



The Role of Ontology in Modern Expert Systems Development

Jason Morris
Morris Technical Solutions LLC

Long-Distance Dedication

Source: <http://newsgroups.derkeiler.com/Archive/Comp/comp.ai/2005-11/msg00011.html>

- *From:* "g_chime@xxxxxxxxxx" <g_chime@xxxxxxxxxx>
- *Date:* Fri, 11 Nov 2005 13:55:58 GMT

I'm new to AI and my boss wants me to develop a knowledge based classifier software so I have to learn fast.

Based on reading a few papers and websites, I think I will use an expert system and an Ontology. (Am I on the right track?)

I think (but I'm not sure) that I will need both forward and backward chaining.

What tools / languages should I use?

CLIPS, JESS, CLIPS/R2 for the expert systems shell?
OWL for the ontology?

Any suggestions / help is appreciated.

Outline

- Prologue: Setting the Stage
- Part I: Knowledge Engineering
- Part II: Ontology Fundamentals
- Part III: Creating Ontologies
- Part IV: SINFERS Example
- Epilogue: References and Q & A

Prologue: Setting the Stage

Concepts and Context

Key Questions

1. What exactly is a **modern expert system**? What does that entail?
2. Briefly, what is an **ontology**?
3. What is the **relationship** between ontologies and expert systems?

To answer these questions, we require some background. Let's tackle question #1 now.

Old vs. New Expert Systems

Old

- Monolithic
- Pre-shell
- As “oracles”
- Computer-centric
- Non object-oriented architecture
- “Waterfall” dev

New

- Componentized
- Shell-derived
- As services
- Network-centric
- Object-oriented architecture
- RAD, spiral, and XP dev

Modern Expert Systems

- Built from **OTS/OSS** components.
- Constructed using **shells**.
- Act as an expert **agent** and provide expertise as a **service**.
- Utilize the **internet** as a source of "**common-sense**".
- **Designed** with the latest OOP concepts , RAD/XP practices, and HCI factors.

The AI Value Proposition

Why care about AI/ expert systems at all?

1

Modern businesses need to make complex decisions.



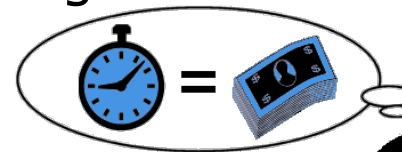
2

Complex decisions require lots of information and applied knowledge.



3

Such decisions must be made quickly and reliably.



=>

By reasoning about information using applied knowledge, expert systems help stakeholders make timely and reliable decisions.

Market Drivers & Enablers

What is driving the apparent renaissance AI?

1

Hardware Power



Dramatic increases in CPU speed, RAM capacity, storage, etc.

2

Hardware Cost



Mass-production has lowered technology costs.

3

INTERNET

An incubator for new technologies and a source for new markets – a “killer-app”!

What is an ontology?

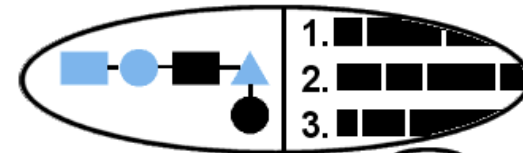
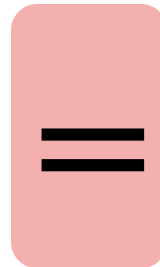
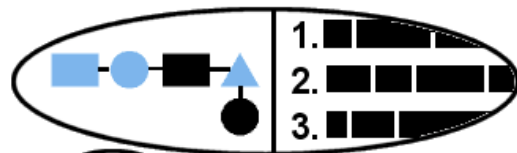
In the AI context, an ontology is:

- A collection of **classes**, their **attributes**, and their **relationships**.
- A **description** or **model** of a **domain of discourse** or knowledge (area of expertise).
- A **vocabulary** for conveying thought and conducting reasoning in a domain.

What is an ontology?

- Ontologies are used to **encapsulate** domain knowledge.
- Artificial Intelligence (AI) applications perform symbolic **reasoning** over those domains.
- Ontologies define the **limits** of symbolic reasoning in AI applications.

Linking ES and Ontologies



**Ontologies
disambiguate
meaning.**



Ontology



Part I: Knowledge Engineering

Moving from Noise
to Knowledge

Our Knowledge Engineer

- May or may not be a **domain expert** in own right – but possible.
- Expert **programmer**.
- Software **architect**.
- **Liaison** skills.
- Great communicator.
- Makes raw knowledge **programmable**.



From the Engineer's POV

Our knowledge engineer would prefer that her project get recognition here...



Rather than here.



Knowledge Representation



Typically in ES development, the first design decision is how knowledge will be represented in the system.

Ontologies provide:

- An approximate **model** or **view** of the world with respect to a domain.
- The **bounds** of an intelligent system's knowledge base.
- A foundation for **sense-making** and **reasoning**.

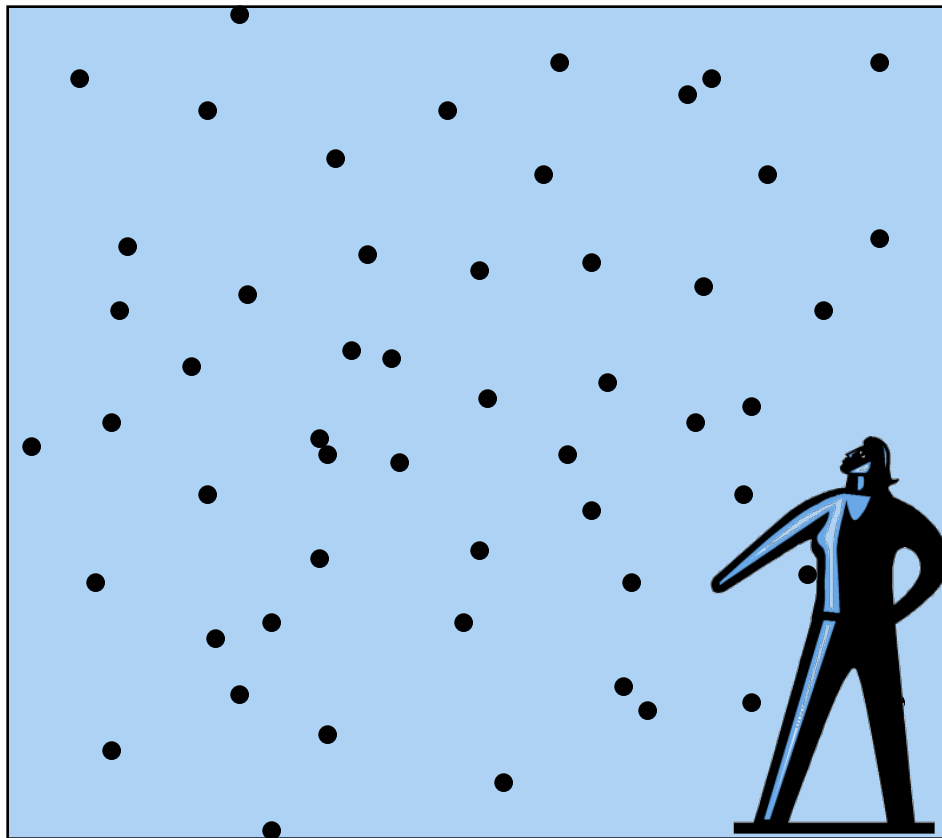
Parlez-Vous AI?

From a popular ontology editor help file...

“Instances are the actual **data** in your **knowledge** base ... If you have to make changes to your class or slot structure after instances have been entered, you may lose some **information**.”

Anything odd about this?

