



APPTUS

Invited lecture
Database tuning, ITU
2008-04-23

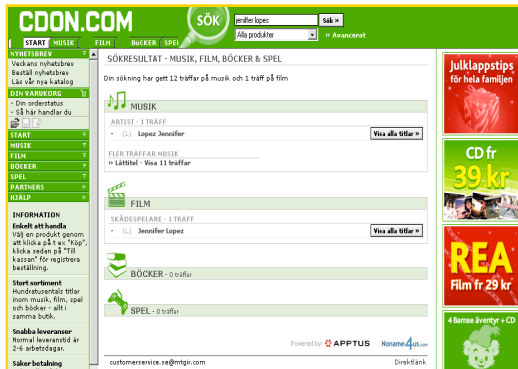
Apptus Technologies



- **Founded 2000 with HQ in Lund, Sweden.**
- **Currently around 40 employees.**
- **Specialized in search and high-performance information retrieval.**
- **We design and develop solutions for world-leading e-commerce and e-directory companies with the highest of demands.**



Apptus in e-commerce



MTG IR/ CDON.COM - leader in e-commerce in Scandinavia

- 4.2 million searches per day
- Search driven navigation
- Increased revenue by 30%
- Cross sales and behavior analysis



Bokus.com - largest bookstore in Scandinavia

- 4 million titles
- Personalization and associative search
- Reduced development cost

Apptus in e-directory

Relevant categories with number of existing ads in each category

Avgränsa

Kategori

- Alla kategorier (427)
- [Antikt, konst & klockor](#) (1)
- [Bilar](#) (13)
- [Datorer](#) (14)
- [Digitalkameror & foto](#) (1)
- [Hemelektronik](#) (380)
- [Mobiler & telefoner](#) (9)
- [Motor & övriga fordon](#) (1)
- [Musik, film & spel](#) (7)
- [Sport & friluftsliv](#) (1)

Support for zoomed filtering

Förfina sökningen

Pris:

 till

Filtering on geographic areas and neighborhoods

Område

- Hela landet (427)
- [Dalarna](#) (3)
- [Gävleborg](#) (8)
- [Holland](#) (3)
- [Jönköping](#) (8)
- [Kalmar](#) (7)

Relevant listings in Personal and Company sections


Annonsör

- Alla (427)
- [Privatperson](#) (171)
- [Företag](#) (256)

Företag i Gula Sidorna

- [ComputerCity](#)

Säljes, 427 annonser: ipod

Bevaka denna sökning via [e-post](#) eller [RSS](#) 

Köpes (9) Säljes (427)

Visa: [utan bilder](#) med bilder

Bild	Datum	Annons	Pris
	13/2	iPhone 1.1.3 Olåst Hej, Jag säljer min 2 månader gamla iPh... Kategori: Mobiltelefoner Län: Skåne	4 300:-
	12/2	Ipod nano 2gb! Säljes Ipod nano 2gbMedföljer gör USB s... Kategori: MP3-spelare & bärbart Län: Västra Götaland	800:-
	12/2	Belkin TuneCast II FM-sändare (F) Lyssna på din MP3-spelare genom FM-stat... Kategori: MP3-spelare & bärbart Län: Västra Götaland	279:-
	12/2	nästan ny iPod nano Köpte för 3-4 måndader sedan en turkos ... Kategori: MP3-spelare & bärbart Län: Jönköping	1 000:-
	12/2	iPod nano 8GB silver oanvänd iPod nano 8GB, ny oanvänd i orginalförp... Kategori: MP3-spelare & bärbart Län: Stockholm	900:-
	11/2	Snygg BLÅ iPod Nano 4GB 2 generation / ny Ny iPod Nano Blue 8 gb i orginalförpac... Kategori: MP3-spelare & bärbart Län: Stockholm	500:-
	8/2	xbox 360 / PSP / playstation 3 ps3 / Wii ps2 Du kan köpa bla:Canon digital kamera, L... Kategori: TV-spel Län: Västra Götaland	49:-
	8/2	Helt Ny Ipod Shuffle I Oöppnad Förpackning Denna mp3-spelaren är ljustblå, och sälj...	550:-

DBMS Platform Not Our Business APPTUS

- We sell solutions/products for (e.g.) e-business and e-directory.
- Tools for developing data management systems is *not* our business, we developed the tools for our own use.
- However, we do find the discussion about future of DBMS interesting, and would like to spread our ideas, and talk to others with similar ideas.

1. Current DBMS state of affairs, trends
2. Problems in traditional DBMS/search engine setup
3. Glimpse of Apptus solution
4. Technology journey, Apptus 2000–2009
5. The relational *model*
6. Apptus Theca: a database and search engine platform

Very little development in database systems for 30 years.

While other fields are always full of new ideas and approaches, the SQL DBMS setup is static and taken for granted.

Development since 1974

- Hardware conditions completely different
 - Gigabytes of RAM, CPU speed ~10 000 times higher.
- New programming languages/environments/paradigms
 - Continuous development
- Enormous increase in data availability and use
- New kinds of data
- New algorithms/data structures

- DBMS: still the same architecture (System R)

Time for Complete Rewrite?



What I see happening is that the large database vendors, whom I'll call the elephants, are selling a one-size-fits-all, 30-year-old architecture that dates from somewhere in the late 1970s.

Michael Stonebraker, 2007

The DBMS vendors (and the research community) should start with a clean sheet of paper and design systems for tomorrow's requirements, not continue to push code lines and architectures designed for yesterday's needs.

