**Universitatea „Alexandru Ioan Cuza” Iaşi**

**Facultatea de Informatică**



LUCRARE DE LICENŢĂ

**SMART QUIZZ**

**Question Answering System**

**Mădălina-Georgiana Chelariu**

**Sesiunea: Iulie, 2015**

**Coordonator ştiinţific**

**Lector, Dr. Ionuţ** **Pistol**

**UNIVERSITATEA „ALEXANDRU IOAN CUZA” IAŞI**

**FACULTATEA DE INFORMATICĂ**

**SMART QUIZZ**

**Question Answering System**

**Mădălina-Georgiana Chelariu**

**Sesiunea: Iulie, 2015**

Coordonator științific

**Lector, Dr. Ionuț**  **Pistol**

**Declarație privind originalitate și respectarea drepturilor de autor**

Prin prezenta declar că Lucrarea de licență cu titlul „SMART QUIZZ – Question Answering System” este scrisă de mine și nu a mai fost prezentată niciodată la altă facultate sau instituție de învățământ superior din țară sau străinătate. De asemenea, declar că toate sursele utilizate, inclusiv cele preluate de pe Internet, sau indicate în lucrare, cu respectarea regulilor de evitare a plagiatului:

* toate fragmentele de text reproduse exact, chiar și în traducere proprie din altă limbă, sunt scrise între ghilimele și dețin referința precisă a sursei;
* reformularea în cuvinte proprii a textelor scrise de către alți autori deține referința precisă;
* codul sursă, imagini etc. preluate din proiecte open-source sau alte surse sunt utilizate cu respectarea drepturilor de autor și dețin referințe precise;
* rezumarea ideilor altor autori precizează referința precisă la textul original.

**Iași, Absolvent Mădălina-Georgiana Chelariu \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Declarație de consimțământ**

Prin prezenta declar că Lucrarea de licență cu titlul „SMART QUIZZ – Question Answering System”, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test etc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea Alexandru Ioan Cuza Iași să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

**Iași, Absolvent Mădălina-Georgiana Chelariu**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Acord privind proprietatea dreptului de autor**

Facultatea de Informatică este de acord ca drepturile de autor asupra programele-calculator, format executabil și sursă, să aparțină autorului prezentei lucrări, Mădălina-Georgiana Chelariu. Încheierea acestui acord este necesară din următoarele motive:

**Iași,**

**Decan: Dr. Adrian Iftene Absolvent Mădălina-Georgiana Chelariu**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Cuprins**

Table of Contents

[Motivație 1](#_Toc422602160)

[1. Introducere 2](#_Toc422602161)

[1.1. Glosar de termeni 5](#_Toc422602162)

[1.2. Realizări din domeniu la nivel național 8](#_Toc422602163)

[2. Aplicații din domeniu 9](#_Toc422602164)

[2.1. Metodologia unui sistem întrebare-răspuns 9](#_Toc422602165)

[2.2 Arhitectura unui sistem întrebare-răspuns 10](#_Toc422602166)

[2.2.1 Analiza întrebării 10](#_Toc422602167)

[2.2.2 Căutarea documentară 10](#_Toc422602168)

[2.2.3 Extragerea răspunsului 10](#_Toc422602169)

[2.3 Prima perioada : jocurile și demonstrarea de teoreme 11](#_Toc422602170)

[2.4 A doua perioadă - înțelegerea limbajului natural 12](#_Toc422602171)

[2.4.1 ELIZA 12](#_Toc422602172)

[2.4.2 AML și A.L.I.C.E. 14](#_Toc422602173)

[2.4.3 SHRDLU 16](#_Toc422602174)

[2.4.4 START 16](#_Toc422602175)

[2.5 A treia perioadă - sisteme expert evoluate 18](#_Toc422602176)

[2.5.1 Watson 19](#_Toc422602177)

[3. Tehnologii utilizate 21](#_Toc422602178)

[3.1. Lematizare şi POS-tagger 21](#_Toc422602179)

[3.2 Lucene 23](#_Toc422602180)

[3.4. ASP.NET MVC 26](#_Toc422602181)

[3.5. JavaScript și CSS 27](#_Toc422602182)

[3.5.1. Bootstrap 28](#_Toc422602183)

[3.6. Cloud computing 29](#_Toc422602184)

[3.6.1. Platforma Windows Azure 31](#_Toc422602185)

[4. Detalii de implementare 33](#_Toc422602186)

[4.1. Model 33](#_Toc422602187)

[4.2. VIEW 39](#_Toc422602188)

# Motivație

Această lucrare descrie scopul, metodele și implementarea unei aplicații care are ca funcționalitate principală rezolvarea automată de teste tip grilă.

Necesitatea unui astfel de produs software provine din nevoia utilizatorilor de a găsi răspunsul la întrebări nu doar într-un mod cât mai rapid ci și suficient de succint.

Un utilizator are, prin internet, acces la o cantitate de informații extrem de vastă. La ora actuală, Internetul poate fi considerat cea mai mare bază de cunoștințe, aflată într-o continuă extindere și reîmprospătare. Acesta este motivul pentru care procesul de căutare a informației este întotdeauna unul dificil.

Aplicația Smart QuizZ, își propune să vină în sprijinul utilizatorilor care doresc să găsească rapid răspunsul la o întrebare adresata în limbaj natural, nefiind nevoiți sa parcurgă o lista de pagini web sau de documente pentru a identifica răspunsul cel mai relevant.

# 1. Introducere

În ultimii ani, un interes ridicat s-a manifestat pentru lingvistica computațională. Comunitatea lingvisticii computaționale ține pasul cu nevoile utilizatorului și încearcă să ofere sisteme informatice menite să ajute la extragerea și prelucrarea informației.

În prezent, cea mai eficientă și utilizată metoda de achiziție de informație este reprezentată de motoarele de căutare. Scopul acestora este de a oferi utilizatorului un set de pagini web sau articole în care acesta poate regăsi informația care îi este necesară.

Pentru informare, utilizatorul recurge frecvent la formularea unei întrebări. Pornind de la aceasta, diverse aplicații oferă documente care, se presupune, includ răspunsul la întrebare sau chiar fragmentele considerate adecvate din documentele estimate a fi cele mai relevante.

Pasul următor în domeniul achiziției informației este constituit de dezvoltarea sistemelor capabile să răspundă la întrebări formulate de utilizator în limbaj natural. Sistemele întrebare-răspuns (Question Answering) pe domenii deschise, reprezintă o cercetare de top și totodată o temă de dezvoltare în cadrul tehnologiilor lingvistice actuale. Spre deosebire de motoarele de căutare standard, care se bazează pe metode tradiționale de achiziții de informații, de la sistemele QA [[1]](#footnote-1) pe domenii deschise, se așteaptă, nu livrarea unei liste de documente care pot fi relevantă pentru cererea utilizatorului, dar a unei propoziții ori alineat care ar răspunde la întrebarea formulată în limbaj natural[[2]](#footnote-2).

Smart QuizZ este un sistem de tip întrebare-răspuns care primește ca input o întrebare în limbaj natural și, în urma procesării atât a întrebării cât și a documentelor sau informațiilor de background pe care utilizatorul trebuie sa le introducă, este capabil să răspundă la întrebare.

Dacă utilizatorul aplicației nu introduce și baza de cunoștințe necesara pentru aflarea răspunsului, sistemul va căuta răspunsul in propria baza de cunoștințe. Răspunsul oferit de sistem trebuie să îndeplinească următoarele condiții: să fie corect, să fie formulat tot în limbaj natural uman și să fie suficient de succint.

Un sistem de răspuns la întrebări necesită o procesare a limbajului natural mult mai complexă decât sistemele de achiziție de documente. Așa numitele documente de background, reprezintă baza de cunoștințe pentru un astfel de sistem.

Un astfel de sistem poate avea un impact major și asupra evaluării în învățământ, subiect foarte controversat în ultimii ani. Evaluarea este un proces complex care presupune un efort continuu de diminuare a subiectivismului din aprecierile profesorilor în acordarea de “note” (calificative, punctaje etc.)

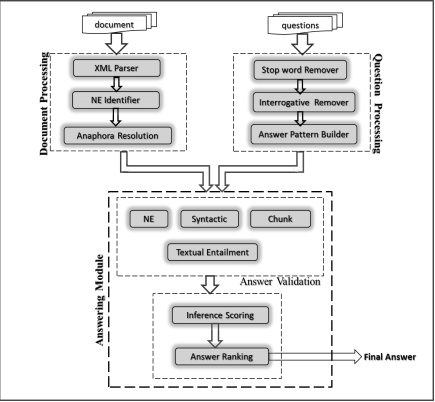
Problema, în folosirea acestui tip de testare, consta în timpul mare necesar pentru stabilirea rezultatelor și posibilitatea apariției unor greșeli datorate factorului uman. Automatizarea procesului de stabilire a rezultatelor apare ca un obiectiv firesc, cu consecință imediată a creșterii fidelității rezultatelor obținute si minimizării timpului consumat.