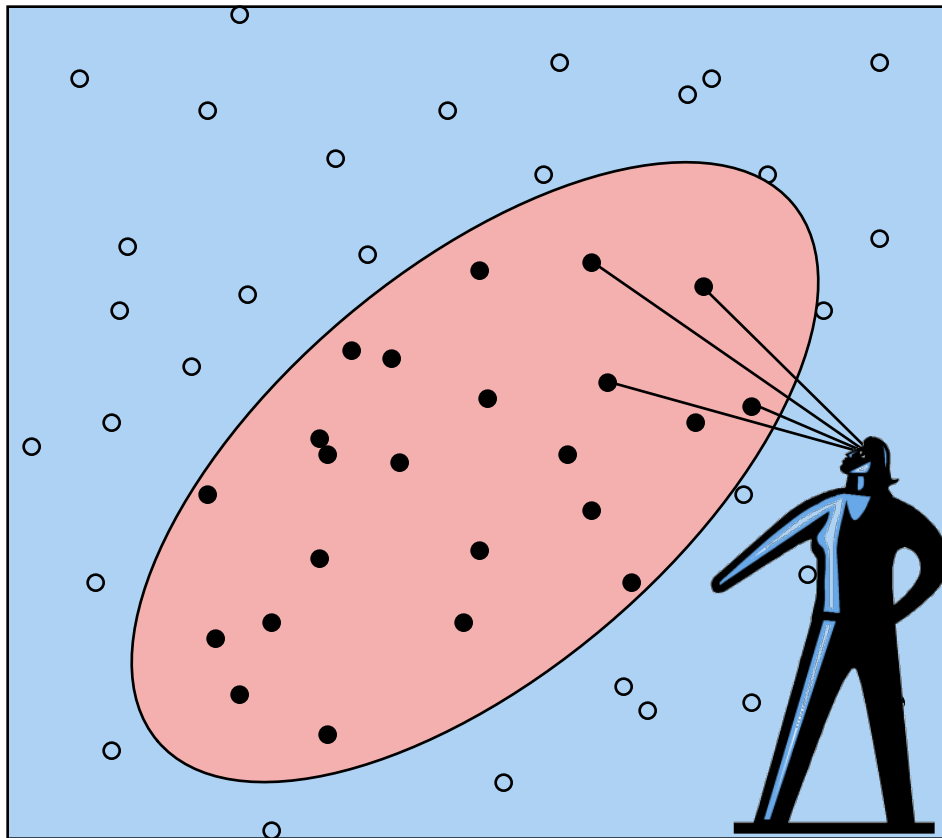
Noise



Defined as the universe of **all** possible **invariants...**

...an endless sea of qualitative and quantitative values without a **cognitive pattern.**

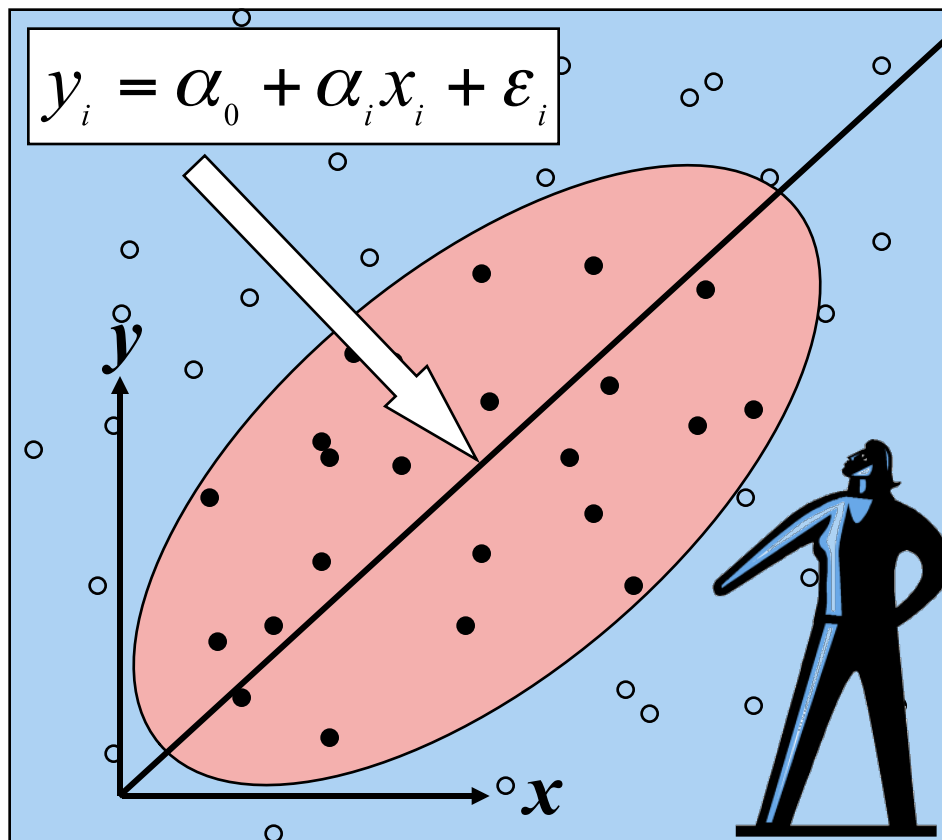
Data



Noise is filtered and sampled to separate **useful** measurements (facts) and to form data.

Thus, in a sense, data is **created** via our cognitive attention.

Information



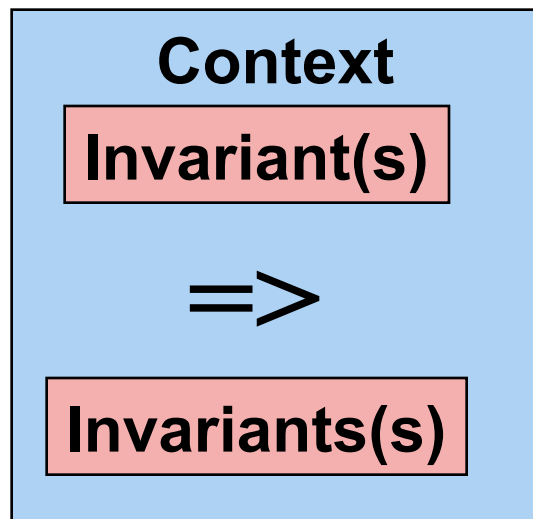
Data is analyzed and interpreted, to uncover **meaning** and **relationships**, producing actionable information.

Information aids **decisions**.

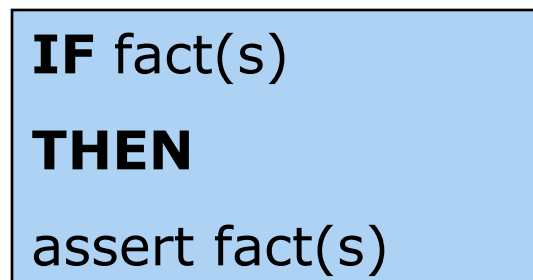
Knowledge

- Knowledge is **derived** from information.
- One application of information is to make **decisions**.
- When we observe the **outcomes** of those decisions, we can uncover new data or information, and generate new knowledge.
- Knowledge comes in two main **types**...

Declarative Knowledge Rules

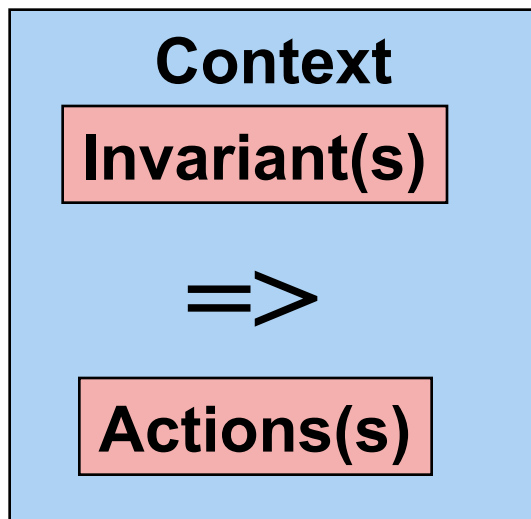


These represent the invariants that can be inferred when one or more invariants hold.



We can represent them by **asserting** a fact or facts when certain other facts are present.

Procedural Knowledge Rules



These represent the **actions** to take when certain invariants hold.

```
IF fact(s)  
THEN  
call function foo
```

We can represent them by calling **functions** when certain facts are present.

Declarative vs. Procedural

An example of
declarative
knowledge.

An example of
procedural
knowledge.

