Chapter 2

Software Engineering Project Management

2.1 Introduction

In this chapter, project management, software engineering problems, software management techniques and functions, and other related topics are discussed. In addition, a comparison between some of the available tools is also highlighted.

- *Management* in 1980 is defined by Koontz as the creation and the maintenance of an internal environment in an enterprise where individuals working together in-groups can perform effectively and efficiently toward the achievement of group goals [KOO72] [CAR91]. In other words, management is the art of getting work done through other people [CAR91] [EAR89] [AVR94].

- *Project* is a one-time effort having well defined objectives and processes within a specific time frame [CAR91] [AVR94].

- *Software Engineering Project Management* is for the management of projects, which produce or are concerned with software. To ensure the success of any project, a manager must be able to perform completely the planning, organizing, staffing, directing, motivating, leading and controlling functions. More details on these fundamentals will be presented in this chapter.
2.2 Problems Affecting Development of Software Projects

Developing software has become the world's main concern. It has been growing very fast, as a result it is inherent with a lot of problems, difficulties, and complexities. Building and maintaining software projects are the most expensive projects. Therefore, a delay in delivery can be very costly and undetected problems may cause a loss of performance and frustrate users [RAM84]. For example, in the 1960s and early 1970s the failure of most software projects were due to problems in software management [IAN92] [IAN95] [RED90]. Most problems in software projects are due to inadequate management rather than technical difficulties [RED90]. Norm Brown stated that the absence of a comprehensive risk management program is a lead indicator of initial project failure [NOR96]. The study conducted on requirement review that people, process, and technology which are software quality aspects, are recommended to solve the review problems. However, there are many project managers who have failed to ensure product quality because they fail to provide standards for their staff. The absence of standard causes an increase in the probability of defects and makes the work difficult to inspect and verify. Moreover, there are no specific procedures for quality assurance [RED90]. Siew and Yaacob reported that the most critical problem in a requirement review process is the inability to detect deficiencies in the requirement specification until at a later review stage of software development [SIE97]. Gilb stated [TOM97] that the problems in requirements are lack of clarity, lack of known quality-controlled defect level, lack of testability, ambiguity, inconsistency, lack of variables measurability, and lack of priorities information.
Richard reported several reasons for the estimation process difficulties such as: i) projects often must satisfy conflicting goals, ii) estimates are required before the product is well defined, and iii) complication arises due to changes in the way software is being built [RIC96].

Results of large-scale software projects indicated the following: i) there is a direct need for strategies, practices, and techniques to substantially accelerate the pace of progress improvement, ii) the large-scale software project problems are due to poor software management and the reason behind this is the poor understanding of the essential management, practices, and techniques (this problem is known as senior management problem) this misunderstanding leads to an unwillingness to support the need of infrastructure, and progress improvement. iii) experiences with DOD (Department Of Defense) has shown that large-scale software projects have a high inherent complexity, and iv) to build successful large-scale software, detailed planning must be specified and based on accurate estimation of resources and time for all activities [NOR96]. Schach noted that, spending too much time in testing is a waste of time and money and may cause delay. However, if too little testing is done, faults and defects may be inherent in the product. Therefore, determining when each phase is adequately tested, is a common problem in software development. The problems which affect software production such as: complexity, conformity, changeability, invisibility, is no Silver Bullet (a way of rapidly obtaining an order of magnitude to increase), and moving target problem which due to the client who keeps changing their requirements [STE97]. The problems that arise when requirements for software system are poorly understood will never totally disappear because of pushing the technology envelope and building product platforms [MAR96]
Michael stated that producing relevant schedules for software development is more problematic than doing the development itself [MIC96]. In addition, investigation for finding the reasons behind poor project control, identified the following problems: i) customer and management changes, ii) technical complexities iii) unrealistic project plans iv) staffing problems v) insufficient front-end planning; and vi) inability to detect problems early. As an out come of their study, they presented problems which caused slips and budget overruns such as i) difficulty or misunderstanding of requirements ii) rushing into project kick-off iii) inability to measure true project performance iv) poor communication with upper management, customer or project leadership v) poor ability to manage conflict; and vi) difficulties in assessing risks [HAN86]. Report highlighted the major problems caused cost overrun of more than one billion dollars in software projects in the U.S. Department of Defense, are represented in lack of cost visibility and control, schedule delays, and lack of capability at the promised time. Abdel-Hamid, Sengupta, and Ronan stated that schedule and resources estimation, accurate and timely status are absolutely vital to effective project control. They also addressed that the difficulty in estimation project is due to poor utility of the available models in estimation [TAR93]. Tom stated that software engineering and system engineering projects are regularly out of control due to the used standards to achieve a statistically stable system and failure to provide motivation to improve [TOM97*].

After years of pursuing fragments of a solution to the challenges of software engineering, including the development of methods, tools, and the like, the software engineering community has come to realize that over emphasizing on any one of these areas to the exclusion of others is incorrect. The community recognizes that well-defined and well-
supported processes are the critical issues for producing high quality software system on time and within budget [BRI95].

