Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Swiss Confederation

Bundesamt für Statistik BFS
Office fédéral de la statistique OFS
Ufficio federale di statistica UST
Federal Statistical Office FSO

RAPs for the Swiss Federal Pension Fund

Christopher Sulkowski
Data Scientist
Data Science Competence Center (DSCC)
FSO Switzerland

Swiss Federal Pension Fund - PUBLICA

Around 42,000 pension recipients

More than 68,000 active members

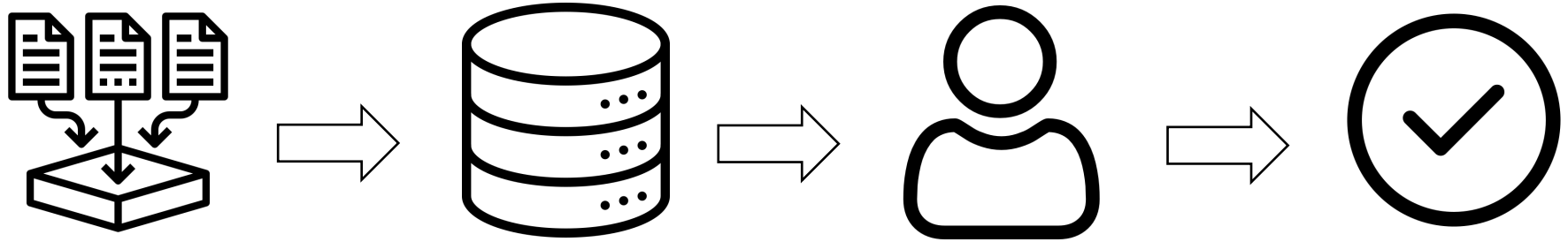
Independent pension provider established under private law

publica
DIE VORSORGE

Total assets:
CHF 40.5 billion

Recipients come from:
- Federal Administration
- ETH Domain
- Decentralized administrative units
- 70 organizations

Initial context – Manual Data Processing and Validation



Raw data for all the clients is collected every month from different sources

The data is integrated inside a single database

A human expert is responsible for executing data aggregations and validation

The aggregated and validated data is disseminated across the organization for reporting

Initial context – Manual Data Processing and Validation

Time consuming	Around 5 days of manual labor every month (around 70 procedures).
Not replicable	Original procedures required manual modifications to adapt over time.
Not analytical	Could not conduct additional procedures on the data after completion.
Not accessible	Others in the fund could not easily access results.
Not integrated	Custom integration of data across various sources.

Collaboration with the DSCC: Objectives

Automate the procedures and ensure that results are consistent over time.

Keep a record of the procedures performed and the different versions of the data.

Store results in a structured way and make them available to employees for potential future PUBLICA objectives (e.g. statistical analysis and production).

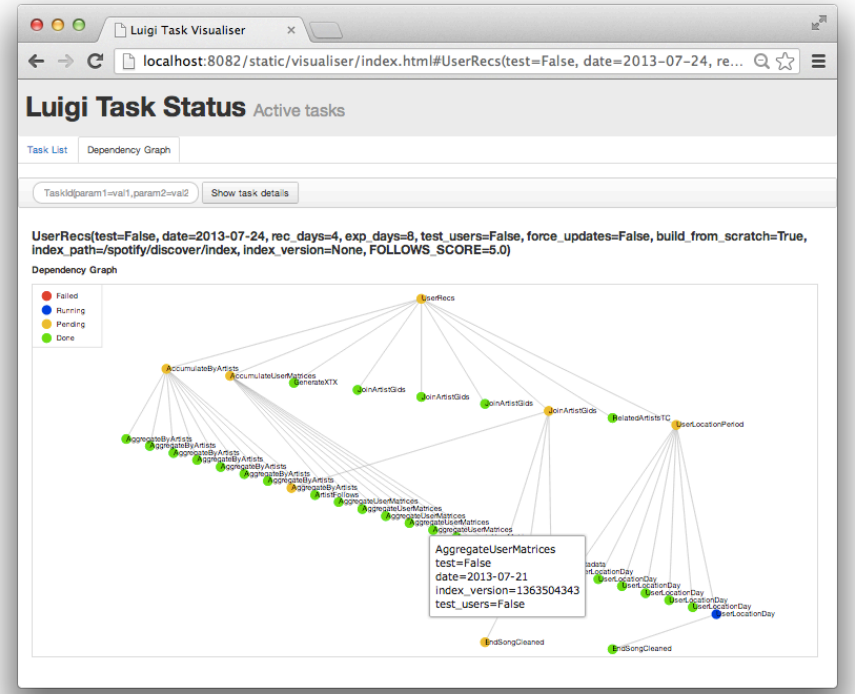
Train PUBLICA staff to use this new platform. Optimize current procedures, but also enable the fund to develop new procedures.

Outcome

Flexible application enabling fund employees to interact with the pipeline and its procedures.

Graphical interface for monitoring and observing automatic pipeline completion.

Tasks that used to take 4-5 days a month to complete manually are now completed automatically in 15 minutes.

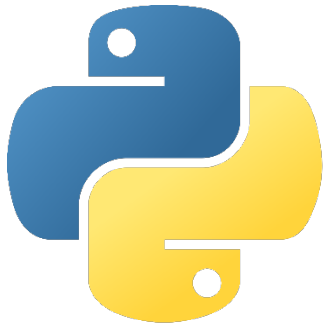
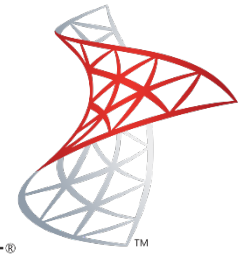


PUBLICA employees have been trained in this platform and are now using it. Plans to produce analytical and statistical work given the improved access to the data.

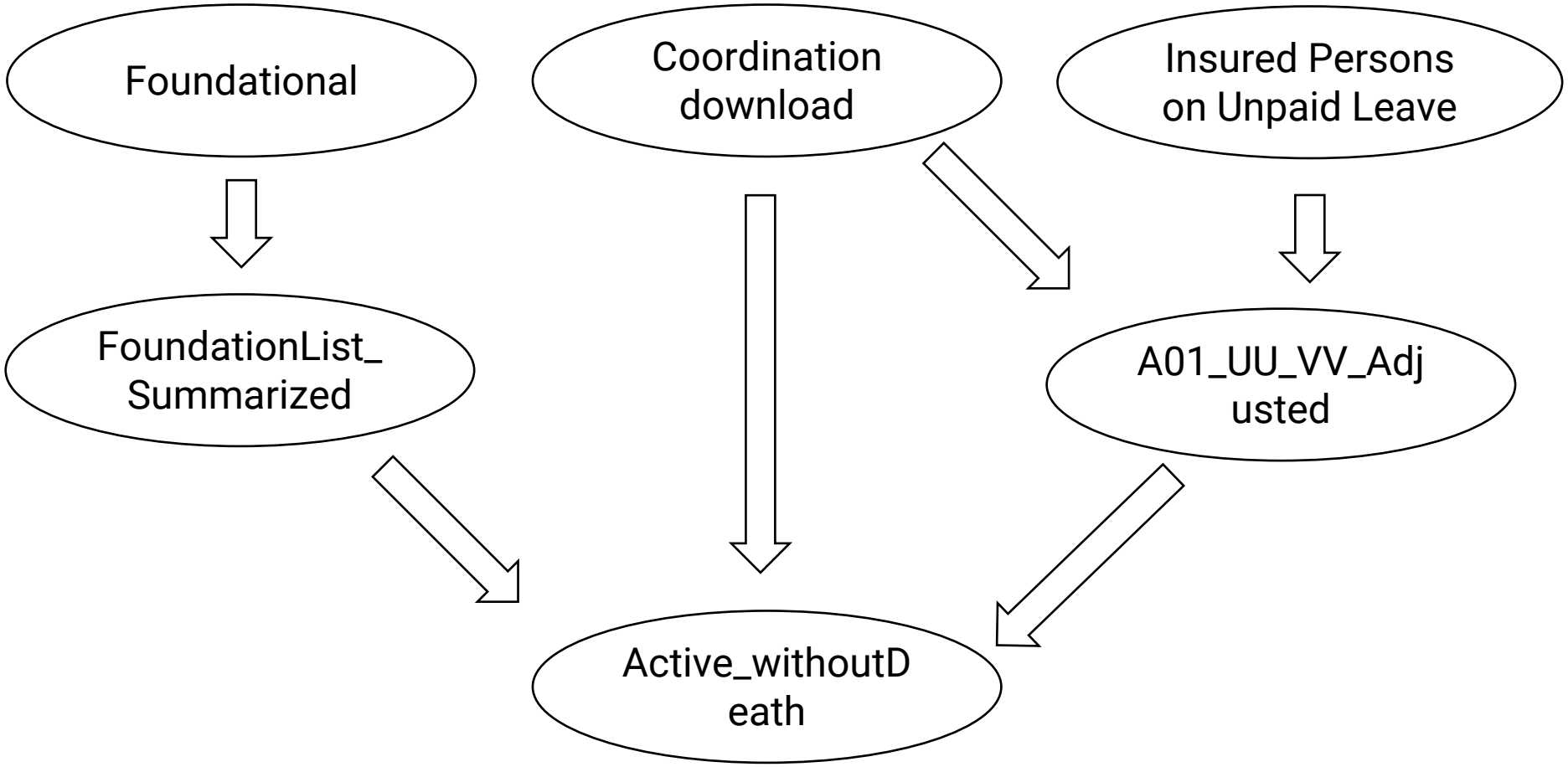
Pipeline implementation: Principles

Automated	All the procedures should automatically run with minimal human intervention.
Replicable	Running pipeline should run the same operations across different versions of the raw databases
Versioned	The history of the pipeline code and runs should be dated and easily retrieved
Accessible	Access to the results of the pipeline runs should be easy for other collaborators (of course with the right credentials)
Easy to use and extend	Building new procedures should be accessible to individuals with basic coding proficiency

