The issues of legacy systems become more pronounced at the time of a major IT upheaval such as implementation of ERP or business process reengineering (BPR) exercise. In this changing scenario, there is a need to update the systems and skills and integrate them with the emerging enterprise-wide infrastructure. The main problems with a legacy system are that it remains insulated from the update attempt that largely follows market trend thus rendering it outdated and also that its documentation is poor.

In this paper, the authors share the experiences of a project undertaken in one of India’s leading multinational pharmaceutical companies (MPC) which was to rework on the existing legacy system and design a new application. The legacy system referred to here is the company’s financial accounting system which was developed in 1993. Originally designed in COBOL, it was subsequently improved as and when the finance department put forth its requirements. The major downside of the system was that it had virtually no documentation and no one from the original team that developed the system was still working with the company. This made it all the more difficult to understand and document the system. Also, the system had a high response time thus leading to lower productivity of the data entry staff and other users. Further, it had a limited reporting capability and was basically used for storing financial data.

When this project was undertaken for rework, the MPC was in the process of implementing an ERP package for its manufacturing and, therefore, it was necessary to bring all its applications to the same database structure. The most obvious question was whether to discard the legacy system and implement ERP’s accounting module. The management, however, decided to retain and rework on the legacy system with the intention of integrating the new system with ERP. The driving point in favour of this decision was the realization that the legacy system was regarded as very critical for the accounting function and also that the users had become conversant with the system despite it being not very user-friendly. Also, there was no risk of failure. Incidentally, the review of the legacy system and ERP implementation coincided thereby easing out concerns of managing organizational changes as the company already had its strategy and preparedness in place for the scenario emerging out of ERP implementation.

The computer-aided systems engineering (CASE) tool was chosen for designing the new system because of its inherent advantages in handling software projects which are as follows:

- The well-documented new system simplifies the maintenance jobs and, therefore, fewer people are required for its maintenance (this was the major problem with the previous system).
- It has removed the dependence of the management on a small set of people who specialized in the maintenance of an undocumented system.
- Financial reporting has become easier and better.

The experience on this project made it amply clear that the top management support can make or mar a project. This is one of the most popular hypotheses in the information systems literature which has been found to be true in the case of the MPC.
In recent years, pressures on pharmaceutical companies to produce new products better, faster, and cheaper are very high. While in the past 20 years, the average number of trials required to get a compound approved has increased from about 40 to 60, the number of procedures performed in a given trial has increased by 50 per cent and the number of patients in each trial has roughly doubled. Most processes are too manual and human resources are in high demand (Charkin, 1998). The biggest challenge facing the pharmaceutical industry today is how to manage and maintain the huge volume of documentation that is generated during the lengthy and complicated process of developing a new drug. The gap between clinical information and clinical knowledge slows down the research process. The clinical data available (information) are vast and difficult to manage compared with the usable end-product (clinical knowledge). A positive change can be achieved by speeding up the flow of data through the pharmaceutical product development pipeline and integrating more fully the data systems used at various stages of the process. There is a need for a more process-based approach to the pharmaceutical pipeline which would incorporate both new technologies and new methodologies into the existing systems of data management. The systems should allow considerable flexibility for the end-user in terms of producing the results needed and in the form required (internet/intranet and on-line analytical processing (OLAP)) and allowing ongoing comparisons with other trials or data. This integration gives research organizations the ability to understand and control the methodology, workflow, and processes for clinical data management and review.

While aiming to achieve this, most organizations come across another challenge arising out of those systems developed long ago in older environment but covering a fairly large size of business functions which are very critical as far as the delivery part is concerned. They are known as legacy applications. In today’s high technology environment, distribution and scalability are important to a software system, especially when it is a part of an enterprise system. The limitations that legacy systems built on centralized system often face in using the intranet/internet environment and multi-tier client-server architecture greatly restrict their connectivity, performance, robustness, and scalability (Chiang, 2001). The issues of legacy systems are mostly recognized during a business process reengineering (BPR) exercise. The problem with the legacy system is that changes in markets or strategy render it outdated. The difficulty is also in the form of lack of or poor documentation. Poor documentation leads to improper maintenance and increasing complexity for the information systems (IS) staff despite the fact that these systems might have proved successful in serving the organizational needs over many years. It becomes necessary, therefore, to phase out and remove the dependence of the management over a small group of people or update and integrate with current and emerging enterprise-wide infrastructure (Lloyd, Dewar and Pooley, 1999).

The major concerns in dealing with the legacy systems are as follows:

- integrating the legacy systems with the current IT infrastructure as migration is driven by either a new pressing critical requirement or a desire to replace with a new technology
- deciding on the level of modification of the existing application which is necessary
- examining whether the organization has the necessary infrastructure to support the new technology
- evaluating the skill set required to migrate the legacy system which might lead to painful retraining or change of job or location
- analysing application interface with other existing systems
- developing the new system by the internal team or by outsourcing to external agency.

