



**UNIVERSITA' DEGLI STUDI DI CAGLIARI**

**FACOLTÀ DI SCIENZE**

Corso di Laurea in Informatica

**Conversione della rappresentazione RDF di FRED  
*Machine Reader* ad *Abstract Meaning Representation***

**Docenti di riferimento:**

Prof. Diego Reforgiato Recupero  
Prof. Gianni Fenu

**Candidato:**

Maurizio Romano  
(matr. 65031)

ANNO ACCADEMICO 2015-2016



**Titolo:** Conversione della rappresentazione RDF di FRED *Machine Reader* ad *Abstract Meaning Representation*

**Autore:** Maurizio Romano

**Relatore:** Prof. Diego Reforgiato Recupero

**Correlatore:** Prof. Gianni Fenu

**Tipo tesi:** Progetto+implementazione di un prototipo

**Abstract:** in un mondo come quello odierno l'intelligenza artificiale è uno degli argomenti più importanti, è dunque fondamentale sviluppare sempre più sofisticati strumenti per consentire alle stesse la comprensione del linguaggio naturale.

Nel parlato comune si è soliti esprimere concetti che trascendono il mero contenuto sintattico di una frase, ovvero espressi tramite l'aspetto semantico.

Per poter rendere tali aspetti è necessario dotare le macchine della capacità di analizzare le frasi nel loro insieme piuttosto che parola per parola.

Questo è esattamente ciò di cui si occupa questa Tesi: partendo dall'analisi sintattica espressa in formato RDF fornita dal *Machine Reading* del tool FRED tramite un vasto insieme di regole è possibile ottenere nella maggior parte dei casi dell'AMR.

Differentemente dall'RDF, l'*Abstract Meaning Representation* fornisce una rappresentazione di tipo semantico tramite struttura ad eventi.

La potenzialità di questa codifica consente dunque di rendere nel medesimo modo frasi che, se pur scritte in modo differente e con termini diversi, intendono esattamente la stessa cosa.

Analizzato l'RDF prodotto da FRED e l'AMR sono state prodotte delle regole di conversione e successivamente è stato dunque sviluppato in *python* un prototipo che consentisse tramite un'interfaccia web di ottenere AMR partendo da una frase utilizzando FRED in modo trasparente per l'utente.

È importante sottolineare come l'AMR sia un formato a cui attinge tutta la comunità *Natural Language Processing*, mentre l'RDF è utilizzato dal *Semantic Web*.

L'obiettivo finale è dunque creare un ponte fra le due comunità cosicché tutte le ricerche ed i lavori prodotti dalle stesse siano fruibili per entrambe.



## INDICE

<b>1</b>	<b>INTRODUZIONE.....</b>	<b>7</b>
1.1	ARGOMENTO DELLA TESI .....	7
1.2	CONTESTO DELLA TESI .....	7
1.3	SCOPO DELLA TESI.....	7
<b>2</b>	<b>STATO DELL'ARTE.....</b>	<b>9</b>
2.1	IL <i>SEMANTIC WEB</i> .....	9
2.1.1	<i>L'RDF</i> .....	9
2.2	IL <i>NATURAL LANGUAGE PROCESSING</i> .....	12
2.2.1	<i>L'AMR</i> .....	12
<b>3</b>	<b>FRED.....</b>	<b>15</b>
3.1	FRED ALL'OPERA.....	15
<b>4</b>	<b>IL PROGETTO APPLICATIVO.....</b>	<b>17</b>
4.1	L'IDEAZIONE .....	17
4.2	LA PROGETTAZIONE .....	17
4.2.1	<i>L'acquisizione dei dati</i> .....	17
4.2.2	<i>Conversione: fattibilità ed ideazione delle regole AMR</i> .....	19
4.2.3	<i>Interfaccia grafica</i> .....	20
4.3	LO SVILUPPO .....	21
4.3.1	<i>Il fulcro dell'Applicazione: le regole di conversione</i> .....	21
4.3.2	<i>Limiti di FRED: regole AMR non implementabili</i> .....	25
4.4	IL PROTOTIPO: "GEORGE" .....	27
4.5	I TEST .....	28
<b>5</b>	<b>CONCLUSIONI.....</b>	<b>29</b>
5.1	SVILUPPI FUTURI.....	29
	<b>RINGRAZIAMENTI.....</b>	<b>31</b>
	<b>APPENDICE A.....</b>	<b>33</b>
	<b>BIBLIOGRAFIA .....</b>	<b>34</b>



# 1 INTRODUZIONE

## 1.1 Argomento della Tesi

Analizzato l'RDF prodotto da FRED e l'AMR sono state prodotte delle regole di conversione e successivamente è stato dunque sviluppato in *python* un prototipo che consentisse tramite un'interfaccia web di ottenere AMR partendo da una frase utilizzando il *Machine Reading* di FRED in modo trasparente per l'utente.

## 1.2 Contesto della Tesi

In un mondo come quello odierno l'intelligenza artificiale è uno degli argomenti più importanti, è dunque fondamentale sviluppare sempre più sofisticati strumenti per consentire alle stesse la comprensione del linguaggio naturale.

Nel parlato comune si è soliti esprimere concetti che trascendono il mero contenuto sintattico di una frase, ovvero espressi tramite l'aspetto semantico.

Per poter rendere tali aspetti è necessario dotare le macchine della capacità di analizzare le frasi nel loro insieme piuttosto che parola per parola.

La comunità NLP e quella del *Semantic Web* nonostante svolgano un lavoro affine possiedono standard e metodologie di lavoro sufficientemente discordanti, ciò rende impossibile l'unione degli studi prodotti da entrambe per un fine comune.

## 1.3 Scopo della Tesi

Lo scopo della Tesi è dunque dotare le intelligenze artificiali di una comprensione di tipo semantico convertendo tramite opportune regole (cap. 4.3.1) l'analisi sintattica espressa in formato RDF fornita dal *tool* FRED in AMR.

Differentemente dall'RDF, l'*Abstract Meaning Representation* fornisce una rappresentazione di tipo semantico tramite struttura ad eventi.

La potenzialità di questa codifica consente dunque di rendere nel medesimo modo frasi che, se pur scritte in modo differente e con termini diversi, intendono esattamente la stessa cosa.

