How to manage a conference special issue

STVR publishes a small number of special issues each year and has associated policies. However, simply following these policies/requirements does not ensure that all will run smoothly. In this editorial, I will focus on a very particular type of special issue: those that contain extended versions of conference papers.

The editors of a special issue have many responsibilities. The first is to determine which authors should be invited to submit extended papers. This is not as simple as looking at the review scores from the conference and choosing the top few papers. For example, I have seen strong papers that have excellent reviews but that also are 'complete' and so there is little scope to extend the work. Editors also have to consider the quality of the reviews, the contents of the reviewer comments, and also read the paper. Only then are they able to make a good assessment.

The next stage of the process involves inviting authors and then advising those who accept the invitation. An extended paper should have significant additional content and not, for example, be the union of conference papers. It is also normal for an extended paper to have a different abstract and to carefully explain how the paper extends the original conference paper; I find that it is often best to place such an explanation in the introduction. Reviewers will be expected to comment on whether the extension is sufficient and authors should be encouraged to make it easy for the reviewers to make this assessment.

The editors also have to choose reviewers. Ideally, this is done well in advance of the papers being received, thus limiting the scope for delays. We normally advise that, where possible, each paper should have one of the original reviewers and at least one reviewer who did not review the conference paper. In choosing one of the original reviewers it is good to look at the quality of the reviews and the confidence scores. I would also recommend that the initial list of possible reviewers includes more than the minimum (three reviewers per paper); in reality, not all those invited to review papers will accept.

Once the reviews are in, the editors make recommendations (accept, minor revisions, major revisions, or reject). The recommendation should normally be made as soon as possible - there is usually no reason to make the authors wait longer than necessary. Making prompt recommendations can help speed up the overall process since, for example, reviews for one paper might arrive quite early but the authors might take longer to revise the paper. I certainly would advise against waiting for all of the reviews of all of the papers. The recommendations/decisions should be independent - we might have an expected number of papers for a special issue but a paper will not be rejected because there is 'no room'.

As we have seen, there are many things to consider when organising a special issue - the above simply describes a few of these. However, there should be real satisfaction in seeing the final issue and hopefully this makes the work worthwhile for the editors as well as the authors and readers.

Now to this issue of STVR, which contains two papers. Both take a general area (testing, model checking) and explore this within a particular context. The first paper is "Pattern-Based GUI Testing: Bridging the Gap Between Design & Quality Assurance" by Moreira, Rodrigo; Paiva, Ana; Nabuco, Miguel; Memon, Atif (recommended by Le Traon). The aim of the paper is to develop new testing techniques, for graphical user interfaces, that take advantage of classic generic solutions (patterns). The authors thus develop a specialised testing technique that takes advantage of such generic solutions having been used. Essentially, this is a model based testing technique that incorporates reuse around the test patterns identified. The work was evaluated through a case study with real faults and a second study that used seeded faults.

The second paper is "Bounded Model Checking of C++ Programs based on the Qt Cross-Platform Framework" by Rodrigues Monteiro Sousa, Felipe; Praia Garcia, Mário Angel; Cordeiro, Lucas; Batista de Lima Filho, Eddie (recommended by Müller). This looks at the problem of verifying embedded systems where the software has been developed using a cross-platform software development framework (Qt) and the C++ programming language. Essentially, the authors modelled the Qt libraries. The approach was evaluated using a benchmark with 711 C++/Qt programs (many extracted from Qt documentation) plus two real applications.

Robert M. Hierons