正規方法應用於建構安全個案
Formal Methods for Safety Cases

楊仁賢
Zen-Shien Yang
元智大學資訊工程研究所
中壢遠東路 135 號
Yuan-Ze Universit
Chung-Li, Taiwan

范金鳳
Chin-Feng Fan
元智大學資訊工程研究所
中壢遠東路 135 號
Yuan-Ze Universit
csfanc@saturn.yzu.edu.tw

林錦銘
Jing-Ming Lin
臺電核技處
台北羅斯福路二段
Taiwan Power Company
Taipei, Taiwan, R.O.C.

摘要

本文提出應用正規方法於安全關鍵計算系統執照審查作業上的新應用。安全個案為安全關鍵系統申請執照時所需提出之一組證據文件，用以說明其系統符合安全要求。現行的安全個案建構方法多數不夠嚴謹，本文以正規規格為核心，發展安全個案建構方法。我們以一數位化核能保護系統為範例，制定軟體正規規格並應用正規規格來證明此系統設計符合法規要求，同時亦發展以規格為基礎的測試及線上偵錯等技術，經由這些方法可提供較客觀嚴謹的安全論證。

關鍵詞: 正規方法、安全個案、安全關鍵計算系統、斷言、反應器保護系統。

Abstract

This paper proposes applying formal methods to license review process for safety-critical computing systems. We developed a formal-specification-based method for safety case construction. A reactor dig ital protection system is used as our case study. Formal specifications were designed, and then formal arguments on these specifications have been developed to verify that the syste design conforms to regulatory requirements. Besides, specification-based testing and run -time assertions have also been used. These methods provide more objective and rigorous safety arguments than those gnerated by conventional approach.

***Keywords**: formal methods, safety case, safety -critical computing systems, assertion, reactor protection systems.*

1 緒論

正規方法是以數學為基礎的軟體發展方法。本研究提出應用正規方法於安全關鍵計算系統執照審查作業上的新應用。正規方法與法規審查為一絕佳的組合，正規方法可提供嚴謹具說明力的論證，此為安全關鍵計算系統審查作業所最需要的。如何確定系統符合安全法規的要求為審查作業最重要的工作，正規方法正可應用於此。安全個案則為近年來安全關鍵系統執照申請的一新趨勢，安全個案為一組證據文件，用以說明其系統符合安全要求。現行的安全個案建構方法多數不夠嚴謹。若採用正規方法來發展及驗證，則可提供較有說服力的證據，減少外界質疑。本研究以一核能軟體為案例，首先制定其正規規格，並發展以正規規格為基礎的安全論證，及相關測試和線上偵錯技術，提供較客觀嚴謹的安全個案，以利法規審查。

2 正規規格的制定

需求規格是整個軟體發展過程中最重要的基準文件，以嚴謹、精確的方法來制定與驗證需求規格，有助於後續階段的發展及安全論證的建構。我們以一核能反應器保護系統為例，使用 VDM[1]、Petri Net[7]及 SCR[3]多種方法撰寫正規規格。

首先我們簡述應用案例如下（如圖 1 所示）：
有四套相同控道（channels）的反應器保護系統，其功能為緊急事故時將控制棒快速插入反應器使系統跳機。當兩個以上的控道達到跳機條件時，此保護系統就會驅動反應器自動跳機。每個控道各有三個功能單位：DTM (Digital Trip Module)、TLU (Trip Logic Unit)、OLU (Output Logic Unit)。此系統使用負邏輯（0 表示沒電力/跳機，1 表示有電力/正常）。各功能單位之功能分述如下：

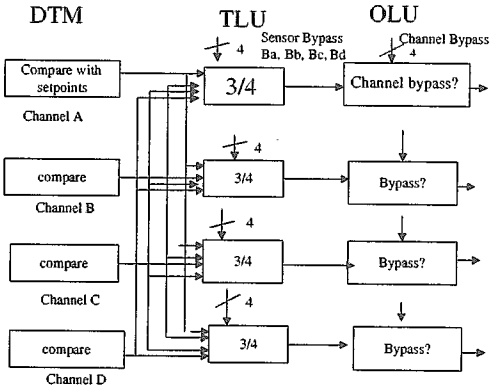

圖 1. 保護系統

DTM (Digital Trip Module) :

