採用本體論及推論於資料語意偵錯之研究

劉艾華
淡江大學資訊管理學系

王茂年
淡江大學資訊管理學系

林英潔
淡江大學資訊管理學系

摘要

隨著網際網路的快速發展，電子商務儼然已成為傳統店面行銷外的另一個新興通路，由於網站資料呈現的多樣化、不受限制的交易時間及交易成本的低廉化，有越來越多的行業透過網路行銷，但網路交易的糾紛卻時有所聞。鑒於大多數網站其資料皆以人工方式輸入，網站資訊一旦發生錯誤，除使網路行銷者的商譽受損外，甚至可能產生嚴重的虧損。因此業者公佈網路資訊時，如何確保資料的正確與合理，實為未來電子商務的重要課題之一。

為使電腦了解資料所隱含的意義進而執行語意偵錯，本研究透過本體論設計語意推論架構，採用專家系統發展程式語言Jess撰寫規則，可早期發現網站資料隱含的錯誤語意，並有效減少網路交易因語意錯誤而導致企業形象與金錢的雙重損失。

本研究係以國內某大型3C賣場之網站商品建構本體論為實證，依商品類別及商品間關係定義，透過本體論界定其領域與範圍，建立具有階層架構的商品本體論，撰寫語意檢查規則，實證結果顯示：本研究成果使廠商在維護網站資訊時，可大幅減少耗用的人力與時間成本，並可確保網路資訊的正確。

關鍵字：語意網路、本體論、專家系統、Jess、語意偵錯

Detecting Semantic Data Errors By Adopting Ontology and Reasoning

Ay-Hwa Andy Liou
Department of Information Management, Tamkang University

Maw-Nian Wang
Department of Information Management, Tamkang University

Ying-Jie Lin
Department of Information Management, Tamkang University

Abstract

Through the rapid development of internet, website content is experiencing major changes in time and space. An important question is how to maintain the rationality, correctness and integrity of data. Most websites input data manually by the user, but this may result in incorrect data and thus huge losses. Although it is not difficult to syntactically check the correctness of the data on website, verifying the semantic meaning of the data involves complexity. Besides, the time and labor costs on maintaining website information is also increasing.

This research presents a semantic reasoning framework using expert system development program language Jess (Java Expert System Shell) for reasoning under specific ontology. This allows the machine to understand the meaning of data and detect semantic errors of those data. This approach reduces unnecessary losses due to semantic error and decreasing the time and effort on checking data while promoting the work efficiency and business reputation.

Key words : Semantic Web, Ontology, Expert System, Jess, Semantic Detect

壹、緒論

自1990年英國牛津大學物理學家Tim Berners-Lee首次提出全球網際網路資訊網（World Wide Web,WWW）的概念後，網路呈現快速地成長與發展。資訊網雖可提供訊息的交換並成為溝通的媒介（media），但電腦本身並不能判斷所呈現之資訊內容是否正確。隨著網路資源日益龐大，1998年Tim Berners-Lee提出了第二代全球資訊網——語意網路（Semantic Web）的概念，他將語意網路定義為「電腦能瞭解並處理網路資源」[Daconta et al. 2003]。若網路資源能被電腦理解，則可推論出其他的資訊或知識，故「理解語意」在某些層面上亦具有「推理」的涵義[Horrocks et al. 2003]。使用本體論(ontology)定義各個不同領域的背景知識，也提供了完整的辭彙及關係，進而使網路資訊能被明確地定義。運用本體論表達網路的資訊或內容，具有資訊共享（Sharing）、領域的定義（Definition）和推論（Reasoning）等優點[Chandrasekaran et al. 1999]。資訊共享是讓某領域有共同的知識表示方式，讓電腦做到真正的語意溝通；領域定義是對某領域之背景知識的相關辭彙加以定義，並將辭彙加以分類，呈現階層化關係；推論則是透過分類與關係，挖掘出隱含的知識。

隨著科技的發展、網際網路的興起，導致資訊快速地膨脹。多樣化的資訊內容隨時間與空間的轉變，需要經常更新以確保網頁的正確、合理及完整。網頁內容的建置、儲存、傳遞與管理必須依賴電腦和網路技術的發展，龐大的資訊量增加了網路使用者與網頁管理者對於網路內容的搜尋、存取、呈現、維護與整合的困難度。因此，語意網路（Semantic Web）的發展讓網際網路具有語意判斷的能力，從而使網路的發展邁入新的紀元。

一般網站內容若發生文數字顯示錯誤的情形，至多給人一種網頁管理不善的觀感，但若網頁內容的文字或數字具有交易或重大特定意義存在時，維護文字或數字正確地呈現則相對極其重要，如：網路上的商品資訊其售價是否符合目前市面的行情，也就是符合大眾認知的合理價位即為一例。2004年4月，日本雅虎購物網站電腦商KATENA將蘋果電腦eMAC價格標為原價的四十分之一，公司旋接獲兩萬名顧客下訂一億台電腦，發現錯誤後，該公司向兩萬名顧客發出電子郵件取消訂單，並在網站上再三強調因價格標示錯誤無法接受訂貨；國內大型購物網站PChome online（網路家庭），2006年4月因數位相機價格標示為原價的十分之一，使消費者大量下單，PChome online聲明因價格少一個0無法出貨、以送贈品的方式予以補償；2006年聖誕節前夕適逢週末，英國HotUKDeals網站公佈：凡透過線上購物者，皆可享有Hamleys toyshop公司累計達60%的折扣，導致短時間內顧客瘋狂透過網路交易，將Hamleys toyshop店內耶誕禮品搶購一空，造成公司嚴重虧損，專家並以“consumer hackers”稱呼這些網購者。許多企業或購物網站對於價格網頁雖投入相當多的時間與人力維護，企圖找出可能潛存的錯誤，但從層出不窮的重大交易糾紛可知，此舉未必有效。

若企業將資料賦予語意，使電腦能理解並自行檢查資料的正確與合理性，不但可有

效節省人員投入偵錯的時間成本，也可將可能產生的嚴重損失降到最低。由於目前網路資料仍缺乏語意使電腦無法理解資料意涵，本研究即嘗試祛除建立資料時可能發生的語意錯誤，並大量節省檢查資料時所耗費的時間與人力。透過合作之大型3C賣場的網站資料，建立網路內容詳細描述的本體論，撰寫檢查語意錯誤的規則，研究結果顯示：本研究所使用方法，確能有效達成語意偵錯的目的，使電腦了解資料隱含的意義、自我檢查資料語意可能的錯誤、維護網頁內容的正確並縮短檢查資料時所需之時間。

貳、文獻探討

一、語意網路（Semantic Web）

（一）語意網路概述

語意網路為第一代網路的延伸，目的在改變我們對網路資源的使用方式，藉事前定義資訊內涵，使資訊得以被有效率的搜尋與應用，並使電腦能理解網路資源所隱藏的意涵，達到網路自動化處理的目的。語意網路建構於現有的網路環境以執行網頁服務與內容管理，使用具有完善表達能力的語言描述資源，提供機器可判讀與了解的技術[Ruckhaus & Vidal 2003]。

語意網路是網路未來的願景[Daconta et al. 2003]，定義網際網路的資源、將關聯的資源連接成網，使機器可理解並使用。不僅連結現有網頁，對於各種相關的資源，均可提供自動化操作、整合及重複利用。

Tim Berners-Lee對未來的網路提出兩種觀點：第一、讓網路成為協同合作的媒介。其次，使網路可被機器了解及處理[Daconta et al. 2003]。他並建構網際網路的未來方向——語意網路，使網路資源的語意內容，可由機器自動處理並引導網路meta-data的使用[Ding 2001]。

（二）語意網路表示方式

1. XML

由於HTML的目的在於呈現網頁內容給使用者，而不是用來紀錄、描述資料的內涵，不能有意義的標示語言，因此沒有邏輯上的意義。加上各大瀏覽器廠商在HTML標準規格之外推出自己的標籤，造成不相容，致XML應運而生。

Extensible Markup Language（XML）為W3C（全球資訊網聯盟）於1998 年二月發布的標準，XML是用來建立描述結構化資料標示的語言，使用者可定義自認為有意義的標籤，並自行建立標示語言的文法或規則。XML具備跨平台資訊的自動轉換、整合、查詢與處理的能力，故可達到「資料分享」的目的。XML具有唯一的根節點，必須以XML文件宣告為起始。其應用有延展性，是可以自訂標籤的開放性語言。

