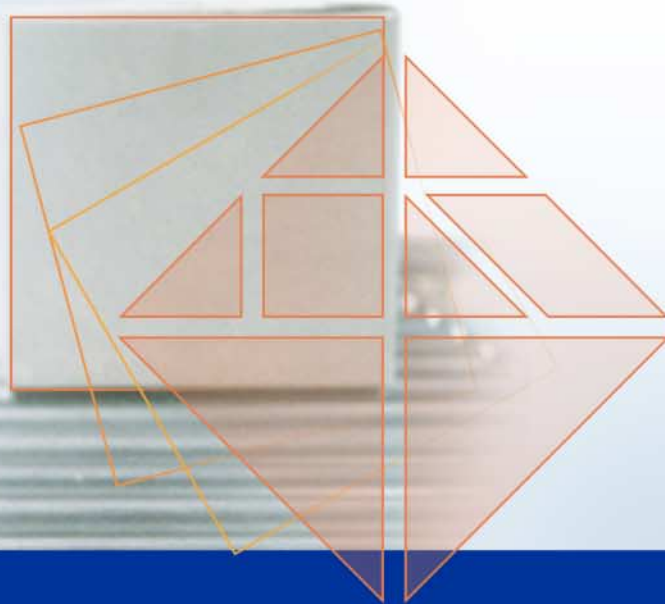


case/4/0

COBOL Reverse Engineering with case/4/0



A lot of the COBOL applications used today – many of which are considered irreplaceable – pose a substantial maintenance burden. Often the only way of reducing this burden in the long run is to improve the internal implementation of the application system – without affecting its outward functionality in the process. Before existing COBOL application systems can be cleaned up and restructured, it's necessary to “rediscover” the original software design on the basis of available COBOL sources. The case/4/0 add-in which we would like to introduce you to here supports you in exactly this step.

Contents

Who was COBOL reverse engineering developed for?.....	3
What can COBOL reverse engineering for case/4/0 do?	3
And which advantages result from these capabilities?	3
How COBOL Reverse Engineering Works	3
Contact.....	5

Who was COBOL reverse engineering developed for?

COBOL reverse engineering was developed for all organizations which:

- ▶ need to restructure COBOL applications which were developed in-house or were otherwise acquired
- ▶ want an easy way into bottom-up application development according to structured methods

What can COBOL reverse engineering for case/4/0 do?

The add-in to **case/4/0** introduced here supports this step by:

- ▶ revealing the structure of COBOL programs
- ▶ creating clear, graphic documentation within **case/4/0**, thus making it easier to understand the programs and their structures
- ▶ providing a basis for analyzing various aspects of the structure and code using **case/4/0**'s evaluation functions
- ▶ offering the structure and code for modification, maintenance, and further development with **case/4/0**.

And which advantages result from these capabilities?

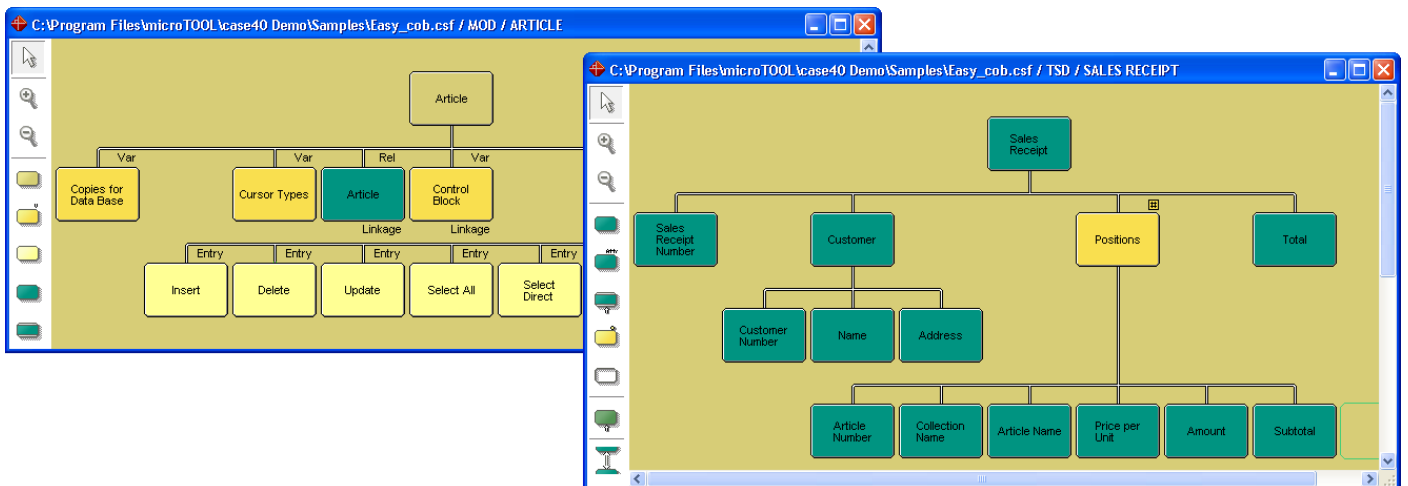
The **case/4/0** add-in for COBOL reverse engineering delivers clear documentation of the software as the basis for restructuring a COBOL application. Once this process is completed, you can expect a variety of advantages from the use of **case/4/0**:

- ▶ distinctly reduced maintenance requirements of the COBOL application due to graphic, consistent structuring
- ▶ high productivity during extension and adaptation of the program through the use of scripts for code generation: User experience has shown that under **case/4/0** can automatically generate up to 100% of database access statements, 30–70% of surface routine statements, and 10–15% of processing logic statements
- ▶ the possibility of realizing new architecture concepts. An example of this is object-based “wrapping” of a COBOL application, and its subsequent integration into a component architecture based on COM or CORBA

How COBOL Reverse Engineering Works

The **case/4/0** add-in for COBOL reverse engineering contains a parser for analyzing COBOL programs and COBOL include elements. Before it can be used, the COBOL code must be syntactically correct. The following elements are created and saved to the **case/4/0** repository during analysis:

- ▶ Module structures, a type of graphic index of COBOL programs, which contain references to called programs and more
- ▶ Type structures, which show the structure of data in groups, fields, repeating groups, etc. and which are referenced in the module structure
- ▶ Data elements in the data catalogue with their formatting, default values, and value sets



Module Structure and Type Structure – Created with case/4/0 via COBOL Reverse Engineering

The COBOL source code is also read into the **case/4/0** repository and the graphic elements created there are structured. The result is a clear, graphic description of the COBOL program, which is directly linked to source code. Results developed directly in **case/4/0** are not formally discernible from those created by the reverse engineering add-in. That means that **case/4/0**'s numerous evaluation functions are available for testing the results of reverse engineering. All results can be edited both graphically or in the code. **case/4/0**'s repository automatically ensures that any changes made are consistently updated throughout the program – thus ensuring a high level of quality. Afterwards, the cleaned-up, restructured and graphically documented source code can be newly generated with **case/4/0** into files.

Contact

Would you like to find out more about **case/4/0**? Then just contact us. We look forward to hearing from you.

microTOOL GmbH
Voltastr. 5
13355 Berlin, Germany
Phone (+49 30) 467 086-0
Fax (+49 30) 464 47 14

info@microTOOL.de

<http://www.microtool.biz>

Or try out the tool for yourself: We offer a free demo version of **case/4/0**. You can download it from:

www.case40.de/Download