Utilizing Modern Computer Development Tools

In Implementing The

Resource Consumption Model for Process Design (RCM)

Richard Jerz

St. Ambrose University[1]

Davenport, Iowa 52803

Abstract

Industrial engineers need to understand and use modern computational tools in advanced research. The Resource Cost Model for Process Design (RCM) is the result of several years of research into better models for analyzing and selecting process design alternatives. RCM methodology was developed using a computer-based model that included several important programming technologies: object-based and object-oriented techniques, a relational database, and a third-party custom component. A comprehensive modeling of process alternatives would have been very difficult without these technologies.

This paper focuses on the important computer-based technologies used to develop RCM. Research objectives, an RCM overview, an explanation of technologies, and conclusions are provided.

Keywords

Process design, cost analysis, object-oriented, SQL, relational databases, object linking.

1. Introduction and Research Objective

Several varieties of production economic models (e.g., return on investment analysis, break-even analysis, cost estimating, and design for manufacture) aid production process design for product manufacture [1]. These models, however, fail to integrate sufficiently the concepts of cost, production cycle time, production capacity, and utilization. The methodologies typically rely upon these factors being separately analyzed, but do not guarantee that they are. Some methodologies use a narrow production volume range, or worse, one production volume in their calculations, which limits additional insight into economies of scale.

[1] This research was completed as a student writing my doctoral dissertation in Industrial Engineering at The University of Iowa, Iowa City, Iowa.

The research objective was to develop a better methodology for analyzing process design alternatives. It was believed that the new methodology needed to incorporate not only cost analysis, but time and capacity analysis as well since these metrics affect each other. But industry has learned that the lowest cost alternative is not necessarily the best. Some companies may choose a more expensive process alternative because it provides a better time advantage (i.e., faster response). Some may choose a process because it provides greater reserve capacity (e.g., when anticipation of higher future product demand exists). RCM concentrates on thoroughly analyzing cost, time, and utilization factors, and clearly illustrating results. It does not attempt to resolve these conflicting criteria. A better analysis and representation of results, however, should lead to better process design decisions.

RCM was designed to perform detailed calculations across a production volume range. Typically, when one chooses to develop a higher level of analysis, analysis time increases. It was believed that a computer-based model would be needed to perform calculations. Concern over the representation of numeric results led to using graphical results. This research provides both an analytical and a computer model.

A brief overview of RCM is provided to better appreciate the application of the specific computer-based technologies. Discussion of RCM's analytical model is provided in [2]. This paper focuses on several specific computer-model technologies used to develop RCM, since they represent significant changes in the way modern research can be conducted.

2. Overview of the Resource Consumption Model

RCM is a decision-support methodology that provides greater understanding, fidelity and sensitivity analysis to process design analysis than do other traditional techniques. RCM's foundational concept is that part production consumes resources that can be translated into cost, time, and utilization metrics. RCM accounts for all resources, which can be equipment, labor, energy, material, tooling, and other consumables used by process designs alternatives. It characterizes resources generically and avoids the need for terms such as "fixed costs," "variable costs," "overhead," and so forth. Figure 1 shows most of the parameters used to describe resources grouped by function in RCM.

Knowing details about individual resource consumption provides information on where improvement opportunity exists. A resource consumption perspective is not new; this is the perspective taken by activity-based costing proponents. However, RCM recognizes that resources can by consumed by production or the passage of time (i.e., they may spoil or become inert when not used fast enough.) Resource consumption is also affected by overall system behavior.

For each resource, RCM performs quantity-based, time-based, and system-based calculations for many production volume points (in RCM, one-hundred points are used) to determine whether a resource's life is controlled by its inherent productive capacity, by time, or by the system in which it is used. The results are plotted, as shown in Figure 2. On this unit cost graph, the controlling cost is the maximum of all individual lines. RCM can perform calculations and display results for any resource within an alternative. The user can begin to understand better the effect that an individual resource has within the alternative.

RCM provides the ability to graph up to six resources on one graph, as shown in Figure 3. This representation of results provides a better understanding about an alternative's component costs at

various production volumes. Resource calculations can be accumulated to compare alternatives. Results are shown in both tabular and graphical formats. Figure 4 shows how RCM illustrates a cost comparison of alternatives.