Booch observed that the complexity in software projects is derived from four elements: problems with project size, difficulty of managing software development process, flexibility possible through the software, and the problems with characterizing the behavior of the direct systems [GRA94].

Studies reported that the design reviews and the code inspection have a measurable impact on product quality. The study also revealed that general industries in Europe have a high degree of confusion about productivity and quality issue, and process improvement [REM95]. Overheads communication, different experience, and stress by delivery date are factors affecting the behaviors of the developers and thereby affecting the whole project [OSA96].

A study done by Redmill found that many problems occurred due to project managers, who lack experience of working where quality culture predominates and without adequate training in management process and in total quality management. Redmill also reported that the solution for many problems is to train and commit both the developers and the user to total quality management (TQM) and the principal reasons for project management shortcomings should be reviewed [RED90].

Moreover, system interfaces are often beyond a project manager’s control [NOR96].
2.3 Software Project Management Functions And Activities

Software engineering project management focuses on five main activities, that is planning, organizing, staffing, controlling, and directing. Among these functions there is an interrelationship such as planning for organizing activity, organizing for planning the undertaking, controlling the planning function, organizing for staffing the organization, staffing for the planning organization, [CAR91] [RIC87] [KOO72] [RUE83] and directing each of them. Generally, software project management is divided into major phases: planning a project, managing and controlling it. These abilities require knowledge such as knowing common management pitfalls, and skills such as the ability to recognize these pitfalls and formulate new project plan [CHM97] [DER97]. The following is an overview of the software project management activities:

2.3.1 Planning

The planning phase is the first step in software project, and the basis of management procedure. Planning is deciding in advance: (1) what objectives are to be achieved which can be done by providing a clear picture and sufficient description, (2) how to work on it which can be implemented by selecting the suitable methods and following the latest techniques, (3) when exactly the project must start and complete, and (4) determine who is/are going to perform it. Additionally, the planning phase provides the user with the detailed knowledge of the schedule, organization, and resources allocation.

Behind every successful project is a good plan and to ensure its success, the test plan must be frequently overlooked. Therefore, testing must be planned like every other activity in software development and must be in every activity or task. Project planning
must include resources for testing, and the test that is to be done during each activity in the detailed schedule of the project must be indicated[STE97]. Moreover, quality plan must be ensured and measured. The following are the fundamental keys for measuring a project plan: (1) is it complete, reviewed and specified, (2) is it accessible (is it in its proper format, the original schedule and the subsequent versions can be found quickly), (3) is it clear, (4) is it able to answer these question: what will be done, when, by whom, and at what costs, and (5) is it accurate [WAT95].

2.3.1.1 Project Estimation

Project estimation is one of the project manager’s responsibility. Software cost estimation, like planning and testing, is a continuing activity which starts at spelling out the user’s requirements and continues throughout the lifetime of a project [IAN92]. The principle components of project costs are: hardware and software costs, overheads such as rent, utility, travel and training costs, and human resources costs such as salaries of development teams, managers and support personnel involved in the project. Hardware, travelling, and training are controllable budget but effort is the critical, due to difficulty of estimating the task and estimating how much task each developer can solve [CAR91] [IAN 92] [STE97]. Forecasting how many engineers will be needed and what is the project’s budget are very difficult factors in software engineering. Estimating the start and finish dates of the project is considered one of the important issues in planning a project. However, still most of the projects go beyond the estimated dates. The development organization will lose money due to overestimation or underestimation of the actual cost. Therefore, proper estimation is needed to ensure the success of a project.
must include resources for testing, and the test that is to be done during each activity in
the detailed schedule of the project must be indicated[STE97]. Moreover, quality plan
must be ensured and measured. The following are the fundamental keys for measuring a
project plan: (1) is it complete, reviewed and specified, (2) is it accessible (is it in its
proper format, the original schedule and the subsequent versions can be found quickly),
(3) is it clear, (4) is it able to answer these question: what will be done, when, by whom,
and at what costs, and (5) is it accurate [WAT95].

2.3.1.1 Project Estimation

Project estimation is one of the project manager’s responsibility. Software cost
estimation, like planning and testing, is a continuing activity which starts at spelling out
the user's requirements and continues throughout the lifetime of a project [IAN92].
The principle components of project costs are: hardware and software costs, overheads
such as rent, utility, travel and training costs, and human resources costs such as salaries
of development teams, managers and support personnel involved in the project.
Hardware, travelling, and training are controllable budget but effort is the critical, due to
difficulty of estimating the task and estimating how much task each developer can solve
[CAR91] [IAN 92] [STE97]. Forecasting how many engineers will be needed and what is
the project’s budget are very difficult factors in software engineering. Estimating the start
and finish dates of the project is considered one of the important issues in planning a
project. However, still most of the projects go beyond the estimated dates. The
development organization will lose money due to overestimation or underestimation of
the actual cost. Therefore, proper estimation is needed to ensure the success of a project.
There are several techniques and models used for cost estimation such as:

II. **Decomposition technique.** As mentioned above it is very difficult to estimate software project (estimating effort and cost) in one piece. Therefore, decompose the problem into smaller sets and estimate each set individually and then add up for the whole project [ROG97].