此階段用感測器 (sensor)的偵測值與系統設定值做比較，如果達到急停條件就傳送跳機 (trip) 信號給本身的 TLU 及其它三個控道的 TLU，否則傳輸正常 (OK) 訊號。跳機條件包括壓力、水位、溫度、中子量、發電功率等超過安全的範圍。

TLU (Trip Logic Unit) :

此階段執行四選三的投票動作。如果收到三個以上控道的 DTM 所傳送來的正常訊號，就傳送正常訊號給自己的 OLU，否則就傳跳機訊號。另外，操作員亦可在此階段輸入感應器旁通 (sensor bypass : SBa，SBb，SBc，SBd) 的指令。但限制只能有一個控道可被設定為感應器旁通，而被旁通的控道的 DTM 訊號將被忽略，(意指只對其他三個控道做三選二的動作)。

此階段公式為

*=[(A+B)+(SBa+SBb)][(A+C)+(SBa+SBc)][(A+D)+(SBa+SBd)]
[(SB+C)+(SBb+SBc)][(SB+D)+(SBb+SBd)][(C+D)+(SBc+SBd)]

*= 0 即需跳機

A，B，C，D 為控道之訊號 (trip = 0，OK = 1)。SBa，SBb，SBc，SBd 為感應器旁通訊號(true=1,false=0)

OLU (Output Logic Unit) :

原則上 OLU 將 TLU 階段傳過來的訊號直接輸出。但若某一控道需檢測，操作員則可下控道旁通 (channel bypass) 的指令至其 OLU。在旁通情況下，不論 TLU 送來的訊號為何，此控道一律都傳正常訊號到最後階段。限制只能有一個控道可為控道旁通。

最後階段 :

因為控制棒的運用需要三個以上控道的電力，即需三個以上控道輸出正常訊號系統才不跳機。

本研究以三種不同的方法來制定此範例的正規規格，即 VDM[1] , Petri Net[7] ,及 SCR[3]規格。VDM 是一個廣被採納的正規規格方法，但由於 VDM 無法表達並行 (concurrency)我們結合狀態轉換圖 (state transition diagram) 和 VD 以表達並行關係。每一控道的上層狀態轉換圖如圖 2 所示。

