CHARACTERISTICS AND EVALUATION METHODS OF THE CASE TOOLS

Lect.Univ.Dr. Ionel Iacob
Facultatea De Informatică Managerială
Universitatea Româno-Americană

ABSTRACT

The acronym CASE – **C**omputer **A**ssisted **S**oftware **E**ngineering – is the term used to indicate a collection of methods, tools and processes used in the development of software products with the assistance of the computer.

The present trend is to replace the terms "Assisted" and "Software" with the terms "**Aided**" and "**System**" in order to point out that such a tool, not only assists, but actively helps not only the creation of simple software programs but also of complex systems ready to offer today solutions for the problems of tomorrow. In the related literature there are also other names for such products that deal with analysis, design and the creation of information systems, such as:

- **I.P.S.E** (**I**ntegrated **P**rogram (/**P**roject) **S**upport **E**nvironment)
- **P.S.E** (**P**rogramming **S**upport **E**nvironments)
- **S.E.E** (**S**oftware **E**ngineering **E**nvironments)

Mentioned for the first time in 1987, the term CASE was used to depict tools that cover the stages of analysis and design, also offering graphic representation facilities that help software development **(Manly, 1987).**

Oxford Computing Dictionary defines a CASE as:" A term from the 'marketing' domain which describes a programming support environment (PSE) that supports only one method and offers a set of software tools which aids the method. Sometimes the term may be used as synonymous with IPSE.

The purpose of CASE tools "is that of offering programmers and software developers, tools that are meant to help them in specifying the functional and design requirements needed for the development of applications. The long term purpose of CASE technologies is to automatically generate the code based on the created deign using such tools**" (Fisher, 1988)**

CASE technology is "a software technology that offers an automatic technology for software development, maintenance and project management that also includes automatic structured methodologies and automated tools." (**McClure4, 1988**)

It may be said that CASE tools support the analysis and design methods true a large automatized area of the field activities, totally or partially assisting the life cycle of the product ensuring the management and quality needed for it.

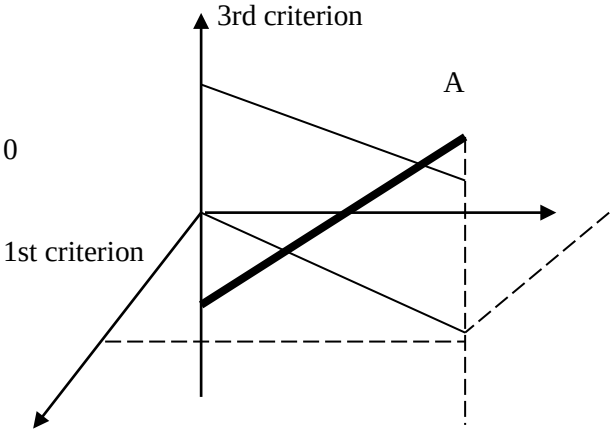
A CASE tool/environment must fulfill at the same time a minimal set of characteristics, such as:

- possibility of using the tool on any hardware platform.
- usage of advanced programming languages for the development of applications.
- ensuring a high quality standard for developing of applications.
- allowing access to information for the whole team.
- offering powerful graphic facilities in order to describe and document the software.
- reuse for the development of new systems, applications and programs.
- integration for permitting an easy data transfer among components.
- storage of information regarding software in a computerized storage named repository.
- possibility of being used as a base for the automatization of software production using one or more methods of analyzing and design.
- possibility of designing interfaces with a high standard of interaction with the user.

Within a team that decides to use such a tool in order to achieve a project, certain questions arise:" Which is the best CASE tool for the project?" or "Is it better to use tool X or Y?".

Even though the elaboration of an ideal model of choosing the optimal CASE tool suited for different projects has been tried, it hasn't been achieved such one that all the specialists unanimously would approve.

In 1992, professor Bordoloi and his team developed a method which seems to be the most close to the optimum one. The method consists of choosing a three dimensional scale in which on each of the three axes are positioned orderly, from analysis to design, the life cycle phases.
There are some opponents of this method who demand that a quality criterion should be introduced so that a comparison of CASE tools could be made no matter their category belongs. The quality indicator should be obtained by placing a straight line from the X origin point, where X represents the intersection of the three point's projections from the axes intersection.



The evaluation of a CASE may also be done from the point of view of restrictiveness defined as being:"the level until which and the manner in which a CASE limits the software's development process by user to a subset of possible processes". The 'ideal' CASE tool must ensure a balance from this point of view in the sense that there must be an equilibrium because the analyst might lose himself in the multitude of options and this might lead to the inefficient or unsuited usage of that CASE.
The restrictive levels have been separated by specialists into seven big points. Therefore a CASE may oscillate between these seven points from a forced CASE to a free one.

Chart - Restrictive levels

During creation/editing			Saving/Exiting		Post-method	Unimplemented
Automatic		On demand	Automatic		On demand	
Obligatory	Optional		Obligatory	Optional		
1	2	3	4	5	6	7

From the point of view of the structured analysis a CASE tool must respect certain rules, such as:

» Each process has a name – level 1;
» The mother-process must be specified before the child-process – level 2;
» A process must be detailed in a data flux diagram or in an elementary process – levels 3,4,5,6 ;
» The process numbering must be made taking into consideration the hierarchy ;
» A process must have at least a data flux of entrance and at least one data flux of exit – levels 5,6 or 4;
» A process must be connected with at least one data storage or with a process or with an external entity levels 4,5 or 6.

Assessing the quality of a CASE tool

From the point of view of quality, a CASE environment is hard to be classified because it is not a material product. A CASE tool may be best qualified taking into consideration the results obtained by using it.

Statistics have showed that by using a CASE environment the work time spent is noticeably reduced and the global quality of the product obtained increases for the problems of average complexity.

In order to obtain a quality optimum for the problems of a low degree of difficulty is recommended to be used a "**lower CASE**" and for those of a higher degree of complexity is recommended to be used the "**upper CASE**" tools.

The conclusions regarding the quality of a CASE tool may be obtained after a high number of experiments regarding the creation of products program by means of that CASE. The quality is analyzed based on the stages performed in the analyzing of program products: analysis stage, design stage, implementation stage, integration and testing and the maintenance stage.

In order to evaluate a CASE, specialists use the first two of the **thermodynamics** laws:

1. Nothing may be obtained without effort (energy conservation law).The efficiency of using CASEs is situated somewhere between 85% and 97%. The effort is substitute by the costs.
2. An isolated process within the system creates entropy at least equal with the system's entropy. The necessary effort rises exponentially with the complexity of the problems and with the entropy also increased of the isolated processes within the systems.

BIBLIOGRAPHY

1 E. E. Anderson, A heuristic for software evaluation and selection, Software—Practice & Experience, 2007

2 2. J. Baker, "An Approach to Specifying Ideal Software Development Environments," 2006

3 G. Baram , G. Steinberg, Selection criteria for analysis and design CASE tools, ACM SIGSOFT Software Engineering Notes, 1999

4 W. R. Beam , J. D. Palmer , A. P. Sage, Systems engineering for software productivity, IEEE Transactions on Systems, Man and Cybernetics, 2007

5 Roberto Bisiani , François Lecouat , Vincenzo Ambriola, A Tool to Coordinate Tools, IEEE Software, 2008

6 Pearl Brereton, Software engineering environments, Halsted Press, New York, NY, 2008

7 7. T. A. Bruce, J. Fuller, and T. Moriarly, "So You Want a Repository," *Database Programming & Design*. 2009.