În cazul testelor evaluatorul nu are altceva de făcut decât să constate alegerea făcută de evaluat dintr-o mulțime de posibilități și să acorde punctele corespunzătoare opțiunii. Erorile care pot apărea țin de factorul uman, dar sunt de altă natură. În funcție de modul concret de lucru, pot apărea erori la transcrierea rezultatelor, la punerea în corespondenta a opțiunilor cu punctajele sau la calculul punctajului total al fiecărui test.

O soluție folosita pe larg în trecut a constat în introducerea redundantei: toate operațiunile erau efectuate de doua ori, de persoane diferite. Ca efect, a crescut fiabilitatea întregului proces dar si timpul consumat si stresul asupra personalului. Fiind vorba despre activități de rutina, care nu implica inteligentă, ele pot fi automatizate, prin folosirea tehnicii de calcul.

În ajutorul acestui proces de stabilire a rezultatelor corecte pentru un astfel de test grilă, vine aplicația Smart QuizZ, care are ca obiectiv principal procesarea atât a întrebărilor din test, cât și a colecției de documente suport oferite de utilizator pentru găsirea răspunsului corect din cele cinci variante de răspuns oferite.

Proiectul de fată propune realizarea unui sistem avansat, parametrizabil și interlingual de întrebare în limba română și răspuns în limba română sau engleza relativ la o colecție dinamică de documente conținând un număr arbitrar de mare de texte (la limita, întregul web[[3]](#footnote-3)). Sistemul va include în arhitectura sa componentele specifice unui sistem deschis de întrebare-răspuns dar va fi capabil să își rafineze precizia și completitudinea răspunsurilor pe măsură ce domeniul de discurs este mai bine precizat.



Figură 1 Arhitectura unui sistem QA ce a participat la CLEF 2013

Sistemul [[4]](#footnote-4)al cărei arhitecturi este prezentată în figura 1, a participat la competiția CLEF din anul 2013 și a avut ca input 9 teste, fiecare test conținând cate 5 întrebări cu câte 4 variante de răspuns. Sistemul trebuia să răspundă la întrebări în limba engleză, alegând unul din cele 4 răspunsuri propuse. În cazul în care sistemul nu este confident pe răspuns, poate să nu răspundă la întrebare.

Lucrarea este structurată pe patru capitole pornind de la un overwiew asupra aplicațiilor reprezentative din domeniu construite până în acest moment, trecând în revistă tot odată și perioadele inteligentei artificiale, detaliind în capitolul următor tehnologiile utilizate care fac posibilă implementarea acestui astfel de sistem. Un capitol foarte important al lucrării este capitolul patru aduce în prim plan arhitectura și implementarea produsului software, acestea fiind în detaliu studiate, urmând ca finalul acestei lucrări să fie reprezentat de concluziile cercetării și de direcțiile în care poate evolua aplicația

## Glosar de termeni

*Inteligenţa artificială* [[5]](#footnote-5) (IA), pe de altă parte, este un domeniu tehnico-stiinţific în care se încearcă realizarea de mașini (roboți) care să poată substitui oamenii. În plus, similar mașinilor care nu numai că înlocuiesc forța fizică umană, ci fac lucruri pe care oamenii nu le pot face, se așteaptă că și roboții vor gândi cu mult mai eficient ca oamenii.

*QA[[6]](#footnote-6) (Question Answering)* Sistemele de tip întrebare-răspuns reprezintă noua generație de unelte pentru extragerea de date din colecții mari de documente.

*QA4MRE[[7]](#footnote-7)*  (în engleză *Question Answering for Machine Reading Evaluation* ) a fost propus ca laborator pentru CLEF 2011.Sistemele construite trebuie să poată utiliza cunoștințele obținute automat din textele date ca suport pentru a răspunde la un set de întrebări. Întrebările au variante multiple de răspuns, doar unul fiind corect. Dacă documentul oferit ca suport nu este suficient, sistemul va folosi și o “bază de cunoștințe alternativa” pentru ajutor în găsirea răspunsului.

*XML[[8]](#footnote-8) (Extensible Markup Language)* este un meta-limbaj de marcare similar cu HTML, o diferență esențială fiind reprezintă de tagurile nu sunt definite. Documentele XML sunt realizate din unități de stocare numite entități, ce conțin date parsate sau neparsate. Datele parsate sunt realizate din caractere, unele dintre ele formând date caracter iar altele ca marcaje. Marcajele codifică o descriere a schemei de stocare a documentului și structura logică. XML furnizează un mecanism pentru a impune constrângeri asupra schemei de stocare și a structurii logice.

*NLP[[9]](#footnote-9)(Natural Language Processing)* este capacitatea unui calculator de a "înțelege" limbajul text natural la un nivel ce permite să aibă sens interacțiunea dintre un calculator și o persoană ce lucrează într-un domeniu particular.

*CLEF[[10]](#footnote-10) (Conference and Labs of the Evaluation Forum)* este o organizație care promovează cercetarea pe accesul de informații, în prezent concentrându-se pe limbile europene. Funcțiile sale specifice sunt de a menține un cadru pentru testarea sistemelor de regăsire de informații dar și de a crea arhive de date pentru cercetătorii care doresc să le utilizeze în dezvoltarea de noi sisteme. Organizația tine o reuniune în fiecare an în luna septembrie. Înainte de fiecare forum, participanții primesc un set de sarcini. Sarcinile sunt concepute pentru a testa diferitele aspecte ale sistemelor de regăsire de informații și să încurajeze dezvoltarea acestora. Grupuri de cercetători propun și organizează campanii pentru a îndeplini aceste sarcini.

*TREC [[11]](#footnote-11)(Text Retrieval Conference)* Scopul acestuia a fost pentru a sprijini cercetarea în comunitatea de regăsire de informații prin asigurarea infrastructurii necesare pentru evaluare la scară largă a metodologiilor de regăsire a textul.

*Servicii de tip SOAP [[12]](#footnote-12)(Simple Object Access Protocol)* sunt servicii în care cererile și răspunsurile au forma unor documente XML transmise peste HTTP. Astfel de servicii expun și o descriere a interfeței API sub forma unui document WSDL, care este tot XML și poate fi prelucrat de client. Un client trebuie să cunoască metodele oferite de către server furnizor de servicii6, pe care le poate afla din descrierea WSDL.

RACAI[[13]](#footnote-13) - Romanian Academy Research Institute for Artificial Intelligence “Mihai Drăgănescu” (RACAI) a fost înființat în 1994 ca un centru de competentă și al diseminării active a cunoștințelor în domeniul inteligenței artificiale.

## Realizări din domeniu la nivel național

Romania participa pentru prima data la competiția CLEF din 2006 cu un sistem in limba romana doi dintre participanți fiind membri ai ICIA si UAIC. Cel mai bun rezultat pentru limba romană, obținut de unul din membrii a fost de 0,1316, modest în comparație cu nivelul mondial atins pentru limba engleză, dar competitiv în raport cu alte limbi nou venite. La competiția CLEF 2007, limba romană a fost din nou prezentă în competiție, ICIA si UAIC s-au înscris în concurs. Rezultatele, arătau o îmbunătățire substanțiala fata de rezultatele din anul trecut (scorul ~ 0,28).

În septembrie 2007 o echipa a Facultății de Informatica (UAIC) , sub atenta coordonare a academiei române de cercetare în domeniul inteligenței artificiale (RACAI) dezvoltă sistemul SIR-RESDEC [[14]](#footnote-14)(Open Domain Question Answering System for Romanian and English) care

s-a desfășurat pe durata a 3 ani. Obiectivul general al sistemului propunea realizarea unui sistem de întrebare răspuns monolingv (română) cât și cros-lingual (întrebare în limba română, răspunsul extras din documente elaborate în limba engleză) în spații de căutare deschise sau circumscrise, sistem parametrizabil atât in raport cu limba cât și cu domeniul de discurs.

Un alt exemplu este FALCON - dezvoltat la UTD[[15]](#footnote-15) de colectivul condus de prof. Dan Modovan și prof. Sanda Harabagiu. Acest sistem este câștigătorul competiției TREC-9 cu un scor de relevanta a răspunsurilor de 0,58.

## 

# 2. Aplicații din domeniu

În acest capitol se face o introducere în metodologia folosită de sistemele inteligente de tip întrebare-răspuns, urmând a fi prezentate si câteva aplicații reprezentative domeniului.

## 2.1. Metodologia unui sistem întrebare-răspuns

Sisteme de tip întrebare – răspuns axate pe evaluarea în învățământ au intrat în acest an și în atenția renumitei competiții CLEF (Cross Language Evaluation Forum), care are ca provocare “Entrance Exams[[16]](#footnote-16)” - sub task pentru sistemele QA.

Prin aceasta se urmărește evaluarea sistemelor în aceeași manieră în care studenții sunt examinați pentru admiterea la facultate.

