ulm university universität ulm

HUMBOLDT-UNIVERSITÄT ZU BERLIN

Simulating the Evolution of Software Variants with VEVOS

EASE 2022

Gothenburg, Sweden

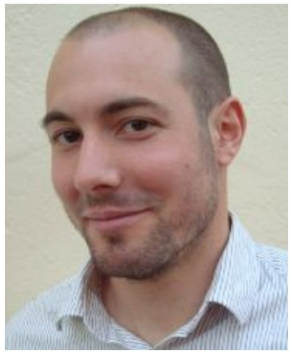
Alexander Schultheiß

b UNIVERSITÄT BERN

Paul Maximilian Bittner

Sascha El-Sharkawy

Thomas Thüm



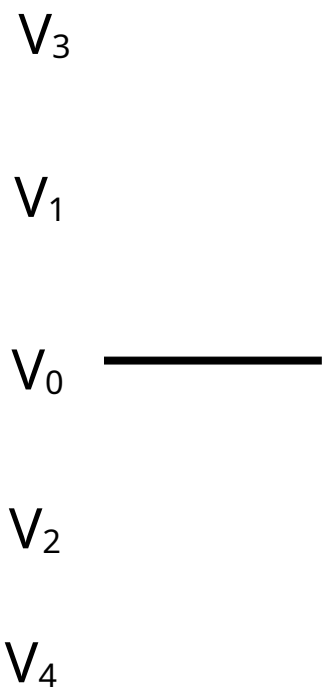
Timo Kehrer

Clone-and-own and variability

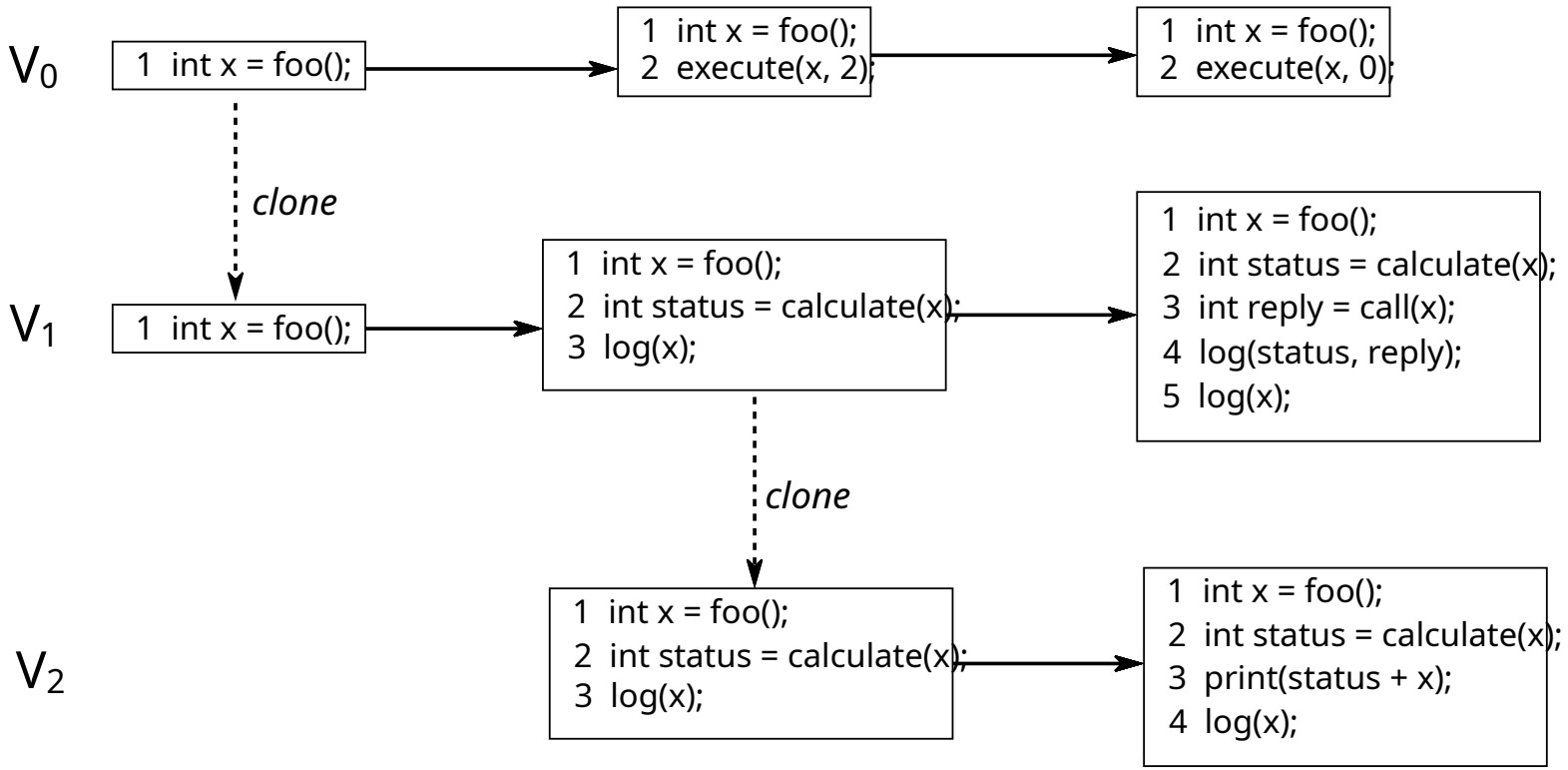
V_0

```
1  int x = foo();
```

Clone-and-own in growing projects



Problem: No explicit knowledge about features



What items are on our data wishlist?

1. Source code of variants

V_0

```
1  int x = foo();
2  execute(x, 0);
```

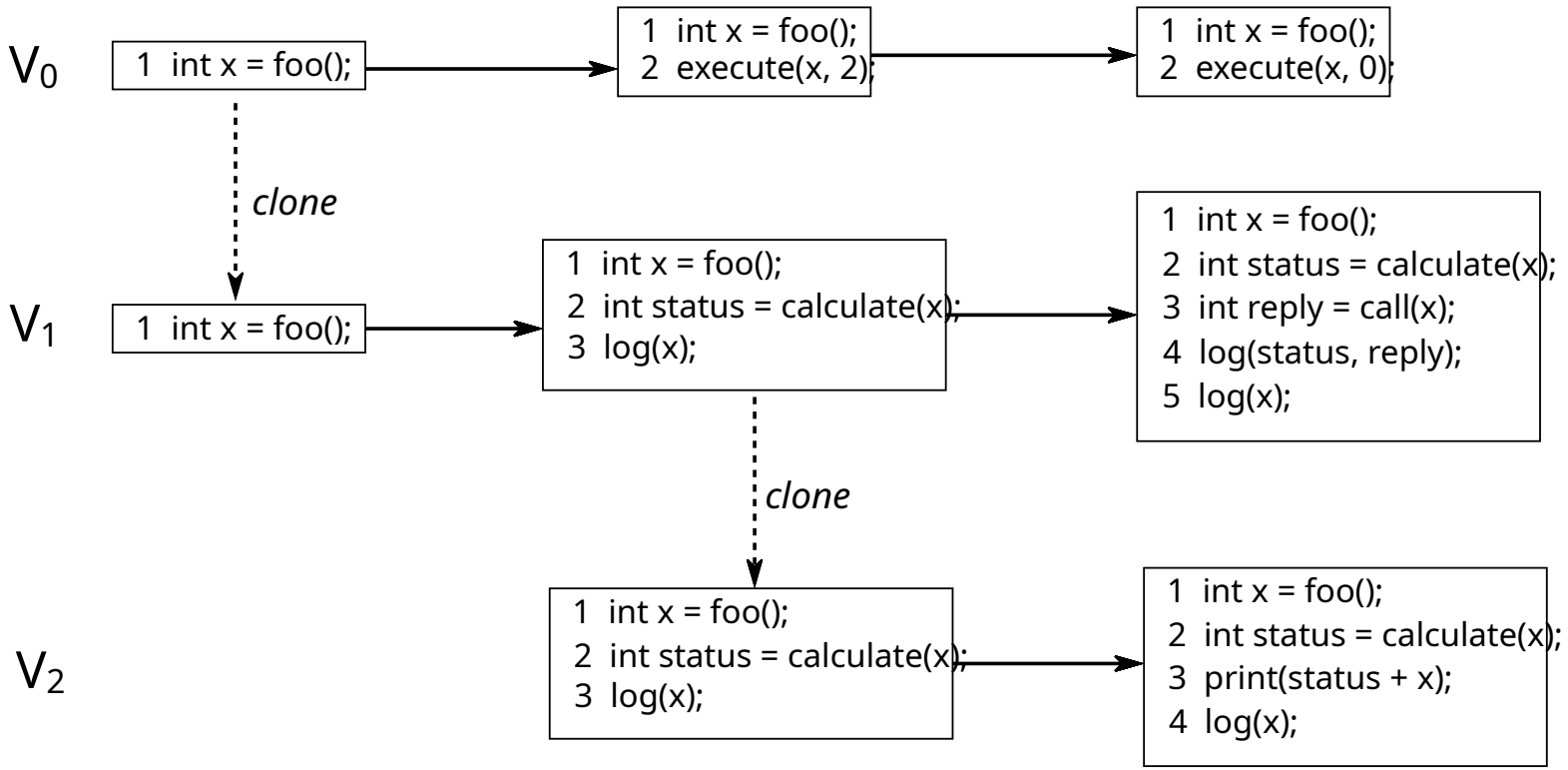
V_1

```
1  int x = foo();
2  int status = calculate(x);
3  int reply = call(x);
4  log(status, reply);
5  log(x);
```

