ANNALES UMCS INFORMATICA DOI: 10.17951/AI.2016.16.2.25

Opening Access To Practice-based Evidence in Clinical Decision Support Systems with Natural Query Language

Paweł Kapłański
Department of Applied Informatics in Management, Faculty of Management and Economics
Gdansk University of Technology,
Gdansk, Poland
e-mail: pawel.kaplanski@zie.pg.gda.pl

Alessandro Seganti
Cognitum
Aleje Jerozolimskie 81, 02-001 Warsaw, Poland
e-mail: a.seganti@cognitum.eu

Krzysztof Cieśliński
Cognitum
Aleje Jerozolimskie 81, 02-001 Warsaw, Poland
e-mail: k.cieslinski@cognitum.eu

Aleksandra Chrabrowa
Cognitum
Aleje Jerozolimskie 81, 02-001 Warsaw, Poland
e-mail: a.chrabrowa@cognitum.eu

Jerzy Koziolkiewicz
Cognitum
Aleje Jerozolimskie 81, 02-001 Warsaw, Poland
e-mail: j.koziolkiewicz@cognitum.eu

Marcin Bryk
Cognitum
Aleje Jerozolimskie 81, 02-001 Warsaw, Poland
e-mail: m.bryk@cognitum.eu

Iwona Ługowska
Maria Sklodowska-Curie Memorial Cancer Center
Institute of Oncology in Warsaw
Warsaw, Poland
e-mail iwona.lugowska@coi.pl

***Abstract*—Evidence-based medicine can be effective only if constantly tested against errors in medical practice. Clinical record database summarization supported by a machine allows allow to detect anomalies and therefore help detect the errors in early phases of care. Summarization system is a part of Clinical Decision Support Systems however it cannot be used directly by the stakeholder as long as s/he is not able to query the clinical record database. Natural Query Languages allow opening access to data for clinical practitioners, that usually do not have knowledge about articial query languages. Results: We have developed general purpose reporting system called Ask Data Anything (ADA) that we applied to a particular CDSS implementation. As a result, we obtained summarization system that opens the access for these of clinical researchers that were excluded from the meaningful summary of clinical records stored in a given clinical database. The most significant part of the component - NQL parser - is a hybrid of Controlled Natural Language (CNL) and pattern matching with a prior error repair phase. Equipped with reasoning capabilities due to the intensive use of semantic technologies, our hybrid approach allows one to use very simple, keyword-based (even erroneous) queries as well as complex CNL ones with the support of a predictive editor. By using ADA sophisticated summarizations of clinical data are produced as a result of NQL query execution. In this paper, we will present the main ideas underlying ADA component in the context of CDSS.**

Keywords— Clinical Decision Support System (CDSS), Natural Query Language (NQL)

I. INTRODUCTION

We define here Clinical Decision Support System (CDSS) after Sim et al.[1] as: *"a software that is designed to be a direct aid to clinical decision-making, in which the characteristics of an individual patient are matched to a computerized clinical knowledge base and patient-specific assessments or recommendations are then presented to the clinician or the patient for a decision"*. If clinical knowledge base of the CDSS *"(...) is derived from and continually reflects the most up-to-date evidence from the research literature and practice-based sources."* We say that it is Evidence-Adaptive CDSS.

Given large number of patient records it is the researcher who discovers correlations and constructs hypotheses. After the theory is statistically tested and published in a literature, systematically developed statements designed to assist medical practitioners and patients with decisions about appropriate health-care for specific clinical circumstances [2] are known as Clinical Practice Guidelines. Due to the fact that they are very formal, the automation of a decision support can be implemented and the computer can make use of patients' clinical data, follow its own algorithm, and present the information relevant to the current clinical situation [3]. In other words, basing on the guidelines automated deductive reasoning tools helps a therapist to provide evidence based diagnosis(4) that is logically followed by a (6) therapy (see Figure 1).

