

The next generation

Most commercially-available training in testing is highly vocational, but testing also needs to develop as an academic subject if it is to improve and grow.

Professor Mike Holcombe describes how it is taught at Sheffield University

Software testing is a very neglected subject in the computer science curriculum at British universities. Few teach it as a stand-alone subject to any depth and for most it is a footnote in a more general software engineering or programming course. Thus it is hardly surprising that most computing graduates have a very sketchy understanding of the subject. Their experience is likely to be of informal, ad hoc testing which they have picked up in the course of assignments and project work. The use of any testing tools in universities is most uncommon.

At Sheffield, our philosophy to software testing and quality assurance is based on two key factors:

- a thorough theoretical understanding of the major principles of the subject
- real exposure to practical testing in the course of commercial software development activities.

We teach Computer Science and Software Engineering as both three year honours degrees (BSc and BEng) and four year honours degrees (MComp, MEng). A Computer Science and Artificial Intelligence degree and as well as a number of advanced full time MSc degrees are also available.

The courses are based on what I call a mushroom model. In the first two years all students do the same set of courses which cover the basics of theory and practice in programming, design, testing, project management, artificial intelligence etc. In the third and fourth years, students can specialise in research-oriented subjects which are based mainly around our research strengths: software engineering, machine learning and natural language processing, speech technology, robotics, graphics, computational biology – and software testing. In this way we can ensure that our students reach a very high academic standard.

A lot of what we do is based around group projects which start on day one, year one. This is a unique and very important aspect of our degree courses.

The course runs alongside studies in introductory programming (in Java), mathematics,

artificial intelligence and network architectures. It starts with a module called *Requirements Engineering* and during this students learn about understanding the requirements gathering process, how to write a requirements document and some of the basics of design. They work in their groups on a project called *Crossover* which will involve them in building a complete working system from scratch. Their tutors are the clients and the systems are fairly simple ones such as a theatre booking system, a stock control system, etc. Essentially these are simple databases with suitable user interfaces. By the end of the first semester they should have produced a requirements document – with emphasis on non-functional requirements as well as functional – and a basic architecture and design and some prototype user interfaces. They use tools – for example *Select* is used for the design and *Forte* is used for the user interfaces.

“Few universities teach testing to any depth. It is hardly surprising that computing graduates have a very sketchy understanding of the subject”

In the second semester Unified Modelling Language (UML) is taught by formal lecture. Students are also expected to broaden their knowledge by taking modules from another department: philosophy and chemistry are popular choices. To complete the second part of *Crossover* students must carry out:

- basic system testing
- principles of testing
- psychology of testing
- functional, structural and statistical testing methods
- implementation testing
- test harnesses and unit testing

- integration testing
- creation of use cases and test cases
- formal specifications and test cases
- implementation and maintenance
- acceptance testing and delivery
- error reporting
- version management.

All of these projects follow the waterfall model and are broken up into stages of 2-4 weeks. The name ‘crossover’ refers to the fact that each team receives, at the start of a new stage, the system documentation from another group – usually on a different system. This is intended to reinforce the message that clear documentation is vital and also gives students experience in reading other people’s material.

In the second year there is a more refined version of *Crossover*. This, the *software hut*, involves students working in groups writing software for an external business client. The following subjects are also studied in the second year:

- functional programming
- machines and languages
- systems analysis and design
- database technology
- symbolic reasoning
- adaptive robotics
- abstract data types
- professional issues
- HCI and graphical interfaces
- pattern processing.

The *software hut* places a lot of emphasis on testing because the last thing we want is clients coming back with maintenance issues after the students have finished! Another way that we ensure quality is that each client has six competing development teams and chooses the best solution at the end of the project. A prize is awarded to the winning teams.

We also use the project as a vehicle for our empirical research in software engineering. The students have to archive everything using CVS (see www.cvshome.com) including minutes of meetings, timesheets, system documentation, tests etc. Strict coding standards are imposed.

**The UK's premier
software testing exhibition**

**Thistle Tower, London
Thursday 25 March 2004**



TestExpo™

TestExpo is the **evaluation tool** for everyone interested in the latest tools, products and techniques for use by software testers. It features **free presentations** from the leading consultancies, **free demonstrations** by the leading suppliers and **free discussions** with leading experts. **By visiting TestExpo you can do a whole year's legwork in one day.** There is no better way to find out about and compare the best the market has to offer, and to meet the top companies in testing, all under one roof.

sponsored by  **n focus**
making **IT** work

Visiting TestExpo is FREE.
Order tickets at www.testexpo.co.uk

This archive is an excellent resource for researchers looking into the way projects develop and we try to compare different approaches to development. For example, recently we have got half of the teams to use Extreme Programming (with very strong emphasis on early testing) and the other half a more traditional design-led approach. We hope to be able to make more valid conclusions about the merits of the two approaches than those of most other experiments which do not include real clients and thus miss the point in my view.

In year three the programme is more flexible and students can choose from a wide range of courses. One of these is software measurement and testing. It is based around the following topics:

- the category partition method
- the cause effect graphing method
- implementation based testing
- mutation and statistical testing
- state-based testing
- software measurement theory
- applied software measurement
- measurements and models in software engineering
- collection and presentation of measurement data
- measurement in the management of software engineering.

Students who progress to the four year programme can then take part in the innovative *Genesys* project. This involves students running a company that offers IT consultancy and software development services to outside organisations. The emphasis of the work will be on learning how small IT companies are created and managed, the legal and financial frameworks within which such companies operate, the practical management of the companies and their successful trading. Testing plays a vital role in this work; we use state-of-the-art methods to ensure that our products are of the highest quality. Full details can be found at www.genesys.shef.ac.uk.

At the end of this course the students should:


- understand the business issues that underlie the development of software systems
- be able to reconcile business and technical requirements from clients, and plan work accordingly
- be able to evaluate the requirements for software systems, both at the level of individual systems and of the organisations which require to use them
- be able to develop successful software systems for clients, starting from imprecisely stated requirements

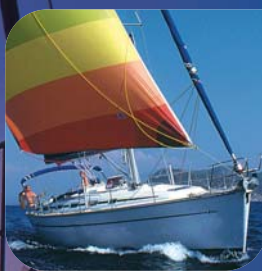


- be able to select and apply appropriate quality management measures at every stage during the development of software systems.

The content is based entirely based on the activities necessary to run the software development organisation *Genesys*, and to undertake the projects that it accepts from external clients. The participants in manage the organisation at regular weekly meetings. These involve them reviewing *Genesys*' operations as a business, deciding which possible projects to accept, and also reviewing the projects that are being worked upon. At the technical level the structure is based on the set of current projects, with requirements being set for how these should be managed and how the technical work for them should be carried out.

The company has a number of groups: marketing, systems administration (the company has its own premises and networks running an open source environment), research and development and around seven project groups each with one or more clients or clients. R&D builds and consolidates the company infrastructure; new testing and management tools are built for use by the project groups. We have been awarded a grant by IBM to develop the Eclipse architecture for agile development methodologies, particularly XP. R&D also carry out QA on the individual project teams. PT

TURKEY • GREECE • CROATIA • ITALY • FRANCE • BALEARICS • CARIBBEAN • SEYCHELLES • WHITSUNDAYS • FRENCH POLYNESIA



Specialists in top quality yacht charter

BAREBOAT, SKIPPED, FULLY CREWED AND GULET HOLIDAYS

Tel 01243 520 950

or email sailing@top-yacht.com
www.top-yacht.com

MENTION
Professional Tester
FOR A SPECIAL
DISCOUNT!

