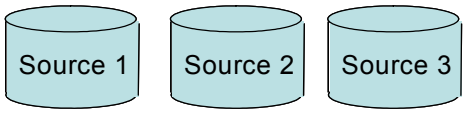
Integration von molekular-biologischen Daten

- Datenintegration
- Bio-Datenbanken und Integrationsprobleme
- Bisherige Ansätze in der Bioinformatik
- Kleisli
- TAMBIS
- GenMapper

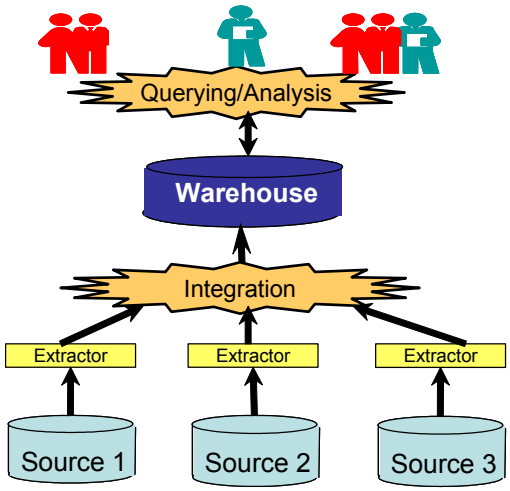
Datenintegration

- Problem: Zugriff auf verwandte Daten in verschiedenen Datenquellen
- Ziel: Einheitliche Sicht auf heterogene Datenquellen für flexible Abfragen und Analysen
- Aufgaben:
  - Schemaintegration: z.B. einheitliche Attributnamen
  - Instanzintegration: z.B. einheitliche Attributwerte
- Traditionelle Ansätze:
  - Data Warehousing
  - Mediation



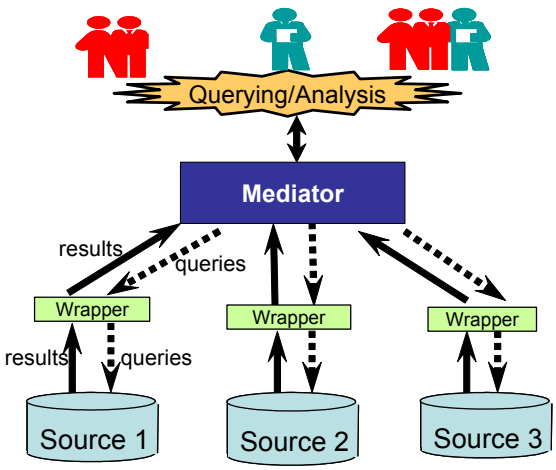
Data Warehousing

- Zentrale Datenbank: Data Warehouse
  - Materialized integration, eager integration, a-priori integration
- Replikation aller relevanten Quelldaten
  - Extraktion, Transformation, Bereinigung der Quelldaten
- Vorteile
  - Verfügbarkeit, Query-Performanz, Analysenmöglichkeiten
- Probleme
  - Hoher Integrationsaufwand, Datenaktualität, Erweiterbarkeit

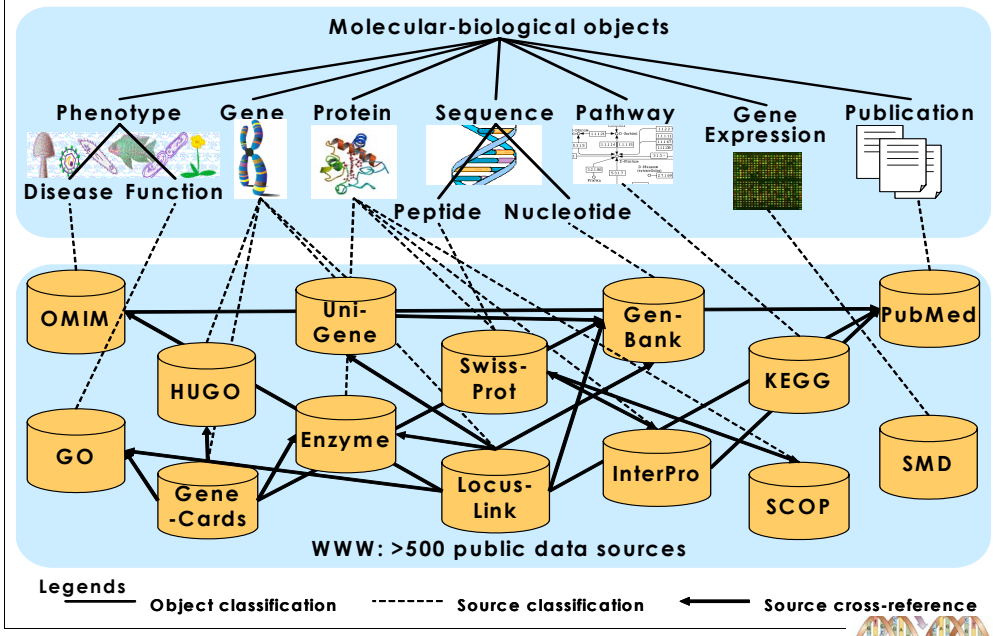


Mediation

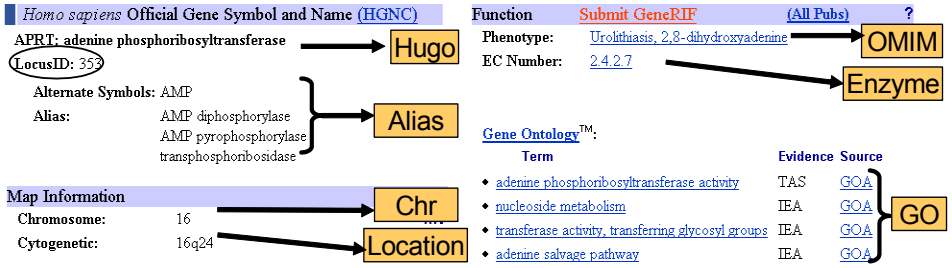
- Zentrale Zugriffsschnitstelle: Mediator
  - Virtual integration, lazy integration, on-demand integration
- Keine Replikation der Quelldaten
  - Bestimmung der relevanten Quellen für eine Abfrage
  - Abfrage der einzelnen Quellen und Kombination der Ergebnisse
- Vorteile
  - Datenaktualität
- Probleme
  - Query-Performanz, Verfügbarkeit, Evolution der Quellen



Bio-Datenbanken



LocusLink Beispiel



- Objekte desselben Typs: Stabile Ids (accessions), feste Anzahl von Annotationsattributen
- Annotation mittels Verlinkung (Cross-References): Semantische Korrespondenzen
- Häufige Abfragen:
  - "*Finde Gene mit Position P auf Chromosom C* ***und*** *Funktionen F, die* ***aber nicht*** *mit Krankheit K im Zusammenhang stehen.*"
- Anwendungsspezifische Annotationssichten
  - Grosse Anzahl von Objekten unterschiedlicher Objekttypen
  - Flexible Auswahl und Kombination der Annotationsattribute

