Abstract

In determining whether to develop and market a new product, a crucial task is to estimate accurately life cycle costs and revenues in order to determine the anticipated profitability of the product. In the rapidly evolving computer hardware marketplace, the fast pace of new product introduction requires short planning cycles to keep up with the competition. In an effort to improve planning and estimating, IBM undertook to develop an interactive tool which would incorporate a number of business and cost estimation functions. This tool, the Product Cost Estimator (PCE), was deployed in the first half of 1994 and is currently providing substantial time savings and improved accuracy in the production of business plans and service cost estimates.

Problem description

Prior to bringing a product to market, it is necessary to produce a detailed business plan for the product. This plan is based on anticipated sales of the product, its anticipated selling price, the manufacturing cost of the product, and other costs such as marketing cost, development cost, and warranty costs. Typically, the business plan will include a breakdown of anticipated sales based on distribution channel (e.g., Retail, OEM) and geographic region (e.g., US, ASIA). When a product is ready to market, it is also necessary to have an accurate estimate of anticipated warranty or maintenance costs. These estimates are based on projected failure rates for the product's components and the labor and replacement costs associated with repairing or replacing a failed component.

Current planning practices at IBM often involve the use of various programs, processes, and spreadsheets, with an individual planner typically being the "owner" of a collection of data which are then customized to a particular planning task. Cost data (e.g., cost of product components or subsystems) are obtained from corporate databases and transferred manually to the spreadsheets. The process of producing a business plan in this manner typically takes days to months to complete. Revising an existing plan can take days or weeks and may involve modification of an existing spreadsheet. Thus the current process is time-consuming and contains many opportunities for error. The nature of the process means that it is cumbersome to explore alternative assumptions. The process is essentially serial in nature and requires people skilled in the use of the software planning tools. The result is a process which is not well suited to shorter product cycle times.

In the current environment, rough estimates of warranty or service costs are typically obtained in the early stages of product planning by simply assuming that they will be a certain percentage of revenues or of manufacturing cost. In later stages, a complex service cost estimation program has historically been used to project the number of failures for each product component. This program operates in batch mode and uses a card-image input format. Significant training is required both to use the program and to interpret its output. More recently custom spreadsheets have sometimes been used instead of this program. The current service cost estimation paradigm can often result in inaccurate estimates being made early in a project. The estimates obtained later can be unnecessarily detailed and cumbersome to obtain.

The current software tools used for business planning and cost estimation do not facilitate teamwork.

Objectives of PCE

In developing PCE, IBM had several parallel but interrelated goals. A primary goal was to develop a tool which would help increase the accuracy of cost estimates made early in the business planning cycle so as to reduce risk. Another goal was to develop a tool flexible enough to serve a variety of planning and cost estimation functions, yet be easy enough to use so that it does not require extensive training or experience to use effectively. A third goal was to have a tool which would be interactive and provide facilities for easy creation and evaluation of plans based on alternate assumptions (so-called "What If?" facilities). A final goal was to have a tool which would allow users to access a common database of product component information and would permit multiple users to contribute...
easily to the development of a plan for a particular product.

IBM's corporate management clearly understood the need for success in achieving the goals above. To "enable" this project, both the Manufacturing and Development organizations at the top management level decided to fund the project. This allowed the project team to work independently of any local funding constraints.

Application Description

Components of a Business Plan and a Service Cost Estimate. A business plan is fundamentally a statement of anticipated revenues and costs for a particular product.

Prior to a commitment to introduce a new product, plans need to be developed for the anticipated marketing lifetime of that product. Plans are usually developed on a calendar year basis and, depending on the corporate unit involved, will often be broken down by market segment - which may be based on geographic region or distribution channel or a combination of the two.

Estimated revenue is easily calculated as the product of the estimated volume of sales times the estimated selling price. The selling price may be estimated as a base price less a discount factor which is market dependent.

![Figure 1. Subsystems Window in PCE](image)
Calculation of product costs is more complex. A particular product is usually assembled from subsystems which may be purchased from other business units within IBM or from outside vendors. The total manufacturing cost for an individual product unit is the sum of the costs of the individual subsystems plus a factor for assembly of the finished product. One complicating factor in the computation of manufacturing cost is that the cost of an individual subsystem may change over time.

