Open Collaborative Data Engineering With Subject-Matter Experts Using Domain-Specific Languages

2025-09-16 Erlangen

Agenda

1. Introduction
2. Problem Identification
3. Design & Development of Jayvee
4. Evaluation
5. Summary (+ Future Work)

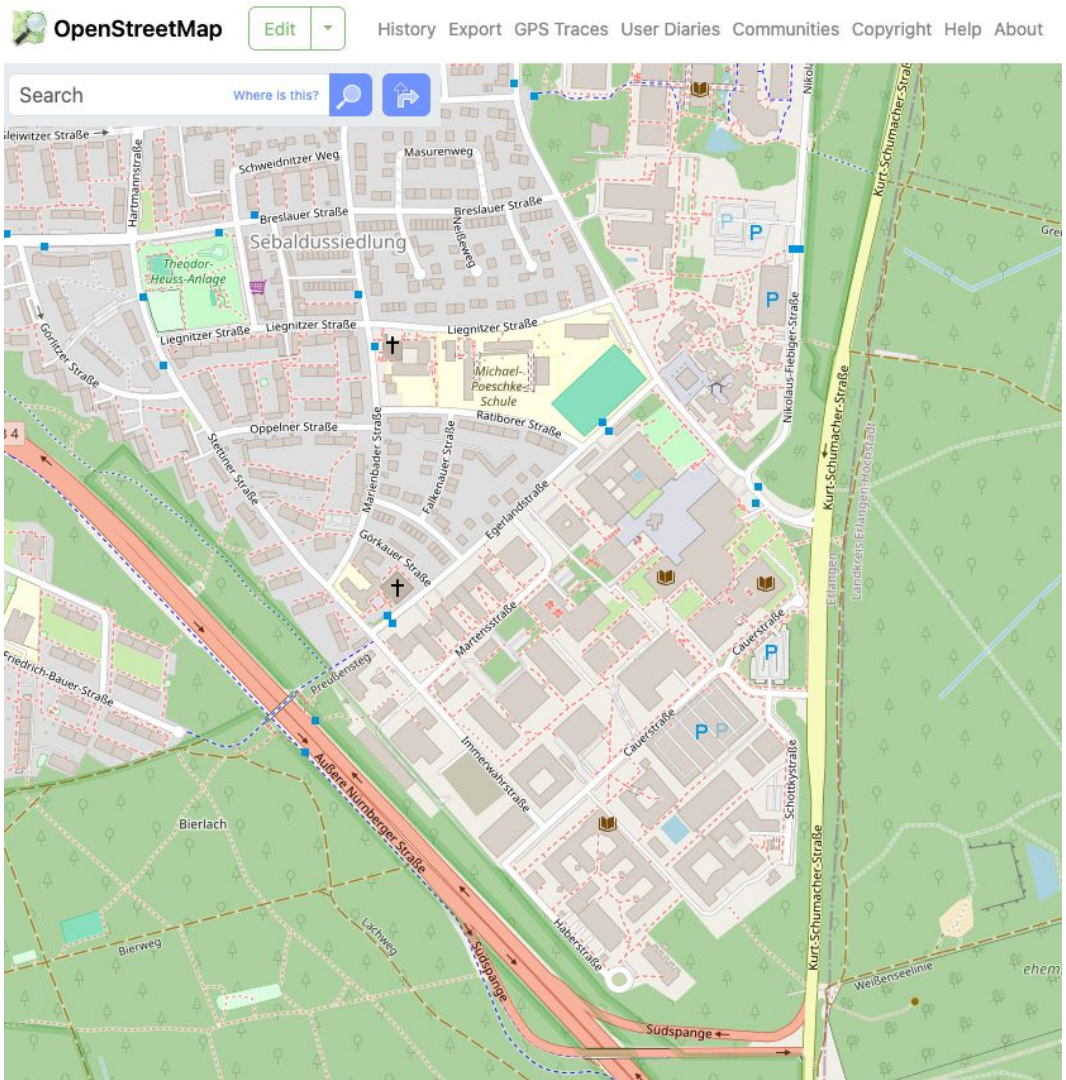
Introduction

Open Data

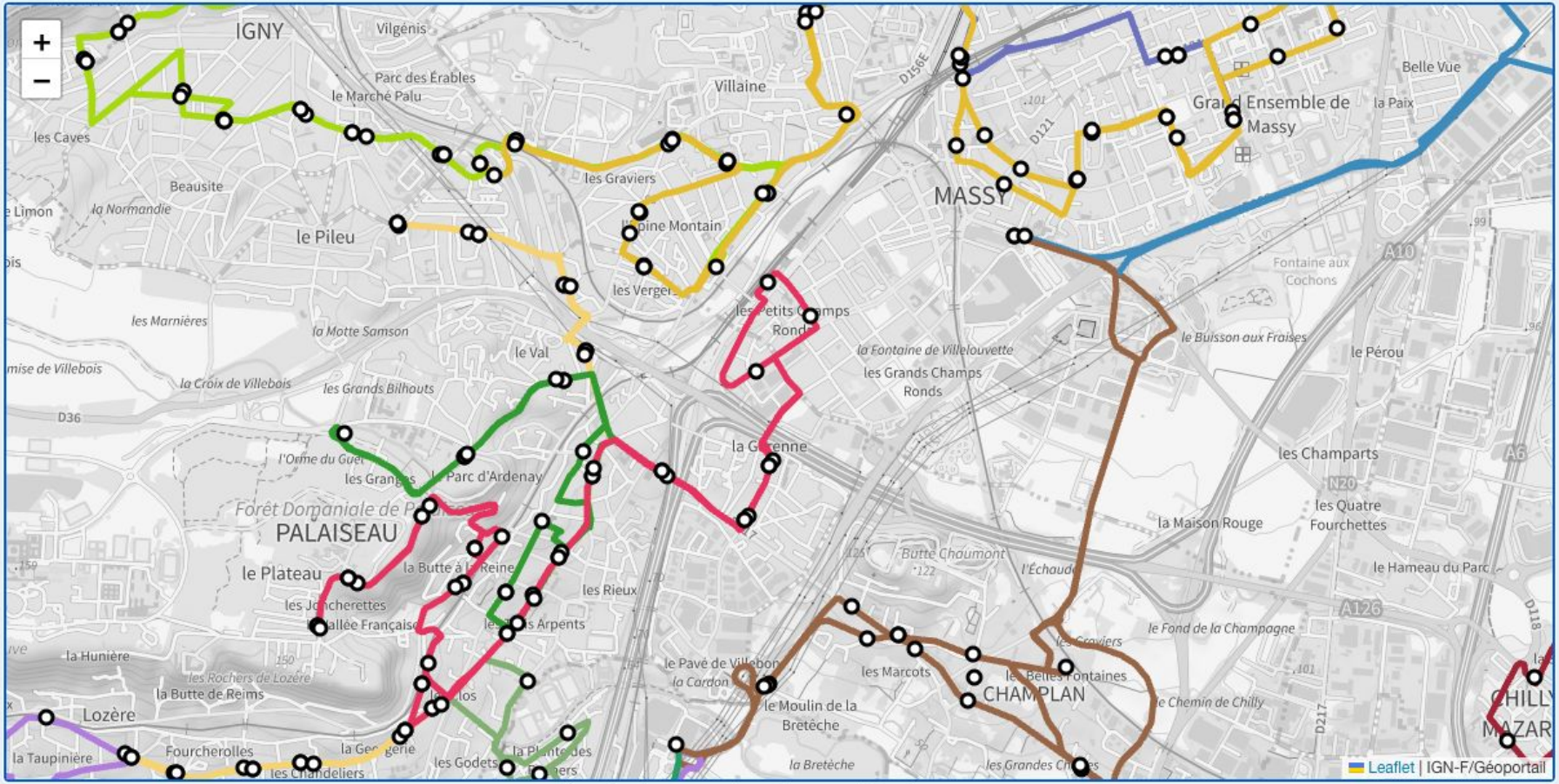
“”

Open data and content can be **freely used**, **modified**, and **shared** by **anyone** for **any purpose.**

The Open Definition, OKFN
https://opendefinition.org


OpenStreetMap
Edit
History
Export
GPS Traces
User Diaries
Communities
Copyright
Help
About
Search
Where is this?
Sebaldussiedlung
Theodor-Heuss-Anlage
Michael-Poeschke-Schule
Schweidnitzer Weg
Masurenweg
Breslauer Straße
Liegnitzer Straße
Görlitzer Straße
Oppelner Straße
Rattiborer Straße
Stettiner Straße
Marienbader Straße
Falkenauer Straße
Egerlandstraße
Görkauer Straße
Martensstraße
Cauerstraße
Immerwahrstraße
Schottkystraße
Haberstraße
Preußensteg
Friedrich-Bauer-Straße
Äußere Nürnberger Straße
Südspange
Bierlach
Bierweg
Lachweg
Kurt-Schumacher-Straße
Nikolaus-Fiebiger-Straße
Landkreis Erlangen-Höchstadt
Weißenseelinie

Stops and routes visualization of the GTFS file



Data Engineering

"" The process of making raw data usable for a specific project.

70%

of time spent on analytic projects according to IBM estimates[1]

Examples

- Manual cleaning in Excel
- Building automated data pipelines

[1] Terrizzano et al. (2015)

Collaborative Work

“” Participation of multiple contributors to achieve a shared goal.

Open collaboration[1]

- Egalitarian
- Meritocratic
- Self-organizing

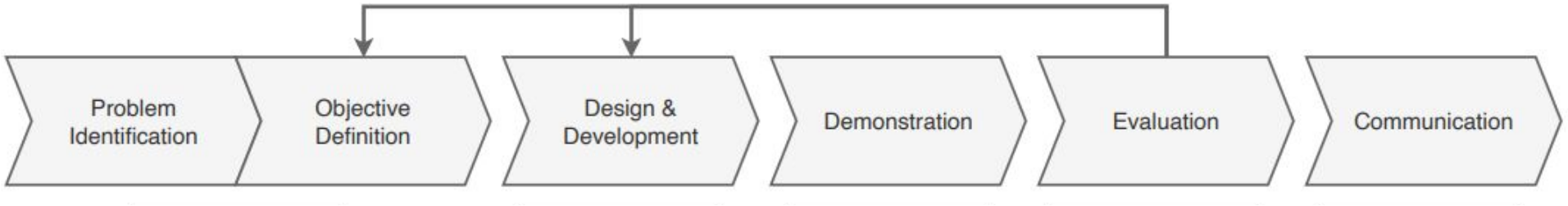
[1] Riehle et al. (2009)

Initial Research Question

RQ1: How can open collaborative data engineering with open data be enabled?

Design Science

Design and develop an innovative artifact



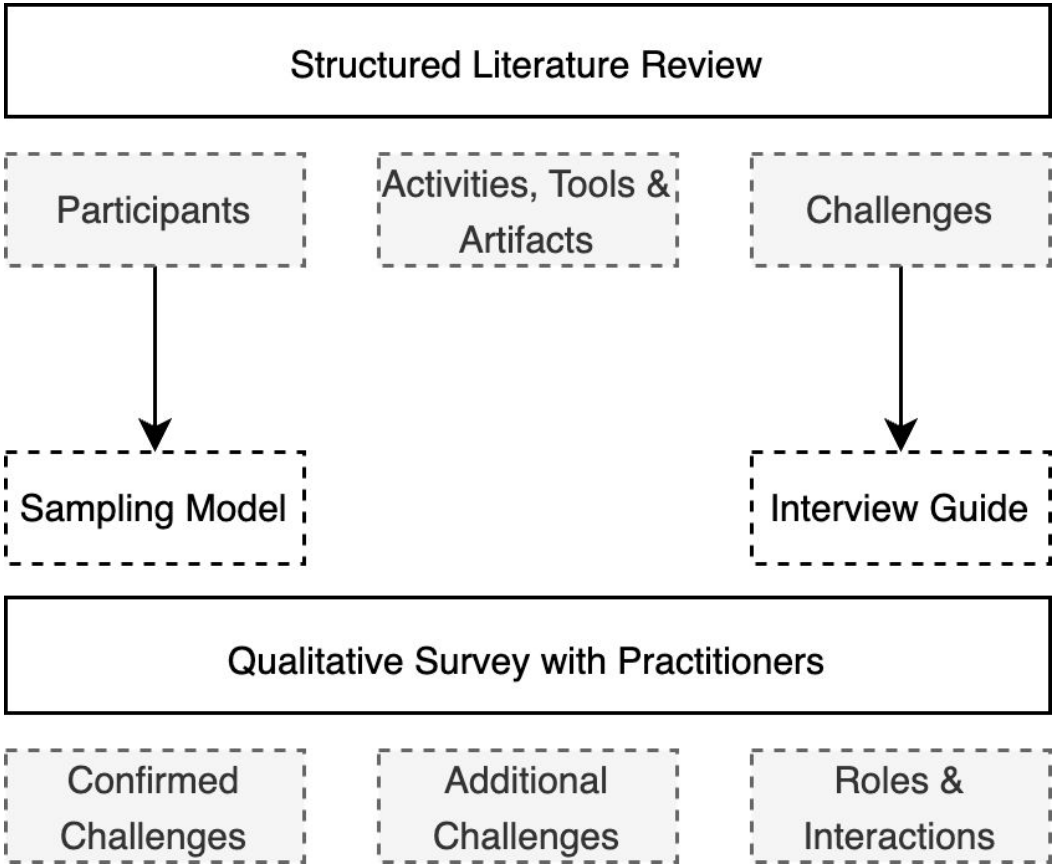
Peffers (2007)

Problem Identification
Objective Definition

[1] Heltweg, P., & Riehle, D. (2023). Challenges to Open Collaborative Data Engineering. In T. X. Bui (Ed.), Proceedings of the 56th Hawaii International Conference on System Sciences (pp. 679–688).

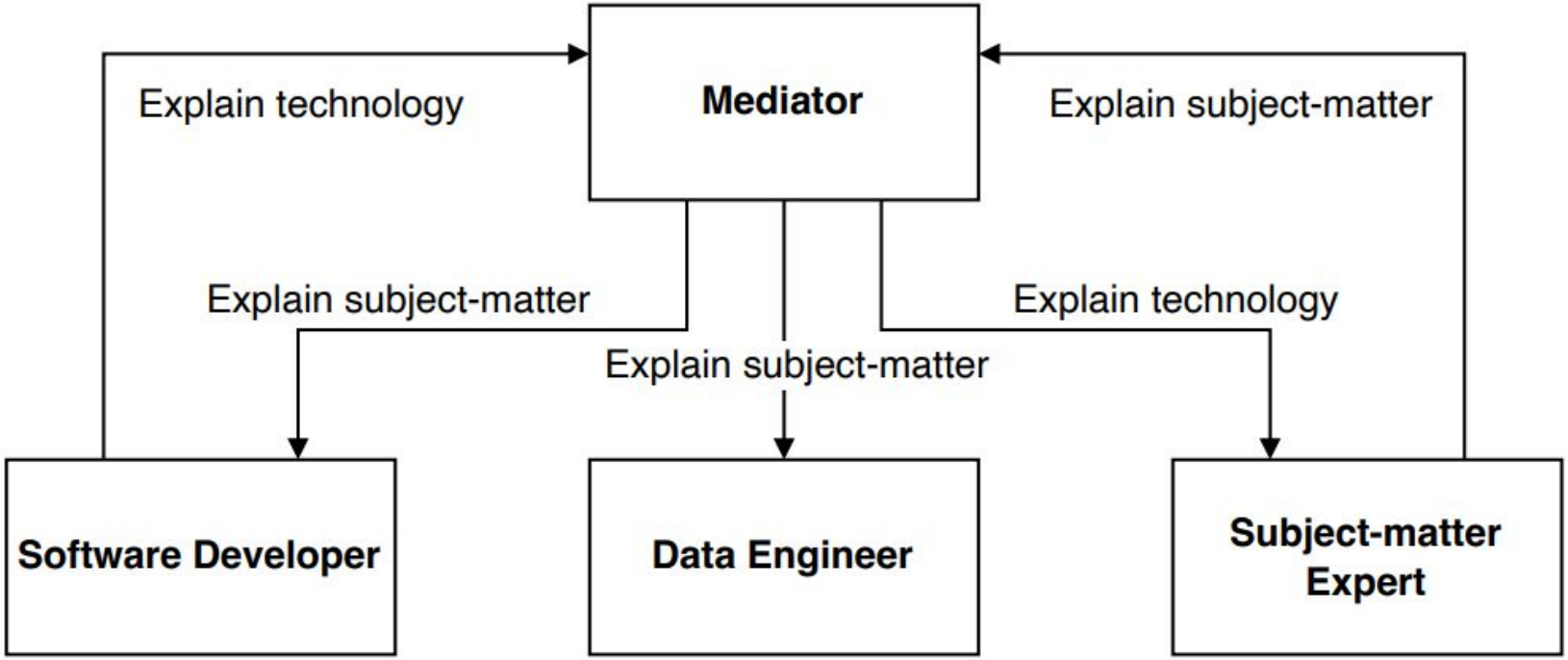
[2] Heltweg, P., & Riehle, D. (2023). A Systematic Analysis of Problems in Open Collaborative Data Engineering. ACM Transactions on Social Computing, 6(3–4), 1–30.

Research Design



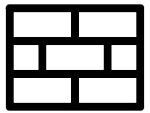
Kitchenham (2004)
Jansen (2010)

Roles in Collaborative Data Engineering

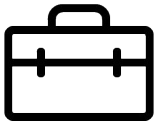


Selected Challenges

Missing semantic knowledge

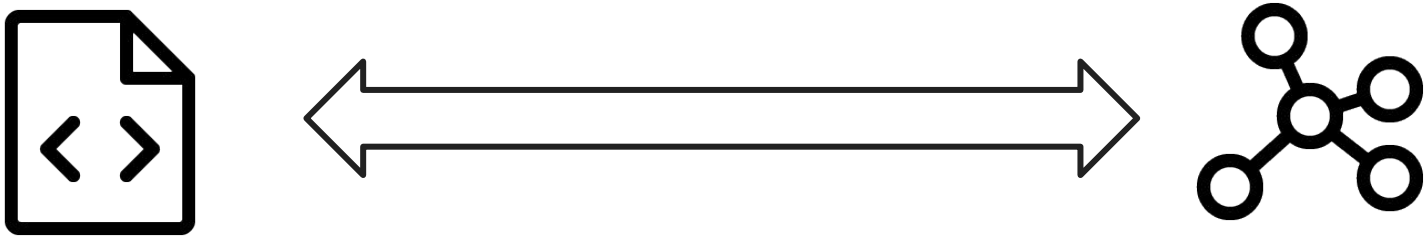


High technical barriers to participation



No standard tools or artifacts, inadequate tools

Tradeoff Programming vs Visual Tools



General Purpose Programming Languages (GPLs)

Visual Tools

Domain-specific languages (DSLs)

General Purpose Programming Languages (GPLs)

Visual Tools

Domain-specific Languages?

Additional Research Questions & Objectives

RQ2: DSL as a viable foundation for a collaboration in data engineering?

➡ Develop a DSL that is easy to use by subject-matter experts

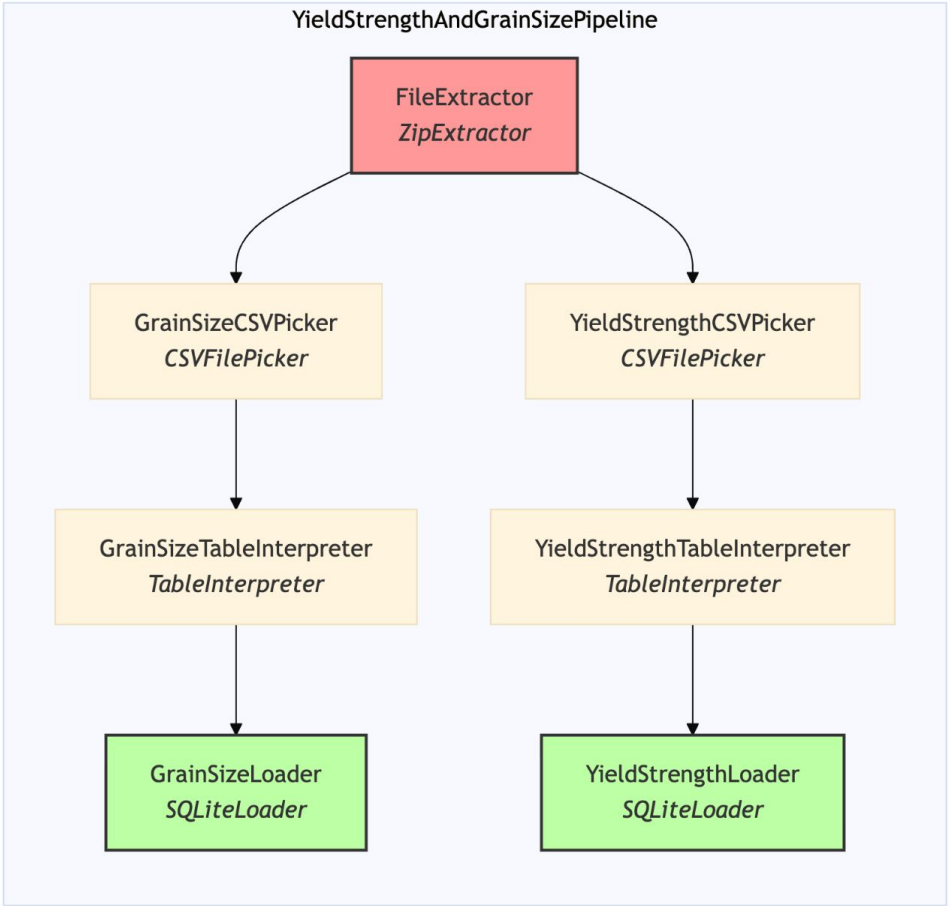
RQ3: Important considerations for a DSL used by subject-matter experts?

➡ Create a test bed for empirically developing insights

Design & Development

Jayvee, a domain-specific language for the creation and processing of automated data pipelines, https://github.com/jvalue/jayvee

Design


YieldStrengthAndGrainSizePipeline
FileExtractor
ZipExtractor
GrainSizeCSVPicker
CSVFilePicker
YieldStrengthCSVPicker
CSVFilePicker
GrainSizeTableInterpreter
TableInterpreter
YieldStrengthTableInterpreter
TableInterpreter
GrainSizeLoader
SQLiteLoader
YieldStrengthLoader
SQLiteLoader

Syntax

```
pipeline CarsPipeline {
    CarsCSVExtractor
        -> CarsTableInterpreter
        -> CarsLoader;

    block CarsCSVExtractor oftype CSVExtractor {
        url: "https://gist.githubusercontent.com/[...]/cars.csv";
    }

    block CarsTableInterpreter oftype TableInterpreter {
        header: true;
        columns: [
            "name" oftype text,
            "mpg" oftype decimal,
        ];
    }

    block CarsLoader oftype SQLiteLoader {
        table: "Cars";
        file: "./cars.sqlite";
    }
}
```

Interpreter

```
→  cars git:(main) ✗ jv cars.jv
Found 1 pipelines to execute: CarsPipeline
[CarsPipeline] Overview:
        Blocks (6 blocks with 2 pipes):
         -> CarsExtractor (HttpExtractor)
                 -> CarsTextFileInterpreter (TextFileInterpreter)
                         -> CarsCSVInterpreter (CSVInterpreter)
                                 -> NameHeaderWriter (CellWriter)
                                         -> CarsTableInterpreter (TableInterpreter)
                                                 -> CarsLoader (SQLiteLoader)

→  cars git:(main) ✗ ls
cars.jv     cars.sqlite
→  cars git:(main) ✗
```

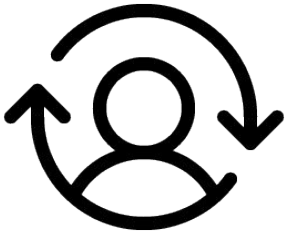
Evaluation

[3] Heltweg, P., Schwarz, G.-D., Riehle, D., & Quast, F. (2025). An empirical study on the effects of Jayvee, a domain-specific language for data engineering, on understanding data pipeline architectures. Software: Practice & Experience, 55(6), 1086–1105.

[4] Heltweg, P., Schwarz, G.-D., & Riehle, D. (2025). Can a domain-specific language improve program structure comprehension of data pipelines? A mixed-methods study. In arXiv [cs.PL].

[5] Heltweg, P., Riehle, D., & Schwarz, G.-D. (2025). Is spreadsheet syntax better than numeric indexing for cell selection? In arXiv [cs.PL].

