

Emissions Impacts of Demand Response in the Northwest US: Final Presentation

Lily Hahn, Chang Liu, James Stadler, and Daniel Lloveras

15 December 2021



Background

- > **The problem: the impact of demand response (DR) programs on greenhouse gas emissions is not well understood**
- > **Our solution: calculate DR emissions impacts in the Northwest US and create a dashboard that shows these impacts**
 - Emissions calculator should allow for alternate file inputs in case demand response projections or emissions factors change
 - Dashboard should be accessible to general public, but also include enough information to inform policymakers and data analysts

demand response: shifts or reduces electricity usage during periods of peak demand

Data

- > **Source: Northwest Power and Conservation Council**
 - Courtesy John Ollis and Tina Jayaweera
- > **Two main datasets:**
 - Projected hourly avoided emissions rates for 2021-2041
 - > *Baseline scenario, with potential to use multiple policy scenarios*
 - DR hours and potential for 2022-2041
 - > *Different DR plans, seasons, bins of products*
 - > *Can multiply by avoided emissions rates to get emissions reductions of DR programs*

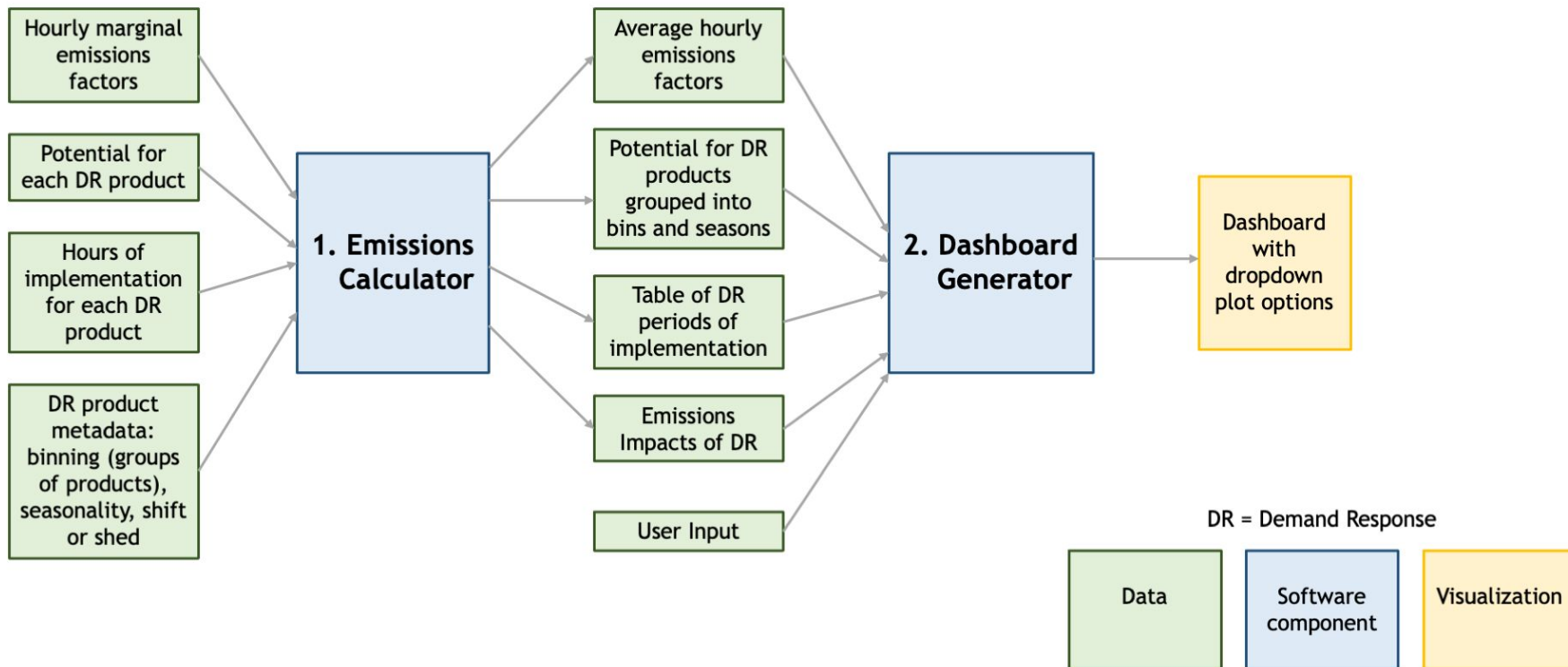
Use case: member of the general public

- > **Objective: view a dashboard to see how DR programs impact emissions in the Northwest US**
- > **Interactions:**
 - User: loads dashboard home page
 - System: shows home page with plots of total emissions reductions and emissions rates during different times of day
 - User: looks at default plots, reads interpretation, and uses dropdown menus to compare different seasons and products
 - System: displays new plots based on user input
 - User: selects more information page and uses dropdown menus
 - System: displays new plots and interpretation with more detail on DR impacts
 - User: downloads plots

