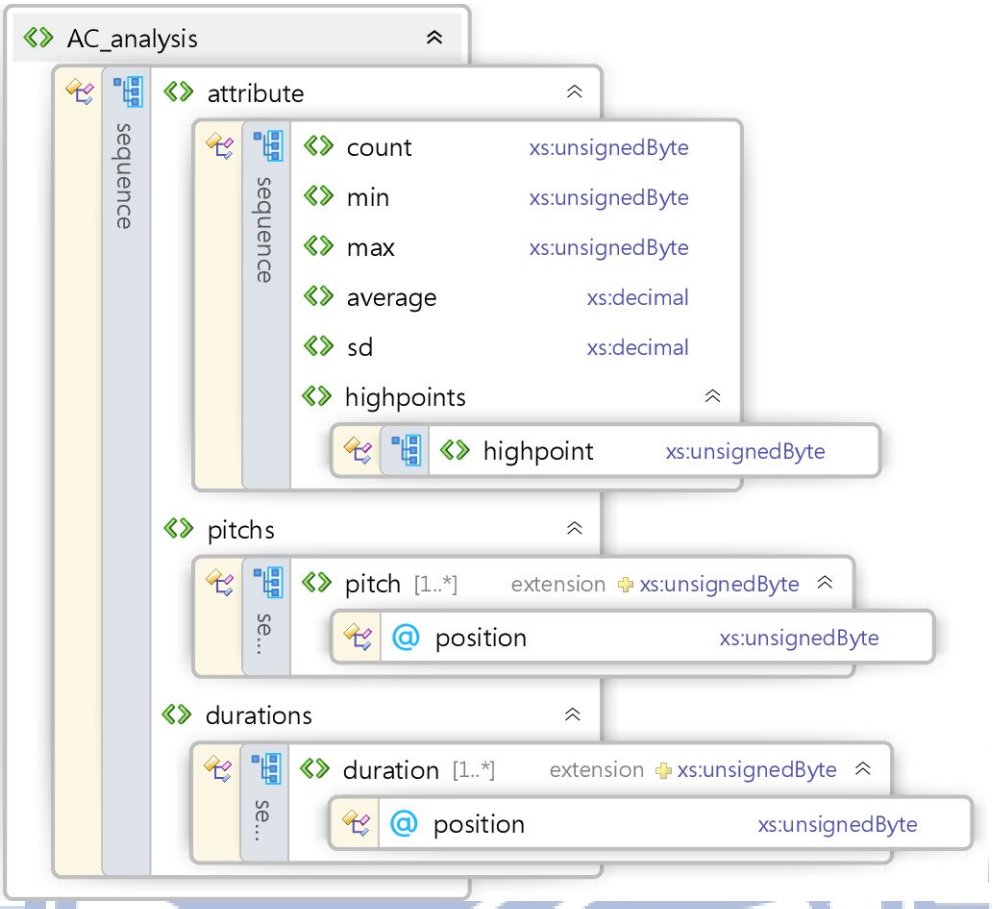


圖 39　成果分析之資料模型

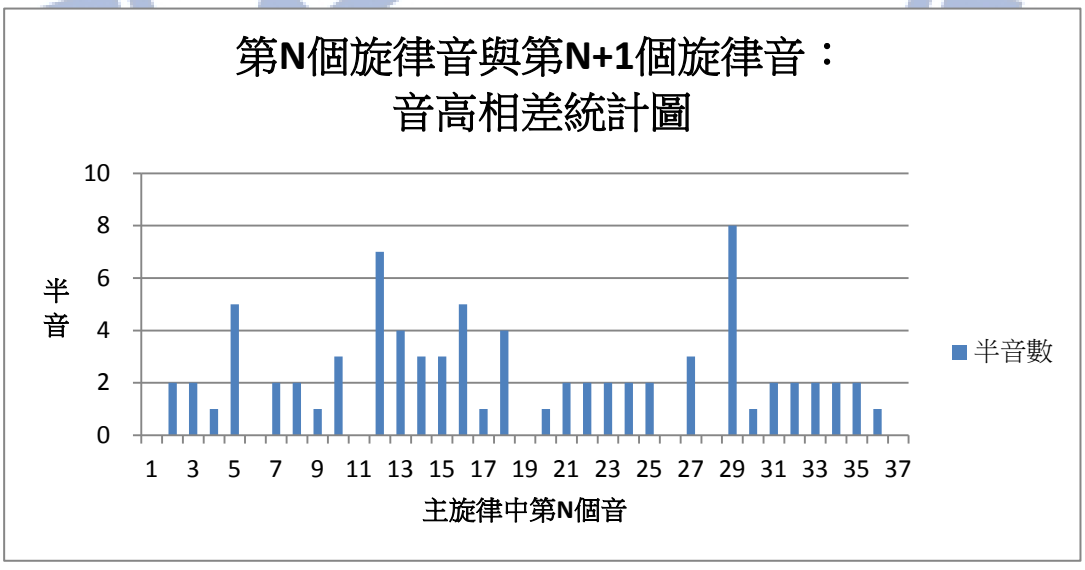


圖 40　第 N 個旋律音與第 N+1 個旋律音：音高相差統計圖

```
<AC_analysis>
  <attribute>
    <count>38</count>
    <min>64</min>
    <max>77</max>
    <average>2.14</average>
    <sd>1.83</sd>
    <highpoints>
      <highpoint>17</highpoint>
    </highpoints>
  </attribute>
  <pitchs>
    <pitch position="1">0</pitch>
    <pitch position="2">2</pitch>
     ……
  </pitchs>
</AC_analysis>
```

圖 41　音高相差統計描述資訊

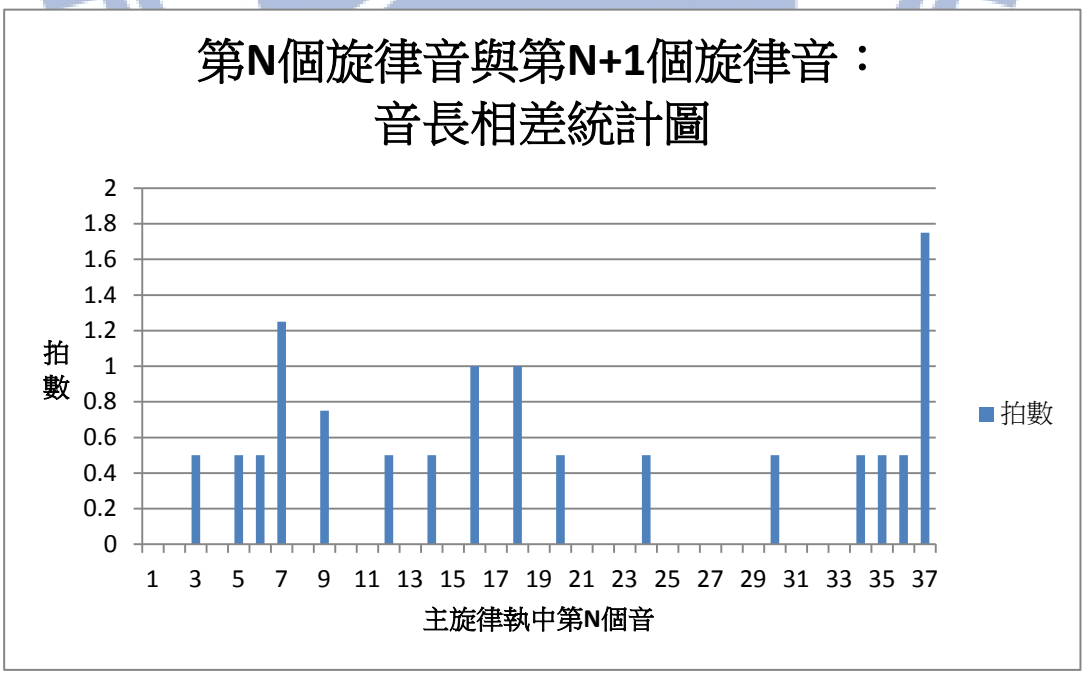


圖 42　第 N 個旋律音與第 N+1 個旋律音：音長相差統計圖

```
<AC_analysis>
  <attribute>
    <count>38</count>
    <min>64</min>
    <max>77</max>
    <average>2.14</average>
    <sd>1.83</sd>
    <highpoints>
      <highpoint>17</highpoint>
    </highpoints>
  </attribute>
  <pitchs>
    ….
  </pitchs>
  <durations>
    <duration position="1">0</duration>
    <duration position="2">0</duration>
    ….
  </durations>
</AC_analysis>
```