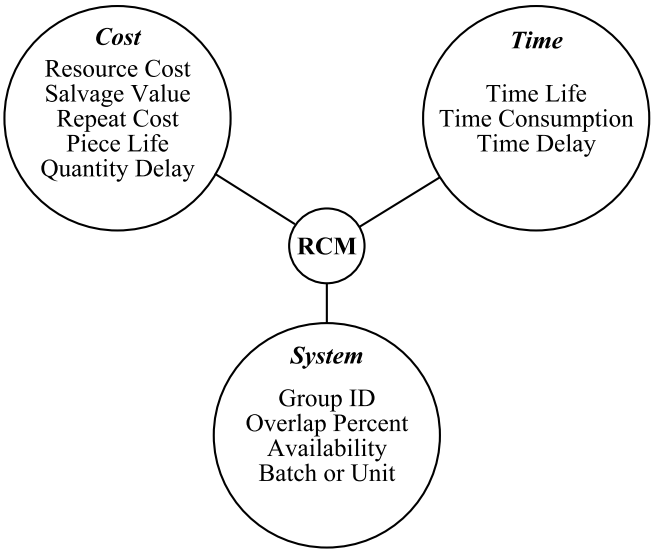

Figure 1. Model parameters by function

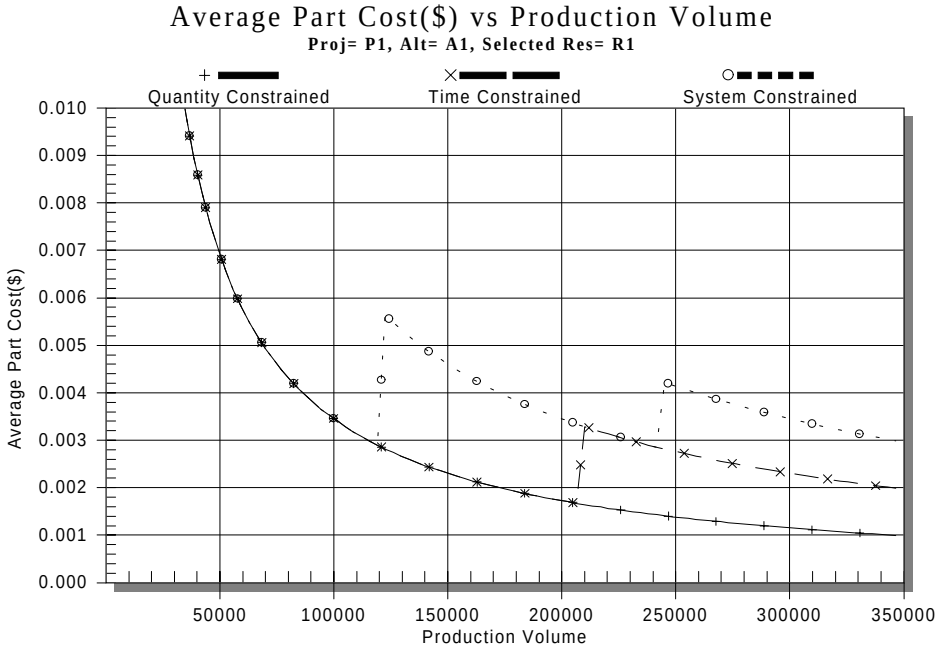

Figure 2. RCM - cost analysis for an individual resource

Figure 2, Figure 3, and Figure 4 provide cost results. Similar graphs are produced for time and utilization results. RCM has many ways in which resources and alternatives can be manipulated and displayed [2]. Only a few are illustrated in this paper.

