應用資料庫代理程式於組織工作警訊之設計與實作

陳炳祥　　徐濟世
國立雲林科技大學資訊管理系
g9323211@yuntech.edu.tw、hsujs@mis.yuntech.edu.tw

摘要

提高組織人員的工作效率與增進人員之負責任態度，一直是管理者所關切的議題。若流程中的上下游部門間工作進行無法如期地順利銜接，部門間的爭吵及推卸責任的情景將不斷地重複上演。然而導入一套新資訊系統來管理工作流程事務，對企業組織來講，將是一筆龐大的費用支出。因此，本研究著眼於企業組織成本費用的考量，以不額外增加資訊系統費用之原則下，利用資料庫代理程式及 SQL Mail 的服務設定，來實作工作逾時之警訊通知。此一設計能有效結合組織內部的現有資訊系統，提供部門人員工作進度落後時自動警訊與進行後續追蹤的動作。

關鍵詞：工作流程、資料庫代理程式、SQL Mail、自動警訊。

Abstract

It has been the key issue of the manager to improve the work efficiency and responsibility of staff in organization all the time. If the interdepartmental work from upstream to downstream in the workflow can't be linked up smoothly on schedule, the situation of argument and shirking duty between each department will be occurred constantly. In organization, however, it would be a huge expense to introduce a new information system to manage the workflow affair.

The main consideration of this research focuses on the expenses of enterprises, without increasing the expenses of information system. We utilize database agent and SQL Mail service to implement alert notice that the work has not been finished on schedule. This design will combine with the existing information system effectively and provide automatic alert on work progress delay and go on follow-up tracking.

Keywords: Workflow, Database Agent, SQL Mail, Automatic Alert.

1. 前言

為了保持組織競爭力，許多企業經常透過採購新資訊系統或內部自行開發新系統的方式，來滿足組織之正常營運作業，以期達到節省人力、節約成本、提昇效率的目標。這些大型的資訊系統所提供的模組功能，一般是依組織內的工作流程來畫分其使用的部門及操作的人員。這些使用系統的部門彼此間也有相當程度的工作關聯，部門間的互動及溝通亦頗為密切。舉例來說，甲部門的工作任務是承續乙部門的工作而來，乙部門的工作任務是承續丙部門的工作而來。也就是說，甲部門人員之工作要視乙部門人員之工作進度而定，乙部門人員之工作要視丙部門人員之工作進度而定，依此類推。若是工作流程的上游部門進度稍微延遲，將連帶影響後續工作流程的下游部門工作進行。如何能確信員工如期完成其應做的事？對管理者而言，是極為重要的，這也是企業組織或政府單位期盼達成的目標。

因此在組織中，無不希望透過一個系統化的管理機制，正式的評估員工工作的績效。這些管理機制的設立，目的不在處罰員工，而是要能適時產生警示以提醒員工應盡的義務與責任。工作管理系統可以用來呈現有關於人員工作績效與回饋資訊[1]，以達到更好的溝通效率及更快速的反應，並以此協助管理者取得績效改善之建議與衡量組織人員之發展。為此許多企業必須投入額外的費用來導入與管理工作流程相關的資訊系統，像是 Lotus Notes 等知名軟體系統。但這些系統往往獨立運作，無法有效整合組織現行運作中的其他資訊系統，像是 ERP(Enterprise Resource Planning) 系統、PDM(Product Data Management)系統，亦不符合部門間實際例行性工作任務之需求，導致各部門人員必須在現行的營運系統及新導入的工作流程管理系統中輸入重覆的資料，不僅浪費人力工時，對於新導入的工作流程管理系統只採用部分的功能，如此更是浪費資訊系統的價值及導入費用。

綜合以上所言，本研究主要以不額外增加資訊系統費用為前提，直接結合組織內部營運中的資訊系統，將資料逾時未處理之判斷邏輯寫成預存程序，並利用資料庫代理程式服務定時由 SQL Mail 將查詢結果發送到電子郵件伺服器，最後傳送通知信件到相關部門人員的電子郵件信箱，以達到日常工作進度落後時自動警訊提醒與追蹤之功效。

2. 相關技術及文獻探討

本章節針對與本研究相關的技術及文獻進行探討整理，主要分為四大部分。第一部分為工作流程之定義，第二部分為 SQL Server Agent，第三部分為 Stored Procedure，第四部分為 SQL Mail。

