Пловдивски университет „Паисий Хилендарски“
Факултет по математика и информатика
Катедра „Компютърна информатика“

МАРИЯ МИНЕВА ЖЕКОВА

Естественоезиков интерфейс за информационна система от тип ‘електронен университет’

АВТОРЕФЕРАТ

на дисертационен труд
за присъждане на образователната и научна степен „Доктор“

Област на висше образование: 4. Природни науки, математика и информатика
Професионално направление: 4.6. Информатика и компютърни науки
Докторска програма: „Информатика“

Научен ръководител:
проф. д.м.н. Георги Атанасов Тотков

Пловдив, 2022

Дисертационният труд е обсъден и насочен за защита пред научно жури, на заседание на катедра „Компютърна информатика“ при Факултета по математика и информатика на Пловдивския университет „Паисий Хилендарски“, на 16.03.2022 г.

Дисертационният труд „Естественоезиков интерфейс за информационна система от тип електронен университет“ съдържа 144 страници. Списъкът на използваната литература включва 82 източника, от които 8 (осем) на кирилица и 74 (седемдесет и четири) на латиница. Списъкът на авторските публикации по темата се състои от 6 заглавия.

Материалите по защитата са на разположение на интересуващите се в Деканата на Факултета по математика и информатика, Нова сграда на ПУ „Паисий Хилендарски“ (бул. „България“ №236), всеки работен ден от 8:30 до 17 часа.

Автор: **Мария Минева Жекова**

Заглавие: **Естественоезиков интерфейс за информационна система от тип ‚електронен университет‘**

NATURAL LANGUAGE INTERFACE FOR AN E-UNIVERSITY TYPE INFORMATION SYSTEM

Mariya Mineva Zhekova

The main goal of the doctoral dissertation is to propose, study and test appropriate tools for creating a natural language interface (NLI) to an information system (IS) for answering user questions in natural language.

The research examines the general theory, existing systems with NLI, models and approaches for natural language processing and information retrieval from databases of IS.

Based on the theoretical research, a number of conceptual and computational models are proposed, as follows:

- general model of a process for creating a natural language interface to an information system;
- domain area model;
- a model of a software tool for forming a database query from user question in natural language, independent of the language of the queries.

As a result, the architecture of the NLI to the information system is defined and a corresponding software tool is built on the existing e-University information system. The general models are specified and applied in the creation of two experimental NLIs, with the help of the software tool, to different IS in the field of higher education, such as:

- e-University type information system;
- electronic platform of NEAA (The National Evaluation and Accreditation Agency).

Applications experiment with user sentences to the databases of relevant systems and prove the adequacy of the models and the effectiveness of the software tool. The results obtained in the thesis could be reproduced to create a NLI to information systems in other domain areas.

Списък на използваните съкращения

БД	–	База данни
ВО	–	Висше образование
ЕЕ	–	Естествен език
ЕЕИ	–	Естественоезиков интерфейс
ЕЕО	–	Естественоезикова обработка
ИИ	–	Изкуствен интелект
ИС	–	Информационна система
НАОА	–	Национална агенция за оценяване и акредитация
ПО	–	Предметна област
СУБД	–	Система за управление на база данни
POS	–	Part of speech – част на речта
SQL	–	Structured Query Language – език за структурни заявки към бази данни

Съдържание

Увод

Стремежът към достъпност на човешкия език от компютрите и ускореното развитие на технологиите, поставят на преден план развитието както на езиковият анализ и обработка, така и извличане и синтез на текст. Това поражда необходимост от създаване на автоматизирани интелигентни приложения с опростен интерфейс, които позволяват естественоезикова обработка (ЕЕО) и превръщане на потребителски въпрос на естествен език в заявка към база данни (БД) на информационна система.

Подходът, който улеснява търсенето за неспециалисти е интерфейс на естествен език, който превежда човешкото намерение в инструкции към БД. Естественоезико-вият интерфейс (ЕЕИ) е подход за извличане на информация от БД, който позволява на потребителите да въвеждат запитванията си към информационна система (ИС) на естествен език (ЕЕ). Предимството на ЕЕИ е тяхната достъпност сред широк кръг от потребители и по-лесната комуникация с база от данни, изразяваща се в задаване на въпроси и извличане на информация, без познаване и използване на изкуствен език (напр. езикът за структурни заявки SQL).

Актуалността на тематиката се обосновава от необходимостта от изграждане на ЕЕИ за ИС с универсален характер. Системите с ЕЕИ, създадени до момента, решават епизодично проблема, ползвайки собствена БД и собствена програмна логика, за конкретна предметна област (ПО).

В дисертацията е представено изследване в областта на компютърната лингвистика, което включва методика за описание на ПО и софтуерен модул с ЕЕИ към ИС за извличане и обработка на потребителски текст-въпроси, зададени на ЕЕ. В предложеният модел на процес за създаване на ЕЕИ са използвани различни средства и подходи, доразвити и модифицирани, за да извличат отговор от БД на различни ИС в една и съща ПО. В допълнение на това е разработен олекотен, интелигентен и user-friendly интерфейс на естествен език към базата данни на системата.

Цели и задачи

Основна цел на дисертационното изследване е да се предложат, изследват и апробират средства (модели, методи и инструменти), подходящи за създаване на естественоезиков интерфейс към информационна система за отговаряне на потребителски въпроси на естествен език. Изпълнението на поставената цел предполага решаване на **следните задачи**:

Задача 1. Проучване на съществуващи теоретични подходи, модели и системи, свързани с ЕЕИ.

Задача 2. Създаване на общ модел, архитектура и прототип на ЕЕИ към информационна система, както и на методика за тяхното прилагане.

Задача 3. Създаване на софтуерен инструмент за формиране на заявка към БД от потребителски текст на естествен език.

Задача 4. Проектиране, реализация и тестване на софтуерен инструмент с ЕЕИ за различни информационни системи (в област „Висше образование“), базиран на предложения модел и следвайки конкретни методики.

Структура на дисертацията

Дисертационният труд се състои от Списък на използваните съкращения, Списък на таблиците, Списък на фигурите, Увод, 4 (четири) глави, Заключение, Приложения, Списък на публикациите, Списък на използвана литература и Декларация за оригиналност и достоверност.

Основният текст на дисертационния труд се състои от 144 (сто четиридесет и четири) страници и е съпроводен от 2 (две) приложения (13 страници).

В **Глава 1. Състояние на изследванията** са разгледани и представени основни понятия, методи и подходи за описание на знания и извличане на важно съдържание от база знания, съществуващи инструменти и системи за ЕЕО, както и средства за автоматизиране на тези обработки. Предложен е подход за създаване на ЕЕИ към ИС.

В **Глава 2. Моделиране и проектиране на ЕЕИ** е представен общ модел на процес за създаване на ЕЕИ към ИС. Предложена е методика за моделиране на ПО на ИС, проектирани са нейните специфични компоненти – списък от понятия, характерни за ИС и областта, тяхното лексикално, лингвистично описание и са моделирани струк-турни шаблони на въпроси, на които ИС може да отговори. На базата на въведените модели е предложена архитектура на софтуерен инструмент с ЕЕИ.

В **Глава 3. Софтуерна реализация на ЕЕИ към ИС** е представен програмен код и алгоритъм за създаване на софтуерен модул за ЕЕИ, който може да се имплемен-тира към различни ИС за отговаряне на потребителски въпроси. Софтуерният инстру-мент е съставен от 2 (два) модула. Описана е последователността от действия, извърш-вани от анализатора и генератора на заявки към БД на ИС.

В **Глава 4. Експерименти по създаване на ЕЕИ към ИС** методиката за създава-не на ЕЕИ от Глава 2. е приложена за два експериментални ЕЕИ към прототипи на БД на електронен университет и БД на електронна платформа на НАОА. Показани са тес-тове, резултати и оценки, свързани с работата на ЕЕИ в двата експеримента. С помощ-та на серия от тестове са изведени необходимите действия за подобряване на моделите от Глава 2. и софтуерното решение в Глава 3.

В **Заключението** са обобщени получените резултати по Задачи 1. – 4. Посочени са основните научни, приложни и научно-приложни приноси на дисертационния труд. Формулирани са перспективи за бъдещо развитие.

Благодарности

Преди всичко искам да благодаря на моя научен ръководител проф. д.м.н. Георги Тотков за предоставената ми възможност, за вярата в мен, за търпението и подкрепата по време на целия период на обучение и работа върху дисертацията.

Искрено благодаря на колегите гл.ас.д-р Георги Пашев и доц.д-р Силвия Гафтанджиева за техните внимателни прегледи, мнения и препоръки на моята теза. Благодаря и на проф.д-р Елена Сомова за това, че бях участник в Зимна докторантска школа в Технически университет, гр.Кошице – Словакия и Лятна докторантска школа в университет ELTE, гр.Будапеща – Унгария.

Изследването е подкрепено и от проект BG05M2OP001-1.002-0002-C02 "Дигитализация на икономиката в среда за големи данни", финасиран от европейския фонд за регионално развитие, в рамките на ОП "Наука и образование за интелигентен растеж".

Благодарна съм и на моето семейство, което ме подкрепя и насърчава да следвам целите и интересите в академичната ми кариера.

Глава 1. Състояние на изследванията

Естественият език предоставя сериозни предизвикателства пред разработчиците и изследователите в областта на ЕЕО. В тази глава са представени основни предизви-кателства, методи и подходи, мотивирали разработчиците към нови инструменти за търсене и представя някои решения, които биха насочили настоящите и бъдещите специалисти в областта на обработката на ЕЕ.

1.1. Основни понятия и определения

В Глава 1. са представени определения на термини, свързани с темата на дисерта-ционното изследване.

Автори	Определение
Тотков'2001	***Понятие*** е мисъл за обобщение на обекти и процеси от действителността
Bjorner'2019	***Предметна област*** (ПО) е рационално описан от човека сегмент от света с неговите физически части и живи същества.
Осенова'2016	***Знания за ПО***, в зависимост от избрания описателен подход, могат да се класифицират като: (а) *декларативни* - термини и понятия от ПО, техни специфични свойства, типове, подкласове, както и отношения между тях; (б) *процедурни* – набор от правила, и процедури, действащи в ПО
Тотков'2001	***Моделиране*** е метод за получаване на информация за конкретен обект чрез: формулиране на цел; анализ на обекта; построяване на модел-аналог; изследване на свойства и отношения; сравняване на моделираните свойства; пренасяне на откритите свойства и отношения върху обекта.
Bjorner'2019	***Моделирането на ПО*** е дейността по описание на понятия и формализация на ПО (дефиниции на клас, тип, функции, правила, аксиоми и условия). Моделът на ПО определя атрибути за всеки от обектите и идентифицира връзките помежду им. Всяка връзка предоставя средство за откриване на информация.
Androutsopoulos'1995	***Естественоезиков интерфейс към БД*** е система, която позволява на потребителя да има достъп до информация, съхранявана в база данни, като въвежда заявки, изразени на някакъв естествен език.
Liddy'2001	***Естественоезикова обработка*** е технология, базирана на компютърни модели на езикови явления, която създава и използва езикови ресурси и методи за анализ и оценка. Включва процеси – сегментация, нормализация, семантично представяне, диференциране на имена, дати, роли, събития, участници и др.

Таблица 1. Определения на термини, свързани с дисертационното проучване

1.2. Информационни системи с ЕЕИ

В настоящия раздел е представено проучване, в което са разгледани различни подходи, за ЕЕО и анализ на свободен текст на естествен език, наложили се във време-то. Системи с ЕЕИ възникват през 70-те години на XX век и до сега продължават да се развиват и усъвършенстват, за да отговорят на търсенията на потребителите и индустрията. Системите, фокусирани в анализ и обработка на естествен език могат да се **категоризират въз основа на подхода**, който използват за описание на знания и за извличане на информация [Affolter'2019].

Подходът за търсене по **ключови думи** се характеризира с набор от индексирани понятия. Този модел за ЕЕО включва: предварителна обработка и параметризация на понятията от входния низ [Vallez'2007]. ИС, основани на **ключови думи** позволяват генериране на заявки към БД на база на ключови думи. След като постъпи потребител-ска заявка, идентифицираните ключови думи се търсят в БД, проверяват се таблиците, които ги съдържат и се установяват релациите помежду им. Заявките в **Precis** [Simitsis'2008] са в свободна форма, чийто отговор е синтез на резултати, съдържащ не само пряко свързана със запитването информация, но неявно свързана информация по различни начини. Системата решава дали да присъедини и други предикати, които са от значение и изгражда ограничения, на която трябва да отговарят резултатите от заявката. **SODA** [Blunschi'2012] използва граф с метаданни за синтез на SQL заявка, йерархична структура и модел на данни с различни нива на описание и детайлност, включително синоними на търсените понятия. Всяко запитване се анализира и категоризира по три критерия: обект, операция, стойност.