In questo modo si andrebbe a produrre un ponte in grado di riflettere il lavoro espresso in ambito del *Semantic Web* in quello della comunità NLP accelerando di non poco lo sviluppo di entrambi gli ambiti.





## 2 STATO DELL'ARTE

Prima ancora d'iniziare a pensare all'applicativo è necessario valutare con attenzione l'attuale mercato, nel particolare verificare l'esistenza di progetti simili il cui utilizzo è ormai consolidato o meno fra l'utenza.

Questo consente di comprendere la bontà di un'idea, la sua attuabilità nonché la sua usabilità concreta affinché si possa evitare un'inefficiente spreco di tempi e risorse.

Benché esistano già progetti che consentono di ricavare RDF partendo da AMR, non ne esistono che facciano l'inverso.

Lo studio trattato in questa Tesi è dunque non confrontabile con nessun altro tipo di lavoro in quanto unico nel suo genere.

La “AMR Bank” è infatti costruita a mano da persone specializzate, non esiste nient'altro che automatizzi il lavoro di conversione da una frase ad AMR.

### 2.1 Il *Semantic Web*

Con il termine *Semantic Web*, termine coniato dal suo ideatore, Tim Berners-Lee, si intende la trasformazione del World Wide Web in un ambiente dove i documenti pubblicati (pagine HTML, file, immagini, e così via) sono associati ad informazioni e dati (metadati) che ne specificano il contesto semantico in un formato adatto all'interrogazione e l'interpretazione (es. tramite motori di ricerca) e, più in generale, all'elaborazione automatica.

Con l'interpretazione del contenuto dei documenti che il *Semantic Web* impone, saranno possibili ricerche molto più evolute delle attuali, basate sulla presenza nel documento di parole chiave, e altre operazioni specialistiche come la costruzione di reti di relazioni e connessioni tra documenti secondo logiche più elaborate del semplice collegamento ipertestuale.

L'RDF (cap. 2.1.1) è quello che meglio si presta alle esigenze del *Semantic Web*.

#### 2.1.1 L'RDF

Il *Resource Description Framework* (RDF) è lo strumento base proposto da W3C per la codifica, lo scambio e il riutilizzo di metadati strutturati e consente l'interoperabilità semantica tra applicazioni che condividono le informazioni sul Web. È costituito da due componenti:

- RDF Model and Syntax: espone la struttura del modello RDF, e descrive una possibile sintassi.
- RDF Schema: espone la sintassi per definire schemi e vocabolari per i metadati.

L'RDF Data Model si basa su tre principi chiave:

1. Qualunque cosa può essere identificata da un Universal Resource Identifier (URI).
2. The least power: utilizzare il linguaggio meno espressivo per definire qualunque cosa.
3. Qualunque cosa può dire qualunque cosa su qualunque cosa.

Qualunque cosa descritta da RDF è detta risorsa. Principalmente una risorsa è reperibile sul web, ma RDF può descrivere anche risorse che non si trovano direttamente sul web. Ogni risorsa è identificata da un URI, Universal Resource Identifier.

Il modello di dati RDF è formato da risorse, proprietà e valori. Le proprietà sono delle relazioni che legano tra loro risorse e valori, e sono anch'esse identificate da URI. Un valore, invece, è un tipo di dato primitivo, che può essere una stringa contenente l'URI di una risorsa.

L'unità base per rappresentare un'informazione in RDF è lo statement. Uno statement è una tripla del tipo Soggetto – Predicato – Oggetto, dove il soggetto è una risorsa, il predicato è una proprietà e l'oggetto è un valore (e quindi anche un URI che punta ad un'altra risorsa).

Il data model RDF permette di definire un modello semplice per descrivere le relazioni tra le risorse, in termini di proprietà identificate da un nome e relativi valori. Tuttavia, RDF data model non fornisce nessun meccanismo per dichiarare queste proprietà, né per definire le relazioni tra queste proprietà ed altre risorse. Tale compito è definito da RDF Schema.

Un modello RDF è rappresentabile da un grafo orientato sui cui nodi ci sono risorse o tipi primitivi e i cui archi rappresentano le proprietà. Un grafo RDF è rappresentato fisicamente mediante una serializzazione.

Le principali serializzazioni adottabili per un grafo RDF sono:

- RDF/XML: documento RDF è serializzato in un file XML.
- N-Triples: serializzazione del grafo come un insieme di triple soggetto - predicato - oggetto.
- Notation3: serializzazione del grafo descrivendo, una per volta, una risorsa e tutte le sue proprietà.

In particolare la serializzazione in XML può avvenire secondo due metodi, quello classico e quello abbreviato, più leggibile per l'uomo.

Si supponga di voler serializzare la frase "Mario\_Rossi" "è\_autore\_di" "Rosso\_di\_sera\_bel\_tempo\_si\_spera", il risultato in RDF/XML sarà:

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:au="http://description.org/schema/">
  <rdf:Description about="http://www.book.it/Rosso_di_sera_bel_tempo_si_spera/">
    <au:author>Mario_Rossi</au:author>
  </rdf:Description>
</rdf:RDF>
```

In RDF si possono rappresentare le risorse come istanze di classi e definire sottoclassi e tipi.

#### 2.1.1.1 Classi RDF

Ogni risorsa descritta in RDF è istanza della classe `rdfs:Resource`.

Le sottoclassi di `rdfs:Resource` sono:

`rdfs:Literal` Rappresenta un letterale, una stringa di testo.

`rdfs:Property` Rappresenta le proprietà.

`rdf:Class` Una classe dei linguaggi object-oriented.

#### 2.1.1.2 Proprietà RDF

`rdf:type` Indica che una risorsa è del tipo della classe che viene specificata.

`rdfs:subClassOf` Indica la relazione classe/sottoclasse fra due classi.

L'ereditarietà può essere multipla.

`rdfs:subPropertyOf` Indica che una proprietà è specializzazione di un'altra.

`rdfs:seeAlso` Specifica che la risorsa è anche descritta in altre parti.

`rdfs:isDefinedBy` Indica la risorsa "soggetto dell'asserzione" ovvero chi ha fatto l'asserzione.

#### 2.1.1.3 Vincoli RDF

`rdfs:range` (codominio) È utilizzato come proprietà di una risorsa; indica le classi che faranno parte di una asserzione con la proprietà.

`rdfs:domain` (dominio) Indica la classe a cui può essere applicata la proprietà.



Una rappresentazione testuale per la medesima frase è così strutturata:

(w / want-01

:ARG0 (b / boy)

:ARG1 (b2 / believe-01

:ARG0 (g / girl)

:ARG1 b))

Si noti come, riferendosi al medesimo ragazzo, la variabile abbia il medesimo identificatore “b”.

Differentemente dall’RDF questa codifica contiene veramente tante proprietà.

Esse verranno affrontate nei capitoli successivi trattando la loro possibile produzione (o meno) partendo dall’RDF.



### 3 FRED

FRED appartiene alla categoria dei *Machine Reader* per il *Semantic Web*, è in grado di processare testo espresso in linguaggio naturale in 48 lingue differenti e di trasformarlo in *Linked Data*.

È implementato in Python ed è disponibile come un servizio REST e come libreria Python (fredlib).

Il background teorico che risiede nel core di FRED è il seguente:

- Combinatory Categorical Grammar [C&C];
- Discourse Representation Theory [DRT, Boxer];
- Frame Semantics [Fillmore 1976];
- Ontology Design Patterns [Ontology Handbook].

FRED sfrutta component del NLP per effettuare diverse operazioni:

- Named Entity Resolution [Stanbol, TagMe];
- Coreference Resolution [CoreNLP];
- Word Sense Disambiguation [Boxer, IMS].

Tutti i grafi prodotti da FRED sono annotazioni testuali e rappresentano la segmentazione del testo espressa per mezzo di EARMARK e NIF.

#### 3.1 FRED all'opera

Come detto in precedenza, FRED si occupa di produrre RDF da frasi espresse in linguaggio naturale.

*Valentina gave Aldo a book by Charlie Mingus*

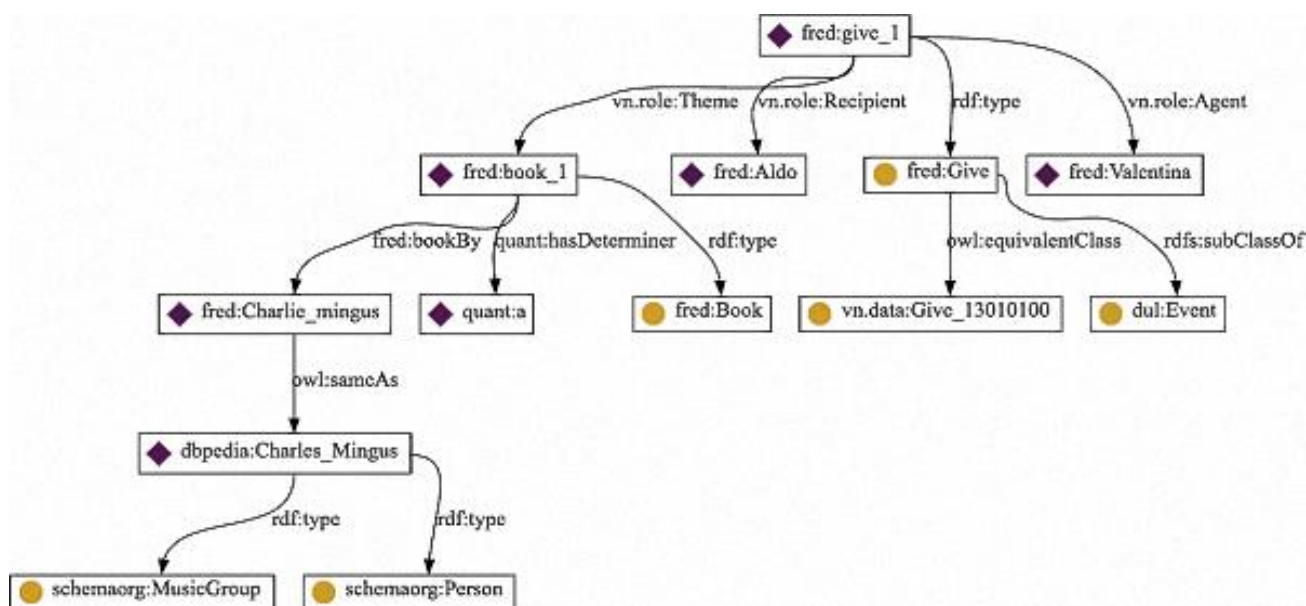


Figura 3.1

Oltre alla rappresentazione tramite grafo, è altresì disponibile l'output in formato RDF.

Grazie alla possibilità di poter attingere ad ambo le modalità è possibile sfruttare la componente in versione grafica per velocizzare l'individuazione dei parametri necessari all'implementazione di una regola di conversione in AMR e, successivamente, usare l'RDF puro per poterlo adoperare con *rdflib* al fine di lanciarvi sopra query SPARQL:

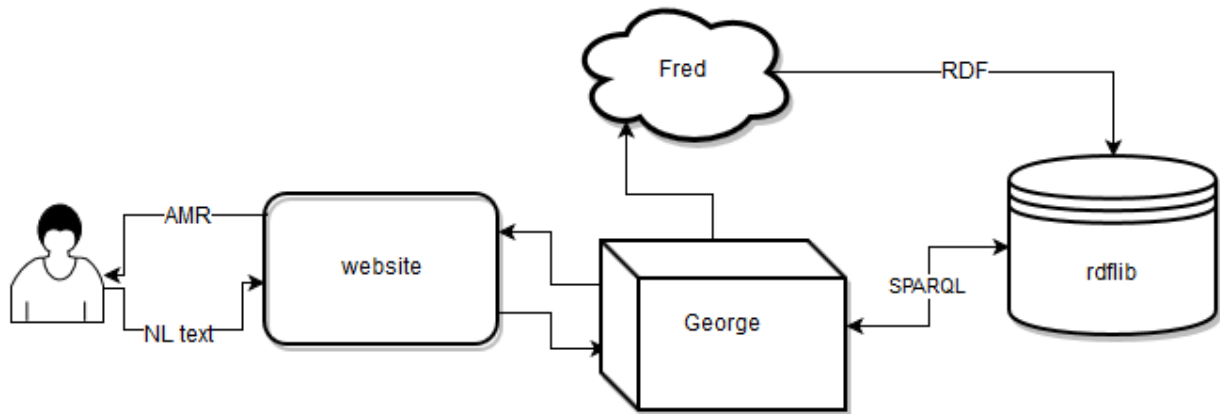


Figura 3.2

Come si può vedere (fig. 3.2) FRED ricopre un ruolo centrale nel Progetto in quanto fornisce tutti i dati che vanno a costituire la base sulla quale applicare tutti gli opportuni accorgimenti atti alla conversione.



## 4 IL PROGETTO APPLICATIVO

Svolgere un lavoro accurato necessita di una suddivisione in fasi: prima di scrivere del codice bisogna valutare le disposizioni dell'elaborato, la fattibilità dello stesso ecc.

Successivamente si prosegue con lo sviluppare ciò che si è appena progettato creando così un prototipo che di volta in volta verrà modificato fino ad arrivare alla versione ottimale che possa finalmente essere testata.

### 4.1 L'Ideazione

Il Progetto Applicativo è stato idealizzato principalmente dal Prof. Diego Recupero Reforgiato.

Essendo fra i produttori di FRED, nonché ricercatore nell'ambito del *Semantic Web*, ha pensato di rendere automatizzabile la produzione di AMR vista la grande potenzialità dello stesso nell'esprimere concetti di natura semantica.

L'Applicativo prodotto consente di far fronte a questa necessità andando dunque a sfruttare il lavoro del Prof. D. R. Reforgiato come intermediario dal testo semplice col fine di produrre AMR.

### 4.2 La Progettazione

L'idea di base consiste in un programma in python che sfrutti, con l'ausilio di rdflib, l'RDF prodotto da FRED applicandovi sopra delle regole ideate ad hoc per la conversione di tale input in AMR mostrando il tutto in un'apposita pagina web.

Questo suggerisce che il Progetto può essere suddiviso in tre parti distinte:

- 1) Acquisizione dei dati;
- 2) Conversione: fattibilità ed ideazione delle regole AMR;
- 3) Interfaccia grafica.

#### 4.2.1 L'acquisizione dei dati

Per prima cosa è necessario acquisire la frase da convertire e ricavarne l'output di FRED.

In questo viene in aiuto *curl* che, con i giusti parametri è in grado di prelevare dal sito web in cui risiede FRED l'opportuno RDF da lui prodotto sulla base della frase sottoposta dall'utente all'applicativo (fig. 4.1).

