Targetted Improvements

Focusing improvement to maximize benefits and minimize impact

Andre Oboler & Simon Lock
Computing Department
Lancaster University
Lancaster, UK
Oboler@comp.lancs.ac.uk, Lock@comp.lancs.ac.uk

Ian Sommerville
School of Computer Science
St Andrews University
St Andrews, UK
ifs@dcs.st-and.ac.uk

***Abstract*— In the creative environment where research takes place not everything can be improved. The creative "essence" of research must be undisturbed while "accident" wasted effort is reduced to a minimum. In this paper we discuss the types of knowledge at play in the research environment, introduce a new abstract model of knowledge, and using the model explain how we should focus our effort on research students and on particular types of knowledge transfer in order to gain an over all improvement in our research processes. Just as we teach to facilitate student learning, so too can we supervise, teach and guide to facilitate better and faster researching in our academic computer science departments.**

Computer science education, process improvement, knowledge management, research environment

I. INTRODUCTION

With increased pressure on the research community a way is needed to improve efficiency and productivity. Researchers in Computer Science are in a prime position since research approaches to date have been ad hoc and thus finding improvement (any improvement) should be a simple matter. To cater for the wide breadth of work in the discipline we believe a high level abstraction of the problem and similarly high level meta solutions are needed at least initially. There will always be new research students constructing their approach to research for the first time. We must gain new insight into the research process and into the way researchers develop their approaches. With this knowledge we can facilitate improved research for them and greater and faster advances for the field.

Before we begin we must understand what we mean by research and what we consider an improvement in it. Improving research efficiency is an optimization problem under certain constraints. Should your definition of research or your measure of improvement differ from ours the approach we recommend may not be for you.

This paper speaks of research set within certain boundaries. Research is "original investigation undertaken in order to gain knowledge and understanding" [1] according to the UK's RAE 2001 guidelines designed to measure research and allocate funding accordingly. Put another way it is "creative work undertaken on a systematic basis in order to increase the stock of knowledge" according to an OECD definition and perhaps more importantly to the Higher Education Statistics Agency who quoted it [2, 3]. Computer Science Research is taken to mean research that focuses on the creation, adaptation or analysis of computer systems. We limit ourselves to the typical projects in the university research environment, those involving a student and a supervisor, with occasional input from colleges working in the same area. The typical computer science research project is an idea subject for this discussion as it is effectively a task of pure knowledge in various forms, from creative ideas, to instantiations of algorithms in code and advances in our understanding of the world through experimental results. The lessons drawn may well apply to a far wider areas of research, but we have not systematically examined this.

To begin, we divide the knowledge needed for research in our field (and presumably all others) into three types. All are required for research, yet only two of these types may be realistically improved. By creating this separation we invalidate the notion that research as a whole cannot be improved because it is *all* a creative process. Fredri

Having isolated the types of knowledge we can target for improvement, we discuss how knowledge can be abstractly measured. We present a model, drawing from it concrete ideas on improving efficiency, and explaining why current software engineering fails to meet the requirements. Finally we discuss one possible way of taking these ideas from abstraction to reality and implementing them in a real research environment.

II. TYPES OF KNOWLEDGE

Lord Kelvin [4] in 1883 said that "When you can measure what you are speaking about, and express it in numbers, you know something about it; but when you cannot measure it… it may be the beginning of knowledge, but you have scarcely, in your thoughts, advanced to the stage of science". We present here three types of knowledge followed by a conceptual model of research as an individual's path to new knowledge. As we try to model something as ethereal as "knowledge" we may not be at an advanced stage of science, but we have at least advanced to the beginning of knowledge, a definition, a model and an encompassing context for further work. Our work relates old concepts in philosophy (stretching back to biblical times) to the modern Information Science and Knowledge Management domains, and finally applies these concepts to the problem of research which is at its core no more than the

pursuit of knowledge. To improve the research processes capacity for knowledge generation it is vital we have an understanding of the various types of knowledge, who has them, how they can be created and transferred, and from this we can see how to aid the transfer and facilitate the generation of relevant knowledge while not expending effort to capture data that is superfluous to researchers needs.

