# SI-Designer: un tool di ausilio all'integrazione intelligente di sorgenti di informazione*

D. Beneventano[1,2], S. Bergamaschi[1,2], A. Corni[1,2], R. Guidetti[1], and G. Malvezzi[1]

[1] Università di Modena e Reggio Emilia, DSI, Via Campi 213/B, 41100 Modena, Italy
{domenico.beneventano,sonia.bergamaschi,corni.alberto}@unimo.it
{guidetti, malvezzi}@sparc20.dsi.unimo.it
[2] CSITE-CNR Bologna V.le Risorgimento 2, 40136 Bologna, Italy

**Sommario** SI-Designer (Source Integrator Designer) è un tool di supporto al progettista per l'integrazione semi-automatica di schemi di sorgenti eterogenee (relazionali, ad oggetti e semistrutturate). Realizzato nell'ambito del progetto MOMIS, SI-Designer esegue l'integrazione seguendo un *approccio semantico*, che fa uso di tecniche intelligenti, basate sulla *Description Logics* OLCD, di tecniche di clustering e di un linguaggio object-oriented per rappresentare le informazioni estratte ed integrate, $ODL_{I^3}$, derivato dallo standard ODMG. Partendo dalle descrizioni delle sorgenti in $ODL_{I^3}$ (gli *schemi locali*) SI-Designer assiste il progettista nella creazione di una vista integrata di tutte le sorgenti (*schema globale*) anch'essa espressa in linguaggio $ODL_{I^3}$.

## 1 Introduzione

Nel corso degli ultimi anni è diventata sempre più rilevante la necessità di accedere ad informazioni distribuite e contestualmente anche il problema dell'integrazione di informazioni provenienti da sorgenti eterogenee. Le imprese si sono via via dotate di sistemi per l'archiviazione delle informazioni costruendo sistemi informativi contenenti dati correlati tra loro ma spesso ridondanti, eterogenei e non sempre consistenti. D'altra parte l'esplosione della rete, sia a livello di internet che intranet, ha accelerato l'esigenza di condivisione e del reperimento delle informazioni presenti sui singoli sistemi di archiviazione dati, giungendo ad una visione integrata in modo da eliminare incongruenze e ridondanze. I problemi che devono essere affrontati in questo ambito sono principalmente dovuti ad eterogeneità strutturali e applicative, nonché alla mancanza di una ontologia comune, che porta a differenze semantiche tra le fonti di informazione. A loro volta, queste differenze semantiche possono dare origine a diversi tipi di conflitti, che vanno dalle semplici incongruenze nell'uso dei nomi (quando nomi differenti sono utilizzati in sorgenti diverse per identificare gli stessi concetti) a conflitti strutturali (quando modelli/primitive diversi sono utilizzati per rappresentare le stesse informazioni).

Il progetto MOMIS [1, 2] (*Mediator envirOnment for Multiple Information Sources*) ha come obiettivo l'integrazione intelligente delle informazioni in sorgenti di dati sia strutturate che semistrutturate. SI-Designer (*Source Integrator Designer*) è un tool di supporto al progettista per l'integrazione semi-automatica di schemi di sorgenti eterogenee (relazionali, ad oggetti e semistrutturate). Realizzato nell'ambito del progetto MOMIS, SI-Designer esegue l'integrazione seguendo un *approccio semantico*, che fa uso di tecniche intelligenti, basate sulla *Description Logics* OLCD [3,

* La ricerca è stata parzialmente finanziata dal progetto INTERDATA - MURST 40% 98-99.

4] (*Object Language with Complements allowing Descriptive cycles*), di tecniche di clustering e di un linguaggio object-oriented per rappresentare le informazioni estratte ed integrate, $ODL_{I^3}$ [5], derivato dallo standard ODMG. Partendo dalle descrizioni delle sorgenti in $ODL_{I^3}$ (gli *schemi locali*) SI-Designer assiste il progettista nella creazione di una vista integrata di tutte le sorgenti (*schema globale*) anch'essa espressa in linguaggio $ODL_{I^3}$.

Lo schema globale viene ottenuto in fasi successive, creando un *Common Thesaurus* di relazioni intra ed inter-schema. Le sorgenti da integrare vengono descritte attraverso il linguaggio $ODL_{I^3}$ e, utilizzando le tecniche di inferenza di OLCD, vengono estratte relazioni intensionali intra-schema che sono inserite nel *Common Thesaurus*. Dopo questa fase iniziale il *Common Thesaurus* viene arricchito aggiungendo relazioni inter-schema ottenute:

- utilizzando il sistema lessicale WordNet [6, 7], che identifica le affinità tra concetti inter-schema sulla base di lessico/significato delle loro denominazioni;
- utilizzando il sistema ARTEMIS [5] che calcola le affinità strutturali di concetti inter-schema.

Partendo dal *Common Thesaurus* ottenuto ed usando ancora le tecniche di inferenza di OLCD e le tecniche di clustering di ARTEMIS, si definisce uno schema globale che contiene la visione d'insieme delle sorgenti integrate.

Nel presente lavoro viene descritto il tool, SI-Designer, che assiste il progettista in tutte le fasi di costruzione del *Common Thesaurus* e dello schema globale. In particolare, rispetto a lavori precedenti [1, 2], la sequenza delle fasi di costruzione dello schema globale è stata modificata traendo vantaggio dall'approfondimento dell'interazione con WordNet, che viene presentato diffusamente nel lavoro, e dall'utilizzo di ARTEMIS anche nella fase di costruzione del *Common Thesaurus*.

Il lavoro è così articolato. Nella sezione 2, vengono presentate le architetture di MOMIS e di SI-Designer e viene introdotto un esempio di riferimento; nella sezione 3 e 4 vengono descritte, rispettivamente, la costruzione del *Common Thesaurus* e dello schema globale. Infine, la sezione 5 contiene alcuni commenti conclusivi.

## 2 L'architettura di MOMIS

MOMIS è stato progettato per fornire un accesso integrato ad informazioni eterogenee memorizzate sia in database di tipo tradizionale (e.g. relazionali, object-oriented) o file system, sia in sorgenti di tipo semistrutturato. Seguendo l'architettura di riferimento $I^3$ [8], in MOMIS si possono distinguere quattro componenti principali per la fase di integrazione delle sorgenti (Fig. 1):

1. *Wrapper*: posti al di sopra di ciascuna sorgente, sono i moduli che rappresentano l'interfaccia tra il mediatore e le sorgenti locali di dati. La loro funzione è duplice:
   - in fase di integrazione, forniscono la descrizione delle informazioni in essa contenute. Questa descrizione viene fornita attraverso il linguaggio $ODL_{I^3}$;
   - in fase di query processing, traducono la query ricevuta dal mediatore (espressa quindi nel linguaggio comune di interrogazione $OQL_{I^3}$, definito a partire dal linguaggio OQL) in una interrogazione comprensibile dalla sorgente stessa. Devono inoltre esportare i dati ricevuti in risposta all'interrogazione, presentandoli al mediatore attraverso il modello comune di dati utilizzato dal sistema.
2. *Mediatore*: è il cuore del sistema, ed è composto da due moduli distinti.