Pentru construirea unui sistem de tip întrebare-răspuns se folosesc două metode:

* Abordare de tip shallow, bazată pe cuvinte cheie. În această metodă se folosesc cuvinte cheie pentru a găsi pasaje și propoziții în text care ar putea reprezenta răspunsuri valide la întrebări.
* Abordarea de tip deep, ce implică o analiză mai sofisticată, o procesare sintactică, semantică și contextuală.

Sistemele din cea de-a doua categorie sunt superioare primelor. În funcție de complexitatea întrebărilor și de gradul de performanță dorit de la sistem, se poate alege o ramură din cele două prezentate mai sus.

## 2.2 Arhitectura unui sistem întrebare-răspuns

La începuturile inteligenței artificiale, în anii 1960, cercetătorii construiau sisteme capabile să răspundă la întrebări aparținând unor domenii restrânse (closed domains), în prezent dezvoltarea Internetului și pașii făcuți în ceea ce privește achiziția informației (information retrieval - IR) şi a tehnicilor de procesare a limbajului natural, precum şi cererea pentru acces facil la informație, a dus la creșterea interesului pentru sisteme care să ofere răspunsuri din domenii largi (open domains).

În aplicațiile care au menirea de a răspunde la întrebări puse în limbaj natural, prelucrarea parcurge trei etape principale: analiza întrebării, căutarea documentară și extragerea răspunsului.

### 2.2.1 Analiza întrebării

Transformă întrebările formulate în limbaj natural uman în interogări pentru motorul de achiziție de documente. Identifică tipului răspunsului așteptat. Și în plus, se identifică tipul întrebării, focusul întrebării, si mulțimea cuvintelor cheie relevante pentru întrebare.

### 2.2.2 Căutarea documentară

Urmărește extragerea paragrafelor relevante atașate fiecărei întrebări. Pentru realizarea acestui proces se indexează mai întâi corpusul pentru a putea identifica și extrage o cantitate cât mai redusă de informație relativ la o anumită întrebare. După care se face extragerea paragrafelor relevante folosind și cuvintele cheie identificate la pasul anterior.

### 2.2.3 Extragerea răspunsului

Etapa de extragerea a răspunsului se bazează pe etapele anterioare. Procesul de extragere depinde de tipul așteptat al răspunsului: când răspunsul are un anumit tip de entitate de tip nume, modulul de extragere a răspunsului identifică aceste entități în fiecare propoziție extrasă. Când tipul răspunsului nu este un nume de entitate, procesul de extragere se bazează în principal pe recunoașterea focusului, în acest caz șabloanele sintactice de găsire a răspunsului bazate pe focus fiind foarte importante.

## 2.3 Prima perioada : jocurile și demonstrarea de teoreme

După al doilea război mondial, inteligența artificială poate fi observată în programe care rezolvau puzzle-uri sau care jucau anumite jocuri. Pe lingă fascinația pe care o exercită asupra multora, au mai existat două motive pentru care jocurile au fost printre primele domenii de aplicare a inteligenței artificiale - întâi, că performanta programului este ușor de măsurat (de cele mai multe ori, sau câștigi sau pierzi un joc); apoi, că regulile sunt ,în general, simple și puține la număr, deci pot fi ușor descrise și folosite.

Deși această primă perioadă a ținut cam până în 1965, performanțele pe care le-a obținut inteligenta artificială n-au fost amețitoare, în primul rând pentru că nici o problemă cu adevărat semnificativă nu a fost rezolvată. Totuși, două concluzii s-au impus (care concluzii guvernează până astăzi disciplina de care ne ocupăm):

• majoritatea problemelor poate fi redusă la o problemă de căutare

• căutarea trebuie să fie călăuzită de anumite cunoștințe despre domeniul problemei

Jocurile de șah și de dame au fost primele alegeri. Ideea era una foarte simplă - fiind dată o poziție pe tablă, se încerca să se genereze toate secvențele posibile de mutări de la acel moment încolo, considerând că adversarul alege întotdeauna mutarea cea mai bună. Dacă o secvență ajungea într-o stare câștigătoare, atunci ea era cea de urmat. Problema practică de care s-a lovit această idee a fost că numărul combinațiilor de explorat era foarte mare (de exemplu, la șah, de ordinul $35^{100}$).

Concluzia imediată care s-a tras din această perioadă este că până și când este vorba de jocuri, e nevoie ca programul aibă cunoștințe adiționale (altele decât regulile jocului).

Programul de jucat dame al lui Samuels a fost unul din programele de referința din această perioadă. Acest program, pe lângă faptul că juca cu un adversar, își folosea experiența dobândită în partidele anterioare ca să-și îmbunătățească performanțele. El ținea minte anumite poziții ca din start câștigătoare sau dezastruoase și nu le mai calcula secvențele ulterioare de mutări (deci se comporta aproximativ ca omul din exemplul de mai sus, care nu-și mai pune problema să cedeze regina).

Demonstrarea de teoreme a fost celălalt domeniu care a suscitat interes în această primă perioadă. Și pentru acest domeniu performanțele sunt simplu de evaluat (supui spre demonstrare o teoremă clasică). Ceea ce trebuie să i se descrie calculatorului sunt setul de axiome și regulile de inferență (adică, regulile prin care se obțin noi adevăruri din niște adevăruri date). În această arie au fost create mai multe programe interesante, printre care ``The Logic Theorist'' al lui Newell, care demonstra teoreme din primul capitol al cărţii ``Principia mathematica'' de Whitehead şi Russell, şi un program al lui Gelenter care demonstra teoreme de geometrie.

## 2.4 A doua perioadă - înțelegerea limbajului natural

Anii 1965-1975 formează cea de a doua perioadă a inteligenței artificiale. Acum lumea se preocupă de ``înțelegere'', adică vrea să facă mașina să înțeleagă limbajul natural, în special povestiri și dialoguri.

Un proiect faimos al timpului a fost ELIZA[[17]](#footnote-17).

### 2.4.1 ELIZA

Este poate cel mai semnificativ exemplu pentru această perioadă. Programul ELIZA simula comportamentul unui psiholog, conversând în engleză cu pacienții. Aceasta a fost numită după Eliza Doolittle din piesa de teatru Pigmalion[[18]](#footnote-18), pentru a sublinia faptul că abilitățile sale lingvistice poate fi îmbunătățite în mod continuu de către un "profesor". Creatorul ei, Joseph Weizenbaum, a fost uimit să constate „cât de rapid și de puternic oamenii au ajuns să se implice emoțional în comunicarea cu calculatorul și cât de mult și-l imaginau ca pe o ființă umană''.

Dacă îi spun Elizei “sunt nebun”, ea pune în mișcare un dispozitiv de căutare a unor cuvinte special, cum ar fi - de pilda – “nebun”. O data găsit cuvântul, programul investighează un repertoriu de reacții si produce o replica de genul “de când ești nebun?”.

Cunoștințele programului despre psihologie sunt de fapt codificate sub forma unui set de reguli simple.

|  |
| --- |
| **Pacient:**Toţi bărbaţii sunt la fel.  **ELIZA:**In ce fel?  **Pacient:**Ne plictisesc întotdeauna cu cîte ceva.  **ELIZA:**Poţi să dai un exemplu?  **Pacient:**Păi, prietenul meu m-a trimis aici.  **ELIZA:**Prietenul tău te-a trimis aici.  **Pacient:**Spune că sunt mai tot timpul deprimată.  **ELIZA:** Îmi pare rău să aud că eşti deprimată.  **Pacient:**E adevărat. Sunt nefericită.  **ELIZA:**Crezi că venind aici ai putea fi ajutată să nu mai fi nefericită?  **Pacient:**Am nevoie de ajutor, măcar asta e clar.  **ELIZA:**Ce-ar însemna să primeşti ajutor?  **Pacient:**Poate aş învăţa să mă împac cu mama.  **ELIZA:**Povesteşte-mi despre familia ta. |

Figură 2 Dialog cu Eliza

Printre principalele probleme tehnice, cu care se confruntă aplicația sunt: identificarea cuvintelor cheie care apar în textul primit la intrare, găsirea contextului minimal în care este cuvântul cheie ales. De asemenea, alegerea unei reguli de transformare adecvat și desigur, realizarea transformări în sine. Furnizarea unui mecanism care va permite să răspundă "inteligent", atunci când textul de intrare nu conținea cuvinte cheie. În final alegerea și afișarea răspunsului.

Procedeul ELIZEI se numește pattern matching, adică “nu-nţelege dar le potrivește”.

Prin urmare, folosind reguli simple de prelucrare a limbajului natural, Eliza creează doar iluzia faptului că reușește să înțeleagă problemele pacientului.

Eliza nu este un partener de dialog. Efectul programului este acela de a întreține fluxul declarațiilor pacientului si de a provoca o ușurare a tensiunii nervoase prin confesiunile făcute Elizei.