圖 43　音長相差統計描述資訊

在完成成果分析後，即可將得到音高與音長的統計描述資訊，記載於修正式 IEEE 1599 標準當中的結構層中，如圖 44 所示。我們即可將所有描述資訊檔案存入對應的資料庫當中，以標準介面方式供程式存取。在最後的應用與評估階段，需求者可以透過 HTTP 方式取得自動作曲的音樂成果及統計資料，並且可以將評價及反饋意見回傳至資料庫當中，廣泛的資訊能夠借助資料探勘技術，協助未來自動作曲演算法的開發，使其成果更臻完美。

```
<structural>
    <analysis>
        <segment>
            <feature_object    name=”AC_ analysis”>
                <file name=”M1_analysis_pitch.xml” notes=” M1” uuid=”
12e536ba-e44c-4caa-ae21-b6d5281d95c6” />
                <file name=”M1_analysis_duration.xml” notes=” M1” uuid=”
7244d117-c4f4-4c50-aa3d-9fe8344b4f90” />
            </feature_object>
        </segment>
    </analysis>
</structural>
```

圖 44　音高與音長的統計描述資訊

第五章、結論與未來方向

本文提出以 IEEE 1599 標準為基礎制訂一個開放式的自動作曲系統架構，並設計了一套適用自動作曲需求的修正式 IEEE 1599 標準，以及記載音樂特徵、成果分析結果的 XML 文件大綱。本系統將目標確立、蒐集相關音樂、MusicXML 曲譜搜尋/轉換、特徵擷取、自動作曲、成果分析、應用與評估等流程當中所產生出的所有描述資訊，存放於原生式 XML 資料庫，以開放式協定提供存取供程式存取。

我們透過本系統架構，以做出有華語流行音樂及快樂情緒的音樂為範例，蒐集了三首音樂之片段並分析其和弦、主旋律音高、主旋律音長的轉換機率，求得一階馬可夫鍊作為自動作曲模型。成果分析則針對自動作曲音樂成果的音高與音長分析，透過觀察主旋律的第 N 個音與第 N+1 個音計算相差之半音數，以得知旋律的級進與跳進現象，並計算主旋律最高音出現的次數及位置，了解旋律的方向性及整體感。在音長方面則計算主旋律的第 N 個音與第 N+1 個音計算相差之拍數，及所有重拍音的平均拍數，以了解其節奏織度。所有成果分析的結果會以 XML 格式記載並標示於修正式 IEEE 1599 標準的整合描述檔中。

本文所提出之架構能夠避免在音樂分析及自動作曲的流程中資料過多，而導致協同研究開發上不清楚彼此之間有何種資料或統計資訊。透過修正式 IEEE 1599 標準使得每一筆產生的資料均能夠確切的被定義與記載，所牽涉的異質資訊都能夠以具有其內涵意義與結構化的的方式存入資料庫，以確保資料的一致性與完整性，更避免浪費資源重複建置相同的資料。

.在未來，使用者或專家能夠針對自動作曲成果反饋意見，或是將其成果修改後再傳回系統，以期能夠透過蒐集大量音樂統計資訊，探勘出尚未發掘對人類心理聽覺感知的元素。本系統未來盼能夠提供對創作有興趣的一般人或是專業作曲家之用，創作出符合音樂理論的作曲，提升電腦自動作曲品質，使其更臻完美。

參考文獻

[1] G. Papadopoulos and G. Wiggins, "AI methods for algorithmic composition: A survey, a critical view and future prospects," 1999, pp. 110-117.

[2] L. A. Hiller and L. A. Isaacson, "Experimental music," ed: McGraw-Hill Book Co New York, 1959.

[3] S. F. Chang, T. Sikora, and A. Purl, "Overview of the MPEG-7 standard," *Circuits and Systems for Video Technology, IEEE Transactions on,* vol. 11, pp. 688-695, 2001.

[4] T. Sikora, "The MPEG-7 Visual standard for content description-an overview," *Circuits and Systems for Video Technology, IEEE Transactions on,* vol. 11, pp. 696-702, 2001.

[5] A. T. Lindsay and J. Herre, "MPEG-7 and MPEG-7 audio-an overview," *JOURNAL-AUDIO ENGINEERING SOCIETY,* vol. 49, pp. 589-594, 2001.

[6] B. Manjunath, P. Salembier, and T. Sikora, *Introduction to MPEG-7: multimedia content description interface* vol. 1: John Wiley & Sons Inc, 2002.

[7] H. G. Kim, N. Moreau, and T. Sikora, *MPEG-7 audio and beyond: Audio content indexing and retrieval*: John Wiley & Sons Inc, 2005.

[8] G. HAUS, "Introduction to the New Standard IEEE 1599: XML Symbols and Layers for Music Encoding," in *Proceddings of the IEEE CS International Comference: The use of symbols to represent music and multimedia objects*, 2008.

[9] L. A. Ludovico, "Key concepts of the IEEE 1599 Standard," in *Proceddings of the IEEE CS International Comference: The use of symbols to represent music and multimedia objects*, 2008.

[10] D. Baggi and G. Haus, "IEEE 1599: Music encoding and interaction," *Computer,* vol. 42, pp. 84-87, 2009.

[11] "IEEE Recommended Practice for Defining a Commonly Acceptable Musical Application Using XML," in *IEEE Standard 1599*, ed, 2008.

[12] E. R. Hammer and M. S. Cole, *Guided Listening: A Textbook for Music Appreciation*: Wm. C. Brown, 1992.

[13] J. O. Smith and S. A. Van Duyne, "Commuted piano synthesis," in *Proceedings of the 1995 International Computer Music Conference, Banff*, 1995, pp. 319-326.

[14] J. Chowning, "The synthesis of complex audio spectra by means of frequency modulation," *J. Audio Eng. Soc,* vol. 21, 1973.

[15] Y.-H. Chen and C.-F. Huang, "Sound Synthesis of the Pipa Based on Computed Timbre Analysis and Physical Modeling," *Selected Topics in Signal Processing, IEEE Journal of,* vol. 5, pp. 1170-1179, 2011.

[16] J. O. Smith III, "Physical modeling using digital waveguides," *Computer Music Journal,* pp. 74-91, 1992.

[17] S. A. Van Duyne and J. O. Smith, "Physical modeling with the 2-D digital waveguide mesh," in *Proceedings of the International Computer Music Conference*, 1993, pp. 40-40.

[18] T. D. Rossing, "Acoustics of percussion instruments: Recent progress," *Acoustical Science and Technology,* vol. 22, pp. 177-188, 2001.

[19] T. D. Rossing, D. S. Hampton, B. E. Richardson, H. J. Sathoff, and A. Lehr, "Vibrational modes of Chinese two-tone bells," *The Journal of the Acoustical Society of America,* vol. 83, p. 369, 1988.

[20] Y.-l. YAN, L.-d. KONG, K.-m. CHAI, and Z.-y. SHENG, "Study on the acoustic characteristics of the ancient Chinese chime-bell [J]," *College Physics,* vol. 2, p. 017, 2004.

[21] W. Dajun, C. Jian, and W. Huijun, "The feature of the dual tones of Chinese music bells," *Mechanics and Its Application,* pp. 12-16, 2003.

[22] M. Karjalainen, V. Välimäki, and Z. Jánosy, "Towards high-quality sound synthesis of the guitar and string instruments," in *Proceedings of the International Computer Music Conference*, 1993, pp. 56-56.

[23] M. Palumbi and L. Seno, "Physical modeling by directly solving wave PDE," in *Proceedings of the International Computer Music Conference (ICMC)*, 1999, pp. 325-328.

[24] G. P. Scavone, "Digital waveguide modeling of the non-linear excitation of single reed woodwind instruments," in *Proc. Int. Computer Music Conference*, 1995.

