B21 DB2 UDB Net Search Extender, the fulltext search solution for your e-business application

Jürgen Metter, Senior Engineer, IBM

Fulltext search is an important requirement of e-business applications. The latest DB2 UDB Net Search Extender product offers a high performance and scalable fulltext search solution. The key features are the integration into the DB2 optimizer, in-memory database technology, administration using the DB2 Control Center, XML and thesaurus support. You will learn how DB2 UDB Net Search Extender can be easily integrated into a WebSphere application environment, the latest benchmark numbers and what the benefits are for your application with respect to fast application development, large scale web site support in a 7*24 environment.

B21

DB2 UDB Net Search Extender, the fulltext search solution for your e-business application

Jürgen Metter

IBM Data Management Technical Conference

Anaheim, CA **Sept 9 - 13, 2002**

RETURN TO INDEX

Overview

- Motivation
- Architecture
- Indexing
- Search

Motivation

- The majority of data in a relational database especially in e-business scenarios are textual data
  - names, addresses, product descriptions, newspaper articles, tenders, advertisements
- E-business starts with search.
- Search results must meet the quality and performance expectations
  even for the inexperienced intranet/internet user.
- Internet users do not want to wait.
  There is always a faster web page.
- The SQL like predicate provides a very limited search capability which may result in table scans.

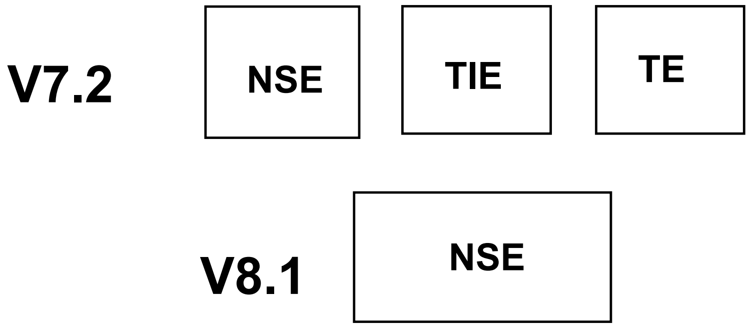
Why Net Search Extender V8

- Provides a rich set of search functions to find the most relevant information.
- Provides search results in subsecond time range for most e-business applications.
- Free upgrade for all DB2 UDB V7 customers.
- Priced feature for new DB2 V8 customers.
- Is available on your preferred platform :

