Conceptual Foundations of Interrogative Agents

Vincenzo Deufemia, Giuseppe Polese, Genoveffa Tortora, Mario Vacca
Dipartimento Matematica e Informatica
Università di Salerno
84084 Fisciano(SA), Italy
{deufemia, gpolese, tortora, mvacca}@unisa.it

***Abstract*—Reasoning by interrogation is one of the most ancient and experimented ways of reasoning. Originated by the Aristotelian *elenchus*, it has been used for many purposes, such as the resolution of mathematical and daily problems [25], [26], the discovery of new knowledge [19], [34], [36], the realization of questioning/answering processes [23]. In this paper we present the conceptual foundations of interrogative agents, a new model of BDI architecture based on interrogative logic. This model allows us to express the properties of agents in a natural way, and to use heuristics for reasoning. Finally, in order to explicate the whole approach and to highlight its main features we describe the application of interrogative agents in the context of database refactoring.**

I. Introduction

In recent years many different agent-based architectures and models have been proposed [39]. In this context, a well known architecture is the so called *Beliefs - Desires - Intentions* (*BDI*, for short), based on the concept of practical reasoning, i.e., the capability of resolving, through reflection, the problem of what to do and how to do it. Informally, this architecture is composed of four data structures: *Beliefs* representing the information that the agent has about the world, *Desires* representing the tasks that the agent has to accomplish, *Intentions* representing the sequence of actions to achieve the agent's desires, and *Plans* representing the procedural knowledge or Know-how. An *interpreter* is responsible for managing these structures.

In the last years, one of the most important issue faced by the agent community is flexibility, i.e., the agents ability to act in unknown situations or to face new situations. Franklin *et al.* defined this feature as the agents ability to have non scripted actions [13]. Many authors stressed the importance of flexibility. For instance, Barklund *et al.* took into account the problem of agent's flexibility observing that the reuse of an existing database in a different context calls for the ability of bridging the differences between the representation of input and the internal one [3]. According to the same authors, agents should be able to use different portions of knowledge depending on users or question classifications. Furthermore, they maintained the importance of using non-classical or non-deductive forms of reasoning and proposed the enrichment of agents with meta-knowledge. From the application point of view, Lin *et al.* showed that web-agents need to manage incomplete and partial information [24]. Therefore, the problem of modeling flexible agents is more a general one, involving both the way agents use knowledge and how they reason.

In this paper we face the problem of finding a BDI-like architecture for flexible agents. To this end, it is important to remark that the problem of flexibility has already been faced in the artificial intelligence field, yielding several models of intelligent systems [33], [34], [36]. These are based on problem solving and on interrogation as the underlying reasoning mechanism. The latter is considered one of the most suitable reasoning mechanisms in order to solve problems [25], [26]. Interrogation is one of the most ancient and experimented methods of reasoning [17]. It originated by the Aristotelian *elenchus* and it has been used for several purposes. This method of interrogation contains questions other than affirmations. It starts the reasoning process with a question, trying to find a plausible answer to it, after producing a sequence of questions and affirmations.

This paper is a foundational work aimed at obtaining an architecture that provides a higher degree of flexibility than other existing ones. The proposed interrogative BDI architecture containing both questions and affirmations. Questions correspond to *stimuli* (request to think) [25], [26], [36], and thinking is always an interrogation of an information source [18]. In this model desires are represented through unanswered *background questions*, intentions are more *urgent questions* to ask for, and *beliefs* contain affirmations as well as previously answered questions. Finally, heuristics, like in the Polya conception, are meta-knowledge questions to help transforming a question into another one. A processing model of the agent (usually called *cycle* in the autonomous agent literature) will be represented by a logical game [17].

The paper is organized as follows: Section 2 discusses the foundations of our proposal, and describes two models of reasoning by questioning. Section 3 introduces interrogative logic as the formalism underlying the proposed model. In Section 4 we describe the architecture of interrogative agents, whereas in Section 5 we describe the application of the proposed model to the problem of database refactoring. Finally, conclusions and further research are discussed in Section 6.

II. Conceptual Foundations of Interrogative Agents

In the last decades several intelligent systems have been presented and many models based on the interrogation paradigm have been developed [4], [19], [25], [29], [36]. According to the analysis of Jung concerning systems for scientific inquiry [19], it is possible to classify these models in two main categories. The first one contains the models that are limited to the linguistic and logical aspects of questions and answers and to the relation question-answer (answerhood). The latter contains the models that deal with the methodological aspects of the question-answering process (*Q/A process*, for short) and, therefore, whose aim is to build interrogative-dialogical schemes. These models, named *I-D models*, are descendant from the Greek dialogical scheme and are based on four essential concepts: the dialogue is a *game*; the moves of the players are: *questioning*, *answering*, and *reasoning*; the aim of dialogic is to build well-organized *sequences of questions*; dialogical reasoning is useful to *discover* new hypothesis, theories, or general assumptions [19].