[25] M. Wertheimer, "Untersuchungen zur Lehre von der Gestalt. II," *Psychological Research,* vol. 4, pp. 301-350, 1923.

[26] D. Temperley, *The cognition of basic musical structures*: The MIT Press, 2001.

[27] H. Schenker, *Der freie Satz*, 1935.

[28] 申克 and 陈世宾, "自由作曲," ed: 北京: 人民音乐出版社, 1997.

[29] F. Lerdahl and R. Jackendoff, *A generative theory of tonal music*: The MIT Press, 1983.

[30] A. Alpern, "Techniques for algorithmic composition of music," *Hampshire College,* 1995.

[31] D. J. Grout and C. V. Palisca, *A history of Western music*: WW Norton & Company, Inc., 1996.

[32] A. Schoenberg, *Style and Idea: Selected Writings*: University of California Press, 1984.

[33] L. A. Hiller and L. M. Isaacson, *Experimental music*: Greenwood Press, 1979.

[34] D. Cope, *Virtual music: computer synthesis of musical style*: The MIT Press, 2004.

[35] D. Cope, M. J. Mayer, and S. C. University of California, *Experiments in musical intelligence* vol. 1: AR Editions Middleton, WI, 1996.

[36] 邱士銓, "Computer Music Composition by Discovered Music Patterns," Master, Department of Computer Science, National Chengchi University, 2005.

[37] K. Jones, "Compositional applications of stochastic processes," *Computer Music Journal,* pp. 45-61, 1981.

[38] C. Ames, "The Markov process as a compositional model: a survey and tutorial," *Leonardo,* pp. 175-187, 1989.

[39] W. Schulze and B. van der Merwe, "Music Generation with Markov Models," *MultiMedia, IEEE,* vol. 18, pp. 78-85, 2011.

[40] E. R. Miranda, *Composing music with computers* vol. 1: Focal Pr, 2001.

[41] K. Ebcioğlu, "An expert system for harmonizing four-part chorales," *Computer Music Journal,* vol. 12, pp. 43-51, 1988.

[42] M. J. Steedman, "A generative grammar for jazz chord sequences," *Music Perception,* pp. 52-77, 1984.

[43] M. Steedman, "The blues and the abstract truth: Music and mental models," *Mental Models in Cognitive Science,* pp. 305-318, 1996.

[44] M. J. Steedman, "Grammar, interpretation, and processing from the lexicon," in *Lexical representation and process*, 1989, pp. 463-504.

[45] D. Cope, "Computer modeling of musical intelligence in EMI," *Computer Music Journal,* pp. 69-83, 1992.

[46] G. Wiggins, G. Papadopoulos, S. Phon-Amnuaisuk, and A. Tuson, "Evolutionary methods for musical composition," *DAI RESEARCH PAPER,* 1998.

[47] P. M. Todd, "A connectionist approach to algorithmic composition," *Computer Music Journal,* vol. 13, pp. 27-43, 1989.

[48] M. C. Mozer, "Connectionist music composition based on melodic, stylistic and psychophysical constraints," *Music and connectionism,* pp. 195-211, 1991.

[49] P. Gibson and J. Byrne, "Neurogen, musical composition using genetic algorithms and cooperating neural networks," in *Artificial Neural Networks, 1991., Second International Conference on*, 1991, pp. 309-313.

[50] P. N. Juslin and J. A. Sloboda, *Music and emotion* vol. 315: Oxford University Press New York, 2001.

[51] C. L. Krumhansl, "An exploratory study of musical emotions and psychophysiology," *Canadian journal of experimental psychology,* vol. 51, pp. 336-353, 1997.

[52] I. Cross, "Music and science: three views," *Revue belge de Musicologie/Belgisch Tijdschrift voor Muziekwetenschap,* pp. 207-214, 1998.

[53] B. Johanson and R. Poli, "GP-music: An interactive genetic programming system for music generation with automated fitness raters," in *Genetic Programming 1998: Proceedings of the Third Annual Conference*, 1998, pp. 181-186.

[54] M. Unehara and T. Onisawa, "Composition of music using human evaluation," in *Fuzzy Systems, 2001. The 10th IEEE International Conference on*, 2001, pp. 1203-1206.

[55] J. A. Biles, "Life with GenJam: Interacting with a musical IGA," in *Systems, Man, and*

Cybernetics, 1999. IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on, 1999, pp. 652-656.

[56] H. Hild, J. Feulner, and W. Menzel, "HARMONET: A Neural Net for Harmonizing Chorales in the Style of J. S. Bach," presented at the NIPS, 1991.

[57] M. Pearce and G. Wiggins, "Towards a framework for the evaluation of machine compositions."

[58] A. R. de Freitas, F. G. Guimaraes, and R. V. Barbosa, "Ideas in automatic evaluation methods for melodies in algorithmic composition," 2012.

[59] V. C. Wilfred, *實驗審美心理學(下)〈音樂、詩歌篇〉*: 台北: 商鼎, 1991.

[60] 莊婕筠, *音樂治療*: 台北：心理, 2004.

[61] G. Husain, W. F. Thompson, and E. G. Schellenberg, "Effects of musical tempo and mode on arousal, mood, and spatial abilities," *Music Perception,* vol. 20, pp. 151-171, 2002.

[62] J. A. Russell, "A circumplex model of affect," *Journal of personality and social psychology,* vol. 39, pp. 1161-1178, 1980.

[63] S. R. Livingstone and A. R. Brown, "Dynamic response: real-time adaptation for music emotion," presented at the Proceedings of the second Australasian conference on Interactive entertainment, Sydney, Australia, 2005.

[64] E. Selfridge-Field, *Beyond MIDI: the handbook of musical codes*: The MIT Press, 1997.

[65] G. Haus and M. Longari, "A multi-layered, timebased music description approach based on xml," *Computer Music Journal,* vol. 29, pp. 70-85, 2005.

[66] D. Sloan, "Aspects of Music Representation in HyTime/SMDL," *Computer Music Journal,* vol. 17, pp. 51-59, 1993.

