Ryax Training

From data lakes to data streams

Ph.D. Pedro Velho - Lead Software Engineer @ Ryax Technologies

Introduction

motivation

- Data flows from everywhere
  - Social media
  - Weather data
  - Traffic cameras
  - Demographic data

Introduction
motivation

"...organizations who implemented a Data Lake are outperforming similar companies by 9% in organic revenue growth.." source AWS: What is a data lake?

- Data flows from everywhere
  - Social media
  - Weather data
  - Traffic cameras
  - Demographic data

Introduction
motivation

"...organizations who implemented a Data Lake are outperforming similar companies by 9% in organic revenue growth.." source AWS: What is a data lake?

- Data flows from everywhere
  - Social media
  - Weather data
  - Traffic cameras
  - Demographic data

"...IoT data will make more than 95% of real-time data by 2025 ..." source IBM ebook: Build a better data lake

Introduction

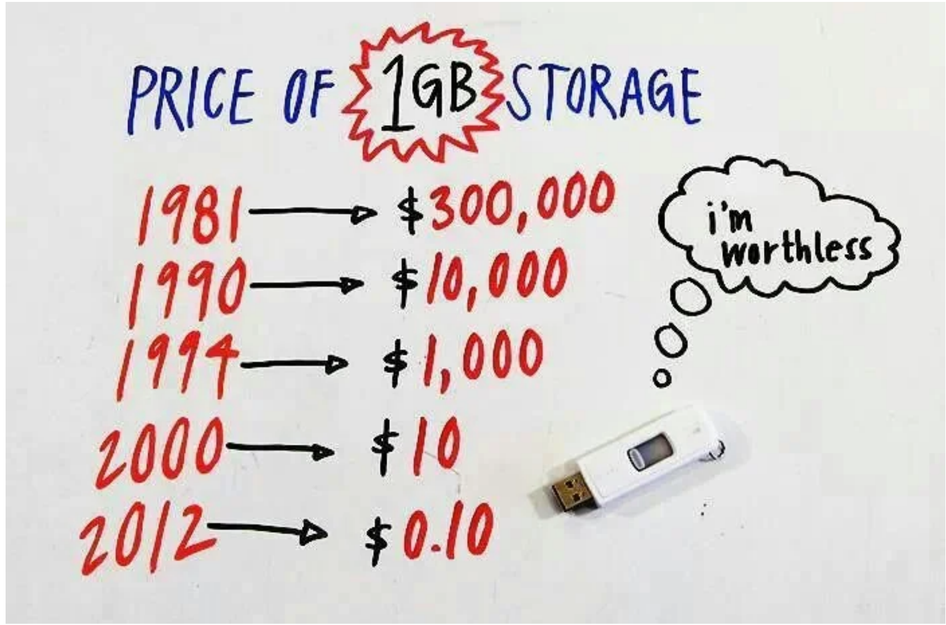
motivation

- In the past data storage was a huge cost

Introduction

motivation

- In the past data storage was a huge cost

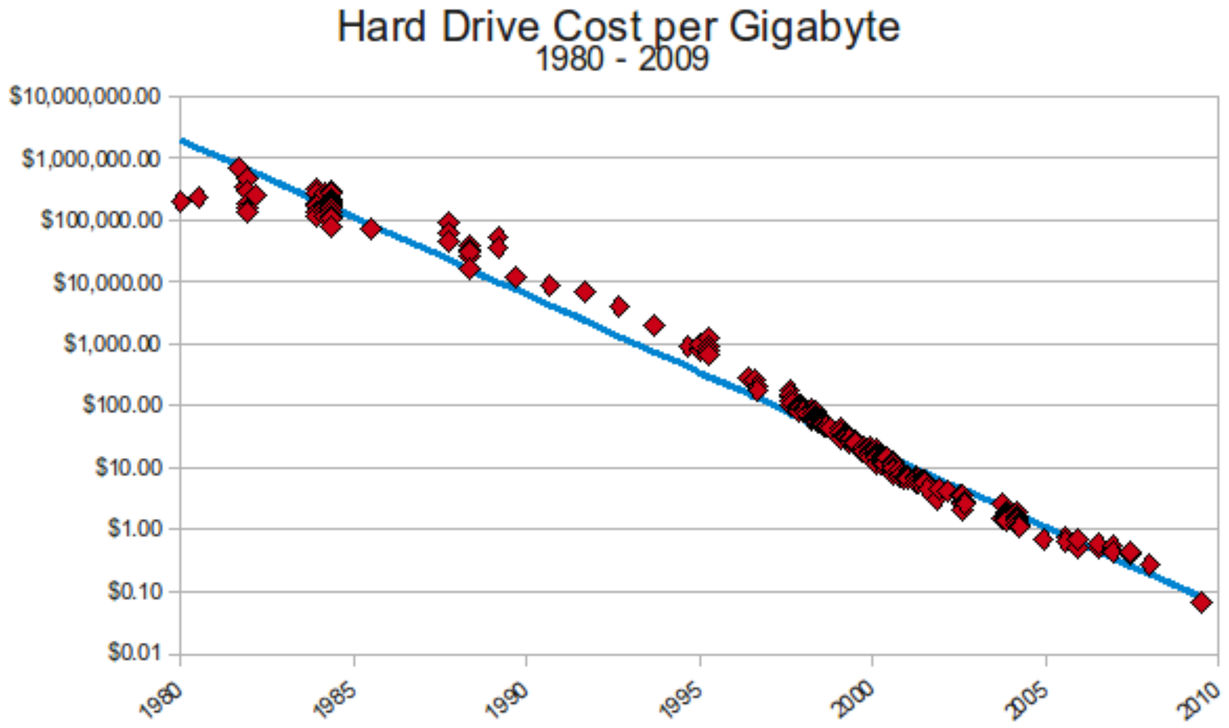


source : reddit post

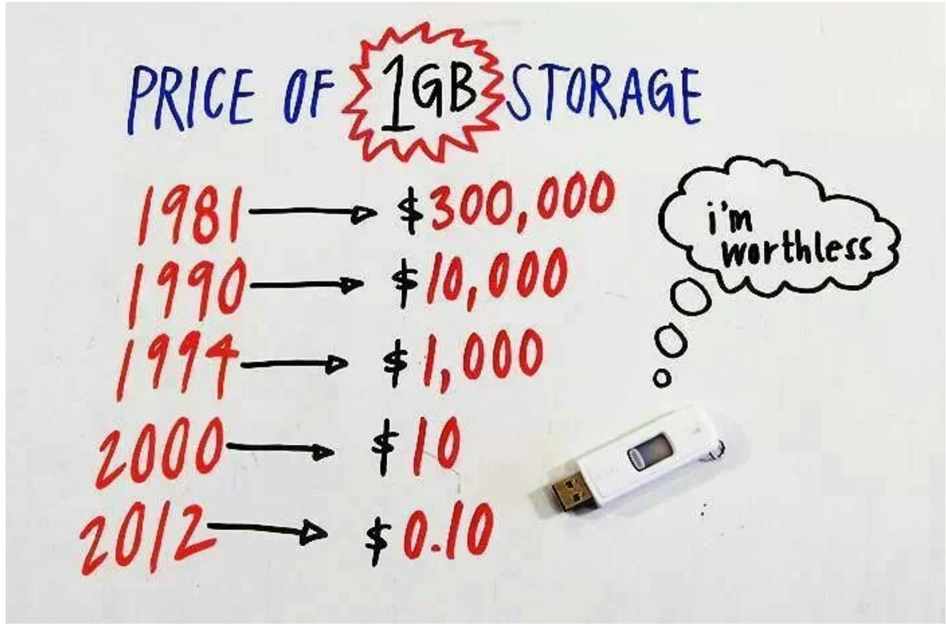
Introduction

motivation

- In the past data storage was a huge cost



source : https://mkomo.com/cost-per-gigabyte



source : reddit post

Introduction

motivation

- In the past to store data was expensive
  - Only store data you use often and is valuable to your business
  - Access data through a well defined schema
- Now data is cheap
  - Store everything you can
  - Often no schema to explore data or cross reference

