Knowledge Engineering

Prof. Mauro Gaspari
Dipartimento di Informatica Scienze e Ingegneria (DISI)
mauro.gaspari@unibo.it

Definitions

- **Knowledge engineering (KE)** is the general process of knowledge base construction:
    - Investigating a particular domain.
    - Learns, together with a domain expert (if needed), the concepts, the processes and the goals of the domain.
    - Represents the knowledge of a domain as objects and relationships among then.
- The knowledge engineer should learn the knowledge of the domain in a process called **knowledge acquisition:** interviews to real experts and, if necessary, studying relevant literature
- This occurs prior to or interleaved with, the process of creating formal representations.

Observations

- One does not become a proficient knowledge engineer just by studying the syntax and semantics of a representation language.
- It takes practice and exposure to lots of examples before one can develop a good style in any language, be it a language for programming, reasoning, or communicating.
- The question of efficiency becomes more critical.
    - The separation between the knowledge base and the inference procedure should be maintained.
    - Ensuring efficient inference is the task of the designer of the inference procedure and should not distort the representation.

The KE process

- **Identifying the task**: possibly interacting with domain experts, to evaluate if the problem can be effectively solved with a KR system.
- **Knowledge acquisition**: extract the relevant knowledge of the domain, in general with the help of domain experts.
    - Delineate the questions that the knowledge base will be able to answer.
    - Individuate the relevant knowledge and where this knowledge is used in the reasoning process.
- **Define the vocabulary**: of predicates, functions and constants of the domain with the associated intended meaning, enabling the expert to check the content.
- **Encoding of knowledge**: the knowledge engineer, enabling the expert to check the content:
    - Represents the knowledge of the domain with one or more KR formalism.
    - Encodes specific problem instances.
- **Test and Debug the knowledge base**: test the KB and used inference engines checking if the answers are correct with respect to the intended meaning.

