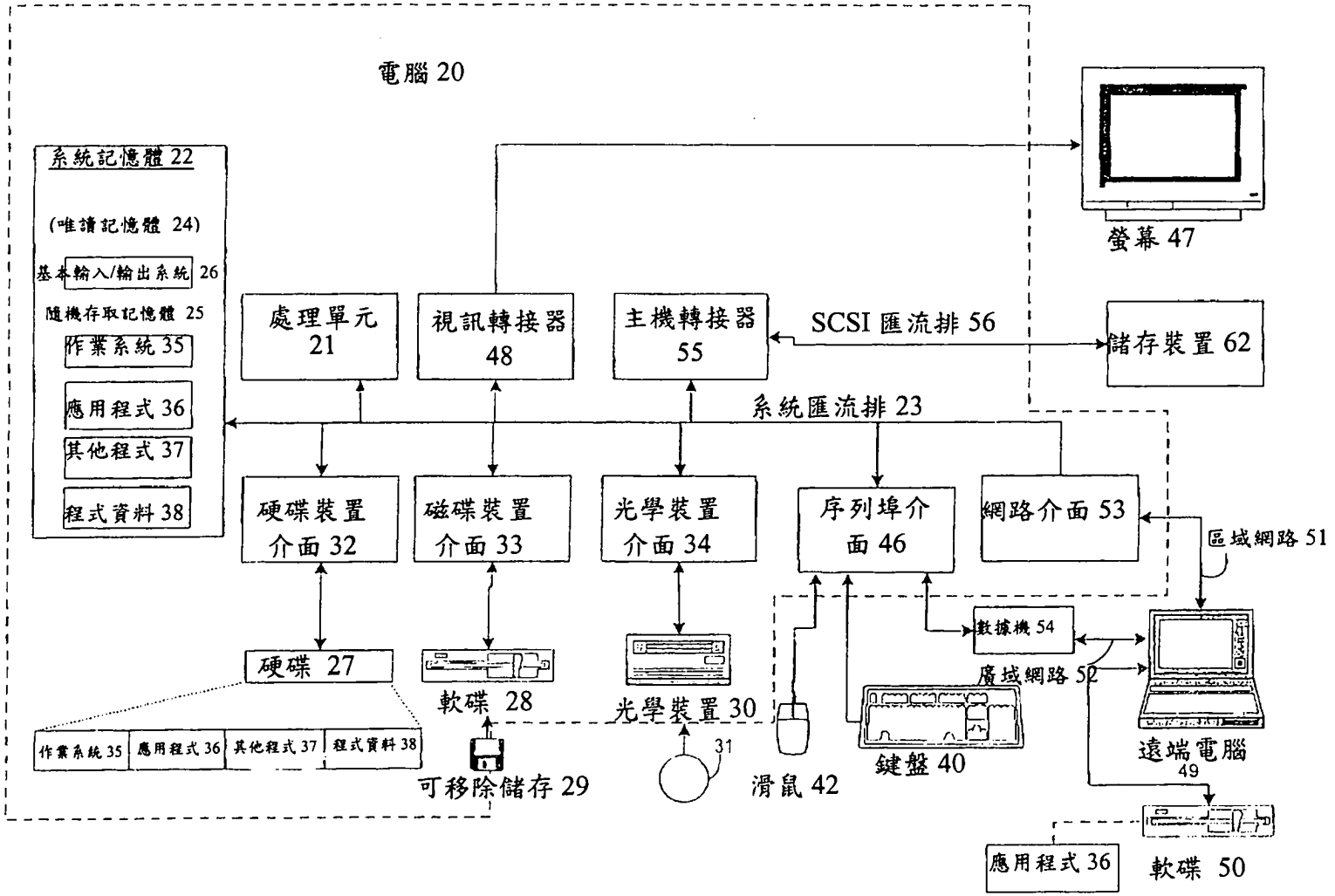
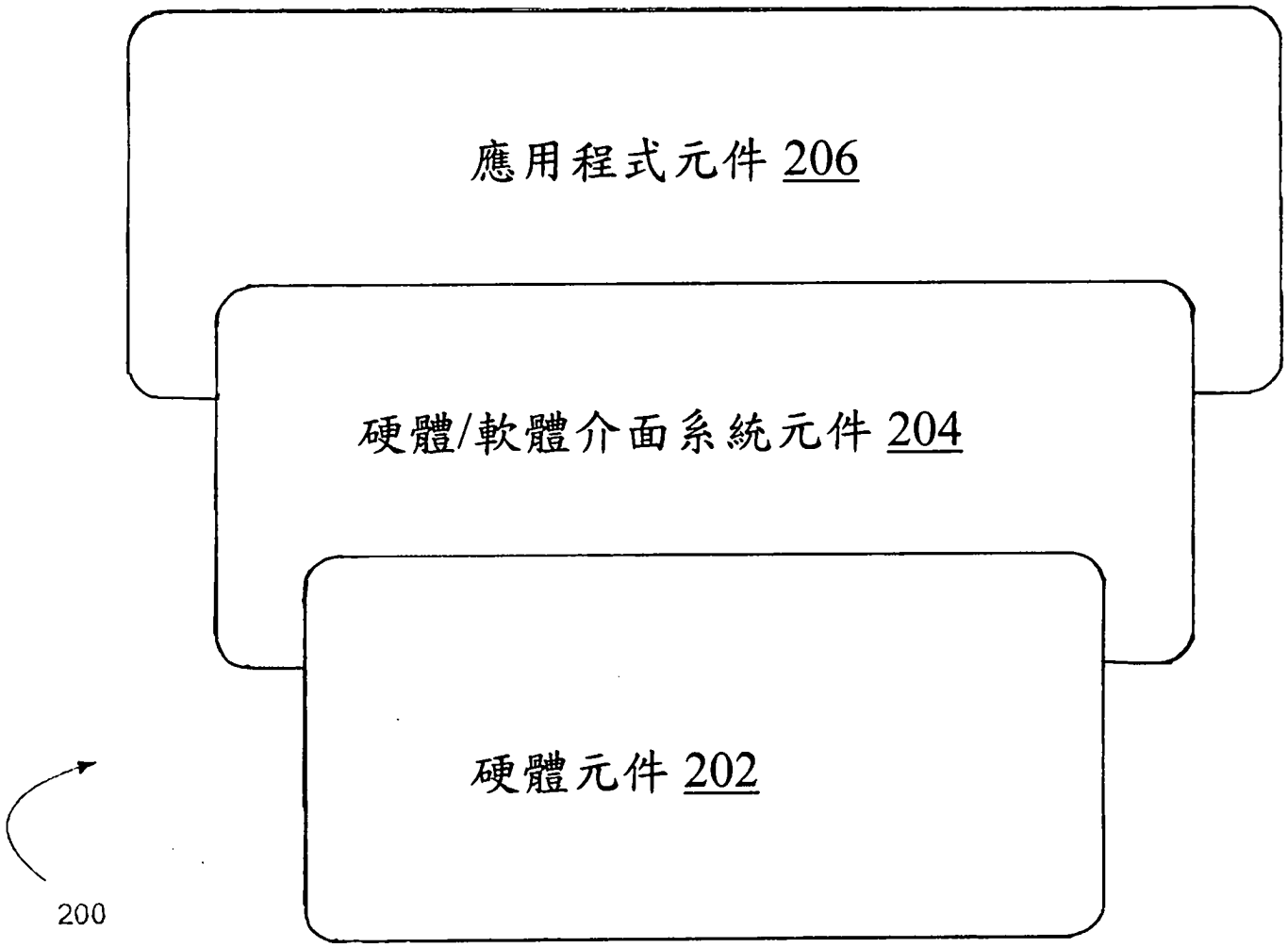
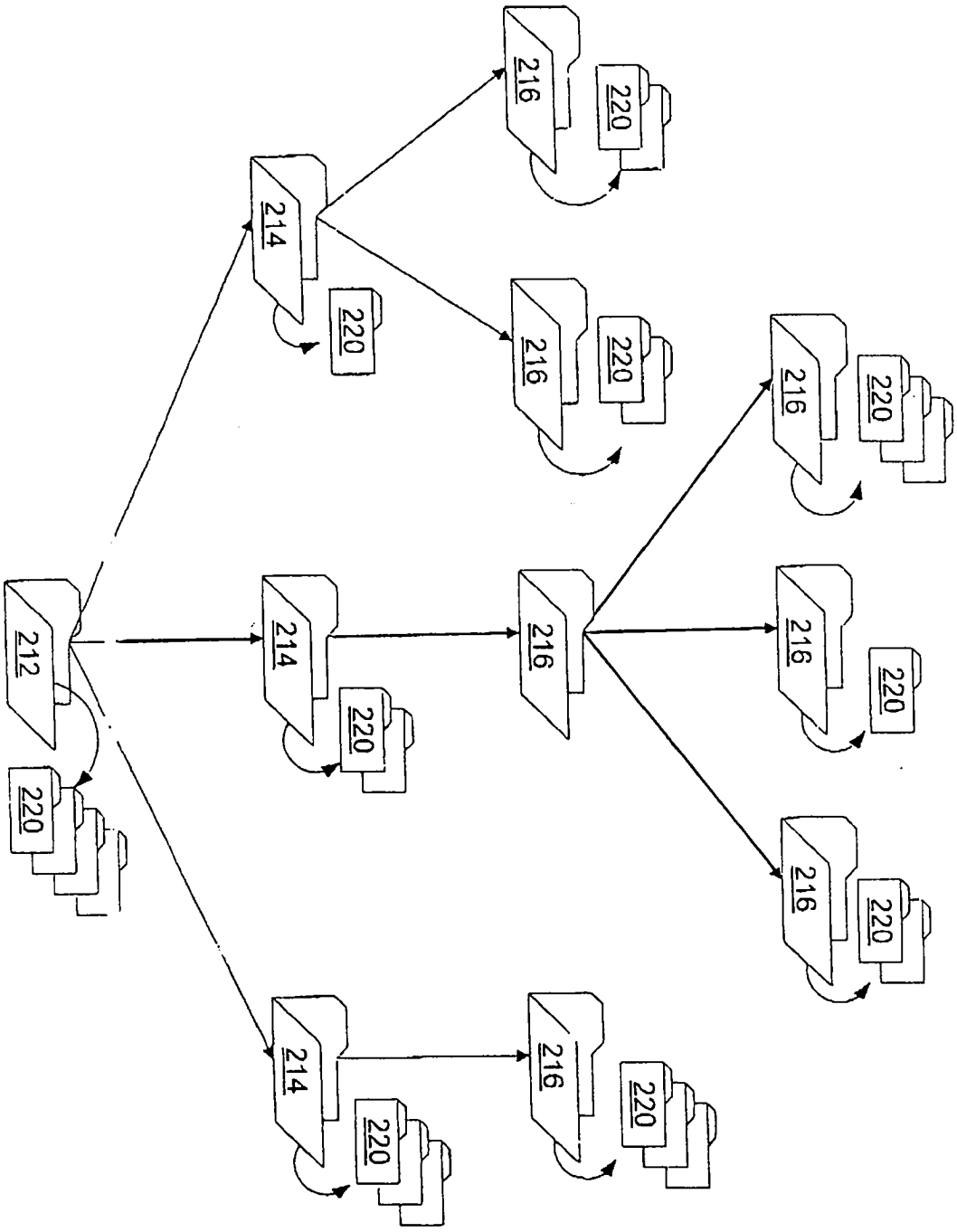

電腦 20
系統記憶體 22
(唯讀記憶體 24)
基本輸入/輸出系統 26
隨機存取記憶體 25
作業系統 35
應用程式 36
其他程式 37
程式資料 38
處理單元 21
視訊轉接器 48
主機轉接器 55
SCSI 匯流排 56
儲存裝置 62
螢幕 47
系統匯流排 23
硬碟裝置介面 32
磁碟裝置介面 33
光學裝置介面 34
序列埠介面 46
網路介面 53
區域網路 51
數據機 54
廣域網路 52
硬碟 27
軟碟 28
光學裝置 30
31
鍵盤 40
滑鼠 42
遠端電腦 49
可移除儲存 29
作業系統 35
應用程式 36
其他程式 37
程式資料 38
應用程式 36
軟碟 50

第 1 圖

I337310

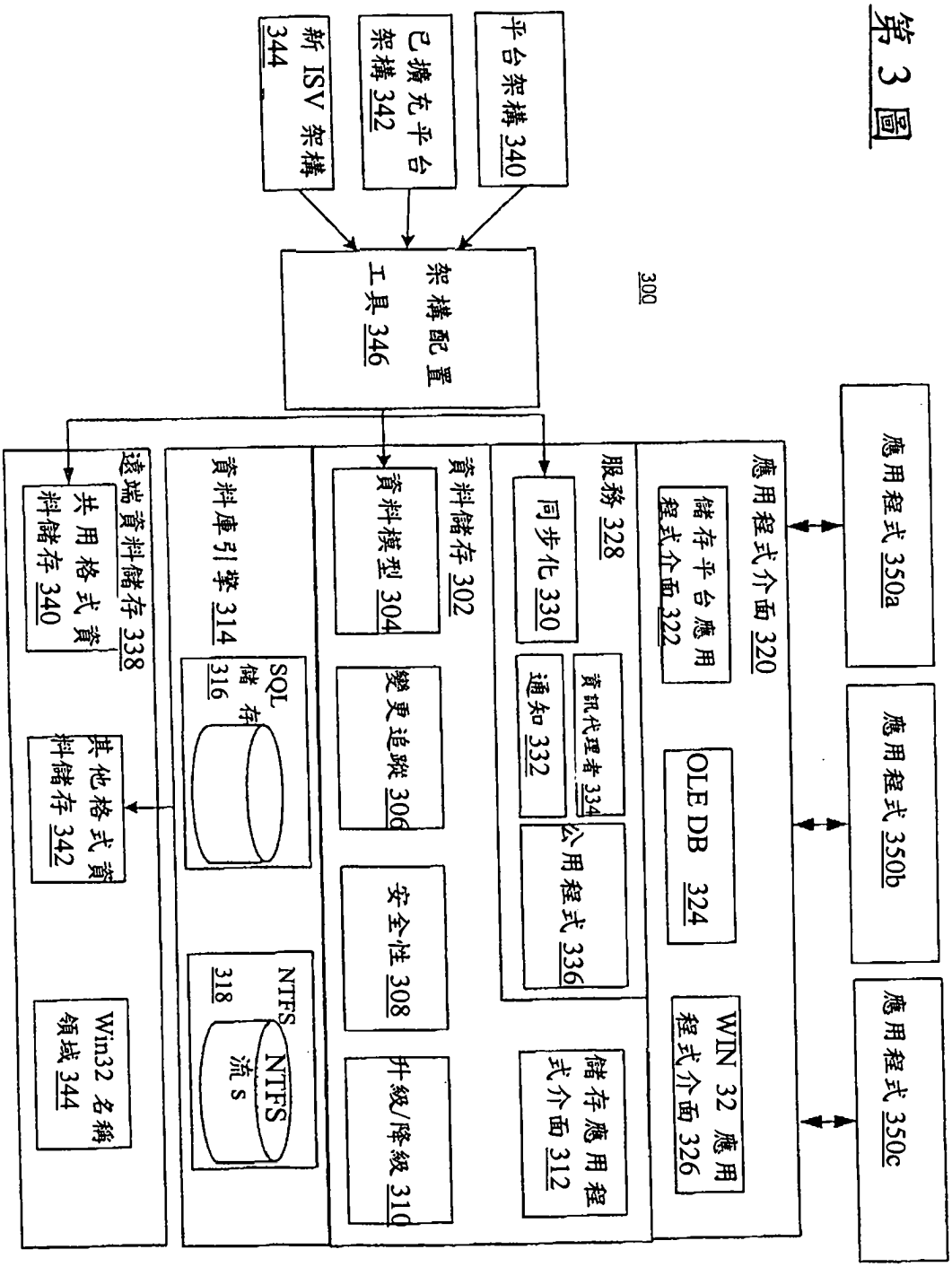


第 2 圖

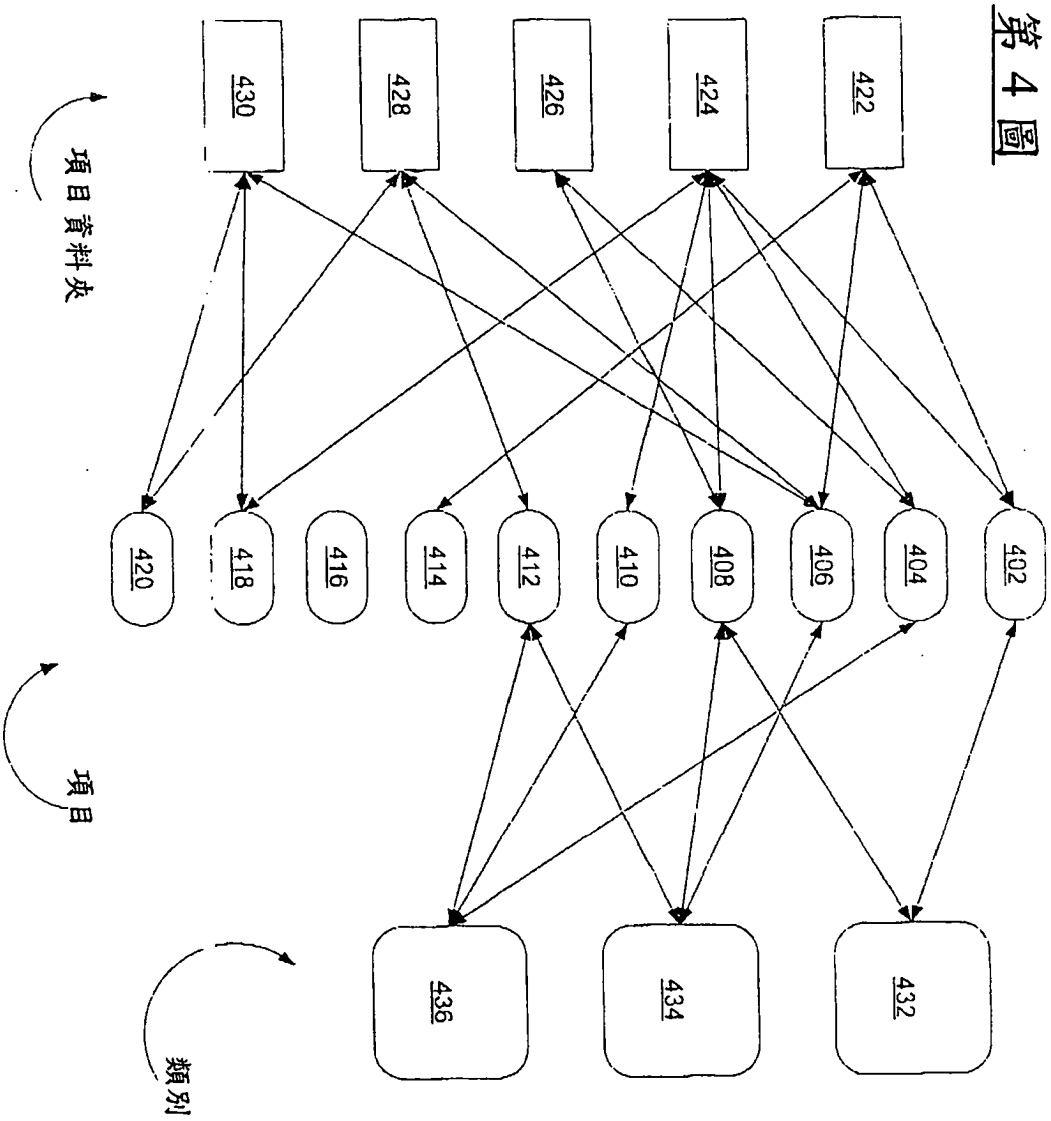


第 2A 圖

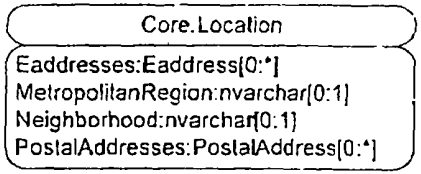
第 3 圖


應用程式 350a
應用程式 350b
應用程式 350c
應用程式介面 320
儲存平台應用程式介面 322
OLE DB 324
WIN 32 應用程式介面 326
300
平台架構 340
已擴充平台架構 342
新 ISV 架構 344
架構配置工具 346
服務 328
同步化 330
資訊代理者 334
通知 332
公用程式 336
儲存應用程式介面 312
資料儲存 302
資料模型 304
變更追蹤 306
安全性 308
升級/降級 310
遠端資料儲存 338
共用格式資料儲存 340
其他格式資料儲存 342
資料庫引擎 314
SQL 儲存 316
NTFS 流 s
318
Win32 名稱領域 344

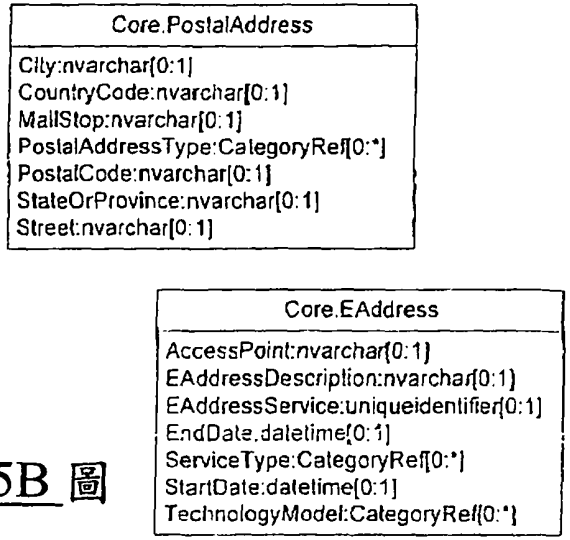
第 4 圖


422
424
426
428
430
402
404
406
408
410
412
414
416
418
420
432
434
436
項目資料夾
項目
類別

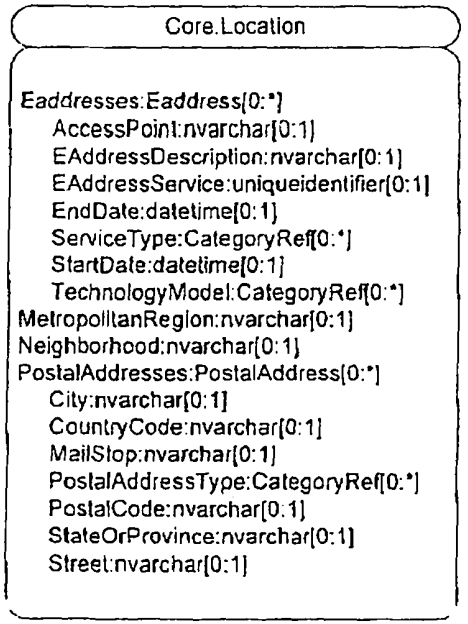
第 5A 圖



第 5B 圖

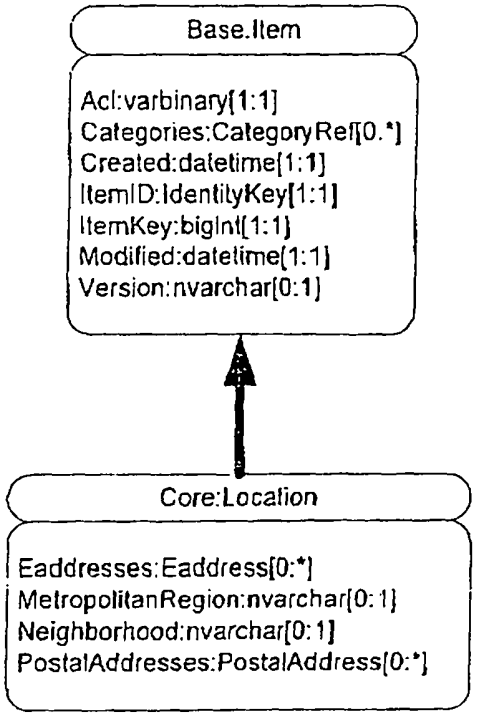


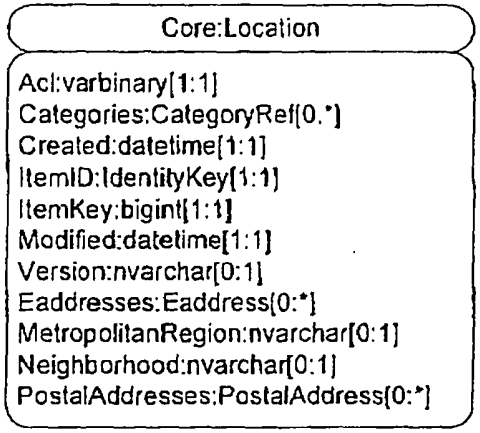
第 5C 圖



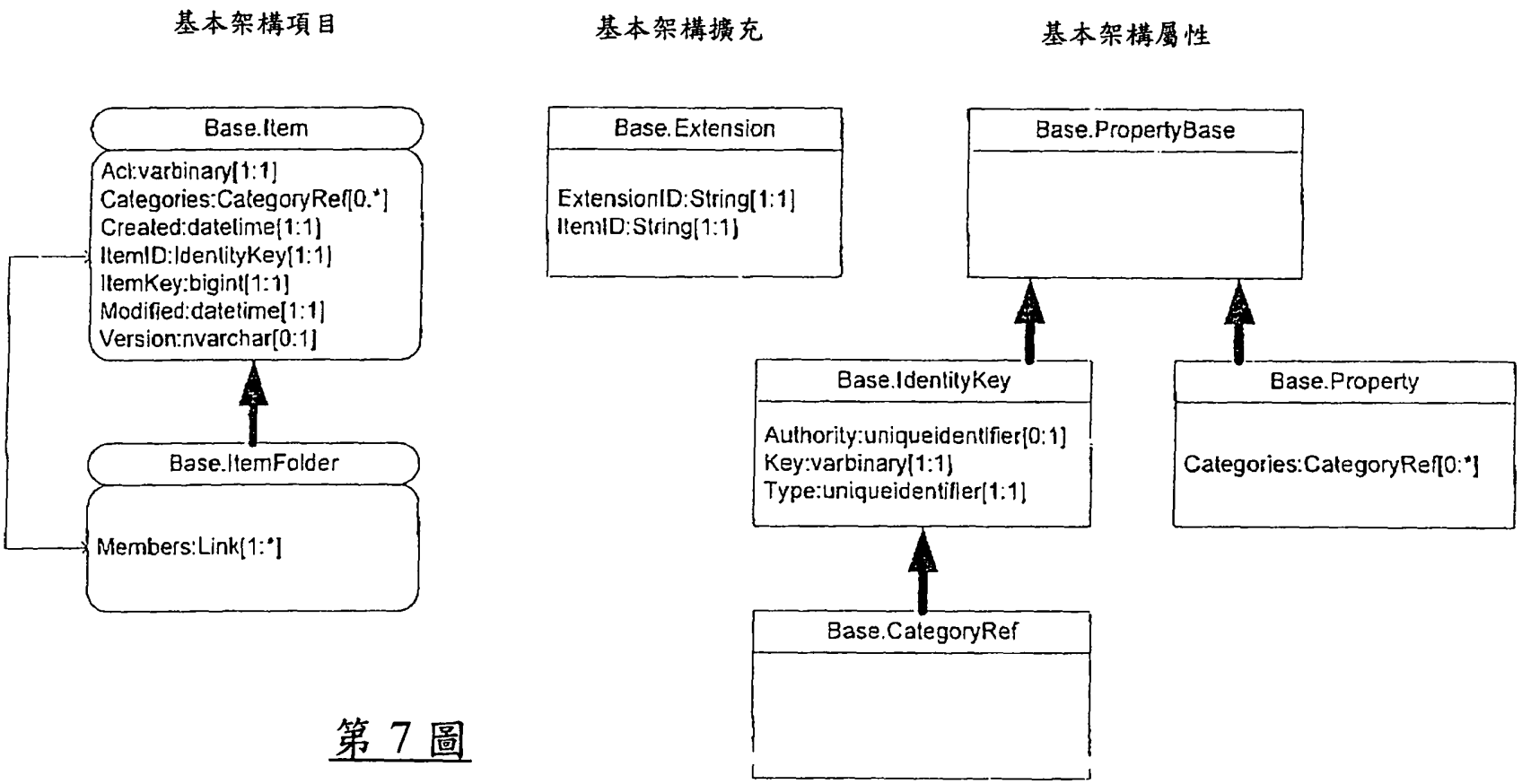
I337310

第 6A 圖


Base.Item
Acl:varbinary[1:1]
Categories:CategoryRef[0.*]
Created:datetime[1:1]
ItemID:IdentityKey[1:1]
ItemKey:bigint[1:1]
Modified:datetime[1:1]
Version:nvarchar[0:1]
Core:Location
Eaddresses:Eaddress[0:*]
MetropolitanRegion:nvarchar[0:1]
Neighborhood:nvarchar[0:1]
PostalAddresses:PostalAddress[0:*]


Core:Location
Acl:varbinary[1:1]
Categories:CategoryRef[0.*]
Created:datetime[1:1]
ItemID:IdentityKey[1:1]
ItemKey:bigint[1:1]
Modified:datetime[1:1]
Version:nvarchar[0:1]
Eaddresses:Eaddress[0:*]
MetropolitanRegion:nvarchar[0:1]
Neighborhood:nvarchar[0:1]
PostalAddresses:PostalAddress[0:*]

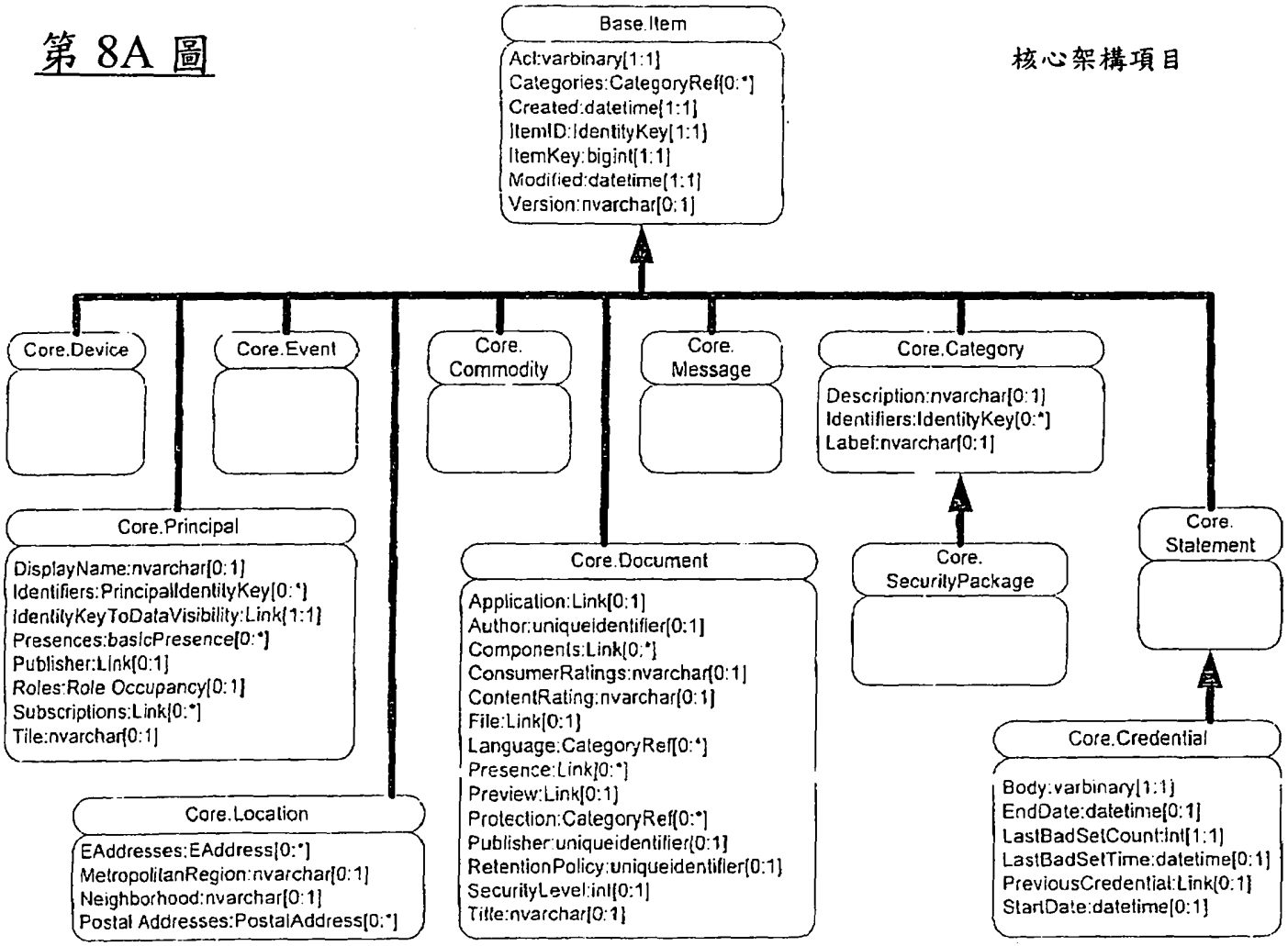
第 6B 圖


基本架構項目
基本架構擴充
基本架構屬性
Base.Item
Acl:varbinary[1:1]
Categories:CategoryRef[0:*]
Created:datetime[1:1]
ItemID:IdentityKey[1:1]
ItemKey:bigint[1:1]
Modified:datetime[1:1]
Version:nvarchar[0:1]
Base.ItemFolder
Members:Link[1:*]
Base.Extension
ExtensionID:String[1:1]
ItemID:String[1:1]
Base.PropertyBase
Base.IdentityKey
Authority:uniqueidentifier[0:1]
Key:varbinary[1:1]
Type:uniqueidentifier[1:1]
Base.Property
Categories:CategoryRef[0:*]
Base.CategoryRef

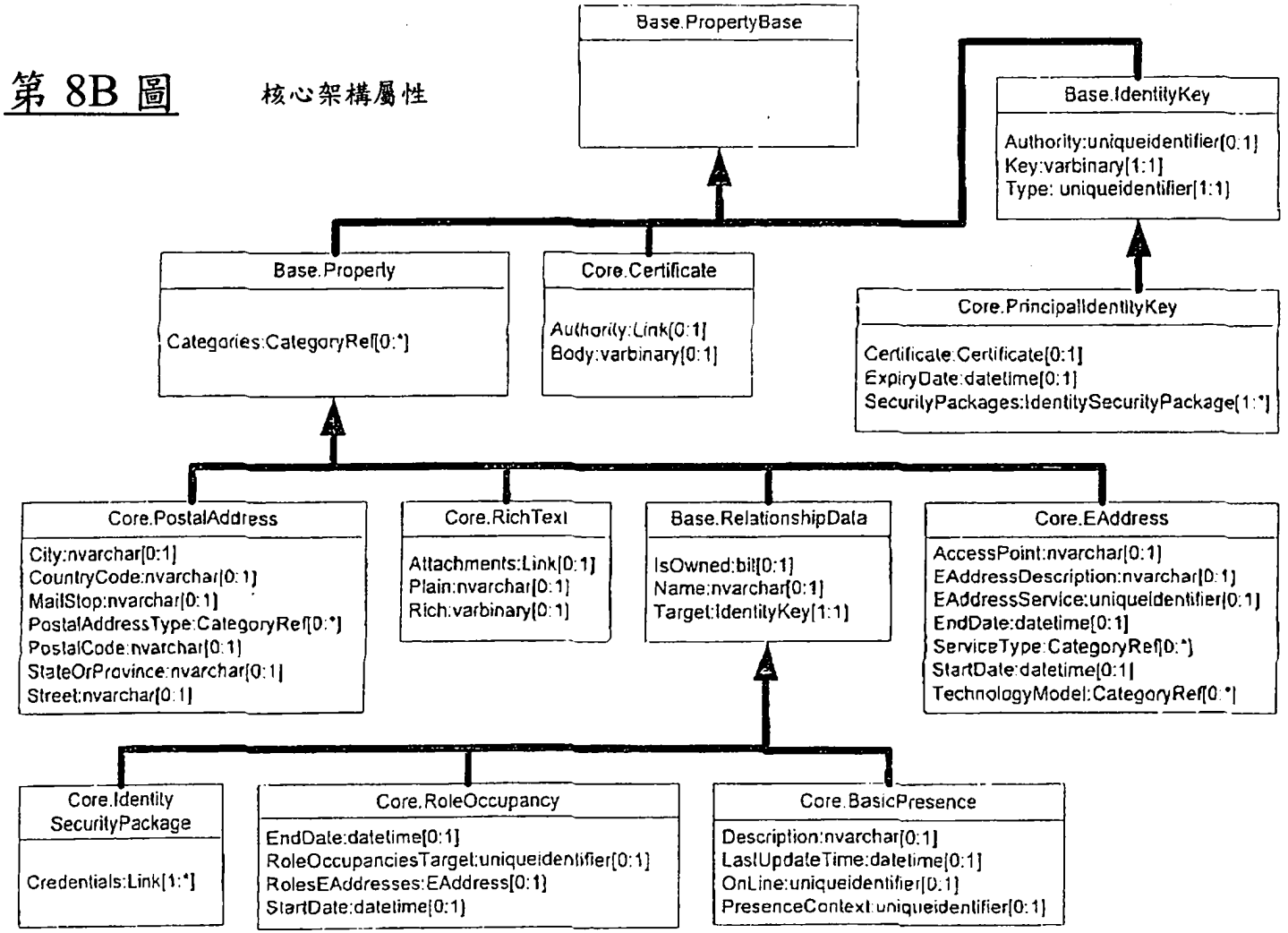
第 7 圖

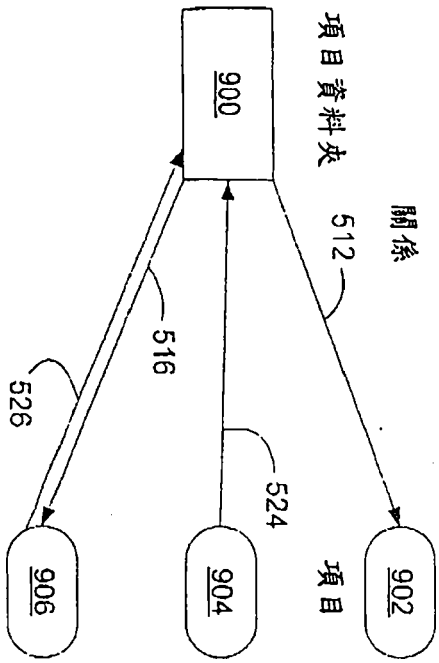
第 8A 圖

核心架構項目

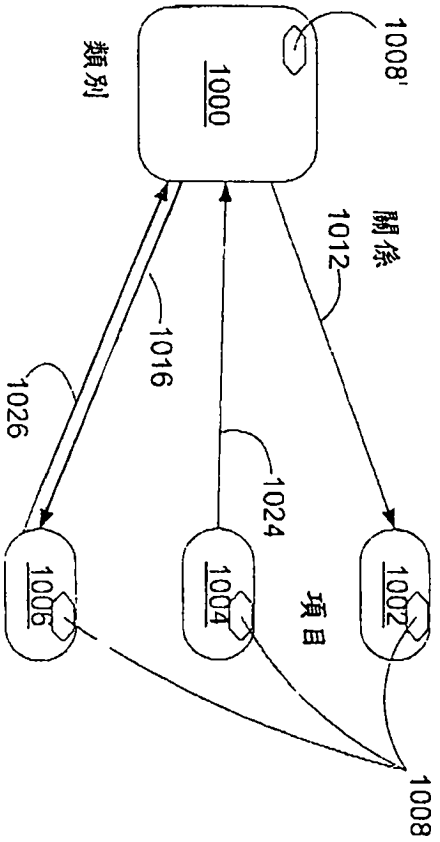

Base.Item
Acl:varbinary[1:1]
Categories:CategoryRef[0:*]
Created:datetime[1:1]
ItemID:IdentityKey[1:1]
ItemKey:bigint[1:1]
Modified:datetime[1:1]
Version:nvarchar[0:1]
Core.Device
Core.Event
Core. Commodity
Core. Message
Core.Category
Description:nvarchar[0:1]
Identifiers:IdentityKey[0:*]
Label:nvarchar[0:1]
Core.Principal
DisplayName:nvarchar[0:1]
Identifiers:PrincipalIdentityKey[0:*]
IdentityKeyToDataVisibility:Link[1:1]
Presences:basicPresence[0:*]
Publisher:Link[0:1]
Roles:Role Occupancy[0:1]
Subscriptions:Link[0:*]
Tile:nvarchar[0:1]
Core.Document
Application:Link[0:1]
Author:uniqueidentifier[0:1]
Components:Link[0:*]
ConsumerRatings:nvarchar[0:1]
ContentRating:nvarchar[0:1]
File:Link[0:1]
Language:CategoryRef[0:*]
Presence:Link[0:*]
Preview:Link[0:1]
Protection:CategoryRef[0:*]
Publisher:uniqueidentifier[0:1]
RetentionPolicy:uniqueidentifier[0:1]
SecurityLevel:int[0:1]
Title:nvarchar[0:1]
Core. SecurityPackage
Core. Statement
Core.Credential
Body:varbinary[1:1]
EndDate:datetime[0:1]
LastBadSetCount:int[1:1]
LastBadSetTime:datetime[0:1]
PreviousCredential:Link[0:1]
StartDate:datetime[0:1]
Core.Location
EAddresses:EAddress[0:*]
MetropolitanRegion:nvarchar[0:1]
Neighborhood:nvarchar[0:1]
Postal Addresses:PostalAddress[0:*]