Introduction

motivation

- Data Warehouses
  - Only store well defined data that means something to your business
  - Structured data, well formed and with stablished relationship
  - Schema-on-write
- Data Lakes
  - Store data as is, a blob of bytes, many formats
  - Schema-on-read : use sophisticated algorithms to read and analyze data

Introduction

Ryax

- A platform for data teams
  - Help to design solutions with focus on data flow
  - Ease deployment on complex infrastructures
  - Makes it easy to plug data sources: API, object storage, SQL
  - Re-use to reduce boiler plate code

Introduction

Ryax

- Automate: multi-node, auto deploy, native error management, resilient to temporary failures
- Observe: monitor infrastructure, application's execution view from data perspective
- Evolve: integrate best dev practices (versioning, modularity, re-use), mutually grow towards new analytics (ML, AI), and infrastructure (cloud, data lake, edge)
- Low code: well defined framework to abstract tedious and boiler plate code, convert ad hoc solution hold by a single individual into company culture

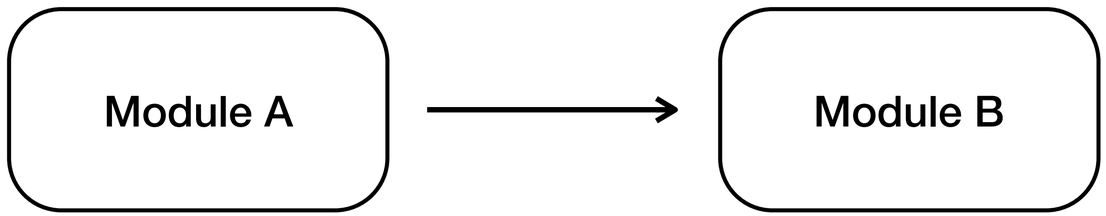
Concepts

introduction

- Workflows
- Modules
  - Sources
  - Processors
  - Publishers
  - Stream operators

