Exploiting Apache Flink's Stateful Operators

Gijsbert van Vliet & Olga Slenders

Berlin • 4 September 2018

ING

Agenda

- About ING
- About the Use Case
- Stateful example + code
- Advanced stateful example + code
- Conclusion

About ING

About ING

Almost 9 million customers in the Netherlands*

9 million transactions per month on ING machines*

1,45 million customer logins per day on Mijn ING website*

4,2 million customer logins per day on ING Banking App*

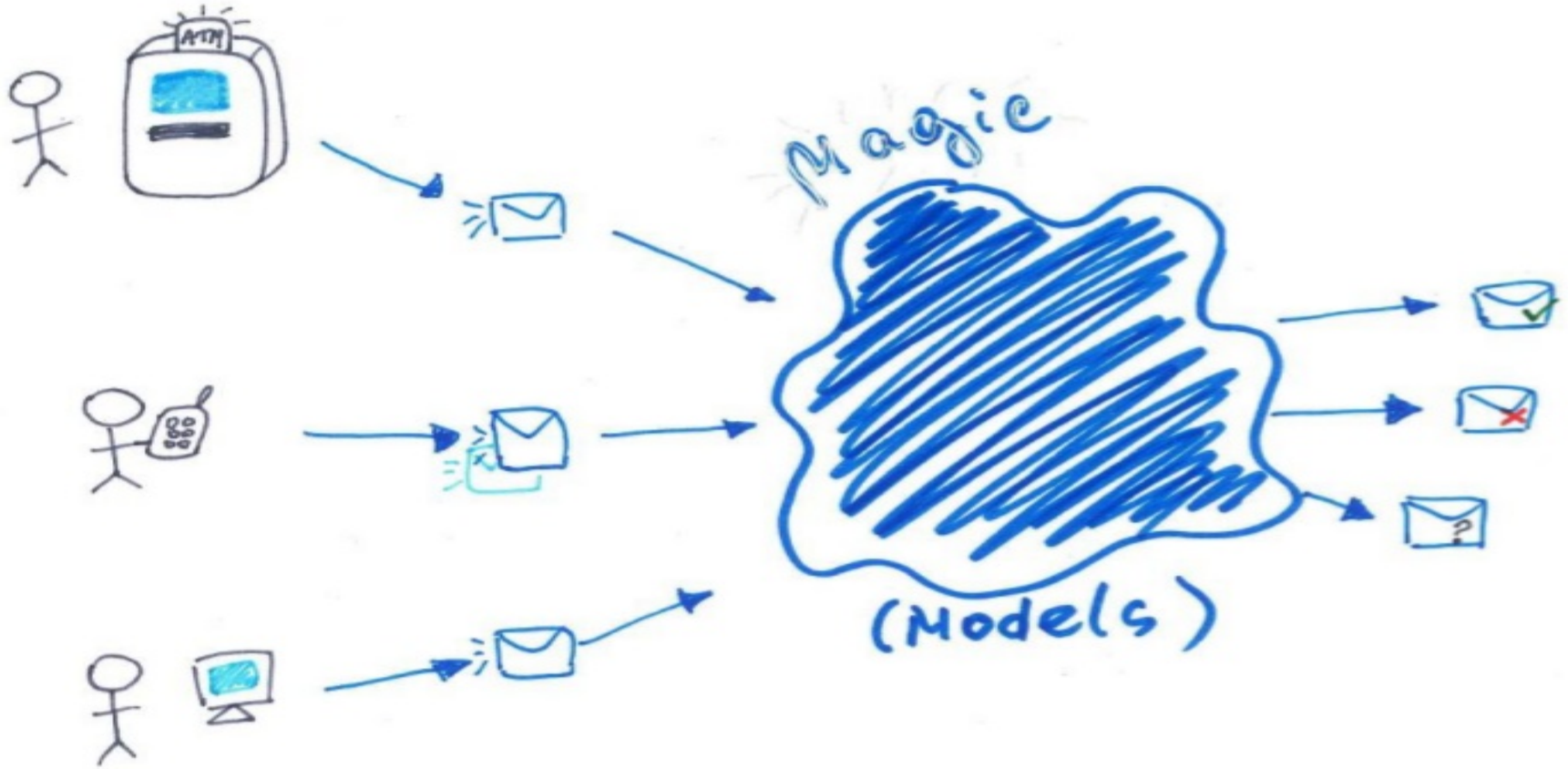
*per january 2018

ING

Use Case

ING

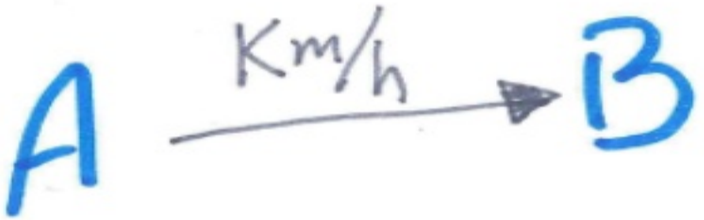
Use Case - Prevent Fraudulent Transactions



ING

Model Input - Features

- Speed of the client (between transactions)
- The debit card has been reported broken or stolen
- The customer has been blocked