III. **Software size techniques** Estimating the project can be considered good, if the size of work estimation is completed. The following are software size techniques which are well described in [AVR94] [STE97] [IAN92] [IAN95] [BAR84]:

a- **Algorithmic models.** It provides one or more algorithms to produce software cost estimation as a function of a number of variables, which are considered the major cost drivers.

b- **Expert Judgment.** This approach involves consulting one or more experts on software development techniques.

c- **Estimation by analogy.** This method is applicable when one or more completed projects are similar in the application domain of the new project. The cost of a new project is estimated by analogy with these completed projects.

d- **Parkinson’s law.** Parkinson’s principle is work expands to fill the available time. That means cost estimation depends on the resources available and not by objective assessment.

e- **Price to win.** Here the software cost estimation is based on the customer budget.
f- **Top down technique.** The total cost is split up among the various project components. In this technique cost estimation is based on logical function rather than the components of implementation function.

g- **Bottom up technique.** In this technique, each component of the project is estimated separately and then all these component costs are added to produce the project cost estimation. Boehm argued that none of these techniques are better or worse than any other technique, and concluded that ‘Parkinson’s low’ and ‘Price to win’ methods are unacceptable because they do not produce satisfactory cost estimation. Several cost estimation techniques should be used in parallel and their results compared in large software projects.

In the late 1970’s, several software cost estimation models were developed and some of them are:

a- **Putnam SLIM models.** This model is based on Putnam’s analysis of the software life cycle and Rayleigh-curve distribution of project skill versus time. This model is useful in estimating quantities as manpower distribution, the main milestone schedules, and cost documentation. In addition, it provides the relationship between software development effort and time for estimate project size.

b- **RCA PRICE S Model.** It provides monthly project cost schedule, expected progress forecasts, and calculate development skills time distribution.
c- **COCOMO (Constructive Cost Model).** COCOMO model exists in a hierarchy of three software estimation models. These models are basic, intermediate, and advanced COCOMO models [Rog97] [STE97] [IAN92] [IAN95] [BAR84].

d- **Lines Of Code technique (LOC).** It is focused on measuring the size of the project by the number of lines of code [STE97].

In the early 1980's, other metrics such as software science, were developed and it is concerned with product size to overcome the disadvantage of the line of code technique, by determining the number of operands and operators in software product. However, the problem in this technique is that the number of operands and operators can be determined only after the product has been completed. Therefore, the concern after that become on developing techniques which focuses on estimate software product size which can be determined early in the software development process [STE97].

e- **File, Flows, and Process (FFP).** This technique attempts to measure the product size early in the software development process [STE97].

f- **Function Point (FP).** It is similar to the FFP concept and is based on the number of inputs and output items [CAR91] [STE97].

g- **Software Equation model.** The software equation model is a multivariable model based on specific distribution of effort over the software life cycle process [ROG97].

There are several techniques for estimating the duration of project activities and some of them are well described in [AVR94]:

---

*Software Project Management*
a- **Stochastic approach.** This approach addresses the probabilistic activities in a project by estimating the expected duration of each activity and its corresponding variance.

b- **Deterministic approach.** In this approach the duration is estimated based on past data which is similar to the activity.

c- **Modular technique.** It is based on breaking down the activity into small modules and estimating them individually.

d- **Benchmark job technique.** This technique is useful for projects containing many repetitions of some standard activities.

e- **Parametric technique.** This technique is based on identifying the relationship between dependent and independent variables, and representing them graphically to estimate time performance.

### 2.3.1.2 Project Schedule

Scheduling software project is considered one of the most difficult tasks. It is a graphic presentation of all related information about the project, and allows the project manager to track the project and help him to make decisions. It is virtually impossible to effectively control the project without a project schedule. Project schedule is responsible for directing the project team by breaking down the work into tasks, determine the duration for each task, and defining the interrelationship among these tasks.

The following are techniques and models used in project scheduling:

a- **Work Breakdown Structure (WBS)**
WBS is a checklist of the activities or tasks to be accomplished to meet the project goal. The concept of this technique is to break down the work into smaller tasks. Each task can in turn be broken down further. This technique is very useful for the project manager to become familiar with the scope of the project, identifies work tasks, needed resources, and cost estimation. It also helps to monitor the project’s progress. WBS is well defined in [CAR91] [BAR84] [ROB80].

b- Milestone Chart

Usually after defining the objectives of the project, the work will be divided into fundamental phases or activities. These phases are called milestones and a milestone can be the completion of system integration, product delivery, or completing a chapter of the user’s manual. The milestone chart is a management tool used to control and monitor milestone progress by listing the milestone such as start and finish dates, and their status [RIC88]. An example of a milestone chart is shown in Figure 2.1.

![Milestone chart](image)

Figure 2.1: Milestone chart

c- Hammock Activities. A Hammock activity is an activity replacing a group of activities having the same start and end points. Thus the duration of a hammock activity is equal to the longest sequence of activities [AVR94].
d- Gantt, Timeline, or Bar Chart.