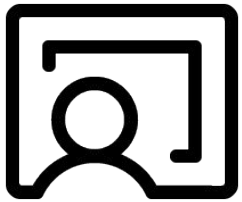
Language Evaluation With Human Participants



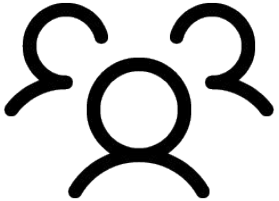
Test bed for continuous language evaluation

Methods
of
Advanced
Data
Engineering

The MADE Module



New Module
Since SS23



~450 Students

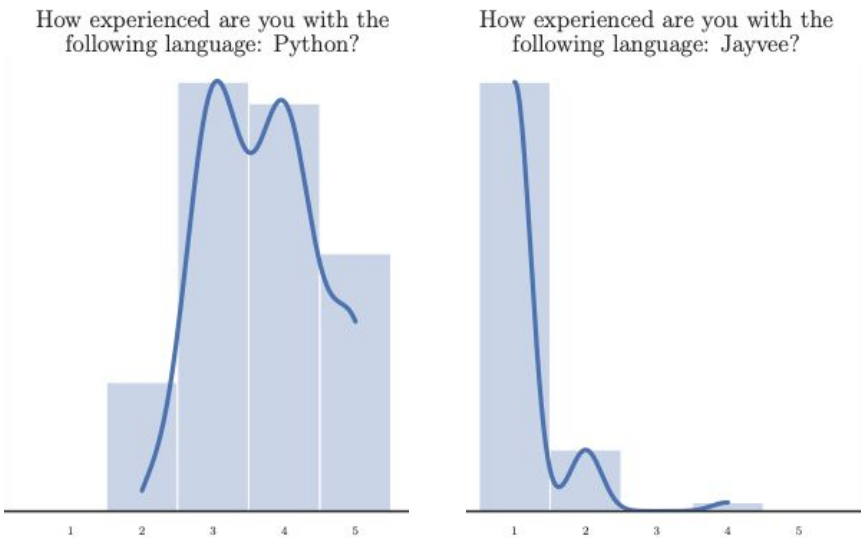
5 Exercises
/semester

Population Description

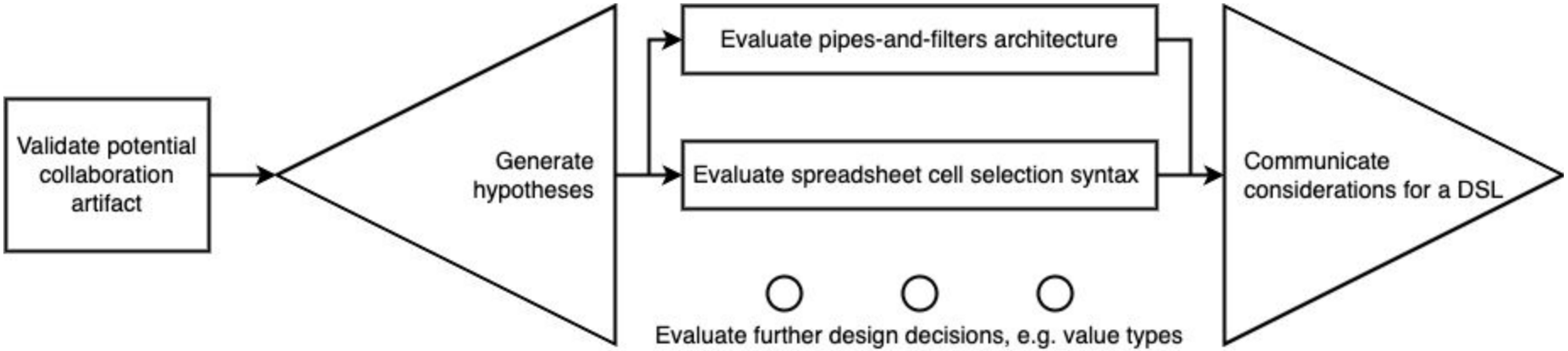
Master's level
Majority AI/Data science

No previous professional programming experience

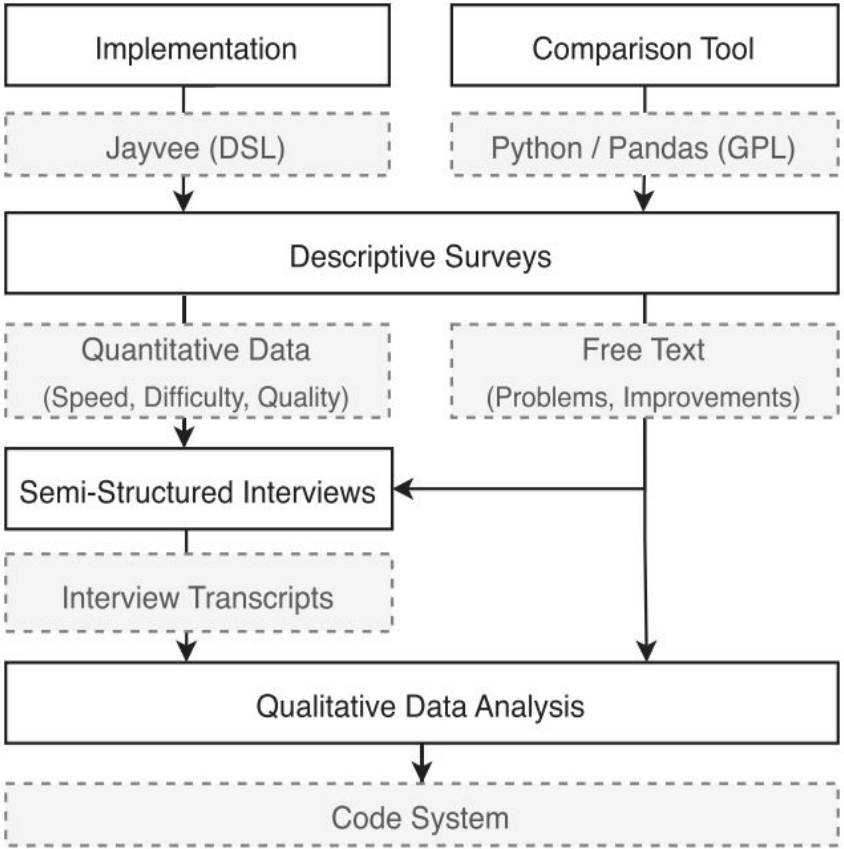
Know Python/Pandas scripting



Evaluation Overview

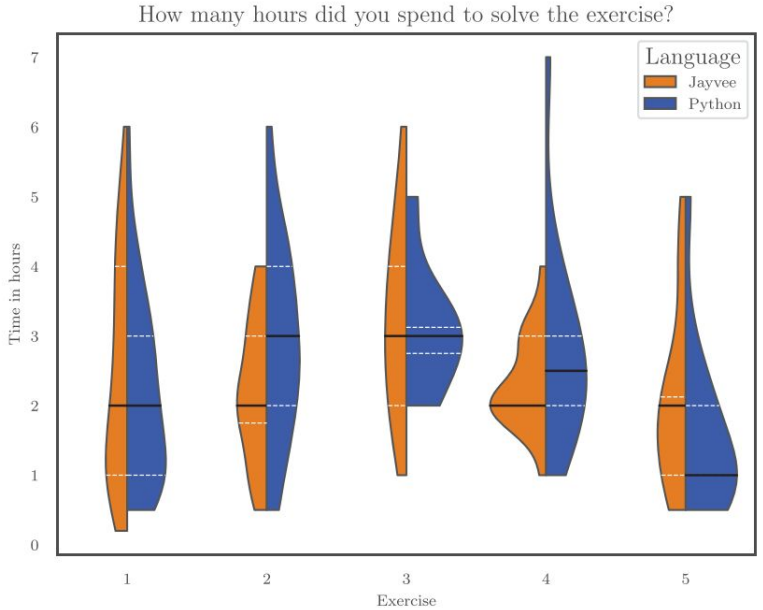


Validation – Research Design



Kitchenham & Pfleeger (2008)
Jansen (2010)
Braun & Clarke (2012)

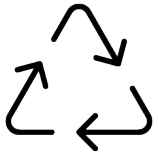
Task Performance



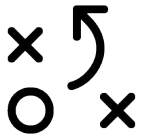
Exercise	n_{jv}, n_{py}	Mdn_{jv}, Mdn_{py}	U	p (less)
Ex1	45, 47	2.0, 2.0	1159.5	0.794
Ex2	28, 24	2.0, 3.0	243.0	**0.042***
Ex3	17, 8	3.0, 3.0	72.0	0.606
Ex4	18, 15	2.0, 2.5	113.0	0.202
Ex5	16, 16	2.0, 1.0	162.5	0.915

*$p \leq 0.05$.

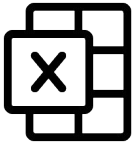
Effects of Using a DSL for Data Engineering



Easier to reuse



Enforced code structure helps understand

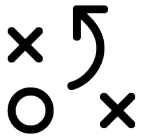


Experience outside of software engineering is relevant

Usability with ChatGPT (in 2023)

Effects of Using a DSL for Data Engineering

Easier reuse

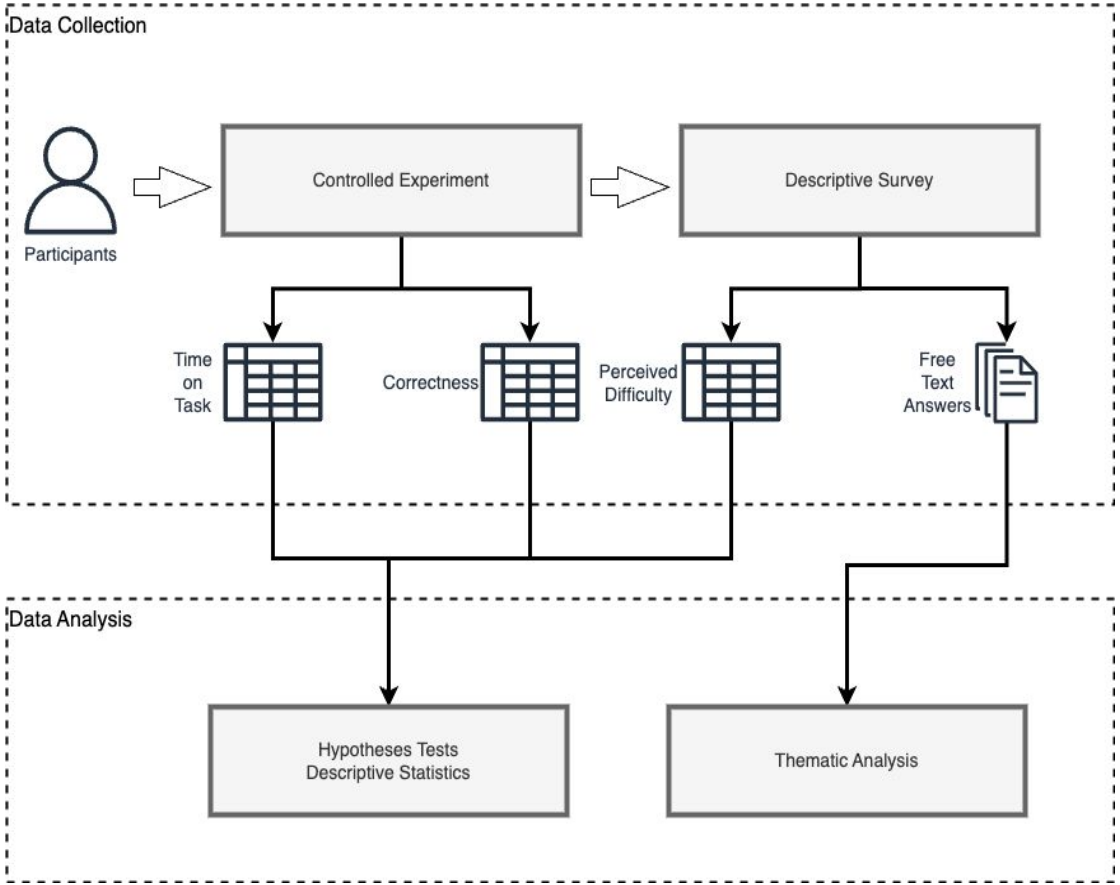


Enforced code structure helps understand

Relevant experience outside of software engineering

Usability with ChatGPT (in 2023)

Program Understanding - Research Design



Ko et al. (2015)
Kitchenham & Pfleeger (2008)
Braun & Clarke (2012)

Tasks and Tooling

Pipeline Code

```
import pandas as pd
from sqlalchemy import create_engine

fileName = 'https://geo.sv.rostock.de/download/opendata/rettungswachen/rettungswachen.csv'
data = pd.read_csv(fileName, delimiter=',', decimal='.')

data = data[
    [
        'uuid',
        'latitude',
        'longitude',
        'bezeichnung',
        'traeger_bezeichnung',
        'traeger_art',
        'website',
    ]
]

data = data.astype({
    'uuid': str,
    'latitude': float,
    'longitude': float,
    'bezeichnung': str,
    'traeger_bezeichnung': str,
    'traeger_art': str,
    'website': str,
})

data = data[data['latitude'].apply(lambda input: input >= -90 and input <= 90)]
data = data[data['longitude'].apply(lambda input: input >= -90 and input <= 90)]

data['publiclyFunded'] = data['traeger_art'].map(
    lambda input: input == 'öffentlich'
)

sinkFile = 'rescuestations.db'
tableName = 'rescuestations'
engine = create_engine(f'sqlite:///{sinkFile}')

data.to_sql(tableName, engine, if_exists='replace', index=False)

engine.dispose()
```

Pipeline Steps

Submit Solution

Steps in Data Pipeline

- Download a file from the internet
- Interpret a file as CSV with the delimiter ','

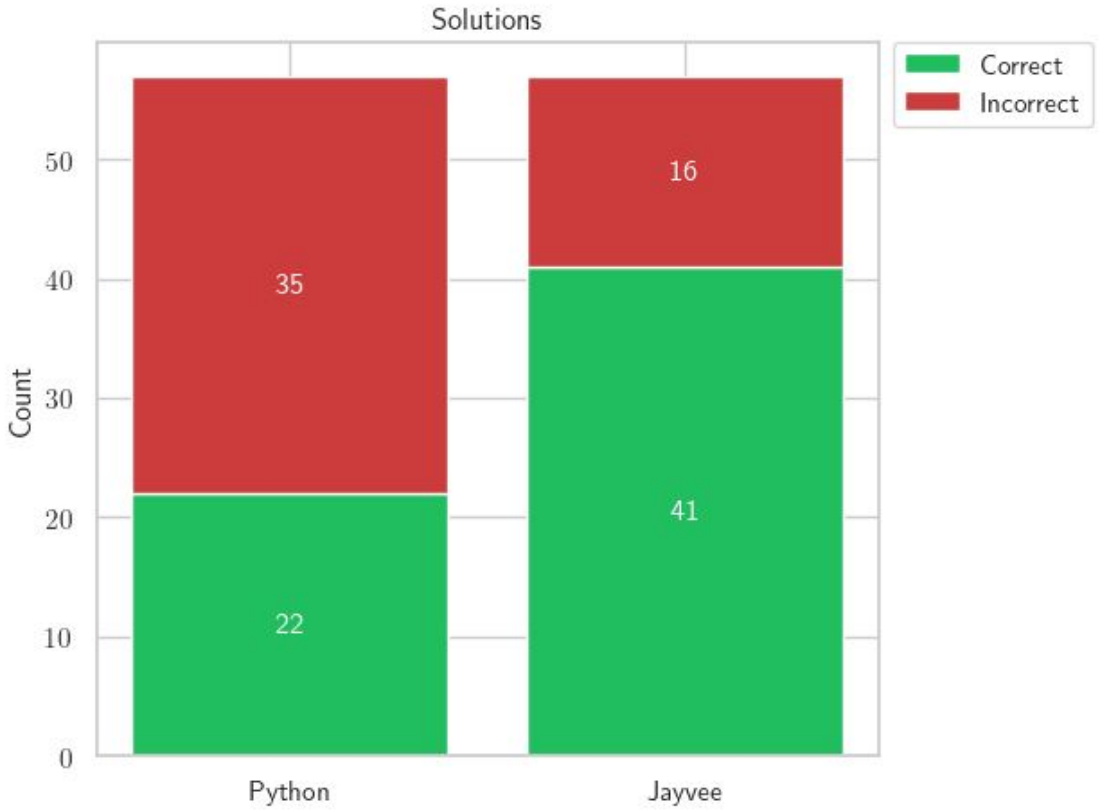
Unused Steps

- Translate column names to English
- Save the data to a PostgreSQL database
- Add a new column based on existing data
- Calculate the average of a column
- Validate that latitude and longitude are valid geographic coordinates
- Save the data to a SQLite database
- Download a ZIP file from the internet
- Transform all temperatures to Fahrenheit
- Delete the last ten rows of data

Crossover Design

		Period	
		Task 1	**Task 2**
Sequence	**AB**	Jayvee	Python/Pandas
	BA	Python/Pandas	Jayvee

Fewer Mistakes, but Not Faster or Easier



Improved Correctness

		Python/Pandas	
		Incorrect	Correct
Jayvee	Incorrect	11	**5**
	Correct	**24**	17

n	χ^2_1	p-val	OR
57	11.17	$< .001^*$	4.8

$^* p \leq .05$

Effects of a DSL on Pipeline Understanding

- ⊕ Pipeline overview
- ⊕ Blocks considered intuitive
- ⊕ Value types in general
- ⊖ Low level language elements
- ⊖ Advanced programming concepts
- ⊖ Value type implementation


Similar pipelines increase learning effects

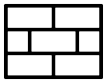

Experience from tools outside of programming feels relevant

Summary

Open Collaborative Data Engineering

RQ 1: How can open collaborative data engineering with open data be enabled?

Collaboration requires technical and subject-matter knowledge



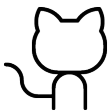
Subject-matter experts are essential collaborators but struggle with technical challenges and inadequate tools

Enable SMEs with easy to use, purpose-built tools

DSLs As Foundation for Collaboration Artifacts

RQ 2: DSL as a viable foundation for a collaboration in data engineering?

A DSL can be learned fast and shows improvements for data engineering by SMEs



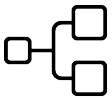
Allows reuse of existing collaboration tools and workflows from software engineering

A text-based DSL is a viable middle ground for data engineering with SMEs

Learnings for DSL Design for Data Engineering

RQ 3: Important considerations for a DSL used by subject-matter experts?

Expect improved correctness, not increased speed



Consistent pipes-and-filters structure helps understand, a syntactically separated pipeline overview is intuitive

Value types promising, but improve our implementation

Appropriate level of abstraction to users is a core challenge

Future Work

1. Develop the DSL using the existing setup for empirical evaluation
2. Controlled experiments with data practitioners
3. Investigate open collaborative workflows for data engineering

Thank you

Encore!

Cell Selection Syntax - Study Design

- Compare *spreadsheet syntax (A5:B10)* to *numerical syntax (iloc, [0:5, 10:15])* for cell selection in 2D data
- Crossover design
- No qualitative part - no explanations for the effects ground in data

Cell Selection Syntax - Tooling

Task: Write Cell Selection Code

1. Look at data

Please look at the data and notice the subset of cells that are highlighted in blue.

Spain	1	1	2	57	24	13	57619	37.417	-6.005
Denma	1	2	2	51	26	20	38052	55.703	12.572
Hungar	1	1	2	40	30	9	65014	47.503	19.098
Finland	0	2	3	54	29	4	59960	59.973	30.221
Germar	2	2	2	79	20	14	75000	48.219	11.625
France	3	3	3	70	24	5	54231	44.438	26.152
Netherl	0	2	2	36	31	9	65014	47.503	19.098
Russia	1	4	2	69	18	19	38052	55.703	12.572
Sweder	1	2	1	53	22	8	51824	55.826	-4.252
Türkiye	0	2	4	61	27	12	68700	40.430	49.920

2. Complete Jayvee code to select cells

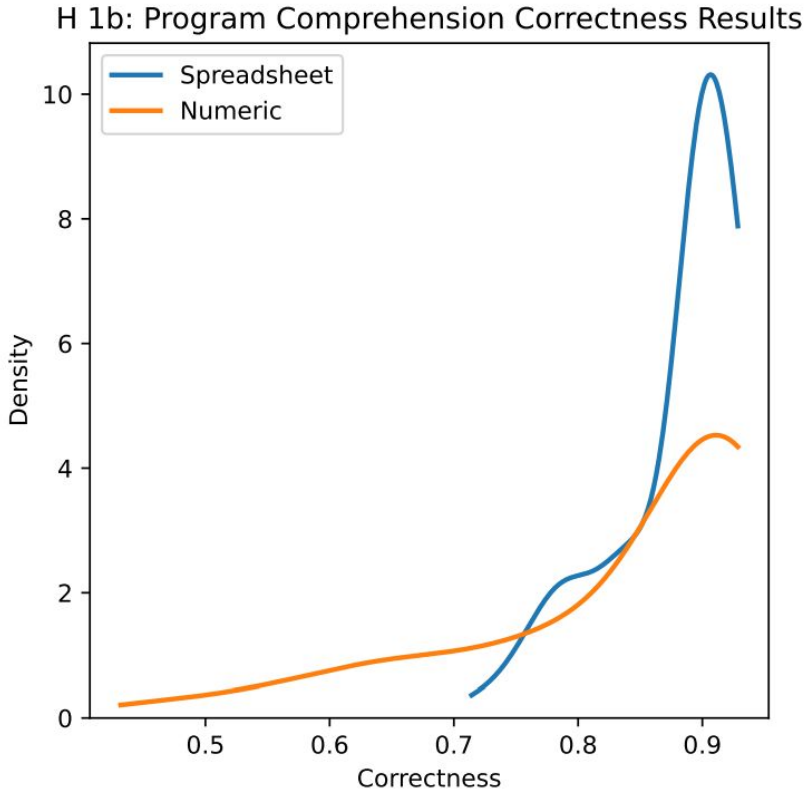
Please complete the code to select the subset of data that is highlighted on the left.

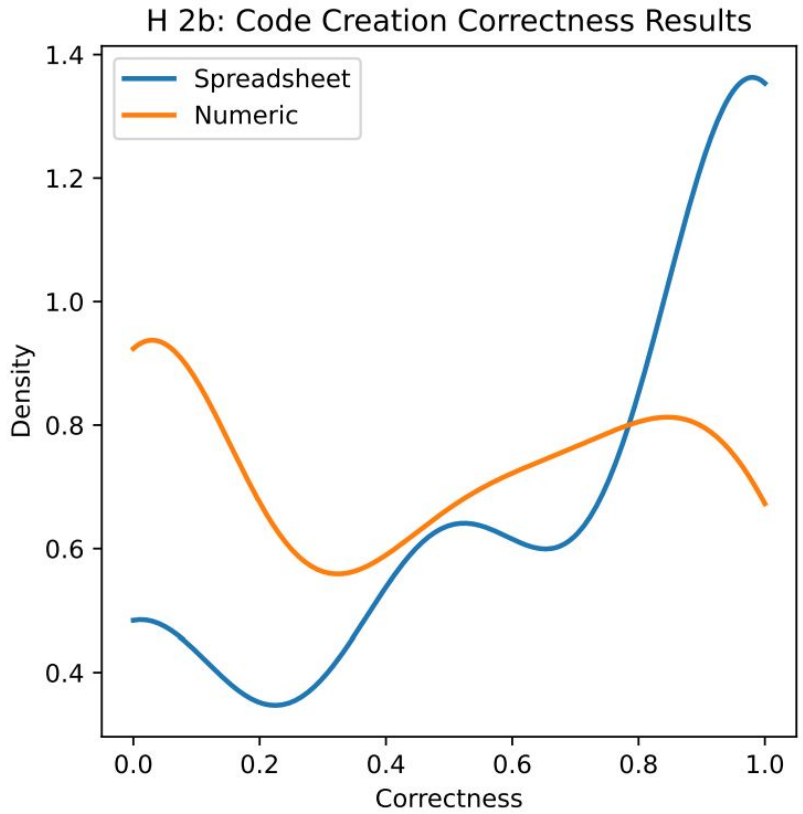
```
// Jayvee code
// Other blocks and pipeline definition...

block DataSelector oftype CellRangeSelector {
  select: range [          ];
}
```

3. Submit Solution

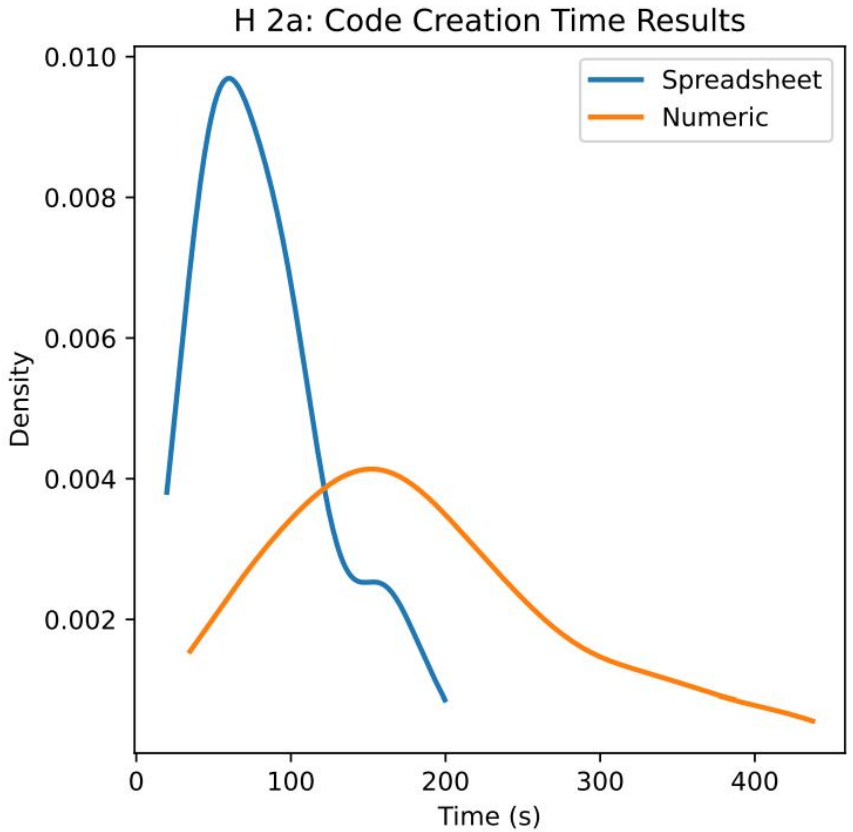
Enter a solution first

Cell Selection Syntax - Improved Correctness





Cell Selection Syntax - Faster Code Creation



Thank you again

References

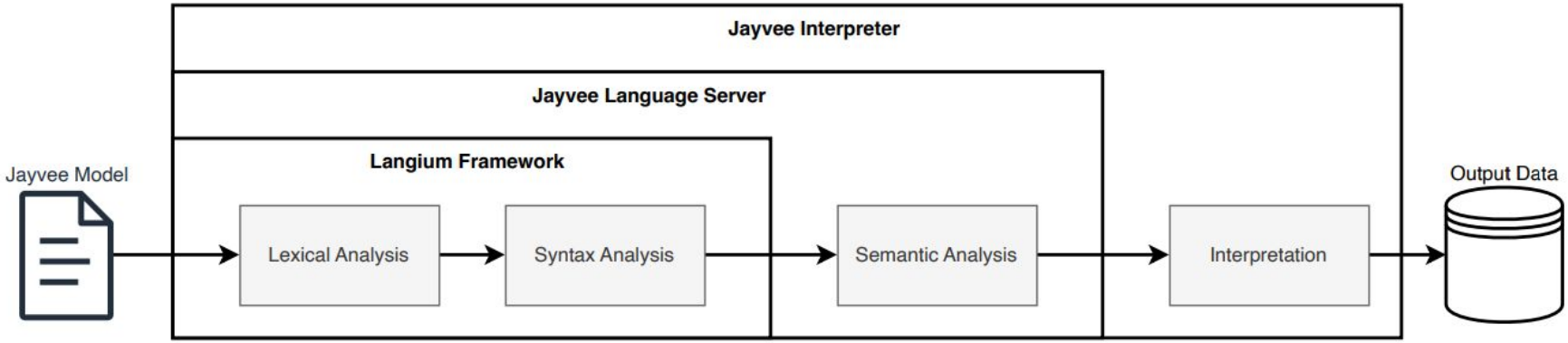
1. Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. Journal of Management Information Systems, 24(3), 45–77.
2. Terrizzano, I. G., Schwarz, P. M., Roth, M., & Colino, J. E. (2015). Data Wrangling: The Challenging Journey from the Wild to the Lake. CIDR. 7th Biennial Conference on Innovative Data Systems Research, Asilomar, California, USA.
3. Riehle, D., Ellenberger, J., Menahem, T., Mikhailovski, B., Natchetoi, Y., Naveh, B., & Odenwald, T. (2009). Open Collaboration within Corporations Using Software Forges. IEEE Software, 26(2), 52–58.
4. Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. Journal of Management Information Systems, 24(3), 45–77.
5. Kitchenham, B. (2004). Procedures for performing systematic reviews. Keele, UK, Keele University, 33(2004), 1–26.
6. Jansen, H. (2010). The logic of qualitative survey research and its position in the field of social research methods. Forum Qualitative Sozialforschung / Forum: Qualitative Social Research, 11(2).
7. Kitchenham, B. A., & Pfleeger, S. L. (2008). Personal Opinion Surveys. In F. Shull, J. Singer, & D. I. K. Sjøberg (Eds.), Guide to Advanced Empirical Software Engineering (pp. 63–92). Springer London.
8. Braun, V., & Clarke, V. (2012). Thematic analysis. In APA handbook of research methods in psychology, Vol 2: Research designs: Quantitative, qualitative, neuropsychological, and biological (pp. 57–71). American Psychological Association.
9. Ko, A. J., LaToza, T. D., & Burnett, M. M. (2015). A practical guide to controlled experiments of software engineering tools with human participants. Empirical Software Engineering, 20(1), 110–141.

