行政院國家科學委員會專題研究計畫成果報告

應用離散事件控制理論及先進資訊電子技術之開放式控制器平台研究 (1/3)

On The Study of Open Architecture Controllers Using Discrete Event Control Theory and State-of-the-art Information and Electronic Design Technology

計畫編號：NSC 88-2212-E-009-027-
執行期限：87 年 8 月 1 日至 88 年 7 月 31 日
主持人：胡竹生教授　　國立交通大學電機與控制工程系

一、中文摘要(關鍵詞：開放式架構控制器，數位訊號處理器，離散事件系統)

本研究報告主要是提出統一化的控制系統與自動化軟體架構，使應用離散事件系統規格於監控器合成的方法能同時應用於單一機器監督器的設計，與多台機器整合時監控模組的設計。在文中應用物件導向的技術來分析設計系統中的基本類別及其相互關係，設計其軟體框架(framework)，並以統一模型化語言(UML)來表示，再配合設計樣板(Design Pattern)的應用，使整個架構更具有彈性。最後配合分散式物件的技術來做爲實際應用時的實作方式。

英文摘要 (Keywords: Open Architecture Controllers, Digital Signal Processors, Discrete Event Systems)

In this report, we propose a unified framework for control system and automation software design. Using this framework, supervisory synthesis methodology with DES specification can be used in design control software not only for single machine but also for machines' integration. In the architectural design, basic classes of our proposed framework were model by Object-Oriented Analysis and Design Technique and presented in UML notation. Furthermore, in the mechanistic design, design pattern was used to improve the modality of our framework and make it more flexible. At last, distributed object technique-CORBA was used for implementation in the detail design.

二、計劃緣由與目的

以開放式架構（Open Architecture）爲號召之工業控制器（Industrial Controller）在 90 年代成爲各項控制系統的設計主流。這也使得自動化設備如數控機械的製造商更具競爭力，其主要原因是由於軟硬体的彈性搭配，使先進的製造及控制技術可快速且容易的實現。這項優點，也是產業界與學術研究能否緊密結合的重要契機。我國亦藉助個人電腦之功能，由工研院等單位研發 PC-based controller，其目的也是希望藉由控制器競爭力的提升，能夠推動我國在自動化及機電產業的進展。

時至今日，由於電子(含電力電子)及資訊技術的長足進步，連帶使得如 PC-based 控制器的發展產生了變化。如大型積體電路化，快速且低價的數位訊號處理器(DSP)的問世，網路功能及速度增加，及軟體技術的演變等。同時 PC 由於普及性提高且競爭日益激烈，其架構不斷變化，然而由於其主導權非工業控制，與其過度緊密結合之控制器，在 PC 改朝換代時將面臨技術更新與穩定性的考驗。

因此如何因勢利導，並善用先進的電子與資訊方面的技術，使 PC-based 控制器無論在表面或實質上均能迎合時代的潮流，是相當重要的課題。尤其是網路技術的演進使得自動化整合時資訊結構改變，而身爲自動化系統最底層之工控器，勢必也將跟上其腳步。本計劃的目的，即是研究在此趨勢下以 PC 爲發展及執行平台的工業控制器結構及軟硬体相關技術。在理論上將嘗試應用離散事件系統(Discrete Event Systems)控制的技巧，設計即時迴授、順序控制、人機界面及網路通訊等模組的整合法則。實作方面將設計以數位訊號處理器爲核心之即時控制模組，並應用即時多工核心、物件導向及先進網路程式設計的技巧，以達成控制軟體模組化及網路連結透明化的目標。

三、研究方法及成果

- 緒論

從軟體的觀點來看以電腦爲基礎的控制系統中，在設計排程或是監控系統時電腦系統主要有兩種[2]，集中式的系統(Centralized System)與分散式的系統(Distributed System)。分散式的系統由於把決策與運算過程分配給其它電腦，使得系統整體的容量與運算量增加，能處理較複雜巨大的系統，再加上整合系統中規格通常只會與某些模組相關，使得在整個系統的建構中，能以由小到大的方式來設計，使得子系統的設計較爲簡單，可是正由於分散式的處理，電腦與電腦間的通訊問題與子系統與子系統間的整合問題，變成是分散式系統中設計因難的原因。因此，本研究的動機就是在找出當機器連接使用的是網路時，它可以爲工業控制帶來什麼樣的影響，原本的通訊協定如何處理，控製架構如何設計，是否存在一標準的設計方式，可以在這樣的架構下，快速地發展應用程式。