In the early 1900's the first tool for software scheduling was developed by Gantt [GUP85]. This tool was developed to help in scheduling, budgeting, and monitoring the progress for each task. Gantt chart is a bar chart, where each activity or task is represented by a bar. These bars are drawn based on their time line, which is the length of the activity or task planned time. Gantt chart is easy to use and understand, therefore it was developed and improved. Now it has the ability to show the relationship, milestone, and the overlap among activities. Therefore, it is used to estimate the cost for project resources. An example of the use of Gantt chart is shown in Figure 2.2 [CAR91] [ROG97] [KEN85].

Figure 2.2: An example of a Gantt chart

e- Full Wall schedule

It is a large wall divided into vertical and horizontal lines. The spaces between each pair in these vertical lines represents the work, and the horizontal lines represent the
members of a project team. The project manager determines the milestone, breaks down the work into tasks and identifies the resources of these tasks. Each task is then written on two index cards, one labeled start and the other labeled finish. Then each developer is given the appropriate cards to put them on the wall schedule under a date they have selected. The arrangement of the cards will be continually updated until the entire team is satisfied. Finally, the full wall schedule is recorded and documented by the project manager, and distributed to the team members. The advantage of this technique is that, any conflicts can be identified, discussed and solved early, but this technique cannot be effective in a large project due to the difficulties in getting a large project team together for a long time, and the interrelationship among the activities are not clear [KEN85].

f- Precedence Network CPM and PERT

(1) In the late 1950's, several network technique were developed, but the most popular and best known are the Critical Path Method (CPM) and the Program Evaluation and Review Technique (PERT). PERT was developed in 1958 as boxes or circles and arrows, each box represent an activity, and the dependencies between these activities are represented by the arrows as shown in Figure 2.3. The CPM is also implemented in a PERT and both of them identifies the critical path. The critical path is the longest sequence of activity in the project, which help the project manager plan and control the project. [CAR91] [AVR94] [KEN85].
2.3.2 Organizing

Organizing in a software development project is required to coordinate the work among the activities in a formal structure when the work cannot be completed by a single person within the scheduled frame time. Project organization provides the procedures and rules to ensure quality, and it helps to distribute the responsibilities between the activities. The following are the project organization's responsibilities:

a- Software requirements must be identified, then divided into groups, and the tasks to be completed individually defined.

b- The funds must be allocated to all groups performing the tasks.
c- Select organization structure for the project, which is considered as one of the most difficult stage in organizing the project.

d- Name and create positions for those groups, which have been identified in more details.

e- Define the authority for each assigned position for self-confidence and to use the judgment for making the decision.

f- Identify the qualification for each position such as years of experience, qualification in specific area, reusability, and error detection.

g- Analyse the company’s capabilities, capacities, and efficiencies, and make final decisions on whether the company supplies or buys hardware and services for the project.

h- Document all information about organizing the project such as positions, and problems that the organization has faced and the reasons for these problems.

i- Identify problems and initiate actions to solve them.

j- Ensure control over all project changes.

There are several types of organizational structure and the following are some of them:

1- **Conventional organization structure.** It is either a line or staff organization type. Line organization has the authority to perform the work that represents the primary mission of the project organization. However, the staff organization is a group of special experts which has the responsibility to perform the activities that help or support the line organization in their work [RIC87].

2- **Functional organization structure.** The project structure is based on software engineering function or a group of similar functions. By passing the project from
function to function, the project will be accomplished. A project manager has the responsibility for completing the project. The main advantage of this structure is that it can handle multiple projects [RIC87] [DEN87] [ROB77] [LIN81].

3- **Project organization structure.** This structure has been established especially for developing large software projects. The project manager in this structure has total control over the project and has the responsibilities to hire, discharge, train, and meet the project goal within the resources of the organization. The disadvantage of this structure is it can be applied only for one project [RIC87] [DEA92] [ROB77] [LIN81].

4- **Matrix organization structure.** This type is a compromise between the functional organization and project organization. A project manager has the responsibility for completing the project, but usually does not have the authority to hire, discharge, or train [RIC87] [ROB77] [LIN81].

5- **Application organization structure.** This type of organizational structure is suitable for a commercial software group. This structure can handle multiple projects too.

**2.3.3 Staffing**

Staffing involves people, and deals with hiring personnel for the positions that are identified by the organizational structure.

Successful software projects identify the strength and the weakness of the team members and require a number of very different technical skills such as analysis, system design, program design, coding, integration, and testing [WAT97]. The main functions and activities of the staffing phase are as follows:
1- *Fill organizational structure:* The project manager is responsible for assigning the right person to the right position (this position will be identified by the organizational structure). The manager must also ensure that he chooses capable and productive managers, software engineers, analysts, and programmers. In staffing any software development project the following factors should be considered to fill in an organizational position: (a) qualification, (b) work experience, (c) intelligence, (d) the capability to learn, to take difficult assignments, (e) resources availability, and (f) training.

2- *Orientation:* The manager is required to introduce the new employee to the company and the company to the employee [RIC87].

3- *Training:* Training means teaching skills, and it is the project manager’s responsibility to ensure that the person will meet the requirements. For example, the managers should be educated in management sciences and business techniques and they should be trained in management techniques and administration duties. Engineers are educated in science, physics, mathematics, programming languages, and software principles and trained on project application [RIC87].