Integrationsprobleme

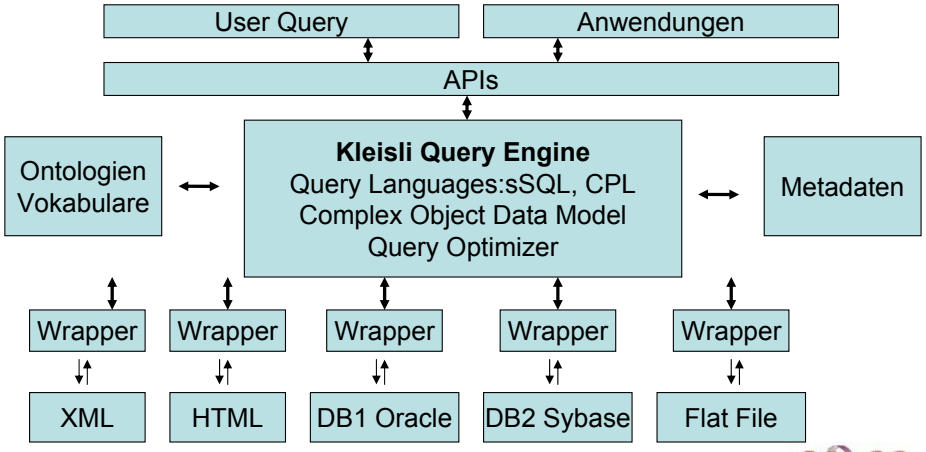
- Zahlreiche öffentliche Datenquellen (>500, wachsend)
  - autonom, eingeschränkter Zugriff (Webbrowser), meistens datei-basiert
- Semantische Heterogenität
  - z.B. Definitionen des Genbegriffs, Synonyme für Gennamen
- Verschiedene Repräsentationen für den gleichen Objekttypen, mit unterschiedlichen Annotationen
  - z.B. Gene in LocusLink, Unigene, Ensembl, ...
- Kontinuierliche Evolution und Pflege (curation)
  - Änderungspropagierung
- Integration mittels Web-Links
  - Hilfreich für interaktive Navigation, jedoch nicht für Analysen grosser Anzahl von Objekten, z.B. genen

Existierende Systeme im Bio-Bereich

- Applikationsspezifisches globales Schema: Probleme für Konstruktion, Evolution, und Skalierbarkeit
  - Data warehouses: IGD, GIMS, GeneExpress, DataFoundry
  - Mediatoren: TAMBIS, P/FDM, KIND
- Mehr Flexibilität: Globales Schema als Vereinigung der lokalen Schemas
  - Mediatoren: DiscoveryLink, K2, Kleisli
  - Einheitliche (low-level) Query-Schnittstelle: SQL (DiscoveryLink), Collection Programming Language (CPL - K2) und sSQL (nested relational - Kleisli)
- Mehr Flexibilität: Verzicht auf ein globales Schema
  - SRS, DBGET/LinkDB: Information Retrieval, Textindexierung und -suche
  - GenMapper: Generisches Datenmodell, flexible View-Generierung

Kleisli

- Mediator Ansatz basierend auf Wrapper-Technologie



Kleisli

- Complex Object Data Model
  - erlaubt zusammengesetzte, verschachtelte Typen
  - allgemeine Syntax:

$$t ::= \texttt{num} \mid \texttt{string} \mid \texttt{bool} \mid \{t\} \mid \{|t|\} \mid [t] \mid (l_1 : t_1, ..., l_n : t_n) \mid \langle l_1 : t_1, ..., l_n : t_n \rangle$$

  - Mengen {t}, Multimengen {|t|}, Listen [t]
  - Records $(l_1 : t_1, ..., l_n : t_n)$
  - Varianten $\langle l_1 : t_1, ..., l_n : t_n \rangle$
- Syntax ist selbstbeschreibend, keine Schema Infos notwendig

Kleisli

- Typ-Beispiel

```
(#title:string,#uid:num,
 #accession:string, #feature:{(
    #name:string, #start:num, #end:num,
    #anno:[(#anno_name:string, #descr:string)])})
```

- Instanz-Beispiel

```
(#title: "PROTEIN-TYROSINE PHOSPHATASE 1C ...",
 #uid: 131470, #accession: "131470", #feature: {(
 #name: "source", #start: 0, #end: 594, #anno: [
 (#anno_name: "organism", #descr: "Mus musculus"),
 (#anno_name: "db_xref", #descr: "taxon:10090")]),
          ...})
```

Kleisli

- Anfragen
  - funktional gekapselt => für verschiedene Quellen wiederverwendbar
- Beispiel
  - create function get-title-from-featureTable (DB) as
    select title: x.title, feature: x.feature
    from DB x where x.title like '%tyrosine'

Kleisli

- Beispiel mit Verschachtelung
  - Konvertiert einen komplexen Type in flache Tabelle

```
create function flatten-featureTable (DB) as
 select title: x.title, feature: f.name, start: f.start, end: f.end,
        anno-name: a.anno_name, anno-descr: a.anno_descr
 from DB x, feature f, f.anno.l2s a
```

  - Verschachtelung

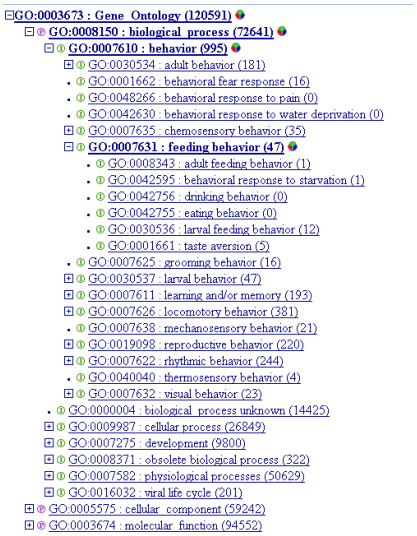
```
create function nest-featureTable-by-organism (DB) as
 select  organism: z, entries: ( select from DB x, x.feature f, f.anno a
                                 where a.anno_name=‘organism’ and a.descr=z)
 from ( select distinct y.anno-descr
        from DB.flatten-featureTable y
        where y.anno-name=‘organism’) z
```

