# ETL & ELT with SSIS

Day 3 - Recap
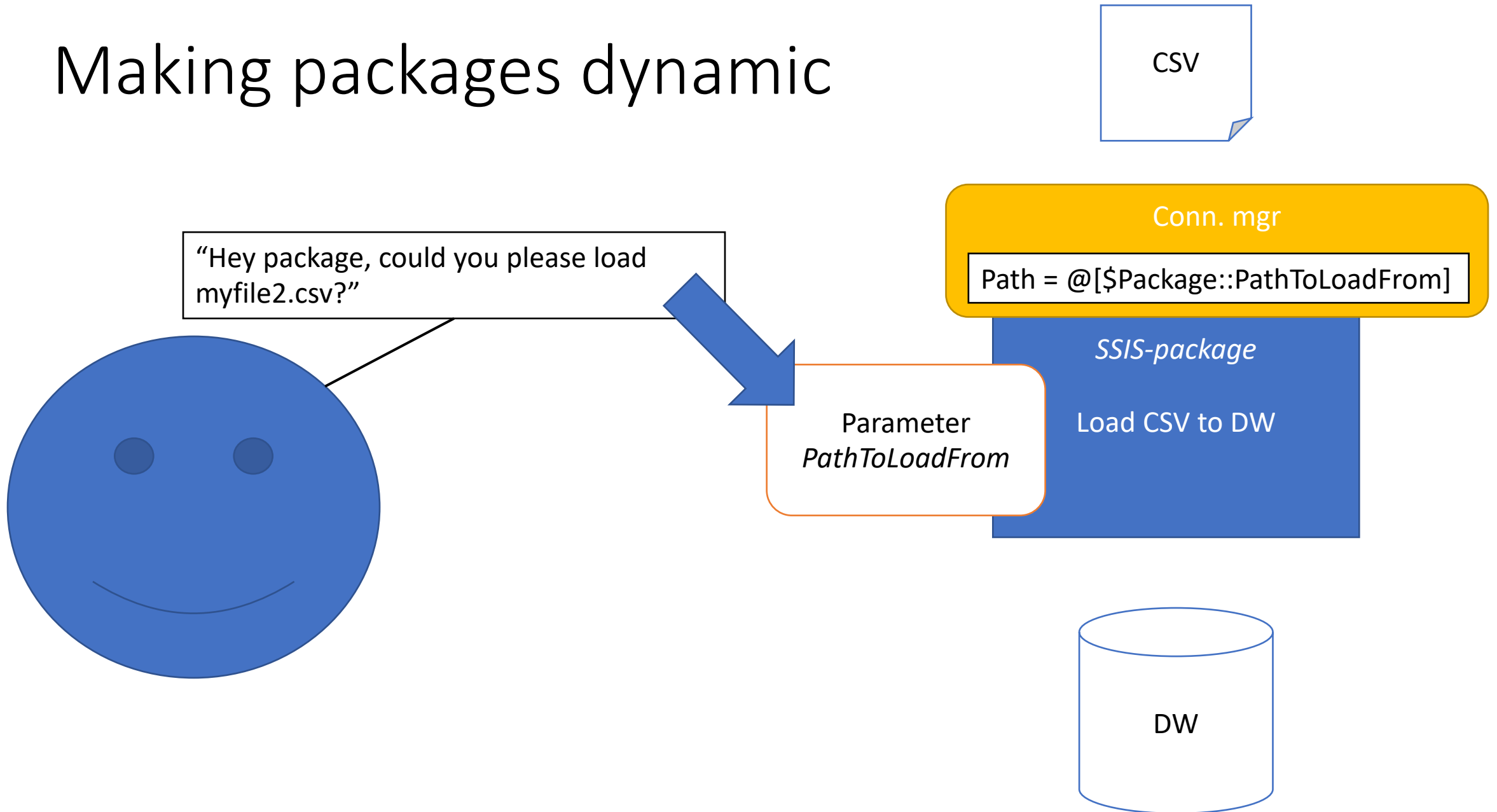
