BOI BETTER ORGANIZED INFORMATION

TABEX4 APPLICATION PROGRAMMING INTERFACE

TABEX4

TABEX4 is the leading cross-platform standard software for table access and maintenance. Highest performance and convenient data maintenance make TABEX4 an optimal tool for efficient and revision-proof table management.

TABEX4 APPLICATION PROGRAMMING INTERFACE

TABEX4 offers application programming interfaces using Standard Link Convention. Therefore, the interfaces can be used for all common programming languages such as C++, C, COBOL, PL/I and Assembler.

Various functions for accessing a table are available such as:

- Search of table content
- Access to table rows
- Queries
- Memory funktions
- Access to table and column definitions

For searching table data, the following methods are available:

- access via primary key
- access via secondary key (each table may have an unlimited number of secondary keys)
- access via sequential search in an arbitrary table column or area
- access via generic and match code search in an arbitrary table column or area
- access row by row, a number of rows or the whole table
- access using a SELECT string similar to SQL SELECT

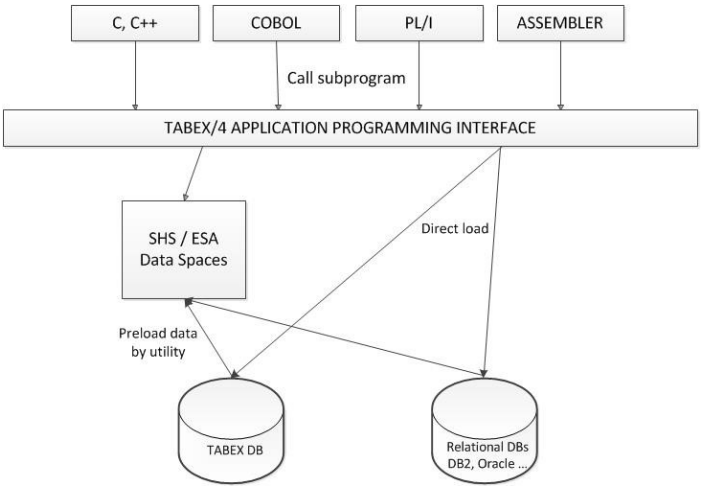

Abb.: API Calls

TABEX4 is noted for its high-performance table access. If tables are preloaded in TABEX-based data spaces (in the figure named as SHS / ESA Data Spaces) using TABEX utilities, the time accessing the table reduces additionally.

BOI Software Entwicklung und Vertrieb GmbH

40 years of experience.

Your specialist for table data management.

BOI LINZ
Spazgasse 4
4040 Linz, Austria

BOI GRAZ
Friedrichgasse 30/1
8010 Graz, Austria

Phone: +43 (0) 732 736423 - 0
Fax: +43 (0) 732 736423 - 2
E-Mail: office@boi.at
https://www.boi.at

TABEX4 APPLICATION PROGRAMMING INTERFACE: EXAMPLE

The following example shows an access via primary key for z/OS:

FU is the function code.
TABNAME is the table name.
SEARCH is the search argument.
ROW contains the result row.
RC is the Return Code.
COMMSTR is a communication structure for TABEX4 access functions.

ACCDATE, ACCDD, SEALNG and RETLNG have to be filled by the calling program:
ACCDATE is the access date for the table (TABEX tables can have a temporal validation area).
ACCDD is the dataset-Id of the table (a kind of qualifier to differ between various instances of the same table).
SEALNG is the length of the search argument.
RETLNG is the desired return length for ROW.

The remaining parts of COMMSTR are filled by the interface. They contain information about the table itself (area of validation, key definition, ...) and about the search result (e.g. number of rows found).

```
01 CPROG PIC X (8) .
01 FU PIC X (2) .
01 TABNAME PIC X (44) .
01 SEARCH PIC X (5) .
01 ROW PIC X (45) .
01 RC PIC X (1) .
01 COMMSTR.
        02 ACCDATE PIC X (8) .
        02 ACCDD PIC X (8) .
        02 SEALNG PIC 9 (4) COMP.
        02 RETLNG PIC 9 (4) COMP.
        02 TABROWN PIC 9 (8) COMP.
        02 TABROWL PIC 9 (8) COMP.
        02 TABROWI PIC 9 (8) COMP.
        02 TABFROM PIC X (8) .
        02 TABTO PIC X (8) .
        02 TABKDEF PIC X (2) .
        02 TABKPOS PIC 9 (4) COMP.
        02 TABKLNG PIC 9 (4) COMP.

Move 'TABIMS ' TO CPROG.
Move 'T ' TO FU.
Move 'TESTTAB ' TO TABNAME.
Move '99999999' TO ACCDATE.
Move 'TESTDB ' TO ACCDD.
Move 'KEY01' TO SEARCH.
Move 5 TO SEALNG.
Move 45 TO RETLNG.
CALL CPROG USING FU, TABNAME, SEARCH, ROW, RC, COMMSTR.
```