Michael Stonebraker et al, sep 2007

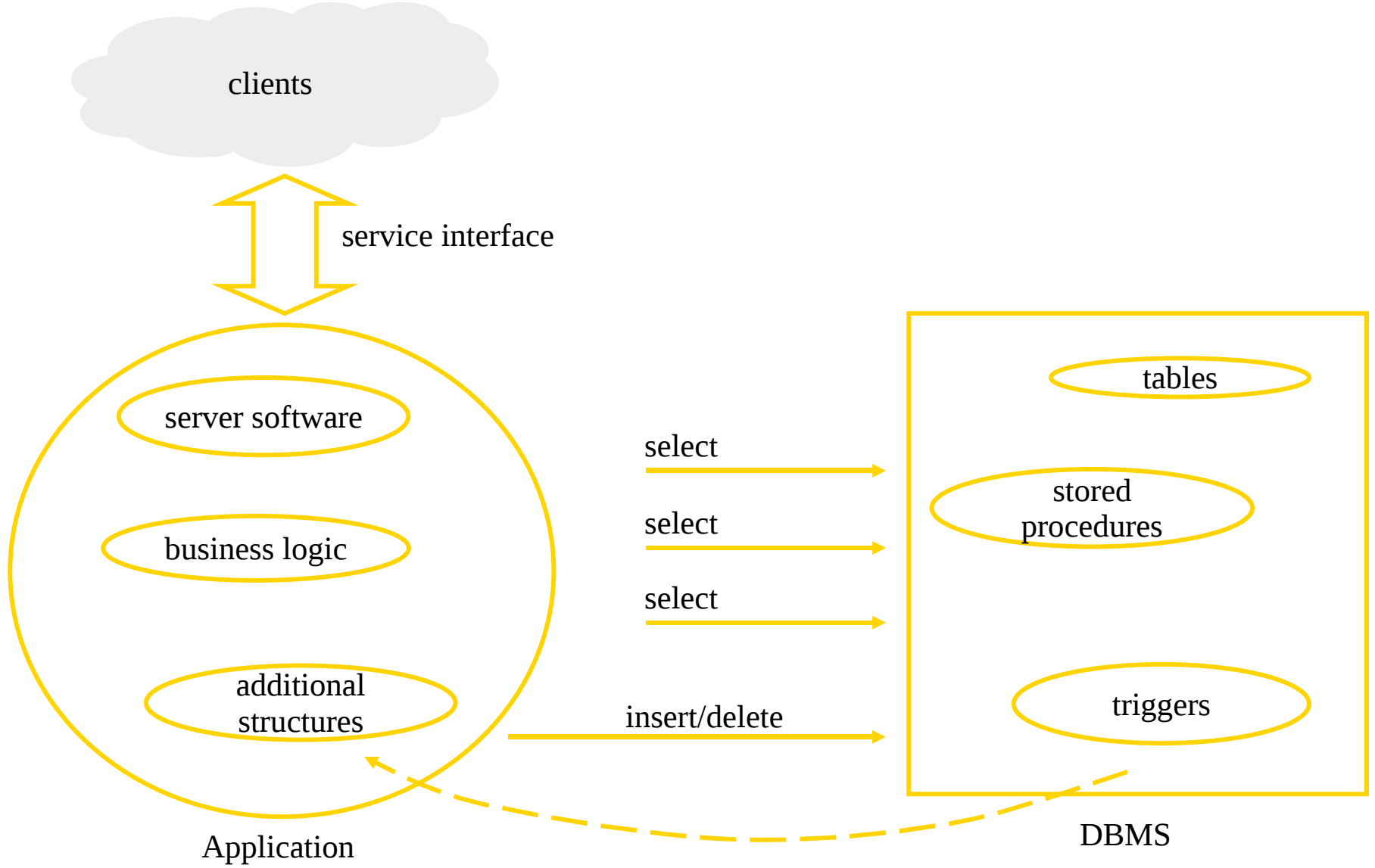
It is a unique opportunity for a fundamental "reformation" of the notion of data management: not as a storage device, but as a broadly applicable programming paradigm.

The Claremont Report on Database Research, aug 2008

- Abandonment of RDBMS
- Ad hoc homegrown tools
- Reverting to (bad) 1960s ideas: tree-structured (XML), record-oriented (objects, networks)

- Date et al.: rescue the relational model
 - Specify complete system base on relational algebra and Pascal-like programming language
- Stonebraker et al.: Specialized tools
 - Create various systems/toolboxes for various needs
- Apptus: give the DBMS a chance
 - Simple logical UI, give DBMS freedom to optimize

The Traditional Setup



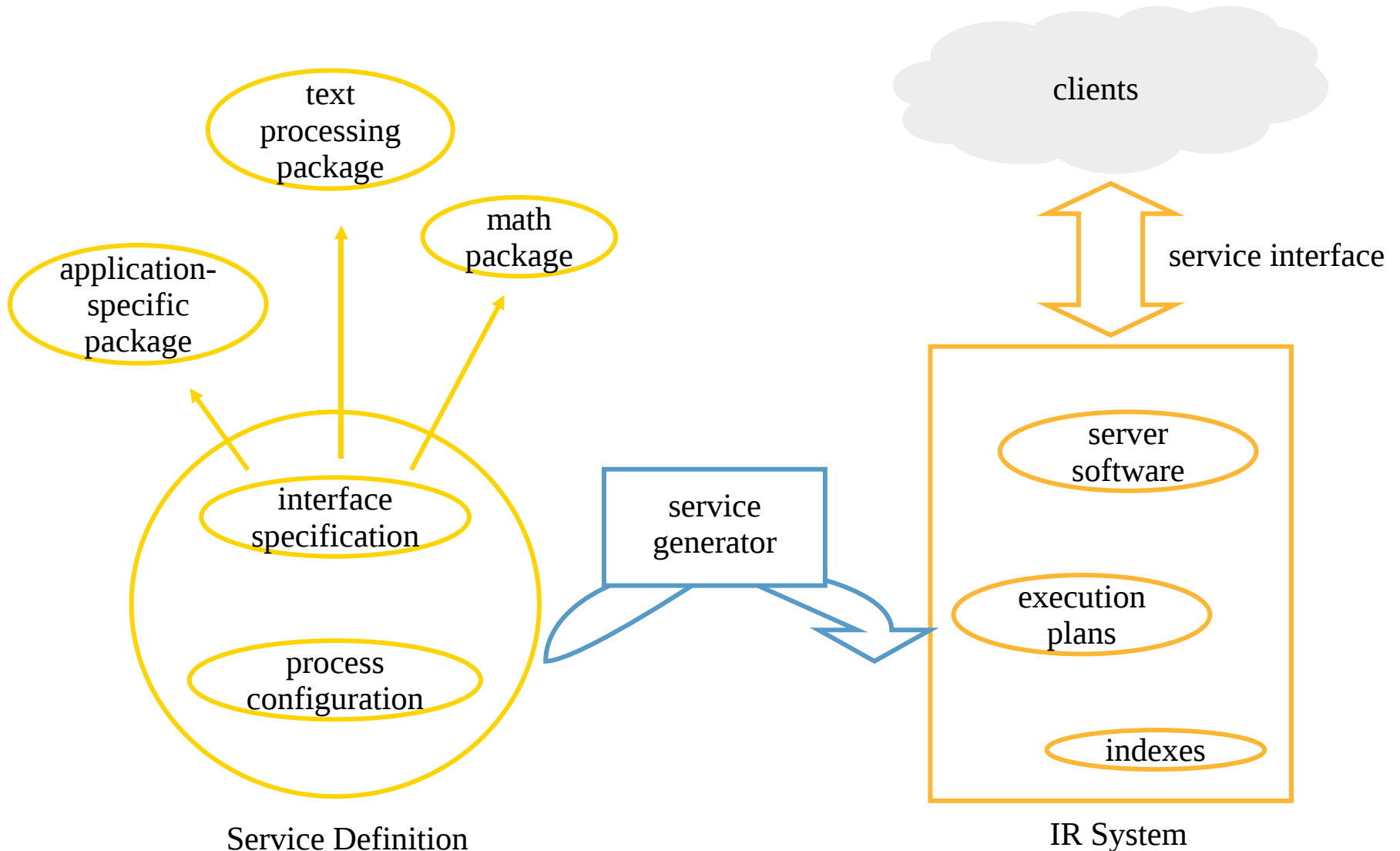
- Lots of small, expensive, requests to DBMS (ripple loading)
- DBMS optimization only on local per-query basis
 - Don't know what's coming (dynamic)
 - No access to business logic
 - Stored procedure and trigger code not comprehensible
- Mainstream DBMS are colossal monoliths
 - *Steers like a cow* – does not follow development in needs or technology (hardware, algorithms)
 - Plug-ins are really “plug-on-tops”