Alte exemple de aplicații asemănătoare sunt Mathbot[[19]](#footnote-19), acesta răspunde la întrebări de matematică simple, cum ar fi adunarea, scădere, înmulțire, împărțire numerelor. De asemenea, Dr. Romulon[[20]](#footnote-20) care in plus față de Eliza are un set mai mare de răspunsuri, dar o memorie foarte slabă. PARRY este un alt exemplu de program, acesta a fost descris ca fiind "ELIZA cu atitudine", simula un bolnav de paranoia, iar motivul pentru care a fost construit era să testeze un model psihologic al paranoicului. Momentan nici una din aceste aplicații nu a reușit să treacă testul  [Turing.](http://ro.wikipedia.org/wiki/Testul_Turing)

### 2.4.2 AML și A.L.I.C.E.

Unii splecialiști susțin că A.L.I.C.E[[21]](#footnote-21) (The Artificial Linguistic Internet Computer Entity) și AIML[[22]](#footnote-22) (Artificial Intelligence Markup Language) sunt o extensie a vechiului program psihiatru Eliza. Comparația este corectă în ceea ce privește arhitectura stimul-răspuns. Dar în prezent aplicația A.L.I.C.E are mai mult de 40.000 de categorii de cunoștințe, în comparație cu Eliza, care are aproximativ doar 200.

A.L.I.C.E a câștigat premiul Loebner, un test Turing anual, în 2000 și 2001. Deși niciun calculator nu s-a dovedit mai inteligent decât oamenii cu care au concurat, A.L.I.C.E a fost clasată pe locul "cel mai uman calculator" de către două grupuri de jurați. Trecerea testul Turing nu este foarte clară. Factori cum ar fi vârsta, nivelul intelectual și așteptările judecătorilor au un rol important. Pentru a putea trece un Test Turing bine conceput, mașina trebuie să folosească limbajul natural, rațiunea, să dispună de cunoștințe și să fie capabilă și să învețe.

ALICE are un model de învățare care se numește învățare supravegheată, pentru că o

persoană (în engleză botmaster), joacă un rol foarte important. Persoana respectivă monitorizează conversațiile robotului și creează conținut nou AIML, pentru a face sa pară că răspunsurile sunt cât mai precise, adecvate și credibile.

Au fost dezvoltați algoritmi pentru detectarea automată a modelului în datele din dialog. Acest proces numit direcționare, oferă persoanei noi modele de intrare care nu au încă răspunsuri specifice, oferind astfel supravegherea și rafinarea continuă a robotului.

AIML a fost creat in perioada 1995-2000, având la bază următoarele concepte:

recunoașterea sau potrivirea pattern-urilor. Acesta este folosit pentru a modela limbajul natural utilizat în comunicarea dintre om și mașină. În acest scop este modelat un set de variante de texte introduse de utilizatori și pentru fiecare din aceste propoziții, sunt afișate utilizatorului răspunsuri pre-stabilite.

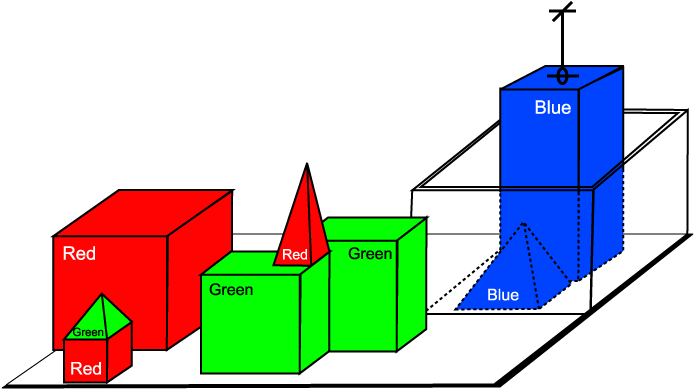
AIML are la bază limbajul XML fiind alcătuit din o serie de tag-uri particulare. Unitatea de bază AIML este categoria. Fiecare categorie este formată dintr-o întrebare (în engleză „pattern”), un răspuns (în engleză „template”) și, opțional mai poate apărea un context („that” sau „topic”). Limbajul AIML este simplu, fiind format din cuvinte, spații și simbolurile \_ și \*, care indică cuvinte sau secvențe de cuvinte.

A.L.I.C.E a fost prima aplicație, care a folosit limbajul și interpretatorul AIML, acesta fiind utilizat pentru specificarea regulilor de conversație. "Creierul" lui A.L.I.C.E este format aproximativ din 41000 de elemente numite categorii. Fiecare categorie combină o întrebare cu un răspuns, numite "model" și "răspuns"("pattern" și "template").

A.L.I.C.E a fost creat în primă instanță de Richard Wallace, în 1995. Funcționalitatea sa fiind împărțită în trei etape. Primul pas constă în primirea textului de la utilizator. În a doua etapă sistemul efectuează o procesare a textului pentru a potrivi textul de intrare cu un format prestabilit de proiectant. În final, răspunsul este afișat utilizatorului.

### 2.4.3 SHRDLU[[23]](#footnote-23)

Alt program interesant este SHRDLU, considerat de bună seamă una din culmile epocii. El era în stare să priceapă comenzi date in engleză. Aceste comenzi erau folosite pentru a modifica o lume de cubulețe, asemănătoare celor pe care copii le folosesc la joacă. De asemenea, SHRDLU răspundea unor întrebări legate de configurația blocurilor (de tipul ``Ce culoare are blocul de sub piramida roșie?''). Mai mult, SHRDLU putea să construiască planuri ca să îndeplinească comenzi de tipul ``Pune piramida albastră peste blocul verde''. Imaginați-vă că peste blocul verde mai erau așezate alte două blocuri. În cazul acesta, programul le îndepărta, ca să facă loc piramidei albastre!



Figură 3 SHRDLU Micro-World

Din păcate, lumea blocurilor era prea simplă, și ca atare SHRDLU n-a adus prea mare folos imediat în rezolvarea vreunei probleme concrete.

### 2.4.4 START

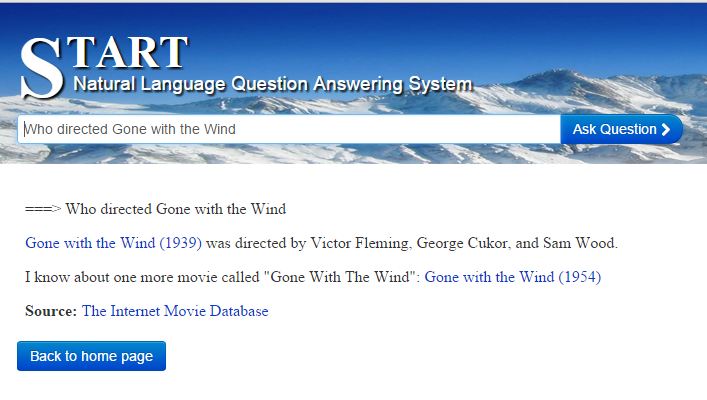
START[[24]](#footnote-24) (SynTactic Analysis using Reversible Transformations) este prima aplicație Web care folosește un sistem de tip întrebare-răspuns.

Acest sistem funcționează din decembrie 1993 a fost dezvoltat de Boris Katz[[25]](#footnote-25) și asociații săi din Grupul Infolab[[26]](#footnote-26) de la *MIT Computer Science and Artificial Intelligence Laboratory[[27]](#footnote-27)*. Spre deosebire de sistemele de găsire a informațiilor (de exemplu motoarele de căutare), START furnizează utilizatorilor "doar informațiile corecte" în loc de o listă mai mare cu informații relative. În prezent, sistemul poate răspunde la milioane de întrebări scrise în engleză, despre locuri (de exemplu orașe, țări, lacuri, coordonate geografice, vreme, hărți, sisteme politice și economice), filme (de exemplu titluri, actori, regizori), oameni (de exemplu date de naștere, biografii), definiții din dicționar și altele.

Prelucrarea limbajului natural de către START se împarte în două module care partajează aceeași gramatică. Modulul de înțelegere care analizează textului în limba engleză și produce o bază de cunoștințe care codifică informațiile găsite în text. Având în vedere un segment corespunzător al bazei de cunoștințe, modulul de generare creează fraze în engleză. Folosit în combinație cu tehnica de adnotare limbajului natural, serviciul multi-media de acces la informații utilizează puterea de procesare la nivel de propoziție a limbaj natural.

START parsează întrebări primite, apoi asociază interogările create de arbori de parsare cu baza sa de cunoștințe și prezintă informația cerută utilizatorului. În acest mod, START oferă utilizatorilor neinstruiți acces rapid la informații, care ar fi greu de găsit chiar și pentru un utilizator expert.

O tehnică importantă, numită "adnotarea limbajului natural" stă la baza aplicației START, ajutând să conecteze utilizatori la sursele de informare. Această tehnică folosește fraze ale limbajului natural și fraze adnotate pentru a descrie conținutul prin asocierea unor segmente de diferite granulații. Un segment de informație este preluat atunci când adnotarea se potrivește cu o întrebare primită la intrare. Această metodă permite sistemului START să se ocupe de o varietate procesări de tip media, cum ar fi texte, diagrame, imagini, videoclipuri, fișiere audio, pagini Web și altele.