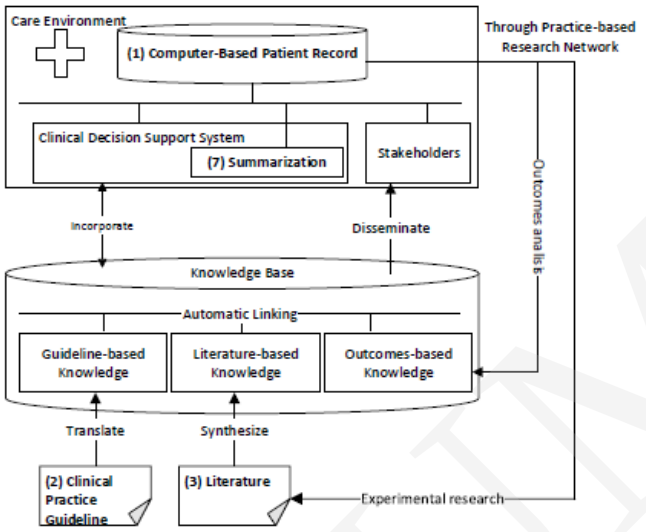

Fig. 1 Knowledge and functionality involved in the use of of CDSSs to support evidence- based medicine (after Sim et al. [1])

Literature and Clinical Practice Guidelines are the main inputs for evidence-based CDSS, however for optimizing health outcomes, local-practice analysis is often required too. Local circumstances like the level of maturity, differences in education, local policy or organizational problems can impact quality of health-care even if made in evidence-based way. To detect these problems it is required to have tools and methods that can provide automatic (7) summarization of knowledge in form of reports that retains the most important points), that after interpretation can help to detect local-practice problems. Moreover, summarization can result in general local practices that will implement local practice-based evidence and ultimately can give source material for a new version clinical practice guideline. On figure 2 involved actors are shown with focus on the Summarization use-case.

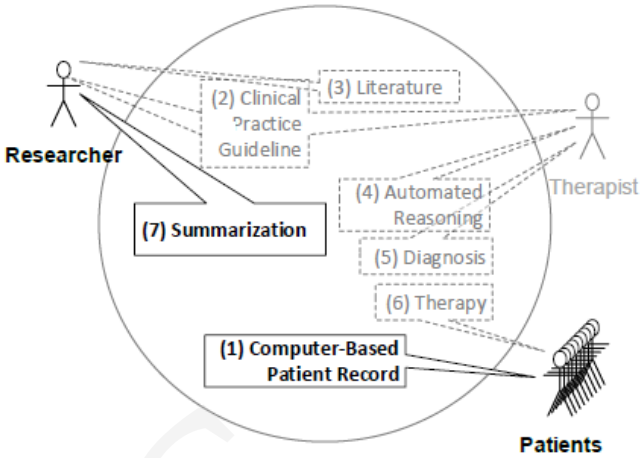

Fig. 2 Clinical data summarization use case

Summarization operates on Computer-Based Patient Record database (PR-DB) and is governed by a set of tools that allows the creation of a potentially unlimited number of machine-generated, data-driven reports, which are calculated by a machine as a response to queries. To create a query to PR-DB it is required to have both: the ability to use language, and knowledge about the structure of the underlying data, and as a consequence, often summarization tools cannot be used directly by the interested stakeholders. In other words: it is desired by the stakeholder to have the ability to examine the data in a query-result loop, where the query is tailored within an interactive process that does not require any large prior learning and preparation. This way of querying data is supported by Natural Query Language (NQL).

The typical architecture of a NQL oriented solution consists of three components: (1) an NQL-based user query interface that is also responsible for the transformation of a natural language query into a formal, machine-readable database query, (2) an underlying database system that in case of CDSS, is PR-DB and (3) a textual or graphical reporting component that presents the results of database computations.

