Reasoning about Hand-Drawn Sketches: An Approach based on Intelligent Software Agents

V. Deufemia, G. Tortora

- Dipartimento di Matematica e Informatica
Università di Salerno

G. Casella, V. Mascardi, M. Martelli

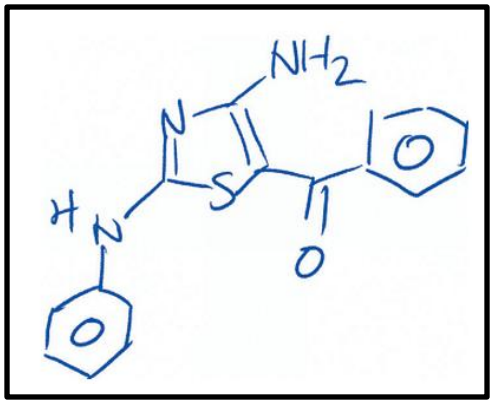
- Dipartimento di Informatica e Scienze dell'Informazione
Università di Genova

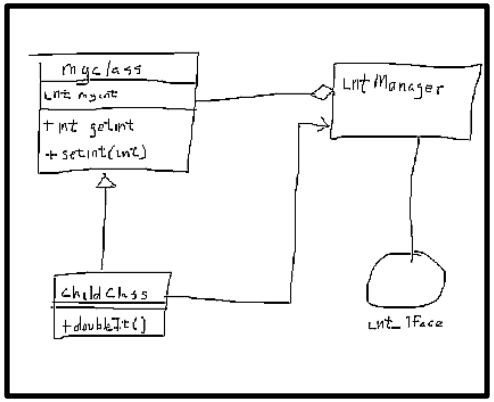
Outline

- Motivation
- The agent based framework
- User Reasoning support
- Building’s safety domain
- Conclusions and Future Work

Motivation: Sketching

- The ability to sketch a diagram and to recognize it automatically is a great advantage allowing:
  - to save time and paper,
  - to share the diagram,
  - to teach how a diagram should be correctly drawn,
  - to archive and retrieve it in an electronic form.





Motivation: Sketch Reasoning

- However, there are situations where the user's needs go far beyond the use of a software system just for recognizing symbols in the correct way. For example,
  - checking that a room or a building respects all the safety criteria
  - it might help a chemical engineer in reasoning on chemical reactions

Sketch Reasoning

- A software system must integrate capabilities coming from three research domains:
- automatic hand-drawn sketch recognition domain
  - to recognize hand-drawn symbols, and to detect and resolve conflicts among their interpretation
- geometric modeling domain
  - to model physical objects and to reason about spatial relationships among them
- artificial intelligence domain
  - to act as a “pro-active” and “situation aware” expert system.

Intelligent Agents

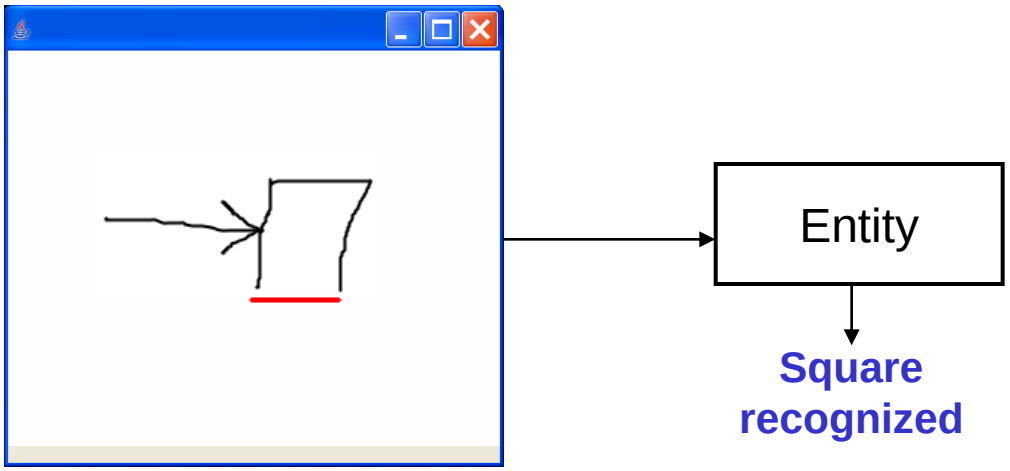
- *Agent*: A computer system that is capable of flexible autonomous action in dynamic, unpredictable, typically multi-agent domains.
  [AgentLink III Technology Roadmap]