2. RDF/RDFS（RDF Primer 2004）

RDF（Resource Description Framework）是用於表達W3C資源的資訊語言，用簡單的

屬性（property）及屬性值來描述資源，因此RDF可用於表達任何有關Web上被標識的事物的資訊。RDF有三個主要的資料模組：即來源、屬性與敘述。

（1）來源(Resources)：用URI定義的物件就稱為資料來源。

（2）屬性(Properties)：描述資料來源的特徵。

（3）敘述(Statements)：資料來源由三重屬性(triple)組成。

敘述的語句以RDF的格式Subject、Predicate和Object表示。如圖1所示。

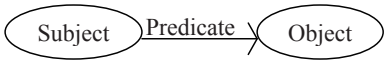

圖1：Triple

其中Subject表示資料來源，Predicate表示屬性，Object是文字或是其他來源。

RDF無法針對特定應用定義類別與特性，須透過RDF辭彙描述語言：即RDF Schema（RDFS）來定義。RDFS提供RDF的規範以供本體論使用，再使用RDF表示本體論的實例。但因RDF Schema對關係型態的描述不夠豐富，而後產生DAML+OIL和OWL。

3. OWL（Web Ontology Language）

OWL是W3C推薦的本體論描述語言[OWL Web Ontology Language Reference 2004]，是由DAML+OIL web 本體語言合併DAML+OIL 的設計與應用之後的修訂版。在語意的表達上，OWL的能力比XML、RDF或RDFS更好，因此OWL在陳述網際網路機內容的能力超過這些語言。DAML的設計是以RDF為基礎，可作為推論語言的撰寫和本體論的建立。OIL是第一個結合敘述邏輯元素的本體論語言，以XML和RDF的方式呈現。而OIL提供良好的對映，藉由SHIQ Description Logic呈現語意[Horrocks et al. 2003]。OWL以XML為基礎，仍使用RDF的語法，而OWL補足RDF Schema缺乏的類別描述關係。OWL依據表達的能力和推理能力再細分三種子語言，分別為OWL Lite、OWL DL和OWL Full三種，OWL Lite的表達能力最弱，適用於描述本體類別較簡單的階層結構。OWL Full表達能力最強，包含所有OWL語言所提供的限制及RDF的使用語法，但OWL Full表示的本體論無法進行推論。OWL DL表達能力介於OWL Lite和OWL Full之間，適用於需要在推理系統進行運算。

（三）語意網路相關研究成果

透過語意網路，從網路上彙整一組已知名稱及其別名當作訓練集，萃取詞彙模式後，用搜尋引擎發出一些與已知名稱相關的部分訊息，得到該名稱的「別名候選集」(Candidate Aliases)，透過原文及超連結等方式，設計字元同時發生模式並依其名稱與候選別名之關聯性的高低給予評分，統計的顯著性高達0.6718[Bollegala et al. 2008]。欲有效的管控語意網系統，須對語意網服務程序實施檢查與監控。學者提出複合式程序(Composite Process)檢查的概念，即透過OWL-S在執行後分析事件時，應同時包含初始事件的偵錯[Vaculín et al. 2007]。

二、本體論 (Ontology)

（一）本體論概述

本體論以樹狀結構及關聯方式對所有的事物做分類，描述事物之間的關係，並以結構化的方式表達知識。在人工智慧的領域上，主要是將人類腦中的知識建構成知識庫。Gruber定義ontology為「對概念的詳細外顯描述」[Gruber 1993; Gruber 1995]，藉由對概念的共同描述，讓不同領域的人有相同的認知，也能讓機器了解人類所要表達的語意。

本體論定義知識領域基本的辭彙（Term）及字彙（Vocabulary）之間的關係，俾描述知識、對知識作推論並將知識重複使用[Noy & McGuinness 2001]。語意網路就是讓機器用「智慧」的方式處理網路資訊。對於網路物件、文字的內容描述，可經由本體論的分類與表示的關係，提供機器判斷網路資訊內容隱含的意義。Berners-Lee應用本體論於語意網路的觀點是：傳統的本體論是一種分類法則，定義類別與類別間的關係，並須具有一組推論規則支援知識推論。在網路上本體論是公開的、知識可重複的使用以達共享目的，智慧型代理人可在不同的系統或平台處理網路資訊。

（二）本體論組成要素

建構本體論的組成要素分別為：classes、slots、facets、instances[Noy et al. 2001]。classes是一個正式且明確的描述某個領域知識的概念，亦稱concepts，如本研究中的商品或電視等都是classes的一種，若再依商品的功能或屬性細分成資訊商品及家電商品，則這兩個子分類就是「商品」的subclass。slots描述classes和classes之間的關聯性或概念的屬性，亦稱properties，例如商品名稱、商品價格。電視類的商品有映像管電視、液晶電視、電漿電視等子類別，這樣的classes和subclass之間的關聯也算是一種slot。facets是slots的限制，或稱為role restrictions，如商品名稱只能為字串、商品價格僅能有一個等。每個instances為class的一個實例，且繼承其屬性及關聯。

（三）建造本體論之目的

本體論定義某領域知識共通的辭彙，使該領域的人可共享資訊，建構本體論的理由如下[Noy et al. 2001]：

1. 人與軟體代理人間共享相同的資訊，此為發展本體論常見的目的[Musen 1992；Gruber 1993]。如不同的網站都有醫學與醫學電子商務服務，若網站共用相同的本體論中的辭彙，則代理人可在不同的網站中擷取、收集資訊，用這些收集來的資訊答覆使用者查詢。
2. 近年來本體論深入探討領域知識的「重複使用」問題。通常不同領域可能有相同要表示的知識，如時間或時間的間隔…等，故建立本體論時，知識可能被重複使用於不同領域。因此如需建構完整的本體論，可以先整合各領域本體論的內容。
3. 可詳述領域的概念。領域概念有時難以用程式語言表達或修正，本體論可以清晰地表達領域概念，當領域概念改變時，吾人可以輕易修改領域知識的概念。本體論能詳述領域知識，新的使用者可以快速的認知領域知識的意涵。
4. 本體論可重複使用並延伸其知識，一旦建立某領域知識的辭彙後，隨即應分析領

域知識的正確性[McGuinness et al. 2000]。

（四）領域本體論之架構

架構某領域的本體論，必須對該領域加以分析[Navigli et al. 2003]，分析重點包括：

1. 檢查描述實體的字彙。
2. 字彙中的term必須有描述的格式，此格式可將term轉換成概念、關係或概念的實例。
3. 描述term之間的概念關係。

領域專家定義領域的本體論並建立知識的架構，圖2是用三階層的架構建構領域的本體論。

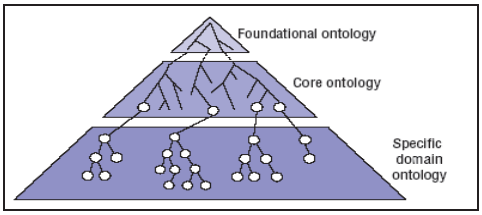

圖2：特定本體論的三階層表示圖

最上層是基本本體論（Foundational Ontology），由哲學家和人工智慧專家定義，這層的本體論和領域本體論並無關係，僅建立一般性的本體論以確保不同領域的再利用。中間層是核心本體論（Core Ontology），是由知識工程師依據該領域的知識而建構，故此階層可能有數百個應用領域的概念。最底層是特定領域的本體論（Specific Domain Ontology），針對個別需求定義的本體論，故本階層的本體論與其他領域知識的觀念可能不盡相同，也有使用限制[Navigli et al. 2003]。

（五）本體論相關研究成果

商業交易過程中廣泛使用XML作資訊交換，但以往並未使用通用的程序管理資料編排。利用整合的XML文件提出架構，將偵錯、日誌、資料探勘…等過程逐一紀錄下來，藉語意網層層定義並建立XML輪廓，允許使用者從自動整合的XML中檢索資訊。將定義在XML的知識構成本體論，使末端使用者可以搜尋和檢索資訊[Cruz et al. 2008]。在資料品質管理方面，特定知識中常潛藏無效資料，只有領域專家方能清除。學者主張用本體論描述特定領域的分類與屬性，屬性值以有效或無效標記，在資料清理方面，偵錯並移除語意錯誤的資料、透過演算法分析無效變數值組(tuple)的替換頻率、建立替換分析的規則，以減少對使用者的影響[Brüggemann 2008]。

