

Sistemi di Elaborazione dell'informazione II

Corso di Laurea Specialistica in Ingegneria Telematica

II anno – 4 CFU

Università Kore – Enna – A.A. 2008-2009

Alessandro Longheu

<http://www.dit.unict.it/users/alongheu>

alessandro.longheu@dit.unict.it

Dati Semistutturati: Ontologie

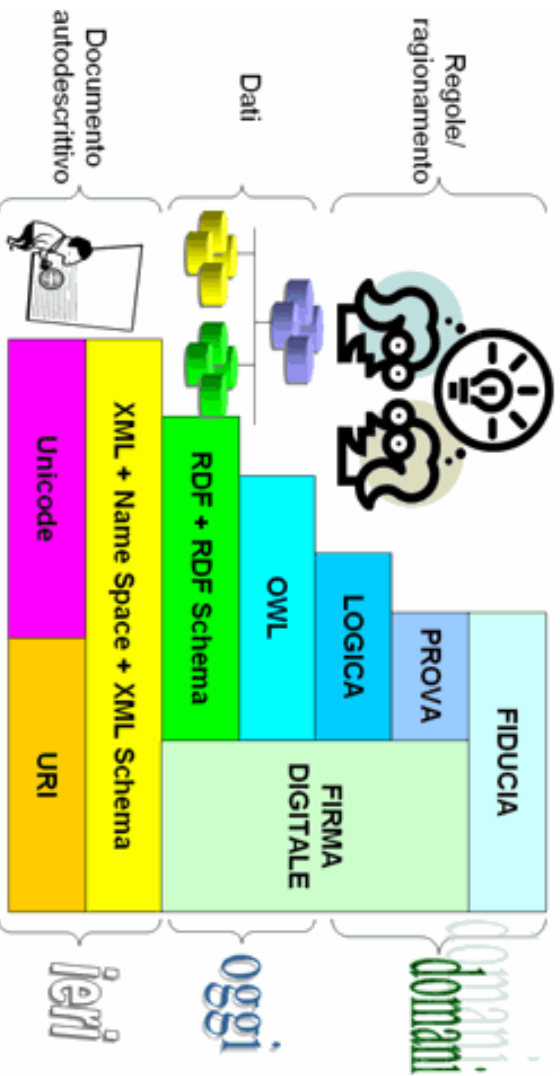
A. Longheu – Sistemi di Elaborazione delle Informazioni II

Schema Lezione

- Introduzione al Livello Ontologico
- Ontologie: Definizione e Obiettivi
- Modelli e Implementazioni di Ontologie: WordNet, Cyc, SUMO, DOLCE
- Linguaggi per la descrizione di Ontologie: OWL
- Linguaggi + Modelli → Creazione di Ontologie: metodologia e tools (OntoLearn, Protege, OntoEdit)
- Esempi d'uso di Ontologie, in particolare:
 - Utilizzo delle Ontologie per l'annotazione semantica, metodi e tools

Introduzione al Livello Ontologico

- Richiamo sul “Semantic Web Wedding Cake”:



3

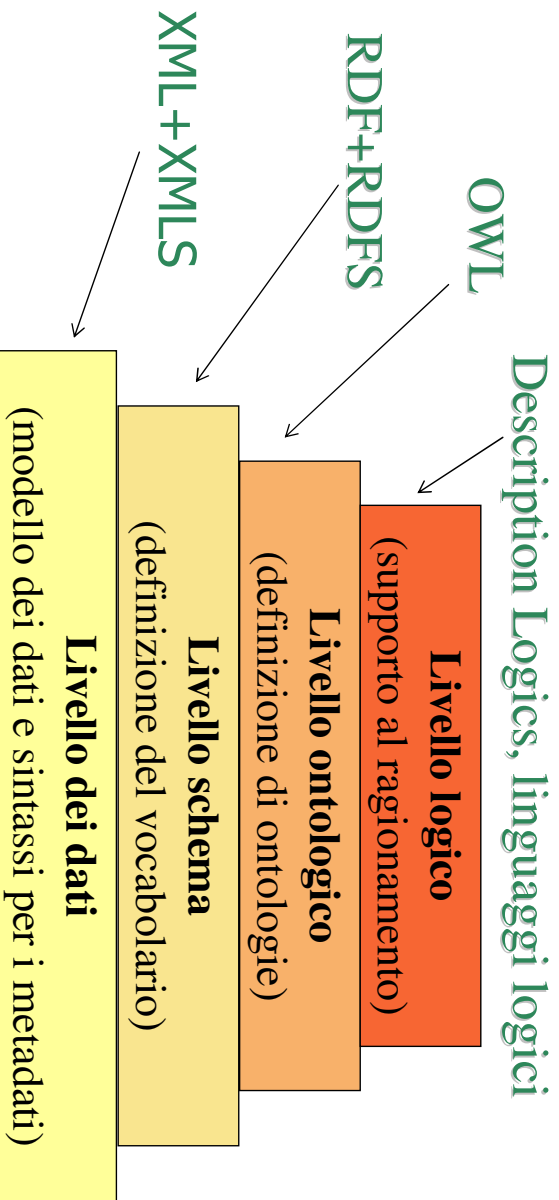
Introduzione al Livello Ontologico

- il primo livello (URI) offre solo uno strumento per la localizzazione delle risorse
- Il secondo livello (annotazione) permette di descrivere un documento, una sua parte, o una intera collezione; questa descrizione può essere manuale (linguaggi di mark-up come XML, strumenti elementari come metadata di HTML) o automatica (software)
- Il terzo livello (RDF) è un modello generale (e minimalista) per fare asserzioni semantiche su oggetti Web (risorse); una Risorsa è qualsiasi entità identificabile mediante un URI (Uniform Resource Identifier); ciò include pagine Web e documenti XML.
- RDF e XML hanno una semantica non esplicita, ed inoltre l'annotazione è manuale. Gran parte del testo è infine annotata con “tag” generiche, che non descrivono il contenuto.
- Il **livello ontologico**, basato sul linguaggio OWL ma non solo, è un modello semantico ed un insieme di regole per “ragionare” sui dati a disposizione.

4

Introduzione al Livello Ontologico

Nell'ambito del Web Semantico, si è giunti a un insieme di linguaggi standard (W3C), ciascuno costruito sul precedente:



5

Introduzione al Livello Ontologico

HTML

Non distingue tra struttura del contenuto e rappresentazione del documento

XML

Permette di definire i propri tag e di strutturare le informazioni indipendentemente dalla loro futura ed eventuale resa grafica

XMLS

Definisce la struttura sintattica di un documento XML

RDF

Permette di esprimere relazioni esistenti tra istanze concettuali
(es. Autore-di(Buzzati, Deserto dei tartari))

RDFS

Permette di definire relazioni e concetti
(es. Autore-di è una relazione, Libro è un concetto)

OWL

Poggia su RDFS e permette di esprimere pienamente classi, relazioni, istanze, vincoli, ecc.

6

Ontologie: Definizione e Obiettivi

- Diverse sono le **definizioni di ontologia**:
 - Filosofia: “a systematic explanation of being”
 - Neches : “...defines the basic terms and relations including the vocabulary of a topic area as well as the rules for combining terms and relations to define extensions to the vocabulary.”
 - Gruber, la più citata: “...an explicit specification of a conceptualization”
 - Borst, leggermente modificata: “...a formal specification of a shared conceptualization”
 - Guarino: “...a logical theory which gives an explicit, partial account of a conceptualization”

7

Ontologie: Definizione e Obiettivi

Formalmente **un'ontologia O è una tripla (C, R, A)**

dove:

- C è un insieme di concetti
 - R è un insieme di relazioni concettuali che esprimono le interconnessioni semantiche tra concetti. Ogni relazione in R è definita su $C \times C$.
 - A è un insieme di assiomi che permettono di inferire nuovi fatti a partire da quelli codificati (se $A = \emptyset$ l'ontologia non è assiomatizzata)
- Gli insiemi C ed R individuano un **grafo** $G = (V, E)$:
- $V \equiv C$
 - $E = \{ (c1, c2) \in C \times C : \exists S \in R : (c1, c2) \in S \}$
- e una **funzione di etichettatura** $L : C \times C \rightarrow 2^R$ tale che $L(c1, c2) = \{ S \in R : (c1, c2) \in S \}$.

8

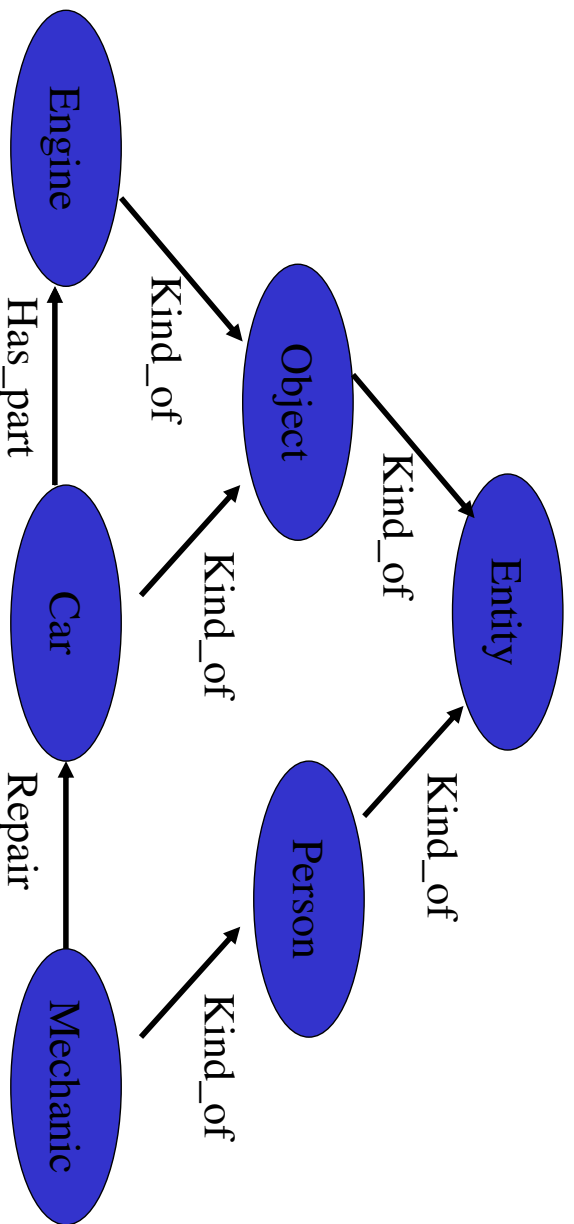
Ontologie: Definizione e Obiettivi

Un **esempio** di ontologia è $O' = (C', R', A')$, dove:

- $C' = \{ \text{Entità, Oggetto, Persona, Meccanico, Automobile, Motore} \}$
- $R' = \{ \text{è-un, ha-un, ripara} \}$
 - è-un = $\{ (\text{Oggetto, Entità}), (\text{Persona, Entità}), (\text{Meccanico, Persona}), (\text{Automobile, Oggetto}), (\text{Motore, Oggetto}) \}$
 - ha-un = $\{ (\text{Automobile, Motore}) \}$
 - ripara = $\{ (\text{Meccanico, Automobile}) \}$
- $A' = \{ \text{"}\forall a \in \text{Automobile } \exists m \in \text{Meccanico} : \text{ripara}(p, a)\text{"} \}$

9

Ontologie: Definizione e Obiettivi



10

Ontologie: Definizione e Obiettivi

- **Ma esattamente a cosa servono le Ontologie?**
- Il Semantic Web: rendere i dati “machine-understandable” → soluzione: esprimere in modo formale il contenuto informativo di risorse web
- In che modo? Attraverso l’ **annotazione semantica**
 - Un’annotazione semantica esprime informazioni semantiche associate a risorse web.
 - Un’ annotazione basata su ontologie esprime in modo formale il significato di una risorsa web o di una porzione di essa, utilizzando la terminologia fornita da un’ **Ontologia di Riferimento (RO)**
 - Nota: un ontologia si esprime attraverso un linguaggio di rappresentazione della conoscenza (XML, RDF, OWL, DAML+OIL...)
- quindi, XML e RDF sono strumenti per l’annotazione; se tale annotazione utilizza una ontologia di riferimento, diventa semantica e permette la realizzazione del Web semantico

11

Ontologie: Definizione e Obiettivi

- **Più in generale, qual è l’obiettivo delle Ontologie?**
- Migliorare la comunicazione fra persone e organizzazioni
- Favorire l’interoperabilità fra sistemi
 - Condividere metodi di modellazione, paradigmi, linguaggi, e strumenti software
 - Supportare l’ingegnerizzazione di sistemi IT (Information Technology):
 - **favorire riusabilità/condivisibilità** : condivisione delle rappresentazioni formali
 - **migliorare la ricerca**: usata come meta-data per indicizzare databases documenti e sistemi informativi in generale
 - **esprimere specifiche**: aiuta nell’identificare i requirements di un sistema IT
 - **acquisire conoscenza** (generalizzare, ragionare)
- Tramite le ontologie abbiamo la possibilità di esprimere direttamente la struttura della nostra conoscenza e permettere alle macchine di elaborare automaticamente la conoscenza stessa, non più solo le semplici informazioni. In questo modo passiamo dalla semplice **informatica** (elaborazione automatica di informazioni) ad una **epistematica**: un’² elaborazione automatica di conoscenza.

