RuM:
declarative process mining, distilled

Anti Alman, U. Tartu

Claudio Di Ciccio, Sapienza

Fabrizio M. Maggi, U. Bolzano

Marco Montali, U. Bolzano

Han van der Aa, U. Mannheim

The Team

Anti Alman
University of Tartu

Claudio di Ciccio
Sapienza University of Rome

Fabrizio Maria Maggi
Free University of Bozen-Bolzano

Marco Montali
Free University of Bozen-Bolzano

Han van der Aa
University of Mannheim

Today

Outline

- Introduction to declarative process mining
- Declarative Process Mining with RuM
- Research opportunities

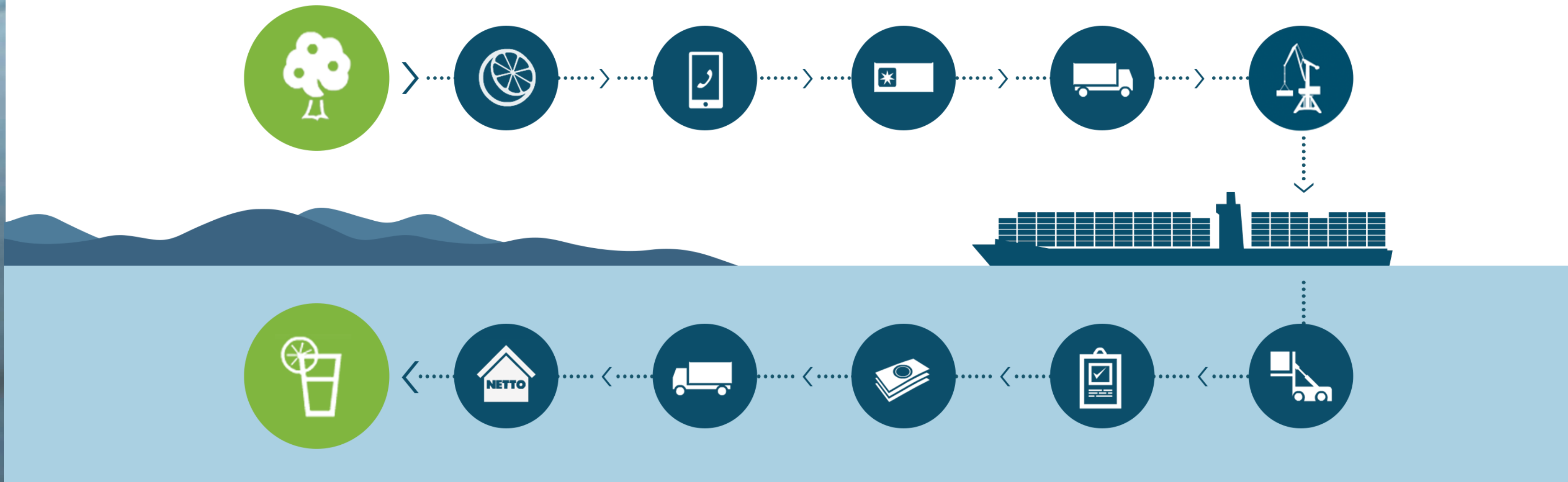
Setting

- Tutorial is hands-on using the RuM tool (download at: www.rulemining.org)
- Questions are welcome at any moment

Introduction to declarative process mining

BPM!

BPM?



Types of Processes

Not all processes are the same

- Even within one organization, processes can be very different in terms of their essential properties.
- Processes can be characterized through three dimensions:
  - complexity
  - predictability
  - repetitiveness

Complexity

Degree of difficulty in collaboration, coordination, and decision making

- Low complexity: exchanging personal email messages and handling travel requests
- High complexity: handling medical treatments

Predictability

How easy it is to **determine in advance** the way the process will be executed.

- Low predictability: exchanging personal email messages.
- High predictability: handling travel requests.