Backup

Sampling Model

Job Role	Employer Type	Project Type	Data Domain	Professional	Open Data
Data Engineer	Academic	Civic Activism	Government	Yes	Yes
Business Development	Business	Data Product	Regulated	No	No
Software Engineer	Nonprofit	Scientific	Scientific		Both

Implementation Choices



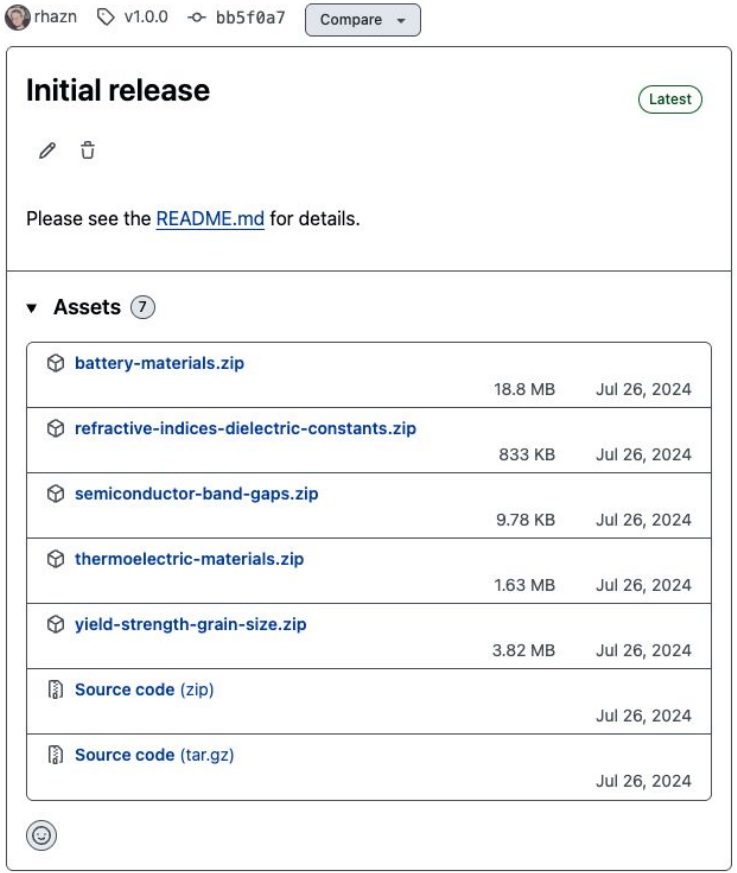
Demonstration

Use in Industry

software campus

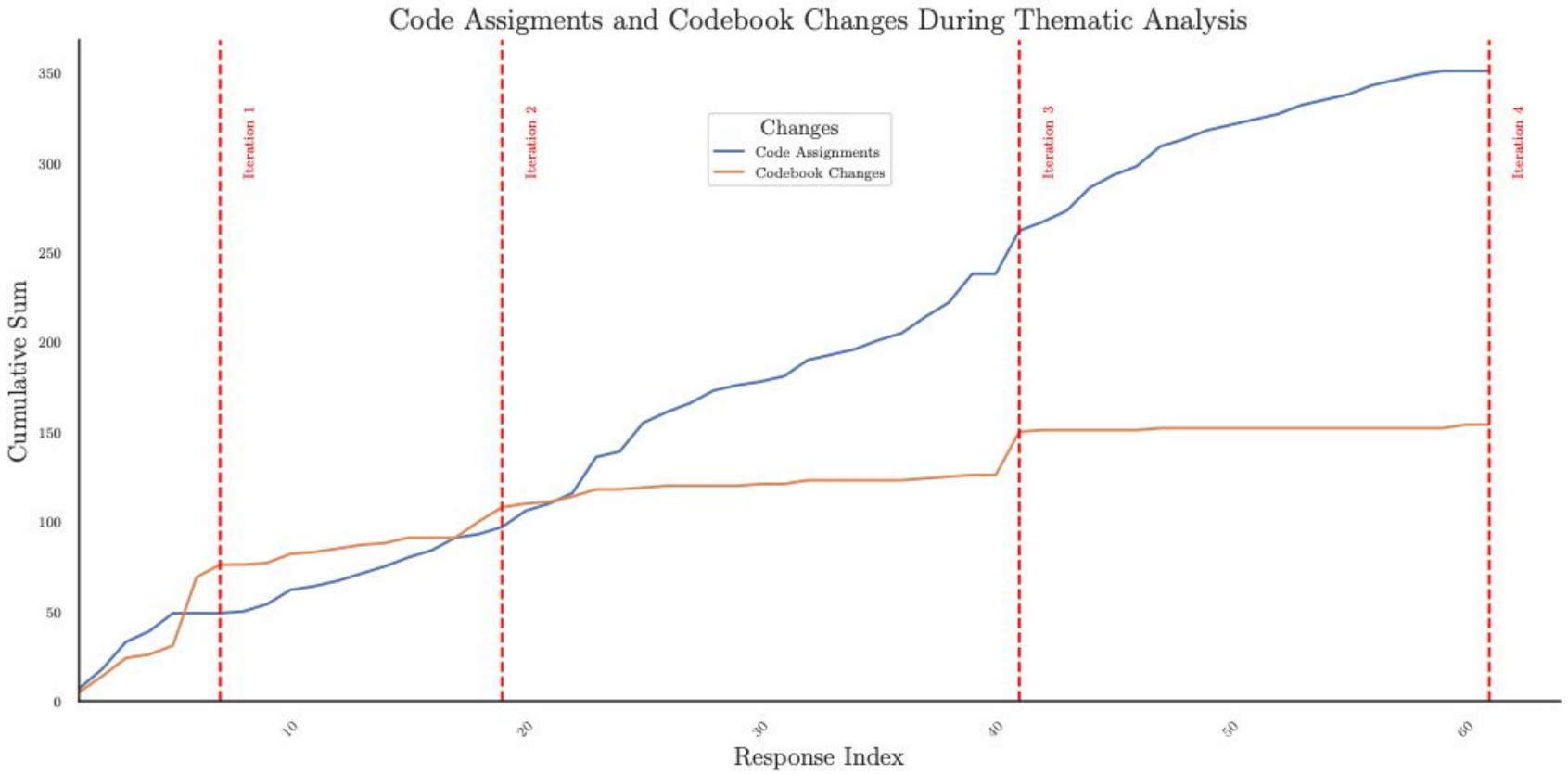
Springer Materials

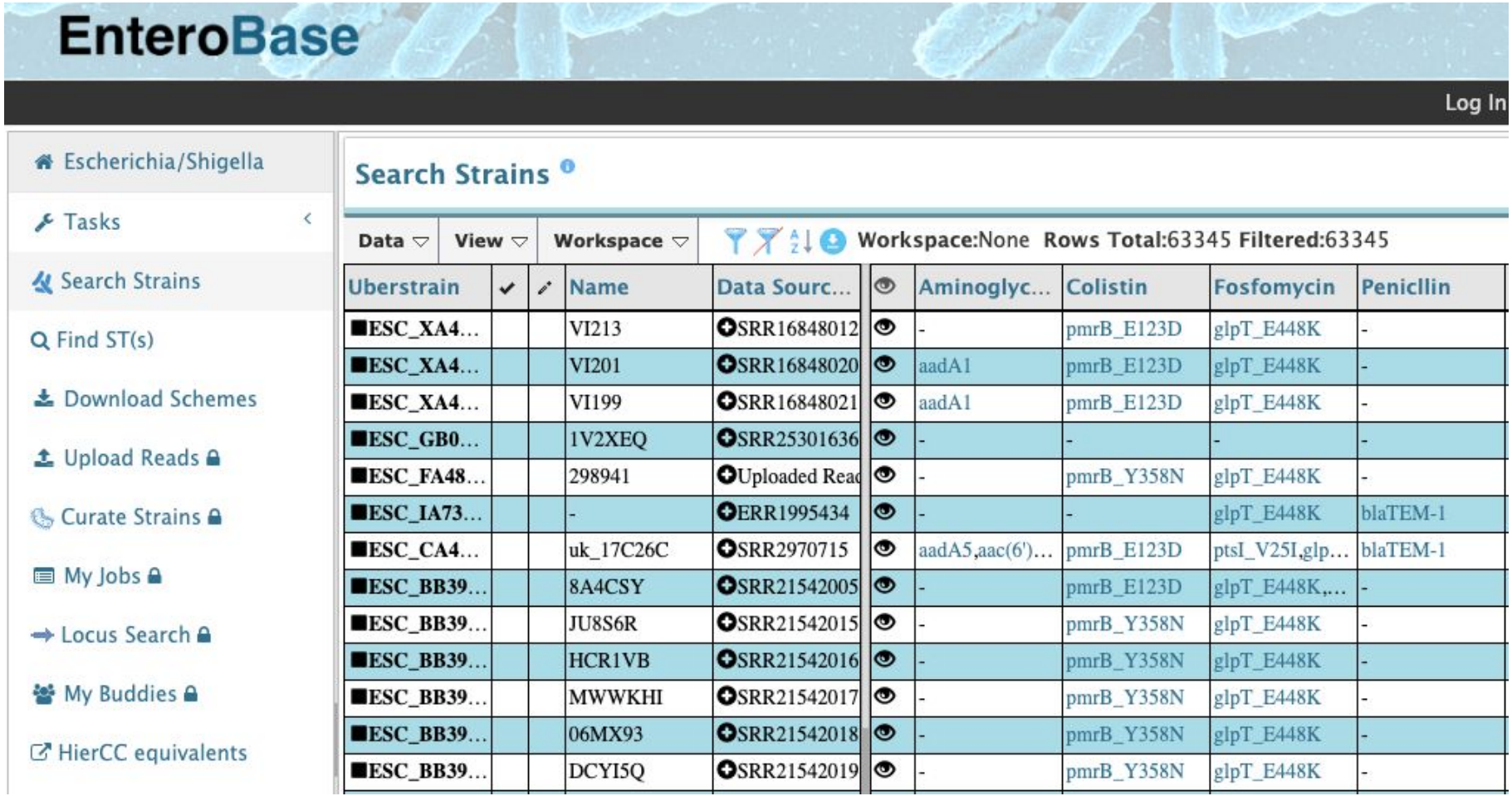
holtzbrinck Publishing Group




01.153
Department Informatik
Rechnerarbeitsraum
MADE experiment
here
Please wait outside, we'll invite you in
Sprechzeiten der CIP-Betreu
Mittwoch /
Freitag /
CIP-Admin Informatik
Kameraaufzeichnung
Dieser Bereich wi
Kamera mit Aufze
überwacht

Thematic Analysis




EnteroBase
Log In
Escherichia/Shigella
Tasks
Search Strains
Find ST(s)
Download Schemes
Upload Reads
Curate Strains
My Jobs
Locus Search
My Buddies
HierCC equivalents
Search Strains
Data
View
Workspace
Workspace:None Rows Total:63345 Filtered:63345
Uberstrain | Name | Data Sourc... | Aminoglyc... | Colistin | Fosfomycin | Penicllin
ESC_XA4... | VI213 | SRR16848012 | - | pmrB_E123D | glpT_E448K | -
ESC_XA4... | VI201 | SRR16848020 | aadA1 | pmrB_E123D | glpT_E448K | -
ESC_XA4... | VI199 | SRR16848021 | aadA1 | pmrB_E123D | glpT_E448K | -
ESC_GB0... | 1V2XEQ | SRR25301636 | - | - | - | -
ESC_FA48... | 298941 | Uploaded Reac | - | pmrB_Y358N | glpT_E448K | -
ESC_IA73... | - | ERR1995434 | - | - | glpT_E448K | blaTEM-1
ESC_CA4... | uk_17C26C | SRR2970715 | aadA5,aac(6')... | pmrB_E123D | ptsI_V25I,glp... | blaTEM-1
ESC_BB39... | 8A4CSY | SRR21542005 | - | pmrB_E123D | glpT_E448K,... | -
ESC_BB39... | JU8S6R | SRR21542015 | - | pmrB_Y358N | glpT_E448K | -
ESC_BB39... | HCR1VB | SRR21542016 | - | pmrB_Y358N | glpT_E448K | -
ESC_BB39... | MWWKHI | SRR21542017 | - | pmrB_Y358N | glpT_E448K | -
ESC_BB39... | 06MX93 | SRR21542018 | - | pmrB_Y358N | glpT_E448K | -
ESC_BB39... | DCYI5Q | SRR21542019 | - | pmrB_Y358N | glpT_E448K | -

Thesis - Figures/Tables

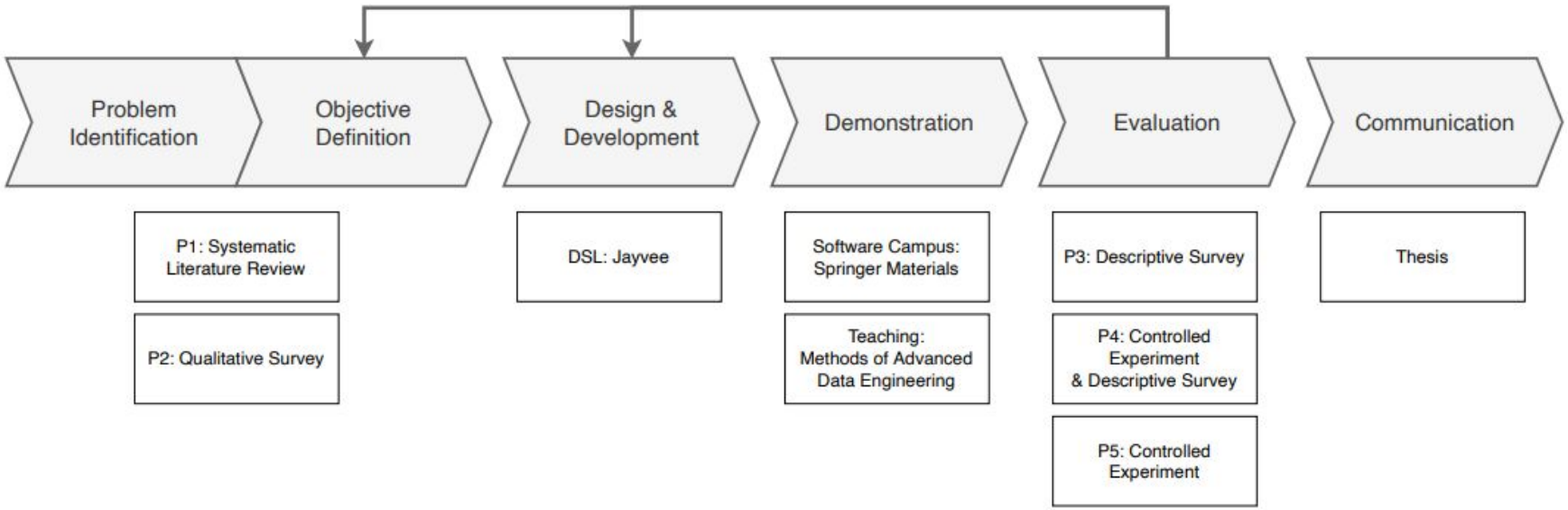


Figure 3.1: Activities of the design science process according to Peffers et al. [47] with associated projects and artifacts described in this thesis.

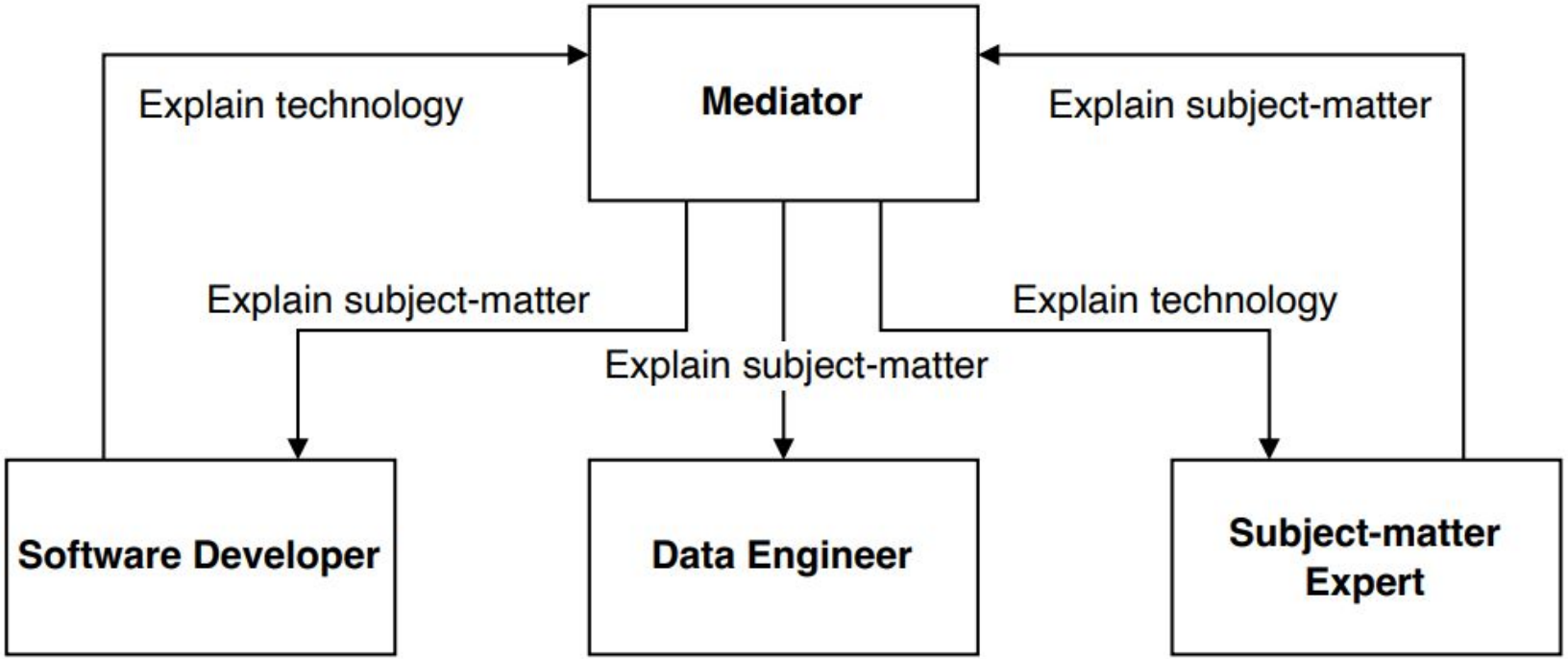
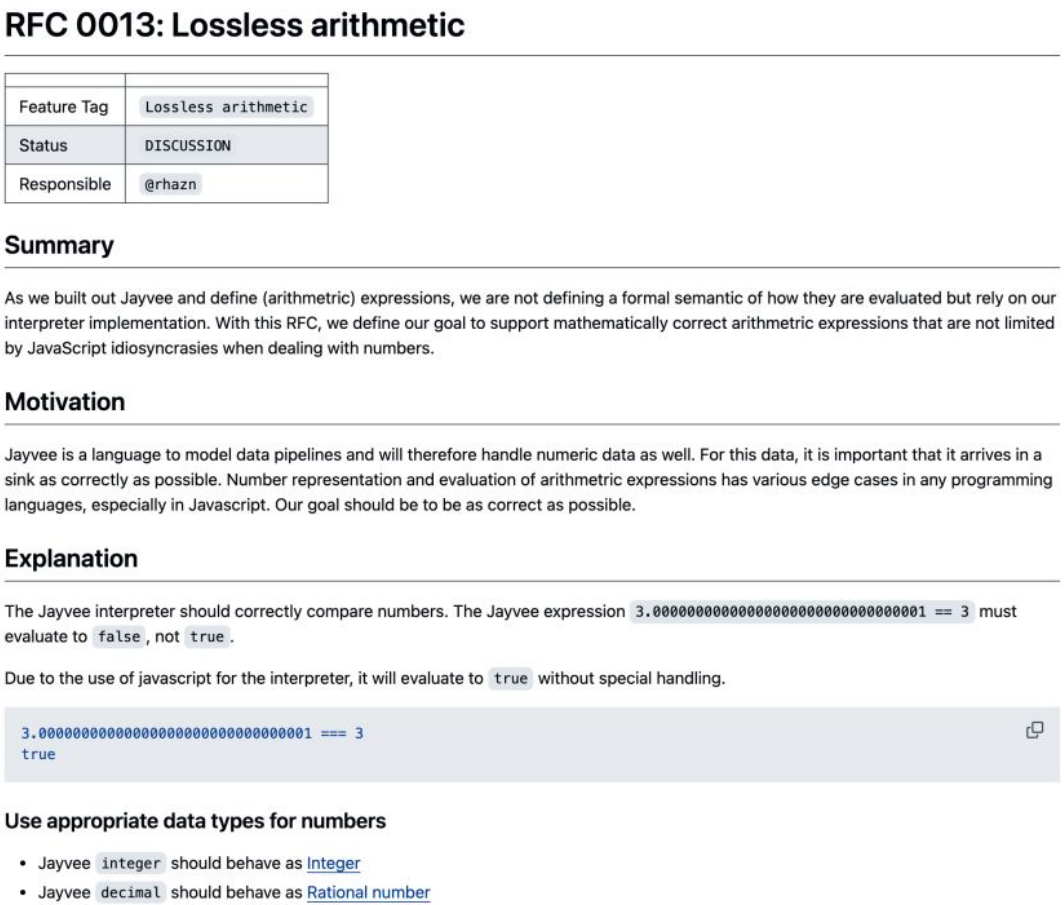


Figure 4.1: Roles and social interactions in a project group during the mediation process as part of collaborative data engineering (adapted from [17]).

Table: stops

	stop_id	stop_name	stop_lat	stop_lon
	Filter	Filter	Filter	Filter
1	FUR_CREEK_RES	Furnace Creek Resort (Demo)	36.425288	-117.133162
2	BEATTY_AIRPORT	Nye County Airport (Demo)	36.868446	-116.784582
3	BULLFROG	Bullfrog (Demo)	36.88108	-116.81797
4	STAGECOACH	Stagecoach Hotel & Casino (Demo)	36.915682	-116.751677
5	NADAV	North Ave / D Ave N (Demo)	36.914893	-116.76821
6	NANAA	North Ave / N A Ave (Demo)	36.914944	-116.761472
7	DADAN	Doing Ave / D Ave N (Demo)	36.909489	-116.768242
8	EMSI	E Main St / S Irving St (Demo)	36.905697	-116.76218
9	AMV	Amargosa Valley (Demo)	36.641496	-116.40094

Figure 5.1: An SQLite database file, created by a Jayvee model downloading an open transport data file. The table shown includes stop ids, names, and locations, extracted and validated from the original GTFS format.



RFC 0013: Lossless arithmetic

Feature Tag	`Lossless arithmetic`
Status	`DISCUSSION`
Responsible	`@rhazn`

Summary

As we built out Jayvee and define (arithmetic) expressions, we are not defining a formal semantic of how they are evaluated but rely on our interpreter implementation. With this RFC, we define our goal to support mathematically correct arithmetic expressions that are not limited by JavaScript idiosyncrasies when dealing with numbers.

Motivation

Jayvee is a language to model data pipelines and will therefore handle numeric data as well. For this data, it is important that it arrives in a sink as correctly as possible. Number representation and evaluation of arithmetic expressions has various edge cases in any programming languages, especially in Javascript. Our goal should be to be as correct as possible.

Explanation

The Jayvee interpreter should correctly compare numbers. The Jayvee expression `3.00000000000000000000000000000001 == 3` must evaluate to `false`, not `true`.

Due to the use of javascript for the interpreter, it will evaluate to `true` without special handling.

```
3.00000000000000000000000000000001 === 3
true
```

Use appropriate data types for numbers

- Jayvee `integer` should behave as Integer
- Jayvee `decimal` should behave as Rational number

Figure 5.2: Screenshot of an RFC in discussion to introduce mathematically correct arithmetic expressions in Jayvee.

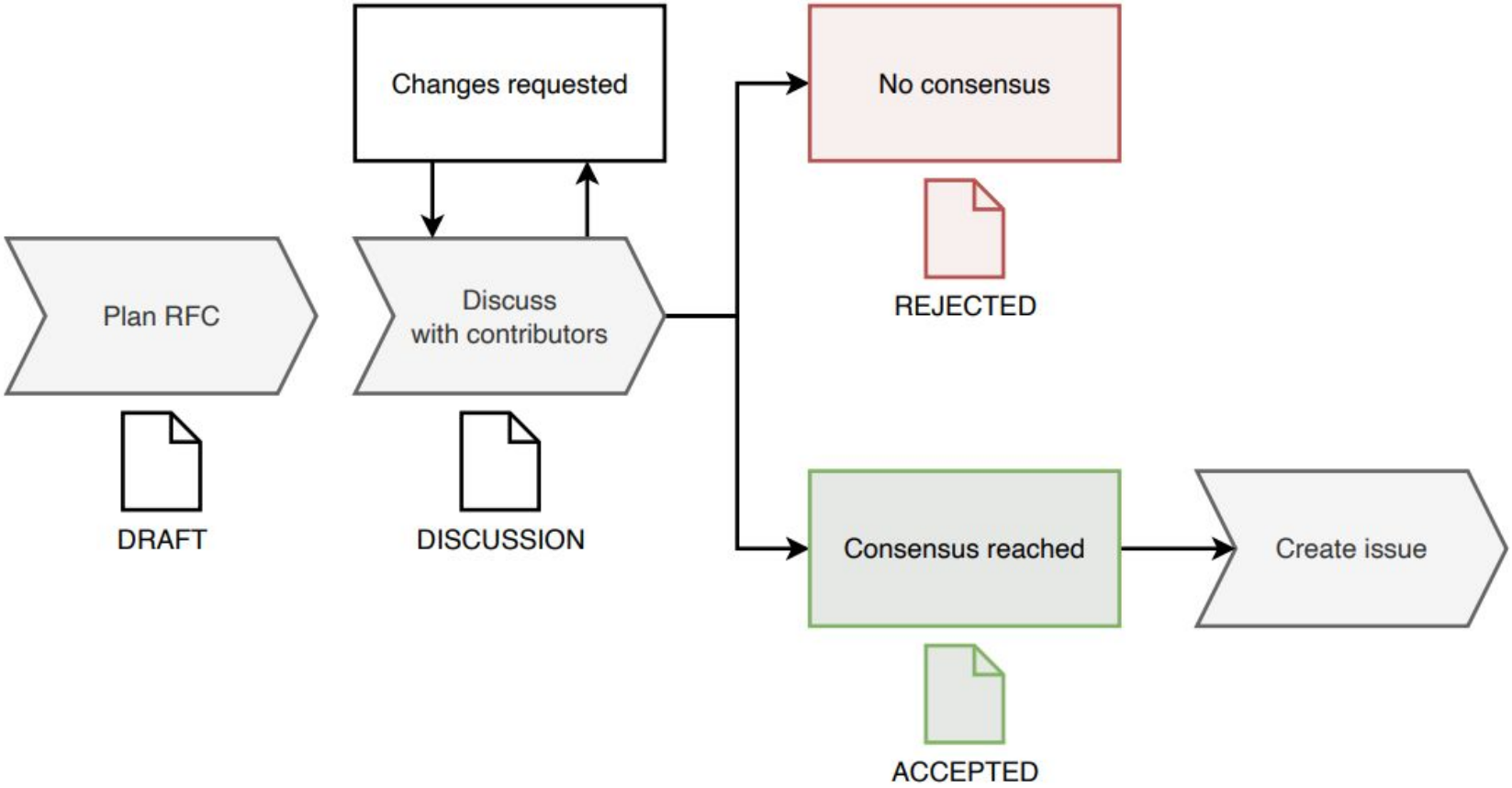


Figure 5.3: RFC process followed during the development of Jayvee.

