

Unit tesztelés

Honfi Dávid, Micskei Zoltán

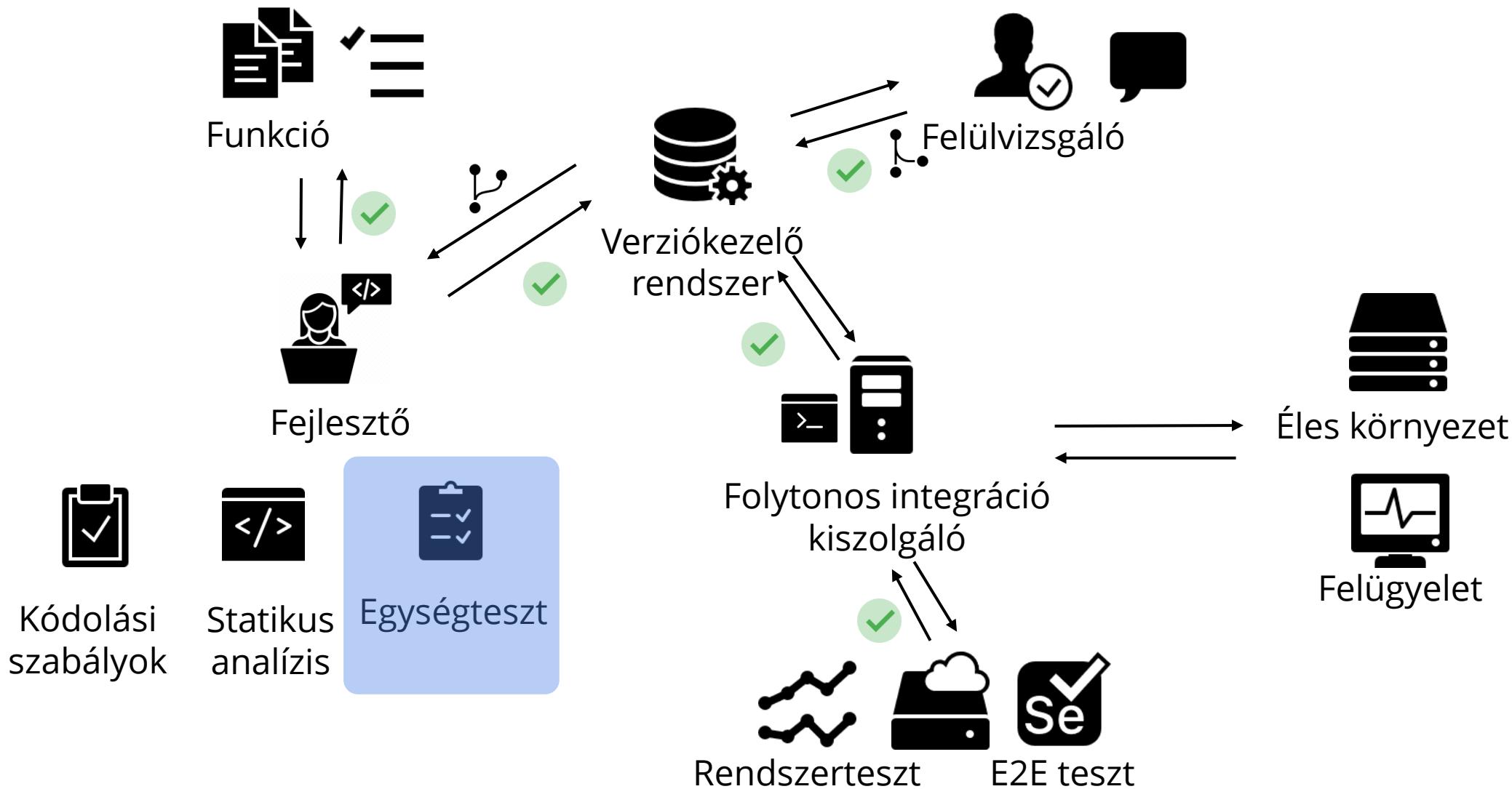


Méréstechnika és
Információs Rendszerek
Tanszék



Critical Systems
Research Group

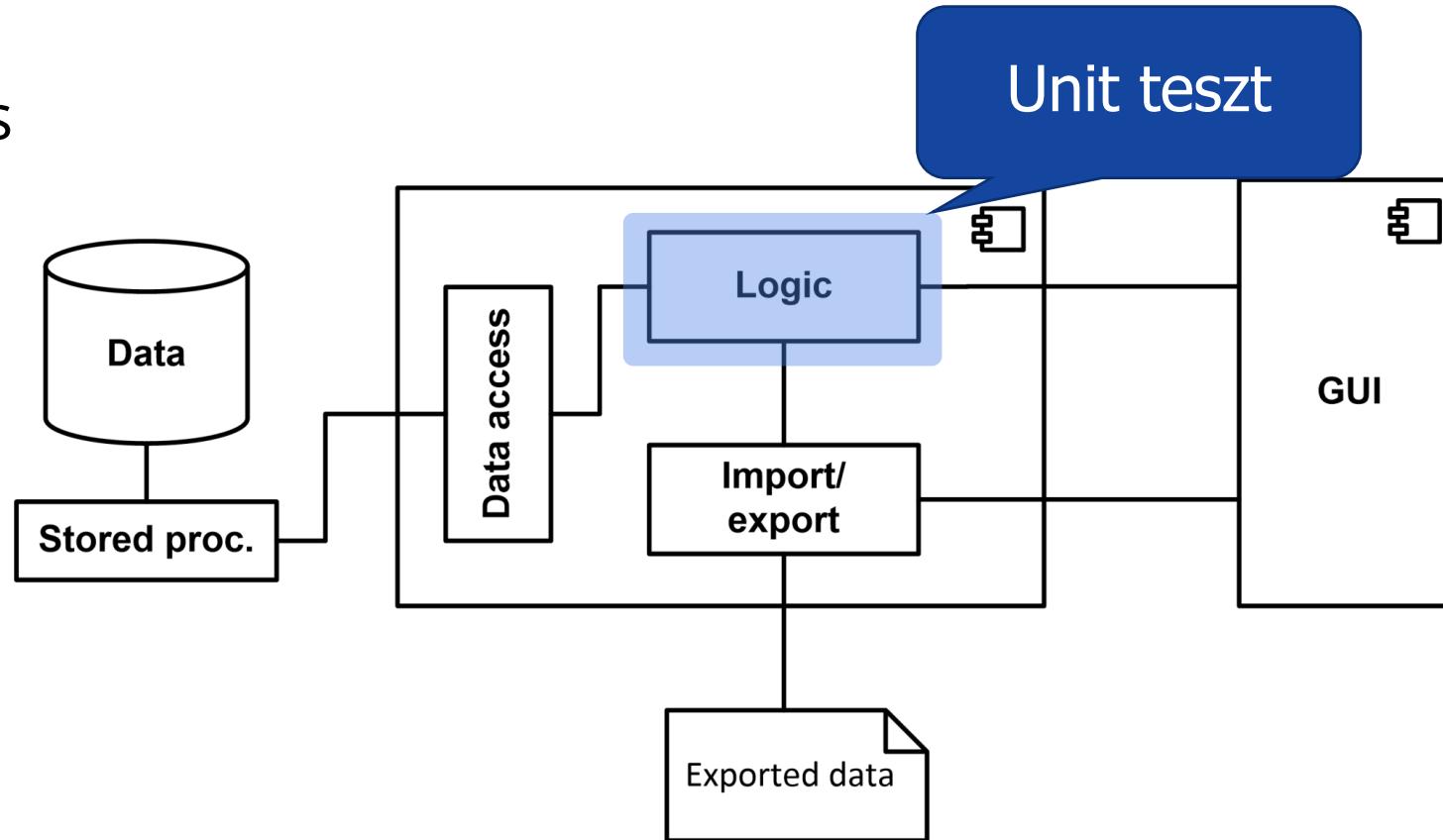
Tantárgy tematikája - Áttekintés



Ikonok: icons8.com

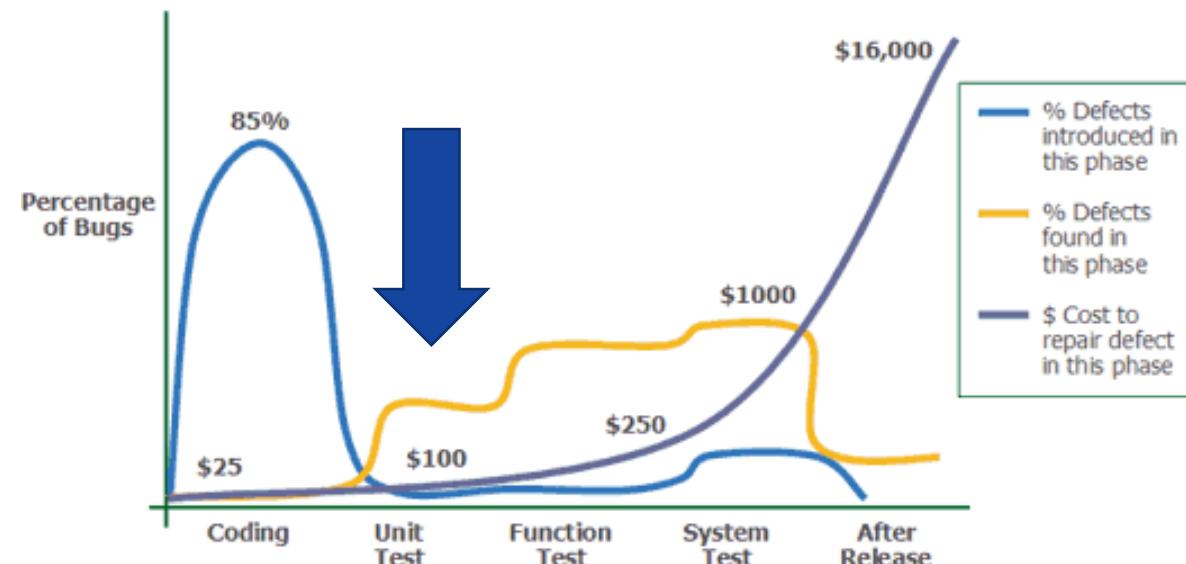
Mi a unit teszt?

- Tesztelt rendszer (SUT) egy részét vizsgálja
- Jól körülhatárolt rész: unit, unit under test (UUT)
 - Osztály
 - Metódus



Miért jók a unit tesztek?

- Könnyen elkészíthetők
- Laza csatolásra kényszerít: tesztelhetőség
- Azonnali visszajelzés
- Korai hibamegtalálási lehetőség



Forrás: *Applied Software Measurement*, Capers Jones, 1996

Hogy néz ki egy unit teszt? (JUnit)

```
public class ListTest{  
    List list; // SUT  
  
    @Before public void setUp(){  
        list = new List();  
    }  
  
    @Test public void add_EmptyList_Success(){  
        list.Add(1);  
        assertEquals(1, list.getSize());  
    }  
}
```

Tesztosztály

1. Beállítás

Teszteset

2. Végrehajtás

3. Ellenőrzés

4. Lebontás?

Alapelvezek



Milyen egy jó unit teszt?

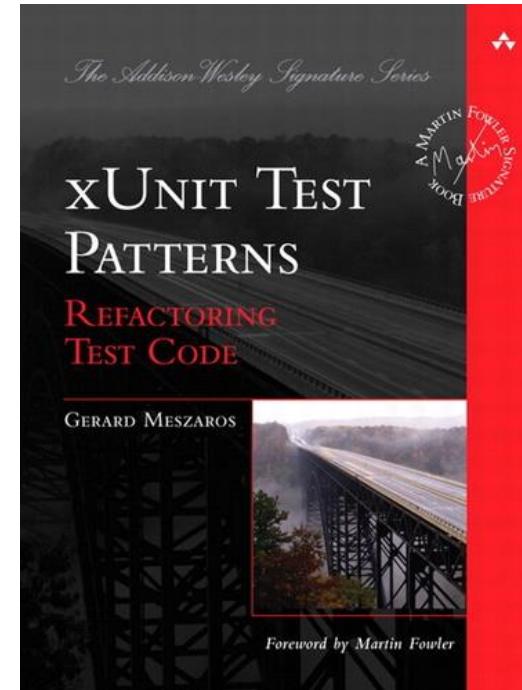
- A kód **egy** speciális funkcióját ellenőrzi
- „**Szerződés**” ellenőrzése a modul működéséről
- Példaként szolgálhat az egység használatához
- **Megbízható** (egyszerű, jó minőségű kód)

DEMO: Festival

- Meglévő unit tesztek átnézése
 - Teszt célok? Segít megérteni az egység működését?
- Elnevezési konvenciók javítása
 - Metódus neve, teszt szerkezete
- Üzleti funkciókat ellenőrizzen a teszt
 - Segédfüggvények, konstansok

Hogyan lehet jó a unit teszt?

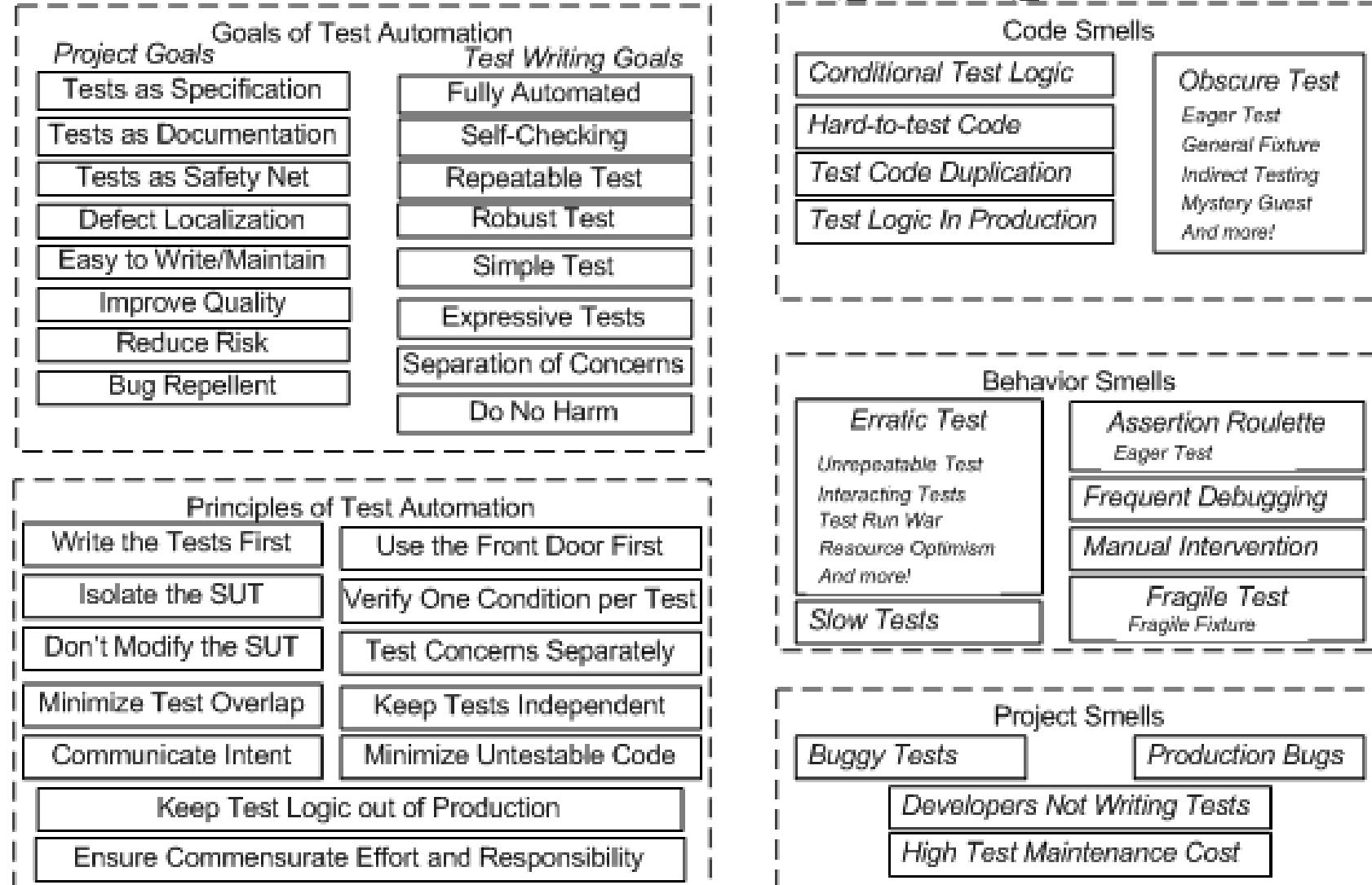
1. Alapelvek betartása
2. Bevált minták követése
3. „Smell”-ek elkerülése



Gerard Meszaros. 2006. *xUnit Test Patterns: Refactoring Test Code*.
Prentice Hall PTR, Upper Saddle River, NJ, USA.

<http://xunit.org>

xUnit Test Patterns könyv ajánlásai



Forrás: Gerard Meszaros, <http://xunitpatterns.com/>

Unit tesztelési minták

Alapelemek

I. Végrehajtás

- **Végrehajtó** (pl. xUnit, JUnit, MSTest)
 - Tesztek felsorolása, felfedezése, kiválasztása
 - Tesztek nyomon követése: darabszám, eredmények
- **Tesztkészlet:** csoportosításhoz
 - Tesztesetek összefogása tesztosztályba
 - Több tesztosztály összefogása
- **Felfedezés:** osztály vagy metódus szinten is
 - Elnevezés alapján
 - Attribútum segítségével
 - Könyvtárstruktúrával

II. Tesztdefiníció (Metódus)

- Egyszerű teszteset
 - „Happy path”: egyszerű lefutási útvonal, nagy kódfedés
 - Eredményeket ellenőrzi, nem vár kivételt
 - Arrange – Act – Assert törzs
- Elvárt kivételeles teszteset
 - A teszt futtatása során kivételt várunk
 - Ha nem keletkezik, a teszt hibát jelez
- Konstruktor teszteset
 - Ha egy konstruktor összetett logikát tartalmaz
 - Egyszerű teszteset, de elvárhat kivételt is (rossz argumentum)
 - Ellenőrzés: objektum állapota megfelelő-e

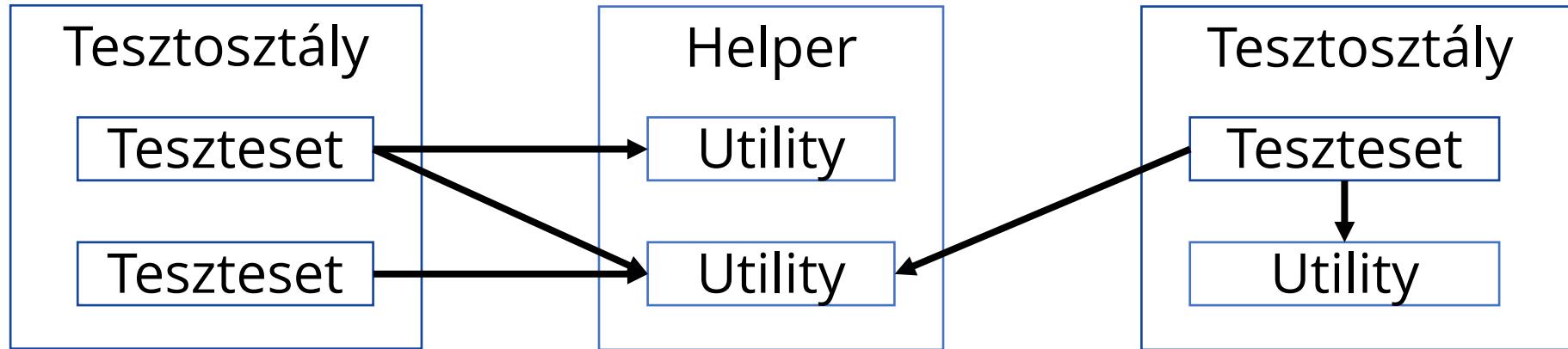
Elnevezési konvenciók

- Teszt osztály neve: **[Unit_neve]Test**
- Teszt metódus neve:
 - **Method_StateUnderTest_ExpectedBehavior**
 - **MethodName_DoesWhat_WhenTheseConditions**
 - **[feature being tested]**
- **Teszt szerkezete:**

// Arrange	// Given
// Act	// When
// Assert	// Then

II. Tesztdefiníció (Osztály)

- Nagyobb részt tesztmetódusokból épül fel
- *Utility metódus*: duplikáció elkerüléséhez
- *Helper*: osztály több tesztosztályhoz tartozó utility metódusok számára



III. Inicializálás

- **Friss**

- minden teszteset „friss” környezetben fut
- A környezet futás előtt közvetlenül kerül létrehozásra
- minden teszteset teljesen független egymástól

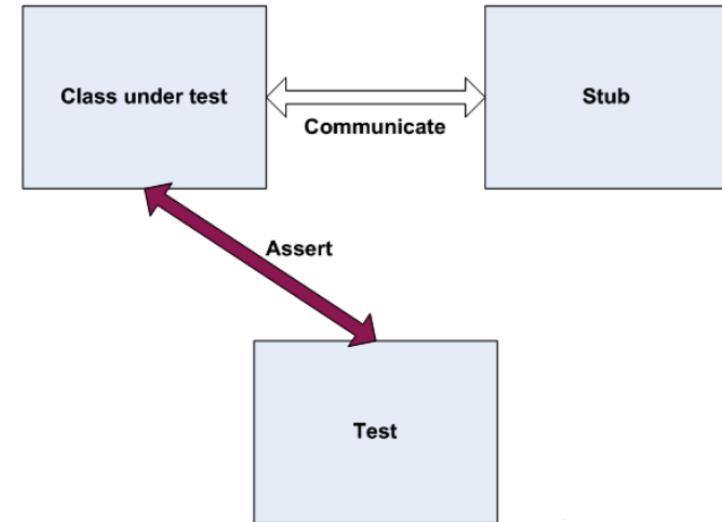
- **Megosztott**

- több eset használhat egy előre definiált környezetet
- futás idő csökkenthető
- **DE:** tesztesetek hatással lehetnek egymásra
- lehetőleg kerülendő megoldás
- használata csak ellenőrzött körülmények között

IV. Eredmények ellenőrzése

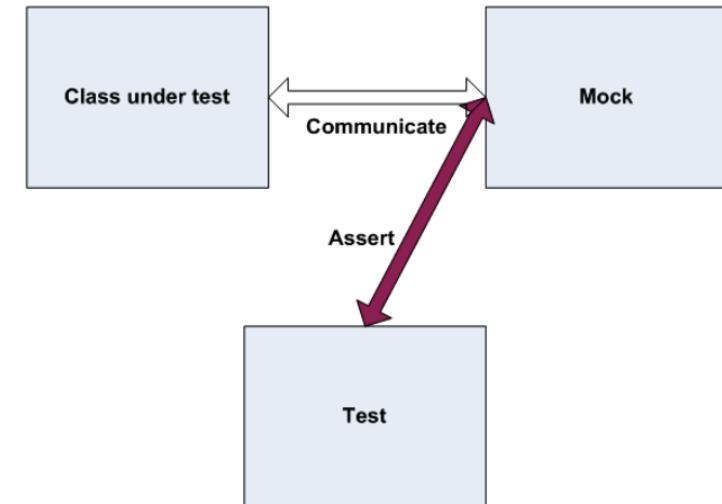
• Állapot ellenőrzése

- *Procedurális*: assertionök sorozatával
- *Objektum*: adott állapotú objektummal



• Viselkedés ellenőrzése

- *Procedurális*: a teszt futása közben elkapva
- *Elvárt*: teszt futása előtt definiált helyettesítővel



V. Befejezés

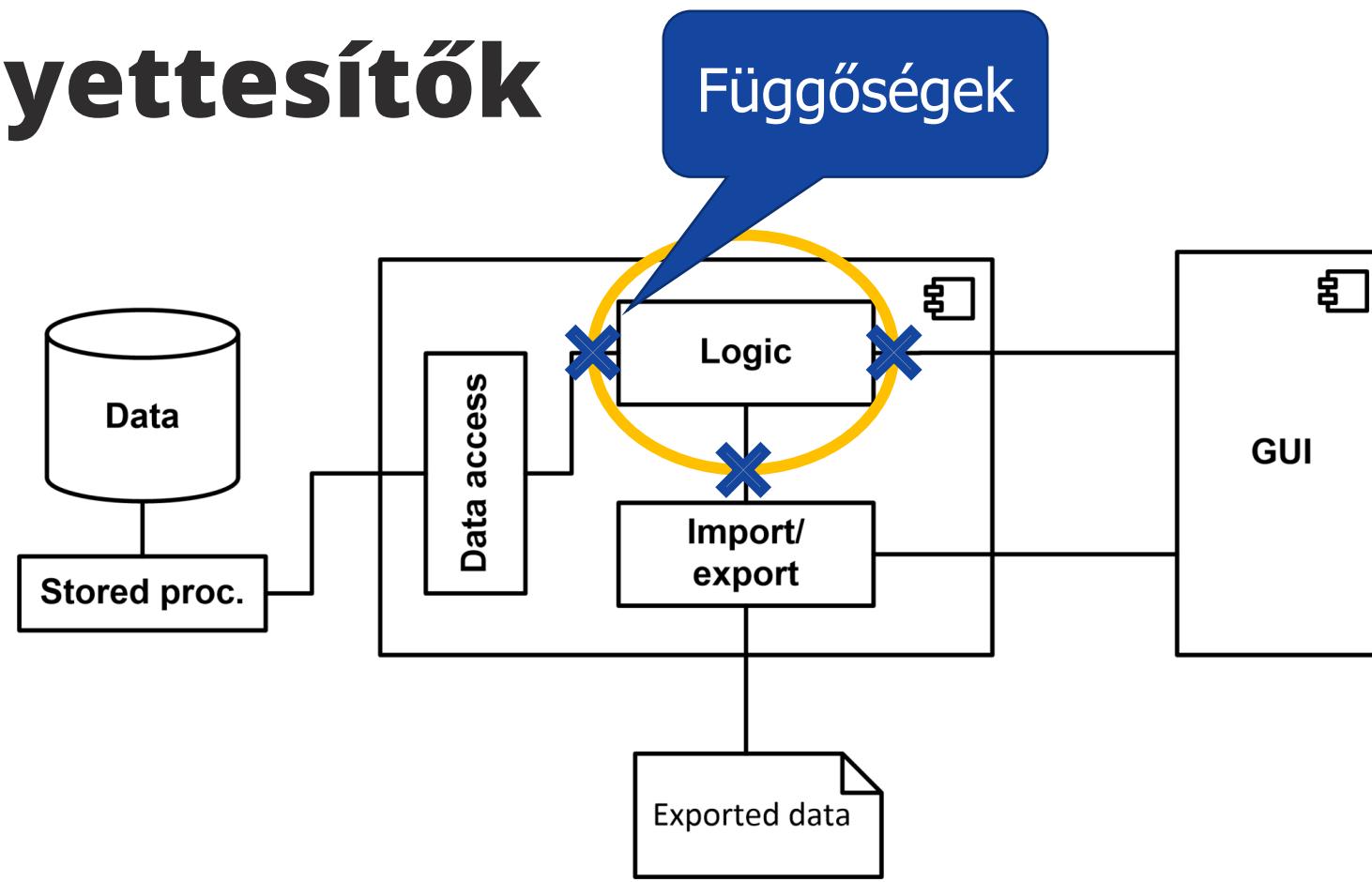
- **Stratégiák**

- *GC*
 - Referenciák megszüntetése
 - „Eszkalálás”
- *Automatikus*
 - Registry használata: pl. kollekció
 - Végén törlés egyesével

- **Szervezés**

- *Inline*: minden teszteset végén
- *Implicit*: futtató rendszer hív egy külön metódust

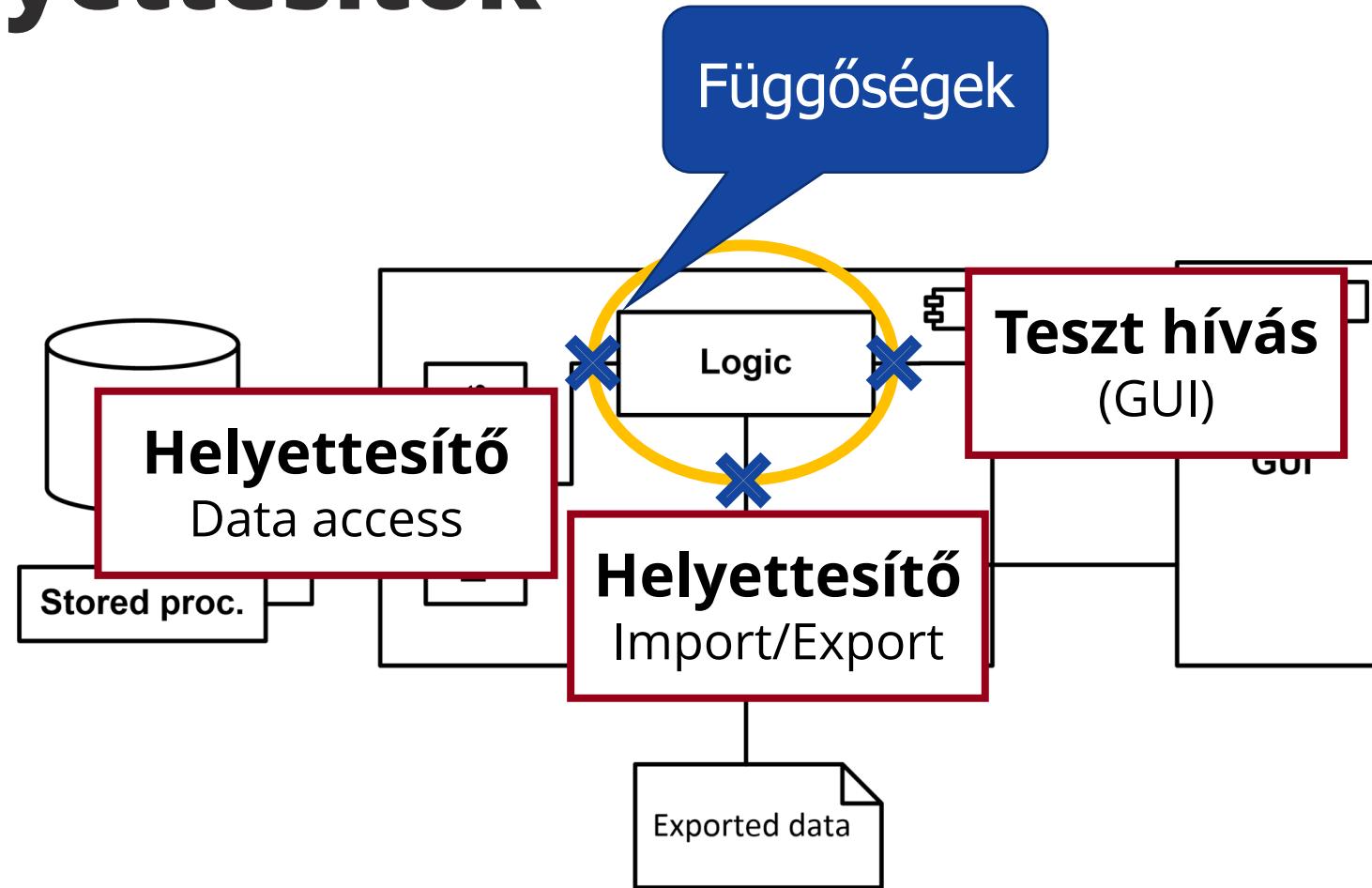
VI. Helyettesítők



• Helyettesítő objektumok

- Eredeti helyett használjuk: megfigyelés, beavatkozás
- Fontos: tesztelhetőségre tervezés

VI. Helyettesítők



A helyettesítők indirekt bemenetet adnak.

VI. Helyettesítők típusai

- **Stub**

- Előzetesen megadott értékeket ad vissza
- Nem tartalmaz ellenőrzést
- *Responder*: valid értékek beadása, „happy path”
- *Saboteur*: invalid értékek beadása
- *Átmeneti*: üres osztály, beégetett értékekkel

- **Spy**

- Indirekt kimenetek/hívások megfigyelésére (!)
- Tesztesetek végén spy ellenőrzése assertekkel
- Pl. anonim függvény átadása SUT-nak, registry

VI. Helyettesítők típusai

- **Mock objektum**

- Előzetesen definiált értékek visszaadása **ÉS** ellenőrzés
- Előzetes elvárások
 - Hívási argumentumok
 - Hívások darabszáma
- Hatással lehet a teszteset kimenetelére (assert)

- **Fake objektum**

- „Light-weight” implementáció (csak a lényeg marad)
- Interakciót képes kezelní (hívási szekvenciák)
- Példák: fake adatbázis, fake web service, fake réteg

DEMO: Festival

- Külső függőségek azonosítása
- Unit tesztelhetővé alakítása
- Helyettesítők használata a tesztben
- Instabil tesztfuttatás megszüntetése

Stub példa

```
public class TimeProviderStub implements TimeProvider {  
    private String time;  
    public TimeProviderStub(int hours, int minutes) {  
        this.time = hours.toString() + ":" + minutes.toString();  
    }  
    public String getTime() { return this.time; }  
}
```

Stub

```
@Test  
public void testDisplayTime_AtMidnight() {  
    // Arrange  
    TimeProviderStub tpStub = new TimeProviderStub(0, 0);  
    TimeDisplay td = new TimeDisplay(tpStub);  
  
    // Act  
    String htmlTime = td.getCurrentTimeAsHTML();  
  
    // Assert  
    assertEquals("Midnight", htmlTime);  
}
```

Teszt

Mock példa

```
public class PriceServiceTest{  
    ...  
    @Before public void init(){  
        mockDA = mock(DataAccess.class);  
        ps = new PriceService(mockDA);  
    }  
  
    @Test public void SuccessfulPriceQuery(){  
        // Arrange  
        when(mockDA.getProdPrice("A100")).thenReturn(50);  
  
        // Act  
        int p = ps.getPrice("A100");  
  
        // Assert  
        verify(mockDA, times(1)).getProdPrice("A100")  
    }  
    ...  
}
```

Megfelelő típusú
test double kérése

„Működési szabályok”
megadása

Milyen működést
kellett megfigyelnie

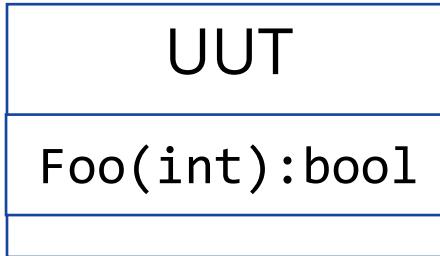
Unit tesztelési minták

Tesztelhetőségre
tervezés

Tesztelhetőségre tervezés

- **Tesztvezérelt tervezés** (Test-Driven Development)
 - Tesztek készülnek el elsőként (TDD)
 - A kész tesztek kikényszerítik a tesztelhető kódot
- **Vezérlési pontok** (Control point)
 - UUT vezérlése kívülről (API vagy back door)
 - UUT adott állapotba állítása
- **Megfigyelési pontok** (Observation point)
 - UUT állapotának lekérése
 - UUT kommunikációjának megfigyelése

Dependency injection



```
public bool Foo() {  
    C c = new C();  
    return c.getData();  
}
```

- A hívó félnek kelljen a függőségek átadását biztosítani!
- Lehetőségek
 - Konstruktorban
 - Paraméterként híváskor
 - Kettő között: setter
- Másik ötlet: *dependency lookup*

Hogyan adjuk át a stubot a C-hez a tesztben?

```
public bool Foo(C c) {  
    return c.getData();  
}
```

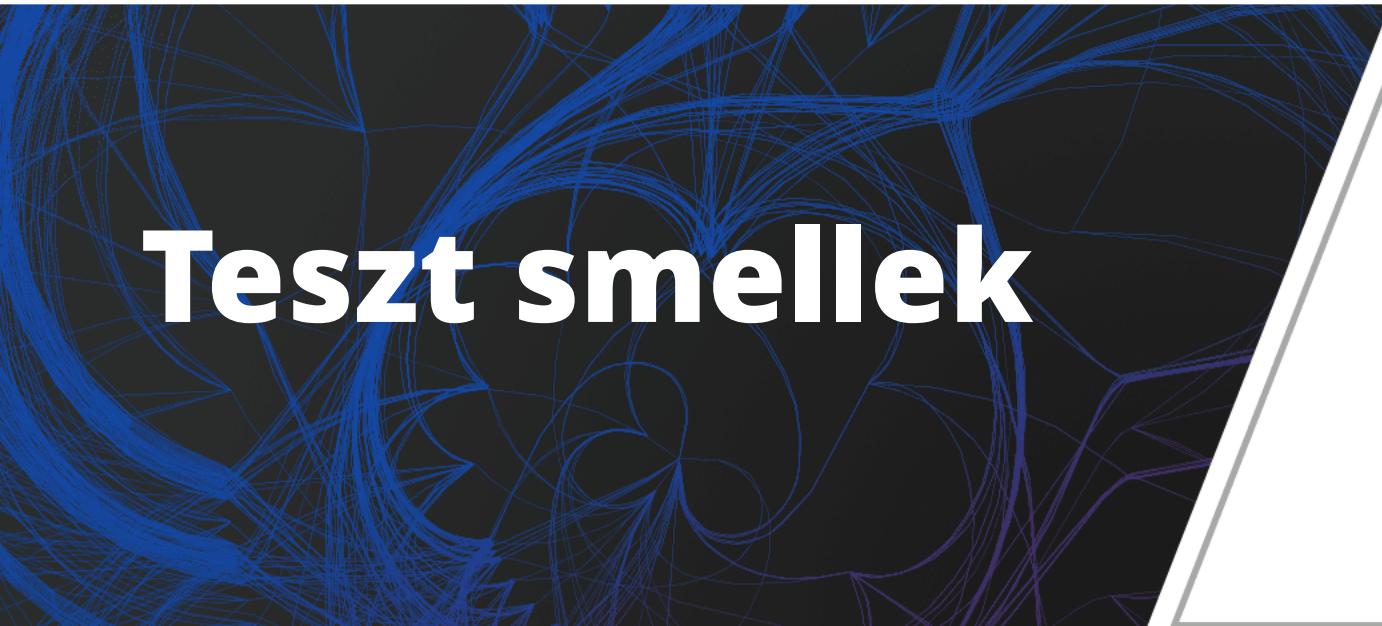
Humble objektum

- A lényeges logikát szeparáljuk a környezettől
- A szeparált rész könnyen tesztelhetővé válik
- Lehetőségek
 - Extract method („Poor man's")
 - Ősosztályok és leszármazottak
 - Külön osztály a kiszervezett logikával (delegálás)

Teszt hook

- Tesztelés módosítja a tesztelt viselkedést
 - Lehetőleg kerülendő – végső megoldás
 - Csak szigorú feltételek mellett használható
- A kód elágazásokat tartalmaz:
`if(testing) ...test... else ...prod...`
- A tesztelhetetlen részek tesztelhetővé válnak
- Pl. dátum lekérdezése – DateTime stub
 - Nem minden lehet injektálni a stub-ot
 - Teszteléskor DateTime.Now helyett adott inicializálás

Teszt smellek



I. Homályos (obscure) tesztek

Nehéz elsőre megérteni a tesztet.

- **Hatások**

- Magasabb fenntartási költségek
- Több bug maradhat benn a rosszul írt kód miatt

- **Okok**

- Nincs kellő figyelem a tesztek egyszerűségén
- In-line megoldások tesztekben: összetett logikák sora

I. Homályos tesztek tipikus esetei

- Kíváncsi (eager) teszt: túl sokat akar ellenőrizni
- Mystery Guest:
 - Nem látszik az akció-hatás összefüggés a tesztben
 - Pl.: a teszeset metódusán kívül is történik lépés
- Túláltalánosítás: felesleges környezet felépítése
- Beégetett tesztadatok
- Indirekció: a teszt az UUT-t indirekten éri el

Példa – Homályos teszt

```
public void testGetFlightsByFromAirport_OneOutboundFlight_mg() throws Exception {  
    loadAirportsAndFlightsFromFile("test-flights.csv");  
    // Exercise System  
    List flightsAtOrigin = facade.getFlightsByOriginAirportCode( "YYC" );  
    // Verify Outcome  
    assertEquals( 1, flightsAtOrigin.size() );  
    FlightDto firstFlight = (FlightDto) flightsAtOrigin.get(0);  
    assertEquals( "Calgary", firstFlight.getOriginCity() );  
}
```

II. Feltételes tesztlogika

**A tesztnek van olyan része,
ami nem fut le minden futtatáskor.**

- Hatások**

- Elbonyolítja a tesztet - nehezíti a megértést
- Nehezebb karbantartani

- Okok**

- Visszatéréstől függő tesztlogika
- Kölcsönhatások, összetett objektumok ellenőrzése
- Újrafelhasználhatóságra törekvés

III. Duplikáció

Ugyanaz a tesztkód többször ismétlődik.

- **Hatások**
 - Karbantarthatóság erős romlása
- **Okok**
 - Idő: CTRL+C és CTRL+V sokat „gyorsít” a tesztek írásán
 - Képesség: tesztfejlesztő refaktorálási képességei

IV. Assertion rulett

Nehéz megállapítani, hogy a sok assertion közül melyik okozza a teszt hibáját.

- **Hatások**

- Problémák azonosítása nehézzé, költségessé válik
- Megnöveli a hibajavítások idejét

- **Okok**

- Tesztesetek számának minimalizálása – több ellenőrzés
- Hiányzó assertion üzenet

Példa – Assertion rulett

```
public void testFlightMileage_asKm2() throws Exception {
    // setup fixture
    // exercise constructor
    Flight newFlight = new Flight(validFlightNumber);
    // verify constructed object
    assertEquals(validFlightNumber, newFlight.number);
    assertEquals("", newFlight.airlineCode);
    assertNull(newFlight.airline);
    // setup mileage
    newFlight.setMileage(1122);
    // exercise mileage translater
    int actualKilometres = newFlight.getMileageAsKm();
    // verify results
    int expectedKilometres = 1810;
    assertEquals(expectedKilometres, actualKilometres);
    // now try it with a cancelled flight:
    newFlight.cancel();
    try {
        newFlight.getMileageAsKm();
        fail("Expected exception");
    } catch (InvalidRequestException e) {
        assertEquals("Cannot get cancelled flight mileage", e.getMessage());
    }
}
```

V. Instabil teszt

**Ugyanaz a teszteset néha hibamentes,
néha hibát jelez.**

- **Okok**

- Nemdeterminizmus a tesztelt kódban
- Erőforrásszivárgás
- Erőforrás optimalizálás
- Interakciót végző teszt (pl. környezet)

VI. Törékeny teszt

Adott teszteset nem fordul, vagy hibásan fut egy olyan változtatás után, amit a teszt nem érint.

- **Okok**

- Indirekt tesztelés
- Kíváncsi teszt
- Interfész-érzékenység
- Viselkedés-érzékenység
- Adatérzékenység
- Kontextus-érzékenység

VII. Lassúság

Tesztfuttatás túl sok időt vesz igénybe.

- **Okok**

- Lassú a tesztelt komponens (pl. nem izolált)
- Általános környezet: túl sok inicializálás az elején
- Aszinkronitás
- Túl sok teszt

Összefoglalás

- **Unit teszt:** fontos a kódminőség/költségek miatt
- *Unit tesztek minősége nem triviális*
- **Alapelvek**
 - Tesztelhetőség
 - Szervezés
 - Automatizálás
- **Bevált minták:** tervezés, definíciók, helyettesítők
- **Smellek elkerülése**

Statikus ellenőrzési technikák

Hajdu Ákos, Micskei Zoltán, Majzik István

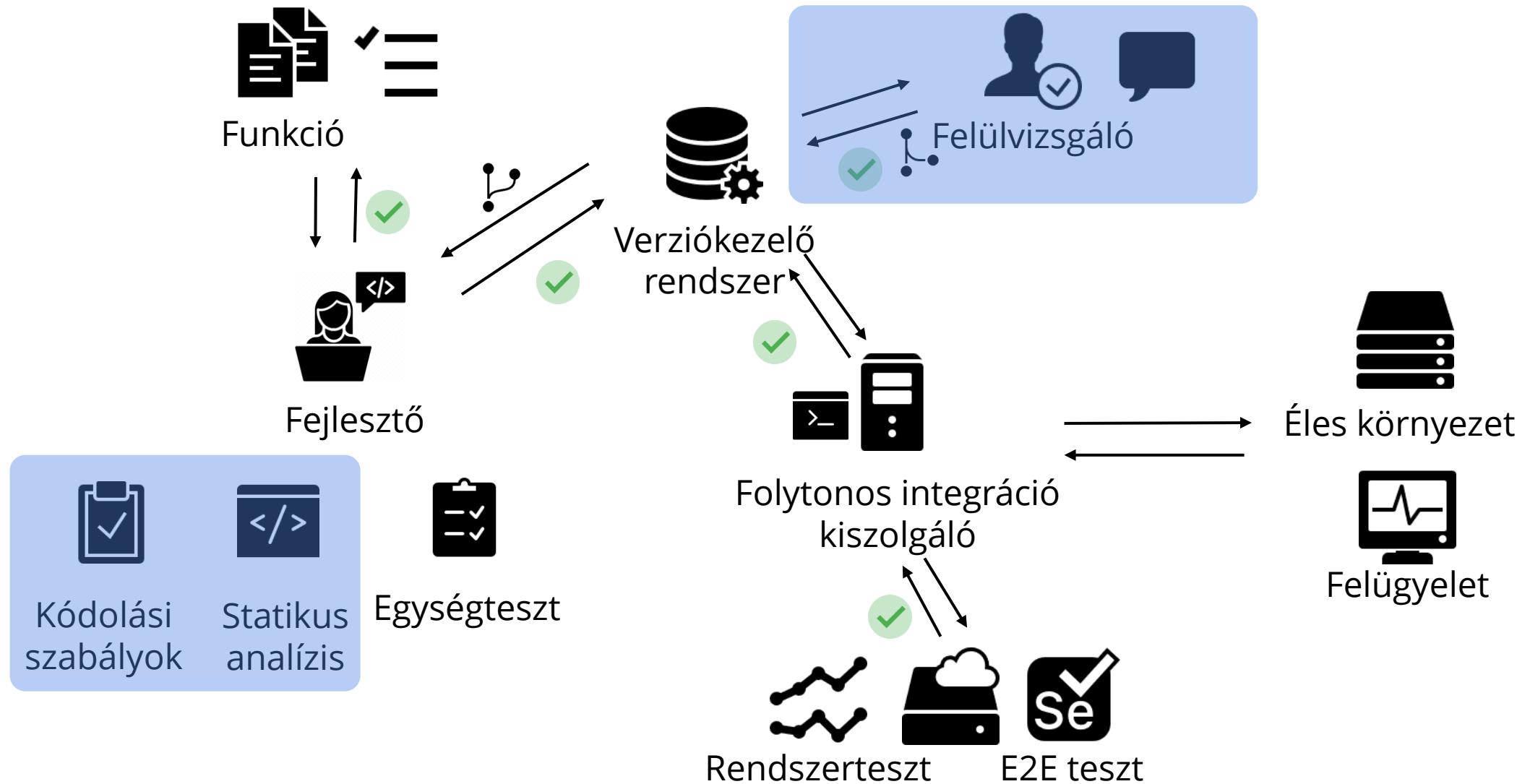


Méréstechnika és
Információs Rendszerek
Tanszék



Critical Systems
Research Group

Tantárgy tematikája - Áttekintés



Ikonok: icons8.com

Bevezető

- **Statikus ellenőrzési technikák**
 - Szoftver vizsgálata [végrehajtás nélkül](#)
- **Előny:** akkor is elvégezhető, ha
 - Még nem futtatható a szoftver
 - Költséges a futtatás
 - Nem áll rendelkezésre bemenet

Motiváció – Rossz példa

```
1 public class Class1
2 {
3     public decimal Calculate(decimal amount, int type, int years) {
4         decimal result = 0;
5         decimal disc = (years > 5) ? (decimal)5/100 : (decimal)years/100;
6         if (type == 1) result = amount;
7         else if (type == 2)
8         {
9             result = (amount - (0.1m * amount)) - disc * (amount - (0.1m * amount));
10        }
11        else if (type == 3) { result = (0.7m * amount) - disc * (0.7m * amount); }
12        else if (type == 4) {
13            result = (amount - (0.5m * amount)) - disc * (amount - (0.5m * amount));
14        }
15        return result;
16    }
17 }
```

<http://www.codeproject.com/Articles/1083348/Csharp-BAD-PRACTICES-Learn-how-to-make-a-good-code>

Jó forráskód tulajdonságai

Szintaktikailag helyes

- Fordító ellenőrzi

Jó minőségű

- Olvasható, újrafelhasználható, karbantartható, ...
- **Kódolási irányelvek** segítenek

Hibától mentes

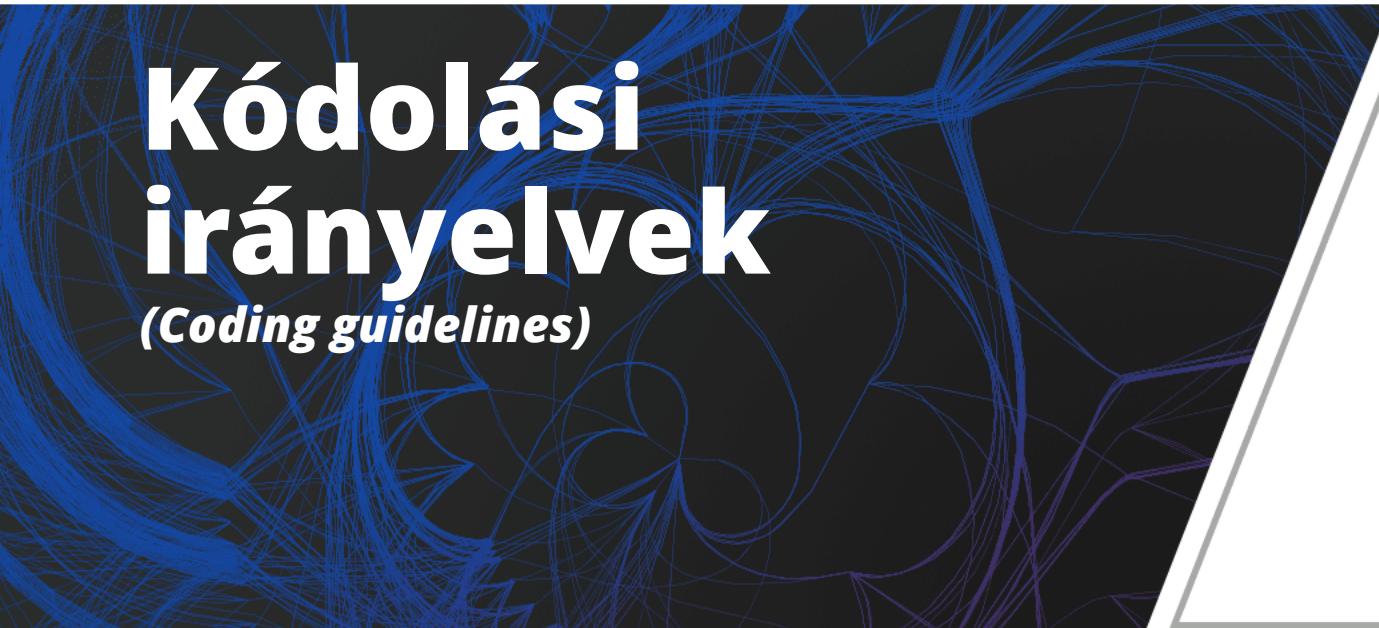
- **Statikus analízis**, tesztelés, ...

Specifikációnak megfelelő

- **Kód felülvizsgálat**, tesztelés

Kódolási irányelvek

(Coding guidelines)



Kódolási irányelvek – Bevezető

- Szabályhalmazok, amelyek ajánlásokat adnak
 - Stílus: formázás, elnevezés, struktúra
 - Programozási tanácsok: konstrukciók, architektúra
- Fő kategóriák
 - Szakterület specifikus
 - Autóipar, vasút, ...
 - Platform specifikus
 - C, C++, C#, Java, ...
 - Szervezet/cég specifikus
 - Google, CERN, ...

Szakterület specifikus: MISRA C

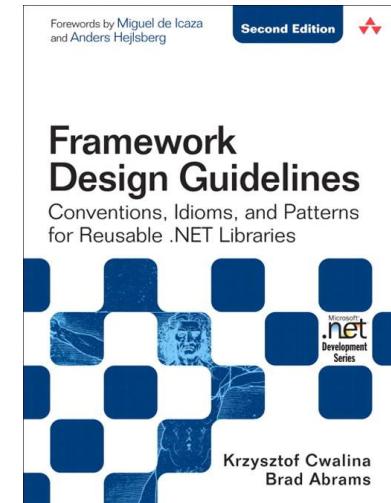
- Motor Industry Software Reliability Association
- Cél: biztonság, megbízhatóság, hordozhatóság
- 143 szabály + 16 direktíva
- Eszközök: SonarQube, Coverity, ...
- Példák
 - *RHS of && and || operators shall not contain side effects*
 - *Test against zero should be made explicit for non-Booleans*
 - *Body of if, else, while, do, for shall always be enclosed in braces*



Platform specifikus: .NET

- Framework Design Guidelines (C#)
 - Cél: keretrendszer és API fejlesztés
- Kategóriák
 - Elnevezés, típusok tervezése, tagváltozók tervezése, kiterjeszthetőség, kivételek, használhatóság, gyakori tervezési minták
 - „Do”, „Consider”, „Avoid”, „Do not”
- Eszköz: StyleCop

[https://msdn.microsoft.com/en-us/library/ms229042\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/ms229042(v=vs.110).aspx)



Platform specifikus: .NET (példák)

- **DO NOT** provide abstractions unless they are tested by developing several concrete implementations and APIs consuming the abstractions.
- **CONSIDER** making base classes abstract even if they don't contain any abstract members. This clearly communicates to the users that the class is designed solely to be inherited from.
- **DO** use the same name for constructor parameters and a property if the constructor parameters are used to simply set the property.

[https://msdn.microsoft.com/en-us/library/ms229042\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/ms229042(v=vs.110).aspx)

Szervezet specifikus: Google

- Java Style Guide
- Cél: „hard-and-fast” szabályok, tanácsok kerülése
- Kategóriák
 - Forrásfájl alapvető szabályok
 - Forrásfájl struktúrája
 - Formázás
 - Elnevezés
 - Programozási ajánlások
 - Javadoc (dokumentáció)
- Továbbiak: C++, C#, Python, JavaScript, R...

<https://google.github.io/styleguide/javaguide.html>



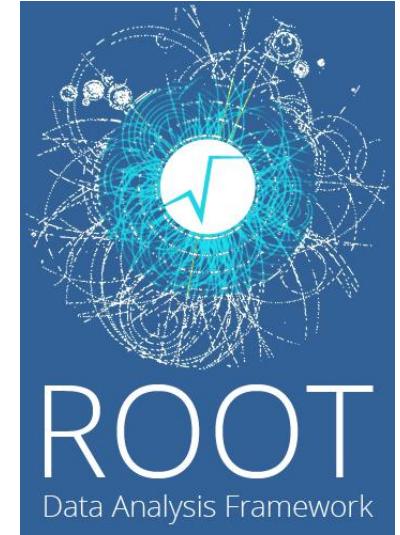
Szervezet specifikus: Google (példák)

- *Never make your code less readable simply out of fear that some programs might not handle non-ASCII characters properly. If that should happen, those programs are broken and they must be fixed.*
- *When a reference to a static class member must be qualified, it is qualified with that class's name, not with a reference or expression of that class's type.*
- *In Google Style special prefixes or suffixes, like those seen in the examples name_, mName, s_name and kName, are not used.*
- *Local variable names are written in lowerCamelCase.*

<https://google.github.io/styleguide/javaguide.html>

Szervezet specifikus: CERN

- ROOT: C++ adatelemző eszköz / keretrendszer részecskefizikához
- Kategóriák
 - Elnevezések
 - Kivételek
 - Névterek
 - Kommentezés
 - Forrásfájl struktúrája
- Eszköz: Artistic Style (astyle)



<https://root.cern/coding-conventions>

Szervezet specifikus: CERN (példák)

- *Avoid the use of raw C types like int, long, float, double when using data that might be written to disk.*
- *For naming conventions we follow the Telligent rules. Types begin with a capital letter (Boolean), base classes begin with „T” (TContainerView), members begin with „f” (fViewList), ...*
- *Each header file has the following layout: Module identification line, Author line, Copyright notice, Multiple inclusion protection macro, Headers file includes, Forward declarations, Actual class definition.*

<https://root.cern/coding-conventions>

Kódolási irányelvezek - Összefoglalás

- Hogyan lehet betartatni?
 - Számos fejlesztőkörnyezetben (IDE) alapfunkció
 - Külső eszközök
 - Fejlesztési folyamatba szorosan integrálva
- Fontos
 - Mindig legyen egy egységes, közös irányelv
 - Legalább egységes IDE formázási szabályok
 - Általában fájlba menthető, ami a verziókezelőbe feltölthető

Kódolási irányelvek - Összefoglalás

- Melyik a legjobb? Melyiket válasszuk?
- Legtöbbször ez már el van döntve
 - A szakterület, platform vagy szervezet által
 - Konzisztencia a meglévő kódbázissal
- Néha viszont eldönthető
 - Sokszor nincs egyetlen legjobb választás
 - Néha még egymással is inkonzisztensek lehetnek
 - Kombináció is lehetséges
 - De azért ne találjuk fel újra a kereket
 - Megnehezíti a belépő fejlesztők dolgát

Kód felülvizsgálat

(Code review)



Kód felülvizsgálat – Bevezető

- Manuális, emberek által végzett módszer
 - Forráskód olvasása, átnézése, elemzése
 - Általában strukturált ellenőrző lista alapján
- Különböző szintek (informális → formális)

Informális felülvizsgálat
(Informal review)

- Informális
- Többi csapattag, csapatvezető által végezve

Átvizsgálás
(Walkthrough)

- Többnyire informális
- Szerző által vezetve

Technikai felülvizsgálat
(Technical review)

- Előkészített, dokumentált folyamat
- Szakértők bevonása

Vizsgálat
(Inspection)

- Formális, előkészített, dokumentált folyamat
- Külső szakértők, moderátor bevonása

<http://www.istqb.org/downloads/syllabi/foundation-level-syllabus.html>

Kód felülvizsgálat - Folyamat

Tervezés

- Dokumentumok, résztvevők, kritériumok meghatározása
- Feladatok elosztása

Kick-off

- Folyamat ismertetése
- Forráskód eljuttatása a felülvizsgálóhoz

Előkészítés

- Forráskód átvizsgálása
- Problémák feljegyzése

Megbeszélés

- Problémák egyeztetése és rögzítése
- Javaslatok

Átdolgozás

- Javítások elvégzése
- Módosítások rögzítése

Követés

- Javítások ellenőrzése
- Kilépési feltétel ellenőrzése

Kód felülvizsgálat - Előnyök

- Formális vizsgálat
 - Hatásos a hibák megtalálásában
 - Időigényes, fáradságos munka
- Modern technikák
 - Kevésbé formális, nagy eszköztámogatás
 - Iparban elterjedt (Microsoft, Google, Facebook, ...)
 - Hibák megtalálásán kívül további előnyök
 - Tudástranszfer
 - Csapatszellem
 - Alternatív megoldások

<http://dl.acm.org/citation.cfm?id=2486882>

Kód felülvizsgálat - Ellenőrző lista

- Ellenőrző lista: szempontok strukturált felsorolása
- Kódolási irányelvekhez hasonló kategóriák
 - Kód olvashatósága, karbantarthatósága
 - Biztonság, sebezhetőség
 - Teljesítmény
 - Bevett programozási minták, gyakorlatok
- Tanácsok
 - Sokféle lista (*code review checklist*) érhető el online
 - Törekedjünk az automatizálásra
 - Pl. formázást egy eszköz is tud ellenőrizni

Kód felülvizsgálat - Eszközök

- Kód felülvizsgálat támogatása
 - Megjegyzések, párbeszédek csatolása kódrészletekhez
 - Fejlesztési folyamatba integrálva
- GitHub: pull request reviews (→ Gyakorlat)
 - Megjegyzések, elfogadás, változtatások kérése

The image consists of two side-by-side screenshots of the GitHub interface.

Left Screenshot: Shows a pull request from the user 'octocat'. The review message says: "This is looking ✨! I've left a few comments that should be addressed before this gets merged. 🐱". Below the message is a diff view of a file named 'data/reusables/open-source.yml'. The diff shows changes to the 'open-source-handbook-repositories' section, with additions like '+ For more information on open source, specifically how to create and grow an open'.

Right Screenshot: Shows the 'Review changes' modal. It displays the same review message: "This is looking ✨! I've left a few comments that should be addressed before this gets merged. 🐱". Below the message are three radio button options:

- Comment**: Submit general feedback without explicit approval.
- Approve**: Submit feedback and approve merging these changes.
- Request changes**: Submit feedback that must be addressed before merging.

A 'Submit review' button is at the bottom of the modal.

<https://help.github.com/articles/about-pull-request-reviews/>

Kód felülvizsgálat - Eszközök

- Gerrit
 - Web-alapú kód felülvizsgálat
 - Git támogatás
 - Munkafolyamat menedzselése

The screenshot shows a code review interface with a green header bar containing icons for file operations. Below is a code diff with line numbers 110 to 122. A yellow box highlights a comment from Stefan Beller and Dave Borowitz:

```
110 private PatchSet patchSet;
111 private ChangeMessage changeMessage;
112 private SshInfo sshInfo;
113 private ValidatePolicy validatePolicy = ValidatePolicy.GERRIT;
114 private boolean draft;
115 private boolean runHooks = true;
```

Stefan Beller Why do you move this out of the constructor? Initially I assumed this... Jan 28 2:55 PM

Dave Borowitz Because it would be identical between the two constructors, so it sa... Jan 28 3:19 PM

```
116 private boolean sendMail = true;
117 private Account.Id uploader;
118 private BatchRefUpdate batchRefUpdate;
119
120 @AssistedInject
121 public PatchSetInserter(ChangeHooks hooks,
122                         ReviewDb db,
```

<https://www.gerritcodereview.com/>

Statikus analízis

(Static analysis)



Statikus analízis – Példa

```
1 public class Sample {  
2     public static void main(String[] args) {  
3         String str = null;  
4         try {  
5             Scanner scanner = new Scanner("file.txt");  
6             str = scanner.nextLine();  
7             scanner.close();  
8         } catch (Exception e) {  
9             System.out.println("Error opening file!");  
10        }  
11        str.replace(" ", "");  
12        System.out.println(str);  
13    }  
14 }
```

Kivétel esetén a
scanner nincs lezárva

str lehet null

str „immutable”

Statikus analízis – Bevezető

- Definíció: program analízise végrehajtás nélkül
 - Általában automatizált eszközökkel
 - De a kézi átvizsgálást is beleérhetjük
- Mintázatok alapján
 - Nagyrészt egyszerű statikus tulajdonságok hibaminták alapján
 - Pl.: nem használt változó, figyelmen kívül hagyott visszatérési érték
 - Eszközök: SpotBugs, ErrorProne, SonarQube, Coverity
- Interpretáció alapján (→ MSc)
 - Dinamikus tulajdonságok
 - Pl.: null pointer hivatkozása, túlindexelés
 - Eszközök: Infer, PolySpace

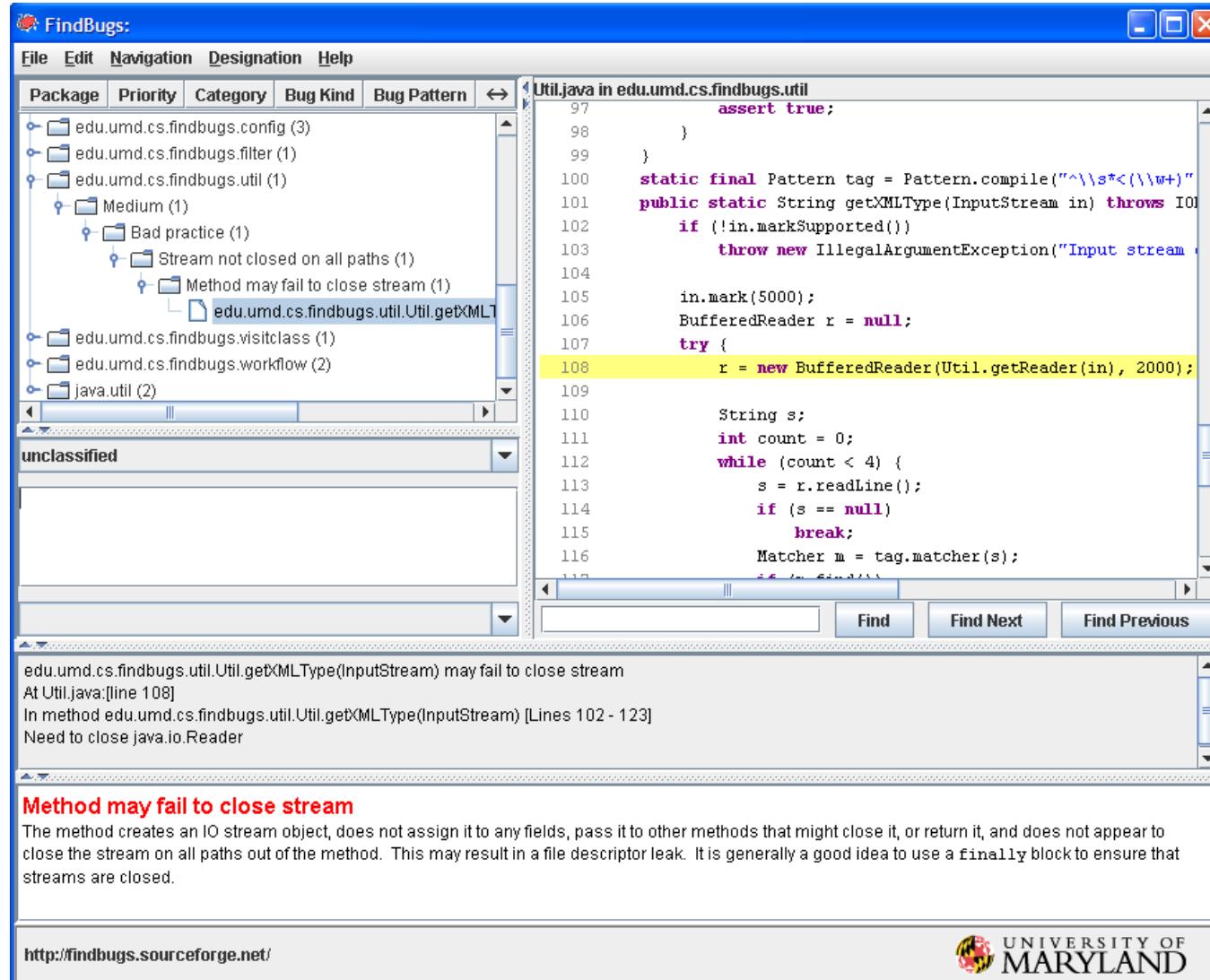
SpotBugs (Java)



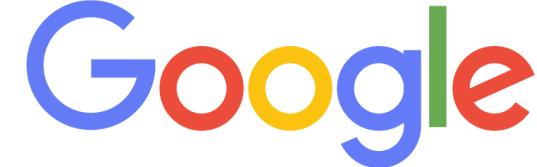
- Nagy, kiterjeszthető szabályhalmaz
- Konzolos/GUI alkalmazás, Eclipse/IntelliJ plug-in
- Példák
 - Rossz szokás: „*random object created and used only once*”
 - Helyesség: „*bitwise add of signed byte value*”
 - Sebezhetőség: „*expose inner static state by storing mutable object into a static field*”
 - Többszálúság: „*synchronization on Boolean could lead to deadlock*”
 - Teljesítmény: „*invoke `toString()` on a string*”
 - Biztonság: „*hardcoded constant database password*”
 - Gyanús: „*useless assignment in return statement*”

<https://spotbugs.github.io/>

SpotBugs (Java)



ErrorProne (Java)



- Google belső fejlesztés
 - Kiegészíthető szabályhalmaz
 - Gradle, Maven, Eclipse, IntelliJ, ...
- Példák
 - „Reference equality used to compare arrays”
 - „Type declaration annotated with @Immutable is not immutable”
 - „Loop condition is never modified in loop body.”
 - „This conditional expression may evaluate to null, which will result in an NPE when the result is unboxed.”
 - „Comparison of a size ≥ 0 is always true, did you intend to check for non-emptiness?”

<https://errorprone.info/>

SonarQube

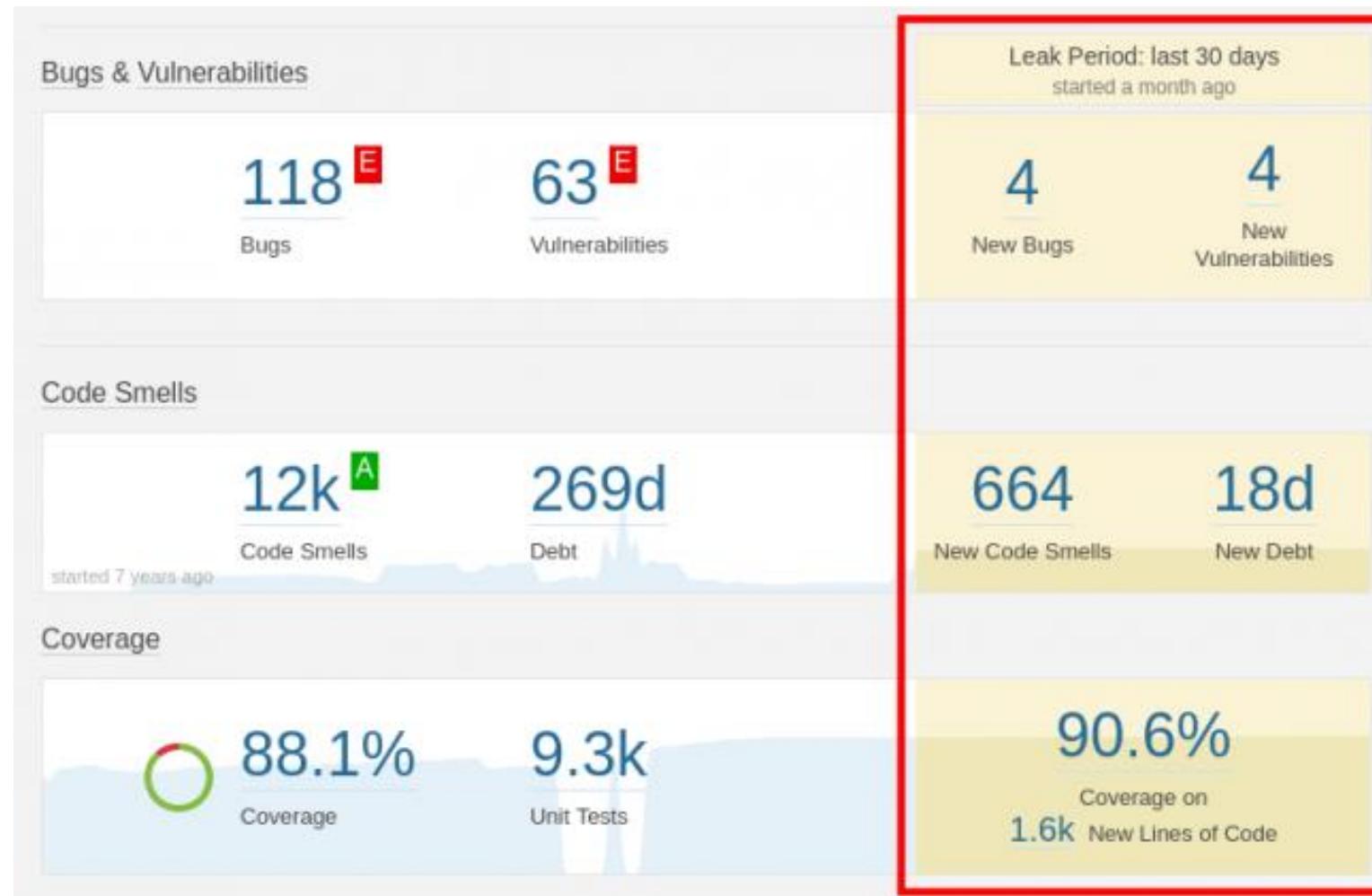


- Kódminőség menedzsment platform
- 20+ nyelv (Java, JS, Kotlin, C, C++, C#, Python, ...)
- Képességek
 - Kódolási irányelvek vizsgálata, kódduplikálás, teszt fedettség, kód komplexitás, lehetséges hibák és sebezhetőségek, költség becslés
 - Jelentések, diagramok generálása
 - Külső eszközökkel integrálható
 - Pl.: fejlesztőkörnyezet, continuous integration (CI) eszközök

(→ Gyakorlat)

<http://www.sonarqube.org/>

SonarQube



SonarQube



SonarQube

Issues Measures Code Dashboards ▾

Issues Effort

Type

Type	Count
Bug	118
Vulnerability	63
Code Smell	12k

Resolution

Resolution	Count	Fixed	Count
Unresolved	118	Fixed	4
False Positive	0	Won't fix	0
Removed	41		

Severity

Status

SonarQube :: Plugin API src/main/java/com/sonar/qube/plugin/api/IssueType.java

Override this superclass' "equals" method. ...

Bug ⚠ Major ⚡ Open Not assigned 30min effort

SonarQube :: Plugin API src/main/java/com/sonar/qube/plugin/api/IssueResolution.java

The return value of "parseDouble" must be used. ...

Bug ⚠ Critical ⚡ Open Not assigned 10min effort

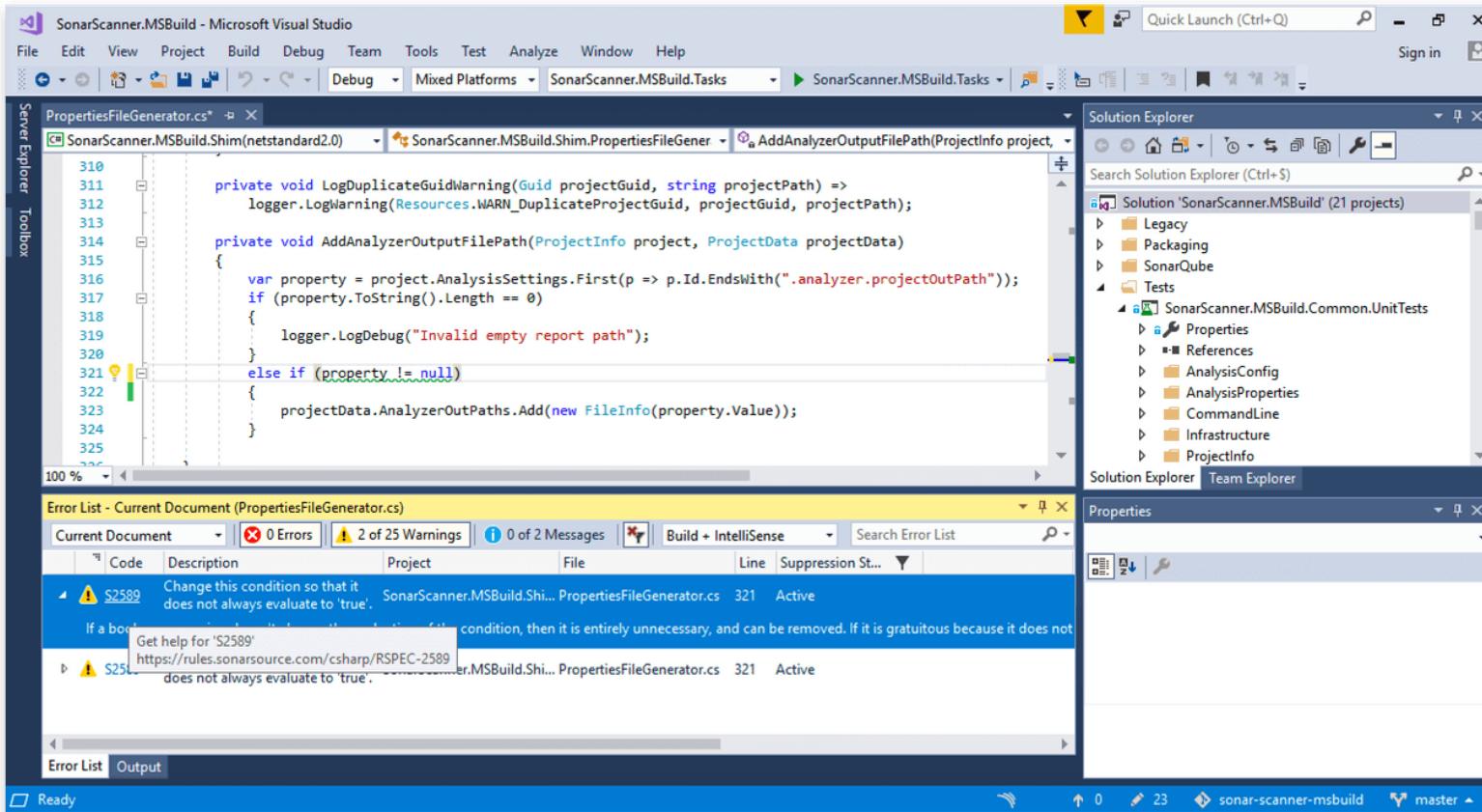
SonarQube :: Plugin API src/main/java/com/sonar/qube/plugin/api/IssueSeverity.java

NullPointerException might be thrown as 'value' is nullable in ...

Bug ⚠ Blocker ⚡ Open 🌟 Simon Brandhof 10min effc

SonarLint

- Sonar plug-in fejlesztőkörnyezetekhez
 - VS Code, Visual Studio, Eclipse, IntelliJ



<https://www.sonarlint.org/>

Coverity

- Synopsys suite statikus analízis eszköze
- Nyelvek: C, C++, C#, Java, JavaScript
- Felhasználók: CERN, NASA, ...
- Példák: erőforrás szivárgás, null pointer, inicializálatlan adat, konkurencia problémák, ...
- Coverity Scan: ingyenes szolgáltatás nyílt forráskódú projektekhez
 - GitHub és Travis CI integráció



<http://www.synopsys.com/software/coverity/Pages/default.aspx>

<https://scan.coverity.com/>

Statikus analízis hatékony használata

- Legyen integrálva a build folyamatba
 - Ellenőrzés a commit előtt/után
 - Jelentések generálása, e-mail értesítés, ...
- Használjuk a projekt kezdetétől fogva
 - Túl sok probléma elriasztja a fejlesztőket
- Konfiguráljuk az eszközöket
 - Szűrés kategóriák és súlyosság szerint
 - Kiegészítés saját szabályokkal

Statikus analízis hatékony használata

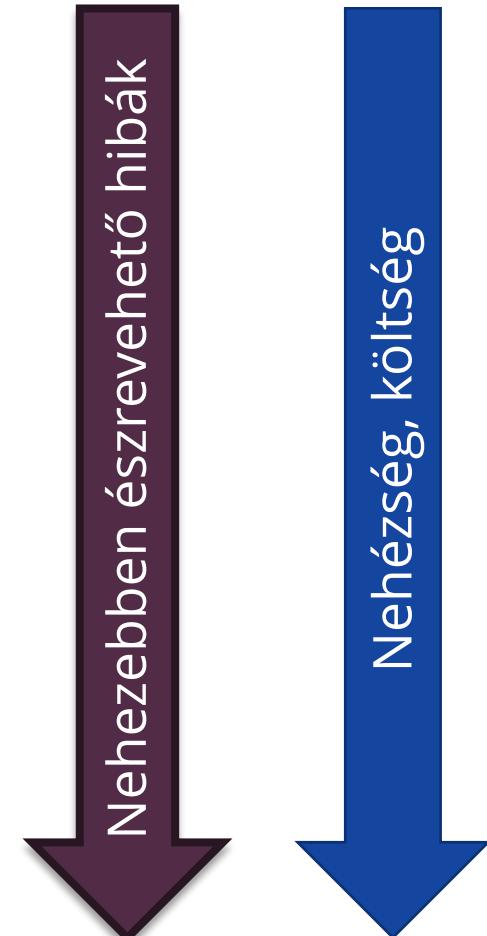
- Az eredményeket óvatosan kezeljük
 - Hamis pozitív és negatív előfordulhat
- Hamis negatív (*false negative*)
 - Ha nem találunk hibát, az nem jelent helyes szoftvert
- Hamis pozitív (*false positive*)
 - Egy megtalált hiba nem biztos, hogy valódi hatást okoz
 - Teljes szabály vagy egy előfordulás elnyomása
 - Mindig legyen megindokolva

Statikus analízis – Összefoglalás

- Szoftver analízise végrehajtás nélkül
 - Analízis lehetséges, mielőtt még végrehajtható lenne, vagy input állna rendelkezésre
 - Végrehajtás költséges lehet
- Nehezen észrevehető hibák megtalálása
 - Tapasztalt programozók számára is érdekes lehet
- Automatizált folyamat
 - Fejlesztési folyamatba integrálva

Statikus ellenőrzési technikák - Összefoglalás

- Kódolási irányelvezek
 - Szakterület, platform, szervezet specifikus
- Kód átvizsgálás
 - „Kézi” átvizsgálás ellenőrző lista alapján
- Statikus analízis eszközök
 - Szoftver analízis végrehajtás nélkül



Ellenőrzési technikák a szoftverfejlesztés során

Dr. Micskei Zoltán

<https://mit.bme.hu/~micskeiz>



Méréstechnika és
Információs Rendszerek
Tanszék



Critical Systems
Research Group

Célkitűzés

Hogyan fejlesszünk jó minőségű szoftvert?

- Előadások: ellenőrzési technikák áttekintése
 - CI, kód átvizsgálás, statikus analízis, tesztelés...
- Gyakorlatok: technológiák kipróbálása

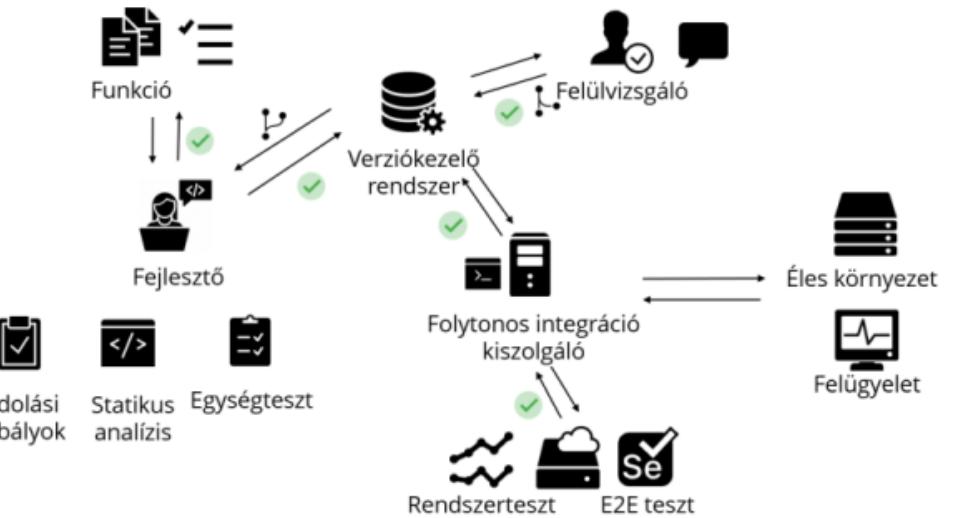


- Házi feladat: tanultak alkalmazása és elmélyítése
 - Egy nyílt forráskódú projekten

Házi feladat

- 4 fős csapatok (jelentkezési űrlap később)
- Vizsgalandó projekt:
 - megadott listából VAGY
 - saját javaslat (-> email micskei AT mit.bme.hu)
Feltétel: GitHub publikus projekt
- Feladat: technológia és termék fókuszú ellenőrzések
- Értékelés:
 - Bemutatás a félév végén
 - Nem mennyiségi, hanem minőségi (!)
 - Jegybe beszámít (pótlás a pótlási héten)

Folyamatos ellenőrzés



Continuous Integration (CI)

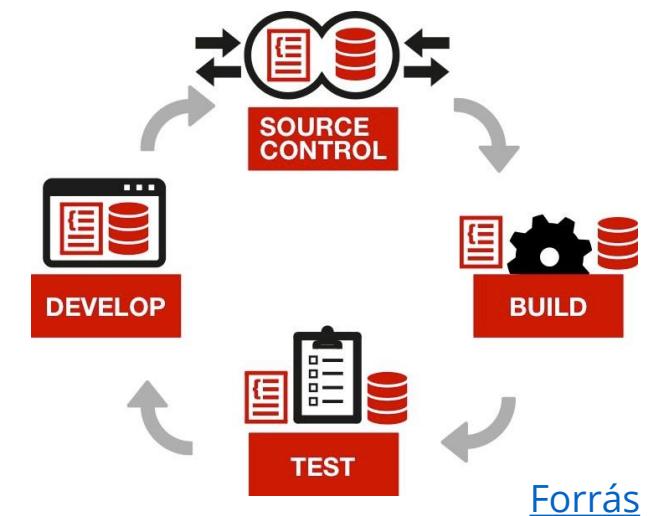
Folytonos integráció

- „*a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily*”
- „*Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible.*”



Martin Fowler

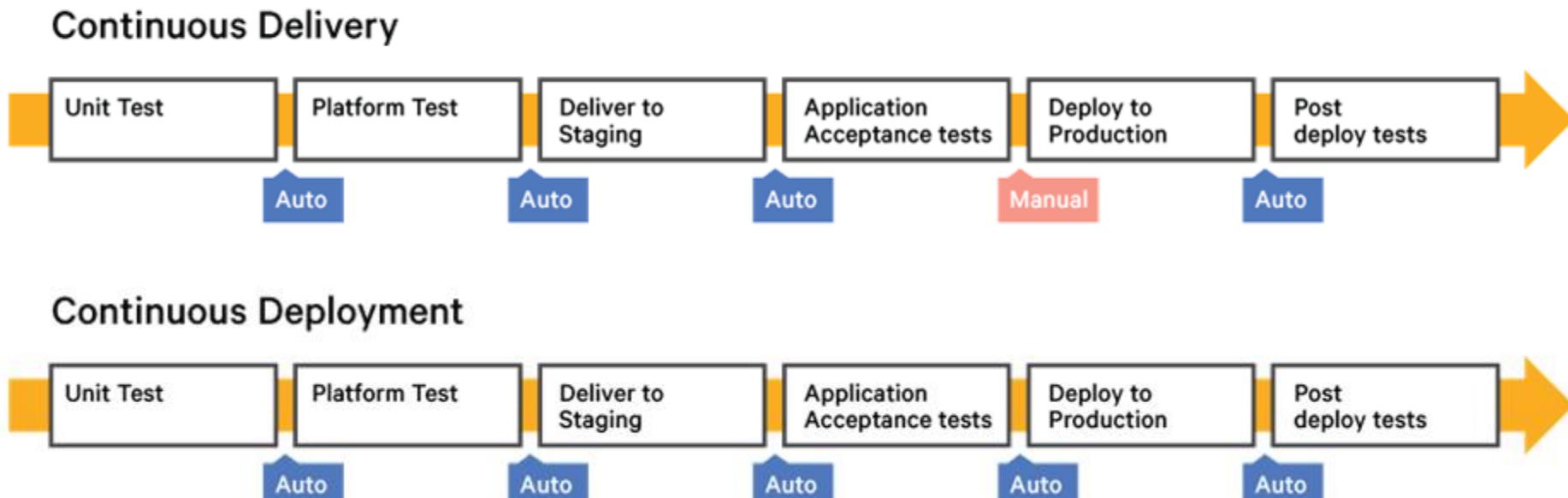
<https://martinfowler.com/articles/continuousIntegration.html>



Continuous Delivery (CD)

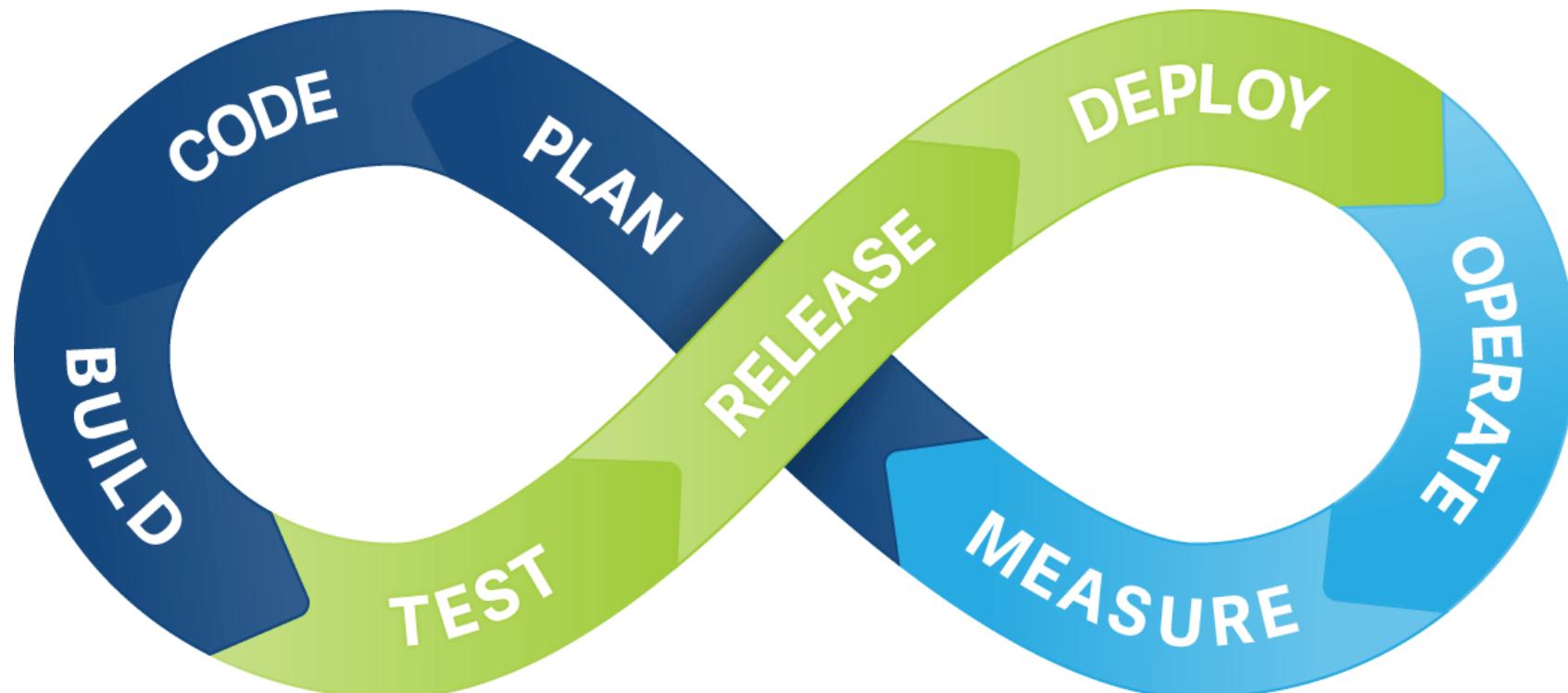
„build software so that it is always in a state where it could be put into production”

Forrás: <https://martinfowler.com/bliki/ContinuousDelivery.html>

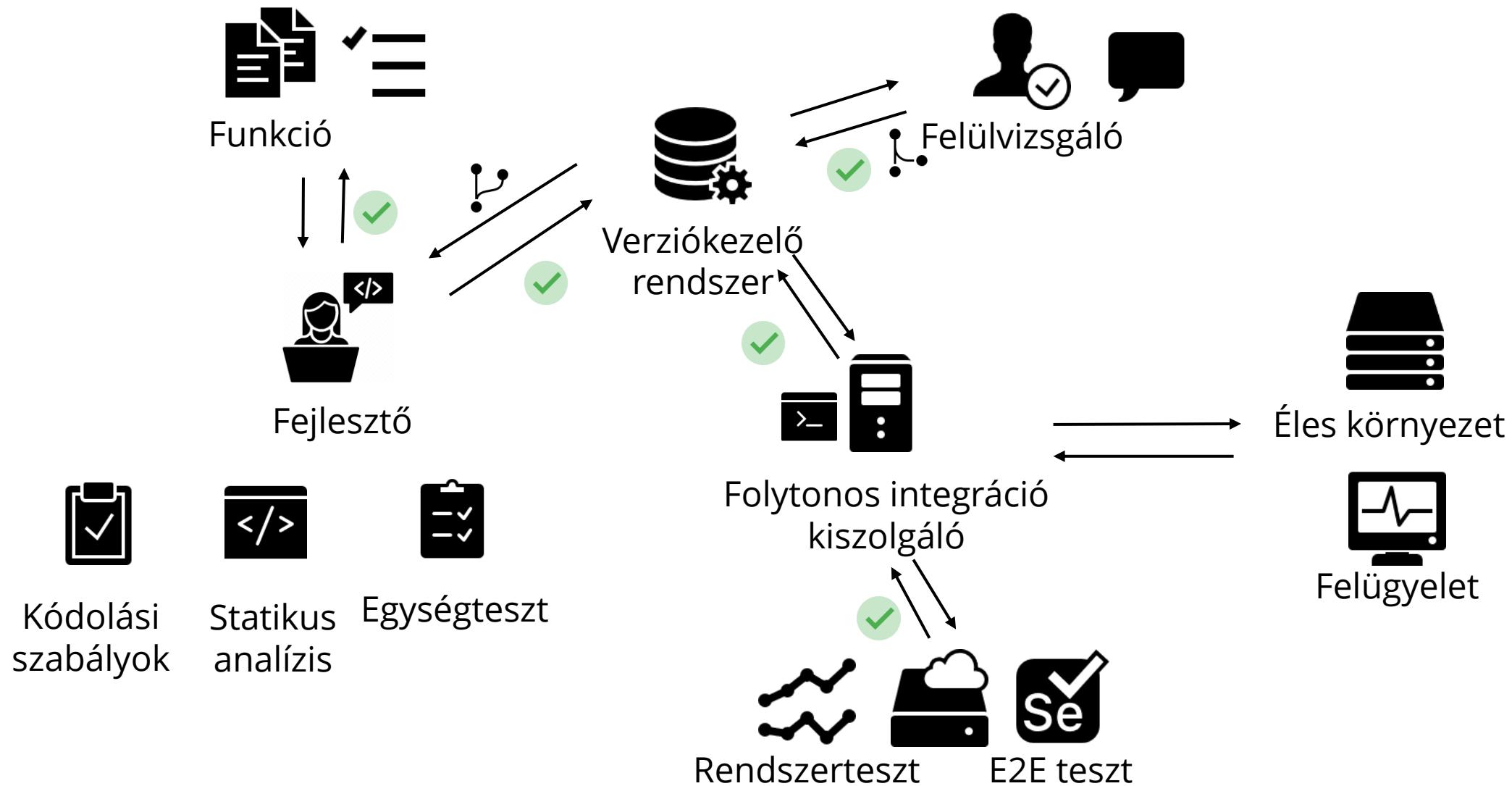


Forrás

DevOps = Development + Operations



Folyamatos ellenőrzés és visszajelzés



Ikonok: icons8.com

Definition of Done



Kész van a funkció?

Persze, 95%-ban már kész!
Működik, már csak ... kell.

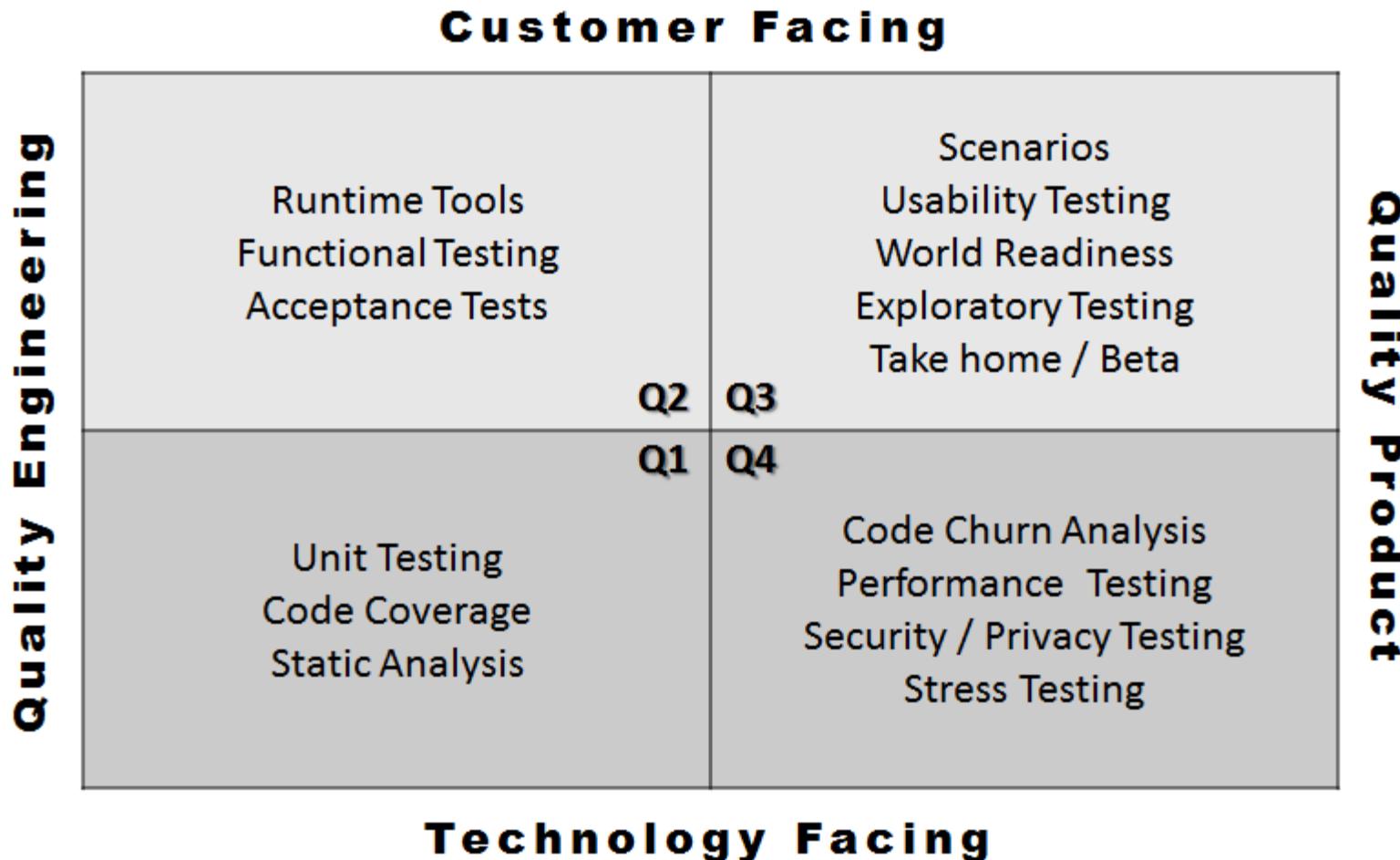


Definition of Done

- Ellenőrző lista, hogy mikor van valami kész
- Kód, tesztek, teszt futtatás, próba telepítés...
- Csapattól függ a tartalma

Lásd: <https://www.agilealliance.org/glossary/definition-of-done>

Agilis tesztelési kvadránsok



Forrás: <http://angryweasel.com/blog/riffing-on-the-quadrants/>
(sok egyéb változata létezik még, pl. [ez](#))

Kitekintés



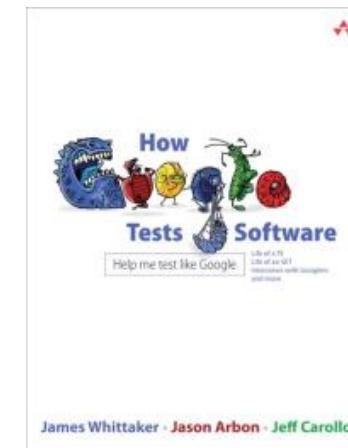
Testing @ Google



„The burden of quality is on the shoulders of those writing the code.”

