Possibili Approfondimenti (vedi slides)

- Ontologie e Web semantico
- Sistemi Esperti (area applicativa, presso DEIS varie applicazioni).

- Avete tutti gli strumenti!

Cosa sono le ontologie

- Filosofia/Computer ScienceAI: area dell'intelligenza artificiale che studia i metodi per rappresentare correttamente l'universo che ci circonda.

Perchè servono in CS?

- Condivisione di conoscenza: per non duplicare sforzi nello sviluppo di sistemi software
- Comunicazione: sia tra agenti software (tra di loro) che tra agenti software e esseri umani

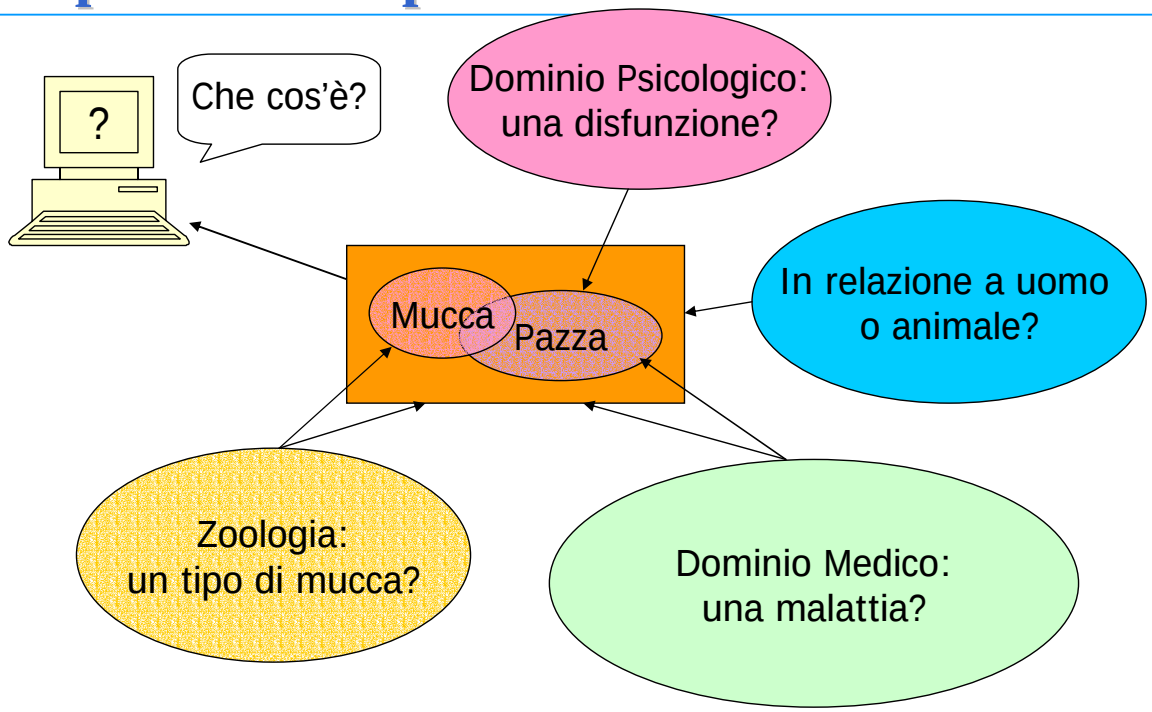
Semantic Web!

Ontologie e Web Semantico

- Possibilità di accesso e acquisizione della conoscenza tramite www
- Necessità di organizzare, integrare e interrogare basi di conoscenza
- Necessità di sorgenti di conoscenza facilmente accessibili da macchine e processi automatici
- Necessità di una conoscenza riutilizzabile e condivisibile (in contesti e forme differenti)

- Riutilizzo dei concetti di Logica, Reti Semantiche, Linguaggi terminologici, Frames con sintassi Web (XML, RDF).

Esempio: “mucca pazza”



Problemi di fondo (II)

1. Occorre eliminare la confusione terminologica e concettuale ed individuare le *entità* cui un pacchetto di conoscenza si riferisce.
2. Organizzare e rendere esplicito il *significato referenziale* permette di *comprendere* l'informazione.
3. Condividere questa *comprensione* facilita il recupero e il riutilizzo della conoscenza tra agenti e in contesti diversi.