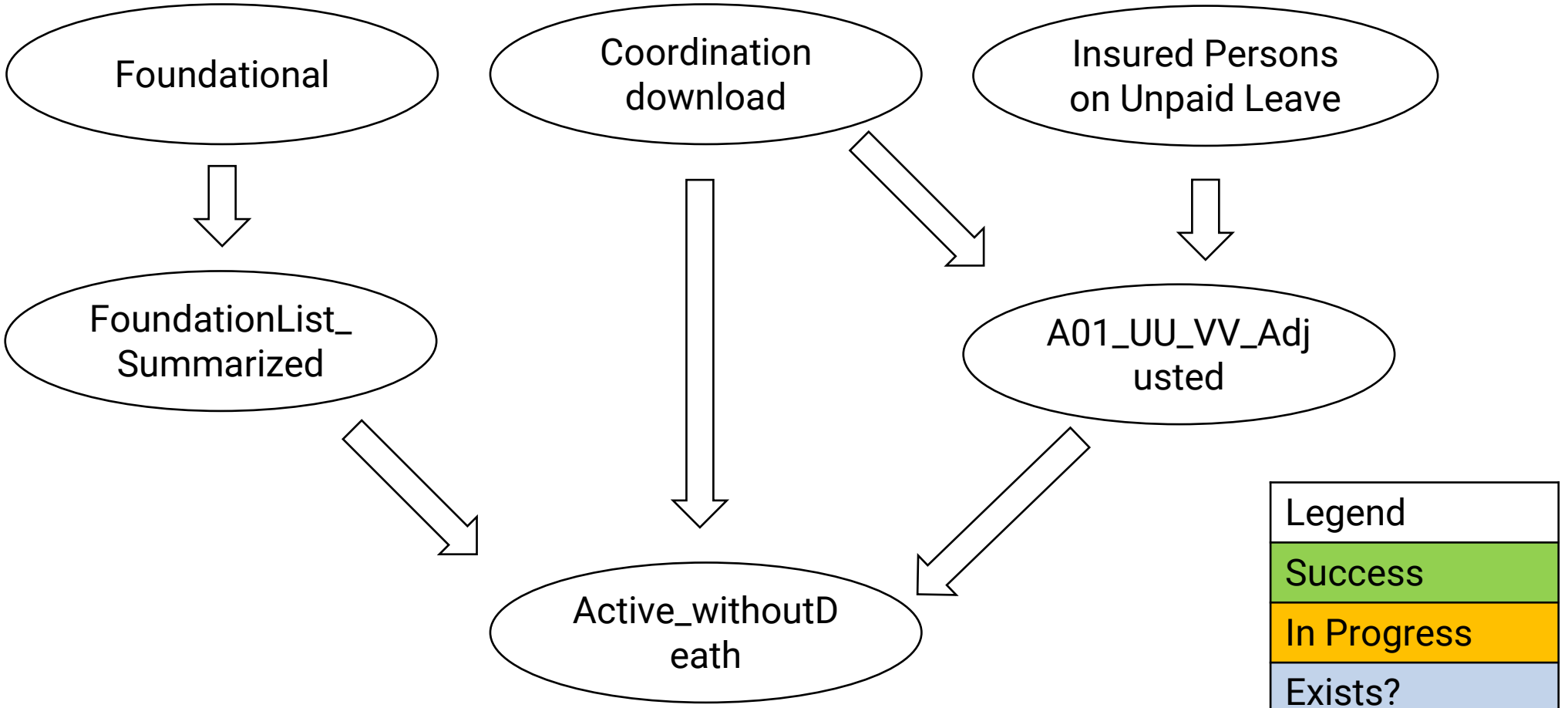
Pipeline implementation: Tools

Programming Language	Pipeline architecture	Data processing	Data storage
		 pandas  NumPy	 Microsoft® SQL Server®

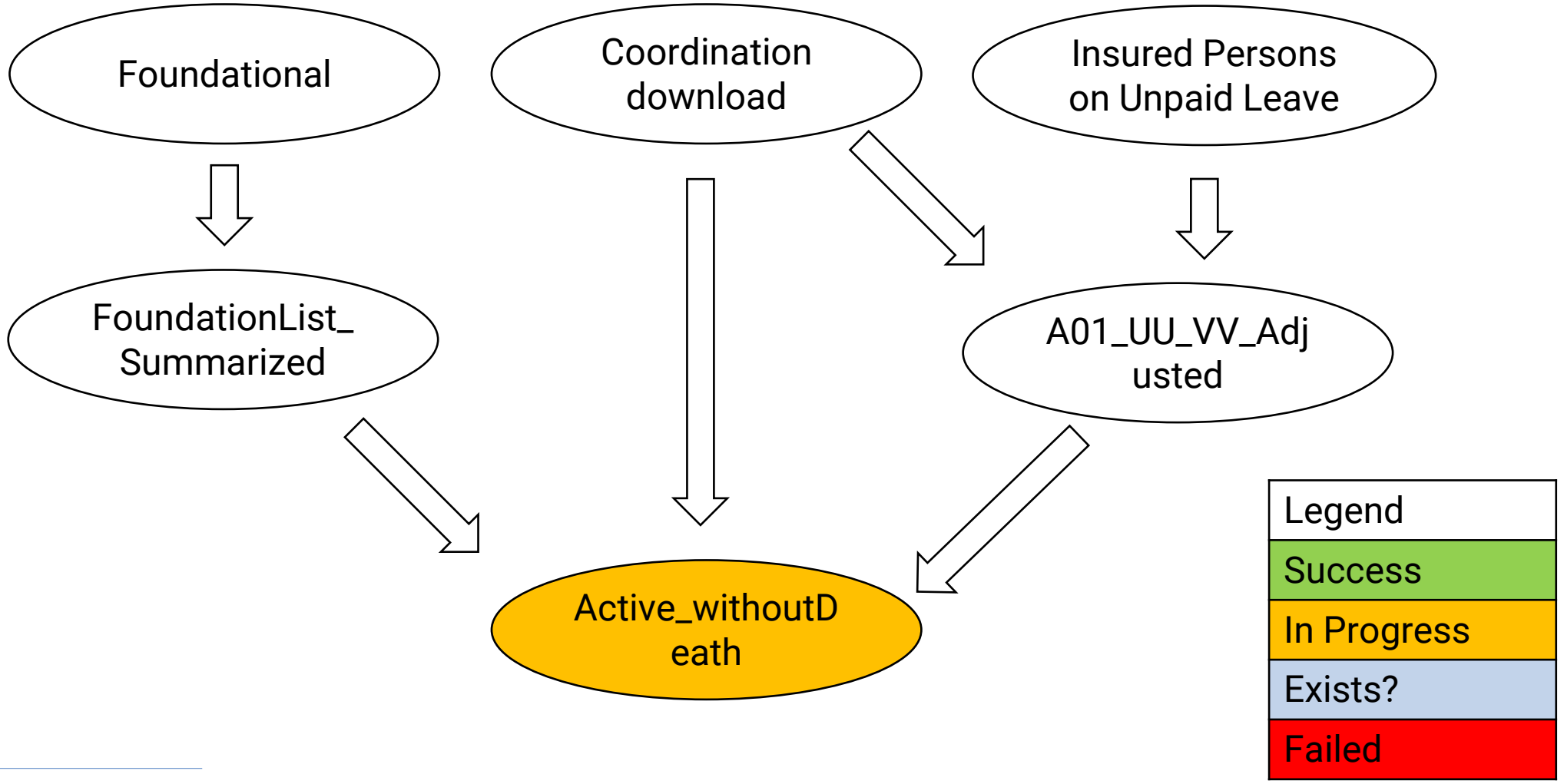
Pipeline implementation: Dependency graphs