Szerepek

- Software Engineer in Test & Infrastructure ([SETI](#))
- Test Engineer (TE)



James A. Whittaker, Jason Arbon, Jeff Carollo. How Google Tests Software. Addison-Wesley Professional, 2012

Circle CI karrierutak

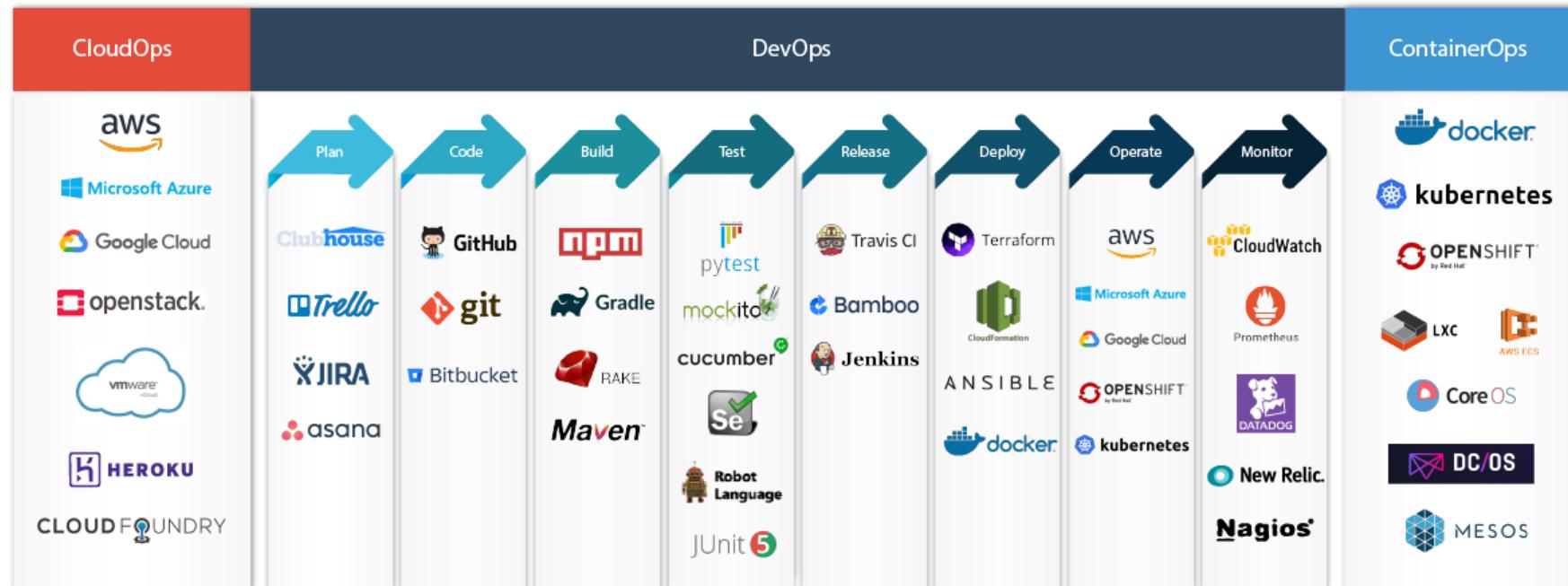
Szükséges kompetenciák az egyes szinteken (részlet)

		E1	E2	E3
		Title	Associate Engineer	Engineer
		Focus	execution of work	
Technical skills	Quality & testing	Writing code	Writes code with testability, readability, edge cases, and errors in mind.	Consistently writes functions that are easily testable, easily understood by other developers, and accounts for edge cases and errors. Uses docstrings effectively.
		Testing	Knows the testing pyramid. Writes unit tests, sometimes with help from more senior engineers.	Understands the testing pyramid, writes unit tests in accordance with it, as well as higher level tests with help from more senior engineers. Always tests expected edge cases and errors as well as the happy path.

<https://circleci.com/blog/how-engineering-managers-can-effectively-support-engineers-teams-and-organizations/>

DevOps választható tárgy (ősszel)

DevOps: A rendszerfejlesztés és üzemeltetés kapcsolódása (VIMIAV21)



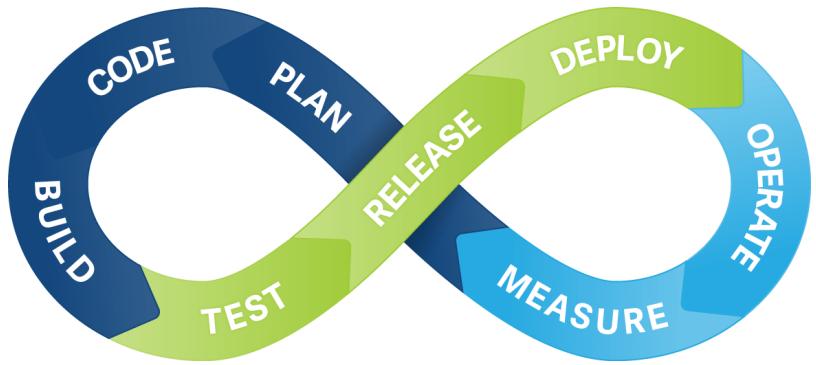
<https://inf.mit.bme.hu/edu/courses/devops>

Hasznos segédletek (érdemesset letölteni!)

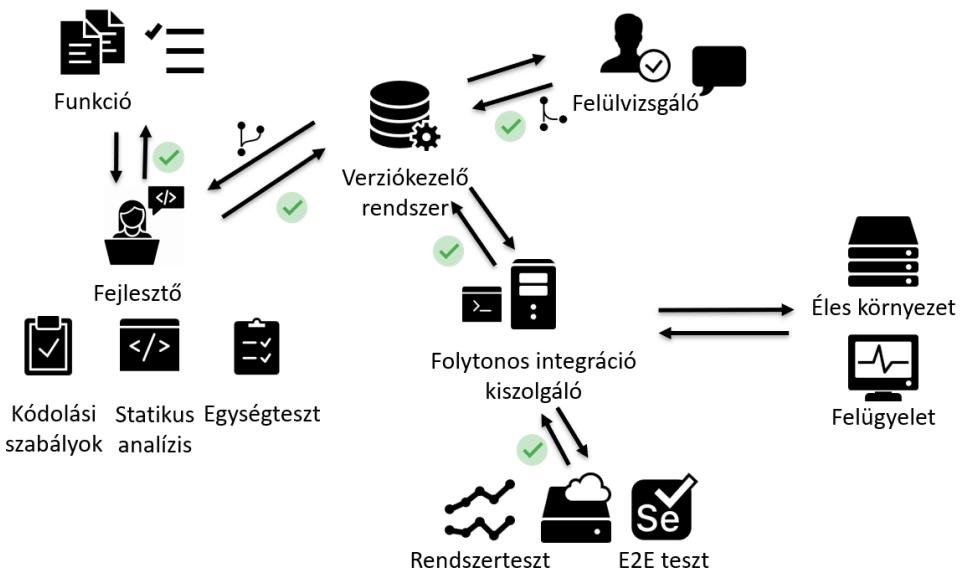
- IEEE szabványok
 - [24765-2010](#) Systems and SW engineering – Vocabulary
 - [SE VOCAB](#) – online kereshető formátum
 - 29119 Software testing
 - Part 1 Concepts and definitions, Part 2 Test processes, Part 3 Test documentation
- International Software Testing Qualifications Board (ISTQB)
 - [Foundation Level Syllabus](#) (2018)
 - [Glossary of Testing Terms](#)
- Hungarian Testing Board (HTB)
 - [Glossary](#) / Kifejezésgyűjtemény (magyar fordítás)



Összefoglalás



Customer Facing		Quality Product
Runtime Tools Functional Testing Acceptance Tests	Scenarios Usability Testing World Readiness Exploratory Testing Take home / Beta	
Q2	Q3	
Q1	Q4	
Unit Testing Code Coverage Static Analysis		Code Churn Analysis Performance Testing Security / Privacy Testing Stress Testing
Technology Facing		



Virtuális információ integrációs technikák

6. Előadás
Integrációs és ellenőrzési technikák (VIMIACo4)

Méréstechnika és Információs Rendszerek Tanszék
2021.

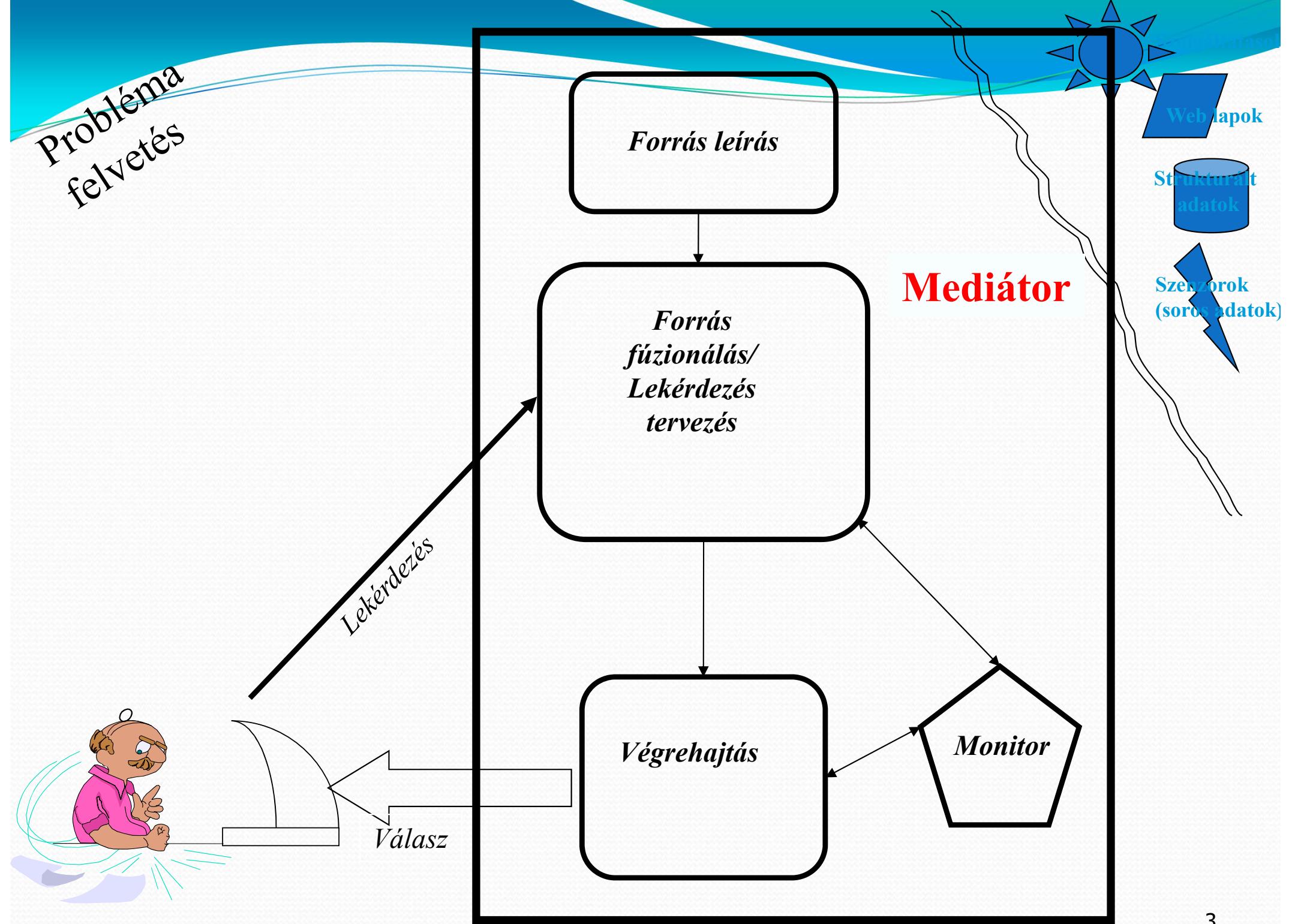
Integrációs és ellenőrzési technikák (VIMIACo4)

Távoktatás

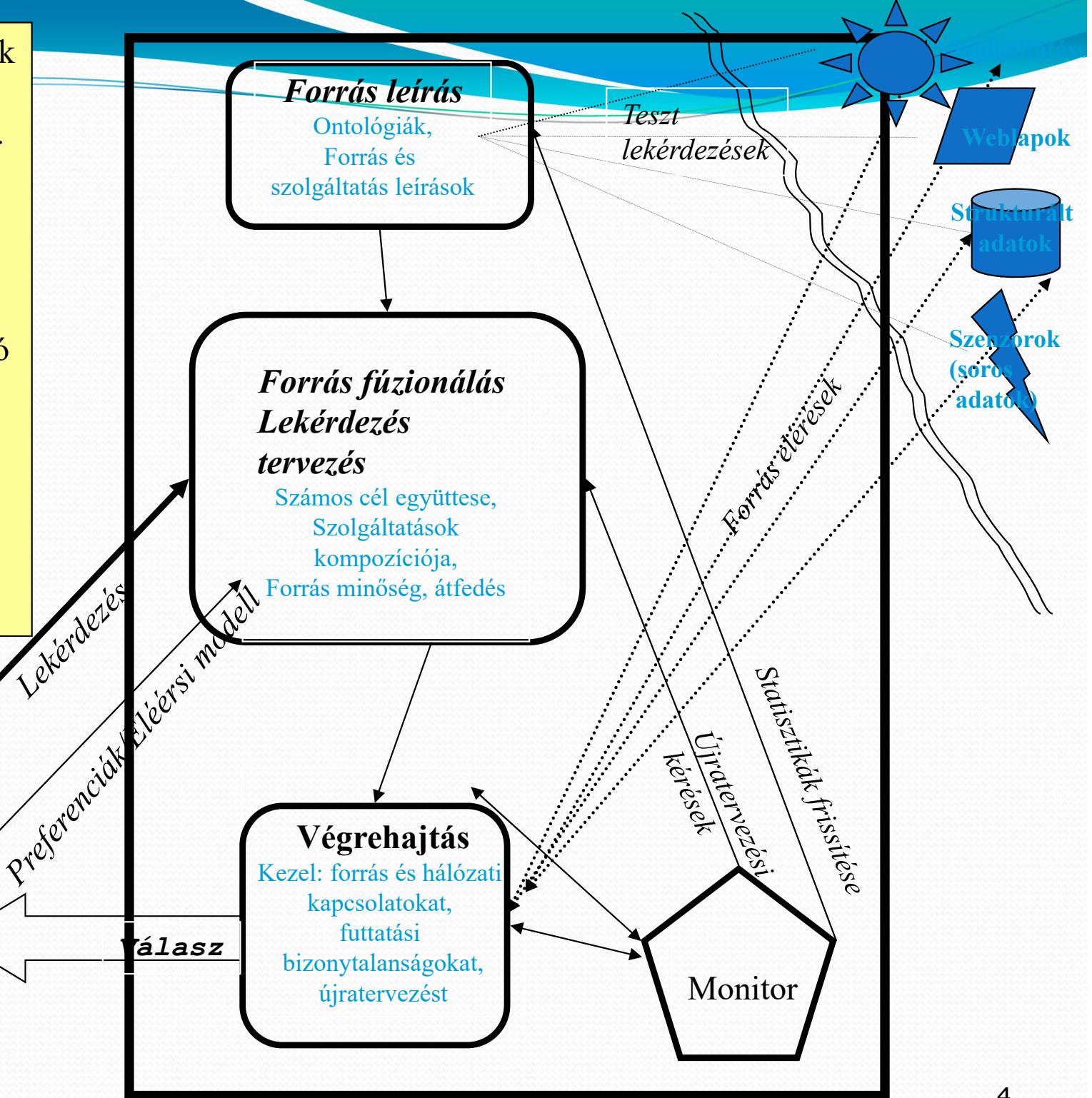
- *A kurzus keretében két házi feladatot kell megoldani*
- *A félév végén egy írásbeli (online vagy jelenléti) vizsga*
- *Mind a házi feladatokat, mind a vizsgát elégséges szinten (40%) teljesíteni szükséges.*
- *A félérvégi érdemjegy számítása:*
házi feladatokra kapott pontok (10-10%)
vizsgára kapott pontok (80%)
(elégtelen: 0-39%, elégséges: 40-49%, közepes: 50-64%, jó: 65-79%, jeles: 80-100%)
- *Gyakorlatok: részvétel szükséges 5 gyakorlaton*
(részvétel ellenőrzése: leadott jegyzőkönyv)
- *IMSC pontok az első házi feladatban kiadott választható feladat megoldásával szerezhető.*

Méréstechnika és Információs Rendszerek Tanszék
2021.

Probléma felvetés

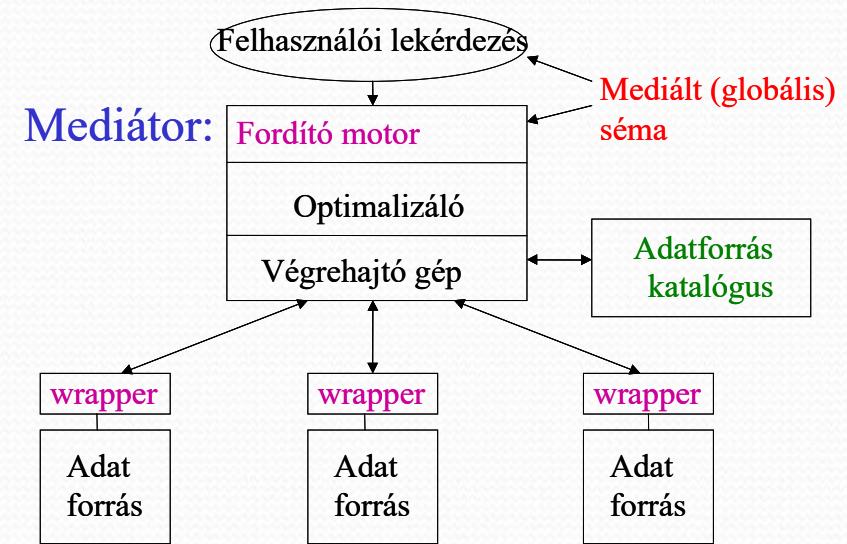
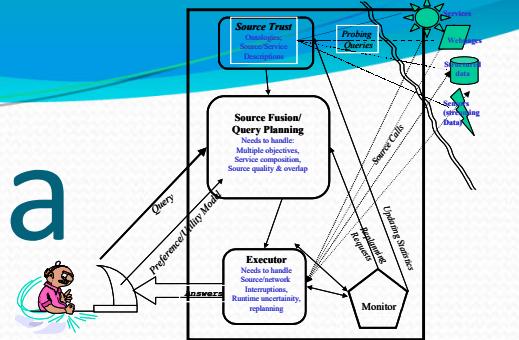


- Felhasználói lekérdezések megfogalmazása a mediált (*globális*) sémán.
- Adatok tárolva *lokális (távoli) sémákban*.
- A tárolt információ (tartalom) ismerete alapján megfogalmazható a leképezés a sémák között.
- A mediátor alkalmazza a leképezést a felhasználói kérdés lefordítására a forrás lekérdezésekre.



Virtuális integrációs séma

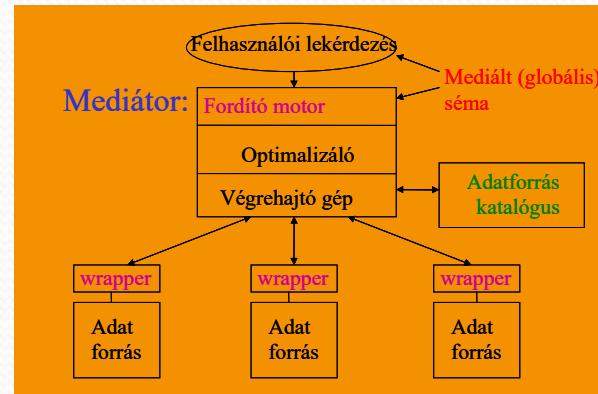
- Adatok a forrásokban maradnak
- Lekérdezés végrehajtásakor:
 - Releváns források meghatározása
 - Lekérdezés szétválasztása forrásokra vonatkozó lekérdezésekre.
 - Válaszok begyűjtése a forrásokból, és megfelelő kombinálása a válasz előállításához.
- Friss adatok
- A megoldás skálázható



Garlic [IBM], Hermes[UMD]; Tsimmis, InfoMaster[Stanford]; DISCO[INRIA]; Information Manifold [AT&T]; SIMS/Ariadne[USC]; Emerac/Havasu[ASU]

Forrás-mediátor relációs sémával szembeni elvárások

- **Kifejező erő:** hasonló adattartalommal rendelkező források megkülönböztetése, irreleváns források felismerése.
- **Egyszerű bővíthetőség:** tegyük könnyűvé források hozzáadását.
- **Fordítás/átalakítás:** felhasználói lekérdezés lefordítása forrásokon értelmezett lekérdezésekre hatékonyan és eredményesen.
- **Veszteségmentesség:** minden lehetséges adatelérés biztosítása



Lekérdezés átalakítás

- **Adott:**
 - Egy Q lekérdezés a mediátor sémára vonatkozóan
 - Adat források leírása
- **Létrehozandó:**
 - Egy Q' lekérdezés az adat forrásokra vonatkozóan, amely:
 - Q' csak *helyes válaszokat* ad a Q lekérdezéshez és
 - Q' minden lehetséges választ megtalál Q-hoz az elérhető forrásokból.

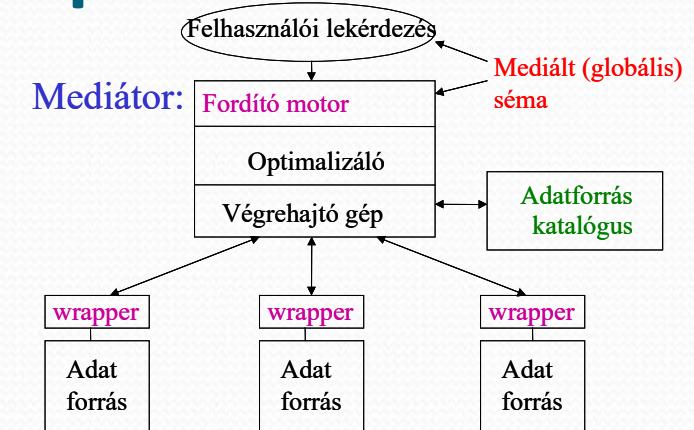
Fordítási/átfogalmazási probléma

- **Adott:**

- Egy Q lekérdezés a mediátor sémára vonatkozóan
- Adat források leírása

- **Létrehozandó:**

- Egy Q' lekérdezés az adat forrásokra vonatkozóan, amely:
 - Q' csak *helyes válaszokat* ad a Q lekérdezéshez és
 - Q' minden lehetséges választ megtalál Q-hoz az elérhető forrásokból.



Forrás és felhasználói sémák reláció leírásának megközelítései

- Globális mediált sémák(*Global-as-view, GAV*):
 - a mediált séma kifejezése a forrásokra vonatkozó nézetek relációjaként
- Lokális mediált sémák (*Local-as-view, LAV*):
 - forrás relációk kifejezése a mediált sémákon értelmezett relációkkal

GAV

VS.

LAV

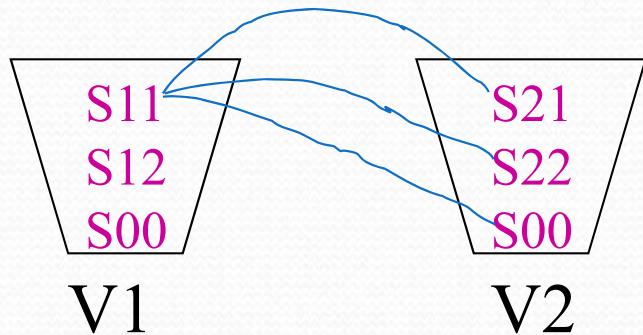
- Nem moduláris
 - Források hozzáadása módosítja a meglévő mediált séma definícióját
- Nehézkes lehet veszteségmentes mediátort készíteni.
- Lekérdezés átalakítás egyszerű
 - Nézetek kibontását jelenti (*polinomiális*)
 - Hierarchikus mediátor sémák létrehozása lehetséges
- Hatékony, ha
 - Kis számú, ritkán változó adatforrás van
 - Feladat teljesen ismert a mediátor tervezésekor (pl. vállalati adatintegráció)
 - Garlic, TSIMMIS, HERMES

- Moduláris—új forrás hozzáadása egyszerű
- Igen rugalmas – a lekérdező nyelv közvetlenül alkalmazható a források leírására
- Lekérdezés átalakítás bonyolult
 - Válaszokat a nézeteken keresztül kell előállítani (nem minden megoldható)
- Hatékony, ha
 - Sok, kevéssé korrelált forrás
 - Források dinamikus hozzáadása és törlése
 - **Information Manifold, InfoMaster, Emerac, Havasu**

Átalakítási algoritmusok

$Q(.) :- V1() \& V2()$

Veder algoritmus



- Veder algoritmus
 - Vedrek kombinációjából előállított terv
 - Utána tartalmazási ellenőrzés

$S11() :- V1()$ $S12() :- V1()$
 $S21() :- V2()$ $S22() :- V2()$
 $S00() :- V1(), V2()$

Inverz szabályok

$Q(.) :- V1() \& V2()$
 $V1() :- S11()$
 $V1() :- S12()$
 $V1() :- S00()$
 $V2() :- S21()$
 $V2() :- S22()$
 $V2() :- S00()$

- Inverz szabályok
 - Rész lekérdezések tervezés

[Levy]

P_1 contains P_2 if
 $P_2 \sqsubseteq P_1$

[Duschka]

Kitérő: Datalog

Deduktív adatbázisok

- **Alap gondolat:** relációk egy része az adatbázisban tárolt (extensional), más része a logikában (intensional).
- Relációkat predikátumokkal írjuk le.
- Relációk közti összefüggéseket datalog szabályokkal írjuk le
 - (Horn klózok, függvényszimbólumok nélkül)
 - Lekérdezések megfelelnek egy datalog programnak

Datalog

- Csak egy másik lekérdező nyelv
- Tiszta elméleti alapok
- Közelebb a logikai megközelítéshez
- Könnyen elemezhető
- Relációs algebrában nem támogatott elemek (rekurzió).
- Nincs group by, order, bags, aggregate.

Predikátumok és atomok

- Relációkat predikátumokkal reprezentálunk
- n-eseket atomokkal írunk le.

Purchase(“joe”, “bob”, “Nike Town”, “Nike Air”)

- Aritmetikai feltételek atomokkal:

$$X < 100, \quad X+Y+5 > Z/2$$

- Negált atomok:

NOT Product(“Brooklyn Bridge”, \$100, “Microsoft”)

Datalog szabályok és lekérdezések

Datalog szabályok alakja:

head :- atom1, atom2,, atom,...

Példa:

PerformingComp(name) :- Company(name,sp,c), sp > \$50

AmericanProduct(prod) :-

Product(prod,pr,cat,mak), Company(mak, sp, "USA")

Minden a lekérdezés fejében szereplő változónak meg kell jelennie a lekérdezés törzsében egy relációban.

Egy szabály egy select-from-where lekérdezést ír le.

Datalog szabályok értelmezése

AmericanProduct(prod) :-

 Product(prod,pr,cat,mak), Company(mak, sp, "USA")

Minden a lekérdezés törzsében szereplő hozzárendelést az adatbázisban található elemeknek feleltetünk meg.

Ha egy a törzsben szereplő atomnak van megfelelő elem az adatbázisban

Akkor

a fejben megjelenő n-es az eredmény.

Példa

```
CREATE VIEW Seattle-view AS  
    SELECT buyer, seller, product, store  
    FROM Person, Purchase  
    WHERE Person.city = "Seattle" AND  
          Person.per-name =  
          Purchase.buyer
```

SeattleView(buyer,seller,product,store) :-

Person(buyer, "Seattle", phone),
Purchase(buyer, seller, product, store).

Példa (negáció, unió)

SeattleView(buyer,seller,product,store) :-

Person(buyer, "Seattle", phone),

Purchase(buyer, seller, product, store)

not Purchase(buyer, seller, product, "The Bon")

Q5(buyer) :- Purchase(buyer, "Joe", prod, store)

Q5(buyer) :- Purchase(buyer, seller, store, prod),

Product(prod, price, cat, maker)

Company(maker, sp, country),

sp > 50.

Biztonságos szabályok

Minden a fej részben található változónak meg kell jelenni egy nem negált relációban a szabály törzsében .

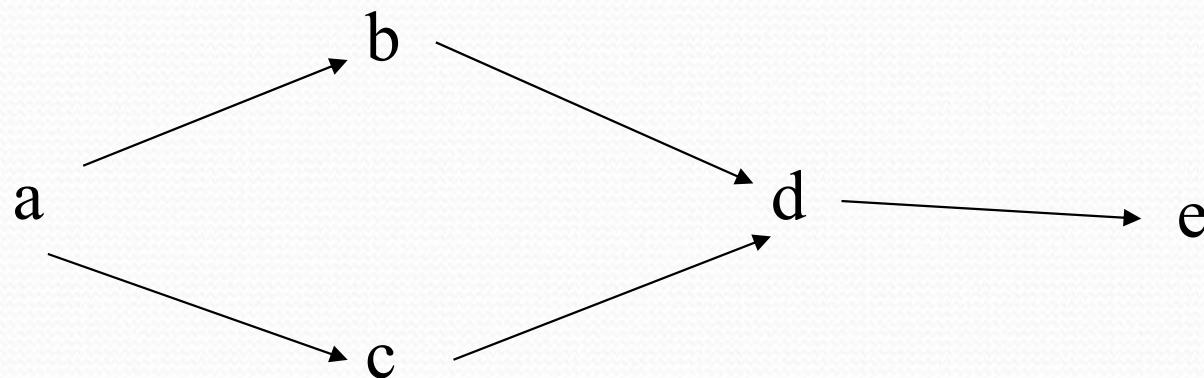
$Q(X,Y,Z) :- R1(X,Y) \& X < Z$ *nem biztonságos*

$Q(X,Y,Z) :- R1(X,Y) \And \text{NOT } R2(X,Y,Z)$

Tranzitív lezárás

Reprezentálunk egy gráfot éleivel: $Edge(X, Y)$:

$Edge(a,b), Edge(a,c), Edge(b,d), Edge(c,d), Edge(d,e)$



Írunk lekérdezést:

Keressük az a-ból elérhető csúcsokat.

Datalog megoldás

Rekurzív lekérdezés:

$Path(X, Y) \ :- \ Edge(X, Y)$

$Path(X, Y) \ :- \ Path(X, Z), \ Path(Z, Y).$

.

Rekurzív lekérdezés kiértékelése

$Path(X, Y) :- Edge(X, Y)$

$Path(X, Y) :- Path(X, Z), Path(Z, Y).$

Szemantika: kiértékelni a szabályokat egy fix pontig:

Iteration #0: Edge: $\{(a,b), (a,c), (b,d), (c,d), (d,e)\}$

Path: $\{\}$

Iteration #1: Path: $\{(a,b), (a,c), (b,d), (c,d), (d,e)\}$

Iteration #2: Path kiegészíthető új kettesekkel:

$(a,d), (b,e), (c,e)$

Iteration #3: Path további kettessel egészíthető ki:

(a,e)

Iteration #4: Nincs változás-> Leállás.

Iterációk száma függ az adatoktól!

Lokális mediált nézetek átalakítása

- Adott nézetek egy halmaza V_1, \dots, V_n , és egy Q lekérdezés.
Megválaszolható-e a Q lekérdezés a V_1, \dots, V_n nézetek felhasználásával?
 - A lekérdezéseket materializált nézeteken futtatjuk végül!
- Megközelítések
 - Veder algoritmus (Bucket algorithm [Levy; 96])
 - Inverz szabályok algoritmusa [Duschka, 99]
 - Hibrid algoritmusok
 - SV-Bucket [2001], MiniCon [2001]

Maximális tartalmazás

- Lekérdezési tervnek helyesnek és teljesnek kell lennie
 - Helyes akkor, ha az új tervet tartalmazza az eredeti lekérdezés
(Például minden n-es válasz az eredeti lekérdezésben is megtalálható)
 - Teljesség?
 - Eredeti adatbázis megközelítés teljességre törekszik
 - Itt a megközelítés: maximális tartalmazás!

P **tartalmazza** Q if $P \vdash Q$
(exponenciális algoritmusok,
még konjunktív lekérdezésekre is.)

Tartalmazás (lekérdezések)

- Legyen $Q_1(\cdot) :- B_1(\cdot)$ $Q_2(\cdot) :- B_2(\cdot)$
- $Q_1 \subseteq Q_2$ ("tartalmaz") ha a Q_1 -re kapott válasz részhalmaza a Q_2 -re kapott válasznak
 - Igaz, ha $f\in B_1(x) \models B_2(x)$
- Ha adott egy Q lekérdezés, és egy Q_1 , válasz lekérdezési terv, akkor:
 - Q_1 helyes lekérdezési terv ha Q_1 -t tartalmazza Q
 - Q_1 teles lekérdezési terv, ha Q -t tartalmazza Q_1
 - Q_1 egy maximálisan tartalmazó lekérdezési terv, ha nem létezik olyan Q_2 amelyik helyes és olyan, hogy Q_1 -t tartalmazza Q_2

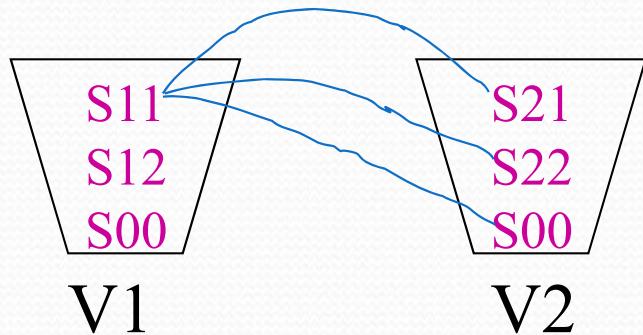
Tartalmazás ellenőrzése

- Tekintsünk két lekérdezést: $Q_1(\cdot) :- B_1(\cdot)$ $Q_2(\cdot) :- B_2(\cdot)$
 - $Q_1 \subseteq Q_2$ („tartalmazó”, “contained in”) ha Q_1 lekérdezésre kapott minden válasz (n -es) részhalmaza Q_2 -nek
 - A tartalmazás áll, ha $B_1(x) |= B_2(x)$
 - (de a vonzat reláció helyessége általánosságba nem eldönthető...)
 - Konjunktív lekérdezések (select/project/join lekérdezések, kényszerek nélkül) ellenőrzését az egyes kifejezések közötti leképezések megadhatóságával ellenőrizzük (exponenciális időigényű algoritmus)
 - m legyen egy (tartalmazási) leképezés $\text{Vars}(Q_2)$ változókról $\text{Vars}(Q_1)$ változókra, ha
 - m leképezi Q_2 törzsében (feltétel részében) található minden részcélját egy Q_1 törzsében található részcélra
 - m leképezi Q_2 fej részét (kvetkezmény részét) Q_1 fej részére
- Eg: $Q_1(x,y) :- R(x), S(y), T(x,y)$ $Q_2(u,v) :- R(u), S(v)$
- Kapcsolódó leképezés: [u/x ; v/y]

Átalakítási algoritmusok

$Q(.) :- V1() \& V2()$

Veder algoritmus



- Veder algoritmus
 - Vedrek kombinációjából előállított terv
 - Utána tartalmazási ellenőrzés

$S11() :- V1()$ $S12() :- V1()$
 $S21() :- V2()$ $S22() :- V2()$
 $S00() :- V1(), V2()$

Inverz szabályok

$Q(.) :- V1() \& V2()$

$V1() :- S11()$

$V1() :- S12()$

$V1() :- S00()$

$V2() :- S21()$

$V2() :- S22()$

$V2() :- S00()$

- Inverz szabályok
 - Rész lekérdezések tervezés

[Levy]

P_1 contains P_2 if
 $P_2 \sqsubseteq P_1$

[Duschka]

Inverz szabályok módszere – példa

Mediált (saját) nézetek:

Önlab(hallgató, tanszék), Tárgyfelvétel(hallgató, kurzus).

S1 távoli adatforrás, amelyet nézettelként definiálunk a mi mediált nézeteink felett:

S1(tanszék, kurzus) :- Önlab(hallgató,tanszék), Tárgyfelvétel(hallgató, kurzus)

Létrehozzunk egy inverz szabályt a nézet minden egyes konjunktjára:

Önlab(fi(T, kurzus) , tanszék) :- S1(tanszék, K)

Tárgyfelvétel(fi(tanszék, K) , kurzus) :- S1(T, kurzus)

(Minden egyes) Z ismeretlenhez, amely több kifejezésben is szerepel, hozunk létre egy funkcionális kifejezést: pl. fi(tanszék, hallgató)

Inverz szabályok módszere – példa

Mediált (saját) nézetek:

Önlab(hallgató, tanszék), Tárgyfelvétel(hallgató, kurzus).

S1 távoli adatforrás, amelyet nézettelként definiálunk a mi mediált nézeteink felett:

S1(tanszék, kurzus) :- Önlab(hallgató,tanszék), Tárgyfelvétel(hallgató, kurzus)

Létrehozzunk egy inverz szabályt a nézet minden egyes konjunktjára:

Önlab(fi(X, kurzus) ,tanszék) :- S1(tanszék, X)

Tárgyfelvétel(fi(tanszék, Y) , kurzus) :- S1(Y, kurzus)

Lekérdezés: q(tanszék) :- Önlab(H,tanszék), Tárgyfelvétel(H, „Adatbázisok”)

S1 tartalmazza a következő ketteseket:

< (TMIT, „Adatbázisok”), (MIT, „Adatbázisok”), (MIT, „Mesterséges intelligencia”)

Inverz szabályok módszere – példa

Önlab(**fi(X, kurzus)** ,tanszék) :- S1(tanszék, **X**)

Tárgyfelvétel(**fi(tanszék, Y)** , kurzus) :- S1(**Y**, kurzus)

Lekérdezés: q(tanszék) :- Önlab(H,tanszék), Tárgyfelvétel(H, „Adatbázisok”)

S1 tartalmazza a következő ketteseket:

< (TMIT, „Adatbázsiok”), (MIT, „Adatbázisok”), (MIT, „Mesterséges intelligencia”)

Eredmény:

Tárgyfelvétel: <(fi(TMIT, "Adatbázisok"), „Adatbázisok”),
(fi(MIT,"Adatbázisok"), "Adatbázisok"),
(fi(MIT,"Mesterséges intelligencia"),"Mesterséges intelligencia") >

Önlab: < (fi(TMIT, "Adatbázisok"), TMIT),
(fi(MIT,"Adatbázisok"), MIT),
(fi(MIT,"Mesterséges intelligencia"), MIT) >

Válasz: TMIT, MIT

Forrás elérés korlátok

- A források nem feltétlenül relációs adatbázisok
 - Jogosultsági korlátok
 - Korlátos elérési minták
 - (Pl. telefonkönyv lekérdezése)
 - Korlátos kiszolgáló erőforrás
 - (Csak attribútumokon értelmezett szűrésen keresztül érhetőek el adatok.)
- Elérési korlátozások modellezhetők:
 - b: kötelezően megadandó keresési attribútum
 - f: szabadon elérhető attribútum

Elérési korlátok – rekurzív algoritmusok

Create Source S_1 as

```
SELECT *  
from Hivatkozások  
adott cikkhez
```

$S1^{bf}(p_1, p_2) :- \text{Hivatkozások}(p_1, p_2)$

$S2(p) :- \text{Asu-C}(p)$

$S3^b(p) :- \text{DíjC}(p)$

Create Source S_2 as

```
SELECT cikk  
from ASU-Cikkek
```

$Q(p) :- \text{DíjC}(p)$

$\text{Díjc}(p) :- \text{ÖsszC}(p), S3^b(p)$

Create Source S_3 as

```
SELECT cikk  
from DíjazottCikkek  
adott cikkhez
```

$\text{Asu-C}(p) :- S2(p)$

$\text{Hivatkozások}(p_1, p_2) :- \text{ÖsszC}(p_1), S1^{bf}(p_1, p_2)$

Query: SELECT * from
DijazottCikkek

$\text{ÖsszC}(p) :- S2(p)$

$\text{ÖsszC}(p) :- \text{ÖsszC}(p_1), S1(p_1, p)$

Veder algoritmus

- A Q lekérdezés minden részcélját lefedjük releváns nézetekkel
- Készítünk egy listát (veder) azokból a nézetekből, amelyek adhatnak eredményt az adott részcélhoz
- Megvizsgáljuk a vedrekben listázott nézetek kombinációját
- Nem feltétlenül minden kombináció megfelelő
- Megtartjuk a megfelelő nézetek kombinációit és minimalizáljuk a megoldást
- Elhagyjuk a redundánsokat
- A megoldás a megmaradó lekérdezések uniója

Veder algoritmus példa

Mediált (saját) nézetek:

$\text{reg}(\text{Std}, \text{Crs}, \text{Qtr})$, $\text{course}(\text{Crs}, \text{Title})$, $\text{teaches}(\text{Prof}, \text{Crs}, \text{Qtr})$

S_1, S_2, S_3, S_4 távoli adatforrások, amelyeket nézettelként definiálunk a mi mediált nézeteink felett:

$S_1(\text{Std}, \text{Crs}, \text{Qtr}, \text{Title}) :- \text{reg}(\text{Std}, \text{Crs}, \text{Qtr}), \text{course}(\text{Crs}, \text{Title}),$
 $\text{Crs} \geq 500, \text{Qtr} \geq \text{Aut98}$

$S_2(\text{Std}, \text{Prof}, \text{Crs}, \text{Qtr}) :- \text{reg}(\text{Std}, \text{Crs}, \text{Qtr}), \text{teaches}(\text{Prof}, \text{Crs}, \text{Qtr})$

$S_3(\text{Std}, \text{Crs}) :- \text{reg}(\text{Std}, \text{Crs}, \text{Qtr}), \text{Qtr} \leq \text{Aut94}$

$S_4(\text{Prof}, \text{Crs}, \text{Title}, \text{Qtr}) :- \text{reg}(\text{Std}, \text{Crs}, \text{Qtr}), \text{course}(\text{Crs}, \text{Title}),$
 $\text{teaches}(\text{Prof}, \text{Crs}, \text{Qtr}), \text{Qtr} \leq \text{Aut97}$

$q(S, C, P) :- \text{teaches}(P, C, Q), \text{reg}(S, C, Q), \text{course}(C, T),$
 $C \geq 300, Q \geq \text{Aut95}$

1. lépés: minden részcélhoz összegyűjtjük a releváns nézeteket

Veder algoritmus példa

S₁(Std,Crs,Qtr,Title) :- reg(Std,Crs,Qtr), course(Crs,Title),
Crs \geq 500, Qtr \geq Aut98

S₂(Std,Prof,Crs,Qtr) :- reg(Std,Crs,Qtr), teaches(Prof,Crs,Qtr)

S₃(Std,Crs) :- reg(Std,Crs,Qtr), Qtr \leq Aut94

S₄(Prof,Crs,Title,Qtr) :- reg(Std,Crs,Qtr), course(Crs,Title),
teaches(Prof,Crs,Qtr), Qtr \leq Aut97

q(S,C,P) :- teaches(P,C,Q), reg(S,C,Q), course(C,T),
C \geq 300, Q \geq Aut95

P \rightarrow Prof, C \rightarrow Crs, Q \rightarrow Qtr

Buckets

teaches	reg	course
S2		
S4		

Megjegyzés: Itt az aritmetikai predikátumok nem okoznak problémát

Veder algoritmus példa

S₁(Std,Crs,Qtr,Title) :- **reg**(Std,Crs,Qtr), course(Crs,Title),
Crs \geq 500, Qtr \geq Aut98

S₂(Std,Prof,Crs,Qtr) :- **reg**(Std,Crs,Qtr), teaches(Prof,Crs,Qtr)

S₃(Std,Crs) :- **reg**(Std,Crs,Qtr), Qtr \leq Aut94

S₄(Prof,Crs,Title,Qtr) :- **reg**(Std,Crs,Qtr), course(Crs,Title),
teaches(Prof,Crs,Qtr), Qtr \leq Aut97

q(S,C,P) :- teaches(P,C,Q), **reg**(S,C,Q), course(C,T),
C \geq 300, Q \geq Aut95

S \rightarrow Std, **C** \rightarrow Crs, **Q** \rightarrow Qtr

Megjegyzés: S₃ nem ad megoldást, az aritmetikai predikátumok nem kielégíthetőek

S₄ nem ad megoldást: S nem szerepel a S₄ kimenetében

Buckets

teaches	reg	course
S ₂	S ₁	
S ₄	S ₂	

Veder algoritmus példa

S₁(Std,Crs,Qtr,Title) :- reg(Std,Crs,Qtr), course(Crs,Title),
Crs \geq 500, Qtr \geq Aut98

S₂(Std,Prof,Crs,Qtr) :- reg(Std,Crs,Qtr), teaches(Prof,Crs,Qtr)

S₃(Std,Crs) :- reg(Std,Crs,Qtr), Qtr \leq Aut94

S₄(Prof,Crs,Title,Qtr) :- reg(Std,Crs,Qtr), course(Crs,Title),
teaches(Prof,Crs,Qtr), Qtr \leq Aut97

q(S,C,P) :- teaches(P,C,Q), reg(S,C,Q), course(C,T),
C \geq 300, Q \geq Aut95

C \rightarrow Crs, T \rightarrow Title

Buckets

teaches	reg	course
S ₂	S ₁	S ₁
S ₄	S ₂	S ₄

Veder algoritmus példa

2. lépés:

- Nézetek összes kombinációját figyelembe kell venni, minden vederből egy elemet véve
- Aritmetikai predikátumok ellenőrzése a kombinációra
 - Például két nézet lehet átfedő vagy inkonzisztens
- Eredmény átírás= a megmaradók uniója

Lekérdezés átírás1:

teaches	reg	course
S2	S1	S1
S4	S2	S4

q1(S,C,P) :- S2(S',P,C,Q), S1(S,C,Q,T'), S1(S'',C,Q',T)

- Nincs probléma az aritmetikai predikátumokkal(nincs a S2-ben)
- Minimális-e a megoldás

Veder algoritmus példa

Átírás kibontása 1:

$q1'(S,C,P) :- r(S',C,Q), t(P,C,Q), r(S,C,Q), c(C,T'), r(S'',C,Q'), c(C,T), C \geq 500, Q \geq \text{Aut98}, C \geq 500, Q' \geq \text{Aut98}$

- Fekete r-ek leképezhetőek a zöld r-ekre:

$S' \rightarrow S, S'' \rightarrow S, Q' \rightarrow Q$

- Fekete c leképezhető a zöld c-re:

leképezés kibővíthető: $T \rightarrow T'$

Minimális kibontása az átírásnak 1:

$q1m'(S,C,P) :- t(P,C,Q), r(S,C,Q), c(C,T'), C \geq 500, Q \geq \text{Aut98}$

Minimális átírás 1:

$q1m(S,C,P) :- S2(S',P,C,Q), S1(S,C,Q,T')$

Veder algoritmus példa

Lekérdezés átírás 2:

teaches	reg	course
S2	S1	S1
S4	S2	S4

$q2(S, C, P) :- S2(S', P, C, Q), S1(S, C, Q, T'), S4(P', C, T, Q')$

$q2'(S, C, P) :- r(S', C, Q), t(P, C, Q), r(S, C, Q),$
 $r(S, C, Q), c(C, T'), C \geq 500, Q \geq \text{Aut98},$
 $r(S'', C, Q'), c(C, T), t(P', C, Q'), Q' \leq \text{Aut97}$

- A kombináció nem kielégíthető: Nézzük meg **S1** és **S4** kombinációját

Lekérdezés átírás 3:

teaches	reg	course
S2	S1	S1
S4	S2	S4

$q3(S, C, P) :- S2(S', P, C, Q), S2(S, P', C, Q), S4(P'', C, T, Q')$

Veder algoritmus példa

Lekérdezés átírás kibontása 3:

$q3'(S,C,P) :- r(S',C,Q), t(P,C,Q), r(S,C,Q), t(P',C,Q), r(S'',C,Q'), c(C,T), t(P'',C,Q'), Q' \leq \text{Aut97}$

- A zöld részcélok lefedhetőek a feketékkel a következő megfeleltetésekkel: $S' \rightarrow S$, $S'' \rightarrow S$, $P' \rightarrow P$, $P'' \rightarrow P$, $Q' \rightarrow Q$

Minimális átírás3:

$q3m(S,C,P) :- S2(S,P,C,Q), S4(P,C,T,Q)$

Tehát két átírás maradt.

Maximális tartalmazó átírás:

$$q' = q1m \cup q3m$$

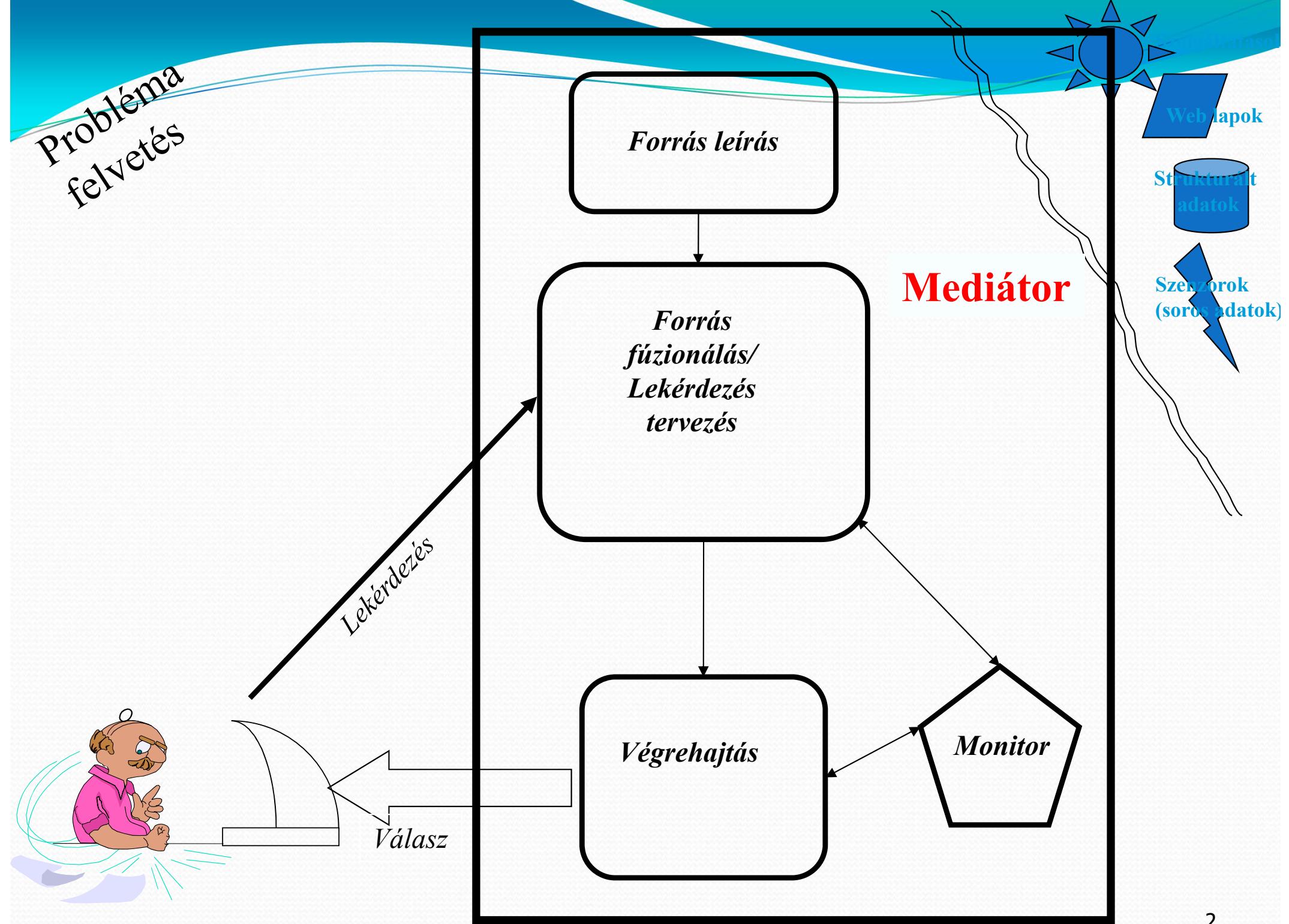
Integrációs és ellenőrzési technikák
VIMIAC04, 2021. tavasz

Információ integráció (Szemantikus Web megközelítés a másik irányból)

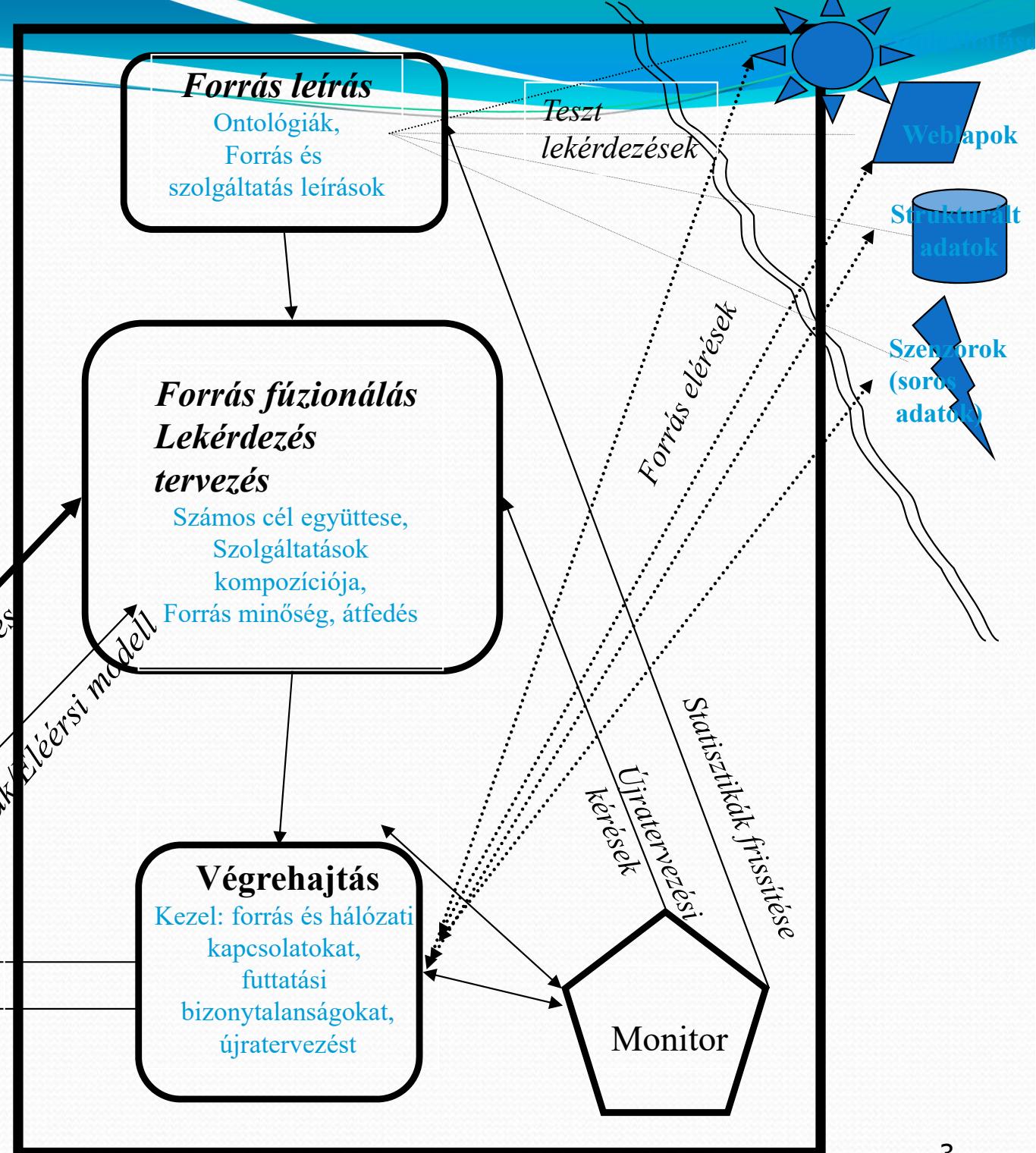
5. Előadás

Méréstechnika és Információs Rendszerek Tanszék

Probléma felvetés



- Felhasználói lekérdezések megfogalmazása a mediált sémán.
- Adatok tárolva *lokális sémában*.
- A tárolt információ (tartalom) ismerete alapján megfogalmazható a leképezés a sémák között.
- A mediátor alkalmazza a leképezést a felhasználói kérdés lefordítására a forrás lekérdezésekre.



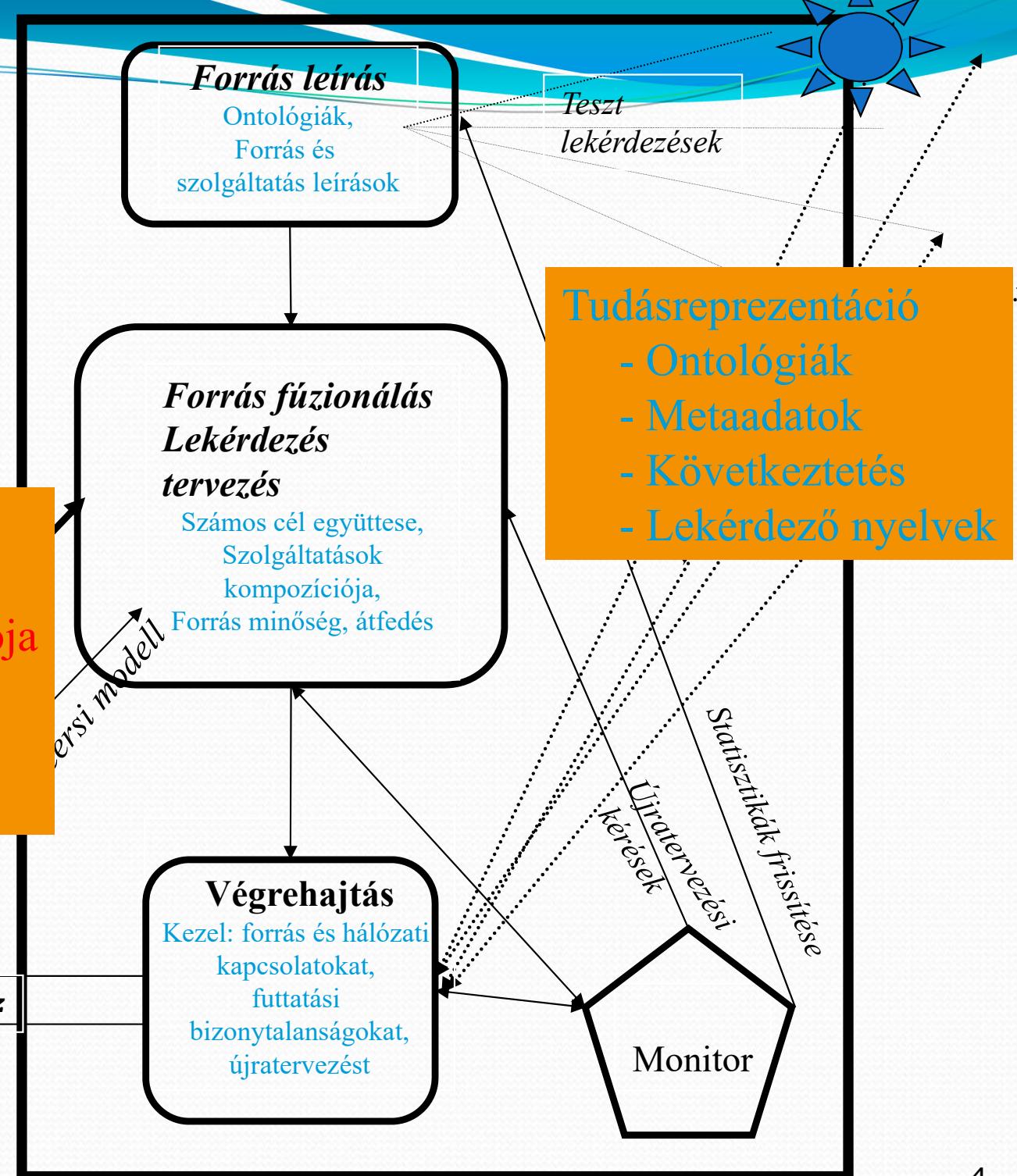
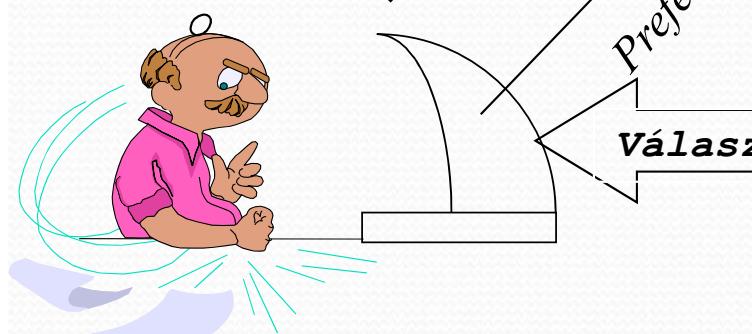
Hol az MI szerepe?

Tanulás/bányászás

- Forrás felkutatás
- Forrás statisztikák
- Wrapper tanulás

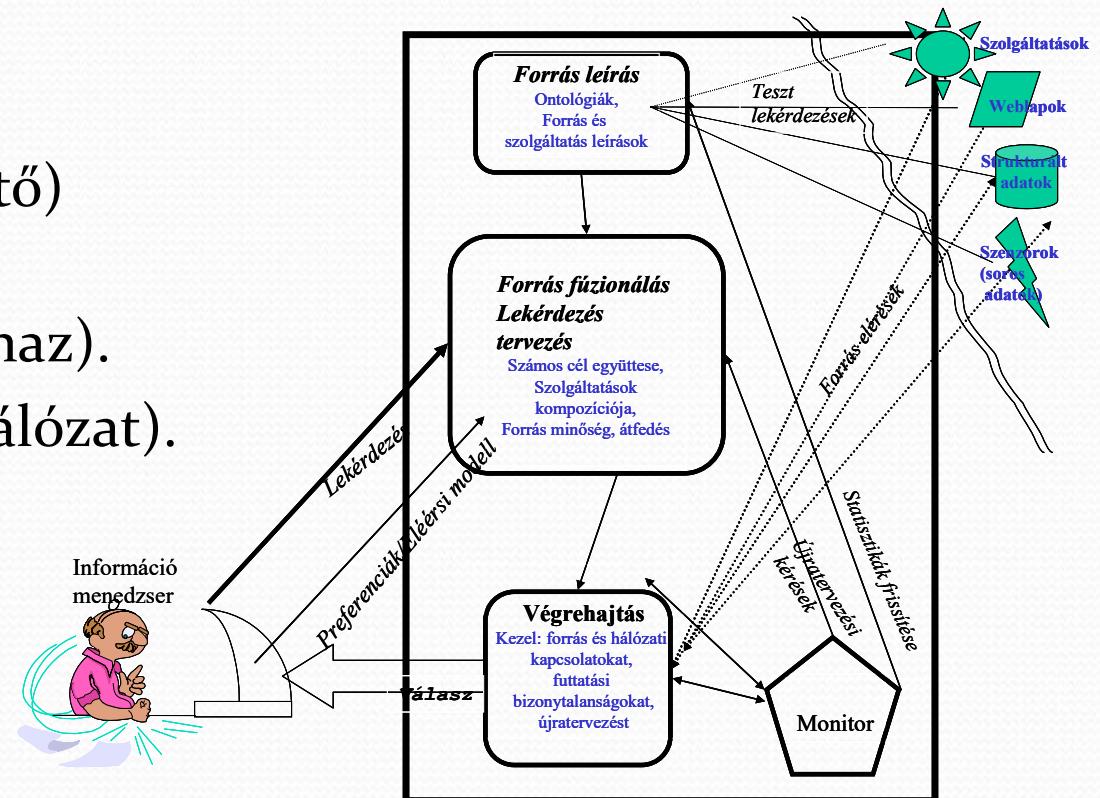
Automata tervezés

- Nyelvek tervezése
- Szolgáltatások kompozíciója
- Reaktív tervezés/
terv monitorozás



Forrás leírások

- minden meta-adat információt tartalmaz a forrásokkal kapcsolatban:
 - Forrás tartalom logikai leírása (könyvek, új autók).
 - Forrás képességek (pl. SQL lekérdezés feltehető)
 - Forrás teljesség (pl. *minden* könyvet tartalmaz).
 - Fizikai jellemzők (forrás, hálózat).
 - Statisztikák az adatokról
 - Tükör források
 - Frissítési frekvencia.

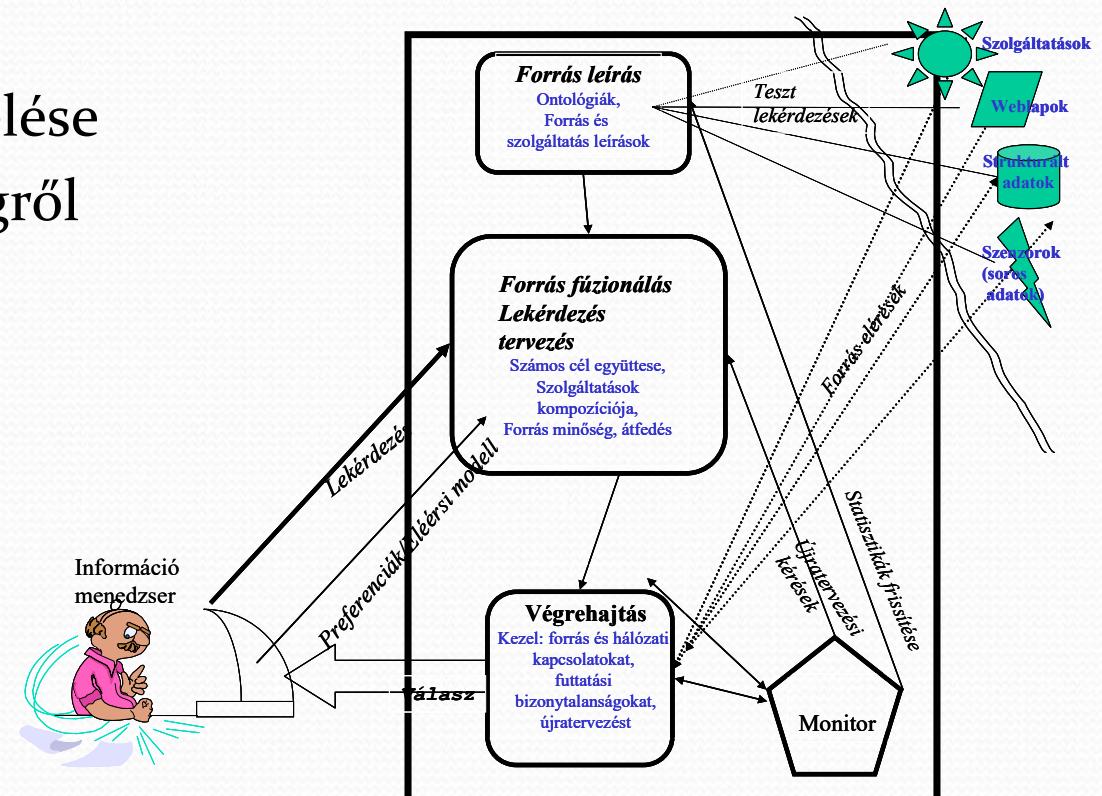


Forrás elérések

- Hogyan kapunk n-eseket
 - Számos forrás strukturálatlan adatokat tartalmaz
 - Bizonyos források inherensen strukturáltalanok, mások természetes nyelvi köntösben vannak
 - Vissza kell csomagolni az adatokat
 - Wrapper építés/információ kinyerés

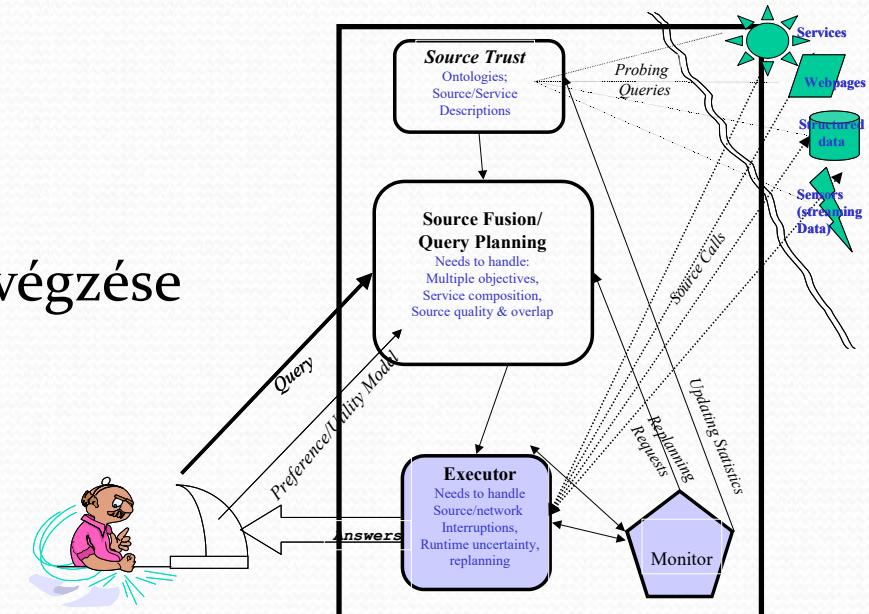
Forrás fúzió/ lekérdezés tervezés

- Feldolgozza a felhasználói lekérdezést és előállítja a végrehajtási tervet:
 - Költség és hatékonyság közti optimalizáció
 - Forrás elérési korlátok kezelése
 - Információ a forrásminőségről



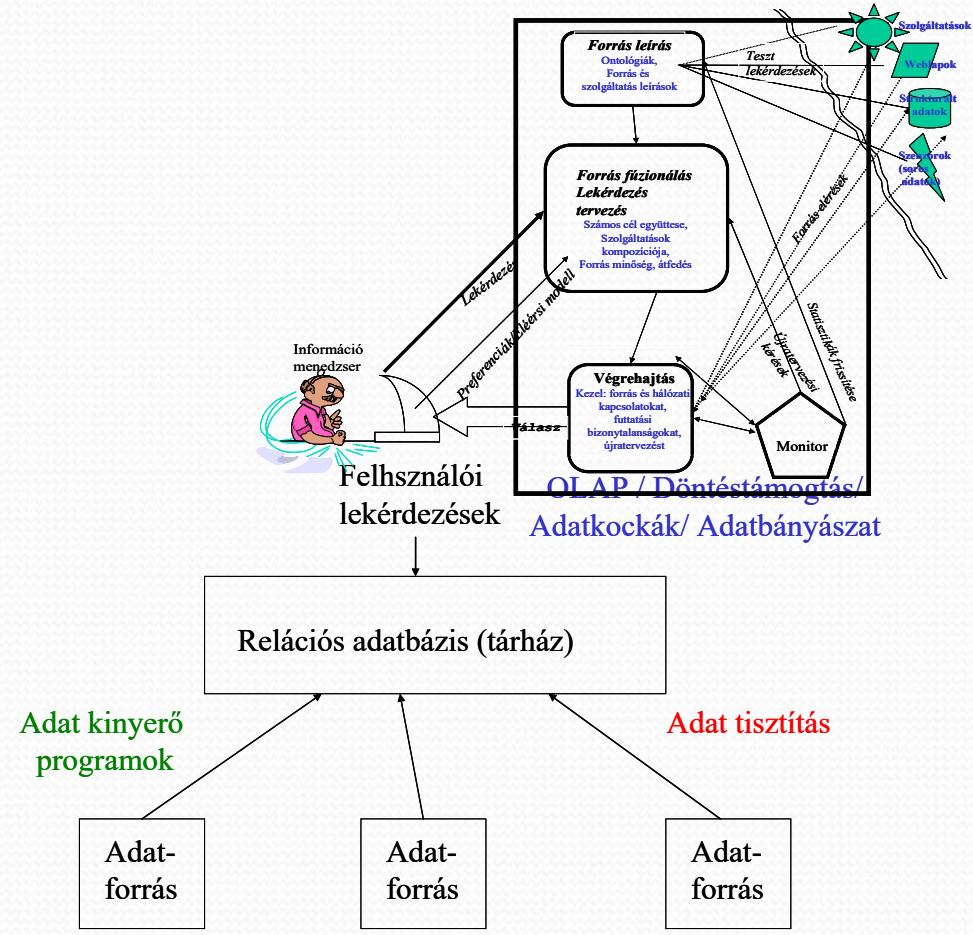
Monitoring / Végrehajtás

- Lekérdezési terv alapján elvégzi az adatok begyűjtését a forrásokból:
 - Forrás késleltetések kezelése
 - Hálózati, tranzíens kimaradások
 - Forrás elérési korlátok
 - Szükséges lehet újratervezések elvégzése
- Méretezés, tervezés:
 - Hány forrást kell elérni?
 - Mennyire autonómok ezek?
 - Van ismeretünk a forrásokról?
 - Strukturáltak az adatok?
 - Csak lekérdezés lehetséges vagy módosítás is?
 - Követelmények: pontosság, teljesség, teljesítmény, inkonzisztenciák kezelése
 - Zárt vagy nyílt világ feltételezés?



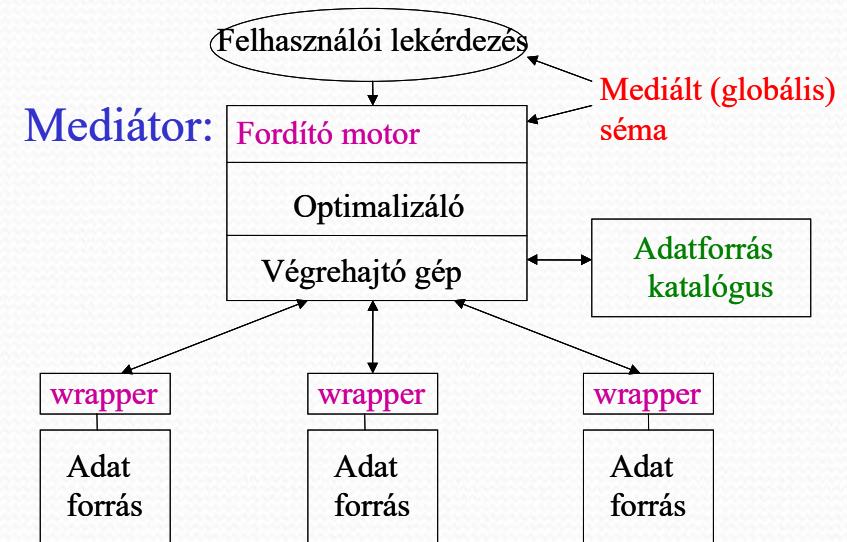
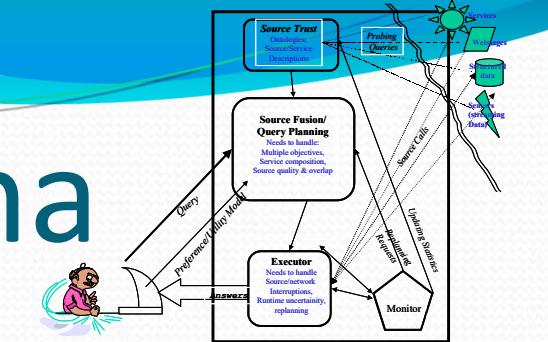
Kis forrás szám melletti integráció

- **Általában ad-hoc programozás:** speciális eset megvalósítása minden esetre, sok konzultáció.
- **Adattárházak:** minden adat periódikus feltöltése az adattárházba.
 - 6-18 hónap bevezetési idő
 - Operációs és döntéstámogatási RDBMS elválasztás. (nem csak adatintegrációra megoldás).
 - Teljesítmény jó,
 - adat lehet, hogy nem friss;
 - Rendszeres adattisztítás szükséges.



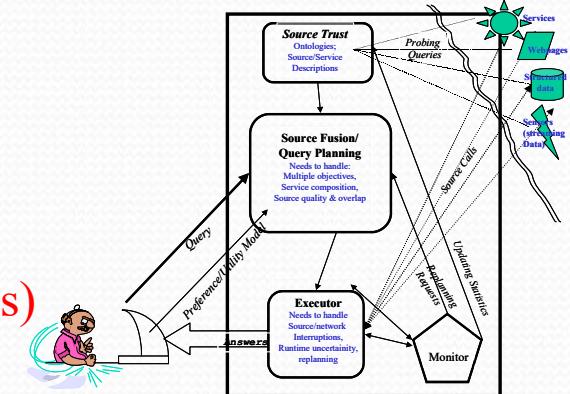
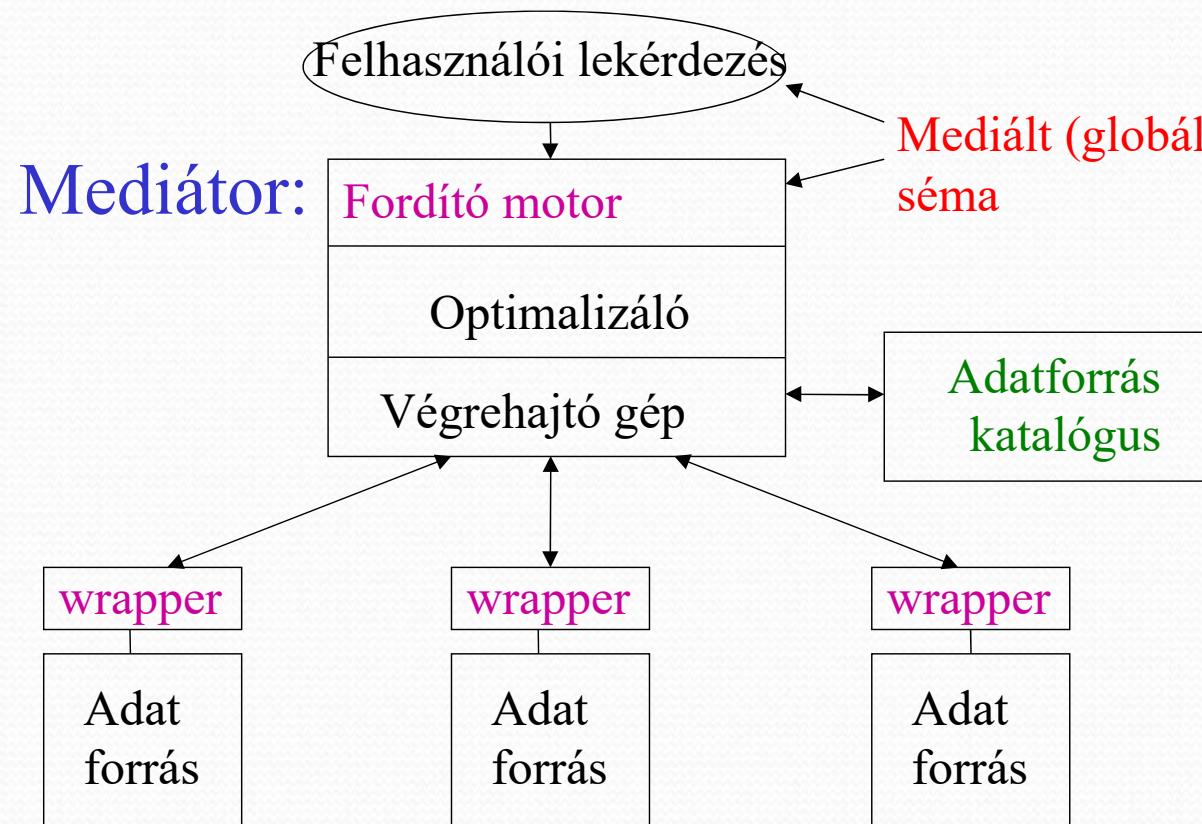
Virtuális integrációs séma

- Adatok a forrásokban maradnak
- Lekérdezés végrehajtásakor:
 - Releváns források meghatározása
 - Lekérdezés szétválasztása forrásokra vonatkozó lekérdezésekre.
 - Válaszok begyűjtése a forrásokból, és megfelelő kombinálása a válasz előállításához.
- Friss adatok
- A megoldás skálázható



Garlic [IBM], Hermes[UMD]; Tsimmis, InfoMaster[Stanford]; DISCO[INRIA]; Information Manifold [AT&T]; SIMS/Ariadne[USC]; Emerac/Havasu[ASU]

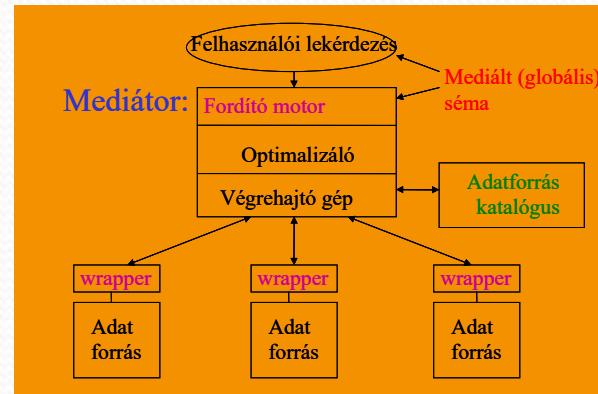
Virtuális integrátor architektúra



Források: relációs adatbázisok, weblapok, szövegek.

Forrás-mediátor relációs sémával szembeni elvárások

- **Kifejező erő:** hasonló adattartalommal rendelkező források megkülönböztetése, irreleváns források felismerése.
- **Egyszerű bővíthetőség:** tegyük könnyűvé források hozzáadását.
- **Fordítás/átalakítás:** felhasználói lekérdezés lefordítása forrásokon értelmezett lekérdezésekre hatékonyan és eredményesen.
- **Veszteségmentesség:** minden lehetséges adatelérés biztosítása



Lekérdezés átalakítás

- **Adott:**
 - Egy Q lekérdezés a mediátor sémára vonatkozóan
 - Adat források leírása
- **Létrehozandó:**
 - Egy Q' lekérdezés az adat forrásokra vonatkozóan, amely:
 - Q' csak helyes válaszokat ad a Q lekérdezéshez és
 - Q' minden lehetséges választ megtalál Q-hoz az elérhető forrásokból.

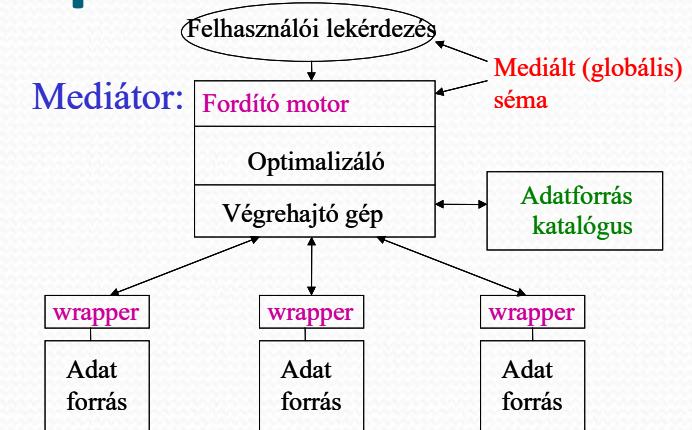
Fordítási/átfogalmazási probléma

- **Adott:**

- Egy Q lekérdezés a mediátor sémára vonatkozóan
- Adat források leírása

- **Létrehozandó:**

- Egy Q' lekérdezés az adat forrásokra vonatkozóan, amely:
 - Q' csak *helyes válaszokat* ad a Q lekérdezéshez és
 - Q' minden lehetséges választ megtalál Q-hoz az elérhető forrásokból.



Forrás és felhasználói sémák reláció leírásának megközelítései

- **Globális mediált sémák (Global-as-view, GAV):** a mediált séma kifejezése a forrásokra vonatkozó nézetek relációjaként
- **Lokális mediált sémák (Local-as-view, LAV):** forrás relációk kifejezése a mediált sémákon értelmezett relációkkal.
- Módszerek kombinációja...?

“Nézet” frissítés

```
CREATE VIEW Seattle-view AS  
  
SELECT buyer, seller, product, store  
FROM Person, Purchase  
WHERE Person.city = "Seattle" AND  
Person.name = Purchase.buyer
```

A nézet felhasználása:

```
SELECT name, store  
FROM Seattle-view, Product  
WHERE Seattle-view.product = Product.name AND  
Product.category = "shoes"
```

Mintapélda

- Hasonlítsuk össze a virtuális mediátor megközelítéseket film/mozi tárgyterületen
- Minta: Mediátor struktúra egy film adatbázishoz
 - Információk szolgáltatása filmekről, illetve mozi programról néhány forrás adatbázis és két mediált nézet felhasználásával

Globális mediált nézet

GAV (Global-As-View)

Mediált/felhasználói séma:

Filmek(cím, rendező, év, típus),
Műsor(mozi, cím, idő).

Mediált séma kifejezése
a forrásokra vonatkozó
nézetek relációjaként.

Create View Filmek AS

select * from S1

[S1(cím, rendező, év, típus)]

union

select * from S2

[S2(cím, rendező, év, típus)]

union

[S3(cím, rendező), S4(cím, év, típus)]

select S3.cím, S3.rendező, S4.év, S4.típus

from S3, S4

where S3.cím=S4.cím

GAV

Mediált/felhasználói séma:

Filmek(cím, rendező, év, típus),
Műsor(mozi, cím, idő).

Create View Filmek AS

select * from S1

[S1(cím, rendező, év, típus)]

union

select * from S2

[S2(cím, rendező, év, típus)]

union

select S3.cím, S3.rendező, S4.év, S4.típus

[S3(cím, rendező), S4(cím, év, típus)]

from S3, S4

where S3.cím=S4.cím

A mediátor séma relációk virtuális nézetek a forrásrelációkon.

Mediált séma kifejezése
a forrásokra vonatkozó
nézetek relációjaként.

GAV: példa 2.

Mediált/felhasználói séma:

Filmek(cím, rendező, év, típus),
Műsor(mozi, cím, idő).

Mediált séma kifejezése
a forrásokra vonatkozó
nézetek relációjaként.

Create View Filmek AS

select * from S1

[S1(cím,rendező,év)]

select cím, rendező, év, NULL
from S1

Null értékek

union

[S2(cím, rendező, típus)]

select cím, rendező, NULL, típus
from S2

GAV: példa 2.

Mediált/felhasználói séma:

Filmek(cím, rendező, év, típus),
Műsor(mozi, cím, idő).

Forrás S4: S4(mozi, típus)

Mediált séma kifejezése
a forrásokra vonatkozó
nézetek relációjaként.

Create View Filmek AS

```
select NULL, NULL, NULL, típus  
from S4
```

Create View Műsor AS

```
select mozi, NULL, NULL  
from S4.
```

*De mit lehetne tenni, ha minket a vígjátékokat játszó
mozik érdekelnének?*

“Veszteséges medáció”

LAV: példa 1

Mediált/felhasználói séma:

Filmek(cím, rendező, év, típus),
Műsor(mozi, cím, idő).

Forrás séma kifejezése
a mediált nézeteken
értelmezett relációkként.

Create Source S1 AS

```
select * from Filmek
```

S1(cím, rendező, év, típus)

Create Source S3 AS

```
select cím, rendező from Filmek
```

S3(cím, rendező)

Create Source S5 AS

```
select cím, rendező, év
```

S5(cím, rendező, év), év >1960

```
from Filmek
```

```
where év > 1960 AND típus=“vígjáték”
```

A források “materializált nézetek”
a mediált sémák felett.

LAV: példa 1

Mediált/felhasználói séma:

Filmek(cím, rendező, év, típus),

Műsor(mozi, cím, idő).

Create Source S4 AS

select mozi, típus

from Filmek m, Műsor s

where m.cím=s.cím

Van remény a vígjátékokat játszó mozik felderítésére!

Forrás séma kifejezése
a mediált nézeteken
értelmezett relációkként.

S4(Mozi,Típus)

GAV vs. LAV

Mediált séma:

Filmek(**cím, rendező, év, típus**),
Műsor(**mozi, cím, idő**).

Forrás S4: S4(mozi, típus)

Create View Filmek AS

```
select NULL, NULL, NULL, típus  
from S4
```

Create View Műsor AS

```
select mozi, NULL, NULL  
from S4.
```

De mit lehetne tenni, ha minket a vígjátékokat játszó mozik érdekelnének?

Create Source S4 AS

```
select mozi, típus  
from Filmek m, Műsor s  
where m.cím=s.cím
```

Veszteséges mediáció

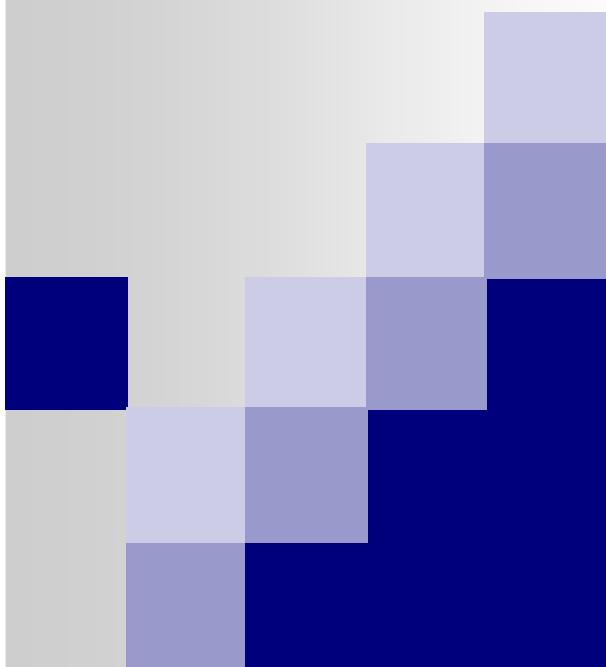
GAV

VS.

LAV

- Nem moduláris
 - Források hozzáadása módosítja a meglévő mediált séma definícióját
- Nehézkes lehet veszteségmentes mediátort készíteni.
- Lekérdezés átalakítás egyszerű
 - Nézetek kibontását jelenti (*polinomiális*)
 - Hierarchikus mediátor sémák létrehozása lehetséges
- Hatékony, ha
 - Kis számú, ritkán változó adatforrás van
 - Feladat teljesen ismert a mediátor tervezésekor (pl. vállalati adatintegráció)
 - Garlic, TSIMMIS, HERMES

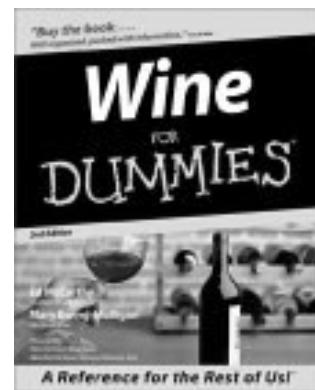
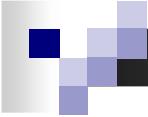
- Moduláris—új forrás hozzáadása egyszerű
- Igen rugalmas – a lekérdező nyelv közvetlenül alkalmazható a források leírására
- Lekérdezés átalakítás bonyolult
 - Válaszokat a nézeteken keresztül kell előállítani (nem minden megoldható)
- Hatékony, ha
 - Sok, kevéssé korrelált forrás
 - Források dinamikus hozzáadása és törlése
 - Information Manifold, InfoMaster, Emerac, Havasu



Ontológiák építése

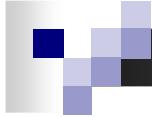
Ontology Engineering

A Protege OWL eszköz modellezéshez javasolt módszertan és példák áttekintése a labor előtt
(forrás: www.protege.stanford.edu)



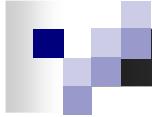
Ontológia borokról és ételekről





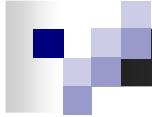
Vázlat

- Mi is az ontológia?
- Miért fejlesszünk ontológiákat?
- Ontológiák építése lépésről lépésre
- Alaposabb elemzés: problémák és megoldások
- Ontológiák a szemantikus web eszközeivel
- Ontológiák fejlesztésének jelenlegi irányai



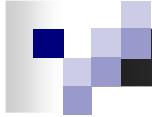
Mi az ontológia?