V_2

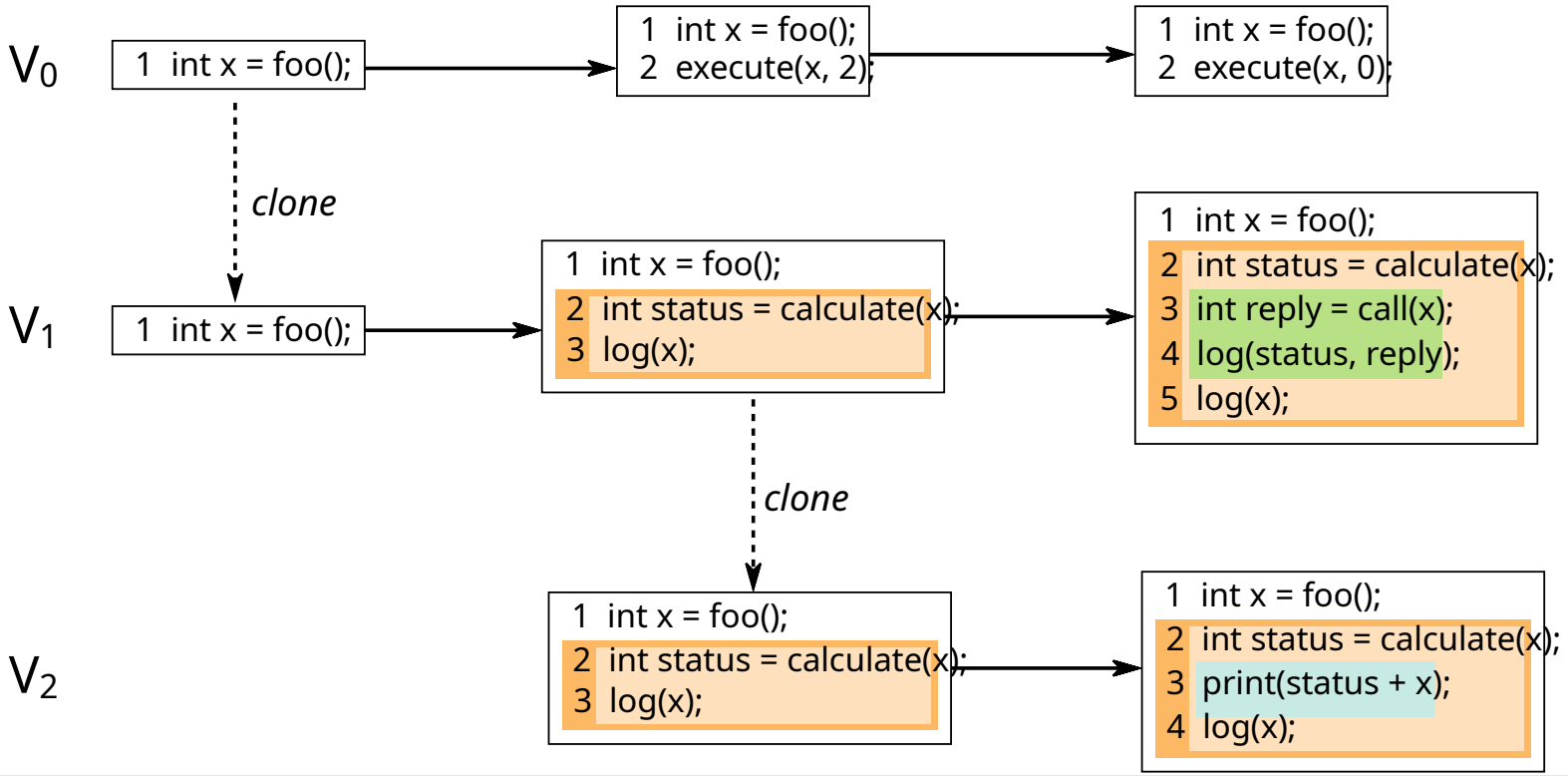
```
1  int x = foo();
2  int status = calculate(x);
3  print(status + x);
4  log(x);
```