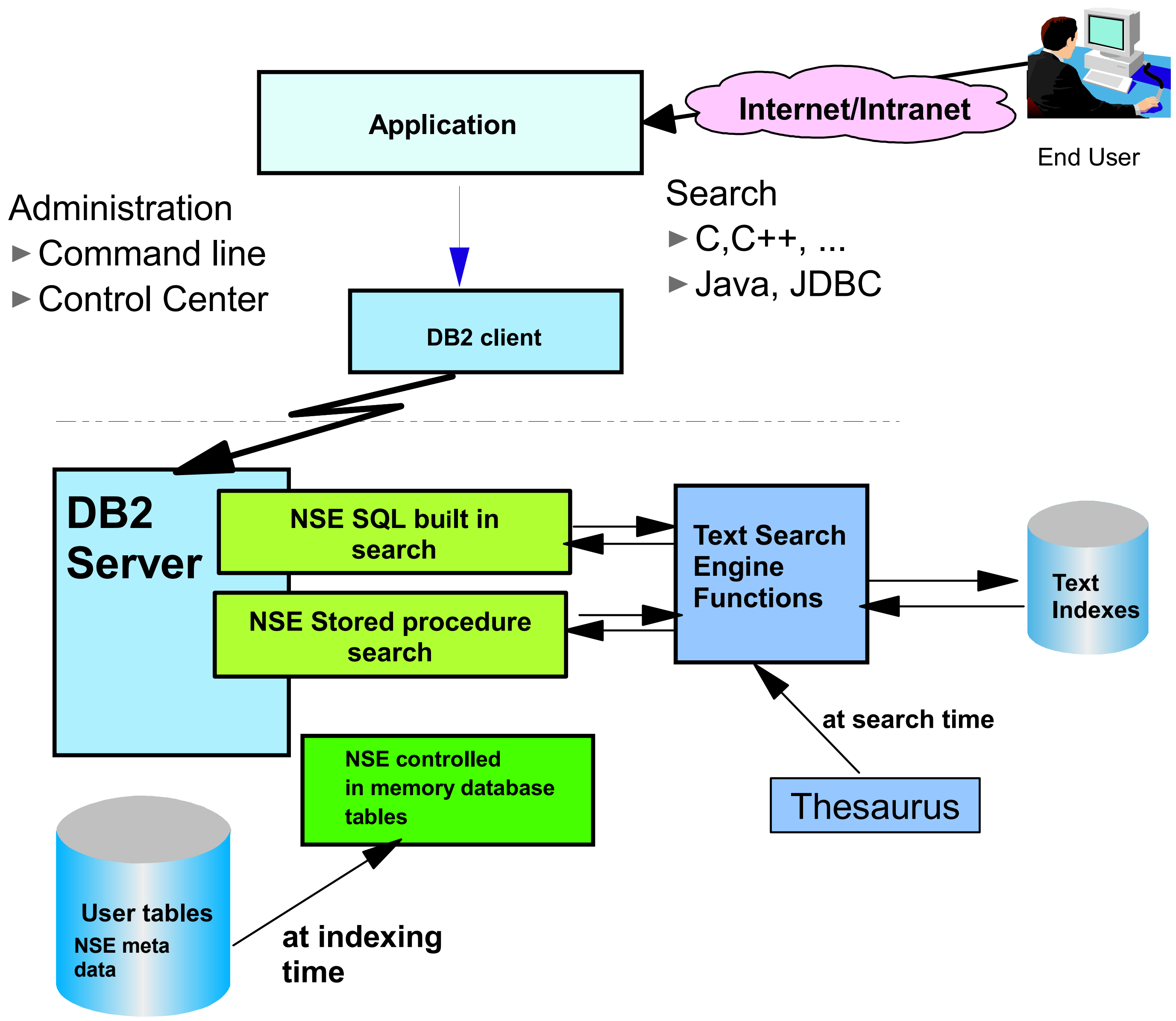
  (AIX, SUN Solaris, HP-UX, Linux/INTL, Linux/390, Windows, 64 Bit)
- Scales with your business
  - prepared to support index partitioning on multiple nodes
- 7*24 Support

Net Search Extender V8

- Delivers complete text search solution in one box. It includes
  - Merged Text Information Extender V7 and Net Search Extender V7
  - Text Extender V7
- Provides improved performance and scalability
- Delivers tighter integration with DB2 UDB
  - Administration integrated into Control center
  - Improved DB2 Optimizer integration