三、專家系統

（一）專家系統簡介

專家系統是將專家知識與經驗建構在電腦上，使成為具推論能力的電腦系統，目的

在模擬人腦功能、使電腦擁有專家的知識，以類似專家解決問題的方式替代或輔助專家，對特定領域給予建議或解答並解釋推論的結果。專家系統亦稱為智慧型知識庫系統（Intelligent Knowledge-Based Systems, IKBS）[Nault & Storey 1998；Speel et al. 2001]，可提供輔助解決問題， 。

第一代專家系統於1950年代即已開發出來，其他專家系統隨後陸續建立，但多屬研究雛型系統。隨程式語言與軟體工具漸次成熟，各種知識表示法被廣泛研究，第一個醫學領域的專家系統MYCIN[Shortliffe 1976；Davis et al.1977] 成功誕生，該系統可針對腦膜炎、細菌感染等症狀，提供知識擷取、功能解釋及智慧型的輔助功能，奠立人工智慧發展的基礎。

典型的專家系統包含六個要素：knowledge base（知識庫）、inference engine（推論引擎）、working memory（工作記憶體）、explanation facility（解釋功能）、knowledge acquisition facility（知識擷取功能）、user interface（使用者介面）。專家系統的結構如圖3所示[Giarratano & Riley 1998]。

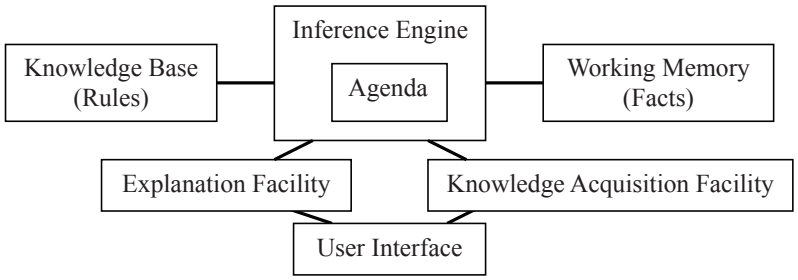

圖3：專家系統的組成結構

（二）Jess

Jess（Java Expert System Shell）是1995年美國Sandia 國家實驗室的Ernest J. Friedman-Hill，以Java為藍本、CLIPS（C Language Integrated Production System）的原理為基礎而擴充的版本[Jovanovic et al. 2003] ，添加新的method使用於Java程式中。除繼承了CLIPS的優點，同時具備獨有的特徵，如：支援正向和逆向推理、可在系統環境下直接存取Java函式庫等，並應用於不同的網路機器中。除CLIPS的功能擴充外，Jess並直接繼承Java語言物件導向的特性，使Jess成為功能更完整的專家系統語言[Friedman-Hill 2003]。

上述Expert System Shell意指具推論引擎、解釋功能和知識擷取功能，套用不同的領域知識即成為不同領域的專家系統[Negnevitsky, 2002]。目前大部分的規則引擎（rule engine）都可視為Expert System Shell， 其組成如圖4所示。

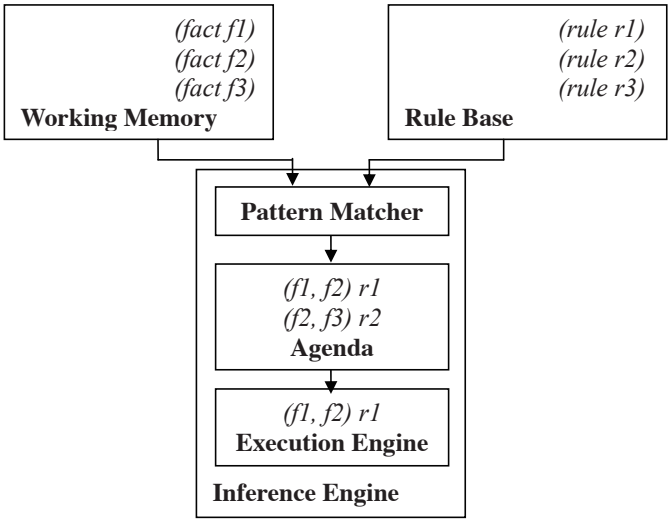

圖4：規則引擎的組成

規則庫存放專家的知識和經驗，為使執行的過程更有效率，規則引擎中包含規則編譯器（Rule Compiler）。在Jess中的Rule Compiler使用Rete演算法， Rete只針對網路中有變動的部分作搜尋，故執行效率佳，但Rete佔用的空間較多，可視為以空間換取時間的演算法[Friedman-Hill 2003]。Jess同時擁有向前推論（Forward Chaining）與向後推論（Backward Chaining）的功能，使專家系統在推論時更有效率。工作記憶體（Working Memory）存放推理過程中相關的資訊。推論引擎對規則庫和事實庫的資料進行比對，找出符合條件且能被activated的規則，找出待執行的規則後決定何者優先被執行，Conflict Set藉Agenda排序，最後Execution Engine執行Agenda 的Fire規則，直到全部的規則執行完畢。

叁、研究方法

國內外購物網站標錯售價的情況屢見不鮮，前述之日本雅虎、PChome Online及英國HotUKDeals等知名網站均曾發生。但Hamleys為保全企業聲譽，除對消費者致歉外並認賠所有訂單，損失極為慘重。

從以上的案例可知：購物網站公佈正確的商品資訊極其重要，資料語法或許正確，但若資料語意有誤，可能不易查出。本研究採用本體論作為網頁資料維護的基礎，針對網站資料內容撰寫語意檢查規則，確可提升網站內容的正確與合理性。

一、系統架構

本研究使用Java語言建置語意偵錯系統，透過Protégé平台建置商品本體論，採用Jess語言撰寫檢查規則，藉由JessTab整合建構於Protégé平台之本體論與檢查推論規則，透過

Java呈現簡潔的使用者及規則介面。本研究的系統架構圖如圖5。

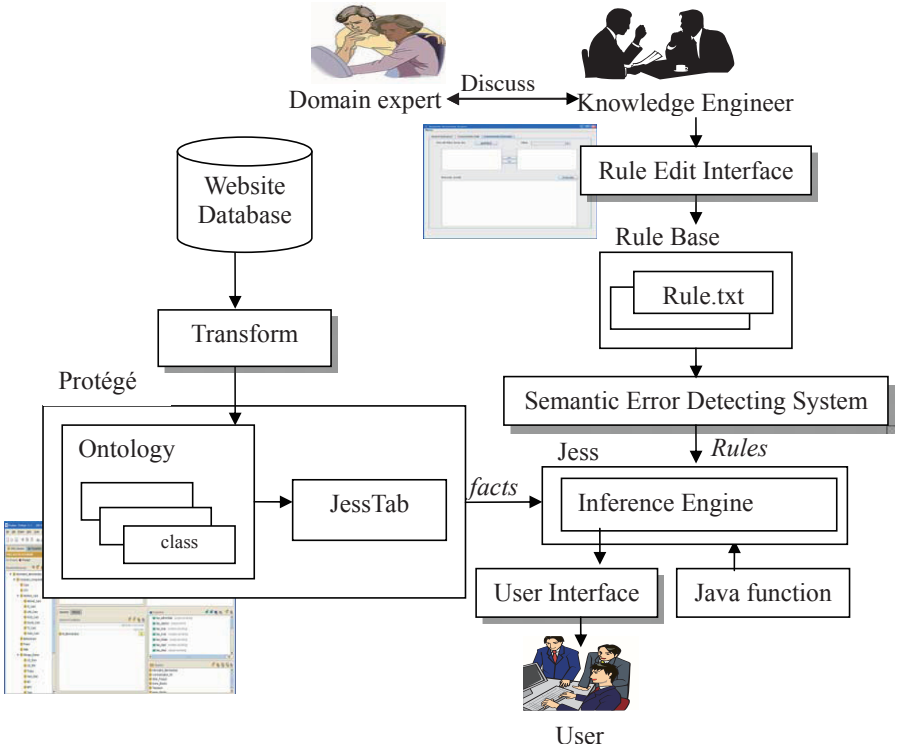

圖5：系統架構圖

本研究的系統架構分成三個部分：Rule Base（規則庫）、Inference Engine（推論引擎）及Interface（介面）。本研究設計首將網站資料轉換成ontology的階層關係，知識工程師（Knowledge Engineer）和領域專家（Domain Expert）討論資料語意可能發生錯誤並影響重大的部分，由知識工程師撰寫規則建立規則庫、透過語意偵錯系統將規則送至Jess推論引擎，藉JessTab讓ontology和Jess溝通，使用者可從規則庫選擇欲檢查的規則，推論可能有誤的相關資料並將結果呈現於使用者介面。以下針對系統各部逐一說明：