In the old testament three types of knowledge are mentioned, Wisdom "chochma", Insight "bina" and Knowledge "daat". Rav Kook, an eminent Jewish scholar and philosopher explained the distinction saying that Wisdom provides the underlying framework that allows understanding. Insight is a vision of the future and how things may fit together, the Hebrew word Bina is related to the Hebrew word "boneh" meaning "to build". Knowledge (Daat) means a complete attention to detail, i.e. to the facts of the current issue [5]. This is summarized in Table 1. While this is not the only scheme for understanding and partitioning knowledge, it does provide a well establish and useful starting point for discussion.

Table 1 The Biblical System

Type	Explanation
Wisdom	underlying framework
Insight	vision of the future and path
Knowledge	attention to detail / facts

In a research setting one can expect a supervisor to have wisdom that the student must master; the student and supervisor together provide insight when they meet and discuss issues; the student alone will generally generate the knowledge that the research is based upon. As Lord Kelvin says, without this building process to test and prove our theories, through coding, experimentation and measurement, we have only the beginning of knowledge. Domain knowledge (which is really Wisdom, not knowledge in this frame work) and a new idea complete with theories on its use and relevance are useful insight, but are not enough to qualify as scientific thought. This is where the postgraduates come in with their knowledge generation and measurement, in the case of computer science usually through the creation of software to test the theories and the production and manipulation of data and information.

One organization that does teach scientists about different types of knowledge is the Intergovernmental Oceanographic Commission (IOC), a part of UNESCO. The IOC maintains a digital library as a training tool on Oceanographic Data and Information Exchange [6]. The libraries overview article on Information Technology and Scientific Communication by Jonathan Hey uses the Data, Information, Knowledge, Wisdom Chain (DIKW) to separate types of knowledge. Hey notes that "the concepts themselves, not to mention the transitions between them still resist clear definition" and proceeds in the paper to use metaphors from every day English to analyse and explain the differences between them. Early work on DIKW did not include data, and some researchers have included a higher stage of "understanding" [7] or "Enlightenment" [8]. The model is shown in chain form in Figure 1, taken from [9].

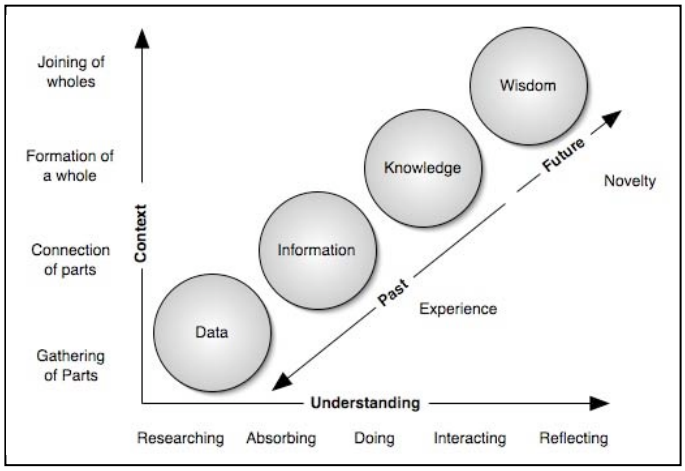

Figure 1 DIKW Chain (from [9])

Hey notes that "like data, we conceptualize Information As A manipulability object". This separates Data and Information from Knowledge and Wisdom. Based on metaphor usage of Data, Information and Knowledge Hey notes that "significantly, knowledge appears to be a quite different 'entity' to either information or data." He finds it metaphorically similar to a liquid or sticky substance that can't pinned down and notes its personal nature.

The DIKW chain is a useful partition of knowledge, and consistent with our previous model, but adding the "physical artifacts" of raw data and information. The DIKW model also adds a temporal relationship which we believe an over simplification. Data may be ordered to produce information, and any researcher with the same knowledge could do the same job of data to information conversion, but at the level of wisdom and insight the creativity of the researcher plays a role. Even at the level of knowledge ones wisdom and insight play a role in creating cognitive links which may differ for different people presented with the same information.