2.1 工作流程之定義

工作流程是指一有組織的工作任務收集，用來完成企業的某些流程[5]。一般來說，一項工作任務的完成可以是一個或多個軟體系統、單一個人或一群人，或是這些軟體系統和人的結合所共同完成。因此，工作流程是由各種來源管道合併資訊，也就是在使用者與系統間轉移資訊[4]。典型的工作流程應用程式包括表單的傳閱/許可、文件的檢閱/發佈及問題追蹤。在企業組織中，各部門間的工作性質常常會有工作流程上的先後關係，組織流程中的上游部門工作完成後，接續其後的部門才能展開工作任務的進行。由此足以顯示出工作流程的順暢，必能促進企業內部資訊的流通，進而達到提升企業生產力及競爭力。

在實際的企業組織中，工作處理的過程會經由一個有次序性的流程來達成一項工作目標。在此一流程進行時，工作任務中所包含的資料與資訊將順著事先定義好的流程順序，依次傳送至下一流程的參與者來執行工作。藉由電腦資訊系統之程式設計，可將流程參與者間的資料、文件、資訊及工作程序予以設計規劃，來達成流程自動化之目標。

2.2 SQL Server Agent

SQL Server Agent(代理程式)主要負責設定 SQL Server 上的週期活動排程，能將伺服器已發生的問題告知系統管理員。SQL Server Agent 主要的功能為作業排程、執行作業(Job)、產生警示(Alert)、事件發生通知管理人員或操作員(Operator)。實作這些功能的 SQL Server 代理程式元件[3]為：

1. Operator(操作員)：設定可解決伺服器問題的管理人員及其通訊方式。當伺服器發生問題時，這些操作員可以透過電子郵件、呼叫器、net send 網路命令而成為警示的目標。
2. Job(作業)：由一或多個欲執行之步驟所構成的已定義物件，這些步驟是可執行的 T-SQL 陳述式。作業可以設定排程，執行排定於特定時間或週期區間所發生的 SQL Server 工作。作業可設定成自動及手動兩種方式。
3. Alert(警示)：針對發生特定狀況時要發信給操作員所採取的動作。其設定的方式可針對某特定事件，例如特定的錯誤、資料庫到達已定義的可用空間限制等，或是指定事件的嚴重性，例如某種嚴重性的錯誤。當警示被觸發時，其所要採取的動作可由管理者事先所定義，例如傳送電子郵件、呼叫操作員或執行作業來解決問題。

SQL Server 代理程式必須使用足夠權限的 Windows 網域帳號來啟動，而非使用 SQL Server 系統帳號[9]，如此 SQL Server 代理程式的元件才能正常運作。在 Microsoft SQL 中，作業、警示以及操作員可以透過 SQL Server Enterprise Manager、SQL 分散式管理物件(SQL-DMO)應用程式、Transact-SQL 應用程式等方式來指定。這些定義也將儲存於 SQL Server 的 msdb 系統資料庫中，以等待 SQLServerAgent 服務啟動時，可於此查詢及決定所要啟用的作業和警示。當資料庫 Server 有狀況發生時，將會傳遞事件給代理程式，代理程式也將會依作業排程的時間來執行作業或是執行警示、發送 SQL Mail 要求給 SQL Server，以及傳送 net send 網路命令給 Windows 或是藉助電子郵件來告知操作員。針對上述的作業、警示以及操作員之定義與動作所使用的主要元件，可由圖 1 來呈現。