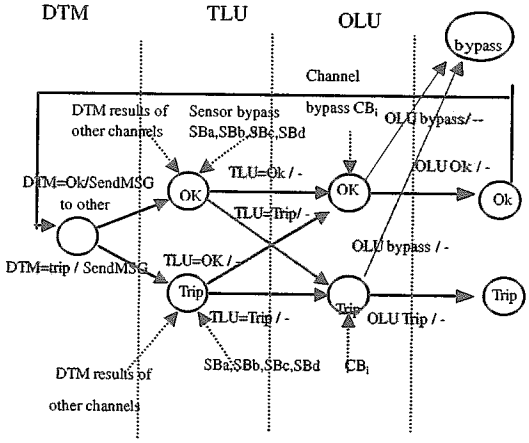

圖 2 狀態轉換圖

控道間使用訊息傳送以達到並行的目的。使用的訊息包括接收訊息及傳送訊息 (SendMSG)。狀態轉換函式為 δ:(狀態 * 輸入訊息 → 動作 * 狀態)。其中輸入訊息為操作員輸入及其他控道訊息(虛線表示) 或前階段結果(陰影表示) ，而前階段動作則以 VD 定義。下列為部份 VD 規格,其中 mode'代表 mode 上一刻的狀態:

```
type
  systemoutput = { OK , trip }
  channels = { A , B , C , D }
  channelstatus = { OK , trip }
  status = channels → channelstatus
  sensorbypasstype = channels → { true , false }
  channelbypasstype = channels → { true , false }

state safe of
  sensorᵢ : { powerᵢ , Maxpowerᵢ , ReactorPᵢ ,
              DrywellPᵢ , ControlRodPᵢ ,
              HighSeismicᵢ : N , ReactorWLᵢ , WaterTᵢ :
              Float , NBSᵢ , NMSᵢ : B }
  mode : status
    sensorbypass : sensorbypasstype
    channelbypass : channelbypasstype
    system : systemoutput
.....

DTM (id : channels) m : systemoutput

  ext       rd mode : status
      wr mode : status
  pre true
  post   mode(id) = trip ⇔
       ((powerid > 40% Maxpowerid)        ∨
        (powerid > 40% Maxpowerid)        ∨
        NMSid      = true                          ∨
        ReactorPid  > 1040                 ∨
        ReactorWLid < 1279.7               ∨
        DrywellPid  > 2   ∨ ...) ∧ m = mode(id)
```

```
TLU (id : channels) m : systemoutput

//input sensorbypass : sensorbypasstype
  ext rd mode : status
      wr mode : status

  pre ¬ ( ∃ i , j ∈ dom sensorbypasstype • i ≠
          j ⟹ sensorbypass (i) = true ∧
               sensorbypass (j) = true )


 post mode(id) = OK ⟺
     ((mode'(A) = OK ∨ mode'(B) = OK ∨
       sensorbypass (A) = true ∨
       sensorbypass (B) = true) ∧

      (mode'(A) = OK ∨ mode'(C) = OK ∨
       sensorbypass (A) = true ∨
       sensorbypass (C) = true) ∧   //四選三公式

         .... ) ∧
       m = mode(id)

OLU (id : channels) m : systemoutput

//input    channelbypass : channelbypasstype
  ext rd mode : status
      wr mode : status

  pre  ¬ ( ∃ i , j ∈ dom channelbypasstype • i
          ≠ j ⟹ channelbypass (i) = true ∧
                 channelbypass (j) = true )

  post ((channelbypass (id) = true ⟹ mode(id) = OK)
       ∨ (channelbypass (id) = false ⟹ mode(id)
        = mode'(id))) ∧ m = mode(id)
```

斐氏圖(Petri Net) 可表達出並行的程序。此範例斐氏圖規格如圖 3 所示。圖中"Delay and fire once" 的定義為：當有符號 (token) 時，不論有幾個符號，此回合只能觸發(fire)一次，並且清除掉其它的符號。Petri Net 提供良好的視覺效果。圖中位置(place) 及 變換(transition) 以 P_{ijk} 及 t_{ijk} 編號，i 表示不同之控道，j 表示階段。

SCR 則以表格的形式呈現，有助於閱讀及分析證明。由於 SCR 亦無法表達並行，我們使用與圖 2 相同的狀態轉換圖和 SCR 結合。此案例只需用到 SCR 的內部事件表 (如表 1-3 所示)。表中@T(條件式)(或@F(條件式)) 表示當條件式為真(或為假)時事件發生。

上述三種規格表達方式各有特色，以模組式功能表達、圖型或表格形式表現，各規格的證明方法亦有差異。本研究以 Petri Net 及 SCR 的規格進行符合法規要項的證明，以 VDM 進行程式碼的衍生 (program transformation)，並進一步加入線上偵錯的斷言。

表 1. DTM 階段之事件表

Mode	Events	
OK	@T(power >40% Maxpower) OR @T(power > 40% Maxpower) OR @T(NMS = true) OR @T(ReactorP > 1040) OR	@F(power >40% Maxpower) AND @F(power > 40% Maxpower) AND @T(NMS = false) AND @T(ReactorP ≤ 1040) AND ....
DTM	Trip	OK

表 2. TLU 階段之事件表

	Events	
	@T(*=1)	@T(*=0)
TLU	OK	Trip

(*: 表示 TLU 之公式。)

表 3. OLU 階段之事件表

TLU	Event		
OK	@T(channelbypass = false)	False	@T(channelbypass = true)
Trip	False	@T(channelbypass = false)	@T(channelbypass = true)
OLU	OK	Trip	Bypass

3 法規安全要項的證明

安全個案主要是一個論證過程，確認此安全關鍵系統符合法規要求的安全準則，當正規規格制定後，本研究提議以正規方法證明此系統之設計符合法規安全性要求。

本文針對法規要求的單一失誤準則 (single failure) 為例。所謂單一失誤準則就是：單一元件失誤不會影響到系統正常運作。單一失誤準則為最重要的法規之一。我們利用 Petri Net 所制定的規格，分段式以路徑圖 (reachability graph) 來證明此系統設計符合單一失誤準則。