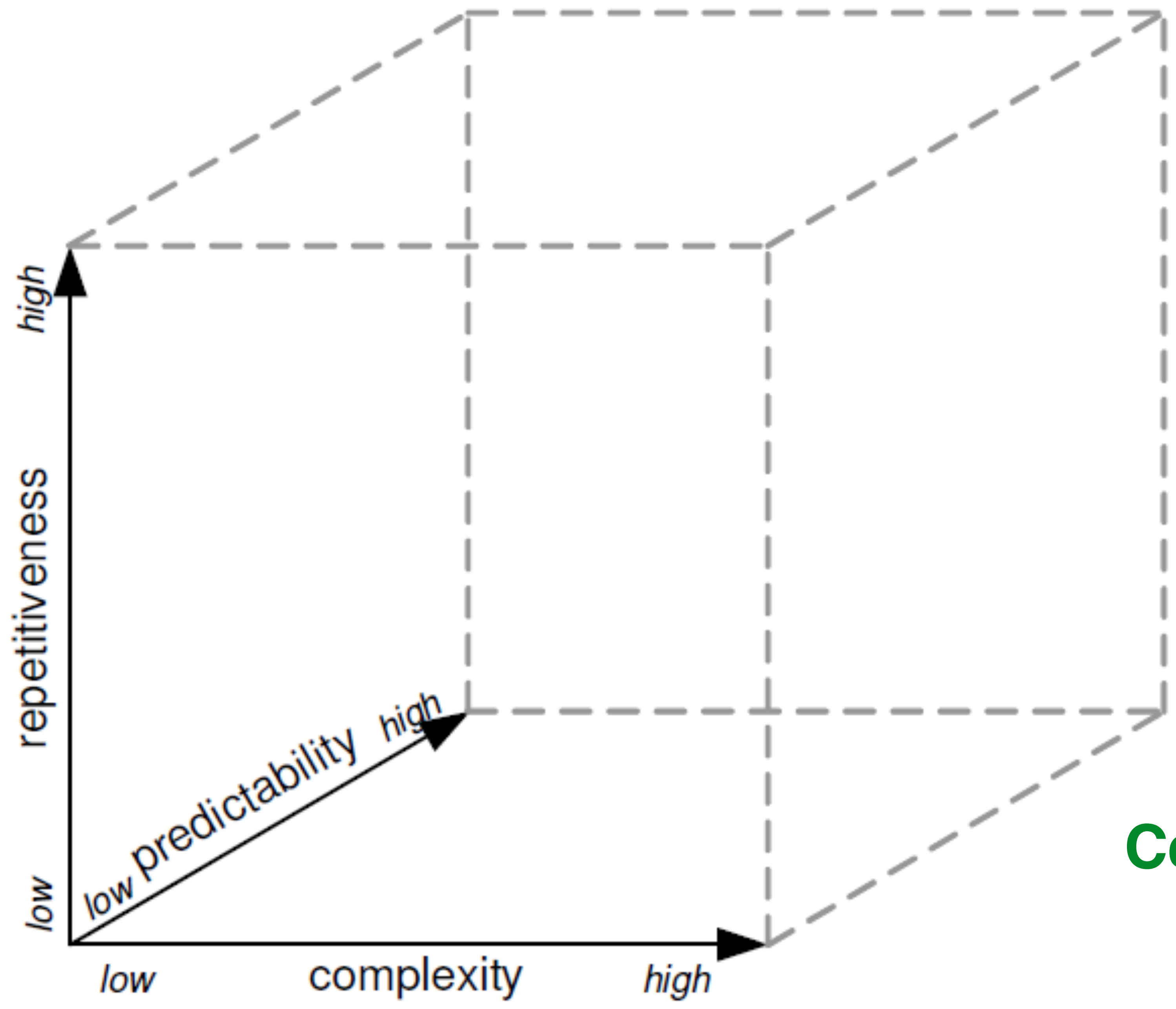
Repetitiveness

Frequency of process execution.

A business process that is executed once per year has a lower degree of repetitiveness than a process executed twice a day.

- Low repetitiveness: disaster handling.
- High repetitiveness: exchanging personal email messages.

