

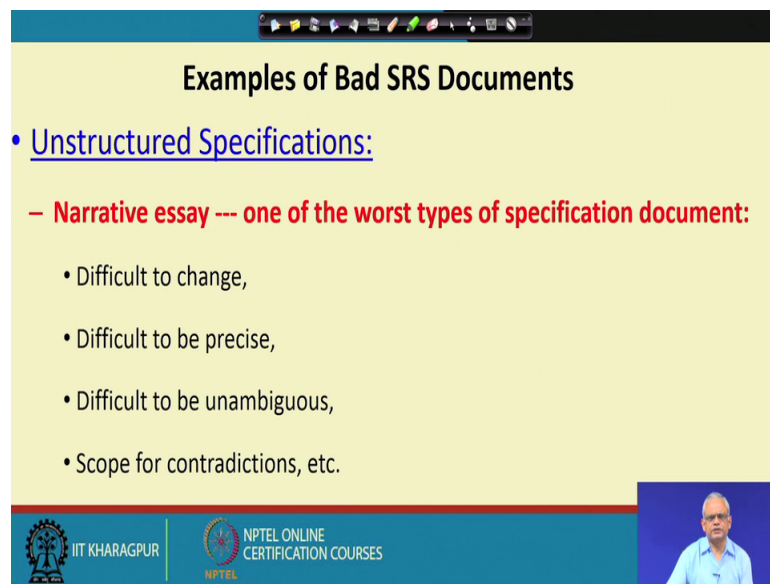
Software Engineering
Prof. Rajib Mall
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 18
Representation of complex programming logic

Welcome to this lecture. So, for in the over the last few lectures we had seen that requirement specification is a very important task, during software development. We had looked at how requirements gathering analysis and specification can be done? We will look at the standard specification template that is I triple E 830.

In this lecture we will look at few other aspects of the requirement specification before we conclude this topic. Let us look at some of the desirable properties of requirement specification document, and what should be should not be there how to write it and how not to write it. Let us proceed with that objective.

(Refer Slide Time: 01:24)



Examples of Bad SRS Documents

- Unstructured Specifications:
 - **Narrative essay --- one of the worst types of specification document:**
 - Difficult to change,
 - Difficult to be precise,
 - Difficult to be unambiguous,
 - Scope for contradictions, etc.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Let us look at some examples of bad SRS document. The worst form of document possibly is just a essay it is unstructured specification just a description of what is required. This is one of the worst types of specification, because it is it has lot of problems.

Typically, if you want the customer to write about their requirements, they would write in the form of a narrative essay. Here, various types of functional nonfunctional requirements constraints are all mixed up. Let us look at what would be the problems? If somebody just uses this kind of a document and narrative essay as a specification document. One of the most difficult problems that will be faced is that it is difficult to change, because various issues the functional requirement, non functional requirements, constraints etcetera are all mixed up.

Let us say we want to change the functionality, during the development or maybe after development. It would become very difficult, because this functionality would not be described just at one place. It, would be all mixed up over various phases and to find out where all to change maybe we can change few, maybe we can you know due to oversight leave some of those. So, changing it is a very tedious task and unless you are careful the changes would not be proper. It is difficult to be precise, because it is all mixed up just a narrative essay and we write the issues as it occurs to the mind. And therefore, we need to refer to various places previously you had described that these does these and here you describe later we will describe these etcetera.

So, it is all interspersed and it becomes very descript. It is difficult to become unambiguous when there is a large set of sentences all mixed up, it can happen that if we read some sentences in isolation for some functionality, it can give a ambiguous meaning. There are scope for contradiction because as we write a large document forget what was written earlier and it is easy to contradict.

If, such problems exist in a document, then it becomes very difficult for the developers for the testers to carry out their task. And, also please remember that requirements change a great deal during development and after development. During the development on the average about 40 percent of the requirements are expected to change.