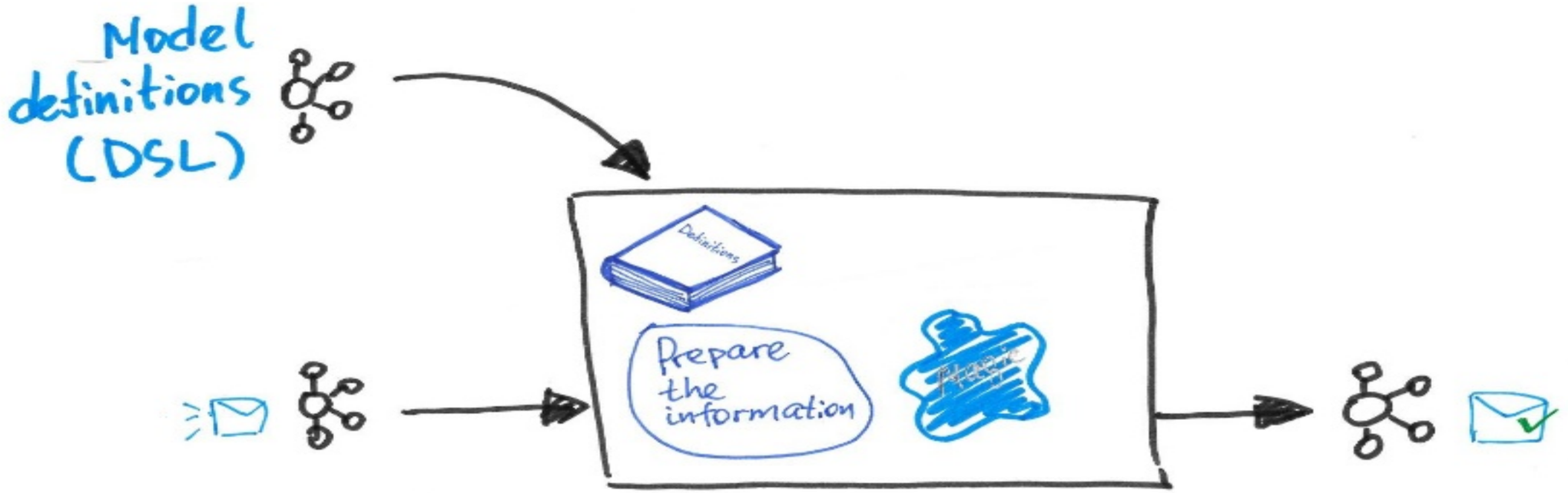


ING

Use Case – Requirements

- **Dynamic behaviour -> DSL**
- Historical data (state)
- Real-time (< 100 ms)
- High volumes (10K+ tps)
- Scalability

Use Case – Architecture



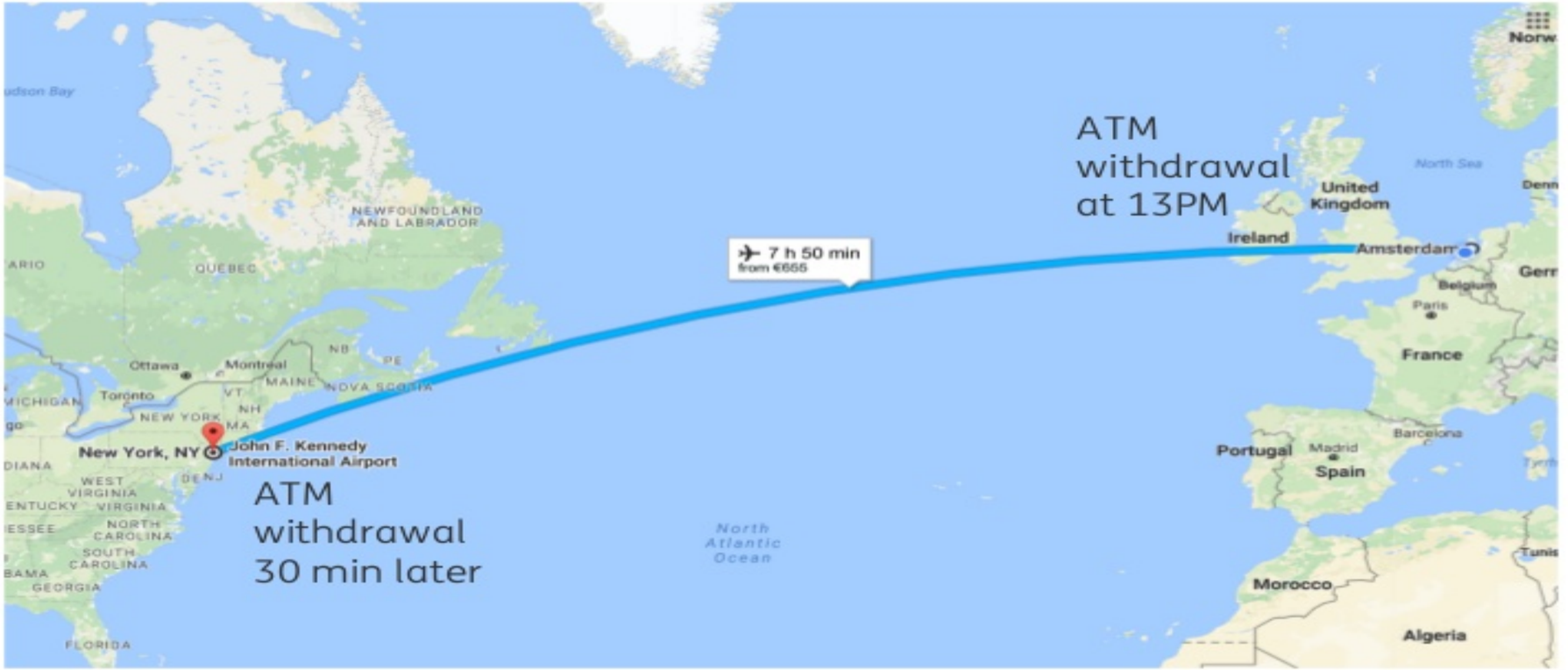
ING

Disclaimer

Scoring Example

ING

Flying Carpet Rule



ING

Need for State

To score we need:
Customer Speed ?

Event: Transaction

customerId
cardId
amount
location
eventTime

ING

Need for State

To score we need:
Customer Speed ?

Event: Transaction

customerId
cardId
amount
location
eventTime

+ prev.Location
prev.eventTime

ING

Need for State

To score we need:
Customer Speed ?

