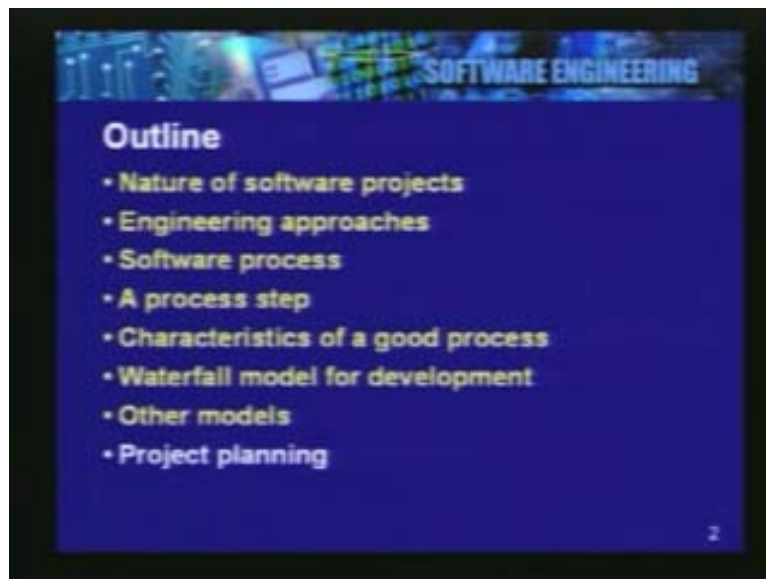


Software Engineering
Prof. N. L. Sarda
Computer Science & Engineering
Indian Institute of Technology, Bombay
Lecture - 1
Introduction to Software Engineering
Challenges, Process Models etc.
(Part 1)

The topic for this lecture is introduction to Software Engineering. Let us first see the outline of this talk. We will begin by looking at the nature of software projects, why they are challenging, why very often the projects are not successful. Then we will see how an engineering approach can be helpful to us. We will particularly look at engineering projects in other domains not necessary only software engineering and see how we can learn from them. After that we will talk about software process. We will define typical step in a software process which can be helpful to us in developing large software systems. We will then define what the characteristics of a good process are and we will look for such a good process for development. We will examine waterfall model for software development, we will also look at some of other well-known models and finally we will briefly touch upon project planning. You will see more on this in another module.

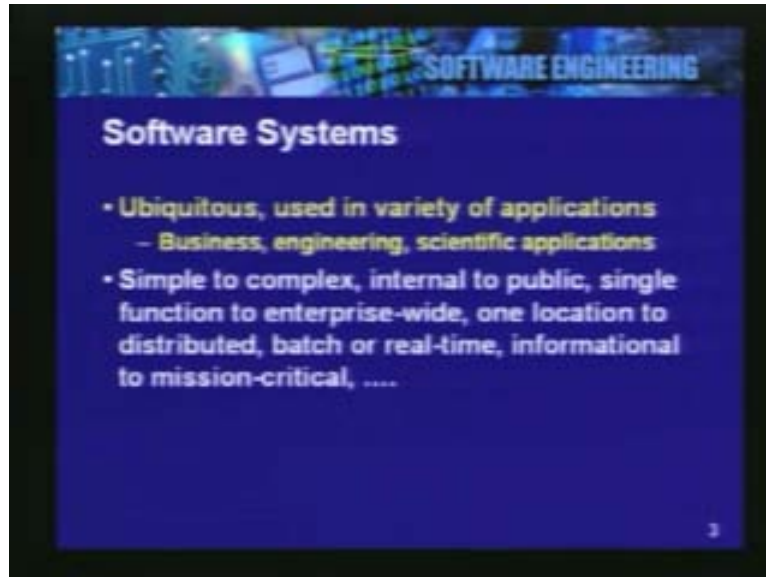
(Refer Slide Time: 02:05)



So let us begin by the nature of software systems. Now all of us have encountered software systems and we see them in a variety of applications. The different types of applications you encounter may be in business domain, in engineering domain or may be in scientific applications. Basically the point is that computer software today is touching all aspects of human life. Now these software applications may be of different categories. They may be very simple or they may be complex.

They may be meant for internal use within an organization or they may be meant for public use such as the railway reservation system that we are all familiar with.

(Refer Slide Time: 02:53)



In fact, it is an example of a very successful software system which is of the public usage category. Or the software solution could be something for a very single well defined task such as a pay roll, or it could be an application which covers all functions of a business organization, that means it could be enterprise wide software. You can have software which functions from one location, or it may be a distributed software solution, software solution which may be designed for batch operation, or it may be a real time, or it may be meant only to give information to public, or it may be a machine critical. So you have a range of software solutions and it's a very important area because software is now used in almost all applications.

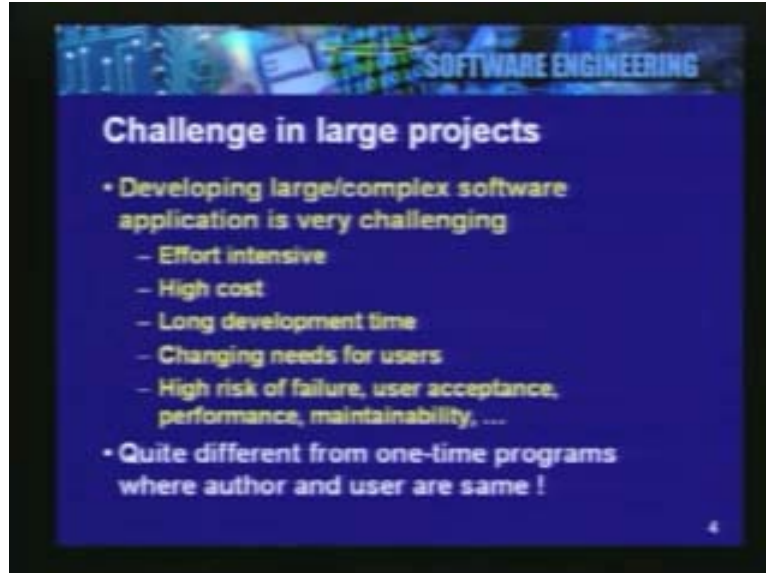
(Refer Slide Time: 03:57)



Now there are major challenges in developing software, what are these challenges, and why do we have this challenging situation? So developing large and complex software is challenging because, firstly it is very effort intensive, we have to organize the work very meticulously, and high cost is involved. So in case the solution is not successful the entire investment goes waste. It also can take long development time. Then we have to take into account the changing requirements of the users. And finally there is a high risk of failure. And this risk may be in terms of variety of aspects.

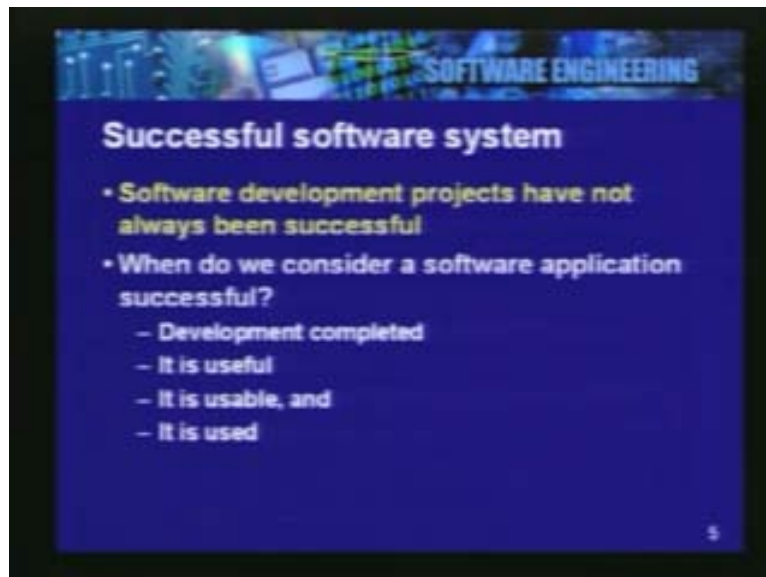
The users may not accept the solution, it may not meet the performance, or the solution is such that it is not possible to maintain it up to date. So these are different challenges in developing large software projects. We must realize that developing a software is quite different from developing one time program which we typically develop as a student or for ourselves, we just develop try it up, see if we get some useful results and throw it away. We are not talking of software solutions of this kind. We are talking about software solutions which are of long duration. They give important service to people or organizations and this is what would be the focus of software in our course.

