

Software Project Management
Prof. Durga Prasad Mohapatra
Department of Computer Science and Engineering
National Institute of Technology, Rourkela

Lecture – 19
Project Estimation Techniques (Contd.)

(Refer Slide Time: 00:23)



Good afternoon. Now, let us start the other Project Estimation Techniques. We will mainly a start about the parametric models, especially the Albrecht IFPUG function point model.

(Refer Slide Time: 00:32)

Parametric model for Size

- Important example: Function Points
- FPs originally used to estimate Lines of Code, rather than effort

Number of file types

Numbers of input and output transaction types

model

'system size'

NPTEL

So, let us see what is a parametric model, and then we will see the parametric model for size. So, one of the important example for parametric model for size is function points. Function points, where normally used to estimate the lines of code, rather than effort. Say effort directly cannot be estimated from function point, first we have to estimate the lines of code and by using the lines of code then we can estimate the effort using this function points. So, this model works like this. It takes the number of file types and the number of input and output transaction types are the input, and then it processes something and then it will produce the system size as the output.

(Refer Slide Time: 01:19)

Parametric models for Size

We shall examine four parametric models more closely :

1. Albrecht/IFPUG function points
2. Symons/Mark II function points
3. COSMIC function points
4. COCOMO81 and COCOMO II

NPTEL

See we have seen there are many drawbacks of the LOC model that we have seen yesterday. So, this function point approach, we will try to resolve some of the issues though they are though they are appeared in case of these LOC. Now, let us see what are the different parametric models for size. So, first one we will see that Albrecht IFPUG function points, then we will discuss Symons Mark II function points, then COSMIC function points, and then COCOMO81 and COCOMO II. So, out of all those parametric models, today we will discuss right now in this class Albrecht IFPUG function point.

(Refer Slide Time: 02:02)

International Function Point Users Group (IFPUG)

Purpose

- To promote and encourage use of Function Points
- To develop consistent and accurate counting guidelines

Benefits

- Networking with other counters
- IFPUG Counting Practices Manual
- Research projects
- Hotline
- Newsletter
- Certification

Extent of Usage:

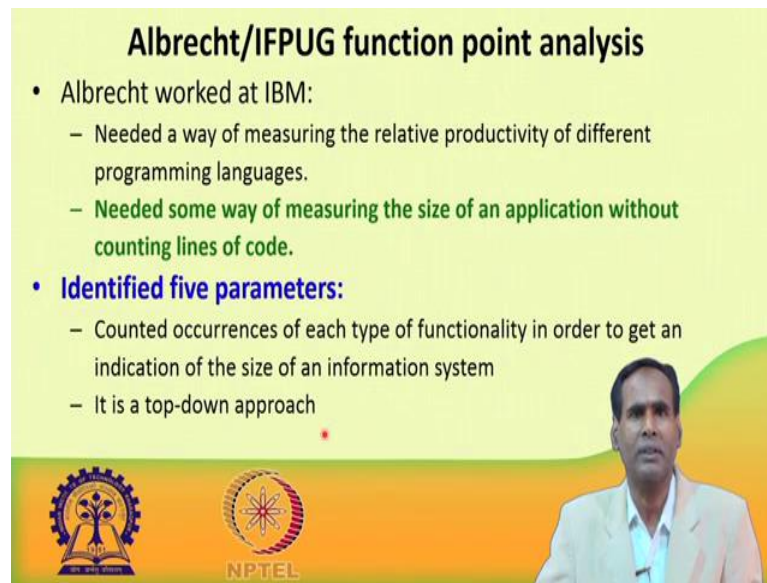
- Member companies include all industry sectors
- Over 1200 members in more than 30 countries

The slide also features logos for IIT Bombay and NPTEL, and a small inset image of a man in a suit.

Now, let us see how does it work; so what are the purpose, what are the benefits and what are the extent of use. So, IFPUG it stands for International Function Points Users Group. So, the basic purpose of this group was to promote and encourage use of function points, because we have already seen LOC has several drawbacks. To another objective of this group is to develop consistent and accurate counting guidelines.

Several benefits are there which can be obtained from IFPUG. So, like the benefits will be networking with the other counters, so you can make networking with the other counters. Similarly, IFPUG counting practices manuals are also available, they it a supports research projects, hotline, news papers, certification or some of the other benefits of this IFPUG. The extent of use is like is the member companies they include almost all industries industry sectors. Over 1200 members are there from across from 30 countries in this group.

(Refer Slide Time: 03:11)



Albrecht/IFPUG function point analysis

- Albrecht worked at IBM:
 - Needed a way of measuring the relative productivity of different programming languages.
 - Needed some way of measuring the size of an application without counting lines of code.
- Identified five parameters:
 - Counted occurrences of each type of functionality in order to get an indication of the size of an information system
 - It is a top-down approach

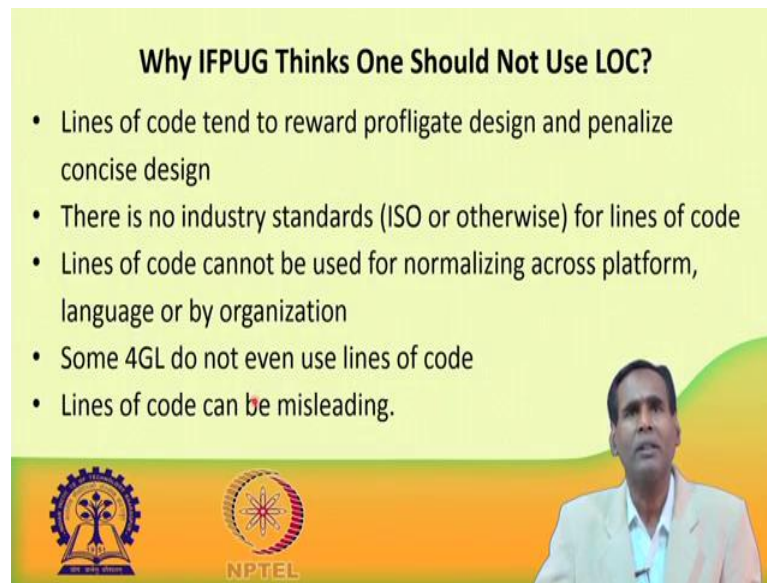
The slide features a green and yellow background. At the bottom, there are logos for a university (left) and NPTEL (center), and a portrait of a man in a light-colored suit (right).

Actually this Albrecht IFPUG function point it was coined by Albrecht's. So, while Albrecht was working at IBM he found that there is a need of measuring the relative productivity of different programming languages, so he was using different programming languages; he observed that there is a pressing need to measure the relative productivity of different programming languages. So, he also needed some way of measuring the size of an application without counting the lines of code, because the counting lines of code has several drawbacks.

So, he has identified five parameters for performing the function point analysis let us see. And he has counted the occurrence of each type of functionality in order to get an indication of the size of an information system. So, in order to estimate the size of an information system, he has counted he has first identify five parameters and counted the occurrences of each type of functionality.