Kleisli, Zusammenfassung

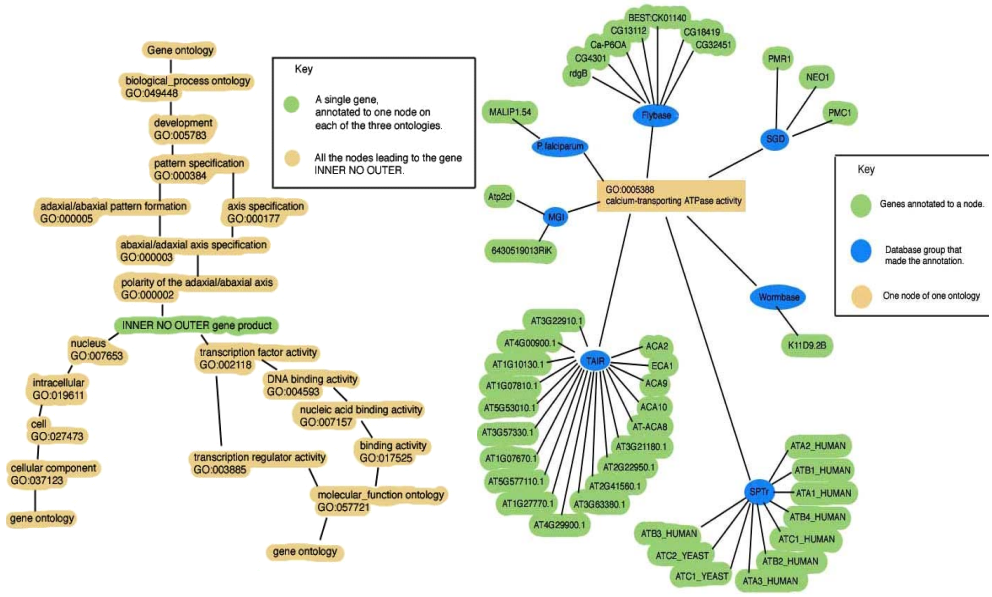
- Integriert mehrere Datenquellen mittels Wrapper
- Verteilte Datenquellen
- Komplexe Datentypen
  - natürlichere Modellierung als rel. Schema
  - Syntax ist selbstbeschreibend
  - notwendig um Ergebnisse von Kleisli-Anfragen zu verarbeiten
- Anfragen, sSQL
  - funktional gekapselt
  - erlauben komplexe Typen als Ergebnis
  - rekursiv aufrufbar
- Collection Programming Language (CPL)
  - erlaubt Anfragen direkt in Programmiersprache (Perl); Ergebnis: Perl-Obj
- sSQL und CPL werden optimiert

Nutzung von Ontologien (1)

- Ontologie: Konzeptualisation einer Domäne
  - Konzepte, Beziehungen, Attribute, Konstraints, Objekte, Werte
- Unterschiedliche Formen
  - Vokabular der relevanten Begriffe
  - Definition der Begriffe
  - Spezifikation der Beziehungen zwischen den Begriffen
- Hauptziel: Wiederverwendung
  - Reduzierung der semantischen Heterogenität zwischen Datenbanken
  - Auf Schemaebene: Einheitliche semantische Sicht auf Daten
  - Auf Instanzebene: Einheitliche Annotation der Objekte

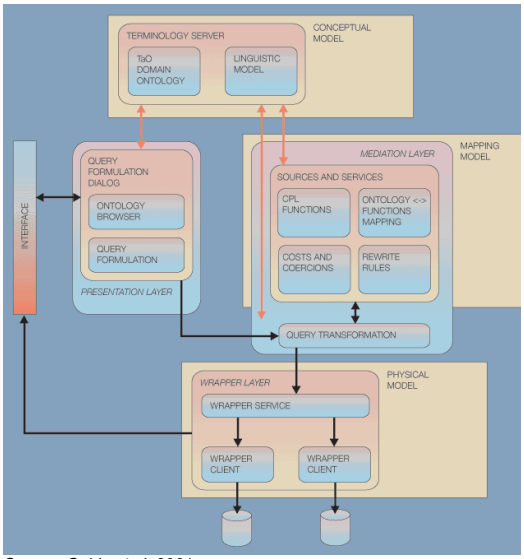


Nutzung von Ontologien (2)



TAMBIS - Architektur

- Transparent Access to Multiple Bioinformatics Sources
  - Forschungsprototyp, Universität Manchester (UK)
- Globales Schema: TAMBIS Ontology
  - domänenspezifisch, 250 Begriffe (von insgesamt 1800 Begriffen)
- Integrierte Quellen:
  - SwissProt: Protein-Sequenzen
  - Prosite: Protein-Motifs
  - Enzyme: Enzyme-Klassifikation
  - CATH: Protein-Domänenstrukturen
  - BLAST (Sequenzhomologie)

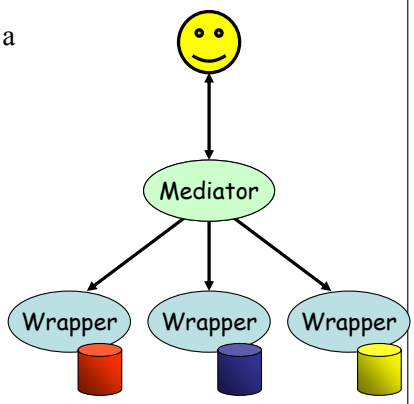


Source: Goble et al, 2001

Tambis: Mediators

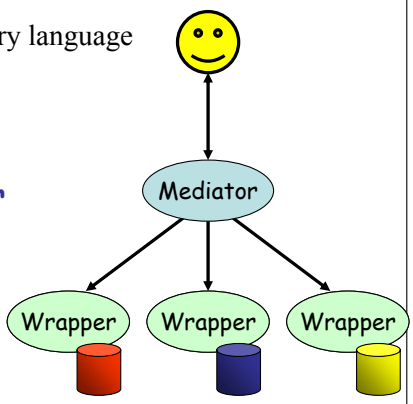
•The mediator is an information broker. It uses a conceptual knowledge base of biology to:

- Describe a universal model
- Help users form queries
- Translate the mediator's model to the sources' model



Tambis: Wrappers

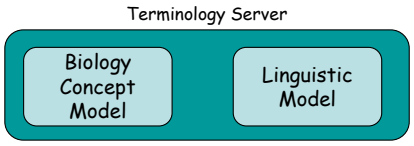
•Wrappers create the illusion of a common query language for each information resource.

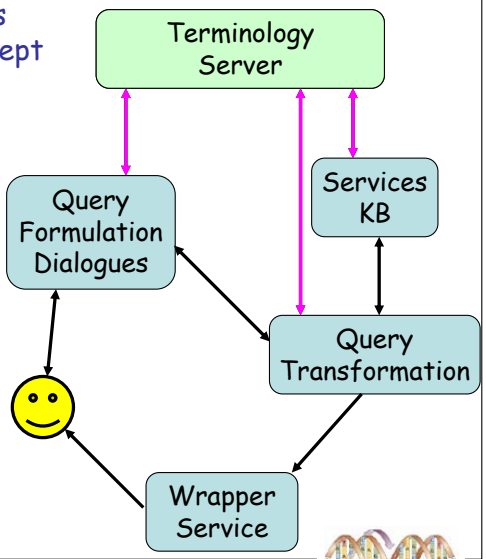


- This insulates the mediator from differences in source access methods
- The current wrapper language is CPL