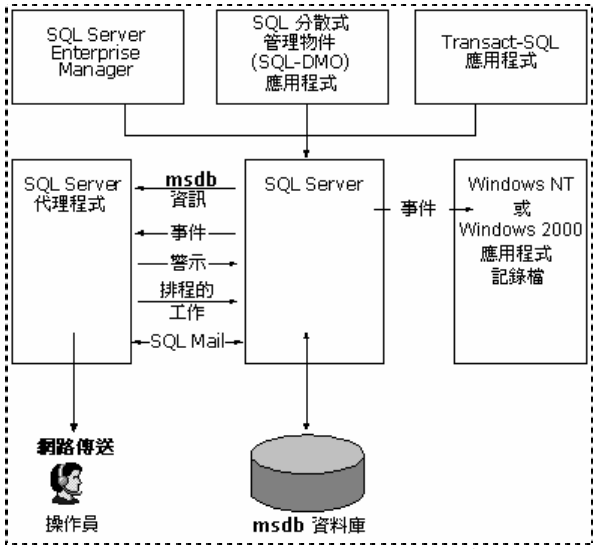

圖 1 作業、警示以及操作員之定義與動作所使用的主要元件
資料來源：Microsoft SQL Server 線上說明

2.3 Stored Procedure

Stored Procedure(預存程序)是一組為了完成特定功能的 T-SQL(Transact-SQL)控制命令語法，將一些寫好的 T-SQL 命令語法包裝成一個程序(Procedure)儲存在資料庫中，一般我們可以將經常需要處理的一些動作寫成預存程序，以方便往後之執行[2]。由於寫好的預存程序是儲存在 Database Server 上，將可加快存取資料庫的效率以及方便日後管理。一般來說，預存程序具有以下優點：(1)減少及避免應用程式之修改及重新部署、(2)實現較快的執行速度、(3)減少網路流量、(4)作為一種安全機制來充分利用。

如表 1 所示，即是完整的預存程序建立語法[6]，在此簡易說明如後。建立預存程序的敘述命令是以 CREATE PROCEDURE 或 CREATE PROC 開始[7]，其後接著預存程序的名稱，亦可在此程序名稱之前指定其擁有者(Owner)。選擇性整數值[;number]可用來建立一組預存程序，例如，spList;1、spList;2。另外也可為預存程序指定參數，包括輸入參數和輸出參數，這些指定參數的名稱各別是以@為前導字元，每個參數的後面需緊接著參數的資料型態，若

該參數是供傳回值用時，可於其後加上 OUTPUT 表示。使用 WITH ENCRYPTION 能對預存程序程式碼加密，使用 WITH RECOMPILE 則每次使用預存程序時會強迫重新編譯。另外在資料庫進行複寫(Replication)時，使用 FOR REPLICATION 來處理篩選。最後在 AS 關鍵字後面的內容則是該預存程序的主體部分，指定程序所要執行的動作，包括在程序中的任何 T-SQL 陳述式的數目或資料型態。

表 1 完整預存程序建立語法

```
CREATE PROC[ EDURE ] procedure_name [ ;
number ]
    [
       { @parameter data_type }
        [ VARYING ] [ = default ] [ OUTPUT ]
    ] [ ,...n ]

[ WITH
     { RECOMPILE
      | ENCRYPTION
      | RECOMPILE , ENCRYPTION }
]

[ FOR REPLICATION ]

AS
    sql_statement [ ...n ]
```

2.4 SQL Mail

SQL Mail 提供了一種從 SQL Server 發送和閱讀電子郵件的簡單方法，藉由電子郵件用戶端來收發電子郵件。由於 SQL Mail 是一個 MAPI(Mail Application Programming Interface)應用程式，因此伺服器上必須存在 MAPI 子系統[8]，例如 Microsoft Outlook 之類的 MAPI 客戶端。不同於 SQL Server Agent 郵件功能(SQLAgentMail)只能發信通知系統操作員，SQL Mail 除了可以發送電子郵件，亦可以在傳送中包含一個結果集(Result Set)，將內含查詢命令的電子郵件傳送給 SQL Server，由 SQL Server 將命令處理過後再透過 SQL Mail 以電子郵件傳回給使用者[2]。由上所述，SQL Mail 讓 SQL Server 可以與郵件伺服器建立用戶端連線，來傳送與接收電子郵件，且通常所使用的郵件伺服器有 Microsoft Exchange Server、Microsoft Windows NT 郵件或 Post Office Protocol 3 (POP3)等。

3. 系統架構及功能實作

如圖 2 之架構圖所示，本研究採用 Microsoft SQL Server 2000 作為資料庫伺服器軟體，配合 SQL Mail 服務之應用，並設定警訊處理規則，再藉由代理程式設定作業排程時間，自動執行警訊處理動作，最後將該動作處理結果發送到郵件伺服器，以達到警訊通知的功能。