第 8B 圖　　核心架構屬性

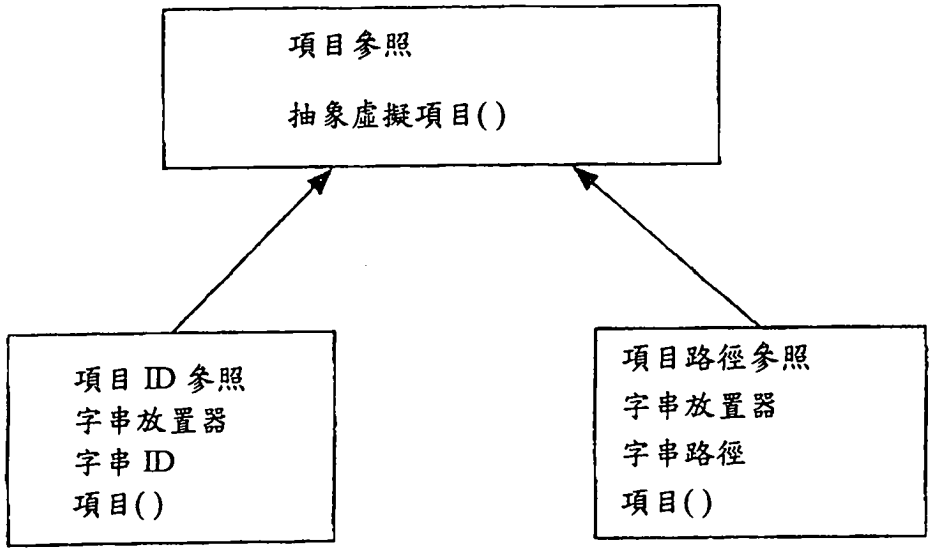

Base.PropertyBase
Base.IdentityKey
Authority:uniqueidentifier[0:1]
Key:varbinary[1:1]
Type: uniqueidentifier[1:1]
Base.Properly
Categories:CategoryRef[0:*]
Core.Certificate
Authority:Link[0:1]
Body:varbinary[0:1]
Core.PrincipalIdentityKey
Certificate:Certificate[0:1]
ExpiryDate:datetime[0:1]
SecurityPackages:IdentitySecurityPackage[1:*]
Core.PostalAddress
City:nvarchar[0:1]
CountryCode:nvarchar[0:1]
MailStop:nvarchar[0:1]
PostalAddressType:CategoryRef[0:*]
PostalCode:nvarchar[0:1]
StateOrProvince:nvarchar[0:1]
Street:nvarchar[0:1]
Core.RichText
Attachments:Link[0:1]
Plain:nvarchar[0:1]
Rich:varbinary[0:1]
Base.RelationshipData
IsOwned:bit[0:1]
Name:nvarchar[0:1]
Target:IdentityKey[1:1]
Core.EAddress
AccessPoint:nvarchar[0:1]
EAddressDescription:nvarchar[0:1]
EAddressService:uniqueIdentifier[0:1]
EndDate:datetime[0:1]
ServiceType:CategoryRef[0:*]
StartDate:datetime[0:1]
TechnologyModel:CategoryRef[0:*]
Core.Identity SecurityPackage
Credentials:Link[1:*]
Core.RoleOccupancy
EndDate:datetime[0:1]
RoleOccupanciesTarget:uniqueidentifier[0:1]
RolesEAddresses:EAddress[0:1]
StartDate:datetime[0:1]
Core.BasicPresence
Description:nvarchar[0:1]
LastUpdateTime:datetime[0:1]
OnLine:uniqueidentifier[0:1]
PresenceContext:uniqueidentifier[0:1]


900
項目資料夾
關係
512
516
526
524
902
904
906
項目

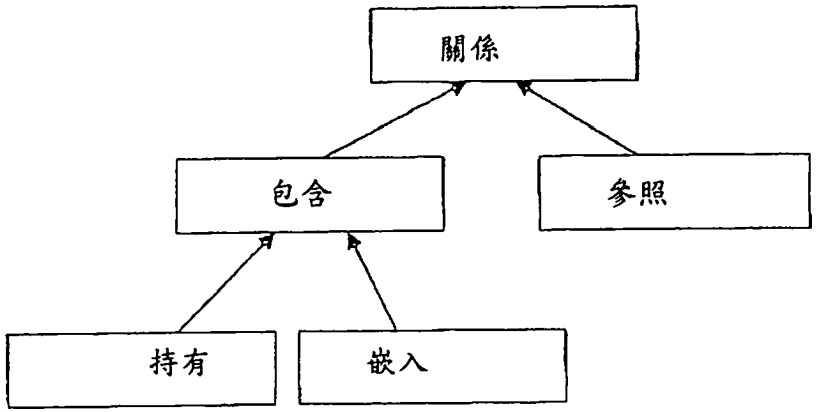
第 10 圖


1000
類別
1008'
關係
1012
1016
1026
1024
1002
1004
1006
項目
1008

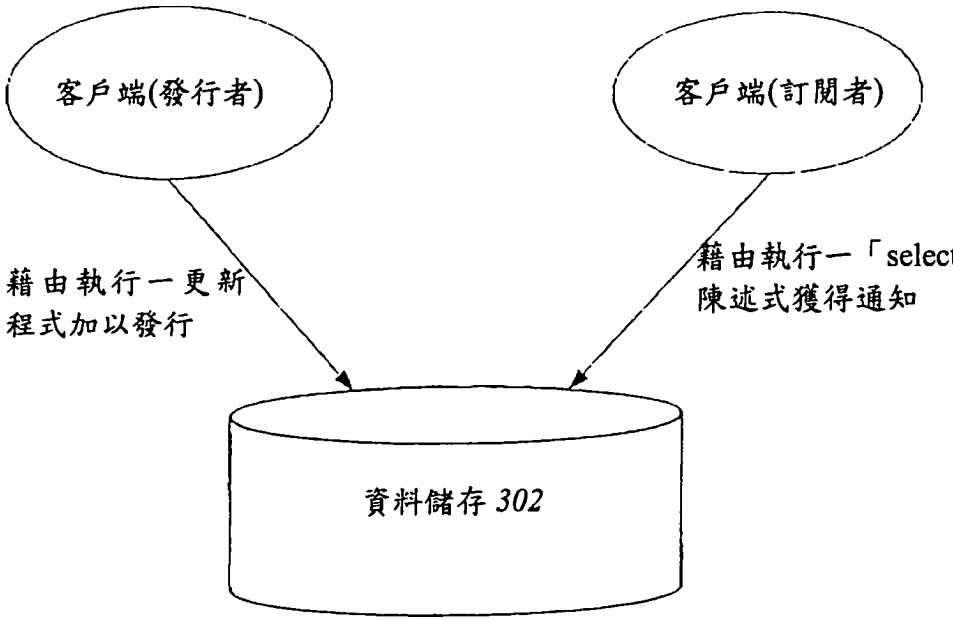
第 9 圖


項目參照
抽象虛擬項目()
項目 ID 參照
字串放置器
字串 ID
項目()
項目路徑參照
字串放置器
字串路徑
項目()

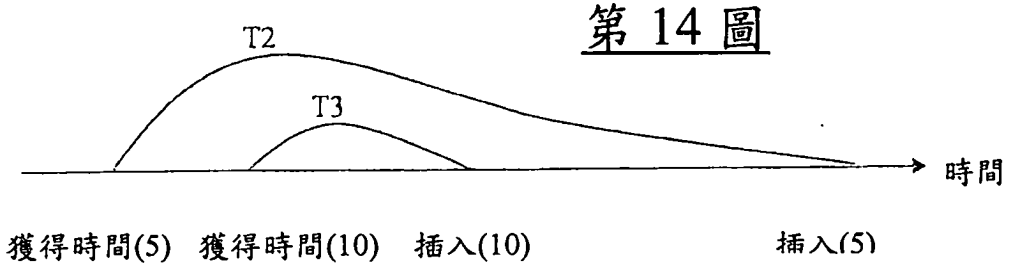
第 11 圖


關係
包含
參照
持有
嵌入

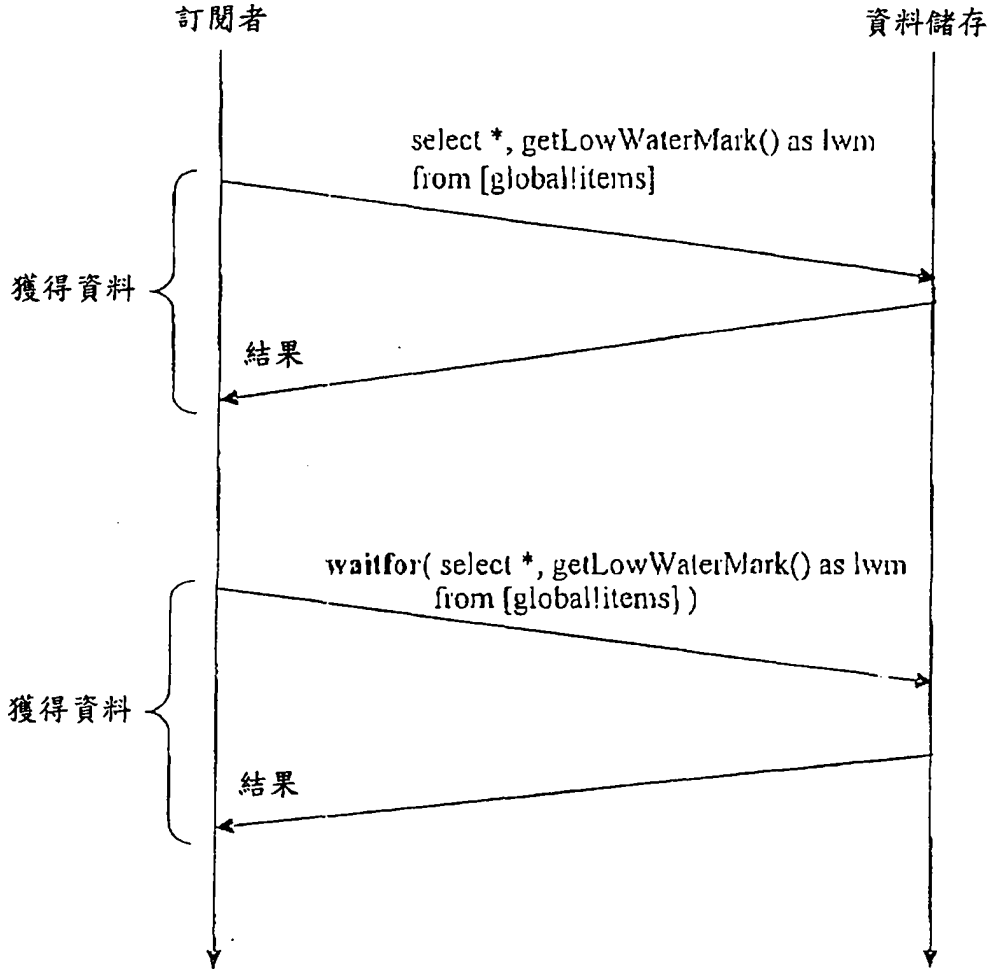
第 12 圖


客戶端(發行者)
客戶端(訂閱者)
藉由執行一更新程式加以發行
藉由執行一「select」陳述式獲得通知
資料儲存 302

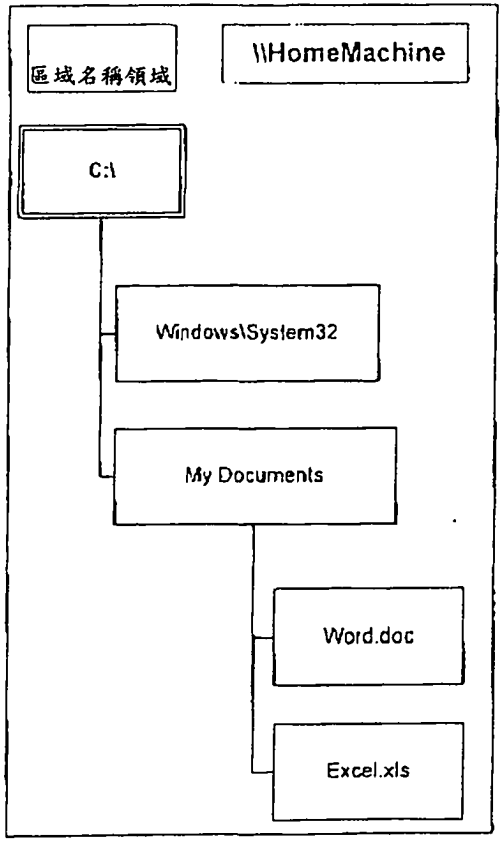
第 13 圖


T2
T3
時間
獲得時間(5)
獲得時間(10)
插入(10)
插入(5)

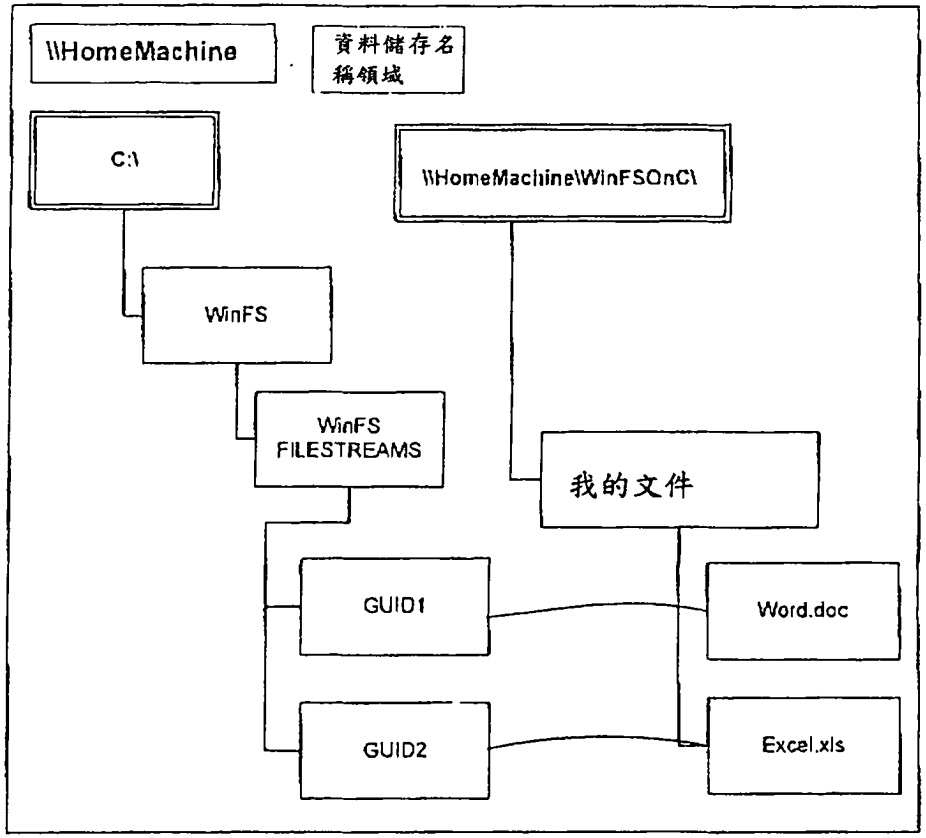
第 14 圖


訂閱者
資料儲存
select *, getLowWaterMark() as lwm
from [global!items]
獲得資料
結果
waitfor(select *, getLowWaterMark() as lwm
from [global!items])
獲得資料
結果

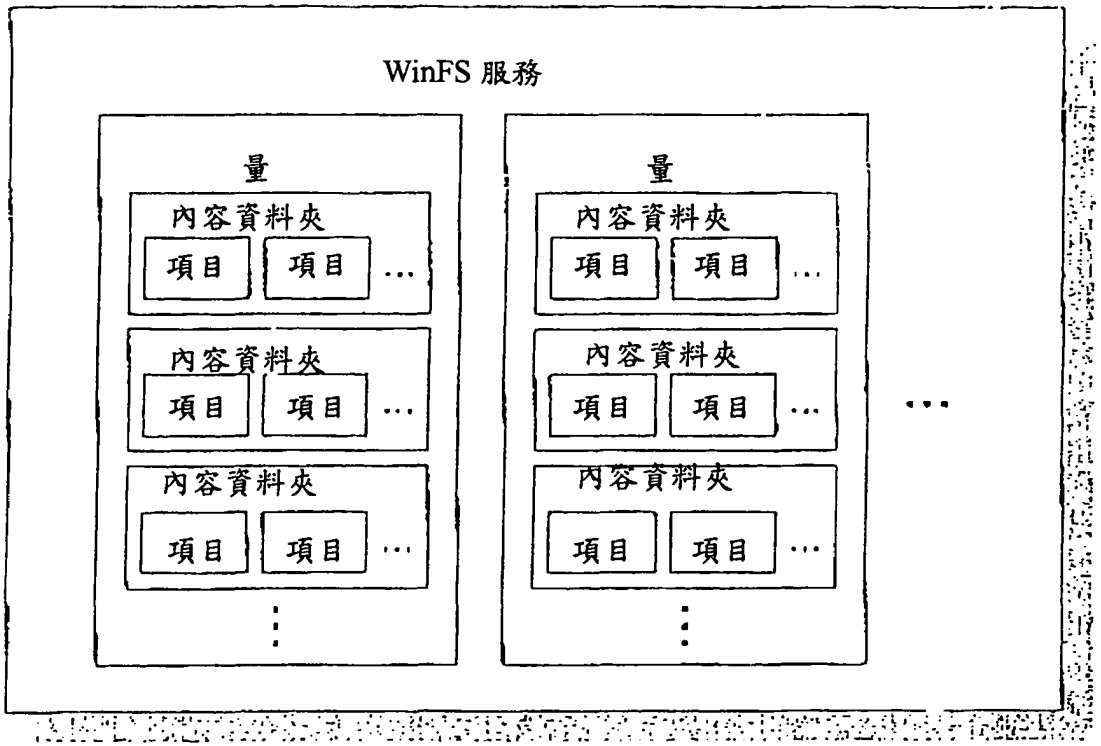
第 15 圖


區域名稱領域
\\HomeMachine
C:\
Windows\System32
My Documents
Word.doc
Excel.xls

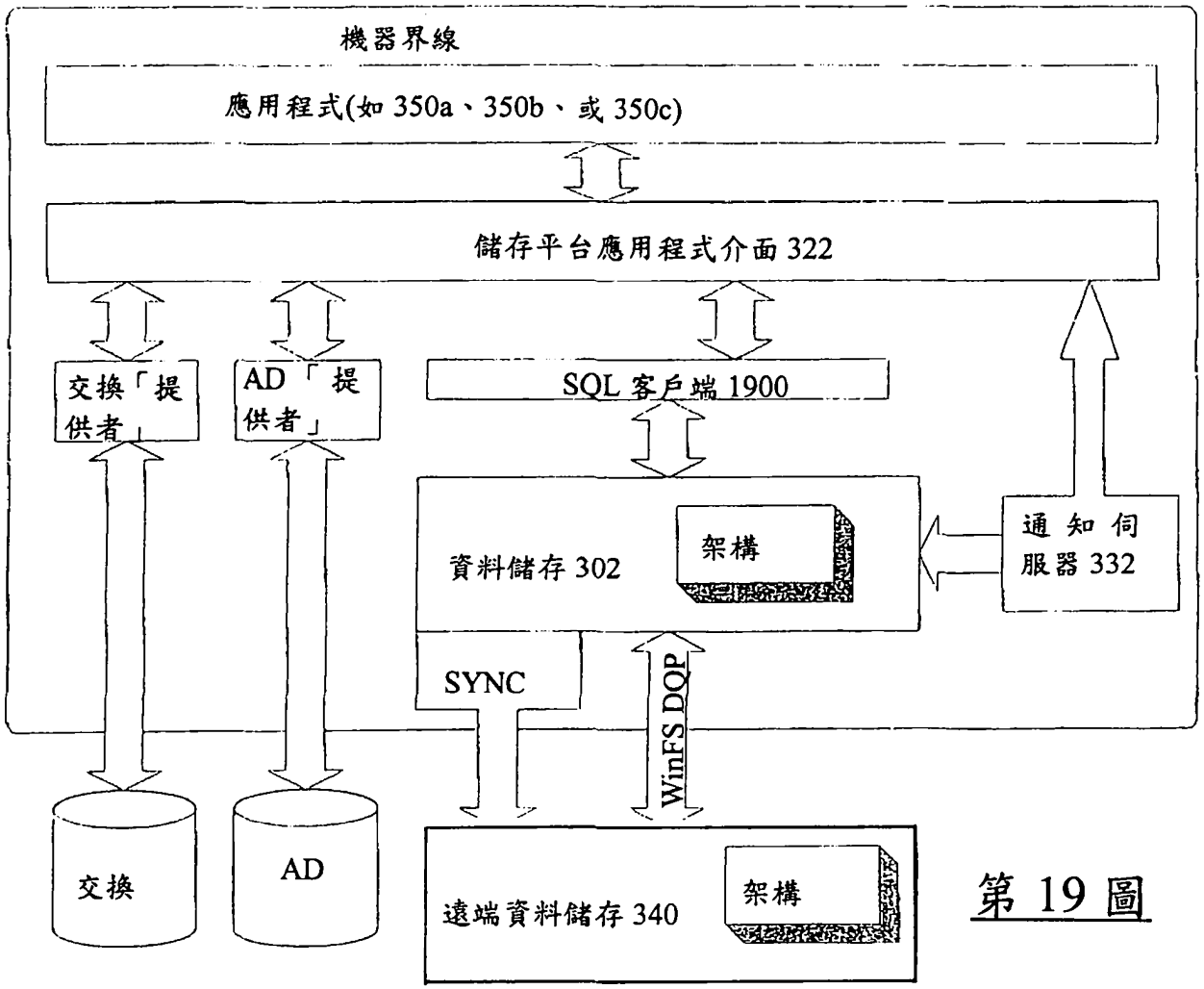
第 16 圖


\\HomeMachine
資料儲存名稱領域
C:\
\\HomeMachine\WinFSOnC\
WinFS
WinFS FILESTREAMS
我的文件
GUID1
GUID2
Word.doc
Excel.xls

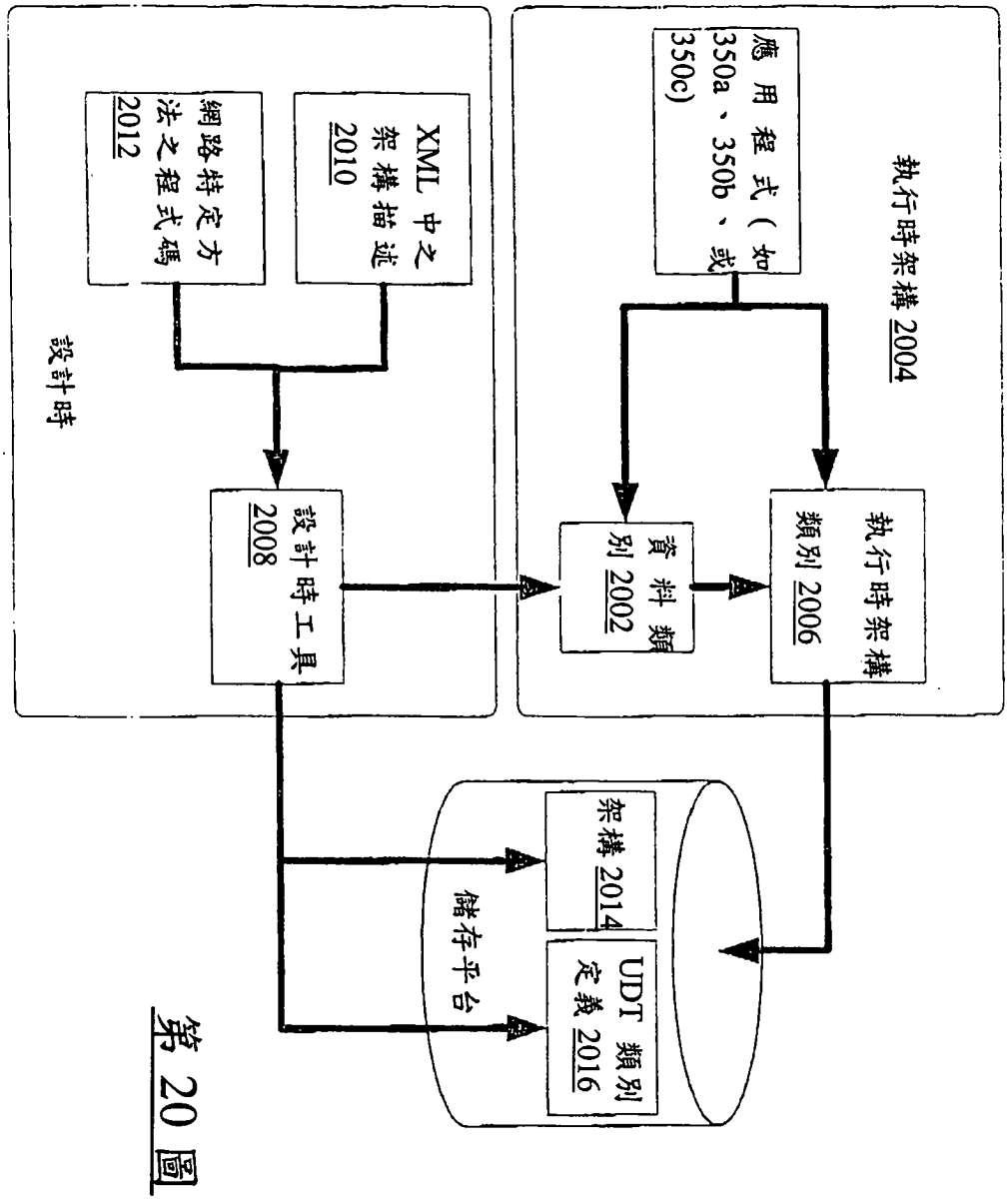
第 17 圖


WinFS 服務
量
內容資料夾
項目
項目
內容資料夾
項目
項目
內容資料夾
項目
項目
量
內容資料夾
項目
項目
內容資料夾
項目
項目
內容資料夾
項目
項目

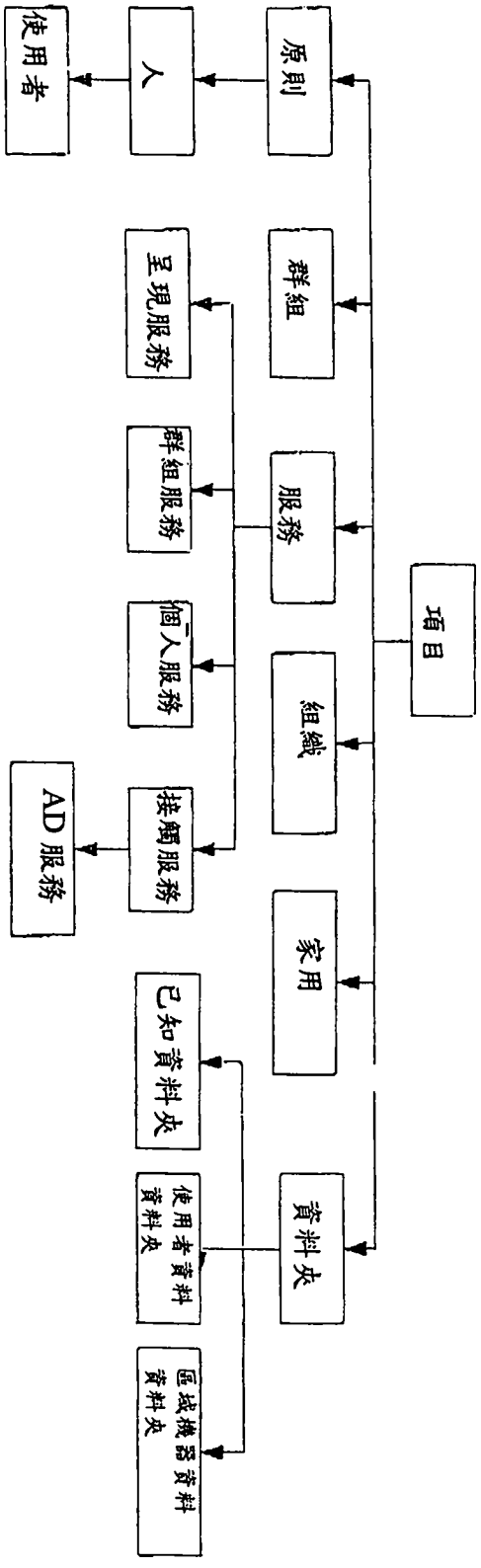
第 18 圖


機器界線
應用程式(如 350a、350b、或 350c)
儲存平台應用程式介面 322
交換「提供者」
AD「提供者」
SQL 客戶端 1900
資料儲存 302
架構
通知伺服器 332
SYNC
WinFS DQP
交換
AD
遠端資料儲存 340
架構

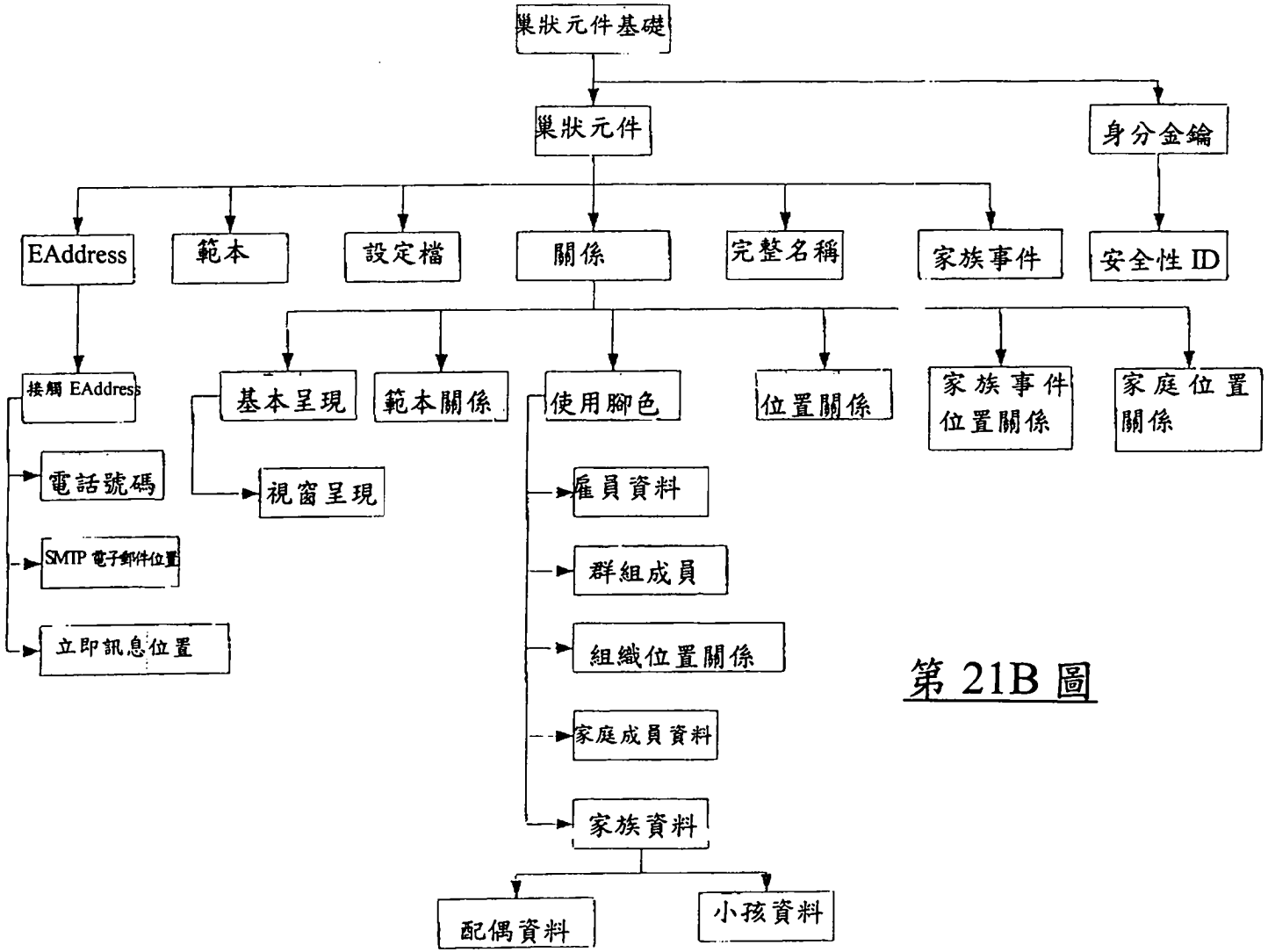
第 19 圖


執行時架構 2004
應用程式（如 350a、350b、或 350c）
資料類別 2002
執行時架構類別 2006
設計時
XML 中之架構描述 2010
網路特定方法之程式碼 2012
設計時工具 2008
儲存平台
架構 2014
UDT 類別定義 2016

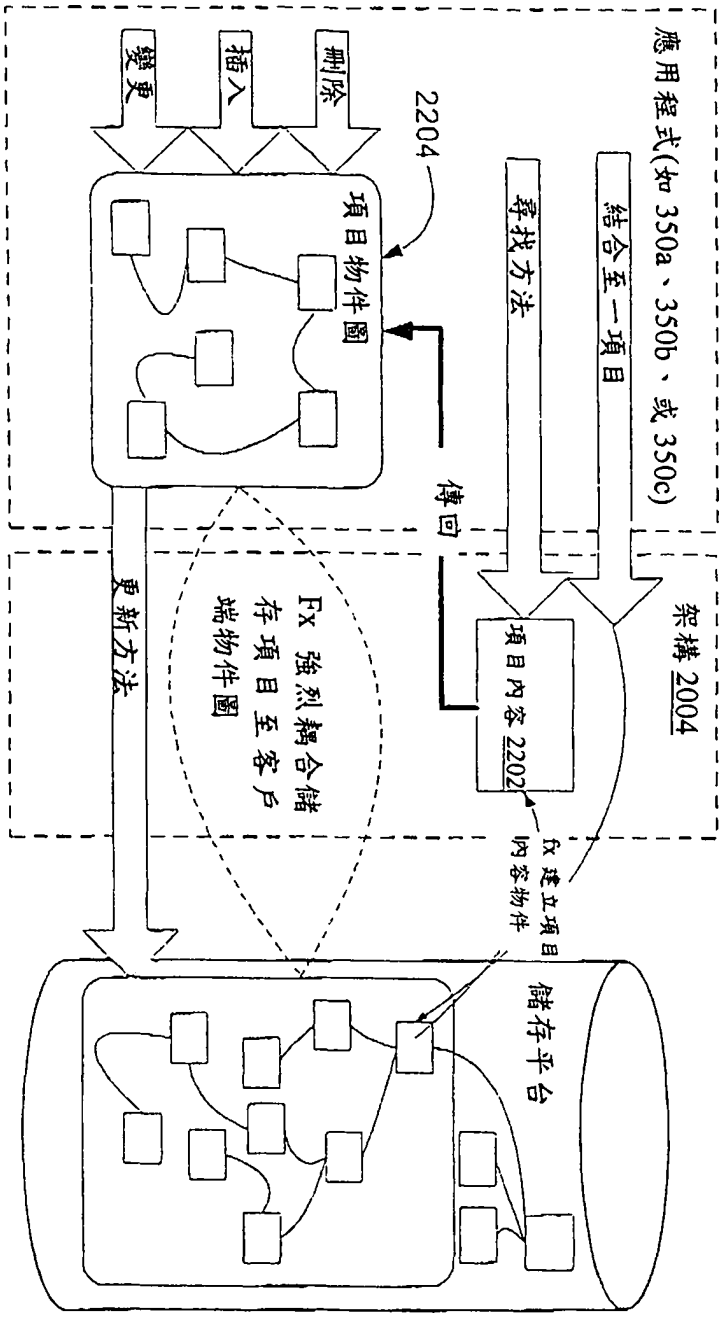
第 20 圖


項目
原則
人
使用者
群組
服務
呈現服務
群組服務
個人服務
接觸服務
AD服務
組織
家用
已知資料夾
資料夾
使用者資料夾
區域機器資料夾

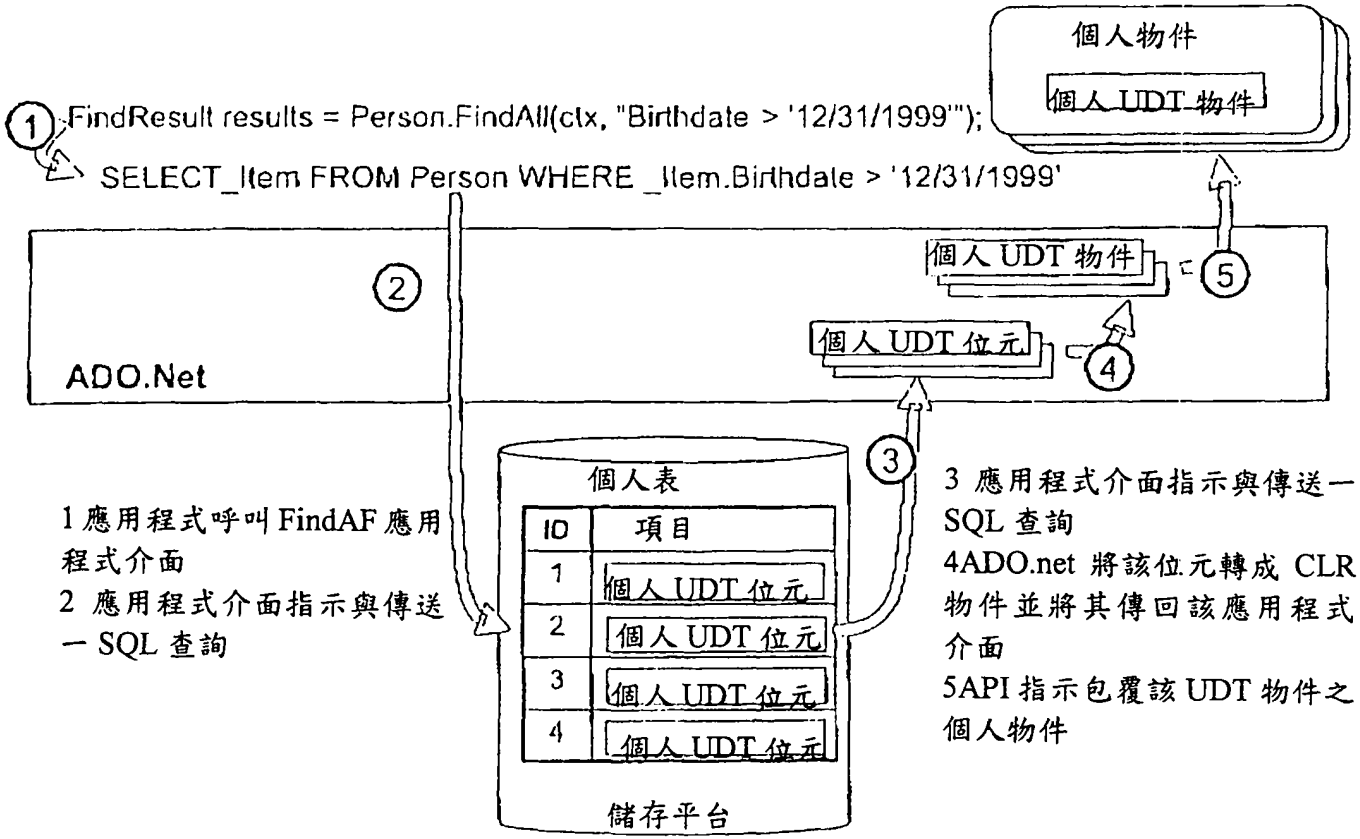
第21A圖


巢狀元件基礎
巢狀元件
身分金鑰
EAddress
範本
設定檔
關係
完整名稱
家族事件
安全性 ID
接觸 EAddress
基本呈現
範本關係
使用腳色
位置關係
家族事件位置關係
家庭位置關係
電話號碼
視窗呈現
SMTP 電子郵件位置
雇員資料
群組成員
立即訊息位置
組織位置關係
家庭成員資料
家族資料
配偶資料
小孩資料

