SE Software Engineering
The University of Auckland

Software Development Methodologies

Lecture 7 - Academic Writing

SOFTENG 750 2013-04-15

Final Report Grading Schedule

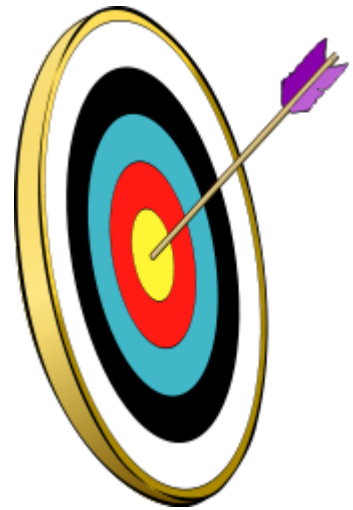
Software Engineering
The University of Auckland

Approx. **5 pages** (including figures) IEEE style

1. **Introduction**: Introduce and motivate the project.
2. **Requirements:** What is the project trying to achieve?
3. **Related Work:** What have others done? Compare it with your project.
4. **Design:** Software architecture (e.g. class diagram)? User interface (e.g. screen diagram)?
5. **Implementation:** What have *you* implemented?
6. **Evaluation:** How have you evaluated your app? What are the results?
7. **Methodology/Management** of your team in the project.
8. **Conclusion**: Conclusions? Lessons? Future work?

Requirements

What is the problem / need you are addressing?



- **Where from?**
  - From related work (what do others think/do?)
  - From real users (ask/survey/read about them)
  - From real products (what do other systems do?)
  - Through analysis (what is logically required?)
- **How important is it? Why?**
- **Organize** in categories (possibly subsections)
  - Functional requirements (what does it do?)
  - Non-functional requirements (how does it do it?)
    E.g. usability, performance, safety, security, …
  - Developer requirements:
    code understandability, maintainability, ...

Finding Related Work

1. **Gather** phase
   - Keyword search
     (e.g. Google Scholar, ACM, IEEE)
   - Follow up the references
     (cited and citing papers)
2. **Filter** phase:
   read only abstract and throw blanks out
3. **Reading** phase

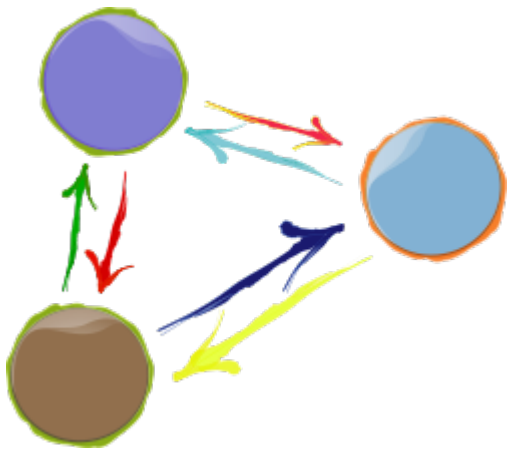
The “someone else has already done it” problem

- Look again, is it really the same?
- Related work is good!

Writing about Related Work

Software Engineering
The University of Auckland

1. **Summarize** in a few bullet points what each related paper is about
   - What did they try to do? What was novel about it?
   - Did they achieve it? Did they evaluate it?
2. **Organize** the related works by grouping them
   - Define categories, write one section per category
   - Possibly subcategories, subsections
3. **Analyze & Compare**
   - What are the difference between the works?
   - Strengths? Weaknesses?
   - Efficiency, Effectiveness, Usability
   - Constraints, Assumptions



Peer-reviewed Publications

- Scientists **submit papers** to journals or conferences that are peer-reviewed (short paper ~5 pages, long paper ~10 pages, journal article ~20 pages)
- Other scientists, established experts in the field, **review the papers**, free of charge
  - Individual reviewers remain **anonymous**, but the board of possible reviewers is known
  - They check for: obvious inconsistencies, dubious statements, good standards of presentation, sufficient degree of completeness, i.e. disclosure of details, and novelty of the results
  - They usually will **request changes**
- Many papers get **rejected** (usually 50% or more)

Isn't Wikipedia Peer-reviewed?

Software Engineering
The University of Auckland

- In some sense, but does not reach a final state
- Wikipedia is good at
  1. Giving a start at a topic (you don't want to know *less* than Wikipedia on your topics)
  2. Providing **links** to more reliable information sources
  3. Naming **alternative views**
  4. Possibilities: tell you what *could* be true
  5. Myth busting (e.g Ford invented the production line, yeah right!)
- Editing policies improve Wikipedia's quality, but are not dependable enough
- Wikipedia should be treated as a **mixed bag**, with lots of pearls - but you have to pick them

Citing Publications

- Citing many sources is good to back up your claims
- But only **quality assured publications**
  1. Peer-reviewed (not just pro forma, e.g. no fake conferences)
  2. Cited by others in the field (means you should know about it if you are working seriously)
  3. Reputation of conferences & journals (e.g. see rankings)

- Other sources?
  - **Industry whitepapers** can be cited, but they are not quality assured, so no strong backup for you
  - **Textbooks** only for very specific things (refer to them with page or chapter number)
  - **Not Wikipedia** (see previous slide)
- Use proper **citation style**

Design

How do you achieve your requirements?
User interface design, software architecture, algorithms & data structures, ...