4- *Maintaining a professional attitude:* The project manager has to ensure that their professional knowledge will increase and they have a positive attitude towards the project [WAT97].

5- *Evaluate personnel:* The manager is required to evaluate the personnel’s performance. It will make the staff compete to do their best. Besides that, it identifies for each person for future assignment.
6- **Compensate:** The manager is also responsible directly and some times indirectly in determining the salary scale and benefits of his people.

7- **Terminate assignment:** Besides hiring people, he is also responsible for terminating them.

8- **Staffing document:** The manager is responsible for producing staffing plan, evaluation, salary, promotion, advancement policies, and work schedule.

### 2.3.4 Directing

Directing a software project is leading the people, by clarifying their assignments, guiding them towards improved performance, solving expected and unexpected problems, and monitoring them to work with enthusiasm to meet the project requirements. Therefore, highly educated, wise, and experienced leadership is one of the major issue of a project’s success. The main functions of a directing phase are as follows.

1- **Leadership:** One of the project manager’s responsibility is to provide leadership to his project management team and to ensure that every person in the project is working towards the project goal [RIC87].

2- **Supervision:** The manager is responsible for providing day to day supervision and direction to his people.

3- **Delegation:** Delegation is the passing on of the responsibility for a given task by one person to another [EUG81].

4- **Motivation:** The best work can be obtained from the staff by motivating them. There are several motivational techniques applicable to software projects such as:
a- *Theory W*: The fundamental principles of theory W are (1) make everyone (project managers, developers, customer, users) win by paying careful attention to people’s interests and expectations, using better technology and early error detection techniques, (2) “Plan the Flight and Fly the Plan”. To make everyone win, several types of plans must be established such as operational plans, installation and training plans, and development plans. Each of these plan must have subsidiary plans such as configuration management plans, quality assurance, and test plans, and (3) risk management, which must be managed to analyse, monitor, and control each risk to ensure that everyone is a winner [BAR89].

b- *Theory X*: Theory X takes the negative view of the people. For example, the average person has an inherent dislike of work, most of the people have little ambition, and most of the people need to be controlled, directed, and treated with punishment [RIC87] [WAT97].

c- *Theory Y*: Theory Y is the opposite of theory X, because it takes into account the positive view of the people. This concept is called modern management thinking. In this issue, the people are psychologically motivated to work and the management should help them rather than coerce or control them. In addition, the people are self-controlled and self-directed. These theories cannot be used as a general technique because it is unfair to treat all the people in the same way, Some of them are energetic and hardworking. Besides that, sometime the external control and threat of punishment is not the right solution [RIC87] [WAT97]. Therefore, the professionals are motivated and
to get better results, the management should be flexible and balanced in using these theories and to understand when it is proper to use of them.

**d- Theory Z:** It is a combination of American and Japanese management styles. Some of the theory principles are: (1) focus on the people’s needs, (2) to obtain good performance the people should be motivated, (3) people’s efforts to achieve the goal may have some mistakes and therefore it must be corrected either by solving the mistake or directing the individual to the right path, and (4) providing a mechanism for gradual change [RIC87].

**5- Coordination:** The project manager is responsible for facilitating communication for both within and between his project and others to ensure that people are working toward the project’s goal.

**6- Resolution:** The manager is responsible for solving any conflicts or problems among his team members. Thereupon, the manager should have enough judgement to recognize the best possible approach to solve the problem [RIC87]. He also must get feedback on the problem, the time consumed, who is/are responsible, how it is solved, and how it has affected the project. This information is very useful in managing future projects.

**7- Managing changes:** Once the project requirements are changed the whole project phases will be changed. Thereby, the managers are responsible for accommodating these changes and controlling them.

**8- Documentation:** One of the most important responsibility of a project manager is to document and keep all the information about tasks, authorities, people’s performance,
used techniques, problems and all decisions made, concerning lines of communication, coordination, and conflicting resources.

2.3.5 Controlling

Controlling is measuring the activities performed against plan and correcting any deviation to ensure that the actual operation is going according to plan [CAR91] [RIC87] [TAR93]. For effective project control, two kinds of information should be provided. Firstly, is the project plan, schedule and resources information as an overview of what is happening. Secondly is the accurate and timely status information.

The fundamental functions in the controlling phase are:

1- **Develop standards of performance**: The project manager is responsible for developing the project’s standards of performance. These standards in software engineering are a set of procedure that defines the process, which in turn defines the quality of the product.

2- **Software Quality Assurance (SQA)**: SQA is one of the major control techniques available for the project management. SQA is an all-planned action which necessary in order to be informed about product quality [RIC87].

3- **Monitoring**: The information about the project’s progress is very important to ensure that everything is going according to plan. Therefore, establishing methods, selecting techniques for monitoring the project is one of the project manager’s responsibilities so as to determine the project status at any time it is needed.
4- **Result Measurements**: The project manager is responsible for measuring the results of the project. The results can either concentrate on the process or the project’s product.

5- **Corrective action**: When standards and requirements cannot be met, the project manager must take corrective actions, which may need to change the project plan or the standards.

