Publication: APD - Financial Times Europe; Date:2008 Oct 08; Section:Digital Business; Page Number: SR1-2

PERSPECTIVES

Software testing remains a Cinderella discipline

Alan Cane

Within minutes of the start of trading on the London Stock Exchange on October 27 1986, there was meltdown. The brand new Seaq computerised pricing system – the heart of the City's "Big Bang" transformation – had failed.

The reason, not uncovered until a few weeks later, was that a series of operations which should have moved chunks of information from one part of the system's memory to another had collapsed under pressure, leaving some data in the wrong place and leading to a crash.

The fault had not shown up during testing because Seaq had never experienced the demand for pricing information that the Big Bang generated. Testing equipment that was capable of generating such a volume of requests was not available.

More than two decades later, a few minutes after the unveiling of Terminal 5 at London's Heathrow airport this year, there was uproar as the baggage handling system crashed.

Willie Walsh, British Airway's chief executive later explained to parliament's Transport Select Committee that a lack of testing had contributed enormously to the problems.

Software engineers had installed filters in the system to prevent messages generated during testing being delivered to live operations. But the filters were accidentally left in place.

Testing was also curtailed because of construction delays. Building work should have finished by September 17 last year but the testers could not start work until the end of October. Several tests were cancelled.

These examples, although widely separated in time, show that decades after the first commercial data processing software went live, testing of non-safety-critical systems is still often a hit or miss affair, commissioned by managers who do not necessarily understand the purpose of the exercise and therefore are unable to set clear goals for testers who are often underskilled or unqualified.

Geoff Thompson, chairman of the UK testing board and consultancy director of the testing specialist, Experimentus, is critical of managers who employ testers without really understanding why they are doing so.

He quotes one manager who claimed that once he had hired someone to do the testing, he neither knew nor cared what they did; he simply stopped worrying about it.

Another horror story concerns the contractor who claimed he had achieved 95 per cent coverage of the testing agenda for a particular system. The business managers were delighted and agreed the system should go live. It failed immediately and mysteriously. It turned out the contractor had carried out 95 per cent of the tests he thought he should do, without reference to any other authority.

In theory, testing is simple. Data are generated, fed into the system and the outputs examined for accuracy.

In practice, as the Seaq programmers discovered, it is devilishly complex. It is thought to be impossible to prove completely that any program of 1,000 lines of code or more will not do something unexpected under certain circumstances.

Some believe that to test a decision-making program consisting of only a few lines of code exhaustively would take several human lifetimes. These stories may be apocryphal but the reality is that testing is still a cottage industry rather than an engineering discipline.

The board chaired by Mr Thompson is the UK arm of an international body, which attempts to set standards and qualifying examinations for would-be testers.

He recognises: "It is better than anything else available, but it is never going to be a perfect body of knowledge until there is proper science behind what we do. It really is just people's views. There are some techniques and some science but it's not complete and it's not engineering."

Meanwhile, a huge amount of time and effort is being expended in universities and elsewhere in attempting to develop problem-proof software.

Object-orientated techniques, where already-proven modules can be linked together to create functional systems may be one approach. But these developments will take time to emerge from the laboratories.

In the meantime, Mr Thompson suggests that managers adopt a four-point approach. First, set clear objectives, so testers understand what they are aiming for.

Second, employ capable individuals as testers (he is critical of managers whose approach is: "You're a contractor with testing in your CV. You'll do.").

Third, test procedures should be properly considered at the beginning of a project and, lastly, testing should begin earlier in a project, so that the software developers take responsibility for the quality of their output.

"When will companies understand," he complains, "that the earlier a test team is involved and the better trained they are, the more they will recognise problems early in the lifecycle, enabling them to be resolved before they become execution defects?"

It goes without saying that testers need a combination of knowledge, demonstrated by passing qualifying exams, and on-the-job experience.

But until the majority of testers fit these requirements, fiascos such as Seaq and Terminal 5 will continue to blot the IT copybook.