- 研究方法

本計劃將提出統一化的(Unified)控制系統與自動化軟體架構。使控制軟體與自動化程序控制能快速地建構，同時讓此架構能符合下列目標：

1. 硬體介面統一化。
2. 分散式(Distributed)的軟體架構。主要指軟體介面統一化。
3. 階層式(hierarchical)的軟體架構。機器的監督器與自動製作系統的監督器設計方式統一化。
4. 可重覆使用的軟體架構。

而研究的方法則先分析的工控環境，找出它在軟體設計上所擁有的特點，分析其所需的資訊，然後將應用物件導向技術與離散事件系統的控制理論來解決問題，這些物件技術包含了設計樣版(Design Pattern)、統一標準化模型語言(UML)與分散式物件環境(Distributed Object Environment)等等，這些技術將被用來設計監控系統的軟體框架(Framework)。

物件導向技術（Object-Oriented，OO）可以提高生產力，增加可靠度及減少缺陷，同時它還是一種新的思維模式（Paradigm），它的目的是爲系統建立物件導向式的分析模型，並依此模型以物件導向程式語言來撰寫應用程式。系統是由很多的獨立物件所組成，這些物件藉由訊息彼此溝通，這種思維模式比關聯式更接近人類思考方式且更易於面對外在環境的改變。同時，在應用程式的開發過程，我們引入設計樣版(Design Patterns)的概念，憑藉程式設計師經驗的累積，在物件導向的特殊軟體設計問題上，提供簡捷的解法。因此在建構複雜或龐大的系統上，Patterns 的使用非但強化模組的結構，更可降低系統複雜度並縮短開發時程。最後，再藉由分散式物件技術的導入，使實作上的設計，更爲簡便。

- 所提出的架構

從硬體上來看，系統的架構如圖 1，可以看成許多的工作單元以網路相連接。工作單元可以是單一具有智慧的機器或是 PC 或 Workstation 所控制的感應器與制動器，同時這些單元須滿足有智慧及即時性時間規格。

若以軟體的角度來看如圖 2，每台機器對外所提供的是模組(module)，如感應器(sensor)模組、控制模組(control module)等等。而且每台機器所提供的模組數目不需相同。而任一項工作所需的模組也是不同的，例如工作 A 爲監督模組，只想知道工作的狀態，而不須控制或不具控制權限，則它可能只使用 machine C 的 sensor module F 與 machine A 中的 sensor module B。而工作 B 只與機器 AB 有關，則工作 B 則只需要利用模組 ABCD 而不需知道 EF。在設計單一機器的控制器時，不需要考慮說像感應器(sensor)或是致動器是不是在機器上，而直接設計控制程式。而當工作 A 與 B 是機器整合的工作時，同樣的我們只需考慮所使用的那些模組(module)，而不需要考慮模組所在的位置，這樣一來從軟體的角度來看，機器的界限等於是被打破了如圖 3，但如以物件的觀點來看，可以看成是物件 Pool。同時，還希望能擁有這些物件能擁有統一的軟體介面如圖 4。

綜合來說，以軟體的觀點，這樣的架構因軟體介面的統一有下列優點。

1. 易於控制程式的撰寫。
2. 機器維修時的便利。
3. 彈性製造的考量。

在這樣的軟體架構下，需解決下面幾面問題。

1. 模組監控的問題
2. 如何讓模組間沒有機器的界限
3. 模組如何設計?