Let us see and basically yesterday we have last class we have discussed about these, the different project estimation techniques such as top-down approach, bottom-up approach, etcetera. So, this Albrecht IFPUG function point analysis is based on a top-down approach.

(Refer Slide Time: 04:31)



Why IFPUG Thinks One Should Not Use LOC?

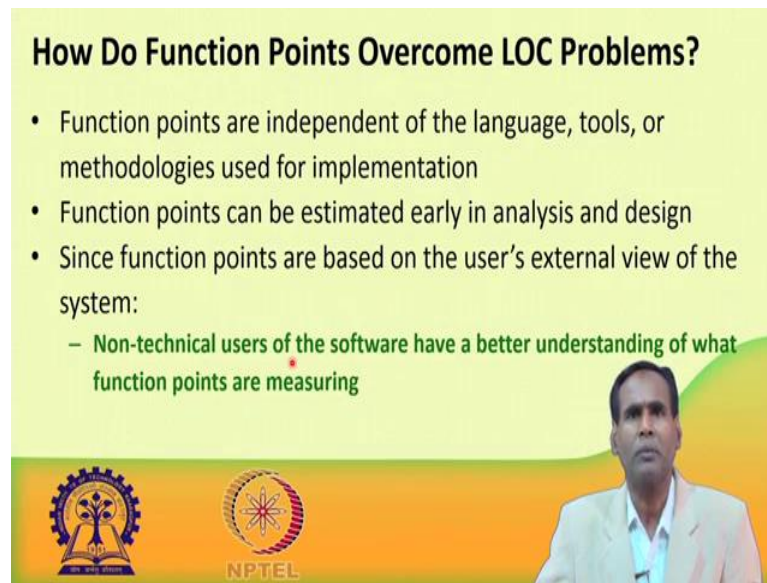
- Lines of code tend to reward profligate design and penalize concise design
- There is no industry standards (ISO or otherwise) for lines of code
- Lines of code cannot be used for normalizing across platform, language or by organization
- Some 4GL do not even use lines of code
- Lines of code can be misleading.

The slide features a video inset of a man in a light-colored suit speaking. At the bottom left, there are logos for IIT Bombay and NPTEL.

Now, let us see why IFPUG thinks that one should not use LOC rather this would use function point. We have already seen the drawbacks of LOC in the last class, till then let us quickly revise what we have seen earlier are the drawbacks of LOC. We know that lines of code it tend to reward profligate design and penalize concise design; so if your design is concise, then it will penalize you.

And there is no industry standard ISO or otherwise for lines of code, so that is another drawback we will see how function point is influenced somewhere else ISO standards. And lines of code we know that it cannot be used for normalizing across different platforms or different languages or by different organizations. Some 4GL they do not have at all the use of these lines of code. So, then lines of code will made might will not be what suitable for those languages, also lines of code it can be misleading. So, these are some of the reasons why one should not use LOC rather than he should try for a better major such as a function point.

(Refer Slide Time: 05:37)



How Do Function Points Overcome LOC Problems?

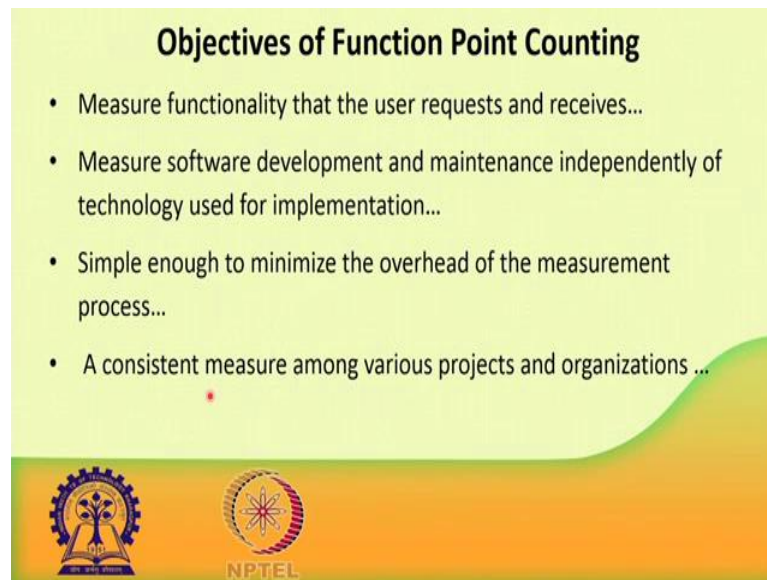
- Function points are independent of the language, tools, or methodologies used for implementation
- Function points can be estimated early in analysis and design
- Since function points are based on the user's external view of the system:
 - Non-technical users of the software have a better understanding of what function points are measuring

The slide features a green and yellow background. At the bottom, there are logos for IIT Bombay and NPTEL, along with a video overlay of a man in a light-colored suit speaking.

How do function points overcome the LOC problems, number one the function points are independent of the language, we have already seen that LOC is somehow language dependent platform dependent, but function points that independent of languages, independent of tools or independent of methodologies those are used for implementation. Then function points they can also be estimated early in analysis and design, we have seen that LOC by LOC approach cannot be used in the earliest stages of analysis design such as requirement analysis, specification and design whereas, function point can be used to estimate the what size in early stages of software development such as analysis and design.

So, since function points are based on the users external view of the system you will see that even if any layman non-technical users of the software, they can also have better understanding of what function points are measuring ok. So, this function points they are based on what the users external behave, the user's perspective, so that is why the non-technical users of the software or the projects they can also better understand what this function points are doing or their measuring.

(Refer Slide Time: 06:47)



Objectives of Function Point Counting

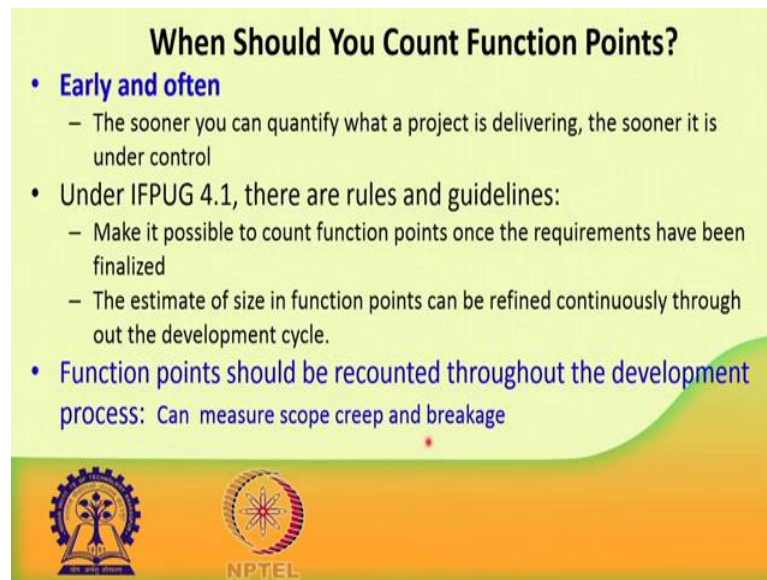
- Measure functionality that the user requests and receives...
- Measure software development and maintenance independently of technology used for implementation...
- Simple enough to minimize the overhead of the measurement process...
- A consistent measure among various projects and organizations ...

