Technical Writing

Lecture 24
What Is “Technical Writing”? 

- Writing we do as part of our jobs
- Possible purposes:
  - Inform
  - Instruct
  - Persuade
  - Call to action
- Missing from this list:
  - Entertain
Effective Technical Writing

1. Engages a specific audience
2. Uses plain and objective language
3. Stresses presentation (obvious structure, understandable at a glance)
4. Employs visual aids
Why Bother?

- Communication is fundamental in society
  - Politics, law, science, personal lives, health, ...

- Fundamental in personal success:
  - Good idea
  - **Ability to communicate that idea**

- Highly valued by employers
Writing in Computer Science

- Taking exams
- Documentation (for users and developers)
- Reports and memos
- Papers (journals, conferences, magazines...)
- Proposals
- Reviews of others’ work
- Books
Good and Bad News

- The bad news
  - Most of us are not very good at it
  - We enjoy “technical” challenges much more

- The good news
  - Writing is actually not too different from computer science!
Writing Code vs Writing Prose

Programming

Need to identify:
- User
- Problem/Need
- Features
- User interface

"preprogramming"

Technical Writing

Need to identify:
- Audience
- Purpose
- Cognitive depth
- Style and tone

"prewriting"
# Software/Document Engineering

<table>
<thead>
<tr>
<th>Soft Development</th>
<th>Doc Development</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements and design</td>
<td>Prewriting</td>
</tr>
<tr>
<td>Implementation</td>
<td>Composition</td>
</tr>
<tr>
<td>Testing</td>
<td>Reviewing</td>
</tr>
<tr>
<td>Debugging</td>
<td>Revising</td>
</tr>
</tbody>
</table>
Analyzing the Audience

- Few people read this stuff for fun
- Must correctly identify a “customer” and what that customer’s needs are
- Consider writing an audience profile
  - Novice, technician, expert, manager, VC, ...
  - Reading level
  - Motivations, biases, expectations, ...
- Explicitly state assumptions made on background, motivation, needs, etc.
  - “The reader is expected to be familiar with the predicate calculus.”
  - “This manual is designed for application programmers who write S47G applications for the insurance industry.”
Technical Audience

- Function-oriented organization
  - E.g., alphabetical listing of all functions

- Want a complete (exhaustive) resource
  - All information they might want is there somewhere

- Willing to spend a great deal of time
  - Will read the document carefully
Customer Audience

- Prefer task-oriented organization
  - E.g., enumerated steps for each possible task

- Want just the necessary information
  - Only the information critical to their jobs is there

- Will spend as little time as possible
  - Must be concise and easy to read
Identify the Purpose

- Golden rule of business communication
  - Begin with clear statement of what you want!
  - "Start with the ask"

- Larger documents are no different
  - What information are you trying to impart?
  - What are you trying to teach?
  - What view do you want the reader to adopt?
  - What action do you want done?
The Depth of Writing

- Bloom’s taxonomy of cognition (1957):
  1. Knowledge
  2. Comprehension
  3. Application
  4. Analysis
  5. Synthesis
  6. Evaluation

- Revised by Anderson & Krathwohl (2001):
  1. Remember
  2. Understand
  3. Apply
  4. Analyze
  5. Evaluate
  6. Create
Verbs in Statement of Purpose

- **Knowledge (Remember)**
  count, define, draw, identify, indicate, list, name, quote, recall, recite, recognize, record, state, tabulate, trace, write

- **Comprehension (Understand)**
  compare, compute, contrast, describe, differentiate, discuss, distinguish, estimate, extrapolate, interpolate, predict, translate

- **Application (Apply)**
  apply, calculate, classify, complete, construct, demonstrate, employ, examine, illustrate, practice, relate, solve, use
Verbs in Statement of Purpose II

- **Analysis (Analyze)**
  - analyze, detect, explain, group, infer, order, relate, separate, summarize, transform

- **Application (Apply)**
  - apply, calculate, classify, complete, construct, demonstrate, employ, examine, illustrate, practice, relate, solve, use

- **Synthesis (Create)**
  - arrange, combine, construct, create, design, develop, formulate, generalize, integrate, organize, plan, prepare, produce, specify

- **Evaluation (Evaluate)**
  - appraise, assess, critique, determine, evaluate, grade, judge, measure, rank, select, test
Prewriting: Getting Started

- Read the question / problem statement carefully
- Make a list of the required cognitive tasks
- Assess what you know
- Compare this knowledge with the level of the required cognitive task
Example

“Compare the performance of two cache replacement algorithms”

- Cognitive tasks
  - Compare
  - Contrast
  - Maybe analyze and recommend?