- 模組監控的問題

本論文採用 S. Balemi[1], 在 1993 年所提出的通用雙層式控制架構。絕大部分的應用中，自動控制系統必須提供兩種操作模式：「自動模式」與「手動模式」。在自動模式中，控制器主動的決定命令事件爲何，不需操作員下達命令；手動模式則可以在機器發生問題或測試時由操作員進行手動的問題排除與處理。本架構如圖 5，適用於此類的控制系統，控制系統分爲二層，控制器（Controller）與監督器（Supervisor）。

控制器接受受控體所產生的回應事件，並更新其內部狀態。根據此內部狀態以及其工作內容，「產生」命令事件。控制器必須無條件接受受控體產生的回應事件。在自動模式中，控制器是會自動執行工作流程的機制，在手動模式中，手動操作的作業員即代表控制器。

監督器與控制器不同，監督器並沒有主動產生事件的能力，而只能被動的禁制事件的發生。事實上監督器就像是一個動態的過濾器，接受受控體傳回的回應事件，更新其內部狀態，並根據此內部狀態決定目前可以被准許的事件集合。所有控制器產生的命令事件都必須經過監督器的核准，才能夠到達受控體，才會被執行並造成控制器與監督器內部狀態的改變。監督器發出的命令事件必須一定被被受控體所接受。這個雙層式控制架構的主要優點爲：當操作員把自動發出命令的控制器關掉，並且以手動模式操作系統時，由於其手動命令仍然得經過監督器的核准，只有符合監督器規格的命令才能夠到達受控體，所以即使在手動模式下，也不會發生因爲操作員疏失而造成危險或誤動作的情形。

介面層的作用在於扮演由控制器與監督器的「邏輯世界」到受控體的「真實世界」間的橋樑。就像一個中間的轉譯器一樣，協助達到監視與控制的目的。

除了控制系統被分爲控制器與監督器兩層

之外，系統規格也可以分爲兩種：隱性規格（Implicit Specification）與顯性規格（Explicit Specification）。隱性規格爲系統無論如何必須符合的規格，它的特性是與機器相關，而與執行的工作無關。無論系統是在自動模式還是手動模式下，隱性規格都應該被滿足。因此，監督器便是根據隱性規格而設計。顯性規格則是與機器無關的部分，其內容與系統所欲執行的工作相關。顯性規格乃是由控制器主動的送出命令事件來完成，其目的在使系統成功的執行到某些標記狀態（完成工作）。通常我們要求顯性規格也同時符合隱性規格，如此一來，控制器所發出的命令事件將必定不會被監督器駁回。

因此設計的目標，變成在設計監督器，本報告所採用監督器設計方法是用Wonham[3][5]所提出之合成的方式，該設計方法以自動機模型爲基礎，將事件分爲可控事件與不可控事件，藉由控制系統的特性，可控性、通暢性等與規格等，從系統的自動機模型中找出一塊子自動機，使系統符合規格要求，至於詳細的實作方式與演算法請參考[11]中所述。

- 機器通訊問題的設計

此問題解決目的在於使模組間不存在機器的界限，讓設計機器間整合的程序控制時，能夠跟在設計機器內部一樣，就好像該模組就在該控制器內。這裡使用的技術是分散式物件技術 Corba。CORBA 規格對一個物件訊息仲裁者 (Object Request Broker) 之標準介面作了詳細的規定，其主要目的是讓物件操作其它系統物件時，不必理會複雜的網路通訊問題，而把這些問題交給一個標準的分散式方法來解決，此標準的分散式方法即爲 Corba 所定義。根據 OMG 的定義，一個物件請求仲介者爲一可提供分散式環境上各個物件透明化 (transparent) 的請求服務與回應接收功能的應用程式建構工具。CORBA 架構是被設計用來支援物件在不同程式語言下被使用的分散式行爲，爲達到此目的，便有能配合大量現存語言的 IDL，來定義特定的分散式物件所提供的服務。CORBA 定義了一廣義的協定，向物件提出請求及替物件回應請求，其中有一 IIOP 的協定能確保用戶端的應用程式與伺服端的基本物件之間的互動。CORBA 藉由定義一組低階的服務在基本的功能上延展。這些服務用來接受應用程式不定時因正常執行所產生的請求，這些服務都包括在 CORBA IDL。當我們設計好 IDL 介面後，便可經由 IDL 編譯器編譯至對應的物件化程式語言(C++)的類別，然後，物件設計者繼承編譯出來的 Server 端類別，並填上實際運作的程式碼。完成物件的設計，因此，應用 Coba 技術最重要的步驟，就是設計介面。