附錄一、修正式 IEEE 1599 XML 描述檔案 DTD

Unmodified Added Removed

```
<?xml version="1.0" encoding="UTF-8"?>

<!--
 File name:                ieee1599.dtd
 Version:          1.0
 Creation date:            25/02/2007
- Last update:              08/04/2007
+ Last update:              06/01/2013
+ Notes:      modified ieee1599 based on AC purposes.
 Description
 IEEE 1599 format is an XML-based language aimed at a comprehensive description of music and
music-related contents.
 It has been designed in response to IEEE Std 1599 - IEEE Recommended Practice for Defining a Commonly
Acceptable Musical Application Using XML Language.

-->

<!--=========================================================================
====================-->

<!-- Import of external DTDs -->

<!ENTITY % svg
  PUBLIC "-//W3C//DTD SVG 1.1//EN"
  "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
%svg;

<!ENTITY % ChannelRequired "#REQUIRED">
<!ENTITY % mididtd
  PUBLIC  "-//MIDI Manufacturers Association//DTD MIDIEvents 1.0//EN"
  "http://www.midi.org/dtds/MIDIEvents10.dtd" >
%mididtd;
```

```
<!--============================================================================================
====================-->

<!-- Common attributes parameter entities -->

<!ENTITY % spine_ref
     "event_ref IDREF #REQUIRED">

<!ENTITY % spine_start_end_ref
     "start_event_ref IDREF #REQUIRED
   end_event_ref IDREF #REQUIRED">

<!ENTITY % accidental
     "(none | double_flat | flat_and_a_half | flat | demiflat | natural | demisharp | sharp |
sharp_and_a_half | double_sharp)">

<!ENTITY % formats
     "(application_excel | application_mac-binhex40 | application_msword |
application_octet-stream | application_pdf | application_x-director | application_x-gzip |
application_x-javascript | application_x-macbinary | application_x-pn-realaudio |
application_x-shockwave_flash | application_x-tar | application_zip | audio_aiff | audio_avi |
audio_mp3 | audio_mpeg | audio_mpeg3 | audio_mpg | audio_wav | audio_x_aiff | audio_x_midi |
audio_x_wav | audio_x-mp3 | audio_x-mpeg | audio_x-mpeg3 | audio_x-mpegaudio | audio_x-mpg |
audio_x-ms-wma | image_avi | image_bmp | image_x-bmp | image_x-bitmap | image_x-xbitmap |
image_x-win-bitmap | image_x-windows-bmp | image_ms-bmp | image_x-ms-bmp | application_bmp |
application_x-bmp | application_x-win-bitmap | application_preview | image_gif | image_jpeg |
image_pict | image_png | application_png | application_x-png | image_tiff | text_html |
text_plain_application_postscript | video_avi | video_mpeg | video_msvideo | video_quicktime |
video_x-msvideo | video_x-ms-wmv | video_x-qtc | video_xmpg2)">


<!-- Common Elements -->

<!ELEMENT rights EMPTY>
<!ATTLIST rights
  file_name CDATA #REQUIRED>

<!--============================================================================================
```

```
====================-->

<!-- Root Element -->

<!ELEMENT ieee1599 (general, logic, structural?, notational?, performance?, audio?)>
<!ATTLIST ieee1599
     version CDATA #REQUIRED
     creator CDATA #IMPLIED>


<!--==========================================================================
====================-->

<!-- General Layer -->

+ <!ELEMENT general (ac?, description, related_files?, analog_media?, notes?)>

+ <!ELEMENT ac (target)>

+ <!ELEMENT target (ac_genres, ac_emotions?, descriptions?)>

+ <!ELEMENT ac_genres (genre*) >

+ <!ATTLIST genre
+   name CDATA #REQUIRED
+   description CDATA #IMPLIED
+   weight CDATA #IMPLIED>

+ <!ELEMENT ac_emotions (emotion*) >

+ <!ELEMENT emotion EMPTY >
+ <!ATTLIST emotion
+   name CDATA #IMPLIED
+   weight CDATA #IMPLIED
+ >

+ <!ELEMENT descriptions (description*) >
```

```
<!ELEMENT description (main_title, author*, other_title*, number?, work_title?, work_number?, date*,
genres?)>

<!ELEMENT main_title (#PCDATA)>

<!ELEMENT author (#PCDATA)>
<!ATTLIST author
     type CDATA #IMPLIED>

<!ELEMENT other_title (#PCDATA)>

<!ELEMENT number (#PCDATA)>

<!ELEMENT work_title (#PCDATA)>

<!ELEMENT work_number (#PCDATA)>

<!ELEMENT date (#PCDATA)>
<!ATTLIST date
     type CDATA #IMPLIED>

<!ELEMENT genres (genre+)>

<!ELEMENT genre EMPTY>
<!ATTLIST genre
     name CDATA #REQUIRED
     description CDATA #IMPLIED
     weight CDATA #IMPLIED>

<!ELEMENT related_files (related_file+)>

<!ELEMENT related_file EMPTY>
<!ATTLIST related_file
     file_name CDATA #REQUIRED
     file_format %formats; #REQUIRED
     encoding_format %formats; #REQUIRED
     start_event_ref IDREF #IMPLIED
     end_event_ref IDREF #IMPLIED
     description CDATA #IMPLIED
```

```
      copyright CDATA #IMPLIED
      notes CDATA #IMPLIED
      + uuid  CDATA #REQUIRED
>

<!ELEMENT analog_media (analog_medium+)>

<!ELEMENT analog_medium EMPTY>
<!ATTLIST analog_medium
      description CDATA #REQUIRED
      copyright CDATA #IMPLIED
      notes CDATA #IMPLIED>

<!ELEMENT notes (#PCDATA)>


<!--=========================================================================
====================-->

<!-- Logic Layer -->

<!ELEMENT logic (spine, los?, layout?)>


<!-- Spine -->
<!ELEMENT spine (event)+>

<!ELEMENT event EMPTY>
<!ATTLIST event
      id ID #REQUIRED
      timing CDATA "null"
      hpos CDATA "null">

<!ELEMENT los (agogics*, text_field*, metronomic_indication*, staff_list, part+,
horizontal_symbols?, ornaments?, lyrics*)>

<!ELEMENT agogics (#PCDATA)>
<!ATTLIST agogics
      bracketed (no | yes) #IMPLIED
```

```
      %spine_ref;>

<!ELEMENT text_field (#PCDATA)>
<!ATTLIST text_field
      extension_line_to IDREF #IMPLIED
      extension_line_shape (normal | dotted | dashed) #IMPLIED
      %spine_ref;>

<!ELEMENT metronomic_indication EMPTY>
<!ATTLIST metronomic_indication
      num CDATA #REQUIRED
      den CDATA #REQUIRED
      dots CDATA #IMPLIED
      value CDATA #REQUIRED
      %spine_ref;>

<!ELEMENT staff_list (brackets | staff)+>

<!ELEMENT brackets EMPTY>
<!ATTLIST brackets
      marker (start_of_staff_group | end_of_staff_group) #REQUIRED
      group_number CDATA #REQUIRED
      shape (square | rounded_square | brace | vertical_bar | none) #REQUIRED>

<!ELEMENT staff (clef | ( key_signature | custom_key_signature) | time_signature | barline |
tablature_tuning)*>
<!ATTLIST staff
      id ID #REQUIRED
      line_number CDATA "5"
      ossia (yes | no) "no"
      tablature (none | french | german | italian) #IMPLIED>

<!ELEMENT clef EMPTY>
<!ATTLIST clef
      shape (G | F | C | gregorian_F | gregorian_C | percussion | doubleG | tabguitar) #REQUIRED
      staff_step CDATA #REQUIRED
      octave_num (0 | 8 | -8 | 15 | -15) "0"
      %spine_ref;>
```

```
<!ELEMENT key_signature (sharp_num | flat_num) >
<!ATTLIST key_signature
     %spine_ref;>

<!ELEMENT sharp_num EMPTY>
<!ATTLIST sharp_num
     number (0 | 1 | 2 | 3 | 4 | 5 | 6 | 7) #REQUIRED>

<!ELEMENT flat_num EMPTY>
<!ATTLIST flat_num
     number (0 | 1 | 2 | 3 | 4 | 5 | 6 | 7) #REQUIRED>

<!ELEMENT custom_key_signature (key_accidental)+>
<!ATTLIST custom_key_signature
     %spine_ref;>

<!ELEMENT key_accidental EMPTY>
<!ATTLIST key_accidental
     step (A | B | C | D | E | F | G) #REQUIRED
     accidental %accidental; "none">

<!ELEMENT time_signature (time_indication)+>
<!ATTLIST time_signature
     visible (yes | no) "yes"
     %spine_ref;>

<!ELEMENT time_indication EMPTY>
<!ATTLIST time_indication
     num CDATA #REQUIRED
     den CDATA #IMPLIED
     abbreviation (yes | no) "no"
     vtu_amount CDATA #IMPLIED>

<!ELEMENT barline EMPTY>
<!ATTLIST barline
     style (dashed | double | final | invisible | standard | medium | short) #REQUIRED
     extension (single_staff | staff_group | all_staves | mensurstrich) #REQUIRED
     %spine_ref;>
```

```
<!ELEMENT tablature_tuning (string*)>
<!ATTLIST tablature_tuning
     type (D | E | G | A | baroque | flat_french | other) #IMPLIED>

<!ELEMENT string EMPTY>
<!ATTLIST string
     string_number CDATA #REQUIRED
     string_pitch (A | B | C | D | E | F | G) #REQUIRED