The slide features a light green background with a yellow-to-orange gradient at the bottom. On the left side of the bottom gradient, there are two logos: the IIT Bombay logo (a gear with a tree) and the NPTEL logo (a star-like symbol). The text is centered and presented in a clear, black font.

Now, let us see what are the objectives of function point counting. It measures functionality that the user requests and receives. So, it will basically measure the functionality that the user request and the user receives. Similarly, it measures the software development and maintenance independently of technology used for implementation, this we have already told you that it independent of any technology used, so that is why it can measure the software development and the maintenance activities efforts.

So, it is very simple enough to minimize the overhead of the measurement process. So, by using function point you can minimize the overhead of the measurement process. And similarly, another objective is it act has a consistent measure among various projects and organizations. So, it acts has a consistent measure among different projects and organizations.

(Refer Slide Time: 07:41)



When Should You Count Function Points?

- **Early and often**
 - The sooner you can quantify what a project is delivering, the sooner it is under control
- Under IFPUG 4.1, there are rules and guidelines:
 - Make it possible to count function points once the requirements have been finalized
 - The estimate of size in function points can be refined continuously throughout the development cycle.
- **Function points should be recounted throughout the development process: Can measure scope creep and breakage**

The slide features a green and yellow gradient background. At the bottom, there are two logos: the Indian Institute of Technology (IIT) logo on the left and the NPTEL logo on the right.

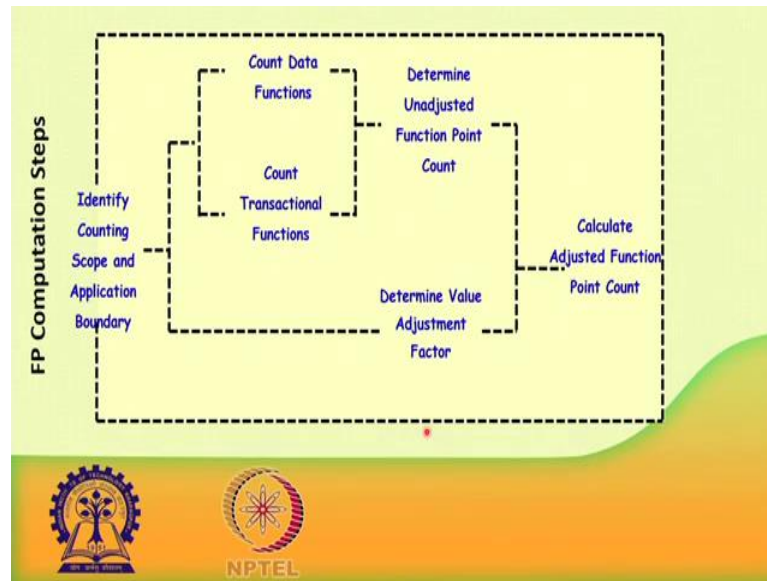
Then why should we use count function points, when should you count the function points? So, as I have already told you that you can use this thing not only coding, but before the coding starts may be requirement analysis phase and design, so that is why I saying it can be what a used in the early phases of software development and vary often it can be used. So, you see that the sooner you can quantify what a project is delivering, the sooner it is control it will under better control, you can monitor it.

Under IFPUG 4.1, there are several rules and guidelines; these rules and guidelines make it possible to count the function points once the requirements have been fixed. So, once the requirements have been frozen, then you can easily a count the function points by using the rules and guidelines provided in IFPUG 4.1.

The estimate of size in function points can be refined continuously throughout the development cycle, please see it is not just estimated only once. So, you first you have to make an initial estimate, then the estimates they can be refined continuously or you get different information updated information throughout the development lifecycle.

The function points please remember that there should be recounted throughout the development process. They can be refined throughout the development process, it hence it can measure the scope or creep and the breakage.

(Refer Slide Time: 09:10)



Let us see what are the steps involved in the function point computation. So, first you have to identify in the first step you have to identify, the counting scope and application boundary what is this scope of the counting and what is the application boundary first that has to be identified. Then you have to count two things, you have to count the data functions as well as you have to count the transactional functions. Here in data functions, like input, output those input data, output data, etcetera you have to count; and in count transactional functions what are the mainly file transactions, etcetera or the interfaces that you have to count.

Then you have to determine the function point and this is now unadjusted, you have not refined anything else. So, we this is the first time we are getting we, we call it as unadjusted function point. So, by using the what count data functions and count transaction functions, you can determine the unadjusted function point count. And then you have to calculate, because I have already told you that you may you have to refine the function points.


So, how this can be refined by using a factor called as value adjustment factor, there is a formula for finding this I will tell after few minutes. So, then we have to determine the value adjustment factor. Now, this unadjusted function point and this what adjust value adjust factor, they will be multiplied to give you the adjusted function point count. So, this is how the function point computation steps they follow.

(Refer Slide Time: 10:44)

Key Components in Function Point Analysis

Five key components (or External User Types) are identified

- External Inputs
- External Outputs
- External Inquiries
- Logical Internal Files
- External Interface Files

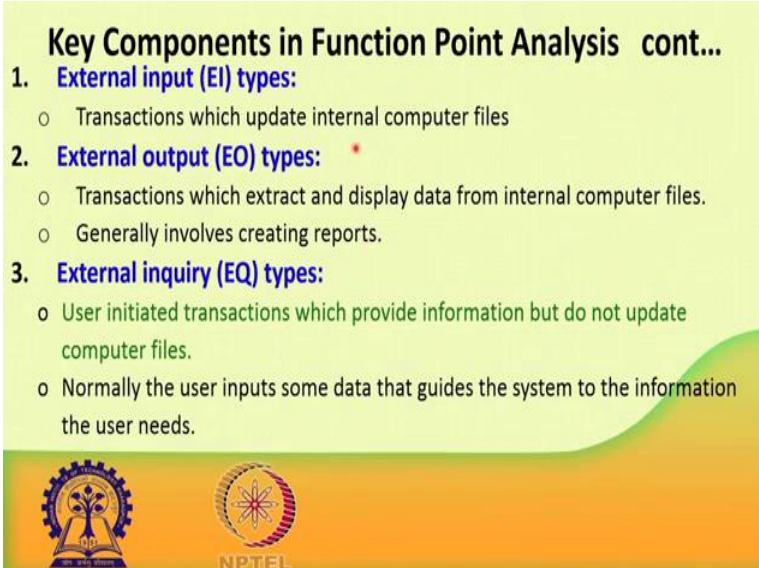


Let us see what are the key components of function point analysis, there are five key components or external use types they are identified in function point analysis that the external inputs, external outputs, external inquiries, logical internal files and external interface files. So, these are the basic five components they will be used in the function point analysis. They are also known as in Albrecht terminology, they are also known as external user types.

