

本體論技術於智慧型家庭監控系統之應用

Ontology-Based Modeling of Security Control for Smart Home Application

劉智漢 李蔡彥

國立政治大學 資訊科學系

Email: andyliu@gate.sinica.edu.tw li@nccu.edu.tw

摘要

一般而言，監控系統通常都是為某種設備或環境所寫出來的特用系統；即便是有圖控程式來協助系統的開發，也只能做到縮短系統開發的時間及簡化系統開發的流程，並不能讓已經完成的邏輯被一再地重複利用。因此造成每一次重新佈建系統或更改設備時，就必須要將程式或圖控作調整，以符合新的需求。這種作法使得程式設計師必須要一再地因應外在需求的變動而修改程式，不僅造成系統維護不易，也使得新設備新功能上線的時間加長。本篇文章嘗試在監控系統中導入 ontology 的概念，將應用程式邏輯從系統程式中抽離出來，結合 OWL[1] 技術來描述監控設備的功能、特性以及程式邏輯，讓伺服器端的軟體（本文稱為 web panel）能夠解讀使用者設定的邏輯，進而自動執行監控功能。如此透過 OWL 技術，使用者只需專心描述不同設備的屬性、功能，就能夠讓 web panel 伺服器瞭解這項設備的屬性、功能並執行設定的動作。因此，使用者日後若有其他的監控系統希望協同監控時，也可以透過相同的技術來瞭解該設備，以求設備組態及應用邏輯能從系統程式中分離。

關鍵詞：Ontology, Security Control, Smart Home

1. 簡介

現有的監控系統大多是依據監控設備的特性及使用者的需求去開發軟體。因此，程式設計師多依據不同設備的特性，開發各種適合當時流程及設備特性的監控系統。例如，機房環境監控系統、電力監控系統等。每套不同的監控系統都各自有各自不同的程式撰寫方式及邏輯，這些適合特定環境及設備的程式很難在日後的新系統中被重複使用。即便是採用相同的監控設備，由不同的程式設計師來設計，還是會有不同的程式邏輯流程及撰寫方式，這對日後程式的修改及維護常常會造成非常大的困擾。有可能因為人員異動或是時間久遠，原始系統開發者已經不知去向或忘記當時的寫作動機及完整構思，以致於無法再維護程式。因此，本文的著眼重點便在於如何抽離出寫死在程式中的設備功能及組態、應用邏輯及反應流程，改用 ontology 方式描述，讓程式邏輯能夠從軟體黑箱中被抽離出來，以便使用者能夠用簡易的描述及理解的概念來記述這些邏輯，進而

讓監控系統的核心程式能被重複使用，程式設計師只需專注在核心程式的維護；至於反應流程及邏輯的部份，就交由使用者去定義。我們希望如此便能有效的減少系統開發及維護修改所需要的時間及成本。

1.1 本篇文章的貢獻

本篇文章提出將設備能立即應用邏輯從系統軟體中抽離出來，透過 ontology 的描述方式來描述感應器的功能、反應流程及監控邏輯；同時讓使用者能夠很方便的透過 ontology 描述檔來表達他們想要達成的監控流程。為了達到此一個目標而實作出一個稱為 web panel 伺服器，用以處理及執行使用者定義出來的 ontology 描述檔，以便這些 ontology 能夠被 web panel 解析並轉化為真正的監控流程；進而根據這些流程來啟動／監控監控區域的各個感應器及感應器回傳的訊息，並依據監控 ontology 事先定義的反應方式，對不同的訊息做出不同的回應及通知。雖然 ontology 目前已經被許多領域的專家學者使用，但在監控的部分，卻較少有研究導入 ontology 的觀念來將監控流程抽離出監控軟體，讓監控流程更加具有彈性及更容易客製化。因此，我們希望以 ontology 方式結合傳統的監控系統，能為傳統的監控系統帶來新的研究方向及思維。