And, even after the development the requirements continue to change, if it is a unstructured specification, it becomes very combustion consumes large effort cost to change the document and yet there will be several mistakes and problems. And, that is one of the reason that hardly there is any commercial organization, who would write their requirement specification as a narrative essay.

(Refer Slide Time: 05:51)

The slide is titled "Examples of Bad SRS Documents". It contains two main bullet points: "Noise" and "Silence". Under "Noise", there is a sub-bullet: "Presence of text containing information irrelevant to the problem." Under "Silence", there is a sub-bullet: "Aspects important to proper solution of the problem are omitted." The slide footer includes the IIT Kharagpur logo and the NPTEL Online Certification Courses logo. A small video inset of a speaker is visible in the bottom right corner.

Examples of Bad SRS Documents

- Noise:
 - Presence of text containing information irrelevant to the problem.
- Silence:
 - Aspects important to proper solution of the problem are omitted.

Another, very undesirable aspect of a bad SRS document is noise. Noise is something which is unrelated to the problem. Basically, when the document is written in a very (Refer Time: 06:15) style there is a tendency to put information, which is not really part of the problem being solved. And, this makes it difficult for the developers and testers to figure out that which sentences to ignore and which sentences to take. Another, very undesirable aspect of a requirement document is silence, which is basically omitting some important issues maybe functional requirement, non-functional requirements, constant interfaces and so on.

(Refer Slide Time: 07:01)

The slide is titled "Examples of Bad SRS Documents". It contains two main bullet points: "Overspecification" and "Contradictions". Under "Overspecification", there are three sub-bullets: "Addressing 'how to' aspects", "For example, 'Library member names should be stored in a sorted descending order'", and "Overspecification restricts the solution space for the designer." Under "Contradictions", there is one sub-bullet: "Contradictions might arise", followed by a further sub-bullet: "if the same thing described at several places to mean different thin". The slide footer includes the IIT Kharagpur logo and the NPTEL Online Certification Courses logo. A small video inset of a speaker is visible in the bottom right corner.

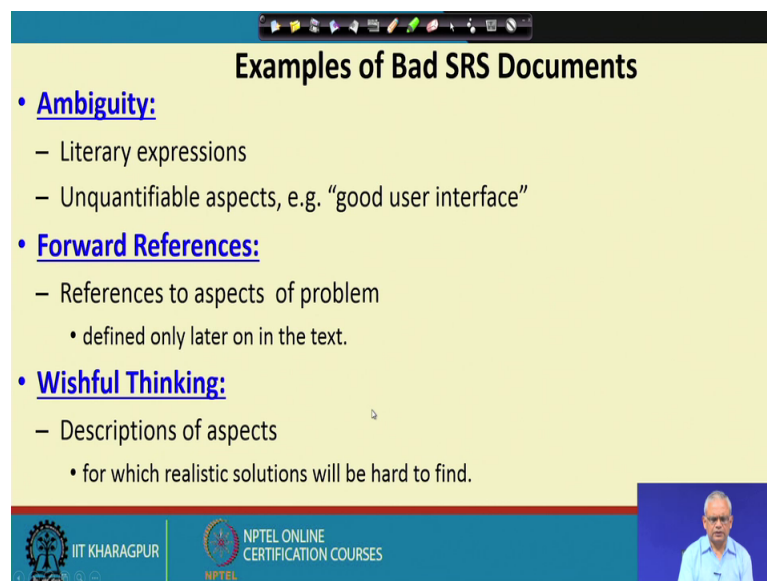
Examples of Bad SRS Documents

- Overspecification:
 - Addressing "how to" aspects
 - For example, "Library member names should be stored in a sorted descending order"
 - Overspecification restricts the solution space for the designer.
- Contradictions:
 - Contradictions might arise
 - if the same thing described at several places to mean different thin

Over specification is another very undesirable characteristic, by over specification we mean addressing “how to” aspects? For example, let us say we say how to store the library member names that they should be stored, in a sorted order, sorted descending order in a file etcetera. And, then once we write it in the SRS document it becomes binding and the designers have very little flexibility, if we have already mentioned here. So, this stored in a sorted descending order in a file then they cannot use a database management system. And, also they have to keep it in a descending order even if for some other task they may need in ascending order.