II. Clinical Decision Support System application for GIST cancer treatment

Cancer treatment is one of the area where CDSS applications can help physicians performing evidence based diagnosis and therapy due to strict recommendations and the need for deciding if patients are eligible to enter clinical trial.

The Clinical Decision Support System application for Gist Cancer (GIST-CDSS) is a pilot study devoted to Gastrointestinal Stromal Tumors (GIST). Oncology is a field where recommendations are well defined and studied and where the quality of the clinical data needs to allow for more complex analysis of these data. Strict formalization of the domain knowledge produces consistent data that can be reused for clinical studies.

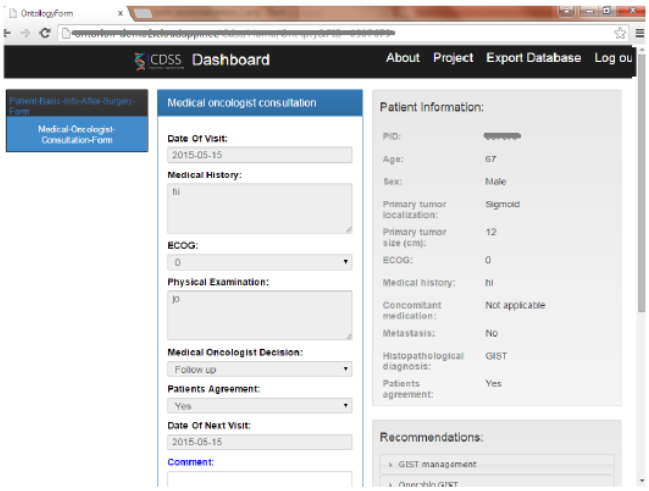

Fig. 3 CDSS

In the GIST-CDSS application (Figure 3) we have modelled the oncological history of a patient to ensure that all the data entering into the application is stored consistently. Furthermore, we were able, together with the domain expert, to model recommendations for the physician that are reasoned depending on the form and on the patient history.

In this paper we present specific component of GIST-CDSS that supports practice-based evidence due to the possibility of automatic summarization via NQL.

The application is currently being tested in the Maria Sklodowska-Curie Memorial Cancer Center and Institute of Oncology in Warsaw.

A. *Architecture of ADA GIST-CDSS component*

Ask Data Anything (ADA) is a NQL system developed by us. It is a general purpose web-browser based application but after tight integrated within GIST-CDSS it becomes to be its core component.

The ADA User Interface (UI) allows a NQL query to be entered and executed with the support of a predictive editor. The result-set of the query execution is presented on a wide range of reports including tables, charts and maps (Figure 4).

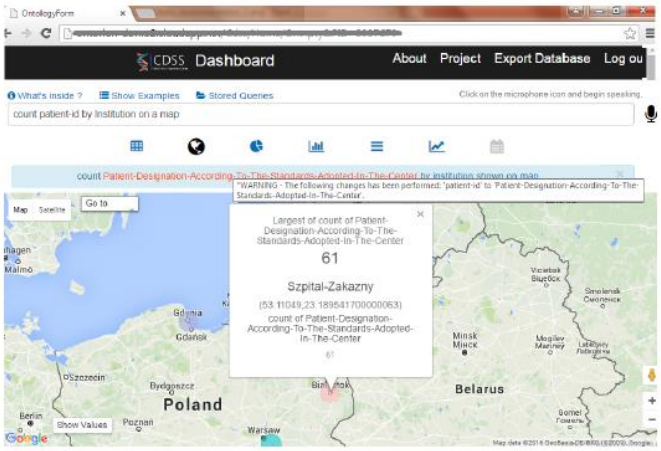

Fig. 4 ADA screen-shot from query execution result-set presented on a map.