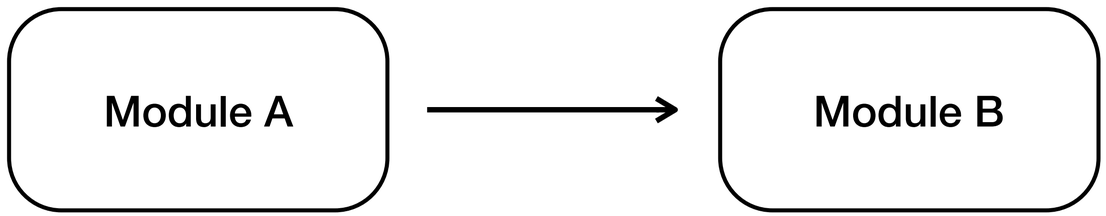
Concepts

workflows



Concepts

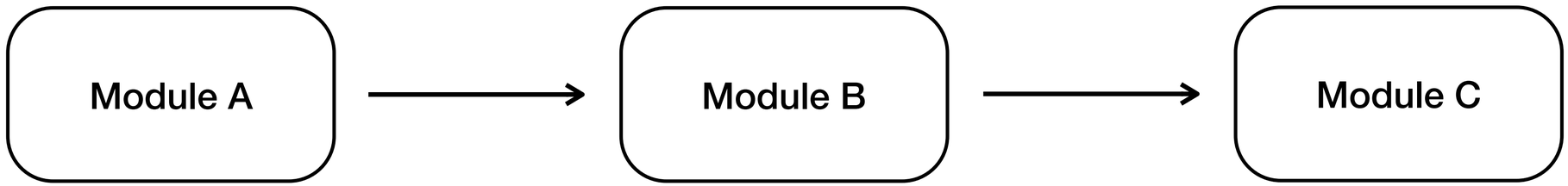
workflows



The arrow means A feed B with data

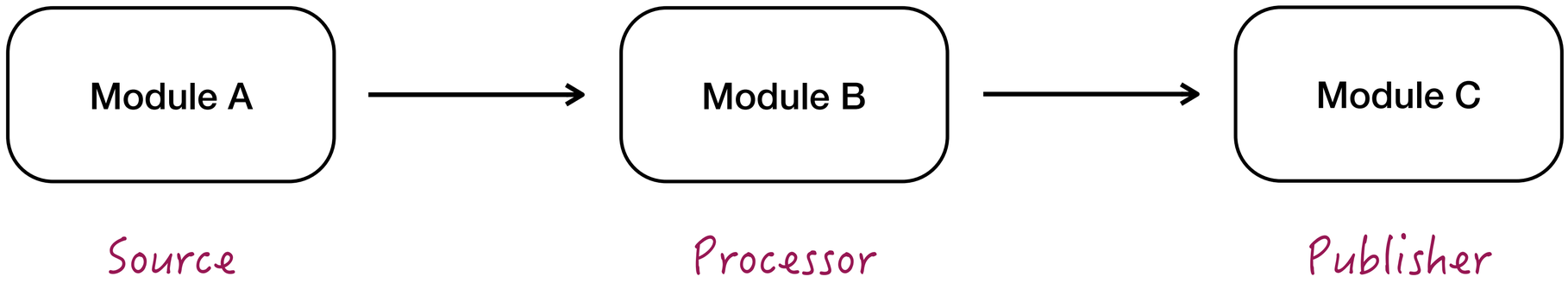
Concepts

workflows



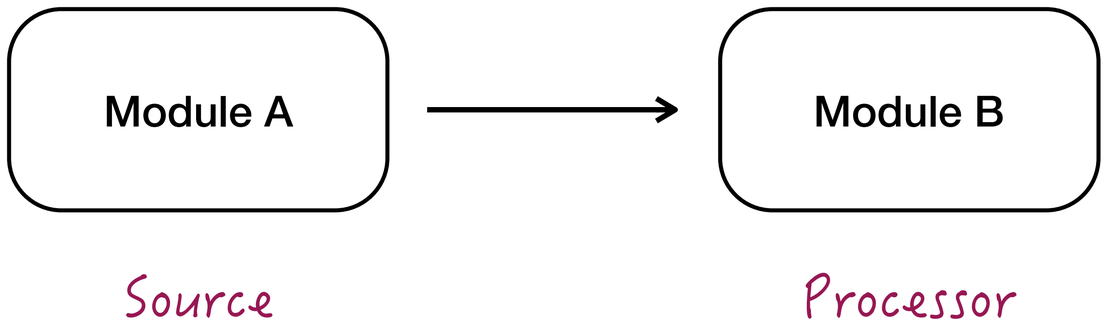
Concepts

workflows



Concepts

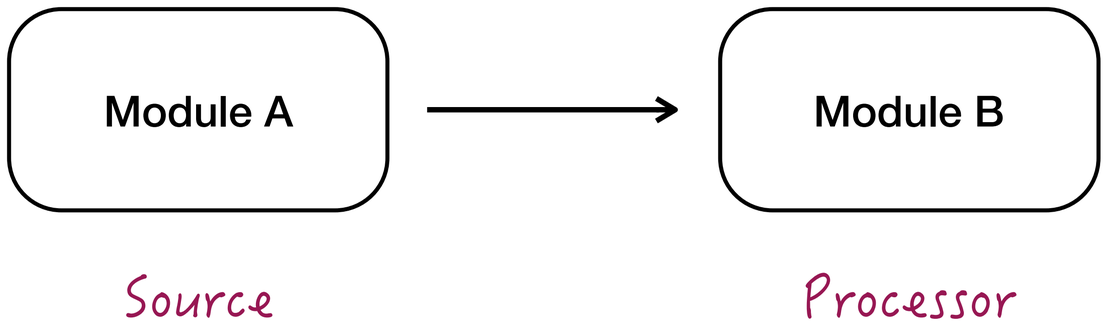
workflows



Concepts

workflows

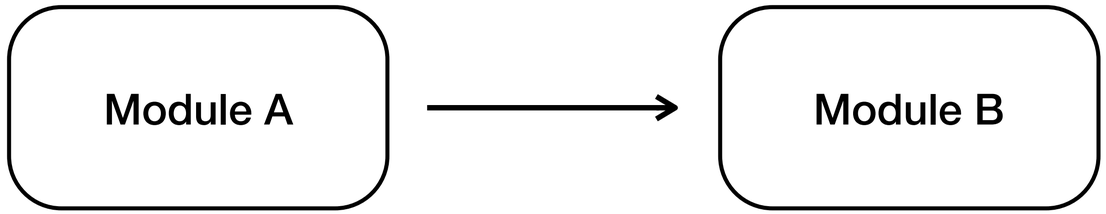
New data arrives: 1



Concepts

workflows

New data arrives: 1



Source

Processor

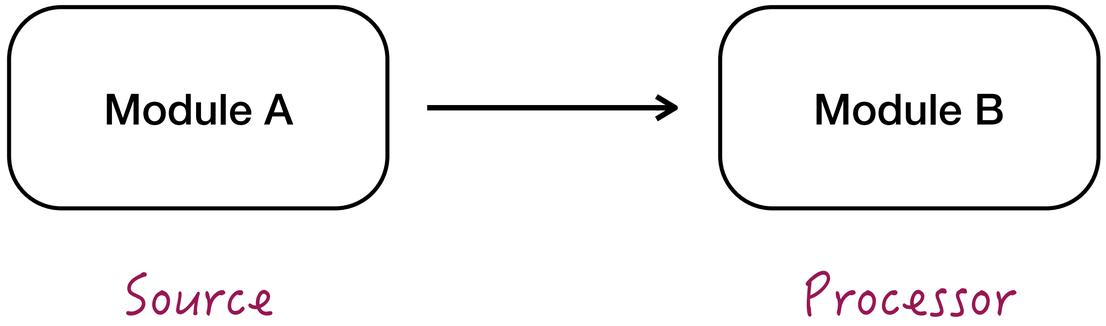
Guarantees that A will process 1 before B process 1