2. Ontology 本體論介紹

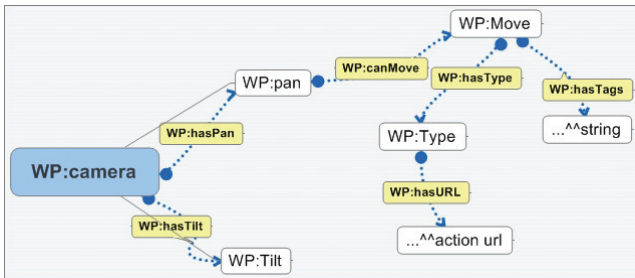
Ontology 一詞源自哲學上的「本體論」，是一套用來解釋存在意義的理論。由於近年語意網路的興起，讓 ontology 在電腦界的應用也日益增多，例如：語意部落格編輯器(semiBlog Editor)[2]，可以讓使用者輕易的在企業內部交換電腦中的資訊；也有使用 ontology 作為監控恐怖主義份子網路訊息的預警系統[3]。由此可見，ontology 的應用在現今的電腦界中可說是非常廣泛。

雖然 ontology 一詞在近年來被各領域的研究人員熱烈的討論，但 ontology 的精確定義及如何建構、如何表達都尚未有一個標準。不過，ontology 嘗試建立統一、共享、可重複使用資訊的精神卻是非常適合拿來應用在監控領域。

總括來說，ontology 為特定領域提供一套共享及可重複使用的知識分類架構，可以用來做為人與機器、機器與機器相互溝通的媒介。透過 ontology 可以將複

雜、需要推理、或是非系統化的資訊以明確的方式表達出來。也由於 ontology 具備上述的特性，因此我們此次的研究便採用 ontology 作為 Web Panel 描述檔的核心，以求得人與機器，機器與機器互相溝通的效能以及資訊完整性。

舉例而言若從監控鏡頭的功能面來看，鏡頭的功能基本上可以分為：是否可以上下(tilt) 及水平(pan)移動，根據這兩個描述功能，就可以用 ontology 將鏡頭功能設計出如下面的架構圖：



圖一 鏡頭 ontology 的架構圖

依據上面的架構圖，我們就可以設計出鏡頭的 ontology 描述檔，用以描述鏡頭的基本功能。

```

.....
<owl:Class rdf:ID="camera">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="hasPan">
          <rdfs:domain rdf:resource="#camera" />
          <rdfs:range rdf:resource="#pan" />
        </owl:ObjectProperty>
        <owl:ObjectProperty rdf:ID="hasTilt">
          <rdfs:domain rdf:resource="#camera" />
          <rdfs:range rdf:resource="#tilt" />
        </owl:ObjectProperty>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="pan">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="canMove" />
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="tilt" />
<owl:Class rdf:ID="Move" />
.....

```

圖二 鏡頭設備描述 ontology 部分內容

在圖二中，我們節錄出 ontology 描述檔的一部份內容，其中<owl:Class rdf:ID="camera">...</owl:Class>表示這是個符合 OWL 語法的 class。Class 是 OWL 語法中最基本的元素[4]，也是整個樹狀架構的頂點(root)。這個 class 主要是用來描述鏡頭功能的相關資料。由此檔案內容來看，鏡頭包含兩個主要的特性：hasPan 與 hasTilt。這表示鏡頭可以有上下及左右移動的功能。在<owl:Class rdf:ID="pan"> ...</owl:Class>這

個區塊的資料中，<owl:ObjectProperty> 裡面就指出這個鏡頭的 pan 是可以被移動的 (canMove)。同樣的，使用者也可以根據實際的狀況來描述 tilt 是否有上下移動或其他的相關功能。

當我們訂定這個架構後，使用者就可以輕易的透過 ontology 描述檔與其他的監控系統分享資料，達到 ontology 資料共享的目的。傳統的監控系統在這個部份的作法，可能採用自訂格式的文字檔或是資料庫來記錄這些訊息。其他的系統若想要讀取這些資料，可能必須要事先了解自訂格式的文字檔內容或是透過資料庫授權的方式，才有可能達到資訊分享的功能。這不僅造成管理上的困難，同時也增加程式撰寫的複雜度及資料安全上的問題。