除此之外，在工業控制上，機器與機器之間的通訊功能主要是提供特定服務如工業資訊服務 Variable service、Message Passing、Resource Sharing、Event Management、Program Management 等等服務，Corba 都正制定標準[6][7]，因此，應用 Corba 處理機器通訊問題也可同時提供這些服務。

- 模組的設計

在物件池中，希望每個物件的樣子都是一樣的，以 Corba 實作來說，就是依據之前所述的監督器理論，來設計 IDL 介面，該介面的目的，主要有二個。

1. 提供所需的自動機模型，以供設計監督控制器使用
2. 接受並處理模型中所提供的事件

首先先設計自動機介面 XAutobase，此介面藉由 EventCount、StateCount、GetEvent、GetState、GetInitialState 等函式取得物件的自動機模型，而以 GetCurrentState、與 HandleAction 等函式來補抓自動機運作時的狀態進行控制。同時爲了達成彈性製造的目的，不同的控制程式皆對應

不同的多個自動機，爲此我們設計虛擬系統介面 Xplant，可以藉由 AddPart 與 RemovePart 增減系統中的自動機物件，而以 GetCurentState，GetAutomaton，HandleAction 等函式供正常運作時獲得系統的狀態，或操作其內部的自動機。完整介面如圖 6。

- 發展工具

再決定了模組的設計方式，與控制架構後，爲能快速地發展應用程式，我們設計了兩套工具程式來幫助我們發展應用程式。

- 互動式的監督器合成環境

監督器設計環境的目的是用來合成監督器，程式以 BCB+Visibroker 實作而成，可以在 Window98 下執行，藉由點選不合規格的狀態，將之標示起來，再加以進行合成，然後可以將合成後的監督器儲存成檔案，其副檔名爲 sup，同時進行合成的模型可以是文字檔，其副檔名爲 atn，或者是在 CORBA 環境下的自動機物件，程式會自動找出有實作 XAutobase 介面的自動機，甚至在同一監督器的合成可以是兩者混合，有的自檔案輸入，有的爲實際存在的 CORBA 自動機物件。同樣地我們採用模組化的設計，每一監督器代表的是一規格。如系統有多個規格則可以藉由在控制環境中，載入多個監督器，進行控制。因此監督器合成環境有幾個特點:

1. 自動機模型可由檔案、實際運作之自動機物件或是混合的方式來載入。
2. 採用離線方式來合成。3.對於規格具有模組化的特性。

同時，應用此合成環境的步驟如下:

1. 設計自動機模型成文字檔或自動機物件。
2. 將規格詮釋成對應於自動機物件狀態組合的刪除。
3. 在發展環境中點選不合格的狀態並按下”Mark as Bad State”的按鈕。
4. 開始合成，完成後選擇存檔並對此規格命名。將檔案留置控制環境使用。

特別注意的是可能某些情形下，監督器在進行合成後，所有的狀態皆被刪除，這種情形有兩種可能。第一是此規格對於進行合成的系統來並不可能被滿足。第二種情形則是規格的詮釋有問題。不管是那一種情形，這樣的環境還是對於設計的過程有所幫助。

- 自動機控制環境

該環境可由監督器存檔載入多個監督器，使控制行爲受到監督，符合多項規格。也可不載入監督器，在沒有監督器限制下，直接對系統進行控制。在此控制環境下所控制的目標物是能獨立運作的自動機程式物件，而不只是單純的訊息模擬。同時，此環境應用了 Polling 的程式技巧，處理非同步事件的發生。而應用此控制環境的步驟如下:

1. 設計自動機模型成自動機物件。
2. 將規格詮釋成對應於自動機物件狀態組合的刪除。
3. 在監督器發展環境合成監督器。
4. 載入欲進行控制的自動機物件
5. 載入欲使系統滿足規格的監督器
6. 進行單步執行，錄製控制器，載入控制器並由控制器自動控制等動作。

四、結論與討論

本報告提出一個統一化的控制系統與自動化的軟體架構，在這裡所謂”統一化”有幾層的涵意與特性。

第一、所有機器的連結皆使用網路，所以連結的方式統一化。

第二、所有機器提供的軟體模組，皆實做自動

機介面，軟體介面統一化，因此，整個控制架構的設計不會因爲控制單元的增加或者減少受到改變。傳統上的機器整合設計，當機器增加或減少時，控制程式勢必重新撰寫，而且在規格的考量上也會因此更加複雜。如果應用本報告所提的架構來發展，則不管自動機模組數目的多寡，都可以用同樣的步驟再合成一次，然後利用控制環進行程序控制，不會因爲模組的增加或減少導致控制程式設計的重新改寫，同時還可確認系統規格不會受到影響。