（一）Rule Base

知識工程師與領域專家進行討論，針對網站內容語意上可能有誤的部分，設訂定條件或限制範圍後分工，知識工程師進行撰寫規則、與領域專家進行第二次討論，將規則分門別類存放、建置規則庫。日後如欲新增、修改或刪除規則，可直接到規則庫進行增刪動作，並可變更不適用之規則。

（二）Inference Engine

透過Patter Matcher進行比對規則和事實，找出能被啟動與可執行規則，藉由Agenda進行排序規則，Execution Engine會依Agenda內的規則依序執行，執行的動作則稱為「啟動」（Fire）。推論引擎的運作過程如圖6所示。

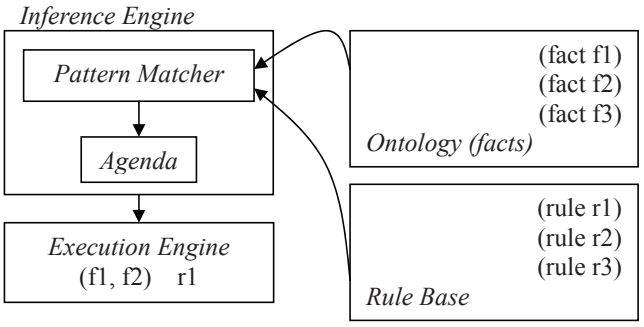

圖6：推論引擎運作過程

（資料來源：Friedman-Hill, E., 2003）

（三）Interface

利用Java程式語言建置使用者介面及編輯規則介面。使用者無需了解系統內部推論過程與運作方式，僅需選擇規則以得到推論結果。編輯規則介面讓知識工程師在編輯rule和function時容易上手，不需透過Protégé內plug-in--JessTab進行編輯撰寫，可將規則儲存成檔案以便日後修改。本研究中的Semantic Reasoning System介面主要有：(1)本體論instance轉換介面：透過此介面可將網站資料進行轉換成具階層關係的本體論。(2)編輯規則介面：知識工程師在此可編輯所需的規則、函式並儲存成檔案，以便規則庫的管理。(3)使用者介面：使用者選擇欲檢查的規則及函式後可進行推論，推論結果呈現於介面上。

二、建置本體論

本研究使用之本體論編輯器為Protégé 3.1.1[Noy et al. 2001]，以W3C組織所制定的RDF和OWL定義某3C家電賣場網站內容之本體論架構，描述其類別、屬性及限制，主要是將網站上各類商品、商品之間的關聯定義清楚，透過本體論了解商品部門和子部門間的關聯，不同商品的屬性和限制條件，這些都是檢查網站內容之基礎。

（一）定義ontology領域範圍

1. 界定本體論的領域與範圍

本研究整理某家電賣場網站內領域的相關名詞，並將網站各類商品、商品之間的關聯定義清楚，建立ontology了解商品部門和子部門間的關聯。不同商品的屬性和限制條件均為日後檢查網站內容的基本要件，以作為日後檢查的基礎。鑑於以往並未有學者從事類似研究，故無現成的本體論可資沿用、修改，本研究所訂之類別，均以研究對象之家電賣場網站內商品分類為基礎，依商品功能屬性分為七大類：電視、重電、影音、小家電、資訊商品、通訊/OA及非本業商品；各類商品都各具一些共同和非共同的屬性。

類別建立完成後、描述每一類別的特徵屬性（亦成為限制條件）；最後透過商品資訊轉換介面建置本體論，此本體論可供各電子商務網站公司使用，作為維護網站內容的

依據。根據上述目的須討論界定：

（1）Ontology涵括的範圍為何？

電子商務網站的商品資訊及不同商品部門間的關係。ontology是維護網站內容的基礎，可用來檢查網站內容錯誤或遺漏的資訊。

（2）有哪些條件是ontology需要具備的？

電子商務網站的商品部門有哪些？合理的商品價格應介於哪個範圍？某商品部門有哪些子部門商品？商品部門間的關係為何？

（3）賦予使用ontology之權限並執行維護及檢查？

網站內容與否正確完整？維護人員使用ontology進行檢查與維護。

2. 是否有已現成的ontology可供使用

目前並無合適的電子商務網站Ontology可供使用，故本研究參酌研究對象之網站的內容自行制訂定ontology，日後如有其他公司欲參酌本網站，僅須一產品種類與屬性適度修改即可。

3.定義電子商務網站詞彙

本研究以某電子商務網站的內容為基礎，分類整理產品名詞，將商品分為七大類，依序為：100代表電視（TV）、 200代表重電（Heavy_Elect）、300為影音（Vedio_Sound）、400為小家電（Home_Elect）、500為資訊商品（Information_Merchandise）、600為通訊/OA（Communication_OA）及700為非本業商品（Other_Product）；各類商品各具一些共同和相異的屬性，如：商品名稱（has_idesc）、商品價格（has_iprice）、商品型號（has_imodel）等。

（二）建立階層、屬性與實例

1. 定義ontology之類別階層關係與繼承關係

依據本研究對象某電子商務網站所提供之商品部門、商品子部門分類，再對各個商品部門子類別的分類，判斷類別間的關係。本研究中的電子商務網站內，所有的商品共分成七大部門，此七大商品部門下各自細分子部門、商品分類與商品子分類，這些皆根據商品的共通性來制定其父類別。如：Information_Merchandise（資訊商品）包含：CD_Rom（光碟機）、Floppy（軟碟機）和Hard_Disk（硬碟機），這三項商品共通性為儲存裝置資訊商品，故這三項商品的superclass即為『Storage_Device』（儲存裝置），而Storage_Device（儲存裝置）和Interface_Card（介面卡）又可進一步歸類至『Computer_Component』（電腦組件），但Computer_Component（電腦組件）和Computer_Peripheral（電腦週邊）都屬於『Information_Merchandise』（訊商品）。依據上述分類方式將商品進行分門別類。圖7為部分資訊商品類別圖。

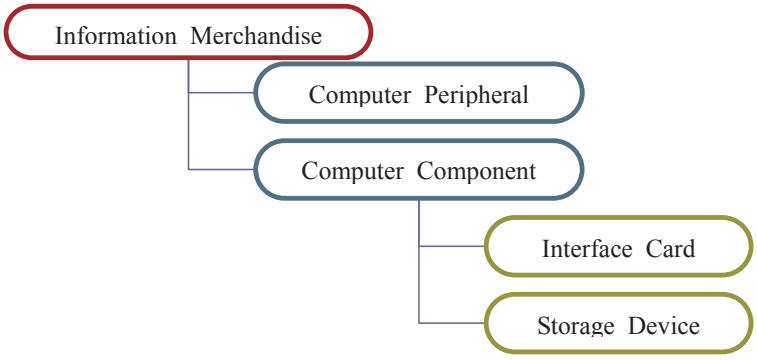

圖7：資訊商品部分類別圖

圖8為Protégé實作本研究的ontology架構圖，每一個節點皆為一個class，呈現各項商品部門、商品子部門、商品類別與商品子類別的階層關係，整個ontology的根節點是以owl:Thing為root，將ontology分成TV（影視）、Heavy_Elect（重電）、Vedio_Sound（影音）、Home_Elect（家電）、Information_Merchandise（資訊商品）、Communication_OA（通訊商品）和Other_Product（非本業商品）七大類，再分別建立其子類別。

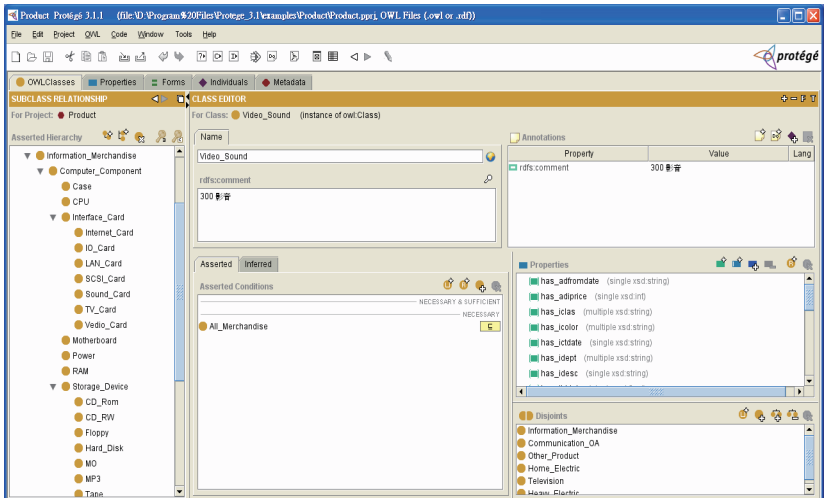

圖8：Protégé實作類別部分架構圖

2. 建立屬性

完成ontology的類別後，需描述各類別的特徵屬性，這些屬性兼具為限制條件。本研究中商品屬性依某電子商務網站商品資料建立，針對網站偵錯部分另增若干個屬性，俾使偵錯網站時增加其可靠度。