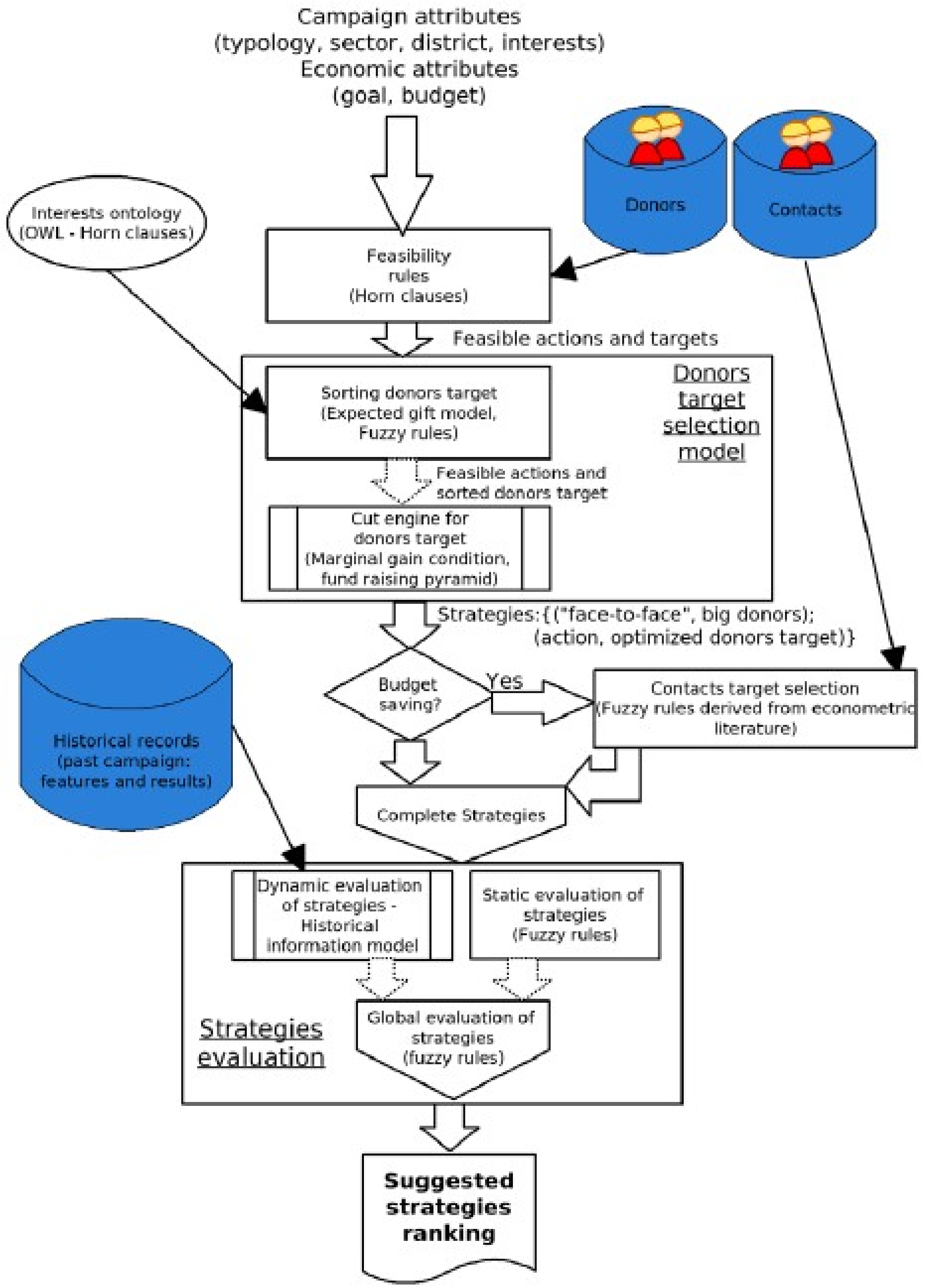
Paradign shift

- From**: Knowledge Engineering as a Transfer Process**: transformation of problem-solving expertise from a knowledge source to a KB.
- To: **Knowledge Engineering as a Modeling process**: Building a KB system means building an intelligent system model with the aim of realizing problem-solving capabilities comparable to a domain expert.
    - Using standardized problem solving methods and tools.
    - Using the right KR tools in the different phases of the problem solving activity (including ML tools).
    - It is a **cyclic process**: new observations may lead to a refinement, modification, or completion of the already built-up model and the further acquisition of knowledge

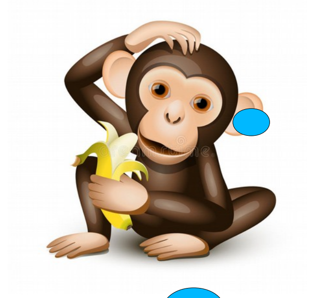
Knowledge Aquisition

- Observe the expert solving real problems. Also, have the expert solve a series of problems verbally and ask the rationale behind each step.
- Through discussions, build a schema of the problem solving approach used by the expert and identify sub-problems.
- Through discussions, define the used terminology.
- Through discussions, identify the kinds of data, knowledge and procedures required to solve each sub-problem.
- Identify, if some of the subproblems can be addressed with more adequate technologies.
- Refinement is usually necessary during the KE process.