The lesson for those wishing to improve the computer science research experience is that the data to information stage should be as automated as possible. There is a difference between students gaining information that could be provided to them, and them gaining wisdom. Sometimes one must move from data to information in order to acquire wisdom, but this is not always needed and if kept to a minimum can greatly reduce wasted effort. For example, not all students understand that the process of coding standard algorithms at undergraduate level is about acquiring wisdom and once gained those algorithms should be reused rather than recoded. This needless recoding (seen in research students work) leads invariably to longer coding times and the introduction of avoidable bugs [10].

While Knowledge Management tries to treat everything as information, at the other extreme many have been of the opinion that "research is pure inspiration and therefore the research process cannot be improved", in our past work we have rejected this notion [11]. In previous work looking at academics working in computer science departments across Australia we have shown figures as high as 40% using an unplanned approach [11]. In a US sample, which was skewed (72%) towards software engineering lecturers, 20% still replied

that their approach was unplanned and a further 20% chose not to use any recognised software development life cycle but said they had another approach [11]. This shows that a systematic approach is not used by a large percent in both the over all population, and even amongst those who we would consider up-to-date and experts in the area there is deep reluctance. Of interest is that the sample bias occurred as a survey e-mailed to hundreds of computer science department heads in the US for distribution was in fact mostly passed to the departments software engineering lecture. This was confirmed not only by the data, but also by a number of e-mails back from heads of department helpfully informing us of this initiative. The bias was removed from the Australian sample by simply referring to software engineering in different terms. The compartmentalization of software engineering knowledge as something taught to students but not applied to computer science research gives some concern. Based on these results we reject the notion that supervisors work in a perfect way and students can learn through osmosis.

We have previously introduced an outline for a software development process (known as RAISER/RESET) designed for the research environment and argued that due to its specific needs, researchers require a different approach to that of industry [12]. The key reason for this is that researchers work not just with information (their observable output), but also with generating more knowledge, wisdom and insight about their topic, this in turns allows more data and information to be produced, but the information is a by-product in the pursuit of knowledge.

To further develop our approach we returned to the definition of research: "creative work undertaken on a systematic basis in order to increase the stock of knowledge" [2]. The task, to improve research, is to make it easier to be systematic without impacting on the creative process. This new work puts this in a context that allows further understanding and analysis. Using out model of knowledge we see that the application of insight (a creative process) cannot be considered as a process for improvement. We have already suggested automating as much of the data to information transfer process as possible. Looking at our model, we can add that knowledge generation and the transfer of wisdom are additional processes ready for systematic improvement in efficiency (time) and effectiveness (depth of understanding). We have created a model and an abstraction of a unit of measurement to further examine this concept.

III. Measuring knowledge

The Standard Knowledge Unit (SKU) is a hypothetical unit of measurement we introduce here as a metric for our model.

There is no way of measuring such a unit, but abstractly, a SKU is a finite amount of new and relevant knowledge. It is an additional step along the road from the knowledge you have now to the knowledge you will have in future. In Figure 2 we introduce the pipe model, a hypothetical model that allows us to apply a degree of abstract thinking to SKU optimisation.

The pipe model shows knowledge as steps of work measured in SKUs, thought with varying level of effort attached to each of them. The effort can be measured as the volume filling the pipe to move one SKU forward.

More formally we can adapt the definition of Work from physics. Imagine all effort is moving forward at an angle to the direct path, with some effort always wasted. This gives the rule that: The Work (W) done by a researcher when exerting a constant effort ($\vec{F}$) and gaining a fixed amount of useful knowledge ($\vec{s}$) equals the magnitude of the displacement, s (how much was learnt), times that component of $\vec{F}$ which is along the direction of $\vec{s}$ (i.e. increase in relevant knowledge). Put another way, work (W) is the part of the effort that is effective, multiplied by the time this effort is applied.

What we observe is that different stages of the research process have different amounts of wastage. By allowing researcher to focus more of their effort on knowledge generation and automating time consuming aspects we can increase research efficiency markedly.