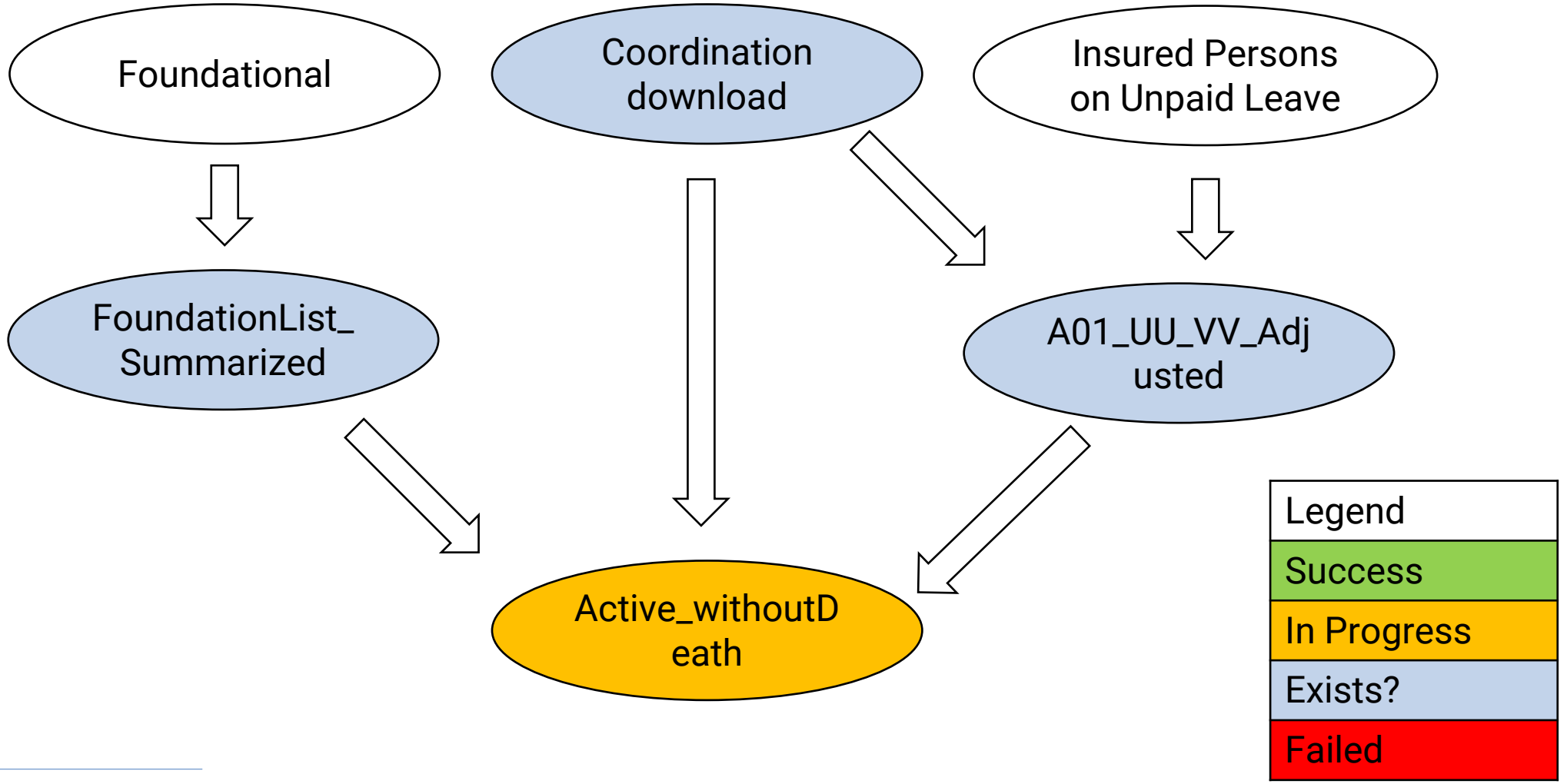
Pipeline implementation: Dependency graphs



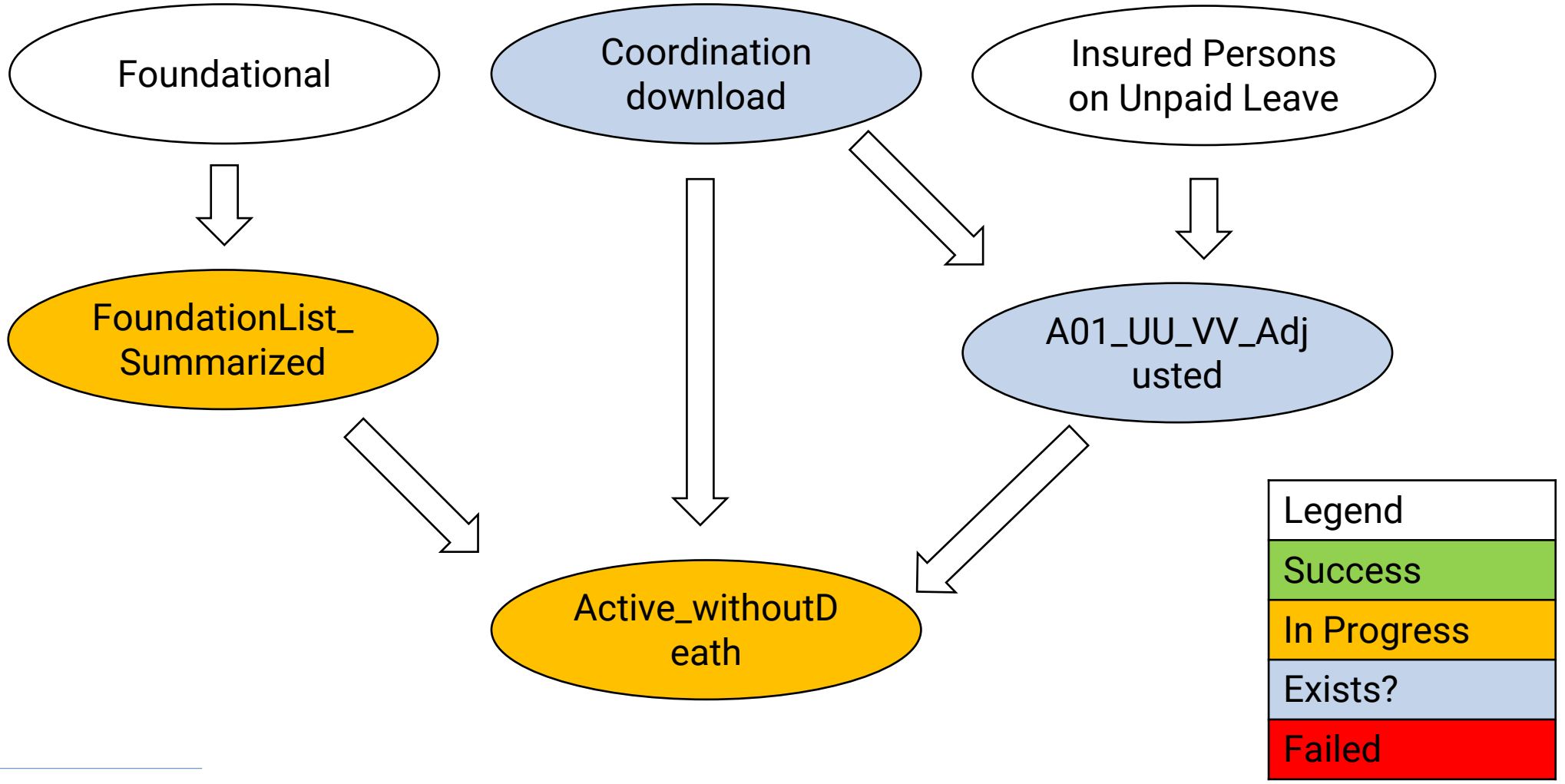
Pipeline implementation: Dependency graphs



