Információ és tudás integrálás (vimim222)

Kurzus: 2011-2012 ősz

Előadó: Strausz György

Tételek:

1. Információs tárházak feladatai, információ integrálás, mediátor megközelítés
2. Mediátor megoldások : Globálisan mediált nézet, lokális mediált nézet
3. Lokálisan mediált nézet., bucket (veder) algoritmus, inverz szabályok módszere
4. Adattér rendszerek
5. Szemantikus web koncepció
6. Szemantikus web technológiák (áttekintés, RDF, RDFS)
7. Nyelvtechnológia alapjai (morfológia, nyelvtanok, fordítás)
8. Információ integrációs modellek (SILK koncepció,IKF megközelítés)
9. Szakértő rendszerek, fejlesztő eszközök (CLIPS)
10. Jogi tudásalapú rendszerek

Források:

* <https://wiki.sch.bme.hu/bin/view/Infoszak/IntegrIntRendszFejl>
* <https://wiki.sch.bme.hu/pub/Infoszak/IntegrIntRendszFejl/iir_tetelek.doc>
* Google ☺

# Információs tárházak feladatai, információ integrálás, mediátor megközelítés

----------------------------------------------------------------------------------------

1. Információs tárházak feladatai, mediátor és integrálási feladatok

----------------------------------------------------------------------------------------

**információs tárházak**

Építsünk rendszert, amellyel le tudjuk kérdezni az információt a forrásokból: adatbázisokból, webről, mindenhonnan (heterogén források), és a felhasználó egységesen lássa az adatokat – egységes nézet kell az adatokhoz.

**információ integrálás szükségessége**

* elosztott heterogén információk weben
* nagy cég -> több információs rendszer (hogy ne 1től függjön a működés) -> össze kell hozni őket

**integrálandó információ lehet:**

* strukturált (adatbázis)
  + itt könnyebb az integráció, de ez is okoz nehézségeket: többféle adatbázis van, ugyanaz a cimke mást jelenthet, specifikálni kell
* félig strukturált (XML)
* strukturálatlan (szöveges)

**integráció formái / irányai:**

* virtuális integráció (csak „linkelés”) (adatbázisoknál: virtuális nézetek)
  + pl.: webáruház
  + előny: nem kell felesleges adatokat betölteni, kisebb adatforgalom -> olcsóbb, rugalmasabb, könnyebb módosítani (integrációs programot kell átírni)
* valódi (materializált) integráció (adatbázisoknál: valódi nézetek)
  + pl. nagyvállalat
  + kell hozzá gép, karbantartás, sok adat (adattárházak)
  + sok adat begyűjtése esetén ez a gyorsabb
* wrapper nem módosítja tartalmilag a lokális nézetet (ugyanaz az információ, de a struktúra más). De pl. szöveges infóból tartalomkinyerés kell

**mediátor feladatok**

* lekérdezés fordítás folyamata:
  + mediátor kap a felhasználótól a felhasználói nézetből egy lekérdezést
  + nem mindegy, hogy melyik forrásokhoz nyúlok először, hogyan join-olok -> optimalizálás
    - lekérdezési terv (lépésenként, időbecsléssel)
* források változásával ne kelljen felhasználói nézetet változtatni és kicsit kelljen csak a rendszert hangolni
* nem triviális, hogy egy lekérdezésre minden lehetséges választ megkapjak (nehéz a lekérdezést úgy átalakítani, hogy a lokális nézetekből minden adatot visszahozzon, főleg ha a források nem jól ismertek). Vagy pl. lehet olyan a forrás, hogy nem DB, hanem csak 1 választ ad mindig..
* minden elérhető forrásból az információk elérése, „értelmezése” a cél - ez a szemantikus web megközelítés a másik oldalról: vagyis hogyan dolgozzuk fel a szemantikus web által nyújtott adatokat? Fix téma, adott témakör -> mediátor dolga

