Design and Implementation of Human – Computer Interfaces

Prof. Dr. Samit Bhattacharya
Department of Computer Science & Engineering
Indian Institute of Technology, Guwahati

Lecture: 33
Black-Box Testing Case Study

Hello and welcome to the NPTEL MOOCS course on design and implementation of human computer interfaces. We will continue our discussion of lecture 28 that is the previous lecture on black box testing in this lecture which is the continuation of the previous lecture we are going to have a look at a case study of a practical system and the corresponding black box testing of that system in fact the system that we are going to talk about in this case study has already been introduced earlier.

That is a student activity monitoring system and alert generation system which we have mentioned in a couple of previous case study lectures we will continue that same system and here in this lecture we will see how to create a black box testing report for the system. Remember that in this course we are learning a stage wise development of interactive system at the end of each stage we are supposed to generate a document.

Same is true for this code testing stage where we are going to generate a testing report. We have already seen such a report for a system in case of review based testing in this lecture we are going to see such a report for our system for black box testing.
(Video Starts: 02:16)

So, the system in question is student activity monitor and alert generator. We have already introduced this system earlier in some other case studies in this lecture we will see a report on black box testing of this system. As usual the cover page of the report contains the person number shown here along with the date of the person that is followed by the name of the persons who have prepared the report the place as well as some additional information like who is supervise the report but of course this is optional you may or may not include this that is the cover page.

Next comes a table of content where all the sections and subsections are listed introduction section having these four subsections. Then the testing report which is the core component of this document having several subsections in these subsections as you can see there is some introduction to black box testing then the goal of testing evaluation of test cases unit testing then the testing report for individual functions that were tested.

Namely these three functions were tested compute orientation calculate proximity and second frequency. So, we will talk about what these functions are and how those are tested for each of these functions details were provided like the domain of input values domain of output values then equivalence class partitioning boundary value analysis and observations that is given for all the three functions that were tested for this system. And the document ends with a conclusion section.

At the end of the document it is advisable to add references if some references were used for preparing the document. After that like before some revision history may be mentioned ideally it should be included to show the historical evolution of the document when it was first created and when it was created next or refined next the whole history along with the version numbers given to each of the persons then comes the first section that is introduction.

So, in this section common things are listed these things are mostly common to all the documents that are outcome of each of the stages namely purpose, purpose of this testing. So, the purpose of black box testing for the particular system is to start planning as soon as the artifacts are ready the main intent of this is that the developers can realize if their requirements will surface and if the program does as based on the functional requirements.

Then the hardware used for testing is mentioned in this case it is mentioned that mobile device used for testing was a particular make and model with the following specification amount of ram processor details and additional memory space used if any. This is followed by the conventions used in the subsequent part of the document that is whenever this term occurs instructor professor it refers to person who shall be using the software for monitoring.

If the term student is used that refers to a person who shall be monitored by the instructor users are basically persons who can be either students or instructors. Device refers to an electronic device using which the instructor is delivering their lecture accelerometer a sensor that helps determine. The second frequency gyroscope a sensor that helps determine the orientation of students device proximity sensor sensor that helps determine the distance between the student and their device.

Whenever the term sensors are used they collectively or the term collectively refers to the three sensors accelerometer gyroscope and proximity sensors on the students device. Android refers to the android operating system Google refers to the Google company. So, this list provides us the conventions terms that are used in subsequent part of the document. Then comes project scope this software is meant to be deployed in an IT enabled large classroom environment where in the lecture delivered by the instructor is via a device through which both audio and video are transmitted.

This software shall allow the instructor to conveniently monitor the attention of their students in real time and if required generate some alert. So, that is the overall scope of this system or the project. So, that is the content of the first section that is introduction. Next comes the main part of the document that is the report on black box testing of the system. So, in the black box testing section which is the main section of this document it starts with an introduction this talks about in brief black box testing and what it does.

So, black box testing attempts to find errors in the external behaviour of the code in the following categories. So, it lists some categories in which the errors are analyzed incorrect or missing functionality interface errors, errors in data structures used by the system is by the interface behaviour or performance errors initialization and termination errors concurrency and timing errors. These are broad categories of errors that are analyzed in black box testing of course in our lectures we did not mention about these categories but these categories are a primary concern here.