Подходът, основан на **шаблони** проектира структурни модели за автоматично съпоставяне на семантиката на ЕЕ в езиците за програмиране и предоставя насоки за тяхното разпознаване [Zhekova'2019A]. ИС разчитат на **съвпадение на шаблони**, за да направят директно съответствие на потребителска заявка със стойности, атрибути и таблици в БД. Принципа, че ако входния стринг съвпада с някой от шаблоните, то

системата е в състояние да генерира заявка към БД. Системата **LADDER** използва семантична граматика, анализира въпроси и прави запитвания към ограничена БД. Тя обработва прости заявки от една таблица или запитвания към няколко таблици с лесни условия за съответствие [Hendrix'1978]. Системата **CHAT-80** [Warren'1982] позволява на потребителите да задават ЕЕ въпроси към Prolog база данни. Системата обработва потребителски въпроси на три етапа – транслация, планиране, изпълнение. Функцията на анализиращия модул определя граматичната структура на изречението, а интерпретацията и обхватът се състоят от различни правила за съответствие, изразени като клаузи на Prolog.

Продукционните правила опростяват, обработват и организират езиковите единици на входа, до единна и предвидима ситуация. Всяко правило съдържа условия, изводи или действия и е вид предикатна логика с добавени компоненти, показващи как информацията в правилото се използва в разсъжденията [Zhekova'2019A]. Опреде-лят се набор от конструкции, за моделиране на действия, които трябва да се предприе-мат на всеки етап [Batra'2004]. ИС, базирани на **правила** позволяват на потребителите да съставят заявки чрез шаблони на изречения и извличат отговори от БД чрез дедуктивни разсъждения [Baxter'2005]. Подходът може да се настрои да връща само сигурни заключения, като ограничава неуспехите [Reynolds'2016].

MASQUE е система с ЕЕИ, която отговаря на въпроси чрез генериране на Prolog заявки [Androutsopoulos'1995]. Модифицираната версия Masque/SQL отговаря на английски въпроси чрез генериране и изпълнение на SQL заявки. Потребителските въпроси се трансформират в подобни на Prolog изрази, написани на логически език за представяне (LQL). Разработен е алгоритъм за превеждане на LQL изрази в SQL заявки [Gonzalez'2015]. **SILT** предлага нов подход за намиране на съответствия между естественоезикови фрагменти и техни формални представяния на езика за структурни заявки, използвайки правила за преобразуване [Kate'2005]. Моделът на правило е съпоставен с фрази в изречението, и когато успешно съвпада, съответният шаблон се инстанциира, за да създаде част от заявка. **CYC** е база от знания и машина за логически изводи, базирани на микроколекции от правила и факти, които се отнасят до една конкретна ПО [Lenat'1985]. Основна цел е построяване на БД за всички общи понятия, семантични връзки между тях и множество от правила за заключения. CYC различава физически лица, константи, колекции, предикати и функции за истинност.

Подходът, базиран на **фрейми** представя знания за обекти като набор от слотове и стойности, като всеки слот описва атрибут или действие на фрейма. Множеството от стандартни ситуации, сценарии, диалози и структури на всяка конкретна изучавана ПО се моделира със система от фрейми. Тази възможност води до по-лесно конструи-ране на система от правила и повишава контрола – кога и с каква цел да се използват отделни колекции от правила [Zhekova'2019A]. Системата **PALKA** се базира на раз-познаването на семантични фразови модели при анализ на входни изречения [Kim'1993]. Тя използва предварително дефинирани структури, кодиращи синтак-тични, семантични и контекстуални знания във вид на фрейми. Базата знания е органи-зирана като мрежа от семантични фреймови структури и концептуална йерархия.

Базираните на **граматики** системи позволяват задълбочен анализ на ЕЕ заявки и семантични уеб технологии за извличане на информация от БД. Голяма част от ИС ползват **езикови корпуси** за извличане на лексикални типове, аргументи и параметри под формата на метаданни. Семантичните анотации позволяват данните да се предста-вят и етикетират с онтологии, като по този начин се улеснява интегрирането, опера-тивната им съвместимост, индексирането и търсенето в БД [Handschuh'2003].

В **OVIS** [OVIS] са разработени два алтернативни модула за ЕЕО: (а) базиран на граматика и (б) ориентиран към данни. Граматиката в OVIS описва потребителски изречения, предлози, времеви маркери, именувани обекти и словосъчетания. В систе-мата

JOOST текстови колекции се анализират както чрез маркиране с части на речта и именувани обекти, така и чрез синтактичен анализ, използвайки релационни отношения. Анализът на ЕЕ въпрос цели да определи типа на въпроса и да идентифи-цира ключови думи в него. Отговорите се формират чрез синтактично припокриване между въпроса и отговора, припокриване на думи, припокриване на свойства, надеждност на модела, използван за извличане на отговора, и честотата на отговора [Bouma'2007]. **COMUNICA** [Wilkens'2010] е система с възможности за търсене както в структурирани, така и в неструктурирани набори от данни. Тя се състои от пет модула: модул, който е отговорен за интеграцията и комуникацията с четирите модула за разпознаване на реч, обработка на текст, достъп до база данни и синтез на реч. Разпознаването и извличането на данни се фокусира върху корпуси на ПО. За да направи това системата използва две онтологии (обща и конкретна за областта). Системата **SIM** цели разбиране на текстове, като предлага две семантични бази: на понятия и на изречения [Pinheiro'2010]. Концептуалната база съдържа понятия, дефинирани и съгласувани за ПО. Базата от шаблони на изречения съдържа модели, които имат синтактична структура и функционират като шаблони, чиито слотове се запълват с термини на естествен език.

Статистическият подход класифицира разпознати от ЕЕО езикови единици и прогнозира техните езикови характеристики и поведение. ИС от този тип се справят със задачата да трансформират потребителски въпрос във вектор в многомерно векторно пространство и отговорят на въпроси, зададени на ЕЕ. **SPYSE** е уеб базирана ИС, която търси и извлича информация от Python програми. SPYSE съдържа файл с дефиниции и изрази на Python, във вид на колекция от – класове, методи, променливи. Анализатор нормализира текста, преди индексиране, търси съвпадения и въз основа на тях класира резултатите [Imminni'2016]. Семантичното търсене в **SemanticFinder** е базирано на категоризиран контролиран речник със синоними и акроними. Речникът унифицира терминологията и улеснява търсенето и сравняването на обекти. Той прецизира и стеснява резултати от търсенето и предоставя достъп до история на търсе-нията, както и търсене чрез промяна на отделни параметри [Gonzalez'2016].

Х**ибридният подход**, комбинирано използва подходи, средства и инструменти за ЕЕО. В **USAS** лексикалните ресурси се конструират ръчно. Тя комбинира POS та-гер, лематизатор, семантични корпуси, шаблони, продукционни правила и програми, изпълняващи алгоритми за двузначност и присвояване на семантични тагове на думи-те в текущ текст [Rayson'2004]. Системата **PRECISE** интерпретира изречения с по-мощта на речници и семантични ограничения [Khapane'2015]. Разработена от [Popescu'2004], системата отговаря на потребителски запитвания, които започват с въпросителна дума – What, Who, When. Базата данни е съставена от три различни типа елементи: отношения, атрибути и стойности [Everling'2015]. **TiQi** е уеб-базиран ЕЕИ, който приема ЕЕ заявки, трансформира ги в SQL заявки и ги изпълнява, като връща опростена версия на генерирана SQL заявка към потребител. Извлечените данни се изобразяват в табличен формат [Lin'2017].

1.3. Изводи

Анализът на разгледаните ИС с ЕЕИ и решения на поставения проблем показа повишен интерес на изследователи по темата. В разгледаните ИС с ЕЕИ се откроиха **следните проблеми**:

- Сложност, нееднозначност на ЕЕ;
- Ограниченост на ПО;
- Ограниченост от вида на БД;
- Недостатъчно решения и механизми за ЕЕО на български език;
- Няма примери за ЕЕИ към ИС в Българското образование;
- Липсва методика за изграждане на ЕЕИ към различни ИС.

В дисертацията се следва **хибриден подход**, който адаптира и надгражда съществуващи подходи, използва съществуващи техники и инструменти за ЕЕО, като разширява и подобрява техния мащаб. Предложеният подход за създаване на ЕЕИ към ИС **включва**:

- Методика за създаване на специфични модели на ПО за ИС;
  - Създаване на модел на ПО;
  - Създаване на лингвистичен модел за ПО;
  - Създаване на модели на въпроси, подходящи за формиране на заявка;
- Софтуерно решение на проблема по транслиране на ЕЕ заявка в заявка към различни ИС.

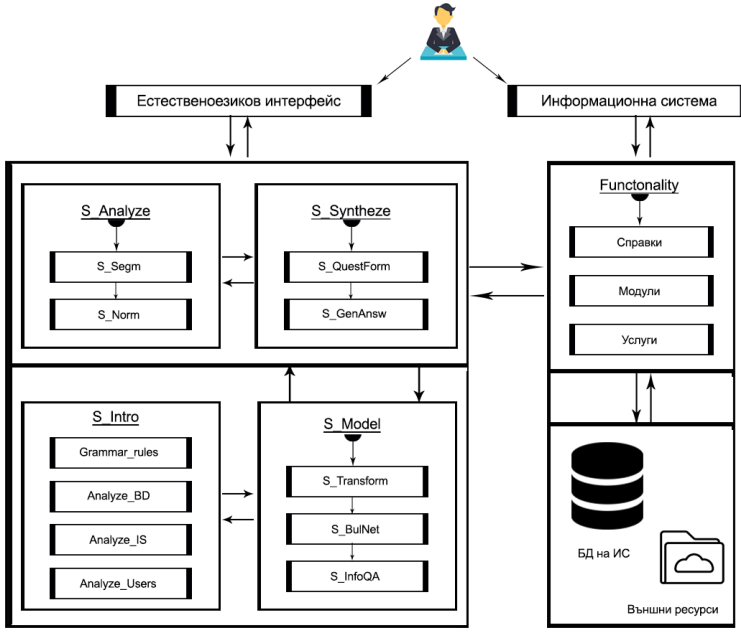
Предложената методика за моделиране на ПО и софтуерен инструмент могат да бъдат използвани при създаване на ЕЕИ за други ИС в същата ПО.

Глава 2. Моделиране и проектиране на ЕЕИ

На база на проведените анализ и заключения в Глава 1. **се откроиха основни етапи, дейности и елементи в процеса по създаване на ЕЕИ** към ИС. Предложената методика в Глава 2. решава проблеми, свързани от една страна с избора на лингвистичен модел за описание с езикови средства, обекти и релации в съществуваща БД на ИС, и от друга с построяването на модели за управление, които съвместно с лингвистичния модел водят до формиране на заявка и връщат коректен отговор на въпрос, зададен на естествен език към ИС.

2.1. Процес за създаване на ЕЕИ за ИС

На фиг. 1. е представен **модел на процес за проектиране на ЕЕИ към ИС**. В него основна роля заемат етапите – *S_Intro, S_Model, S_Analyze* и *S_Syntheze*, в които се извършват съществени за ЕЕИ дейности.



Фигура 1. Модел на процес по създаване на ЕЕИ

Някои означения, които се използват в текста на дисертацията, са:

- *QP* – шаблон на въпрос, който може да се зададе към ЕЕИ на системата;
- *S_Intro* – предварителен етап в процеса на моделиране на ПО;

- *Grammar_rules* – дигитализиране на първични данни от външни ресурси – норми, критерии и правила в граматиката на даден ЕЕ, включително спецификации и стандарти за ПО;
- *Analyze_User* – проучване на изискванията на потребителите на ЕЕИ;
- *Analyze_GP* – анализ на добри практики, свързани с работата на ЕЕИ;
- *Analyze_DB* – анализ на БД на конкретната ИС;
- *Analyze_IS* – анализ на функционалността на ИС;

• *S_Model* – компютърно моделиране на ПО;
- *S_Transform* – създаване на модел на ПО, с която е свързана ИС;
- *S_BulNet* – създаване на лингвистичен модел на ПО на ИС;
- *S_InfoQA* – създаване на набор от шаблони на въпроси към ИС.

• *S_Analyze* – осъществяване на ЕЕ анализ;
- *S_Sedm* – сегментира постъпилата заявка до списък от токени;
- *S_Norm* – нормализиране на получените токени до основна форма;

• *S_Syntheze* – формиране на заявка към ИС;
- *S_QuestForm* – извличане на информация, закодирана във входния низ, проверки за условни, логически и агрегатни функции и т.н.;
- *S_GenAnsw* – формиране на заявка към БД и получаване на отговор от ИС.

При създаването на софтуерен инструмент за ЕЕО на потребителски въпроси и извличане на отговори от ИС трябва да се отчете факта, че за обектите и техните характеристики съществува БД на ИС и тя не зависи задължително от предложената мето-дика за транслиране на потребителския въпрос в заявка към БД.

Методиката по създаване на **компютърен модел на ПО** за дадена ИС обхваща процесите в **етап *S_Model*** и включва следните **три стъпки:**

• *S_Transform* – създаване на компютърен модел на ПО, с която е свързана ИС. По същество представлява списък от понятия, описващи БД на ИС, в това число и релациите между тях, вкл. понятия, които са част от GUI, участващи в менюта, справки и др. Моделът на ПО описва отношения на свързаност, при-надлежност и близост между понятията;

• *S_BulNet* – създаване на лингвистичен модел от полученият списък, надграден с метаданни за граматични свойства, категории, синонимни редове и др.;

• *S_InfoQA* – създаване на набор от езикови шаблони на въпроси, на които системата с ЕЕИ ще отговаря. Обхванати са основните сценарии на въпроси и е формирана рамката на общите решения за формиране на заявка към БД.