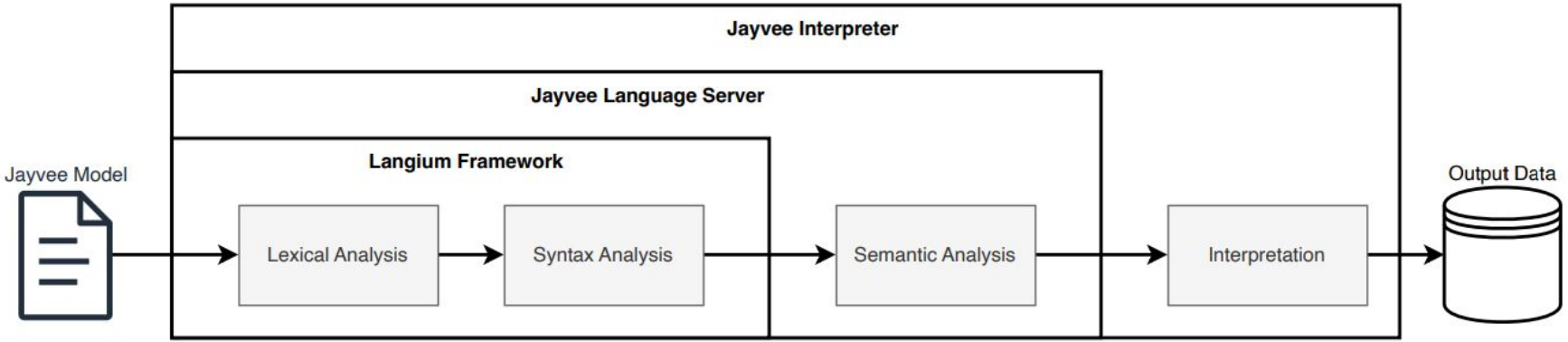


Figure 5.4: Overview of the phases and projects involved in interpreting a Jayvee model, adapted from the Jayvee developer documentation.

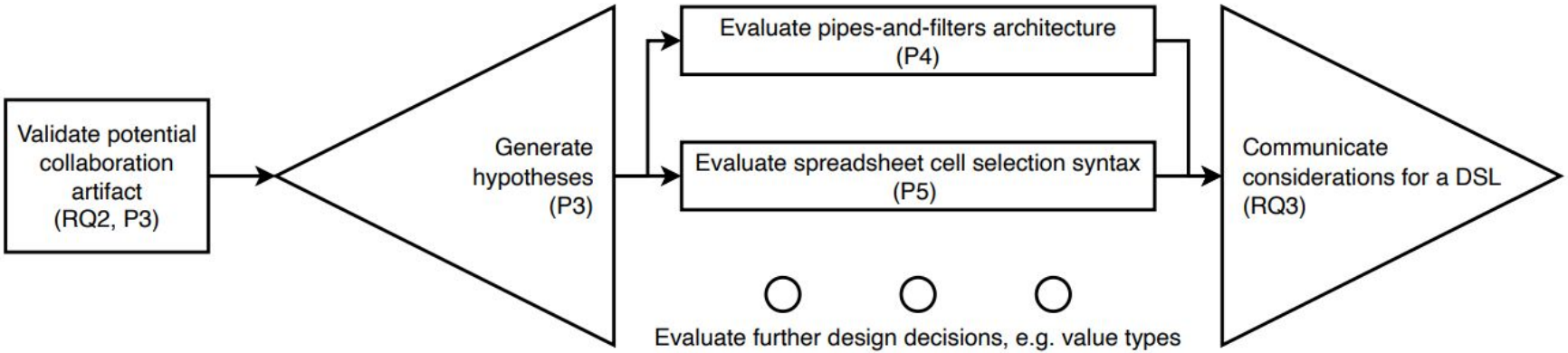


Figure 7.1: Research approach during evaluation and how it relates to research questions (RQ, section 3.1) and publications (P, Table 3.1).

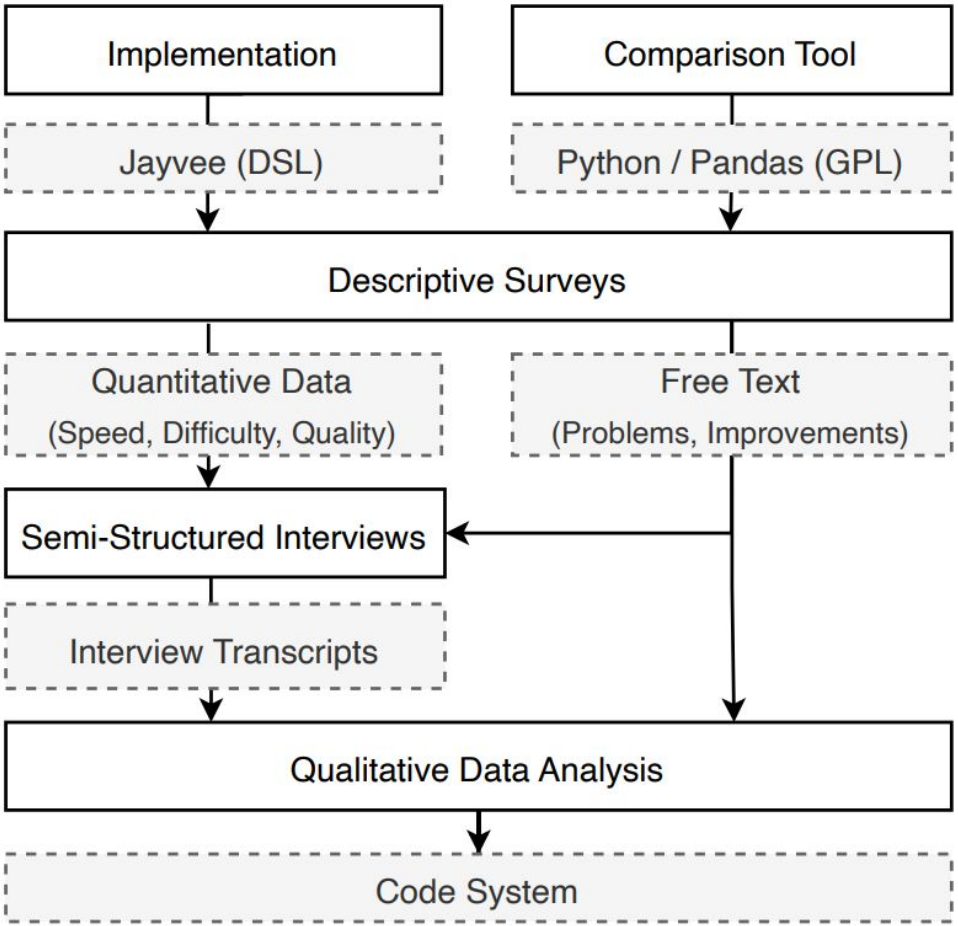


Figure 7.2: Study design as performed for Heltweg et al. [20].

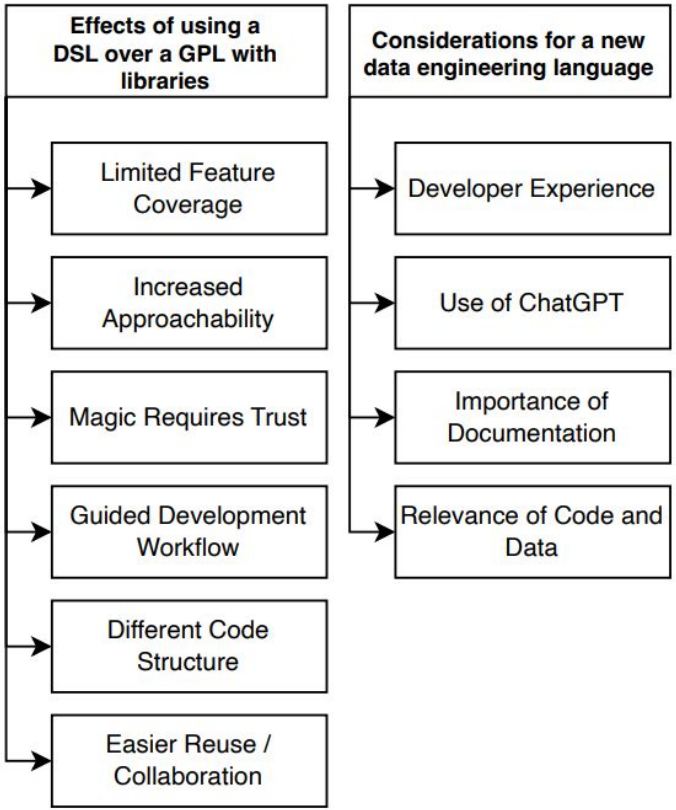


Figure 7.3: Effects of using a pipes-and-filters based DSL on data engineering (adapted from [20]).

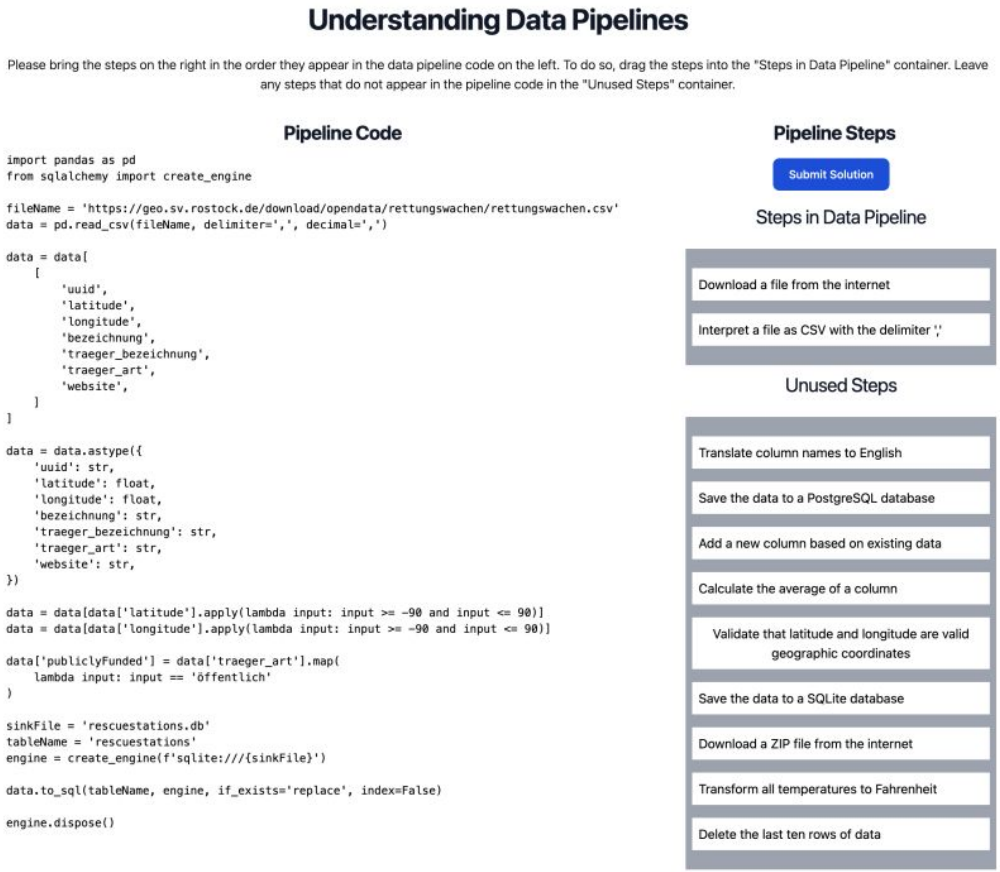


Figure 7.4: Screenshot of a program understanding task as it was displayed in the web-based experiment tool.

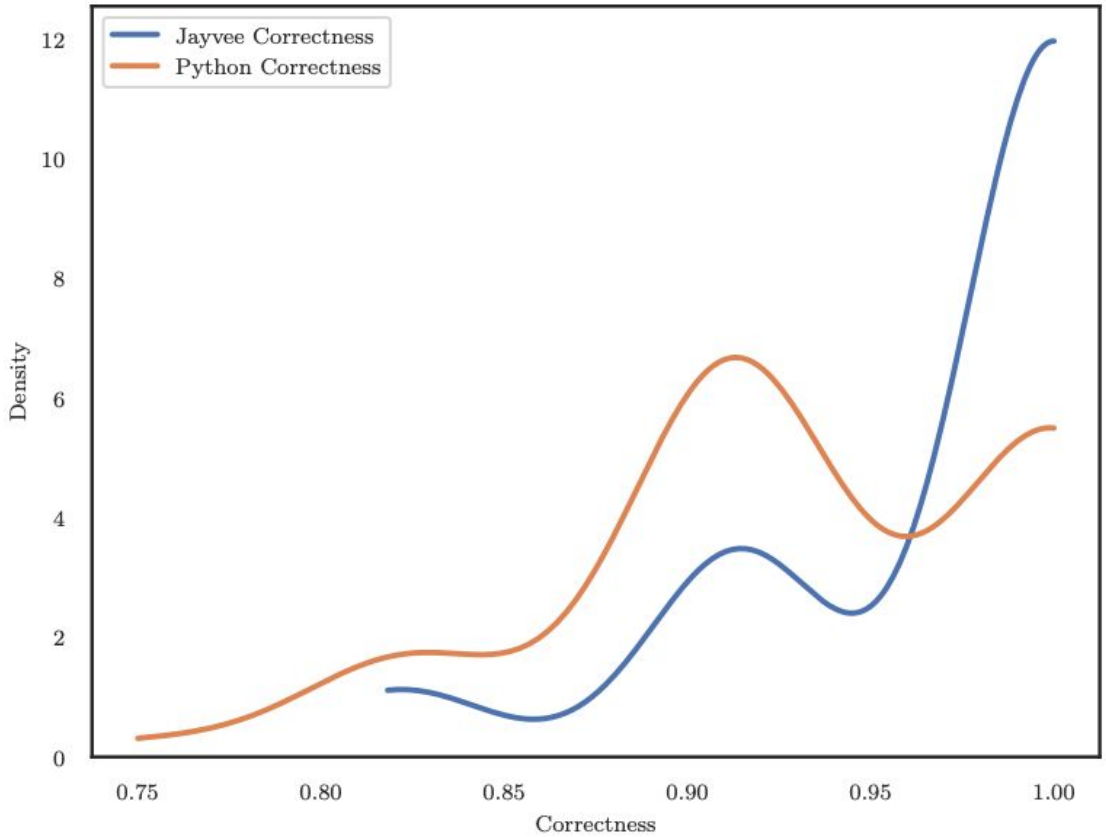


Figure 7.5: Kernel-density plot comparing correctness of task solution using Jayvee and Python/Pandas (adapted from [19]).

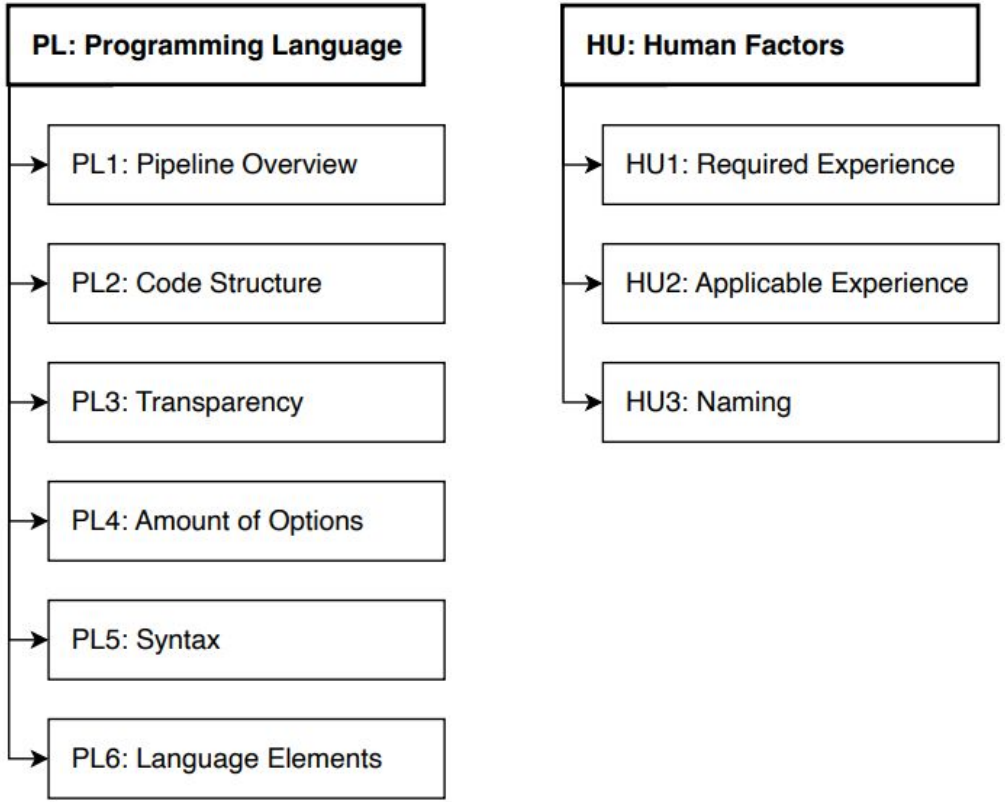


Figure 7.6: Reasons for differences in program structure understanding between Jayvee and Python/Pandas (adapted from [19]).

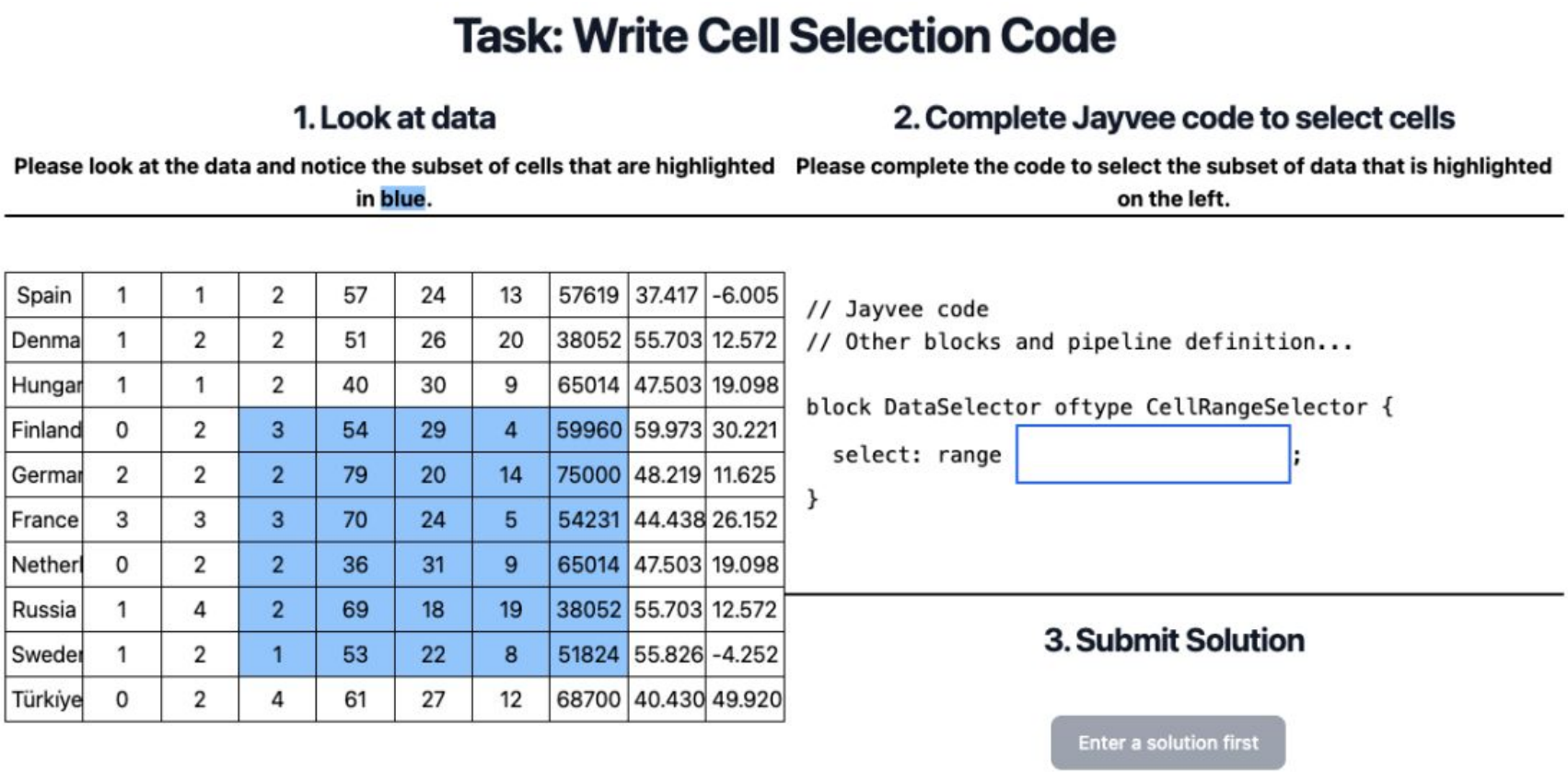


Figure 7.7: Screenshot of a code creation task as it was displayed in the web-based experiment tool.

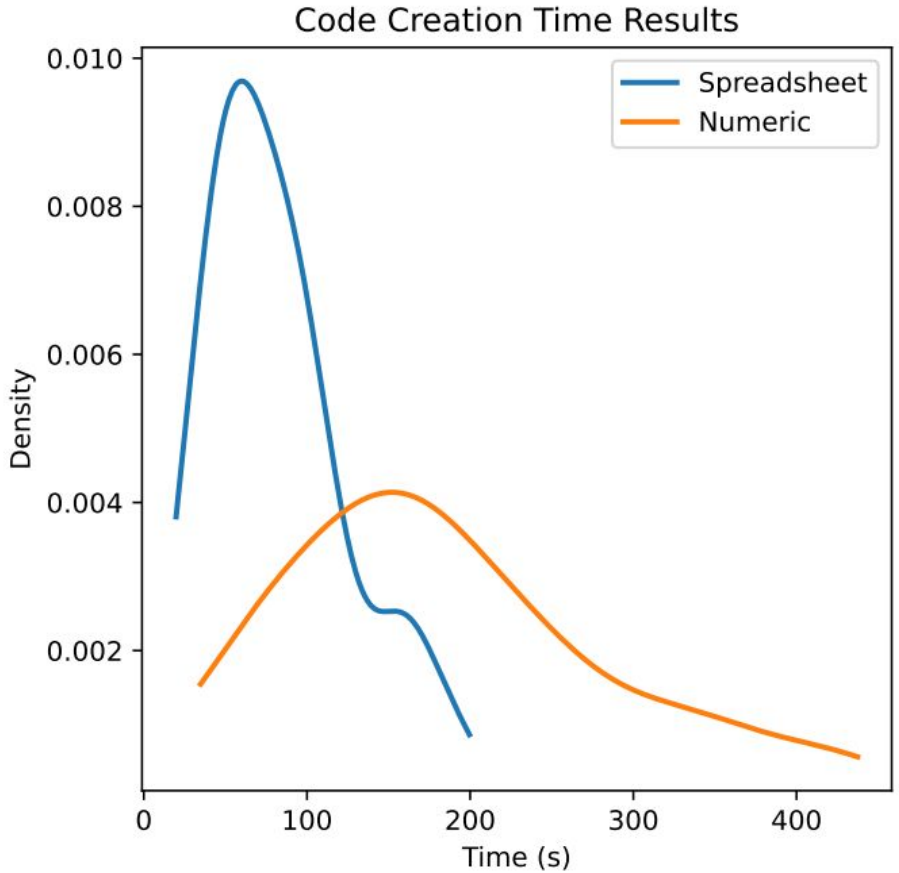


Figure 7.8: Kernel-density plot comparing time on task for spreadsheet syntax compared to numerical syntax (adapted from [18]).

Project	Methods	Publication	Results	Described In
P1	Systematic Literature Review [30] Descriptive Data Synthesis [30]	[16]	1. Overview of participants, activities, tools used and artifacts created during collaborative data engineering 2. Description of first challenges to open collaboration	Section 4.2
P2	Qualitative Survey [24] Descriptive Data Synthesis [30]	[17]	1. Identified challenges to open collaboration 2. Guidelines to enable open collaboration	Section 4.3
P3	Descriptive Survey [31] Thematic Analysis [4]	[20]	1. Viability of a DSL for data engineering by SMEs 2. Description of major effects when using a DSL for data engineering 3. Important hypotheses to test	Section 7.2
P4	Controlled Experiment [33] Descriptive Survey [31] Thematic Analysis [4]	[19]	1. Effects of a DSL on data pipeline structure understanding 2. Descriptions of reasons for the effects	Section 7.3
P5	Controlled Experiment [33]	[18]	1. Effects of spreadsheet syntax on DSL efficiency	Section 7.4

Table 3.1: Overview of projects, methods, and publications that were completed as part of the thesis.

Acquire	Assess	Communicate	Extend	Improve	Maintain	Understand
Build Infrastructure	Ensure Anonymity	Ask Publisher	Add Metadata	Aggregate	Archive	Analyze
Discover	Evaluate	Discuss	Create Features	Clean	Document	Ask Experts
Extract	Preview	Find Community	Label	Combine	Refresh	Experiment
Read Documentation	Measure Availability	Find Skilled Users	Rate	Curate		Learn Subject-matter Knowledge
Search	Verify License	Give Feedback	Translate	Enrich		Learn Structure
Select	Visualize / Plot Data	Request Data		Link		
Store		Share Data (Publisher)		Normalize		
Validate		Share Data (Stakeholders)		Reformat		
		Share Information		Repair		
				Structure		

Table 4.1: Activities performed during data engineering (adapted from [16]).

Participants	
Businesses	Mediators
Citizen Scientists	NGOs
Civil Servants	Open Data Experts
Data Scientists	Organisations
Subject-matter Experts	Private Citizens
Goverment Agencies	Researchers
Hackathon Participants	Software Developers
Infomediaries	Startups/Entrepreneurs
Journalists	Students
Legal Advisors	

Table 4.2: Participants in data engineering, by user role (adapted from [16]).

ID	Type	Title
C1	Technical	Need for specialized skills but high barriers to participation
C2	Social	Finding and connecting with community members
C3	Social	No well-understood collaboration practices
C4	Technical	No standard tools or artifacts
C5	Technical	Data representation
C6	Technical	Inadequate tools
C7	Technical	Infrastructure for data projects
C8	Technical	Bad data sources
C9	Social	Conflicts with data publishers
C10	Social	Unclear data use cases
C11	Social	Data semantics
C12	Social	Missing incentives
C13	Social	Missing knowledge

Table 4.3: Challenges to open collaborative data engineering (adapted from [17]).