(Refer Slide Time: 04:57)



When do we call a software solution to be successful? Basically software projects have not always been very successful. The success rate in fact is much lower compared to the success rate in other engineering domains. When do we consider a software application to be successful? These are the different criteria on which we will say that software developed by somebody is successful.

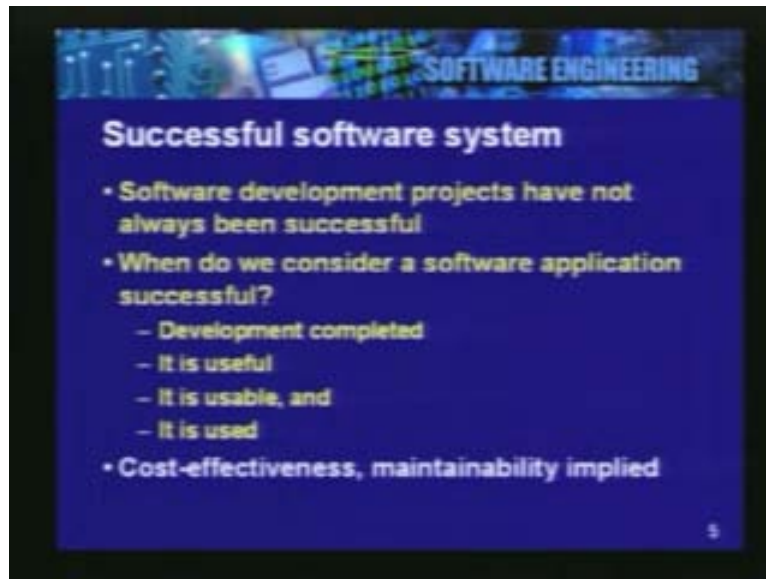
(Refer Slide Time: 06:03)



Firstly the development should be completed. Now that sounds obvious. Unless the development is completed the software cannot be used. But this is the reality that very often projects are abandoned in between and then they are not completed.

Now having completed the software then the other criteria are that the software should be useful. It should be usable and finally it must be used. This is what it really means is that users would find attracted to the solution that you have developed, and they would be keen to use it in their day to day functioning.

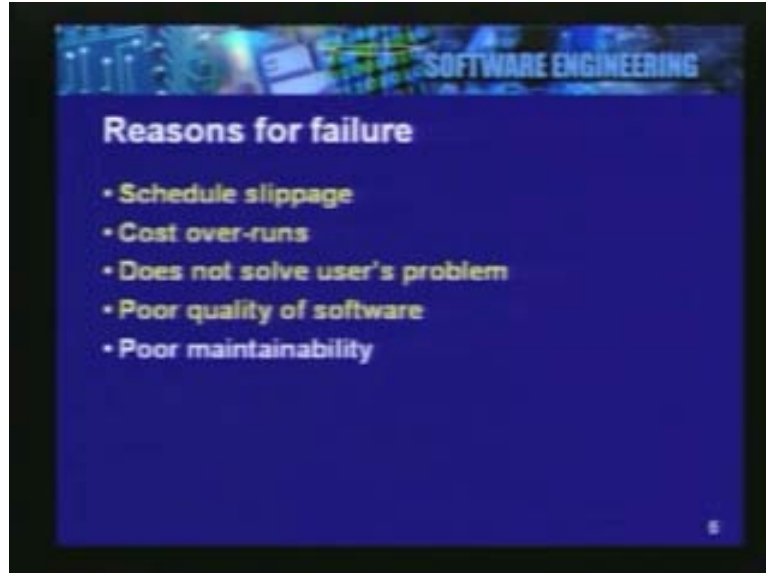
(Refer Slide Time: 07:00)



What all these criteria are implying is that, besides being useful, the software should be also cost effective and should be maintainable. The point we are making is that software has a fairly long life and it needs to be maintained. It should be kept up to date just like we need to maintain other equipment. So only the software solutions which are usable, cost effective and maintainable would be considered successful by any user.

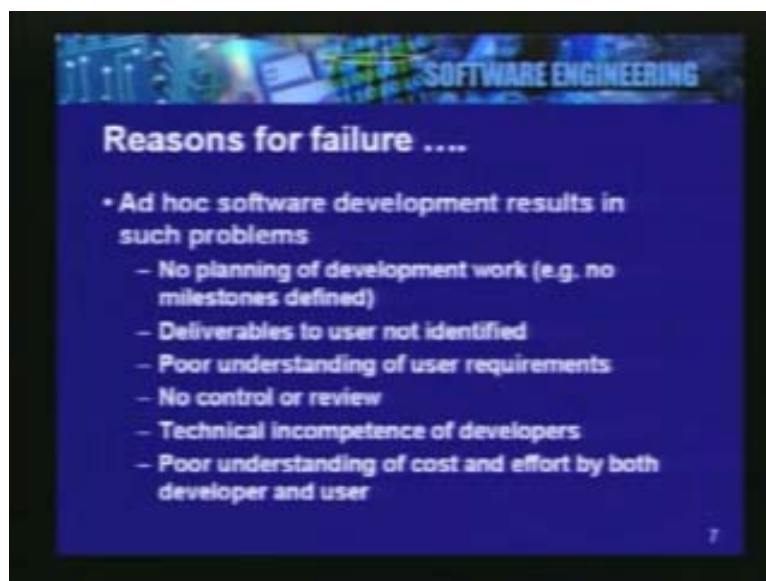
What are the reasons for failure? There can be many reasons; the first is the slippage in the schedule, meaning the software is not ready in time. Or it is becoming too costly to complete the project and it may be therefore abandoned in between. It does not solve the user problem, if the software development has been completed but the users do not find it useful, it is a failure. The software may be of very poor quality. It may not be maintainable. So these are the different reasons for failure. What leads to such problems?

(Refer Slide Time: 08:06)



Primarily the main reason or the important reason for such problems is the ad hoc approach to software development. Very often we assume that it is so straight forward you have to just sit across a machine and talk to the user, quickly understand the requirement and start coding. This is not the way to develop large software projects. You need to understand that such approaches are likely to or very likely to lead to failures. So let us see what these reasons are.

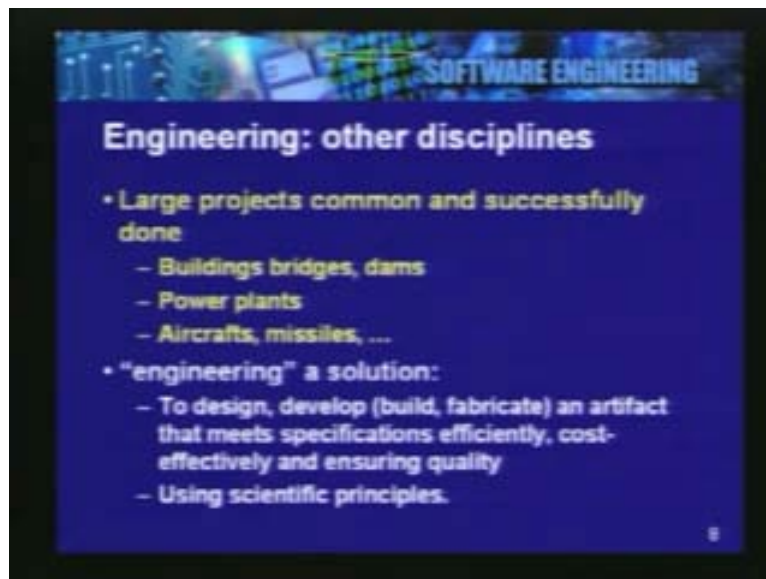
(Refer Slide Time: 09:00)



Very often there is no planning done for the development work. So when a project is undertaken we need to define different milestones, how the development would be completed step by step. Deliverables to the users are not identified. There is a very poor understanding of user requirements on what exactly does the user expect from software. There is no control or review. In fact for any large project it is very important that we control its development very systematically, we review the progress systematically. Because we are committing lot of resources, we are committing lot of cost and time. Therefore it is very important that we control and review the progress continuously. Of course the reason for failure could also be technical incompetence of the developer. They need to understand the changes in technologies, the new techniques and tools which are available for software development. So the developers also need to keep themselves up to date with the latest trends and technologies. Finally there is another reason that both the users as well as the developer have a very poor understanding of the cost and effort involved in large software project.