Making optimization possible



- Define business logic in language of DBMS
 - Language must be powerful and simple, not SQL
- Specify Queries ahead of time, and compile
 - DBMS can choose implementation (indexes etc.)
- Provide extensibility inside DBMS
 - Data structures and access methods defined with an API that the DBMS can invoke

Future Setup, Apptus Style

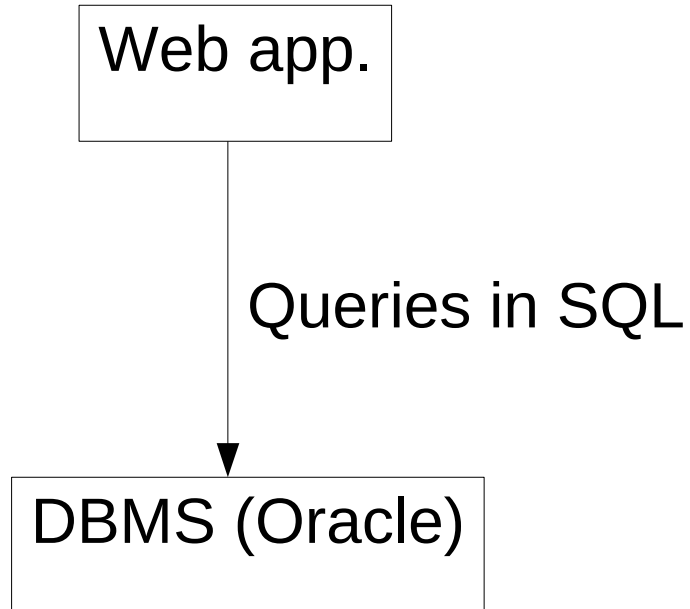


Technology Retrospective

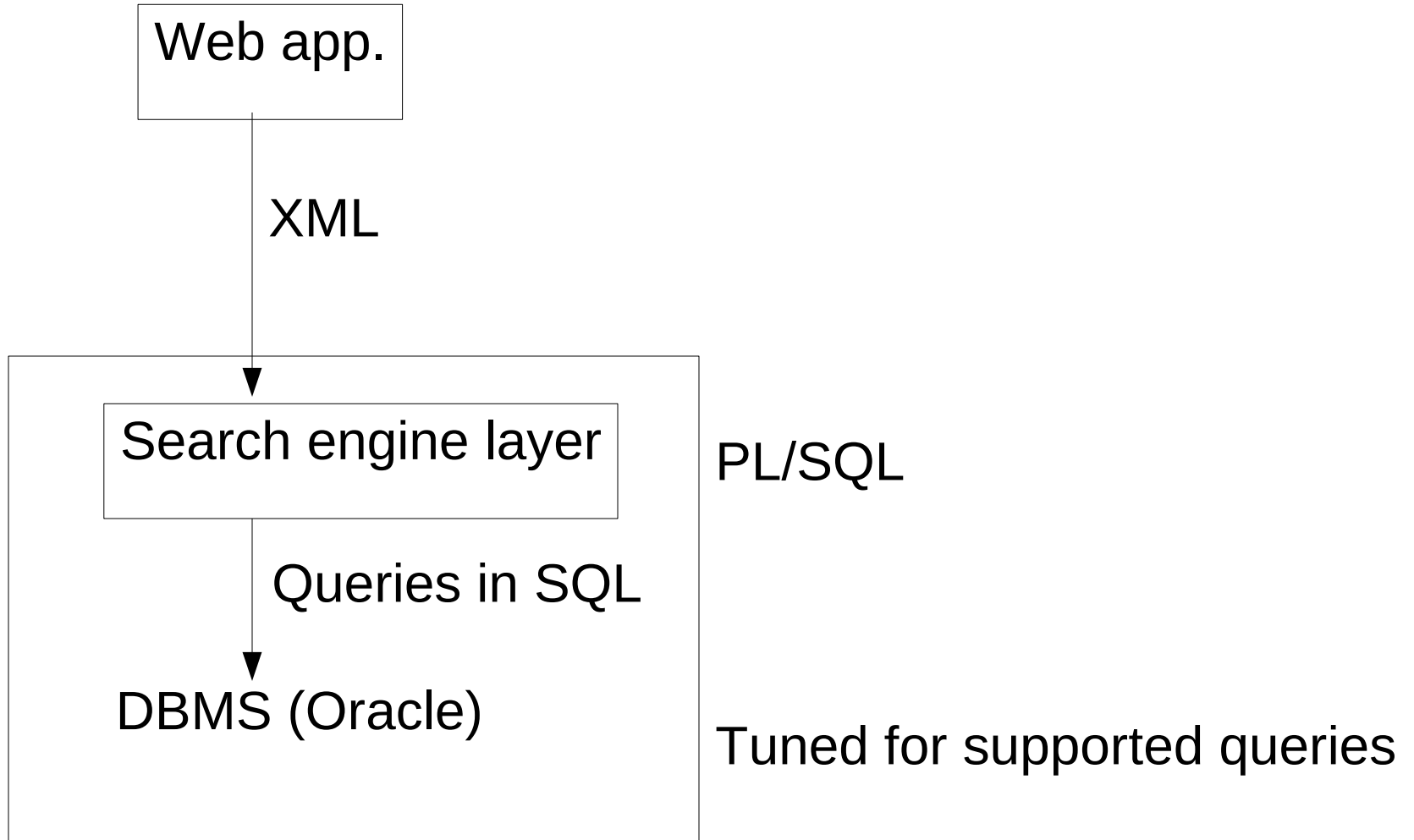


- Apptus has offered (among other things) search engine solutions since the year 2000
- Now at fifth step in our technological evolution

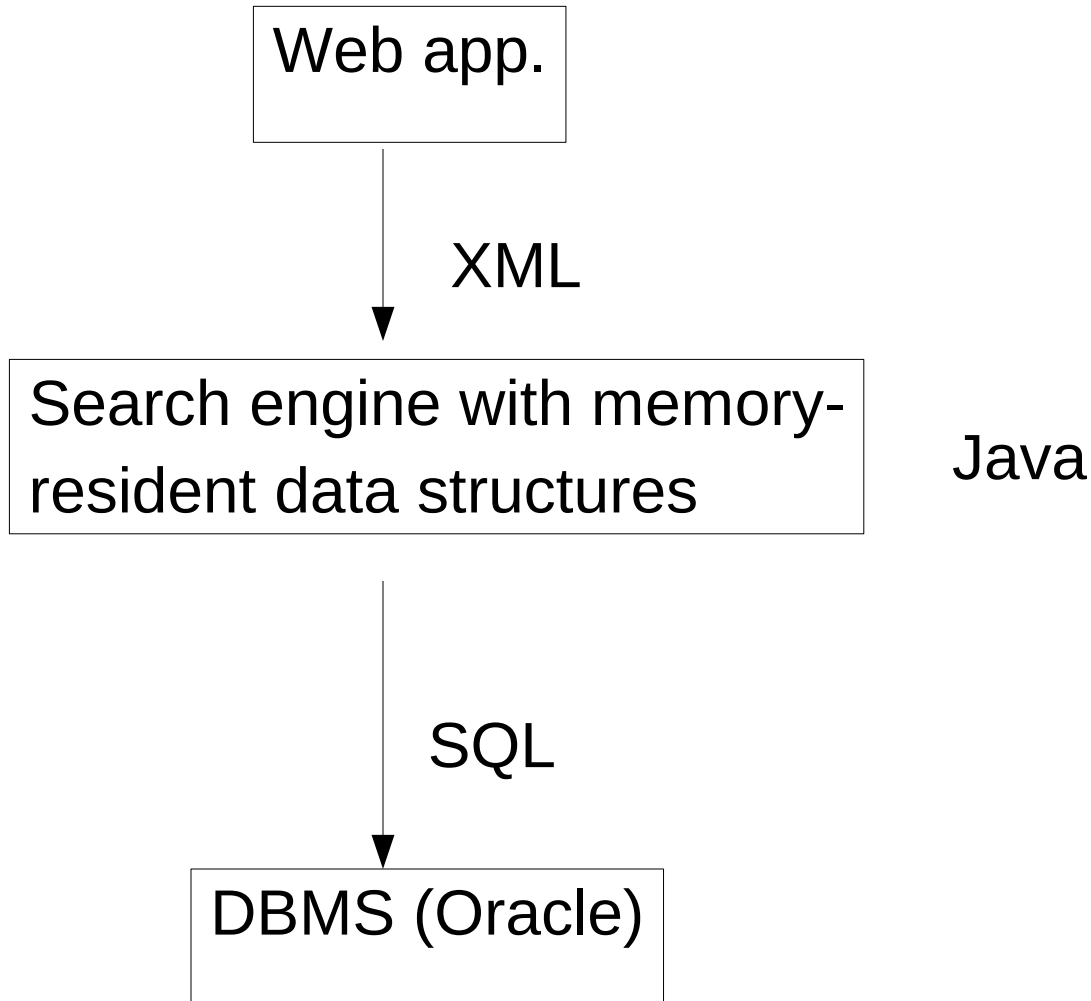
Setup 1 (2000)