- **Explore the design space** of your project analytically.
  What are the possibilities?
  What are the limits? Sweet spots?
- **Top-down**: start with an overview of your design and then go down into the details
- **Design alternatives**: What are they? Advantages/disadvantages?
- Always argue with your **requirements** (they are your aim)
  - What is the best way to meet your requirements?
  - Your design does not need to do more than meet them
  - If the requirements are not met, be honest about it and explain why (you may have good reasons, e.g. time)

Implementation

Software Engineering
The University of Auckland

How did you build your system?

- What tools/technologies were used?
- Implementation challenges and how you solved them
  - What was hard? Why? Solutions?
  - Small code snippets for illustration
- Use screenshots to illustrate your user interface
  (if you haven't done so already in the Design)
- Limitations: often your implementation is just a prototype
- What information would have been useful for you
  when you started coding?

Evaluation

What evidence do you have that your work meets the requirements (that it is useful)?

- What **methods** are you using? Why?
  (e.g. think-aloud, questionnaire, ...)
- What are your **results**?
- What do the results mean
  (for the success of your work)?
- What limitations did you find in your work (be honest)?
  What are the limitations of the way you evaluated it?

Introduction

What are you doing? Why are you doing it?

1. **Introduce** the topic and the context
2. **Motivate** the research
   - Gap in the literature (unexplored territory)?
   - Interesting applications?
   - Significant consequences (e.g. cheaper, faster)?
3. **Research question**(s):
   What are you trying to find out or trying to show?
   What are your contributions (briefly)?
4. **Outline** of the paper
   ("Section 2 gives an overview of related work…")

Conclusion and Abstract

Conclusion

1. Sum up what you have investigated
2. Sum up your conclusions / contributions
3. Point out some future directions (e.g. new research questions)

Abstract (typically ~200 words)

4. What is your project about? -> Problem, Motivation
5. How did you do it? -> Methodology
6. What are your results and why are they significant? -> Solution/Contribution

Writing Style

1. **Organize your paper clearly:**
   Every part of your paper should have a clear function,
   i.e. typically make one point clearly
   - **Sections:** make sure you understand what goes where
   - **Paragraphs:**
     - Each expresses one idea clearly
     - Typical structure: make main argument, elaborate/ explain argument, conclude and transition to next argument
     - Split larger ones, join smaller ones (< 3 sentences)
2. **Be concise:** say things once and say them clearly
   - Redundancy is a sign of poor organization
   - You can always refer to the place where something was explained
3. **Write for your audience**: Make sure your explanations can be understood by others (not just yourself);
   also make sure your arguments convince them

Beating Writer's Block

- Vary the **structure**:
  - Just write section/subsection headers
  - Just write bullet points, flesh out later
- Vary the **topic**:
  - Write about anything that comes to your mind (e.g. some related work, design, introduction, …)
  - Organize/ reshuffle the parts later on
- Vary the **modality**:
  - **Visual**: create figures first, the simply describe what you see
  - **Auditive**: talk to others about it; write exactly as you would explain it verbally
  - **Kinesthetic**: Do some development or some experiments, then describe what you have done

Polishing your Paper

- Same as with software development: iterative and incremental **refinement**
- Get (early) **feedback** from others:
  - Is it easy to understand? Concise?
  - Spelling/grammar
  - Obvious omissions? Undiscussed limitations?
  - Could there be more/less figures?
  - Other relevant references?
- Emphasize your **contribution** (in abstract, intro, conclusion)
  - How is your work different? Better?
  - How have you evaluated your work?

Conclusion

- Overall strategy for writing a paper
  1. Start with the structure
  2. Collect information as you go & write it down
  3. Get feedback
  4. Revise
- As beginner, stick to the structure proposed here
- Revise, revise, revise...

References:

- Tips for Academic Writing (on my homepage): http://www.cs.auckland.ac.nz/~lutteroth/other.html
- Gopen & Swan (1990). The Science of Scientific Writing. http://www.americanscientist.org/issues/pub/the-science-of-scientific-writing
- IEEE style template for Word: http://www.cs.auckland.ac.nz/~lutteroth/students/IEEETemplate.doc

Quiz

1. What are important sections of a typical SE research paper?
2. What goes into each of the sections?
3. What are good references to cite? How do you find them?