A legacy system should be treated less as a technical problem and more as a problem driven by social factors such as people, skills, and business processes. Often, dealing with a legacy system has tended to be biased towards a software engineering perspective and to concentrate on technical properties. Some of the common approaches used to migrate legacy systems are:

- Function extraction in which portions of legacy application code are migrated to and recompiled on the new technology.
- Front-ending, commonly called screen-scraping, in which old character-mode terminal presentations are rehosted on graphical PCs.
- Data access, i.e., data warehouse strategies in which PC users wielding intelligent decision-support tools tap legacy data stores.
- Selective redevelopment in which parts of a legacy application are ‘carved out,’ rewritten, and then renewed on PCs and workstations.
Brooke and Ramage (2001) have designed an interdisciplinary approach entitled ‘software as a business asset’ (SABA) which brings together an organizational scenarios tool (based on concepts from the field of organizational development) and a technical scenarios tool (based on concepts from the field of software engineering). It is further possible to reuse the software components of the legacy systems by recovering the behaviour of the systems (Guo, 2003). These tools can be applied in an iterative way so that technical options are tested out against the business needs. It, thus, seeks to mimic the nature of organizational change as far as practicable.

This paper describes the experiences of a project undertaken to solve the problems faced by one of India’s leading multinational pharmaceutical companies (MPC) regarding its legacy financial accounting system. The financial accounting system at the MPC was a critical legacy application developed in-house in 1993. It was originally developed in COBOL and, subsequently, a need was felt to improve the system by migrating it to a better platform. Sybase 10.0 was chosen as the ‘better’ platform at that time since it was one of the leading relational database management systems (RDBMS) available. Sybase is known to be one of the most technically advanced RDBMS of all times, yet, it has problems of low user-friendliness and lack of an entire suite of compatible products like MS SQL server.

In this paper, we share the experiences of analysing the existing financial accounting application and designing a new application which:

- is built on the Microsoft platform
- has a client server architecture
- interfaces with other Microsoft products
- is well documented and easy to maintain
- provides extensive reporting capabilities
- is user-friendly with a graphical user interface
- has an acceptable response time.

The entire approach followed in this paper has the intent of the SABA approach suggested by Brooke and Ramage (2001).

REVIEWING THE LEGACY FINANCIAL ACCOUNTING SYSTEM

Company Profile

Founded in 1961, MPC has historically been an India-based generic drugs company manufacturing drugs and active pharmaceutical ingredients. The company is now in the process of moving up the value-chain to become a research-driven pharmaceutical company with an increasing presence in the international marketplace. It has operations in 34 countries worldwide including manufacturing infrastructure in seven countries around the world. For the 12 months ended December 31, 2003, global sales at US$ 972 million registered a robust growth of 27 per cent. Overseas markets accounted for 76 per cent of the global sales. USA and Europe contributed over 50 per cent to global sales with a combined turnover of US$ 500 million. The growth rate is over 25 per cent over the past two decades and its major markets are USA, Western Europe, Russia, and China. With the recent acquisition of RPG (Aventis) SA, MPC has become the fifth largest generic company in France and has considerably expanded its overall presence in Europe. With a target of achieving US$ 1 billion sales in 2004, the company is now aspiring to achieve US$ 5 billion sales by 2012 and be amongst the top five generic players. On the domestic front, it has enhanced its therapeutic span through brand acquisitions/company takeovers. It has entered into a deal with Bayer AG, Germany, to develop Ciprofloxacin. This has upgraded its status to a transnational research-driven pharmaceutical company. It is structured across four international regions: India/Middle East, Europe/CIS/Africa, Asia-Pacific, and North and South America.

MPC’s major strengths and core competencies lie in the development and commercialization of generic active pharmaceutical ingredients and dosage forms in line with the international regulatory and quality requirements. In pursuance with its corporate mission of becoming a research-based international pharmaceutical company, it has recently established infrastructure and capabilities in the area of new drug discovery research and novel drug discovery systems. The company’s strategy involves an intelligent anticipation of emerging markets which have the maximum potential as well as a review of the increasing needs of the developing markets. The strategic choices made by the management require a few key transitions towards which the company is already repositioning itself (Figure 1). They relate to creating value-added products and migrating to more powerful markets. MPC has played a pioneering role in gaining acceptance for the Indian bulk substances in the advanced markets of the world. It is now set to repeat this success with dosage forms too.
IT Policy

MPC has an IT-friendly work environment and is known to have been an early adapter to the computer culture. Since the early seventies, its IT budget has been consistently increasing. The top management understands that IT would give impetus to the company in its move towards a better growth path. To realize this, the company has recently combined the two functions of strategic planning and information services and has enhanced the IT budget which is now more than 1.5 per cent of its annual turnover. The current focus is on upgrading to new technology and enhancing its IT infrastructure in order to cope with the ever increasing competition in the pharmaceutical industry. When the company started its operations, it built an IT infrastructure using mini computers which were very popular at that point of time and were used primarily for its drug research and development activities. Today, its technology infrastructure is a mix of best-of-breed technologies. It has built up a heterogeneous network that spans across the globe (see Box 1 for a detailed description of the network).

