Intelligent Information Management, 2009, 25-31
doi:10.4236/iim.2009.11005 Published Online July 2009 (http://www.scirp.org/journal/iim)

Reasoning with Ontology Model Based on Jena

Jing WU[1], Yongxin HUANG[2]
[1]*National Library of China, Beijing* 100081, *China*
[2]*Information Technology Institute, Beijing Union University, Beijing* 100101, *China*

Abstract: As the core of knowledge representation, reasoning plays an important role in the Semantic Web. In this paper, introduce Description Logic as the theoretical foundation of reasoning in the Semantic Web and OWL which needs to be expressive enough for representing knowledge and computationally tractable for inference of large-scale applications. Then construct an ontology model depicted with OWL by protégé and Jena API , and also prove the validity of reasoning with the model.

Keywords: OWL, description logic, jena, protégé

基於 Jena 的本體模型推理

吳　靜[1]，黃永欣[2]

[1]中國國家圖書館，北京 100081；[2]北京聯合大學，信息技術研究所，北京 100101

摘　要：推理作為知識表示的核心在語義 Web 中扮演著重要的角色。本文先介紹了語義 Web 中推理的理論基礎描述邏輯，以及提供較強知識表達能力並適合大規模應用推理的本體語言 OWL，然後結合建模工具 protégé 和語義網系統開發工具 Jena API 構建了一個用 OWL 語言描述的本體模型，並實現了對此模型的有效推理。

關鍵詞：本體網絡語言，描述邏輯，Jena，protégé

1. 背景和介紹

本體被定義成“概念模型的顯示表示”[1]，目前本體的應用熱點集中在語義 Web (Semantic Web) 中，本體實現了對 Web 內容的形式化與結構化描述，使得電腦也能夠在“理解”的前提下更好地處理、利用 Web 上的資訊和知識。因此,本體是語義 Web 的基石[2]。

本體語言如 RDF、DAML+OIL、OWL[3][4]是基於 XML 標準的，XML 是元資料(描述資料的資料)實現的技術手段,從資料與文檔的底層實現格式化。RDF 資源描述框架(Resource Description Framework)是建立在 XML 標準之上來描述元資料以及元資料與元資料之間的關係，其定義了一個簡單的資料模型，通過性質(property)和值(value)來描述資源以及資源與資源之間關係，因此 RDF 模型可以看成一個實體關係圖。RDF 是一種相對原始的語言，為了更詳細的描述資源，需要更強的表達能力，只有擁有自動推理功能這些描述機制才能在自動化處理中得到應用。基於以上考慮推動了 DAML+OIL 和 OWL 等本體語言的發展。這些本體語言是建立在描述邏輯研究的基礎上引入了 Web 的特點而建立起來的。其中 OWL 是 W3C 工作組在 2001 年創建的 Web 本體語言，2004 年 2 月 10 日起已正式成為 W3C 推薦標準[5]。

各種各樣的本體編輯工具也推動了本體技術進一步的發展,目前主要有 1)由美國 Manchester 大學開發的 OILEd[6]，它使用 OIL(DAML 十 OIL)本體語言來構建本體，較好地結合了框架表示(Frame

Representation) 和描述邏輯(Description Logical)表示兩者的長處。OILEd 編輯後的本體以 DAML 的檔形式存儲,載入了 FaCT(Fast Classification of Terminologies)推理機,可實現類的分層推理功能。2) 由美國 Stanford 大學開發的 Protege2000[7],它是用 Java 開發的一個開源專案，它採用 Plugin 機制使結構容易擴展，Protege 2000 beta 版開始以 OWL 插件的形式支持 OWL 格式的本體，此 OWL 插件以 Java 開發，用 Jena[8]包(HP 實驗室開發的 Java API)支援 OWL 文檔的讀取。Protege2000 工具本身沒有嵌入推理工具，不能實現推理，但該工具很容易嵌入其他的系統或與其他系統聯繫使用。本文使用外掛 Racer[9]推理機進行推理。3) 由英國 Knowledge Media Institute of the Open University 開發的 WebOnto[10]，它支持本體的協作創建和編輯，目的就是容易使用，也更容易地擴展形成大的本體，WebOnto 中本體的建模語言是 OCML。