- Recall various issues in cache algorithms
Prewriting Tasks

- **Quick list**
  - Specific points for each cognitive task

- **Brainstorm**
  - List everything you know about the topic
  - Do not judge or weed anything out
  - Objective: quantity

- **Review list**
  - Assess where research is needed
Prewriting Tasks II

- Choose a single point that will be in the final product and outline a section to develop that point.
- Involve others as appropriate.
- Do the research.
- Plan the format.

This is not a linear process!
Prewriting Tasks III

- Outlining
  - Get the planned structure down
  - Avoid forgetting a key point
  - Check for the logical flow of arguments and information

This is an easy step to skip, but good work here will pay dividends in the future!
Writing the Document

- The better the preparation in the prewriting phase, the smoother this goes.
- Regardless, it’s still work!
- Requires tools, skills, practice, experience, and motivation.

- Component = section of the document.
- Often has its own heading.
- Large components consist of smaller ones.
- Each (large enough) element from the outline becomes a component
Advantages of Components

- The whole document is too intimidating
- Obvious milestones
  - Reduces panic (you know where you stand)
  - Permits time budgeting
- Reduces writing to a step-by-step process
- Instant gratification
- Easy cure for writer’s block: work on a different section
Writing a Component

- Know the purpose
- Have all the information
- Different strategies:
  - Write a draft using sentences
  - Jot down points in any form, then flesh out into sentences
  - Combination (sentences, phrases, points)
Overcoming Writer’s Block

- Start *anywhere*
- If the problem is lack of information, go back and do more research
- Explain it to someone else (verbally)
- Work on a different section
- Take a walk
- Imagine life when you are done
Overcoming Writer’s Block II

- Force yourself to sit at your desk until done
- Revise your outline or organization
- Revise some section you’ve already written
- Change your environment
- Diagram the structure of the component
- Set an impossible schedule... and then panic!
- Take a break
Rhetorical Patterns

- Every culture has well-established patterns of exposition
  - Ready-made structures into which specific information may be dropped
- The reader is already familiar with these patterns
- The technical writer does not have the time (or skill) to invent new ones
General-to-specific Pattern

- Often used for introductory section
- Start with the most general statement
  “More computing resources are devoted to the management of data than to any other task”
- Gradually get more specific
  “This program simplifies the manipulation of numeric data on a personal computer”
- Finally specific statements
Classification Pattern

- Organize information by dividing it into categories
  - E.g., a section on each of initialization, data entry, selection, access control, etc...

- Within each category, present parallel information
  - E.g., purpose, prerequisites, results, error messages, alternatives, references, etc...
Comparison-contrast Pattern (Point-by-point)

- Consider one aspect at a time
  - In football, the ball may be thrown forward from behind the line of scrimmage
  - In rugby, only lateral passes are allowed
  - In football, play ends when the ball-carrier is tackled
  - In rugby, play continues after a tackle, but the tackled player must release the ball
Comparison-contrast Pattern (Whole-to-whole)

- Two ways to create a new object (zack)
  1. From the command line
     - On the command line, type “edit zack”
     - Fill in attributes of the presented template
     - Select Save from the File menu
  2. From within the application
     - Select New from the File menu
     - Fill in attributes of the presented template
     - Select Save As from File menu, and type “zack”
Definition Pattern

- Typically short and simple
- Example
  - "Undo is a function that restores an object to its state immediately prior to that last operation"
  - Places “Undo” in the class of functions, then distinguishes it from other functions
Chronological Pattern

- Typical for task-oriented instructions
- Given in the order in which they must be performed
Cause-and-effect Pattern

- Often used for error messages
- Give a list of error messages
  - ordered alphabetically, by error number, ...
- For each message, list the possible causes
  - ordered most to least likely
- After each cause, give the action(s) the user should take to recover
Putting Components Together

- Add headings (part, chapter, section, ...)
- Add transitions where needed
  - Important to prompt reader for what to expect, or to reinforce that some change is coming
Possible Transitions

- Moving to the next point in a sequence
  - “Firstly, secondly, ...”

- Contrasting item or viewpoint
  - “However, on the other hand, otherwise,...”

- A result or conclusion
  - “Therefore, in consequence, ...”

- Relating things in time
  - “Now, then, soon, immediately,...”
Possible Transitions

- Introducing an example
  - “For example, that is, ...”

- Further strengthening a point
  - “Moreover, similarly, further, ...”