     string_accidental %accidental; #IMPLIED
     string_octave CDATA #REQUIRED>

<!ELEMENT part (voice_list, measure+)>
<!ATTLIST part
     id ID #REQUIRED
     performers_number CDATA "unknown"
    transposition_pitch (A | B | C | D | E | F | G) #IMPLIED
    transposition_accidental %accidental; #IMPLIED
     octave_offset CDATA #IMPLIED>

<!ELEMENT voice_list (voice_item+)>

<!ELEMENT voice_item EMPTY>
<!ATTLIST voice_item
     id ID #REQUIRED
     staff_ref IDREF #REQUIRED
     notation_style (normal | rhythmic | slash | blank) #IMPLIED>

<!ELEMENT measure (voice+ | multiple_rest | measure_repeat?)>
<!ATTLIST measure
     number CDATA #REQUIRED
     id ID #IMPLIED
     show_number (yes | no) #IMPLIED
     numbering_style (arabic_numbers | roman_numbers | small_letters | capital_letters) #IMPLIED>

<!ELEMENT multiple_rest EMPTY>
<!ATTLIST multiple_rest
     number_of_measures CDATA #REQUIRED
  event_ref IDREF #IMPLIED>
```

```
<!ELEMENT measure_repeat EMPTY>
<!ATTLIST measure_repeat
     number_of_measures CDATA #REQUIRED
  event_ref IDREF #IMPLIED>

<!ELEMENT voice (chord | rest | tablature_symbol | gregorian_symbol)+>
<!ATTLIST voice
     voice_item_ref IDREF #REQUIRED
     ossia (yes | no) "no">

<!ELEMENT chord (duration, augmentation_dots?, (notehead+ | repetition), articulation?)>
<!ATTLIST chord
     id ID #IMPLIED
     %spine_ref;
     stem_direction (up | down | none) #IMPLIED
     beam_before (yes | no) "no"
     beam_after (yes | no) "no"
     cue (yes | no) "no"
     tremolo_lines (no | 1 | 2 | 3 | 4 | 5 | 6) #IMPLIED>

<!ELEMENT repetition EMPTY>

<!ELEMENT duration (tuplet_ratio?)>
<!ATTLIST duration
     num CDATA #REQUIRED
     den CDATA #REQUIRED>

<!ELEMENT tuplet_ratio (tuplet_ratio*)>
<!ATTLIST tuplet_ratio
     enter_num CDATA #REQUIRED
     enter_den CDATA #REQUIRED
     enter_dots CDATA #IMPLIED
     in_num CDATA #REQUIRED
     in_den CDATA #REQUIRED
     in_dots CDATA #IMPLIED>

<!ELEMENT rest (duration, augmentation_dots?)>
<!ATTLIST rest
     id CDATA #IMPLIED
```

```
      %spine_ref;
      staff_ref IDREF #IMPLIED
      hidden (no | yes) #IMPLIED>

<!ELEMENT tablature_symbol (duration, augmentation_dots?, key+)>
<!ATTLIST tablature_symbol
      id ID #IMPLIED
      %spine_ref;
      stem_direction (up | down | none) #IMPLIED
      beam_before (yes | no) "no"
      beam_after (yes | no) "no">

<!ELEMENT key (tablature_pitch, tablature_articulation?, tie?, tablature_fingering?)>
<!ATTLIST key
      id ID #IMPLIED
      staff_ref IDREF #IMPLIED>

<!ELEMENT tablature_pitch EMPTY>
<!ATTLIST tablature_pitch
      string_number CDATA #IMPLIED
      key_number CDATA #REQUIRED>

<!ELEMENT tablature_articulation EMPTY>
<!ATTLIST tablature_articulation
      shape (cross | tie | other) #REQUIRED>

<!ELEMENT tablature_fingering (#PCDATA)>
<!ATTLIST tablature_fingering
      shape (number | dot | other) #REQUIRED>

<!ELEMENT gregorian_symbol (notehead+)>
<!ATTLIST gregorian_symbol
      id ID #IMPLIED
      neume (punctum | virga | punctum_inclinatum | quilisma | apostrofa | oriscus | podatus | pes
| clivis | flexa | epiphonus | cephalicus | bistropha | bivirga | trigon | torculus | porrectus |
scandicus | salicus | climacus | tristropha | trivirga | strophicus | pressus | custos) #REQUIRED
      inflexion (no | resupinus | flexus) "no"
      subpunctis (no | praepunctis | subpunctis | subbipunctis | subtripunctis | subquadripunctis
| subquinquipunctis) "no"
```

```
      interpretative_mark (no | vertical_episema | horizontal_episema | liquescens) "no"
      mora (yes | no) "no"
      %spine_ref;>

<!-- Articulation signs -->

<!ELEMENT articulation (normal_accent | staccatissimo | staccato | strong_accent | tenuto |
stopped_note | snap_pizzicato | natural_harmonic | up_bow | down_bow | open_mute | close_mute |
custom_articulation)*>

<!ELEMENT normal_accent EMPTY>

<!ELEMENT staccatissimo EMPTY>

<!ELEMENT staccato EMPTY>

<!ELEMENT strong_accent EMPTY>

<!ELEMENT tenuto EMPTY>

<!ELEMENT stopped_note EMPTY>

<!ELEMENT snap_pizzicato EMPTY>

<!ELEMENT natural_harmonic EMPTY>

<!ELEMENT up_bow EMPTY>

<!ELEMENT down_bow EMPTY>

<!ELEMENT open_mute EMPTY>

<!ELEMENT close_mute EMPTY>

<!ELEMENT custom_articulation (svg)>

<!ELEMENT notehead_ref EMPTY>
<!ATTLIST notehead_ref
      %spine_ref;>
```

```
<!ELEMENT notehead (pitch, printed_accidentals?, tie?, fingering?)>
<!ATTLIST notehead
  id ID #IMPLIED
  staff_ref IDREF #IMPLIED
  style (normal | harmonic | unpitched | cymbal | parenthesis | circled | squared) #IMPLIED>

<!ELEMENT pitch EMPTY>
<!ATTLIST pitch
     step (A | B | C | D | E | F | G | none) #REQUIRED
     octave CDATA #REQUIRED
     actual_accidental %accidental; #IMPLIED>

<!ELEMENT printed_accidentals (double_flat | flat_and_a_half | flat | demiflat | natural | demisharp
| sharp | sharp_and_a_half | double_sharp)+>
<!ATTLIST printed_accidentals
     shape (normal | small | bracketed) "normal">

<!ELEMENT tie EMPTY>

<!ELEMENT fingering EMPTY>
<!ATTLIST fingering
     number (1 | 2 | 3 | 4 | 5) #REQUIRED>

<!ELEMENT double_flat EMPTY>
<!ATTLIST double_flat
     parenthesis (yes | no) "no">

<!ELEMENT flat_and_a_half EMPTY>
<!ATTLIST flat_and_a_half
     parenthesis (yes | no) "no">

<!ELEMENT flat EMPTY>
<!ATTLIST flat
     parenthesis (yes | no) "no">

<!ELEMENT demiflat EMPTY>
<!ATTLIST demiflat
     parenthesis (yes | no) "no">
```

```
<!ELEMENT natural EMPTY>
<!ATTLIST natural
     parenthesis (yes | no) "no">

<!ELEMENT demisharp EMPTY>
<!ATTLIST demisharp
     parenthesis (yes | no) "no">

<!ELEMENT sharp EMPTY>
<!ATTLIST sharp
     parenthesis (yes | no) "no">

<!ELEMENT sharp_and_a_half EMPTY>
<!ATTLIST sharp_and_a_half
     parenthesis (yes | no) "no">

<!ELEMENT double_sharp EMPTY>
<!ATTLIST double_sharp
     parenthesis (yes | no) "no">

<!ELEMENT augmentation_dots EMPTY>
<!ATTLIST augmentation_dots
     number CDATA "1">

<!ELEMENT lyrics (syllable+)>
<!ATTLIST lyrics
     part_ref IDREF #REQUIRED
     voice_ref IDREF #REQUIRED>

<!ELEMENT syllable (#PCDATA)>
<!ATTLIST syllable
     start_event_ref IDREF #REQUIRED
     end_event_ref IDREF #IMPLIED
     hyphen (yes | no) "no">

<!ELEMENT horizontal_symbols (arpeggio | bend | breath_mark | chord_symbol | dynamic | fermata |
glissando | hairpin | octave_bracket | pedal_start | pedal_end | percussion_beater |
percussion_special | slur | special_beam | tablature_hsymbol | repeat | coda | segno | fine |
```

```
multiple_ending | custom_hsymbol)*>

<!ELEMENT arpeggio (notehead_ref+)>
<!ATTLIST arpeggio
     shape (wavy | line | no_arpeggio) #REQUIRED
     direction (up | down) "down">

<!ELEMENT bend EMPTY>
<!ATTLIST bend
  id ID #IMPLIED
     %spine_ref;
  type (single | double) "single"
     to_pitch (A | B | C | D | E | F | G | up | down) #REQUIRED
     to_accidental %accidental; #IMPLIED
  to_octave CDATA #IMPLIED>

<!ELEMENT breath_mark EMPTY>
<!ATTLIST breath_mark
     id CDATA #IMPLIED
     type (comma | caesura) #REQUIRED
     staff_ref IDREF #IMPLIED
     %spine_start_end_ref;>

<!ELEMENT chord_symbol (#PCDATA)>
<!ATTLIST chord_symbol
     id CDATA #IMPLIED
     %spine_ref;>

<!ELEMENT dynamic (#PCDATA)>
<!ATTLIST dynamic
     id CDATA #IMPLIED
  extension_line_to IDREF #IMPLIED
  extension_line_shape (normal | dotted | dashed) #IMPLIED
     staff_ref IDREF #IMPLIED
     %spine_ref;>

<!ELEMENT fermata (#PCDATA)>
<!ATTLIST fermata
     id ID #IMPLIED
```

```
      %spine_ref;>

<!ELEMENT glissando EMPTY>
<!ATTLIST glissando
  id ID #IMPLIED
      start_event_ref IDREF #REQUIRED
      end_event_ref IDREF #IMPLIED>

<!ELEMENT hairpin EMPTY>
<!ATTLIST hairpin
      id CDATA #IMPLIED
      type (crescendo | diminuendo) #REQUIRED
      staff_ref IDREF #IMPLIED
      %spine_start_end_ref;>

<!ELEMENT octave_bracket EMPTY>
<!ATTLIST octave_bracket
      id CDATA #IMPLIED
      type (8va | 8vb | 15ma | 15mb) #REQUIRED
      staff_ref IDREF #REQUIRED
