PristineLCA: Making Sustainability Research easier for Academics

PristineLCA, easier sustainability research

OpenLCA, the premier open source life cycle analysis tool used widely in the academic community, is powerful but extremely intuitive to use. Our solution, PristineLCA, puts the focus on users.

Ibrahim T. Miraj

McGill University, Montreal, QC, H3A 0G4, ibrahim.miraj@mail.mcgill.ca

Osman Warsi

McGill University, Montreal, QC, H3A 0G4, osman.warsi@mail.mcgill.ca

Clemence Granade

Université du Québec à Montréal, Montreal, QC, H2L 2C4, clemence.granade@mila.quebec

The world is waking up to the fact that climate change is a grave threat. Industry, academia, and society are realizing that we need to assess the environmental impact of everything that we do. A crucial part of this is doing a complete assessment of the environmental impact associated with all the stages of a product's life cycle. In other words, they need to do a complete life cycle assessment (LCA). To do this, researchers need to use LCA software. The most popular free and open-source option available right now is openLCA. While it is a powerful platform, it is extremely difficult to use. Easier to use commercial options are available, but they cost thousands of dollars. We worked with researchers to create PristineLCA, an open-source application that enables them to perform the exact same calculations with much greater ease, at a much lower cost. This will increase their efficiency and save researchers around the world crucial time and money, thus speeding up the global effort to fight climate change.

CCS CONCEPTS • Life Cycle Assessment • Open source software • UX Design • UI Redesign

Additional Keywords and Phrases: openLCA, user testing, iterative design

ACM Reference Format:
Ibrahim T. Miraj, Osman Warsi, Clemence Granade. 2023. PristineLCA: Making sustainability research easier for academics.

1 INTRODUCTION

Life cycle assessment (LCA) is a method for evaluating the environmental impact of a product from raw material extraction to waste disposal. It involves analyzing the impacts of all materials and processes involved in the product's creation [1]. Each product has an entire supply chain whose impacts need to be assessed. This is known as cradle-to-grave analysis. It is necessary to fully understand the environmental impacts of the entire supply chain and the product's impact. For example, to assess the environmental impact of a plastic bottle, an LCA would consider the impacts of oil extraction, plastic manufacturing, transportation of materials, disposal etc.

Due to the extremely long chain of impacts that needs to be examined, it is prohibitive to conduct an LCA by hand. Software is necessary. Indeed, without appropriate software, LCAs cannot be conducted [2]. Because LCA software is essential, there are a wide array of options available.

2 PRIOR LCA SOFTWARE

The current options leave a significant gap; existing LCA software is either expensive or hard to use. We examined several paid LCA software options available on the market, including GaBI, SimaPro, and Umberto. Many of these have well designed UIs and efficient workflows. However, these options have high licensing costs [3]. Costs range from $1500 for GaBI [4] to € 4,500 - €7,000 for SimaPro [5]

As a result, many turn to free open-source alternatives, such as openLCA. Of all open source LCA tools, openLCA is the only open-source software compared to commercial products in terms of functionality [3]. While openLCA is certainly an improvement on older open-source tools, its user interface has been noted to be outdated and confusing. The learning curve is extremely steep.

To address this issue, our project, PristineLCA, aims to create an interface layer that is empirically simpler to use on top of the existing openLCA backend. This will allow for more efficient LCA research and easier onboarding for new users. This should lead to a net increase in LCA research, and thus has the potential to enable improved sustainability practices through society. By making LCA research cheaper and more efficient, our project addresses key UN Sustainable Development Goals. These include SDG 9 (sustainable industrialization), SDG 12 (sustainable consumption and production), and SDG 13 (climate action).

3 CONFIRMING THE PROBLEM

We conducted informal interviews with three sustainability researchers and students at McGill University who were familiar with openLCA. These interviews confirmed various usability issues, including a steep learning curve, lack of feedback, and unpredictable software behavior. When asked about openLCA's ease of use, an LCA teaching assistant we interviewed noted, 'This is something even my professor struggles with. I did everything right and still got an inconsistent result. This is a very fundamental problem with LCA. It's not just that you have uncertainty in the data, but you have very unstable software as well'.

He also reported extreme difficulty for new students learning the software. He further mentioned that he himself struggled to provide tutorials with openLCA due to its confusing user interface and stability issues. The TA further explained, 'They should have made a step-by-step guideline, like this is the most efficient way of doing it. Because that is the key problem. There are so many ways of doing steps in openLCA that it just ends up… (being confusing)'

A student we interviewed said, 'This is what the problem is with openLCA: it is very unstable. If I've made a mistake, I don't try to correct it, I just start from scratch because fixing mistakes will cause more bugs… It is a very fragile kind of software'. These interviews confirmed the existence of usability issues within openLCA.