- Az ontológia egy tárgyterület explicit leírása:
 - Fogalmak
 - Fogalmak attribútumai és jellemzői
 - Attribútumok és jellemzők kényszerei
 - Egyedek
- Az ontológia tartalma
 - Közös szótár
 - Egy megosztható értelmezés



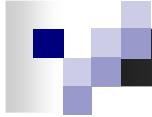
Ontológia példák

- Weben elérhető taxonómiák
 - Yahoo! kategóriák
- On-line boltok kategóriái
 - Amazon.com termék katalógus
- Tárgyterület függő standard terminológiák
 - Unified Medical Language System (UMLS)
 - UNSPSC – termékek és szolgáltatások terminológiái



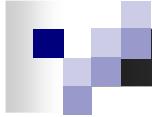
Vázlat

- Mi is az ontológia?
- Miért fejlesszünk ontológiákat?
- Ontológiák építése lépésről lépésre
- Alaposabb elemzés: problémák és megoldások
- Ontológiák a szemantikus web eszközeivel
- Ontológiák fejlesztésének jelenlegi irányai



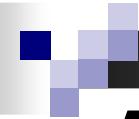
Miért fejlesszünk ontológiát

- Az információk struktúrájáról lévő értelmezés megosztása
 - emberek között
 - szoftver ágensek között
- Egy tárgyterületről leírt ismeretek újrafelhasználhatóságának biztosítása
 - elkerüljük a kerék újrafelfedezését
 - standardok bevezetése a osztott feladatmegoldás támogatásához

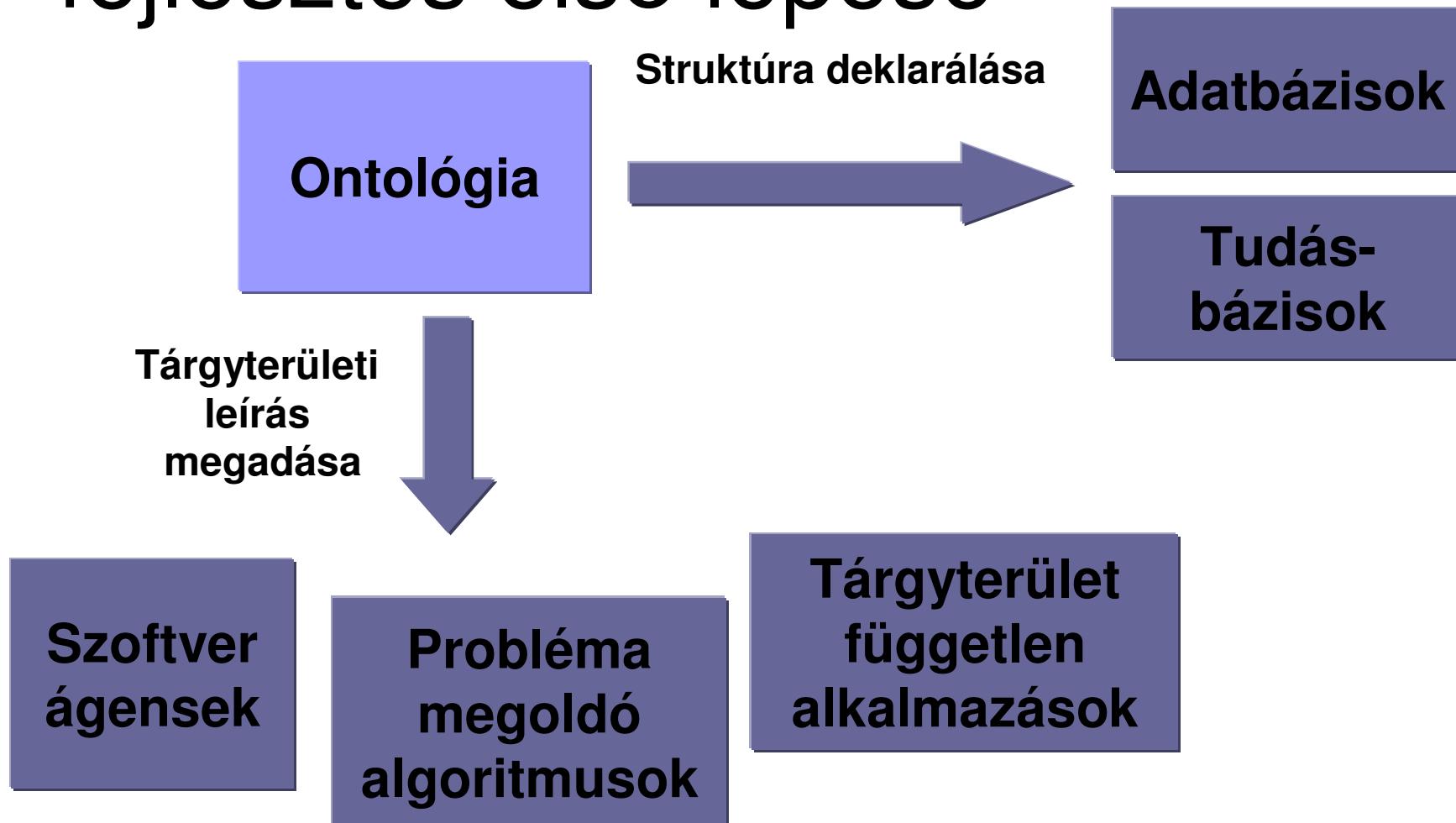


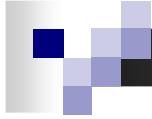
További érvek

- Tárgyterületi feltételezések explicit leírása
 - könnyebb változatathatóság a tárgyterület leírásában
 - Könnyebb megérteni és módosítani az öröklött információkat
- Tárgyterületi tudás szétválasztása a működéssel kapcsolatos tudástól
 - A tárgyterületi és operatív tudás újrafelhasználhatósága külön-külön (kényszerekkel leírt konfigurációk)



Az ontológia van amikor a fejlesztés első lépése

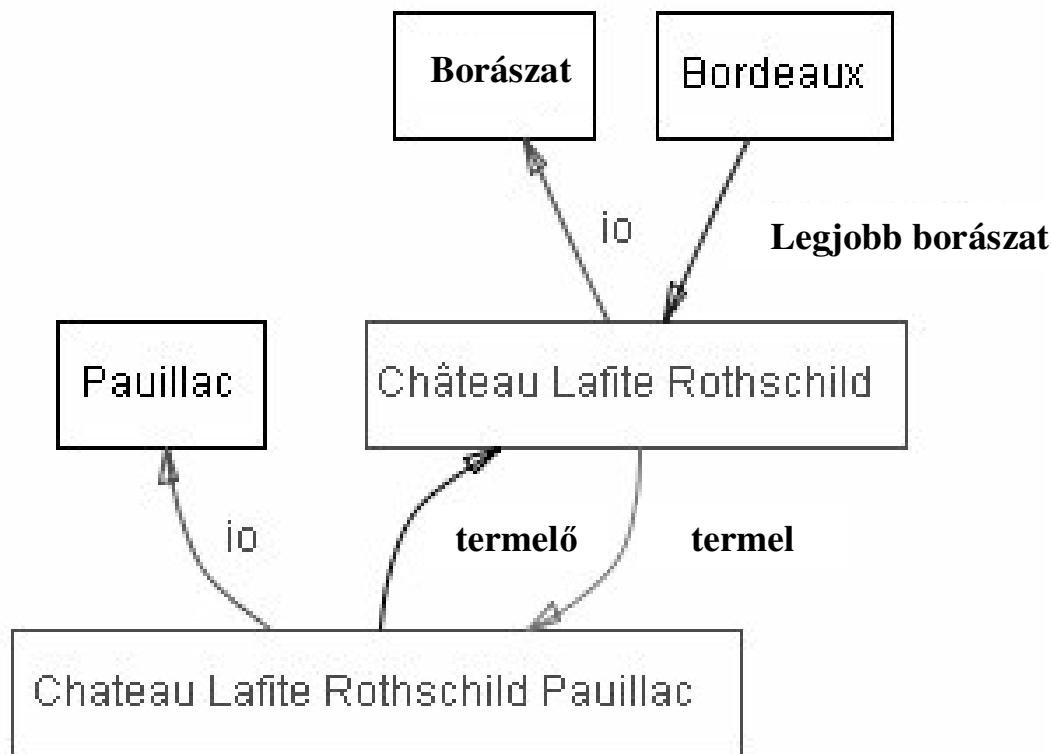


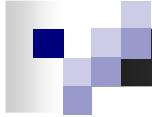


Vázlat

- Mi is az ontológia?
- Miért fejlesszünk ontológiákat?
- Ontológiák építése lépésről lépésre
- Alaposabb elemzés: problémák és megoldások
- Ontológiák a szemantikus web eszközeivel
- Ontológiák fejlesztésének jelenlegi irányai

Borok és borászatok





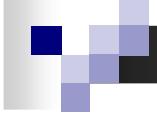
Ontológia fejlesztési folyamat

Fontosabb lépések:



Valóságban egy iteratív folyamat:





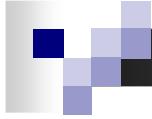
Ontológi építés vs. objektumorientált programstruktúrák

Ontológia

- a világ struktúrájának jellemzése
- fogalmak struktúrája
- nem foglalkozunk a fizikai reprezentációval

Egy OO osztály struktúra

- Az adat és a kód struktúrájának leírása
- Viselkedés struktúrája (metódusok)
- Az adatok fizikai reprezentációjának leírása (long int, char, etc.)



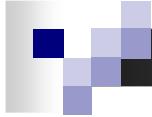
Eszközök (előzetes)

■ Protege 2000:

- Grafikus ontológia fejlesztő eszköz
- Gazdag tudás modell támogatása
- Nyitott forráskód, szabadon elérhető
(<http://protege.stanford.edu>)

■ Más eszközök:

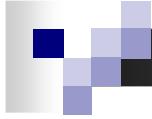
- Ontolingua és Chimaera
- OntoEdit
- OilEd



Tárgyterület és látókör meghatározása

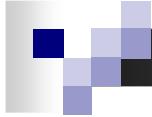


- Ontológia tárgyterületének megválasztása?
 - Mire kívánjuk használni az ontológiát?
 - Milyen típusú kérdésekre kell tudni választ adni (kompetencia kérdések)?
- A fejlesztés ideje alatt ezekre a kérdésekre adott válaszok változhatnak*



Kompetencia kérdések

- Milyen jellemzők alapján választunk bort?
- A bordói borok vörös vagy fehér borok?
- A Cabernet Savignon illik-e halételhez?
- Mi a legjobb választása sült húsokhoz?
- A bor milyen jellemzői befolyásolják, hogy illik-e egy ételtípushoz?
- Változik-e egy bor íze, testessége az évjárat függvényében?
- Melyek voltak Napa Zinfandel legjobb évjáratai?

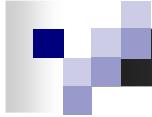


Újrafelhasználhatóság



■ Miért fontos az újrafelhasználhatóság?

- ráfordítások csökkentése
- más ontológiákat használó eszközökkel kooperálás
- Olyan ontológiákat használhassunk, amelyek igazolhatóak más alkalmazások által



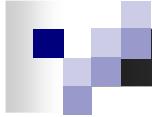
Mit akarunk újrafelhasználni?

■ Ontológia könyvtárak

- DAML ontológia könyvtár (www.daml.org/ontologies)
- Ontolingua könyvtár
(www.ksl.stanford.edu/software/ontolingua/)
- Protégé ontológia könyvtár (protege.stanford.edu/plugins.html)

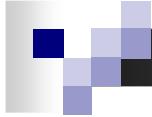
■ Felsőbb ontológiák

- IEEE Standard Upper Ontology (suo.ieee.org)
- Cyc (www.cyc.com)



Mit akarunk újrafelhasználni?(II)

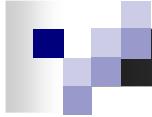
- Általános ontológiák
 - DMOZ (www.dmoz.org)
 - WordNet (www.cogsci.princeton.edu/~wn/)
- Tárgyterületi ontológiák
 - UMLS Semantic Net
 - GO (Gene Ontology) (www.geneontology.org)



Fontos termek felsorolása



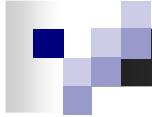
- Mik azok a termek amiket használni akarunk?
- Milyen tulajdonságai vannak ezeknek?
- Mit akarunk ezekről kijelenteni?



Termek felsorolása – A bor ontológia

*wine, grape, winery, location,
wine color, wine body, wine flavor, sugar
content*

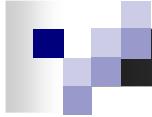
*white wine, red wine, Bordeaux wine
food, seafood, fish, meat, vegetables,
cheese*



Osztályhierarchia definiálása



- Egy osztály egy fogalmat jelöl
 - Borok osztálya
 - Borászatok osztálya
 - Vörös borok osztálya
- Egy osztály hasonló tulajdonságú elemek gyűjteménye
- Osztály példányok
 - Egy pohár burgundi vörösbor

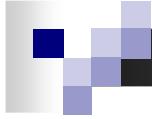


Öröklődés osztályok között

- Taxonómiai hierarchiába rendeződnek az osztályok (subclass-superclass)
- IS-A hierarchia:

Egy példány tekinthető egy alosztálynak

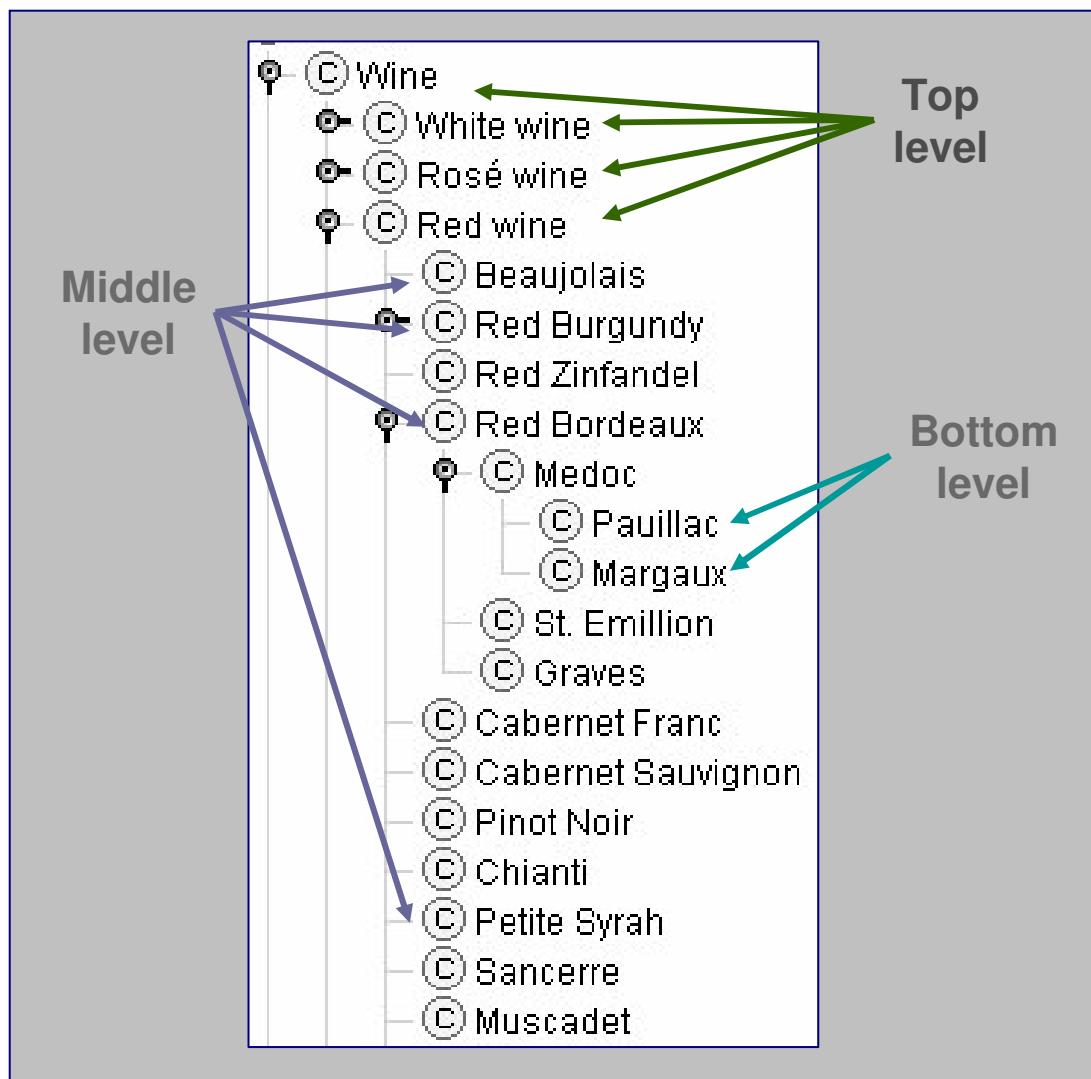
- Az osztályokat halmazokként is definiálhatjuk, egy alosztály egy részhalmaz

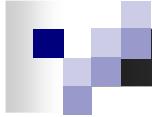


Öröklődés - Példa

- Az alma a gyümölcsök egy alosztálya
Every apple is a fruit
- Vörös borok a borok alosztálya
Every red wine is a wine
- Chianti bor a vörös borok egy alosztálya
Every Chianti wine is a red wine

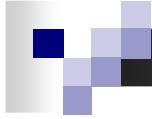
Hierarchia szintek





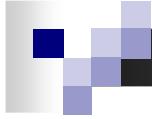
Fejlesztési megközelítések

- top-down – általános elemeket definiálunk először és aztán specializáljuk ezeket
- bottom-up – a legspecifikusabb fogalmakat definiáljuk és aztán megkeressük az összetartozásokat
- kombinált – a legtipikusabb elemeket megkeressük, ezt általánosítjuk, majd ezek speciális példányait is bevesszük



Dokumentáció

- Osztályokat (és slot-okat) rendszerint dokumentáljuk
 - Osztály leírása természetes nyelven
 - Tárgyterületi feltételezések felsorolása
 - Szinonímák felsorolása
- Az osztályok és slot-ok dokumentálása hasonlóan fontos mint a programkód kommentezése!

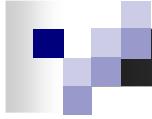


Osztályok tuéajdonságainak definiálása –slot-ok



Az osztályok slot-jai a példányok attribútumait és a példányok relációit írják le.

Each wine will have color, sugar content, producer, etc.



Tulajdonságok (Slot)

■ Tulajdonság típusok

- “belső” tulajdonságok: bor zamata és színe
- “külső” tulajdonságok: a bor neve ás ára
- részek: egy étel össztevői
- Más objektumokhoz kapcsolatok: a bor készítője (borászat)

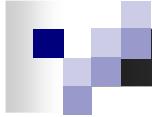
■ Egyszerű és összetett tulajdonságok

- Egyszerű tulajdonságok (attribútumok): egyszerű értékeket tartalmaznak (stringek, számok)
- Összetett tulajdonságok: tartalmaznak (vagy hivatkoznak) más objektumokat

A bor osztály attribútumai

Template Slots				Y	M	C	X	+	-
Name	Type	Cardinality		Other Facets					
S body	Symbol	single		allowed-values={FULL,MEDIUM,LIGHT}					
S color	Symbol	single		allowed-values={RED,ROSÉ,WHITE}					
S flavor	Symbol	single		allowed-values={DELICATE,MODERATE,STRONG}					
S grape	Instance	multiple		classes={Wine grape}					
S maker ^I	Instance	single		classes={Winery}					
S name	String	single							
S sugar	Symbol	single		allowed-values={DRY,SWEET,OFF-DRY}					

Protégé-2000)



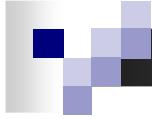
Attribútum és osztály öröklődés

- Egy alosztály megörököl minden attribútumot

Ha a bornak van színe és zamata, akkor a vörösbornak is van színe és zamata.

- Ha több szülőosztálya van egy osztálynak,akkor megörökli mindegyik szülő tulajdonságait

Egy portói bor desszert bor is és vörös bor is.



Tulajdonság kényszerek



- Tulajdonság kényszerek (facets) leírják vagy korlátozzák a felvehető értékek körét

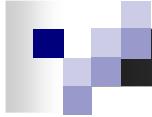
A bor neve egy string

A bor gyártója egy példánya a Borászat osztálynak

Egy borászat pontosan egy helyhez tartozik

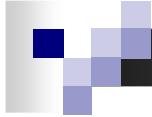
Kényszerek a bor osztályhoz

Template Slots				V	V _c	C	X	+	-
Name	Type	Cardinality	Other Facets						
S body	Symbol	single	allowed-values={FULL,MEDIUM,LIGHT}						
S color	Symbol	single	allowed-values={RED,ROSÉ,WHITE}						
S flavor	Symbol	single	allowed-values={DELICATE,MODERATE,STRONG}						
S grape	Instance	multiple	classes={Wine grape}						
S maker ^I	Instance	single	classes={Winery}						
S name	String	single							
S sugar	Symbol	single	allowed-values={DRY,SWEET,OFF-DRY}						



Tipikus kényszerek

- Slot számosság – a felvehető értékek száma
- Slot érték típus – a felvehető érték típusa
- Minimum és maximum értékek – értékkészlet
- Alapértelmezett érték – a slot által felvett érték, ha nem definiálják máshogy.



Tipikus kényszerek: Slot számosság

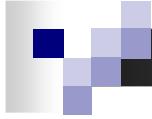
■ Számosság

- Az N számosság azt jelzi, hogy a slot-nak N értéke kell, hogy legyen

■ Minimum számosság

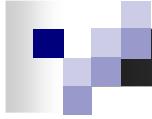
- Általában az 1 érték azt jelzi, hogy kell, hogy legyen egy értéke
- A 0 érték azt jelzi, hogy a minimum számosság tetszőleges

■ Maximum számosság



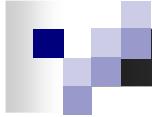
Tipikus kényszerek: Érték típus

- String: “Château Lafite”
- Szám: 15, 4.5
- Boolean: igaz vagy hamis
- Enumeráció: megengedett értékek lista (magas, közepes, alacsony)
- Komplex típusok: egy másik osztály példánya
 - Meg kell adni az osztályt, amihez a példány tartozik
A bot osztály gyártó slot-ja a borászatok közül veszi fel értékét



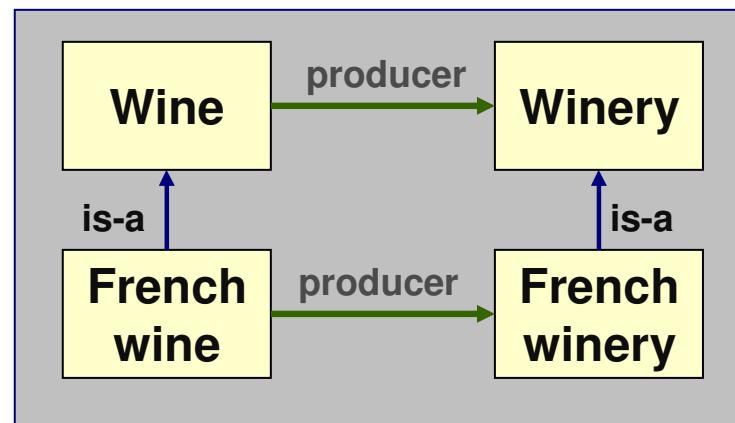
Tárgyterület és értékészlet

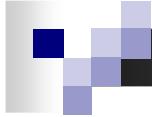
- Tárgyterület (Domain) – az osztály(ok), mi tartalmazza a tulajdonságot
 - Pontosabban: osztálypéldányok, amik felvehetik az értéket
- Értékkészlet (Range)– az osztály, amhez a rekesz tartozik



Kényszerek és öröklődés

- Egy alosztály örökli a tulajdonságot a szülőosztálytól
- Egy alosztály felülírhatja a kényszereket, hogy szűkítse az értékkészletet



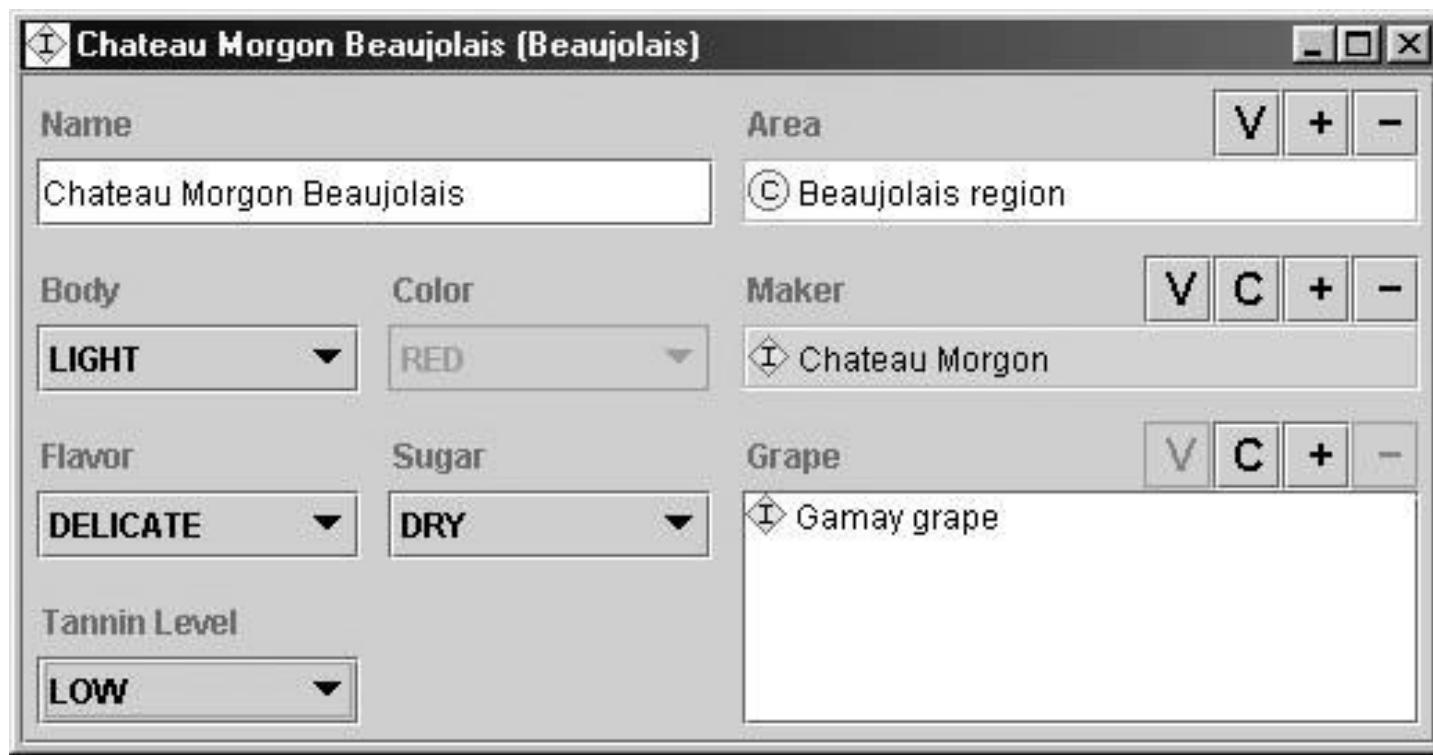


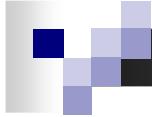
Példányok létrehozása



- Egy osztály példányának létrehozása
 - Az osztály és bármely szülőosztály a példány közvetlen típusa lesz
- Attribútum értékek meghatározása a példányokhoz
 - Az értékeknek teljesíteniük kell a rájuk vonatkozó ékényszereket

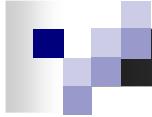
Egy példány létrehozása





Vázlat

- Mi is az ontológia?
- Miért fejlesszünk ontológiákat?
- Ontológiák építése lépésről lépésre
- Alaposabb elemzés: problémák és megoldások
- Ontológiák a szemantikus web eszközeivel
- Ontológiák fejlesztésének jelenlegi irányai



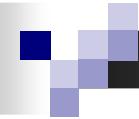
Osztályok és hierarchiák tervezése

- Fontos:

- Nincs egy egyetlen helyes osztályozása
 - De vannak iránymutatások

- Ellenőrző kérdés:

“Egy osztály minden példánya tagja-e az összes szülőosztálynak?”



Az osztályhierarchia tranzitivitása

■ Az is-a kapcsolat tranzitív:

B is-a A

C is-a B

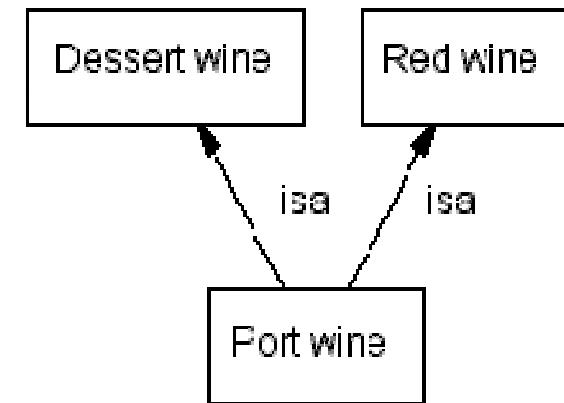
C is-a A

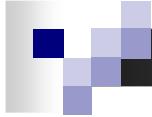
■ Egy közvetlen szülőosztály a legközelebbi szülőosztály



Többszörös öröklődés

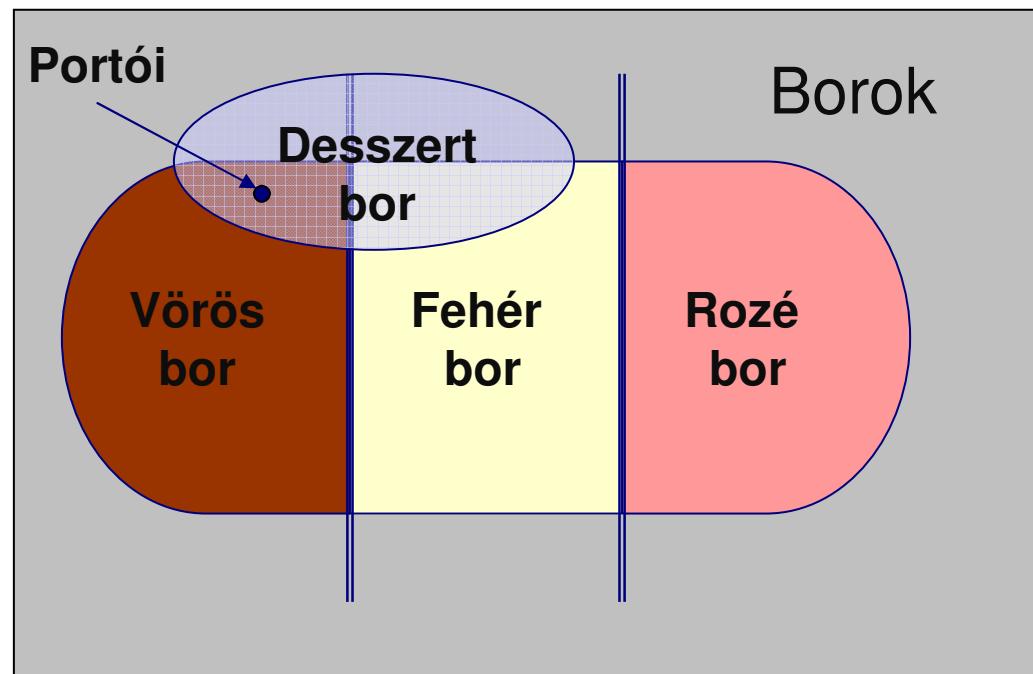
- Egy osztálynak több mint egy szülőosztálya létezhet
- A konfliktusok feloldása nehéz probléma



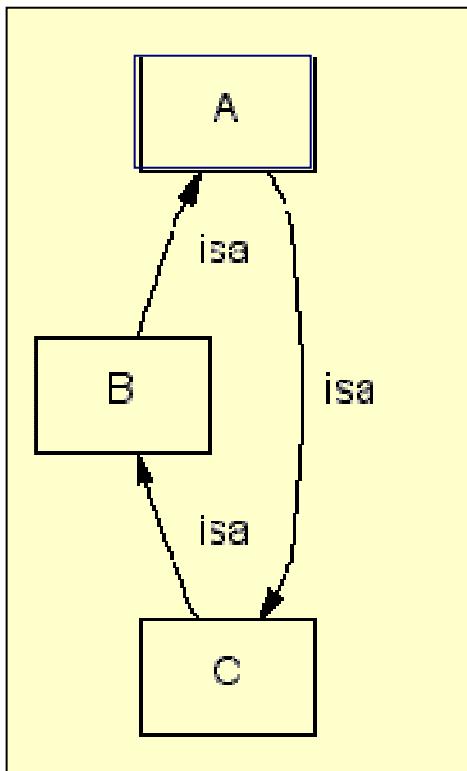


IKizáró osztályok

- Osztályok kizáják egymást, ha nincs közös példányuk
- Kizáró osztályoknak nem lehet közös alosztályuk sem



Ciklusok elkerülése az osztályhierarcíában



- Többszörös öröklődés veszélye: ciklusok az osztályhierarciában
- Az ábra A, B és C osztályai elvileg ekvivalensek, mert ugyanazok a példányai

Kedvencek az osztályhierarchiában



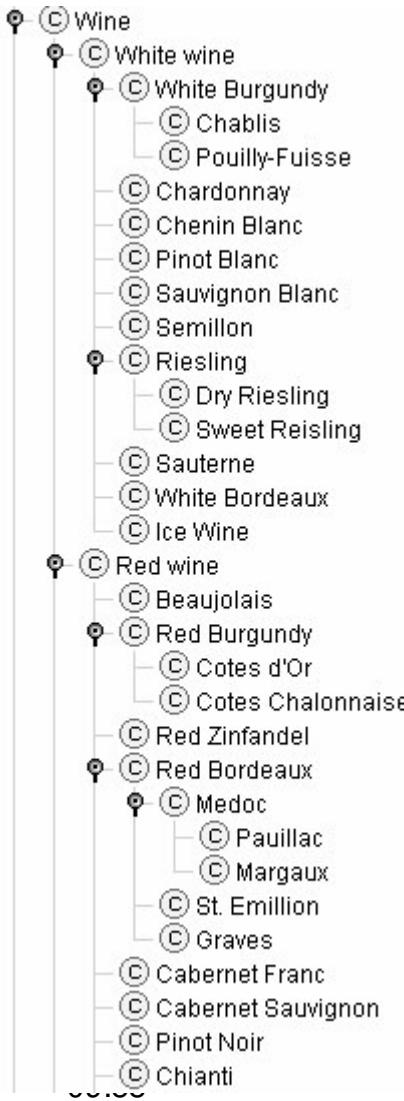
- minden azonos szintű fogalomnak egy hierarchia szinten kell lennie

Ideális méret megválasztása

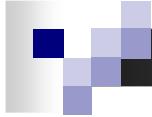


- Ha egy osztálynak egy gyereke van az valószínüleg helytelen modellezése
- Ha csak egy példány van, akkor minek neki osztály?

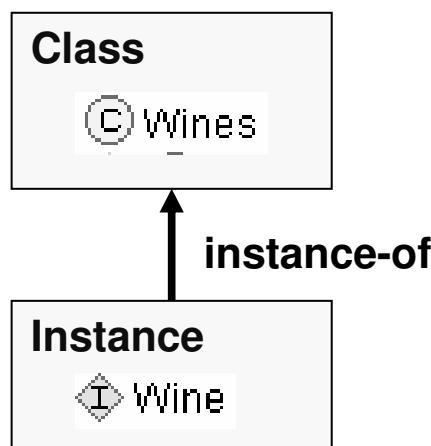
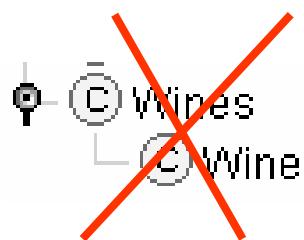
Ideális méret megválasztása (II)



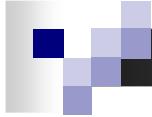
- Ökölszabály: ha egy osztálynak egy tucatnál több alosztálya van érdemes egy közébúlső szintet felvenni
- Lehet azonban egy hosszú lista természetes is.



Azonos osztálynevek problémája



- A bor ne legyen egyfajta bor is
- Esetleg lehet példány, de az sem szerencsés
- Osztálynevek lehetnek
 - egyszeresek
 - többszörösek

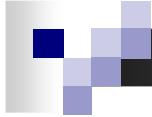


Osztályok és neveik

- Az osztályok fogalmakat reprezentálnak és nem a nevüket
- Az osztálynév akár változhat is egy fogalomhoz
- Synonym names for the same concept are not different classes
- Sok rendszer megenged szinonim nevek felvitelét

Egy teljes hierarchia a borokhoz



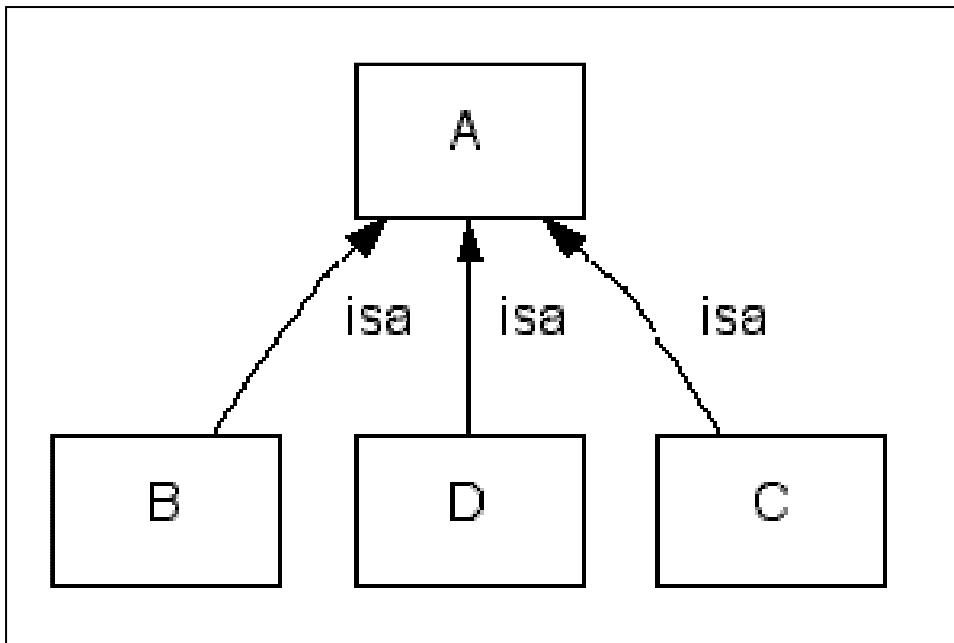


Attribútumok: Tárgyterület és értékkészlet

- A tárgyterület és értékkészlet meghatározásához találjuk meg a legáltalánosabb szülőosztályt
- Attribútum jellegének megérzése
 - Tárgyterület: Vörös bor, fehér bor
 - Tárgyterület: bor
- Például a termék slot-ra:
 - Értékkészlet: Vörös bor, Fehér bor, ...,
 - Értékkészlet: Bor



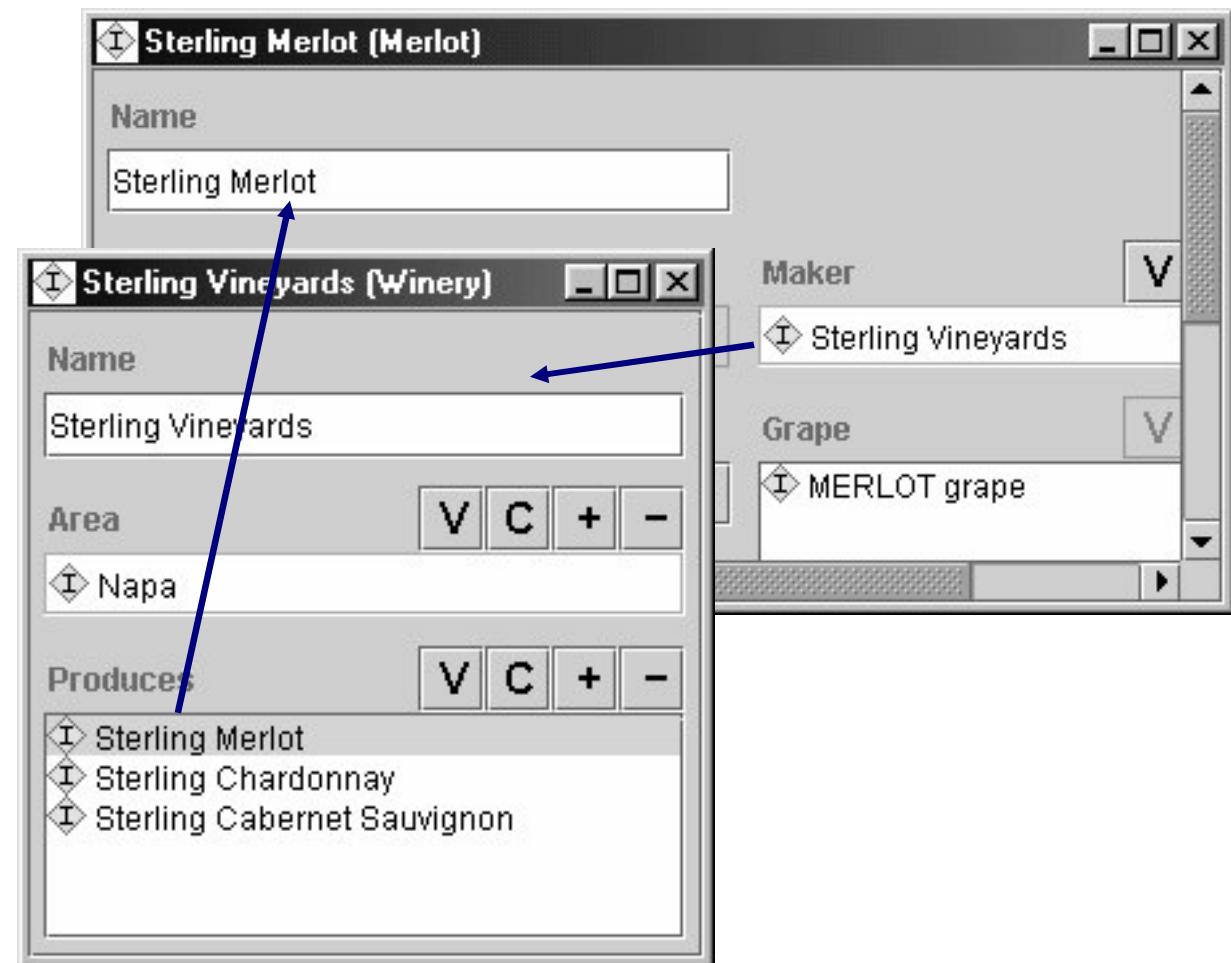
Tárgyterület és értékkészlet meghatározása

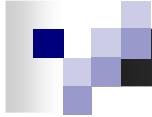


- Ha ugyanaz az osztályra és szülőjénél, akkor helyettesítsük a szülővel

Inverz attribútumok

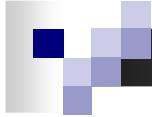
A gyártó például
egy inver slot.





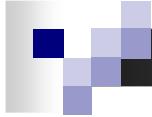
Inverz slot-ok (II)

- Az inverze slot-ok redundáns információt tartalmaznak
 - Többirányú információ elérést engednek meg
 - További ellenőrzést tesznek lehetővé
- A megvalósítás eltérő a rendszerekben
 - Mindkét értéket tároljuk?
 - Az inverz értéket mikor töltük fel?
 - Mi történik, ha az inverz érték valamelyik kapcsolatát változtajuk?



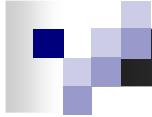
Alapértelmezett értékek

- A létrehozáskor érték
- Változtatható
- Nem szükséges, de általában tipikus



A vizsgált terület korlátozása

- Az ontológiának nem szabad tartalmaznia minden tárgyterületi információt
 - Az adott alkalmazás határozza meg az általánosítás és a specializáció szintjét
 - Nincs szükség minden tulajdonság felsorolásáá
 - Alapvető információk
 - Alkalmazás által igényelt speciális információk



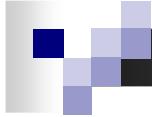
A vizsgált terület korlátozása(ii)

- A példa tárgyterület valószínűleg nem tartalmaz
 - Üveg méret
 - Címke jellemzése
 - Az én kedvenc borom
- Biológiai vizsgálatokról szóló ontológia tartalmaz:
 - Biológiai organizmusok
 - Vizsgálat végzője
- A vizsgálat végzője egy biológiai organizmus?



Szemantikus web nyelvek ontológiák építéséhez

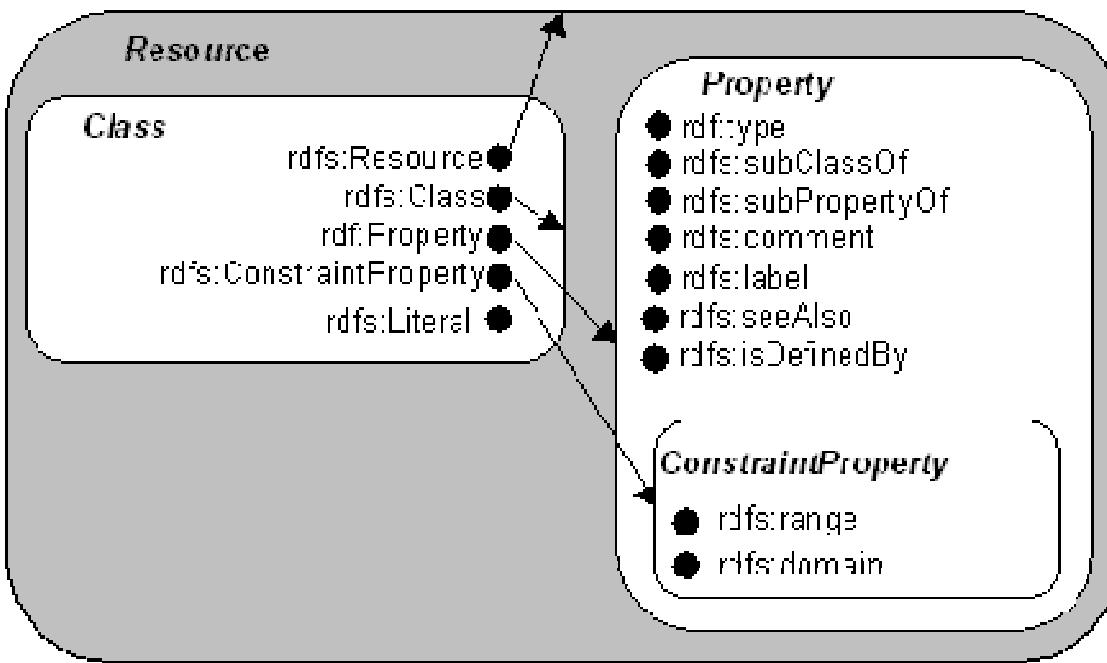
- A szemantikus web nyelvek kifejezetten ontológiák leírására lettek tervezve
 - RDF Schema
 - DAML+OIL
 - SHOE
 - XOL
 - XML Schema



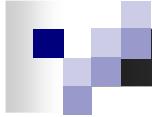
SzW nyelvek

- A nyelvek különböznek
 - szintakszis
 - (nem elsődleges kérdés)
 - terminológia
 - Osztály - koncepció
 - Példány - objektum
 - Slot- tulajdonság
 - kifejezőerő
 - Van amit az egyik nyelvben ki tudunk fejezni, míg a másikban nem
 - szemantika
 - Ugyanaz az állítás más jelentést hordozhat

RDF és RDF Séma osztályok

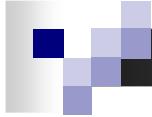


RDF Schema Specification 1.0 (<http://www.w3.org/TR/2000/CR-rdf-schema-20000327/>)



RDF(S) terminológia és szemantika

- Osztályok és osztályhierarchiák
 - minden osztály leszármazottja az rdfs:Class osztálynak
 - Osztályhierarchia: rdfs:subClassOf
- Példányok
 - Definíció: rdf:type
- Tulajdonságok
 - A tulajdonságok globálisak:
Közös névterben a különböző osztályok azonos nevű tulajdonságai
azonos jelentéssel bírnak
 - Tulajdonság-hierarchiák (rdfs:subPropertyOf)



Tulajdonság kényszerek RDF(S)

■ Számosság kényszerek

- Nincs explicit számosság kényszer
- Bármely tulajdonság több értéket vehet fel

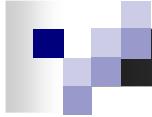
■ Tulajdonság értékkészlete

- Egy értékkészlet lehet

■ Tulajdonság tárgyterülete

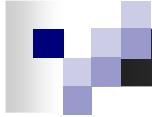
- Több tárgyterülethez tartozhat

■ Nincsenek kezdeti értékek



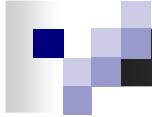
Kutatási irányok az ontológiák fejlesztésével kapcsolatosan

- Tartalom generálás
- Analízis és kiértékelés
- Karbantartás
- Ontológia nyelvek
- Eszközök fejlesztése



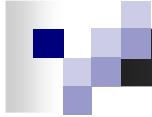
Tartalon: Top-Level Ontológiák

- Mit jelent a “felső-szint”?
 - Objektumok: érzékelhetők, nem érzékelhetők
 - Folyamatok, események, szereplők, szerepek
 - Ágensek, szerveződések
 - Terek, korlátok, helyek
 - Idő
- IEEE Standard Upper Ontology próbálkozás
 - Cél: Tervezni upper-level ontology-t a mérnöki terület igényei szerint
 - Folyamat: Meglévő ontológiák összeválogatásával



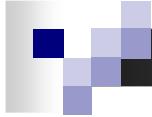
Tartalom: Tudásmegszerzés

- Tudás összegyűjtése a szűk keresztmetszet
- Megosztás és újrafelhasználhatóság segít a problémán
- Automatikus tudáskinyerési módszerre is szükség volna:
 - Nyelvtechnológia: ontológia kinyerése szövegekből
 - Gépi tanulás: ontológiák létrehozása strukturált szövegekből (pl. XML dokumentumok)
 - A Web struktúra kibővítése: ontológiák létrehozása Web portálok átolvasásával
 - Tudás kinyerési sémák: a szakértők definiálják egy részét az ismerteknek



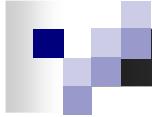
Analízis

- Analízis: szemantika konzisztenciája
 - Tulajdonság kényszerek megsértése
 - Ciklusok az osztályhierarchiában
 - Nem definiált, de használt kifejezések
 - Intervallum korlátok megsértése
(min > max)
- Analízis: stílus
 - Osztályok egyetlen alosztállyal
 - Osztályok és slotok definíció nélkül
 - Slot-ok kényszerek nélkül (típus, számosság)
- Automatikus analízis eszközök
 - Chimaera (Stanford KSL)
 - DAML Validator



Kiértékelés

- Egyik legbonyolultabb feladat az ontológia tervezésben
- Az ontológia tervezés szubjektív
- Mikor mondhatjuk, hogy egy ontológia helyes (objektívan)?
- A legjobb teszt az alkalmazás építése



Ontológia karbantartás

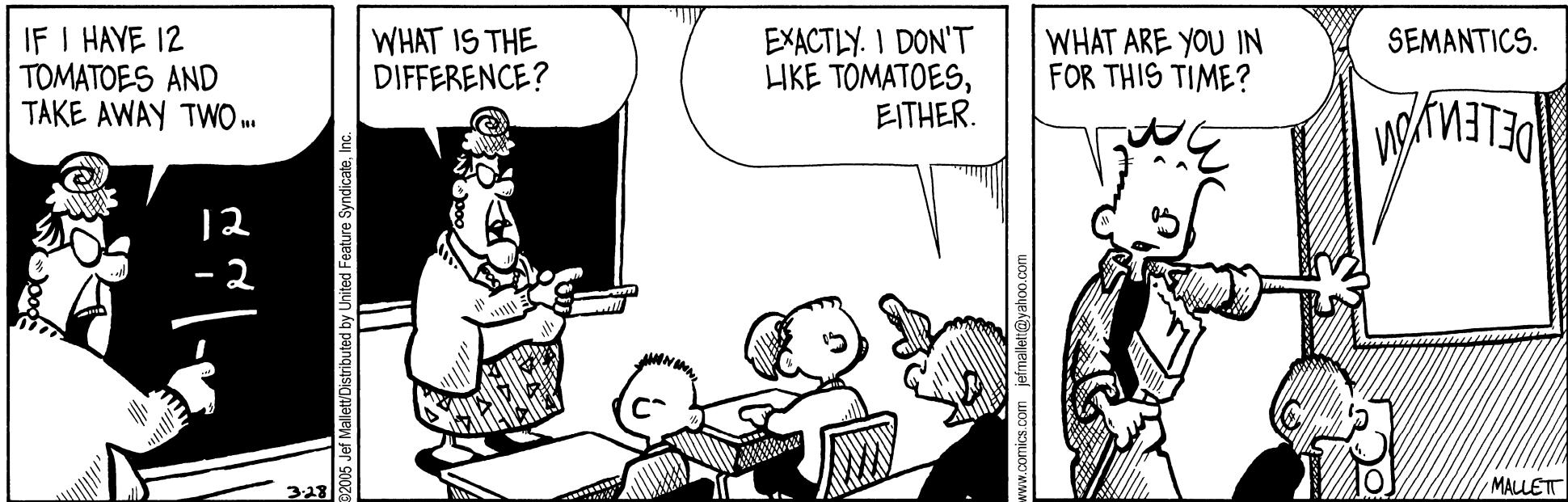
- Ontológiák összeillesztése
 - Két vagy több átfedő ontológia összerakása
- Ontology leképezés
 - Ontológiák közötti leképezés építése
- Verziókezelés
 - Kompatibilitás biztosítása a különböző verziók között
 - Kompatibilitás biztosítása a verziók és a példányok között

Integrációs és ellenőrzési technikák
VIMIAC04, 2021. tavasz

ONTOLÓGIÁK, OWL₂, DL

Méréstechnika és Információs Rendszerek Tanszék

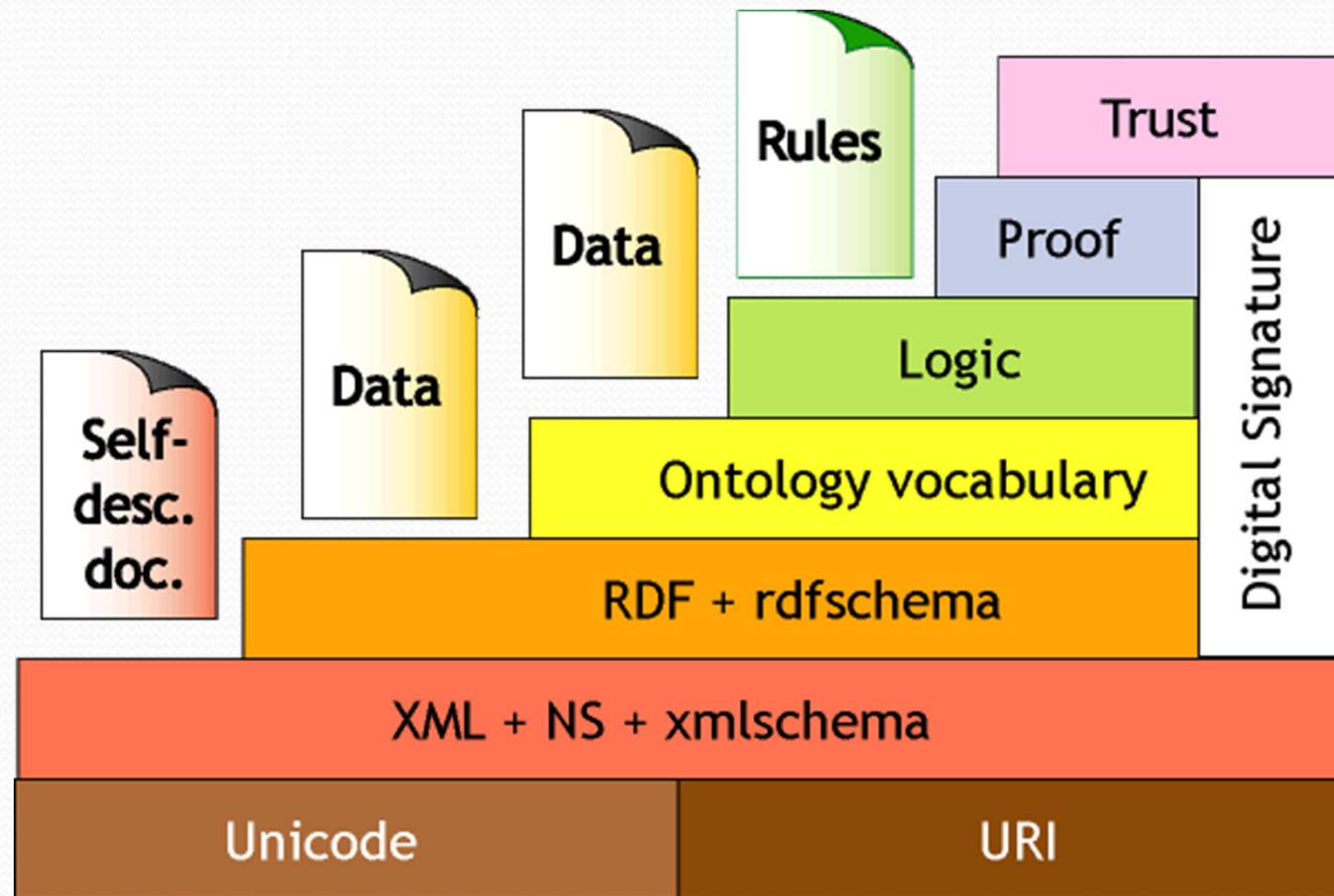
Szemanika



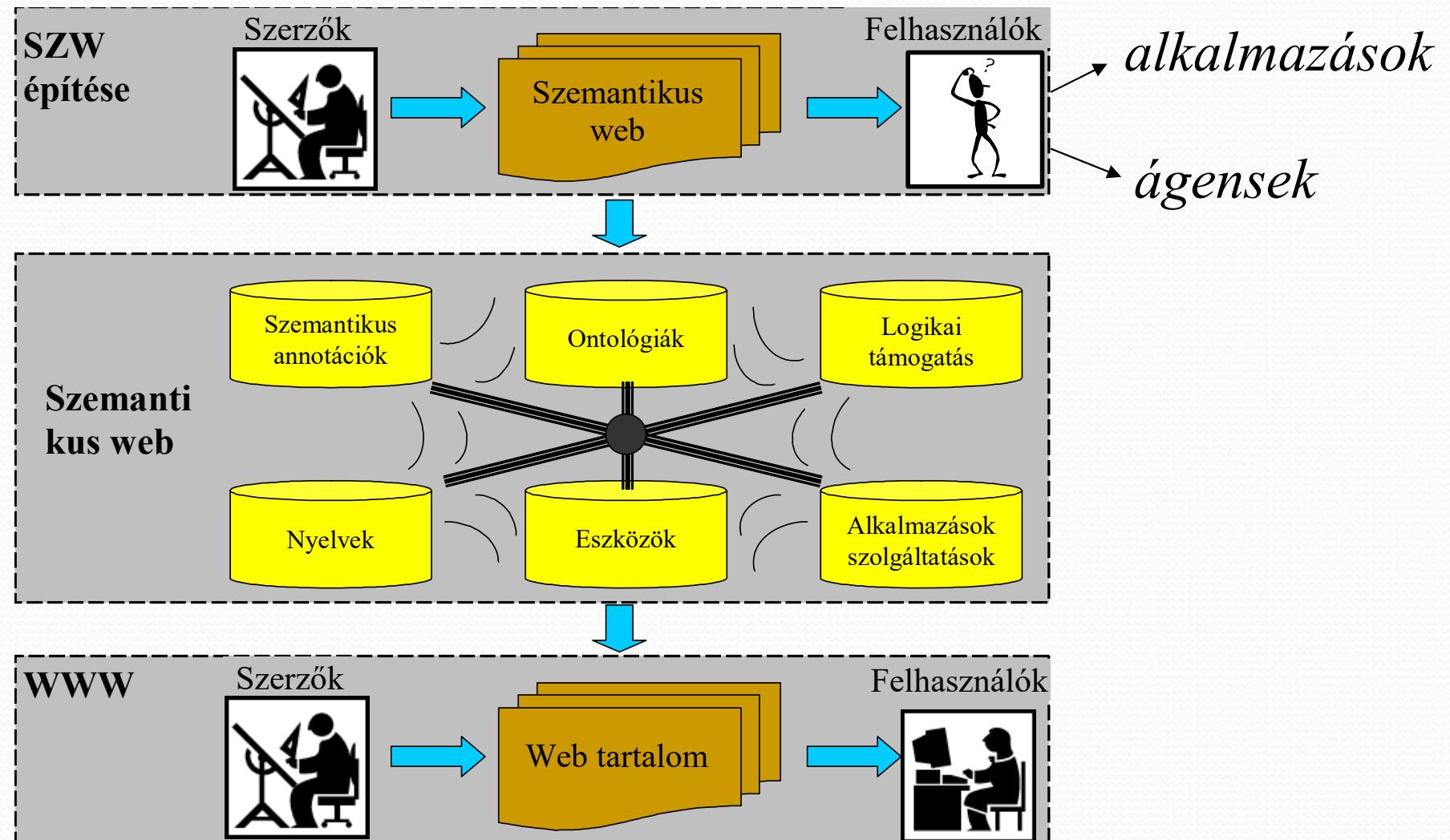
A szemantikus web koncepció

"The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation."

-- Tim Berners-Lee



Szemantikus web – új felhasználók



Ontológiák, az OWL nyelv

Szemantikus web példa

- Legyen ismert ez az állítás:
 - Budapest Magyarország fővárosa
- Háttértudásunk segítségével levezethetjük:
 - Budapest Magyarország fővárosa ✓
 - Magyarország egy állam ✓
 - Budapest egy város ✓
 - Budapest Magyarországon van ✓
 - Debrecen nem Magyarország fővárosa ✗
 - Budapest nem Ausztria fővárosa ✗
 - Budapest nem egy állam ✗
 - ...

Szemantikus web példa

- RDFS szintig nem tudjuk leírni a következőket:
 - minden államnak pontosan egy fővárosa van
 - Tulajdonság számosság
 - minden város csak egy ország fővárosa lehet
 - Függvény tulajdonság
 - egy város nem lehet egyben egy állam is.
 - Diszjunkt osztályok
 - ...

OWL elemek, példák

- Osztályok
 - Person osztály
 - Man, Woman alosztály
- Tulajdonságok (egyedi)
 - isWifeOf, isHusbandOf
- Tulajdonság jellemzők, korlátok
 - inverseOf
 - domain
 - range
 - cardinality
- Osztályok közti relációk
 - disjointWith

Ontológiák

- RDFS hasznos, de nem ad megoldást a szemantika pontos leírására.
- Összetett alkalmazások további igényei:
 - Tulajdonságok leírása, jellemzése
 - Különböző URI-val rendelkező objektumok azonosságának leírása (ekvivalencia)
 - Osztályok diszjunkt vagy éppen ekvivalens jellege
 - Osztályok konstruálása (nemcsak megnevezése)
 - Következtetési igények támogatása:
 - Pl.: "Ha két «Person» erőforrás «A» és «B» azonos «**foaf:email**» tulajdonsággal rendelkeznek, akkor «A» és «B» identikus.

Ontológiák

- Az SZW világban az ontológiákat a következő értelemben használjuk:

Fogalmak és ezek relációnak definiálása
egy adott tudásterület leírása céljából.

- Az RDFS is tekinthető egy egyszerű ontológia nyelvnek
- Nyelvek definiálása mindig egyfajta kompromisszum
 - gazdag szemantika tudás gazdag alkalmazásokhoz
 - ésszerűség (fizibilitás, megvalósíthatóság, fordító, következtetőgép)

Web Ontology Language = OWL

- OWL az SZW struktúrába egy újabb réteg, az RDFS bővítése
 - Saját névterek, saját elemek, kifejezések
 - RDFS-re épül (tartalmazza)
- Önálló SZW ajánlás
 - “OWL 2” – 2010 óta gyakorlatilag csak ezt használjuk
 - Akit a részletek érdekelnek:
<https://www.w3.org/TR/owl2-overview/>

Ekvivalencia relációk

- Osztályokra:
 - **owl:equivalentClass**: két osztálynak azonosak az elemei
 - **owl:disjointWith**: nincs közös elemük
- Tulajdonságokra:
 - **owl:equivalentProperty**
 - Példa: **a:author** vs. **f:auteur**
 - **owl:propertyDisjointWith**
- Egyedekre:
 - **owl:sameAs**: két URI ugyanazt a fogalmat vagy egyedet reprezentálja
 - **owl:differentFrom**: negált kifejezése az **owl:sameAs** kifejezésnek

Osztályok az OWL-ben

- RDFS: létező osztályokat alosztály struktúrába rendezhettük – osztályhierarchia építése, semmi több...
- OWL osztályok konstruálhatók más osztályok vagy példányok alapján, :
 - Elemek felsorolásával
 - Osztályok relációinak alkalmazásával: metszet, unió, komplemens, stb.

Eddig...

- OWL: erős leíró elemeket definiáltunk
- pl., adatbázisok összeköthetőek **owl:sameAs**, vagy inverse functional tulajdonságokkal.
- Számos kapcsolatot felderíthetünk hagyományos következtetési eljárásokkal

OWL 2

- Korlátozások definiáltak
 - classes, individuals, object , datatype properties - megkötésekkel
 - object properties csak individuals -re
 - *datatype* property nincs tovább specifikálva
 - ...
- Hatékony következtető algoritmus létezik!

OWL 2

- Korlátozások megadásával nagy méretű ontológiák építhetők,
és alkalmazhatóak pl. orvosi, robotika, biológia
tárgyterületeken
- OWL 2 lett a formális ontológiák nyelve
 - Nem feltétlenül a weben használjuk

OWL alapú ontológia

- "Debrecen nem Magyarország fővárosa ✗
- Miért nem?
 - Országoknak pontosan egy fővárosa van,
 - Debrecen és Budapest nem ugyanaz a város

- OWL:

```
:capitalOf a owl:InverseFunctionalProperty .  
:Budapest :capitalOf :Magyarország .  
:Budapest owl:differentFrom :Debrecen .
```

Lekérdezés { :Debrecen :capitalOf :Magyarország . } → false

OWL alapú ontológia

- Budapest nem Ausztria fővárosa ✗
- Miért nem?
 - Egy város csak egy ország fővárosa lehet
 - Ausztria és Magyarország nem azonos

- Hasonlóan:

```
:capitalOf a owl:FunctionalProperty .  
:Budapest :capitalOf :Magyarország .  
:Ausztria owl:differentFrom :Magyarország .
```

Lekérdezés { :Budapest :capitalOf :Ausztria . } → false

Következtetés OWL DL nyelvben

- RDFS következtetés
 - Előrefele láncolás
 - Új axiómák vezetése ismertek alapján
- OWL DL következtetés bonyolultabb:
 - Előrefele láncolás nem hatékony nagy adatbázisokon
 - Konjunkció (pl., unionOf) nem támogatott
 - Más megközelítés: Tableau következtetés
 - Alapgondolat: találunk ellentmondást az ontológiában
 - Egy állítás és negáltja is szerepel

Következtetés OWL DL nyelvben

- Mit várunk a következtetőgéptől?
 - Alosztály definiálás
 - Pl. minden madár repülő állat?
 - Ekvivalens osztályok
 - Diszjunkt osztályok
 - Pl. lehet-e egy állat egyszerre emlős és madár is?
 - Osztály konzisztencia
 - Pl. Emlősök szaporodhatnak-e tojással?
 - Példányok kategorizálása
 - Osztályok tagjainak felsorolás

Példa: Egy egyszerű ellentmondás

- Adott:

:Man a owl:Class .

:Woman a owl:Class .

:Man owl:disjointWith :Woman .

:Alex a :Man .

:Alex a :Woman .

Példa: Egy egyszerű ellentmondás

- Levezethető:
 - $:Man \cap :Woman = \emptyset$
`owl:Nothing owl:intersectionOf (:Man :Woman) .`
 - $:Alex \in (:Man \cap :Woman)$
`:Alex a [a owl:Class; owl:intersectionOf (:Man :Woman)] .`
- Pl.:
 - $:Alex \in \emptyset$
`:Alex a owl:Nothing .`
 - ***Tehát a példány nem létezhetne, de van!***

Érvelési feladatok

- Alosztály relációk

Például: $\text{Student} \subseteq \text{Person} \Leftrightarrow \text{"Every student is a person"}$
- Bizonyítási módszer: Reductio ad absurdum
 - "Hozzunk létre" egy i példányt
 - Definiáljuk: $\text{Student}(i)$ és $\neg\text{Person}(i)$
 - Ellenőrizzük az ellentmondás-mentességet
 - Ha létezik ilyen: $\text{Student} \subseteq \text{Person}$ igaz
 - Ha nem létezik: $\text{Student} \subseteq \text{Person}$ nem levezethető
 - (Ettől még lehet igaz)

Példa: alosztály relációk

- **Ontológia:**

```
:Student owl:subClassOf :UniversityMember .  
:UniversityMember owl:subClassOf :Person .
```

- **Bevezetett példányok:**

```
:i a :Student .  
:i a [ owl:complementOf :Person ] .
```

- **Így adott most:**

```
:i a :Student .  
:Student owl:subClassOf :UniversityMember .
```

Ezáltal:

```
:i a :UniversityMember .
```

- .. és ez is igaz:

```
:UniversityMember owl:subClassOf :Person .
```

- **Ebből pedig levezethető:**

```
:i a Person .
```

Példa: alosztály relációk

- Tehát ismerjük:

```
:i a :Person .
```

```
:i a [ owl:complementOf :Person ] .
```

Pl.,

```
:i a [ owl:intersectionOf (:Person  
[ owl:complementOf :Person  
]) ] .
```

- Amiből következik, hogy:

```
:i a owl:Nothing .
```

Érvelési feladatok

- Osztály ekvivalencia
 - Person \equiv Human
- Részfeladatra bontható:
 - Person \subseteq Human and
 - Human \subseteq Person

Tehát két alosztály definíció igazolása a feladat
- Osztályok diszjunktsága
 - C és D diszjunkt osztályok?
 - Definiáljunk példányokat: C(i) és D(i)
 - - Ellentmondás vizsgálat: létezik-e ilyen i

Osztály konzisztencia vizsgálata

- Léteznek-e az osztályhoz tartozó példányok?

- Pl., női aggregények:

```
:Bachelor owl:subClassOf :Man .
```

```
:Bachelor owl:subClassOf
[ a owl:Restriction;
owl:onProperty :marriedTo;
owl:cardinality 0 ] .
```

```
:MarriedPerson owl:subClassOf [
a owl:Restriction;
owl:onProperty :marriedTo;
owl:cardinality 1 ] .
```

```
:MarriedBachelor owl:intersectionOf
(:Bachelor :MarriedPerson) .
```

- Ezután definiálunk egy példányt, és ellenőrizzük, hogy van-e ellentmondás

Érvelési feladatok

- Példányok osztályba tartozása
 - Flipper egy delfin?
- Vizsgáljuk:
 - define `Dolphin(Flipper)`
 - Vizsgálni a konzisztenciát
- Osztály felsorolás
 - Ismételni az osztályba tartozás vizsgálatot minden ismert példányra.

Tipikus következtetési feladatok (összefoglalás)

- Mit várunk egy következtetőtől?
 - Alosztály relációk
 - pl., minden madár repülő állat?
 - Ekvivalens osztályok
 - pl., minden madár állat, és fordítva is igaz?
 - Diszjunkt osztályok
 - Pl., létezik-e olyan állat, amely emlős és madár egyszerre?
 - Osztály konzisztencia
 - Pl.: létezik olyan emlős, amely tojásokat rak?
 - Osztályok példányai
 - Pl.: Flipper egy delfin?
 - Osztályok felsorolással
 - Pl.: adjuk meg az összes delfint

Tipikus következtetési feladatok (összefoglalás)

- Eddig láttuk:
 - minden következtetési feladat levezethető néhány alap következtetési feladatra, pl. szubsumáció, konzisztencia ellenőrzés
- Tehát építsünk egy következtetőt konzisztencia ellenőrzésre

Leíró logikai notáció

- Osztályok és példányok
 - $C(x)$ $\leftrightarrow x \text{ a } C .$
 - $R(x,y)$ $\leftrightarrow x \text{ R } y .$
 - $C \sqsubseteq D$ $\leftrightarrow C \text{ rdfs:subClassOf } D$
 - $C \equiv D$ $\leftrightarrow C \text{ owl:equivalentClass } D$
 - $C \sqsubseteq \neg D$ $\leftrightarrow C \text{ owl:disjointWith } D$
 - $C \equiv \neg D$ $\leftrightarrow C \text{ owl:complementOf } D$
 - $C \equiv D \sqcap E$ $\leftrightarrow C \text{ owl:intersectionOf } (D \ E) .$
 - $C \equiv D \sqcup E$ $\leftrightarrow C \text{ owl:unionOf } (D \ E) .$
 - T — owl:Thing
 - \perp — owl:Nothing

Leíró logikai notáció

- Tárgyterület, értékkészlet, egyéb korlátozások

$$-\exists R.T \sqsubseteq C \qquad \qquad R \text{ rdfs:domain } C$$

Példa: \exists Apja.leánygyermek fogalom jelöli azon dolgokat, akik Apja szereben (relációban) vannak leánygyermekekkel, tehát a C= Apák osztályából veszi elemeit, azaz a tárgyterülete az Apák osztálya

$$-\forall R.C \qquad \qquad R \text{ rdfs:range } C$$

Példa: \forall Leányapa.leánygyermek fogalom jelöli azon dolgokat, akiknek LeányApa relációban vannak leánygyermekekkel, tehát a Leányapa szerep a leánygyermek osztályából veszi értékeit, azaz értékkészlete a leánygyermek osztály.

$$-\mathbf{C} \sqsubseteq \forall R.D \leftrightarrow C \qquad \text{owl:subClassOf} \\ [\text{ a owl:Restriction;} \\ \text{ owl:onProperty } R; \\ \text{ owl:allValuesFrom } D] .$$

$$-\mathbf{C} \sqsubseteq \exists R.D \leftrightarrow C \qquad \text{owl:subClassOf} \\ [\text{ a owl:Restriction;} \text{ owl:onProperty } \\ R; \text{ owl:someValuesFrom } D] .$$

$$-\mathbf{C} \sqsubseteq \geq n R \leftrightarrow C \qquad \text{owl:subClassOf} \\ [\text{ a owl:Restriction;} \text{ owl:onProperty } \\ R; \text{ owl:minCardinality } n] .$$

Negációs normál forma (NNF)

- Ontológiák átalakítása NNF-re
 - \sqsubseteq és \equiv nem használható
 - Negációt csak atomi formulákra és osztályokra alkalmazható
- Egyszerűbb notáció
- Tabló következtetés formalizmusa

Negációs normál forma (NNF)

- \sqsubseteq eliminálás:
 - Helyettesítsük $C \sqsubseteq D$ -t: $\neg C \sqcup D$
 - Rövidített notáció az ismert formára: $\forall x: \neg C(x) \vee D(x)$
- Miért
 - $C \sqsubseteq D$ ekvivalens $C(x) \rightarrow D(x)$

$C(x)$	$D(x)$	$C(x) \rightarrow D(x)$	$\neg C(x) \vee D(x)$
true	true	true	true
true	false	false	false
false	true	true	true
false	false	true	true

Negációs normál forma (NNF)

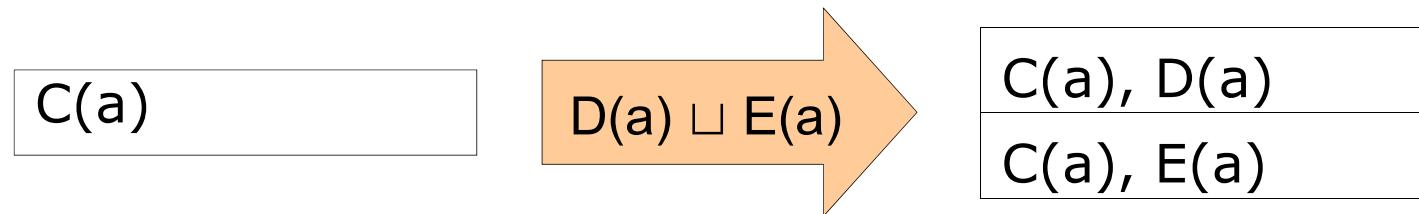
- Elimináljuk \equiv :
 - Helyettesítsük $C \equiv D$ –t: $C \sqsubseteq D$ és $D \sqsubseteq C$
 - Innen használható a korábbi megoldás
- Pl.: $C \equiv D$ átírva:
 - $C \sqsubseteq D$
 - $D \sqsubseteq C$
 - És így:
 - $\neg C \sqcup D$
 - $\neg D \sqcup C$

Negációs normál forma (NNF)

- További transzformációs szabályok:
 - $\text{NNF}(C) = C$ (for atomic C)
 - $\text{NNF}(\neg C) = \neg C$ (for atomic C)
 - $\text{NNF}(\neg \neg C) = C$
 - $\text{NNF}(C \sqcup D) = \text{NNF}(C) \sqcup \text{NNF}(D)$
 - $\text{NNF}(C \sqcap D) = \text{NNF}(C) \sqcap \text{NNF}(D)$
 - $\text{NNF}(\neg(C \sqcap D)) = \text{NNF}(\neg C) \sqcup \text{NNF}(\neg D)$
 - $\text{NNF}(\neg(C \sqcup D)) = \text{NNF}(\neg C) \sqcap \text{NNF}(\neg D)$
 - $\text{NNF}(\forall R.C) = \forall R.\text{NNF}(C)$
 - $\text{NNF}(\exists R.C) = \exists R.\text{NNF}(C)$
 - $\text{NNF}(\neg \forall R.C) = \exists R.\text{NNF}(\neg C)$
 - $\text{NNF}(\neg \exists R.C) = \forall R.\text{NNF}(\neg C)$

Tabló következtető (Tableau Algorithm)

- Tabló: Levezetett axiómák gyűjteménye
 - Folyamatosan bővítjük
 - Előrefele láncoláshoz hasonló eljárással
- Például konjunkcióra:
 - Bontsuk ketté a tablót:



Mikor ellentmondásmentes egy ontológia?

- Tablót bővítjük és bontjuk
- Nincs ellentmondás, ha
 - További axióma nem hozható létre
 - Legalább egy rész tabló ellentmondás mentes
 - Egy rész tabló tartalmaz ellentmondást, ha egy axióma és ellentettje is eleme:
 - Pl.. $\text{Person}(\text{Peter})$ és $\neg\text{Person}(\text{Peter})$
 - Ilyenkor a rész tablót lezárjuk.

Tabló következtető

- Adott: egy ontológia NNF formában
Amíg nem minden rész tabló lezárt
 - * Válasz egy nyitott tablót T és egy $A \in O \cup T$
 - Ha A-t nem tartalmazza T, akkor
 - Ha A atomi formula, akkor
 - adjuk A-t
 - T-hez és
 - vissza *
 - Ha A nem atomi formula, akkor
 - Válasszunk egy $i \in O \cup T$
 - példányt
 - Adjuk $A(i)$ -t T-hez
 - Vissza *
 - egyénként
 - Bővítsük a tablót A következményeivel és
 - vissza *

Tabló következtető

- Bővítsük a tablót a következményekkel:

Nr	Axiom	Action
1	$C(a)$	Add $C(a)$
2	$R(a,b)$	Add $R(a,b)$
3	C	Choose an individual a , add $C(a)$
4	$(C \sqcap D)(a)$	Add $C(a)$ and $D(a)$
5	$(C \sqcup D)(a)$	Split tableau into T_1 and T_2 . Add $C(a)$ to T_1 , $D(a)$ to T_2
6	$(\exists R.C)(a)$	Add $R(a,b)$ and $C(b)$ for a <i>new</i> Individual b
7	$(\forall R.C)(a)$	For all b with $R(a,b) \in T$: add $C(b)$

Példa

- Adott egy ontológia:
 :Animal owl:unionOf (:Mammal :Bird :Fish :Insect :Reptile) .
 :Animal owl:disjointWith :Human .
 :Seth a :Human .
 :Seth a :Insect .
- Konzisztens-e a tudásbázis?

Példa

- Adott a következő ontológia:
 :Animal owl:unionOf (:Mammal :Bird :Fish :Insect :Reptile) .
 :Animal owl:disjointWith :Human .
 :Seth a :Human .
 :Seth a :Insect .

– A fenti ontológia DL-NNF alakban:
 $\neg\text{Animal} \sqcup \neg\text{Human}$
 $\text{Animal} \sqcup (\neg\text{Mammal} \sqcap \neg\text{Bird} \sqcap \neg\text{Fish} \sqcap \neg\text{Insect} \sqcap \neg\text{Reptile})$
 $\neg\text{Animal} \sqcup (\text{Mammal} \sqcup \text{Bird} \sqcup \text{Fish} \sqcup \text{Insect} \sqcup \text{Reptile})$
 $\text{Human}(\text{Seth})$
 $\text{Insect}(\text{Seth})$
- Végezzük el a következtetést!

Példa

Human(Seth), Insect(Seth)

Nr	Axiom	Action
1	C(a)	Add C(a)

Példa

Human(Seth), Insect(Seth),
 $(\neg \text{Animal} \sqcup \neg \text{Human})(\text{Seth})$

Nr	Axiom	Action
3	C	Choose an individual a, add C(a)

Példa

Human(Seth), Insect(Seth),
 \neg Animal(Seth)

Human(Seth), Insect(Seth),
 \neg Human(Seth)

Nr	Axiom	Action
5	$(C \sqcup D)(a)$	Split the tableau into T1 and T2. Add C(a) to T1, D(a) to T2

Példa

Human(Seth), Insect(Seth),
 \neg Animal(Seth)

Animal \sqcup (\neg Mammal \sqcap \neg Bird \sqcap \neg Fish \sqcap \neg Insect)(Seth)

Human(Seth), Insect(Seth),
 \neg Human(Seth)

Nr	Axiom	Action
3	C	Choose an individual a, add C(a)

Példa

Human(Seth), Insect(Seth),
 \neg Animal(Seth)

Animal(Seth)

Human(Seth), Insect(Seth),
 \neg Animal(Seth)
 $(\neg$ Mammal \sqcap \neg Bird \sqcap \neg Fish \sqcap \neg Insect \sqcap \neg Reptile)(Seth)

Human(Seth), Insect(Seth),
 \neg Human(Seth)

Nr	Axiom	Action
5	$(C \sqcup D)(a)$	Split the tableau into T1 and T2. Add C(a) to T1, D(a) to T2

Példa

Human(Seth), Insect(Seth),
 \neg Animal(Seth)
Animal(Seth)

Human(Seth), Insect(Seth),
 \neg Animal(Seth)
 $(\neg$ Mammal $\sqcap \neg$ Bird $\sqcap \neg$ Fish $\sqcap \neg$ Insect $\sqcap \neg$ Reptile)(Seth)
 \neg Mammal(Seth) $\sqcap \neg$ Bird(Seth) $\sqcap \neg$ Fish(Seth) \sqcap
 \neg Insect(Seth) $\sqcap \neg$ Reptile(Seth)

Human(Seth), Insect(Seth),
 \neg Human(Seth)

Nr	Aussage	Aktion
4	$(C \sqcap D)(a)$	Add C(a) and D(a)

Integrációs és ellenőrzési technikák
VIMIAC04, 2021. tavasz

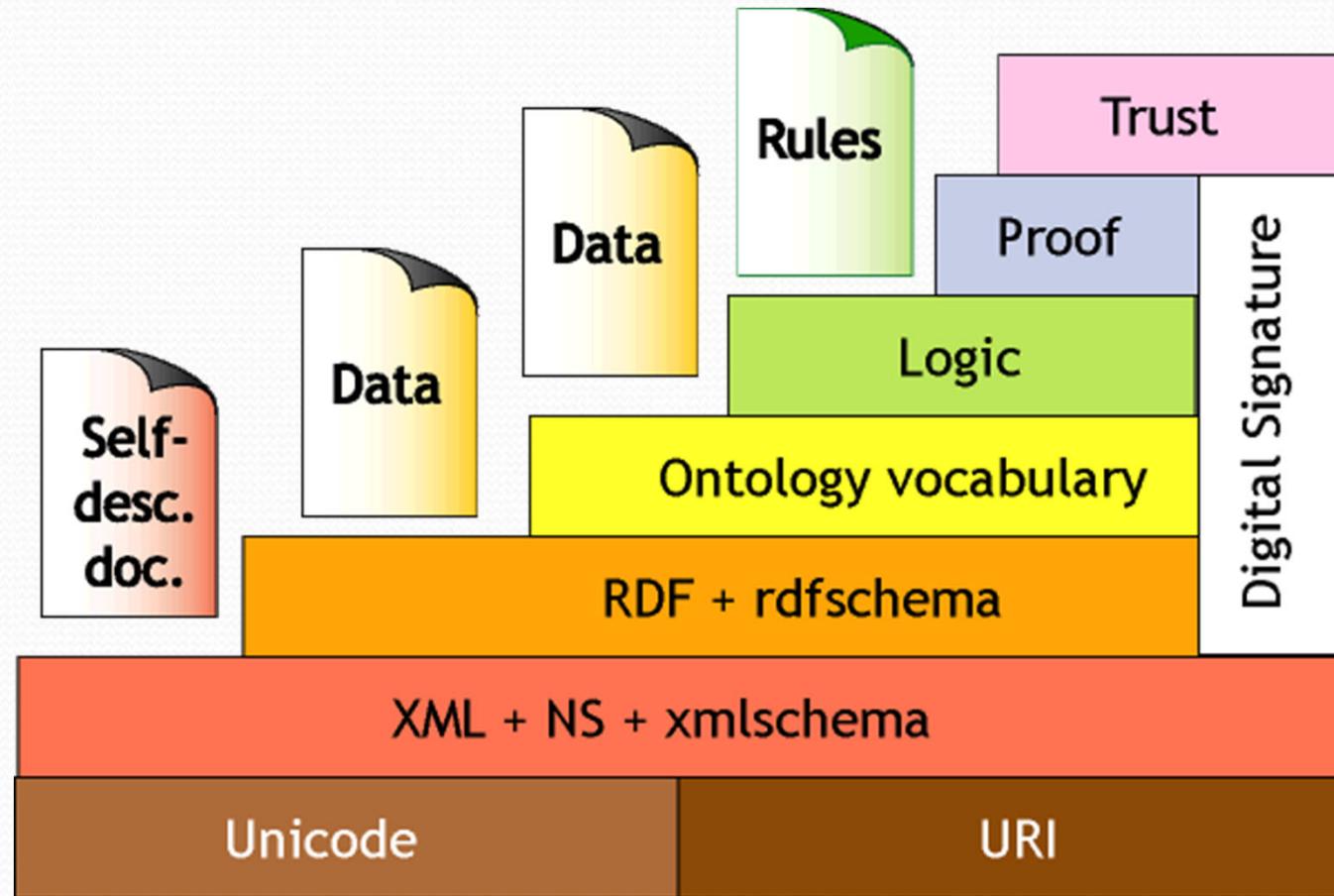
ONTOLÓGIÁK, OWL₂, DL

Méréstechnika és Információs Rendszerek Tanszék

A szemantikus web koncepció

"The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation."

-- Tim Berners-Lee



Ontológiák, az OWL nyelv

Szemantikus web példa

- Legyen ismert ez az állítás:
 - Budapest Magyarország fővárosa
- Háttértudásunk segítségével levezethetjük:
 - Budapest Magyarország fővárosa ✓
 - Magyarország egy állam ✓
 - Budapest egy város ✓
 - Budapest Magyarországon van ✓
 - Debrecen nem Magyarország fővárosa ✗
 - Budapest nem Ausztria fővárosa ✗
 - Budapest nem egy állam ✗
 - ...

Szemantikus web példa

- RDFS szintig nem tudjuk leírni a következőket:
 - minden államnak pontosan egy fővárosa van
 - Tulajdonság számosság
 - minden város csak egy ország fővárosa lehet
 - Függvény tulajdonság
 - egy város nem lehet egyben egy állam is.
 - Diszjunkt osztályok
 - ...

OWL logika igény példa

- Állatok két csoportja: **Male** (hím) és **Female** (nőstény).

```
<rdfs:Class rdf:ID="Male">
  <rdfs:subClassOf rdf:resource="#Animal"/>
</rdfs:Class>
```

- A **subClassOf** kifejezés felhasználásával definiálhatjuk, hogy a **#Male** osztály alosztálya az **#Animal** osztálynak

```
<rdfs:Class rdf:ID="Female">
  <rdfs:subClassOf rdf:resource="#Animal"/>
  <owl:disjointWith rdf:resource="#Male"/>
</rdfs:Class>
```

- Nőstény állatok és alosztályát alkotják az állatok osztálynak, de tudjuk hogy nincs közös elem a hím osztállyal (**disjointWith** elem használata).

OWL elemek, példák

- Osztályok
 - Person osztály
 - Man, Woman alosztály
- Tulajdonságok (egyedi)
 - isWifeOf, isHusbandOf
- Tulajdonság jellemzők, korlátok
 - inverseOf
 - domain
 - range
 - cardinality
- Osztályok közti relációk
 - disjointWith

Ontológiák

- RDFS hasznos, de nem ad megoldást a szemantika pontos leírására.
- Összetett alkalmazások további igényei:
 - Tulajdonságok leírása, jellemzése
 - Különböző URI-val rendelkező objektumok azonosságának leírása (ekvivalencia)
 - Osztályok diszjunkt vagy éppen ekvivalens jellege
 - Osztályok konstruálása (nemcsak megnevezése)
 - Következtetési igények támogatása:
 - Pl.: "Ha két «Person» erőforrás «A» és «B» azonos «**foaf:email**» tulajdonsággal rendelkeznek, akkor «A» és «B» identikus.

Ontológiák

- Az SZW világban az ontológiákat a következő értelemben használjuk:

Fogalmak és ezek relációnak definiálása
egy adott tudásterület leírása céljából.

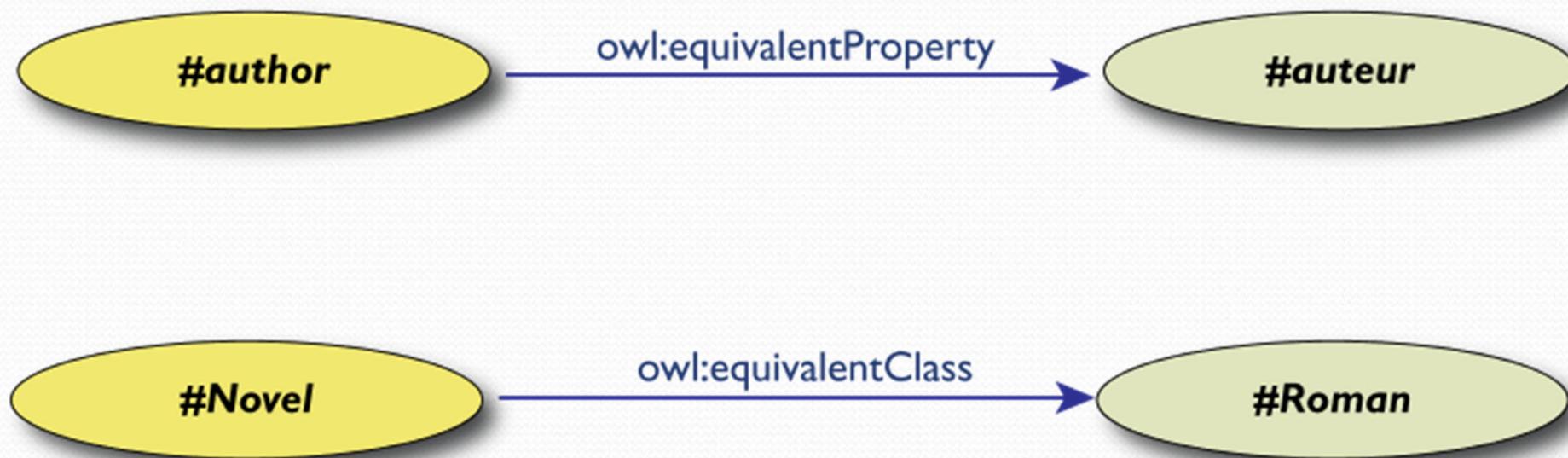
- Az RDFS is tekinthető egy egyszerű ontológia nyelvnek
- Nyelvek definiálása mindig egyfajta kompromisszum
 - gazdag szemantika tudás gazdag alkalmazásokhoz
 - ésszerűség (fizibilitás, megvalósíthatóság, fordító, következtetőgép)

Web Ontology Language = OWL

- OWL az SZW struktúrába egy újabb réteg, az RDFS bővítése
 - Saját névterek, saját elemek, kifejezések
 - RDFS-re épül (tartalmazza)
- Önálló SZW ajánlás
 - “OWL 2” – 2010 óta gyakorlatilag csak ezt használjuk
 - Akit a részletek érdekelnek:
<https://www.w3.org/TR/owl2-overview/>

Ekvivalencia relációk

- Osztályokra:
 - **owl:equivalentClass**: két osztálynak azonosak az elemei
 - **owl:disjointWith**: nincs közös elemük
- Tulajdonságokra:
 - **owl:equivalentProperty**
 - Példa: **a:author** vs. **f:auteur**
 - **owl:propertyDisjointWith**
- Egyedekre:
 - **owl:sameAs**: két URI ugyanazt a fogalmat vagy egyedet reprezentálja
 - **owl:differentFrom**: negált kifejezése az **owl:sameAs** kifejezésnek



Használat: owl:sameAs

- Utazási példából, Amsterdam leírása két forrásban:(Dbpedia és Geonames):

```
<http://dbpedia.org/resource/Amsterdam>
owl:sameAs <http://sws.geonames.org/2759793>;
```

Példa logikai kapcsolatra

Ha egy tulajdonság inverse functional–nak lett definiálva, akkor, akkor a tulajdonság tárgya egyértelműen meghatározza a tulajdonság alanyát.

- Ha a következő igaz két állításra:

```
:email rdf:type owl:InverseFunctionalProperty.  
<A> :email "mailto:a@b.c".  
<B> :email "mailto:a@b.c".
```

Akkor ebből levezethető:

```
<A> owl:sameAs <B>.
```

- Így új relációkhoz jutunk (RDFS-ben erre nem volt lehetőség)

Osztályok az OWL-ben

- RDFS: létező osztályokat alosztály struktúrába rendezhettük – osztályhierarchia építése, semmi több...
- OWL osztályok konstruálhatók más osztályok vagy példányok alapján, :
 - Elemek felsorolásával
 - Osztályok relációinak alkalmazásával: metszet, unió, komplemens, stb.

Osztályok OWL

- Osztályok, egyedek (*classes, individuals*)
 - Önálló **owl:Class** osztály van definiálva (az **rdfs:Class** specializációjaként)
 - Egyedek (individuals) definiálása külön osztályban történik: **owl:Thing**
- Pl. egy pontos definícióra:

```
ex:Person rdf:type owl:Class.
```

```
<uri-for-Amitav-Ghosh>
  rdf:type owl:Thing;
  rdf:type owl:Person .
```

```
:£ rdf:type owl:Thing.
:€ rdf:type owl:Thing.
:$ rdf:type owl:Thing.
:Currency
  rdf:type owl:Class;
  owl:oneOf (:€ :£ :$).
```

Unió

```
:Novel          rdf:type owl:Class.  
:Short_Story    rdf:type owl:Class.  
:Poetry         rdf:type owl:Class.  
:Literature     rdf:type owl:Class;  
                 owl:unionOf (:Novel :Short_Story :Poetry) .
```

- További lehetőségek: **complementOf**, **intersectionOf**, ...

Példa...

Ha:

```
:Novel          rdf:type owl:Class.  
:Short_Story    rdf:type owl:Class.  
:Poetry         rdf:type owl:Class.  
:Literature     rdf:type owl:Class;  
                 owl:unionOf (:Novel :Short_Story :Poetry).  
  
<myWork> rdf:type :Novel .
```

akkor

```
<myWork> rdf:type :Literature .
```

Eddig...

- OWL: erős leíró elemeket definiáltunk
- pl., adatbázisok összeköthetőek **owl:sameAs**, vagy inverse functional tulajdonságokkal.
- Számos kapcsolatot felderíthetünk hagyományos következtetési eljárásokkal

Korlátozások formálisan is...

- Létezik **owl:Restriction** osztály
 - Hivatkozva a korlátozandó tulajdonságokat
 - Megadva a korlátozás tartalmát
- Példa: subClass

```
:Listed_Price rdfs:subClassOf [  
    rdf:type owl:Restriction;  
    owl:onProperty p:currency;  
    owl:allValuesFrom :Currency.  
].
```

Lehetséges felhasználás...

Ha:

```
:Listed_Price rdfs:subClassOf [  
    rdf:type owl:Restriction;  
    owl:onProperty p:currency;  
    owl:allValuesFrom :Currency.  
].  
  
:price rdf:type :Listed_Price .  
  
:price p:currency <something> .
```

Akkor a következő állítás igaz:

```
<something> rdf:type :Currency .
```

További korlátozások

- **allValuesFrom** helyettesíthető:
 - **someValuesFrom**
 - Példa: valamelyik devizában fejezzük ki az árfolyamot (legalább egyben)
 - **hasValue**, legalább egy értéke kell, hogy létezzen
- Számosság korlátozások (értékvizsgálat helyett)
 - Pl. legalább egyszer elő kell, hogy forduljon egy elem egy kategóriában

OWL 2

- Korlátozások definiáltak
 - classes, individuals, object , datatype properties - megkötésekkel
 - object properties csak individuals -re
 - *datatype* property nincs tovább specifikálva
 - ...
- Hatékony következtető algoritmus létezik!

OWL 2

- Korlátozások megadásával nagy méretű ontológiák építhetők,
és alkalmazhatóak pl. orvosi, robotika, biológia
tárgyterületeken
- OWL 2 lett a formális ontológiák nyelve
 - Nem feltétlenül a weben használjuk

OWL2: Könnyítések (szintaktika)

- Diszjunkt osztályok

- OWL 1:

```
:Wine owl:equivalentClass [  
    a owl:Class ;  
    owl:unionOf (:RedWine :RoséWine :WhiteWine) ] .  
  
:RedWine owl:disjointWith :RoséWine, :WhiteWine .  
:RoséWine owl:disjointWith :WhiteWine .
```

- OWL 2:

```
:Wine owl:disjointUnionOf (:RedWine  
    :RoséWine :WhiteWine) .
```

- Megengedett formalizmus:

```
_:_x a owl:AllDisjointClasses ;  
    owl:members (:RedWine :RoséWine WhiteWine) .
```

OWL2: Könnyítések (szintaktika)

- Állítások, tulajdonságok negáltja kifejezhető
- pl.: Paul nem Peter apja
- ```
_x [a owl:NegativeObjectPropertyAssertion;
 owl:sourceIndividual :Paul ;
 owl:targetIndividual :Peter ;
 owl:assertionProperty :fatherOf] .
```

VAGY

```
Paul a
[owl:complementOf
 [a owl:Restriction ;
 owl:onProperty :fatherOf ;
 owl:hasValue :Peter]
].
```

# OWL2: Reflexiv osztály korlátzások

hasSelf reláció használata

Példa: autodidakta tanulás definíciója

```
:AutoDidact owl:equivalentClass [
 a owl:Restriction ;
 owl:onProperty :teaches ;
 owl:hasSelf "true"^^xsd:boolean] .
```

# OWL alapú ontológia

- "Debrecen nem Magyarország fővárosa ✗
- Miért nem?
  - Országoknak pontosan egy fővárosa van,
  - Debrecen és Budapest nem ugyanaz a város

- OWL:

```
:capitalOf a owl:InverseFunctionalProperty .
:Budapest :capitalOf :Magyarország .
:Budapest owl:differentFrom :Debrecen .
```

Lekérdezés { :Debrecen :capitalOf :Magyarország . } → false

# OWL alapú ontológia

- Budapest nem Ausztria fővárosa ✗
- Miért nem?
  - Egy város csak egy ország fővárosa lehet
  - Ausztria és Magyarország nem azonos

- Hasonlóan:

```
:capitalOf a owl:FunctionalProperty .
:Budapest :capitalOf :Magyarország .
:Ausztria owl:differentFrom :Magyarország .
```

Lekérdezés { :Budapest :capitalOf :Ausztria . } → false

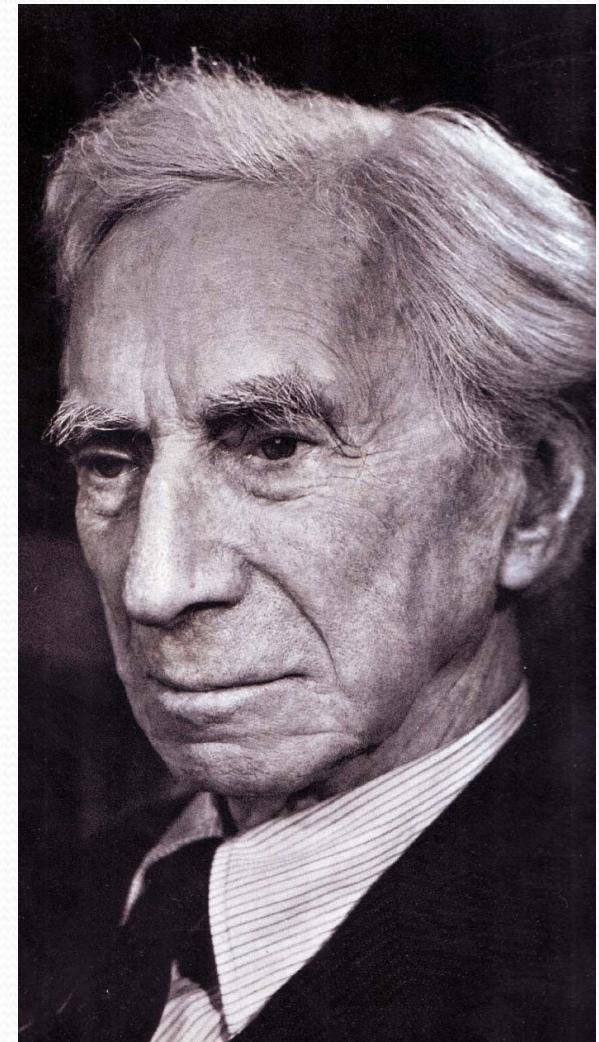
# OWL2 példa: Russell paradoxon

Paradoxon

Bertrand Russell, 1918

Egy városban, ahol pontosan egy borbély van, mindenkit megborotvál, aki nem borotválja meg önmagát.

