Tabella delle Incertezze

Model Driven Engineering A.A. 2017/2018

1. Introduzione

L'obiettivo del lavoro svolto è quello di creare un plug-in il cui scopo è quello di realizzare un modello che sia conforme ad un meta-modello generale di una tabella e che rappresenti le differenze tra due o più modelli. Tali differenze verranno successivamente utilizzate per creare un modello conforme al meta-modello delle incertezze.

Per raggruppare le differenze di un numero *n* di modelli che siano conformi ad uno stesso meta-modello in una struttura unica che sia leggibile e facilmente interpretabile, è stata creata una tabella delle differenze.

In tale struttura, sono state create tante colonne quanti sono i modelli che sono analizzati in modo da poter confrontare in modo semplice ed efficace tutti gli elementi, ed i relativi attributi, di ogni modello.

Per la creazione delle righe invece, la strategia adottata è stata quella di esaminare separatamente ogni modello e per ognuno di essi analizzare tutti i suoi elementi. Per questi ultimi, vengono analizzati tutti gli attributi e le referenze ad essi associate creando una riga all'interno della tabella avente:

- Tipo dell'elemento (eClass),
- nome dell'elemento,
- xmi_ID dell' elemento,
- eventuale nome dell'attributo o referenza.

Inoltre, se un determinato elemento, o sotto elemento, è stato precedentemente inserito questo non viene aggiunto nuovamente come riga.

Il formato della colonna contenente le chiavi appena descritte è il seguente:

tipoElemento.nomeElemento.xmi_ID.[nome attributo/referenza]

Family.Flanders.5.familiesInverse

Per identificare intuitivamente le differenze all'interno della tabella, è necessario analizzare i parametri di ogni riga scorrendo tutte le colonne, e quindi tutti i modelli presi in analisi.

Andando a vedere quelli che sono i valori di ogni colonna, se lo stesso valore è presente per più righe si può affermare che tali elementi sono appartenenti ad uno stesso modello.

Osservando due colonne differenti invece, se su una riga, in entrambe le colonne, è presente lo stesso valore sta a significare che i due modelli appartenenti alle rispettive colonne hanno tale elemento in comune (quindi non ci sono differenze per quell'elemento). Se i due valori sono differenti invece, allora i due modelli non hanno tale elemento in comune e questo può individuare un punto di incertezza.

Inoltre, all'interno di una cella il valore *'null'* rappresenta la mancanza del rispettivo elemento, attributo o referenza.

Ad esempio, nella tabella mostrata in figura 4 è possibile notare che il modello '*sampleFamily10*' contiene due diverse famiglie *Flanders,* mentre il modello *'sampleFamily11'* include una sola famiglia *Flanders*.

2. Struttura

Il plug-in implementato può essere suddiviso in due parti differenti:

1. La componente che attiva quest'ultimo da interfaccia grafica (Run Configuration 'Create Table'),
2. La componente che si occupa di creare il modello della tabella in formato XMI.

Il punto 1 viene usato per settare le configurazioni di run del plug-in, dando la possibilità all'utente di selezionare la cartella contenente i modelli input da analizzare e la cartella all'interno della quale salvare il modello di output.