IF

(instance-of ?x THING)

(composed-of ?x CLAY)

(composed-of ?x SAND)

(composed-of ?x SILT)

THEN

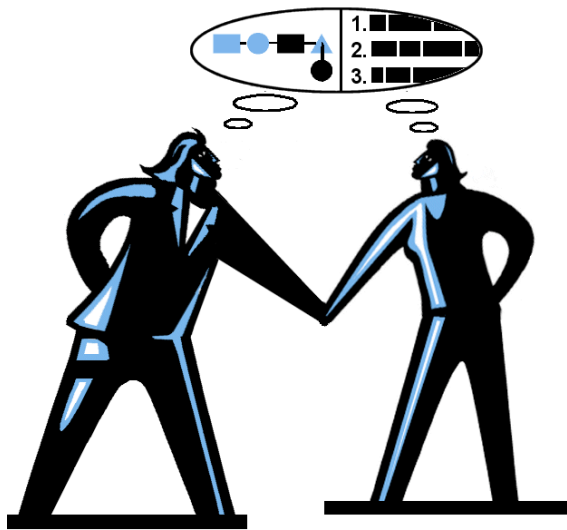
(instance-of ?x SOIL)

IF we have the soil property
values $x_1 \wedge x_2 \wedge \dots \wedge x_n$

THEN we can compute
the soil property

$$y_m = f(x_1, x_2, \dots, x_n)$$

Ontological Commitment



“An **agreement** to use a **vocabulary** (i.e., ask queries and make assertions) in a way that is consistent (but not complete) with respect to the theory specified by an **ontology** ...

An agent **commits** to an ontology if its **observable actions** are consistent with the definitions in the ontology.” – Tom Gruber

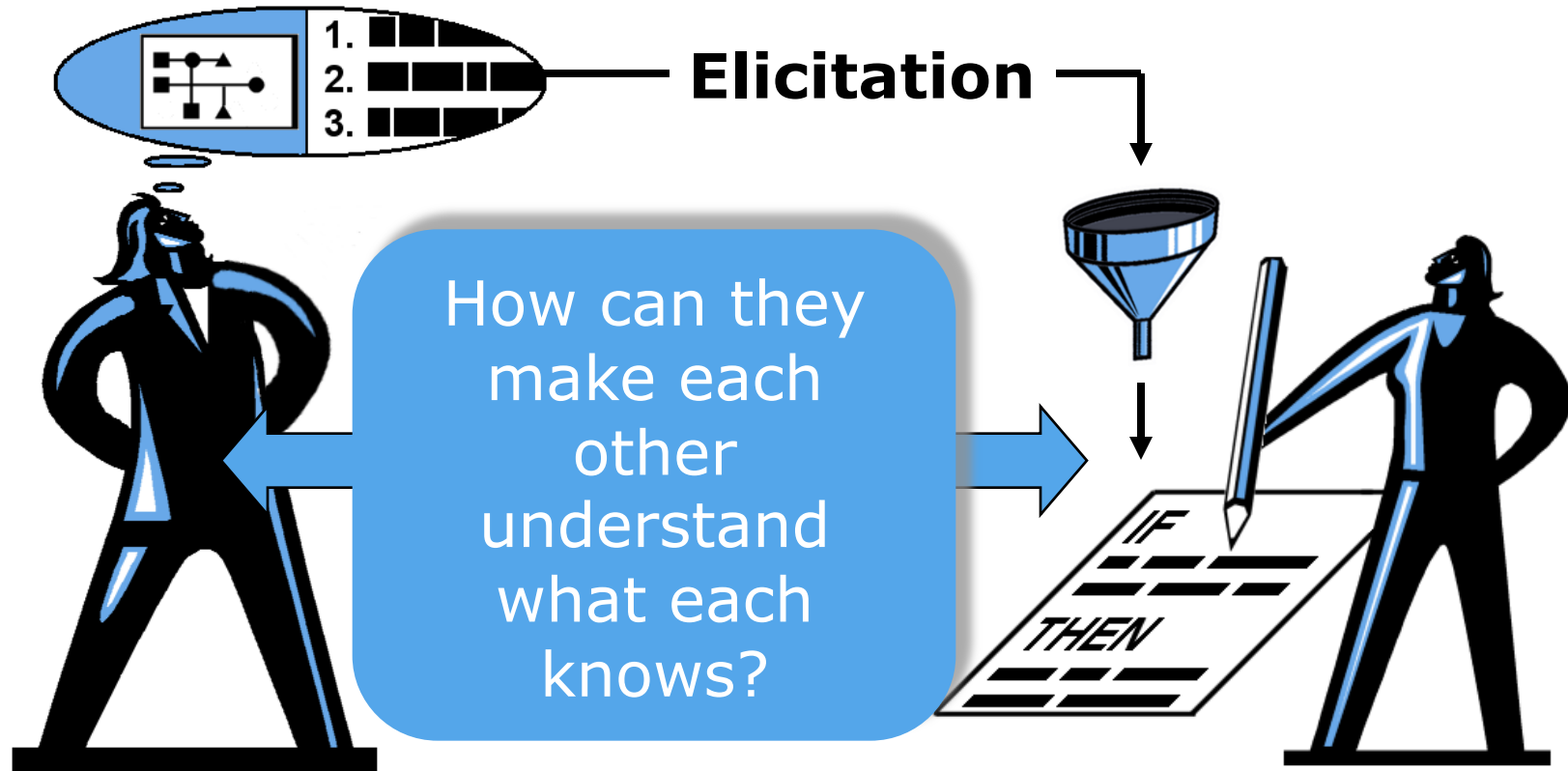
Tacit (Implicit) Knowledge



- Most expertise is **tacit** or **implied**.
- Tacit knowledge is hard or impossible to **quantify** or **qualify**.
- Often the result of extensive **experience**.
Hard to verify.
- Inseparable from original problem **context**.

The “Bottleneck” Issue

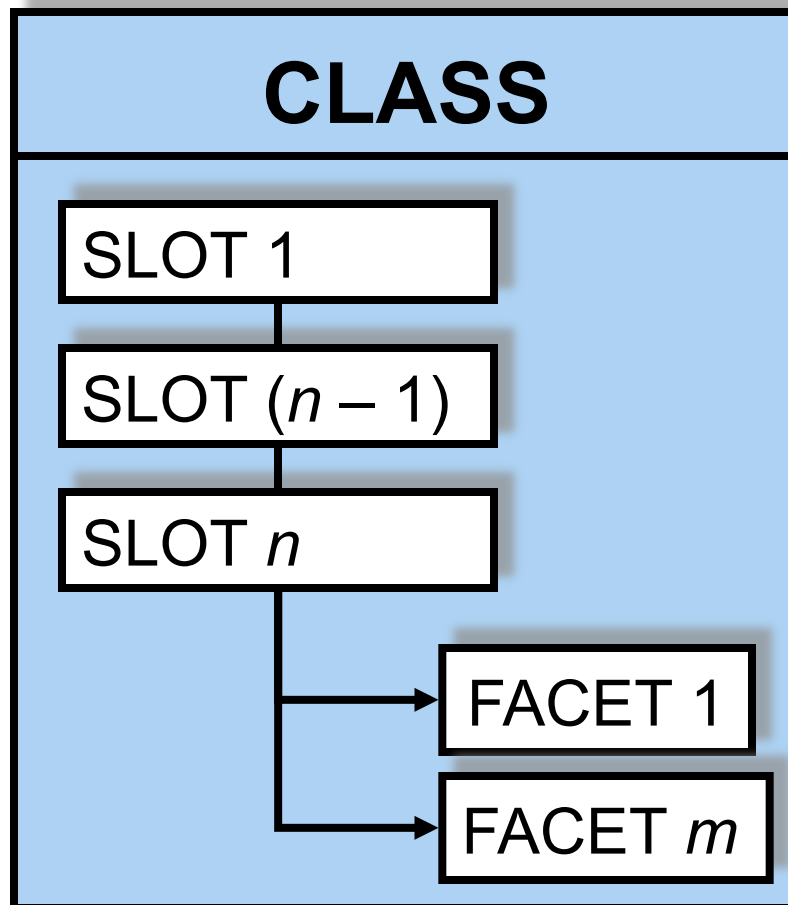
Converting Expert Knowledge To Rules



Part I: Ontology Fundamentals

The How, What, Where,
When, and Why of Ontologies

Ontological Classes



CLASS

Represents a “thing” or concept.

SLOT

A data field within a class. Type is optional.

FACET

An allowed value for a slot [optional].