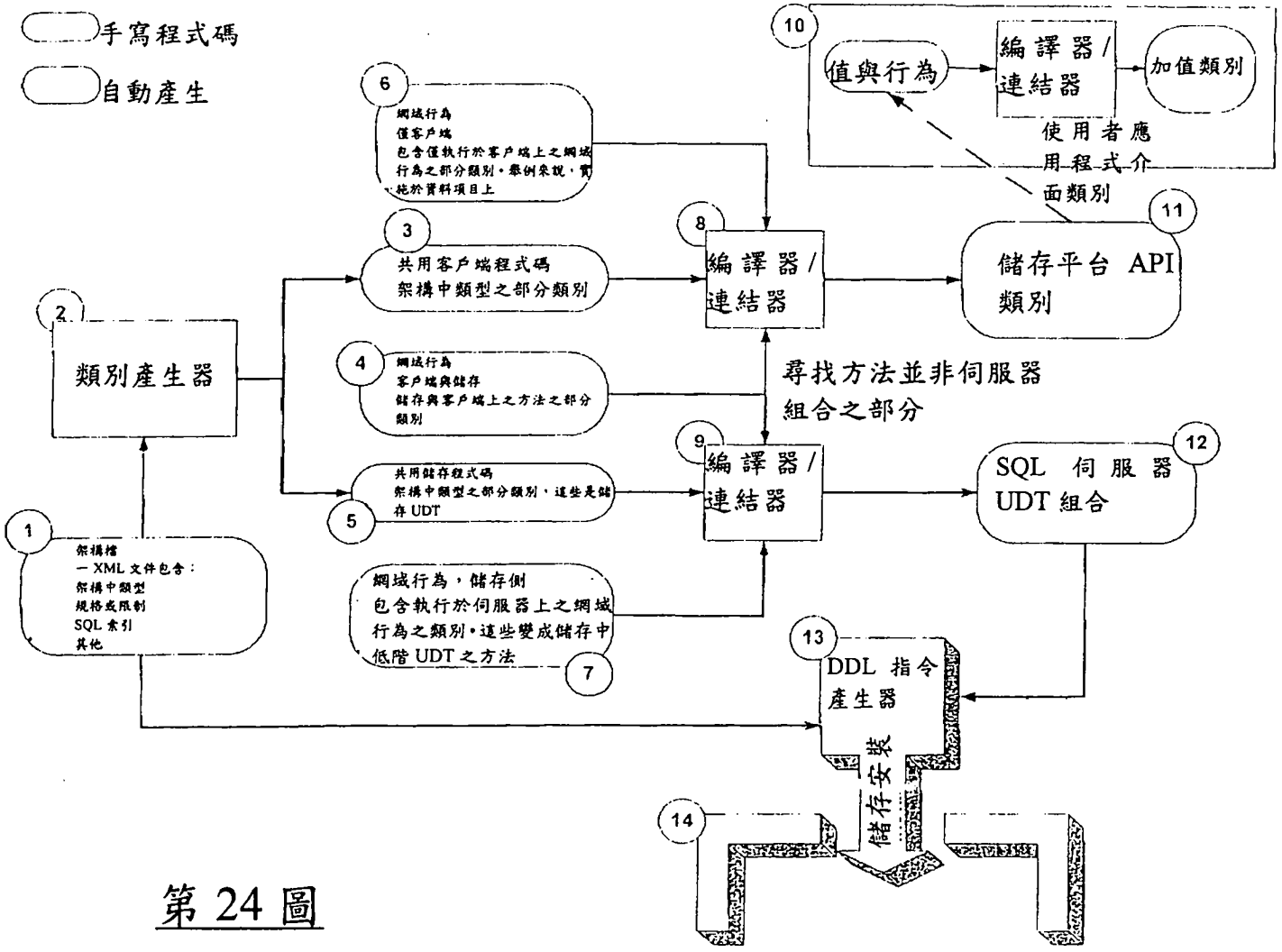
第 21B 圖


應用程式(如 350a、350b、或 350c)
結合至一項目
尋找方法
架構 2004
項目內容 2202
Fx 建立項目內容物件
2204
項目物件圖
刪除
插入
變更
傳回
更新方法
Fx 強烈耦合儲存項目至客戶端物件圖
儲存平台

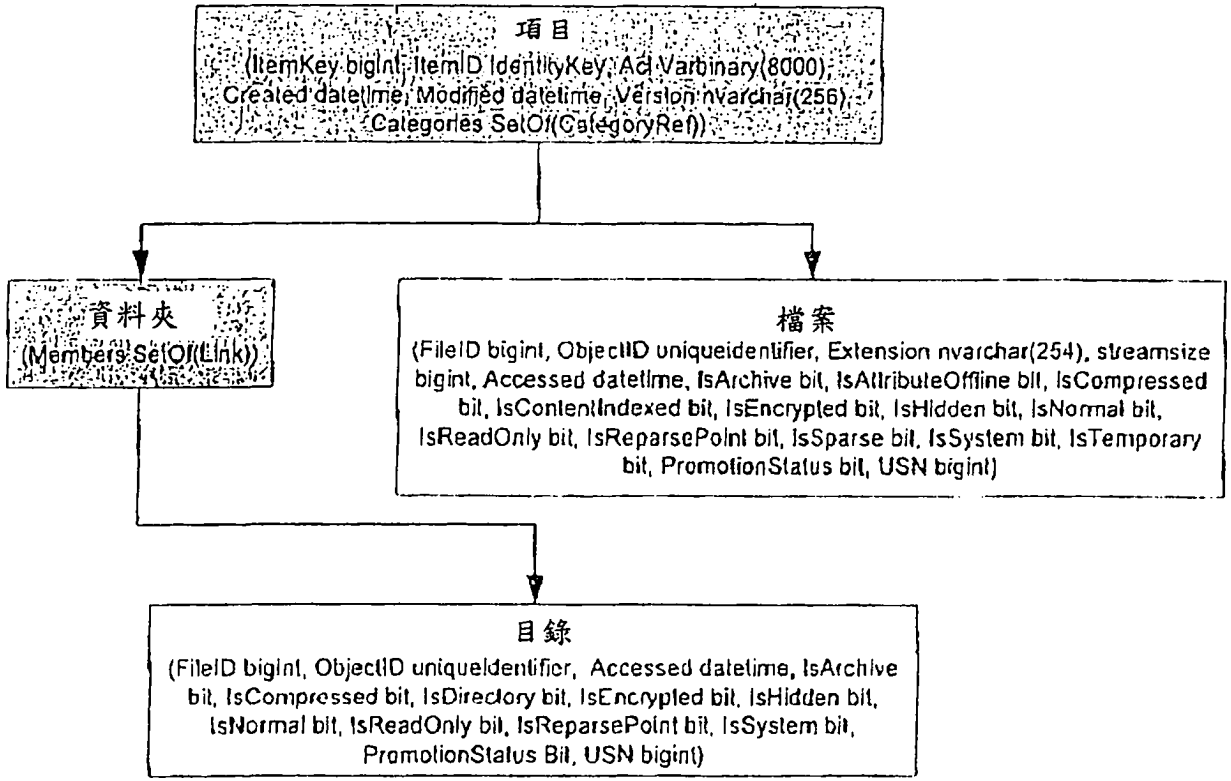
第 22 圖


個人物件
個人 UDT 物件
1 FindResult results = Person.FindAll(ctx, "Birthdate > '12/31/1999'");
SELECT_Item FROM Person WHERE _Item.Birthdate > '12/31/1999'
2
ADO.Net
個人 UDT 物件
5
個人 UDT 位元
4
3
個人表
ID
項目
1
個人 UDT 位元
2
個人 UDT 位元
3
個人 UDT 位元
4
個人 UDT 位元
儲存平台
1 應用程式呼叫 FindAF 應用程式介面
2 應用程式介面指示與傳送一 SQL 查詢
3 應用程式介面指示與傳送一 SQL 查詢
4ADO.net 將該位元轉成 CLR 物件並將其傳回該應用程式介面
5API 指示包覆該 UDT 物件之個人物件

第 23 圖


手寫程式碼
自動產生
1
架構檔
一 XML 文件包含：
架構中類型
規格或限制
SQL 索引
其他
2
類別產生器
3
共用客戶端程式碼
架構中類型之部分類別
4
網域行為
客戶端與儲存
儲存與客戶端上之方法之部分類別
5
共用儲存程式碼
架構中類型之部分類別，這些是儲存 UDT
6
網域行為
僅客戶端
包含僅執行於客戶端上之網域行為之部分類別，舉例來說，實施於資料項目上
7
網域行為，儲存側
包含執行於伺服器上之網域行為之類別，這些變成儲存中低階 UDT 之方法
8
編譯器/連結器
9
編譯器/連結器
10
值與行為
編譯器/連結器
加值類別
使用者應用程式介面類別
11
儲存平台 API 類別
12
SQL 伺服器 UDT 組合
13
DDL 指令產生器
14
儲存安裝
尋找方法並非伺服器組合之部分

第 24 圖

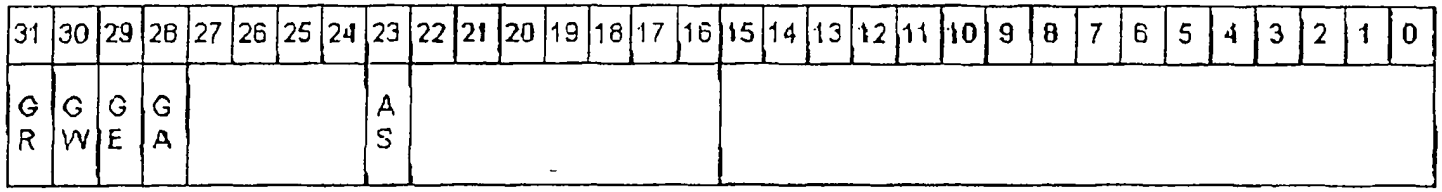


註：灰色方塊係定義於基本架構中之

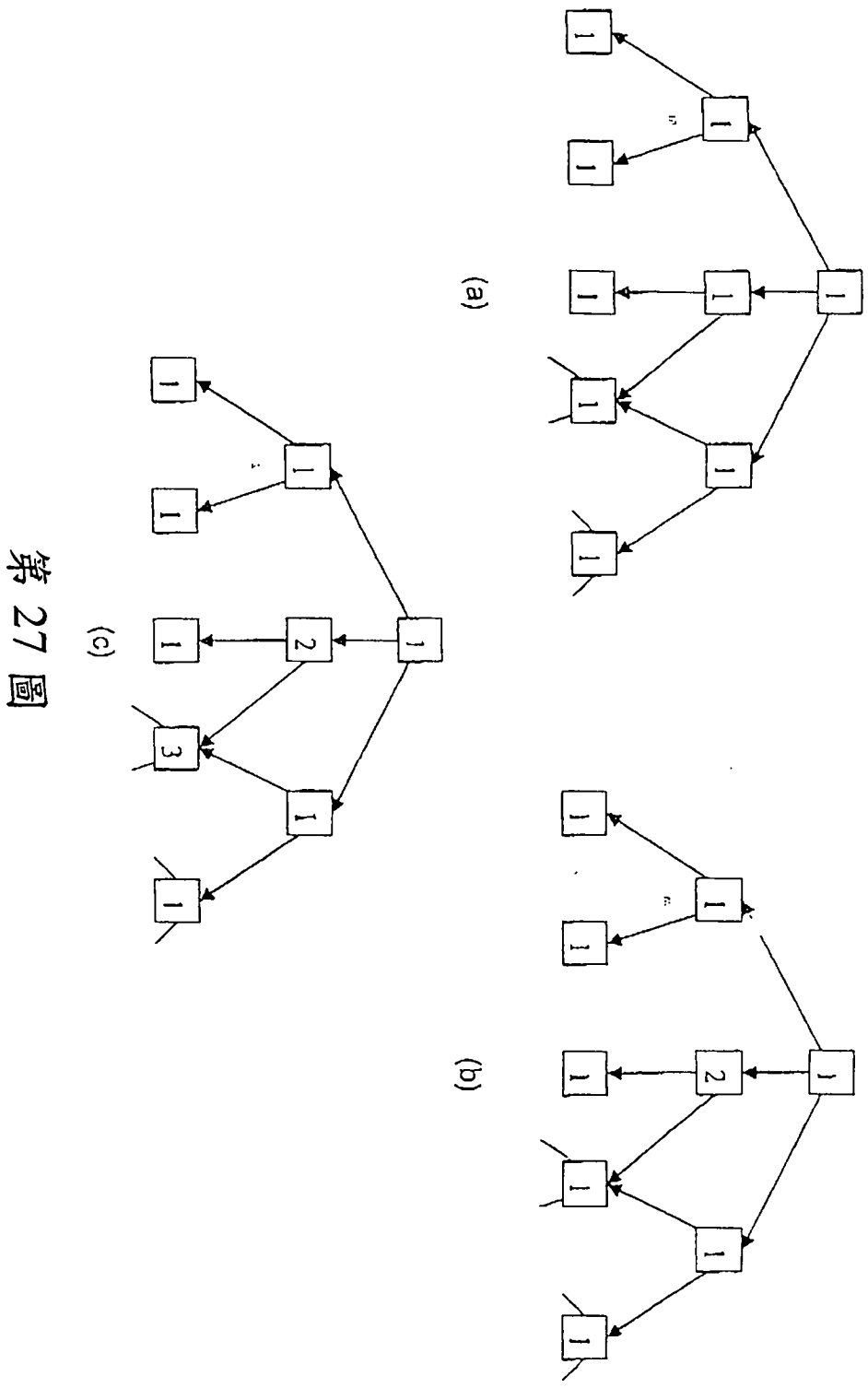
類別，但在此為了完整性加以顯示

定義於檔案架構中之項目

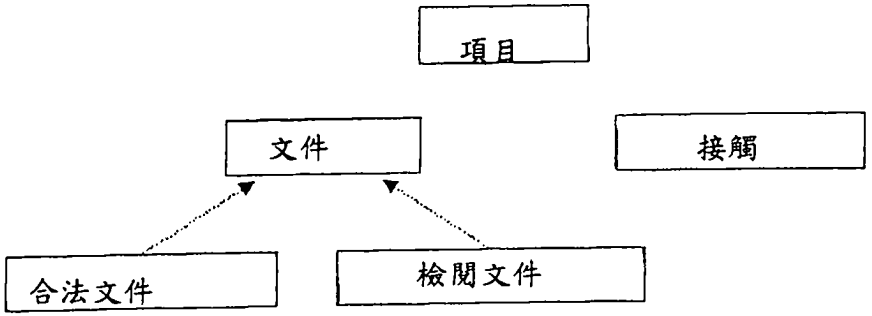
第 25 圖


31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
GR
GW
GE
GA
AS
GR --> Generic_Read
GW --> Generic_Write
GE --> Generic_Execute
GA --> Generic_ALL
AS --> Right to access SACL

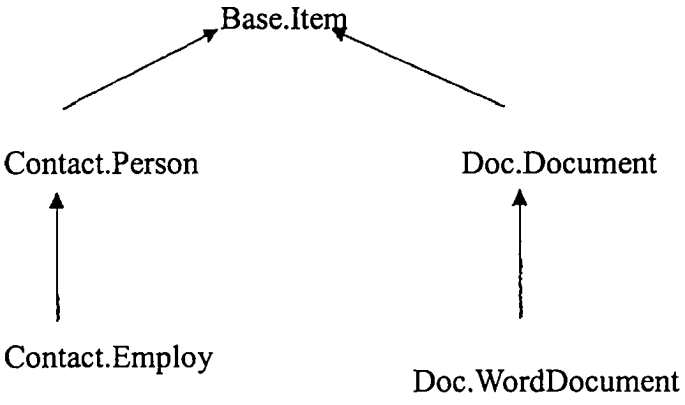
第 26 圖



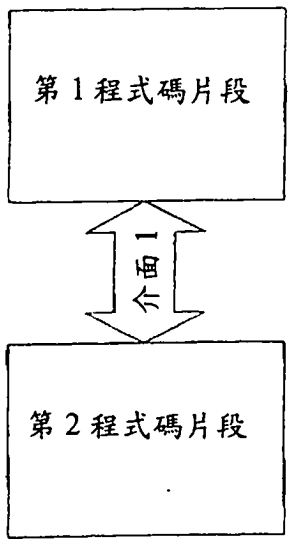
第 27 圖


項目
文件
接觸
合法文件
檢閱文件

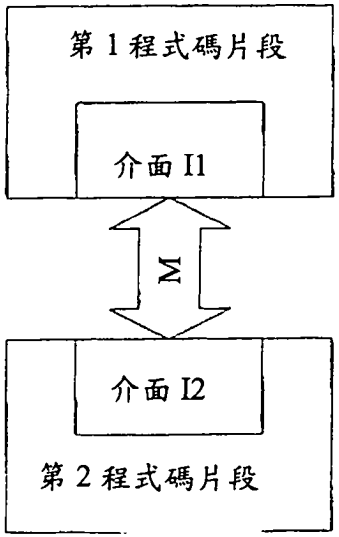
第 28 圖


Base.Item
Contact.Person
Doc.Document
Contact.Employ
Doc.WordDocument

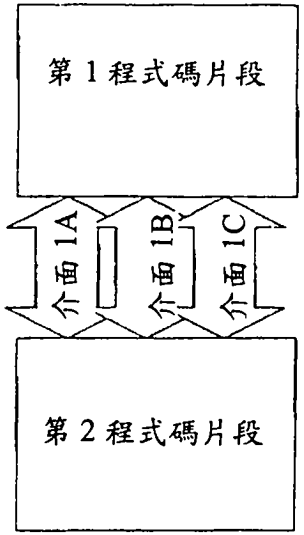
第 29 圖



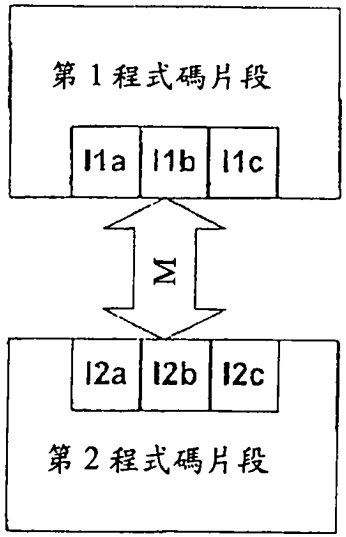
第 30A 圖



第 30B 圖

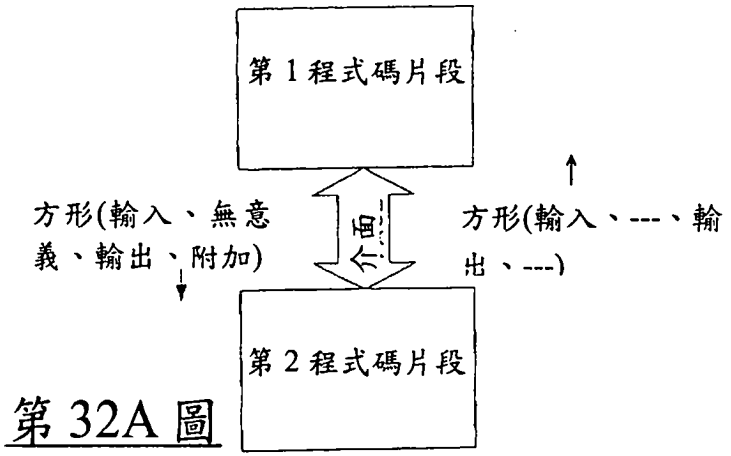


第 31A 圖

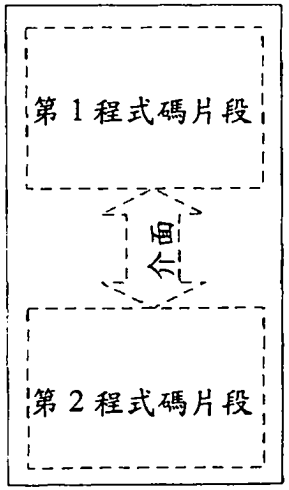


第 31B 圖

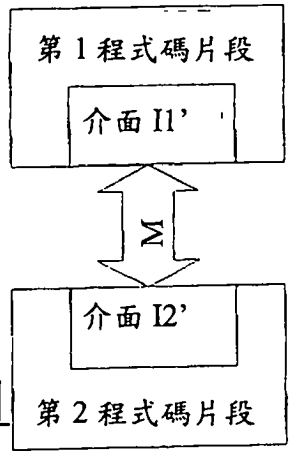
I337310



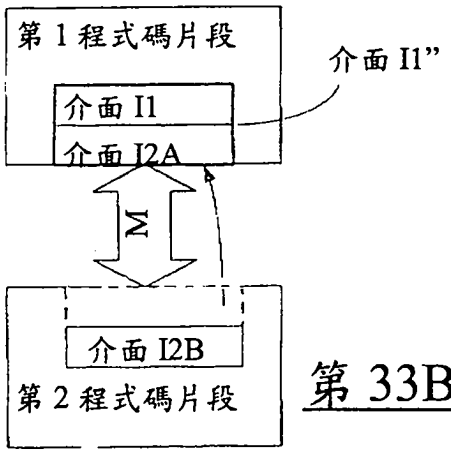
第 32A 圖



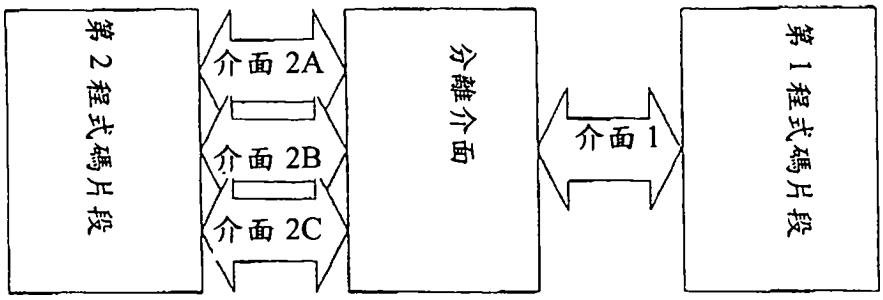
第 33A 圖



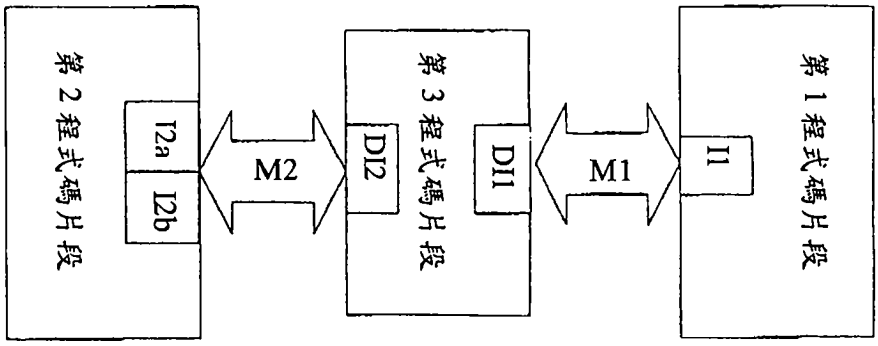
第 32B 圖



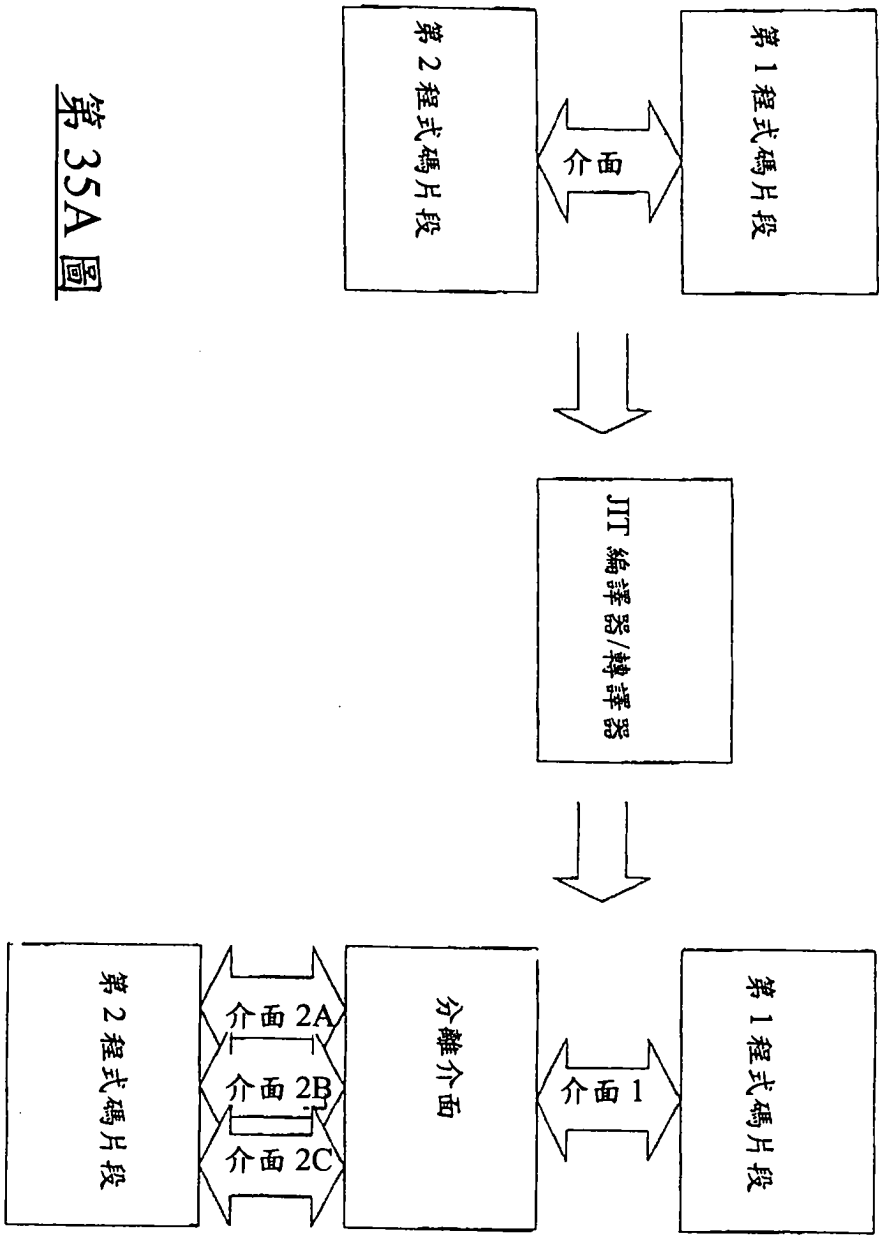
第 33B 圖


第 1 程式碼片段
介面 1
分離介面
介面 2A
介面 2B
介面 2C
第 2 程式碼片段

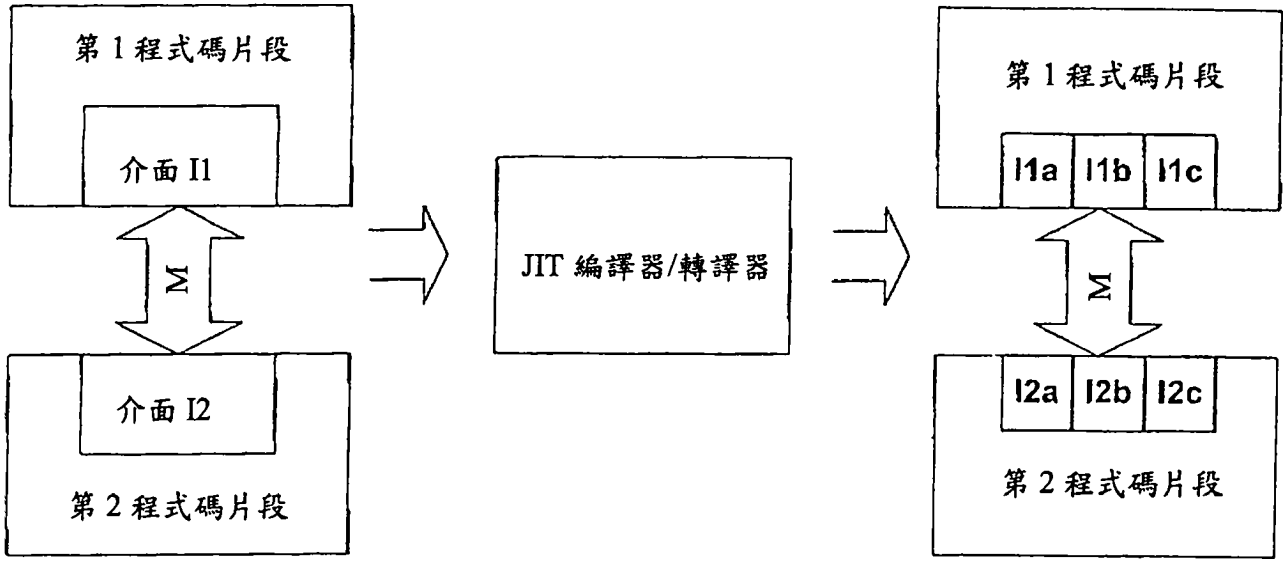
第 34A 圖


第 1 程式碼片段
I1
M1
DI1
第 3 程式碼片段
DI2
M2
I2a
I2b
第 2 程式碼片段

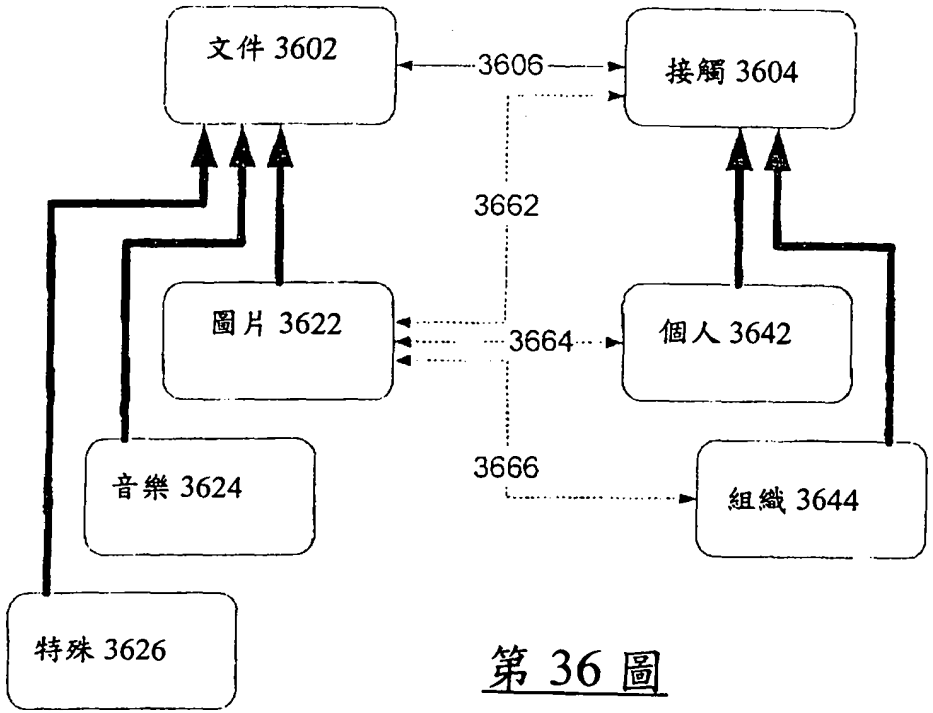
第 34B 圖


第 1 程式碼片段
介面
第 2 程式碼片段
JIT 編譯器/轉譯器
第 1 程式碼片段
介面 1
分離介面
介面 2A
介面 2B
介面 2C
第 2 程式碼片段

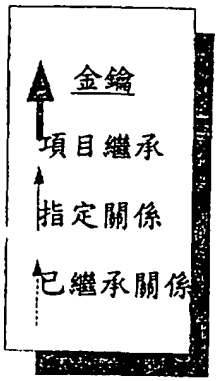
第 35A 圖

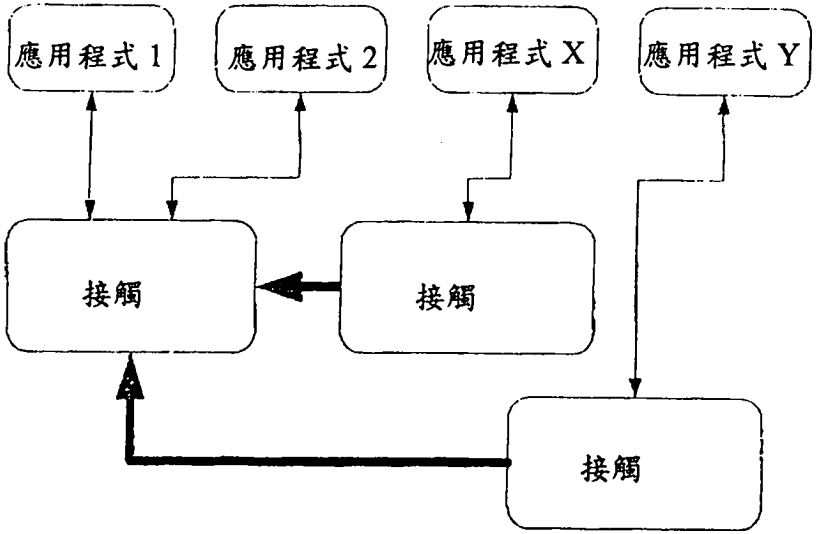

第 1 程式碼片段
介面 I1
M
介面 I2
第 2 程式碼片段
JIT 編譯器/轉譯器
第 1 程式碼片段
I1a
I1b
I1c
M
I2a
I2b
I2c
第 2 程式碼片段

第 35B 圖

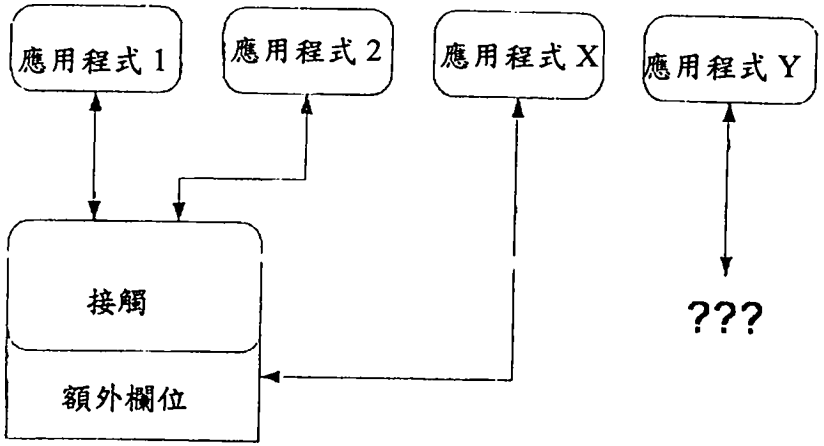

文件 3602
3606
接觸 3604
3662
圖片 3622
3664
個人 3642
音樂 3624
3666
組織 3644
特殊 3626

第 36 圖

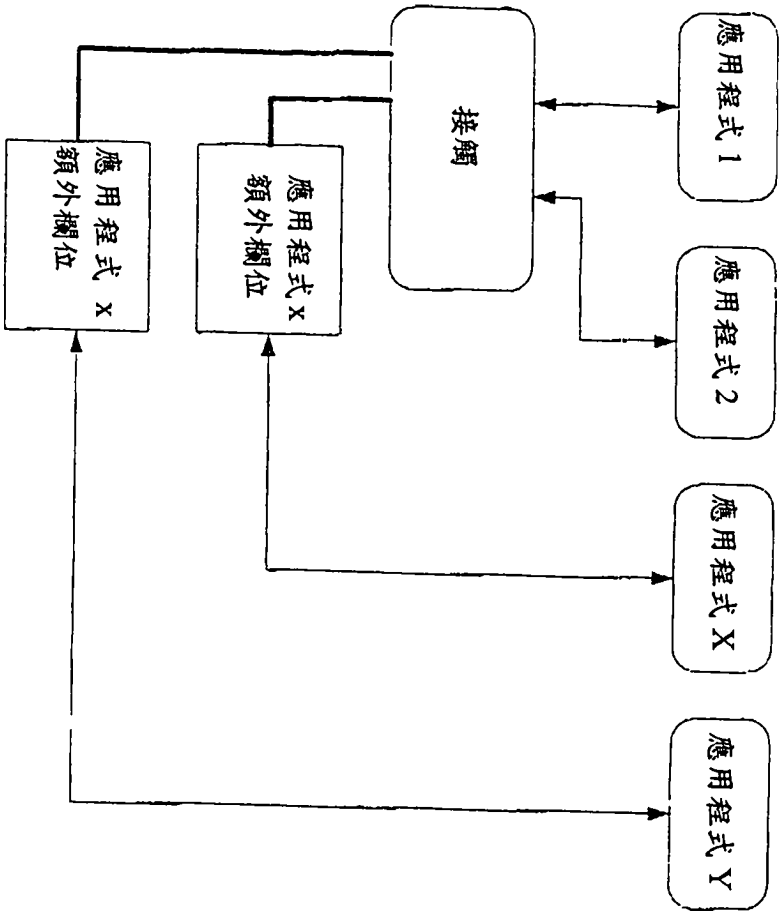

金鑰
項目繼承
指定關係
已繼承關係


應用程式 1
應用程式 2
應用程式 X
應用程式 Y
接觸
接觸
接觸

第 37A 圖


應用程式 1
應用程式 2
應用程式 X
應用程式 Y
接觸
額外欄位
???

