

News: conferences

The fourth International Conference on Software Testing took place in Cologne at the beginning of April and proved as usual an important, interesting and influential event

Extreme Programming (XP) was the hot topic at this conference. Strong and convincing advocates spoke, but sceptics and those undecided but eager to learn more were also heard and, in my opinion, clear progress was made to better understanding. The opening keynote by **Kent Beck**, “the father of XP”, explained the thinking behind the XP approach using an analogy with manufacturing processes. Beck described how innovative manufacturing experts in Japan discovered that their processes could be improved by reducing inventory; the unfinished parts held in stock might be faulty, but that would not be discovered until they came to be used in the next stage. By attempting to finish them immediately, feedback is obtained more quickly so any problems further back in the process can be corrected; thus fewer defective parts are made and less time is spent correcting their defects. He extended this theory to software by suggesting that the ‘inventory’ of work not yet in production - code, tests, design decisions, documentation, specifications, known defects etc - should be reduced or eliminated. Until these are put into production there is no feedback on them and they don’t really ‘exist’: documentation is already obsolete by the time it comes to be used; design detail which has been decided upon but not yet been implemented cannot be relied upon to work as expected; and defects which have not yet been fixed slow the project. In the ideal XP project, detailed design decisions are not discussed until at most a few hours before the programmers implement them; before this, only the high-level ideas exist. Even though I am familiar with the XP ideas, having read books by Kent and others, I found this new explanation fascinating and clarifying.

After this opening, each of the three days began with another keynote. The first keynote was given by **Joachim Hauser** of BMW. He described in broad terms how development and testing of the many computer systems present in modern and future cars, including drive-by-wire, navigation, entertainment, Internet access, personalisation, auto configuration and telematics (your car tells the manufacturer directly how it is performing) is organised in BMW and their supplier companies. It was suggested that a complimentary sample of such a car should be given to conference participants but that didn’t happen this

time. It was interesting to hear that BMW are requiring their suppliers to apply the SPICE (ISO 15504) process improvement model. Finally, the next generation of BMW cars may include software which can be upgraded - perhaps by installing a disc supplied by the manufacturer, or downloaded using a communications link; I wonder if this will give rise to a market in unauthorized third-party ‘hotroding’ and other software?



Following the keynotes, ICSTEST split into three ‘tracks’ of six or seven further talks each. However, the event was being run concurrently with two others: the Conference on Software Validation for Health Care (CSVHC) and the Congress of Software Quality Management (SQM). Both of these featured a full programme of talks and workshops (some given in German, but with simultaneous translation) and delegates were able to attend any of these. There was also a special “tools track” for vendors of test software. This added up to a bewildering number of intriguing choices for which ICSTEST delegates may not have been prepared, having received only the ICSTEST programme.

On day one I attended the *Test Preparation: Test Cases and Test Data* track. Highlights included **Wolfgang Grieskamp** of Microsoft Research who demonstrated *Abstract state modelling Language* (AsmL), a standard language his team uses for modelling systems and then validating the model. This language looks like pseudocode, and Grieskamp demonstrated an application which generates a state transition diagram and test suites from it, but it is actually executable and can be animated (monitored) by the addition of small modules written in C#. Of course, this impressively

powerful approach requires skills many testers do not possess, and Grieskamp acknowledged this but said that while during the dotcom boom (“2-3 years ago”) non-computer scientists had to be hired as testers, Microsoft’s experience suggests that this is no longer the case! This is obviously a controversial view, but it may be interpreted as a message to testers to avoid complacency and continue to update their skills, especially in the more technical and practical areas.

On day two I chose the *Test Process Automation* track which was opened by **Terry Stephens** who described fascinatingly, and frankly, how he managed test environments during the huge project required by the takeover of NatWest, one of the UK’s largest banks, by the much smaller Royal Bank of Scotland, and the homogenisation of their account management systems. This involved some clever use of IBM mainframe architecture, and the cunningly-justified purchase of infrastructure for use in first in testing and then the production environment. According to Stephens, the maxim “testers come last” is not true - in fact “everyone gets the testers, then the testers come and get us - the test environment builders.”

The next speaker, **Stefan Mohacsi** of Siemens Austria, described a product which seemed almost too good to be true. *Integrating Design and Automated Test Case Generation* (IDATG) is used to create a GUI specification and generates test cases; it does not require a working GUI, but rather just a non-working prototype, showing window and dialog box designs, for all or even only part of the system. The hierarchy and behaviour of these is specified using the simple IDATG user-interface, then the steps required to complete a task (function) are defined. At this point test cases can be generated and are displayed as a flow diagram which can be edited graphically. The test cases are stored in an Access database in “a simple language using logical expressions” (this language was not shown) and can be exported as scripts for all the well-known test running tools! According to Mohacsi, if the GUI specification changes as the system is developed, regenerating the test scripts is a simple matter of making the necessary edit in IDATG and then clicking a button. His explanation of this was clear and generated a tangible sense of excitement in the room.

If this application is as powerful as it appears it may be a major step to solving the automated test script maintenance problem (see *John Kent's column in previous issues of this magazine*). One possible place to look for the catch might be in the integration of test input data which was not mentioned and is presumably done in the test running tool so may be out of IDATG's reach. However anyone doing automated testing should investigate this fascinating possibility of *generate-play* as opposed to the current *capture-replay*. A later speaker, **Dagmar Pill**, referred to this as "a paradigm shift ... the fourth generation of test automation".