As observed by Jung "The dialogical scheme has some unique features that are ideal for the logic of discovery" [19]. The most important feature is that it can include non inferential moves (*heuristics*) in a natural way, since this kind of inferences are crucial in the context of discovery and problem solving. Another important feature is that I-D models are goal-oriented and the goal is expressed as a question to be answered. One problem with dialogical models is that dialogic is a content logic and, hence, domain-specific. In fact, the criticisms to these models in the A.I. field are related to the lack of an underlying formalism [21].

In the following we describe two main models of intelligent systems which reason by questioning: the model of Schank and the one of Polya. Both can be viewed as agents' models, and therefore they constitute a foundation for an interrogative model of agents.

A. The Schank model of agent

The Schank group work [31]–[36] is a very large and long lasting, influencing or leading to different branches, trends and areas of research (see for example [1], [20], [22], [27]).

The Schank's theory models an intelligent (human or machine) agent able to learn occurring events and to devise plans for achieving goals. The agent is a problem solver one and its main reasoning tools are questions and interrogation. In the following, we show that the Schank's theory can be seen as a BDI model with an underlying interrogative reasoning.

1) The knowledge structures: There are different kinds of structures depending on the representation level: the actions are represented by conceptual dependencies [31], scripts and scenes are devoted to represent more complex situations [35], MOPs (Memory Organization Packets) and meta MOPs organize the high level knowledge [32], [33].

The theory of conceptual dependency (*CD theory*, for short) is a pictorial formalism developed for representing complex events through elementary ones. A *CD representation* of an event (also called *conceptualization*) is composed of objects linked together by rules. As an example, the conceptualization

$$\begin{tikzcd} & & & & Israel \\ Vance \arrow[r, Leftrightarrow] & PTRANS & Vance \arrow[l, "o"'] & \arrow[l, "R"'] \arrow[ur] \arrow[dr] & \\ & & & & USA \end{tikzcd}$$

$$Diplomat \Longleftrightarrow Vance$$

means that the diplomat Vance goes from USA to Israel.

Scripts were introduced to model the daily life stereotypical situations, such as "eating in a restaurant" or "taking a plane" [35]. Scripts are frame-like knowledge structures and represent prototypical knowledge. They contain sequences of *scenes* involving a set of objects (*Props*) and a set of people (*Roles*). Scenes contain general actions aiming to reach the same goal. Entry conditions enable to activate scripts, whereas exit conditions give the status of *Roles* and *Props* after the script has been executed (used). For example, a script $DIPLOMATIC_VISIT could involve state secretaries and translators as *Roles*, and airplane and tables as *Props*. It can contain the scenes ARRIVAL and STATE_DINNER describing, respectively, what generally happens when diplomats *arrive* at a foreign state and what are the typical actions and states during a state dinner.

The model of Schank has also *goals* and *plans* [35], which are script-like structures. Indeed, both scripts and their successor MOPs can be considered as orderers of scenes and can also be used as plans [33].

In conclusion, the Schank model contains all the elements of a BDI architecture.

2) The tasks of Schank agents: Since the theory of Schank was born in the area of natural language processing, the task of the first Schank "agent" was *understanding* the occurring events. The first paradigm used was "*understanding as finding a place in the memory*" for the occurring events (e.g., see the systems MARGIE [31], SAM [35], or FRUMP [8]).

This first Schank model was very simple. In fact, in the light of theory of autonomous agents, the Schank understander was scantly autonomous, it had little or none ability to react to unknown stimuli, and it processed the same input always in the same way (i.e., it did not change through its experiences). To overcome these limitations, it was introduced the concept of memory reorganization (a form of *belief revision*), which enables the understander to learn by experience and, therefore, to react in a more timely way to external stimuli [32], [33]. In [34], [36], nevertheless, it was recognized that a serious drawback of the model were still the lack of *flexibility*, that is the attitude to change both its knowledge and its behavior to face the stimuli coming from the environment. The rigidity in the behavior was due to the systematic use of scripted activities (see Schank [36] for criticisms on script-based system). The introduction of *problem solving* in the Schank model made the systems more flexible since it enables to solve problems using the previously solved problems and stored knowledge, but also reasoning (i.e. applying both deduction and heuristics) or communication (asking other information sources for), and, hence, anomalous input or situations are treated like problems. Thus, a problem solving system trying to solve a problem will not stop the processing. In this way any activity becomes the solution to a problem which may not have a standardized solution.