Classes vs. Instances

Soil Property

id INTEGER

symbol STRING

value FLOAT

stdev STRING

units STRING

organic_carbon

id 42

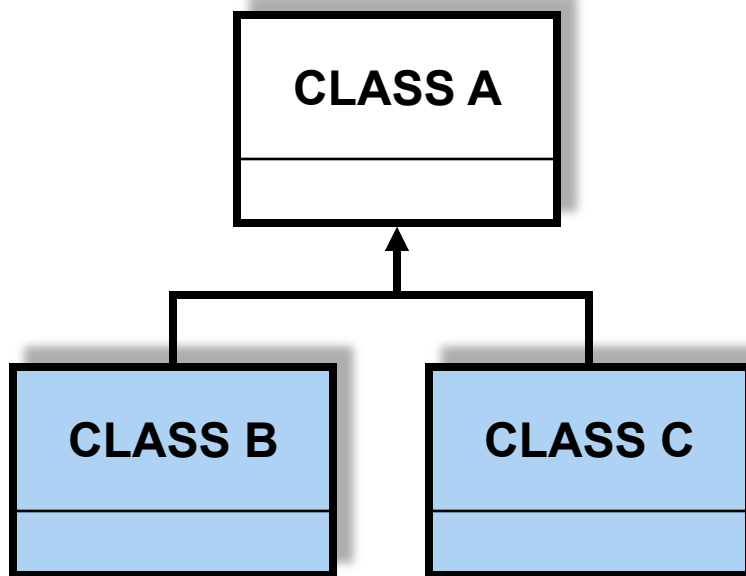
symbol "_6A1"

value 0.06

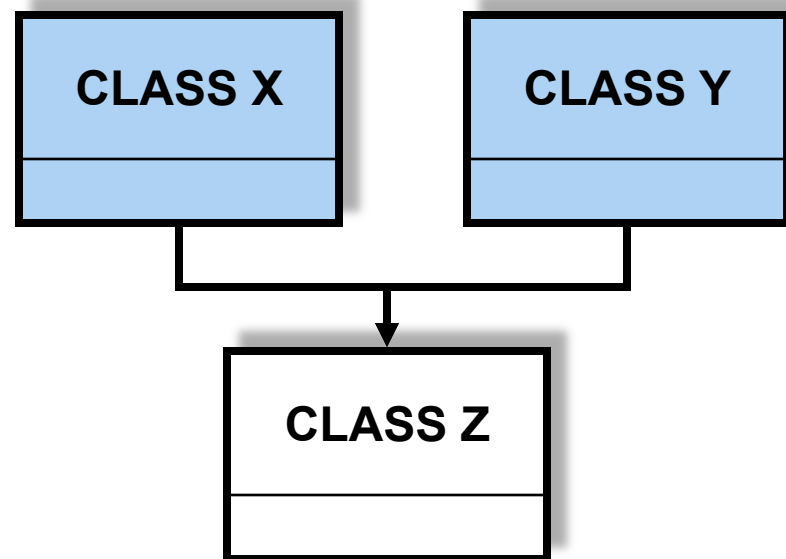
stdev 0.01

units "kg/kg"

Specializing vs. Inheriting

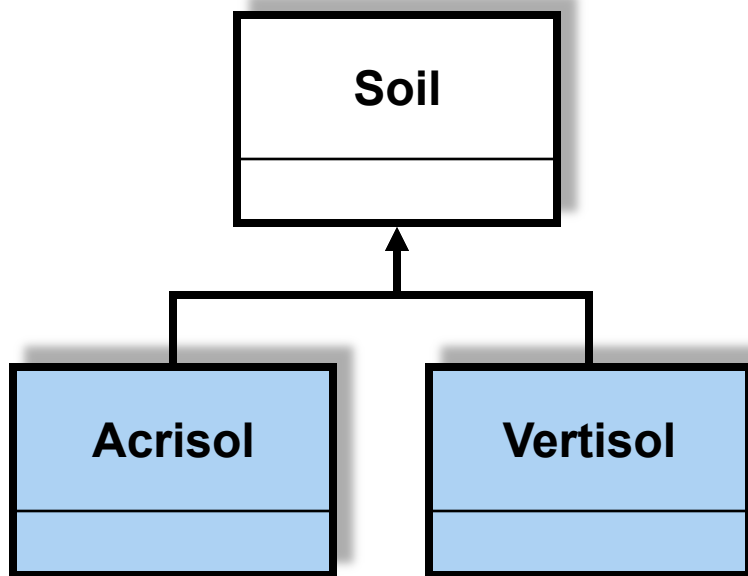


Subclassing or extending a parent class, superclass, or base-class

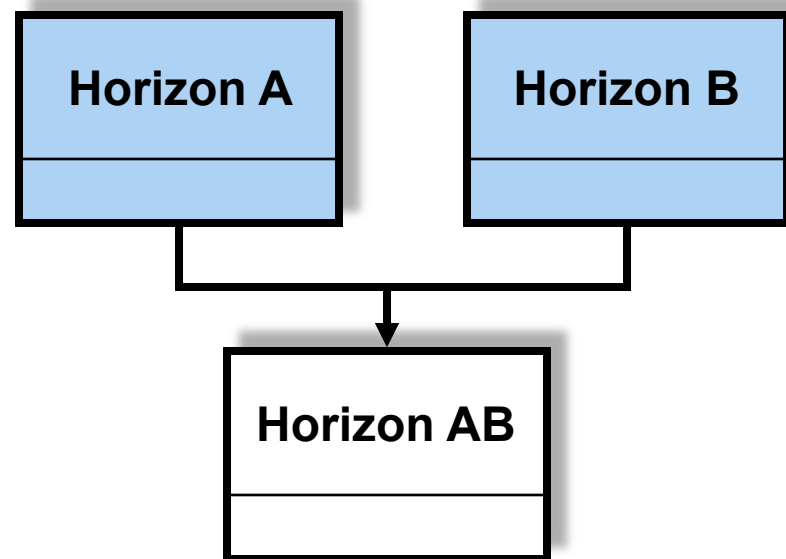


Inheritance from one or more parent classes

Subclass / Inherit Examples

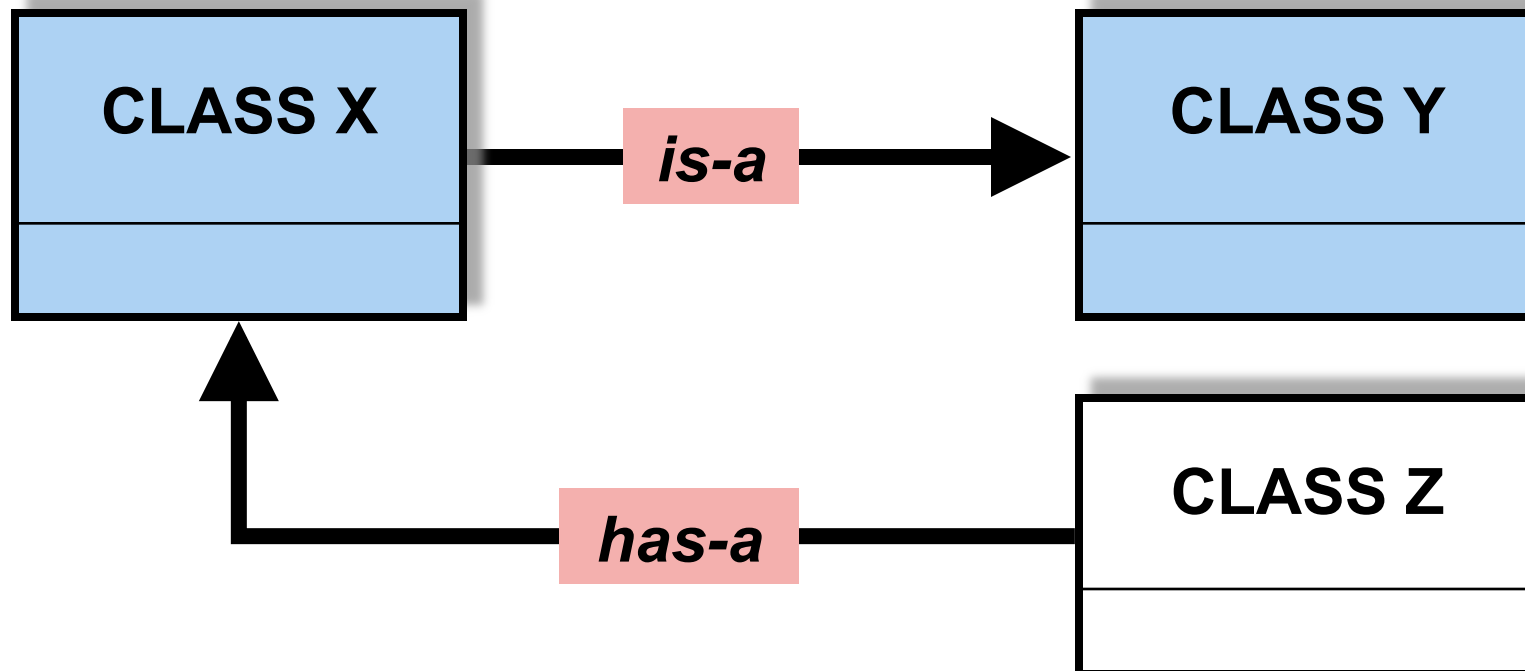


Acrisol and **Vertisol** are **specializations** of Soil.