(Refer Slide Time: 11:11)

Key Components in Function Point Analysis cont...

- 1. External input (EI) types:**
 - Transactions which update internal computer files
- 2. External output (EO) types:**
 - Transactions which extract and display data from internal computer files.
 - Generally involves creating reports.
- 3. External inquiry (EQ) types:**
 - User initiated transactions which provide information but do not update computer files.
 - Normally the user inputs some data that guides the system to the information the user needs.



Now, let us see we will define so five components I have told, let us quickly define although there be or their usual meanings. So, external input in sort we called as EI, they represent the transactions which update the internal computer files. So, which will update the internal computer files; so, these operations these functions are known as external input types. And external output types, here these are the transactions which extract and display the data from internal computer files from the internal computer files what will do, so the transaction which will extract the information, so basically these are output types. And generally involves creating different reports of course, if you are where creating report this will be an output type of what function.

And next one is external enquiry types. So, here the user initiated actions which provide information, but do not update the computer files. So, you can see that in the external input type the transactions they update the computer files, but this is just inquiry and inquiry no updation will be there, only the user initiated the transactions which provide only the information, but do not update the computer files. Normally the user inputs what some data that guides the system to the information the user needs ok. So, here in external enquiry types the user will input some data which will guide the system to the information, to which guides the system to the information that the user needs.

(Refer Slide Time: 12:58)

Key Components in Function Point Analysis cont...

- 4. Logical interface file (LIF) types**
 - Equates roughly to a data store in systems analysis terms. Created and accessed by the target system
- 5. External interface file types (EIF)**
 - Represents data retrieved from a data store which is actually maintained by a different application.



The slide features a green and yellow background with a speaker's video feed in the bottom right corner. Logos for IIT Bombay and NPTEL are visible at the bottom left.

So, next one its logical interface or LIF files. So, this equates roughly to data store in a system analysis design terms. So, those who have studies system analysis design terms,

you must have study some data stores. So, here this LIF this equates roughly, this is equivalent to the what data stores. Created and so normally these types are created and accessed by the target system. And external interface file types this represents or these things represent, the data retrieved from a data store which is actually maintained by another different application that is why it is known as external interface file types.

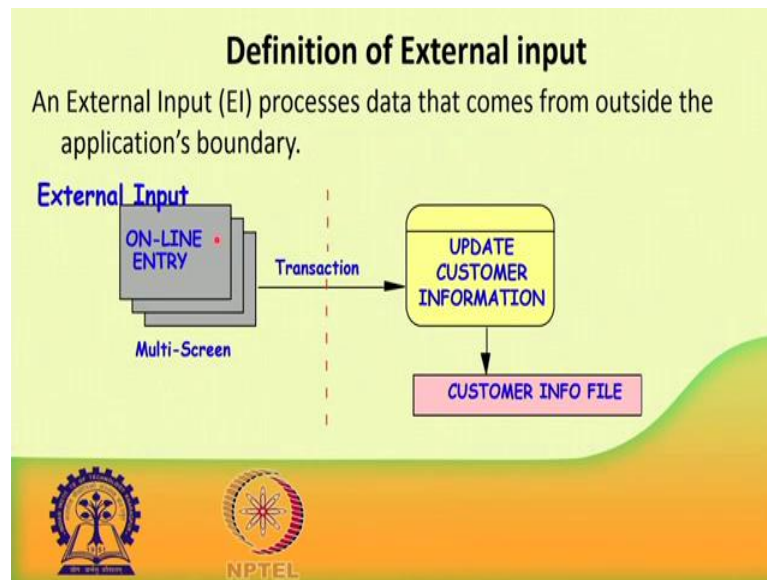
(Refer Slide Time: 13:40)

Function Point Parameters	
External Input (Inputs)	Count each unique user data or user control input type that (i) enters the external boundary of the software system being measured and (ii) adds or changes data in a logical internal file.
External Output (Outputs)	Count each unique user data or control output type that leaves the external boundary of the software system being measured.
Internal Logical File (Files)	Count each major logical group of user data or control information in the software system as a logical internal file type. Include each logical file (e.g., each logical group of data) that is generated, used, or maintained by the software system.
External Interface Files (Interfaces)	Files passed or shared between software systems should be counted as external interface file types within each system.
External Inquiry (Queries)	Count each unique input-output combination, where an input causes and generates an immediate output, as an external inquiry type.

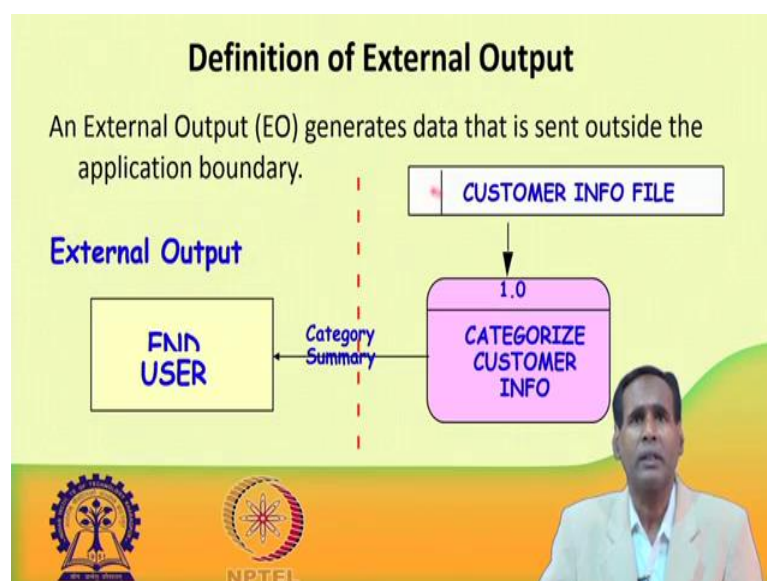
See these are the function point parameters that we have discussed five and basically what we have discussed all those things have been summarized here; more explanation is given you can see yourself.