圖 4 為以控道 A 為主的主要路徑圖。所謂的路線圖就是整個系統所可能達到的情況。其中 t_{ijk} 與圖 3 相同，P_{SB} 與 P_{CB} 分別代表感應器旁通與控道旁通的位置。圖右下角方塊代表以控道 B, C, D 為主的路徑圖，他們的輸出會接到 P500。 圖 4 含蓋了 DTM 階段中，四個控道皆正常、一個跳機、兩個跳機及三個跳機的狀況，以及其後續的發展，虛框部份為註解。單一控道 A 故障，有兩種狀況：(1) A 應該為跳機狀況 (Trip)，因故障變成正常 (OK)，產生額外之符號 (token)。(2) A 該為正常狀況 (OK)，卻因故障成了跳機狀況而沒有符

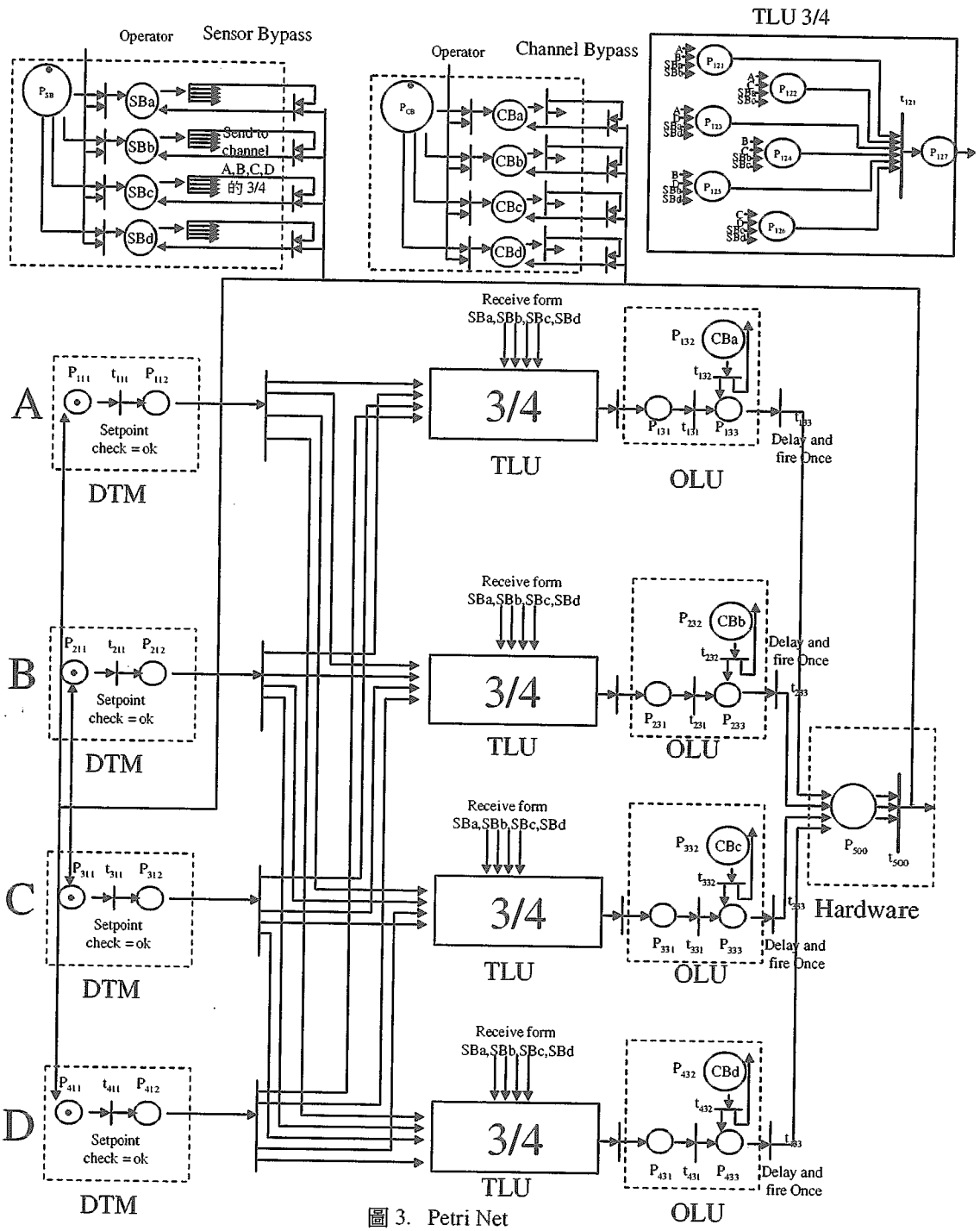
TLU 3/4
Operator
Sensor Bypass
Operator
Channel Bypass
SBa
SBb
SBc
SBd
Send to channel A,B,C,D 的 3/4
CBa
CBb
CBc
CBd
Receive form SBa,SBb,SBc,SBd
3/4
TLU
OLU
Delay and fire Once
Setpoint check = ok
DTM
A
B
C
D
Hardware

