SOFTWARE TESTING - ISYS1085-7

**WEEK 10 Tutorial Assessment**

Software Inspections: Post-mortem & Test techniques

horizontal line

# 

## Instructions

Please answer all these questions as part of the team that conducted the Inspection, and submit in canvas using the same group (Tutorial 8 Assessment <Number> ).

## 

## 

## Section A) About the Inspection (1.5 points)

1. **Typo Lists are really not necessary for Inspections. Who cares about spelling, grammatical, formatting and style errors for Software Inspections? We are programmers not technical writers or journalists. Do you agree? Provide explanations.**

I disagree, for one; typos, grammatical errors and formatting inconsistencies can hurt readability which has a number of ratifications. For example, it might hinder how widely accepted the software is; who wants to use software that is “sloppily written”.

Also, these errors could also cause confusion when a third party needs to examine the code. Extending from this point, such confusion could lead to misinterpretations or general errors due to preconceived notions of what certain bits of code means. Furthermore, such errors could lead to financial catastrophes. <https://mentalfloss.com/article/49935/10-very-costly-typos> lists times where typos have resulted in huge financial losses, not necessarily code and software related; but it illustrates the importance of pedantic things such as spelling, grammar and formatting. Some misunderstandings are more detrimental than others; best to test whether they are done right and save the future hassle.

1. **There six roles specified for Inspections (Author, Moderator, Recorder, Reader, Inspector and Verifier). What is the minimum team members required for an Inspection? Provide explanations for your answer.**

5. The exclusive roles are Author, Moderator, Recorder, Reader, and Verifier. Meaning, only 1 team member can be each of these roles, resulting in needing at least 1 member for each role.

Inspector however, is a shareable role, and a role that can be held by members with the other inspection roles. The only exception is the author, who cannot inspect his/her own work. Other than that, all the other team members can double as an inspector.

1. **Who decides on suitable Inspectors for an Inspection? Who would make suitable Inspectors?**

If the decider is within the inspection team, the Moderator, as the leader of the inspection, would decide who is a suitable tester for the task.

A suitable tester is someone who is skeptical, someone who is always checking if more bugs exist, believing the build given is not bug free.

Someone who does not compromise on quality, maintains their level of work throughout all stages of testing; any point of lowered quality testing can lead to a defective product.

A good inspector would think if the user would be happy with the product after testing; testing can go beyond the requirements to ensure the user is happy using the product.

An inspector that thinks from the customers perspective can give the inspector insight on how the customer will use the product. Lack of considering their perspective can lead to the inspector missing bugs.

A good inspector will utilise his/her testing time by prioritising their tests. Focusing on test cases with higher importance, reducing the time pressure in the latter end of testing.

A good inspector has to be realistic, and set their own testing scope. Do not promise 100% coverage, as it is unrealistic and essentially impossible.

As this is an inspection team, the inspector must be able to work within their team and be open to suggestions.

A punctual inspector is a good inspector, it’s a sign of good time management as they can have optimal time for planning tests and executing tests.

While testing, it is important to consider the risk factor. A good inspector will be able to identify, manage and mitigate risks. Priorities of software testing are based on risk evaluation.

Being proactive and looking beyond the product requirements can give the inspector context and further insight on the product and how to test it.

Fundamentally, being able to analyse the requirements well will allow the tester to accurately write tests to see if the product meets said requirements. A must for inspectors.

Use negative testing, using invalid data and trying to “break” the product is essential to test the robustness of the software.

An inspector needs to be able to judge whether what their testing is simply right or wrong and develop the reasoning.

If the inspection team has collaboration with the development team, it is good to negotiate with them with the test results. Negotiating why the tests yielded the outcome they did.

The inspection team needs to work well together and now throw blame around when an error occurs. A good and robust team of inspectors will analyse the error, discuss it, plan how to fix it, and move on.

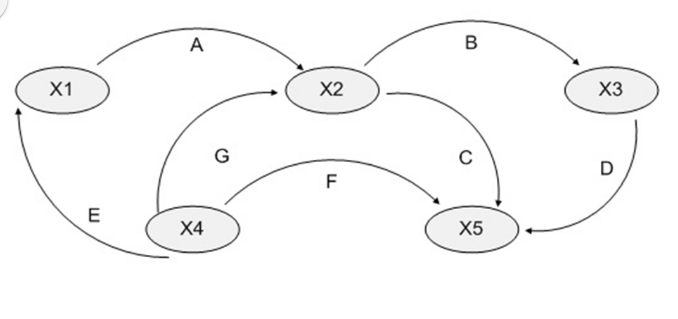
Overall, an inspector needs to be observant of the testing process. Being flexible and being able to reschedule activities of the testing process as deemed fit based on actual progress in relation to the planning progress (Use of the testing time given/planned). This will ensure the inspectors work is on track for completion.

