"All engineering is characterised by the engineer's dissatisfaction with the achievement of just a solution. Engineering seeks the best solution in established terms, within recognized limitations and making compromise required by working in the real world."

E. Yourdon and L. Constantine
Structured Design (YoCo75).

*A modified version of this chapter was published in Proceedings of Annual Convention of Society for Information Science, (ToSr84).
3.1 Introduction

The history of human civilization is marked by ages such as stone, bronze, medieval and industrial. The computer revolution has been responsible for the 'age of information'. The impact of this age was felt in the 1950s when the information processing devices - the computers were marketed and used in academic and business organizations. Although, now the information can be processed at a reduced cost, total expenditures for data processing are increasing because of meteoric demand for software products (Dolo76).

3.2 An Example - Airline Software

A generalised view of a software product is shown in Fig. 3.1. What is a software product and what are its functions will be evident from the description of the classical example of Airline Software.

The need for a software product for reservation was recognized by American Airlines in 1954 because the company was finding it difficult to maintain accurate and timely records of passengers. It took ten years and more then $30 million to develop this software product for the first time. The project was named SABER (Semi-Automatic Business Environment Research). It is now completely redesigned, incorporating new technology and
additional information services. Such product (Jenk69) is now commonly used by all airlines.

The software works as follows. A passenger inquires about a flight. The agent keys in the coded flight number and proposed data of travel on a terminal. Within 2-3 seconds the agent receives information regarding seat availability and answers to client questions such as total plane capacity, number of seats in first class, meals served and plane connections. Once the client reaches a decision of flight and date, this information is keyed along with the client's name, phone number, number of seats requested and passenger names. Special service needs such as salt free diet, wheel chair or assistance in boarding, car rental and escort for children can be added to the client's records. If there are no errors in the input, reservation is confirmed and transaction completed.

In addition to making reservation and keeping passenger records, the software has control functions like informing each travel agency in case the flight is cancelled or there are alterations in flight. Also, the information is used internally to provide estimates of meals, passenger lists, flight control, loading information and funds.

3.3 Historical Background

The problems of software development were first recognized in mid 1960s when system analysts and programmers failed to deliver complex software systems (products). In fact, the term SE was coined in 1968/1969 as a theme for NATO
Conference (NaRa76) held at Geneva and Italy in a provocative sense because the software developed at that time was costly, late and unreliable. So it was proposed that engineering type discipline should be applied for producing software. Zelkowitz (Zelk79) defines SE as follows:

"Software Engineering is concerned with the production of correct computer programs that are also efficient, modifiable, maintainable and reliable. These programs must be implementable within reasonable time periods and at acceptable costs. As such it incorporates many ideas from engineering, management, and computer science."

3.4 Software Life Cycle

Life cycle concept has been derived from engineering. No two software products are alike because every software has unique input-output. A user can get many useful reports with the help of information package within few seconds. But at the back lies months or years of efforts and investment. This is because the implementation of the system requires a set of activities performed as a project, like constructing a bridge or building (GuTo84). The overview of these activities called the life cycle is given in this section. The stages are discussed in separate sections.

The main activities of development of software are shown in Fig. 3.2. This does not show the looping or recycling process. Fig. 3.4 shows how to identify logical decisions leading to the development process.

The need of a new software initiates a feasibility
study. The proposed software is tested for feasibility, if unsatisfactory, it is examined for further study. If it is totally unfeasible, effort is dropped. However, if another design of the software with revised objectives and constraints is feasible the process goes until the proposed study is accepted as feasible (Huss81).

Next, the stages of software requirements, specification, design, coding and testing are executed sequentially along with parallel activities on organizational changes. If the testing is not satisfactory the process will recycle. Multistage looping must be avoided (Boeh83) for successful project management. Recycling continues till the test results are satisfactory. After conversion new software is operational.

When the software is operational it is periodically evaluated for efficiency and effectiveness. If evaluation is satisfactory operation continues, if unsatisfactory then it calls for maintenance or redevelopment. The development recycle may start at any point depending on the nature of the problem. The recycling may start into a new feasibility study if the situation has changed radically and another feasibility is required (Fig. 3.3).