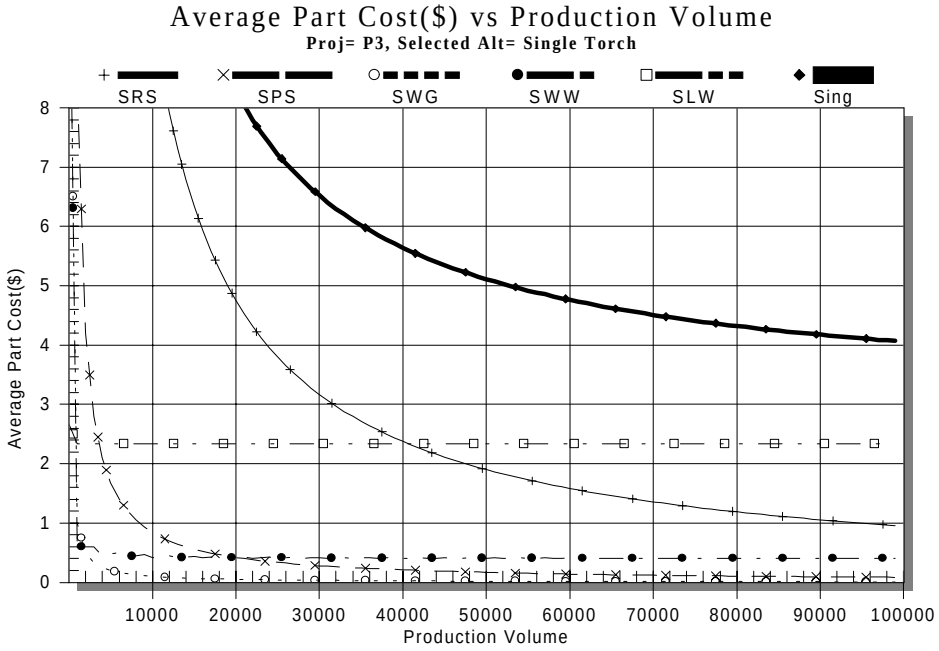

Figure 3. Alternative component costs

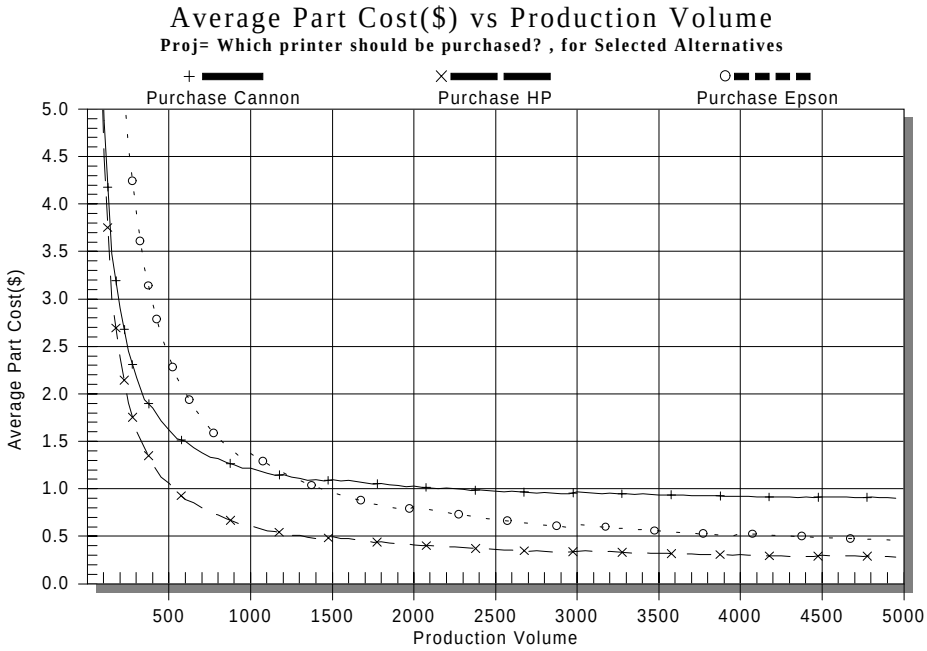

Figure 4. RCM - cost comparison of alternatives

3. Computer Program Development

Today, engineers need to be able to explore many process design scenarios in a timely manner. One of the greatest advantages of modeling RCM on a computer is that sensitivity analysis can be quickly performed. RCM is a computationally extensive methodology. The computer model

provides quick and easy access to all parameter values and results. Without powerful personal computers and software, the development and understanding of RCM would not have occurred.

3.1. Object-Based Programming Application

The modern computing environment uses a graphical user interface (GUI) and a mouse or similar pointing device. It makes little sense to go back to the days of text-based programming environments, which were common over much of the last fifty years. RCM was developed on a personal computer running the Windows95™ operating system. This graphical environment is characterized as object-based, meaning that the user directs a program's actions by clicking on different objects in the display window. For example, Windows programs provide a command button labeled "Exit" or "Quit" that needs simply to be pressed to exit the application. To create a program's functionality, the programmer must master creating and using various visual objects, such as command buttons, pull-down menus, grids, and forms. This is called "object-based programming," but should not be confused with object-oriented programming, described later.