3.實例測試

建立完成類別後，將實際的商品資訊填入ontology，將本研究中所採用的電子商務

網站之商品資料庫內容，直接存進ontology相對應的屬性即可。圖9為類別Computer_Conponent（電腦組件）之子類別Storage_Device（儲存裝置）下的子類別CD_ROM（光碟機）建立實例。

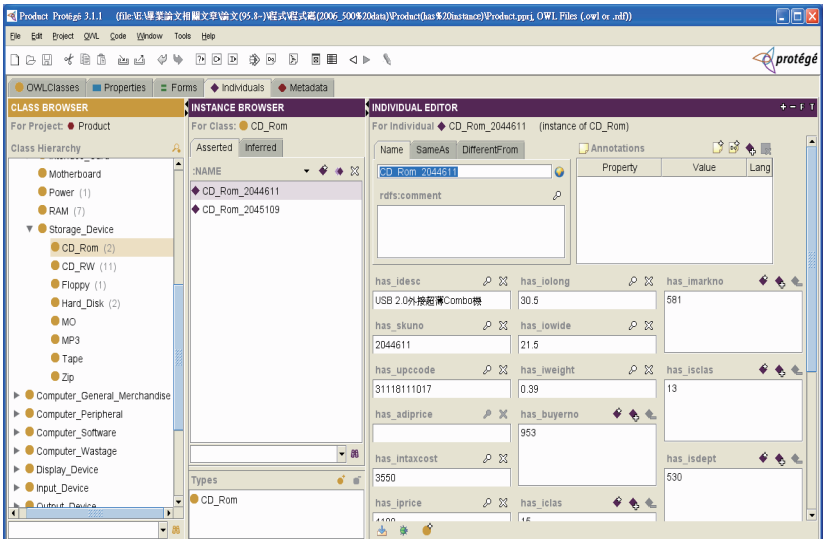

圖9：CD_ROM（光碟機）之實例

透過商品資訊轉換介面，將實際的商品資訊建置到本體論。圖10為本研究建置的部分商品本體論。

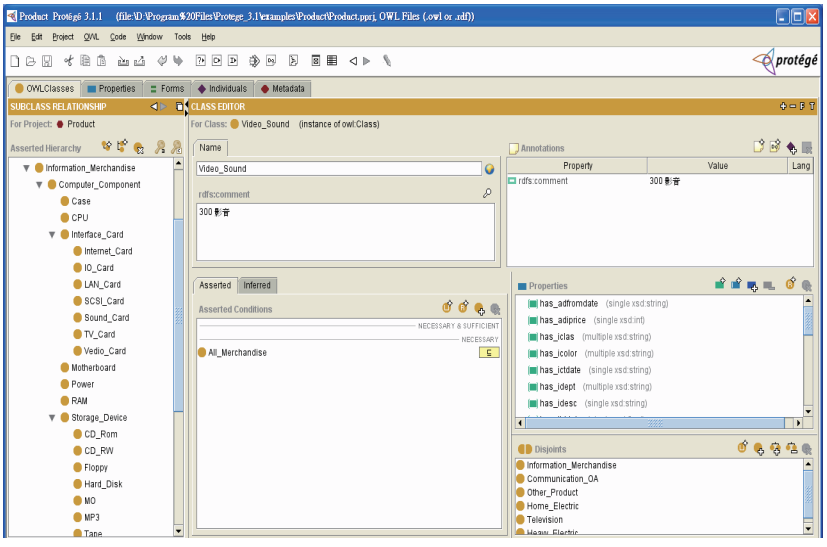

圖10：部分商品本體論

三、Jess語言

本研究使用Sandia National Laboratories以Java所開發的專家系統發展程式Jess撰寫規則。根據某大型3C家電賣場的領域專家，彙整以往偵錯時網站資料可能發生語意錯誤的原因，撰寫規則及規則所需的函式[Friedman-Hill 2003]：

（一）Jess Function：

為輔助規則執行所建立。依據規則內容，定義Jess Function的語法如下：

```
(deffunction <name>
[<comment>]
(<regular-parameter>    * [<wildcard-parameter>])
<action>*)
```

deffunction表示定義一個function，name為function的名稱，comment描述function待執行事項，regular-parameter和wildcard-parameter皆為function內會使用的參數，各為單一欄位與多欄位值的變數，action為function執行的內容，當function被呼叫時，action的內容會依序被執行。

圖11為一簡單的Jess Function範例，此範例為計算兩個日期間的相差天數，使用or、not布林運算元判斷條件為true或false，integerp指令判斷變數為整數值與否，printout t指令為列印。

```
(deffunction CompareDate (?t1 ?t2)
  (bind ?t1 (new StringBuffer))
  (bind ?t2 (new StringBuffer))
  (try
(bind ?da1 (call ?str parse (call ?t1 toString)))
(bind ?da2 (call ?str parse (call ?t2 toString)))
(bind ?result (call Math abs (-(call ?da1 getTime)(call ?da2 getTime))))
(if (> ?result 604800000) then
  (printout t "Factorial Error!" crlf))
catch))
```

圖11：Jess Function範例

（二）Jess Rule：

由一到多個conditional-element和action組合而成，當conditional-element皆成立時才執行action。用「=>」符號分為LHS和RHS兩個部分，當LHS成立時，就執行RHS的動作。定義Jess Rule的語法如下：

```
(defrule <rule-name>
[<comment>]
[<declaration>]                          ; Rule Properties
<conditional-element>*   ; Left-Hand Side (LHS)
=>
<action>*)                       ; Right-Hand Side (RHS)
```

defrule表示開始定義rule，rule-name為rule的名稱，comment描述rule要執行的內容，conditional-element表示一連串的條件（即LHS），action為該rule要執行的內容（即RHS），當rule的條件都成立時，便執行action。

圖12為一簡易之Jess Rule範例，此範例描述“商品價格小於含稅成本”狀況成立時，即列印所有價格小於含稅成本的商品資訊及警告訊息，以達預警目的。

```
(defrule Check_Price_Intaxcost
"Check price is cheaper than intaxcost "
(object (is-a ?x) (has_skuno ?name)
(has_iprice ?ip) (has_intaxcost ?itc))
(test (< ?ip (* ?itc 1.05) ) )
=>
(printout t "Warning: price may have problems !! crlf))
```

圖12：Jess Rule範例

二、整合本體論和Jess

建構本體論和Jess Rule、Jess Function後將二者整合，建立完整的系統。本研究擬將Jess嵌入Java應用程式中，使用JessTab 讓Jess與Protégé溝通，以建置語意偵錯系統。

（一）將Jess嵌入Java應用程式中

Jess的推論引擎是以Java撰寫的jess.Rete類別為基礎，在Java應用程式中欲建立jess.Rete類別的物件時，其語法如下：

```
import jess.*;
……
        Rete engine = new Rete();
```

上述指令宣告了JESS的推論引擎在Java應用程式中，可直接撰寫JESS推論規則與推論功能。在Java程式中執行JESS語法時，常用executeCommand。executeCommand接受的參數為String回傳一個jess.Value，假設吾人新增一個「事實」（fact），語法可撰寫為：

```
import jess.*;
……
Rete engine = new Rete();
Value v = engine.executeCommand("(assert (color red))");
```

上述指令表示JESS的fact皆是以jess.Value儲存，jess.Value可將JESS所有的格式內容呈現出來，如：String、Integer等，故Java應用程式中的JESS推論引擎和其他外部資料有輸入/輸出的動作時，須將Java格式轉換成JESS格式，方可正確地運作。