III. THE ADA NQL QUERY LANGUAGE

The ADA NQL is a language developed by us to support the users of our ADA system. It is general purpose language, however it can be easily tailored to a specific domain (like here: GIST oncology) with appropriate domain-specific ontology. ADA uses On-Line Analytical Processing (OLAP) cube approach together with a combination of formal logic and statistical analysis to extract dimensions from the data and to expose the dimensions through a natural query language based interface. In this approach the ontology of a given domain and the metadata coming from databases are merged together, therefore it is tightly related to the data and ontology that the user is currently using. The user can write a natural language query while underneath, the query is matched to a more formal CNL, which is finally translated to a query to the underlying PR-DB.

A. *The ADA NQL syntax*

ADA NQL queries should follow grammar presented on Figure 5 and if they do not stick to the grammar, parser first tries to tailor them accordingly. Usually a query starts with an operation (1) specification (sum, average,...) followed by (possibly more than one) dimension(2) specification. The dimension specification(s) is (are) the only required grammatical part(s), all others are optional. The next part of a query defines the subsetting(3) of the data represented by the dimension, by which it is possible to filter the results. The fourth part is the aggregation(4) which allows data to be grouped in subsets. Finally, it is possible to specify the expected visualization(5) type (that can be changed later-on).

Operation. An Operation (optional) is an action we can perform on data to get the desired information: **sum, average, count, maximum and minimum**.

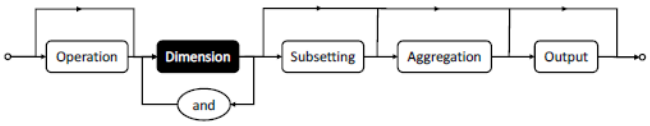

Fig. 5 Syntax for ADA NQL

Dimension. Every action requires at-least one Dimension specification to act on. A Dimension is assigned with a type inferred by parsing a subset of the data together with the information modelled in the supporting ontology. Currently, the type supported by the ADA NQL language are: **numerical, date/time** and **text**, for the types understood directly from the data and: **location/geolocation**, **latitude** and **longitude, hierarchical** (text dimension defined in the supporting ontology that can have super concepts grouping the values (e.g. infectious-disease for a column with diseases,...) and **row** (dimensions that are defined in the supporting ontology and represent data from multiple columns in a single row).

Operations and types are matched in the parser to check that the query makes sense (e.g. “Sum Patient” where patient is a dimension with Text values inside is not allowed but “Sum Some-Row” where Some-Row is a row that contains a numerical dimension is allowed).

The query language allows also the use of:

- **symbols** - defined in the supporting ontology that are evaluated to concepts, mainly used for subsetting and grouping, described later
- **literal values** (e.g. Contract 123, Rome,...) used for subsetting

these components (as for the dimensions) are dependent on the dataset currently loaded.

Subsetting. The subsetting part of the query can be used to define the filters that the user wants to apply to his/her query. The general syntax is (Dimension, Relation, Data) where as described before, the Relation and the Dimension are matched by the dimension type (thus Dimension > 4 is allowed only for Numerical dimensions). In this part of the query, it is also possible to use **"in"** constraints. After the in constraint, we expect an entity declared in the ontology (e.g. location "city", class of abstractions like "infectious disease") or the content of some column.

Example. Lets consider the following query: *Average age of patient that has-tumour-size greater-than 10 and is not a diabetic and-or is a paracetamol-tolerant on a piechart*. This query contains the following complex expression: *"patient that has-tumour-size greater-than 10 and is not a diabetic and-or is a paracetamol-tolerant"* that evaluates into DL concept expression: patient ⊓((∋have-tumour-size>10 ⊓ ¬diabetic) ⊔paracetamol-tolerant)

During reasoning process, that takes place in Ontology Management System, we obtain set of instances of the aforementioned complex concept expression: (Patient-1,Patient-2,...) that are then injected into final SQL query:

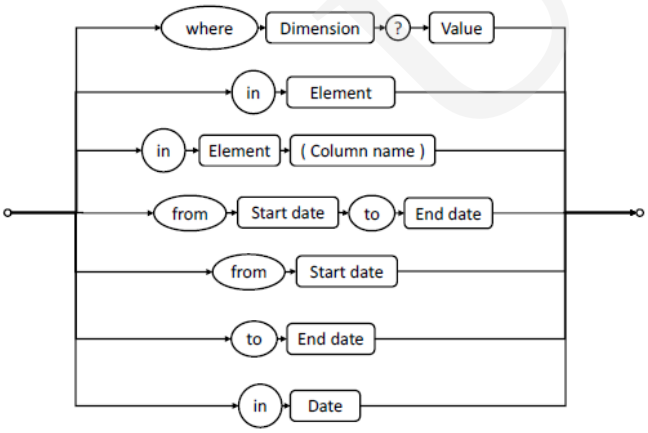

Fig. 6 Subsettings Syntax

```
Select avg(age) from PR_DB
where fish in (Patient-1,Patient2,…)
```

Subsetting by date is very expressive, for example the user can write: "from year 2015 to/until year 2016", "from July 2015 to/until September 2015", "from 1st of July 2015 to/until 23rd of October 2015", "from 07/01/2015 to/until 08/02/2015" or "from 07/01/2015 12:23 to/until 08/02/2015 09:22".

Aggregation. Aggregation is the action of grouping the result using one of the Dimensions and/or entities which were defined in the ontology; the syntax for aggregation is described on Figure 7 with **"by"** together with dimension, location (i.e. city, country), and time period (year, month, day, date). Multiple aggregations are allowed (e.g. by country and by day). Some aggregations require operations and others do not (e.g. by day can be used with or without operations on the dimension, while by country needs an operation).

Outputs. It is possible to specify in the query language the output on which the query result should be shown. ADA currently support following types of outputs: **table, histogram, stacked-bar, map, piechart, line or timeline**. After the query is parsed, the parser decides which of the outputs are allowed depending on the type of dimensions that will be returned.

IV. Evaluation

A. Advantages and disadvantages of ADA NQL

To evaluate our system, firstly we tried to place it in the spectrum of well-known advantages and disadvantages of NQL (see [4])

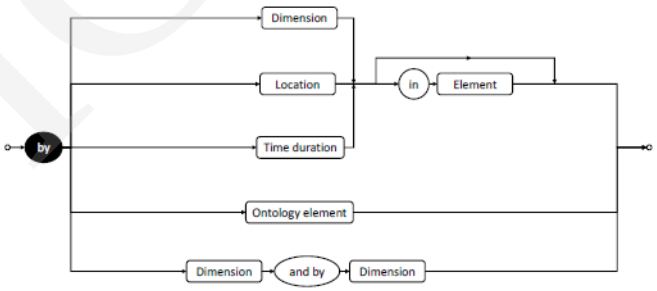

Fig. 7 Aggregation Syntax

1) Advantages

- Using ADA NQL does not require prior learning of a database query language like SQL or SPARQL. Following SQL query that uses embedded SPARQL and is aimed to select average value of age of diabetic patient from a given PR-DB in SQL+SPARQL has a form of:

```
select avg(age)from PR_DB where
product in (
select distinct ?x {
?x rdf:type ns:patient.
?x rdf:type ns:diabetic
})
```

  It is required to learn how to construct valid SQL/SPARQL queries to execute them on top of the given dataset. The same query in ADA NQL has a form: "*Average age by a patient in diabetics*". This form do not require extensive prior learning - the user writes it in English.

- ADA NQL is simple but expressive. E.g.: form-based GUI presented on a left side of a Figure 8 contain multiple fields that need to be correctly filled making the overall process of querying the database a

complex task. To enter the same into single ADA NQL query is much easier with predictive editor: entering "*Average age by patient in diabetics …"* and then select one of the appearing options to continue the query.

