– ICSE-18 WOW! –

18th International Conference on Software Engineering

Window on the World

A Fraunhofer ISST Sponsored Publication **Volume 2 Number 1** March 27, 1996

Parnas Ignites ICSE-18 Debate

By Bashar Nuseibeh

In a commentary on software engineering research and ICSE programmes, Prof. David Parnas prompted heated responses from ICSE organisers. Parnas provocatively asserted that most software engineering research had little industrial relevance, and that the apparent separation of researchers from developers in the ICSE-18 programme exacerbated this problem. Programme co-chairs, Profs. Tom Maibaum and Marvin Zelkowitz, chose to provide separate responses to Parnas - written with varying degrees of indignation. ICSE 97 organisers also chose to respond, outlining their plans for next year's event in Boston.

From an organisational perspective, the registration numbers for ICSE-18 provide happy reading to General Chair, Prof. H. Dieter Rombach, and his team. Rombach reported over 800 pre-conference registrations for the main conference and associated workshops and tutorials. These represent a significant increase from previous years, and are expected to rise with on-site registrations today. If such numbers provide any measure of interest or success, then ICSE-18 organisers should give each other a pat on the back.

Parnas' article and the responses to it are included in this issue of the newsletter. Readers are encouraged to send comments and contributions to the editor (icsewow@cs.tu-berlin.de). A dedicated workstation for this purpose is available in the terminal room. The web homepage of the newsletter also contains sneak previews of tomorrow's articles. Have a peek and let us know what you think!

CONTENTS

ICSE: Separating Research from Development?

By David Parnas

Once again we have an ICSE in which the overwhelming majority of the papers are by academics, many of the "industrial" papers are by researchers who happen to have industrial jobs, and very few papers are by actual system developers. Even worse, many of the industrial papers are in segregated sessions. Are we trying to keep researchers and developers apart?

In the "classical" engineering fields, researchers find their problems in industrial development, then retreat to research labs to find solid solutions. In Software Engineering we see that researchers often find their problems in papers by other researchers, rather than in developers' problems. Further, when researchers look at development groups, they focus on the most advanced, and ignore the quiet majority. As a result, we see a growing gap between the "real world" of software development, and the researchers who report results at ICSE and in IEEE TSE. Increasingly, industry finds ICSE and IEEE TSE irrelevant and must be invited to special sessions. In many places, software is written much as it was written 30 years ago.

About a year ago, one of the most prestigious research groups on programming methods was discussing a workshop. I proposed that this be a real *work*shop in which we would invite someone with a real program and all of us would try our methods on the same real program. This proposal was met with almost complete silence. Eventually, I received the following response, "First, I don't want to look at anyone's 'program' -

continued on page 2

■

CHECK OUT the *Window on the World* on the **World Wide Web: http://www.cs.tu-berlin.de/~icsewow**

I don't mind getting involved in exercises that investigate design histories (incl. programs) - but I learned long ago that post-facto analysis of programs is not very productive".

Long ago, I learned quite the opposite. In 1969, working with developers who were product-oriented rather than method-oriented, I learned that I had been working on the wrong problems. I found that the real problems were far more interesting, and often more tractable, than the problems discussed in the research literature.

Unfortunately, working with industry has become harder. Whereas 25 years ago, industries were eager to have researchers work in their development groups, the growing gap between research and development has made us less welcome. Developers, always worried about the next deadline, assume that academic research is going to be irrelevant and impractical. They expect researchers to be unresponsive to their needs. Often, when we get a chance to work with them, we confirm their prejudice.

ICSE should become a conference where the academic papers and industrial papers are mixed and mutually relevant. It should also become a meeting ground for researchers who want to work with real programs and developers who still have some hope that researchers will say something relevant. For their part, researchers must recognise that to develop useful tools and methods they must work with real problems. On the other side, developers must recognise that, if they wish to exploit research results, they will have to change their work habits.

Small Ads

Response to Parnas (1)

By Marvin Zelkowitz (ICSE-18 Programme Co-Chair)

David Parnas has the uncanny ability to say great truths in a very antagonistic manner. My only disagreement with him is his comments on ICSE-18. The separation of researchers from practitioners *is* a concern, and we made a significant attempt to avoid such problems in this Berlin meeting.