# Subjects of day 2

- Dynamic packages
- The Not-So-Happy flow
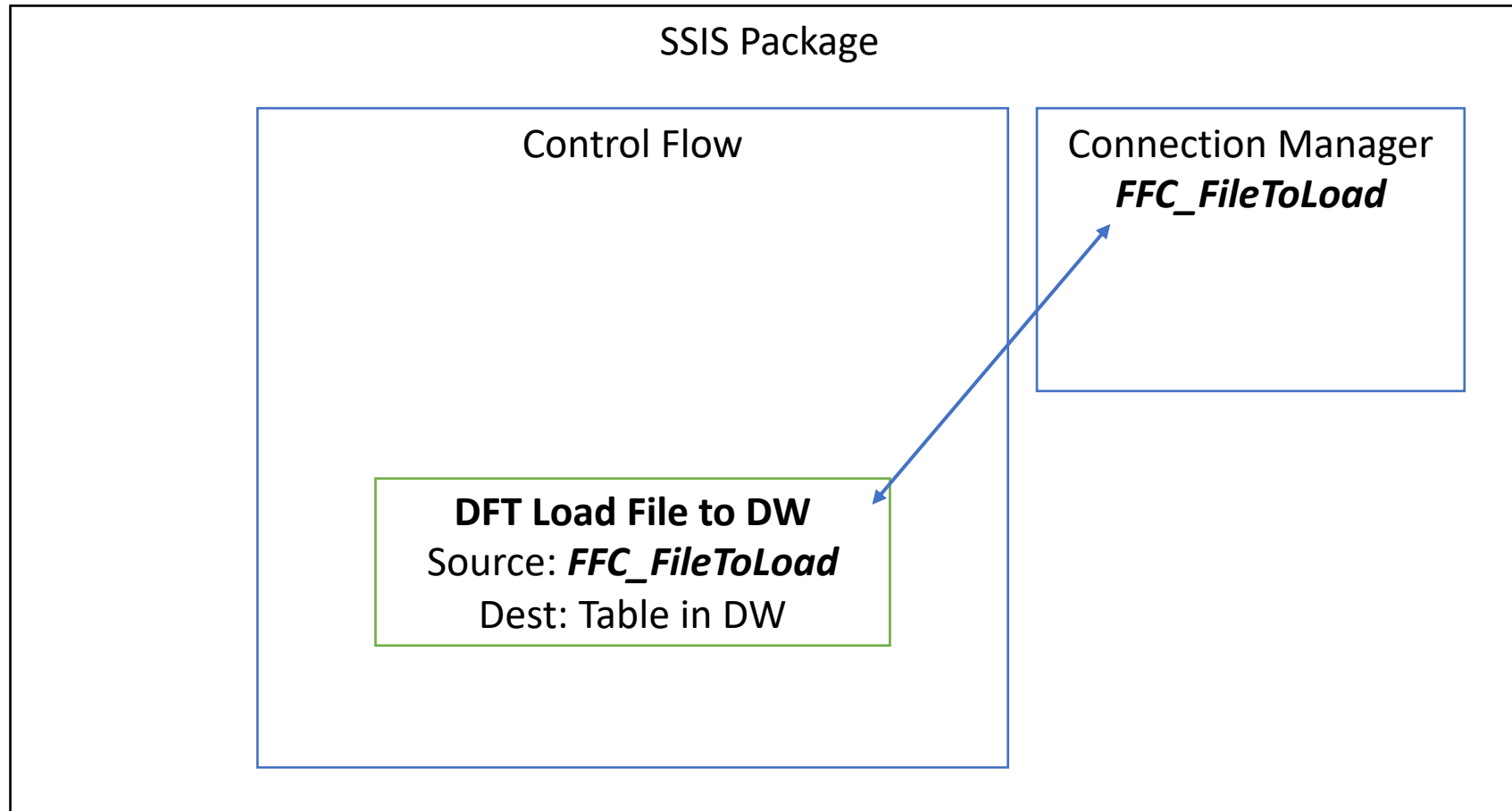