So these different types of reasons basically which is implying some kind of ad hoc ness in software development. These are the reasons which are typical for a failure of a software project. Now we see in day to day life that there are many other successful engineering projects. These are large projects and they carry out typical functions which are requiring a large effort in development. So some of the projects which are mentioned here you can see that they are very common and they are very successfully completed.

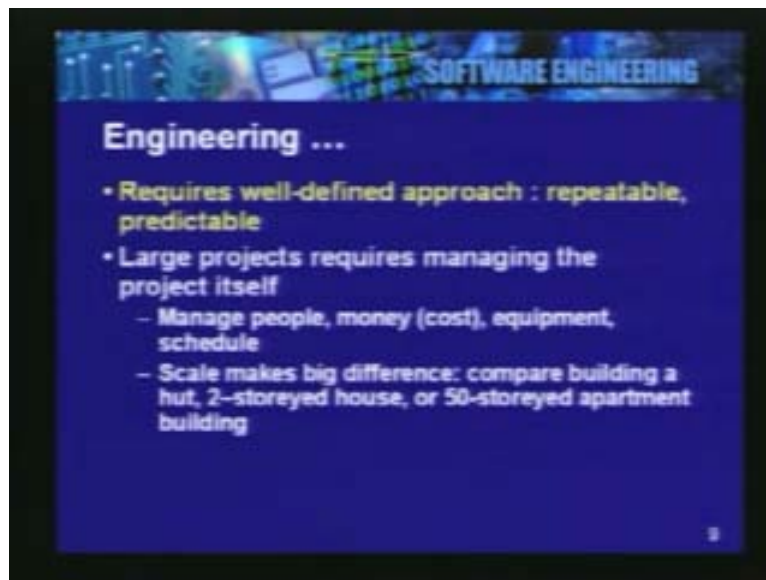
(Refer Slide Time: 10:57)



For example these projects could be building large bridges, dams, power plants, even aircraft, missiles. These are big engineering artifacts, these are big engineering projects and they generally you see that they are successful. So what we would like to see is what is that these other disciplines have, which we need to learn and also apply for developing software. Now what is common here is an engineering way of doing things. What does this engineering way consists of.

Basically what it emphasizes is that, we must go through a cycle of defining what we want to build, what are the functions an artifact should do, how to design it and then we should develop it. So there is a specified design by develop cycle which goes through the process and which leads to the development of the solution. So an engineering approach consists of developing something, which meets the specifications efficiently, cost effectively and at the same time they ensure quality. It is the responsibility of developers to ensure that the project would meet the specifications cost effectively and also that we have good quality build into the solution. So there are some scientific principles in every engineering domain that we must apply to arrive at a good solution. Engineering approach basically means that, we should have a well-defined repeatable and predictable approach. Large projects will benefit by a very systematic approach. It should be Repeatable. So that if we are successful once, then the same approach, the same methodology can also be used in another approach, in another project and we also will have a very high probability of a success. So repeatability and predictability are important characteristics of an engineering approach. Large projects where we have large number of people, funding involved, need to be managed. And the scale makes a big difference.

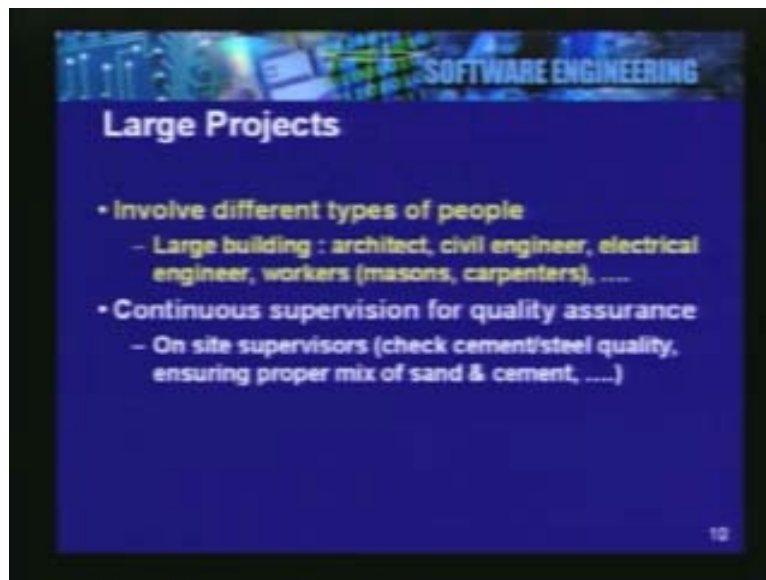
(Refer Slide Time: 13:36)



Compare building of a house, if you are building a very small house, may be a hut or you are building a 2-storeyed house or a 15-storeyed apartment building, it is a question of scale. The projects are becoming larger and larger, and therefore you are committing more resources, more people, more cost. And therefore it becomes very important that these projects are handled systematically and we do appropriate the management of the project. So large projects have to have a good systematic approach, the project itself requires management. Quality is of course important and what does it relate to. Quality relates to failures obviously. So something that we design must not fail. It must meet the specifications and it should also have efficiency, good performance, and usability, besides probably other quality factors. We must note that people are willing to pay for quality.

As we see in daily life that when we go and buy something, may be you are buying a T.V, you have a sense of quality. You know that different equipment or different T.V sets have different prices. And you are willing to pay a higher price, because you feel that the equipment offers a better quality. Now same thing would happen with the software. If we are developing good high quality software, we can expect users to pay for such high quality software. So quality is important in any engineering approach. Large engineering projects of course, involve different types of people and we need to manage the work carried out by these people. For example, again considering a development of a large building here you will have people who are architects, civil engineers, electrical engineers, masons, carpenters etcetera will be involved in the team, which will develop the building. And they will all have specific qualifications for contributing towards the project.

(Refer Slide Time: 16:24)

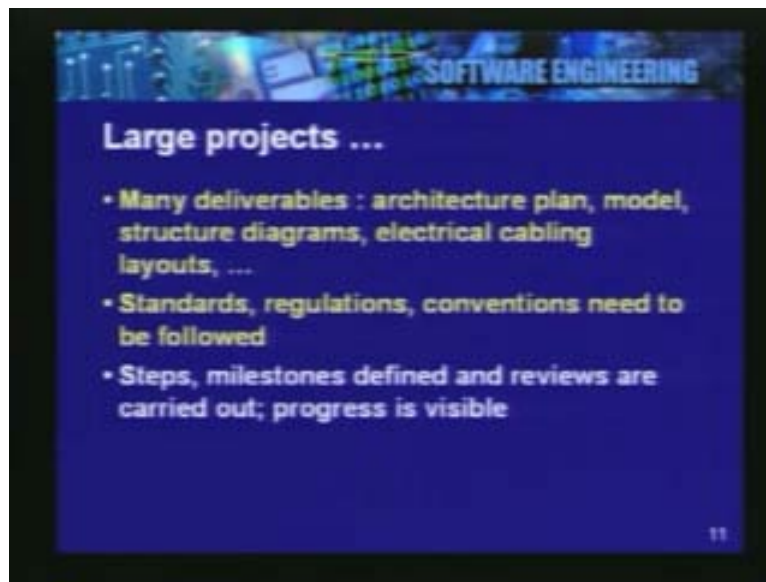


When they carry out this project, they also need to be continuously supervised. Supervision is an essential aspect of quality assurance and we know that this is done by different people whose role is to carry out day to day on sight monitoring when buildings are being constructed. In a large project, there are many deliverables. Things are not developed over night.

