

## SAViT Redokumentation Tool

**S**ource **A**nalysis and **V**isualisation **T**ool

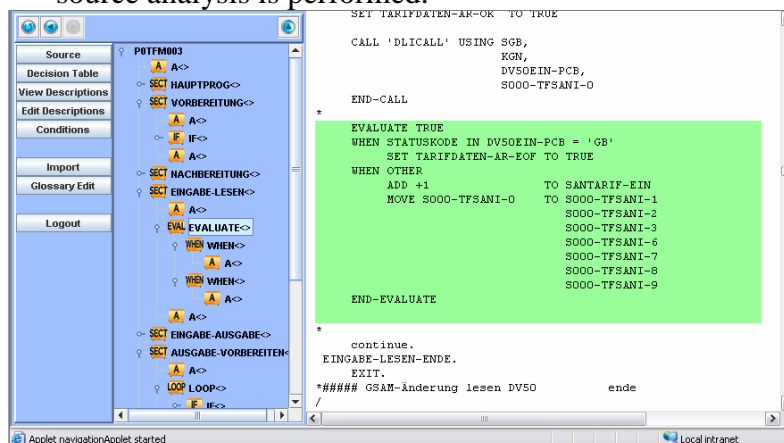
SAViT is a centralized source documentation management system including features for COBOL source parsing, analysis and visualization. The tool makes it possible and eases the documentation and analysis of existing program sources even when there is little or no documentation present about them. Tool implements bottom-up documentation technique, where it is possible to describe smaller blocks of the program, and with the gained understanding move to higher blocks, while all program is documented.

SAViT is based on the application with WEB interface for online documentation management, but it is also possible to export the produced documentation in HTML form for offline usage. By using additional tool GRADE (developed by Institute of Mathematics and Computer Science of University of Latvia together with DATI and recognized as one of the first top-quality modeling tools), it is possible also to visualize the program not only by traditional statement-level logic diagrams, but also by higher level and more descriptive logic diagrams.

SAViT also provides some analysis functionality allowing to detect problematic blocks in programs at various levels of detail (program-level, section-level, ...). Also, when another problem patterns are identified, they can be plugged into the tool and used for overall analysis processes and statistics. The problem patterns are not fixed and can be defined individually.

Main functions of the tool include:

- **Source import, parsing, building of the source tree.** The tool is capable of importing and parsing the “as-is” COBOL sources. During the import phase the source is parsed, an overview of a program – “source tree” is made and source analysis is performed.



```

SET TARIFFDATEN-AR=OK TO TRUE

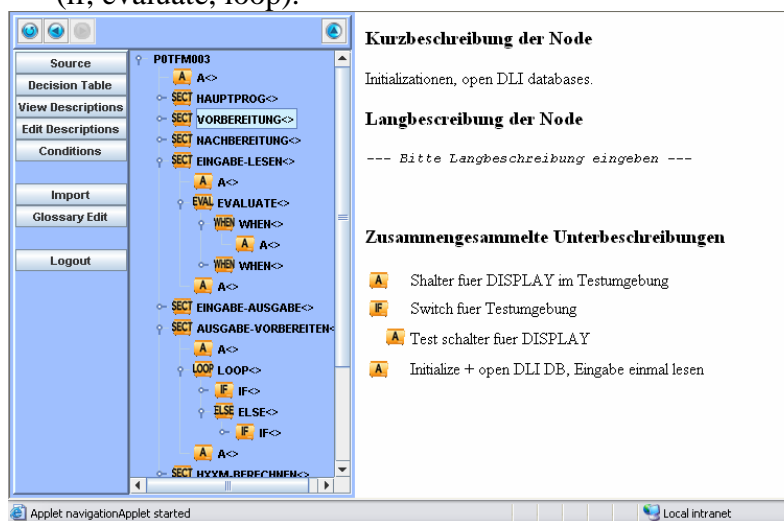
CALL 'DLICALL' USING SGB,
      KGN,
      DVSOEIN-PCB,
      S000-TFSANI-0

END-CALL

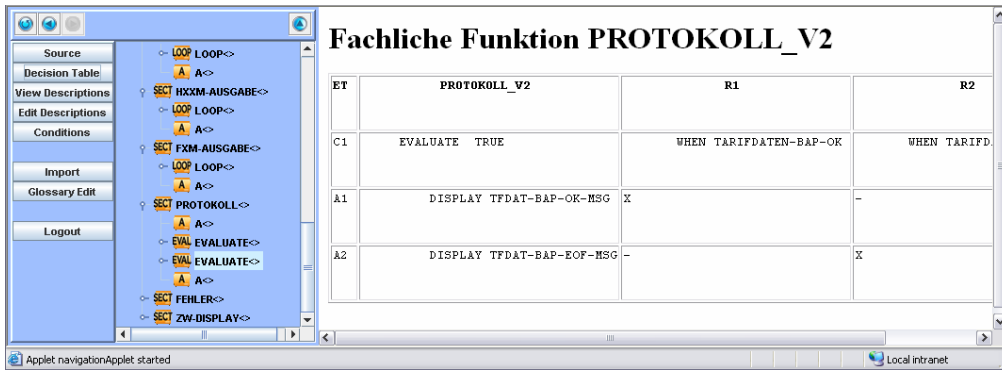
*
EVALUATE TRUE
WHEN STATUSKODE IN DVSOEIN-PCB = 'GB'
  SET TARIFFDATEN-AR=EOF TO TRUE
WHEN OTHER
  ADD +1          TO SANTARIF-EIN
  MOVE S000-TFSANI-0 TO S000-TFSANI-1
                    S000-TFSANI-2
                    S000-TFSANI-3
                    S000-TFSANI-6
                    S000-TFSANI-7
                    S000-TFSANI-8
                    S000-TFSANI-9