ID	Guideline
G1	Plan with data problems like distributed sources, updates, low-quality, and limited access to publishers
G2	Make projects accessible to data engineers, software developers, and subject-matter experts
G3	Enable collaboration by agreeing on standards, improving project visibility, and curating data
G4	Support projects with tools built specifically for collaborative data engineering

Table 4.4: Guidelines to enable open collaborative data engineering (adapted from [17]).

Paper 1 - Figures/Tables

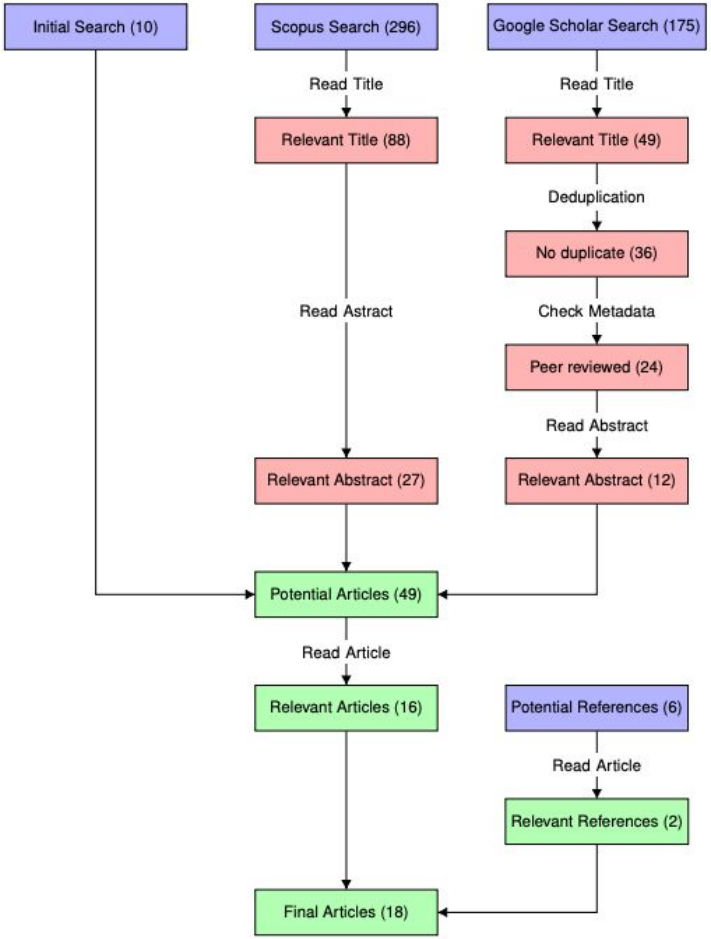


Figure 1. Process of the systematic literature review

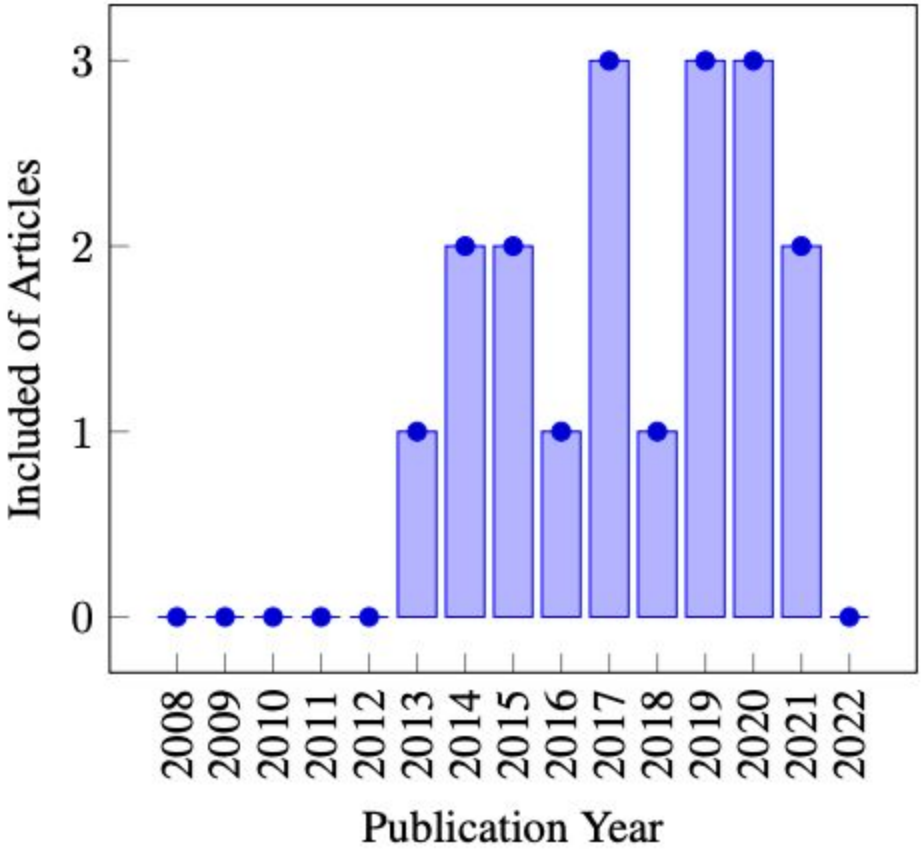


Figure 2. Publication date of included articles

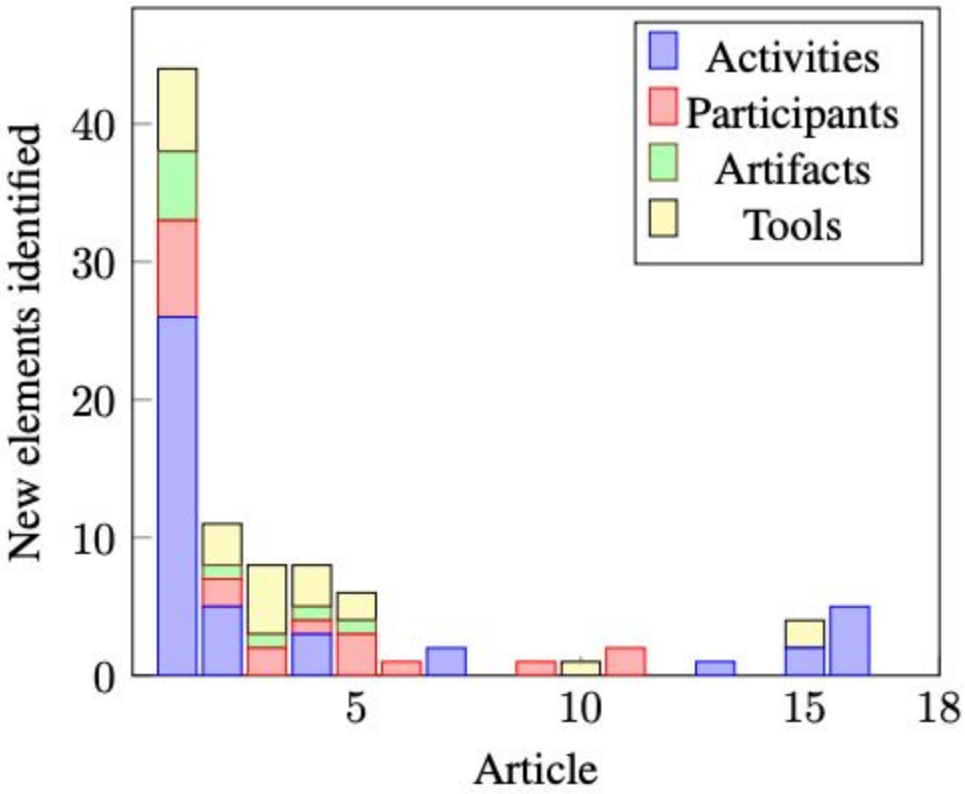


Figure 3. New elements identified by article

Participants	
Businesses	Mediators
Citizen Scientists	NGOs
Civil Servants	Open Data Experts
Data Scientists	Organisations
Domain Experts	Private Citizens
Goverment Agencies	Researchers
Hackathon Participants	Software Developers
Infomediaries	Startups/Entrepreneurs
Journalists	Students
Legal Advisors	

Table 2. Participants in data engineering, by user role

Acquire	Assess	Communicate	Extend	Improve	Maintain	Understand
Build Infrastructure	Ensure Anonymity	Ask Publisher	Add Metadata	Aggregate	Archive	Analyze
Discover	Evaluate	Discuss	Create Features	Clean	Document	Ask Experts
Extract	Preview	Find Community	Label	Combine	Refresh	Experiment
Read Documentation	Measure Availability	Find Skilled Users	Rate	Curate		Learn Domain Knowledge
Search	Verify License	Give Feedback	Translate	Enrich		Learn Structure
Select	Visualize / Plot Data	Request Data		Link		
Store		Share Data (Publisher)		Normalize		
Validate		Share Data (Stakeholders)		Reformat		
		Share Information		Repair		
				Structure		

Table 3. Activities performed during data engineering by open data users

Tools used	
Auth Providers	Kaggle
Big Data Processing Tools	Notebooks
Blogs / Websites	Official Discussion Board
Command Line Tools	Open Data Repositories
Data Science Libraries	Open Refine
Databases	Sheet Software
Domain Specific Languages	Statistical Computing Languages
Domain Specific Software	Translation Software
General Purpose Languages	Travis
git	Visualization Tools
GitHub	Wikis

Table 4. Tools used during data engineering by open data users

Created Artifacts	
CI Definitions	Notebooks
Comments on Data	Processed Data
Data Quality Ratings	Raw Data
Documentation	Software Applications
Feedback-/Experience Reports	Source Code
Metadata	

Table 5. Created artifacts by open data users during data engineering

Paper 2 - Figures/Tables

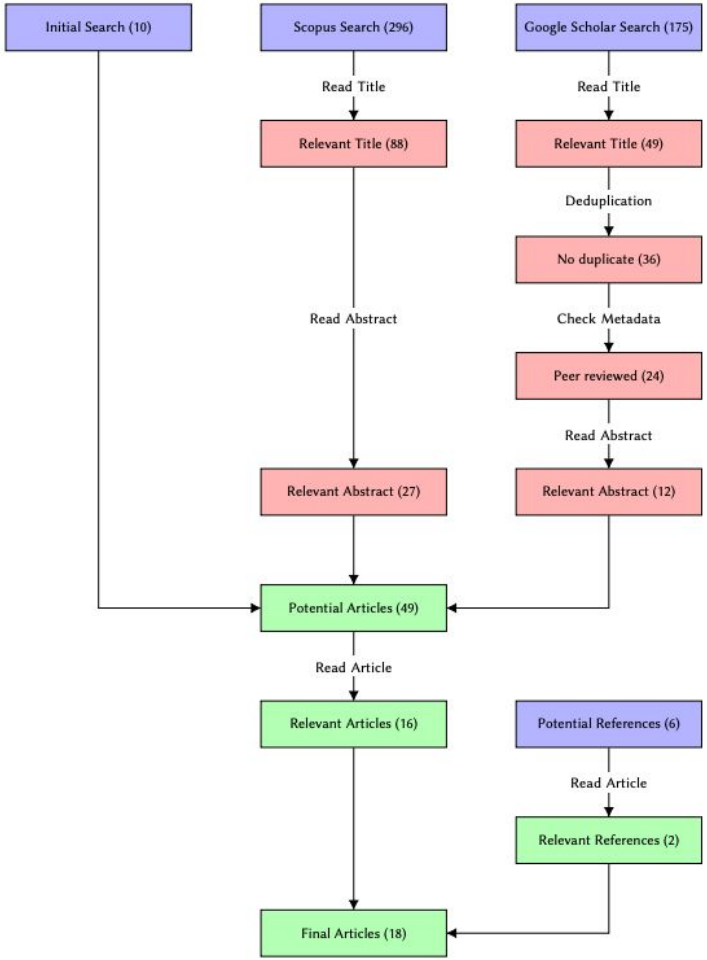


Fig. 1. Process of the systematic literature review

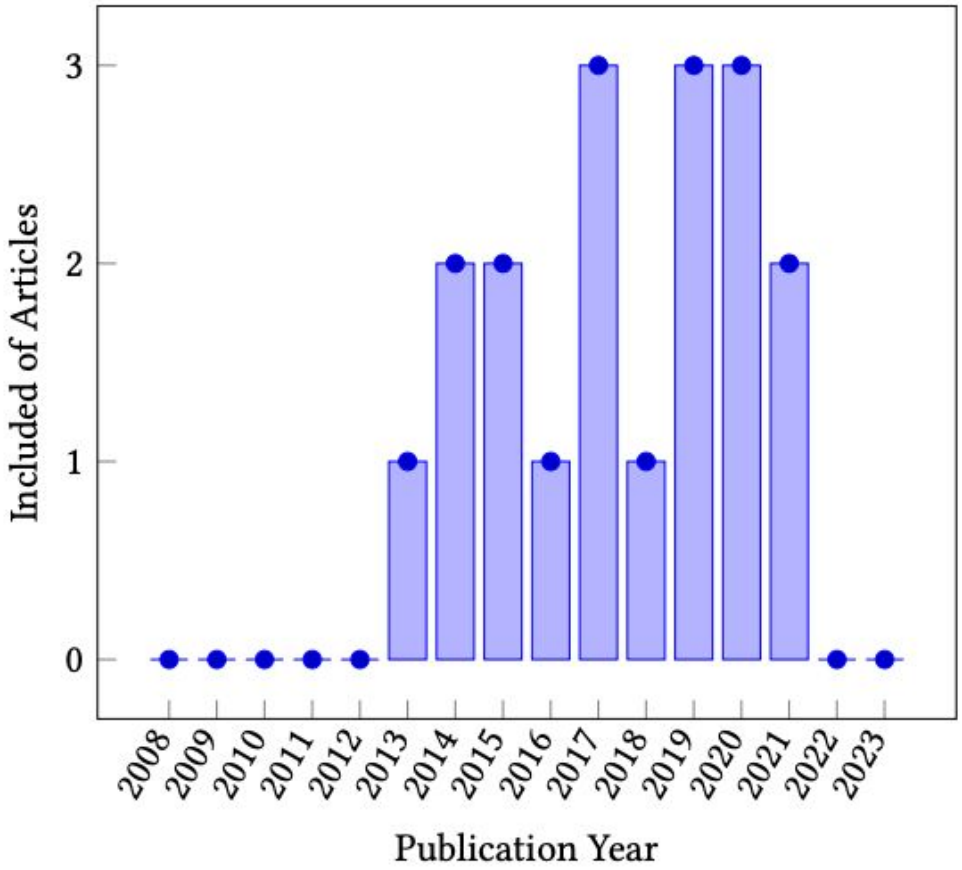


Fig. 2. Publication date of included articles

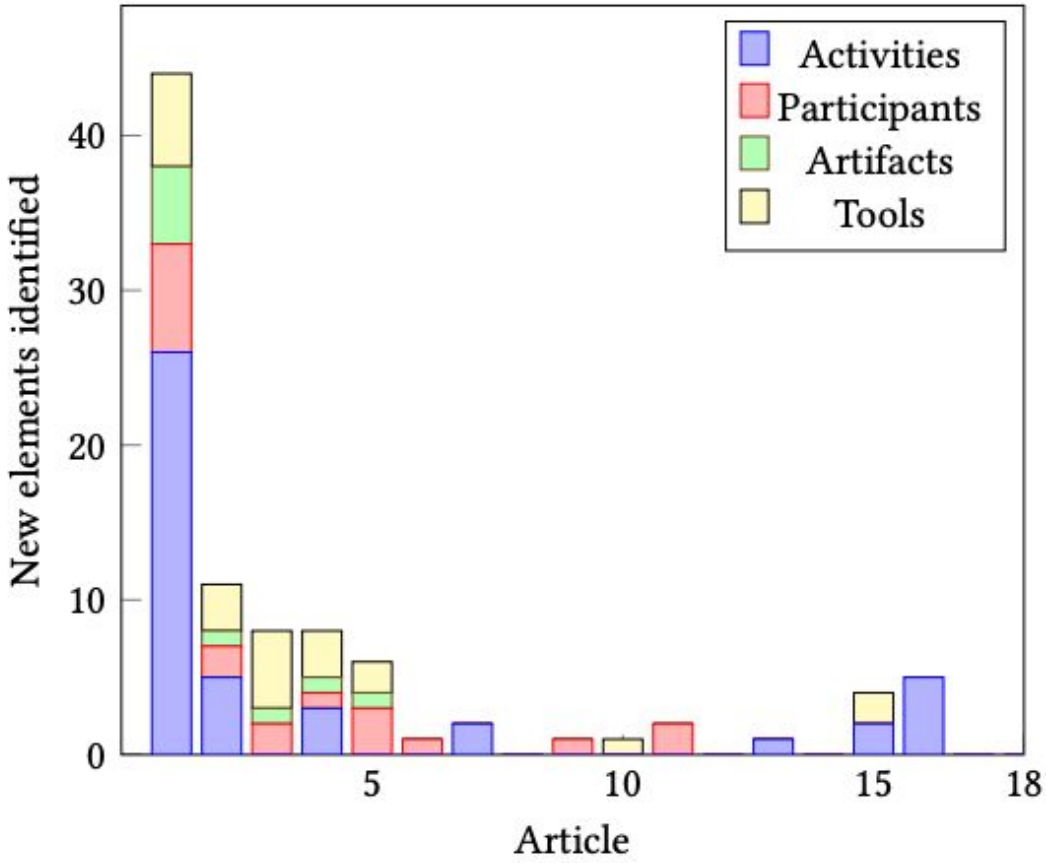


Fig. 3. New elements identified by article

Pseudonym	Job Role	Employer	Project Types	Data Domains	Professional	Open Data
E1	Data acquisition	Academic publisher	Data curation	Material science	Yes	Both
E2	Software engineer	Software agency	Civic society	Transport	No	Yes
E6	Data engineer	Nonprofit organisation	Hackathons	Transport, Energy, Political	No	Yes
E8	Executive	Software agency	Hackathons	Geographical, Financial	Both	Yes
E9	Data engineer	Nonprofit organisation	Scientific	Medical, Biological	Yes	No

Table 1. Participants in semi-structured interviews

Participants	
Businesses	Mediators
Citizen Scientists	NGOs
Civil Servants	Open Data Experts
Data Scientists	Organizations
Subject-matter Experts	Private Citizens
Government Agencies	Researchers
Hackathon Participants	Software Developers
Infomediaries	Startups/Entrepreneurs
Journalists	Students
Legal Advisors	

Table 3. Participants in data engineering

Acquire	Assess	Communicate	Extend	Improve	Maintain	Understand
Build Infrastructure	Ensure Anonymity	Ask Publisher	Add Metadata	Aggregate	Archive	Analyze
Discover	Evaluate	Discuss	Create Features	Clean	Document	Ask Experts
Extract	Preview	Find Community	Label	Combine	Refresh	Experiment
Read Documentation	Measure Availability	Find Skilled Users	Rate	Curate		Learn subject-matter knowledge
Search	Verify License	Give Feedback	Translate	Enrich		Learn Structure
Select	Visualize / Plot Data	Request Data		Link		
Store		Share Data (Publisher)		Normalize		
Validate		Share Data (Stakeholders)		Reformat		
		Share Information		Repair		
				Structure		

Table 4. Activities performed during data engineering by open data users

Tools used	
Auth Providers	Kaggle
Big Data Processing Tools	Notebooks
Blogs / Websites	Official Discussion Board
Command Line Tools	Open Data Repositories
Data Science Libraries	Open Refine
Databases	Sheet Software
Domain-Specific Languages	Statistical Computing Languages
Domain-Specific Software	Translation Software
General Purpose Languages	Travis
git	Visualization Tools
GitHub	Wikis

Table 5. Tools used during data engineering by open data users

Created Artifacts	
CI Definitions	Notebooks
Comments on Data	Processed Data
Data Quality Ratings	Raw Data
Documentation	Software Applications
Feedback-/Experience Reports	Source Code
Metadata	