6- **Documentation**: The project manager is responsible for documenting and publishing reports on the standards, every corrective action taken, the measurement method, and the actual result.

### 2.4 Process Management

Process management consists of defining, analyzing, and evaluating models. These models describe and specify how the work should be done but not when, what, and by whom in a specific project. However, when, what, and by who in a specific project are project management decisions [MAR93].

The fundamental role of software process management is to ensure that the project meets its commitments and this requires planning a schedule. The traditional project management model, such as PERT, CPM, Gantt chart, and GERT are inadequate, especially with large software development. Petri nets based model, such as ITCPN and DesignNet have been applied into software project management. However, they could not solve all the problems [CHU93] [VIC96]. The following are some other software project management models:
1. **BSPN model:** It is an integrated model of PERT and Petri nets. This model is used to overcome the uncertainty planning problem. BSPN can be used as a complexity metrics of a project [CHU93].

2. **Parallel decision making:** The concept of this model is used to support the current market demand which is concerned with the communication technology (networking) [VIC96].

3. **Capability Maturity model (CMM):** CMM provides the guidance on how to control the developing and maintaining software, and to guide software organizations in selecting process improvement strategies by determining the current process maturity and identifying the few issues most critical to software quality and process improvement [ALY96].

### 2.5 Acquisition Management

Acquisition management is a process of preparing and administrating the contract agreement and managing the performance of the developer. Figure 4 illustrates the relationship between software acquisition management, software engineering, and project management [JPH94]. The main activities for software acquisition management are as follows: (1) planning, (2) contracting, (3) budgeting, (4) evaluating performance, and (5) future support System [JPH94].
2.6 Metrics Management

The metrics and the measurement selected for controlling the project from the areas that have the greatest risk of failing to meet the requirements. It is useful for determining if the initial set of requirements or size of the effort has changed. In addition, it helps to estimate the progress in developing the system and to estimate the quality of the product. Most of the available metrics serves as a management indicator but a combination metrics usually provides more information [JPH94].

This issue is very important. However, the available software project management tools are incapable of supporting this aspect. Advises for using metrics in project planning are as follows:

a- Start with a few key metrics for the organization.

b- Motivate people to supply information and use metrics for them to help the project but not used to judge them.

c- When the data collected and the indicators used earlier, corrective action will be easier and less expensive.

d- The data correlation should be from similar projects and significant differences in cost derives should be accounted [JPH94].
2.7 Risk Management

An engineering project is expected to produce a reliable product on time and within budget. However, any project runs the risk of not producing the desired product. Project risks identify potential overspending (resources budget), overrunning (schedule time), and accompanies any human activity (customer, staffing and organization) and can be in requirement problems and natural disaster. All these risks have an impact on software product.

In building and operating large systems, the management must face up to such questions as: What can go wrong and, how can it happen? What range of consequences might there be, when and how could they be averted or mitigated? How can it be measured, reduced, monitored, and managed? [AVR94] Which risks can be tolerated or accepted, how and in which level can each risk affect the project?

Risks can be reduced by quality management which include effective software configuration management [STE94] [GRE97].

Team risk management is a concept for managing projects by developing a shared product vision, focused on results, and using the principles and tools of risk management to cooperatively manage risks. This team has an advantage compared to individual or group risk management such as improved communications and sharing both the customer and the supplier for mitigation. The functions of team risk management are : 1) recognize the need, 2) identify risks and set project priorities, 3) analyse risks, 4) plan for mitigating actions, 5) monitor risk indictor and mitigation plans, 6) control and maintain, and 7) report and document [RON95].
“Risk analysis and control is a topic of management theory” [CAR91]. Risk analysis is an attempt to pin down and quantify the answers to these questions wherever possible. Therefore, the first step is identifying all the known and predictable risks for the project manager to manage them.

There are several standards and techniques for identifying project risks, assessing their impact, monitoring and controlling them.

The effective strategy dealing with risks are (1) **Risk avoidance**: The best strategy for a software team to adopt is the risk avoidance approach. In order to achieve this strategy, project management must develop plan for risk mitigation. (2) **Risk monitoring**: Risk monitoring is a project tracking activity. The fundamental objectives of monitoring are to assess whether the risk in fact occurs, to ensure that the risks are defined, and to collect information which can be used in future risk analysis. A project manager is responsible for monitoring the risk level and the factors that make the risk higher or lower. In addition, it is to monitor the effectiveness of the risk mitigation. (3) **Risk management and contingency planning**: The management will issue a contingency plan when the mitigation efforts failed and the risk becomes worse.

Risk evaluation is how the various risks interact with each other and how the total development exposure to risk is conducted. Risk evaluation is generally concerned with assessing and prioritizing the risks, by considering the necessary trade off to recommending a desirable course of action and determining what aversion strategies are applicable [JPH94].

The metrics for software risk management [SHE96] introduced two risk management metrics. The first metric deals with requirement stability throughout the phase of software
development, while the second deals with incremental validation of requirement as the project progresses from phase to phase.