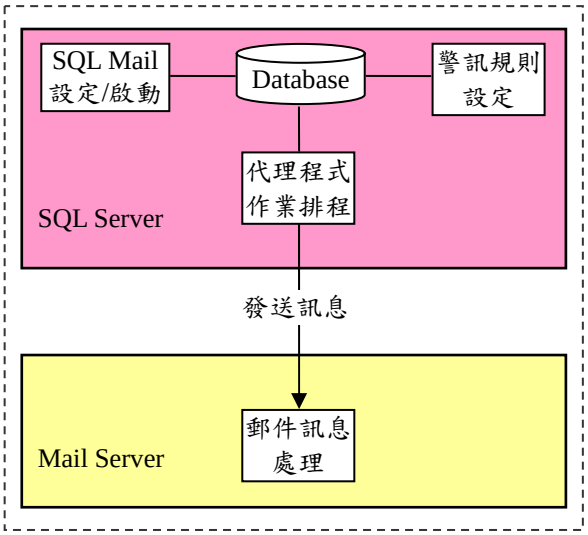

圖 2 系統架構圖

3.1 系統特色

本研究之警訊機制是架構在 Microsoft SQL Server 2000 及 Microsoft Exchange 2000 Server 上，此二部伺服器作業系統皆採用 Microsoft Windows 2000 Server。透過此二部伺服器來設計及實作本研究之工作警訊通知，主要由 SQL Server 之代理程式每日清晨來觸發自訂的預存程式，針對資料庫中逾期未處理之資料進行過濾與彙總後，再將所產生的結果透過 SQL Mail 傳送到 Exchange Server。此一運作模式如圖 3 所示，實現了以組織現有營運中的系統來自動發送工作警訊通知的功能。

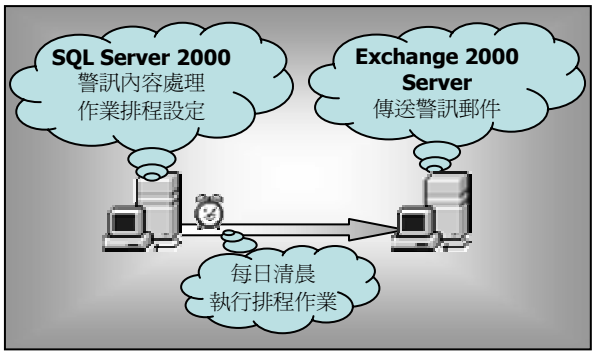

圖 3 警訊運作示意圖

3.2 SQL Mail 設定

Microsoft SQL Server 提供了 SQL Mail 服務，為了使用 SQL Mail，首先在實作的伺服器上必須要有 SMTP(Simple Mail Transfer Protocol)服務。由於 Microsoft Windows 2000 不提供 MAPI 子系統，因此，我們先行安裝了 Microsoft Outlook 作為 MAPI 客戶端。此外，SQL Mail 需要有郵局連線、郵件存放(信箱)、用戶端郵件設定檔(Mail Profile)與用來登入 SQL Server 執行個體(Instance)的 Windows 網域

使用者帳戶。在安裝和設定好客戶端郵件設定檔後，為使 SQL Mail 能取得經由電子郵件所傳送之查詢要求，尚須指定 SQL Mail 組態設定檔，如圖 4 所示。

圖 4 指定 SQL Mail 組態設定

3.3 警訊通知規則設計

我們的目標是要藉助 SQL Mail 寄出查詢命令給 SQL Server，讓 SQL Server 處理完命令後隨即將查詢結果寄給相關的部門人員，以提醒相關人員工作處理之進度。因此必須先給定警訊查詢的邏輯規則，將實際處理資料查詢的 SQL 語法寫成預存程序，供稍後 SQL Mail 發送查詢命令時使用，以加快執行速度及方便後續規則修改。警訊通知的查詢規則敘述如下：

1. 設定警訊通知所規範的工作逾期天數(以參數@Term 來指定)，同時規範通知 Team Leader(以參數@ CTLTerm 來指定)以及高層主管(以參數@ AdminTerm 來指定)之逾期天數，如表 2 所示。
2. 從企業組織現有的營運系統資料庫中，找出逾期未處理的資料(將查詢規則另寫成預存程序來處理)。
3. 找出工作逾期未處理的人員名單(以參數@Developer 來記錄)及其電子郵件(以參數@MyReceiver 來記錄)。
4. 找出工作逾期未處的人員所對應之 Team Leader 電子郵件(以參數@Mycopy_recipients 來記錄)。
5. 送出電子郵件要求 SQL Server 執行資料查詢彙總，並將查詢結果自動產生附檔通知相關人員。