Concepts

workflows

New data arrives: 1

New data arrives: 2

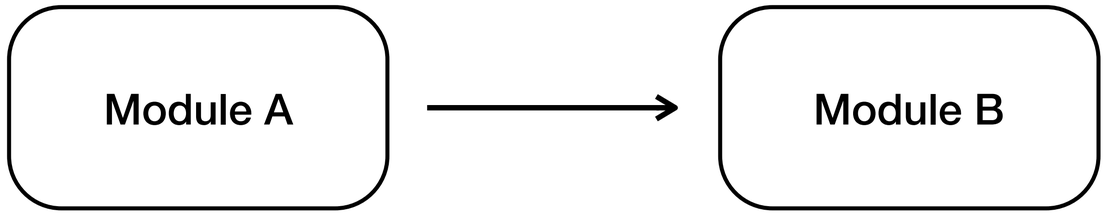


Concepts

workflows

New data arrives: 1

New data arrives: 2



Source

Processor

Guarantees that A will process data before B, but order is not granted

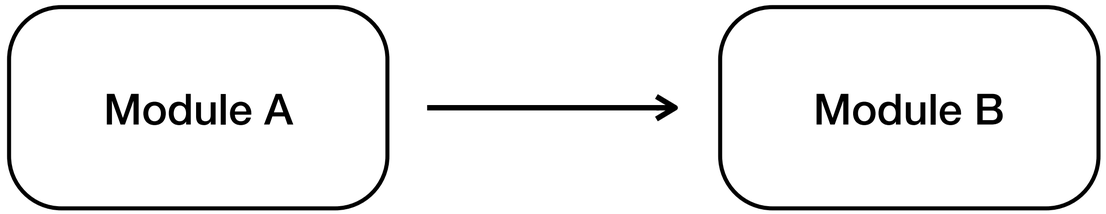
A1, B1, A2, B2

Concepts

workflows

New data arrives: 1

New data arrives: 2



Source

Processor

Guarantees that A will process data before B, but order is not granted

A1, B1, A2, B2

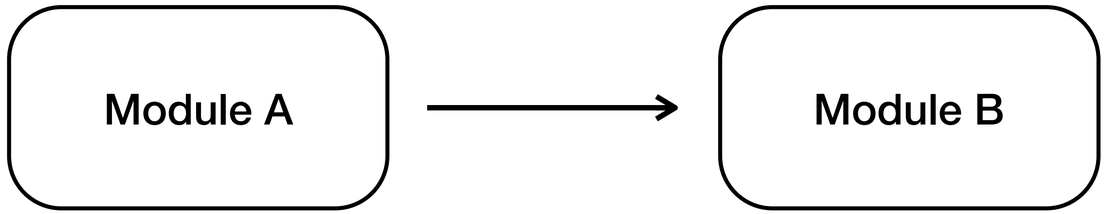
A1, A2, B1, B2

Concepts

workflows

New data arrives: 1

New data arrives: 2



Source

Processor

Guarantees that A will process data before B, but order is not granted

A1, B1, A2, B2

A1, A2, B1, B2

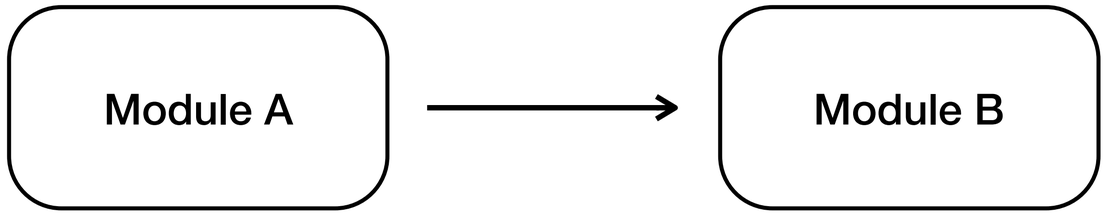
A1, B2, A2, B1

Concepts

workflows