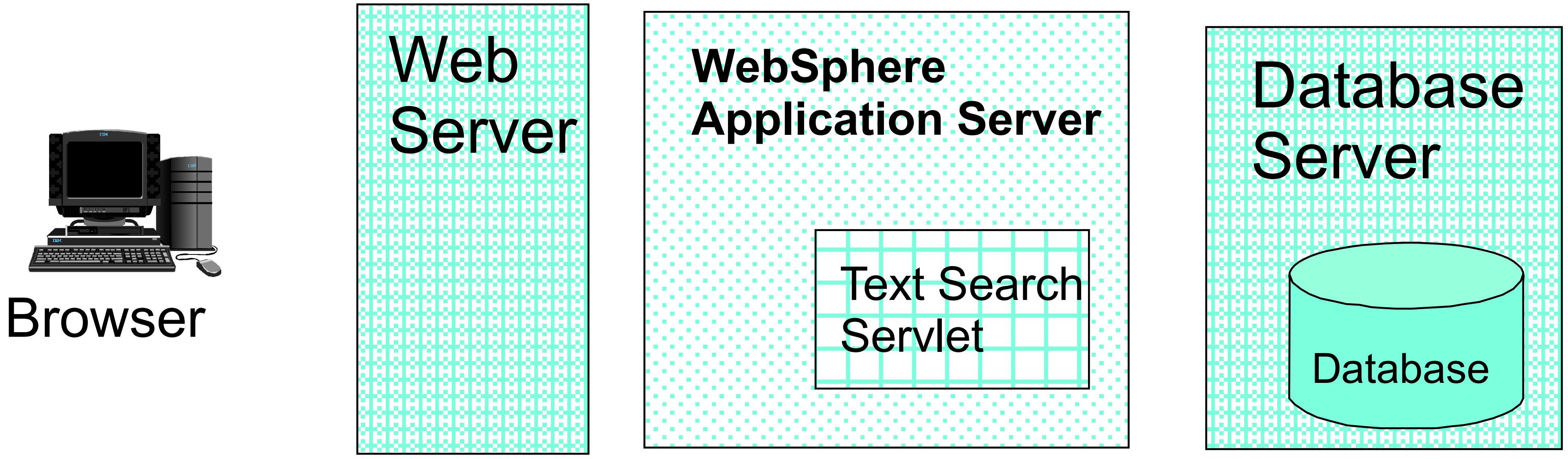
Architecture



Administration
- Command line
- Control Center

Search
- C,C++, ...
- Java, JDBC

Text Search in a WebSphere Application



Indexing ...

- **Fulltext indexing is the base for fast fulltext search.**
  - intuitive administration using command line interfaces or the new control center integration
  - indexing speed depending on document size and environment up to 3 GB
  - index update outside database transaction as asynchronous process benefit : indexed table/columns not locked during index creation or update
  - it is possible to search on an index while the index is updated
  - the fulltext index is an inverted file index together with additional information about word distance, section information, statistical information and is stored in outside the database on a user defined file system location
- **Data to be indexed can be stored in multiple locations**
  - table/column with any text datatype
  - via a datalink column (file, http,UNC,DFS protocol are supported) outside of the database
  - numeric column
  - view
    - avoid multiple indexes and additional joins to increase performance
  - federated database
  - via exit that supports text data stored in arbitrary data sources

Indexing ...

- **Supported text formats**
  - HTML, XML, documents with user defined tags
    - Markup tags and their logical field names are defined in a document model file. This file is used at indexing time for parsing and at search time to be able to restrict the search to sections in the documents.
      The document model is based on XPATH expressions.
  - plain text
  - formats which are not supported can be mapped to plain text using a third party format converter which can be plugged into the indexing process with a User Defined Function.

Indexing ...