(Refer Slide Time: 13:51)



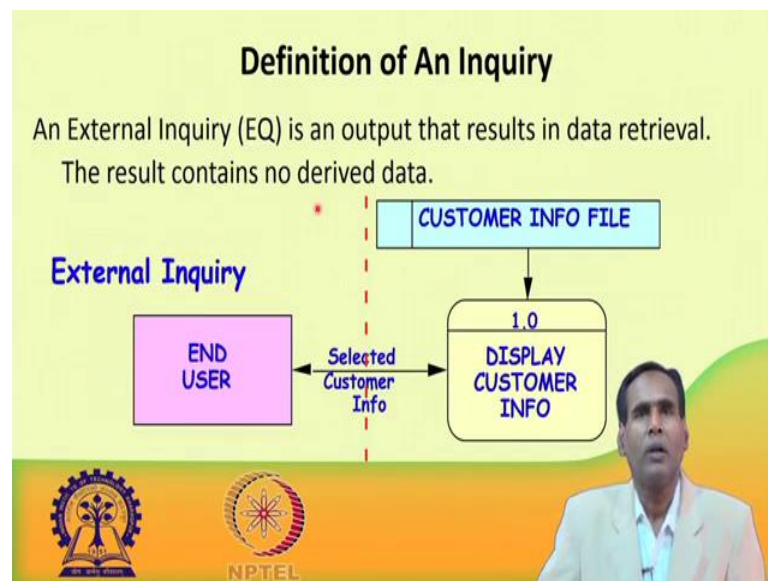
Then let us see with some examples of the definitions we have given, as I have already told you that external input it process the data that comes from outside the applications boundary from somewhere. External input it is giving and it is a so for example that some online entry and the external input is giving, and then what is there this transaction what will happen, it will update the customer information and then the where the customer information file, this file will be updated. So, here this external input it will process the data that comes from the outside application from a boundary, it will update the data.

(Refer Slide Time: 14:29)



And in the external output, it generates data that is sent outside the application boundary. Here data is coming from outside application boundary in external input, but in external output the data is sent outside the application boundary. Please, you see here that say this is the customer inform information file and then there is a process what categories customer info, this sends the information like the category summery to the outside to the end user which is an external output, so that is why these an example of external output.

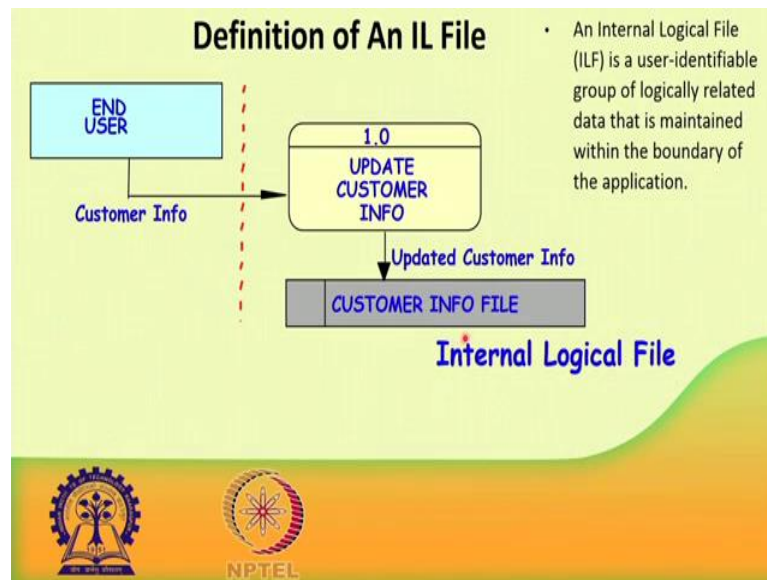
(Refer Slide Time: 15:00)



What is about an inquiry? So, on external enquiry is an output that results in data retrieval. The results contains no derive data. So, here you cannot what update anything else only that produces what some data it results in some data retrieval. So, here and this result cannot contain any derived the data.

So, here the end user asking some query inquiry and like what this inquiry might be what get the details of the customer information, then these are the easy process display customer info. This is getting the data from the customer info file and after getting data it is passing to the end user, so that is why it is an example of external enquiry.

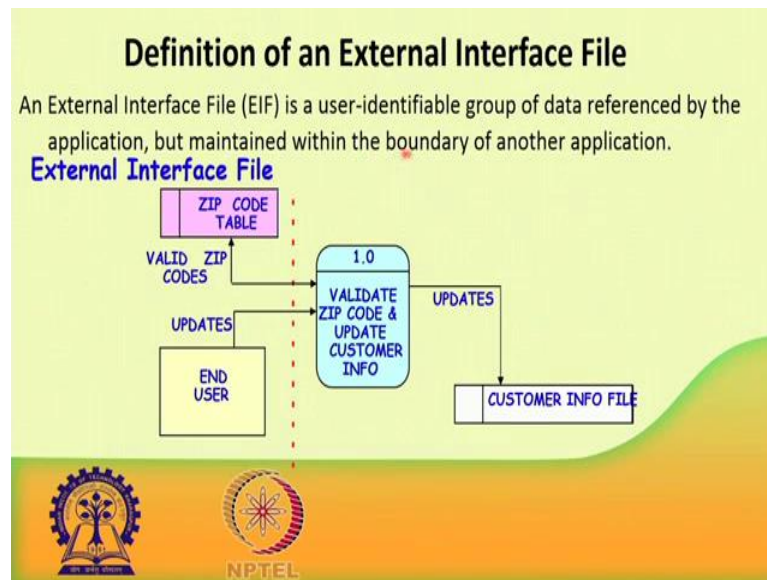
(Refer Slide Time: 15:43)



Next is a what definition of what internal logic file. So, an internal logic file is a user-identifiable group of logically related data. So, basically this is a what group of logically related data that is maintained where, within the boundary of the application; not external please remember this is not external, this is maintained within the boundary of the application, like this you have see.

The end user what send some customer info, then this process update customer info it is receives that information and accordingly it updates the customer info where in the customer info file. So, customer info file will be treated an internal logical file.

(Refer Slide Time: 16:23)



And last one is the external interface file. So, this is a user-identifiable group of data which are reference by the application. So, basically this is a group of data which are referenced by the application, but it is maintained within the boundary of another application; it is not internal it is maintained within the boundary of another application that is why the name is external interface file.

We have already seen in internal what is happening this is within the boundary of the application that is why this is internal logical file, but here you can see this file its outside the boundary so ok, so that is why this is maintained within the boundary of another application. This is outside the boundary of this present application it is somewhere outside that is why the name is external interface file. For example, here of the end user update something like validate zip code and zip code in stores somewhere else zip code table. So, for validation the message has to go here and come here, and then it will update the customer info file. So, here zip code table represents on external interface file.

(Refer Slide Time: 17:29)

Example

- **Place Purchase Order:**
 - **Input data items:**
 - Date, Supplier Number,
 - Product Code
 - Quantity Required
 - Date Required
 - **Output data items:**
 - PO Number (system generated)
 - **Entities referenced:**
 - Product, Purchase Order
 - Supplier, Purchase Order Item





Now, let us quickly take a small example we want to place a purchase order and the input data items are like date, supplier, number, product code, quantity required, date required, etcetera. Output data items are like purchase number; this will be generated by the system. Entities referenced are these types are entities are referred like product, purchase order, supplier, purchase order item, etcetera. Now, we want to estimate the size of the system, let us see how can do it.

(Refer Slide Time: 17:58)

Calculating the System Size

- For each function count:
 - Number of input data items n_i
 - Number of output data items n_o
 - Number of entities read/updated n_e
- Add these up for the whole system, giving:
 - Number of input data items N_i
 - Number of output data items N_o
 - Number of entities read/updated N_e



So, for calculating the system size you are supposed to do these things what you do, for each function count first find the number of input data items that is n_i , then the number of output data items that is n_o , then the number of entities which are read or updated that is n_e . And then after finding out the function count for all, after finding out the function count after finding out these for each function count, then you are these for the whole system. Then these are to be added, this will give you the number of input data items N_i for the whole system, number of output data items N_o for the whole system and number of entities read or updated that is N_e for the whole system.