Many software development products were available for the development of RCM. After considering the specific needs for RCM and the direction of programming tools, Microsoft's Visual FoxPro® was selected. This product not only provided an object-based development environment, but also object-oriented programming and a database – both to be described below. Other products at the time this research commenced, such as Visual Basic® and Microsoft Access®, did not offer all of these capabilities.

A challenge for RCM was to design the user interface so that it was easy finding and changing any model parameter values, and easy getting to various results. The solution was a tabbed-page approach called a "page frame," which is analogous to having tabs within a notebook. By using this approach, the programmer can group similar information and the program's work space is great. The user can quickly tab to a page of interest. RCM's page approach is shown in Figure 5. This figure shows only the "Data" page, which is one of six pages. The Plotting page gathers information on what is to be plotted; alternatives or resources; how it is to be plotted; the production volume range; and cost, time, and utilization axes ranges. "Cost," "Time," and "Utilization" pages display graphic results, and the "Summary" page provides tabulated results.

The Data page contains three grids, called grid objects. Three grids were used because RCM data was organized in three relational database tables. Functionally, the user selects a problem to investigate in the "Projects" grid by clicking on it with the mouse. When this happens, all alternatives that the problem contains appear in the second grid. When an alternative in the "Alternative" grid is selected with the mouse, all resources that compose the alternative appear in the "Resources" grid. The three grids make it easy to "zoom in" and "zoom out" to different detail levels. Parameter values can also be changed within these grids.

An example of the Cost page is shown in Figure 6. It becomes obvious from Figure 5 and Figure 6 that the graphics environment and the use of objects accomplishes the goal of making RCM easy to use and informative.

3.2. Object-Oriented Programming

A discussion about object-oriented programming (OOP) is not complete without a discussion of three specific terms always associated with OOP: inheritance, encapsulation, and polymorphism.