- Agents should be
  - responsive
  - pro-active
  - social
  - situated in some environment

  M. Wooldridge and N. R. Jennings. “Intelligent agents: Theory and practice”, The Knowledge Engineering Review, 1995.

Our Proposal: Recognition

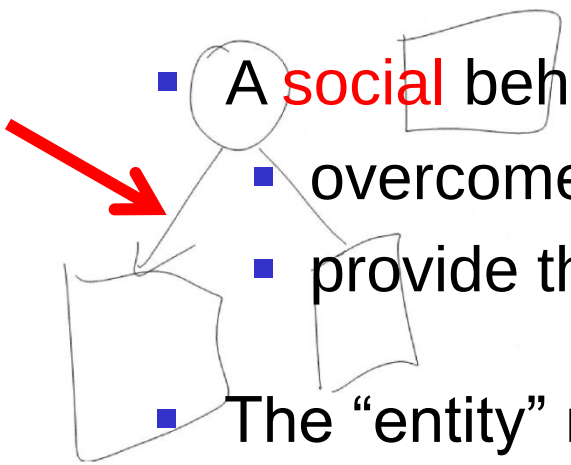
- The “virtual blank sheet” represents a dynamic and unpredictable environment
- An “entity” devoted to recognizing a specific symbol must be situated in it



- reacts to changes that take place in the virtual blank sheet
- has a complex long term goal
- operates in an autonomous way to reach this goal
- has a social behavior to overcome conflicts and ambiguities

Our Proposal: Recognition

- By working alone an entity cannot easily resolve ambiguities and conflicts
- A social behavior is required to
  - overcome conflicts and ambiguities, and
  - provide the right interpretation of the sketch.
- The “entity” must be responsive, pro-active, situated, autonomous, and social