So again taking example of a civil engineering project, you have different deliverables which are produced in a specific order and they are reviewed and only when they are accepted we go to the further development. For example the architectural plan is made by an architect. The user may even want to see a model of the building, you make them structural diagrams, you make foundation diagrams, and you make electrical cable layouts and so on. So there are many deliverables and these are reviewed and only when they are approved, we go ahead and carry out the task. More over to further guarantee that work and its quality, we have some well-defined standards.

These standards or regulations must be followed in order to get a successful engineering solution. In every engineering field such standards, or regulations, or conventions are well established and we must be aware of these. The purpose of the standard is to ensure high quality and to contribute towards success of a project. So as we carry out a large project, we go through multiple steps, we achieve different milestones. These milestones, which may be for example, be development of an architectural plan. These milestones are delivered, reviews are carried out and when they are accepted, we go further. Because of these well-defined milestones in every other engineering project, the progress is visible.

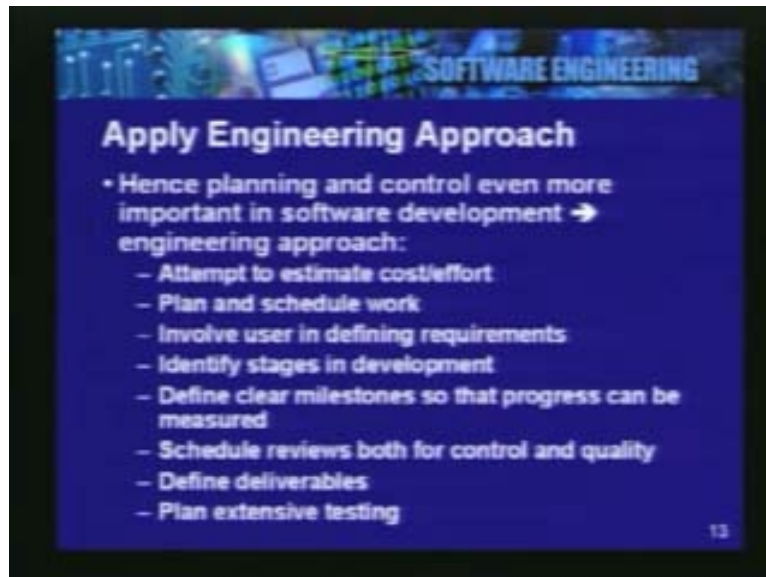
(Refer Slide Time: 18:33)



And also have a continuous management in place, so that we ensure a smooth progress in the project that we are doing. Now software, of course is different from other engineering domains. We can list some of the essential differences; firstly the cost of production in software is concentrated in development. It is easy to make copies. So once you develop a successful product it can be installed at many locations. That by itself does not have a cost associated. So much of the cost is concentrated in the development itself and therefore we need to ensure that this is a cost effective in its approach.

Maintenance in the case of software consists of making corrections and making enhancements. And progress in development of software is often difficult to measure. Sometimes we hear that software is eighty percent complete or 90% complete. Actually it is very difficult to get a sense of what this eighty or ninety percent may mean. This is a problem typical of software development because it does not have any physical dimension as you see in the case of a building. So when you see that the building which probably is a 50-storeyed building has already reached thirty five stories, you get a sense of progress. But this doesn't happen in the case of software. It is very important that the methodology or the approach we follow is a step by step approach and every step when completed gives us a signal that we have made some definite progress in the development. So it is very important that we define these stages very clearly.

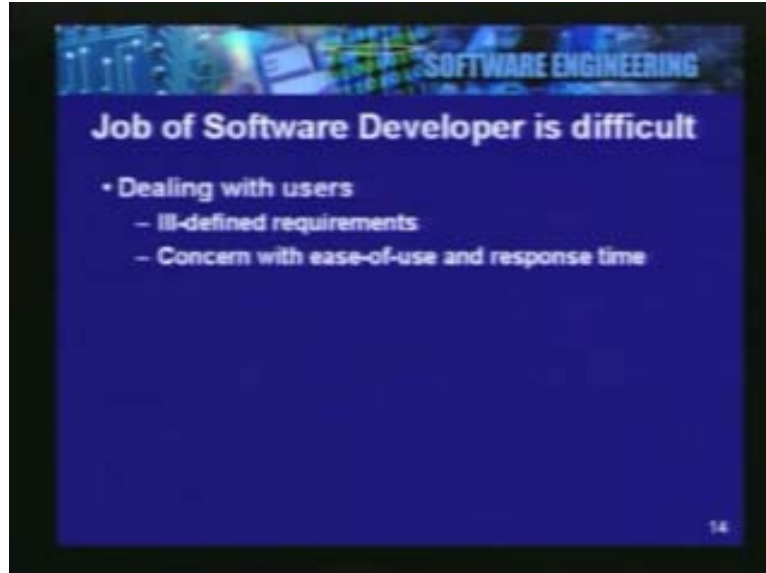
(Refer Slide Time: 21:05)



We need to apply an engineering approach to the software. Because software is different from other products it is even more important that we take an engineering approach in software development. What are the characteristics, what do we need to do in following an engineering approach for the software? First we should attempt to estimate the cost and effort involved in a software project. We should plan and schedule the work. We should involve users in defining requirements, what exactly is expected from the software, only users can tell us. So we must involve them in the beginning itself to guarantee that they will accept the solution that we are developing. Then we should identify the stages in the development. We should define clear milestones. So that once those milestones are achieved, we know how far we have come and how much further we have to go. So progress can be seen. We must schedule reviews and the purpose of scheduling of reviews is both for the quality and for the control. We should define deliverables.

And any large project requires extensive planning. And in order to control quality and the workability of the product we should include some steps which will ensure quality. For example testing in software is an important step and we must plan extensive testing. So that what we develop and before it gets used is usable. And this can be ensured through extensive and proper testing. These are the different guidelines for ensuring a successful software project. Now what do we need to do is, somehow we need to put all this in the form of a proper process. And use that process as an engineering approach to software development. We need to develop a methodology which will take into account, these aspects of software and these issues in development of any project to define some kind of methodology or an approach.

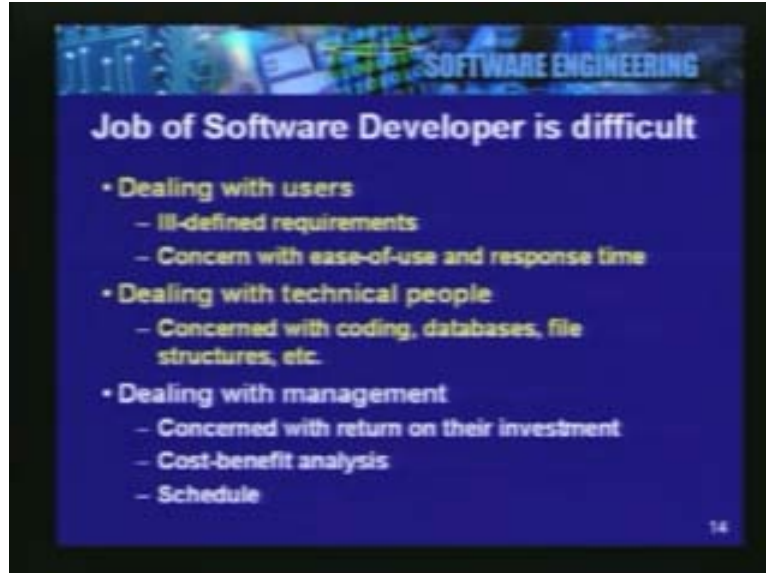
(Refer Slide Time: 23:30)



Now job of software developer naturally involves interacting with different people, and that is what makes it slightly challenging or difficult. Software developer on one hand has to deal with users. These are the users of the software. They are having some expectations. They are faced with some problems and they expect the software to solve those problems. But very often they are not in a position to clearly define what they expect from the software. So we are dealing with users who have ill-defined requirements in their mind. Their focus is ease of use and response time. They want software to be very simple to use, it should also be very efficient. So these are what users expect from software. On the other hand software developer also needs to deal with technical people and these are the people who are going to develop the solution. And they do not share the same concerns as expressed by the users. But their focus is to develop the code for the software, do the database design etc. That is, they are more technically inclined people. They are not so much concerned with ease of use or functionality and so on.