- Arbitrary complex queries can be not possible to be expressed in form-based UI. E.g.: a left side of a Figure 8 presents an UI setup for the application that executes a query equivalent to the following ADA NQL query: "*Average age by patient in diabetics …"* however with ADA NQL by extending this query with additional *"... and paracetamol-tolerant"* we easily can make the query arbitrary complex. Only generic, Tree-View based, dynamic UI (presented on the right side of Figure 8), that for complex queries can potentially occupy more space that is available on any computer screen allows one to specify arbitrary complex queries.

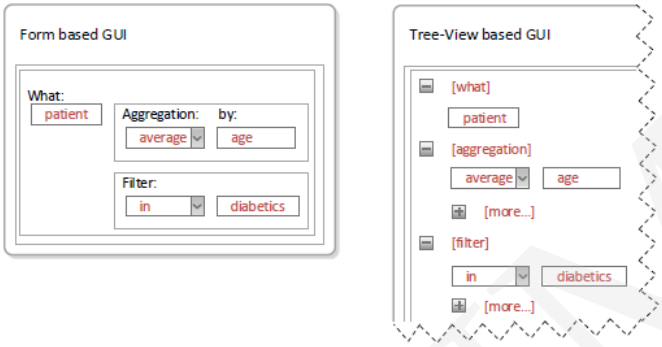

Fig. 8 Example of form based UI for database queries

- ADA is fault-tolerant, providing the user with a result-set together with an information about what is "understood" as a query (right side of Figure 9). For example the following erroneous ADA query: "*avrage agee inpatnts by municipality on map*" is rewritten into: "*average age in patients by municipality on a map*", prior being evaluated by the ADA engine.

2) Disadvantages

Now we explain how ADA NQL approach reduces common NQL disadvantages.

- ADA provides many hints for an inexperienced user, that allow her/him to understand a linguistic coverage of the ADA NQL. One of them is a predictive editor (left side of Figure 9). Predictive editor provides the user with kind of rails during the query construction process. It actively suggests continuation of the query based on current position and context (loaded dataset and ontology).
- ADA provides the user with an "understood" query together with a result-set (right side of Figure 9). The understood query is colorized and extended with explanations to explain why some part has been added or modified making easier to understand the obtained result-set - and ultimately the problems within an entered query. By comparing the result-set with differences between "entered" and "understood" query user is able to learning how to specify the "correct" queries.
- ADA learning loop, described previously, communicates to the user the abilities of the ADA so even if the user ask questions that include judgements and beliefs the "understood" query will explain him/her that parts of the query were dropped.

B. NQL Parser Evaluation

We can distinguish three general approaches to the NQL:

1. keyword based, which allows for free writing similar to the full-text search approach, but supports only very basic queries,
2. patten based, which detects common query patterns and generates responses based on a set of rules, and therefore allows for more complex queries but limited in the number of rules,
3. grammar based, which requires strict grammar and syntax followed by a structural/predictive editor as it is hard for the inexperienced user to enter a grammatically valid query, but once entered it can be very complex, deep and meaningful.

In each case the NQL query is rewritten into the underlying database query language but each of the approaches has its limits and advantages. Our ADA NQL combines all the approaches together. The main characteristics of the NQL parser we have built are: Our ADA NQL combines all the approaches together. The main characteristics of the NQL parser we have built are:

- it is robust (queries like : sum Patients or Patients summed or sum of the best Patient I know of are parsed in the same way),
- it tries to understand what the user meant by his/her query (e.g. the query Age by country is automatically translated to Sum Age by Country as we cannot make aggregation without operations),
- it is flexible (the dictionary used to match the words in the query is taken from the input data and from an ontology),
- it is dedicated for making analytical queries to sets of data.
- part of its content is defined in the data and another part is defined in an ontology associated to the data.

So on the one hand we have defined a controlled natural language with a strict syntax (see Section 3.1), while on the other hand the parser tries to match the written query to the controlled natural language query in all possible ways.