Tambis: Architecture

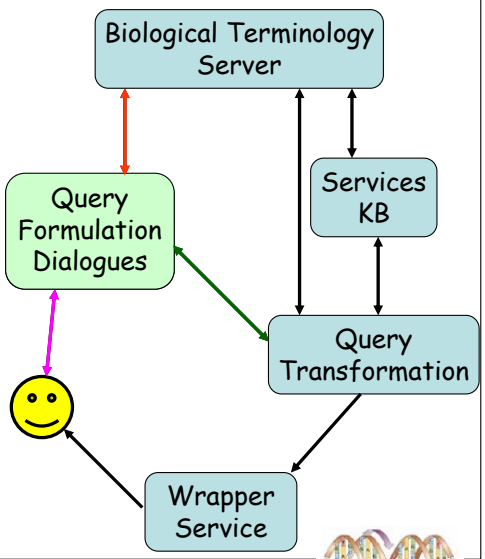
- The Terminology Server provides services for reasoning about concept models, answering questions like:
  - What can I say about Proteins?
  - What are the parents of concept X?
- It communicates with other modules through a well-defined interface





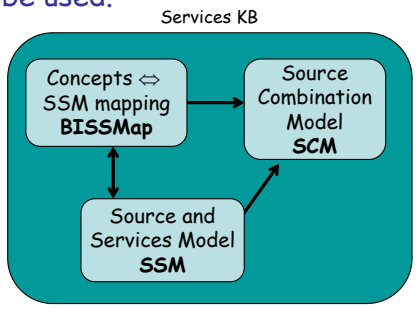
Tambis: Architecture

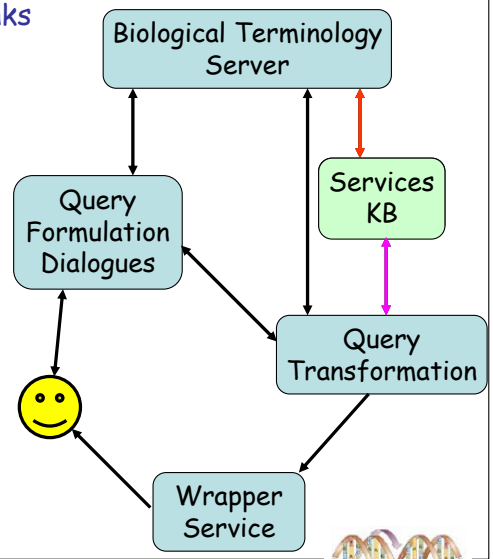
- The user interacts with Query Formulation Dialogues, expressing queries in terms of the biological model.
- The dialogues are driven by the content of the model, guiding the user towards sensible queries.
- The query is then passed to the transformation process, which may require further user input to refine and instantiate the query.



Tambis: Architecture

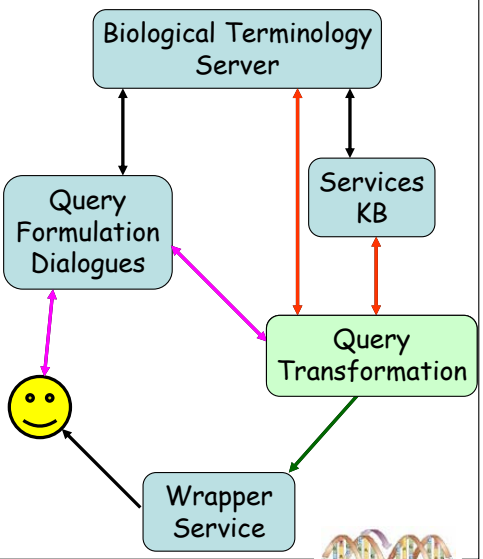
- The Services Knowledge Base links the biological ontology with the sources and their schemas.
- This information is used by the transformation process to determine which source should be used.





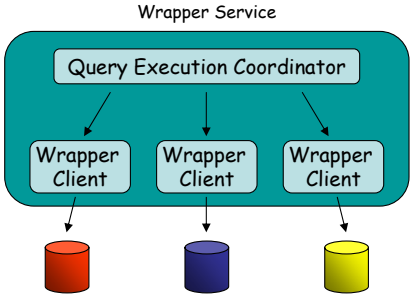
Tambis: Architecture

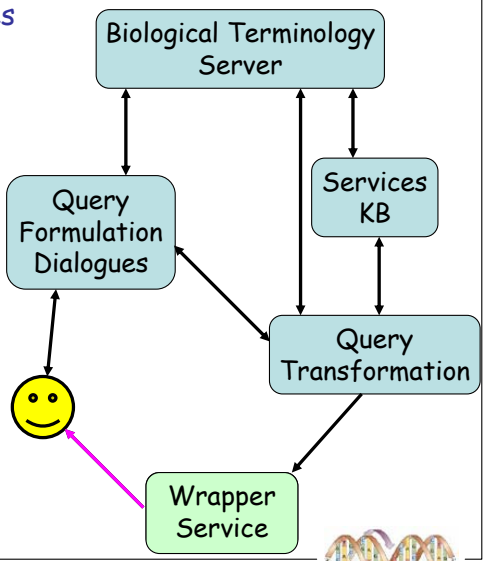
- Query Transformation takes the conceptual source-independent queries and rewrites to produce executable query plans.
- To do this it requires knowledge about the biological sources and the services they offer.
- Information about particular user preferences - say favourite databases or analysis methods - may also be incorporated by the query planner.
- The query plans are then passed to the wrappers.



Tambis: Architecture

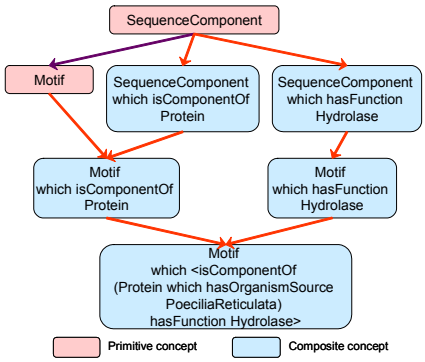
- The Wrapper Service coordinates the execution of the query and sends each component to the appropriate source.
- Results are collected and returned to the user.





TAMBIS - Ontology

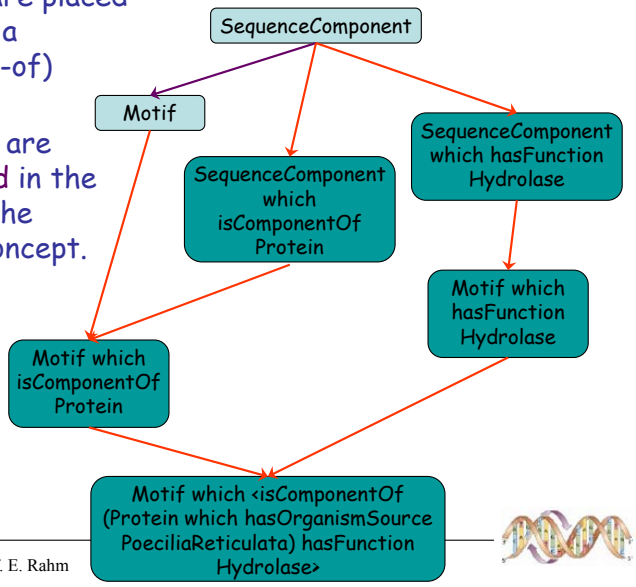
- TAMBIS Ontologie: Spezifikation mittels Description Logic (DL)
- Primitive Konzepte (atomic concepts)
  - z.B. *Protein*, *Motif*
- Rolen: Binäre Beziehung zwischen Konzepten
  - z.B. *hasOrganismSource*, *isComponentOf*
- Zusammengesetzte Konzepte (composite concepts): Kombination von Konzepten und Rolen
  - z.B. *Motif which isComponent of Protein*
- Sematische Datenintegration: Mapping der Begriffe zu Funktionen auf Quellen




```
< concept: Protein
            which hasFunction CatalysisProtein
            which catalyzes Reaction,
  function: < name: "get-all-enzyme-entries",
              arguments: [],
              resultType: "enzyme_record",
              cardinality: 5000,
              cost: 200,
              source: "Enzyme" >
>
```

Modelling Biology with DLs

- Primitive concepts are placed by the modeller into a subsumption (or kind-of) hierarchy.
- Composite concepts are automically classified in the hierarchy based on the description of the concept.