第 37B 圖



第 37C 圖

第93122603號專利案96年2月修正

發明專利說明書

96.7.30

（本說明書格式、順序及粗體字，請勿任意更動，※記號部分請勿填寫）

※ 申請案號：93122603

※ 申請日期：93 年 7 月 28 日　　※IPC 分類：

公告本

一、發明名稱：（中文/英文）

可藉由硬體／軟體介面系統管理之資訊單元的擴充及繼承之系統及方法

SYSTEMS AND METHODS FOR EXTENSIONS AND INHERITANCE FOR UNITS OF INFORMATION MANAGEABLE BY A HARDWARE/SOFTWARE INTERFACE SYSTEM

二、申 請 人：（共 1 人）

姓名或名稱：（中文/英文）

美商．微軟公司

Microsoft Corporation

代表人：（中文/英文）

艾苹那諾爾 D 巴特萊

EPPENAUER, D. BARTLEY

住居所或營業所地址：（中文/英文）

美國華盛頓州列德蒙微軟路 1 號

One Microsoft Way, Building 8, Redmond, WA 98052-6399, U.S.A.

國　籍：（中文/英文）

美國/USA

三、發明人：（共 15 人）

姓　名：（中文/英文）

1.戴米羅司基貝金/DEMIROSKI, BEKIM

2.威特尼羅伯特 T./WHITNEY, ROBERT T.

3.湯普森 J.派崔克/THOMPSON, J. PATRICK

4.諾利阿尼 K./NORI, ANIL K.

5.阿佳沃沙米特/AGARWAL, SAMEET

6.賽利斯派德羅/CELIS, PEDRO

7.坎貝爾大衛 G./CAMPBELL, DAVID G.

8.泰瑞克 F.蘇尼/TEREK, SONER F.

9.卡麥隆金/CAMERON, KIM

10.史密斯瓦特 R./SMITH, WALTER R.

11.莎基戴倫 A./SHAKIB, DARREN A.

12.巴隆那山尼 H./BALLOU, NATHANIEL H.

13.阿佳亞史瑞尼凡摩斯 P./ACHARYA, SRINIVAMURTHY P.

14.羅門巴蘭沙丘/RAMAN, BALAN SETHU

15.史皮歐彼德 M./SPIRO, PETER M.

國　籍：（中文/英文）

1.馬其頓/MACEDONIA

2.美國/USA

3.英國/UK

4.美國/USA

5.印度/INDIA

6.美國/USA

7.美國/USA

8.土耳其/TURKEY

9.加拿大/CANADA

10.美國/USA

11.美國/USA

12.美國/USA

13.美國/USA

14.美國/USA

15.美國/USA

四、聲明事項：

☐ 主張專利法第二十二條第二項☐第一款或☐第二款規定之事實，其事實發生日期為：　年　月　日。

☑ 申請前已向下列國家（地區）申請專利：

【格式請依：受理國家（地區）、申請日、申請案號　順序註記】

☑ 有主張專利法第二十七條第一項國際優先權：

1. PCT；2003 年 8 月 21 日；PCT/US03/26144
2. 美國；2003 年 8 月 21 日；10//646,580
3. 美國；2003 年 10 月 24 日；10/693,574

☐ 無主張專利法第二十七條第一項國際優先權：

☐ 主張專利法第二十九條第一項國內優先權：

【格式請依：申請日、申請案號　順序註記】

☐ 主張專利法第三十條生物材料：

☐ 須寄存生物材料者：

國內生物材料　【格式請依：寄存機構、日期、號碼　順序註記】

國外生物材料　【格式請依：寄存國家、機構、日期、號碼　順序註記】

☐ 不須寄存生物材料者：

所屬技術領域中具有通常知識者易於獲得時，不須寄存。

九、發明說明：

【發明所屬之技術領域】

本發明係有關於一種資訊儲存與取出之領域，且由其關於一種用以於一計算型系統中組織、搜尋、與共用不同類型資料之主動儲存平台。本發明之各種不同實施例導向由一硬體/軟體介面系統利用擴充與繼承方法以管理資料。

【先前技術】

個別碟片容量已經在過去十年間大約每年成長百分之七十(70%)。莫爾定律精確的預測中央處理單元(CPU)的能力的急速成長也已經在近年來發生。有線與無線技術已提供了龐大的連結與頻寬。假設目前的趨勢繼續下去，在幾年之內，平均膝上型電腦將具有大約一兆位元(TB)儲存，並包含幾百萬個檔案，且500百萬位元(GB)裝置將會變得很常見。

消費者主要使用其電腦於通訊或組織個人資訊，無論其為傳統個人電腦管理員(PIM)類型資料或媒體，例如數位音樂或照片。數位內容的量以及儲存原始位元的能力已經大大增加；然而可讓消費者用於組織與統一此資料之方法並沒有跟上腳步。有知識的工作者花費大量的時間管理與共享資訊，也某些研究估計有知識的工作者花費其時間的15-25%於沒有生產力的資訊相關動作上。其他研究估計一典型有知識的工作者每天約花費2.5小時於搜尋資訊上。

研發人員與資訊科技(IT)部份投資大量的時間與金錢於建立其自己的共用儲存摘要之資料儲存，以呈現例如

人、地、時、與事件之此類事項。此不僅導致重複的工作，且亦建立沒有共用搜尋或共享此資料之機制之共用資料島。只要想許多通訊錄如何在今日存在於執行 Microsoft Windows 作業系統之一電腦上。許多應用程式，例如電子郵件客戶端與個人財務程式，維持個人通訊錄，且極少共用此程式個別維護之通訊錄資料之應用程式。因此，一財務程式(如同 Microsoft Money)不以一電子郵件接觸資料夾(如 Microsoft Outlook 中者)中維護之位址共用付費者的位址。實際上，許多使用者具有多個裝置與邏輯上應同步化其本身與跨各種不同額外來源間之個人資料，包含行動電話至商業服務，例如 MSN 與 AOL；然而，共用文件之合作藉由附加文件至電子郵件訊息而大量達成—也就是手動而無效率。

此缺乏合作的理由是電腦系統中組之資訊之傳統方法著重於使用檔案-資料夾-與-目錄-型系統(「檔案系統」)，以根據用於儲存檔案之儲存媒體之實體組織以組織複數檔案至資料夾目錄階層中。於 1960 年代間研發之 Multics 作業系統可歸功於使用檔案、資料夾、及目錄以管理作業系統層的資料的可儲存單元的先鋒。尤其是，Multics 使用檔案階層中之符號位址(藉此引入檔案路徑之點子)，其中檔案之實體位址對使用者並非透明(應用程式與使用者端)。此檔案系統完全未考慮任何個別檔案的檔案格式及於檔案系統層不相關的檔案間的關係(也就是指階層中檔案位置之外者)。由於 Multics 的問世，可儲存資料已經於作業系

統層規劃至檔案、資料夾、與目錄中。這些檔案一般包含實施於由該檔案系統所維持之一特殊檔案中之檔案階層本身(該「目錄」)。接著，此目錄維護對應至該目錄中所有其他檔案與該階層中此檔案之節點位置(在此稱為資料夾)之一項目清單。此係約四十年來的習知技藝。

然而，提供位於該電腦的實體儲存系統中之一合理資訊呈現方式，一檔案系統卻是該實體儲存系統之一抽象化，且因此該檔案之利用需要該使用者操控(具有內容、特性、及與其他單元之關係之單元)與該作業系統提供(檔案、資料夾、及目錄)間之一間接手段層(轉譯)。因此，使用者(應用程式與/或使用者端)沒有選擇，只能強迫資訊單元至一檔案系統結構中，即使這麼做沒有效率、不一致、或不想要。此外，既有檔案系統了解極少關於儲存於個別檔案中之資料結構，也由於此，大部份的資訊維持鎖定於僅可存取(及可理解)於撰寫其之應用程式之檔案中。因此，此缺乏資訊之架構描述及管理資訊之機制導向具有極少該個別儲存間共用資料之資料儲存之建立。舉例來說，許多個人電腦(PC)使用者具有包含關於於某程度上互動之人之資訊之五個不同儲存—舉例來說，Outlook Contacts、線上帳戶位址、Windows Address Book、Quicken Payees、及立即訊息(IM)兄弟清單—由於組織檔案對這些 PC 使用者呈現重大挑戰。由於大部份既有檔案系統利用用以組織檔案與資料夾之一巢狀資料夾比喻，當檔案數量增加，則用於維護有彈性且有效率之一組織架構所需之努力使人畏

縮。在此情況中，具有單一檔案之多個分類會是非常有用的；然而，利用既有檔案系統中之硬或軟連結係累贅且難以維護的。

過去所做出之幾個不成功嘗試已點出檔案系統的缺點。某些先前的嘗試已經包含使用內容可定址記憶體提供一機制，其中資料可由內容而非由實體位址加以存取。然而，這些努力已證明不成功，由於內容可定址記憶體已經證明為對裝置的小範圍使用是很有用的，例如快取記憶體與記憶體管理單元，基於各種不同原因還不可能用於裝置的大型使用，例如實體儲存媒體，且因此此一解決方法簡直等於不存在。利用物件導向資料庫(OODB)系統之其他嘗試已被實行，但在特徵化強烈資料庫特徵與良好非檔案呈現方式中，這些嘗試在處理檔案呈現方式中是無效的，也無法根據該硬體/軟體介面系統層的階層結構複製速度、效率、及簡化檔案與資料夾。其他努力，例如嘗試使用SmallTalk(及其他衍生物)，證明處理檔案與非檔案呈現方式相當有效，但缺乏必須有效組織與利用各種不同資料檔案間存在之關係之資料庫特性，且因此此系統之全面效能無法被接受。又其他嘗試使用 BeOS(及其他此作業系統研究)證明處理非檔案呈現方式是不足的—與傳統檔案系統相同之核心缺點—除了可足夠呈現檔案，且提供相同必須資料庫特性之外。

資料庫技術係存在類似挑戰之技藝之另一區域。舉例來說，該相關資料庫模型已在商業上大大成空，事實上獨

立軟體廠商(ISV)一般練習相關資料庫軟體產品(例如Microsoft SQL伺服器)中有效之功能之一小部份。然而，大部份與此一產品之一應用程式互動係為簡單「獲得」與「放入」形式。對此有許多快速明顯之理由，例如不可論知之平台或資料庫—常常被忽略之一關鍵理由係該資料庫不需要提供一主要商業應用程式廠商真正需要之確實抽象化。舉例來說，該真實世界具有「項目」之標記，例如「顧客」或「訂單」(以及一訂單嵌入「線上項目」，作為其本身之項目)，相關資料庫僅代表表格與行。因此，該應用程式可能想要具有各方面之一致性、鎖定、安全性、及/或觸發該項目層(以命名一些)，一般的資料庫僅於該表格/行層提供這些特性。若各項目獲得映射致該資料庫中某些表格中之單一行，此會運作良好，在具有多行項目之一訂單之一情況中，可能有一項目為何真正獲得映射至多個表格之理由，以及當在該情況中時，該簡單相關資料庫系統不提供相當正確的抽象化。因此，一應用程式必須建立該資料庫頂端之邏輯，以提供這些基本抽象化。換句話說，該基本相關模型不提供一足夠平台以儲存資料，其中高階應用程式可容易被研發，由於該基本相關模型需要該應用程式與該儲存系統間之一間接手段層，其中該資料之語義結構僅可見於特定範例中之應用程式中。某些資料庫廠商建立更高階功能至其產品中—例如提供物件相關功能、新組織模型等等—沒有一者提供需要的全面解決方法，其中一真正的全面解決方法係為有用的網域抽象化(例如「個人」、

「位置」、「事件」等等)提供有用的資料模型抽象化(例如「項目」、「擴充」、「關係」等等)。

有鑑於前述既有資料儲存與資料庫技術中之缺乏，需要有一新儲存平台提供一改良能力，以組織、搜尋、與共享一電腦系統中所有類型資料—擴充與延伸該資料平台超出既有檔案系統與資料庫系統，且設計以儲存所有類型資料之一儲存平台。本發明結合之前在此併入參照之相關發明滿足此需求。尤其是，本發明為由一硬體/軟體介面系統操控之物件提供擴充與繼承之方法。

本發明所聲明之優先權係美國專利申請案第10/693,574號(律師文件號碼 MSFT-2847)，標題為「可藉由硬體/軟體介面系統管理之資訊單元的擴充及繼承之系統及方法」；申請於2003年8月21日之美國專利申請案第10/646,580號(律師文件號碼 MSFT-2735)，標題為「於一項目型儲存平台中資料模型化之系統與方法」；及申請於2003年8月21日之國際專利申請案第PCT/US 03/26144號(律師文件號碼 MSFT-2779)，標題為「於一項目型儲存平台中資料模型化之系統與方法」；其內容在此併入參照。

本案主要關於下列共同指派申請案所揭示者，其整體內容在此併入本發明參照(及為了方便，在此摘要其部份)：申請於2003年8月21日之美國專利申請案第10/647,058號(律師文件號碼 MSFT-1748)，標題為「可藉由硬體/軟體介面系統但獨立於實體呈現方式之外管理之資訊單元的呈現之系統及方法」；申請於2003年8月21

日之美國專利申請案第 10/646,941 號(律師文件號碼 MSFT-1749)，標題為「可藉由來自其實體組織之硬體/軟體介面系統管理之資訊單元的分離之系統及方法」；申請於 2003 年 8 月 21 日之美國專利申請案第 10/646,940 號(律師文件號碼 MSFT-1750)，標題為「可藉由硬體/軟體介面系統管理之資訊單元的組織之一基本架構之實施方式之系統及方法」；申請於 2003 年 8 月 21 日之美國專利申請案第 10/646,632 號(律師文件號碼 MSFT-1751)，標題為「可藉由硬體/軟體介面系統管理之資訊單元的組織提供一項階結構之一核心架構之實施方式之系統及方法」；申請於 2003 年 8 月 21 日之美國專利申請案第 10/646,645 號(律師文件號碼 MSFT-1752)，標題為「可藉由硬體/軟體介面系統管理之呈現資訊單元間關係之系統及方法」；申請於 2003 年 8 月 21 日之美國專利申請案第 10/646,575 號(律師文件號碼 MSFT-2733)，標題為「以一項目型儲存平台介面化應用程式之系統及方法」；申請於 2003 年 8 月 21 日之美國專利申請案第 10/646,646 號(律師文件號碼 MSFT-2734)，標題為「用以組織、搜尋、及共享資料之儲存平台」；申請於 2003 年 10 月 24 日之美國專利申請案第 10/692,779 號(律師文件號碼 MSFT-2829)，標題為「可藉由硬體/軟體介面系統管理之組織資訊單元之一數位影像架構之實施方式之系統及方法」；申請於 2003 年 10 月 24 日之美國專利申請案第 10/692,515 號(律師文件號碼 MSFT-2844)，標題為「可藉由硬體/軟體介面系統管理之資

訊單元提供同步化服務之系統及方法」；申請於與本案同一日之美國專利申請案第 10/692,508 號(律師文件號碼 MSFT-2845)，標題為「可藉由硬體/軟體介面系統管理之資訊單元提供關聯與階層性同步化服務之系統及方法」；以及申請於 2003 年 10 月 24 日之美國專利申請案第 10/693,362 號(律師文件號碼 MSFT-2846)，標題為「可藉由硬體/軟體介面系統管理之資訊單元之同步化架構之實施方式之系統及方法」。

【發明內容】

下列簡介提供描述於之前在此併入參照之相關發明之內容中(該「相關發明」)之本發明各方面之一簡介。此簡介並非用於提供本發明之所有重要方面之一徹底描述，亦非用於定義本發明之範圍。而是，此簡介用於做為下列詳細描述與圖式之一介紹。

本發明以及該相關發明統合導向用以組織、搜尋、及共享資料之一儲存平台。本發明之儲存平台擴充與延伸資料儲存之概念超出既有檔案系統與資料庫系統之外，且設計以做為包含結構化、非結構化、或半結構化資料之所有資料類型之儲存。

本發明之儲存平台包含實施於一資料庫引擎上之一資料儲存。該資料庫引擎包含具有物件相關擴充之一相關資料庫引擎。該資料儲存實施支援組織、搜尋、共享、同步化、及資料安全性之一資料模型。描述於架構中之特定類型資料及該名台提供一機制以擴充該架構組以定義新類型

資料(基本上為由該架構提供之基本類型之子類型)。一同步化功能促進使用者或系統間之資料共享。提供類似檔案系統之功能允許具有既有檔案系統之資料儲存之相互操作，但並無此傳統檔案系統之限制。一變更追蹤機制提供該資料庫儲存之追蹤變更之能力。該儲存平台另包含一組應用程式介面，允許應用程式存取所有前述儲存平台之功能，並存取該架構中描述之資料。

由該資料儲存實施之資料模型定義代表項目、元件、及關係之資料儲存單元。一項目係可儲存於一資料儲存中之一資料單元，且可包含一或多個元件或關係。一元件係包含一或多個欄位(在此亦稱為一屬性)之一類型例。一關係係二項目間之一連結。(如在此所使用者，這些及其他特定項目可使用大寫，以自其他用於類似近接性中之其他詞中突顯出來；然而，並非用於區別一大寫詞，如「項目(Item)」，且當非大寫時為相同詞，如「項目(item)」，且不應假設或暗示此有所不同。)

該電腦系統另包含複數項目，其中各項目由可由一硬體/軟體介面系統操控之一不連續的可儲存資訊單元組成；由該項目之一組織結構組成之複數項目資料夾；及用以操控複數項目之一硬體/軟體介面系統，且其中各項目屬於至少一項目資料夾，且可屬於超過一個項目資料夾。

一項目或該項目屬性值之某些部份可動態計算，此係相對於由一永久儲存取出者而言。換句話說，該硬體/軟體介面系統不需要該項目儲存，且支援特定動作，例如列舉

該目前項目組之能力，或取出給定該儲存平台之其識別符之一項目(其更完整描述於描述該應用程式介面，或 API 之各節中)—舉例來說，一項目可為一行動電話之目前位置或一溫度感應器上讀取之溫度。該硬體/軟體介面系統可操控複數項目，且另可包含由該硬體/軟體介面系統管理之父數個關係互連之項目。

該電腦系統之一硬體/軟體介面系統另包含一核心架構，以定義該硬體/軟體介面系統了解且可直接以一預定及可預測方式處理之一組核心項目。為了操控複數項目，該電腦系統與複數關係互連，並於該硬體/軟體介面系統層管理該關係。

該儲存平台之 API 為定義於該組儲存平台架構中之各項目、項目擴充、及關係提供資料類別。除此之外，該應用程式介面提供一組架構類別，為該資料類別定義一組共用行為，且與該資料類別一同為該儲存平台 API 提供該基本程式設計模型。該儲存平台 API 提供一簡化查詢模型，允許應用程式設計人員根據該資料儲存中項目之各種不同屬性，以隔離該應用程式設計人員與該基本資料庫引擎之查詢語言之細節之方式形成查詢。該儲存平台 API 亦收集由一應用程式所做至一項目之變更，且接著將其組織至由該資料庫引擎(或任何種類儲存引擎)要求之正確更新中，其中實施該資料儲存。此允許應用程式設計人員做出變更至記憶體中之一項目，而保留資料儲存更新至該 API 之複雜性。

透過其共用儲存基礎與架構化資料，本發明之儲存平台允許為消費者、有之事的工作者與專家更有效的應用程式研發。其提供一豐富可擴充應用程式介面，不只可用於其資料模型中繼承之功能，亦可囊括且擴充既有檔案系統與資料庫存取方法。

鑑於此互相相關發明之拱形架構(在實施方式之第 II 節中詳細描述)中，本發明尤其導向使用擴充以擴充項目與屬性類型之功能，以及使用繼承以促進有效搜尋與組織相關項目之間(在實施方式之第 III 節中詳細描述)。本發明之其他特性與優點可自下列本發明實施方式與附圖明顯獲得。

【實施方式】

1. 簡介

本發明之標的描述具有符合法定需求之特徵。然而，說明本身並非用於限制本專利之範圍。而是，發明人覺得宣示之標的亦可於其他方法中實施，包含與此文件中描述相似之不同步驟或步驟組合，結合其他發明或未來之技術。此外，雖然「步驟」一詞可能在此用於暗示實施方法之不同元件，該詞不應解釋為暗示在此揭示之各種不同步驟之中或之間之任何特定順序，除非當個別步驟之順序明確加以描述之外。

A. 範例計算環境

本發明之許多實施利可執行於一電腦上。第 1 圖與下列討論用於提供本發明可實施之適用計算環境之簡要一般

說明。雖然不需要，本發明之各種不同方面可描述於正由一電腦，例如一客戶端工作站或一伺服器，執行之電腦可執行指令之通用內容中，例如程式模組。一般來說，程式模組包含執行特定工作或實施特定抽象資料類型之副程式、程式、物件、元件、資料結構等等。此外，本發明可以其他電腦系統配置實施，包含手持式裝置、多處理器系統、微處理器型或可程式化消費者電子元件、網路 PC、微電腦、主機電腦等等。本發明亦可實施於工作藉由透過一通訊網路連結之遠端處理裝置執行之分散式計算環境中。在一分散式計算環境中，程式模組可放置於區域與遠端記憶體儲存裝置中。

如第 1 圖中所示，一範例通用目的計算系統包含一傳統個人電腦 20 或類似者，包含一處理單元 21、一系統記憶體 22、及耦合包含系統記憶體之各種不同系統元件至該處理單元 21 之一系統匯流排 23。該系統匯流排 23 可為任何類型之匯流排結構，包含一記憶體匯流排或記憶體控制器、一週邊匯流排、及使用任何匯流排架構之一區域匯流排。該系統記憶體包含唯讀記憶體(ROM)24 及隨機存取記憶體(RAM)25。包含例如於開機期間幫助轉換該個人電腦 20 中元件間之資訊之基本程式之一基本輸出/入系統 26(BIOS)儲存於 ROM 24 中。該個人電腦 20 另可包含用以讀取與寫入一硬碟(未顯示)之一硬碟裝置 27、用以讀取與寫入一可移除磁碟 29 之一磁碟裝置 28、及用以讀取與寫入一可移除光碟 31，例如一 CD ROOM 或其他光學媒體，

之一光碟裝置30。該硬碟裝置27、磁碟裝置28、及光碟裝置30分別藉由一硬碟裝置介面32、一磁碟裝置介面33、及一光碟裝置介面34連結至系統匯流排23。該裝置與其相關電腦可讀取媒體提供電腦可讀取指令、資料結構、程式模組及該個人電腦20之其他資料之非揮發性儲存。雖然在此描述之範例環境實施一硬碟、一可移除磁碟29及一可移除光碟31，熟知該項技藝人士應了解可儲存可由一電腦存取之資料之其他類型電腦可讀取媒體，例如磁匣、快閃記憶卡、數位影碟、伯努利匣、隨機存取記憶體(RAM)、唯讀記憶體(ROM)等等亦可用於該範例作業環境中。同樣的，該範例環境亦可包含許多類型之監控裝置，例如熱感應器與安全性或火警系統，及其他資訊來源。

許多程式模組可儲存於該硬碟、磁碟29、光碟31、ROM 24或RAM 25上，包含一作業系統35、一或多個應用程式36、其他程式模組37與程式資料38。一使用者可透過輸入裝置輸入指令與資訊至該個人電腦20中，例如一鍵盤40與指標裝置42。其它輸入裝置(未顯示)可包含一麥克風、搖桿、遊戲墊、衛星碟、掃描器等等。這些與其他輸入裝置常常透過耦合至該系統匯流排之一序列埠介面46連結至該處理器單元21，但亦可由其他介面加以連結，例如一並列埠、遊戲埠或通用序列埠(USB)。一螢幕47或其他類型顯示裝置亦可經由一介面連結至該系統匯流排23，例如一視訊轉接器48。除了該螢幕47之外，個人電腦一般包含其他週邊輸出裝置(未顯示)，例如喇叭與印表

機。第 1 圖之範例系統一包含一主機轉接器 55、小型電腦系統介面(SCSI)匯流排 56、及連結至該 SCSI 匯流排 56 之一外部儲存裝置 62。

該個人電腦 20 可利用邏輯連結至一或多個遠端電腦操作於一網路環境中，例如一遠端電腦 49。該遠端電腦 49 可為另一個人電腦、一伺服器、一路由器、一網路 PC、一對等裝置或其他共用網路節點，且一般包含上述關於該個人電腦 20 之許多或所有元件，雖然僅一記憶體儲存裝置 50 舉例說明於第 1 圖中。第 1 圖中所述之邏輯連結包含一區域網路(LAN)51 與一廣域網路(WAN)52。此網路環境常見於辦公室、專業型電腦網路、內部網路與網際網路。

當使用於一 LAN 網路環境中時，該個人電腦透過一網路介面或轉接器 53 連結至該 LAN 51。當使用於一 WAN 網路環境中時，該個人電腦 20 一般包含一數據機 54 或其他用以建立透過該廣域網路 52 之通訊之構件，例如網際網路。可為內建或外接之數據機 54 經由該序列埠介面 46 連結至該系統匯流排 23。在一網路環境中，關於該個人電腦 20 描述之程式模組或其部分可儲存於該遠端記憶體儲存裝置中。應了解所示之網路連結僅為範例，且亦可使用其他建立該電腦間之一通訊連結之構件。

如第 2 圖之方塊圖中所示，一電腦系統 200 可粗略分成三個元件群：該硬體元件 202、該硬體/軟體介面系統元件 204、及該應用程式元件 206(在此特定內容中亦稱為該「使用者元件」或「軟體元件」)。

在一電腦系統200之各種不同實施例中，且參照回第1圖，該硬體元件202可包含該中央處理單元(CPU)21、該記憶體(ROM 24與RAM 25)、該基本輸出/入系統(BIOS)26、及各種不同輸入/輸出(I/O)裝置，例如一鍵盤40、一滑鼠42、一螢幕47、及/或一印表機(未顯示)。該硬體元件202為該電腦系統200包含該基本實體架構。

該應欲程式元件206包含各種不同軟體程式，包含但不限於編譯器、資料庫系統、文書處理器、商用程式、視訊遊戲等等。應用程式提供電腦資源用於解決問題、提供解決方法、及為各種不同使用者(機器、其他電腦系統、及/或使用者端)處理資料之構件。

該硬體/軟體介面系統元件204包含(且在某些實施例中可主要組成)一作業系統，在大部分情況中其本身包含一殼與一核心。一「作業系統」(OS)係一特殊程式，作用如同應用程式與電腦硬體間之一媒介物。該硬體/軟體介面系統元件204亦可包含一虛擬機器管理員(VMM)、一共用語言執行時間(CLR)或其功能相同者、一Java虛擬機器(JVM)或其功能相同者、或其他取代或加入一電腦系統中之作業系統中之此軟體元件。一硬體/軟體介面系統之目的係提供一使用者可執行應用程式之一環境。任何硬體/軟體介面系統之目標係使該電腦系統方便使用，以及以一有效率的方法利用該電腦硬體。

該硬體/軟體介面系統一般於開機時載入於一電腦系統中，且其後管理該電腦系統中所有應用程式。該應用程

式藉由經由一應用程式介面(API)要求服務與該硬體/軟體介面系統互動。某些應用程式允許使用者端經由一使用者介面與該硬體/軟體介面系統互動，例如一命令語言或一圖形使用者介面(GUI)。

一硬體/軟體介面系統傳統上為應用程式執行各種不同服務。在多個程式可同時執行之一多工硬體/軟體介面系統中，該硬體/軟體介面系統決定何應用程式應以何順序執行，且在輪流切換至另一應用程式之前應為各應用程式允許多少時間。該硬體/軟體介面系統一管理多個應用程式間內部記憶體之共享，並處理輸入與輸出至附加硬體裝置，例如硬碟、印表機、及撥號埠。該硬體/軟體介面系統亦傳送關於操作狀態與可能發生之任何錯誤之訊息至各應用程式(且在特定情況中至該使用者端)。該硬體/軟體介面系統亦可卸載批次工作之管理(如列印)，使得該初始應用程式可自此工作被釋放，且可恢復其他處理與/或操作。在可提供平行處理之電腦上，一硬體/軟體介面系統亦管理切割一程式，使得其一次執行於超過一處理器上。

一硬體/軟體介面系統殼(在此簡稱為一「殼」)係至一硬體/軟體介面系統之一互動式使用者端介面。(一殼亦可稱為一「指令轉譯器」，或在一作業系統中為一「作業系統殼」。)一殼係直接可由應用程式及/或使用者端存取之硬體/軟體介面系統之外部層。相對於一殼，一核心係直接與該硬體元件互動之一硬體/軟體介面系統之最內層。

雖然想像本發明之許多實施例特別適用於電腦型系

統，此文件中完全無欲限制本發明於此實施例中。相反的，在此使用的「電腦系統」一詞用於包圍任何與所有可儲存與處理資訊及/或可使用該已儲存資訊控制該裝置本身之行為或執行之裝置，不論是否此裝置為電子、機械、邏輯、或虛擬之本質。

B. 傳統檔案型儲存

在今日大部分的電腦系統中，「檔案」為可包含該硬體/軟體介面系統以及應用程式、資料組等等之可儲存資訊之單元。在所有現代硬體/軟體介面系統(Windows、Unix、Linux、Mac OS、虛擬機器系統等等)中，檔案為可由該硬體/軟體介面系統操控之之基本不連續的(可儲存與可取出)資訊(如資料、程式等等)單元。檔案群一般組織於「資料夾」中。在 Microsoft Windows、Macintosh OS、及其他硬體/軟體介面系統中，一資料夾係可取出、移動、及操控為單一資訊單元之一檔案集合。接著，這些資料夾組織於稱為一「目錄」(將在下方詳加描述)之一樹狀階層配置中。在特定其他硬體/軟體介面系統中，例如 DOS、z/OS 及大部分 Unix 型作業系統，「目錄」及/或「資料夾」一詞是可互換的，且早期的 Apple 電腦系統(如 Apple IIe)使用「類」一詞而非目錄；然而，如在此所使用者，這些詞本質是類似的且可互換，且用於另包含所有其他相等的詞，用於參照至階層資訊儲存結構與其資料夾與檔案元件。

傳統上，一目錄(也稱為資料夾之一目錄)係一樹狀階層結構，其中檔案群組化至資料夾中，接著根據包含該目

錄樹之相關節點位置加以排列。舉例來說，如第 2A 圖中舉例所示，一 DOS 型檔案系統基本資料夾(或「根目錄」)212 可包含複數資料夾 214，各可另包含額外資料夾(如同該特定資料夾之「子資料夾」)216，且各亦可包含無限個額外資料夾 218。各個資料夾可具有一或多個檔案 220，雖然於該硬體/軟體介面系統層，一資料夾中之個別檔案沒有共通性，除了其於樹狀階層中之位置之外。不令人驚訝的，此將檔案規劃至資料夾階層之方法間接反映用於儲存這些檔案之典型儲存媒體之實體組織(如硬碟、軟碟、CD-ROM 等等)。

除了前述之外，各資料夾係其子資料夾及其檔案之一容器—也就是說，各資料夾擁有其子資料夾與檔案。舉例來說，當一資料夾藉由該硬體/軟體介面系統刪除時，該資料夾之子資料夾與檔案亦被刪除(在各子資料夾之情況中，另遞迴包含其本身之子資料夾與檔案)。同樣的，各檔案一般由僅一資料夾擁有，雖然一檔案可被複製，且此副本位於一不同資料夾中，一檔案之一副本本身係一不同且不連續的之單元，沒有直接連結至原始者(如至原始檔之變更並不會於該硬體/軟體介面系統層鏡設至該複製檔中)。如此一來，檔案與資料夾因此特徵化於「實體」本質中，由於資料夾如同實體容器般被處理，且檔案如同這些容器中之單獨且個別之實體元件般被處理。

II. 用以組織、搜尋、及共享資料之 WINFS 儲存平台

本發明，結合在此之前討論併入參照之相關發明，導

向用以儲存、搜尋、及共享資料之一儲存平台。本發明之儲存平台將該資料平台延伸與擴張至既有檔案系統與上述資料庫系統種類之外，且設計為所有類型之資料之儲存，包含稱為項目之一新資料形式。

C. 詞彙表

如在此所使用及在申請專利範圍中者，下列詞彙具有下列意義：

- 一「項目」係可由一硬體/軟體介面系統存取之一可儲存資訊單元，其不像一簡單檔案，係具有跨所有由該硬體/軟體介面系統殼暴露至一使用者端之物件支援共用之一基本組屬性之一物件。項目亦具有跨所有項目類型共用支援之屬性與關係，包含允許引入之新屬性與關係之特性(且將在下方詳細討論)。
- 一「作業系統」(OS)係一特殊程式，做為應用程式與電腦硬體間之一媒介物。在大部分的情況中，一作業系統包含一殼與一核心。
- 一「硬體/軟體介面系統」係一軟體，或一硬體與軟體之組合，做為一電腦系統之基本硬體元件與執行於該電腦系統上之應用程式間之介面。一硬體/軟體介面系統一般包含(且在某些實施例中，僅由其組成)一作業系統。一硬體/軟體介面系統亦可包含一虛擬機器管理員(VMM)、一共用語言執行時間(CLR)或其功能相同者、一 Java 虛擬機器(JVM)或

其功能相同者、或其他取代或加入一電腦系統中作業系統之此軟體元件。一硬體/軟體介面系統之目的係提供一使用者可執行應用程式之一環境。任何硬體/軟體介面系統之目標係使該電腦系統方便使用，以及以一有效率之方法使用該電腦硬體。

D. 儲存平台簡介

請參閱第 3 圖，一儲存平台 300 包含實施於一資料庫引擎 314 上之一資料儲存 302。在一實施例中，該資料庫引擎包含具有物件相關擴充之一相關資料庫引擎。在一實施例中，該相關資料庫引擎 314 包含該 Microsoft SQL 伺服器相關資料庫引擎。該資料儲存 302 實施一資料模型 304，支援資料之組織、搜尋、共享、同步化、及安全性。特殊類型之資料描述於架構中，例如架構 340，且該儲存平台 300 提供用以配置該架構及用以擴充該架構之工具 346，如下方將更完整描述者。

實施於該資料儲存 302 中之一變更追蹤機制 306 提供追蹤變更至該資料儲存之能力。該資料儲存 302 亦提供安全性功能 308 與一升級/降及功能 310，兩者皆將於下方完整討論。該資料儲存 302 亦提供一組應用程式介面 312，以將該資料儲存 302 之功能揭示至利用該儲存平台之其他儲存平台元件與應用程式(如應用程式 350a、350b、及 350c)。本發明之儲存平台另仍包含一應用程式介面(API)322，其允許應用程式，例如應用程式 350a、350b、及 350c，存取該儲存平台之所有前述功能，並存取描述於

該架構中之資料。該儲存平台 API 322 可由應用程式其他 API 結合加以使用，例如該 OLE DB 324 與該 Microsoft Win32 API 326。

本發明之儲存平台 300 提供各種不同服務 328 至應用程式，包含促進使用者或系統間之資料共享之一同步化服務 330。舉例來說，該同步化服務 330 可允許與具有如同資料儲存 302 之相同格式之其他資料儲存 340，以及存取至具有其他格式之資料儲存 342 相互操作的能力。該儲存平台 300 亦提供檔案系統功能，允許該資料儲存 302 與既有檔案系統相互操作的能力，例如該 Windows NTFS 檔案系統 318。在至少某些實施例中，該儲存平台 320 亦可提供具有額外功能之應用程式，用以允許資料動作，且允許與其他系統互動。這些功能可以額外服務 328 之形式實施，例如一 Info Agent 服務 334 及一通知服務 332，以及以其他公用程式 336 之形式。

在至少某些實施例中，該儲存平台係實施於一電腦系統之硬體/軟體介面系統中，或形成其整合部分。舉例來說但非加以限制，本發明之儲存平台可實施於一作業系統、一虛擬機器管理員(VMM)、一共用語言執行時間(CLR)或其功能相同者、或一 Java 虛擬機器(JVM) 或其功能相同者，或形成其整合部分。透過其共用儲存基礎及架構化資料，本發明之儲存平台為消費者、有知識之工作者與專家允許更有效的應用程式研發。其提供一豐富且可擴充程式設計表面區域，不只有用於其資料模型中繼承之功能，亦

包含且擴充既有檔案系統與資料存取方法。

在下列描述中，且在各種不同圖式中，本發明之儲存平台300可稱為「WinFS」。然而，使用此名稱參照該儲存平台主要方便說明用，且並非用於做任何限制。

E. 資料模型

本發明之儲存平台300之資料儲存302實施一資料模型，支援位於該儲存中之資料之組織、搜尋、共享、同步化、及安全性。在本發明之資料模型中，一「項目」係儲存資訊之基礎單位。該資料模型提供用以宣告項目與項目擴充及用以建立項目間之關係及用以組織項目資料夾及類目中之項目之一機制，如下方將更完整說明者。

該資料模型依賴於兩個原始機制，類型與關係。類型為提供一格式之結構，其管理一類型例之形式。該形式表示為一已排序屬性組。一屬性係一值或一給定類型之值組之一名稱。舉例來說，一USPostalAddress可具有屬性街道、城市、郵遞區號、州，其中街道、城市與州為字串類型，而郵遞區號為Int32類型。街道可為允許該地址為該街到屬性具有超過一個值之多重值(即一組值)。該系統定義特定原始類型，可用於其他類型之架構中—這些包含字串、二進位、布林值、Int16、Int64、單、雙、位元組、日期時間、十進位與GUID。一類型之屬性可利用該原始類型或(具有下列所述之某種限制)任何建構類型加以定義。舉例來說，一位置類型可能定義具有屬性座標與地址，其中該地址屬性係為如上述USPostalAddress類型。屬性亦