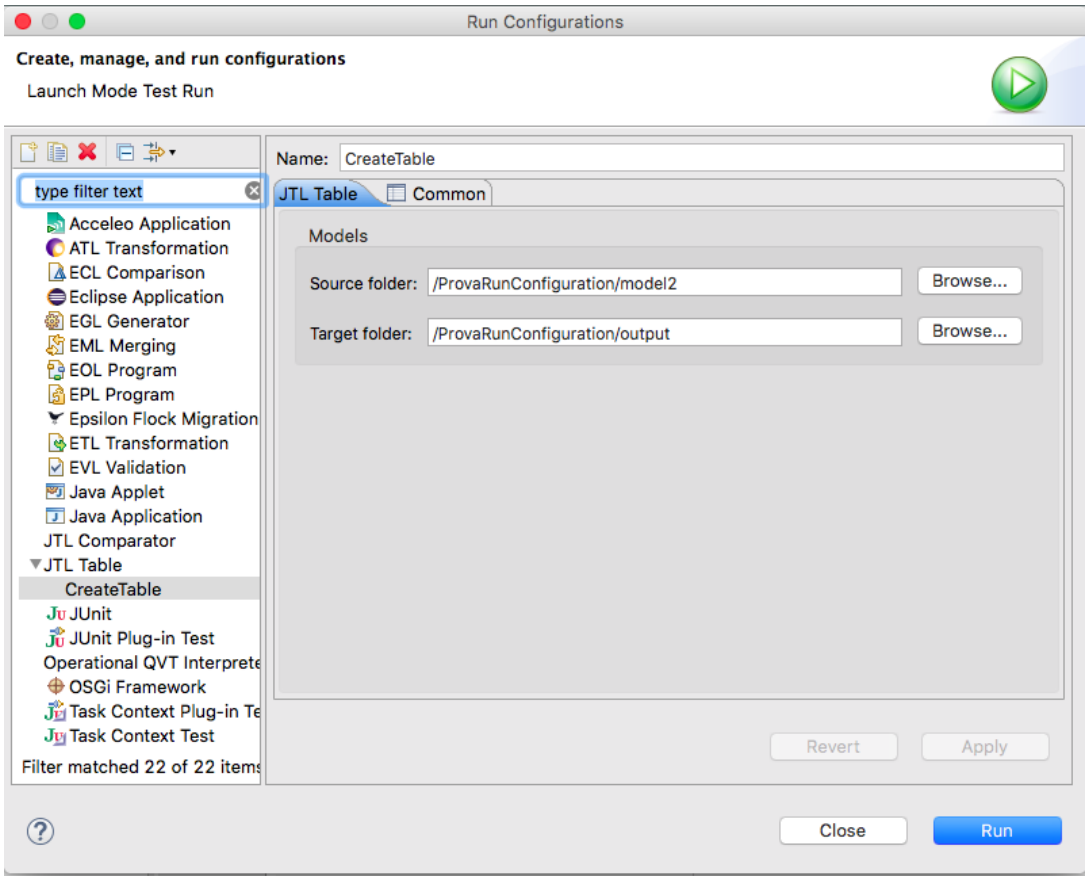

Figura 1: Run Configuration

Per implementare la Run Configuration, vengono specializzate le classi contenute nelle seguenti librerie:

- *org.eclipse.debug.core:* Libreria responsabile della creazione e della configurazione della Launch Configuration.
- *org.eclipse.debug.ui:* Libreria che fornisce le funzionalità di base per la visualizzazione della Lunch Configuration.

Le seguenti classi sono state create per l'implementazione del plug-in:

- CustomTab.java: Sottoclasse di AbstractLaunchConfigurationTab che definisce le caratteristiche generali di una form. Nella classe CustomTab vengono definite il layout, l'etichetta e gli eventi a cui deve rispondere una tab rispettivamente attraverso l'oggetto Label, GridData e SelectionAdapter.

- CustomTabGroup.java: Sottoclasse di AbstractLaunchConfigurationTab che definisce le caratteristiche generali di un gruppo di form. All'interno di essa vengono raggruppati i tab passandogli le istanze di CustomTab nella seguente riga di codice: new ILaunchConfigurationTab[] { new CustomTab(), new CommonTab() }

- LaunchConfigurationAttributes.java: Interfaccia che contiene gli identificativi dei campi della form della Run Configuration. Ad ogni campo viene assegnato un id che permette di ottenere il valore inserito dall'utente.

- CustomLunchConfigurationDelegate.java: Sottoclasse di LaunchConfigurationDelegate che fornisce i metodi per calcolare l'esecuzione del codice all'avvio del plug-in. In particolare, nella sottoclasse è richiamato il codice per la creazione della tabella delle incertezze.

Il punto 2 invece, è implementato attraverso le classi '*Table*' e '*GenerateTable*'.