V. RELATED WORK

On-Line Analytical Processing (OLAP) cube approach, is already used with CDSS. It provides decision-makers with online access to analytical capabilities based on the idea of dimensions. deals with dimensions and measurements and

therefore it is suitable solution for summarization. Decision support systems that use advanced technologies such as (OLAP) and data mining to deliver advanced capabilities is presented in [5]. In [6] integrated data warehousing, OLAP, and data mining techniques are presented to support process of care-givers and clinical managers. The possibilities of using data warehousing and OLAP technologies in public health care in general is presented in [7].

LUNAR [8] was the first NQL-database system that allowed natural language to be used to query a database about samples of moon rocks, however, nowadays this system is considered to be very limited in linguistic capabilities [9]. RENDEZVOUS [10] was the system that implemented the "man-in-the-loop" way of human-database interaction based on a dialogue with a machine. Within the dialogue system it was possible to clarify all the difficulties found during the initial user input by helping the user to formulate queries. LADDER [11] was a general purpose NQL-database that was able to be connected to different underlying DBs, but at the same time it used grammars that were application-dependent making the system hardly portable. CHAT-80 [12] transformed English into Prolog expressions that were then evaluated against an existing database. CHAT-80 was a foundation for other experimental systems e.g.: MASQUE [13] and PRECISE [14]. ACE - Attempto Controlled English [15] is a Prolog-based, widely adopted general purpose language that allows a CNL-based NQL to be built. CNLs like ACE, being very precise and expressive require, at the same time, the use of a predictive editor that forms a kind of rails on which the user can write a syntactically correct sentence.

Also, modern NQLs are configurable with certain domain-specific ontologies, making the NQL core domain-agnostic. In the {AskMe* } system [16] an ontology is generated when the system is connected to a database. The generator processes the schema of a given database and generates an ontology that contains knowledge about the domain, properties, relationships and constraints that already exists in the given database. The ontology is then used to automatically generate a specific parser. Another example of a modern approach is SWSNL [17]. It is a semantic search engine equipped with a natural language interface. The user input in natural language is analyzed by the linguistic component and produces its formal representation. The linguistic component combines a few Natural Language Preprocessing (NLP) technologies like: Named Entity Recognition (NER) and semantic analysis. As a result, a SPARQL [18] query is generated and executed.

VI. Conclusion and Future work

We implemented NQL oriented summarization solution as a part of GIST-CDSS. We used general purpose ADA solution and tailored it to the needs of oncology-specific CDSS with specific ontology. ADA ontologies contain both: the knowledge about the configuration as well as the general knowledge that can be easily reused. The main query language is ADA NQL - the query language that accepts a large spectrum of (even erroneous) natural queries. Obtained summarization system is currently being tested in the Maria Sklodowska-Curie Memorial Cancer Center and Institute of Oncology in Warsaw.

VII. Conflict of interest

All authors work for Cognitum, the company behind the Ontorion Server, Fluent Editor and OCNL.

VIII. Acknowledgments

This research was financially supported by The Foundation for Polish Science - Parent Bridge Grant.

References

[1] I. Sim, P. Gorman, R. A. Greenes, R. B. Haynes, B. Kaplan, H. Lehmann, P. C. Tang, Clinical Decision Support Systems for the Practice of Evidence-based Medicine, Journal of the American Medical Informatics Association : JAMIA 8 (6) (2001) 527-534. URL http://www.jamia.org/cgi/content/abstract/8/6/527?ijkey=64927082e12701fb37247928059d7a6b9eb97ac9

[2] M. J. Field, K. N. Lohr, Clinical Practice Guidelines: Directions for a New Program, The National Academies Press, Washington, DC, 1990. doi:10.17226/1626. URL http://www.nap.edu/catalog/1626/clinical-practice-guidelines-directions-for-a-new-program

[3] D. C. Stahl, L. Rouse, D. Ko, J. C. Niland, Gdsi: a web-based decision support system to facilitate the eficient and effective use of clinical practice guidelines, in: System Sciences, 2004. Proceedings of the 37th Annual Hawaii International Conference on, 2004, pp. 10 pp.{. doi:10.1109/HICSS.2004.1265377.