END-EVALUATE

*
continue.
EINGABE-LESEN-ENDE.
EXIT.
***** GSAM-änderung lesen DV50      ende
  
```

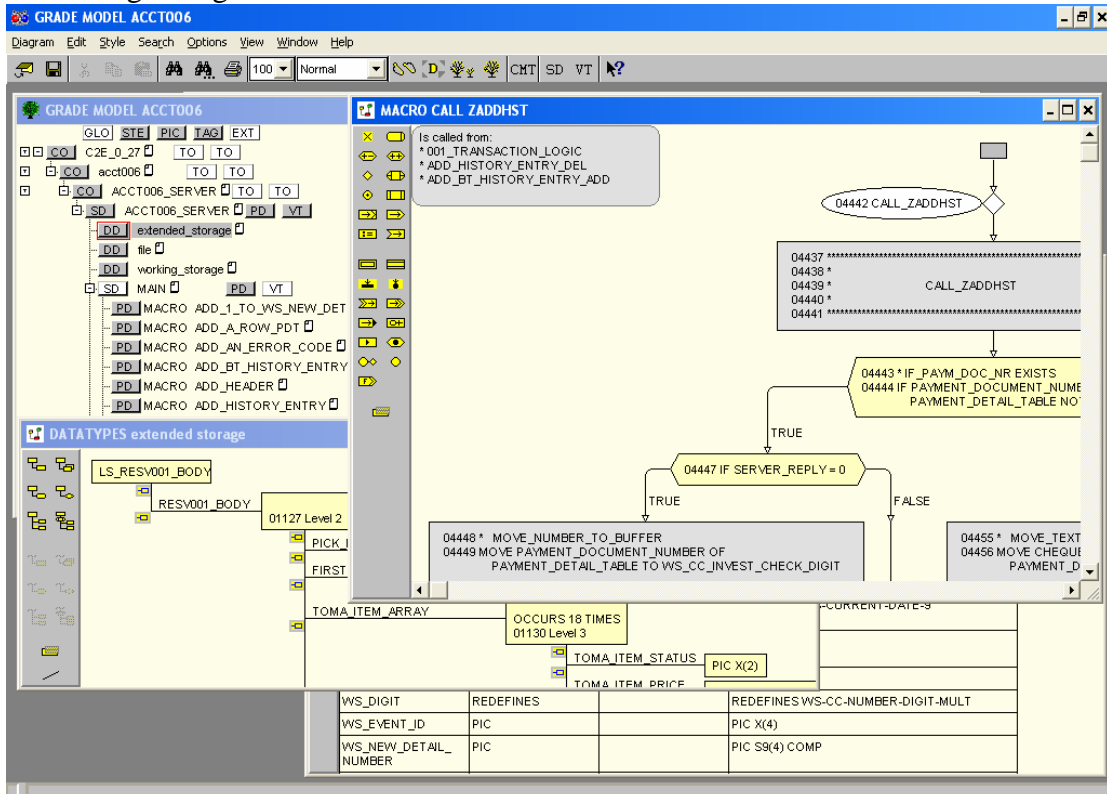
- **Navigation through source code.** Navigation through various parts of the source code is done through use of hyperlinks. For example, PERFORM statement will be given in a form of a hyperlink, clicking on which would display the called part of the program.
- **Entering and browsing descriptions of a source at various levels of detail.** It is possible to add descriptions to any source tree node and use “bottom-up” describing technique: first write some descriptions for lower level source parts such as action blocks, if statements, loops, etc, then – going up and creating a descriptions for higher level constructs as sections and container statements (if, evaluate, loop).