Ki borotválja meg a borbélyt?



# OWL2 példa: Russell paradoxon

## Osztály definíciók

```
:People owl:disjointUnionOf
(:PeopleWhoShaveThemselves
:PeopleWhoDoNotShaveThemselves) .
```

## Relációk:

```
:shavedBy rdfs:domain :People .
:shavedBy rdfs:range :People .
:shaves owl:inverseOf :shavedBy .
```

## Szabály:

```
:People rdfs:subClassOf [
a owl:Restriction ;
owl:onProperty :shavedBy ;
owl:cardinality "1"^^xsd:integer] .
```

# OWL2 példa: Russell paradoxon

- Borbély definíciója:

- :Barbers rdfs:subClassOf :People ;owl:equivalentClass [
  - rdf:type owl:Class ;
  - owl:oneOf ( :theBarber )
- ] .

# OWL2 példa: Russell paradoxon

Magukat borotválók osztálya:

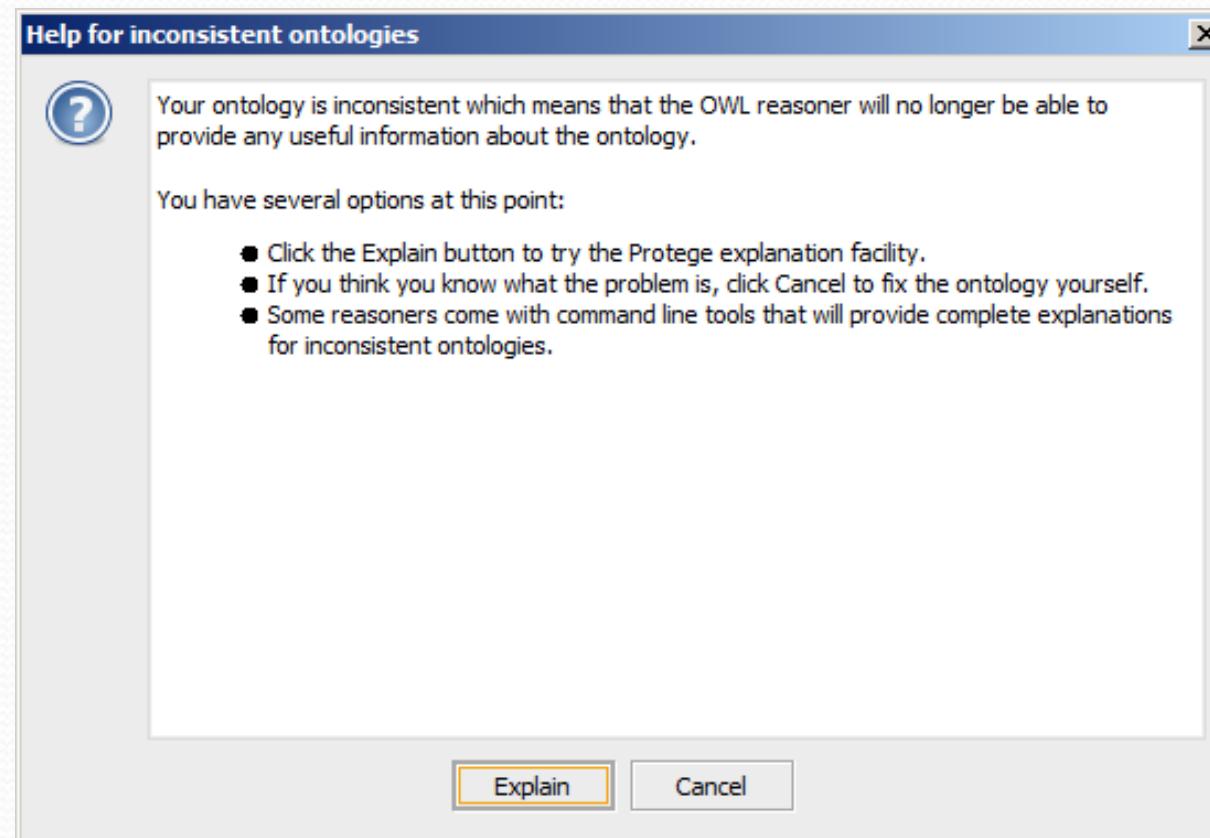
```
:PeopleWhoShaveThemselves owl:equivalentClass [
 rdf:type owl:Class ;
 owl:intersectionOf
 (:People
 [
 a owl:Restriction ;
 owl:onProperty :shavedBy ;
 owl:hasSelf "true"^^xsd:boolean
]
)
] .
```

# OWL2 példa: Russell paradoxon

Akik nem magukat borotválják:

```
:PeopleWhoDoNotShaveThemselves owl:equivalentClass [
 a owl:Class ;
 owl:intersectionOf (
 :People
 [a owl:Restriction
 owl:onProperty :shavedBy ;
 owl:allValuesFrom :Barbers
]
)
] .
```

# OWL2 példa: Russell paradoxon



Inconsistent ontology explanation X

Show regular justifications    All justifications  
 Show laconic justifications    Limit justifications to  
1 1

Explanation 1  Display laconic explanation

Explanation for: Thing SubClassOf Nothing

|                                                                                    |                             |                   |
|------------------------------------------------------------------------------------|-----------------------------|-------------------|
| 1) PersonsWhoDoNotShaveThemselves(?x) -> shaves(the-barber, ?x)                    | In 1 other justifications   | <a href="#">?</a> |
| 2) PersonsWhoDoNotShaveThemselves DisjointWith PersonsWhoShaveThemselves           | In ALL other justifications | <a href="#">?</a> |
| 3) Barber SubClassOf Person                                                        | In ALL other justifications | <a href="#">?</a> |
| 4) shaves(?x, ?x) -> PersonsWhoShaveThemselves(?x)                                 | In ALL other justifications | <a href="#">?</a> |
| 5) shaves(the-barber, ?x) -> PersonsWhoDoNotShaveThemselves(?x)                    | In 1 other justifications   | <a href="#">?</a> |
| 6) PersonsWhoShaveThemselves(?x) -> shaves(?x, ?x)                                 | In ALL other justifications | <a href="#">?</a> |
| 7) Person EquivalentTo PersonsWhoDoNotShaveThemselves or PersonsWhoShaveThemselves | ALL other justifications    | <a href="#">?</a> |
| 8) the-barber Type Barber                                                          | In ALL other justifications | <a href="#">?</a> |

OK

# SZEMANTIKUS WEB

## 3. előadás

Méréstechnika és Információs Rendszerek Tanszék  
<https://www.mit.bme.hu/oktatas/targyak/vimiaco4>

# Zh feladat 1/1

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:ex="http://example.org/">

 <rdf:Description rdf:about="http://example.org/Magyarország">
 <rdf:type rdf:resource="http://example.org/Ország"/>
 </rdf:Description>

 <rdf:Description rdf:about="http://example.org/Fővárosa">
 <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
 <rdfs:domain rdf:resource="http://example.org/Város"/>
 <rdfs:range rdf:resource="http://example.org/Ország"/>
 </rdf:Description>
```

# Zh feladat 1/2

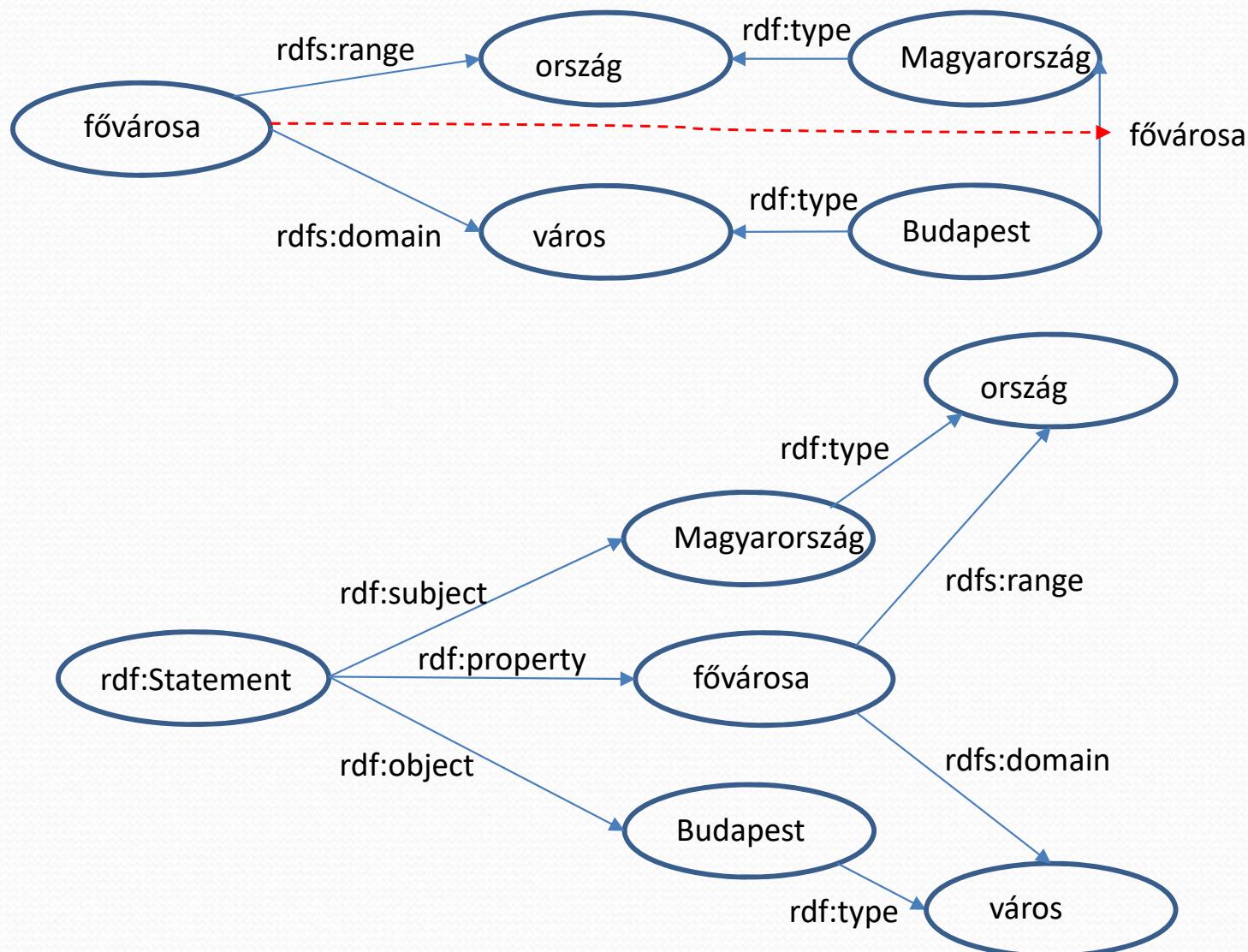
```
<rdf:Description rdf:about="http://example.org/Ország">
<rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
<rdfs:label xml:lang="en">country</rdfs:label>
</rdf:Description>
<rdf:Description rdf:about="http://example.org/Budapest">
<rdfs:label xml:lang="en">Budapest</rdfs:label>
<rdf:type rdf:resource="http://example.org/Város"/>
<ex:Fővárosa rdf:resource="http://example.org/Magyarország"/>
</rdf:Description>
<rdf:Description rdf:about="http://example.org/Város">
<rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
<rdfs:label xml:lang="en">city</rdfs:label>
</rdf:Description>
</rdf:RDF>
```

# Zh feladat 1/3

a.) Milyen információt ír le a fenti RDF(S) dokumentum? Adjon meg egy lehetséges interpretációt saját szavaival! (4 pont)

*Minden országnak (angolul: country) van fővárosa, ami egy város (angolul: city). Magyarország egy ország, amelynek fővárosa Budapest egy város.*

b) Adja meg (rajzolja meg) az RDF(S) gráf grafikus reprezentációját!  
(4 pont)



# Zh feladat 1/4

c) Fogalmazzon meg egy lekérdezést SPARQL/SQL formátumban az RDF gráfban található országok (ex:Ország) fővárosainak (ex:Fővárosa) kigyűjtésére. (4 pont)

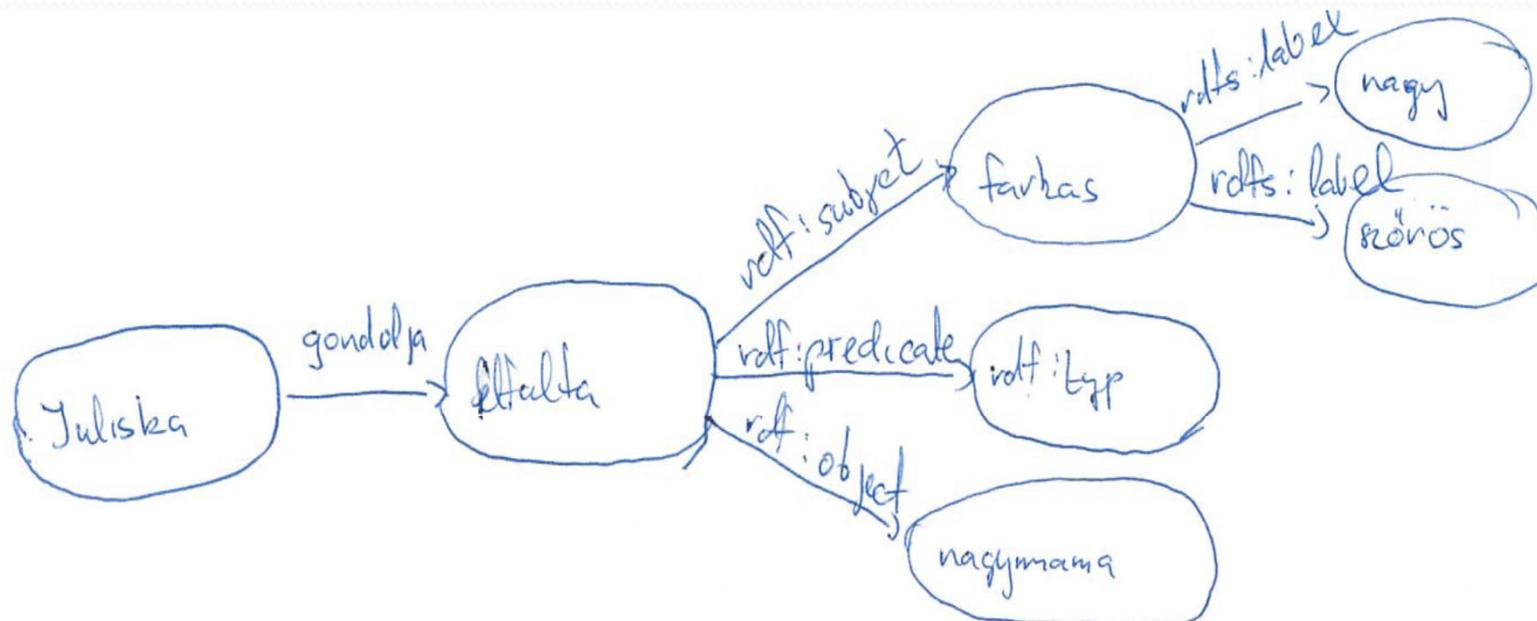
PREFIX ex:<http://example.org/>

SELECT ?f

WHERE { ?v a ex:Ország .  
?f ex:Fővárosa ?v}

# Zh feladat 1/5

Modellezze a következő mondatokat RDF-ben, a gráf leírásához bármelyik ismert RDF szintaxist használhat (pl.: XML, gráf, n3 predikátumok)! (6 pont)  
*Júliska azt gondolja, hogy a nagy, szőrős farkas felfalta a nagymamáját.*

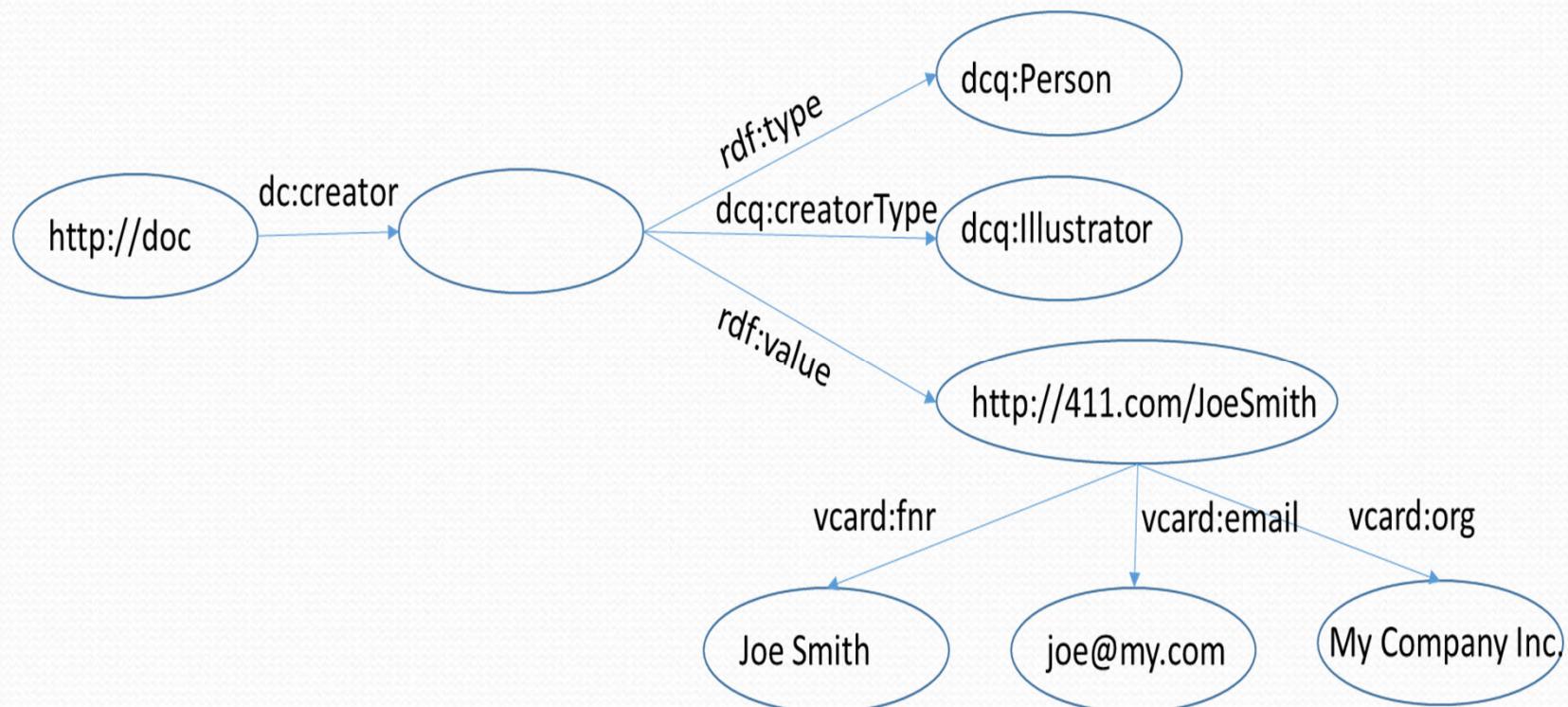


# Zh feladat 2/1

```
<rdf:Description rdf:about = "http://doc">
 <dc:creator>
 <rdf:Description>
 <rdf:type rdf:resource = "dcq:Person"/>
 <dcq:creatorType rdf:resource = "dcq:Illustrator"/>
 <rdf:value rdf:resource = "http://411.com/JoeSmith"/>
 </rdf:Description>
 </dc:creator>
</rdf:Description>
<rdf:Description rdf:about = "http://411.com/JoeSmith">
 <vcard:fn> Joe Smith </vcard:fn>
 <vcard:email> joe@my.com </vcard:email>
 <vcard:org> My Company Inc.</vcard:org>
</rdf:Description>
```

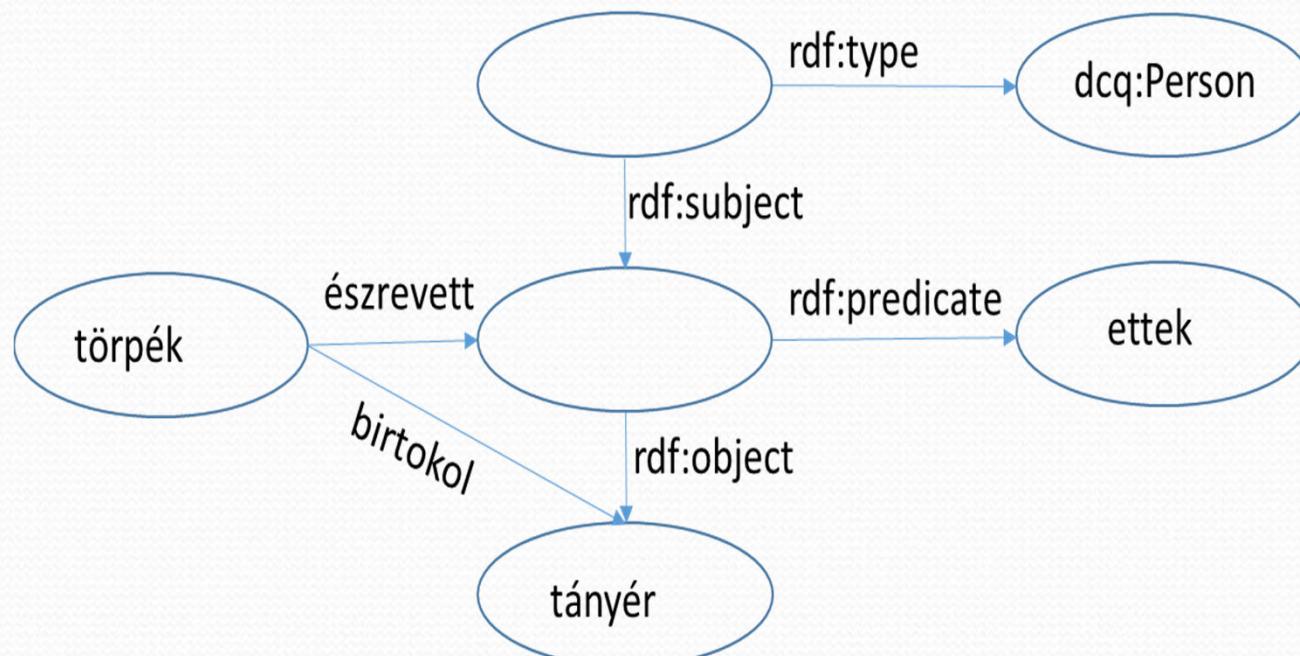
# Zh feladat 2/2

A modell egy dokumentum (egyik) szerzőjének, illusztrátorának tulajdonságait írja le, aki egy személy, Joe Smith, és akinek az email címe [joe@my.com](mailto:joe@my.com) és a MyCompany vállalatnál dolgozik.



# Zh feladat 2/3

A törpék észrevették, hogy valaki evett a tányérjukból.

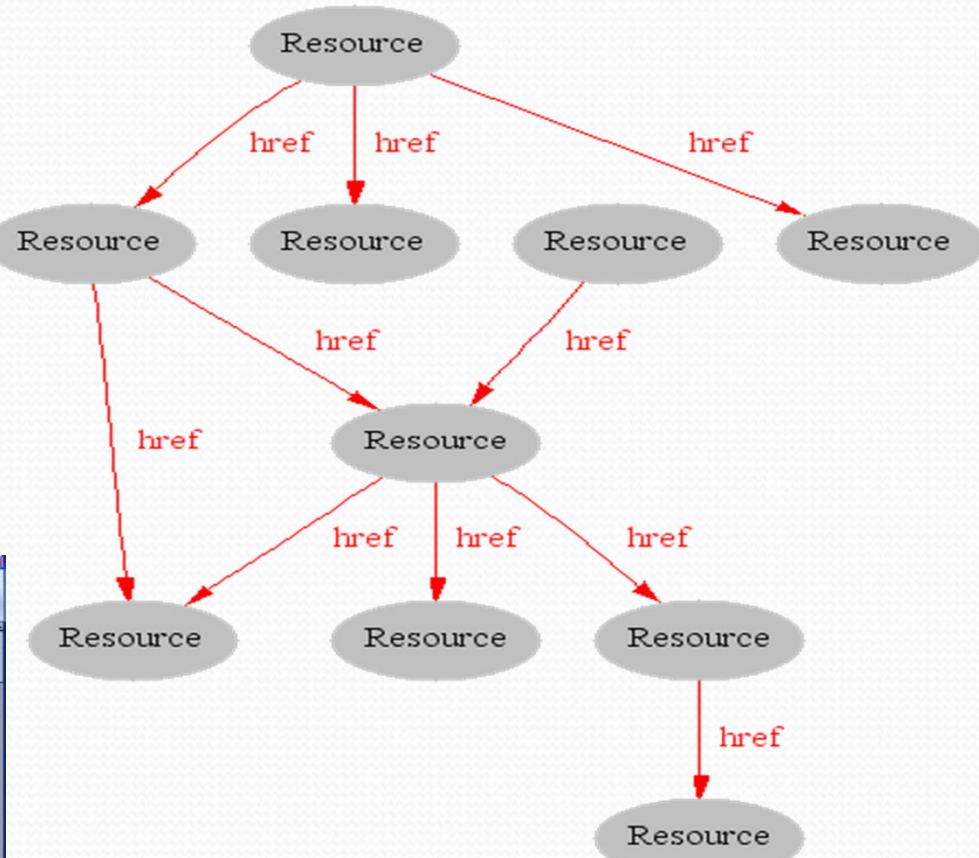


# „Szintakitikus” web

The image shows two screenshots illustrating the concept of a "syntactic web".

The top screenshot is from the "WWW2002" conference website, held in Honolulu, Hawaii, from May 7-11, 2002. It features a banner with the conference logo and title, navigation links for "Conference Proceedings", "Call for Participation", "Program", "Registration Information", "Hotel Accommodation", "Conference Committee", "Sponsorship/Exhibition Opportunities", "Volunteer Information", "Information about Hawaii", and "Previous & Future WWW Conferences". Below this is a section titled "FEATURED SPEAKERS (CONFIRMED)" featuring portraits of Tim Berners-Lee and Richard A. DeMillo.

The bottom screenshot is from Tim Berners-Lee's homepage at <http://www.w3.org/People/Berners-Lee/>. It includes a "Contents" sidebar with links like "Home", "Bio", "Before you mail me", "Address", "Talks", "Articles", "Speaking engagements", and "Press interviews". The main content area features a portrait of Tim Berners-Lee and a bio section. The bio notes his role as the 3Com Founders chair at the Laboratory for Computer Science and Artificial Intelligence (CSAIL) at the Massachusetts Institute of Technology (MIT), his work on the World Wide Web Consortium, and his book "Weaving the Web".

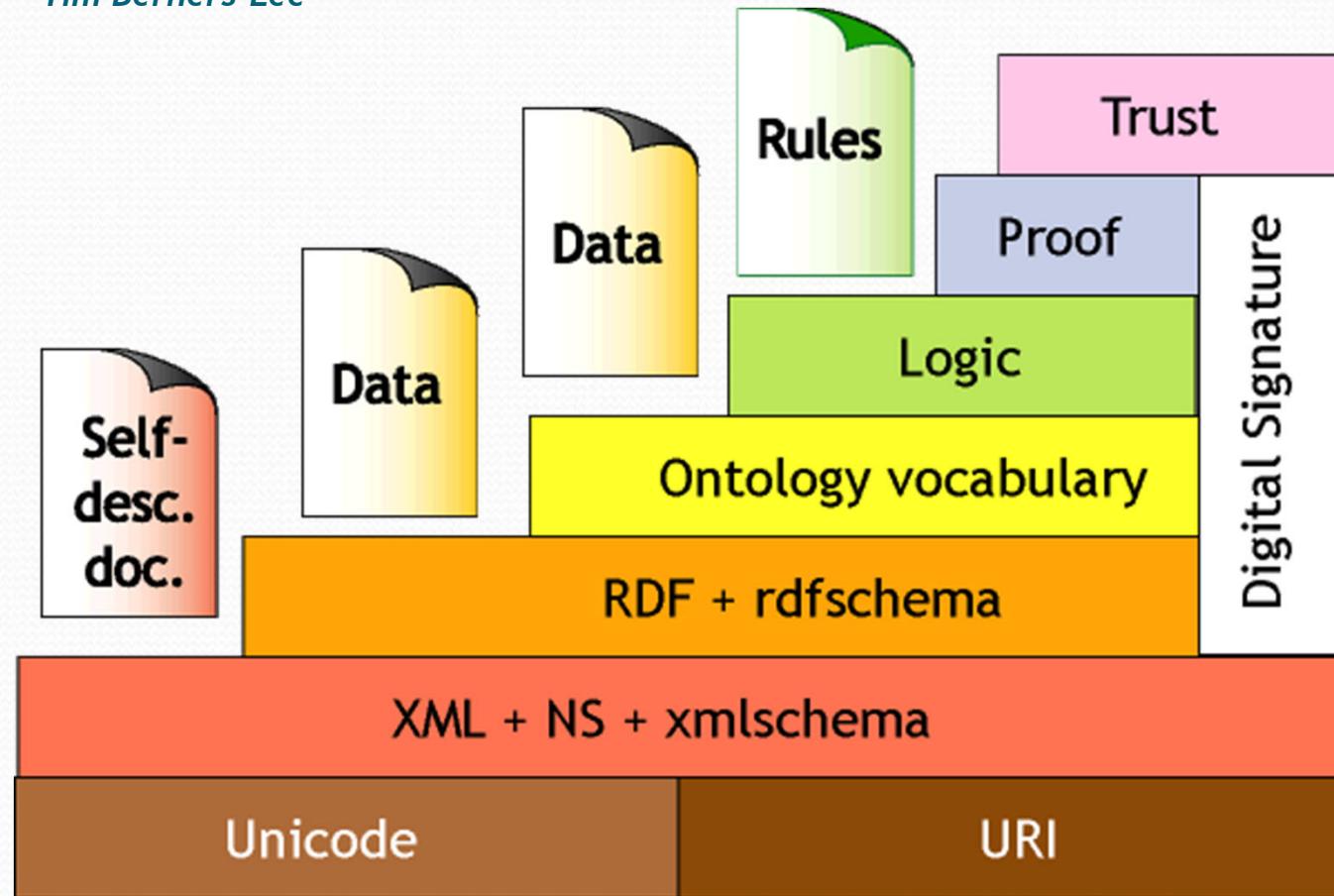


[Hendler & Miller 02]

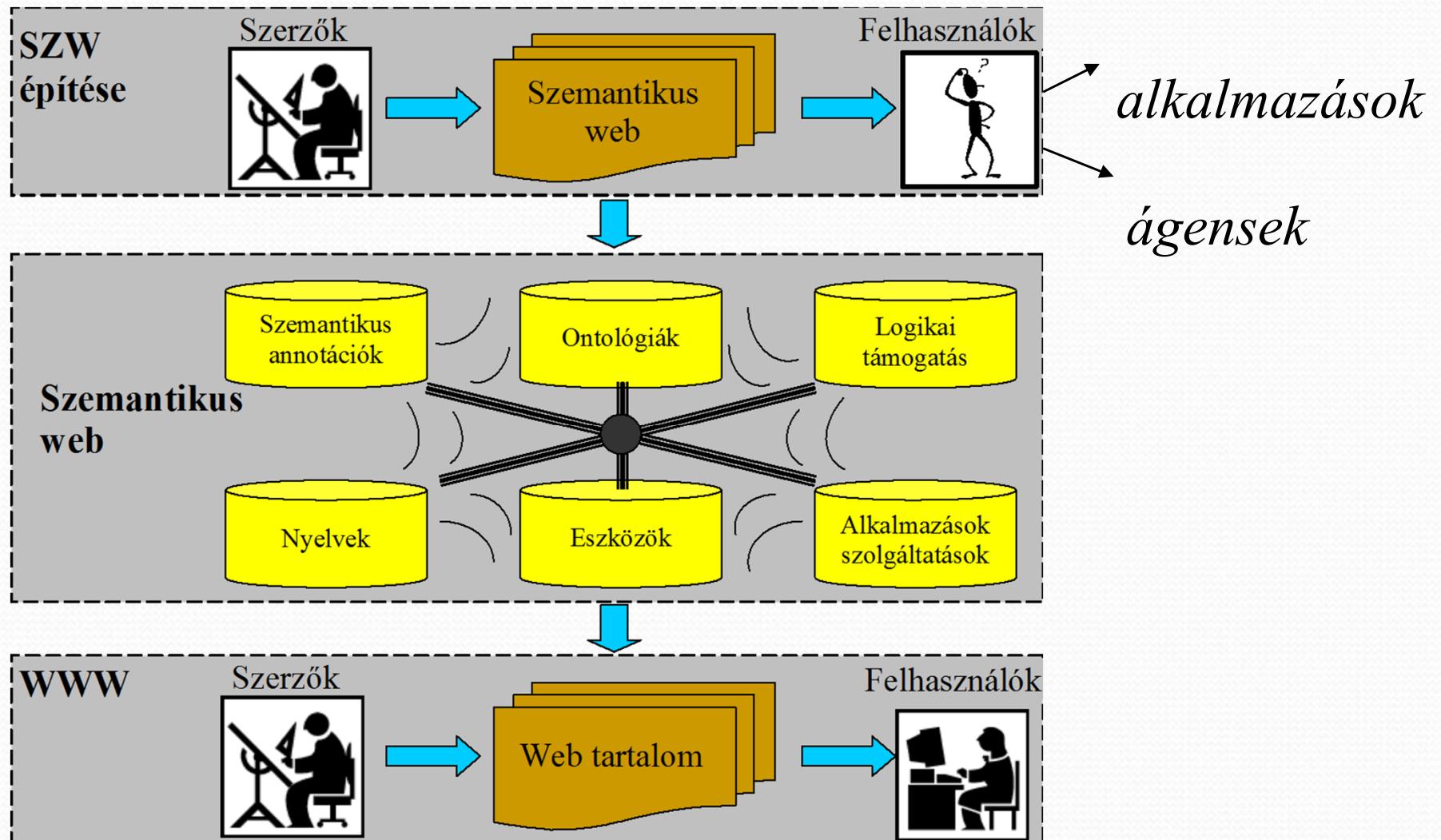
# A szemantikus web koncepció

*"The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation."*

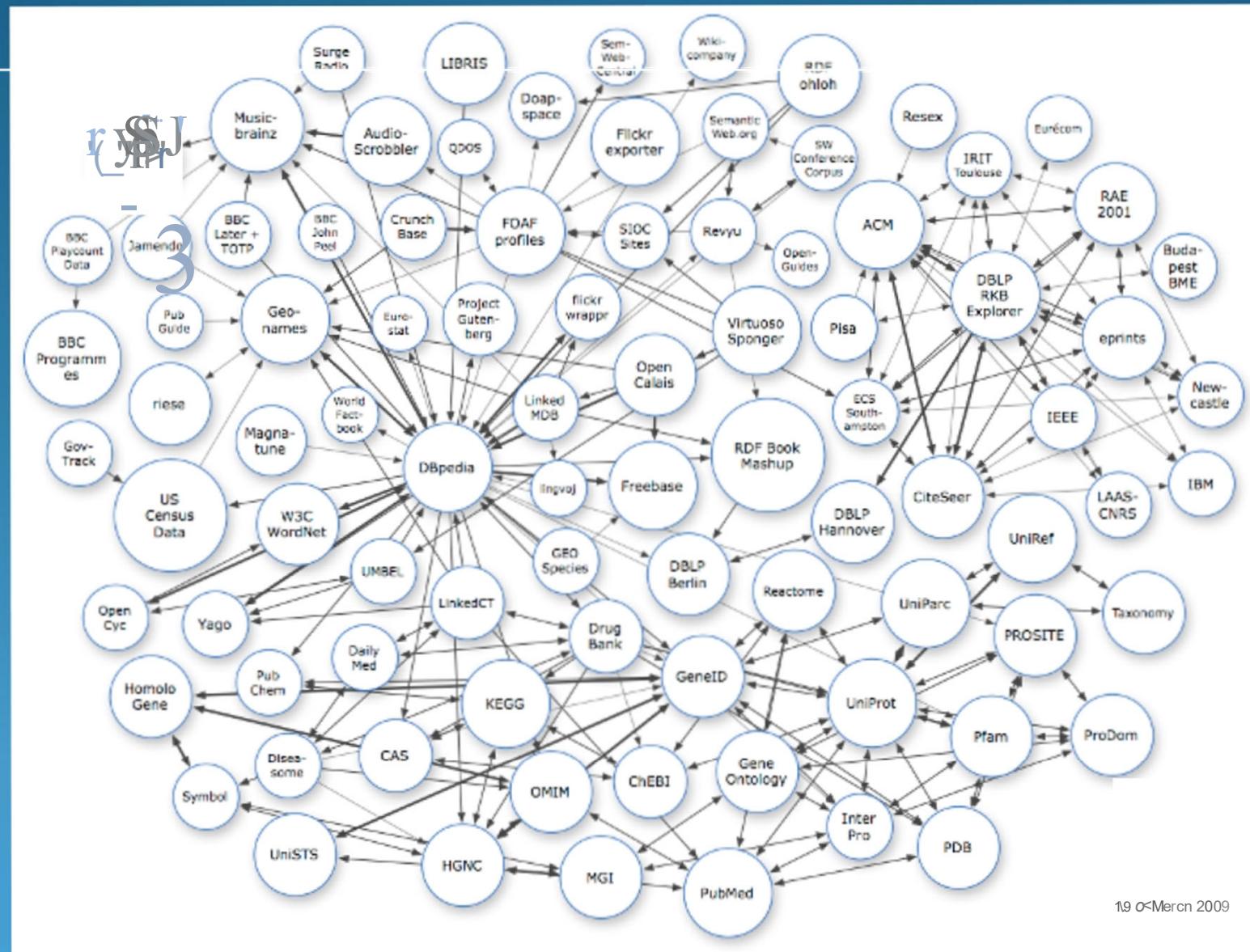
-- Tim Berners-Lee



# Szemantikus web alkalmazása



# Linked Data



# Linked Open Data projekt

- Cél: elérhetővé tenni a nyitott RDF adatbázisokat
- *Hozzunk létre kapcsolatokat az RDF adathalmazok között*
- Rögzítsünk lekérdezési lehetőségeket
- Ma is elérhetőek: milliárdnyi hármasok, sok millió kapcsolat



# Linked Open Data projekt

- A szemantikus web alapja a nagy mennyiségű tartalom szerint kapcsolható, a webes elérhető RDF adatok létezése
- A Linked Data nem specifikáció, hanem jó gyakorlatok gyűjteménye adatok megosztására a weben
- További szemantikus web technológiák (RDFS, OWL, SPARQL) segítik az adatok alkalmazásokba építését

# 5-star data



- ★ make your stuff available on the Web (whatever format) under an open license
- ★★ make it available as structured data (e.g., Excel instead of image scan of a table)
- ★★★ make it available in a non-proprietary open format (e.g., CSV as well as of Excel)
- ★★★★ use Linked Data format (URIs to identify things, RDF to represent data)
- ★★★★★ link your data to other people's data to provide context

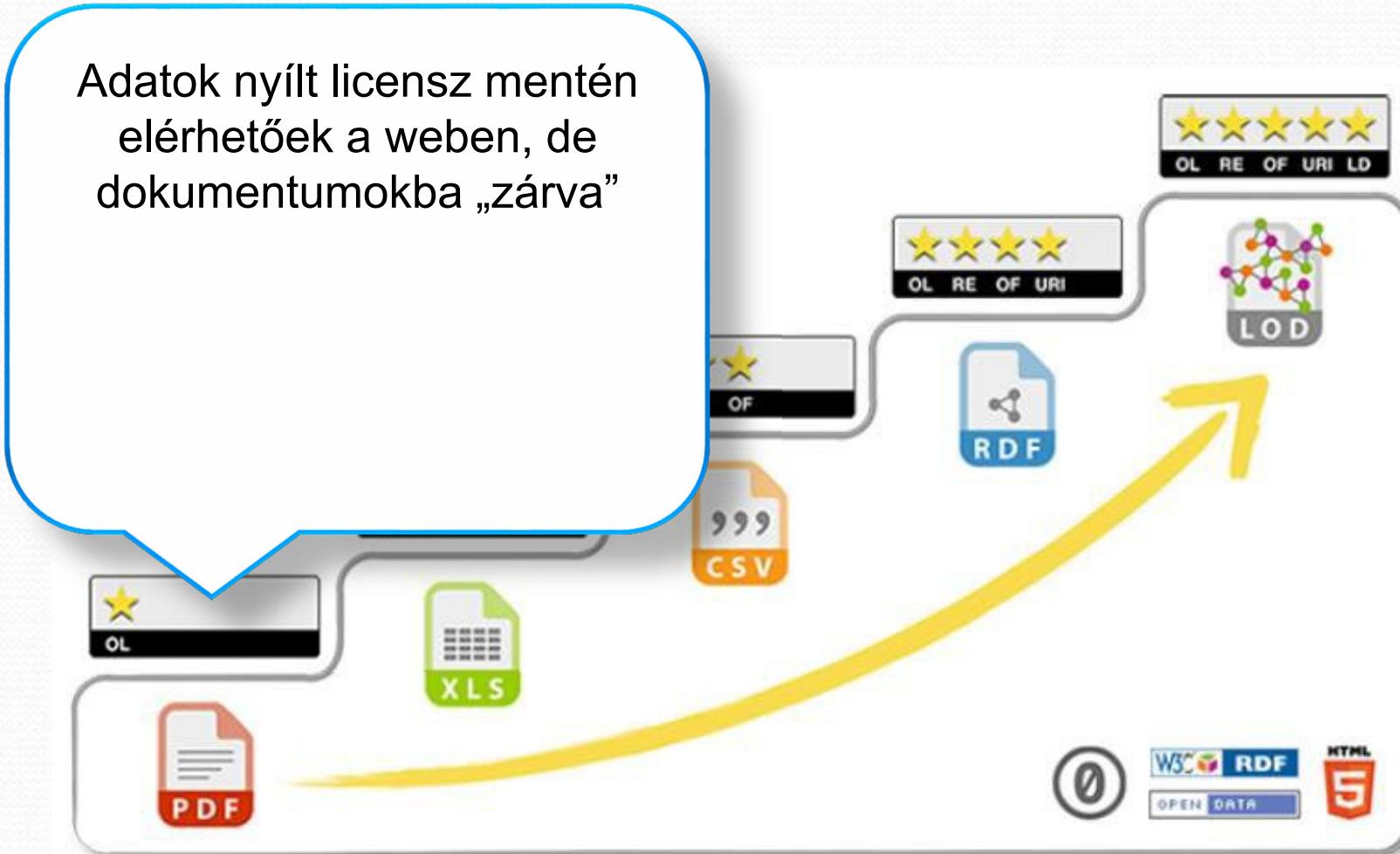
<https://5stardata.info/en/>

# 5-star data



# 5-star data

Adatok nyílt licensz mentén  
elérhetőek a weben, de  
dokumentumokba „zárva”

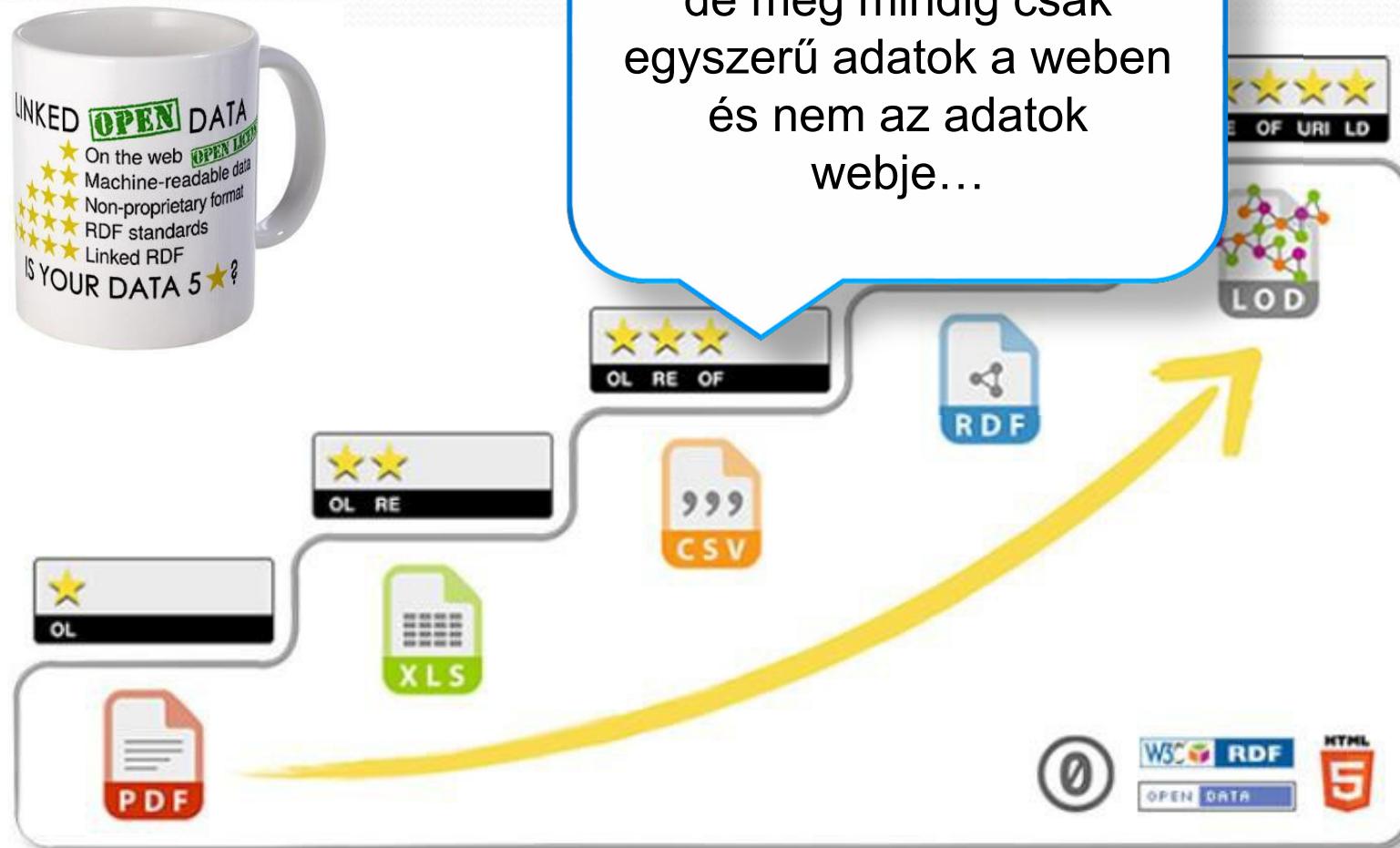


# 5-star data

Adatok nyíltan elérhetőek,  
itt már strukturált formában,  
de „zárt”, licensssel védett  
technológiákon keresztül



# 5-star data

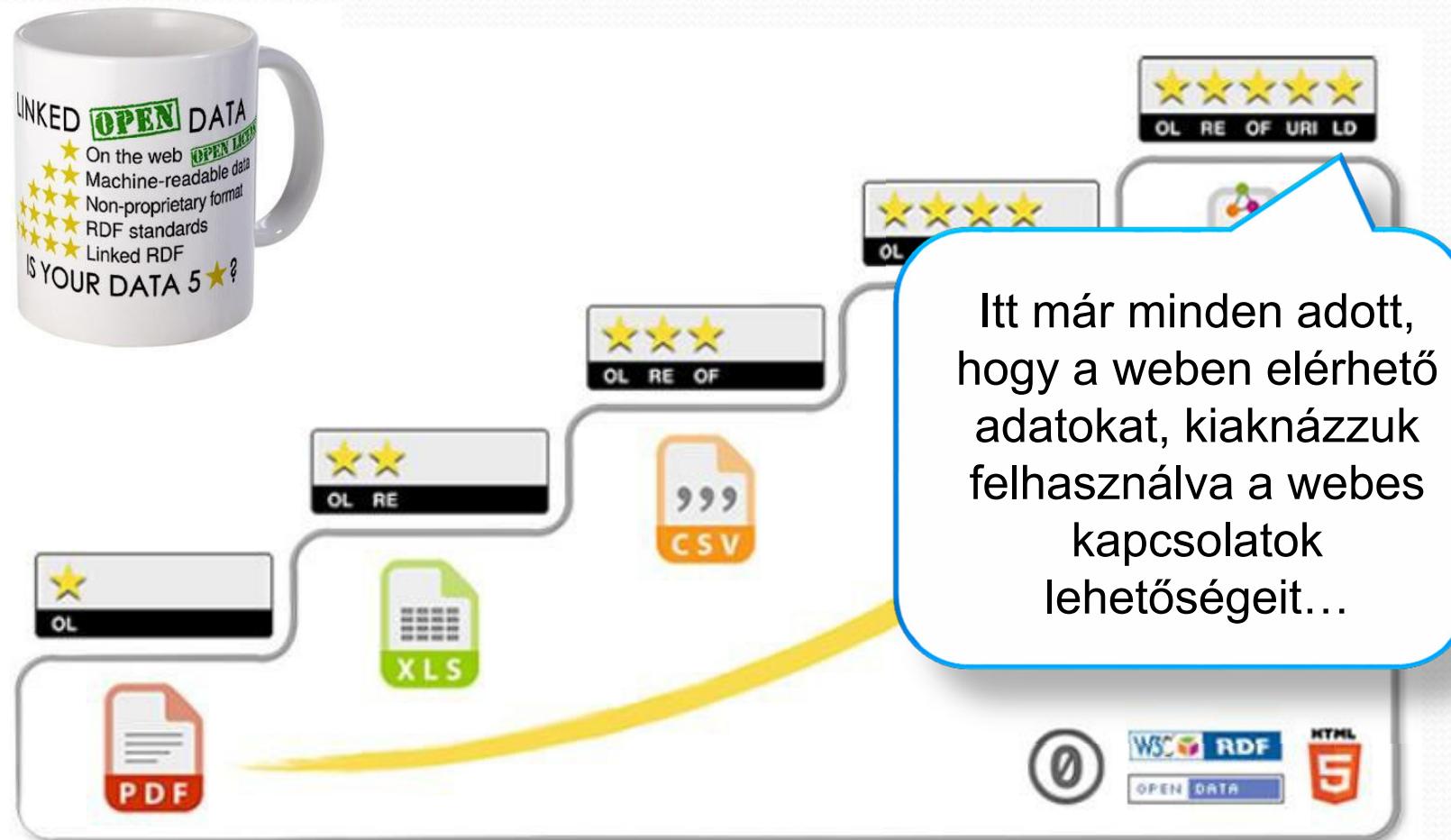


# 5-star data

Itt már az adatok közvetlenül elérhetőek a weben az egyes forrásokban, de hiányzanak a kapcsolatok



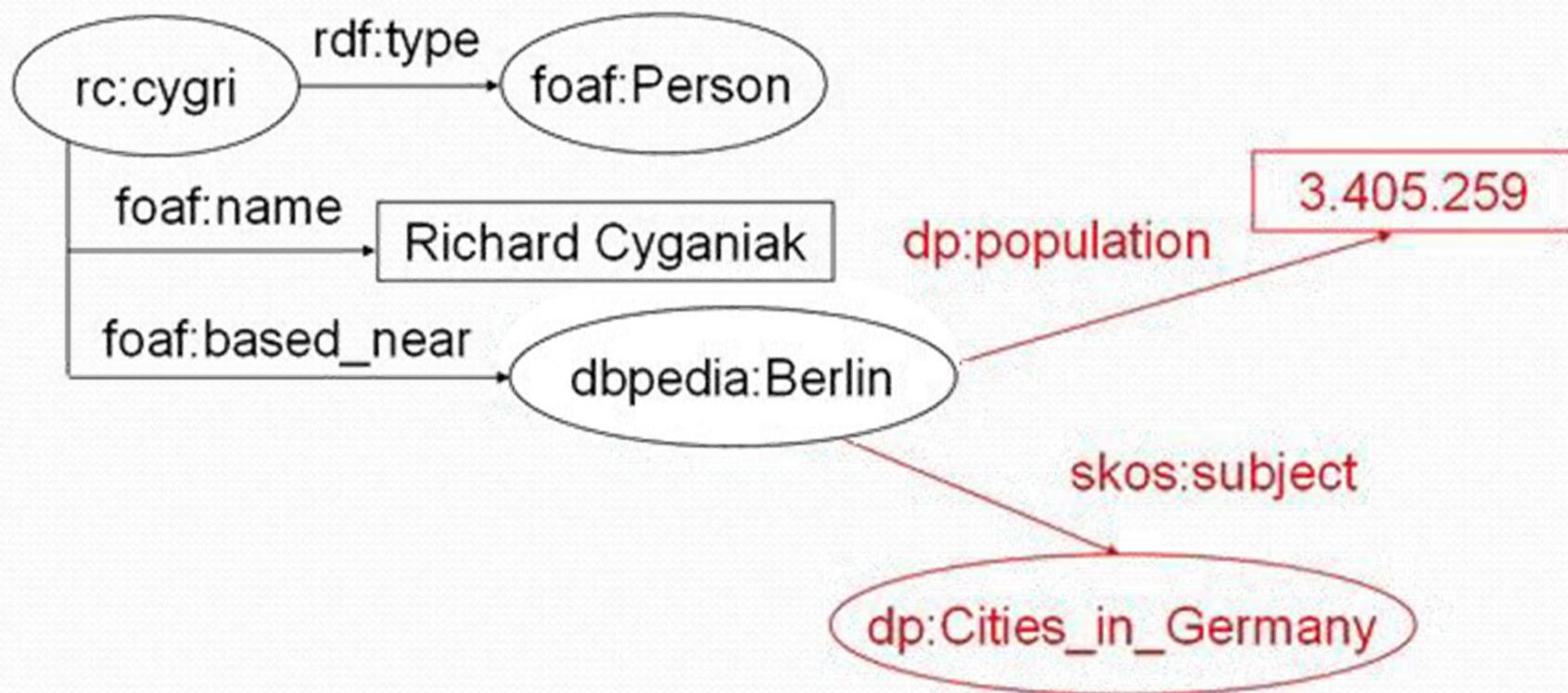
# 5-star data



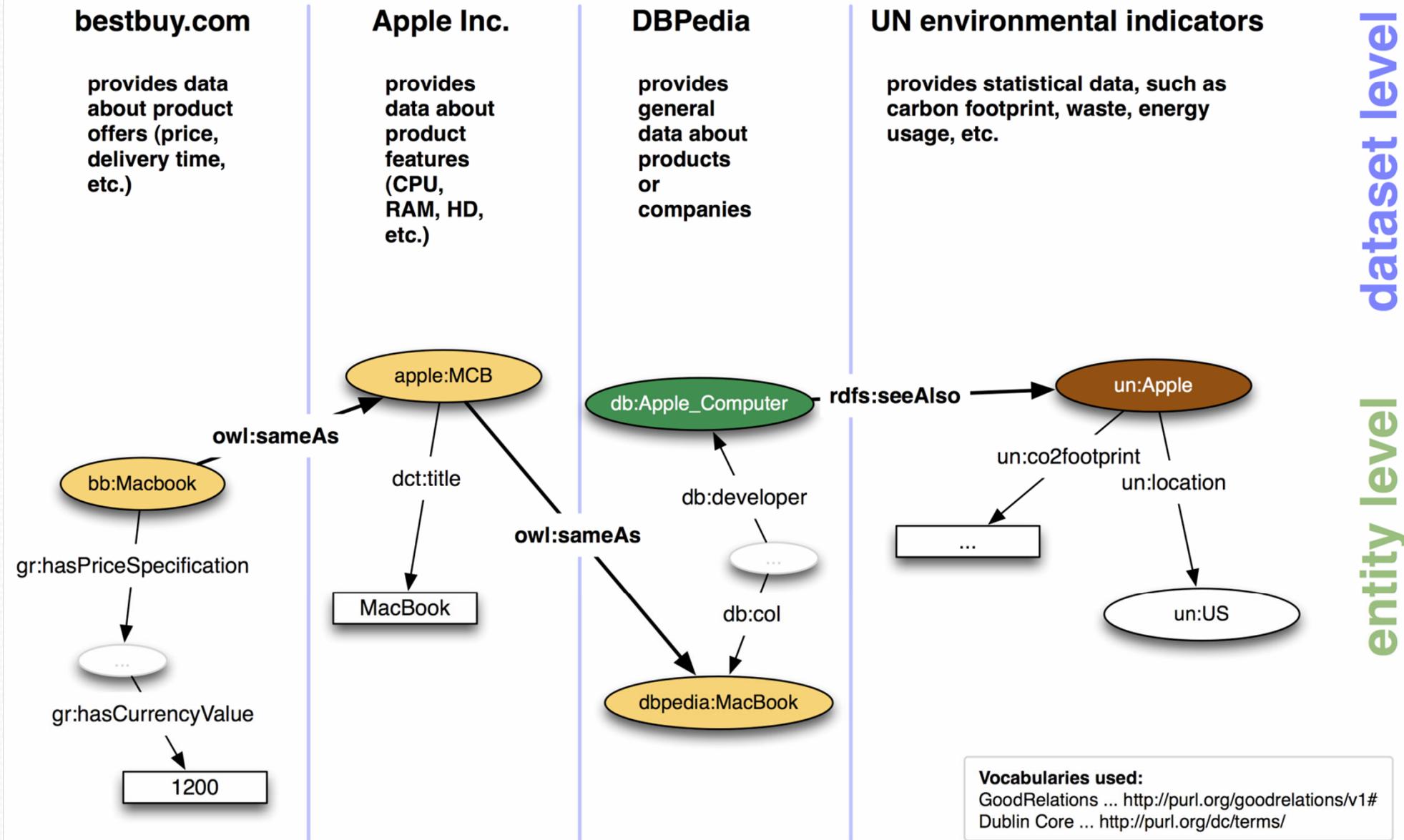
# Linked Data példa



# Linked Data példa

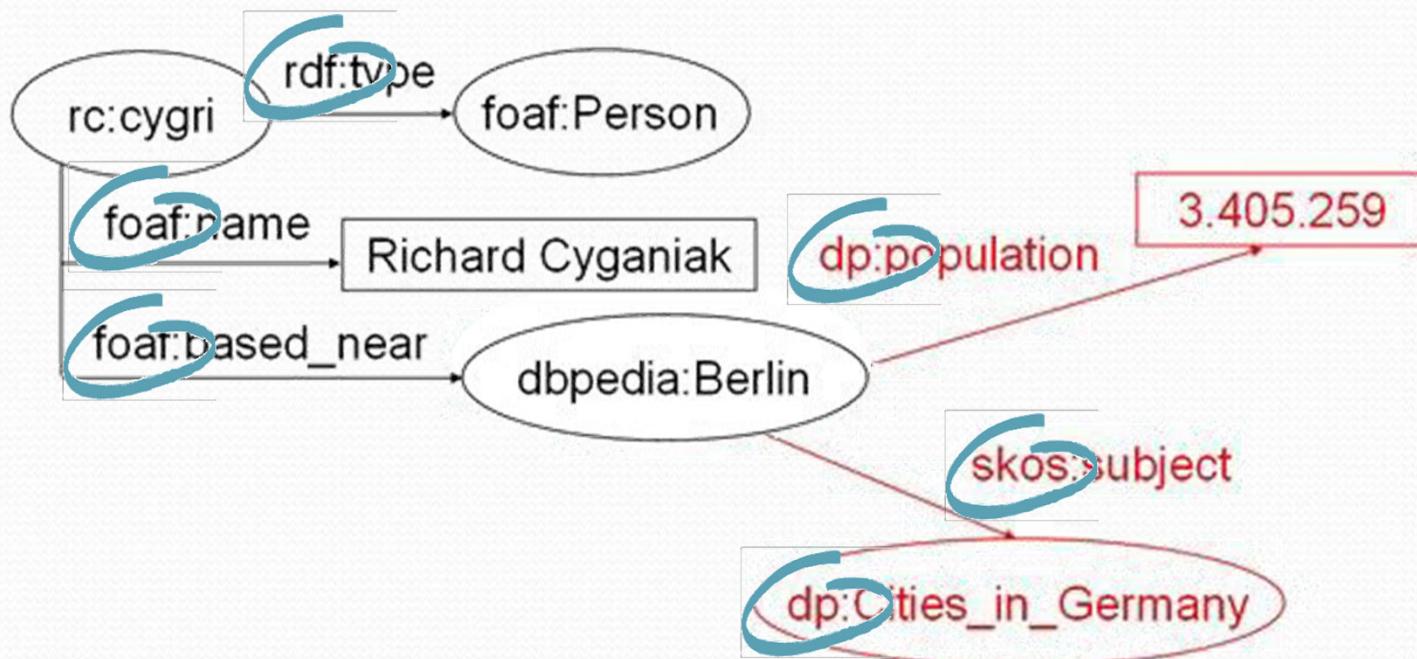


# Linked Data példa



# Szótárak

- Kifejezések definiálása (osztályok és tulajdonságok)
- Leggyakrabban RDFS vagy OWL leírások



# DBpedia: Wikipedia adatok

| Amsterdam                                                                         |                                                                                                                  |
|-----------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------|
|  |                                                                                                                  |
| The Keizersgracht at dusk                                                         |                                                                                                                  |
| Location of Amsterdam                                                             |                                                                                                                  |
| Coordinates:                                                                      | 52°22'23"N 4°53'32"E                                                                                             |
| Country                                                                           | Netherlands                                                                                                      |
| Province                                                                          | North Holland                                                                                                    |
| Government                                                                        |                                                                                                                  |
| - Type                                                                            | Municipality                                                                                                     |
| - Mayor                                                                           | Job Cohen <sup>[1]</sup> (PvdA)                                                                                  |
| - Aldermen                                                                        | Lodewijk Asscher<br>Carolien Gehrels<br>Tjeerd Herrema<br>Maarten van Poelgeest<br>Marijke Vos<br>Erik Gerritsen |
| - Secretary                                                                       |                                                                                                                  |
| Area <sup>[2][3]</sup>                                                            |                                                                                                                  |
| - City                                                                            | 219 km <sup>2</sup> (84.6 sq mi)                                                                                 |
| - Land                                                                            | 166 km <sup>2</sup> (64.1 sq mi)                                                                                 |
| - Water                                                                           | 53 km <sup>2</sup> (20.5 sq mi)                                                                                  |
| - Urban                                                                           | 1,003 km <sup>2</sup> (387.3 sq mi)                                                                              |
| - Metro                                                                           | 1,815 km <sup>2</sup> (700.8 sq mi)                                                                              |
| Elevation <sup>[4]</sup>                                                          | 2 m (7 ft)                                                                                                       |
| Population (1 October 2008) <sup>[5][6]</sup>                                     |                                                                                                                  |
| - City                                                                            | 755,269                                                                                                          |
| - Density                                                                         | 4,459/km <sup>2</sup> (11,548.8/sq mi)                                                                           |
| - Urban                                                                           | 1,364,422                                                                                                        |
| - Metro                                                                           | 2,158,372                                                                                                        |
| - Demonym                                                                         | Amsterdammer                                                                                                     |
| Time zone                                                                         | CET (UTC+1)                                                                                                      |
| - Summer (DST)                                                                    | CEST (UTC+2)                                                                                                     |
| Postcodes                                                                         | 1011 – 1109                                                                                                      |
| Area code(s)                                                                      | 020                                                                                                              |
| Website: <a href="http://www.amsterdam.nl">www.amsterdam.nl</a>                   |                                                                                                                  |

```
@prefix dbpedia <http://dbpedia.org/resource/> .
@prefix dbterm <http://dbpedia.org/property/> .
```

## dbpedia:Amsterdam

```
dbterm:officialName "Amsterdam" ;
```

```
dbterm:longd "4" ;
```

```
dbterm:longm "53" ;
```

```
dbterm:longs "32" ;
```

```
...
```

```
dbterm:leaderTitle "Mayor" ;
```

```
dbterm:leaderName dbpedia:Job_Cohen
```

```
;
```

```
...
```

```
dbterm:areaTotalKm "219" ;
```

```
...
```

## dbpedia:ABN\_AMRO

```
dbterm:location dbpedia:Amsterdam ;
```

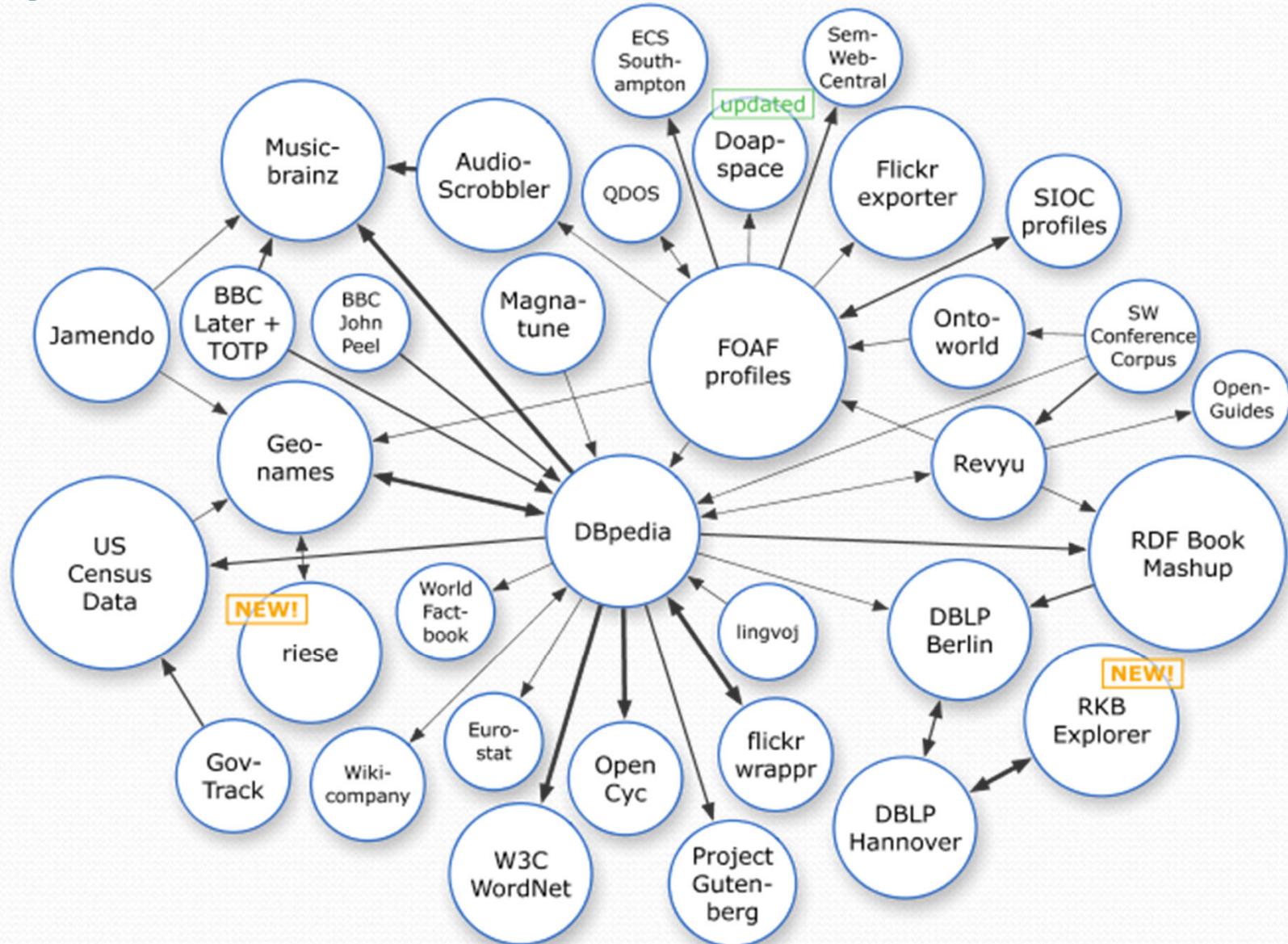
# Automatikus, felderíthető kapcsolatok

```
<http://dbpedia.org/resource/Amsterdam>
 owl:sameAs <http://rdf.freebase.com/ns/...> ;
 owl:sameAs <http://sws.geonames.org/2759793> ;
 ...
```

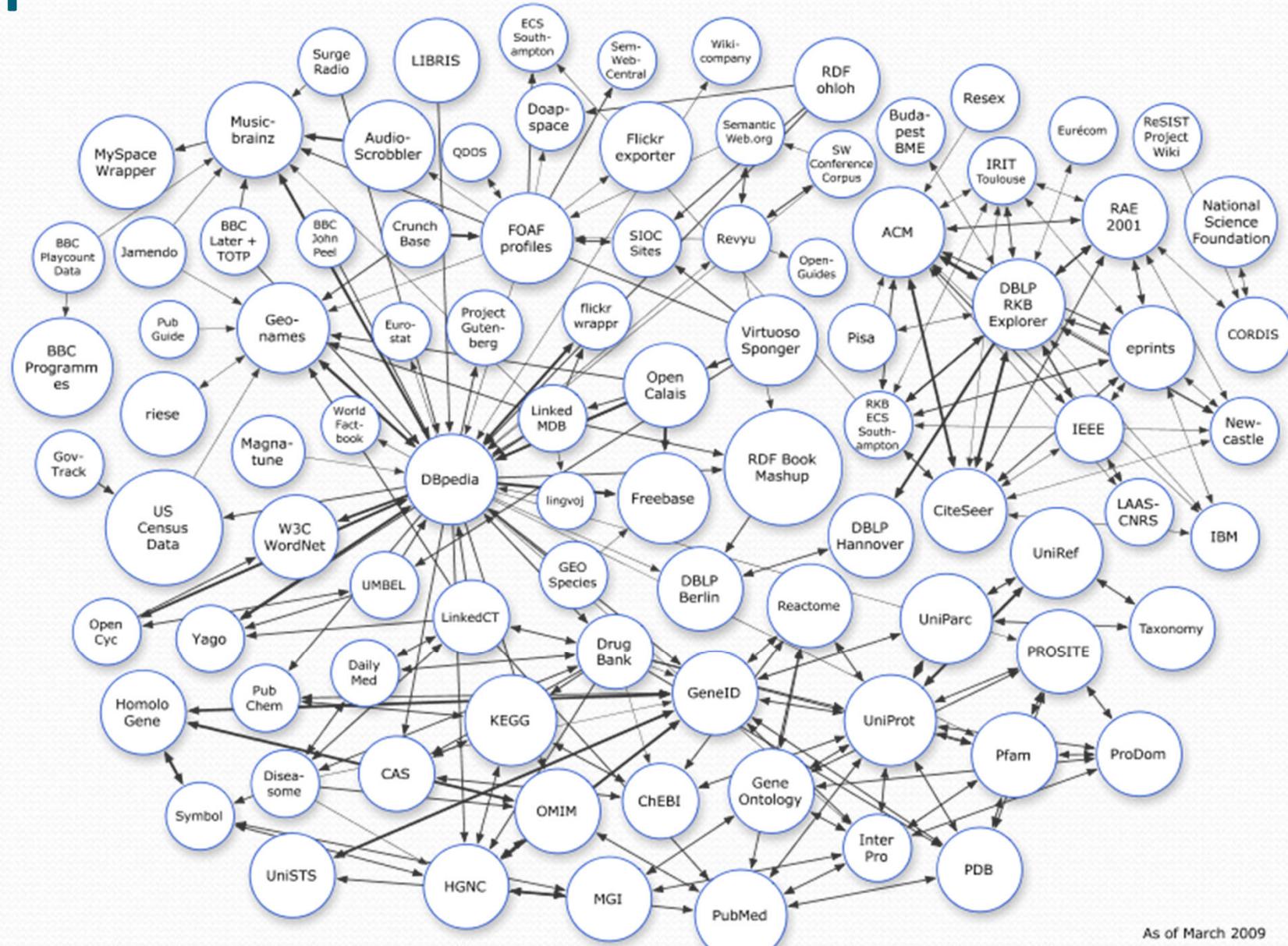
```
<http://sws.geonames.org/2759793>
 owl:sameAs <http://dbpedia.org/resource/Amsterdam>
 wgs84_pos:lat "52.3666667" ;
 wgs84_pos:long "4.8833333" ;
 geo:inCountry <http://www.geonames.org/countries/#NL> ;
 ...
```

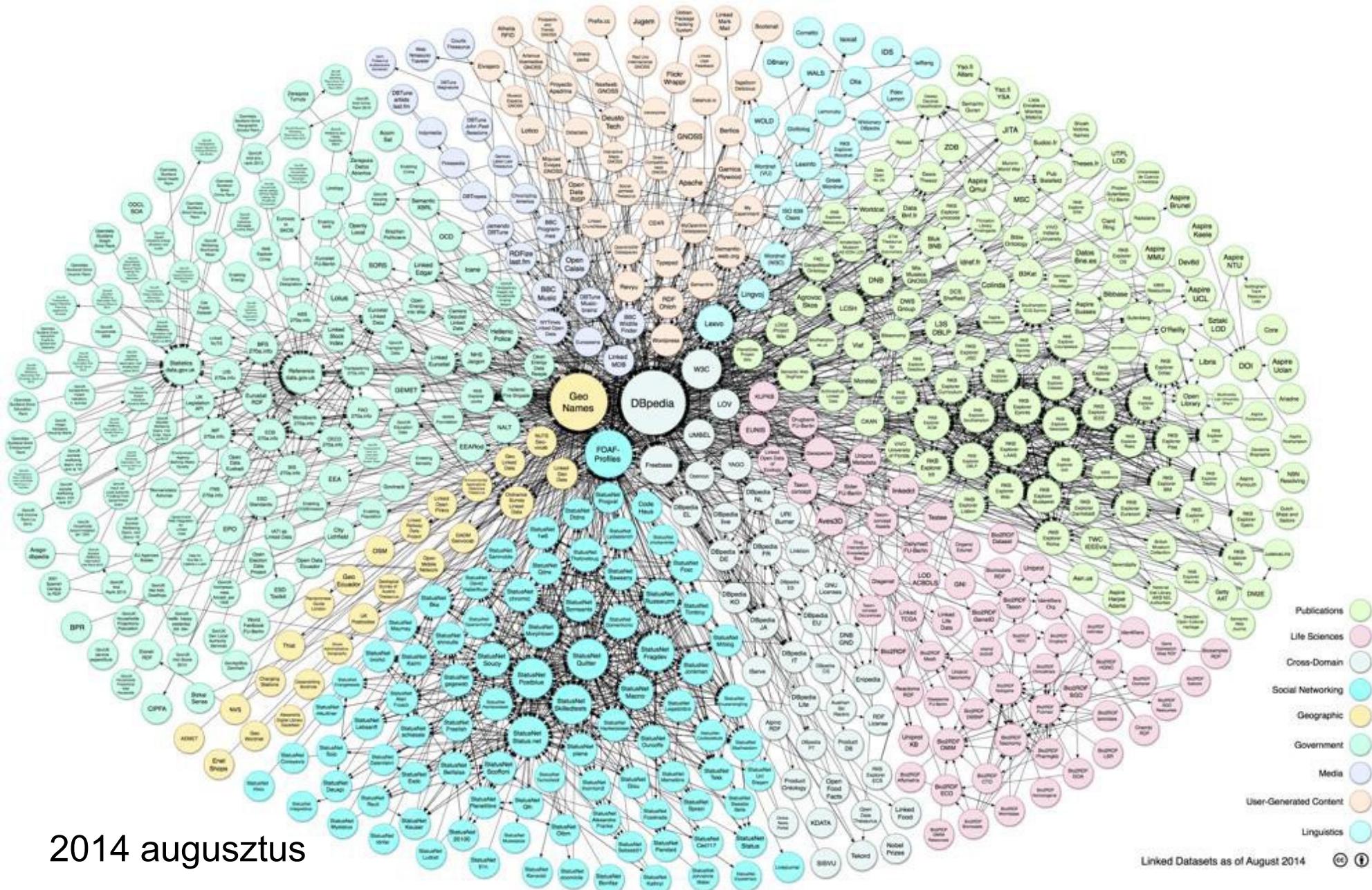
Szolgáltatások maguk döntik el honnan gyűjtenek információt

# A kapcsolati felhő – kiindulás 2008



# Kapcsolati felhő - 2009





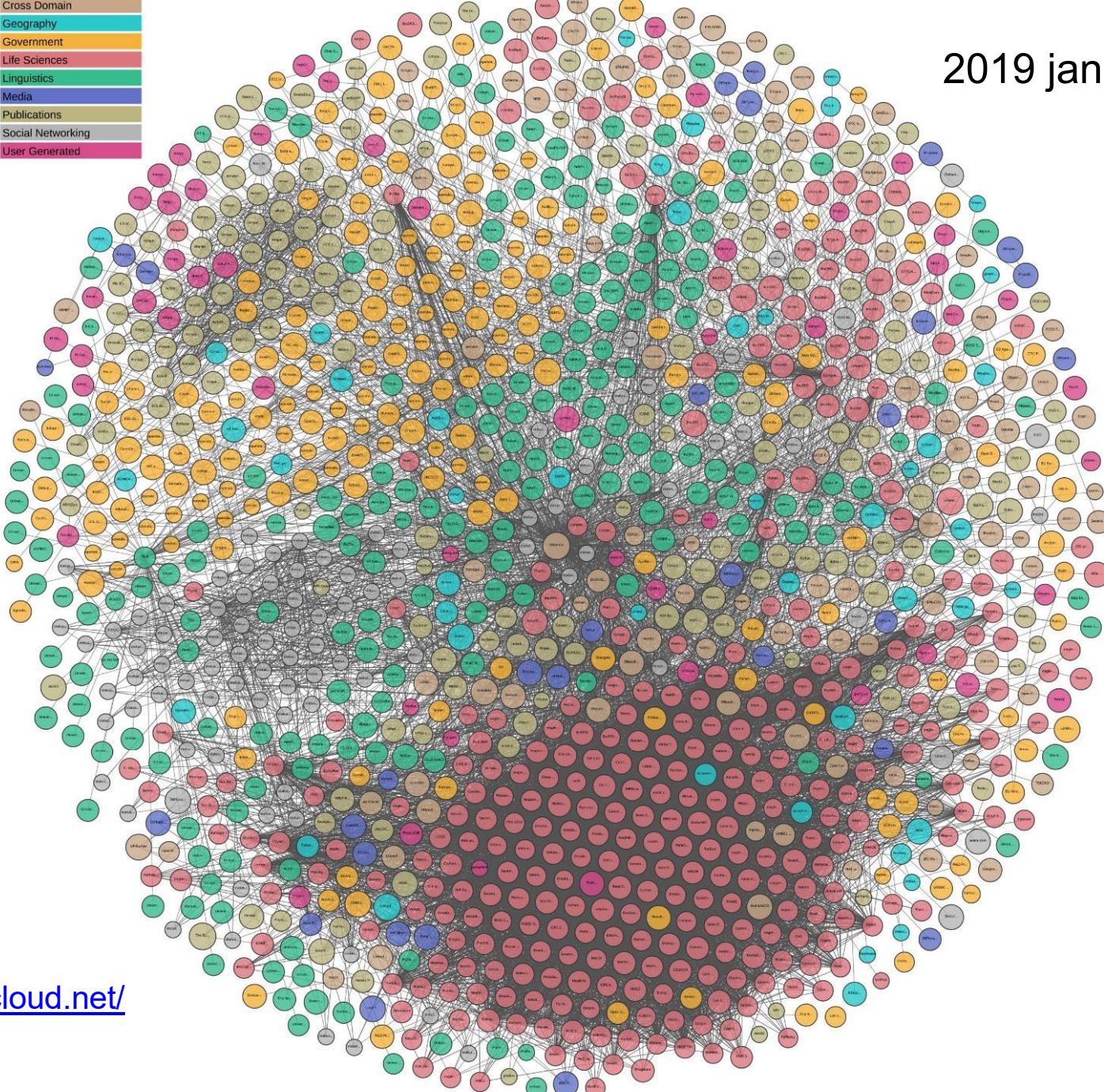
2014 augusztus

1/15/2019

2019 január

Legend

- Cross Domain
- Geography
- Government
- Life Sciences
- Linguistics
- Media
- Publications
- Social Networking
- User Generated



<https://lod-cloud.net/>

The Linked Open Data Cloud from lod-cloud.net

1/15/2019



# Alkalmazások...



# Alkalmazások...



# Tehát mi is a Linked Data?

- A Linked Data megközelítés (szabvány) realizálja az eredeti Szemantikus Web koncepciót az összekötöttség tekintetében
- Összekapcsol információ forrásokat FOAF, RDF, OWL és egyéb formátumokban

# A Linked Data szabályai

- minden adat megnevezése URI- keresztül
- A kapcsolatok URI-jai legyenek érvényes URL-ek
- Legyen egy URL lap, amely tartalmazza az URI-val hivatkozott adatot, ennek a lapnak az URL címe ne változzon
- Az adatok dokumentumokban, fájlokban legyenek elérhetőek a weben

# Célok

- Számos forrás tartalmaz hasonló adatokat a weben
  - Pl.személyes adatok
- Az adatokat kössük össze fogalomtáron keresztül, számos fogalomtár egyidejű felhasználásával.
- Az eredmény egy hatalmas háló, gráf.

# Fontosabb adattárak

[DBpedia](#) - a dataset containing extracted data from [Wikipedia](#); it contains about 2.18 million concepts described by 218 million triples, including abstracts in 11 different languages

[DBLP Bibliography](#) - provides bibliographic information about scientific papers; it contains about 800,000 articles, 400,000 authors, and approx. 15 million triples

[GeoNames](#) provides RDF descriptions of more than 6,500,000 geographical features worldwide.

[Revyu](#) - a Review service consumes and publishes Linked Data, primarily from [DBpedia](#).

[riese](#) - serving statistical data about 500 million Europeans (the first linked dataset deployed with [XHTML+RDFa](#))

[UMBEL](#) - a lightweight reference structure of 20,000 subject concept classes and their relationships derived from [OpenCyc](#), which can act as binding classes to external data; also has links to 1.5 million named entities from [DBpedia](#) and [YAGO](#)

[Sensorpedia](#) - A scientific initiative at [Oak Ridge National Laboratory](#) using a [RESTful](#) web architecture to link to sensor data and related sensing systems.

# SKOS

## Simple Knowledge Organisation Systems

Egyszerű megközelítés fogalomtárrak  
összekötésére

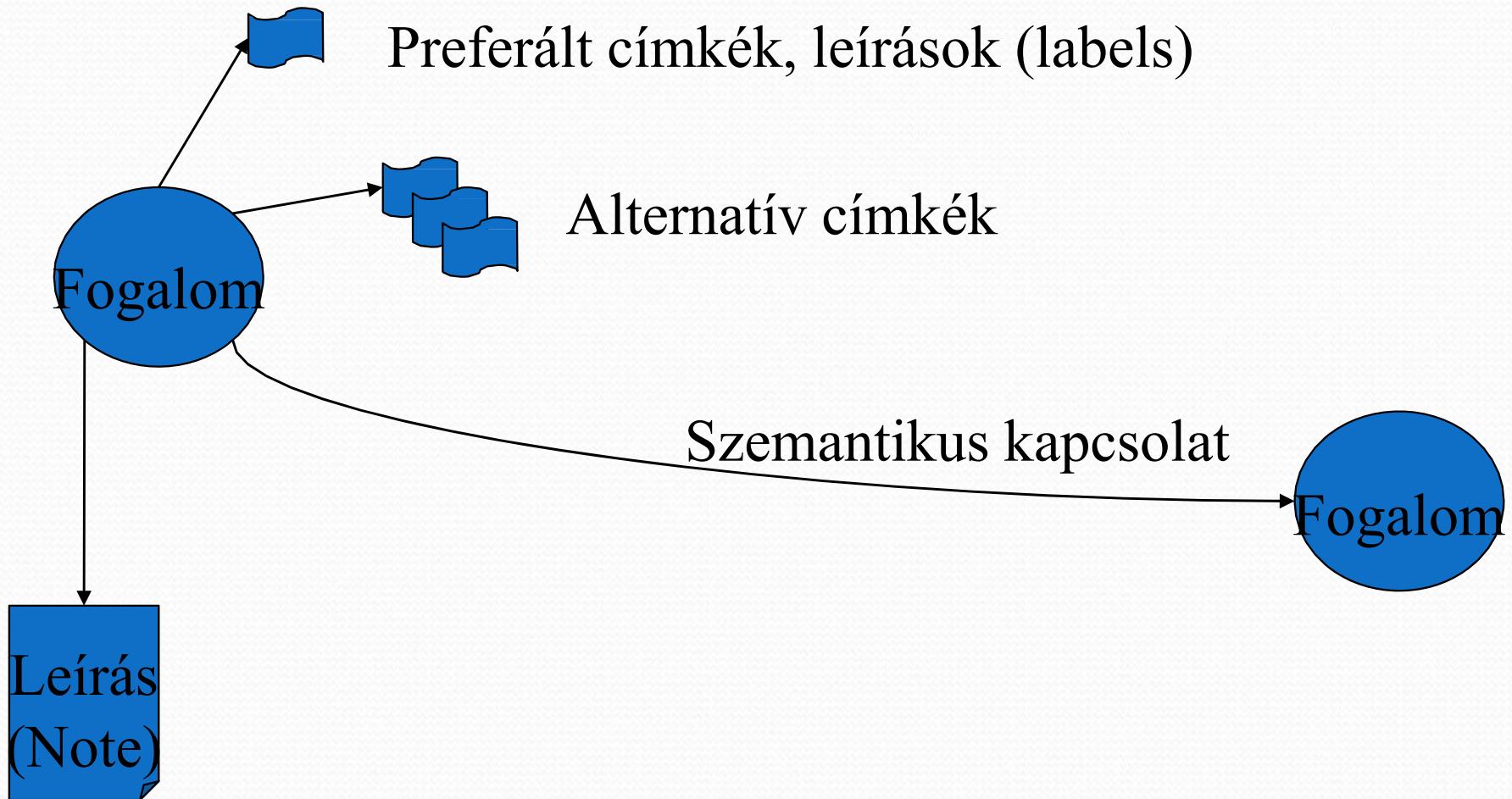
# SKOS alapok

- SKOS alapvető RDF szótárak
  - Pl. DC dokumentumok meta adatai
  - Pl. VCard kontakt részletek
  - Pl. FOAF social networks
  - Pl. OWL ontológiák

# Az SKOS Core és az OWL

- OWL egy W3C ajánlás, ontológiák leírására
  - Logika orientált
    - ⇒ Erős következtetés
    - ⇒ pontos szemantika
- SKOS
  - Nyelv orientált
    - ⇒ Egyszerűbb következtetés
    - ⇒ Rugalmas szemantika

# SKOS alapvető elemei



Pl. definíció, szkóp, háttér információk

# SKOS alapvető tulajdonságok

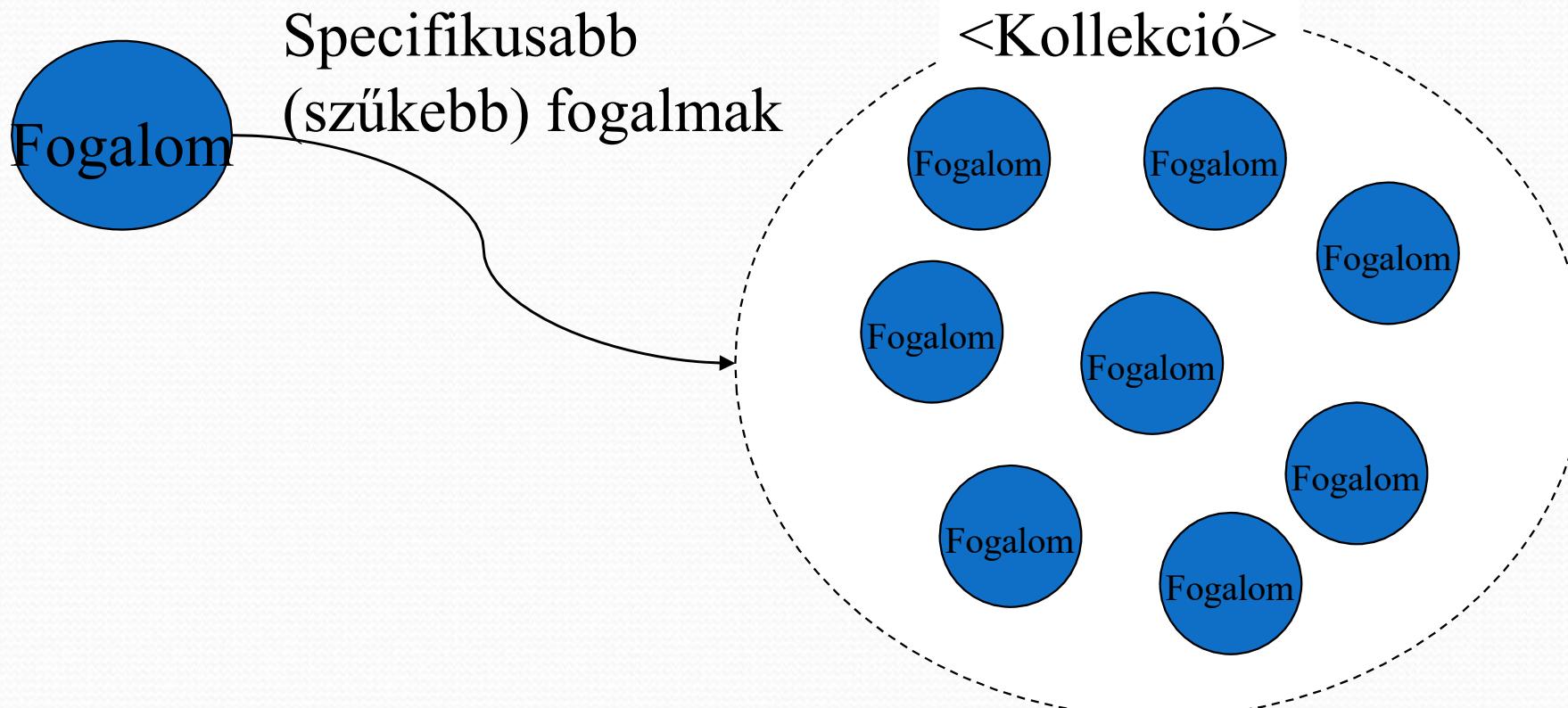
- Kiterjeszthető ...
    - Tulajdonságok és résztulajdonságok hierarchiája
  - Címkék
    - leírások
  - Egyedi relációk (általánosabb, specifikusabb fogalmak)
    - BroaderGeneric, BroaderInstantive ...
    - PartOf ...
- ⇒ Specializált alkalmazások

# SKOS Core bonyolultabb elemek

- Fogalmak több sémában
  - ⇒ Virtuális sémák - nézetek
  - ⇒ Séma újrafelhasználás
  - ⇒ Sémák összekapcsolása

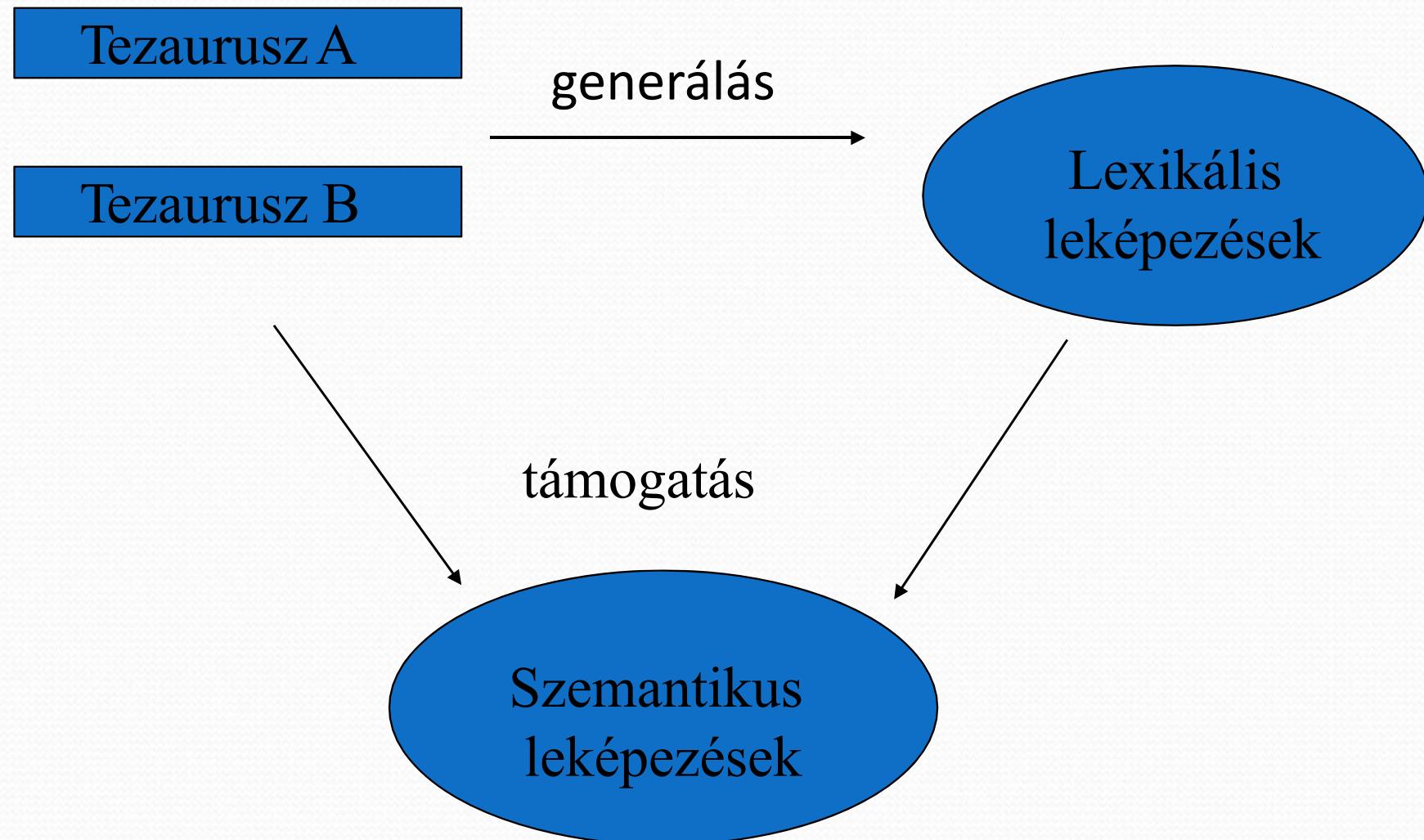
(Például SKOS alkalmazás virtuális integrációs megoldásokban.)

# SKOS alapok – fogalom gyűjtemények



# SKOS leképezések...

- Támogassuk a lexikális leképezéseket ...



# SKOS core szótár elemel

- skos:Concept: egy OWL osztály SKOS erőforrások leírására
  - <MyConcept> rdf:type skos:Concept
- skos:prefLabel, skos:altLabel, skos:hiddenLabel példányai az owl:DatatypeProperty osztálynak (gyakorlaton Protege-ben majd vizsgáljuk)
  - <MyResource>  
skos:prefLabel "animals"@en ;  
skos:altLabel "fauna"@en ;  
skos:hiddenLabel "aminals"@en ;  
skos:prefLabel "animaux"@fr ;  
skos:altLabel "faune"@fr .

# SKOS core szótár elemek II.

- skos:note, skos:changeNote, skos:definition, skos:editorialNote, skos:example, skos:historyNote skos:scopeNote példányai owl:ObjectProperty osztálynak.
- Példa:
  - <MyResource> skos:note <MyNote> .
  - <Protein> rdf:type owl:Class ;  
skos:definition "A physical entity consisting of a sequence of amino-acids; a protein monomer; a single polypeptide chain. An example is the EGFR protein."@en .

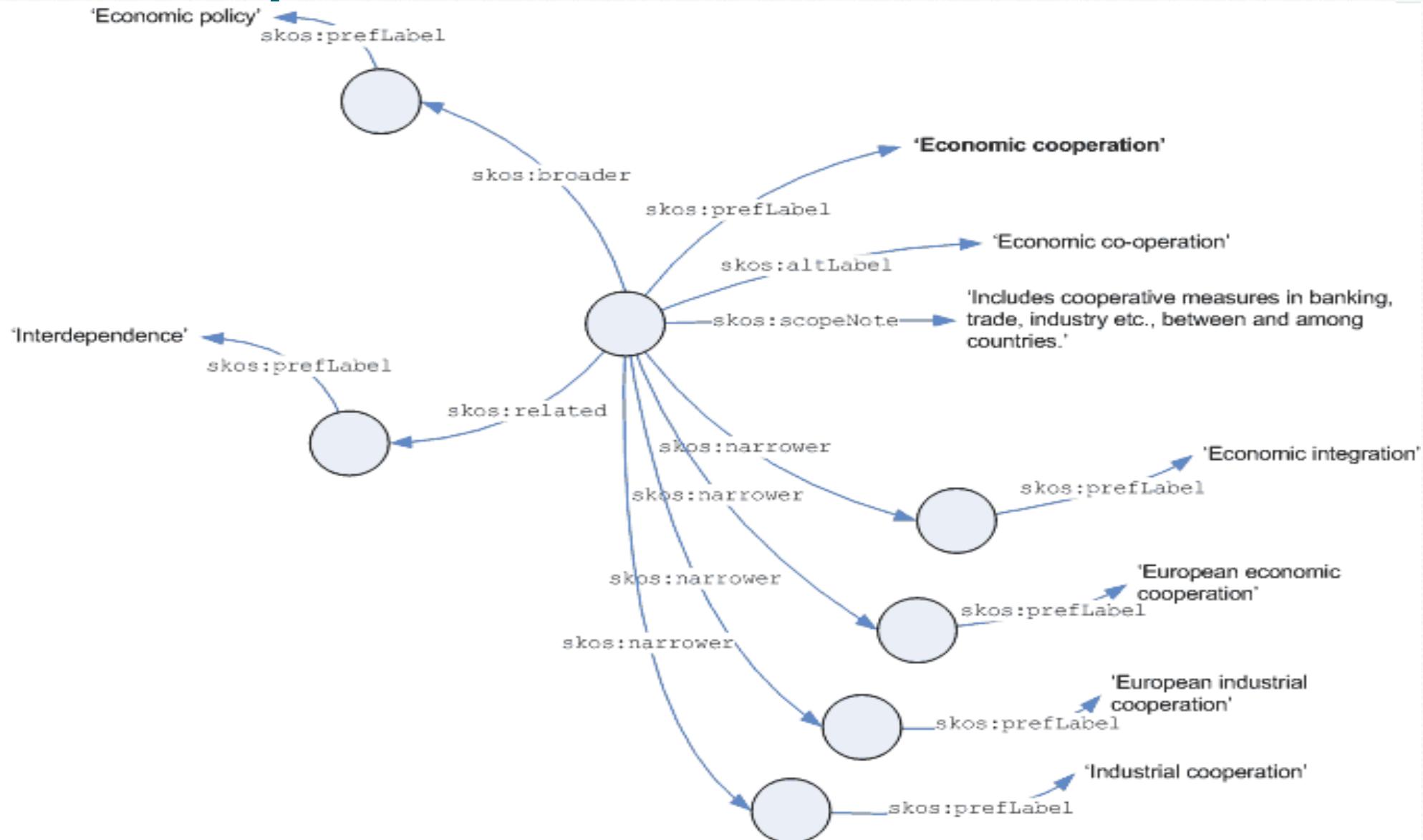
# SKOS core fogalom kapcsolatok

- skos:semanticRelation, skos:broader, skos:narrower, skos:related példányai az owl:ObjectProperty osztálynak:
  - <A> skos:broader <B> ; skos:related <C> .
  - <A> skos:broader <B,> <C> .  
<B> skos:broader <D> .  
<C> skos:broader <D> .

# SKOS core további elemek

- skos:Collection, skos:OrderedCollection, skos:member, skos:memberList
  - <MyCollection> rdf:type skos:Collection ; skos:member <X> , <Y> , <Z> .
- skos:mappingRelation, skos:exactMatch, skos:broadMatch, skos:narrowMatch, skos:relatedMatch szintén owl:ObjectProperty elemek
  - <A> skos:broadMatch <B> ; skos:relatedMatch <C> .
  - <A> skos:exactMatch <A> . <B> skos:broadMatch <B> . <C> skos:relatedMatch <C> .

# SKOS példa

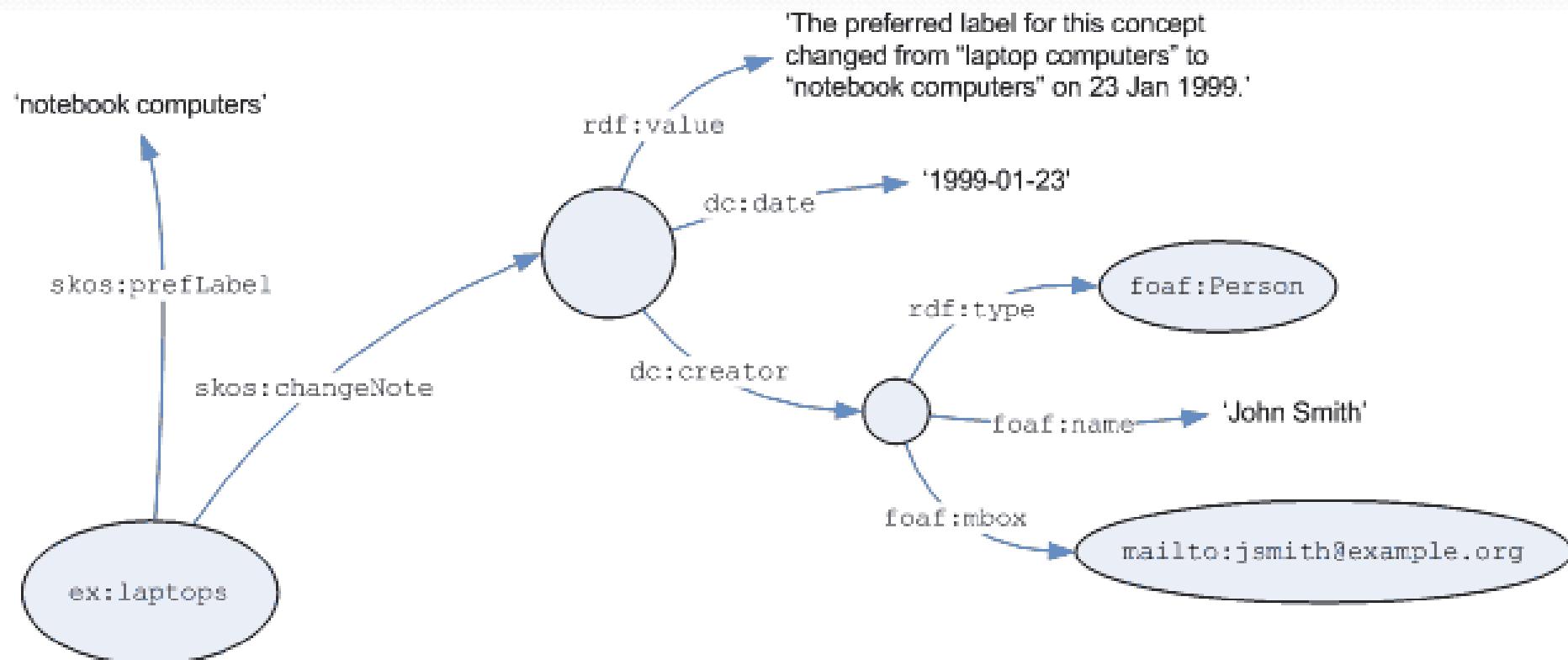


# SKOS példa (xml szintaxiszal)

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:skos="http://www.w3.org/2004/02/skos/core#">

<skos:Concept rdf:about="http://www.ukat.org.uk/thesaurus/concept/1750">
 <skos:prefLabel>Economic cooperation</skos:prefLabel>
 <skos:altLabel>Economic co-operation</skos:altLabel>
 <skos:scopeNote>Includes cooperative measures in banking, trade,
 industry etc., between and among countries. </skos:scopeNote>
 <skos:inScheme rdf:resource="http://www.ukat.org.uk/thesaurus"/>
 <skos:broader rdf:resource="http://www.ukat.org.uk/thesaurus/concept/4382"/>
 <skos:narrower rdf:resource="http://www.ukat.org.uk/thesaurus/concept/2108"/>
 <skos:narrower rdf:resource="http://www.ukat.org.uk/thesaurus/concept/9505"/>
 <skos:narrower rdf:resource="http://www.ukat.org.uk/thesaurus/concept/15053"/>
 <skos:narrower rdf:resource="http://www.ukat.org.uk/thesaurus/concept/18987"/>
 <skos:related rdf:resource="http://www.ukat.org.uk/thesaurus/concept/3250"/>
</skos:Concept>
</rdf:RDF>
```