4 FORMATIVE TESTING AND ANALYSIS

Three 60-minute recorded usability studies were conducted using the think-aloud method to understand its specifics. The think aloud method was chosen as it is a cost-effective, robust and flexible way to gain insight into users' thought processes [6]. We used three participants, the minimum recommended number of subjects for this

type of study [7]. The study was part of a class project for a senior undergraduate course on Human-Computer Interaction and conducted under the approval of our university's Research Ethics Board.

We ensured that the user observations consisted of users completing tasks representative of what users of openLCA would typically do. This 'Standard LCA Case Study' consisted of importing a database, creating three flows, editing/deleting a flow, creating three processes, adding flows to these processes, editing/deleting a process, and finally creating a report. The case study allowed us to observe users go through the full workflow of openLCA. Note that this case study is used for all user testing in this paper.

A representative sample of openLCA users, including one complete beginner, one intermediate student user, and one expert teaching assistant was recruited. This variety in users allowed us to examine how users of different skill levels interact with openLCA. The results of our formative testing were then analyzed to guide the project's direction. First, we reviewed the recordings and listed distinguishing behaviors that we observed. Task-based user segments were created from these distinguishing behaviors using the methodology of task-based user segments as described by Indi Young [8] Empathy maps were used to track user thoughts and behaviors. This resulted in the creation of two main user groups: the openLCA learner (new to openLCA) and the openLCA expert (extensive experience using openLCA. Additionally, through analysis of user observation footage, 30 critical UI issues were identified in the current design of openLCA. These issues fall into several broad categories

1. **Lack of workflow indication**: The current UI of openLCA does not provide indications of the typical workflow for conducting a life cycle assessment, which includes database import, flow and process creation, and report setup. This lack of guidance causes confusion for users.
2. **Multiple methods for completing actions:** The presence of multiple methods for completing actions leads to UI bloat and confusion for users.
3. **Inefficient information display:** The display of relevant information across multiple windows and the prominence of irrelevant or rarely used information leads to inefficient information management and decreased user productivity. Functionality is also obscured by multiple layers of windows.
4. **Inefficient information input:** Efficient information input is hindered by long, unsearchable drop-down lists, cut-off windows, and lengthy creation wizards with multiple windows. These issues contribute to user frustration and decreased productivity.
5. **Lack of feedback to user actions:** There is a lack of feedback to user actions in the current openLCA design, such as the system freezing when aborting a database import process or the failure to reflect changes made to flows within processes unless the database is closed and then reopened. This lack of feedback causes confusion and frustration for users.

5 DESIGN GOALS

In order to improve these issues, we came up with 6 usability goals for PristineLCA.

1. **Capable of replicating core functions of openLCA**
2. **One Way to Do Things Only**: For each action that the user can take, there should be a single overarching UI element responsible for handling it to improve usability. This does not imply a lack of shortcuts.
3. **Efficient information display**: Only necessary information that is relevant to the user should be displayed. All information of the same type should be displayed together
4. **Essential information display:** No essential information should be omitted from the display. If the user is confused, there must be additional guidance available in the UI. Functionality should be clearly signaled.

5. **Feedback and Responsiveness**: When users make easily detectable errors, the system should alert them. Changes in the names of processes and flows should reflect across the UI immediately.
6. **Efficient Input:** Should be able to efficiently input information needed for an LCA as efficiently as possible.

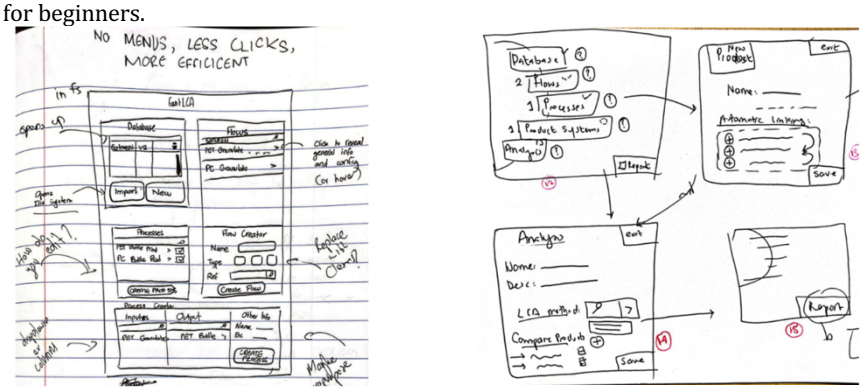
6 ITERATIVE DESIGN PROCESS

Our iterative design process consisted of four stages: brainstorming, low fidelity prototyping, computer prototyping and alpha system. Recorded think aloud usability tests with at least three participants were conducted and recorded at the end of each stage to inform design decisions in the subsequent stage. Our usability tests were always conducted with real users of openLCA to maximize the validity of our observation. These participants included professors, academics, instructors and students. Our testing tasks followed the steps of the 'Standard LCA Case Study' described in section 4.