3) The processing: The reasoning tools of the Schank theory are based on questions. Indeed, understanding is realized through the application of a structure of knowledge, which is a process consisting in posing a set of questions on what could occur. The flexibility is fulfilled substituing the ”filling the slots” concept of understanding by the more complex process of explanation, whose aim is to link anomalous inputs to the existent knowledge. In particular, the main phases of the explanation process [34], which embodies all the features of the questioning process, can be compared to the *cycle* of agent architectures:

a) *Finding an anomaly*
   This process starts with the application of the base questions, such as the scripts and the scenes of MOPs. If any of these questions do not obtain an answer, or the answers are different from the forecast ones, then an anomaly has been found and the system needs additional creative explanations.
b) *Posing the explanation question – Finding the explanation pattern*
   This process starts when an anomaly has been detected. The explanation question (*EQ*, for short) tries to explain the understanding failure by reformulating the question, and, if there exists an answer, it will be used as a new expectation.
c) *Reorganizing the memory*
   If there exists an answer to an EQ, the system tries to generalize and story it in memory. This is done by reminding similar cases, and by attempting to find a suitable generalization. Both reminding and generalizing are accomplished through an interrogative process.

Thus, the reasoning starts with a question and generates new ones as variations of existing questions, i.e., are obtained from another one by *transformation* which “is a way of getting an answerable question from an unanswerable one” [36], p. 287. Typical transformation mechanisms are specialization, generalization, simplification, and every one enabling to get an answer. Once an answer for the transformed question is found, the tweaking process will try to adapt it to the starting question.

Compared with the first approach based on the “question as expectation” paradigm, here the questions have an active role since they start elaboration processes and, hence, the association question-answer is dynamic.

The following very famous example about the meeting between Vance and Begin wives shows the working of the interrogative reasoning proposed by Schank:

“Q1. Did your wife ever meet Mrs. Begin?

  Q2. Where would they have met?

    Q3. Under what circumstance do diplomats’ wives meet?

      Q4. Under what circumstance do diplomats meet?

      A4. On state visits to each other’s countries.At international conferences.

    A3. When they accompany their husbands on these visits.

    Q3a. When did Vance go to Israel?

    Q3b. When did Begin go to the U.S.?

    A3a/A3b. Various dates can now be retrieved from memory.

    Q3c. Did their wives accompany them on any of these trips?

    A3c. A trip where this happened is found.

  Q2a. During what part of a trip would wives meet?

  A2a. During a state dinner.

Final revised question: Was there a state dinner on may 24th, 1977, during the diplomatic visit that Vance made to Israel with his wife?

Answer (A1): Probably on May 24, 1977, in Jerusalem at the state dinner at which they were both present.” [35] pag. 286.

B. The Polya problem solving process

The model of problem solver proposed by Polya can also be described in terms of a BDI architecture [25], [26].

A problem solver requires two kinds of *knowledge*: one about abstract problems and the other about concrete (already solved) problems. Actually, the first one is a meta knowledge about the structure and the components of the problems.

The *cycle*, whose goal is to link the given problem to the existing knowledge in a closer and closer way, can be described by two lines going from the $Mobilization$ of knowledge to its connection to the problem ($Organization$) [26]:

1) $Mobilization \rightarrow Recognizing \rightarrow Regrouping \rightarrow Organization$
   Recognizing consists in examining the problem and *recognizing* some familiar features, whereas regrouping refers to a new way to see the existing elements after having recognized them. For example, the drawing of a bisector of the vertex angle in an isosceles triangle enables the regrouping of its elements in two equal triangles.
2) $Mobilization \rightarrow Remembering \rightarrow Supplementing \rightarrow Organization$
   When a feature has been recognized, it can allow to *remember* other problems with that feature or theorems about it. This new knowledge enriches the problem *supplementing* it by this new information.

The cycle is realized by questions. For instance, questions like “What is the unknown?”, “What are the data?”, “What is the condition?” can be useful for recognizing; the questions “Do you know a related problem?” or “Do you know a theorem that could be useful?” can help for remembering; “Could you restate the problem?”, “Could you restate it still differently? Go back to definitions.” aim to regroup (see [25] for a very rich list of questions).

However Polya warns that “Do not, however, use the checklist in a haphazard way, taking the questions at random, and do not use it mechanically, going through the questions in a fixed order. Instead, use this list of questions as an expert workman use his *tool chest*.” [26].

III. The interrogative logic

A serious problem with models that reason by questioning, like the Schank's one, is the lack of underlying formalisms. In fact, it is well-known that the debate on the role of mathematical logic in artificial intelligence has characterized the developments of A.I. itself, producing currents of thought and different positions [21]. After the recent developments, we think that erotetic logic can constitute an adequate formalism for the models based on interrogation.

Erotetic logic (from the Greek word erotema meaning question) is the branch of logic studying the logic of questions and answers. According to the *received view* [5], [37], the task of erotetic logic is twofold: studying formal languages containing both questions and answers, and studying the relations between question and answer (answerhood). Nevertheless, many authors think that it is possible to reason (making inferences) by using both questions and answers [15], [17], [37], [38], yielding models close to the concept of Greek dialogic (the logic of reasoning by interrogation), as they recognize the importance of reasoning by questioning [19].