New data arrives: 1

New data arrives: 2



Source

Processor

Guarantees that A will process data before B, but order is not granted

A1, B1, A2, B2

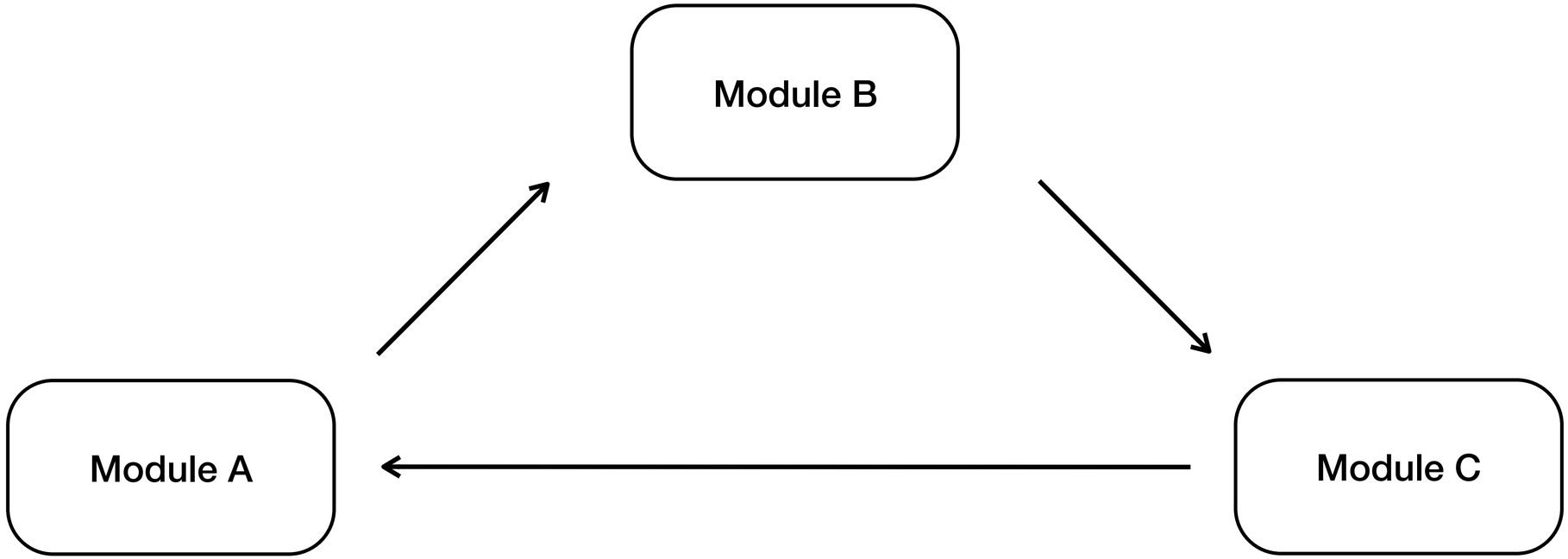
A1, A2, B1, B2

A1, B2, A2, B1

B2 must execute after A2

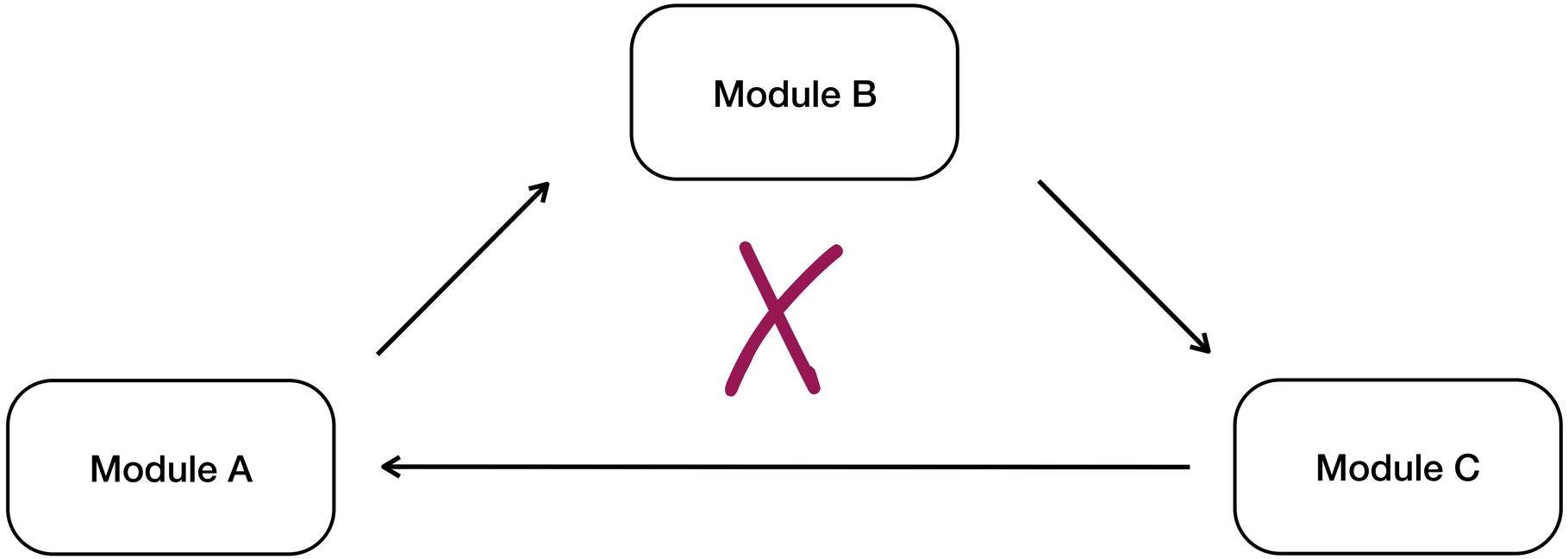
Concepts

workflows



Concepts

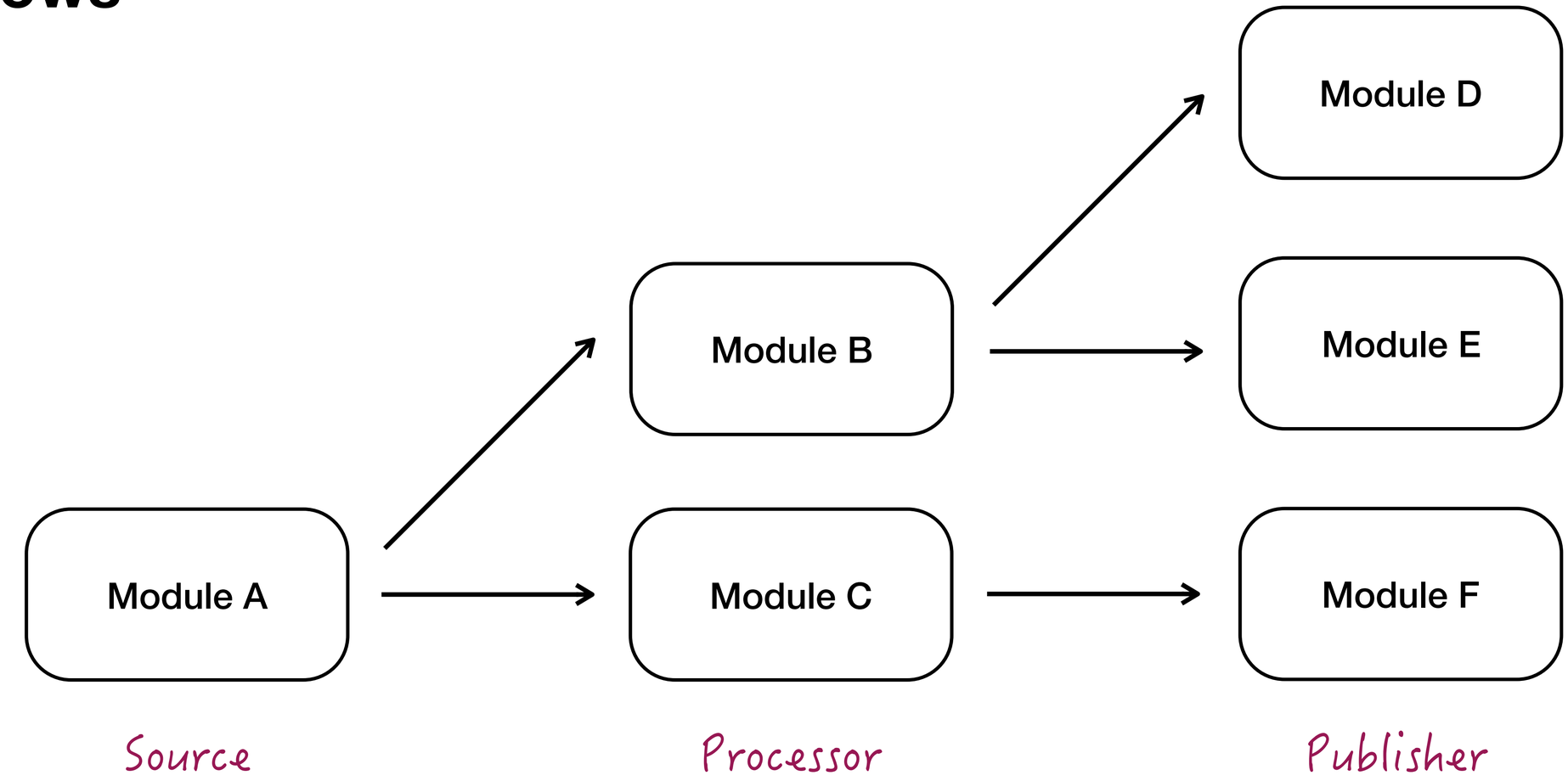
workflows



Cycles are not allowed

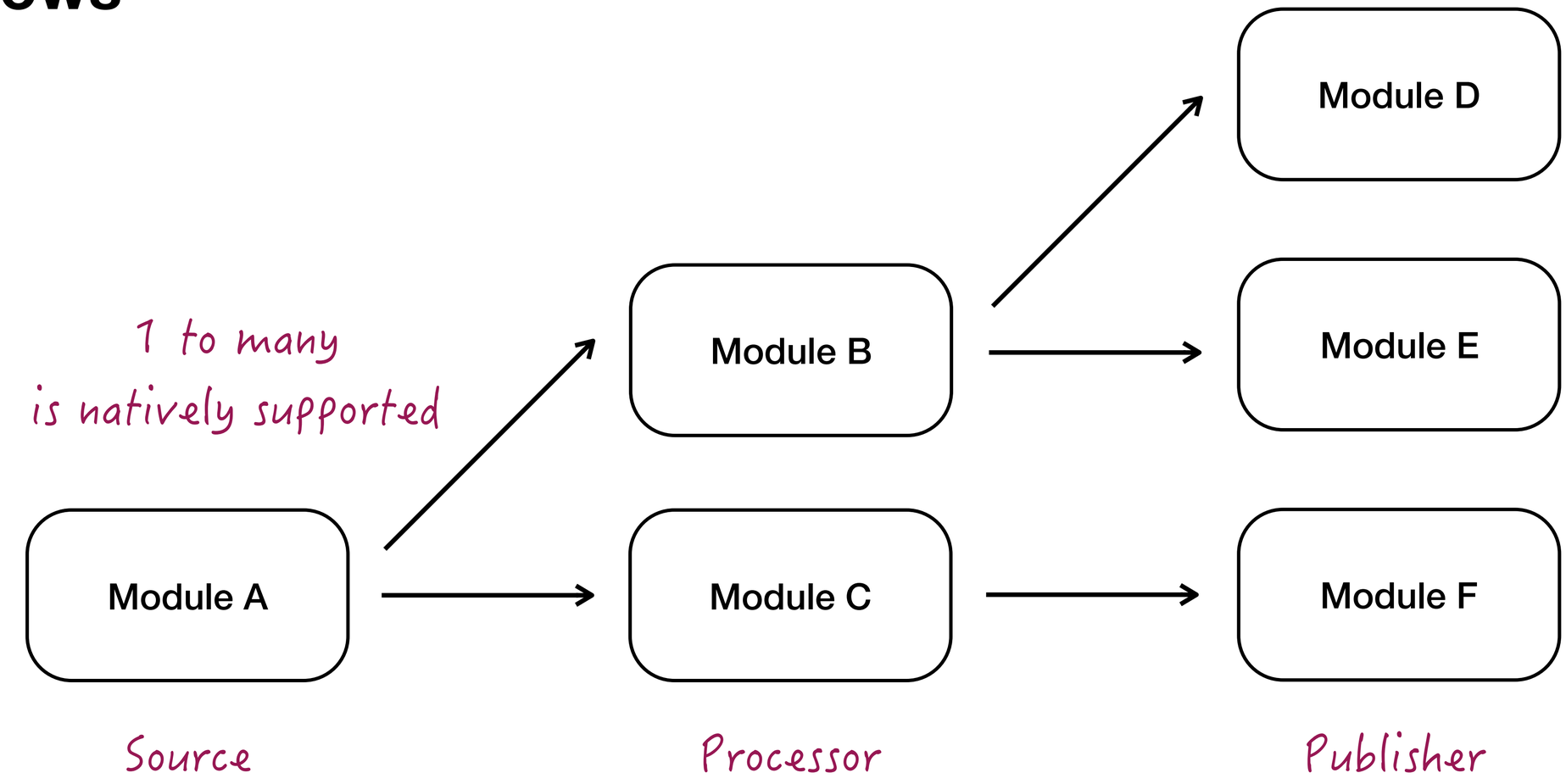
Concepts

workflows



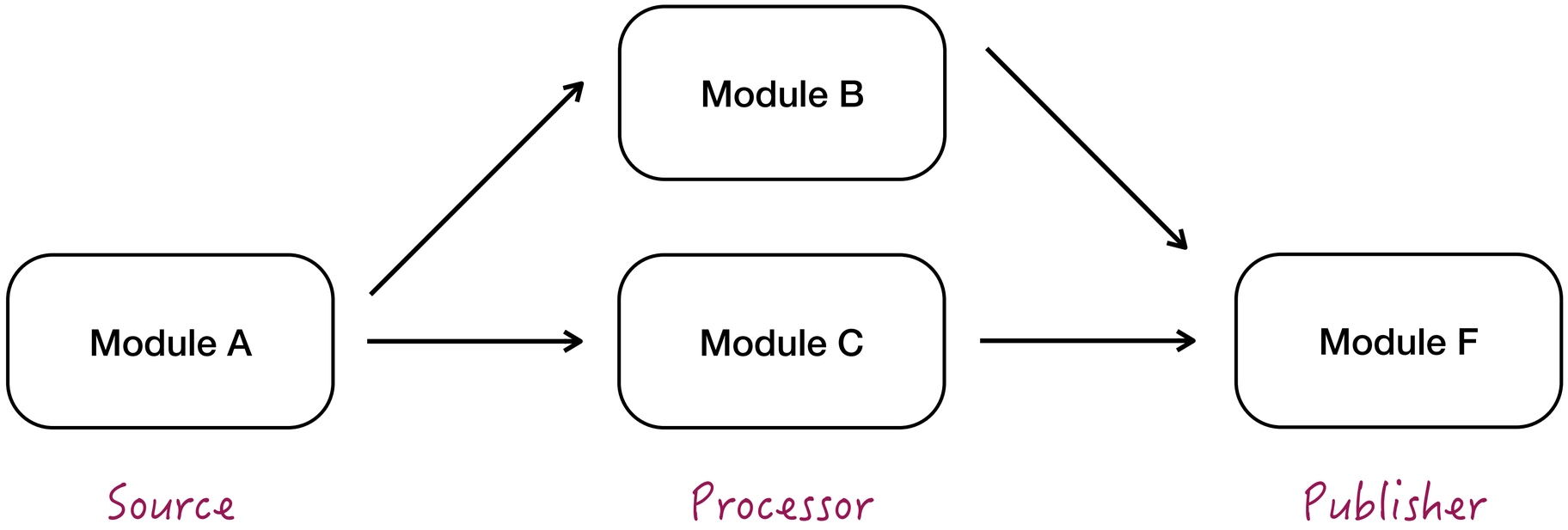
Concepts

workflows



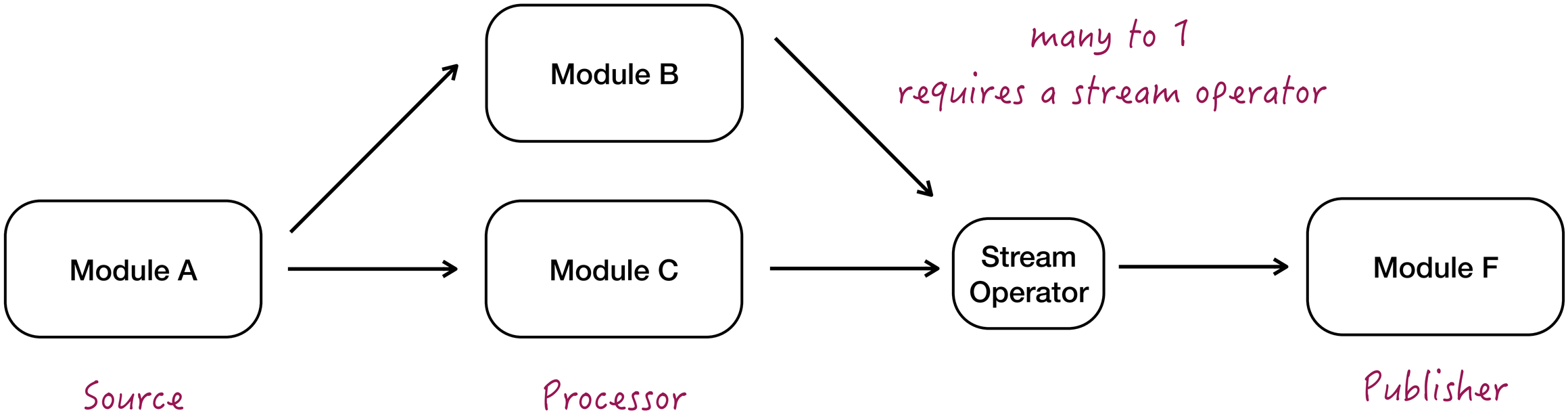
Concepts

workflows



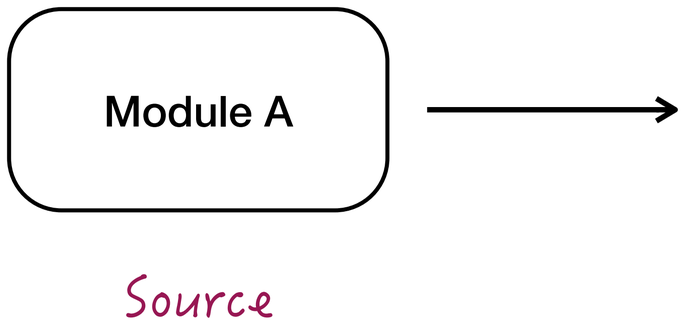
Concepts

workflows



Concepts

sources



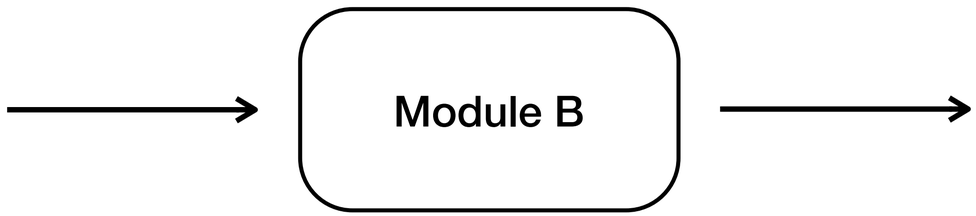
Concepts

sources

- Python3 module to inject external data into Ryax application
