Conference Report for APSEC04

Carl Cook

1 Preliminaries

The conference this year is held at Haeundae Beach, Busan, South Korea. Haeundae Beach is a tourist resort in the south east of the Korean peninsular, and even in winter (as it is now), it is a very busy place. The conference venue is the Grand Haeundae Hotel, which has a swimming pool, bowling alley, and golf driving range inside the complex. It also had a discounted rate of about $200 a night for APSEC members, so I stayed at another hotel around the corner for quarter of the price.

2 Tuesday, 30th November

2.1 Welcome Reception

The welcome reception was exactly the same as the previous year. It was here I met the usual people from University of Queensland, University of NSW, etc. It was particularly useful to meet these people, as they had already discovered the best places to sit and drink beer in and around Haeundae.

3 Wednesday, 1st December

3.1 Keynote Speech 1

"Real Time Software Design for Embedded Systems" by Professor Hassan Gomaa.

This talk covered the tools used to design embedded systems. It was very much a history of UML, UML 2.0, etc. The talk was based around "product line engineering" which seems just like software reuse to me.

3.2 Session 1

"Architectural Modelling for Self-Adaptation"

This paper talked about applications that provide middleware frameworks with metadata so that the frameworks can adjust to meet specific performance requirements. This appears to be fairly important to middleware frameworks such as J2EE and IBM's websphere.

"Evaluation of Architectural Alternatives for J2EE and Web Services"
This was an excellent presentation by Ian Gorton from ICT, Australia. The presentation covered synchronous and asynchronous modes of communications between a client and a web-service based server, and a well performed evaluation of the performance of the system under various configurations was detailed.

3.3 Session 2

"How to Have a Successful Free Software Project"
This was an interesting talk about the merits of open source software and how to manage it. It discussed the different types of developers, type of programs, and management styles. I feel that the authors have not done a thorough literature search, as whist this paper makes an interesting commentary on open source software, I'm all of the content and more has been published elsewhere. The section on the "ideal open source software license", however, was worth publishing at APSEC in my opinion.

"A Case Study in Specification and Implementation Testing"
This paper discussed a framework and tool support for testing and formal specification of functional requirements. To make this paper interesting, they attempted to test the software not only via the system's API, but via the GUI as well. They conclude, fairly obviously, that testing via the GUI is restrictive and limited, but the methodology behind the framework still allowed flaws to be detected.

3.4 Session 3

This was the session for my topic "CSCW/Distributed and Parallel SE". The other two papers in the session were relevant for me, and have been added to my further reading pile.

"An Exploratory Study of Groupware Support for Distributed Software Architecture Evaluation Process"
This paper presented a study of groupware support for distributed software development. The context was the evaluation of software architectures (a fairly subjective task). A pilot study was undertaken, with power analysis to validate the study — a fantastic idea for SE research. Software architecture evaluations are normally face to face, but the study found that users actually preferred distributed meetings, as it eliminated strong personalities and provided concurrent input facilities.

"Towards Collaborative Software Engineering"

The paper went down well. I fielded a few good questions afterwards, but the attendance to this session was very poor due to the very long schedule of the day and the timing of my talk. I have, however, had many good discussions with researchers during coffee breaks, lunch, etc, which has been the most valuable aspect of the conference to date.

"A Dual-Mode Exerciser for a Collaborative Computing Environment"
I'm not sure about the validity of the presented research in this paper. It describes a framework for collaborative software engineering, with support for both coding and runtime inspection. The architecture seems fine, but the evaluation is rather weak, and the detail about the types of tool support is also limited. Interestingly enough, they use an application sharing approach for CSE which appears to work well, but also places limits on the types of tools that can join the environment. They claim the system is mainly for educational use, and I suspect that this might be as far as the system can go.

4 Thursday, 2nd December

4.1 Session 4

"Assigning Tasks in a 24 Hour Software Development Model"
This paper discussed a scheduling algorithm for assigning tasks within a software development project, where the team is distributed across time zones. I only caught the end of this talk, but the concept seemed interesting, and the paper presented several simulations and empirical evaluations of the proposed method. I suspect that in practice such algorithms might be too academic for real SE use, but the concept is interesting.