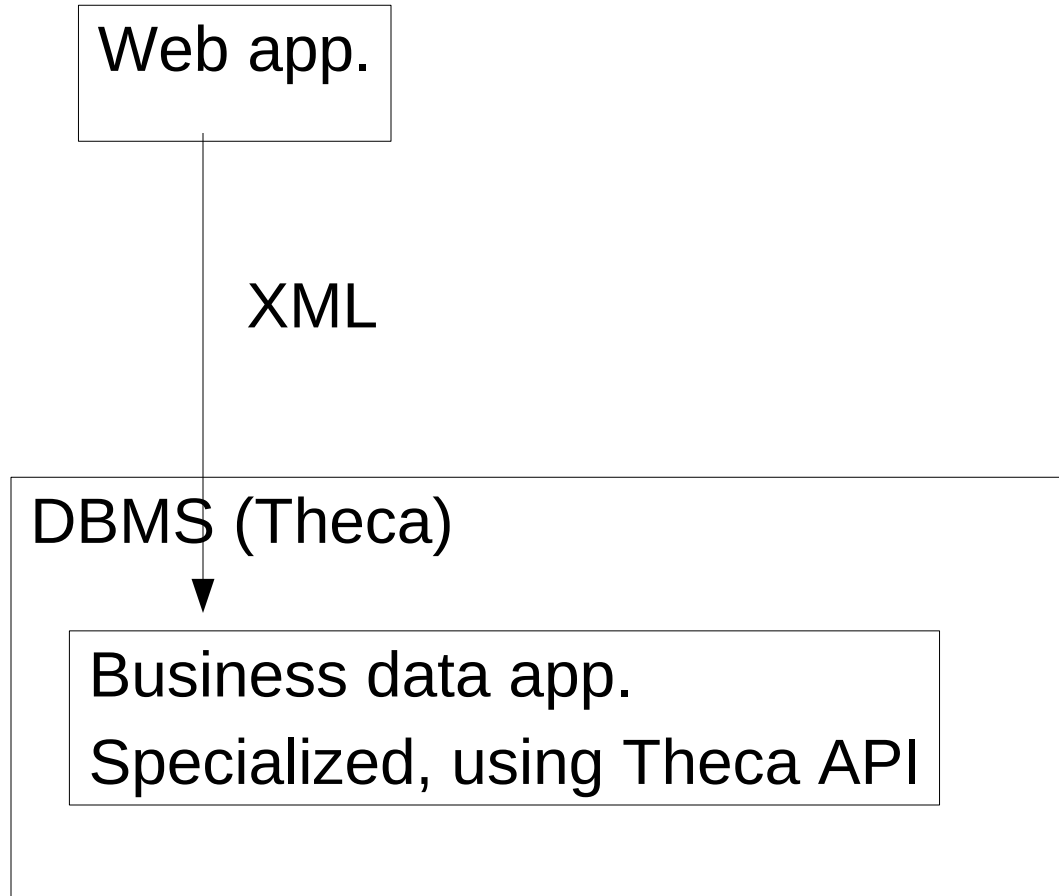
Setup 2 (2001)



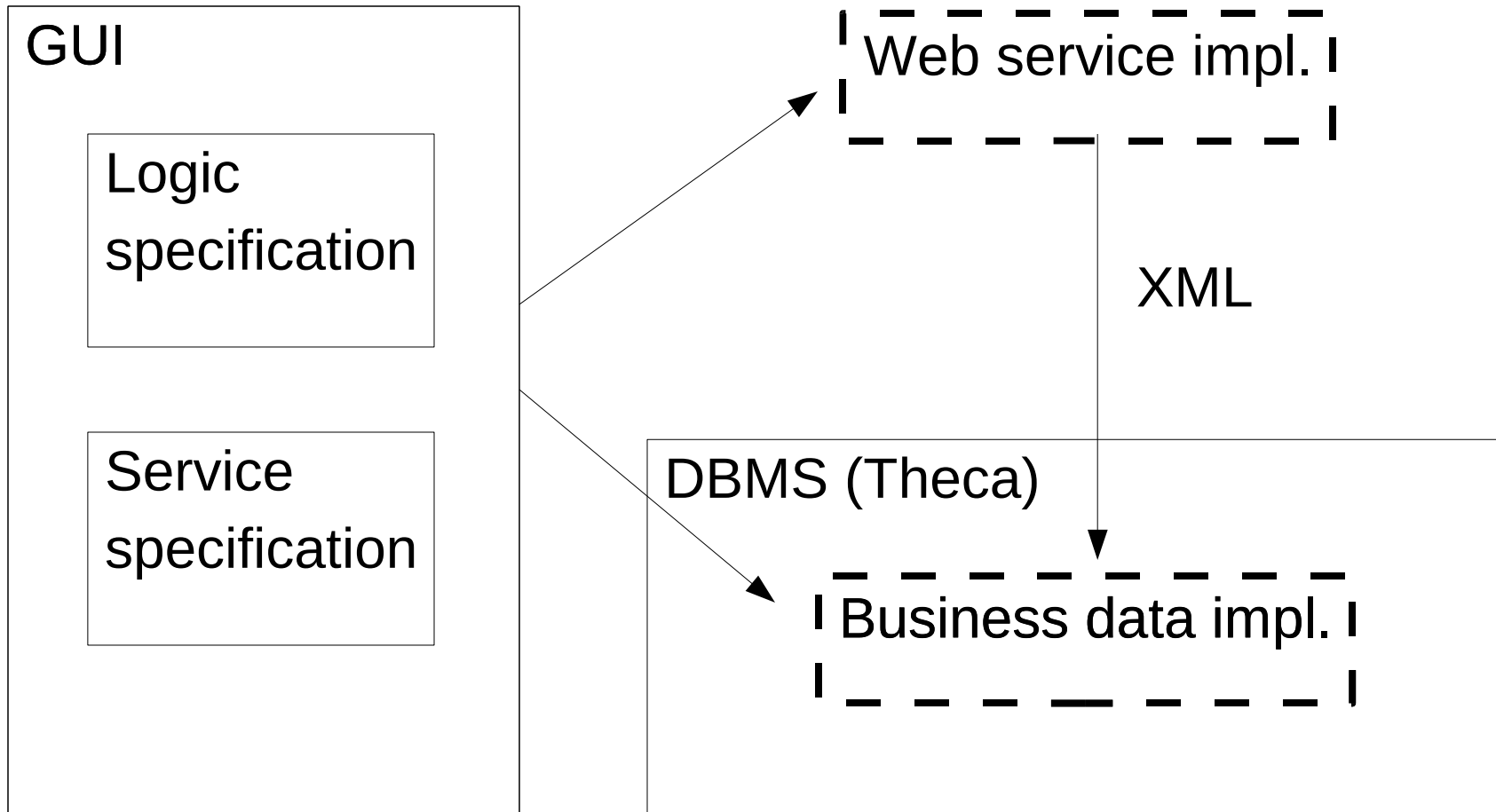
Setup 3 (2003)



Setup 4 (2005)



Setup 5 (2008)



Language: back to basics



- The relational data model & the model-implementation misconception
- Logic and the relational model
- A graphical interface based on logic

What is a model?



- Simple abstract representation of something, often reusing concepts from other domains
- Framework for reasoning
- A good model helps us see clearly, draw conclusions
- A bad model obscures and confuses our thoughts

“In the relational model, data are stored as tables”

- The model says nothing of storage
- Tables are just a way of drawing relations

Correct:

- In the relational model, data are perceived as relations, which may be *visualized* as tables.