Стъпки ***S_Transform, S_BulNet* и *S_InfoQA,*** в които се моделира ПО, са част от *S_Model*. Те допълват съществуващата базова информационна основа и предхождат програмното осигуряване. Компонентите, получени в резултат на стъпките в *S_Model* са достатъчни, за да функционира ЕЕИ и да връща отговор на зададен въпрос към БД на ИС. За езиково анотиране на понятията от ПО, могат да се използват съществуващи средства и инструменти за ЕЕО, анализ и генериране на лингвистична информация.

Обработките в ***S_Analyze*** и ***S_Syntheze*** се реализират изцяло в софтуерната рамка на ЕЕИ. Програмните обработки в ***S_Analyze*** са свързани с граматичен анализ и извличане на метаданни за понятията от потребителския низ, необходими за по-ната-тъшните разсъждения. Финалният **етап *S_Syntheze*** обхваща процеса по формиране на заявка към БД.

2.2. Моделиране на ЕЕИ

Създаването на ЕЕИ за дадена ИС изисква да се предложат и изследват съответни модели и тяхната конкретна реализация за всеки от елементите в етап *S_Model* от фиг.1. За дигитализация на ПО са необходими: а) знания за ЕЕ и връзката му с обекти в БД; б) метаданни, които описват понятия за ПО; в) шаблони на въпроси.

2.2.1. Моделиране на ПО

За създаване на концептуален **модел на ПО** е необходимо да се обобщят и опи-шат понятия от ПО, като се определи мястото и ролята им в БД на ИС и поведението и взаимодействието им с други обекти в нея.

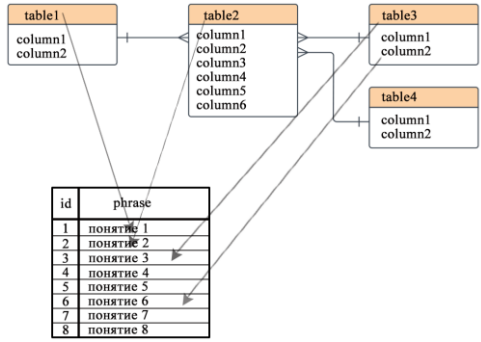
Вход: Информационна система за дадена преметна област.

Изход: Концептуален модел на ПО, база данни от понятия, описващи ПО.

Създаването на **Модел на ПО** е част от етап *S_Model* (вж. фиг. 1.) и се попълва след осъществен анализ на:

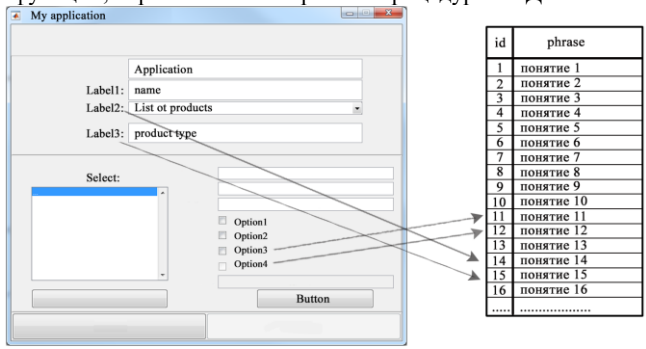
- БД на ИС (вж. фиг. 2.);
- съответната ИС (вж. фиг. 3.).

След направения в етап *S_Intro* **анализ на БД** на ИС, полученият списък с понятия, описва какво стои зад името на всяка таблица и колона от БД на ИС (вж. фиг. 2.). Моделът на ПО се разширява, подобрява и обогатява с всяка промяна в БД на ИС.



Фигура 2. Попълване на модела на ПО с понятия от БД на системата

Самата ИС също може да допълни списъка с понятия с фрази, участващи в GUI, менюта, контроли, справки, филтри и т.н (вж. фиг. 3.). От интерфейса на ИС се откроя-ват езикови фрази, носещи калкулирана, филтрирана и т.н. информация, получена с помощта на програмни функции, справки и/или съхранени процедури в БД.



Фигура 3. Попълване на модела на ПО от интерфейса на системата

В полученият **модел на ПО** езиковите понятия се сдвояват с именувани обекти от БД на системата [Zhekova'2021B]. Така се осъществява връзката на системата с ЕЕ. За тази цел са избрани категории свойства (*TableN, ColumnN, DefaultColumn, Aggr_fun, Value и т.н.*), които съответстват на роля и място, което заема обект в БД на ИС. По този начин, концептуалният модел на ПО задава скелета на БД на ЕЕИ, чрез определя-не на йерархия от класове и свойства, както и посоките на наследяване и откриване на релационни връзки между обектите.

property	description
TableN	Име на таблица
ColumnN	Име на колона
DefaultColumn	колона, която се отъждествява с таблицата
Aggr_fun	Агрегатна функция
Function	Функция – определена, специфична
Value	Стойност
isColumnOf	Отношение между ColumnN и TableN
relatesColToTable	Връзка между две таблици
DataType	Указва типа данни на всеки атрибут

Таблица 2. Свойства за описание на елементи от БД на ИС

Таблица ***Domain_area*** съхранява информация за понятия от ПО, като е възможно те да се дублират (вж. табл. 3.), поради факта, че: (а) едно и също понятие или фраза може да присъства като запис в няколко таблици; (б) няколко таблици може да съдър-жат поле с едно и също име.

id	phrase	db_element	property	parent_id	data_type
1	Понятие 1	table1	TableN	-	Text
2	Понятие 2	table1.column2	ColumnN	1	Text
3	Понятие 3	table1.column4	ColumnN	1	Text
4	Понятие 4	table2	TableN	-	Number
5	Понятие 5	table2.column2	Value	4	Number
		table1.column2	Value	1	Number
..	..	..	..	..	..
..	колко	count	Aggr_fun	-	Number
..	минимален	min	Aggr_fun	-	Number
..	максимален	max	Aggr_fun	-	Number
..	..	..	..	..	..
n	Понятие n	table2.column1	DefaultColumn	4	Number
		table3.column4	DefaultColumn	..	Text

Таблица 3. Таблица Domain_area, описваща модела на ПО

Езиковите единици, влизащи в състава на display клаузите, кванторите за отрица-ние, агрегатни и собствени функции, математически и логически оператори и т.н. (вж. табл. 3.), също намират място в модела на ПО [Жекова'2019В]. Полученият списък от понятия в таблица ***Domain_area*** ще наричаме **Модел на ПО.** За друга ИС в същата ПО специфичните езикови изрази лесно биха могли да се попълнят.

2.2.2. Лингвистично моделиране

Втора стъпка в етап *S_Model* е лексикалното и декларативно описание на поня-тията в таблица ***Domain_area***, получени при създаване модела на ПО.

Вход: Моделът на ПО, получен в етап *S_Transform*.

Изход: Лексикално, лингвистично описание на понятията от модела на ПО, т.нар. лингвистичен модел (редуцирана версия на BulNet).

Полученият **лингвистичен модел** от знания не е статичен, нерелевантен и представен от несвързани записи от данни и играе ключова роля в работата на ЕЕИ към ИС. Смисловите единици (думи, фрази) в него си взаимодействат чрез устойчиви синонимни релации помежду си. Той е отворен за разширяване и свободен за преизползване от други ЕЕИ в същата ПО. **Синонимните редове** в лингвистичният модел могат да бъдат попълнени **автоматично**, при наличие на технически средства (инструменти) с отворен достъп за извличане на тази информация, или ръчно с помощта на експерти в областта на лингвистиката.

Метаданни	
Понятие	Понятие – *Phrase*
	Основна форма – *Lemma*
	Синонимен ред – *Synsets*
	Дефиниция – *Definition*

	Принадлежи към – *Parent*
	Сематична роля – *Role*
	Синтактичен клас – *Part Of Speech (PAS)*
	Въпросителни думи – *Who, What, Which*

Таблица 4. Граматически атрибути на понятие от ПО

Лингвистичният модел има пълно съответствие с модела на ПО, т.е. таблица *Lexicon* припокрива изцяло таблица *Domain_area*.

id	phrase_id	synsets	pos
1	Понятие 1	синоним1 @@ синоним2 @@ синоним3	Adj
2	Понятие 2	синоним на понятие 2	P
3	Понятие 3	синоним на понятие 3	N
4	Понятие 4	синоним на понятие 4	Adv
..	….	….	..
..	колко	брой @@ общ брой @@ сума @@ обобщи	QW
..	минимален	най-малко @@ мининум @@	QT
..	максимален	най-голямо @@ най-големия @@ максимум	QT
..	….	…	…

Таблица 5. Таблица Lexicon – лингвистичен модел на ПО

В състава на концептуалния и лингвистичния модел на ПО влизат понятия свързани с ПО. Това прави моделите **универсални** за употреба в различни ИС. Естествено всяка ИС има своите специфики и тогава **моделите могат да се преизползват** без да се разработват повторно, единствено съответствията на понятия с обекти от БД, в модела на ПО, се правят за всяка конкретна ИС. Това обуславя **акумулативния характер на моделите**. В случай че ЕЕИ се внедри към платформа, обединяваща данни за различни висши училища, и съществуват предварително създадени съответствия между понятия и обекти в съответните БД, то ЕЕИ ще бъде в състояние да отговаря на потребителски запитвания за всички висши училища.

2.2.3. Моделиране на въпроси

Анализа на постъпилият ЕЕ въпрос позволява да се генерира идея за модел на съответна SQL конструкция, отговаряща на неговата структура. Моделите на въпроси не зависят от конкретната БД и могат да се разширяват и усъвършенстват при необходимост.

Вход: Шаблони (регулярни изрази) на потребителски въпроси.

Изход: Модели/видове SQL конструкции към БД.

Избраният подход при моделиране на потребителски въпроси кореспондира с търсене на структура и намиране на възможни общи модели за проектиране на заявки и предоставя рамка за подходящите общи решения. Зад тези общи модели на заявки стоят алгоритми и евристики за обработка на множество лингвистични събития.

Този компонентен модел има за задача да формализира управляващи последователности, които ще наричаме **шаблони на потребителски въпроси (QP)**.

Списъкът с езикови шаблони на възможните конструкции на SQL заявки към БД може да се развива и актуализира с всяка ИС, ако естеството на задачата го изисква [Zhekova'2021C]. Предложените архитектури на въпроси са проектирани достатъчно общо, че да могат да обхванат всички възможни конструкции на заявки към БД. В случай, че останат езикови интерпретации, които не са открити, а в последствие се утвърдят като такива, QP моделите могат да се актуализират и развият във времето.

Шаблон на въпрос	SQL заявка	QP
Кои са / Изведи / Намери / Покажи [списък на] [всички / всичко за] X?	SELECT * FROM X	QP1

Кои са / Изведи / Покажи / Намери A1 [A2, A3,…] [от / за / в / за] X?	SELECT A1, A2, ... FROM X	QP2
Кои са / Изведи / Покажи / Намери [всички] X за които [е изпълнено] A1='value_1' [and | or A2='value_2']...	SELECT * FROM X WHERE A1='value_1' and (or) A2='value_2' ...	QP3
Кои са / Намери / Покажи / Изведи C1 [и | ,] C2 [и | ,]…. [от | на] X [за | от] [които] [е изпълнено] A1='value_1' [и | или | ,] [A2='value_2'],….	SELECT C1, C2, C3.... FROM X WHERE A1='value_1' and (or) A2='value_2'	QP4
Кои са / Намери / Покажи C1 [и |, C2] [от | на] X [и | ,] D1 [и | , D2]… [от | на] Y [за | от] които [е изпълнено] A1='value_1' [и | или|,] [A2='value_2']	SELECT X.C1, X.C2,…. Y.D1, Y.D2,….. FROM X WHERE A1='value_1' and A2='value_2' and	QP5
Кои са / Намери / Покажи [всички] [от | на] X [и | ,] [всички] Y [за | от] които [е изпълнено] A1='value_1' [и | или | ,] [A2='value_2'], ….		

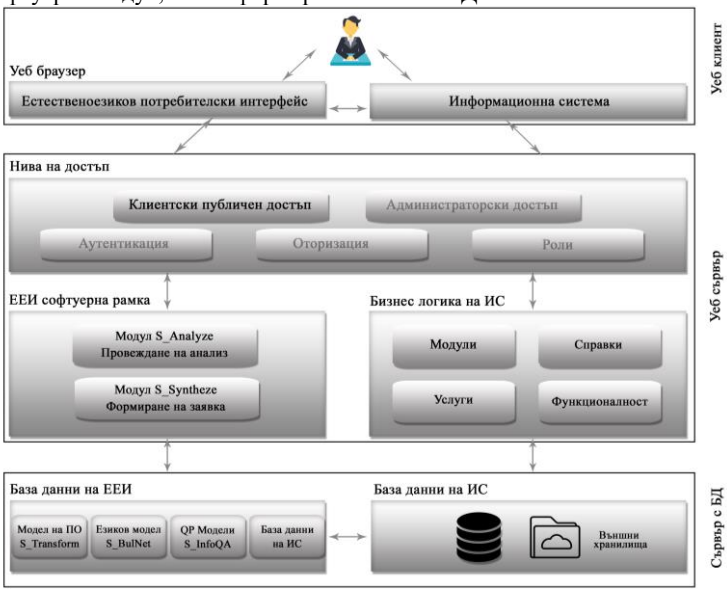
Таблица 6. Езикови конструкции и условия за приложимост на QP3

Предложеният набор от шаблони на въпроси осигурява цялостен **подход за разбиране значението на потребителските запитвания**.