Classifying processes



Prepare coffee

Register birth

Produce shoes

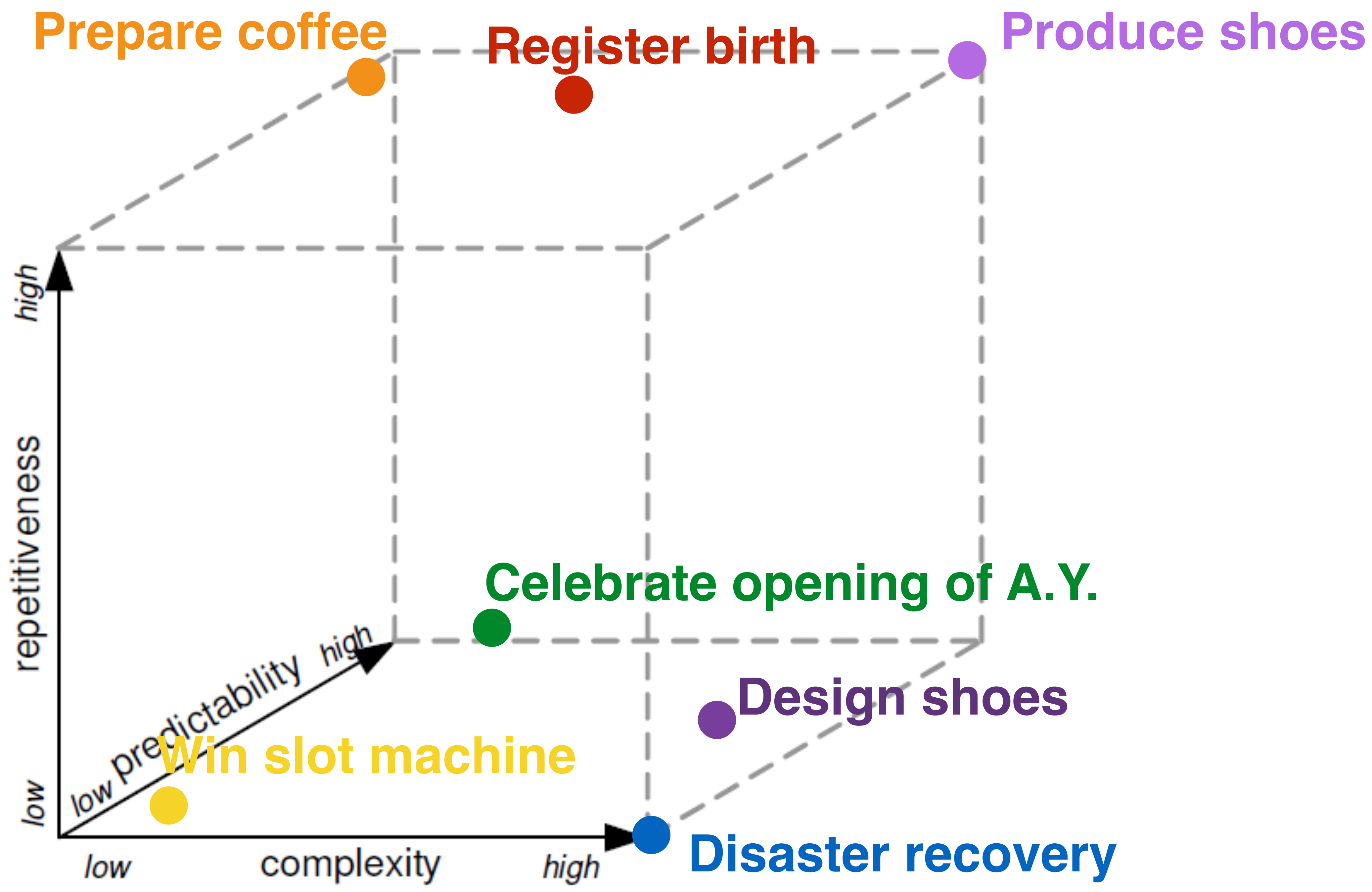
Design shoes

Win slot machine

Celebrate opening of A.Y.