（二）使用JessTab整合Jess與Protégé

Protégé具有可透過各種Plug-in以擴充功能的優點，而JessTab可有效整合Jess與Protégé，藉由撰寫Jess 程式以存取Protégé建置的本體論。JessTab提供Java函式和指令以執行對本體論的新增、修改、刪除等功能，透過函式及Jess Rule完成規則判斷，並以推論結果對本體論進行變更，成功整合本體論和Jess。

Java應用程式應用JessTab，以整合函式和Jess推論引擎，達到存取本體論、透過撰寫規則進行網站資料偵錯的目的，並將推論結果顯示於介面上。其步驟為：導入本體論、撰寫推論規則以獲致結果。

1. 導入本體論

Protégé與Jess透過mapping整合，將本體論的instance map（對映）至Jess，成為Jess的fact，故Jess Rule便可存取本體論資料，進行推論[Eriksson 2003；Eriksson 2004]，其語法如下：

(mapclass <class-name> [nonreactive | reactive])

透過mapclass指令，將Protégé內的instance導入Jess。透過mapclass將ontology各個類別下的instance map到JESS內，圖13為Protégé內的部分fact。這些fact是推論的基礎，可撰寫Jess Rule與fact進行Pattern-Mather。

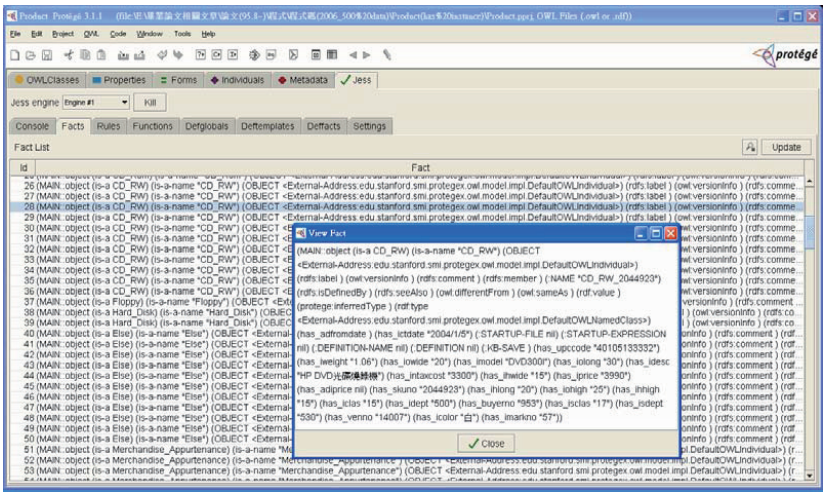

圖13：JESS的fact

執行mapclass後，每個fact會產生一個object，每個object會將其所屬的類別及屬性等資訊紀錄起來，例如：資訊商品有CD_ROM（光碟機）這個instance，那內容可能就包含（is-a CD_RW）（has_iprice "3990"）（has_skuno "2044923"）等等，將來這些fact都將是推論的基礎，接下來我們就可撰寫JESS rule與這些fact進行比對、檢查。因此，在Java程式中就可宣告jess.Rete建立JESS推論引擎，接著將JessTab引入使用，並透過mapclass的指令導入ontology之事實，片段程式碼如下：

```
import jess.*;
import JessTab.*;
……
Rete engine = new Rete(null);
                                engine.addUserpackage(new JessTabFunctions());
                                engine.executeCommand("(load-project Product.pprj)");
                                engine.executeCommand("(mapclass Television)");
……
```

2. 撰寫推論規則以獲致結果

當本體論的instance導入至Jess後，知識工程師便可透過規則編輯介面編輯Jess Rule與Jess Function，編輯方式同第三節Jess語言所述，建立的規則具有以下兩個特點：

（1）一般性：所有規則皆可適用於案例的全部事實，僅需撰寫一項規則即可同時檢查是否有語意錯誤，不必為不同或特殊性的商品個別撰寫規則。

（2）可塑性：規則可隨檢查的範圍或內容進行新增、修改或刪除。檢查過程中如需變更檢查範圍時，可針對特定規則量身修改、新增或刪除規則。

肆、驗證與結果

本研究針對某大型3C家電賣場網站進行實證，將此家電賣場內所有商品資訊建置本體論，根據領域專家的知識和經驗，建立檢查語意錯誤的規則。

一、系統使用說明

本系統所採用的開發語言為Java，資料來源為該3C家電賣場網站之2006年1月到6月份資料，應用Jess與JessTab兩個元件。建構本體論的平台為Protégé，透過JessTab整合Protégé和Jess，以Java語言設計一套應用系統，提供使用者規則編輯介面，方便管理、維護規則庫。本研究的語意偵錯系統介面總共分三個部分：（一）建立本體論instance介面：將家電賣場的商品資訊進行轉換，轉換成具階層架構的本體論之instance。（二）規則編輯介面：知識工程師在此介面編輯Jess Function及Jess Rule，且立即執行，若結果無誤並符合Funciton或Rule要求時，便可存至Rule Base。（三）使用者介面：使用者選擇所需執行的規則，此介面可呈現執行結果。

依據該3C家電賣場之領域專家過去的檢查經驗、本體論類別性質及資料特性歸納後，制訂本研究所使用的規則。從領域專家檢查經驗可得知，網站資料往往因人為因素，輸入過程中容易產生語意錯誤，如某項商品價格遠高於或低於一般市價範圍；另依據本體論類別性質可推估商品未依尺寸進行分類，以致發生分類錯誤，產生尺寸和商品名稱明顯不符，此亦為語意的另一錯誤；再者歸納資料特性後可知：某些商品其價格應與尺寸大小成相關。但若尺寸越大之商品，反之價格卻便宜，如冰箱、液晶電視等商品，依常理判斷：同一品牌，尺寸越大價格應較昂，否則就是語意產生錯誤。

依上述思考模式結合知識工程師與領域專家進行討論，針對網站內容資料可能的語意錯誤配合行銷策略，如：七天內不可更動售價等，由知識工程師逐一撰寫語意檢查規則，與領域專家進行二度討論、修改，完成規則並進行推論。

以下是本研究所建立之規則案例，規則內容以Psudocode呈現：

（一）規則一（如圖14）：Check_Price_Vary，檢查商品原始與變動後的價格，變動是否過大，若變動比率大於等於10%，則商品變動比率過大，應執行警示。

For every instances
if product_vary_price ≠ nil
*and if ((product_vary_ price-product_ price) / product price * 100) > or == ten*
print the instance's product_number & message

圖14：規則一：Check_Price_Vary

（二）規則二（如圖15）：Check_Vary_Day，檢查商品其定價日期和變價日期之相差天數是否少於七天，若少於七天，可能導致退差價情況。

For every instances
if product_set_price_date and product_vary_price _date ≠ nil
and call function CompareDate to compute the interval between the two dates
if interval < seven days
print the instance's product_number & message

圖15：規則二：Check_Vary_Day

（三）規則三（如圖16）：Check_Price_Intaxcost，檢查商品其原價是否小於成本乘上1.05 (即含稅成本，以國內加值型營業稅5%為計算基準)，若小於含稅成本，則可能發生賠售情形。

For every instances
*if product_price < (product_Tax_excluded_price * 1.05)*
print the instance's product_number & message

圖16：規則三：Check_Price_Intaxcost

（四）規則四（如圖17）：Check_Brand_Size，檢查同類商品，相同品牌其尺寸越大、價格應正比成長，若尺寸越大、價格反而便宜，則可能發生價格標示錯誤的情況。

For every instances
if any two instance's product_supplier_number, product_department, product_sub-department, product_class == each other
and if one product_size > the other
and if one product_price < the other
print the two instance's product_number & message

圖17：規則四：Check_Brand_Size

（五）規則五（如圖18）：Check_Price_Range，檢查商品價格是否超出其最大或最小價格範圍，避免商品價格發生異常。

For every instances
*if (product_Tax_excluded_price * 1.05) isn't between this product_class_price_range*
print the instance's product_number & message

圖18：規則五：Check_Price_Range

（六）規則六（如圖19）：Check_Weight_Range，如儲存裝置的光碟片，檢查商品重量是否超出其最大最小重量合理範圍。

For every instances
if product_weight isn't between this product_class_ weight_range
print the instance's product_number & message

圖19：規則六：Check_Weight_Range

（七）規則七（如圖20）：Check_Size_Range，檢查商品長寬高是否超出其最大最小長寬高範圍，避免商品尺寸標示異常。

For every instances
if product_long, wide, high aren't between this product class_long, wide, high range
print the instance's product_number & message