And finally the developer has to deal with the management, whose concern is the cost benefit from the project. They consider the project as involving some investment; they are expecting proper return on this investment. They are concerned about the successful project and they want to develop the software in limited cost and in limited schedule. So the job of the software developer is challenging because he has to balance the expectations of different people who are associated with the project.

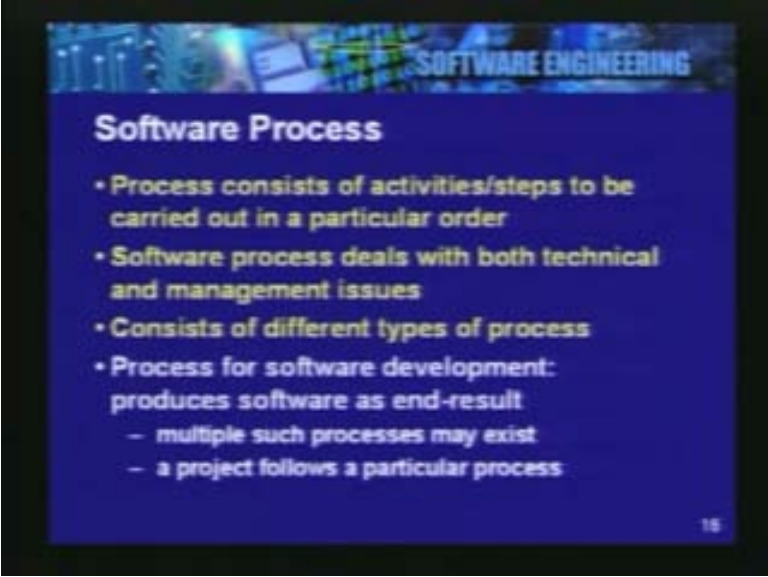
(Refer Slide Time: 25:16)



So in summary what we have really said is that a large software development is very challenging activity, in order to ensure its success, it is important to follow an engineering approach. This engineering approach like other engineering approaches, in other engineering fields, consists of a well-defined process. So process is essential in software development. So let us talk of what kinds of processes are involved in the development of the large software. Before that first let us understand the term process itself. Process consists of different types of activities. It defines a set of steps, these steps or activities need to be carried out in a particular order. Software process deals with both the technical and the management issues. It has to develop a solution using computer technology, it has to deal with management and come up with a cost effective solution.

There are different types of processes involved in a software domain. Firstly there is a process for software development itself. The purpose of this software is obviously is to develop software. So software development is a process whose goal is to produce the required software. There may be multiple such processes in an organization. And when you undertake a new project you may decide to follow an appropriate development process. The first type of process we have identified here is the process for software development, using which the actual development happens. Then there is a process for managing the project itself. Because as we said earlier large numbers of people are involved in the project, many activities will be carried out, and it is going to be involving some cost. In order to control all these we need to have efficient management. So you need to define management process itself, which will consist of defining the plan for the project.

(Refer Slide Time: 27:27)



The slide features a blue background with a header banner at the top that reads "SOFTWARE ENGINEERING" in white capital letters. The banner includes a graphic of a circuit board. Below the banner, the title "Software Process" is displayed in white. The main content consists of a bulleted list in yellow text. The list defines the software process, its scope, and its types, with specific details about the software development process and its iterative nature.

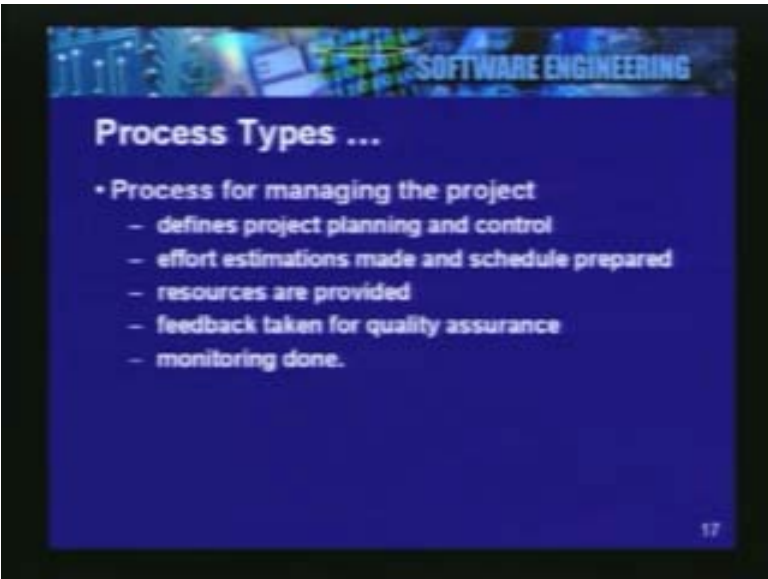
Software Process

- Process consists of activities/steps to be carried out in a particular order
- Software process deals with both technical and management issues
- Consists of different types of process
- Process for software development: produces software as end-result
 - multiple such processes may exist
 - a project follows a particular process

16

What does the plan consist of? Plan consists of identifying the different activities, when and how they will be done, who will be responsible for doing them. These are the components of the software project plan and the plan needs to be defined. Then we need to also have a process for supervising that plan. That is the control part. In order to prepare a software project plan you need to do the effort estimation, you need to prepare schedule, you need to provide resources, and you need to continuously take a feedback from the development process which will help you in controlling the project and also ensuring the quality. So you need to look constant monitoring. So the second type of process that we have defined now is the process for managing the software project.

(Refer Slide Time: 28:32)



The slide features a blue background with a header banner at the top that reads "SOFTWARE ENGINEERING" in white capital letters. The banner includes a graphic of a circuit board. Below the banner, the title "Process Types ..." is displayed in white. The main content consists of a bulleted list in yellow text. The list describes the process for managing the project, including planning, control, effort estimation, resource provision, feedback, and monitoring.

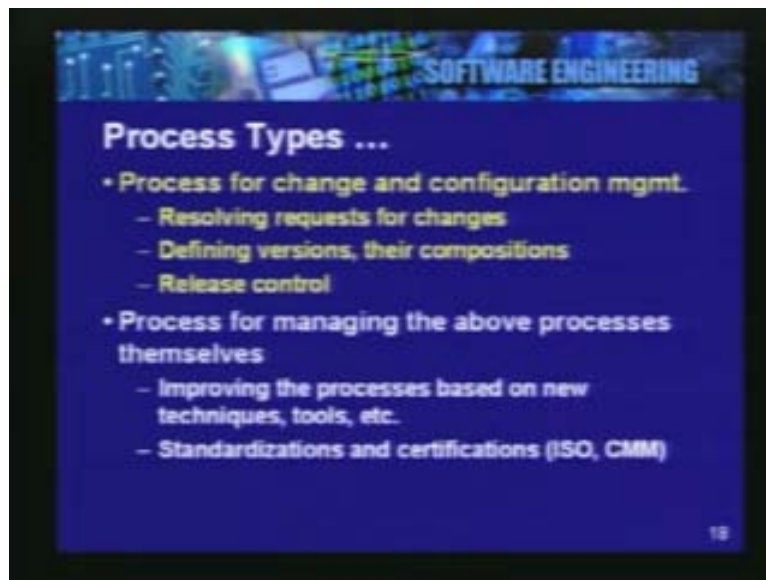
Process Types ...