Disaster recovery

Classifying processes

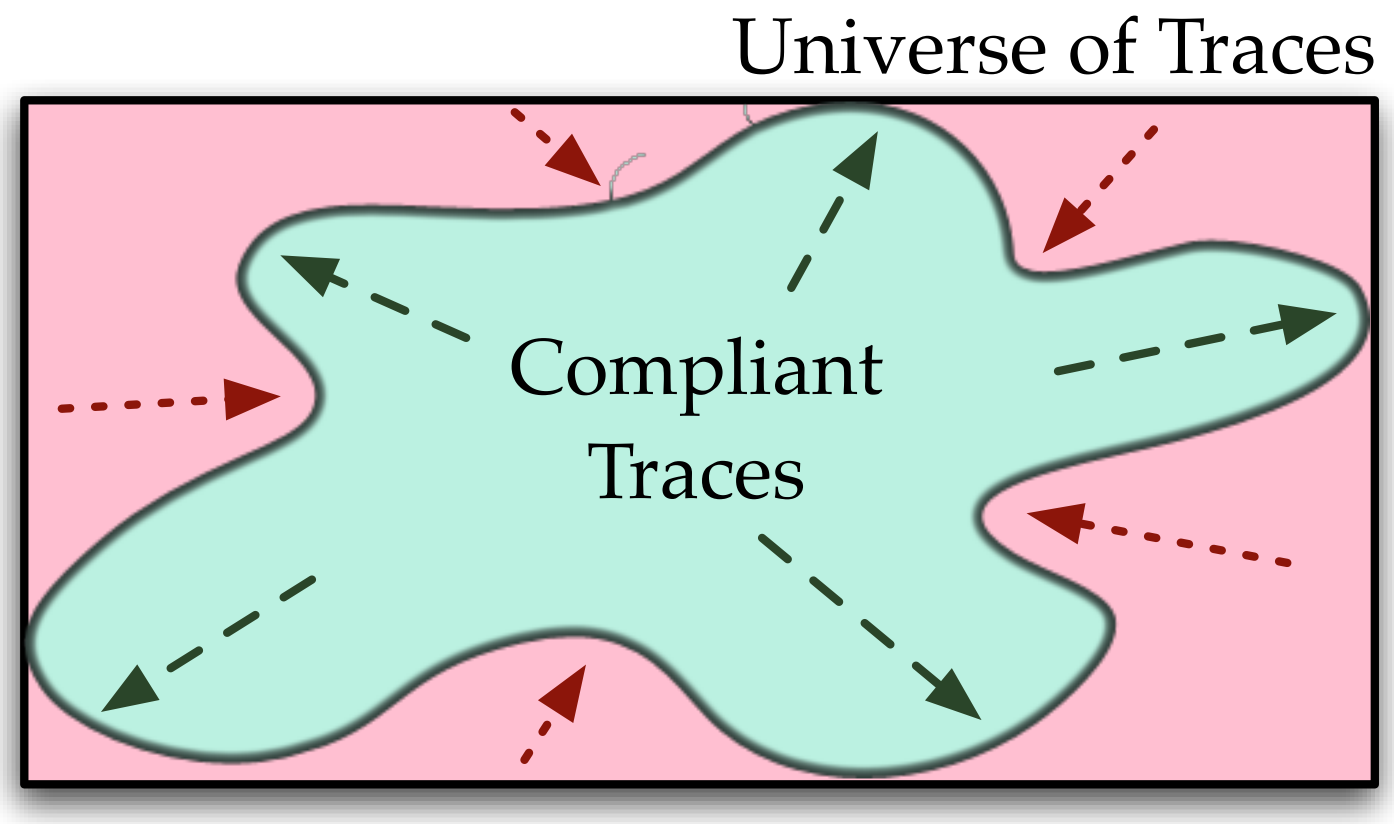


Environment!