Fund-raising Management

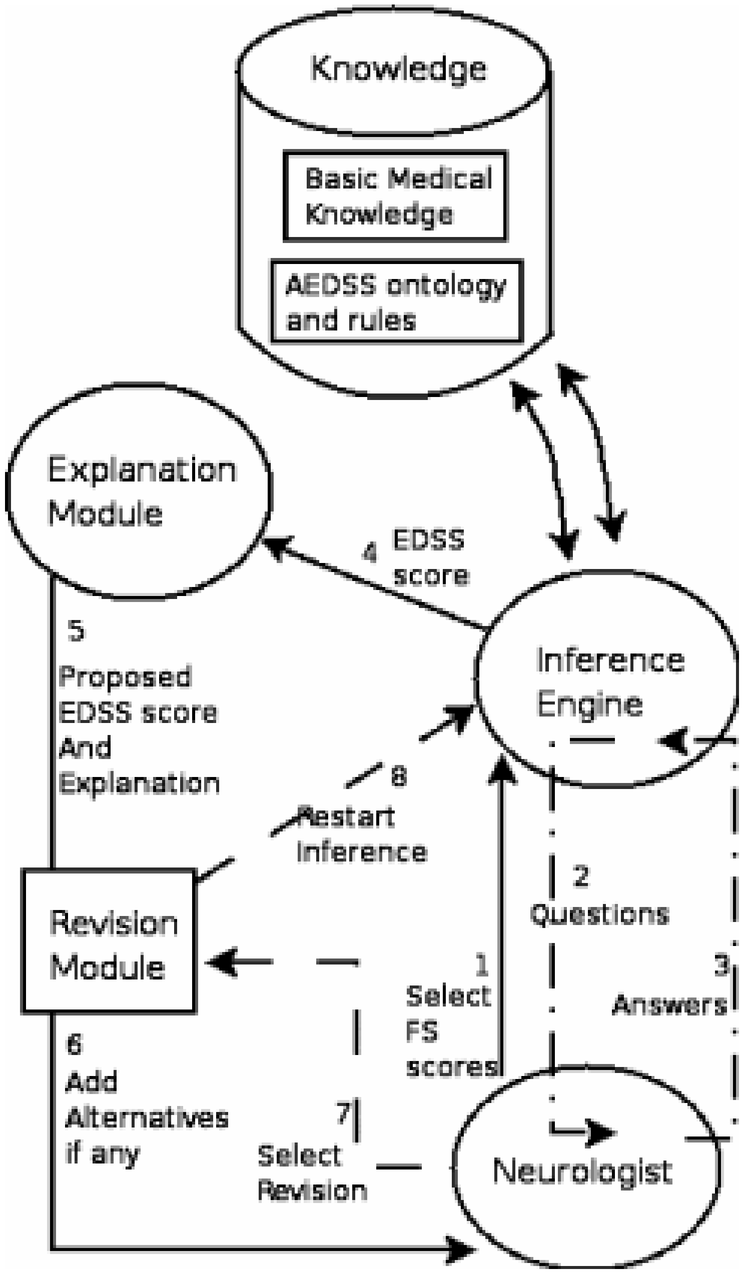


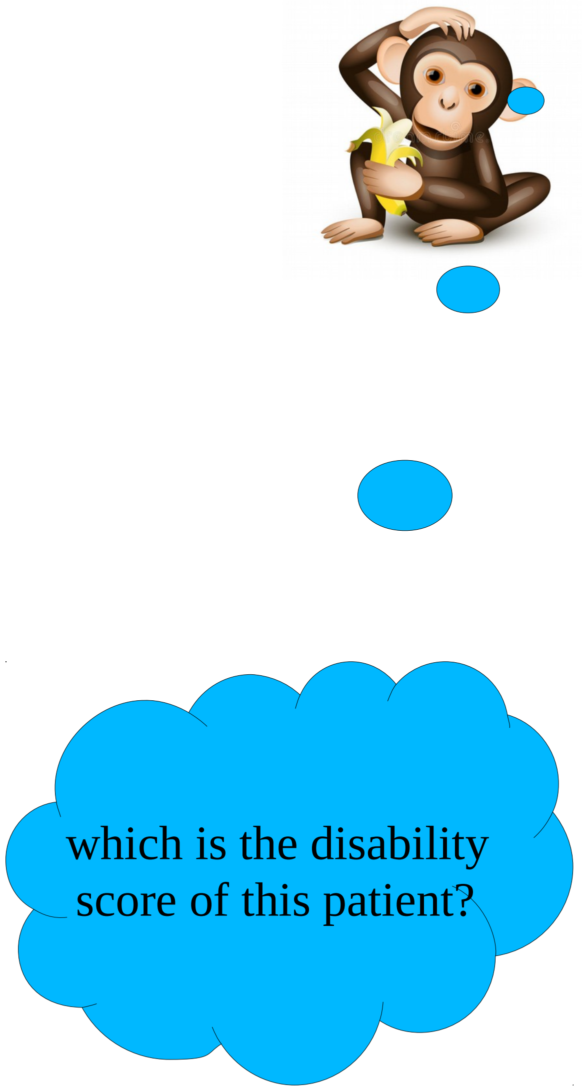
Campaign attributes (typology, sector, district, interests) Economic attributes (goal, budget)
Donors
Contacts
Interests ontology (OWL - Horn clauses)
Feasibility rules (Horn clauses)
Feasible actions and targets
Sorting donors target (Expected gift model, Fuzzy rules)
Donors target selection model
Feasible actions and sorted donors target
Cut engine for donors target (Marginal gain condition, fund raising pyramid)
Strategies:{("face-to-face", big donors); (action, optimized donors target)}
Historical records (past campaign: features and results)
Budget saving?
Yes
Contacts target selection (Fuzzy rules derived from econometric literature)
Complete Strategies
Dynamic evaluation of strategies - Historical information model
Static evaluation of strategies (Fuzzy rules)
Strategies evaluation
Global evaluation of strategies (fuzzy rules)
Suggested strategies ranking



I would like to start a fund raising campaign to build a school in Africa.
What should I do?

EDSS Evaluation





AI based Software Engineering

- Role of AI in the software development process vs Role of AI in the design process.
- AI based application design.
    - Emphasis on problem solving activities.
    - Knowledge oriented process design (which knowledge where?)
    - Use of standard software technologies.
    - Use AI in specific components.
    - Use the right AI tool.