      %spine_start_end_ref;>

<!ELEMENT pedal_start EMPTY>
<!ATTLIST pedal_start
      id ID #IMPLIED
      %spine_ref;>

<!ELEMENT pedal_end EMPTY>
<!ATTLIST pedal_end
      id ID #IMPLIED
      %spine_ref;>

<!ELEMENT percussion_beater (#PCDATA)>
<!ATTLIST percussion_beater
  id CDATA #IMPLIED
  type (bow | snare_stick | snare_stick_plastic | spoon_shaped | guiro | triangle | knitting_needle
| hand | hammer | metal_hammer | wooden_timpani_mallet | light_timpani_mallet |
medium_timpani_mallet | heavy_timpani_mallet | light_vibraphone_mallet | medium_vibraphone_mallet
| heavy_vibraphone_mallet | light_bassdrum_mallet | medium_bassdrum_mallet | heavy_bassdrum_mallet
```

```
| steel_core_bassdrum_mallet | coin | brush | nails) #REQUIRED
  start_event_ref IDREF #REQUIRED
  end_event_ref IDREF #IMPLIED>

<!ELEMENT percussion_special (#PCDATA)>
<!ATTLIST percussion_special
  id CDATA #IMPLIED
  type (play_on_border | stop_drumhead | muffle_with_harmonics | muffle | rub | shake) #REQUIRED
  %spine_ref;>

<!ELEMENT slur (svg?)>
<!ATTLIST slur
  id ID #IMPLIED
     %spine_start_end_ref;
     shape (normal | dashed | dotted) "normal"
     bracketed (no | yes) "no">

<!ELEMENT special_beam (notehead_ref+)>
<!ATTLIST special_beam
     id CDATA #IMPLIED
     fanned_from CDATA #IMPLIED
  fanned_to CDATA #IMPLIED>

<!ELEMENT tablature_hsymbol (tablature_element | barre)+>
<!ATTLIST tablature_hsymbol
     id CDATA #IMPLIED
     %spine_ref;
  string_number CDATA #REQUIRED
  start_fret CDATA #REQUIRED
  fret_number CDATA #REQUIRED>

<!ELEMENT tablature_element EMPTY>
<!ATTLIST tablature_element
     shape (empty_circle | full_circle | cross | rhombus | 1 | 2 | 3 | 4 | T) #REQUIRED
  string_position CDATA #REQUIRED
  fret_position CDATA #REQUIRED>

<!ELEMENT barre EMPTY>
<!ATTLIST barre
```

```
      start_string_position CDATA #REQUIRED
    end_string_position CDATA #REQUIRED
    fret_position CDATA #REQUIRED>

<!ELEMENT repeat (repeat_text?, (jump_to, end?)+)>
<!ATTLIST repeat
      id ID #IMPLIED
      %spine_ref;>

<!ELEMENT repeat_text (#PCDATA)>

<!ELEMENT jump_to EMPTY>
<!ATTLIST jump_to
      id ID #IMPLIED
      %spine_ref;>

<!ELEMENT end EMPTY>
<!ATTLIST end
      id CDATA #IMPLIED
      %spine_ref;>

<!ELEMENT segno (#PCDATA)>
<!ATTLIST segno
      id ID #IMPLIED
      %spine_ref;>

<!ELEMENT coda (#PCDATA)>
<!ATTLIST coda
      id ID #IMPLIED
      %spine_ref;>

<!ELEMENT fine (#PCDATA)>
<!ATTLIST fine
      id ID #IMPLIED
      %spine_ref;>

<!ELEMENT multiple_endings (multiple_ending+)>
<!ATTLIST multiple_ending
      id ID #IMPLIED>
```

```
<!ELEMENT multiple_ending EMPTY>
<!ATTLIST multiple_ending
     id ID #IMPLIED
  number CDATA #REQUIRED
  %spine_start_end_ref;
  return_to IDREF #IMPLIED>

<!ELEMENT custom_hsymbol (svg)>
<!ATTLIST custom_hsymbol
  id ID #IMPLIED
     start_event_ref IDREF #REQUIRED
     end_event_ref IDREF #IMPLIED>

<!ELEMENT ornaments (acciaccatura | baroque_acciaccatura | appoggiatura | baroque_appoggiatura |
mordent | tremolo | trill | turn)*>

<!ELEMENT acciaccatura (chord+)>
<!ATTLIST acciaccatura
  id ID #IMPLIED
     %spine_ref;
     slur (yes | no) "no">

<!ELEMENT baroque_acciaccatura EMPTY>
<!ATTLIST baroque_acciaccatura
  id ID #IMPLIED
     %spine_ref;
     style (vertical_turn | mordent | flatte | tierce_coulee | slash | backslash) #REQUIRED>

<!ELEMENT appoggiatura (chord+)>
<!ATTLIST appoggiatura
  id ID #IMPLIED
     %spine_ref;
     slur (yes | no) "no">

<!ELEMENT baroque_appoggiatura EMPTY>
<!ATTLIST baroque_appoggiatura
  id ID #IMPLIED
     %spine_ref;
```

```
      style (hairpin | plus | slash | backslash | pipe | double_slur | up_hook | down_hook) #REQUIRED>

<!ELEMENT mordent EMPTY>
<!ATTLIST mordent
  id ID #IMPLIED
     %spine_ref;
     type (upper | lower) "upper"
     length (normal | double) "normal"
     accidental %accidental; "none"
     style (normal | up_hook | down_hook) "normal">

<!ELEMENT tremolo EMPTY>
<!ATTLIST tremolo
  id ID #IMPLIED
  %spine_start_end_ref;
     tremolo_lines (1 | 2 | 3 | 4 | 5 | 6) #REQUIRED>

<!ELEMENT trill EMPTY>
<!ATTLIST trill
  id ID #IMPLIED
     %spine_ref;
     accidental %accidental; "none"
     style (t | tr | tr- | plus | double_slash | caesura_double_slash | slur_double_slash | mordent
| double_mordent) #IMPLIED
     start_hook (none | up | down) "none"
     end_hook (none | up | down) "none">

<!ELEMENT turn EMPTY>
<!ATTLIST turn
  id ID #IMPLIED
     %spine_ref;
     type (over | after) #REQUIRED
     style (normal | inverted | cut | vertical) #REQUIRED
     upper_accidental %accidental; "none"
     lower_accidental %accidental; "none">

<!-- Layout -->

<!ELEMENT layout (page+, text_font?, music_font?)>
```

```
<!ATTLIST layout
      hpos_per_unit CDATA #REQUIRED
      measurement_unit (centimeters | millimeters | inches | decimal_inches | points | picas | pixels
| twips) #REQUIRED>

<!ELEMENT page ((standard_page_format | custom_page_format), layout_system*, layout_images*,
layout_shapes*)>
<!ATTLIST page
      id CDATA #REQUIRED
      number CDATA #IMPLIED>

<!ELEMENT standard_page_format EMPTY>
<!ATTLIST standard_page_format
  format (a0 | a1 | a2 | a3 | a4 | a5 | a6 | a7 | a8 | b0 | b1 | b2 | b3 | b4 | b5 | b6 | b7 | b8
| c0 | c1 | c2 | c3 | c4 | c5 | c6 | c7 | c8 | ansi_a | ansi_b | ansi_c | ansi_d | ansi_e | arch_a
| arch_b | arch_c | arch_e | arch_e1 | quarto | foolscap | executive | monarch | government_letter
| letter | legal | ledger | tabloid | post | crown | large_post | demy | medium | royal | elephant
| double_demy | quad_demy | statement) #REQUIRED>

<!ELEMENT custom_page_format EMPTY>
<!ATTLIST custom_page_format
      width CDATA #REQUIRED
      height CDATA #REQUIRED>

<!ELEMENT layout_system (layout_staff+)>
<!ATTLIST layout_system
      id CDATA #IMPLIED
      upper_left_x CDATA #REQUIRED
      upper_left_y CDATA #REQUIRED
      lower_right_x CDATA #REQUIRED
      lower_right_y CDATA #REQUIRED>

<!ELEMENT layout_staff EMPTY>
<!ATTLIST layout_staff
      id ID #IMPLIED
      staff_ref CDATA #REQUIRED
      vertical_offset CDATA #REQUIRED
  height CDATA #REQUIRED
      show_key_signature (yes | no) "yes"
```

```
      show_clef (yes | no) "yes"
      show_time_signature (yes | no) "no"
      ossia (yes | no) "no">

<!ELEMENT layout_images EMPTY>
<!ATTLIST layout_images
  file_name CDATA #REQUIRED
  file_format %formats; #REQUIRED
  encoding_format %formats; #REQUIRED
  horizontal_offset CDATA #REQUIRED
  vertical_offset CDATA #REQUIRED
  description CDATA #IMPLIED
  copyright CDATA #IMPLIED
  notes CDATA #IMPLIED>

<!ELEMENT layout_shapes (svg)>
<!ATTLIST layout_shapes
  horizontal_offset CDATA #REQUIRED
  vertical_offset CDATA #REQUIRED>

<!ELEMENT text_font (font)>

<!ELEMENT music_font (font)>

<!--=========================================================================
====================-->

<!-- Structural Layer -->

<!ELEMENT structural (chord_grid*, analysis*, petri_nets*, mir*)>

<!-- Chord Grid -->

<!ELEMENT chord_grid (chord_name+)>
<!ATTLIST chord_grid
  id ID #IMPLIED
  author CDATA #IMPLIED
     description CDATA #IMPLIED>
```

```
<!ELEMENT chord_name (#PCDATA)>
<!ATTLIST chord_name
      root_id IDREF #REQUIRED>

<!-- Analysis -->

<!ELEMENT analysis (segmentation, relationships?, feature_object_relationships?)>
<!ATTLIST analysis
  id ID #IMPLIED
  author CDATA #IMPLIED
  description CDATA #IMPLIED>

<!ELEMENT segmentation (segment+)>
<!ATTLIST segmentation
  id ID #IMPLIED
  description CDATA #IMPLIED
  method CDATA #IMPLIED>

<!ELEMENT segment (segment_event+, feature_object*)>
<!ATTLIST segment
  id ID #REQUIRED>

<!ELEMENT segment_event EMPTY>
<!ATTLIST segment_event
  %spine_ref;>

<!ENTITY % added_feature_object_classes "">

<!ELEMENT feature_object (simple_description
   %added_feature_object_classes;)>
<!ATTLIST feature_object
  id ID #IMPLIED
  name CDATA #REQUIRED
+  notes CDATA #REQUIRED
+  uuid CDATA #REQUIRED