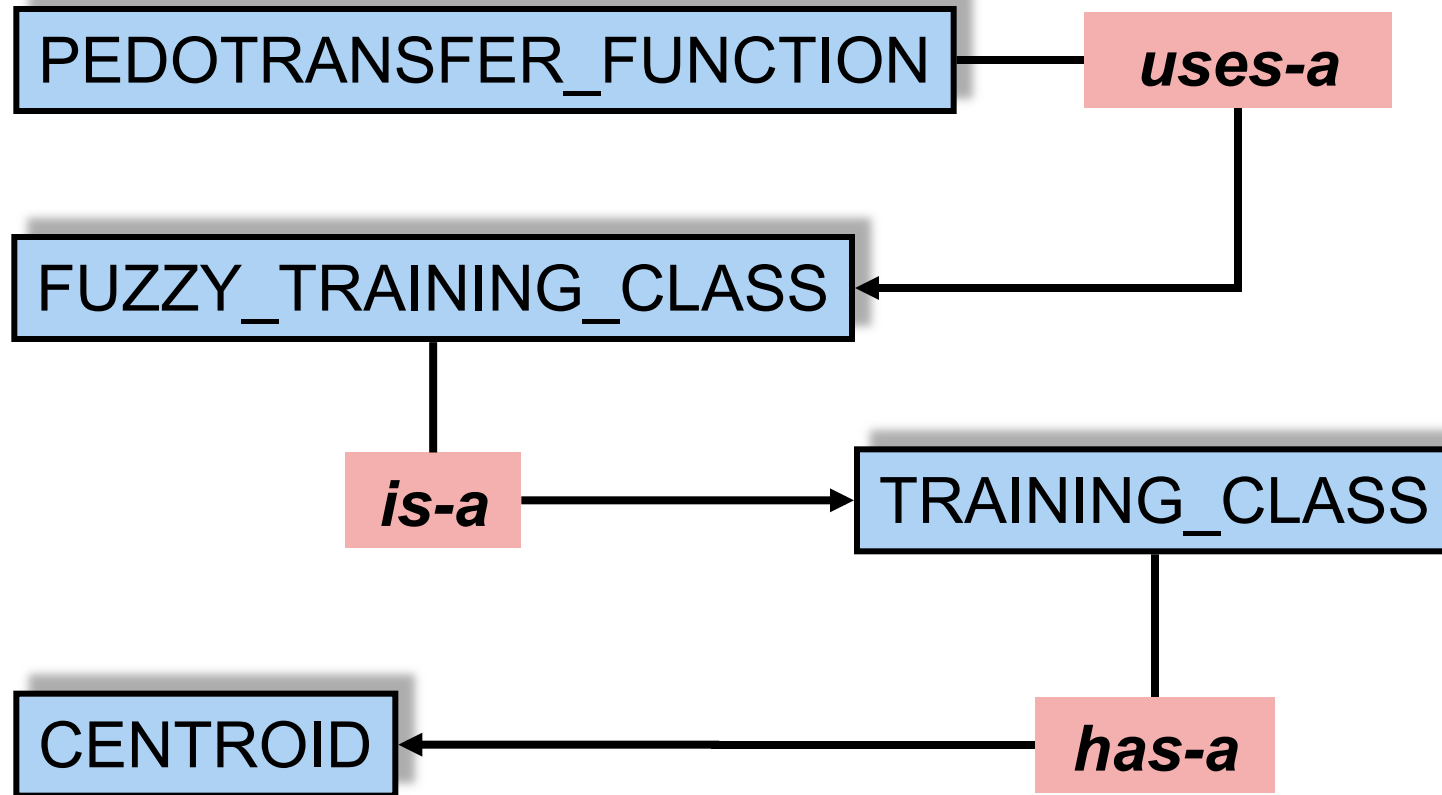
A soil horizon can **inherit** properties from distinct types.

Ontological Relationships



Relationships connect pairs of classes.

Ontological Relationships



Relationships are modeled as slots.

Taxonomy vs. Ontology

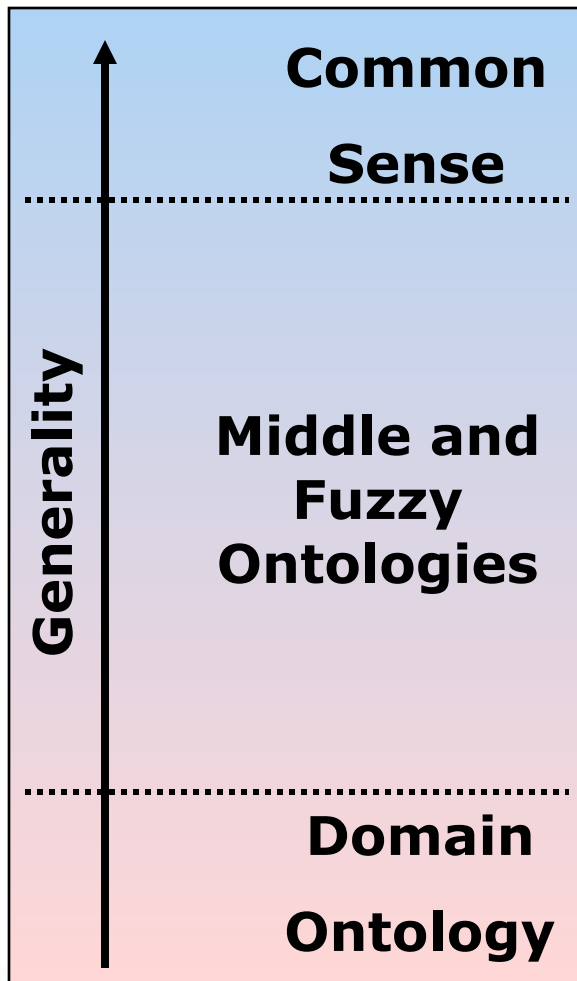
Taxonomies:

- Usually are a single, hierarchical classification within a subject
- Primarily focused on “is-a” relationships between classes
- Limited in inferencing potential due to lack of relational expressiveness.

Ontologies:

- Subsume taxonomies.
- Include attributes with cardinality and restricted values.
- Unlimited relationships between entities.
- Superior inferencing support due to relational expressiveness.

Upper Ontologies



- **Benefit:** Provide common sense.
- **Example:** Cyc Project.
- **Issues:**
 - Web 2.0 (Semantic Web)
 - Is this the key to real AI?
 - Harder to generalize knowledge.
 - Where to stop? Granularity?
 - Overlap and integration?