Ontologie: Definizione e Obiettivi

- La creazione, gestione e sfruttamento delle ontologie richiede:
 - un **modello**, che offre delle linee guida per la organizzazione ragionata della conoscenza
 - un **linguaggio**, che rappresenta invece lo strumento con cui l'organizzazione ragionata viene effettivamente implementata
- **Diversi sono i modelli attualmente presenti**, che hanno portato a diverse ontologie: *WordNet*, *Cyc*, *SUMO*, *DOLCE*
- I **linguaggi disponibili** sono diversi, ad esempio *Loom*, *DAML* (definito dall'americana DARPA) ed *OIL* (sponsorizzato dalla Unione Europea nell'ambito del programma IST), successivamente confluiti in *DAML+OIL*. Su questa base il W3C ha definito *OWL*, linguaggio attualmente principe
- Con *OWL* è possibile scrivere delle ontologie che descrivono la conoscenza che abbiamo di un certo dominio, tramite classi, relazioni fra classi e individui appartenenti a classi. La conoscenza così formalizzata è processabile automaticamente da un calcolatore, tramite un ragionatore automatico che implementa i processi inferenziali e deduttivi.¹³

Ontologie: Modelli e Implementazioni

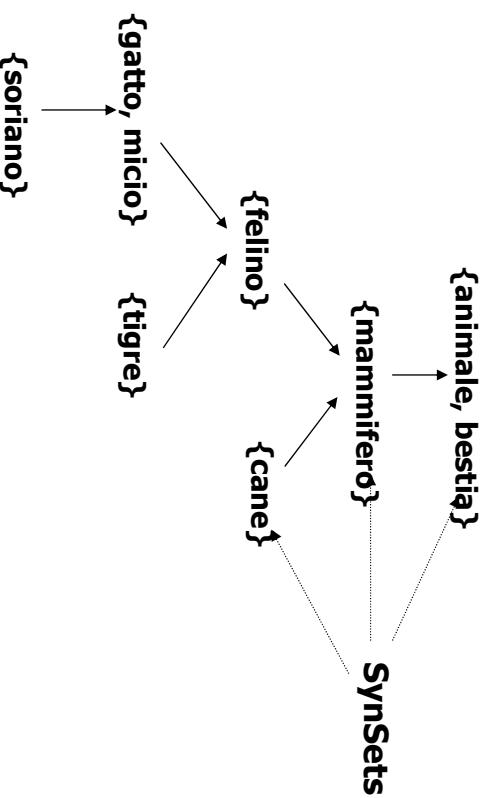
- Nel seguito si considerano alcuni modelli di Ontologie presenti sul mercato:
 - *WordNet*
 - *Cyc*
 - *SUMO*
 - *DOLCE*

Ontologie: Modelli e Implementazioni

WORDNET: Da un elenco di parole:

<tigre, cane, animale, mammifero, bestia, micio, soriano, gatto, felino>

A un dizionario strutturato:



NB: La stessa parola può appartenere a più SynSets

15

Ontologie: Modelli e Implementazioni

Alcune versioni:

WordNet: Cognitive Science Laboratory dell'Università di Princeton (inglese) Fine anni '80

EuroWordNet: su Fondi dell'Unione Europea.

(multilingue - ILC-Pisa per l'italiano) Metà anni '90

ItalWordNet: IRST-ICT (Trento). Un progetto nazionale (italiano) Fine anni '90

Ovviamente, molte versioni per altre lingue

16

Ontologie: Modelli e Implementazioni

Le relazioni di WordNet (originale)

Relazione	POS collegate	Esempi
Sinonimia	nome/nome; verbo/verbo; aggettivo/aggettivo; averbio/averbio	book/volume; to eat/to take in; great/outstanding; greatly/drastically
Antonimia	nome/nome; verbo/verbo; aggettivo/aggettivo	man/woman; to enter/to exit; long/short
Iponimia	nome/nome; verbo/verbo	slice/knife; to walk/to move
Meronomia	nome/nome	head/nose
Implicazione	verbo/verbo	to buy/to pay
Causa	verbo/verbo	to kill/to die
Somiglianza	aggettivo/aggettivo	wet/humid
Attributo	aggettivo/nome	tall/stature
Relativo al nome	aggettivo/nome	fraternal/brotherhood

17

Ontologie: Modelli e Implementazioni

Un difetto di Wordnet è che alcune delle relazioni adottate sono alquanto vaghe

POS sta per Part Of Speech (nome, verbo, ecc.)

POS diverse collegate solo in casi particolari (es. partecipazione (nome) e partecipare(verbo))

18

Ontologie: Modelli e Implementazioni

Le relazioni di EuroWordNet:

Relazione	Ordini collegati	Esempi
Sinonimia	1/1; 2/2; 3/3	barriera/ostacolo;comprare/acquistare;conoscenza/cognizione
Quasi sinonimia	1/1; 2/2; 3/3	ordigno/congegno; certificare/assicurare
Xpos quasi sinonimia	2/2	arrivo/arrivare
Antonimia	1/1; 2/2; 3/3	incredibile/credibile
Quasi antonimia	1/1; 2/2; 3/3	sopra/sotto; arrivare/partire
Xpos quasi antonimia	2/2	arrivo/partire
Iponimia	1/1; 2/2; 3/3	cane/animale; agitarsi/muoversi
Xpos iponimia	2/2	arrivo/andare; martellata/colpire
Meronomia	1/1	braccio/corpo; mano/dito
Causa	2/2	uccidere/morire; giustificare/condanna
Sotto-evento	2/2	comprare/pagare; dormire/russare
Ruolo	1/2	martello/martellare; pedone/camminare
Co_ruolo	1/1	chitarrista/chitarra
Stato_di	1/2	povero/povertà; vecchio/vecchiaia
Maniera_per	2/2	bisbigliare/a-bassa-voce; trucidare/barbaramente
Derivazione	Tra tutti	acqua/acquaiolo
Relativo_a	2/2	presidenziale/presidente
Classe	1/1	Po/ fiume; Roma/città

19

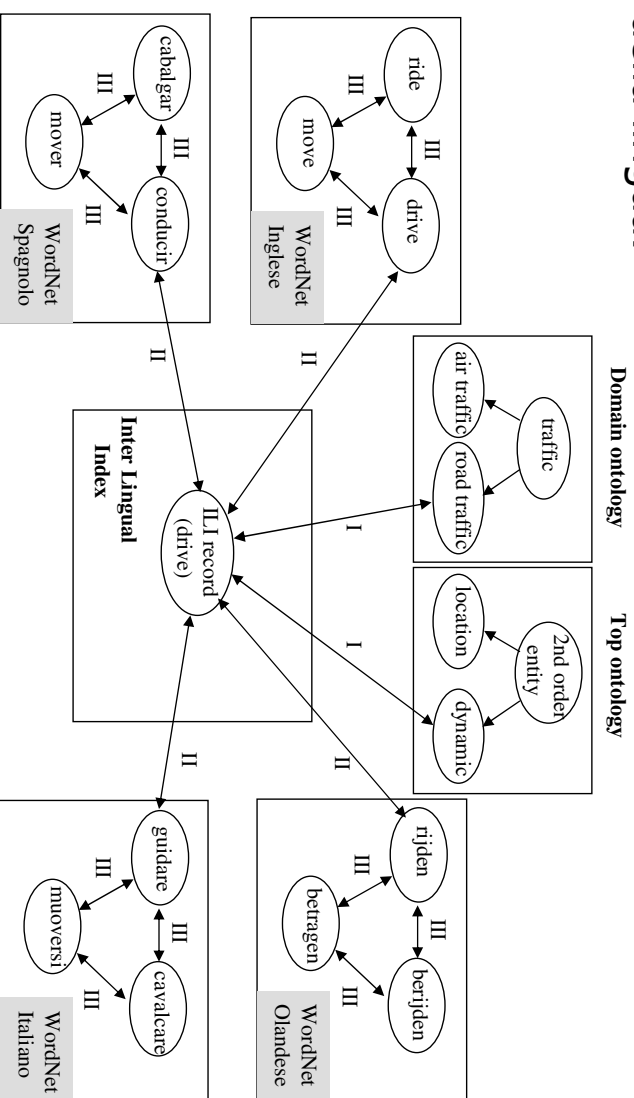
Ontologie: Modelli e Implementazioni

Osservazioni:

- La seconda colonna è molto diversa da quella della tabella precedente. I numeri (1, 2, 3) si riferiscono ai cosiddetti **“ordini semantici”**, così definiti:
 - 1: nomi concreti
 - 2: nomi, verbi, aggettivi o averbi indicanti proprietà, stati, processi o eventi
 - 3: nomi astratti indicanti proposizioni indipendenti dal tempo e dallo spazio
- Per molte relazioni sono definite anche le inverse, che per semplicità non sono riportate in tabella (ad es. iponimia → iperonimia; meronomia → olonomia; causa → causato_da; ...)
- Alcune relazioni non sono definite tra Synset, ma tra singole parole. Questo vale ovviamente per la sinonimia, ma anche per la derivazione e per l'antonimia
- Tutti i Synset coinvolti si riferiscono a classi (chitarra, andare, ...) eccetto quelli che compaiono nell'ultima relazione, in cui uno dei due elementi collegati è un'istanza (Po, Roma)

Ontologie: Modelli e Implementazioni

- L'Architettura di EuroWordNet, che tiene conto anche della lingua:



21

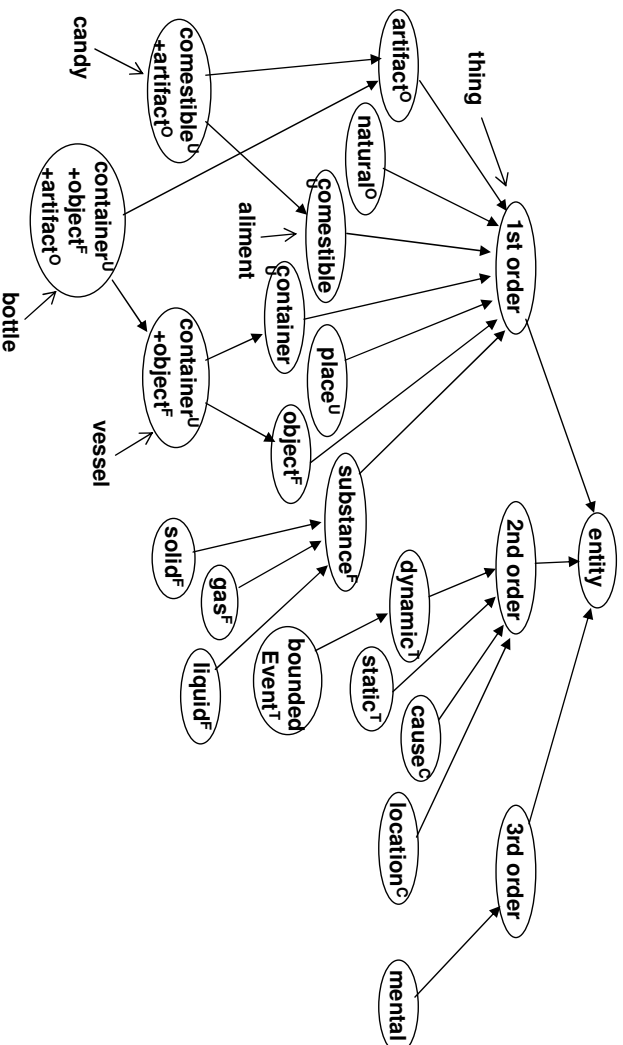
Ontologie: Modelli e Implementazioni

- Nell'architettura di EuroWordNet:
- La **top-ontology** è una rappresentazione strutturata dei concetti più generali
- Le Domain-Ontologies sono elenchi (parzialmente strutturati) di Campi Semantici (Sport, calcio, astronomia, ...)
- L'Inter Lingual Index (ILI) è solo una tabella di mapping tra Synset, non strutturata
- Vi sono tre tipi diversi di archi:
 - I: archi indipendenti dalle varie lingue, che collegano un record dell'ILI con le top e domain ontologies
 - II: archi che collegano i synset dei vari WordNet all'ILI (e viceversa)
 - III: archi, dipendenti dalla particolare lingua, che collegano i vari synset.

22

Ontologie: Modelli e Implementazioni

- La Top-Ontology di EuroWordNet:



23

Ontologie: Modelli e Implementazioni

- La definizione dei concetti è basata su **features**:
- Per le entità del primo ordine (concreti) esse sono:
 - Origin (se è naturale o artificiale; indice O nella figura)
 - Natural
 - Living
 - Plant
 - Human
 - Creature
 - Animal
- Artifact
 - Form (se è una sostanza o un oggetto con forma definita; indice F nella figura)
- Substance
 - Solid
 - Liquid
 - Gas
- Object

24

Ontologie: Modelli e Implementazioni

- Composition (se è un oggetto unitario o un gruppo; non compare in figura)
 - Part
 - Group
- Function (la funzione)
 - Vehicle
 - Representation
 - MoneyRepresentation
 - LanguageRepresentation
 - ImageRepresentation
- Software
- Place
- Occupation
- Instrument
- Garment
- Furniture
- Covering
- Container
- Comestible
- Building

25

Ontologie: Modelli e Implementazioni

- Per le **entità del secondo ordine** (azioni, astrazioni, qualità) esse sono:
 - Situation Component (una caratteristica o un partecipante della situazione descritta; indice C nella figura)
 - Cause
 - Communication
 - Condition
 - Existence
 - Experience
 - Location
 - Manner, Mental, Modal, Physical, Possession, Purpose, Quantity, Social, Time
 - Situation Type (il tipo della situazione; indice T in figura)
 - Dynamic
 - BoundedEvent
 - UnboundedEvent
 - Static
 - Property
 - Relation
- Le **entità del terzo ordine non hanno features**

26

Ontologie: Modelli e Implementazioni

- Le features possono essere combinate per ottenere concetti più specifici (ad esempio, si veda in figura `containerU+objectF+artifatto`)
- I concetti del Top-level non sono dei SynSets! Per cui termini come Container debbono comparire sia nella top ontology, che nei SynSet
- Alla Top-ontology sono agganciati 1310 Base Concepts, concetti fondamentali comuni a tutte le lingue, individuati in base a 3 criteri:
 - Numero delle relazioni associate ad essi
 - Posizione nella gerarchia tassonomica
 - Frequenza in un corpus
- Non sono predefiniti dei meccanismi inferenziali, che permettano di sfruttare la semantica delle relazioni. ItalWordNet è memorizzato in un DB relazionale, ed è compito dell'utente scrivere le query opportune per spostarsi tra i Synsets. Esistono però dei browser grafici che permettono di navigare (manualmente) nella rete

27

Ontologie: Modelli e Implementazioni

- Il **progetto Cyc** (da enCYClopedia) nasce nel 1984 ed è ancora in corso (si veda il sito <http://www.opencyc.org/>).
- Attualmente, Cyc include oltre un milione di concetti, mentre la versione pubblica OpenCyc comprende circa centinaia di migliaia di concetti e milioni di relazioni tra di essi (asserzioni)
 - How much does a program need to know to begin with? The annoying, inelegant, but apparently true answer is: a non-trivial fraction of consensus reality - the millions of things that we all know and that we assume everyone else knows" (Güha & Lenat, 1990). Cyc cerca di soddisfare questa "banale" esigenza coprendo una "non-trivial fraction" di termini

28

Ontologie: Modelli e Implementazioni

- Cyc Presenta due livelli

Constraint Language (Logica dei predicati, livello inferenziale)
Cycl (linguaggio basato su frame, descrive le asserzioni elementari - Units)

29

Ontologie: Modelli e Implementazioni

Livello Units:

- I frame di Cyc si chiamano Units. Esse includono, come tutti i frames, degli slots
- Esempio di Unit relativa ad un'istanza
 - # \$Texas
 - # \$capital: (# \$Austin)
 - # \$residents: (# \$Doug Guha Mary)
 - # \$stateOf: (# \$UnitedStatesOfAmerica)
- Tutti i simboli che hanno il prefisso # \$ sono Units. Una caratteristica fondamentale di Cyc è che, come si vede dall'esempio, anche gli slots sono Units (SlotUnits)

30

Ontologie: Modelli e Implementazioni

- Esempio di Unit relativa ad uno slot

```

#residents
#$instanceOf: (##$Slot)
#$inverse: (##$residentOf)
#$makesSenseFor: (##$GeopoliticalRegion)
#$entryISA: (##$Person)
#$specSlots: (##$lifelongResidents #illegalAliens #registeredVoters)

```

- Da un punto di vista formale, gli slots sono relazioni binarie
- Quindi, è necessario definire il dominio (##\$makesSenseFor) e il range (##\$entryISA)
- E' anche possibile definire relazioni tra relazioni (##\$inverse e ##\$specSlots)

31

Ontologie: Modelli e Implementazioni

- Alcuni meccanismi inferenziali in Cycl

Mantenimento di relazioni inverse: esempio precedente (\$inverse)

Mantenimento di specSlot-genSlot: alcuni slot di una unit possono essere legati da una relazione di specializzazione-generalizzazione. Ad es.

```

#$padreDi ##$specSlot ##$genitoreDi
Se viene inserita l'informazione
(##$Luigi ##$padreDi ##$Marta)
Cycl introduce automaticamente
(##$Luigi ##$genitoreDi ##$Marta)

```

TransferThro: Il valore di uno slot può essere trasferito a Unit collegate:

```

##$libro ##$scrittoIn ##$linguaggio
##$libro ##$parteDiTesto ##$capitolo
##$scrittoIn ##$transfersThro ##$parteDiTesto
Se viene inserita l'informazione
(##$I_demoni ##$scrittoIn ##$russo)
(##$I_demoni ##$parteDiTesto ##$I-demoni-cap-1)
Cycl introduce automaticamente
(##$I_demoni-cap-1 ##$scrittoIn ##$russo)

```

32

Ontologie: Modelli e Implementazioni

■ Alcuni meccanismi inferenziali in CycL

Ereditarietà: ben noto: “Se tutte le persone hanno un codice fiscale e i professori sono persone, allora i professori hanno un codice fiscale”.

In realtà, Cyc estende questo meccanismo:

Ereditarietà standard:

Si applica allo slot *#\$allInstances* (tutte le istanze di una unit);
Se

#\$persona

 #\$nazionalità: (#\$stato)

#\$studenteUnivRoma

 #\$genl: (#\$persona)

#\$nazionalità

 "default per #\$studenteUnivRoma = #\$Italia”

Allora, quando si asserisce

#\$studenteUnivRoma #\$allInstances (... #\$Sandra ...)

Cyc ottiene (per default)

#\$Sandra #\$nazionalità #\$Italia

33

Ontologie: Modelli e Implementazioni

Conclusioni su Cyc

■ Un sistema enormemente complesso, che include sia una parte di rappresentazione di conoscenza e inferenza, sia una ontologia vera e propria

■ VANTAGGI:

- Dimensione
- Potenza inferenziale
- Ottimizzazione dei ragionamenti

■ SVANTAGGI:

- Troppo complesso
- Non chiare le scelte ontologiche
- Alcuni insuccessi (es. Legami col linguaggio naturale)

34

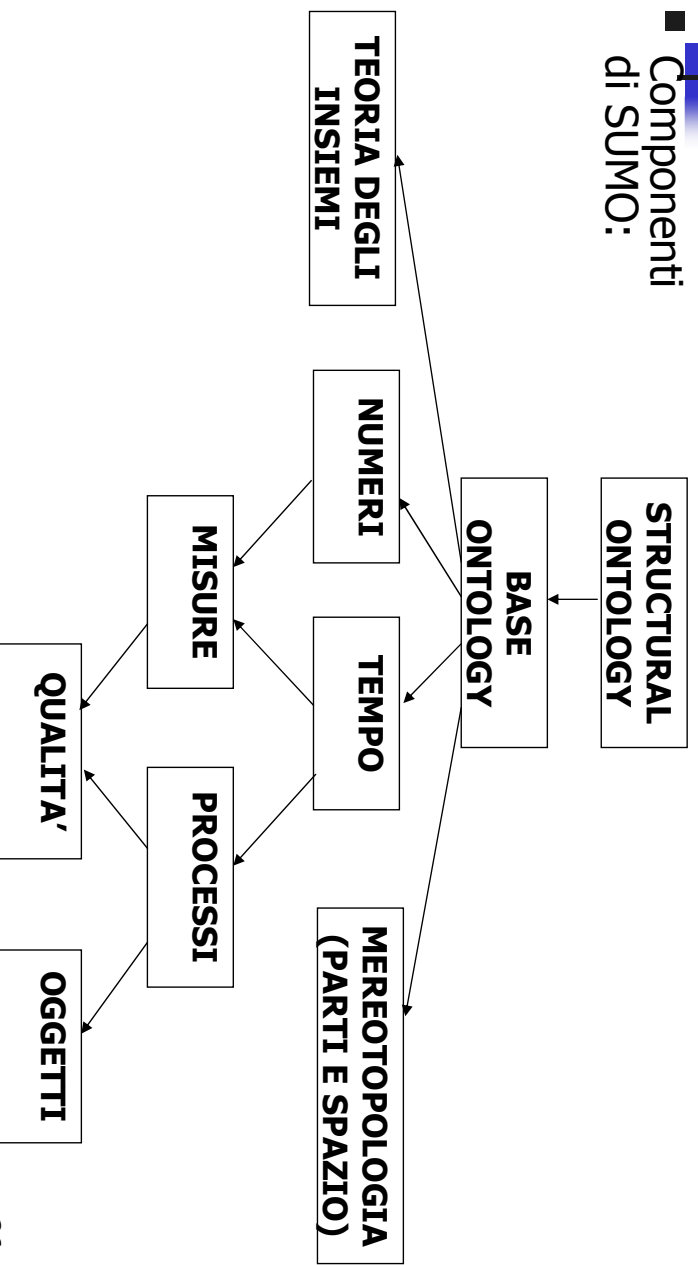
Ontologie: Modelli e Implementazioni

- SUMO (Suggested Upper Merged Ontology) è il risultato di uno sforzo dell'IEEE (Institute of Electrical and Electronic Engineering), iniziato negli anni '90: <http://sumo.ieee.org>
- Questo standard specificherà una 'upper ontology' che i computer potranno utilizzare per applicazioni quali l'interoperabilità dei dati, la ricerca e il reperimento di informazioni, il ragionamento automatico e l'elaborazione del linguaggio naturale.
- Un'ontologia consiste di un insieme di concetti, assiomi e relazioni che descrivono un dominio di interesse. Una **'upper ontology'** è limitata a concetti che sono 'meta', generici, astratti e filosofici e, di conseguenza, sono sufficientemente generali da coprire (ad alto livello) un ampio range di domini. I concetti relativi ai domini specifici non saranno inclusi, ma questo standard fornirà una struttura e un insieme i concetti generali sulla base dei quali potranno essere costruite le ontologie di dominio (ad es. medicina, finanza, ingegneria, ecc.)."

35

Ontologie: Modelli e Implementazioni

- Componenti di SUMO:



36



Caratteristiche di SUMO

- Una vera ontologia: non ci si preoccupa di come si effettuano i ragionamenti, ma solo di 'descrivere' i concetti e le loro proprietà
- Il linguaggio in cui è espresso SUMO si chiama KIF (Knowledge Interchange Format), ed è esso che si occupa della parte inferenziale (l'equivalente di Cycl+Constraint)

VANTAGGI:

- Separazione conoscenza-reasoning
- Ontologia piuttosto ampia
- Integrazione di conoscenza da fonti diverse

SVANTAGGI:

- Scelte ontologiche più chiare di Cyc, ma ancora dubbie
- L'insieme degli assiomi è piuttosto limitato
- Efficienza dei ragionamenti

37

Dolce

(un ponte tra filosofia e computer science)

- Dolce (Descriptive Ontology for Linguistic and Cognitive Engineering) è il risultato dell'attività svolta sulle ontologie presso l'Istituto per le Scienze e le Tecnologie Cognitive del CNR (Trento-Roma)
- Dolce non si propone come candidata per un'ontologia universale, ma come un punto di partenza per confrontare e chiarire la relazioni esistenti tra diversi moduli ontologici e per rendere esplicite le assunzioni presenti nelle ontologie esistenti
- Dolce ha un orientamento cognitivo, nel senso che tende ad esprimere le categorie ontologiche alla base del linguaggio naturale e del senso comune umano.
- Insieme a Dolce c'è Ontoclean, che è una metodologia per la verifica di ontologie.

38

Dolce

(un ponte tra filosofia e computer science)

- Quattro nozioni fondamentali nella caratterizzazione dei concetti di Ontoclean:
 - **Identità**: possibilità di distinguere due istanze di una classe mediante una proprietà caratteristica (anche in tempi diversi), ad esempio per 'Person': 'avere le stesse impronte digitali'
 - **Dipendenza**: la proprietà P è dipendente dalla proprietà Q, se, quando Q è vera, è vera anche P.
 - Ex. 'avere figli' dipende da 'essere genitore'
 - **Rigidità**: dice, di una proprietà, se è 'stabile' per le istanze.
 - Ex. 'Person' è rigida; 'Student' non è rigida
 - **Unità**: possibilità di identificare tutte le parti di un'entità mediante una relazione unificante
 - Ex. 'Azienda': 'essere stati assunti in quell'azienda'

39

Dolce

(un ponte tra filosofia e computer science)

- A cosa servono le quattro nozioni? A creare vincoli sulla relazione di sussunzione ("incorporare qualcosa in un una categoria più generale")
- Una classe non rigida (-R) non può sussumere una classe rigida (+R), es. 'Legal Agent' non può sussumere 'Person'
- Una classe che ha una condizione di identità (+I) non può sussumere una classe che manca di una condizione di identità (-I), es. 'Park' non può sussumere 'Location'
- Una classe che ha una condizione di unità (+U) non può sussumere una classe che manca di una condizione di unità (-U), ad es. 'Amount of Matter' non può sussumere 'Physical Object': se una quantità d'acqua ne perde una parte è una diversa quantità d'acqua, ma se a una persona si tagliano i capelli rimane la stessa persona

40

Dolce

(un ponte tra filosofia e computer science)

- Gli Endurants sono 'completamente' presenti (cioè con tutte le loro parti) quando l'Endurant stesso è presente. Esempio classico: gli oggetti (ad es. una casa)
- Ad ogni istante temporale, solo una parte (una 'fetta' temporale) di un Perdurant è presente (ad es. gli eventi)
- Correlato al concetto di cambiamento: solo gli Endurant possono cambiare (pur mantenendo la loro identità, concetto questo non facile da caratterizzare ontologicamente), mentre è ovvio che i Perdurant non cambiano, essendo le loro 'parti' diverse nel tempo per la definizione stessa di Perdurant
- Relazione fondamentale tra Endurant e Perdurant: partecipazione; gli Endurant partecipano (e cioè hanno un ruolo) nei Perdurants; ad esempio io, che sono un Endurant, partecipo nella mia azione (un Perdurant) di andare dall'Università a casa.

41

Dolce

(un ponte tra filosofia e computer science)

- DOLCE in sintesi:
- Nè una ontologia, nè un insieme di metodi di ragionamento, bensì una metodologia (OntoClean)
- Ma sulla base di questa metodologia, proposta di un "top-top level" (Dolce)
- Non direttamente confrontabile con le altre ontologie viste, ma possiamo dire:

VANTAGGI:

- Connessione con i fondamenti filosofici
- Basi solide per valutare le scelte ontologiche

SVANTAGGI:

- Richiede un lavoro manuale non indifferente
- Non essendo un'ontologia, non è direttamente usabile, o, meglio, la parte usabile è molto ridotta

42

Linguaggio OWL

- Il linguaggio OWL (Ontology Web Language) è uno Standard del W3C basato su RDF (per le istanze) e RDFS (per classi e proprietà) ma che ne estende l'espressività
- Unifica tre aspetti importanti:
 - Semantica formale e supporto al ragionamento efficiente (Description Logic)
 - Ricche primitive di modellazione (Frame)
 - Proposte standard per notazioni sintattiche (W3C, comunità Web)

43

Linguaggio OWL

Principali caratteristiche di OWL

- Classi
 - subclassOf, intersectionOf, unionOf, complementOf, enumeration, equivalence, disjoint
- Proprietà
 - symmetric, transitive, functional, inverse Functional
 - range, domain, subPropertyOf, inverseOf, equivalentProperty
- Affermazioni sulle istanze
 - sameIndividualAs, differentFrom, AllDifferent

44

Linguaggio OWL

Classi in OWL

- E' possibile esprimere una classe come sottoclasse:

```
<owl:Class rdf:ID="Fiume">
  <rdfs:subClassOf rdf:resource="#CorsodAcqua"/>
</owl:Class>
```

- E' possibile definire una sottoclasse esprimendo delle restrizioni rispetto a un'altra classe:

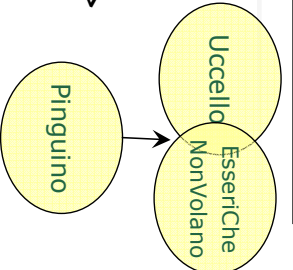
```
<rdfs:subClassOf rdf:ID="Fiume">
  <owl:Restriction>
    <owl:onProperty rdf:resource="#sfocia"/>
    <owl:allValuesFrom rdf:resource="#Mare"/>
  </owl:Restriction>
</rdfs:subClassOf>
```

45

Linguaggio OWL

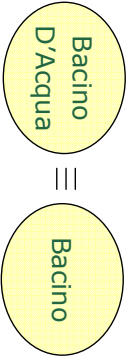
IntersectionOf:

```
<owl:Class rdf:ID="Pinguino">
  <rdfs:subClassOf>
    <owl:IntersectionOf rdf:parseType="Collection">
      <owl:Class rdf:about="#Uccello"/>
      <owl:Class rdf:about="#EsseriCheNonVolano"/>
    </owl:IntersectionOf>
  </rdfs:subClassOf>
</owl:Class>
```



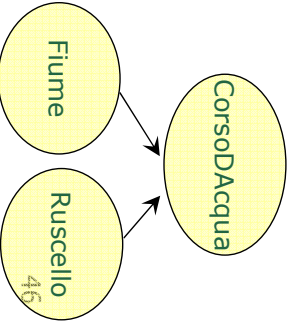
equivalentClass:

```
<owl:Class rdf:ID="BacinoD'Acqua">
  <owl:equivalentClass rdf:resource="http://www.other.org#Bacino"/>
</owl:Class>
```