3. Web Panel 簡介

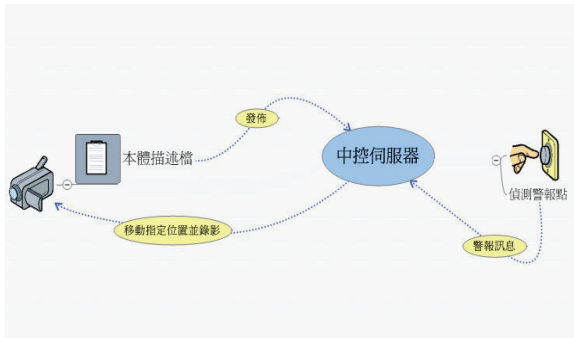
為了要讓使用者定義的 ontology 描述檔能夠被程式轉譯並且執行，我們實作了一個名為 web panel 的伺服器來達成這個目的。web panel 伺服器使用 ontology 作為監控流程的描述檔，因此使用者可以自行定義感應器類別及監控感應器所需要的流程。定義完成後，便可以將此 ontology 以 SOAP 的方式透過 HTTP 通訊協定發佈到 web panel 伺服器，web panel 中的 XML/RDF(s) 解譯模組即會開始處理使用者傳送過來的 ontology 描述檔，建立監控流程並將監控設備備的硬體功能對應到系統中的功能函數，啟動使用者定義的監控流程，完成整個解譯動作，並交由其他模組接手監控種個架構的設備及感應器，並對監測結果作出適當的反應。

目前 web panel 採用的是 client-server 架構，在 Linux 環境下由 java 所寫成，以便日後能夠移植到其他平台使用。

3.1 情境描述

以下我們將以新設備如何加入以 ontology 為基礎的 web panel 伺服器作為範例，來說明使用者要如何設計 ontology 監控流程描述檔。

假設我們希望在 web panel 伺服器架構中加入新鏡頭設備並希望架構中的大門感應器被啟動警訊時，此鏡頭會自動移動到指向大門的位置同時啟動錄影直到警訊解除。為了能夠達到上面的功能，使用者首先需要將此鏡頭的 ontology 描述檔寫好並發佈到 web panel 伺服器，web panel 伺服器在接收到此描述檔後，即會開始解析此設備的 ontology 描述檔，看看此鏡頭所擁有的功能，並啟動相關功能進入監控流程。因此，若大門的感應器發生警訊時，web panel 即會參照事先建立好的鏡頭功能，對鏡頭送出控制命令，將鏡頭移到使用者事先設定的位置（即正對大門），並啟動錄影程序直到警報解除。



圖三 Web panel 監控流程

由上面的描述來看，此鏡頭之 ontology 描述檔必須要包含：監視鏡頭的名稱、URL、功能特性（上下/左右移動、Zoom-in/out、及光圈/焦距等）、以及是否要與某個感應器作連動（此範例為：若大門的感應器偵測到門被打開，則移動鏡頭照向大門，並啟動錄影功能）。為了達到這一個功能除了鏡頭 ontology 外，使用者還必須要建立關於大門感應偵測器的 ontology 描述檔部分，如圖四。Baroswitch 這個 class 描述關於大門感應器若是遇到 Alarm 發生時，應該如何反應以及解除 Alarm 時所應該做的動作。由 ontology 描述檔中可以看到，若感應器發生 alarmOn 時，web panel 伺服器就會透過 `<owl:Class rdf:ID="alarmOn">...</owl:Class>` 這個區段定義的內容來執行移動鏡頭的動作。

```

<!-- 大門感應器 -->
.....
<owl:Class rdf:ID="baroswitch">
  <rdfs:subClassOf rdf:resource="#wp">
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="hasAlarm">
          <rdfs:domain rdf:resource="#baroswitch" />
          <rdfs:range rdf:resource="#alarmOn" />
        </owl:ObjectProperty>
        <owl:ObjectProperty rdf:ID="releaseAlarm">
          <rdfs:domain rdf:resource="#baroswitch" />
          <rdfs:range rdf:resource="#releaseAlarm" />
        </owl:ObjectProperty>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="alarmOn">
  <rdfs:subClassOf rdf:resource="#baroswitch">
    <owl:onProperty>
      <owl:ObjectProperty rdf:ID="moveCamera" />
    </owl:onProperty>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="moveCamera" />
.....