可為必要或選擇性。

關係可被宣告且呈現兩個類型例組間之一映射。舉例來說，可能有一關係宣告於個人類型與位置類型之間，稱為 LiveAt，其定義何人住在何位置。該關係具有一名稱、兩個端點、命名為一來源端點與一目標端點。關係亦可具有一已排序屬性組。來源與目標端點具有一名稱與一類型。舉例來說，該 LiveAt 關係具有一來源稱為類型個人之居住者，及一目標稱為類型位置之住所，且除此之外具有屬性起始日期與結束日期，指示該居住者住在該住所之期間。注意一個人可在不同時間住在多個住所，且一住所可具有多個居住者，所以放置起始日期與結束日其資訊最可能的位置是在該關係本身上。

關係定義由給定為端點類型之類型所限制之範例間之一映射。舉例來說，該 LivesAt 關係無法為一汽車係該居住者之一關係，由於一汽車並不是一個人。

該資料模型允許類型間之一子類型-超級類型之定義。該子類型-超級類型關係亦稱為基礎類型關係，以如果類型 A 係類型 B 之基本類型之方式加以定義，其必須是每個 B 範例亦為 A 範例之情況。另一個表示方法是與 B 一致的每個範例亦必須與 A 一致。舉例來說，若 A 具有類型為字串之一名稱屬性，而 B 具有類型為 Int16 之一年齡屬性，其跟隨任何 B 範例必須具有一名稱與一年齡。該類型階層可想像為於根處具有單一超級類型之一樹。來自該根之分支提供該第一層子類型，於此層之分脂提供第二層子類

型，以下類推至其本身不具有任何子類型之最接近葉子類型。該樹並非限制為一統一深度，但無法包含任何循環。一給定類型具有零個或許多子類型，及零或一個超級類型。一給定範例可與具有該類型之超級類型之最多一類型一致化。另一方面，對於該樹中任何層之一給定範例來說，該範例可於該層與大部份子類型一致。如果該類型之範例亦必須為該類型之子類型之一範例，則一類型稱為抽象。

1. 項目

一項目係一可儲存資訊單元，不像一簡單檔案，其係具有一基本屬性組之一物件，其共同支援跨所有物件，其藉由該儲存平台暴露至一使用者端或應用程式。項目亦具有跨所有項目類型共用支援之屬性與關係，其包含允許新屬性與關係引入之特性，如下所討論者。

項目為共用操作之物件，例如拷貝、刪除、移動、開啟、列印、備分、恢復、複製等等。項目為可被儲存與取出之單元，且所有形式之可儲存資訊皆藉由存在為項目、項目屬性、項目間之關係之儲存平台加以操控，各將在下方詳加討論。

項目用於呈現真實世界與快速可了解資料單元，像是接觸、個人、服務、位置、文件(各種不同順序)等等。第5A圖係舉例說明一項目之結構之一方塊圖。該項目之不合格名稱係「位置」。該項目之合格名稱係「Core.Location」，其指出此項目結構定義為核心架構中之一特定項目類型。(該何新架構將在稍後詳加討論。)

該位置項目具有複數屬性，包含 Eaddresses、MetropolitanRrgion、Neighborhood、及 PostalAddresses。各者之特定類型屬性立即跟隨該屬性名稱加以指示，且以一冒號(「：」)與該屬性名稱隔開。至於該類型名稱之右側，為該使性類型允許之值數量指示於括弧(「[]」)之間，其中一星號(「*」)至該冒號(「：」)右側，指示一不指定及/或無限數量(「許多」)。一「1」至該冒號右側指示可有最多一值。一「0」至該冒號左側指示該屬性系選擇性(可能完全沒有值)。一「1」至該冒號左側指示必須至少有一值(該屬性係必要)。Neighborhood 與 MetropolitanRegion 皆為類型「nvachar」(或相等者)，其係一預定資料類型或「簡單類型」(且在此指示為缺乏大寫)。然而，EAddresses 與 PostalAddresses 分別係類型 Eaddress 與 PostalAffresses 之已定義類型或「複雜類型」之屬性(在此指示為大寫)。一複雜類型係自一或多個簡單資料類型及/或自其他複雜類型取出之類型。一項目之屬性之複雜類型亦由「巢狀元件」組成，由於該複雜類型之細節巢狀至該中間項目中，以定義其屬性，且關於這些複雜類型之資訊已具有這些屬性(在項目之邊界中，如稍後會加以討論者)之項目加以維護。這些類型化之概念為已知，且由熟知該項記憶人士可快速了解。

第 5B 圖係舉例說明複雜屬性類型 PostalAddress 與 EAddress 之一方塊圖。該 PostalAddress 屬性類型定義屬性類型 PostalAddress 之一項目可預期具有零或一個城市

值、零或一個國家代碼值、零或一個郵寄站值；及任何數量(零至許多)PostalAddress 類型依此類推等等。如此一來，在此定義一項目中一特定屬性之資料之形狀。該EAddress屬性類型類似定義，如圖所示。雖然在此選擇性使用此應用程式，另一個代表該位置項目中複雜類型之方法是以其中所列之各複雜類型之個別屬性繪製項目。第 5C 圖係舉例說明該位置項目之一方塊圖，其中其複雜類型將另加描述。然而，應了解第 5C 圖中位置項目之另一呈現方式係用於第 5A 圖中舉例說明之相同項目。本發明之儲存平台一允許子類型化，藉此一屬性類型可為另一屬性類型(其中該屬性類型繼承另一屬性類型之屬性、父屬性類型)之一子類型。

類似於屬性與其屬性類型但與其不同，項目繼承呈現其本身之項目類型，其亦可為子類型化之主題。換句話說，本發明之數個實施例中之儲存平台允許一項目做為另一項目之一子類型(藉此該項目繼承另一項目之屬性、父項目)。此外，對於本發明之各種不同實施例來說，每個項目係該「項目」項目類型之一子類型，其係該基礎架構中發現之第一與基礎項目類型。(該基礎架構亦會在稍後加以討論。)第 6A 圖舉例說明一項目，此範例中之位置項目，做為該基礎架構中找到之項目項目類型之一子類型。在此圖式中，該箭頭指示該位置項目(如同所有其他項目)係該項目項目類型之一子類型。該項目項目類型，如同所有其他項目自此取出之基礎項目，具有許多重要屬性，例如

ItemID 與各種不同時間戳，且藉此定義一作業系統中所有項目之標準屬性。在目前圖式中，這些項目項目類型之屬性由位置繼承，且藉此變成位置之屬性。

另一個呈現自該項目項目類型繼承之位置項目中之屬性之方法是以來自其中列示之父項目之各屬性類型之個別屬性繪製位置。第 6B 圖係舉例說明該位置項目之一方塊圖，其中描述之繼承類型附加於其立即屬性。應注意且了解此項目係第 5A 圖中舉例說明之相同項目，雖然在本圖式中，位置舉例說明具有其所有屬性，皆為立即—顯示於此圖與第 5A 圖中—且為繼承—顯示於此圖中但並非於第 5A 圖中(然而在第 5A 圖中，這些屬性藉由以一箭頭顯示加以參照，其中該位置項目係該項目項目類型之一子類型)。

項目係獨立物件；因此，若您刪除一項目，所有項目立即與繼承屬性也會被刪除。同樣的，當取出一項目時，接收的是該項目與所有其立即與繼承屬性(包含關於其複雜屬性類型之資訊)。本發明之特定實施利可允許於取出一特定項目時要求一子組屬性；然而，許多此實施例之預設值係提供當取出時具有其立即與繼承屬性之項目。此外，該項目之屬性亦可藉由新增新屬性至該項目之類型之既有屬性加以擴充。這些「擴充」接著為該項目之真正屬性，且該項目類型之子類型可自動包含該擴充屬性。

該項目之「邊界」係由其屬性(包含複雜屬性類型、擴充等等)加以呈現。一項目之邊界亦呈現執行於一項目上之一操作之限制，例如複製、刪除、移動、建立等等。舉例

來說，在本發明之數個實施例中，當一項目被複製時，該項目邊界中之所有亦會被複製。對於各項目來說，該邊界包含下列者：

- 該項目之項目類型及若該項目係另一項目之一子類型(如在本發明之數個實施例之此情況中，其中所有項目皆自該基本架構中之單一項目與項目類型取出)，任何可應用子類型資訊(也就是指關於該父項目類型之資訊)。若該原始項目被複製至另一項目之一子類型，該副本亦可為該相同項目之一子類型。
- 該項目之複雜類型屬性與擴充，若有的話。若該原始項目具有複雜類型之屬性(原生或擴充)，該副本亦可具有相同複雜類型。
- 在「擁有關係」上之項目之紀錄，也就是指何其他項目(該「目標項目」)由目前項目(該「擁有項目」)擁有之項目之擁有清單。此尤其關於項目資料夾，將在下方完整討論，且該規則說明於下方所有項目必須屬於至少一項目資料夾。此外，關於嵌入項目—將在下方完整討論—一嵌入項目視為該項目之部分，其中其為操作加以嵌入，例如複製、刪除等等。

2. 項目辨識

項目唯一辨識於具有一 ItemID 之全域(global)項目空間中。該 Base.Item 類型定義類型 GUID 之一欄位 ItemID，

其為該項目儲存該身分。一項目必須具有確實一身分於該資料儲存 302 中。

一項目參照係包含資訊以定位與辨識一項目之一資料結構。在該資料模型中，一抽象類型係定義命名為 ItemReference，所有項目參照類型自此取出。該 ItemReference 類型定義一虛擬方法，命名為 Resolve。該 Resolve 方法解決該 ItemReference，並傳回一項目。此方法係由 ItemReference 之具體子類型加以置換，其實施取出一項目給定參照之一功能。該 Resolve 方法係呼叫為該儲存平台 API 322 之部分。

ItemIDReference 係 ItemReference 之一子類型。其定義一 Locator 與一 ItemID 欄位。該 Locator 欄位命名(即辨識)一項目網域。期由一定位器解決方法加以處理，其可解決該 Locator 之值至一項目網域。該 ItemID 欄位係為類型 ItemID。

ItemPathReference 係一特殊化 ItemReference，第一一 Locator 與一 Path 欄位。該 Locator 欄位辨識一項目網域。期由一定位器解決方法加以處理，其可解決該 Locator 至一項目網域之值。該 Path 欄位包含該儲存平台名稱領域中之一(相關)路徑，根植於由該 Locator 所提供之項目網域。

此類型之參照無法用於一組操作中。該參照一般必須透過一路徑解決方法處理加以解決。該儲存平台 API 322 之 Resolve 方法提供此功能。

上述討論之參照形式透過第 11 圖中舉例說明之參照

類型階層加以呈現。繼承自這些類型之額外的參照類型可定義於架構中。其可用於一關係宣告中做為該目標欄位之類型。

3. 項目資料夾與類目

如下方將完整討論者，項目群組可組織至稱為項目資料夾之特定項目中(其不要與檔案資料夾搞混了)。然而，不像在大部分的檔案系統中，一項目可屬於超過一項目資料夾，使得當一項目存取於一項目資料夾並修改時，此修改之項目接著可直接從另一項目資料夾存取。基本上，雖然存取至一項目可能自不同項目資料夾發生，實際正存取者事實上正是相同的項目。然而，一項目資料夾不必須擁有其所有成員項目，或可僅與其他資料夾結合共同擁有項目，使得一項目資料夾之刪除不一定會導致該項目之刪除。然而，在本發明之數個實施例中，一項目必須屬於至少一項目資料夾，使得若一特定項目之主項目資料夾被刪除，則對某些實施例來說，該項目會自動被刪除，或在其他實施例中，該項目會自動變成一預設項目資料夾之一成員(如一「垃圾桶」項目資料夾在概念上類似於各種不同檔案-與-資料夾-型系統中名稱類似的資料夾)。

如下方將完整討論者，項目所屬之類目亦可根據共用描述特徵，例如(a)一項目類型(或類型)、(b)一特定立即獲繼承屬性(或屬性)、或(c)對應至一項目屬性之一特定值(或值)。舉例來說，為個人接觸資訊包含特定屬性之一項目可自動屬於一接觸類目，且具有接觸資訊屬性之任何項目同

樣會自動屬於此類目。同樣的，任何具有一位置屬性具有一「紐約市」值之項目可自動屬於一 NewYorkCity 類目。

類目在概念上係不同於項目資料夾，然而項目資料夾可包含沒有互相關聯性之項目(即沒有一共同描述特徵)、一類目中之各項目具有用以描述該類目之一共用類型、屬性、或值(一「通用性」)，且正是此通用性在該類目其他項目中形成與其他項目之關係的基礎。此外，一特殊資料夾中一項目之成員並非根據任何該項目之特殊方面加以強制，對於特定實施例來說，具有可分類為相關於一類目之一通用性之所有項目可於該硬體/軟體介面系統層自動變成該類目之一成員。概念上，類目亦可視為一虛擬項目資料夾，其成員係根據一特定查詢之結果而定(例如一資料庫中之內容)，且符合此查詢條件之項目(由該類目之通用性加以定義)因此會包含該類目之成員。

第 4 圖舉例說明項目、項目資料夾、及類目間之結構關係。複數項目 402、404、406、408、410、412、414、416、418、及 420 為各種不同項目資料夾 422、424、426、428、及 430 之成員。某些項目可屬於超過一個項目資料夾，如項目 402 屬於項目資料夾 422 與 424。某些項目，如項目 402、404、406、408、410、及 412，亦為一或多個類目 432、434、及 436 之成員，而其他時候，如項目 414、416、418、及 420，可不屬於任何類目(雖然這很可能是在特定實施例中，任何屬性自動暗示一類目中成員之所有權，且恩此一項目可能必須完全沒有特性，才不會在此一

實施例中為任何類目之一成員)。相對於資料夾之階層結構，類目與項目資料夾具有很像圖中導向圖的結構。在任何事件中，項目、項目資料夾、及類目都是項目(不同項目類型)。

相對於檔案、資料夾、及目錄，本發明之項目、項目資料夾、及類目並非特徵化於「實體」本質中，由於其沒有實體容器的對等概念，且因此項目可存在於超過一個此位置中。項目存在於超過一個項目資料夾位置中以及規劃至類目中之能力於該硬體/軟體介面層提供一遞增與充實程度之資料操控與儲存結構功能，超出目前業界中可用的之外。

4. 架構

a) 基礎架構

為了為項目之建立與使用提供一通用基礎，本發明之儲存平台之各種不同實施例包含有一基礎架構，歧見利用以建立與組織項目與屬性之一概念架構。該基礎架構定義特定特殊類型之項目與屬性，且這些特殊基礎類型之特性子類型可另自其取出。此基本架構之使用允許一程式設計人員在概念上分辨項目(及其各自類型)與屬性(及其各自類型)。此外，該基礎架構說明基礎屬性組，所有項目可處理為所有項目(及其對應項目類型)自該基本架構(及其對應項目類型)中之基礎項目取出。

如第7圖中舉例說明所示，且關於原本發明之數個實施例，該基本架構定義三個頂階類型：項目、擴充、及屬

性基礎。如圖所示，該項目類型係由此基礎「項目」項目類型之屬性所定義。相反的，該頂階屬性類型「PropertyBase」沒有預定屬性，且僅為支柱，所有其他屬性類型自此取出，且所有取出之屬性類型透過此互相關聯(自該單一屬性類型共同取出)。該擴充類型屬性定義該擴充擴充何項目，以及當一項目具有多個擴充時，辨識分辨擴充之間。

項目資料夾係該項目項目類型之一子類型，除了自該項目繼承之屬性之外，訂出用以建立連結至其成員(若有的話)之一關係之特性，然而 IdentityKey 與 Property 皆為 PropertyBase 之子類型。接著，CategoryRef 係 IdentityKey 之一子類型。

b) 核心架構

本發明之儲存平台之各種不同實施例另包含有一核心架構，其位頂階項目類型結構提供一概念架構。第 8A 圖係舉例說明該核心架構中之項目之一方塊圖，而第 8B 圖係舉例說明該核心架構中之屬性類型之一方塊圖。具有不同副檔名(*.com、*.exe、*.bat、*.sys 等等)之檔案間之差異與檔案-與-資料夾型系統中之此公式類似於該核心架構之功能。在該項目型硬體/軟體介面系統中，該核心架構定義一組核心項目類型，其直接(由項目類型)或間接(由項目子類型)特徵化所有項目至一或多個核心架構項目類型中，該項目型硬體/軟體介面系統了解它，且可直接以一預定且可預期之方式處理。該預定項目類型反映該項目型硬

體/軟體介面系統中之最常見項目，且因此一有效層由該項目型硬體/軟體介面系統了解這些包含該核心架構之預定項目類型加以獲得。

在特定實施例中，該核心架構係不可擴充—也就是說，沒有額外的項目類型可為直接來自該基本架構中項目類型之子類型，除了為自核心架構部分之特殊預定取出項目類型之外。藉由避免擴充至該核心架構(也就是藉由避免增加新項目至該核心架構)，由於每個後續項目類型必須是一核心架構項目類型之一子類型，該儲存平台要求使用該核心架構項目類型。此結構允許定義額外項目類型中彈性之一合理程度，且亦保留具有一預定組核心項目類型。

對本發明之各種不同實施例來說，且參照第 8A 圖，該核心架構支援之特定項目類型包含一或多個下列各者：

- 類目：此項目類型之項目(及自其取出之子類型)代表該項目型硬體/軟體介面系統中之有效類目。
- 商品：可辨識值之項目。
- 裝置：具有支援資訊處理功能之一邏輯結構之項目。
- 文件：具有不由項目型硬體/軟體介面系統轉譯，但由對應至該文件類型之一應用程式轉譯之內容之項目。
- 事件：記錄特定發生事件於該環境中之項目。
- 位置：代表實體位置(如地理位置)之項目。
- 訊息：兩個或多個原則(定義如下)間之通訊之項

目。

- 原則：具有一 ItemID 旁至少一明確可驗證身分之項目(如一個人、組織、群組、家庭、授權、服務等等之辨識)。
- 陳述：具有關於不包含限制、策略、訂閱、憑據等等之環境之特殊資訊之項目。

同樣的，且參照第 8B 圖，由該核心架構支援之特殊屬性類型可包含一或多個下列各者：

- 憑據(自該基本架構中之基礎 PropertyBase 類型取出)
- 原則身分金鑰(自該基本架構中之 IdentityKey 類型取出)
- 郵件地址(自該基本架構中之屬性類型取出)
- 豐富文字(自該基本架構中之屬性類型取出)
- EAddress(自該基本架構中之屬性類型取出)
- IdentitySecurityPackage(自該基本架構中之關係類型取出)
- RoleOccupancy(自該基本架構中之關係類型取出)
- BasicPresence(自該基本架構中之關係類型取出)

這些項目與屬性另參照第 8A 圖與第 8B 圖之其各自屬性加以描述。

5. 關係

關係係二進位關係，其中一項目指定為來源，而另一項目指定為目標。該來源項目與該目標項目係由該關係相

關連。該來源項目一般控制該關係之生命週期。也就是說，當該來源項目被刪除時，該項目間之關係亦被刪除。

關係分類為：包含與參照關係。該包含關係控制該目標項目之生命週期，而該參照關係不提供任何生命週期管理語義。第 12 圖舉例說明關係分類之方式。

該包含關係類型另分類至持有與內嵌關係。當至一項目之所有持有關係被移除時，該項目也會被刪除。一持有關係透過一參照計算機制控制該目標之生命週期。該內嵌關係允許合成項目之模型化，且可被視為互斥持有關係。一項目可為一或多個持有關係之目標，但一項目可為一個內嵌關係之目標。為一內嵌關係之一目標之一項目無法為任何持有或內嵌關係之一目標。

參照關係不控制該目標項目之生命週期。其可為搖擺—該目標項目可能不存在。參照關係可用於該全域項目名稱領域(即包含遠端資料儲存)中任何處模型化參照至項目。

擷取一項目不會自動擷取其關係。應用程式必須明確要求一項目之關係。除此之外，修改一關係不會修改該來源或目標項目；同樣的，增加一關係不會影響該來源/目標項目。

a) 關係宣告

該明確關係類型以下列元件定義：

- 一關係名稱係指定於該名稱屬性中。
- 關係類型為下列其中之一：持有、內嵌、參照。此

指定於該類型屬性中。

- 來源與目標端點。各端點指定該參照項目之一名稱與類型。
- 該來源端點欄位一般為類型 ItemID(不宣告)，且其必須參照相同資料儲存中之一項目做為該關係範例。
  - 對於持有與內嵌關係來說，該目標端點欄位必須為類型 ItemIDReference，且其必須參照相同資料儲存中之一項目做為該關係範例。對於參照關係來說，該目標端點可為任何 ItemReference 類型，且可參照其他儲存平台資料儲存中之項目。
  - 一純量或 PropertyBase 類型之選擇性一或多個欄位可被宣告。這些欄位可包含與該關係相關之資料。
  - 關係範例儲存於一通用關係表中。
  - 每個關係範例係由該組合(來源 ItemID、關係 ID)為一辨識。不論其類型為何，該關係 ID 為一給定項目中來源之所有關係於一給定來源 ItemID 中係唯一的。

該來源項目係該關係之擁有者。指定為擁有者之一項目控制該關係之生命週期，該關係本身係分離自其關聯之項目。該儲存平台 API 322 提供用以揭示與一項目相關之關係之機制。

在此為一關係宣告之一範例：

```
<Relationship Name="Employment" BaseType="Reference" >
```

```
<Source Name="Employee" ItemType="Contact.Person"/>
<Target Name="Employer" ItemType="Contact.Organization"
    ReferenceType="ItemIDReference" />
<Property Name="StartDate" Type="the storage
platformTypes.DateTime" />
<Property Name="EndDate" Type="the storage
platformTypes.DateTime" />
<Property Name="Office" Type="the storage
platformTypes.DateTime" />
</Relationship>
```

此為一參照關係之一範例。若由該來源參照所參照之個人項目不存在，該關係無法被建立。同樣的，若該個人項目被刪除，該個人與組織間之關係範例也會被刪除。然而，若該組織項目被刪除，該關係不會被刪除且其會搖擺。

b) 持有關係

持有關係用於模型化該目標項目之參照計算型生命週期管理。

一項目可為另或更多關係至項目之一來源端點。不為一內嵌項目之一項目可為一或多個持有關係中之一目標。

該目標端點參照類別必須為 ItemIDReference，且其必須參照該相同儲存中之一項目做為該關係範例。

持有關係加強該目標端點之生命週期管理。一持有關係範例之建立與搖擺之項目係一原子操作。額外之持有關係範例可被建立為將目標放置該相同項目。當具有一給定項目為目標端點之最後持有關係範例被刪除時，該目標項目亦被刪除。

當該關係之一範例建立時，於該關係宣告中指定之端點項目之類型一般會被加強。該端點項目之類型無法在該關係被建立之後建立。

持有關係做為行程該項目名稱領域中之一關鍵角色。

其包含定義與該來源項目相關之目標項目之名稱之「名稱」屬性。此相關名稱對來源自一給定項目之所有持有關係來說是唯一的。開始自該根項目至一給定項目之此相關名稱之排序清單形成至該項目之完整名稱。

該持有關係形成一導向非循環圖(DAG)。當一持有關係建立時，該系統確保一循環被建立，因此確保該項目名稱領域形成一 DAG。

該持有關係控制該目標項目之生命週期，然而其無法控制該目標端點項目之操作一致性。該目標項目操作獨立自透過一持有關係擁有其之項目。於為一持有關係之一來源之一項目上之複製、移動、備分及其他操作不會影響相同關係之一目標之項目—舉例來說，也就是說，備份一資料夾項目不會自動備份資料夾中所有項目(該資料夾成員關係之目標)。

下列係一持有關係之一範例：

```
<Relationship Name="FolderMembers" BaseType="Holding" >
    <Source Name="Folder" ItemType="Base.Folder"/>
    <Target Name="Item" ItemType="Base.Item"
        ReferenceType="ItemIDReference" />
</Relationship>
```

該資料夾成員關係允許一資料夾之概念做為項目之一通用集合。

c) 內嵌關係

內嵌關係模型化該目標項目之生命週期之互斥控制之概念。其允許合成項目之概念。

一內嵌關係範例之建立及目標化之項目係一原子操作。一項目可為零或更多內嵌關係之一來源。然而，一項

目可為一與僅一內嵌關係之一目標。係一內嵌關係之一目標之一項目無法為一持有關係之一目標。

該目標端點參照類型必須是 ItemIDReference，且其必須參照相同資料儲存中一項目做為該關係範例。

當該關係之一範例建立時，指定於該關係宣告中之端點項目之類型一般會被加強。在該關係被建立之後，該端點項目之類型無法變更。

內嵌關係控制該目標端點之操作一致性。舉例來說，一項目之序列化動作包含所有內嵌關係之序列化，其來源來自該項目以及所有其他目標，複製一項目亦複製所有其內嵌項目。

下例係一範例宣告：

```
<Relationship Name="ArchiveMembers" BaseType="Embedding" >
    <Source Name="Archive" ItemType="Zip.Archive" />
    <Target Name="Member" ItemType="Base.Item "
        ReferenceType="ItemIDReference" />
    <Property Name="ZipSize" Type="the storage platformTypes.bigint" />
    <Property Name="SizeReduction" Type="the storage platformTypes.float" />
</Relationship>
```

d) 參照關係

該參照關係不控制其參照之項目之生命週期。此外，該參照關係不保證該目標之存在，亦不保證該目標之類型如指定於該關係宣告中一般。此意指該參照關係可搖擺。同樣的，該參照關係可參照其他資料儲存中之項目。參照關係可視為與網頁中連結相似之概念。

參照關係宣告之一範例如下所示：

```
<Relationship Name="DocumentAuthor" BaseType="Reference" >
    <Sourc  ItemType="Document" ItemType="Base.Document"/>
    <Target ItemType="Author" ItemType="Base.Author"
        ReferenceType="ItemIDReference" />
    <Property Type="Role" Type="Core.CategoryRef" />
```

```
    <Property Type="DisplayName" Type="the storage
    platformTypes.nvarchar(256)" />
</Relationship>
```

任何參照類型允許於該目標端點中。參與於一參照關係中之項目可為任何項目類型。

參照關係用於模型化項目間最沒有生命週期之管理關係。由於該目標之存在並非強制，該參照關係便於模型化放鬆耦合關係。該參照關係可用於將目標放於其他資料儲存中之項目，包含其他電腦上之儲存。

e) 規則與限制

下列額外規則與限制應用於關係：

- 一項目必須是(一內嵌關係)或(一或多個持有關係)之一目標。一例外係該根項目。一項目可為零或多個參照關係之一目標。
- 為內嵌關係之一目標之一項目無法維持有關係之來源。其可為參照關係之一來源。
- 若自檔案升級，一項目無法維持有關係之一來源。其可為內嵌關係與參照關係之一來源。
- 可自一檔案升級之一項目無法為一內嵌關係之一目標。

f) 關係之排序

在至少一實施例中，本發明之儲存平台支援關係之排序。該排序係透過該基礎關係定義中一名稱為「Order」之屬性加以達成。該 Order 欄位上沒有唯一性限制。不保證具有相同「排序」屬性值之關係之排序，然而保證其可於具有較低「排序」值之關係之後與具有較高「排序」欄位

值之關係之前加以排序。

應用程式可藉由排序該組合(SourceItemID、RelationshipID、Order)以獲得預設順序中之關係。不論該集合中之關係之類型為何，源自一給定項目之所有關係範例排序為單一集合。然而，此保證一給定類型(如FolderMembers)之所有關係為一給定項目之關係及何之一排序子組。

用以操控關係之資料儲存 API 312 實施之緣關係排序之一組操作。引入下列項目以幫助解釋操作：

- *RelFirst* 係具有排序值 *OldFirst* 之排序集合中之第一關係。
- *RelLast* 係具有排序值 *OrdLast* 之排序集合中之最後一關係。
- *RelX* 係具有排序值 *OrdX* 之集合中之一給定關係。
- *RelPrev* 係具有小於 *OrdX* 之排序值 *OrdPrev* 之集合中至 *RelX* 之一最接近關係；及
- *RelNext* 係具有大於 *OrdX* 之排序值 OrdNext 之集合中至 RelX 之一最接近關係。

該操作包含但不限於：

- *InsertBeforeFirst(SourceItemID, Relationship)* 插入該關係做為該集合中之第一關係。該新關係之「Order」屬性之值可小於 OrdFirst。
- *InsertAfterLast(SourceItemID, Relationship)* 插

入該關係做為該集合中之最後一關係。該新關係之「Order」屬性之值可大於 OrdLast。

- *InsertAt(SourceItemID, ord, Relationship)* 為該「Order」屬性插入具有該指定值之一關係。
- *InsertBefore(SourveItemID, ord, Relationship)* 插入該關係在具有該給定順序值之關係之前。該新關係可被指派 OrdPrev 與 ord 間之「Order」值，前後不包含。
- *InsertAfter(SourceItemID, ord, Relationship)* 插入該關係在具有該給定順序值之關係之後。該新關係可被指派 ord 與 OrdNext 間之「Order」值，前後不包含。
- *MoveBefore(SourceItemID, ord, RelationshipID)* 移動具有給定關係 ID 之關係至具有給定「Order」值之關係之前。該關係可被指派 OrdPrev 與 ord 間之一新「Order」值，前後不包含。
- *MoveAfter(SourceItemID, ord, RelationshipID)* 移動具有給定關係 ID 之關係至具有給定「Order」值之關係之後。該關係可被指派 ord 與 OrdNext 間之一新「Order」值，前後不包含。

如前所述，每一項目必須為一項目資料夾之一成員。根據關係，每個項目必須具有含有一項目資料夾之一關係。在本發明之數個實施例中，特定關係由該項目間既有之關係所呈現。

如為本發明各種不同實施例所實施者，一關係提供由一項目(該來源)「擴充」致令一項目(該目標)之一導向二進位關係。一關係由該來源項目(擴充其之項目)所擁有，且因此若該來源被移除，該關係亦會被移除(如當該來源北刪除時，該關係也會被刪除)。此外，在特定範例中，一關係可共享(共同擁有)該目標項目之所有權，且此擁有權必須反映於該關係(如為該關係屬性類型顯示於第 7 圖中者)之 IsOwned 屬性(或其對等者)中。在這些實施例中，新 IsOwned 關係之建立會自動遞增該目標項目上之一參照計數，且刪除此一關係會遞減該目標項目上之參照計數。對於這些特定實施例來說，若具有大於零之一參照計數，項目繼續存在，且若當該計數達到零時會自動被刪除。同樣的，一項目資料夾係具有(或可具有)一組至其他項目之關係之一項目，這些其他項目包含該項目資料夾之成員。關係之其他實際實施方式可能且由本發明參與以達成在此所描述之功能。

不論實際實施方式為何，一關係係來自一物件至另一物件之一可選擇連結。一項目屬於超過一項目資料夾、以及屬於一或多個類別之能力，及是否這些項目、資料夾、及類別為公用或私用，係由給予存在(或其缺乏)於一項目型結構中之意義加以決定。這些邏輯關係係指派給一組關係之意義，不論實際實施方式為何，其為特定應用以達成在此所述之功能。邏輯連結係建立於該項目與其項目資料夾或類別(反之亦然)之間，基本上，由於項目資料夾與類

別各為一特定類型之項目。因此，項目資料夾與類別可與任何其他項目相同之方法運作—複製、增加至一電子郵件訊息、嵌入一文件中等等而無限制—且項目資料夾與類別可利用如同為其他項目般之相同機制被序列化與解序列化(匯入與匯出)。(舉例來說，在 XML 中，所有項目可具有一序列化格式，且此格式應用相等於項目資料夾、類別、及項目。)

代表一項目與其項目資料夾間之關係之前述關係可自該項目邏輯擴充至該項目資料夾、自該項目資料夾邏輯擴充至該項目、或兩者皆是。自一項目邏輯擴充至一項目資料夾之一關係代表該項目資料夾對該項目係公用，且共享具有該項目之成員資訊；相反的，缺乏自一項目至一項目資料夾之一邏輯關係代表該項目資料夾對該項目係私用，且不共享具有該項目之成員資訊。同樣的，自一項目資料夾邏輯擴充至一項目之一關係代表該項目對該項目係公用，且可共享該項目資料夾；然而，缺乏自該項目資料夾至該項目之一邏輯關係代表該項目係私用且不可共享。因此，當一項目資料夾匯入至另一系統時，其係共用於該新內容中之「公用」項目，且當一項目為其他可共享項目搜尋其項目資料夾時，其係提供具有關於其屬於之可用享項目之資訊之項目之「公用」項目資料夾。

第 9 圖係舉例說明一項目資料夾(同樣的係一項目本身)、其成員項目、及該項目資料夾與其成員項目間之互連關係之一方塊圖。該項目資料夾 900 具有複數項目 902、

904、及 906 之成員。項目資料夾 900 具有來自其本身至項目 902 之一關係 912，其代表該項目 902 係公用且可共享至項目資料夾 900、其成員 904 與 906、及任何可存取項目資料夾 900 之其他項目資料夾、類別、或項目(未顯示)。然而，沒有來自項目 902 至該項目資料夾 900 之關係代表項目資料夾 900 對項目 902 係私用，且不與項目 902 共其成員資訊。另一方面，項目 904 的確具有來自其本身至項目資料夾 900 之一關係 924，其代表該項目資料夾 900 係公用，且與項目 904 共用其成員資訊。然而，沒有來自項目資料夾 900 至項目 904 之關係代表項目 904 係私用且不可共享於項目資料夾 900、其其他成員 902 與 906、及可存取項目資料夾 900 之任何其他項目資料夾、類別、或項目(未顯示)。相對於其對項目 902 與 904 之關係(或其缺乏)，項目資料夾 900 具有來自其本身至該項目 906 之一關係 916，且項目 906 具有一關係 926 回到項目資料夾 900，其共同代表項目 906 係公用且可共享於項目資料夾 900、其成員 902 與 904、及任何其他可存取項目資料夾 900 之項目資料夾、類別、或項目(未顯示)，且項目資料夾 900 係公用且與項目 906 共享其成員資訊。

如前所討論者，一項目資料夾中之項目不需要共享一通用性，由於項目資料夾並非「加以描述」。另一方面，類別由共用於所有其成員項目之一通用性加以描述。因此，一類別之成員有限繼承至具有所述通用性之項目，且在特定實施例中，所有符合一類別之描述之項目會自動製作該

類別之成員。因此，項目資料夾允許欲由其成員所呈現之瑣碎類型架構，而類別根據該定義通用性允許成員。

當然，類別描述本質上為邏輯，且因此一類別可由類型、屬性、及/或值之任何邏輯呈現方式加以描述。舉例來說，一類別之一邏輯呈現方式可為其成員，以包含具有兩個屬性之一或兩者之項目。若這些為該類別之所述屬性為「A」與「B」，則該類別關係可包含具有屬性A但非B之項目、具有屬性B但非A之項目、及具有屬性A與B之項目。此屬性之邏輯呈現方式係由該邏輯運算子「OR」，其中由該類別描述之成員組係具有屬性A OR B之項目。同樣的，邏輯運算元(包含但不限於「AND」、「XOR」、及「NOT」之單獨或組合)亦可用於描述一類別，此應由熟知該項技藝人士所了解。

除了項目資料夾(未描述)與類別(已描述)間之差異之外，至項目之類別關係與至類別之項目關係基本上與在上方本發明之許多實施例中為項目資料夾與項目所揭示之相同方式。

第10圖係舉例說明一類別(同樣的係一項目本身)、其成員項目、及該類別與其成員項目間之互連關係之一方塊圖。該類別1000具有複數項目1002、1004、及1006之成員，其所有共享如由類別1000描述(共通性描述1008')之共同屬性、值、或類型1008之某些組合。類別1000具有來自其本身至項目1002之一關係1012，其代表該項目1002係公用且可共享至類別1000、其成員1004與1006、及任

何其他可存取類別 1000 之類別、項目資料夾、或項目(未顯示)。然而，沒有來自該項目 1002 至該類別 1000 之關係代表類別 1000 對項目 1002 係私用，且不與項目 1002 共享其成員資訊。另一方面，項目 1004 的確具有自其本身至類別 1000 之一關係 1024，代表該類別 1000 係公用且與項目 1004 共享其成員資訊。然而，沒有自類別 1000 擴充至該項目 1004 之關係代表項目 1004 係私用且不可共享於類別 1000、其其他成員 1002 與 1006、及任何其他可存取類別 1000 之類別、項目資料夾、或項目(未顯示)。相對於與項目 1002 與 1004 之關係(或其缺乏)，具有來自其本身至項目 1006 之一關係之類別 1000 具有回到類別 1000 之一關係 1026，同時代表項目 1006 係公用且可共用至類別 1000、其項目成員 1002 與 1004、及任何其他可存取類別 1000 之類別、項目資料夾、或項目(未顯示)，且該類別 1000 係公用且與項目 1006 共享其成員資訊。

最後，由於類別與項目資料夾本身為項目，且項目可互相具有關係，類別可對項目資料夾具有關係，且反之亦然，而類別、項目資料夾、及項目可於其他特定實施例中分別對其他類別、項目資料夾、及項目具有關係。然而，在各種不同實施例中，在該硬體/軟體介面系統層，項目資料夾結構及/或類別結構禁止包含循環。項目資料夾與類別結構類似於導向圖，禁止循環之實施例類似於導向非循環圖(DAG)，其藉由圖形理論業界中之數學定義，為導向圖，其中沒有路徑開始與結束於相同頂點處。

6. 可擴充性

該儲存平台用於提供有一初始架構組 340，如上所述。然而，除此之外，在至少某些實施例中，該儲存平台允許顧客，包含獨立軟體廠商(ISV)建立新架構 344(即新項目與巢狀元件類型)。此節點出用以藉由擴充該項目類型與定義於該初始架構組 340 中之巢狀元件類型(或簡稱「元件」類型)建立此架構之機制。

最好，該初始項目組與巢狀元件類型之擴充限制如下：

- 一 ISV 允許引入新項目類型，即 subtype Base.Item；
- 一 ISV 允許引入新巢狀元件類型，即 subtype Base.NestedElement；
- 一 ISV 允許引入新擴充，即 subtype Base.NestedElement；但，
- 一 ISV 無法子類型化任何由該初始儲存平台架構組 340 所定義之類型(項目、巢狀元件、或擴充類型)。

由於由該初始儲存平台架構組所定義之一項目類型或巢狀元件類型可能不真正符合一 ISV 應用程式之需求，其必須允許 ISV 自訂該類型。此以擴充之概念加以允許。擴充為強烈類型例，但(a)其無法獨立存在及(b)其必須附加於一項目或巢狀元件。

除了點出架構可擴充性之需求之外，擴充亦用於點出該「多類型化」問題。由於在某些實施例中，該儲存平台

無法支援多個繼承或重複子類型，應用程式可使用擴充做為模型化重複類型例之一方法(如文件係一合法文件以及一安全文件)。

a) 項目擴充

為了提供項目可擴充性，該資料模型另定義一抽象類型，命名為 Base.Extension。此為擴充類型階層之一根類型。應用程式可子類型化 Base.Extension 以建立特定擴充類型。

