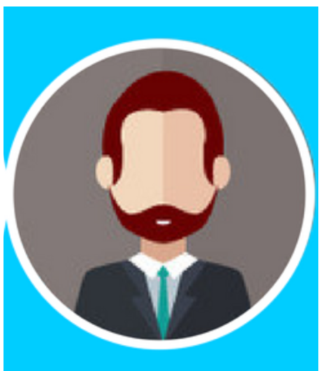
Project Proposal
Team one

Table of Contents

Team Members

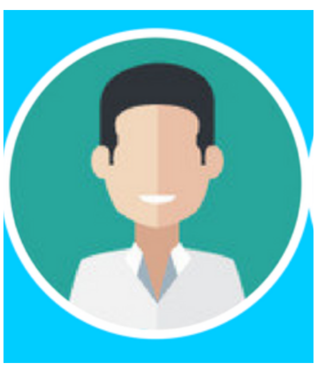
Tyler Pomietlarz

- **Student Number:** T00052042
- **Background:** Fourth-year Bachelor of Computing Science student. Wolfpack Alumni. Successfully completed 11 months of Co-op / Full-Time work with Teck Resources in Fernie and Kamloops.
- **Specialties:** Industrial Processing Control Systems, Cyber Security, abstract project management, time management, support



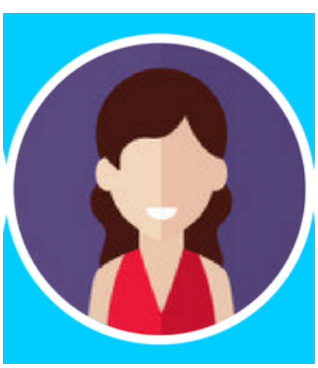
Kai Theobald-Coates

- **Student Number:** T00591522
- **Background:** Third-year Bachelor Of Computing Science Student. Currently employed as a Student Research Assistant.
- **Specialties:** Audio production, synthesis, programming



River Williams-More

- **Student Number:** T00230616
- **Background:** Fourth-year Bachelor of Computing Science. Graduated an Academy of Arts majoring in media arts.
- **Specialties:** Media Arts, design



Camila Castillo

- **Student Number:** T00564151
- **Background:** Third-year Bachelor of Computing Science. Successfully completed an 8 month Co-op with the City of Edmonton on Networks
- **Specialties:** Programming, documentation, design, networks and project planning

Problem

Since the 1960s when synthesizers started being developed, they have been a staple of modern music, especially the subtractive synthesizer. Entire genres revolve around the synthesizer, and now almost all music employs synthesizers to some degree. With the advent of cheap computing and personal music studios, it has never been easier to create music with synthesizers - you can do it with nothing but a laptop and free software.

However, synthesizers can be very daunting to newcomers. They are not intuitive to use, and many people who want to make music with synthesizers may be turned off by their apparent complexity.

Unlike most musical instruments, synthesizers do not have discoverable interfaces. If one doesn't know how to play guitar, through experimentation it is easy to figure out how guitars work - you can discover that plucking a string creates a sound, and pressing your fingers on the fretboard will change the pitch. Same with a piano - pressing keys make a sound, pressing them harder makes a louder sound. However, synthesizers have arrays of knobs and labels which, while descriptive to someone knowledgeable, are very cryptic to new users. Because different parts of the synthesizer interact with each other, many knobs and switches on a synthesizer will have no immediate effect on the sound. Also, there is no consistency in labelling across different synthesizers - depending on the synthesizer, the same parameter may be labelled "Resonance", "Res", "Peak", or even just "Q".

Because of this, subtractive synthesizers are very difficult for new users to understand, despite the fact they are conceptually simple. Teaching someone to use a subtractive synthesizer is not a hard task, but learning yourself through experimentation can be a struggle. Hence, many people who want to make music with synthesizers are turned off by the initially steep learning curve.

Fig 1: Native Instruments' Pro-53 software synthesizer, emulating a Sequential Circuits Prophet 5. Note that this relatively simple synthesizer has a layout that would be very confusing to a newcomer.

Motivation

Brainstorming session 1
We want to build an audio synthesizer software program. Subtractive synthesizers are complex hardware and software systems and our goal is to develop a set of interfaces and modes that will help new users quickly advance their audio production skills. The project ideas gathered from the brainstorm session were broken down into sections:

Tutorial

- Intro mode with a minimal number of dials available to experiment with.