```

try:
    disableInput = False
    if(not disableInput):
        frase = input("Inserisci la frase:")
        #ottengo rdf da fred tramite curl
        curlString = "curl -G -X GET -H \"Accept: application/rdf+xml\" --data-urlencode text=\""+
            +frase
            +"\" http://etna.istc.cnr.it/stlab-tools/fred/demo"
        toParse = check_output(curlString, shell=True).decode()
        toParse = toParse[:-1]

        #salvo il tutto in un file temporaneo
        out_file = open("TEMP.rdf", "w")
        out_file.write(toParse)
        out_file.close()
except:
    print("\nATTENZIONE: la connessione internet sembra non funzionare!")
    sys.exit(0)

```

Figura 4.1

Ottenuta la frase bisogna dunque ricavare tutte le informazioni utili alle regole di conversione in AMR tramite opportune query SPARQL con l'appoggio di rdflib (figg. da 4.2 a 4.4)

```

#query SPARQL principale
gres = g.query(
    """SELECT DISTINCT ?mainverb ?j_1 ?j_2 ?j_3 ?that ?question ?j_4 ?j_5
    WHERE {
        ?mainverb a ?verb .
        ?verb rdfs:subClassOf <http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#Event> .
        OPTIONAL{?mainverb j.0:Theme ?j_1} .
        OPTIONAL{?mainverb j.0:Agent ?j_2} .
        OPTIONAL{?mainverb j.0:Experiencer ?j_3} .
        OPTIONAL{?mainverb j.4:that ?that} .
        OPTIONAL{?mainverb j.2:associatedWith ?question} .
        OPTIONAL{?mainverb j.0:Patient ?j_4} .
        OPTIONAL{?mainverb j.0:Source ?j_5}
    }""")

#guardo se ci sono delle coref
gres2 = g.query(
    """SELECT DISTINCT ?x ?y
    WHERE {
        ?x j.1:other_coref ?y
    }""")

#guardo se ci sono delle modality (rule 3)
gres3 = g.query(
    """SELECT DISTINCT ?x ?y
    WHERE {
        ?x j.2:hasModality ?y
    }""")

```

Figura 4.2

```

#guardo se ci sono dei TruthValue (rule 4)
gres4 = g.query(
    """SELECT DISTINCT ?x ?y
    WHERE {
        ?x j.2:hasTruthValue ?y
    }""")

#guardo il focus (rule 5)
gres5 = g.query(
    """SELECT DISTINCT ?m ?arg ?th
    WHERE {
        ?arg <http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#hasQuality> ?m .
        OPTIONAL{?arg <http://www.ontologydesignpatterns.org/ont/fred/quantifiers.owl#hasDeterminer> ?th}
    }""")

gres6 = g.query(
    """SELECT DISTINCT ?mainvarb ?situationChild
    WHERE {
        ?mainvarb a ?verb .
        ?verb rdfs:subClassOf <http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#Event> .
        OPTIONAL{
            ?mainvarb j.4:that ?that .
            ?that j.3:involves ?situationChild}
    }""")

gres6That = g.query(
    """SELECT DISTINCT ?m ?that
    WHERE {
        ?that j.2:involves ?m
    }""")

```

Figura 4.3