**alkalmazások:**

* www (pl. infoMaster)
  + összehasonlításra képes vásárlóportálok -> depo.hu heterogén infókat nem képes dinamikusan kezelni, kiskerarak.hu -> csak megmutat
  + B2B, elektronikus piacterek
  + portálok több adatforrással
* tudomány és kultúra
  + géninformációk
  + asztrofizikai adatok
  + kulturális adatbázisok egységes elérése
* vállalati adatintegráció
  + bonyolult jelentések az 50 adatbázisból
  + modell alapú leírás kellene? Metaadat leírás xml-ben

Google kevés, mert dokumentumonként keres, nem integrál több dokumentumból adatokat

**InfoMaster**

* integrátor: sok szabályt tartalmaz (dim. Átváltások) (hierarchiában következtetés)
* Regal: szolgáltató, Payless: vásárló (majd tovább eladja)
* a Payless a Regal áruit árulja tovább, pl árak $-ból £-ba konvertálva
  + pl. tudjuk, hogy a Regal csak aluminium és rozsdamentes acélból gyárt árukat
  + a Payless a Regal-tól 33 terméket ajánl
  + a Regal nem azt adja meg, hogy miből van... (terméken hiányzik az info)
    - csak alu -> 0
    - csak acél -> 0
    - alu vagy acél -> 33 termék : feljebbmegy a hierarchiában
* meta-adatokat, kategória adatokat is tárol. Webes felületek a metaadatok megadására.
* attribútumok, értékkészletek.

**Try**

* ontológia szerű
* szabályok: adott szintaktika, alap és meta-szint keverésére

----------------------------------------------------------------------------------------

2 Információ integrálás, mediátor megközelítés

----------------------------------------------------------------------------------------

**adatintegráció, információ integráció**

* szemantikus web megközelítés a másik irányból

**nem elég adatbázisokat/elosztott adatbázisokat kezelni:**

* közös séma nincs (heterogén fogalmak, ontológiák, félig strukturált fogalmak)
* régi források: nem relációs sémák + eltérő elérési módok
* független források: nincs közös adminisztráció, nehezen előrejelezhető viselkedés
* általában csak olvashatók – bár terjednek a tranzakciókezelési megoldások

**feladatok:**

1. forrásleírás

* logikai szinten tudni, hogy miről van szó (pl. új autók)
* forrás-képességek (hogy kérdezhető le? SQL?)
* forrás teljessége (minden elemet tartalmaz?)
* fizikai jellemzők (hálózat technikai jellemzők)
* statisztikák az adatokról

2. forrás fúzió, lekérdezés tervezés, optimalizálás; forráselérés korlátai, minősége

3. monitorozás, végrehajtás

**MI dolga itt:**

* forrásleírásnál
  + tanulás -> bányászás fejlesztése
  + források felkutatása
  + forrás statisztikák készítése
  + wrappereket induktív módon tanítani
* forrás fúziónál
  + automata tervezés
  + nyelvek tervezése (amin a felhasználó megfogalmazhatja a kérést; transzformáció leírására)
  + szolgáltatások kompozíciós algoritmusának tervezése
  + monitorozás
* tudásreprezentálás
  + ontológiák
  + metaadatok
  + következtetés
  + lekérdező nyelvek

**méretek figyelembevétele:**

* hány forrás?
* strukturált adatok?
* követelmények: pontosság, teljesség
* zárt vagy nyilt világ feltételezés?

**kis forrásszám esetén (max 100):**

* nem virtuális mediátor, hanem kézzel írt ad-hoc program
* vagy adattárházak, periódikusan (éjszakánként) sok adat letöltése 12-18 hónap bevezetési idővel
* operációkhoz és döntéstámogatáshoz kellő adatok szétválasztása
* jó teljesítmény (helyben az adatok)
* rendszeres adattisztítás kell (+ források gyűjtése, stb.)