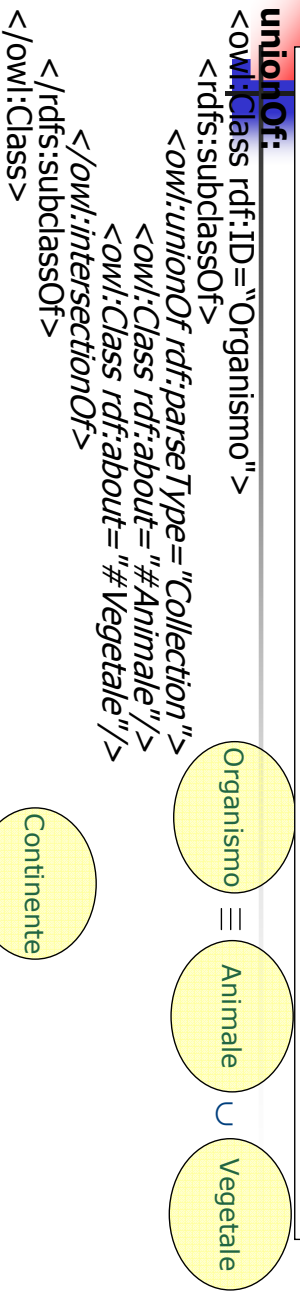


disjointWith:

```
<owl:Class rdf:ID="Fiume">
  <rdfs:subClassOf rdf:resource="#CorsodAcqua"/>
  <owl:disjointWith rdf:resource="#Ruscello"/>
  <owl:disjointWith rdf:resource="#Rivolo"/>
  <owl:disjointWith rdf:resource="#Torrente"/>
</owl:Class>
```



Linguaggio OWL



Linguaggio OWL

Due tipi di proprietà:

- owl:ObjectProperty è la classe di proprietà che hanno come valore un oggetto istanza di una classe



- owl:DatatypeProperty è la classe di proprietà che hanno come valore un dato di tipo semplice o strutturato (rdfs:Literal o XML Schema built-in datatype)



Linguaggio OWL

E' possibile restringere la cardinalità delle proprietà con:

```
owl:minCardinality
owl:maxCardinality
```

Esempio:

```
<owl:Class rdf:ID="Automobile">
  <rdfs:subClassOf rdf:resource="Veicolo"/><rdfs:subClassOf>
  <owl:Restriction>
    <owl:onProperty rdf:resource="#ha-parte"/>
    <owl:minCardinality
      rdf:datatype="xsd:nonNegativeInteger">4</owl:min
      Cardinality>
    <owl:maxCardinality
      rdf:datatype="xsd:nonNegativeInteger">4</owl:min
      Cardinality>
  </owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
```

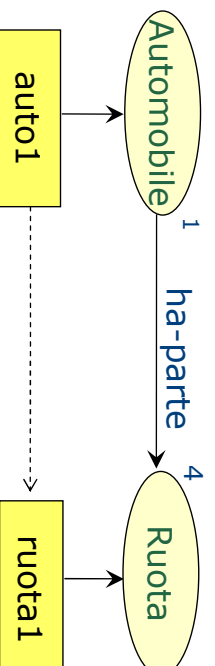
49

Linguaggio OWL

Si istanzia creando un tag con il nome della classe:

```
<Fiume rdf:ID="Tevere"/>
```

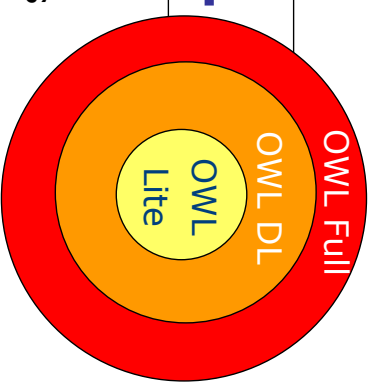
Si lega l'istanza ad altre istanze mediante le ObjectProperty definite:



```
<Ruota rdf:ID="ruota1">
<Automobile rdf:ID="auto1">
  <ha-parte rdf:resource="#ruota1"/>
</Automobile>
```

50

Linguaggio OWL



OWL Full: piena espressività

OWL DL: la stessa espressività delle Description Logics

No metaclassi (classe di una classe)

Non è possibile restringere la cardinalità di proprietà transitive

OWL Lite:

owl:minCardinality o owl:maxCardinality.

I soli valori consentiti sono per owl:cardinality sono 0 e 1.

Non sono disponibili owl:hasValue, owl:disjointWith, owl:one of, owl:complementOf, owl:unionOf.

OWL esiste in tre forme, caratterizzate da diversi gradi di complessità e - conseguentemente - di computabilità. OWL-Light è computabile (ossia è possibile trovare tutte le soluzioni in un tempo finito) ma poco espressivo; OWL-Light è poco usato, poiché esiste OWL-DL, che è ugualmente computabile ma più ricco. Infine esiste OWL-Full, che copre tutta la ricchezza della logica predicativa, ma non è computabile e non è quindi adatto al ragionamento automatico.

51

Metodologie per Ontologie

- Abbiamo visto modelli ed esempi di ontologie, e linguaggi per la loro creazione
- ma come modelli e linguaggi possono essere utilizzati efficacemente per la creazione di ontologie?
- ...overo, quali sono le **metodologie di creazione** per le ontologie?



Costruire un'ontologia

1. Da zero
2. Ri-ingegnerizzando ontologie esistenti
3. Integrando ontologie esistenti

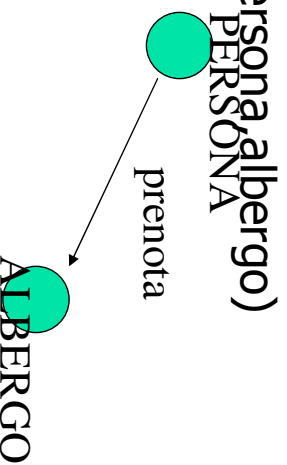
53



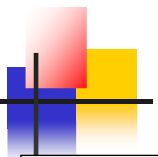
Costruire un'ontologia

Se dobbiamo costruire un'ontologia da zero, occorre:

- Identificare gli scopi
- Identificare i “termini” rilevanti (albergo, prenotazione)
- Distinguere concetti e relazioni fra i termini usati per denotare entrambi
 - Es: prenota (persona albergo)
- Codificare l'ontologia



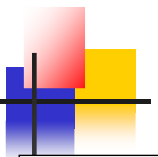
54



Costruire un'ontologia da zero

- Utilizzare risorse semi-formali disponibili (glossari, tesauri, *data-document warehouses*)
- Usare strumenti di *consensus building* e collaborative working
- Integrare competenze diverse (lessicografi, *knowledge engineers*, esperti di dominio, utenti dell'applicazione)

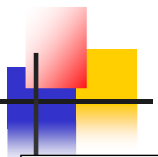
55



Costruire un'ontologia

- Costruire un'ontologia interamente a mano è faticoso e time-consuming
- Poche ontologie contengono più di qualche centinaio di concetti
- Sforzi soprattutto per la definizione di core ontologies
- Strumenti di apprendimento automatico e NLP (Natural Language Processing) per popolare automaticamente core ontologies

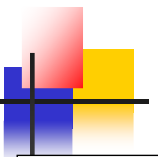
56



Costruire un'ontologia

- In genere si utilizzano metodi basati su NLP e machine learning; alcuni di essi sono:
 - Cercare nei testi "patterns" sintattici che sussumano relazioni (ad esempio, l'apposizione (es. "Shakespeare, the poet" → hypernim(N2,N1):-appositive(N2,N1)))
 - Metodi statistici per estrarre termini e "string inclusion" per derivare relazioni di iperonimia es:
 - color laser printer $\xrightarrow{\text{Kind_of}}$ laser printer)
 - Metodi di machine learning "apprendono" regole di assegnazione di relazioni semantiche fra termini utilizzando training sets di testi manualmente etichettati

57



Costruire un'ontologia

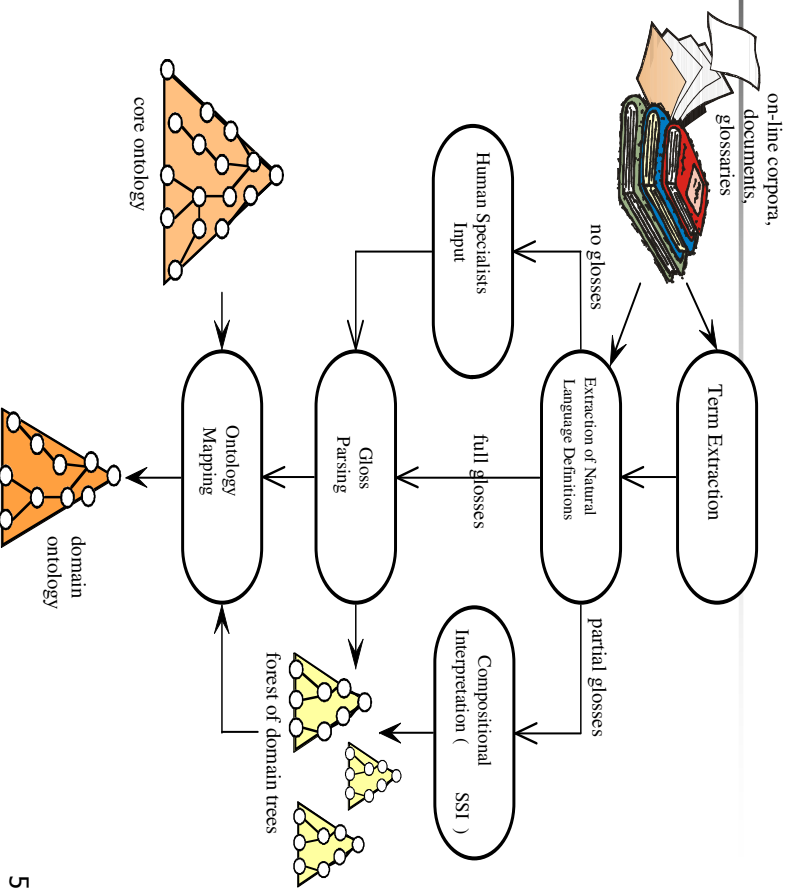
Il sistema ONTOlearn:

- Integra tecniche di machine learning, natural language processing e analisi statistica
- Utilizza vari tipi di risorse (lessici semantici, corpora di testi, glossari)
- Sperimentato in vari progetti nazionali e internazionali in vari ambiti (e-learning, interoperabilità , turismo, economia, computer networks, arte)

58

Costruire un'ontologia

Architettura
di ONTOLearn:



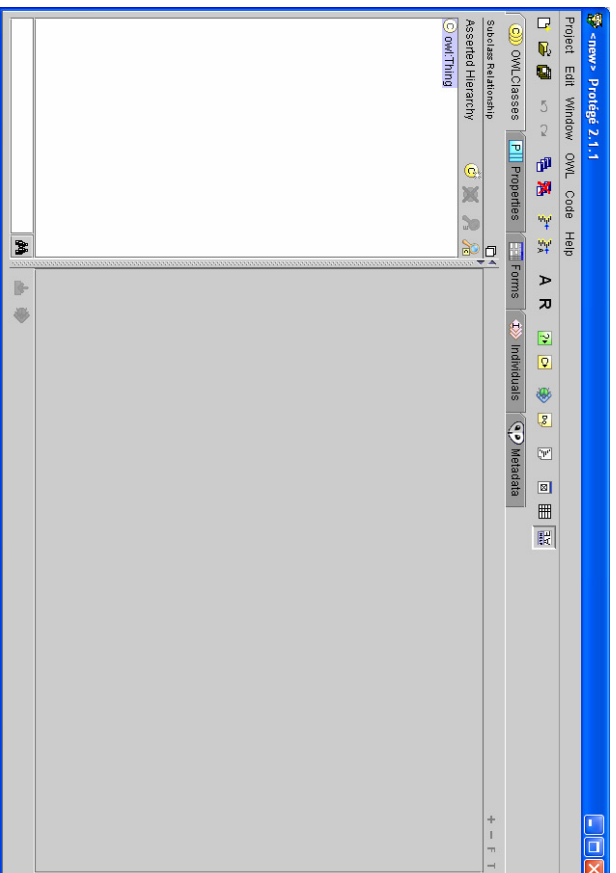
59

Costruire un'ontologia

- Oltre ONTOLearn, il cui scopo è prevalentemente quello di **popolare un'ontologia** (il compito piu' gravoso), esistono anche altri software, piu' leggeri, in grado di permettere la **creazione manuale** e la gestione di ontologie, fra questi *Protege* e *OntoEdit*

Costruire un'ontologia

- Protege: <http://protege.stanford.edu>



61

Costruire un'ontologia

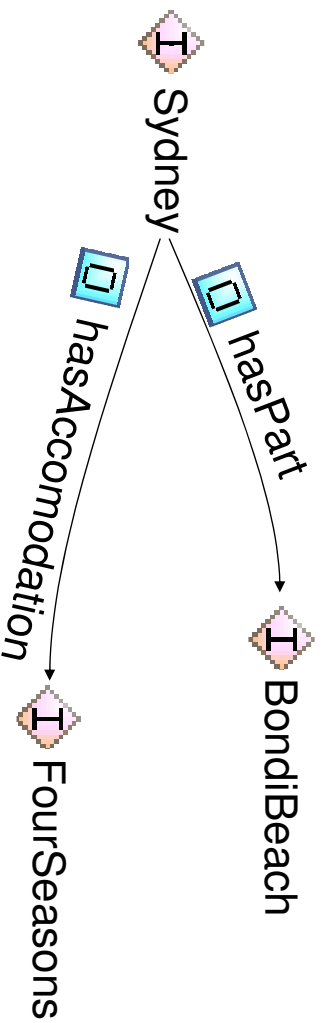
- Protege permette di lavorare con:
 - Individuals (istanze, es "fiat 500")
 - Properties
 - ObjectProperties (references)
 - DatatypeProperties (simple values)
 - Classes (concetto astratto, es "auto")

62

Costruire un'ontologia

■ Object Properties

- Collega due istanze (individuals)
- Tipi di relazioni (0..n, n..m)