Through this testing we can determine if the functions appear to work according to specifications that is the overall objective of this testing it is suggested that the black box testing is performed by an actor who is not a developer as the tests are made to ensure the functions perform as needed by the customer. Black box testing is also called functional testing and behavioural testing. These are about some information on black box testing some general information.

Next comes the subsection goals of the testing because black box testing purposely disregards the program's control structure that means the internal structure of the code attention is focused primarily on the information domain that is the data that goes in and the data that comes out. The goal is derive sets of input conditions or test cases that fully exercise the external functionality. So, it says that black box testing that does not consider internal structure.

So, the goal is to come up with suitable test suits to test different functional aspects of the code. Next section is evaluating the test cases again it talks about the general idea of the testing considering the number of input classes and enumeration of test cases in each of the total number of test cases will be too long or too large for testing. To avoid this issue these two things are done first we perform equivalence class partitioning.

That is followed by boundary value analysis all these things we have already discussed in details in the lectures hence we use the technique called equivalence class partitioning which partition the input domain into equivalence classes where the set of test cases in the same class as the property that set of data should be treated the same by the module under test and should produce the same answer.

In other words any one input value taken from an equivalence class should be representative of the whole class of input values thus all test cases can be assigned to a limited number of equivalence classes and output is found immediately. In other words we can manage the number of test cases by dividing the input domain into equivalence classes. Now in this document it is mentioned that black box testing was performed for both system testing and unit testing.

However the document primarily lists unit testing findings. So, it first talks about unit testing. Here we divide the system into modules each pertaining to a single function of computation and evaluate the test cases according to it. Now one module or function that is mentioned is compute orientation that is one module then it presents the input domain or the domain of input values for this module. So, input domain is device orientation.

Now this orientation has three components there are three degree of rotations namely azimuth Pitch and Roll. These are measured in radians azimuth's angle of rotation about the z axis this. So, there is some short description of this concept this value represents the angle between the devices y axis and the magnetic north pole. When facing north this angle is 0 when facing south this angle is pi likewise when facing east this angle is pi by 2 and when facing west this angle is minus pi by 2.

Thus the range of values is minus pi to pi pitch is the angle of rotation about the x axis this value represents the angle between a plane parallel to the device's screen and the plane parallel to the ground. Assuming that the bottom edge of the device faces the user and that the screen is face up tilting the top edge of the device towards the ground creates a positive pitch angle the range of values is minus pi to pi.

Then comes Roll, Roll is the angle of rotation about y axis. So, earlier we talked about of rotation about z axis, x axis. Now it is about y axis this value represents the angle between a plane perpendicular to the devices screen and a plane perpendicular to the ground assuming that the bottom edge of the device faces the user and that the screen is face up tilting the edge left edge of the device towards the ground creates a positive role angle the range of values is minus pi by two to pi by two.

So, here in this description as you can see it talked about the nature of the input domain for this particular function a brief explanatory note on the input values including its components which is helpful for the reader to understand the concept. Hence the information is input as or the input is defined as a triplet Azimuth, Pitch, Roll in this form. Then comes the output explanation. So, what is the output domain for this input the output is a Boolean value true or false which is displayed when correct orientation function determines whether it is correct or not.

So, there is some function mentioned which produces this output. So, overall function which contains some other functions of course but this overall function takes as input this triplet Azimuth, Roll, Pitch and produces an output as a Boolean value true or false. Now with that knowledge let us now proceed to the next stage of black box testing that is to identify the equivalence classes for design of the test suit.

So, we need to now identify the classes or the equivalence classes by partitioning the input domain from the input domain we observe the range of the parameters and set the equivalence classes in the following way note some note is mentioned which is all values are taken to be in degrees as we are converting them in the compute orientation function. So, some other function is also mentioned which is used to convert from radian to degrees.