So, the how to aspect should be carefully avoided, while writing the SRS document. Another, problem with the bad SRS document is contradictions specific specifically this kind of problem occur in narrative essay types specification, where contradictions between various parts of the document exist and this is to be consciously avoided.

(Refer Slide Time: 08:46)



The slide is titled "Examples of Bad SRS Documents" and lists three categories of bad SRS documents:

- **Ambiguity:**
 - Literary expressions
 - Unquantifiable aspects, e.g. “good user interface”
- **Forward References:**
 - References to aspects of problem
 - defined only later on in the text.
- **Wishful Thinking:**
 - Descriptions of aspects
 - for which realistic solutions will be hard to find.

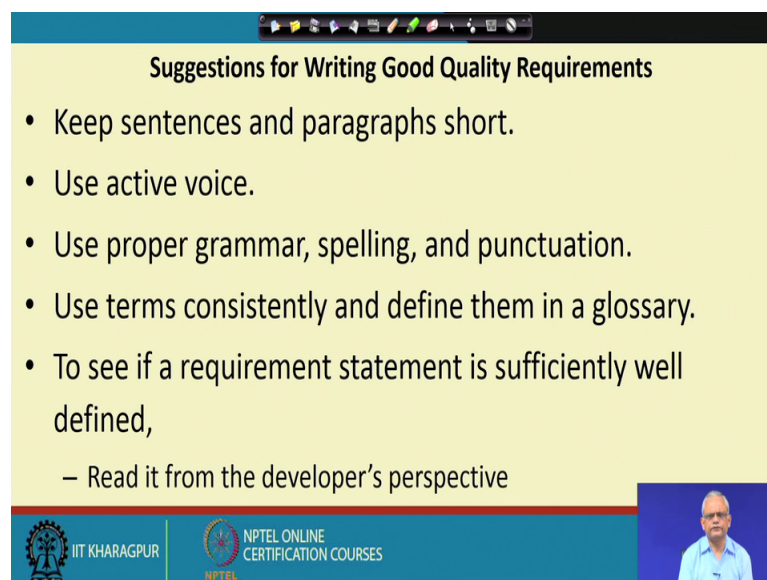
The slide also features logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES, and a small video inset of a speaker in the bottom right corner.

Ambiguity, if there is anything mention in the document which cannot be quantified, becomes very difficult to test. For example, if we write in the document that it should have a good user interface. Then the tester the developers will all be in confusion what is meant by good user interface? How does somebody test and say that it is a good user interface? Another problem with the SRS document is forward reference as read the document we only know what has been already written? And, if we write in the document that referred to later section, then somebody would have to turn pages read

that and again come back and if that happens many times becomes really difficult for somebody to read it.

One more problem is wishful thinking, if the person writing the specification document is not careful he might agree to some functional requirements, which cannot be implemented, realistic solutions to these, cannot be obtained. And therefore, it is important that the analyst as a person who is writing the document should have a idea that how to solve? The functionality that is required or a non-functional requirement, commitment to unrealistic requirements, becomes very difficult and later would have to again change the requirement after the developers have tried implementing the.

(Refer Slide Time: 10:49)



Suggestions for Writing Good Quality Requirements

- Keep sentences and paragraphs short.
- Use active voice.
- Use proper grammar, spelling, and punctuation.
- Use terms consistently and define them in a glossary.
- To see if a requirement statement is sufficiently well defined,
 - Read it from the developer's perspective

The slide includes logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES, and a small video inset of a man in a light blue shirt.