# Dynamic packages
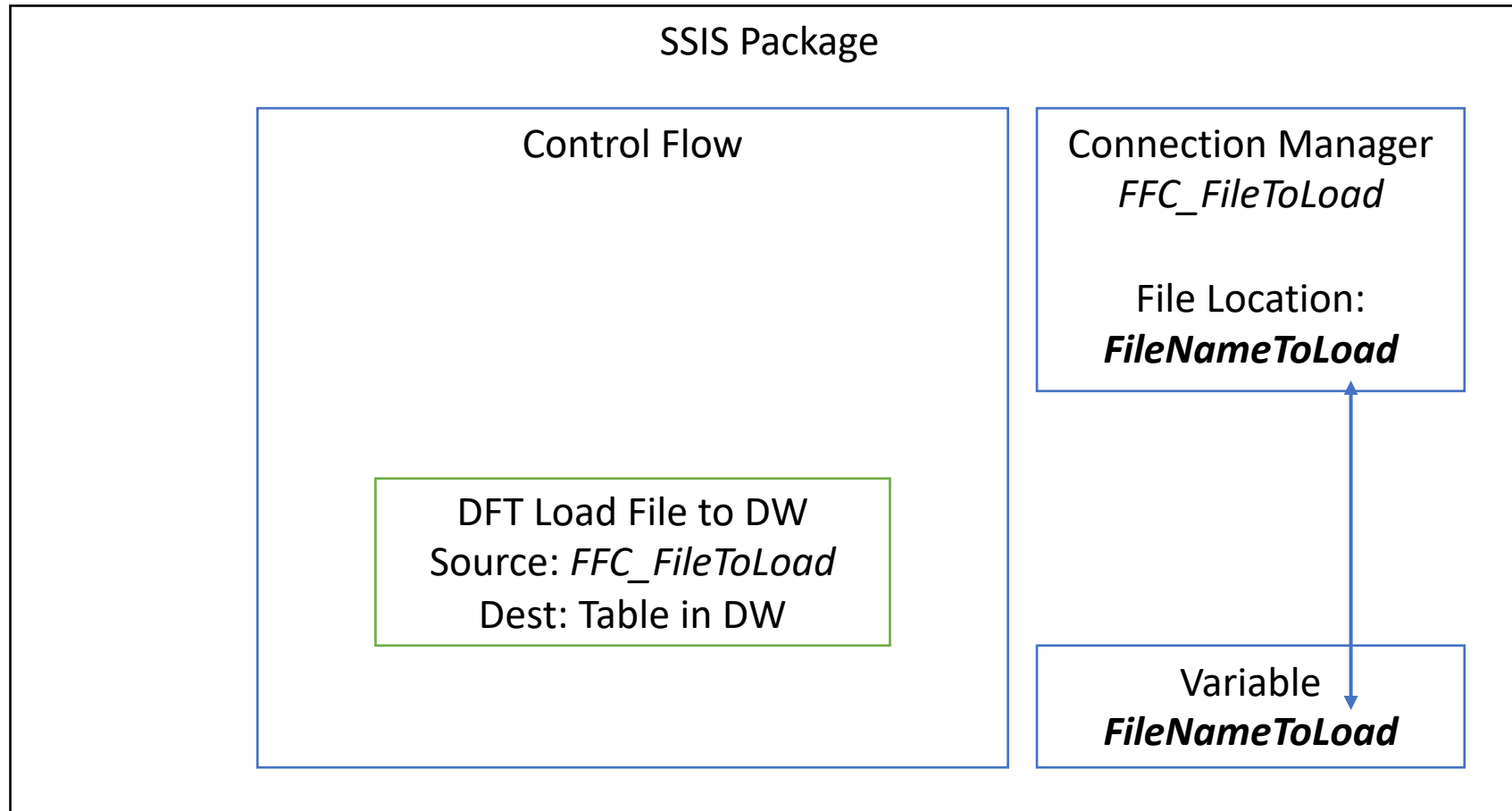
Recap