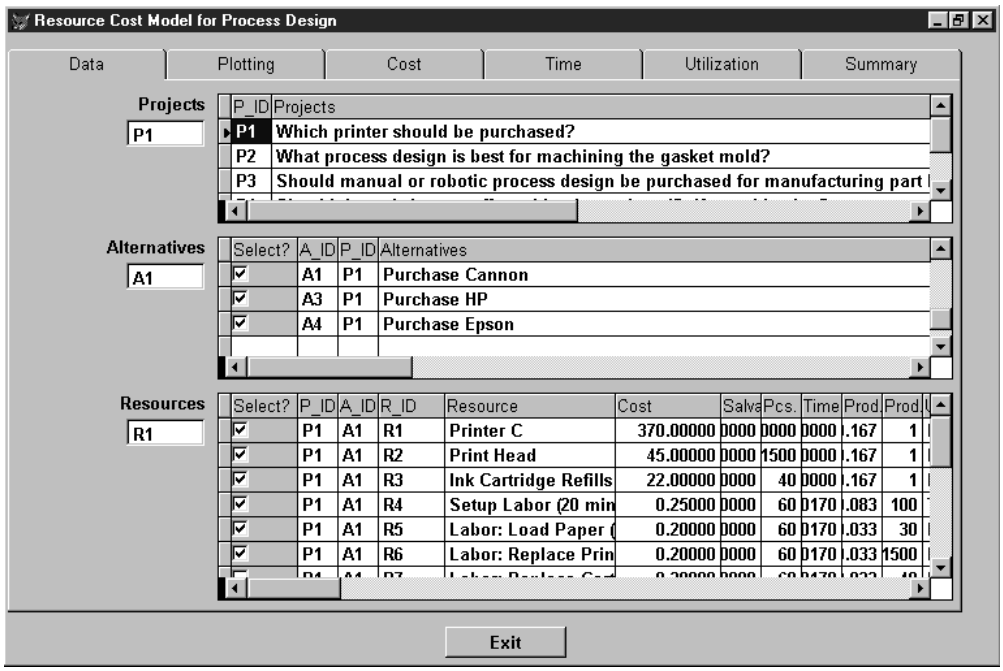

Figure 5. Data page

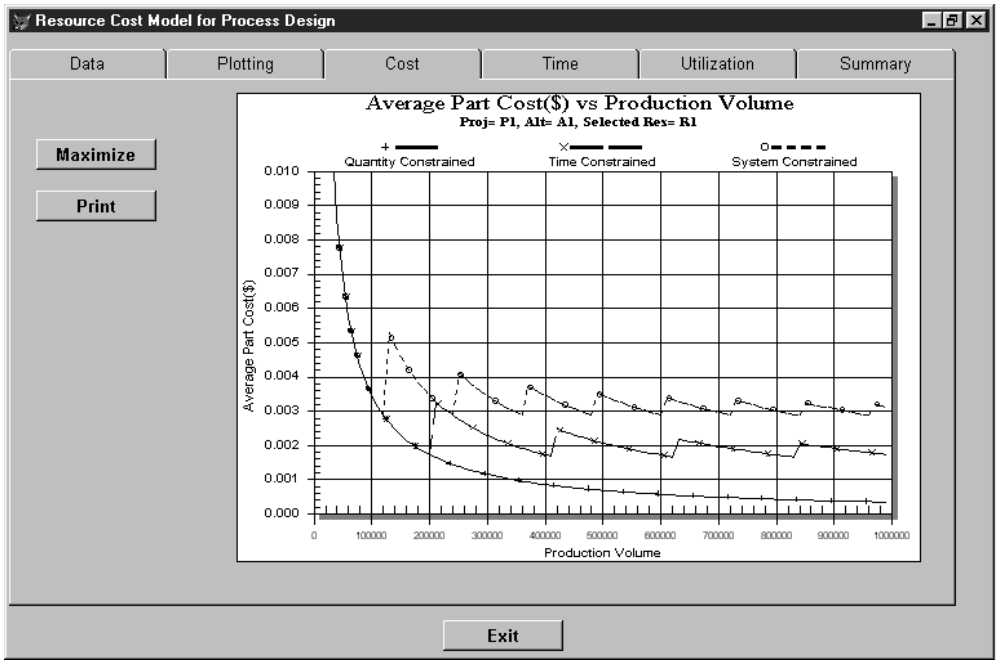

Figure 6. Cost page

Inheritance allows one to create new objects that are similar to, but a little different from existing objects. For example, the programmer may want to define new command buttons that are similar to, yet a little different from the standard button. The programmer may want the button's length and width to be sized specifically for the application. By "sub-classing" the standard button, a

programmer's modification to the button's size automatically changes all instances of the button throughout the application.

Encapsulation, simply defined, is the ability to add properties (i.e., object features, such as value parameters) and methods (i.e., program code) to objects that become active only when a specific object is used, or "instantiated." This is similar to declaring variables "local" or "private" within a subroutine, but encapsulation is better. Instead of having an application made up of module after module of spaghetti code, one object simply sends a message to another object, referencing that second object's properties or methods [3]. The programming burden to remember whether variable names are local or private is removed. Both data and code can be encapsulated within an object. For example, in RCM, a "form" method named RCMCalculations was created to do most of RCM's calculations and analysis. Being encapsulated within the form, the method is always available whenever the form is active. Different objects within the RCM application communicate their values and value changes with RCMCalculations automatically. RCMCalculations creates an array (known in database terminology as a cursor) whenever needed. The Cost, Time, Utilization, and Summary pages, when selected by the user, use this array in different ways to create their visual results. These result pages need not worry about the data structure of the array - they can simply use the array data as needed. Passing of data values is greatly simplified with encapsulation.

Polymorphism is often thought to be a more difficult OOP concept than inheritance and encapsulation, but it is really quite simple. It means that different objects can have methods or properties with the same method name [4]. In other words, the programmer does not need to worry about coming up with unique names for methods encapsulated in different objects. For example, code that performs calculations for one command button can be called "Calculate," while other code that performs different calculations for another object can also be called "Calculate." Each object knows about its own "Calculate" routine. Code is executed by calling the method with commands similar to "ThisButton.Calculate" or "ThatButton.Calculate."

When properly implemented, OOP concepts improve the programmer's efficiency. However, it is a shift from the traditional programming paradigm. For the industrial engineer, the ability to use OOP techniques translates into faster application development. For example, at one point in the development of RCM, the Utilization page did not exist. When the need for this page emerged, it was added to RCM in approximately fifteen minutes of programming. Without OOP, this task might have taken several hours to accomplish.

In its research form, RCM implements only a small subset of object-oriented programming. The development of RCM into a commercially viable product would take greater advantage of OOP.

3.3. Database Modeling Application and SQL

RCM needed to capture cost, time, and capacity information for every resource. A generic characterization of parameters was believed to be beneficial for more efficient use and management of information. Eventually, it was determined that fifteen parameters could adequately define any resource. Since many resources can compose an alternative, and many alternatives can compose a problem, data that RCM must manage can become enormous. A database approach to the problem relieves the burden of information management [5].