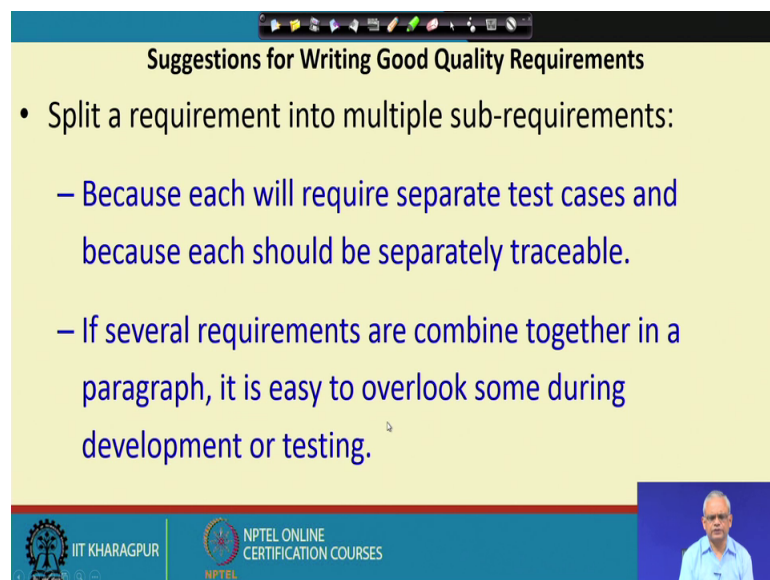
Let us just discuss a few suggestions on how to write good quality requirement document? One thing about the requirement document is that there are many stakeholders. Who would like to read the document for various purposes? And therefore, readability of the document is an important concern, for the for anybody to read the document sentences should be short, paragraph should be short, a sentence which is 20 lines in a long would become very difficult to read the sentences should be simple and typically one line and so on.

Similarly, the paragraph should be small paragraphs, active voice use of active voice, use of proper grammar spelling and punctuation, the terms should be consistently used and

should be defined in a glossary. So, that somebody who does not understand a term as reading through the document one can refer to the glossary.

And, also before completing writing the document, the reviewer, the developer, the document, analyst, and also the reviewers so, read it from the perspective of the user and also from the perspective of the developer. From the perspective of the user understandability readability is very important and from the perspective of the developer sufficient details should be there and from the perspective of the tester it should be able to write test cases it should be possible to write test cases to test the requirement.

(Refer Slide Time: 12:56)



The slide is titled "Suggestions for Writing Good Quality Requirements" and contains the following content:

- Split a requirement into multiple sub-requirements:
 - Because each will require separate test cases and because each should be separately traceable.
 - If several requirements are combine together in a paragraph, it is easy to overlook some during development or testing.

The slide footer includes the IIT Kharagpur logo, the NPTEL logo, and the text "NPTEL ONLINE CERTIFICATION COURSES". A small video inset of a speaker is visible in the bottom right corner.

Sometimes the requirements are complex, but in those cases it is important to split the requirement into sub requirements. One of the major reason for doing that is that traceability. Since, it is a complex large requirement, it is implementation will require many design elements, many modulus or functions in the code, but if we have split into sub requirements.

Then, each sub requirement can be traced into a few design elements and a few functions modules in the code. Also, it becomes easier for the tester to write test cases, because for every sub requirement the tester can write test cases. Otherwise if it a huge complex requirement the tester may overlook some of the aspects to test.

(Refer Slide Time: 14:10)

SRS Review

- Review done by the Developers along with the user representatives.
- To verify that SRS confirms to the actual user requirements
- To detect defects early and correct them.
- Review typically done using standard inspection process:
 - Checklists.