# SKOS bővítése FOAF és DC névterekkel



```
prefix ex: <http://www.example.com/concepts#>
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix skos: <http://www.w3.org/2004/02/skos/core#>
prefix dc: <http://purl.org/dc/elements/1.1/>
prefix foaf: <http://xmlns.com/foaf/0.1/>
```

# SKOS bővítése FOAF és DC elemekkel

<rdf:RDF

```
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:skos="http://www.w3.org/2004/02/skos/core#"
 xmlns:dc="http://purl.org/dc/elements/1.1/"
 xmlns:foaf="http://xmlns.com/foaf/0.1/">

 <skos:Concept rdf:about="http://www.example.org/concepts#laptops">
 <skos:prefLabel>notebook computers</skos:prefLabel>
 <skos:changeNote rdf:parseType="Resource">
 <rdf:value>The preferred label for this concept changed from 'laptop
 computers' to 'notebook computers' on 23 Jan 1999.</rdf:value>
 <dc:creator>
 <foaf:Person>
 <foaf:name>John Smith</foaf:name>
 <foaf:mbox rdf:resource="mailto:jsmith@example.org"/>
 </foaf:Person>
 </dc:creator>
 <dc:date>1999-01-23</dc:date>
 </skos:changeNote>
</skos:Concept>

</rdf:RDF>
```

Integrációs és ellenőrzési technikák  
VIMIAC04, 2021. tavasz

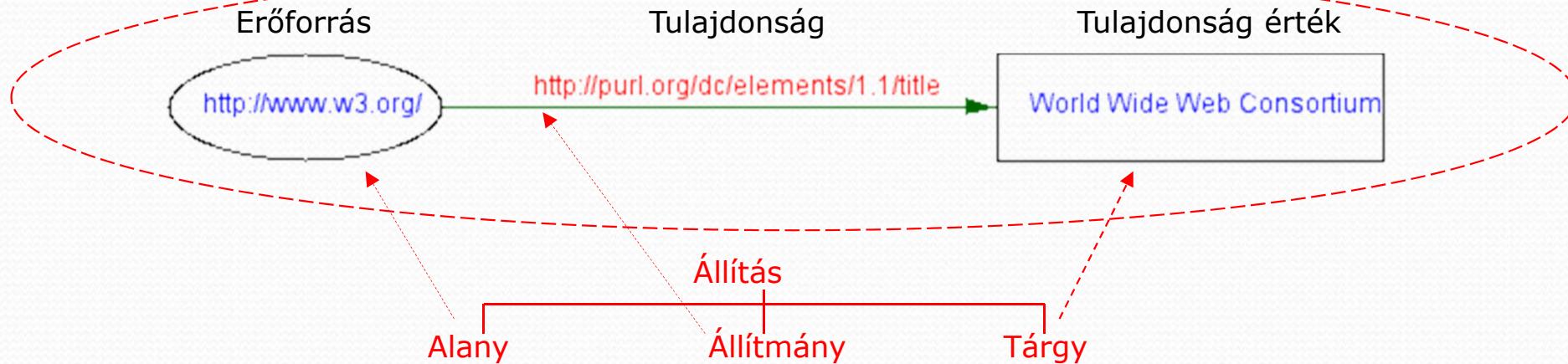
RDFS

(Resource Description Frameworks  
Schema)

*Méréstechnika és Információs Rendszerek Tanszék*  
<https://www.mit.bme.hu/oktatas/targyak/vimiac04>

# RDF

- Eredetileg webes metaadat kezelés javasolt megközelítése volt.
- Objektum típusok: erőforrás, tulajdonság, és állítások.
- RDF XML szintaxis
- Alkalmazási terület független

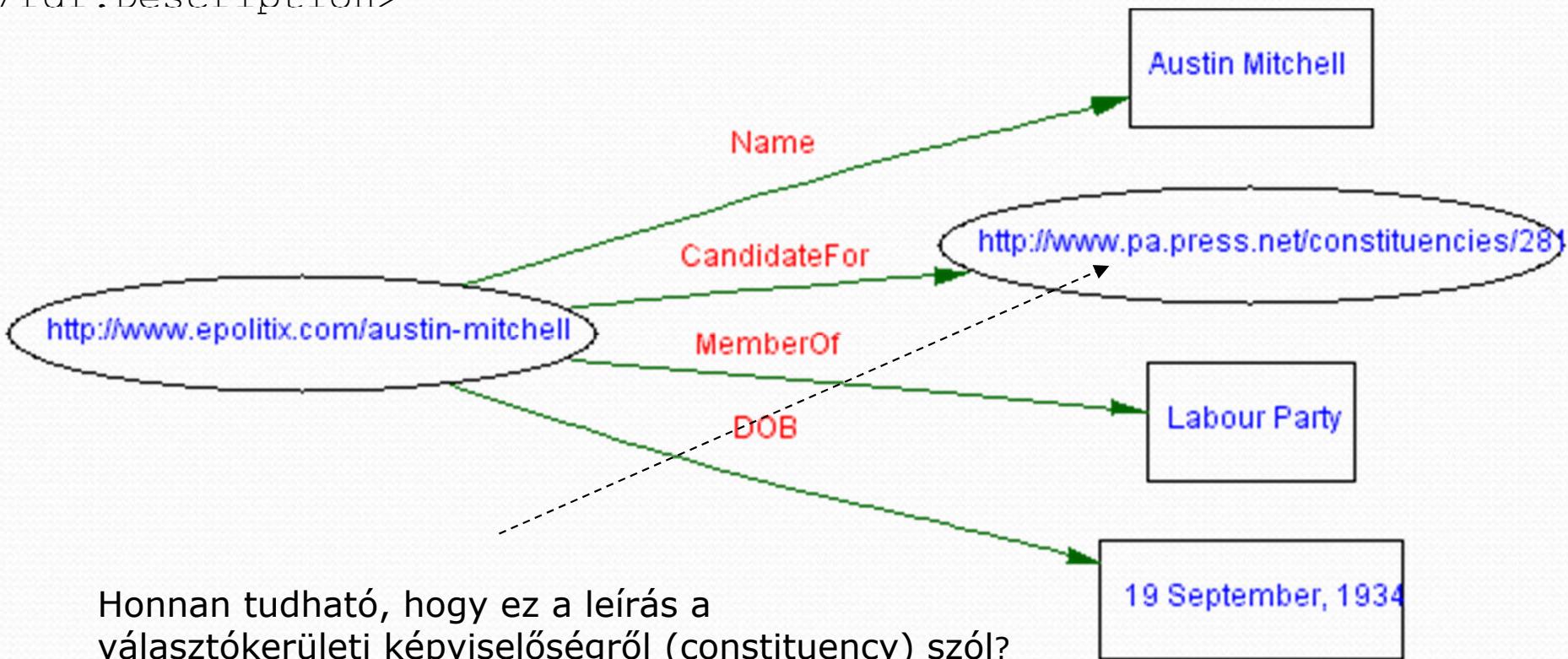


# Mire használható az RDFS

- Tekinthető RDF kifejezés szótárként is, hagyományos megközelítésként séma
- Alkalmazási területek szemantikájának leírása.
- Tulajdonságok újrafelhasználhatóságának biztosítása. Értelmezési tartomány és értékkészlet megadása
- Tulajdonságok osztályokhoz, alosztályokhoz rendelése specifikálható

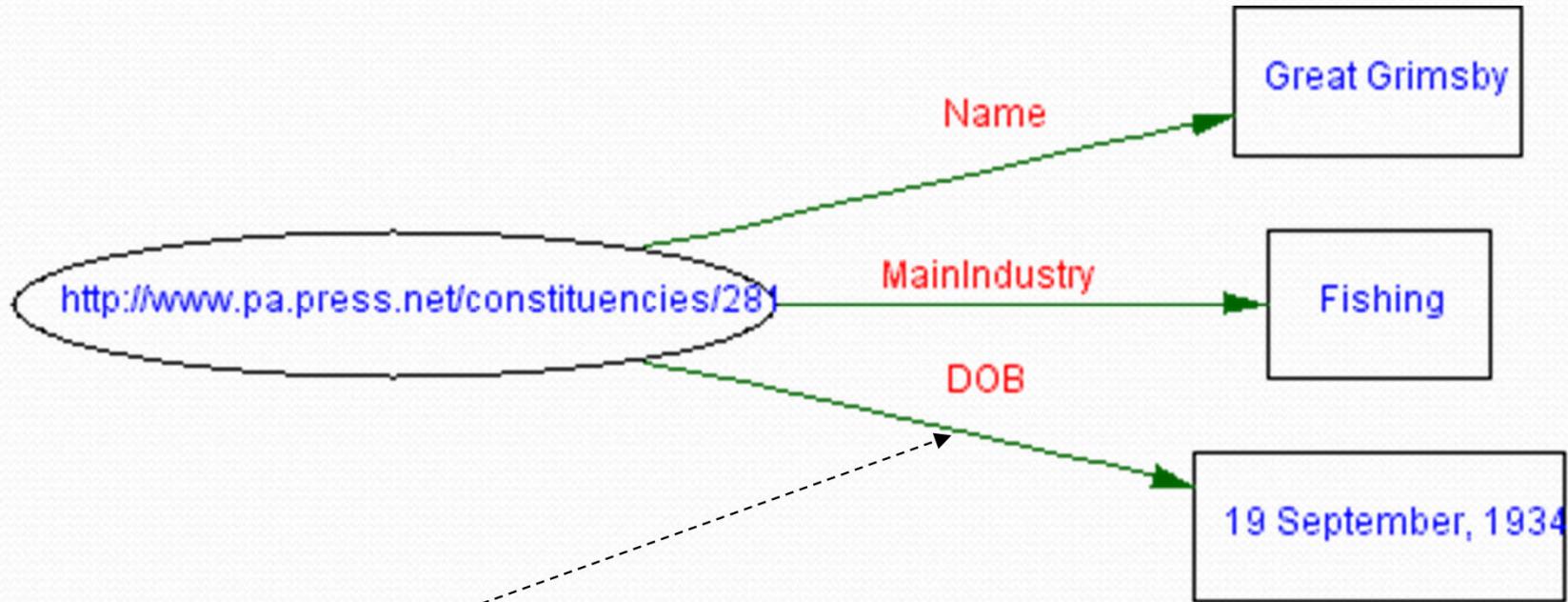
# Adat integrációs probléma

```
<rdf:Description rdf:about="http://www.epolitix.com/austin-mitchell">
 <epx:Name>Austin Mitchell</epx:Name>
 <epx:CandidateFor
 rdf:resource="http://www.pa.press.net/constituencies/281" />
 <epx:MemberOf>Labour Party</epx:MemberOf>
 <epx:DOB>19 September, 1934</epx:DOB>
</rdf:Description>
```



# Adat integrációs probléma

```
<rdf:Description rdf:about="http://www.pa.press.net/constituencies/281">
 <epx:Name>Great Grimsby</epx:Name>
 <epx:MainIndustry>Fishing</epx:MainIndustry>
 <epx:DOB>19 September, 1934</epx:DOB>
</rdf:Description>
```



Egy választókerületnek van születési dátuma?

# Állítások validációja

- Alkalmas-e az alany az állításhoz?

```
{
 epx:CandidateFor,
 [http://www.epolitix.com/austin-mitchell],
 [http://www.microsoft.com/]
}
```

- Alkalmas-e az állítmány az alanyhoz?

```
{ epx:DOB,
[http://www.pa.press.net/constituencies/281],
"19 September, 1934" }
```

# Erőforrások típizálása

- Honnan tudjuk, hogy ez egy választókörzet?

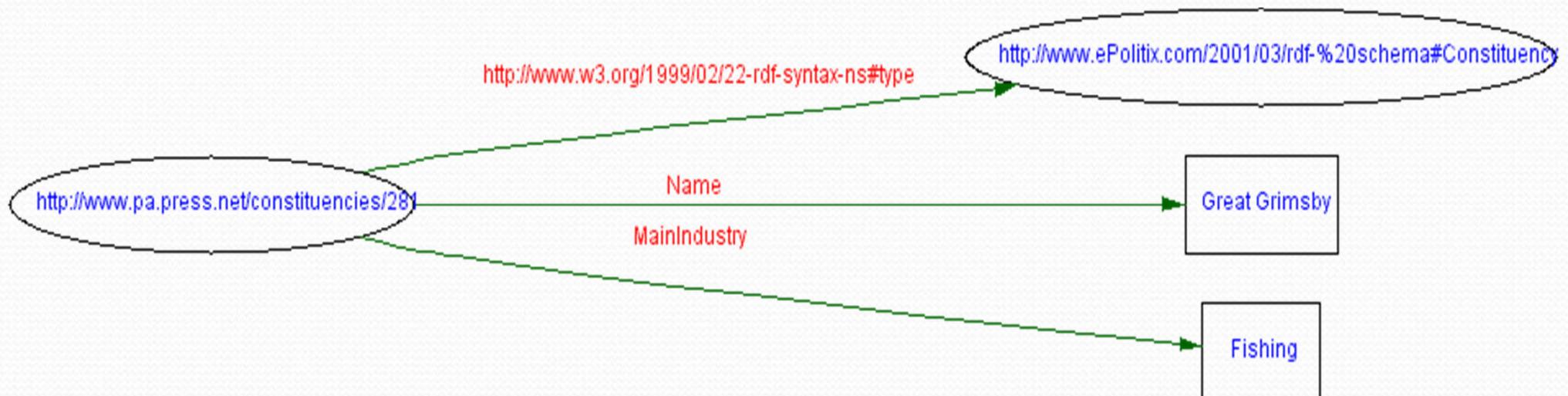
```
<rdf:Description rdf:about="http://www.pa.press.net/constituencies/281">
 <epx:Name>Great Grimsby</epx:Name>
 <epx:MainIndustry>Fishing</epx:MainIndustry>
</rdf:Description>
```

- „választókerület” típusba kell sorolni az erőforrást.

# Erőforrás típus: Consistency

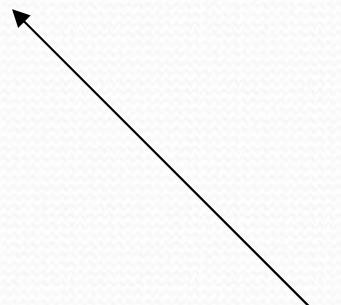
```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:epx="http://www.ePolitix.com/2001/03/rdf-schema#" >

 <rdf:Description rdf:about=
 "http://www.pa.press.net/constituencies/281">
 <rdf:type resource=
 "http://www.ePolitix.com/2001/03/rdf-schema#Constituency"/>
 <epx:Name>Great Grimsby</epx:Name>
 <epx:MainIndustry>Fishing</epx:MainIndustry>
 </rdf:Description>
</rdf:RDF>
```



# Tulajdonság értékek specifikálása: megfelelő állítmányok használata az alanyokhoz

```
{
 epx:CandidateFor,
 [http://www.epolitix.com/austin-mitchell],
 [http://www.microsoft.com/]
}
```

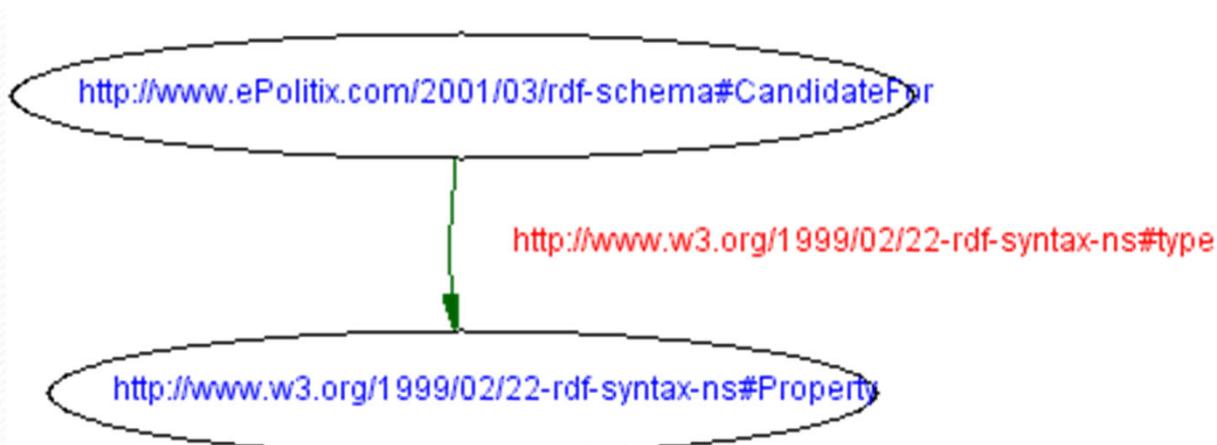


Érvénytelen: Nem választókerület

# Tulajdonság értékek specifikálása: megfelelő állítmányok használata az alanyokhoz

Hozzunk létre: *rdf:property CandidateFor*

```
<rdf:Description
 rdf:about="http://www.ePolitix.com/2001/03/rdf-schema#CandidateFor"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
 <rdf:type
 resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property" />
</rdf:Description>
```



# Tulajdonság értékek specifikálása: megfelelő állítmányok használata az alanyokhoz

*A CandidateFor tulajdonság használatát korlátozzuk, hogy csak Constituency típushoz legyen alkalmazható*

```
<rdf:Property
 rdf:about="http://www.ePolitix.com/2001/03/rdf-schema#CandidateFor" >
 <rdfs:range rdf:resource=
 "http://www.ePolitix.com/2001/03/rdf-schema#Constituency" />
</rdf:Property>
```



# RDFS Névterek

Az RDFS séma névteréhez tartozó prefix,  
URI: <http://www.w3.org/2000/01/rdf-schema#>

```
<rdf:Property
 rdf:about="http://www.ePolitix.com/2001/03/rdf-schema#CandidateFor">
 <rdfs:range rdf:resource=
 "http://www.ePolitix.com/2001/03/rdf-schema#Constituency"/>
</rdf:Property>
```



# rdfs:range

A *ConstraintProperty* használata megadja, hogy az adott osztályok milyen tulajdonságúak lehetnek.

A *range* tulajdonság értéke minden esetben egy osztály. Egy tulajdonsághoz maximum egy *range* értéket lehet megadni.

```
<rdf:Property
 rdf:about="http://www.ePolitix.com/2001/03/rdf-schema#CandidateFor" >
 <rdfs:range rdf:resource=
 "http://www.ePolitix.com/2001/03/rdf-schema#Constituency" />
</rdf:Property>
```



# Kifejezések validációja

- Alkalmas-e az alany az állításhoz?

```
{
 epx:CandidateFor,
 [http://www.epolitix.com/austin-mitchell],
 [http://www.microsoft.com/]
}
```

- Az állítmány alkalmazható-e az alanyhoz?

```
{ epx:DOB, [http://www.pa.press.net/constituencies/281],
 "19 September, 1934" }
```

# Erőforrás tulajdonságok specifikálása: állítmányok meghatározása alanyokhoz

Korlátozzuk a *CandidateFor* tulajdonságot, hogy csak *Person* típusú alanyokhoz legyen használható

```
rdf:Property
 rdf:about="http://www.ePolitix.com/2001/03/rdf-schema#CandidateFor"
 <rdfs:domain
 rdf:resource="http://www.Schemas.org/2001/01/rdf-schema#Person" />
 <rdfs:range
 rdf:resource=
 "http://www.ePolitix.com/2001/03/rdf-schema#Constituency" />
</rdf:Property>
```



# rdfs:domain

A *domain* tulajdonság értéke minden esetben egy osztály. Egy tulajdonsághoz maximum egy *domain* értéket lehet megadni.

```
rdf:Property
 rdf:about="http://www.ePolitix.com/2001/03/rdf-
schema#CandidateFor">
 <rdfs:domain
 rdf:resource="http://www.Schemas.org/2001/01/rdf-
schema#Person" />
 <rdfs:range
 rdf:resource="http://www.ePolitix.com/2001/03/rdf-
schema#Constituency" />
</rdf:Property>
```



# Specifikációk:

## rdfs:ConstraintProperty

Ez az erőforrás az *rdf:Property* alosztályát definiálja, melynek példányai kényszerek megadására alkalmazható tulajdonságok.

Ez az osztály alosztálya az *rdfs:ConstraintResource* osztálynak, ennek az osztálynak azt a részhalmazát definiálja, amelyek tulajdonságok.

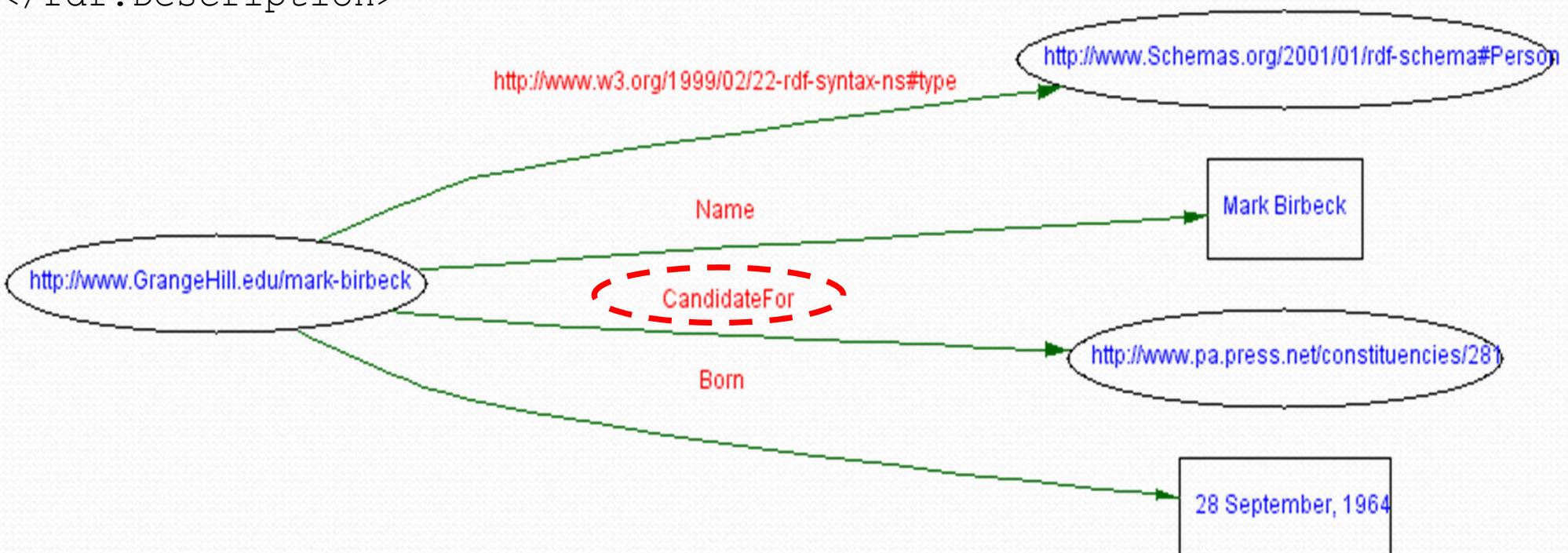
Az *rdfs:domain* és az *rdfs:range* példányai *rdfs:ConstraintProperty* osztálynak.

## rdfs:ConstraintResource

Ez az erőforrás az *rdfs:Resource* alosztályát definiálja, melynek példányai olyan RDFS séma struktúrák, amelyek alkalmazhatóak kényszerek megadásában. Az osztály szerepe, hogy mechanizmusokat lehessen létrehozni RDF feldolgozó motorok számára, amellyel az RDF modell elemei értékelhetőek ki.

# Tulajdonságok hasznosítása: hamis iskolai választás példa (RDF-statements)

```
<rdf:Description rdf:about="http://www.GrangeHill.edu/mark-birbeck">
 <rdf:type
 resource="http://www.Schemas.org/2001/01/rdf-schema#Person"/>
 <gh:Name>Mark Birbeck</gh:Name>
 <epx:CandidateFor
 rdf:resource="http://www.pa.press.net/constituencies/281" />
 <gh:Born>28 September, 1964</gh:Born>
</rdf:Description>
```



# Tulajdonságok hasznosítása : Esemény web lap (egy RDFS részlet)

```
<rdf:Property rdf:about="http://www.ePolitix.com/2001/03/rdf-schema#Region">
 <rdfs:domain
 rdf:resource="http://www.WhatsOnWhere.com/2001/01/rdf-schema#Event" />
</rdf:Property>
```



# Több típus megadása

```
<rdf:Description
 rdf:about="http://www.epolitix.com/austin-mitchell"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:s="http://www.Schemas.org/2001/01/rdf-schema#"
 xmlns:epx="http://www.ePolitix.com/2001/03/rdf-schema#" >

 <rdf:type
 resource="http://www.Schemas.org/2001/01/rdf-schema#Person" />
 <s:Name>Austin Mitchell</s:Name>
 <s:DOB>19 September, 1934</s:DOB>
 <rdf:type
 resource="http://www.ePolitix.com/2001/03/rdf-schema#Candidate" />
 <epx:CandidateFor
 rdf:resource="http://www.pa.press.net/constituencies/281" />
 <epx:MemberOf>Labour Party</epx:MemberOf>
</rdf:Description>
```

# Több típus megadása



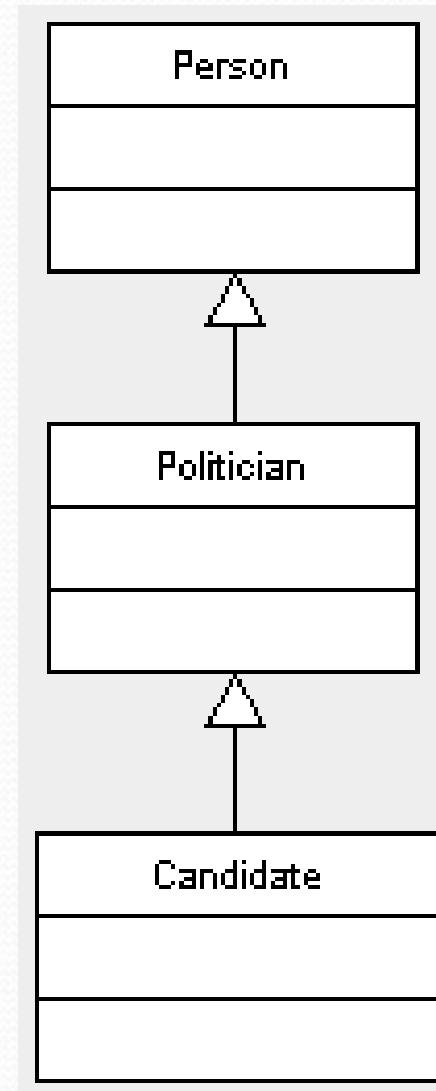
# Osztályok az RDFS-ben

- Nincsenek metódusok, csak tulajdonságok
- Tulajdonság centrikus. OO világban az osztály metódusok és tulajdonságok halmazával definiált. RDF tulajdonságot az alkalmazhatóság (osztály) definiálja.
- Tulajdonságok osztályok kapcsolata nélkül is definiálható.

# Kategória rendszer építése: rdfs:Class

```
<rdfs:Class
 rdf:about="http://www.Schemas.org/2001/01/rdf-schema#Person"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
/>
```

# Alosztályok létrehozása



# Alosztályok létrehozása (XML szintaxis)

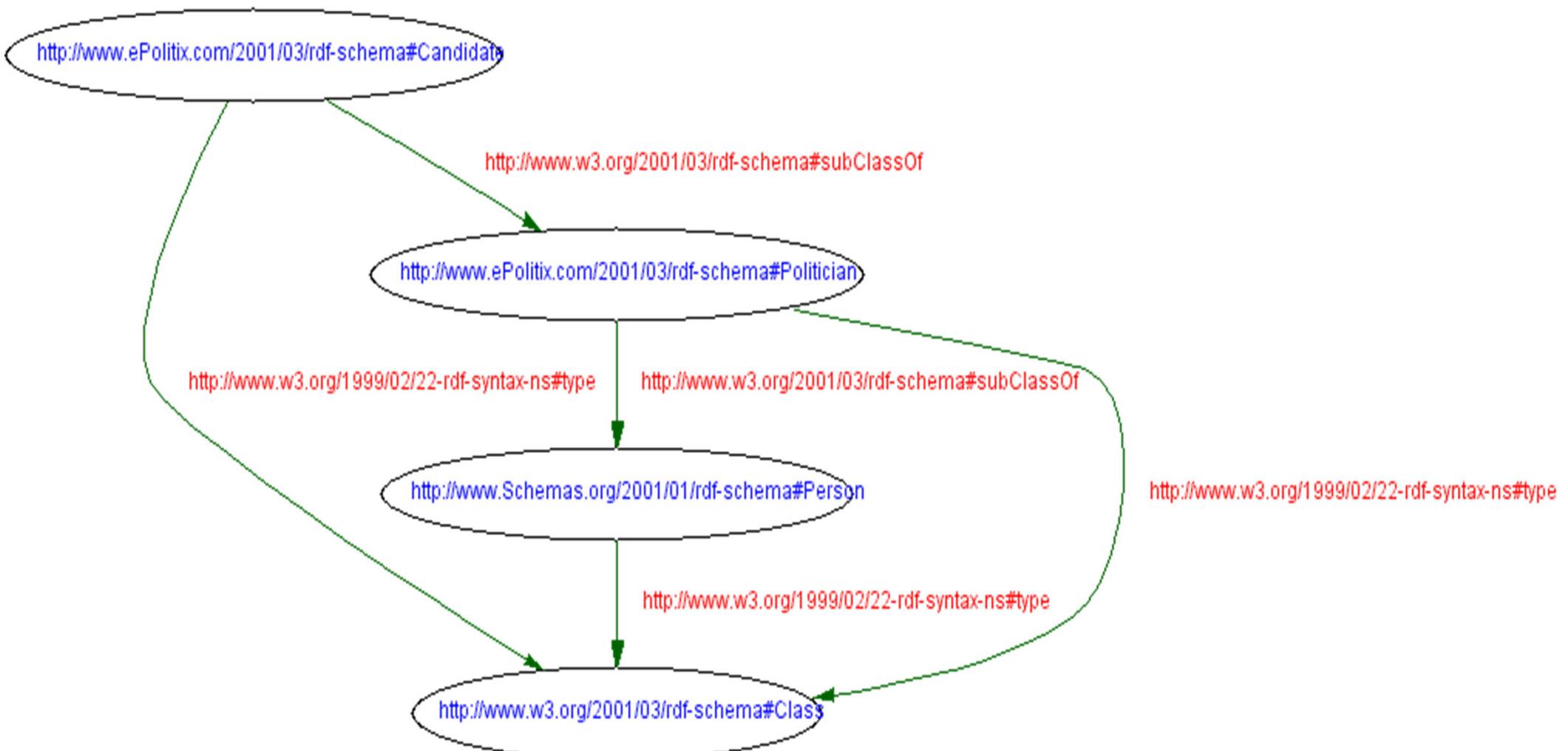
```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#" >

 <rdfs:Class rdf:about="http://www.Schemas.org/2001/01/rdf-schema#Person">

 <rdfs:Class
 rdf:about="http://www.ePolitix.com/2001/03/rdf-schema#Politician">
 <rdfs:subClassOf
 rdf:resource="http://www.Schemas.org/2001/01/rdf-schema#Person">
 </rdfs:Class>

 <rdfs:Class
 rdf:about="http://www.ePolitix.com/2001/03/rdf-schema#Candidate">
 <rdfs:subClassOf
 rdf:resource=
 "http://www.ePolitix.com/2001/03/rdf-schema#Politician" >
 </rdfs:Class>
</rdf:RDF>
```

# Alosztályok létrehozása

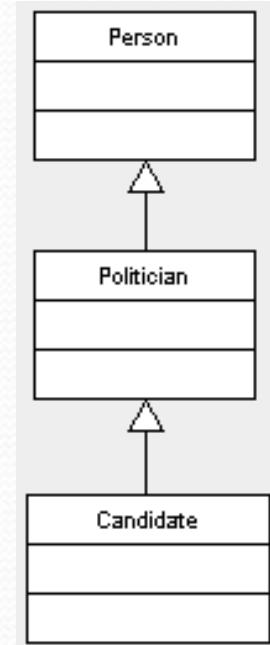


# Alosztályok alkalmazása: RDFS Property

```
<rdf:RDF>
 <rdf:Property rdf:about="http://www.Schemas.org/2001/01/rdf-schema#Name">
 <rdfs:domain rdf:resource="http://www.Schemas.org/2001/01/rdf-schema#Person">
 <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal">
 </rdf:Property>
 <rdf:Property rdf:about="http://www.Schemas.org/2001/01/rdf-schema#DOB">
 <rdfs:domain rdf:resource="http://www.Schemas.org/2001/01/rdf-schema#Person">
 <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal">
 </rdf:Property>
</rdf:RDF>

<rdf:Property
 rdf:ID="MemberOf">
 <rdfs:domain rdf:resource="#Politician" >
 <rdfs:range rdf:resource="#PoliticalParty" >
</rdf:Property>

<rdf:Property
 rdf:ID="CandidateFor">
 <rdfs:domain rdf:resource="#Candidate" >
 <rdfs:range rdf:resource="#Constituency" >
</rdf:Property>
```

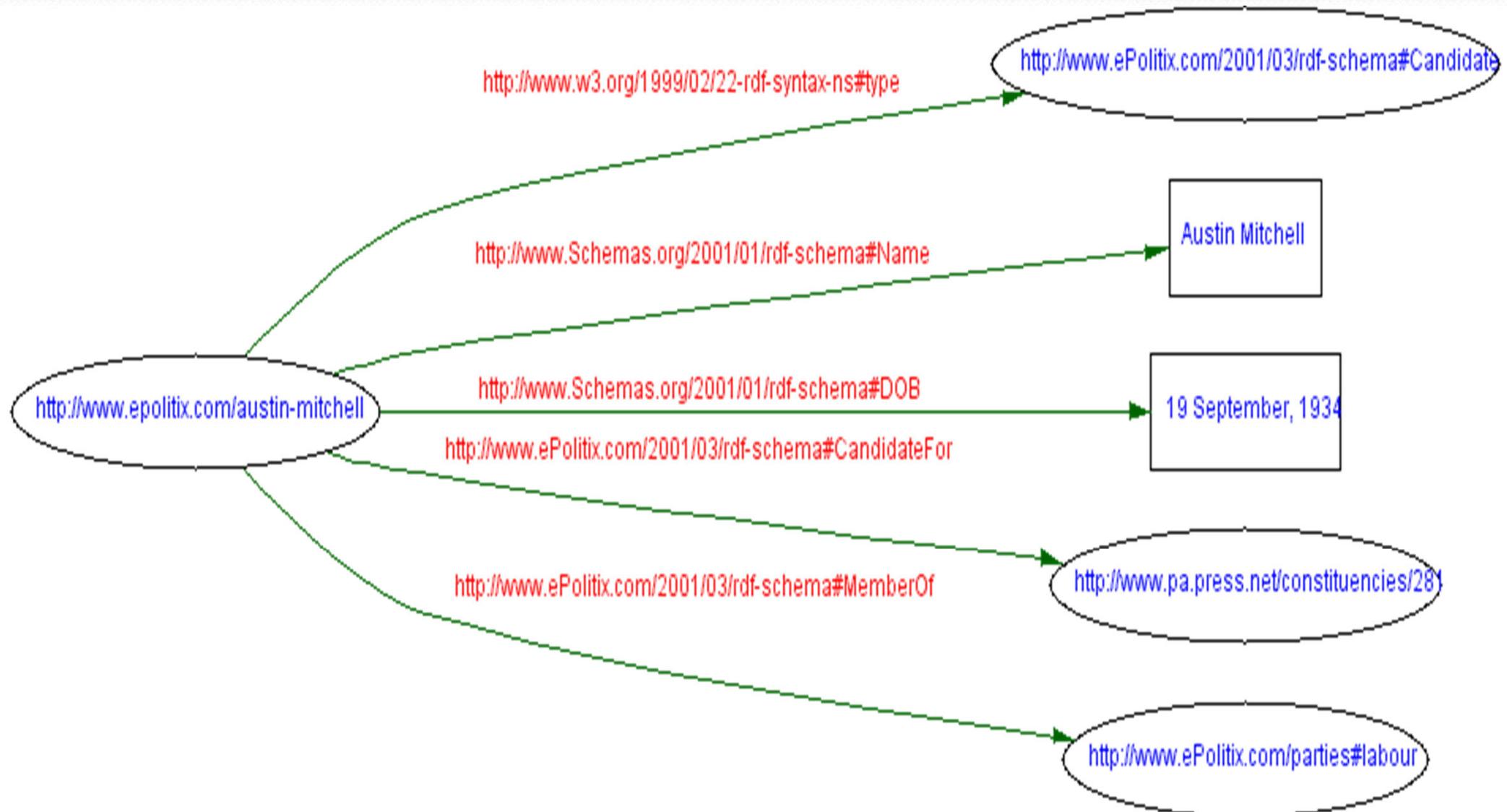


# Alosztályok alkalmazása : RDF-statements

```
<rdf:Description
 rdf:about="http://www.epolitix.com/austin-mitchell"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:s="http://www.Schemas.org/2001/01/rdf-schema#"
 xmlns:epx="http://www.ePolitix.com/2001/03/rdf-schema#" >

 <rdf:type resource="http://www.ePolitix.com/2001/03/rdf-
schema#Candidate" />
 <s:Name>Austin Mitchell</s:Name>
 <s:DOB>19 September, 1934</s:DOB>
 <epx:CandidateFor rdf:resource=
 "http://www.pa.press.net/constituencies/281" />
 <epx:MemberOf rdf:resource=
 "http://www.ePolitix.com/parties#labour" />
</rdf:Description>
```

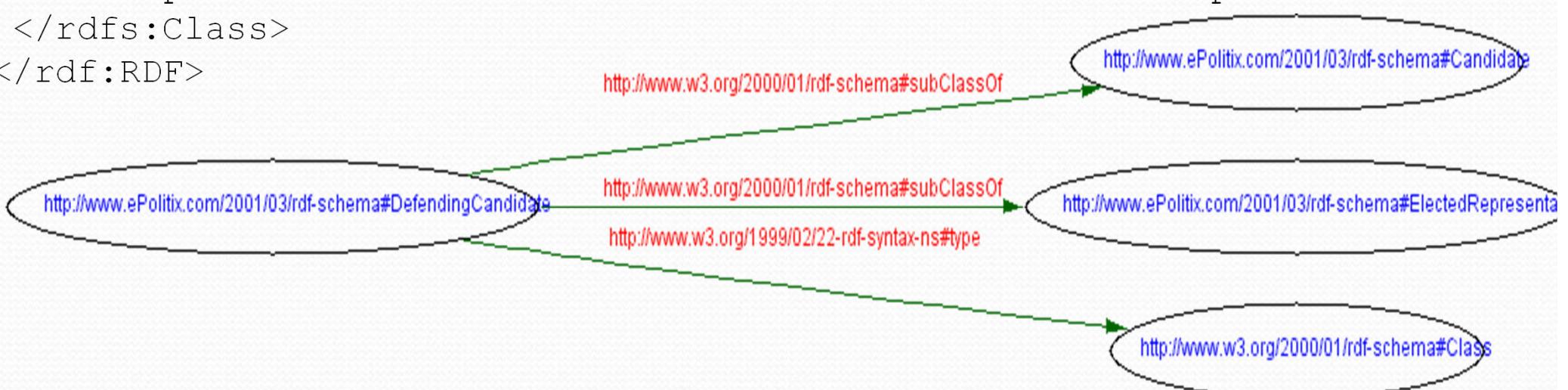
# Alosztályok alkalmazása



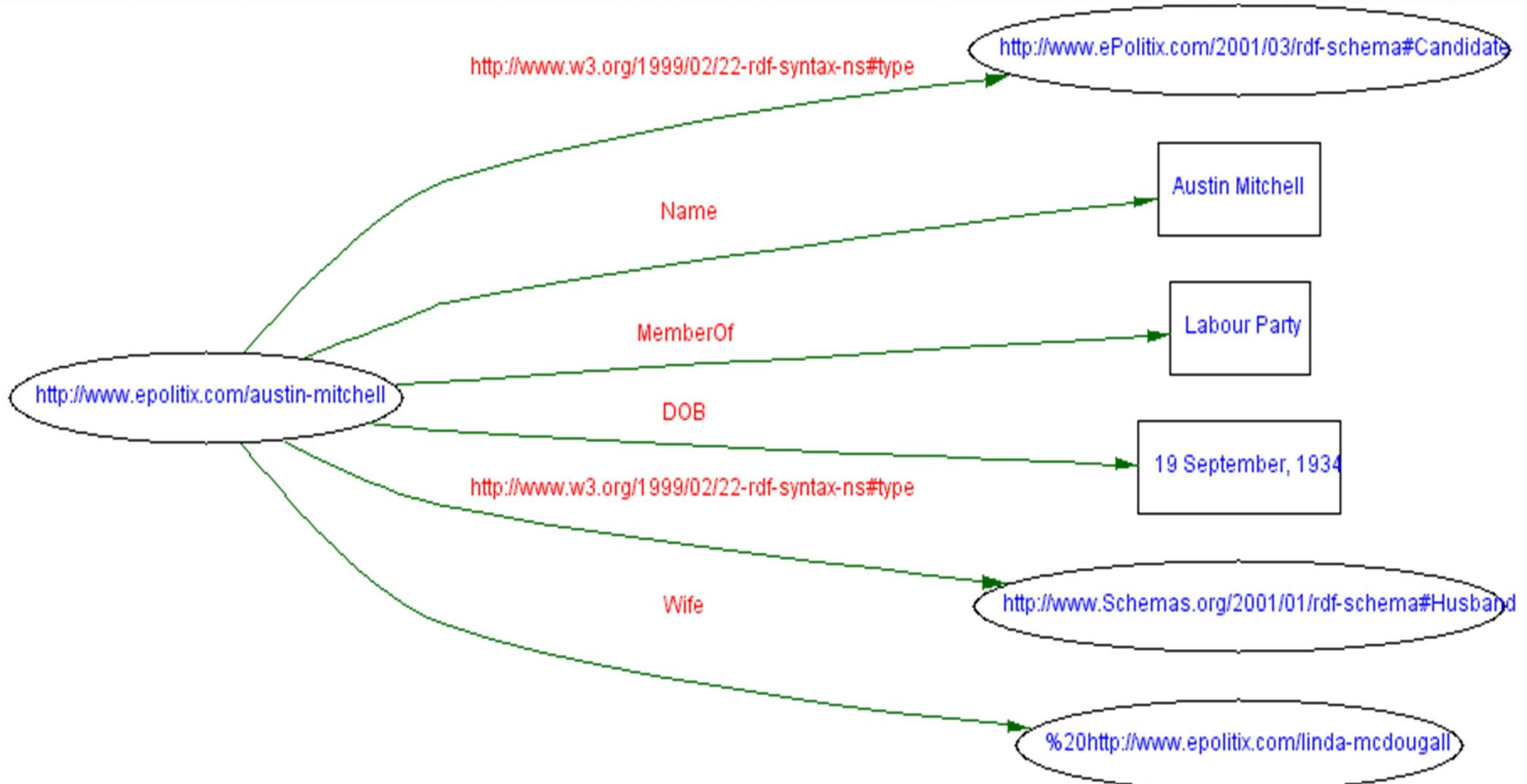
# Többszörös öröklődés

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#" >

<rdfs:Class
 rdf:about=
 "http://www.ePolitix.com/2001/03/rdf-schema#DefendingCandidate">
<rdfs:subClassOf
 rdf:resource=
 "http://www.ePolitix.com/2001/03/rdf-schema#Candidate"/>
<rdfs:subClassOf
 rdf:resource=
 "http://www.ePolitix.com/2001/03/rdf-schema#ElectedRepresentative" />
</rdfs:Class>
</rdf:RDF>
```



# Több példány



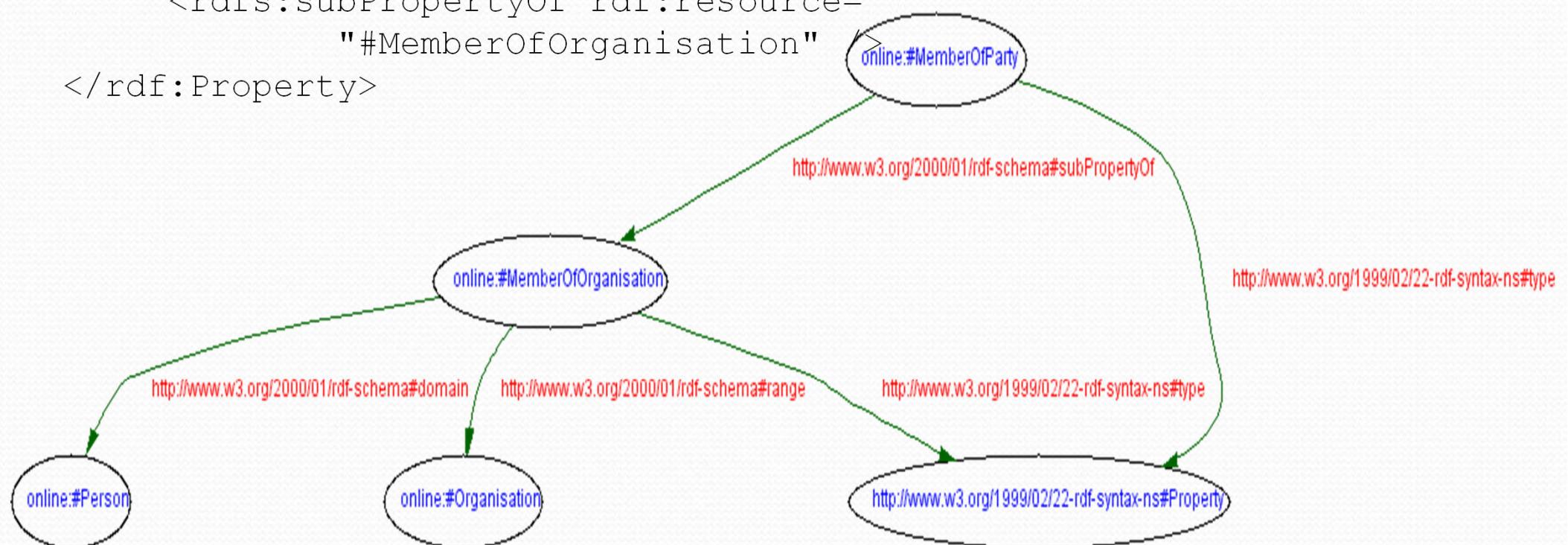
# Több példány (XML szintaxis)

```
<rdf:Description
 rdf:about="http://www.ePolitix.com/austin-mitchell">
 <rdf:type
 resource="http://www.ePolitix.com/2001/03/rdf-schema#Candidate"/>
 <s:Name>Austin Mitchell</s:Name>
 <epx:MemberOf>Labour Party</epx:MemberOf>
 <s:DOB>19 September, 1934</s:DOB>
 <rdf:type
 resource="http://www.Schemas.org/2001/01/rdf-schema#Husband" />
 <s:Wife rdf:resource=" http://www.ePolitix.com/linda-mcdougall" />
</rdf:Description>
```

# Rész tulajdonságok (Subproperties)

```
<rdf:Property rdf:ID="MemberOfOrganisation">
 <rdfs:domain rdf:resource="#Person" />
 <rdfs:range rdf:resource="#Organisation" />
</rdf:Property>

<rdf:Property rdf:ID="MemberOfParty">
 <rdfs:subPropertyOf rdf:resource=
 "#MemberOfOrganisation" />
</rdf:Property>
```

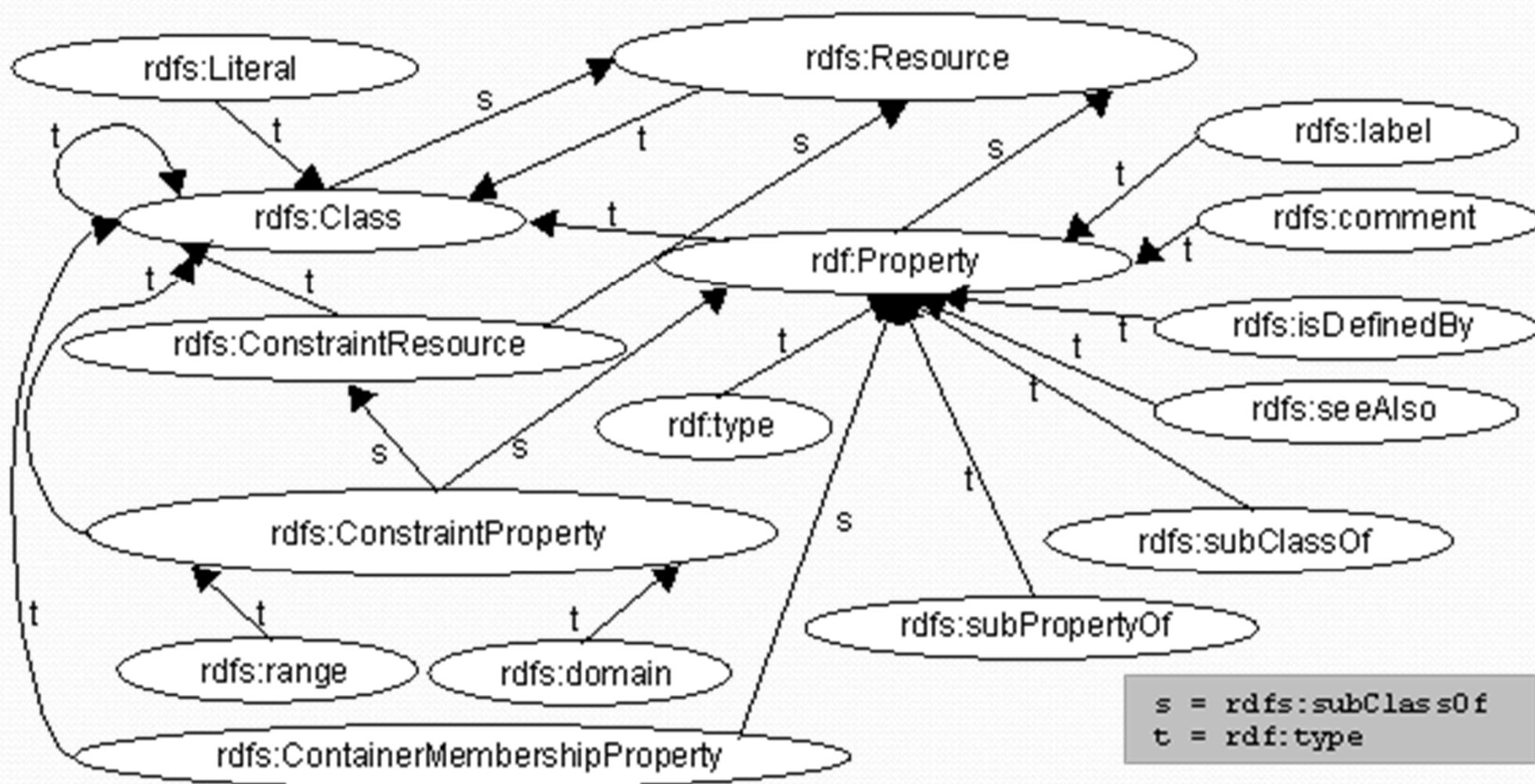


# További RDFS elemek

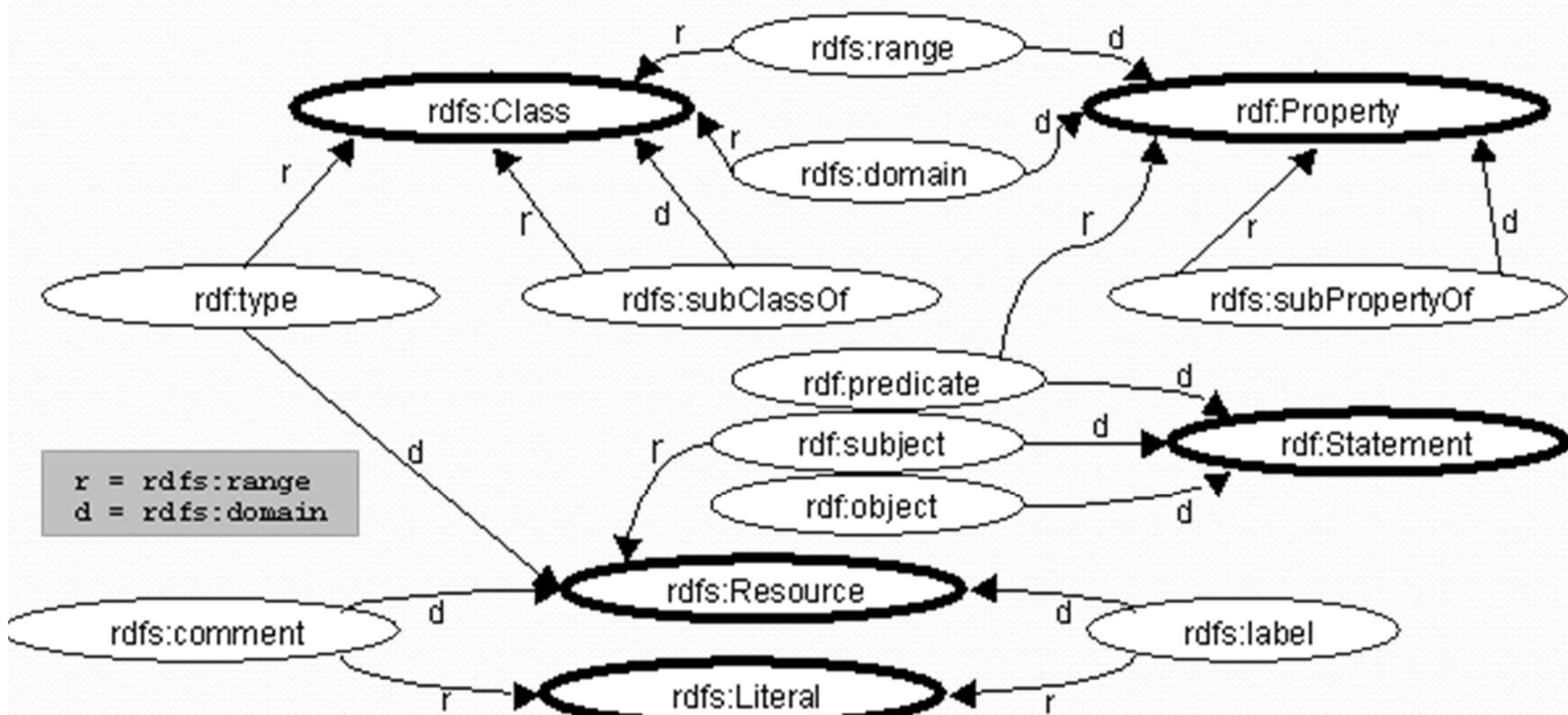
- rdf:Resource
- rdfs:label
- rdfs:comment
- rdfs:seeAlso
- rdfs:isDefinedBy
- rdf:Statement
- rdf:subject
- rdf:predicate
- rdf:object
- rdfs:container
- rdf:bag
- rdf:seq
- rdf:alt

Szabvány:  
<http://www.w3.org/TR/rdf-schema/>

# RDFS Osztály hierarchia



# RDFS Kényszerek



# Wordnet (célok, motivációk)

## Szótár alapú, pszicho-lingvisztikai alapok

- Egy jelentésalapú feldolgozása a lexikonoknak.
  - Fogalom alapú keresés egy lexikális adatbázison
- A fogalmakat szemantikus hálóba rendezzük
  - A lexikális információkat a szavak jelentése szerint rendezzük és nem a szavak formája szerint
- A Wordnet egy tezaurusz

| Elements       | Refinements            | Encodings | Types           |
|----------------|------------------------|-----------|-----------------|
| 1. Identifier  | Abstract               | Box       | Collection      |
| 2. Title       | Access rights          | DCMType   | Dataset         |
| 3. Creator     | Alternative            | DDC       | Event           |
| 4. Contributor | <i>Audience</i>        | IMT       | Image           |
| 5. Publisher   | Available              | ISO3166   | Interactive     |
| 6. Subject     | Bibliographic citation | ISO639-2  | Resource        |
| 7. Description | Conforms to            | LCC       | Moving Image    |
| 8. Coverage    | Created                | LCSH      | Physical Object |
| 9. Format      | Date accepted          | MESH      | Service         |
| 10. Type       | Date copyrighted       | Period    | Software        |
| 11. Date       | Date submitted         | Point     | Sound           |
| 12. Relation   | Education level        | RFC1766   | Still Image     |
| 13. Source     | Extent                 | RFC3066   | Text            |
| 14. Rights     | Has format             | TGN       |                 |
| 15. Language   | Has part               | UDC       |                 |
|                | Has version            | URI       |                 |
|                | Is format of           | W3CTDF    |                 |
|                | Is part of             |           |                 |

# Creator

- “An entity primarily responsible for making the content of the resource”
- In other words – Author, Photographer, Illustrator, ...
  - Potential refinements by creative role
  - Rarely justified
- Creators can be persons or organizations
- Key Point - Dealing with names is a big issue in data quality:
  - Ron Daniel
  - Ron Daniel, Jr.
  - Ron Daniel Jr.
  - R.E. Daniel
  - Ronald Daniel
  - Ronald Ellison Daniel, Jr.
  - Daniel, R.
- Name fields may contain other information
  - <dc:creator>Case, W. R. (NASA Goddard Space Flight Center, Greenbelt, MD, United States)</dc:creator>

## Refinements

**None**

## Encodings

**None**

# Example – Name mismatches

- One of these things is not like the other:
  - Ron Daniel, Jr. and Carl Lagoze; “Distributed Active Relationships in the Warwick Framework”
  - Hojung Cha and Ron Daniel; “Simulated Behavior of Large Scale SCI Rings and Tori”
  - ✓ Ron Daniel; “High Performance Haptic and Teleoperative Interfaces”
- Differences may not matter
- If they do
  - This error cannot be reliably detected automatically
  - Authority files and an **error-correction** procedure are needed

# Contributor

- “An entity responsible for making contributions to the content of the resource.”
- In practice – rarely used. Difficult to distinguish from Creator.

## Refinements

**None**

## Encodings

**None**

# Publisher

- “An entity responsible for making the resource available”.
- Problems:
  - All the name-handling stuff of Creator.
  - Hierarchy of publishers (Bureau, Agency, Department, ...)

**Refinements**

**None**

**Encodings**

**None**

# Title

- “A name given to the resource”.
- Issues:
  - Hierarchical Titles
    - e.g. Conceptual Structures: Information Processing in Mind and Machine (The Systems Programming Series)
  - Untitled Works

**Refinements**

***Alternative***

**Encodings**

***None***

# Date

- “A date associated with an event in the life cycle of the resource”
- Woefully underspecified.
- Typically the publication or last modification date.
- Best practice: YYYY-MM-DD

## Refinements

*Created*

*Valid*

*Available*

*Issued*

*Modified*

*Date Accepted*

*Date Copyrighted*

*Date Submitted*

## Encodings

*DCMI Period*

*W3C DTF (Profile of ISO 8601)*

# Identifier

- “An unambiguous reference to the resource within a given context”
- Best Practice: URL
- Future Best Practice: URI?
- Problems
  - Metaphysics
  - Personalized URLs
  - Multiple identifiers for same content
  - Non-standard resolution mechanisms for URIs

**Refinements**

***Bibliographic Citation***

**Encodings**

***URI***

# Subject

- The topic of the content of the resource.
- Best practice: Use pre-defined subject schemes, not user-selected keywords.
- Factor “Subject” into separate *facets*.
  - People, places, organizations, events, objects, services
  - Industry sectors
  - Content types, audiences, functions
  - Topic
- Some of the facets are already defined in DC (Coverage, Type) or DCTERMS (Audience)

## Refinements

**None**

## Encodings

**DDC**

**LCC**

**LCSH**

**MESH**

**UDC**

# Coverage

- “The extent or scope of the content of the resource”.
- In other words – places and times as topics.
- Key Point – Locations important in SOME environments, irrelevant in others.

## Refinements

*Spatial  
Temporal*

## Encodings

*Box (for Spatial)  
ISO3166 (for Spatial)  
Point (for Spatial)  
TGN (for Spatial)  
W3CTDF (for Temporal)*

# Description

- “An account of the content of the resource”.
- In other words – an abstract or summary
- Key Point – What’s the cost/benefit tradeoff for creating descriptions?
  - Quality of auto-generated descriptions is low
  - For search results, hit highlighting is probably better

## Refinements

*Abstract  
Table of Contents*

## Encodings

**None**

# Type

- “The nature or genre of the content of the resource”
- Best Current Practice: Create a custom list of content types, use that list for the values.
  - Try to avoid “image”, “audio”, and other format names in the list of content types, they can be derived from “Format”.
  - No broadly-acceptable list yet found.

## Refinements

**None**

## Encodings

***DCMI Type***

# Format

- The physical or digital manifestation of the resource.
- In other words – the file format
- Best practice: Internet Media Types
- Outliers: File sizes, dimensions of physical objects

## Refinements

*Extent  
Medium*

## Encodings

*IMT*

# Language

- “A language of the intellectual content of the resource”.
- Best Practice: ISO 639, RFC 3066
- Dialect codes: Advanced practice

## Refinements

**None**

## Encodings

***ISO639-2***  
***RFC1766***  
***RFC3066***

# Relation

- “A reference to a related resource”
- Very weak meaning – not even as strong as “See also”.
- Best practice: Use a refinement element and URLs.

## Refinements

***Is Version Of***  
***Has Version***  
***Is Replaced By***  
***Replaces***  
***Is Required By***  
***Requires***  
***Is Part Of***  
***Has Part***  
***Is Referenced By***  
***References***  
***Is Format Of***  
***Has Format***  
***Conforms To***

## Encodings

***URI***

# Source

- “A reference to a resource from which the present resource is derived”
- Original intent was for derivative works
- Frequently abused to provide bibliographic information for items extracted from a larger work, such as articles from a Journal

**Refinements**

***None***

**Encodings**

***URI***

# Rights

- “Information about rights held in and over the resource”
- Could be a copyright statement, or a list of groups with access rights, or ...

## Refinements

***Access Rights  
License***

## Encodings

***None***

# A Wordnet szemantikus modellje

A „szó” egy asszociációs kapcsolat:

- a szótárba gyűjtött fogalmak, és
- a szó alakja között (szintaktika).

Lexikalis mátrix:

- Szó alakok(oszlopok)
- Szó jelentések (sorok).

# Lexikális mátrix

Szinonimák

| Szó jelentések | Szó alakok |      |      |     |      |
|----------------|------------|------|------|-----|------|
|                | F1         | F2   | F3   | ... | Fn   |
| M1             | E1,1       | E1,2 |      |     |      |
| M2             |            | E2,2 |      |     |      |
| M3             |            |      | E3,3 |     |      |
| ...            |            |      |      | ... |      |
| Mm             |            |      |      |     | Em,n |

Többértelmű szavak

# Reprezentációs módszer

- Konstruktív

- A reprezentáció tartalmazzon elégséges információt a fogalom felépítéséhez

- Differenciális

- A jelentések úgy legyenek fűzérekkel reprezentálva, hogy megkülönböztethetőek legyenek

# Wordnet fogalmak

## Hipotézis:

- A *szinonima halmaz* egy megfelelő megközelítés egy fogalom definiálására.

## Differenciális megközelítés

- A szavak jelentését reprezentálhatjuk egy szó-listával:  
**synset**.

# Angol wordnet tartalma

Tartalom: 95600 szóalak

- 51.500 egyszerű szó
- 44.100 kollokáció

70100 szó jelentés alak

Wordnet relációk

- Lexikális relációk (szóalakok között)
  - szinonimák
  - antonímák
- Szemantikus relációk (szó jelentések között)
  - Hiponima/Hiperonímia (alárendelt/fölérendelt)
  - Meronímia/Holonímia (része/magában foglalja)
  - Vonzat

# Szinonima

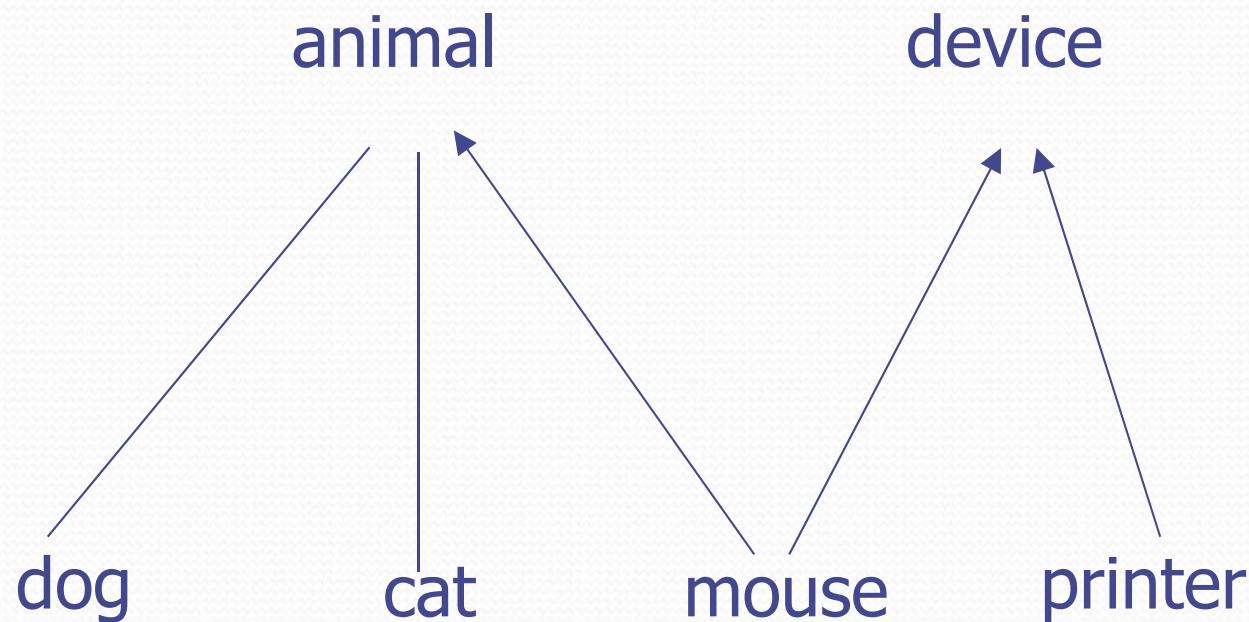
Két szó szinonima, azonos az értelmük, ha a következő kapcsolatok fennálnak:

- minden szemantikus tulajdonságuk értéke megegyezik
- Ugyanannak a fogalomnak a megjelenései
- Kielégítik a Leibniz féle helyettesítési szabályt:
  - Ha felcseréljük a szinonimákat egy mondatban, akkor a mondat igazságtartalma nem változik

*A Synset nem magyarázza el a fogalom jelentését, de megjeleníti, igazolja a fogalom létezését*

# Hiponima

A hiponima egy olyan szókapcsolat, ahol a gyűjtő szó tartalmazza a kapcsolt szavak jelentését (alárendelés).



# Meronima/Holonima

„Része” kapcsolat a jelentésben.

- Tranzitív és aszimmetrikus
- Egy tartalmazó fogalomhoz sok tartalmazott kapcsolódhat

# Példa a Wordnet gazdagságára:

Rész-egész kapcsolat típusai a wordnet-ben:

- Component-object (branch/tree)
- Member-collection (tree/forest)
- Portion-mass (slice/cake)
- Stuff-object (aluminium/airplane)
- Feature-activity (paying/shopping)
- Place-area (Lausanne/Vaud)
- Phase-process (adolescence/growing up).

# Szó kategóriák

## Főnevek

- Hierarchiába szervezve – több (pl. hiponimák vagy meronimák szerint)

## Igék

- Vonzat kapcsolatokon keresztül rendezve

## Melléknevek

- Relációk (pl. ellentét) mentén rendezve

# Kiindulópontok

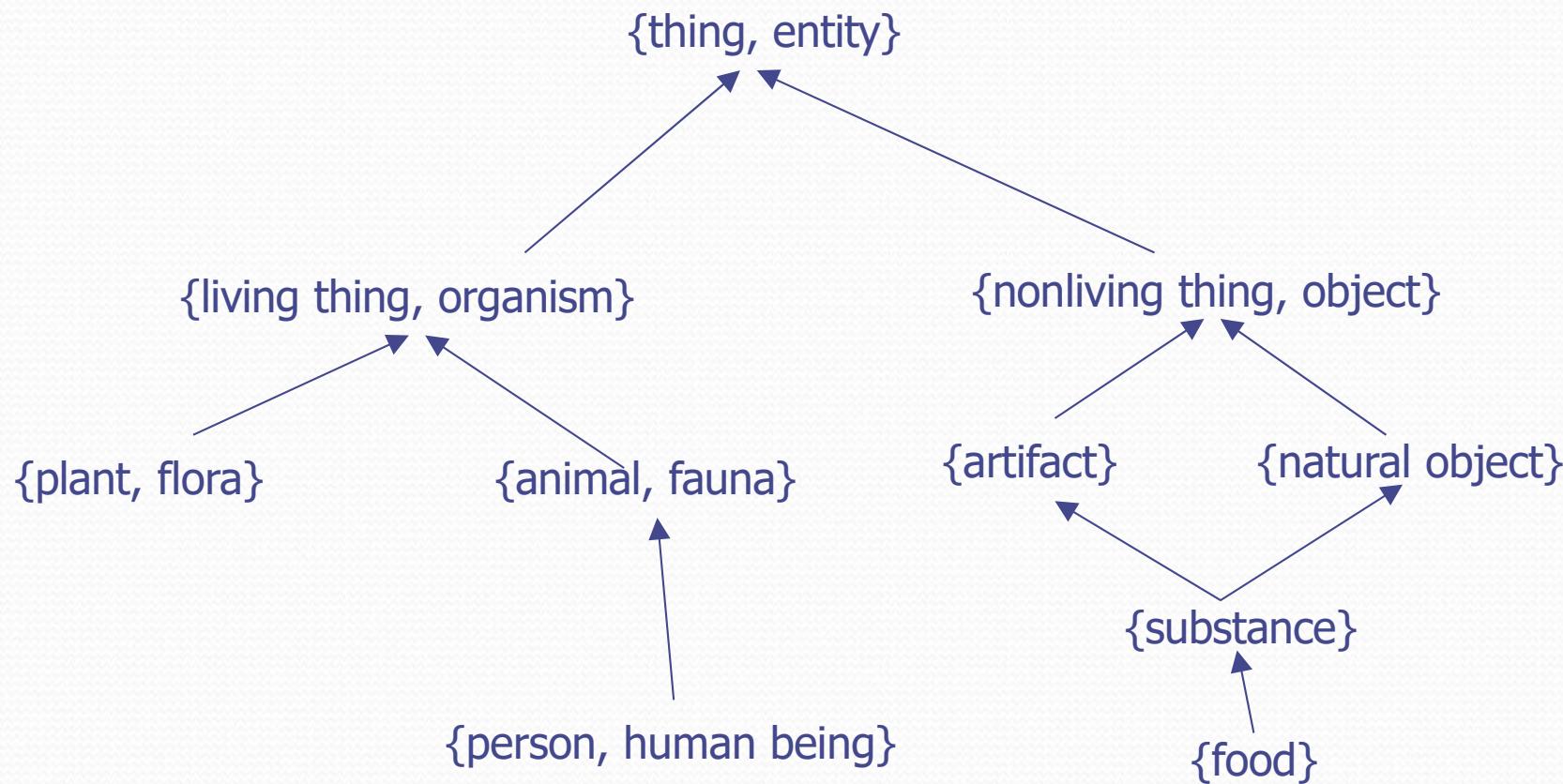
## 25 egyedi hierarchia

- Nem kölcsönösen kizáró kategóriák
- Keresztkapcsolatok megengedettek

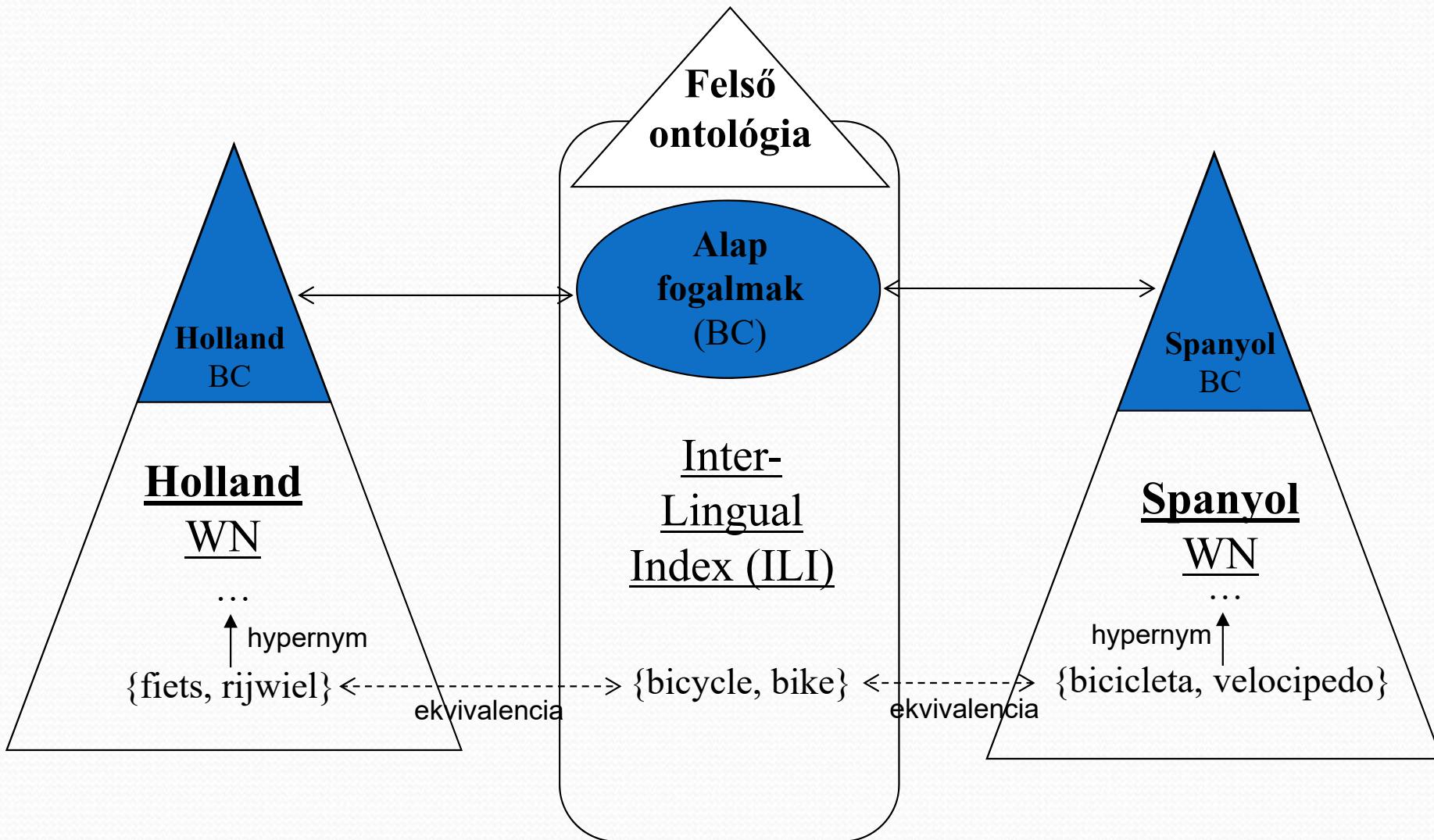
## Fogalmi, lexikai területek:

|                       |                       |                    |
|-----------------------|-----------------------|--------------------|
| {act, activity}       | {food}                | {possession}       |
| {animal, fauna}       | {group, grouping}     | {process}          |
| {artifact}            | {location}            | {quantity, amount} |
| {attribute}           | {motivation, motive}  | {relation}         |
| {body}                | {natural, object}     | {shape}            |
| {cognition,knowledge} | {natural phenomenon}  | {state}            |
| {communication}       | {person, human being} | {substance}        |
| {event, happening}    | {plant, flora}        | {time}             |
| {feeling,emotion}     |                       |                    |

# Alap kategóriarendszer a Wordnet-ben

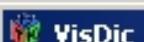


# EuroWordNet: többnyelvű WN



# Magyar WN ontológia (HuWN)

- **BalkaNet** projekt erőforrások használata
  - „Mag” rész: BN Concept Set (8 516 synset 13 nyelv alapján)
  - BN Interlingual Index (PWN 2.0 + SUMO hierarchia)
  - VisDic editor
- **Kiterjesztéses modell** (+ más)
  - Angol synsetek fordítása, relációk átvétele
  - Alapos kézi ellenőrzés és javítás
- **Fél-automatikus** módszerek
  - Korábban kifejlesztett fordító heurisztikák
  - 70% körüli pontosság (főnevek)
- **Meglévő erőforrások integrációja**
  - Magyar Értelmező Kéziszótár meghatározásai
  - NYTI igei vonzatkeret-adatbázis



Files Dictionary Help

English WordNet 2.0.c

+[n] dog:1, domestic dog:1, Canis

View Tree RevTree BCS1,2 E

POS: n ID:  
ENG20-02001223-n

BCS: 3

Synonyms: dog:1,  
domestic dog:1, Canis  
familiaris:1

Definition: a member of the  
genus *Canis* (probably  
descended from the  
common wolf) that has  
been domesticated by man  
since prehistoric times;  
occurs in many breeds  
Usage: the dog barked all  
night

Domain: zoology  
SUMO/MILO: + Canine  
-->> [hypernym] \*[n]  
canine:2, canid:1  
-->> [holo\_member] +[n]  
Canis:1, genus Canis:1  
-->> [holo\_member] +[n]  
pack:6  
<<< [hyponym] [n]  
pooch:1, doggie:1,

Hungarian BCS

[n] *Canis familiaris*:1, házikutya:1, kutya:1, eb:

[n] korcs kutya:1, keverék kutya:

[n] őrző-védő kutya:

View Tree RevTree Edit Nouns Verbs Verl

#### Definiton

A ragadozók közé tartozó, ház- és nyáj-

#### Non Lexicalized



#### Usage

A szomszéd kutyája egész éjszaka

[+/-]

#### Note

kutya\_1\_1

[+/-]

#### Part of Speech

n

lagyar Ertelmező Keziszotar (

[fn] kutya\_1\_1

[fn] kutya\_1\_2

[fn] kutya\_1\_3

View XML

POS: fn ID: kutya\_1\_1  
Headword(s): kutya

Definition: A ragadozók köztartozó, ház- és nyáj-

vadászatra stb. haszn. vagy kedvtelésből tartott háziállat.

-->> [hypernym] háziállat

MorphoLogic Tezaurusz (r)

[a] borzasztó, éktelen, csípő

[n] kutya, eb, kutyus, kutyák

[n] senki, senki fia, senki

View XML

POS: n ID: TEZ-06661-n

Synonyms:  
kutya  
eb  
kutyus  
kutyuli

# HuWN: igék

- Problémák
  - Homályos jelentésbeli megkülönböztetések
  - Inkonzisztens angol WN  
Thematikus szerepek, metaforikus jelentések, szelekciós megkötések stb.
- Megoldás
  - „Vegyes” metodológia:  
BCS fordítás + MNSZ vonzatkeret-gyakoriság alapján kiválasztott igék, saját rendezés
  - Specifikus magyar relációk  
Igekötők, -képzők kezelése stb.

# HuWN

- Mag rész kiterjesztése
  - MNSZ és ÉKSz korpuszgyakoriságok alapján
- Ontológia további bővítése
  - Főnevek, melléknevek:
    - Iteratív koncentrikus bővítés PWN alapján
    - ÉKSz-ben feltárt szemantikai relációk alapján
  - Igék:
    - MNSZ vonzatkeret-gyakoriság alapján
    - PWN alapján
  - 2007: kb. 40K synset

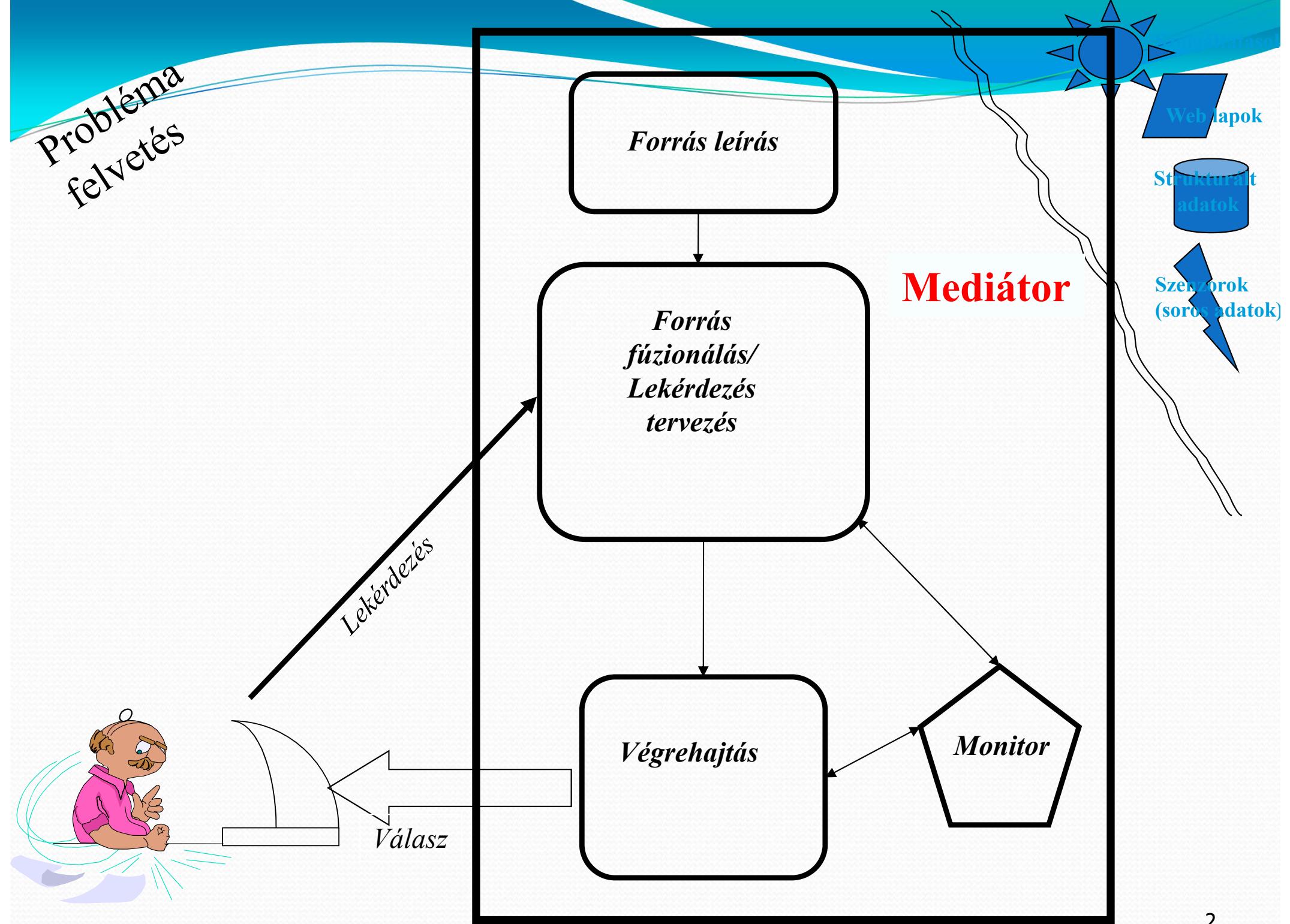
# Integrációs és ellenőrzési technikák

## VIMIAC04, 2021. tavasz

### SZEMANTIKUS WEB

Méréstechnika és Információs Rendszerek Tanszék  
<https://www.mit.bme.hu/oktatas/targyak/vimiac04>

# Probléma felvetés

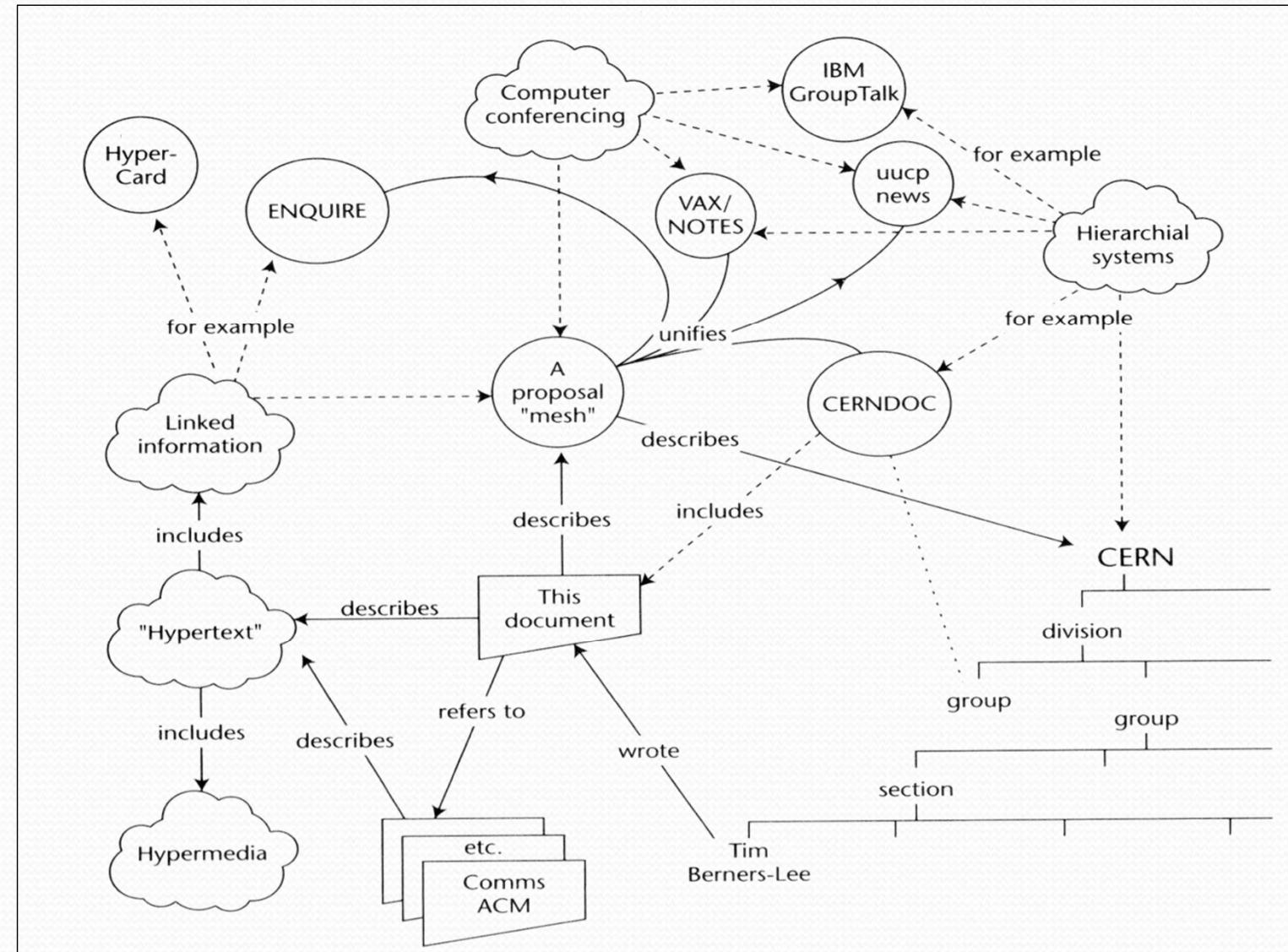


# Szemantikus Web

- A szemantikus web alkalmas megközelítés, megfelelő nyelvekkel, eszközökkel támogatja az intelligens információs rendszerek fejlesztését az elosztott információs környezetben.
- A SzW alapja a hagyományos web hálózat, így egyáltalán nem nyilvánvaló, hogy alkalmas a feladatra.
- A SzW technológia lehetőséget teremt az ágens alapú intelligens megoldások felhasználására a web területen.

# A Szemantikus Web eredete

- Tim Berners-Lee eredeti 1989-es WWW javaslata a Web-et információ menedzselő funkciókkal ellátott objektumok kapcsolataiként jellemzi.



<http://www.w3.org/History/1989/proposal.html>

# W3C szervezet célkitűzései

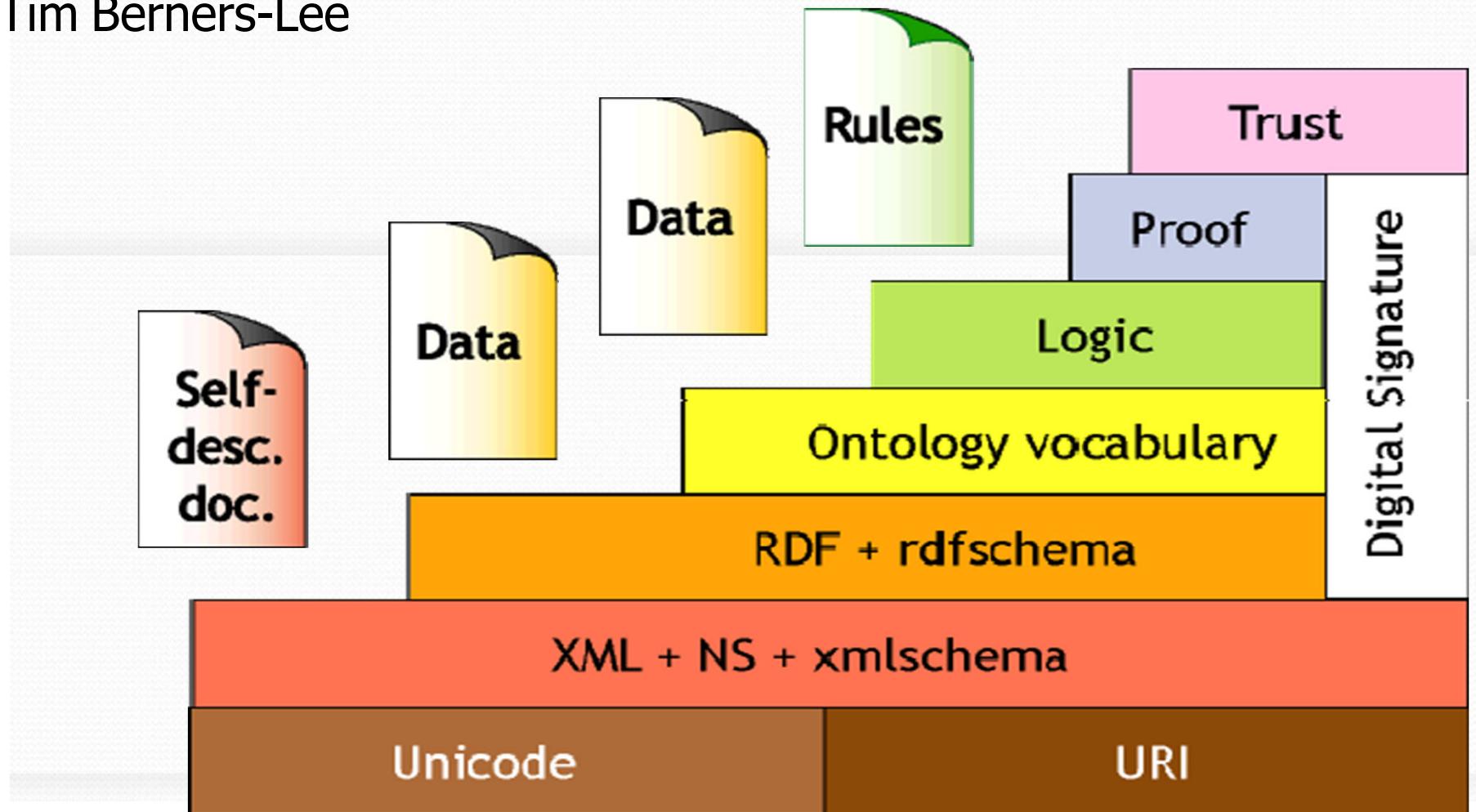
- Megközelítés – számítógépek jobb kihasználtságának biztosítása:

*„A szemantikus web egy kiterjesztése a jelenlegi web-nek, amelyben az információknak jól definiált jelentést adhatunk, lehetővé téve a gépek és felhasználók jobb együttműködését..”* -- Berners-Lee, Hendler and Lassila, The Semantic Web, Scientific American, 2001

- A jelenlegi web tárol dolgokat, míg a szemantikus web képes működtetni dolgokat.

# TBL szemantikus web felépítése

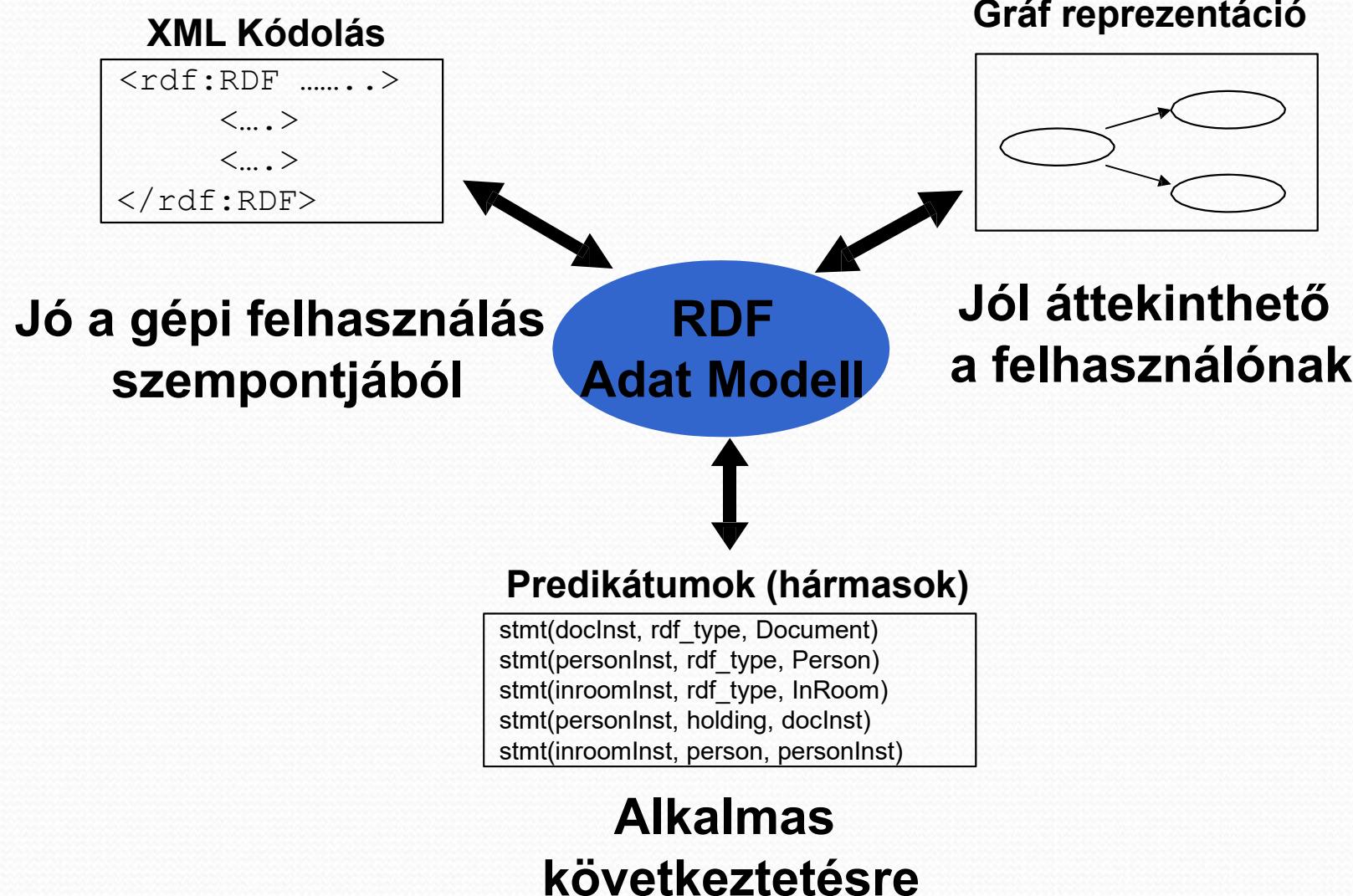
"A szemantikus web  
elérhetővé teszik a tudást,  
mint a web a hipertext-et --  
Tim Berners-Lee



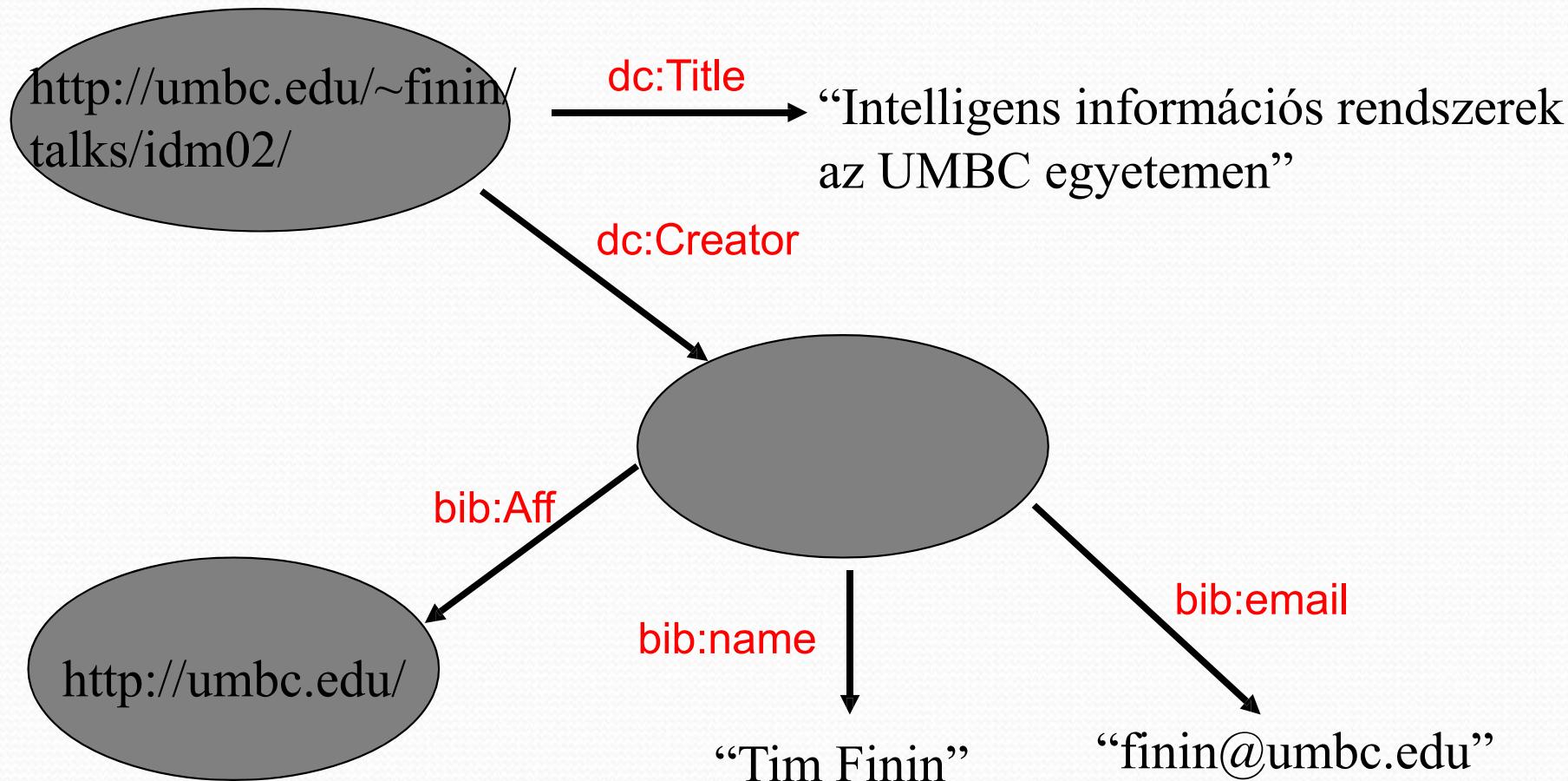
# Dokumentumok

- **RDF Primer**
  - URI: <http://www.w3.org/TR/rdf-primer>
- **OWL Guide**
  - URI: <http://www.w3.org/TR/owl-guide/>
- **RDF Test Cases**
  - URI: <http://www.w3.org/TR/rdf-testcases/>
- **RDF: Concepts and Abstract Syntax**
  - URI: <http://www.w3.org/TR/rdf-concepts/>
- **RDF szemantika**
  - URI: <http://www.w3.org/TR/rdf-mt/>
  - Precíz, gráfokon alapuló szemantika
- **RDF/XML szintaxis**
  - URI: <http://www.w3.org/TR/rdf-syntax-grammar/>
- **RDF Vocabulary Description Language (RDF Schema)**
  - URI: <http://www.w3.org/TR/rdf-schema/>
- **Semantic Web/RDF Interest Group**
  - Vitafórum, alkalmazások
  - URI: <http://www.w3.org/RDF/Interest>
- **RDF Logic**
  - Nyilvános levelezési lista részletesebb szakmai vitákhoz
  - URI: <http://lists.w3.org/Archives/Public/www-rdf-logic/>
- **Annotation and Collaboration**
  - Nyilvános levelezési lista RDF-alapú annotációs rendszerekről
  - URI: <http://lists.w3.org/Archives/Public/www-annotation/>
- **W3C Semantic Web Home page**
  - URI: <http://www.w3.org/2001/sw/>

# RDF - az első SzW nyelv



# Egyszerű RDF példa



# A példa XML szintaxiszáll

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:dc="http://purl.org/dc/elements/1.1/"
 xmlns:bib="http://daml.umbc.edu/ontologies/bib/">
<description about="http://umbc.edu/~finin/talks/idm02/">
 <dc:title>Intelligent Information Systems on the Web and in the
Aether</dc:Title>
 <dc:creator>
 <description>
 <bib:Name>Tim Finin</bib:Name>
 <bib:Email>finin@umbc.edu</bib:Email>
 <bib:Aff resource="http://umbc.edu/">
 </description>
 </dc:Creator>
</description>
</rdf:RDF>
```