2.3. Архитектура на ЕЕИ към ИС

Основните компоненти в общата архитектурна рамка на предложения ЕЕИ към ИС може да се класифицират по следния начин:

- Компютърни модели, които генерират съдържание за ЕЕИ към ИС;
- Софтуерен модул, който се грижи за управлението на метаданни в ЕЕО;
- Софтуерен модул, който формира заявка към БД.



Фигура 4. Архитектура на ЕЕИ към информационна система

Инстанции на компонентите, генериращи съдържание се създават автоматично или ръчно, обработват се от модула, който управлява и доставя метаданни за процеса по ЕЕО, резултата от който се изпълнява в модула, формиращ заявка към БД. Схема на архитектурата на софтуерния инструмент с ЕЕИ е представена на фиг. 4.

Архитектурната рамка демонстрира логическия процес по обработка на потребителски въпрос от ЕЕИ и връщане на отговор от БД на информационна система.

Базовите компоненти в архитектурната рамка са: *уеб клиент, уеб сървър* и *сървър с БД*.

В уеб базирани приложения браузърът е този, който изпълнява ролята на **web client**. Web server служи като посредник между уеб-клиента и сървъра с базите данни. Потребителите изпращат своите заявки от уеб-браузер през ЕЕИ, който приема потребителски въпрос, след което препраща заявката към **web server**. Той го насочва към модулите в софтуерната рамка, изчаква да приключат back-end обработките и връща отговор до потребителя. **Сървърът с БД** предоставя ресурси, необходими за потребителите. Той включва компоненти, модели и външни ресурси за съответния ЕЕИ и съществуващата БД на ИС.

2.4. Изводи

Представен е общ модел на процес по създаване на ЕЕИ към ИС. Предложена е методика за моделиране на ПО, която може да се пренесе за коя да е ПО и нейната връзка с естествен език. По този начин предложеният модел за ЕЕИ към ИС има познания както за ЕЕ, за интерпретиране на потребителски заявки, така и за начина на формиране на заявки към БД на ИС. Моделите в предложената методика, може да интегрират данни от външни източници – инструменти, ресурси, електронни таблици.

Представеният ЕЕИ за обработка на потребителски текст-заявки на ЕЕ не е единственият подобен инструмент, но навярно е ***единствен с поддръжка и обработка на български език*** и е ***единственият – адекватен на съвременната българска грама-тика и документация***.

Глава 3. Софтуерна реализация на ЕЕИ

Описаният модел на процес за създаване на ЕЕИ към ИС предполага използва-нето на софтуерен инструмент за автоматизиране на включените в него процеси и обработки. В настоящата глава главно място заемат описание на алгоритми, функции и обработки в софтуерната реализация на ЕЕИ към ИС, който с помощта на средства за естественоезиков анализ се справя със задачата да извлече информация под формата на отговор на потребителски въпрос.

3.1. Технологии и инструменти за разработка

За софтуерната разработка на ЕЕИ са използвани различни технологии. Описа-ната интеграция на модули за ЕЕО предполага съвместна работа на няколко различни технологии и езици за програмиране.

Клиентският интерфейс е постигнат с интегрирана **среда за разработка** на компанията Microsoft – Visual Studio 2019 и **език за програмиране** от високо ниво, с многобройни библиотеки – **C#**. За софтуерния прототип с ЕЕИ използва достъпна **среда за изпълнение** с отворен код за всички операционни системи – ASP.NET Core. Кли-ентският интерфейс е реализиран като ASP.NET приложение. Интеграцията и **упра-влението на БД** е възможно благодарение на Microsoft SQL Server. С помощта на SQL Server Menagement Studio са създадени тестовите прототипи на БД в Глава 4.

В началото целия програмен код беше реализиран изцяло на C#, но в последствие класовете отговорни за ЕЕО в C# бяха заменени с **командно приложение** – скрипт, написан на езика **Python**. В web интерфейса става извикването на Python **интерпретатора със съответния скрипт**. Резултата от него се връща в **JSON обект** и се предава на web приложението. Python е избран, заради всеобхватния набор от библиотеки, които предлага и функционалността му в обработката на лингвистични данни. Използ-вани са и **собствени ресурси**, които подобряват функционалността на приложението по извличане на неявна информация, скрита във въпроса.

Реализацията на софтуерния инструмент се базира на предложената архитек-тура

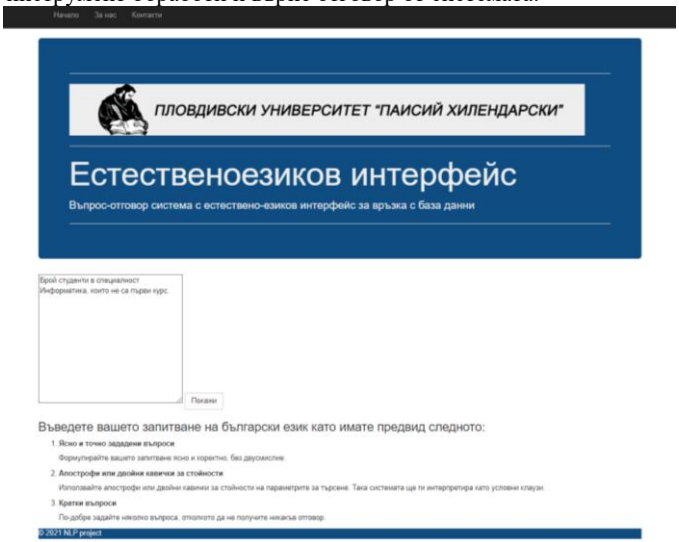
на ЕЕИ към ИС. За комуникация с потребителите ползва **клиентски интерфейс**, а програмната реализация обединява следните **два модула**:

- **Модул S_Analyze** *се грижи за:*
  - Доставяне на метаданни за езикови обекти;
  - Обработка и анализ на получените лингвистични данни;
- **Модул S_Syntheze** *отговаря за:*
  - Формиране на заявка от явната и неявна информация във въпроса;
  - Извличане на отговор от БД на конкретна ИС.

3.2. Дизайн на естественоезиков интерфейс

Клиентският интерфейс е входна точка за потребители на приложението и връз-ката им с ИС, към която те отправят своите запитвания на ЕЕ. Дизайнът е опростен и лесно достъпен за масовият потребител и за създаването му са използвани възможнос-тите на езика C# и .NET framework. Интерфейсът е минимизиран и предоставя *входно текстово поле* за задаване на въпрос, *бутон за изпращане* към сървъра след извършване на необходимите back-end обработки и *поле за резултати*, върнати до потребителя.

Основното поле за въвеждане на запитвания (вж. фиг. 5.) е стандартно многоре-дово TextBox поле и допуска въвеждане на текст до 10 реда с вертикална лента за превъртане (vertical scroll bars). Бутона „Покажи“ е стандартен *input* елемент от тип *button*. В **информационното поле** резултата от потребителското търсене се показва тогава, когато софтуерния инструмент обработи и върне отговор от системата.



Фигура 5. Потребителски интерфейс на ЕЕИ

В случай, че софтуерният инструмент не успее да синтезира заявка в **полето за грешки** се изписва евентуална причина за липсата на отговор. Причините биха могли да бъдат: двусмислено или неточно формулиран въпрос, без основен субект или няма данни на поставеното условие.

В табл. 7. са дадени предварително дефинирани изрази, генерирани в зависимост от резултата от работата на процедури в модул *S_Syntheze*. Във всички останали случаи, когато ЕЕИ успешно генерира заявка към БД с еквивалентни съответствия на идентифицираните във въпроса елементи се счита за успешен.

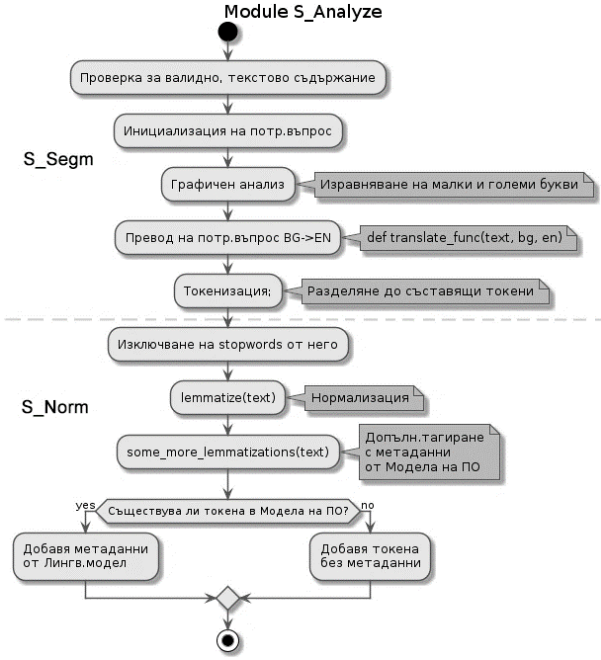
Крайни състояния	Отговори
Заявката се изпълнява	Резултат от заявка към БД
Невалидна заявка	Невалидна заявка. Откритите понятия не отговарят на обекти в БД.
Корекция на запитване	За съжаление няма данни в БД. Моля, преформулирайте!
Перифразиране	Въпроса е двусмислен. Моля, преформулирайте въпроса!

Таблица 7. Предварително дефинирани отговори за обратна връзка с потребител

3.3. Модул S_Analyze

Задача на софтуерният инструмент е да преведе въпрос на ЕЕ до SQL заявка, като извършва търсене в удобно структурирана релационна БД на ИС с богато съдържание, в която намират място търсените отговори, свързани с идентифицираните параметри.

Модулът S_Analyze е ключов елемент и ядро в софтуерната рамка на ЕЕИ. Модулът борави с информация от предварително изградени модели от понятия за ПО, етикетирани с метаданни, описани с официалната граматика на езика. Поради факта, че неявната информация за понятията от ПО е свързана с техни характеристики е необходимо да се приложат методи за доставка на метаданни, обработка и анализ.



Фигура 6. Логическа схема на обработки в модул S_Analyze

Фокус в процеса по доставяне на необходимата информация е да се идентифи-цират маркери, които обозначават обекти, състояния или отношения, представляващи интерес за потребителя. ***Модул S_Analyze*** включва подмодули *S_Segm* и *S_Norm.* Програмните обработки в двата подмодула са изобразени на фиг. 6.

За дейности, свързани с анализ и ЕЕО могат да се ползват и външни софтуерни **средства и инструменти** (езикови корпуси, напр. WordNet, синонимни речници и др.). Тези технически средства зависят от вида и спецификата на всеки естествен език. Софтуерната реализация на *S_Segm* и *S_Norm* (и съответните програмни обработки) също **зависят от езика на запитванията** и на очакваните отговори. Обработките в тях могат да се реализират по **три начина**:

- **BG – > EN** – входния текст се превежда до EN език и обработките продължават на EN език, тъй като за EN език има много средства и инструменти за ЕЕО;
- **BG – > BG** – входния текст си остава на BG език, при наличие на инструменти и средства за ЕЕ анализ и обработка на български език. Обработките продължават на BG език;
- **EN – > EN** – входния потребителски въпрос се задава на EN език. Тогава няма нужда от превод и отново се ползват съществуващите средства за ЕЕО за EN.

В **общия случай** може да се ползва първата стратегия. При него предложеният подход за създаване на ЕЕИ към ИС не зависи от езика на потребителските запитва-ния. Така методиката за моделиране на ПО както и разработеният софтуерен инстру-мент добиват **универсален характер**.

В дисертацията е решено ЕЕ анализи и обработки в двата софтуерни модула да се извършват на английски език, защото за него има достатъчно средства и инструменти за естественоезикова обработка.

Реализацията на процесите по обработка и анализ на езиковите единици в подмодул ***S_Segm*** започва с постъпване на естественоезиков въпрос. Задача на гра-фичният анализ е да се разпознаят границите на отделни изречения от входния текст. Като формални разделители служат сигнали като интервали, главни букви, препина-телни знаци, абзацни отстъпи и др. При условие, че текста е съставен от повече изрече-ния се извършва **сегментация (разделяне)** до отделни изречения, а изреченията до отделни текстови елементи.

В ***S_Norm*** се извършва анализ, който включва **нормализация** (лематизация) на получения низ дума по дума с метод *lemmatize(word)*, който използва речник WordNet, за да сведе всеки токен до основната му форма. WordNet съдържа и списък на понятия (*synsets*) с една и съща концепция.

В *S_Analyze* се определят вида и отношенията между получените токени. Диференцираните лексеми се тагират с таг : *tn* – маркер за име на таблица БД; *cn* – име на колона в таблица; *val* – маркер, който аташира стойност към атрибут; *dc* – default колона; *aggr_fun* – агрегатна, вградена функция; *fun* – специфична за контекста функция. В края на *S_Analyze* лингвистичните знания за вход-ните токени се заменят с обекти от БД, което прави възможно формирането на заявка.