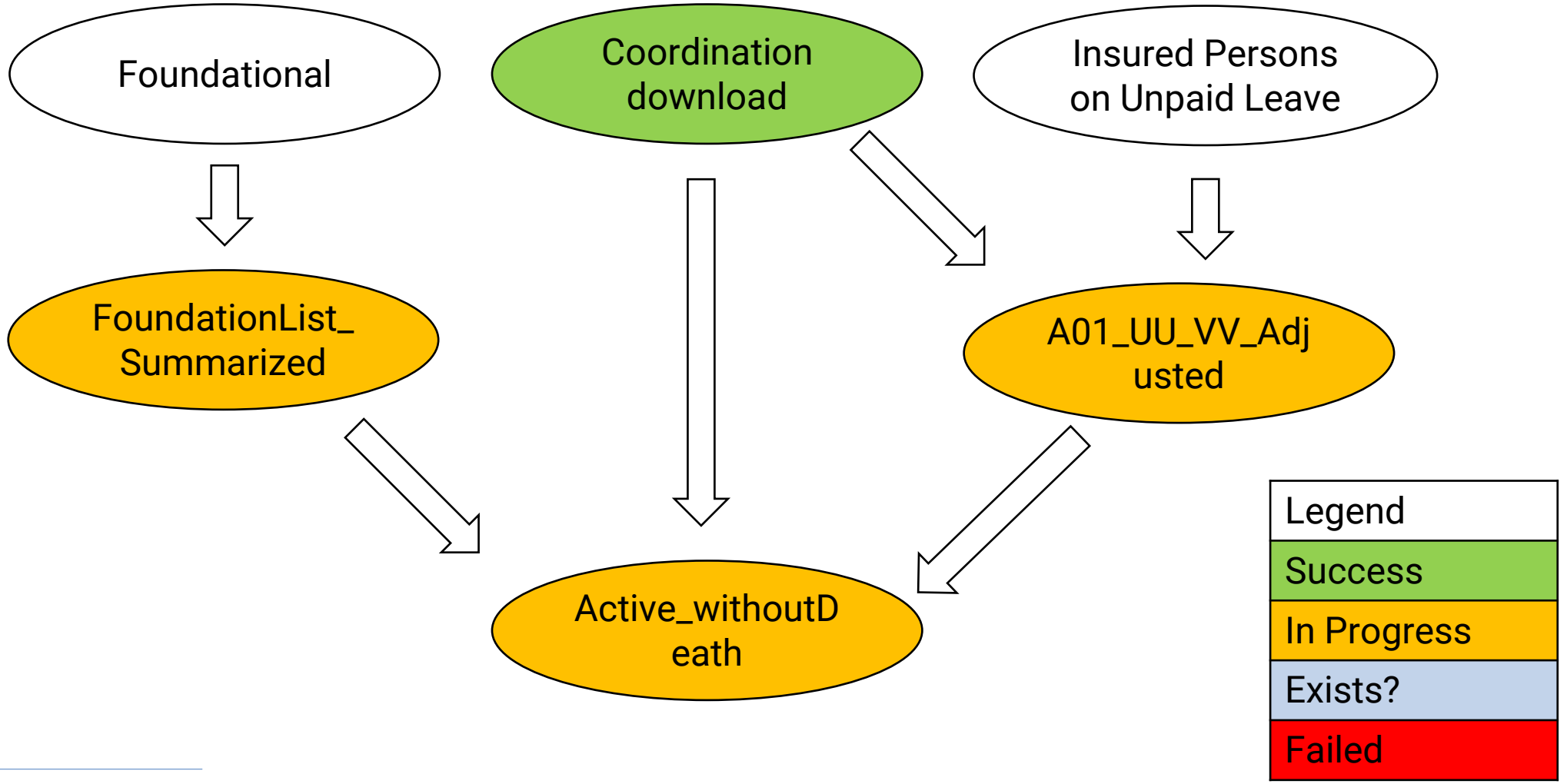
Pipeline implementation: Dependency graphs



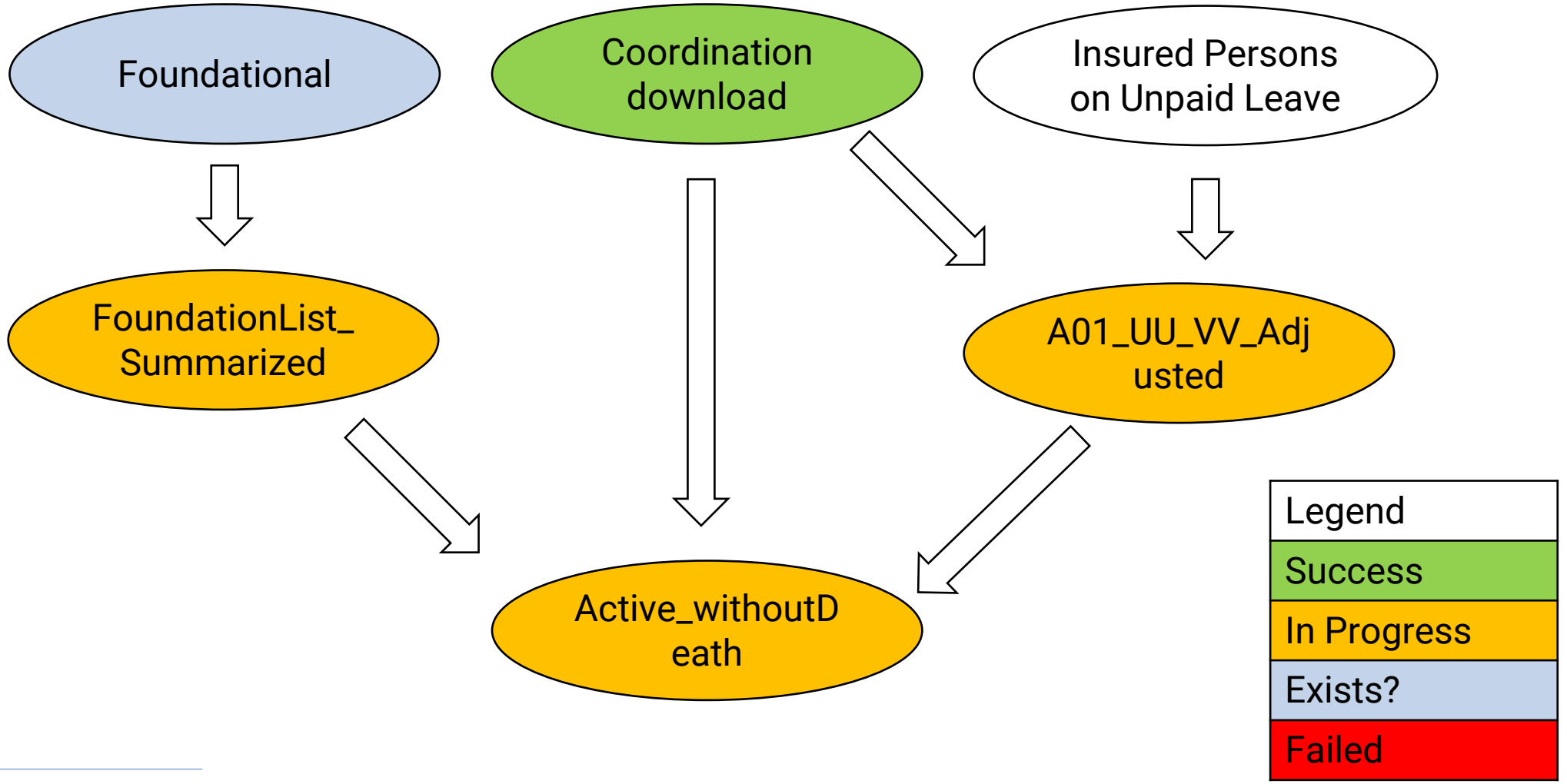
Pipeline implementation: Dependency graphs



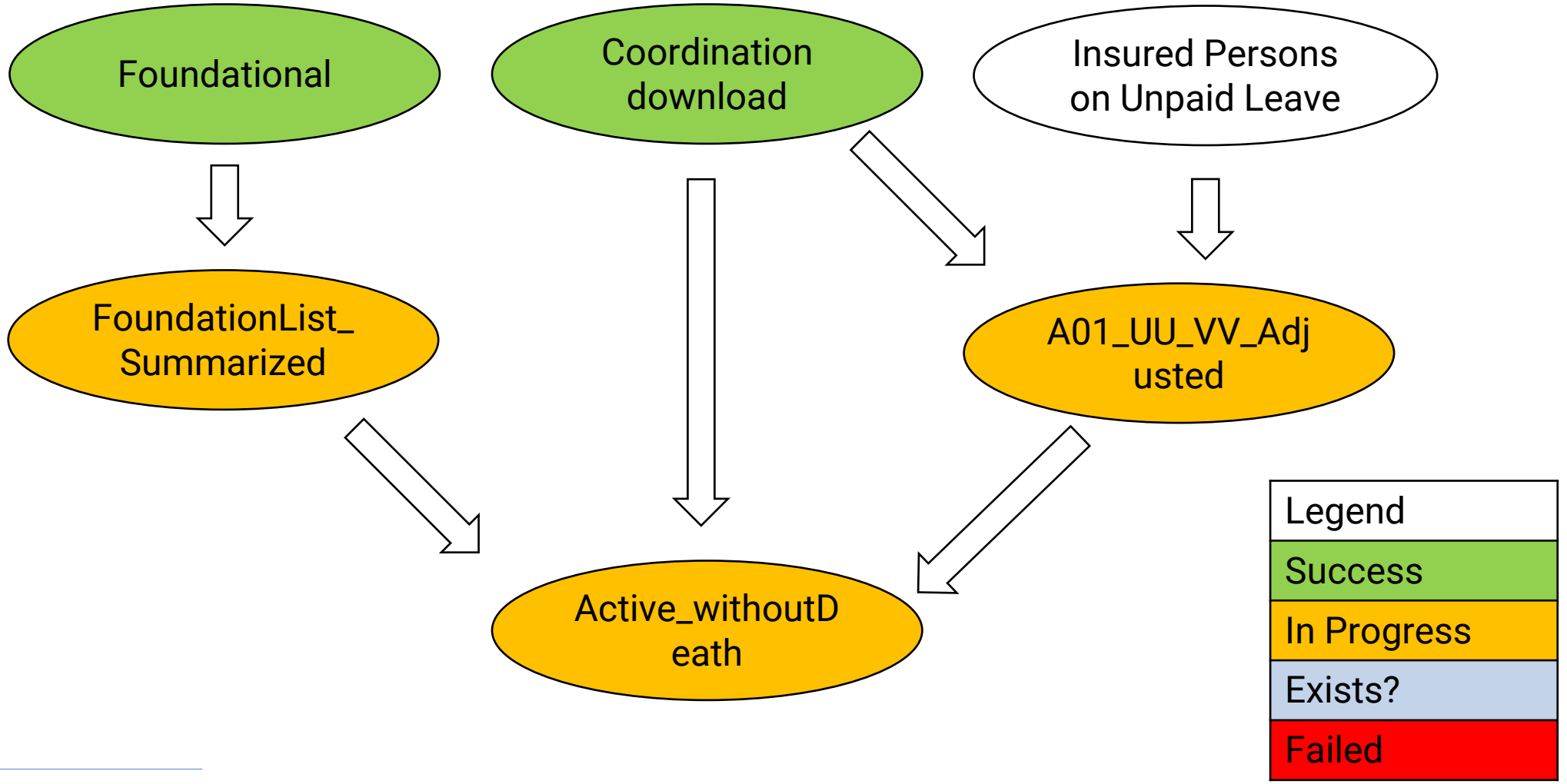
Pipeline implementation: Dependency graphs