圖 3. Petri Net

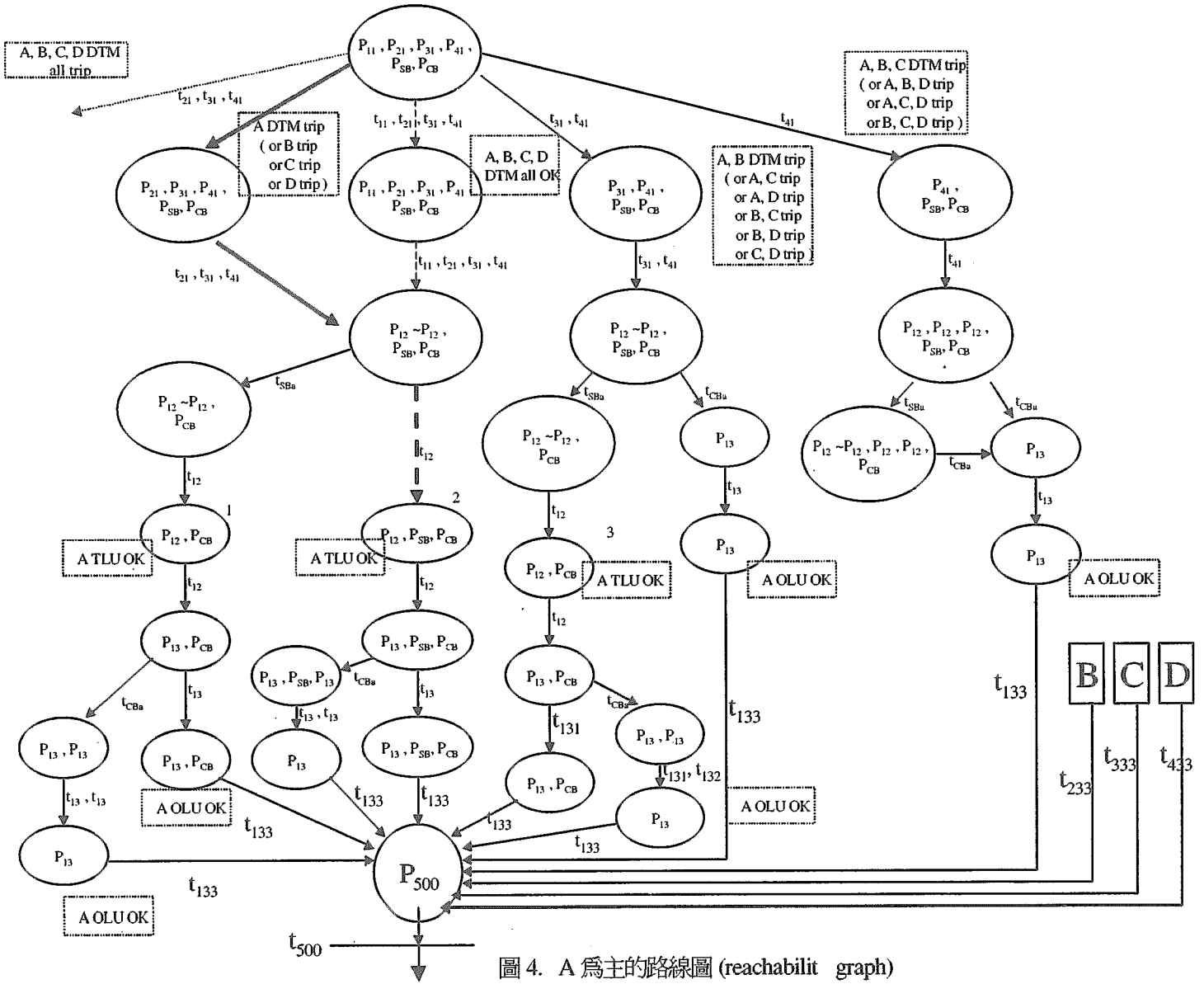

圖 4. A 爲主的路線圖 (reachabilit graph)