6.1 Brainstorming

In the initial phase of the design process, we used the 10 plus 10 method as described by Bill Buxton [9] to sketch out designs. We then evaluated the usability of each sketch through group discussion and by conducting quick mock LCAs to identify potential usability issues. The most usable sketches were then refined and elaborated on through four rounds of iteration, resulting in 39 final concept sketches.

During the design process, we considered the tradeoff between sequential and parallel workflow in the software. A strictly sequential workflow (Figure 1, right) may benefit novice users by guiding them through the steps of an LCA but may be too restrictive for advanced users. On the other hand, a design that allows for a non-sequential, parallel workflow (Figure 2, left) may be more suitable for experienced users but may not provide sufficient guidance for beginners.



f

Figure 1: Examples of a single window (left) vs. a multi window (right)

The design choice between a multiple window or single window setup was also considered. Multiple windows can effectively organize information but may lead to confusion and difficulty navigating for users. Single window designs can alleviate this issue but may result in information overload. Our final design consisted of a single window displaying all panes at once. We hoped the separation of panes would prevent information overload while keeping everything accessible to the user at once.

6.2 Low fidelity prototyping

We created a low fidelity paper prototype of our UI, which we then used to conduct a usability test. The test involved a user interacting with the printed screen, while facilitators simulated interactivity by moving around pieces of paper. This method was found to disrupt the user's workflow, and thus interfere with our observation.

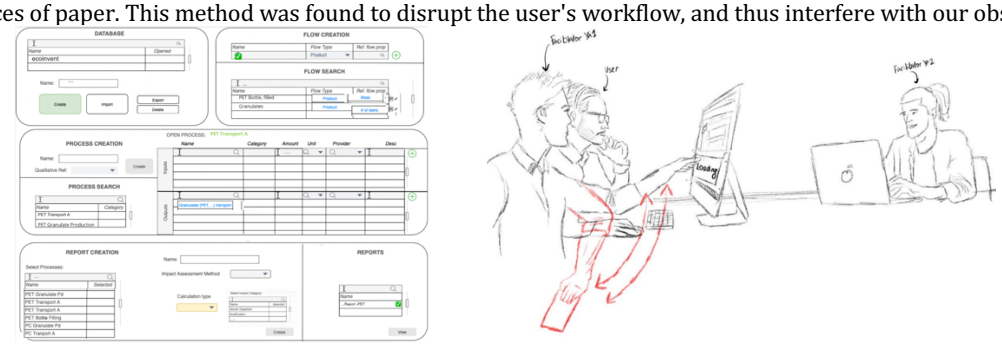


Figure 2: Low fidelity computer prototype (left), Sketch of our usability test sessions using Miro (right). In red is the movement of the paper loading 'screen

Thus, we implemented a computer-based low-fidelity prototype, using Miro.com's collaborative online whiteboard. This allowed users to interact with the prototype more realistically, resulting in the collection of more valid usability testing data. Previous research has shown that computer-based low-fidelity prototypes are more comfortable for users [10]. During testing, users input text into preset fields while facilitators move images to simulate interactivity, with a paper "loading screen" obscuring the computer display while these images are moved to prevent the user from doing anything during the simulation.

During think-aloud usability testing, it was discovered that our design resulted in users experiencing information overload, causing confusion about where to begin. This was unexpected as we had assumed users would naturally start at the Database Pane located in the upper-left corner. Additionally, it was found that users often neglected to fill in crucial input fields, which our prototype had no functionality to prevent.

6.3 Computer Prototype

We implemented changes to the design of a low-fidelity prototype based on the low fidelity user testing results. These included the addition of detailed error messages on input fields and a scrolling-through-panes layout, with only one pane visible at a time. An interactive prototype was developed using React using this design and tested with think-aloud methodology. The results of this testing indicated that users struggled to understand the purpose of each panel and made mistakes while creating flows and processes that could not be corrected in the current design. Additionally, inconsistent prototype styling was found to interfere with the user's workflow.