第三、設計方式統一化。設計機器的監控器與設計機器整合時的監控系統，其設計方式皆是相同的，設計機器時，將機器內部的元件模組化成自動機，再依規格合成監督器，並應用控制環境來控制，而將控制器本身也實作一個自動機介面，讓機器整合時能用來合成使系統符合規格的監督器，同時也讓控制環境能直接對機器的控制器進行控制，因此整個架構的設計有著階層式的特點。

第四、通訊方式統一化。本報告的設計，以和物件溝通的方式取代傳統通訊協定的複雜，這種設計架構的應用，帶給工業界最大的影響就是以後機器間整合工作的重擔，大部分由機器製造者來負擔，機器製造商提供實作部分或全部的機器控制功能的物件，同時此物件也提供自動機模型，再藉由 Corba 的技術應用，使得在應用程式設計時可以直接操作物件就如同操作機器內的物件，以操作物件的方式取代傳統的通訊協定複雜的溝通問題，同時也減輕系統整合者的負擔。

更因爲這個架構在設計上可達到提供所謂的二進位可用(binary reusable)物件，這使得當介面(interface)制定完成後，許多的機器製造者可以再自行設計控制器，而只露出介面部分，讓廠商能保持良好的智慧財產權，同時又可使機器本身與系統或網路整合運作得很好。

應用此架構的設計方式於機器整合中，帶給工業界最大的影響就是以後機器間整合工作的複雜度，可以由機器製造者來負擔而減輕。機器製造商以物件模組的方式提供自動機物件，該物件實作部分或全部的控制功能，同時提供自動機模型，再藉由 Corba 的技術應用，使得在應用程式設計時可以直接操作物件就如同操作機器內的物件，以操作物件的方式取代傳統的通訊協定複雜的溝通問題，同時也減輕系統整合者的負擔。

此外，在本報告中所提出的軟體架構主要是依據自動機爲理論基礎的監督器合成設計方式，然而在離散事件系統的模型化方式還有著不同的模型如 Petri Net，如果模型不同，假設使用 Petri Net 的理論來進行監督器合成，架構的精神仍可使用，要解決的問題仍然不變，原來的自動機物件，只要改成能提供 Petri Net 的模型資訊的 Petri Net 物件，其分析與建立軟體框架的方法仍然一樣，因此找出一個更適合設計監督器的模型化方式也是一個可以努力的未來方向。畢竟，某些情形下，Petri Net 來模型化比以自動機來模型化來得簡單，關於以 Petri Net 來進行合成的理論[9]中有較爲詳盡的介紹。

還有在 UML 中狀態圖的表示法 Statechart 比起自動機表示法多了階層式的概念，同時還有同步事件的機制，很適合用來模組化一個系統，而狀態圖 Statechart[8]表示的模型也更適合在程式寫作上的實際運用。關於如何用 Statechart 來合成監督器，在應用時有什麼限制等，可在[10]找到參考。

五、計畫成果自評

項 目	完成情況
與原計畫相符程度	100%
達成預期目標	90%
研究成果學術價值	新型控制器設計
研究成果應用價值	具實用性
學術期刊發表合適否	投稿中

申請專利合適否	否
主要發現或其他價值	軟體著作權

六、參考文獻

[1] S. Balemi, et al., “Supervisory control of a rapid thermal multiprocessor,” *IEEE Trans. Automat. Contr.*, vol.38, no.7, pp.1040-1059, Jul. 1993.

[2] Dilts, D. M., Boyd, N. P., “The Evolution of Control Architechtures for Automated Manufacturing Systems”, Journal of Manufacturing Systems, Vol.10 No. 1 pp.79-93, 1990

[3] P. J. Ramadge and W. M. Wonham, “The control of discrete event systems,” *Proc. IEEE*, vol.77, pp.81-98, Jan. 1989.