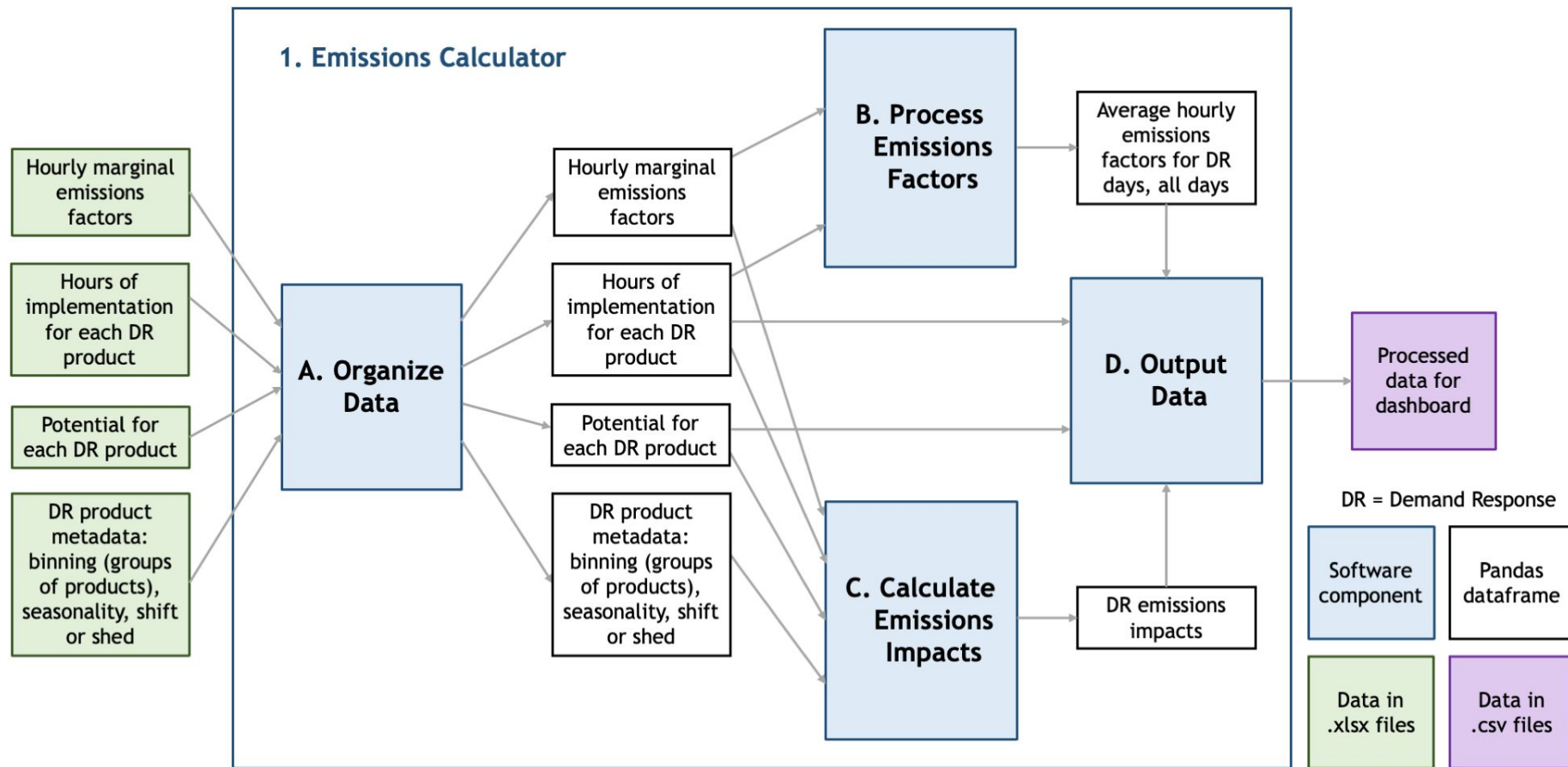
Use case: data analyst

- > **Objective: update the dashboard using new datasets**
- > **Interactions:**
 - User: clones GitHub repository
 - System: provides code for analyzing data and deploying dashboard
 - User: runs setup.py, pushes new datasets, and runs the emissions calculator
 - System: outputs processed data
 - User: edits the dashboard generator and deploys it to the web using Heroku
 - System: displays updated dashboard