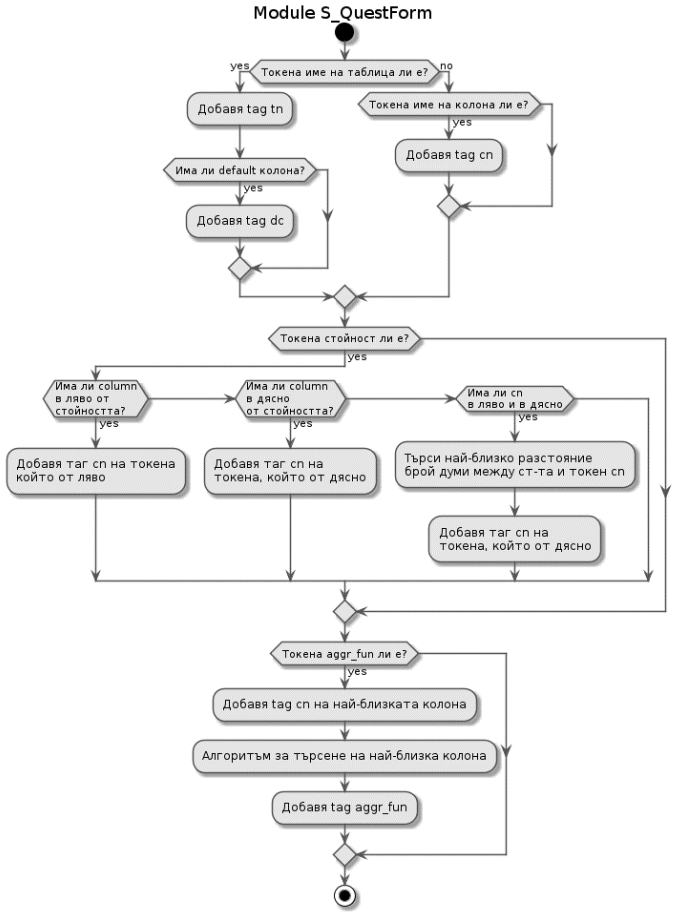
3.4. Модул S_Syntheze

В **подмодул *S_QuestForm*** се търсят релативни отношения и асоциативност между получените токени. Проверява се за наличие на съединителни и дисплей клау-зи, агрегатни функции, условности и свързани с тях математически оператори, числа, дати и стойности на атрибути, формиращи конструкции на заявка към БД.

За токени, маркирани с таг *val* или *aggr_fun* се проверява за наличие на най-близко стоящ токен *cn* в низа, към който да се приложат. Логическата схема от фиг. 7. представя реда на програмните обработки и условията за маркиране на думите в потребителския низ.

Алгоритъм за търсене на токен *cn* в близко разстояние до токен *val* или *aggr_fun*:

- Проверява се разстоянието между текущия токен и намерените имена на коло-ни от двете страни на входния низ;
- Сравнява броя думи между текущия и намерения *cn* токен в ляво;
- Сравнява броя думи между текущия и намерения *cn* токен в дясно;
- Ако разстоянието до левия токен *cn* е по-малко, предпочита него;
- Ако разстоянието до десния токен *cn* е по-малко, предпочита него;
- Ако са еднакво близки, предпочита десният токен.

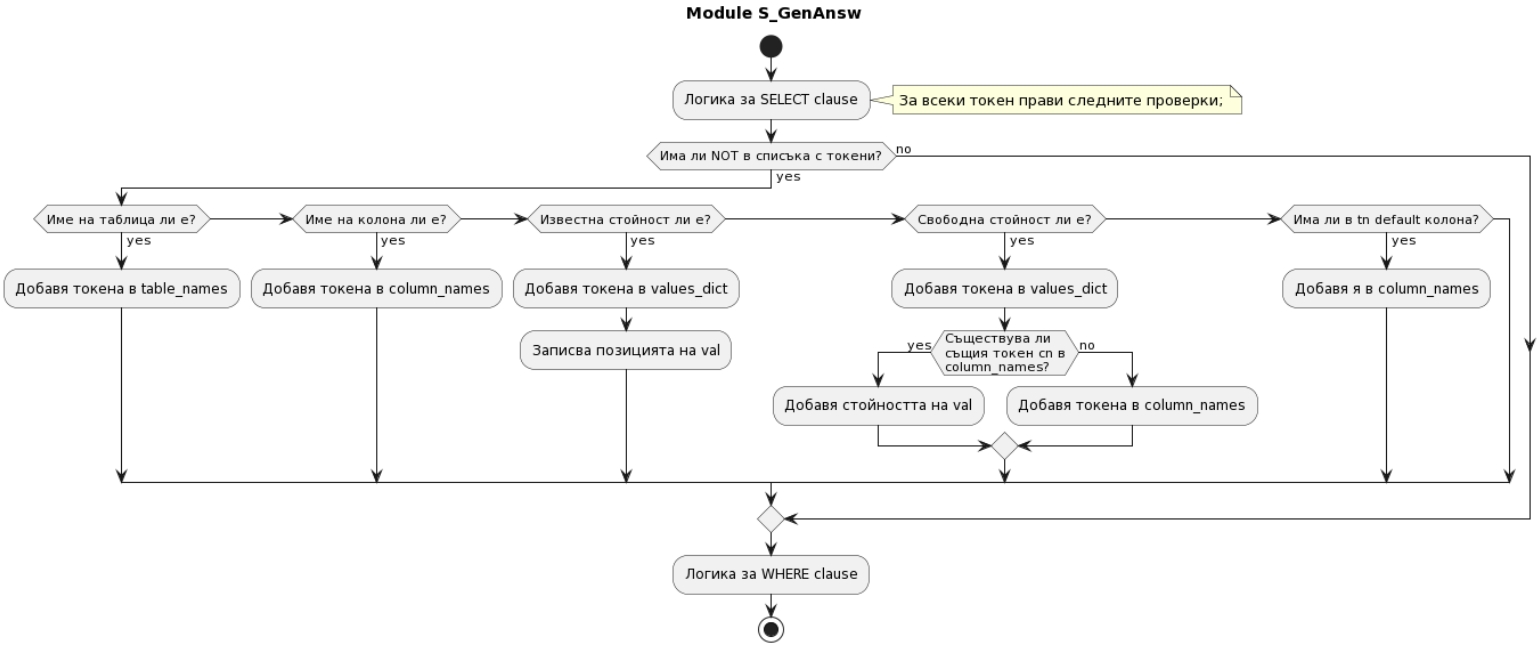


Фигура 7. Логическа схема на обработките в подмодул S_QuestForm

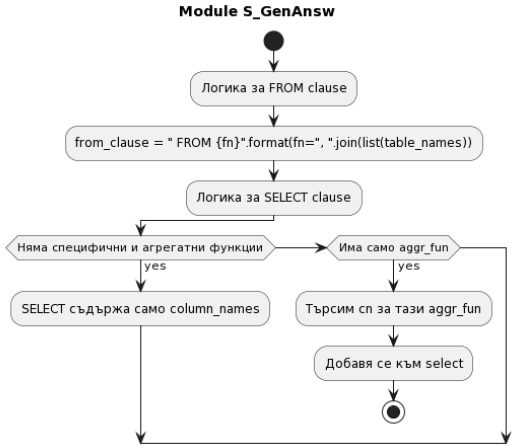
Подмодул *S_GenAnsw* е натоварен със задачата да формира заявка към БД, отговаряща на структурата и елементите от въпроса и кодираната информация в тях. Тук се извършва „срещата“ между елементи, изграждащи заявките и структурните шабло-ни на заявки към БД, които наричаме модели на въпроси.

Програмната логика включва проверка за наличие на квантор за отрицание NOT. Позицията на NOT в последователността от елементи е важна от гледна точка на това, всички стойности, колони, функции, които бъдат открити след него трябва да влязат в заявката след NOT, т.е. в негативна релация, а в случай, че не бъде открит условията се удовлетворяват в положителна посока.

В резултат на дейности в *S_GenAnsw* се получава списък с имена на таблици – *table_names*, като в *key* се съхраняват именуваните обекти, а във *value* – за улеснение се запазва поредността на намерените имена на таблици *1, 2,...* Аналогично имената на колони се натрупват в списък *column_name*. При съвпадение в *column_name* се запазват имената на откритите колони, като стойността, записана в *key* има вида *tableName.columnName*.

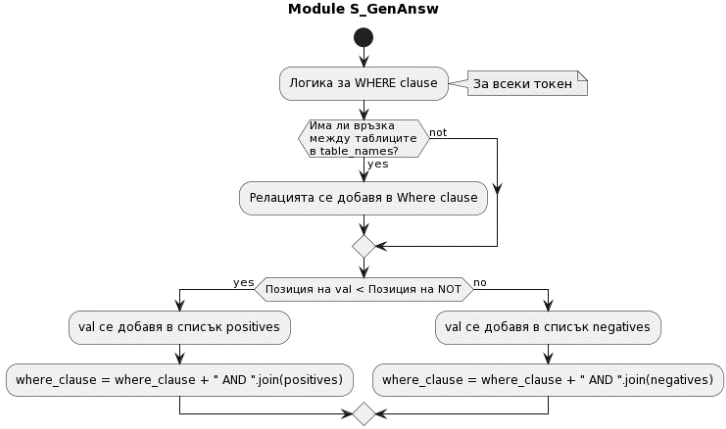


Фигура 8. Логическа схема на подмодул S_GenAnsw за синтез на Select clause



Фигура 9. Логическа схема на подмодул S_GenAnsw за синтез на From клауза

В метаданните има предварителна информация за **релационните връзки** между таблиците. Програмно това е реализирано като хеш-таблица, за по-добро търсене. В нея данните са представени като **ключ-стойностни** двойки. Всички възможни връзки между таблици, подредени по азбучен ред оформят ключовете, като не се допуска повторение по key, а value (стойностите) са изрази, които се поставят в тялото на join. Ключът се използва като индекс за достъп до стойностите. Имената на таблиците идентифицирани от входния низ ги има в dictionary table_names[]. Проверява се дали има такъв ключ – подредени по азбучен ред таблици и се взема неговата стойност – израза, който влиза в Join конструкцията. Фиг. 10. показва реда в логическите стъпки, при формиране на Where клаузата на заявката.



Фигура 10. Логическа схема на подмодул S_GenAnsw за синтез на Where клауза

Окончателната заявка към БД съдържа толкова таблици, колкото са елементите в *table_names*, толкова колони, колкото са атрибутите в *column_names*, вградени или калкулирани функции, както и условия с положителен или отрицателен смисъл за стойностите в *values_dict*, ако има такива, идентифицирани в семантиката на входния потребителския въпрос.

3.5. Изводи

Реализацията на софтуерният модул е част процес по създаване на ЕЕИ към ИС. Предложен е алгоритъм за неговата програмна реализация. Езикът за програмиране и използваните технологии не ограничават решението. Представената софтуерна идея и реализация за семантично търсене и извличане на информация чрез ЕЕИ може да **се внедри абсолютно независимо** към всяка друга ИС функционираща в същата ПО.

Глава 4. Експерименти по създаване на ЕЕИ към ИС

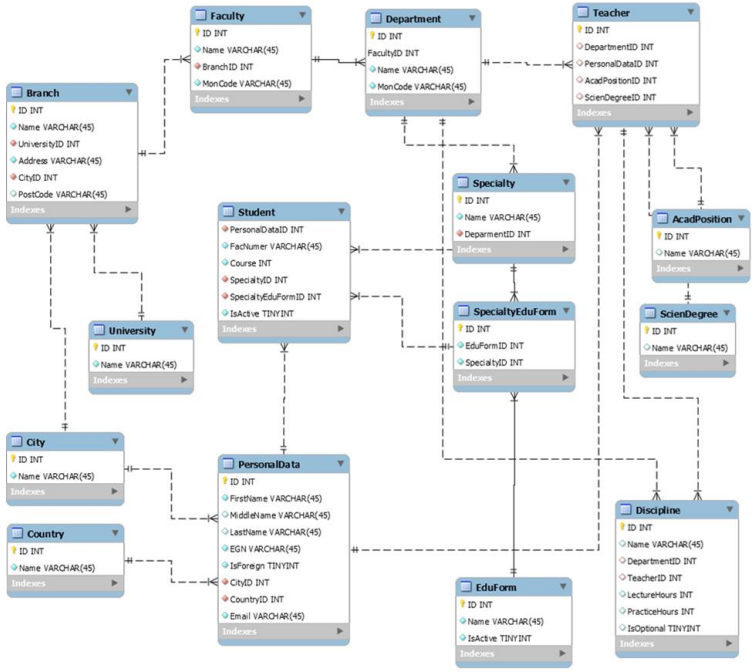
В 4 глава общите модели от Глава 2 се конкретизират и прилагат в създаване на **2 експериментални ЕЕИ** към прототипи на БД за **различни ИС в** област ВО.

4.1. Експеримент 1. ЕЕИ към БД на еУниверситет

Информационната система в Експеримент 1 е от тип ‚електронен университет'. Базата данни, върху която са насочени разсъжденията и тестовете е *dbUniversity* (вж. фиг. 11.). Тя се състои от таблици: студент, преподавател, дисциплина и т.н. и отношения между тях, базирани на процеси, протичащи в областта на ВО.