*In this chapter we have treated feasibility study as a separate phase (0th phase), subsequently it is merged with requirements analysis.*
3.4.0 **Feasibility Study**

This is basically a fact-finding activity which provides the information necessary for the decision whether or not to implement a new software. The study also provides the basic data necessary for design and evaluation, in case the project is green signalled. There are four phases in a feasibility study - (1) Organization for a feasibility study, (2) search for solution, (3) feasibility analysis and (4) decision on implementation. Fig. 3.5 shows the steps in each of these phase and their inter-relationship.

Proposed solutions are checked for economical, financial, organizational and technological feasibility. The shortage of funds is the most frequent reason for rejecting a system change. Economic nonfeasiblity - costs exceeding benefits -is next. Non-economic reasons include time constraints, the lack of necessary organizational environment or the technological inability to achieve objectives (KiSc78).

Fig. 3.5 identifies three important decision points. Two are taken by the feasibility team (12), (22), where recommendations either to continue or terminate the feasibility study are made. The third is management's (28). This decision is crucial. If management errs and decides to discontinue the study when the project should have been implemented, the organization has lost benifits that could have been achieved from new software. An implementation when the project should have been discontinued will result in loss of resources and unnecessary organizational disrupts. Feasibility studies are costly and time-consuming but they are significant. If properly conducted
they reduce the risk of making wrong decision.

3.4.1 Requirements Analysis

Determining users' requirements follows the feasibility study. The objectives stated in general terms in feasibility study are restated in details to produce product requirements. This phase is quite significant because most of the deficiencies in software can be avoided by properly defining the required contents of the system (Zelk79).

Fig. 3.6 shows the steps in this crucial phase of development. In order to analyse systems requirements a series of sessions are held so that top management as well as operating personnel can express their informational needs as users (3), (4), (6). Software engineers participate in these sessions but primarily as coordinators and documentors. A final session (8) resolves conflicts between users' groups.

Study Organizational Plan (SOP)

This was developed by IBM and is a systematic method of collecting information and designing a new system partcularly for breakthrough type of software. It consists of three phases - (1) understanding the present system, (2) determining the systems requirements and (3) designing the new system. SOP involves the completion of a number of forms - the Resource Usage Sheet showing the resources (personnel, equipments and materials) consumed by each organizational units; the Activity Sheet identifying the inputs, outputs and file usage of each activity; the Operation Sheet listing the volumes and lapse
times of each operations; the File Sheet specifying each files characteristics; the Message Sheet identifying recorded and unrecorded communications entering or leaving an operation; the I/O Sheet showing the input-output specifications; the Required Operation Sheet recording details of operational elements within flowchart diagrams and Resource sheet providing quantity and cost data on each resource used.

3.4.2 **Product Specifications**

The objectives, policies and constraints as they relate to users' requirements must next be specified in operational terms as product specifications (also called definition). This is the information which will be used by designers in the next stage.

Although each project will vary in actual specification needs, following is a general list showing the details required:

**Output**: Contents, format, quantity, availability, response time, frequency and distribution list

**Processing**: Decision rules, accuracy, ratio or absolute range, significance of results, current and future capacity

**Input**: Source, media, procedures and validity checks

**Security**: Input, organization, maintenance, decision rules, output, nature of access, list of those allowed access

**Backup**: Items needing backup, nature of backup procedures and maintenance.
Once users' requirements have been explained and documented as product specifications, the proposal must be checked for factual and statistical accuracy and the procedures analysed to make sure they accomplish stated objectives. Then the users and software engineers must decide whether the specifications are adequate (Fig. 3.6). If any information is missing the job recycles. Management may also decide to terminate the project if unsatisfied. This may be the result of disappointment with the product specification or could be due to reevaluation of project in the light of changed environmental conditions such as unavailability of expected resources changes in assumptions made during feasibility study, changes in priorities or appointment of new personnel in management.

