TLEF Project – Final Report

Report Completion Date: (2021/04/01)

1. PROJECT OVERVIEW

1.1. General Information

Project Title:	Pilot for teaching systematic compiler design in CPSC 411		
Principal Investigator:	William J. Bowman		
Report Submitted By:	William J. Bowman		
Project Initiation Date:	April 1 2020	**Project Completion Date:**	April 1 2021
Project Type:	☐ Large Transformation ☒ Small Innovation ☐ Flexible Learning ☐ Other: [please specify]		

1.2. Project Focus Areas – *Please select all the areas that describe your project.*

☒ Resource development (e.g. learning materials, media)

☐ Infrastructure development (e.g. management tools, repositories, learning spaces)

☒ Pedagogies for student learning and/or engagement (e.g. active learning)

☐ Innovative assessments (e.g. two-stage exams, student peer-assessment)

☐ Teaching roles and training (e.g. teaching practice development, TA roles)

☒ Curriculum (e.g. program development/implementation, learning communities)

☐ Student experience outside the classroom (e.g. wellbeing, social inclusion)

☐ Experiential and work-integrated learning (e.g. co-op, community service learning)

☐ Indigenous-focused curricula and ways of knowing

☐ Diversity and inclusion in teaching and learning contexts

☒ Open educational resources

☐ Other: [please specify]

1.3. Final Project Summary

In CPSC 411, students learn to implement a compiler, a large software project used to implement a programming language by translating programs into binary machine code. The prior CPSC 411 curriculum was based on a textbook published in 1998; it predates the UBC computer science (CS) focus on SYSTEMATIC SOFTWARE DESIGN, and 20 years of compiler and programming language technology. As a result, we are revising the CPSC 411 curriculum and pedagogy to reflect modern practices and technologies.

With this Small TLEF, we developed and piloted course activities and assignments that teach students to DESIGN new programming languages and compilers, in addition to implementing them. This design aspect is new; instead of simply giving students a design they then use to implement a compiler, we redeveloped the CPSC 411 assignments to include design components, incorporated design into classroom activities, and developed accompanying open course materials, including course support software and an open access book.

1.4. Team Members – *Please fill in the following table and include **students**, undergraduate and/or graduate, who participated in your project.*

Name	Title/Affiliation	Responsibilities/Roles
Lily Bryant	GTA/GAA, UBC	Redesigning weekly assignments; testing weekly assignments for scope; developing support software; collecting and analyzing evaluation data
Adam Geller	GTA, UBC	Redesigning weekly assignments; testing weekly assignments for scope; developing support software; collecting and analyzing evaluation data; updating lecture notes
Paulette Koronkevich	GAA, UBC	Assisting developing support software
Ron Garcia	Associate Professor, UBC	Assit with curriculum development

1.5. Courses Reached – *Please fill in the following table with **past**, **current**, and **future** courses and sections (e.g. HIST 101, 002, 2017/2018, Sep) that have been/will be reached by your project, including courses not included in your original proposal (you may adapt this section to the context of your project as necessary).*

Course	Section	Academic Year	Term (Summer/Fall/Winter)
CPSC 411	201	2020	Winter 2 (January)
CPSC 411		Indefinitely	

2. OUTPUTS AND/OR PRODUCTS

2.1. *Please **list** project outputs and/or products (e.g. resources, infrastructure, new courses/programs). Indicate the current location of such products and provide a URL if applicable.*

Product(s)/Achievement(s):	Location:
Finished draft of open access course book. This includes weekly milestones for building the course project, and chapters with running software examples and figures accompanying these weekly milestones.	Source: https://github.com/cpsc411/cpsc411-book Published: https://www.students.cs.ubc.ca/~cs-411/2020w2/index.html
Open source tools to interrogate the reference implementation of the course project. This essentially allows students to ask questions about the design	Source: https://github.com/cpsc411/cpsc411-interrogator Example: https://www.students.cs.ubc.ca/~cs-411/2020w2/interrogator.cgi?test=%27%28module+6%29&an=a6&traced%3F=%23t Example: https://www.students.cs.ubc.ca/~cs-411/2020w2/lang-differ.cgi?lang1=paren-x64-v2&lang2=paren-x64-v1
Open source support library for the course project and documentation.	Source: https://github.com/cpsc411/cpsc411-pub Source: https://github.com/cpsc411/cpsc411-skeletons Published: https://www.students.cs.ubc.ca/~cs-411/docs/cpsc411/index.html?q=cpsc411

2.2. Item(s) Not Met – *Please list intended project outputs and/or products that were not attained and the reason(s) for this.*

Item(s) Not Met:	Reason:

3. PROJECT IMPACT

3.1. Project Impact Areas – *Please select all the areas where your project made an impact.*

☒ Student learning and knowledge

☐ Student engagement and attitudes

☐ Instructional team-teaching practice and satisfaction

☐ Student wellbeing, social inclusion

☐ Awareness and capacity around strategic areas (indigenous, equity and diversity)

☐ Unit operations and processes

☐ Other: [please specify]

3.2. What were you hoping to change or where were you hoping to see an impact with this project? – *Please describe the intended benefits of the project for students, TAs, instructors and/or community members.*

The main objective of this project was to redevelop CPSC 411 to include the development and use of systematic designs of compilers as a core learning objective in the course, providing benefits to the students learning.