```

#casi di possesso di oggetti da parte di un individuo (boatOf, DoorOf, egg.)
gres7 = g2.query(
    """SELECT DISTINCT ?x ?y ?z ?others ?k
    WHERE {
        OPTIONAL{?x j.3:"" + realName + ""Of <http://www.ontologydesignpatterns.org/ont/fred/domain.owl#male_1>} .
        OPTIONAL{?y j.3:"" + realName + ""Of <http://www.ontologydesignpatterns.org/ont/fred/domain.owl#female_1>} .
        OPTIONAL{?z j.3:"" + realName + ""Of <http://www.ontologydesignpatterns.org/ont/fred/domain.owl#neuter_1>} .
        OPTIONAL{?others j.2:"" + realName + ""Of ?k}
    }""")

```

Figura 4.4

#### 4.2.2 Conversione: fattibilità ed ideazione delle regole AMR

A questo punto, si entra nel cuore del progetto: la conversione in AMR.

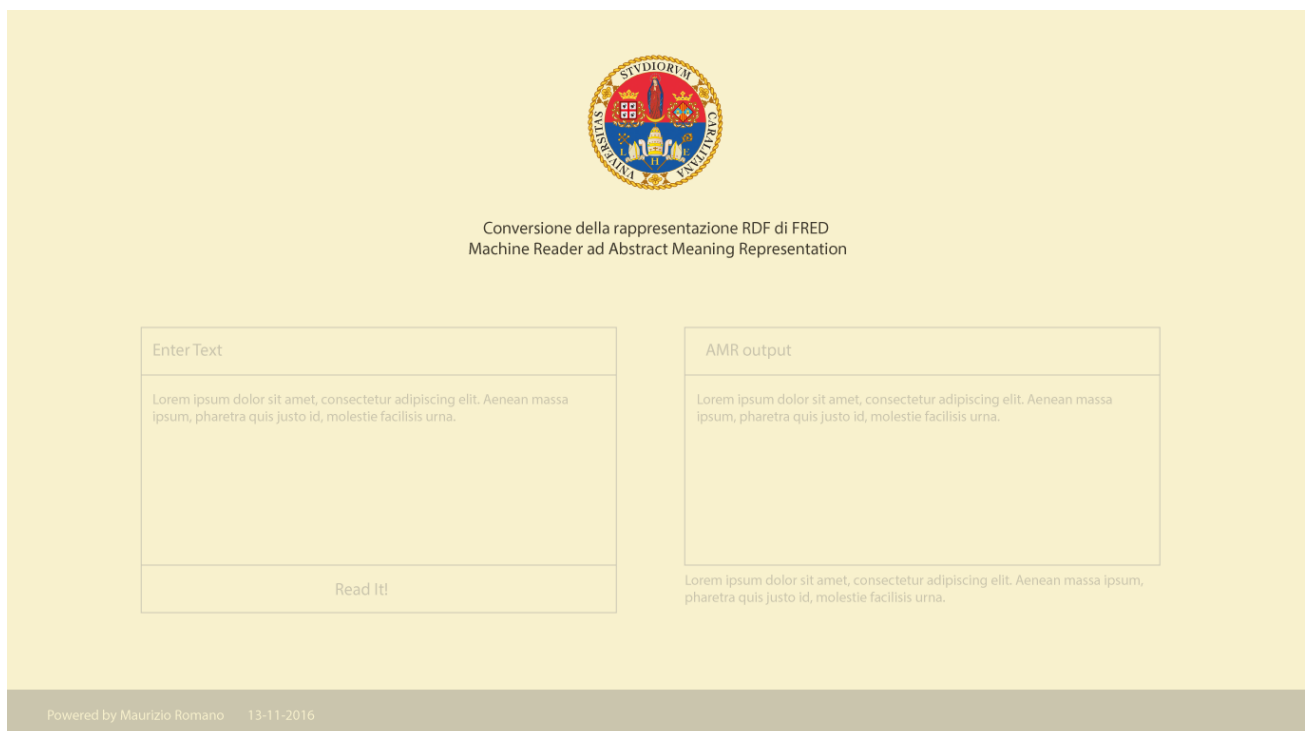
Come detto in precedenza (cap. 2.2) l'AMR possiede un gran numero di regole e, per poter effettuare un'opportuna conversione è fondamentale osservare prima regola per regola se essa possa essere implementata o meno (a causa di limiti dovuti da FRED), successivamente si dovrà procedere con la stesura delle regole fattibili sulla base dei dati ottenuti andando a vedere quali parametri della regola considerata hanno un corretto corrispettivo in FRED e dunque procedere alla relativa implementazione della stessa.

### 4.2.3 *Interfaccia grafica*

La grafica è stata strutturata nel modo più semplice possibile, andando dunque a programmare un'unica pagina costituita da due caselle: una di input della frase in linguaggio naturale ed una in cui verrà figurato l'output della stessa in AMR.

Lo stile minimale assicura un'immediata facilità di comprensione per l'utente delle azioni per lui intraprendibili.

L'uso di FRED, di curl e di rdflib è totalmente trasparente per l'utente in quanto non se ne deve preoccupare.



**Figura 4.5**

### 4.3 Lo Sviluppo

Conclusa la fase di progettazione si è passati a sviluppare concretamente l'Applicativo tramite il programma "Notepad++".

In primis si è pensato alla suddivisione del progetto in opportuni metodi che andassero ad espletare le funzionalità necessarie senza però causare ridondanza all'interno del codice:

- *getName* che fornisce il nome dell'attributo considerato (nome, verbo ecc.);
- *updateAmrVarList* che si occupa di gestire le variabili create dall'AMR;
- *getPrintableAmrName* che si occupa di associare la giusta variabile AMR;
- *printAMR* che stampa tutto l'output AMR nel corretto formato;
- *recArgs* funziona ricorsiva e cuore del programma, effettua tutte le operazioni decisionali a comprendere quando e come invocare la stampa di una data cosa.

#### 4.3.1 Il fulcro dell'Applicazione: le regole di conversione

Il fulcro dell'Applicazione è dunque la creazione delle regole di conversione in AMR:

##### 1 Main verb

Per prima cosa è necessario estrarre i verbi principali dalla frase

*The boy wants the girl to believe him.*

(w / want-01

:ARG0 (b / boy)

:ARG1 (b2 / believe-01

:ARG0 (g / girl)

:ARG1 b))

SELECT DISTINCT ?mainverb WHERE {

?mainverb a ?verb .

?verb rdfs:subClassOf <<http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#Event>>

}

##### 2 Verbs role

Successivamente è necessario estrarre gli argomenti del verbo e mapparli nei ruoli richiesti da AMR:

SELECT DISTINCT ?mainverb ?j\_1 ?j\_2 ?j\_3 ?j\_4 ?j\_5 WHERE {

?mainverb a ?verb .

?verb rdfs:subClassOf <<http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#Event>>

OPTIONAL{?mainverb j.0:Theme ?j\_1} .

OPTIONAL{?mainverb j.0:Agent ?j\_2} .