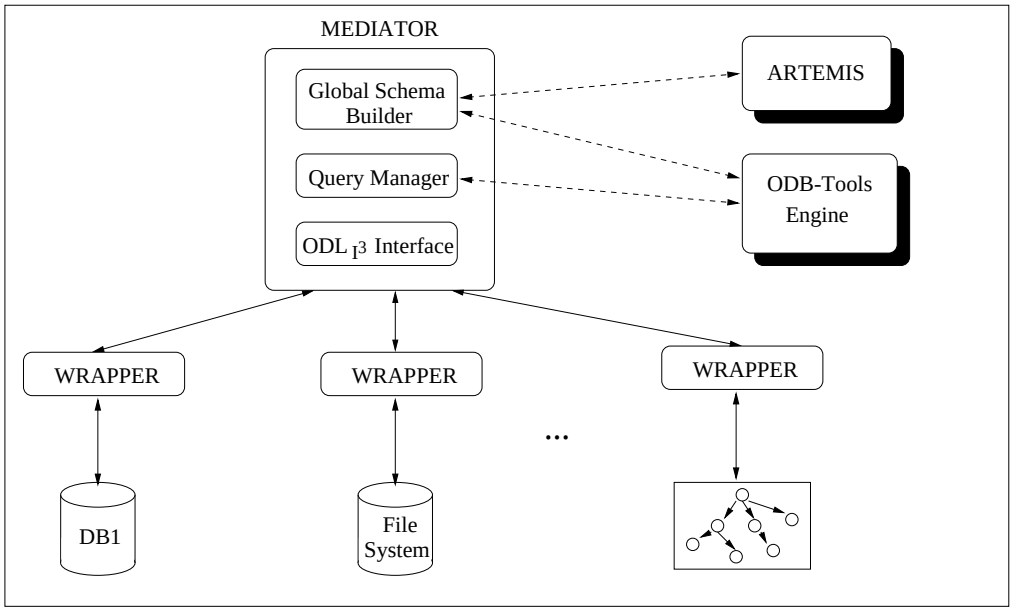


**Figura 1.** MOMIS Architecture

- *Global Schema Builder (GSB)*: è il modulo di integrazione degli schemi locali, che, partendo dalle descrizioni delle sorgenti espresse in $\text{ODL}_{I^3}$, genera un unico schema globale da presentare all'utente.
- *Query Manager* (QM): è il modulo di gestione delle interrogazioni. In particolare, genera le query in linguaggio $\text{OQL}_{I^3}$ da inviare ai wrapper partendo dalla singola query formulata dall'utente sullo schema globale. Servendosi di tecniche delle *Description Logics* di ODB-Tools il QM genera automaticamente la traduzione della query sottomessa nelle corrispondenti sub-query delle singole sorgenti.

3. SI-Designer, un tool di ausilio al progettista per l'integrazione.
4. *ODB-Tools Engine*, un tool basato sulle *Description Logics* OCDL [3, 4] che compie la validazione di schemi e l'ottimizzazione di query [9–11].
5. *ARTEMIS-Tool Environment*, un tool basato sulle tecniche di clustering *affinity-based* che compie l'analisi ed il clustering delle classi $\text{ODL}_{\text{I}^3}$ [5].

### 2.1 L'architettura di SI-Designer

L'integrazione delle sorgenti realizzata dal *Global Schema Builder* si basa sull'individuazione di una ontologia, comune alle diverse sorgenti, sottoforma di thesaurus: un insieme di relazioni terminologiche chiamato *Common Thesaurus*. Come mostrato in Fig. 2, *GSB* è composto da due moduli:

- SIM (*Source Integrator Module*): estrae relazioni intensionali intra-schema sulla base della struttura delle classi $\text{ODL}_{\text{I}^3}$ e delle sorgenti relazionali, utilizzando ODB-Tools. Oltre a ciò, il modulo si occupa della "validazione semantica" delle relazioni e ne inferisce di nuove tramite ODB-Tools
- SLIM (*Schemata Lessical Integrator Module*): estrae relazioni intensionali inter-schema tra nomi di attributi e classi $\text{ODL}_{\text{I}^3}$ utilizzando la conoscenza espressa in WordNet.

Il tool SI-Designer (vedi Fig. 2) mette a disposizione una interfaccia per interagire con i moduli SIM, SLIMed ARTEMIS, mostrando, via via, le relazioni estratte ed assistendo il progettista nella creazione del *Common Thesaurus*, cioè di un'ontologia comune alle diverse sorgenti.