>

<!ELEMENT simple_description (#PCDATA)>
```

```
<!ELEMENT relationships (relationship+)>

<!ELEMENT relationship EMPTY>
<!ATTLIST relationship
  id ID #REQUIRED
  description CDATA #IMPLIED
  segment_a_ref IDREF #REQUIRED
  segment_b_ref IDREF #REQUIRED
  feature_object_a_ref IDREF #IMPLIED
  feature_object_b_ref IDREF #IMPLIED
  feature_object_relationship_ref IDREF #IMPLIED>

<!ELEMENT feature_object_relationships (feature_object_relationship+)>

<!ELEMENT feature_object_relationship (ver_rule)>
<!ATTLIST feature_object_relationship
  id ID #REQUIRED>

<!ELEMENT ver_rule (#PCDATA)>

<!-- Petri Nets -->

<!ELEMENT petri_nets (petri_net+)>

<!ELEMENT petri_net (place | transition)+>
<!ATTLIST petri_net
     id ID #IMPLIED
  author CDATA #IMPLIED
     description CDATA #IMPLIED
  file_name CDATA #REQUIRED>

<!ELEMENT place EMPTY>
<!ATTLIST place
     place_ref CDATA #REQUIRED
     segment_ref IDREF #REQUIRED>

<!ELEMENT transition EMPTY>
<!ATTLIST transition
     transition_ref CDATA #REQUIRED
```

```
      feature_object_relationship_ref IDREF #REQUIRED>

<!-- Music Information Retrieval -->

<!ELEMENT mir (mir_model+)>

<!ELEMENT mir_model (mir_object+, mir_morphism*)>
<!ATTLIST mir_model
  id ID #IMPLIED
  description CDATA #IMPLIED
  file_name CDATA #IMPLIED>

<!ELEMENT mir_object (mir_subobject+, mir_feature*)>
<!ATTLIST mir_object
  id ID #IMPLIED
  description CDATA #IMPLIED
  displacement_ref CDATA #IMPLIED>

<!ELEMENT mir_subobject (mir_feature*)>
<!ATTLIST mir_subobject
  id ID #IMPLIED
  description CDATA #IMPLIED
  displacement_ref CDATA #IMPLIED
  segment_ref IDREF #IMPLIED>

<!ELEMENT mir_feature EMPTY>
<!ATTLIST mir_feature
  id ID #IMPLIED
  description CDATA #IMPLIED
  displacement_ref CDATA #IMPLIED>

<!ELEMENT mir_morphism (mir_feature*)>
<!ATTLIST mir_morphism
  id ID #IMPLIED
  description CDATA #IMPLIED
  domain_ref IDREF #REQUIRED
  codomain_ref IDREF #REQUIRED
  displacement_ref CDATA #IMPLIED>
```

```
<!--=========================================================================================
====================-->

<!-- Notational Layer -->

<!ELEMENT notational (graphic_instance_group | notation_instance_group)+>

<!ELEMENT graphic_instance_group (graphic_instance+)>
<!ATTLIST graphic_instance_group
     description CDATA #REQUIRED>

<!ELEMENT graphic_instance (graphic_event+, rights?)>
<!ATTLIST graphic_instance
  description CDATA #IMPLIED
     position_in_group CDATA #REQUIRED
     file_name CDATA #REQUIRED
     file_format %formats; #REQUIRED
     encoding_format %formats; #REQUIRED
     measurement_unit (centimeters | millimeters | inches | decimal_inches | points | picas | pixels
| twips) #REQUIRED>

<!ELEMENT graphic_event EMPTY>
<!ATTLIST graphic_event
     %spine_ref;
     upper_left_x CDATA #REQUIRED
     upper_left_y CDATA #REQUIRED
     lower_right_x CDATA #REQUIRED
     lower_right_y CDATA #REQUIRED
     highlight_color CDATA #IMPLIED
  description CDATA #IMPLIED>

<!ELEMENT notation_instance_group (notation_instance+)>
<!ATTLIST notation_instance_group
     description CDATA #REQUIRED>

<!ELEMENT notation_instance (notation_event+, rights?)>
<!ATTLIST notation_instance
  description CDATA #IMPLIED
```

```
      position_in_group CDATA #REQUIRED
      file_name CDATA #REQUIRED
      format CDATA #REQUIRED
  measurement_unit CDATA #REQUIRED>

<!ELEMENT notation_event EMPTY>
<!ATTLIST notation_event
  %spine_ref;
  start_position CDATA #REQUIRED
  end_position CDATA #REQUIRED
  description CDATA #IMPLIED>


<!--=========================================================================
====================-->

<!-- Performance Layer -->

<!ELEMENT performance (midi_instance | csound_instance | mpeg4_instance)+>

<!ELEMENT midi_instance (midi_mapping+, rights?)>
<!ATTLIST midi_instance
  file_name CDATA #REQUIRED
      format (0 | 1 | 2) #REQUIRED>

<!ELEMENT midi_mapping (midi_event_sequence+)>
<!ATTLIST midi_mapping
      part_ref IDREF #REQUIRED
      voice_ref IDREF #IMPLIED
      track CDATA #REQUIRED
      channel CDATA #REQUIRED>

<!ELEMENT midi_event_sequence (midi_event | sys_ex)+ >
<!ATTLIST midi_event_sequence
      division_type  (metrical | timecode) #REQUIRED
      division_value NMTOKEN #REQUIRED
  measurement_unit (ticks | sec) #REQUIRED>

<!ELEMENT midi_event (%MIDIChannelMessage;)*>
```

```
<!ATTLIST midi_event
     timing CDATA #REQUIRED
  %spine_ref;>

<!ELEMENT sys_ex (SysEx)>
<!ATTLIST sys_ex
     %spine_ref;>

<!ELEMENT csound_instance (csound_score | csound_orchestra)+>

<!ELEMENT csound_score (csound_spine_event+, rights?)>
<!ATTLIST csound_score
     file_name CDATA #REQUIRED>

<!ELEMENT csound_spine_event EMPTY>
<!ATTLIST csound_spine_event
     line_number CDATA #REQUIRED
     %spine_ref;>

<!ELEMENT csound_orchestra (csound_instrument_mapping*, rights?)>
<!ATTLIST csound_orchestra
     file_name CDATA #REQUIRED>

<!ELEMENT csound_instrument_mapping (csound_part_ref | csound_spine_ref)+>
<!ATTLIST csound_instrument_mapping
     instrument_number CDATA #REQUIRED
     start_line CDATA #IMPLIED
     end_line CDATA #IMPLIED
     pnml_file CDATA #IMPLIED>

<!ELEMENT csound_part_ref EMPTY>
<!ATTLIST csound_part_ref
     part_ref IDREF #REQUIRED>

<!ELEMENT csound_spine_ref EMPTY>
<!ATTLIST csound_spine_ref
     %spine_ref;>

<!ELEMENT mpeg4_instance (mpeg4_score | mpeg4_orchestra)+>
```

```
<!ELEMENT mpeg4_score (csound_spine_event+, rights?)>
<!ATTLIST mpeg4_score
     file_name CDATA #REQUIRED>

<!ELEMENT mpeg4_spine_event EMPTY>
<!ATTLIST mpeg4_spine_event
     line_number CDATA #REQUIRED
     %spine_ref;>

<!ELEMENT mpeg4_orchestra (mpeg4_instrument_mapping*, rights?)>
<!ATTLIST mpeg4_orchestra
     file_name CDATA #REQUIRED>

<!ELEMENT mpeg4_instrument_mapping (mpeg4_part_ref | mpeg4_spine_ref)+>
<!ATTLIST mpeg4_instrument_mapping
     instrument_name CDATA #REQUIRED
     start_line CDATA #IMPLIED
     end_line CDATA #IMPLIED
     pnml_file CDATA #IMPLIED>

<!ELEMENT mpeg4_part_ref EMPTY>
<!ATTLIST mpeg4_part_ref
     part_ref IDREF #REQUIRED>

<!ELEMENT mpeg4_spine_ref EMPTY>
<!ATTLIST mpeg4_spine_ref
     %spine_ref;>

<!--=========================================================================
==================-->

<!-- Audio Layer -->

<!ELEMENT audio (track+)>

<!ELEMENT track (track_general?, track_indexing?, rights?)>
<!ATTLIST track
     file_name CDATA #REQUIRED
```

```
      file_format %formats; #REQUIRED
      encoding_format %formats; #REQUIRED
    md5 CDATA #IMPLIED>

<!-- General Sub-Layer -->

<!ELEMENT track_general (recordings?, genres?, albums?, performers?, notes?)>
<!ATTLIST track_general
      geographical_region CDATA #IMPLIED
      lyrics_language CDATA #IMPLIED>

<!ELEMENT recordings (recording+)>

<!ELEMENT recording EMPTY>
<!ATTLIST recording
      date CDATA #REQUIRED
      recorded_part CDATA #IMPLIED
      studio_name CDATA #IMPLIED
      studio_address CDATA #IMPLIED>

<!ELEMENT albums (album+)>

<!ELEMENT album EMPTY>
<!ATTLIST album
      title CDATA #REQUIRED
    track_number CDATA #IMPLIED
      carrier_type CDATA #IMPLIED
      catalogue_number CDATA #IMPLIED
      number_of_tracks CDATA #IMPLIED
      publication_date CDATA #IMPLIED
      label CDATA #IMPLIED>

<!ELEMENT performers (performer+)>

<!ELEMENT performer EMPTY>
<!ATTLIST performer
      name CDATA #REQUIRED
      type CDATA #REQUIRED>
```

```
<!-- Indexing Sub-Layer -->

<!ELEMENT track_indexing (track_region*, track_event+)>
<!ATTLIST track_indexing
     timing_type (samples | time | seconds | time_frames | frames | measures | smpte_24 | smpte_25
| smpte_2997 | smpte_30) #REQUIRED>

<!ELEMENT track_region EMPTY>
<!ATTLIST track_region
     name CDATA #REQUIRED
     description CDATA #IMPLIED
  %spine_start_end_ref;>

<!ELEMENT track_event EMPTY>
<!ATTLIST track_event
     start_time CDATA #REQUIRED
     end_time CDATA #IMPLIED
  %spine_ref;
     description CDATA #IMPLIED>
```