- Generate executions each time a new event arrives
- Propagate execution to directly connected modules
- Examples:
  - Retrieve information periodically from external data lake
  - Listen to API requests

Concepts

processors



Processor

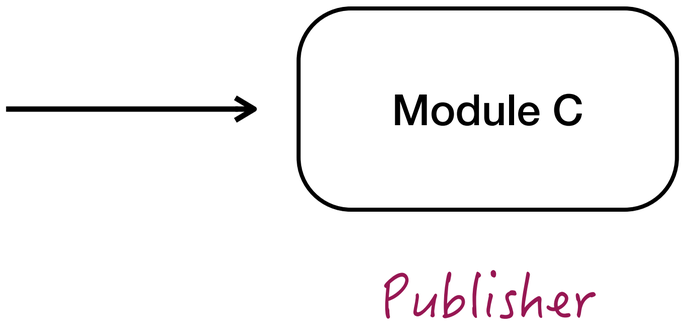
Concepts

processors

- Modules that **do the actual computing**
- Python3 modules with dependencies
- Other dependencies can be installed using nix expression
- Examples:
  - Compute analytical model
  - Use tensorflow for ML or AI
  - Aggregate results

Concepts

publishers



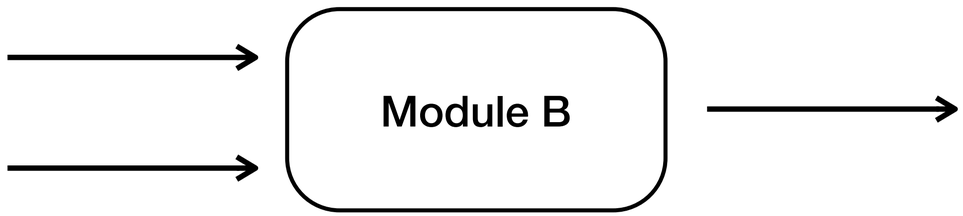
Concepts

publishers

- Python3 module to export results
- Several providers supported: AWS S3, GoogleCloud Storage, Minio
- Examples:
  - Send an e-mail
  - Export file to S3
  - Export blob of data to datalake