Warranty costs for a product are based on the anticipated failure rates for the subsystems comprising the product, the length of the warranty period, labor rate, and replacement costs for failed subsystems. Some of these factors may be market dependent - in particular, labor rates and warranty period. The costs - for instance, labor rates - entering into the computation of warranty cost may also change over time.

Figure 2: Object Edit Window in PCE
Development costs and marketing costs may be fixed or may be modeled as a percentage of manufacturing cost and hence may be dependent on the volume of sales. In other cases, these costs may be fixed for the product as a whole, but allocated to the various market segments on the basis of some allocation formula such as the fraction of sales anticipated to occur within that segment.

The ultimate output from a business plan is the “bottom line” - that is, the net earnings before taxes.

The output from a service cost estimate is an estimate of the cost of providing warranty or maintenance service on a product. In addition, a service cost estimate includes an estimate by calendar month of the number of subsystems which will need to be replaced and the number of labor hours which will be required to service machines which have already been sold and for which warranty or service contracts exist. These numbers are used so that appropriate numbers of spare parts can be stocked and the required number of Customer Engineers can be estimated.

PCE is a shell which produces business plans and service cost estimates for new products. The user enters the configuration of the proposed product - that is, the subsystems which comprise it. These subsystems may be selected from a database which contains cost and reliability information. The user then specifies the market segments for the product and enters estimates for sales volume and prices and other costs in those markets. PCE then produces complete business plans and cost estimates based on this data.

At any time after the initial creation of a proposed product, the current specification can be saved to the PCE database. This database contains all user-modifiable information about a product, its subsystems, and business plans, as well as system-maintained information about the user who last modified a product, plan, or subsystem and the last date of modification.

Figure 1 shows the Subsystems Window for PCE. It contains a list of the subsystems which comprise the product. From this window, the user can add or delete a subsystem from the specification of the product. A new subsystem may be selected from a catalog of existing subsystems or from subsystems which are components of a product already existing in the PCE database. When a subsystem is added to or deleted from a proposed product, manufacturing and service costs for that product are recomputed.

In addition, the user may modify cost and reliability information for a component subsystem. Since PCE is intended as a multi-user tool, the specification for a subsystem must be “checked out” from the database before it can be modified. The checkout mechanism ensures that no two users can be modifying the same subsystem simultaneously.

<table>
<thead>
<tr>
<th>Modify Value for Subsystem Cost per Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Date/Value Pairs</strong></td>
</tr>
<tr>
<td><strong>From Date</strong></td>
</tr>
<tr>
<td>01/01/94</td>
</tr>
<tr>
<td><strong>From Date</strong></td>
</tr>
<tr>
<td>01/01/95</td>
</tr>
</tbody>
</table>

**Figure 3: Attribute Edit Window in PCE**
Once a subsystem has been checked out, the user can bring up an edit window to modify various attributes of the subsystem (see Figure 2). The attributes of a subsystem, business plan, or product used within PCE may be modifiable or derived. The values for user-modifiable attributes are displayed in black, those of derived attributes in red. To change a user-modifiable value, e.g., Subsystem Cost, the user mouses on the button to the right of the displayed value. This in turn causes an Attribute Edit window to be displayed (Figure 3).

Most user-modifiable attributes may be time varying or "event-driven." This means that their values can change over time. For example, Figure 3 shows that the cost of the Keyboard is assumed to be $15 starting on 1/1/94 but will increase to $16 on 1/1/95.

The subsystem edit screen allows the user to change the value of an arbitrary number of attributes before "applying" these changes. When the changes are applied, all business plan and service cost information is reanalyzed (by rules). Then, new or updated business plans are produced.

Figure 4: Business Plan Selection Window
In addition to specifying a product in terms of its subsystems, the user must specify the markets in which the product will be sold and for which business plans must be created. Figure 4 shows the Business Plan Selection Window. From this window, the user can create a plan for a new market or delete the plan for an existing market. A market is characterized by a marketing channel and a geographic region. PCE automatically creates "composite" plans which consist of all the plans for a single channel or region. For example, the plan for the "RETAIL" channel is composed of all markets whose channel is RETAIL. By default there is an "Aggregate" Plan for a product.