**virtuális integráció leírására**

* datalog progik
* a relációkat predikátumokkal írjuk le
* szabályok: Horn-klózok, amik nem tartalmazhatnak függvényeket
  + Pl.: személy (név, szül, anyjaneve) :- S1(név, szül), S2(név, anyjaneve); SQL-nél tágabb, de itt ekvivalens
* Virtuális integrációs sémák:
  + GAV
  + LAV (a forrásokat írjuk le a globális nézetben értelmezett „forrásokkal”)

# Mediátor megoldások : Globálisan mediált nézet, lokális mediált nézet

<http://en.wikipedia.org/wiki/Data_integration#Definitions>

----------------------------------------------------------------------------------------

3 Mediátor megoldások : Globálisan mediált nézet, lokális mediált nézet

----------------------------------------------------------------------------------------

**Mediátor:**

- forrásleírás

- virtuális, vagy materializált integráció

**GAV (global-as-view): globálisan mediált nézet**

* a globális nézetet írjuk le a lokális nézetek valamilyen relációjával
* pl.:

CREATE VIEW filmek AS

SELECT cím, rendező FROM S1

UNION

SELECT cím, évszám FROM S2

UNION: azonos címeknél összerakja 1 sorba

* ha változik S1, S2 (pl. más neve lesz) -> át kell írni a lekérdezést
* tehát jobb egy absztrakció: a forrásokat fordítva tekintem -> LAV

**LAV (local-as-view):**

* írjuk le a forrás nézeteket a globális relációjában

CREATE SOURCE S1

SELECT cim, evszam FROM filmek

* így a forrásokat írtam le, mintha lent ülne a felhasználó
* ha egy forrás kicserélődik -> elég az adott forrás definícióját kicserélni
* a felhasználói lekérdezések pl.: SELECT rendező FROM filmek WHERE filmek.evszam = 1966
  + nem triviális (GAV-ban egyszerűen csak behelyettesítés)
  + LAVnál visszafordítás kell: 2féle módszer
    - adatkiolvasás közben invertál
    - definíciót invertál
* bonyolítja a lekérdezés menetét, de a mediátor univerzálisabban működik: források jöhetnek-mehetnek: csak 1 szabályt kell átírni/törölni/beszúrni

**példa**

Globális nézet:

filmek(cím, rendező, év, típus)

források

S1 (cím, rendező, év, típus)

S2 (cím, rendező, év, típus)

S3 (cím, rendező)

S4 (cím, év, típus)

**GAV megoldás:**

CREATE VIEW filmek AS

SELECT \* FROM S1

UNION

SELECT \* FROM S2

UNION

SELECT S3.cím, S3.rendező, S4.év, S4.típus FROM S3, S4 WHERE S3.cím=S4.cím

Benne vannak a forrásokról való feltételezéseink. Független szolgáltatások -> unió. Tervezési kérdés lenne pl. hogy a cím különböző szolgáltatóknál egységes kulcsnak tekinthető-e (mások lehetnek az ontológiák -> S3.cím != S4.cím)

**LAV megoldás:**

CREATE SOURCE S1 AS

SELECT \* FROM filmek

CREATE SOURCE S2 AS

SELECT \* FROM filmek

CREATE SOURCE S3 AS

SELECT cím, rendező FROM filmek

CREATE SOURCE S5 AS

SELECT cím, rendező, év FROM filmek

WHERE év>1960 AND típus = „komédia”

SOURCE: ez is view tkp., csak megkülönböztetésként így nevezzük

LAV: a források „materializált nézetek” a mediált sémák felett. Mintha a globális sémában lenne adat.

**GAV vs. LAV összehasonlítás**

GAV

* nem moduláris (új forrásnál a globális sémát módosítani kell)
* nehéz úgy leírni a definíciót, hogy ne vesszen el információ (SELECT \* -ból nem derül ki egy megkötés pl. évre, típusra, stb.)
* lekérdezés fordítása egyszerűbb (kibontás, behelyettesítés)
  + könnyebb hierarchikus megközelítést alkalmazni (M1-M2-M3 .. mediátorok, nézetek egymás felett: bonyolult transzformációk több egyszerűbb lépcsőben leírhatók)
* akkor jó, ha: néhány, stabil forrás van

LAV

* moduláris (új forrásnál csak egy - a forrást definiáló nézetet kell beszúrni)
* teljes információ leírható könnyebben (rugalmas leírási lehetőség a források definiálására, a forrásokhoz vannak meta-információk)
* lekérdezés fordítása bonyolult
  + csak 1 szintre használják
* akkor jó, ha: sok, dinamikusan változó forrás
* pl.: InfoMaster

LAV-GAV példa:

Filmek (cím, rendező, év, típus)

Műsor (mozi, cím, idő)

S4 (mozi, típus)

GAV – veszteséges mediáció: hol játszanak vígjátékot?