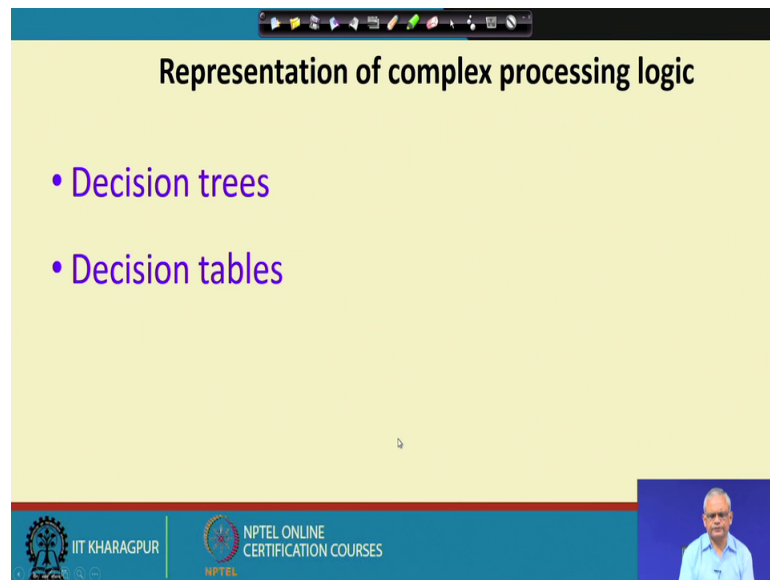
```
graph TD; needs((needs)) --> Gathering[Gathering]; Gathering --> Analysis[Analysis]; Analysis --> Specification[Specification]; Specification --> Review[Review]; Review --> SRS[SRS Document]; Analysis --> Gathering; Specification --> Analysis;
```

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

In the requirement specification process, we said that after the requirements document is written the specification is done that is first is requirement gathering, analysis of the requirements in an iterative manner. And then finally, trying to write the specification document using a standard template, again we might have to do some gathering and analysis.

But, after the requirement document is completely written it is submitted for review. Review team typically consists of the customer, the developer, tester, basically all stakeholders. The customer checks whether the requirements confirm to their actual requirements. The developers check if any details are missing, the tester sees if every requirement can be tested meaningfully. And, the defects that are detected by the reviewers can be corrected. Typically the review is done by developing a checklist and it is checked that every element in the checklist is satisfied by the SRS document.

(Refer Slide Time: 15:49)



Representation of complex processing logic

- Decision trees
- Decision tables

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Let us look at if the processing logic is complex how do a meaningfully represent it? The processing logic is in terms of if certain conditions are satisfied, then do something and check for further conditions and so on.

If, this processing logic is very complex, just text description becomes very difficult to understand and for the developers to be able to meaning fully develop solutions to that and also for the testers to write test cases. And, also in a text form if we write a very complex processing logic it is likely that some of the conditions can be overlooked. And, that is the reason why some semi formal techniques are used for representing complex processing logic, 2 techniques are very popular decision trees and decision tables.

(Refer Slide Time: 17:04)

Decision Trees

- Decision trees:
 - Edges of a decision tree represent conditions
 - Leaf nodes represent actions to be performed.
- A decision tree gives a graphic view of:
 - Logic involved in decision making
 - Corresponding actions taken.

```
graph LR;
  A[User input] --> B[New member];
  A --> C[Repeat];
  A --> D[Cancel];
  A --> E[Invalid option];
  B --> B1[Get details];
  B --> B2[Create record];
  B --> B3[Print bills];
  C --> C1[Get Details];
  C --> C2[Update record];
  C --> C3[Print bills];
  D --> D1[Get Details];
  D --> D2[Print Cheque];
  D --> D3[Delete record];
  E --> E1[Print error message];
```

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Both the techniques are not specific to this domain there used across various other disciplines, but then let us see how it can be used to represent complex processing logic. These are very simple concepts in decision tree as an example is given here along the edge, it is decision is represented. So, junction here represents some decision to be checked and depending on the decision some actions are taken. And, this gives it a tree like appearance that every node of the tree some check is made and conditions. And, depending on which condition is satisfied that specific branch is taken. And, the leaf nodes here on the tree this can be a multilevel tree and each node of the sorry the leaf node of the tree to write the actions to be taken.

So, if it satisfies certain conditions on the path, then these actions should be taken. One of the advantage of the decision tree is that it gives a graphic view of the logic that is involved and the actions that will be taken, if certain logic conditions hold and we can easily detect if certain conditions are missed.