Figură 4 Raspuns oferit de SMART

## 2.5 A treia perioadă - sisteme expert evoluate

Această perioadă se întinde cam din 1975 până în zilele noastre. Se poate spune că inteligenta artificială a devenit mai lucidă, mai critică cu privire la ea însăși, și într-o anumită măsură, mai pragmatică. Entuziasmele cu iz psihologic legate de înțelegere s-au mai temperat, și, în același timp, au apărut primele sisteme expert eficiente și cu folosire rentabilă în industrie. Interesul cade mai mult pe o reprezentare compactă și uniformă a cunoștințelor; lumea începe să se îndoiască de oportunitatea unor metode generale de rezolvare a problemelor.

Prin anii 80 a fost creat unul din primele sisteme expert folosite în industrie - R1 (acum numit XCON[[28]](#footnote-28)), construit la Universitatea Carnegie Mellon[[29]](#footnote-29), S.U.A cu colaborarea DEC (Digital Equipment Corporation[[30]](#footnote-30)). R1 se ocupă de configurarea unor sisteme de calcul (orice calculator VAX fabricat de DEC e configurat cu R1). El a fost pus în producție, și de atunci interesul lumii pentru inteligenta artificială a crescut considerabil. Ca urmare, informaticienii din această sfera s-au împărțit în ``implementatori'' de tehnici de inteligenta artificială și în cercetători.

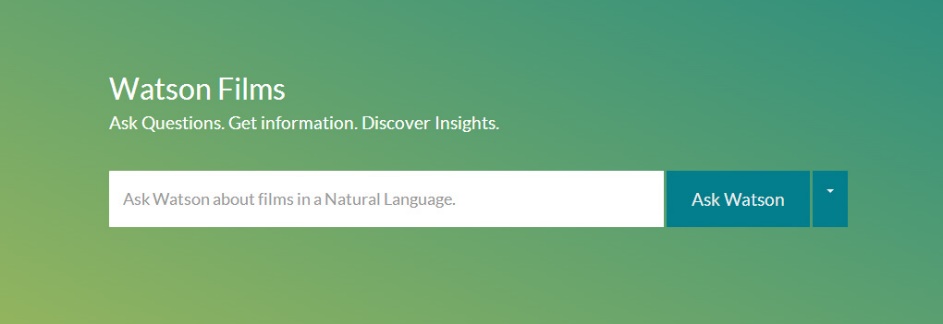
Tot acum s-au făcut unele progrese în domeniul învățării automate. De pildă, programul AM (Automated Mathematician [[31]](#footnote-31)- Matematicianul Automatizat), proiectat să descopere legi matematice, reușește să inducă concepte precum cardinalitatea și aritmetica întregilor, având drept cunoștințe inițiale conceptele și axiomele teoriei mulțimilor. AM este selectiv în sensul că reține numai cunoștințe ``interesante''; de asemenea, el își modifică gradual cunoștințele.

### 2.5.1 Watson[[32]](#footnote-32)

Watson este un sistem computerizat cu inteligență artificială, capabil de a răspunde la întrebări puse în limbaj natural. A fost dezvoltat prin proiectul DeepQA - IBM [[33]](#footnote-33)de către o echipă de cercetare condusă de David Ferrucci. Watson a fost numit după primul președinte IBM, Thomas J. Watson.

În 2011, pentru a se testa abilitățile sale, Watson a concurat în concursul Jeopardy!, fiind în acel moment prima înfruntare de acest fel între om și mașină. În trei emisiuni televizate, Watson l-a învins pe Brad Rutter[[34]](#footnote-34), cel care a câștigat cea mai mare sumă la acest concurs și pe Ken Jennings[[35]](#footnote-35), cel care a avut cele mai multe emisiuni câștigate (74). Watson a primit marele premiu de 1 milion dolari, în timp ce Ken Jennings și Brad Rutter au primit 300.000 dolari și respectiv 200.000 dolari. Jennings și Rutter s-au angajat să doneze jumătate din câștiguri, în timp ce IBM[[36]](#footnote-36) a împărțit câștigul lui Watson la două organizații de caritate.

Watson în mod constant și-a depășit adversarii umani, dar a avut probleme să răspundă la câteva categorii, în special întrebările care au indicii scurte și care conțin numai câteva cuvinte. Pentru fiecare indiciu, primele trei răspunsuri cele mai probabile ale lui Watson erau afișate pe un ecran de televiziune. Watson a avut acces la 200 de milioane de pagini cu conținut structurat și nestructurat care au însumat 4 Teraocteți de stocare pe disc, inclusiv textul integral al site-ului Wikipedia. Watson nu a fost conectat la Internet în timpul jocului.



Figură 5 Pagina principala a sistemului Watson

Watson este format dintr-un grup de 90 de servere IBM Power 750 (plus controlere suplimentare I/O, de rețea, etc) cu un total de 2880 procesoare multi-nucleu POWER7 și 16 Teraocteți de memorie RAM. Fiecare server POWER 750 utilizează un procesor de 3,5 GHz POWER7 cu opt nucele.

Potrivit lui John Rennie, Watson poate procesa 500 gigaocteți, echivalentul a unui milion de cărți, pe secundă. Tony Pearson estimează costul hardware al lui Watson la circa 3 milioane dolari.

IBM a anunțat achiziția AlchemyAPI[[37]](#footnote-37), un startup din Denver, ca parte a dezvoltării sistemului cu inteligență artificială Watson, relatează Associated Press. Până la acest moment, Watson putea procesa 500GB, echivalentul a unui milion de cărți, pe secundă.

AlchemyAPI foloseşte tehnologia pentru a ajuta companiile să gestioneze datele şi să le transforme în informaţii. Acesta lucrează cu 40.000 de programe din întrega lume pentru a analiza informații de pe site-uri, bloguri, mailuri și alte documente în opt limbi. Documentele preluate de AlchemyAPI din online ar putea rezolva problemele întâmpinate de vechiul sistem. Printre hibele vechiului Watson se numără şi răspunderea la întrebări care au indicii scurte formate din puţine cuvinte.

# Tehnologii utilizate

# 

## NP- Chunking, Lematizare și POS-tagger

Identificarea secvențelor nominale și verbale este una din sarcinile de nivel jos din procesarea limbajului natural, cu o importanță mare. Datorită complexității mai mici a problemei, față de, de exemplu, parsarea sintactică sau chiar si cea de dependență, metodele ce o abordează direct sunt mai robuste și erorile sunt mai locale.

Lematizarea este o operaţie de normalizare a formei ocurente a unui cuvânt, normalizare care elimină orice tip de flexiune[[38]](#footnote-38) din forma ocurentă a cuvântului. Este operaţia care transformă orice formă a unui cuvânt, folosind reguli de eliminare a flexiunilor achiziţionate, într-o formă standard care de obicei este adoptată de dicţionare. Lematizarea nu trebuie confundată cu indentificarea rădăcinii cuvântului.

Operaţia de lematizare urmează adnotarea cu etichete morfosintactice pentru că lema unui cuvânt depinde de categoria gramaticală a acestuia iar categoria gramaticală a unui cuvânt depinde la rândul ei de contextual de apariţie a cuvântului.

Procesul de lematizare va furniza, pentru fiecare formă următoarele rezultate:

* lema, unitate lexicală, reprezentată în mod convențional printr-o formă de bază (de exemplu infinitivul prezent activ al verbului, nominativul singular nearticulat al substantivului);
* clasa lexico-gramaticale a cuvântului (substantiv, adjectiv, verb etc.);
* subclasa lexico-gramaticală, dacă este cazul (exemplu substantiv propriu sau comun);
* în cazul cuvintelor flexibile, se stabilesc în plus celelalte determinări, dependente de clasa morfologică: pentru substantive și adjective sunt genul, numărul și cazul; pentru verbe sunt modul, timpul , persoana, numărul; pentru pronume sunt felul pronumelui, genul, numărul, cazul, persoana; pentru numerale sunt felul numeralului, genul, cazul (unde există).

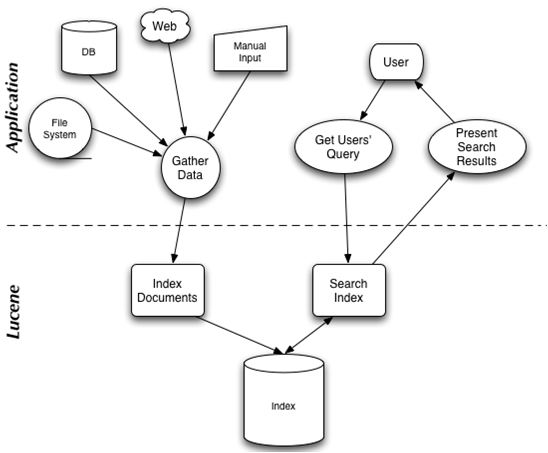
Categorisirea în părți gramaticale a cuvintelor dintr-un text are loc prin procesul de etichetare gramaticală, în engleză: ”part-of-speech tagging” prescurtat ”POS tagging”. Acest proces asociază fiecărui cuvânt un tag descriptiv luând în considerare mai multe trăsături relative la acesta precum cuvintele din dreapta și din stânga, prezența cifrelor sau a literelor majuscule.