4.1.1. Моделиране на ЕЕИ

За *dbUniversity* се моделира ПО по стъпките от предложената в Глава 2 методика. В резултат от **стъпка *S_Transform*** се получава таблица ***Domain_area*** – списък от понятия, влизащи в състава на ИС и нейната БД, т.нар. **модел на ПО.** Изграждането на списъка, тръгвайки от БД е лесно осъществимо, тъй като структурният модел пресъздава реални обекти и ситуации за ПО. Моделът на ПО е попълнен със стандартна терминология от съвременния български език и позволява интеграция, разширяване и актуализация на данните и избягва дублирането.



Фигура 11. Част от dbUniversity

В резултат от **стъпка *S_Transform*** се получава таблица ***Domain_area***, която съдържа йерархично подредени понятия, описващи ПО, свързани със съответните им обекти в БД

на ИС.

id	phrase	db_elements	property	parent_id	data_type
1	2	3	4	5	6
1	университет	University	TableN	-	Text
2	факултет	Faculty	TableN	-	Text
3	дисциплина	Discipline	TableN	-	Text
4	задължителна	Discipline.IsOptional	ColumnN	5	Text
5	студент	Student	TableN	-	Text
6	курс	Student.Course	ColumnN	8	Number
7	номер	Student.FacNumber	ColumnN	8	Number
8	факултетен	Student.FacNumber	ColumnN	8	Number
9	преподавател	Teacher	TableN	-	Text
10	план	uPlan	TableN	-	Text
11	лекции	uPlan.LectureHour	ColumnN	13	Number
12	редовно	EduForm.Name	Value	18	Text
13	задочно	EduForm.Name	Value	18	Text
14	студентски статус	StudentStatus	ColumnN	8	Number
15	прекъснал	Student.Status	Value	34	Number
16	напуснал	Student.Status	Value	34	Number
17	магистър	EduDegree.Name	ColumnN	37	Text
18	Информатика	Specialty.Name	Value	46	Text
19	име	Teacher.Name	DefaultColumn	12	Text
20	име	Specialty.Name	DefaultColumn	4	Text
21	име	Discipline.Name	DefaultColumn	5	Text
..	…	..	..	..	..

Таблица 8. Модел на ПО за dbUniversity

В резултат от **стъпка *S_BulNet*** се получава таблица ***Lexicon*** от езиково аноти-рани думи и фрази и категории за семантична еквивалентност, които обединяват думи-те в синонимни редове, т.нар. **лингвистичен модел за ПО**. Таблица ***Lexicon*** за *dbUniversity* изглежда по следния начин:

id	phrase	synsets	pos
1	2	3	4
1	университет	ВУЗ, институция, институт, висше училище	N
2	факултет	звено, отдел	N
3	дисциплина	предмет	N
4	задължителна	по учебен план	Adj
5	студент	учащ, обучаем, обучаван	N
6	курс	ниво, степен	N
7	номер	номер, факултетен номер	N
8	факултетен	албумен	Adj
9	преподавател	учител, лектор, педагог	N
10	план	програма, учебен план	N
11	лекции	лекционен курс, лекционни часове	N
12	редовно	нормално, постоянно	Adj
13	задочно	неприсъствено	Adj
14	договор	трудов договор	N
15	основен	основен, първи	N
16	граждански	действащ	Adj
17	прекъснал	нередовен	Adj
18	напуснал	недействителен	Adj
19	действащ	редовен, активен	Adj
20	област	сфера, професионална област	N
..	…..	……	….

Таблица 9. Лингвистичен модел на ПО за dbUniversity

Моделите на въпроси в **стъпка *S_InfoQA*** в Експермент 1 съвпадат с шаблоните, представени в раздел 2.2.3.

4.1.2. Тестове

Проведени са **три теста** от 20 въпроса с база данни *dbUniversity*. **Тест 1** съдържа 20 въпроса. В **Тест 2** бяха тествани 20+20 въпроса. **Тест 3** обхвана 40 + 20 въпроса.

Повторението на първоначалните запитвния се обосновава с необходимостта от потвърждение верността на получените резултати.

В табл. 10. са представени част от резултати от **Тест 1**, проведен с 20 въпроса към *dbUniversity*. Междинните обработки в процеса на трансформиране дават видима представа за работата на всяка отделна функция.

Потребителски въпрос	Лематизация	Тагиране на фрази	Допълнително тагиране	Крайна заявка
Съотношението между брой и максимална възраст на студенти, които се казват Васил и са първи курс и не са специалност Информатика.	ratio number maximum age student_name vasil first course not specialty informatics	fun(ratio) aggr_fun(count) aggr_fun(max) cn(age) cn(username) vasil first cn(course) negation(not) tn(specialty) val(informatics)	fun(ratio) aggr_fun(count,cn=age) aggr_fun(max,cn=age) cn(age) cn(username) val(vasil,cn=name) val(first,cn=course) cn(course) negation(not) tn(specialty) val(informatics)	SELECT count(age)/max(age) FROM users, student, specialty WHERE users.id = student.user_id AND student.spec_id = specialty.id AND users.username like('%vasil%') AND student.course like('%first%') AND NOT (specialty.specname like('%informatics%'))
Намери имената на студентите, които са в първи курс.	student_ name first course	cn(username) first cn(course)	cn(username) val(first,cn=course) cn(course)	SELECT users.username, student.course FROM users, student WHERE users.id = student.user_id AND student.course like('%first%')
Намери максималния курс на студенти, които са в специалност Информатика.	max course student specialty informatics	aggr_fun(max)cn(course) tn(student) tn(specialty) val(informatics)	aggr_fun(max,cn=course) cn(course) tn(student) tn(specialty) val(informatics)	SELECT max(course) FROM student, specialty WHERE student.spec_id = specialty.id AND specialty.name like('%informatics%')
Брой студенти в специалност Информатика, които не са първи курс.	number student informatics not first course	aggr_fun(count) tn(student) val(informatics) negation(not) first cn(course)	aggr_fun(count,cn=course) tn(student) val(informatics) negation(not) val(first,cn=course) cn(course)	SELECT count(course) (fac.number) FROM student, specialty WHERE student.spec_id = specialty.id AND specialty.specname like('%informatics%') AND NOT (student.course like('%first%'))
Намери факултетните номера на студенти, които са втори курс и не се казват Васил.	faculty_number student second course not name vasil	cn(faculty_number) tn(student) second cn(course) negation(not) cn(username) vasil	cn(faculty_number) tn(student) val(second,cn=course) cn(course) negation(not) cn(username) val(vasil,cn=username)	SELECT student.faculty_number, student.course, users.username FROM student, users WHERE users.id = student.user_id AND student.course like('%second%') AND NOT (users.username like('%vasil%'))
Кои са факултетните номера на студенти, които са втори курс и не се казват Васил.	faculty_number student second course not name vasil	cn(faculty_number) tn(student) second cn(course) negation(not) cn(username) vasil	cn(faculty_number) tn(student) val(second,cn=course) cn(course) negation(not) cn(username) val(vasil,cn=username)	SELECT student.faculty_number, student.course, users.username FROM student, users WHERE users.id = student.user_id AND student.course like('%second%') AND NOT (users.username like('%vasil%'))
Покажи имената на всички студенти, подредени по факултетен номер.	student_name ordered faculting_number	cn(username) order(order_by) cn(faculty_number)	cn(username) order(order_by) cn(faculty_number)	SELECT users.username, student.faculty_number FROM users, student WHERE users.id = student.user_id ORDER BY faculty_number

Таблица 10. Част от резултати от Тест 1 към dbUniversity

Тест 1 показа следните проблеми (*П*):

- ***П1*** – при идентифицирано име на колона без да фигурира име на таблица в низа, таблицата не влиза в състава на Select и не построява правилна заявка;
- ***П2*** – има проблем с всички *имена* на: дисциплини, специалности, факултети, образователни степени и т.н.;
- ***П3*** – за езикови фрази като ‚*Компютърни мрежи*‘, намира думите поотделно и ги игнорира, защото не ги открива в модела на ПО;
- ***П4*** – за въпроси от QP1 не поставя символа ‚*‘ – Select * from student;
- ***П5*** – за въпроси от тип – *Брой ..., Колко са...?* не калкулира резултат.

Отстранени бяха идентифицираните проблеми. В ***П1*** се наблюдава непълен шаблон QP2. В случая липсва име на таблица X. Решение е да се добави шаблон QP6, като задачата се сведе до откриване име на таблица X, която е обща за идентифи-цираните във въпроса колони. В ***П2*** се търси *име на.....*, а думата „*име*“ отсъства във въпроса. Решението е да се добави свойство *DefaultColumn* в модела на ПО (най-често за колона *Name* в таблица). Въпроси от вида: Кой / Коя / Кое / Кои ....? – търсят субект, отбелязан като *DefaultColumn*, който се отъждествява с име на таблица. Лингвис-тичният модел бе актуализиран и разширен с редове за phrase = име. При ***П3*** търсената фраза липсва в модела на ПО. След разширяване на модела с липсващият елемент и попълване на неговите свойства, проблема бе отстранен. Проблем ***П4*** е технически и лесно бе премахнат. При ***П5*** в лингвистичният модел е добавен синонимен ред за думата *колко*. По този начин ЕЕИ е в състояние да се справи с потребителски въпроси, съдържащи въпросителни думи: Колко, Брой и др. Добавени са шаблони на въпроси, които са специфични за ИС:

Езикова конструкция	QP6
Кои са / Изведи / Покажи / Намери A1 [A2, A3,…] ?	SELECT A1, A2,.. FROM X
Колко са / Брой / Общ брой [на] A1?	SELECT COUNT(A1) FROM X WHERE P

В **Тест 2** взеха участие 20+20 въпроса към *dbUniversity*. Идентифицирани са следните проблеми:

- ***П1*** – в WHERE има излишен операнд AND;
- ***П2*** – когато намери *value*, а името на колоната, за която е стойност това *value* не присъства в низа, не добавя съответната колона в Select.

П1 е технически и лесно бе премахнат. ***П2*** доведе до разширяване списъка със структурни модели на потребителски въпроси, като добави още един модел **QP8**, условието на който е зададено неявно.

Езикова конструкция	QP8
Изведи / Намери / Покажи A1, A2,.. [от | на] X [за | от] които [е изпълнено] C1=’value_1’ [и | или|,] [C2=‘value_2‘]....	SELECT X.A1, .., Y.C1,… FROM X WHERE X.FK = Y.PKand C1=’value_1’ and C2=’value_2’ …

Тест 3 се проведе с 40+20 потребителски изречения върху *dbUniversity*. Анализът показа 95% успеваемост в синтеза на коректна SQL заявка към БД.

Тест	Брой въпроси	Резултати
1	20	2 грешки (10%) – грешка в обобщаваща *aggr_fun* функция;
		2 грешка (10%) – неразпознат именуван обект *entity* от БД;
		1 грешка (5%) – не разпознава числов формат;
		15 (75%) – успешно формирани заявки.
2	20+20	1 грешка (5%) – неразпознат обект тип value в БД;
		2 грешка (10%) – в синтез на Where клауза, излишен операнд;
		17 (85%) – успешно формирани заявки.
3	40+20	**95% – успешно формирани заявки.**

Таблица 11. Обобщени резултати от проведените тестове към dbUniversity

От резултатите в Експеримент 1 може да се заключи, че софтуерният инструмент ефективно трансформира текст на ЕЕ в структуриран език на заявки към БД на ИС.

Потребителски въпрос	Лематизация и замяна	Тагиране на фразите	Допълнително тагиране	Крайна заявка
Брой студенти втори курс.	number student second course	aggr_fun(count) tn(student) second cn(course)	aggr_fun(count) tn(stu-dent) cn(faculty_number) second cn(course)	SELECT count(faculty_number) FROM student WHERE AND student.course like('%second%')

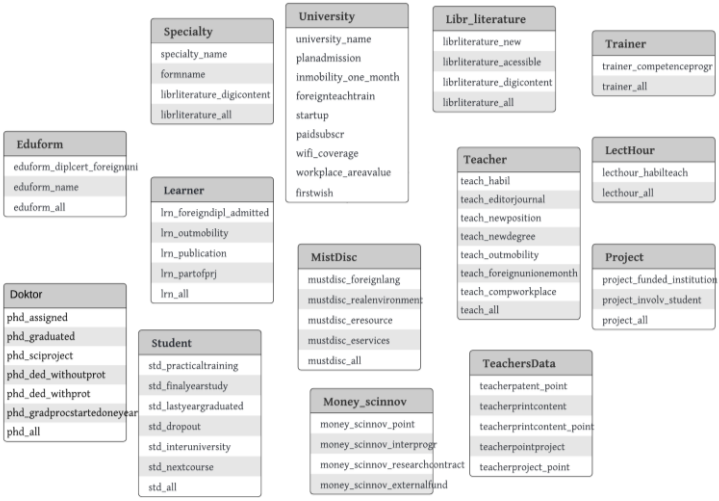
Брой студенти специалност български с немски.	number student bulgarian_german	aggr_fun(count) tn(student) val(bulgarian_german)	aggr_fun(count,cn=faculty_number) tn(student) cn(faculty_number) val(bulgarian_german)	SELECT count(faculty_number) FROM student, specialty WHERE student.spec_id = specialty.id AND specialty.specname like('%bulgarian_german%')
Колко на брой са чуждестранните студенти в специалност Информатика?	how_many foreign student informatics specialty	aggr_fun(count) val(foreign) tn(student) val(informatics) tn(specialty)	aggr_fun(count,cn=isforeign) val(foreign) tn(student) val(informatics) tn(specialty)	SELECT count(isforeign) FROM users, student, specialty WHERE users.id = student.user_id AND student.spec_id = specialty.id AND users.isforeign like('%foreign%') AND specialty.specname like('%informatics%')
Брой на всички студенти от мъжки пол?	number male student	aggr_fun(count) val(male) tn(student)	aggr_fun(count,cn=faculty_number) val(male) tn(student) cn(faculty_number)	SELECT count(faculty_number) FROM users, student WHERE users.id = student.user_id AND users.sex like('%male%')
Преподаватели с докторска степен.	teacher doctor	tn(teacher) val(doctor)	tn(teacher) cn(teachname) val(doctor)	SELECT teacher.teachname, degree.degrname FROM teacher, degree WHERE teacher.sciendegree_id = degree.id AND degree.degrname like('%doctor%')
Преподавател, които преподава Компютърни мрежи?	teacher teach computer_network	tn(teacher) tn(discipline) val(computer_network)	tn(teacher) cn(teachname) tn(discipline) cn(discname) val(computer_network)	SELECT teacher.teachname, discipline.discname FROM teacher, discipline WHERE teacher.discipline_id = discipline.id AND discipline.discname like('%computer_network%')

Таблица 12. Част от тестове към dbUniversity след отстраняване на грешките

4.2. Експеримент 2. ЕЕИ към прототип на БД – НАОА

В този раздел е разгледана примерната постановка по създаване на ЕЕИ, който обработва въпрос на ЕЕ за типични обекти от област – Оценяване качеството на висшето образование (ВО) в България. Базата данни, върху която са насочени разсъжденията и тестовете е ***dbNAOA***. Тя включва информация за обекти (учебни планове, обучавани, проекти, преподаватели и др.) и техни количествени измерения.