[4] I. Androutsopoulos, G. D. Ritchie, P. Thanisch, Natural language interfaces to databases - an introduction, CoRR cmp-lg/9503016. URL http://arxiv.org/abs/cmp-lg/9503016

[5] S. Palaniappan, C. S. Ling, Clinical decision support using olap with data mining, International Journal of Computer Science and Network Security 8 (9) (2008) 290{296.

[6] N. Stolba, A. M. Tjoa, The relevance of data warehousing and data mining in the field of evidence-based medicine to support healthcare decision making (2005).

[7] H. N. Khraibet, A. H. Mousa, A. Bakar, M. Shahbani, Intelligent Iraqi Health System (IIHS) using Online Analytical Process (OLAP) model.

[8] W. Woods, R. Kaplan, B. Nash-Webber, The Lunar Sciences Natural Language Information System: Final Report, BBN report, Bolt Beranek and Newman, 1972. URL https://books.google.pl/books?id=RhuEMwEACAAJ

[9] M. N. Nihalani, S. Silakari, M. Motwani, Natural language interface for database: a brief review, IJCSI International Journal of Computer Science Issues 8 (2) (2011) 600{608.

[10] E. F. Codd, Seven steps to rendezvous with the casual user., in: IFIPWorking Conference Data Base Management, 1974, pp. 179{200, iBM Research Report RJ 1333, San Jose, California. URL http://dblp.uni-trier.de/db/conf/ds/dbm74.html#Codd74

[11] G. G. Hendrix, E. D. Sacerdoti, D. Sagalowicz, J. Slocum, Developing a natural language interface to complex data, ACM Trans. Database Syst. 3 (2) (1978) 105{147.

doi:10.1145/320251.320253. URL http://doi.acm.org/10.1145/320251.320253

[12] D. H. D. Warren, F. C. N. Pereira, An efficient easily adaptable system for interpreting natural language queries, Comput. Linguist. 8 (3-4) (1982) 110{122. URL http://dl.acm.org/citation.cfm?id=972942.972944

[13] I. Androutsopoulos, G. Ritchie, P. Thanisch, Masque/sql - an efficient and portable natural language query interface for relational databases, Database technical paper, Department of AI, University of Edinburgh.

[14] A.-M. Popescu, A. Armanasu, O. Etzioni, D. Ko, A. Yates, Modern natural language interfaces to databases: Composing statistical parsing with semantic tractability, in: Proceedings of the 20th International Conference on Computational Linguistics, COLING '04, Association for Computational Linguistics, Stroudsburg, PA, USA, 2004. doi:10.3115/1220355.1220376.URL http://dx.doi.org/10.3115/1220355.1220376

[15] N. E. Fuchs, U. Schwertel, R. Schwitter, Attempto controlled English – not just another logic specification language, in: LOPSTR '98: Proceedings of the 8th International Workshop on Logic Programming Synthesis and Transformation, Springer-Verlag, London, UK, 1990, pp. 1{20.

[16] M. Llopis, A. Ferrandez, How to make a natural language interface to query databases accessible to everyone: An example., Computer Standards & Interfaces 35 (5) (2013) 470{481. URL http://dblp.uni-trier.de/db/journals/csi/csi35.html#LlopisF13

[17] I. Habernal, M. Konopik, Swsnl: Semantic web search using natural language., Expert Syst. Appl. 40 (9) (2013) 3649{3664. URL http://dblp.uni-trier.de/db/journals/eswa/eswa40.html#HabernalK13

[18] S. Harris, A. Seaborne, “Sparql 1.1 query language”, http://www.w3.org/TR/sparql11-query/, accessed 21st September 2015 (2013).