[4] H. Wong-Toi and G. Hoffmann, “The control of dense real-time discrete event systems,” in *Proc. 30th Conf. Decision Contr.*, Brighton, UK, Dec. 1991, pp.1527-1528.

[5] W. M. Wonham and P. J. Ramadge, “On the supremal controllable sublanguage of a given language,” *SIAM J. Contr. and Optimization*, vol.25, no.3, pp.637-659, May. 1987.

[6] “CORBA-based Machine Control White Paper”，OMG Document:mfg/98-03-10

[7] “Data Acquisition from Industrial Systems”, OMG Document:dtc/99-01-02

[8] Bruce Powel Douglass, “UML Statecharts”, Embedded Systems Programming,1999 January

[9] Ratnesh Kumar and Lawrence E. Holloway, "Supervisory control of deterministic Petri Nets with regular specification languages", IEEE trans. on Automatic Control, vol. 41, no. 2, 1996.

[10] Y. Brave and M. Heymann, "Control of discrete event systems modeled as hierarchical state machines" IEEE transaction on Automatic Control, vol 38, no.12, 1993.

[11] 賴勇孝，“改良式離散事件監督器合成與控制器的設計方法”，國立交通大學碩士論文，民國 86 年。

七、圖表

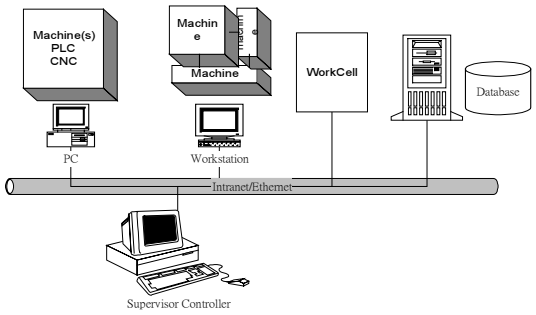

圖 1 所提之硬體架構

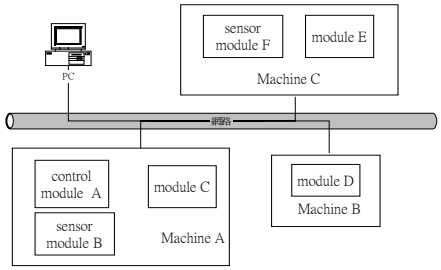

圖 2 軟體觀點

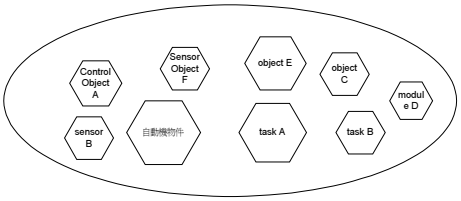

圖 3 模組間不存在機器界限

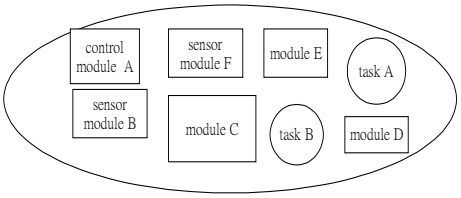

圖 4 物件的軟體介面統一

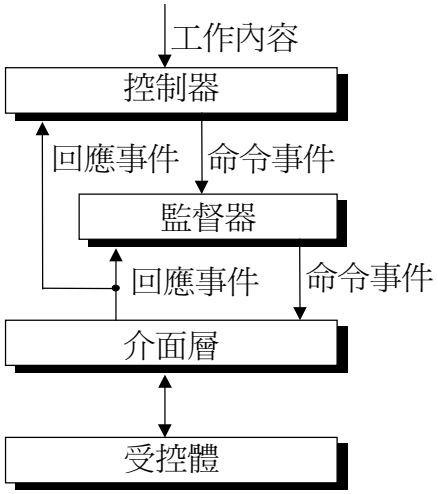

圖 5 雙層式的控制架構

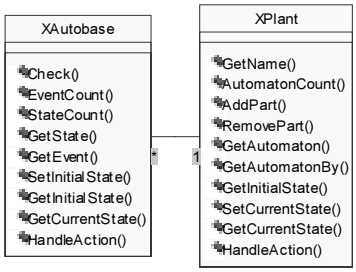

圖 6 自動機與虛擬系統介面