Interfaces

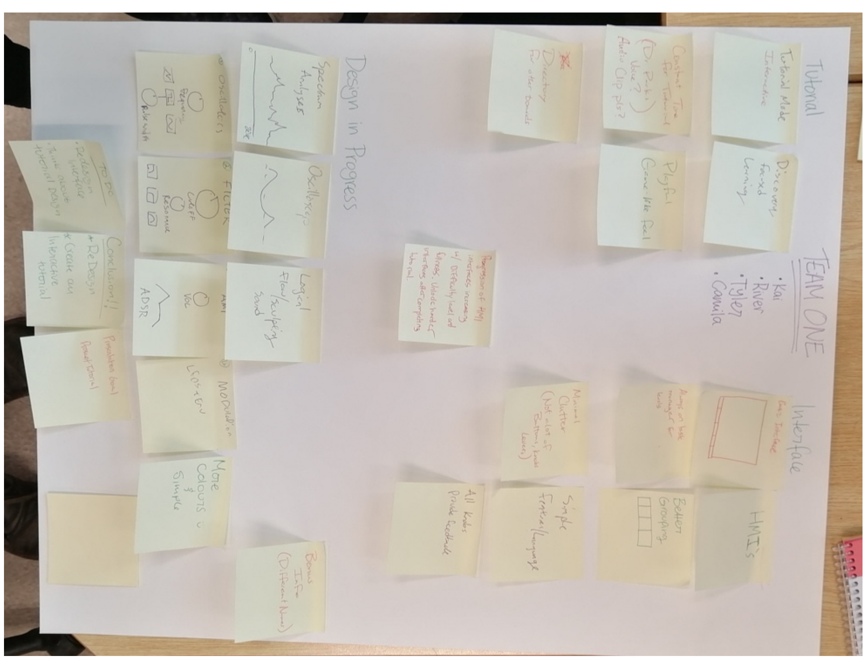
- Minimalist design with bottom tabbed navigation bar and logo upon program load
- Visual audio output breakdown draggable sub interface similar to Task Manager processes
- Visual Directory for other interfaces
- Tight and highly contrasting button grouping
- Minimal clutter
- All knobs will provide feedback

Design Progressions

- Progression of HMI's with ascending interface difficulty. The further a user progresses through the material the more options they have available on the GUI's.

Overall vision

- Progression of HMI interfaces increasing with difficulty level and fullness. Unlock harder interfaces after completing the tutorial.
- We were able to accomplish this by staying focused during the brainstorming ideas and building on each other's ideas. This is what we were able to accomplish in the first 20 minutes.



Brainstorming session 2

- We decided on the team name and started working on the tutorial design and made a few drafts on the synthesizer's interface.

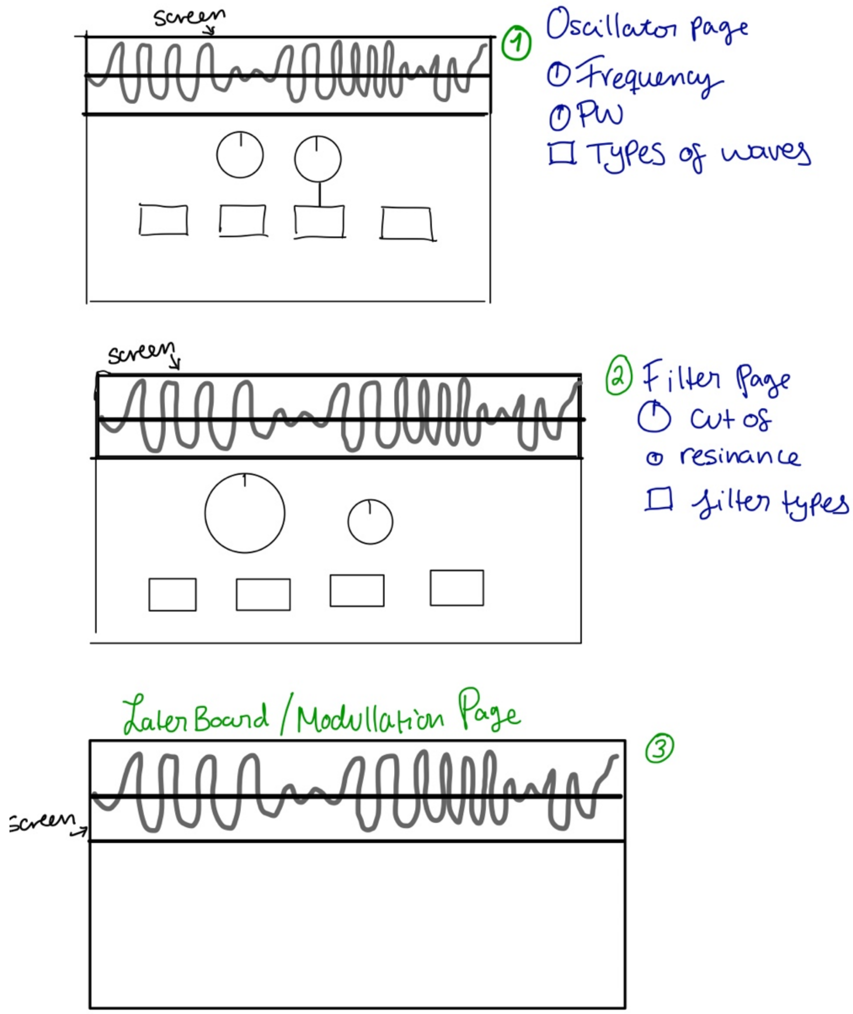


Fig 2: Early sketches of our initial design. Note the multi-page interface, the constant visual feedback provided by the oscilloscope, and the low number of parameters per page.

