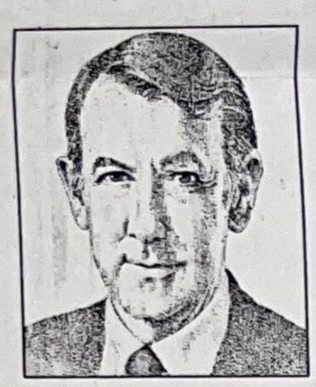
APPLIED INTELLIGENCE

Modify Your Methods To Take Advantage of I-CASE Tools



JAMES MARTIN

In this, the last of a series of columns on I-CASE technology, we discuss the improvement in productivity that can be realized with the use of CASE tools.

To achieve high productivity in building computer applications, it's necessary not only to select the best CASE and I-CASE (integrated computer-aided software engineering) tools, but also to adapt your organization and methods to take full advantage of these tools.

The simplest CASE tools are little more than diagramming aids. They simplify the drawing of diagrams and enable them to be kept tidy and modified quickly.

Code generators enable implementors to produce a working program quickly. However, if the generator is not linked to a dictionary, data model or design tools, the programs generated may be incompatible fragments, ill-designed and not linked together.

To achieve high productivity, the tools for design need to be tightly coupled to the code generator. The tools should employ a data model and should enable the design to be represented in a powerful, visual, easy-to-modify form from which code is generated directly.

The programs should be quickly executable so that the designer can observe what they do, adjust or add to the design, rerun the programs, enhance the design and so on, until a comprehensive system is created.

The principle of "what you see is what you get" should apply to the combination of visual design tool and code generator. The need for manual coding of procedures should be removed to the maximum extent.

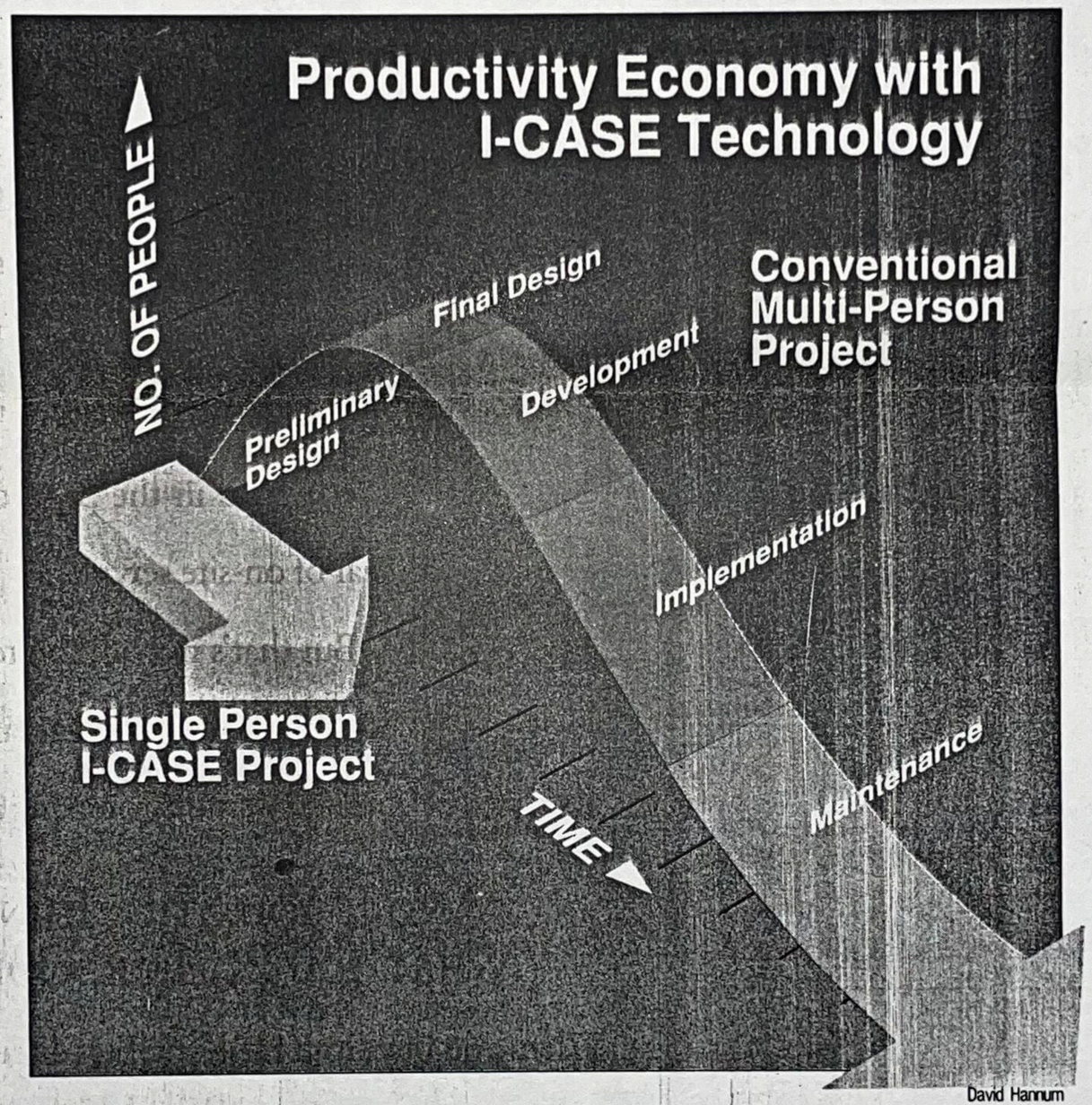
The generator may initially be used to produce structured code that relates to the design screens. This code may be used for prototyping and debugging. Because structured code does not give optimal machine performance, for heavy-duty applications the code may be fed into an optimizer, which creates code with optimal machine performance. This code will never be touched by maintenance programmers; all maintenance is performed at the specification level.