```

OPTIONAL{ ?mainverb j.0:Experiencer ?j_3 } .
OPTIONAL{ ?mainverb j.0:Patient ?j_4 } .
OPTIONAL{ ?mainverb j.0:Source ?j_5 }
}

```

### 3 Modality

*The boy must not go.*

(o / obligate-01

:ARG2 (g / go-02

:ARG0 (b / boy)

:polarity -))

Mainverb j.2:hasModality ?y

Se ?y è uguale a *Necessary* allora la radice sarà obligate\_1, se invece è uguale a *Possible* allora la radice sarà permit\_1.

### 4 Modality

Mainverb j.2:hasTruthValue ?y

Se ?y è uguale a *False* allora il verbo corrispondente otterrà una polarità negativa in aggiunta ai suoi parametri.

### 5 Focus

*The marble is white.*

(w / white-03

:ARG1 (m / marble))

```

SELECT DISTINCT ?m ?arg
WHERE {

```

```

    ?arg <http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#hasQuality> ?m

```

```

}

```

?m ed ?arg risultati da questa selezione devono essere trattati come un mainverb col suo argomento.

## 6 Focus and situation

*The boy sees that the marble is white.*

(s / see-01

:ARG0 (b / boy)

:ARG1 (w / white-03

:ARG1 (m / marble)))

OPTIONAL{

?mainverb j.4:that ?that .

?that j.3:involves ?situationChild

}

Se uno degli argomenti del mainverb è pari a *thing\_1* ed ha un ?that che riferisce ad una situation, il ?situationChild sostituisce il valore di quell'argomento.

## 7 Possessive adjectives

*His boat*

(b / boat

:poss (h / he))

OPTIONAL{ ?x j.3: "" + thingName + ""Of  
<[http://www.ontologydesignpatterns.org/ont/fred/domain.owl#male\\_1](http://www.ontologydesignpatterns.org/ont/fred/domain.owl#male_1)> } .

OPTIONAL{ ?y j.3: "" + thingName + ""Of  
<[http://www.ontologydesignpatterns.org/ont/fred/domain.owl#female\\_1](http://www.ontologydesignpatterns.org/ont/fred/domain.owl#female_1)> } .

OPTIONAL{ ?z j.3: "" + thingName + ""Of  
<[http://www.ontologydesignpatterns.org/ont/fred/domain.owl#neuter\\_1](http://www.ontologydesignpatterns.org/ont/fred/domain.owl#neuter_1)> } .

OPTIONAL{ ?others j.2: "" + thingName + ""Of ?k }

Se troviamo un nominativo presente come proprietà esprimente una possessione, se è nei primi tre casi allora il parametro poss sarà *he*, *she*, *it* altrimenti sarà pari a ?k

## 8 ARGx-of

*The boy saw the girl who wanted him.*

(s / see-01

:ARG0 (b / boy)

:ARG1 (g / girl

:ARG0-of (w / want-01

:ARG1 b)))

Se un argomento di un verbo è anche argomento di un altro verbo, allora questo collegamento viene espresso con la clausola ARGx-of

#### 9 Question where

*Where did the girl find the boy?*

(f / find-01

:ARG0 (g / girl)

:ARG1 (b / boy)

:location (a / amr-unknown))

OPTIONAL{ ?mainverb j.2:associatedWith ?question }

Se ?question è pari a location, allora il verbo al quale si riferisce avrà in aggiunta un location impostato a amr-unknown

#### 10 Question how

*How did the girl find the boy?*

(f / find-01

:ARG0 (g / girl)

:ARG1 (b / boy)

:manner (a / amr-unknown))

OPTIONAL{ ?mainverb j.2:associatedWith ?question }

Se ?question è pari a manner, allora il verbo al quale si riferisce avrà in aggiunta un manner impostato a amr-unknown

#### 11 That, those, these, this

*That boy*

(b / boy

:mod (t / that))

OPTIONAL{ ?arg  
<<http://www.ontologydesignpatterns.org/ont/fred/quantifiers.owl#hasDeterminer>> ?th }

Se un ARG ha un ?th non nullo, allora tale ARG otterrà in aggiunta un mod impostato a ?th

#### 12 Time

OPTIONAL{ ?mainverb j.0:Source ?j\_5 }

Se un mainverb ha un ?j\_5 non nullo, allora tale mainverb otterrà in aggiunta un time impostato a ?j\_5



### 13 Amr-unknown

*What did the girl find?*

Se alcuni argomenti sono thing\_1 e non sono presenti situationChild allora converti in amr-unknown.

### 14 Quantifiers

OPTIONAL{ ?arg j.6:hasQuantifier ?y }

Se un arg ha un ?y non nullo, allora tale arg avrà un :mod pari a ?y

### 15 Degree

OPTIONAL{ ?arg j.3:\* ?y }

Se un arg ha un derivato di j.3 pari a \*, allora il corrispettivo mainverb avrà un :degree pari a \*

### 16 Numeric Quantities

OPTIONAL{ ?x j.2:denotes ?y }

OPTIONAL{ ?x j.2:hasInterpretant ?k }