ONTOLOGIE

Ontologia

Definizione formale di un dominio di conoscenza

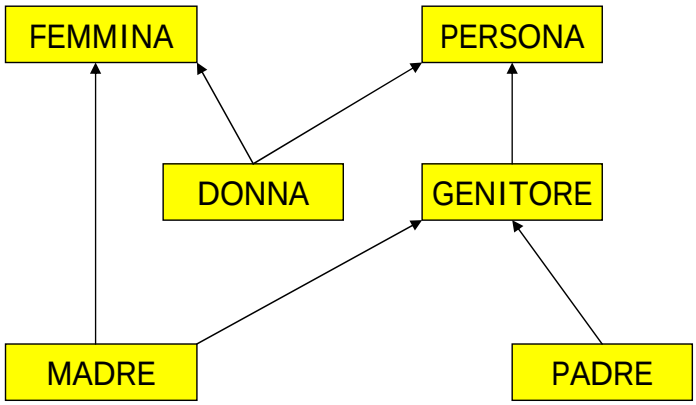
Isolare una parte del mondo e i suoi concetti fondamentali

Enumerare e definire (in modo più o meno formale) i concetti e le relazioni che tra essi sussistono:
→ classi, proprietà, assiomi, individui

Una descrizione strutturata gerarchicamente dei concetti importanti e delle loro proprietà che trovi il consenso di diversi attori interessati a condividerla e utilizzarla.

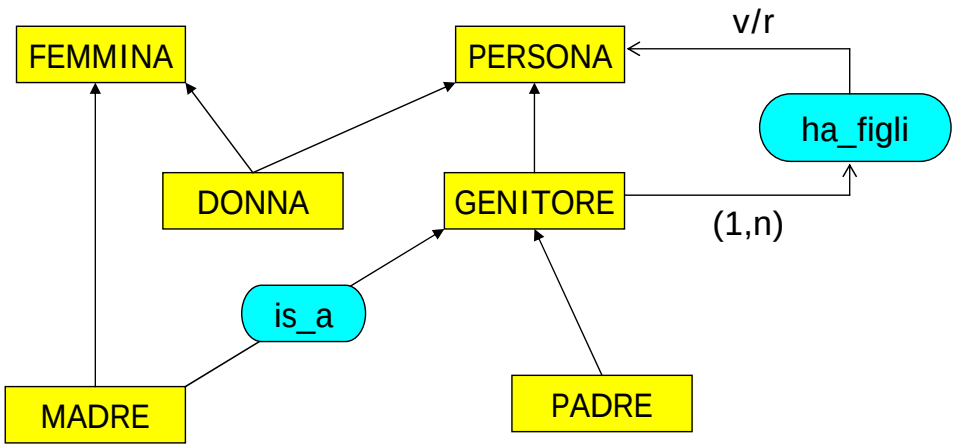
Elementi di un'ontologia: classi

Generalmente organizzati in tassonomie. La "classe" viene utilizzata in senso lato per descrivere i concetti (le classi di individui) che si vuole rappresentare (es. PERSONA, GENITORE, MADRE…)



Elementi di un'ontologia: proprietà e attributi (I)

Rappresentano le relazioni esistenti tra le classi del dominio
(es. "le MADRI *sono* GENITORI", "i GENITORI *hanno figli* che *sono* PERSONE"…)



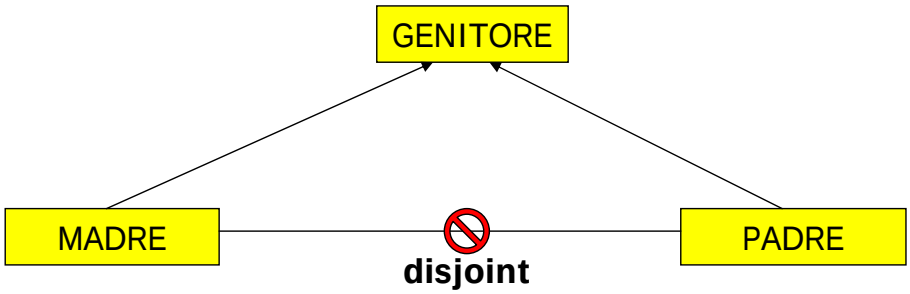
Elementi di un'ontologia: Proprietà e attributi (II)

- Il legame tra MADRE e GENITORE (**is_a**) indica che "le MADRI *sono* GENITORI" e definisce una *gerarchia* tra concetti, provvedendo una base per l'*eredità* di proprietà: un concetto specifico eredita le proprietà del concetto più generale che lo sussume.
- E' possibile rappresentare anche proprietà più complesse della relazione **is_a**.
- Ad esempio, è possibile definire la proprietà **ha_figli,** che connette le due classi GENITORE e PERSONA, specificando degli *attributi* che ne vincolano l'applicazione: v/r denota una restrizione sulle classi che possono soddisfare la proprietà, mentre (1,n) rappresenta una restrizione di cardinalità.
- L'esempio può essere letto come "un GENITORE **è** una PERSONA che **ha almeno 1 figlio** e **tutti** i figli che ha sono PERSONE".

