Modeling Sampling Workflows for Code Repositories

MSR 2026

Romain Lefeuvre
University of Rennes
France

Maïwenn Le goasteller
University of Rennes
France

Jessie Galasso
Mcgill University
Canada

Quentin Perez
INSA Rennes
France

Benoit Combemale
Inria
France

Houari Sahraoui
DIRO, Université de Montréal
Canada

Mining Software Repository studies

- Research question(s) on a **population of software artifacts** of interest
- Different kinds of artifacts: source code, binaries, etc.
- Often not feasible to study the entire population
- “Generalizability crisis” in software engineering [1]

Mining Software Repository studies

- Research question(s) on a **population of software artifacts** of interest
- Different kinds of artifacts: source code, binaries, etc.
- Often not feasible to study the entire population
- “Generalizability crisis” in software engineering [1]

Representativeness of studied sample is key for generalization

[1] Sebastian Baltes and Paul Ralph. 2022. Sampling in software engineering research: a critical review and guidelines. 27, 4 (2022)

Challenges with MSR sampling strategies

*[...] We collected projects from the **Software Heritage archive**, filtering out repositories with no commits after January 1, 2023, in order to focus on recently **active projects.** As OSS projects with a high number of contributors are underrepresented in the archive, we divided the filtered set into **two groups based on contributor count**: projects with fewer than 5 contributors and projects with 5 or more. We **randomly sampled 10,000 projects from each group**. The final dataset comprises 20,000 repositories. [...]*

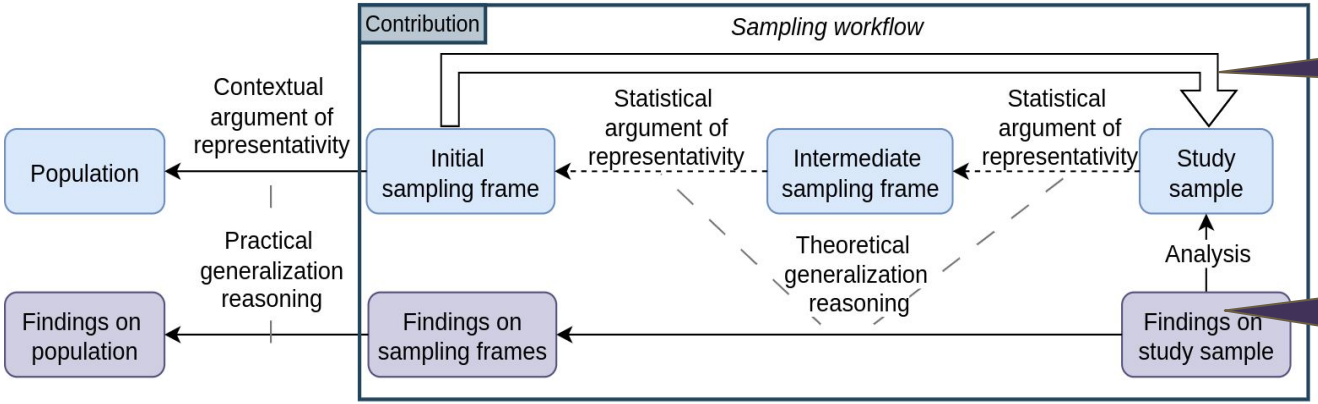
Running example: fictitious MSR study textual sampling strategy

Is this description sufficiently explicit ?

Is my sample representative enough to generalize my findings?

Need for a framework, that explicitly model multistage sampling strategies and provide statistical argument of representativeness