La classe *Table* serve per generare una mappa contenente tutte le informazioni per creare la struttura della tabella delle incertezze. Inizialmente, viene chiamato il metodo *createTable* che permette di scorrere tutti i modelli in input nella cartella e ricavare tutti gli elementi del modello corrente. È dunque presente un ciclo che scorre e carica tutti i modelli xmi presenti nella cartella input indicata dall'utente nella Run Configuration e crea un iteratore per scorrere tutti gli elementi presenti in ciascun file.

Successivamente viene chiamato il metodo *createModelMap* che permette di creare una mappa avente come chiave il nome dell'oggetto corrente e come valore una lista contente i valori di quell'elemento, attraverso l'oggetto Java *TreeMap<String, LinkedList<EObject>>.* Tale metodo andrà dunque a scorrere tutti gli elementi ed inserire le chiavi relative a ciascun elemento nella mappa risultante. Le chiavi verranno costruite con il principio descritto nell'introduzione, in cui gli *xmi_ID* sono ricavati tramite il metodo *retrieveID*. Inoltre, se un determinato elemento, o sotto elemento, è stato precedentemente inserito, questo non viene aggiunto nuovamente nella mappa. Per ogni elemento o sotto elemento, nella corrispettiva chiave, verranno inseriti i suoi valori.

Nella mappa risultante verranno dunque inserite tutte le informazioni di ciascun elemento per ogni singolo modello.

Tale mappa verrà poi utilizzata per la creazione di un ulteriore mappa del tipo *TreeMap<Integer, TreeMap<String, LinkedList<EObject>>>* in cui le chiavi saranno gli ID di ciascun modello input e i valori le mappe risultanti dalla chiamata del metodo *createModelMap.*

Infine, verrà chiamato il metodo *parseMap* che serve per creare la struttura da trasformare in tabella. Tale struttura sarà un ulteriore mappa del tipo *TreeMap<String, LinkedList<LinkedList<EObject>>>* in

cui la chiave rappresenta la riga di ciascun elemento e ogni valore è implementato tramite una doppia lista. L'insieme delle chiavi formerà la prima colonna della tabella delle incertezze, mentre la doppia lista conterrà gli elementi di ciascuna cella della tabella risultante.

Successivamente, verrà quindi generato un oggetto di tipo *GenerateTable* che permette la generazione del file *xmi*, preso in input il *TreeMap* descritto precedentemente.

Scorrendo tutti gli elementi contenuti in essa, viene generato un modello conforme al meta-modello '*Table.ecore*' visibile in figura 2**.** Tale struttura è organizzata come segue:

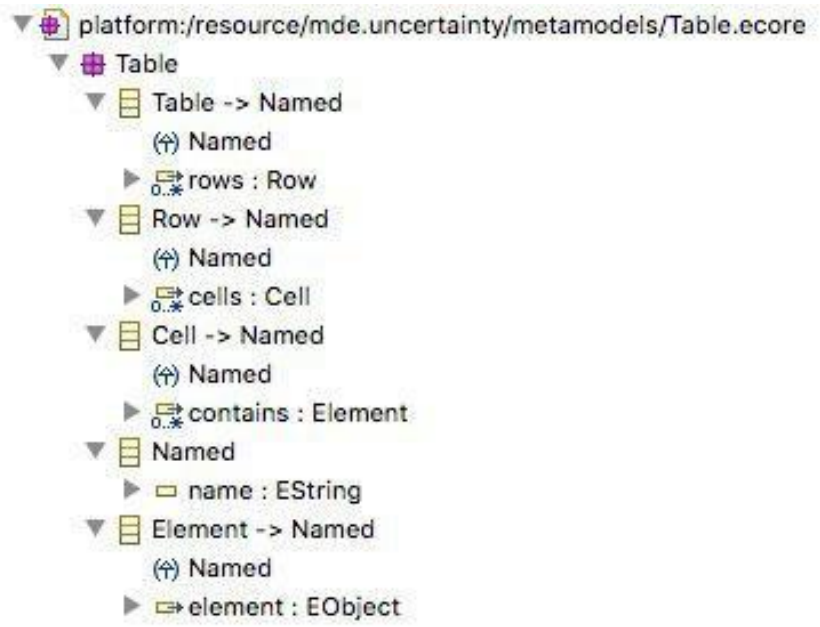

Figura 2: Meta-Modello di una tabella

Una Table può contenere zero o più righe e a sua volta ogni riga può contenere zero o più Cell (rappresentanti le colonne della tabella). Infine, ogni *Cell* può contenere zero o più elementi in modo da poter correttamente rappresentare il caso in cui ci siano più referenze. La tabella risultante verrà generata tramite il metodo *createTableModel.*

Un modello di tabella in formato xmi è visibile nella figura 3.