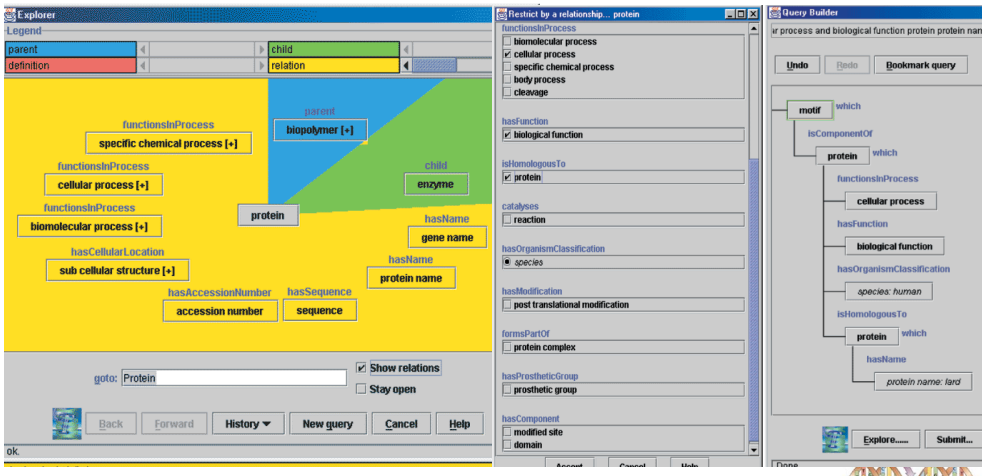
Modelling Biology with DLs

- The combination of concepts with roles is tightly controlled. We use these controls together with the classification to check the coherency of a concept.
- Two concepts are permitted to be related via some role through the use of sanctions. Composite concepts can't be formed without sanctioned permission.
  - Motif isComponentOf Protein
  - NucleicAcidComponent isComponentOf Protein
- Sanctions ensure that
  - only semantically valid compositions are formed;
  - a large number of compositions can be inferred from a sparse model.
- They also allow us to answer questions like "what can I say about this concept?"

TAMBIS - Abfragen

Query 1: Select motifs for antigenic human proteins that participate in apoptosis and are homologous to the lymphocyte associated receptor of death (also known as lard).

Translation: Select patterns in the proteins that invoke an immunological response and participate in programmed cell death that are similar in their sequence of amino acids to the protein that is associated with triggering cell death in the white cells of the immune system.



TAMBIS - Query-Verarbeitung

- Nutzerqueries: ontologie-basiert, quellunabhängig
- Mediator: Übersetzung der Nutzerabfragen zu quellspezifischen Query-Plänen (CPL)
- Wrappers: Kleisli, uniforme CPL-Sicht auf Quellen, Ausführung von Funktionen

(A) Concept expression in GRAIL:

```
Motif which
<isComponentOf (Protein which
  <hasOrganismClassification Species
    FunctionsInProcess Apoptosis
    HasFunction Antigen isHomologousTo
    Protein which <hasName
                    ProteinName>)>)>

Species: Is instantiated by value "human"
ProteinName: Is instantiated by value "lard"
```

(B) Equivalent expression in ALC standard Description Logic notation:

A ≡ Protein ⊓ ∃ hasName.ProteinName
B ≡ Protein ⊓ ∃ isHomologousTo.A
⊓ ∃ hasFunction.Antigen
⊓ ∃ functionsInProcess.Apoptosis
⊓ ∃ hasOrganismClassification Species

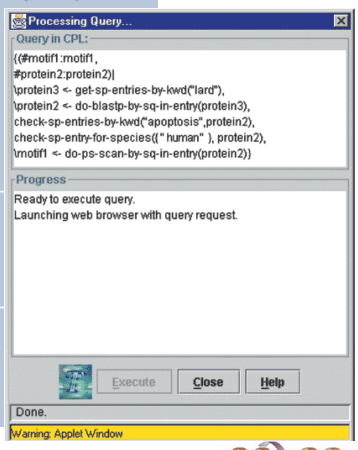
Motif ⊓ ∃ isComponentOf.B

(C) Informal query plan:

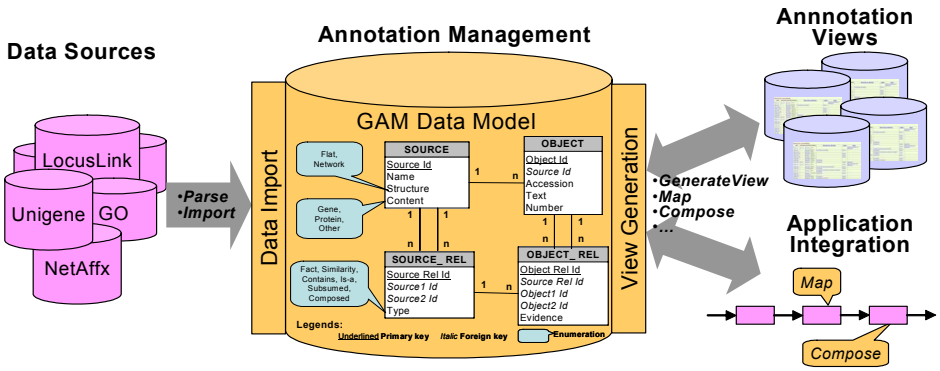
- Select proteins with protein name "lard" from SWISS-PROT
- Execute a BLAST sequence alignment process against SWISS-PROT results
- Check the entries for apoptosis process and antigen function
- Pass the resultant sequences to PROSITE to scan for their motifs

(D) CPL expression:

```
set-unique {(#motif1:motif1)|
\protein3 <- get-sp-entries-by-de("lard"), \protein2 <- do-blastp-by-sq-in-entry(protein3),
Check-sp-entries-by-kwd("apoptosis",protein2), check-sp-entries-by-de("antigen",protein2),
Check-sp-entry-for-species("human",protein2), \motif1 <- do-ps-scan-by-sq-in-entry(protein2)}
```