Se si trova un ?x con ?y non nullo e contenente in ?y la keyword “Quantity”, allora tale ?x verrà sostituito da un :quant con valore pari a ?k

#### 4.3.2 Limiti di FRED: regole AMR non implementabili

Purtroppo, nonostante le immense capacità espressive di FRED, vi sono delle limitazioni, ciò impedisce di sviluppare opportune conversioni per alcune regole AMR in quanto FRED fornisce dati errati o addirittura non ne fornisce.

*It may rain.*

*It might rain.*

*Rain is possible.*

*It's possible that it will rain*

Impossibili, non appare una modality nell'output di FRED.

*I don't have any money*

FRED in questo caso non restituisce una modalità

*The dress is inappropriate*

In questo caso FRED non restituisce una modalità False col verbo appropriate, impossibile convertire in AMR correttamente

*How fast did the girl run?*

Fred non fornisce alcuna informazione in merito al verbo run od a fast, impossibile tradurre.

*I know who you saw*

FRED non collega “who” ad una persona, impossibile tradurre

*AMR uses :mode to indicate yes-no questions:*

Fred non fornisce alcuna informazione in merito agli interrogativi, impossibile implementare

*:mode is also used for imperatives. Exclamatory imperatives are just imperatives in AMR*  
 Impossibile, FRED non fornisce alcuna informazione in merito agli imperativi.

*:mode expressive is used to mark exclamational words such as ah, ha, hmm, oh, wow, yippee that express emotions, but don't refer to a clear event, object or property. Do not use :mode expressive for mere emphasis (text in ALLCAPS), exclamation marks (!) or disfluency markers (uh), which are not annotated in AMR.*

Fred non fornisce alcuna informazione di questo tipo.

*If a hyphenated word can be broken down into component meanings, we do it:*

(a / account

:mod (m / market

:mod (m2 / money)))

Impossibile, fred considera e fornisce dati solo sulla parola composta e non sulle sue component

*In any case, we never make the hyphen itself (" - ") into an AMR concept.*

Fred invece lo utilizza per le parole composite, l'unica cosa possibile è dunque rimuovere ogni parola composite dalla conversione in AMR.

*The man who is a lawyer*

*The man is a lawyer.*

Problema con FRED ed il verbo "be".

*The boy destroyed the room.*

*The boy's destruction of the room*

*The destruction of the room by the boy*

Dovrebbero tutte essere rappresentate come la prima, ma FRED fornisce dati sensibilmente diversi nei tre casi.

*Investor*

(p / person

:ARG0-of (i / invest-01))

FRED non fornisce scompone una parola nei significati che la compongono.

*Adverbs get stemmed to adjective form*

FRED non fornisce dati in merito.

*The boy and the girl*

FRED non fornisce alcuna informazione inerente l'*and* e le congiunzioni in generale, impossibile tradurre.

*Non-core roles*

Impossibile produrre gli altri ruoli in quanto FRED fornisce informazioni errate od insufficienti.

### *Reification*

Impossibile in quanto FRED non collega *because* ed affini alla causante.

### *The car that is not black*

FRED fornisce dati inconsistenti.

### *Problems behaving.*

FRED fornisce dati inconsistenti.

### *Ordinals, Exact numbers, Approximate numbers, Quantities, Subsets, Mathematical operators*

FRED non converte la rappresentazione testuale dei numeri, dunque non fornisce informazioni in merito.

### *Named Entities, Special Frames for Roles*

FRED non fornisce alcuna informazione inerente ai nomi propri ed ai ruoli delle persone identificate da tali nominativi.

### *dates, times, percentages, phone, email, URLs*

FRED non fornisce alcuna informazione in merito.

## 4.4 Il Prototipo: “GEORGE”

Completando la fase di sviluppo si è giunti ad un prototipo funzionante comprendente tutte le funzionalità di base richieste in fase di progettazione o durante lo sviluppo stesso (figg. da 4.6 a 4.7).

Conversione della rappresentazione RDF di FRED  
Machine Reader ad Abstract Meaning Representation

Enter Text

The boy sees that the marble is white

Read It!

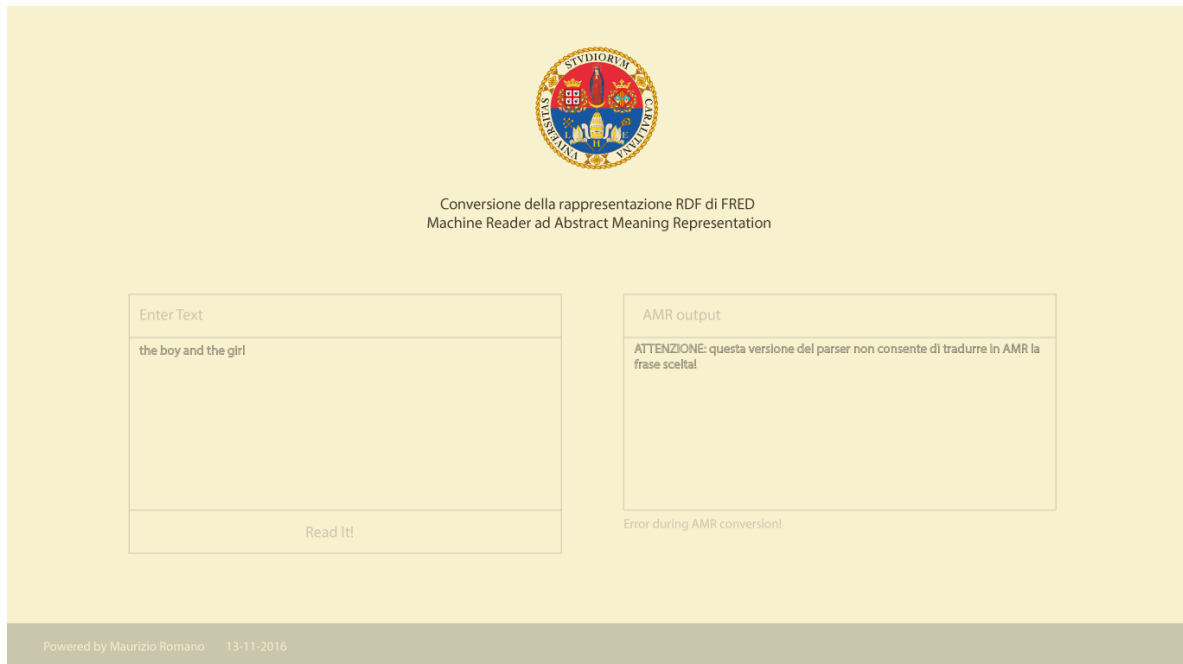
AMR output