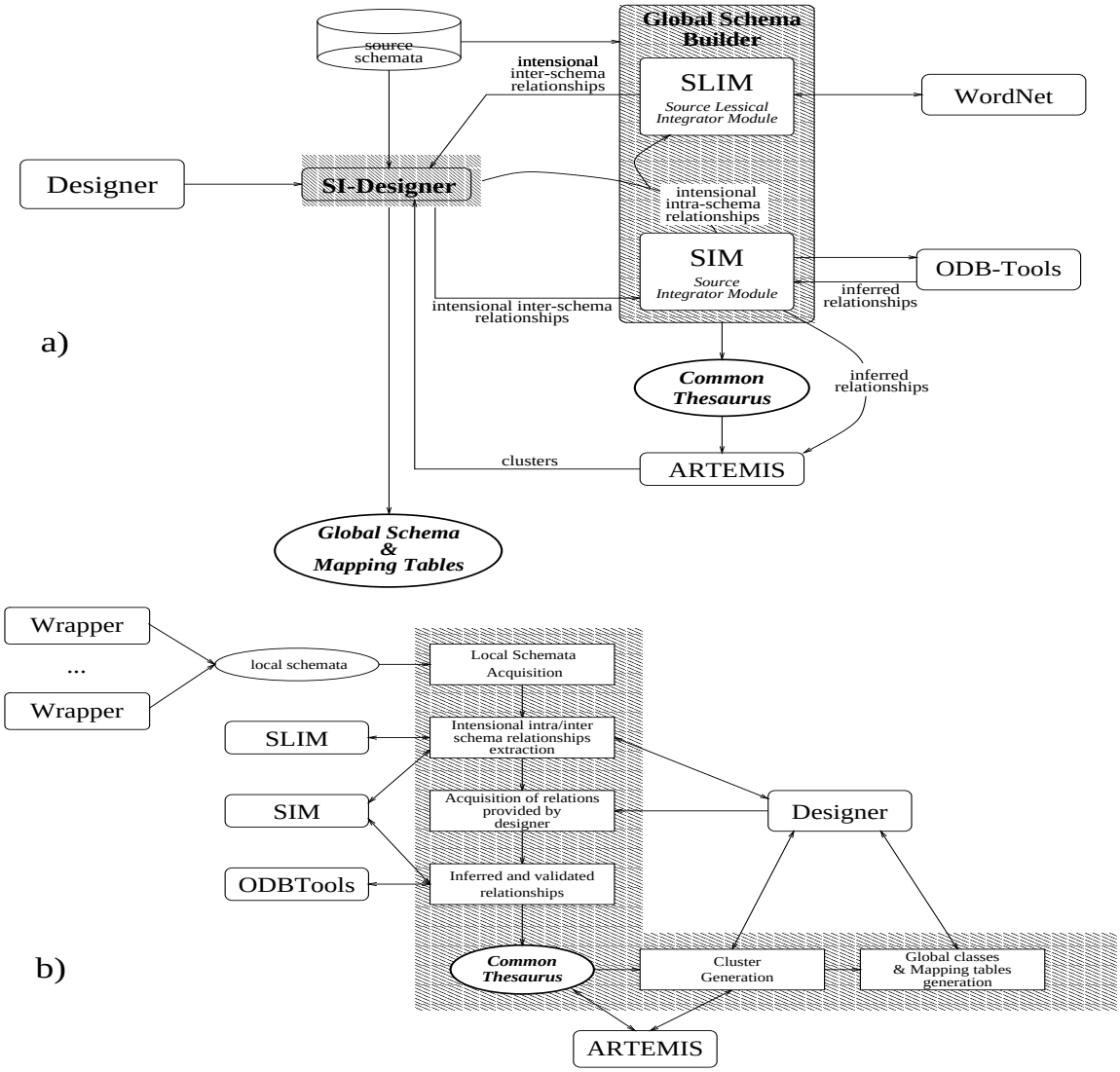


**Figura 2.** a) architettura del GlobalSchemaBuilder, b) architettura dell'SI-Designer.

Costruito il *Common Thesaurus*, SI-Designer utilizza nuovamente il modulo ARTEMIS per individuare, tramite un algoritmo di clustering, gli insiemi disgiunti di classi affini: i *cluster*. Ad ogni cluster corrisponderà una *classe globale* (una vista su tutte le classi affini appartenenti al cluster) caratterizzata da un insieme di attributi globali e da una *mapping-table*. SI-Designer costruisce quindi un insieme di attributi globali per ogni classe globale e la relativa *mapping table*: questo viene realizzato attraverso un processo semi-automatico in cui il progettista deve revisionare l'insieme di attributi globali e la mapping-table ed assegnare un nome ad ogni classe globale, in modo da pervenire ad uno schema globale chiaro. In sostanza, il procedimento di integrazione può essere

suddiviso in due fasi: (**1**) Generazione del *Common Thesaurus*, (**2**) Generazione delle classi globali.

### 2.2 Esempio di riferimento

La Fig. 3 mostra l'esempio che verrà utilizzato in questo articolo per illustrare il funzionamento di SI-Designer. Sono presenti 3 sorgenti eterogenee: una sorgente relazionale (`University`), una sorgente ad oggetti (`Computer_Science`) e una sorgente costituita da un semplice file (`Tax_Position`).

**Sorgente University**

```
Research_Staff(first_name,last_name,relation,email,dept_code,section_code)
School_Member(first_name,last_name,faculty,year)
Department(dept_name,dept_code,budget,dept_area)
Section(section_name,section_code,length,room_code)
Room(room_code,seats_number,notes)
```

**Sorgente Computer_Science**

```
CS_Person(name)
Professor:CS_Person(title,belongs_to:Division,rank)
Student:CS_Person(year,takes:set⟨Course⟩,home_email,phd_email,rank)
Division(description,address:Location,fund,sector,employee_nr)
Location(city,street,number,county)
Course(course_name,taught_by:Professor)
```

**Sorgente Tax_Position**

```
University_Student(name,student_code,faculty_name,tax_fee)
```

**Figura 3.** Esempio di riferimento

La sorgente `University` contiene le informazioni sugli studenti e lo staff di un'università ed è composta da cinque relazioni: `Research_Staff`, `School_Member`, `Department`, `Section` e `Room`. Ogni professore in `Research_Staff` ha associati un dipartimento (`dept_code`) e una sezione (`section_code`). Ogni sezione (in `Section`) è associata ad un'aula (`room_code`) mentre ogni studente (in `School_Member`) è caratterizzato da un `first_name`, un `last_name`, una `faculty` e l'anno di immatricolazione `year`.

La sorgente ad oggetti `Computer_Science` contiene informazioni sulla facoltà di informatica della medesima università descritta dalla sorgente relazionale. Ci sono sei classi: `CS_Person`, `Professor`, `Student`, `Division`, `Location` e `Course`. Le informazioni rappresentate sono simili a quelle della sorgente relazionale: anche qui sono presenti professori e studenti, e un professore è associato ad una `Division`, che è una logica specializzazione di `Department`. Di uno studente possiamo conoscere che corsi frequenta, l'anno di immatricolazione e il suo stato (`rank`), cioè se è in corso, fuori corso, laureando oppure dottorando.

L'ultima sorgente rappresenta un file, `University_Student`, in cui è rappresentata la posizione fiscale di uno studente. In particolare ci sono i campi `name`, `student_code`, `faculty_name` e `tax_fee`.

Nel seguito dell'articolo, le relazioni tra classi o attributi verranno indicate specificando solamente il nome delle classi, o degli attributi, e il tipo di relazione. In realtà SI-Designer, per evitare

ambiguità tra nomi uguali in sorgenti diverse, considera una rappresentazione dei nomi in *dot notation*: per esempio, la relazione tra i due di attributi `dept_code BT belongs_to` viene rappresentata in SI-Designer come: `University.Department.dept_code BT Computer_Science.Division.belongs_to`.

## 3 Generazione del *Common Thesaurus*

Le relazioni terminologiche del *Common Thesaurus* esprimono la conoscenza inter-schema e intra-schema riguardo gli schemi delle sorgenti in esame, sono di tipo intensionale e possono essere espresse per classi ed attributi. Le relazioni sono di tre tipi:

- SYN (SYNonym-of, *relazioni di sinonimia*): definita tra 2 termini che possono essere scambiati senza modificare il concetto rappresentato. Esempio: `faculty SYN faculty_name`.
- BT (Broader-Term, *relazioni di specializzazione)*: definita tra 2 termini $t_i$ e $t_j$ tali che $t_i$ ha un significato più generale di $t_j$; NT (Narrower-Term) è la relazione opposta di BT. Esempio: `CS_Person BT Professor`, che è equivalente a `Professor NT CS_Person`.
- RT (Related-Term, *relazioni di aggregazione*): definita tra 2 termini $t_i$ e $t_j$ tra i quali esiste un legame generico. Esempio: `Research_Staff RT Department`.

La costruzione del *Common Thesaurus* è un processo incrementale durante il quale vengono aggiunte relazioni secondo il seguente ordine: (1) *relazioni intensionali intra-schema*, (2) *relazioni intensionali inter-schema*, (3) *relazioni aggiunte dal progettista* e (4) *relazioni intensionali inferite*. Nel seguito analizzeremo queste relazioni, soffermandoci su quelle inter-schema, la cui estrazione rappresenta uno dei contributi principali del presente lavoro.