From this window, the user can also modify an existing plan. The edit window for a business plan is identical in format to that for a subsystem. A number of the attributes of a business plan are ones in which values are specified for a particular time period, with the total value representing the sum or aggregate of the individual values. For example, Estimated Volume of sales may be specified by estimating sales by calendar year or by month or by some other unit of time.

PCE produces summary plans and reports as well as charts and bar graphs. Figure 5 shows a Summary Business Plan for a product. Various bar graphs show the contribution of the individual subsystems to manufacturing cost, warranty parts cost, and warranty labor costs.

![Financial Summary - Leibniz - RETAIL/ EUROPE](image)

Figure 5: Summary Business Plan

Product: Leibniz
Market: RETAIL/ EUROPE

<table>
<thead>
<tr>
<th>Estimated Volume</th>
<th>350.0(K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimated Revenue</td>
<td>$175,000.0</td>
</tr>
<tr>
<td>Base Manufacturing Cost</td>
<td>$ 99,475.0</td>
</tr>
<tr>
<td>Warranty Costs</td>
<td>$ 8,963.3</td>
</tr>
<tr>
<td>Gross Profit</td>
<td>$ 66,561.7 (38.0%)</td>
</tr>
<tr>
<td>Development Costs</td>
<td>$ 1,500.0</td>
</tr>
<tr>
<td>Marketing Costs</td>
<td>$ 9,275.0</td>
</tr>
<tr>
<td>Contingency Costs</td>
<td>$ 500.0</td>
</tr>
<tr>
<td>Net Earnings Before Taxes</td>
<td>$ 55,286.7 (31.6%)</td>
</tr>
<tr>
<td>Installation Costs</td>
<td>$ 0.0</td>
</tr>
</tbody>
</table>

Contribution of Subsystems to Base Manufacturing Cost
One further feature of the system is the capability to specify and store constraints which apply to a group of products. A constraint may be in the form of limit on the value of attributes of plans or subsystems. For example, a constraint could specify that the net earnings before taxes for a product must be greater than a certain value. A constraint may also specify the number of subsystems of a particular type which the product must contain - e.g., all products in a certain category must have an air blower. PCE provides a form-based editor (see Figure 6) for entering the most common kinds of constraints and a constraint specification language for entering more complex constraints.

**PCE System Architecture**

This application is extremely well suited to hierarchical, object-oriented architecture. The basic object classes in the system are products, subsystems, and business plans.

Product objects are organized into a hierarchy. Within a particular business organization or line of business, there are product families. In the PCE software, these are represented as object classes. Within each family, there are product models. Within PCE each model is an instance of its parent class.

Subsystems naturally fall into categories - e.g., keyboard, disk drive, CPU, etc. Within each category there are many subsystem models. Within PCE, each category is represented as an object class, with models represented as class instances.

Business plans are specified on a market by market basis. The plan for each market is an object within PCE.
Each product model is manufactured from subsystems and has business plans. Within PCE these relationships are represented by having each product model object point to a collection of subsystem objects and to a collection of business plan objects. In addition to the objects representing plans for individual markets, there are objects representing the composite plans for channels, regions, as well as the aggregate of all individual markets. These relationships are shown in Figure 7.

Each object category has various attributes associated with it. For example, subsystems have a Name, Subsystem Cost, Warranty Cost, Number of Hours to Repair, among other attributes. A business plan has attributes for Estimated Volume, Development Cost, etc. The value for each of these attributes may be a data structure consisting of a list of dates and a second list of the value of the attribute at each of these dates.

Some object attributes - e.g., Estimated Revenue - are derived from other attributes - e.g., Estimated Volume and Wholesale Price. If each of these attributes were represented as a single number, then revenue would simply be the product of volume and cost. However, by allowing these attributes to be time varying a more complex calculation is required. In this case, PCE forms the union of the dates at which revenue and price are specified. It then computes the volume at the new dates by prorating the originally specified volumes. Price at a newly added date is the same as the price at the closest earlier date at which price was originally specified. Once these adjusted structures have been constructed, the "product" is the product of volume and price on the common interval. Such a computation is shown in Figure 8.

PCE contains a set of functions which compute sums, products, and differences of event driven attribute values. These functions implement spreadsheet-like computations but with far more complex data types. The use of event driven attribute values allows the user to specify values at any desired level of granularity. For example, volumes can be specified as a program total, on an annual basis, month by month, or at any other set of time intervals chosen by the user.