號（token）（即A失誤為0）。因為狀況(1)中，單一失誤的假設下，控道B，C，D無故障，皆為跳機狀況，根據系統運作規則，最終的系統狀態一定為跳機，故此情況下，系統仍正常運作。所以本文只針對第2種狀況加以探討。進行方法如下：

1. 列出單一控道故障的所有狀況。
2. 對每一狀況，用路徑圖推導其他控道的行為，以檢查在單一失誤下，系統是否仍正常或安全運作。

上述步驟1中，每一個個案都含蓋前題（assumption）和結果狀態（resulting status）。步驟2中，我們依路線圖(圖 4)找出觸發前題而發展至結果狀態的路線。若個案結果為正常，推導方法如下：

1. 在路線圖中首先標註前題及結果相對應的位置(place)及變換(transition)。
2. 由結果往回推導到前題狀態。刪去不符合前題假設狀態的路徑，則可找到符合前題到結果的中間過程路徑。

若個案結果為跳機，推導方法如上所述，除了下列部份：

1. 仍標註結果為正常的位置(即其相反狀態)，則步驟2所推導出的為不可能到達的情況。

首先我們列出控道A故障的所有狀況，包括：

(1)（前題）A在DTM階段故障且A沒有感應器旁通，（結果）A的TLU結果為正常。

(2)（前題）A在DTM階段故障且A沒有感應器旁通，（結果）A的TLU結果為跳機。

等等8個個案。

試述個案1,2的推導如下：

個案1：在此個案中，前題假設部份為□（A DTM故障而無符號（token））,結果為P_{127}（A的TLU結果為正常）。我們首先標註前題及結果相對應的位置(place)及變換(transition),在圖4中P_{127}出現在以陰影顯示的3個位置(place)，由此往回推導中，其中註明1,3部分的位置皆不合前題(A無感應器旁通，t_{SBa}不會觸發)故中間可能過程，如粗體箭頭線所示。即此個案B,C,D的DTM結果必需為正常。此種情況符合系統仍正常運作。

個案2,推導過程有如上述。此個案結果為A的結果為跳機，即路線圖4中，陰影部分的位置(places)沒有符號(token)。再由此往回推至前提，推導出的為不可能的情況，即P_{121}~P_{126}不能全有符號(token)，換言之，B,C,D的DTM不能全為正常訊息(OK)，即虛線部份不為真，因此此種情況A故障系統仍安全運作。

所有A故障的情形，皆可如此驗證，結論為單一失誤下系統仍正常/安全運作。

4 SCR的模式恆真式分析

更進一步，我們將推導單一失誤下的SCR模式恆真式，進而再次證明系統設計符合單 失誤準則。根據Jeffords and Heitmeyer所提出之SCR模式恆真式(mode invariant)推導[4]的方法，我們進行了SCR規格之恆真式推導。在此我們針對控道A故障下對第二階段(TLU)所做之恆真式推導（表4與表5)。

表4是說明狀態經過事件之後會從原有狀態轉變到新狀態。因為第二階段是做四選三的動作，因此可以用此方法找出控道A失效的狀態恆真式。其中"state"為DTM_i的狀態且只能有一個 sensorbypass(i) = true。假設控道A失效，我們對控道B做驗證。

表 4. TLU的模式轉變表（以B為例）

Channel B (Old Mode)	Event	New Mode
OK	$\exists$ i ∈ Channels (C , D) · state(i) = trip ∧ sensorbypass(i) = false	Trip
OK	State(C) = OK ∧ state(D) = OK ∨ State(C) = OK ∧ sensorbypass(D) = true ∨ State(D) = OK ∧ sensorbypass(C) = true	OK
Trip	State(C) = OK ∧ state(D) = OK ∧ Sensorbypass(B) = true	OK
Trip	Sensorbypass(B) = false	Trip

表5根據Jeffords and Heitmeyer的方法所推導的狀態恆真式的過程。其中N_i(m)是模式進入條件（mode entry condition），表示從其它模式經由N_i(m)後會到達模式m。X_i(m)是無條件跳出集合（unconditional exit set），表示模式m經由X_i(m)後仍維持模式m。而P_i(m)是狀態恆真式，由N_i(m) ∧ X_i(m)即可得到。