2. Evolution of variants

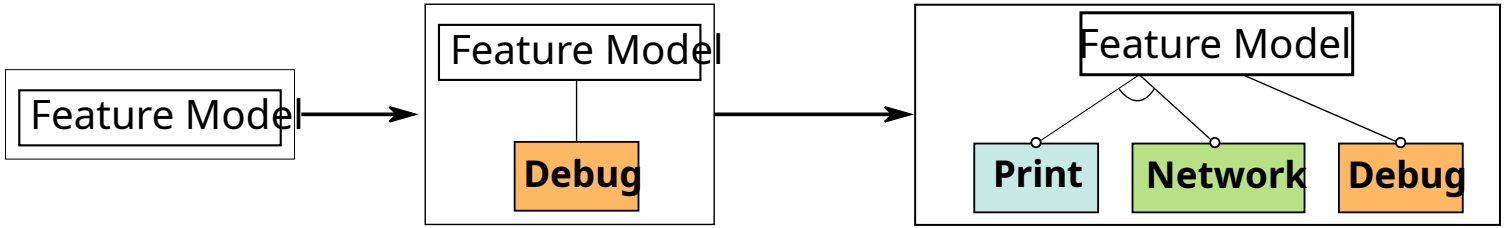


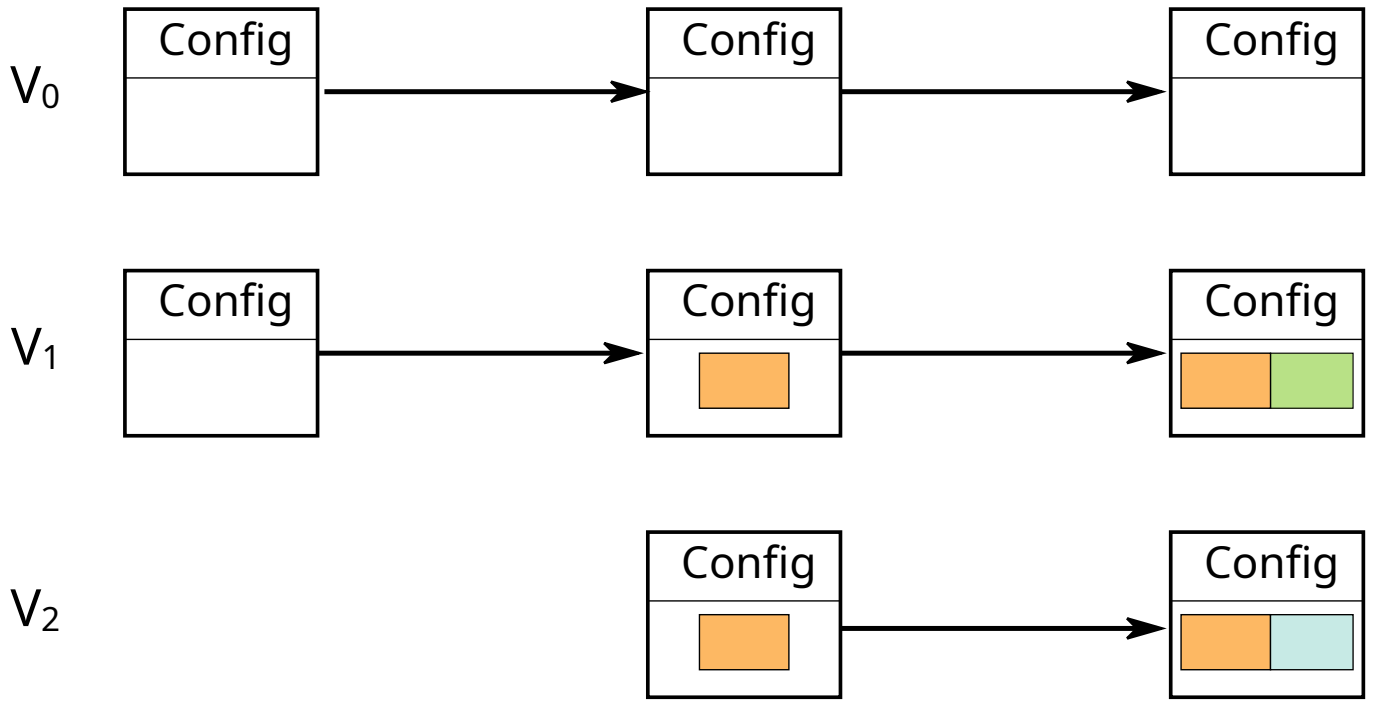
4. Evolution of the feature model

3. Feature mappings of source code

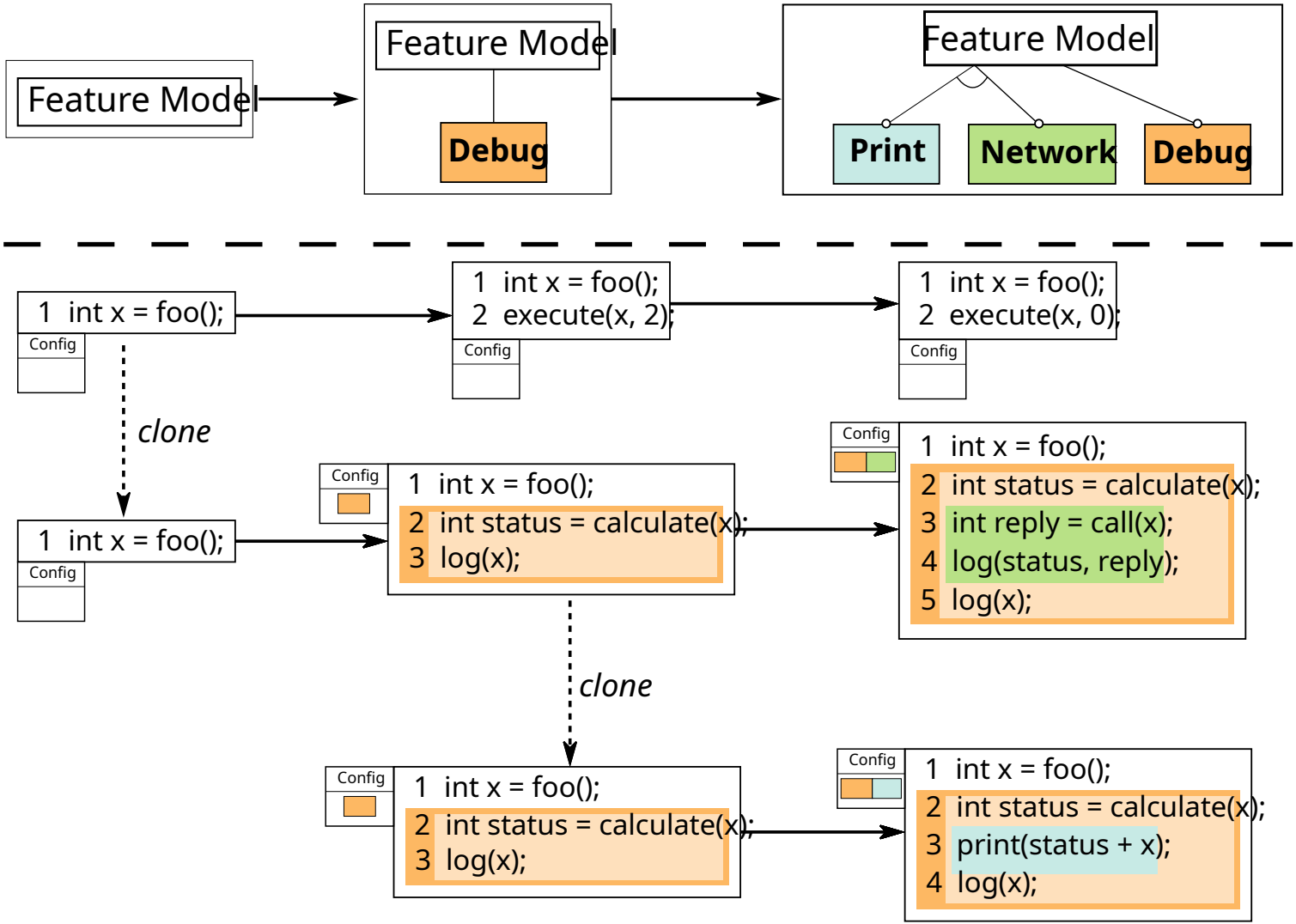


5. Variant configurations

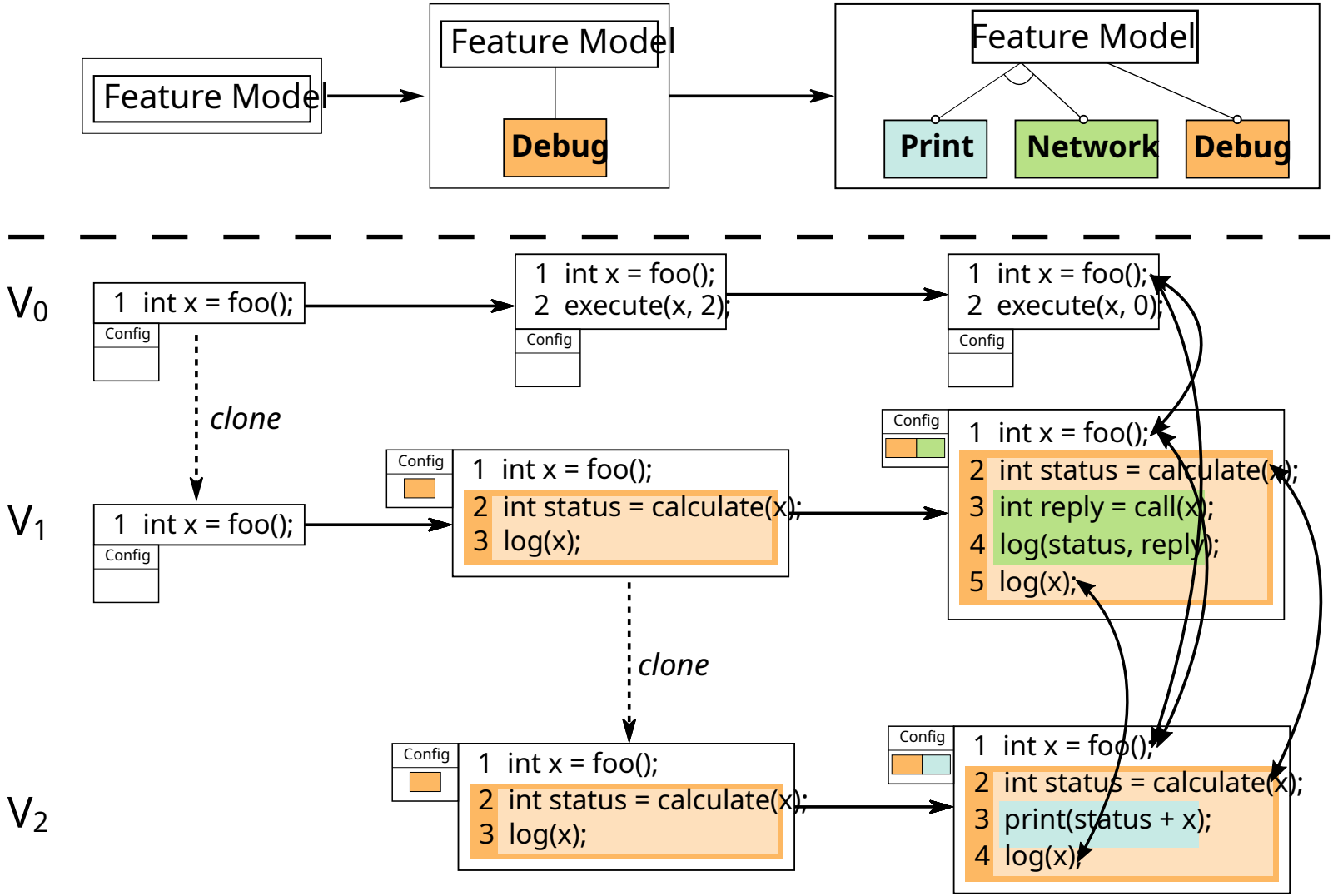




5. Variant configurations

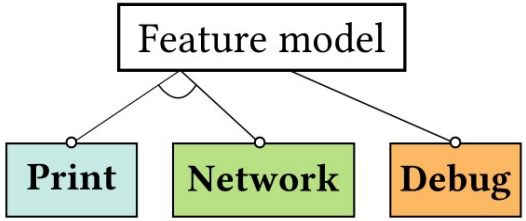


6. Code matching



Where can we get this data?