(Refer Slide Time: 18:48)

Example: LMS

- A Library Membership automation Software (LMS) should support the following three options:
 - New member,
 - Renewal,
 - Cancel membership.

```
graph LR;
  UI[User input] --> NM[New member];
  UI --> R[Renewal];
  UI --> C[Cancel];
  UI --> IO[Invalid option];
  NM --> NM1[Get details];
  NM --> NM2[Create record];
  NM --> NM3[Print bills];
  R --> R1[Get Details];
  R --> R2[Update record];
  R --> R3[Print bills];
  C --> C1[Get Details];
  C --> C2[Print Cheque];
  C --> C3[Delete record];
  IO --> IO1[Print error message];
```

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Let us take an example. So, very simple concept I am sure that everybody would be able to draw this kind of diagram, but just to illustrate how to draw? Let us take a simple problem. Let us say we are trying to develop a library membership automation software and as the user selects create member there are 3 options which come out. New member membership renewal and cancel membership so, let me first describe the problem and then we will see how to do the solution?

So, the problem is that this is a requirement create member and as the create member option is chosen there are 3 sub options, which are displayed. Create new member, membership renewal, and cancel membership.

(Refer Slide Time: 19:56)

Example: LMS

- When the new member option is selected,
 - The software asks details about the member:
 - name,
 - address,
 - phone number, etc.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

If, the new member is selected then you just asks the some details about the member like name, address, phone number etcetera and then creates a record and then prints the bill. So, during in the menu if we select the new member the actions taken is get the details of the member create the record and print the bill.

(Refer Slide Time: 20:35)

Example (cont.)

- If proper information is entered,
 - A membership record for the member is created
 - A bill is printed for the annual membership charge plus the security deposit payable.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, that is what it says that once the details are obtained membership record is created and the bill is printed for the membership charge.

(Refer Slide Time: 20:46)

Example_(cont.)

- If the **renewal** option is chosen,
 - LMS asks the member's name and his membership number
 - checks whether he is a valid member.
 - If the name represents a valid member,
 - the membership expiry date is updated and the annual membership bill is printed,
 - otherwise an error message is displayed.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

If the renewal option is chosen, then and then the name represents are valid member, then the expiry date is updated and the bill for the renewal is printed.

(Refer Slide Time: 21:07)

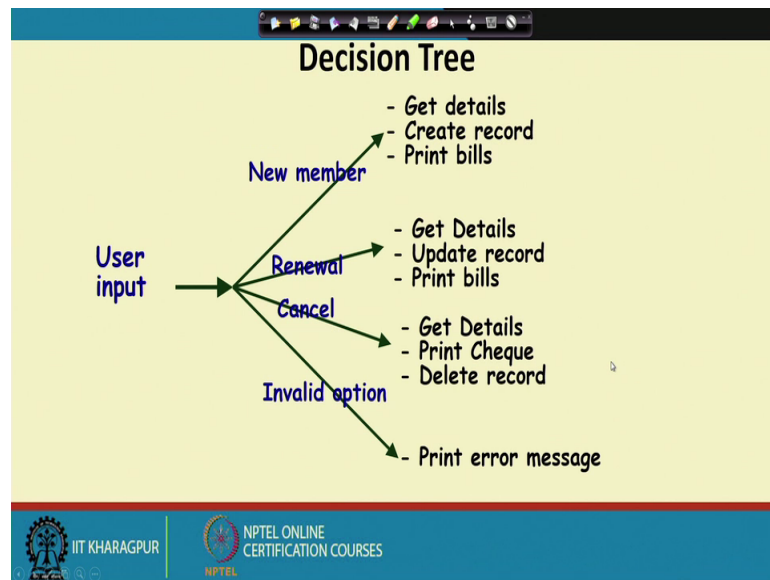
Example_(cont.)