表 2 警訊通知設計之部分參數設定

```
/* 宣告參數 */
DECLARE @AdminReceiver varchar(100) --高層主管收件者
DECLARE @Pno int
DECLARE @Developer varchar(400) --工作逾期未處理人員
DECLARE @MyReceiver varchar(255) --收件者
DECLARE @Mycopy_recipients varchar(255) --副本收件者
DECLARE @MySubject varchar(255) --郵件主旨

/* 指定參數值 */
SET @Computetime =getdate()
SET @Term=3 --設定工作逾期天數
SET @CTLTerm=9 --設定幾天後給 Team Leader
SET @AdminTerm=12 --設定幾天後給高層主管
```

透過上述規則設定後，只要在企業組織日常的營運系統中，工作逾期未處理之相關人員，每天將會收到由 SQL Server 所發出的警訊通知，直至該人員在營運系統中將其負責之工作任務完成為止。為避免該警訊通知過於頻繁，同一工作任務之負責人每天收到相同的訊息，不僅造成人員情緒低落、壓力變大，亦不人性。因此在警訊通知規則中，增加「每隔幾天提醒一次」的設定。若「本次警訊發生之日期」設為 AlertDate，「該工作上次之追蹤基準日期」設為 TraceDate，「警訊通知週期」設為 Cycle，則此新增的規則可以公式表示為：

$$(\text{AlertDate} - \text{TraceDate}) \% \text{Cycle} = 1 \quad \text{...公式(1)}$$

在公式(1)中的%表示兩數字相除後取餘數。有了此公式，我們便可以很容易計算出本次警訊發生時，應該將那些未完成的逾期資料列入警訊通知。假設本次警訊發生之日期(AlertDate)為 2005/6/6，警訊通知週期(Cycle)設為 3 天時，套用公式(1)可得到
(2005/6/6 - TraceDate) % 3 = 1
所以，TraceDate = {2005/6/5, 2005/6/2, 2005/5/30, 2005/5/27, 2005/5/24, …}

此結果指出，在營運系統中分別屬於 2005/6/5、2005/6/2、2005/5/30、2005/5/27、2005/5/24、…等日期的工作，這些逾期未處理完畢之資料必須列入警訊通知。換言之，若營運系統中，某一工作自從上次追蹤基準日期重設後，一直未被負責人處理完畢，則以此追蹤基準日期往後算的第 1、4、7、10、...(被 Cycle 除，餘數均為 1)天的資料將彙總後產生警訊附檔，而且附檔中這些資料的負責人也將成為警訊通知的對象，這些人的電子郵件地址將會逐一被指定到郵件收件人中，自動收到工作延遲警訊通知的電子郵件。警訊通知的部分邏輯規則設計如表 3 所示。

表 3 警訊通知設計之部分邏輯規則

```
/* 找出工作逾期未處理人名字及其電子郵件 */
Select @Pno=@pno+1, @Developer= @Developer
+ convert(varchar(2),@Pno)+'.'+( d.dvlptmp1) +'   ',
@MyReceiver = @MyReceiver + u.mailalias + ';'
From colorspec s join pccdvlp d on
s.devlopid=d.devlopid and s.purpid=d.purpid join
v_specusr u on d.dvlper=u.usracct
Where (   ( d.clrdel='C' ) and ( d.status<>'C' ) and
( s.usecheck<>'' ) and ( s.colorcfm is null ) and
( DATEDIFF(D, s.trace , @Computetime ) >
@Term   ) and (( DATEDIFF(D, s.trace ,
@Computetime ) % @Term ) = 1) )
Group By d.dvlptmp1, u.mailalias

/* 找出工作逾期未處理人其所對應的 Leader */
Select @Mycopy_recipients =
@Mycopy_recipients + ';' + t.ctlemail
From colorspec s join pccdvlp d on
s.devlopid=d.devlopid and s.purpid=d.purpid join
v_specusr u on d.dvlper=u.usracct join team t on
d.dvlpgrp=t.dvlpgrp
Where (   ( d.clrdel='C' ) and ( d.status<>'C' ) and
( s.usecheck<>'' ) and ( s.colorcfm is null ) and
( DATEDIFF(D, s.trace , @Computetime ) >
@CTLTerm   ) and (( DATEDIFF(D, s.trace ,
@Computetime ) % @Term ) = 1) )
Group By d.dvlpgrp, t.ctlemail

/* 找出本次工作逾期通知警訊之名稱 */
Select @MySubject = JobName
From JobList
Where JobId = 'ClrunCFM'

/* 送出電子郵件給 SQL Server */
Exec master.dbo.xp_sendmail @recipients =
@MyReceiver , @copy_recipients =
@Mycopy_recipients , @message = @MyContent,
@query = 'sp_ClrunCFM', @subject = @MySubject,
@attach_results='true',@attachments='unCFM.xls',
@no_header='false',@width=2000,@separator='
        '
```