1. **What kind of work items are considered high risk and therefore a good choice for Inspections?**

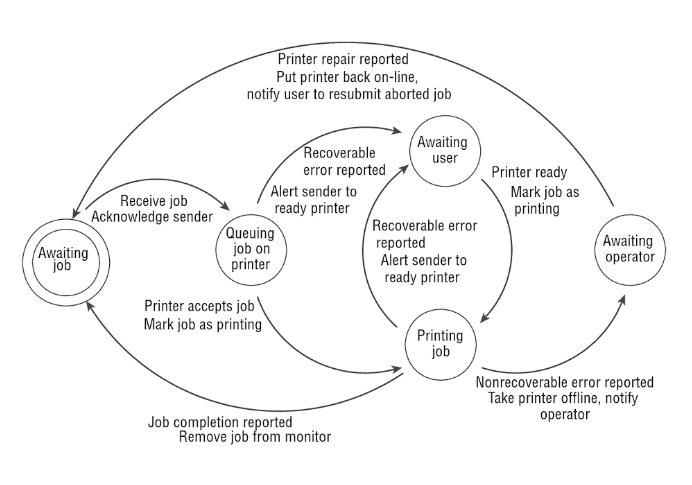
Items which hinder the expected performance of the product in relation to the requirements; or that cause the product to be incapacitated entirely, should be of the highrest priority in inspection; mainly operational risk items. After that, items with associated financial risk should also be prioritised; as large financial loss can result in project cancellation or other halting consequences.

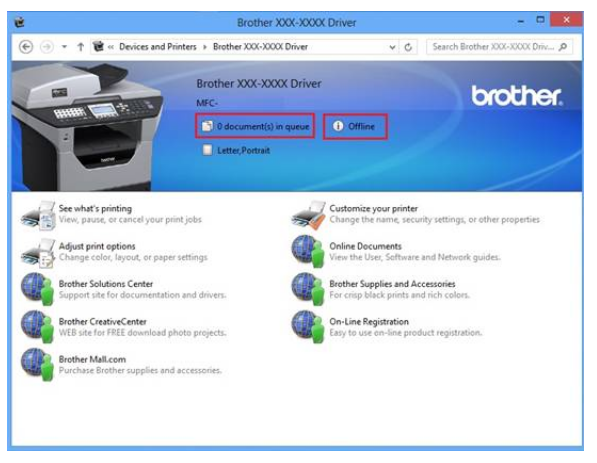
1. **What are 'Entry Criteria' for Inspections? Name 2 important entry criteria.**
2. **Typical mistakes for the Inspection meeting were: (1) Not identifying meeting date, (2) Missing line numbers in the Issues Log, (3) Excessive or inflated time spent in preparation, (4) Individual time spent does not equal overall time spent for the meeting.   
   What is the impact of these mistakes on the software team and organisation?**
3. **Please consider how your team ran the Inspection exercise. How closely did you follow the documented processes and how can you improve on this for a future Inspection meeting?**
4. **Would you consider using software Inspections in your future workplace? Why/Why not?**

## Section B) About State Graphs (1.5 point)



1. How many 1-Switch transitions are possible from the state X4? [[1]](#footnote-0)
2. What is the minimum number of valid transitions to cover all states? How many test cases would you need? For example, X4-X2-X5 are the transitions for one test case, in a 1 switch transition.





1. Printer software is an example where state transition testing is used. Can you outline (and briefly describe) how you would plan for the minimum test cases to test all states?
2. What if you want to test all transitions? What other test cases would you add?

## Section C) About Decision Table testing (2 points)



1. Refer to the above table to send a letter in Australia. Prepare a decision table to cover the possible price outcomes, for a web API that will calculate the price for posting such letters.   
     
   You can suggest what the fields for the web API would be - for instance, the dimensions, weight of the item, as well as whether priority is required.  
     
   (Ref.https://auspost.com.au/sending/send-within-australia/compare-letter-services/regular-letters-cards )

(Ref. Decision Tables in textbook - section 4.3.2[[2]](#footnote-1) )

1. What if you want to perform boundary analysis testing to[[3]](#footnote-2) your testing of this Australia Post web API. Show with one Rule of your above proposed decision table, how you might use boundary analysis to create more test cases.

1. <https://www.getsoftwareservice.com/state-transition-testing/> [↑](#footnote-ref-0)
2. <https://www.getsoftwareservice.com/decision-table-testing/> [↑](#footnote-ref-1)
3. [↑](#footnote-ref-2)