Concepts

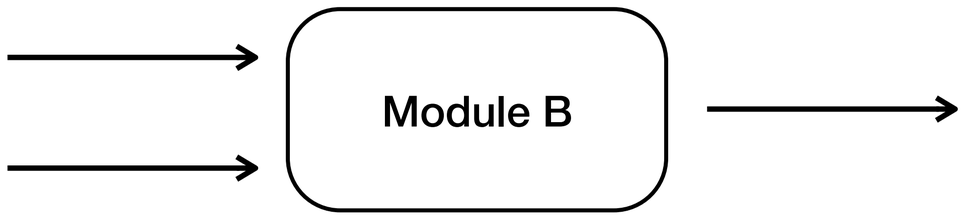
stream operators



Stream Operator

Concepts

stream operators



Stream Operator

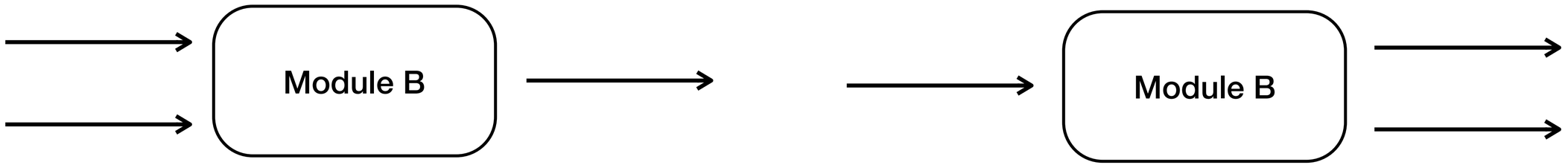
many to 1

select or combine data

enable multiple sources

Concepts

stream operators

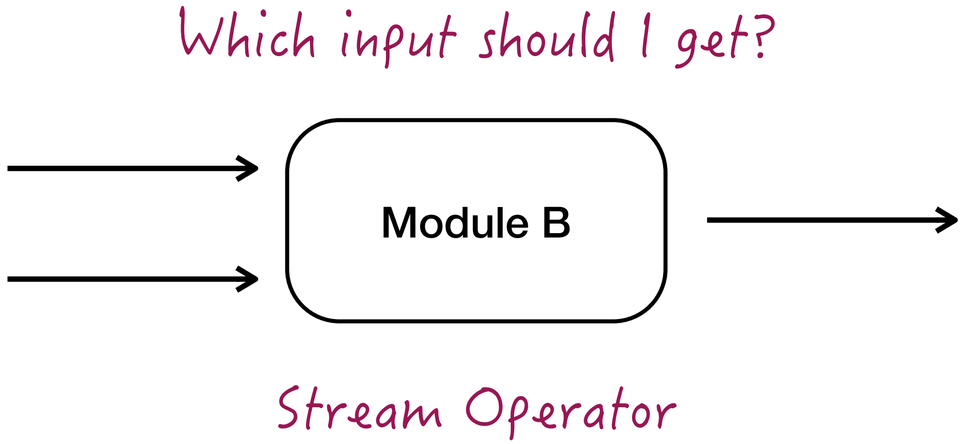


many to 1

1 to many

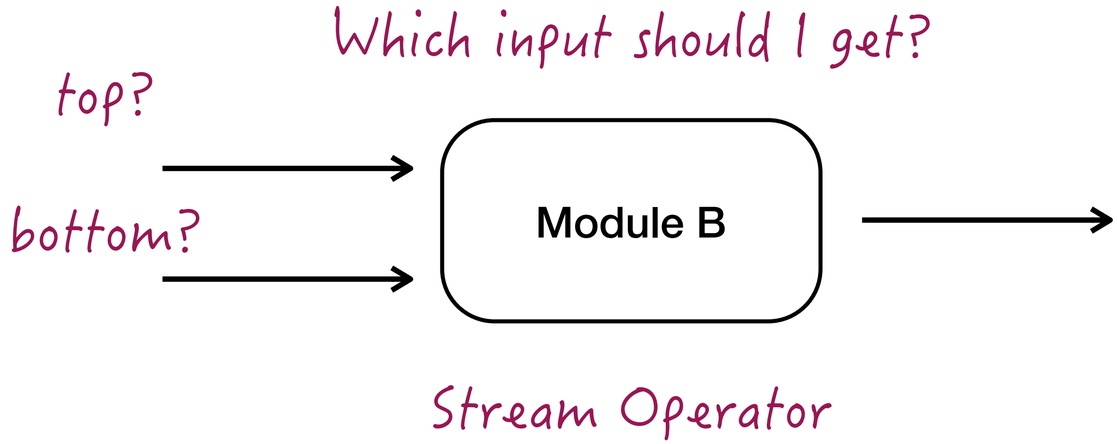
Concepts

stream operators



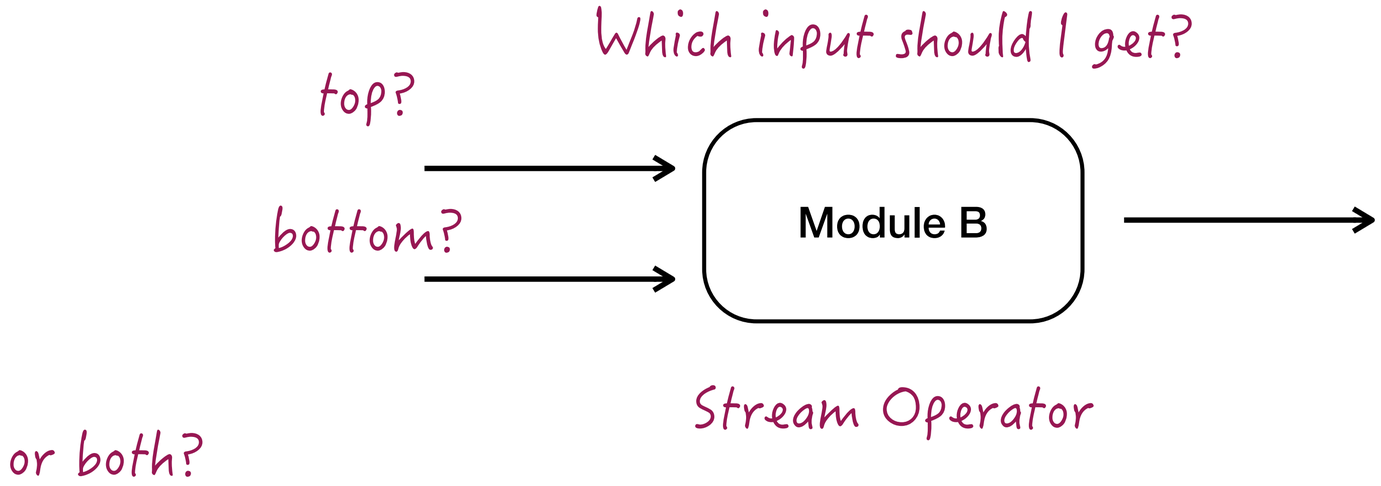
Concepts

stream operators



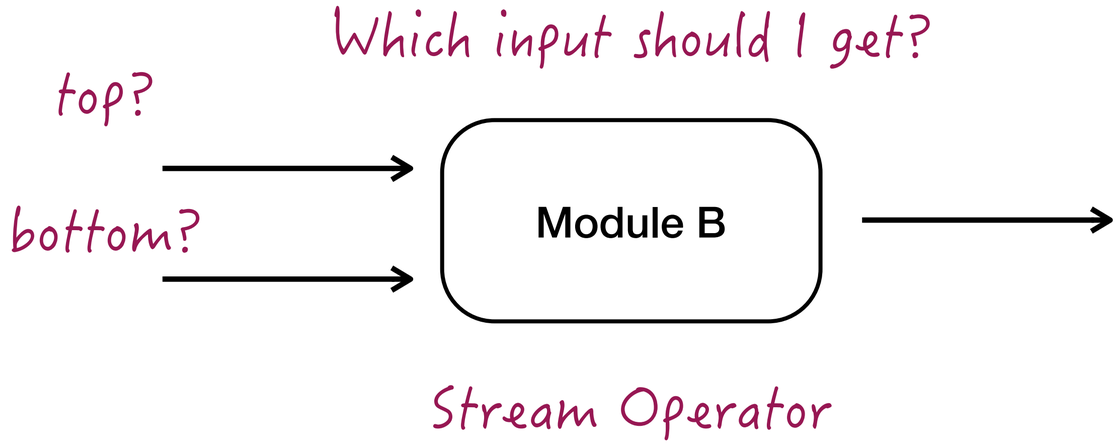
Concepts

stream operators



Concepts

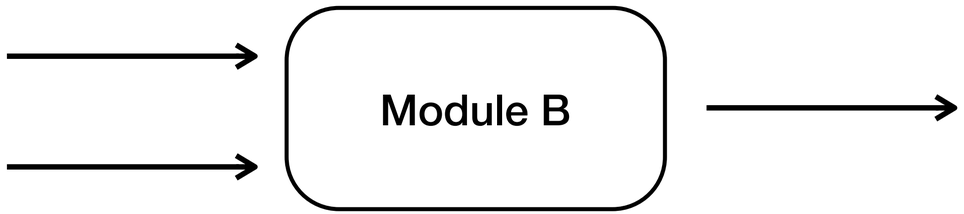
stream operators



Concepts

stream operators

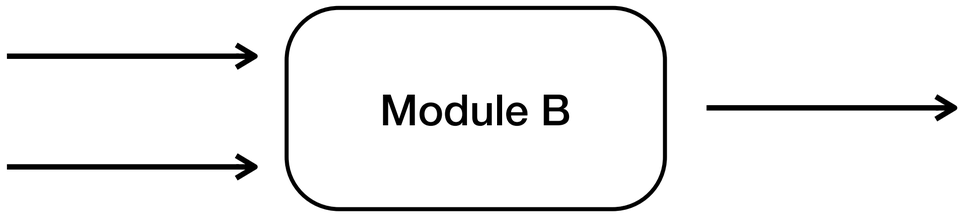
- For instance **withlatestfrom:**



Concepts

stream operators

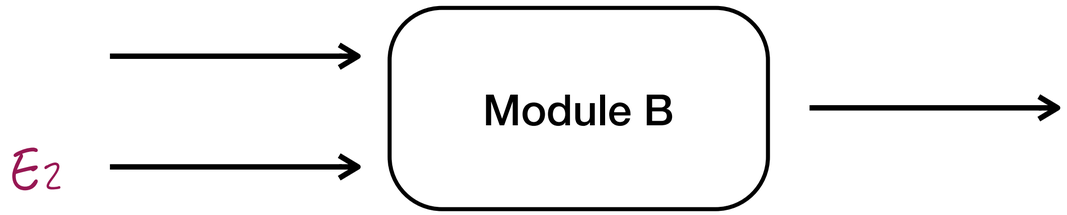
- For instance **withlatestfrom:**
  - Wait for both events to arrive E_1 and E_2