3.4 代理程式設定

完成了上述的設計動作，我們還需要進行代理程式的設定才能透過排程功能讓SQL Server自動定期處理特定的作業。我們將依序建立作業的定義，設定執行的步驟，設定作業的排程，掌控作業完成、成功以及失敗時的通知訊息。在圖 5 所示的代理程式作業(Job)中，需指定作業的步驟名稱、命令種類、需要處理的資料庫以及想要執行的程式(在此輸入 exec sp_mailClrunCFM，來執行我們所自訂的預存程序)。最後再新增作業排程，指定作業執行的時間及頻率(在此設定每天上午 6 點執行作業)，並啟用此排程，完成之設定如圖 6 所示。當然，為確保作業執行失敗時能將訊息告知系統管理員以作緊急處理，我們也可在圖 7 的選項中亦進行相關設定動作。

圖 5 新增作業步驟

圖 6 排定作業執行時間及頻率

圖 7 作業執行後告知訊息設定

4. 系統功能實例

為了驗證企業組織內部營運系統的資料庫資料，能夠由代理程式自動定期觸發 SQL Mail 發送查詢命令給 SQL Server，將逾期未處理的資料，依據給定的規則條件來篩選出來並寄出電子郵件警訊通知相關的人員，本研究以企業內部目前實際營運的系統為例來進行實作。此系統的資料庫中有上百萬筆資料，其中有許多早已超過其應該被負責人

處理而仍未處理的資料，也正因為有這些實際的資料，讓本實驗的運作結果受到部門內基層人員的重視與震驚，更受到高層主管的關切與支持。觀察實際發送出來的警訊通知郵件中(如圖 8 所示)，內文中確實只將工作未如期完成的負責人名單列出，並將這些人的電子郵件地址指定到收件者的地方，確保他們能收到警訊通知，並且可以開啟附檔之 Excel(如圖 9 所示)來檢視逾期未完成的工作內容。