"A Project Management Support Tool Using Communication for Agile Software Development"
I was pretty disappointed with this talk. The authors suggest that the state of a project can be predicted by observing the communication and bug logs of a project. They have come up with a metric to predict the future state of projects (danger!), and claim that the metrics correlate highly to the number of bug counts (which apparently is a predictor of quality). The user study was badly designed, executed, and reported, and the analysis related to open source statistics was also poorly done. I spent the entire talk finding faults in the paper.

4.2 Session 5

"Requirements Engineering for e-Business Systems"
This was a paper on trying to derive the requirements for e-Business systems via traditional Software Engineering requirements analysis techniques. It was all a bit over my head really—I don't know enough about the business processes

domain.

"Accurate Call Graph Extractions"

This was a surprisingly interesting paper. It talked about type signature filtering for languages that use function pointers. The point of this paper was to propose a way to build call graphs for program analysis. This is very similar research to Wal's work, and I talked about the similarities with the author afterwards. The system presented in this paper performs a fast analysis of function calls (including calls via function pointers), and then filters the call set by basic rules of the languages. The paper then did some analysis on different programs and four call-graph generation algorithms.

"Binary Level Date Integration to Develop Program Understanding Tools"

This was another paper talking about the impreciseness of the C programming language and the difficulty of resolving symbols. This paper suggests that full parsing (as opposed to partial parsing) is the way to do, but function pointers, assembly code, etc, causes problems. Therefore, this paper presents a system called DwarfRX to figure out the symbol table and references more accurately. In a way it was kind of cool; it had a Javadoc like tool that showed the fully expanded code, op-codes, and symbol table. It also has an XML interface.

The number of papers that are based on figuring out how to correctly analyse source code is really interesting, and the more I think about it, the more I am confident that Wal's approach is far superior to any other research I have come across.

4.3 Conference Excursion to Silla Dynasty, Gyeongju

For the conference excursion, we spent four hours on a bus and about three minutes looking around a Buddist Temple. We got to see a bit of the Korean countryside, so I guess that was a good thing.

4.4 Conference Dinner (at the conference hotel)

The dinner was great. We eat a lot of food, plus they put a bottle of whiskey on every table in the room. That probably explains the headache this morning.

5 Friday, 3rd December

5.1 Keynote Speech 2

"Improving the Quality of Deployed Systems" by Professor Mary Jean Harrold

This was a well presented talk. Professor Mary talked about the "gamma approach", where programs gather information about their execution to provide feedback to their developers. The presentation discussed how to classify and recognise user and program behaviour based from such feedback. The system executes tests over a range of programs, and figures out which lines are likely to have errors in them. It visualises such errors by highlighting the likely erroneous lines of code within a "code age" like editor.

During lunch, I managed to meet with Professor Harrold (Mary). It turns out that she is interested in visualising her data in more detail, so I suggested that we take some of her data from the Gamma project and process it via Neville's visualisation pipeline. She was very interested in our research at Canterbury, and suggested that I try publishing at the 3rd Annual Software Visualisation conference, which seems to be a rather relevant place for a lot of our work. We exchanged cards, and it will be good to keep in contact with Mary. We both observed that we produce very similar code editors and visualisations of software data, but we are doing so for different aspects of software engineering.

5.2 Session 6

"Performing High Efficiency Source Code Static Analysis"

This paper was based from industry experiences in preventing software defects within Cisco routers. The system, SCAS, is a source code analysis system. This paper is relevant to Wal's research, as it is another system which attempts to model software for analysis. It is also very relevant to my system, as it supports the sharing of source files amongst several users, and incorporates the bug tracking database, etc. An interesting aspect of the system is that it performs filtering of "invalid warnings" — responses generated by the system that are not appropriate for a particular developer. Another interesting feature is that users can also add watches for particular "dangerous" code structures, such as breaks after a return statement. This is specified via regular expressions. The idea of doing code analysis via regular expressions isn't great, but the system as a whole seems quite cool, and it is similar to my own work. It is also good to see that this tool is used in industry by senior software engineers at Cisco in fact it appears to be a compulsory tool for all Cisco software releases. A final point is that if users flag a new defect, analysis is performed instantly and a bug report with some genuinely useful information is generated. The research component of this paper was based around the savings made by filtering false warnings of bug reports.