CREATE VIEW filmek AS

SELECT NULL, NULL, NULL, típus

FROM S4

CREATE VIEW műsor AS

SELECT mozi, NULL, NULL

FROM S4

LAV – itt nem biztos, hogy le tudom fordítani a lekérdezést

CREATE SOURCE S4 AS

SELECT mozi, típus

FROM filmek f, műsor m

WHERE f.cím = m.cím

# Lokálisan mediált nézet., bucket (veder) algoritmus, inverz szabályok módszere

Logic\_based\_techniques\_in\_data\_integration\_levy-di00.pdf 12. oldaltól

----------------------------------------------------------------------------------------

4. Lokálisan mediált nézet., bucket (veder) algoritmus, inverz szabályok módszere

----------------------------------------------------------------------------------------

LAV fordításra:

* q(): koncepcionális nézeten megfogalmazott lekérdezés
* q’(): a forrásokon értelmezett válaszokat tartalmazza
* átírás: q() -> q’()

**Veder algoritmus**

példa:

Koncepcionális / globális nézet:

Szakirány (hallgató, tanszék)

Kurzus (kurzus, szám)

Beiratkozott (hallgató, kurzus, év)

Források:

V1 (hallgató, szám, év):- Beiratkozott(hallgató, kurzus, év), Kurzus(kurzus, szám), szám>500, év>1992

V2 (hallgató, tanszék, kurzus):- Beiratkozott(hallgató, kurzus, év), Szakirány(hallgató, tanszék)

V3 (hallgató, kurzus):- Beiratkozott(hallgató, kurzus, év), év<1990

V4 (hallgat, kurzus, szám):- Beiratkozott(hallgató, kurzus, év), Kurzus(kurzus, szám), Szakirány(hallgató, tanszék), szám<100

Lekérdezés:

q(H,T) : Szakirány(H,T), Beiratkozott(H,K,É), Kurzus(K,SZ), SZ>300, É>1995

(olyan hallgatók és tanszékeik, amely hallgatók 300 fősnél nagyobb kurzusra jártak 1995 után)

Vedrek:

|  |  |  |
| --- | --- | --- |
| **Szakirány (H,T)** | **Beiratkozott (H,K,É)** | **Kurzus (K,SZ)** |
| V2 (H,T,K’) | V1 (H,SZ’,É’) | V1 (H’,SZ,É’) |
| V4 (H,K’,SZ’) | V2 (H,T’,K) |  |
|  | V4 (H,K,SZ’) |  |
|  | nincs V3 (H,K), mert év<1990 miatt inkonzisztens feltételek |  |

Összes (H,T) 2-est 3-ast kiválasztva q átírható 6 darab lekérdezéssé, innen q’:

q’(H,T):-V1(H,SZ,É), V2(H,T,K), É>1995

**működés**

* lekérdezés a globális nézeten, ezt kell átfordítani a források nyelvére
* átalakítás jósága:
  + ekvivalens: ha tetszőleges adatbázis esetén ugyanazt a választ adja a lefordított kérdés, mint az eredeti adná
  + maximálisan tartalmazó: csak olyan válaszokat ad, ami az eredeti kérdésre is jó válasz (és a módszerrel többet nem lehet kinyerni) (pl. veder algoritmus)
* a veder algoritmus lényege: gyűjtsük össze az összes nézetet, ami a lekérdezés összes részcéljához adatot adhat -> minden részcélhoz veder, ezekbe gyűjtjük a nézeteket

nem elég vedreket csinálni az egyes részcélokhoz, mert néha csak a részcélok kombinációi adnak mintát és maguk a részcélok üresen maradnak

Pl. a részcélok tartalmaznak azonos változókat (megosztott, közös változók)