根據實際需要，本文採用 Protégé 本體建模工具創建和編輯本體，利用 Jena 開發工具包對基於 OWL 語言描述的本體模型在應用程式中進行推理。

2. 描述邏輯

描述邏輯（Description Logic, DL）又稱為術語邏輯（Terminological Logic）或類 KL-ONE 系統，是在命題邏輯(Propositional Logical)和一階謂詞邏輯(First Order Logic, FOL)上發展起來的,是本體語言推理的重要設計基礎。描述邏輯的目的是在表達能力和推理複雜度之間取得平衡，便於提供自動推理服務，尤其是一致性（satisfiability）和歸約(subsumption)關係檢查。

一個標準的描述邏輯系統的體系結構由四個部分組成：(1)表示概念和關係(角色 Role)的構造集；(2)TBox (Terminology Box)一個描述領域結構的公理集,包含概念定義 (如: Wife≡Woman ∩ hasHusband. Man ，即 Wife 定義為“有丈夫的女人”)及公理(如:Woman Man)；(3)ABox (Assertional Box)一個描述關於具體個體事實的公理集，包含概念斷言(如：實例 Woman (marry),marry 是 woman)和關係斷言(如：hasHusband (Marry, Jack) Marry 的丈夫是 Jack)；(4)TBox 和 ABox 上的推理機制。一個基於 DL 的知識庫就是 K=TBox+ABox，簡寫成 KB(T,A)。

定義 1 知識庫解釋：對 KB(T,A)的解釋記為 I，$I=(\Delta^I,\cdot^I)$。其中 Δ^I 為 $\cdot^I$ 的論域，$\cdot^I$ 為映射函數，每個概念映射為 Δ^I 的子集，把關係映射為 $\Delta^I\times\Delta^I$ 一個子集，把實例映射為 Δ^I 的一個元素。一個描述邏輯主要依賴於構造運算元在簡單概念和關係的基礎上構造複雜的概念和關係。構造運算元如表 1 所示：根據構造運算元的不同對 DL 進行分類，最基本的 DL 是 ALC，它只包含合取∩、析取∪、非¬、存在量詞∃、全稱量詞∀這些構造運算元。SHIQ .DL 是在 ALC 的基礎上包含了原始關係/角色傳遞閉包，稱這種類型的 DL 為 S 是因為它和 the proposition (multi) modal logic S4(m)有關係，增加了逆 (inverse Role I)、關係/角色層次(role hierarchies H)、數量約束(Q)這幾種構造運算元,所以稱之為 SHIQ 。OWL DL 就是基於 SHIQ .DL[11]。

綜上所述描述邏輯是一種用來描述概念和概念層次關係的知識表示語言，儘管大多數本體模型採用的一階邏輯具有很強的表達能力，但其推理過程複雜不利於本體模型的檢驗，而描述邏輯可以看成謂詞邏輯的子語言，雖沒一階邏輯的表達能力強，但推理複雜度可知且更適合本體檢驗，而且 DL 的語法容易轉換成 XML/RDF 形式，因此基於 DL 的本體模型更適合 Web 環境下概念建模與知識共用。

表 1. DL 算子的語法與語義
Table 1. Syntax and semantics of DL

構造算子	語法	語義
原子概念	C,D	$C\subseteq\Delta^I$, $D\subseteq\Delta^I$
原子關係	R	$R^I\subseteq\Delta^I\times\Delta^I$
合取	$C\cap D$	$(C\cap D)^I=C^I\cap D^I$
析取	$C\cup D$	$(C\cup D)^I=C^I\cup D^I$
非	$\neg C$	$(\neg C)^I=\Delta^I\setminus C^I$
存在量詞	$\exists R.C$	$(\exists R.C)^I=\{x\mid\exists y.(x,y)\in R^I \text{ and } y\in C^I\}$
全稱量詞	$\forall R.C$	$(\forall R.C)^I=\{x\mid\forall y.(x,y)\in R^I \text{ inplies } y\in C^I\}$
數量約束	$\geq nR.C$	$(\geq nR.C)^I=\{x\mid\#\{y.(x,y)\in R^I \text{ and } y\in C^I\}\geq n\}$
	$\leq nR.C$	$(\leq nR.C)^I=\{x\mid\#\{y.(x,y)\in R^I \text{ and } y\in C^I\}\leq n\}$
逆	R^-	$(R^-)^I=\{(x,y)\mid(y,x)\in R^I\}$
傳遞閉包	R^+	$R^I=(R^I)^+$