Table 6. Created artifacts by open data users during data engineering

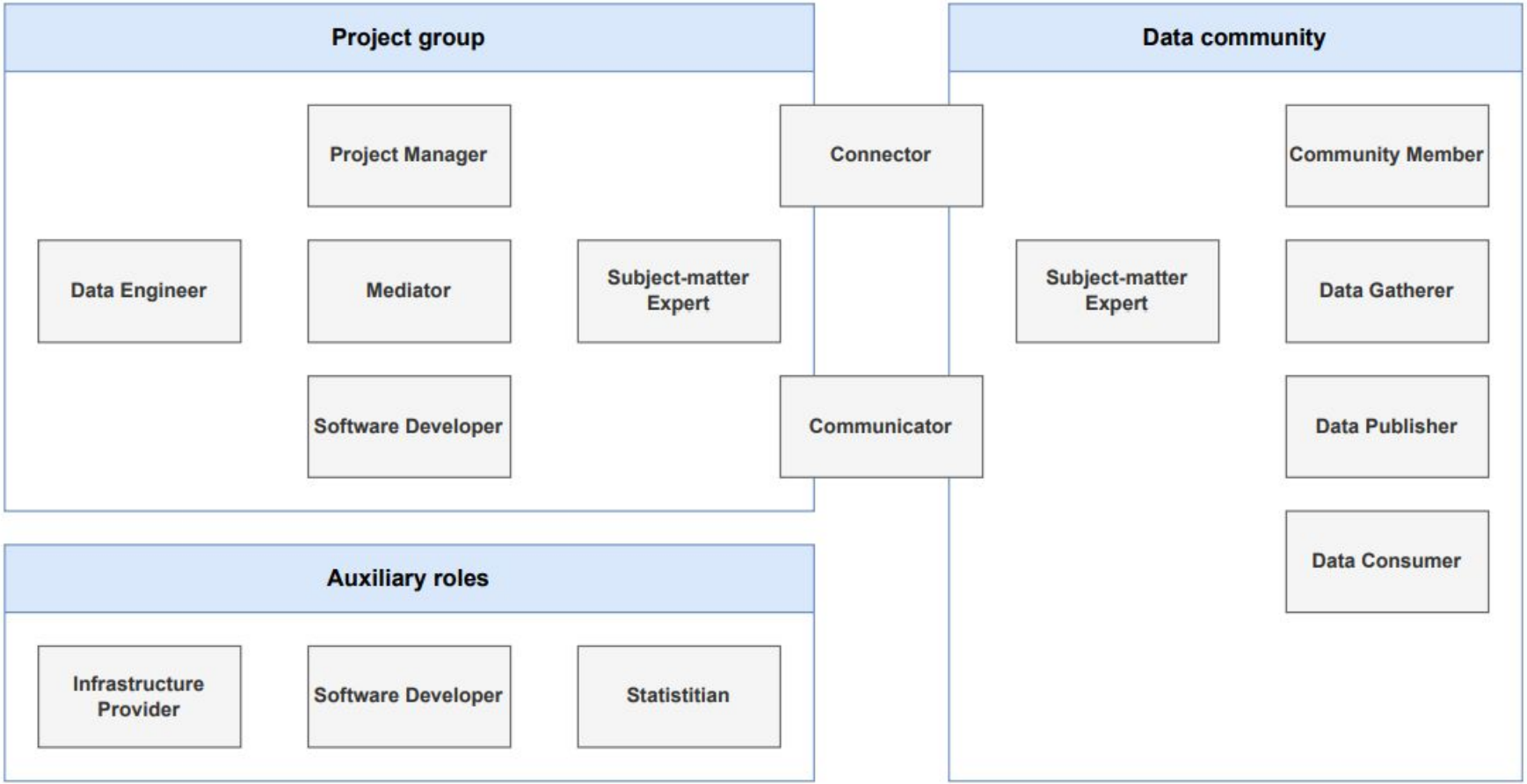


Fig. 4. Roles in collaborative data engineering as identified in interviews

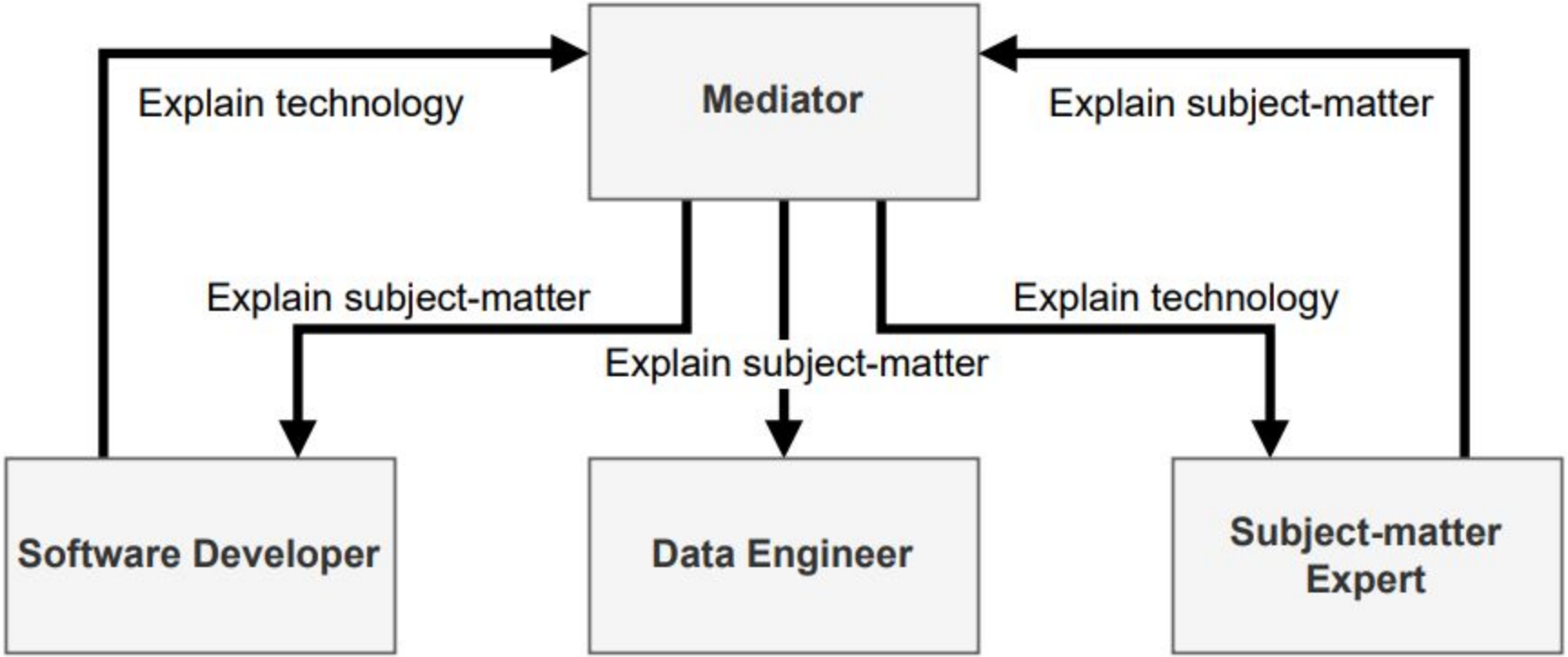


Fig. 5. Mediate during collaborative data engineering

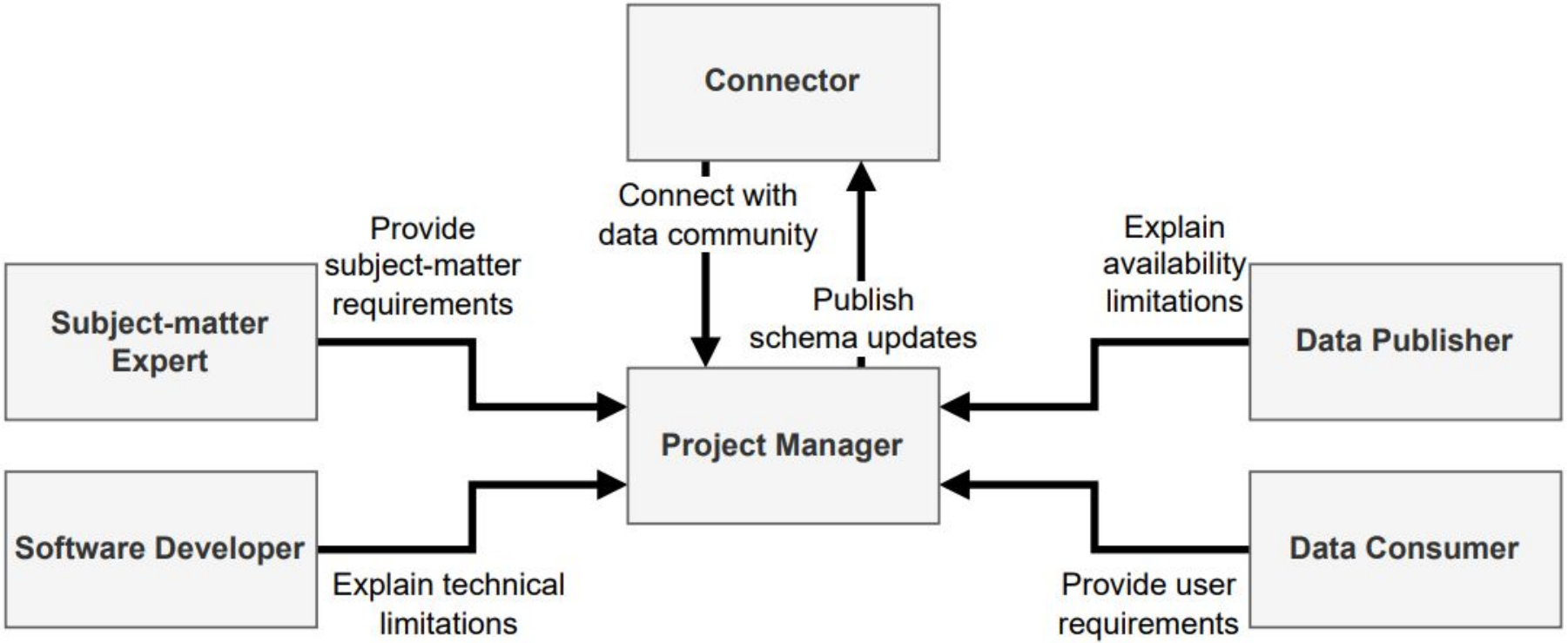


Fig. 6. Develop data schemas during collaborative data engineering

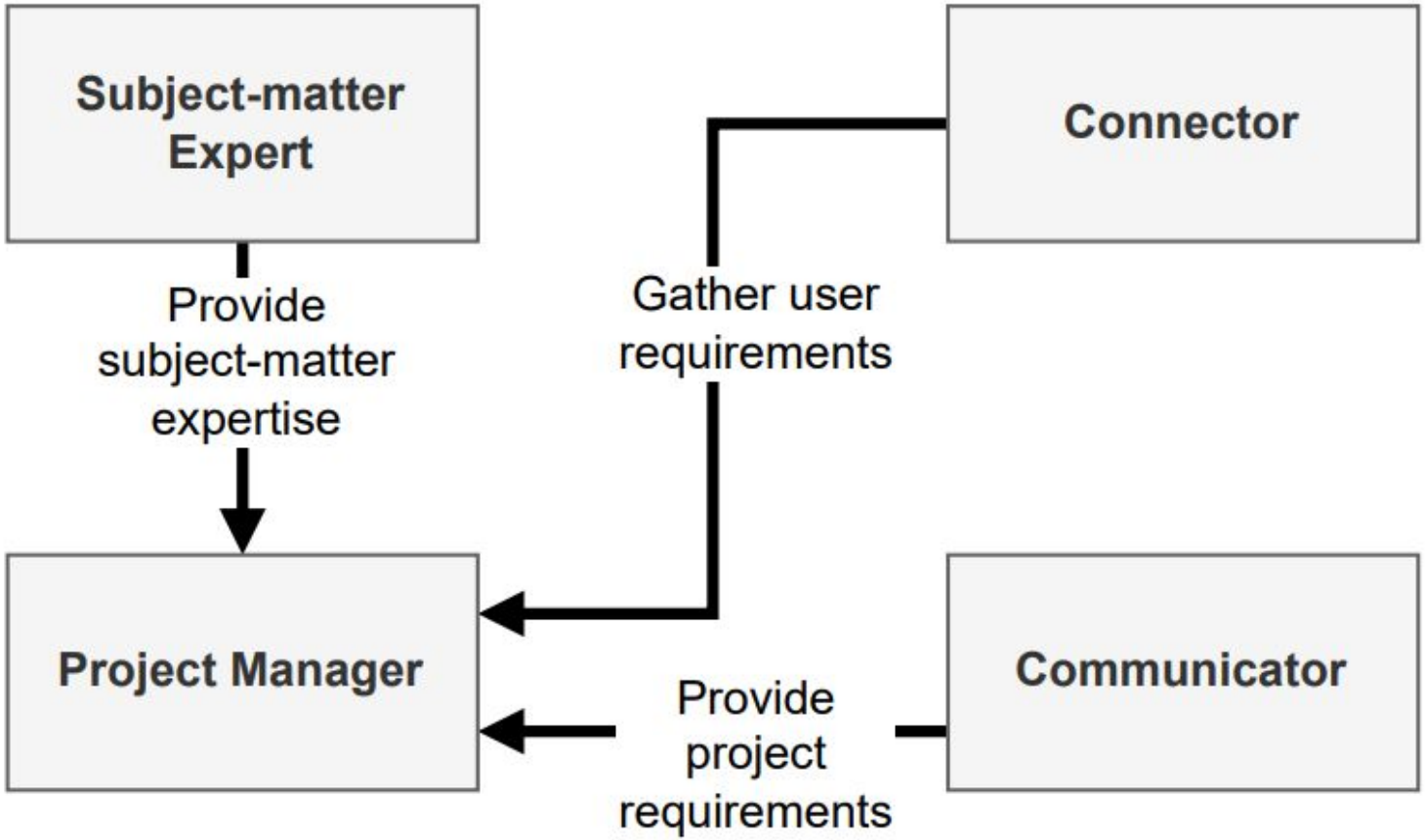


Fig. 7. Prioritize during collaborative data engineering

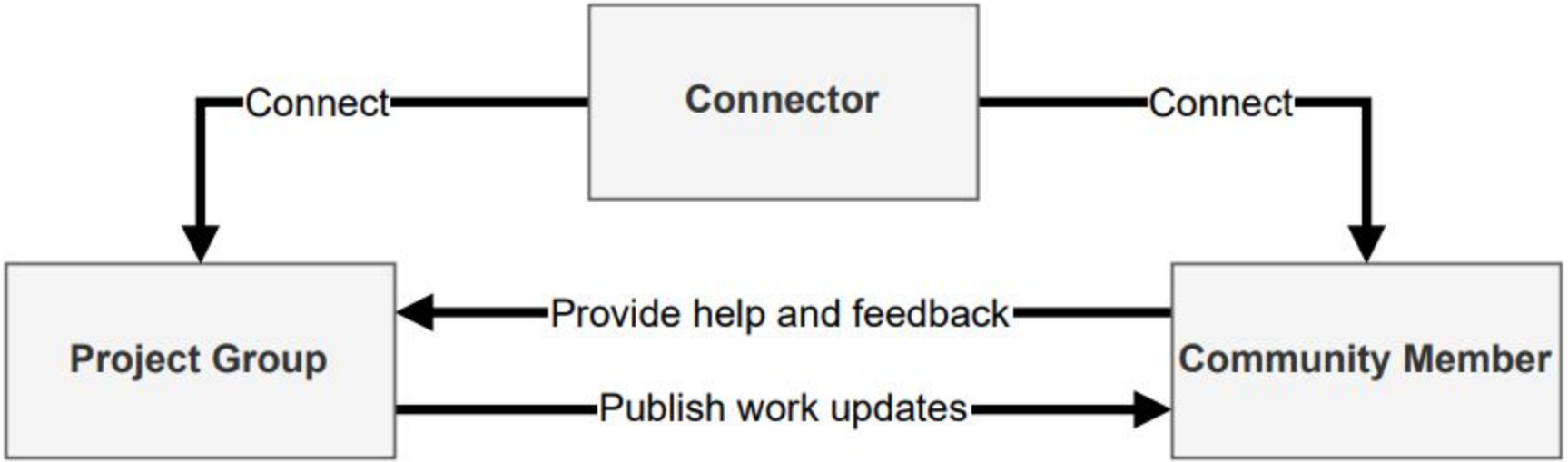


Fig. 8. Public work during collaborative data engineering

ID	Type	Title	Mentioned by
C1	SLR	Need for specialized skills but high barriers to participation	E1, E2, E8, E9
C2	SLR	Finding and connecting with community members	E1, E2, E6, E8
C3	SLR	No well-understood collaboration practices	E1, E2, E6, E8, E9
C4	SLR	No standard tools or artifacts	E1, E2, E6, E8
C5	Technical	Data representation	E1, E2, E6
C6	Technical	Inadequate tools	E2, E8, E9
C7	Technical	Infrastructure for data projects	E8
C8	Technical	Bad data sources	E1, E2, E8, E9
C9	Social	Conflicts with data publishers	E1, E2, E8
C10	Social	Unclear data use cases	E1, E2
C11	Social	Data semantics	E1, E2, E6
C12	Social	Missing incentives	E1, E6, E8
C13	Social	Missing knowledge	E1, E2, E6, E8, E9

Table 7. Challenges identified from literature and interviewee experiences

ID	Guideline	Based on
G1	Plan with data problems like distributed sources, updates, low-quality and limited access to publishers	C5, C8, C9, C12
G2	Make projects accessible to data engineers, software developers and subject-matter experts	Social Systems, C1, C13
G3	Enable collaboration by agreeing on standards, improving project visibility and curating data	C2, C3, C4, C5, C7, C8, C12
G4	Support projects with tools, built specifically for collaborative data engineering	C1, C4, C6

Table 8. Guidelines To Enable Open Collaborative Data Engineering

ID	Recommendation	Based on
R1	Define a standard artifact to collaboratively develop data pipelines	G1, G2, G3
R2	Adapt proven open collaborative workflows for data engineering	G2, G3
R3	Provide a project forge to drive adoption of standards	R1, R2, G4
R4	Develop tools that make running data pipelines and hosting data projects easier	G4
R5	Support the creation of data communities	G3

Table 9. Recommendations to Increase Open Collaborative Data Engineering in Open Data

Paper 3 - Figures/Tables

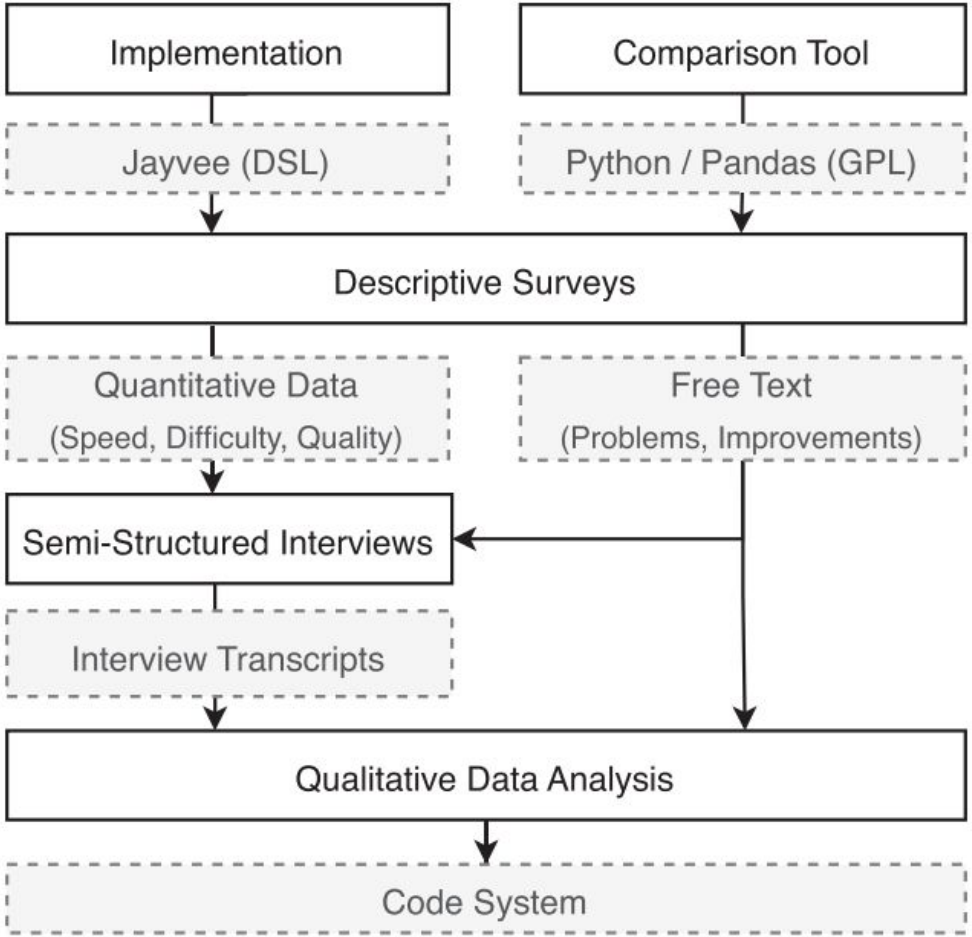


FIGURE 1 | Overview of the research design.

TABLE 1 | Task summaries.

No	Task summary
1	Extract a CSV dataset from an HTTP source and assign fitting data types to each column. Save the data to a SQLite database.
2	Extract a CSV dataset from an HTTP source. Transform data shape. Validate data, as defined by categories, integer ranges, and regex patterns. Remove all rows that contain invalid values. Choose fitting data types and save the data to SQLite.
3	Extract a CSV dataset from an HTTP source. Fix invalid format due to included metadata. Handle uncommon encoding to preserve German umlaut characters. Transform data shape by dropping multiple, not adjacent columns. Validate data, handling a special value type of numeric data with leading zeros. Remove any rows containing invalid data. Choose fitting data types and save the data to SQLite.
4	Extract a ZIP file from an HTTP source. Pick one CSV file from the multiple files in the source. Transform the data shape by renaming and dropping columns. Transform data values from Celsius to Fahrenheit. Choose fitting data types and save the data to SQLite.
5	Extract a GTFS file (an open format for transit data in one ZIP file with multiple CSV files) from an HTTP source. Pick one file from the archive. Transform the data shape by dropping columns. Filter the data to only keep rows related to one ID. Validate data values according to integer ranges and keep German umlaut characters intact. Choose fitting data types and save the data to SQLite.

TABLE 3 | Sample size, median and Mann–Whitney U and p-value for previous population experience for H_A^{Exp}.

Experience	n_1, n_2	Mdn_1, Mdn_2	U	p (two-sided)
Programming	110, 113	7, 7	6224.0	0.986
Python	110, 113	4, 4	6198.0	0.971
Jayvee	110, 113	1, 1	6547.5	0.223
Jayvee vs. Python	223, 223	4, 1	48398.0	**1.406e-74***

*$p \leq 0.05$.

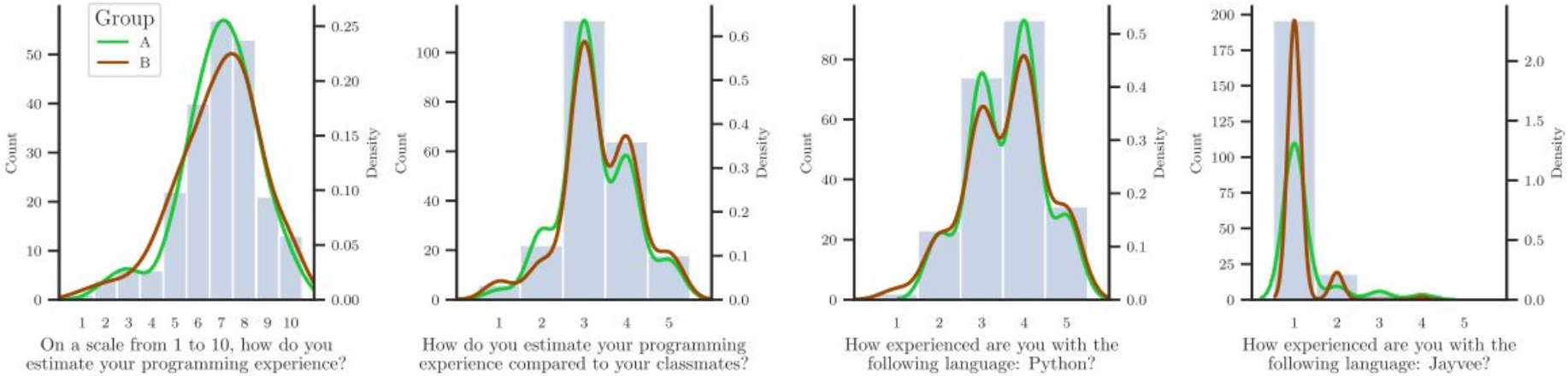


FIGURE 2 | Results regarding student's previous experience from the course entry survey.

TABLE 4 | Sample size, median and Mann–Whitney U and p-value for time spent on exercises for H_A^{Speed}.

Exercise	n_{jv}, n_{py}	Mdn_{jv}, Mdn_{py}	U	p (less)
Ex1	45, 47	2.0, 2.0	1159.5	0.794
Ex2	28, 24	2.0, 3.0	243.0	**0.042***
Ex3	17, 8	3.0, 3.0	72.0	0.606
Ex4	18, 15	2.0, 2.5	113.0	0.202
Ex5	16, 16	2.0, 1.0	162.5	0.915

*$p \leq 0.05$.

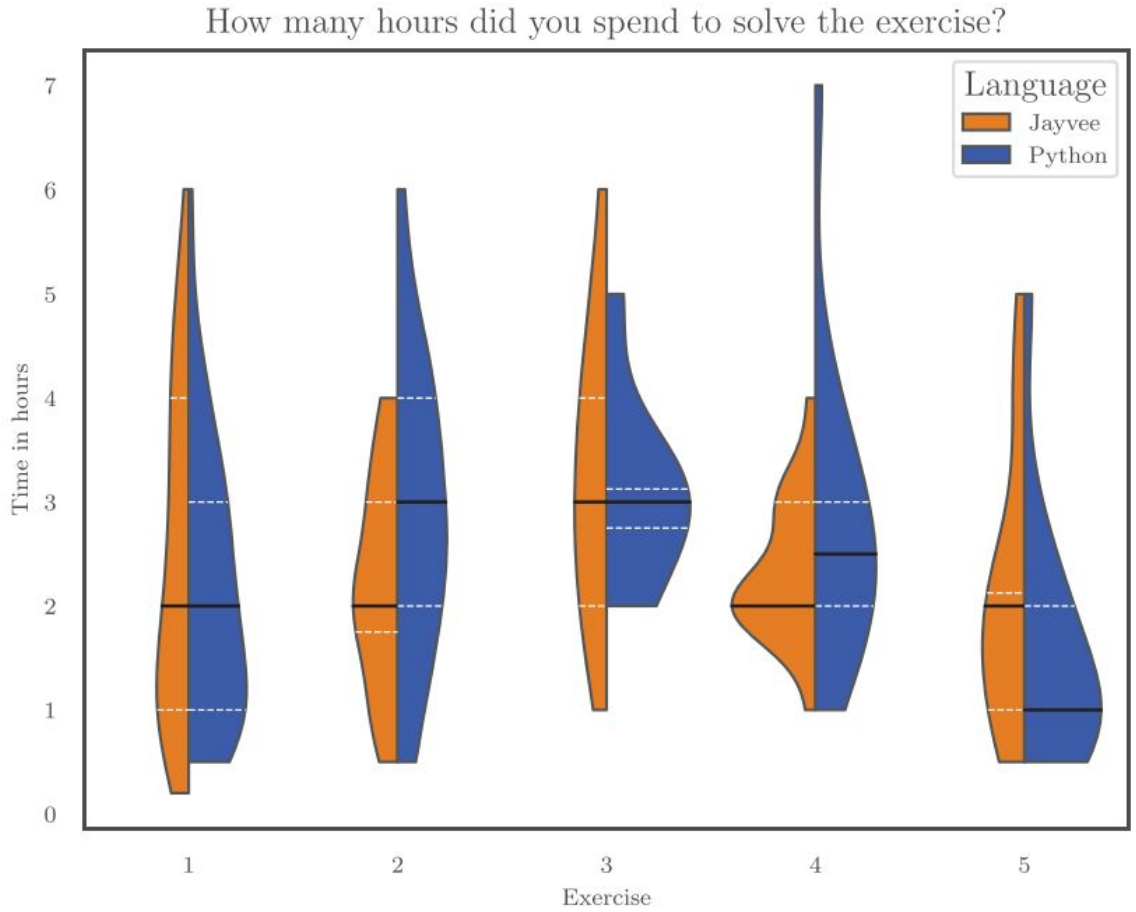


FIGURE 3 | Distribution of time spent per exercise, depending on the language used (lower is better). White bars represent Q_1 and Q_3, the black bar denotes Q_2.

TABLE 5 | Sample size, median and Mann–Whitney U and p-value for the difficulty of exercises for H_A^{Diff}.

Exercise	n_{jv}, n_{py}	Mdn_{jv}, Mdn_{py}	U	p (less)
Ex1	47, 48	2.0, 2.0	1086.5	0.372
Ex2	33, 28	2.0, 3.0	397.5	0.158
Ex3	17, 8	3.0, 3.0	93.0	0.943
Ex4	19, 16	2.0, 3.0	102.0	**0.040***
Ex5	16, 17	2.5, 3.0	141.0	0.584

*$p \leq 0.05$.

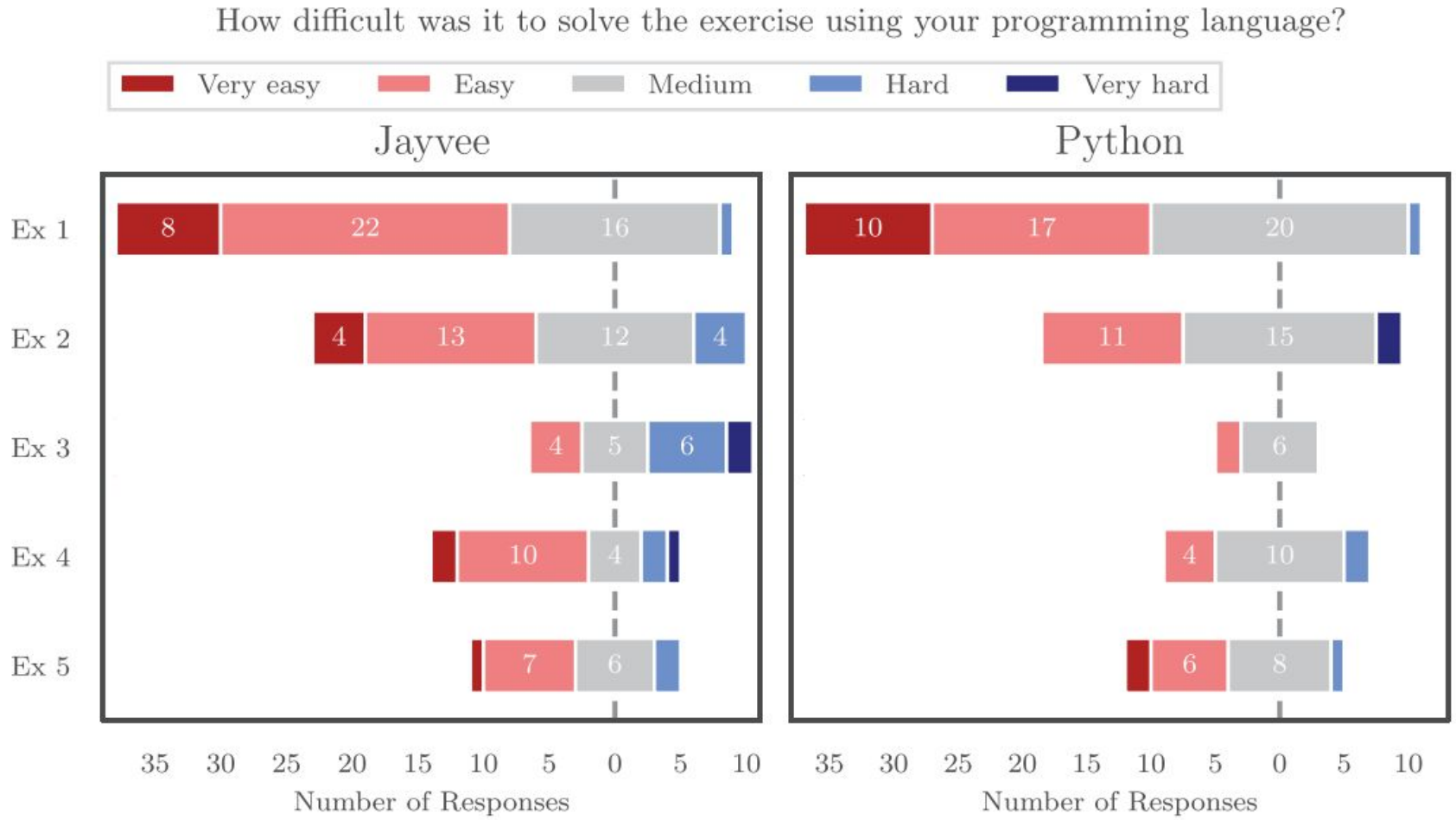


FIGURE 4 | Impressions of the difficulty of completing the exercise, depending on the language used (lower is better).

TABLE 6 | Sample size, median and Mann–Whitney U and p-value for quality of exercise results for H_A^{Qual}.

Exercise	n_{jv}, n_{py}	Mdn_{jv}, Mdn_{py}	U	p (greater)
Ex1	47, 48	4.0, 3.0	1377.0	**0.021***
Ex2	33, 28	3.0, 3.0	515.0	0.200
Ex3	17, 8	3.0, 4.0	47.5	0.911
Ex4	19, 16	4.0, 4.0	122.5	0.873
Ex5	16, 17	4.0, 4.0	107.5	0.884

*$p \leq 0.05$.