When we classified the 52 papers into 16 sessions, our only concern was to group papers that would appeal to a cohesive segment of the attendees, not to "researcher" or to "programmer" sessions. As such, we had 9 sessions with a mixture of industrial and university research papers, 6 that are university research and only one that can be classified as industrial. 10% of the papers represent joint efforts by both groups.

That a large majority of the papers represent research papers (either by industry or university researchers) is just a simple fact of conference submission. The overwhelming majority of submitted papers comes from this environment. The Program Committee made an attempt to include less well-written industrial papers by "shepherding" them through the final writing cycle, if we believed they had something relevant to say.

The sad truth is that many of the submitted industrial papers were of the type that Parnas classified as "software ... written much as it was written 30 years ago." Presenting such papers does not help the field in presenting these ideas to the community at large. We hope that those authors are here this week to learn from the others of how to change their future practices.

Parnas is right in the segregation of the field into industrial and academic meetings. Much as physics has learned to tolerate and even respect the theoretical and applied branches of the discipline for mutual benefit, software engineering needs to continue to look at industry for problems and for the research establishment to propose solutions that can later be validated in industry. On this issue, I have no disagreement with David.

Response to Parnas (2)

By Tom Maibaum (ICSE-18 Programme Co-chair)

Unlike my co-chair Marv Zelkowitz, I do not feel that David Parnas "has the uncanny ability to say great truths in a very antagonistic manner." Rather, I think what he gives us are simple truths mixed with a large dollop of misconceptions, packaged in an antagonistic (and sometimes offensive) manner.

ICSE-18 Organisers: *(from left to right): Maibaum, Zelkowitz, Rombach, and Jähnichen*

Now that I have vented my spleen and done with being offensive myself, let me explain why I take exception to David's statement. At the first TAPSOFT (in Berlin!) in 1985, David gave an invited talk during which he attacked the theory community with the accusation that it had not come up with answers to his questions: What formalism should I use for X? What method should I use for Y? etc. I also recall Maurice Nivat providing a highly articulate response. This was based on an explanation of the relationship between science and engineering, the roles and world views of practitioners in these (intersecting) domains and the lengthy time scales over which movement of scientific knowledge into engineering practice took place.

David is singing the same tune again! Let me consider some of his assertions. "In the classical engineering fields, researchers find their problems in industrial development, then retreat to research labs to find solid solutions". This is patently untrue. If one walks around the research laboratories at Imperial College (and I daresay that the same is true of any engineering research establishment), one sees a combination of research projects, like the one described by David and other ones inspired by scientific development with potential applications to engineering. One sees lots of work "inspired" by having read the papers of other researchers. We do not hear the cries of 'irrelevant' from their lips. Quite the contrary, they have found this melange to be somewhat efficacious in their fields. It is just as legitimate to find one's problems in others' research papers as in industrial practice. After all, the main purpose of research is to create concepts / abstractions which are the essence of something 'real' (whether in science or engineering practice), so as to study them, and certainly, to communicate them.

There is a natural reason to focus on "advanced" development groups as the reason they are considered advanced is that they have (partly) articulated their problems and perspectives, thus being able to communicate them more easily. "As a result, we see a growing gap between the 'real world' of software development, and the researchers who reports results at ICSE and in IEEE TSE." David should not insult our intelligence by making this kind of simplistic connection between the phenomena.

My view is that life and truth are far more complicated than the simplistic view presented by David. There is a need to promote research at all parts of a broad spectrum ranging from pure science to the "archaeology of software" proposed by David. And what is the role of ICSE? What should it do to promote a balanced view of the discipline? Firstly, I think ICSE is a scientific meeting. Not because there is only science, but because we can only communicate effectively in the abstractions and concepts of science. We should have contributions (papers, special presentations, tools, demos,...) from industrial participants but we should demand more from a paper than what I characterise as the 'Veni, vidi, vici' message (and we should make the same demand of 'research' papers!). After all, the purpose of the meeting is to trade in exchangeable goods, i.e. concepts and abstractions, and not make the communication biased in one direction.

What is patently the case, is that we do not yet have this 'balance' (this is probably not that different from other engineering disciplines - but does not excuse us from trying). Many of the discussions about the future of ICSE have focussed on how to overcome this imbalance. I hope we have taken some steps in this direction, but it is up to you to judge to what extent we have succeeded.