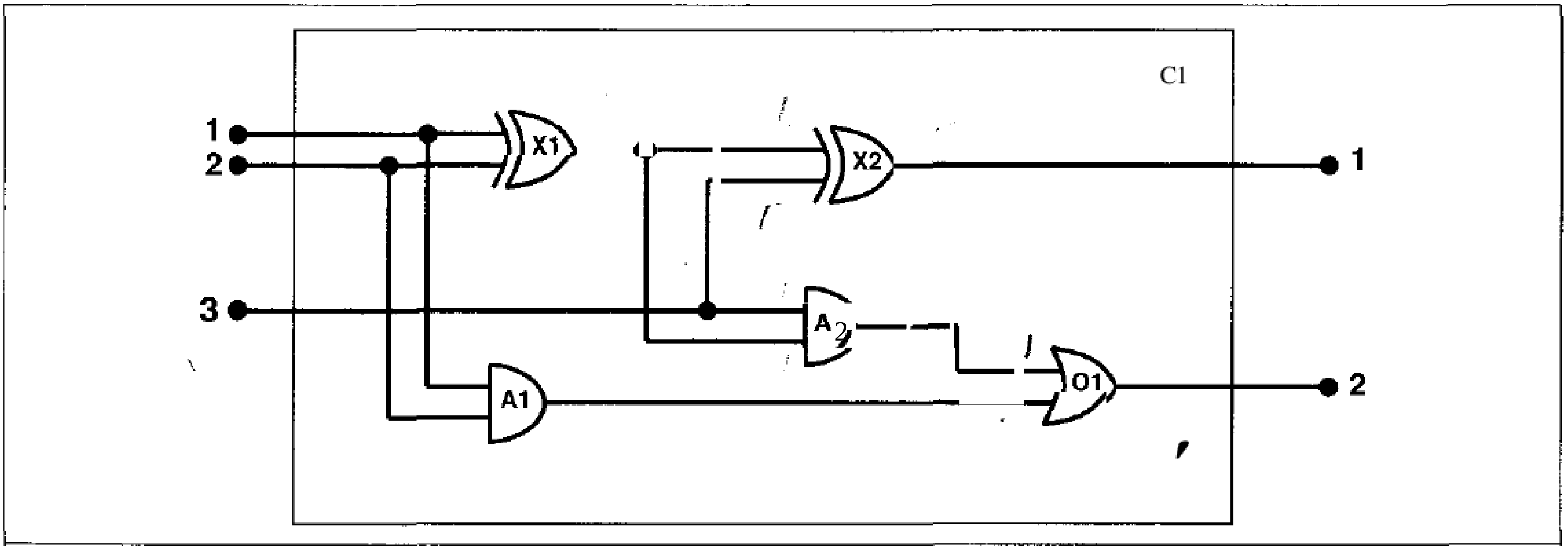
KE and programming

- A useful analogy can be made between knowledge engineering and programming for all the KB components.
- Both activities can be seen as consisting of four steps:

I prefer KE,
it is amazing!

	Knowledge Engineering	Programming
1.	Choosing a KR formalism	Choosing a programming language
2.	Building a KB	Writing a program
3.	Choosing an engine	Choosing a compiler
4.	Inferring new facts	Running a program

Electronic Circuits



- The circuit is one-bit full adder, where the first two inputs are the two bits to be added, and the third input is a carry bit. The first output is the sum, and the second output is a carry bit for the next adder.
- The goal is to provide an analysis that determines if the circuit is in fact an adder, and that can answer questions about the value of current flow at 3 various points in the circuit.

Define the vocabulary

- There are four types of gates: AND, OR, and XOR gates have exactly two input terminals, and NOT gates have one. All gates have exactly one output terminal. Circuits, which are composed of gates, also have input and output terminals.
- The purpose is to prove if the design of circuits match their specification. We need to represent:
    - Circuits
    - Their terminals, and the signals at the terminals.
    - Individual gates, and gate types: AND, OR, XOR, and NOT.

Terms and Predicates

- Using equality =, different encoded as not equal.
- A↔B is defined as A→B ∧ B→A
- Gates: X1,X2,A1,A2,O1,O2,N1,N2 where the first letter represents the type of the gate, it is just a naming convention.
- Gates types: XOR,AND,OR,NOT
- In(N,Gate)==> A terminal that represents the input N of Gate.
- Out(N,Gate)==> A terminal that represents the output N of Gate.
- Signal(Terminal)==>a function that associates a value in {0,1} to a terminal.
- Type(Gate,Gtype)==> This predicate states that Gate has type Gtype.
- Connected(D1,D2)==> This predicates represents connected Out and In.