Legacy Financial Accounting System

The accounting information system (AIS) is often the most widely used and the largest of the information sub-systems in a business organization. In some organizations, AIS is the only formally designated information system. In organizations with a formal management information system, accounting plays a key role in system administration and operation. AIS serves two categories of users — those external to the business organization and those internal to it (Figure 2). The subset of accounting that is concerned with the information needs of external users is known as financial accounting. Management accounting is the subset of accounting concerned with internal information needs. AIS supports the day-to-day operations by collecting and storing data...
regarding an organization’s transactions. The information produced by the AIS is made available to all levels of management for use in planning and controlling an organization’s activities. AIS has all the characteristics of a management information system. The major difference between the two is one of scope. While the management information system encompasses all data entering the organization, the accounting information system is concerned with financial information generated from processing transaction data. As a result, AIS is often referred to as the transaction processing systems.

The integrated financial accounting system at the MPC was an online interactive system built in the year 1993. This was the first information system developed in-house by the IT department of the organization. The system was based on Sybase 10.0 running on SCO UNIX, and consisted of about 130 tables and 800 procedures. The entry of the data into the system was online with various validations and checks provided for at the time of data entry itself. This was a ‘green screen’ application with the execution of all jobs being done through menus. Various financial documents could be entered into the system through forms. The financial module generated a number of reports varying in nature and frequency. There was a provision to print as well as view each report on the screen. ‘Selecting the Query’ option generated a large number of queries. Some special features of the system were recovery procedure, automatic cheque printing, authorization of documents, mirror accounting, etc. The system had virtually no documentation. It had been enhanced a number of times by the IT department.

Problems with Legacy Financial Accounting System

- The accounting system was the first system built in-house in COBOL in the year 1993 when the users were not very IT-savvy. They were not very clear about their requirements as well as what the computing power could do for them. As a result, the system was built in phases as and when the finance department put forth its requirements. After using the system for a few years, the users became knowledgeable enough to demand a new system fulfilling their enhanced requirements of improved access to the data. Sybase 10.0 was chosen as the ‘better’ platform at that time since it was one of the leading RDBMS available.
- Since the IT department was not an experienced one, the Sybase system was not developed entirely...
in-house but by a combination of internal and external agencies. Since several people had worked on it over the years, the system had expanded so many times that no one really knew how it worked. The major downside was that the system had virtually no documentation. None from the original team that developed the system was still working with the company. This made it all the more difficult to understand and document the system.

- System maintenance was in the form of writing new SQL modules whenever the need arose for a particular kind of report or functionality. As the accounting system was large in size (about 130 tables and 800 procedures), the effort to locate the required procedures was more than rewriting the changed procedure again and this was what precisely was done. The system was a typical ‘green screen’ application with a character-based interface designed on Sybase’s APT workbench. It was entirely menu-driven and ran on a SCO UNIX platform off a central server. The user nodes were dumb terminals with a few special key mappings. The system had a high response time thus leading to lower productivity of the data entry staff and other users.

- The system had limited reporting capability and was basically used for storing financial data. All the financial documents were filled in on paper and approved before being keyed into the system. In order to cater to management reporting needs, as a stop-gap arrangement, the required tables of the database were duplicated on an SQL server database every night automatically using a scheduling program. An MS Access front-end interfaced with this SQL database using open database connectivity (ODBC) and prepared the desired reports to be given to the responsibility cost centre (RCC) heads every week.

- MPC was in the process of implementing an ERP package for its manufacturing which would also be extended to its research centre in the future. Thus, it was necessary to bring all its applications to the same database structure.

- The company had decided to have a common Microsoft platform for all the products it was using. The current system, being on SCO UNIX and Sybase, would be incompatible with the rest of the systems. Thus, there was a need for migration. It had also implemented SAP R3 for its manufacturing facilities at Mohali and Dewas. The implementation also rolled over to the research facilities including the one at Gurgaon where this project was undertaken. Therefore, it was required to interface the financial accounting application with SAP.

- The system was to be linked to the materials management system, rent and lease system, fixed deposits system, etc., so that it would be capable of supporting data exchange between a multitude of information systems. Thus, migration became essential to remove the incompatibility between these systems and to enable improvements in terms of data accessibility, data integrity, system availability, trouble shooting, and data integration; reduce training costs due to GUI interface; reduce cost associated with the distribution of frequent software updates; and ensure fulfilment of requirements of the new system.

Choice of Solution

Attempting a solution to the problems of the legacy accounting system should begin by considering it in a wider framework of which the software is merely a part. Other components might include people, expertise, hardware, data, business processes, business environment, and approaches to software maintenance and development. Understanding a legacy system requires taking account of its relationship to the changes occurring among all these components. This would guarantee that any technical changes undertaken are driven by business needs and not vice versa. For the management, it is important to ensure that assets remain assets and potential liabilities are minimized.

One pertinent question in this context was whether to discard the legacy system altogether and build the functionalities within the SAP R3 environment. The management evaluated this option and decided to retain and rework on the legacy system with an intention of integrating the new system with ERP. The driving point in favour of this decision was the realization that the legacy system was regarded as very critical for the accounting function and also users had become conversant with the system despite it not being very user-friendly. However, with changes happening on all the dimensions with the company’s decision to move to ERP system, a review of the legacy system became inevitable. Also, there was no risk of failure.