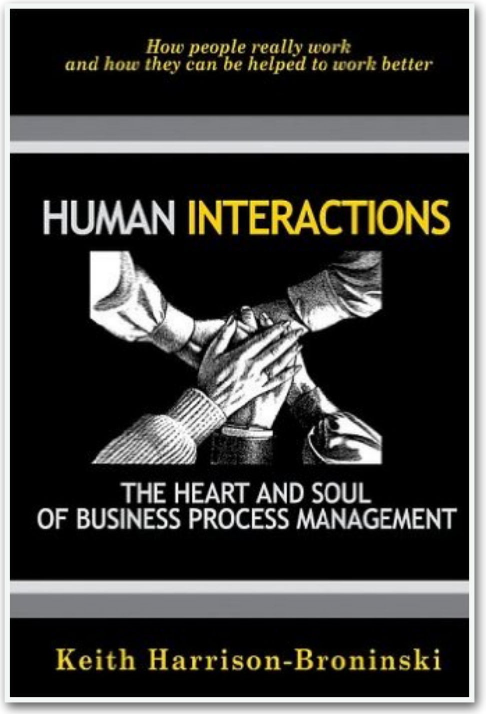
Flexibility vs support

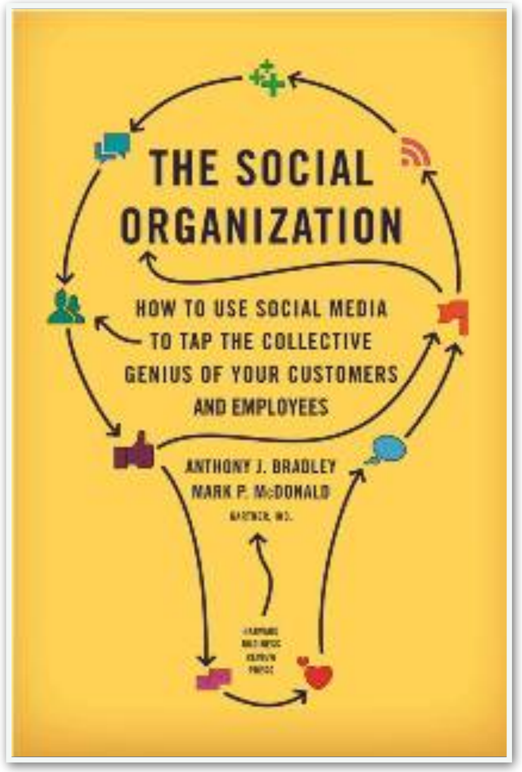
Flexibility: degree to which users can make local decisions about how to execute processes.

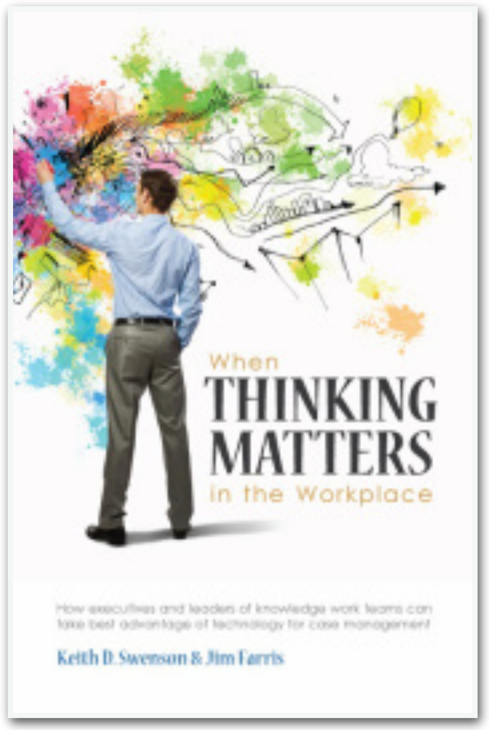
Support: degree to which a system makes centralized decisions about how to execute processes.