```

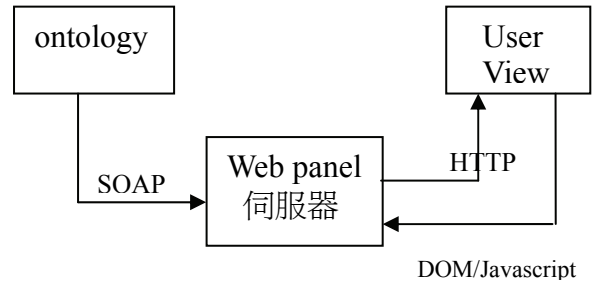
圖四 大門感應器的 ontology 描述檔範例

如果每個監控伺服器都能夠接受以 ontology 為基礎的設備描述檔，則要讓監控伺服器能自動查詢彼此的設備及功能，將會是一件容易被達成的事。因為 ontology 原本的設計就是為了要讓資訊能夠輕易的在不同的系統中交換。例如，當警察要透過散佈各地的路口監視器來追蹤搶匪的行蹤時，監控伺服器可以透過查詢，向其他負責此機控區域的伺服器查詢並了解某個路口有哪些監視器，分別由哪些監控伺服器控制，蒐集到

這些資訊後，警察就可以透過監控伺服器立刻知道有那些鏡頭可以照到該路口，並讓這些鏡頭轉向該路口，以獲得現場的即時影像。

3.2 Web panel 系統架構及內部資料流向

圖五是資料訊息的流動方向。左邊是使用者定義的 ontology 描述檔，透過 SOAP 傳送到 web panel 伺服器後，會經由 web panel 內部的 xml/RDF(s) parser 去分析此描述檔，建立內部 process 啟動各個相關服務，並透過使用者介面將監控資料及畫面呈現給使用者，或透過使用者控制模組接收使用者的指令，將指令轉換成相關設備可以接受的 control 命令。狀態偵測模組則可以偵測設備目前的狀態，警訊處理模組則可以處理相關的警訊並與感應器目前狀態傳送給訊息通知模組，該模組會依據使用者事先定好的訊息處理流程做出適合的回應。



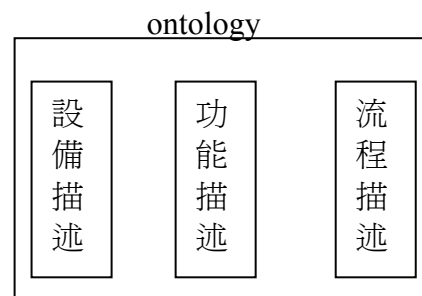
圖五 Web panel 資料流向圖

4. Web panel 設計及實作

以下我們將針對 web panel 伺服器的各項核心功能作一詳細的描述，以便使用者能更清楚整個流程及內部架構。

4.1 Ontology 描述檔

Ontology 描述檔是由使用者自行定義的 XML 文字檔，主要分成三個部分：設備描述、功能描述與流程描述（如圖七所示）。以下將針對這三個區塊舉例作較詳細的說明。



圖七 ontology 描述檔之組成

設備描述主要是用來描述設備的基本資料，如設備名稱、製造廠商、韌體版本等，ontology 的設計內容如

圖六所示。

```

.....
<owl:DatatypeProperty rdf:ID="cameraName">
  <rdfs:domain rdf:source="#camera" />
  <rdfs:range rdf:resource="xsd:string" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="manufacturer">
  <rdfs:domain rdf:source="#camera" />
  <rdfs:range rdf:resource="xsd:string" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="firmwareVersion">
  <rdfs:domain rdf:source="#camera" />
  <rdfs:range rdf:resource="xsd:string" />
</owl:DatatypeProperty>
.....