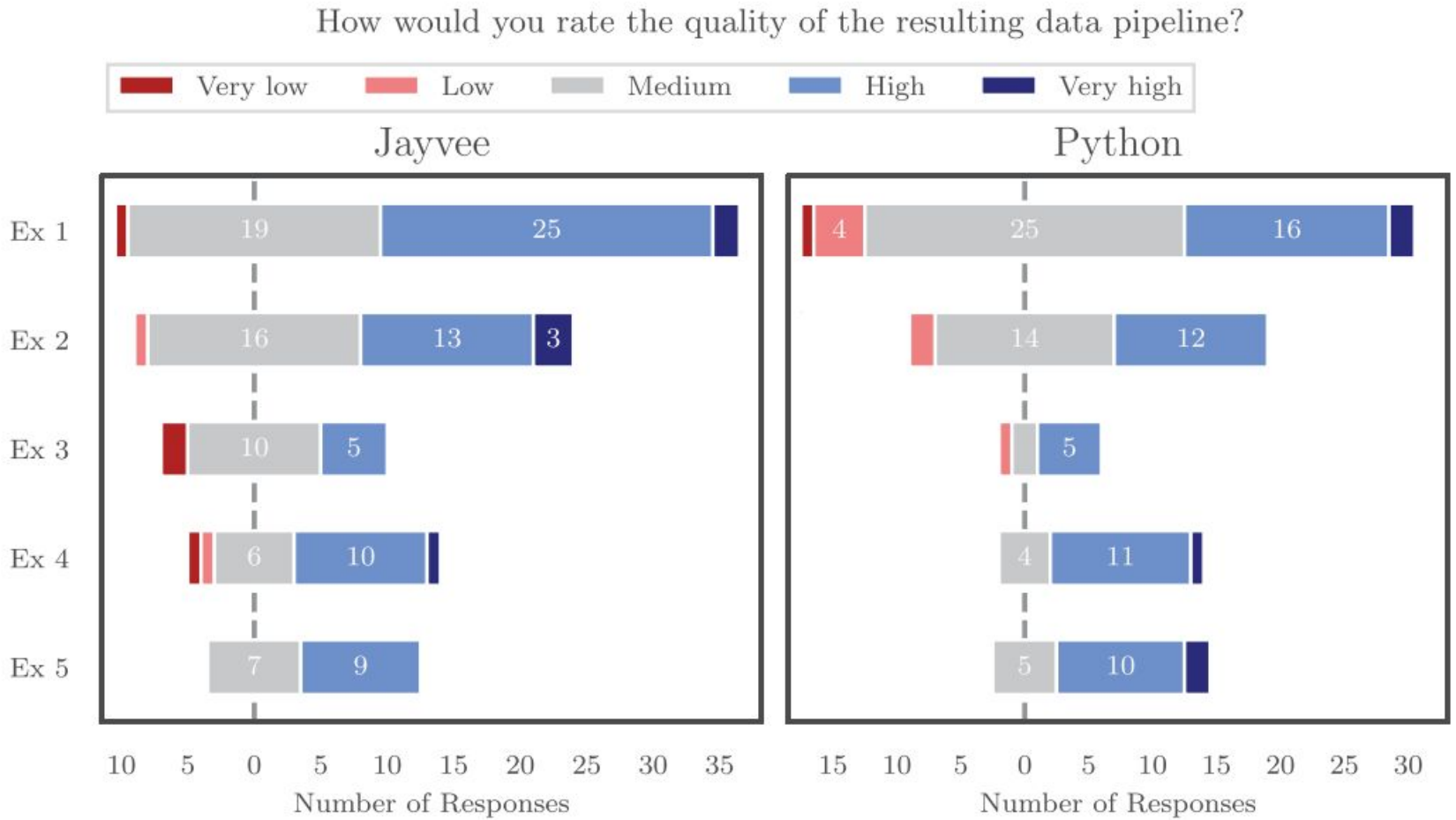


FIGURE 5 | Impressions of the quality of the resulting pipeline, depending on the language used (higher is better).

TABLE 7 | Experience of interview participants after completing the data engineering course.

Participant	Programming (of 10)	Python (of 5)	Jayvee (of 5)
S0	8	4	5
S2	7	4	3
S3	6	3	2
S5	9	5	4
S7	7	3	3
S8	8	3	3
S10	9	4	4
S11	7	2	4

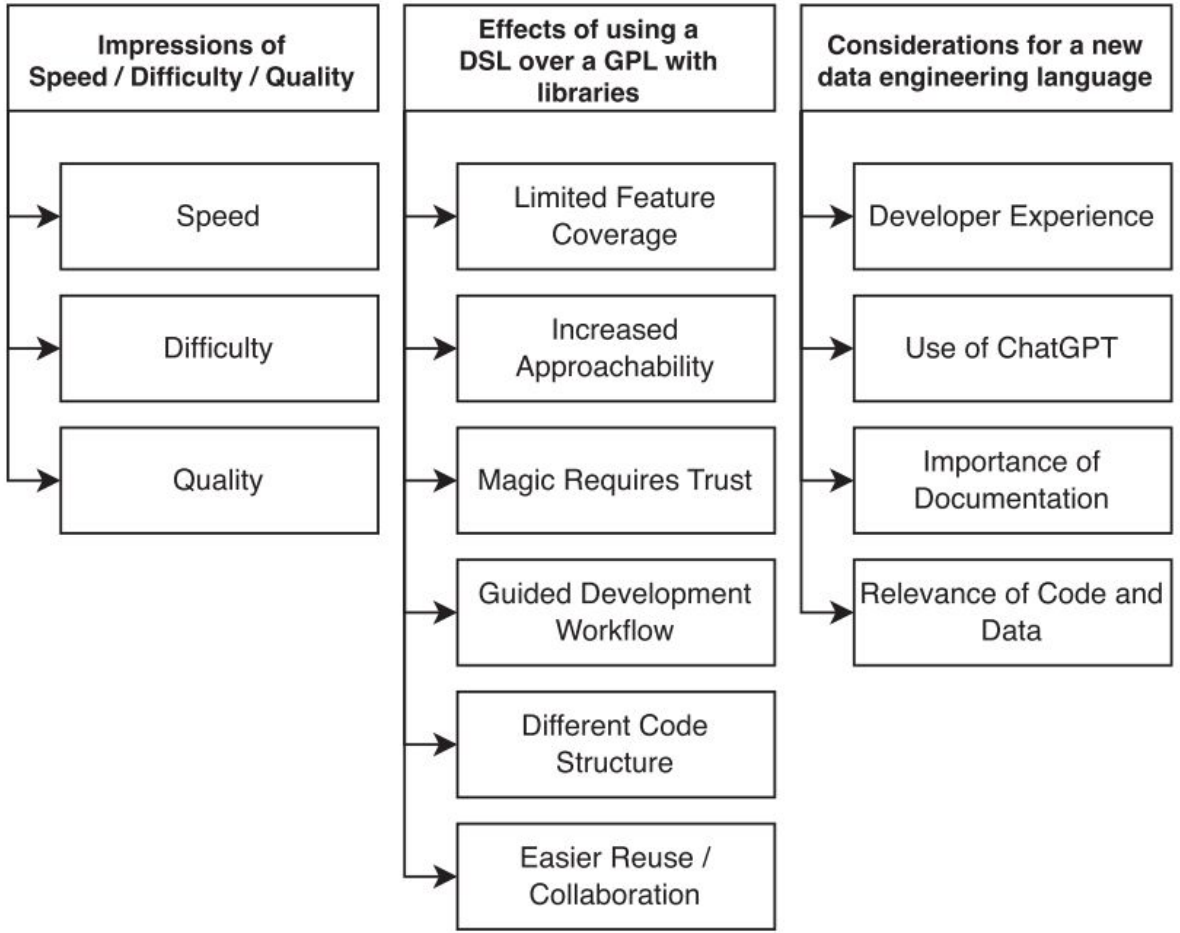
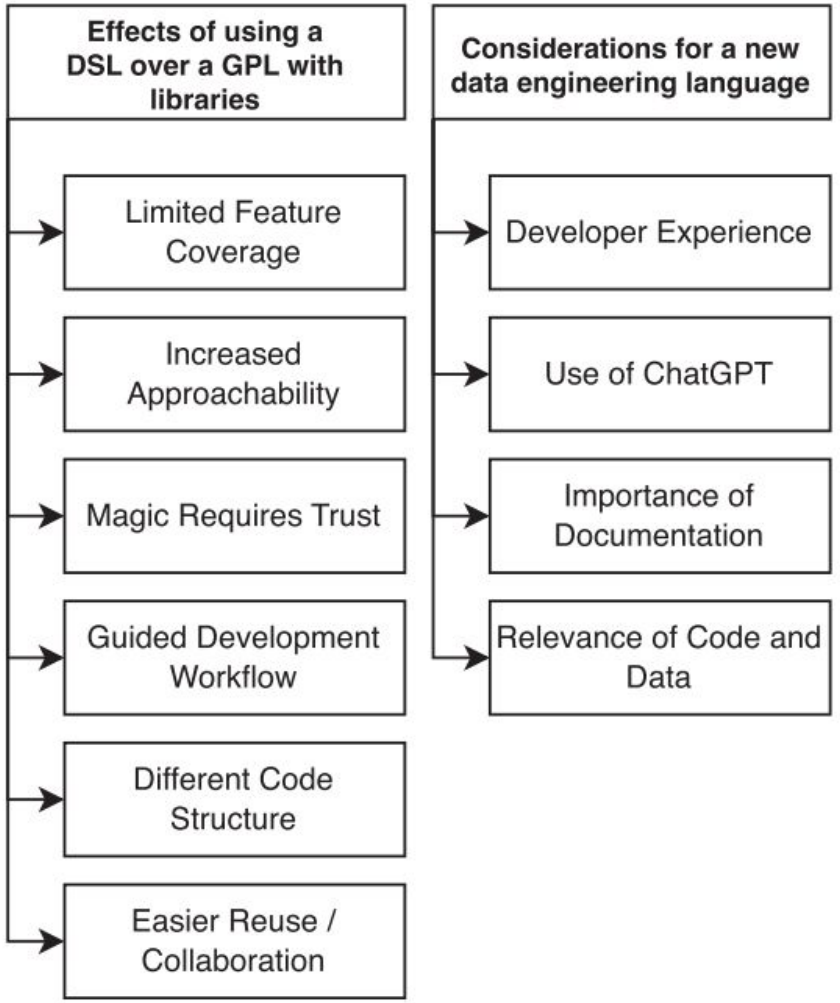


FIGURE 6 | Thematic map from thematic analysis of students' interviews and survey responses.

Validation – Effects

- Easier reuse, potentially easier collaboration
- Strongly enforced code structure guides development
- Approachability and relevant experience outside of software engineering
- Honorable mention: Use of ChatGPT (in 2023)



Paper 4 - Figures/Tables

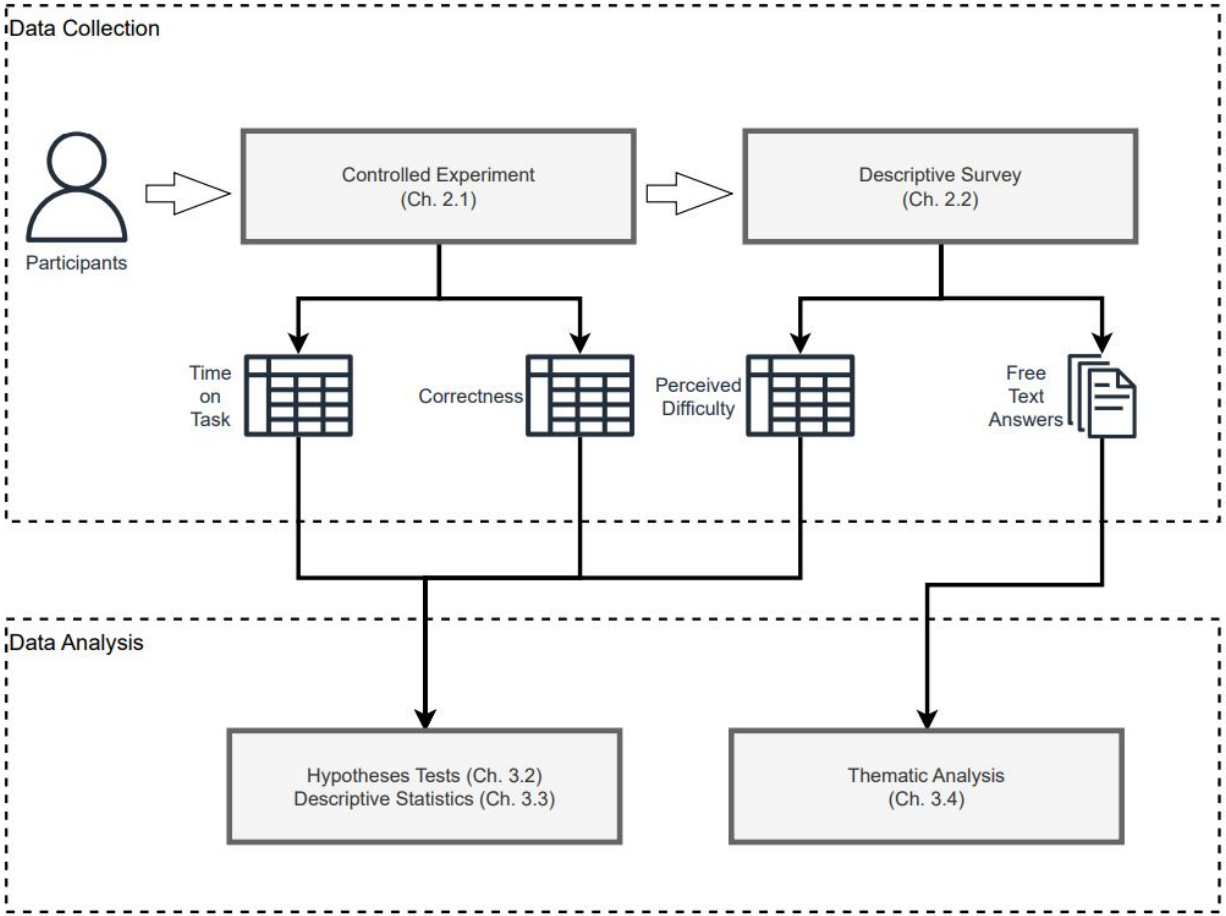


Figure 1: Overview of the mixed method research design, split into data collection and data analysis.

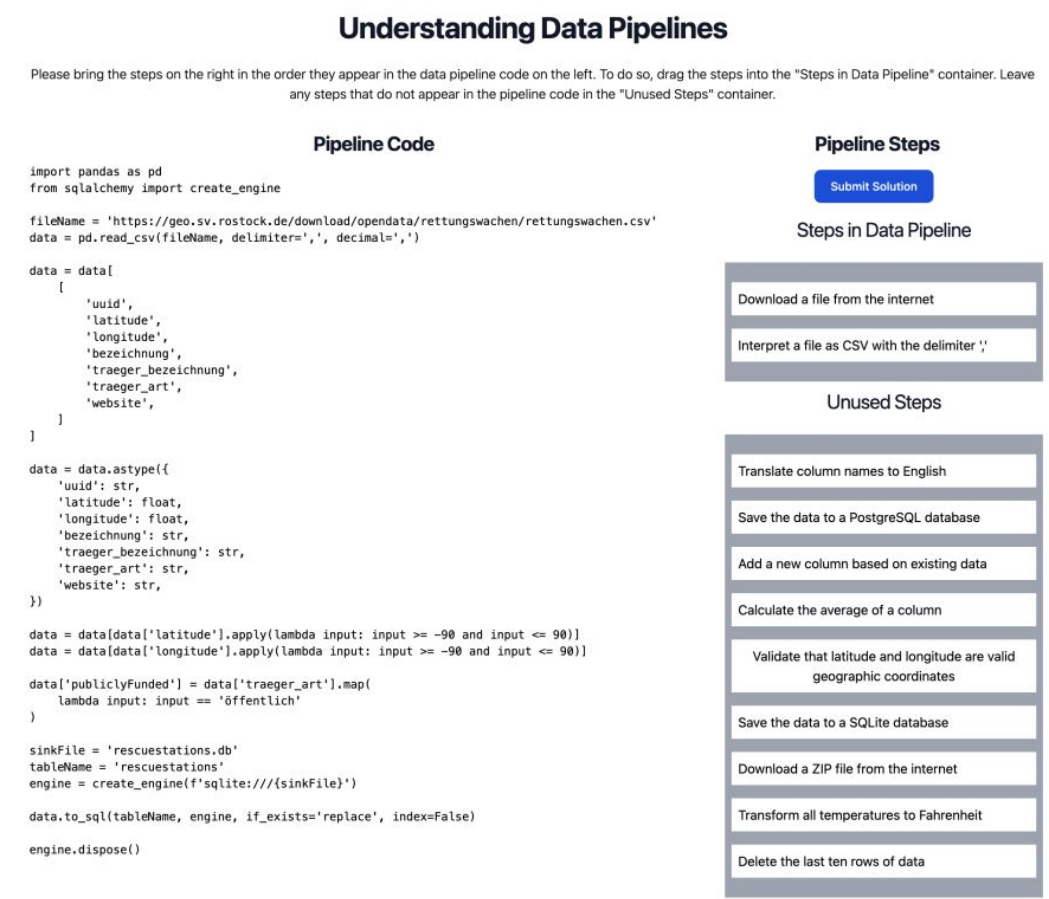


Figure 2: The experiment tool during task 2 in Python/Pandas. Pipeline source code is shown on the left, the recreation using ordered steps on the right.

Table 1: Factorial crossover design of the controlled experiment according to Vegas et al. (2016)

	Period	
Sequence	Task 1	Task 2
AB	Jayvee	Python/Pandas
BA	Python/Pandas	Jayvee

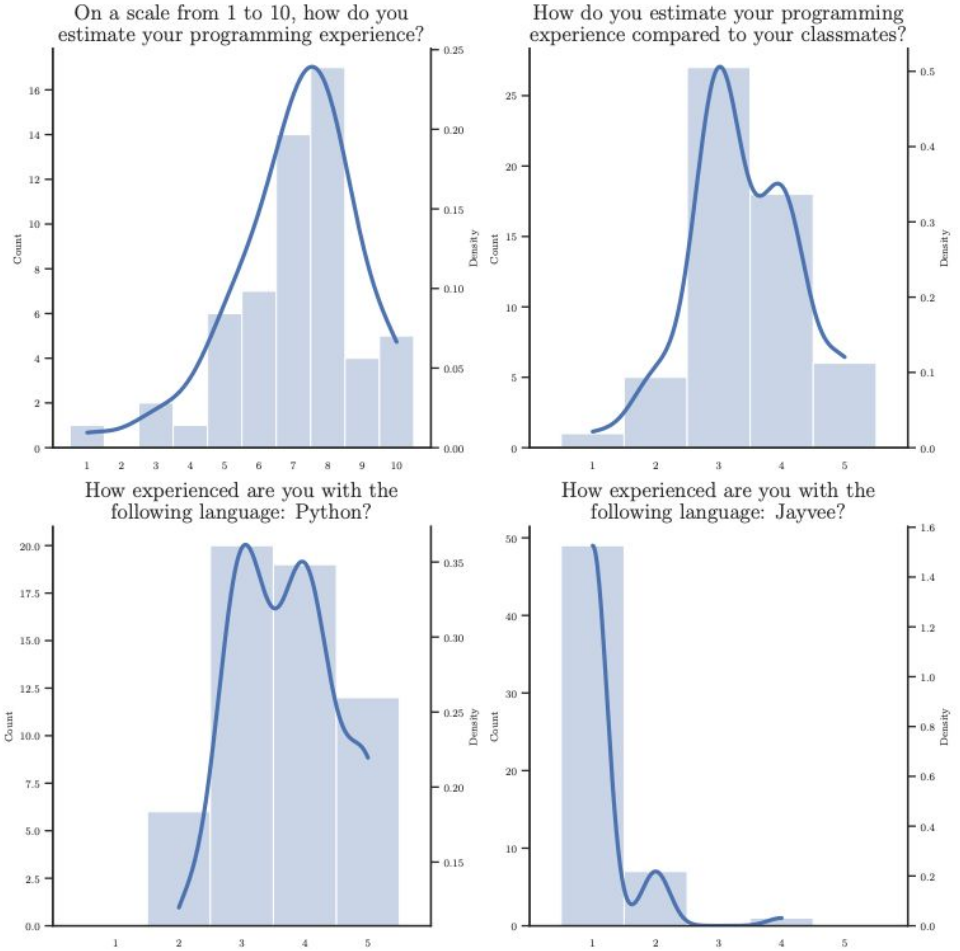


Figure 8: Previous experience of experiment participants.

Table 2: Wilcoxon signed-rank test for $H_{0,1} : time(JV) = time(PY)$

n	Mdn_{JV}	Mdn_{PY}	W-val	alternative	p-val	RBC	CLES
57	252.37	234.23	750	two-sided	.546	.093	.52

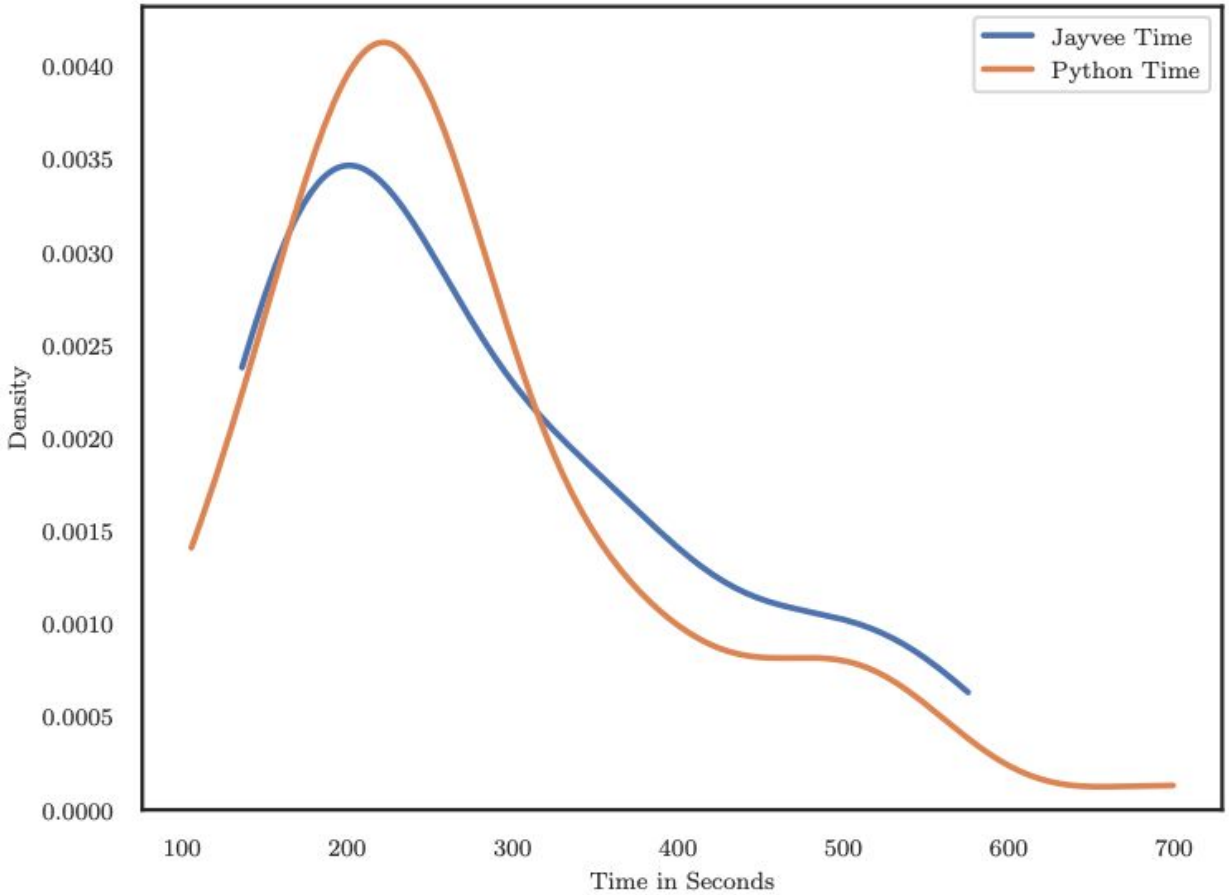


Figure 9: Kernel-density-plot of time on task for Jayvee compared to Python/Pandas.

Table 3: Wilcoxon signed-rank test for $H_{0,2}$: $correctness(JV) = correctness(PY)$

n	Mdn_{JV}	Mdn_{PY}	W-val	alternative	p-val	RBC	CLES
57	1.0	.92	183	two-sided	.002*	.55	.67

* $p \leq .05$

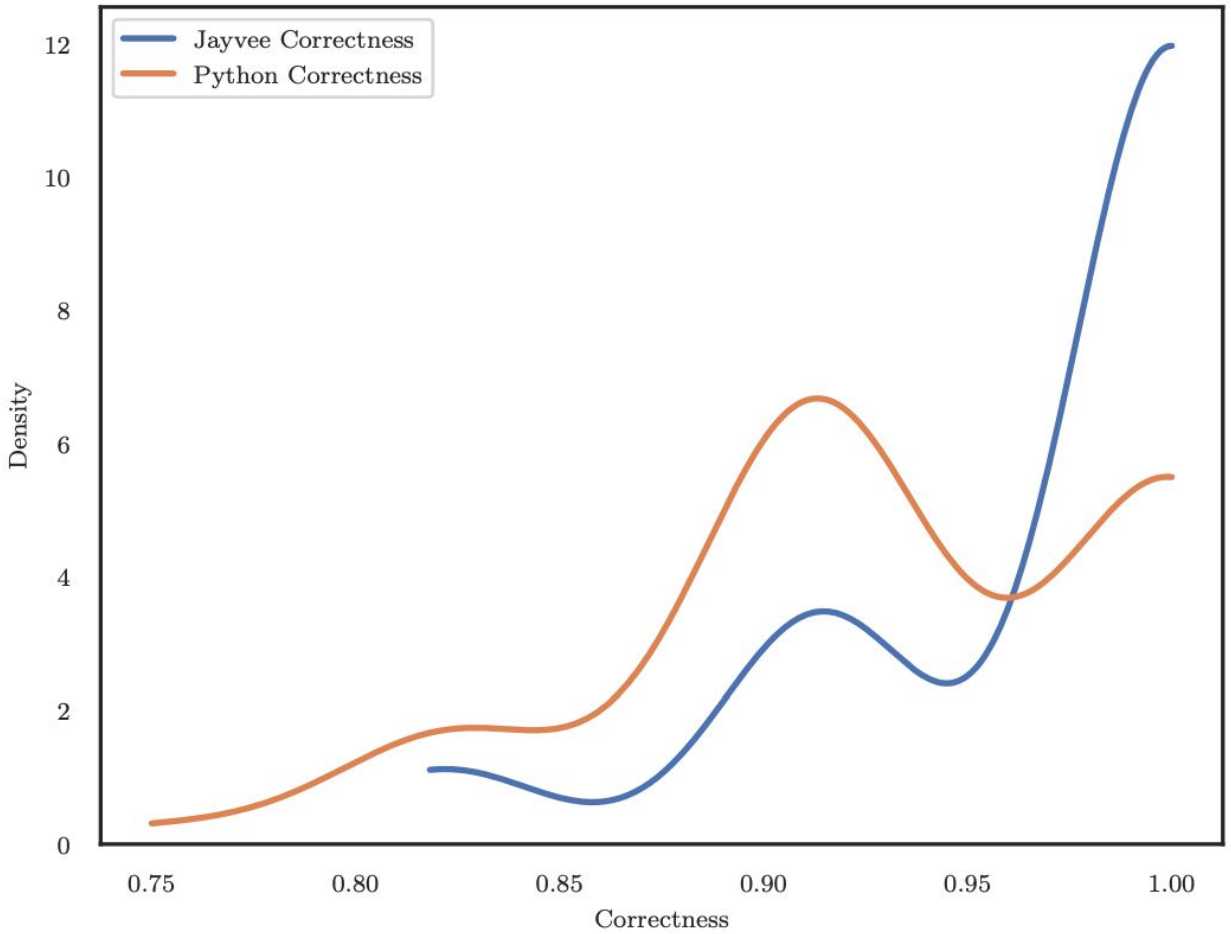


Figure 4: Kernel-density-plot of correctness of solution for Jayvee compared to Python/Pandas.