For the orientation triplet Azimuth, Pitch, Roll the equivalence classes for each parameter are for azimuth the range of values are all floating point numbers and hence there are two equivalence classes as defined by the creators of this document. All values between 180 and minus 180 in other words this range minus 180 80 with a representative value 0. Now this as we have already mentioned equivalence class is nothing but a set.

So, in set notation we can represent this as set of all x's whereas x is the variable where x belongs to the real numbers and x is greater than minus 80 and less than x is greater than minus 180 and less than 180 that is one equivalence class. One more equivalence class for Azimuth input values all values not between minus 180 and 180 that means the set that does not belong to this range. A representative value chosen is 250 and in the set notation this can be represented as set of all x's where x does not belong to r minus does not belong to this set and x is greater than minus 180 and less than 180.

Next is the pitch component. So, for pitch there are three equivalence class partitioning why because we have a pitch error margin of 5 and an absolute value of 0 hence there will be 3 equivalence classes as defined by the evaluators with values specified in degrees. One equivalence class is all float values between minus five and zero that is minus five zero representative value taken is minus two in the set notation this is represented as set of all x, x belongs to r x greater than minus five less than zero.

Next equivalence class is all floats between zero and five that is this range zero and five representative value taken is three the set form set of x, x belongs to r x greater than 0 less than 5 and the third equivalence class is all values not between minus 5 and 5 that is a set excluding these values representative value taken is 45 in the set form set of x and the x is defined in this way. Then comes roll we have a role error margin of 15 and an absolute value of 45 hence there will be three equivalence classes pertaining to the error margin.

Class 1 all floats between 30 and 45 that is this range 30 to 45 representative value taken 37 set form set of x, x belongs to r where x is greater than 30 less than 45 then all floats between 45 and 60 this range 45 to 60 representative value 52 set form set notation this class can be represented as set of all x, x belongs to r x greater than 45 less than 60 and the other class is all values not between 30 to 60 that is the set excluding these values representative value taken is 45 and set notation is shown here.

So, this is how the equivalence classes are chosen note that this is one way of choosing the equivalence classes this need not be the best way or need not be the only way as we have already discussed during the previous lectures that choice of equivalence classes is very important to cover as many errors as possible and here in this document one such choice is shown where for the input domain comprising of this triplet azimuth rule pitch total eight equivalence classes were chosen.

Now total number of equivalence classes then will be actually ten when we combine them together for each class there will be three role giving us nine combinations and one class for azimuth being anything other than between minus 180 and 180 and accordingly there are 10 test cases chosen which is the test suit. So, it shows the input combination the output combination for each of these test cases as you can see here test case 1 2 3 4 5 6 7 8 9 and 10.

For each test case the input and output combination is shown to take care of the different scenarios. Now once these test cases are chosen actual execution were performed for each of the test cases for this total 10 test cases and the output was observed for each test case. So, this column shows the actual output that is observed during execution of the code as you can see here expected output is true and the observed output is also true.

So, there is no error true true no error false false that means no error output is as expected true true no error true true again no error false false no error false false no error false false no error false false no error and instead of error message it exits that means there is some issue here ideally it should have shown some error messages but instead the program simply exits. So, some correction may be required at this stage for this particular input combination.

In fact it can be also concluded as for this equivalence class or the set of input values belonging to this equivalence class there is a mismatch between the expected outcome and the actual outcome and the program should be suitably rectified to take care of this equivalence class of input. Now once the equivalence class partitioning is done next is boundary value analysis for this module for azimuth the boundaries for the minus 180 180 class will be as mentioned in this line that is minus 181 minus 179 180 179 and 181.

For pitch boundaries for minus 5, 0 and 0 5 will be the test suit minus six minus five minus four minus one zero one four five six for roll boundaries for 30 to 45 range and 45 to 60 ranges will be the test suite 29 30 31 44 45 46 59 60 61. So, for each of these boundary cases some test cases can be designed and then the program behaviour can be observed to be double shear about the program be aware at the boundaries.

So, for azimuth there can be four such test cases designed with input value and expected output and then it can be compared with the actual output which is shown in this table. So, in the first case instead of error message the program exits. So, there is some issue in other cases true true true true. So, no issue no error but again in the last case instead of error message it exits. So, again sum is similarly for pitch for the boundary cases three such cases can be defined and here as we can see the expected output and the actual output are matching true true true true false false.