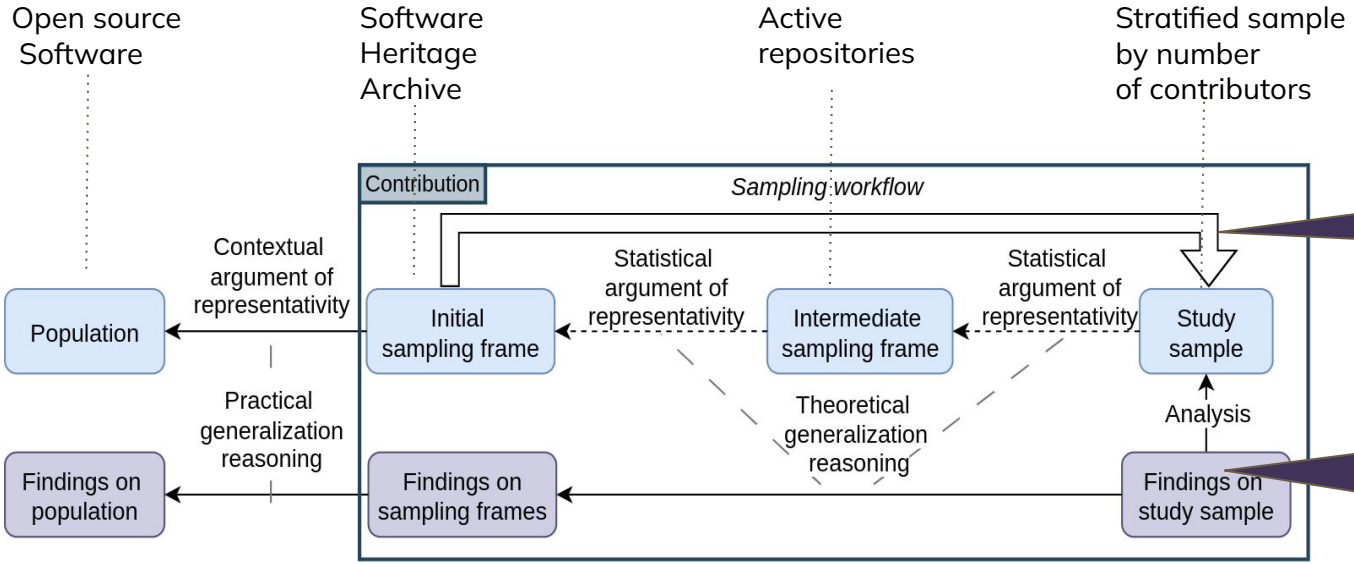
Conceptual framework



Is this description sufficiently explicit ?

Is my sample representative enough to generalize my findings?

Conceptual framework



Is this description sufficiently explicit ?

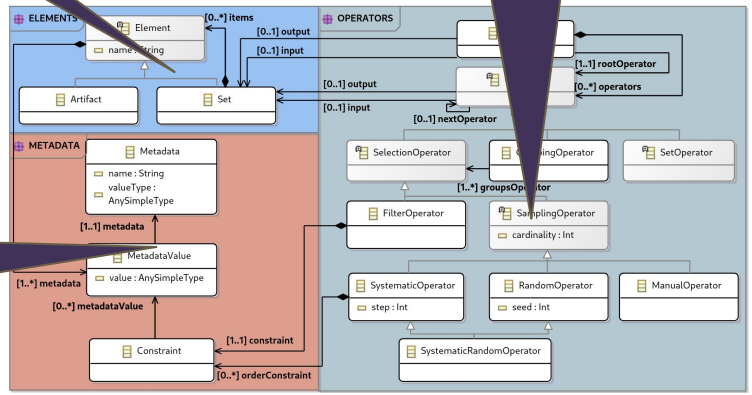
Is my sample representative enough to generalize my findings?

Approach : A Python DSL to model sampling workflows

Elements
Software Artifacts

Operators
Filtering, random ...

Metadata
Element Metadata



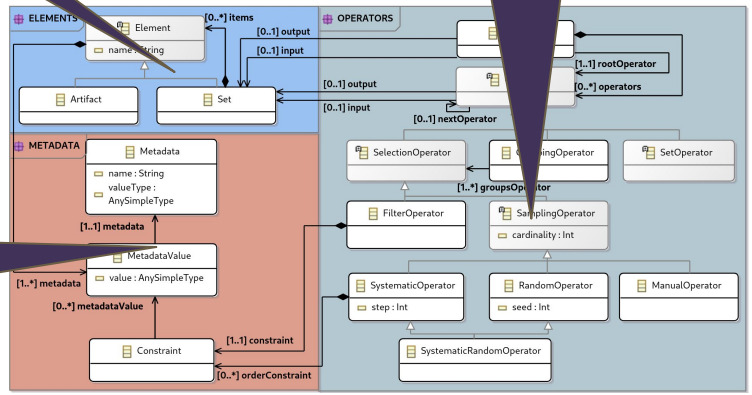
DSL metamodel

Approach : A Python DSL to model sampling workflows

Elements
Software Artifacts

Operators
Filtering, random ...