Software product lines

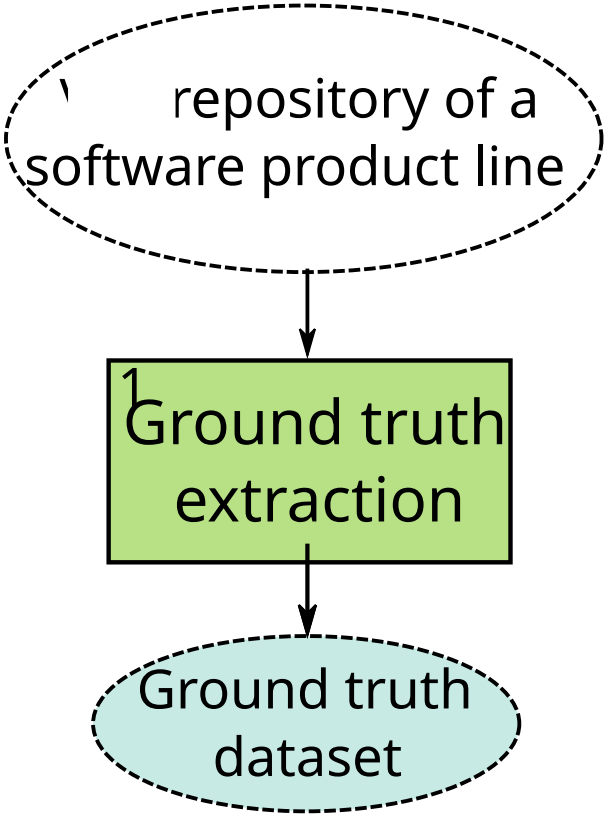


VEVOS

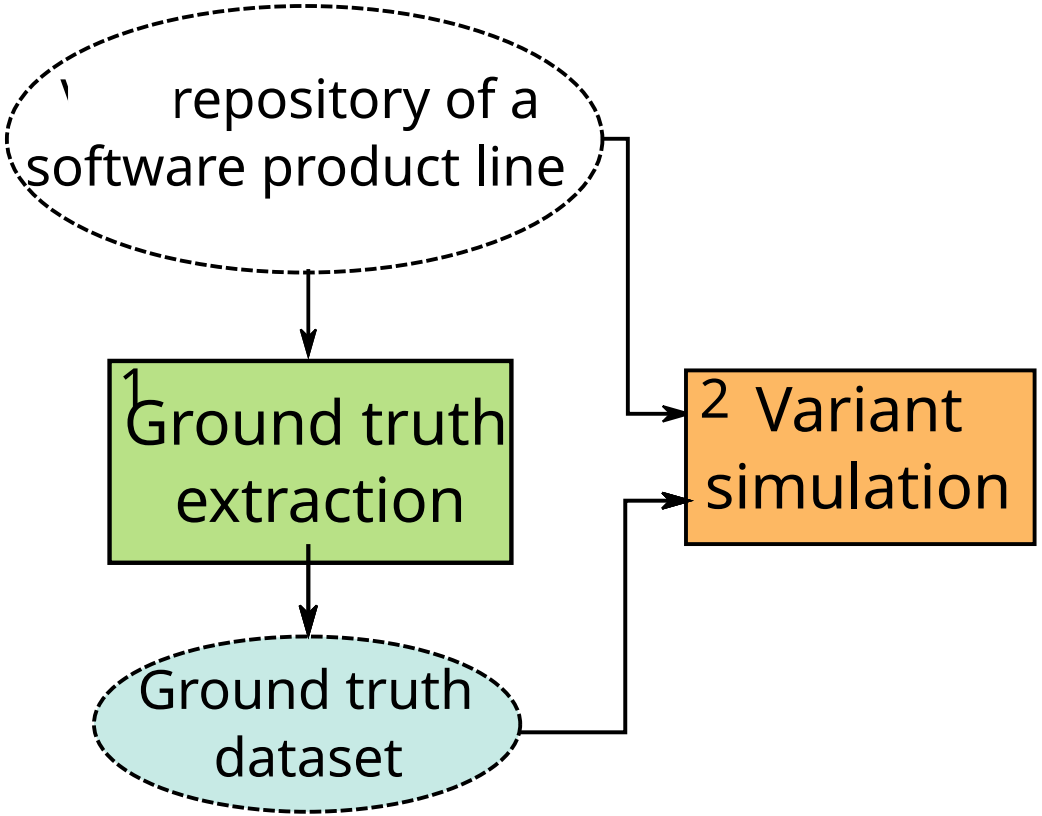
VEVOS

repository of a
software product line

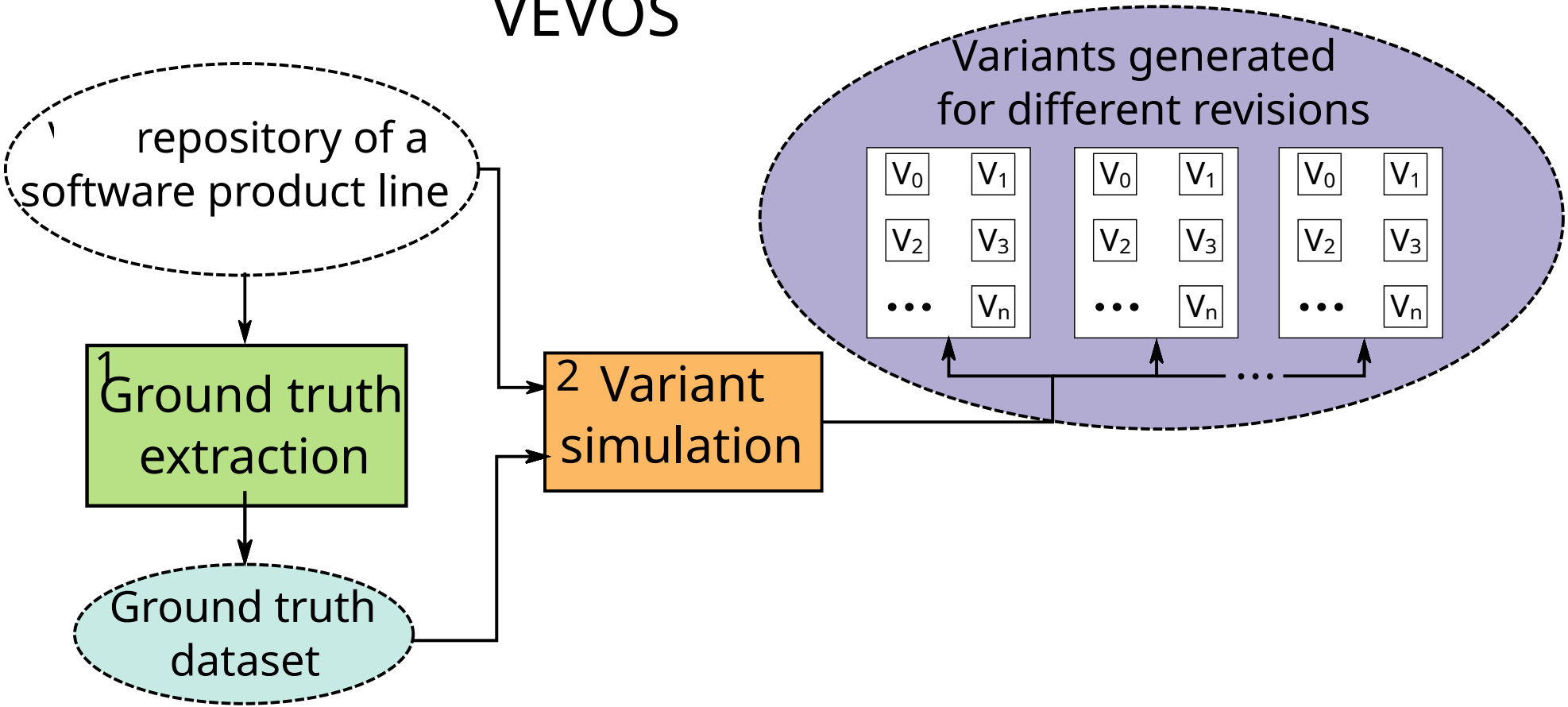
VEVOS


repository of a
software product line
1
Ground truth
extraction
Ground truth
dataset