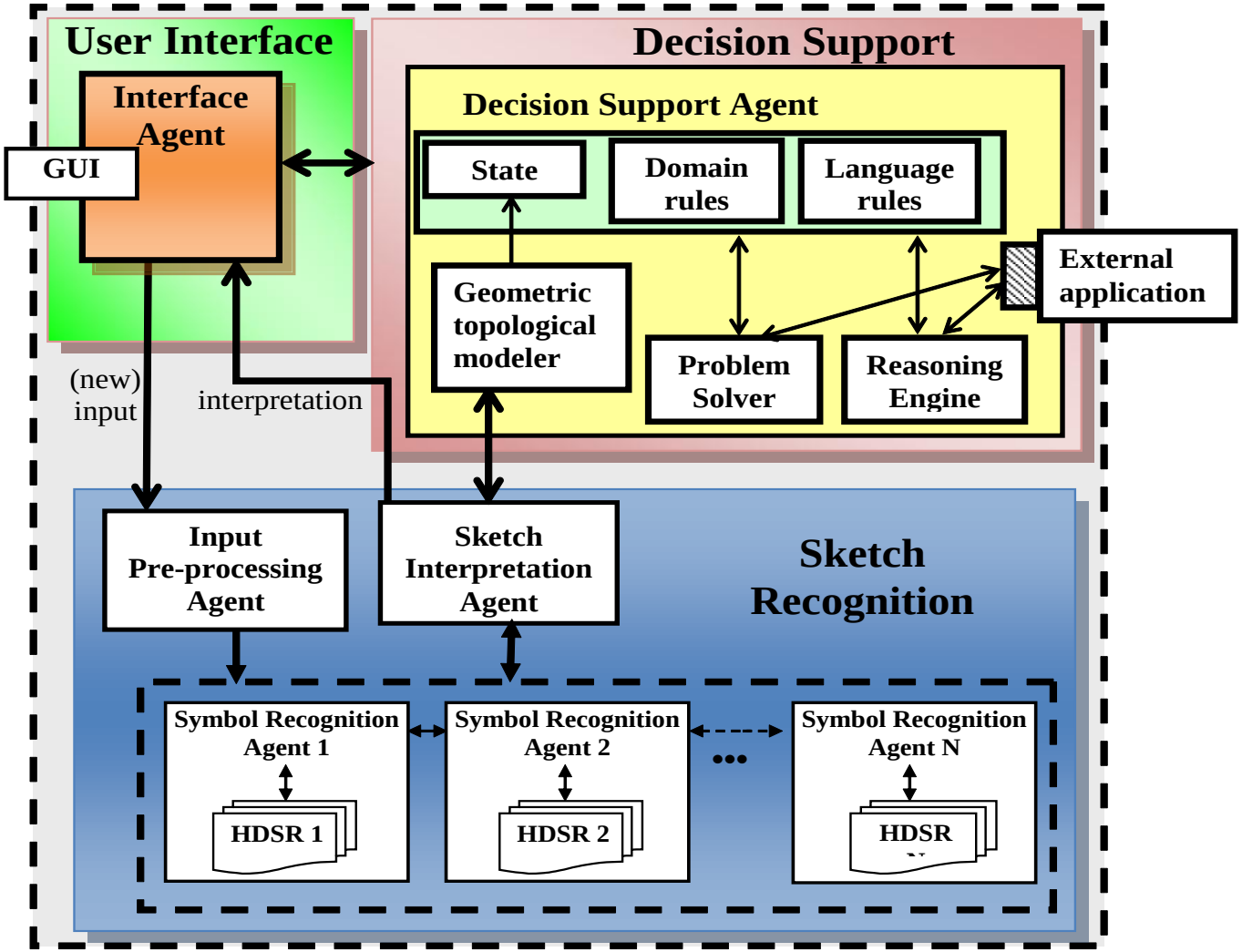


an intelligent agent

Our Proposal: Reasoning

- The Reasoning module is an agent as well equipped with
  - geometric modeling rules
  - application domain rules
- The agent pursues the goal of avoiding rule violation (pro-activeness).
- In case of violation, the agent looks for an alternative arrangement of the drawn objects (autonomy).
- If an “easy” solution cannot be found in a reasonable amount of time (reactivity), the agent starts interacting with the user for finding a solution (sociality).