# Making packages dynamic

CSV

"Hey package, could you please load myfile2.csv?"

Conn. mgr

Path = @[$Package::PathToLoadFrom]

*SSIS-package*

Load CSV to DW

Parameter
*PathToLoadFrom*

DW

# 1. Static Load

# 2. Using a Variable (still static though)

# 3. Looping over files (variable = dynamic)



SSIS Package

Control Flow

Connection Manager
*FFC_FileToLoad*

File Location:
*FileNameToLoad*

ForEach Loop (files; *.csv)
Iterates over Fixed Path (C:\exports)
Puts filenames in **FileNameToLoad**

DFT Load File to DW
Source: *FFC_FileToLoad*
Dest: Table in DW

Variable
**FileNameToLoad**
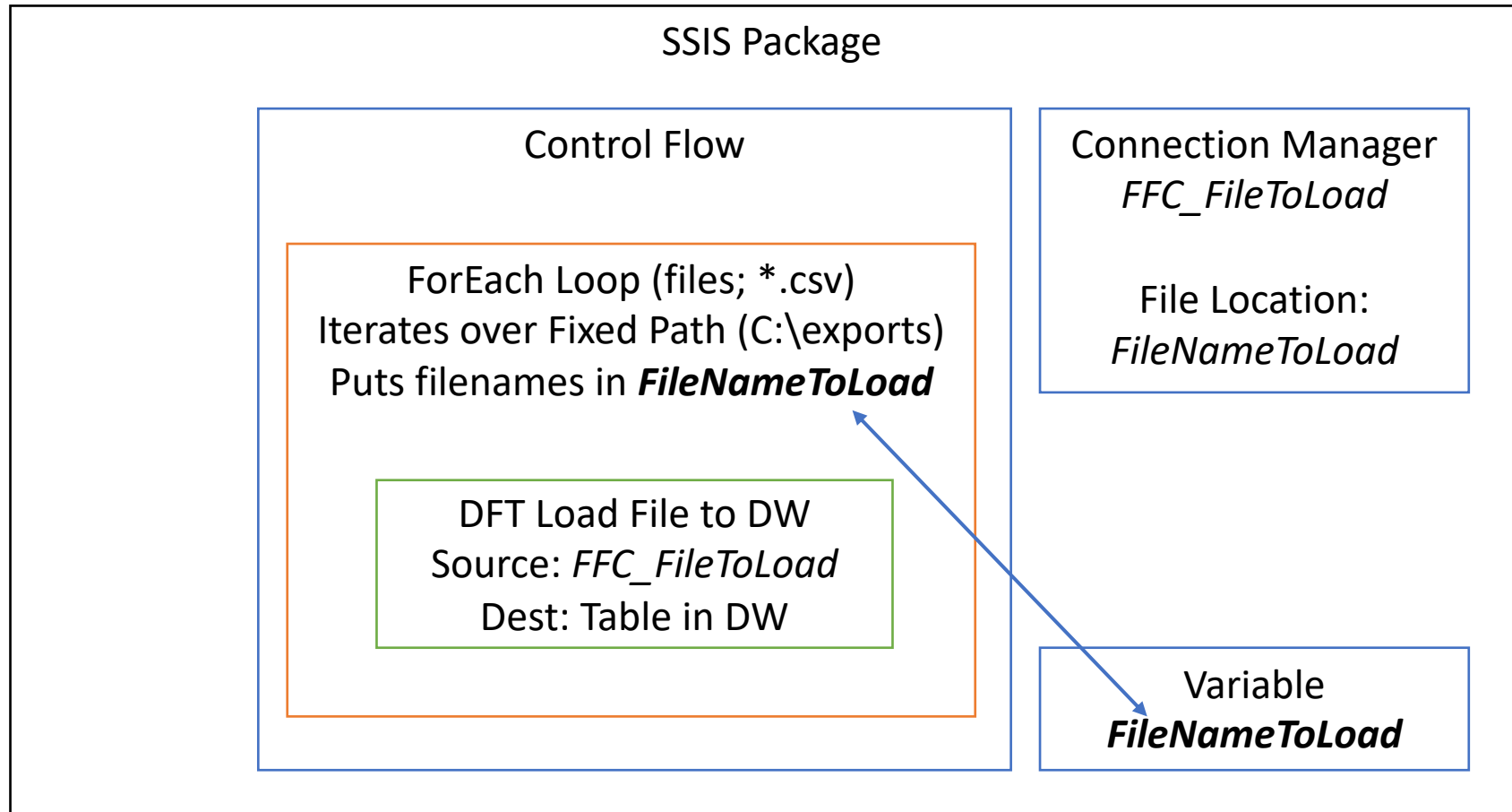
# 4. Set the path to load from at runtime
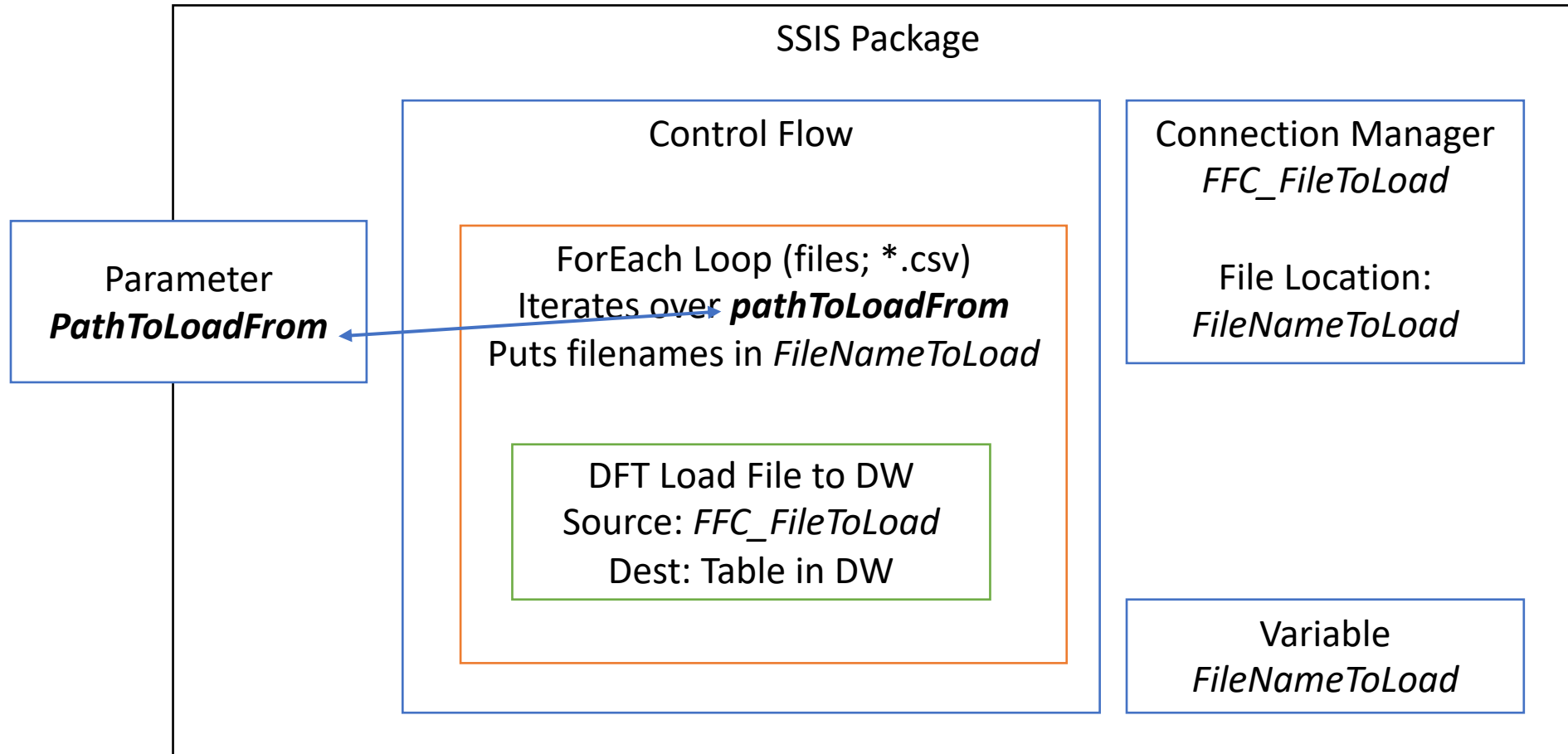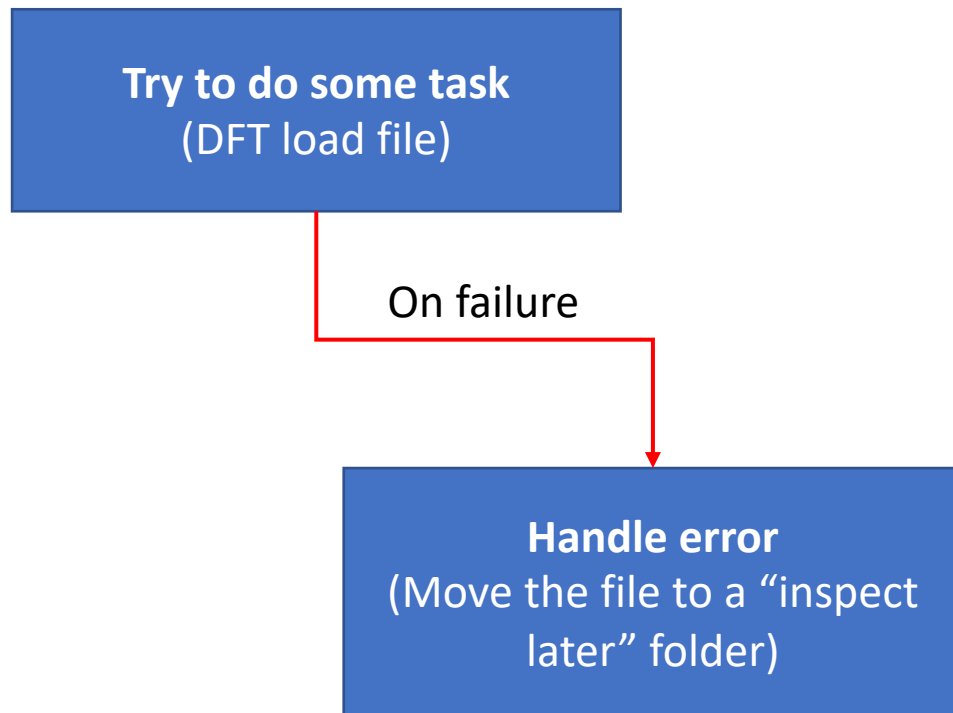
# The Not-So-Happy-flow