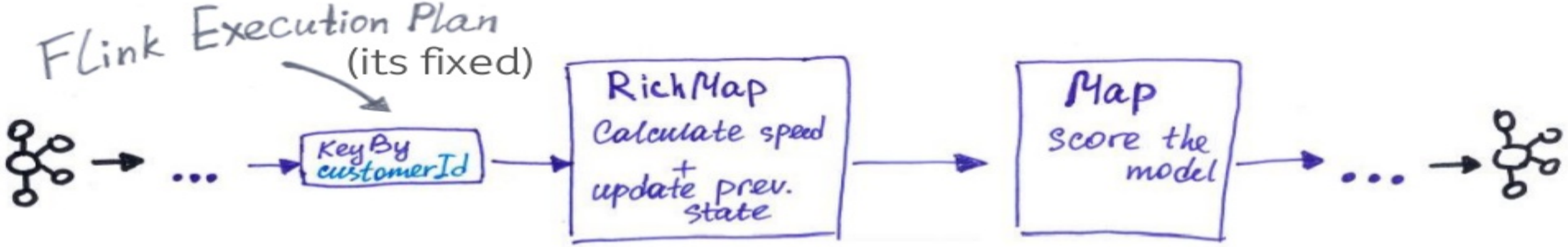
Event: Transaction

customerId
cardId
amount
location
eventTime

+ prev.Location
prev.eventTime
State

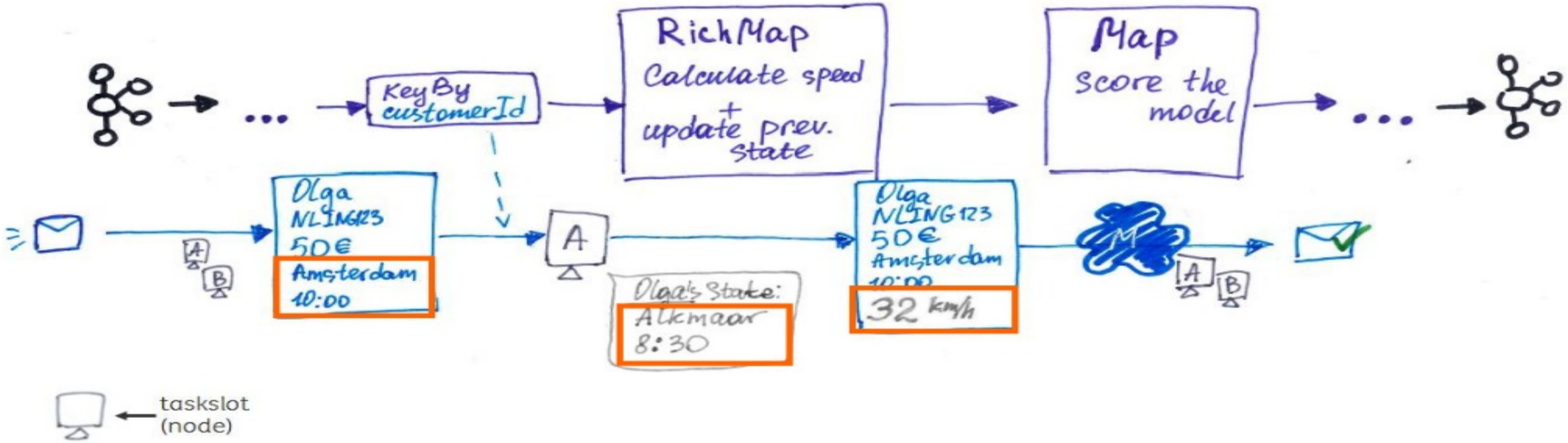
ING

Stateful Flying Carpet



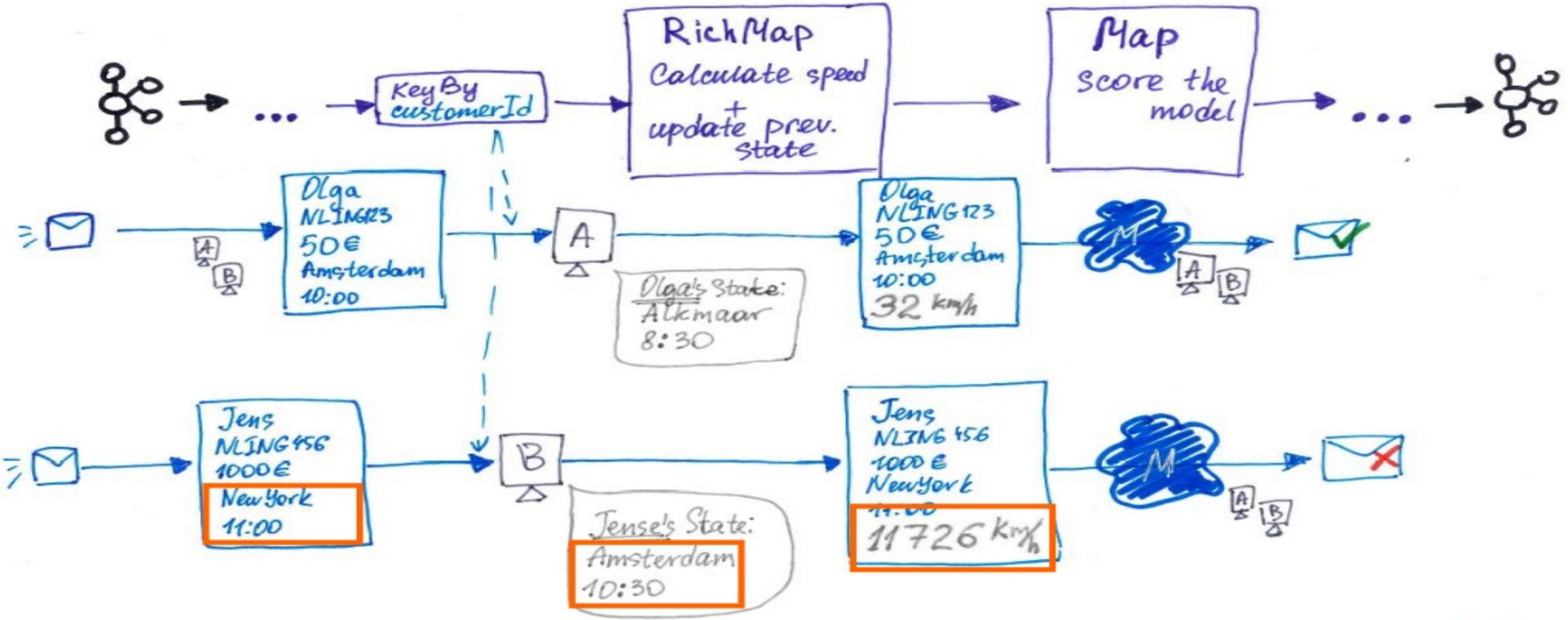
ING

Stateful Flying Carpet



ING

Stateful Flying Carpet



ING

Stateful Flying Carpet

CODE

Improved Stateful Example

ING

Flying Carpet Model Improved

To score better we need:

Customer Speed

Event: Transaction

...