- If the **cancel membership** option is selected and the name of a valid member is entered,
 - The membership is cancelled,
 - A cheque for the balance amount due to the member is printed
 - The membership record is deleted.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

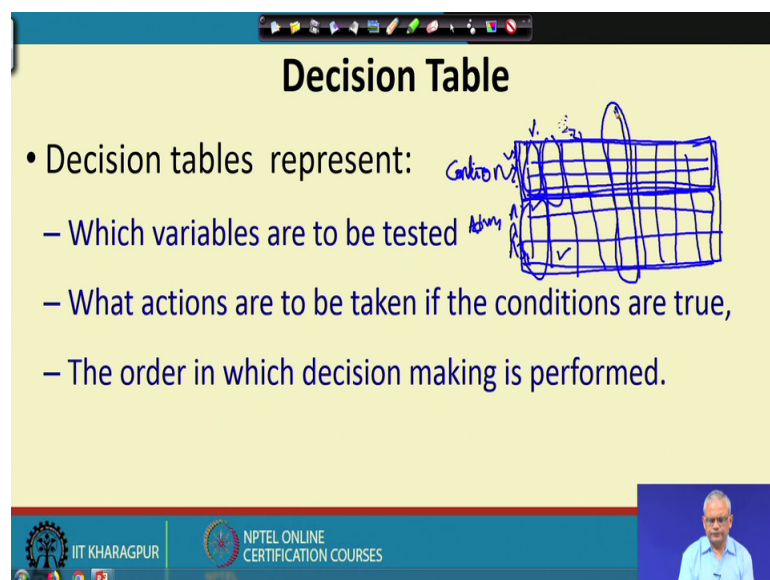
Similarly, if the cancel membership option is chosen, then the details like name etcetera are obtained and the balance amount is printed in the on a cheque and then the membership is deleted.

(Refer Slide Time: 21:31)



If any other option is chosen, then there is a printer error message. So, this is the representation of the simple this not every complex processing logic, but then just to illustrate how to write the decision tree just explain with respect to a simple example, but then for complex processing logic there can be many decisions at each level and this will become a multilayer tree.

(Refer Slide Time: 22:00)



Another, alternative is the decision table, the decision table represents an alternative way the same thing here we represent, which variables are to be tested to determine which

condition is satisfied and then what actions to be taken if the conditions are true and the order of the decision making. So, it is a table as the name says the decision table is a table and the first few rows these actually define the conditions, the conditions and here each variable here V 1 V 2 etcetera what are the values for that sorry. So, V 1 V 1 V 2 V 1 V 2 etcetera and these setup combinations of the variable values V 1 V 2 etcetera these define a condition and the lower part of the table these define the actions.

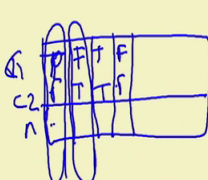
So, actions can be A 1 A 2 A 3 etcetera. So, if these sets of conditions are satisfied that is some variable has some specific values, then maybe you will say that action A 1 will be taken. And, if the conditions of some other set of values, then the let us say action A 3 will be taken.


So, here if you notice that the upper part of the table this represents the conditions are specified in terms of certain variables having certain values. So, V 1 V 2 V 3 etcetera these are some variables and whether they have some values it can be Boolean variables, which can be let say on off etcetera. These can be integer variables also. And, the lower part of the table represents the actions if the variables have certain values, then the corresponding action is taken and each of these columns here is called as a rule.

(Refer Slide Time: 25:26)


Decision Table

- A decision table shows in a tabular form:
 - Processing logic and corresponding actions
- Upper rows of the table specify:
 - The variables or conditions to be evaluated
- Lower rows specify:
 - The actions to be taken when the corresponding conditions are satisfied.