General Rules

1. If two terminals are connected, then they have the same signal:
   $\forall t1,t2\ Connected(t1,t2) \rightarrow Signal(t1)= Signal(t2)$

2. The signal at every terminal is either on or off (but not both):
   $\forall t\ Signal(t) =1\ \ v\ Signal(t) = 0$
   $\neg(1=0)$

3. Connected is a commutative predicate:
   $\forall t1,t2\ Connected(t1,t2) \leftrightarrow Connected(t2,t1)$

4. An OR gate's output is On if and only if any of its inputs are On:
   $\forall g\ Type(g,OR) \rightarrow Signal(Out(1,g))=1 \leftrightarrow \exists n\ Signal(In(n,g))=1$

5. An AND gate's output is oOf if and only if any of its inputs are Off:
   $\forall g\ Type(g,AND) \rightarrow Signal(Out(l,g))=0 \leftrightarrow \exists n\ Signal(In(n,g))=0$

6. An XOR gate's output is on if and only if its inputs are different:
   $\forall g\ Type(g,XOR)\rightarrow Signal(Out(1,g))=1 \leftrightarrow \neg(Signal(In(1,g)=Signal(In(2,g)))$

7. A NOT gate's output is different from its input:
   $\forall g\ (Type(g,NOT)) \rightarrow\ \ \neg(Signal(Out(1,g)) = Signal(In(1,g)))$

Gates and Connections

- **Gates types and axiom:**
  Type(X1, XOR)
  Type(X2, XOR)
  Type(Al, AND)
  Type(A2,AND)
  Type(O1,OR)
  ∀g∃t1,t2 Type(g,t1)∧Type(g,t2)→ t1=t2

- **Connections:**
  Connected(Out(1,X1),In(1,X2))
  Connected(Out(1,X1),In(2,A2))
  Connected(Out(1,A2),In(1,O1))
  Connected(Out(1,A1),In(2,O1))
  Connected(Out(1,X2), Out(1,C1))
  Connected(Out(1,O1), Out(2,C1))
  Connected(In(1 C1),In(1,X1))
  Connected(In(1,C1),In(1,A1))
  Connected(In(2,C1),In(2,X1))
  Connected(In(2,C1), In(2,A1))
  Connected(In(3,C1),In(2,X2))
  Connected(In(3, C1),In(1,A2))

Goals

- What combinations of inputs would cause the first output of C1 (the sum bit) to be Off and the second output of C1 (the carry bit) to be On?

  ∃i1,i2,i3 Signal(In(1, C1))=1 ∧ Signal(In(2,C1))=i2 ∧ Signal(In(3,C1))=i3 ∧ Signal(Out(1,C1))=0 ∧ Signal(Out(2, C1))=1

- What are the possible sets of values of all the terminals for the adder circuit?
- ∃i1,i2,i3,o1,o2 Signal(In(1,C1)) = i1 ∧ Signal(In(2, C1)) = i2 ∧ Signal(In(3,C1)) = i3 ∧ Signal(Out(1, C1)) =o1 ∧ Signal(Out(2, C1)) = o2
- Circuit verification: This final query will return a complete input/output table for the device, which can be used to check If it works as expected.

Limitations of FOL

- The advantage with respect to the Prolog approach is that there is a clear distinction between the KB and the engine.
- However, the reasoning process is more complex.
- Moreover, certain aspects of the real world are hard to capture in FOL:
    - All the generalizations have exceptions.
    - Same assertions are uncertain, they hold only to a given degree.
- As a result, more specific formalism are needed for many domains, providing efficient tools for specific representation needs.

Exercises

1. Encode the circuit example in z3.
2. Implement the circuit example in Prolog.

References

- Rudi Studer, Richard Benjamins, Dieter Fensel. Knowledge engineering: Principles and methods. Data & Knowledge Engineering Volume 25, Issues 1–2, March 1998, Pages 161-197.
- Steels, Luc L.. “Components of Expertise.” AI Magazine 11 (1990): 28-49.
- M. Gaspari, D. Saletti, C. Scandellari, S. Stecchi. Refining an Automatic EDSS Scoring Expert System for Routine Clinical Use in Multiple Sclerosis In IEEE Transactions on Information Technology in Biomedicine. 13(4):501-11 2009.
- L. Barzanti, M. Gaspari, D. Saletti. Modelling decision making in fund raising management by a fuzzy knowledge system. In Expert Systems With Applications 46(2) 2009.