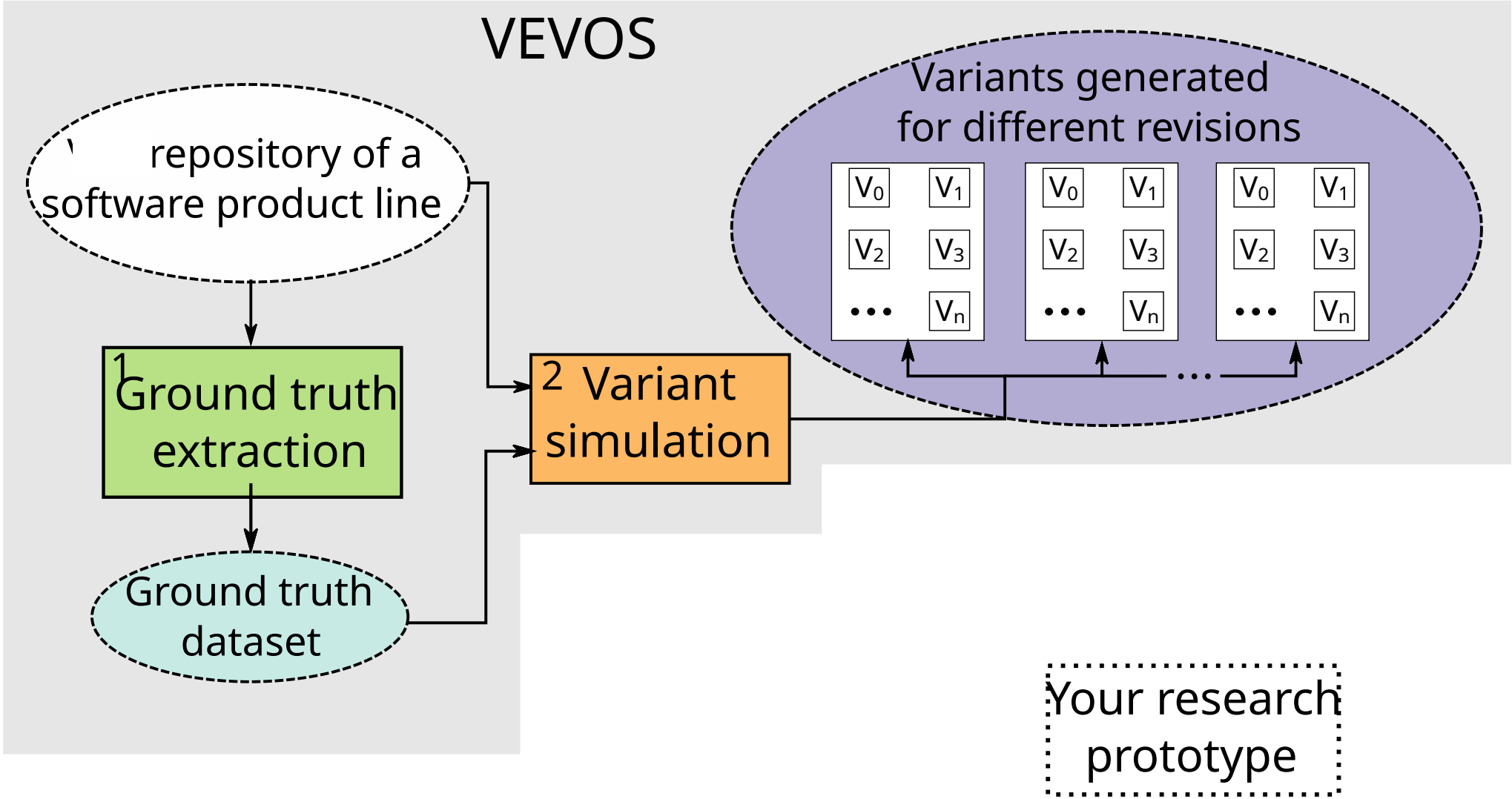
VEVOS


repository of a software product line
1 Ground truth extraction
Ground truth dataset
2 Variant simulation