Harry Sneed has been instrumenting code - ie inserting his own "probe" code snippets which create reports on how often control passes through them and the environment when it does, giving telling information on code quality and test coverage - since the 1970s, and explained the challenges involved in doing this with modern languages and development environments. This was amazing stuff and I have to admit a lot of it went over my head (and, I suspect, that of many of those present); one could only sit back and admire. He went on to link the time a test was executed (recorded by the test running tool) with the time the probe was called (recorded from the system clock by the probe itself), enabling discovery of exactly what code was exercised by each test case. Sneed's insights into software development, especially the effect of human nature and relationships, were informed by long and varied experience and were sometimes bitter and often hilarious.

On the evening of day two a conference banquet took place at what seemed from a distance a beautiful venue on the right bank of the Rhine. I did not attend this, but my colleague who staggered back to the hotel in the early hours of the morning assures me that it was most enjoyable.

Day three was opened by **Lloyd Holder** of NSTL with a keynote on certification; this may sound boring but it was quite the opposite. He began by explaining how NSTL created the process by which PCs can earn the right to carry the familiar "Designed for Windows" logo, and then went on to contrast this with the mobile/embedded arena for which dominant standards have yet to emerge. Here a new player is involved - the carriers/operators who operate the mobile networks. Suppose a developer creates a program, ie a game, which can be downloaded and played on a phone: who has the responsibility for finding out how it will behave, on different networks, if a call is received during play? How can/will the game interact with the network? If it contains defects or code incompatible with the network it may cause failure of the phone, or even affect other users.

Because the developer is removed from the carrier/operator, Holder argued, this model for development of future software products cannot scale; the answer NTSL are working on is certification of standards for phones, networks and applications - and because it is unlikely that one phone manufacturer or network will become dominant, this may be a chance for the first time to settle upon a few standards which will co-exist rather than on one, destroying the others.

After the keynote, the conference returned to Agile and XP development and I chose this track, although another track continued with the certification theme and I am told this was extremely good. Meanwhile the SQM conference was running three tracks including one on web security; this really was a case of too much choice.

Penny Rich of Merck, Sharp and Dohme described how her team delivered a new client-server CRM system to a worldwide army of sales reps with handheld (PDA) and notebook computers on the road using, and sticking to, agile development principles (similar but not identical to XP, although some people appear to be using the terms interchangeably). Very early builds were made available to users via download and they were encouraged to try these out and give feedback, giving rise to change ideas which were implemented immediately. Rich spoke with great enthusiasm and energy and it was obvious that all working on this successful project had really enjoyed the experience. She feels this enthusiasm was the key to success, but also spoke frankly about the problems which arose: people becoming "iteration happy" and releasing new builds for very minor changes, and "scope creep" - users becoming accustomed to very fast development and requesting more and more functionality. In summary, she opined that the agile approach requires *greater* discipline than traditional development lifecycles, and that traceability is even more important - this was achieved using traceability matrices. (See *Angela Edwards' article in this issue for more information on these - Ed*) However, other tools proved too cumbersome for agile development - test managers reported that in many cases requirements had been implemented before they had time to enter them into requirement management applications.

The final speaker on XP/agile was **Martin Lippert** of the University of Hamburg. He wanted to give the message that XP really works and is more than just a theory, and described how in recent projects on which he has worked the design emerged from the development - that is, the programmer creates the design bit-by-bit. In XP, testing is done constantly; the test-first technique requires programmers to formulate tests which will show that a requirement has been met *before*

writing code to meet it; thus testing is considered to be the work of the developers and the customer/stakeholder/user, rather than a separate team. Obviously this raises questions as to the role of testers and Lippert said that testers and developers need to cooperate closely. He offered practical advice on working around situations which challenge the XP approach. For example, Java Beans are hard to unit test because they will not work without the supporting container and application server. Solution: write and test a Java class instead, and then convert it into a bean for deployment by enclosing it in a wrapper. Finally, what if a programmer finds they can solve a problem more elegantly by changing the structure of the database? According to Lippert's view of XP, they should do so, changing their own copy of the database only. At integration, a database expert applies these changes to the 'master' database, referring any conflicting changes back to the programmers who made them. The discussion of these 'difficult' XP issues raised more questions than answers, but rather than detracting from the attractiveness of the concept, I feel they showed that there is more work to be done and lessons to be learned before XP can be fully employed with confidence on business-critical projects. I hope and expect that XP's exponents, and converts, will continue this work and report on their experiences and ideas at the next conference.

Nearly six hundred people attended the conference held in the vast Koelnmesse centre, and I imagine most will have found it more than worthwhile. Most of the speakers I heard were very good; the venue was comfortable and well-equipped, and the catering first class. Keynotes took place in the inspiring space of two very large halls; tracks were in smaller rooms which were nonetheless comfortable and suitable. The fact that the three ICSTEST tracks were held in adjacent rooms meant that there was no difficulty in being sure one was in the right place at the right time and also made it easy to switch tracks at will. Although choosing which talks to attend was not always easy, all delegates received the proceedings (mostly sets of printed PowerPoint slides - for ICSTEST, not the other two conferences) in a large folder and also the same material on a CDROM. However reading this is, of course, not the same as being there. Similarly papers, books and websites all have their place, but for improving your understanding of a topic there's no substitute for having it explained face-to-face by someone who already understands it, wants you to as well, and can react to your questions. I left Cologne feeling better informed, challenged and stimulated, and above all with a much clearer picture of the matters I heard discussed; I recommend the experience to everyone.

Report by Edward Bishop

PT