試加說明如下：例如表5中控道B TLU階段，狀態OK的進入模式（mode entry condition Ni(m))為表4中的第二及第三列的聯集(由狀態OK至狀態OK，及Trip至OK)。此處我們為表達清楚之故，將B原始狀態亦放入式中。其無條件跳出集合(unconditional exit set, Xi(m))則為表4中控道在TLU階段由OK至狀態Trip的條件不為真(即表4中第一列的事件)，因而推導出B在TLU中狀態OK的恆真式(即 (Pi(m)) = Ni(m) ∧ Xi(m)) 。

由上表得知其結論為：在單一失誤（A故障）的條件下，另一控道（例如B）的TLU為正常（OK）的恆真式為除了控道A外任兩個控道之DTM為正常，且剩餘一個控道有感應器旁通，或除A外另三個控道皆為正常。而其TLU為跳機（trip）的恆真式為除了控道A外任一個控道之DTM為跳機且沒有感應器旁通。在單一失誤的條件下（A故障），這兩條恆真式皆能符合正

表 5 TLU 階段模式恆真式的產生（以 B 為例）

Mode M	N_i (m)	X_i (m)	P_i (m)
OK	I (State(C) = OK ∧ state(D) = OK ∧ Sensorbypass(B) = true) ∨ (State(B) = OK ∧ State(C) = OK ∧ state(D) = OK) ∨ (State(B) = OK ∧ State(C) = OK ∧ sensorbypass(D) = true) ∨ (State(B) = OK ∧ State(D) = OK ∧ sensorbypass(C) = true)	II ¬ (∃i ∈ Channels (C , D) • state(i) = trip ∧ sensorbypass(i) = false)	I (State(C) = OK ∧ state(D) = OK ∧ Sensorbypass(B) = true) ∨ (State(B) = OK ∧ State(C) = OK ∧ state(D) = OK) ∨ (State(B) = OK ∧ State(C) = OK ∧ sensorbypass(D) = true) ∨ (State(B) = OK ∧ State(D) = OK ∧ sensorbypass(C) = true)
Trip	III ∃i ∈ Channels (C , D) • state(i) = trip ∧ sensorbypass(i) = false ∨ State(B) = trip ∧ Sensorbypass(B) = false	IV ¬ (State(C) = OK ∧ state(D) = OK ∧ Sensorbypass(B) = true)	III ∃i ∈ Channels (C , D) • state(i) = trip ∧ sensorbypass(i) = false ∨ State(B) = trip ∧ Sensorbypass(B) = false

常系統運作狀況，換言之，此設計符合單一失誤準則。

5 線上偵錯及其他

除了上述正規規格及符合法規要項的論證外，我們亦採用了其它軟體發展階段的正規/半正規方法以完成一完整的安全個案。我們採用的方法包括：進一步由 VDM 規格正規轉換為程式碼、發展以規格為基礎的測試、以及測試結果之預測 (test oracle)[6] 來提供安全個案中軟體正確之證據。我們以等價分割 (equivalence partition) 方法及邊界值產生作測試資料，並實作 Petri Net 制定的需求規格以做為測試結果預測(test oracle)。

安全關鍵之軟體多數運作在嵌入式 (embedded) 即時 (real time) 系統中，因此軟體執行時的行為會受到環境、硬體及使用者的影響，而正規方法亦不能完全保證程式執行時安全無誤。因此一個完整的安全個案應進一步包含插入失誤 (fault injection) 測試評估及線上(run time) 偵錯的證據，以確保系統的安全。

一般軟體之插入失誤 (fault-injection) 包含儲存區的資料壞損(例如暫存器或磁碟機中的資料壞損)、網路通訊資料的壞損以及軟體的缺失。在我們的案例中，邏輯錯誤除外下，可能插入的錯誤包括 DTM 中的設定點 (set point) 資料壞損；其他控道傳輸來的訊號 (即 DTM 結果) 由 0 變 1 或由 1 變 0；DTM、TLU、OLU 階段間的傳輸錯誤引起的正常訊號變成跳機訊號，或反之；再者是感應器輸入及旁通輸入之錯誤。