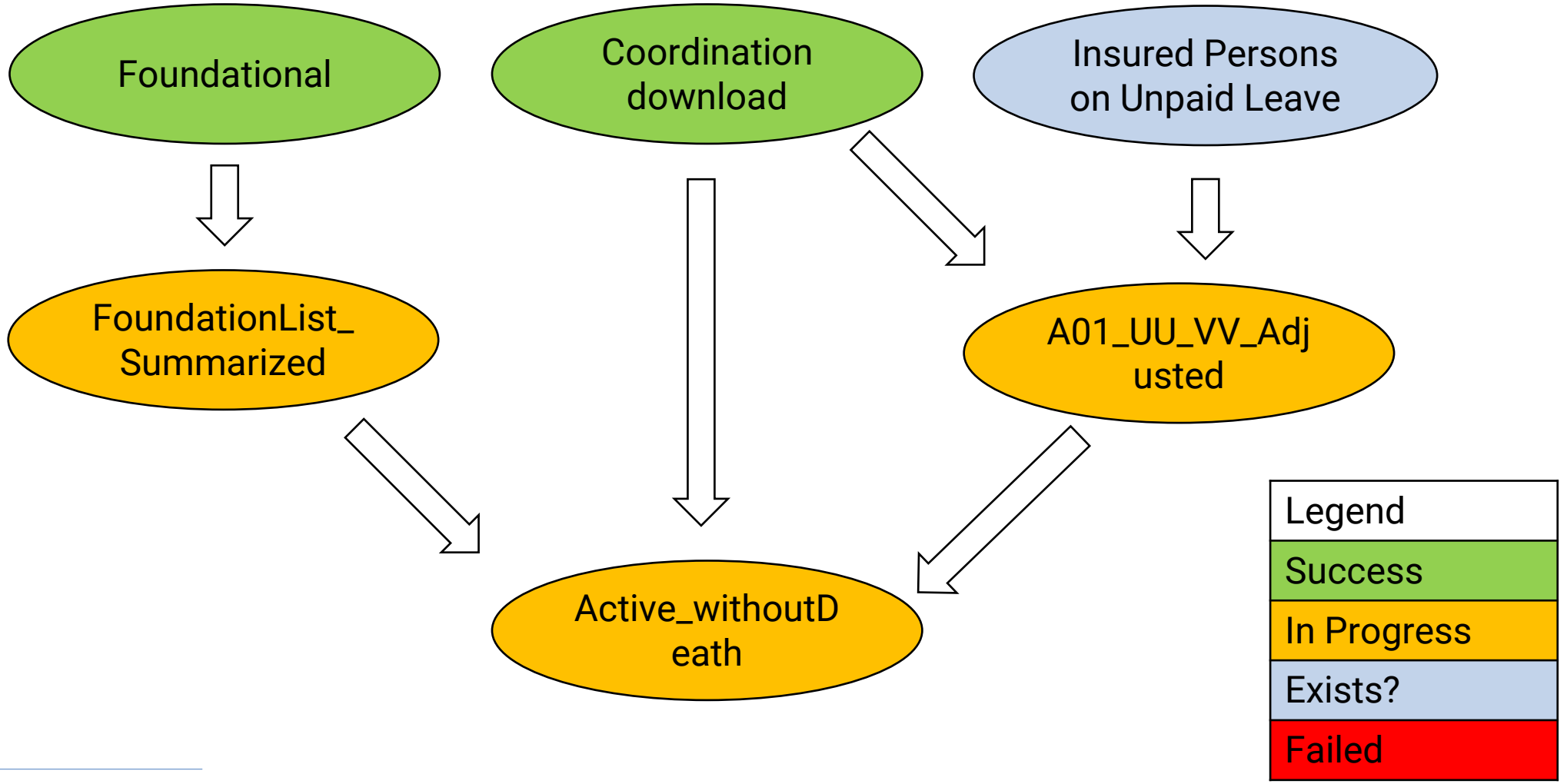
Pipeline implementation: Dependency graphs



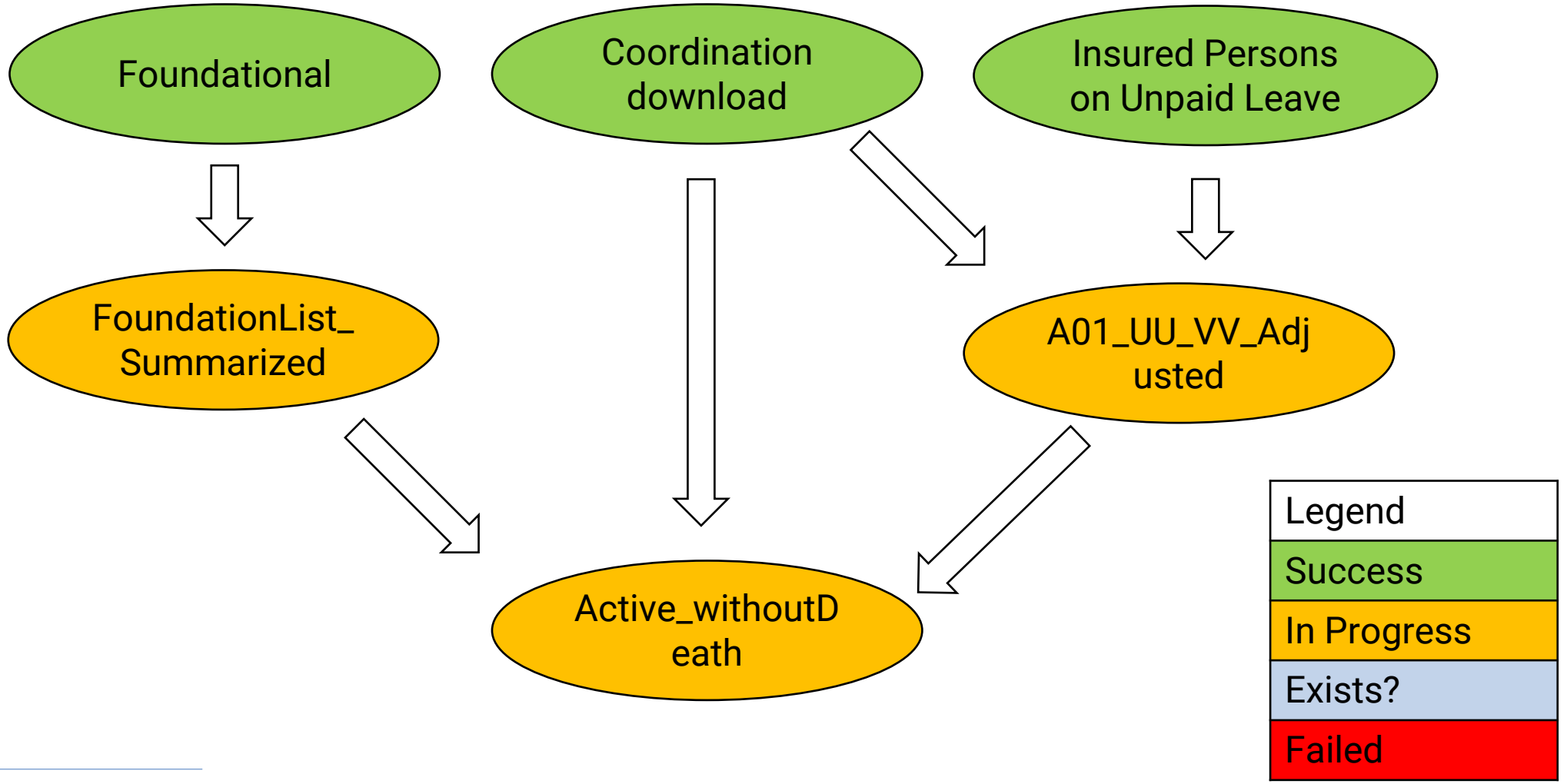
Pipeline implementation: Dependency graphs