- Concluding
  - “In conclusion, in summary, finally,...”
Preliminary Draft

- Starting is always difficult
  - Helps to remember it’s just a draft!
- Don’t worry about spelling, grammar, form
- Spend effort on sound communication of major points
- Fill in your outline
Middle Draft

- Build on the base of the preliminary draft
- Refine the organization and fill in points
- Ensure each point belongs in that paragraph
- Cut and paste
- Play with the text
  - Font, layout, spacing, page count
Final Draft

- Spelling and grammar
  - Run the spell checker, but that’s not enough!
- A “which” hunt
- Word choice
- Transitions
- Typos
- Pagination
Revising

- Where bad writing becomes good writing
- First draft is always bad
  - Tempting to become attached to text we’ve written
  - Write the first draft anticipating that it will change in the future
Revising Tasks

- Add flow and smooth transitions
- Careful, accurate movement from one point to the next, one section to the next
- Make decisions that have been put off
- Reduce wordiness
- Clarify subordination relationships between points
Red Flag – Inconsistent View

- Changing from 2\textsuperscript{nd} to 3\textsuperscript{rd} person
  - “Limit your disk storage to 100 Mb. The user can submit a request for more storage space to the system administrator.”
  - “Limit your disk storage to 100 Mb. For more storage space, you can submit a request to the system administrator.”
Red Flag – Passive Voice

- The verb expresses what is done to the subject (by someone or something)
- Occasional use is OK (and even unavoidable in many technical documents)
- But excessive use weakens your writing
  - “This error is used by the parser to indicate…”
  - “The parser issues this error to indicate…”
In the final analysis, the end result of a wordy document is increased cost in terms of pages of paper, bytes on a disk, and inefficient use of the reader’s time and the writer’s effort.

Words cost money. It is cheaper to print a short book than a long one.
Red Flag – Faulty Parallelism

- The use of different grammatical constructs in a parallel structure
- Consider the list:
  - Preparing for installation
  - How to configure
  - Do you want the advanced options?
Red Flag – Dangling Modifier

- The use of a verbal phrase that does not connect (or modify) anything else in the sentence
  - “After typing enter, the system will continue with the second pass over the program”
  - “After you type enter, the system will continue with the second pass over the program”
Red Flag – Ending a Sentence With a Preposition

Example:
- “Before using the software, you must set it up”
- “Before you can use it, you must set up the software”
- “You must set up the software before using it.”

Winston Churchill:
- “That is criticism up with which I will not put.”
Red Flag – Splitting a Verb

- Dogma: Avoid splitting an infinitive
  - Infinitive is “to” + verb, or “will” + verb
    - eg “to think”, “to breathe”, “will dance”

- Avoid putting an adverb inside
  - Patients should try to if possible avoid going up and down the stairs
    - If possible, patients should try to avoid...

- Famous violation of dogma: Captain Kirk
  - “Its five-year mission, to boldly go…”

- Famous over-use of dogma: Chief Justice Roberts
  - “…will execute the office of the president of the United States faithfully…”
Red Flag – Provincial and Sexist Language

- Unless you are sure your readership is homogeneous, be sensitive to and inclusive of many cultures and both genders

- Example a list of names:
  - “Bill White, Ken Williams, and Bob Smith”
  - “Chris Amini, Lea Sanchez, and Rei Chi Lee”

- Simple rule: if “that” sounds OK, use it!
- Use “which” with nonrestrictive clauses
- Use “that” with restrictive clauses
- Example:
  - “Ed’s country house, which is located on five acres, had bats in the attic.”
  - “The house that sat on the top of the hill had bats in the attic.”
Red Flag - Utilize

- Why would anyone utilize the word “utilize” when the word “use” works just as well?
Things to Check – Appropriate Style

- Amount of detail
  - Verbose vs. terse

- Formality
  - Formal vs. informal
  - Use of contractions, informal language, slang

- Tone
  - Distant, warm, familiar, intrusive

- For whatever style is chosen, *consistency* is very important
Examples

The component type which contains the math definition of the component is stored in the file called Type.h which provides the client with the general component description.

The component type, which contains the math definition of the component, is stored in a file called Type.h. This file also provides the client with a general description of the component.
If users input a file has invalid header record which contains more than 13 characters, the program will pop up an error message: “Parse Error: Invalid header record”, then it terminates.

If the input file given by the user does not begin with a valid header record, the following error message is displayed: “Parse Error: Invalid header record”.
The overall is contained in `Machine_Program.cpp`

The main program is contained in the file `Machine_Program.cpp`.
If the user wishes to quit, then a “q” will be entered.