3. Web 本體語言 OWL

OWL 是 W3C 推薦的標準本體表示語言，它的最大特點是關聯描述邏輯，這就意味著描述邏輯推理機可以推理 OWL 本體。OWL 是未來的 web 本體語言標準，它是建立在已有的 RDFS 和 DAML+OIL 技術之上的。OWL 通過對概念、概念屬性及其相互間關係的描述，構成概念的複雜關係網絡。

OWL 中的概念由類 Class 來表示，它可以是名字（如 URI）或運算式，而且提供大量的構造運算元來建立運算式，這裏的構造運算元與描述邏輯中的構造運算元相對應，OWL 強大的表達能力正是由它所支持的概念構造運算元、性質構造運算元以及各(Axioms)所決定的。

OWL 是用於在 Web 上描述本體的標準語言，有 3 個表達能力遞增的三個子語言：OWL Lite、OWL DL、OWL Full[12]。

（1）OWL Lite 提供了類層次的能力和簡單的約束能力。支持基數為 0 或 1 的基數約束。語言構造運算元可以分以下 6 個類別：

- RDF 模式特性包括類(Class)、子類（rdfs:subClassOf)、屬性(rdfs:property)、子屬性(rdfs:subPropertyOf)、域(rdfs:domain)、範圍(rdfs:range)和個體(individual)
- 相等/不相等包括類相等(equivalentClass)、屬性相等(equivalentProperty)、個體相同(sameIndividualAs)、不同於(differentFrom)、全不同(allDifferent)、區分成員(distinctMembers)
- 性質特徵包括類屬性(ObjectProperty)、資料類型屬性（Datatype Property）、逆反（inverseOf）、傳遞(transitiveProperty）、對稱（sysmetricProperty）、函數式（FunctionalProperty）、反函數式（InverseFunctionalProperty）
- 性質限制包括限制（Restriction）、限制於屬性（onProperty）、全部取值於(allValuesFrom)和一些取值於(someValuesFrom)
- 基數限制包括最小基數(minCardinality 0 或 1)和 最大基數(maxCardinality 0 或 1)
- 類相交(intersectionOf)

對於版本資訊、頭資訊和評注性質這裏不一一介紹。

（2）OWL DL（其中的 DL 表示描述邏輯）在保持計算完整性（computational completeness 所有的結論可以保證計算出來）和可判定性（decidability 所有的計算在有限時間內結束）的前提下，提供了盡可能大的表達能力，OWL 包含了 OWL 的全部語言構造成分但他們的使用受到一些限制（如一個類可以是許多類的子類，但不能是另一個類的實例）。OWL DL 的主要擴充是增加了布林邏輯運算操作，即邏輯否定和邏輯析取等。描述邏輯是 OWL 的形式化基礎，OWL DL 提供了描述邏輯的推理功能。

（3）OWL Full 包含 OWL 的全部語言構造成分並取消了 OWL DL 的限制，在 OWL Full 中一個類可以看成個體的集合，也可以看成是一個個體。由於 OWL Full 取消了 OWL DL 中的保證可計算的某些限制，在沒有計算保證的語法自由的 RDF 上進行最大程度表達，允許在一個 Ontology 在預定義的（RDF、OWL）辭彙表上增加辭彙，從而任何推理軟體均不能支援 OWL FULL 的所有特點，因此不存在完整的推理演算法支援 OWL Full 的全部特性。

OWL DL 和 OWL Full 中用來擴展 OWL Lite 的構造運算元如下分為 3 類：

- 類公理包括枚舉類型（one of,dataRange）、分離(disjointWith)、類相等（equivalentClass 用於類運算式）、子類(rdfs:subClassOf 用於類運算式)
- 任意基數限制包括最小基數(minCardinality)和最大基數(maxCardinality)
- 類運算式的布林組合包括析取（unionOf）、合取(intersectionOf)和取反（complementOf）
- 填充資訊包括擁有值（hasValue）

4. 開發工具介紹

Protégé[7]是一個斯坦福大學開發的本體編輯器，為開放源碼軟體，具有優秀的設計和眾多的插件，是目前使用最廣泛的本體編輯器之一。它採用圖形化介面，介面上以多個 Tab 分別支持 Classes,Slots,Forms,Instances,Queries 的編輯。它的結構容易擴展，Protege 2000 beta 版開始以 OWL 插件的形式支持 OWL 格式的本體，此 OWL 插件以 Java 開發，用 Jena 包(HP 實驗室開發的 Java API)支援 OWL 文檔的讀取如圖 1 所示。