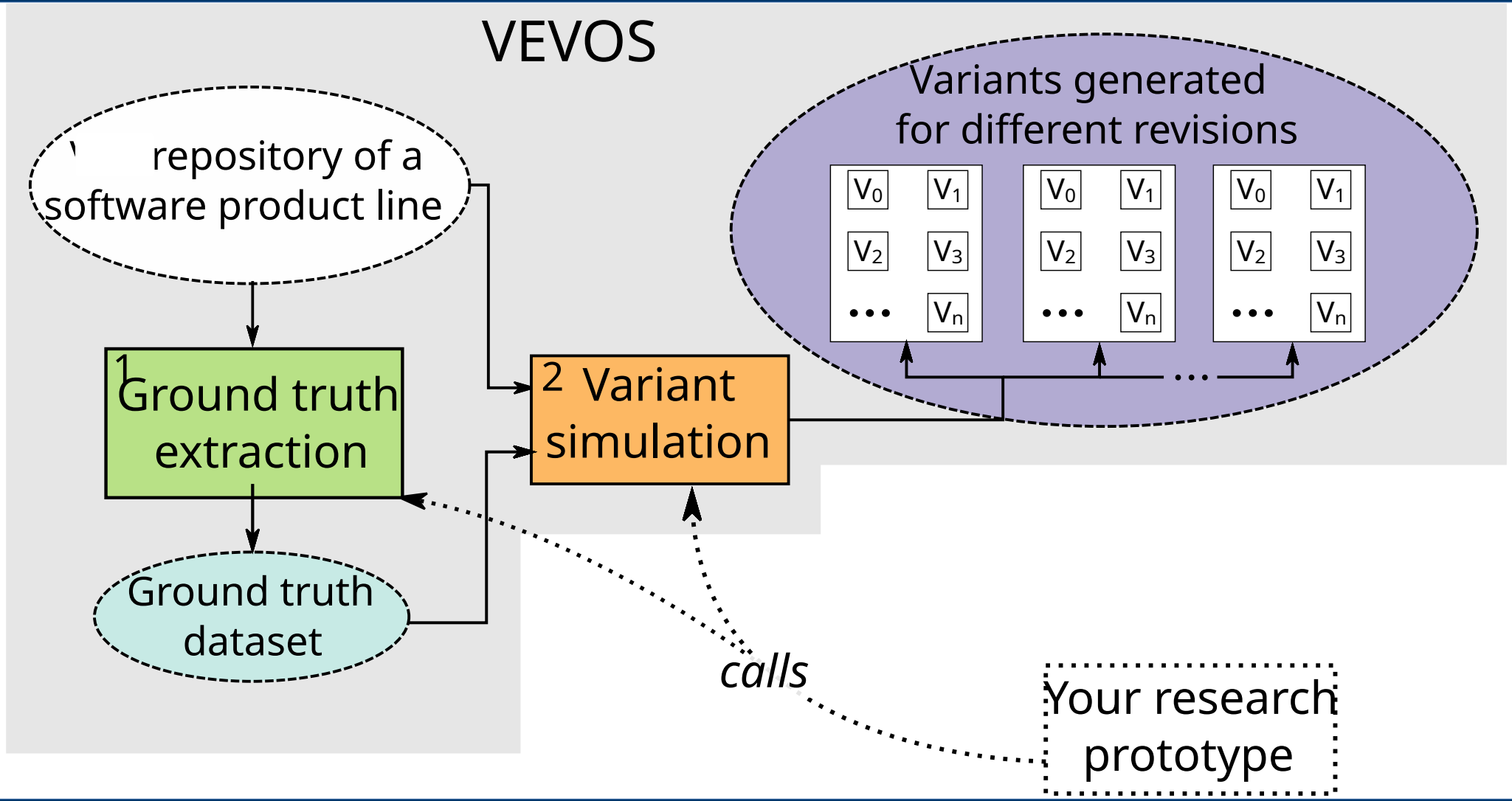
VEVOS

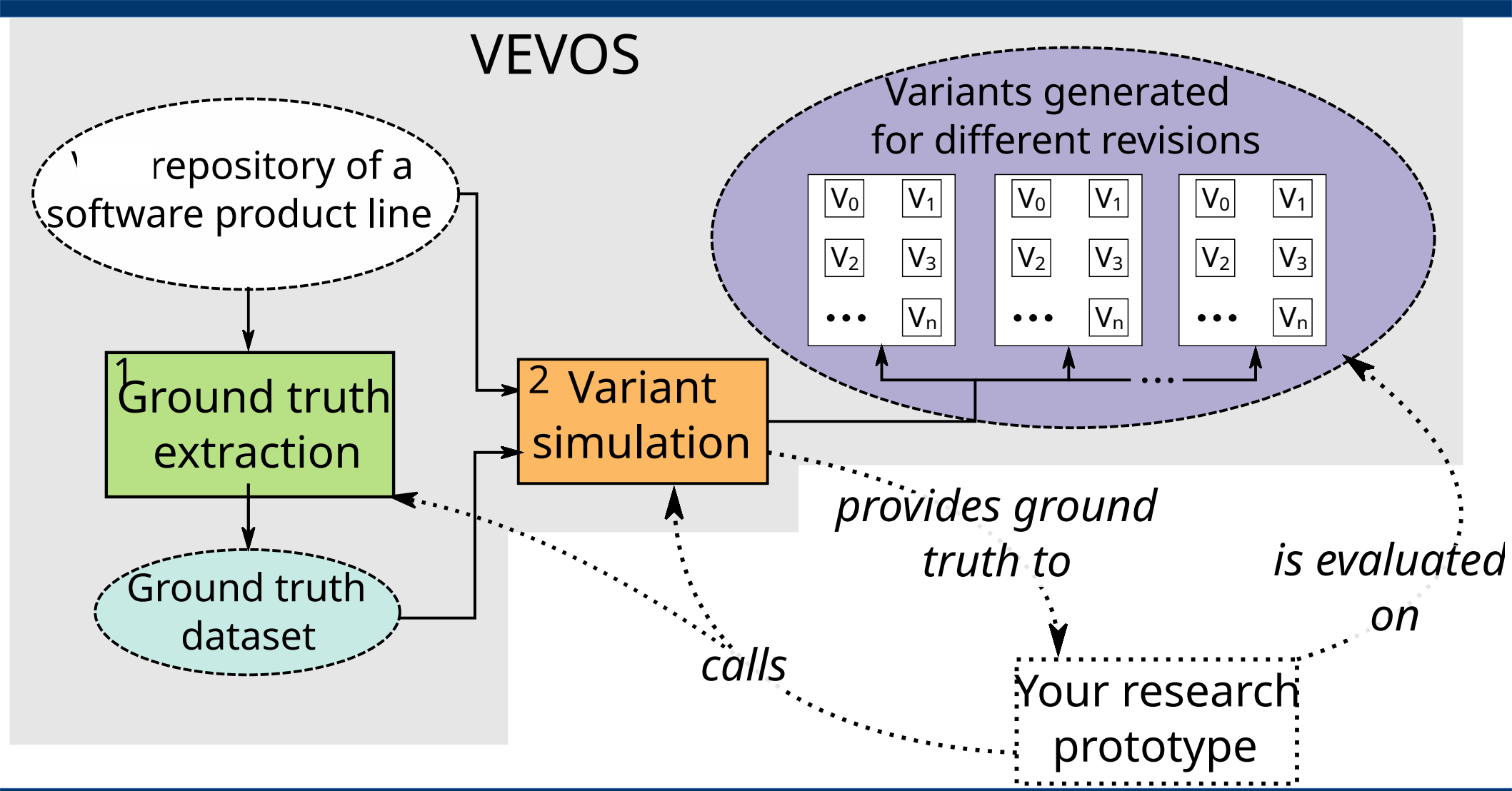

repository of a software product line
1 Ground truth extraction
Ground truth dataset
2 Variant simulation
Variants generated for different revisions
V0
V1
V2
V3
Vn

VEVOS


repository of a software product line
1 Ground truth extraction
Ground truth dataset
2 Variant simulation
Variants generated for different revisions
V0 V1
V2 V3
... Vn
Your research prototype

VEVOS


repository of a software product line
1 Ground truth extraction
Ground truth dataset
2 Variant simulation
Variants generated for different revisions
V0 V1 V2 V3 ... Vn
V0 V1 V2 V3 ... Vn
V0 V1 V2 V3 ... Vn
...
calls
Your research prototype


VEVOS
repository of a software product line
1 Ground truth extraction
Ground truth dataset
2 Variant simulation
Variants generated for different revisions
V0
V1
V2
V3
Vn
provides ground truth to
is evaluated on
calls
Your research prototype

Ground truth extraction

Extracting a ground truth

SPL history

```
1  int x = foo();
```

```
1  int x = foo();
2  #if Debug
3    int status = calculate(x);
4    log(x);
5  #endif
```

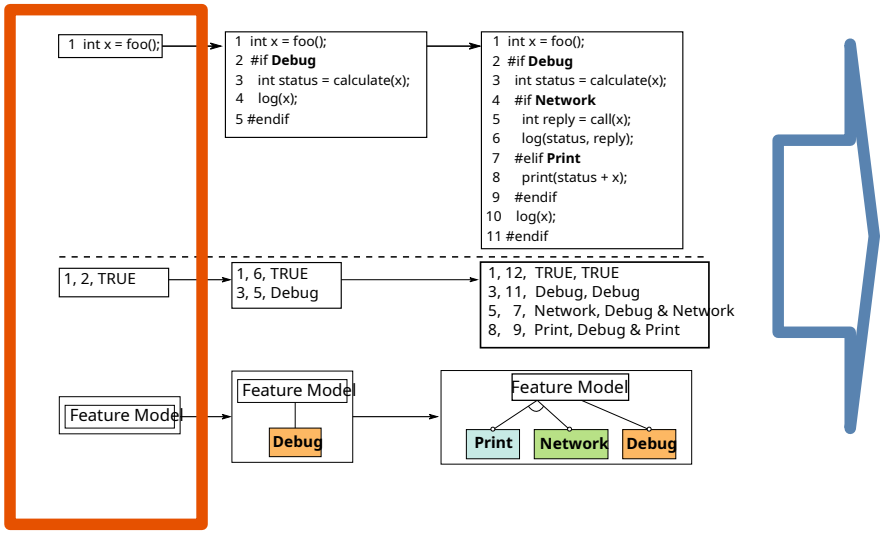
```
1  int x = foo();
2  #if Debug
3    int status = calculate(x);
4    #if Network
5      int reply = call(x);
6      log(status, reply);
7    #elif Print
8      print(status + x);
9    #endif
10   log(x);
11 #endif
```

Variant simulation

Simulation of variants

Simulation of variants

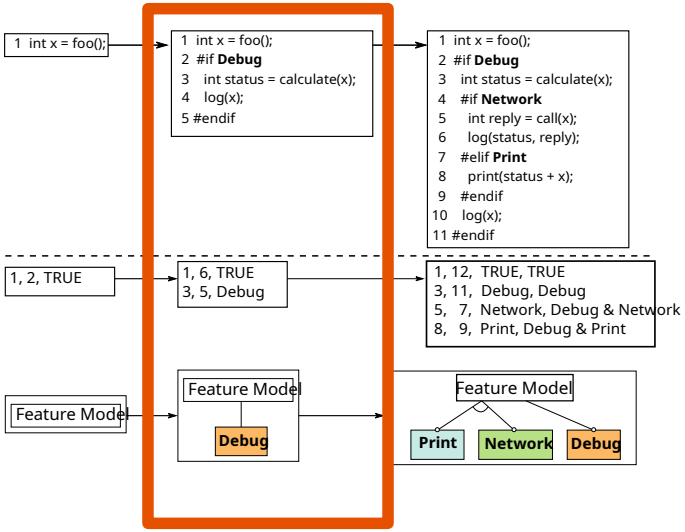
Ground truth dataset



	Source code	Configuration	Feature mapping and presence condition
V0	1 int x = foo();	Config	1, 2, TRUE, TRUE

Simulation of variants

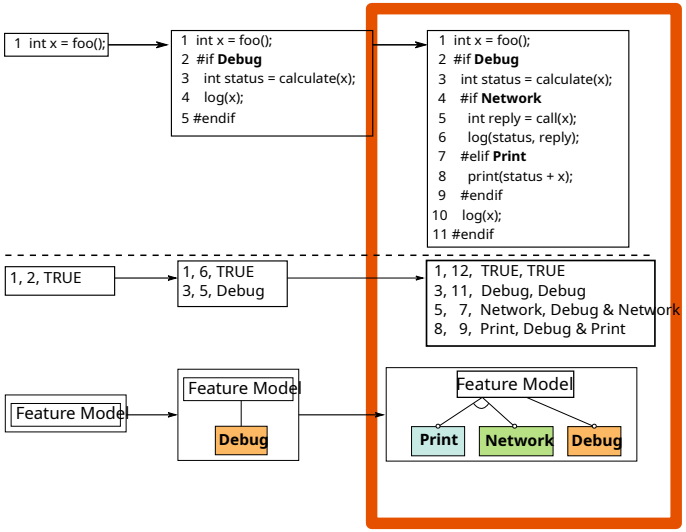
Ground truth dataset



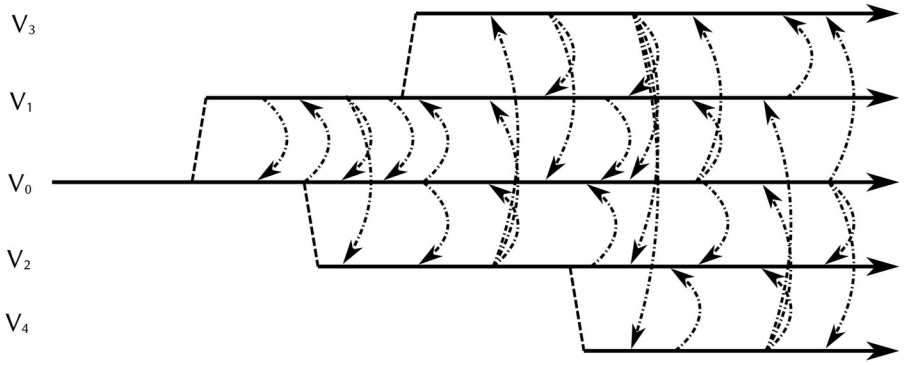
	Source code	Configuration	Feature mapping and presence condition
V0	1 int x = foo();	Config	1, 2, TRUE, TRUE
V1	1 int x = foo(); 2 int status = calculate(x); 3 log(x);	Config	1, 4, TRUE, TRUE 2, 4, Debug, Debug