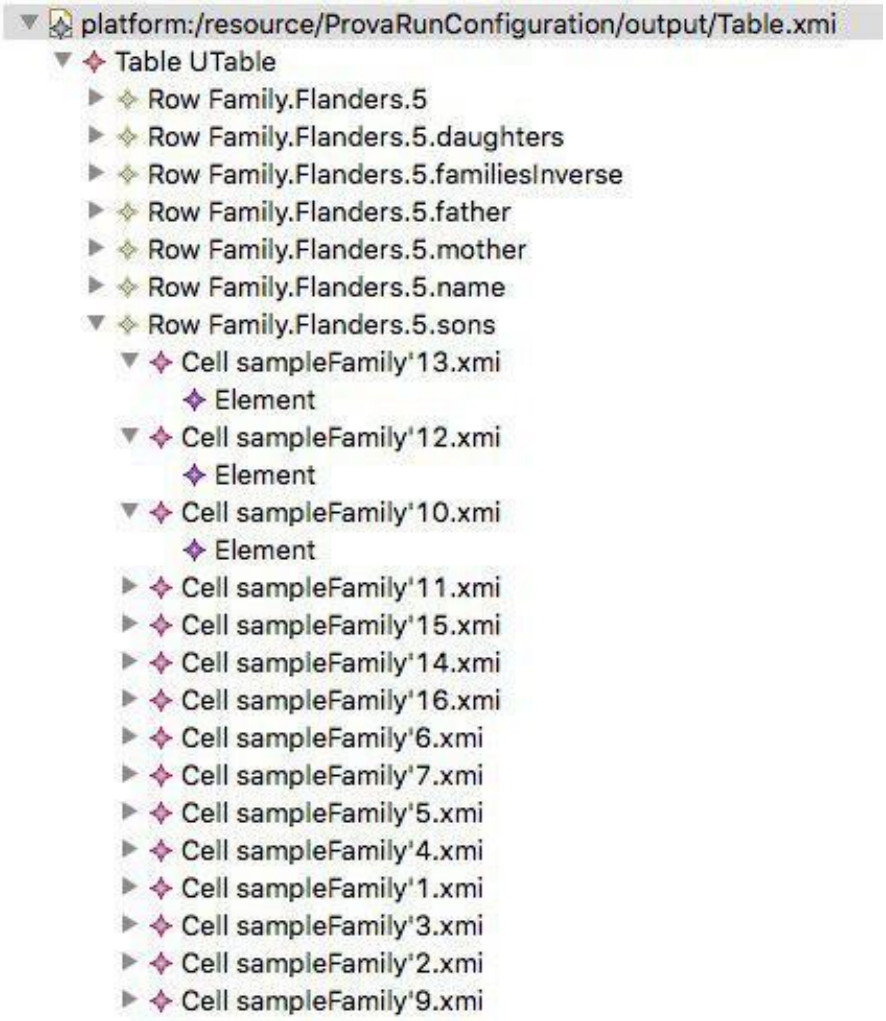

Figura 3: Modello Tabella in formato xmi

	sampleFamily'13	sampleFamily'12	sampleFamily'10	sampleFamily'11	sampleFamily'15	sampleFamily'14	sampleFamily'16	sampleFamily'6	sampleFamily'7	sampleFamily'5	sampleFamily'4
Family.Flanders.5	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders
Family.Flanders.5.daughters	null	null	null	null	null	null	null	null	null	null	null
Family.Flanders.5.sons	Rod	Rod	Rod	Rod, Tod	Rod	Rod, Tod	Rod	Rod	Rod	Rod, Tod	Rod
Family.Flanders.5.father	null	null	null	null	null	null	null	null	null	null	Tod
Family.Flanders.5.mother	null	null	null	null	null	null	null	null	null	null	null
Family.Flanders.5.familiesInverse	FamilyRegistry	FamilyRegistry	FamilyRegistry	FamilyRegistry	FamilyRegistry	FamilyRegistry	FamilyRegistry	FamilyRegistry	FamilyRegistry	FamilyRegistry	FamilyRegistry
Family.Flanders.5.name	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders
Family.Flanders.8	Flanders	Flanders	Flanders	null	Flanders	null	Flanders	Flanders	Flanders	null	null
Family.Flanders.8.daughters	null	null	null	null	null	null	null	null	null	null	null
Family.Flanders.8.sons	Todd	null	Todd	null	null	null	Todd	null	Todd	null	null
Family.Flanders.8.father	null	Todd	null	null	Todd	null	null	null	null	null	null
Family.Flanders.8.mother	null	null	null	null	null	null	null	null	null	null	null
Family.Flanders.8.familiesInverse	FamilyRegistry	FamilyRegistry	FamilyRegistry	null	FamilyRegistry	null	FamilyRegistry	FamilyRegistry	FamilyRegistry	null	null
Family.Flanders.8.name	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders
FamilyMember.Rod.7	Rod	Rod	Rod	Rod	Rod	Rod	Rod	Rod	Rod	Rod	Rod
FamilyMember.Rod.7.daughtersInverse	null	null	null	null	null	null	null	null	null	null	null
FamilyMember.Rod.7.fatherInverse	null	null	null	null	null	null	null	null	null	null	null
FamilyMember.Rod.7.motherInverse	null	null	null	null	null	null	null	null	null	null	null
FamilyMember.Rod.7.name	Rod	Rod	Rod	Rod	Rod	Rod	Rod	Rod	Rod	Rod	Rod
FamilyMember.Rod.7.sonsInvers	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders	Flanders

Figura 4: Tabella delle differenze human readable

3. Modello delle incertezze