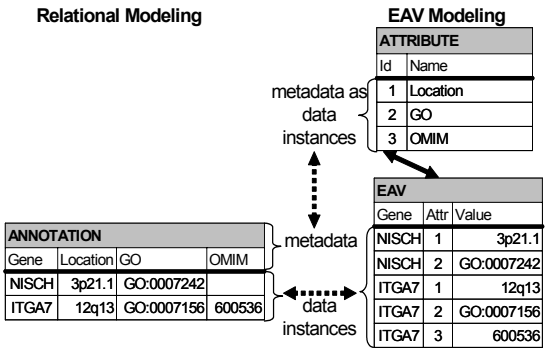
GenMapper Architektur



- Materialisierte Integration der Quelldaten in einer Zentraldatenbank
- Generisches Datenmodell zur einheitlichen Repräsentation der Objekte und Annnotations-daten aus verschienden Quellen, einschliesslich Ontologien
- Nutzung existierender semantischer Korrespondenzen für Datenintegration
- Flexible Generierung anwendungsspezifischer Annotationssichten mittels high-level Ope-rationen

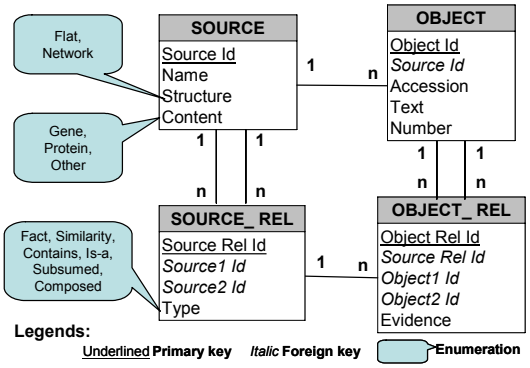
Generische Datenmodellierung

- Prinzip: *Entity-Attribute-Value*-Tripel (EAV)
  - Metadaten und Daten in einem Tupel
- Einheitliche Repräsentation unterschiedlicher Daten und Metadaten
  - Erweiterbarkeit
  - Evolution
  - Effiziente Speicherung
- Anwendungen
  - Repository: Verwaltung der Datenbankschemas unterschiedlicher Datenmodelle
  - E-Commerce: Verwaltung und Integration von elektronischen Katalogen
  - Krankenhausinformationsysteme: Verwaltung von unvollständigen Patientdaten
  - Semantic Web (RDF): Beschreibung und Austausch von Metadaten
  - ...



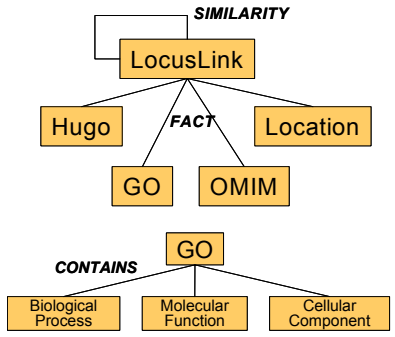
Generic Annotation Model (GAM)

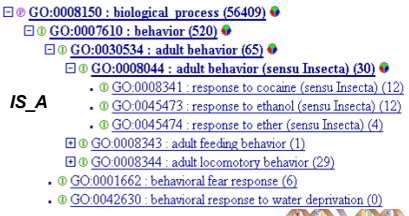
- *Source*: Gruppierung von Objekten
  - flach oder strukturiert
  - klassifiziert nach Inhalt
- *Object*: Einheitliche Attribute
  - Accession, Name
  - Beschreibung
  - ...
- *Relationship* / *Mapping*: Beziehungen
  - Zwischen Objekten und zwischen Quellen
  - Mit unterschiedlicher Semantik und Kardinalität
  - Innerhalb einer Quelle oder zwischen verschienden Quellen
  - *Evidence*: Ähnlichkeit zwischen 2 Objekten / Plausibilität ihrer Beziehung



GAM-basiertes Datenmanagement

- Sources (Quellen): Öffentliche Datenquellen oder Ontologien, Taxonomien, Vokabulare
- Objekte: Einträge einer öffentlichen Datenquelle oder Konzepte in einer Ontologie
- Annotationsbeziehungen
  - FACT: Bestätigte Annotationen
  - SIMILARITY: Berechnete Annotationen
- Strukturbeziehungen
  - CONTAINS: Partitionierung einer Quelle
  - IS_A: Semantische Struktur von Taxonomien
- Abgeleitete Beziehungen
  - SUBSUMES: abgeleitet von IS_A
  - COMPOSED: abgeleitet mittels Compose





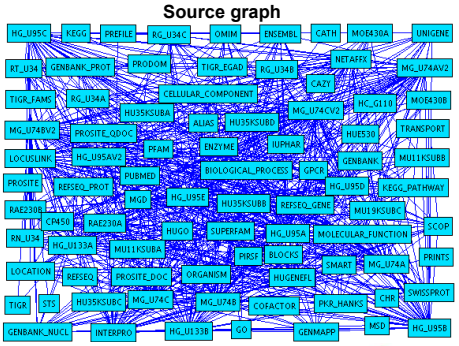
Datenintegration

- Integration in 2 Phasen: *Parse* und *Import*
- *Parse*: Transformation externer Quellen einheitlich nach EAV-Format
  - Einfache Abbildung der Web-Seiten öffentlicher Datenquellen
- *Import*: Generische EAV-zu-GAM Transformation und Migration
  - Duplikateliminierung durch Vergleich von Objekt-Ids, Quellennamen
  - Audit-Informationen, z.B. Datum, Release einer Quelle
- Schnelle Integration neuer Datenquellen
  - Einfache Konstruktion von Parsern für öffentliche Datenquellen

Pre-defined source names

Locus	Target	Accession	Text
353	Hugo	APRT	adenine phosphoribosyltransferase
353	Location	16q24	
353	Enzyme	2.4.2.7	
353	GO	GO:0009116	nucleoside metabolism
...	...	...	...

Source graph



Datenzugriff

- High-level Operationen: Vereinfachung des Zugriffs aufs generische Datenmodell
  - Anwendbar auf ganze Sources/Mappings oder Teilmengen

- Primitive Operationen

Operations	Definition	Example
Map(S, T)	Identify associations between S and T	map = *Map*(S, T) = {$s_1 \leftrightarrow t_1$, $s_2 \leftrightarrow t_2$}
Domain(map)	SELECT DISTINCT S FROM map	*Domain*(map) = {s_1, s_2}
Range(map)	SELECT DISTINCT T FROM map	*Range*(map) = {t_1, t_2}
RestrictDomain(map, s)	SELECT * FROM map WHERE S in s	*RestrictDomain*(map, {s_1}) = {$s_1 \leftrightarrow t_1$}
RestrictRange(map, t)	SELECT * FROM map WHERE T in t	*RestrictRange*(map, {t_1}) = {$s_1 \leftrightarrow t_1$}

map = *Map*(Unigene, Go)

UNIGENE	GO	GO_evidence
Hs.183803	GO:0005739	IEA
Hs.183803	GO:0008372	ND
Hs.183803	GO:0005524	IEA
Hs.183803	GO:0005164	NAS
Hs.183803	GO:0003754	IEA
Hs.183803	GO:0000004	ND
Hs.183803	GO:0006457	IEA
Hs.194691	GO:0016020	IEA
Hs.194691	GO:0005887	TAS
Hs.194691	GO:0004872	IEA
Hs.194691	GO:0008067	IEA