Implied Ontologies

(deftemplate soil-property

"A fact describing a soil property"

(slot symbol)

(slot value)

(slot error)

(slot units))

(deftemplate ptf

"A fact describing a pedotransfer function"

(slot symbol)

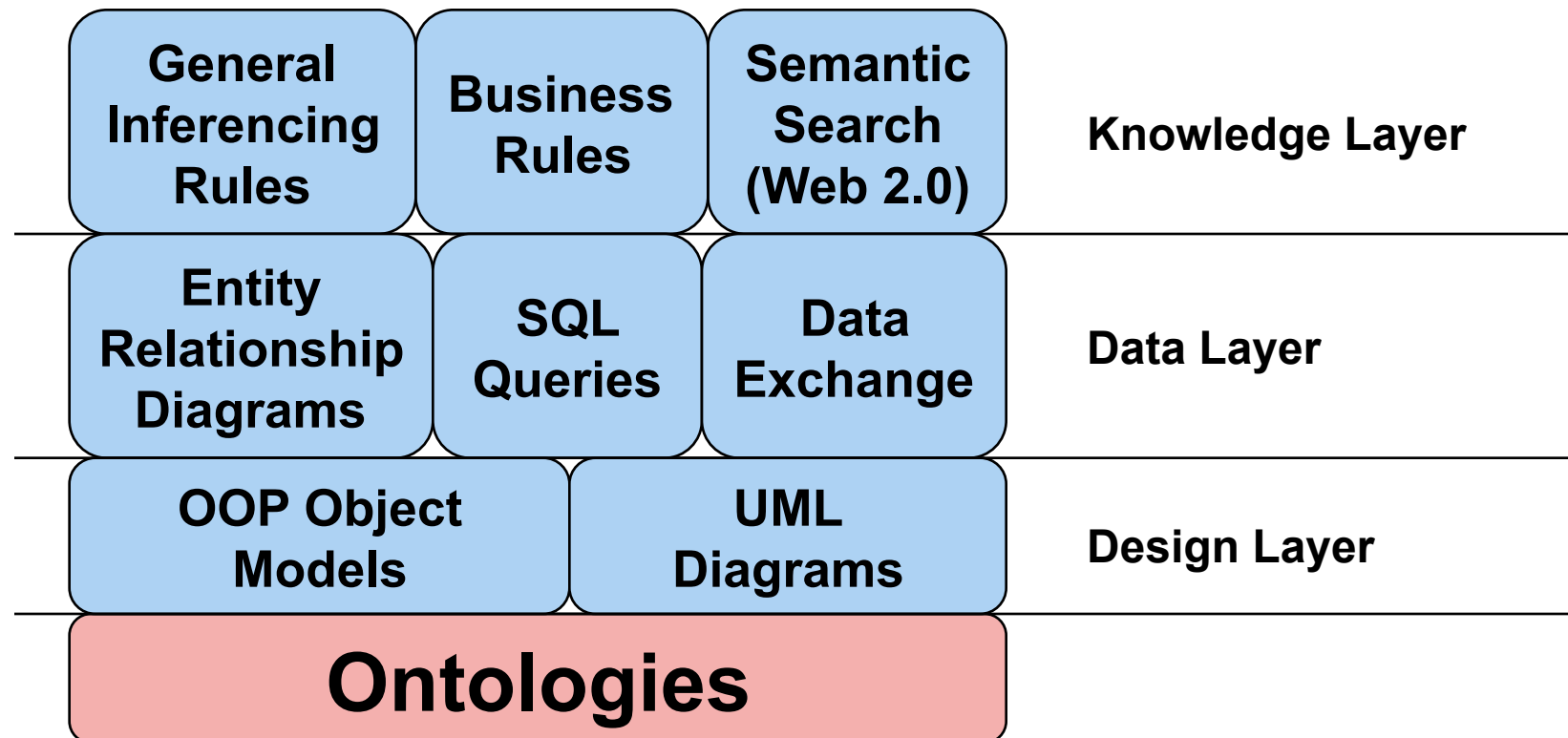
(multislot args)

(slot value)

(slot units))

Using these templates, what could an expert system reason about implicitly?

How do ontologies help?



Why create ontologies?

- To share **common** understanding of the structure of information among **people** or software **agents**.
- To enable **reuse** of domain knowledge.
- To make domain assumptions **explicit**.
- To separate domain knowledge from the **operational** knowledge.
- To **analyze** domain knowledge.

Source: Noy, N.; McGuinness, D. 2001

Why create ontologies?

- To accommodate future domain **growth**.
- To provide for inter-operability with **legacy** and future intelligent systems.
- To accommodate **complex domains** with many classes and relationships.
- To support systems where **implicit ontologies** are insufficient.

Problem Solving Methods

Defined as reasoning strategies for solving certain problem types. *See Also* Generic Tasks.

Classical PSM

- Heuristic Classification
- Generate & Test
- Propose & Revise
- Cover & Differentiate
- Acquire & Present

Key PSM Authors

William Clancey
B. Chandrasekaran
John McDermott
Allen Newell
Robert Wielinga

You design your application ontology to support your problem solving method.

Problem Solving Methods

In “Preliminary Steps Towards a Taxonomy of Problem Solving Methods”, McDermott says...

“...In traditional expert systems terminology, a problem-solving method is called an **inference engine.**”

In our case, we are using a **rule-engine** as our inference engine. So, **forward-chaining** through rules **is** our problem-solving method.

PSMs, Tasks, and Goals

PSM

goal_1:

task_1_1

task_1_2

goal_2

task_2_1

task_2_2

task_2_m

goal_n

task_n_1

Rule Engine

module_1:

rule_1_1

rule_1_2

module_2

rule_2_1

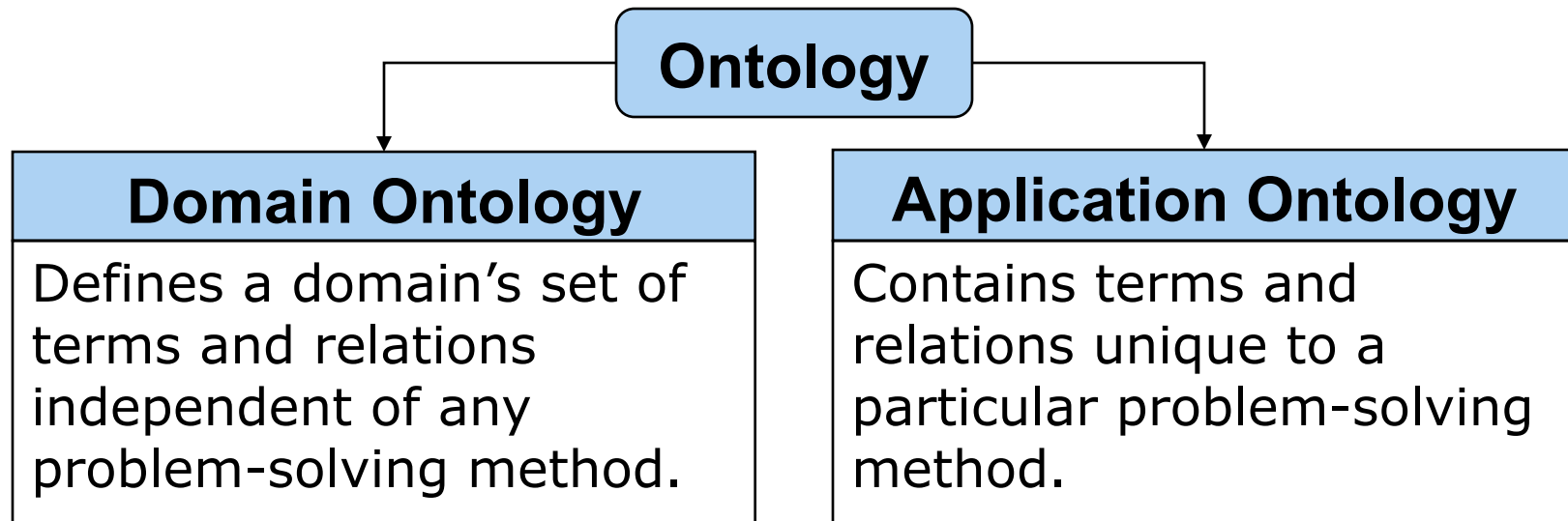
rule_2_2

rule_2_m

module_n

rule_n_1

Ontology Bifurcation



Doing this allows developers to make explicit tradeoffs between **reusability** and **granularity** of the required domain knowledge. Separates out **control** and **procedural** knowledge.

Closed World Assumption

If a thing or concept is **not** explicitly defined in an ontology where CWA holds, then:

- All references to it are logically **FALSE**.
- You **cannot** reason about it.

CWA is used when a “best” answer is required despite an incomplete knowledge base.

Linking Rules & Ontology

- Ontological **instances** are the **input** and **output** of expert systems backed by ontologies.
- Rules **operate** on ontological instances.
- The **granularity** of your ontologies determines the **expressiveness** of your **facts** and **rules**.
- The expressiveness of your facts and rules determines the degree to which you can make **inferences** within your problem scope.

