

## CSci 780 Advanced Software Engineering

### Effective Technical Communication

“Writing Good Software Engineering Research Papers” – Mary Shaw

“An Evaluation of the Ninth SOSP Submissions”  
– Roy Levin and David D. Redell

9/1/2003

1

### “Writing Good Research Papers”, and “An Evaluation of the Ninth SOSP Submissions”

- Both of these papers address how to write good papers that have a higher chance of publication.
- Mary Shaw’s paper relates to software engineering, and the ICSE in particular
  - It:
    - Provides guidance on what a good paper should say
    - Provides statistics on papers submitted to ICSE 2002
    - Talks about the kinds of research that software engineers do
- Levin and Redell’s paper relates to operating systems research, and SOSP in particular
  - It:
    - Presents a set of questions that reviewers use to decide which papers get in
    - Is very oriented toward a systems audience
    - Is written in a humorous way, but has a serious point. (The subtitle is “How (and How Not) to Write a Good Systems Paper”)
- Today we’ll:
  - Discuss papers
  - Look at examples

9/1/2003

### Outline

- Papers
  - Types
  - Structure
  - Hints
- Presentations
  - Audience
  - Structure
  - Presentation

9/1/2003

3

### Types of Papers—Forums (1/4)

- Position papers (2-3 pages)
  - State a point of view: challenge the readers
  - Often solicited from experienced researchers
  - Research results not necessary
- Workshop papers (3-6 pages)
  - Position papers, or unevaluated ideas
  - Used to spur discussion, or decide who can attend

*Neither of these gets you "credit"*

9/1/2003

4

### Types of Papers—Forums (2/4)

- Technical papers at top conferences (10-12 pages)
  - Present one good idea with recent supporting results
  - Reviewed by 3 people from the program committee
  - One chance to get in, one chance to revise
  - Publication time is usually about a year
  - Best paper awards

*Top conferences give you "credit"  
( <20% acceptance rate)*

9/1/2003

5

### Types of Papers—Forums (3/4)

- Journal papers (10-50 pages)
  - Present multiple related ideas with well-developed supporting results
  - Carefully reviewed by editorial board
  - Author usually has multiple chances to revise (if you get “accept with revisions”)
  - Publication time is usually about 2-5 years

*Usually full "credit"*

9/1/2003

6

## Types of Papers—Forums (4/4)

---

- Technical Reports (10-50 pages)
  - No "credit"
  - Useful for user's manuals, long appendix-like material
  - No review
  - Publication time is nil
- PhD Theses (200-500 pages)
  - 4-8 papers work of ideas
  - Brings together a body of related work
  - Reviewed by PhD committee
  - Published like a technical report

9/1/2003

7

## Types of Papers—Content [Shaw] (1/2)

---

- Method or means of development
  - "Automated Support for Classifying Software Failure Reports"
    - 13% accepted, 42% of accepted papers
- Method for analysis or evaluation
  - "Cost Estimation for Web Applications"
    - 20% accepted, 44% of accepted papers
- Design, evaluation, or analysis of instance
  - "Software Technology in an Automotive Company—Major Challenges"
    - 12% accepted, 12% of accepted papers

9/1/2003

8

## Types of Papers—Content [Shaw] (2/2)

---

- Generalization or characterization
  - "Toward an Understanding of the Motivation of Open Source Developers"
    - 6% accepted, 2% of accepted papers
- Feasibility study or exploration \*
  - "Multiple Mass Market Applications As Components"
    - 0-100% accepted, 0% of accepted papers

9/1/2003

9

## Structure

9/1/2003

10

## "The Scientific Method"

---

- Problem
- Background
- Hypothesis
- Materials
- Procedure
- Data
- Data analysis
- Conclusion

9/1/2003

11

## Conceptual Paper Structure

---

- Problem Statement
- Proposed Solution
- Evaluation Methodology
- Data
- Evaluation
- Related Work
- Conclusion

9/1/2003

12

## Physical Paper Structure

- Abstract 150-200 words
- 1: Introduction 1-1½ pages
- 2: Background \* 1-2 pages
- 3: Description 4-6 pages
- 4: Experiment & Data 2-3 pages
- 5: Evaluation/Discussion ½-2 pages
- 6: Related Work \* ½-1 page
- 7: Conclusion 2 paragraphs
- Acknowledgements 2 sentences
- References ½-1 page

9/1/2003

13

## The Abstract [Shaw]

- 2-3 sentences about current state of the art
  - The "problem statement"
- 1-2 sentences about the how you plan to fix it
  - The "approach" or "contribution"
- 1-2 sentences about the specific details
  - The "experiment"
- 1 sentence about how the result is defended
  - The "results"
  - Sometimes omitted for space

9/1/2003

14

## Example #1

Software maintenance and evolution are the dominant activities in the software lifecycle. Modularization can separate design decisions and allow them to be independently evolved, but modularization often breaks down and complicated global changes are required. Tool support can reduce the costs of these unfortunate changes, but current tools are limited in their ability to manage information for large-scale software evolution. In this paper we argue that the map metaphor can serve as an organizing principle for the design of effective tools for performing global software changes. We describe the design of Aspect Browser, developed around the map metaphor, and discuss a case study of removing a feature from a 500,000 line program written in Fortran and C.