To quit, the user must enter “q”.
The function will return 0 if successful, and an integer value if an error of any type is recorded.

The function returns an integer. A return value of 0 indicates successful completion, while a non-zero return value indicates an error.
So now your wondering how these classes will work together? Well it’s pretty simple these classes will extend the functionality of Score.java and Error.java. So, what does that mean? Well, it means that...

Please give me an E.
Summary

- Technical writing requires work, practice, skill, technique, time; *not talent*.
- The first draft is *always* bad writing. Allow time (and energy) for revisions.
- There is no substitute for having something to say. You can’t bluff it.
Time

Bonus material
Interval vs Point

- Different questions:
  - “How long did it take to run 5k?”
  - “When is our final exam?”

- Answering “how long?” is easy
  - Count the number of elapsed seconds
  - Easy to code

- Answering “when?” is tricky
  - 8 am (Dec 17, 2014) is not sufficient
  - Meaning depends on geolocation!
  - Even dates (Dec 17th) have this problem
Solving Time/Place Problem

- Fix one place on earth, and use *that* location's time
  - We agreed (in 1884): Greenwich, England
  - Same location as used for longitude
  - "Prime Meridian" of longitude (ie 0°)
    - Aside: What are the co-ordinates of the oval?
  - Used to be called "Greenwich Mean Time"

- Example
  - CSE 3901 final exam is at *1 pm* on Dec 17, 2014
  - So why does it say 8 am on SIS?
Notation: Encoding Date/Time

- Computer scientists understand the importance of *representation/encoding*
- Big Endian
  - `year-month-day hour:minute:second`
  - *Example*: `2014-12-17 08:00:00`
  - Benefit: lexicographic = chronological
- **Start at 0, not 1**
  - Non-CS folks call this a "24-hour clock"!
  - CS folks call this... normal
  - 00:00 is midnight, 12:00 is noon
  - Benefit: Avoids am/pm ambiguities
Time Zones

- Unfortunately, people want their clocks to show 12:00 when the sun is "highest"
- **Solution**: time zones
  - Politically defined region that uses the same offset from Greenwich
- **Abbreviations**
  - EST = UTC-5:00 (**Standard**, ie winter)
  - EDT = UTC-4:00 (**Daylight savings** summer)
- **To report a time, append** time zone
  - 2014-12-17 08:00:00 EST
  - 2014-12-17 08:00:00 UTC-5:00
  - 2014-12-17 13:00:00 UTC
Mixing Intervals and Points

- Mapping between these is difficult
- Eg: Run a task every day at 9 am
  - Naïve solution: java.util.Timer’s schedule
    schedule (TimerTask t, Date first, long period)
  - Period is an interval (number of milliseconds)
    schedule(job, today, 86400000);
- Problem?
  - Not every day has 24 hrs!
  - Daylight savings ==> a day can be 24, 23, or 25 hours long
From Intervals to Points

- Measure interval from a fixed point
  - Called “epoch”
  - Needed for both date (BC/AD) and time

- Unix: chose Jan 1, 1970
  - long time_t, count of elapsed seconds
  - What time is it? Approx. 1,416,580,000
  - Stored as a (signed!) 32-bit integer
    - “max time” = $2^{32}-1 = 2.1$ billion = 68 years!

- Will overflow on Jan 19, 2038

- Solution: use 64 bit!
  - Postpones the problem for 290 billion years...
This hour has ?? minutes...
Seconds, Minutes, Hours, Days

- Days do not divide years evenly
  - About 365.242199 days/year
- But seconds do divide days evenly!
  - Exactly 24 * 60 * 60 = 86,400 s/day
- Why?
  - Days & years are set independently by nature
  - Seconds are our invention
- How long is a second?
  - Defined to be 1/86,400th of a day
  - SI second = 9,192,631,770 oscillations of a caesium-133 atom (at rest, sea level, 0 Kelvin)
  - Just one problem... how long is a day?
1st Problem: Apparent Solar Day
2nd Problem: Elliptical Orbit

- We are closest to the sun in winter
  - Speed of orbit \(\propto\) distance to sun
- Earth is also tilted
  - Sun moves along ecliptic
  - Earth rotates along celestial equator
- Result: A “24 hour” day varies in length!
  - Can be +/- 30 seconds of average length
- Even worse: Variation is correlated!
  - Long days are consecutive during the year
  - Difference (local noon vs watch) accumulates
- Result: Net difference of +/- 15 minutes
Equation of Time
Do We Care?

