Università degli Studi di Roma "La Sapienza"
Facoltà di Ingegneria – Corso di Laurea Specialistica in Ingegneria Informatica
Corso di Metodi Formali nell'Ingegneria del Software
Prof. Toni Mancini

Esercizio **E.III.20060920**

versione del 4 luglio 2007

Nell'applicazione da realizzare sono di interesse i docenti e i corsi da essi insegnati. Ogni docente insegna al più un corso. Alcuni docenti sono professori. Un professore può essere titolare di corsi, e quando lo è li insegna. Ogni professore è titolare di almeno due corsi.

1. Rappresentare i requisiti mediante un opportuno diagramma Uml.
2. Rappresentare i requisiti in un linguaggio formale fra quelli considerati durante il corso.
3. Considerando la rappresentazione formale, quali deduzioni interessanti possono essere effettuate?
4. Quale strumento software fra quelli utilizzati nel corso utilizzereste per dimostrare tali deduzioni?
5. Per lo strumento software scelto, fornire il file di input e l'output atteso.

Una possibile soluzione è riportata nelle pagine seguenti.

Soluzione

1. UML: cfr. figura 1 (diagramma delle classi).

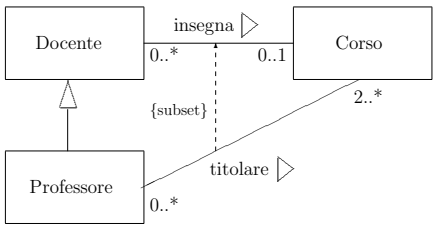

Figura 1: Diagramma UML per la domanda

2. Rappresentiamo i requisiti in logica del prim'ordine:

$$
\begin{array}{rl}
 & \text{// ISA fra classi} \\
\forall X & Professore(X) \rightarrow Docente(X), \\
 & \text{// Associazione } \textit{Insegna} \\
\forall XY & Insegna(X,Y) \rightarrow Docente(X) \wedge Corso(Y), \\
 & \text{// Associazione } \textit{Titolare} \\
\forall XY & Titolare(X,Y) \rightarrow Professore(X) \wedge Corso(Y), \\
 & \text{// Subset fra associazioni} \\
\forall XY & Titolare(X,Y) \rightarrow Docente(X,Y), \\
 & \text{// Vincoli di molteplicità associazioni} \\
 & \text{// } \textit{Insegna}\text{: 0..1} \\
\forall XYZ & Insegna(X,Y) \wedge Insegna(X,Z) \rightarrow Y = Z, \\
 & \text{// } \textit{Titolare}\text{: 2..*} \\
\forall X & Professore(X) \rightarrow \exists YZ\ Titolare(X,Y) \wedge Titolare(X,Z) \wedge Y \neq Z,
\end{array}
$$

3. La classe *Professore* è inconsistente (o insoddisfacibile). Di conseguenza i requisiti contengono un errore (ad esempio, dobbiamo cambiare i vincoli di molteplicità delle associazioni).

   Formalmente, detta Φ la formula di cui al punto 2, vale la seguente implicazione logica:

$$\Phi \models \forall X\ \neg Professore(X).$$

4. Per dimostrare la deduzione di cui al punto 3 possiamo usare OTTER, chiedendo una refutazione.

5. Il file OTTER è il seguente:

```
%%% File: docenti.ott
%%% Time-stamp: "2006-08-01 11:11:52 cadoli"
%%% Formato: file di input per OTTER 3.3

set(auto).
formula_list(usable).

%% ASSOCIAZIONE Insegna
all X Y (Insegna(X,Y) -> Docente(X) & Corso(Y)).
%% ASSOCIAZIONE Titolare
all X Y (Titolare(X,Y) -> Professore(X) & Corso(Y)).

%% ISA FRA CLASSI
all X (Professore(X) -> Docente(X)).

%% SUBSET FRA ASSOCIAZIONI
all X Y (Titolare(X,Y) -> Insegna(X,Y)).

%% VINCOLI DI MOLTEPLICITÀ ASSOCIAZIONE Insegna
all X Y Z (Insegna(X,Y) & Insegna(X,Z) -> Y=Z).                 % {0..1}
%% VINCOLI DI MOLTEPLICITÀ ASSOCIAZIONE Titolare
all X (Professore(X) -> (exists Y Z (Titolare(X,Y) &
                                     Titolare(X,Z) & Y != Z))). % {2..*}

%% CONSEGUENZE LOGICHE:
%% La classe Professore è insoddisfacibile
exists X (Professore(X)).

end_of_list.
```

Ci si aspetta che OTTER dimostri una contraddizione, ovvero che generi la clausola vuota.

Infatti, l'output di OTTER comprende:

```
-----> EMPTY CLAUSE at   0.00 sec ----> 38 [hyper,35,10,11] $F.
```