Reflection Report on BeamBending

Team Drasil
Jason Balaci

April 20, 2023

1 Changes in Response to Feedback

Please note that all feedback is recorded on the issue tracker used to manage my project. I will only be discussing a select few things here.

1.1 SRS ~~and Hazard Analysis~~

Note: we did not do a Hazard Analysis, and so I've striked it out of the subsection title.

The original structure of my SRS remained in tact after feedback primarily because I had Dr. Smith's assistance (and textbook [1]) to give me advice from the start.

The majority of the feedback related to my SRS noted minor mistakes, (my) misunderstanding of the intent behind some sections, or issues related to theories and their explanations. The explanations of my theories still need work, but I feel a lot more comfortable with them. Dr. Smith's feedback along with Sam and others has cleaned up my document significantly. A notable change is the Software and Physical constraints related to my inputs (where I previously had none) and Dr. Smith suggested concrete ones with ample justification.

The feedback given to me during the presentations was especially beneficial at the start, when I was still working on my SRS. I'm very thankful that I had the very first SRS presentation, which led to fruitful discourse and concrete advice that allowed me to continue to write my SRS. I think that taking the advice initially given to me, allowed me to later receive less problematic feedback later.

1.2 Design and Design Documentation

Since my software is relatively simple, the design and design documentation related to my prototype was minimal and there was no real feedback that needed to be addressed (other than using realistic sizes of forces).

Related to the Drasil implementation of my SRS, the feedback is ongoing, and I'm continuing to try to get my project working fully with code generation.

1.3 VnV Plan ~~and Report~~

Note: Sam and I did not do a VnV Report, and so I've striked it out of the subsection title.

Other than general advice on cleaning up my VnV Plan, fixing grammar and typos, adjusting numbers, and other minor issues, the nontrivial feedback focused on clarifying tests (specifically those non-functional) where I deferred them to Drasil and the Drasil research team. I tried my best to clarify why, but I believe that I will need to discuss with the Drasil team further about *how* and *why* I felt it was appropriate to defer the onus of testing to Drasil. If the Drasil team believes it was inappropriate/incorrect to do so, then I will also ask about and explore how we can make it appropriate/correct to do so.

2 Design Iteration (LO11)

Unfortunately, the software has not yet been created. The prototype is too simplistic (and reductive) to be notable to discuss here[1]. Furthermore, the design of the solution program is mostly decided by Drasil (up to user configuration options).

3 Design Decisions (LO12)

When we started the project, our scope was slightly larger: focusing on building solvers for general distributed load application functions. However, we quickly had to shift into focusing on only one kind, with a numerical solution as opposed to analytical: third-order polynomials. Since we do not generate software yet, we will need to revisit this section when we can.

One other notable constraint is that I did not want to compromise the quality of the SRS (nor background mathematics) in favour of code generation when converting the SRS into the Drasil-encoded form. This constraint restricts my usage of Drasil a bit, specifically with regard to solving differential equations, where Drasil's code generator expects the SRS to mostly provide the solution design in a readily digestible format[2]. This also means I was using functionality that wasn't fully flushed out yet in Drasil (function definitions, vectors, and manually-written code snippets).

The remainder of the "design decisions" are deferred to Drasil which has its own ready-made program design to solutions for SRS-captured problems.

[1] It is a single file that calculates the solution to the deflection problem, but it does no validation on inputs and does not adequately format outputs.

[2] For example, "functions" are expected to be represented by vectors to capture sampling points across a numerically-solved differential equation.

4 Economic Considerations (LO23)

I believe the market for my project would primarily be educational. The online educational scene has plenty of free and open source resources for typical beam configurations taught in an undergraduate degree. One notable difference is that we *technically* support general loading functions but are constrained by the ability of Drasil to generate programs with complex input terms (here, general mathematical expressions). Should Drasil be able to read in general expressions, then the program might have a relatively uncommon feature in free and open source programs.

Since the program is specifically intended for educational purposes, it likely won't have a market on its own. However, the impact (along with the other case studies) on Drasil's market impact might be fruitful for Drasil. If, for example, a company were interested in using Drasil to analyze beams or other various architectural or engineering-focused tasks, then Drasil (or at least the Drasil ethos) might be able to change current industry development practices.

Drasil itself could be something that is "sold" on a consulting or service-as-needed basis. Companies involved with operating systems or managing data centers for profit (Amazon, Apple, Microsoft, Red Hat, Canonical, Cloudflare, and such) could benefit from having Drasil take a front seat in their software development stack. Using Drasil (or the Drasil ethos) should allow them to make unparalleled positive impacts to their software quality and even decrease their (current) variable and long-term expenses.

5 Reflection on Project Management (LO24)

5.1 How Does Your Project Management Compare to Your Development Plan

As I am the sole team member of my project, there were no team activities (only personal). I have professional experience as a software developer and project manager, and so I followed my normal strategy for developing software[3]. Professionally, I prefer to use JIRA, but GitHub's basic issue tooling sufficed for my project. Co-ordination with reviewers and Dr. Smith was quite smooth as our class was quite friendly (and so in-person chats were frequent) and active on GitHub (and so online chats are productive).

[3] My strategy is fairly conventional: understand the background problem domain, understand objectives, break down objectives into a series of tiny "calls to action," gradually break down the tickets in an adaptive manner (focusing on easy tickets at times to get the ball rolling, or the normal logical order to make sure that everything is "good" the first time around), and revising objectives, work, and calls to action (with external reviewers!) until I've achieved the objectives.

5.2 What Went Well?

I'm quite comfortable with all tools I used during this project, and so I was comfortable with all project management efforts, processes, and technology used. Thankfully, during this course, I was able to focus my time more on understanding the SRS scheme that Dr. Smith theorized. I'm confident it has made me a better software developer. While it might be difficult to get hypothetical, future companies I work with to adopt the SRS-first approach to software development and the Drasil ethos, I am confident that I will enforce some sort of usage of it. The SRS brings upon a coherent line of questioning and problem reduction and exploration that I've never had the chance to do in the past. Admittedly, quite a lot of my development experience feels chaotic in comparison to the principled, document-driven approach that this class encourages us to take.

5.3 What Went Wrong?

I don't believe there were any processes or technology that I had difficulty with where I would claim that "something went wrong." One might claim that the difficulties I experienced with Drasil might go here, but I will push that to the design section (Section 3) since we (Dr. Smith, Sam Crawford, and I) anticipated problems. Thankfully, I was able to spend the majority of my time without fiddling with tools to get past issues.

5.4 What Would you Do Differently Next Time?

This is my first time using a principled document-driven approach to software development. The next time I will do it again will likely be just practice where I can try to make further improvements to my work plan. However, before the "next time," I will make sure to make suggest (or implement) some quality of life improvements to Drasil to accommodate me. I've already made tickets on the Drasil's issue tracker and I will likely be making more. One important thing I would like to have for future authors is a "guided tutorial for learning about Drasil," where users are encouraged to create a local development environment with Drasil, start the guided tutorial template, and use it as a basis for their own projects.

In the future, I will definitely recommend colleagues to adopt the document-driven approach to software development.