Metadata
Element Metadata



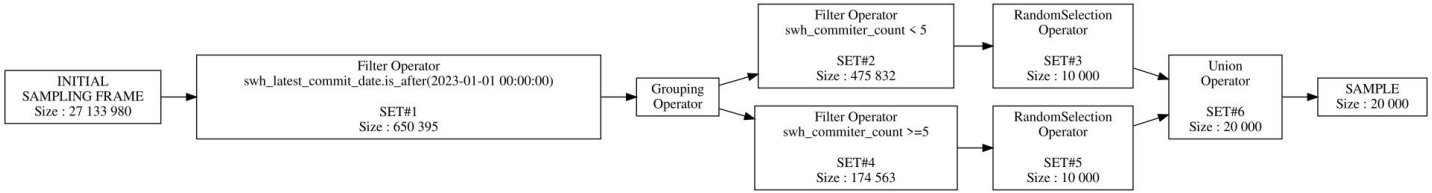
DSL metamodel

```python
(WorkflowBuilder()
.input(SwhLoader("2024-05-16-history-hosting-subgraph",url,
swh_id, commit_count, commiter_count, latest_commit_date))
    .filter_operator("latest_commit_date > datetime(2023,1,1)")
    .grouping_operator(
        # First stratum: projects with less than 5 committers
        (WorkflowBuilder()
        .filter_operator("commiter_count < 5")
        .random_selection_operator(10000)),
        # Second stratum: projects with 5 or more committers
        (WorkflowBuilder()
        .filter_operator("commiter_count >=5")
        .random_selection_operator(10000)))
    .union_operator() # Merge the two samples
    .execute())
```

Python internal DSL/ fluent API

- Formalize the sampling workflow with no ambiguity
- Executable and reproducible

Supporting representativeness reasoning with automatic statistical indicator derived from workflow analysis



Reason on the model to compute **statistical or descriptive indicator** to support **representativeness arguments**

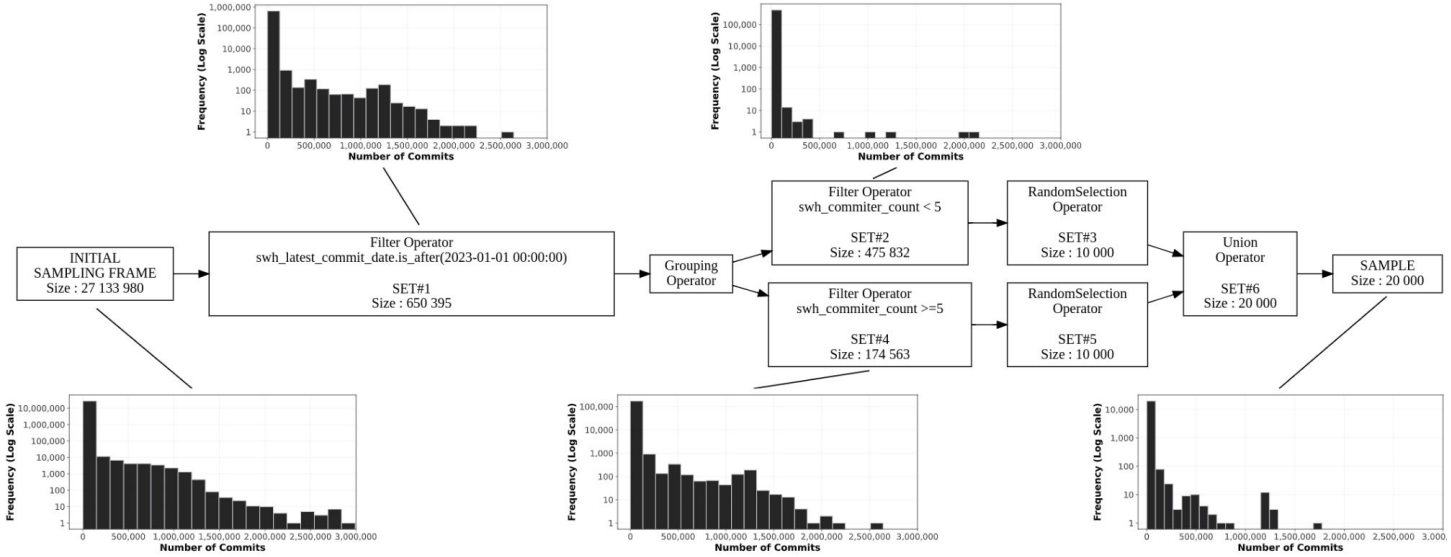
Similar distribution

Kolmogorov-Smirnov test on all intermediate test

Sample Size

Cochran's formula on random operator

Supporting representativeness reasoning with automatic statistical indicator derived from workflow analysis