Protege 的安裝是獨立的。它可支援多種格式的本

體檔的輸入編輯和編輯後的輸出，類(Concept)的編輯是分層的，有具體類和抽象類。Protégé 可以根據使用者的需要進行定制，如可以定制用戶的介面以更好地適應新語言的使用；具有可擴展的結構，可根據需要添加所需要的功能模組。Protégé 2000 工具本身沒有嵌入推理工具，不能實現推理，但該工具有良好的可擴展結構，很容易嵌入其他的系統中或與其他系統聯繫使用，如直接與外部的具有推理功能的語義模組聯繫[14]，本文的例子採用外掛 Racer 推理機。

一般來說，我們在 Protégé 這樣的編輯器裏構建了本體，就會想在應用程式裏使用它，這就需要一些開發介面。用程式操作本體是很必要的，這裏我們使用 Jena API。Jena[8]是惠普實驗室語義網研究項目的開放資源，是一個基於 W3C 的 RDF、DAML/OIL、OWL 進行語義網建模和推理的 Java API 開發包。Jena 框架主要包括如下幾部分：

（1）An RDF API：用於對 RDF 檔和模型進行處理的 RDF 應用程式介面，其主要功能包括創建和讀、寫、操作和檢索 RDF 模型，語義網應用程式可以通過 Jena 提供的 RDF API 在 RDF 檔（例如用本體建模工具 Protégé 建立的或是網路上已經存在的 RDF 檔）和 RDF 模型之間實現轉換，並且用戶可以對 RDF 模型實施簡單而必要的操作和檢索，這是整個應用程式開發的基礎支撐條件。

（2）ARP：Jena 的 RDF/XML 解析器是隨 Jena 發佈內嵌的重要功能，解決了對基於 XML 語法和格式的 RDF 文檔的解析問題，通常在調用 Jena API 的 Read 方法時會調用解析器的功能，為獲取 RDF 文檔中的資訊以及創建 RDF Model 提供支援。

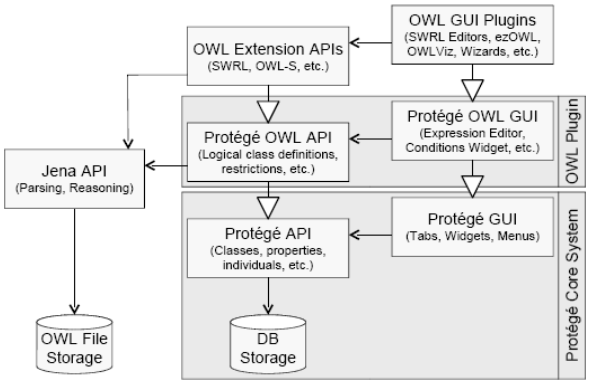

圖 1. Protégé 的 owl plugin
Figure 1. Owl plugin of Protégé

（3）Persistence：持續化功能，除了在記憶體中存儲 RDF、RDFS、OWL 等資料模型，對於資料量較大的本體模型可以通過 Jena 提供的資料庫引擎將其模型存儲到關聯資料庫中，用戶可以用 Jena 的查詢語言 RDQL API 將這些資料從資料庫中檢索出來從而創建 RDF 模型以供進一步使用和處理。目前 Jena 2 資料庫引擎支援 MySQL、Oracle 和 ProtegeSQL 等資料庫，支援 Linux 和 Windows XP 等作業系統。

（4）Reasoning subsystem：推理子系統，Jena 提供基於規則的推理機（如 RDFS Reasoner、OWL Reasoner 等），它包含了一般的推理功能，此外用戶還可以根據需要自定義推理規則，也可以註冊使用第三方推理引擎（例如：本文採用的是 Racer 推理機）。