A relational database is used to provide persistent storage for PCE objects and to permit sharing of information among multiple users.
In general, the basic object classes within PCE - product, business plan, and subsystem categories - each correspond to a table in the relational database. In order to reduce storage requirements and reduce the possibility of inconsistency, only user-modifiable attribute values (and a few bookkeeping values) are stored in the database. The values of derived attributes are computed when a product and its subobjects are loaded. Each user-modifiable attribute corresponds to a column in the database. Thus a row in a table maps into a PCE object and contains the value for each attribute having event-driven or time varying value. Corresponding to each of the basic tables is an "event" table which contains date/value pairs for objects and attributes which have events associated with them. Objects and attribute values are linked via system-maintained object ID numbers.

PCE also incorporates a constraint-specification grammar which was written using IBM versions of YACC and LEX. This language allows the user to specify relationships which must be maintained between a product and its subobjects - e.g., a product of a certain type must have a certain number of subsystems of a particular type. The grammar allows constraints to be put on the values of attributes of objects - e.g., the manufacturing cost for a particular product must be in a certain range. The grammar also allows the construction of complex constraints by forming any logical combination of simpler constraints using the conjunction, disjunction, and negation operators. Each constraint is transformed by PCE into the specification of a function which can be applied to a product to test for constraint satisfaction. By default PCE provides for checking whether all applicable constraints are satisfied whenever a product or any of its subobjects are modified. If an applicable constraint is not satisfied, the user is warned of the violation.

Hardware and Software Environment. PCE runs under OS/2, Versions 2.1 and 3.0. It interfaces to a DB2/2 database. The software was developed using ART*Enterprise®, an object-oriented, rule-based language which contains a graphical user interface builder operating on top of Presentation Manager. Considerable procedural code (e.g., to implement the intelligent spreadsheet functions) was written in C. An IBM-internal graphics package, ChartAPI, was used to provide charts and graphs. The constraint language was written using IBM versions of YACC and LEX. The program operates on computers using a 66MHZ DX2 486 processor with 16 MB of real memory.

Benefits from Object-Oriented and Rule-Based Architecture. The problem of producing business plans and service cost estimates is not a new one nor is it one which has a single software solution. However, previous software approaches had proved to have significant disadvantages.

In particular, spreadsheets, appropriate for relatively small problems, become cumbersome as the problem size increases. Adding data, such as an additional market or subsystem, to a proposed product, essentially requires reprogramming the spreadsheet. This requires a user with programming expertise and is a source of delay and error. It would be difficult if not impossible to provide with a spreadsheet the same flexibility in the level of granularity of data that PCE supplies. Exploration of simple alternatives with a spreadsheet - e.g., what happens if we assume we can raise the price by 10%? - is easy. However, making more
and a half years, with requirements being added based on early versions of the software. The basic object input from a variety of users and potential users who tried prototype. However, as time went on, additional system development took place over a period of a two and error-prone in terms of their input requirements. Their operation is never real time, and it requires a trained user to interpret the output.

The benefits of the object-oriented and rule-based approach became clear as the system evolved. PCE system development took place over a period of a two and a half years, with requirements being added based on input from a variety of users and potential users who tried early versions of the software. The basic object architecture of PCE remains unchanged from the early prototype. However, as time went on, additional attributes were added to objects to provide additional detail. This proved to be quite easy to do. Adding an attribute or group of attributes basically required modifying the list of attributes to be edited and writing a rule or two to compute any new derived attributes. This could be done in a completely modular manner, in most cases with no changes to existing code and no changes to the user interface.

The use of a rule-based architecture ensures that computations are truly data-driven. Thus a recomputation is done only when the data required by that computation changes. This is an important for efficiency, since the tool is intended to be interactive.

The use of an object model also permits multiple users to participate easily in the planning or cost estimating process. For example, one user can be modifying service cost information for a subsystem in his domain of expertise, while another is providing the same type of information for a second subsystem; at the same time, the business planner can be specifying volume and price data for a variety of markets.