Incidentally, the review of the legacy system and
ERP implementation coincided thereby easing out concerns of managing organizational changes as the company had already put in place its strategy and preparedness due to the emerging scenario of ERP implementation. These included top management steering the core committee represented by all the major departments, budget allocation, necessary business process reengineering, upgradation of IT infrastructure, and skill upgradation programmes and other related initiatives.

Since the legacy system had already undergone several iterations in the past to accommodate necessary changes from time to time, no major process change was required. Hence, most of its functionality was to be retained. Further, regular meetings with the users and their close involvement in the pre-design stage helped in achieving clarity about the system in its totality.

Further, a consideration of the issues of migration for a legacy system throws up the following solutions:

- **Bringing in a state-of-the-art package** to replace the entire system. This might not be a realistic solution.
- **Rewriting** some of the software to provide an enhanced resource within the existing system.
- **Connecting the state-of-the-art software and hardware** to the system in order to enhance it in a more comprehensive fashion than rewriting some of the software.

There are numerous permutations of the last two solutions with the precise permutation most applicable to a particular organization being entirely dependent on its specific and unique needs. To the design team, a combination of the second and the third solution seemed most appropriate for the following reasons:

- The number of new or changed business requirements is less than the amount of function to be retained. Hence, modifying the existing application is the best solution.
- The application is mission-critical; modification of the existing system minimizes the risk of failure.

The computer-aided systems engineering (CASE) tool was chosen to deal with designing the new system of MPC because of its inherent advantages in handling software projects. Such kinds of software projects are often very large involving a large number of people and span over many years. The development of these systems, if conducted in an *ad hoc* fashion, often proves disastrous. Hence, choosing an appropriate technique and tool is important. The CASE tool offers an opportunity to alleviate these problems and is most effective in managing changes. The system life cycle cost distribution shows that 67 per cent of the cost is allocated to the maintenance phase. This was found to be particularly true in the present case. Due to the absence of documentation, the complete functionality of the system could not be exploited and hence, even now, whenever requirements of some functionality crop up, new modules are written. The number of procedures in the system continues to grow making the task of maintenance virtually impossible. The Oracle’s Designer/2000 was chosen for designing the new system. The present case requires this CASE tool as it has the ability to customize the data dictionary and analysis prompting within graphs and depth of description available in the data dictionary.

### PROJECT IMPLEMENTATION

The project was initiated by gaining familiarity with the Sybase application suite which includes Sybase10.0, APT5.2, Database Workbench (DWB), and Reports Workbench (RWB). It was important to do so even before one could start understanding the working of the application. The application was explored next to see its operation and features. This involved understanding the inherent structure and logic of the application, its forms, inputs, reports, etc. After achieving an acceptable level of understanding of the Sybase tool set and the application, the next task was to understand the processing which was taking place behind the screen. Two basic types of procedures were found — Database procedures and APT procedures. The APT procedures were available as text files which had to be tracked and copied on to a Windows platform for ease of browsing. The definitions of the database procedures had to be extracted with the database using built-in utilities. The same procedure had to be repeated for extracting the definitions of tables, triggers, rules, indexes, datatypes, and views.

The above exercise resulted in a huge amount of information (code for about 800 procedures, definitions of about 130 tables and so on). Next, the front-end objects were to be mapped to the back-end objects so as to find out which form or menu item calls which procedure when different events are actuated by the user. This was done by studying the code and running the forms in an ‘informational’ mode. There were 60 forms and each had elements of entry processing, key processing, run processing, and exit processing. This means the procedures that are called when the form is activated, a special key is
pressed, commands are run, and the form is deactivated, respectively. Further, many of the APT procedures call database procedures for the actual retrieval and update of data. Even they had to be mapped.

The output of this phase was a holistic understanding of the entire application. Ultimately, due to the sheer number of these procedures, they had to be documented in the form of hyperlinks to the actual text files which were generated in an Access database. Using this database, one could select a menu item of the financial accounting application and get a list of hyperlinks for the related form and different procedures. The list of menu items and their related forms and procedure is not shown here due to space limitation. Another result of this exercise was the isolation of the objects that were actually used in the application.

The next step was to understand the actual financial accounting procedures at the MPC. This was done by talking to the direct as well as the indirect users of the system. Besides understanding the actual financial system, this phase also served as an opportunity to evaluate the existing system deficiencies and the new user expectations and requirements. Along with the process of the existing system documentation, the CASE method was also invoked. The CASE method supports almost the entire range of software development process such as conceptual modeling, requirement analysis, system modeling and design, and debugging and testing with the help of editors. The output of the process was entity relationship diagrams, data flow diagrams, and function hierarchy diagrams.

Some of the major processes identified were:
- login process
- validation of data
- budget update
- report generation
- query processing
- table updation.

Some major groups of data flows identified were:
- budget
- requests for queries
- requests for reports
- bank payment vouchers (BPV)
- cash payment vouchers (CPV)
- journal vouchers (JV)
- payment advice (PAD)
- provisional journal voucher (PJV)
- reports.