В Експеримент 2 моделите, които вече бяха създадени за ПО са актуализирани със специфични за ИС допълнителни езикови единици, синонимни представяния и специфични шаблони на въпроси. Експеримента протича в два етапа. В първият етап се прилага методиката за моделиране на ПО и шаблони на въпроси. Във втория етап отново са проведени три серии от тестове.



Фигура 12. Част от dbNAOA

4.2.1. Моделиране на ЕЕИ

Някои от понятията вече присъстват в **модела на ПО** (тъй като област „Висше образование“ е обща за разглежданите ИС). В резултат от **стъпка *S_Transform*** таблица ***Domain_area*** се актуализира и разширява с липсващите езикови фрагменти.

id	phrase	db_element	property	parent_id	data_type
1	2	3	4	5	6
съществуващи понятия					
1	университет	university	TableN	-	Text
2	факултет	Faculty	TableN	-	Text
3	дисциплина	Discipline	TableN	-	Text
4	задължителна	Discipline.IsOptional	ColumnN	5	Text
5	студент	Student	TableN	-	Text
…	…	…	…	…	…
нови специфични за ИС понятия					
	Общ планов прием	planadmission	ColumnN	1	Number
	Кандидат-студенти по първо желание	firstwish	ColumnN	1	Number
	Входящи мобилности над 1 месец	inmobility_one_month	ColumnN	1	Number
	Стартиращи предприятия	startup	ColumnN	1	Number
	Средна площ на работно място	workplace_areavalue	ColumnN	1	Number
	Покритие безжичен интернет	wifi_cover	ColumnN	1	Number
	Скорост на интернет връзка	wifi_speed	ColumnN	1	Number
	Платени абонаменти за научни БД	paidsubscr	ColumnN	1	Number
	Чуждестранни преподаватели	foreignteacherspartintra	TableN	-	Number
	Проекти	project	TableN	-	Text
	Проекти, финансирани от ВУ	project_funded_instituti	ColumnN	14	Number
	Проекти с участие на студенти,	project_involv_student	ColumnN	14	
	Лекционни часове	lecthour_all	TableN	-	Number
	Обучители	trainer	TableN	-	Number
	Обучители общ брой	trainer_all	ColumnN	18	
	Обучавани	learner	TableN	-	Number
	Обуч, с чуждестранни дипломи	lrn_foreigndipl_admitte	ColumnN	22	Number
	Обуч, в изходящи мобилности	lrn_outmobility	ColumnN	22	Number
	Обуч, с публикации или изяви	lrn_publication	ColumnN	22	Number
	Обуч, участвали в проекти	lrn_partofprj	ColumnN	22	Number
	Студенти	student	TableN	-	Number
	Студ, последна година	std_finalyearstudy	ColumnN	26	Number
	Студ, дипломирали се в последна год.	std_lastyeargraduated	ColumnN	26	Number
	Докторанти	doctor	TableN	-	Number
	Защитили докторанти	phd_graduated	ColumnN	32	Number
	Докторанти в научноизследов. проект	phd_sciproject	ColumnN	32	Number
	Докторанти, отчислени	phd_deducted	ColumnN	32	Number
	Преподаватели	teachers	TableN	-	Number
	Хабилитирани препдаватели	teach_habilitated	ColumnN	40	Number
	Задължителни дисциплини	mustdisc	TableN	-	Number
	Задълж. дисциплини на чужд език	mustdisc_foreignlang	ColumnN	50	Number
	Литература	librliterature	TableN	-	Number
	Новопостъпили заглавия	librliterature_new	ColumnN	55	Number
…	...	…	…	…	…

Таблица 13. Списък с понятия в Модела на ПО за dbNAOA

В резултат от **стъпка *S_BulNet*** таблица ***Lexicon*** – **лингвистичният модел** за *dbNAOA* изглежда по следния начин:

id	phrase	synsets	pos
1	2	3	4
съществуващи понятия			
1	университет	ВУЗ, институция, институт, висше училище	N
2	факултет	звено, отдел	N
3	дисциплина	предмет	N
4	задължителна	по учебен план	Adj
5	студент	учащ, обучаем, обучаван	N
...	...	...	...

специфични за ИС понятия			
4	член	участник, в състава на, е включен в	N
5	проект	идея, предложение	N
8	регионален	местен, локален, градски, град, област	Adj
9	национален	държавен, български	Adj
10	международен	чуждестранен, чужд	Adj
11	публикация	статия, труд, творба, работа	N
12	издание	база данни, тираж, списание, книга	N
13	редакционна колегия	колектив, екип, състав, група	N
16	рецензия	мнение, отзив, преценка, критика, позиция, преглед	N
22	авторски	уникален, собствен	Adj
29	нереферирaн	неразрешен, съмнителен, несигурен	Adj
30	публикуван	издаден, отпечатан	N
37	оценка	оценка, проверка	N
38	съвместно	заедно, общ,	Adj
39	продължаващо	последващо, непосредствено след	Adj

Таблица 14. Част от понятията в Лингвистичния модел за dbNAOA

На **стъпка *S_InfoQA*** моделите на въпроси са допълнени със специфични езикови конструкции на въпроси за конкретната ИС. На база структурни модели на въпроси, част от които изискват изчисления, бяха категоризирани няколко базови типа заявки.

Езикова конструкция	SQL заявка	QP
Общ |Брой | Колко са [всички] S?	SELECT * FROM S	Q1
Общ | Брой | Колко са [всички] S, [за| които |е изпълнено] P?	SELECT * FROM S WHERE P	Q2
Относителен дял [в проценти] на X, спрямо S, за които P?	SELECT (X / S * 100) FROM S WHERE P	Q3
Какъв е процента на X, спрямо S, за които P?		
Процентно отношение на X, спрямо S, за които P?		
Средна стойност [брой] на A [на елементи] в S?	SELECT SUM(A) / COUNT(S) FROM S WHERE P	Q4
Средноаритметичен [брой] на A за S?		

Таблица 15. Езикови конструкци и условия за приложимост на заявки за dbNAOA

Граматичните структури на индикаторите са **категоризирани в шаблони**, според метода на изчисление и получаване на числената им стойност. В методиката за определяне шаблоните на въпроси стойността на базовите понятия е определяща за изчисляване на количествени измерения, като напр. *отношение, относителен дял, процент* и др.. Остойностяването на количествени индикатори от даден шаблон следва логиката, характерна за съответния тип.

4.2.2. Тестове

Проведени са **три теста** с въпроси, зададени на български език към *dbNAOA*, състояща се от количествени индикатори, по утвърдени стандарти, оценяващи качеството на висшите училища в България.

Първоначално бе проведен **Тест 1** с 20 въпроса. В **Тест 2** към първите 20 въпроса бяха добавени нови 20 (20+20). **Тест 3** включва последните 40, като бяха допълнени нови 20 въпроса (40+20). Първоначалните въпроси се провериха, тъй като имаше нужда от потвърждаване на получените резултати.

Анализа на резултатите в **Тест 1** откри **следните проблеми**:

- ***П1*** – при открито име на таблица, без да е спомената колона от нея, игнорира таблицата. Образува JOIN с нея, но не я добавя в SELECT клаузата. Върнатите резултати не отговарят точно на запитването;
- ***П2*** – при търсене на обучаеми – извежда само студенти, не включва докторанти в търсенето, а те също са обучаеми;
- ***П3*** – не бяха обхванати общи стойности на абстрактни понятия, като – общ брой студенти, общ брой докторанти, общ брой преподаватели и т.н.

След обобщение и анализ на забелязаните проблеми, бяха изведени конкретни изводи за това как всеки от тях да бъде отстранен.

Отстранени бяха идентифицираните технически и логически проблеми. На база на откритите проблеми са добавени нови термини в лингвистичния модел, шаблони на въпроси и нова функционалност. За ***П1*** когато във въпроса е открита таблица, без колона/и/ в нея, нищо от тази таблица не се визуализираше в Select. Решение на проблема е добавяне на шаблон на този тип потребителски въпрос и създаване на функция за извличане на неявната във въпроса информация за попълване параметрите на заявката. Взето бе решение в списъка с понятия за ПО, всяко абстрактно понятие (име на таблица) да се аташира и със свойство DefaultColumn, за да може когато в запитването липсва атрибут, а е налична само таблица да извежда нейната default колона.

За решение на проблем ***П2*** бе добавена фразата *обучаем* в лингвистичния модел на ПО, която заменя лексемите *студент и докторант* и обобщава резултатите за тези обекти в заявките. Друг проблем ***П3*** се явяваха обобщени данни за абстрактните понятия, които са действителни имена на таблици. В тях не беше предвидено поле за общите стойности на тези понятия и нямаше как да се отговори на въпрос от вида – общ брой преподаватели, общ брой дисциплини и др. В тази връзка бе добавен нов шаблон, съответен на конструкция от вида: Select count(id) from X.

Езикова конструкция	**Q6**
Кои са / Намери / Покажи X и C1, C2,.. от Y [за | от] които [е изпълнено] C1='value_1' [и | или|,] [C2='value_2']	SELECT X.A1, X.A2,…., Y.C1,… FROM X WHERE X.FK = Y.PK and C1='value_1' ….
Колко са / Брой / Общ брой на X ?	SELECT COUNT(id) FROM X

След отстраняване на проблемите бе проведен **Тест 2**, проведен с нови 20 потребителски въпроси към *dbNAOA* (вж. табл. 27.). В нея участваха общо 20+20 изречения. В теста участваха и първите 20 изречения, за да се потвърдят получените резултати. **Тест 2** показа няколко проблема:

- ***П1*** – за търсене на хабилитиран преподавател да разбира доцент или професор;
- ***П2*** – в WHERE има излишен операнд AND, независимо от броя условия;
- ***П3*** – наблюдава се объркване при наличие на логически квантори ИЛИ и И.

Потребителски въпрос	**Лематизация и замяна**	**Тагиране на фразите**	**Допълнително тагиране**	**Крайна заявка**
Относителен дял кандидат-студенти приети на 1-во желание спрямо общия планов прием.	ratio first_wish planadmission	fun(ratio) cn(firstwish) cn(planadmission)	fun(ratio) cn(firstwish) cn(planadmission)	SELECT firstwish/planadmission FROM university
Процент обучавани, приети по чуждестранни дипломи, спрямо общия планов прием на ПУ.	ratio lrn_foreigndipladmitted planadmission plovdiv_university	fun(ratio) cn(lrn_foreigndipladmitted) cn(planadmission) val(plovdiv_university)	fun(ratio) cn(lrn_foreigndipladmitted) cn(planadmission) val(plovdiv_university)	SELECT lrn_foreigndipladmitted/planadmission FROM learner, university WHERE university.id = learner.university_id AND university.university_name like('%plovdiv_university%')
Брой докторанти, защитили докторска степен в УХТ.	phd_graduated uni_food_technology	cn(phd_graduated) val(uni_food_technology)	cn(phd_graduated) val(uni_food_technology)	SELECT doctor.phd_all, university.university_name, doctor.phd_graduated FROM doctor, university WHERE university.id = doctor.university_id AND university.university_name like('%uni_food_technology%')
Колко общо са лекционните часове за специалност Информатика на ПУ.	lecthour plovdiv_university informatics	tn(lecthour) val(plovdiv_university) val(informatics)	tn(lecthour) cn(lecthour_all) val(plovdiv_university) val(informatics)	SELECT lecthour.lecthour_all, university.university_name, specialty.specname FROM lecthour, university, specialty WHERE lecthour.specialty_id = specialty.id AND specialty.university_id = university.id AND univer-

				sity.university_name like('%plovdiv_university%') AND specialty.specname like('%informatics%')
Средна площ (предоставена) на преподавател, за осигуряване на неговата работа в ПУ	workplace_areavalue plovdiv_university	cn(workplace_areavalue) val(plovdiv_university)	cn(workplace_areavalue) val(plovdiv_university)	SELECT university.university_name, university.workplace_areavalue FROM university WHERE AND university.university_name like('%plovdiv_university%')
Брой платени абонаменти за научни бази данни.	paidsubscr	cn(paidsubscr	cn(paidsubscr)	SELECT university.university_name, university.paidsubscr FROM university

Таблица 16. Част от въпросите в Тест 2 на Експеримент 2

За ***П1*** към модела на ПО е добавена фраза *хабилитиран*, която в синонимния си ред се заменя с думите *доцент* и *професор*. Така заявката извлича информация едновременно за двата субекта. Проблем ***П2*** е технически и много лесно е отстранен. Проблем ***П3*** предстои да бъде решен в близка перспектива.