Research

Most synthesizers are designed for an audience of people who are already familiar with synthesizers, without much consideration for novice users. Because of this, most synthesizers look very confusing to an untrained eye. Without an understanding of the underlying structure of a synthesizer, the layout and labels do not help a user, or give them any affordances as what to do.

Broadly, there are two categories of subtractive synthesizers. Software synthesizers have the benefit of being low cost, or even free, but have no tactile feedback. Hardware synthesizers have the advantage of being physical and tactile to operate, but they are expensive to acquire and are therefore mostly inaccessible to newcomers.

The first synthesizers were Modular Synthesizers. These synthesizers were large cabinets filled with “modules”, individual components of a synthesizer. These modules included oscillators, filters, amplifiers, envelopes, step sequencers, and many other components. In order to create a sound, you had to use cables to plug components into each other in the order you wanted them in the audio chain. This gave the user complete flexibility, but also made these synthesizers difficult to use, as the variable audio path meant that you had to comprehensively understand each element of the synthesizer to make any sound at all. Additionally, these synthesizers were expensive, and generally only owned by universities and research groups

Fig 3: A Moog Modular Synthesizer. Note the multiple modules, and the ¼ inch sockets for patching an audio path.

In response to this, the company Moog Music introduced the Minimoog Model D. Arguably the first synthesizer designed with usability in mind, this synthesizer featured a fixed audio path that would become the de facto standard for subtractive synthesizers - oscillators into filter into amplifier, with ADSR envelopes and oscillator modulation available for these sound generating components. This fixed audio path gave the user constraints to work within and also ensured the user had a consistent, working audio path, which made the synthesizer much more usable for musicians and people unfamiliar with the internal workings of a synthesizer. The lower cost also put more of these instruments in the hands of actual musicians. However, there is still a steep learning curve, and the labelling and layout didn't give the user much understanding of what was actually going on behind the front panel of the synthesizer.

Efforts have been made to make simple, introductory synthesizers. One of the most classic examples of an easy-to-use synthesizer is the Roland SH-101. This synthesizer features a stripped back design, with very limited features (one oscillator, one filter, one envelope, one LFO). It is also designed in such a way that it has a wide "sweet spot" (that is, a wide range of usable states). However, there are some notable drawbacks to the interface.

One drawback is that almost all parameters are controlled by these identical sliders - for a more accomplished user, this means that you can move multiple parameters with one hand (which you cannot do with knobs), but for a novice user, this means there are no clear parameters that are more important than others, nor is there much to distinguish parameters from each other.

Another drawback is that there are few affordances given for the more complicated parameters. Parameters such as filter cutoff and noise amount have a simple, one-to-one correspondence to the final sound, but modulation parameters like the Low-Frequency Oscillator (LFO) or the Attack-Decay-Sustain-Release (ADSR) envelope have many parameters that interact with each other in subtle ways that are more difficult to divine through experimentation. The Roland SH-101 is also rare and expensive to acquire due to its "classic" status.

Fig 4: A Roland SH-101 synthesizer. Note the simple design with clearly delineated sections, but also notice the lack of affordances given to a new user

Another synthesizer that attempts to be very user friendly is Sonic Charge's Synplant. Synplant is a software synthesizer providing a radically novel interface. Instead of having an array of parameters to modify, Synplant abstracts all of its parameters away from the user. Users start off with a "seed" synth patch, and by "growing" the seed in different directions, the sound of the synthesizer changes. Using this method, the user can organically create a wide variety of usable synthesizer tones with little to no prior experience. If the user wants to be more specific, they can access the back end of the synthesizer and tweak all of the individual parameters manually.

From an ease of use perspective, this synthesizer is a huge success - anyone can create useful synth sounds very quickly, and more knowledgeable users can modify any parameters they want, giving them full control. However, this is a very poor introductory synthesizer, as its inconsistency with the informal design standards gives the user no general knowledge of how synthesizers work. Nothing the user learns while using this product is applicable to any of the thousands of other synthesizers available on the market. Additionally, at a retail price of 99 USD, it is very expensive as an introductory synthesizer.