In this paper we abide by this conception of erotetic logic, which we refer to as dialogic, in order to avoid confusion with the received view. In particular, we think that the theory of Hintikka can be exploited to derive the logic formalism of our model [17].

Hintikka view of reasoning can be summerized as follows: a line of reasoning is constituted by a sequence of sentences; a new sentence in such a line is either obtained by deduction or (in the case of a rational agent) by asking for an information source (named *oracle* in the Hintikka terminology). Such rational line of reasoning is an interrogative process, which is performed by an *inquirer* [17].

The logic of Hintikka is modelled by a game played by the inquirer against one or more oracles, and the semantics used is the tableau method. Affirmations are on the left side of the tableau, the questions on the right side. The inquirer starts the moves, and the role of the oracles is to answer to the inquirer's questions. There are two kinds of moves: *logical inference moves* and *interrogative moves*. The formers are the typical deductive rules, and they are tableau-building rules (see [17] for a complete list).

Rules for questioning serve to generate questions. A rule for questioning is the following:

- If the presupposition of a question occurs on the left side of a subtableau, the inquirer may address the corresponding question to the oracle. If the oracle answers, the answer is added to the left side of the subtableau.

The system also has structural rules, which allow to manipulate the tableau.

IV. Interrogative Agents

The proposed model abides by the tradition of I-D models and is presented using some of the terminology by Jung [19].

Our ideal agent is Sherlock Holmes, whose behavior has been widely studied by philosophers and logicians (see [11] for a collection of essays). Sherlock Holmes is a problem solver with a diversified knowledge[1] and who is able to reason analytically[2]. Any activity starts with a problem (question) and it possibly ends with a plausible answer to that question. In order to solve a problem, the agent activates a *process of problem solving* which consists of linking the problem to the existing knowledge. According to the analytic method [6], a problem P is reduced to another problem that, if it is known or built, solves P. This is made trying to answer by reasoning or asking some other (also itself) for. Asking to itself means to search its own memory for some information.

Therefore, the agent has two abilities, *deducing* and *asking for/answering* (*communicating*). While deducing is a process of argumentative bridge-building, communicating serves to ask for other information sources. Also the task of an information source (named *oracle* in the Hintikka's terminology) is to answer questions, but the strategy used to accomplish it does not matter. Hence, an oracle can represent a mathematician using reasoning heuristics, an experimental physicist answering a question on the ground of experimental data, or it can implement vision recognition modules in computer systems.

A. The model

Formally, an *interrogative agent* is a quadruple

$$IA = (I, K, L, R_G)$$

where

I is an *interpreter*;

K is a set of *Information sources*. Three special sources of information are the *Problem source* P, the *Environment source* E, and the *Goal source* G;

L is the interrogative logic of the agent. It has a language constituted by two disjoint sets: Q (the set of possible *questions*) and A (the set of possible affirmations), and by rules like those in [17];

R_G is a set of rules (*guidance* rules).

The interpreter I has the goal to answer questions (called *principal questions* or *main problems*) according to the guidance rules R_G. To this aim it applies deductive rules or asks for another information source.

[1]"SHERLOCK HOLMES – his limits. 1. Knowledge of Literature. – Nil. 2. Philosophy. – Nil. 3. Astronomy. – Nil. 4. Politics. – Feeble. 5. Botany. – Variable. Well up in belladonna, opium, and poisons generally. Knows nothing of practical gardening. 6. Geology. – Practical, but limited. Tells at a glance different soils from each other. After walks has shown me splashes upon his trousers, and told me by their color and consistence in what part of London he had received them. 7. Chemistry. – Profound. 8. Anatomy. – Accurate, but unsystematic. 9. Sensational Literature. – Immense. He appears to know every detail of every horror perpetrated in the century. 10. Plays the violin well. 11. Is an expert singlestick player, boxer, and swordsman. 12. Has a good practical knowledge of British law." [10] pp. 13–14.

[2]"In solving a problem of this sort, the grand thing is to be able to reason backwards. That is a very useful accomplishment, and a very easy one, but people do not practice it much. In the every-day affairs of life it is more useful to reason forwards, and so the other comes to be neglected. There are fifty who can reason synthetically for one who can reason analytically." [10] pp. 115–116.

An information source is a couple (K_i, O_K_i), where K_i is the information store and O_K_i, named *oracle*, is the manager of the information in the store. The aim of an oracle is to answer questions and to make questions to the interpreter, by also using meta-knowledge or heuristics. In particular, an oracle

- retrieves knowledge from the source and stores new knowledge in it;
- generates questions starting from knowledge;
- revises the knowledge.

The *Goal* source contains information about the general goals of the processing agent. Typical goals are *plan coherency*, *contextual place*, *individual prediction*, *group prediction*, *new facts*, *rule copying*, *truths* [34]. Goals play an important role in our model. In fact, the model is goal-oriented, i.e., every problem the agent poses has to be filtered through the goals of the agent itself. This principle has its roots in the fact that many questions are very generic and become operative only when a goal is applied.