* vedrek kellenek:
  + minden a leképezésben szereplő részcélhoz
  + minden a leképezésben szereplő megosztott változóhoz

**Inverz szabályok módszere (pl. InfoMaster)**

* cél: invertálásnál ne veszítsük el, hogy milyen reláció volt az eredetiben
  + Pl.: V5(tanszék, kurzus) :- Szakirány(hallgató, tanszék) (-> 2-es, maradjanak egybe), Beiratkozott(hallgató, kurzus)
* gyártsunk annyi inverz szabályt, ahány tagja van a leírásnak. Vigyük át a kellő infókat:
  + minden hallgató, akinek tanszéke ez, kurzusa az
  + -> spec. függvény az összekötésre:

Szakirány ( f1(tanszék,x), tanszék) :- V5 (tanszék, x) (x: kurzus lehet csak)

Beiratkozott ( f1(y,kurzus), kurzus) :- V5 (y, kurzus) (y: tanszék lehet csak)

**Lekérdezés**

* azon tanszékeket keressük, melyekre szakirányos hallgatók járnak, és a kurzus neve amire a hallgatók feliratkoztak: IIR
* futtatható program:

q(T):-Szakirány(H,T), Beiratkozott(H,”IIR”)

--

(az eredeti nézet lefordítása:)

Szakirány ( f1(tanszék,x), tanszék) :- V5 (tanszék, x)

Beiratkozott ( f1(y,kurzus), kurzus) :- V5 (y, kurzus)

Pl. V5:

Tanszék Kurzus

MIT IIR

QIT IIR

MIT AI

A megoldás:

* a futtatás: behelyettesítjük q-ba + a szabályokba
* szakirány sor adja:

Szakirány ( f1(MIT, IIR), MIT )

Szakirány ( f1(QIT, IIR), QIT )

Szakirány ( f1(MIT, AI), MIT )

* beiratkozott sor adja:

Beiratkozott ( f1(MIT, IIR), IIR)

Beiratkozott ( f1(QIT, IIR), IIR)

Beiratkozott ( f1(MIT, AI), AI)

* illesztés : f1(MIT, IIR) ugyanaz fent, mint lent -> MIT-IIR lesz a találat

**Veder és Inverz-szab összehasonlítása:**

* inverz
  + a fordítást offline végzi (1 datalog program hajtja végre a lekérdezést)
* veder
  + online csinálja az átírást is
  + ezért előnye, hogy ha a lekérdezés olyan ami egyszerűsíti a modellt, akkor a lekérdezés is egyszerűbb lesz

# Adattér rendszerek

Principles of Dataspace Systems PDF 🡪 3., 4. rész

A Dataspace megközelítés a napjainkban felhasználásra kerülő nagy mennyiségű, heterogén adatforrásokból származó adatok kezeléséhez kíván segítséget nyújtani.

A dataspace résztvevőkből(participants), és a köztük lévő kapcsolatokból(relationships) áll.

A résztvevők az egyes adatforrások: relációs adatbázisok, XML fileok...

* Eltérő képességek: fejlett lekérdezőnyelvek VS korlátozott interface-ek
* Eltérő struktúráltság: struktúrált - rel. adatbázis, félig struktúrált - XML, struktúrálatlan
* Eltérő módosíthatóság: módosítható, csak hozzáírás, nem módosítható...

A kapcsolatok általánosak

* az egyik részvevő a másik egy nézete
* általánosabb is: az egyik két másikból lett kézzel előállítva...