Recap

# When loading fails...

- 100 CSV (or XML, Excel, whatever..) files to load
- How to not break the process when one of them is corrupt?
  - Error handling!

# Error handling – Control Flow

**Try to do some task**
(DFT load file)

On failure

**Handle error**
(Move the file to a "inspect later" folder)

When an error occurs:
- Package "fails gracefully"
- Error is handled
  - Files with errors are collected for later inspection
- Package will continue running for non-erroneous files
- Error messages are logged
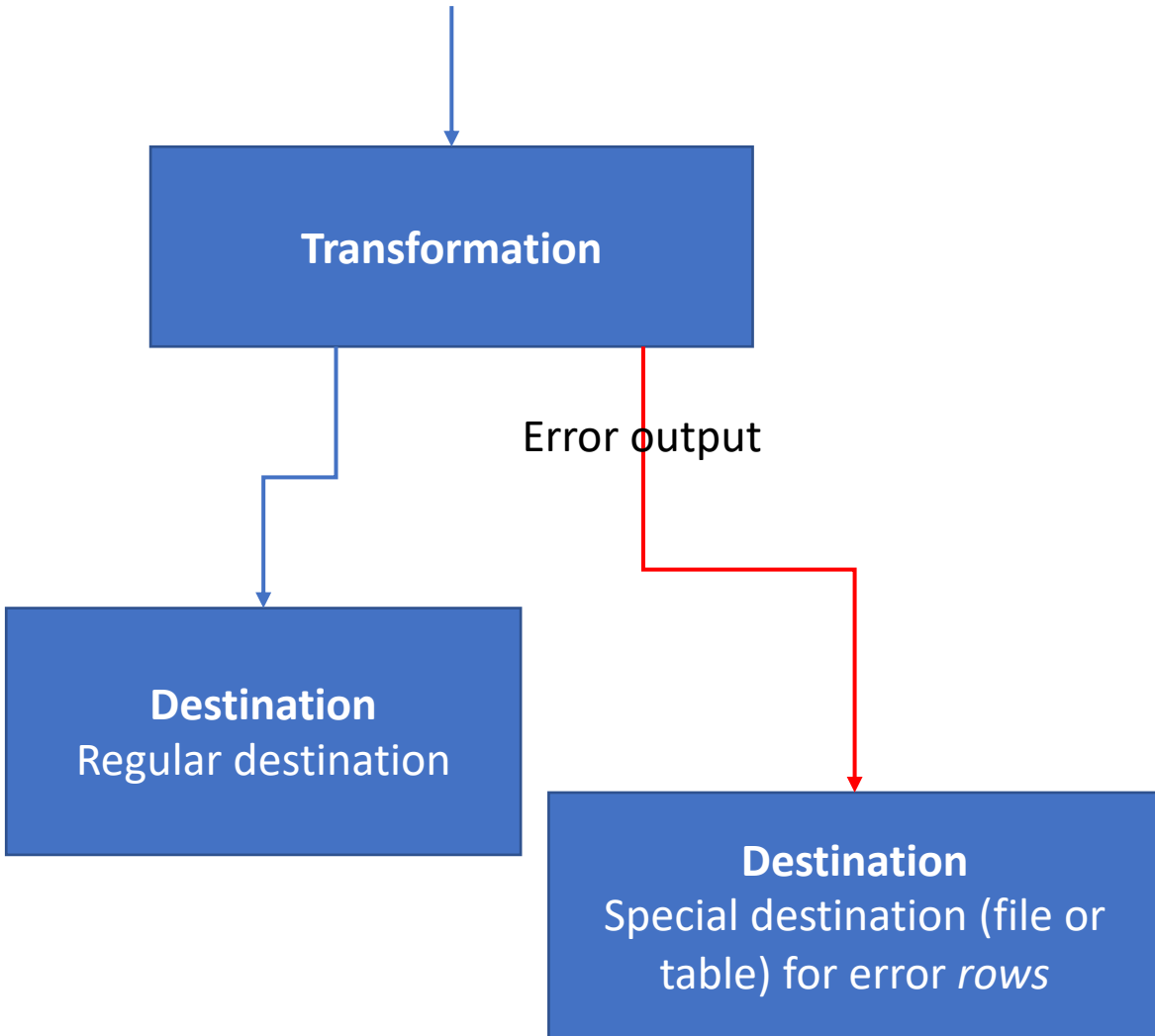- Execution from the package is reported as "failed" (not "succeeded")