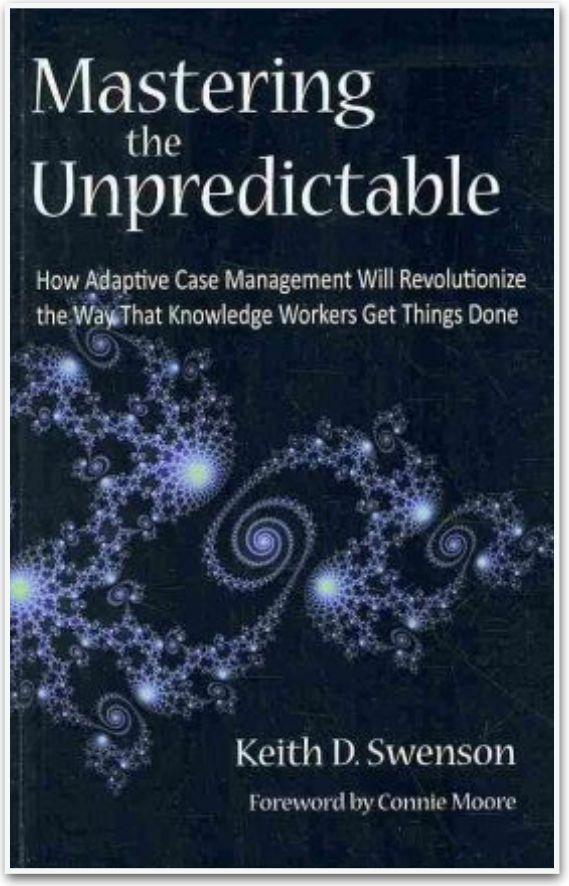


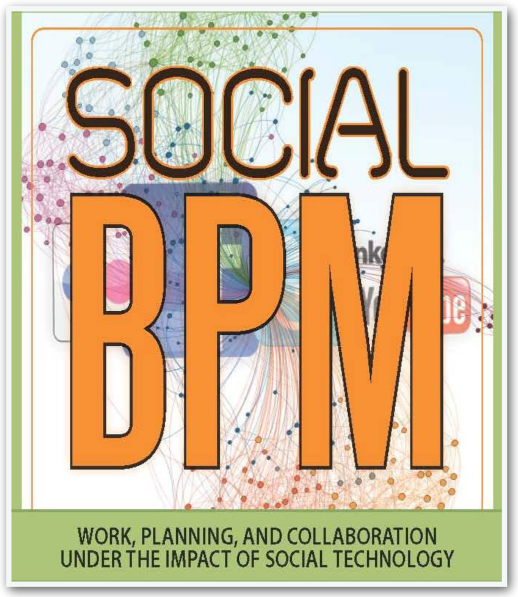
The issue of flexibility



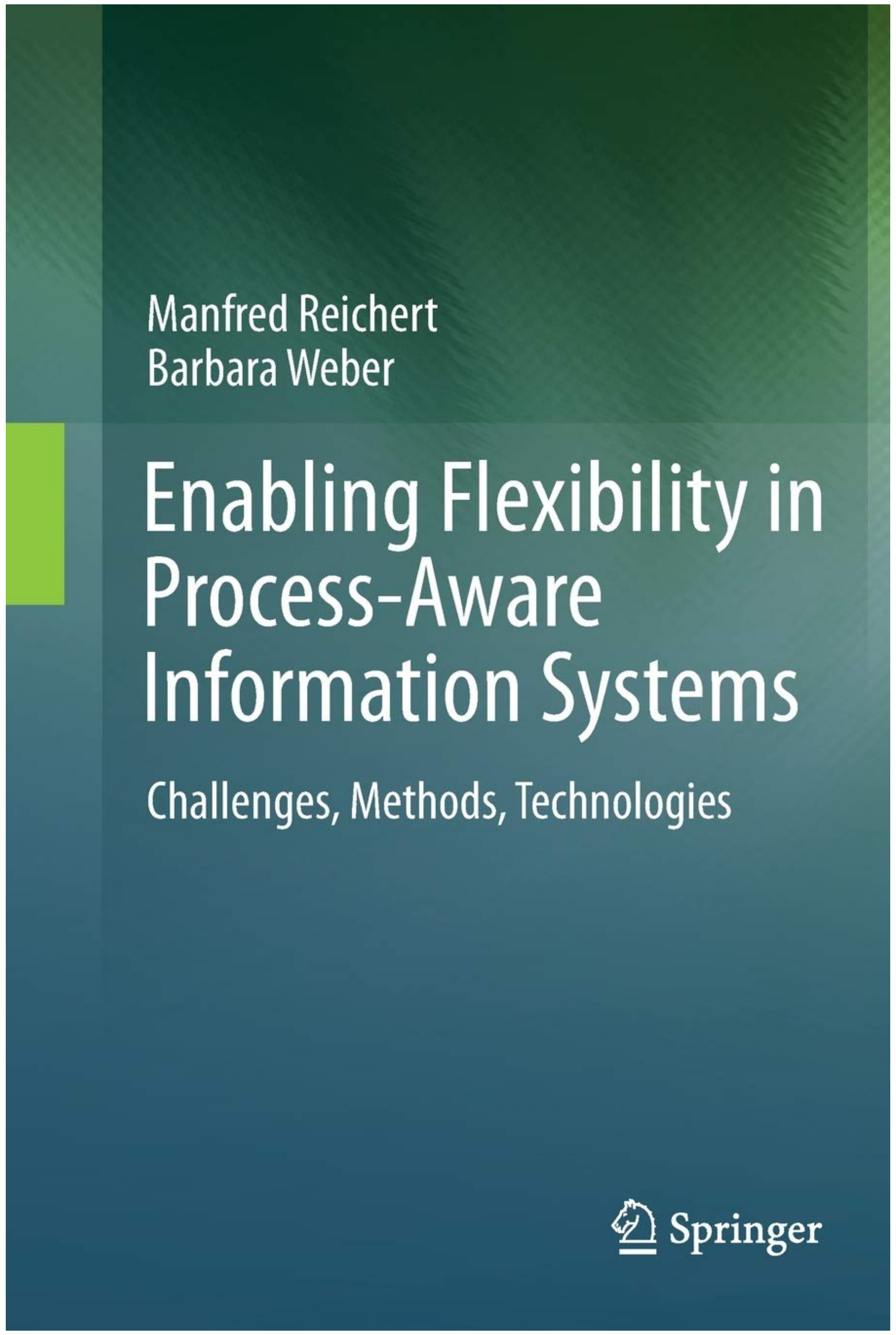

THE SOCIAL ORGANIZATION
HOW TO USE SOCIAL MEDIA TO TAP THE COLLECTIVE GENIUS OF YOUR CUSTOMERS AND EMPLOYEES
ANTHONY J. BRADLEY
MARK P. McDONALD
GARTNER, INC.