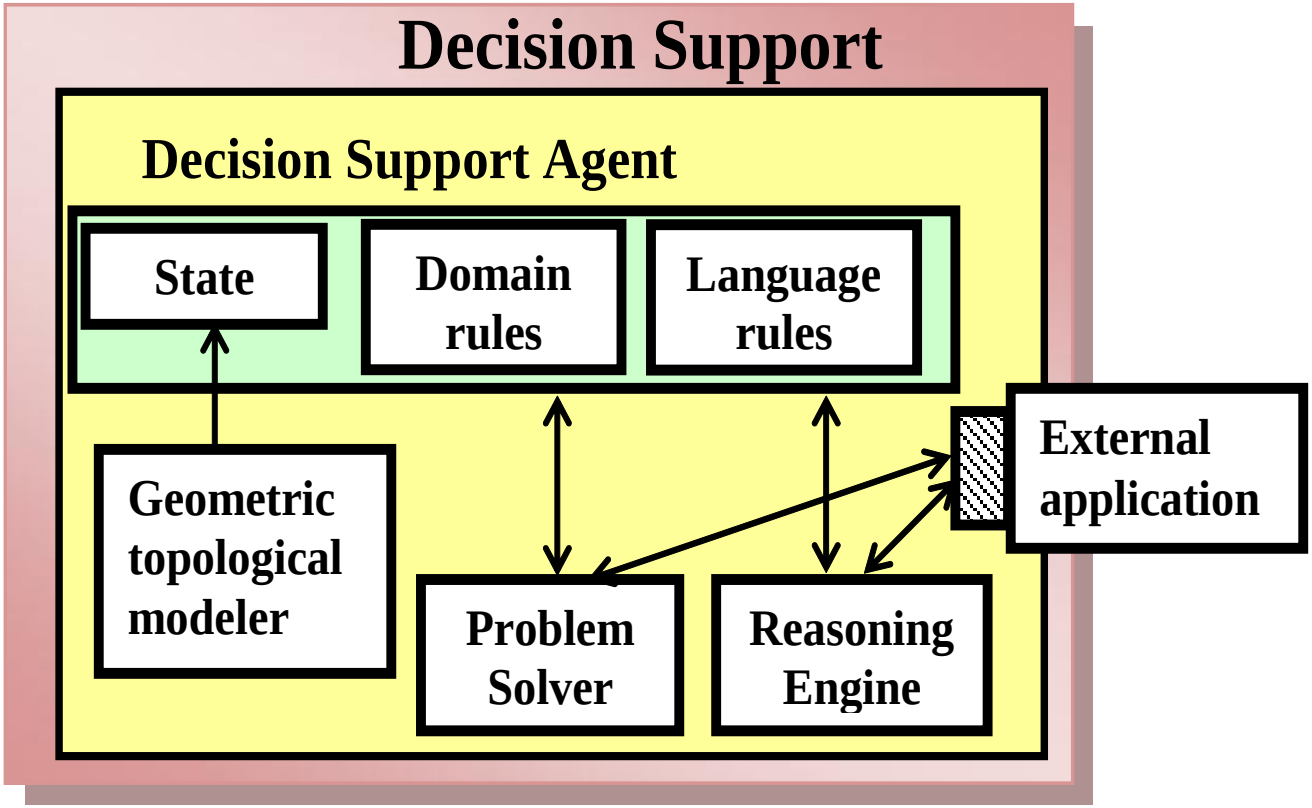
The Agent-based Architecture



Decision Support module

- Computing both simple and complex geometric and topological relations among symbols.
- Modeling language dependent constraints on symbols.
- Modeling domain dependent constraints on symbols.
- Verifying that all constraints are satisfied by the current sketch.
- Helping the user in finding alternative solutions.

Decision Support Architecture

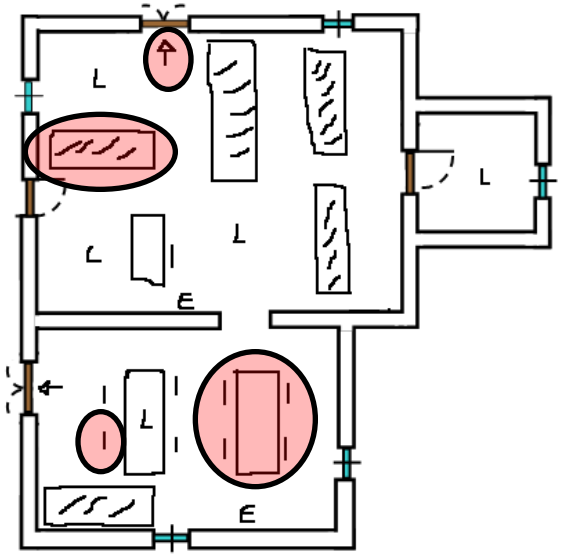


State and rules' representation

- The agent's state is represented by first-order logic atoms
  - *represents(S, Sym)*, where *S* is an identifier and *Sym* is one among the symbols of the language,
    - es. *has_bounding_box(S, BB)*;
- The relations between symbols are also represented as atoms: *on_the_left_of(S1, S2)*, *contained_in(S1, S2)*.
- The agent's rules are represented by using *deontic logic*, which provides operators for *permission*, *obligation* and *forbidden*.
  - Language rule: $\forall$ S1, S2 **F** *intersects*(S1, S2)
  - Electronic engineering domain:

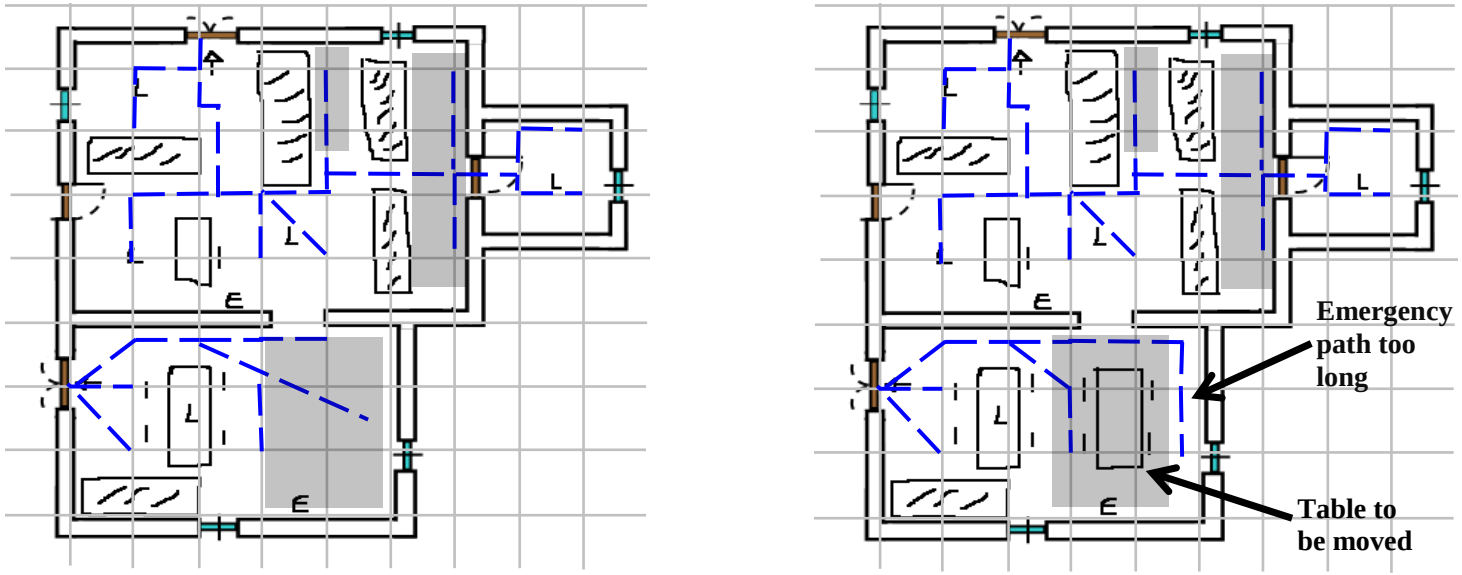
$$\forall S_1, S_2 \; ((represents(S_1, cpu) \wedge represents(S_2, motherboard)) \Rightarrow \mathbf{O}\; included\_in(S_1, S_2))$$

Checking safety criteria in buildings



- To face the problem of building's safety, the user has to sketch
- furniture having a height that could occlude the light (i.e., wardrobe, libraries etc.), represented by rectangles containing dotted lines;
- furniture that cannot obstruct the light (i.e., tables, desks, etc.), represented by empty rectangles;
- chairs represented by lines placed near low furniture;
- an arrow to specify an emergency exit, an "L" to represent a light, and an "E" to represent an extinguisher.

Checking safety criteria in buildings



- To support the user in finding the right placement of lights, furniture, and fire extinguishers, the DSA uses a couple of external modules to compute the emergency paths and the poorly illuminated areas.
- When the user draws a new symbol (or moves an existing one) the DSA checks if any rule is violated.

Conclusions and Future Work

- In this paper we have proposed a system to support user's reasoning through sketching by exploiting intelligent agents.

- Future work
  - we will focus on the completion of the system implementation
  - we will concentrate our efforts on finding appropriate solutions to present the information provided to the user, feedback and solutions.