+ prev. Location
prev. event Time

ING

Flying Carpet Model Improved

To score better we need:

Customer Speed
+ Av. amount per card

Event: Transaction
...

+ prev. Location
prev. event Time

+ total Amount
total nr. transactions

ING

Flying Carpet Model Improved

To score better we need:

Customer Speed
+ Av. amount per card

Event: Transaction
...

+ prev. Location
prev. event Time
← Customer state

+ total Amount
total nr. transactions

ING

Flying Carpet Model Improved

To score better we need:

Customer Speed
+ Av. amount per card

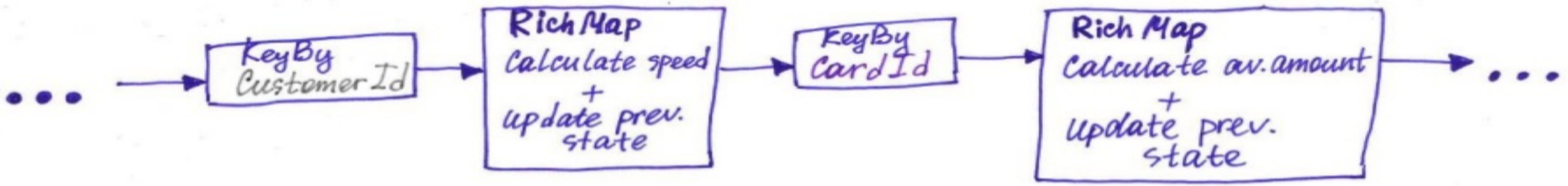
Event: Transaction
...