**PCE - Project History**

In 1991 IBM management understood that the process in use at that time for Initial Business Planning of new products had several shortcomings. One problem was that some checkpoints in the process occurred at the wrong points in time. Another problem was the need for better control over estimation techniques and the need for different business units to use common techniques. Yet another problem was the total elapsed time required to do a business plan. Based on just these perceived shortcomings, management decided to create a taskforce to understand and recommend changes to the planning process. As a result of the taskforce meetings, the PCE project was begun and identified as an opportunity to re-engineer the Initial Business Planning Process itself. The PCE project team presented management with a description of the specific problems that the application would address, a plan to implement the application, and a plan to deliver an initial prototype.

The PCE project team was given the initial “go ahead” to start the project in December, 1991. Work on a prototype started in January, 1992. The prototype won support of corporate management, and the project was given full authorization in June, 1992. The development plan called for several releases of the application, with the final release completed in the first half of 1994.

Before the development of the prototype began, the project team decided that it would use a technique called “Rapid Application Development” (RAD) as the development methodology and that it would establish a user support team with membership coming from representatives of different “lines of business” from within IBM. At this time a general contents plan was laid out for the project. The development plan called for five incremental releases of the application over two and one half years. In the plan, the user teams identified the critical needs for each release. As each release was deployed, a user meeting would be held to validate the contents of the next release. This technique worked!!

As the business world changed in IBM, the team was able to identify changes in requirements and did not get bogged down with a plan cast in concrete. Several times in the development cycle, the project team presented status to management. At the management review checkpoints, management made the decision to fund/support the next release of the application.

The project team included a knowledge engineer (acting as the project manager) and one development process engineer, both from IBM. There was at all times one full time consultant/developer and, at key times, an additional developer, both from Inference Corporation. The user support team had between three and five members during the lifetime of the project, acting in a consulting role.

The first releases of PCE were deployed in the areas represented by the user support team. Initially, this involved three major lines of business in IBM. After each of the first four (DOS-based) releases, the user support team tested PCE to determine its accuracy and to formulate requirements for the next release. Thus the functional capabilities of PCE increased with time. These early releases were considered development releases and were not used to do production work.

The final development release went to the user community in the second quarter of 1994. This release was the first to be based on the OS/2 operating system using DB2/2 as the relational database manager. One line of business in IBM has moved PCE to a production level. In that line of business, several product plans have been completely constructed using the PCE application. Although specific numbers are IBM
Confidential, IBM was able to do business planning with the same degree of accuracy as previous techniques, but with fewer people and in approximately one tenth the elapsed time.

Part of the original agreement with the user support communities in January, 1992, was that maintenance would be done by the project team. As IBM has evolved over the last few years, the ownership of the PCE application has moved into the Integrated Systems Solutions Company (ISSC). ISSC will put service level agreements in place with each user organization for maintenance and support of this application.

Knowledge does change over time, and it is quite clear that every business in the world is going through change. Change will affect the rule set of PCE. The PCE project team firmly believes that because it chose an object-oriented and rule-based architecture, changes can be accommodated straightforwardly in the PCE design. Attributes can be added, rules can be added, screens can be added, and new functions can be added without major restructuring of the application.

**Application Payoff**

The PCE tool is opening up additional benefits for IBM. The previous method of planning was time consuming, labor intensive, and basically serial in nature. PCE is changing that paradigm. PCE can be used by more than one person in parallel. At times, when using the "what if" capabilities, the user can make assumptions about products, plans, or subsystems and explore the ramifications easily and quickly. PCE can guide users, allowing them to understand prior information, showing the users results immediately, and allowing the user to create alternate versions of business plans.

Specifically, as mentioned above, the line of business that has been using the OS/2 release of PCE has seen a reduction in the number of people required to do planning and a tenfold reduction in elapsed time to complete a plan, all with the same accuracy as prior methods.

**Summary**

The use of artificial intelligence, particularly in terms of knowledge representation (objects/frames), inferencing (rules), and constraint processing has enabled IBM to significantly improve its business planning function for evaluating new products. PCE enables easier and more timely exploration of alternative strategies, helping IBM to respond effectively to meet its competitive challenge.

**Acknowledgments.** The authors want to gratefully acknowledge all those who participated in the development of this software and contributed to its success. Thanks to the IBM user support team consisting of Greg Ahlman, Dave Beacham, Jim Henry, Ed Hickman, Bill Palladino, and Beth Reed, to the IBM management team consisting of Mike Davis and Rufus White, and to Inference developers Ken Anderson, Sean Burke, Karen Graham, and Bill Hutton.