該 Base.Extension 類型係定義於該基本架構中，如下所示：

```
<Type Name="Base.Extension" IsAbstract="True">
    <Propety Name="ItemID"
        Type="the storage platformTypes.uniqueidentified"
        Nullable="false"
        MultiValucd="false"/>
    <Property Name="ExtensionID"
        Type="the storage platformTypes.uniqueidentified"
        Nullable="false"
        MultiValued="false"/>

</Type>
```

該 ItemID 欄位包含該擴充與其相關之項目之 ItemID。具有此 ItemID 之一項目必須存在。若具有該給訂 ItemID 之項目不存在，該擴充無法被建立。當該項目被刪除時，具有相同 ItemID 之所有擴充也會被刪除。該元組(ItemID,ExtensionID)唯一辨識擴充例。

一擴充類型之結構類似於一項目類型之結構：

- 擴充類型具有欄位；
- 欄位可為原始或巢狀元件類型；及
- 擴充類型可被子類型化。

下列限制為擴充類型加以應用：

- 擴充無法為關係之來源與目標；
- 擴充類型例無法獨立存在於一項目之外；及
- 擴充類型無法使用為該儲存平台類型定義中之欄位類型。

可與一給訂項目類型有關之擴充之類型上沒有限制。任何擴充類型允許以擴充任何項目類型。當多個擴充例附加於一項目時，其互相獨立於結構與行為中。

該擴充例個別自該項目儲存與存取。所有擴充類型例可自一全域擴充檢視加以存取。可產生一有效查詢以傳回一給定擴充類型之所有實體，不論其相關於何種項目類型。該儲存平台 API 提供可儲存、取出與修改項目上擴充之一可程式化模型。

該擴充類型可為利用該儲存平台單一繼承模型子類型化之類型。自一擴充類型取出建立一新擴充類型。一擴充之結構或行為無法取代或置換該項目類型階層之結構或行為。類似於項目類型，擴充類型例可直接透過與該擴充類型相關之檢視加以存取。該擴充之 ItemID 指出其屬於何項目，且可用於自該全域項目檢視取出該對應項目物件。該擴充為操作一致性之目的可視為該項目之部分。該儲存平台定義之複製/移動、備份/恢復與其他共用操作可操作於該擴充上，如同該項目之部分。

考慮下列範例。一接觸類型定義於該視窗類型組中。

```
<Type Name="Contact" BaseType="Base.Item" >
    <Property Name="Name"
            Type="String"
```

```
            Nullable="false"
            MultiValued="false"/>
        <Property Name="Address"
            Type="Address"
            Nullable="true"
            MultiValued="false"/>

    </Type>
```

一 CRM 應用程式研發人員會想要附加一 CRM 應用程式擴充至儲存於該儲存平臺中之接觸上。該應用程式研發人員會定義一 CRM 擴充，其包含該應用程式可操控之額外資料結構：

```
    <Type Name="CRMExtension" BaseType="Base.Extension" >
        <Property Name="CustomerID"
            Type="String"
            Nullable="false"
            MultiValued="false"/>

    </Type>
```

一 HR 研發人員可能亦想要以該接觸附加額外資料。此資料獨立於該 CRM 應用程式資料之外。同樣的，該應用程式研發人員可建立一擴充：

```
    <Type Name="HRExtension" EBaseType="Base.Extension" >
        <Property Name="EmployeeID"
            Type="String"
            Nullable="false"
            MultiValued="false"/>

    </Type>
```

CRMExtension 與 HRExtension 為可附加於接觸項目之兩個獨立擴充。其互相獨立建立與存取。

在上述範例中，該 CRMExtension 類型之欄位與方法無法取代該接觸階層之欄位或方法。應注意該 CRMExtension 類型之範例可附加於接觸之外之其他項目類型。

當該接觸項目被取出時，其項目擴充不會自動被取

出。給定一接觸項目，其相關項目擴充可藉由為具有相同ItemId之擴充查詢該全域擴充檢視加以存取。

該系統中所有CRMExtension擴充可透過該CRMExtension類別檢視加以存取，不論其屬於何項目。一項目之所有項目擴充共用相同項目id。在上述範例中，該接觸項目例與該附加之CRMExtension與HRExtension例具有相同之ItemID。

下表歸納項目、擴充與巢狀元件類型間之相似性與差異：

項目vs項目擴充vs巢狀元件

	項目	項目擴充	巢狀元件
項目ID	具有其本身之項目id	共享該項目之項目id	不具有其本身之項目id。巢狀元間係該項目之部分
儲存	項目階層儲存於其本身表中	項目擴充階層儲存於其本身表中	以項目儲存
查詢/搜尋	可查詢項目表	可查詢項目擴充表	一般僅可於包含項目內容中查詢
查詢/搜尋範圍	可跨一項目類型之所有	可跨一項目擴充類型之	一般僅可在單一(包含)

	範例搜尋	所有範例搜尋	項目之巢狀元件類型例中搜尋
關係語義	可具有至項目之關係	沒有至項目擴充之關係	沒有至巢狀元件之關係
至項目之關聯	可經由持有、內嵌與軟關係關聯於其他項目	一般僅可經由擴充加以關聯。該擴充語義類似於內嵌項目語義	經由欄位關聯至項目。巢狀元件為該項目之部分

b) 擴充巢狀元件類型

巢狀元件類型並非以與該項目類型之相同機制加以擴充。巢狀元件之擴充以與巢狀元件類型之欄位相同之機制加以儲存與存取。

該資料模型為命名為 Element 之巢狀元件類型定義一根：

```
<Type Name="Element"
      IsAbstract="True">
   <Property Name="ElementID"
           Type="the storage platformTypes.uniqueidentifier"
           Nullable="false"
           MultiValued="false"/>
</Type>
```

該巢狀元件類型自此類型加以繼承。該巢狀元件元件類型額外定義為一多元件組之一欄位：

```
<Type Name="NestedElement" BaseType="Base.Element"
    IsAbstract="True">
    <Property Name="Extensions"
           Type="Base.Element"
           Nullable="false"
```

```
                MultiValued="true"/>
</Type>
```

該巢狀元件擴充於下列方式不同於項目擴充：

- 巢狀元件擴充並非擴充類型。其不屬於根植於該Base.Extension類型之擴充類型階層。
- 巢狀元件擴充儲存與該項目之其他欄位，且不可全域存取—無法做出取出一給定擴充類型之所有範例之一查詢。
- 這些擴充與(該項目之)其他巢狀元件儲存之相同方式加以儲存。如同其他巢狀組，該巢狀元件擴充儲存於一 UDT 中。其可透過該巢狀元件類型之擴充欄位加以存取。
- 用於存取多值屬性之集合介面易用於存取與循環類型擴充組。

下列表歸納與比較項目擴充與巢狀元件擴充。

項目擴充 vs 巢狀元件擴充

	項目擴充	巢狀元件擴充
儲存	項目擴充階層儲存於其本身表中	儲存如同巢狀元件
查詢/搜尋	可查詢項目擴充表	一般僅可於該包含項目內容中查詢
查詢/搜尋範圍	可跨一項目擴充類型之所有範例加以搜尋	一般僅可於單一(包含)項目之巢狀元件類型例中

		加以搜尋
可程式化能力	需要特殊擴充 API 與擴充表上之特殊查詢	巢狀元件擴充如同巢狀元件之任何其他多值欄位；使用一般巢狀元件類型 API
行為	可關聯行為	不允許行為(？)
關係語義	沒有至項目擴充之關係	沒有至巢狀元件擴充之關係
項目 ID	共用該項目之項目 id	沒有其本身之項目 id。巢狀元件擴充係該項目之部分

F. 資料庫引擎

如上所述，該資料儲存係實施於一資料庫引擎上。在本實施例中，該資料庫引擎包含實施該 SQL 查詢語言之一相關資料庫引擎，例如該 Microsoft SQL 伺服器引擎，具有物件相關擴充。此節描述根據本發明，該資料儲存實施之資料模型映射至該相關儲存，且提供由儲存平台客戶消耗之邏輯 API 上之資訊。然而，應了解當應用一不同資料庫引擎時，可應用一不同之映射。事實上，除了於一相關資料庫引擎上實施該儲存平台概念資料模型之外，其亦可實施於其他類型之資料庫上，如物件導向與 XML 資料庫。

一物件導向(OO)資料庫系統提供用於程式設計語言

(如 C++、Java)之持久性與處理。一「項目」之儲存平台概念相當映射至物件導向系統中之一「物件」，雖然內嵌集合必須加入至物件中。其他儲存平台類型概念，像是既成與巢狀元件類型，亦映射物件導向類型系統。物件導向系統一般已經支援物件身分；因此，項目身分可映射至物件身分。該項目行為(操作)相當映射至物件方法。然而，物件導向系統一般缺乏組織功能，且很難搜尋。同樣的，物件導向系統不支援非結構化與半結構化資料。為了支援在此描述之完整儲存平台資料模型，如同關係、資料夾、及擴充之概念必須加入該物件資料模型中。除此之外，如同升級、同步化、通知、及安全性之機制必須加以實施。

類似於物件導向系統，根據 XSD(XML 架構定義)，XML 支援單一繼承型類型系統。本發明之項目類型系統可映射至該 XSD 類型模型。XSD 亦不提供行為之支援。項目之 XSD 必須以項目行為參數化。XML 文件處理單一 XSD 文件，且缺乏組織與廣泛搜尋功能。如同以物件導向資料庫，為了支援在此描述之資料模型，其他如同關係及資料夾之概念必須併入此 XML 資料庫中；同時，如同同步化、通知與安全性之機制必須加以實施。

關於下列子節，提供一些舉例說明以促進揭示之一般資訊。第 13 圖係舉例說明一通知機制之一示意圖。第 14 圖係舉例說明一範例之一示意圖，其中二處理皆插入一新紀錄至相同 B 樹中。第 15 圖舉例說明一資料變化偵測處理。第 16 圖舉例說明一範例目錄樹。第 17 圖顯示一範例，

其中一目錄型檔案系統之一既有資料夾移入該儲存平台資料儲存中。

1. 利用 UDT 之資料儲存實施方式

在本實施例中，於包含該 Microsoft SQL 伺服器引擎之一實施例中之相關資料庫引擎 314 支援內建純量類型。內建純量類型係「自然」及「簡單」的。其自然於該使用者無法定義其本身類型，且其簡單於其無法封裝一複雜結構。使用者定義類型(此後稱：UDTs)為上述類型可擴充性提供一機制，且藉由允許使用者藉由定義複雜結構類型擴充該類型系統超出該自然純量類型之外。一旦由一使用者加以定義，一 UDT 可用於可能使用一內建純量類型之類型系統中任何地方。

根據本發明之一方面，該儲存平台架構映射至該資料庫引擎儲存中之 UDT 類別。資料儲存項目映射至自該 Base.Item 類型取出之 UDT 類別。如同項目，擴充亦映射至 UDT 類別且使用繼承。該根擴充類型係 Base.Extension，所有擴充類型自此取出。

一 UDT 係一 CLR 類別—其具有狀態(即資料欄位)與行為(即程式)。UDT 利用任何管理語言加以定義—C#、VB、.NET 等等。UDT 方法與操作子可針對該類型之一範例呼叫於 T-SQL 中。一 UDT 可為：一行中之一列之類型、T-SQL 中一程式之一參數類型、或 T-SQL 中一變數之類型。

儲存平台架構至 UDT 類別之映射於一高階相當直覺。一般來說，一儲存平台架構映射至一 CLR 名稱領域。

一儲存平台類型映射至一 CLR 類別。該 CLR 類別繼承鏡射至該儲存平台類別繼承，而一儲存平台屬性映射至一 CLR 類別屬性。

2. 項目映射

為提供欲全域搜尋之項目的需求，且提供本實施例之相關資料庫的支援，以供繼承與類型取代之用，該資料庫儲存中項目儲存的一可能實施方式可儲存所有項目於具有一列類型 Base.Item 之單一表中。利用類型可取代性，所有類型之項目皆可加以儲存，且搜尋可利用 Yukon 的「is of(Type)」操作子由項目類型與子類型加以過濾。

然而，由於考量與此一方法相關之負載，在本實施例中，該項目由頂階類型加以切割，使得各類型「家族」之項目儲存於一分離表中。在此切割架構之下，一表為各項目類型直接自 Base.Item 繼承加以建立。這些下繼承之類型利用類型可取代性儲存於適當類型家族表中，如上所述。僅繼承自 Base.Item 之第一層被特殊對待。

一「陰影(shadow)」表用於為所有項目儲存全域可搜尋屬性之副本。此表可由該儲存平台 API 之 Update()方法加以維護，所有資料變更透過此作成。不向該類型家族表，此全域項目僅包含該項目之頂階純量屬性，並非完整 UDT 項目物件。該全域項目表藉由揭示一 ItemID 與一 TypeID 允許瀏覽至儲存於一類型家族表中之項目物件。該 ItemID 一般會唯一辨識該資料儲存中之項目。該 TypeID 可利用元資料加以映射至一類型名稱與包含該項目之檢視，在此

不加以描述。由於由其 ItemID 尋找一項目可為一共用操作，兩者皆於該全域項目表之內容中，且否則，提供一 GetItem()函數以取出給予一項目之 ItemID 之一項目物件。

為了方便存取與隱藏實施方式細節至該可能之內容，項目之所有查詢可能針對建立於上述項目表上之檢視。尤其是，檢視可為各項目類型針對該適當類型家族表加以建立。這些類型檢視可選擇相關類型之所有項目，包含子類型。為了方便，除了該 UDT 物件之外，該檢視可為該類型之所有頂階欄位揭示列，包含繼承欄位。

3. 擴充映射

擴充類似於項目，也具有某些相同要求。如同其他根類型支援繼承，擴充傾向於許多相同的考量與斟酌於儲存中。基於此，一類似類型家族映射應用至擴充，而非單一表方法。當然，在其他實施例中，可使用單一表方法。在本實施例中，一擴充由 ItemID 與一項目相關，且包含於該項目內容中為唯一之一 ExtensionID。如同具有項目，可能提供一功能以取出給定其身分之一擴充，其由一 ItemID 與 ExtensionID 對所組成。一檢視為各擴充類型加以建立，類似於該項目類型檢視。

4. 巢狀元件映射

巢狀元件為可嵌入項目、擴充、關係、或其他巢狀元件中，以形成深巢狀結構之類型。如同項目與擴充，巢狀元件實施如同 UDT，但其儲存於一項目與擴充之中。因此，巢狀元件沒有儲存映射超出其項目與擴充容器之外。

換句話說，沒有表於直接儲存巢狀元件類型之範例之系統中，且沒有檢視特別專用於巢狀元件。

5. 物件身分

該資料模型中之各身分，即各項目、擴充與關係，具有唯一金鑰值。一項目由其 ItemId 唯一辨識。一擴充係由(ItemId,ExtensionId)之一合成金鑰唯一辨識。一關係係由(ItemId,RelationshipId)一合成金鑰唯一辨識。ItemId、ExtensionId 與 RelationshipId 為 GUID 值。

6. SQL 物件命名

該資料儲存中建立之所有物件可儲存於由該儲存平台架構名稱取出之一 SQL 架構名稱中。舉例來說，該儲存平台基本架構(通常稱為「基礎」)可產生「[System.Storage]」SQL 架構中之類型，例如「[System.Storage].Item」。產生之名稱由一限定子加入字首，以減少命名衝突。在適當處，使用一驚嘆號(!)做為該名稱各邏輯部分之一分隔器。下表繪出用於該資料儲存中物件之命名規則之輪廓。列出各架構元件(項目、擴充、關係與檢視)與用於存取該資料儲存中範例之裝飾命名規則。

物件	名稱裝飾	說明	範例
主要項目搜尋檢視	**Master!Item**	提供一項目簡介於該目前項目網域	[System.Storage].[**Master!Item**]

		中。	
類型化項目搜尋檢視	*ItemType*	提供來自*項目*之所有屬性資料與任何父類型。	[AcmeCorp.Doc].[OfficeDoc]
主要擴充搜尋檢視	**Master!Extension**	提供該項目網域中所有擴充之一簡介。	[System.Storage].[**Master!Extension**]
類型化擴充搜尋檢視	**Extension!***extensionType*	為*擴充*提供所有屬性資料。	[AcmeCorp.Doc].[**Extension!**StickyNote]
主要關係檢視	**Master!Relationship**	提供該項目網域中所有	[System.Storage].[**Master!Relationship**]

		關係之一簡介。	
關係檢視	**Relationship!***relationshipName*	提供與一給定*關係*相關之所有資料	[AcmeCorp.Doc].[**Relationship!**AuthorsFormDocument]
檢視	**View!***viewName*	根據該架構*檢視*定義提供該列/類型。	[AcmeCorp.Doc].[**View!**DocumentTitles]

7. 列命名

當映射任何物件模型至一儲存中時，由於與一應用程式物件儲存之惡外資訊，會發生撞名之可能性。為了避免為了避免撞名，所有非類型特定列(不直接映射至一類型宣告中之一命名為 Property 之列)以一底線(_)字元加於字首。在本實施例中，不允許底線(_)字元做為任何識別符屬性之開頭字元。此外，為了統一 CLR 與該資料儲存間之命名，一儲存平台類型或架構元件(關係等等)之所有屬性應具有一大寫第一字元。

8. 搜尋檢視

由用於搜尋已儲存內容之儲存平台提供檢視。為各項目與擴充類型提供一 SQL 檢視。此外，提供檢視以支援關係與檢視(如同由該資料模型定義者)。該儲存平台中之所有 SQL 檢視與基本表係唯讀。資料可利用該儲存平台 API 之 Update()方法加以儲存或變更，如下方將完整說明者。

明確定義於一儲存平台架構(由該架構設計人員加以定義，且不自動由該儲存平台加以產生)中之各檢視可由命名為 SQL 檢視[<schema-name>].[**View!**<view-name>]加以存取。舉例來說，該架構「AcmePublisher.Books」中命名為「BookSales」之一檢視可利用該名稱「[AcmePublisher.Books].[View!BookSales]」加以存取。由於一檢視之輸出格式自訂於一各檢視基礎上(由定義該檢視者提供之一任意查詢加以定義)，該列直接根據該架構檢視定義加以映射。

該儲存平台資料儲存中之所有 SQL 搜尋檢視為列使用下列排序規則：

- 檢視結果之邏輯「關鍵」列，例如 *ItemId*、*ElementId*、*RelationshipId*、...
- 結果之類型上之元資料資訊，例如 *TypeId*。
- 變更追蹤列，例如 *CreateVersion*、*UpdateVersion*、...
- 類型特定列(該宣告類型之屬性)
- 類型特定檢視(家族檢視)亦包含傳回該物件之一物件列

各類型家族之成員可利用一連串項目檢視加以搜尋，其為該資料儲存中各項目類型之一檢視。第 28 圖係舉例說明一項目搜尋檢視之概念之一示意圖。

a) 項目

各項目搜尋檢視包含各範例之一行與該特定類型或其子類型之一項目。舉例來說，文件之檢視可傳回 Document、LegalDocument 與 ReviewDocument 之範例。給予此範例，該項目檢視可被概念化，如第 29 圖中所示。

(1) 主項目搜尋檢視

一儲存平台資料儲存之各範例定義一特殊項目檢視，稱為該*主項目檢視*。此檢視提供該資料儲存中各項目上之簡介資訊。該檢視為每項目類型屬性提供一列，一列描述該項目類型，而數列用於提供變更追蹤與同步化資訊。該主項目檢視利用該名稱「[System.Storage].[Master!Item]」辨識於一資料儲存中。

列	類型	說明
ItemId	ItemId	該項目之儲存平台身分
_TypeId	TypeId	該項目之 TypeId — 辨識該項目之實際類型，並可用於利用一元資料類別取出該類型上之資訊。

_RootItemId	ItemId	第一非嵌入祖先之 ItemId 控制此項目之生命週期。
<global change tracking>	...	全域變更追蹤資訊
<Item props>	n/a	各項目類型屬性之一列

(2) 類型化項目搜尋檢視

各項目類型亦具有一搜尋檢視。類似於該根項目檢視，此檢視亦提供經由該「_Item」列存取至該項目物件。各類型化項目搜尋檢視利用該名稱[schemaName].[itemTypeName]於一資料儲存中加以辨識。舉例來說，[AcmeCorp.Doc].[OfficeDoc]。

列	類型	說明
ItemId	ItemId	該項目之儲存平台身分
<type change tracking>	...	類型變更追蹤資訊
<parent props>	<property specific>	每個父屬性之一列
<item props>	<property specific>	此類型每個互斥屬性之一列
_Item	屬性之 CLR 類型	CLR 類型—已宣

		告項目之類型

b) 項目擴充

一WinFS儲存中所有項目擴充亦可利用搜尋檢視加以存取。

(1) 主擴充搜尋檢視

一資料儲存之各範例定義一特殊擴充檢視，稱為主擴充檢視。此檢視提供該資料儲存中各擴充上之簡介資訊。該檢視具有各擴充屬性之一列，一列描述該擴充之類型，而數列用於提供變更追蹤與同步化資訊。該主擴充檢視係利用該名稱「[System.Storage].[Master!Extension]」於一資料儲存中加以辨識。

列	類型	說明
ItemId	ItemId	此擴充相關聯之項目之儲存平台身分
ExtensionId	ExtensionId (GUID)	此擴充例之 Id
_TypeId	TypeId	該擴充之 TypeId — 辨識該擴充之實際類型，且可用於利用該元資料類別取出該擴充上之資訊。
<global change	…	全域變更追蹤資

tracking>		訊
<ext properties>	<property specific>	各擴充類型屬性之一列

(2) 類型化擴充搜尋檢視

各擴充類型亦具有一搜尋檢視。類似於主擴充檢視，此檢視議題公雞由該_Extension 列存取至項目物件。各類型化擴充搜尋檢視利用該名稱[schemaName].[Extension!extensionTypeName]辨識於一資料儲存中。舉例來說，[AcmeCorp.Doc].[Extension!OfficeDocExt]。

列	類型	說明
ItemId	ItemId	此擴充與其相關之項目之儲存平台身分
ExtensionId	ExtensionId (GUID)	此擴充例之 Id
<type change tracking>	…	類別變更追蹤資訊
<parent props>	<property specific>	各父屬性之一列
<ext props>	<property specific>	此類別之各互斥屬性之一列
_Extension	擴充例之 CLR 類別	CLR 物件—已宣告擴充之類型

c) 巢狀元件

所有巢撞元件儲存於項目、擴充、或關係範例中。如同以往，其藉由查詢該適當項目、擴充、或關係搜尋檢視加以存取。

d) 關係

如上所述，關係形成一儲存平台資料儲存中項目間之連結之基礎單元。

(1) 主關係搜尋檢視

各資料儲存提供一主關係檢視。此檢視提供該資料儲存中所有關係範例上之資訊。該主關係檢視利用該名稱「[System.Storage].[Master!Relationship]」辨識於一資料儲存中。

列	類型	說明
ItemId	ItemId	來源端點之身分(ItemId)
RelationshipId	RelationshipId (GUID)	關係範例之id
_RelTypeId	RelationshipTypeId	該關係之RelTypeId—利用該元資料類別辨識該關係範例之類型。
<global change	…	全域變更追蹤

tracking>		資訊。
TargetItemReference	ItemReference	目標端點之身分
_Relationship	Relationship	為此範例之關係物件之範例

(2) 關係範例搜尋檢視

各已宣告關係亦具有一搜尋檢視，其傳回該特定關係之所有範例。類似於該主關係檢視，此檢視亦為該關係資料之各屬性提供已命名列。各關係範例搜尋檢視利用該名稱 [schemaName].[Relationship!relationshipName] 於一一資料儲存中加以辨識。舉例來說，[AcmeCorp.Doc].[Relationship!DocumentAuthor]。

列	類型	說明
ItemId	ItemId	來源端點之身分(ItemId)
RelationshipId	RelationshipId (GUID)	該關係範例之 id
<type change tracking>	...	類型變更追蹤資訊
TargetItemReference	ItemReference	目標端點之身分
<source name>	ItemId	來源端點身分之命名屬性(別名為 ItemId)
<target name>	ItemReference 或已取出類別	目標端點身分之命名屬性(別名為 TargetItemReference)
<rel property>	<property specific>	該關係定義之各屬性之一列
_Relationship	關係範例之 CLR 類型	CLR 物件—宣告關係之類型

e)

9. 更新

該儲存平台資料儲存中所有檢視為唯讀。為了建立一資料模型元件之一新範例(項目、擴充或關係)，或為了更新一既有範例，必須使用該儲存平台 API 之 ProcessOperation 或 ProcessUpdategram 方法。該 ProcessOperation 方法係由該資料儲存定義之單一儲存程序，其消耗詳述域執行之一動作之一「操作」。該 ProcessUpdategram 方法係採取一組已排序操作之一已儲存程序，亦稱為一「更新程式」，其集合詳述一組欲執行之動作。

該操作格式係可擴充，且跨該架構元件提供各種不同操作。某些共用操作包含：

1. 項目操作：

a. CreateItem(建立一內嵌或持有關係之內容中之一新項目)

b. UpdateItem(更新一既有項目)

2. 關係操作：

a. CreateRelationship(建立一參照或持有關係之一範例)

b. UpdateRelationship(更新一關係範例)

c. DeleteRelationship(移除一關係範例)

3. 擴充操作：

a. CreateExtension(加入一擴充至一既有項目)

b. UpdateExtension(更新一既有擴充)

c. DeleteExtension(刪除一擴充)

10. 變更追蹤&墓碑

變更追蹤與墓碑服務由該資料儲存加以提供，將在下方詳加討論。本節提供揭示於一資料儲存中之變更追蹤資訊之一輪廓。

a) 變更追蹤

由該資料儲存提供之各搜尋檢視包含用於提供變更追蹤資訊之列；該列跨所有項目、擴充及關係檢視共用。由架構設計人員明確定義之儲存平台架構檢視不會自動提供變更追蹤資訊—透過該檢視本身建立於其上之搜尋檢視間接提供此資訊。

對於該資料儲存中各元件來說，變更追蹤資訊可用於兩處—該「主」元件檢視與該「類型化」元件檢視。舉例來說，該 AcmeCorp.Document.Document 項目類型上之變更追蹤資訊可用於該主項目檢視「[System.Storage].[Master!Item]」及類型化項目搜尋檢視[AcmeCorp.Document].[Document]。

(1)「主」搜尋檢視中之變更追蹤

該主搜尋檢視中之變更追蹤資訊提供一元件之建立與更新版本上之資訊，同步夥伴建立該元件於資訊上、同步夥伴最後更新元件之資訊及來自建立與更新之各夥伴之版本號碼。同步關係(下方描述)中夥伴由夥伴金鑰加以辨識。類型[System.Storage.Store].ChangeTrackingInfo 之命名為_ChangeTrackingInfo 之單一 UDT 物件包含所有此資訊。該類型定義於該 System.Storage 架構中。

_ChangeTrackInfo 可用於項目、擴充與關聯之所有全域搜尋檢視中。ChangeTrackingInfor 之類型定義為：

```
<Type Name="ChangeTrackingInfo" BaseType="Base.NestedElement">
    <FieldProperty Name="CreationLocalTS"        Type="SqlTypes.SqlInt64"
    Nullable="False" />
    <FieldProperty Name="CreatingPartnerKey"
    Type="SqlTypes.SqlInt32"        Nullable="False" />
    <FieldProperty Name="CreatingPartnerTS"
    Type="SqlTypes.SqlInt64"        Nullable="False" />
    <FieldProperty Name="LastUpdateLocalTS"
    Type="SqlTypes.SqlInt64"        Nullable="False" />
    <FieldProperty Name="LastUpdatingPartnerKey
    Type="SqlTypes.SqlInt32"        Nullable="False" />
    <FieldProperty Name="LastUpdatingPartnerTS"      Type="SqlTypes.SqlInt64"
    Nullable="False" />
</Type>
```

這些屬性包含下列資訊：

列	說明
_CreationLocalTS	該區域機器之建立時間戳
_CreatingPartnerKey	建立此時體之夥伴之PartnerKey。若該實體為區域建立，此為區域機器之PartnerKey。
_CreatingPartnerTS	此實體建立於對應至_CreatingPartnerKey 之夥伴處之時間之時間戳。
_LastUpdateLocalTS	對應至該區域機器之更新時間之區域時間戳
_LastUpdatingPartnerKey	最後更新此時體之夥伴之PartnerKey。若該最後更新

	至該實體於區域完成，此為該區域機器之 PartnerKey。
_LastUpdatingPartnerTS	此實體更新於對應至 _LastUpdatingPartnerKey 之夥伴處之時間之時間戳。

(2)「類型化」搜尋檢視中之變更追蹤

除了提供相同資訊做為全域搜尋檢視之外，各類型化搜尋檢視提供記錄該同步拓樸中各元件之同步狀態之額外資訊。

列	類型	說明
<global change tracking>	...	來自全域變更追蹤之資訊
_ChangeUnitVersions	MultiSet<ChangeUnitVersion>	該特定元件中變更單元之版本號碼之說明
_ElementSyncMetadata	ElementSyncMetadata	關於僅對該同步執行時間有興趣之此項目之額外版本獨立元資料。
_VersionSyncMetadata	VersionSyncMetadata	關於僅對該同步執行時間有興趣之此版本之額外版本特定元資料

b) 墓碑

該資料儲存為項目、擴充與關係提供墓碑資訊。該墓碑檢視於一處提供關於生活與墓碑實體(項目、擴充與關係)之資訊。該項目與擴充墓碑檢視不提供存取至該對應物

件，而該關係墓碑檢視提供存取至該關係物件(該關係物件於一墓碑關係之情況中係 NULL)。

(1) 項目墓碑

項目墓碑經由該檢視[System.Storage].[Tombstone!Item]自該系統取出。

列	類型	說明
ItemId	ItemId	該項目之身分
_TypeID	TypeId	該項目之類型
<Item properties>	...	為所有項目定義之屬性
_RootItemId	ItemId	包含此項目之第一非嵌入項目之 ItemId。
_ChangeTrackingInfo	類型 ChangeTrackingInfo 之 CLR 範例	此項目之變更追蹤資訊
_IsDeleted	BIT	此為一旗標，為生活項目為 0，而為墓碑項目為 1。
_DeletionWallclock	UTCDATETIME	根據刪除該項目之夥伴之 UTC 牆鐘日期時間。若該項目活著時其為 NULL。

(2) 擴充墓碑

擴充墓碑係利用該檢視[System.Storage].[Tomstone!Extension]自該系統取出。擴充變更追蹤資訊類似於為項目所提供者，加上該ExtensionId屬性。

列	類型	說明
ItemId	ItemId	擁有該擴充之項目之身分
ExtensionId	ExtensionId	該擴充之擴充Id
_TypeID	TypeId	該擴充之類型
_ChangeTrackingInfo	類型ChangeTrackingInfo之CLR範例	此擴充之變更追蹤資訊
_IsDeleted	BIT	此為一旗標，為生活項目為0，而為墓碑擴充為1。
_DeletionWallclock	UTCDATETIME	根據刪除該擴充之夥伴之UTC牆鐘日期時間。若該擴充活著時為NULL。

(3) 關係墓碑

關係墓碑經由該檢視[System.Storage].[Tombstone!Relationship]自該系統取出。關係墓碑資訊類似於為擴充所提供者。然而，額外資訊提供於該關係範例之目標ItemRef上。除此之外，該關

係物件亦可選擇。

列	類型	說明
ItemId	ItemId	擁有該關係之項目身分（關係來源端點之身分）
RelationshipId	RelationshipId	該關係之RelationshipId
_TypeID	TypeId	該關係之類型
_ChangeTrackingInfo	類型ChangeTrackingInfo之CLR範例	此關係之變更追蹤資訊
_IsDeleted	BIT	此為一旗標，為生活項目為0，而為墓碑擴充為1。
_DeletionWallclock	UTCDATETIME	根據刪除該關係之夥伴之UTC牆鐘日期時間。若該關係活著則為NULL。
_Relationship	一關係之CLR範例	此為生活關係之關係物件。其為墓碑關係為NULL。
TargetItemReference	ItemReference	目標端點之身分

(4) 墓碑清除

為了避免墓碑資訊毫無邊際的成長，該資料儲存提供一墓碑清除工作。此工作決定墓碑資訊何時可被捨棄。該工作計算該區域建立/更新版本上之一邊界，且接著藉由丟棄所有稍早墓碑版本截斷該墓碑資訊。

11. 幫助器 API 與功能

該基本映射亦提供許多幫助器功能。這些功能提供以透過該資料模型幫助共用操作。

a) Function [System.Storage].GetItem

```
Returns an Item object given an ItemId
//
Item GetItem (ItemId ItemId)
```

b) Function [System.Storage].GetExtension

```
// Returns an extension object given an ItemId and ExtensionId
//
Extension GetExtension (ItemId ItemId, ExtensionId ExtensionId)
```

c) Function [System.Storage].GetRelationship

```
// Returns an relationship object given an ItemId and
    RelationshipId
//
Relationship GetRelationship (ItemId ItemId, RelationshipId
    RelationshipId)
```

12. 元資料

有兩類型的元資料呈現於該儲存中：立即元資料(一項目之類型等等)與類型元資料。

a) 架構元資料

架構元資料儲存於該資料儲存中，做為來自該元架構之項目類型之範例。

b) 立即元資料

立即元資料由一應用程式使用以查詢一項目之類型，並尋找與一項目相關之擴充。為一項目給予該 ItemId，一應用程式可查詢該全域項目檢視，以傳回該項目之類型，並使用此值以查詢該 Meta.Type 檢視，以傳回該項目之已宣告類型上之資訊。舉例來說，

```
// Return metadata Item object for given Item instance
//
SELECT m._Item AS metadataInfoObj
FROM [System.Storage].[Item] i INNER JOIN [Meta].[Type] m
    ON i._TypeId = m.ItemId
WHERE i.ItemId = @ItemId
```

G. 安全性

一般來說，所有可安全性物件利用第 26 圖中所示之存取遮罩格式安排其存取權限。在此格式中，該低順序 16 位元為物件特定存取權限，下 7 個位元為標準存取權限，應用至大部分物件類型，且該 4 個高順序位元用於指定一般存取權限，各物件類型可映射至一組標準與物件特定權限。該 ACCESS_SYSTEM_SECURITY 位元對應至存取該物件之 SACL 之權限。

在第 26 圖之存取遮罩結構中，項目特定權限放置於該物件特定權限一節中(低順序 16 位元)。由於在本實施例中，該儲存平台揭示二組 API 以管理安全性—Win32 與該儲存平台 API，必須考慮該檔案系統物件特定權限，以激勵該儲存平台物件特定權限之設計。

本發明之儲存平台之安全模型完整描述於稍早在此併入參照之相關申請案中。如此一來，第 27 圖(部分 a、b、

及c)根據一安全性模型之一實施例描述自一既有安全區域取出之一新相同保護安全區域。

H. 通知與變更追蹤

根據本發明之另一方面，該儲存平台提供允許應用程式追蹤資料變更之一通知功能。此特性主要用於維護揮發性狀態或執行商業邏輯於資料變更事件上之應用程式。應用程式為項目、項目擴充及項目關係上之通知加以登記。通知於資料變更已遞送之後非同步化傳遞。應用程式可由項目、擴充與關係類型以及操作類型過濾通知。

根據一實施例，該儲存平台API 322為通知提供兩類介面。首先，應用程式為由變更至項目、項目擴充及項目關係觸發之簡單資料變更事件加以註冊。第二，應欲程式建立「觀察者」物件，以監控項目、項目擴充與項目間關係組。於一系統失敗後，或於一系統於一擴充時段後離線後，一觀察者物件之狀態可儲存與重新建立。單一通知可反映多個更新。

關於此功能之額外細節可於稍早併入參照之相關申請案中找到。

I. 同步化

根據本發明之另一方面，該儲存平台提供一同步化服務330，(i)允許該儲存平台之多個範例(各具有其本身之資料儲存302)以根據一彈性規則組同步化其內容之部分，及(ii)為第三者提供一基礎架構以同步化本發明之儲存平台之資料儲存與實施專屬通訊協定之其他資料來源。

儲存平台-至-儲存平台同步化發生於一群參與副本中。舉例來說，參照第3圖，可能想要在該儲存平台之另一範例之控制下提供該儲存平台300之資料儲存302與另一遠端資料儲存338間之同步化，也許執行於一不同電腦系統上。此群組之總成員不需要於任何給定時間了解於任何給定副本。

不同副本可做獨立變更(即同時)。同步化之處理定義為其他副本所作變更發覺之每個副本。此同步化功能係繼承多主。

本發明之同步化功能允許副本以：

- 決定另一副本發覺之何變更；
- 要求關於此副本未發覺之變更之資訊；
- 傳達關於其他副本未發覺之變更之資訊；
- 決定而變更何時互相衝突；
- 區域應用變更；
- 傳達衝突解決方法至其他副本，以確保收斂性；及
- 根據衝突解決方法之特定策略解決該衝突。

1. 儲存平台-至-儲存平台同步化

本發明之儲存平台之同步化服務330之主要應用係同步化儲存平台之多個範例(各具有其本身資料儲存)。該同步化服務操作於該儲存平台架構層(而非該資料庫引擎314之基礎表)。因此，舉例來說，「範圍」用於定義同步化組，如下所討論者。

該同步化服務操作於「網變更」之原則上。並非記錄與傳送個別操作(例如具有處理複製)，該同步化服務傳送這些操作之最終結果，因此常常將多個操作之結果統一於單一結果變更上。

該同步化服務一般不尊重處理界線。換句話說，若二變更做於單一處理中之一儲存平台資料儲存，不保證這些變更會自動應用於所有其他副本—可能一個顯示，而另一不會顯示。此原則之例外是若於相同處理中對相同項目做二變更，則這些變更保證會自動被傳送且應用至其他副本。因此，項目會是該同步化服務之一致單元。

a) 同步化控制應用程式

任何應用程式可連結至該同步化服務，並開始一同步化操作。此一應用程式提供必須執行同步化之所有參數(請見下方之同步化設定檔)。此應欲程式在此稱為同步化控制應用程式(SCA)。

當同步化兩個儲存平台範例時，同步化由一 SCA 初始化於一側。該 SCA 通知該區域同步化服務以同步化該遠端夥伴。另一方面，該同步化服務由來自該原始機器之同步化服務所傳送之訊息喚醒。其根據呈現於該目的機器上之永久配置資訊(請見下方之映射)加以回應。該同步化服務可執行於排程上或響應至事件。在這些情況中，實施該排程之同步化服務會變成該 SCA。

為了啟動同步化，必須採取兩個步驟。首先，該架構設計人員必須以適當同步語義(指定變更單元，如下所述)