圖20：規則七：Check_Size_Range

（八）規則八(如圖21)：Check_Category，檢查商品是否依據其尺寸大小分類無誤，如果某商品尺寸為20吋，卻分類在21吋的類別，則明顯有誤。

For every instances
if product_size ≠ the product_class's_ size or isn't between product class's size range
print the instance's product_number & message

圖21：規則八：Check_Category

二、使用本系統實地驗證、偵錯

本研究對象係以某「大型3C家電賣場」之網站2006年1月到6月資料進行驗證，評估是否能使網站管理人員使用語意推論系統實際偵錯，上述規則應用於各推論結果如下：

（一）圖22顯示：本研究應用規則一找出價格變動比率大於等於10%的商品資訊，共計12項商品。依據家電賣場制定售價的原則，若商品價格變動超過10%時，表示商品標價異常。故一旦價格變動過劇，隨即啟動執行規則一列示異常商品資訊，避免因價格變動過劇導致虧損，或因價格昂貴使消費者裹足發生滯銷。

（二）圖23顯示：本研究應用規則二找出價格變動日期相差日數少於七日之商品，共計1項。依據家電賣場規定：若價格變動日期相差日數少於七日之商品，賣場須承擔”退還差價”風險。因此商品變價後，執行規則二可列示變價天數少於七日之商品，可有效防杜退還差價之困擾。

（三）圖24顯示：本研究應用規則三找出售價低於含稅成本商品，共計達134項。商品成本可能因各店成本不同而有差異；在規則三列示商品售價小於含稅成本的全部商品資訊，提供網站管理人員再確認，以避免“consumer hackers”大筆下單訂購。

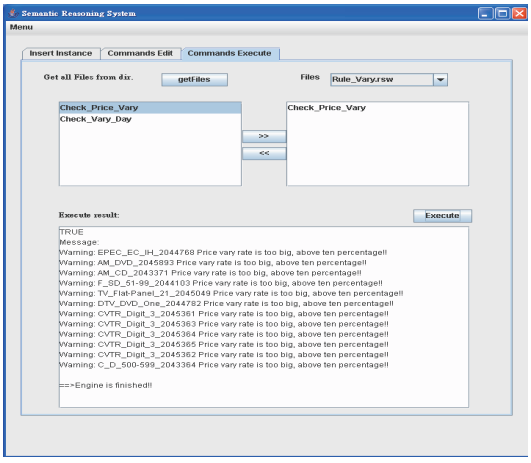

圖22：規則一推論結果

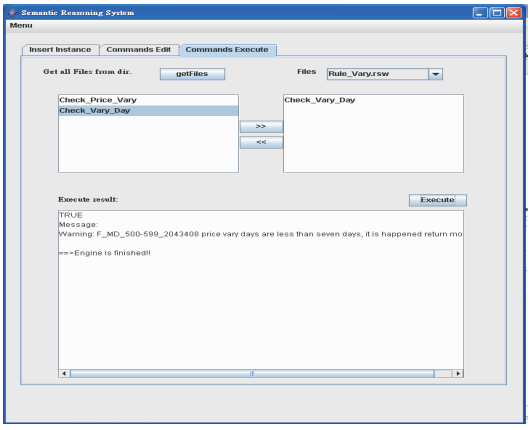

圖23：規則二推論結果

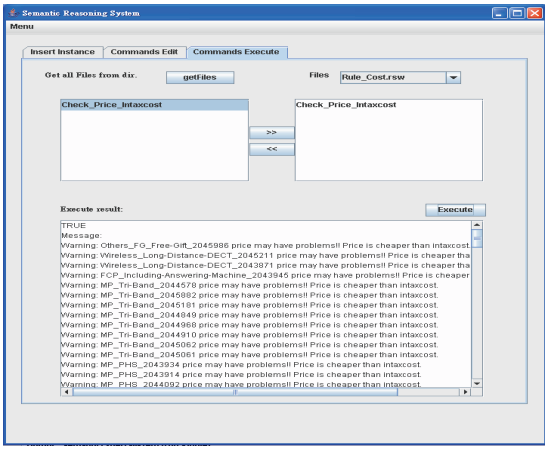

圖24：規則三推論結果

（四）圖25顯示：本研究應用規則四找出尺寸越大、價格反而便宜的商品，共計達172項。商品本體論中，如：相同品牌之電視螢幕，依“尺寸”屬性而與價格間有依存關係。規則四可檢查商品尺寸和價格是否成正相關，以避免網站商品資訊顯示錯誤。

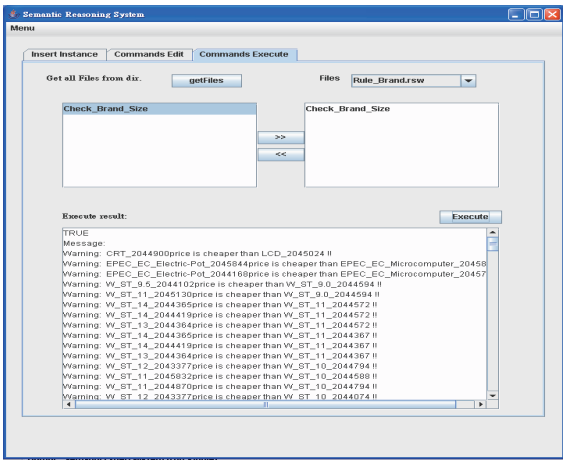

圖25：規則四推論結果

（五）圖26顯示：本研究同時應用規則五、六、七找出異常的商品資訊，共計1項。規則五為檢查各類商品價格是否標示異常、規則六則檢查商品重量是否超出最大或低於最小重量。如PDA、手機等產品，除功能外一般強調輕薄短小的外觀，其重量約僅數百公克。若某PDA商品重量之網站資訊顯示為數公斤，則該商品的重量語意顯有錯誤。規則七檢查商品外觀，是否大於或小於某尺寸。某些小筆電商品的厚度可能僅5公分以內，但若某筆電商品高度網站資訊顯示為5公分以上，則該商品的高度語意顯有錯誤。綜合應用規則五、六及七，可同時將商品價格、重量與外觀尺寸的語意錯誤檢查出來，使合理的呈現商品資訊。

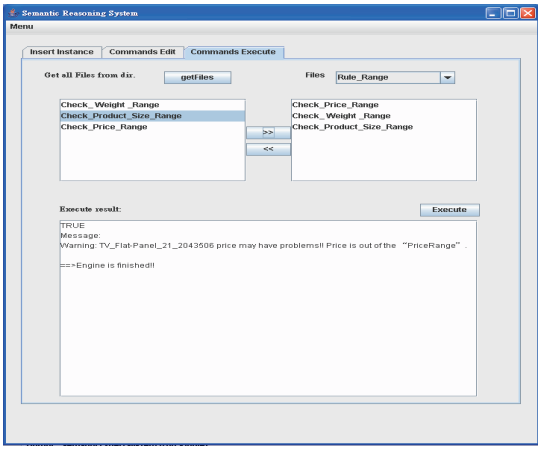

圖26：規則五推論結果

（六）圖27顯示：應用規則八找出尺寸分類錯誤的商品，共計51項。本研究案例的五類商品(電視機、冰箱、洗衣機、照相機和防潮箱)，依據尺寸大小進行商品子分類，雖商品尺寸值的語法無誤，但若商品未正確分類，亦可顯示語意錯誤的資訊。故應用規則八可列示分類錯誤的商品。

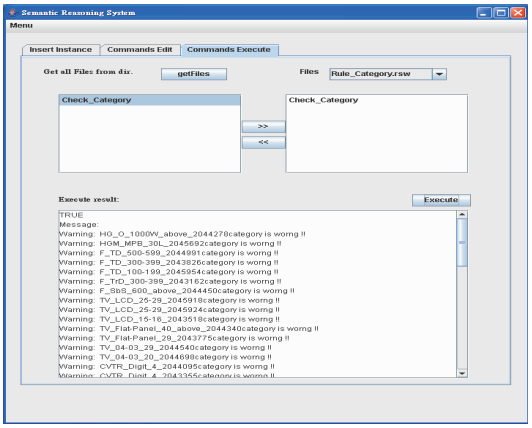

圖27：規則八推論結果

本研究利用語意偵錯系統執行網站資訊檢查，檢查結果深獲家電賣場網站管理人員認同，經確認、比對檢查語意可能有誤的商品資訊，對維護網站資訊的正確、完整與合理助益頗大。不僅大幅減少網站檢查時所耗費之人力及時間成本，並將網站內容可能因語意錯誤而導致的損失減低至零。故本研究提出之架構與偵錯機制，對網站資料語意偵錯確有實際之效益，可提供網站管理者成為網站偵錯的有效輔助工具。