Elementi di un'ontologia: assiomi

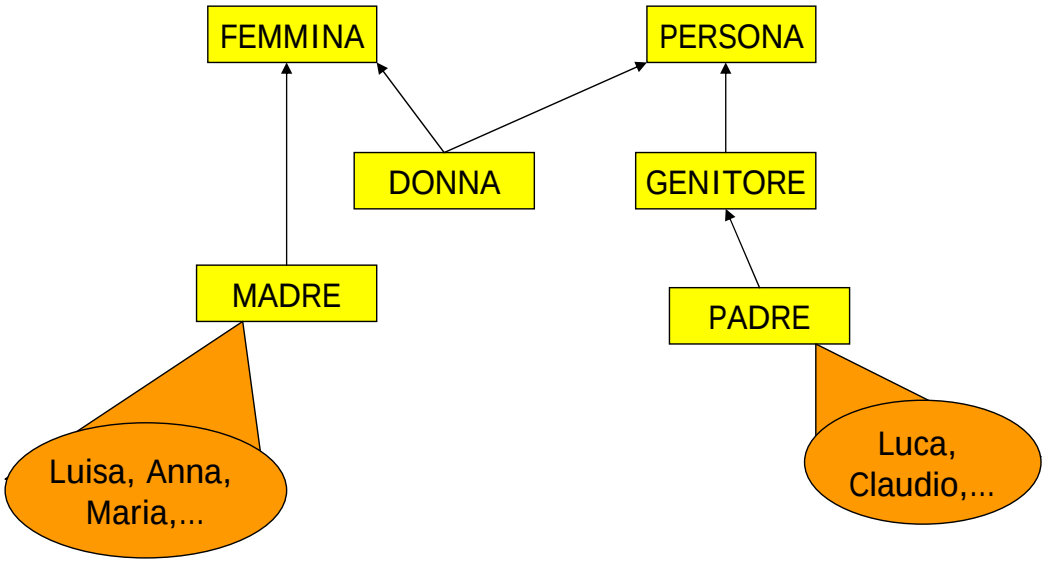
- Vengono utilizzati per modellare in maniera esplicita espressioni in ogni caso vere.
- Possono essere utilizzati per diversi scopi: definire il significato dei vari componenti dell'ontologia, definire relazioni complesse, verificare la correttezza dell'informazione specificata o dedurre nuova informazione.

(es. *disjoint* (MADRE, PADRE) esprime il fatto che un elemento della classe PADRE non può mai essere anche un elemento della classe MADRE)



Elementi di un'ontologia: individui

Sono i singoli oggetti contenuti in una classe, a vari livelli di generalità, a seconda dello scopo dell'ontologia



Linguaggio

Una volta selezionati i componenti dell'ontologia occorre definirli in maniera esplicita. E' possibile distinguere ontologie diverse sulla base del linguaggio utilizzato per definirne i componenti:

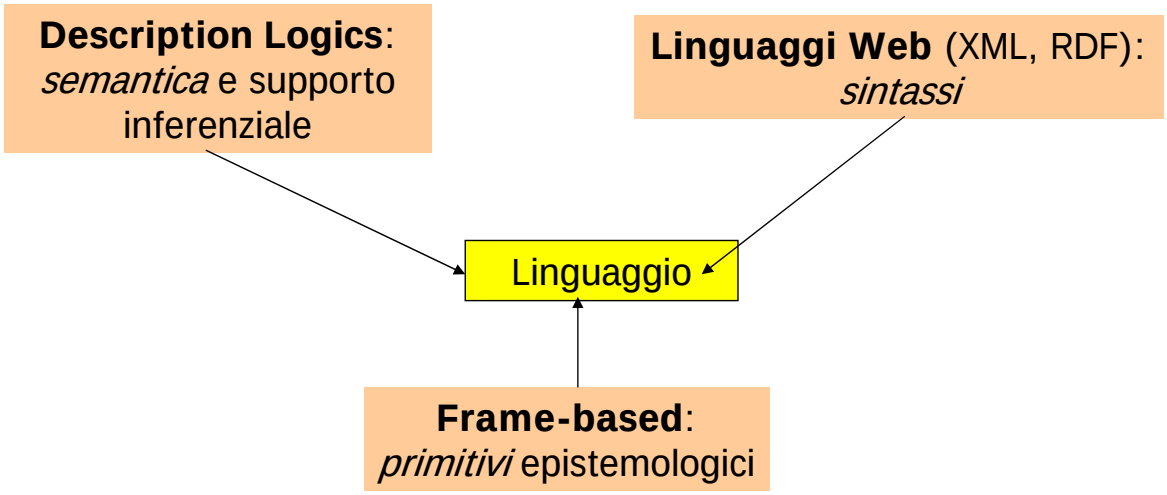
- **Informali**: espresse tramite linguaggio naturale
- **Semi-formali**: espresse tramite linguaggio artificiale definito formalmente
- **Formali**: espresse in un linguaggio dotato di semantica formale, teoremi e proprietà quali validità e completezza.

Scelta del linguaggio

- Necessità di un linguaggio comune per lo sviluppo delle ontologie (rende più semplice la condivisione e il riuso di ontologie tra progetti differenti)
- Povertà dei linguaggi di rappresentazione largamente utilizzati, quali XML e RDF.
- Necessità di linguaggi espressivi e rigorosi ma di facile utilizzo

Linguaggio di Rappresentazione

Fonde tre caratteristiche ereditate da linguaggi suoi progenitori che definiscono gli aspetti fondamentali di un linguaggio di rappresentazione:



Rappresentazione frame-based

- Primitive di rappresentazione basate su classi (frame) dotate di proprietà dette attributi (slot): è possibile definire una classe come una collezione di superclassi o attributi. Es:

Madre italiana con molti figli	
superclasse	madre
ha_figli	≥ 3 figli
vive_in	Italia

- Questo tipo di rappresentazione manca però di una semantica rigorosamente definita: problemi computazionali nell'utilizzo della rappresentazione frame-based.

Una madre italiana con molti figli è una madre con almeno 3 figli che vive in Italia.

Description logics

- Permette di definire le classi in termini di descrizioni che specificano le proprietà degli oggetti che appartengono ad una determinata classe.
- Fornisce gli operatori per definire le proprietà: congiunzione, disgiunzione, negazione, quantificazione…
- Semantica formale
- Supporto inferenza