Jena 是 RDF 和 OWL 的 API，也提供 OWL 和 RDF 的推理功能，Jena 核心只是提供了有限的查詢元語。DL 推理機的實現比較複雜，而 Racer 提供了一個很好的 DL 推理機（描述邏輯推理機），而且還提供標準的 DIG interface 支援。但目前的 DIG interface 只支援標準的 DL 推理服務，尚未支援 OWL 推理。Racer 剛剛開始試驗性的支持推理 OWL 不是一個推理查詢語言最多能用 OWL 描述本體，但不能用 OWL 來詢問本體。這就需要對 OWL 和 DIG 作適當的結合。就推理方面，Jena 自身提供的是基於規則的推理機，是通過規則來實現對 OWL 的推理，不能對所有的推理功能支援，性能上也比不過 Racer，Racer 算是 DL 推理機裏性能最強悍的了，可以支援工業使用，Jena 裏的 Rule Reasoner 性能並不是很理想。Racer 使用的是 DIG 介面，需要通過 HTTP 或是 TCP 訪問，如果知識庫不是很複雜的話，可能會由於通信的時間消耗反而使性能下降。另一方面 Racer 一定程度來說只是單純的推理機，它不對知識庫的存儲做很多考慮，而 Jena 對知識庫的存儲乃至持久化支援的都很完整（可以把知識庫直接存在關聯資料庫中）。新版的 Jena 2.2 已經開始支援集成使用 DIG 介面的 Reasoner 了。Jena 是一定要用的，用不用 Racer 要看知識庫的複雜程度和對推理功能的要求。在推理方面可以通過 DIG 介面連接專門的推理機，由於推理需要 DL 推理機，所以這裏我們採用 Racer，通過 Dig 介面 Racer 直接使用 jena 進行編碼實現推理。

5. 本體推理應用實例

實例使用 Protégé 本體建模工具建立本體模型，該本體描述了六個原始概念（原始類 primitive class）：人（people）、男人（man）、女人（woman）、父親（father）、母親（mother）和父母（parent），一個屬性（關係）：有孩子（hasChild），三個實例：Jim、Marry 和 Terry 和一個定義類（defined class）：誰有孩子（WhoHasChild），用 OWLViz（Protege3.1 版本的一個 plugin）顯示結果本體模型如圖 2 所示：

Man、Woman 和 Parent 均是 People 的子類，Man 和 Woman 兩個類沒有交集是互斥關係（用 Disjoint 來聲明），Parent 至少存在一個 hasChild 關係（最小基數 minCardinality 為 1）說明有孩子的個數大於等於 1 的 People 是 Parent，Father 是 Man 和 Parent 的共同子類，Mother 是 Woman 和 Parent 的共同子類和 Father 沒有交集，這樣就定義了只有既是 Man 同時也是 Parent 的 People 才是 Father，同理既是 Woman 同時也是 Parent 的 People 才是 Mother。聲明三個實例：Terry、Marry 和 Jim，三個實例均為 People 類的實例，同時定義了 Marry 有孩子是 Jim（通過屬性 hasChild 關聯），Terry 是 Parent.最後定義一個 Definded 類 WhoHasChild（用 equivalentClass 來聲明）用於推理，它的充要條件是 WhoHasChild≡hasChild.People≥1。本例主要根據給定的實例描述得出哪些 People 有孩子，也就是推理機 Racer 推理後劃分在 WhoHasChild 類下的實例既是我們想要的結果。圖 3 是在 Protégé 運行 Compute inferred types 命令後得到的推理前與推理後的類結構對比圖。