```

圖六 ontology 設備描述檔範例

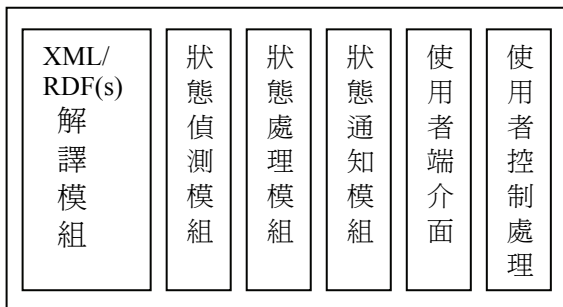
功能描述則是用來描述此設備可以被 web panel 控制或執行的相關功能，如關於鏡頭描述的部分，可以描述此鏡頭是否可以縮放、左右移動、焦距是否可調等，其中也包含這些功能的相關控制命令要如何執行，web panel 的使用者控制模組即是透過此處描述的控制命令來對設備下達控制命令，ontology 功能描述檔的部分則可以參考圖二或是圖四關於鏡頭及大門感應器的內容。

流程描述則是負責描述 web panel 要如何監控設備，以及當設備發生問題時或達到某個臨界值時要如何處理或與其他設備間如何互動的描述。例如，當某設備發生問題時，自己這個設備需要做什麼反應。

4.2 Web panel 核心

如圖八所示，Web panel 核心包含了六大模組：XML/RDF(s)解譯模組、狀態偵測模組、狀態處理模組、狀態通知模組、使用者端介面、使用者控制處理模組。

Web Panel 核心模組



圖八 Web panel 六大核心模組

XML/RDF(s)解譯模組：此模組主要的功能在於檢查使用者傳送到 web panel 的 ontology 描述檔是否符合正確的格式，並將描述檔的內容轉化為 web panel 內部的執行程序供 web panel 執行。

狀態偵測模組：若 ontology 中有需要啟動設備狀

態偵測的時候，XML/RDF(s)模組便會將需要啟動狀態偵測的設備資訊加入到狀態偵測模組中，狀態模組即會根據 ontology 中定義的流程描述部分，對設備的狀態開始進行監控。

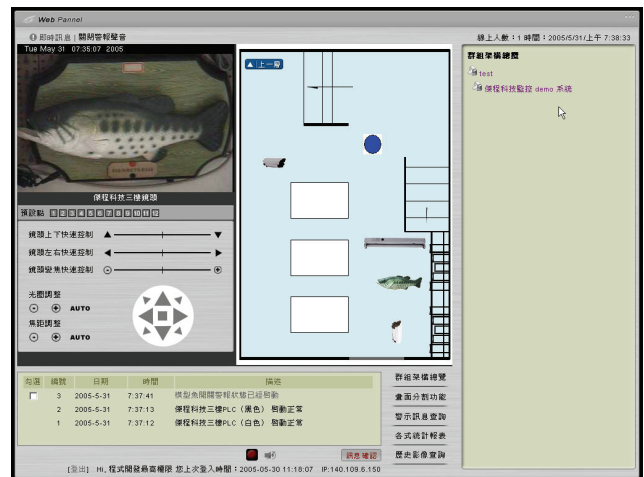
狀態處理模組：當狀態偵測模組接收到設備狀態時，便會將狀態值傳送給狀態處理模組處理，此模組會根據使用者設定的 ontology 流程描述來判斷臨界值是否符合使用者的設定，以決定是否要將反應訊息傳送到訊息通知模組及使用者端介面模組。

使用者端介面模組：主要是將 web panel 偵測到的訊息及設備資訊透過 html 形式呈現給使用者，並可以隨時更新畫面上各項設備的即時狀態，以便使用者隨時上網查看。

使用者控制處理模組：是用來接收使用者傳送過來對設備的控制命令。當有警訊發生時，web panel 伺服器會在使用者介面出現警示訊息。使用者可以點選該訊息並查看內容，同時對該訊息作出處理或解除的動作。若是使用者解除該警訊，使用者控制處理模組便會將解除命令回傳至 web panel 狀態處理模組作出解除的相關動作。