Most DBMS centered on one kind of storage structure

Statement 1:

- “the relational model is not well suited to represent the structure of text”

Statement 2:

- “Observations on inverted files: Can be implemented directly in the relational model”
- Both true?
- Any true?

- One data representation:

"Observations on inverted files: Can be implemented directly in the relational model"

- Another:

```
{ (Observations, 1), (on, 2),  
  (inverted, 3), (files, 4), (:, 5),  
  (Can, 6), (be, 7), (implemented, 8),  
  (directly, 9), (in, 10), (the, 11),  
  (relational, 12), (model, 13) }
```

Text and relations

Id	Title
1	Art of war
2	War and peace
3	Modern art

Book titles

Word	Id
art	1, 3
war	1, 2
modern	3
peace	2

Inverted index

Word	Id
art	1
art	3
war	1
war	2
modern	3
peace	2

Relational form

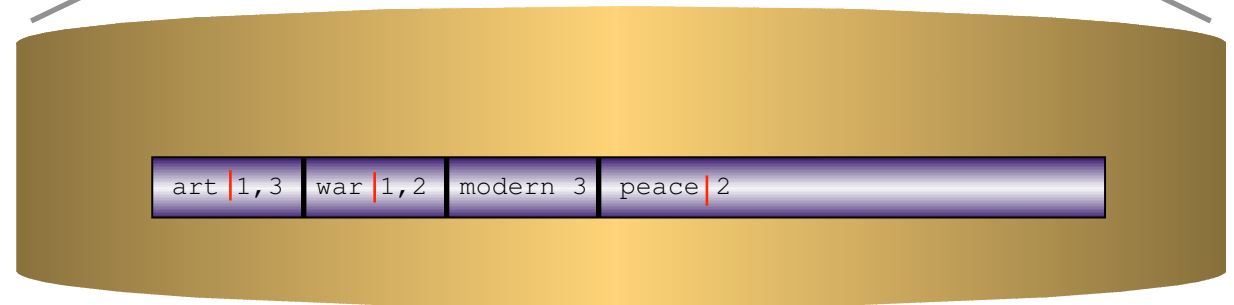
Relational model inefficient for storing text index?

Separation of layers

Logical representation

Word	Id
art	1
art	3
war	1
war	2
modern	3
peace	2

Physical representation



“Freetext” data structure



B-tree with:

- key “word”
- data either
 - id list (for short lists, less than ~100 ids)
 - root block identifier for another B-tree, which contains ids

“Freetext” data structure



- Increasing integers: append to id lists/trees only (fill blocks completely)
- No deletions: because it is very difficult to make efficient
- Joining with document relation filters out deleted records
- Update a record: delete and insert with higher id

Rel. model not based on logic



- Core idea: all data are represented as relations, but...
- Relational algebra: directional operations. Not naturally declarative.
- Plethora of operations, very far from minimal set.
- Types, metadata, constraints: other forms of data entered by the user.

Deductive databases

- Accessing database with logical language. Prolog, datalog, variants.
- AI, logic programming
- Not generally known among database researchers and practitioners
- Died out in the 1990s

Relational algebra and logic



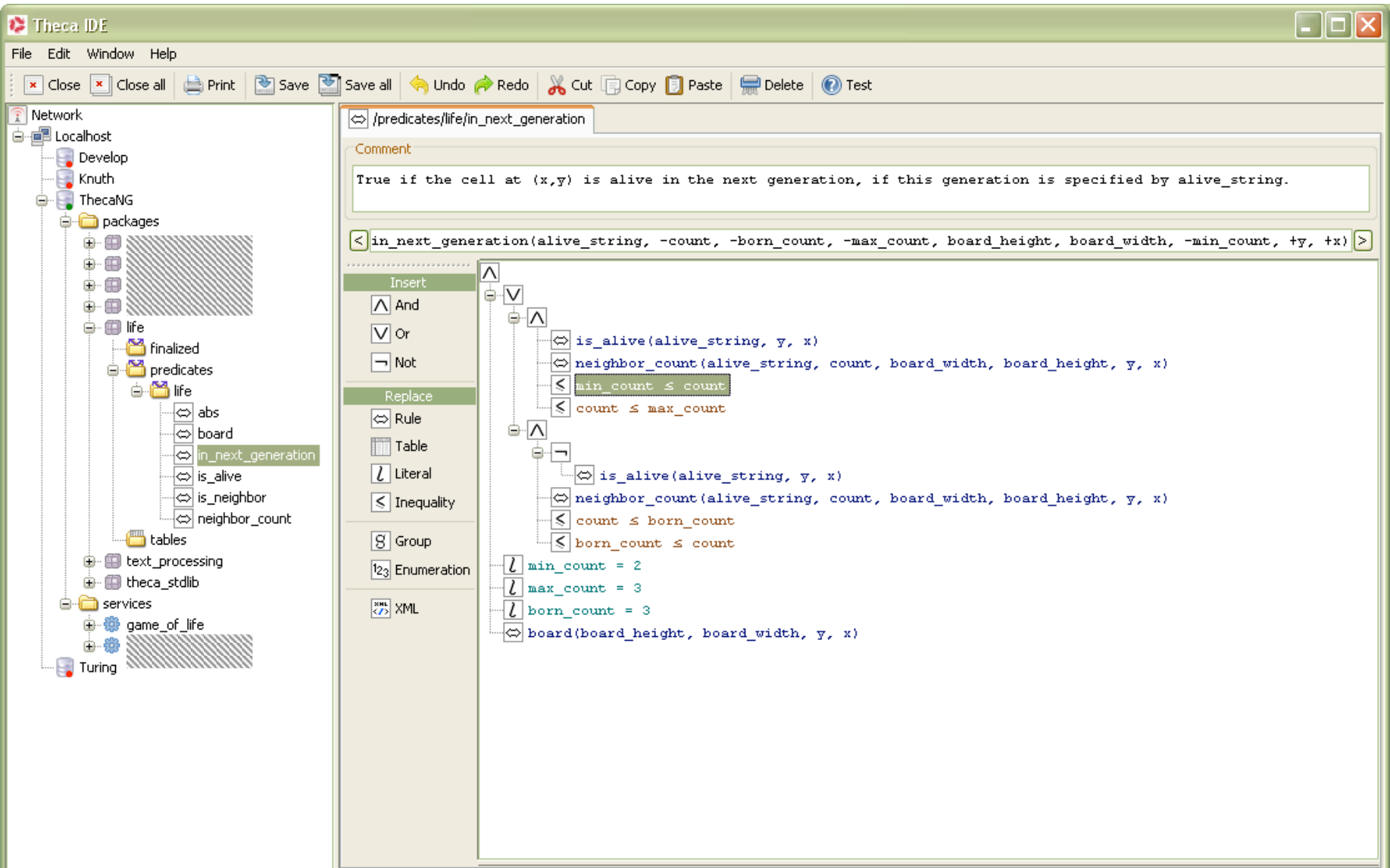
Rel. Operation	SQL example	FOL symbol	FOL example
Join	<code>(select * from t) natural join (select * from u)</code>	\wedge	$t(x, y) \wedge u(x, z)$
Cartesian product	<code>select * from t, u;</code>	\wedge	$t(x, y) \wedge u(z, w)$
Restriction	<code>select * from t where x < 5;</code>	\wedge	$t(x, y) \wedge x < 5$
Projection	<code>select x from t;</code>	$\exists x$	$\exists y t(x, y)$
Rename	<code>select x as a, y from t;</code>	<i>actual arguments</i>	$t(a, y)$
Union	<code>(select * from t) union (select * from u)</code>	\vee	$t(x, y) \vee u(x, y)$
Semidifference	<code>(select * from t) except (select * from u)</code>	$\wedge \neg$	$t(x, y) \wedge \neg u(x, y)$
Extension	<code>select add(x, 1) as y from t;</code>	\wedge	$t(x) \wedge \text{sum}(x, 1, y)$