} Frammento decidibile di logica del prim'ordine

Es: Madre italiana con molti figli ≡ madre ∩ ≥ 3 ha_figli ∩ vive_in.Italia

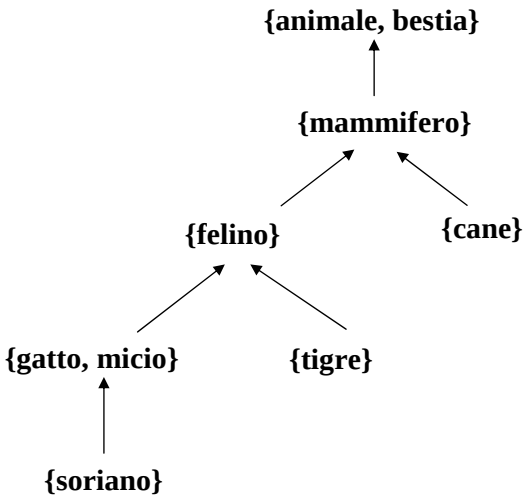
Linguaggi Web

- Un linguaggio per la rappresentazione di ontologie necessita, oltre a primitivi epistemologici e semantica, una sintassi concreta e un formato che ne permetta lo scambio e la lettura da parte di una macchina: sintassi XML e RDF.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
  <!DOCTYPE rdf:RDF (View Source for full doctype...)>
- <rdf:RDF xmlns:rdf=http://www.w3.org/1999/02/22-rdf-
syntax-ns#
   xmlns:a="http://www.w3.org/2000/01/rdf-schema#">
  <rdf:Property rdf:about="madre#ha_figli" />
  <a:Class rdf:about="madre#madre" />
- <a:Class
rdf:about="madre#madre+italiana+con+molti+figli">
  <a:subClassOf rdf:resource="madre#madre" />
  </a:Class>
  <a:Class rdf:about="madre#persona" />
  <rdf:Property rdf:about="madre#vive_in" />
  </rdf:RDF>
```


Esempio di Ontologia: WordNet

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



Cyc

- **Il progetto Cyc (da enCYClopedia) nasce nel 1984 ed è ancora in corso (si veda il sito http://www.opencyc.org/).**

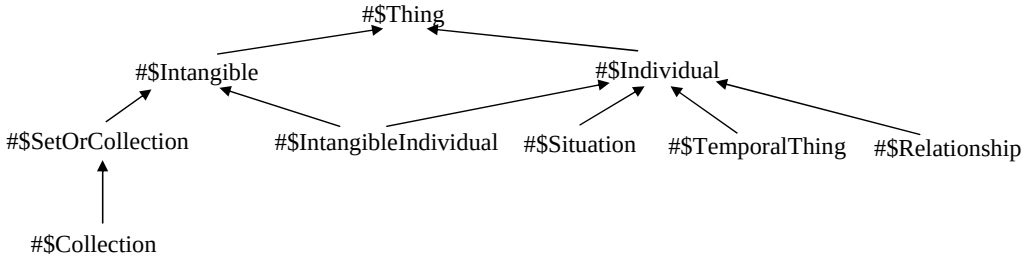
- **Attualmente, Cyc include oltre un milione di concetti, mentre la versione pubblica OpenCyc comprende circa 6.000 concetti e 60.000 relazioni tra di essi**

- **So, the mattress in the road to AI is lack of knowledge, and the anti-mattress is knowledge. But 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" (Guha & Lenat 90, p.4)**

2 componenti

Constraint Language (variante della Logica dei predicati)
CycL (linguaggio basato su frame/units)

Il top level di Cyc



Alcune descrizioni di concetti (dalla documentazione Cyc)

#$Thing:
è l'insieme universale: la collezione di ogni cosa! Ogni costante Cyc nella Base di Conoscenza è membro di questa collezione. Inoltre, ogni collezione della Base di Conoscenza è membro della collezione *#$Thing*.

#$Intangible:
la collezione di cose che non sono fisiche - non sono fatte di, o codificate nella, materia. Ogni *#$Collection* è *#$Intangible* (anche se le sue istanze sono tangibili) e tali sono anche alcuni *#$Individual*. Attenzione: non si confonda 'tangibilità' con 'percettibilità' - gli esseri umani possono percepire la luce anche se essa è intangibile.

#$Individual:
la collezione di tutte le cose che non sono insiemi o collezioni. Così *#$Individual* include, tra le altre cose, oggetti fisici, sottoastrazioni temporali di oggetti fisici, numeri, relazioni e gruppi. Un elemento di *#$Individual* può avere parti o una struttura (incluse parti che sono discontinue); ma NESSUNA istanza di *#$Individual* può avere elementi o sottoinsiemi.

#$IntangibleIndividual:
la collezione degli individui intangibili. I suoi elementi non hanno massa, volume, colore, ecc. Ad esempio, ore, idee, algoritmi, interi, distanze, e così via. D'altra parte, in quanto sottoinsieme di *#$Individual*, questa collezione ESCLUDE insiemi e collezioni, che sono elementi di *#$Intangible*, ma non di *#$IntangibleIndividual*

#$TemporalThing:
la collezione delle cose che hanno una particolare estensione temporale, cose delle quail uno potrebbe ragionevolmente chiedere 'Quando?'. Essa include molte cose; come le azioni, gli oggetti tangibili, gli accordi, e porzioni astratte di tempo. Alcune cose NON sono istanze di *#$TemporalThing* perchè sono astratte, atemporali, come un insieme matematico, un intero, ecc.