We can also further develop the idea of knowledge being a series of steps to "right size" the research process. In addition to efficiency increases, we can begin further into the process (though better transfer of Wisdom), and we can end earlier by correctly identifying when enough steps have been made that the required contributed of new knowledge is assured. While some might argue that these steps only work for new researchers, as has previously pointed out, it is new researchers who are doing the bulk of the detailed knowledge work. A gain for these researchers is a gain for the entire knowledge system.

In addition, there seems to be a mythological assumption that more established academics know something the rest of us don't. Waite explained that "the major problem is that research projects tend to be opportunistic rather than planned" [13] therefore this can't be true. Bertrand Meyer [14] adds "I know about designing software, I don't know about designing research". An improved way for approaching research more systematically can potentially benefit even the experts.

IV. Change without damage

In today's academic environment there are pressures pushing for faster through-put of PhD students. These range from official pressures on supervisors to ensure students complete "on time" to students own demands as funding runs out. With students ever more the consumers of education, the student voice in particular is getting louder. How do we balance the demand for quality research with the push for quantity and rapid progression?

Prof Farr poses the problem when he remarks that he thought "there's a bit of a trend to turn research into a process that can be managed like an industrial process and I think that is really antithetical to the nature of research...we still need to recognise that it is at its heart a creative and somewhat unpredictable process" [15]. Pressman [16] however suggests the first step to a solution "the implication (to me) is that any SE approach for research software (that would have any chance of adoption) would have to be agile and evolutionary in nature". This is due to the fast pace of change in research as knowledge is acquired and plans adapted. In the case of

research students this is an accelerated process as students' underlying understand is still forming and greatly influenced by the work at hand.

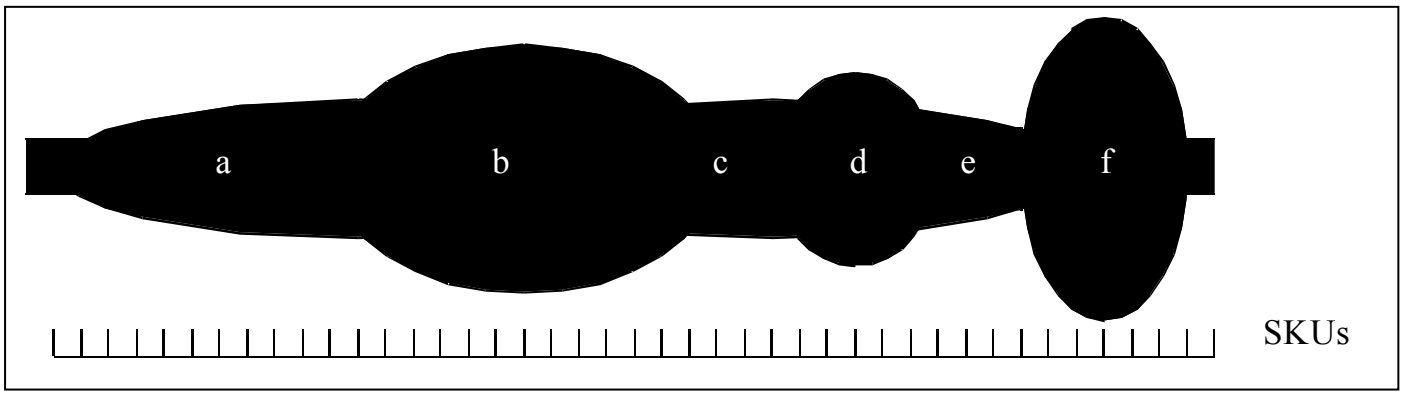

Figure 2 The effort vs. progress pipe Model

What we need is a way to improve the research process that does not damage the insight of the researcher nor devalue the wisdom of either them or their supervisor, nor limit creativity and innovation. By focusing on the knowledge transfer process, that is the way researchers (particularly students) gain their background understanding, we can add value to wisdom. By helping researchers record their insight we can share these visions with others and prevent a researcher losing touch with them unintentionally. By finding the right tools we can automate aspects of the research work. Finally, through a loose and adaptable planning tool we can make better use of time, allowing more consistent levels of productivity and greater quality in the final stages when time pressures inevitably start having their effects.