The analysis evaluation was done by discussing with the system users and referring to the existing system to ensure that all aspects had been covered. Decisions were also made concerning the following design standards:
- screen layout (including colours, fonts, and buttons)
- navigation tools (including menus and buttons)
- help systems
- documentation
- functionality
- coding standards
- naming conventions.

The system was designed to have the following features:
- Online data entry.
- GUI interface.
- Validation and checking — Through this feature, various items in the document are checked for their validity and correctness to the extent possible at the time of entry itself.
- Integration with other systems — This helps in interfacing with materials management, payroll, fixed deposits, and rent and lease systems.
- Recovery procedure — In case of unavoidable break or corruption of data, this ensures that data integrity is maintained till the last entered document.
- Automatic cheque printing — This provides for automatic cheque printing. An automatic cheque may be printed on entry of bill entry voucher.
- Employee and supplier outstanding — This feature monitors supplier and employee outstanding transactions and provides for automatic as well as manual matching of outstanding transactions.
- Authorization of documents — This enables checking for the validity of the approval code or the amount limit up to which the approving authority is authorized to pass documents.
- Audit trails.
- Automated messaging — This facility helps to automatically generate messages for the concerned people on occurrence of a pre-specified event.
- Mirror accounting — In case vouchers are being passed for a different activity, the mirror account code for that activity is fed into the system. At the end of the month, mirror entries from the parent division and corresponding entries for the child division are created which, in turn, are merged with the transactions of the child division. This process
saves the labour of making manual JV’s for the child division.

- **Multiple runs.**
  
The reports of the financial accounting system were as follows:
  - **Daybooks** — Daybooks for any type of document (CPV, BPV, JV, PV) can be generated as and when required. The user has an option of printing the same for a given range of date and documents.
  - **Cashbook** — Cash is available month-wise as well as day-wise. The cashbook reflects the opening balance, lists of all the cash transactions date-wise, and the closing balance.
  - **Bankbook** — The bankbook is available month-wise for a range of bank codes. The bankbook reflects the opening balance, lists of all the bank transactions date-wise, and the closing balance.
  - **Trial balance** — The trial balance summarizes the general ledger account heads by listing out for each account head the opening balance, debit, credit, and the closing balance.
  - **General ledger** — This report reflects the detailed movements in each account during the month/period. For each and every account, starting with opening balance, the system calculates the net closing balance after listing out all transactions affecting the account. The printing of the general ledger is account-wise, activity-wise, and RCC-wise.
  - **Sub-ledger** — This statement gives an image of the dealings for a particular control account (party, medical, and vehicle) during a given period. This report gives the opening balance, list of transactions, and the closing balance.
  - **Employee ledger** — This statement is an image of the dealings with employees for a particular control account (staff advance, travel account) or a range of control accounts during a given period.
  - **RCC-wise expense variable** — RCC-wise expense variable shows the break-up of expenditure incurred account head-wise for a particular RCC. For each RCC, the controllable expenses and personnel cost are listed separately.
  - **Account-wise expense variable** — This shows the break-up of the expenditure incurred RCC-wise for a particular account.
  - **Sub-code trial balance** — This report summarizes the sub-code-wise status of a particular sub-code control general ledger account head by listing out for each sub-code the opening balance, debit, credit, and the closing balance for the month.
  - **Missing documents list** — This report lists down all the documents (month-wise and document type-wise) which are missing in the system during a specified period. This report is useful in tracking the documents which have not been entered into the system.

**Effort Analysis**

There are several effort estimation methods which have been widely disseminated and experienced that include ‘function point’ (Albrecht, 1979) and ‘constructive cost model’ (COCOMO) (Boehm, 1981). In the present case, an attempt is made to estimate the effort involved in developing the above system by using function point to measure the productivity aspect of the effort and the ‘COCOMO model’ to estimate the actual effort (person-month) or cost. These methods are suitable as they follow the basic principle of ‘break-down – sum up.’ The functionality of the intended system is broken down to smaller better manageable units (e.g. called ‘functions’), the complexity of which is estimated and then summed up and modified by one or several factors representing general system and project characteristics.

**Function Point Analysis**

This enables system designers to estimate a project directly from the attributes of a project that are easily counted usually early in the project life cycle which are internal logical files; external input; external output, and external inquiries (Box 2). For the financial accounting system at the MPC, the five information domain characteristics described in Box 2 are determined and counts are provided in the appropriate table location as shown in Table 1 of the Appendix. In database development, certain objects can be equated with the function point counterparts. Reports can be roughly equated with external outputs; forms and dialogue boxes with external inputs, and tables with logical internal files. External interface files are files that communicate to other applications. External inquiries are screens which let the user ask for help in the system.