OWL
(Web Ontology Language)

- **il legame con il World Wide Web**

- **OWL è un prodotto del W3C (organizzazione internazionale per lo sviluppo di WWW); un linguaggio di markup semantico per pubblicare e condividere ontologie.**
- **OWL è derivato dal precedente DAML+OIL**
- **OWL è basato sui linguaggi di markup (XML) ed è ancorato all'RDF (Resource Description Framework), una notazione uniforme e non ambigua (sintatticamente definita come estensione di HTML tramite XML) per esprimere le risorse presenti sul Web.**

OWL: Web Ontology Language

- **Un aggancio tra strumenti ontologici ed il WWW, nell'ottica di creare un Semantic Web**

VANTAGGI:

- **E' una layer supplementare di strumenti quasi universalmente accettati e in uso**
- **Sono disponibili vari livelli di potere espressivo**

SVANTAGGI:

- **Manca un reasoner incorporato**
- **La sintassi (derivata da HTML e RDF) è pesante e poco leggibile**

Conclusioni sulle Ontologie

- **Le ontologie sono uno strumento essenziale per l'interoperabilità e la comunicazione tra agenti**
- **Negli ultimi 10 anni una quantità impressionante di finanziamenti e attività di ricerca e sviluppo**
- **Molte proposte e sistemi, ma ancora nessuno standard; poichè un'ontologia è utile se è ampiamente condivisa, siamo ancora a metà strada**
- **Alcuni studiosi hanno forti dubbi sulla possibilità (e, forse, sull'opportunità culturale) di realizzare un'ontologia standard (difficoltà tecniche e salvaguardia di diversità culturali)**

ma

- **Si tratta comunque di una strada obbligata**

SISTEMI ESPERTI

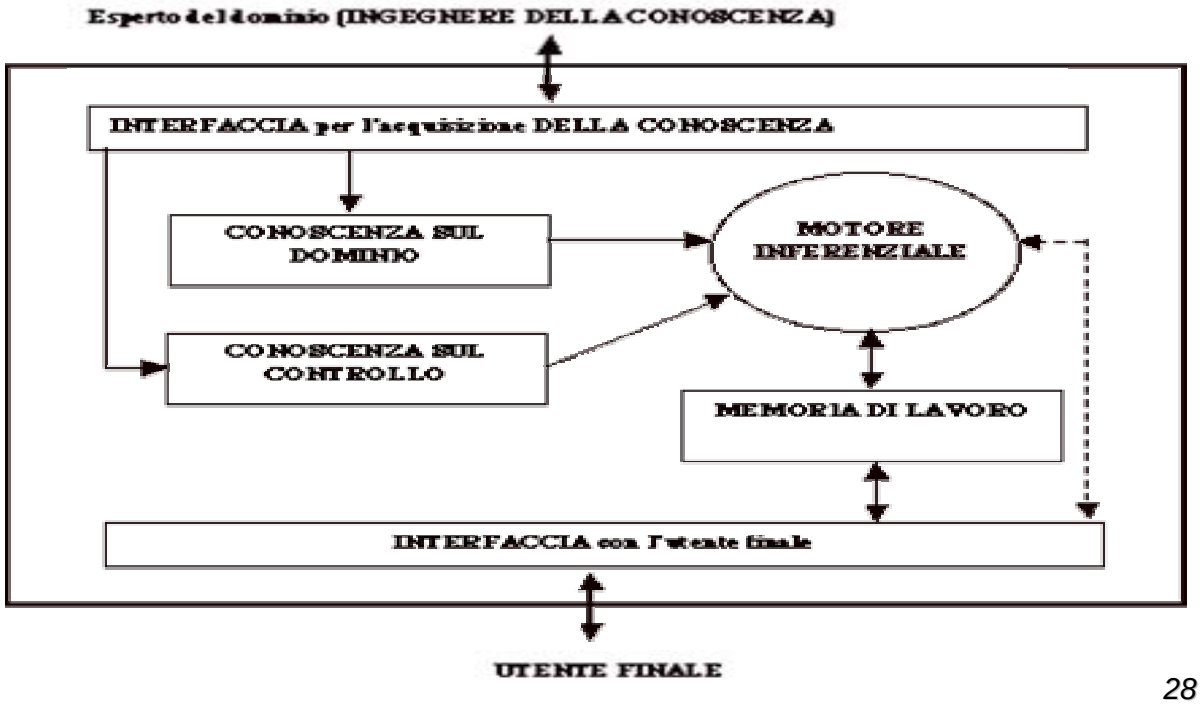
- **Sistemi Basati Sulla Conoscenza (1980)**
- Un sistema **basato sulla conoscenza** è un sistema in grado di risolvere problemi in un **dominio limitato** ma con prestazioni **simili** a quelle di un **esperto** umano del dominio stesso.
- Generalmente esamina un largo numero di possibilità e costruisce dinamicamente una soluzione.
- "*La potenza di un programma intelligente nel risolvere un problema dipende primariamente dalla **quantità e qualità** di conoscenza che possiede su tale problema". (Feigenbaum)*

SISTEMI DI PRODUZIONE

- Un sistema a regole di produzione (production system) è costituito da tre componenti fondamentali:
  - **Base di conoscenza a regole** (che prende spesso il nome di "memoria a lungo termine") in cui sono contenute le regole di produzione;
  - **Memoria di lavoro** (memoria a breve termine) in cui sono contenuti i dati e in cui vengono mantenute le conclusioni raggiunte dal sistema;
  - **Motore inferenziale**.
- Ogni regola di produzione ha la seguente forma:
  - `if <condizione> then <conclusione/azione>`

SISTEMI BASATI SULLA CONOSCENZA ARCHITETTURA

- Rappresentazione della Conoscenza:



SISTEMI BASATI SULLA CONOSCENZA ARCHITETTURA

- **Rappresentazione della Conoscenza:**
  - Regole;
  - Frames;
  - Proposizioni Logiche;
  - Vincoli;
  - Procedure;
  - Demoni;
  - Oggetti;
  - Fattori di Certezza, Variabili Fuzzy.....

- **Modalità di Inferenza:**
  - Ragionamento *forward*;
  - Ragionamento *backward*;
  - Risoluzione;
  - Propagazione di vincoli;
  - Strategie di ricerca euristiche;
  - Ragionamento Ipotetico ed Abduttivo....

INGEGNERIA DELLA CONOSCENZA AMBIENTI

- a) **Skeletal systems** (SHELL)