V. FACILITATING CHANGE

It's obvious that students still need to do the volume of the leg work. One major improvement would be to reduce their leg work. It would be even better, if we could give them a map and the tools to undertake (and document) their journey more efficiently. A journey with an initial plan and the tools to reduce effort is still a journey of discovery. It's just quicker, has fewer dead ends and greater thought before taking a diversion.

So how do we achieve this? What form should this map and tool set take? We suggest starting by asking researchers to plan their route. With a written plan, supervisors can highlight skills and tools students will need to know well in advance of the need arising. Relevant papers on the methodology may come to mind. This is currently done in some sense during the project proposal stage, however many proposals are (like other required documentation) written after the fact, that is after the direction has already been worked out and agreed. In other cases the proposal is very vague and completed early on. In both cases it tends to be a static document. Instead (or given external requirements, as well,) we propose that computer science researchers create their research plan as set of classes with comments explaining what each class does, what steps are planned and what tools and skills will be needed. This new idea is not the static research proposal many are used to, but rather a live "process plan".

The key difference between a proposal and a process plan is that the plan adapts and changes, at times on a daily basis. The plan can also be recorded as short hand notes, to be expanded upon later. There is no due date, no "finalizing" of the plan and no penalty for adapting the project as new information is learned.

A live document discusses the mechanics, approach, and questions that still need to be answered. We've reached this conclusion after three years of incremental improvement on a facilitation process for students undertaking Masters level research projects. Our conclusion is that research improvement can only come with self-reflection and a quality peer review process. It has much in common with more familiar facilitation processes such as that to improve teaching standards. Our approach integrated guidance and tools both for the job, and for reflection on it.

Originally we used a journal approach, later some of the process information was integrated into internal code comments, and finally the process plan idea took shape. In implementation, students in the final year of our three year experiment were given a "generic template" java source code file. The file contained a class called ResearchProcess and within this class, a list of subclasses was declared. Each subclass referred to a different aspect of the research process. The subclasses were:

- ProblemDefinition
- LiteratureReview
- MethodologyCreation
- ResultsCollection
- AnalysisOfResult
- DrawConculsions
- CreationOfOutput

Some of these contained further subclasses, and all were documented with questions about this stage to help get students started. Once basic information was recorded students were encouraged to adapt the model to suit their own purposes.

Three students used the process model, and did so very extensively. One of these in a post project interview described the benefit of the process model approach as "clarity, also in terms of neatness and presentation". Another student, when introduced to the process model saw immediate benefit, they explained, "I have a learning difficulty that effects my memory of sequential reasoning, which means that, if in 10 minutes time you meet me outside and ask to go through everything we've discussed today I couldn't do it, I have all the pieces but can't link them together" the student went on to say that the idea of a structured document for storing their ideas would help them over come this difficulty. After their project the same student commented, "although it was very handy and I got quite a lot out of it for the report, if I'd done that sooner I probably would have got a little bit extra in terms of planning".

A tool called dOxygen was used in conjunction with the process plan. The Open Source dOxygen is a package that can generate class diagrams from code and comments. We've increased its relevance by providing addition guidance on documenting code for research [17], providing process templates, and integrating a technical review into our facilitation process. We've also begun to use dOxygen along with the template as a meta-tool to graphically model students' process plans and generate the map mentioned above as a class diagram. Each time a student changes their plans it takes only a click to update the diagram and documentation. Documentation is a useful tool for discussion and the sharing of wisdom, particularly with a supervisor or through self-reflection. There is an additional benefit as much of the information can be copied to the final thesis or papers, increasing the information value and decreasing the work that takes place in the final stages when time is critical.