Reason on the model to compute **statistical or descriptive indicator** to support **representativeness arguments**

Similar distribution

Kolmogorov-Smirnov test on all intermediate test

Sample Size

Cochran's formula on random operator

Evaluation and Conclusion

- **Contributions :** A DSL to explicitly model multistage sampling strategy
  - Automatic statistic indicator to support representativeness claim

Evaluation and Conclusion

- **Contributions :** A DSL to explicitly model multistage sampling strategy
  - Automatic statistic indicator to support representativeness claim
- **Case study on MSR papers :**
  - Validation of DSL expressivity
  - **31% present ambiguities in their textual description**
  - **88% discuss representativeness, but rarely backed by statistical indicators**

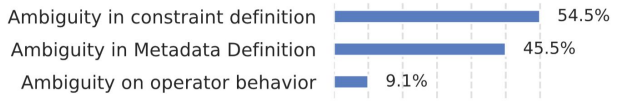


Figure 7: D4.2 Distribution of Causes (The percentages correspond to the proportion of the 11 papers with a specific ambiguity cause; a paper can have multiple ambiguity causes).

Evaluation and Conclusion

- **Contributions :** A DSL to explicitly model multistage sampling strategy
  - Automatic statistic indicator to support representativeness claim
- **Case study on MSR papers :**
  - Validation of DSL expressivity
  - **31% present ambiguities in their textual description**
  - **88% discuss representativeness, but rarely backed by statistical indicators**

Figure 7: D4.2 Distribution of Causes (The percentages correspond to the proportion of the 11 papers with a specific ambiguity cause; a paper can have multiple ambiguity causes).

- **Call for action at MSR :**
  - Make sampling strategies **explicit** and **reproducible**
  - Support generalization reasoning with statistical indicator

Source code

pip install sampling-mining-workflows-dsl

Paper

Evaluate applicability and expressiveness of the DSL

Case Study RQ3: Can the concepts defined in our DSL adequately capture the complexity of sampling strategies used in MSR studies?

- 24 out of 35 papers successfully modeled
- 11 were not entirely modeled due to ambiguities in the descriptions of sampling strategies

The concepts of our DSL enable us to capture the studied sampling strategies

Case Study RQ4: What are the causes of incomplete modeling of sampling strategies?

- Ambiguities in 11 out of 35.

"sufficient number of commits"

*"1. Accessibility, i.e., the libraries **should be popular,** widely used, and open-source, 2. Maturity, i.e., the libraries should have been **actively developed** for a considerable amount of time"*

*"[..] we mitigate the effects of potential confounds in our analysis, for example, **choosing a similar amount of small, medium, and large-sized projects for both ML and non-ML projects.**"*

The lack of explicit description prevents unambiguous interpretation and implementation of the sampling process.

Evaluate applicability and expressiveness of the DSL

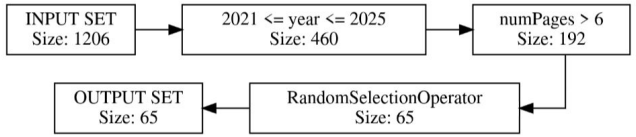
Case study on MSR sampling strategies

Ideal population : “all empirical works that apply sampling to software repository artifacts”

Initial sampling frame : “publication from MSR conference”

```
WorkflowBuilder()
    .input(CsvLoader(input_path,doi,title,year,numPages))\
    .filter_operator("2021 <= year <= 2025")\
    .filter_operator("numPages > 6")\
    .add_metadata(CsvLoader(IEEE_path,doi,ieee_keyword_list))\
    .random_selection_operator(cardinality=65,seed=4242)\
```

(a) Workflow source code



(b) Generated Workflow diagram

Sample Size

- Cochran’s formula, (95%, 0.1)
- **To randomly sample with such constraint : 64 < 65**