4.3 使用者介面

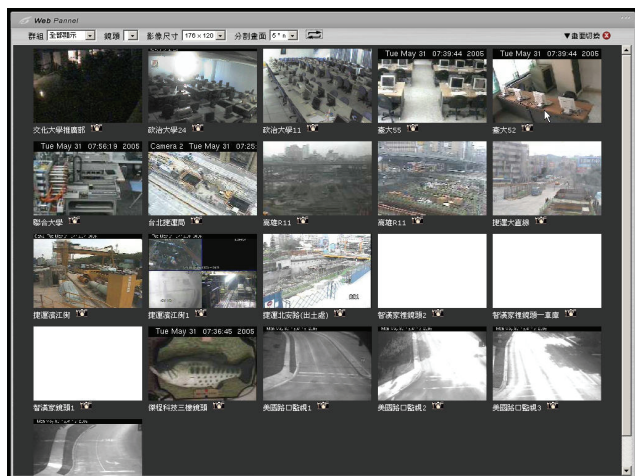
使用者介面是 Client 端的呈現介面，主要使用 html 及 javascript 來作為呈現畫面背後的語言程式。使用者可以透過此介面查看設備狀態、鏡頭影像、或控制設備、檢視報表等功能。



圖九 Web panel Client 端主要執行畫面

Web panel client 端畫面分成左右兩部份，右邊為功能列表項，主要用來顯示功能列表或監控區域使用；左邊分成上下兩部，上方為控制區域的地圖及監控設備的分布圖，包含影像顯示及控制畫面（左），用來顯示影像及控制設備（如鏡頭）的各項功能。設備位置分布圖（右），主要是用來顯示設備在實際區域的相對位置。

左下部則為即時訊息顯示區域，用來顯示系統訊息使用。若警訊發生時，系統會在此區塊顯示警示訊息，使用者可以點選訊息看更詳細的內容，並決定處理方式。Web panel 系統實際運作的監控畫面範例如圖十所示。



圖十 Web panel 鏡頭影像實際監控畫面

5. 未來研究方向

由於 ontology 以及 OWL 的研究仍然在繼續，因此未來不排除會更改 ontology 的設計，以符合標準的語法。web panel 與 user view 的連結部份，目前是使用 javascript + DOM 的方式取得 web panel 伺服器目前的狀態，每隔一段時間由 client 端主動向 server 端取得感應器的目前狀態，然後再解析訊息顯示在網頁上。本系統目前無法由 server 端直接 push 資訊到 client 端；未來可能會針對此點作改進，改由 java applet 的方式來被動接收訊息，server 則在有警訊發生時，才主動通知 client 端顯示相關訊息。這種做法將可以有效的減輕 server 一直接受 client 查詢訊息的負擔。

此外，如何讓使用者能夠輕易的設計出符合規範的 ontology 描述檔對 web panel 來說也是一項挑戰。目前使用者是採用手動 key-in 的方式來建立 ontology 描述

檔，不僅非常耗時間同時也容易出錯，未來也會針對這點，開發客製化的程式介面，讓使用者能容易輸入這些描述檔的內容。

6. 結論

本文展示了如何透過 ontology 描述監控設備的功能以及監控的流程，並與 web panel 結合，讓監控系統的設計流程簡單化。如此使用者只需專注在設備 ontology 描述檔的設計，而不需要管程式設計師是如何設計他們的程式。日後設備或功能有變更也只需要更動 ontology 即可完成系統流程的重設。

Ontology 不是 web panel 伺服器最先使用，但 web panel 伺服器卻希望能夠將 ontology 便利可重複使用的觀念引導進入監控領域中，讓封閉型的監控領域能夠與更容易的與其他領域（如 smart home 等）相結合，為人類的生活帶來更大的便利。

參考文獻

- [1] OWL Web Ontology Language(OWL), <http://www.w3.org/2004/OWL/>, 2004.
- [2] K. Moller and S. Decker, "Harvesting Desktop Data for Semantic Blogging", .in *Proceedings of ISWC 2005 Workshop*, 2005.
- [3] M. Crubezy, M. O'Connor, Z. Pincus, and M.A. Musen, "Ontology-Centered Syndromic Surveillance for Bioterrorism," *IEEE Intelligent Systems*, vol. 20, no. 5, pp. 26-35, Sept/Oct, 2005.
- [4] OWL Web Ontology Language Guide, <http://www.w3.org/TR/2004/REC-owl-guide-20040210/#StructureOfOntologies>, 2004