9/1/2003

15

## Example #2

Modeling languages and the software tools which support them are essential to engineering. However, as these languages become more sophisticated, it becomes difficult to assure both the validity of their semantic specifications and the dependability of their program implementations. To ameliorate this problem we propose to develop shared semantic domains and corresponding implementations for otherwise unrelated modeling languages. The idea is to amortize investments at the intermediate level across multiple language definitions and implementations. To assess the practicality of this approach for modeling languages, we applied it to two languages for reliability modeling and analysis. In earlier work, we developed the intermediate semantic domain of *failure automata* (FA), which we used to formalize the semantics of *dynamic fault trees* (DFTs). In this paper, we show that a variant of the original FA can serve as a common semantic domain for both DFTs and reliability block diagrams (RBDs). Our experiences suggest that the use of a common semantic domain and a shared analyzer for expressions at this level can ease the task of formalizing and implementing modeling languages, reducing development costs and improving their dependability.

9/1/2003

16

## Example #3

This paper describes a methodology for designing Open Implementations—software modules that can adapt or change their internals to accommodate the needs of different clients. Analysis techniques are used for capturing domain knowledge, user requirements, and domain properties that influence the module's eventual implementation. Design techniques are used for determining and refining the interfaces by which clients control the modules implementation strategies. The methodology has evolved over the past two years in several pilot projects.

9/1/2003

17

## Introduction

- A long form of the abstract
- Discuss background if necessary
- Convince the reader that your problem is important, and your solution worth reading about
- Make your contributions and claims clear
- End with a "roadmap" for the rest of the paper

9/1/2003

18

## Background/Related Work

---

- Background:
  - If the reader needs a lot to understand the paper
- Related work:
  - Discuss (carefully!) the point of the work
  - Relate it to what you are doing—How is your work better or different?
  - Gives you credibility

9/1/2003

19

## Example #1

---

Our experience using Z/Eves is similar to that of Knight et al. [20], who used the PVS theorem prover on a modest-sized nuclear power plant specification. They too found it hard to formulate theorems and proof strategies, and were hindered by poor tool usability.

9/1/2003

20

## Example #2

---

Another prototyping environment has been developed for the formal specification language TROLL [Jungclaus91]. It is called Tbench [Kusch95] and uses an editing and execution system for the specifications. In contrast to our environment, Tbench does not automatically generate a first software model, as a first step in the development process of the final system. There is another CASE tool for LCM [Wieringa94] called TCM (Toolkit for Conceptual Modeling), but in this case, the generation of prototypes is not allowed, to the best of our knowledge.

9/1/2003

21

## Description

---

- Usually multiple sections
- Usually the easiest part to write
- Usually the longest part of the paper
  - Unless the whole paper is an empirical study
- A good example goes a long way
- Beware of strong statements: “never”, “always”
- Support every claim

9/1/2003

22

## Experiment & Data

---

- Convince the reader the experiment is sound
- Use good statistics
  - Error bars/confidence intervals
  - Beware scalar values
    - \$16 billion in foreign aid = 0.135% of US GDP
  - Beware percent values
    - 1 penny + 1 penny = 100% increase

*Unfortunately, collecting hard data in SE is hard*

9/1/2003

23

## Evaluation/Discussion

---

- Discuss the weaknesses of your approach
- Discuss the weaknesses of your experiment
- Explain what you learned
- Explain how your hypothesis is supported

*A negative result is still a result*

9/1/2003

24

## Conclusion

---

- Summarize in 1 paragraph
- Wax philosophical:
  - Broader research/impact
  - Future work
- Some people merge discussion and conclusion

9/1/2003

25

## Acknowledgements and References

---

- Acknowledge
  - Grants: So you can show funding agencies what they're paying you for
  - Helpful people: implementors, industrial collaborators, discussion people
  - Anonymous reviewers
- References
  - Simple: use BibTeX

9/1/2003

26

## Hints

9/1/2003

27

## Hints: Use Keywords (1/6)

---

- Problem: "unfortunate", "difficult", "tedious", "poor"
- Approach: "In this paper we present", "We address this problem by"
- Experiment: "To evaluate", "To assess", "case study", "implemented a system"
- Results: "Our experiences suggest", "our data supports"
- Protection: "To the best of our knowledge", "As far as we are aware"