Response to Parnas (3)

By Rick Adrion, Alfonso Fuggetta, Dick Taylor, Tony Wasserman (ICSE '97)

We agree with many of David Parnas' observations about papers and participation in ICSE. Unfortunately, few people involved in the day-to-day development of commercial software products write and submit papers to ICSE. In general, they have no incentive to do so, since their companies do not encourage or reward such publication. Furthermore, writing such papers takes considerable effort and the chances of acceptance are disappointingly low, since only about 15% of submissions are accepted.

The absence of papers describing "practical" software development also affects attendance. Potential industrial attendees see a conference program dominated by researchers

ICSE-97 organisers: *(from left to right): Adrion, Wasserman, Taylor and Fuggetta.*

and elect to use their limited budgets to attend conferences with a more apparent short-term practical benefit. Conversely, the research community is noticeably absent from major commercially-sponsored conferences such as UniForum and Object World, which often highlight important new products and technologies.

With these concerns in mind, we are making significant changes for ICSE '97. We have been strongly influenced by the SIGGRAPH, SIGCHI, and OOPSLA conferences, each of which brings together a broad community of interest. While you will still see the traditional carefully reviewed papers, you will also see that they are blended into an overall program that includes numerous invited technical presentations from people involved in "real" software development. We have an industry-based committee to address the needs of practicing professionals, but would welcome additional suggestions for speakers and topics.

ICSE '97 will also have an expanded set of tutorials and workshops, covering topics of interest to both researchers and SE professionals. We strongly encourage you to propose a workshop. Finally, we are planning a commercial exhibition and hope that many software engineering tools vendors will participate. Each of these steps is intended to provide a setting that will be equally attractive to researchers and to the professional software development community.

It is our hope that the ICSE series can be recognised as an annual "must attend" event meeting for software engineering professionals, whether their interest be formal methods, graphical user interfaces, embedded systems, tools, architecture, or any other topic related to the organised development of software.

Small Ads

Agents of Change

By David Notkin

One of the primary roles of ICSE is to give practitioners and researchers a chance to meet with each other. The intent, of course, is twofold. The researchers are provided with a chance to learn from practitioners both about problems faced in the engineering of real software systems and also about how well (or poorly) various approaches have performed in practice. That is, the practitioners provide both problems and data. The practitioners, on the other hand, are provided with a chance to learn about new technologies and approaches to solving their problems.

ICSE has some success in providing an environment in which these two groups (and others, including educators) can meet. But like many chemical reactions, simply mixing the two groups is often not enough. Sometimes you also need heat or enzymes or some additional agent to cause a true intermingling of the parts. Unfortunately, it is harder to describe and to understand what that added agent is in software engineering than it is in chemistry.

The world of most practitioners has changed recently: the ever-increasing focus on short-term results is widely reported. Similarly, the world of most researchers has changed: conventional funding structures are rapidly disappearing, with a similar focus on short-term relevance, as opposed to basic research, appearing from almost every side. These changes make it more critical for practitioners and researchers to interact. They also place enormous pressure on any interactions to succeed very quickly; as we all know, much research has high potential payoff, but with commensurate risk. Short-term pressures discourage this. In any case, a key piece of the needed chemical agent between practitioners and researchers is a genuine desire to interact. It's difficult, time-consuming, not well-prescribed, and pressured. So unless both sides are knowledgeable, eager, and willing to work despite the problems, such interactions are likely to fail.

David Notkin: *ICSE Steering c'ttee chair*

Another key part of the chemical agent is money. Despite all the hype about the Internet and the WWW, it takes time for practitioners and researchers to interact. And as the old adage goes, time is money. Asking a practitioner to "simply" gather data is asking a lot: massaging it into a useful form is part of that time, but this is probably dominated by the time to get the distribution of the data cleared by management. Asking a researcher to "simply" provide a distribution of tools may be asking a lot: are there licensing issues? are there international export issues? how are various platforms handled? how are later bug reports handled?