- Process for managing the project
 - defines project planning and control
 - effort estimations made and schedule prepared
 - resources are provided
 - feedback taken for quality assurance
 - monitoring done.

17

And the next process is process for handling the change and for handling the configuration of the software that we are developing. What we have to realize is that software development could be a long activity, may require months to complete. And during this time there could be some changes in the scope of the software. We need to accept these changes, we need to resolve them, we need to identify whether we include those changes in the software that is being developed. And in the process we need to define the configuration of the software system that we are developing. If necessary we may create different versions of the software and then release them as and when the changes are implemented. So the third process is really the change management process and it is also very important for success of software.

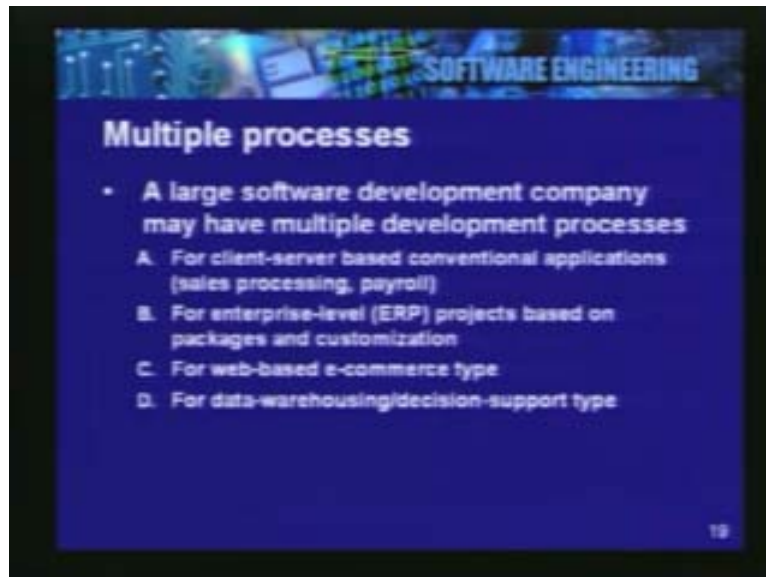
(Refer Slide Time: 30:25)



Finally there is a process for managing these software processes themselves. We talked about three processes so far; there was a development process, there was a project management, then there was the change management. It is important that all these three processes, should be kept up to date and they can be improved as new technologies and new tools get developed. And we should see whether we can benefit from new developments and new standardizations and so on. Many of you must have heard about ISO and CMM. These are basically the standards which identify improvement in the software processes and we may adopt them as we gather more experience.

So these are the different types of processes that we encounter in software development. So while talking about software processes we identified four categories one for development, one for project management, one for change control and finally something which will keep these processes themselves up to date. I mentioned that in an organization, there can be different types of processes and for development itself there may be more than one process that they may use depending on the type of project. For example, here an organization which has defined four development processes, for four different types of application.

(Refer Slide Time 32:07)

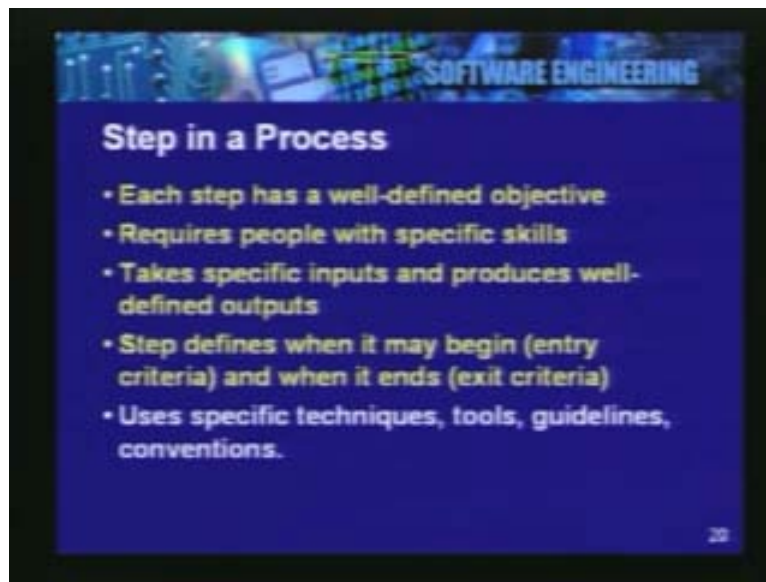


For example, Process A is to be used when we are developing client-server type of applications of conventional type such as Sales processing, Pay roll, and so on. Whereas Process B is a development process which we would use when we are developing an enterprise level solution, which consists of packages available in the market. So we will be doing development around those packages and we will be customizing them. So naturally Process B requires some different approach than the process A. Similarly in the case of process C, this is the process which the company may say that we should use when we develop e-commerce type of applications which are primarily web based applications. The process D is to be used when we are developing projects which are not for online transaction processing type of projects. But for projects which are more for management information systems or decision support systems, which would be based on new technology such as data warehousing.

So you can see here that an organization may have different development processes and depending on the type of the project they may choose one of these. So there may be many projects going on under each category. For example there may be three projects going on for three different clients using the methodology A. So we must keep in mind that a large organization needs to have a good repertoire of software development processes and based on their experience they should refine these processes to suit different type of projects. Now we said earlier that a process consists of steps. Let us now define this most important element of a process. Each step has a well-defined objective. It requires certain activities to be carried out by people with specific skills. For example it could be design of a database. So you need certain skills in order to carry out such an activity. It requires specific inputs and the step produces well defined outputs. So each step has certain input it takes, it carries out some task, and produces some outputs. Also more over we need to clearly define when the step should begin, are we ready to start that step.

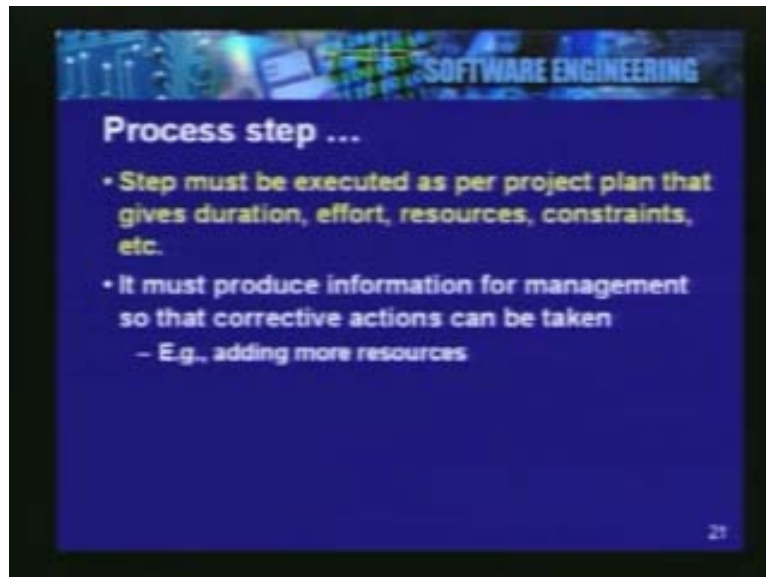
So there must be some entry criteria which tell you that if these criteria are met, this step can begin. And also you should clearly know when the step ends. That means that there must be clear exit criteria. There must be well established goals, and these goals must be achieved by the step. So these are exit criteria. Finally to carry out its task the step will use certain techniques, may use some tools, guidelines, conventions, standards and so on. Ultimately what we are saying here is that in order to carry out a step with well-defined goals, you need specific skills, you have certain commitments to fulfill, and you must use some guidelines, standards or techniques in order to reach the goals of that step.