  - Ottenuti togliendo da Sistemi Esperti già costruiti la conoscenza propria del dominio e lasciando solo il motore inferenziale e le *facilities* di supporto.
  - EMYCIN (Empty MYCIN) deriva da MYCIN;
  - KAS deriva da PROSPECTOR;
  - EXPERT deriva da CASNET.
- b) **Sistemi general-purpose** (TOOLS);
  - KEE, ART, Knowledge-Craft, Nexpert, KAPPA
  - non sono strettamente legati a una particolare classe di problemi: essi permettono una più ampia varietà di rappresentazione della conoscenza e strutture di controllo.
- c) **Linguaggi simbolici** (Prolog, Lisp) **e non** (C, C++).

PASSI DI PROGETTAZIONE

- **IDENTIFICAZIONE** delle caratteristiche del problema.
- **CONCETTUALIZZAZIONE**.
- **FORMALIZZAZIONE**: progetto della struttura in cui organizzare la conoscenza.
- **IMPLEMENTAZIONE**: scrittura effettiva della conoscenza sul dominio.
- **VALIDAZIONE**.

- **SCELTA DI UN TOOL:**
  - non bisogna utilizzare un Tool più generale del necessario;
  - bisogna testare il Tool costruendo un piccolo prototipo del sistema;
  - bisogna scegliere un Tool che sia affidabile ed mantenuto da chi lo ha sviluppato;
  - quando il tempo di sviluppo è critico, è bene scegliere un Tool con *facilities* di spiegazioni/interazione incorporate;
  - bisogna considerare le caratteristiche del problema per determinare le caratteristiche che deve avere il Tool.

APPLICAZIONI

- **Migliaia** di Sistemi Esperti nei settori più svariati.
  - **Interpretazione**: Si analizzano dati complessi e potenzialmente rumorosi per la determinazione del loro significato (Dendral, Hearsay-II).
  - **Diagnosi**: Si analizzano dati potenzialmente rumorosi per la determinazione di malattie o errori (Mycin, ...).
  - **Monitoring**: I dati si interpretano continuamente per la generazione di allarmi in situazioni critiche. Al sistema è richiesta una risposta in tempo reale soddisfacente (VM).
  - **Planning e Scheduling**: Si determina una sequenza intelligente di azioni per raggiungere un determinato obiettivo (Molgen).
  - **Previsione** (economica, politica ecc.) : Si desidera costruire un sistema in grado di prevedere il futuro in base a un appropriato modello del passato e del presente (Prospector).
  - **Progetto e configurazione**: Il Sistema Esperto deve essere in grado di progettare sistemi partendo da ben determinate specifiche (R1, XCON).

CLASSIFICAZIONE E DIAGNOSI

- **Un esempio: Mycin**
  - Sviluppato da E.M. Shortliffe a partire dal 1972;

- **Obiettivi:**
  - decidere se il paziente ha un'infezione che deve essere curata;
  - determinare, se sì, quale è probabilmente l'organismo infettivo;
  - scegliere fra le medicine adatte per combattere l'infezione quella più appropriata in rapporto alle condizioni del paziente.
  - Mycin risolve il problema di identificare un oggetto sconosciuto dalle culture di laboratorio, mediante un *matching* dei risultati di laboratorio con la gerarchia di batteri.

FATTORI DI CERTEZZA

- Compromesso rispetto a un sistema bayesiano puro.
- Introdotti per la prima volta nel Sistema Esperto Mycin.
- Regole non certe (l'implicazione corretta sarebbe invertita) mediate da un **fattore di certezza**.
- Un fattore di certezza è un numero intero "ad hoc" che varia fra +1 e -1.
- Permette l'inserimento di fatti apparentemente contraddittori, ambedue plausibili con differenti valori di certezza.
- **Fatti:**