Having had some experience with industry interaction, I can attest both to the massive benefits of doing so, as well as the costs. It requires enormous efforts by both sides, as well as resources to support the efforts. If ICSE is to succeed, industry-academia partnerships must be formed and exploited to solve the most important and perhaps the hardest problems in software engineering. We cannot be blind to the difficulties and the costs. But, as Grace Hopper used to say, we have to do it: the costs of not doing it far outweigh the costs of doing it.

Let's get real!

By Ian Sommerville

Why do software engineering researchers ignore reality when they are devising new techniques? If we want our research to make an impact, we have to think a lot more about how to integrate it with practice rather than focusing on new, complete, consistent, elegant but practically useless methods and tools.

Many of us work in universities. These are normally driven with political disputes, suffer what seems to be perpetual financial crises, have incompatible equipment and lots of legacy software. We all know the problems of convincing our colleagues to use new software and, if it wasn't for Microsoft, we'd still be using about a hundred different word processor systems.

All large organisations suffer from these problems: politics, economics and simple human cussedness dominate progress everywhere. We never seem to think about this when we moan about the slow rate of technology transfer from research to practice. Perhaps we think our ideas are so good that they will transcend these sordid realities. Managers should beat a path to our door pleading for our research results.

Of course, they don't and they won't. Managers are too busy surviving. They are only interested in new ideas if they don't pose significant risks, don't cost too much to develop and can be introduced incrementally, without organisational change. Sadly, there isn’t much software engineering research (and I'm guilty too) that recognises these realities.

Of course, we could forget all this reality stuff and decide that research is about increasing awareness and long-term technology diffusion. Our students will go out and spread the word on new techniques. This works! After 30 years, some people are even using OO techniques, although it's still a pretty small minority.

Alternatively, we can accept that industry needs 'lightweight' technology that it can run in slowly alongside existing systems. This needs to be standards-based, low-risk and must not require any kind of revolutionary change. We probably need to be pretty informal and not worry too much about whether our methods are complete and consistent. Perfection isn't necessary so long as we can fit in with the organisation and its ways of working. Time is the scarcest commodity so lightweight technology has to have a shallow learning curve and be easy to use.

If we do not do this, we researchers do not have much of a future. No-one could possibly argue that the millions of dollars spent on software engineering research over the past 30 years have generated a worthwhile return. Unless we take technology transfer seriously there will not be any more funding for research. We will have to change our ways of thinking, working, writing, and reviewing. Can we do it?

From the titles of the papers in the Proceedings here, it looks to me that we have a long way to go!

Humphrey on Top!

By Wolfgang Emmerich

***Watts Humphrey**: Over the moon!*

Over 350 delegates attended the 14 Tutorials held on Monday and Tuesday. Top attendance figures (59 delegates) went to Watts Humphrey speaking on "the personal process in software engineering". According to Humphrey, PSP combined with the CMM provides organisations with highly efficient control over their software processes.

Our metrics also reveal that Vic Basili and Frank McGarry were runners-up with 56 delegates attending their tutorial on "the experience factory: how to build and run one". Basili suggested that "the attendance level was due to an interest in alternate approaches on the part of the software community to improving software in their organisation". He then went on to say that "part of the interest was in hearing the interaction between theory and practice, the application of the approach to a real organisation, and the complementary perspectives of industry and university".

Proponents of software process technology will be interested to learn that Bill Curtis' tutorial on the methods and lessons learned on software process improvement attracted 52 delegates. This brought the number of attendees of process-related tutorials to over 160. Care to comment?

The tutorial speaker of the lowest attended tutorial, however, was unavailable for a comment ...

Multimedia Workshop

By Stephen Morris

The first International Workshop on Multimedia Software Development (MMSD'96) fed us a solid mixture of technical talks and discussion covering a wide range of topics, beginning with a long session about possible architectures for distributed multimedia systems and ending with a demo of a security enhanced MPEG player.

What made it an especially memorable event was the keynote talk given by Peter Krieg, Head of the new High Tech Center at the Babelsberg Film Studios. The first studios ever established in Europe, they were once in the forefront of film production; they intend to regain that position with the use of the most advanced technology.

Having disclaimed any technical expertise, Herr Krieg, once a producer of documentaries, gave an overview of digital Hollwood that would put most to shame. On the way he gave us some home truths about what film makers want: not MPEG because it loses too much information, not ATM because it does not provide enough bandwidth, but yes to lossless compression, very wide bandwidth distribution and image recognition techniques.