(Refer Slide Time: 36:03)



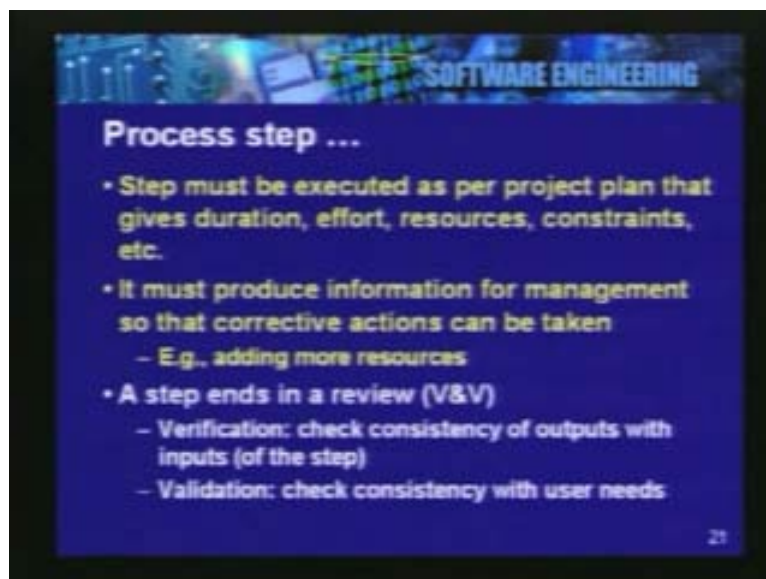
So carrying out the development consists of executing the steps one by one. They must be executed as per the project plan. You must remember that every step would be assigned some duration, that you must complete this job or this step in one month. We will also identify the effort required; we will identify the resources needed, and the constraints. All these are inputs which come from the project management into the step. These are the goals that the execution of the step should identify or meet. It must produce information for the management. Every step is not executed in isolation, but it is part of the overall process. It must produce some information which the management can use, so that corrective actions can be taken.

(Refer Slide Time: 37:31)



For example the effort estimated may not be appropriate. We may discover that more work needs to be done and we need more people in order to complete the process in the given duration. Now management needs to be alerted about these corrective actions. So every step must produce some inputs for the management.

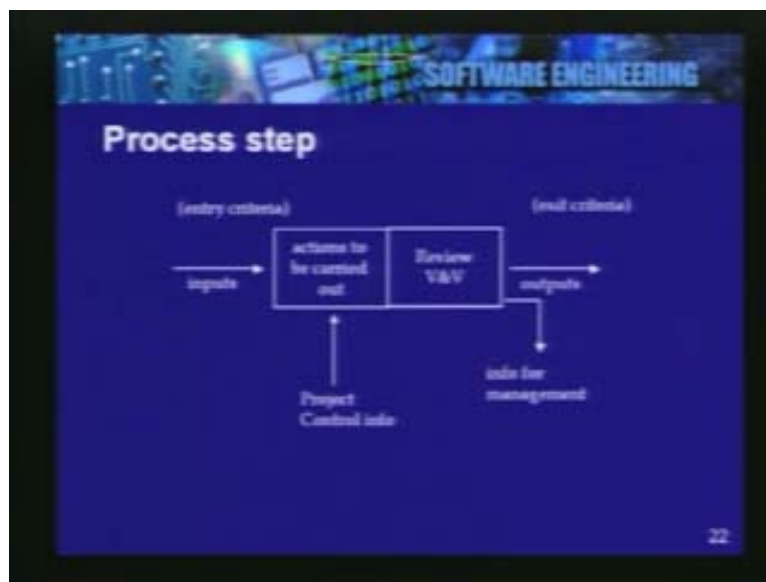
(Refer Slide Time: 38:00)



Finally a step ends in a review. This review we can call it as a validation and verification step. Purpose of this review is to ensure that the step has been successfully done. For example, if the purpose of the step was to design the software, then has it achieved its goal?

We do this by first verifying, that the step has produced outputs which are consistent with the input given to the step. This is called verification. Here the focus is only the step. If something was wrong in the input, naturally the step would continue with that mistake. So it should not happen, but the purpose of verification is to see that outputs are consistent with the inputs that are given. Then we have the validation step. The validation step checks the consistency with the user needs. That is the overall context is now taken into account - What is the purpose? Why are we doing all this? What are the criteria by which we can say that we are on the track? And what we have done, meets the user specified requirements? This is the validation. So every review must consist of both the verification and the validation.

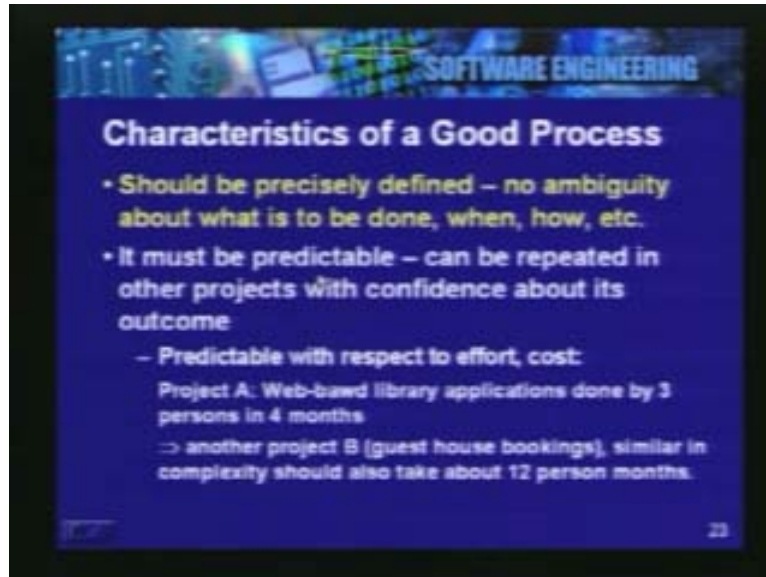
(Refer Slide Time: 39:39)



Here is a graphical picture of a step. As you can see here, the step consists of certain actions to be carried out. These actions would depend on the purpose of the step. And the step also has a review. It takes certain inputs, it produces certain outputs. There is an entry criteria clearly specified. This tells you what conditions must be met, so that this step can be initiated. It also tells you the exit criteria, when do we consider the step to have done its job properly. Then there are management inputs here. There is project control information such as the duration and the resources that we talked about. So these are specified and the step has to do its job within that duration and by using those resources. And the step also produces some information for the management so that the management can take some further corrective actions. So this is the overview of what a step consists of and a process will contain many such steps. Let us first understand the characteristics of a good process.

Then we will see how steps can be combined to construct such good processes. Now first criteria that we see here is that the process should be very precisely defined. There should be no ambiguity about what is to be done, when and how. That means each step which defines the process has a very precise meaning.