3.3. Were these changes/impacts achieved? How do you know they occurred? – *How did you measure changes/impacts? (e.g. collected survey data, conducted focus groups/interviews, learning analytics, etc.) Describe what was learned from this process. You are encouraged to include copies of data collection tools (e.g. surveys and interview protocols) as well as graphical representations of data and/or scenarios or quotes to represent and illustrate key themes.*

We believe the changes were achieved.

We integrated design lessons into the project, book, and lectures, and made assessed student learning of these design lessons in the following ways:

1. We integrated short-answer exercises into early project milestones. We asked students to explain the implication of some design decisions in the software they are asked to implement in the weekly milestone. This provided an early measure of how students understood design decisions. In the second week, approximately 1/3rd of students could identify what piece of the software was improved by a particular design decision.

2. We performed interviews early in the course with each student in we asked them about various aspects of the design of their software. The questions were designed to assess whether students could explain:

a. The overall design of the software project. Most students could explain that the software project was designed around small transformations each with a single responsibility, and could explain how these transformations related to individual language features.

b. How the design of the project addressed major implementation challenges in the software. Approximately half of students could explain which transformations were affected by various design decisions and identify the main implementation challenge in these transformations. About half of students struggled with these transformations because they did not understand the design or how the design could be used to simplify the transformation.

c. Key differences between two versions of the software project. Most students could explain the key differences in terms of the language features each version implemented, and how these features affected the user of the software.

These interviews suggested that early in the course, students were understanding the overall design of the compiler, but not yet able to connect particular design decisions to particular parts of the software implementation process.

3. The midterm exam was a coding exam and asked students to design and implement 2 small compiler transformations, given only a loose specification.

a. On the first, approximately half of students identified and solved the major design decisions successfully. This indicates an increase in competency over the early milestone exercises, where only 1/3rd could identify how design implications relate to the software implementation.

b. On the second, most students could identify the major design decisions, while about half could successfully implement the software during the exam. This suggests an increase in competency relating design and implementation compared to the earlier milestone exercises and interviews.

Some of this struggle translating their design into implemented software seems to be related to the virtual exam format and the unusual COVID stress, and some confusion with respect to the first exercise.

4. At the end of the semester, we provided students the opportunity to revise their midterm exam by explaining what design mistakes led to errors in their software, how the design could be altered, and correcting the software's design.

Half of students took advantage of this opportunity, and of those, all of them were able to identify and explain design mistakes, how to address them, and correct their software implementation. A small number of students still had minor implementation problems, and one student still demonstrated a design mistake. This suggests that by the end of the course, many students had mastered the implementation aspect of CPSC 411 and the new design aspect.

3.4. Dissemination – *Please provide a list of **past** and **upcoming** scholarly activities (e.g. publications, presentations, invited talks, etc.) in which you or anyone from your team have shared information regarding this project. Be sure to include author names, presentation title, date, and presentation forum (e.g., journal, conference name, event).*

N/A

4. TEACHING PRACTICES – *Please indicate if **your** teaching practices or those of **others** have changed as a result of your project. If so, in what ways. Do you see these changes as sustainable over time? Why or why not?*

N/A

5. PROJECT SUSTAINMENT – *Please describe the sustainment strategy for the project components. How will this be sustained and potentially expanded (e.g. over the next five years). What challenges do you foresee for achieving the expected long-term impacts listed above?*

The project materials are open source and will be maintained and updated by my team as we teach the course. The materials have also been adopted by a professor at Vrije Universiteit Brussel. Their team is contributing to the maintenance of these materials.

The primary challenge in maintaining the design learning objectives is in assessment; assessing students learning of design is significantly more involved than assessing whether they are able to implement software, since the latter requires essentially running the software to see how well it works. The assessment this year used interviews and written exam revision, which worked primarily due to the additional resources and unusually low course enrolment. We cannot rely on these additional resources in the long-term, and expect course enrolment to increase in the coming years. However, assessment of the design learning objective will be less important if we establish that the course project and materials are indeed teaching students design as they implement their project.