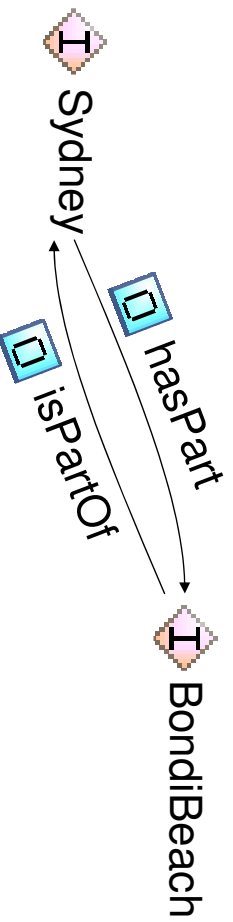


63

Costruire un'ontologia

■ Inverse Properties

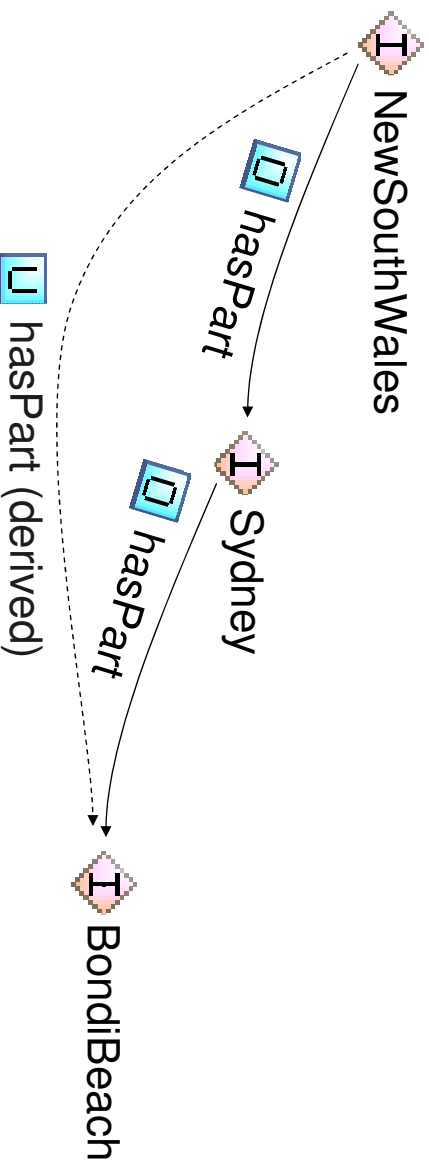
- Rappresentano relazioni bidirezionali
- Aggiungere un valore ad una proprietà implica aggiungerlo alla sua inversa



64

Costruire un'ontologia

■ Proprietà transitive




65

Costruire un'ontologia

■ DatatypeProperties

- Collega individui a valori primitivi (integers, floats, strings, booleans ecc)

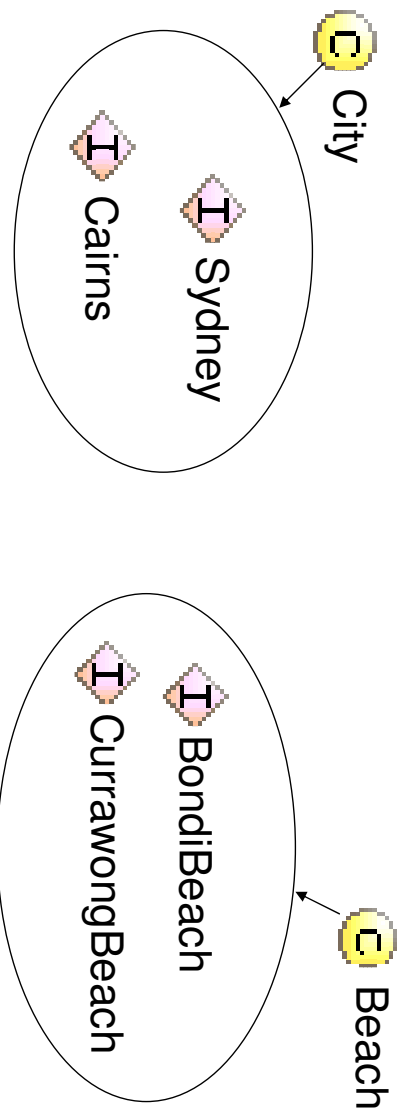
 Sydney
hasSize = 4,500,000
isCapital = true
rdfs:comment = "Don't miss the opera house"

66

Costruire un'ontologia

■ Classi

- Gruppi di individui con caratteristiche comuni
- Tutti gli individui sono istanza almeno di una classe

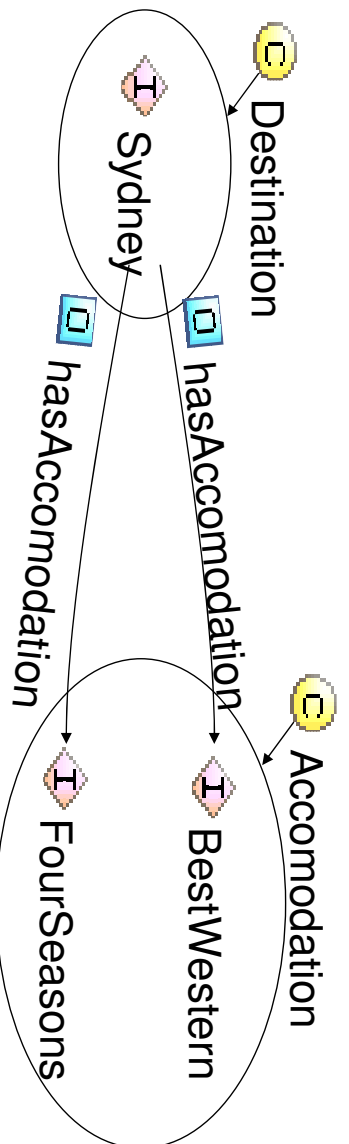


67

Costruire un'ontologia

■ Range e Domain

- Specifica di una proprietà
- Domain: "è il lato sinistro di una relazione" (Destination)
- Range: "il lato destro" (Accommodation)



68

Costruire un'ontologia

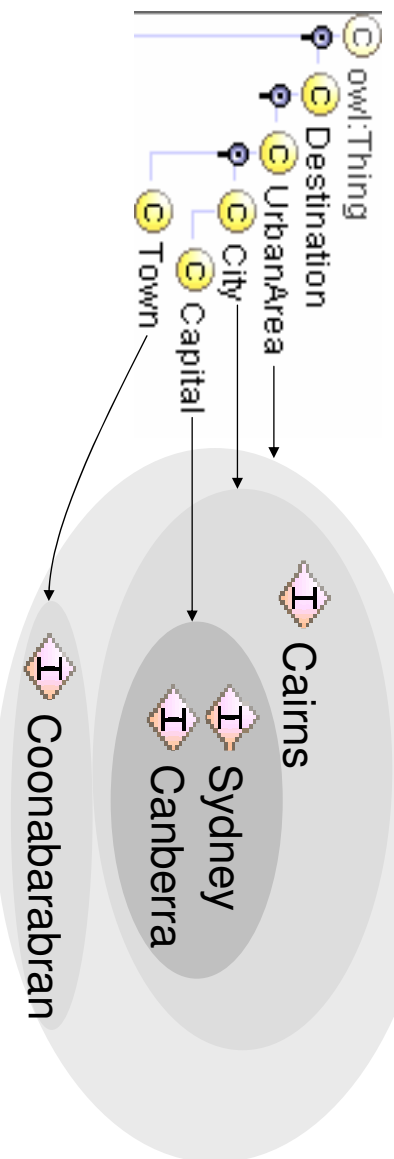
■ Dominio

- Gli individui possono assumere solo i valori di proprietà con domini corrispondenti, es: "Only Destinations can have Accomodations"
- I domini possono contenere classi multiple
- Objects and Animates have Parts
- I domini possono essere indefiniti: le proprietà possono essere usate ovunque

69

Costruire un'ontologia

- **Relazioni fra Classi e Superclassi**
- Le classi sono strutturate in gerarchie
- Le istanze dirette di classi sono anche istanze indirette di super-classi

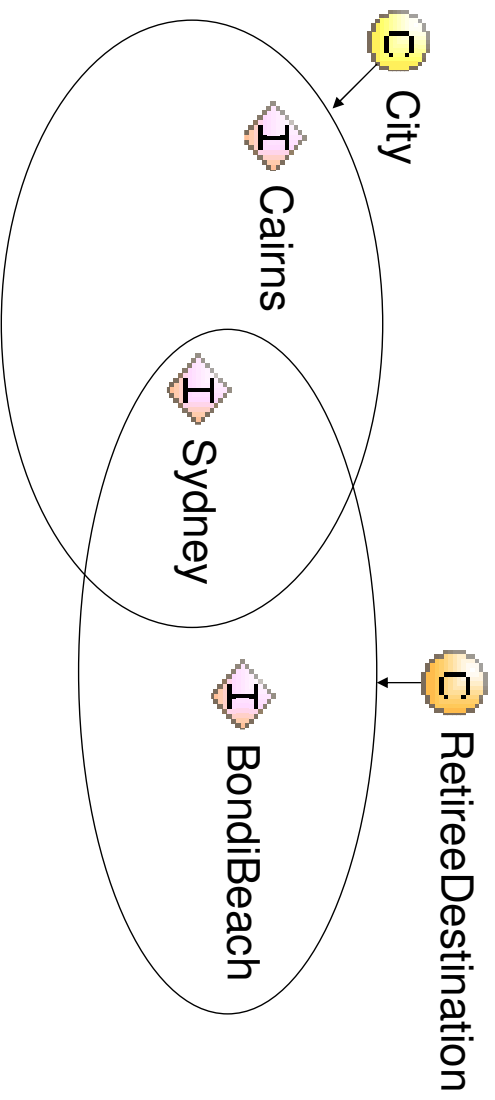


70

Costruire un'ontologia

■ Relazioni fra classi

- Le classi si sovrappongono arbitrariamente

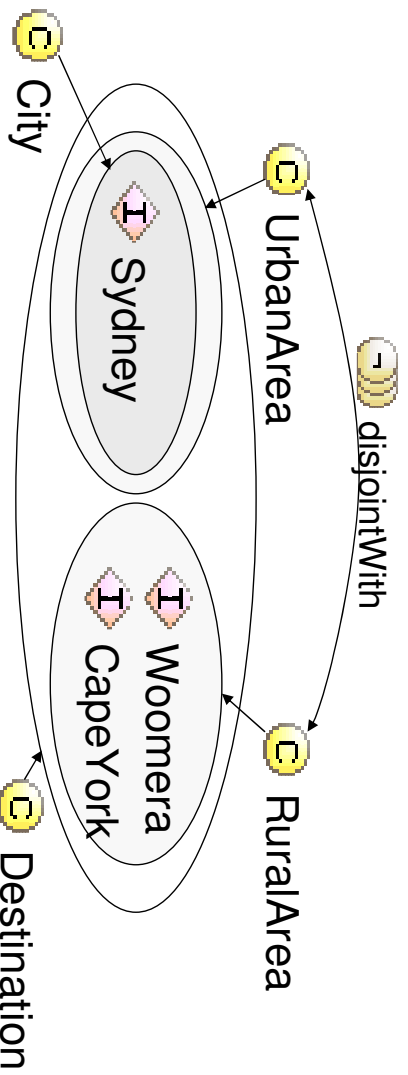


71

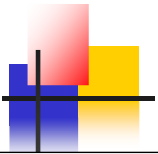
Costruire un'ontologia

■ Disgiunzione fra classi

- Le classi possono sovrapporsi
- Ma si può fare che non abbiano istanze in comune



72



Costruire un'ontologia

OntoEdit

<http://www.ontoprise.de/products/ontoeedit>

- Ambiente grafico per editing di ontologie
- Architettura estensibile per aggiungere plug-in
- Ontologia viene memorizzata in un database relazionale
- Implementata in XML, Flogic, RDF(S) DAML+OIL

73



Costruire un'ontologia

- **Altri tool di ontology management**
 - Chimaera
 - OILED
 - Apollo
 - MOMIS
 - SymOntoX

74

Utilizzo di Ontologie

- Abbiamo visto modelli ed esempi di ontologie, linguaggi per la loro creazione e come modelli e linguaggi possano essere utilizzati per la creazione e gestione di ontologie
- Le ontologie trovano numerosi **campi di applicazione**; nel seguito si presentano diversi esempi, in particolare l'ultimo relativo al contesto del semantic web, ovvero **l'annotazione semantica**...

75

Utilizzo di Ontologie

■ Un caso d'uso: classificazione di documenti

- Supponiamo che ad un documento storico digitalizzato sia associato il seguente descrittore: "Discorso di Raffaello Lambruschini in occasione dell'adunanza tenuta in onore di Vincenzo Gioberti".
- Un sistema di information retrieval "classico" può estrarre le 4 parole chiave: discorso, Raffaello Lambruschini , adunanza, Vincenzo Gioberti, e con queste indicizzare il documento.
- Sinonimia: Ad esempio, la parola adunanza ha, nella lingua italiana, i seguenti sinonimi : riunione, adunata, incontro, raduno, assemblea, consiglio, convegno, simposio, compagnia, folla, assembramento, e molti (se non tutti) fra questi termini avrebbero potuto essere usati dall'autore della descrizione in esame.
- Una annotazione del tipo "meeting, get toghether" (il nome dell'annotazione è tratto dal lessico semantico on-line WordNet 1.7) potrebbe riconoscere l'equivalenza di tutti questi termini.

76

Utilizzo di Ontologie

SCHEDA DI ORIGINE

REF 820

TITOLO *Discorso di Raffaello Lambruschini in occasione dell'adunanza tenuta in onore di Vincenzo Gioberti*

DESCRIZIONE / Raffaello Lambruschini. - [documento manoscritto]. -

5 luglio 1853. - 8 c.

AUTORE Lambruschini, Raffaello

INVENTARIO 58671

COLLOCAZIONE Busta 81.1394

NOTA Pubbl. in Atti, C. 31, 1853, p. 344.