# Hármasokat alkalmazó reprezentáció

- RDF kifejezések leírhatóak hármasokkal:
- <alany> <állítmány> <tárgy>

Megengedett szintaxis:  
<URI><URI><URI>  
<URI><URI><string>

# RDF tervezési szempontok

- Egyszerű adatmodell
- Formális szemantika és egyszerű következtetési lehetőségek
- Bővíthető URI
- XML alapú szintaktika (is)
  - XML séma adattípusok
- Bárki megfogalmazhat állításokat az erőforrásokról

# Alapelvek (1/2)

- Külön értelmezhetően definiálva
  - Modell struktúra (RDF gráf)
  - Interpretációs szemantika (vonzatok)
  - Szintaktikák (XML, TN, N3, ...)
- Mindössze két alap adattípus
  - URI/URIref: minden URI-val azonosított
  - Literálisok
    - String vagy más XSD adattípus

# Alapelvek (2/2)

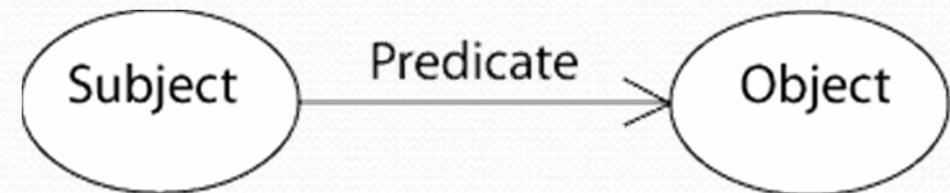
- Integrálható a webes információkkal
  - XML séma adattípusok
  - Referenciák http elérésű információkhoz
- Nyílt világ feltételezés
  - Bárki megfogalmazhat állításokat bármilyen erőforráshoz
  - Nem garantált a teljesség
  - Nem garantált a konzisztencia

# Alapelemek

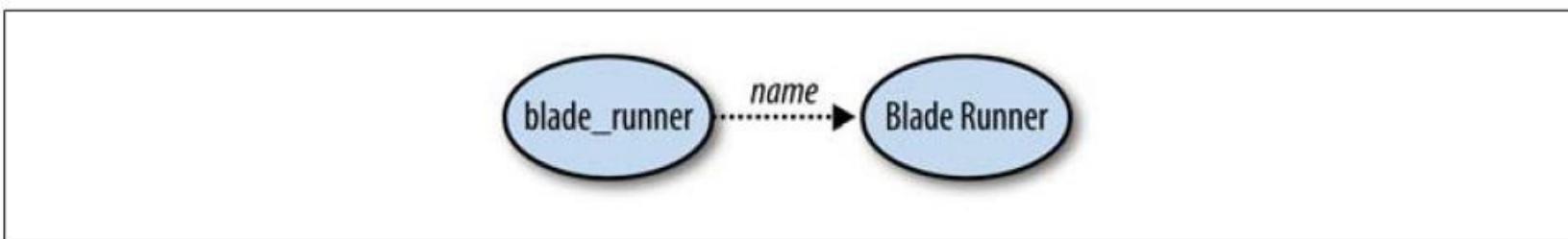
- Gráf adatmodell
- URI alapú szótárak
- Adattípusok
- Literálisok
- XML szerIALIZÁCIós szintaktika
- Egyszerű tények leírása
- Következtetés

# Graph adatmodell

- Hármasok: alany, állítmány, tárgy
- Kifejezések: hármasok gyűjteménye

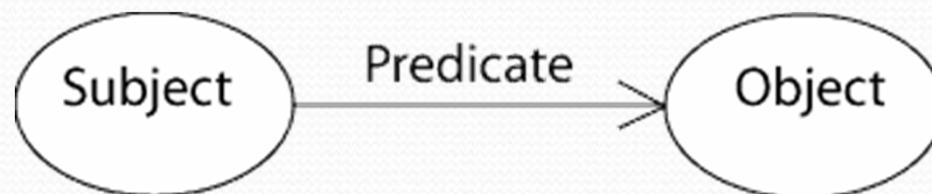


# Példa

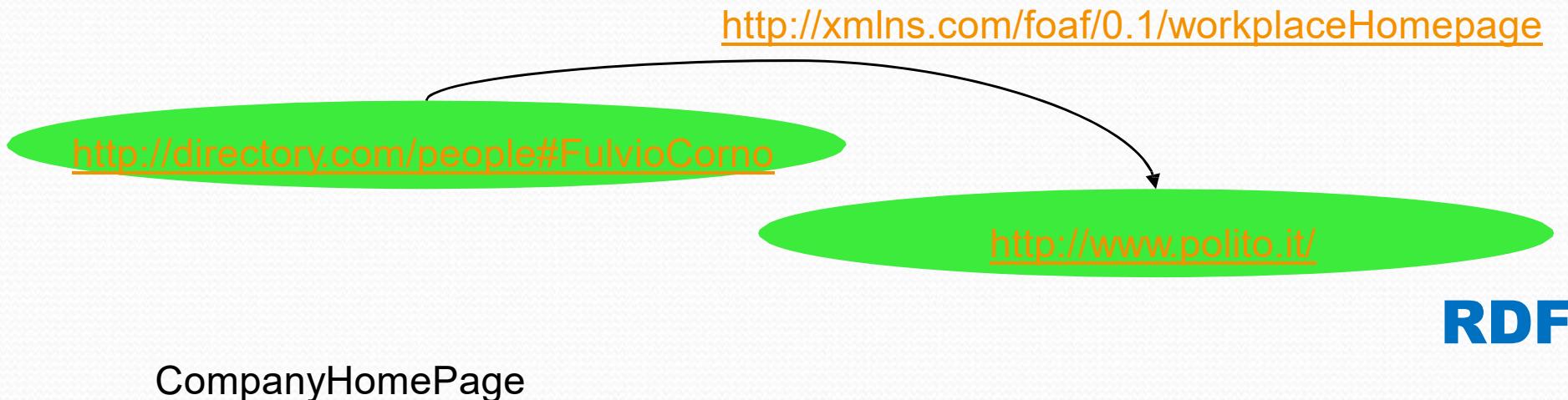


# Terminológia és kényszerek

- Alany és tárgy a csomópontok
- Állítmány és tulajdonság szinonimák
- Különleges meg nem nevezett csomópontok: üres csomópontok
- Alany: URI referencia vagy üres csomópont
- Állítmány: URI referencia
- Tárgy: URI referencia, literális, üres csomópont



# Információ a hármasokban



| PersonID    | Homepage                                                  |
|-------------|-----------------------------------------------------------|
| FulvioCorno | <a href="http://www.polito.it/">http://www.polito.it/</a> |

## Relációs adatbázis

**Első rendű logikai  
predikátum**

```
HasCompanyHomePage(
 'FulvioCorno',
 'http://www.polito.it/');
```

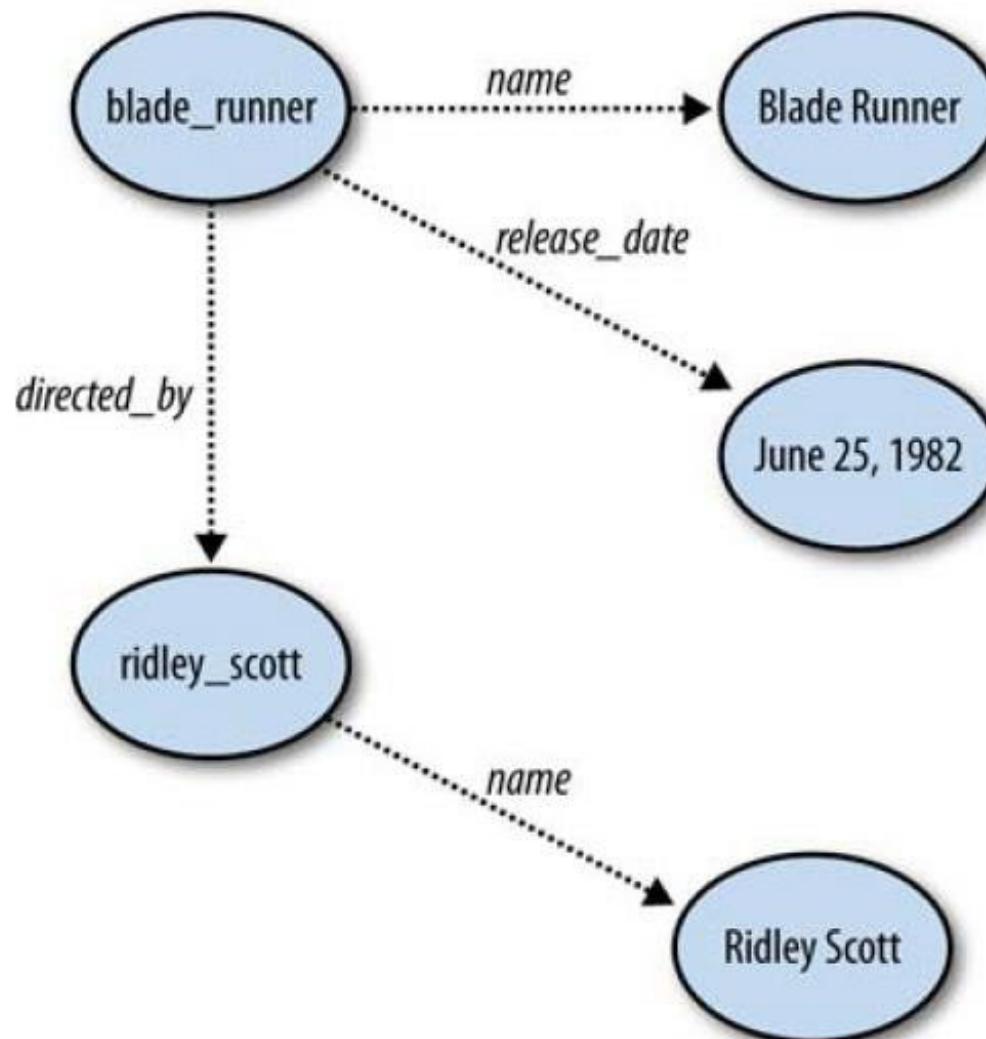
# Hármások vs adatbázisok

| FEATURE                 | RELATIONAL DATABASE           | KNOWLEDGEBASE           |
|-------------------------|-------------------------------|-------------------------|
| Structure               | Schema                        | Ontology statements     |
| Data                    | Rows                          | Instance statements     |
| Administration language | DDL                           | Ontology statements     |
| Query language          | SQL                           | SPARQL                  |
| Relationships           | Foreign keys                  | Multidimensional        |
| Logic                   | External of database/triggers | Formal logic statements |
| Uniqueness              | Key for table                 | URI                     |

# Összehasonlítva...

- Relációs adatbázisban tetszőleges számú oszlopot definiálhatunk
- Elsőrendű logikában a predikátumok tetszőleges elemet (argumentumot) tartalmazhatnak
- Egy RDF hármas csak egy alanyt és egy tárgyat tartalmazhat
  - Komplex kijelentéseket dekomponálni kell

# Példa



# Példa

- Írjuk le RDF állításokkal a következő kijelentést
- "Ilétezik egy személy, akit a <http://www.w3.org/People/EM/contact#me> oldallal azonosíthatunk, akinek neve Eric Miller, email címe [em@w3.org](mailto:em@w3.org), és a titulusa Dr."

# Példa



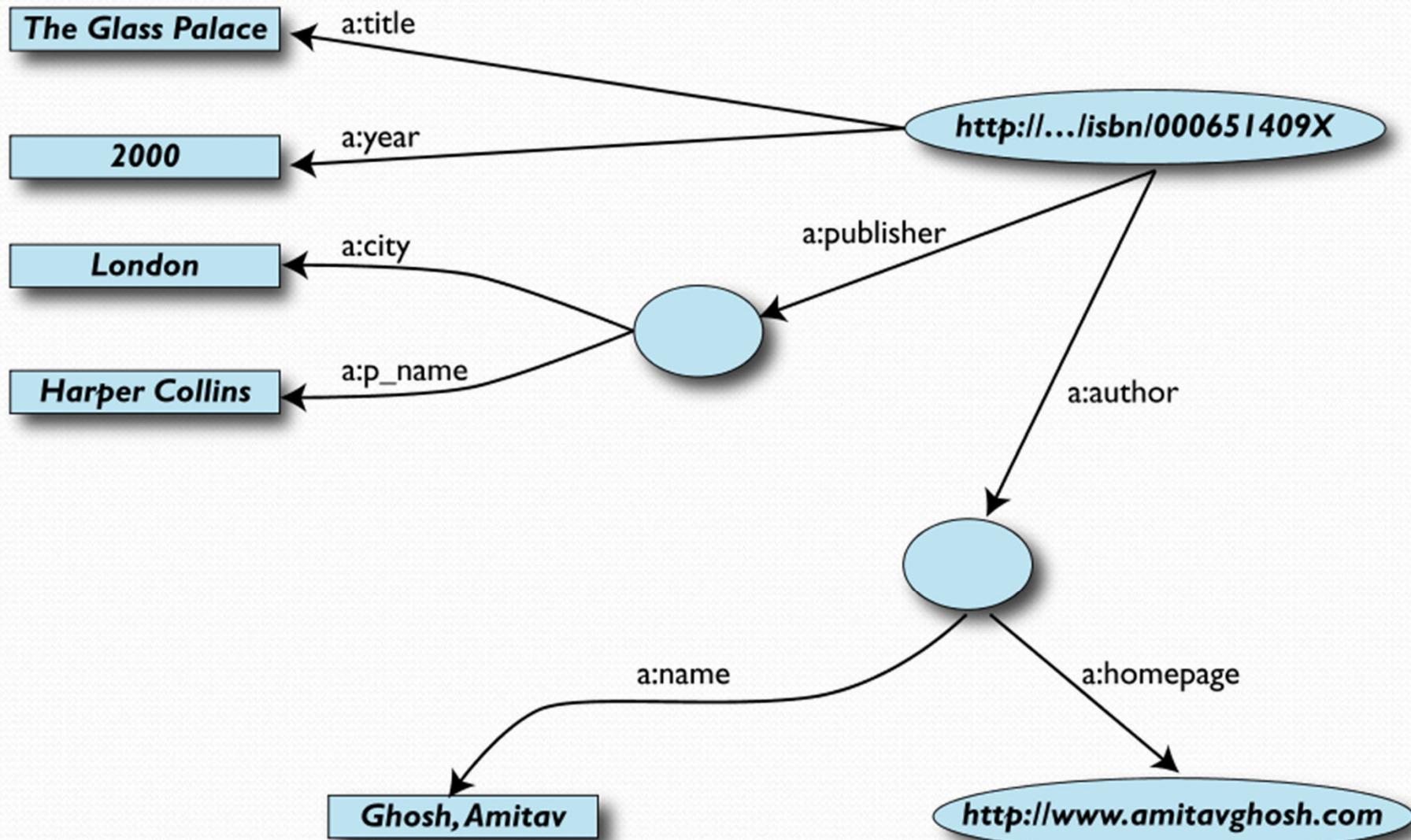
# Példa: könyvesbolt adatai

| ID                | Author | Title            | Publisher | Year |
|-------------------|--------|------------------|-----------|------|
| ISBN0-00-651409-X | id_xyz | The Glass Palace | id_qpr    | 2000 |

| ID     | Name          | Home Page                                                           |
|--------|---------------|---------------------------------------------------------------------|
| id_xyz | Ghosh, Amitav | <a href="http://www.amitavghosh.com">http://www.amitavghosh.com</a> |

| ID     | Publ. Name     | City   |
|--------|----------------|--------|
| id_qpr | Harper Collins | London |

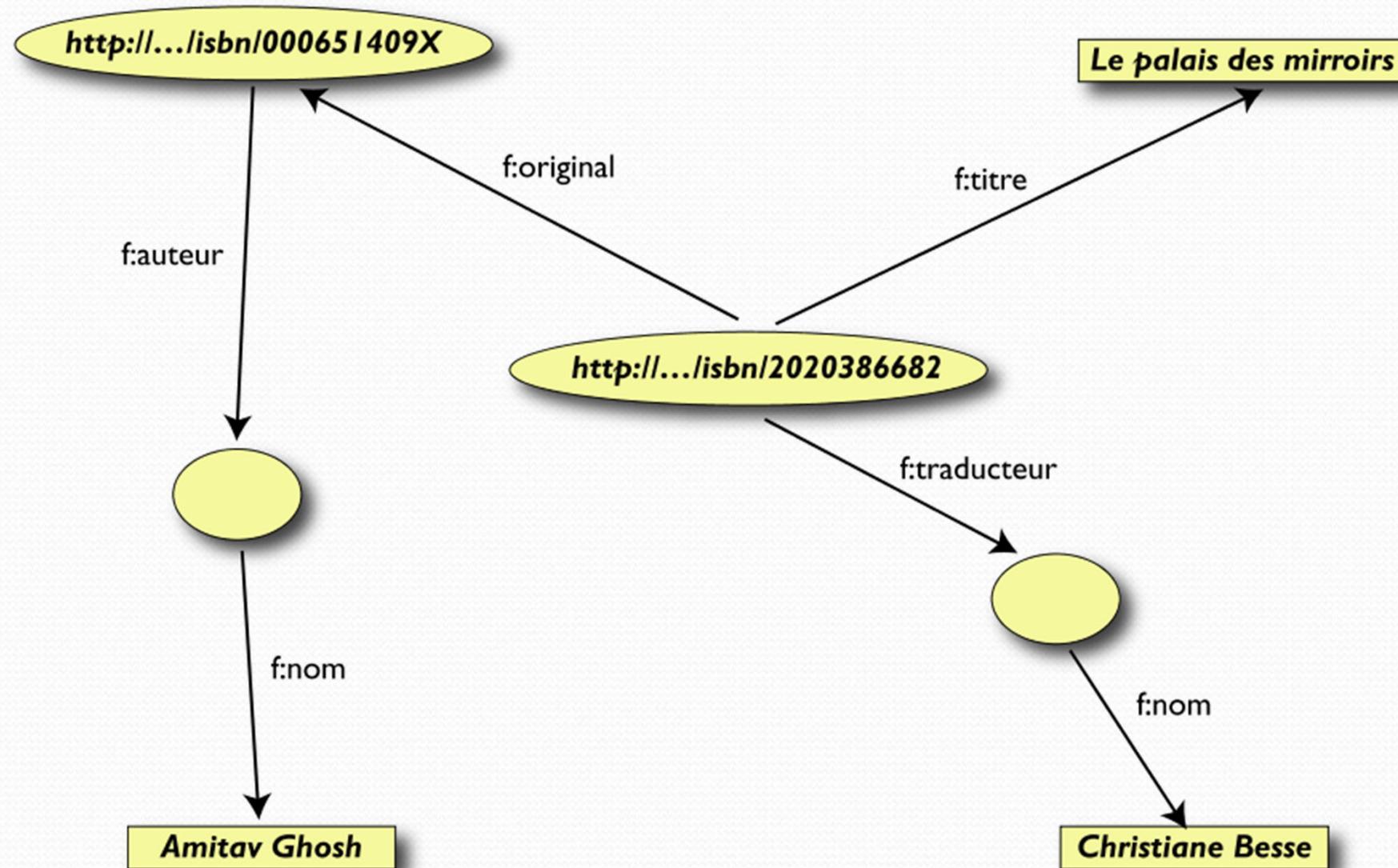
# Adatok exportálása relációkként



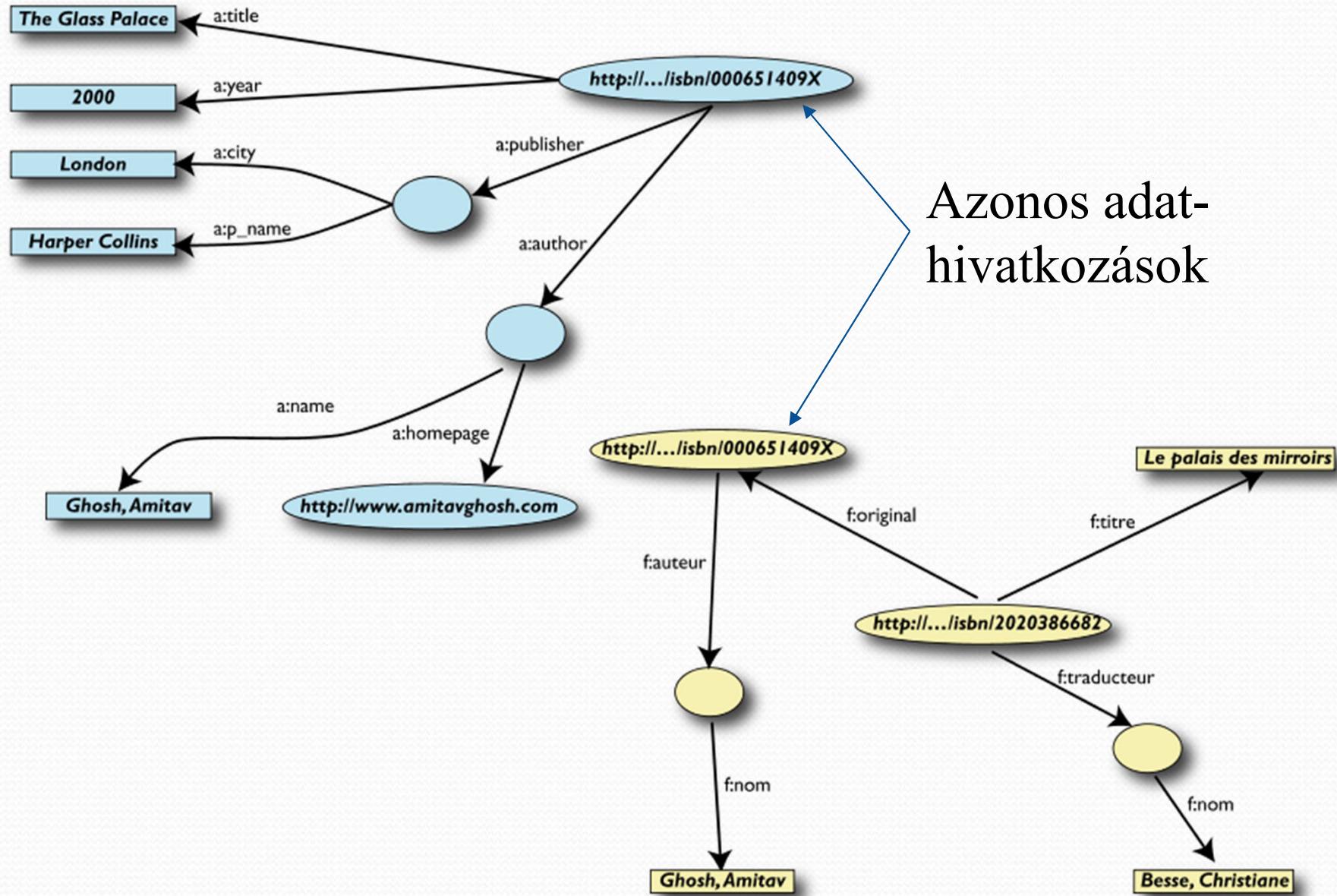
# Egy másik könyvesbolt adatai

|    | A                  | B                           | D                 | E                  |
|----|--------------------|-----------------------------|-------------------|--------------------|
| 1  | <b>ID</b>          | <b>Titre</b>                | <b>Traducteur</b> | <b>Original</b>    |
| 2  | ISBN0 2020386682   | Le Palais<br>des<br>miroirs | A13               | ISBN-0-00-651409-X |
| 3  |                    |                             |                   |                    |
| 6  | <b>ID</b>          | <b>Auteur</b>               |                   |                    |
| 7  | ISBN-0-00-651409-X | A12                         |                   |                    |
| 11 | <b>Nom</b>         |                             |                   |                    |
| 12 | Ghosh, Amitav      |                             |                   |                    |
| 13 | Besse, Christianne |                             |                   |                    |

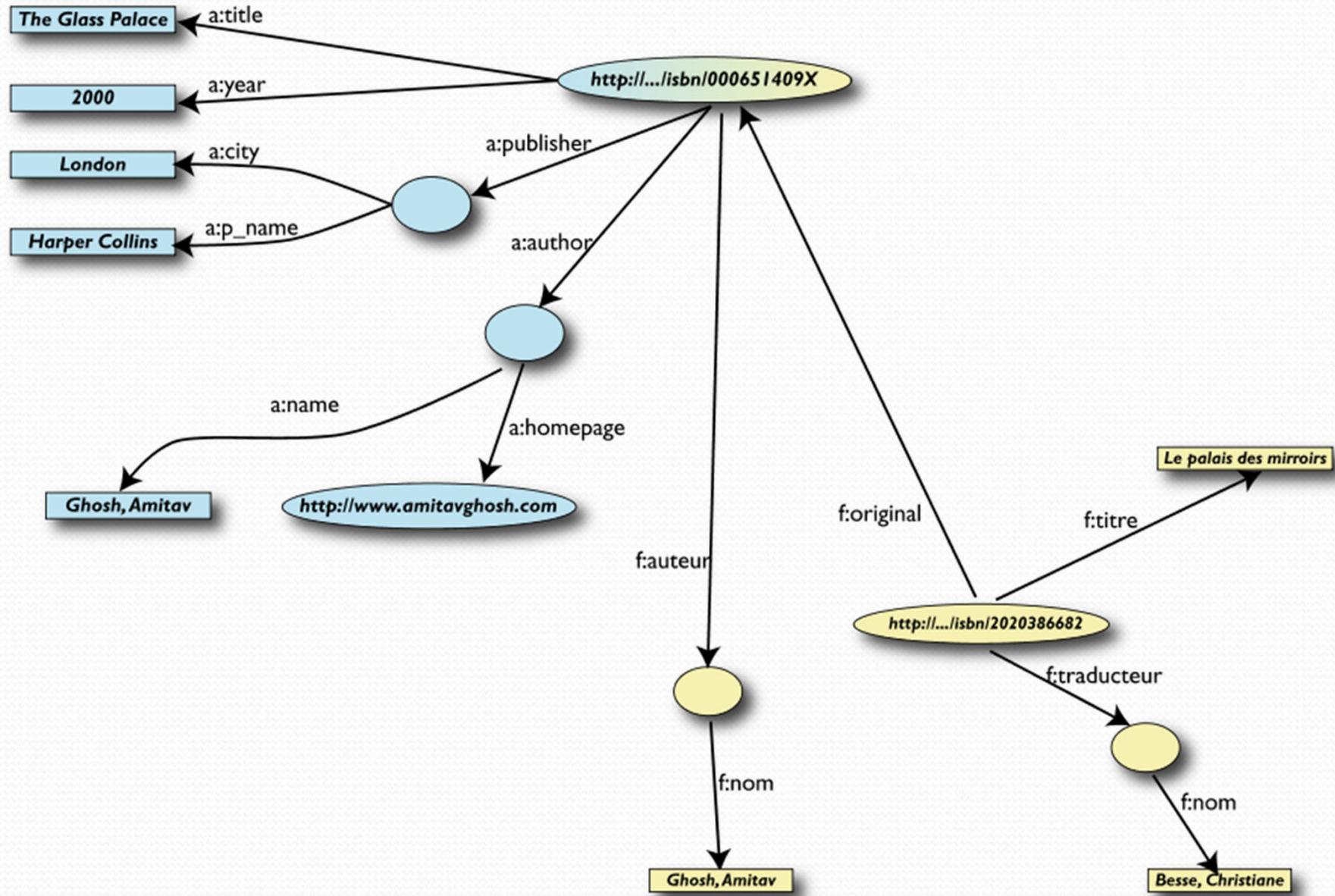
# A második könyvesbolt adatainak exportja



# Kapcsoljuk össze az adatokat

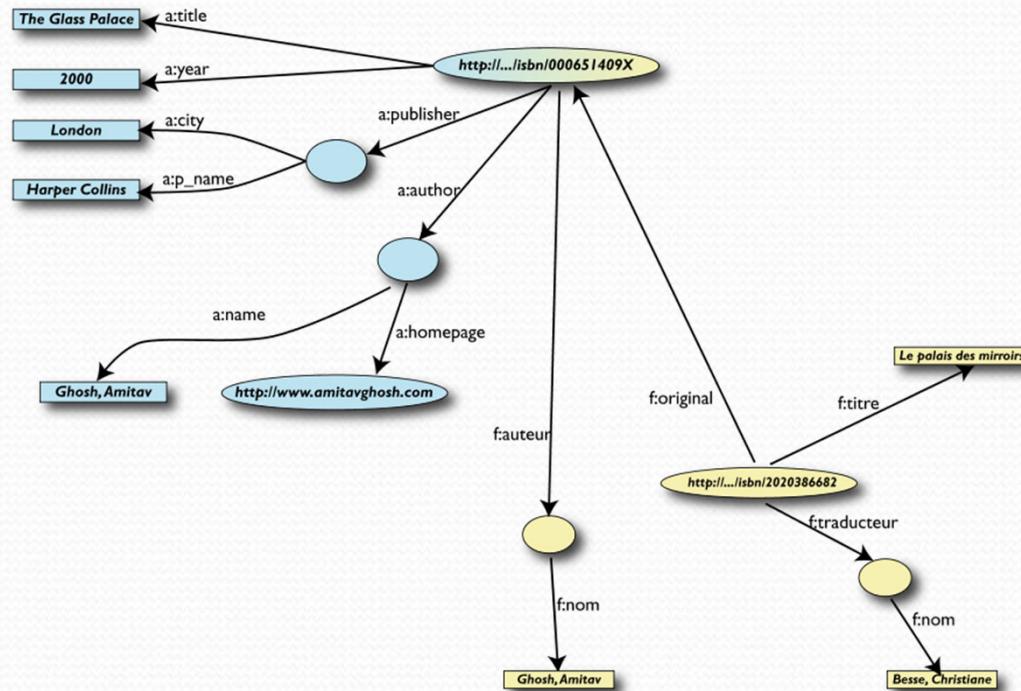


# Amennyiben identikusak az elemek:



# Írunk lekérdezéseket a kapcsolatokon keresztül:

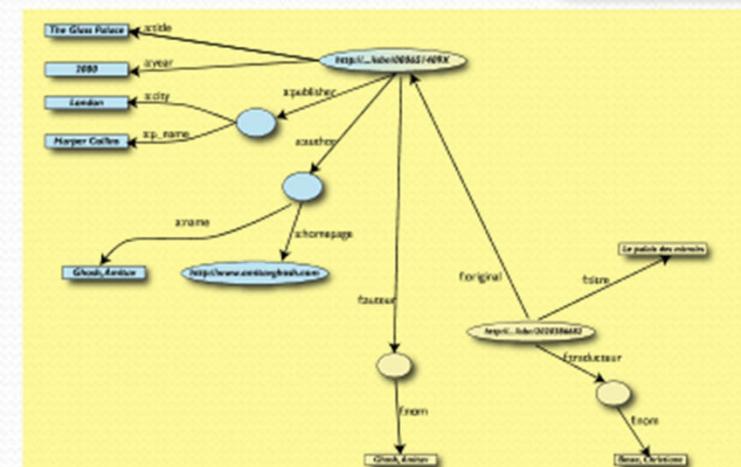
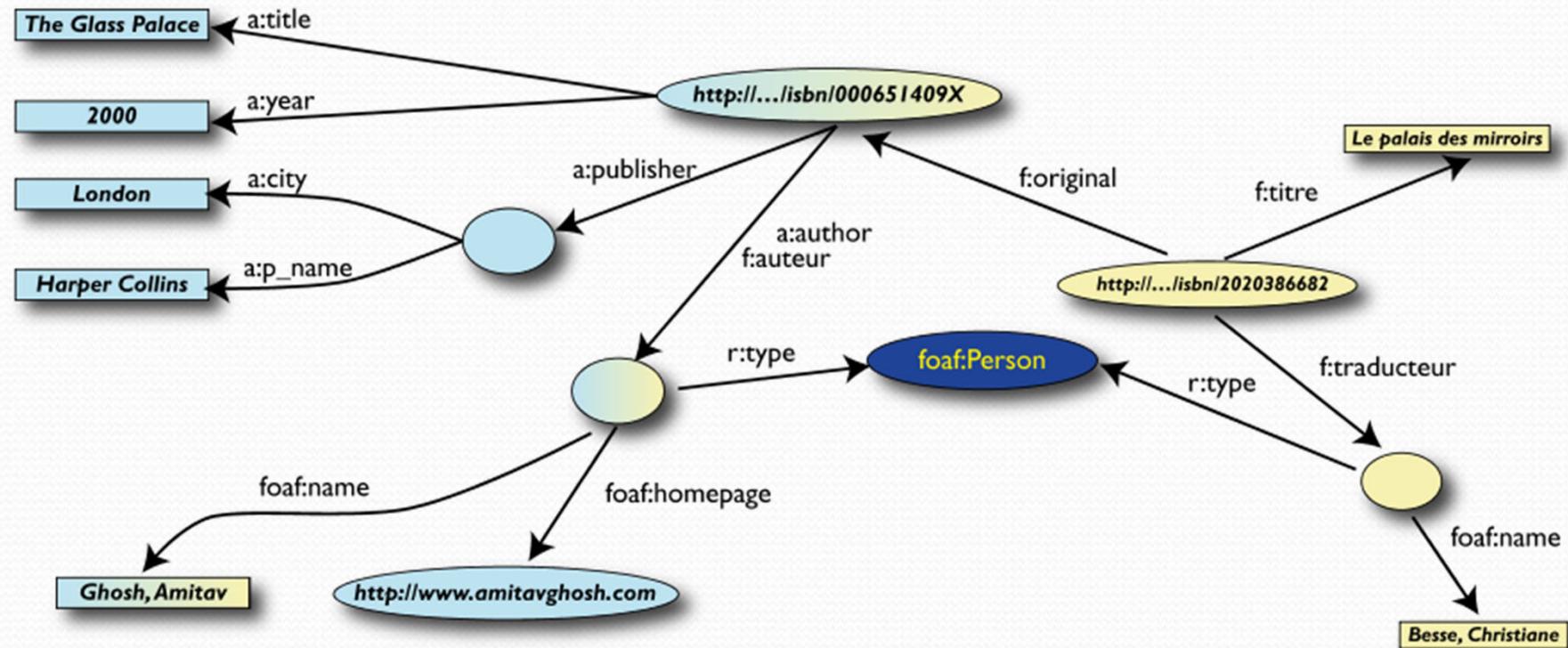
- Az első könyvesbolt adatai kiegészíthető például az eredeti könyvre vonatkozó információkkal



# További kapcsolatok is felfedezhetőek...

- Vélhetően az **a:author** és az **f:auteur** azonos elemre mutat
- Automatikus összekapcsoláshoz: adjunk további információt a leíráshoz
  - **a:author** legyen azonos **f:auteur** erőforrással
  - Mindkettő személyt azonosít
  - Ilyen fogalmakat már a webes közösség definiált:
    - egy “Person” elem azonosítható a nevével és a honlapjával
    - Használjuk ezt kategóriaként

# Adatháló kiegészíthető, lekérdezhető így:



# RDF áttekintés

## A formális modell alapelemei:

Két alaphalmaz: erőforrások (resources) és literálisok (literals)

Az erőforrások egy fontos részhalmaza: Tulajdonságok (properties).

Definiálunk egy hármaskóból álló halmazt: Állítások (statements), amelyek formája: {alany, állítmány, tárgy}, ahol az alany egy erőforrás,  
az állítmány egy tulajdonság,  
a tárgy vagy erőforrás vagy literális.

# RDF adatmodell

- Erőforrások (Resources)
  - URI azonosítja
  - Kijelentés vonatkozik rá
- Tulajdonságok (Properties)
  - Erőforráshoz kapcsolt jellemző
  - A tulajdonság is erőforrás
- Literálok (Literals)
  - Karaktersorozatok
- Kijelentések (Statements)
  - Alany (subject); erőforrás
  - Állítmány (predicate); tulajdonság
  - Tárgy (object); erőforrás vagy literál

# Ezek segítségével megfogalmazhatunk leírásokat:

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:s="http://description.org/schema/">
 <rdf:Description about="http://www.w3.org/Home/Lassila">
 <s:Creator>
 <rdf:Description about="http://www.w3.org/staffId/85740">
 <rdf:type resource="http://description.org/schema/Person"/>
 <v:Name>Ora Lassila</v:Name>
 <v:Email>lassila@w3.org</v:Email>
 </rdf:Description>
 </s:Creator>
 </rdf:Description>
</rdf:RDF>
```

# Az RDF szerepe az SZV hierarchiában

- Technológiát és módszert ad ahhoz, hogy dokumentumainkhoz jelentést rendeljünk egy jól olvasható formában
- Jó lehetőség, de a szemantikai információk nem túl hasznosak, amíg strukturáltalan és nem tudjuk konzisztens módon értelmezni.

(XML séma kevés: csak szintaktikáról szól, nem ad lehetőséget a dokumentumon kívüli dolgok leírására)

# Séma hiányában ugyanannak a tartalomnak sokféle reprezentációja lehetséges:

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-
syntax-ns#" />

<rdf:Description
 about="http://www.w3.org/Home/Lassila">
 <Creator>
 <rdf:Description
 about="http://www.w3.org/staffId/85740">
 <rdf:type
 resource="http://desc.org/schema/Person"/>
 <Name>Ora Lassila</Name>
 <Email>lassila@w3.org</Email>
 </rdf:Description>
 </Creator>
 </rdf:Description>
</rdf:RDF>
```

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-
syntax-ns#" />

<rdf:Description
 about="http://www.w3.org/Home/Lassila">
 <author>
 <rdf:Description
 about="http://www.w3.org/staffId/85740">
 <rdf:type
 resource="http://desc.org/schema/Person"/>
 <name>
 <surname>Lassila</surname>
 <given>Ora</given>
 </name>
 <email>lassila@w3.org</email>
 </rdf:Description>
 </author>
 </rdf:Description>
</rdf:RDF>
```

# SPARQL

- Lekérdező nyelv RDF-hez
- SPARQL gráf illesztésen alapuló lekérdező nyelv
- Gráf minták

- példa:

```
<http://example.org/book/book1>
<http://purl.org/dc/elements/1.1/title>
 ?title .
```

- ?title - változó.

# Egy egyszerű SPARQL Query

- **Adat:**

```
<http://example.org/book/book1>
<http://purl.org/dc/elements/1.1/title>
"SPARQL Tutorial".
```

- **Lekérdezés:**

```
SELECT ?title
WHERE { <http://example.org/book/book1>
 <http://purl.org/dc/elements/1.1/title>
 ?title . }
```

- **Eredmény:**

|                   |
|-------------------|
| <b>title</b>      |
| "SPARQL Tutorial" |

# További példa

- **Adat:**

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
```

```
_ :a foaf:name "Johnny Lee Outlaw" .
```

```
_ :a foaf:mbox <mailto:jlow@example.com> .
```

```
_ :b foaf:name "Peter Goodguy" .
```

```
_ :b foaf:mbox <mailto:peter@example.org> .
```

```
_ :c foaf:mbox <mailto:carol@example.org> .
```

- **Lekérdezés:**

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox
WHERE { ?x foaf:name ?name . ?x foaf:mbox ?mbox }
```

- **Eredmény:**

| <b>name</b>         | <b>mbox</b>                |
|---------------------|----------------------------|
| "Peter Goodguy"     | <mailto:peter@example.org> |
| "Johnny Lee Outlaw" | <mailto:jlow@example.com>  |

# Lekérdezésel RDF literálisokkal

- **Példa RDF adatokra**

```
@prefix dt: <http://example.org/datatype#> .
@prefix ns: <http://example.org/ns#> .
@prefix : <http://example.org/ns#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:x ns:p "cat"@en .
:y ns:p "42"^^xsd:integer .
:z ns:p "abc"^^dt:specialDatatype .
```

# RDF Literálisok illesztése

- Lekérdezés 1:

```
SELECT ?v WHERE { ?v ?p "cat" }
```

Lekérdezés 2:

```
SELECT ?v WHERE { ?v ?p "cat"@en }
```

eltérő eredményt ad.

- Csak a második találja meg az előző példában az eredményt:

|                                                                   |
|-------------------------------------------------------------------|
| v                                                                 |
| < <a href="http://example.org/ns#x">http://example.org/ns#x</a> > |

# Üres csomópontok a lekérdezésekben

- **Adat:**

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
```

```
_ :a foaf:name "Alice" .
_ :b foaf:name "Bob" .
```

- **Lekérdezés:**

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?x ?name
WHERE { ?x foaf:name ?name . }
```

- **Eredmény:**

| x    | name    |
|------|---------|
| _ :c | "Alice" |
| _ :d | "Bob"   |

# Üres csomópontok a lekérdezésekben (modell bővítés)

- **Adat:**

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
```

```
_ :a foaf:name "Alice" .
_ :b foaf:name "Bob" .
_ :a foaf:knows _ :b .
_ :b foaf:knows _ :a .
```

- **Lekérdezés:**

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?x ?name1 ?y ?name2
WHERE { ?x foaf:name ?name1 . ?y foaf:name ?name2 .
 ?x foaf:knows ?y }
```

- **Eredmény:**

| ?x   | name1   | ?y   | name2   |
|------|---------|------|---------|
| _ :c | "Alice" | _ :d | "Bob"   |
| _ :d | "Bob"   | _ :c | "Alice" |

# RDF esettanulmányok

- Dublin Core
  - Magas szintű szótár definiálása
  - Elektronikus dokumentumok megtalálása
- Open Directory Project (OPD)
  - Webes katalógus keresők számára
- MusicBrainz
  - Hanganyagok (cd, mp3 ...) metaadatainak lekérésére
- RSS: RDF Site Summary
  - Hírek, események közzététele
- Wordnet
  - Szabadon letölthető szótár
  - Nem csak címszavakat, hanem kapcsolataokat is leír

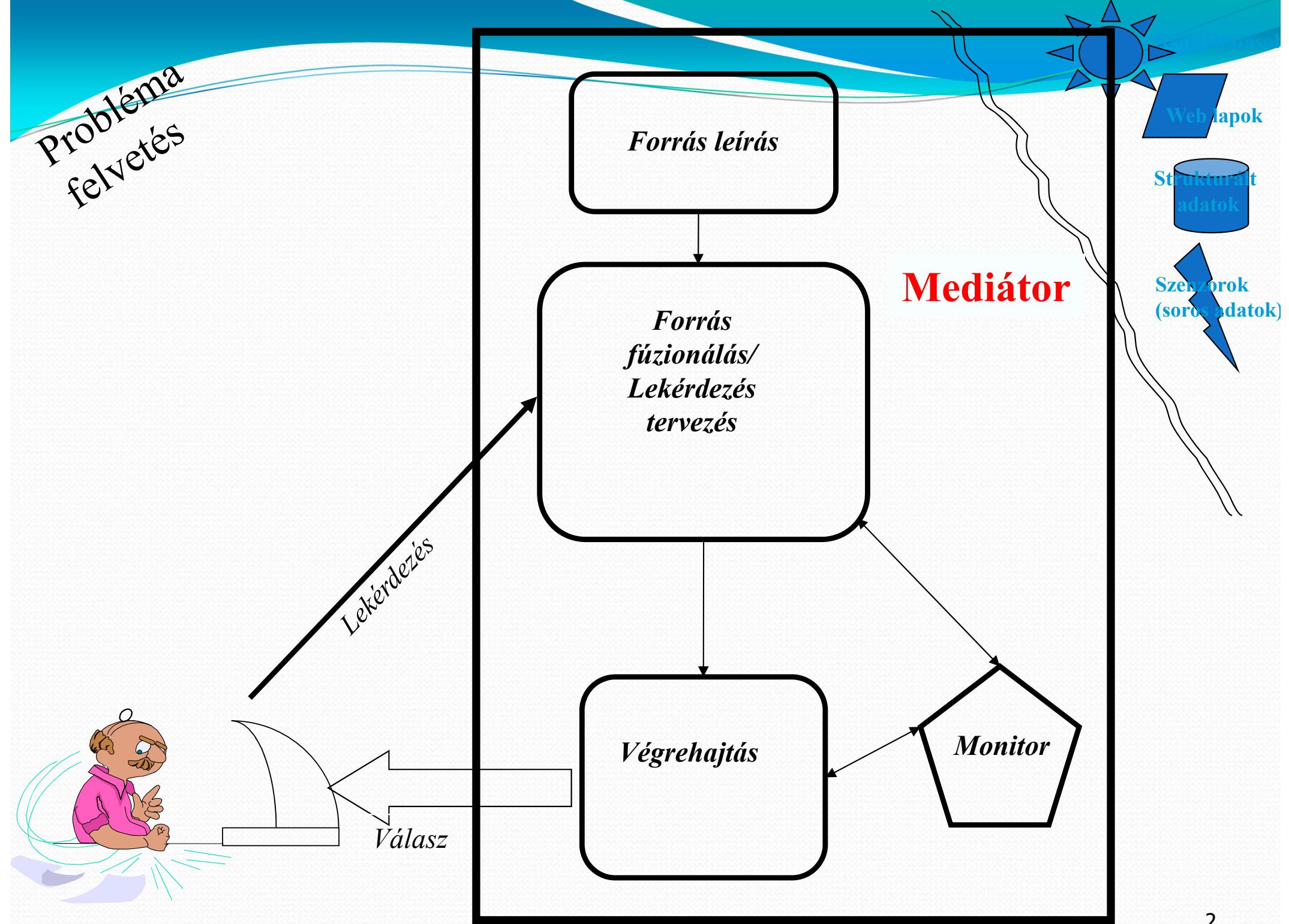
# Integrációs és ellenőrzési technikák

## VIMIAC04, 2021. tavasz

### SZEMANTIKUS WEB

Méréstechnika és Információs Rendszerek Tanszék  
<https://www.mit.bme.hu/oktatas/targyak/vimiac04>

# Probléma felvetés



# A web napjainkban

- Résztvevők dokumentumokat, adatokat publikálnak, URL címeket adva elérhetővé teszik az információkat
  - Kapcsolódás, hivatkozás linkekkel
- Integrációs próbálkozások: mashup oldalak (ad hoc)
  - Webszolgáltatások integrációja  
(eltérő API, logika, struktúra)
  - Adatgyűjtés kereső robotokkal (crawler programokkal)
  - Újra és újra felfedezzük a „kereket”
- Egészítsük ki a webet standard adatelérési módokkal:  
“Adatweb”

# Az adatok webje

- Amire szükségünk van:
  - Publikáljuk adatainkat úgy, hogy felfedezhetőek legyenek a weben:
    - Standardizáljuk az adatok leírását, elérését
    - Dokumentumok eléréhez hasonló, de általános címzés: URI
  - Az URI-kal elérhető forrásokat kapcsoljuk
    - És engedjük a hálózat hatását érvényesülni, ahogy mi is böngészünk a weben...
  - Példák: 2009 Semantic Technology Conference, San Jose, California, USA, June 15, 2009, Ivan Herman, W3C, ivan@w3.org

# Weblapok értelmezése

- A felhasználók megértik, hogy a link egy munkahely honlapjára mutat
- Tudjuk értelmezni, hogy ez egy kutatóhely leírása
- *Ami hiányzik az „adatok webjének” építéséhez:*
  - *jelentéssel kiegészített, értelmezhető linkek*

Tehát bővítsünk:

- Adjunk a linkekhez kiegészítő információt, címkézzük fel őket
- A címkék legyenek géppel értelmezhetőek
  - Kategorizálás
  - Esetleg következtetés

# Adatok hálója

- Az adatok webje:
  - Használunk URI-kat adatok és (nemcsak) dokumentumok publikálására
  - Kapcsoljuk össze az adatokat
  - Jellemzzük/osztályozzuk a linkeket információk hozzáadásával
  - Használunk standard technológiákat
- *Ez a szemantikus web alapja*



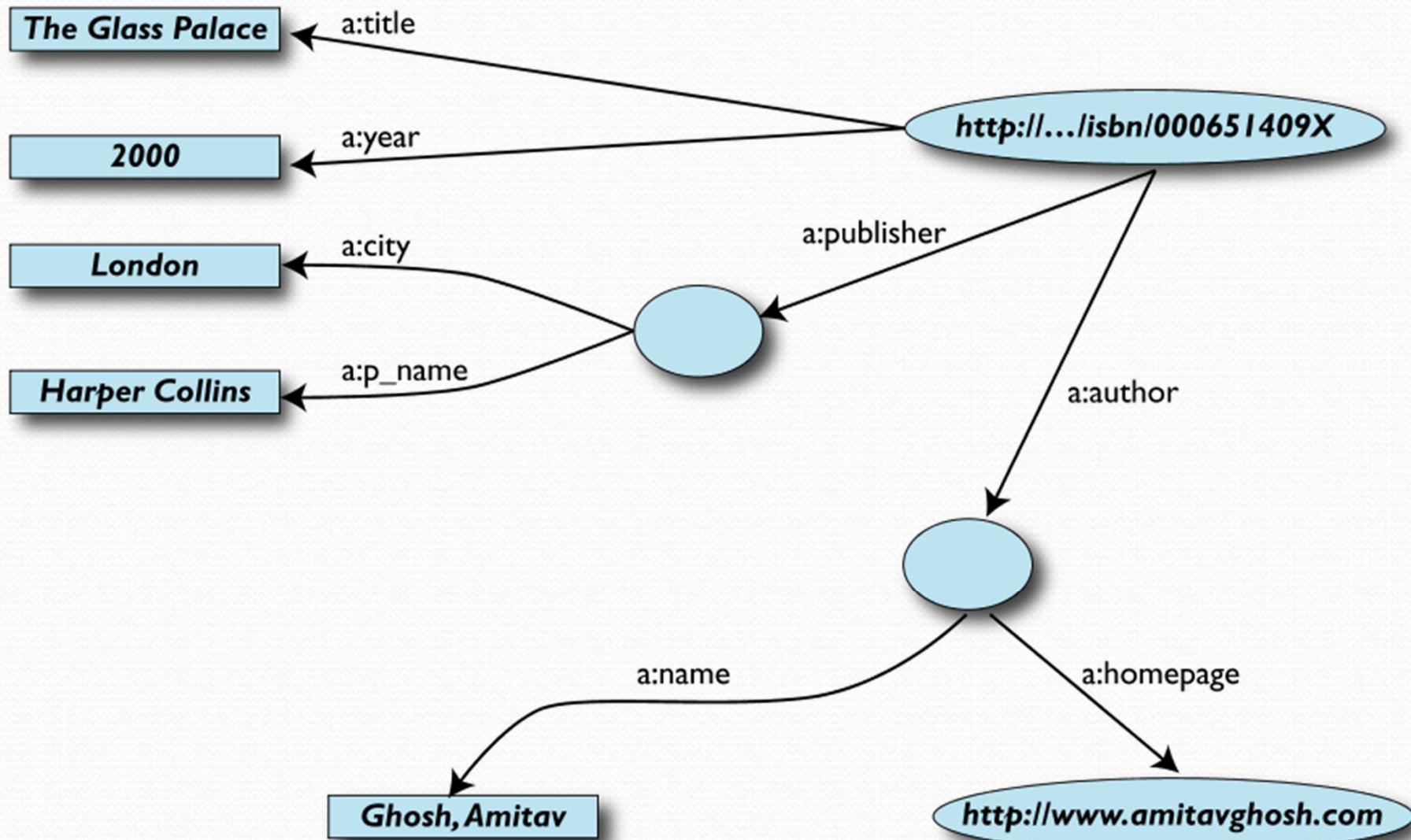
# Példa: könyvesbolt adatai

| ID                | Author | Title            | Publisher | Year |
|-------------------|--------|------------------|-----------|------|
| ISBN0-00-651409-X | id_xyz | The Glass Palace | id_qpr    | 2000 |

| ID     | Name          | Home Page                                                           |
|--------|---------------|---------------------------------------------------------------------|
| id_xyz | Ghosh, Amitav | <a href="http://www.amitavghosh.com">http://www.amitavghosh.com</a> |

| ID     | Publ. Name     | City   |
|--------|----------------|--------|
| id_qpr | Harper Collins | London |

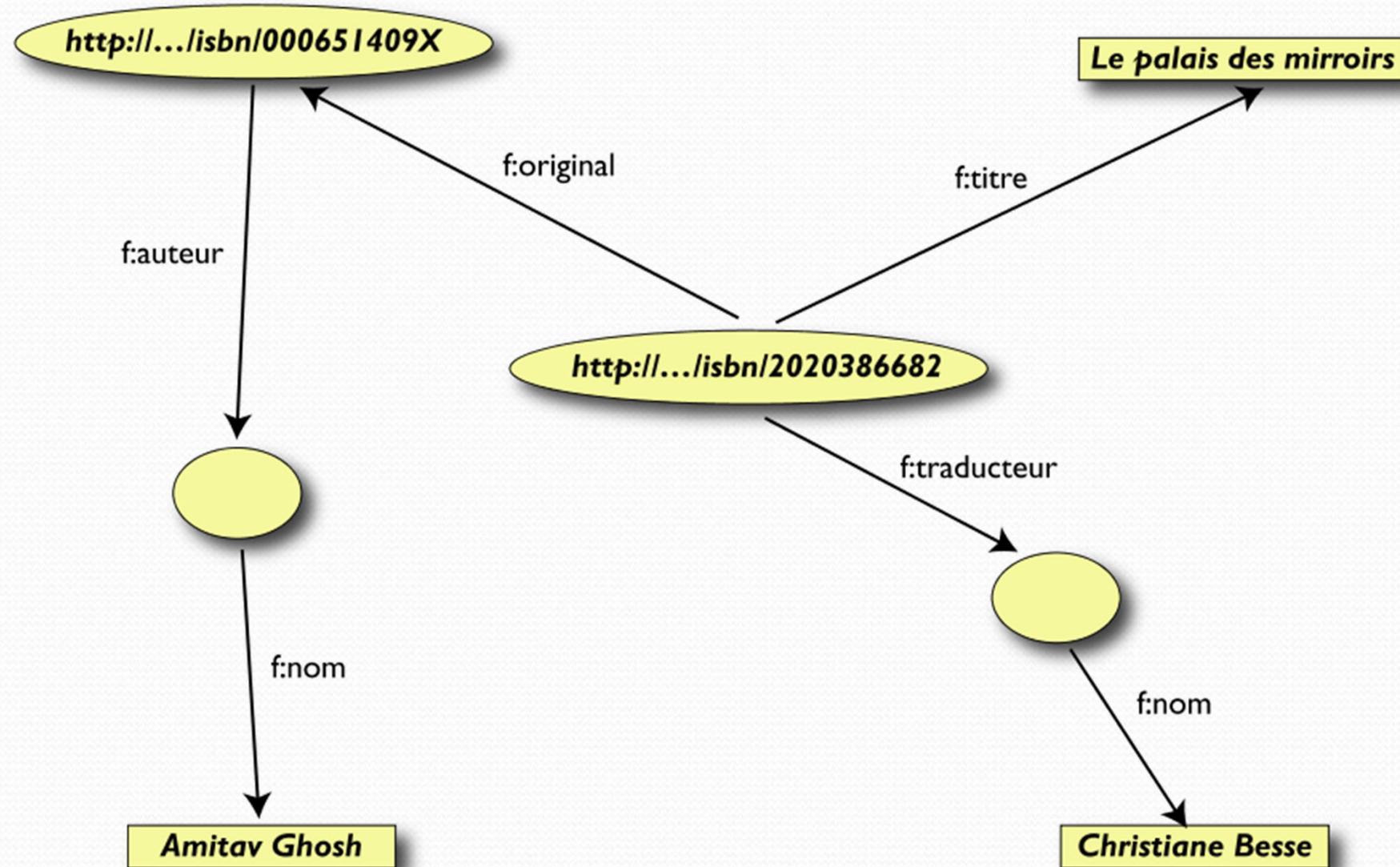
# Adatok exportálása relációkként



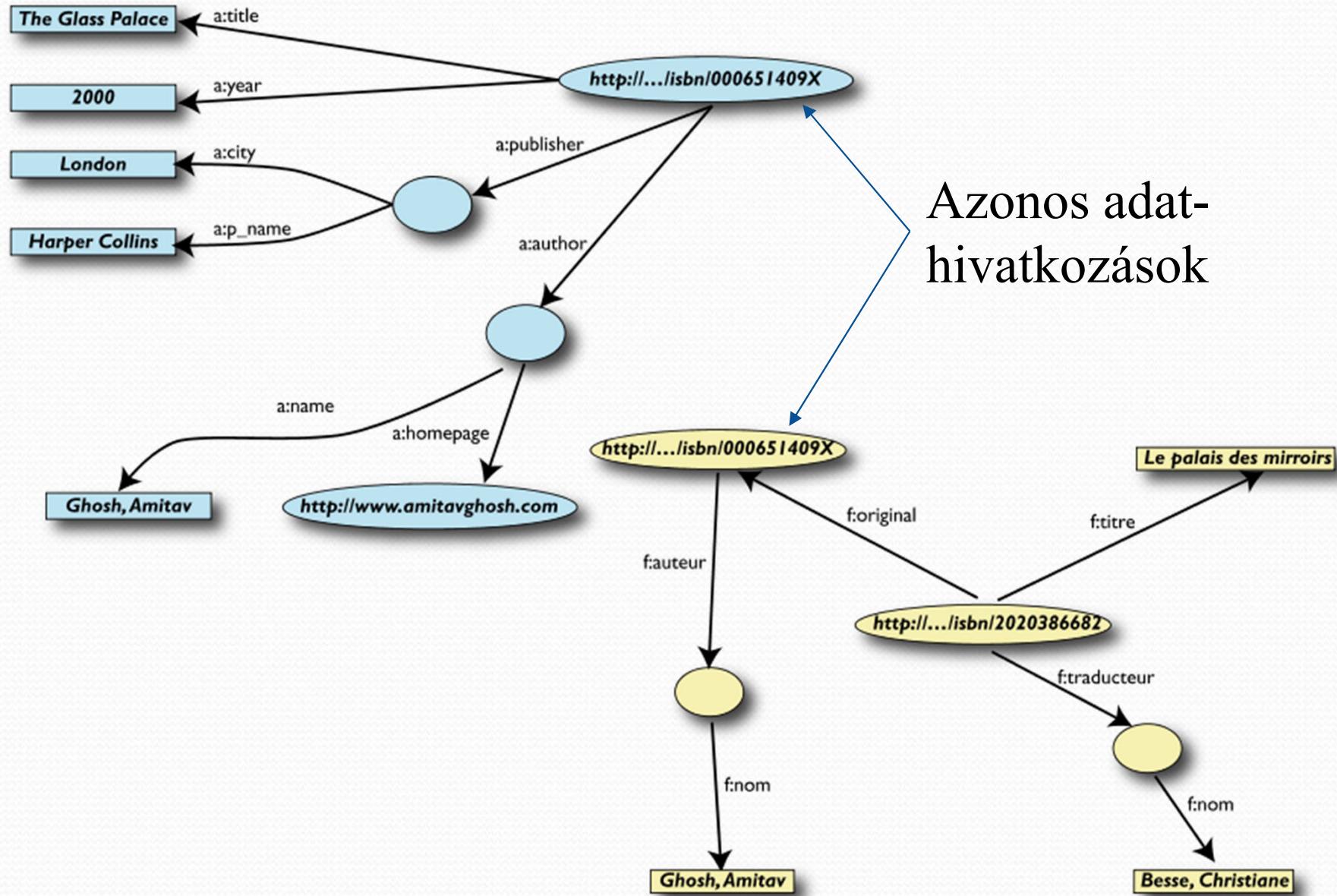
# Egy másik könyvesbolt adatai

|    | A                  | B                           | D                 | E                  |
|----|--------------------|-----------------------------|-------------------|--------------------|
| 1  | <b>ID</b>          | <b>Titre</b>                | <b>Traducteur</b> | <b>Original</b>    |
| 2  | ISBN0 2020386682   | Le Palais<br>des<br>miroirs | A13               | ISBN-0-00-651409-X |
| 3  |                    |                             |                   |                    |
| 6  | <b>ID</b>          | <b>Auteur</b>               |                   |                    |
| 7  | ISBN-0-00-651409-X | A12                         |                   |                    |
| 11 | <b>Nom</b>         |                             |                   |                    |
| 12 | Ghosh, Amitav      |                             |                   |                    |
| 13 | Besse, Christianne |                             |                   |                    |

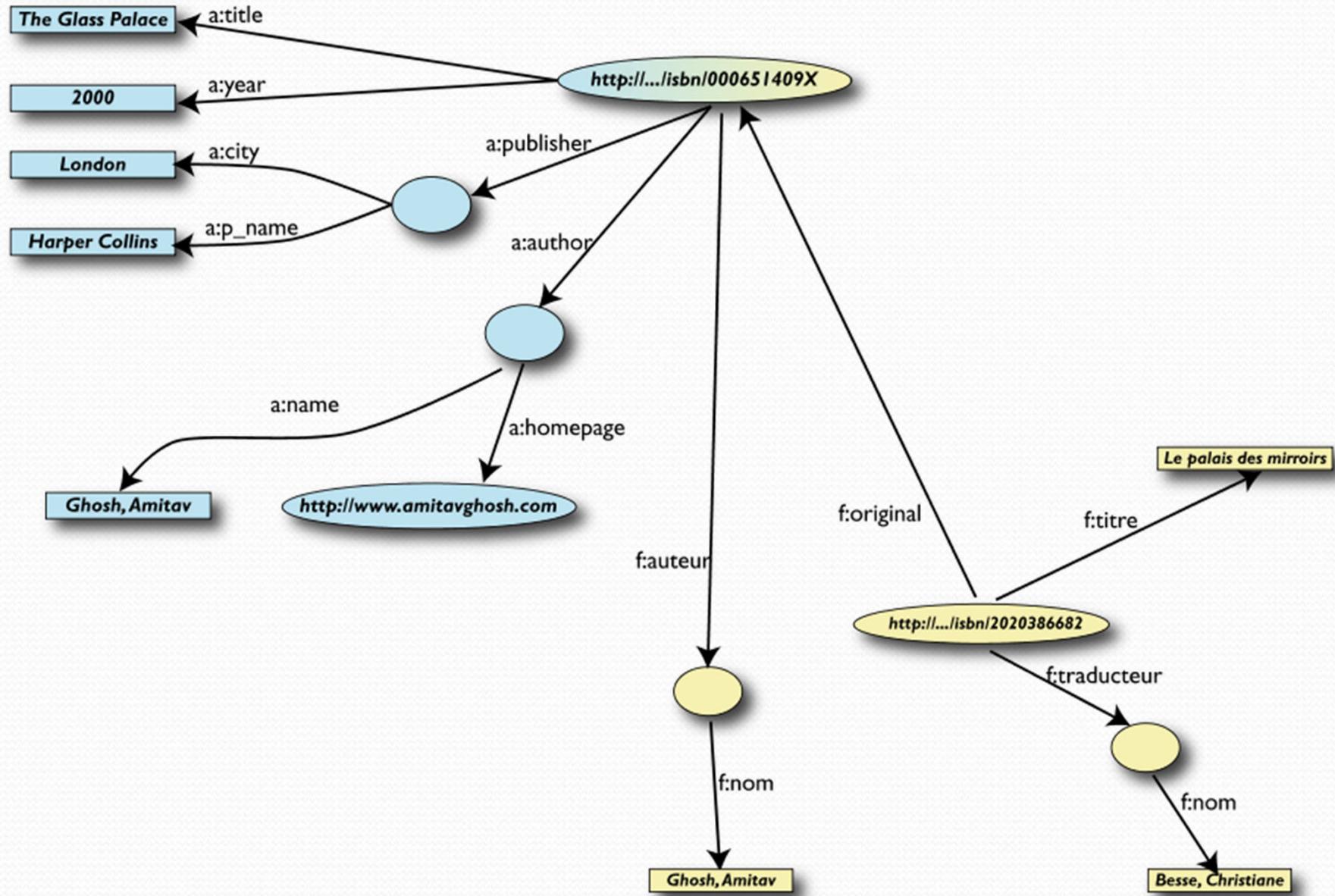
# A második könyvesbolt adatainak exportja



# Kapcsoljuk össze az adatokat

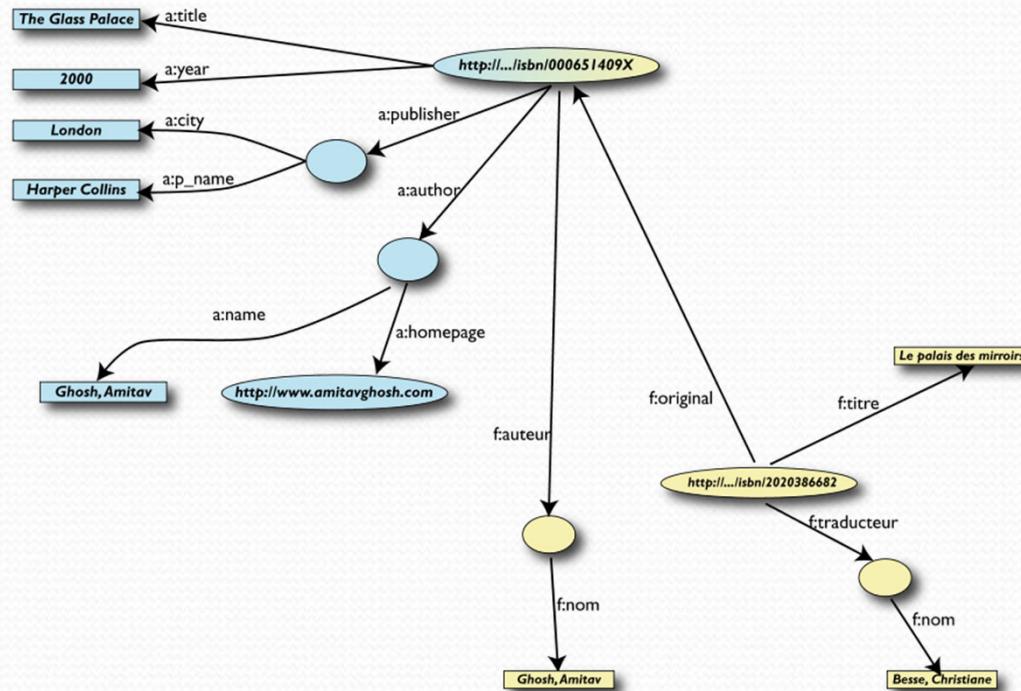


# Amennyiben identikusak az elemek:



# Írunk lekérdezéseket a kapcsolatokon keresztül:

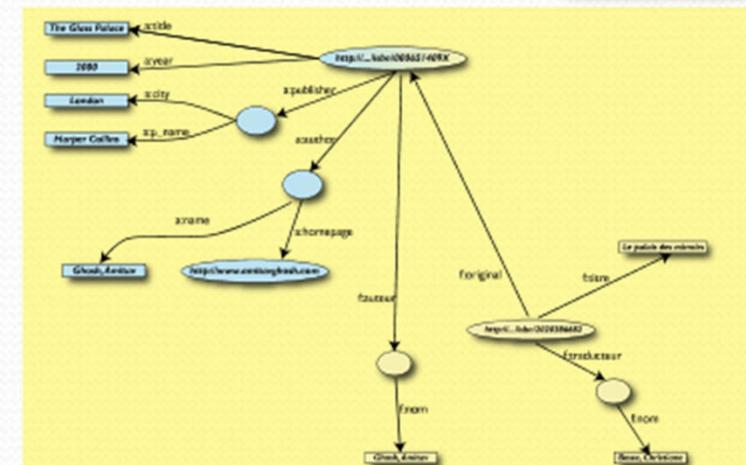
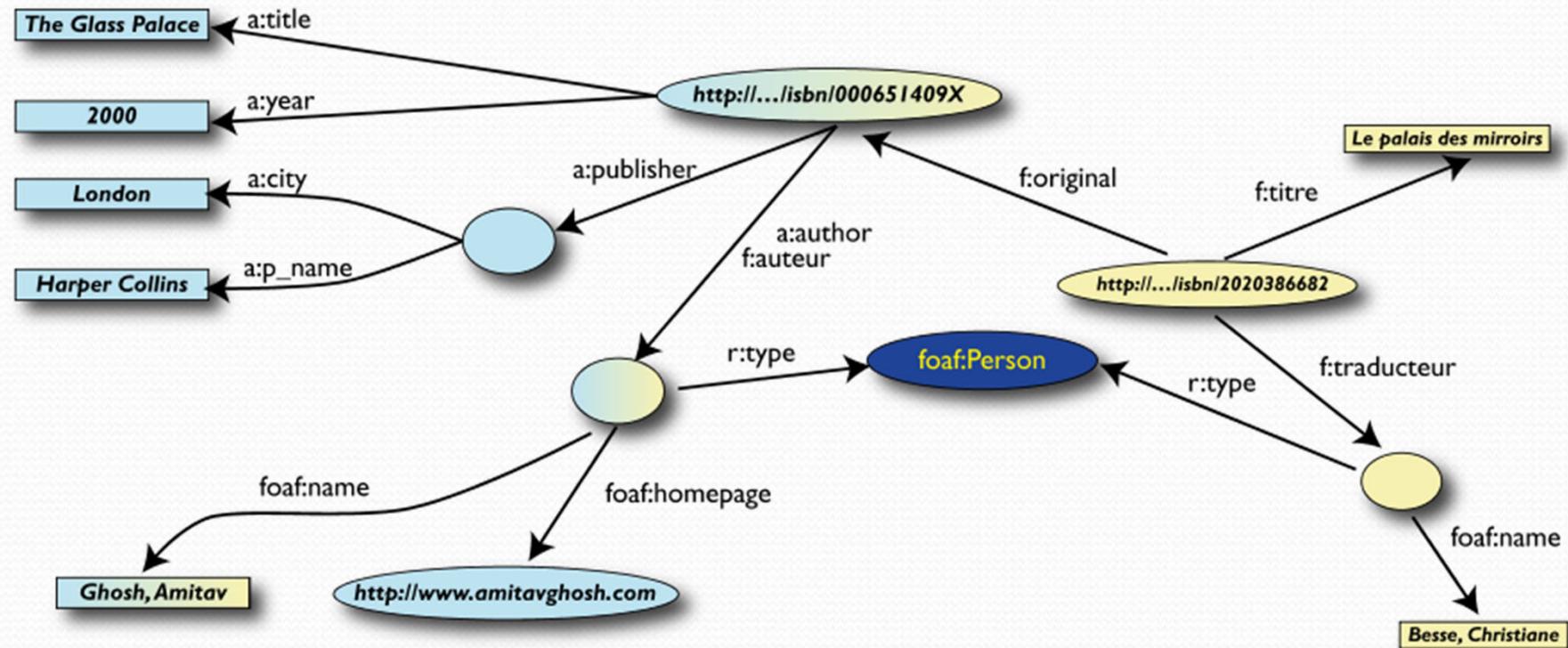
- Az első könyvesbolt adatai kiegészíthető például az eredeti könyvre vonatkozó információkkal



# További kapcsolatok is felfedezhetőek...

- Vélhetően az **a:author** és az **f:auteur** azonos elemre mutat
- Automatikus összekapcsoláshoz: adjunk további információt a leíráshoz
  - **a:author** legyen azonos **f:auteur** erőforrással
  - Mindkettő személyt azonosít
  - Ilyen fogalmakat már a webes közösség definiált:
    - egy “Person” elem azonosítható a nevével és a honlapjával
    - Használjuk ezt kategóriaként

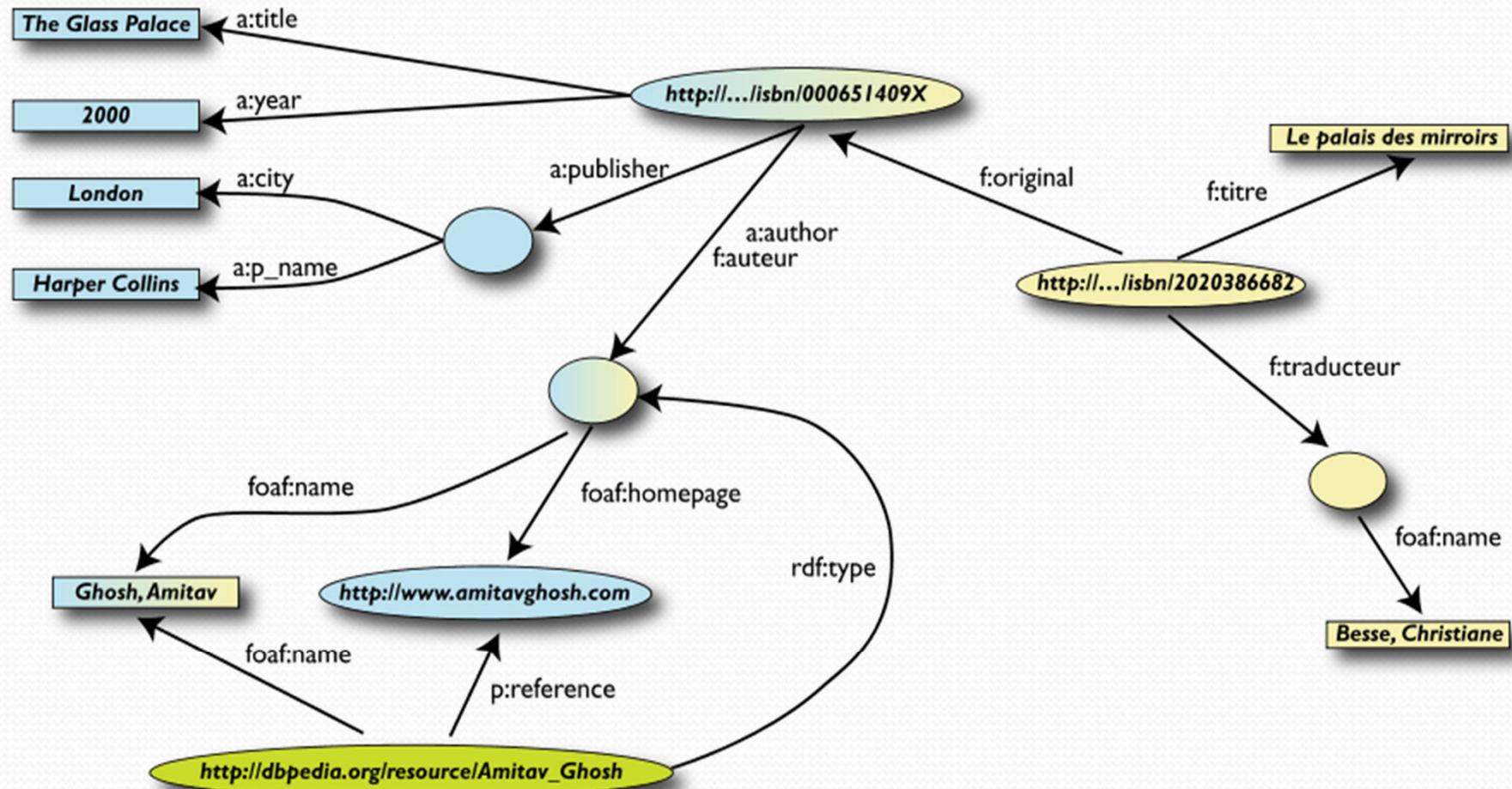
# Adatháló kiegészíthető, lekérdezhető így:



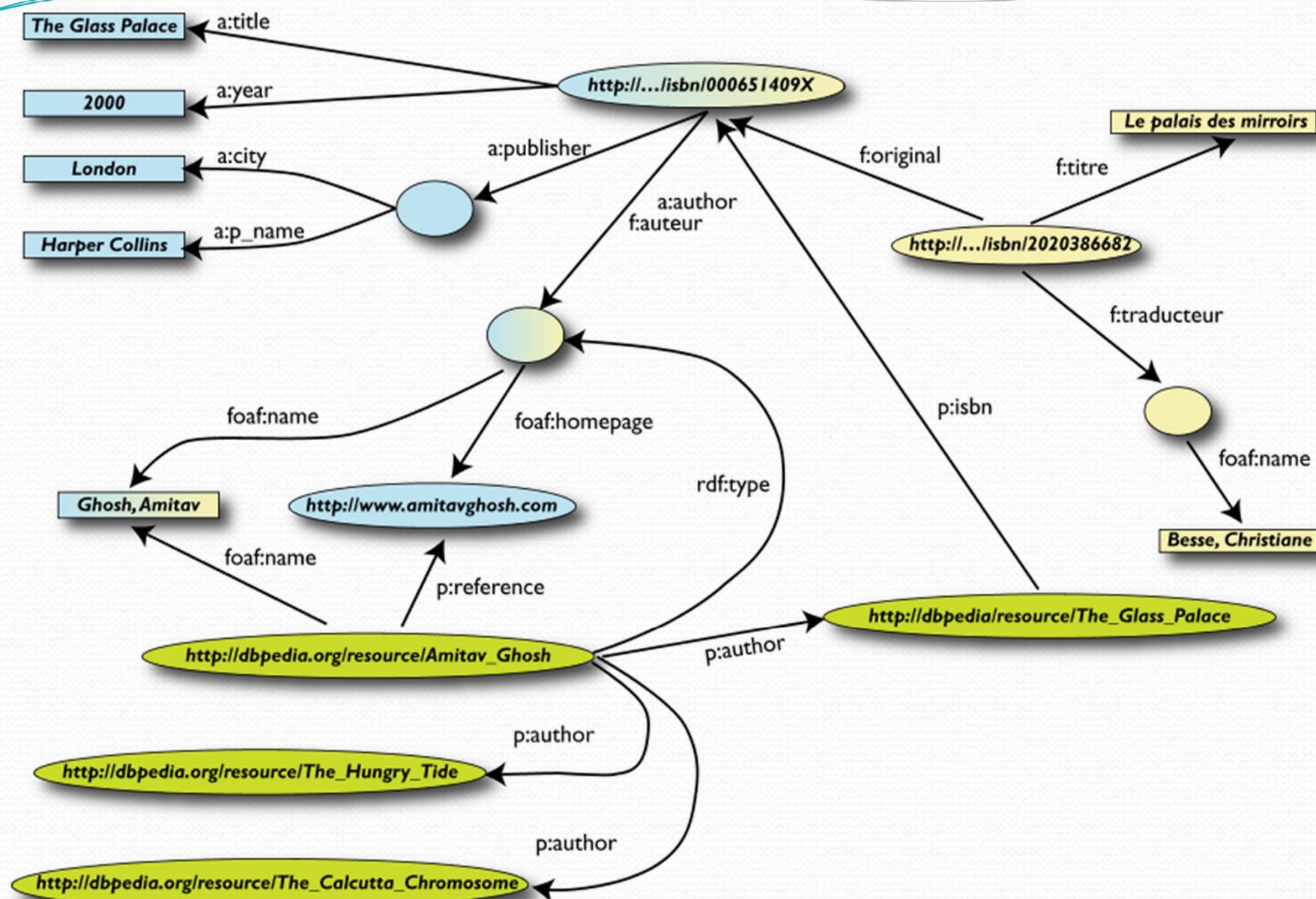
# Bővítés újabb adathalmazokkal

- Például a “Person” típus esetén, használhatóak a Wikipédia adatai:
  - pl., a “[dbpedia](#)” projekt már feldolgozta a Wikipédia adatait

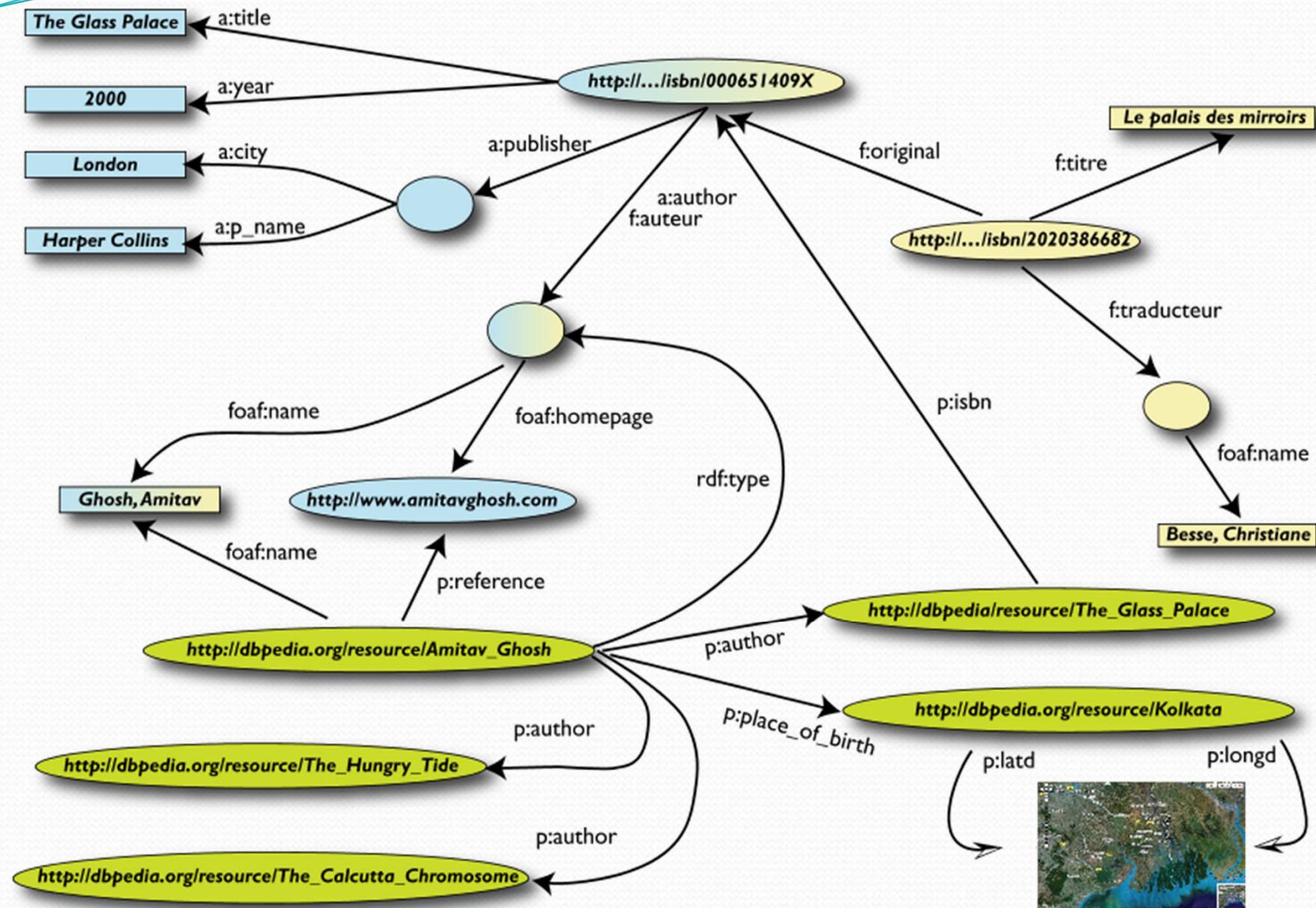
# Összekapcsolás Wikipedia adatokkal



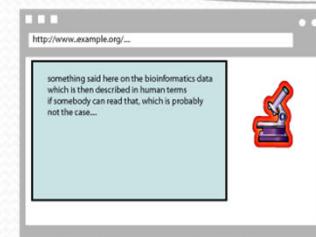
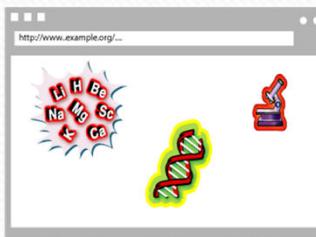
# Összekapcsolás Wikipedia adatokkal



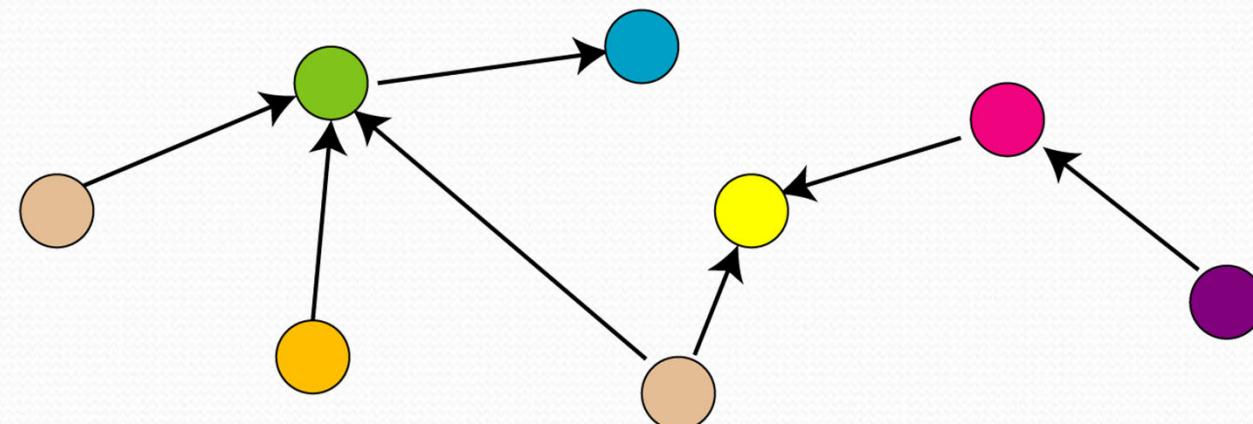
# És így tovább...



# Az adatok webje kialakulófélben...

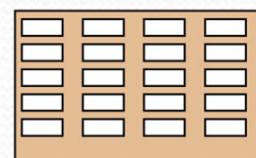
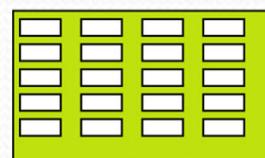
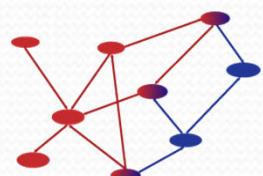


Alkalmazások



Lekérdezés,  
adatmódosítás

Absztrakt adatstruktúra



Leképezés,

Forrás adatok különböző formátumokban



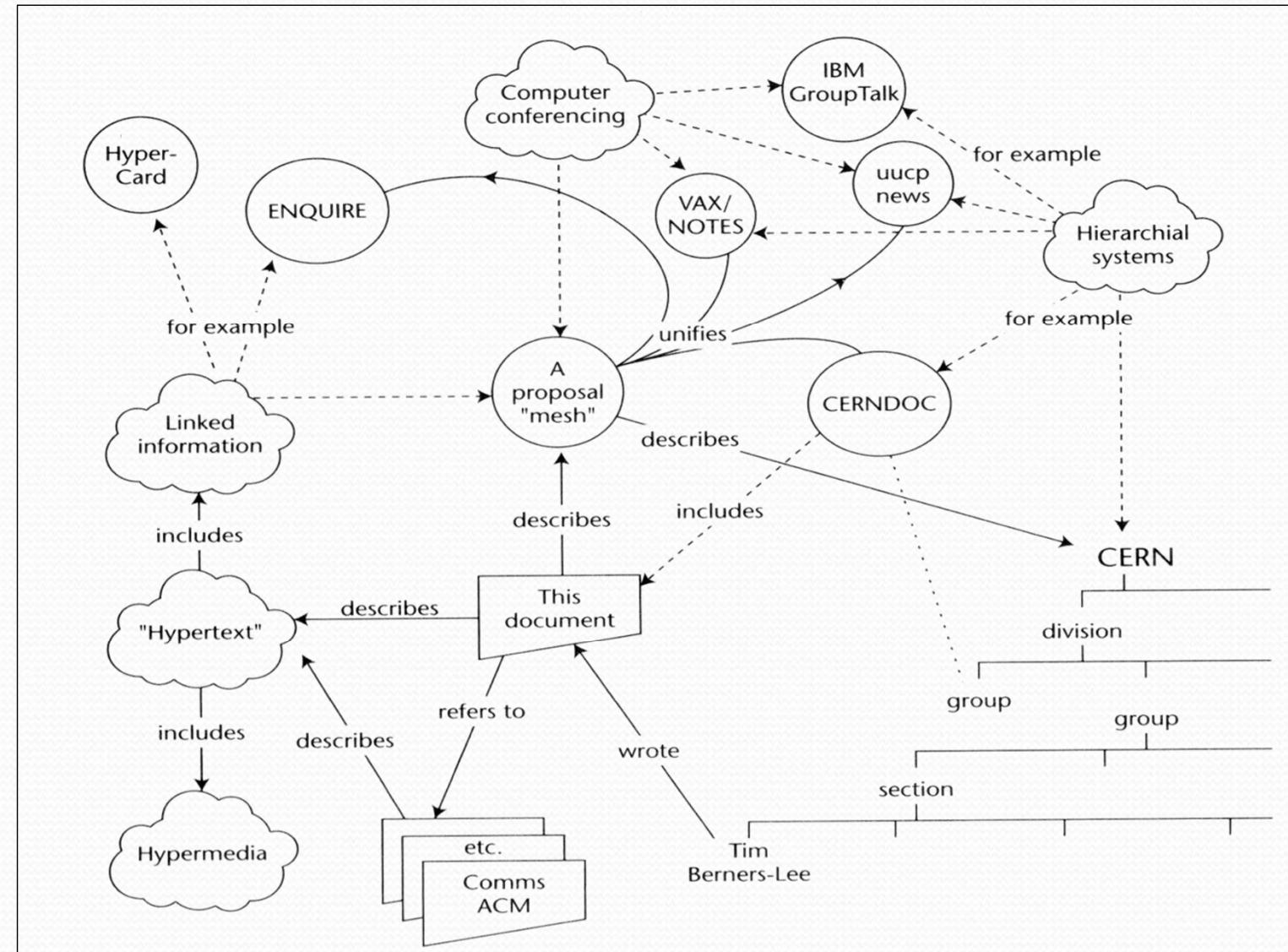
Szemantikus web

# Szemantikus Web

- A szemantikus web alkalmas megközelítés, megfelelő nyelvekkel, eszközökkel támogatja az intelligens információs rendszerek fejlesztését az elosztott információs környezetben.
- A SzW alapja a hagyományos web hálózat, így egyáltalán nem nyilvánvaló, hogy alkalmas a feladatra.
- A SzW technológia lehetőséget teremt az ágens alapú intelligens megoldások felhasználására a web területen.

# A Szemantikus Web eredete

- Tim Berners-Lee eredeti 1989-es WWW javaslata a Web-et információ menedzselő funkciókkal ellátott objektumok kapcsolataiként jellemzi.



<http://www.w3.org/History/1989/proposal.html>

# W3C szervezet célkitűzései

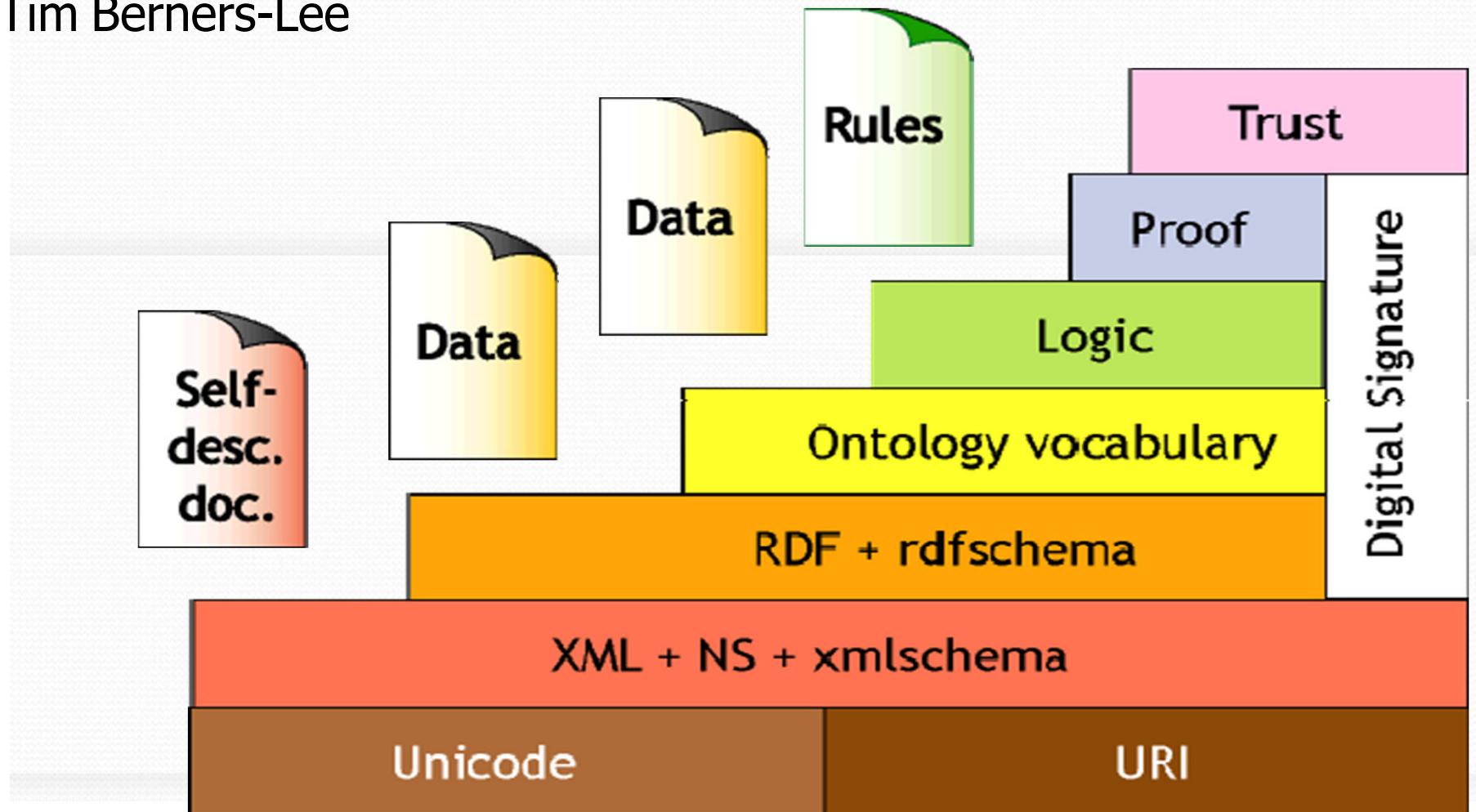
- Megközelítés – számítógépek jobb kihasználtságának biztosítása:

*„A szemantikus web egy kiterjesztése a jelenlegi web-nek, amelyben az információknak jól definiált jelentést adhatunk, lehetővé téve a gépek és felhasználók jobb együttműködését..” -- Berners-Lee, Hendler and Lassila, The Semantic Web, Scientific American, 2001*

- A jelenlegi web tárol dolgokat, míg a szemantikus web képes működtetni dolgokat.

# TBL szemantikus web felépítése

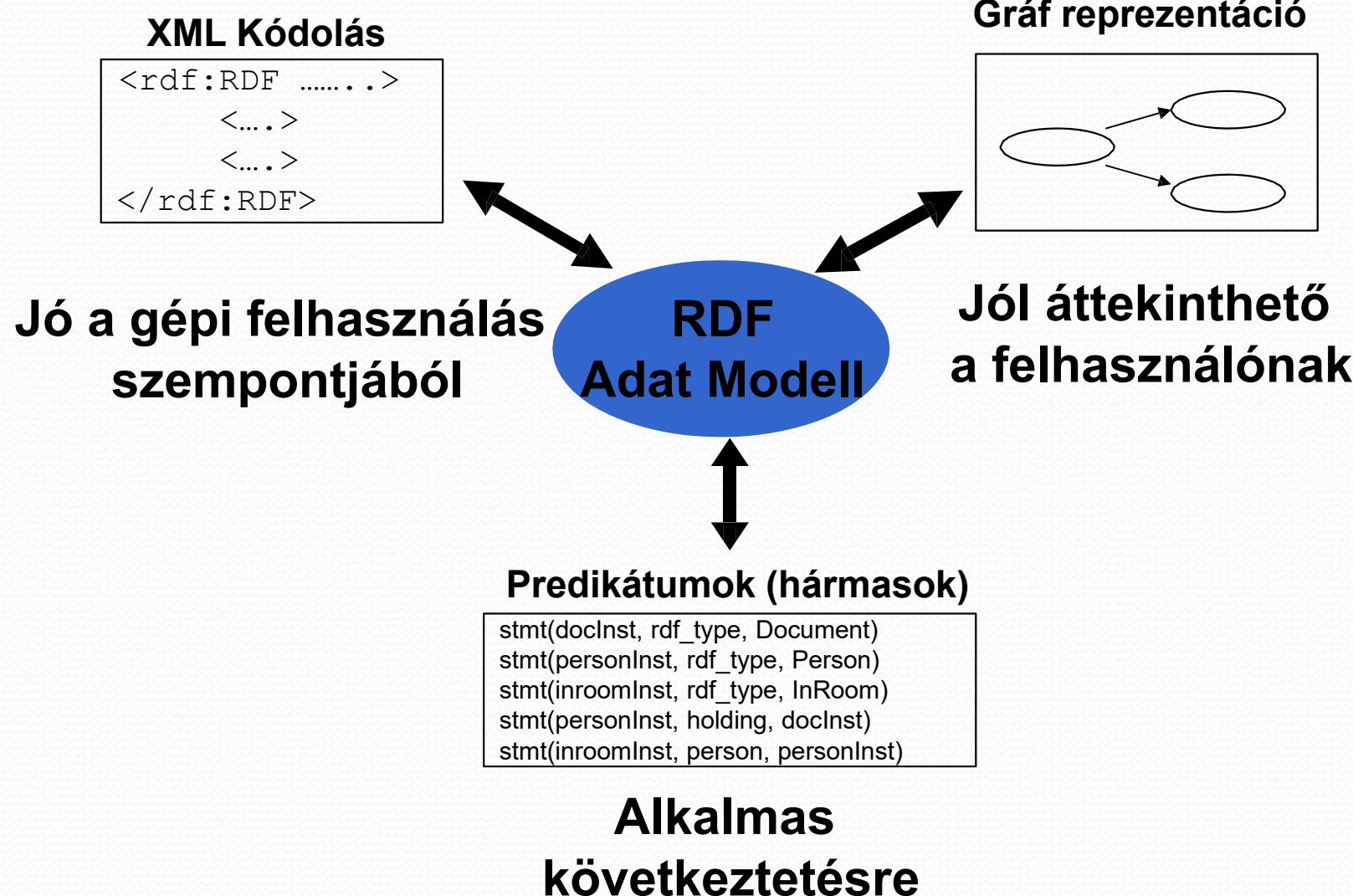
"A szemantikus web  
elérhetővé teszik a tudást,  
mint a web a hipertext-et --  
Tim Berners-Lee



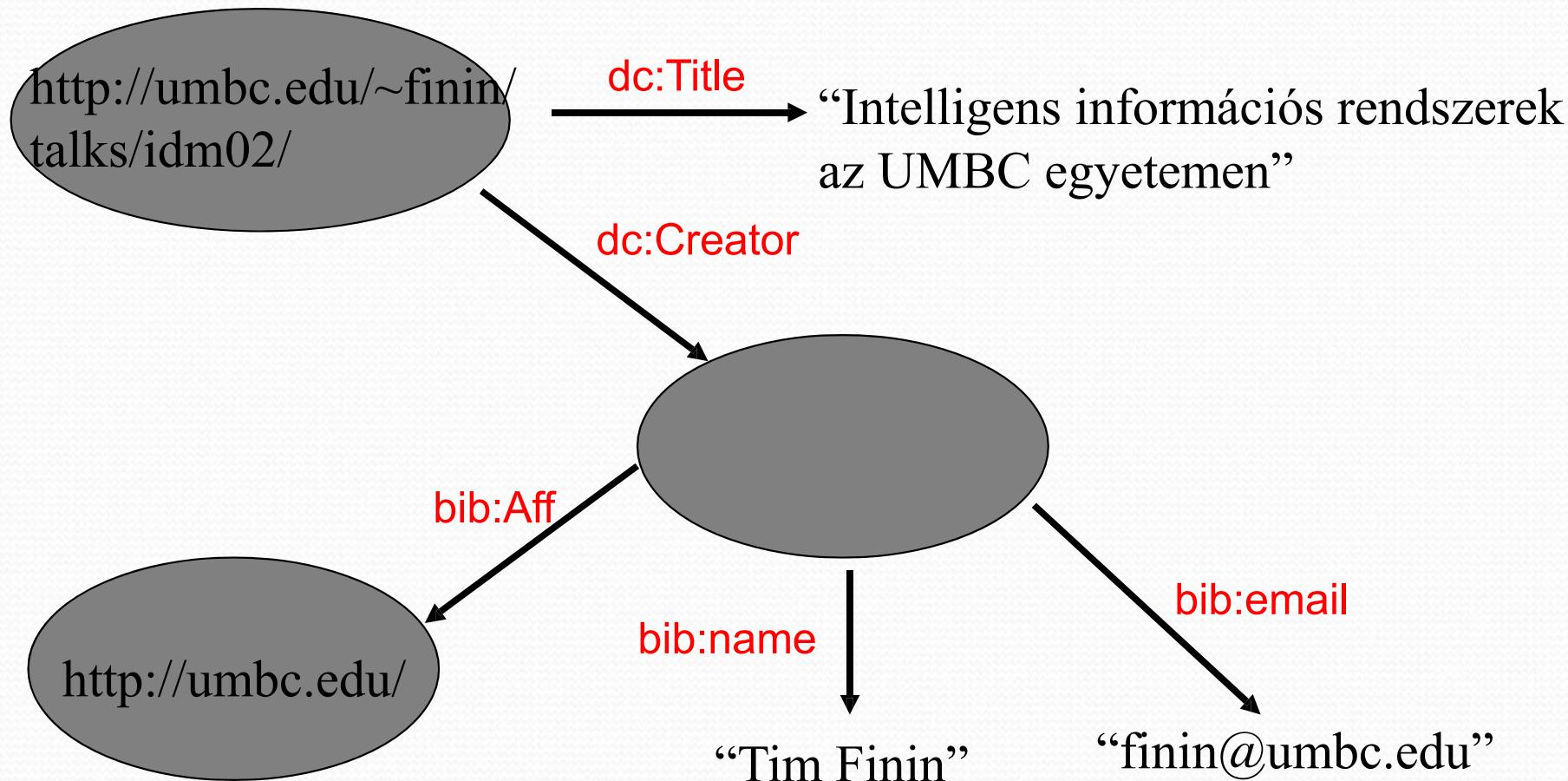
# Dokumentumok

- **RDF Primer**
  - URI: <http://www.w3.org/TR/rdf-primer>
- **OWL Guide**
  - URI: <http://www.w3.org/TR/owl-guide/>
- **RDF Test Cases**
  - URI: <http://www.w3.org/TR/rdf-testcases/>
- **RDF: Concepts and Abstract Syntax**
  - URI: <http://www.w3.org/TR/rdf-concepts/>
- **RDF szemantika**
  - URI: <http://www.w3.org/TR/rdf-mt/>
  - Precíz, gráfokon alapuló szemantika
- **RDF/XML szintaxis**
  - URI: <http://www.w3.org/TR/rdf-syntax-grammar/>
- **RDF Vocabulary Description Language (RDF Schema)**
  - URI: <http://www.w3.org/TR/rdf-schema/>
- **Semantic Web/RDF Interest Group**
  - Vitafórum, alkalmazások
  - URI: <http://www.w3.org/RDF/Interest>
- **RDF Logic**
  - Nyilvános levelezési lista részletesebb szakmai vitákhoz
  - URI: <http://lists.w3.org/Archives/Public/www-rdf-logic/>
- **Annotation and Collaboration**
  - Nyilvános levelezési lista RDF-alapú annotációs rendszerekről
  - URI: <http://lists.w3.org/Archives/Public/www-annotation/>
- **W3C Semantic Web Home page**
  - URI: <http://www.w3.org/2001/sw/>

# RDF - az első SzW nyelv



# Egyszerű RDF példa



# A példa XML szintaxiszáll

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:dc="http://purl.org/dc/elements/1.1/"
 xmlns:bib="http://daml.umbc.edu/ontologies/bib/">
<description about="http://umbc.edu/~finin/talks/idm02/">
 <dc:title>Intelligent Information Systems on the Web and in the
Aether</dc:Title>
 <dc:creator>
 <description>
 <bib:Name>Tim Finin</bib:Name>
 <bib:Email>finin@umbc.edu</bib:Email>
 <bib:Aff resource="http://umbc.edu/">
 </description>
 </dc:Creator>
</description>
</rdf:RDF>
```