Concepts

stream operators

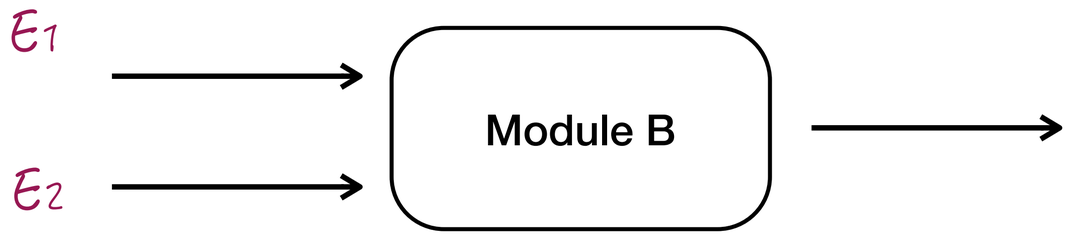
- For instance **withlatestfrom:**
  - Wait for both events to arrive E_1 and E_2



Concepts

stream operators

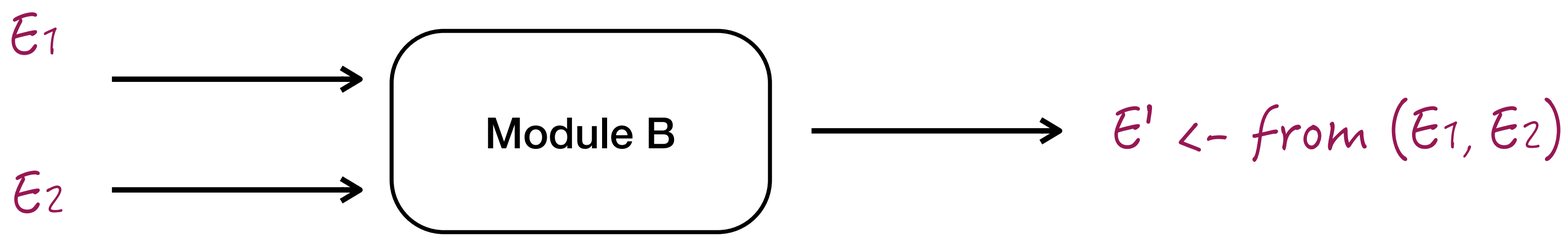
- For instance **withlatestfrom:**
  - Wait for both events to arrive E_1 and E_2



Concepts

stream operators

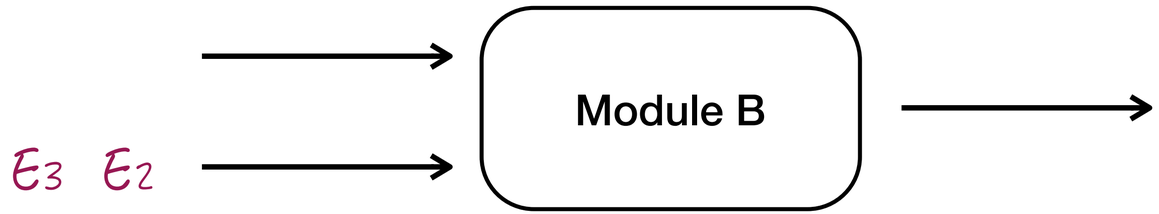
- For instance **withlatestfrom:**
  - Wait for both events to arrive E_1 and E_2



Concepts

stream operators

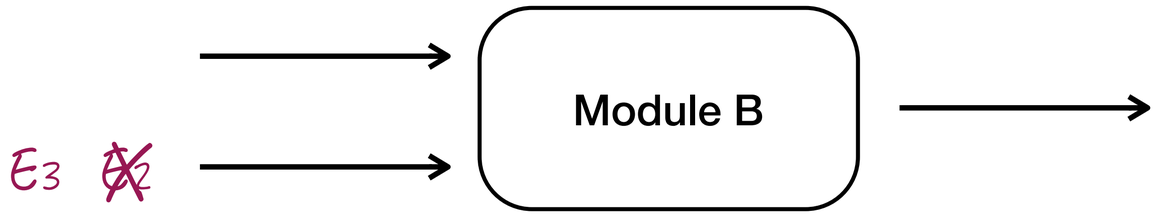
- For instance **withlatestfrom:**
  - Wait for both events to arrive E_1 and E_2
  - If a new event E_3 arrives before E_1, replace with latest



Concepts

stream operators

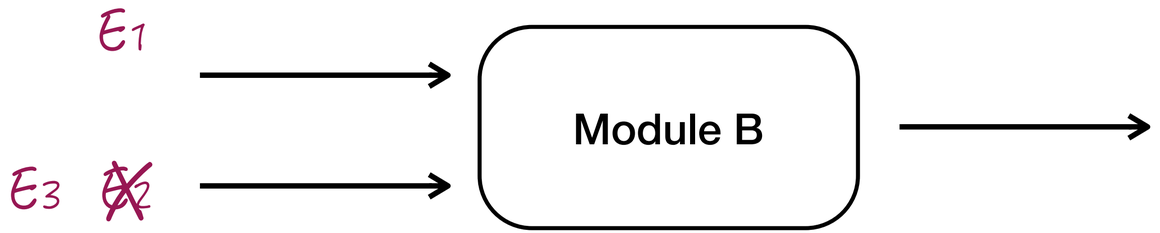
- For instance **withlatestfrom:**
  - Wait for both events to arrive E_1 and E_2
  - If a new event E_3 arrives before E_1, i.e. E_2 is lost



Concepts

stream operators

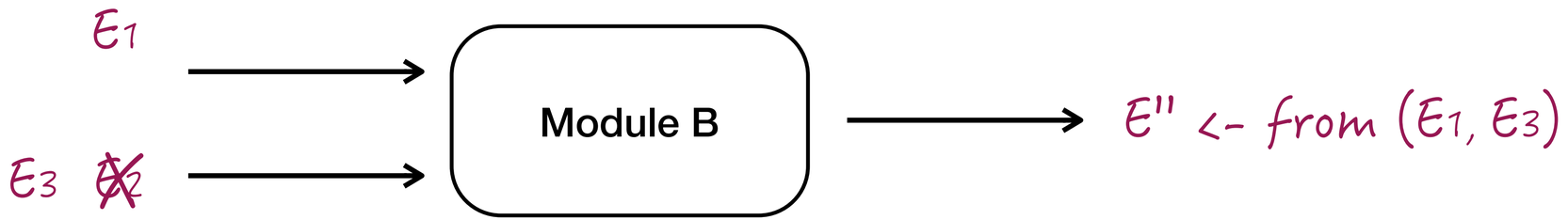
- For instance **withlatestfrom:**
  - Wait for both events to arrive E_1 and E_2
  - If a new event E_3 arrives before E_1, i.e. E_2 is lost



Concepts

stream operators

- For instance **withlatestfrom:**
  - Wait for both events to arrive E_1 and E_2
  - If a new event E_3 arrives before E_1, i.e. E_2 is lost



Executions

introduction

- Data generate events that generate data
- Events change the status of deployed modules: submitted, running, ran, done
- Executions show detailed information on data and module reactions
- Can easily track which data produced which result
- Show logs for produced results