### 3.1 Relazioni intensionali intra-schema

Le relazioni intra-schema esprimono le gerarchie di generalizzazione (relazioni BT, NT) e di aggregazione (relazioni RT) più altre relazioni intra-schema estratte dai sorgenti relazionali dalle definizioni di chiavi esterne (relazioni RT, BT).
Per esempio, dalla sorgente ad oggetti `Computer_Science` sono estratte le relazioni di ereditarietà `Professor NT CS_Person` e `Student NT CS_Person`, mentre dalla sorgente relazionale è estratta la relazione `Research_Staff RT Department` derivante dal fatto che l'attributo `dept_code` della relazione `Research_Staff` è foreign key per la relazione `Department`. Queste relazioni sono estratte dal modulo SIM (*Source Integrator Module*) analizzando direttamente le classi $\text{ODL}_{\text{I}^3}$ delle sorgenti relazionali e sfruttando ODB-Tools per estrarre le relazioni dagli altri tipi di sorgente.

### 3.2 Relazioni intensionali inter-schema

Le relazioni inter-schema esprimono un legame semantico tra i nomi delle classi e degli attributi appartenenti a sorgenti diverse (possono essere relazioni SYN, BT, NT o RT).

Queste relazioni vengono estratte sulla base delle relazioni lessicali tra i nomi delle classi e degli attributi, derivanti dal significato delle parole usate: un tipo di conoscenza non esplicitata tramite costrutti di un linguaggio di definizione dei dati, ma impliciti nel nome assegnato dal progettista. È compito del progettista attribuire nomi descrittivi o che possano essere interpretati correttamente; per cui c'è un'incertezza di interpretazione insita nell'ambiguità del linguaggio; Bates [12] scrive *"the probability of two persons using the same term in describing the same thing is less than 20%"*.

Questa conoscenza è comunque un'opportunità che deve essere sfruttata per estrarre relazioni, e non si può pensare di farlo in modo manuale soprattutto con il crescere del numero e della dimensione degli schemi, si è quindi deciso di sperimentare l'utilizzo di WordNet per estrarre e proporre al progettista relazioni intensionali inter-schema.

**Il database WordNet** WordNet è un database lessicale sviluppato dal Cognitive science Laboratory alla Princeton University [6, 7]. WordNet è ispirato alle attuali teorie psicolinguistiche legate alla memoria lessicale umana, ed è considerato la più importante risorsa disponibile per i ricercatori nei campi della linguistica computazionale, dell'analisi testuale, e di altre aree associate. Il database lessicale Wordnet, attualmente giunto alla versione 1.6, conta 64089 lemmi organizzati in 99757 insiemi di sinonimi (*synset*). La semantica lessicale è basata sull'associazione convenzionale fra la forma delle parole (il modo in cui, cioè, vengono pronunciate e scritte) e il concetto/significato che esse esprimono; tale associazione è di tipo molti-a-molti, dando luogo alle proprietà di:

**Sinonimia:** proprietà di un concetto/significato di avere due o più parole in grado di esprimerlo. Un gruppo di sinonimi è detto *synset*. Si noti che per ogni significato/concetto esiste uno ed un solo *synset*. Nel seguito un *synset* è denotato con $s$, mentre $\mathcal{S}$ denoterà l'insieme di *synset*.

**Polisemia:** proprietà di una stessa parola di avere due o più significati.

La corrispondenza tra la forma delle parole e il significato che esprimono viene sintetizzata nella cosiddetta *Matrice Lessicale* $\mathcal{M}$, nella quale le righe riportano il significato delle parole (quindi una riga rappresenta un *synset*) e le colonne rappresentano la forma delle parole (forma/lemma base). Ogni elemento della matrice è una definizione (*entry*), $e = (f, m)$, dove $f$ è la *forma base* e $m$ (*meaning*) è il contatore del significato; ad esempio (`address`, `2`) si riferisce all'indirizzo presso il quale si può trovare una persona mentre (`address`, `1`) si riferisce all'indirizzo di un computer. Nel seguito la forma base ed il meaning di una definizione $e = (f, m)$ verranno denotati rispettivamente con $e.f$ e con $e.m$. Un elemento della matrice $\mathcal{M}$ può essere *nullo* o *indefinito*. Siccome ad un *synset* è associato un'unica riga di $\mathcal{M}$, nel seguito useremo $s \in \mathcal{S}$ come indice di riga di $\mathcal{M}$. In altre parole, gli elementi non nulli della riga $\mathcal{M}[s]$, rappresentano tutti e soli gli elementi di $s$. Nello stesso modo, siccome ad una forma base è associata un'unica colonna di $\mathcal{M}$ nel seguito useremo le forme basi come indice di colonna di $\mathcal{M}$.

**Relazioni semantiche tra i termini degli schemi** Con il concetto di *termine* associamo ad ogni nome di classe e di attributo una definizione. Un *termine* è costituito dalla coppia $t = (n, e)$, dove $n$ denota un nome di classe o di attributo ed $e$ una definizione. Un nome di classe o di attributo $n$ si intende qualificato: un nome di classe è qualificato con il nome del suo schema sorgente (`nome_sorgente.nome_classe`), un nome di attributo è qualificato anche con il nome della sua classe (`nome_sorgente.nome_classe.nome_attributo`). L'insieme dei nomi delle classi e degli attributi è denotato con $\mathbf{N}$; l'insieme dei termini su $\mathbf{N}$ è indicato con $\mathbb{I}$. Le relazioni semantiche tra termini sono definite sulla base delle relazioni definite in WordNet tra *synset*. Nel presente contesto si usano le seguenti relazioni tra *synset*: **Sinonimia**, **Ipernimia**, **Iponimia**, **Olonimia**, **Meronimia**, **Correlazione**[1]. Siccome iponimia e meronimia sono le relazioni inverse di ipernimia e olonimia, rispettivamente, l'insieme delle relazioni di interesse tra *synset* è il seguente: $\mathcal{W} =$

[1] La correlazione è la relazione che lega 2 *synset* che condividono uno stesso ipernimo, cioè lo stesso padre.

$\{\mathbf{S}_{inonimia}, \mathbf{I}_{pernimia}, \mathbf{O}_{lonimia}, \mathbf{C}_{orrelazione}\}$.
Dato l'insieme di *synset* $\mathcal{S}$ e l'insieme di relazioni $\mathcal{W}$, si introduce la funzione $\phi : \mathcal{S} \times \mathcal{W} \longrightarrow 2^{\mathcal{S}}$ che per ogni *synset* $s$ restituisce l'insieme dei *synset* ad esso associati tramite la relazione $r \in \mathcal{W}$:

$$\phi(s, r) = \{s' \mid s' \in \mathcal{S}, r \in \mathcal{W}, \langle s' r s \rangle\}.$$

Dato un insieme di *synset* $\mathcal{S}$ ed un insieme di termini $\mathbb{I}$, si definisce la funzione $\mathcal{H} : \mathcal{S} \longrightarrow 2^{\mathbb{I}}$ che associa ad un certo *synset* un insieme di termini sulla base della matrice lessicale:

$$\mathcal{H}(s) = \{t = (n, e) | n \in \mathbf{N}, \mathcal{M}[s][t.e.f] = t.e\}$$

Possiamo quindi ricavare le relazioni tra termini sulla base delle relazioni stabilite tra i *synset* che li contengono. Dato un insieme di termini $\mathbb{I}$, l'insieme delle relazioni tra termini $\mathcal{R}$, $\mathcal{R} \subseteq \mathbb{I} \times \mathcal{W} \times \mathbb{I}$, è definito come segue:

$$\mathcal{R} = \{\langle t_i r t_j \rangle | r \in \mathcal{W}, t_i, t_j \in \mathbb{I}, \exists s : t_i \in \mathcal{H}(s), t_j \in \phi(s, r), t_i \neq t_j\}$$

Le relazioni derivanti da WordNet vengono proposte come relazioni semantiche da inserire nel *Common Thesaurus* in base alla seguente corrispondenza:

**Sinonimia :** corrisponde ad una relazione SYN.
**Ipernimia :** corrisponde ad una relazione BT.
**Olonimia :** corrisponde ad una relazione RT.
**Correlazione :** corrisponde ad una relazione RT.