# Hármasokat alkalmazó reprezentáció

- RDF kifejezések leírhatóak hármasokkal:
- <alany> <állítmány> <tárgy>

Megengedett szintaxis:  
<URI><URI><URI>  
<URI><URI><string>

# RDF tervezési szempontok

- Egyszerű adatmodell
- Formális szemantika és egyszerű következtetési lehetőségek
- Bővíthető URI
- XML alapú szintaktika (is)
  - XML séma adattípusok
- Bárki megfogalmazhat állításokat az erőforrásokról

# Alapelvek (1/2)

- Külön értelmezhetően definiálva
  - Modell struktúra (RDF gráf)
  - Interpretációs szemantika (vonzatok)
  - Szintaktikák (XML, TN, N3, ...)
- Mindössze két alap adattípus
  - URI/URIref: minden URI-val azonosított
  - Literálisok
    - String vagy más XSD adattípus

# Alapelvek (2/2)

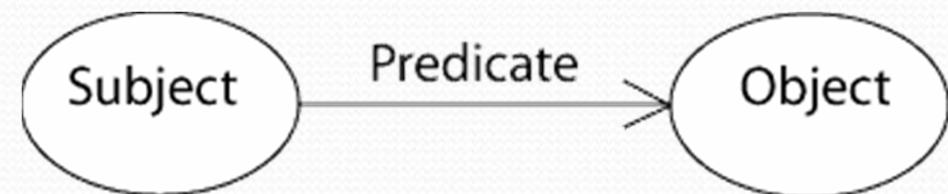
- Integrálható a webes információkkal
  - XML séma adattípusok
  - Referenciák http elérésű információkhoz
- Nyílt világ feltételezés
  - Bárki megfogalmazhat állításokat bármilyen erőforráshoz
  - Nem garantált a teljesség
  - Nem garantált a konzisztencia

# Alapelemek

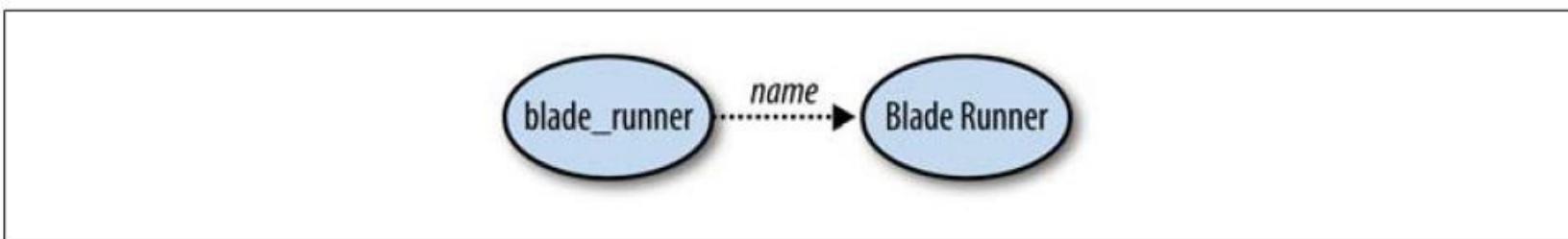
- Gráf adatmodell
- URI alapú szótárak
- Adattípusok
- Literálisok
- XML szerIALIZÁCIós szintaktika
- Egyszerű tények leírása
- Következtetés

# Graph adatmodell

- Hármasok: alany, állítmány, tárgy
- Kifejezések: hármasok gyűjteménye

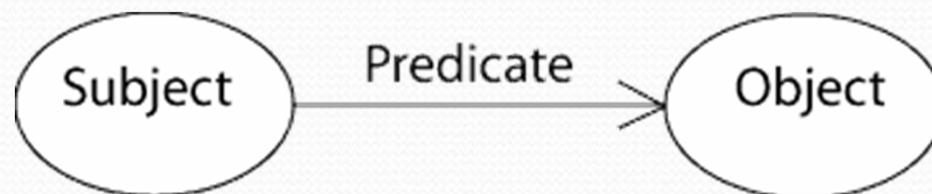


# Példa

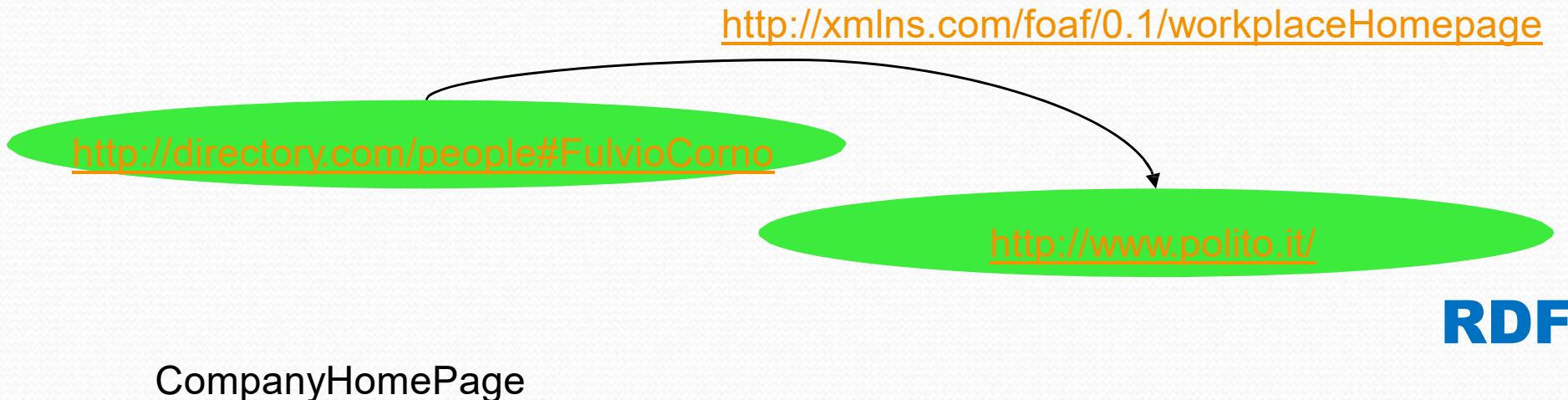


# Terminológia és kényszerek

- Alany és tárgy a csomópontok
- Állítmány és tulajdonság szinonimák
- Különleges meg nem nevezett csomópontok: üres csomópontok
- Alany: URI referencia vagy üres csomópont
- Állítmány: URI referencia
- Tárgy: URI referencia, literális, üres csomópont



# Információ a hármasokban



| PersonID    | Homepage                                                  |
|-------------|-----------------------------------------------------------|
| FulvioCorno | <a href="http://www.polito.it/">http://www.polito.it/</a> |

## Relációs adatbázis

**Első rendű logikai  
predikátum**

```
HasCompanyHomePage(
 'FulvioCorno',
 'http://www.polito.it/');
```

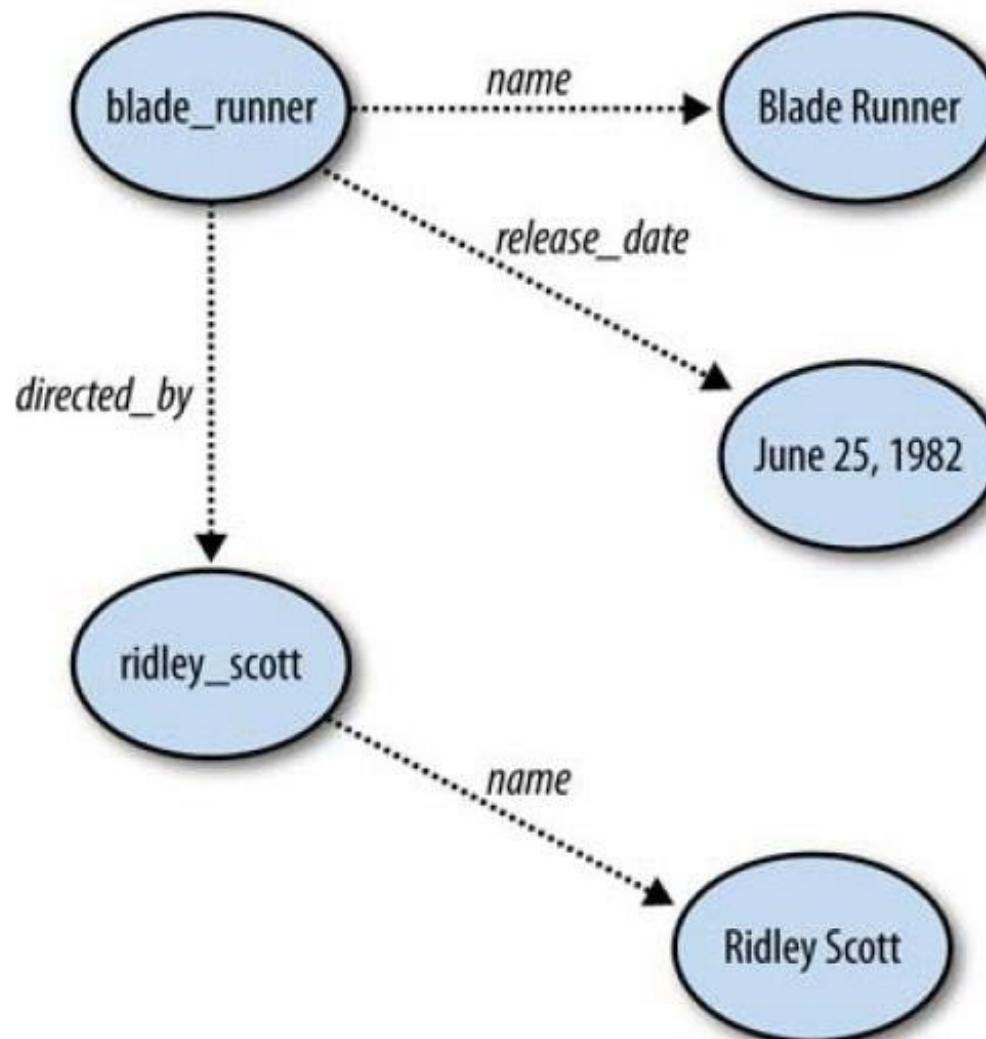
# Hármások vs adatbázisok

| FEATURE                 | RELATIONAL DATABASE           | KNOWLEDGEBASE           |
|-------------------------|-------------------------------|-------------------------|
| Structure               | Schema                        | Ontology statements     |
| Data                    | Rows                          | Instance statements     |
| Administration language | DDL                           | Ontology statements     |
| Query language          | SQL                           | SPARQL                  |
| Relationships           | Foreign keys                  | Multidimensional        |
| Logic                   | External of database/triggers | Formal logic statements |
| Uniqueness              | Key for table                 | URI                     |

# Összehasonlítva...

- Relációs adatbázisban tetszőleges számú oszlopot definiálhatunk
- Elsőrendű logikában a predikátumok tetszőleges elemet (argumentumot) tartalmazhatnak
- Egy RDF hármas csak egy alanyt és egy tárgyat tartalmazhat
  - Komplex kijelentéseket dekomponálni kell

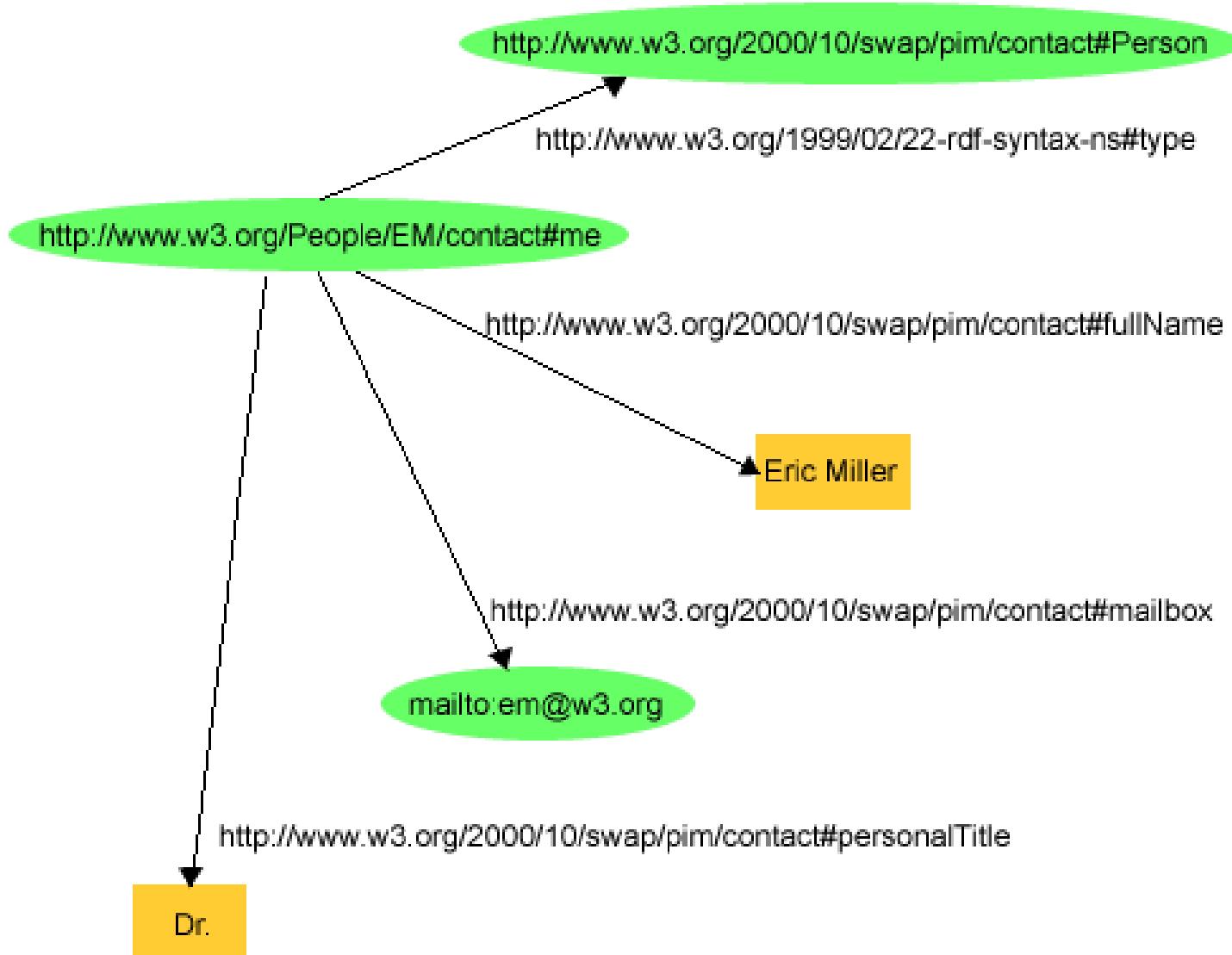
# Példa



# Példa

- Írjuk le RDF állításokkal a következő kijelentést
- "Ilétezik egy személy, akit a <http://www.w3.org/People/EM/contact#me> oldallal azonosíthatunk, akinek neve Eric Miller, email címe [em@w3.org](mailto:em@w3.org), és a titulusa Dr."

# Példa



# RDF áttekintés

## A formális modell alapelemei:

Két alaphalmaz: erőforrások (resources) és literálisok (literals)

Az erőforrások egy fontos részhalmaza: Tulajdonságok (properties).

Definiálunk egy hármaskóból álló halmazt: Állítások (statements), amelyek formája: {alany, állítmány, tárgy}, ahol az alany egy erőforrás,  
az állítmány egy tulajdonság,  
a tárgy vagy erőforrás vagy literális.

# RDF adatmodell

- Erőforrások (Resources)
  - URI azonosítja
  - Kijelentés vonatkozik rá
- Tulajdonságok (Properties)
  - Erőforráshoz kapcsolt jellemző
  - A tulajdonság is erőforrás
- Literálok (Literals)
  - Karaktersorozatok
- Kijelentések (Statements)
  - Alany (subject); erőforrás
  - Állítmány (predicate); tulajdonság
  - Tárgy (object); erőforrás vagy literál

# Ezek segítségével megfogalmazhatunk leírásokat:

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:s="http://description.org/schema/">
 <rdf:Description about="http://www.w3.org/Home/Lassila">
 <s:Creator>
 <rdf:Description about="http://www.w3.org/staffId/85740">
 <rdf:type resource="http://description.org/schema/Person"/>
 <v:Name>Ora Lassila</v:Name>
 <v:Email>lassila@w3.org</v:Email>
 </rdf:Description>
 </s:Creator>
 </rdf:Description>
</rdf:RDF>
```

# Az RDF szerepe az SZV hierarchiában

- Technológiát és módszert ad ahhoz, hogy dokumentumainkhoz jelentést rendeljünk egy jól olvasható formában
- Jó lehetőség, de a szemantikai információk nem túl hasznosak, amíg strukturáltalan és nem tudjuk konzisztens módon értelmezni.

(XML séma kevés: csak szintaktikáról szól, nem ad lehetőséget a dokumentumon kívüli dolgok leírására)

# Séma hiányában ugyanannak a tartalomnak sokféle reprezentációja lehetséges:

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-
syntax-ns#" />

<rdf:Description
 about="http://www.w3.org/Home/Lassila">
 <Creator>
 <rdf:Description
 about="http://www.w3.org/staffId/85740">
 <rdf:type
 resource="http://desc.org/schema/Person"/>
 <Name>Ora Lassila</Name>
 <Email>lassila@w3.org</Email>
 </rdf:Description>
 </Creator>
 </rdf:Description>
</rdf:RDF>
```

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-
syntax-ns#" />

<rdf:Description
 about="http://www.w3.org/Home/Lassila">
 <author>
 <rdf:Description
 about="http://www.w3.org/staffId/85740">
 <rdf:type
 resource="http://desc.org/schema/Person"/>
 <name>
 <surname>Lassila</surname>
 <given>Ora</given>
 </name>
 <email>lassila@w3.org</email>
 </rdf:Description>
 </author>
 </rdf:Description>
</rdf:RDF>
```

# SPARQL

- Lekérdező nyelv RDF-hez
- SPARQL gráf illesztésen alapuló lekérdező nyelv
- Gráf minták

- példa:

```
<http://example.org/book/book1>
<http://purl.org/dc/elements/1.1/title>
 ?title .
```

- ?title - változó.

# Egy egyszerű SPARQL Query

- **Adat:**

```
<http://example.org/book/book1>
<http://purl.org/dc/elements/1.1/title>
"SPARQL Tutorial".
```

- **Lekérdezés:**

```
SELECT ?title
WHERE { <http://example.org/book/book1>
 <http://purl.org/dc/elements/1.1/title>
 ?title . }
```

- **Eredmény:**

|                   |
|-------------------|
| <b>title</b>      |
| "SPARQL Tutorial" |

# További példa

- **Adat:**

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
```

```
_ :a foaf:name "Johnny Lee Outlaw" .
```

```
_ :a foaf:mbox <mailto:jlow@example.com> .
```

```
_ :b foaf:name "Peter Goodguy" .
```

```
_ :b foaf:mbox <mailto:peter@example.org> .
```

```
_ :c foaf:mbox <mailto:carol@example.org> .
```

- **Lekérdezés:**

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox
WHERE { ?x foaf:name ?name . ?x foaf:mbox ?mbox }
```

- **Eredmény:**

| <b>name</b>         | <b>mbox</b>                |
|---------------------|----------------------------|
| "Peter Goodguy"     | <mailto:peter@example.org> |
| "Johnny Lee Outlaw" | <mailto:jlow@example.com>  |

# Lekérdezésel RDF literálisokkal

- **Példa RDF adatokra**

```
@prefix dt: <http://example.org/datatype#> .
@prefix ns: <http://example.org/ns#> .
@prefix : <http://example.org/ns#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:x ns:p "cat"@en .
:y ns:p "42"^^xsd:integer .
:z ns:p "abc"^^dt:specialDatatype .
```

# RDF Literálisok illesztése

- Lekérdezés 1:

```
SELECT ?v WHERE { ?v ?p "cat" }
```

Lekérdezés 2:

```
SELECT ?v WHERE { ?v ?p "cat"@en }
```

eltérő eredményt ad.

- Csak a második találja meg az előző példában az eredményt:

|                                                                   |
|-------------------------------------------------------------------|
| v                                                                 |
| < <a href="http://example.org/ns#x">http://example.org/ns#x</a> > |

# Üres csomópontok a lekérdezésekben

- **Adat:**

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
```

```
_ :a foaf:name "Alice" .
_ :b foaf:name "Bob" .
```

- **Lekérdezés:**

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?x ?name
WHERE { ?x foaf:name ?name . }
```

- **Eredmény:**

| x    | name    |
|------|---------|
| _ :c | "Alice" |
| _ :d | "Bob"   |

# Üres csomópontok a lekérdezésekben (modell bővítés)

- **Adat:**

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
```

```
_ :a foaf:name "Alice" .
_ :b foaf:name "Bob" .
_ :a foaf:knows _ :b .
_ :b foaf:knows _ :a .
```