2.8 Software Configuration Management (SCM)

Changes are inevitable in software development. The customer changes the requirement, the developer changes the technical approach, and the manager changes the project approach. Software change if not properly managed can quickly lead to serious problems. One of the reasons for the falling behind schedule is the change of requirements, which lead the programmer and the designer waste time and effort [JPH94] [CAP96]. For this reason, the configuration management is there to control and monitor these changes. The fundamental task in software configuration management is the baseline concept, which helps to manage these changes by providing the managers with the visibility and the control over the software development process, coordinate the activities of different individuals and teams involved in developing software. Baseline management provides the basis for establishing project milestones and it is also subjected to a critical design review. In other words, it is a set of procedures for tracking and documenting the product throughout its life cycle, to ensure that all changes are recorded [STE94] [JPH94].

The configuration management activities are:

1. **Configuration identification:** It sets the function of the configuration management system, which starts with the selection of configuration items (CI) (the configuration items is an information that is created during software engineering process) and ends with the coding system.

Software Project Management 36
II. Change control: The mechanism of controlling change comes from the combination between human procedures and automated tools. Controlling the change involves the following steps: (1) the preparation of a change request, (2) evaluation of a change request, and (3) implementation approved changes.

III. Audits: The configuration management should provide the customer with the assurance that the test plans represent the required performance and the results proven conformance to all the requirements.

IV. Status report: It is an updated information about what happens, by whom, when it happens, and what else will be effected [EDW84] [AVR94] [ROG97] [IAN92] [IAN95].

Configuration management tools must provide the following to ensure the effectiveness of the configuration management system: (1) the capability of controlling accesses to libraries of components. For example, preventing the overlap between two members of the software team checking out the same component at the same time; (2) the capability of determining variants and revisions of a program; (3) the capability of keeping the system in a consistent state after modification [CAR91].

In order to ensure the completeness of the proposed configuration, it must be verified and validated. There are two types of audit. First, the physical configuration audit (PCA), which determines whether the product and the documents match the requirements. Second, the functional configuration audit (FCA) which determines whether the functional capabilities of the product match the specification. The report and correct actions required if there are any defects resulting from the audition. Due to the very high
cost of large software systems and the time taken for the configuration, auditing prevents a lot of problems [STE94].

2.9 Software Project Management Quality

The quality has been defined in different aspects such as zero defects in the delivered software and also defined as the user requirement satisfaction [ROG97].

The main concern of software project is to deliver the product on time, on budget or with the lowest cost possible, with zero defects, high quality, and meeting the user requirement. It is well known that there are three factors which have a strong influence on software quality and organizational performance, that are people, process, and the complexity of the user requirement.

Quality control involves the inspections, reviews, and tests used in software development life cycle to ensure that each work product is based on project plan and meets the requirements [ROG97]. Thereupon, the software quality is completed through software quality control. The standards such as how dates will be collected, how the format be specified, what necessary documentation will be needed, and how design and implementation should proceed, are considered the basis of measuring quality control [THO97].

Total Quality Management (TQM) is a system combining organizational models and quality control. It is designed as an organizational management concept, which is directed by the customer.

Quality Cost is the cost of performing the quality for each activity. Quality Cost is divided into costs associated with (1) prevention costs, such as quality planning, technical
reviews, test equipment, and training, (2) appraisal costs, such as in process and interprocess inspection, equipment maintenance, and testing for each process, and (3) failure costs which is divided into internal failure costs such as rework, repair, and failure mode analysis and external failure costs such as complaint resolution, product return and replacement, help line support, and warranty work.

The Japanese management is considered the character of successful management due to their views and approaches to the cost. Thereby, the quality of their projects have improved and increased. Their achievement are based on the following: (1) Japanese managers find that it is important to win the customer’s trust and they realize poor quality can cause high costs. Thus, they invest more on the reasons to improve the quality, (2) Total Quality Management (TQM) is a system that combines quality control techniques and organizational models. Studies have shown that adopting this practices have achieved better relations among the staff, higher productivity, and ensured customer satisfaction, (3) they recognize that the cost of poor quality could be reduced more effectively by concentrating on the process of new product development, (4) they discovered that by developing good products with high quality standards, the company can charge the customer more by educating the customer to demand on the highest quality [AVR94] [VOS90].

2.10 Software Quality Assurance (SQA)

Quality assurance is audited information about each work produced to inform the management about the product quality [ROG97]. The SQA team is responsible for ensuring that every phase in software development is correct and audits the software. The SQA team is also responsible for reviewing and auditing a product when it is completed.
The fundamental aspects that the SQA emphasizes on are: (1) software requirement, (2) the standards specification and the development criteria, and (3) the failure to meet implicit requirements although the system conforms to its explicit requirement. The main activities of the SQA team are as follows: (a) prepare SQA plan, (b) participate in a software process, (c) review, verify, and audits, (d) work document, (e) coordinate and control, (f) record and report, and (g) analysis and statistical [ROG97].