A database approach for engineering process evaluation is not common to most traditional methodologies. Most engineering analyses deal with variables and equations, not databases.

However, as the quantity of data available to companies grows, there is a need to understand databases and information management concepts better.

In 1970, E.F. Codd, then a member of the IBM Research Laboratory in San Jose, California, published a classic paper [6]: "A Relational Model of Data for Large Shared Data Banks." It laid down a set of abstract principles for database management: the so-called relational model. The many advantages of the relational approach are too well known to need repeating here. Another particular aspect to relational database research was the development of a standard relational language to define, access, and manage databases. This language eventually became known as SQL, an abbreviation for "Structured Query Language." SQL is now an American National Standards Institute (ANSI) standard [7]. SQL has been implemented in many database products. RCM uses both the relational database approach and SQL to efficiently organize and manage its data.

The SQL SELECT command is used to retrieve information. The basic form of SELECT has three clauses: SELECT <attribute>, FROM <table>, and WHERE <condition>. The modifiers (<>) to the clauses can become very complex, and it often takes much practice to master all its features. Because grouping and aggregation are required in many database applications, SQL has included COUNT, SUM, MAX, MIN, and AVG functions. An example of SQL might help understand its benefits.

RCM needed to calculate the overall controlling cycle time and the minimum availability for alternatives based upon the alternative resource's use of time. RCM does consider that some resources' cycle times can overlap with others, but the determination of overall cycle time and minimum availability is not as simple as one might think. In many traditional programming languages, the programmer would do these calculations using FOR/NEXT loops. In a database, a solution using the SQL SELECT statement becomes easier. The code associated with these calculations is shown in Figure 7

Although the code appears long and "wordy," anyone who understands the SQL SELECT statement would see that the code is simple. Some wordiness in code is the result of descriptive object names that enhance readability. The advantage to using SQL over FOR/NEXT loops is speed – most databases are optimized to execute SQL commands.

The ability to manipulate database information efficiently is not commonly considered a skill that industrial engineers should have. However, engineers who possess this skill can manage information more efficiently and make better decisions.

3.4. Object Linking and Embedding (OLE)

RCM needed to generate specific graphs to represent results. However, Visual FoxPro's graphing capability was very limited. Facing this problem, the author had to decide whether to develop a special purpose graphing routine, or to purchase a commercially available routine. Since one primary research goal is to complete the research in a timely fashion, developing one's own code was an unattractive alternative.

One of Microsoft's strategies for its Windows operating system is the ability of discrete applications to talk to each other. Their vehicle to implement this ability is Object Linking and Embedding, or OLE. OLE is not one tool; rather, it consists of two similar, but different methods to allow applications to talk to each other. The first is the use of OLE controls in applications.

The idea is to be able to extend an application with third-party objects. These third party objects, called ActiveX controls, custom controls, or OCX's can be placed in any application that supports the OLE standard. For instance, a graphics control written to the OLE standard could be placed in a Visual FoxPro form, or a Visual Basic form, or any other design tool that also conforms to the OLE specification. The second is OLE automation, or the ability of one application to run another application. RCM takes advantage of the first type of OLE function.

```
* Calculate the overall controlling cycle time accounting for
* all overlaps.
*  (used for system constraint calculations)

* First, get the overall time for groups in series without overlap
SELECT MAX(nresprodtime*(1-nrespcntover)/nresprodpcs) as ControlTime;
 FROM rcm!resources;
 WHERE Resources.cprojid = lcCurrentProjectID;
   AND Resources.caltid = lcCurrentAlternativeID;
 GROUP BY Resources.ngroup;
 into cursor lnControlTime
* Combine the controlling sequence time and the largest individual
* resource time for an alternative.
select sum(controlTime);
 from lnControlTime;
 union;
 SELECT  MAX(Resources.nresprodtime/nresprodpcs);
 FROM rcm!resources;
 WHERE Resources.cprojid = lcCurrentProjectID;
   AND Resources.caltid = lcCurrentAlternativeID;
 into cursor lnControlTime

* Select the largest time between the sequential resource times and
* the largest individual resource time.
* This become the controlling cycle time.
select max(sum_controltime) as CycleTime;
 from lnControlTime;
 into array lnControlTime

* Now get the minimum resource availability for unit costs (not batch).
SELECT MIN(Resources.nresavail) as MinAvail;
 FROM rcm!resources;
 WHERE Resources.cprojid = lcCurrentProjectID;
   AND Resources.caltid = lcCurrentAlternativeID ;
   AND Resources.lbatch = .F.;
   INTO array lnMinAvailability
```

Note: the *'s in the code are only comment statements.

