## DOES THE FUSION METHOD WORK?

*by Kris Oosting,*

In the April 1996 Fusion Newsletter, we reported some metrics that indicated that the Fusion method indeed leads to software with lower complexity (and less defects). Many people asked us to report on it again when the application built using Fusion was finished. Well, the application is finished so the final metrics can be presented. In April 1996 the size of part of the application was 15 KNCSL—now it is 80 KNCSL (Project D).

On each of the projects the following were measured:

- Highest complexity[3]—to show the limit
- Average complexity—to give an indication of the whole product
- Percentage of modules with complexity above 40—an indicator for maintenance effort and defects to expect
- Complexity increases by 1 for every x NCSL—to show how many lines of code are needed to increase the complexity by one. Just to compare the projects here. Lines of code (NCSL) means only code, not empty lines and comment lines.

All projects use a windows environment (OSF-Motif, MS-Windows), and C++. Project A has also parts written in C. The results of the measurements are:

**Project A:** bits and pieces of OMT and Coad/Yourdon used, 85 K lines of code.
Highest Complexity: 320
Average Complexity: 68
Percentage of modules with complexity above 40: 26%
Complexity increases by 1 for every: 4 NCSL

**Project B:** OMT used, 35 K lines of code.
Highest Complexity: 140
Average Complexity: 20
Percentage of modules with complexity above 40: 20%
Complexity increases by 1 for every: 7 NCSL

**Project C:** OMT used, 18K lines of code.
Highest Complexity: 240
Average Complexity: 40
Percentage of modules with complexity above 40: 30%
Complexity increases by 1 for every: 10 NCSL

**Project D:** Fusion used during whole process, 80 K lines of code. (54 KNCSL where generated by GUI tools which are not included here)
Highest Complexity: 132 (one class with very complex arithmetic)
Average Complexity: 6.7
Percentage of modules with complexity above 40: 3.2%
Complexity increases by 1 for every: 16 NCSL

The complexity level of 40 was chosen because we're dealing with complex applications and C++.

Still we can conclude that Project D is easy to adapt and maintain. Complexity is low and therefore Defects were easier to locate and there were much less critical and severe defects. This saves valuable time and money.

It is nice to see that a complete project done with Fusion gives such good results!

3. Complexity is the McCabe Cyclomatic number per class.