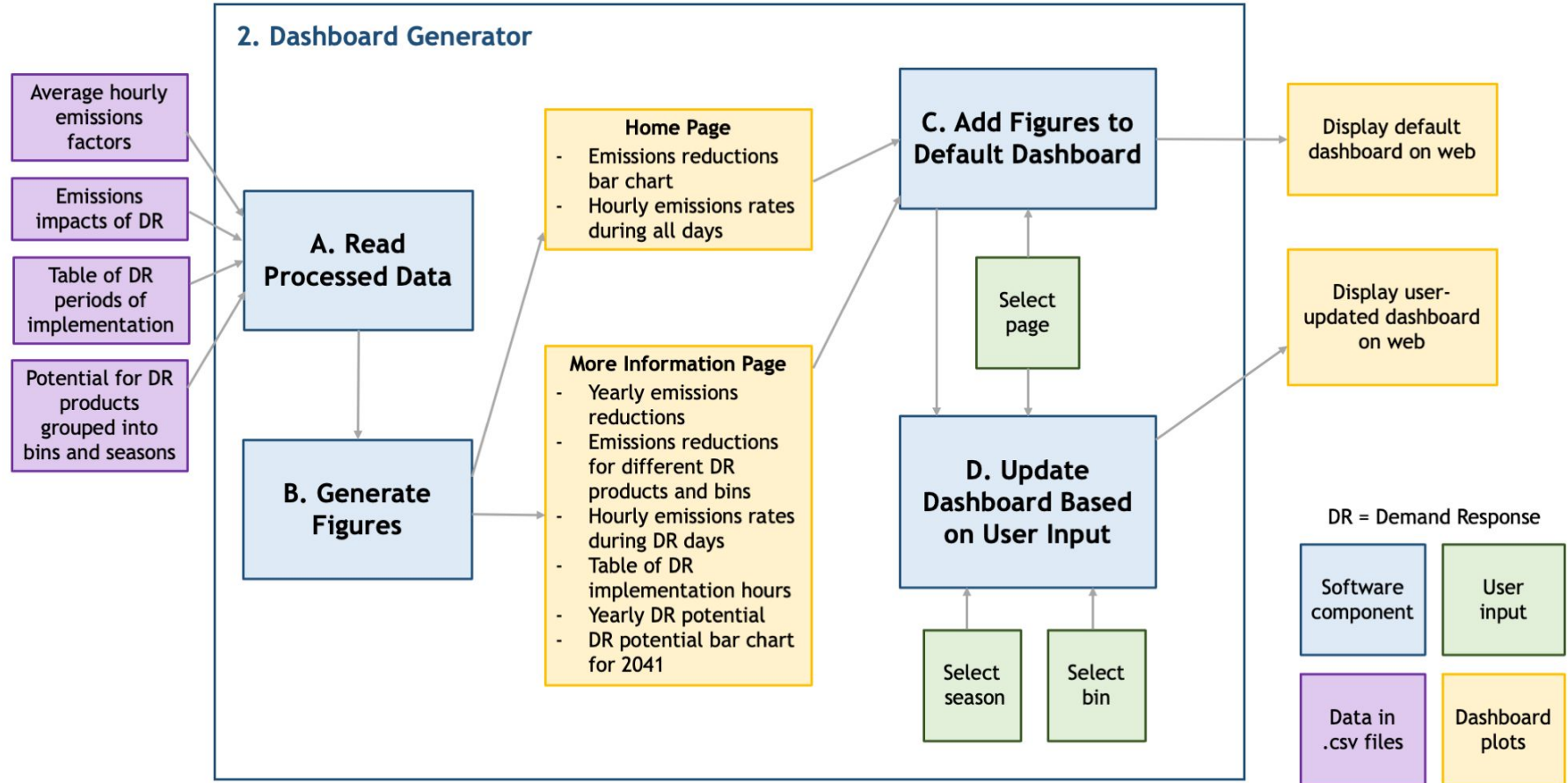
Overall design



Emissions calculator design



Dashboard generator design



Demo

- > Our dashboard:

<https://demand-response-impacts.herokuapp.com/home>
















- > Screenshots [here](#) for backup purposes



Project structure

> GitHub repository:

https://github.com/NW-Demand-Response-Emissions-Impacts/emissions_calculator/

	Ichahn Merge branch 'main' of https://github.com/NW-Demand-Response-Emissions-Impacts/emissions_calculator/ ... 3ea525f 38 minutes ago	
	assets	Update logo
	data	added subcomp d tests, pylint scores, test results for all but subcom...
	docs	Resolving merge conflict
	emissions_calculator	Resolving merge conflict
	examples	added user guide part 1
	.gitignore	Update .gitignore
	LICENSE	Initial commit
	Procfile	Testing dashboard with rest of project
	README.md	Update README.md
	app.py	Updated dashboard and pylint scores
	index.py	Updated index.py
	requirements.txt	added openpyxl
	runtime.txt	Testing dashboard with rest of project
	setup.py	added openpyxl

```
├── LICENSE
├── Procfile
├── README.md
├── app.py
├── assets/
├── docs/
│   ├── component_specification.pdf
│   ├── flow_charts/
│   ├── functional_specification.pdf
│   ├── pylint_scores/
│   ├── technology_review.pdf
│   └── test_results/
├── data/
│   ├── input_data/
│   │   ├── AvoidedEmissionsRates/
│   │   └── DRPotentialandHours/
│   └── processed_data/
│       ├── dr_hours/
│       ├── dr_potential/
│       ├── emissions_impacts/
│       └── emissions_rates/
├── emissions_calculator/
│   ├── phase1_emissions_calculator/
│   │   └── tests/
│   │       └── test_data/
│   └── phase2_dashboard_generator/
├── examples/
├── index.py
├── requirements.txt
├── runtime.txt
└── setup.py
```

Lessons learned and next steps

> Emissions calculator lessons:

- Tricky to keep things generalized
- To enable testing, use function arguments rather than importing parameters
- `**kwargs` are helpful in functions that generate exceptions

> Dashboard generator lessons:

- Very difficult to make plotting module generalizable with Plotly/Dash because of callbacks
- Learned a lot of HTML/CSS!

> Next steps:

- Incorporate additional policy scenarios
- Contextualize emissions impacts as a percentage of total emissions
- Calculate the maximum possible emissions reduction if DR hours are optimized to reduce emissions