```
(s / see_1
  :ARG0 (b / boy_1)
  :ARG1 (w / White
    :ARG1 (m / marble_1)
  )
)
```

Successful AMR conversion!

Powered by Maurizio Romano 13-11-2016

Figura 4.6



The screenshot shows a web application with a yellow background. At the top center is the logo of the University of Bari, featuring a circular emblem with a crown and various symbols. Below the logo, the text reads: "Conversione della rappresentazione RDF di FRED Machine Reader ad Abstract Meaning Representation".

The interface consists of two main panels. The left panel has a text input area labeled "Enter Text" containing the text "the boy and the girl". Below this input is a button labeled "Read It!". The right panel is labeled "AMR output" and contains the message: "ATTENZIONE: questa versione del parser non consente di tradurre in AMR la frase scelta!". Below this message, the text "Error during AMR conversion!" is visible.

At the bottom of the page, a footer bar contains the text: "Powered by Maurizio Romano 13-11-2016".

Figura 4.7

#### 4.5 I Test

Il prototipo, in quanto tale, necessita di test che vadano a verificare la bontà del lavoro svolto.

Per questo motivo sono state prese le frasi utilizzate negli esempi illustrativi delle varie funzionalità dell'AMR e, date in input all'applicativo, si è osservato se l'output fosse esattamente quello riscontrabile in tale guida.

## 5 CONCLUSIONI

L'Applicazione è una valida soluzione ad un problema reale, nonché un considerevole apporto allo sviluppo delle intelligenze artificiali cosicché siano esse sempre più all'avanguardia.

### 5.1 Sviluppi futuri

Essendo un prototipo, l'applicativo necessita sicuramente di opportuni aggiornamenti che in futuro verranno effettuati al fine di migliorarne l'usabilità nonché ampliare funzionalità preesistenti od inserirne di nuove.

A tal fine sono state proposte le seguenti modifiche, necessarie all'evoluzione dello stesso, che in futuro verranno attuate secondo i tempi ed i mezzi a disposizione:

- Arricchire l'insieme delle regole di conversione per poter coprire un numero maggiore di frasi convertibili;
- Ottimizzare ulteriormente il codice in modo da favorirne la manutenzione nonché una maggior celerità di risposta;
- Incremento della portabilità;
- Inserimento di una funzionalità che consenta allo stesso di migliorarsi caso per caso secondo i feedback ricevuti dagli utenti.

*“Ogni tecnologia sufficientemente avanzata  
è indistinguibile dalla magia.”  
Arthur C. Clarke*

---



## **RINGRAZIAMENTI**

Si ringraziano in particolar modo il Relatore Prof. Diego Recupero Reforgiato ed il Correlatore l'Ing. Prof. Gianni Fenu per tutto il supporto dato alla realizzazione dell'Elaborato Finale nonché per la gentilezza e la cordialità usata nei miei confronti nel rispondere alle mie più disparate domande.

Si ringrazia tutto il Corso di Laurea in Informatica per i preziosi insegnamenti e l'impeccabile preparazione ricevuta.

Si ringraziano inoltre tutte quelle persone che direttamente o indirettamente hanno preso parte alla realizzazione del Progetto.





## APPENDICE A

**Ambiente di sviluppo:** Notepad++ 6.7.1, Python 3.5.1, pip 8.1.2, rdflib 4.2.1, pyrcurl 7.43.0 in Windows 7 Professional 64bit

**Numero di linee di codice prodotto inclusive di commenti:** 617

**Valutazione del tempo speso nella costruzione del sistema in ore uomo:** 65 giorni/uomo

**Commento sul lavoro svolto:** il lavoro svolto è stato alquanto soddisfacente poiché trattasi di un prodotto inesistente al mondo, inoltre, cosa ben più importante, è in grado di rendere automatizzato un lavoro che attualmente viene fatto a mano.

Particolarmente interessante è stato l'utilizzo della libreria rdflib che ha consentito di effettuare query SPARQL sull'RDF ottenuto da FRED con molta semplicità.

Qui in Figura A è possibile osservare l'utilizzo della query principale per l'estrapolazione del mainverb e di tutte le proprietà ad esso affini tramite rdflib:

```
#creo un grafo
g = Graph()

#inserisco l'RDF ottenuto da FRED
g.parse("TEMP.rdf")

#lancio la query SPARQL principale
qres = g.query(
    """SELECT DISTINCT ?mainverb ?j_1 ?j_2 ?j_3 ?that ?question ?j_4 ?j_5
    WHERE {
        ?mainverb a ?verb .
        ?verb rdfs:subClassOf <http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#Event> .
        OPTIONAL{?mainverb j.0:Theme ?j_1} .
        OPTIONAL{?mainverb j.0:Agent ?j_2} .
        OPTIONAL{?mainverb j.0:Experiencer ?j_3} .
        OPTIONAL{?mainverb j.4:that ?that} .
        OPTIONAL{?mainverb j.2:associatedWith ?question} .
        OPTIONAL{?mainverb j.0:Patient ?j_4} .
        OPTIONAL{?mainverb j.0:Source ?j_5}
    }""")
```

Figura A

## BIBLIOGRAFIA

Stephanie Strassel, “Abstract Meaning Representation”, 2016, <http://amr.isi.edu/>, 13/11/2016

Wikipedia, “Resource Description Framework”, 2016,  
[https://it.wikipedia.org/wiki/Resource\\_Description\\_Framework](https://it.wikipedia.org/wiki/Resource_Description_Framework), 13/11/2016

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, Nathan Schneider, “Abstract Meaning Representation (AMR) 1.2.2 Specification”, 2016, <https://github.com/amrisi/amr-guidelines/blob/master/amr.md#part-i-introduction>, 13/11/2016

Aldo Gangemi, Valentina Presutti, Diego Reforgiato Recupero, Andrea Giovanni Nuzzolese, Francesco Draicchio, Misael Mongiovì, “Semantic Web Machine Reading with FRED”, 2016, <http://semantic-web-journal.org/system/files/swj1379.pdf>, 13/11/2016

Wikipedia, “Web semantico”, 2016, [https://it.wikipedia.org/wiki/Web\\_semantico](https://it.wikipedia.org/wiki/Web_semantico), 14/11/2016

Wikipedia, “Elaborazione del linguaggio naturale”, 2016,  
[https://it.wikipedia.org/wiki/Elaborazione\\_del\\_linguaggio\\_naturale](https://it.wikipedia.org/wiki/Elaborazione_del_linguaggio_naturale), 14/11/2016