ETA Week 5

The Session in Review

We began this session with a broad opening question – “why do we test?”. This was to start a mini discussion on many of the themes we have covered during our classes so far. It was good to hear some answers that reflected ideas we have discussed but stated in your own way. That shows you are learning things and gaining an understanding. From this discussion, we reviewed the homework task of static testing the functional specification we have been using. A couple of things came out of this. You are all capable of applying critical analysis, asking questions that reveal possible omissions, ambiguities, uncertainties, etc. It also became clear that when you put a little bit of time into your homework task you develop a better understanding of the subject. Remember our discussion from the previous week, just an hour a week.

We spent part of this session talking about a concept we had briefly touched on a few classes previous – exploratory testing. We mentioned how many people hear exploratory testing and give it meanings it is not intended to have. Terms such as “monkey testing” and “keyboard bashing” are not what we mean by exploratory testing. We looked at the term “ad-hoc testing” and referenced the ISTQB meaning. In brief the definition refers to ad-hoc testing as informal with no preparation. As an interesting aside, the term “ad-hoc” means “for a purpose”. You might want to consider how that fits with the ISTQB definition (if you Google ISTQB glossary you’ll be able to download a PDF document of all the ISTQB definitions).

We then spoke more about exploratory testing the way Lee and Paul use the term. In short, when we use this term, we mean testing that allows us to explore and learn as we go. Lee spent a little bit of time talking about Session Based Test Management (SBTM). This is a method where our test goals are set out in a charter. The charter is executed during a session. While testing we take notes. The session has a time limit. At the end of the designated session time limit we debrief with selected people and go through our findings (notes). From our debrief discussions we might decide no further testing is required in the charter area or we might decide further testing is needed as we need to know more before we can say we satisfied with what we know.

We applied exploratory testing, as a group, to a calculator written in Ruby. We had no specification to tell us about the calculator, our charter was to learn about it and gather information that might be useful to our stakeholders. Very quickly the group suggested that the calculator had behaviour that was inconsistent with other calculators they had used (the consistency heuristic). You also assessed usability as problematic. That is, you reported the calculator as not being user friendly. We spent some time adding inputs and found some unusual patterns in output (sometimes the result was +1 above our desired result, at other times it was -1 below our desired result.

To close this exercise, we had a brief look at the code. Approximately 30 lines of code that held a lot of complexity when you had to work out its behaviour. A lesson here – it doesn’t take a lot of code for programs to become complex. Most code you test will be much, much bigger than this.

To close the session we started discussing agile, what it means to be agile and some of the attributes that are related to agility. We will revisit these as part of the next session before we move on to discussing scrum (the most popular framework for agility).

Thanks all for another very interesting session.