Part II: Ontology Creation

How to design and build a domain ontology.

Ontology Truisms

- An ontology is a **designed artifact**.
- There is no **right** or **wrong** way to begin.
- A **bottom-up** or **top-down** approach is fine.
- Expressiveness comes from relations not the classes and their slots.
- Separate **domain ontologies** from **control ontologies** so that control implementation is free to vary.

Ontology Creation Basics

1. Identify as many “**things**” and **concepts** in your domain as you can.
2. For each thing or concept, enumerate its **attributes** and **properties**.
Specify any **restricted** values.
3. For each pair of things or concepts, decide if there is some **relationship** between them. Look for hierarchy, composition, cooperation, and dependence.

Markup Technologies

- RFD - Resource Description Framework
- RDF Schema -
- DAML - DARPA Agent Markup Language
- SHOE - Simple HTML Ontology Extensions
- OIL - Ontology Inference Layer
- DAML+OIL
- OWL - Web Ontology Language

Using an Ontology IDE

- Saves you time and effort **designing**.
- Easy to **export** and **import** your ontology via different formats (i.e., RDF, OWL, etc.).
- Often coupled with an **instance editor** to create a “knowledge-base”.
- Protégé is the defacto OSS standard.
<http://protege.stanford.edu/>

Ontology Languages / Tools

- Ontolingua
<http://www.ksl.stanford.edu/software/ontolingua/>
- Loom / Power Loom / Ontosaurus
<http://www.isi.edu/isd/LOOM/LOOM-HOME.html>
- SWOOP
<http://code.google.com/p/swoop/>
- OntoEdit
<http://www.ontoknowledge.org/tools/ontoedit.shtml>
- TopBraid Composer
<http://www.topquadrant.com/topbraid/composer/index.html>

Part IV: SINFERS Example

Lessons Learned from the Soil
Inferencing System
(SINFERS) Project

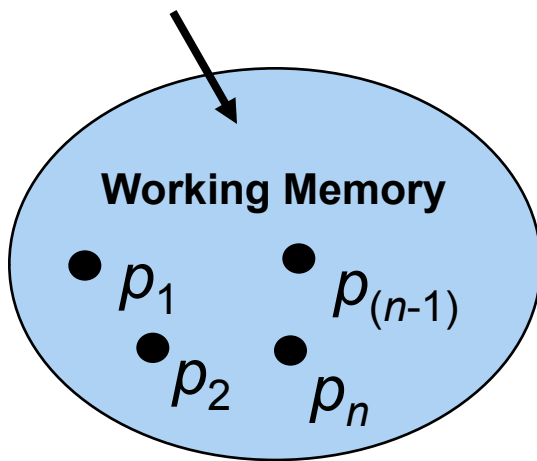
SINFERS Project



- University of Sydney, AU
- Estimation of soil properties via **pedotransfer functions** (PTF).
- Use **rules** to select PTFs.
- Used by soil scientists, farmers, civil engineers, and conservationists.

SINFERS Design Task

Given an initial set,
 $\{P\}$ of n soil
properties



rule compute-ptf-p*

IF $\{q\}: \{q\} \subseteq \{P\}$
PTF($\{q\}$)

THEN compute p^* and
add to working memory

$$p^* = f(\{q\})$$

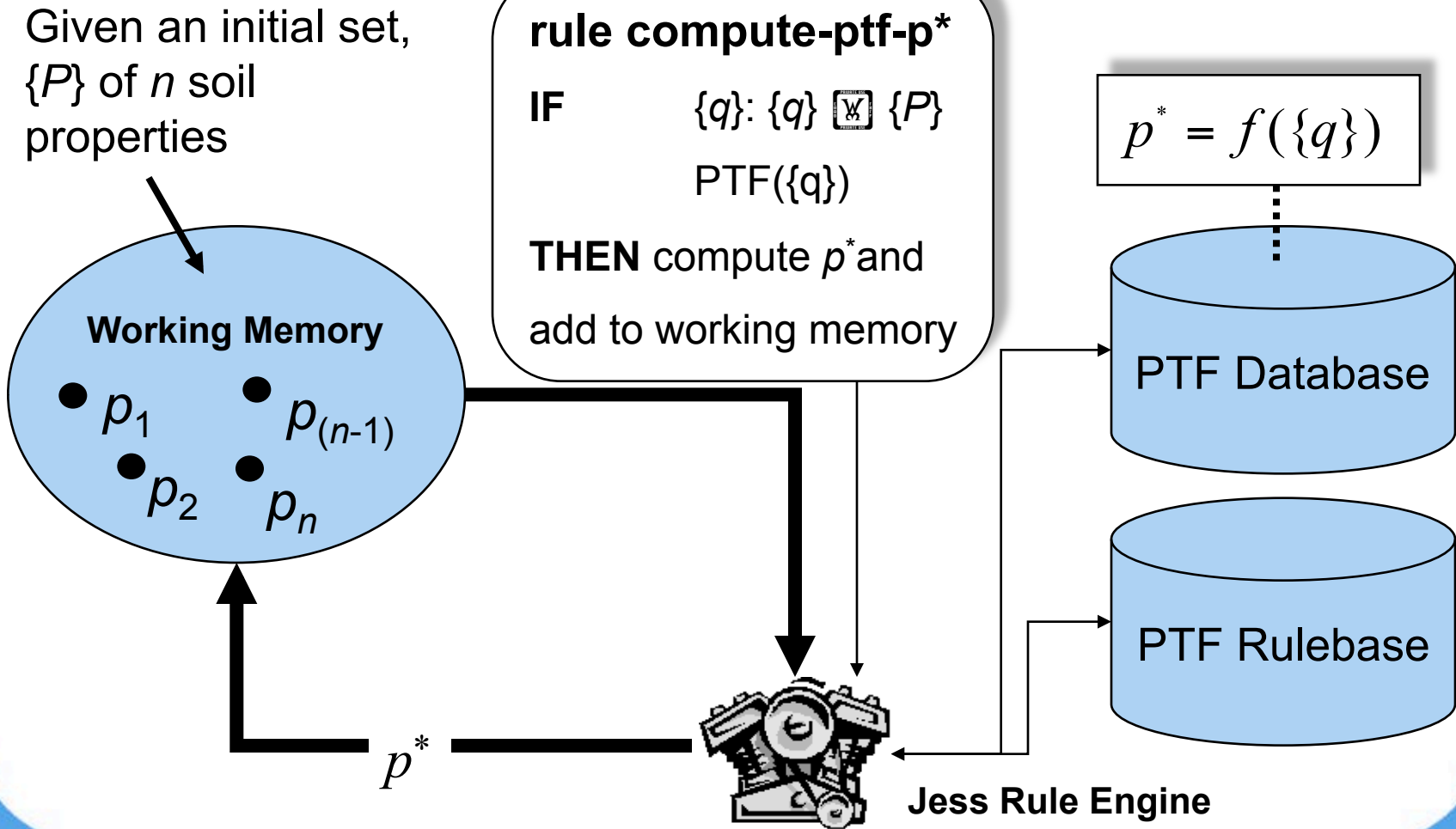
PTF Database

PTF Rulebase



Jess Rule Engine

p^*



SINFERS in Protégé

SINFERS Protégé 3.3.1 (file:IC:\TEMP_DOWNLOADS\SINFERS.pprj, Protégé Files (.pont and .pins))

File Edit Project Window Tools Help

Classes Slots Forms Instances Queries Jess Script Console

CLASS BROWSER

For Project: SINFERS

Class Hierarchy

- :THING
 - :SYSTEM-CLASS
 - PROJECT
 - SITE
 - OBSERVATION
 - SOIL_SAMPLE
 - SOIL_PROPERTY
 - HORIZON
 - LOCATION
 - COLOR
 - HORIZON_DESCRIPTOR

Superclasses

- :THING

CLASS EDITOR

For Class: SOIL_PROPERTY (instance of :STANDARD-CLASS)

Name: SOIL_PROPERTY

Documentation: Models a SINFERS soil property.