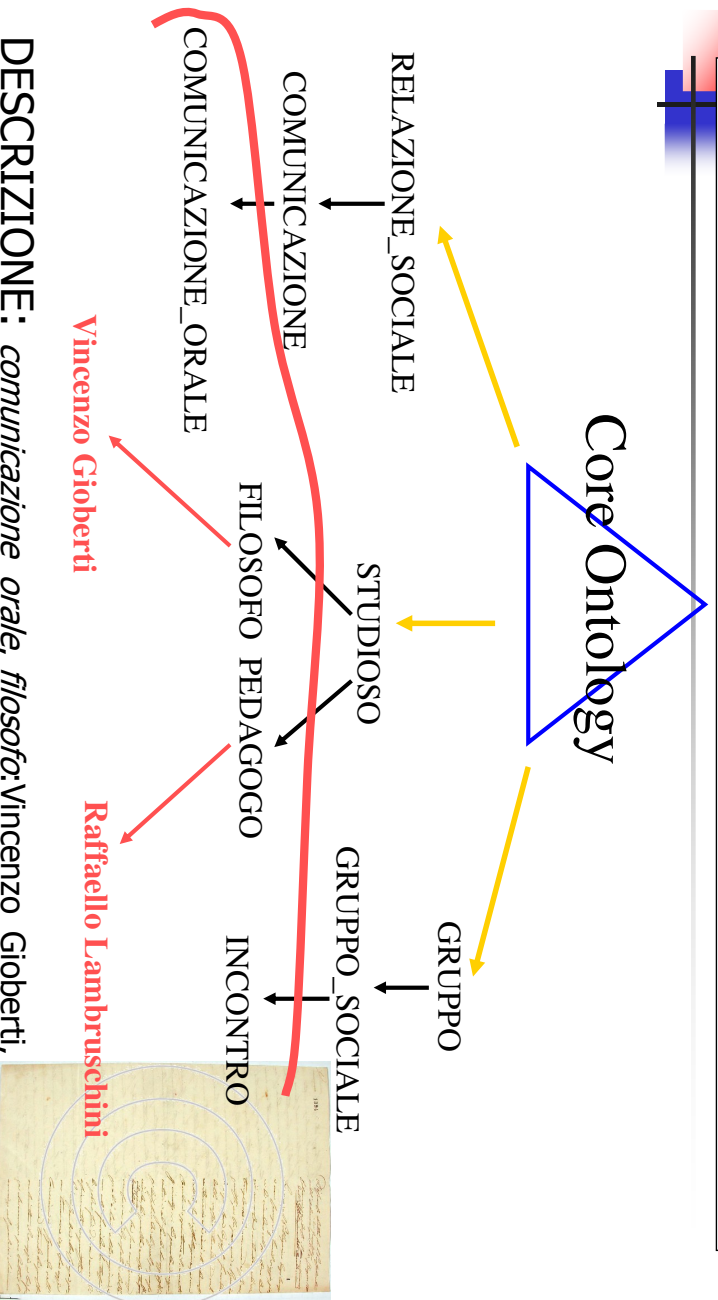
KEYWORDS Accademia dei Georgofili

Annotazioni o metadati



77

Utilizzo di Ontologie

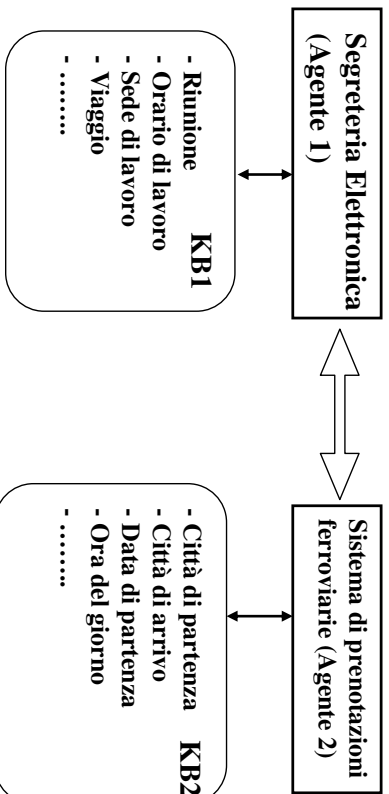


DESCRIZIONE: *comunicazione_orale, filosofo: Vincenzo Gioberti, pedagogo: Raffaello Lambruschini, incontro*

78

Utilizzo di Ontologie

Esempio 2: Interoperabilità fra servizi web



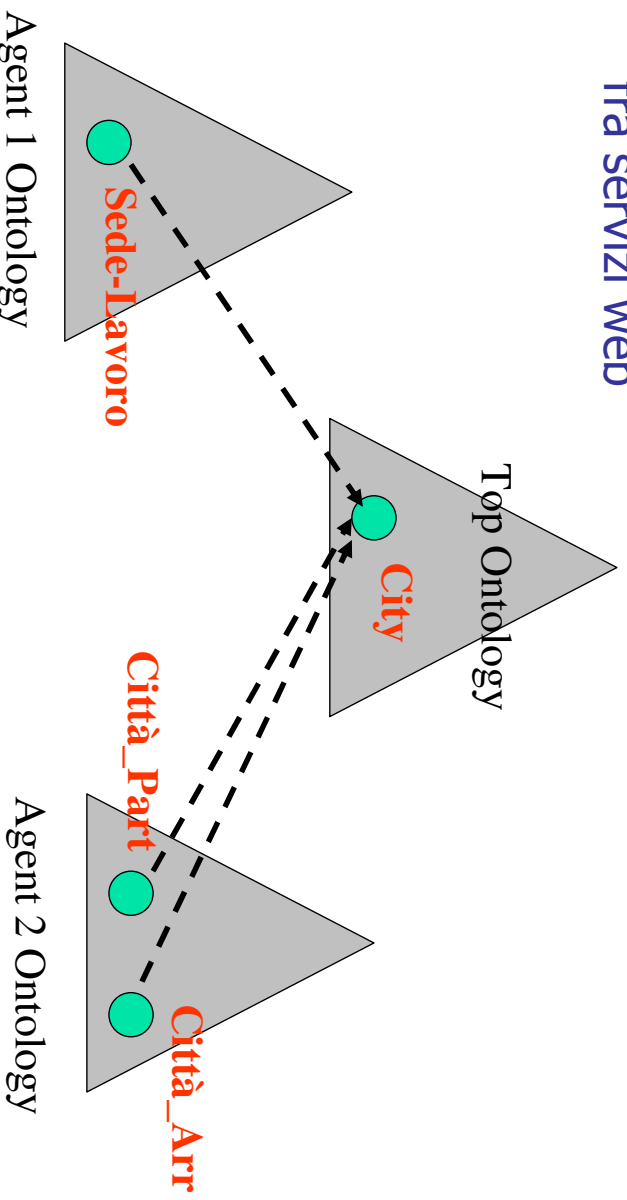
ma:

cittaDiPartenza := sedeDiLavoro;
cittaDiArrivo := luogoDelMeeting;

**E se cittaDiPartenza si chiama
CittaPart o DepartureCity?**

Utilizzo di Ontologie

- Le ontologie possono garantire l'interoperabilità fra servizi web



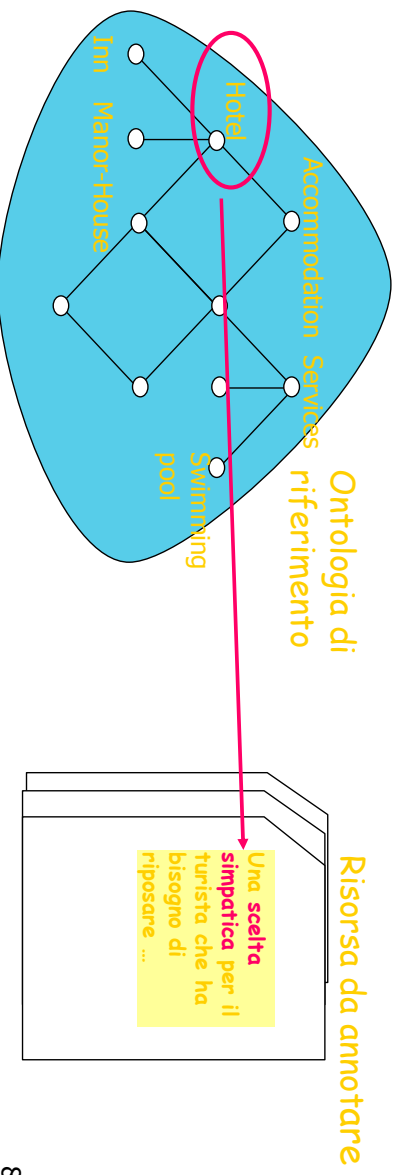
Utilizzo di Ontologie

- Un altro caso di utilizzo di Ontologie e' nel campo dell'**E-Learning**
- E' un "servizio" quindi in realtà rientra nel caso dell'esempio precedente, però:
- Navigazione "semantica" dei corsi
- Classificazione materiale didattico
- Reperimento di materiale didattico (ricerca per concetti)

81

Annotazione semantica

- Uno degli esempi piu' significativi di utilizzo di ontologie era nell'ambito del web semantico, in cui esse hanno lo scopo di permettere l'**annotazione semantica delle risorse web**, in seguito analizzata nel dettaglio...



82

Annotazione semantica

Criteri per la classificazione delle annotazioni:

- Tipo di risorsa annotata (Documenti / webServices / schemi di dati / processi)
- Destinatario dell'annotazione (Human User / Computer)
- Posizionamento dell'annotazione (Attached / Embedded)
- Livello di formalità del linguaggio usato (formale / informale)
- Livello di restrizione terminologica imposta sul linguaggio (absence, advised, mandatory)
- Annotazione attraverso: istanze / concetti

83

Annotazione semantica

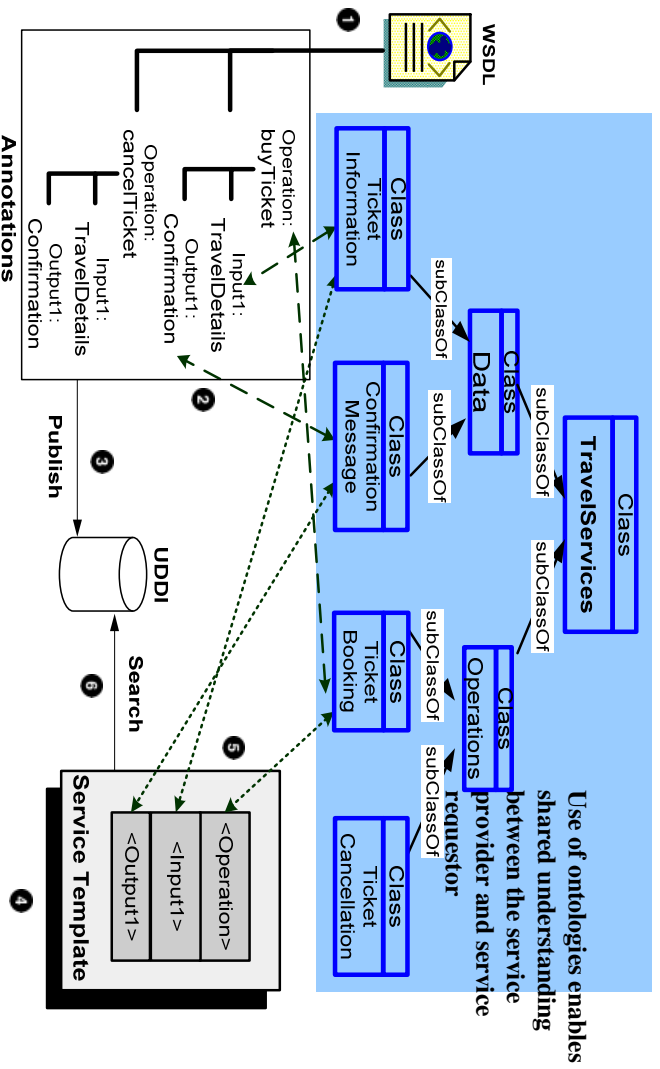
1. Tipo di risorsa annotata: cosa annotare?

- Annotazione di **documenti**
 - Frammento di testo, pagina HTML, immagini, ecc.
- Annotazione di **Web Services**
 - I servizi web sono associati a descrizioni in un linguaggio formale (es. WSDL) dei parametri di input e output necessari ad usufruire di un certo servizio (es. prenotazione on-line). Queste descrizioni possono essere annotate.
- Annotazione di **schemi di dati, processi,...**
 - E' possibile annotare dati e processi coinvolti nello *scambio di informazioni tra applicazioni software che cooperano*

84

Annotazione semantica

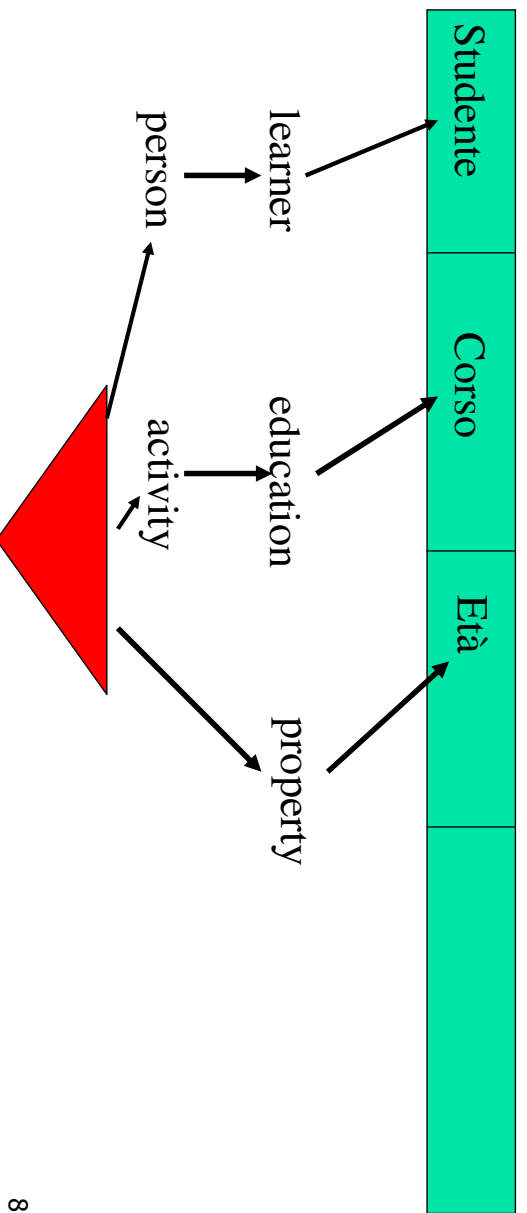
Annotazione di un Web Service:



85

Annotazione semantica

Annotazione di dati



86

Annotazione semantica

2. Destinatario dell'annotazione:

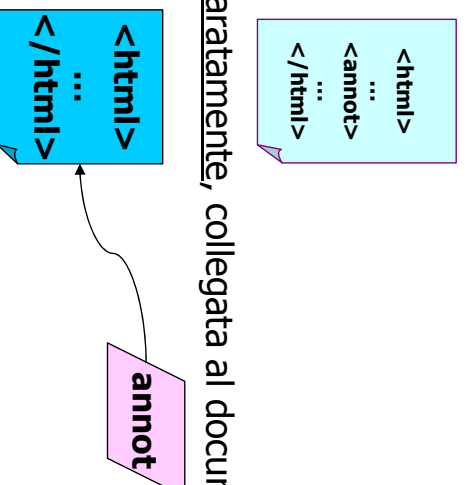
- *Utente umano*
 - L'annotazione ha lo scopo di fornire **informazioni aggiuntive** su una risorsa
 - in questo caso l'annotazione in genere sarà **poco formale** probabilmente realizzata in linguaggio naturale
- *Computer*
 - Ha lo scopo di **esplicitare il significato** di una risorsa
 - in tal caso l'annotazione deve essere di tipo **formale** per poi essere processabile da una macchina

87

Annotazione semantica

3. Posizionamento dell'annotazione

- *Embedded*
 - inserita all'interno dello stesso documento annotato
- *Attached*
 - Memorizzata separatamente, collegata al documento da un link.