Table 4: Wilcoxon signed-rank test for perceived difficulty of using Jayvee compared to Python/Pandas, $H_{0,3}$: $difficulty(JV) = difficulty(PY)$.

n	Mdn_{JV}	Mdn_{PY}	W-val	alternative	p-val	RBC	CLES
56	2.0	2.0	380.5	two-sided	.153	-.23	.41

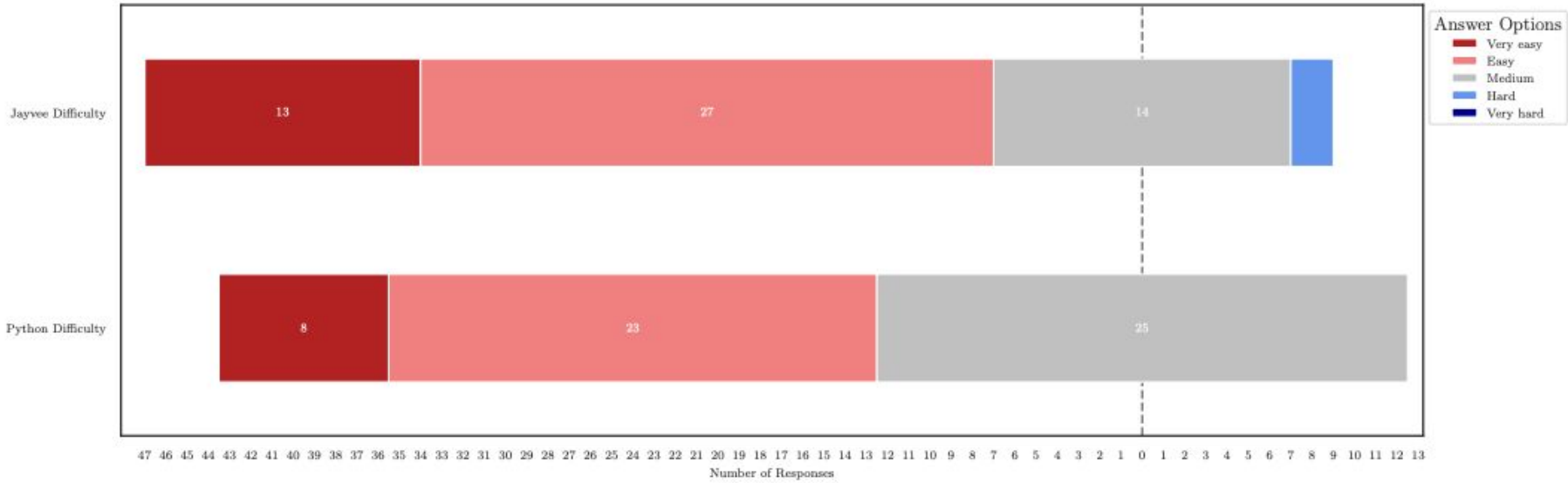
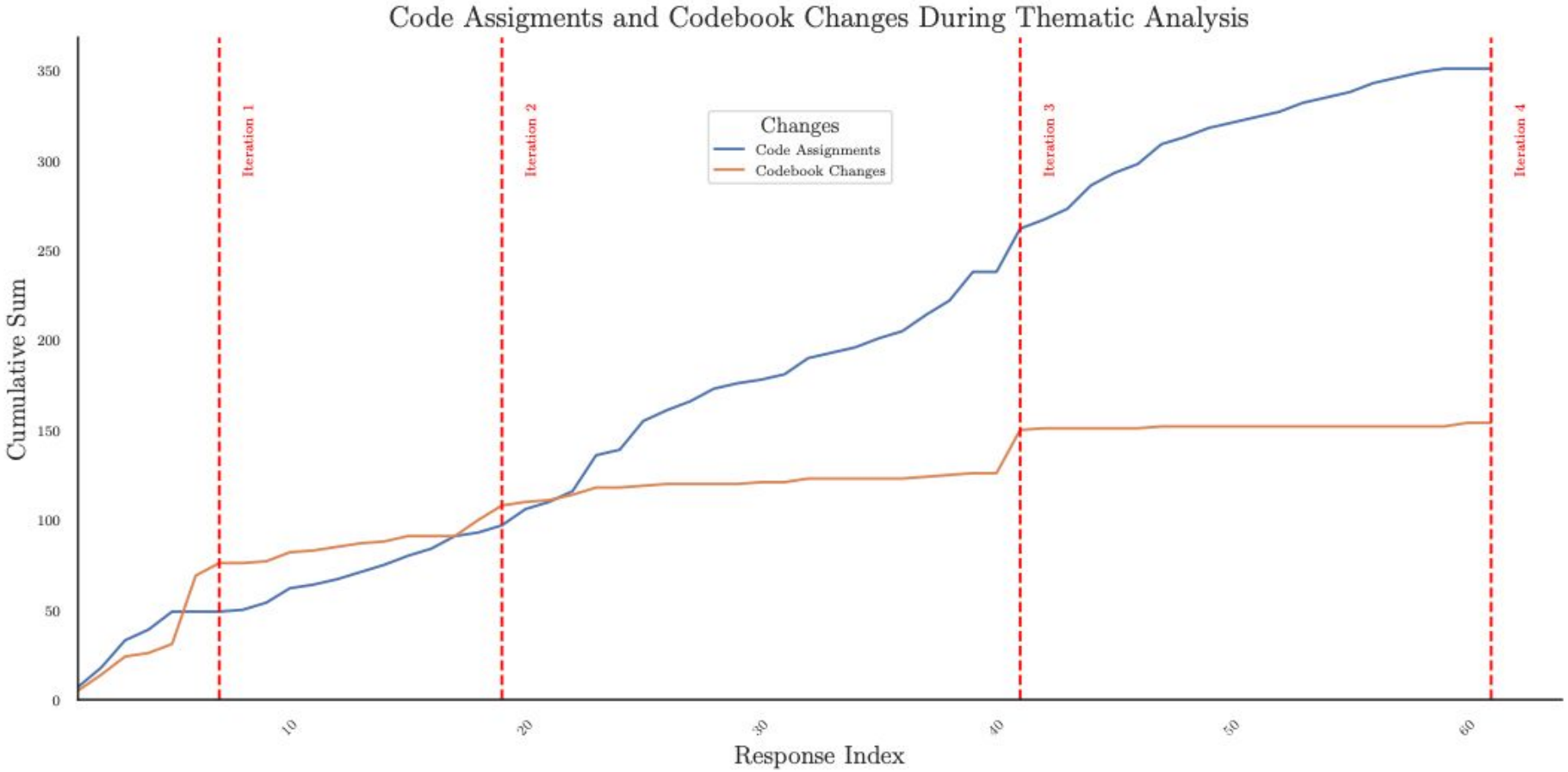


Figure 10: Diverging stacked bar charts according to Robbins et al. (2011) and Heiberger and Robbins (2014) for perceived difficulty of using Jayvee compared to Python/Pandas.*

* One outlier participant (S25) considered using Jayvee hard (and Python/Pandas easy) due to their lack of previous experience with Jayvee and did not provide more details, writing: "(Jayvee) is new so I think it was not easy to understand or read."


Code Assigments and Codebook Changes During Thematic Analysis
Changes
Code Assignments
Codebook Changes
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Cumulative Sum
Response Index
0
50
100
150
200
250
300
350
10
20
30
40
50
60

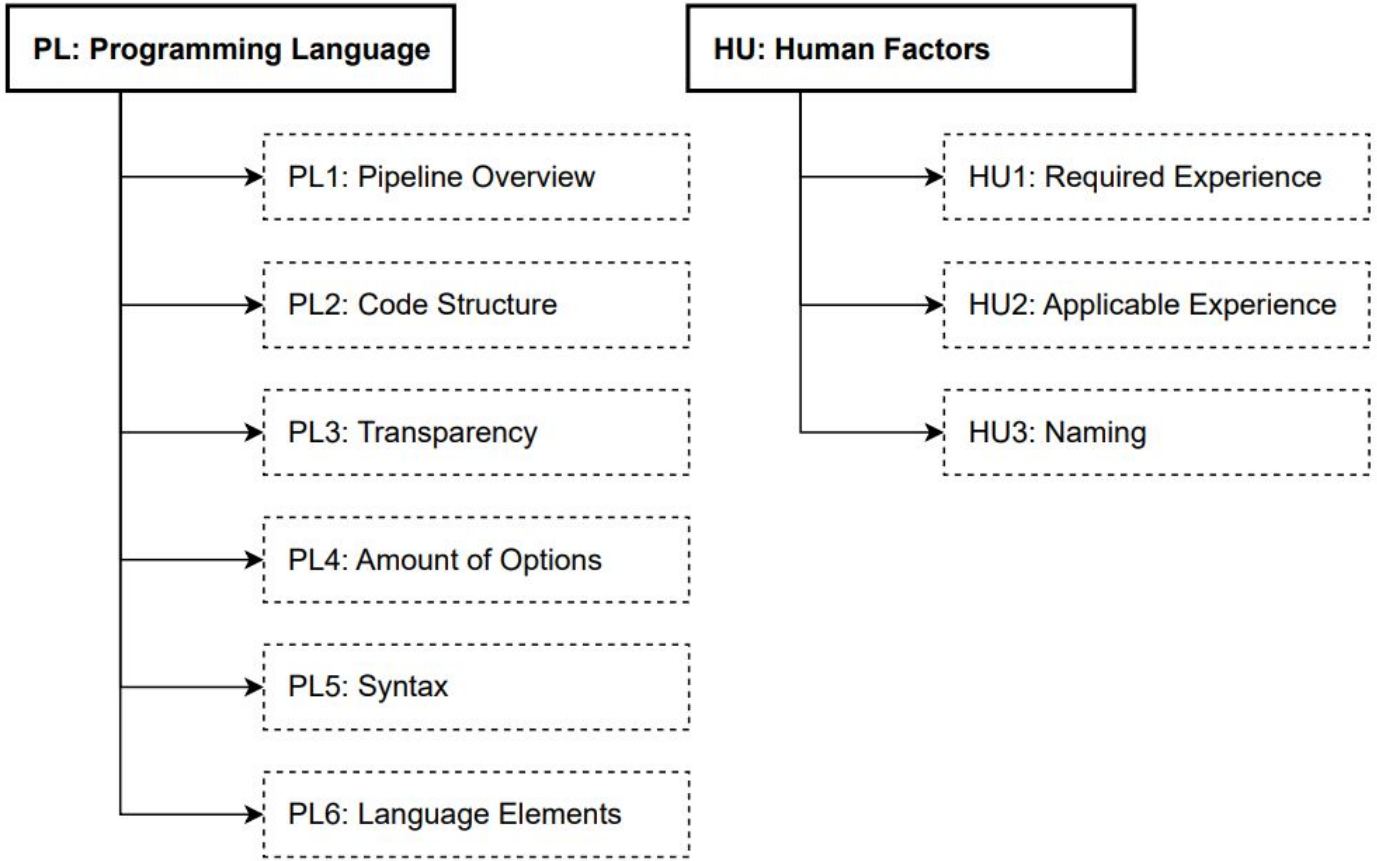


Figure 6: Overview of the codebook with two categories of themes, one related to the programming language directly and additional human factors.

Paper 5 - Figures/Tables

Table 1: Task descriptions.

Task	Description
1	Understand code that selects complete rows
2	Understand code that selects connected cells, no complete rows/columns
3	Understand code that selects complete columns
4	Understand code that selects connected cells, no complete rows/columns
5	Write code to select complete rows
6	Write code to select connected cells, no complete rows/columns
7	Write code to select complete columns
8	Write code to select connected cells, no complete rows/columns

Table 2: Sequences and intervention assignment for code creation.

Sequence	Period			
	Task 1	Task 2	Task 3	Task 4
AB	Spreadsheet	Numeric	Spreadsheet	Numeric
BA	Numeric	Spreadsheet	Numeric	Spreadsheet

Table 3: Sequences and intervention assignment for program comprehension.

Sequence	Period			
	Task 5	Task 6	Task 7	Task 8
AB	Spreadsheet	Numeric	Spreadsheet	Numeric
BA	Numeric	Spreadsheet	Numeric	Spreadsheet

Task: Write Cell Selection Code

1. Look at data

Please look at the data and notice the subset of cells that are highlighted in blue.

Spain	1	1	2	57	24	13	57619	37.417	-6.005
Denma	1	2	2	51	26	20	38052	55.703	12.572
Hungar	1	1	2	40	30	9	65014	47.503	19.098
Finland	0	2	3	54	29	4	59960	59.973	30.221
Germar	2	2	2	79	20	14	75000	48.219	11.625
France	3	3	3	70	24	5	54231	44.438	26.152
Netherl	0	2	2	36	31	9	65014	47.503	19.098
Russia	1	4	2	69	18	19	38052	55.703	12.572
Sweder	1	2	1	53	22	8	51824	55.826	-4.252
Türkiye	0	2	4	61	27	12	68700	40.430	49.920

2. Complete Jayvee code to select cells

Please complete the code to select the subset of data that is highlighted on the left.

```
// Jayvee code
// Other blocks and pipeline definition...

block DataSelector oftype CellRangeSelector {
  select: range                    ;
}
```

3. Submit Solution

Enter a solution first

Figure 1: Code creation task in the experiment tool.

Task: Understand Cell Selection

1. Read Python code to select cells

Please read the code and understand what cells it will select.

```
// Python code
// Imports and pipeline definition...

df = pd.read_csv('./data.csv')

df.iloc[6:10, 0:3]
```

2. Highlight data

Highlight the subset of data that the code selects with blue using clicks or click and dragging.

Australi	7.141	6.973	1.854	1.461	0.692	0.756	0.225	0.323	1.745
Tunisia	4.505	4.338	1.306	0.955	0.579	0.254	0.024	0.018	1.285
Netherl	7.383	7.256	1.901	1.462	0.706	0.725	0.247	0.372	1.906
Ecuado	5.85	5.599	1.315	1.151	0.64	0.606	0.087	0.078	1.846
Gabon	5.243	4.969	1.403	1.038	0.344	0.516	0.045	0.1	1.66
Kosovo	6.667	6.455	1.364	1.277	0.599	0.739	0.254	0.073	2.255
Serbia	6.522	6.3	1.538	1.391	0.585	0.663	0.2	0.101	1.932
South Africa	5.549	5.295	1.389	1.369	0.322	0.537	0.078	0.034	1.693
Zambia	3.636	3.368	0.899	0.809	0.264	0.727	0.168	0.109	0.526
Moldov	5.93	5.702	1.385	1.277	0.542	0.695	0.077	0.044	1.795

3. Submit Solution

Submit Solution

Figure 2: Program comprehension task in the experiment tool.

Table 4: Wilcoxon signed-rank test results for program comprehension.

	n	W-val	p-val	RBC	CLES
H_{1a}	84	1459.0	0.146597	-0.182633	0.464427
H_{1b}	83	1230.0	0.018937	0.29432	0.541951

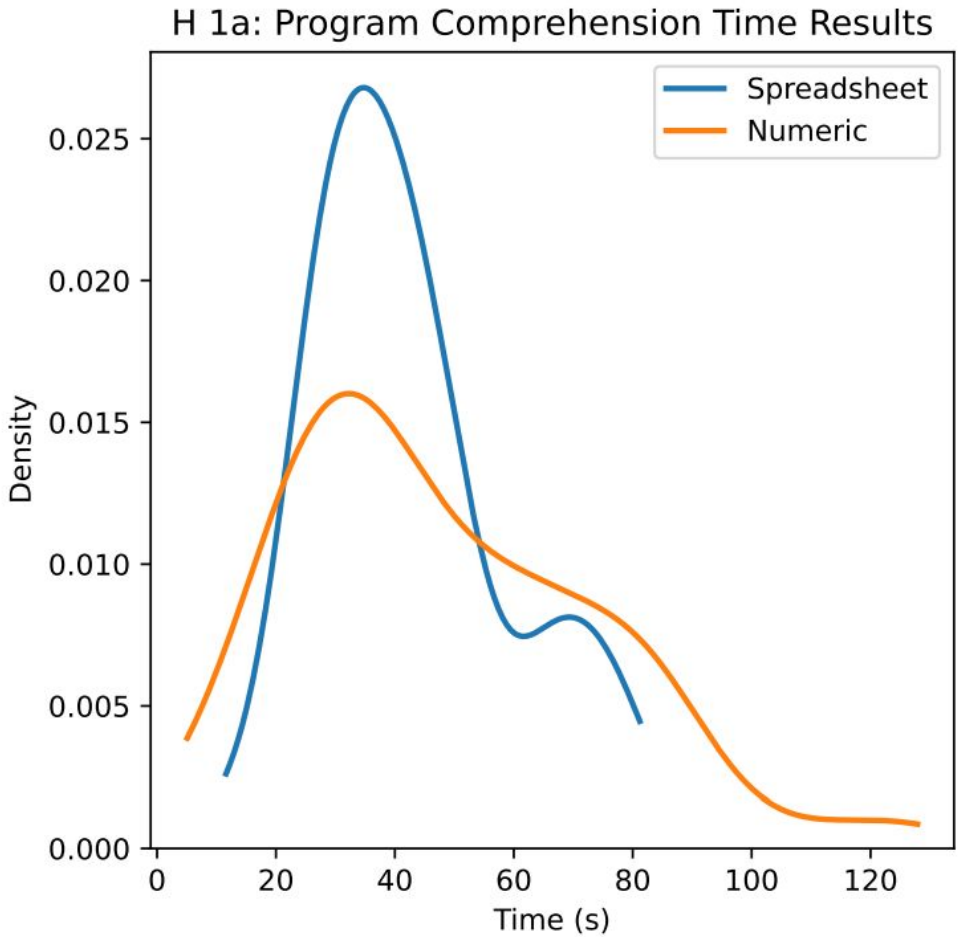


Figure 3: Kernel density plot for the results of H_{1a}, time, program comprehension

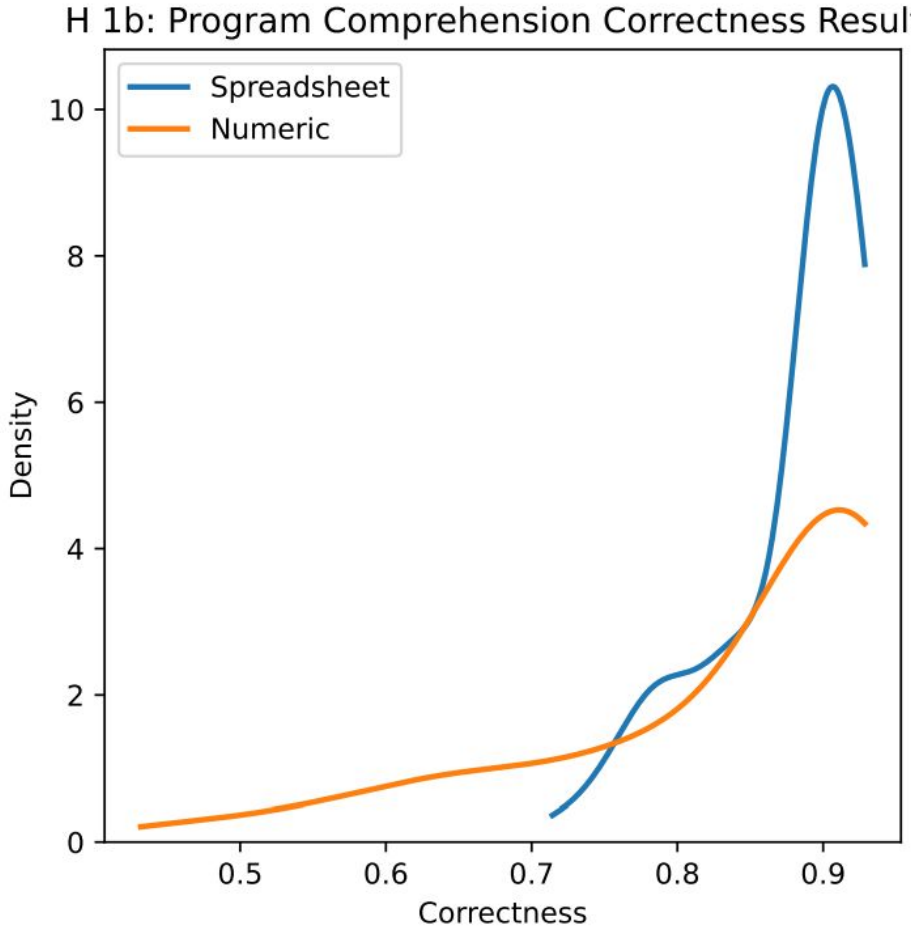


Figure 4: Kernel density plot for the results of H_{1b}, correctness, program comprehension

Table 5: Wilcoxon signed-rank test results for code creation.

	n	W-val	p-val	RBC	CLES
H_{2a}	84	163.0	4.775870e-13	-0.908683	0.160289
H_{2b}	93	463.5	0.000003	0.637324	0.658053

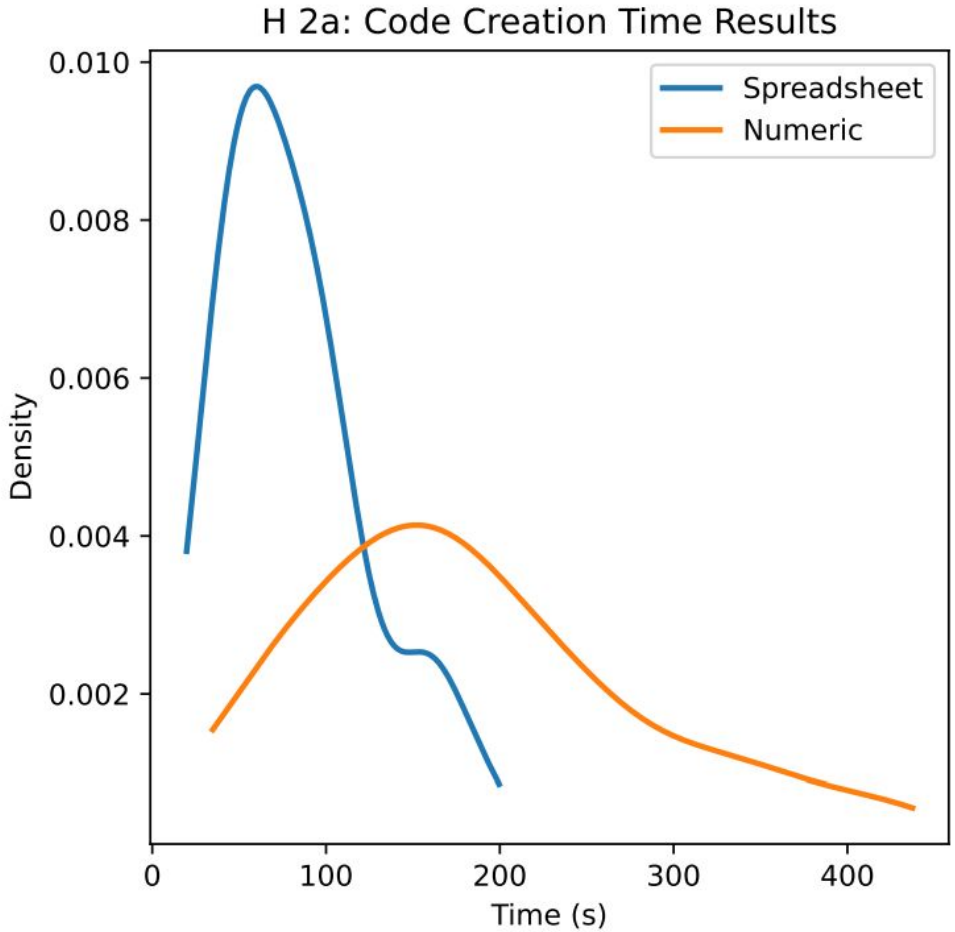


Figure 5: Kernel density plot for the results of H_{2a}, time, code creation

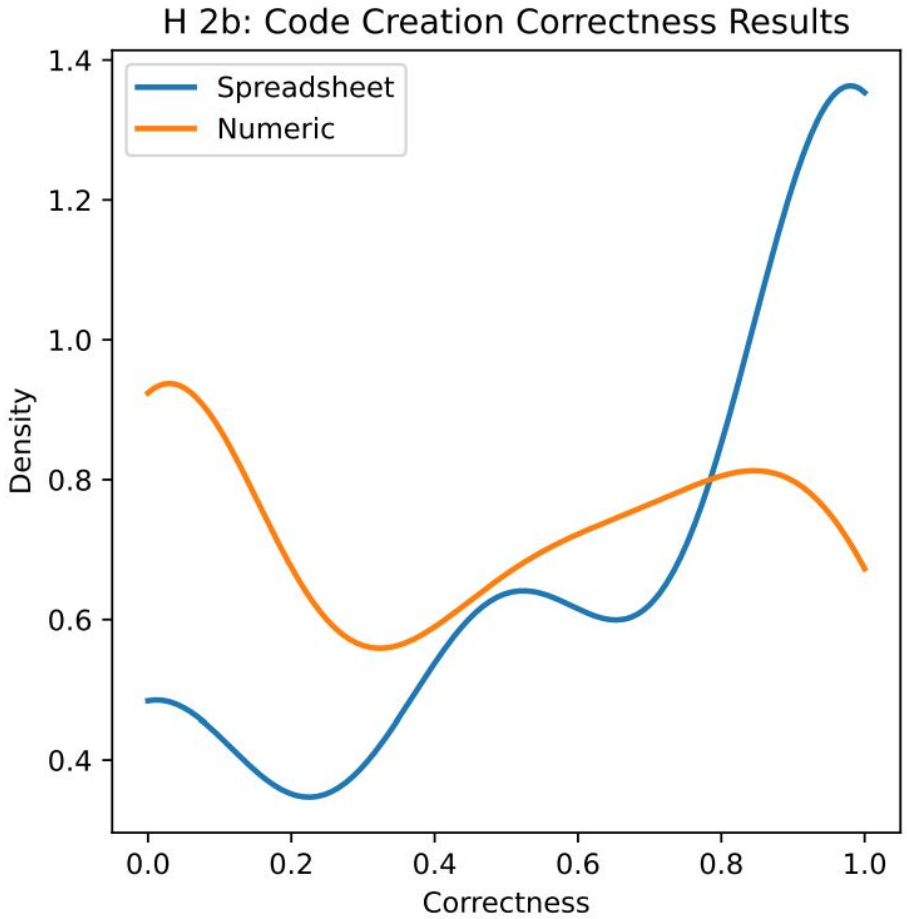


Figure 6: Kernel density plot for the results of H_{2b}, correctness, code creation