### Dataspace szolgáltatások

* cataloging adatforrások leírása
* search interaktív, részleges eredmény után keresési kifejezés frissítése
* query
* monitorozás
* esemény detektálás

Mivel nem biztos, hogy minden résztvevő rendelkezik megfelelő interface-szel a DS funkciók használatához, ezért szükséges az adatforrások kiterjeszthetősége. Pl. résztvevő nem tárolja a saját meta adatait --> külső metadata repository

#### Catalog

Az adattér résztvevőinek és a kapcsolatok leírása

résztvevők leírása: séma forrás teljesség változási gyakoriság tulajdonos...

model management environment

* kapcsolatok létrehozása és manipulálása

#### Search & Query

* Kulcsszavas lekérdezés

Az eredmény mellé a lekérdezés finomítására szolgáló felület

* Struktúrált lekérdezés

Mediált sémákon lekérdezések - több forráson is A lekérdezések tövvféle nyelven is lehetnek - veszteségmentes fordítás közöttük

Metaadat lekérdezés

#### Local store & Index

* gyorsabb hozzáférés bizonyos információkhoz (cache)

#### Discovery component

résztvevők felfedezése

kapcsolatok létrehozása később további finomítás lehetséges (kézi)

#### Source extension componenet

Az egyszerűbb résztvevők kiterjesztése a szükséges kompnensekkel

# Szemantikus web koncepció

<http://www.w3.org/2002/Talks/2409-Budapest-IH/OverviewPrint.html>

<http://tmt.omikk.bme.hu/show_news.html?id=3648&issue_id=452>

# Szemantikus web technológiák (áttekintés, RDF, RDFS)

<http://www.w3.org/2002/Talks/2409-Budapest-IH/OverviewPrint.html>

<http://tmt.omikk.bme.hu/show_news.html?id=3648&issue_id=452>

# Nyelvtechnológia alapjai (morfológia, nyelvtanok, fordítás)

BME 2007-1.pdf - BME 2007-4.pdf nyelvtechnológiás diák, de elég érthetetlenek

**nyelvtechnológia**

* formális nyelvek a természetes nyelvek kutatásában
* nyelvmodellek és a nyelv távolsága nagy
  + a nyelv nem csak mondatok halmaza.. de a modellek erre épülnek (és fix szótárakra)
  + technikai eszközökkel leírhatjuk, hogy formálisan egy fix szótárból milyen mondatok állhatnak elő -> végtelen sok, egy része mondat, más része nem. De nincs éles határ.
* Elemzés > generálás (elemzés: megértem, de nem mondok ilyet)
  + Passzív szókincs, amit megértünk 100000 is lehet, amit használunk ~4-5000
* új szavak keletkeznek, toldalékok
* nyelvmodell jellemzők: pontosság és lefedettség (recall)
* túl- és alulgenerálás
  + nyelvhelyesség ellenőrző legyen elnéző -> összetett szavakat higyje el (túlgenerálás)
  + de amikor javaslatot ad, akkor kevesebbet adjon (alulgenerálás)

**paradigmák**

* statisztikai megközelítés
  + írott nyelv, ma már sok anyag (internet) -> statisztikai módszerek működhetnek (de hátrány: ami még sose fordult elő) -> szabály alapú
* szabály alapú
* példa-alapú rendszerek
  + szabály + statisztika (ez már előfeldolgozott adatokra)
  + ahol nincs elég példa, ott szabályt alkalmaz
  + kivételeket egyedileg írja le, többire egyszerű szabályokat ad

**-> korpusz-nyelvészet**

* sok szöveg van
* keressük: együttes előfordulások, kölcsönös információ
* treebank: szöveget elemez, az elemzést, az elemzett mondatokat tárolja el, ezek minták, így minden nyelvi szerkezetre van minta; gyorsan, jól használható
* Pl szeged corpus

**nyelvi kutatások szintjei**

* fonológia (hangzás)
* morfológia (formális nyelv, ragozás)
* szintaxis (formális, logikai, ragozás)
* szemantika (jelentés)
* pragmatika (használat)
* Szövegnyelvészet, dialóguskutatás: még magasabb szint, pl.:
  + A: el kéne menni.
  + B: el. -> önmagában nem mondat, de dialógussal érthető.