The integrated designer-generator tool should aid in rapid construction and modification of prototypes. It should generate test data and provide testing tools. It should generate database code and job-control code, so that the program can be executed quickly when design changes are made.

Productivity in system development is strongly affected by the number of people in the development team. Large teams tend to give low development productivity, because the number of interactions between team members increases rapidly as team size increases.

One objective when using CASE tools is to avoid having large teams of programmers. As shown in the figure, conventional technology often requires large development teams and long development cycles. In contrast, many projects or subprojects developed with I-CASE tools can be completed by one person. The brilliant, fast or hard working individual then has the opportunity to excel. Management should encourage the most capable implementors to learn the full power of an I-CASE tool set in order to maximize productivity.

Big projects should be subdivided into small projects, each of which can be completed relatively quickly by one, two or, at most, three implementors. The CASE tool should make it possible to define with computer precision the interfaces between the components generated by separate teams.



David Hannum

Integrated CASE tools promote the use of highly productive small development teams and increase the quality of developed applications.

At the start of a large project, an entity-relationship diagram should be created for the data that will be used, the data elements should be defined and the data correctly normalized. The same data model should be used for all subprojects. The flow of data and control among subprojects should be defined with the CASE tool.

Today's programmers constantly reinvent the wheel. They struggle to create something that has been created endless times before. Major productivity gains will result from employing reusable designs, data models or code. Reusable components should be catalogued in a CASE encyclopedia so that they can be selected when needed and modified as required.

A large enterprise should employ the same I-CASE tool set at all locations where systems are built so that common data models, designs and program components can be used. Telecommunications access to mainframe encyclopedias aids in the sharing of applications, document design, accounting procedures and so on.

In systems developed by traditional manual techniques, maintenance is a major problem. Systems are often difficult and time-consuming to change. After being modified many times, they often become fragile; even minor changes result in bugs and breakdowns.

A goal of I-CASE tool sets is to produce systems that are quick and easy to change. Maintenance isn't done by digging around in spaghetti code but by modification of the design specifications, followed by regeneration of code.

Traditional maintenance is often made more difficult by inadequate documentation. When maintenance programmers make changes, they often neglect to make corresponding changes to the documentation. With I-CASE tools, the content of the encyclopedia is the documentation. Paper documentation can be generated from the encyclopedia when required. When changes are made to the design, the encyclopedia is automatically updated.

The use of I-CASE tools avoids the spaghettilike mess of the past and promotes cleanly structured engineering with relatively fast and easy techniques for maintenance.

Although in some corporations the main thrust with CASE tools is to increase the productivity of application building, in others the goal is to improve the quality of systems and achieve coordination across a complex enterprise.

The most impressive system is not created with a single design and implementation. It evolves, being improved in many steps at different times and places.

The future will bring impressive software and corporate computer systems, and these will also be grown over many years with many people and organizations adding to them.

It's difficult or impossible to grow software that's a mess. To achieve long-term evolution of software, we need structured models of data and structured models of processes.

Designs too complex for one person to know all the details of must be represented in an orderly fashion in an encyclopedia so that many people in many places can add to the design. The design needs standards and reusable components and an architecture that facilitates the incremental addition of new functions.

For executives to control the behavior of computers that automatically place orders, select suppliers, make trades and so on, the behavior of these systems should be expressible in rules and diagrams that the executives understand.

When It Works, It Really Works

Some corporations have impressive computer systems designed to give them a competitive advantage: Take American Airlines with its on-line terminals in travel agents' offices, or Benetton with its worldwide information system that makes the world activities "transparent" to the decision makers near Rome.

Systems like these demonstrate how a corporation can pull ahead of its competition by using information and automation better. Efficient corporations will evolve computing systems that are worldwide and exceedingly complex, but nevertheless enable procedures to be adapted quickly to changing needs.

To do this requires engineering-style methodologies carried out with automated tools. It requires encyclopedia-based I-CASE tools. Simple software engineering is not enough—to build a computerized corporation we need information engineering. ■

The James Martin Productivity Series, an information service updated quarterly, is available through High Productivity Software Inc., of Marblehead, Mass. (800) 242-1240. For information on seminars, contact (in the United States and Canada) Technology Transfer Institute, 741 10th St., Santa Monica, Calif. 90402 (213) 394-8305. In Europe, contact Savant, 2 New St., Carnforth, Lancs., LA5 9BX United Kingdom (0524) 734 505.