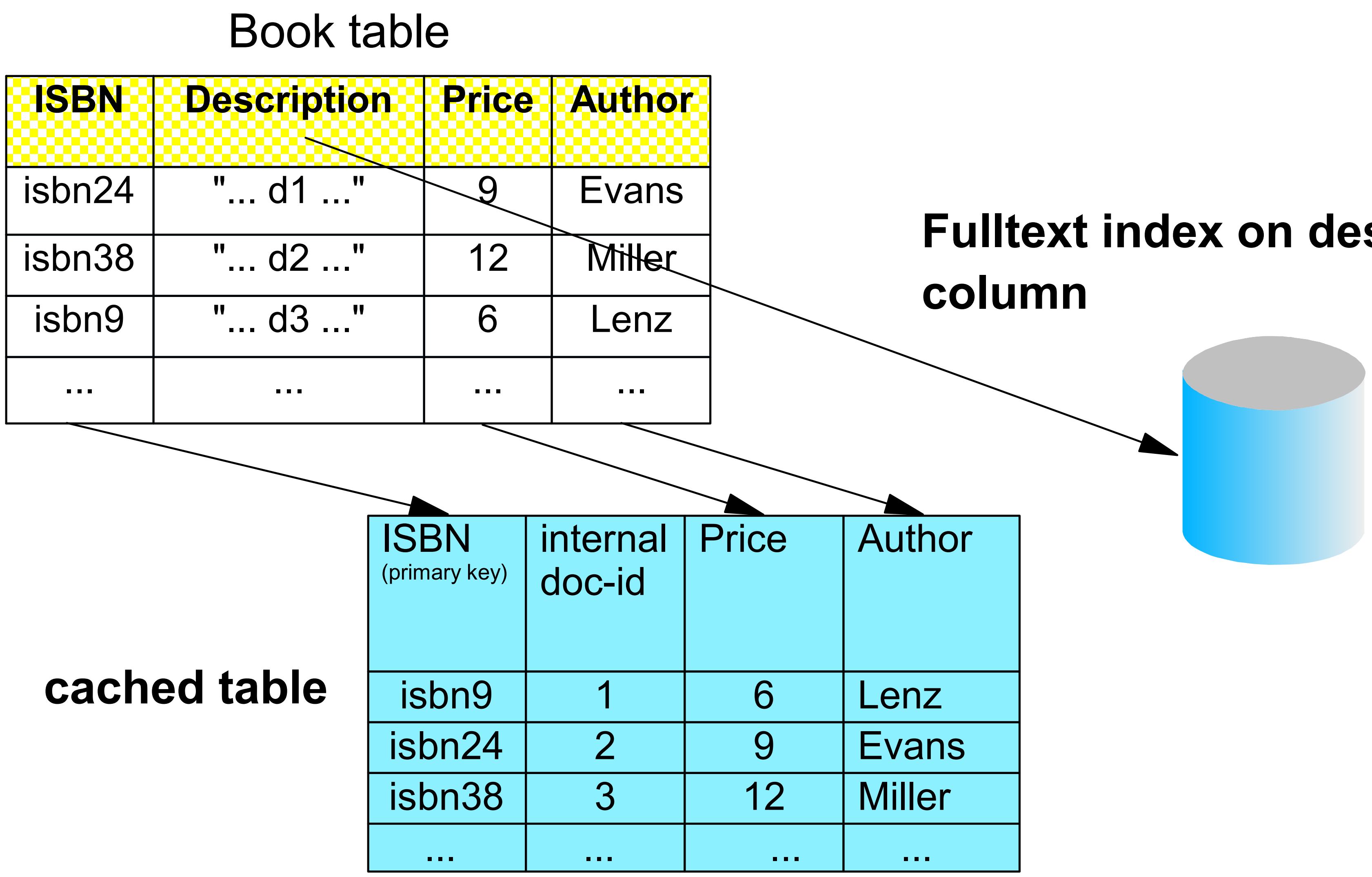
- **Languages**
  - 59 languages and their corresponding CCSIDs are supported
  - the language specification is used to determine sentence and paragraph recognition
  - Unicode support allows to store documents with different languages within one fulltext index.
    The index itself is encoded in UTF8.
  - GB18030 certified

Indexing ...

- **Improve search performance by creating an in-memory table**
  - A cached table is built in addition to the fulltext index. The cache table is built based on a SQL column expression which defines the columns to be extracted from the base table.
  - The benefit is that the search result set can be built by the search stored procedure without having to access the base table at search time.
- **Pre-ordered search result sets**
  - The user defined order by which the data to be indexed are retrieved, is preserved and used to define the search result list ordering.
    Multiple pre-ordered indexes can be defined per column.
  - Improved performance as no ordering is required at search time

Create index with cache table

db2text "create index desc_I on Book (Description) for text cache table (ISBN,Price, Author) initial search result order (price asc)



Book table

ISBN	Description	Price	Author
isbn24	"... d1 ..."	9	Evans
isbn38	"... d2 ..."	12	Miller
isbn9	"... d3 ..."	6	Lenz
...	...	...	...

Fulltext index on description column

cached table

ISBN (primary key)	internal doc-id	Price	Author
isbn9	1	6	Lenz
isbn24	2	9	Evans
isbn38	3	12	Miller
...	...	...	...