Тест	Брой въпроси	Резултати
1	20	2 грешки (10%) – семантични грешки с абстрактни понятия;
		1 грешка (5%) – разменен словоред не разпознава обекти от БД в дълги фрази;
		1 грешка (5%) – не разпознава числов формат;
		16 (80%) – успешно формирани заявки.
2	20+20	2 грешки (10%) – при формиране на функция за отношение
		18 (90%) – успешно формирани заявки.
3	40+20	**95% – успешно формирани заявки.**

Таблица 17. Обобщени резултати от проведените тестове към dbNAOA

От експериментите, направени с потребителски въпроси към *dbNAOA* може да се направи извод, че софтуерният инструмент се справя със задачата да трансформира естественоезиков текст в заявка към БД.

4.3. Изводи

Експерименталните ЕЕИ показват ефективността на софтуерното решение, неговите предимства, гъвкава функционалност и представяне на резултатите в две направления от разглежданата област. В тях софтуерният модул отговаря на въпроси, свързани с академичните постижения на висшите училища и извлича достоверни, адекватни и релевантни резултати.

Заключение

В рамките на дисертационното изследване **са решени следните основни задачи**:

Задача 1. Проучване на съществуващи теоретични подходи, модели и системи, свързани с ЕЕИ.

Задача 2. Създаване на общ модел, архитектура и прототип на ЕЕИ към информационна система, както и на методика за тяхното прилагане.

Задача 3. Създаване на софтуерен инструмент за формиране на заявка към БД от потребителски текст на естествен език.

Задача 4. Проектиране, реализация и тестване на софтуерния инструмент с ЕЕИ за различни информационни системи (в област „Висше образование“), базиран на предложеният модел и следвайки конкретни методики.

С решаването на задачи 1. – 4. се постига основната цел на дисертационното изследване – предлагане, изследване и апробиране на средства (модели, методи и инструменти), подходящи за автоматизиране на процеса по създаване на ЕЕИ към ИС в

областта на ВО.

Приноси на дисертационния труд

Основните приноси на дисертационния труд могат да се характеризират като научни, научно-приложни и приложни.

Научни приноси на дисертационното изследване са:

- **Н1.** Предложена е методика за създаване на ЕЕИ към информационна система;

Научно-приложни приноси на дисертационния труд са:

- **НП1.** Разработена е архитектура на софтуерен инструмент с ЕЕИ към ИС;
- **НП2.** Реализиран е софтуерен инструмент с ЕЕИ към ИС в област ВО с различно предназначение и обхват.

Приложни приноси на дисертационното изследване са:

- **П1.** Създадени и тествани са компютърни модели на ПО по методиката от Н1;
- **П2.** Създадени са експериментални ЕЕИ за две ИС в област ВО, в които са проведени тестове с потребителски въпроси.

Приноси		**Задачи**	**Глава**
Научни	**Н1**	2	2
Научно-приложни	**НП1**	2	2
	НП2	3	3
Приложни	**П1**	4	4
	П2	4	4

Таблица 18. Приноси и задачи, разпределени по раздели

Перспективи за бъдещо развитие

Резултатите на дисертационното изследване от Глава 2. могат да бъдат мултиплицирани за създаване на ЕЕИ в други информационни системи в различни предметни области, поради своята общност.

Перспективите за развитие може да се разглеждат в по-близък и в по-далечен план. В **близка перспектива** е предвидено:

- развитие и обогатяване модела на ПО;
- усъвършенстване на лингвистичния модел;
- подобряване преносимостта на информация от БД и автоматизиране на съответствия между понятия и обекти в БД;
- разширяване на приложението до диалогова система;
- връзка с други съществуващи компоненти и източници на информация;
- приложение на ЕЕИ в други информационни системи.

Като по-далечна перспектива е запланувано:

- приложение на ЕЕИ за други предметни области;
- възможност за интерактивна (графична, таблична) визуализация на отговора;
- разширяване функционалността на софтуерния инструмент за генериране на INSERT заявка от параметри, идентифицирани от текст на документ в свободна форма, използвайки предложените модели на ПО.

Апробация

Основните резултати на изследването са докладвани на национални и международни научни форуми. Резултатите от изследването са представени в 6 (шест) публикации – 3 (три) в специализирани списания от международни конференции и 3 (три) в сборници с трудове от български конференции. Две от публикациите са индексирани в международни бази от данни Scopus, и двете са публикувани в издания с SJR.

Публикации по темата на дисертационния труд

1. **Жекова М.** Преглед и проучване на съществуващи системи за изкуствен интелект, Годишник на Националната научна конференция за студенти и докторанти, Пловдив 2019, Plovdiv University Press, ISSN 2682-9460, Verba iuvenium, Issue 2, p. 205 – 218.
2. **Жекова М.**, Тотков Г., Метод за генериране на SQL заявки към база данни, зададени с текст на естествен език, Научни трудове на Съюза на учените в България – Пловдив. Серия В. Техника и технологии. Том XVIII, Пловдив 2019, ISSN 1311-9419 (Print), ISSN 2534-9384(Online), p. 98 – 101.
3. **Жекова М.**, Тотков Г., Методология за създаване на система от шаблони за съответствие във въпрос-отговор система с естественоезиков интерфейс, Сборник доклади на Национален младежки форум пролет „Наука, технологии, иновации, бизнес", Пловдив 2019, ISSN 2367-8569, p. 139 – 145.
4. **Zhekova M.**, Totkov G., Model of process and model of natural language processing system, IOP Conference Series: Materials Science and Engineering, Volume 878, Issue 1, 2020, ISSN: 1757-8981, DOI: 10.1088/1757-899X/878/1/012028.
5. **Zhekova M.**, Test and analysis of experiments for the applicability of a semantic analyzer for natural language interface to the database, Knowledge for research and practice International Journal Scientific Papers Vol. 39.1, 2020, ISSN: 2545-4439, p. 49 – 55.
6. **Mariya Zhekova**, George Totkov, Question patterns for natural language translation in SQL queries, International Journal on Information Technologies and Security, Vol. 13, Issue 2, 2021, ISSN 1313-8251, p. 43 – 54.

Библиография

[Жекова'2019B] Жекова М., Тотков Г., Методология за създаване на система от шаблони за съответствие във въпрос-отговор система с естественоезиков интерфейс, Сборник на Национален младежки форум, ISSN 2367-8569, стр. 139-145

[Тотков'2001] Тотков Г., Вл. Шкуртов, Р. Донева, Основи на компютърната информатика, Университетско издателство, ПУ, Пловдив, стр. 35-40, 2001

[Тотков'2014] Тотков Г., Пловдивски е-университет, „Ракурси", Пловдив, 2014

[Androutsopoulos'1995] Androutsopoulos I., Ritchie G.D. and Thanisch P., Natural Language Interfaces to Databases – An Introduction, Natural Language Engineering. 1(1): 29-81, Cambridge University Press, 1995

[Batra'2004] Batra D., Wishart N. A. Comparing a rule-based approach with a pattern-based approach at different levels of complexity of conceptual data modelling tasks. Jour. of Human-computer Studies, International Journal of Man-machine Studies, Vol 61 (4), p. 397-419

[Baxter'2005] Baxter D., Shepard B., Siegel N., Gottesman B., Schneider D., Interactive Natural Language Explanations of Cyc Inferences, Explanation-Aware Computing, Papers from AAAI Fall Symposium, Virginia, November 2005

[Bjorner'2019] Bjorner D., Domain Analysis and Description Principles, Techniques, and Modelling Languages. ACM Trans. Softw. Eng. Methodol. 28, 2, Article 8, 67 pages

[Blunschi'2012] Blunschi, L., Jossen, C., Kossmann, D., Mori, M., & Stockinger, K., SODA: Generating SQL for Business Users. PVLDB, 5, p. 932-943, 2012

[Bouma'2007] Bouma G., Kloosterman G., Mur J., Noord G., Plas L., Tiedemann J., Question Answering with Joost at CLEF 2007, Advances in Multilingual and Multimodal Information Retrieval, Springer-Verlag, p. 257-260

[Everling'2015] Everling N.,Investigating precise implementing a portable natural language interface to databases, Stocholm, Sweden 2015

[Gonzalez'2015] Gonzalez J., Florencia R., Fraire H., Pazos R., Cruz-Reyes L., Gómez C., Semantic representations for knowledge modelling of a Natural Language Interface to Databases using ontologies, International J. of Combinatorial Optim. Problems and Informatics vol.6 no.2, 2015

[Gonzalez'2016] Gonzalez D., Popovich A., Mirakhorli M. (2016). TestEX: A Search Tool for

Finding and Retrieving Example Unit Tests from Open Source Projects. 2016 ISSREW, p. 153-159, 2016

[Imminni'2016] Imminni S. K. et al., SPYSE - A Semantic Search Engine for Python Packages and Modules, 2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C), p. 625-628, 2016

[Handschuh'2003] Handschuh, S., Staab, S., Annotation for the Semantic Web, Frontiers in Artificial Intelligence and Applications, vol. 96. IOS Press, ISBN print 978-1-58603-345-3, 2003

[Hendrix'1978] Hendrix G., Sacerdoti E., Sagalowicz D., Slocum J., Developing a natural language interface to complex data, ACM Transactions on database systems, p. 105-147, NewYork USA: SRI International, june 1978

[Kate'2005] Kate R. J., Wong Y. W., Mooney R. J., Learning to transform natural to formal languages. In Proc. of AAAI-05, p. 1062-1068, Pittsburgh, PA, July 2005

[Kim'1993] Kim J., Moldovan D., PALKA: A System for Lexical Knowledge Acquisition, CIKM '93: Proceedings of the second international conference on Information and knowledge management, p. 124-131, December 1993

[Khapane'2015] Khapane A., Kapadane M., Patil P., Siraj S., Natural language database interface, International Journal of Engineering Research and General Science Vol 3, Issue 2, Part 2, ISSN 2091-2730, 2015

[Lenat'1985] Lenat D., Prakash M., Shepherd M., CYC: Using Common Sense Knowledge to Overcome Brittleness and Knowledge Acquisition Bottlenecks, Microelectronics ~3 Computer Technology Corporation, p. 65-85, 1985

[Lin'2017] J. Lin et al., TiQi: A natural language interface for querying software project data, 32nd International Conference on Automated Software Engineering (ASE), 2017, p. 973-977

[Miller'1995] Miller G., WordNet: A Lexical Database for English, Communications of the ACM Vol. 38, No. 11: p. 39-41

[OVIS] https://www.let.rug.nl/~vannoord/papers/oviseval/node7.html

[Pinheiro'2010] Pinheiro V., Furtado V., Pequeno T. and Nogueira D., Natural Language Processing based on Semantic inferentialism for extracting crime information from text, International Conference on Intelligence and Security Informatics, 2010, p. 19-24

[Popescu'2004] Popescu A., Armanasu A., Etzioni O., Ko D., and Yates A., Modern Natural Language Interfaces to Databases: Composing Statistical Parsing with Semantic Tractability, COLING 2004, p. 141-147, Geneva, Switzerland 2004

[Rayson'2004] Rayson P., Archer D., Piao SL. and McEnery T., The UCREL semantic analysis system, (LREC 2004), p. 7-12, Lisbon, Portugal 2004

[Reynolds'2016] Reynolds R., Russian natural language processing for computerassisted language learning, The Arctic University of Norway Norway February 4, 2016

[Simitsis'2008] Simitsis A., Koutrika G., Ioannidis Y., Precis: From unstructured keywords as queries to structured databases as answers, The VLDB Journal 17(1):117-149, January 2008

[Vallez'2007] Vallez M., Pedraza-Jimenez R., Natural Language Processing in Textual Information Retrieval and Related Topics, "Hipertext.net", num. 5, 2007

[Warren'1982] Warren, D., Pereira, F., An efficient easily adaptable system for interpreting natural language queries. Computational Linguistics, 8(3-4): p. 110-122, 1982

[Wilkens'2010] Wilkens R.,Villavicencio A., Muller D., Wives L., Silva F., Loh S., COMUNICA - A question answering system for brazilian portuguese, Coling 2010, p. 21-24

[Zhekova'2021C] Mariya Zhekova, George Totkov, Question patterns for natural language translation in SQL queries, International Journal on Information Technologies and Security, Vol. 13, Issue 2, ISSN 1313-8251, p. 43-54, 2021

Бележки