So, no issue for role three cases and expected and actual outputs are matching in all the three cases, so, no issues here as well. So, finally it can be concluded the observation can be concluded as all the outputs match the expected case except when the azimuth does not lie between -180 and 180 where instead of an error message the program exits. So, some rectification may be done there the next function that is or the next module that was tested is named as calculate proximity.

Here the input domain is defined as proximity value the proximity value for our used mobile device Moto G play gets the proximity range from 0 to 100 and output domain is again boolean true or false which comes at the checkoff is correct proximity in the check attention function some details are given which is irrelevant for our discussion. So, given this input and output domain information then let us see equivalence class partitioning for this function.

The units for the values are mentioned in centimeter four equivalence classes were designed all floats between zero and twenty with the representative value taken as eleven all floats between twenty and twenty five with a representative value taken as twenty four all floats between 25 and 30 with the value taken as 27 and all floats between 30 and 100 with the representative value taken as 55. So, where it should be range 30, 200 again I would like to repeat here is that.

This is not the only way to create equivalence classes this is only a sample document showing how the classes were designed of course the rationale was not mentioned in details that is also not required you can simply list the classes as per your choice. Now like before here also for these four classes four test cases were designed with the representative values and the expected output. So, in the first test case representative value or the input is 11 outputs is false where after execution they got the false output as well.

So, perfect match second case 24 input true output and actual output also observed to be true 27 true third case actual output also matches 55 false and here also actual output matches with the expected output next is the boundary value analysis for these equivalence classes with some assumption that we can ignore here for proximity the boundaries for the equivalence classes 0 to 20 20 to 25 25 to 30 and 3200 and the corresponding test cases will have 0 1 19 20 21 24 25 26 29 30 31 99 100.

So, accordingly four test cases were designed for a group of boundary values one expected output false and program was executed and for those values false output was observed. So, match for another group of boundary values true was the expected output and true was actually observed after execution another group of boundary values observed value was true whereas the expected value was also true.

So, again perfect match and one value this should be twenty five instead of minus 25 for 25 one boundary value expected output was true actual output was also true, so, no deviation. So, it behaved as per expectation. So, finally the observation is all the outputs correspond to the expected output and hence there are no ambiguous results that mean no error was noticed in this black box testing of the particular function.

The third function that was considered is checking frequency note that here we are not bothered about what is actually there in the function that is not the idea of black box testing instead we are just mentioning the name of the function and what input it takes and what output it produces that is good enough for performing black box testing as we have repeatedly mentioned in our earlier lectures.

So, the two functions that we have discussed you may be wondering what were those functions how those were written and more details about those functions that is not at all required because in black box testing we simply assume that the functions are black box what we need is only the name its input and its output and that is what we are seeing in this document as well you need not bother about the details of the function or how they work you should only learn its name and the input domain and the output values.

So, the third function that was considered is checking frequency its input domain was defined as proximity value the second frequency for the mobile device that was used for testing gets the checking frequency from anything above 0 and the output domain was defined as the output is again a binary or Boolean value true or false which comes at the Checkoff is correct frequency in the check attention function. The details that is provided here how the output comes is actually not required we can as well remove this part.

And that is true for the earlier cases as well although it is given here how this output comes is not of any use for this particular testing method we can simply ignore that. So, in your document whether you keep it or not keep it is up to you. Then let us see the equivalence classes for this function from the input domain the classes were designed two equivalence classes were designed by the designers by the evaluators one is all floats between 0 and 5 within this range with a representative value 4 a set notation given.

And second class is all floats above 5 that is between 5 to infinity representative value taken as 20 and a set notation is also given. Accordingly two test cases were designed in each case the input value and the expected output were mentioned and the program was executed with this input and the output was observed the actual or observed output was found to be matching with the input both are true for the other range other test case both are found to be false.