注釋該儲存平台架構。第二，同步化必須適當配置於具有參與該同步化之儲存平台之一範例之所有機器上(如下所述)。

b) 架構注釋

該同步化服務之一基礎概念係一變更單元。一變更單元係由該儲存平台個別追蹤之一最小架構片段。對於每個變更單元來說，該同步化服務可決定是否其由於該最新同步化變更與否。

指定該架構中之變更單元做為幾個目的。首先，其決定該同步化服務在該線上有多繞舌。當於一變更單元中做一變更時，由於該同步化服務不知道該變更單元之何部分變更，該整個變更單元會傳送至其他副本。第二，其決定衝突偵測之顆粒度。當對相同變更單元做兩個同時變更(這些詞在下幾節中會詳細定義)時，該同步服務會發生一衝突；另一方面，若對不同變更單元做同時變更，則沒有衝突會發生，且該變更會自動合併。第三，其強烈影響由該系統保持之元資料數量。許多同步化服務元資料為各變更單元加以保持；因此，將變更單元做更小會增加同步之負載。

定義變更單元需要尋找正確的斟酌。基於這個理由，該同步化服務允許架構設計人員參與該處理中。

在一實施例中，該同步化服務不支援大於一元件之變更單元。然而，其的確支援架構設計人員指定小於一元件之變更單元之能力，也就是將一元件之多個屬性群聚於一

不連續的變更單元中。在此實施例中，利用下列與法加以完成：

```
<Type Name="Appointment" MajorVersion="1" MinorVersion="0"
ExtendsType="Base.Item"
        ExtendsVersion="1">

    <Field Name="MeetingStatus“  Type="the storage platformTypes.uniqueidentifier
Nullable="False"/>
    <Field Name="OrganizerName“ Type="the storage platformTypes.nvarchar(512)"
Nullable="False"/>
    <Field Name="OrganizerEmail“  Type="the storage platformTypes.nvarchar(512)"
        TypeMajorVersion="1“            MultiValued="True"/>
    ...
    <ChangeUnit Name=”CU_Status”>
        <Field Name=”MeetingStatus”/>
    </ChangeUnit>

    <ChangeUnit Name=”CU_Organizer”/>
        <Field Name=”OrganizerName” />
        <Field Name=”OrganizerEmail” />
    </ChangeUnit>
    ...

</Type>
```

c) 同步化配置

希望保持其資料之特定部份同步之一群儲存平台夥伴稱為一同步化社區(community)。該社區之成員想要保持同步，然而其不需要以完全相同之方式呈現該資料；換句話說，同步化夥伴可轉換其同步化之資料。

在一對等情況中，對等者為所有其夥伴維護轉換映射是不實用的。而是，該同步化服務採取定義「社區資料夾」之方法。一社區資料夾係代表所有社區成員同步化之一假想「共享資料夾」之一抽象。

此概念最好由一範例加以舉例說明。若 Joe 想要保持他幾台電腦的我的文件資料夾同步，Joe 定義一社區資料

夾，稱為 JoesDocuments。接著，在每一電腦上，Joe 規劃該假想 JoesDocuments 資料夾與該區域我的文件資料夾間之一映射。從此觀點來看，當 Joe 的電腦互相同步時，其會根據 JoesDocument 中之文件加以討論，而非其區域項目。如此一來，所有 Joe 的電腦會互相了解，而不需要知道其他人是誰—該社區資料夾變成該同步化社區之社交語言。

規劃該同步化服務由三個步驟所組成：(1)定義區域資料夾與社區資料夾間之映射；(2)定義決定何者或得同步化(如與誰同步化及應傳送何子組及接收何者)之同步化設定檔；及(3)定義不同同步化設定檔應執行之排程，或手動執行。

(1) 社區資料夾—映射

社區資料夾映射儲存為個別機器上之 XML 配置檔。各映射具有下列架構：

/mappings/communityFolder

此元件為此映射代表之社區資料夾命名。該名稱跟隨資料夾之語法規則。

/mappings/localFolder

此元件為該映射轉換至之區域資料夾命名。該名稱跟隨資料夾之語法規則。該資料夾必須已經為有效之映射加以存在。此資料夾中之項目視為各映射之同步化。

/mappings/transformations

此元件定義如何自該社區資料夾轉換項目至該區域資

料夾，及轉換回去。若不存在或為空，不執行任何轉換。尤其是，此意指沒有 ID 被映射。此配置主要用於建立一資料夾之一快取記憶體。

/mappings/transformations/mapIDs

此元件要求新產生的區域 ID 指派至映射自該社區資料夾之所有項目，而非重新使用社區 ID。該同步化執行時間會維護 ID 映射，以來回轉換項目。

/mappings/transformations/localRoot

此元件要求該社區資料夾中所有根項目必須做該特定根之子。

/mappings/runAs

此元件控制處理何授權針對此映射要求。若不存在，假設為傳送者。

/mappings/runAs/sender

此元件之存在指示必須扮演對此映射之訊息傳送者，且要求其憑證下之處理。

(2) 設定檔

一同步化舍弟檔戲必須發動同步化之一總參數組。由一 SCA 提供至該同步化執行時間以初使同步化。儲存平台-至-儲存平台同步化之同步化設定檔包含下列資訊：

- 區域資料夾，做為變更之來源與目的；
- 同步化之遠端資料夾名稱—此資料夾必須界猶如上所定義之一映射自該遠端夥伴公佈；
- 方向—該同步化服務支援僅傳送、僅接收、及傳

送-接收同步化；

- 區域過濾器—選擇何區域資訊傳送至該遠端夥伴。表現為透過該區域資料夾之一儲存平台查詢；
- 遠端過濾器—選擇何遠端資訊自該遠端夥伴取出—表現為透過該社區資料夾之一儲存平台查詢；
- 轉換—定義如何轉換項目自或至該區域格式；
- 區域安全性—指示自該遠端端點取出之變更是否於該遠端端點(扮演)許可下或該使用者區域初始化該同步化加以應用；及
- 衝突解決方法策略—指示是否應拒絕、登錄、或自動解決衝突—在後者的情況中，其指示使用何衝突解決器，以及其配置參數。

該同步化服務提供允許簡單建立同步化設定檔之一執行時間CLR類別。設定檔亦可為簡易儲存被序列化至與自XML檔(通常跟著排程)。然而，該除殂平台中沒有標準位置是所有射並檔儲存處；SCA歡迎建立一設定檔於該點上，而不會加以堅持。注意沒有必要具有一區域映射以初始同步化。所有同步化資訊可指定於該設定檔中。然而，需要該映射，以回應該遠端側所初始之同步化要求。

(3) 排程

在一實施例中，該同步化服務部提供其本身之排程基礎架構。而是，其依賴另一元件以執行此工作—該Windows

Scheduler，可於該 Microsoft Windows 作業系統獲得。該同步化服務包含做為一 SCA 之一指令行公用程式，並根據儲存於一 XML 檔中之一同步化設定檔觸發同步化。此公用程式使其非常容易配置該 Windows Scheduler 執行同步化於該排程上，或響應至該事件，例如使用者登入或登出。

d) 衝突處理

該同步化服務中之衝突處理分成三階段：(1)衝突偵測，發生於變更應用程式時間—此步驟決定是否一變更可安全應用；(2)自動衝突解決方式與登錄—在此步驟期間(立即發生於該衝突被偵測到之後)，會諮詢自動衝突解決器，以看看是否該衝突可被解決—若否，該衝突可被選擇性登錄；及(3)衝突偵查與解決方式—此步驟發生於若某些衝突被登錄時，且發生於該同步節之內容之外—在此時，已登錄衝突可被解決並自該登錄移除。

(1) 衝突偵測

在此實施例中，該同步服務偵測兩類型之衝突：知識型與限制型。

(a) 知識型衝突

一知識型衝突發生於兩個副本對相同變更單元做獨立變更時。若不互相知會，兩個變更稱為獨立—換句話說，該第一版本不被該第二之知識所涵蓋，反之亦然。該同步化服務根據該副本之知識自動偵測所有此類衝突，如上所述。

有時候想想一變更單元之版本歷史中之衝突是很有幫

助的。若沒有衝突發生於一變更單元之生命中，其版本歷史係一簡單連鎖—各變更發生於前一者之後。在一知識型衝突之情況中，兩個變更平行發生，導致該連鎖分開，並變成一版本樹。

b) 限制型衝突

有些情況獨立變更會在一起應用時違反一整合限制。例如，建立具有該相同目錄中相同名稱之一檔案之二副本可能導致此一衝突發生。

一限制型衝突包含兩個獨立變更(正如同一知識型衝突)，但其不會影響相同的變更單元。而是，其會影響不同變更單元，但具有一限制存在於期間。

該同步服務偵測限制違反於變更應用時間，並自動發生限制型衝突。解決限制型衝突通常需要修改該變更之自訂碼，如此一來便不會違反該限制。該同步服務不會為這麼做提供一通用目的機制。

(2) 衝突處理

當偵測到一衝突時，該同步服務可採取三個動作其中之一(由該同步化設定檔中同步化初始器加以選擇)：(1)拒絕該變更，將其傳回傳送者；(2)登錄一衝突至一衝突登錄中；或(3)自動解決該衝突。

若該變更被拒絕，該同步化服務做為如同該變更不到達該副本。一負面知識傳送回該原創者。此解決方式策略原始有用於無頭副本(例如檔案伺服器)上，其中登錄衝突不可行。而是，此副本強迫其他者計拒絕其處理該衝突。

同步化初始器配置衝突解決方法於其同步化設定檔中。該同步化服務以下列方式支援結合多個衝突解決器於一單一設定檔中—首先，藉由一一指派欲嘗試之一衝突解決器清單，直到其中之一成功為止；且第二，藉由關聯衝突解決器與衝突類型，如將更新-更新知識型衝突導向一解決器，但是所有其他衝突至該登錄。

(a) 自動衝突解決方法

該同步化服務提供許多預設衝突解決器。此清單包含：

- 區域 wins：若與區域儲存資料相衝突，則不管輸入變更；
- 遠端 wins：若與輸入變更相衝突，則不管區域資料；
- 最新撰寫者 wins：根據該變更之時間戳為各變更挑出區域 wins 或遠端 wins(注意該同步化服務一般不會依賴時鐘值；此衝突解決器係該規則之主要例外)。

確定性：以保證於所有副本上相同之方式挑出一贏家，否則並無意義—該同步化服務之一實施例使用夥伴 ID 之詞典比較，以實施此特性。

除此之外，ISV 可實施與安裝其本身之衝突解決器。自訂衝突解決器可接受配置參數；此參數必須由該同步化檔案之衝突解決方式部分中之 SCA 加以指定。

當一衝突解決器處理一衝突時，其傳回必須執行之操作清單(而非該衝變更)回該執行時間。該同步化服務接著

應用這些操作，已適當調整遠端知識，以包含已考慮何衝突處理器。

當應用該解決方法時偵測到另一衝突是可能的。在此一情況中，該新衝突必須在該原始處理繼續之前解決。

當將衝突考慮為一項目之版本歷史之分支時，衝突解決方法可視為結合—合併兩個分支以形成單一點。因此，衝突解決方法會將版本歷史轉入 DAG 中。

(b) 衝突登錄

一衝突解決器之一非常特殊種類係該衝突登錄器。該同步化服務登錄衝突為類型 ConflictRecord 之項目。這些記錄關於回衝突之項目(除非該項目本身已被刪除)。各衝突記錄包含：導致該衝突之輸入變更，該衝突之類型：更新-更新、更新-刪除、刪除-更新、插入-插入、或限制；且該輸入變更之版本與該副本之知識傳送它。登錄衝突可用於偵查與解決方式，如下所述。

(c) 衝突偵查與解決方式

該同步服務為應用程式提供一 API，以檢查該衝突登錄，並建議其中衝突之解決方式。該 API 允許應用程式列舉所有衝突，或關於一給定項目之衝突。其亦允許此應用程式以三種方式其中之一解決已登錄衝突：(1)遠端 wins—接受該已登錄變更與覆寫該衝突區域變更；(2)區域 wins—忽略該已登錄變更之衝突部份；及(3)建議新變更—其中該應用程式提議一合併，在其建議中解決該衝突。一旦衝突由一應用程式解決，該同步化服務會自該登錄加以移除。

(d) 副本之收斂與衝突解決方式之傳播

在複雜同步化情況中，相同衝突可於多個副本處被偵測。若此發生，幾件事可能會發生：(1)該衝突可於一副本上解決，且該解決方法會傳送至其他副本；(2)該衝突自動於兩個副本上解決；或(3)該衝突手動於兩個副本上解決(透過該衝突偵查 API)。

為了確保收斂，該同步化服務將衝突解決方式傳至其他副本。當解決一衝突之一變更到達一副本，該同步化服務會自動尋找由此更新加以解決之登錄中之任何衝突記錄並加以消滅。如此一來，於一副本之一衝突解決方式結合至所有其他副本。

若不同贏家由於相同衝突由不同副本選擇，該同步化服務應用結合衝突解決方法之原則，並挑出兩個解決方法其中之一，以自動贏另外一者。該贏家以一決定性方式挑出，保證每次會產生相同結果(一實施例使用複製 ID 詞典比較)。

若不同「新變更」由不同副本為相同衝突加以建議，該同步化服務會將此新衝突視為一特殊衝突，並使用該衝突登錄器以避免其傳播至其他副本。此情況一般會以手動衝突解決方式時發生。

2. 同步化至非儲存平台資料儲存

根據本發明之儲存平台之另一方面，該儲存平台為 ISV 提供一架構，以實施允許該儲存平台同步化至遺留系統之同步化轉接器，例如 Microsoft Exchange、AD、Hotmail

等等。同步化轉接器對許多由該同步化服務所提供之同步化服務很有利，如下所述。

除了該名稱之外，同步化轉接器不需要實施為插入某個儲存平台架構。若想要，一「同步化轉接器」可僅為利用該同步化服務執行時間介面以獲得服務之任何應用程式，如變更列舉與應用程式。

為了讓其他人較容易配置與執行同步化至一給定後端，同步化轉接器被鼓勵以揭示該標準同步化轉接器介面，其執行同步化給定該同步化設定檔，如上所述。該設定檔提供配置資訊至該轉接器，某些轉接器傳送至該同步化執行時間，以控制執行時間服務(如該資料夾以同步化)。

a) 同步化服務

該同步化服務提供許多同步化服務至轉接器撰寫者。對於此節之剩餘部份來說，很方便將該儲存平台執行同步化之機器稱為該「客戶端」，而將該轉接器談話對象之非儲存平台後端為該「伺服器」。

(1) 變更列舉

根據由該同步化服務所維護之變更追蹤資料，變更列舉允許同步化轉接器很容易列舉已發生於一資料儲存資料夾之變更，由於已嘗試與此夥伴之最後一次同步化。

變更根據一「支柱」之概念加以列舉—呈現關於該最新同步化之資訊之一不透明結構。該支柱採取該儲存平台知識之形式，如前幾節中所述。利用變更列舉服務之同步化轉接器落入兩個廣泛類別中：使用「儲存支柱」者 vs.

使用「供應支柱」者。

差異係根據關於該最新同步化之資訊儲存處而定—在該客戶端上，或在該伺服器上。對轉接器來說常常較容易儲存此資訊於該客戶端上—該後端常常不可便利儲存此資訊。另一方面，若多個客戶端同步化至相同後端，儲存此資訊於該客戶端上係較無效率且在某些情況中是不正確的—其使一客戶端不查覺其他客戶端已經推至該伺服器之變更。若一轉接器想要使用一伺服器儲存支柱，該轉接器必須於變更列舉時將其提供回該儲存平台。

為了讓該儲存平台維護該支柱(無論為區域或遠端儲存)，該儲存平台必須查學已成功應用於該伺服器之變更。這些與僅這些變更可包含於該支柱中。在變更列舉期間，同步化轉接器使用一認可介面已報告何變更已成功應用。在同步化尾聲，利用提供支柱之轉接器必須讀取該新支柱(併入所有已成功應用之變更)，並將其傳至其後端。

通常，轉接器必須儲存轉接器特定資料與其插入該儲存平台資料儲存中之項目。此資料之常見範例為遠端 ID 及遠端版本(時間戳)。該同步化服務提供用以儲存此資訊之一機制，且變更列舉提供一機制以提供此額外資料與正傳回之變更。這會在大部分情況中為轉接器減少重新查詢該資料庫之需求。

(2) 變更應用程式

變更應用程式允許同步化轉接器應用自其後端接收之變更至該區域儲存平台。轉接器預期將該變更轉換至該儲

存平台架構。第 24 圖舉例說明該處理，其中儲存平台 API 類別自該儲存平台架構產生。

變更應用程式之主要功能係自動偵測衝突。如同在儲存平台-至-儲存平台同步化之情況中，一衝突定義為做成之二重複變更，而無互相知會。當轉接器使用變更應用程式時，其必須指定關於執行何衝突偵測之支柱。若偵測不由該轉接器之知識所涵蓋之一重複區域變更，變更應用程式發生一衝突。與變更列舉相似，轉接器可使用儲存或提供支柱。變更應用程式支援轉接器特定元資料之有效儲存。此資料可由該轉接器附加於被應用之變更，且可由該同步化服務加以儲存。該資料可於下一變更列舉上傳回。

(3) 衝突解決方式

上述衝突解決方式機制(登錄與自動解決方式選項)同樣對同步化轉接器有效。同步化轉接器可在應用變更時指定衝突解決方式之策略。若指定，衝突可傳送至該特定衝突處理器且被解決(若可能)。衝突亦可被登錄。當嘗試應用一區域變更至該後端時，該轉街器偵測一衝突是可能的。在此一情況中，該轉接器可仍傳送該衝突至欲根據策略解決之同步化執行時間。除此之外，同步化轉接器可要求任何由該同步化服務偵測之衝突傳回加以處理。此尤其在該後端可儲存或解決衝突的情況中特別方便。

b) 轉接器實施方式

某些「轉接器」為利用執行時間介面之簡單應用程式時，轉接器被鼓勵以實施該標準轉接器介面。這些介面允

許同步化控制應用程式以：要求該轉接器根據一給定同步化設定檔執行同步化；取消同步化；及接收一進行中同步化之進度報告(完成百分比)。

3. 安全性

該同步化服務努力盡少引入由該儲存平台所實施之安全性模型中。不為同步化定義新權限，使用既有之權限。尤其是，

- 可讀取一資料儲存項目之任何人可列舉變更至該項目；
- 可寫入一資料儲存項目之任何人可應用變更至該項目；及
- 可擴充一資料儲存項目之任何人可將同步化元資料與該項目相關聯。

該同步化服務不維護安全性作者資訊。當一變更由使用者U於副本A做成且傳至副本B時，事實是於A(或由U)原始做成之變更會遺失。若B將此變更傳至副本C，此會在B的授權下完成，而非A的授權下。此導致下列限制：若不相信一副本做成其本身變更至一項目，其無法傳送由其他者所做之變更。

當初使化該同步化服務時，其由一同步化控制應用程式加以完成。該同步化服務扮演該SCA之身分，並執行該身分下之所有操作(區域與遠端)。為了舉例說明，觀察使用者U無法導致該區域同步化服務，以為使用者U沒有讀取權限之項目自一遠端儲存平台取出變更。

4. 可管理性

監控副本之一分散式社區係一複雜問題。該同步化服務可使用一「掃除」演算法，以收集與散佈關於該副本狀態之資訊。該掃除演算法之屬性確保關於所有已配置副本之資料最終會被收集，且會偵測失敗(不回應)之副本。

此社區型監控資訊可於每個副本處做成。監控工具可執行於一任意選擇副本處，以檢查此間控資訊並做成管理決策。任何配置變坑必須直接於該已影響之副本做成。

J. 傳統檔案系統相互操作性

如上所述，在至少某些實施例中，本發明之儲存平台想要實施為一電腦系統之硬體/軟體介面系統之一整合部分。舉例來說，本發明之儲存平台可實施為一作業系統之一整合部分，例如作業系統之 Microsoft Windows 家族。在該容量中，該儲存平台 API 變成該作業系統 API 之一部分，其中應用程式透過此與該作業系統互動。因此，該儲存平台變成應用系統透過此儲存資訊於該作業系統上之構件，且該儲存平台之項目型資料模型因此會取代此一作業系統之傳統檔案系統。舉例來說，如同實施於作業系統之 Microsoft Windows 家族中者，該儲存平台可能取代實施於該作業系統中之 NTFS 檔案系統。目前，應用程式透過由該作業系統之 Windows 家族所揭示之 Win32 API 存取 NTFS 檔案系統之服務。

然而，了解完全以本發明之儲存平台取代該 NTFS 檔案系統會需要記錄既有的 Win32 型應用程式，且此記錄可

能不想要，對本發明之儲存平台來說提供與既有檔案系統之某些相互操作性可能是有利的，例如 NTFS。因此，在本發明之一實施例中，該儲存平台允許依賴該 Win32 程式設計模型之應用程式存取該儲存平台之資料儲存以及該傳統 NTFS 檔案系統之內容。最後，該儲存平台使用該 Win32 命名規則之一超級組之一命名規則，以促進容易的相互操作性。此外，該儲存平台支援透過該 Win32 API 儲存於一儲存平台量中之檔案與目錄。

關於此功能之額外細節可於稍早在此併入參照之相關申請案中找到。

K. 儲存平台 API

該儲存平台包含允許應用程式存取上面所討論之儲存平台之特性與功能，以及存取儲存於該資料儲存中之項目之一 API。本節描述本發明之儲存平台之一儲存平台 API 之一實施例。關於此功能之細節可於稍早在此併入參照之相關申請案中找到，為了方面，在下方歸納某些資訊。

請參閱第 18 圖，一包含資料夾係包含至其他項目之持有關係，且係相等於一檔案系統資料夾之一共用概念之一項目。各項目係「包含」於至少一包含資料夾中。

第 19 圖根據本發明舉例說明該儲存平台 API 之基本架構。該儲存平台 API 使用 SQL 客戶端 1900 與該區域資料儲存 302 對談，且亦可使用 SQL 客戶端 1900 與遠端資料儲存對談(如資料儲存 340)。該區域儲存 302 亦可利用 DQP(分散式查詢處理器)或透過上述儲存平台同步化服務

(「同步化」)與該遠端資料儲存 340 對談。該儲存平台 API 322 亦為資料儲存通知做為該橋 API，傳送應用程式的訂閱至該通知引擎 332，並將通知路由至該應用程式(如應用程式 350a、350b、或 350c)，亦如上所述。在一實施例中，該儲存平台 API 322 亦可定義一有限「提供者」架構，使得其可存取 Microsoft Exchange 與 AD 中之資料。

第 20 圖以架構方式呈現該儲存平台 API 之各種不同元件。該儲存平台 API 由下列元件所組成：(1)資料類別 2002，其呈現該儲存平台元件與項目類型，(2)執行時間架構 2004，其管理物件持久性並提供支援類別 2006；及(3)工具 2008，其用於自該儲存平台架構產生 CLR 類別。

自一給定架構導致之類別階層會直接反映該架構中之類型階層。做為一範例，考慮定義於該接觸架構中之項目類型，如第 21A 圖與第 21B 圖中所示。

第 22 圖舉例說明操作中之執行時間架構。該執行時間架構操作如下：

1. 一應用程式 350a、350b、或 350c 結合至該儲存平台中之一項目。
2. 該架構 2004 建立對應至該已結合項目之一 ItemContext 物件 2202。
3. 該應用程式傳送此 ItemContext 上之一尋找，已獲得一項目及何；該傳回之集合在概念上係一物件圖 2204(由於關係)。
4. 該應用程式變更、刪除、及插入資料。

5. 該應用程式藉由呼叫該 Update()方法儲存該變更。

第 23 圖舉例說明一「FindAll」操作之執行。

第 24 圖舉例說明該處理，其中該儲存平台 API 類別自該儲存平台架構產生。

第 25 圖舉例說明該檔案 API 依據之架構。該儲存平台 API 包含用以處理檔案物件之一名稱領域。此名稱領域稱為 System.Storage.Files。System.Storage.Files 中類別之資料成員直接反映儲存於該儲存平台儲存中之資料；此資訊自該檔案系統物件「升級」，或可利用該 Win32 API 原生建立。該 System.Storage.Files 名稱領域具有兩個類別：FileItem 與 DirectoryItem。這些類別之成員及其方法可藉由看著第 25 圖中之架構圖快速估計。FileItem 與 DirectoryItem 係來自該儲存平台 API 之唯讀。為了修改它們，必須使用該 Win32 API 或 System.IO 中之類別。

關於 API，一程式設計介面(或更簡單的說，介面)可被視為任何允許一或多個程式碼片段與一或多個其他程式碼片段所提供之功能通訊或加以存取之機制、程序、通訊協定。或者，一程式設計介面可被視為一或多個可通訊耦合至其他元件之一或多個機制、方法、函數呼叫、模組、物件之一系統之一元件之機制、方法、函數呼叫、模組、物件等等。前一句中「程式碼片段」一詞意欲包含一或多個程式碼指令或行，且包含如程式碼模組、物件、副程式、函數等等，不論應用的術語為何，或是否該程式碼片段個

別編譯，或是否該程式碼片段提供為來源、立即、或物件程式碼，或是否該程式碼片段用於一執行時間系統或處理中，或是否為於相同或不同機器上或跨多個機器散佈，或是否由該程式碼片段所呈現之功能完全實施於軟體中、完全實施於硬體中、或硬體與軟體之一組合。

概念上，一程式設計介面可大致檢視，如第30A圖或第30B圖中所示。第30A圖舉例說明一介面介面1做為一導管，第一與第二程式碼片段透過其通訊。第30B圖舉例說明一介面包含介面物件I1與I2(其可為或可不為該第一與第二程式碼片段之部分)，允許一系統之第一與第二程式碼片段經由媒體M加以通訊。在檢視第30B圖中，可將介面物件I1與I2考慮為相同系統之個別介面，且亦可考慮物件I1與I2加上媒體M包含該介面。雖然第30A與第30B圖顯示雙向流程且介面於流程各側上，特定實施例可僅具有資訊流程於一方向(或如下所述般沒有資訊流程)，或僅具有一介面物件於一側。藉由範例但非加以限制，例如應用程式介面(API)、輸入點、方法、函數、副程式、遠端程序呼叫、及元件物件模型(COM)介面之詞皆囊括於程式設計介面之定義中。

此一程式設計介面之方面可包含該第一程式碼片段傳送資訊(其中「資訊」用於其最廣泛解釋中，且包含資料、命令、要求等等)至該第二程式碼片段之方法、該第二程式碼片段接收該資訊之方法、及該資訊之結構、順序、語法、組織、架構、時間及內容。如此一來，該基本傳輸媒體本

身可能對該介面之操作、是否該媒體為有線或無線或兩者之一組合並不重要，只要該資訊以該介面所定義之方法加以傳輸。在特定情況中，當該資訊傳輸可經由另一機制(如放置於一緩衝器、檔案等等中之資訊自該程式碼片段間之資訊流程分離)或不存在時，或當一程式碼片段存取由一第二程式碼片段所執行之功能時，資訊可能不在傳統所知中之單或雙向傳送。任何或所有方面可能在一給定情況中很重要，如根據是否該程式碼片段為一鬆耦合或緊耦合配置中一系統之部分，且因此此清單應被視為舉例說明而非加以限制。

此一程式設計介面之概念對熟知該項技藝人士來說係為已知，且在本發明前述詳細說明中可清楚獲得。然而，有其他方法以實施一程式設計介面，除非表示排除在外，且這些也意欲由此說明書尾端之申請專利範圍所囊括。此其他方法可出現為比第30A與第30B圖之簡單檢視更精細或複雜，但然而，其執行一類似功能以完成相同之全面結果。我們現在將簡單描述一程式設計介面之某些舉例說明之其他實施方式。

因數。來自一程式碼片段至另一程式碼片段之一通訊可間接藉由將該通訊分成多個不連續的通訊加以完成。此以架構方式描述於第31A與第31B圖中。如圖所示，某些介面可描述代表可切割功能組。因此，第30A與第30B圖之介面功能可被因數化，以達成相同之結果，正如同可以數學方式提供24，或2乘上2乘上3乘上2。因此，如第

31A 圖中所示，介面介面 1 所提供之功能可被子切割，以將該介面之通訊轉成多個介面介面 1A、介面 1B、介面 1C 等等而仍達成相同結果。如第 31B 圖中所示，介面 I1 所提供之功能可被子切割至多個介面介面 I1a、I1b、I1c 等等而仍達成相同結果。同樣的，自該第一程式碼片段接收資訊之第二程式碼片段之介面 I2 可被因數化至多個介面 I2a、I2b、I2c 等等。當因數化時，包含該第 1 程式碼片段之介面數不需與包含該第 2 程式碼片段之介面數相符。在第 31A 與第 31B 圖之情況中，介面介面 1 與 I1 之功能精神分別維持與第 30A 與第 30B 圖相同。介面之因數化亦可跟隨相聯、交換、及其他數學屬性，使得該因數可能很難辨認。例如，操作之順序可能不重要，且因此，由一介面執行之一功能可能藉由另一程式碼片段或介面，或藉由該系統之一個別元件加以執行以執行進一步到達該介面。此外，熟知該程式設計技藝人士可了解有各種不同方法做出達成相同結果之功能呼叫。

重新定義。在某些情況中，可能在仍完成想要結果時忽略、新增或重新定義一程式設計介面之特定方面(如參數)。此舉例說明於第 32A 與第 32B 圖中。舉例來說，假設第 30A 之介面介面 1 包含一功能呼叫 Square(input,precision,output)，包含三個參數 input、precision 與 output 之一呼叫，且其自該第 1 程式碼片段提出至該第 2 程式碼片段。若該中間參數精度不於一給定情況中加以考慮，如第 32A 圖中所示，可以僅忽略，或甚至

以一無意義(在此情況中)參數加以取代。亦可加入無關之一額外參數。在任一事件中，可達成平方之功能，只要在輸入由該第二程式碼片段平方後傳回輸出。精度對某些下流或該計算系統之其他部分來說可能是一有意義之參數；然而，一旦認知到精度對於計算平方之狹窄目的並非必要時，其可被取代或被忽略。舉例來說，不傳送一有效精度值，例如一生日之一有意一值可被傳送，而不會反向影響該結果。同樣的，如第32B圖中所示，介面I1由介面I1'取代，重新定義以忽略或新增參數至該介面。介面I2可類似重新定義為介面I2'，重新定義以忽略不必要的參數，或可在別處處理之參數。在此之重點在於在某些情況中，一程式設計介面可包含不為某種目的所需之方面，例如參數，且因此其可被忽略或重新定義，或為其他目的在別處處理。

內部編碼。合併兩個分別程式碼模組之一些或所有功能亦可行，使得其間之「介面」變更形式。舉例來說，第30A與第30B圖之功能可分別轉換成第33A與第33B圖之功能。在第33A圖中，之前第30A圖之第1與第2程式碼片段合併至包含兩者之一模組中。在此情況中，該程式碼片段仍可互相通訊，但該介面可調適為一形式，更適用於單一模組。因此，舉例來說，正式呼叫與傳回陳述式可能不再必要，但依據介面介面1之類似處理或回應可能仍有效。同樣的，顯示於第33B圖中，來自第30B圖之介面I2之部分(或全部)可於內部寫入於介面I1中，以形成介面

I1''。如圖所示，介面 I2 分成 I2a 與 I2b，且介面部分 I2a 以介面 I1 內部編碼，以形成介面 I1''。對於一具體範例來說，考量來自第 30B 圖之介面 I1 執行一函數呼叫 square(input,output)，其由介面 I2 接收，其在由該第二程式碼片段處理以輸入(加以平方)傳送之值之後，會以輸出傳回已平方之結果。在此一情況中，由該第二程式碼片段(平方輸入)執行之處理可由該第一程式碼片段執行，而不需至該介面之一呼叫。

分離(divorce)。來自一程式碼片段至另一程式碼片段之一通訊可間接藉由將該通訊分成多個不連續的(discrete)通訊加以完成。此以架構方式描述於第 34A 與第 34B 圖中。如第 34A 圖中所示，提供中間媒介(分離介面，由於其分離來自該原始介面之功能及/或介面辜能)之一或多個片段，以將該第一介面(介面 1)上之通訊轉至使其與一不同介面一致，在此情況中為介面介面 2A、介面 2B 與介面 2C。此可能完成於，如有一已安裝基本應用程式設計為根據一介面 1 通訊協定與一作業系統通訊，但接著該作業系統變更為使用一不同介面，在此情況中為介面介面 2A、介面 2B 與介面 2C。重點是由該第 2 程式碼片段使用之原始介面變更，使得其不再與由該第 1 程式碼片段使用之介面相容，且因此一媒介物用於使該舊與新介面相容。同樣的，如第 34B 圖中所示，一第三程式碼片段可以分離介面 DI1 引入，以接收來自介面 I1 且與分離介面 DI2 之通訊，以傳送該介面功能至重新設計以與 DI2 運作之介面 I2a 與 I2b，

但提供相同之功能結果。同樣的，DI1 與 DI2 可共同運作，以翻譯第 30B 圖之介面 I1 與 I2 之功能至一新作業系統，而提供相同或相似之功能結果。

重寫。又另一可能的變化係動態重寫該程式碼，以用其他者取代該介面功能，但達成相同之全面結果。舉例來說，可能有一系統，其中以一中間語言(如 Microsoft IL、Java ByteCode 等等)呈現之一程式碼片段提供至一執行環境(例如由該.Net 架構、該 Java 執行時間環境、或其他類似執行時間類型環境所提供者)中之一即時(JIT)編譯器或轉譯器。該 JIT 編譯器可被寫入，以動態轉換來自該第 1 程式碼片段至該第 2 程式碼片段之通訊，即將其與可能由該第 2 程式碼片段(該原始或一不同第 2 程式碼片段)所要求之一不同介面一致。此描述於第 35A 與第 35B 圖中。如可於第 35A 圖中所見，此方法類似於上數之分離情況。其可完成於，如一已安裝基本應用程式設計以根據一介面 1 通訊協定與一作業系統通訊，但接著該作業系統變更以使用一不同介面處。該 JIT 編譯器可用於半途一致化來自該已安裝型應用程式至該作業系統之新介面之通訊。如第 35B 圖中所述，此動態重寫該介面之方法可應用至動態因數化，或同時改變該介面。

亦應注意上述經由其他實施例做為一介面達成相同或類似結果之情況亦可以各種不同方式結合，串聯及/或並聯，或與其他介入程式碼結合。因此，上方呈現之其他實施例不互斥，且可混合、相符與合併，以產生與第 30A 與

第 30B 圖中所呈現之通用情況相同或相等之情況。亦應注意當以大部分程式設計建構時，有其他類似方法達成一介面之相同或相似功能，其可能不在此描述，但然而由本發明之精神與範圍所呈現，即應注意其至少部分功能由一介面之值下之一介面呈現，且其有利結果。

III. 擴充與繼承

本發明之基礎概念係對一特定擴充利用已由一架構所描述且由該硬體/軟體介面系統所強化之複雜結構、行為與操作模型化真實世界應用程式物件之項目。為了提供豐富子類型化功能，且在本發明之各種不同實施例中，一硬體/軟體介面系統(為了方便，我們在此後應簡稱為「WinFS」)可提供項目(與項目類型)可利用「擴充」擴充之一機制。擴充提供額外資料結構(屬性、關係等等)至目前存在之項目類型結構。

如之前在此所討論者(且由其在第 II.E.6.(a)與 II.F.3 節中所討論者)，該儲存平台用於提供有一初始架構組，在至少某些實施例中，該儲存平台允許顧客，包含獨立軟體廠商(ISV)，建立新架構(即新項目與巢狀元件類型)。由於由該初始儲存平台架構組所定義之一項目類型或巢狀元件類型可能不真正符合一 ISV 應用程式之需要，必須允許 ISV 以自訂該類型。此以擴充之概念加以允許。擴充係強烈類型例，但(a)其無法獨立存在且(b)其必須附加於一項目或巢狀元件。同樣的，除了點出架構可擴充性之需要，擴充亦用於點出「多類型」問題。在某些實施例中，由於該

儲存平台可能不支援多繼承或重疊子類型，應用程式可使用擴充做為模型化重複類型例之一方法(如一文件可為一「合法文件」以及一「安全文件」)。

A. 類型系統

在本發明之各種不同實施例中，該 WinFS 類型系統提供用以定義資料結構之一機制。該類型系統用於呈現儲存於 WinFS 中之資料。一 WinFS 類型宣告於一 WinFS 架構中。一 WinFS 架構定義一名稱領域，其為一組類型與其他 WinFS 架構元件做為一邏輯群組。WinFS 架構可利用可能使用一 XML 格式之一 WinFS 架構定義語言(SDL)加以宣告。下列係一可能架構宣告之一範例。

```
<Schema Namespace="System.Storage" >
    Type definitions
</Schema>
```

WinFS 架構亦為類型版本化做為一單元。WinFS 定義啟動該系統之數個系統架構。有包含該系統中根類型之類型宣告之 System.Storage 架構名稱領域，及宣告該系統中所有原始純量類型之 System.Storage.WinFS 架構名稱領域。

該 WinFS 類型系統宣告一組簡單純量類型。這些類型為該 WinFS 類型系統中之所有其他類型用做為大部分原始建立方塊。這些類型宣告於該架構名稱領域 System.Storage.WinFS 中。下表定義該原始類型組：

WinFS	已管理	CLR 類型	說明

類型	SQL 類型		
字串	SqlString	字串	變數—具有一最大長度為 2^31 字元長度之長度統一碼資料。長度可固定自 1-4000 字元，或利用該「max」關鍵字不加以限制。
二進位	SqlBinary	Byte[]	變數—具有 2^32 位元組長度之一最大長度之長度二進位資料。長度可固定自 1-8000 位元組，或利用「max」關鍵字不加以限制。
布林值	SqlBoolean	布林值	可為 Null 的布林值
位元組	SqlByte	位元組	單一無正負位元組
Int16	SqlInt16	Int16	自 -2^15(-32,768) 至 2^15-1(32,767) 之整數資料。
Int32	SqlInt32	Int32	自 -2^31(-2,147,483,648) 至 2^31-1(2,147,483,647)

			之整數(全數字)資料。
Int64	SqlInt64	Int64	自 -2^63(-9223372036854775808) 至 2^63-1(9223372036854775807)之整數(全數字)資料。
單	SqlSingle	單	自 -1.79E+308 至 1.79E+308 之浮點數資料
雙	SqlDouble	雙	自 -3.40E+38 至 3.40E+38 之浮點數資料
十進位	SqlDecimal	十進位	SQLDecimal 具有比 CLR 十進位類型較大之值範圍。 在儲存中，精度總是 28 個數字且純量為 0。
日期時間	SqlDateTime	日期時間	自 1753 年 1 月 1 日至 9999 年 12 月 31 日之日期與時間，具有三百分之一秒或 3.33 釐秒之一精確度。
Guid	SqlGuid	Guid	全域唯一識別符(GUID)。

Xml	SqlXmlReader	XmlReader	對里程碑B來說，WinFS.Xml會映射至「字串」類型。真正的XML資料類型支援預期於里程碑C中。
流	TBD	TBD	為有效存取使用檔案流回儲存之一二進位資料類型。此類型會於里程碑C中支援。

一WinFS列舉係宣告稱為一值清單之一組已命名常數之一純量類型。一列舉類型可使用於可使用一純量類型之任何地方。在此為一列舉宣告之一範例：

```
<Enumeration Name="Gender" >
  <Value Name="Male" />
  <Value Name="Female" />
</Enumeration>
```