IEEE key word distribution

- Categorical : χ^2 goodness of fit test
- $\chi^2 = 197.3$ $p = 0.99$
- **no significant distribution difference**

IEEE key word coverage

- 100 % of coverage of most frequently used keyword in 2021-2025 MSR study

Sample representativeness

- Well-documented concern across scientific fields[1], including software engineering [2]
- Defined as the extent to which **“a sample’s properties of interest resemble those of the target population"**
- **Dimension specific**
- Representativeness can be supported with different arguments :
  - Large and random sample
  - Breadth of a sample
  - Similar distributions

[1] William Kruskal and Frederick Mosteller. 1979. Representative Sampling, II: Scientific Literature, Excluding Statistics. International Statistical Review / Revue Internationale de Statistique 47, 2 (1979), 111–127

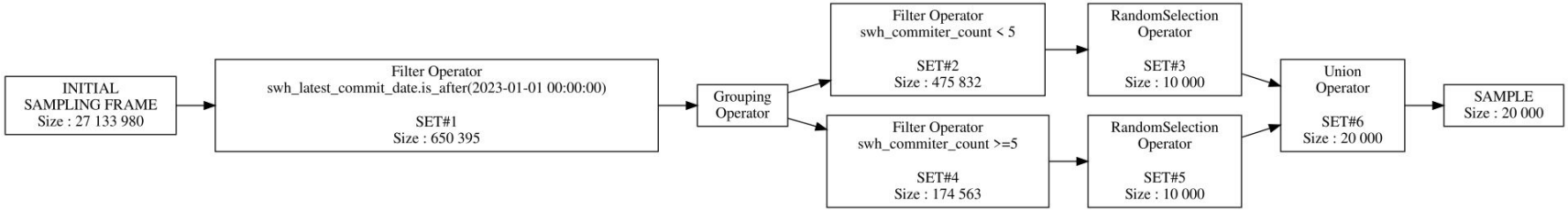
[2] Sebastian Baltes and Paul Ralph. 2022. Sampling in software engineering research: a critical review and guidelines. 27, 4 (2022), 94. doi:10.1007/s10664-021-10072-8

Modelling sampling workflow : Python internal DSL

```
(WorkflowBuilder()
.input(SwhLoader("2024-05-16-history-hosting-subgraph",url,
swh_id, commit_count, commiter_count, latest_commit_date))
    .filter_operator("latest_commit_date > datetime(2023,1,1)")
    .grouping_operator(
        # First stratum: projects with less than 5 committers
        (WorkflowBuilder()
        .filter_operator("commiter_count < 5")
        .random_selection_operator(10000)),
        # Second stratum: projects with 5 or more committers
        (WorkflowBuilder()
        .filter_operator("commiter_count >=5")
        .random_selection_operator(10000)))
    .union_operator() # Merge the two samples
    .execute())
```

- Python fluent API
- Prototype with connector to SWH

Running example workflow



Generated graphical representation of an execution of the running example workflow on 27M repositories of Software Heritage

Sampling strategies

Approach	Capsule Description
Convenience	Select items based on expediency
Purposive	Select items most useful for study's objective
Referral-chain	Select items based on relationship to existing items
Respondent-driven	Bias-mitigating variant of referral-chain
Whole frame	Select the entire sampling frame
Simple random	Select items entirely by chance
Systematic random	Select every xth item from a random start
Stratified	Select items from different groups randomly but in equal proportion
Quota	Select items from different groups purposively but in equal proportion
Cluser	Select items in stages, where each stage is a subset of the previous

Probabilistic and non probabilistic techniques.

Argument of representativeness dependent on sampling type

Sebastian Baltes and Paul Ralph. 2022. Sampling in software engineering research: a critical review and guidelines. Empirical Softw. Engg. 27, 4 (Jul 2022). https://doi.org/10.1007/s10664-021-10072-8

Challenge related to sampling approach design

- No formalism : Incomplete textual description of methodology

- Lack of probabilistic sampling
  - only 8% of analysed study use random sampling [1]
  - “Generalisability crisis” [1]

Need for a framework, that explicitly model multi-stage sampling strategies

[1] Sebastian Baltes and Paul Ralph. 2022. Sampling in software engineering research: a critical review and guidelines. Empirical Softw. Engg. 27, 4 (Jul 2022). https://doi.org/10.1007/s10664-021-10072-8