(8)

FORTEST Meeting 13 September 2002

The role of testing in a Formal software/systems development

Question:

What is the role of testing in a formal software development?

Imagine a scenario:

Customer: "What are you doing?"

Supplier: "Testing the software we are about to deliver to you"

Customer: "But I thought you've proved it correct?"

Supplier: "Oh yes, we have!"

Customer: "Why then do you need to waste time and money testing it? If you have proved it correct there is no need to, surely?"

Supplier: "Er, well we just wanted to make sure."

Consultant in charge of supplier: "Just in case there is a fault in the proof!"

Customer: "Well, that doesn't sound too good. I thought if the software was proved correct, it was certain to be correct!"

But I felt that there was more to testing in a formal development than just a kind of belt and braces approach, a making sure that there was no fault in the proof. I felt in my bones that testing has a proper, mathematically respectable role in a formal development process.

Proofs and Refutations

I was reminded of Imre Lakatos's famous thesis "Proofs and Refutations". In it he illustrates the proof of Euler's theorem V+F-E=2 for the vertices, faces and edges of a solid figure bounded by polygons. Each refutation consists of a "test" in which the calculation of V+F-E for a slightly unusual polygonal solid figure is carried out, to give a result other than 2.

The role of testing in a formal development

Testing can be seen as the search for a refutation of the claim that one has constructed a proof that a program is correct. Searching for a refutation is an activity entirely within the traditions of mathematics.

It can therefore be built into a formal development life-cycle, of whatever flavour.

A Socratic dialogue reflecting 300+ years of search for counter-examples and refining the proof.