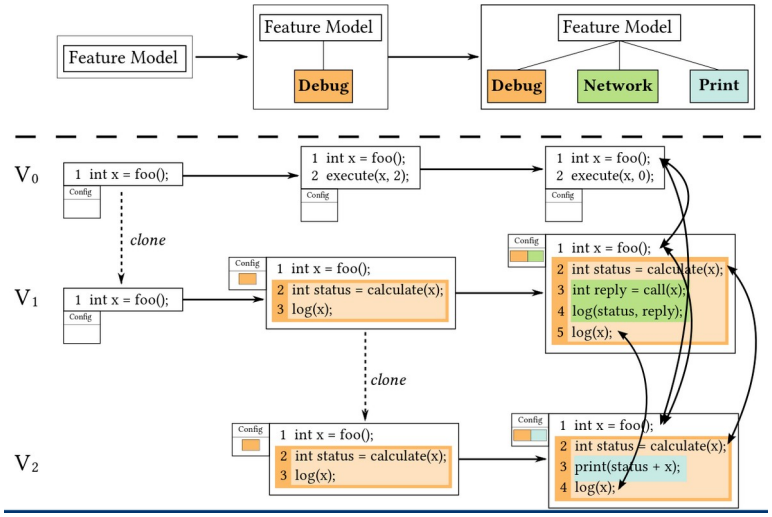
Simulation of variants

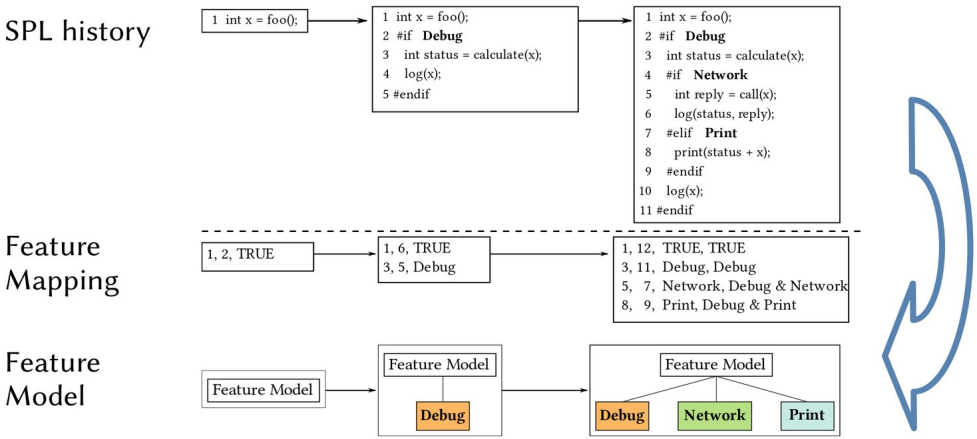
Ground truth dataset

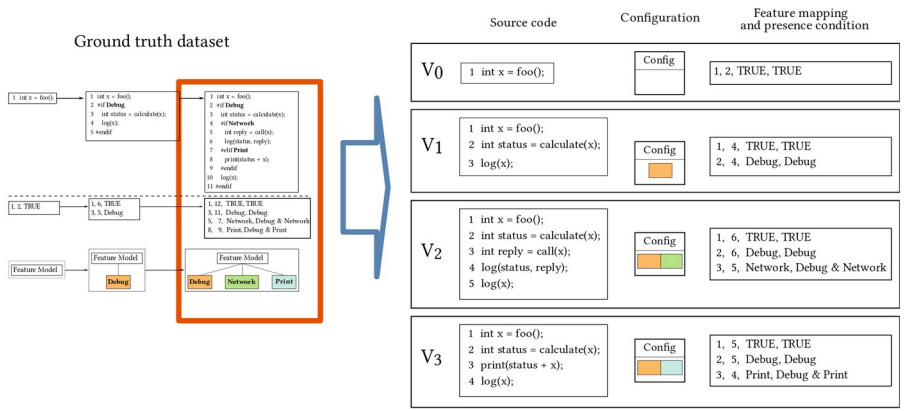


	Source code	Configuration	Feature mapping and presence condition
V0	1 int x = foo();	Config	1, 2, TRUE, TRUE
V1	1 int x = foo(); 2 int status = calculate(x); 3 log(x);	Config	1, 4, TRUE, TRUE 2, 4, Debug, Debug
V2	1 int x = foo(); 2 int status = calculate(x); 3 int reply = call(x); 4 log(status, reply); 5 log(x);	Config	1, 6, TRUE, TRUE 2, 6, Debug, Debug 3, 5, Network, Debug & Networ
V3	1 int x = foo(); 2 int status = calculate(x); 3 print(status + x); 4 log(x);	Config	1, 5, TRUE, TRUE 2, 5, Debug, Debug 3, 4, Print, Debug & Print


V3
V1
V0
V2
V4


Feature Model
Feature Model
Debug
Feature Model
Debug
Network
Print
V0
1 int x = foo();
Config
1 int x = foo();
2 execute(x, 2);
Config
1 int x = foo();
2 execute(x, 0);
Config
clone
V1
1 int x = foo();
Config
Config
1 int x = foo();
2 int status = calculate(x);
3 log(x);
Config
1 int x = foo();
2 int status = calculate(x);
3 int reply = call(x);
4 log(status, reply);
5 log(x);
clone
V2
Config
1 int x = foo();
2 int status = calculate(x);
3 log(x);
Config
1 int x = foo();
2 int status = calculate(x);
3 print(status + x);
4 log(x);


SPL history
1 int x = foo();
1 int x = foo();
2 #if Debug
3 int status = calculate(x);
4 log(x);
5 #endif
1 int x = foo();
2 #if Debug
3 int status = calculate(x);
4 #if Network
5 int reply = call(x);
6 log(status, reply);
7 #elif Print
8 print(status + x);
9 #endif
10 log(x);
11 #endif
Feature Mapping
1, 2, TRUE
1, 6, TRUE
3, 5, Debug
1, 12, TRUE, TRUE
3, 11, Debug, Debug
5, 7, Network, Debug & Network
8, 9, Print, Debug & Print
Feature Model
Feature Model
Debug
Feature Model
Debug
Network
Print


Ground truth dataset
Source code
Configuration
Feature mapping and presence condition
V0
1 int x = foo();
Config
1, 2, TRUE, TRUE
V1
1 int x = foo();
2 int status = calculate(x);
3 log(x);
Config
1, 4, TRUE, TRUE
2, 4, Debug, Debug
V2
1 int x = foo();
2 int status = calculate(x);
3 int reply = call(x);
4 log(status, reply);
5 log(x);
Config
1, 6, TRUE, TRUE
2, 6, Debug, Debug
3, 5, Network, Debug & Network
V3
1 int x = foo();
2 int status = calculate(x);
3 print(status + x);
4 log(x);
Config
1, 5, TRUE, TRUE
2, 5, Debug, Debug
3, 4, Print, Debug & Print

Appendix

What are VEVOS' current extraction capabilities?

- Linux
  - Successful for ~140,000 of ~1,076,000 commits
  - V4.0 through v4.10
  - Development slice: 2015 - 2017
- BusyBox
  - Successful for ~5,600 of ~17,000 commits (most recent commits)
  - Development slice: 2010 - 2022

What are VEVOS' current simulation capabilities?