+ prev. Location
prev. event Time
← Customer state

+ total Amount
total nr. transactions
← Card state

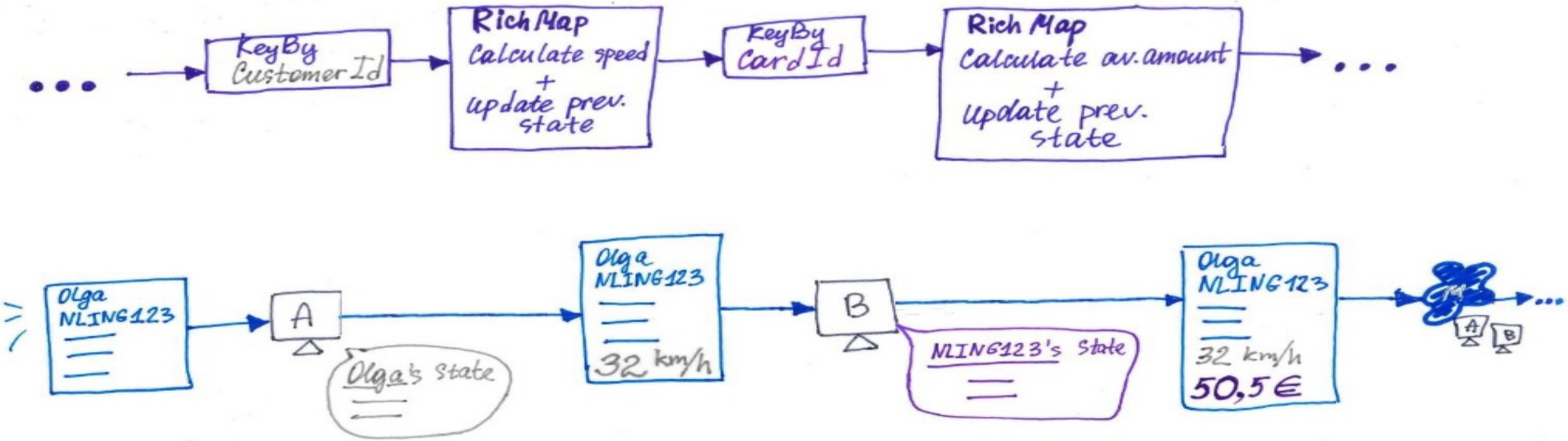
ING

Flink - Sequential Approach



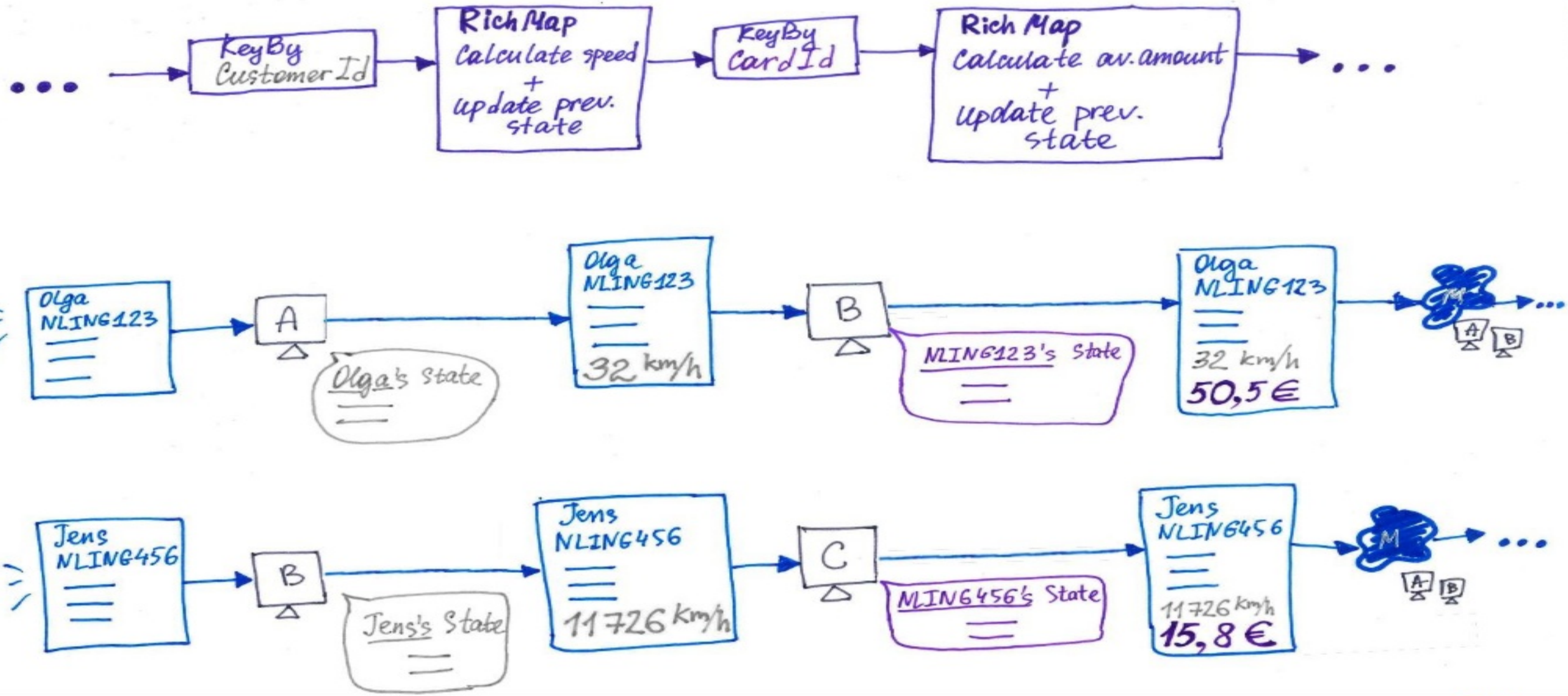
ING

Flink - Sequential Approach



ING

Flink - Sequential Approach



ING

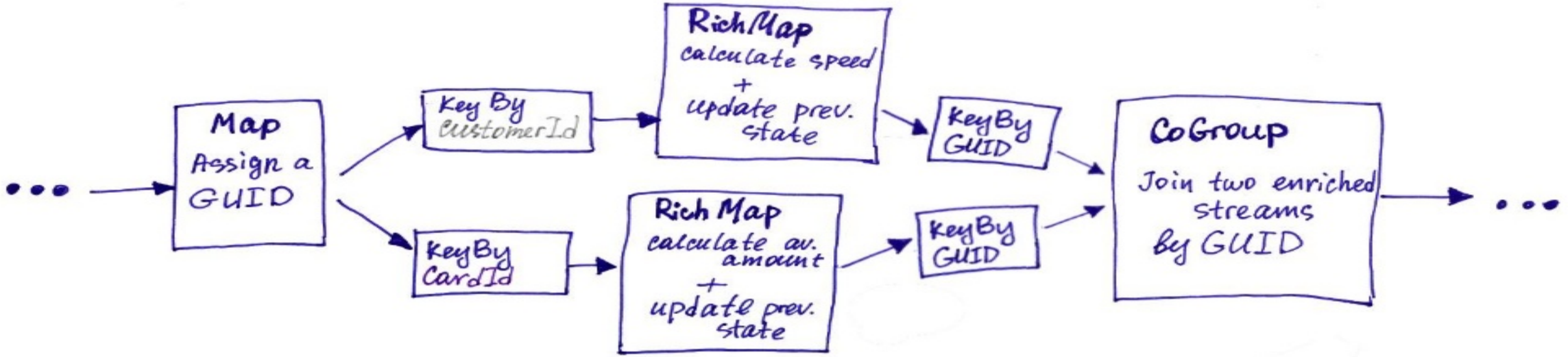
Sequential Approach

CODE

Sequential Approach - Evaluation

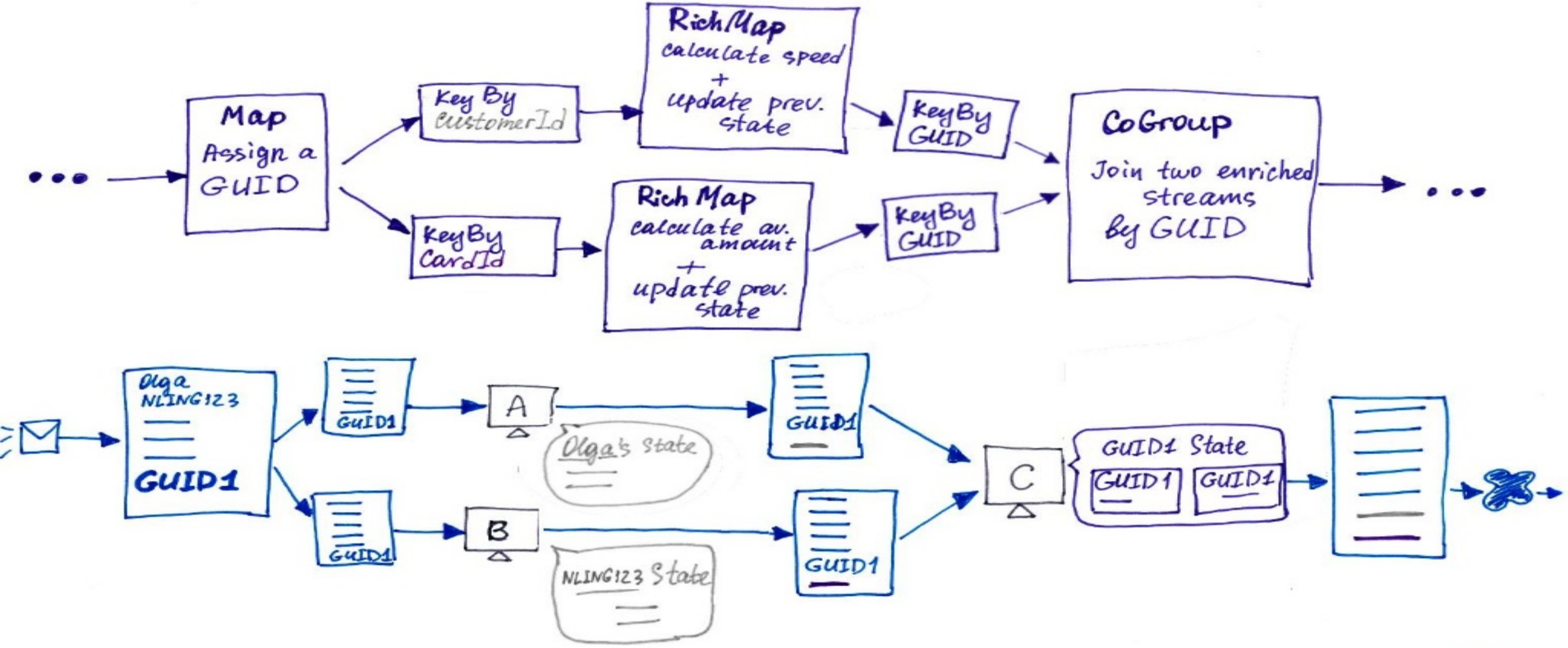
+ It works!
– Latency
– Not so scalable
– Event has to pass nodes even if their state is not required
  (could be solved with a filter)
– Cannot add new keys without restarting a job
  (remember: The execution plan is fixed)