(Refer Slide Time: 18:42)

Requirement	inputs	outputs	entity accesses
A1	10	2	4
A2	10	3	6
A3	1	25	1
A4	10	10	9
A5	4	10	5
A6	26	9	2
A7	5	11	8
A8	14	4	5
A9	22	7	4
A10	6	6	4
A11	9	9	7
A12	3	24	5
	$N_i = 120$	$N_o = 120$	$N_e = 60$

FP Counting - Example



If you will take a small example, this is what a sample example where these are the requirements ok. So, for these are the functionalities for these requirements are functionalities. So, for A 1, 10 inputs are there; for A 1, 2 outputs are there and for A 2 entity accesses is 4. Like this for there are 12 requirements find out the inputs, for all 12 requirements find out the outputs, for all the what 12 requirements then found out the entity accessed the counts, for all the entity accessed.

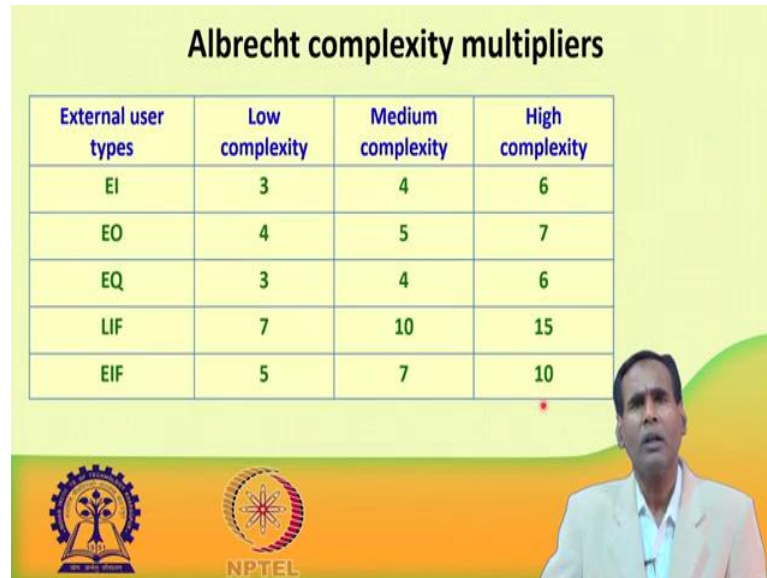
Then what you do, these are all small ends, these are these are all we can see this for each count for each function count we represent with small n and for the whole system it will be depended with capital N . So, you can see the capital N_i will be just summation of these counts that is 120; and similarly, the number of outputs of for the whole system that is N_o is equal to 120; similarly for the number of entities accessed for the whole

system will be the sum of this counts which is equal to 60. So, in this way we can find out the what if inputs, outputs, entity accessed which roughly what correspond to the size of the whole application in this manner.

(Refer Slide Time: 20:02)

Albrecht complexity multipliers

External user types	Low complexity	Medium complexity	High complexity
EI	3	4	6
EO	4	5	7
EQ	3	4	6
LIF	7	10	15
EIF	5	7	10



Now, what will see, that Albrecht has the proposed a formula for finding out the function point for different applications.

$$UFP = \sum(\text{number of elements of given type}) \times (\text{weight})$$



So, he has actually categorized the external user types into different categories such as some of the external user types might be of low complex, some of them are medium complex and some of them are high complex. Accordingly he has proposed some weights, some multipliers for these external user types depending upon the complexity.

So, like for the for external input if it is a low complex or low complex application, it is 3; for a medium complex application EI is equal to 4; for high complex application the value of EI is equal to 6. So, in this way he has proposed somewhere some multiplies for the what different external user types, this is known as Albrecht, these are known as Albrecht complexity multipliers, those multipliers I will use them in this calculation of function points.

(Refer Slide Time: 21:02)

Example - 1

The Spell-Checker accepts as input a document file and an optional personal dictionary file. The checker lists all words not contained in either of these files. The user can query the number of words processed and the number of spelling errors found at any stage during processing.



Let us quickly take a small example, how you can find out the there is a you want to develop a software for spell-checking. The spell-checker accepts as input what a document file and an optional personal directory file. Then the spell checker, it lists all words which are not content in either of these files. The user can query the number of words processed the user can query the number of words processed and the number of spelling errors which are found at any stage during the processing.

(Refer Slide Time: 21:36)



So, if you can draw if you have I hope you have known data flow diagram, so you can easily draw the data flow diagram or the context diagram for this, where you can see that user can send what this information to the process. And, the process can what do some processing, this spell checker can do some processing and give this outputs and now we want to find out the function point.

(Refer Slide Time: 22:01)


Example 1 - cont...

- 2 users inputs: document file name, personal dictionary name (average)
- 3 users outputs: error report, word count, misspelled error count (average)
- 2 users requests: # processed words?, #spelling errors? (average)
- 1 internal file: dictionary (average)
- 2 external files: document file, personal dictionary (average).

We know that, for average (medium) complexity parameters,
UFP= # inputs*4+ # outputs*5+ # inquiries*4+ # files*10+ # interfaces*7

$$= 2 \times 4 + 3 \times 5 + 2 \times 4 + 1 \times 10 + 2 \times 7$$

$$= 55$$
(Σ Parameters)



How can be done, see if you look at this you can see that the 2 inputs; what are the 2 inputs, if you will see the program the example 2 inputs, on a document file and an optional personal directory. So, these are two inputs I have shown that are 2 user inputs document filename and personal dictionary name and they can be what rated as average complex or medium complex. There are 3 user outputs, what are they error or report, word count and misspelled error count, also they will treated as average complexity or medium complexity.

The user request two things there will be 2 queries, what are the 2 queries; number of a processed words and number of spelling errors, they can be made as queries. So, they may be treated as average, so that are 2 user queries or 2 user request. There is 1 internal logical file that is the dictionary, where all those what is have stored and there is a 2 external files, what are they; the document file as well as the personal directory, personal dictionary average. See those things have so described in the problem as well as they

have been also represented in the diagram as the input and output that you can look at the diagram and see yourself.

Now, our objective is to find out the function point. So, this Albrecht has proposed a formula and according to this formula, let us see where this formula must be there, this formula ok. So, now the formula says like that the

$$\text{UFP} = \# \text{ inputs} * 4 + \# \text{ outputs} * 5 + \# \text{ inquiries} * 4 + \# \text{ files} * 10 + \# \text{ interfaces} * 7$$

So, there is a formula you can use. So, the UFP can be computed as like this that summation of this parameter whatever you are taking and into it will be the weight. So, you have to take up this parameter and then the weight there will be multiplied, there will be 'into' here and take the summation. So, if we can see that here, the UFP is computed as the here what are the parameters, the parameters are ok. So, now we can see that the number of UFP is equal to number of inputs into 4 plus number of outputs into 5 plus number of inquiries into 4 plus number of files into 10 plus number of interfaces into 7.