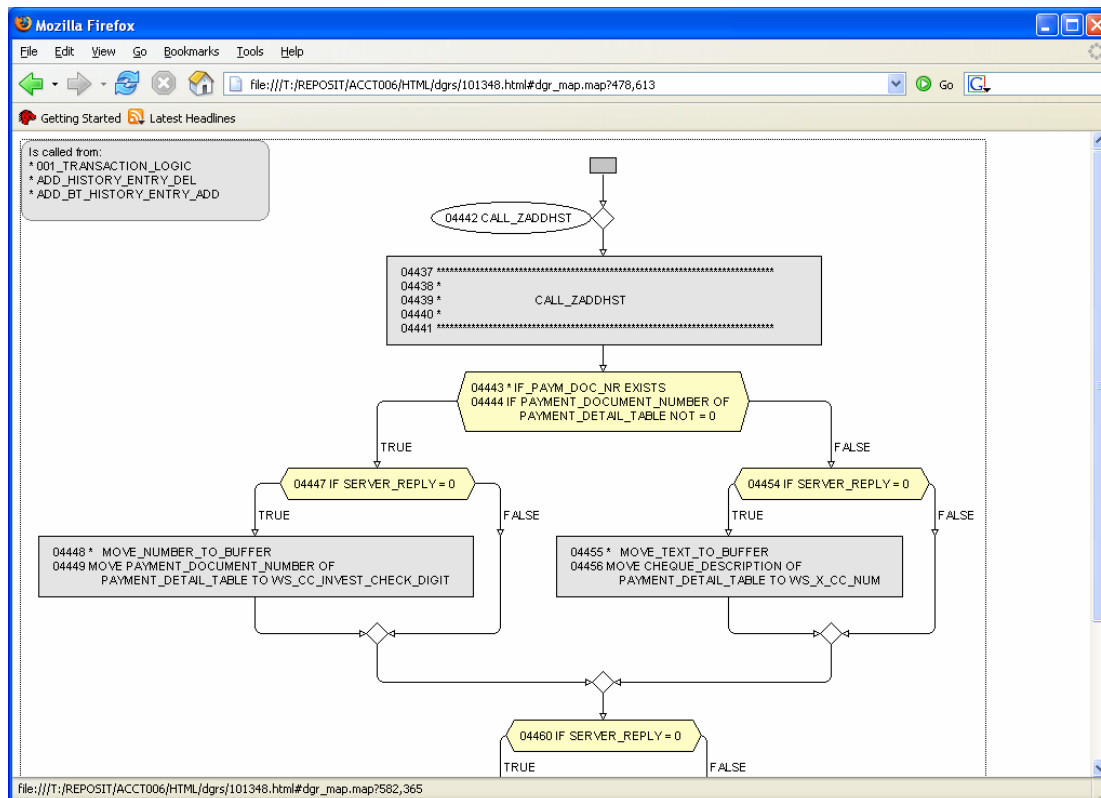


- **Entering and browsing descriptions for variables, business terms, etc.** The tool provides so-called Glossary, in which it is possible to put descriptions for variables, business terms, or any other arbitrary text. These descriptions will be displayed while browsing through code in any place where the variable / term / text appears.
- **Analyzing the execution flow.**
  - It is possible to view the execution conditions for each source tree node. All the conditional statements that affect the execution of a statement block will be taken into consideration and a summary will be displayed.
  - It is possible also to view all possible call hierarchies of a section. This allows tracking all situations under which a section gets called.
  - For easier understanding of complicated formulas coded in sections, tool provides means to display so-called decision tables. Logic can be automatically presented in form of Decision tables. Decision tables present if-then-else and EVALUATE statements, and associate conditions with actions to perform. (Definition: a table of contingencies to be considered in the definition of a problem, together with the actions to be taken; sometimes used in place of a flow chart for program documentation).



- **User management.** Application access can be controlled by assigning special roles to the users. For example, some users may have access rights to edit descriptions, some – just to view them.
- **HTML export.** Tool allows exporting the produced documentation in form of HTML files. This allows creating offline manuals for the sources.
- **Keywords, indexing.** It is possible to define keywords for each source tree node, which can later be used for keyword search.
- **Input and output field reporting.** For each section the tool can produce a report about used input and output variables.
- **Versioning and source change management.** The tool allows keeping track of source code versions and transfer of the documentation from one version to another. Also descriptions can be versioned.
- **Visual representation of the source code logic.** By using additional tool GRADE, it is possible also to visualize the program not only by traditional statement-level logic diagrams, but also by higher level and more descriptive logic diagrams.





Visualization capabilities include:

- Data Diagram for File, Working Storage and Extended storage data structures;
- Variable Table for Working Storage and Extended storage data structures;
- Process Diagram for each COBOL paragraph;

Diagrams can be produced both with statement level descriptions and also with descriptions entered in documentation tool, thus providing a possibility to produce also higher-level and more descriptive diagrams.

Functional capabilities include following:

- browse and edit the diagrams;
- add hyperlinks and attachments to the diagram elements;
- navigate to called paragraphs (top-down navigation);
- navigate to calling paragraphs (bottom-up navigation);

GRADE, the advanced business and system modeling tool, has been successfully used in following DATI projects:

- Business analysis of loan management for Finanz IT – an IT subsidiary of German Saving Banks. The analysis and modeling of a set of corresponding business processes was realized with GRADE;

- KSK - Artist's social insurance system for LVA Oldenburg-Bremen, Germany. GRADE was used to design the whole application;
  - PROVIT - Reservation system for travel agencies and tour operators for LTU, Germany. We realized reengineering of existing Cobol programs and design of new functionality with GRADE;
  - PHOENIX Business Travel - travel reservation system for DER Tour (Germany). We realized reengineering of existing Cobol programs and design of new functionality with GRADE.
- 
- **Source code quality analysis.** The analysis functionality allows to detect problematic blocks in programs at various levels of detail (program-level, section-level, ...). Also, when another problem patterns are identified, they can be plugged into the tool and used for overall analysis processes and statistics. The problem patterns are not fixed and can be defined individually. As candidates there are during design phase detected problems:
    - Using GOTOs;
    - Avoid using COPY-structures for file definitions;
    - Reference on absolute address in record;
    - Missing file status check;
    - Recursive performs;
    - Sentences as legacy of COBOL74 standard.

**Contact:**

Claus-Jürgen Moessinger  
Postfach 1321  
25465 Halstenbek  
Tel: 04101-8556-0  
[www.dati-hamburg.com](http://www.dati-hamburg.com)  
[zentrale@dati-hamburg.com](mailto:zentrale@dati-hamburg.com)