CS CAPSTONE

Grading Technical Writing Quick-Reference

Eck Doerry

Overview

One of the main learning outcomes targeted in the CS Capstone sequence is technical communication, i.e., effective technical writing and strong oral technical communication skills. As instructors, we have all had to learn to write effectively and have developed our own styles and priorities in technical writing. This document is not an attempt to erase those stylistic differences!

At the same time, students benefit from consistent feedback and expectations between teams and between faculty mentors. Thus, this document draws on 15 years of experience teaching Capstone to just lay out a few of the major areas of trouble in student technical writing within the CS Capstone, and to share some of my thoughts on "best practices". The hope is that this will sort of get us all on the same basic page with regard to expectations…and bring to conscious awareness what we all do already in our own writing.

Trouble areas and Best Practices

There are many entire books dedicated to describing the myriad details of what makes good technical writing. The problem is what exactly constitutes "good technical writing" varies tremendously by domain, audience, and intent of the document. Plus, who has time to read whole books?! In the following, I describe not so much precise detail, but overall "trouble areas" that I've noticed that our students consistently have…and give my $0.02 on what I expect in those areas.

The easy stuff: simple formatting expectations

Let's start with "easy" problem areas that center around proper and effective formatting and presentation of technical content. The following are the things here that I comment/red-line the most:

Cover sheet and presentation: There is continual emphasis in CS476 that this must be considered as a realistic consulting experience. Unlike normal classes, where the cover sheet is purely functional to convey key info, here it is the "tuxedo" for your deliverable in the real world. It should be really nicely formatted, with a nice team logo, and preferably in color where applicable. Deliverables longer than five pages should be delivered in some sort (could be cheap one) binding, rather than stapled. Basically what you'd expect a pro consultant to give you.

Headings and subheadings: A reader should be able to rapidly flip through the pages of a technical documents and *see the flow of argument*, just by reading the headings/subheadings. As you're reading, the headings/subheading show whether you've moved to a new topic, or are going into more detail on the current one. We all have different style preferences for how we like to communicate this. I personally prefer:

- Use of a different (sans serif) font for headings, like Arial or Helvetica. While the body is usually a Times variant.

- Different sizes/typeface for headings. Larger (+2-4pts) font size for Level 1,2 headings. Use of boldface for level 1-3 headings, possibly italics for level 2+. Point is that heading level should be visibly different.
- No more than three levels (usually two), unless strongly justifiable exception. Otherwise it gets too cluttered and hierarchical. When you get down to 4th level headings, you should probably using narrative plus itemizing.
- I like the number my headings: "1.0 Introduction" "1.1 subsection" "1.1.2 Subsubsection"

You may have different preferences…and student teams may make yet different choices. It's all okay; the point is that heading structure must exist, and it must be clear and effective.

Use of bullets and itemizing: It's really common to see students write a full page or more of dense prose…with many "listings of things" buried within the text, e.g., "First, we must consider this….Second, there is this…and third, there is this." This is a persistent problem, which probably stems from taking freshman English, where you were taught to write in long blocks of unbroken narrative. Following my overall "the argument structure should pop out at you at first glance" rule, listings of things – points to consider, things to do, priority lists, whatever – should *always be extracted into itemizations*. This means "bullets" for unordered listings, and items (usually numeric) for ordered listings.

Use of diagrams and figures: This is another area of weakness for students. They have somehow grasped that "diagrams/pics are good for my grade", and so they often paste something in there, more or less randomly. Here are my key peeves about them:

- They must have a label! At very least something like "Figure 3.2" or "Table 2", and preferably with a small caption as well, e.g., "Table 2: Distribution of widgets in the field".
- They must be referred to in the narrative. You can't just paste things in apropos nothing! Presumably your diagram/table supports some point you're trying to make in the narrative, so you must connect the two. Look for "As indicated in Fig. 1," or the like.
- They should preferably appear *before* the point of reference in the text. Sometimes page breaks and other formatting make this impractical; generally they should appear on same or very next page as reference.

Again, we probably all have our own detailed preferences on how to present graphics, but we can at least all agree that they have to be a coherent part of what's going on in the narrative.

White space issues: In general, you should not have large areas of white space in your technical document, with the possible exception of very large documents, where you might want to start major sections on a new page. It is common to see sloppy management of white space, particularly in relation to graphics and tables, resulting in random huge gaps of white space. Professional presentation means formatting these in a way that breaks happen naturally, text flows around smaller graphics, etc.

The harder stuff: Telling a coherent story

The last section focused on fairly mechanical things that are easy to teach and learn. Of course, quality technical writing is much more than good formatting; you could have an fantastic-

looking document that nonetheless reads like a migraine headache and leaves you with little clue of what it's about.