9/1/2003

28

## Hints: Remember That It's *Science* (2/6)

---

- The cool system is not the point—it's the idea that the system supports
- Don't say "prove" unless you mean it
- Describe your experiment as such
- What is your null hypothesis?
- Your experiment is one data point
  - "Our results demonstrate that the approach works"
  - "Our results indicate that the approach can work"

9/1/2003

29

## Hints: Your Own Work (3/6)

---

- Cite your own previous work
- Relate the paper to your previous work
- You can plagiarize yourself!
- Minimally publishable unit? (MPU)

9/1/2003

30

## Hints: Spin (4/6)

---

- Write the abstract first to clarify the paper
- “Spin” can make a mediocre paper a great one
  - More than fluff—putting the research in context
- Maybe write half the paper before starting work!
- SOSP theme: 5 person-years of effort, no research contribution!
  - Do a little revisionist history: find out what the right problem/context would have been
  - Dangerous: need to work in “fruitful” areas

9/1/2003

31

## Hints: Writing (5/6)

---

- Use simple writing—the ideas are complicated enough
- Make sure the paper tells a “research story”
- Avoid third-level subsections
- Make sure figure text is readable
- Be honest and up-front, but not meek or defensive
- Learn the art of compression
- Always spell check and grammar check

9/1/2003

32

## Hints: Miscellaneous (6/6)

---

- Get as many people to read it as possible
  - Finish it 1 month before the deadline
  - Reread it yourself after a couple weeks
- Learn your document system
  - Figure positioning, captions, cross-references
  - BibTeX in LaTeX
  - Styles in Word
- Use the conference or journal template
- *Always* get co-author sign-off before submission

9/1/2003

33

# Presentations

9/1/2003

34

## Audience: Conference

---

- Goal: Convince listeners to read the paper
- 25 minutes with 5 minutes of questions
- No interruptions (except by session chair)
- Small room with 20 people, or large hall with 100-200 people
- Heterogeneous crowd

9/1/2003

35

## Audience: Class

---

- 50-75 minutes
- Need more examples, interaction via questions
- PowerPoint can be a detriment at times
- Lots of interruptions
- *Paper summaries*
  - Present the paper’s problem, contribution, etc.
  - Gain perspective by reading other’s thoughts
  - Pull in additional relevant material
  - Interrupt for clarification, but discuss after

9/1/2003

36

# Structure

9/1/2003

37

## Title Slide

---

- Don't say the title and your name—the session chair will do it for you
- *Do* acknowledge co-authors and indicate if they are in the room

9/1/2003

38

## Outline

---

- Key bullets of talk
- Consider using transition slides
- Mini-outline in the corner?

9/1/2003

39

## Main Slides

---

- First make the problem and contribution clear
- Follows general paper structure, except:
  - You may need to explain more background
  - You will definitely need to summarize content
  - Use an example up front to make your work concrete
  - Diagrams can help a lot
  - Sometimes no time for related work
- Enough detail to convince them there's "meat"

9/1/2003

40

## Questions

---

- *Always* repeat/rephrase the question
  - Lets others hear it
  - Allows time to think, or to "correct" the question
  - Ensures that you didn't misunderstand
- Don't be defensive
  - Turn a criticism into a positive statement
- Defer lengthy answers to a private meeting

*You will be judged on your presentation,  
but also how well you handle questions*

9/1/2003

41

# Presentation

9/1/2003

42

## Length and Content

---

- Length
  - 1 minute/slide is *fast*
  - HASE: 20, ICSE 2000: 21, ICSE 2003: 24, ISSRE 2000: 27, PhD Defense: 30, Job talk: 38
- Verbosity
  - Shoot for 1 line bullets, with suitable whitespace
  - Avoid many levels, lots of noise, full sentences
  - These are your notes—You don't want people reading

9/1/2003

43

## Practice, Practice, Practice

---

- Practice by yourself
  - Learn the right words, transitions
  - Learn timing
  - Do it until you don't need to look
- Practice in front of friends who are critical
  - Find unclear spots
  - Get suggestions
  - Learn common questions

9/1/2003

44

## Speaking

---

- Be excited about what your work
  - Especially after practicing the talk 10 times
- Don't put your hands in your pockets
- Face the audience, not the screen
  - You've memorized it right? (Use the laptop display)
  - Don't block the screen
- Speak with confidence—no "um"s
- Avoid telling them too much detail
- Avoid reading the slides

9/1/2003

45

## At the Conference

---

- Large Room Considerations
  - Be aware of your microphone
  - Make sure you have a pointer
  - Do you need a driver?
  - Repeat the question
- Use your own laptop whenever possible
- Use power supply; disable screensaver
- Otherwise check fonts, figures, etc.
- Always bring backup disk & plastic just in case

9/1/2003

46

## Miscellaneous

---

- Number slides so people can refer to them
- Put a URL or the title slide at the end
- Use animations wisely
- Put extra slides at the end for common questions
- Bring something to drink

9/1/2003

47

**Demo?**

9/1/2003

48