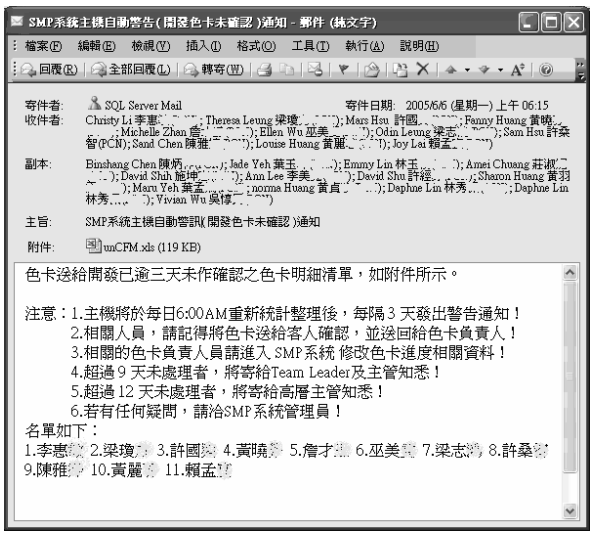

圖 8 警訊通知郵件

	D	E	F	G	H	I
1	開發員	色號	材料名稱	送開發日	追蹤基準日	色卡狀況備註
2	---	---	---	---	---	---
3	李惠	21Z/生膠色	大底	2005/6/2	2005/6/2	
4	巫美	42Y	8mm\IK 鞋	2005/6/2	2005/6/2	5月30日
5	黃曉	77A	0.5mm\TPI	2005/6/2	2005/6/2	5/24CFM00A/0
6	賴孟	01B	01B/44B大	2005/5/30	2005/5/30	
7	賴孟	64M	64M\GIGA	2005/5/27	2005/5/27	5/26 CFM單布
8	賴孟	10C	\MICRO M	2005/5/30	2005/5/30	
9	賴孟	64M	\MICRO M	2005/5/30	2005/5/30	
10	賴孟	30H	30H\GIGA	2005/5/27	2005/5/27	5/26 CFM單布
11	賴孟	00A	1.2mm\NE	2005/5/27	2005/5/27	
12	黃麗	43J	印刷...眼	2005/5/27	2005/5/27	
13	黃麗	54J	1.2MM\HU	2005/5/24	2005/5/24	
14	黃麗	54J	1.2MM\HU	2005/5/24	2005/5/24	
15	許國	08F	10mm\尼	2005/6/2	2005/6/2	
16	許國	08F	\280D/3	2005/6/2	2005/6/2	04B-新色
17	詹才	01B	\190D 尼	2005/5/23	2005/5/30	
18	詹才	05B	0.7mm\FL	2005/5/27	2005/5/30	
19	詹才	05B	0.7mm\FL	2005/5/27	2005/5/30	

圖 9 逾期未完成的工作內容

5. 結論與未來研究方向

本研究的貢獻主要為有效結合組織內部之營運系統的資料庫資料，並應用資料庫代理程式服務，搭配 SQL Mail 的設定，來實作工作逾時之警訊通知。此一人員工作延遲警訊通知的功能設計，可避免增加企業組織導入新資訊系統的費用以及人員重複輸入資料之工作負荷，並提供充份可靠的逾期資料內容來證明警訊通知的正確性。由於未導入此一功能之前，部門與部門間時常因為上游部門工作疏忽未按時完成，導致下游部門工作無法正常銜接而遭受高層主管的責備，更可能造成產品延遲到市場推出而使企業獲利損失。本文所實作之功能目前已經在企業組織內順利運作，並成為上級主管考核部屬工作績效之參考，也增進企業組織協調合作及資源利用的效用。

除了 MS SQL Server 之外，其他的資料庫管理系統也有代理程式的機制，例如在 Oracle Enterprise Manager(OEM)架構中，當 Intelligent Agent 接收到由 Management Server 所傳送而來的命令或 Job 時，便會去執行這些命令，將執行結果回傳給 Management Server，並可透過 Email 等方式寄出結果通知。本研究提出了此一運作機制的設計之後，未來期許能將此機制逐漸擴展至其他的組織及領域上，讓資料庫的代理程式不再只是針對伺服器異常時的訊息告知，或是資料庫中作業執行完成的回報動作而已，更要將其賦予新的使命來創造出更有價值的應用。

參考文獻

[1] 林建煌編譯。2002。現代管理學。華泰文化事業公司。台北。

[2] 施威銘研究室。2001。SQL Server 2000 管理實務。旗標出版股份有限公司。台北。

[3] 陳俊源。2001。SQL Server 系統管理實務。知城數位科技。台北。

[4] C. Frye, “Move to Workflow Provokes Business Process Scrutiny,” Software Magazine, April, 1994.

[5] D. Georgakopoulos, M. Hornick and A. Aheth, "An Overview of Workflow Management:From Process Modeling to Workflow Automation Infrastructure," Distributed and Parallel Databases, Vol. 3, No. 2, pp. 119-153, 1995.

[6] Dejan Sunderic, Tom Woodhead。吳東賢 譯。2001。SQL Server 2000 預存程序程式設計。參格羅希爾。台北。

[7] Ken Henderson. 2002. SQL Server Stored Procedure Basics: Creating a Stored Procedure. http://www.informit.com/articles/article.asp?p=25288&seqNum=4

[8] Mark Linsenbardt, Shane Stigler。許智森 等譯。2001。SQL Server 2000 系統管理-徹底研究。參格羅希爾。台北。

[9] SQL Server Reference Guide: SQL Mail. http://www.informit.com/guides/content.asp?g=sqlserver&seqNum=31&rl=1