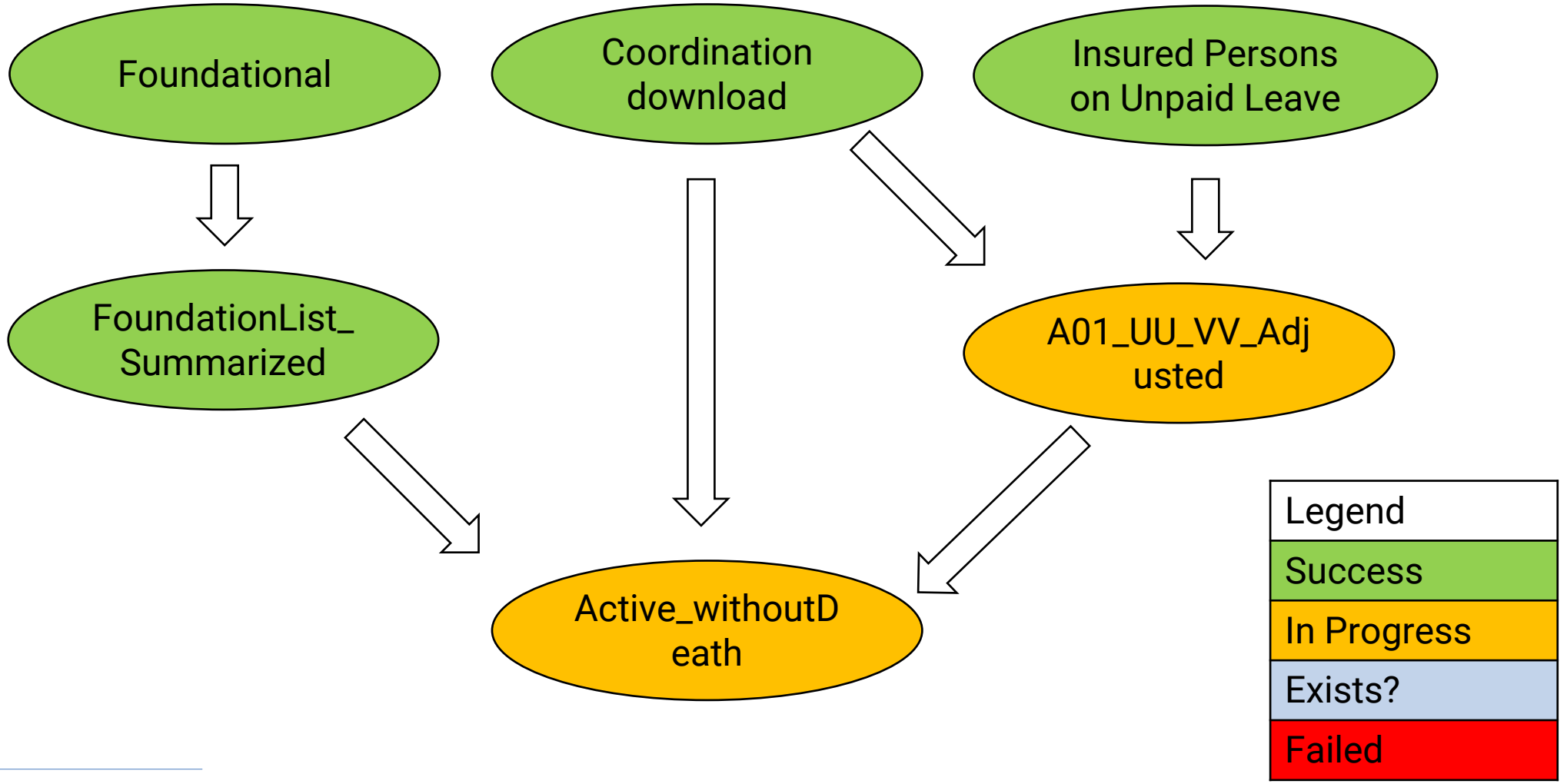
Pipeline implementation: Dependency graphs



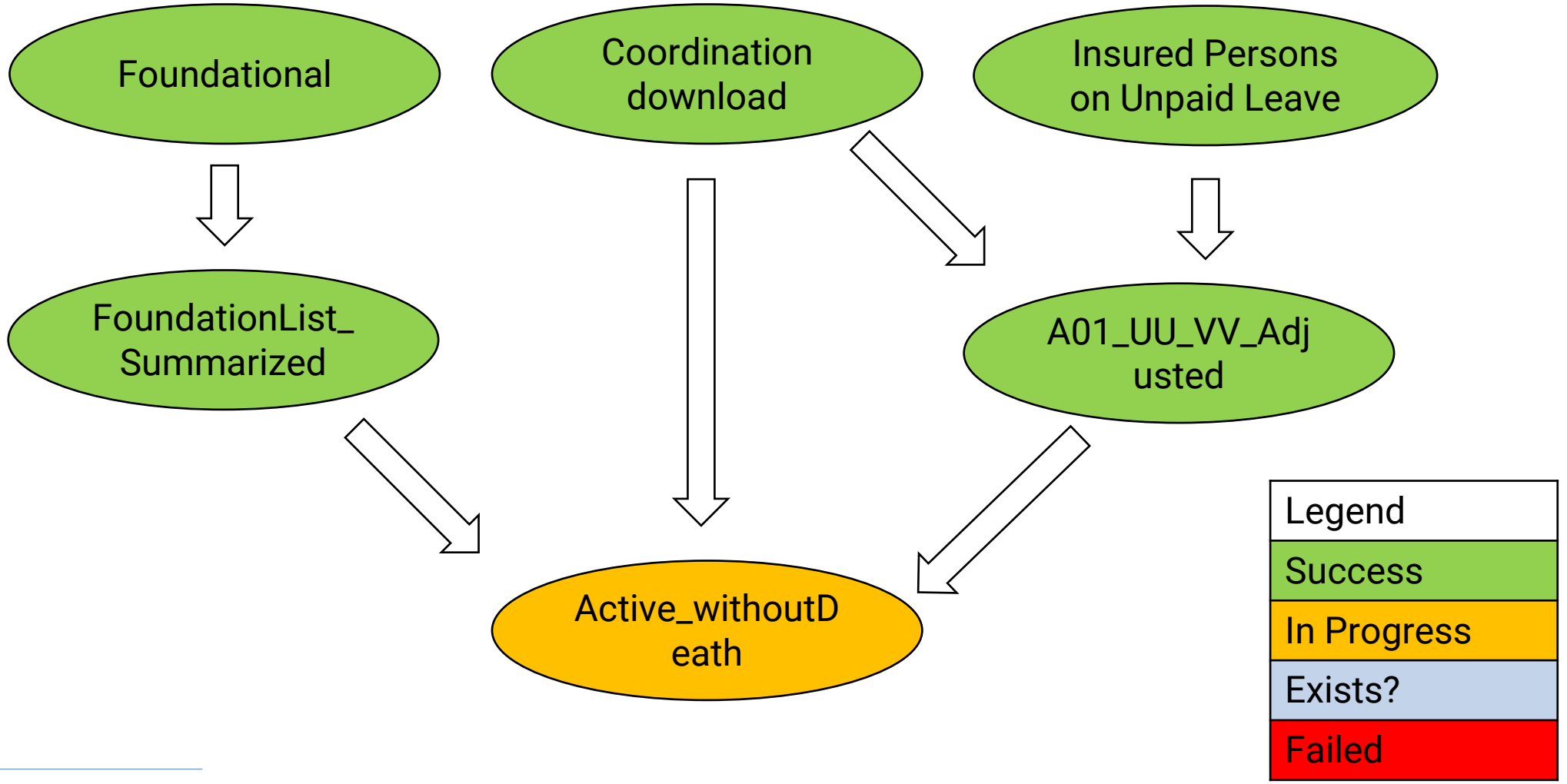
Pipeline implementation: Dependency graphs