Each of the five information domain characteristics is individually assessed for complexity and given a weighted value that varies from 3 (for simple external inputs) to 15 (for complex internal files). Here, the weighted values proposed by Albrecht (1979) are used.
The unadjusted function point (UFP) count is computed by multiplying each row count by the estimated weight and summing all values (details of the computation are explained in the Appendix):

\[ UFP = \sum \text{(number of the information domain characteristics)} \times \text{(weight)} \]

This UFP needs to be further modified by factors to take into account the complexity introduced in the project.

\[ FP = UFP \times [0.65 + 0.01 \times \sum(Fj)] \]

For the financial accounting system at the MPC,

\[ UFP = 892 \]
\[ FP = 892 \times (0.65 + 0.01 \times 56) = 1079 \]

The number of function point (FP) is used to estimate the final code size. Past data indicate that one FP translates into 20 Line of Code (LOC). Hence:

\[ LOC = 1079 \times 20 = 21580 \]

**COCOMO Model Analysis**

The COCOMO cost estimation model is used by thousands of software project managers and is based on a study of hundreds of software projects. The most fundamental calculation in the COCOMO model is the use of the effort equation to estimate the number of person-months required to develop a project. The COCOMO calculations are based on the estimates of a project’s size in delivered source instructions (DSI). Most of the other COCOMO results, including the estimates for requirements and maintenance, are derived from this quantity. DSI is defined as follows:

- Only source lines that are delivered as part of the product are included — test drivers and other support software are excluded.
- Source lines are created by the project staff — code created by application generators is excluded.
- Declarations are counted as instructions.
- Comments are not counted as instructions.

Generally, a project is considered to be developed in one of the following three modes: Organic, semi-detached, and embedded modes. The financial accounting system is taken to be in the semi-detached mode for which effort is empirically established as:

\[ \text{Effort} = \text{EAF} \times 3.0 \times (\text{KDSI})^{1.12} \]

where

- Effort is in units of person-months
- EAF is the effort adjustment factor derived from the cost drivers
- KDSI is thousands of delivered source instructions

For the present system, effort is estimated as follows (details of the computation are explained in the Appendix):

\[ \text{Effort} = 2.37 \times 3 \times 22 \times 1.12 = 175 \text{ person-months} \]
\[ \text{Duration} = 2.5 \times 175 = 16 \text{ months} \]
\[ \text{Number of people} = \text{Effort} / \text{Duration} = 11 \]

These estimates provide important planning information useful in assessing the completeness of the review and testing activities. The project manager is able to adopt the project work flow and technical activities in order to minimize the development schedule by making the necessary adjustments to avoid delays and mitigate potential problems and risks. Further, these results are useful in assessing product quality on an ongoing basis and, when necessary, the technical approach to improve quality.

**LEARNINGS FROM THE PROJECT**

The above project was initiated with the consent of the management and is now under implementation, a phase where the major challenge lies. The well-documented new system has simplified the maintenance jobs and, therefore, fewer people are required for its maintenance. This was the major problem with the previous system. The new system has also removed the dependence of the management on a small set of people who specialized in the maintenance of an undocumented system. Also, financial reporting has become easier and better.
Replacement of the old system by a new system required training of all the users who needed some time to reach the same level of efficiency as they had achieved with the old system. During that period, work and performance suffered to some extent and the users expressed feelings of dissatisfaction with the new system. However, since they were involved at all the stages of project implementation, the magnitude of these feelings was less. Overall, the users were cooperative with each other and with the design team. This has led to successful adoption of the new system.

Replacement of the old system also resulted in a shift in the job of the present maintenance staff. They were made to pick up new skills for the system with the promise of better avenues. The IT team was also expanded in view of the ERP implementation requirements and some members from the team were identified to look after the specialized jobs of database management, network architecture, trouble shooting, etc.

The experience on this project made it amply clear that the top management support can make or mar a project. This is one of the most popular hypotheses in the information systems literature and has been found to be true in the case of MPC. The top management of MPC has traditionally been very positive about the new IT application as it believed that this would lead the company to rapid growth. It is further evident from the fact that the company has recently combined the two functions of strategic planning and information services by investing around Rs. 400 million in ERP implementation, upgrading the network infrastructure, and adding VPN to its network design. The latest in line is the deployment of wireless LANs to empower the senior level executives by running all the mission critical applications that included indigenously developed applications, SAP, and regular applications being run on the wired network and recently added business intelligence solutions.

With improved network infrastructure, it is possible to integrate these applications with the web and the internet. While the earlier information systems were isolated and difficult to access, they can now be accessed using the web and interfacing software. Internet technologies have provided us with universal connectivity, proven and cost-effective middleware, hardware independence, and shareability of applications. This has resulted in the web becoming the de facto backbone environment for distributed computing applications in an enterprise and the browser becoming the universal end-user interface. While new applications can be developed for the web environments, legacy applications have to be migrated to the web environment. Legacy application migration to web environments is the third largest area of investments planned by enterprises after ERP implementation in the next few years.

The integration of legacy application into the web environment can be achieved in various degrees depending upon end-user requirements. The simplest method is to provide a web interface for the legacy application. The end-user would be able to access the legacy application through the browser. Alternatively, legacy data alone can be made accessible through the web. That is, a newly developed web application can have access to legacy databases. In a third scenario, there could be two-way communication between the newly developed web application and the legacy application for exchanging programs (code) and data.

The integration described above is achievable by a technique called wrapping where the legacy code and data are encapsulated using a wrapper and deployed on to the web. These wrapped contents can be accessed using standard APIs and other middleware available for web environments. They may be designed for various operations such as application, services (system-level services, e.g., transaction services, file system services), and database wrappers (wraps only data).