3.4.3 Product Design

In this stage from the detailed systems specifications, a blue print of the software is prepared. The software engineers prepare the details of data base, physical preparations, procedures and program solutions. The systems specifications are now converted into specific operational terms designating input, output, the flow of information, standards for personnel, forms to be used and equipment performance required. Computer programming specifications and system test specifications must also be determined. Following are the main activities of this phase:

- management of design phase
- determine operational/hardware specifications
• select and develop model/algorithm
• determine specifications for documentation standard, testing, organizational, programming, quality control, output, input, files, forms and procedures
• revise project schedule
• management approval

Formal Design Techniques

There are two basic strategies that can be used in defining product design: bottom-up approach and top-down approach. Some techniques of product design are integrated with computer program development i.e. computer will generate the necessary sets of programs for the implementation of the software product given the product specifications as input. PSL/PSA (Program Statement Language/Analyser), HIPO (Hierachy plus Input Process Output), SODA (System Optimization and Design Algorithm), ISDOS (Information System Design and Optimization System), HOS (Highe Order Software) are such methodologies.

3.4.4 Program Coding

Next to the product design comes coding phase. If the design has been prepared elaborately the programmers need only translate each design specification into computer programs. Depending on the type of software i.e. systems, control or application a suitable language is selected or developed and programs are coded in that language. Modern high-level languages, top-down design, structured programming and interactive facility simplify the task. In one study, Boehm (Boeh83) found that in a software project 64% of all errors
occurred in design but only 36% in coding (GuTo84).

Following are the main activities of this phase:

- write main programs
- write programs for storing data
- write validity programs
- write maintenance programs
- installation of hardware
- file preparation
- organizational changes.

The use of Modern Programming Practices (MPP) including Top Down Structural Programming and other practices such as information hiding, help to get a good deal more visibility into the coding process, contribute greatly to getting errors out early, produce much more understandable code, and make many other jobs like integration and testing.

3.4.5 Testing

This phase is meant for quality assurance. Testing involves comparison of desired performance, as stated in users requirements, with actual performance. If any deviations are detected they should be eliminated. Reliability Testing Standards state the allowable occurrence of error in processing and Performance Testing Standards include specification of response time, throughput and staff equipment efficiency, Load Specifications state the factor of safety needed to peak periods, emergencies and future expansions.

The preparation of test data is also significant. The
project team has to decide how test data is available for testing, but in projects performing functions never done before, test data has to be created. What data to use and how much to use is a problem of sampling. A well designed sample produces results as significant as results from the data collected from the population as a whole.

Testing a new product must be done at four levels—component, function, subsystem and system testing. Testing is a labour-intensive activity and takes a great share of the total efforts required for developing a software product. The testing process ends when test results at the system level are satisfactory. At the conclusion of favourable testing, the test results must be presented to management for approval.

Responsibility of testing lies with users and software engineers. Users can best formulate the conditions for outside testing i.e. examining product response to varying conditions of input, including instability and overloading. Inside testing i.e. examining the structure for completeness, consistency, reliability, fall to software engineers.

3.4.6 Documentation

In the life cycle of a new product, this phase plays an important role. Documentation is a written description of choices and decisions during the development process. Although documentation is done throughout the process yet this separate phase is necessary for a particular type of documentation from users point of view.
There are two types of documentation: (1) Developmental—this is a description of the software itself i.e. objectives, characteristics, decision rules etc. This is required for operation and use of the new product. (2) Project control documentation—this concerns project development organization i.e. personnel, time, materials, money etc. This is required for auditing and evaluation of project. Proper documentation helps an organization in the following ways—

1. It is a record of all commitments and expectations.
2. It helps in introducing and training newcomers to the software.
3. It provides instructions required for altering the software.
4. It removes monopoly of few individuals who resign or are assigned other duties.
5. It facilitates routine evaluation, auditing and control.

The fact that documentation is not a popular activity means that project team must carefully spell out documentation standards and insist upon strict compliance. System documentation is normally divided into four manuals: system manual, programmers manual, operators manual and users manual. Apart from these program listings, hard copy printouts and micro-fiches etc. must also be documented properly.