我們將進一步以線上斷言 (run time assertion) 的方式加以偵測防衛上述可能環境及硬體所引發的動態錯誤。所謂的斷言就是邏輯關係式子，用以檢查軟體執行到某一指令時的正確狀況。以領域為主的斷言相當稀少[8]，我們定義了五種符合此領域的斷言類型：

1. 有效範圍檢查 (range assertion)。
2. 恆真式 (invariant) 檢查。
3. 安全性 (safety) 檢查。
4. 相依性 (dependency) 檢查。
5. 三階段間 (DTM / TLU / OLU) 一致性關係檢查。

分述如下：

1. 有效範圍檢查：

規定軟體中每一個變數的有效範圍，預防因範圍外的變數值所導致的錯誤。例如：

- DTM state == (OK | trip)
- ReactorP == (0≤ReactorP<32767) 等。

2. 恆真式檢查：

規定在軟體的任何一個時間中，都必須滿足、不得違反的條件。例如感應器及控道旁通訊息只能有一個控道被設為真。

3. 安全性 :
規定安全的條件，例如四個控道中必須有三個以上為正常才能正常運作（即四選三功能等）。

4. 相依性 :
規定一些有相關聯性的條件，任何不合理且不連續的行為都可以被偵測出來。例如 :
- TLU state = OK ⟹ OLU state = OK
- 控制棒的位置越高 ⟺ 水溫越高

5. 階段間一致性關係 :
此系統有三個階段（包括輸入），各階段間的資料傳輸必須具有一致性。例如 :
- DTM_a output = TLU_a input
- 4 個 DTM all OK ⟹ OLU_A output = OK
- 3 個 DTM OK ⟹ OLU_A output = OK

斷言可插入程式碼中與程式碼同時執行，亦可由另一個斷言處理器 (assertion processor) 來執行。當執行到斷言，處理器會自動檢查這些關係式，如果檢查有錯誤就表示系統可能受到環境影響或是 EMI 干擾等因素，導致系統發生錯誤。如此此軟體由發展過程到線上執行皆有嚴謹的證據，以確保系統的安全性。

6 結論

本文發展以正規方法應用在安全個案的論證建構上的方法及案例。現行的安全個案多採用較主觀的論證，不夠嚴謹，說服力亦不強。本研究應用正規規格為基礎；以論證系統符合法規安全要項的技術，對安全關鍵計算系統的審查作業最有助益，可擴大使用範圍。除了單一失誤準則外，尚可考慮用以討論系統設計是否符合獨立性(independence)，可測性(testability)及深度防禦 (defense-in-depth) 等法規要求事項。而正規方法應用於法規審查的作業為正規方法的一新應用，值得進一步探討。

誌謝

本研究承臺電及國科會之贊助，計劃編號 NSC 88-TPC-E-155-003.

參考文獻

[1] Derek Andrews and Darrel Ince, "Practical Formal Methods With VDM", McGraw-Hill, 1991.

[2] P. G. Bishop and R. E. Bloomfield, "The SHIP Safety Case Approach", SafeComp'95, pp. 437-451, October 1995.

[3] Constance L. Heitmeyer, Ralph D. Jeffords, and Bruce G. Labaw, "Automated Consistency Checking of Requirements Specifications", *ACM Trans. On Software Eng. and Methodology*, (5) 3, pp. 231-261, July 1996.

[4] Ralph Jeffords and Constance Heitmeyer, "Automatic Generaton of State Invariants from Requirements Specifications", SIGSOFT'98, pp. 56-69, Nov 1998.

[5] Nancy G. Leveson and Janice L. Stolzy, "Safet Analysis Using Petri Nets", *IEEE Transactions on Software Engineering*, Vol. SE-13, No. 3, pp. 386-397, March 1987.

[6] D. Peters and D. Parnas, "Generating a Test Oracle from Program Documentation", Proc. of International Symposium on Software Testing and Analysis, pp. 58-65, Aug. 1994.

[7] J. L. Peterson, *Petri Net Theory and the Modeling of Systems*, Englewood Cliffs, Prentice Hall, 1981.

[8] S. Yih and J. Tian, "Developing and Checking Prescriptive Specifications for Safety Improvement", *Microprocessors and Microsystems*, Vol. 21, Issue 10, pp. 585 -593, April, 1998.