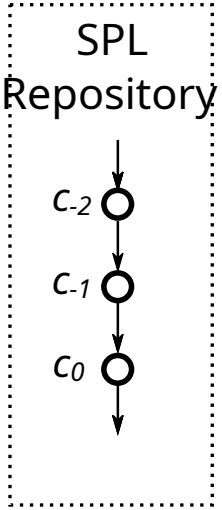
For each version of a software product line

- Random sampling of configurations
- Generation of feature-aware variants
  - Feature mapping
  - Configuration
  - Source code
  - Code matching

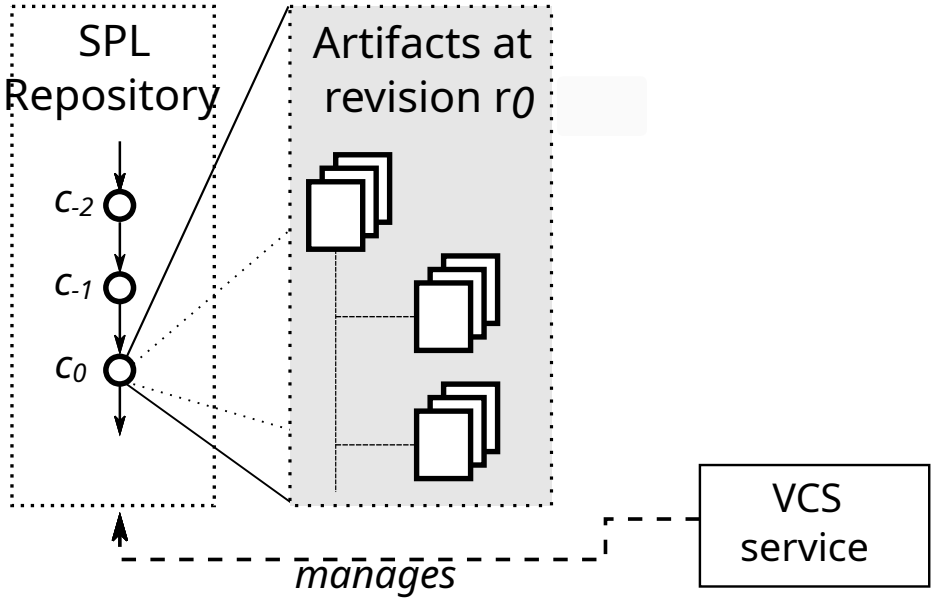
Unresolved Challenges

- Build system analysis is not robust
- Generated variants expose no unintentional divergences
  - These can occur in clone-and-own (e.g., refactoring in one variant)
  - Leads to evaluation bias

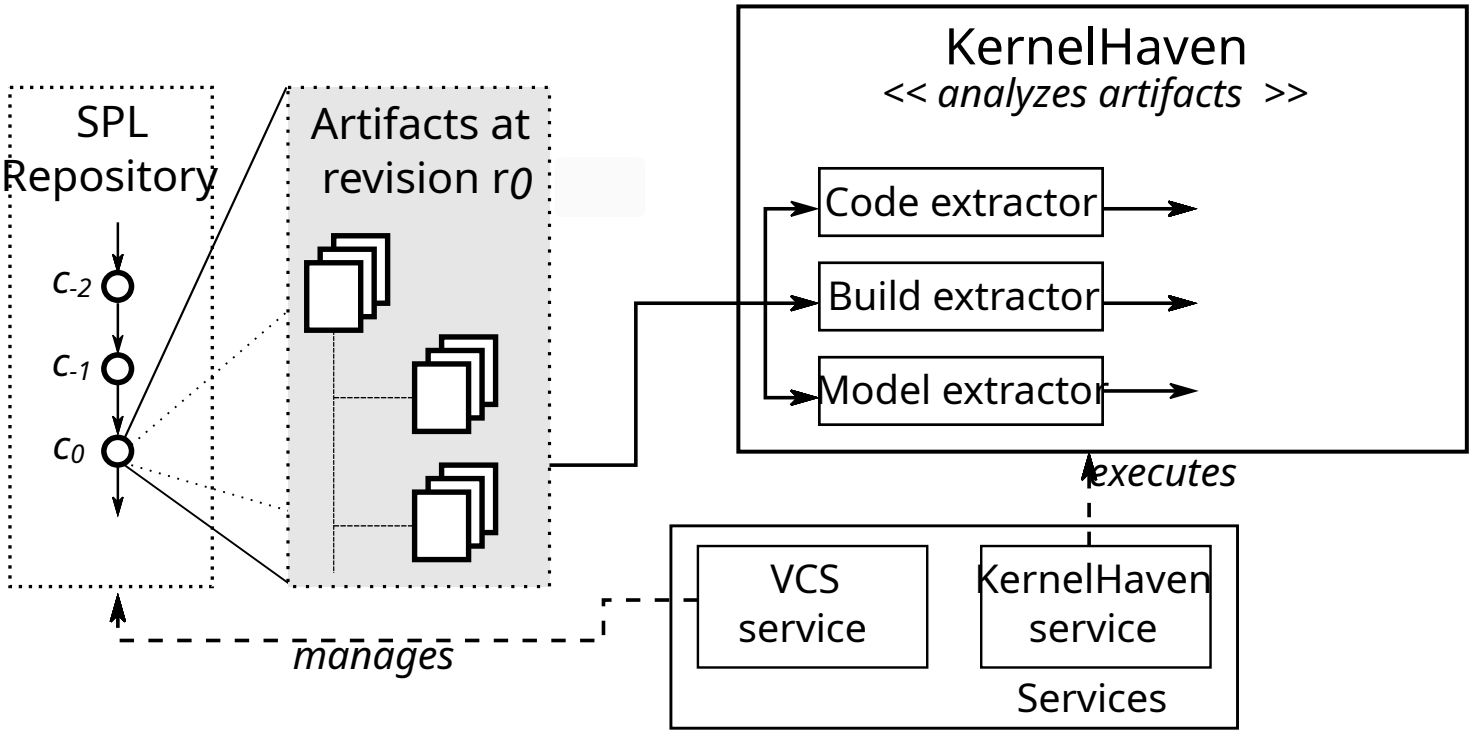
The Ground Truth Extraction



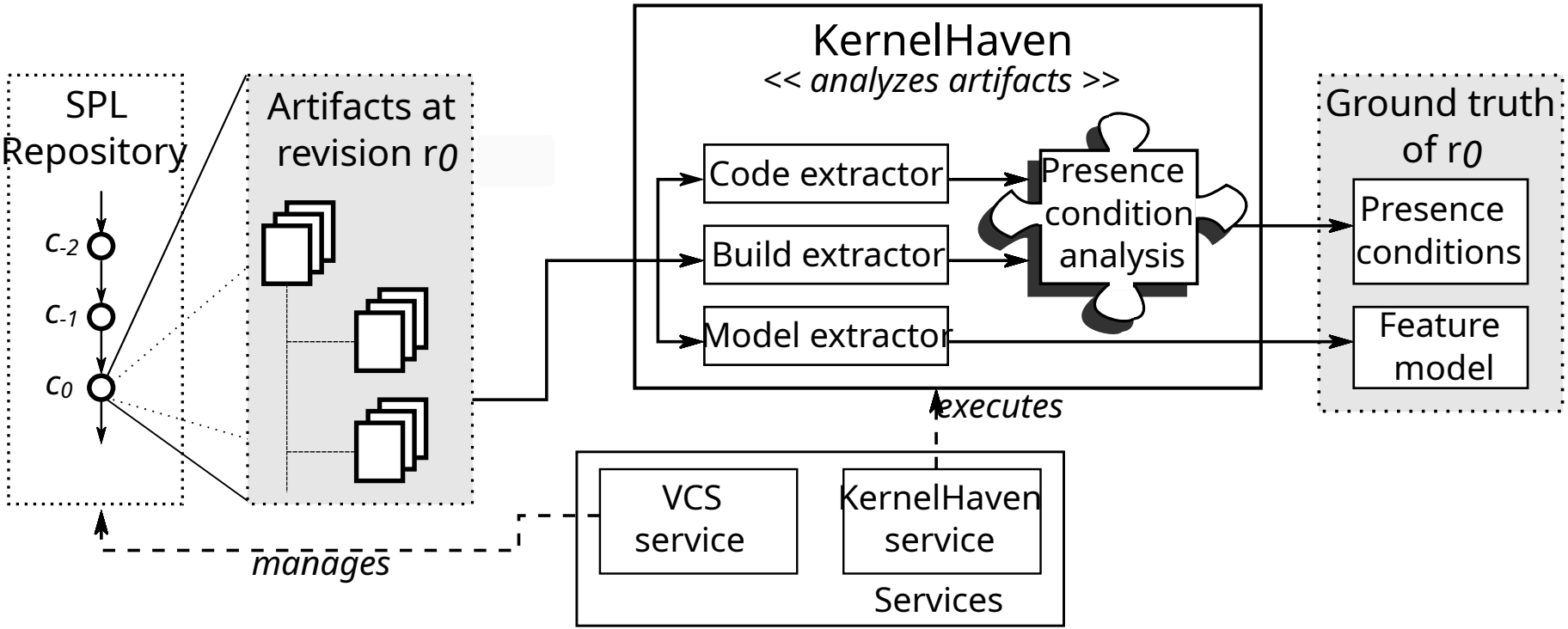
The Ground Truth Extraction



The Ground Truth Extraction



The Ground Truth Extraction



The Ground Truth Extraction