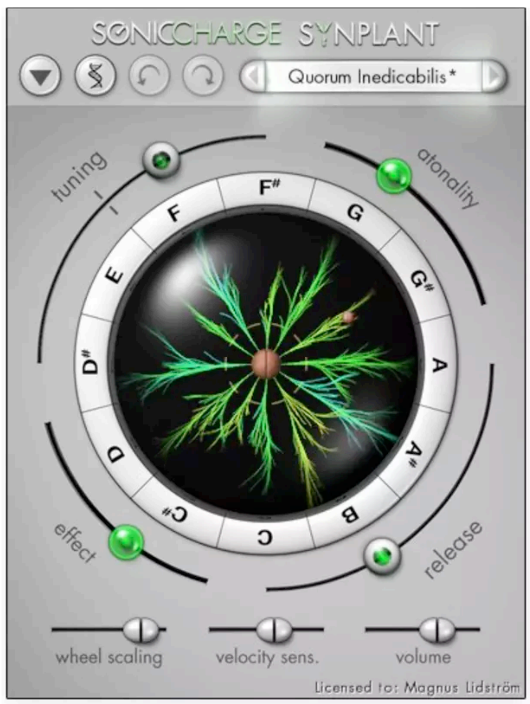


Fig 5: Sonic Charge's Synplant software synthesizer. Note the radical interface with few, well labeled parameters and unique method of interaction, but also notice that the radical interface preclude transferable skills that can be applied to other synthesizers.

From our research, we have found that there is a niche for an accessible software synthesizer, which has an easy to use interface, where strategies learned by the user can be applied to other synthesizers. We have not found any synthesizers focused on learning, discoverability, and knowledge reapplication.

Target users (stakeholders)

- Our target users are any people who want to learn subtractive synthesis, especially those who enjoy learning through experimentation and discovery.
- These users include prospective musicians, sound designers, and keyboardists, especially those who are just beginning to explore synthesis as a means of musical expression.

Solution

- As mentioned before synthesizers can be very daunting to newcomers that is why we think the best solution is to redesign the interface of a basic subtractive synthesizer and create a user-friendly tutorial based on user discovery
    - Redesign the interface so it doesn't look intimidating and it's more "obvious" and straightforward for beginners.
    - User-friendly discoverable tutorial for people to learn by doing, making it fun and interactive.
    - Give visual feedback paired with audio feedback to explain what parameters do (eg constant oscilloscope and spectrum analyzer feedback, graphical representations of envelopes and filters, etc)

Technical feasibility

We plan on building a digital subtractive synthesizer that runs on Windows as either a standalone application or as a Virtual Studio Technology Instrument (VSTi), which is an industry-standard plugin format for running instruments and effects in Digital Audio Workstations (DAWs). Standalone software synthesizers and VST plugins have been around since the 1990s, and are well-vetted technologies.

We are planning on making a very basic synthesizer - our focus is on the user interface and on learning, not on feature set or flexibility. We believe this perspective product is well within the range of technical feasibility.

User experience goals

Our ideal user experience would be one where the user wouldn't feel overwhelmed by their options.

- **Satisfying**
  - The user can navigate the interface exactly as they intend to.
- **Cognitively Stimulating**
  - The information perceived by the user through the interface progressively motivates them to continue learning.
- **Engaging**
  - The interfaces entice the user to experiment.
- **Enjoyable**
  - Mimics the structure of video games, allowing you to unlock other features in the tutorial
  - It's appealing, engaging and increases learning motivation making the tutorial and interface enjoyable to the user.
- **Supporting Creativity**
  - The interface and tutorial will be simple, interactive and explorable. This will stimulate and support the creativity of the user.
- **Provocative**
  - The program encourages the user to think about how else they could improve their audio production skills.
  - The program will also provoke curiosity about the synthesizer and music itself and will motivate the user's desire to learn.

Project plan

Week 3	• Submit Phase 1- proposal
Week 4	• Weekly Meeting - September 21st 11:00AM COMP 1360 • Complete Phase 2- Requirements • Begin Conceptual Modelling Phase
Week 5	• Weekly Meeting - September 28th 11:00AM COMP 1360 • Start working on the interface and tutorial design
Week 6	• Weekly Meeting - October 5th 11:00AM COMP 1360 • Complete Phase 3 - Design
Week 7	• Weekly Meeting- 11:00AM COMP 1360 • Work on the prototype for synthesizer's interface • Begin Physical Modelling Phase
Week 8	• Weekly Meeting- 11:00AM COMP 1360 • Work on the prototype for the interactive tutorial
Week 9	• Weekly Meeting- 11:00AM COMP 1360 • Continue working on prototypes
Week 10	• Weekly Meeting- 11:00AM COMP 1360 • Complete Phase 4- Prototype
Week 11	• Weekly Meeting- 11:00AM COMP 1360
Week 12	• Weekly Meeting- 11:00AM COMP 1360 • Prepare final product presentation
Week 13	• Present and complete project