**CONCLUDING REMARKS**

Legacy systems are known to pose serious challenges of various kinds. This paper describes a case where the review of a legacy financial accounting system was delayed for a long time mainly because of absence of any documentation. On the other hand, the system had been developed over the years by different people and modified several times. While working on this project, we realized that legacy systems should be handled more as an organizational problem and less as a technical one. The new system resulting out of this consideration would then succeed in meeting the requirements of the changed scenario. The system discussed in this paper is now well documented and is built on a Microsoft platform incorporating the user requirements. The documentation yielded the actual system components that were being used. These components were helpful in designing the new system since the functionality required in the new
system was not much different from the old one. The major limitation of this project was the inability to reverse-engineer the existing application using a Designer to recover most of its components. This approach would have accelerated the development since the new application was required to keep a significant portion of the old system intact. It could not be done due to technical difficulties and non-availability of certain Sybase drivers. Finally, since the direct users of the accounting application are only a handful, discussions about the new requirements were limited. A large number of users would have produced more issues and avenues for improvement. Efforts can be made to roll over this application to other parts of the company over various locations and have a centralized database so that the entire accounting process is integrated. This has been made easier by the fact that SAP R3 has already been implemented.

Appendix: Effort Estimation for the Financial Accounting System at the MPC

Function Point Analysis
This estimates attributes of a project that are easily counted such as internal logical files, external interface files, external input, external output, and external inquiries. These are individually assessed for complexity and given a weighting value resulting in the unadjusted function point (UFP) count:

\[
\text{UFP} = \sum \text{(number of information domain characteristics) x (weight)}
\]

For the financial accounting system at the MPC, the UFP total is given in Table 1.

Table 1: UFP Total

<table>
<thead>
<tr>
<th>Object</th>
<th>Number</th>
<th>Weightage Factor</th>
<th>Simple</th>
<th>Average</th>
<th>Complex</th>
<th>UFP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forms/Inputs</td>
<td>51</td>
<td>X</td>
<td>-</td>
<td>-</td>
<td>6</td>
<td>306</td>
</tr>
<tr>
<td>Reports/Output</td>
<td>26</td>
<td>X</td>
<td>-</td>
<td>-</td>
<td>7</td>
<td>182</td>
</tr>
<tr>
<td>Tables/Files</td>
<td>27</td>
<td>X</td>
<td>-</td>
<td>10</td>
<td>-</td>
<td>270</td>
</tr>
<tr>
<td>External interface files</td>
<td>10</td>
<td>X</td>
<td>-</td>
<td>7</td>
<td>-</td>
<td>70</td>
</tr>
<tr>
<td>External inquiry</td>
<td>16</td>
<td>X</td>
<td>-</td>
<td>4</td>
<td>-</td>
<td>64</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>892</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This UFP is then further modified by factors whose value is based on the overall complexity of the project. The factors correspond to 14 questions concerning the overall complexity (Table 2), which may affect the estimation of the effort. Each of the factors is rated on a scale that ranges from 0 (not important or applicable) to 5 (absolutely essential).

Table 2: Technical Complexity Factor

<table>
<thead>
<tr>
<th>Function</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Data communication</td>
<td>5</td>
</tr>
<tr>
<td>2. Distributed functions</td>
<td>4</td>
</tr>
<tr>
<td>3. Performance objectives</td>
<td>5</td>
</tr>
<tr>
<td>4. Heavily used configuration</td>
<td>5</td>
</tr>
<tr>
<td>5. Transaction rate</td>
<td>5</td>
</tr>
<tr>
<td>6. On-line data entry</td>
<td>5</td>
</tr>
<tr>
<td>7. End-user efficiency</td>
<td>4</td>
</tr>
<tr>
<td>8. On-line update</td>
<td>5</td>
</tr>
<tr>
<td>9. Complex processing</td>
<td>5</td>
</tr>
<tr>
<td>10. Reusability</td>
<td>4</td>
</tr>
<tr>
<td>11. Installation ease</td>
<td>2</td>
</tr>
<tr>
<td>12. Operational ease</td>
<td>2</td>
</tr>
<tr>
<td>13. Multiple sites</td>
<td>2</td>
</tr>
<tr>
<td>14. Change facilitation</td>
<td>3</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>56</strong></td>
</tr>
</tbody>
</table>

The following relationship is used to compute the function point (FP):

\[
\text{FP} = \text{UFP} \times [0.65 + 0.01 \times \sum (F_j)]
\]

The \( F_j \) (j=1 to 14) are the complexity adjusted values also termed as technical complexity factor (TCF) based on the responses as given in Table 2. The constant values in the equation are determined empirically. This reflects the effect of increasing or decreasing the FP total to that of the UFP.

\[
\text{FP} = 892 \times (0.65 + 0.01 \times 56) = 1079
\]

COCOMO Model Analysis
Generally a project is considered to be developed in one of the following three modes:

- **Organic mode:** The project is developed in a familiar, stable environment and the product is similar to previously developed products. The product is relatively small and requires little innovation.
- **Semi-detached mode:** The project’s characteristics are intermediate, i.e., between organic and embedded.
- **Embedded mode:** The project is characterized by right, inflexible constraints and interface requirements. An embedded mode project will require a great deal of innovation.