O_Goal is a kind of goal monitor [35], which selects the appropriate goal, generates the problem to solve, and asks for it to the interpreter (pro-activity). For instance, once selected the goal ACHIEVE(p), the oracle will generate the problem $?\exists S.(exit\_condition(S) = p)$ (that is "Is there a plan whose exit condition is p?"), and will ask the interpreter for it. The *O_Goal* could also generate new goals by unsolved problems or partially solved ones. This process can be summarized as follows:

$GOAL\ selection \rightarrow PROBLEM\ generation \rightarrow Solution \rightarrow NEW\ GOAL$

The Environment source E models the environment in which the agent operates.

As said above, the set of *guidance* rules R_G drives the interpreter in the achievement of its goals. Therefore, the R_G rules define the general behavior of the interpreter specifying the features of the game played by the interpreter.

The R_G rules are:

1) the interpreter plays a game with *O_Goal*;
   It builds a two columns tableau. Questions are placed on the right column (erotetic part of the tableau), whereas affirmations are placed on the left (assertoric part of the tableau).
2) the game starts with a question;
   This question is called *principal question*.
3) interrogative, assertoric, and communicative moves can be performed;
   Interrogative and assertoric moves are those in L. A communicative move consists in asking a question to an oracle, receiving an answer from an oracle, being asked a question by an oracle, answering back the oracle.
4) the first move is a communicative one: the interpreter asks the *O_Goal* for the principal problem (question) by asking it for "What is the problem associated with P" or, more simply, in absence of environmental commands "What is the problem?".
5) the game ends either when a conclusive answer is found (i.e., it is in the assertoric part of the tableau) [17] or when it is not possible to find it. In the first case the I wins, in the second the *O_Goal* wins.

B. The architecture

Figure 1 shows an example of architecture for an interrogative agent. It is composed of four sources of information: *Goals*, *DS-Knowledge*, *Environment*, and *Problems*, each one having associated an oracle, and an *Interpreter*.

The *Problems* source contains information about problems and the *O_Problems* is able to apply heuristics in order to transform a problem into another one (reduction method). For instance, the heuristic "generalization of individuals" could be informally described as: if a problem P contains one or more individuals a, search for a predicate F such that $F(a)$ holds and try to solve $P' \equiv P(a \leftarrow x)\&F(x)$. In order to apply such an heuristic, *O_Problems* will ask the interpreter for a feature F such that $F(a)$ holds, by a question like "$?K\exists F.F(a)$". In particular, let us consider the problem

$?\exists m.(meeting(m)\&Involve(m, X, Y)\&$
$wife(X, Vance)\&wife(Y, Begin))$

to be transformed by *O_Problems*. By applying the previously described rule, the oracle will ask the interpreter for the existence of a common feature of Vance and Begin. The interpreter, in turn, will ask this question to the *O_DS-Knowledge* which will answer that both Vance and Begin are diplomats. The interpreter will give this information to the *O_Problems* which, finally, will answer the first question

$\exists m.(meeting(m)\&Involve(m, X, Y)\&$
$wife(X, Z)\&wife(Y, T)\&diplomat(Z)\&diplomat(T))$

The *DS-Knowledge* source contains information about a specific domain. For instance, an agent able to solve the previous problem (i.e., the existence of a meeting involving the wives of Vance and Begin) should have knowledge about meetings and diplomatic meetings. In this case, the *O_DS-Knowledge* has the task to search the knowledge base in order to find information.

V. An application: the database refactoring

In this section we describe the application of the proposed model to the problem of database refactoring, which is introduced in the following.

Software refactoring is intended as the restructuring of an existing body of code, aiming to alter its internal structure without changing its external behavior [14]. It consists of a series of small behavior preserving transformations, which altogether can produce a significant software structural change. System modifications resulting in changes to the database structure are also relatively frequent [30]. These changes are particularly critical, since they affect not only the data, but also the application programs relying on them [2].

The refactoring is a very special problem, as it requires that the system coherently changes its own knowledge after a change occurred. If we look at the schema as a knowledge base, the refactoring becomes a process of changes in the

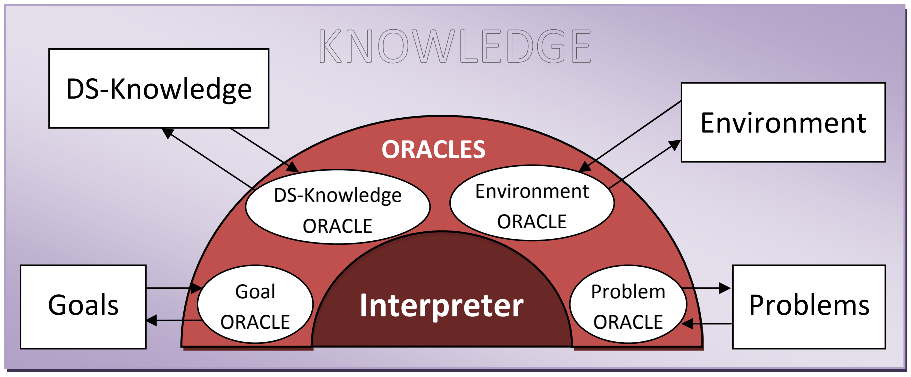

Fig. 1. The architecture of the proposed agent model.