Create index including numerics

This provides a way to allow parametric searches combined with fulltext searches using the high performance stored procedure search interface.

Administration command to create the index :

db2text create index *desc_I* on *Book (Description)* for text attributes (*Price*)

Search using a stored procedure :

DB2 call db2ext.textsearch('attribute *"Price"* between *9* and *12'* and *"Roman history"')*','DB2EXT ','COMMENT ',0,20,1,0,?,?)

Search all documents that contain "Roman history" and have a prize between 9 and 12.

Return the first 20 records together with a scoring value.

Indexing ...

- **Stopword filtering**
  - language specific stopword lists are used to filter out not relevant terms at indexing time
  - Reduces index size
  - Improves search performance as stopwords are filtered out automatically if used in a query.
- **Fulltext Index update**
  - Triggers are created automatically at index creation time on the indexed table which write entries in a Net Search Extender created log table.
  - The frequency of the index update can be specified.
  - Search is available during index update.

Indexing ...

- **Reliability**
  - ► Indexing errors are logged in an event table together with the primary key of the text data that caused the error.
    This is important as the fulltext index creation is much more complex than creating a standard index and errors do not stop the indexing process.
- **Usability**
  - ► Administration functions are integrated into the DB2 Control Center
  - ► A command line interface and views on the Net Search Extender created meta data allow an easy application integration.

Search functions

In order to provide the best performance depending on the application needs, three different search functions are provided :

- **DB2 built in function**
  - provides best search results for queries that combine fulltext search with other search conditions.
  - The DB2 optimizer rewrites the query internally
- **Table valued function**
  - Allows to search on views
  - exploits presorted indexes
- **Stored procedure**
  - enhanced performance and scalability
  - increased main memory requirements for cached table columns
  - can only be used for fulltext searches

Search - Ranking

- **Large result lists require flexible, configurable ranking**

- **Features**
  - **weight** - specify how much a search term should contribute to the ranking value if found in the document
  - **accum -** group search terms together that will increase the rank value if they appear together
  - **minus -** decrease rank value if a search term is found

Search term expansion

- **Increase recall by automatically adding additional search terms.**
  - **Thesaurus**
    Create your own thesaurus which is used at search time to expand a search term according to a given relationship and a given depth of the relationship expansion. The semantics of a relation are defined by the application.
    Advantages :
    - Helps inexperienced users to find documents even if the search terms are not part of the documents.
    - Provides a feature that lets the user find the documents you want to be found.

Thesaurus search example

```
select price, author,
from Book
where contains (content, 'thesaurus "bookthes" expand
synonym term of "NSE" for 2 levels  ')=1
```

Thesaurus

```
 :WORDS
  NSE
 .SYNONYM_OF Fulltext Search
 .SYNONYM_OF Net Search Extender
:WORDS
  Fulltext Search
 .SYNONYM_OF Text Search
 .SYNONYM_OF Information Retrieval
```

Would find a document containing "Information Retrieval" or "Fulltext Search"

Search term expansion ...

- **fuzzy search**
  Search for variations of a search term with a specified degree of similarity.
  Advantages :
  - Helps to find documents even if the search term is not spelled correctly.
  - Helps to search in documents that may have been created by OCR

Increase the search speed

- Limiting the result set size to a specified number n.
  Only the best ranked n entries are returned if scoring is requested.
- Stop internal processing if a specified number of documents or words has been reached.
  The result list might not contain the best ranked entries.

Restrict search to sections of a document and return a ranked result list

```
select price, author,
score (content, 'sections ("index" weight 10, "h2" ) "DB2"') as score
from Book
where contains
        (content, 'sections ("index" weight 10, "h2" ) "DB2"')=1
order by score
```

Select price, author and score value from the book table for all books that contain "DB2" in the "index" or "h2" section and increase the rank value of a found book if the search term is found in the "index" section. Return the result ordered by the returned score value.

RETURN TO INDEX