Algoritmii care fac astfel de procesări sunt de obicei de două tipuri: bazați pe reguli sau bazați pe modele statistice. Metodele bazate pe reguli oferă mai mult control asupra rezultatelor decât modelele statistice, însă sunt greu de întreținut. Când există multe reguli care trebuiesc aplicate, apar conflicte care sunt dificil de urmărit. În ambele cazuri, rezultatele sunt imprecise.

## 3.2 Lucene[[39]](#footnote-39)

Apache Lucene este o bibliotecă performantă de funcții gratuită folosită în căutarea documentară, creată inițial în Java [[40]](#footnote-40)de Doug Cutting. Ea este un proiect open-source. Crearea ei a fost suportată de fundația Apache Software și este distribuită sub licența oferită de Apache Software. În prezent Lucene are suport pentru mai multe limbaje de programare, din care amintim Delphi, Perl, C#, C++, Python, Ruby și PHP.

Potrivită pentru orice aplicație care necesită indexare după text și posibilități de căutare, Lucene este recunoscută pentru facilitățile de căutare folosind atât motoarele de căutare de pe Internet cât și pentru căutarea locală. Librăria Lucene permite doar indexarea și căutarea și nu conține opțiuni de crawling (căutarea automată de pagini web) sau funcționalități de parsare HTML.



Figură 6 Modul de funcționare al Lucene

Conceptele fundamentale Lucene sunt documentul, câmpul și termenul. Documentul reprezintă o colecție de câmpuri. Termenul este unitatea de bază folosită în căutare. Acesta constă din perechi de șiruri de caractere formate din numele câmpului și valoarea acestui câmp.

În urma operațiilor de indexare a unei colecții de documente după regulile Lucene se obține un index. Un index stochează informații statistice despre termeni și permite o căutare foarte eficientă bazată pe acestea.

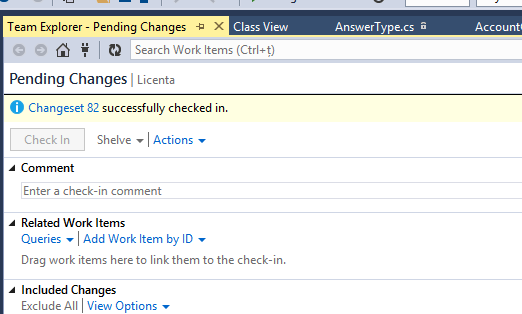
Interogările pe baza cărora se vor face căutările sunt create folosind lista de cuvinte cheie determinată în cadrul modulului de procesare a întrebării. Este important pentru succesul sistemului ca interogările să fie bine formate pentru că în documentele rezultate vor fi căutate răspunsurile finale. Dacă interogările sunt prea generale atunci relevanța documentelor rezultate va fi mult scăzută, iar dacă interogările sunt prea stricte numărul documentelor întoarse va fi mic și posibilitatea ca răspunsul să nu fie într-unul dintre ele va creste.

Relevanța unui document obținut de Lucene pornind de la o interogare a utilizatorului este un număr cuprins între 0 și 1 și este calculată cu ajutorul unei combinații dintre Modelul Spațiu Vector (VSM) al Regăsirii Informațiilor și Modelul Boolean. Ideea generală a VSM este următoarea: cu cât crește numărul de apariții ale unui termen din interogare într-un document relativ la numărul de apariții ale termenului în toate documentele din colecție, cu atât crește relevanța documentului respectiv în ceea ce privește interogarea.

Lucene este un motor de căutare care, deși utilizată tehnici complexe de indexare și regăsire, este extrem de ușor de folosit. O facilitatea importantă a motorului Lucene vine din faptul că putem adăuga oricâte câmpuri la index pe care le putem folosi mai apoi în procesul de căutare.

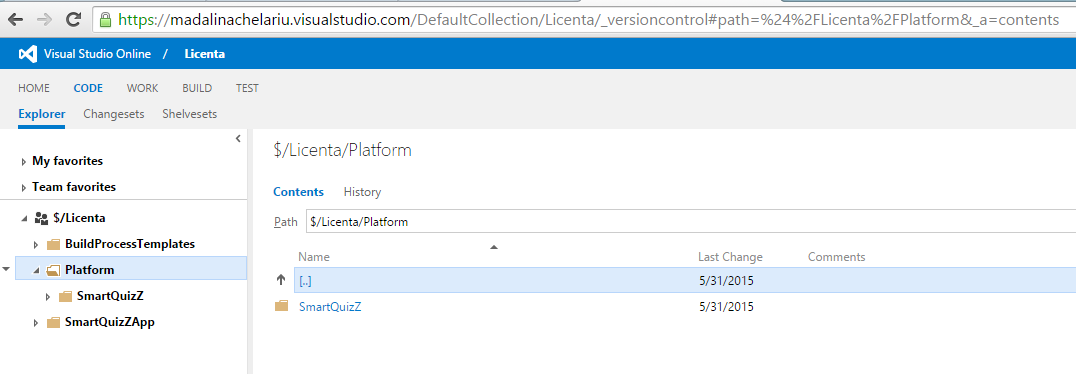
**3.3 TFS – Team Foundation Server**

Calitatea controlului codului sursă are un impact enorm asupra calității eventualului software construit, personalizat. De aceea, am lucrat cu instrumente pentru controlul codului sursă inovatoare şi de înaltă fiabilitate. Visual Studio Team Foundation Server (TFS) este produsul Microsoft pentru controlul codului sursă, pentru raportarea şi urmărirea proiectului. Depozitul de control al sursei este numit Team Foundation Version Control (TFVC). Cu acest produs se pot crea o mare varietate de rapoarte. De exemplu, rata de modificare a codului sursă în timp, sau o listă de bug-uri care nu au scenarii de test.



Figură 7 Changeset 82 pentru aplicatia Smart Quiz

Un alt aspect foarte important pentru TFS este posibilitatea de a hosta codul sursa în cloud, acesta putând fi accesat oricând de pe oricare mașina care are infrastructura necesară instalată.



Figură 8 Codul sursa al proiectului hostat online

## 3.4. ASP.NET MVC [[41]](#footnote-41)

ASP.NET MVC este o platformă de dezvoltare pentru aplicații web (site-uri, API-uri etc.) modernă, performantă, extensibilă, ușoară de învățat și de folosit. Bazându-se pe tehnologia .NET, se poate dezvolta utilizând orice limbaj de programare (C#, VB.NET, F# etc.) suportat de această tehnologie.

Fiind inspirată și de „Ruby on Rails[[42]](#footnote-42)”, platforma oferă multe facilități de accelerare a dezvoltării (model binding, validare, securitate etc.) fără să restricționeze însă creativitatea programatorului.

Pentru a modulariza aplicația, pentru a delimita în mod clar părțile componente spre a putea fi ușor modificate și ca după modificare sa fie încă compatibile cu celelalte module ce formează aplicația, este foarte utilă folosirea pattern-ul Model-View-Controller[[43]](#footnote-43) (MVC). Acesta separă aplicația în trei complonete prinicipale: modelul (Model), vizualizarea (View) și controller-ul (Controller).

*Figură 5: Modul in care funcționează o aplicație ASP.NET MVC*

Framework-ul ASP.NET MVC[[44]](#footnote-44) oferă o alternativă la ASP.NET Web Forms, model pentru crearea de aplicații web și este definit în namespace System.Web.Mvc.

Pattern-ul Model-View-Controller (MVC) separa o aplicație in trei componente principale:

*M: model (Model).* Modelul conține o mulțime de clase ce constituie datele cu care se lucrează si regulile de interacțiune ale acestora.

Model: Conține clase ce reprezintă domeniul aplicației. Aceste obiecte încapsulează adesea date memorate într-o baza de date precum si cod folosit pentru a manipula datele si a executa acțiuni specifice logicii aplicației. Cu ASP.NET MVC, acesta este văzut mai ales ca un Data Acces Layer – DAL – de un anumit tip, utilizând de exemplu Entity Framework sau NHibernate combinat cu cod specific logicii aplicației

*V: vizualizare (View User Interface).* View definește cum va arata interfața aplicației. View: View este un template pentru a genera in mod dinamic HTML.

*C: controler (Controller).* Controller-ul conține o mulțime de clase ce gestionează comunicația de la utilizator si logica specifica aplicației.

Controller: Controller este o clasa speciala ce gestionează relațiile dintre View si Model. Controller-ul răspunde la acțiunile utilizatorului, comunica cu modelul si decide ce vizualizare va afișa (daca exista una). In ASP.NET MVC[[45]](#footnote-45), numele acestei clase conține sufixul Controller.

## 3.5. JavaScript și CSS

JavaScript[[46]](#footnote-46) este folosit în general ca limbaj de scripting client-side pentru aplicaţii web, este inclus în pagini şi este interpretat de browser. A fost creat în 1995 de Brendan Eich pentru a aduce un plus de dinamism paginilor de web.