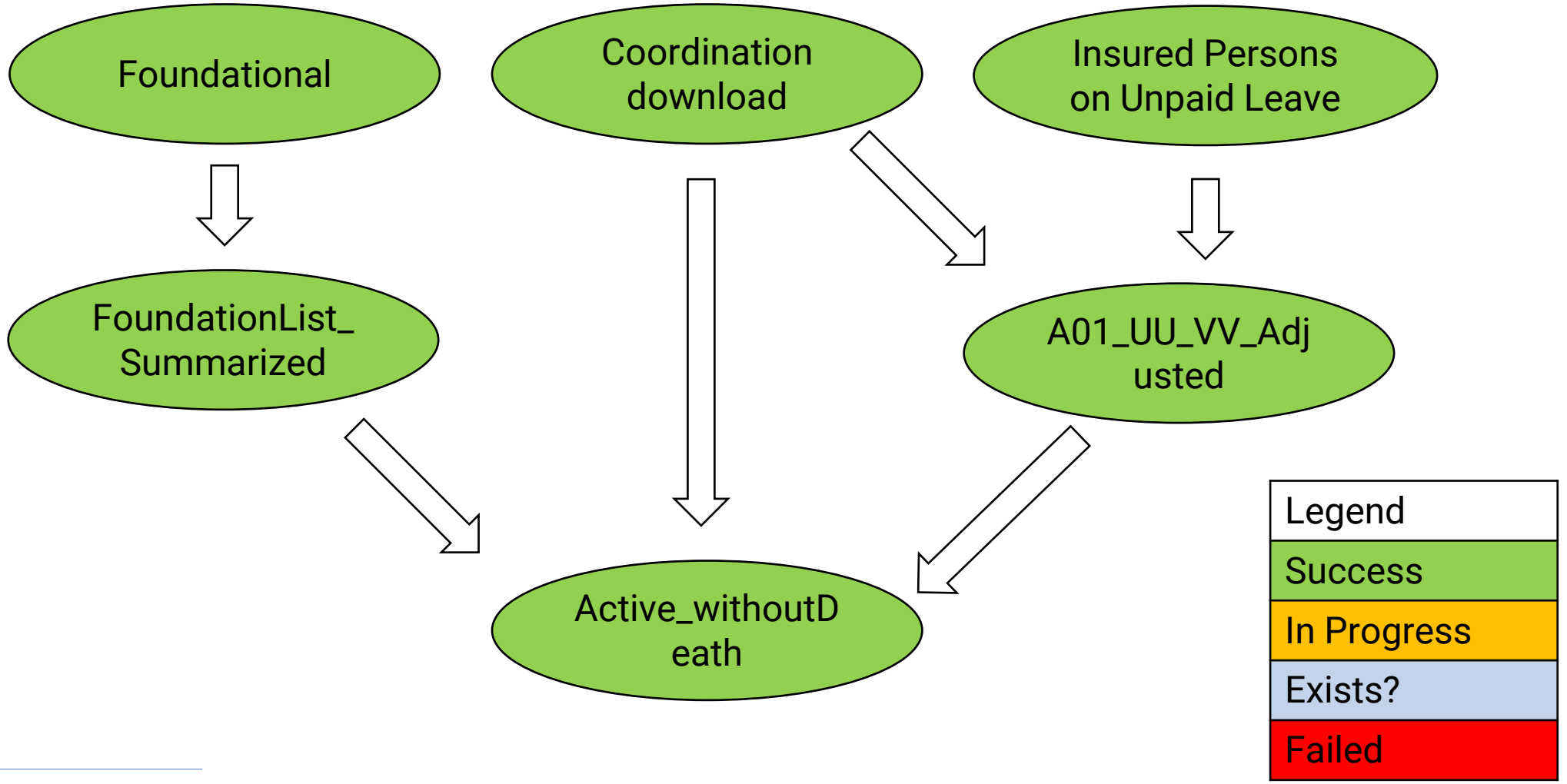
Pipeline implementation: Dependency graphs



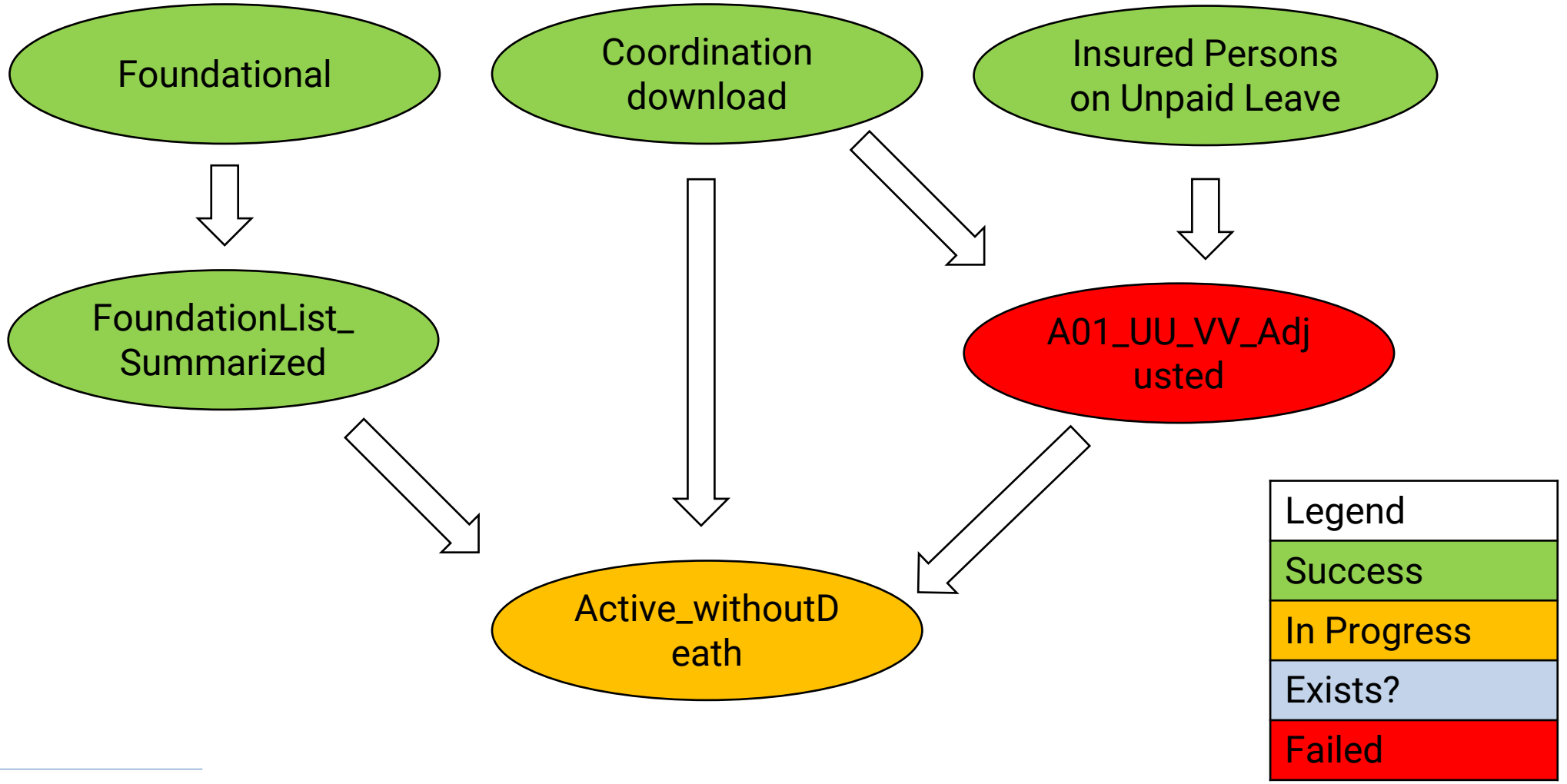
Pipeline implementation: Dependency graphs