Constraints

Role: Concrete

Template Slots

Name	Cardi...	Type	Other Facets
display-label	single	String	
laboratory-code	required ...	Symbol	allowed-values={P4_50_McK,P3B_NR_001,P3E
method-uncertainty	required ...	Float	
method-units	required ...	String	
property-type	single	Symbol	allowed-values={biological,chemical,morpholog
sinfers-label	required ...	String	

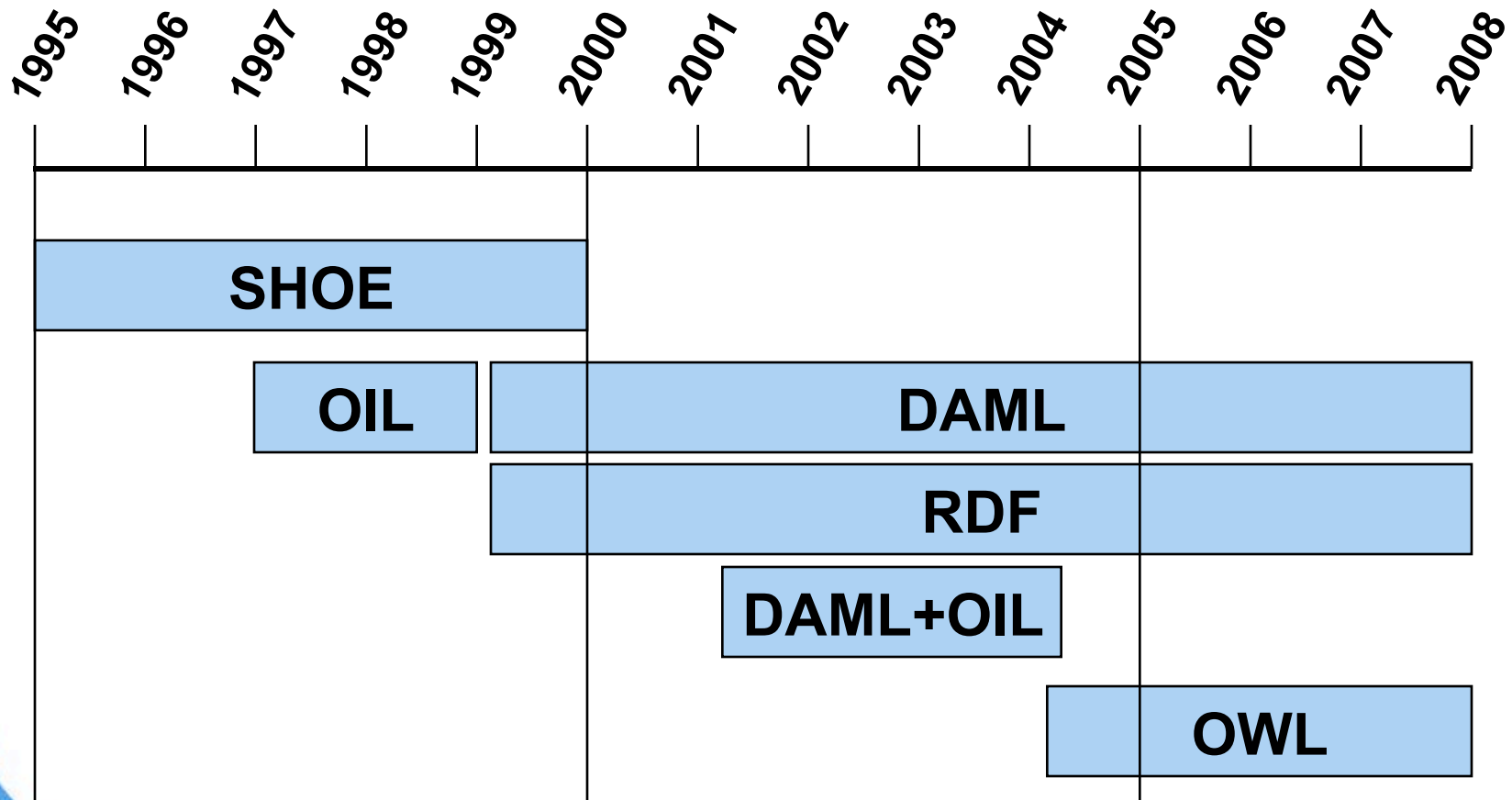
How Ontology Pays Off

- We avoided hard-coding PTFs as functor-like Java classes.
- We avoided ugly GUI issues via early detection.
- We were able to more rapidly agree on design intent once we had a common vocabulary.
- We can write XSLTs to map our input spec to other Australian government database schema.

Epilogue

References and Q & A

Web Ontology History



Famous Ontology Folks



Tom Gruber



Deb McGuinness



Dieter Fensel

Many of their colleagues and students are also great sources of ontology literature.

Take-Away Points

Developing a domain ontology can help:

- Facilitate **communication** and **understanding** between **knowledge engineers** and **domain experts**.
- Establish **relationships** and **semantics**.
- Fulfill **Closed-World** assumptions.
- Facilitate creation of other **design artifacts**:
 - UML object models
 - Inference rules

References I

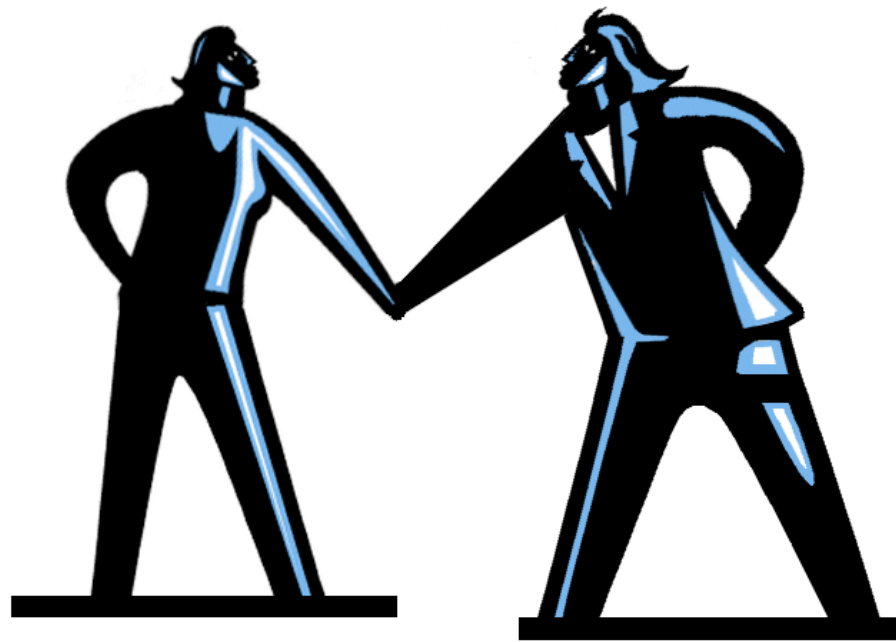
- **Boicu M.** et. al., "Ontologies And The Knowledge Acquisition Bottleneck", Proceedings of IJCAI-2001 Workshop on Ontologies and Information Sharing, pp. 9-18., Aug 2001.
- **Fensel, D.** et al., "On-To-Knowledge: Ontology-Based Tools For Knowledge Management", Proceedings of the eBusiness and eWork 2000 (EMMSEC 2000) Conference, Madrid, Spain, 2000.
- **McCarthy, J.**, "Some Expert System Need Common Sense", Computer Science Department, Stanford University, found at <http://www-formal.stanford.edu/jmc/>, 1984.
- **Noy, N.; McGuinness, D.**, "Ontology Development 101: A Guide To Creating Your First Ontology", Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001.

References II

- **Parry, D.**, "Fuzzification Of A Standard Ontology To Encourage Reuse", Proceedings of 2004 IEEE International Conference on Information Reuse and Integration, pp. 582-587, 2004.
- **Staab, S. et. al.**, "Knowledge Processes and Ontologies", IEEE Intelligent Systems, pp. 26-34, Jan/Feb 2001.
- **Uschold, M.; Gruninger, M.**, "Ontologies: Principles, Methods, And Applications", Knowledge Engineering Review, vol. 11, no. 2, pp. 1-63, Jun 1996.
- **Weber, R; Kaplan, R.**, " Knowledge-Based Knowledge Management", Innovations in Knowledge Engineering, vol. 4, pp. 151-172, Jul 2003.

**The entire archive of
ontology papers used in
this presentation is
available as a ZIP**

Please ask me or James
Owens if you'd like a copy.



**Thank
you all for
kind your
attention!!**

15 Minutes for Questions