- **Lekérdezés:**

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?x ?name1 ?y ?name2
WHERE { ?x foaf:name ?name1 . ?y foaf:name ?name2 .
 ?x foaf:knows ?y }
```

- **Eredmény:**

| ?x   | name1   | ?y   | name2   |
|------|---------|------|---------|
| _ :c | "Alice" | _ :d | "Bob"   |
| _ :d | "Bob"   | _ :c | "Alice" |

# RDF esettanulmányok

- Dublin Core
  - Magas szintű szótár definiálása
  - Elektronikus dokumentumok megtalálása
- Open Directory Project (OPD)
  - Webes katalógus keresők számára
- MusicBrainz
  - Hanganyagok (cd, mp3 ...) metaadatainak lekérésére
- RSS: RDF Site Summary
  - Hírek, események közzététele
- Wordnet
  - Szabadon letölthető szótár
  - Nem csak címszavakat, hanem kapcsolataokat is leír

# Integrációs és ellenőrzési technikák

## VIMIAC04, 2021. tavasz

Micskei Zoltán  
Strausz György

*Méréstechnika és Információs Rendszerek Tanszék*

<https://www.mit.bme.hu/oktatas/targyak/vimiac04>

# Integrációs és ellenőrzési technikák

Hogyan építsünk információ gazdag megoldásokat?

- Információ/adat integráció
  - Webes környezetek
  - Intézményi környezetek
- Technikák: szemantikus technológiák

Hogyan fejlesszünk jó minőségű szoftvert?

- Kód gyakori integrálása és ellenőrzése
- Kód review, kód analízis, tesztelés...
- Technikák: continuous integration (CI), mocking, automatikus tesztelés...

# Integrációs és ellenőrzési technikák

## *Kurzus menetrendje/követelmények*

Előadások: hétfő 10.15 – 12.00., IB-025

Gyakorlatok: péntek (8.15-10.00., 10.15-12.00, 12.15 – 14.00)

6 gyakorlat, ~~kéthetente, IB. 413. terem~~

5 gyakorlaton kötelező a részvétel

Gyakorlatok téma:

- 1-3. alkalmak: Szemantikus technológiák  
(adatmodellezés, integráció)
- 4-6. alkalmak: Tesztek írása, CI beüzemelése, kód átvizsgálása...

# Integrációs és ellenőrzési technikák

## *Kurzus menetrendje/követelmények*

A távoktatási üzemmódban nem tartunk ZH-t, a félévközi követelmények két HF megoldásával teljesíthetők.

**HF 1:** *Kiadás a 6. héten – Információ-keresés fejlesztés  
(IMSC pontok szerezhetőek)*

**HF 2:** *követelmény: elfogadás; 4 fős csapatok,  
GitHub infrastruktúra használata*

**Vizsga:** írásbeli

# Integrációs technikák - tartalom

- Bevezetés, probléma felvetés
- Szemantikus technológiák
  - Szemantikus web koncepció
  - RDF, SPARQL
  - RDFS, OWL
  - Linked Data
- Adatintegráció
  - Elosztott, webes környezetek
  - Mediátorok (lekérdezés átalakítások nézetek felhasználásával)
  - Intézményi környezetek, adattárházak

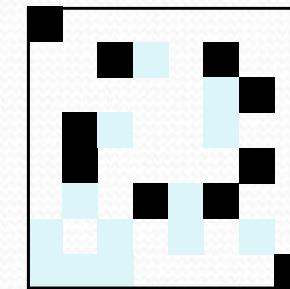
# Integráció: Tartalom vs Forma

Szemantikus nézet



*A megbeszélés sikere az érvelésen és a logikus következtetésen múlik.*

Szintaktikus nézet



A megbeszélés sikere  
\_az\_ érvelésen \_és\_ a\_ lo  
gikus\_ következtetésen  
\_múlik.

# Miért van szükségünk információ integrációra? (Alkalmazások)

- **WWW:**
  - Összehasonlítás alapú vásárlás
  - Portál építések több adatforrás felhasználásával
  - B2B, elektronikus piacterek
- **Tudomány és kultúra:**
  - Genetika: gén információk integrálása
  - Asztrofizika: égi jelenségek gyűjtése.
  - Kultúra: kulturális információs adatbázisok egységes elérése országhatáron túl
- **Vállalati adatintegráció**
  - Egy átlagos KNV 49 adatbázist alkalmaz és IT költségvetésének 30%-át az adatintegrációra költi (US, 2009)

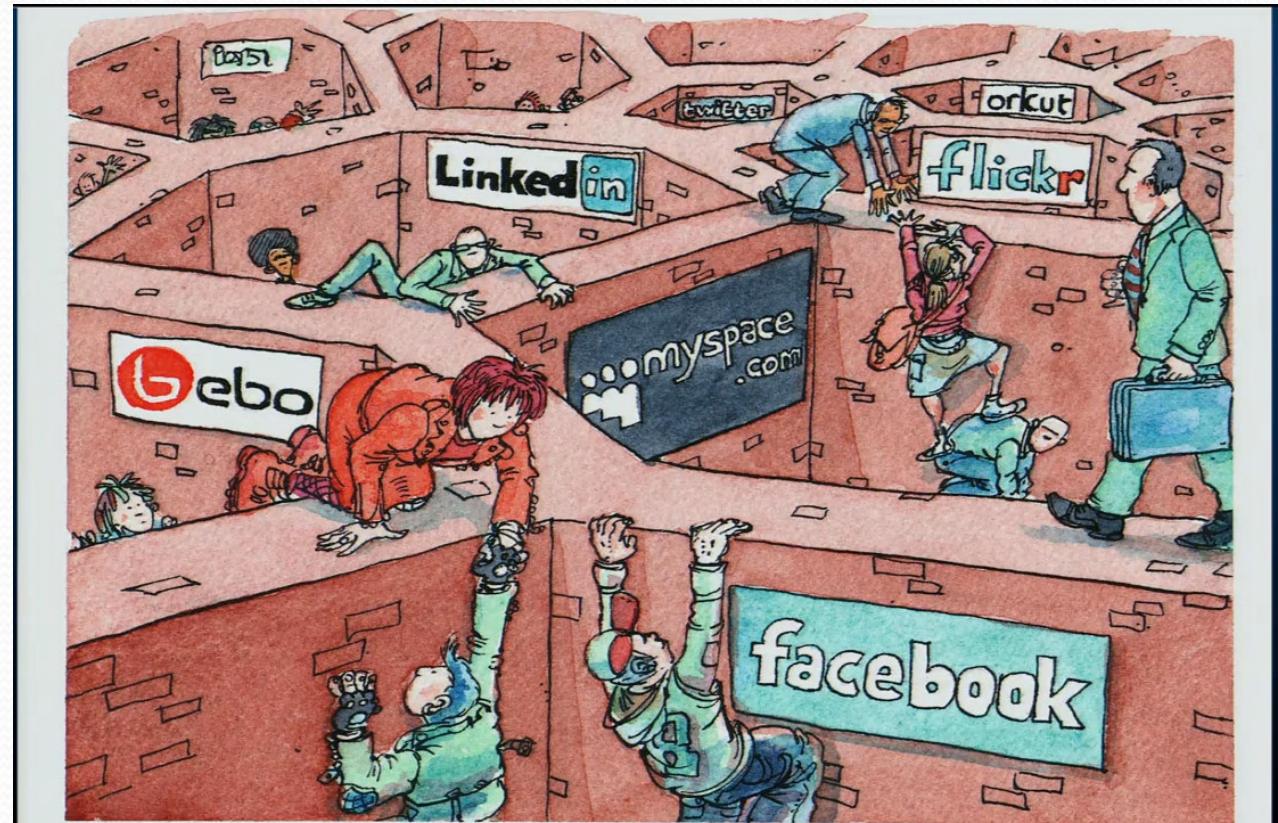


# Miért nem elég:

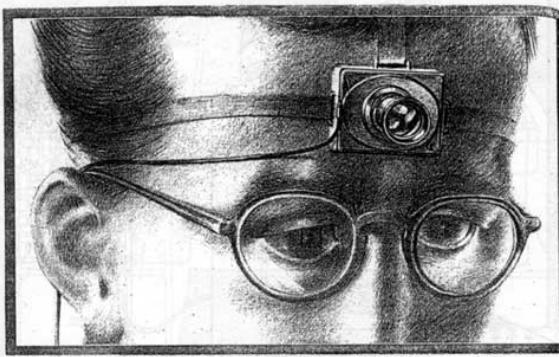
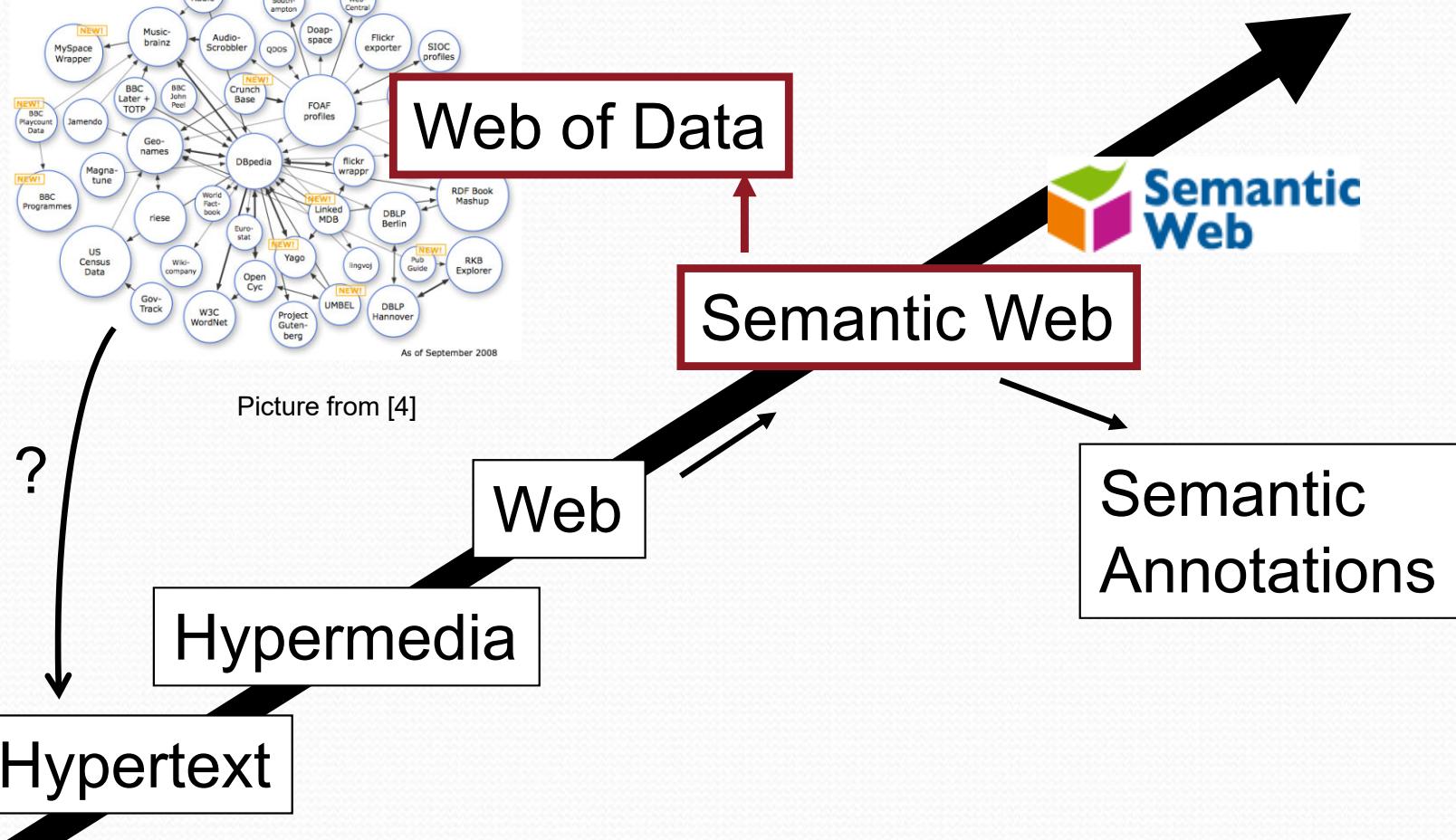
- Keresőgépek szövegalapú keresést végeznek
  - Jól működik egyedi dokumentumokon
    - Nem tudnak integrálni több dokumentumból származó információkat
    - Nem képesek hatékony általánosításra
    - Nem tudnak dokumentumokat és adatbázisokat összekapcsolni
  - Az információ integráció célja strukturált és félig-strukturált információforrások együttes kezelése

# Ennél azért már több is elérhető:

- Szövegalapú keresés -> tudásháló (knowledge graph)  
<https://www.google.com>
- Wikipedia -> DBPedia
- Profil hálók



# A szemantikus web irányában



“As We May Think”, 1945

Picture from <http://www.theatlantic.com/doc/194507/bush>

# Példa: tudás hálók

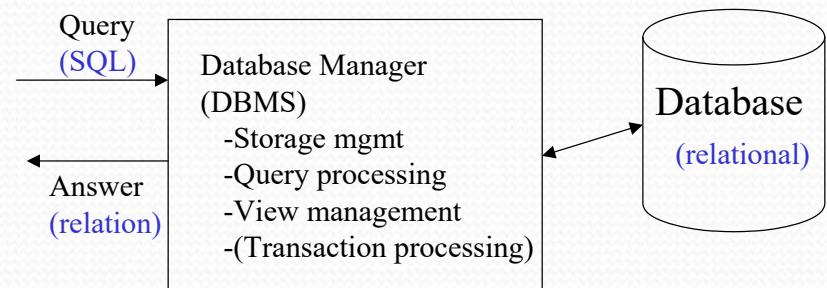
| Name                     | Instances   | Facts          | Types   | Relations |
|--------------------------|-------------|----------------|---------|-----------|
| DBpedia (English)        | 4,806,150   | 176,043,129    | 735     | 2,813     |
| YAGO                     | 4,595,906   | 25,946,870     | 488,469 | 77        |
| Freebase                 | 49,947,845  | 3,041,722,635  | 26,507  | 37,781    |
| Wikidata                 | 15,602,060  | 65,993,797     | 23,157  | 1,673     |
| NELL                     | 2,006,896   | 432,845        | 285     | 425       |
| OpenCyc                  | 118,499     | 2,413,894      | 45,153  | 18,526    |
| Google's Knowledge Graph | 570,000,000 | 18,000,000,000 | 1,500   | 35,000    |
| Google's Knowledge Vault | 45,000,000  | 271,000,000    | 1,100   | 4,469     |
| Yahoo! Knowledge Graph   | 3,443,743   | 1,391,054,990  | 250     | 800       |

Forrás: Knowledge Graph Refinement:A Survey of Approaches and Evaluation Methods, Editor(s): Philipp Cimiano, Universität Bielefeld, GermanySolicited review(s): Natasha Noy, Google Inc., USA; Philipp Cimiano, Universität Bielefeld, Germany; Semantic Web 0 (2016) 1–0 IOS Press

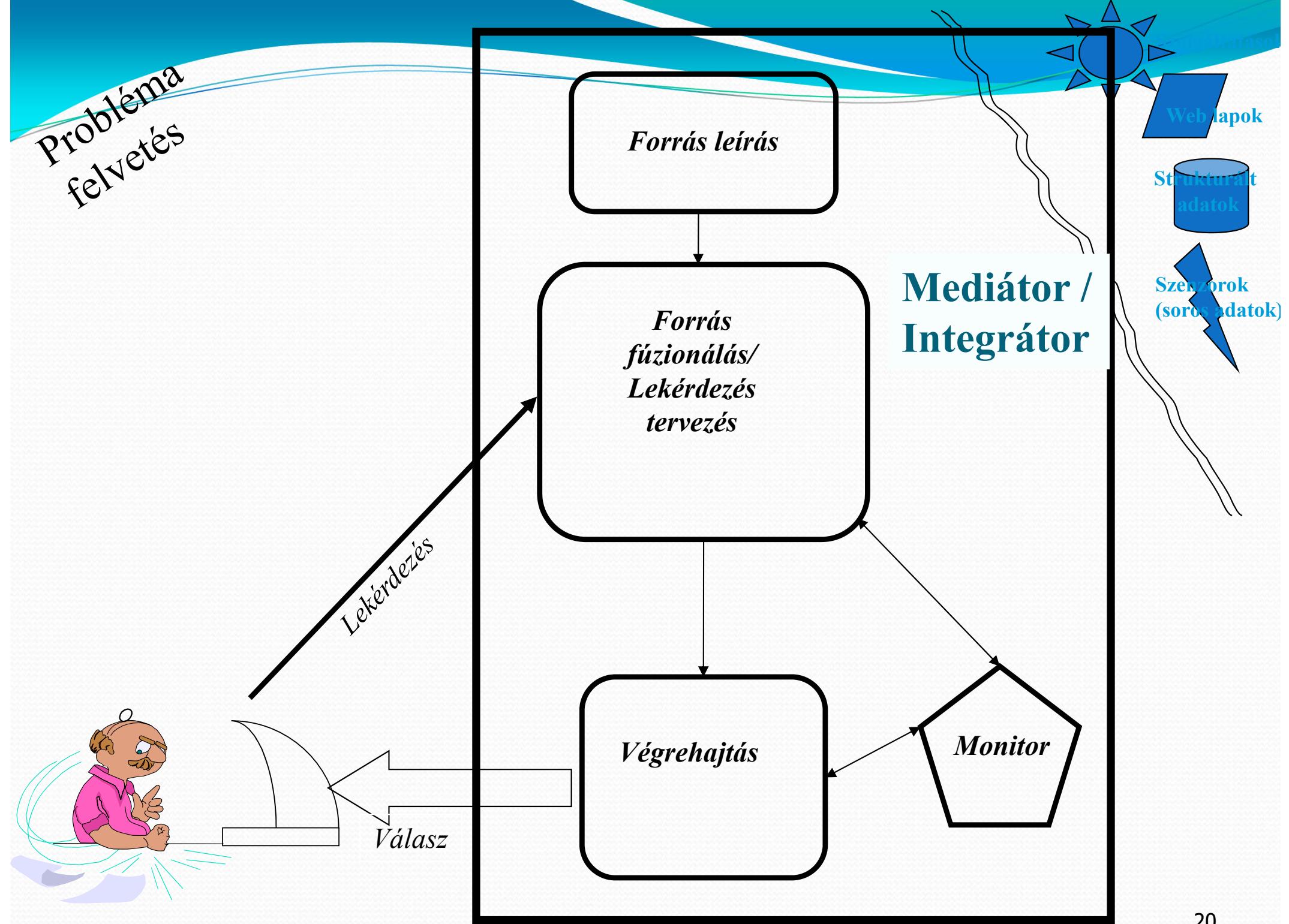
# Miért nem csak

adatbázisok  
elosztott adatbázisok

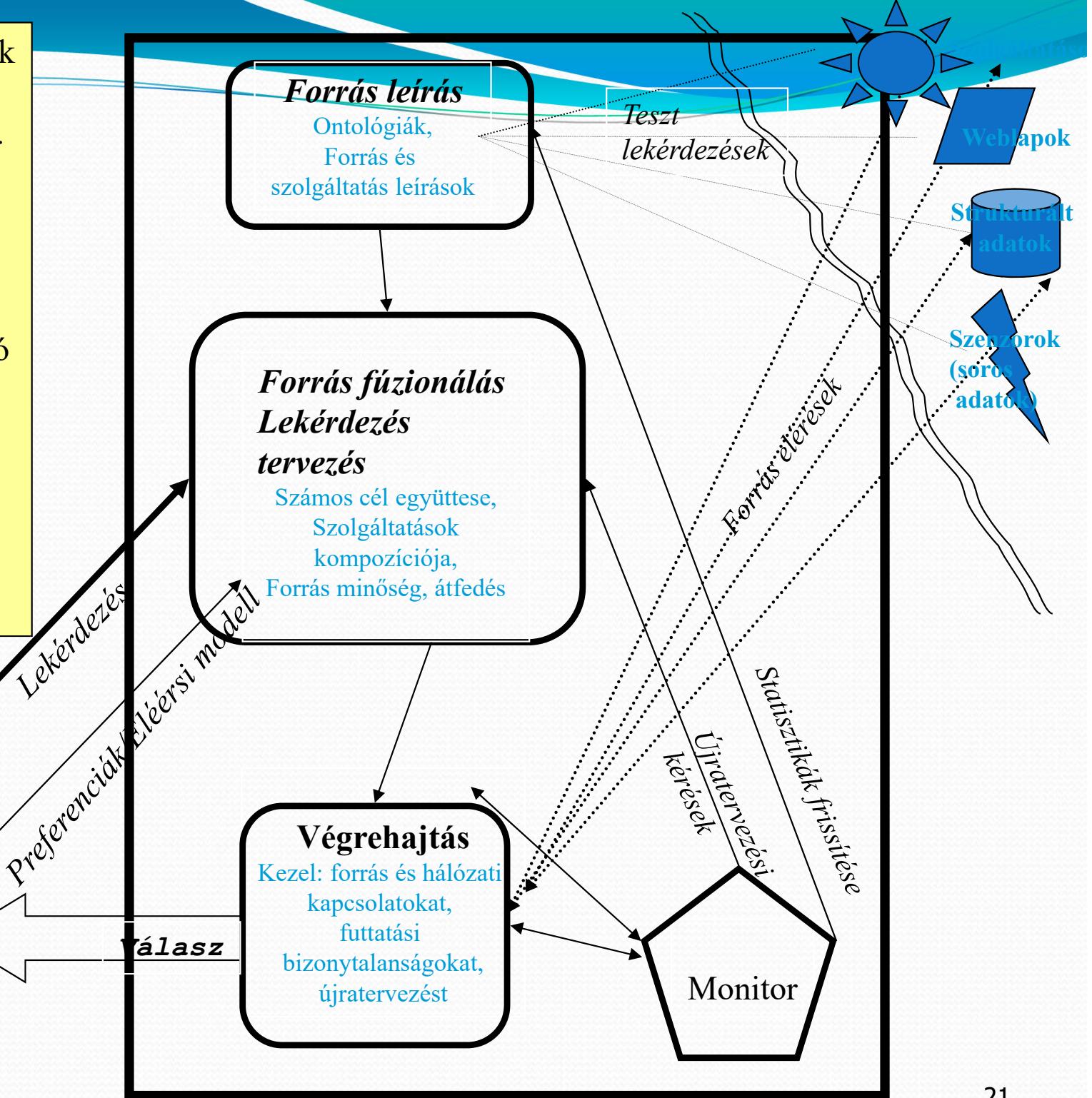
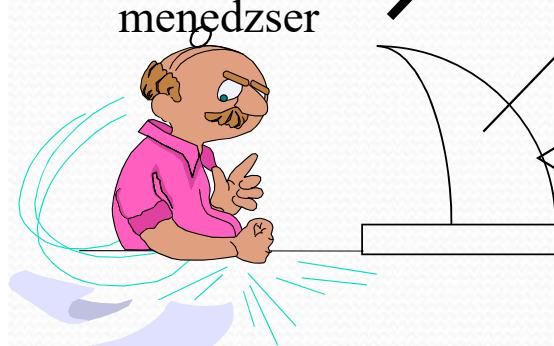
- **Közös séma hiánya**
  - Források heterogén sémákkal (és fogalmakkal, ontológiákkal) rendelkeznek
  - Félig-strukturált források
- **Régi források**
  - Nem relációs sémák
  - Eltérő elérési módok
- **Független források**
  - Nincs közös adminisztráció
  - Nem kezelt forrás tartalmi átfedések
- **Nehezen előrejelzhető viselkedés**
  - Lekérdezés végrehajtás bonyolult
- **Általában csak olvashatóak**
  - Ez lehet szerencsés is
  - Bár terjednek a tranzakció kezelési megoldások a weben



# Probléma felvetés



- Felhasználói lekérdezések megfogalmazása a mediált (*globális*) sémán.
- Adatok tárolva *lokális (távoli) sémákban*.
- A tárolt információ (tartalom) ismerete alapján megfogalmazható a leképezés a sémák között.
- A mediátor alkalmazza a leképezést a felhasználói kérdés lefordítására a forrás lekérdezésekre.



# Mesterséges intelligencia

## Tanulás/bányászás

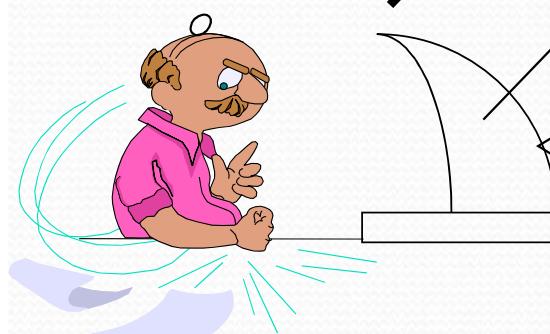
- Forrás felkutatás
- Forrás statisztikák
- Wrapper tanulás

## Tudásreprezentáció

- Ontológiák
- Metaadatok
- Következtetés
- Lekérdező nyelvek

## Automata tervezés

- Nyelvek tervezése
- Szolgáltatások kompozíciója
- Reaktív tervezés/  
terv monitorozás



### Forrás leírás

Ontológiák,  
Forrás és  
szolgáltatás leírások

### Forrás fúzionálás Lekérdezés tervezés

Számos cél együttese,  
Szolgáltatások  
kompozíciója,  
Forrás minőség, átfedés

### Végrehajtás

Kezel: forrás és hálózati  
kapcsolatokat,  
futtatási  
bizonytalanságokat,  
újratervezést

Teszt  
lekérdezések

Webapok

Strukturált  
adatok

Szenzorok  
(soros  
adatok)

Forrás elérések

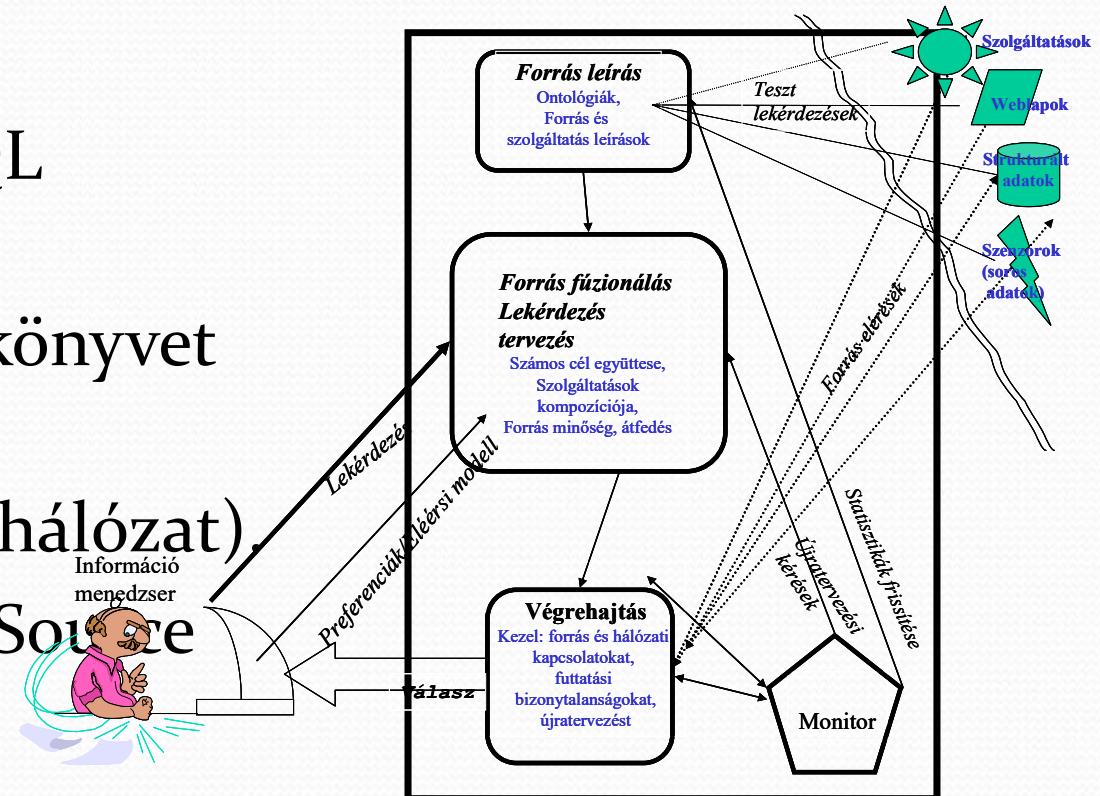
Újratervezési  
kérések  
Statisztikák frissítése

Monitor

Lekérdezés  
Elérési modell  
Preferenciák  
Válasz

# Forrás leírások

- minden meta-adat információt tartalmaz
  - Forrás tartalom logikai leírása (könyvek, új autók).
  - Forrás képességek (pl. SQL lekérdezés feltehető)
  - Forrás teljesség (*minden* könyvet tartalmaz).
  - Fizikai jellemzők (forrás, hálózat)
  - Statisztikák az adatokról Source reliability
  - Tükör források
  - Frissítési frekvencia.

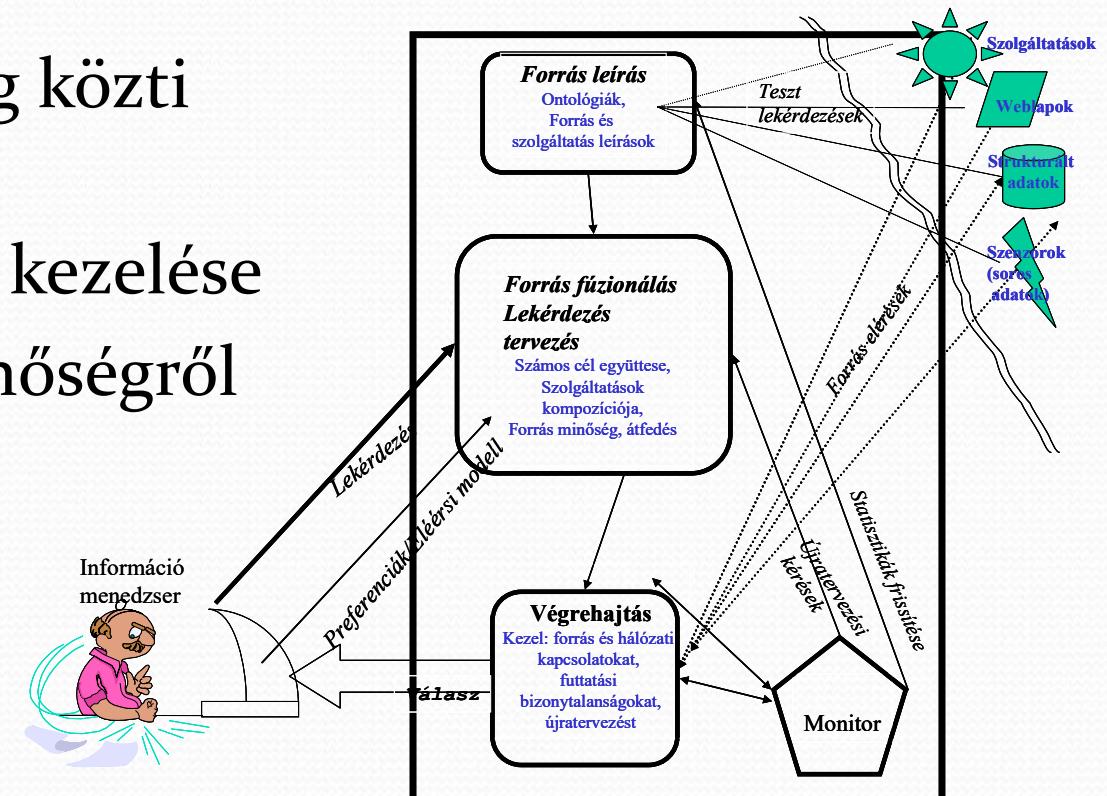


# Forrás elérések

- Hogyan kapunk n-eseket
  - Számos forrás strukturálatlan adatokat ad
    - Néhány inherensen strukturálatlan, mások természetes nyelvi köntösben vannak
  - Vissza kell csomagolni az adatokat
  - Wrapper építés/információ kinyerés
  - Kézi munka/fél-automatikus

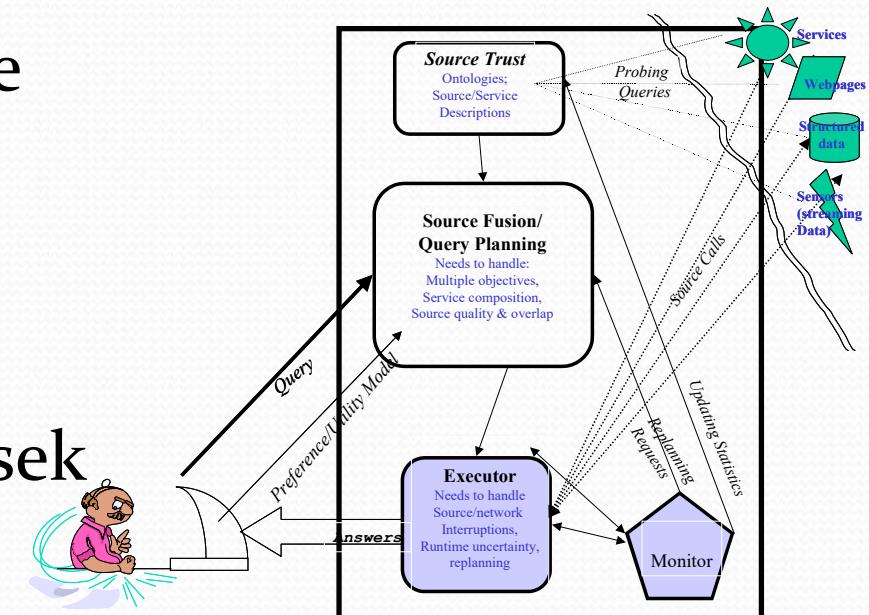
# Forrás fúzió/ lekérdezés tervezés

- Feldolgozza a felhasználói lekérdezést és előállítja a végrehajtási tervet
  - Költség és hatékonyság közti optimalizáció
  - Forrás elérési korlátok kezelése
  - Információ a forrásmínőségről



# Monitoring/ Végrehajtás

- Lekérdezési terv alapján elvégzi a feladatot a forrásokon
  - Forrás késleltetések kezelése
  - Hálózati, tranziens kimaradások
  - Forrás elérési korlátok
  - Szükséges lehet újratervezések elvégzése

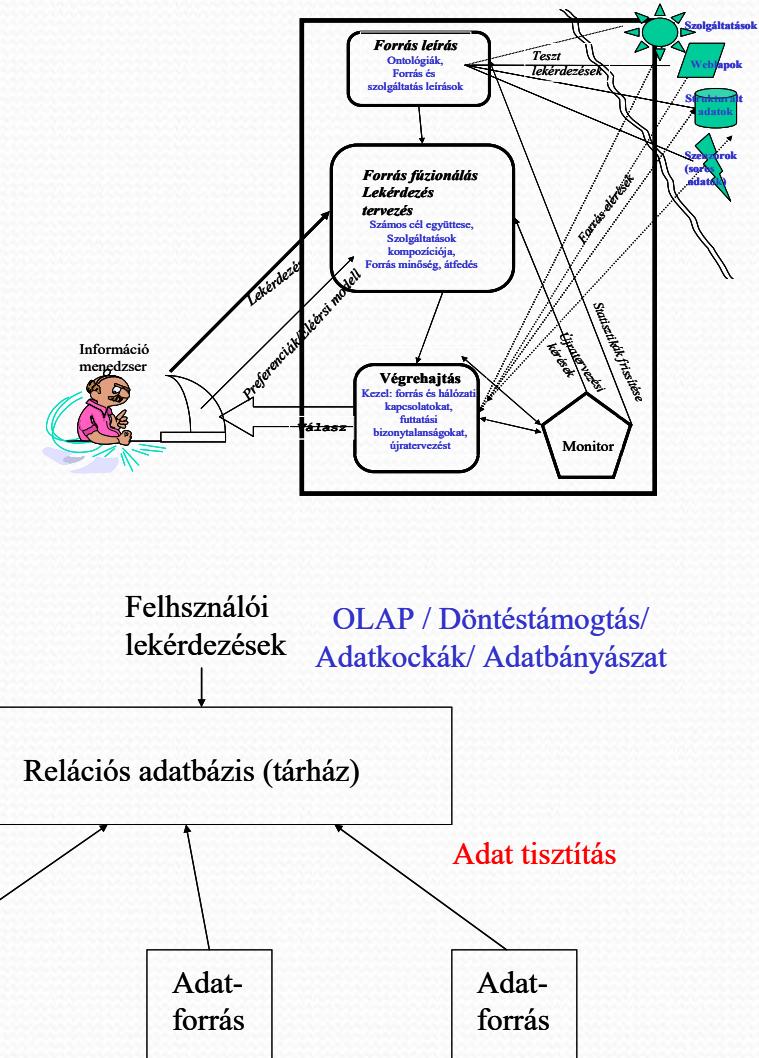


# Algoritmus optimalizálása

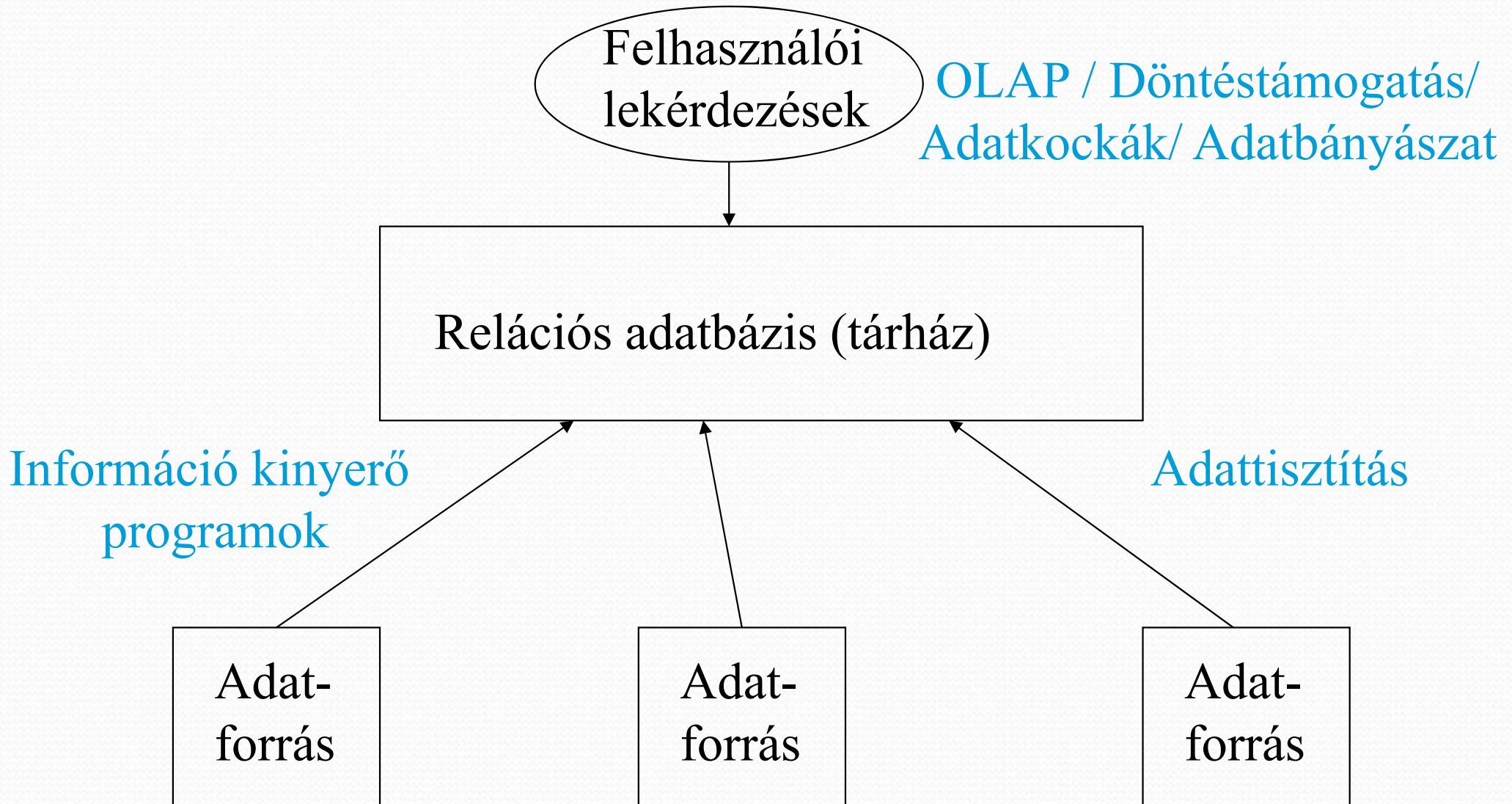
- Hány forrást kell elérni?
- Mennyire autonómok ezek?
- Van ismeretünk a forrásokról?
- Strukturáltak az adatok?
- Csak lekérdezés lehetséges vagy módosítás is?
- Követelmények: pontosság, teljesség, teljesítmény, inkonzisztenciák kezelése
- Zárt vagy nyílt világ feltételezés?

# Kis forrás szám melletti integráció

- **Általában ad-hoc programozás:** speciális eset megvalósítása minden esetre, sok konzultáció.
- **Adattárházak:** minden adat periodikus feltöltése az adattárházba.
  - 6-18 hónap bevezetési idő
  - Operációs és döntéstámogatási RDBMS elválasztás. (nem csak adatintegrációra megoldás).
  - Teljesítmény jó,
  - adat lehet, hogy nem friss;
  - Rendszeres adattisztítás szükséges.

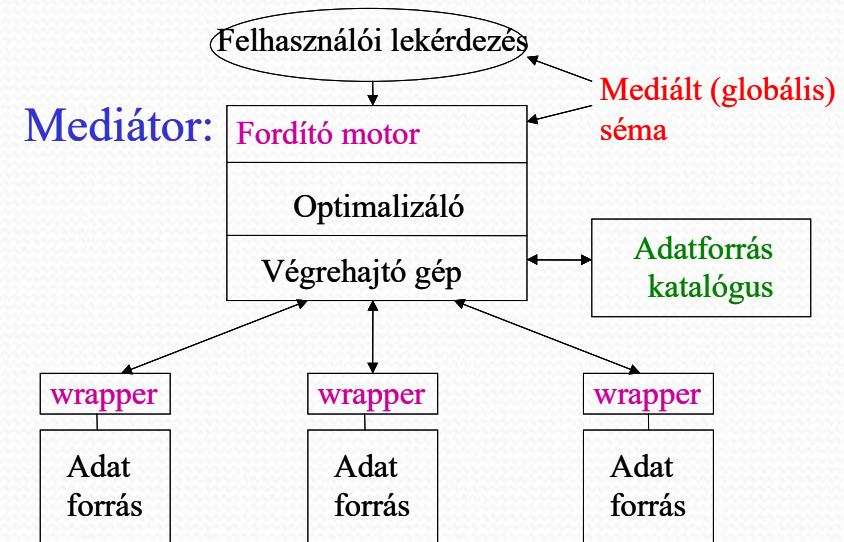
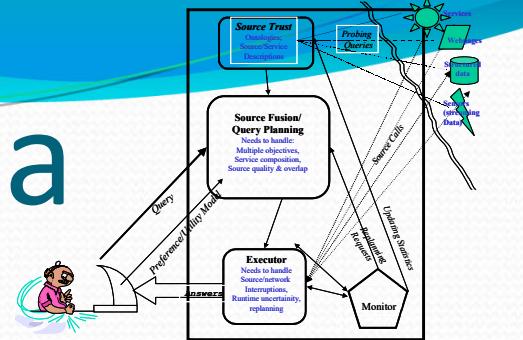


# Integrátor séma



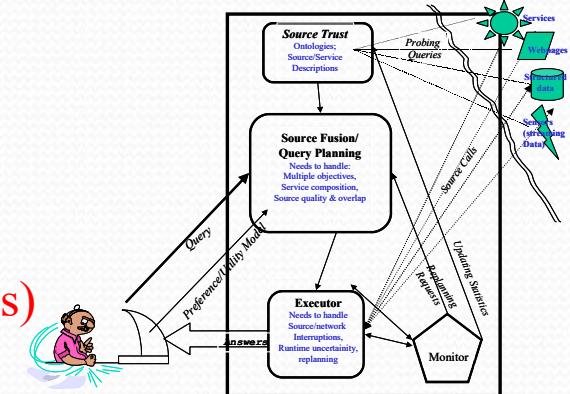
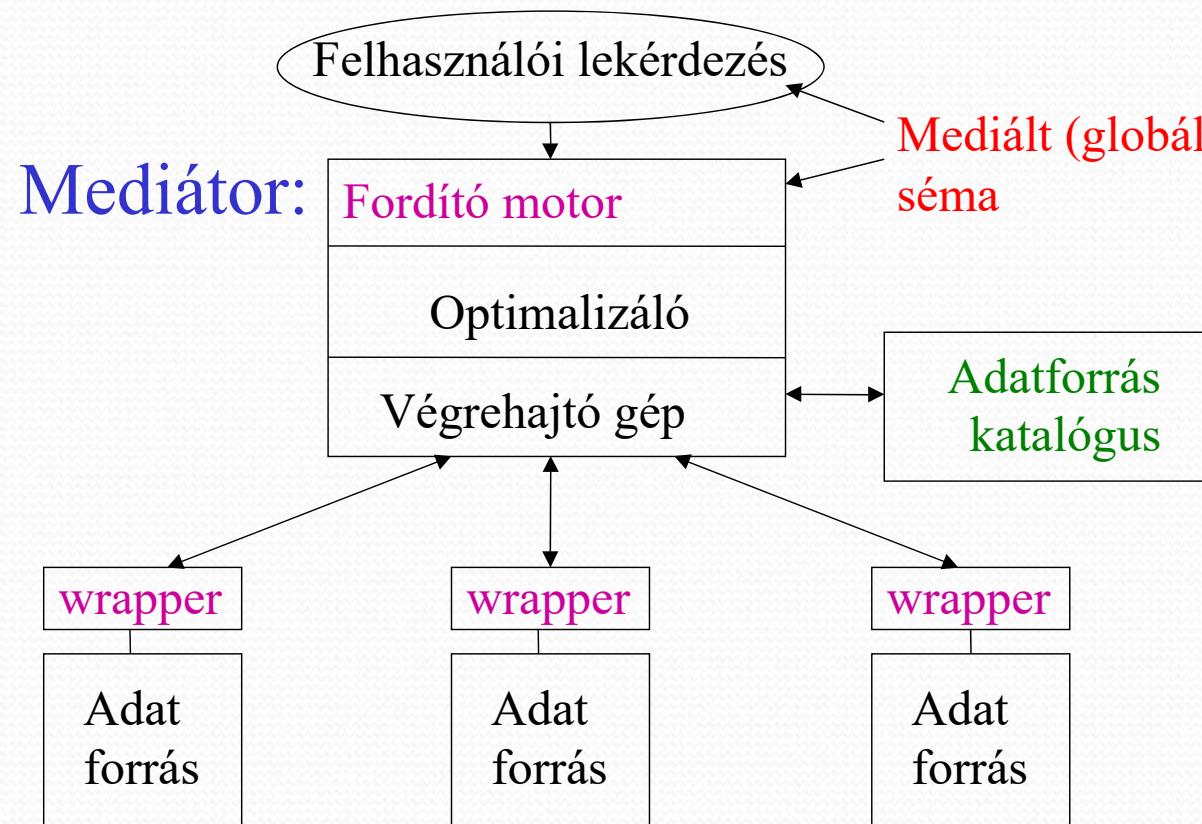
# Virtuális integrációs séma

- Adatok a forrásokban maradnak
- Lekérdezés végrehajtásakor:
  - Releváns források meghatározása
  - Lekérdezés szétválasztása forrásokra vonatkozó lekérdezésekre.
  - Válaszok begyűjtése a forrásokból, és megfelelő kombinálása a válasz előállításához.
- Friss adatok
- A megoldás skálázható



Garlic [IBM], Hermes[UMD]; Tsimmis, InfoMaster[Stanford]; DISCO[INRIA]; Information Manifold [AT&T]; SIMS/Ariadne[USC]; Emerac/Havasu[ASU]

# Virtuális integrátor architektúra



**Források:** relációs adatbázisok, weblapok, szövegek.

# Az Archicad tesztelése

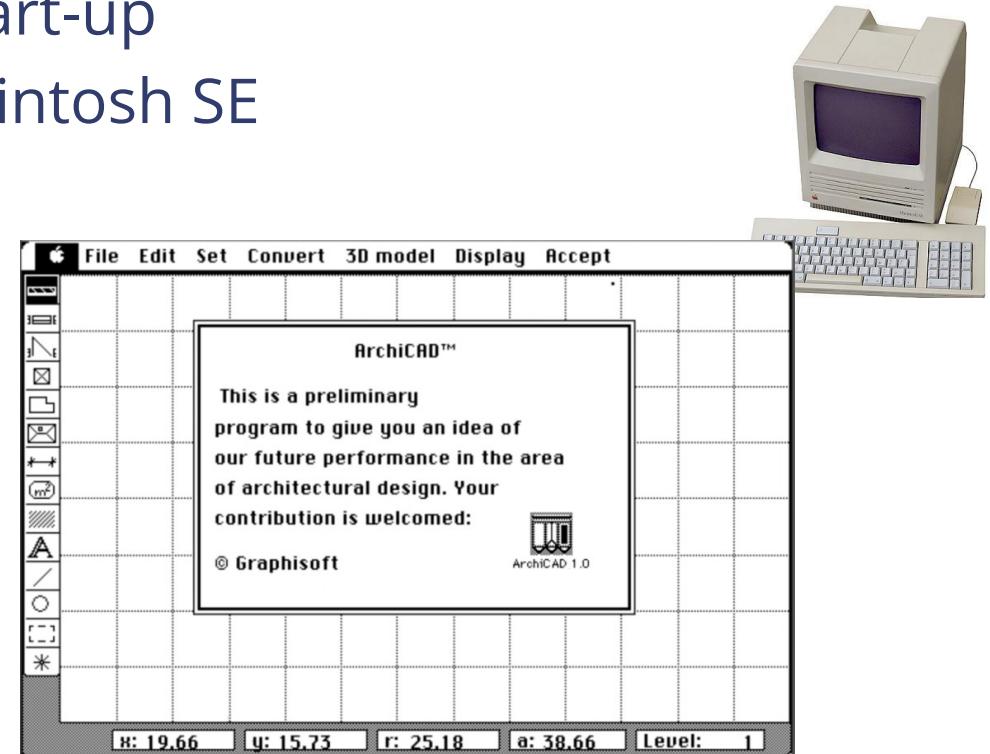
**Előadók:** Árvai László  
Batári Dorottya  
Székely Zoltán

# Agenda

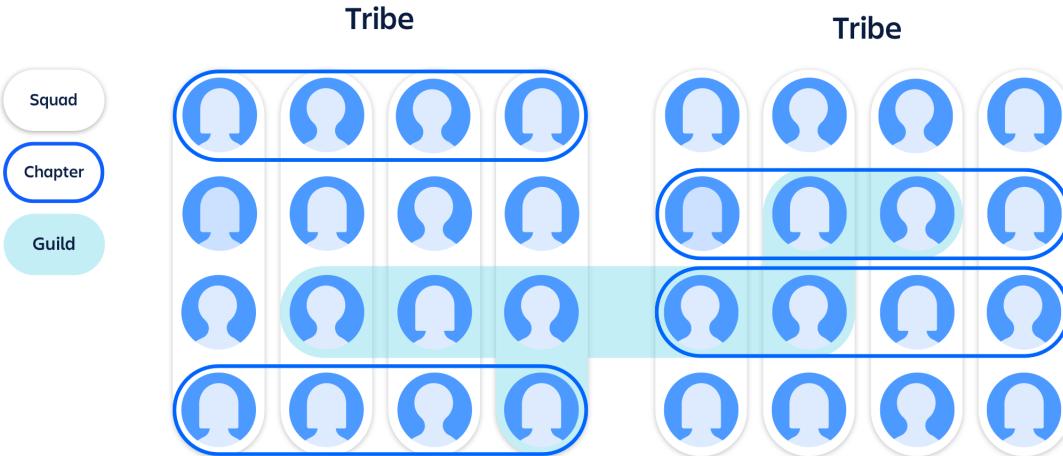
1. Cég bemutatása, csapatok felépítése
2. Tesztfolyamat bemutatása
3. A termék - Archicad
4. Manuális tesztelés demo
5. Autoteszt rendszer felépítése
6. Autoteszt futás demo

# Cég, csapatok

- 1982-ben alapítva, első magyar start-up
- első kiadott verzió AC2, 1986, Macintosh SE
- aktuális kiadott verzió: AC24
- HQ: Graphisoft park
- APAC, Európa, Amerika
- jelenleg ~700 kolléga
- fejlesztés ~ 300 kolléga



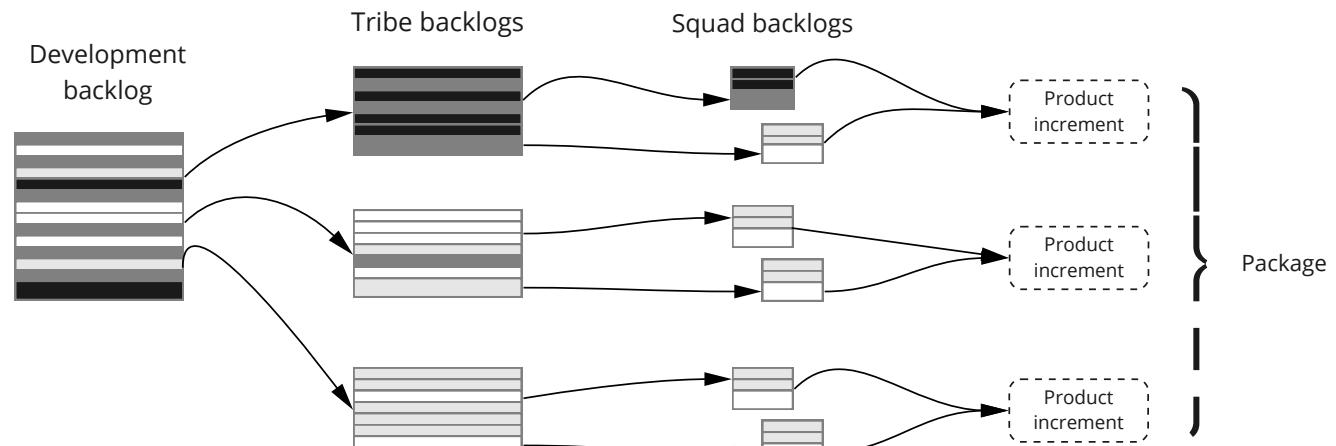
- Spotify-modellen alapuló szervezet



- crossfunctional csapatok
- külön fejlesztési ágak: squad, tribe, főág
- QA guild szinten összehangolás, szakmai beszélgetések, workgroupok, konferenciák

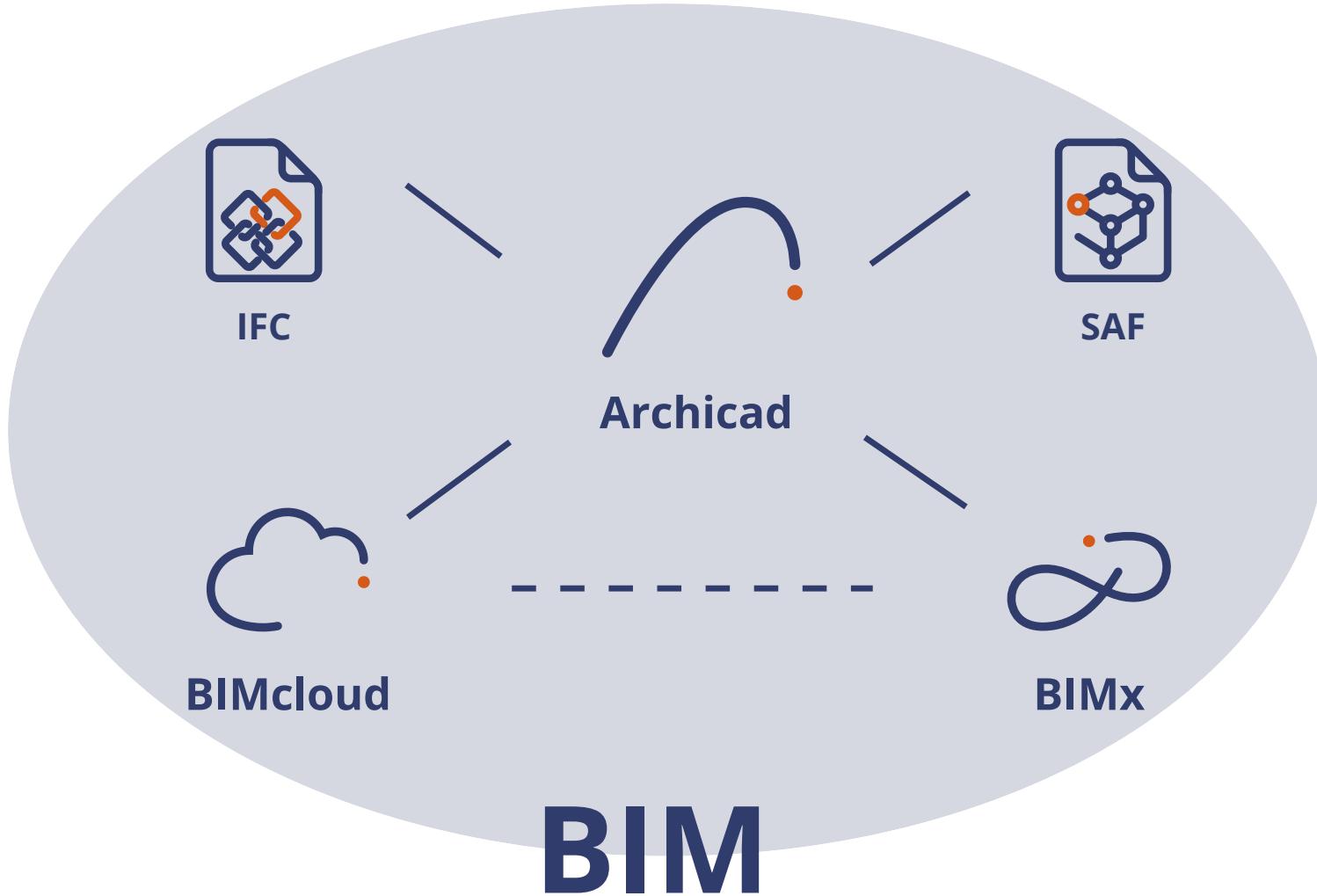
# Tesztfolyamat

- "shift left" elv
- párhuzamosítás
- követelmények tisztázása, tesztek tervezése
- csapatszintű fejlesztések, hibajavítások tesztelése
  - fejlesztések összeérésének tesztelése, integrációs teszt
  - funkcionális tesztek, workflow, exploratory
  - acceptance criteria teljesülésének validációja
  - manuális vagy automata tesztek



- **in-house tesztelés**
  - valós user szemléletben
- **béta tesztelés**
  - partnercégek, végfelhasználók
- **végtesztelés release előtt**
  - végleges termékcsomag
  - installer, modulok, dokumentáció
  - lokalizációs tesztelés
- **nem-funkcionális tesztelés**
  - saját fejlesztésű rendszerben, webes reportfelületen
  - sebesség, stabilitás
  - MTBF

# Termékek



## Archicad

- fő termék
- CAD szoftver
- 3D modellezés

## BIMcloud

- projekten dolgozó mérnökök együttműködését segíti
- szerver (?)

## BIMx

- projekt vizualizációja
- applikáció

# Archicad

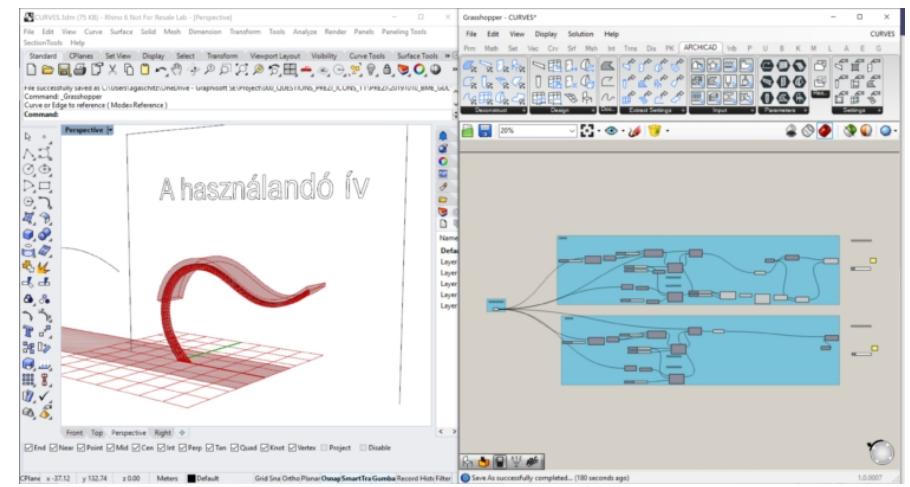
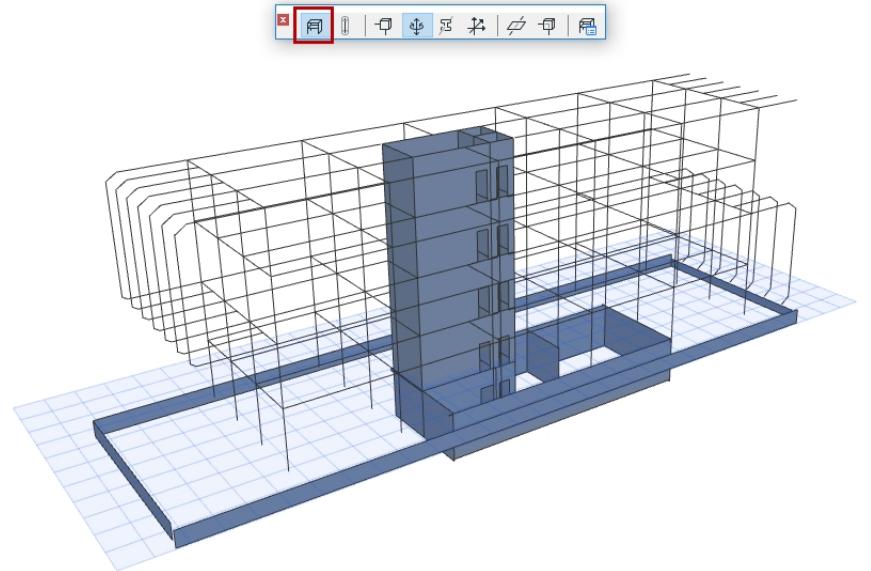


- CAD - Computer-Aided Design, számítógép által támogatott tervezés
- **építész** tervezés segítése
- **3D modell**
  - modellépítés, szerkesztés 3D-ben
  - végtermék is lehet 3D
    - 3D-s fájlformátumok
    - BIMx
  - a programmal létrehozhatóak 2D-s tervrajzok is

# Archicad



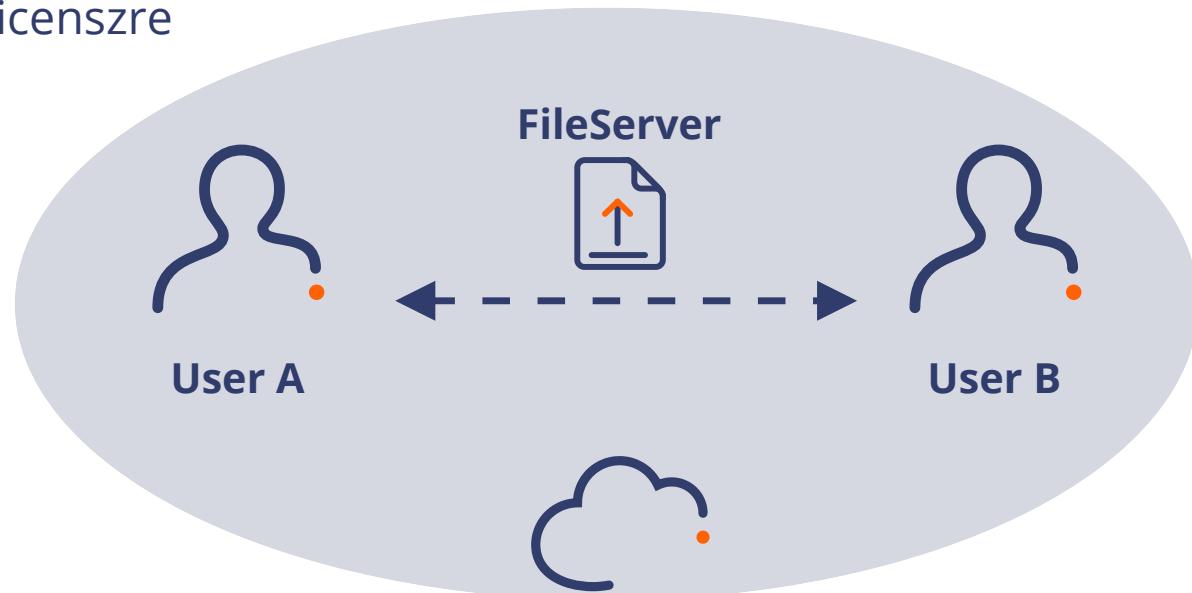
- együttműködés, kompatibilitás:
  - éves kiadási ciklus - kompatibilitás a **verziók között**
  - **cégen belüli** termékekkel - BIMx, BIMcloud
  - különböző **szakági szoftverekkel**
    - kompatibilis fájlformátum
    - pl: gépész, statikus szoftverek (SAF)
  - együttműködés **egyéb CAD szoftverekkel**
    - élő kapcsolat
    - pl: Rhino-Grasshopper



# BIMcloud

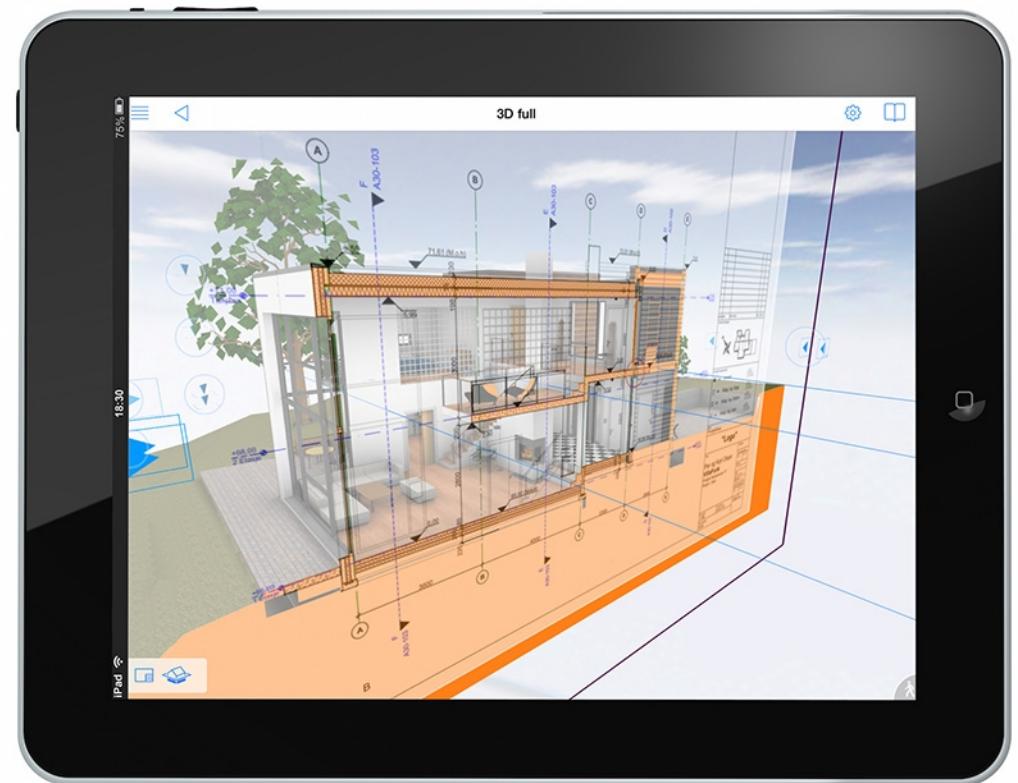


- Teamwork funkció: egy Archicad-projekten több user dolgozhasson egy időben
- a BIMcloud a Teamwork funkció kiszolgálását biztosítja
- hardver kiszolgálás: user vagy Graphisoft (SaaS)
- FileServer: dokumentációk, projektfájlok tárolása (különböző szakágak, nincs szükség Archicad licenszre)





- Archicaddel készített projektfájlok megosztása, megtekintése
- lehetőségek:
  - 3D modell "bejárása"
  - integrált 2D tervlapok megtekintése
  - elemekhez tartozó információk megtekintése
  - mobil (iOS, Android), desktop és webes alkalmazás
- <https://bimx.graphisoft.com/>



## Steps

| # | Action                                        | Input                                                                                                                                             | Expected result                                                                                                                                                                                                             | Status      |     |
|---|-----------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|-----|
| 1 | Pattern megjelenés                            | <ul style="list-style-type: none"> <li>▪ belépés edit módba</li> <li>▪ pattern megjelenés ki/be kapcsolása</li> </ul>                             | <ul style="list-style-type: none"> <li>▪ on/off működik</li> <li>▪ piros keret</li> <li>▪ keret széle=minta széle</li> <li>▪ egy keret egy curtain wall-ban</li> <li>▪ csak edit módban</li> </ul>                          | In progress | ... |
| 2 | Patternméret változtatás                      | <ul style="list-style-type: none"> <li>▪ Pattern szélénél elmozgatása</li> </ul>                                                                  | <ul style="list-style-type: none"> <li>▪ feedback</li> <li>▪ frissül a CW edit módban</li> <li>▪ frissül a CW setting dialogja</li> <li>▪ Edit módból kilépve megmarad a beállítás</li> </ul>                               | Pass        | ... |
| 3 | Pattern kihúzás CW-n kívülre                  | <ul style="list-style-type: none"> <li>▪ Pattern szélénél elmozgatása a CW-n kívülre</li> </ul>                                                   | <ul style="list-style-type: none"> <li>▪ feedback</li> <li>▪ CW-n nincs változás</li> <li>▪ CW setting dialogban Deleted Panel jelenik meg</li> <li>▪ Edit módból kilépve is megmarad a Deleted Panel a mintában</li> </ul> | Fail        | ... |
| 4 | Pattern mozgatás                              | <ul style="list-style-type: none"> <li>▪ Patternbox elmozgatás</li> </ul>                                                                         | <ul style="list-style-type: none"> <li>▪ feedback</li> <li>▪ frissül a CW edit módban</li> <li>▪ frissül a CW setting dialogja</li> <li>▪ Edit módból kilépve megmarad a beállítás</li> </ul>                               | To do       | ... |
| 5 | Undo/Redo                                     | <ul style="list-style-type: none"> <li>▪ Undo parancs</li> <li>▪ Redo parancs</li> </ul>                                                          | <ul style="list-style-type: none"> <li>▪ Undo parancsra megtörténik a visszavonás</li> <li>▪ Redo parancsra mégis végrehajtódik a módosítás</li> </ul>                                                                      | To do       | ... |
| 6 | Edit módon kívüli változtatás hat a patternre | <ul style="list-style-type: none"> <li>▪ CW settingsben változtatunk a mintán</li> <li>▪ belépünk Edit módba és bekapcsoljuk a Pattern</li> </ul> | <ul style="list-style-type: none"> <li>▪ Edit módban jó helyen legyen a patternbox</li> </ul>                                                                                                                               | To do       | ... |
| 7 | ...                                           |                                                                                                                                                   |                                                                                                                                                                                                                             | To do       | ... |
| 8 | ...                                           |                                                                                                                                                   |                                                                                                                                                                                                                             | To do       | ... |
| 9 | ...                                           |                                                                                                                                                   |                                                                                                                                                                                                                             | To do       | ... |

[Export to CSV](#)[Edit](#)

# Autotesztrendszer

- saját fejlesztésű autotesztrendszer
- saját eszközöket használunk az autotesztjeink futtatására (Autotest Manager) és kiértékelésére (Result Processor) egyaránt -> ezek napi szinten több branch-re is lefutnak
- szorosan kapcsolódik a perforce-hoz, ami a jelenleg használt verziókezelő software-ünk
- jelenleg kb 2500 tesztünk fut ebben a rendszerben, windowson és macen is (funkcionális, performance tesztek)
- ezen kívül fejlesztői (unit) teszteket is tartalmaz a teljes autoteszt készletünk (GSAT)

# Autotesztrendszer

Autotest Manager:

- platform független desktop applikáció (egyelőre még)
- futtatásra és tesztek debug-olására is alkalmazható
- ~140 tesztgéppel rendelkező gépparkot használunk a futásainkhoz
  - különböző hardware, software és OS verziókkal
- saját gépeken is van lehetőség a futtatásra
- debug esetén bármely ponton meg tudjuk nézni az aktuális állapotot
  - tesztgépekre VNC-zve belenyúlhatunk a teszt bármely részébe

# Autotesztrendszer

Framework:

- saját fejlesztésű FW-öt használunk
- kattintásokhoz nem direktben képernyőkoordinátákat használunk
  - kihasználjuk, hogy CAD software-t fejlesztünk
  - belső koordináta rendszerben adjuk meg a helyes pontokat, amiket az AC visszaszámolja képernyő koordinátává
- dialógokon elhelyezkedő gombokra GUI map-et tartunk karban
  - ha a dialógon megtalálható és a GUI rendben van, akkor bárhol meg tudjuk kattintani
- interakciók végrehajtására, dialógok megnyitására command-okat használunk

# Autotesztrendszer

Autotesztek:

- 2 fő típusba tartozó autotesztet tervezünk
  - workflow (lásd videó)
  - data-driven: külső, változó paramétereket tartalmazó vezérlő fájllal meghajtott, ciklikusan futó teszttípus
- a tesztterveket PERL-ben implementáljuk
  - szükség esetén a Test Automation Team segítségével
- az agilis átállás óta igyekszünk minden folyamatunkat a saját csapatunkon (squad) belül elvégezni a tervezéstől, az implementáláson át a referencia update-ekig

# Autotesztrendszer

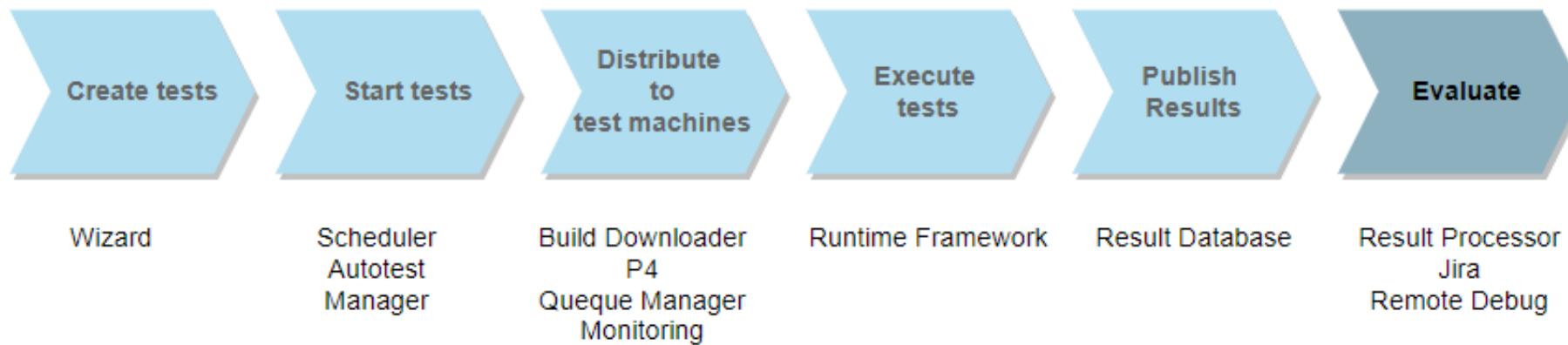
Kiértékelés:

- futás közben hasonlítja össze a rendszer a háttérben az aktuális build-en futott tesztek eredményeit a perforce-ban lévő utolsó referenciával
  - amennyiben megegyezik, akkor eldobjuk az "actual" checkpoint-ot
  - eltérés esetén megtartjuk és készítünk egy "diff"-et is
  - ekkor dönthetünk arról, hogy elfogadjuk-e a változásokat vagy hibát fogott a tesztünk



# Autotesztrendszer

- a különböző folyamatokhoz különböző tool-ok tartoznak



**Köszönjük a figyelmet!**

**Q&A**

# Tesztelési szintek Tesztautomatizálás

Majzik István, Micskei Zoltán

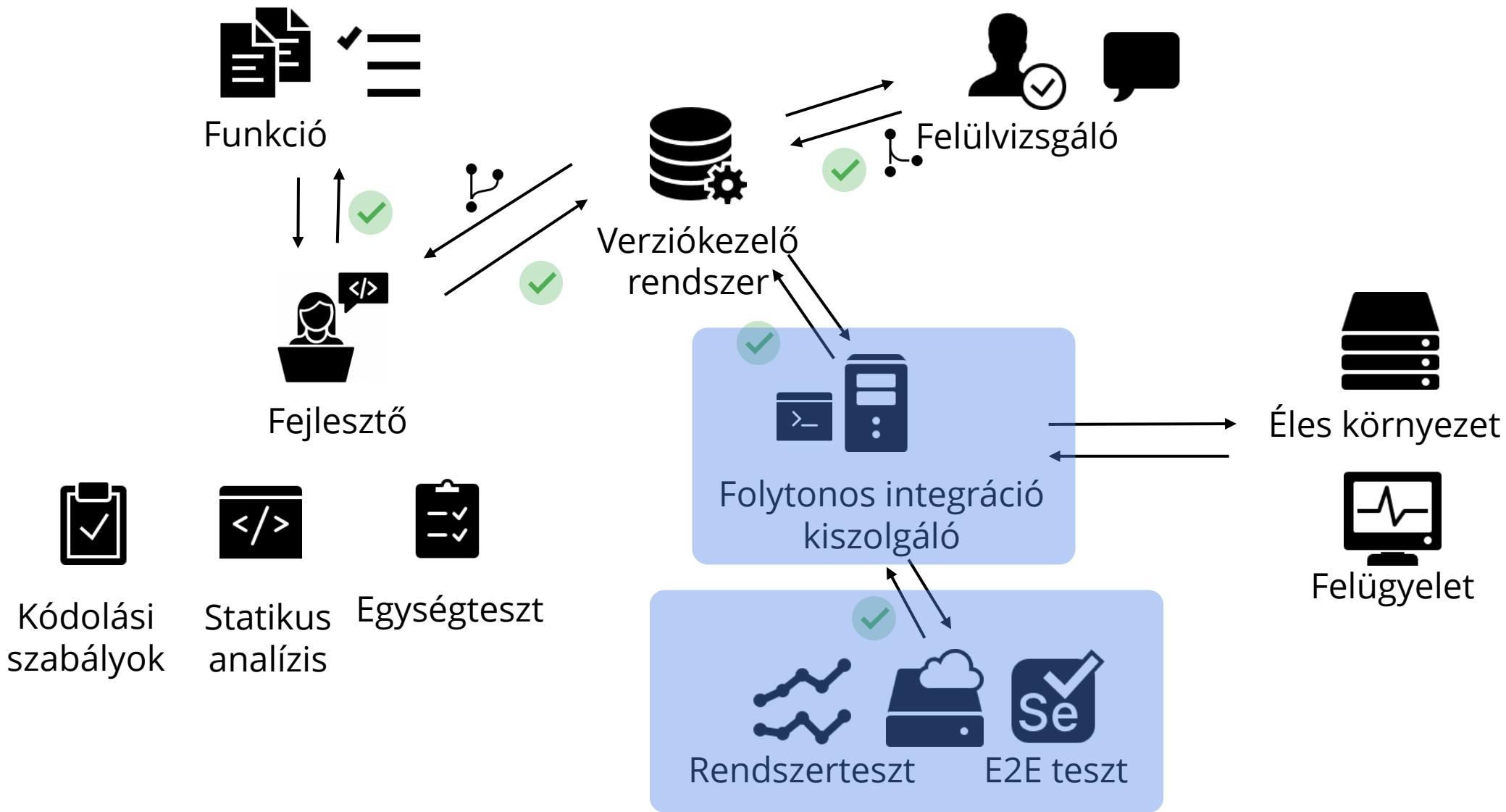


Méréstechnika és  
Információs Rendszerek  
Tanszék



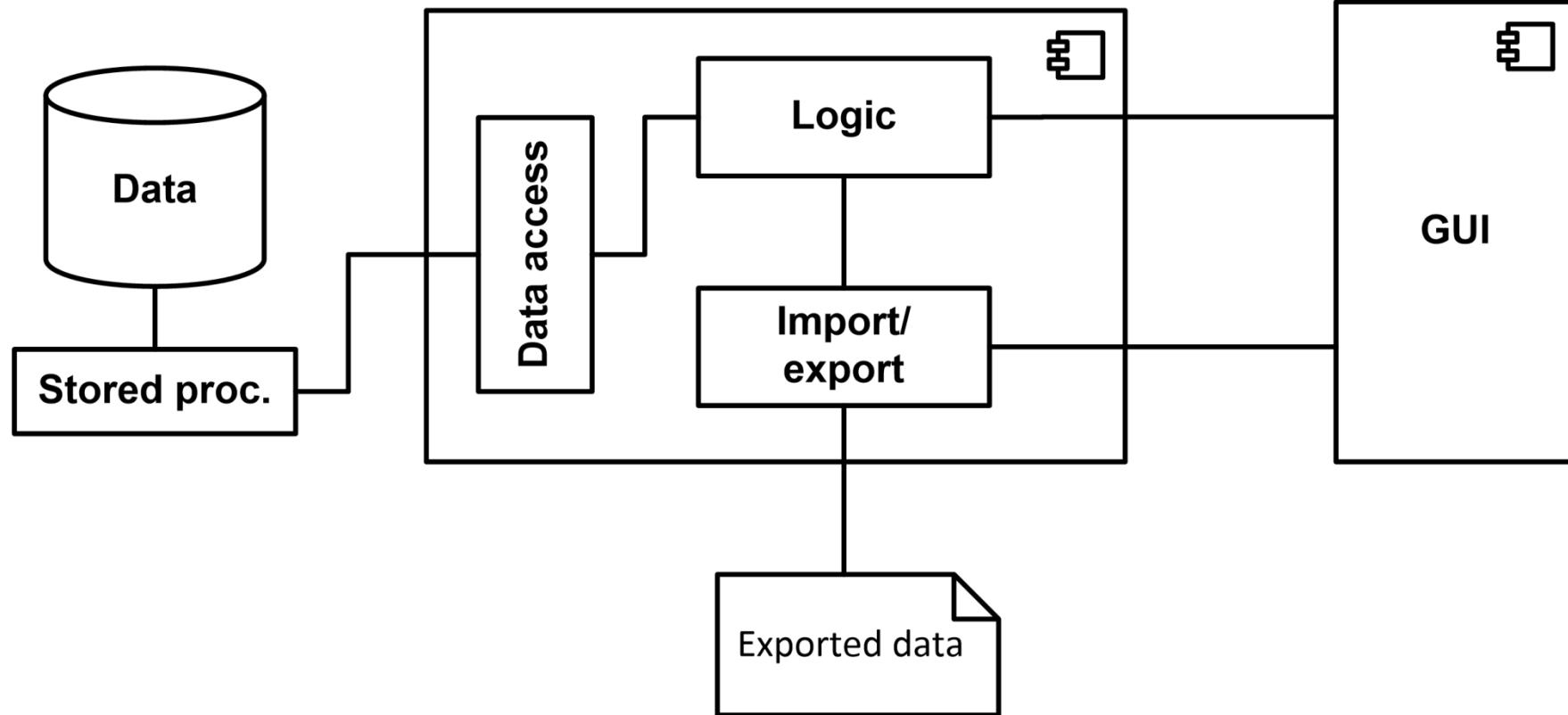
Critical Systems  
Research Group

# Tantárgy tematikája - Áttekintés

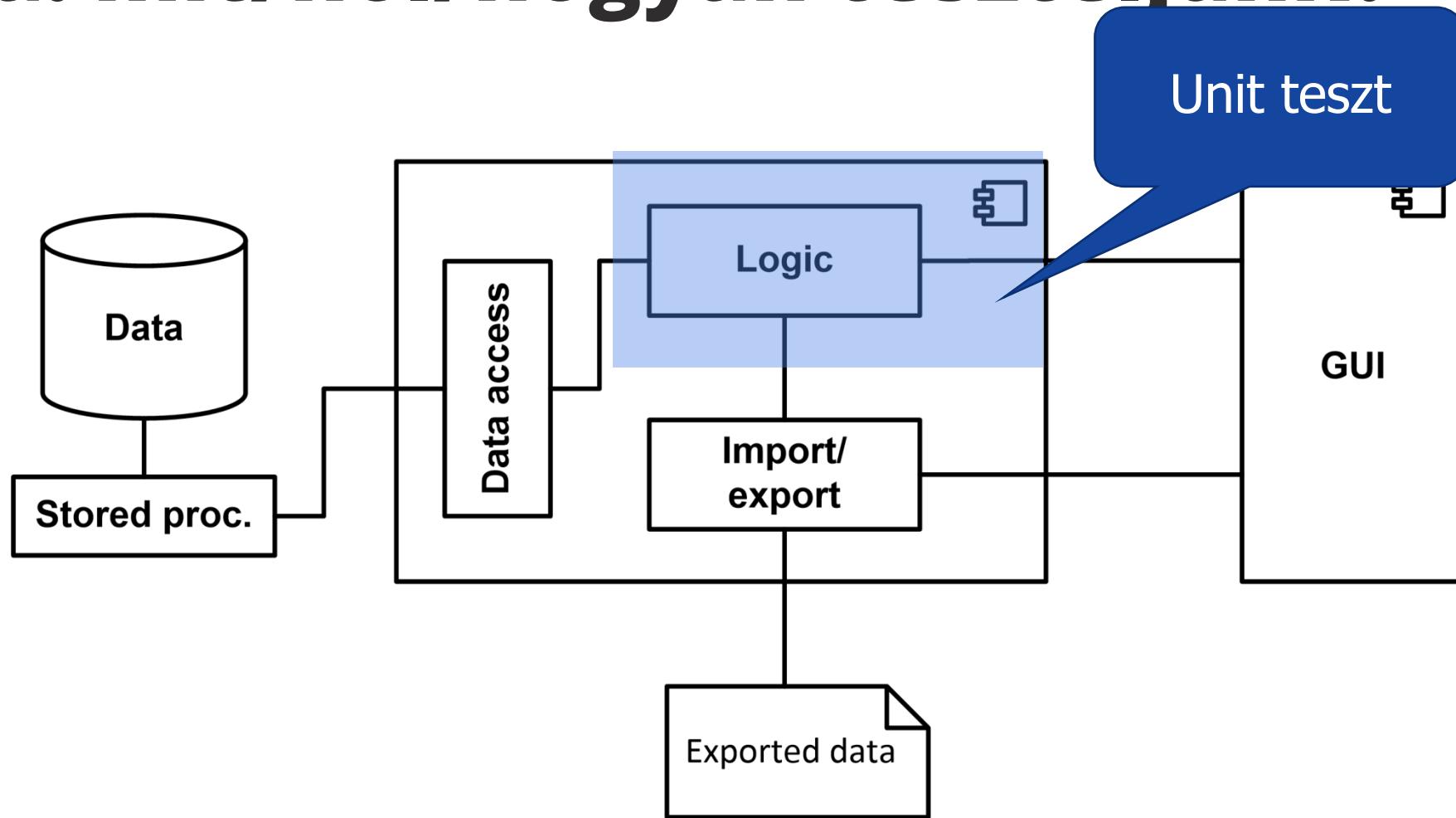


Ikonok: icons8.com

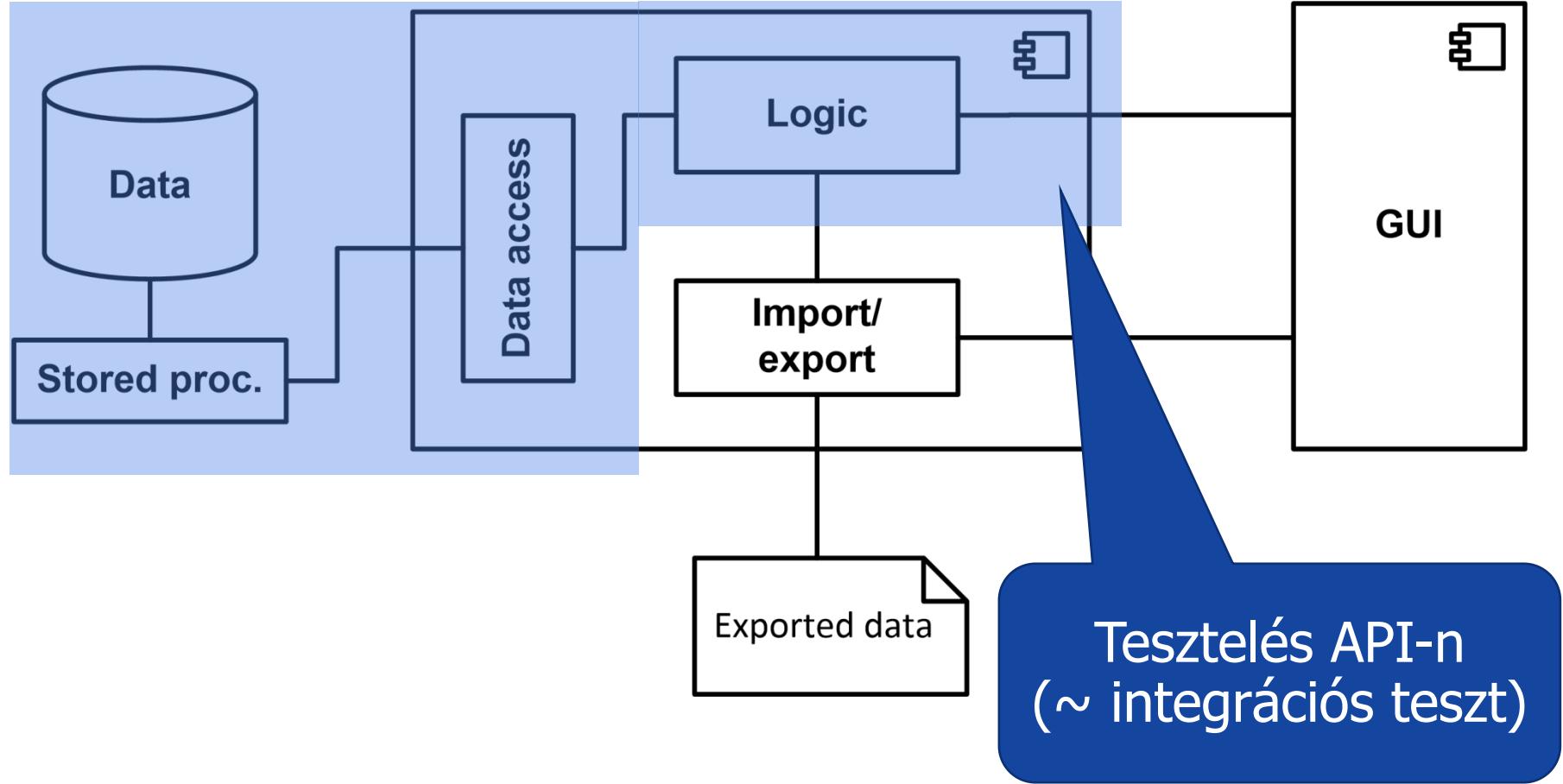
# Példa: mit/hol/hogyan teszteljünk?



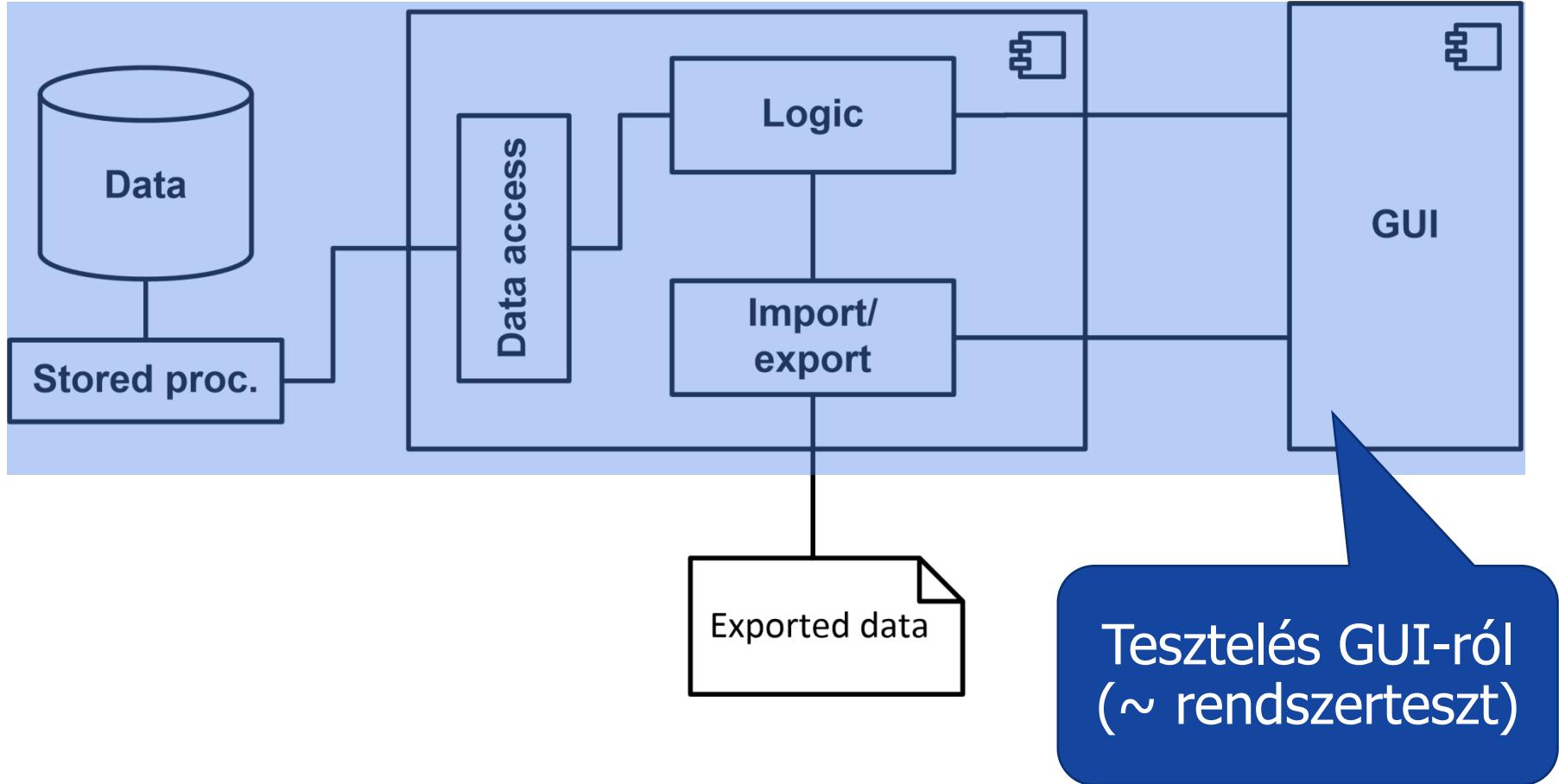
# Példa: mit/hol/hogyan teszteljünk?



# Példa: mit/hol/hogyan teszteljünk?

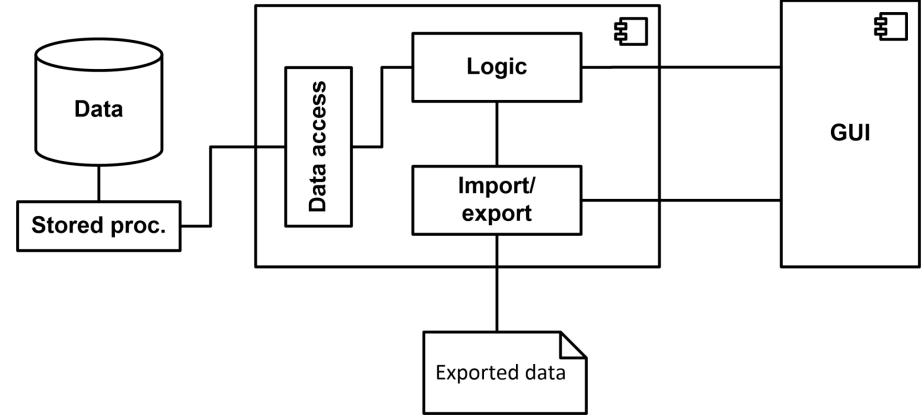


# Példa: mit/hol/hogyan teszteljünk?



# Tesztelési szintek

Integrációs, rendszer, elfogadási



# I. Integrációs tesztelés



Modulok **együttműködésének** ellenőrzése

- **Motiváció:**

- A rendszer annak ellenére hibás lehet,  
hogy minden modul egyenként hibátlan!

- **Módszerek:**

- Funkcionális tesztelés: **forgatókönyvek** tesztje
    - Ez sokszor a specifikáció része (scenario)
    - (Strukturális tesztelés inkább csak modulszinten!)

- **Megközelítések:**

- “**Big bang**”: minden modult egyszerre integrálni
  - **Inkrementális**: egyenként összerakni a modulokat

# II. Rendszertesztelés

## Teljes rendszer tesztelése

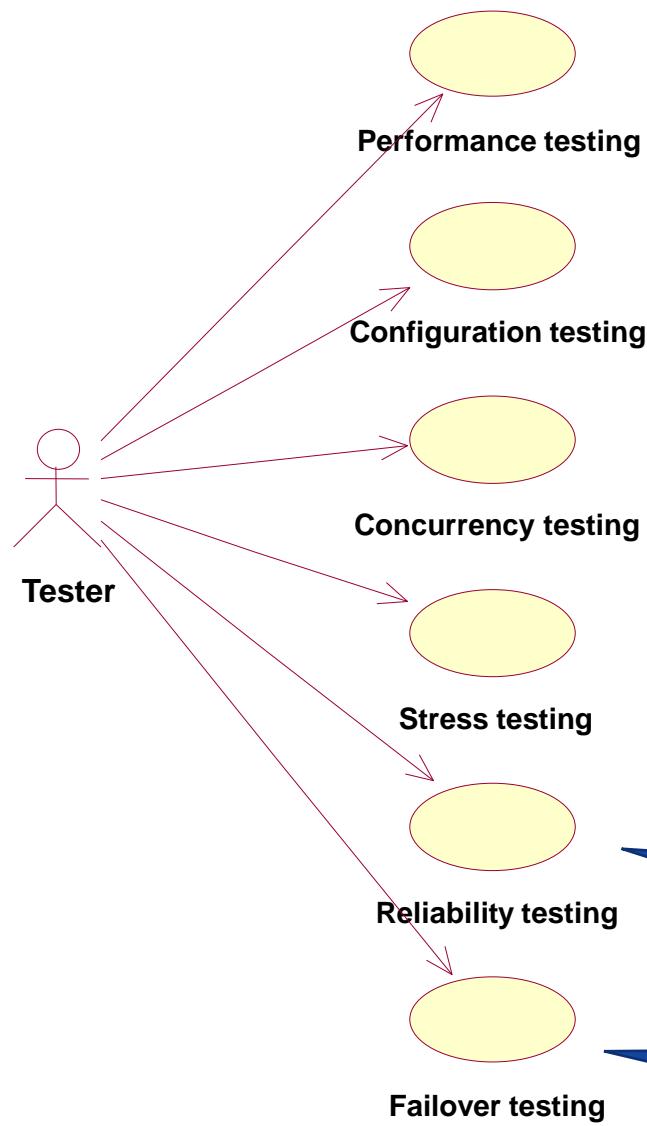
- **Jellemzők:**

- E2E: end-to-end
- Funkcionális tesztek + **nem-funkcionális jellemzők** is!

- **Kiemelhető:**

- Éles környezethez hasonló
- Felhasználói profil figyelembe vétele (terhelés)
- Rendszer **alkalmazhatósági korlátok** megállapítása  
(erőforrás-használat, telítődés)
- **Hibahatások vizsgálata**

# Rendszerteszt típusok (példa)



Teljesítmény teszt

- Valós terhelés, válaszidők

Konfiguráció teszt

- HW és SW beállítások

Konkurens viselkedés tesztje

- Holtpont, kiéheztetés

Terhelés (löket) teszt

- Telítődés vizsgálata

Megbízhatóság tesztje

- Hibahatások vizsgálata

Hibakezelés tesztje

- Hibadetektálás, redundancia

# III. Átvételi tesztelés (acceptance testing)

Gyakran **vevő** vagy **végfelhasználó** végzi

- Cél: Valóságos környezet hatásának tesztelése
  - Rendszerbe vetett **bizalom** megteremtése
  - **Felhasználói elvárások** figyelembe vétele  
(nem specifikált is)
- Lehet **szerződéses** elvárás is
  - Dobozos vagy nagyvállalati termék
- Alfa és béta teszt
- A/B testing, „testing in production”

# Összefoglalás: Különbség a szintek között

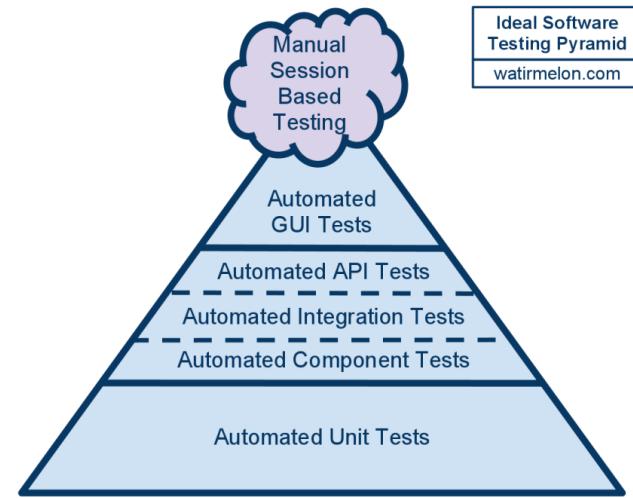
*How Google Tests Software* könyv ajánlásai:

(small ~ unit, medium ~ integration, large ~ system)

|                        | Small        | Medium         | Large            |
|------------------------|--------------|----------------|------------------|
| Javasolt futási idő    | < 100 ms     | < 1 sec        | minél gyorsabban |
| Időkorlát (kill)       | 1 perc       | 5 perc         | 1 óra            |
| Erőforrás              | Small        | Medium         | Large            |
| Hálózat (socket)       | Mocked       | csak localhost | Igen             |
| Adatbázis              | Mocked       | Igen           | Igen             |
| Fájl hozzáférés        | Mocked       | Igen           | Igen             |
| Rendszerhívás          | Nem          | Nem javasolt   | Igen             |
| Több szál              | Nem javasolt | Igen           | Igen             |
| Sleep utasítás         | Nem          | Igen           | Igen             |
| Rendszer tulajdonságai | Nem          | Igen           | Igen             |

# Tesztautomatizálás

Miért, mit, hogyan, mikor, hol

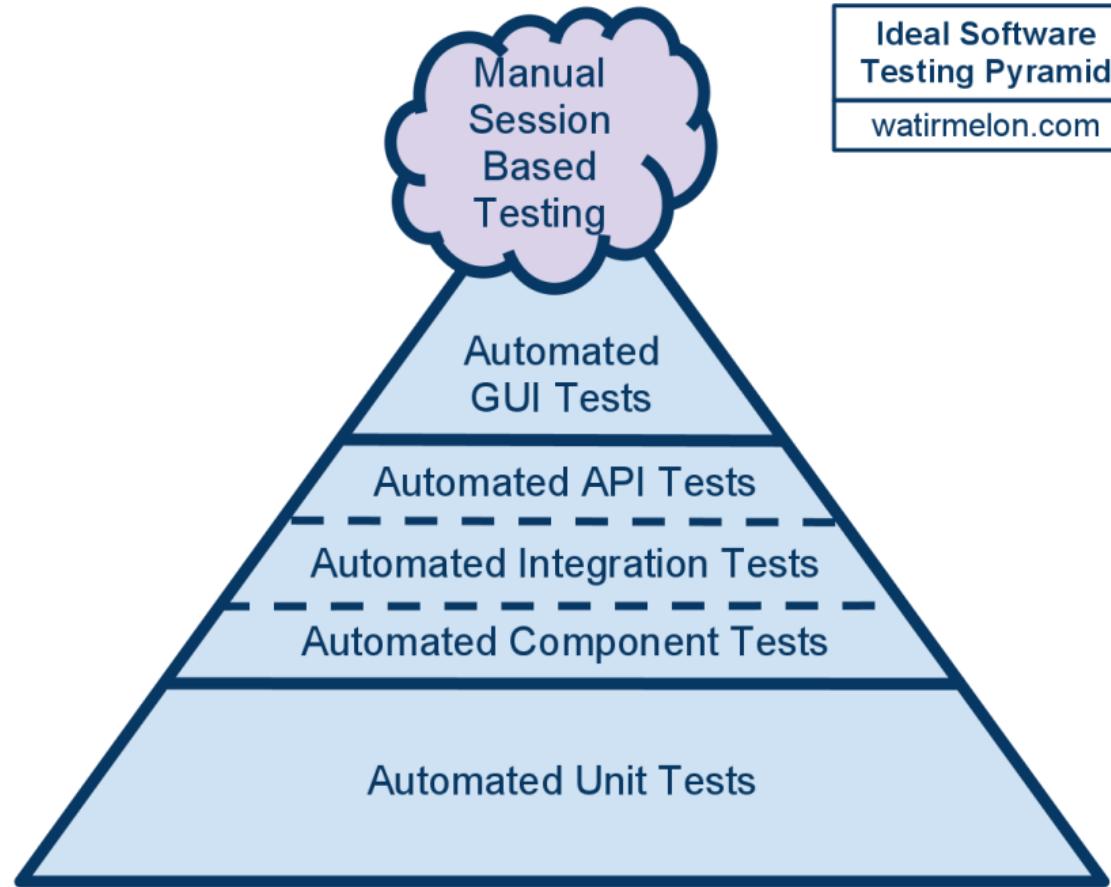
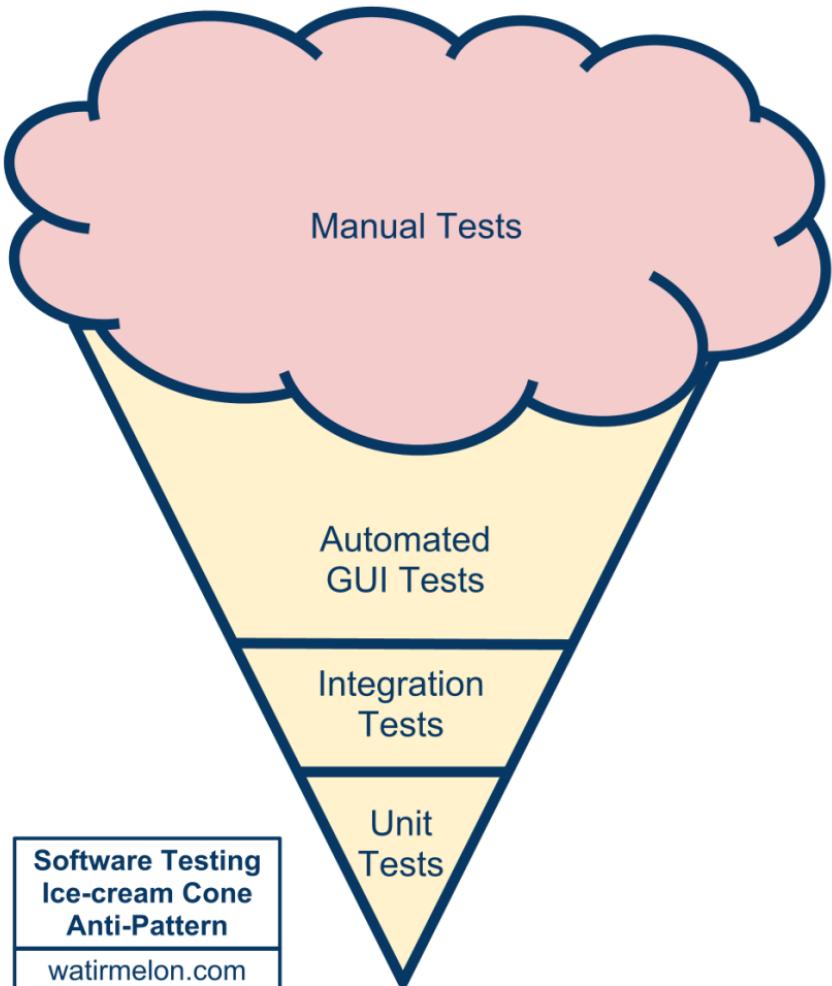


# MIÉRT: Automatikus tesztelés

- Teszt futtatás és/vagy kiértékelés automatizálása
  - Emberi kiértékelés lassú/drága (?)
- Manuális vagy automatikus?
  - **Sok mindenről függ!**
  - Nehézség
    - Pl. GUI, CD írás, rajzolás
  - Tesztelés élethossza
    - Meddig kell a teszt, milyen gyakran
  - Pontosság (false positive, false negative)

Manuális ÉS automatikus!

# MIT: Tesztelési piramis



Source: [Alister Scott](#)

Lásd még: [Mike Cohn](#), [Martin Fowler](#)...

# HOGYAN: Tesztautomatizálási módszerek

## Capture/replay

- Könnyű létrehozni
- Nehéz karbantartani

## Structured Scripting

- Közös műveletek szkript könyvtárban
- Teszt logika, adat és kód összefonódik

## Data-driven

- Tesztek be/kimenete külső forrásból (fájl, DB...)

## Keyword-driven

- Teszt üzleti/szakterületi kulcsszavakból áll
- minden kulcsszó mögött kódrészlet

## Model-based

- Tesztkiválasztás is (részben) automatikus

Lásd: [ISTQB syllabus](#)

# HOGYAN: Tipikus részfeladatok (SEARCH)

## Setup

- Legfrissebb verzió fordítása/telepítése
- Különböző platform, OS, böngésző... → konténerek, cloud...

## Execution

- Egyszerű script / xUnit / keretrendszer
- Naplázás

## Analysis

- Teszt kiértékelése
- Sokszor nem triviális (elosztott rendszer...)

## Reporting

- Tesztek ezrei esetén nem elegek a naplófájlok
- Összesítő információk

## Cleanup

- Ismert, tiszta állapotba visszaállítás
- Cél: tesztek ne befolyásolják egymás futását

## Help

- Teszt kód is ugyanolyan kód, azt is dokumentálni kell
- Sokszor a teszt kód hosszabb, mint az éles

# **MIKOR:** Végrehajtási stratégiák

Teljes  
( minden teszt )

- Legalább minden (jelentős) kiadás előtt

Smoke test

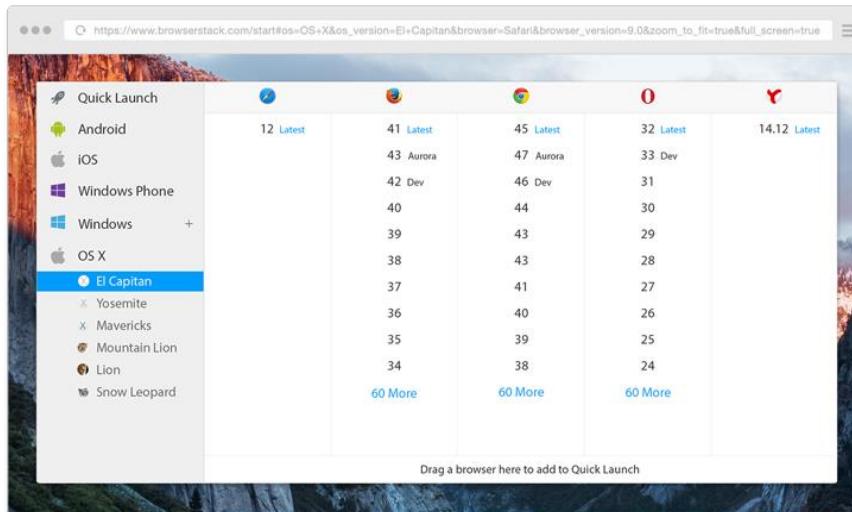
- Kis tesztkészlet alap funkcionális ellenőrzésre
- Gyors visszajelzés, de kicsi pontosság
- Sokféle név, pl. build verification test (BVT)

Regressziós  
tesztelés

- Részleges újratesztelés (hatáselemzés)
- Teszt prioritizálás

# HOL: Tesztvégrehajtó platformok

- Web: különböző böngésző, OS...
- Mobil: emulált vagy fizikai eszköz...
- Sokféle kulcsrakész megoldás
  - Telepített: Selenium, Robot framework...
  - Fehő: Browserstack, SauceLabs...



## Real Device Coverage List

### IOS DEVICES



iPhone 6  
iOS 8.4

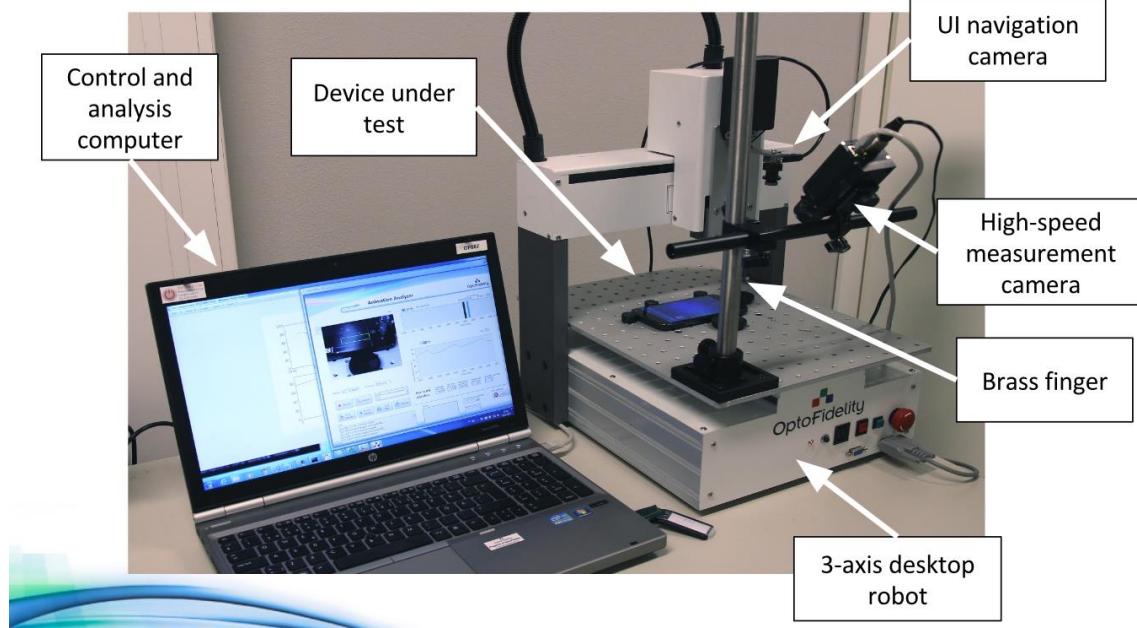


iPhone 6s  
iOS 9.3



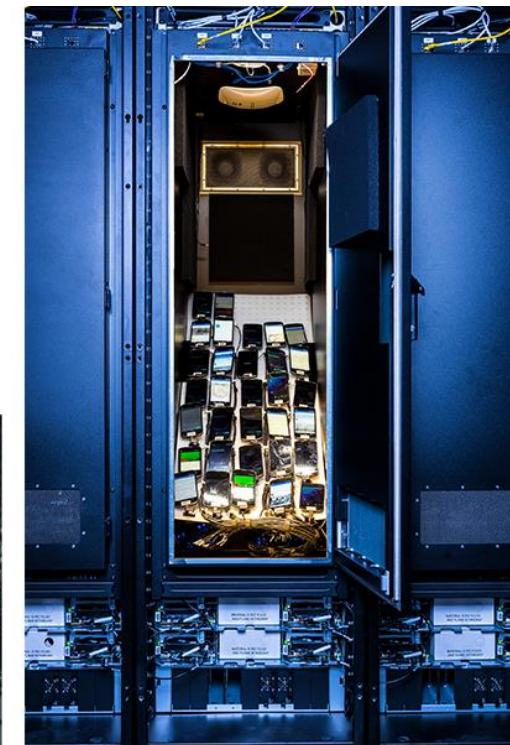
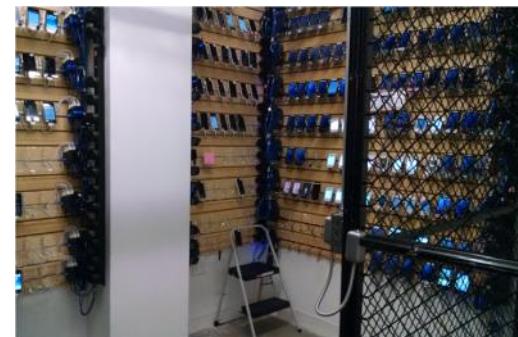
iPhone 6s  
iOS 9.3

# HOL: Tesztelési laborok (web, mobil)

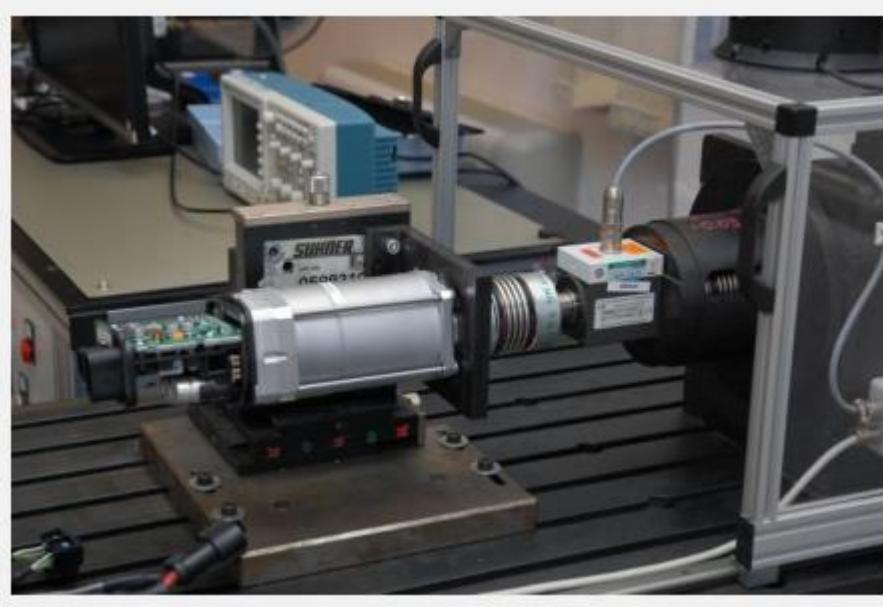


[Robot Assisted Test Automation \(GTAC 2015\)](#)

[Facebook mobile device lab](#)



# HOL: Tesztelési laborok (kritikus rendszer)



[Functional test challenges in safety critical EPAS systems](#), ThyssenKrupp  
Presta (Test&Tea 2015)

[Video and radar test](#), Bosch  
(Test & Tea 2015)



# INFO: ISTQB Test Automation Engineer

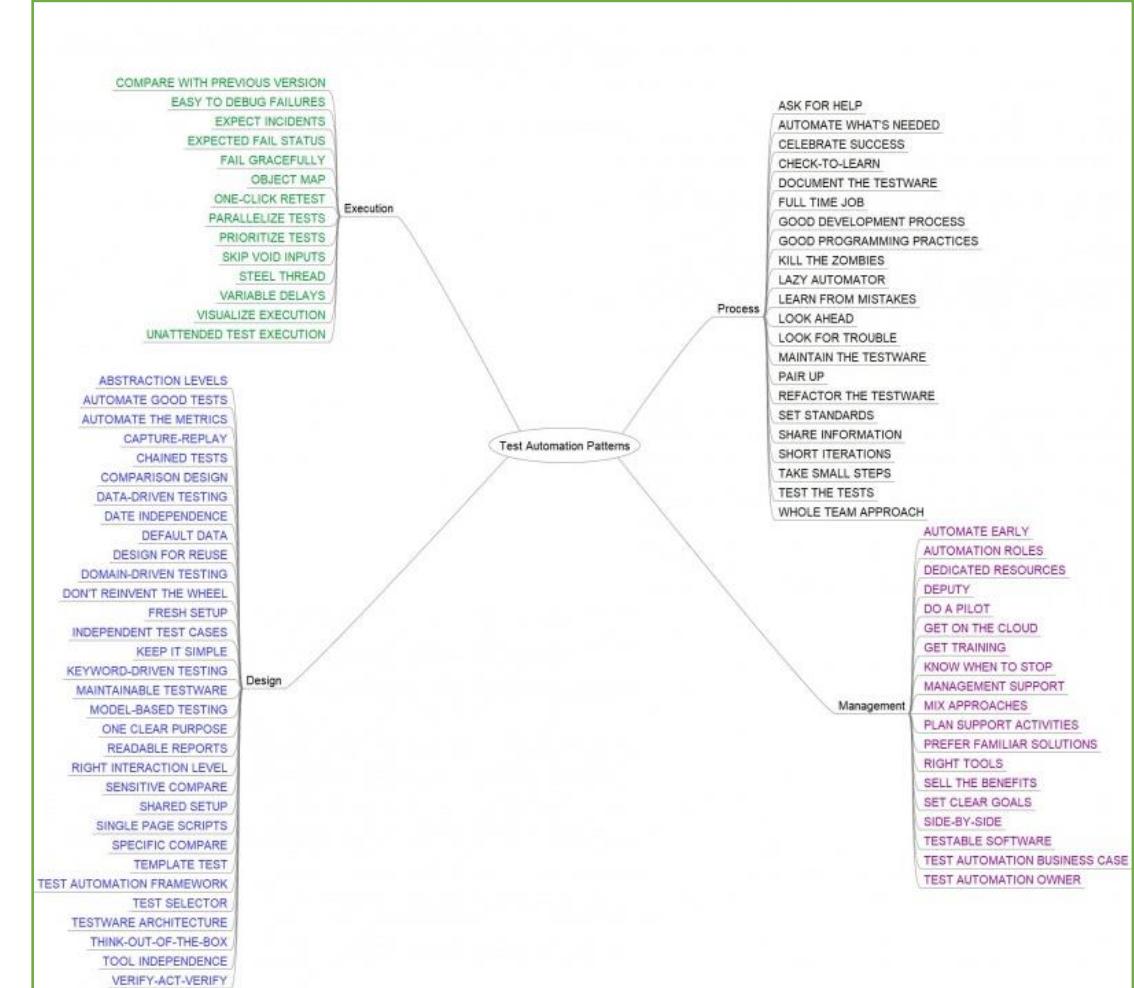
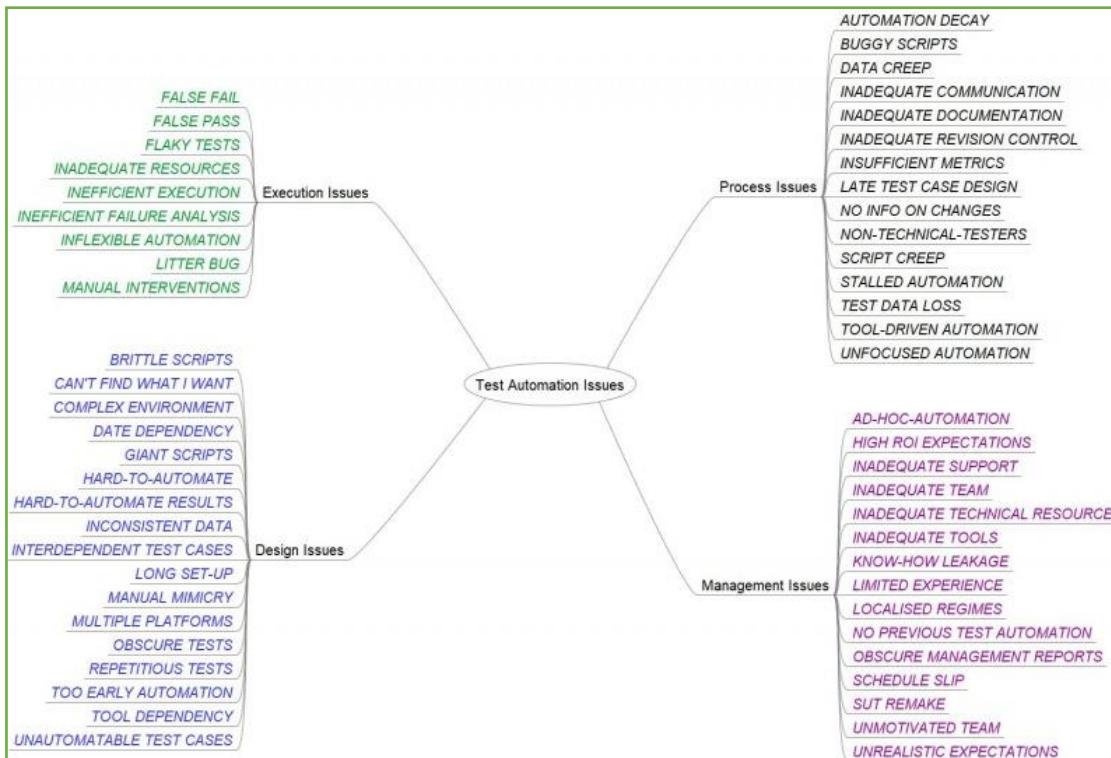
## ISTQB – ADVANCED LEVEL TEST AUTOMATION ENGINEER

|                            |                                         |                                          |                                                             |                                       |                                                          |                                                 |                                        |
|----------------------------|-----------------------------------------|------------------------------------------|-------------------------------------------------------------|---------------------------------------|----------------------------------------------------------|-------------------------------------------------|----------------------------------------|
| Test Automation            | Preparing for Test Automation           | The Generic Test Automation Architecture | Deployment Risks and Contingencies                          | Test Automation Reporting and Metrics | Transitioning Manual Testing to an Automated Environment | Verifying the TAS                               | Continuous Improvement                 |
| Purpose of Test Automation | SUT Factors Influencing Test Automation | Introduction to gTAA                     | Test Automation Approach and Planning of Deployment/Rollout | Selection of TAS Metrics              | Criteria for Automation                                  | Verifying Automated Test Environment Components | Options for Improving Test Automation. |
| Success Factors            | Tool Evaluation and Selection           | TAA Design                               | Risk Assessment and Mitigation Strategies                   | Implementation of Measurement         | Automation within Regression Testing                     | Verifying the Automated Test Suite              | Test Automation Improvement            |
|                            | Design for Testability and Automation   | TAS Development                          | Test Automation Maintenance                                 | Logging of the TAS and the SUT        | Automation within New Feature Testing                    |                                                 |                                        |

Forrás: [ISTQB](#)

# INFO: Test automation patterns

<https://testautomationpatterns.org>



# INFO: Tesztelési konferenciák

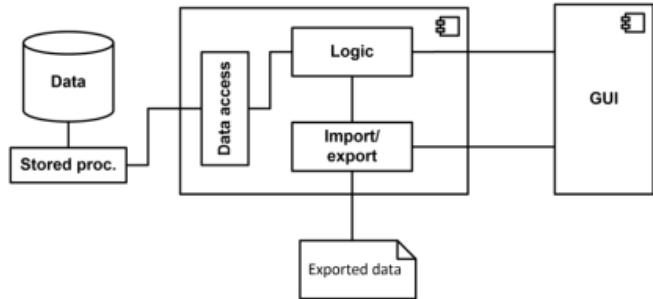


User Conference on  
Advanced Automated Testing



# Összefoglalás

## Példa: mit/hol/hogyan teszteljünk?

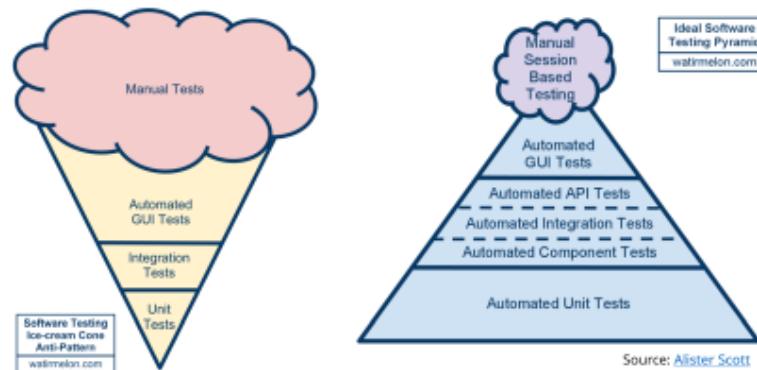


Integrációs és ellenőrzési technikák (vimiac04)

3



## MIT: Tesztelési piramis



Lásd még: [Mike Cohn](#), [Martin Fowler](#)...

Integrációs és ellenőrzési technikák (vimiac04)

15



## Összefoglalás: Különbség a szintek között

How Google Tests Software könyv ajánlásai:

(small ~ unit, medium ~ integration, large ~ system)

|                        | Small        | Medium         | Large            |
|------------------------|--------------|----------------|------------------|
| Erőforrás              | Small        | Medium         | Large            |
| Javasolt futási idő    | < 100 ms     | < 1 sec        | minél gyorsabban |
| Időkorlát (kill)       | 1 perc       | 5 perc         | 1 óra            |
| Hálózat (socket)       | Mocked       | csak localhost | Igen             |
| Adatbázis              | Mocked       | Igen           | Igen             |
| Fájl hozzáférés        | Mocked       | Igen           | Igen             |
| Rendszerhívás          | Nem          | Nem javasolt   | Igen             |
| Több szál              | Nem javasolt | Igen           | Igen             |
| Sleep utasítás         | Nem          | Igen           | Igen             |
| Rendszer tulajdonságai | Nem          | Igen           | Igen             |

Integrációs és ellenőrzési technikák (vimiac04)

12



## HOGYAN: Tesztautomatizálási módszerek

### Capture/replay

- Könnyű létrehozni
- Nehéz karbantartani

### Structured Scripting

- Közös műveletek szkript könyvtárban
- Teszt logika, adat és kód összefonódik

### Data-driven

- Tesztek be/kimenete külső forrásból (fájl, DB...)

### Keyword-driven

- Teszt üzleti/szakterületi kulcsszavakból áll
- minden kulcsszó mögött kód részlet

### Model-based

- Tesztelési kiválasztás is (részben) automatikus

Lásd: ISTQB syllabus



Integrációs és ellenőrzési technikák (vimiac04)

16

# Teszttervezés

Majzik István, Micskei Zoltán

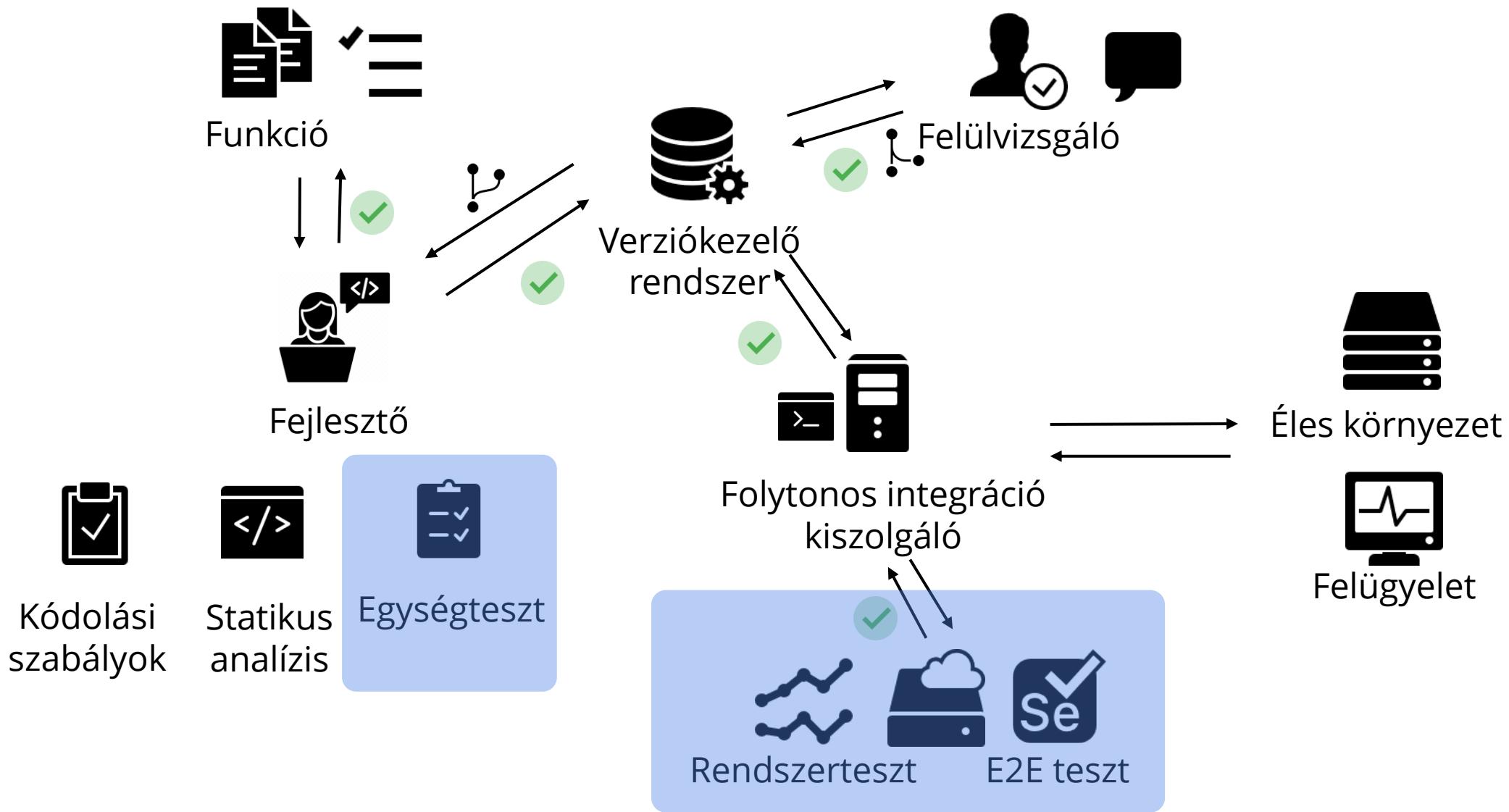


Méréstechnika és  
Információs Rendszerek  
Tanszék



Critical Systems  
Research Group

# Tantárgy tematikája - Áttekintés



Ikonok: icons8.com

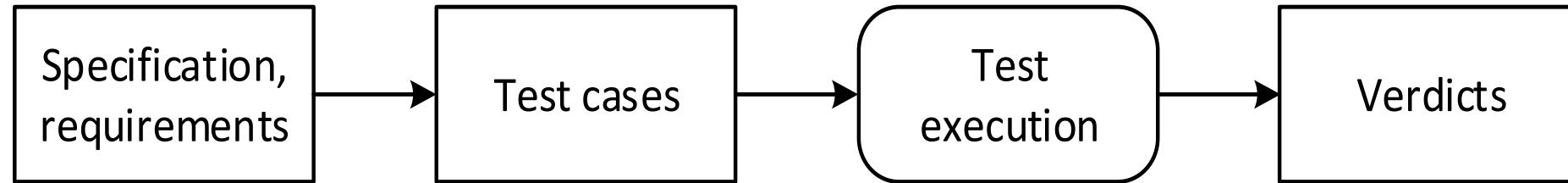
# Miért fontos a teszttervezés?

„More than the act of testing, the act of designing tests is one of the best bug preventers known.”



**Black-Box Testing**  
Techniques for Functional Testing  
of Software and Systems  
**Boris Beizer**

# Tesztelési alapfogalmak (ismétlés)



- SUT: system under test
- Teszteset (test case)
  - bemeneti értékek, végrehajtási feltételek és elvárt eredmények halmaza, amelyeket egy konkrét célért fejlesztettek
- Tesztkészlet (test suite)
- Teszt-orákulum (test oracle)
  - Olyan elv vagy módszer, ami segít eldönten a tesztelőnek, hogy sikeres volt-e a teszt
- Eredmény (verdict): pass / fail / error / inconclusive...

# Megoldandó feladatok

## Tesztkiválasztás (test selection)

- Milyen bemeneti értékeket és adatok használunk?

## Orákulum problémája

- Honnan lesz megbízható orákulum?

## Kilépési feltétel (exit criteria)

- Meddig folytassuk a tesztelést?

# Teszttervezési technikák

Cél: tesztesetek kiválasztása tesztcél alapján

## Specifikáció alapján

- SUT: black box
- Csak a specifikáció ismert
- Előírt funkcionalitás vizsgálata

## Struktúra alapján

- SUT: white box
- Belső felépítés ismert
- Felépítés és működés alapján bejárás

# Lefedettségi metrikák

- Tesztelhető elemek mekkora %-a volt tesztelve
- Tesztelhető elemek
  - Specifikáció: követelmény, funkció...
  - Struktúra: utasítás, döntés...
- Lefedettségi feltétel (coverage criterion)
  - $X$  % az  $Y$  metrikánál
- Ez nem hibafedés!

# Mire jók a lefedettségi metrikák?

## Kiértékelés (mérés)

- Meglévő tesztek kiértékelése
- Hiányzó tesztek azonosítása

## Kiválasztás (cél)

- Teszt tervezése metrika mentén
- Cél adott % elérése

# Specifikáció alapú tesztelés

Ekvivalencia  
partícionálás

Határérték-  
analízis

Use case /  
user story

Kombinatorikus  
módszerek

Döntési  
táblák

...

# Teszttervezési technikák

Cél: tesztesetek kiválasztása tesztcél alapján

## Specifikáció alapján

- SUT: black box
- Csak a specifikáció ismert
- Előírt funkcionalitás vizsgálata

## Struktúra alapján

- SUT: white box
- Belső felépítés ismert
- Felépítés és működés alapján bejárás

# Specifikáció alapú technikák

Ekvivalencia  
partícionálás

Határérték-  
analízis

Use case /  
user story

Kombinatorikus  
módszerek

Döntési  
táblák

...

# Ekvivalencia partícionálás

- Bemenet és kimenet **ekvivalencia osztályai**:
  - Olyan adatok, amelyek várhatóan ugyanazt a hibát fedik le (ugyanazt a programrészét járják be)
  - **Cél**: Egy-egy ekvivalencia osztályból egy-egy teszt adat kiválasztása (többire elvileg hasonló)
- Nagyon **környezetfüggő** (context-dependent)
  - Ismerni kell a környezetet és a SUT-ot!
  - Tesztelő tudásán múlik a módszer hatékonysága

# Ekvivalencia osztályok meghatározása

- Meghatározás heurisztikus folyamat:
  - Kezdés: érvényes és érvénytelen bemeneti adatok
  - Partíciók tovább finomítása
- Tipikus heurisztikák:
  - **Tartomány** (pl. 1-1000)
    - < min, min-max, >max
  - **Halmaz** (pl. RED, GREEN, BLUE)
    - érvényes elem, érvénytelen
  - **Specifikus** (pl. @ az elején)
    - feltétel teljesül, feltétel nem teljesül
  - **Egyéni** (pl. február hónap)

# Tesztesetek származtatása ekv. osztályból

- Több bemenet ekv. osztályainak összekapcsolása
- Érvényes (normál) ekv. osztályok esetén:
  - egy teszt minél több osztályt fedjen le
- Érvénytelen ekv. osztályok esetén:
  - először minden érvénytelen osztályhoz külön teszt legyen
  - egymás hatását ne oltsák ki
  - majd több osztály kombinációja is

# FELADAT: Ekvivalencia partícionálás

**Követelmény:** A hitelfelvételt meg kell tagadni, ha a kért összeg 1M Ft-nál nagyobb és az igénylő 25 évnél fiatalabb, kivéve ha az összeg 3M Ft-nál kisebb és az igénylőnek már volt korábban visszafizetett hitele.

- Bemeneti paraméterek? Ekvivalencia osztályok?
- Van kérdésünk a követelménnyel kapcsolatban?

# Specifikáció alapú technikák

Ekvivalencia  
partícionálás

Határérték-  
analízis

Use case /  
user story

Kombinatorikus  
módszerek

Döntési  
táblák

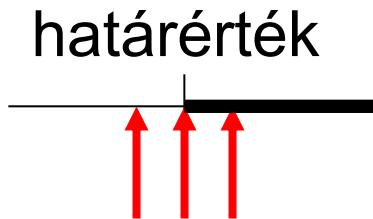
...

# Határérték-analízis

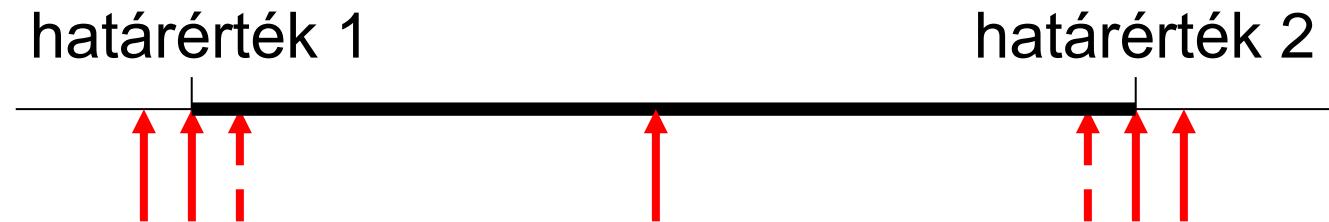
- Adattartományok határait vizsgálja
  - Egy-egy ekvivalencia osztály határaira koncentrál
  - Bemeneti és kimeneti tartományokra is
- Tipikus megtalált hibák:
  - relációs operátorok
  - ciklusok be- és kilépési feltételei
  - adatstruktúrák mérete
  - ...

# Tipikus teszt adatok határértékeknél

- Egy határérték 3 tesztet jelent:



- Egy tartomány 5-7 tesztet jelent:



# FELADAT: Határérték-analízis

**Követelmény:** Legalább három könyvet kell rendelni a kedvezményes ár eléréséhez, és tíz könyv rendelése esetén további kedvezmény jár.

- Milyen értékeket használunk a tesztelésre?
- Van kérdésünk a követelménnyel kapcsolatban?

# Specifikáció alapú technikák

Ekvivalencia  
partícionálás

Határérték-  
analízis

Use case /  
user story

Kombinatorikus  
módszerek

Döntési  
táblák

...

# Tesztek származtatása használati esetekből

- Tipikus tesztesetek:
  - 1 teszt: fő ág („happy path”, „mainstream”)
    - Ellenőrzés: utófeltételek vizsgálata
    - Alternatív lefutások: mindegyikhez külön teszteset
    - Előfeltételek (nem)teljesülése
- Jobbára magasabb szintek (rendszer, elfogadási...)

# FELADAT: Használati esetek (use case)

## 3.2.5 Vásárlás

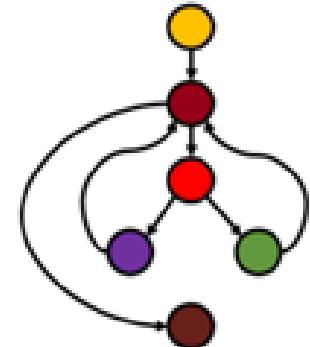
|                       |                                                                                                                                                                                                |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ID / Név:             | UC6 / Buy                                                                                                                                                                                      |
| Verzió:               | 1.0                                                                                                                                                                                            |
| Leírás:               | A felhasználó a megvásárolni kívánt könyvek kosárba tétele után kifizetheti azokat, ha megad ehhez egy érvényes bankkártya számot, amiről a vételár levonható.                                 |
| Előfeltétel:          | Van legalább egy könyv a felhasználó kosarában, megadott egy érvényes bankkártya számot a kosár megtekintésénél és ezt követően nem navigált el a kosár tartalmát listázó oldalról.            |
| Utófeltétel:          | Az ügyfél kosara kiürül, és a könyveket megvásárolja.                                                                                                                                          |
| Trigger:              | A felhasználó a fizetés funkciót választja.                                                                                                                                                    |
| Normál lefutás:       | <ol style="list-style-type: none"><li>A kosárban lévő könyv példányok kikerülnek az adatbázisból.</li><li>A kosár is kiürül.</li><li>A fizetés ténye belekerül a tranzakció naplóba.</li></ol> |
| Alternatív lefutások: | <ul style="list-style-type: none"><li>- Ha nincs megadva vagy érvénytelen a bankkártya szám, akkor nem változik sem a készleten lévő, sem a kosárban lévő könyvek listája.</li></ul>           |

# Struktúra alapú tesztelés

Forráskód:

```
int a = read();
while(a < 16) {
 if(a < 10) {
 a += 2;
 } else {
 a++;
 }
}
a = a * 2;
```

Control-flow graph (CFG):



# Teszttervezési technikák

Cél: tesztesetek kiválasztása tesztcél alapján

## Specifikáció alapján

- SUT: black box
- Csak a specifikáció ismert
- Előírt funkcionalitás vizsgálata

## Struktúra alapján

- SUT: white box
- Belső felépítés ismert
- Felépítés és működés alapján bejárás

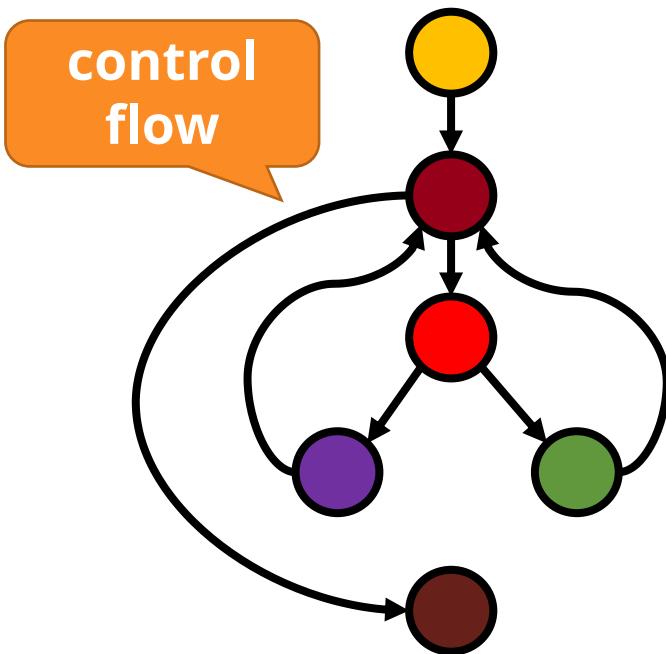
# Mi a belső felépítés?

Kód esetén: forráskód struktúrája

Forráskód:

```
int a = read();
while(a < 16) {
 if(a < 10) {
 a += 2;
 } else {
 a++;
 }
}
a = a * 2;
```

Control-flow graph (CFG):



Megjegyzés: CFG pontos definíciójával most nem foglalkozunk

# Alapfogalmak

```
int t = 1;
Speed s = SLOW;
```

```
if (! started){
 start();
}
```

```
if (t > 10 && s == FAST){
 brake();
} else {
 accelerate();
}
```

Utasítás

Blokk

Feltétel

Döntés

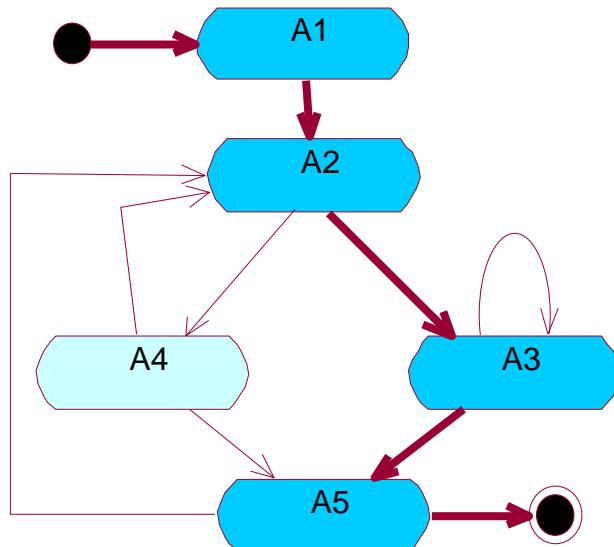
Döntési ág

# Alapfogalmak

- Utasítás (statement)
- Blokk (block)
  - Utasítások egybefüggő sorozata, amik között nincs elágazás vagy függvényhívás
- Feltétel (condition)
  - Egyszerű vizsgálat, amiben nincs logikai (Boole) operátor
- Döntés (decision)
  - Nulla vagy több logikai operátorral összekötött feltételből álló kifejezés
- Döntési ág (branch)
  - Egy döntés lehetséges kimenetele
- Út (path)
  - Utasítások sorozata, tipikusan a modul be és kilépési pontja között

# 1. Utasítás lefedettség

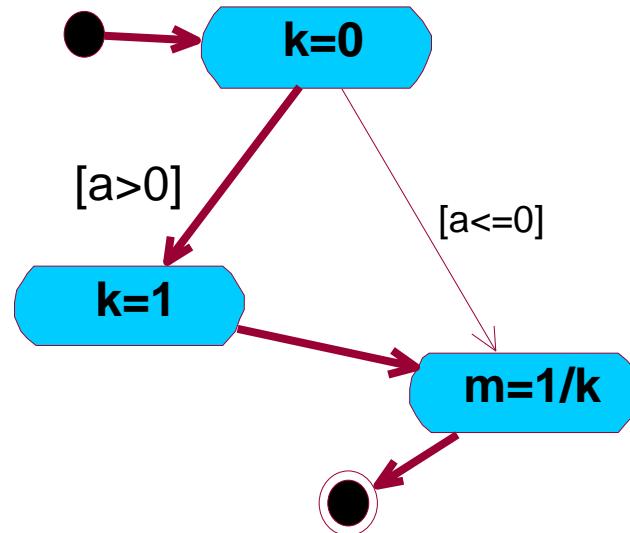
Tesztelés során végrehajtott utasítások száma  
Összes utasítás szám



Utasítás lefedettség:  $4/5 = 80\%$

# Utasítás lefedettség értékelése

Minden utasítást legalább egyszer végrehajtunk



Utasítás lefedettség: 100%

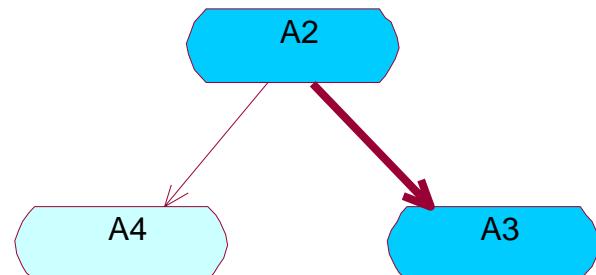
DE: hiányzik a  $[a \leq 0]$  ág

Üres ágak lefedését nem garantálja

## 2. Döntési lefedettség

Tesztelés során előfordult döntési eredmények száma

Döntési eredmények lehetséges száma



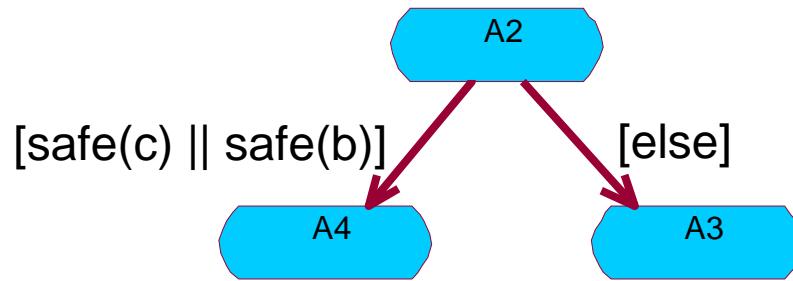
Döntési lefedettség: 50%

Hány eredménye lehet egy döntésnek?

# Döntési lefedettség értékelése

Minden utasítást legalább egyszer végrehajtunk

Minden ágat lefedünk (üres ágakat is)



| # | safe(c) | safe(b) | safe(c)    safe(b) |
|---|---------|---------|--------------------|
| 1 | T       | F       | T                  |
| 2 | F       | F       | F                  |

100% döntési lefedettség

DE: safe(b) = true eset hiányzik!

Feltételek kombinációjára nem figyel

# További lefedettségi metrikák (lásd MSc)

- Condition Coverage
- Condition/Decision Coverage (C/DC)
- Modified Condition/Decision Coverage (MC/DC)
- Multiple Condition Coverage (MCC)
- Loop Coverage
- ...
- All-Defs Coverage
- All-Uses Coverage
- ...

# FELADAT: Struktúra alapú tesztelés

```
1 int pow(int n, int k) {
2 if (n < 0 || k < 0) {
3 return -1;
4 }
5 int p = 1;
6 for (int i = 0; i < k; i++) {
7 p *= n;
8 }
9 return p;
}
```

Rajzold fel a kód CFG-jét!  
Tervezz teszteket:

- 100% utasítás lefedettség
- 100% döntési lefedettség

# Kód lefedettség számolása a gyakorlatban

- Mindegyik eszköznél más definíciók
- Megvalósítás
  - Forrás/bináris kód instrumentálása
  - Lefedettséget számoló utasítások hozzáadása

```
if (a > 10){
 CoveredBranch(1, true);
 b = 3;
} else {
 CoveredBranch(1, false);
 b = 5;
}
send(b);
```

Lásd például: [Is bytecode instrumentation as good as source code instrumentation](#), 2013.

# Kód lefedettség használata

Mire  
használható

- Egyáltalán nem tesztelt kódrészlet azonosítása
- Tesztkészlet „teljességének” értékelése
- Kilépési feltételekben

Mire nem  
használható

- Hiányzó/hiányos követelmények azonosítása
- Kód/teszt minőségével csak indirekt kapcsolat

# Kód lefedettség használata a gyakorlatban

- Microsoft tapasztalata:
  - „Test suite with **high code coverage** and **high assertion density** is a good indicator for code quality.”
  - „**Code coverage alone** is generally **not enough** to ensure a good quality of unit tests and should be used with care.”
  - „The **lack of code coverage** to the contrary clearly indicates a **risk**, as many behaviors are untested.”
- (Forrás: „Parameterized Unit Testing with Microsoft Pex”)
- Kapcsolódó cikkek:
  - „*Coverage Is Not Strongly Correlated with Test Suite Effectiveness*”, 2014.  
DOI: 10.1145/2568225.2568271
  - „*The Risks of Coverage-Directed Test Case Generation*”, 2015. DOI:  
10.1109/TSE.2015.2421011

# Teszttervezési technikák összefoglalása

- Specifikáció és struktúra alapú technikák
  - Több, egymást kiegészítő módszer
  - Mindegyik módszert gyakorolni kell
- Technikák kombinációja hasznos
  - Példa (Microsoft):  
specifikáció alapú: 83% kód lefedettség  
+ felderítő tesztelés: 86% kód lefedettség  
+ struktúra alapú: 91% kód lefedettség

# Összefoglalás

## Teszttervezési technikák

Cél: tesztesetek kiválasztása tesztcél alapján

### Specifikáció alapján

- SUT: black box
- Csak a specifikáció ismert
- Előírt funkcionalitás vizsgálata

### Struktúra alapján

- SUT: white box
- Belső felépítés ismert
- Felépítés és működés alapján bejárás

7

ftsrG

## Specifikáció alapú technikák

Ekvivalencia  
partícionálás

Határérték-  
analízis

Use case /  
user story

Kombinatorikus  
módszerek

Döntési  
táblák

...

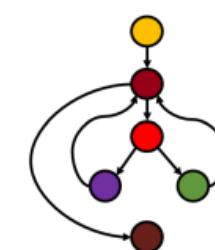
## Mi a belső felépítés?

Kód esetén: forráskód struktúrája

Forráskód:

```
int a = read();
while(a < 16) {
 if(a < 10) {
 a += 2;
 } else {
 a++;
 }
 a = a * 2;
```

Control-flow graph (CFG):



12

ftsrG

28

ftsrG