伍、結論

一、結論

本研究應用本體論的概念建立商品本體論、依據領域專家的知識與經驗，撰寫網站語意檢查的規則並建置規則庫，經實地驗證此語意偵錯系統，確能有效執行規則檢查使網站資訊合理化、企業損失極小化。本研究之具體貢獻如下：

（一）首度嘗試應用本體論概念建立商品本體論

本研究依據國內某3C家電賣場資訊，建置商品本體論，建立銷售商品的類別、屬性與類別間關係，但須對商品概念做詳盡之描述。未來將可藉本體論共享之特性，提供其他3C家電賣場定義其商品本體論時，建構可共用或擴充之版本，尚可依據不同之產業別提供商品資訊變更其本體論，以達知識共享。

（二）自行建立網站語意偵錯規則

本研究建立之語意偵錯規則，結合該3C賣場之領域專家依過去的經驗及知識，配

合訪談、討論及並參酌相關網站資訊自行建立規則庫，推論網站可能有語意誤的商品資訊，共同對網站資料執行偵錯，經驗證確能發揮預期功能。

（三）自行建置語意推論架構

本研究提出網站商品資訊之語意偵錯架構，從建立商品本體論、撰寫檢查網站資料語意之Jess規則，藉JessTab整合本體論及Jess ，經由推論引擎獲得推論結果。本架構可應用於其他領域，依領域屬性建立該領域之本體論和規則。

（四）語意偵錯系統有效減少人力消耗

目前網站資訊是否正確性、合理，須藉人工方式一一輸入及檢查，受時間、體力或外在環境因素所限，無法察覺可能的輸入錯誤或資訊闕漏狀況。本研究建置之語意偵錯系統，藉由撰寫規則結合本體論管理、維護網站資料，可隨時保持網站內容的正確性及合理性。

二、未來研究方向

本研究雖自行建立商品本體論與規則庫的架構，但仍有可改善的空間。未來尚可朝下列方向研究。

（一）建構不同行業本體論：

本研究所建置的本體論是依某大型3C家電賣場網站內容所設計，其涵蓋之內容範圍僅侷限於該家電賣場所制定之類別，未來不同的家電賣場可依此本體論進行修改及共享，以建構完整之商品本體論。另目前「語意」的概念仍嫌不足，若日後能加入更多語意規則或概念到本體論，則未來網站偵錯功能將更加完整。

（二）規劃發展新事實的規則：

目前規則設計部份仍侷限可能有問題的商品資訊，尚未規劃推出新事實的多個規則，再將事實推回到本體論instance。未來研究可朝建置規則的方向，藉規則推出新事實，再將事實推回本體論，從而衍生更多的規則進行推論，如此便可建置完整的語意偵錯規則庫。

（三）擴充系統的介面：

系統目前僅提供Jess Rule、Jess Function的編輯介面、使用者介面及簡易的規則管理，未來可朝所規劃系統介面能符合使用者需求，如排程功能可依檢查內容與時間點進行規則排程，俾具隨時檢查的功能。

參考文獻

1. Bollegala D., Honma, T., Matsuo, Y.,and Ishizuka, M.,“Mining for personal name aliases

on the web aliases,” *Proceeding of the 17th International Conference on World Wide Web 2008, WWW'08 2008*, Pages 1107-1108

2. Brüggemann, S. “Rule mining for automatic ontology based data cleaning,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) Volume 4976 LNCS*, 2008, Pages 522-527.
3. Chandrasekaran, B., Josephson, J. R., and Benjamins, V. R., “What Are Ontologies, and Why Do We Need Them,” *IEEE Intelligent Systems*, Vol. 14, Iss. 1, 1999, pp. 20-26.
4. Cruz, C., and Nicolle, C. “XML-IS: Ontology-based integration architecture,” *WEBIST 2008-4th International Conference on Web Information Systems and Technologies, Proceedings Volume 1*, 2008, Pages 273-277?
5. Daconta, M. C., Obrst, L. J., and Smith, K. T., *The Semantic Web: A Guide to the Future of XML, Web Services and Knowledge Management*, Wiley Publish, 2003.
6. Davis, R., Buchanan, B. G., and Shortliffe, E. H., “Production Rules as a Representation for a Knowledge-Based Consultation Program,” *Artificial Intelligence*, No. 8, 1977, pp.15-45.
7. Ding, Y., “Ontology: The enabler for the Semantic Web,” *Journal of Information Science*,” Vol. 27, No. 6, 2001, pp. 377-384.
8. Eriksson, H., “JessTab Manual--Integration of Protégé and Jess,” *http://www.ida.liu.se/~her/JessTab*, Linköping University, 2004.
9. Eriksson, H., “Using JessTab to Integrate Protégé and Jess,” *IEEE Intelligent Systems*, Vol. 18, Iss. 2, 2003, pp. 43-50.
10. Extensible Markup Language(XML), W3C Consortium, (*http://www.w3.org/*)
11. Friedman-Hill, E., “Jess in Action: Rule-Based Systems in Java,” Manning Publications , 2003.
12. Giarratano, J. C., and Riley, G. D., “Expert Systems: Principles and Programming,” *PWS Publishing*, 1998.
13. Gruber, T. R., “A Translation Approach to Portable Ontology Specifications,” *Knowledge to Acquisition,* Vol. 5, Iss. 2, 1993, pp.199-220.
14. Gruber, T. R., “Towards Principles for the Design of Ontologies Used for Knowledge Sharing,” *International Journal of Human-Computer Studies*, Vol. 43, Iss. 5-6, 1995, pp.907-928.
15. Horrocks, I., Patel-Schneider, P. F., and Harmelen, F. V., “From SHIQ and RDF to OWL: The Making of a Web Ontology Language,” *Journal of Web Semantics*, Vol. 1, No. 1, 2003, pp. 7-26.
16. Jovanovic, J., Gasevic, D. and Devedzic, V. “A GUI for Jess,” *Expert Systems with Applications*, Vol. 26, No. 4, 2003, pp.625-637.
17. McGuinness, D.L., Fikes, R., Rice, J. and Wilder, S., “An Environment for Merging and Testing Large Ontologies,” *Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning*, 2000.

18. Musen, M. A., "Dimensions of Knowledge Sharing and Reuse," *Computers and Biomedical Research*, Vol. 25, Iss. 5, 1992, pp. 435-467.
19. Nault, B. R. and Storey, V. C., "Using Object Concepts to Match Artificial Intelligence Techniques to Problem Types," *Information and Management*, Vol. 34, Iss. 1, 1998, pp.19-31.
20. Navigli, R., Velardi, P., and Gangemi, A., "Ontology Learning and Its Application to Automated Terminology Translation," *IEEE Intelligent Systems*, Vol. 18, Iss. 1, 2003, pp.22-31.
21. Negnevitsky, M., "Artificial Intelligence-A Guide to Intelligent Systems," *Addison Wesley*, 2002.
22. Noy, N. F., and McGuinness, D. L., "Ontology Development 101: A Guide to Creating Your First Ontology," *Stanford Knowledge Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880*, 2001.
23. Noy, N. F., Sintek, M., Decker, S., Crubézy, M., Fergerson, R. W., and Musen, M. A., "Creating Semantic Web Contents with Protégé-2000," *IEEE Intelligent Systems*, Vol. 16, Iss. 2, 2001, pp. 60-71.
24. RDF Primer, W3C Recommendation, (*http://www.w3.org/TR/rdf-primer/*)
25. Ruckhaus, E. and Vidal, M. E., "XWebSOGO: An Ontology Language to Describe and Query Web Sources," *Workshop On Web Information And Data Management*, 2003, pp.62-65.
26. Shortliffe, E. H., "Computer-Based Medical Consultations : MYCIN," *Elsevier*, 1976.
27. Speel, P-H., Schreiber, A. Th., van Joolingen, W., van Heijst, G., and Beijer, G. J. "Conceptual Modelling for Knowledge-Based Systems," *Encyclopedia of Computer Science and Technology*, 2001.
28. Vaculín, R., and Sycara, K. "Specifying and monitoring composite events for semantic web services," *Proceedings of the 5th IEEE European Conference on Web Services, ECOWS 07*, 2007, Article number 4399738, Pages 87-96