International Obfuscated C Code Contest (ioccc.org) - Nick Johnson 2004

A maze game.

```
#include <ncurses.h>/*******************************************************/
            int             m[256                   ] [         256   ],a
 ,b   ;;;   ;;;   WINDOW*w;   char*l=""    "\176qxl"    "q"    "q"    "k"    "w\
xm"    "x"    "t"          "j"          "v"          "u"          "n"          ,Q[
 ]=    "Z"    "pt!ftd`"    "qdc!`eu"    "dq!$c!nnwf"/**    ***    */"t\040\t";c(
int    u ,          int          v){                        v?m    [u]          [v-
 1]    |=2,m[u][v-1] &    48?W][v-1    ] &    15]]):0:0;u?m[u    -1][v]|=1    ,m[
 u-             1][    v]&          48?                  W-1    ][v          ]&
15]    ]):0:0;v<    255    ?m[    u][v+1]|=8,m[u][v+1]&    48?    W][    v+1]&15]]
):0          :0;          u <                 255    ?m[    u+1          ][v    ]|=
4,m[u+1][    v]&48?W+1][v]&15]]):0:0;W][    v]&    15]    ]);}cu(char*q){    return
 *q                ?cu    (q+          1)&          1?q    [0]                ++:
q[0    ]--    :1;    }d(    int    u ,    int/**/v,    int/**/x,    int    y){    int
Y=y    -v,    X=x          -u;    int          S,s    ;Y<          0?Y    =-Y    ,s,
s=-    1:(    s=1);X<0?X=-X,S    =-1    :(S=    1);    Y<<=    1;X<<=1;    if(X>Y){
int    f=Y                -(X    >>1    );;                while(u!=          x){
f>=    0?v+=s,f-=X:0;u    +=S    ;f+=    Y;m[u][v]|=32;mvwaddch(w,v    ,u,    m[u
 ][           v]&    64?    60:          46)          ;if          (m[    u][
v]&16){c(u,v);;    ;;;    ;;;    return;}}    }else{int    f=X    -(Y>>1);;    while
 (v    !=y          ){f    >=0          ?u    +=S,                f-=          Y:0
 ;v    +=s    ;f+=X;m[u][v]|=    32;mvwaddch(w,v    ,u,m[u][v]&64?60:46);if(m[u
 ][                v]&          16)    {c(    u,v                          );
  ;    return;;;}}}}Z(    int/**/a,    int    b){    }e(    int/**/y,int/**/    x){
int                i ;          for          (i=          a;i                <=a
+S;i++)d(y,x,i,b),d(y,x,i,b+L);for(i=b;i<=b+L;i++)d(y,x,a,i),d(y,x,a+    S,i
 );                ;;;          ;;;          ;;;                ;;;    ;
  mvwaddch(w,x,y,64);    ;;;    ;;;    ;;;    prefresh(    w,b,a,0,0    ,L-    1,S-1
);}              main(          int                V ,    char                *C[
  ]    ){FILE*f=    fopen(V==1?"arachnid.c"/**/    :C[    1],"r");int/**/x,y,c,
v=0          ;;;       initscr              ();                Z(Z          (raw
 ()    ,Z(    curs_set(0),Z(1    ,noecho())))),keypad(    stdscr,TRUE));w    =newpad
  (    300,    300                ) ;    for          (x=    255    ; x    >=0    ;x--
  )    for    (y=    255    ;y>=0;y--    )m[    x][    y]=    0;x=y=0;refresh( );while
  (    (c=                          fgetc (f)    )+1)                          {if(
0||c==10||    x==    256){x=0;y++;if(y==256    )break;;}    else{m[x][y]=(c    ==
'~'    ?64    : c    ==32                ?0:          16)    ;;x                ++;
      }}for(x=0    ;x<    256;x++)m    [x][0]=16    ,m[    x][    255]=16;for(y=0
;y<          256    ; y          ++)    m[0          ][y    ] =                16,
m[255][y]    =16    ;a=b=c=0;    x=y    =1;    do{v++;mvwaddch    (w,    y,x    ,m[
x][          y]&                32?    m[x                          ][y    ] &    16?
 0|    acs_map[l[m[x][y]&15]]:46 :    32);c==0163&&!(m[x][y+1]&16)?y++:    0;c
 ==    119          &&!          (m[                                          x][
 y-    1]&    16)    ?y--:0;;c    ==97    &&!(m[x-1][y]&16)?x--:0;c==100&&!(m[x+1
 ][    y]&    16)          ? x    ++:0                ;if(                c==
 3-    1+1    ){endwin(    );;    return(0)    ;}x    -a<5?a>S-    5?a-=S-5:(a=0):
0;x                -a>          S-5?a<255    -S*          2?a                +=S
-5:(a=256-S):0;    y-b<5?b>L-5?b-=L-5:(b    =0)    :0;    y-b>L-5?b<255-L    *2?
b+=                             L-5    :(b                          =256
-L)    :0;e(x,y);if(m[x][y]&64)break;}while((c=getch())!=-1);endwin();cu(Q);
printf(Q,v);}
```