What is Theca?

- A logic DBMS.
- Includes
 - server software for web services,
 - a fully ACID transaction manager,
 - meta data accessible with ordinary queries,
 - a JDBC interface with an SQL-like language (RQL),
 - cluster software for distributed systems.
- Written entirely in java.
- Pluggable.
- Built for high-performance applications.
- Comes with a graphical development tool (Theca IDE).

- Graphical programming language based on first-order logic.
- Predicate: definitions of logic that can be re-used.
 - System predicates, written in java
`lowercase(x, lc_x).`
 - Rule predicates, defined by first-order logic
`fahrenheit(c, f) \Leftrightarrow mul(c, 1.8, prod) \wedge add(32, prod, f).`
 - Table predicates, modifiable by clients during run-time
`person(id, name, number_of_ears).`
- Packages: libraries of predicates and supporting algorithms and data structures.
- Web functions: published predicates with specified input and output arguments.
- Services: Database instances with web functions.

Game of life example



Theca IDE

File Edit Window Help

Close Close all Print Save Save all Undo Redo Cut Copy Paste Delete Test

Network

- Localhost
 - Develop
 - Knuth
 - ThecaNG
 - packages
 - life
 - finalized
 - predicates
 - life
 - abs
 - board
 - in_next_generation
 - is_alive
 - is_neighbor
 - neighbor_count
 - tables
 - text_processing
 - theca_stdlib
 - services
 - game_of_life
 - Turing

/predicates/life/in_next_generation

Comment

True if the cell at (x,y) is alive in the next generation, if this generation is specified by alive_string.

`in_next_generation(alive_string, -count, -born_count, -max_count, board_height, board_width, -min_count, +y, +x)`

Insert

- And
- Or
- Not

Replace

- Rule
- Table
- Literal
- Inequality
- Group
- Enumeration
- XML

is_alive(alive_string, y, x)

neighbor_count(alive_string, count, board_width, board_height, y, x)

$\min_count \leq count$

$count \leq \max_count$

is_alive(alive_string, y, x)

neighbor_count(alive_string, count, board_width, board_height, y, x)

$count \leq \text{born_count}$

$\text{born_count} \leq count$

$\min_count = 2$

$\max_count = 3$

$\text{born_count} = 3$

board(board_height, board_width, y, x)

Demo snapshots: movie db



simple_search

search_field:

Search

Number	Title	Description
8	<i>ABBA THE MOVIE</i>	Lasse Hallström använder en enkel handling om en journalist till att prodera ut Abbaexponeringen till en långfilm. Specialutgåva med extra intervjuer etc.
34	<i>BATMAN - THE MOVIE</i>	Bioversionen av pilotavsnittet till tv-serien från 60-talet med Adam West som den tontkitschiga versionen av mörkrets riddare.
49	<i>BULLET IN THE HEAD</i>	Tre unga män på mardrömskt äventyr i Vietnamkriget. Som en Hongkong-action-version av THE DEER HUNTER. Regi: John Woo.
70	<i>DE DIMHÖLJDA BERGENS GORILLOR (GORILLAS IN THE MIST)</i>	Sigourney Weaver som gorillaentusiasten Dian Fossey i Rwanda.
72	<i>DE MISSTÄNKTA (THE USUAL SUSPECTS)</i>	Intrigtät film om ett brott som blev Kevin Spaceys och regissören Bryan Singers stora genombrott.
85	<i>DET VILDA GÄNGET (THE WILD BUNCH)</i>	Sam Peckinpahs brutala död-i-slow-motion-västernklassiker. Den som hävdar att amerikansk film är våldsammare nu än förr har här ett kraftfullt motargument.
92	<i>DR JÄKEL & MR HYDE (THE NUTTY PROFESSOR)</i>	Jerry Lewis regisserar sig själv som närsynt kemilärare som blandar ihop en dryck som förvandlar honom till självsäker festprisse.
359	<i>EL ESPINAZO DEL DIABLO (THE DEVIL'S BACKBONE)</i>	Barnhemsbarnsryssare som utspelas nära slutet av det spanska inbördeskriget.