6.4 Alpha System

Based on the feedback obtained through user testing of the computer prototype, information icons were added next to the title of each panel. These icons provide context-sensitive text tutorials that appear when users hover their cursor over them, helping to clear up any confusion users may have had about the purpose of each panel. Dynamic edit and delete buttons were added for the flow and process pane to allow users to correct their mistakes, alongside applying consistent styling across the application

7 FINAL DESIGN DESCRIPTION AND EVALUATION

7.1 High level description

The final design for the UI consists of multiple panes for different steps of conducting an LCA. These panes are organized within a single scrollable window. To change contexts, users can simply scroll down the page as they progress through the LCA. The panes are also designed to minimize information overload by being large enough that only one pane is fully visible at a time. The UI includes searchable dropdown inputs and error message functionality for all text fields. Additionally, next to the title of each page, there is an information icon that can be hovered over, which opens a text tooltip providing guidance to users on how to use that pane.

7.2 Database Pane

The database pane allows for the import or creation of databases, which are displayed in the pane with their names and an indicator of active use. The pane contains buttons for creating and importing databases, as well as a display table for imported databases.

7.3 Flow Pane

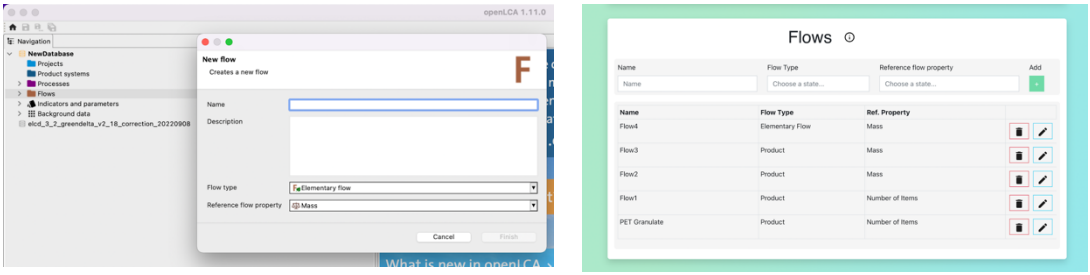


Figure 3: Flow creation in openLCA (left), Flow Pane in PristineLCA (right)

The right-hand figure shows flow creation in openLCA. Each time a user creates a flow, they must right click the flow folder (there is no indication of this fact in the UI) and click 'new flow' which brings up the pop-up window shown in the figure. One must then click through this pop up, and individually click on each text field (cannot use tab and enter keys), and then scroll through the unsearchable drop downs (which contain hundreds of entries) to find the option they want to select.

This process is much quicker in PristineLCA, where all functionality related to flows is segregated into the Flow Pane. In PristineLCA, there is no pop-up window, no need to right click on anything, users can swiftly navigate creation with tab and enter, and all dropdowns are searchable. Furthermore, if any required fields are left blank, an error message will be displayed.

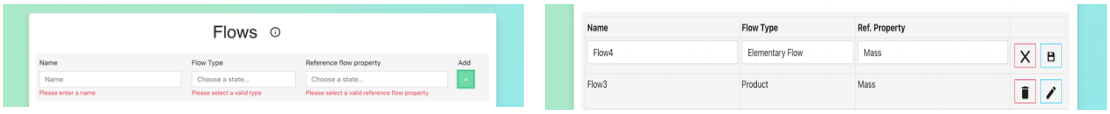


Figure 4: Example of error messages due to blank input (left). Example of flow being edited (right)

The created flows are displayed in a scrollable table, with properties shown alongside delete and edit buttons. The edit button allows users to modify a flow, and clicking the save button saves the changes. If the edit operation is canceled, the 'X' button can be clicked to return to the original flow.

7.4 Process Pane

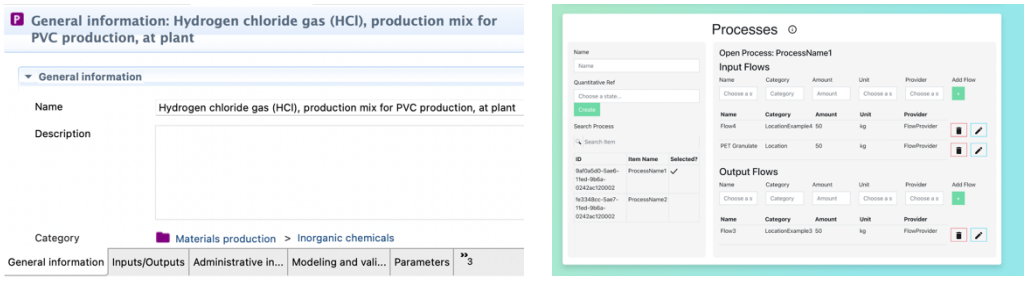


Figure 5. An open process in openLCA (left), Process Pane in PristineLCA (right)

Creating processes in openLCA is similar to creating flows; users right click the processes folder and then click through a pop up window to create a process. Then, a window with multiple tabs appears, through which the user needs to navigate to input the data needed for the LCA, which consists mainly of inserting input and output flows into the process. In short, the functionality for conducting CRUD operations on processes is spread throughout the application.