2.11 Software Project Management Tools: Analysis & Comparison

Project management tools allow the managers throughout the organization to see what is happening and to understand the impact of not maintaining their schedule and providing better data to make better decisions. Furthermore, it saves a lot of time spent on scheduling, calculating duration, evaluating cost, and producing reports in a standard format. There are many project management tools and most of them are focused on: Work Breakdown Structure (WBS), Gantt chart, resources-driven scheduling, calculates the activity duration (based on the resources assignment and the effort factor), displays the project information and generates progress reports. However, they are insufficient to cover all project management problems. During this research, some of the project management tools were analyzed and evaluated (Appendix A) for the following reasons: (1) to present the overview of the current features, (2) to investigate the limitation and, (3) to evaluate the need for development tools to help minimize problems and increase productivity in software projects [HAN98].
2.11.1 Tools Evaluation

The tools evaluated are as follows: Open Plan (OP) professional V 1.2 by Welcome Software Technology (WST) 1994-96, Microsoft Project (MP) V 4.1 –1995, and Enterprise PM (EP) V 3.3 by Platinum 1995. These tools have been developed to be used under the windows environments.

The tools were evaluated according to the following criteria:

1. Usability

1- Easy to learn:

a- MP is the easiest to use, from the starting screen until creation and modification of a project. It provides multiple ways of creating a project, and includes “help” facilities such as “answer wizard” and “screen tips”.

b- EP uses the calendar screen to assign and identify dates, rather than typing in.

c- EP has the unique feature of linking methodology for each task or activity. This is one of the useful features in project management that we have gathered from our survey [KHA98].

d- OP uses icons for representing resources, histogram, costs, relationships and the project management director (PMD). This simplifies the selection of items.

2- The amount of effort needed:

EP has the lowest effort and time needed to create a project by using sequences. For example, first the user starts creating a project by choosing ‘new project’ from the menu to fill in general information. Second, the user
creates the activities by using a button in the toolbar or double click on the spreadsheet to fill in the activity's form. A longer time is needed to create a project using other tools.

3- Flexibility

a- EP provides methodology links where each activity can be linked to their methodology to monitor or view the tasks. This is not available either in OP or MP.

b- OP has a unique methodology module that allows the user to implement project management practices and procedures beyond just scheduling. The Project Management Director (PMD) guides the user through the entire project management process offering guidance and tools in the area of scope, time, cost, human resources, communications, quality, and risk.

4- General satisfaction.

a- EP provides a convenient way of assigning dates while OP and MP requires the user to type them in.

b- Both the MP and OP, uses the milestone technique to indicate an important checkpoint in the schedule such as completing phase of the project, and viewing it on the bar chart.

c- OP enables the user to use the web publisher to distribute reports over the Internet or company’s Intranet.

d- EP indicates the percentage of the resource’s time that is consumed by non-project events in the resource calendar.
e- MP organizes large projects that have a number of phases, and refer to these subprojects in a master project.

f- MP can avoid the resource conflicts when the projects share the same resources by consolidating the projects to create a common resource pool.

II. Multi-project environments.

a- Hierarchical structures with roll-up functionality, inter-project relationships, cross-project resources analysis, security, and the ability to handle large amounts of data are features built into the Open Plan allowing the user to manage work, resources, cost and risk across many projects.

b- EP can schedule any number of projects in the priority order entered. Based on resource availability and relationships to other events, multi project scheduler performs finite scheduling for all project in the list.

III. Import and Export.

Each of the tools has this feature, which allows the exchange or transfer of project management information from other applications support MPX (Microsoft Project Exchange) format, such as spreadsheet, database, or word-processing documents, or importing files saved as MPX format.

2.11.2 Analysis And Limitation Of Current Tools

Analysis of the tools can be summarized in the following manner:

- Each of the tool has individual useful features, which, if combined, would provide a very powerful tool. Some of the useful features are:
a- Methodology link for activity.
b- Screen tips, Auto correct, Answer wizard.
c- Assigning of dates through selection in a calendar to click on the chosen date.
d- View the milestone on the bar chart for tracking the projects.
e- Calculate the total float, free float, and hammock between the activity to monitor the projects.
f- Filter and sort the activity based on some criteria, for tracking the project.
g- Multi-projects.
h- Integrated system.
i- Using schedule with electronic mail.
j- Using Project Management Director (PMD).
k- Reduce the complexity in big projects by creating a separate project file for each phase, and then refer to these subprojects in a master project.
l- While the companies compete to add new features to the available tool, it is noted that they attempt to be unique from each other. This results in changes to the vocabulary and terms used. This confuses the users. Time has to be spent to search for the meaning of these terms.

All the previous features are concentrated scheduling, planning and monitoring activities. However, scheduling resources in a large project involving sub-projects is still a problem. If there is a delay in any project, the delay might cause delays in other sub-projects especially if the project is dependent on few resources. Furthermore, statistical quality assurance based on past projects can help enormously in identifying problem areas.
2.12 Conclusion

This chapter presents an overview of the software project management principles, techniques, processes and other related subjects such as risk management, quality, and configuration management. An analysis and evaluation of three project management tools is also presented in this chapter which discusses both the general and specific features and the limitation of each tool. It is hoped that through analysing and making comparisons, combined features including additional new features can form the basis and requirements of a new project management tool. The next chapter will discuss the results of survey done in Malaysia to investigate the problems in software project management.