Sulla base di queste considerazioni è stato sviluppato ed implementato un algoritmo che acquisiti in input i termini degli schemi da integrare, restituisce le relazioni semantiche individuate:

$$\text{Input } \mathbb{I} = \{t_i | t_i.n \in \mathbf{N}\}$$
$$\text{Output } \mathcal{R} = \{\langle t_i \, r \, t_j \rangle, \quad r \in \{\text{SYN}, \text{BT}, \text{RT}\}\}$$

Gli aspetti implementativi dell'algoritmo sono riportati in [13]. Nel seguito faremo alcune considerazioni sull'utilizzo del tool. Partendo dagli schemi da integrare, il progettista deve fissare l'insieme $\mathbb{I}$. Cioè dato un nome $n$ deve scegliere i termini ad esso associati. Tale scelta è eseguita in due fasi:

1. **Scelta della forma base.** In tale scelta il progettista è assistito dal sistema che propone la forma base (word form) usando il processore morfologico di WordNet. Ad esempio, in Fig. 4, selezionando `address`, si ottiene la sua forma base. Se tale forma non è trovata, oppure se c'è ambiguità[2], oppure non è soddisfacente, il progettista può immetterla direttamente.
2. **Scelta del significato.** Il progettista può decidere di far corrispondere ad un nome zero, uno o più significati. La scelta di non far corrispondere ad un nome alcun termine può essere fatta perchè: (**a**) è un concetto troppo complesso per essere espresso da una sola parola: `seats_number`; (**b**) appartiene ai *tops*, cioè ai concetti più generici, e dunque si troverebbe ad essere in relazione con tutto: `relation`; (**c**) è una chiave surrogata, non aggiunge conoscenza: `dept_code` della tabella `Department`; (**d**) è usata come *foreign key*, dunque questa relazione è già stata espressa durante l'estrazione di relazioni dalla struttura degli schemi: `dept_code` della tabella `research_staff`. Il progettista sceglie uno o più significati tra quelli trovati in WordNet a partire dalla forma base scelta al punto 1. Pertanto, tutti i termini associati allo stesso nome, condividono la stessa forma base. Ad esempio in Fig. 4 per la forma base `address` ottengo tutti i 15 significati che WordNet le attribuisce.

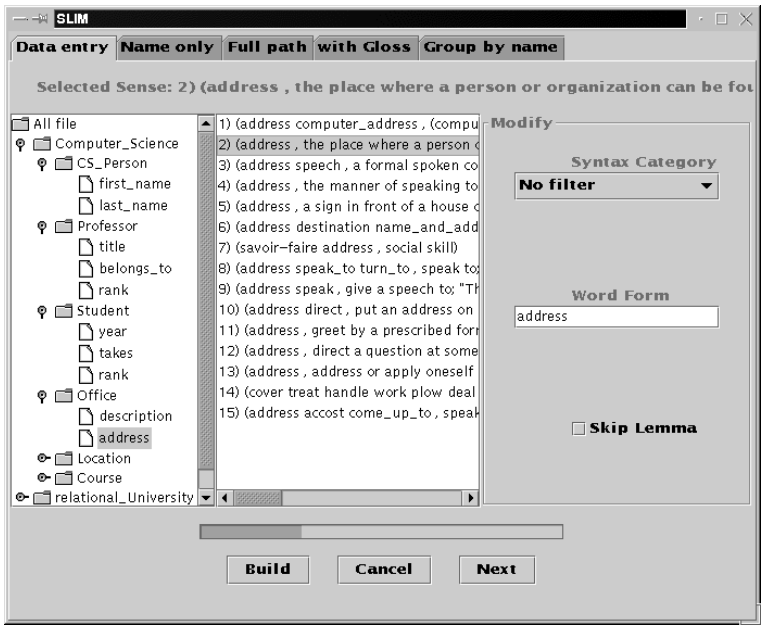


**Figura 4.** Significati per `address`

Selezionandoli tutti, cioè considerando 15 termini per l'attributo `address`, si otterrebbero risultati "*sbagliati*", cioè non calzanti con il contesto in esame, alcuni dei quali mostrati nel seguito:

⟨`Computer_Science.Office.address` NT `Tax_Position.Un_Student.name`⟩
⟨`Computer_Science.Office.description` NT `University.Research_Staff.relation`⟩
⟨`Computer_Scienze.Location.number` NT `University.Research_Staff.name`⟩
⟨`University.School_Member.faculty` RT `Computer_Science.Student.rank`⟩
⟨`University.Section` RT `Computer_Science.Professor.title`⟩

Per quanto alcune di tali relazioni possano sembrare strane, in contesti particolari esse sono vere.

Se invece il progettista sceglie solo il significato *"giusto"*, le relazioni estratte sono quelle indicate come inter-schema relationship in Fig. 5, che sono in numero minore e "giuste". Il problema, quindi, si sposta nel risolvere l'ambiguità sul significato, per poter fornire a WordNet, per ogni identificatore, una coppia (forma base, numero del significato) appropriata al contesto.

Per aiutare il progettista a scegliere il significato che ritiene giusto, si può applicare un filtro in base alla categoria sintattica (nomi, verbi, aggettivi, avverbi) (Syntax Category in Fig. 4).

Questo approccio semiautomatico riduce la complessità per il progettista: infatti, viene scomposto un solo problema "*difficile*", trovare le relazione tra tutti i termini, in tanti problemi "*facili*", scegliere da una lista il senso di ogni termine, in numero lineare rispetto alla quantità degli identificatori. In pratica questo è un problema 80/20 cioè l'80% dei termini si risolve nel 20% del tempo, solo il tempo di leggere le definizioni, mentre il restante 20% occupa l'80% del tempo, perchè bisogna scegliere tra significati molto simili. Per velocizzare l'80% ci si serve di una "cache" dei termini già decisi. Infatti l'ambito, che ci permette di eliminare le ambiguità, è diviso in 3 strati:
**Basi di dati:** se appare l'attributo `name` in qualunque tabella, è sempre nel senso di nome usato per identificare una cosa o persona, e non, per esempio, il verbo "dare il nome".

[2] Per esempio di `axes` vengono trovate 3 forme base: `ax` (1 senso), `axis` (5 sensi), `axe` (2 sensi).