# Error handling – Data Flow
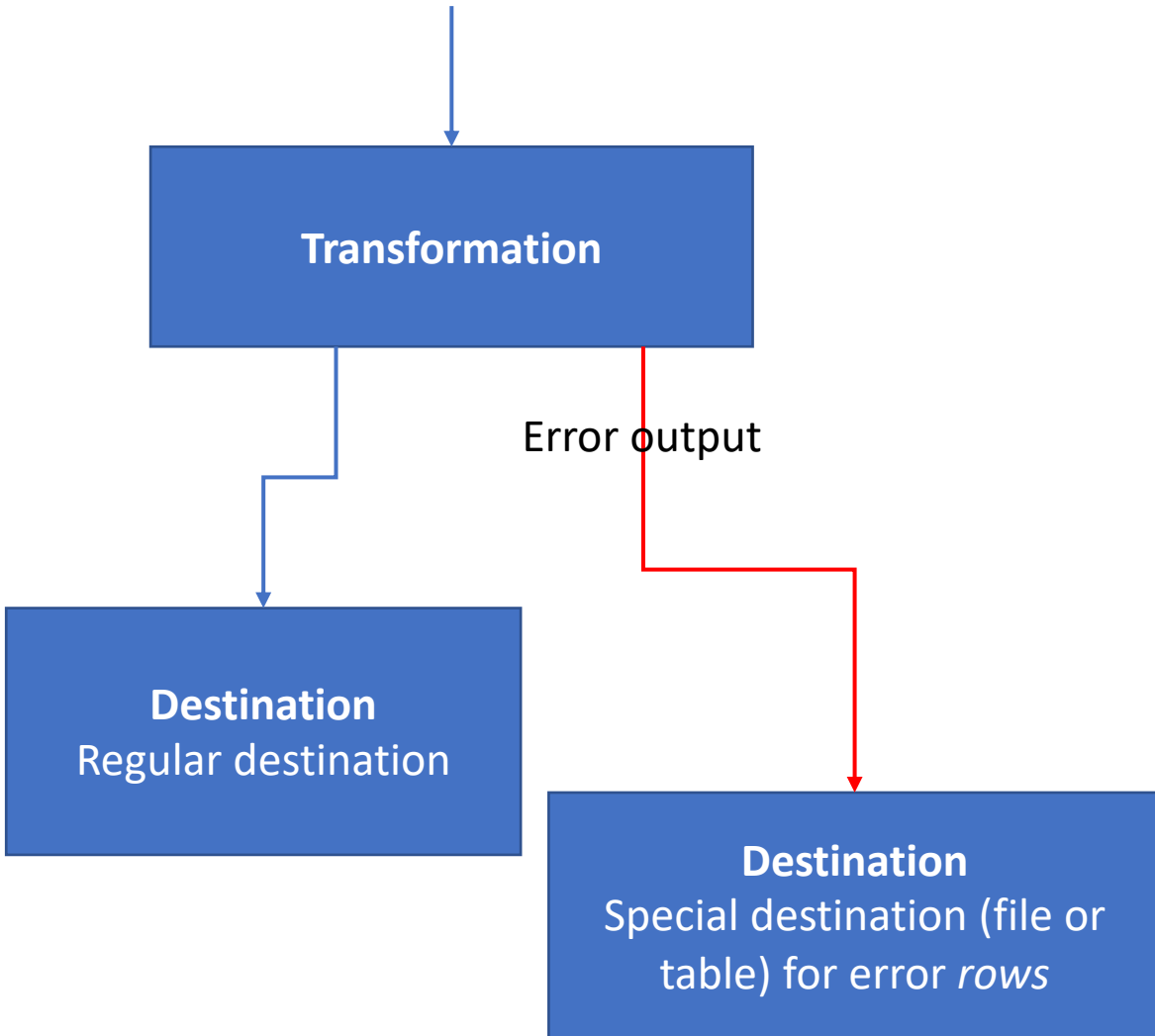


When an error occurs:
- Rows with errors are redirected to another destination
- All "good" rows arrive at the regular destination
- Package will continue running
- Error messages are logged
- Execution from the package is reported as "failed" (not "succeeded")

# Handling other situations– Data Flow

```
        │
        ▼
┌─────────────────────┐
│                     │
│   Transformation    │
│                     │
└─────────────────────┘
    │             │  Error output
    ▼             │
┌──────────────┐  │
│ Destination  │  ▼
│ Regular      │  ┌──────────────────────┐
│ destination  │  │     Destination      │
└──────────────┘  │ Special destination  │
                  │ (file or table) for  │
                  │ error rows           │
                  └──────────────────────┘
```

When something is not an *error* (but just a regular exception to the general rule)
- Split rows based on how you should handle them
- Lookup component (no-match)
- Conditional Split