88

Annotazione semantica

4. Livello di formalità del linguaggio di annotazione:

- **formale**
 - espressa in un linguaggio formale di rappresentazione della conoscenza
 - Alcuni linguaggi
 - RDF e RDFS
 - DAML, DAML+OIL, OWL
 - Permette di elaborare i dati senza l'intervento umano (*machine understandable*)
 - **informale**
 - Espressa in linguaggio naturale o in un linguaggio controllato
 - Permette di aggiungere informazioni su documenti fruibili da umani
- Esempio Annotazione Formale
- ```
<Hotel rdf:about="file:///C:\PagineHtml\alberghi.htm #Pierrot">
 <Name> Pierrot</Name>
 <is_situated rdf:resource="file:///C:\...\alberghi.htm #S.Benedetto"/>
 <has_service rdf:resource="file:///C:\...\alberghi.htm #Piscina"/>
</Hotel>
```
- Esempio Annotazione Informale in linguaggio naturale
- Descrizione caratteristiche hotel...*

Il Pierrot, posto piacevole ed accogliente per soggiornare, si trova sul lungomare di S.Benedetto. E' dotato di piscina...

89

# Annotazione semantica

## 5. Livello di restrizione imposto al linguaggio

- **Absence (nessuno)**
  - Non ci sono restrizioni sull'utilizzo del linguaggio (i termini del linguaggio possono essere usati senza vincoli)
- **Advised (consigliato)**
  - È possibile annotare utilizzando o meno i termini di un glossario, di un ontologia, parole chiavi in alternativa al linguaggio naturale.
- **Mandatory (imposto)**
  - L'annotazione può essere costituita utilizzando esclusivamente termini definiti in un glossario o un'ontologia di riferimento (Annotazione Ontology-based)

90

# Annotazione semantica

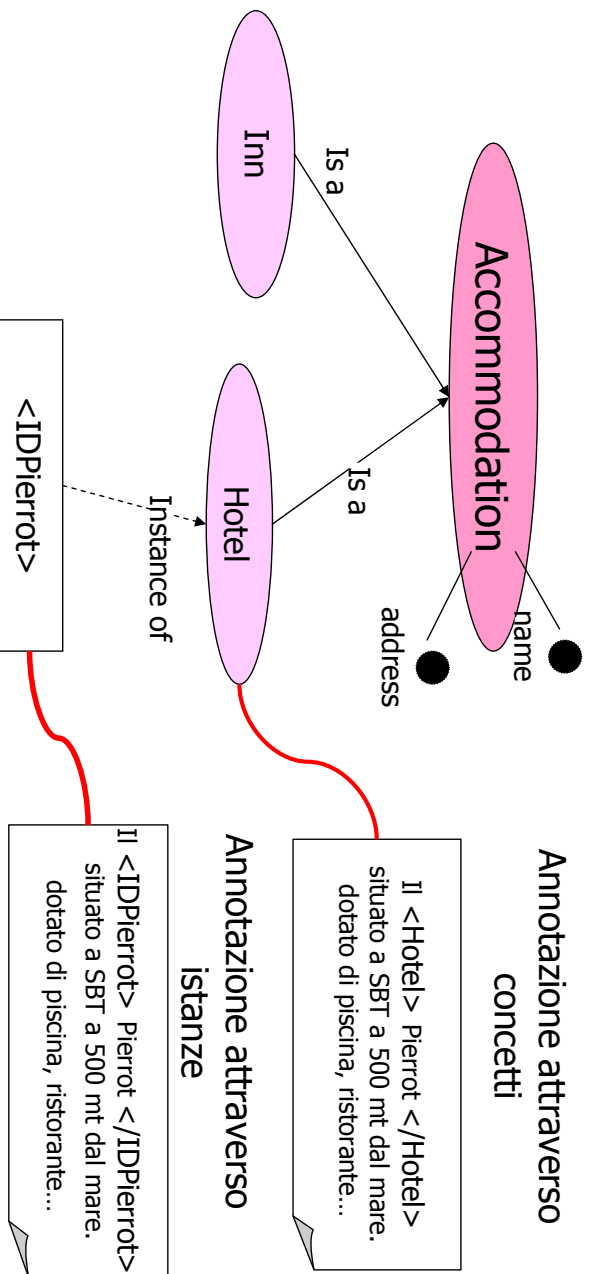
## 6.1 Modalità di Annotazione

- *Considerando solo le annotazioni **Ontology-based...***
  - Annotazione attraverso **istanze** "ontologiche"
    - L'annotazione consiste in
      - Associazione di una istanza all'elemento annotato
      - Valorizzazione delle proprietà che descrivono l'istanza
  - Annotazione attraverso **concetti** dell'ontologia
    - L'annotazione consiste in
      - Associazione di un concetto dell'ontologia all'elemento annotato
      - Associazione di una composizione di concetti dell'ontologia (attraverso opportuni operatori) all'elemento annotato
- **Istanza**: un individuo della classe descritta da un concetto (es: "Pierrot" è un'istanza della classe Hotel)
- **Concetto**: una astrazione che rappresenta le proprietà comuni di insiemi di istanze

91

# Annotazione semantica

- Esempio di annotazione su concetti o istanze:



92

# Annotazione semantica

## Annotazioni complesse

- Una risorsa (documento multimediale, servizio web, base di dati o processo) possono essere annotati semanticamente associando all'intera risorsa o a frammenti di essa un concetto dell'ontologia, oppure un'istanza
- Tuttavia sono possibili annotazioni più complesse: ad esempio, posso associare alla descrizione di un hotel una espressione formale (ad esempio usando linguaggi logici) che sia una composizione di più concetti, ad esempio:

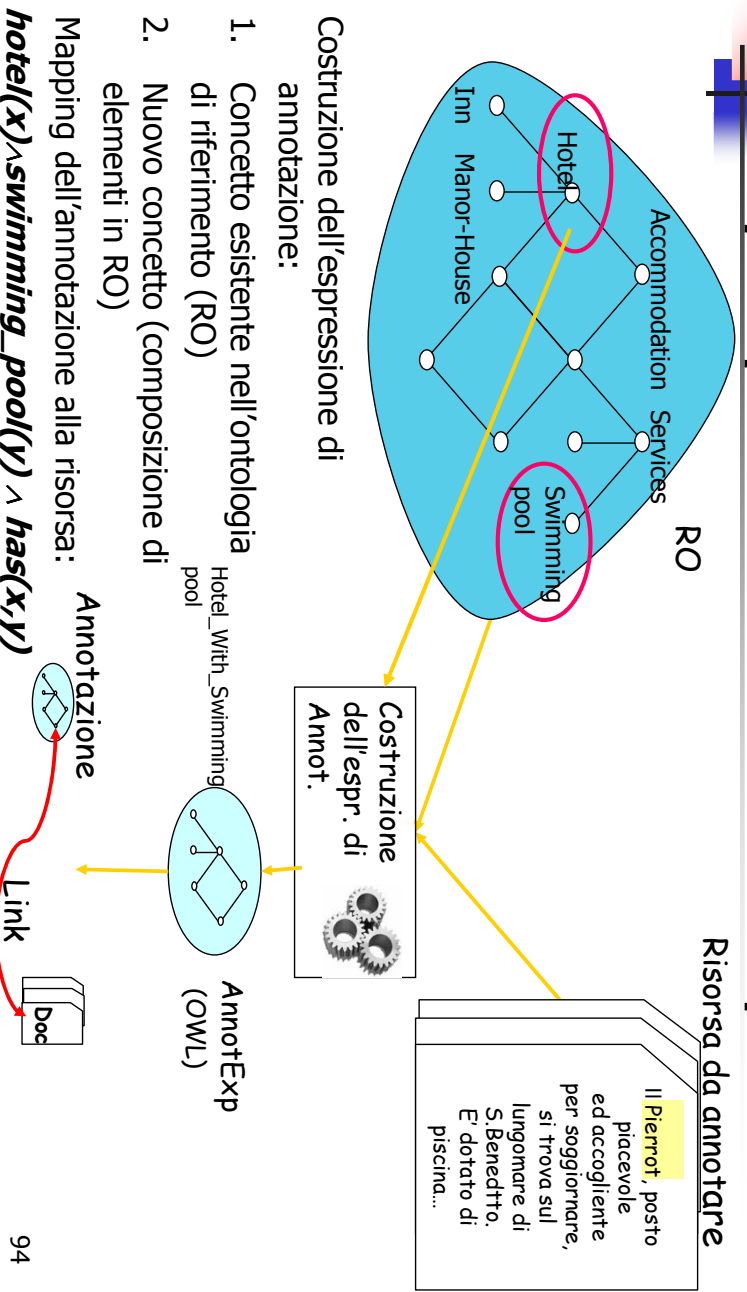
*beach*  $\wedge$  *hotel*

- In un'ontologia infatti non è opportuno generare concetti singoli per ogni possibile caratterizzazione di un concetto più generale

93

# Annotazione semantica

## Un esempio di processo di annotazione complessa



94

## Annotazione semantica

Principali tool Ontology-based, annotazione di documenti testuali

- Ontomat
- Mnm
- Smore
- Cohse
- Trellis
- Melita
- Kim
- Shoe

Altri tool

- Semantic Word (annotazione di documenti Word in Daml+oil)
- Gate (annotazione in NPL)
- Annotea (non onotogy-based)
- Yavas (non onotogy-based)
- Semantic murk-up plugin...

95

## Annotazione semantica

Strumenti di annotazione disponibili:

- 1) *per costruire ontologie e realizzare una annotazione Automatica*
- 2) *per annotazione manuale basate su Reference Ontology*

Strumenti di Annotazione per costruire ontologie:

- Si parte da archivi testuali annotati manualmente
- Algoritmi di Information Extraction (IE) e/o word sense disambiguation a partire da tali corpora di testi apprendono regole per l'estrazione di informazioni da altri documenti.
- Le regole derivate da questo processo possono essere usate per popolare un'ontologia usata poi per annotare.
- L'annotazione può, da questo momento in poi, essere automatica o semi-automatica (semi-automatica se richiede conferma o correzione da parte di un umano)

Strumenti di Annotazione Manuale basata su Reference Ontology:

- Lo scopo di tale annotazione è di "spiegare" il contenuto delle risorse rispetto ad un ontologia stabilita, eventualmente anche componendo più concetti della RO (esempio: Ontomat)

96



## Annotazione semantica

### Ontomat (Univ. of Karlsruhe, Germany)

- Applicazione java (stand-alone), Permette la visualizzazione tree-like di ontologie nel linguaggio DAML+OIL
- Ha un browser per l'esplorazione dell'ontologia e delle istanze e un browser Html che visualizza le parti di testo annotate.
- Implementato in Java, fornisce delle API d'interfaccia per l'estensione tramite plugin.
- Le annotazioni avvengono mediante drag'n'drop (si seleziona una porzione del documento e si trascina un concetto o istanza)
- Input :
  - ontologia in DAML+OIL
  - pagina html
- Output :
  - Pagina Html con annotazioni inserite nell' header
  - Istanze esportate (DAML+OIL)
  - Ontologia (DAML+OIL)
  - Ontologia + Istanze.

97

## Annotazione semantica

### Ontomat:

The screenshot shows the Ontomat application interface. On the left, there is a hierarchical ontology tree with nodes for 'Person', 'Employee', 'Faculty\_Member', 'Researcher', 'Student', 'Project', and 'Publication'. The 'Researcher' node is selected, showing a list of instances including 'Michele Misikoff'. On the right, the HTML view displays the biography of Michele Misikoff, detailing his roles at IERS, CNR, and various international organizations, along with his education and research interests.

Attributes	Values
address	
email	
fax	
first_name	Michele
homepage	
last_name	Misikoff

**Michele Misikoff**

Coordinator of IERS - Laboratory for Enterprise Knowledge and Systems, at IASI-CNR (Istituto di Analisi dei Sistemi ed Informatica del Consiglio Nazionale delle Ricerche), where he works since 1980. Previously he had positions in leading industrial software and computer companies. In parallel he maintains tight cooperations with the University of Rome "La Sapienza", Computer Science Department, and Physics Department, where he is lecturer on Knowledge Representation and Management. Furthermore, he is adjunct professor at the Master in Business Administration of LIUSS University, where he teaches a course on ERP.

Former director of the Institute of Systems and Informatics (ISI) of CNR, in Coenza. Elected as a member of the CNR National Consulting Committee for Engineering and Architecture and as the Statute Secretary of the CNR National Consulting Committee for Computer Science, where he served from 1994 to 1999.

Past President of the International Foundation for Extending Database Technology (EEDBT), he is currently member of the Executive Board.