Movie Service Specification

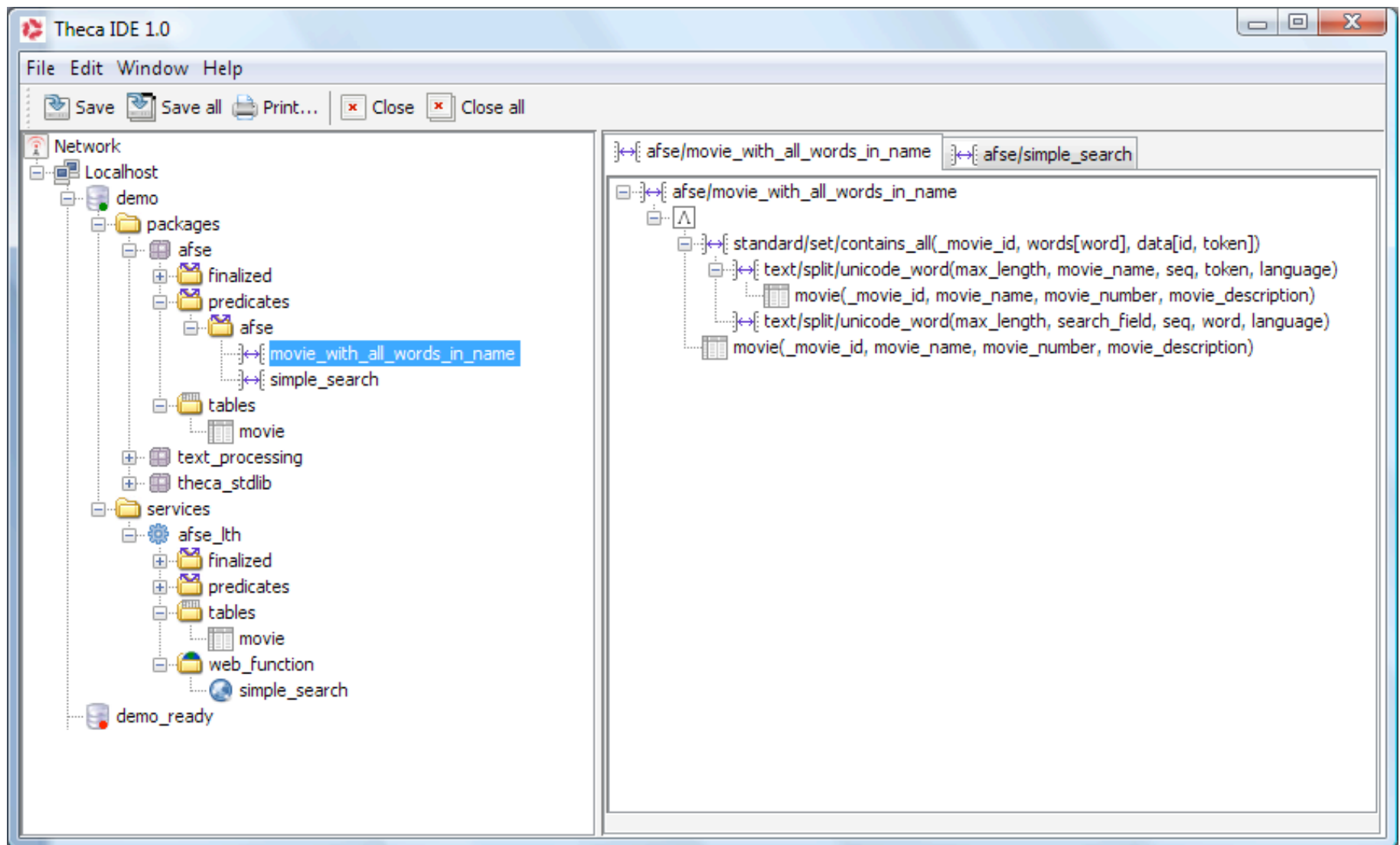


Data: A set of movies. Each movie has a unique number, and name and a description. The "numbers" are not really numbers, since some of them start with an R.

Input: A search string. A set of words divided by spaces.

Output: Any movie that have all search words in the movie name. All information about each movie is of interest. If more than one movie is found, they should be ordered by their names, A-Z.

Logical definition



The screenshot displays the Theca IDE 1.0 interface. On the left, a project tree shows the following structure:

- Network
 - Localhost
 - demo
 - packages
 - afse
 - finalized
 - predicates
 - afse
 - movie_with_all_words_in_name
 - simple_search
 - tables
 - movie
 - text_processing
 - theca_stdlib
 - services
 - afse_lth
 - finalized
 - predicates
 - tables
 - movie
 - web_function
 - simple_search

The main editor window shows the logical definition for the selected file `afse/movie_with_all_words_in_name`. The definition is as follows:

```
}↔{ afse/movie_with_all_words_in_name }↔{ afse/simple_search
|
|
|↔{ standard/set/contains_all(_movie_id, words[word], data[id, token])
|
|↔{ text/split/unicode_word(max_length, movie_name, seq, token, language)
|
|   movie(_movie_id, movie_name, movie_number, movie_description)
|
|↔{ text/split/unicode_word(max_length, search_field, seq, word, language)
|
|   movie(_movie_id, movie_name, movie_number, movie_description)
```

Execution Plan



The screenshot shows the Theca IDE 1.0 interface. On the left is a project tree under 'Localhost' with a 'demo' folder containing 'packages', 'tables', 'text_processing', 'theca_stdlib', 'services', and 'demo_ready'. The 'packages' folder is expanded to show 'afse', which contains 'finalized', 'predicates', and 'afse'. The 'afse' folder is further expanded to show 'movie_with_all_words_in_name' and 'simple_search'. The 'simple_search' folder is selected.

The main window displays the execution plan for the query 'afse/movie_with_all_words_in_name simple_search (query plan)'. The 'Test Proposition' section shows 'search_field' with the value 'DEN'. The execution plan is shown in XML format:

```
<x> xml
  P (movie_name,movie_number,movie_description)
    A (_movie_id)
      R hit_id->_movie_id
        P (hit_id)
          f
            C
              @@@@tab
```

A tooltip is visible over the 'C' node, displaying the following execution statistics:

- time for one tuple=223 microsecs
- time=338 microsecs
- tuples=13
- estimated cost for one tuple=143.0
- estimated cost=161.0
- tuples=7.0
- isEmpty=unknown
- method=com.theca.method.SpecialsBunch\$MergeJoin
- schema=[_movie_id id:'movie', movie_name string, movie_number string, movie_description string]
- orders=[_movie_id]
- keys=[_movie_id], [movie_number]
- deps=[

Thank you!



- jesper.larsson@apptus.com
- <http://apptus.com/>
- <http://completerewrite.blogspot.com/>