Figure 7. SQL SELECT code

For this research, a special graphing control produced by Gigasoft, Inc. called ProEssentials [8] was purchased. The cost graph shown above in Figure 6 is a ProEssentials object. Functionally, the graph control contains many properties that define its appearance. More important, data generated by RCM's methods are simply communicated to the graph object. The program code required to produce the cost lines in Figure 6 is shown below.

```
thisform.pgfMain.Page3.olegphCost.XData[k, i-1] = lnProdVolume
thisform.pgfMain.Page3.olegphCost.YData[0, i-1] = lnCost[1],0
thisform.pgfMain.Page3.olegphCost.YData[1, i-1] = lnCost[2],0)
thisform.pgfMain.Page3.olegphCost.YData[2, i-1] = lnCost[3],0)
```

Table 1 provides an example of some ProEssentials property settings as they occur within RCM and a description of what they do.

Table 1. Graph object properties and function in RCM

ProEssentials Command	Function
.points = 100	Plot 100 points
.graphbackcolor = RGB(0,0,0)	Sets background to black
.subsetcolors(0) = RGB(255,255,0)	Sets color of the first plotted line
.subsetpointtypes(0) = 3	Sets first line point type to a solid circle
.subsetLineTypes(5) = 1	Sets sixth line type to a dashed line
.xaxislabel = “Production Volume”	X-axis label

Only a small subset of ProEssentials capabilities was needed for this research. ProEssentials can also produce many other charts, including bar charts, logarithmic charts, pie charts, polar charts, Pareto charts, and real-time data tracking charts. It is a product that serves the engineering community well. It is important that industrial engineers be aware of third-party custom controls and OLE so that one does not reinvent the wheel, and so that applications can be developed more quickly.

4. Conclusions

The programming environment and the set of tools engineers have available today have changed dramatically over the last fifty years. More complex research analysis and models can be developed and their results communicated more effectively. A comprehensive modeling of process alternatives for RCM would have been very difficult without these technologies.

The graphical user interface is here to stay. It has replaced writing lines of codes with an object-development environment. Any engineer who programs computer applications should master graphical environment tools.

There are many advantages to understanding databases and the SQL standard. Engineers will have greater access to and management of company information. Knowledge of SQL will allow one to move from one database environment to another without the need for expensive retraining. Applications written by engineers will have greater portability to different products, and the lifetime of applications will be longer.

Engineers should be aware of third-party custom controls. Custom controls reduce programming time and improve the quality and reliability of the application. In the future, RCM may implement OLE automation. RCM may eventually communicate directly with other applications, such as computer-aided design (CAD), resource planning and scheduling, financial planning, and other computer-aided process planning systems.

5. Acknowledgments

This work has been partially supported by a predoctoral fellowship from the United States Department of Energy (DOE) in "Integrated Manufacturing," 1995.

References

[1] Vonderembse, M.A. and White, G.P., Operations Management: Concepts, Methods, and Strategies, 3rd ed., West Publishing Company, New York, 1996.

[2] Jerz, R., "A Resource Consumption Model (RCM) for Process Design," Ph.D. Dissertation, The University of Iowa, Iowa City, Iowa, December 1997.

[3] Hentzen, W., Programming Visual FoxPro 3.0, Ziff-Davis Press, Emeryville, California, 1995.

[4] Granor, T.E., and Roche, T., Hacker's Guide to Visual FoxPro 3.0, Addison-Wesley Developers Press, New York, 1996.

[5] Elmasri, R. and Navathe, S.B., Fundamentals of Database Systems, 2nd ed., Addison-Wesley Publishing Company, Menlo Park, California, 1994.

[6] Codd, E.F., "A Relational Model of Data for Large Shared Data Banks," Communications of the ACM, Vol. 13, No. 6, June 1970.

[7] Date, C.J. and Darwen, H., A Guide to The SQL Standard, 3rd ed., Addison-Wesley Publishing Company, New York, 1994.

[8] Dede, R., GigaSoft ProEssentials User's Guide, Gigasoft, Inc., Keller, Texas, 1996.