98

## Annotazione semantica

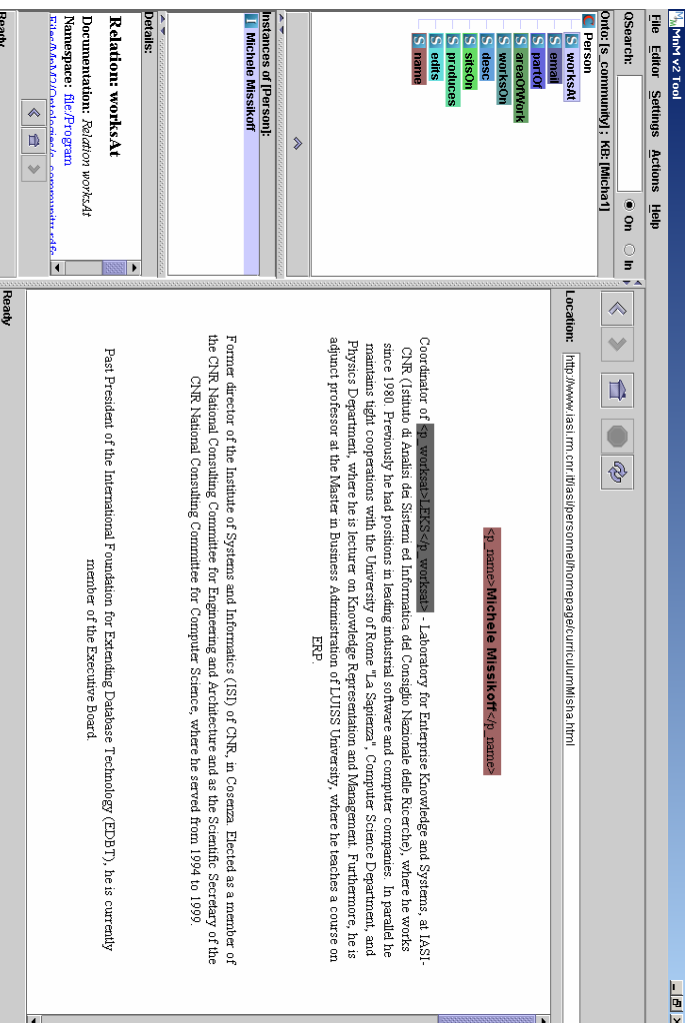
### MHM (University of Sheffield. UK)

- Applicazione java (stand-alone)
- Fornito di browser
  - Browser Html
  - Browser dell'ontologia
- Si basa su un algoritmo di apprendimento automatico (*Amilicare*)
- Annotazione semi-automatica
- Input:
  - ontologia in DAML/ DAML+OIL/ RDF
  - Pagina Html
- Output:
  - Le annotazioni prodotte sono
    - *Embedded*: in un file xml copia della pagina html annotata (utilizzata da Amilicare per IE)
    - *Attached*: un file espresso nello stesso formato dell'ontologia usata per annotare. Il collegamento con il file originale è realizzato tramite un ulteriore file espresso in un formato proprietario (con estensione “.dymmy”).

A. Longheu – Sistemi di Elaborazione delle Informazioni II

## Annotazione semantica

MHM:



## Annotazione semantica

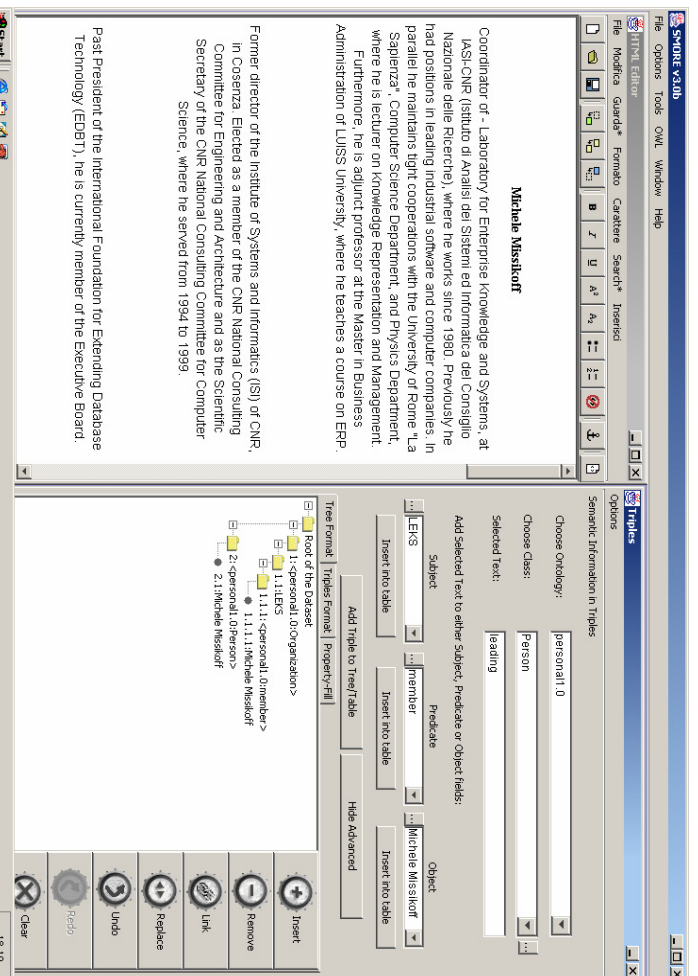
### Smore (University of Mariland)

- Applicazione java stand-alone
- Fornisce Editor pagine html e di Ontologie (fornisce un collegamento alle esistenti ontologie su Internet per renderle utilizzabili all'interno della propria web-page)
- L'utente può classificare semanticamente il proprio insieme di dati con cui annotare, cioè marcare le frasi del documento in base al modello base Soggetto-predicato-oggetto.
- Input:
  - Reference Ontology espressa in RDF, DAML, DAML+OIL, OWL
  - Documento: Pagina html
- Output
  - File dataset rappresentate l'ontologia + istanze utilizzate per annotare
  - annotazione in RDF, DAML... (è possibile salvare sia l'annotazione pura, sia il documento con annotazione "embedded" nell'header)
- Nota. E' necessaria una minima conoscenza della sintassi e dei termini del RDF da parte dell'utente.

101

## Annotazione semantica

### Smore!



102

## Annotazione semantica

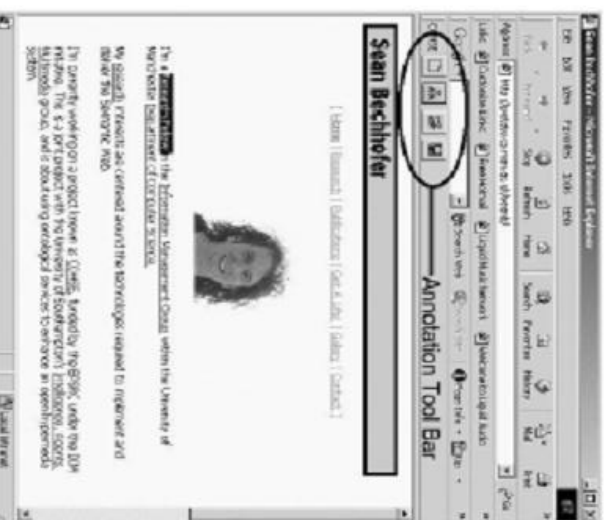
### Cohse (Univ. of Manchester UK)

- E'un plug-in di Mozilla, (barra delle applicazioni aggiuntiva)
- E' possibile navigare la RO utilizzando l'interfaccia fornita da OilEd, editor di ontologie in DAML+OIL
  - OilEd è sviluppato in Java, i componenti dell'interfaccia sono stati riusati dall'annotatore.
  - Permette di scegliere un concetto in RO o di costruirne uno per composizione
- Permette di evidenziare porzioni di testo
- Annota attraverso concetti, lega le istanze ai concetti dell'ontologia attraverso il connettivo istanceOf
- Input:
  - RO: DAML+OIL
  - documenti html
- Output:
  - annotazione formale il DAML+OIL
  - Annotazione attached utilizzando Xpointer

103

## Annotazione semantica

### Cohse (Univ. of Manchester UK)



104

## Annotazione semantica

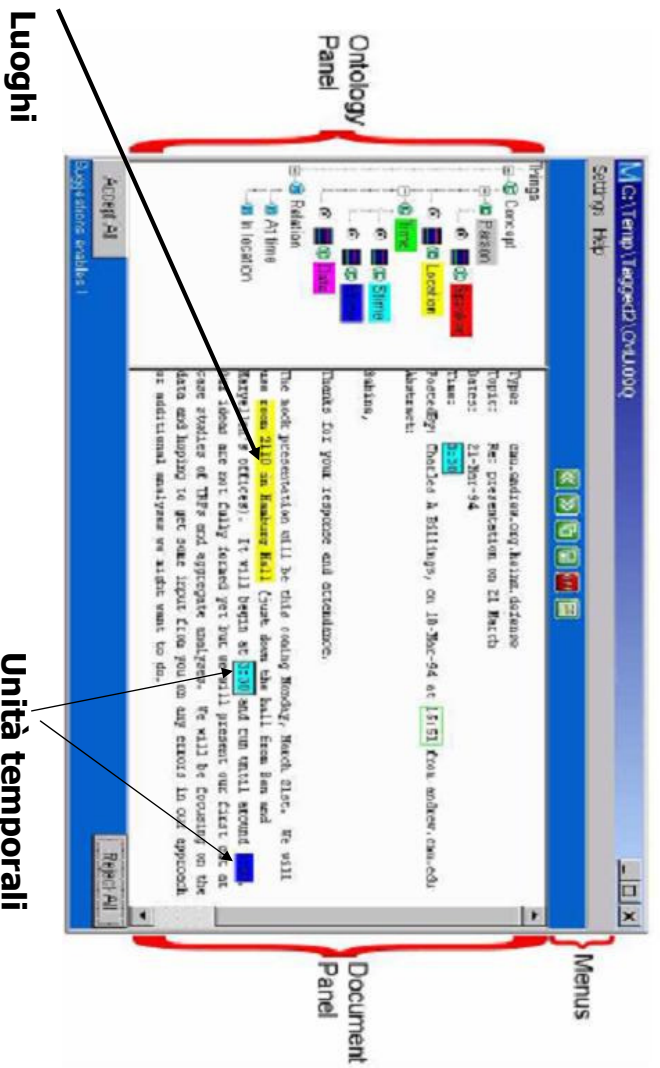
### Melita (Univ. of Sheffield, UK)

- Melita è un dimostratore di annotazioni ontology-based
- Applicazione client-server
- Apprendimento automatico (Amilcare)
- Le annotazioni sono graficamente presentate in modi differenti in base al processo che le ha definite:
  - annotazione manuale (definite da un utente)
  - annotazioni automatica (suggerita da Amilcare)
- l'utente può scegliere quale mantenere tra le due
- Input
  - RO.:sce (formato Amilcare), .ont (formato proprietario, di tipo logico)
  - Documenti testuali
- Output:
  - File Xml
  - Contenente tutte le annotazioni riguardanti un documento annotato da un utente

105

## Annotazione semantica

### Merita:



Luoghi

Unità temporali

106

## Annotazione semantica

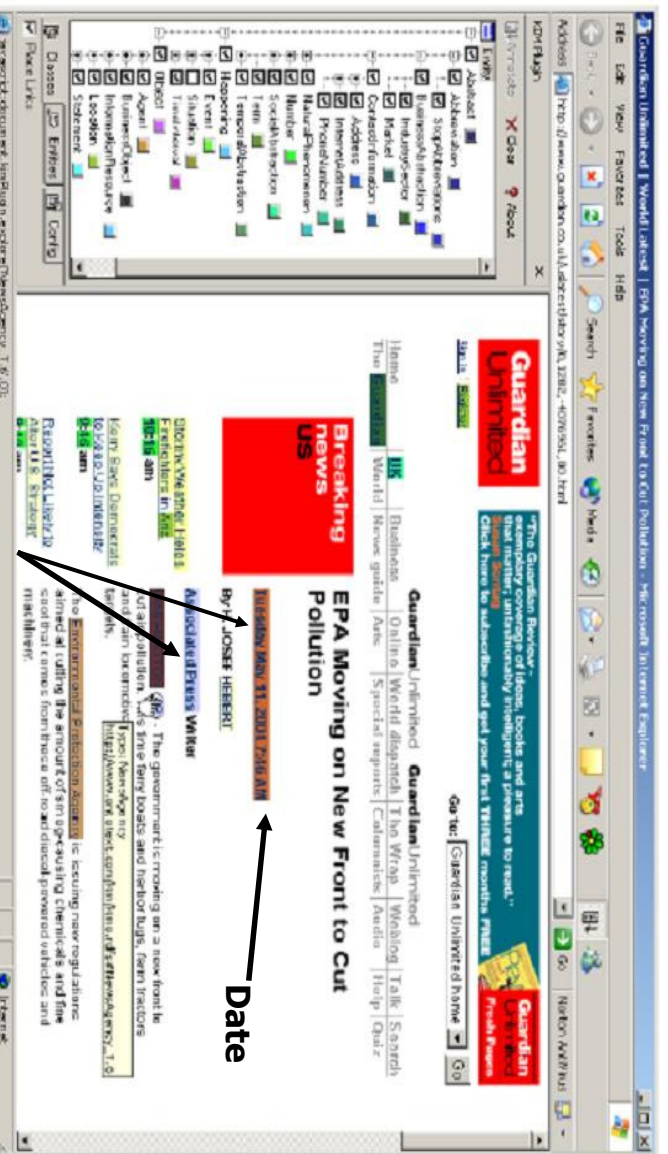
### Kim (Ontotext Lab, Sirma AI, Bulgaria)

- E' un plugin Internet Explorer.
- Prevede apprendimento automatico basato sul sistema Gate (utilizzato per la creazione di annotazioni a partire da linguaggio naturale)
- Le annotazioni sono memorizzate sul server e rese disponibili per realizzare "recupero intelligente" di informazioni
- Input:
  - RO espressa in RDF(S) OWL lite / o tassonomia
  - Documenti in formato txt, html, xml
- Output
  - l'annotazione è attached

107

## Annotazione semantica

### Kim:




Nomi propri complessi

108

## Annotazione semantica: Classificazione riassuntiva

ii

	Risorsa	Destinat.	Posizionam.	Formalità	Restriz. term.	Elem Ann.	Sviluppato presso
<b>Ontomat</b>	Htntl	Computer	Embedded (nell'header)	Formale (DAML+OIL)	Ontology-based	Istan ze	Univ. of Karlsruhe, Germany
<b>Mirtn</b>	Htntl	Computer	Embedded (file XML), Attached tramite (file propr.)	Formale (DAML+OIL, RDF)	Ontology-based	Istan ze	KMI The open Un. - Dept. of CS Univ. of Sheffield. UK
<b>Smore</b>	Foto, mail, htntl	Computer	Embedded (header) / (e-mail, Img.)	Formale (RDF, DAML+OIL, OWL)	Advised	---	University of Mariland
<b>Cohse</b>	Doc/ htntl	Computer	Attached (Xpointer)	Formale DAML+OIL	Ontology-based	Conc etti	Depart. of CS Univ. of Manchester UK
<b>Trellis</b>		Computer	Attached	Formale (OWL)	Ontology-based	---	USC Information Science Institute
<b>Melita</b>	Htntl	Computer	attached (file XML)	Formale (formato proprietario)	Ontology-based (.ont .sce)	Istan ze	Depart. of CS Univ. of Sheffield. UK
<b>Kim</b>	Txt,Htntl, xml	Computer	Attached (inserte in RO)		Advised RO oppure tassonomia	---	Ontotext Lab, Sirmia AI, Bulgaria
<b>Annotea</b>	Htntl/xml	Computer Human user	Attached (Xpointer)	Informale: formatati RDF/XML	Advised	???	W3C INRIA Rhône-Alpes W3C MIT/LCS

109