knowledge, and hence it can be interpreted as an epistemic process. According to this view, it becomes natural to see refactoring as an agent managed process aiming to operate on the schema in order to perform the required changes, and trying to preserve original properties in terms of knowledge and queries [7].

The refactoring system can have many different goals, such as the checking of the schema consistency, supporting database administrators in the process of refactoring, or automatically apply suitable consistency maintenance changes. These examples of database refactoring support can be accomplished by three kinds of agents, each one characterized by some specific goals and abilities. For instance, the first agent needs only of deductive capabilities, the second one has a communicative nature, whereas the latter needs to know how to use some heuristics. According to the Schank classification [34], the goal of an agent performing the refactoring is *plan coherency*, i.e., the agent asks itself for the possibility to perform a certain change operation preserving coherency. Thus, the agent is modeled as a problem solver capable to perform changes which are triggered upon the detection of database schema anomalies.

More formally, a *refactoring interrogative agent* is the quadruple

$$R = (I, \{Schema, Problems, G, E\}, L, R_G)$$

Let us consider a database system storing data about employees of a company, and having a query for retrieving all employees of the Computer Science department. The *Schema* knowledge can be represented by

Attributes
$A = \{Employee\_ID, Name, Department\_ID, Salary,$
$\quad Address\}$

Tables
$T = \{R(Employee\_ID, Name, Department\_ID, Salary,$
$\quad Address)\}$

Functional dependencies
$F = \{f_1 : Employee\_ID \rightarrow Name;$
$\quad f_2 : Employee\_ID \rightarrow Department\_ID;$
$\quad f_3 : Employee\_ID \rightarrow Salary;$
$\quad f_4 : Employee\_ID \rightarrow Address\}$

Queries
$Q = \{q(x, y, w, z) \equiv R(x, y, \text{“}CS\text{”}, w, z)\}$

Properties
$P = \{1) primary\_key(R, Employee\_ID)$
$\quad 2) \forall r \in T\ \exists k \subseteq Attr(r)$ such that $primary\_key(r, k)$
$\quad 3)\ key\_dep(r, k) \equiv \forall a \in (attr(r) - k)$
$\quad (\exists f \in F$ such that $(LHS(f) = k \wedge RHS(f) = \{a\}) \wedge$
$\quad (\neg\exists f$ such that $(LHS(f) \neq k \wedge RHS(f) = \{a\}))$
$\quad 4) \forall r \in T\ (primary\_key(r, k) \rightarrow key\_dep(r, k))\}$

The properties in P state that every relation has a primary key, and the attributes fully depend on the primary key only. $Schema$ also contains information about the ϵ operations. For instance, the splitting of a table t into two tables t' and t'', due to the introduction of a new functional dependency f and to the subsequent normalization process, can be defined as follows:

$split\_table(t, t', t'', f) \leftarrow$
$(A' = A\ \&$
$T' = (T - \{t\}) \cup \{t', t''\}\ \&$
$F' = F\ \&$
$Q' = \{q' |\ var(q') = var(q),\ body(q') = \rho(body(q), t, t' \& t'')\}\ \&$
$attr(t') = attr(t) - RHS(f)\ \&$
$attr(t'') = LHS(f) \cup RHS(f))$

An agent for the refactoring has three main problems to solve: *?Consistent(change-operation)*, *?Hold(p)*, and those generated by the predicate *Resolve(change-operation, p)*. The answer to the former is yes when the set of properties P' obtained by the application of the *change-operation* is consistent. The answer to the second one is yes when proposition p holds. Finally, the latter is true when proposition p holds after the application of *change-operation*. All of these problems can be expressed by epistemic logic using the K operator [17]. The K operator is applied to a proposition p using the expression Kp, whose meaning can be informally expressed by “it is known that p". As a consequence, *Hold(p)* is simply expressible as Kp, *Consistent(change-operation)* as $K(\forall p \in \epsilon(P)).p$, whereas *Resolve(change-operation, p)* is nothing else that Kp applied after *change-operation*.

The Environment E is constituted by predicates modeling the ϵ operations.

The agent way of working is as follows. When E gives a command like $split\_table(T, T', T'', f_5 : Department\_ID \rightarrow Address)$ to the *O_Environment* and the latter, in turn, passes it to I, the interpreter has to decide what to do. The interpreter asks the *O_Goal* for the principal problem ("What is the problem associated with $split\_table(T, T', T'', f_5)$?").