At the end of the day it's not so important our researchers get to their initial goal. Any novel contribution will do, as long as it is of interest and they know how they got there. Add a plan not only lets our researcher know how they got there, but also lets them know where they are going as the research progresses. These plans may be of use not only to them but to future researchers who continue the process. Better yet, it might allow us to share insight, increase wisdom and retain a little more of the knowledge our students generate.

VI. MAKING IT EASIER TO TACKLE HARD PROBLEMS

The problem of improving research is hard, largely because research problems themselves are by their nature difficult to solve. Brooking in his famous paper "No silver bullet - Essence and Accidents of Software Engineering" [18] note how the hard part of software development is the problem solving essence. He notes how breakthroughs to that time had tackled the accidental and not the essential difficulties. By dividing up our types of knowledge we separate the accident and essential factors and allow improvement to be focused on removing the remaining accidental factors. We also eliminate not the essential problem, but the side effects that come with complex problems – such as difficulties in communication, or size of the task of keeping documentation up to date in a rapidly changing research project. This won't solve hard problems, but it will make them easier to tackle.

In discussing the DIKW model we mentioned the additional final stage of enlightenment which some include. In our initial work which led to the RAISER/RESET SDLC, one of our case study subjects noted that initially class diagrams were used and found useful for learning about the existing code, yet slowly they became a burden as the information in them no longer needed to be looked up and was rapidly changing [19-21]. "We were so involved in it that everything was in our heads. We were sort of living and breathing it" [21] the time it was taking to update document "just got radiculous" [21]. The participants in the project were at this point at a level of enlightenment about their project. In our current research with MSc student projects we advise students to create high level architectural models of their projects, but to leave detailed class diagrams for dOxygen to generate from their source code. Not only does this save them time, it also ensures the accuracy of the class diagrams and better abstraction of the architecture. This reduced non essential effort.

Our recommendation of the process modeling tool allows researchers to be more systematic with their project planning while still being highly adaptable over time. We borrow the concept of a rapidly improving and changing information source from participants' knowledge of code, and have the information entered using their usual programming tools to reinforce this. The hard problem of ideas must still solved by the researcher, but by recording and presenting these ideas it becomes possible to facilitate more useful communication with their peers and supervisor. This allows additional wisdom and insight to be provided to them, in exchange for their own knowledge and perceived wisdom (their explicit rational for decisions made) being presented and explained. This bridges the knowledge gap, a non essential hurdle to making research progress, but one that allows far greater collaboration once subdued.

Gorman notes that as a consequence of the modern 'information explosion' "instead of storing information in our brains, we design our environments to make it easy to find the information we need." [22] Our approach to documentation moves students from storing information, to planning how it can be made available and used. This is based directly on past case study observations. Gorman also cites Wegner who said that "knowing where things are to be found can be a more important consequence of education than merely knowing things" [23]. Our approach provides a map, or at least the outlines of a map that students can fill in based on their understanding and then check over with their supervisors. For supervisors and later researchers it records essential information about rational, design or the lack thereof in certain aspects of a piece of work. We may not generate this essential knowledge, but with a new approach to research we enable knowledge that is currently lost to be saved. This in turn makes it easier to solve the harder problems.

Our discussion of knowledge and pipe model will we hope encourage others to find areas for improvement, and steer clear of approaches that disrupt the essential tasks. With our own work on the Software Engineering in the Research Environment project (SERE) we have created one approach, but this only a start. With a better understanding of the nature of research far more improvements can be made. Computer

science research is not just behind most established fields in eliminating accidental work, it is also behind the computing industry where appropriate tools and methodologies have been worked on since the advent of software engineering. We may have a beginning to knowledge, but there is much work still to be done.

VII. Conclusion

Knowledge comes in three forms. In the beginning the supervisor is king, bringing wisdom, knowledge and insight. In time as the student grows, the knowledge they generate on a particular subject will surpass that of the supervisor. They will absorb much of the relevant wisdom of their supervisor. When they meet, the insight of one will be shared with the other. For all of this to happen though, knowledge is required. Detailed knowledge, the sort generated with practical experience and experimentation. This is where the research students excels as they explore their chosen question and gain a unique insight into their work. To improve research, this time consuming step of knowledge generation must be made faster and more efficient, with the generated insight and knowledge being shared more easily. The research student, properly equipped and with a realist and adaptable plan (process model) can move quickly and efficiently through their work and better share their findings with others.