IIT KHARAGPUR



NPTEL ONLINE
CERTIFICATION COURSES



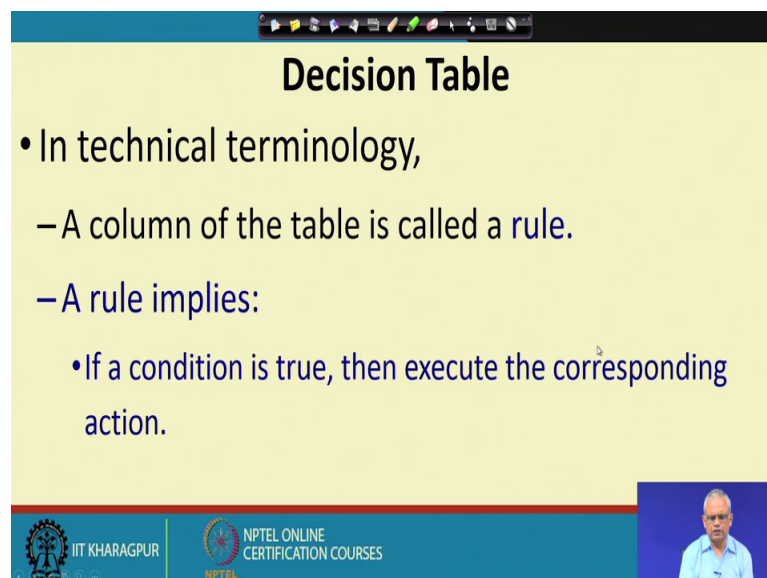
So, this is a tabular representation on like the tree like representation in decision tree.. Typically these becomes very compact representation the tree is 10 to become large for very complex processing logic, but here it is a compact representation in the upper rows

if the table, what are the variables are conditions and what will be the values are mentioned? And, typically all possible combinations of the conditions and values should be written. So, for example, if a condition V 1 or let me just say C 1 and C 2 are 2 conditions, then we have to check for C 1 is let us say true on and off or C 1 is let me write true and C 2 is let us say false.

Similarly, C 1 is false C 2 is true; C 1 is true C 2 is true and C 1 is false C 2 is false. All possible combinations of the condition should be written and this forms the rule and what is the action to be taken? If, this is true and this is false those are specified here. And, each of these becomes a rule these are Boolean conditions, but then if there is a variable, then we have to consider all possible combinations of the variables.

You can try representing this in a decision tree one of the difficulty in a decision tree is that become difficult to check whether all possible combinations of conditions have been taken care or not. And, also the trees are not very compact whatever we represent in the form of a slightly larger table battery a tree may spillover couple of phases. And, also the table will see later that it becomes very easy to design test cases for each rule, we write each rule here becomes one test case.

(Refer Slide Time: 28:14)



Decision Table

- In technical terminology,
 - A column of the table is called a rule.
 - A rule implies:
 - If a condition is true, then execute the corresponding action.

Each column of the table is called as a rule and each rule implies that if some conditions are true then execute the corresponding action.

(Refer Slide Time: 28:24)

Conditions				
Valid selection	NO	YES	YES	YES
New member	--	YES	NO	NO
Renewal	--	NO	YES	NO
Cancellation	--	NO	NO	YES

Actions				
Display error message	--	--	--	--
Ask member's name etc.	--	--	--	--
Build customer record	--	--	--	--
Generate bill	--	--	--	--
Ask membership details	--	--	--	--
Update expiry date	--	--	--	--
Print cheque	--	--	--	--
Delete record	--	--	--	--

And, for the same library system this is the decision table representation. So, if it is the conditions are whether it is a valid selection new member, it is a renewal or cancellation. If it is a it is not a valid selection, then it is immaterial it is do not care for all these 3. And, then the specific action to be taken is to print a error message display error message not completed the lower part should have written here display error message. If, it is a valid selection and then it is a new member then you should print ask for the members name, build customer record, generate bill.

Similarly, for if it is a valid selection and renewal, then you have to update the expiry date print cheque sorry not print cheque. And, then update the expiry date and then display message, similarly for if it is the cancellation then update, delete record and print cheque. As, you can see that decision table decision tree are very simple representation of complex processing logic, it has the advantage that if anything missed during specification can be easily checked here. It is easier to understand the decision tree or a decision table and also it helps the developers and the testers.

Now, we are running out of time we will stop here and continue in the next lecture.

Thank you.