```
(<attributo> <entità> <valore>
<fattore di certezza>).
```

- Esempi di fatti espressi in Mycin sono (sintassi del Lisp):

```
(SITE CULTURE-1 BLOOD 1.0)
(IDENT ORGANISM-2 KLEBSIELLA .25)
(IDENT ORGANISM-2 E.COLI 0.73)
(SENSITIVS ORGANISM-1 PENICILLIN -1.0)
```

REGOLE

```
PREMISE <premessa> ACTION <azione>.
PREMISE
($AND
(SAME CNTXT INFECT PRIMARY-BACTEREMIA)
(MEMBF CNTXT SITE STERILESITES)
(SAME CNTXT PORTAL G1))
ACTION
(CNTXT IDENT BACTEROIDES  0.7).
```

- Significato della regola

IF
(1) the infection is primery-bacteremia,
(2) the site of the culture is one of sterilesites, and
(3) the suspected portal of entry of the organism is the gastro-intestinal tract,
THEN there is a suggestive evidence
(0.7) that the identity of the organism is bacteroides.

R1

- Sviluppato all'Università Carnegie-Mellon da John Mc Dermott per conto della Digital a partire dal 1978.
- COMPITO PRINCIPALE: configurare il calcolatore VAX-11/780 automaticamente.
- In base all'ordine del cliente, R1 è in grado di:
  - assicurare che l'ordine sia completo;
  - determinare le relazioni spaziali fra le componenti.
- Un tipico sistema ha più di 100 componenti con varie possibilità di interazione.

- Esperti: technical editors
- * Nato con un nucleo di 250 regole ora ne possiede circa 2800.
- Dal 1980 è un prodotto funzionante Digital.
- R1 è implementato in OPS-5.

S.E. SVILUPPATI DAL GRUPPO DI IA Univ. Bologna e Univ. Ferrara

- Sistemi **utilizzabili** (almeno allo stato prototipale) nelle Aree:Progetto, Monitoring, Diagnosi, Scheduling.

- **ADES** (ATP Design Expert System) per il **progetto** dei sistemi per il controllo delle stazioni ferroviarie (SASIB);
- **SMA** (Station Master Assistant) per il **monitoring** e la pre-**diagnosi** degli enti della stazione al fine di determinare la fattibilità degli itinerari (SASIB);
- **TSA** (Train Scheduling Assistant) per regolare il traffico dei treni all'interno di una stazione di grosse dimensioni (SASIB).
- **FUN** (Function Point Mesaurement) per il calcolo dei Function Point per un sistema software.
- Identificazione di difetti in semilavorati meccanici (BERCO S.p.A, approccio mediante apprendimento automatico di regole).

Sistemi Esperti in campo medico

- Diagnosi, verifica degli esami medico-clinici, interpretazione dei dati (DIANOEMA SpA, S..Orsola-Malpighi Bologna). In particolare:

- DNSEV (Expert System for clinical result Validation), per migliorare la qualita` del processo di validazione eseguito dai laboratori di analisi biochimica.
- ESMIS (Expert System for Microbiological Infection Surveillance), per migliorare la qualita` del processo di validazione eseguito dai laboratori di analisi microbiologica e per monitorare gli eventi infettivi all'interno di un ospedale.
- DNTAO (Expert System for supporting the Oral Anticoagulation Treatment) per il supporto ai medici (ematologia) per le prescrizioni e visite per la Terapia Anticoagulante Orale.
-

PROSPETTIVE

- Verso gruppi di **agenti intelligenti** cooperanti (Progetto SOCS-UE)
- Cooperazione con un essere umano
- Mondo "virtuale" Internet.
- Integrazione con vari sistemi informatici (interfacce grafiche, database ecc..), tecnologie tradizionali e non (reti neurali, sistemi fuzzy)
- Da costose workstations special-purpose (LISP-Machine) verso sistemi Unix o PC con l'utilizzo di Tools scritti in linguaggi più tradizionali (C, C++).