The story-telling metaphor: In talking to students, I hammer (hammer hammer) the idea that every document and every presentation has to tell a clear, coherent story from start to finish…and preferably a compelling one at that. I tell them to think about a favorite movie or book: it has to hook you in the first few pages/minutes, make you believe in its integrity/value, and then lead you coherently through the entire story. Movies that jump around between timelines and characters or have disjointed chapter topics are very challenging to follow…and often hard to enjoy. **In a technical document, you want to get your message across with as much clarity and as little reader challenge as possible.** Thus, the whole thing has to flow coherently, keeping the reader unconfused and well-oriented throughout. Here are the key problem areas I see the most:

Intro to the project/document. This is HUGE…and a continual problem for teams. The tendency is to skip the intro and just dive into technical detail. This leaves the reader saying "what the heck are we talking about anyway"; there is no big picture. My teams spend the whole term re-writing and refining their intro (usually about the 1-2 pages), which motivates the domain, explains the sponsor's business workflow in that domain, and describes the nature of the problem. Depending on the document, the intro may also include an overview of the solution (unless this is done in a separate subsequent section). If they develop a really good intro to domain/sponsor/problem section, they can adapt it to lead every document they deliver.

Segues. A "segue" is a connection between parts, and these are key elements of good "argument flow" in technical writing. Just as the main intro sets the stage for the story, small segues (a short paragraph, just a sentence or two) re-orient the reader in the argument flow, smoothing the transition between (sub)sections. Every sizeable section should have one, e.g., something like "Having gained a general overview of the system architecture in the previous section, the subsequent sections now present a detailed analysis of all major modules". See? It tells the reader "here is where we are now…and this is what's coming next".

Hierarchical levels of detail. Appropriate "flow" of detail granularity is critical to keeping your reader unconfused. You can't just start right in by dumping out low level technical discussion or, conversely, jump abruptly from really detailed discussion in one sentence to vague high-level stuff in the next. *In general, I look for "well-signaled, progressive deepening".* That means making sure the reader knows where you're headed by starting with broad sketches of a document/section topic at the front of the doc/section, closing these by signaling (e.g. in intros/segues) what's coming up, and then presenting deeper detail in subsections. What teams often do is just abruptly start a section by diving into high detail on a specific topic…no segue, no intro, no overview. This leaves the reader with little basis for understanding or evaluating all that detail; you've lost the storyline. Often this results from teams assigning subsections to different people to write…but not putting in the critical editorial effort to seam it together and make it all flow coherently. This renders the document an incoherent collection of unconnected puzzle pieces that, however decent individually, are impossible to get a coherent story from. This usually pushes the document immediately into the 'C' range for me.

Quality professional prose. Let's face it: *how* you are able to express things is just as important to the message as *what* you say content-wise. The credibility (and rewards that go

with it) of technical writing are based on the level of English used, clarity of expression, and…well…just how good it reads. Here are a few of my peeves:

- Use of slang. This is a common problem with students; they don't take the time to formulate carefully, and just throw in slang, e.g., "this choice was a no-brainer" or "we all thought this solution was lame". Credible technical writing should avoid this in favor of clear expression of what you're trying to say.
- Bad grammar and spelling. There can't be any excuse for this in the age of smart word processors, plus a whole team of proof-readers. Crappy spelling/grammar instantly reduces your credibility.
- Fragmented, poorly-constructed sentences. You often see sentences that have missing subject or object, or use an ambiguous pronoun to refer to…who knows what.
- Choppy writing. Often you'll see whole paragraphs of short, poorly-connected sentences. Very hard to read!
- Coherence. This is "argument flow" at the lowest level. A paragraph of narrative should read and flow naturally, with one sentence cohering with the next. If you find yourself re-reading and struggling to get what they mean, you're having this problem. I usually comment with "weak/confusing writing. Must re-write".

Effective use of conclusion sections. Every story has to have an end; at the end of a technical tale, you need to revisit the highlights of what was presented. We all know why: time-pressed readers may read only the intro and conclusion sections. Even readers who've slogged through the whole thing need to be reminded of the big picture implications after wading around in previous detail. There are two places this becomes relevant:

- Internally, for long sections. If a section was several pages long, with many subsections discussing details, then the section needs to be closed with a short conclusion paragraph that summarizes and sews together the key points of the preceding discussion in that large section…and (segue!) primes the transition to the next topic.
- Document conclusion. Teams have real problems here, often just throwing in a sentence or two or (!!) simply leaving off the conclusion altogether. It often feels like they are just tired of writing or feeling lazy. What I look for in a document conclusion is:
  1. Quick reminder of the domain importance and clients role/problem in this. Reminds how compelling it is to do this project.
  2. Summary of what was presented in this document. Usually just an overview sentence or two, plus 3-5 bullets of key points/conclusions drawn.
  3. Closing discussion. This brings it together by saying something about the effect/value of the findings presented on the project as a whole, e.g., something to the effect that "As a result of our technical analysis and, in particular the outcomes summarized above, it is clear that <implications for the project>". The final paragraph should make a positive statement of how the discussion presented in the doc will affect the outcome for the client/project.

Parting Shot

Again, this document is not meant as a detailed "style guide" for CS Capstone writing, to be follow by everyone in draconian detail! Teams have simply reported getting inconsistent level of feedback that often emphasize wholly different things from different faculty mentors. Thus, my

goal here has been to provide a “reminder” document, based on my own long experience, that faculty can simply (re-)scan at the start of each Capstone season, to sort of prime everyone’s brain on what to look for. Individual preferences will exist and are great; there’s no one right way to present something, and it’s nice to explore alternatives to find one’s own style.

Happy marking!