Avantajele folosirii Javascript în paginile unui sit web sunt: un plus de interactivitate cu utilizatorul, integrarea de validări/prelucrări suplimentare la nivelul clientului.

. CSS[[47]](#footnote-47) (Cascading Style Sheets) este un limbaj pentru stiluri folosit pentru a descrie aspectul și formatarea unui document scris într-un limbaj de marcare. Cel mai important rol al său este realizarea stilului paginilor web scrise în HTML și XHTML, dar limbajul poate fi aplicat oricărui tip de document XML.

CSS3 reprezintă un upgrade ce aduce câteva atribute noi și ajută la dezvoltarea noilor concepte in webdesign.

Unele dintre cele mai importante segmente (module) noi adăugate acestui standard pentru formatarea elementelor HTML aduc un plus considerabil in dezvoltarea activități webdesign.

### 3.5.1. [Bootstrap](http://getbootstrap.com/) [[48]](#footnote-48)

Framwevork-ul dezvoltat inițial de Twitter, ajuns acum la versiunea 3, permite realizarea de site-uri web responsive, care se adaptează la orice rezoluție de dispozitiv: desktop, tablete si telefoane mobile.

Este in momentul de fata cel mai utilizat framework pentru dezvoltarea interfetelor web devenind foarte rapid standardul in crearea template-urilor pentru principalele sisteme CMS cum sunt WordPress [[49]](#footnote-49)si Joomla[[50]](#footnote-50).

Cel mai important aspect pentru Bootstrap este acela că paginile web ce sunt construite cu acest framework sunt pagini adaptabile (în engleză responsive). Prin urmare paginile au capacitatea să-și adapteze layout-urile (designul) în funcție de mediul de vizionare**,** de rezoluția ecranului (dacă e desktop, laptop, tableta sau telefon) fără a suferi degradări majore, pentru a le oferi utilizatorilor o experiență excelentă de navigare.

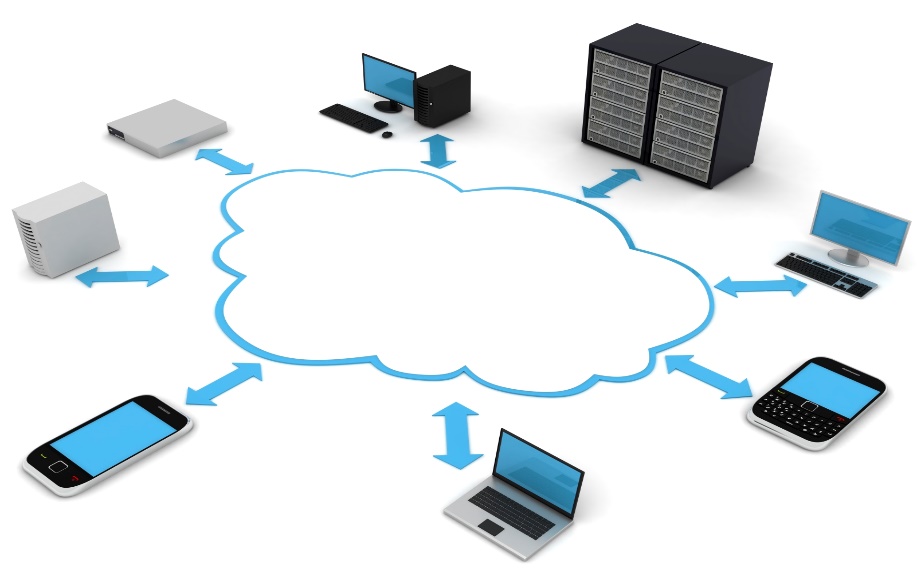
Twitter Bootstrap este un framework pentru design HTML/CSS și Javascript prin încapsularea unor funcționalități din jQuery. De asemenea, este un framework care are ca principală sarcină oferirea elementelor necesare pentru realizarea unui produs finisat într-un timp scurt. Ca proiect open-source, inițiat de echipa de dezvoltare de la Twitter, a prins foarte repede în comunitatea online, având un trend exponențial pe mai multe direcții. Astfel, nu doar că a ajuns la versiunea 3.0.x și că este printre cele mai populare proiecte open-source pe GitHub, dar în același timp librăria a declanșat dezvoltarea unor terțe librării și extensii care au ca scop completarea și rafinarea unui pachet întreg de resurse necesare în jurul acestui framework.

În privința suportului pentru device-uri mobile, Twitter Bootstrap nu poate înlocui un framework destinat exclusiv acestora, cum ar fi jQuery Mobile, care e construit și definit ca un framework web optimizat pentru smartphones și tablete, pe când Twitter Bootstrap oferă doar suport pentru acestea prin crearea de layout-uri responsive în funcție de device-ul care îl accesează.

## 3.6. Cloud computing

Cloud Computing este un domeniu relativ recent, dar de mare viitor. Este soluția pe care marile companii, si nu numai, o așteptau pentru a simplifica infrastructura de rețea, necesara pentru desfășurarea activității, pentru a reduce costurile echipamentelor si a întreținerii lor.

Cloud-ul are câteva avantaje clare, acesta simplifică sincronizarea datelor utilizatorului care folosește mai multe dispozitive legate (de exemplu un smartphone, o tabletă, un notebook, un PC). Documentele online din cloud se pot prelucra cu ajutorul unor aplicații web.Viteză de calcul și capacitate de stocare sporite, dar fără investiții în propria configurație. Datele nu pot fi furate, purtătorul de date nu se poate defecta.



*Figură 6: Cloud Computing[[51]](#footnote-51)*

Pentru a trece la nivelul următor era nevoie de o gândire noua. Astfel a apărut conceptul de Cloud Computing. Tehnicienii s-au gândit că un calculator poate funcționa și dacă tot hardware-ul sau este conținut de o sursa externa. Mai mult, tot software-ul sa fie conținut in acea sursa. Este o soluție perfecta pentru ca mai multe “calculatoare” pot apela la un singur calculator central si astfel dimensiunile sunt mult reduse, întreaga putere de calcul fiind într-un singur dispozitiv.

Se poate lua exemplul unei firme care are un număr de angajați. Fiecare angajat are propriul calculator cu monitor, unitate centrala si tastatura. Deja este o risipa mare de spațiu. Fiecare calculator are instalate pe el mai multe programe, fiecare cu propria licența. Chiar și sistemul de operare trebuie licențiat. Costurile pentru licențe sunt extrem de mari ca sa nu mai luam in considerare situația in care soft-urile trebuie actualizate sau situația in care ele cedează. Investițiile in aparatura sunt de obicei cele mai mari.

Rezolvarea situației vine din partea tehnologiei Cloud Computing. In loc sa existe in firma multe unitate centrale, va exista doar una cu putere de calcul mai mare, care va prelua sarcinile tuturor calculatoarelor. Pe acea unitate va fi instalat un soft necesar desfășurării activității care va fi distribuit pe toate calculatoarele. Singurul lucru de care vor trebui sa fie capabile calculatoarele este sa suporte o conexiune la internet si lucrul cu un browser.

Spațiul necesar muncii va fi micșorat si va trebui cumpărata licența unui singur program. Toate calculatoarele au fost înlocuite de unul singur. controla aplicațiile implementate și are posibilitatea de a configura setările mediului aplicație-gazdă.

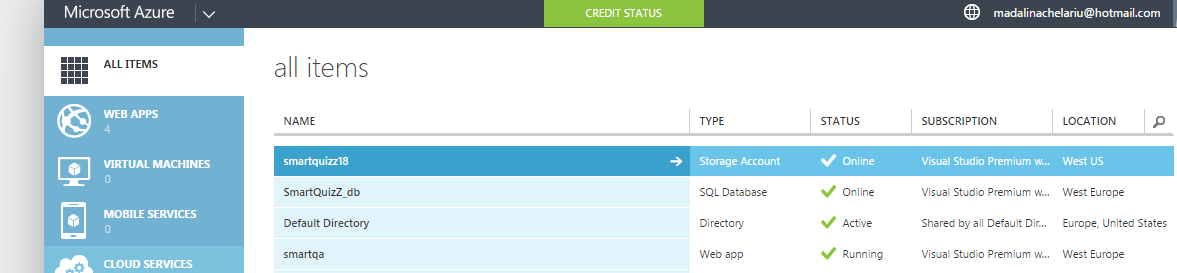
Serviciul platformă oferă clientului posibilitatea de previzionarea prelucrării, stocării, și a altor resurse de calcul fundamentale, în timp ce consumatorul are posibilitatea de a implementa si rula software-uri arbitrare, care pot include sisteme de operare și aplicații. Clientul are posibilitatea de a controla sistemul de operare și aplicațiile implementate și acces limitat la anumite componente ale rețelei.