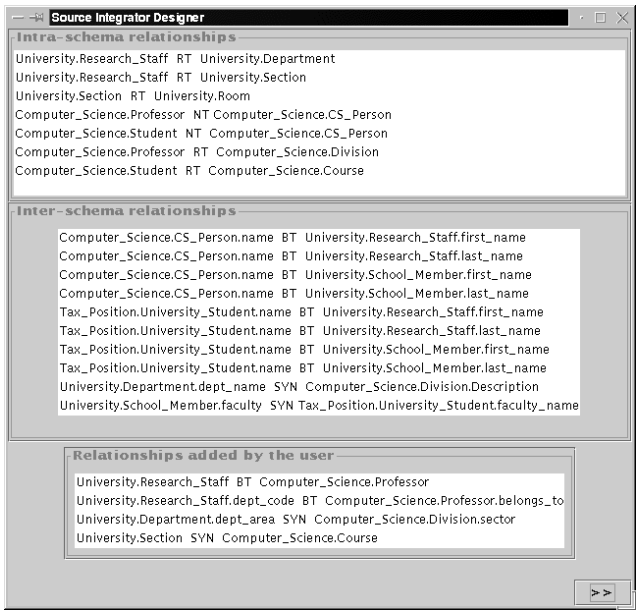


**Figura 5.** Relazioni intensionali ottenute dopo la fase 3.3

**Schema sorgente:** in un contesto universitario `course` è sempre usato nel senso di una serie di lezioni, e continua a mantenere questo significato in tutto lo schema.
**Classe:** sono i termini che vengono resi chiari solo dal nome della tabella o dagli altri campi; in questo caso la cache non ci può aiutare.

I termini in "cache" vengono visualizzati al progettista che sceglie se usarli o no.

### 3.3 Relazioni intensionali inter-schema aggiunte dal progettista

Il progettista può aggiungere al *Common Thesaurus* ottenuto nelle fasi precedenti delle nuove relazioni intensionali inter-schema (non individuate tramite WordNet perchè non "affini" in termini linguistici). Per aiutare il progettista in tale compito, vengono calcolate le *affinità* strutturali tra classi, tramite il modulo ARTEMIS. L'affinità tra ogni coppia di classi è espressa attraverso un coefficiente, chiamato *Global Affinity (GA)*, calcolato come combinazione lineare di due fattori: il primo valuta l'affinità strutturale delle classi, e il secondo dei nomi delle classi e degli attributi [2]. Tali coefficienti vengono calcolati riorganizzando il *Common Thesaurus* in una struttura simile alle Associative Networks [14], dove i nodi (ciascuno dei quali rappresenta genericamente un termine, sia esso il nome di una classe o il nome di un attributo) sono uniti attraverso le relazioni che sono presenti nel *Common Thesaurus*, assegnando un peso a ciascuno dei tre tipi di relazione intensionale. Ad esempio, analizzando i seguenti coefficienti *Global Affinity* calcolati da ARTEMIS:

$$GA(\texttt{University.Research\_Staff}, \texttt{Computer\_Science.Professor}) = 0.4$$
$$GA(\texttt{University.Section}, \texttt{Computer\_Science.Course}) = 0.66$$

si può decidere di inserire o meno una relazione (SYN o BT o RT) fra le classi:

$$\langle \texttt{University.Research\_Staff} \text{ BT } \texttt{Computer\_Science.Professor} \rangle$$
$$\langle \texttt{University.Section} \text{ SYN } \texttt{Computer\_Science.Course} \rangle$$

Il progettista può inoltre aggiungere esplicitamente altre relazioni, quali ad esempio:

⟨`University.Research_Staff.dept_code` BT `Computer_Science.Professor.belongs_to`⟩
⟨`University.Department.dept_area` SYN `Computer_Science.Division.Sector`⟩

Il *Common Thesaurus* ottenuto fino a questo punto viene visualizzato da SI-Designer (vedi Fig. 5).

### 3.4 Validazione delle relazioni tra attributi valore

Le relazioni tra attributi valore inserite nei passi precedenti nel *Common Thesaurus*, devono essere esaminate per verificare la compatibilità dei loro domini; tale fase è detta *validazione* e viene svolta dal modulo SIM attraverso ODB-Tools. Considerata una coppia di attributi posti in relazione, $a_i = \langle na_i, da_i \rangle$ e $a_j = \langle na_j, da_j \rangle$, la validazione viene attuata secondo i seguenti criteri:

- $\langle na_i$ SYN $na_j \rangle$: è *valida* se i domini $da_i$ e $da_j$ sono equivalenti o se uno è più specializzato;
- $\langle na_i$ BT $na_j \rangle$: è *valida* se $da_i$ contiene o è equivalente a $da_j$.

Come esempio si mostrano alcune delle relazioni tra attributi che hanno subito la validazione ('[1]' significa relazione validata mentre '[0]' indica il contrario):

⟨`Tax_Position.Un_Student.name` BT `Computer_Science.CS_Person.first_name`⟩ [1]
⟨`Tax_Position.Un_Student.name` BT `Computer_Science.CS_Person.last_name`⟩ [1]
⟨`University.Research_Staff.dept_code` BT `Computer_Science.Professor.belongs_to`⟩ [0]

Nell'esempio si nota come la relazione `dept_code` BT `belongs_to` non abbia superato la validazione poichè il dominio dell'attributo `dept_code` è un numero o una stringa (tipici domini di codici) mentre il dominio dell'attributo `belongs_to` è un attributo complesso.

### 3.5 Relazioni intensionali inferite

In quest'ultima fase si vogliono inferire nuove relazioni semantiche, partendo da quelle già introdotte nel *Common Thesaurus* ed utilizzando le tecniche di inferenza di OLCD. Siccome le relazioni semantiche di generalizzazione (BT) ed equivalenza (SYN) stabilite nel *Common Thesaurus* tra nomi di classi possono essere in conflitto con le descrizioni strutturali delle classi correlate, si produce uno *schema virtuale*, sempre espresso in $\mathrm{ODL}_{I^3}$, che contiene una descrizione degli schemi sorgenti "ristrutturata" sulla base delle relazioni semantiche stabilite nel *Common Thesaurus*:

- per le relazioni $\langle C_1$ `SYN` $C_2 \rangle$ è necessario "uniformare" le descrizione delle classi $C_1$ e $C_2$, in modo che la struttura sia la stessa;
- per le relazioni $\langle C_1$ `BT` $C_2 \rangle$ è necessario "uniformare" la descrizione della classe $C_1$ a quella della classe $C_2$, in modo che la struttura di $C_1$ sia una specializzazione di quella di $C_2$. In sostanza, questo significa aggiungere gli attributi della classe $C_2$ alla struttura della classe $C_1$;
- per le relazioni $\langle C_1$ `RT` $C_2 \rangle$ è necessario aggiungere alla descrizione della classe $C_1$ un nuovo attributo di aggregazione che si riferisce alla classe $C_2$.