So, then what is happening, so here basically the these are the weights; 4, 5 and 4, 10 these are the weights, the weights are chosen for an average project, because see all the parameters are type average. So, these are average values has been taken and then you multiply them and they take the summation. So, finally you are you are observing that UFP is coming to be 55. So, this is the unadjusted function point. So, after this unadjusted function point we have to this is the initial value, then there can be what then that can be refined.

(Refer Slide Time: 25:53)



Function Point: Refinement

14 General System Characteristics are evaluated and used to compute a Value Adjustment Factor (VAF)

General System Characteristics

Data Communication	On-Line Update
Distributed Data Processing	Complex Processing
Performance Objectives	Reusability
Heavily Used Configuration	Conversion & Install Ease
Transaction Rate	Operational Ease
On-Line Data Entry	Multiple-Site Use
End-User Efficiency	Facilitate Change

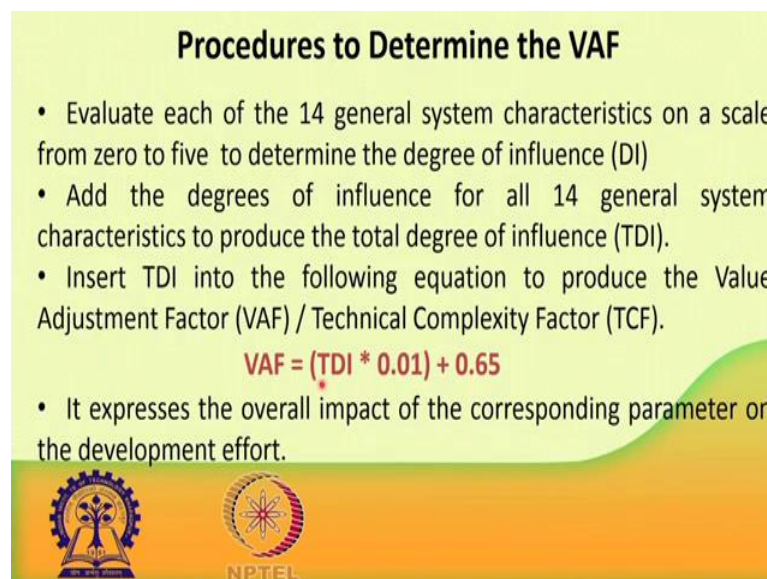
The final calculation is based upon the Unadjusted FP count X VAF





Now, let us see how this can be refined. So, Albrecht had proposed a 14 general system characteristics these are evaluated and used to compute a value adjustment factor known as VAF or sometimes it is known as TCF technically Technical Complexity Factors. So, these are the general system characteristics data come like data communication, distributed data processing, etcetera. And the final calculation is based on what finding out the unadjusted function point which we have already used this formula, then this what value adjustment factor.

(Refer Slide Time: 26:32)




Procedures to Determine the VAF

- Evaluate each of the 14 general system characteristics on a scale from zero to five to determine the degree of influence (DI)
- Add the degrees of influence for all 14 general system characteristics to produce the total degree of influence (TDI).
- Insert TDI into the following equation to produce the Value Adjustment Factor (VAF) / Technical Complexity Factor (TCF).

$$\text{VAF} = (\text{TDI} * 0.01) + 0.65$$

- It expresses the overall impact of the corresponding parameter on the development effort.



So, there is a formula for computing the value adjustment factor, this is like given like this

$$\text{VAF} = (\text{TDI} * 0.01) + 0.65$$

So, what is TDI, let us say.

(Refer Slide Time: 26:41)



So, so what is happened that these are 14 parameters they are given some values within the range 0 to 5, these values are known as degree of influence or DI; where 0 represent not present or no influence one is incidental influence and 5 with this strong influence of the parameter.

So, here they how to process calculate the value of the VAF that is written here that we have to evaluate each of the 14 general system characteristics on the scale of 0 to 5 in order to determine the value of a DI, which is known as degree of influence. So, then we have to add all the degree of influence for all the 14 general systems, this will give raise to the TDI that is total degree of influence.

Then we have to insert the value of TDI in these formula to find out the value of VAF that the value adjustment factor or complexity factor why, we want to refine the value of the function point. So, VAF is equal to TDI into 0.01 plus 0.65, this Albrecht has already found out from his research. So, this VAF or this TCF, it expresses the overall impact of the corresponding parameter on the development effort.


(Refer Slide Time: 27:50)

Procedures to Determine the VAF

The following VAF is calculated, if the degree of influence (DI) for each of the 14 GSC descriptions is 3, (i.e. 3×14):

$$\text{VAF} = (42 * 0.01) + 0.65 = 1.07$$

The Adjusted Function Point is calculated as follows:

$$\begin{aligned} \text{FP} &= \text{UFP} * \text{VAF} \\ &= 55 * 1.07 \\ &= 58.85 \end{aligned}$$


So, this is the procedure to do. Now, let us see that same example will continue where there are suppose 14 general system characteristics. And let us assume that these values almost average, it is 3, because I have already told you the value lie in between 0 to 5, suppose this is 3 then how you can compute. So, VAF formula I have already told you here that TDI and how TDI will be computed; if for 1, the value is at the degree of influence is 3 and 14 parameters so 3 into 14, so that is 42. So, value adjustment factor will be 42 into how much zero point see TDI into 0.01. So, this will be 42 into 0.01 plus a constant 0.65 like this it is 1.07.

So, finally the adjusted function point it can be calculated as follows,

$$\text{FP} = \text{UFP} * \text{VAF}$$

you have to multiply the unadjusted function point into with the value adjustment factor, so after these you will get that 58.85. So, this is the unadjusted function point for the given what spell-checker application that is the adjusted function point. Now, let us take an example here I have chosen all the what degree of influence for every what that parameter, it is of what 3 uniform.


(Refer Slide Time: 29:10)

Procedures to Determine the VAF cont ...

14 general system characteristics with different DIs:

1. Data Communication	3	8. End-user Efficiency	3
2. Distributed Data Processing	0	9. Complex Computations	0
3. Performance Criteria	4	10. Reusability	3
4. Heavily Utilized Hardware	0	11. Ease of Installation	3
5. High Transaction Rates	3	12. Ease of Operation	5
6. Online Data Entry	3	13. Portability	3
7. Online Updating	3	14. Maintainability	3

Total Degree of Influence (TDI)=36




But if it is difficult different like the 14 general system characteristics with the different DIs, like for something 3, something 0, something 4 and something 5; so it is ranging in between it is ranging in between 0 to 5. So, then the total degree of influence is the summation of these, this is 36.

(Refer Slide Time: 29:26)

Example 1 cont...