Parallel Approach



ING

Parallel Approach



ING

Parallel Approach

CODE

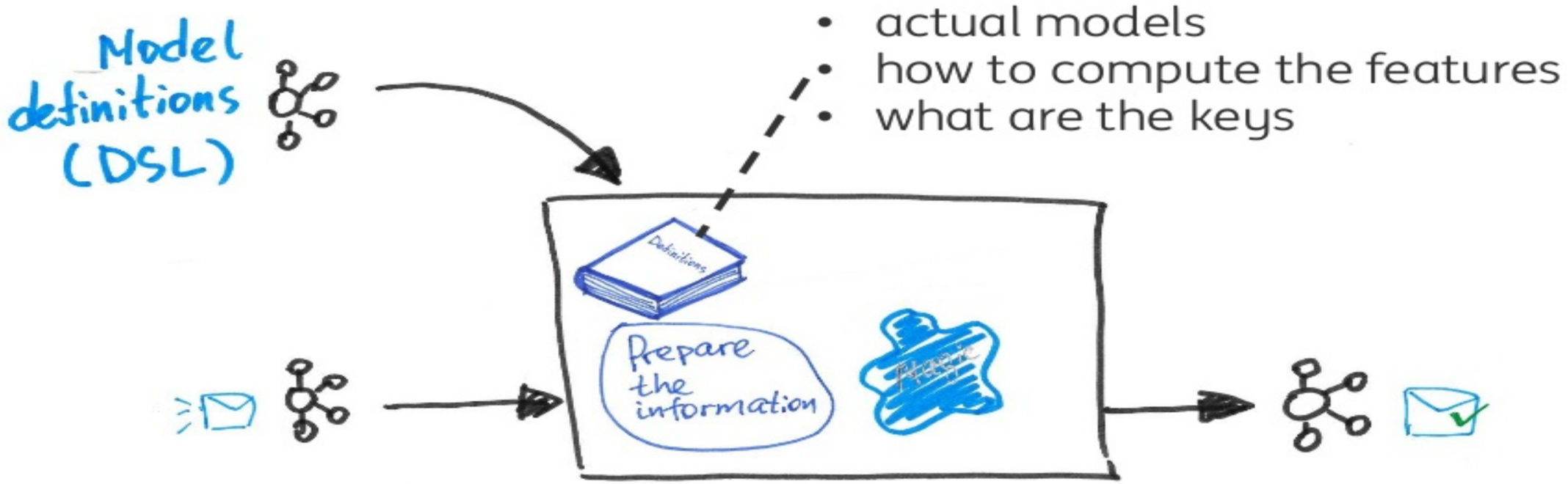
Parallel Approach - Evaluation

+ Less latency than when using sequential approach
+ More scalable
− Event has to pass nodes even if their state is not required
− Complex execution plan
− Cannot add new keys without restarting a job
  (remember: The execution plan is fixed)