該列舉之值以零為基礎。在上述範例中，Gender.Male呈現值0，而Gender.Female呈現值1。

一複雜類型由一名稱與一組屬性加以定義。一屬性係該類型之一成員領域，且由一名稱與一類型加以定義。一屬性之類型可為純量(包含列舉類型)或其他複雜類型。一WinFS類型可使用為稱為一巢狀類型之一屬性類型。一巢狀類型之一範例僅可存在為一複雜WinFS類型之一屬性值—該範例係巢狀於一複雜類型之一範例中。一巢狀類型係利用該巢狀類型架構元件加以宣告。在此為有效類型宣告

之一些範例：

```
        <NestedType Name="Address"
BaseType="System.Storage.NestedObject">
          <Property Name="Street" Type="WinFS.String"
Size="256"
        Nullable="false" />
          <Property Name="City" Type=" WinFS.String"
Size="256"
        Nullable="false" />
          <Property Name="State" Type=" WinFS.String"
Size="256"
        Nullable="false" />
          <Property Name="Country" Type="WinFSString"
Size="256"
        Nullable="false" />
        </NestedType>

        <ItemType Name="Person"
BaseType="System.Storage.Item">
          <Property Name="Name" Type="WinFS.String"
Size="256"
        Nullable="false" />
          <Property Name="Age" Type="WinFS.Int32"
        Nullable="false" Default="1"/>
          <Property Name="Picture" Type="WinFS.Binary"
Size="max"/>
          <Property Name="Addresses" Type="MultiSet"
MultiSetOfType="Address"/>
        </ItemType>
```

對於類型為字串與二進位之屬性來說，必須指定一大小屬性。此屬性指定包含於該屬性中之值之最大大小。一屬性可選擇性利用該可為空之屬性宣告一可為空之限制。一值「false」為此屬性指定當建立該類型之一範例時，該應用程式必須提供一值。另一選擇性屬性係為該屬性指定該預設值之預設屬性。若該應用程式不提供，此值會於範例建立時指派給該屬性。

上述範例中之地址屬性係為類型 MultiSet。類型

MultiSet 之一屬性亦稱為一多值屬性。在該範例中，該 MultiSet 包含一組類型位址之範例。一 MultiSet 類似於一集合。其可包含一複雜類型之零或多個範例。該 MultiSet 中範例之類型必須為一複雜巢狀類型。MultiSet 類型不支援 WinFS 純量類型之範例(包含列舉類型)。類型 MultiSet 之屬性無法為空，且無法具有一預設值。

WinFS 支援單繼承類型。WinFS 中之所有類型必須自一且僅於 WinFS 類型上繼承。該繼承類型稱為該取出類型，且此類型取出之類型稱為該基本類型。該 WinFS 之 BaseType 屬性中之基本類型類型化宣告元件。假設類型 A 自基本類型 B 取出，基本類型 B 接著自類型 C 取出。該類型 C 係該類型 A 與 B 之祖先類型。該類型 A 係該類型 B 與 C 之一後代類型。儲存於 WinFS 中之一資料範例永遠為單一類型之一範例。然而，我們可將資料範例視為包含該類型與所有其祖先類型之一組類型之一範例。對於係此一組類型之一範例之一資料範例來說，我們稱並非該組中任何其他類型之祖先之類型為最取出類型。單類型資料範例係一最取出類型之一範例。一般來說，我們較偏好一單類型元件之最取出類型作為其類型。該取出類型繼承宣告於其基本類型中之所有屬性。該取出類型可宣告新屬性，但無法覆寫定義於該基本類型中之屬性。宣告於該取出類型中之一屬性必須不使用與該基本類型之一屬性之相同名稱。

該資料模型中繼承之主要優點來自已繼承類型之可取

代性。考慮下列範例：

```
<NestedType Name="Name"
BaseType="System.Storage.NestedObject" >
        <Property Name="FirstName"
Type="WinFS.String" />
        <Property Name="LastName"
Type="WinFS.String" />
      </Nestedype>

      <NestedType Name="NameWithMiddleInitial"
BaseType="Name" >
        <Property Name="MiddleInitial"
Type="WinFS.String" />
      </NestedType>

      <NestedType Name="Person"
BaseType="System.Storage.Item" >
        <Property Name="RealName" Type="Name" />
        <Property Name="OtherNames" Type="MultiSet"
MultiSetOfType="Name" />
      </NestedType>
```

在上述範例中，類型個人具有一類型為 Name 之屬性 RealName，及係一組類型 Name 之一屬性 OtherNames。一般來說，會要求該屬性 RealName 僅具有其類型為 Name 之範例。然而，藉由繼承，單值範例允許 RealName 之值，只要該類型 Name 係該元件之最取出類型之祖先其中之一。所以，名稱 WithMiddleInitial 之一範例會允許該屬性 RealName 之值。

相同規則擴充至設定屬性。該屬性 OtherNames 包含一組元件。對於係該組之一成員之各單類型例來說，該範例之最取出類型必須具有 Name 作為其祖先其中之一。所以，該組 OtherNames 中某些範例可為類型 Name 之範例，而其他可為類型 NameWithMiddleInitial 之範例。

繼承亦允許方便查詢，其中於該 WinFS 系統尋找一特定類型之所有範例是可能的。當尋找一類型之所有範例時，該查詢引擎亦會傳回其最取出類型為此類型後代之所有範例。然而，這些操作僅為項目、擴充、及關係類型(並非屬性類型)加以支援。對巢狀類型(亦稱為巢狀元件、屬性、或複雜屬性類型)來說，該操作僅為單一多組屬性中包含之範例加以支援。

B. 類型家族

簡言之，該 WinFS 類型系統定義四個不同類型家族：

- 巢狀元件類型(亦稱為巢狀類型或屬性類型)
- 項目類型
- 關係類型
- 擴充類型

各類型家族具有一不同屬性組與使用於該 WinFS 類型系統中。該 System.Storage 架構名稱領域為各類型家族宣告做為根類型之四個類型。下列幾節詳細描述該類型家族。

1. 巢狀元件類型

不像其他 WinFS 類型家族，巢狀類型可使用為複雜 WinFS 類型之屬性之類型。一巢狀類型之範例僅可巢狀於另一範例類型之一範例中。然而，巢狀類型之範例無法全面查詢—也就是說，應用程式無法做出傳回該 WinFS 儲存中一給定巢狀類型之所有範例之一簡單查詢。

2. 項目類型

一 WinFS 項目係一類型之一範例，其祖先係該類型

System.Storage.Item。此類型係一複雜類型，其為該項目類型家族之根。System.Storage.Item 宣告類型 Guid 之名稱 ItemId 之一屬性。此為一項目之一特殊屬性，做為該項目之一主要金鑰。此屬性之值為一給定 WinFS 儲存中之項目保證為唯一值。此屬性係非不可為空，且當建立一項目類型之一範例時，必須由該應用程式加以指派。該 ItemId 屬性亦不可變—其可能永遠不會變更且必須不被重新使用。

該查詢引擎可傳回一 WinFS 儲存中一給定項目類型之範例。此查詢可傳回該類型之所有範例與其所有類型後代類型。稍後會描述該項目具有於該 WinFS 系統操作語義中之中央角色。

3. 關係類型

關係類型允許關係存在於項目之間。WinFS 關係類型描述二進位關係，其中一項目指定為該來源而另一項目指定為該目標。一關係係一類型之一範例，其祖先係該類型 System.Storage.Relationship。此類型係該關係類型階層之一根：

System.Storage.Relationship 類型宣告下列屬性：

- SourceItemId—係該關係範例之一來源之項目之 ItemId
- RelationshipId—關於該來源項目之關係之一為一識別符；該對(SourceItemId,RelationshipId)為 WinFS 中之關係類型形成主要金鑰。
- TargetItemId—該關係之目標之 ItemId；

- Mode—3個可能關係範例模式其中之一：持有、內嵌或參照
- Name—為持有關係包含該關係之名稱
- IsHidden—應用程式可選擇性使用以過濾不需要被顯示之關係之一布林值屬性

該SourceItemId、RelationshipId、TargetItemId與Mode屬性值為不可變。其於關係範例建立時被指派且無法被變更。

一關係類型宣告為具有下列額外限制之一範例類型：

- 來源與目標端點規格：各端點指定一名稱與該已參照項目之類型
- 該關係類型之範例之已允許模式：一關係範例無法為不於該關係宣告中允許之Mode屬性具有一值

這裡係一關係宣告之一範例：

```
<RelationshipType Name="DocumentAuthor"
BaseType="System.Storage.Relationship"
AllowsHolding="true"
AllowsEmbedding="false"
AllowsReference="true" >
  <Source Name="Document" Type="Core.Document"/>
  <Target Name="Author" Type="Core.Contact" />
  <Property Name="Role" Type="WinFS.String" />
  <Property Name="DisplayName" Type="WinFS.String" />
</RelationshipType>
```

該DocumentAuthor關係宣告有至持有或參照模式之範例限制。此意指該DocumentAuthor關係之一範例可具有值Mode="Reference"或Mode="Holding"之範例。具有值Mode="Embedding"之範例不被允許。

該關係宣告項目類型「Core.Document」之一來源端點名稱為「Document」及類型「Core.Contact」之一目的端點。該關係亦宣告二額外屬性。該關係範例與該項目個別地儲存與存取。所有關係類型例可自一全域擴充檢視加以存取。可做出會傳回一給定關係類型之所有範例之一查詢。

給定一項目，該項目係一來源之所有關係可根據該關係之 SourceItemId 屬性加以列舉。同樣的，為一給定項目來說，該項目係一目標之相同儲存中之所有關係可利用該關係之 TargetItemId 屬性加以列舉。

a) 關係語義

下列各節描述不同關係範例模式之語義：

持有關係：持有關係用於模型化該目標項目之參照計算型生命週期管理。一項目可為至項目之另或多個關係之一來源端點。並非為一內嵌項目之一項目可為一或多個持有關係之一目標。該目標項目必須與該關係範例於相同儲存中。

持有關係強化該目標端點之生命週期管理。一持有關係範例之建立及被設定目標之項目係一原子操作。額外持有關係範例可建立為設定目標於相同項目上。當具有一給定項目做為目標端點之最後持有關係範例被刪除時，該目標項目亦會被刪除。

當建立該關係之一範例時，指定於該關係宣告中之端點項目之類型會被強化。在建立該關係之後，該端點項目之類型無法被變更。

持有關係於形成該WinFS項目名稱領域中扮演一關鍵角色。所有持有關係參與於該名稱領域宣告中。該關係需告中之「名稱」領域定義關於該來源項目之目標項目之名稱。此相關名稱對源自一給定項目且無法為空的所有持有關係來說是唯一的。始自該根項目至一給定項目之相關名稱之排序清單形成該完整名稱至該項目。

該持有關係形成一導向非循環圖(DAG)。當一持有關係建立時，該系統確保一循環不會被建立，且因此確保該WinFS項目名稱領域形成一DAG。若要獲得該WinFS名稱領域與項目路徑之更多資訊，請參照該「WinFS名稱領域」規格。

內嵌關係：內嵌關係模型化該目標項目之生命週期之互斥管理之概念。其允許合成項目之概念。一內嵌關係範例之案例與被放置目標之項目係一原子操作。一項目可為零或多個內嵌關係之一來源。然而，一項目可為一且僅一內嵌安係之一目標。係內嵌關係之一目標之一項目無法為一持有關係之一目標。該目標項目必須於與該關係範例相同之儲存內。

當該關係之一範例建立時，於該關係宣告中指定之端點項目之類型會被強化。於該關係建立後，該端點項目之類型無法變更。該內嵌關係不參與於該WinFS名稱領域中。一內嵌關係之名稱屬性之值必須為空。

參照關係：參照關係不控制其參照之項目之生命週期。參照關係不保證該目標存在，也不保證該目標之類型

如同該關係宣告中所指定者。此意指該參照關係可搖擺。同樣的，該參照關係可參照其他 WinFS 儲存中之項目。

在 WinFS 中，參照關係會被用於模型化項目間最無生命週期之管理關係。由於該目標之存在並非強制，該參照關係便於模型化鬆耦合關係。該參照關係可用於將項目之目標放置於其他 WinFS 儲存中，包含其他機器上之儲存。該內嵌關係不參與於該 WinFS 名稱領域中。一內嵌關係之名稱屬性之值必須為空。

b) 關係規則與限制

下列額外規則與限制應用於關係：

- 一項目必須為(一內嵌關係)或(一或多個持有關係)之一目標。為一例外係該根項目。一項目可為一或多個參照關係之一目標。
- 為內嵌關係之一目標之一項目無法為持有關係之來源。其可為參照關係之一來源。
- 若已自檔案升級，一項目無法維持有關係之一來源。其可為內嵌關係與參照關係之一來源。
- 自一檔案升級之一項目無法為一內嵌關係之一目標。

當一關係類型 A 自一基本關係類型 B 取出時，應用下列規則：

- 關係類型 A 可另限制該端點類型。該端點類型必須為該基本關係 B 中對應端點類型之子類型。若該端點另加以限制，必須宣告該端點之一新名

稱。若該端點不另加以限制，該端點之規則係選擇性。

- 關係類型 A 可另限制宣告於該基本關係中已允許之範例模型。該已限制範例模式組必須為已允許範例類型之基本類型組之一子組。
- 該端點之名稱視為屬性名稱：其無法與一屬性隻一名稱或該類型或其雜本類型之一端點之一名稱相同。
- 若該對應端點類型不另由該取出關係加以限制，該來源與目標元件係選擇性。

在此為自定義如上之文件作者關係取出之一關係類型之一宣告之一範例：

```
<RelationshipType Name="LegalDocumentAuthor"
BaseType="Core.DocumentAuthor"
AllowsHolding="false"
AllowsEmbedding="false"
AllowsReference="true" >
  <Source Name="LegalDocument" Type="Legal.Document
"/>
  <Property Name="CaseNumber" Type="WinFS.String" />
</RelationshipType>
```

該 LegalDocumentAuthor 關係另限制該來源端點，但非該目標端點。該來源端點類型係自 Core.Document 取出之 Legal.Document。該目標端點在此情況中不另加以限制，因此該目標元件被省略。該關係另限制該已允許範例模式。其拒絕該持有模式，且唯一剩餘已允許模式係參照。

4. 擴充類型

一 WinFS 擴充係一類型之一範例，其祖先係該類型

System.Storage.Extension。此類型係一複雜類型，其為該擴充類型家族之根。

- System.Storage.Extension 定義二屬性：
- ItemId—與該擴充相關之 Item 之 ItemId。
- ExtensionId—關於該 ItemId 之擴充之唯一識別符。該對(ItemId,ExtensionId)唯一辨識一擴充例。

下列限制應用於擴充類型：

- 擴充類型例無法獨立存在於一項目之外。在該擴充類型例建立之前，具有與該擴充 ItemId 相同之 ItemId 之一項目類型例必須存在於該儲存中。若具有該給訂 ItemId 之項目不存在，該擴充無法被建立。當該項目被刪除時，所有具有相同 ItemID 之擴充也會被刪除。
- 一給定最取出擴充類型之至多一範例可與一個別項目相關。
- 擴充無法為關係之來源與目標。

沒有限制用於可與一給定項目類型相關之擴充類型上。任何擴充類型允許擴充任何項目類型。當多個不同擴充類型例附加於一項目上時，其結構與行為互相獨立。該擴充例個別地自該項目加以儲存與存取。所有擴充類型例可自一全面擴充檢視加以存取。可做出會傳回一給定擴充類型之所有範例之一查詢，不論其關於何類型。該擴充之 ItemId 指出其屬於何項目且可用於自該全面項目檢視取出

該對應項目物件。同樣的，給定一項目，與該項目相關之所有擴充例可利用該擴充之 ItemId 屬性加以列舉。

C. 增強功能

在本發明之數個實施例中，一硬體/軟體介面系統利用擴充與繼承，以正式化各種不同項目間之關係，且藉此增強查詢複數項目之能力。

1. 繼承

第 36 圖舉例說明一連串互相相關之項目與其關係之一子組。該文件項目 3602 與一接觸項目 3604 直接由一指定關係 3606 相關聯，在此情況中其係一「作者關係」—也就是說，該接觸 3604 係該文件 3602 之「作者」。在此範例中，該圖片項目 3622、該音樂項目 3624、及該特殊項目 3626 皆自該文件項目 3604 繼承，由於各項目之類型係該文件項目類型之一子類型。同樣的，該個人項目 3642 與該組織項目 3644 自該接觸項目 3604 繼承。在本發明數個實施例中，這些繼承項目(圖片 3622、音樂 3624、特殊 3626、個人 3642、及組織 3644)不僅繼承個別父項目(文件 3602 與接觸 3604)之屬性，但其亦繼承這兩個父項目間之指定關係。舉例來說，圖片 3622 繼承至接觸 3604 之一關係 3662、至個人 3642 之一關係 3664、及至組織 3644 之一關係 3666。一類似關係組亦由顯示之各其他項目加以繼承。

然而，注意到關係繼承並非自動，且不會發生於每個內容中是很重要的。舉例來說，描述一類型何時可繼承(即繼承控制)之屬性不可自我繼承。繼承參數由該硬體/軟體

介面系統加以維護與調整。

2. 擴充

第37A圖舉例說明應用程式特定目的之一項目之標準子類型化之缺點。在此圖中，一接觸可由四個應用程式APP1、APP2、APPX、及APPY加以存取。APP1與APP2存取該標準接觸，但APPX及APPY各需要一已擴充控制物件(加入額外欄位)，且因此取出接觸'與接觸''，其各自接觸加以繼承。然而，問題是現在有三個基本接觸項目之不同範例—一個於接觸中，一個於接觸'中，而一個於接觸''中。

舉例說明於第37B圖中對此問題之一部分解決方式係擴充該接觸屬性，以包含需要此之一應用程式所需之欄位。在此情況中，擴充接觸以包含APPX所需之額外欄位。然而，直接擴充一項目之欄位，例如接觸，僅可完成一次，且因此APPY無法應用此方法。

在本發明之一實施例中，一更複雜之解決方法係擴充具有不同與分離自控制本身之一擴充之接觸，如第37C圖舉例說明者。如此一來，APPX可擴充接觸，以包含其APPX額外欄位，而APPY亦可個別地擴充接觸，以包含其APPY額外欄位。這些擴充接著可搜尋與可查詢本身，且因此這些擴充為該硬體/軟體介面系統允許多類型化之一形式。

IV. 結論

如同前述舉例說明者，本發明導向用以組織、搜尋、及共享資料之一儲存平台。本發明之儲存平台擴充與擴張

資料儲存之概念超出既有檔案系統與資料庫系統之外，且設計為所有類型資料加以儲存，包含結構化、非結構化、或半結構化資料，例如相關(表狀)資料、XML、及一新資料形式，稱為項目。透過其共用儲存基礎與架構化資料，本發明之儲存平台允許為消費者、知識工作者及專家更有效之應用程式研發。其提供一豐富且可擴充應用程式介面，不僅使於其資料模型中繼承之功能有效，亦囊括與擴充既有檔案系統與資料存取方法。應了解在布超出廣泛發明概念之外，可對上述實施例做出變更。因此，本發明並不限於所揭示之特定實施例，但用於涵蓋於所附申請專利範圍中所定義知本發明精神與範圍之所有修改。

從上述中很明顯看到，本發明之各種不同系統、方法、及方面之所有或部分可實施於程式碼(即指令)之形式中。此程式碼可儲存於一電腦可讀取媒體上，例如一磁性、電性、或光學儲存媒體，包含但不限於一軟碟匣、CD-ROM、CD-RW、DVD-ROM、DVD-RAM、磁帶、快閃記憶體、硬碟裝置、或任何其他機器可讀取儲存媒體，其中，當該程式碼載入一機器且由該機器執行時，例如一電腦或伺服器，該機器變成用以實施本發明之一裝置。本發明亦可實施於程式碼之行式中，其透過某種傳輸媒體加以傳輸，例如透過電線或電纜、透過光纖、透過一網路，包含該網際網路或一內部網路，或經由任何其他傳輸形式，其中，當該程式碼由一機器接收與載入與執行時，例如一電腦，該機器變成實施本發明之一裝置。當實施於一通用目的處理

器時，該程式碼結合該處理器提供唯一裝置，操作類似於特定邏輯電路。

【圖式簡單說明】

前述簡介以及前述本發明實施方式在結合附圖閱讀時較容易了解。為了舉例說明本發明之目的，顯示於本發明各方面之範例實施例之圖式中；然而，本發明並不限於所揭示之特定方法與手段。在圖式中：

第 1 圖係呈現本發明之各方面可併入之一電腦系統之一方塊圖；

第 2 圖係舉例說明分成三個元件群組之一電腦系統之一方塊圖：該硬體元件、該硬體/染體介面系統元件、及該應用程式元件；

第 2A 圖為一檔案型作業系統中一目錄中之資料夾中群組化之檔案舉例說明傳統樹狀階層結構；

第 3 圖係舉例說明一儲存平台之一方塊圖；

第 4 圖舉例說明項目、項目資料夾、及類目間之結構關係；

第 5A 圖係舉例說明一項目之結構之一方塊圖；

第 5B 圖係舉例說明第 5A 圖之項目之複雜屬性類別之一方塊圖；

第 5C 圖係舉例說明該「位置」項目之一方塊圖，其中另描述其複雜類型(明確列示)；

第 6A 圖舉例說明一項目做為該基本架構中發現之項

目之一子類型；

第 6B 圖係舉例說明第 6A 圖之子類型項目之一方塊圖，其中明確列示其繼承類型(除了其立即屬性之外)；

第 7 圖係舉例說明包含二項階類別類型，項目及屬性基礎，及自其取出之額外基本架構類型之基本架構之一方塊圖；

第 8A 圖係舉例說明該核心架構中之項目之一方塊圖；

第 8B 圖係舉例說明該核心架構中之屬性類別之一方塊圖；

第 9 圖係舉例說明一項目資料夾、其成員項目、及該項目資料夾與其成員項目間之互連關係之一方塊圖；

第 10 圖係舉例說明一類目(同樣的，其係一項目本身)、其成員項目、及該類目與其成員項目間之互連關係之一方塊圖；

第 11 圖係舉例說明該儲存平台之資料模型之一參照類型階層之一示意圖；

第 12 圖係舉例說明關係如何被分類之一示意圖；

第 13 圖係舉例說明一通知機制之一示意圖；

第 14 圖係舉例說明一範例之一示意圖，其中二交易皆插入一新紀錄至該相同 B 樹中；

第 15 圖舉例說明一資料變更偵測程序；

第 16 圖舉例說明一範例目錄樹；

第 17 圖顯示一範例，其中一目錄型檔案系統之一既有資料夾被移入該儲存平台資料儲存中；

第 18 圖舉例說明包含資料夾之概念；

第 19 圖舉例說明該儲存平台 API 之基本架構；

第 20 圖有架構的呈現該儲存平台 API 堆疊之各種不同元件；

第 21A 圖係一範例接觸項目架構之一圖示呈現方式；

第 21B 圖係為第 21A 圖之範例接觸項目架構之元件之圖示呈現方式；

第 22 圖舉例說明該儲存平台 API 之執行時架構；

第 23 圖舉例說明一「尋找全部」動作之執行；

第 24 圖舉例說明儲存平台 API 類別自該儲存平台架構產生之程序；

第 25 圖舉例說明一檔案以其為基礎之一架構；

第 26 圖係舉例說明用於資料安全性目的之一存取遮罩格式之一示意圖；

第 27 圖(部份 a、b、及 c)描述開闢一既有安全性區域之一新相同保護安全性區域；

第 28 圖係舉例說明一項目搜尋檢視之概念之一示意圖；

第 29 圖係舉例說明一範例項目階層之一示意圖；

第 30A 圖舉例說明一介面介面 1 做為一導管，第一與第二程式碼片段透過其加以通訊；

第 30B 圖舉例說明一介面包含介面物件 I1 與 I2，其允許一系統之第一與第二程式碼片段經由媒體 M 加以通訊；

第31A圖舉例說明介面介面1提供之功能如何被子區分以將該介面之通訊轉換成多個介面介面1A、介面1B、介面1C；

第31B圖舉例說明介面I1提供之功能如何被子區分成多個介面I1a、I1b、I1c；

第32A圖舉例說明一情況，其中可忽略一無意義參數精確性或可亦一任意參數加以取代；

第32B圖舉例說明一情況，其中一介面可由一替代介面加以取代，其定義為忽略或新增參數至一介面中；

第33A圖舉例說明一情況，其中一第1與第2程式碼片段合併至包含兩者之一模組中；

第33B圖舉例說明一情況，其中部份所所有一介面可於內部寫入另一介面中，以形成一合併介面；

第34A圖舉例說明一或多個中間片段可如何將該第一介面上之通訊轉換以使其符合一或多個不同介面；

第34B圖舉例說明一程式碼片段可如何以一介面引入，以接收來自一介面之通訊，但傳輸該功能至該第二與第三介面；

第35A圖舉例說明一即時編譯器(JIT)可如何自一程式碼片段之通訊轉換至另一程式碼片段；

第35B舉例說明動態重新撰寫一或多個介面之一JIT方法可應用至動態因素或改變該介面；

第36圖舉例說明一連串內連項目與其關係之一子組；

第37A圖為應用程式特定目的舉例說明一項目之標準

子類型之缺點；

第37B圖舉例說明標準子類型之問題之一部份解決方法；及

第37C圖舉例說明本發明之一實施例，以獨特且與接觸本身不同之一擴充加以擴充一項目，且因此啟動一多類型功能。

【主要元件符號說明】

20	電腦	21	處理單元
22	系統記憶體	24	ROM
25	RAM	26	BIOS
27	硬碟	29	可移除儲存
30	光學裝置	31	可移除光碟
32	硬碟裝置介面	33	磁碟裝置介面
34	光學裝置介面	35	作業系統
36	應用程式	37	其他程式
38	程式資料	40	鍵盤
42	滑鼠	46	序列埠介面
47	螢幕	48	視訊轉接器
49	遠端電腦	50	軟碟
51	區域網路	52	廣域網路
53	網路介面		
54	數據機	55	主機轉接器
56	SCSI匯流排	62	儲存裝置

200 電腦系統
202 硬體元件
204 硬體/軟體介面系統元件
206 應用程式元件
212 DOS 型檔案系統基本資料夾
214 資料夾
216 額外資料夾
220 檔案
300 儲存平台
302 資料儲存
304 資料模型
306 變更追蹤
308 安全性
310 升級/降級
312 儲存應用程式介面
314 資料庫引擎
316 SQL 儲存
318 NTFS
320 應用程式介面
322 儲存平台應用程式介面
324 OLE DB
326 WIN 32 應用程式介面
328 服務
330 同步化
332 通知
334 資訊代理者
336 公用程式
338 遠端資料儲存
340 平台架構
342 已擴充平台架構
344 新 ISV 架構
346 架構配置工具
350a 應用程式
350b 應用程式
350c 應用程式
402 項目
404 項目
406 項目
408 項目
410 項目
412 項目
414 項目
416 項目
418 項目

420 項目

422 項目資料夾

424 項目資料夾

426 項目資料夾

428 項目資料夾

430 項目資料夾

432 類別

434 類別

436 類別

512 關係

524 關係

516 關係

526 關係

900 項目資料夾

902 項目

904 項目

906 項目

1000 類別

1002 項目

1004 項目

1006 項目

1008 共用屬性、值、或類型之組合

1008' 共用描述

1012 關係

1024 關係

1016 關係

1026 關係

1900 SQL 客戶端

2002 資料類別

2004 執行時架構

2006 執行時架構類別

2008 設計時工具

2010 XML 中之架構描述

2012 網路特定方法之程式碼

2014 架構

2016 UDT 類別定義

2202 項目內容

2204 項目物件圖

3602 文件

3604 接觸

3606 已指定關係

3622 圖片

3624 音樂

3626 特殊

3642 個人

3662 關係

3666 關係

3644 組織

3664 關係

五、中文發明摘要：

藉由模型化具有以由該硬體/軟體介面系統強制之一架構所描述之複雜結構、行為、及動作之真實世界應用程式物件，本發明之各種不同實施例藉由利用提供額外資料結構(屬性、關係等等)之「擴充(Extention)」擴充項目(及項目類型)提供豐富子類型功能至已存在的項目類型結構。擴充係無法獨立存在且必須附加於一項目或一巢狀元件上之強烈類型例 (strongly typed instances)。擴充亦藉由允許類型例重疊，來解決「多類型」問題(如一文件可為一「合法文件」以及一「安全文件」)。

六、英文發明摘要：

By modeling real-world application objects with complex structures, behaviors, and operations described by a schema which is enforced by the hardware/software interface system, various embodiments of the present invention provide rich sub-typing functionality by extending Items (and Item types) using "Extensions" which provide additional data structures (Properties, Relationships, etc.) to already existing Items type structures. Extensions are strongly typed instances that cannot exist independently and must be attached to an Item or a Nested Element. Extensions are also intended to address "multi-typing" issues by enabling the overlap of type instances (e.g., a Document may be a "legal document" as well a "secure document").

十、申請專利範圍：

1. 一種用以客製化一不連續的(discrete)可儲存資訊單元之方法，包含下列步驟：

定義一不連續的可儲存資訊單元，其具有一類型結構及一第一識別符；

定義一擴充類型(an extension type)，其表示一所欲額外資料結構；

定義該擴充類型之一擴充例，該擴充例係由該第一識別符及一擴充識別符進行識別，並儲存於一電腦可讀取儲存媒體中，可從該不連續的可儲存資訊單元個別存取；以及

產生一客製化之不連續的可儲存資訊單元，其儲存於一電腦可讀儲存媒體中，且可從該不連續的可儲存資訊單元個別存取，其中產生該客製化之不連續的可儲存資訊單元之步驟，係包含將該擴充類型之該擴充例附加至該不連續的可儲存資訊單元。

2. 如申請專利範圍第1項所述之方法，其中上述擴充例無法獨立於該客製化之不連續的可儲存資訊單元的該類型結構而存在。

3. 如申請專利範圍第1項所述之方法，更包括下列步驟

定義複數擴充，其中每一擴充表示一所欲額外資料結構；以及

附加該些擴充至該不連續的可儲存資訊單元的該

類型結構。

4. 如申請專利範圍第3項所述之方法，其中上述複數擴充係用於模型化多個重疊類型例(model overlapping type instances)。

5. 一種用以擴充一屬性（Property）之方法，上述屬性構成一可由一硬體/軟體介面系統操控之複雜屬性類型，上述方法包含利用一強烈類型例(一「擴充」)擴充該屬性，上述擴充構成一不連續的可儲存資訊單元，其可由該硬體/軟體介面系統操控且與該屬性相關。

6. 如申請專利範圍第5項所述之方法，其中上述擴充係附加於該屬性上。

7. 如申請專利範圍第5項所述之方法，其中上述擴充無法獨立存在於該屬性之外，使得若上述屬性停止存在，該擴充亦停止存在。

8. 如申請專利範圍第5項所述之方法，其中上述屬性係由複數擴充加以擴充。

9. 如申請專利範圍第8項所述之方法，其中上述複數擴充係用於模型化多個重疊類型例。

10. 一種為一硬體/軟體介面系統來組織及有效查詢複數項目之方法，上述項目構成可由一硬體/軟體介面系統操控之多個不連續的可儲存資訊單元，上述複數項目包含與一第一項目與一第二項目相關之一第一關係，上述方法包含：為將一第三項目的例示化(instantiation)，上述

第三項目係該第一項目之一子類型例，上述第三項目自動自該第一項目繼承一與該第二項目之關係。

11.如申請專利範圍第 10 項所述之方法，其中為將一第四項目例化，上述第四項目係該第二項目之一子類型例，上述第四項目自動自該第二項目繼承一與該第一項目之關係。

12.如申請專利範圍第 11 項所述之方法，其中上述第四項目另自動自該第二項目繼承一與該第三項目之關係。

13.如申請專利範圍第 10 項所述之方法，其中為該第一項目之一第一複數子類型例之各者，上述第一複數子類型例之各者自動自該第一項目繼承一與該第二項目之關係。

14.如申請專利範圍第 13 項所述之方法，其中為該第二項目之一第二複數子類型例之各者，上述第二複數子類型例之各者另自動自該第二項目繼承與該第一項目之多個關係。

15.如申請專利範圍第 14 項所述之方法，其中上述第一複數子類型例之各者自動自該第一項目繼承一與該第二複數子類型例之各者之關係。

16.如申請專利範圍第 15 項所述之方法，其中上述第二複數子類型例之各者自動自該第二項目繼承一與該第一複數子類型例之各者之關係。

17.一種用以操控複數不連續的可儲存資訊單元之硬體/軟

體介面系統，上述系統包含：

一處理器，其經組態為執行處理器可執行指令；

與該處理器通訊之一記憶體，該記憶體儲存該處理器可執行指令；

一子系統，其常駐於該記憶體中，且包含用以執行下列步驟之處理器可執行指令：

定義多個不連續的可儲存資訊單元，其具有一類型結構及各自之第一識別符；

定義至少一擴充類型，其表示所欲額外資料結構；

定義該至少一擴充類型的至少一擴充例，該至少一擴充例係由該至少一擴充例所附加至之一不連續的可儲存資訊單元的一個別第一識符別及一個別擴充識別符來進行識別，該至少一擴充例係儲存於一電腦可讀取儲存媒體中，且可從該等不連續的可儲存資訊單元個別存取；以及

產生多個客製化之不連續的可儲存資訊單元，其儲存於該電腦可讀取儲存媒體中，且可從該等不連續的可儲存資訊單元個別存取，其中產生該等客製化之不連續的可儲存資訊單元之步驟，係包含將該擴充類型之該擴充例附加至該等不連續的可儲存資訊單元。

18.如申請專利範圍第17項所述之硬體/軟體介面系統，其中該至少一擴充例無法單獨存在於該至少一擴充例所附加至之該等客製化之不連續的可儲存資訊單元的該

類型結構之外。

19.如申請專利範圍第17項所述之硬體/軟體介面系統，該系統更包含一子系統，該子系統常駐於該記憶體中，且包含用以執行下列步驟之處理器可執行指令：

定義複數擴充，其中每一擴充表示一所欲額外資料結構；以及

附加該等擴充至該等不連續的可儲存資訊單元的該類型結構。

20.一種用以操控複數屬性（Properties）之硬體/軟體介面系統，上述屬性構成可由一硬體/軟體介面系統操控之多個複雜屬性類型，上述系統包含一子系統，用以藉由一強烈類型例(一「擴充」)來擴充一屬性，上述擴充構成可由該硬體/軟體介面系統操控之一不連續的可儲存資訊單元。

21.如申請專利範圍第20項所述之系統，其中上述擴充係附加於該屬性上。

22.如申請專利範圍第20項所述之系統，其中上述擴充無法獨立存在於該屬性之外，使得若該屬性停止存在，該擴充亦停止存在。

23.如申請專利範圍第20項所述之系統，其中上述屬性係由複數擴充加以擴充。

24.一種用以操控複數項目之硬體/軟體介面系統，其中一項目構成可由該硬體/軟體介面系統操控之一不連續的

可儲存資訊單元，上述系統包含一子系統，用以組織與有效查詢該複數項目，上述複數項目包含與一第一項目及一第二項目相關之一第一關係，其中上述子系統：

為將一第三項目例示化（instantiation），該第三項目係該第一項目之一子類型例，自動在該第三項目與該第二項目間建立一關係，

為將一第四項目例示化，該第四項目係該第二項目之一子類型例，自動在該第四項目與該第一項目間建立一關係；及

自動在該第四項目與該第一項目間建立一關係。

25. 如申請專利範圍第 24 項所述之系統，其中為該第一項目之一第一複數子類型例之各者，且為該第二項目之一第二複數子類型例之各者，該子系統：

自動為該第一複數子類型例之各者與該第二項目建立一關係；

自動為該第二複數子類型例之各者與該第一項目建立一關係；及

自動為該第一複數子類型例之各者與該第二複數子類型例之各者建立一關係。

26. 一種用以操控複數項目之硬體/軟體介面系統，其中一項目構成一可由該硬體/軟體介面系統操控之不連續的可儲存資訊單元，上述系統包含一子系統，用以藉由一強烈類型例(一「擴充」)來擴充一項目，上述擴充構成

一可由該硬體/軟體介面系統操控之不連續的可儲存資訊單元。

27.如申請專利範圍第26項所述之系統，其中上述擴充係附加於該項目上。

28.如申請專利範圍第26項所述之系統，其中上述擴充無法單獨存在於該項目之外，使得若該項目停止存在，該擴充亦停止存在。

29.如申請專利範圍第26項所述之系統，其中上述項目係由複數擴充加以擴充。

30.一種包含電腦可讀取指令之電腦可讀取媒體，該等電腦可讀取指令經組態為執行一用以客製化一不連續的可儲存資訊單元之方法，該方法包含下列步驟：

定義一不連續的可儲存資訊單元，其具有一類型結構及一第一識別符；

定義一擴充類型，其表示一所欲額外資料結構；

定義該擴充類型之一擴充例，該擴充例係由該第一識別符及一擴充識別符進行識別，並儲存於一電腦可讀儲存媒體中，可從該不連續的可儲存資訊單元個別存取；以及

產生一客製化之不連續的可儲存資訊單元，其儲存於一額外電腦可讀儲存媒體中，且可從該不連續的可儲存資訊單元個別存取，其中產生該客製化之不連續的可儲存資訊單元之步驟，係包含將該擴充類型之該擴充例

附加至該不連續的可儲存資訊單元。

31.一種包含用以擴充一屬性（Property）之電腦可讀取指令之電腦可讀取媒體，該屬性構成可由一硬體/軟體介面系統操控之一複雜屬性類型，上述電腦可讀取指令包含用以利用一強烈類型例(一「擴充」)來擴充該屬性的多項指令，該擴充構成可由該硬體/軟體介面系統操控之一不連續的可儲存資訊單元，其中上述擴充係附加於該屬性，且其中當上述屬性停止存在時，該擴充亦停止存在。

32.一種包含用以組織與有效查詢複數項目之電腦可讀取指令之電腦可讀取媒體，該項目構成可由一硬體/軟體介面系統操控之不連續的可儲存資訊單元，上述電腦可讀取指令包含指令用以：

將一第一項目、一第二項目、及與一第一項目與一第二項目有關之一第一關係例示化（instantiation）；

將一第三項目例示化，上述第三項目係該第一項目之一子類型例；及

自動在該第三項目與該第二項目間建立一已繼承關係。

33.如申請專利範圍第32項所述之電腦可讀取媒體，另包含指令以：

將一第四項目例示化，上述第四項目係該第二項目

之一子類型例；及

自動在該第四項目與該第一項目間建立一已繼承關係。

34.如申請專利範圍第33項所述之電腦可讀取媒體，另包含指令，以自動建立該第三項目與該第四項目間之一已繼承關係。

七、指定代表圖：

(一)、本案指定代表圖為：第 36 圖。

(二)、本代表圖之元件代表符號簡單說明：

3602　文件

3604　接觸

3606　已指定關係

3622　圖片

3624　音樂

3626　特殊

3642　個人

3644　組織

3662　關係

3664　關係

3666　關係

八、本案若有化學式時，請揭示最能顯示發明特徵的化學式：

無