### 3.6.1. Platforma Windows Azure

Windows Azure platform este o platformă la nivel de Internet pentru calcule și servicii de tip cloud găzduită în centrele de date Microsoft. Windows Azure platform include stratul de bază Windows Azure, precum și un set de servicii de dezvoltare ce se pot utiliza individual sau la un loc. Această platformă poate fi utilizată în mai multe feluri, de exemplu poate fi folosit pentru a dezvolta o aplicație care rulează și stochează datele în datacentere. Am ales să folosesc Azure deoarece este o platformă deschisă ce suporta orice aplicatie, facând ca implementarea unor sisteme informatice complexe să fie cat mai ușoară. Clienții închiriază resurse de calcul și stocare dintr-un volum foarte mare disponibil, și o fac în funcție de cerințele de moment. Clienții pot alege sa ruleze aceste aplicații în cloud sau pe mașini virtuale sau pe hdd, iar tranziția intre acestea se face rapid, fără modificări si costuri mari. Suportul oferit de tehnicienii Windows Azure este de cea mai buna calitate, la nivel Enterprise, si se adresează atât clienților din cloud, cat si celor ce folosesc mașini virtuale sau servere dedicate.

Una din cele mai importante funcții ale platformei Windows Azure este rularea de aplicații. Azure furnizează 4 opțiuni pentru a face asta: mașini virtuale, servicii cloud, site-uri Web, precum și servicii mobile.

Windows Azure oferă serviciul de infrastructură, permitând astfel dezvoltatorilor, specialistilor IT crearea și utilizarea mașinilor virtuale în cloud.



Figură 9 Portalul azure pentru gestionarea resusrselor stocate in cloud pentru acest sistem dezvoltat de mine

Windows Azure Web Sites oferă suport pentru site-uri web[[52]](#footnote-52), aplicații web diferite ce se pot baza pe tehnologii diverse: PHP, Java, Node.js, Pyton, .NET. Acesta rulează Windows Server și IIS[[53]](#footnote-53) într-o mașină virtuală.

Am ales ca și aplicația web și baza de date folosită să fie hostate in Azure, pentru a avea un control cât mai bun asupra lor de oriunde dar și pentru că platforma oferă posibilitatea ca site-ul web să fie publicat online și să poată fi accesat de toți utilizatorii de internet.

Baza de date Microsoft Windows Azure SQL extinde capabilitățile unui server obișnuit prin găzduirea în cloud. Folosind serviciul Windows Azure SQL Database un utilizator își poate crea o bază de date ușor de relaționat și de întreținut. Beneficiile sunt: manevrabilitate, grad foarte mare de disponibilitate, posibilitatea scalarizării, un model deja cunoscut de baze de date și un model de bază de date relațional.

Serviciul de Storage furnizat de Microsoft oferă aplicației acces la servicii de tip blob pentru date binare de dimensiuni mari, tabele pentru stocare non-relațională și queue pentru transmiterea de mesaje între aplicații.

# 4. Detalii de implementare

#### 5. Concluzii

Bibliografie

**Dan Cristea, Mădălina Ioniță, Ionuț Cristian Pistol** *Inteligență Artificială-* http://profs.info.uaic.ro/~dcristea/cursuri/IA/carteaIA.pdf, Editura Universității "Alexandru Ioan Cuza", Iași , 2007 .

**Radu Simionescu** *POS-tagger hybrid* http://nlptools.infoiasi.ro/WebPosRo/docs/POS%20tagger%20hibrid.pdf, [Lucrare], Iaşi, 2011.

1. https://en.wikipedia.org/wiki/Question\_answering [↑](#footnote-ref-1)
2. http://profs.info.uaic.ro/~alaiba/mw/index.php?title=Concepte\_%C3%AEn\_procesarea\_de\_limbaj\_natural [↑](#footnote-ref-2)
3. https://ro.wikipedia.org/wiki/World\_Wide\_Web [↑](#footnote-ref-3)
4. http://www.clef-initiative.eu/documents/71612/f39a1aa7-1946-4015-8732-3f22ef2d5560 [↑](#footnote-ref-4)
5. https://en.wikipedia.org/?title=Artificial\_intelligence [↑](#footnote-ref-5)
6. https://web.stanford.edu/class/cs124/lec/qa.pdf [↑](#footnote-ref-6)
7. http://www.celct.it/newsReader.php?id\_news=74 [↑](#footnote-ref-7)
8. https://gconea.files.wordpress.com/2009/01/pag7-formatul-xml.doc [↑](#footnote-ref-8)
9. http://www.bloguldesprestiinta.ro/2013/01/20/inteligenta-artificiala-si-prelucrarea-limbajului-natural/ [↑](#footnote-ref-9)
10. http://clef2015.clef-initiative.eu/CLEF2015/ [↑](#footnote-ref-10)
11. http://trec.nist.gov/ [↑](#footnote-ref-11)
12. http://profs.info.uaic.ro/~busaco/teach/courses/web/presentations/web10ServiciiWeb-SOA-SOAP-WSDL-UDDI.pdf [↑](#footnote-ref-12)
13. http://www.racai.ro/ [↑](#footnote-ref-13)
14. http://profs.info.uaic.ro/~corinfor/teach/syntax/proiecte/QA-RESDEC/SIR-RESDEC-scientific.pdf [↑](#footnote-ref-14)
15. http://www.utdallas.edu/ [↑](#footnote-ref-15)
16. http://nlp.uned.es/entrance-exams/ [↑](#footnote-ref-16)
17. https://www.cs.cmu.edu/~mihaib/articole/ai/ai-html.html [↑](#footnote-ref-17)
18. este o piesă de teatru scrisă în anul 1914 de dramaturgul irlandez George Bernard Shaw. [↑](#footnote-ref-18)
19. http://nlp-addiction.com/chatbot/mathbot/ [↑](#footnote-ref-19)
20. http://nlp-addiction.com/chatbot/dr.romulon/ [↑](#footnote-ref-20)
21. http://alice.pandorabots.com [↑](#footnote-ref-21)
22. http://www.pandorabots.com/pandora/pics/wallaceaimltutorial.html [↑](#footnote-ref-22)
23. http://hci.stanford.edu/winograd/shrdlu/ [↑](#footnote-ref-23)
24. http://start.csail.mit.edu/index.php [↑](#footnote-ref-24)
25. https://www.csail.mit.edu/user/820 [↑](#footnote-ref-25)
26. http://groups.csail.mit.edu/infolab/people.rhtml [↑](#footnote-ref-26)
27. http://en.wikipedia.org/wiki/MIT\_Computer\_Science\_and\_Artificial\_Intelligence\_Laboratory [↑](#footnote-ref-27)
28. http://www.prenhall.com/divisions/bp/app/alter/student/useful/ch12dec.html [↑](#footnote-ref-28)
29. http://www.cmu.edu/index.shtml [↑](#footnote-ref-29)
30. https://ro.wikipedia.org/wiki/Digital\_Equipment\_Corporation [↑](#footnote-ref-30)
31. https://en.wikipedia.org/wiki/Automated\_Mathematician [↑](#footnote-ref-31)
32. http://www.ibm.com/smarterplanet/us/en/ibmwatson/ [↑](#footnote-ref-32)
33. http://researcher.watson.ibm.com/researcher/view\_group.php?id=2099 [↑](#footnote-ref-33)
34. https://en.wikipedia.org/wiki/Brad\_Rutter [↑](#footnote-ref-34)
35. http://www.ken-jennings.com/ [↑](#footnote-ref-35)
36. http://www.ibm.com/en-us/ [↑](#footnote-ref-36)
37. http://www.alchemyapi.com/ [↑](#footnote-ref-37)
38. Totalitatea schimbărilor pe care le suferă forma unui cuvânt pentru a exprima diferite raporturi gramaticale. [↑](#footnote-ref-38)
39. http://lucene.apache.org/core/index.html [↑](#footnote-ref-39)
40. https://www.java.com/en/1 [↑](#footnote-ref-40)
41. http://www.asp.net/mvc [↑](#footnote-ref-41)
42. http://rubyonrails.org/ [↑](#footnote-ref-42)
43. http://msdn.microsoft.com/en-us/library/dd381412(v=vs.108).aspx [↑](#footnote-ref-43)
44. http://www.asp.net/mvc/overview/getting-started/introduction/getting-started [↑](#footnote-ref-44)
45. http://thor.info.uaic.ro/~iasimin/Special/ASP\_4\_MVC\_2014.pdf [↑](#footnote-ref-45)
46. http://www.slideshare.net/hksilviu/javascript-ajax [↑](#footnote-ref-46)
47. http://www.irdasys.ro/tehnologii-web-software/CSS/2 [↑](#footnote-ref-47)
48. http://getbootstrap.com/ [↑](#footnote-ref-48)
49. https://wordpress.com/wp-login.php [↑](#footnote-ref-49)
50. http://www.joomla.org/ [↑](#footnote-ref-50)
51. http://stiintasitehnica.com/inspector-gadget/cloud-computing-procesarea-intangibila [↑](#footnote-ref-51)
52. Web hosting [↑](#footnote-ref-52)
53. Internet Information Services [↑](#footnote-ref-53)