Domain(map)

UNIGENE	UNIGENE_text_rep
Hs.183803	heat shock protein 75
Hs.194691	retinoic acid induced 3
Hs.20084	retinoid X receptor, alpha
Hs.202453	v-myc myelocytomatosis viral oncogene homolog (avian)
Hs.343586	zinc finger protein 36, C3H type, homolog (mouse)
Hs.410597	cytokine induced protein 29 kDa
Hs.411125	mitochondrial ribosomal protein S12
Hs.435290	nischarin
Hs.511945	block of proliferation 1
Hs.74369	integrin, alpha 7

Range(map)

GO	GO_text_rep
GO:0000004	biological_process unknown
GO:0006118	electron transport
GO:0006350	transcription
GO:0006355	regulation of transcription, DNA-dependent
GO:0006357	regulation of transcription from Pol II promoter
GO:0006402	mRNA catabolism
GO:0006412	protein biosynthesis
GO:0006417	regulation of protein biosynthesis
GO:0006445	regulation of translation
GO:0006457	protein folding
GO:0006766	vitamin metabolism

Ableitung neuer Annotationen

- *Compose*: Ableitung neuer Mappings von existierenden
  - Transitivität der Beziehungen (Cross-References): $(a \leftrightarrow b, b \leftrightarrow c) \Rightarrow (a \leftrightarrow c)$
  - Eingabe: Mapping-Pfad S1↔S2↔...↔Sm
  - Ausgabe: Mapping S1↔Sm
- Nutzung
  - Generierung von Annotationen wenn nicht verfügbar
  - Überprüfung/Vergleich von Annotationen aus unterschiedlichen Quellen
  - Qualität/Plausibilität der Transitivität?
- Bestimmung von Mapping-Pfaden
  - Automatisch mit einem Shortes Path-Algorithmus
  - Manuelle Konstruktion durch Nutzer

Compose($S_1 \leftrightarrow S_2 \leftrightarrow ... \leftrightarrow S_m$)

```
map = Map(S1, S2)
For i = 2..m-1
   next = Map(Si, Si+1)
   new = SELECT A.S1, B.Si+1
         FROM map A, next B
         WHERE A.Si = B.Si
   map = new
End for
```

Compose(Unigene↔LocusLink↔GO)

UNIGENE	LOCUSLINK	GO
Hs.183803	10131	GO:0000004
Hs.183803	10131	GO:0006457
Hs.183803	10131	GO:0005739
Hs.183803	10131	GO:0008372
Hs.183803	10131	GO:0005524
Hs.183803	10131	GO:0005164
Hs.183803	10131	GO:0003754
Hs.194691	9052	GO:0007165
Hs.194691	9052	GO:0016020
Hs.194691	9052	GO:0005887
Hs.194691	9052	GO:0004872
Hs.194691	9052	GO:0008067
Hs.20084	6256	GO:0007165

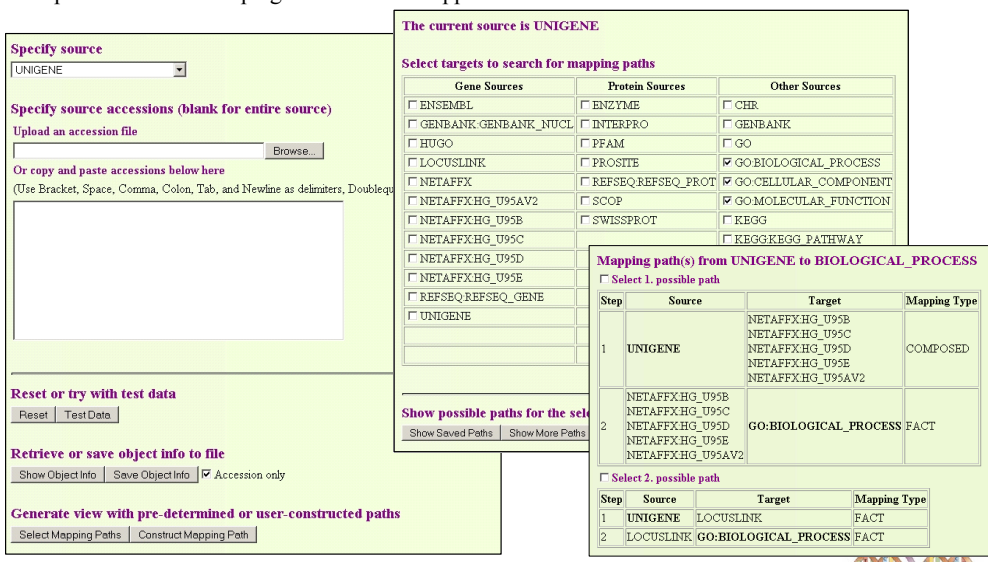
Generierung von Annotationssichten

- Abfragen zur Analyse der Objektbeziehungen
  - *Finde Locuslink-Gene, die auf Chromosom 12 liegen UND bestimmte GO-Funktionen besitzen, ABER NICHT mit bestimmten OMIM-Krankheiten im Zusammenhang stehen*
- Typische Query-Struktur:
  - Mappings zwischen einer Quelle (z.B. LocusLink) und mehreren Zielen (z.B. GO, OMIM)
  - Quelle und Ziele einschränkbar auf eine Submenge von relevanten Objekten
  - Kombination der Mappings mit AND oder OR, Negation von Mappings, z.B. LocusLink↔OMIM
- *GenerateView*: Unterstützung häufige Analysen und Abfragen