Lo schema virtuale costituisce solo uno strumento tecnico per l'inferenza di nuove relazioni e rimane trasparente all'utente. Nell'esempio, alcune relazioni inferite sono:

⟨`University.Research_Staff` NT `Computer_Science.CS_Person`⟩
⟨`University.School_Member` NT `Computer_Science.CS_Person`⟩
⟨`University.Section` RT `Computer_Science.Professor`⟩

## 4 Generazione delle classi globali

Una volta costruito il *Common Thesaurus*, SI-Designer può generare le classi globali. Tale operazione viene attuata dal tool in questo modo: (1) Calcolo delle affinità, (2) Generazione dei cluster e (3) Generazione degli attributi globali e delle *mapping-table*. Nella prima fase, SI-Designer funge da interfaccia tra il modulo ARTEMIS e il progettista che può interagire più volte con ARTEMIS, fino a quando non è soddisfatto dell'insieme di cluster ottenuto. Nella seconda fase invece, il tool costruisce, per ogni cluster, una classe globale a cui è associato un insieme di attributi globali e la relativa mapping-table, mettendo a disposizione del progettista una interfaccia per revisionare gli insiemi di attributi globali e le mapping-table proposti dal tool. Con la medesima interfaccia, il progettista può infine assegnare un nome ad ogni classe globale.

### 4.1 Generazione dei cluster

Per la generalizzazione dei cluster, ARTEMIS utilizza tecniche di clustering attraverso le quali le classi sono automaticamente classificate una struttura ad albero dove: le foglie rappresentano tutte le classi locali (foglie contigue sono classi caratterizzate da alta affinità, mentre foglie tra loro molto lontane rappresenteranno invece classi a bassa affinità) e ogni nodo rappresenta un livello di clusterizzazione ed ha associato il coefficiente di affinità tra i due sottoalberi (cluster) che unisce. Con SI-Designer, si può immettere ad ogni iterazione un valore di soglia: ogni cluster sarà costituito da tutte le classi appartenenti ad un sottoalbero che al nodo radice ha un coefficiente maggiore del valore di soglia. La Fig.6 mostra l'albero delle classi locali e i cluster individuati da ARTEMIS sulla base dell'esempio presentato, ponendo un coefficiente di soglia $\sigma$=0.5.

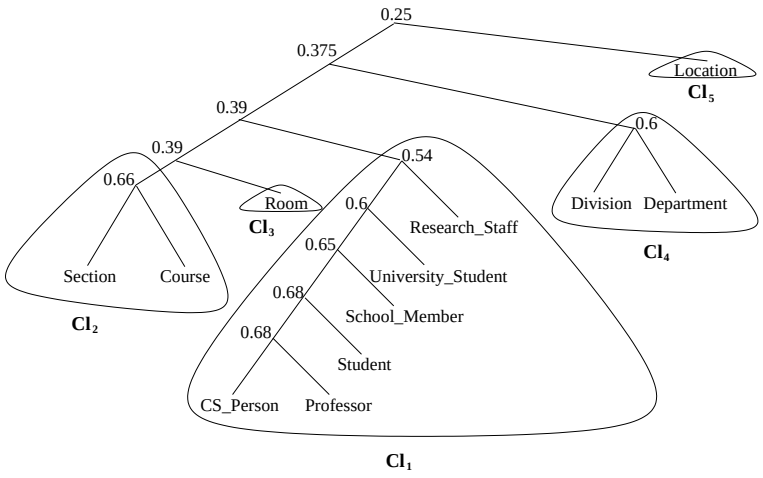


**Figura 6.** Albero delle affinità: clusters con $\sigma = 0.5$.

### 4.2 Generazione degli attributi globali e delle mapping-table

Per ogni cluster, SI-Designer crea un insieme di attributi globali e, per ognuno di essi, determina la corrispondenza con gli *attributi locali* (quelli delle classi appartenenti al cluster cui corrisponde

la classe globale). In alcuni casi, la corrispondenza è unica mentre in altri ci sono diversi tipi di corrispondenza che il tool individua ma di cui non può risolvere l'ambiguità: in questo caso il tool chiede al progettista di scegliere quella giusta.

Il tool costruisce l'insieme degli attributi globali da associare ad un cluster in due fasi: (1) Unione degli attributi di tutte le classi affini appartenenti al cluster e (2) Fusione degli attributi "simili". L'unione degli attributi delle classi affini consiste in una mera raccolta di tutti gli attributi di ogni classe del cluster in un insieme. Tale insieme di attributi è un possibile insieme di attributi globali, in generale ridondanti. Nella seconda fase SI-Designer tenta di eliminare queste ridondanze considerando le relazioni terminologiche del *Common Thesaurus*. Il procedimento di fusione è automatico per gli attributi legati da relazioni validate mentre lo è solo in certi casi se gli attributi sono legati da relazioni non validate. In particolare SI-Designer opera in questo modo:

- **Attributi in relazioni validate.** Per questi attributi la fusione è sempre automatica:
  - Agli attributi legati da relazioni `SYN` corrisponderà un attributo globale il cui dominio è lo stesso di quelli locali ed il nome può essere scelto dal progettista esplicitamente o tra le proposte di SI-Designer. Per esempio: agli attributi `description` e `dept_name`, legati dalla relazione `description SYN dept_name`, viene associato l'attributo globale `description`.
  - Gli attributi legati da relazioni `BT` vengono sostituiti con un attributo globale che ha lo stesso nome e lo stesso dominio dell'attributo generalizzazione. Per esempio gli attributi `name`, `first_name` e `last_name`, con `name BT first_name` e `name BT last_name`, saranno rappresentati dall'attributo globale `name`.
- **Attributi in relazioni non validate.** SI-Designer è in grado di individuare un attributo globale in modo automatico solo per un numero limitato di casi, lasciando al progettista il compito di aggiungere altri attributi globali per completare l'integrazione. In particolare, l'individuazione automatica di un attributo globale, in presenza di relazioni non validate, è possibile se gli attributi nelle relazioni soddisfano i seguenti requisiti: (1) sono legati da relazioni `SYN` o `BT`, (2) le classi in relazione appartengono ad uno stesso cluster e (3) rappresentano gerarchie di aggregazione (sono attributi complessi o foreign key). Come esempio, gli attributi della relazione non validata `dept_code BT belongs_to` soddisfano tutte le condizioni elencate prima e dunque è possibile aggiungere un attributo globale `works` che corrisponde ai due attributi locali `dept_code` e `belongs_to` (vedere mapping-table di Fig.7).