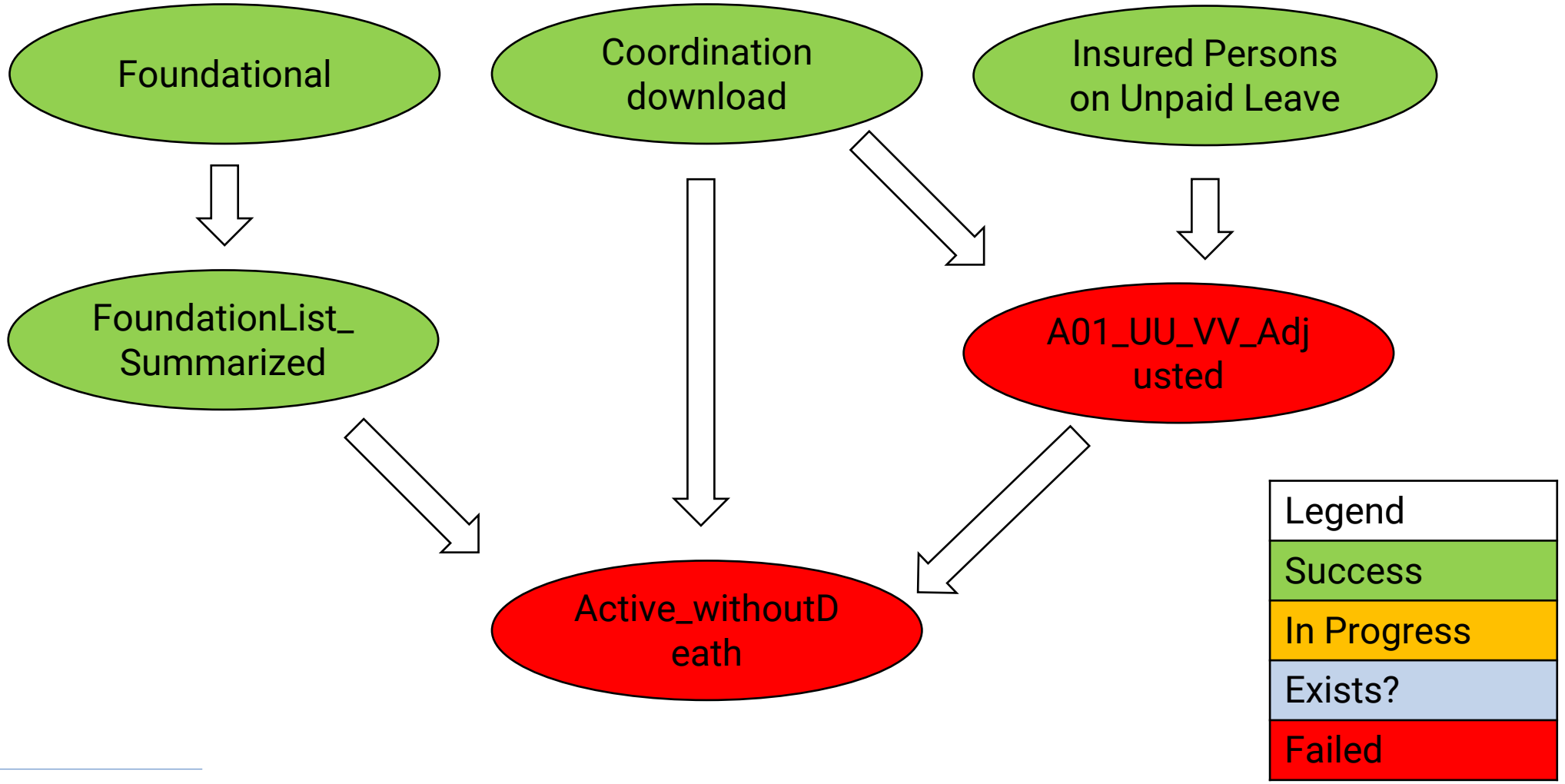
Pipeline implementation: Dependency graphs



Pipeline implementation: Dependency graphs



Pipeline implementation: Dependency graphs



Pipeline implementation: Testing

Log

log_test_date=2023-11-01 00-00-00_execution_date=2024-03-25 10-43-00_local=sql.txt - Notepad

File Edit View

```
Based on test_130 with len(out)>0 cases: There are cases with active individuals that are too old for insurance
Based on test_216 with len(out)>0 cases: There are cases with pensions with missing last payment
Based on test_241b with len(out)>0 cases: There are cases for a child pension with age above 25
Based on test_212 with len(out)>0 cases: There are duplicates in the DK_Renten_gruppiert
Based on test_211 with len(out)>0 cases: There are duplicates in the DK_Renten_Export
```

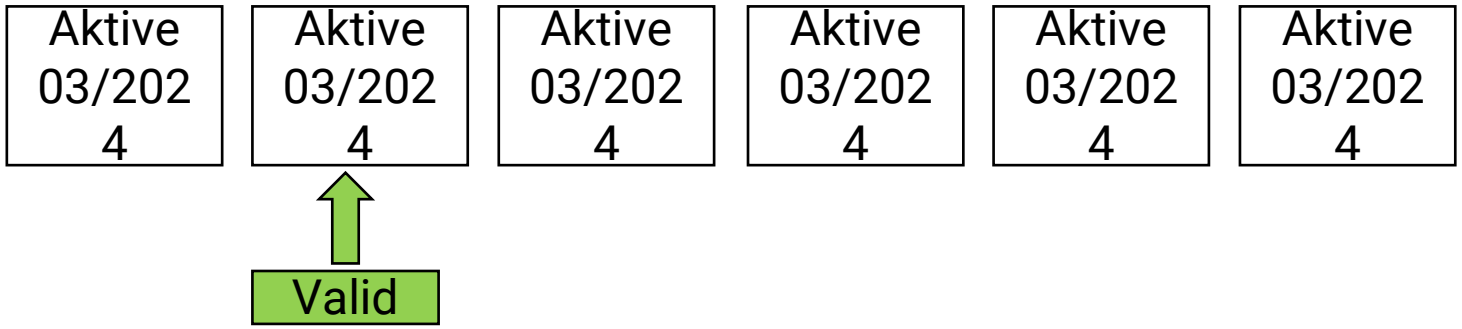
Code

```
if len(out) > 0:
    self.log("There are cases for a child pension with age above 25")
```

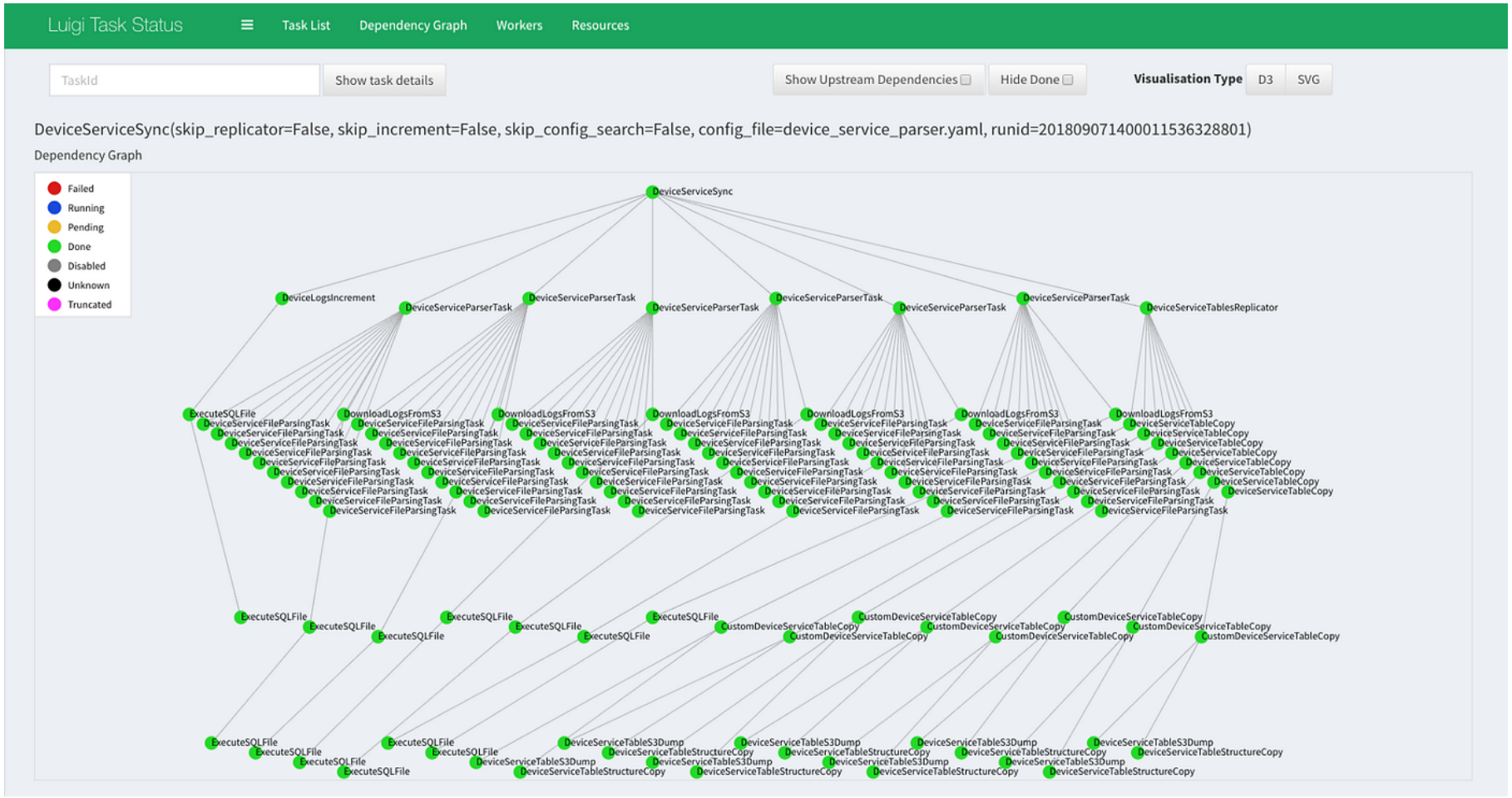
Pipeline implementation: Validation

Some tasks/tests require previous versions of the data (i.e. if running tests for 01/04/2024, may need data from 01/03/2024)

Finalized runs of the data are validated and can be reused for future runs of the pipeline



Pipeline implementation: Demo



Thank you!