附錄二、音樂特徵 XML 描述檔案 Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="feature">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="chord">
          <xs:complexType>
            <xs:sequence>
              <xs:element maxOccurs="unbounded" name="markov_instance">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="from" type="xs:string" />
                    <xs:element name="to" type="xs:string" />
                    <xs:element name="pro" type="xs:unsignedByte" />
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="melody_duration">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="markovs">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element maxOccurs="unbounded" name="markov_instance">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="from" type="xs:decimal" />
                          <xs:element name="to" type="xs:decimal" />
                          <xs:element name="pro" type="xs:decimal" />
                        </xs:sequence>
                      </xs:complexType>
                    </xs:element>
```

```
                    </xs:sequence>
                  </xs:complexType>
                </xs:element>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
</xs:schema>
```

附錄三、成果分析結果 XML 描述檔案 Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="AC_analysis">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="attribute">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="count" type="xs:unsignedByte" />
              <xs:element name="min" type="xs:unsignedByte" />
              <xs:element name="max" type="xs:unsignedByte" />
              <xs:element name="average" type="xs:decimal" />
              <xs:element name="sd" type="xs:decimal" />
              <xs:element name="highpoints">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="highpoint" type="xs:unsignedByte" />
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="pitchs">
          <xs:complexType>
            <xs:sequence>
              <xs:element maxOccurs="unbounded" name="pitch">
                <xs:complexType>
                  <xs:simpleContent>
                    <xs:extension base="xs:unsignedByte">
                      <xs:attribute name="position" type="xs:unsignedByte" use="required" />
                    </xs:extension>
                  </xs:simpleContent>
                </xs:complexType>
              </xs:element>
```

```
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="durations">
      <xs:complexType>
        <xs:sequence>
          <xs:element maxOccurs="unbounded" name="duration">
            <xs:complexType>
              <xs:simpleContent>
                <xs:extension base="xs:unsignedByte">
                  <xs:attribute name="position" type="xs:unsignedByte" use="required" />
                </xs:extension>
              </xs:simpleContent>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
</xs:schema>
```