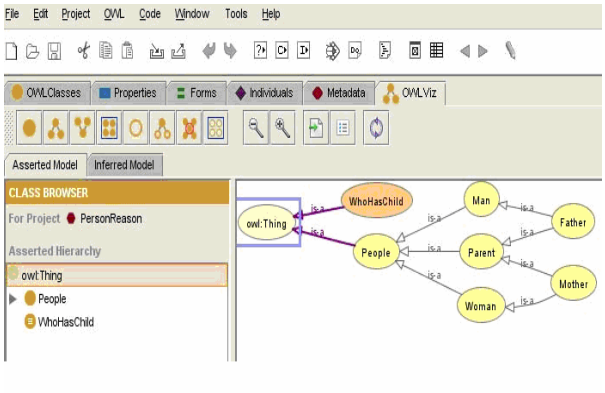

圖 2. 本體模型可視化顯示
Figure 2. Visualization of the ontology model

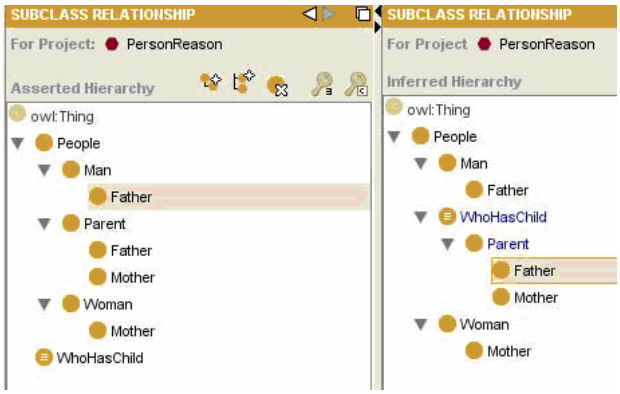

圖 3. protégé 中的斷言類結構和推理類結構對比
Figure 3. Contrast between asserted hierarchy and inferred hierarchy of Protégé

可以看出獲取本體中的知識包含兩個方面：

（1）獲取 TBox 中類的關係和屬性的關係，具體來說就是獲得類的子類集（就是包含推理）、屬性的子屬性集、屬性的特徵、屬性的定義域和值域等等，它更接近於概念的查詢，因此推理可靠性較高，不能出現概念錯誤。本例中對應的是推理後將 Parent 類劃分到 WhoHasChild 類下（因為 Parent 類的屬性 hasChild≥1 符合 WhoHasChild 類的充要條件），說明屬於 Parent 類的概念一定屬於 WhoHasChild 類的概念，同時將 WhoHasChild 類作為 People 的子類。

（2）獲取 ABox 中的資訊，具體來說有獲得一個類的所有實例、實例的屬性、獲得滿足條件的所有實例等等，它更接近於實例的檢索和資訊的流覽，因此推理的速度和效率要求較高，所獲得資訊的完整性就成為次要需求。本例中對應的是推理後 WhoHasChild 類下的實例為 Marry (實例 Marry 是 People 同時又存在 hasChild 關係，則 Marry 是 Parent，而 Parent 又是 WhoHasChild 的子類，則 Marry 是 WhoHasChild 類的實例)，這就是實例規類，目的是把一個描述好的實例歸入一個最具體的、最能反映它特徵的類中。

上述推理過程採用 Jena API 實現代碼如下(只寫出核心代碼)：

```
public class Reason {
//Racer 推理機所需的 URL
  string REASONER_URL="http://localhost:8080";
ProtegeOWLReasoner  reasoner=null;
//該 OWLModel 模型是從 owl 檔中讀取建立的
//該檔對應的本體模型是在 Protégé 中建立
```



```
public void reason(OWLModel model)
throws DIGReasonerException{
//如果連接不上推理機直接返回
if(this.isConnected(model)==null)  return;
//得到推理後 People 下的子類
Collection inferredSubClass=
reasoner.getSubclasses(peopleClass,null);
for (Iterator iter = inferredSubClass.iterator();
iter.hasNext();) {
OWLNamedClass cls =
(OWLNamedClass) iter.next();
System.out.println("inferredSubClass
name:"+cls.getName());
//得到某類所有實例(包含推理出來的)
Collection Individuals=
reasoner.getIndividualsBelongingToClass(cls,null);
for (Iterator iterator = Individuals.iterator();
iterator.hasNext();) {
OWLIndividual element =
(OWLIndividual) iterator.next();
System.out.println("—individualsBelongingTo
Classname:"+element.getName());
}
}
//推理機是否鏈結成功
Private ProtegeOWLReasoner
isConnected(OWLModel model){
//從推理機管理工廠中獲得推理機實例
ReasonerManager reasonerManager
=ReasonerManager.getInstance();
reasoner=
reasonerManager.getReasoner(model,true);
reasoner.setURL(REASONER_URL);
//連接推理機是否成功注意 Race 推理機一定要啟
動
if(reasoner.isConnected()){
DIGReasoner Identityidentity
=reasoner.getIdentity();
return reasoner;
}else{//列印連接失敗資訊
return null;}
}
}
```

運行結果:

```
inferredSubClass name:Woman
inferredSubClass name:Man
inferredSubClass name:WhoHasChild
--individualsBelongingToClass name:Marry
--individualsBelongingToClass name:Terry
```

可以看出Marry和Terry是有孩子的People.