I managed to talk to the author after this paper. We had a great discussion and exchanged cards. I was very interested in the overlap between his system and mine, and he was interested in the general concept of my system. He is going to describe the system to some people at Cisco in the US, and see what they think of the idea. I will also email him a couple of papers on JST, as I think that his research could benefit from the use of a full semantic model of software, such as our system provides.

"Supporting Knowledge Collaboration in Software Development"

I missed this talk because I was talking with the presenter from another session. But, from reading the paper, I realised that this paper is very relevant to my research. This is research done by one of Gerhard Fischer's students (from what I can tell), in which they define a new term — "dynamic community". This term defines a field where knowledge is obtained from traditional knowledge management systems, as well as from software sources and inter-programmer communication as the system is being developed. The system works as an agent behind standard text editors, and determines whenever a code comment has been inserted into a source file. Through pattern matching and index searching, programmers can query the doc comments of a method, search for methods, usages of the method elsewhere, etc. It is interesting to see that they are only performing pattern matching, because for object oriented code, only a certain amount of information can be derived from the source code at a lexical level. In a way, this paper has the same research goals of our own work. The main difference, however, is that this paper is very heavy on theory and formal justification, whereas we are more interested in producing new software tools that are realistic to software engineering in the large.

5.3 Session 7

"Use Case Refactoring: A Tool and Case Study"

The tool presented in this paper allows one to regenerate use case diagrams by modelling the software and then applying the refactoring changes. The talk was very hard to understand, and I'm not convinced that there is much of a need for this sort of technology. Perhaps there is, but the purpose needs to be demonstrated better. The paper presented a few case studies, but these were again less than convincing.

"Patterns for Requirements Engineering"

This paper introduced a new framework for describing patterns related to requirements engineering. It was quite hard to extract useful information from the talk, but I thought I should attend this presentation because I am considering designing some new patterns for collaborative software engineering and wanted to see how others go about publishing new patterns. I don't think I will use this author's approach, as it seems a bit confusing and very much an overkill. I will probably read his paper at some stage just to see if I can get some useful pointers out of it (the suggested framework looks like it might have some potential). Also, it was the last paper of the conference, and I couldn't really justify sneaking away this time.

5.4 Other Sessions

I also attended a few more talks other than the ones listed here that looked potentially useful, but they turned out to be less than relevant to my research, even in an optimistic frame of mine.

5.5 Conference Closing Address

In the tradition of APSEC, there was no closing address. After the last session, everybody just goes home. Most odd. I have two more days to spend in Korea before I get to catch my flight home (thanks to my friends at the university travel agency). I think I might go for a swim tomorrow at the beach it is winter, but I think it will be okay. I will also check out a few of the sights around Busan, because the planned excursion was a bit of a flop.

6 Final Comments

This year's APSEC conference again proved to be highly valuable to me as a researcher. Whilst not all the papers were first class, there were many researchers at the conference who I had very interesting discussions with, both about my research and other more general topics. Again I found that this conference helped me renew /refresh my interest in SE research, and helped me to put my little corner of research into the context of SE as a whole.

Many of the presentations that I attended were of great interest to me, and I also came across several papers in the proceedings (in which I could not attend) to be very useful; these have been added to my reading list. By talking with others, and reading the papers, I have also come across a few more research projects useful as reference material to my area of work.

As a final word, I noticed that my tolerance for sitting through research presentations is increasing. I put this down to one of two things: either I am becoming more knowledgeable about software engineering in general, or I am getting old and can fall asleep with my eyes open. I'm not sure what it is, but this is a useful skill to have.

My apologies for the poor standard of English, but I am finding that my grammar over the week has slowly diverged into a term I call "Korenglish".