So, $VAF = (36 * 0.01) + 0.65 = 1.01$

The Adjusted Function Point is calculated as follows:

$$FP = UFP * VAF$$
$$= 55 * 1.01$$
$$= 55.55$$


And you can say that so now, this value adjustment factor will be how much 36 will put in the above formula that will give raise to 1.01 and once the VAF is calculated, you can easily calculate the adjusted function of the adjusted function point will be equal to

multiplication of UFP – Unadjusted Function Point into the value adjustment factor, so this is coming to the 55.55. So, in this you have taken what are the two cases where the values of GSC – General System Characteristics are same, in another case the value of the general system characteristics are different, how to find out the adjusted function points ok.


(Refer Slide Time: 30:02)

Example 3

A Payroll application has:

1. Transaction to input, amend and delete employee details – an EI that is rated of medium complexity
2. A transaction that calculates pay details from timesheet data that is input – an EI of high complexity
3. A transaction of medium complexity that prints out pay-to-date details for each employee – an EO of medium complexity
4. A file of payroll details for each employee – assessed as of medium complexity LIF
5. A personnel file maintained by another system is accessed for name and address details – a simple EIF

What would be the FP counts for these?



So, there is another example is a very small example, this you can see yourself.


(Refer Slide Time: 30:07)

FP counts

1. Medium EI	= 4 FPs
2. High complexity EI	= 6 FPs
3. Medium complexity EO	= 5 FPs
4. Medium complexity LIF	= 10 FPs
5. Simple EIF	= 5 FPs
Total (UFP)	= 30 FPs

External user types	Low complexity	Medium complexity	High complexity
EI	3	4	6
EO	4	5	7
EQ	3	4	6
LIF	7	10	15
EIF	5	7	10

If previous projects delivered 5 FPs a day, implementing the above should take $30/5 = 6$ days




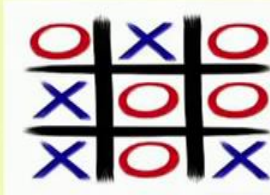
Where and reading that example you can find out, how many inputs are there and here I have taken there are two inputs; one is of medium, another is high. And there is a one what output that is of medium complexity, one internal file that is medium complexity, and one simple interface file that is simple, so add all those things. So, will get total UFP is equal to how much 30 FPs.

So, if a previous projects, if you have already develops similar kind of projects and the if the previous projects delivered 5 function points for a day, then you can easily calculate that you have to take 6 days 35 by 30 by 5, so that 6 days you have to take in order to develop this project.

(Refer Slide Time: 30:52)

Exercise 1: Tic-Tac-Toe Computer Game

- As soon as either of the human player or the computer wins,
 - A message announcing the winner should be displayed.
- If neither player manages to get three consecutive marks along a straight line,
 - And all the squares on the board are filled up,
 - Then the game is drawn.
- The computer always tries to win a game.






So, other exercises are given, so that you can practice.

(Refer Slide Time: 30:56)

Exercise 2

- It is needed to develop an Alumni Repository software for IIM, Ranchi. The software will extract the details of students from the existing academic software of IIM, Ranchi. It will provide an online display of the Alumni names. The details of the Alumni can be entered by any one by double clicking on the Alumni name. The details of Alumni would be stored in a file. It should be possible to print out a report detailing all alumni.

Determine UFP





For each exercise find out what is the unadjusted function point, then what are the what VAFs, multiply all those things.

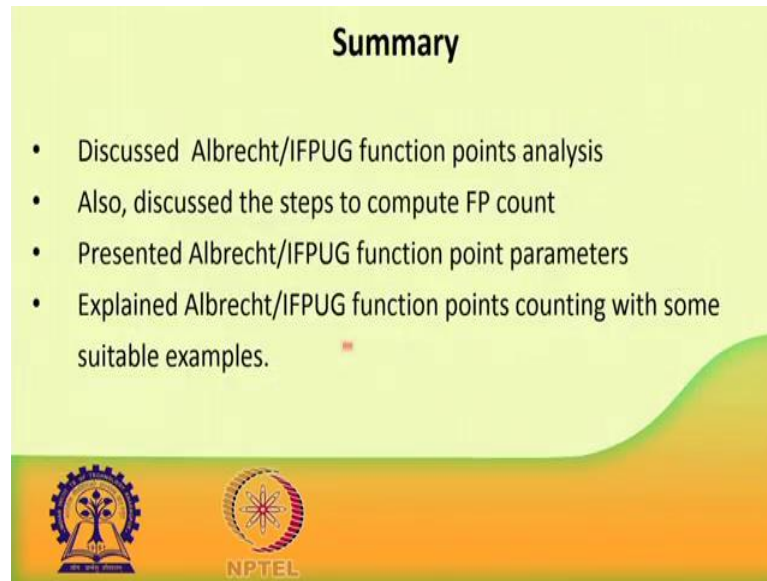
(Refer Slide Time: 31:02)

Exercise 3: Supermarket Prize Scheme

- A supermarket needs to develop the following software to encourage regular customers.
- TO register, a customer needs to supply his/her residence address, telephone number, and the driving license number.
- Each customer who registers for this scheme is assigned a unique customer number (CN) by the computer.
- Based on the generated CN, a clerk manually prepares a customer identity card after getting the market manager's signature on it.
- A customer can present his customer identity card to the check out staff when he makes any purchase. In this case, the value of his purchase is credited against his CN.
- At the end of each year, the supermarket intends to award surprise gifts to 10 customers who make the highest total purchase over the year. Also, it intends to award a 22 caret gold coin to every customer whose purchase exceeded Rs.10,000/-.
- The entries against the CN are reset on the last day of every year after the prize winners' lists are generated.



(Refer Slide Time: 31:06)



Summary

- Discussed Albrecht/IFPUG function points analysis
- Also, discussed the steps to compute FP count
- Presented Albrecht/IFPUG function point parameters
- Explained Albrecht/IFPUG function points counting with some suitable examples.

The slide features a light green background with a decorative orange and yellow wave at the bottom. On the left side of the wave, there are two logos: the Indian Institute of Technology (IIT) logo and the NPTEL logo.

Then you can get the total function point using Albrecht, what function point method. So, finally, here we have discussed Albrecht IFPUG function point analysis; we have taken some examples related this. And in the next class also, I will take few more examples for this, then we have to discuss this steps to compute the FP count. Then we have presented also the function point different parameters, parameters I have told and then we have explained the IFPUG function point counting with some suitable examples.

(Refer Slide Time: 31:35)



References :

1. B. Hughes, M. Cotterell, R. Mall, *Software Project Management*, Fifth Edition, McGraw Hill Education (India) Pvt. Ltd., 2018.
2. R. Mall, *Fundamentals of Software Engineering*, Fifth Edition, PHI Learning Pvt. Ltd., 2018.

The slide features a light green background with a decorative orange and yellow wave at the bottom. On the left side of the wave, there are two logos: the Indian Institute of Technology (IIT) logo and the NPTEL logo.

We have taken mainly it is a concepts on these two books; we can have along to look into this.

Thank you very much.