Many thanks to Max Muelhaeuser and Wolfgang Effelsberg, the Co-Chairs.

Food for Thought

By Will Tracz

Decisions! Decisions! Decisions!

ICSE-18 attendees probably have a hard enough time choosing between what tutorial, workshop, or session to attend, but if they still feel up to a challenge, they can go to KaDeWe the "Department Store of Limitless Possibilities," where they can choose from more than 300,000 different articles in over 60,000 square metres of sales floor space. (That is the size of an Olympic Stadium -- plus three football stadiums as well!). For the cultivated palate, a legendary epicurean paradise awaits on the 6th floor.

Covering 5,800 square metres it is the largest food department in Europe (and second largest in the world). Here shoppers can choose from over 30,000 edible items, including 400 different kinds of bread, 1,800 different kinds of cheese, 1,500 varieties of sausages and cold cuts, up to 15 types of living fish, 120 different kinds of oil and vineger, over 100 types of tea, not to mention a wine-cellar stock of varieties worth over 1.6 million DM. Of course, one can't help but notice the usual range of meats, (charolais from France, angus from Aberdeen, filet and rumpsteak from Argentina, and entrecote from Ireland, etc.) complimented with deer, stag, wild boar, rabbit, elk, reindeer, wild duck, pheasant, quail, and partridge. For the health-conscious customer, there is a large selection of fruits and vegetables from all over the world, as well as a large range of spices, sauces, and ingredients from Japan, Indonesia, China, Taiwan, Thailand, and India.

KaDeWe is located a brisk 15 minute walk (longer if you don't catch the lights) from the conference hotel at Tauentzienstr. 21-24 (two blocks past the Kaiser-Wilhelm Church and Europa Center). If you go there hungry, there are more than 28 snack bars to eat at. The dishes they serve range from beer and bulette to champagne and lobster or vodka and caviar (as well as sushi and tea). If you don't go there hungry, there are five other floors of merchandise (plus the Technical Center in the Basement where the computers are) to help you work up an appetite.

ICSE-18 Window on the World

Editor-in-Chief: Bashar Nuseibeh
(Imperial College London)

Reporters:
Boris Bokowski (FU Berlin)
Steve Easterbrook (NASA/WVU)
Wolfgang Emmerich (City Univ, London)
Christian Kanele (FU Berlin)
Jyrki Kontio (Univ. of Maryland)
Stefan Tai (TU Berlin)
Scott Tilley (SEI)
Will Tracz (Loral Federal Systems)
Ken Wong (Univ. of Victoria)

Photographer: Keng Ng (Imperial College)

Technical Support: Sabine Lembke, Lutz Friedel and **Stephanos Mantoulidis** (TU Berlin)

Technical Coordination: Ingo Claßen (Fraunhofer ISST, TU Berlin)

Sightseeing in Berlin

By Will Tracz

The two hour Berlin City tour is a must-do for those ICSE attendees who may be visiting Berlin for the first-time and only have a short-time to spend. The tour route travels past the major historical monuments, palaces, churches, and memorials. It includes a 15 minute stop at the Brandenburg gate, where you can haggle with local street vendors for momentos left over pre-unification days. I personally found the remnants of the Berlin Wall and Checkpoint Charlie leaving the biggest impression.

The tour is offered in eight different languages (you listen to a tape on comfortable headsets). If you go, try to sit on the left hand side of the top of the bus, if you cannot get in the first three front seats.

Small Ads

Wanted: Software Engineers. Programmers need not apply. Box 1.

Contest: Win a bottle of champagne

The WOW newsletter is running a contest to select the most bizarre title of a paper that could potentially be submitted to ICSE ten years from now. So what's going to be the latest software craze of 2006? Enter now and enter often! Write the title, your name and contact details on a piece of paper and drop it off in the contest box at the registration area. Or just send us an email. The winner will be announced in Friday's issue.

Here are some titles. They're terrible! Think of something better!

- Thermodynamics and the heat death of legacy software systems.
- Gourmet recipe for component selection: where's the beef?
- Plunger strategies for unclogging software bottlenecks.

■