So, again there is a match between the expected behaviour and the observed behaviour after execution when we come for boundary value analysis for this case for this particular module again the assumption is made that the rest of the parameters will lead to a true case. So, based on this assumption the boundary values were decided for frequency the boundaries for these two ranges 0 to 5 and 5 to infinity we will have test cases 0 4 5 6.

So, there will be 4 test cases in the test suit designed for boundary value analysis input 0 expected output true and the observed output after execution was also found to be true input 4 expected output true observed output true input 5 expected output true observed output true input 6 expected false observed false. So, in all the cases there is a match between the expected and observed.

So, what we did here or what the evaluators did they first identified the equivalence classes correspondingly test suit was designed then the boundary cases were identified and correspondingly another test suit was designed for each function then these test cases belonging to the test suits were used to execute the program for those inputs the outputs that were produced by the programs or the functions were observed those observations were matched against the expected outcomes.

And if there is a match then conclusion is no issues are there with the programs and if there is a mismatch then the conclusion was that there were some issues and some rectification may be required. So, in this case in the case of third function and the corresponding boundary value analysis the observations were all the outputs for both equivalence classes and boundary values corresponding to the expected output and hence there were no ambiguous results or rather in other words there were no issues with the program.

So, that is about observing the functions that are primary functions in the code and after performing the black box testing we compile the observations and come to a general conclusion. So, the last section is the conclusion section in this section we list the overall observations the only discrepancy found while testing was in the checked orientation case. When the value of azimuth that is the z axis rotation variable was out of bounds or not in the given range.

In that case ideally an error message should have been there but in reality when the program was executed with such a value then the program abruptly exited this is resolved by adding an else statement to the required program part with error message. So, in the conclusion also it is mentioned how the issue was resolved however that is not mandatory that need not be added here. All other outputs correctly match expected results also we have exhaustively gone through the input parameters and validated all the test cases.

And some conclusion that black box testing is completed for the units and we have ascertained that all the modules output the correct result for a particular pair of inputs belonging to an equivalence class. So, what it tells is that what is the final observation how that issues if any that were observed during the testing were resolved and then it says that the testing is complete. Now resolving the issues as I just mentioned need not be part of this document however if you keep it for later use that is still all right.

So, overall what we can say is that in this document in this black box testing report what you should include primarily are the functions or the modules that you have tested detailed description of their input domains and output values that each function or module takes then based on those input and output information what are your equivalence classes optionally you may include some rational behind going for those equivalence classes.

Although that was not clearly mentioned here but ideally if you can keep such an information that it will help others to understand your logic then details of those classes including representing them with set notations for clarity. Corresponding test cases with input and output pairs result of observation after executing the program or the function or the unit whatever you are terming it with the input and output pairs for each test case.

Then comparing the output that is observed with the expected output that you have already identified and then finally concluding whether there are any mismatches that you have noticed the same thing you have to do for boundary value analysis also. So, for each of the equivalence classes that you have identified you have to identify the boundary cases and correspondingly up to design test cases and for those test cases also you have to identify the expected output execute the program to get the actual output.

The actual output has to be compared with the x with the expected output and see whether there are mismatches. If miss matches occur you have to note it down for later actions you have to do it for each and every function. So, these are the key things that you should include in the test report including set notations for all the equivalence classes. Detailed explanation of the input output is mandatory set notations for equivalence classes is mandatory.

Justification for choice of equivalence classes although optional ideally should be part of the document for understanding by others. And conclusion about the test observations if there are any issues found and optionally you may also include like shown here how those issues were resolved at the code level at the end you may add some references like those shown here. So, this section says that to prepare this document the developers have gone through these references.

So, nine references were mentioned of course you have to follow some standard although here it was not done following proper standard conventions but that is what is also required.

(Video Ends: 42:55)

So, with that we have come to the end of this case study where we have seen how to prepare a report on your black box testing. The key components of the reports are highlighted some additional things you may keep including justification for equivalence classes some things are optional like details on how the issues are resolved although if you keep it there is no harm in it. I hope you have understood the document the lecture content.

And you will be able to prepare your own black box testing report without any problem with that I would like to end this lecture here hope to meet you all soon in the next lecture thank you and goodbye.