Therefore, the *O_Goal* answers

$$Consistent(split\_table(T, T', T'', f_5))$$

and the principal problem is

$$?K(\forall p \in P).p$$

At this point the game starts. The interpreter calls the *O_DS-Knowledge* in order to answer the question "$?K(\forall p \in \epsilon(P)).p$". The answer will be communicated to the *O_Goal*. If the answer is negative, it is necessary to solve the problem trying to apply some heuristic, and the *O_Goal* will select the new goal to be achieved *fixing the incoherence* and the correspondent question:

$$?K\exists\epsilon'(\forall p \in \epsilon'(P)).p$$

A negative answer to this question means that it is not possible to find an ϵ' change operation directly, but that alternative strategies have to be used. To this end, the *O_Goal* can apply the goal *searching for the causes of incoherence* to which corresponds the question "Why is $\neg p$?" or the more concrete "What makes $\neg p$?".

In our concrete case the answer is

$$primary\_key(T'', Department\_ID) \notin P$$

The new goal is *fixing the incoherence*

$$?K\exists\epsilon' primary\_key(T'', Department\_ID) \in P$$

If the answer is negative, the *O_Problems* can suggest to generalize the question in

$$?K\exists\epsilon' p \in P$$

and

$$?K\exists\epsilon' x \in Y$$

this could require many refinement steps.

The answer is $\{add(x, Y)\}$ and therefore $\{add(p, P)\}$.

As this answer is obtained by generalization, it is uncertain (bracketed) and I can ask the environment for the possibility to apply it.

In conclusion, the process we propose is a *trial and error* one, during which the agent tries finding a solution to a given problem by deducting and/or consulting other information sources, possibly restating the problem. While doing so it is driven by its own goals.

VI. Conclusions and Future Works

In this paper we presented interrogative agents, a new model of agents that reason and communicate by using both affirmations and questions. The conceptual foundations of the model are manyfold. They can be found in the Greek dialogical, as well as in recent developments of interrogation logic (philosophy of science and logic), and in artificial intelligence (cognitive science and problem solving).

It is important to highlight both analogies and differences with respect to the classical BDI architecture. In fact, all of the structures of the BDI model are also in our interrogative model, and the functionalities of the BDI interpreter are provided through the interpreter and the oracles. On the other hand, the underlying logic of the proposed model is the logic of interrogation. Thus,our model has both deductive reasoning and heuristics.

The main advantage of the proposed approach is flexibility, which has historically been a characteristic of interrogation logic. However, several issues should further be investigated, such as the development of a logic formalism (based, for example, on the Hintikka formal logic [17]) to model the presented architecture. Moreover, we will focus our future works on the following aspects:

- the study of a logic of interrogation suitable for the interrogative architectures;
- the design of an interrogative agent-oriented language;
- the development of software tools to support the development of interrogative agent-based applications.

Finally, we plan to apply the proposed model in different application fields, such as, database refactoring [7], active video surveillance [9], e-learning of Euclidean plane geometry, and automated FAQ systems.

References

[1] A. Aamodt and E. Plaza, "Case-based reasoning: foundational issues, methodological variations, and system approaches", *Artificial Intelligence Communications*, vol. 7, no. 1, pp. 39–52, 1994.

[2] S. W. Ambler and P. J. Sadalage, *Refactoring Databases: Evolutionary Database Design*. Addison Wesley Professional, 2006.

[3] J. Barklund, S. Costantini, P. Dell'Acqua, and G.A. Lanzarone, "Metareasoning agents for query-answering systems", in T. Andreasen, H. Christiansen, and H. L. Larsen (Eds.), *Flexible Query-Answering Systems*, pp. 103–122, Kluwer Academic Publishers, 1997.

[4] S. Bromberger, *On What We Know We Don't Know: Explanation, Theory, Linguistics, and How Questions Shape Them*. The University of Chicago Press, 1992.

[5] N. D. Belnap, *The Logic of Questions and Answers*. Yale Univ. Press, 1976.

[6] C. Cellucci, *Filosofia e Matematica*. Laterza, 2002.

[7] S. K. Chang, V. Deufemia, G. Polese, and M. Vacca, "A logic framework to support database refactoring", to appear in *Proc. of 18th International Conference on Database and Expert Systems Applications (DEXA'07)*, Regensburg, Germany, 2007.

[8] G. F. De Jong, "Skimming newspaper stories by computer", in *Proc. of the 5th International Joint Conference on Artificial Intelligence (IJCAI '77)*, Cambridge, MA, 1977.

[9] V. Deufemia, M. Giordano, G. Polese, and M. Vacca, "A conceptual approach for active surveillance of indoor environments", to appear in *Proc. of International Conference on Distributed Multimedia Systems (DMS'07)*, San Francisco, USA, 2007.