The project team must prescribe what material is to be contained in each manual and establish the format for recording information. Abstract, details and summary should appear in each manual. The team must also specify the timing for completion of the document because it loses its value if not available when
Finally, the team should establish control standards for documentation. Librarians responsible for the manuals should be assigned and given authority to control access and a physical location for the manuals must be chosen. Persons authorized to use and/or modify the documentation must be identified and procedures for revision and validity testing must be set.

3.4.7 Operation and Maintenance

Once a product is developed and implemental operations (production) begin; a prime responsibility of EDP department is to optimize efficiency of operations i.e. ratio of output to input. Optimization of operation means either maximizing output within a given budget or minimizing input for the production of a required output. How computer hardware, software, database and EDP personnel are deployed will determine results. The department may change equipment or reorganize existing equipment in a new configuration to improve efficiency, organization logic may be revised, software rewritten, the database restructured and/or data on storage device reorganized. All of these technical decisions will effect both efficiency and critical factors of performance such as timeliness, accuracy and quality of information, factor by which users measure the effectiveness of the software product.

The monitors must be used to collect data on operations for evaluation process. Hardware monitors can collect data on the utilization of the CPU and peripherals as well as
data that can be analyzed to identify bottlenecks, down time, and saturation conditions of channels, storage devices and other peripherals. Software monitors can provide management with information on how the system resources have been used, by whom, and for what application. Monitor programs embedded in software may keep track of application program in demand, language favoured, as well as software features and routines used. Monitors can assist in the collection of quantitative data used to evaluate effectiveness but qualitative evaluation requires a survey of user attitudes.

The process of evaluation and monitoring is shown in Fig. 3.7. Changes in operations to improve efficiency and effectiveness are called fine-tuning of the product. With constant technological development in hardware and software the importance of regular evaluation cannot be underrated.

When a predetermined level of error-tolerance is exceeded system modification is needed. Modification may be minor called maintenance or major requiring redevelopment. This is shown in Fig. 3.8.

There is a good correlation between high standard in the original development process and low maintenance. If validation is performed continuously, it will reduce maintenance. Rigid standards of documentation would also reduce time and efforts required to make changes. Many software products go through many versions (release) during their lifetime. It is widely acknowledged that the cost of maintenance far exceed the development costs.
3.5 Summary

A software product is an extremely useful and vital entity. The main thrust of this chapter is on the analysis of various steps in the development and maintenance of this product. The process is quite complex and is accomplished in several stages called life cycle.

Once the need for a new product is perceived a feasibility study is conducted to ascertain whether or not a product can be designed to produce the output according to requirements. Product specifications are then prepared. Next comes the design and computer programming. A thorough testing and validation is then performed. Then the operation begins which is periodically evaluated for actual performance with objectives. This may lead to maintenance or redevelopment.

For successful project management each activity needs to be scheduled, allocated resources, and completed within time constraints. The entire set of activities from feasibility study phase through operation and maintenance phase requires planning and coordination. A successful project depends on a close working relationship between project management and software personnel without any communication gap. In developing large software products, project control techniques such as Gantt Chart, CPM/PERT and GERT are required. These techniques utilize computer programs to calculate critical path i.e. sequence of activities that can be delayed without prolonging the entire project.

The life cycle propounded in this chapter is
applicable to all types of software products large, medium or small. In case of actual product development, in different environments, several activities are combined, added or omitted. Also some other names are used for same activities.
FIG. 3.1 - GENERALISED VIEW OF A SOFTWARE PRODUCT

FIG. 3.2 - PHASES IN THE LIFE CYCLE OF A SOFTWARE PRODUCT

FIG. 3.3 - LIFE CYCLE
FIG. 3.4 - FLOWCHART OF THE LIFE CYCLE
FIG. 3.5 - FLOWCHART FOR FEASIBILITY STUDY
FIG. 3.6 - USERS REQUIREMENTS AND PRODUCT SPECS
FIG. 3.7 - EFFICIENCY - EFFECTIVENESS OF A PRODUCT
FIG. 3.8 - MAINTENANCE AND REDEVELOPMENT