- The equation of time lets you correctly convert time to/from position of sun
  - At what time will be “local noon” today?
  - See: [www.timeanddate.com/sun](http://www.timeanddate.com/sun)

- This only matters if you care about the exact position of the sun any given day!
  - So mostly no one cares

- All we need is average length of full day
  - A “mean solar day”
  - Horizontal axis in graph of equation of time
  - Measure it, super accurately, then divide by 86,400
Now For the *Really* Bad News
The Earth is Slowing Down

- Planet has been slowing down (and will continue to slow down)
- Today's "mean solar day" is *longer* than it was 200 years ago!
  - We use the mean solar day of 1750-1892 (averaged)
- Bad news: There are a bit more than 86,400 SI seconds / mean solar day
- *Really* bad news: We can't predict the size of this effect very far into future
GMT vs UT1 vs UTC

- **GMT**: Greenwich mean time
  - Antiquated: Should not be used today

- **UT1**: Universal time
  - Time at prime meridian
  - Determined by celestial movements

- **TAI**:
  - Ticks in SI seconds
  - Was equal to UT1 in the early 70's

- **UTC**: Universal Coordinated Time
  - Ticks in SI seconds, like TAI
  - Periodically *modified* to match UT1
Leap Seconds

- Mean solar day is longer than 86400 SI seconds
  - Tidal forces have slowed the rotation of the earth
  - Must correct clock time to stay synched with solar days
- Leap second: 1 second insertion/deletion
  - Irregular occurrence, UTC decides
  - Based on observation, impossible to predict
  - Since 1972, there have been 25 additions, no deletions
  - Most recent: June 30, 2012 (an addition)
Leap Second Episodes

<table>
<thead>
<tr>
<th>Year</th>
<th>Jun 30</th>
<th>Dec 31</th>
</tr>
</thead>
<tbody>
<tr>
<td>1972</td>
<td>+1</td>
<td>+1</td>
</tr>
<tr>
<td>1973</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>1974</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>1975</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>1976</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>1977</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>1978</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>1979</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>1980</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1981</td>
<td>+1</td>
<td>0</td>
</tr>
<tr>
<td>1982</td>
<td>+1</td>
<td>0</td>
</tr>
<tr>
<td>1983</td>
<td>+1</td>
<td>0</td>
</tr>
<tr>
<td>1984</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1985</td>
<td>+1</td>
<td>0</td>
</tr>
<tr>
<td>1986</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1987</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>1988</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1989</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>1990</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>1991</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1992</td>
<td>+1</td>
<td>0</td>
</tr>
<tr>
<td>1993</td>
<td>+1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Year</th>
<th>Jun 30</th>
<th>Dec 31</th>
</tr>
</thead>
<tbody>
<tr>
<td>1994</td>
<td>+1</td>
<td>0</td>
</tr>
<tr>
<td>1995</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>1996</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1997</td>
<td>+1</td>
<td>0</td>
</tr>
<tr>
<td>1998</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>1999</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2000</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2001</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2002</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2003</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2004</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2005</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>2006</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2007</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2008</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>2009</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2010</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2011</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2012</td>
<td>+1</td>
<td>0</td>
</tr>
<tr>
<td>2013</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2014</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Total: 25
Current TAI - UTC: 35
1 minute ≠ 60 seconds

- Addition/removal occurs during the last minute of Dec 31 or Jun 30
- Those minutes have 61 or 59 seconds!
  23:59:58, 23:59:59, 23:59:60, 00:00:00...
- Screen capture of the clock at time.gov during a leap second:

Right now, the official U.S. time is:

23:59:60
Saturday, June 30, 2012
Accurate within 0.2 seconds
More Complications

- GPS satellites don’t reset their clocks
  - GPS time was equal to UTC time in 1980
  - Since then, has missed 16 leap seconds
  - [http://leapsecond.com/java/gpsclock.htm](http://leapsecond.com/java/gpsclock.htm)

- Unix time *decrements* during leap second
  - Monotonic timer provided by NTP protocol

- Not all countries have adopted UTC

- Leap seconds will become more frequent
  - Proposals to abolish, replace with leap *hours*
And we care because...

The Inside Story of the Extra Second That Crashed the Web

By Robert McMillan and Cade Metz 07.02.12 7:54 PM
Summary

- **Intervals vs points**
  - Intervals are easy, points are tricky
  - Unix time: Seconds from 01/01/1970
  - Date/time is coupled to geolocation

- **Interval between 2 points is hard**
  - # days / year can vary
  - # hours / day can vary
  - # seconds / minute can vary

- **Standardization**
  - Mean solar day, SI seconds
  - They don't match: need leap seconds

- **UT1, UTC, TAI**