[10] A. C. Doyle, *Sherlock Holmes: The Complete Novels and Stories*. Bantam Classics, 2006.

[11] U. Eco and T. A. Sebeok (eds.), *The Sign of Three: Peirce, Holmes, Dupin*, Indiana University Press, 1983.
[12] R. Fikes and N. Nilsson, "STRIPS: a new approach to the application of theorem proving to problem solving", *Artificial Intelligence*, vol. 2, pp. 189–208, 1971.
[13] S. Franklin and A. C. Graesser, "Is it an agent, or just a program?: A taxonomy for autonomous agents", in *Proc. of ATAL'96*, 1996, pp. 21–35.
[14] B. Du Bois, P. Van Gorp, A. Amsel, N. Van Eetvelde, H. Stenten, S. Demeyer, and T. Mens, "A discussion of refactoring in research and practice", *Technical report*, no. 2004-03, University of Antwerp, Belgium, 2004.
[15] J. A. G. Groenendijk, "The logic of interrogation", in *Proc. of the Ninth Conference on Semantic and Linguistic Theory*, 1999.
[16] V. F. Hendricks, "Active agents", *Journal of Logic, Language and Information*, vol. 12, no. 4, pp. 469–495, 2003.
[17] J. Hintikka, I. Halonen, A. Mutanen, "Interrogative logic as a general theory of reasoning", in D. M. Gabbay, R. H. Johnson, H. J. Ohlbach, and J. Woods (eds.), *Handbook of the logic of argument and inference. The turn towards the practical*, North-Holland, Stud. Log. Pract. Reason., vol. 1, pp. 295–337, 2002.
[18] J. Hintikka, "L'épistémologie sans connaissance et sans croyance", *Journée de la Philosophie á l'UNESCO*, 2002.
[19] S. Jung, *An Interrogative Approach to Scientific Inquiry*. Peter Lang, 1996.
[20] J. L. Kolodner, "Reconstructive memory: A computer model", *Cognitive Science*, vol. 7, no. 4, pp. 281–328, 1983.
[21] R. A. Kowalski, "The limitation of logic", in *Proc. of ACM Conference on Computer Science*, 1986, pp. 7–13.
[22] W. G. Lehnert, N. G. Dyer, P. N. Johnson, C. J. Yang, and S. Harley, "BORIS - An experiment in in-depth understanding of narratives", *Artificial Intelligence*, vol. 20, no. 1, pp. 15–62, 1982.
[23] W. G. Lehnert, *The Process of Question Answering*. Erlbaum Associates, 1978.
[24] S. de Lin and C. Knoblock, "SERGEANT: A framework for building more flexible web agents by exploiting a search engine", *Journal of Web Intelligence and Agent Systems*, vol. 3, no. 1, pp. 1–15, 2005.
[25] G. Polya, *How to Solve It*. 2nd edition, Princeton University Press, 1957.
[26] G. Polya, *Mathematical Discovery: On Understanding, Learning and Teaching Problem Solving, Combined*. Wiley Press, 1981.
[27] A. Ram, "A theory of questions and question asking", *The Journal of the Learning Sciences*, vol. 1, no. 2/3, pp. 273–318, 1991.
[28] A. S. Rao, M. P. Georgeff, "Modeling rational agents within a BDI-architecture", in *Proc. of KR'91*, 473–484, 1991.
[29] N. Rescher, *Inquiry Dynamics*. Transaction Publishers, 2000.
[30] J. F. Roddick, "A survey of schema versioning issues for database systems", *Information and Software Technology*, vol. 37, no. 7, pp. 383–393, 1995.
[31] R. C. Schank, *Conceptual Information Processing*, North-Holland Publishing Co., 1975.
[32] R. C. Schank, "Language and memory", *Cognitive Science*, vol. 4, no. 3, pp. 243–284, 1980.
[33] R. C. Schank, *Dynamic Memory: A Theory of Reminding and Learning in Computers and People*. Cambridge University Press, New York, 1982.
[34] R. C. Schank, *Explanation Patterns. Understanding Mechanically and Creatively*. Lawrence Erlbaum Associates, 1986.
[35] R. C. Schank and R. Abelson, *Script Plans Goals and Understanding*. Lawrence Erlbaum Associates, 1977.
[36] R. C. Schank and P. Childers, *The Creative Attitude: Learning to Ask and Answer the Right Questions*. Macmillan, New York, 1988.
[37] A. Wisniewski, *The Posing of Questions. Logical Foundations of Erotetic Inferences*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.
[38] A. Wisniewski, "Socratic proofs", *Journal of Philosophical Logic*, vol. 33, n. 3, pp. 299–326, 2004.
[39] M. Wooldridge, "Agent-based computing", *Interoperable Communication Networks*, vol. 1, no. 1, pp. 71–97, 1998.