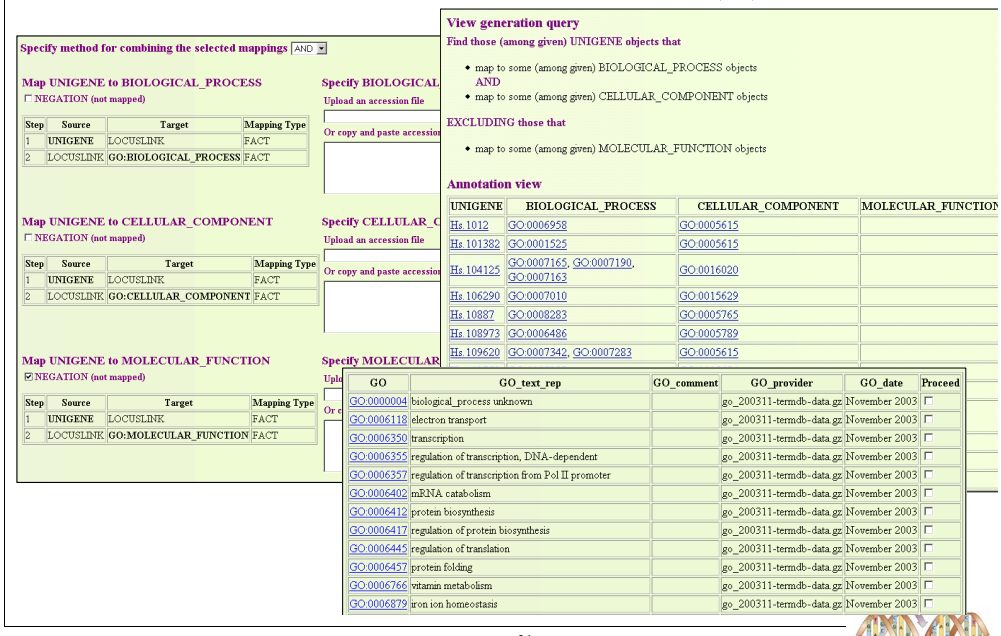
GenerateView(S, s, T1, t1, ..., Tm, tm, [AND|OR], {negated})
V = s //Start with all given source objects
For i = 1..m
  Determine mapping M_i: S↔T_i //Using either the *Map* or *Compose* operation
  m_i = *RestrictDomain*(M_i, s) //Consider the given source and target objects
  m_i = *RestrictRange*(m_i, t_i)
  If negated[i]
    s_i = s \ *Domain*(m_i) //Source objects not involved in the sub-mapping
    m_i = *RestrictDomain*(M_i, s_i) //Find associations for these objects
    m_i = m_i *right outer join* s_i *on* S //Preserve objects without associations
  End if
  V = V *left outer join / inner join* mi *on* S //OR: left outer join, AND: inner join
End for

Nutzerschnittstelle (1)

- http://sun1.izbi.uni-leipzig.de:8080/GenMapper/



Nutzerschnittstelle (2)



Specify method for combining the selected mappings: AND

Map UNIGENE to BIOLOGICAL_PROCESS

NEGATION (not mapped)

Step	Source	Target	Mapping Type
1	UNIGENE	LOCUSLINK	FACT
2	LOCUSLINK	GO:BIOLOGICAL_PROCESS	FACT

Specify BIOLOGICAL

Upload an accession file

Or copy and paste accession

Map UNIGENE to CELLULAR_COMPONENT

NEGATION (not mapped)

Step	Source	Target	Mapping Type
1	UNIGENE	LOCUSLINK	FACT
2	LOCUSLINK	GO:CELLULAR_COMPONENT	FACT

Specify CELLULAR_C

Upload an accession file

Or copy and paste accession

Map UNIGENE to MOLECULAR_FUNCTION

NEGATION (not mapped)

Step	Source	Target	Mapping Type
1	UNIGENE	LOCUSLINK	FACT
2	LOCUSLINK	GO:MOLECULAR_FUNCTION	FACT

Specify MOLECULAR

View generation query

Find those (among given) UNIGENE objects that

- map to some (among given) BIOLOGICAL_PROCESS objects

AND

- map to some (among given) CELLULAR_COMPONENT objects

EXCLUDING those that

- map to some (among given) MOLECULAR_FUNCTION objects

Annotation view

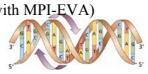
UNIGENE	BIOLOGICAL_PROCESS	CELLULAR_COMPONENT	MOLECULAR_FUNCTION
Hs.1012	GO:0006958	GO:0005615	
Hs.101382	GO:0001525	GO:0005615	
Hs.104125	GO:0007165, GO:0007190, GO:0007163	GO:0016020	
Hs.106290	GO:0007010	GO:0015629	
Hs.10887	GO:0008283	GO:0005765	
Hs.108973	GO:0006486	GO:0005789	
Hs.109620	GO:0007342, GO:0007283	GO:0005615	

GO	GO_text_rep	GO_comment	GO_provider	GO_date	Proceed
GO:0000004	biological_process unknown		go_200311-termdb-data.gz	November 2003	
GO:0006118	electron transport		go_200311-termdb-data.gz	November 2003	
GO:0006350	transcription		go_200311-termdb-data.gz	November 2003	
GO:0006355	regulation of transcription, DNA-dependent		go_200311-termdb-data.gz	November 2003	
GO:0006357	regulation of transcription from Pol II promoter		go_200311-termdb-data.gz	November 2003	
GO:0006402	mRNA catabolism		go_200311-termdb-data.gz	November 2003	
GO:0006412	protein biosynthesis		go_200311-termdb-data.gz	November 2003	
GO:0006417	regulation of protein biosynthesis		go_200311-termdb-data.gz	November 2003	
GO:0006445	regulation of translation		go_200311-termdb-data.gz	November 2003	
GO:0006457	protein folding		go_200311-termdb-data.gz	November 2003	
GO:0006766	vitamin metabolism		go_200311-termdb-data.gz	November 2003	
GO:0006879	iron ion homeostasis		go_200311-termdb-data.gz	November 2003	

Anwendung: Gene Functional Profiling*

- Vergleichende Analyse zwischen Menschen und Schimpanzen (MPI Projekt)
  - Genexpressionmessungen mittels Affymetrix Microarrays
  - 40.000 Gene untersucht: 20.000 identifiziert mit Expression, 2.500 mit signifikanten Änderungen im Expressionsmuster zwischen zwei Spezies
- Funktionelle Analyse der exprimierten und unterschiedlich exprimierten Gene
  - Bestimmung der konservierten bzw. geänderten Genfunktionen zwischen den Spezies
- Automatisches Verfahren für Analyse grosser Genmengen notwendig
  - Abbildung von Affymetrix Probesets zur generell akzeptierten Genrepräsentation Unigene, Elimi-nierung von duplikaten Probesets
  - Ableitung der GO-funktionen für UniGene-Clustern (nicht verfügbar in UniGene)
  - Nutzung der Struktur der GO-Taxonomien für statistische Analysen
- Ansatz anwendbar auf andere Genrepräsentationen (Hugo, LocusLink, etc.) und andere Annotationen (InterPro, Enzyme, etc.)

* Khaitovich et al: *Evolution of Gene Expression in the Primate Brain*. Submitted (Joint-work with MPI-EVA)
Mützel et al: *Functional Analysis of Gene Expression Data Using the Gene Ontology Database*. In preparation (Joint-work with MPI-EVA)



Zusammenfassung

- Traditionelle Datenintegrationsansätze eingeschränkt einsetzbar in Bioinformatik
- Trade-off zwischen (Schema- und Instanz-) Konsistenz und Skalierbarkeit, Flexibilität
- TAMBIS:
  - Mediator, semantische Integration auf Schema- und Instanzebene
- DiscoveryLink, Kleisli:
  - Mediator, globales Schema als Vereinigung der lokalen Schemas
  - Keine Integration der Instanzdaten
- SRS:
  - Mediator, kein globales Schema, sondern Mengen abfragbarer Attribute, IR-basierte Indexierung
    und Suche in einzelnen Quellen
  - Nutzung der Verweisen zur Navigation zwischen Datenquellen
- GenMapper: Forschungsprototyp
  - Materialisierte Integr., kein globales Schema, sondern generisches Datenmodell
  - Flexible View-Generierung, Nutzung der Verweise zur semantischen Integration der Instanzdaten