L'insieme di attributi globali così individuato può essere ampliato dal progettista per rappresentare tutte le informazioni delle sorgenti locali. Contemporaneamente alla creazione degli attributi globali, SI-Designer costruisce una *mapping-table* (vedere Fig.7). Essa é una tabella $MT[CL][AG]$ dove $CL$ é l'insieme delle classi locali che appartengono al cluster cui la mapping-table si riferisce e $AG$ é l'insieme degli attributi globali creato da SI-Designer. Indicando con $C$ il nome di una classe locale, con $A$ il nome di un attributo globale e con $AL$ il nome di un attributo locale, ogni elemento $MT[C][A]$ della tabella può assumere i seguenti valori:

- $AL$, con $AL \in C$. Questo valore viene inserito quando:
  - l'attributo globale $A$ deve rappresentare l'informazione contenuta nel solo attributo locale $AL$. Per esempio, l'attributo globale `name` corrisponde a `name` di `CS_Person`;
  - ci sono relazioni di specializzazione che legano tra loro attributi appartenenti a classi diverse. Per esempio, l'attributo globale `faculty` di `University_Person` corrisponde a `faculty` di `School_Member` e a `faculty_name` di `University_Person`;

| **University_Person** | **name** | **rank** | **works** | **faculty** | **email** | **...** |
|---|---|---|---|---|---|---|
| Research_Staff | first_name *and* last_name | 'Professor' | dept_code | null | email | ... |
| School_Member | first_name *and* last_name | 'Student' | null | faculty | null | ... |
| CS_Person | name | null | null | 'Computer_Science' | null | ... |
| Professor | name | rank | belongs_to | 'Computer_Science' | null | ... |
| Student | name | rank | null | 'Computer_Science' | *tag* rank<br>'course':home_mail<br>'phd':phd_mail | ... |
| University_Student | name | 'Student' | null | faculty_name | null | ... |

| **Workplace** | **name** | **area** | **employee_nr** | **budget** | **...** |
|---|---|---|---|---|---|
| Department | dept_name | dept_area | null | budget | ... |
| Division | description | sector | employee_nr | fund | ... |

**Figura 7.** Mapping-table per University_Person e Workplace.

- $AL_1$ ***and*** $AL_2$ ***and ... and*** $AL_n$, con $AL_i \in C, i = 1, ..., n$. Usato quando il valore dell'attributo $A$ é il concatenamento dei valori di piú attributi appartenenti alla medesima classe locale $C$. Per esempio, l'attributo globale name di University_Person corrisponde al concatenamento degli attributi first_name e last_name di School_Member;
- ***case of*** $AL$ $\quad cost_1$**:** $AL_1 \quad cost_2$**:** $AL_2 \quad$ **...** $\quad cost_n$**:** $AL_n$
  dove $AL, AL_i \in C, i = 1, ..., n$ e $cost_i, i = 1, ..., n$ sono delle costanti. Questa situazione avviene quando l'attributo globale $A$ può assumere il valore di uno tra un insieme di attributi locali $\{AL_i\}$ appartenenti alla medesima classe e la scelta avviene attraverso un terzo attributo locale *selettore* $AL$, appartenente sempre alla stessa classe locale. Per esempio, l'attributo globale email deve assumere il valore di home_email o phd_email a seconda del valore assunto dall'attributo locale rank di Student: se rank = 'course' allora email assume il valore di home_email altrimenti se rank = 'phd' allora email assume il valore di phd_email;
- *costante*. È il caso in cui l'attributo globale $A$ non corrisponde ad alcun attributo della classe locale $C$. Il valore di $A$ viene attribuito dal progettista in base al significato dato ad $A$. Per esempio, l'attributo globale rank di University_Person assume il valore 'Professor' se occorre accedere ad Research_Staff e il valore 'Student' se occorre accedere ad University_Student.
- *null*. Questo é il caso in cui l'attributo globale $A$, durante un accesso alla classe locale $C$, non assume alcun valore. Per esempio, l'attributo globale $A$ faculty non assume alcun valore nella classe locale Research_Staff: ciò viene indicato nella mapping-table con la costante *null*.

SI-Designer crea una *mapping-table* per ogni classe globale e mette a disposizione una interfaccia sia per avere una visione completa di tutte le classi globali (nomi ed attributi), incluse le relative mapping-tables sia per l'inserimento dei nomi delle classi globali e la modifica delle *mapping table*. Il tool conclude l'integrazione con la creazione della descrizione $ODL_{I^3}$ dello schema globale [2].

## 5 Conclusioni

In questo lavoro è stato presentato il tool SI-Designer per assistere il progettista nell'integrazione di sorgenti eterogenee di informazioni. Questo tool opera nell'ambito del progetto MOMIS, in corso di sviluppo presso l'Università di Modena e Reggio Emilia. Nel presente lavoro, particolare attenzione è stata rivolta alla descrizione del componente SLIM che è stato sviluppato per interagire con il sistema WordNet. Un'interazione diretta con WordNet senza il supporto di SLIM è improponibile come strumento di effettivo ausilio al progettista nella fase di integrazione di schemi a causa del numero eccessivo delle relazioni proposte da WordNet per ogni termine e dal fatto che solo una minima parte di queste relazioni è consistente con le sorgenti oggetto dell'analisi. L'interazione con il progettista realizzata attraverso SLIM è allo stato attuale di buon livello e costituisce quindi un valido strumento per l'attività di integrazione di sorgenti.

## Riferimenti bibliografici


1. S. Bergamaschi, S. Castano, and M. Vincini. Semantic integration of semistructured and structured data sources. *SIGMOD Records*, 28(1), March 1999.
2. S. Bergamaschi, S. Castano, S. De Capitani di Vimercati, S. Montanari, and M. Vincini. Exploiting schema knowledge for the integration of heterogeneous sources. In *Sesto Convegno Nazionale su Sistemi Evoluti per Basi di Dati - SEBD98, Ancona*, pages 103–122, 1998.
3. D. Beneventano, S. Bergamaschi, S. Lodi, and C. Sartori. Consistency checking in complex object database schemata with integrity constraints. *IEEE Transactions on Knowledge and Data Engineering*, 10:576–598, July/August 1998.
4. S. Bergamaschi and B. Nebel. Acquisition and validation of complex object database schemata supporting multiple inheritance. *Journal of Applied Intelligence*, 4:185–203, 1994.
5. S. Castano and V. De Antonellis. Deriving global conceptual views from multiple information sources. In *preProc. of ER'97 Preconference Symposium on Conceptual Modeling, Historical Perspectives and Future Directions*, 1997.
6. J. Gilarranz, J. Gonzalo, and F. Verdejo. Using the eurowordnet multilingual semantic database. In *Proc. of AAAI-96 Spring Symposium Cross-Language Text and Speech Retrieval*, 1996.
7. A.G. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.
8. R. Hull and R. King et al. Arpa $i^3$ reference architecture, 1995. Available at http://www.isse.gmu.edu/I3_Arch/index.html.
9. D. Beneventano, S. Bergamaschi, C. Sartori, and M. Vincini. ODB-QOPTIMIZER: A tool for semantic query optimization in oodb. In *Int. Conference on Data Engineering - ICDE97*, 1997. http://sparc20.dsi.unimo.it.
10. D. Beneventano, S. Bergamaschi, and C. Sartori. Semantic query optimization by subsumption in OODB. In H. Christiansen, H. L. Larsen, and T. Andreasen, editors, *Flexible Query Answering Systems*, volume 62 of *Datalogiske Skrifter - ISSN 0109-9799*, Roskilde, Denmark, 1996.
11. ODB-Tools staff. Odb-tools Project. Available at http://sparc20.dsi.unimo.it.
12. Bates M. Subject access in online catalogs: A design model. *Journal of the American Society for Information Science*, 11:357–376, 1986.
13. MOMIS staff. Momis (mediator environment for multiple information sources) project part of the interdata project. Available at http://sparc20.dsi.unimo.it/Momis.
14. N.V. Findler, editor. *Associative Networks.* Academic Press, 1979.