Other Approach to Improved Stateful Example

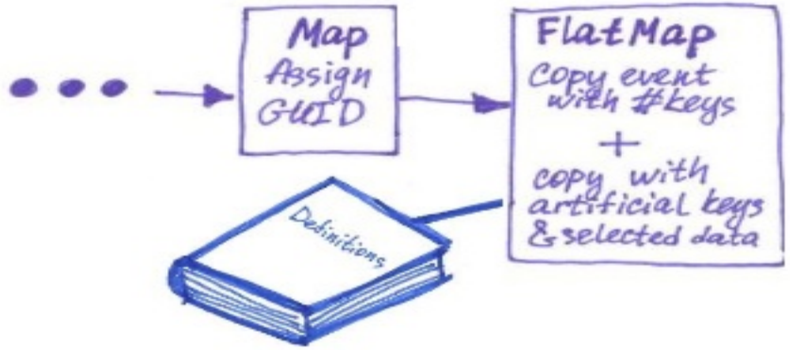
ING

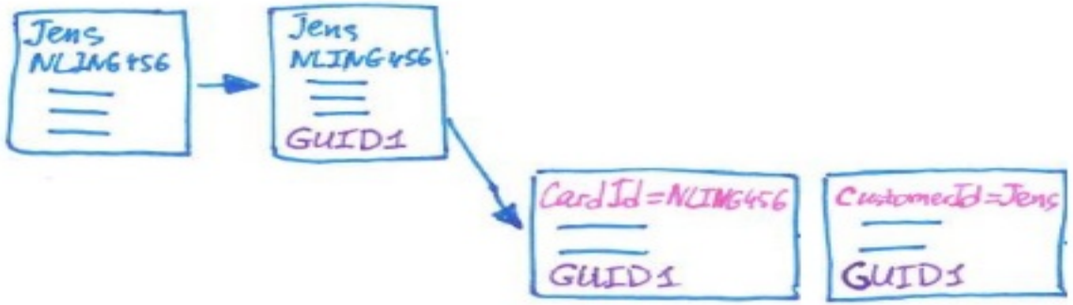
Architecture recap



ING

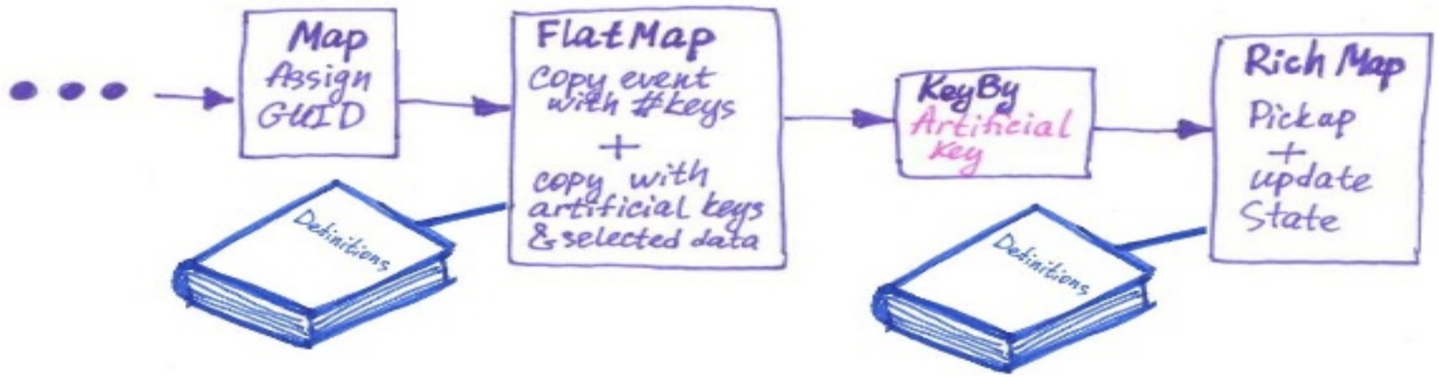
Advanced Approach

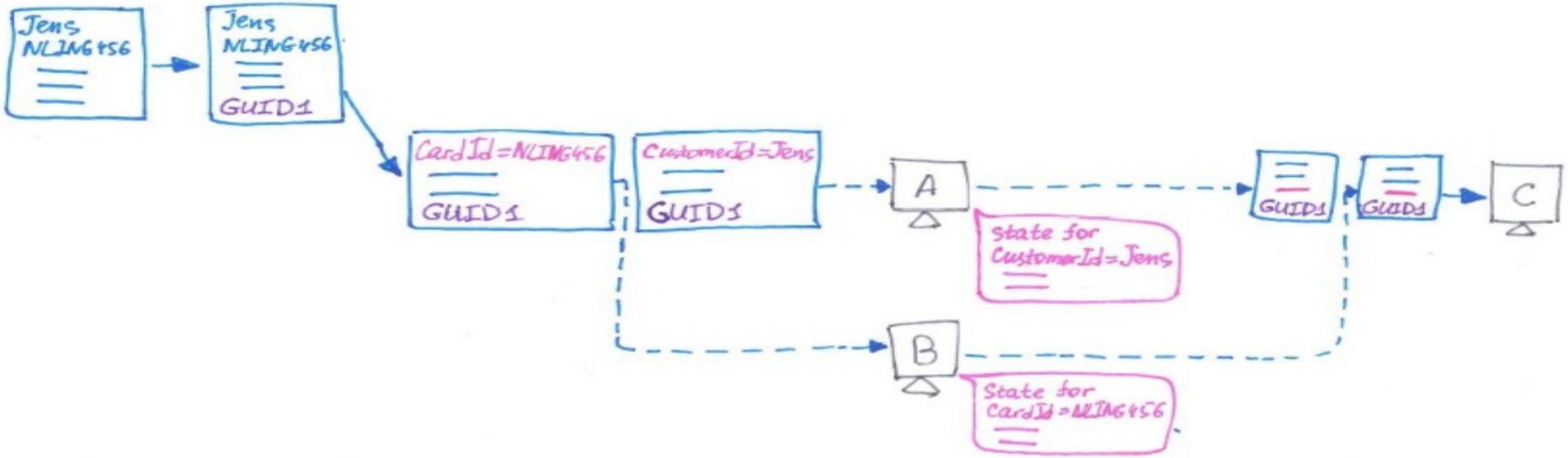




ING

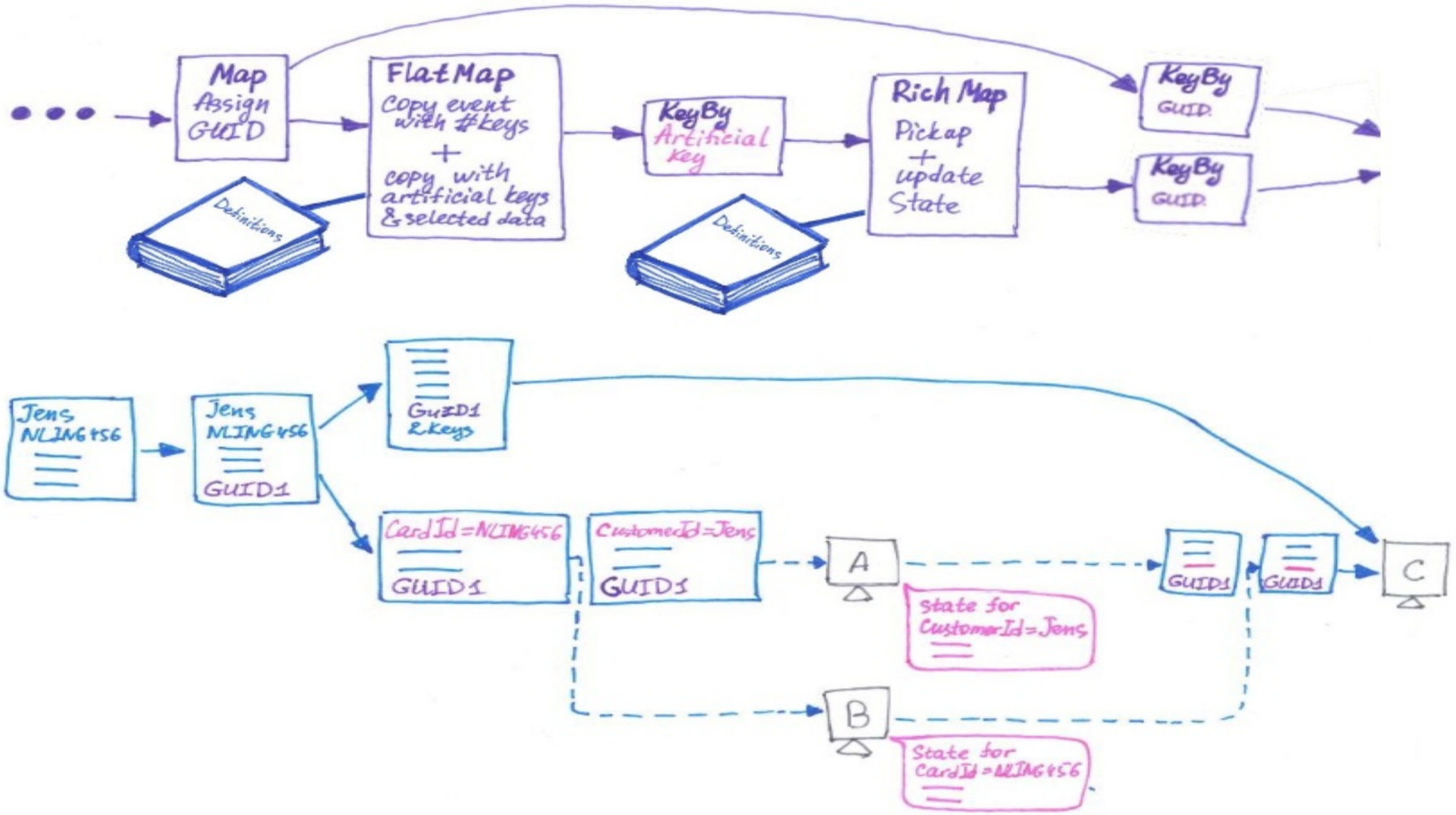
Advanced Approach





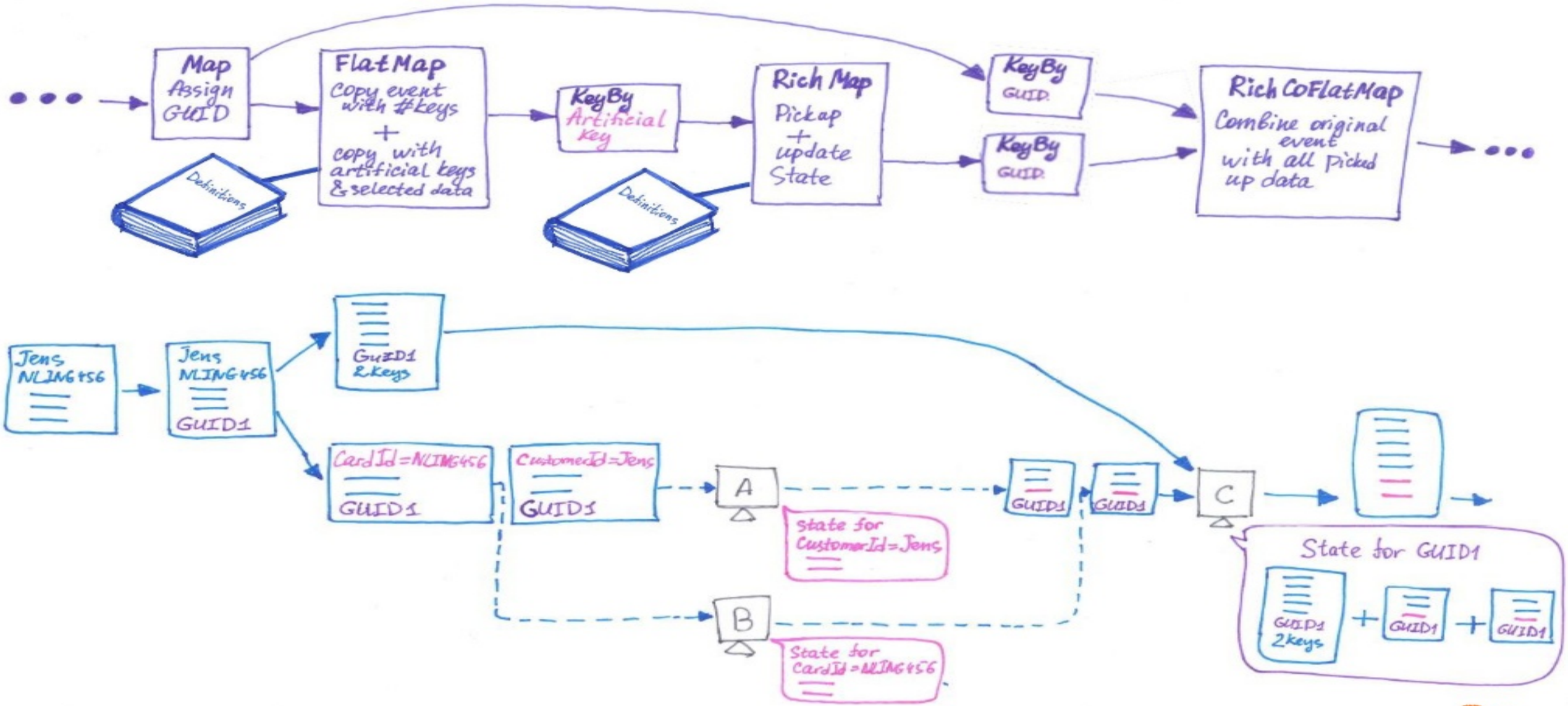
ING

Advanced Approach



ING

Advanced Approach



ING

Flink - Advanced Approach

CODE

Advanced approach - evaluation

+ Can add new keys without restarting a job
+ Event is not passing through nodes that are not needed
+ Simple execution plan
– Quite complex implementation
– Requires a way to deliver a description of the behaviour in some way

Conclusion

Conclusion

Although the execution plan is fixed, your business logic does not have to be

The End

Thank you!

Check out the
demo code on GitHub:

github.com/rezolya/exploiting-flinks-stateful-operators

ING