References

[1] HERO, "RAE Circular 2/00," vol. 2006: Higher Education Research Organisation, 2000.

[2] OECD, *The Measurement of Scientific and Technological Activities - Frascati Manual 2002 : Proposed Standard Practice for Surveys on Research and Experimental Development* Frascati, France: Organisation for Economic Co-operation and Development, 2002.

[3] HESA, "HESA Finance Record Coding Manual," 2004.

[4] W. T. Lord Kelvin, "Lecture to the Institution of Civil Engineers, May 3rd 1883," in *Popular Lectures and Addresses*, vol. 1. London & New York: Macmillan and Co., 1891, 1883, pp. 80.

[5] A. I. Kook, *Ein Eyah*, vol. II.

[6] J. Hey, "The Data, Information, Knowledge, Wisdom Chain: The Metaphorical link," Intergovernmental Oceanographic Commission 2004.

[7] R. L. Ackoff, "From Data to Wisdom," *Journal of Applied Systems Analysis*, vol. 16, 1989.

[8] M. Zeleny, "Management Support Systems: Towards Integrated Knowledge Management," *Human Systems Management*, vol. 7, 1987.

[9] D. Clark, "Understanding," in http://www.nwlink.com/~donclark/performance/understanding.html 2004.

[10] A. Oboler, "USE CSR," in *School of Computer Science and Software Engineering*, vol. B. Computer Science (Honours): Monash University, 2002.

[11] A. Oboler, "Examining the use of Software Engineering by Computer Science Researchers," presented at In Proceedings of Education Students' Third Regional Research Conference, Graduate School in Humanities University of Cape Town, Cape Town, South Africa, 2003.

[12] A. Oboler, D. M. Squire, and K. B. Korb, "Software Engineering for Computer Science Research - Facilitating Improved Research Outcomes," *International Journal of Computer and Information Science*, vol. 5, pp. 24-34, 2004.

[13] W. Waite, "Re: Use of software engineering in computer science research, personal communication," A. Oboler, Ed., 2002.

[14] B. Meyer, "Interview with Prof Bertrand Meyer, Interview conducted for the USE CSR project," 2002.

[15] G. Farr, "Interview with Prof Graham Farr," in *Interviews from the USE CSR Project*, A. Oboler, Ed. Melbourne: Monash University, 2002.

[16] R. Pressman, "Re: SE practise in Comp. Sci. Research, personal communication," A. Oboler, Ed., 2002.

[17] A. Oboler and I. Sommerville, "Research Documentation Guidelines: Capturing knowledge, improving research," in *Fourth International Conference on Information Technology : New Generations (ITNG 2007)*. Las Vegas, Nevada, USA: IEEE Computer Society, 2007.

[18] F. P. Brooks, "No silver bullet: essence and accidents of software engineering," presented at Information Processing 86, North Holland, 1986.

[19] L. Fitzgibbon, " Interview with Leigh Fitzgibbon," in *Interviews from the USE CSR project*, A. Oboler, Ed. Melbourne: Monash University, 2002.

[20] L. Alison, "Interview with Dr Lloyd Alison," in *Interviews from the USE CSR project*, A. Oboler, Ed. Melbourne: Monash University, 2002.

[21] J. Comley, "Interview with Josh Comley," in *Interview conducted for the USE CSR project.*, A. Oboler, Ed. Melbourne: Monash University, 2002.

[22] M. E. Gorman, "Types of Knowledge and Their Roles in Technology Transfer," *The Journal of Technology Transfer*, vol. 27, pp. 219-231, 2002.

[23] D. M. Wegner, "Transactive Memory: A Contemporary Analysis of the Group Mind," in *Theories of Group Behavior*, B. Mullen and G. R. Goethals, Eds. New York: Springer-Verlag, 1986.