6. 結論和展望

本文我們概括的介紹了在 Web 語義推理中所需要理解掌握的一些知識如描述邏輯、本體描述語言 OWL、本體建模工具 Protégé、推理機 Racer 和開發工具包 Jena。在此基礎上我們用一個簡單例子說明了這些知識在推理中的作用。需要指出的是：在應用本體時,除了這些基本的推理，還可以定義很多附加規則來實現高層次的推理，這些上層次的推理主要是針對具體領域、具體應用的。本例中的實例是存儲在 owl 檔中的也就是將實例和概念存儲在一起，在 ontology 中針對海量實例可以把實例和其他概念分開存儲，例如把實例存儲在關聯資料庫中，Jena 本身提供了持久化存儲方案,還有開源項目 The OWL Instance Store 也提供了針對 ABox 推理的海量資料問題的解決方案[13]。推理作為知識表示的核心在語義 Web 中扮演著重要的角色，隨著 OWL 語言在語義 Web 領域中廣泛應用，會有很多基於這種語言的推理技術出現，這將會進一步引發 Web 上本體應用的快速發展，各種推理的應用將會被用來輔助建立和應用 Web 上的本體。Jena 作為惠普實驗室開發的開放資源是實現語義 Web 應用系統的有力工具，Stanford 大學開發的 Protégé 工具使我們更加方便的建立本體模型，借助這些工具我們可以加深對 ontology 等研究物件的理解和認識，推動語義 Web 技術的發展。

REFERENCES

[1] Gruber T R. A Translation Approach to Portable Ontology Specifications[S]. Knowledge Acquisition, 5/1993: 199−200.

[2] Chen Kai,He Keqing,Li Bing,Liang Peng. Research on Ontology Modeling Using Object-Oriented Technology.Computer Engineering and Applications, 2005, 41(2): 40-43 (in Chinese)(陳凱, 何克清, 李兵, 梁鵬. 面向對象的本體建模研究[J], 電腦工程與應用[J],2005, 41(2): 40-43).

[3] Asunción Gómez-Pérez, Oscar Corcho. “Ontology Specification Languages for the Semantic Web. [S]” IEEE Intelligent Systems, 2002, 17(1), 54−60.

[4] Gao Qi,Chen Huajun.Comparison And Analysis Of Web Ontology Languages And Reasoning. Computer Applications and Software,2004,21(10) (in Chinese)(高琦, 陳華鈞. 互聯網Ontology語言和推理的比較和分析[J]. 電腦應用與軟體,2004, 21(10)).

[5] Frank Manola, Eric Miller. RDF Primer[EB/OL]. W3C Working Draft, 2004-02-10. http://www.w3.org/TR/2004/REC-rdf- primer -20040210/.

[6] Bechhofer S, Horrocks I, Goble C, *et al*. OILEd: A reasonable ontology editor for the semantic web[R]. Joint German/Austrian conference on Artificial Intelligence, 2/2001, 174: 396−408.

[7] Knublauch, H., Fergerson, R. W., Noy, N. F., Musen, M. A.: The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications [R], Third International Semantic Web Conference - ISWC 2004, Hiroshima, Japan, 2004.

[8] restol.Jena2.ASemanticWebFramework[EB/OL].http://Jena.Sourceforge.net (Accessed Oct.12, 2004).

[9] V. Haarslev, R. Möller, A.-Y. Turhan. RACE User’s Guide and Reference Manual Version 1.1[R]. Technical report, University of Hamburg, Computer Science Department, 1999.

[10] Domingue J. Tadzebao. WebOnto: Discussing, browsing, and editing ontologies on the web[R]. Proceedings of the 11th Knowledge Acquisition for Knowledge-Based Systems. Workshop, 1998.

[11] I. Horrocks, U. Sattler, S. Tobies. Reasoning with individuals for the description logic shiq[J]. In D. MacAllester, editor, Proc. of the 17th Conf. on Automated Deduction (CADE-17), number 1831 in Lecture Notes in Computer Science, Germany, 2000. Springer-Verlag.

[12] Mike Dean, Guus Schreiber. OWL Web ontology language reference [EB/OL]. W3C Working Draft, http://www.w3c.org/TR/2003/WD-owl-ref-200330331/, 2003.

[13] Instance Store website. [EB/OL] http://instan- cestore. man.ac.uk.

[14] Tao Wan,LIAO Ahu-mei.Analysis and research of current ontology editing tools.Computer Engineering and Design,2005,26(3)(in Chinese)陶皖，廖述梅.當前本體編輯工具的分析與研究[J].電腦工程與設計,2005,26(3)).