The financial accounting system is taken to be in the semi-detached mode. Table 3 shows the basic COCOMO models of effort estimation for different types of projects.
Table 3: COCOMO Models of Effort Estimation

<table>
<thead>
<tr>
<th>Development Mode</th>
<th>Effort = EAF \times (KDSI)^{1.05}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organic</td>
<td>Effort = EAF \times 3.2 \times (KDSI)^{1.05}</td>
</tr>
<tr>
<td>Semi-detached</td>
<td>Effort = EAF \times 3.0 \times (KDSI)^{1.12}</td>
</tr>
<tr>
<td>Embedded</td>
<td>Effort = EAF \times 2.8 \times (KDSI)^{1.20}</td>
</tr>
</tbody>
</table>

where

- Effort is in units of person-months
- EAF is the effort adjustment factor derived from the cost drivers.
- There are 15 cost factors as given below:

### Personal Factors
- ACAP: Analyst Capability
- AEXP: Applications Experience
- PCAP: Programmer Capability
- LEXP: Programming Language Experience
- VEXP: Virtual Machine Experience

### Platform Factors
- TIME: Execution Time Constraint
- STOR: Main Storage Constraint
- TURN: Computer Turnaround Time
- VIRT: Virtual Machine Volatility

### Product Factors
- RELY: Required Software Reliability
- DATA: Database Size
- CPLX: Software Product Complexity

### Project Factors
- TOOL: Use of Software Tools
- MODP: Modern Programming Practices
- SCED: Required Development Schedule

The EPFL in the effort equation is simply the product of the effort multipliers corresponding to each of the cost drivers for a project. For example, if a project is rated very high for complexity (effort multiplier of 1.34), and low for language and tools experience (effort multiplier of 1.00), and all the other cost drivers are rated to be nominal (effort multiplier of 1.00), the EAF is then product of 1.34 and 1.09.

For the financial accounting system, EAF has been found by multiplying the factors (as given in Table 4) to be 2.37.

Table 4: Cost Drivers

<table>
<thead>
<tr>
<th>Cost Drivers</th>
<th>Very High</th>
<th>Low</th>
<th>Low</th>
<th>Nominal</th>
<th>High</th>
<th>Very High</th>
<th>Extra High</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACAP</td>
<td>1.19</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AEXP</td>
<td>1.13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPLX</td>
<td></td>
<td></td>
<td></td>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DaTA</td>
<td></td>
<td></td>
<td></td>
<td>1.08</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LEXP</td>
<td>1.14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MODP</td>
<td></td>
<td></td>
<td></td>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCAP</td>
<td></td>
<td></td>
<td>1.17</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RELY</td>
<td></td>
<td></td>
<td></td>
<td>1.15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATA</td>
<td></td>
<td></td>
<td></td>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPLX</td>
<td></td>
<td></td>
<td></td>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
<td>1.11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOOL</td>
<td></td>
<td></td>
<td>1.10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TURN</td>
<td></td>
<td></td>
<td></td>
<td>0.87</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VEXP</td>
<td></td>
<td></td>
<td></td>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VIRT</td>
<td></td>
<td></td>
<td></td>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Effort = 2.37 \times 3^2 \times 2^1.12 = 175 person-months

Duration = 2.5 \times 175 = 437.5 = 16 months

Number of people = Effort/Duration = 11

REFERENCES

M P Gupta is Associate Professor at the Department of Management Studies, Indian Institute of Technology, Delhi. His research interests are in the areas of IS/IT planning, e-business, and e-governance. He has authored an acclaimed book titled Government Online and edited two others titled Towards e-Government and Promise of e-Governance, published by Tata McGraw Hill, 2004. His research papers have appeared in national and international journals/conference proceedings. He was the recipient of the prestigious Humanities and Social Sciences (HSS) fellowship of Shastri Indo Canadian Institute, Calgary (Canada) in the year 1995-96 and a Visiting Fellow at the University of Manitoba. He supervised the e-governance portal Gram Prabhat which won the IBM Great Mind Challenge Award for the year 2003. He has coordinated several national level seminars and is a member of the International Programme Committee of the International Conference on Business Information Systems.

Deepak Bhatia holds an MBA in Systems Management from Indian Institute of Technology, Delhi. He has worked for leading companies in various sectors like automobiles, construction equipment, pharmaceuticals, and consulting over the past 12 years. He participated in critical projects like migration of legacy systems to enable ERP, leveraging IT for new drug discovery, and project management for accelerated new drug approvals with FDA. Currently, he is associated with McKinsey & Co as the IT Project Manager. He recently played a key role in shoring Global IT support services for his organization from existing locations in Europe, North America, and Australia to India and Eastern Europe.

e-mail: Deepak_Bhatia@mckinsey.com

There are two barriers that often prevent communication between the young and their elders. The first is middle-aged forgetfulness of the fact that they themselves are no longer young. The second is youthful ignorance of the fact that the middle aged are still alive.

Jessamyn West