**az angol morfológia**

kis ragozás: walk, walks, walking, walked (magyar: eltarthatósági idő)

(magyar:) melléknév – főnévként is ragozható / néhány főnév még fokozható is -> jobb a szavakat a lehetséges toldalékaik szerint osztályozni

**morfológiai elemzés**

* egy feketedoboz ami az alábbi lépéseket végzi:
  + elemi morfémára bontás (szótő, ragok) (tő: lehet nehéz, pl. goes->go, de: went -> go)
  + meghatározza a morfémák lexikális alakját
  + meghatározza a morfémák morfoszintaktikai tulajdonságait
* praktikus tárolás:
  + tövek tára
  + toldalékok tára
  + trie: szófa (hatékonyan nem kezelhető szg-vel, de leíráshoz jó)

példa szófára:

o - to

that - that

e - the

is - this

magyarban: szófa és véges fordító

afa - almafa

almák - almák

val – almával

* morfológiai elemzéshez kapcsolódó alapfogalmak
  + szókészlet (ami nincs a szótárban azt nem tudom felismerni)
  + szótár ábrázolás
  + keresési lépések
  + túlgenerálás
  + zártság (formális módszerek zárt halmazokon dolgoznak, nem jöhet új elem; pedig a szg-nek kell új szavakat tanulnia -> nyilt szótár -> formálisan nem kezelhető)
* elemzés szabályhalmaz alapján: nem mindegy a szabályok sorrendje; elemzés nehéz (több-ből uaz lehet, vissza nem egyértelmű)
  + -> kétszíntű megfogalmazások (több szabály figyelembevétele)

|

-----------------

| | |

rule rule rule

FST FST FST

---> 1 nagy FST lesz (környezetfüggőnek látszik, de szűk nyelv miatt reguláris kifejezéssel leírható, hatékonyan működő FST készíthető

* két szint: felszíni és lexikális
  + lexical: move + ed
  + felszíni: moved
  + két véges automata az elemzéshez: egyik dolgozza folyamatosan a betűket, fenti példánál ’e’ betűnél megáll: a másik automata kezd el működni, gyakorlatilag előreolvas következő mélységig, és megállapítja, hogy ’ed’ toldalékolt szóról van szó
* a duplaszalagos egyszer végigmenős szalaggal nem kezelhetők a cserék:
  + teher-terhek, kehely-kelyhek
  + új szint: „terh” kaphat toldalékot, ez lesz a szótő, de csak toldalékoláskor
  + -> bonyolultabb lesz így a rendszer

Szabályok és lexikonok kompozíciója:

Szabályok véges automatája + lexikon véges automatája = 1 nagy véges automata

**folytatási osztályok** (pl. magyar ragozásnál)

|---------acc|dat|ins----------|

(S) --fn-- (A) --pl-- (C) --acc|dat|ins-- (D)

\ |

\ |

(B)--+ás-|

**szóalaktani alapséma**:

tő -> nemterminális toldalékok -> terminális toldalékok

* toldalék követelményeket szab az előtte lévő tőre? pl magas hangrend
* vagy a tőnek lehetnek követelményei a toldalékra (magas hangrend)

**HUMOR** (High Speed Unification Morphology)

* 2szintes morfológia helyett
* folytatási osztályok (mátrix formában: toldalék és tőtípusok összepárosítása)
* egyéb info : jegy-érték párokként tárolva. gyorsításhoz: pl magas hangrendű? előre legenerálva és eltárolva
* unifikáció: prologban is (jegynek neve, értéke van.. illesztés?)
  + összeférhetőség: baj ha 1 jegyre 2 érték akar jönni? a rszr csak megnézi hogy összeférnek-e, nem csinál új struktúrát az unifikációval
* jegy-érték nem csak atomi, hanem egy ösvény is lehet -> DAG-ok
* az unifikáció rekurzív definíciója:
  + 2 DAG (struktúra) unifikálható, ha az ösvényeken nem juthatok ellentmondásra (mindig "vagy" tipusúak, "kizáró vagyok" csak a leveleknél)

**szinonimaszótár**

egy másikra szóra cseréléshez magyarban elemezni kell, aztán az új szót megfelelően ragozni

szó -> szótő -> rokonértelmű szó -> ragozás (problémás)

**szintaxis**

* a nyelvi feldolgozás formális szintje
* közvetlen összetevős szerkezetek (pl be kellene fejezni) - fával leírható, rekurzív, zárójelezhető
* függőségi szerkezet
* nyelv szavai - gráf csomópontjai
* szavak hierarchikus viszonya : függőség
* csp-címke: a szó
* él-címke: nyelvi relációk (alanya, tárgya)