In PristineLCA, process creation and editing are consolidated into the process pane. The process pane consists of the process creation section on the upper left side, and the process selection section on the lower left side. Additionally, it includes the open process section on the right side. Processes can be created by inputting a name and selecting a quantitative reference from the searchable dropdown, and then clicking the create button. This will add it to the list below the create button. From this list, it can be clicked and it will open up on the right hand side. Once open, input flows and output flows can be added and removed from the process using a similar process as the previously described flow pane.

7.5 Report Pane

The report pane contains all the functionalities to create, update and view the final analyses. It functions in much the same way as the process pane. The upper right-hand side is used to input a name and description for a report, after which clicking the create button will generate a report.

Once the report has been generated it appears in the table below the create button. The table is a searchable, scrollable table. Upon clicking a report on the table, it will appear on the right-hand side of the screen, where it can be edited. Now the right-hand side can be used to input details about the report generation.

7.6 Design Evaluation

In order to evaluate the performance of the final design of PristineLCA, a comparison was conducted between the time taken to perform the "Standard LCA Case Study" using PristineLCA and openLCA. The results of the comparison revealed that PristineLCA had an average completion time of 22 minutes, while openLCA required 72 minutes on average, indicating a significant improvement in performance of 3.3 times for PristineLCA. Additionally, the total number of clicks needed to complete the LCA was also compared, with PristineLCA requiring a minimum of 70 clicks, while openLCA required 182 clicks, resulting in an improvement of 2.6 for PristineLCA. Furthermore, the user interface of PristineLCA was observed to be more efficient, as all tasks are completed within a single window, whereas openLCA required navigation through multiple windows. These findings were supported by the results of post-task questionnaires, where participants reported a lower level of difficulty when using PristineLCA compared to openLCA.

8 CONCLUSION, LIMITATIONS AND ACKNOWLEDGEMENTS

In conclusion, we leveraged an iterative design process, combined with user testing with real academics and students who used openLCA. This resulted in a fully interactive React front end that could be used to input information needed to perform the core functions that openLCA is intended for. Our user interface allowed basic LCA to be conducted with fewer clicks, and in approximately 1/3 of the time it took to conduct identical LCAs using openLCA. This increase in efficiency has the potential to improve the net amount of LCA research conducted on a whole, which directly addresses SDG 9, 12 and 13. Future work includes properly open sourcing the project, refining documentation to allow other developers to contribute to the project, and fully integrating the backend with openLCA to allow information entered using PristineLCA to be processed using the openLCA backend. Finally, we would like to thank our user testers, Prof. Jeremy Cooperstock and Julliete Regimbal for their invaluable contributions to this paper in terms of time and knowledge. Without their help, this paper would not be possible.

REFERENCES

[1] W. Kloppfer, “Life cycle assessment”, Environmental Science and Pollution Research, vol. 4, pp. 223-228, December 1997

[2] M.A Curran, Life Cycle Assessment Handbook: A Guide for Environmentally Sustainable Products, Wiley, 2012

[3] Marta Ormazabal, “Analysis and Comparison of Life Cycle Assessment and Carbon Footprint Software” in Proceedings of the Eighth International Conference on Management Science and Engineering Management: Focused on Intelligent System and Management Science, 2014.

[4] “GaBi Software”, 2023. [Online]. Available: https://sourceforge.net/software/product/GaBi-Software/.

[5] “SimaPro Licenses”, 2023. [Online]. Available: https://simapro.com/licences/#/business/compare

[6] J. Nielsen, "Thinking Aloud: The #1 Usability Tool," 15 January 2012. [Online]. Available: https://www.nngroup.com/articles/thinking-aloud-the-1-usability-tool/.

[7] J. Nielsen, "Estimating the number of subjects needed for a thinking aloud test," in International Journal of Human-Computer Studies, 1994.

[8] I. Young, Mental Models: Aligning Design Strategy with Human Behavior, Rosenfeld Media, 2008.

[9] B. Buxton, Sketching User Experiences, Elsevier, 2007.

[10] R. Sefelin, "Paper prototyping - what is it good for?: a comparison of paper- and computer-based low-fidelity prototyping," in CHI EA '03: CHI '03 Extended Abstracts on Human Factors in Computing Systems, 2003.