Per la creazione del modello delle incertezze è stato utilizzato il linguaggio di trasformazione ATL. Per il nostro lavoro è stato preso in considerazione il caso di studio delle Famiglie. La trasformazione implementata potrà poi essere utilizzata per sviluppare un High Order Transformation in modo tale da creare modelli di incertezze generici e non specifici per il modello delle Famiglie.

La trasformazione ATL sviluppata prenderà in input un modello della tabella delle differenze creato tramite Java e genererà un modello di incertezza per le famiglie.

La trasformazione è stata chiamata Table2UFamily.atl. Nella prima parte di esse possiamo trovare tutti gli helpers mentre nella seconda parte troviamo le regole di trasformazione.

La prima regola applicata alla nostra trasformazione è "RowToFamilyRegister" che si occupa di trasformare una riga della tabella delle incertezze in un familyRegister. "RowToFamilyRegister" verrà applicata sempre come prima regola in quanto siamo sicuri di dover creare un familyRegister. Tramite questa regola verranno create famiglie certe o famiglie incerte ricorrendo alle lazy rules RowToFamilies e RowToUFamilies.

Procediamo ad analizzare RowToFamilies. Questa regola ci permette di creare una famiglia certa, nel caso ce ne fosse una. Per famiglia certa intendiamo una famiglia che si ripete in ogni modello allo stesso modo. Con questa regola creiamo anche i membri della famiglia andando di nuovo a distinguere se sono membri certi o membri incerti. I membri vengono creati tramite le lazy rules RowToFamilyMember e RowToUFamilyMember. Se il membro è certo andiamo semplicemente a recuperare l'oggetto corrispondente ed inserirlo nella rispettiva famiglia. Se il membro è incerto andiamo a richiamare la regola "CellToFamilyMember" che recupera l'oggetto e lo inserisce in un UFamilyMember nel campo uleft.
Quando dobbiamo creare una famiglia incerta invece applichiamo la regola RowToUFamilies che ci permette di creare tutti i membri certi ed incerti come definito precedentemente per una famiglia incerta. La famiglia incerta è una famiglia che non si ripete in tutti i modelli, ovvero che è presente solo in alcuni.