(Refer Slide Time: 41:30)

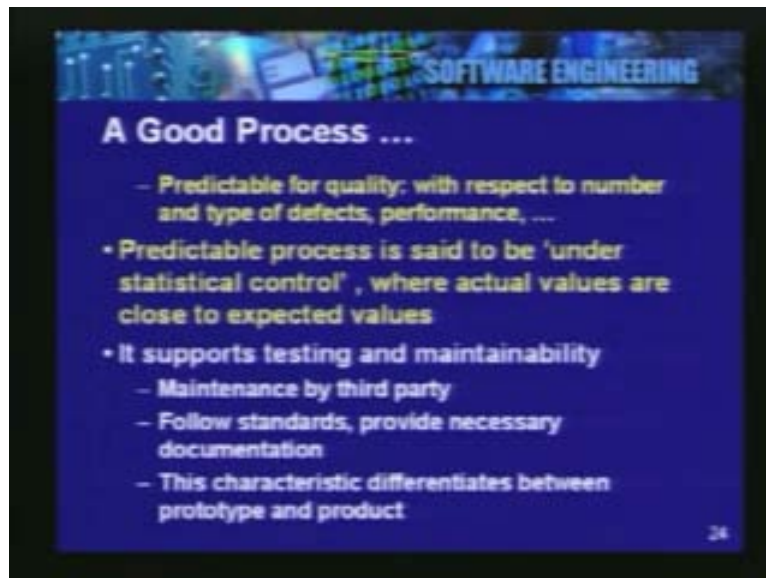


The step must be predictable. That means we define activities and we also identify techniques by which it should carry out the job. Predictability means, that step can be repeated in other projects or the process can be repeated in other projects with the same level of confidence about the outcome. Predictability is with respect to effort, with respect to cost, with respect to quality with respect to overall success. So here is a simple example. Suppose we did one project - Project A. This project was a web based library application done by three persons in four months. This was successfully done. Now, suppose we get another project B which is of a similar complexity. We feel that it should have a similar effort, which is the effort consisting of twelve person months, three persons in four months, which means the total effort of twelve persons.

We are able to relate the project A which was of a different category, with project B which is a different application. But, we are able to extrapolate the effort required because we have properly defined process. We know how to carry out such projects and what is involved, what are the different steps. So there is predictability about the development. This predictability is important characteristic of all processes. Then we should have predictability for quality, in terms of kind of defects, in terms of overall performance we can expect from the software. Primarily, the most important characteristic which is the predictable nature is essential. When do we say that a process is predictable? Another equivalent term is when do we say that the process is under statistical control? We say this when the actual values are close to the expected values.

Predictable process is said to be under statistical control. Implying that when we carry out that process again, we can reasonably expect what we will achieve and how much effort and time it will need. What it really means is that actual values are close to the expected values. In fact in real life we encounter and we always try to plan processes like this. Suppose you are going from point A to point B and you say that I may need half an hour to reach point B. Now we are always actually implying some kind of predictability here. That means that there is a very good chance that we will achieve for the objective of reaching in a specified time. So this is the predictability that we want to ensure in a software process.

(Refer Slide Time: 45:06)

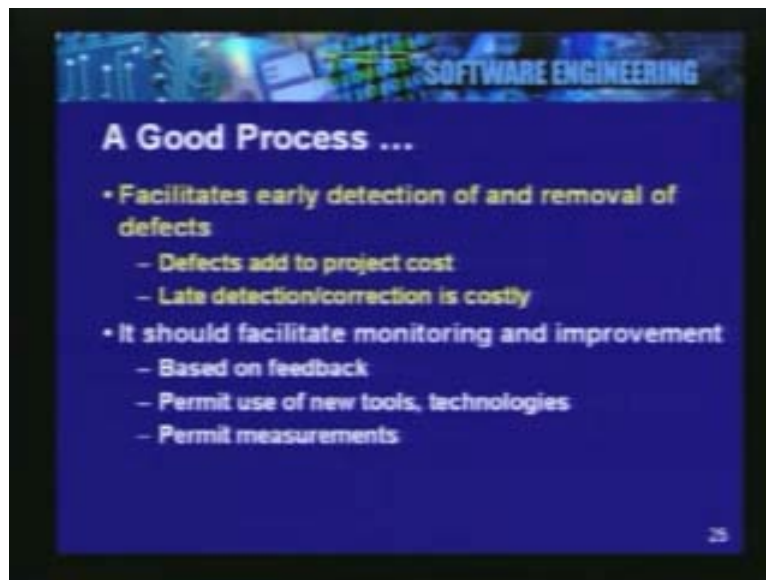


Then the software process should support testing and maintainability. Testing is important to ensure that software will serve the user without any defects. We have to also remember that the software developer may be different from the person who would maintain the software. In fact the maintenance is done by an entirely different organization. We can ensure maintainability if we follow proper standards in development, we carry out proper documentation. In fact the most important difference between a prototype and a product as you see in most other engineering activities is that an engineering product is a rugged usable and maintainable product.

Generally we do not care much about the maintainability of a prototype which is typically developed only to demonstrate the feasibility or the functionality. But when you develop a product, you must ensure that the customers are satisfied about its usage and they are also satisfied about its maintainability in the future. So these characteristics identify a good process. The processes that we are trying to define and identify the steps that they contain should be such that, they will ensure the quality by allowing us to detect, and remove the defects as early as possible. We must remember that defects add to the project cost. Especially when in the case of software, all the cost is concentrated in the development itself.

So a defect if it goes as part of the delivered software requires maintenance in future and that requires an additional cost. So late detection and correction of the defects is a very costly process and we must ensure that the software is defect free to extend possible that we take care to remove all defects during the development itself. That is a part of the quality software. A good process needs to ensure that. Finally a good process is one which it is straight forward to carry out proper monitoring.

(Refer Slide Time: 47:50)

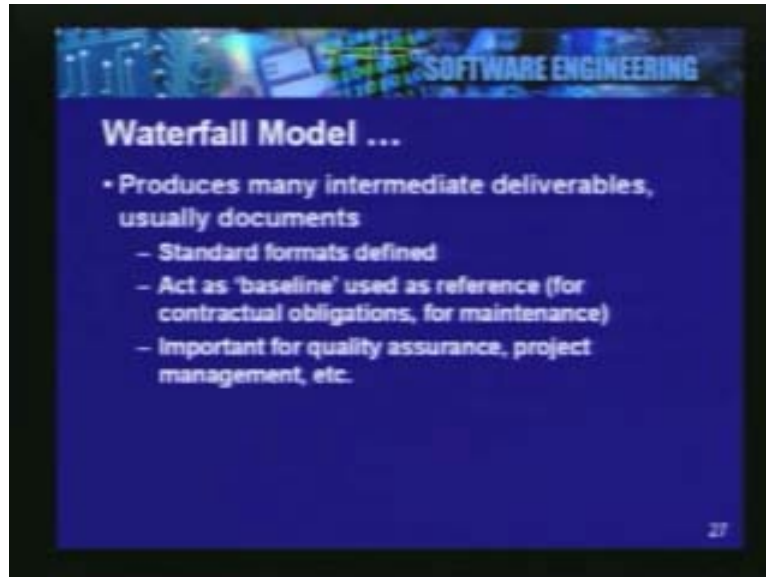


It gives enough data for the project management and it also enables improvement in future. So as we do multiple projects, as we learn, we get feedback from the ongoing projects, we should be able to improve our process, we should be able to improve new tools and technologies, and we should be able to evaluate by carrying out proper measurements what kind of improvements are possible and worthwhile. So there are these processes that we need to prepare for development of software. One of the popular processes is the waterfall model for software development. Here the various steps that the process consists of are arranged sequentially.

One step takes inputs from its previous step and gives its output to the next step. Exit criteria of a step must match with the entry criteria of the succeeding step. That means we are going to define the different steps and put them sequentially. So here the different steps are placed in a sequential linear order and waterfall model follows the sequence consisting of specification design and building. Now this sequence intuitively appears to be very obvious and natural. Therefore the waterfall model is commonly used in spite of some of the limitations that it has. It produces many intermediate deliverables. These deliverables are in the form of documents and there are well identified standards for these documents. So the important feature of the waterfall model is that the different steps that it defines have identified deliverables. These deliverables are actually part of the exit criteria of different steps and they have a well-defined standard. More over these deliverables act as a baseline, so that we can use them as a future reference.

They may be used to clearly define what our contractual obligations for the user are. Similarly these documents can be used for future maintenance. So it is very important that the process identify these baselines and waterfall model has very clearly identified baselines like this.

(Refer Slide Time: 51:00)



These are also very important for quality assurance project management and so on. So we will speak in details about the waterfall model. We will see what its phases are, what the different steps are, in what sequence they are carried out, and what the deliverables are. That will clearly define this important process of software development.