# Információ integrációs modellek (SILK koncepció,IKF megközelítés)

IKF koncepció (IKFBemutato.pdf) + leírás: <http://tmt.omikk.bme.hu/show_news.html?id=3616&issue_id=450>

silk2005.pdf diasor jó, ennek kivonata:

modellalapú integrációs eszközkészlet

SILK: system integration using logic and knowledge

* cél: heterogén információforrások integrációja
* jellemzők:
  + metaadat-kezelés, modellezés
  + logikai következtetés felhasználása
  + deklaratív információintegrálás, mediálás

**az evolúciós integrációs módszerek összehasonlítása**

**Típus Hozadék Tárgy Fő komponens Felhasználó**

alkalmazásint szinkronizáció folyamatok információbróker szervezet

adatint trendanalízis adattörténet adattárház döntéshozók

információint termelékenység élő adatok mediátor végfelhasználók

**ontológiák/metaadatok szerepe**

* Rengeteg adat áll a rendelkezésünkre
* A változással a metaadatok szintjén könnyebb lépést tartani
  + A modellek kisebbek
  + A modellek alapján automatikusan kell feldolgozni az adatokat
  + Az alkalmazásfejlesztésben is érvényes ez az elv (ld. MDA)
* Elosztott rendszereket nem igazán lehet metaadatok nélkül üzemeltetni

**SILK információintegráció**

* A három réteg alulról felfele:
  + különböz˝o forrásokból származó adatok
  + SILK: a mediálást végző komponens
  + információ különböző fogyasztók számára
* A fő SILK komponensek:
  + SILK Server a „futási idejű” komponens, a lekérdezéseket válaszolja meg
  + Modelltárház az integrációhoz szükséges modelleket tárolja
  + SILK Integrator a modelltárházat tartja karban

**A SILK felhasználási módjai**

* Modellezés
  + meglévő modellek betöltése
    - külső alkalmazásokból
    - magukból az adatforrásokból
  + új modellek kialakítása
  + modellek összekapcsolása
* Lekérdezés
  + külső eszközökkel vagy közvetlenül
  + a modelltárházbeli modelleken
  + egyszerre több adatforrást felhasználva
* Integrációs visszacsatolás
  + a kialakított új modellek és kapcsolatok alapján
  + javaslat egy integrált rendszer kialakítására
  + pl. ad hoc integrációhoz vagy adattárház kialakításához

**A SILK fő komponenseinek feladatai**

* Wrapper
  + elrejti a technológiai sokféleséget
  + adatot szolgáltat
  + metaadatot szolgáltat
* Mediátor
  + magasszintű, felhasználóspecifikus kérdéseket
  + visszavezet az egyes adatforrásokra vonatkozó részkérdésekre, és
  + a részeredmenyeket összeállítja egy értelmes válasszá
* Integrátor
  + támogatja a modellek kialakítását és kezelését
  + tartja a kapcsolatot a felhsználókkal és a külső alkalmazásokkal
  + tartja a kapcsolatot a Wrapper és a Mediator komponenssel

**modelltárház tartalma**

* objektumorientált modellek
* OCL korlátok az egyéb jellemzők számára
* leképezések (kapcsolatok a modellek között)

**adatforrások modellezése**

* Minden forrást az UML osztálydiagramok nyelvén kell modellezni
* Relációs adatbázisok
  + adatbázis -> modell
  + tábla -> osztály
  + oszlop -> attribútum
* XML-fájlok (DTD alapján)
  + fájl -> modell
  + elem -> osztály
  + attribútum -> attribútum
  + gyerekviszony -> kompozíciós asszociáció

# Szakértő rendszerek, fejlesztő eszközök (CLIPS)

CLIPS diasor (bogdan\_szakerto\_clips.ppt)

# Jogi tudásalapú rendszerek

diasor (JogiTudasAlapuRSZbevezetes.pdf)

<http://jesz.ajk.elte.hu/toth11.html>