When
THINKING MATTERS
in the Workplace
How executives and leaders of knowledge work teams can take best advantage of technology for case management
Keith D. Swenson & Jim Farris


Mastering the Unpredictable
How Adaptive Case Management Will Revolutionize the Way That Knowledge Workers Get Things Done
Keith D. Swenson
Foreword by Connie Moore


SOCIAL BPM
WORK, PLANNING, AND COLLABORATION UNDER THE IMPACT OF SOCIAL TECHNOLOGY

The issue of flexibility


Manfred Reichert
Barbara Weber

Enabling Flexibility in Process-Aware Information Systems

Challenges, Methods, Technologies

Springer

Our focus:

flexibility

by design

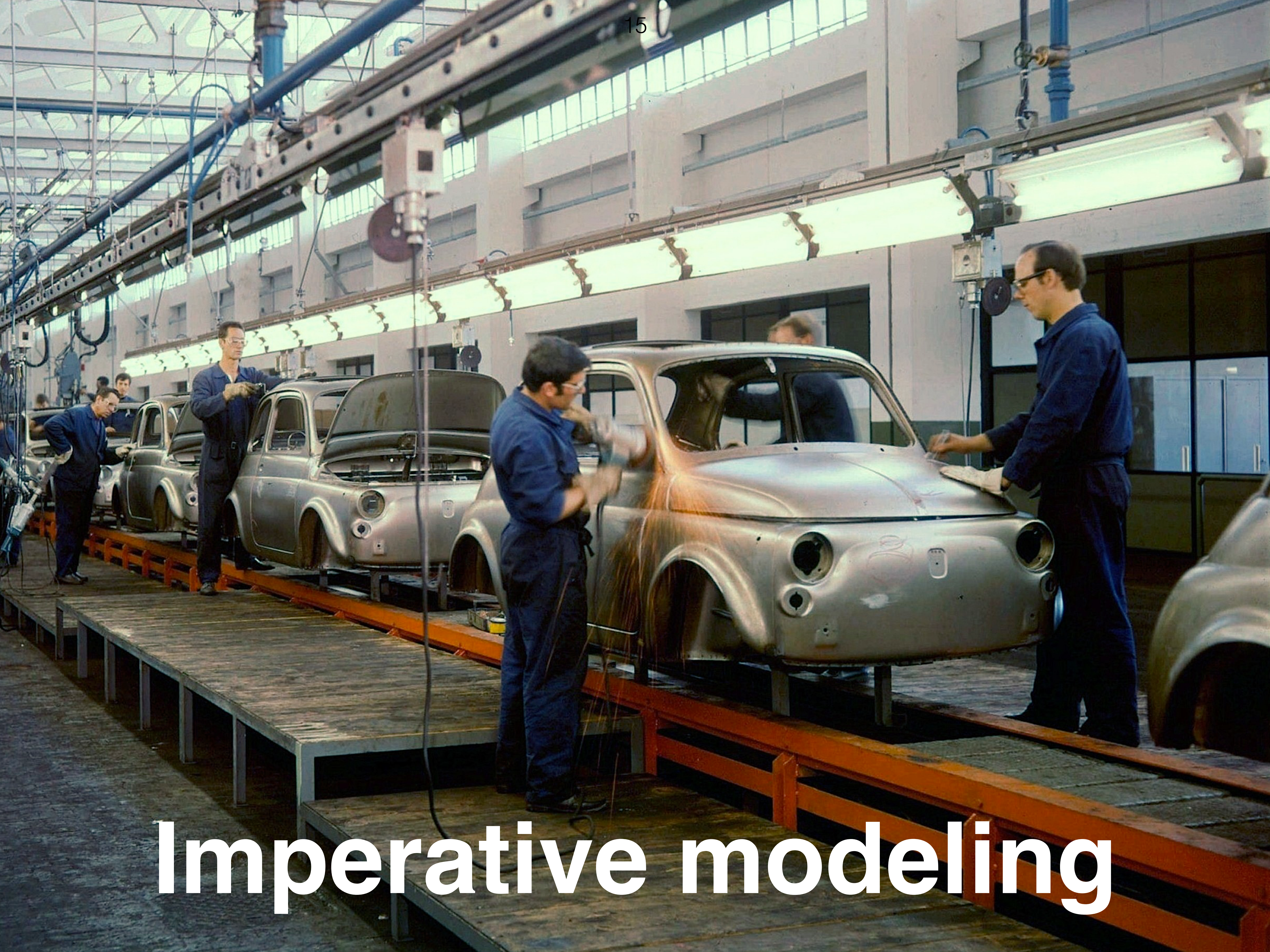
Two paradigms

Imperative modeling

- Traditional repetitive BPs
- Metaphor: **flow-chart**

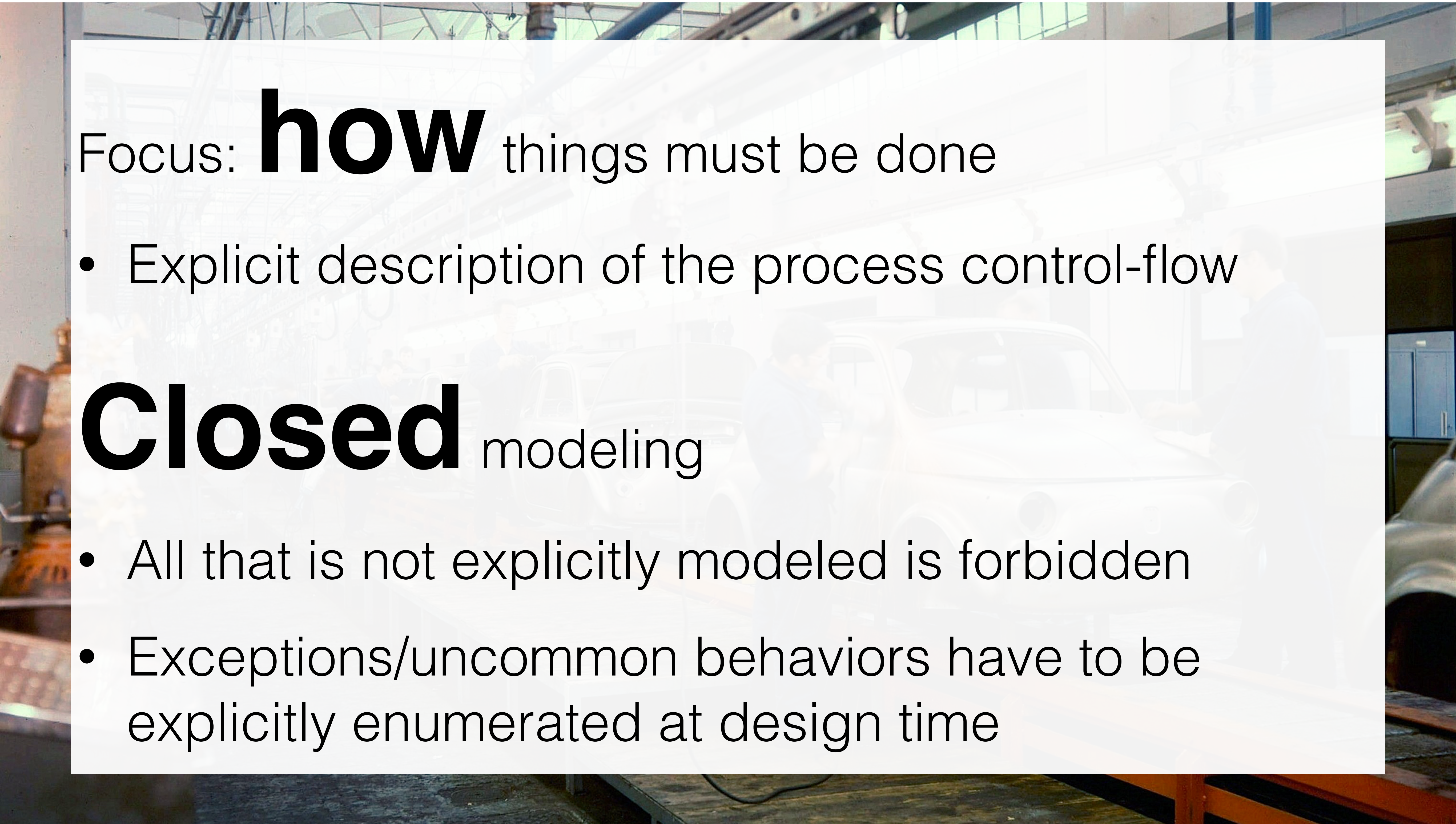
Declarative modeling

- Highly-variable BPs
- Metaphor: rules/**constraints**



Imperative modeling

Idea



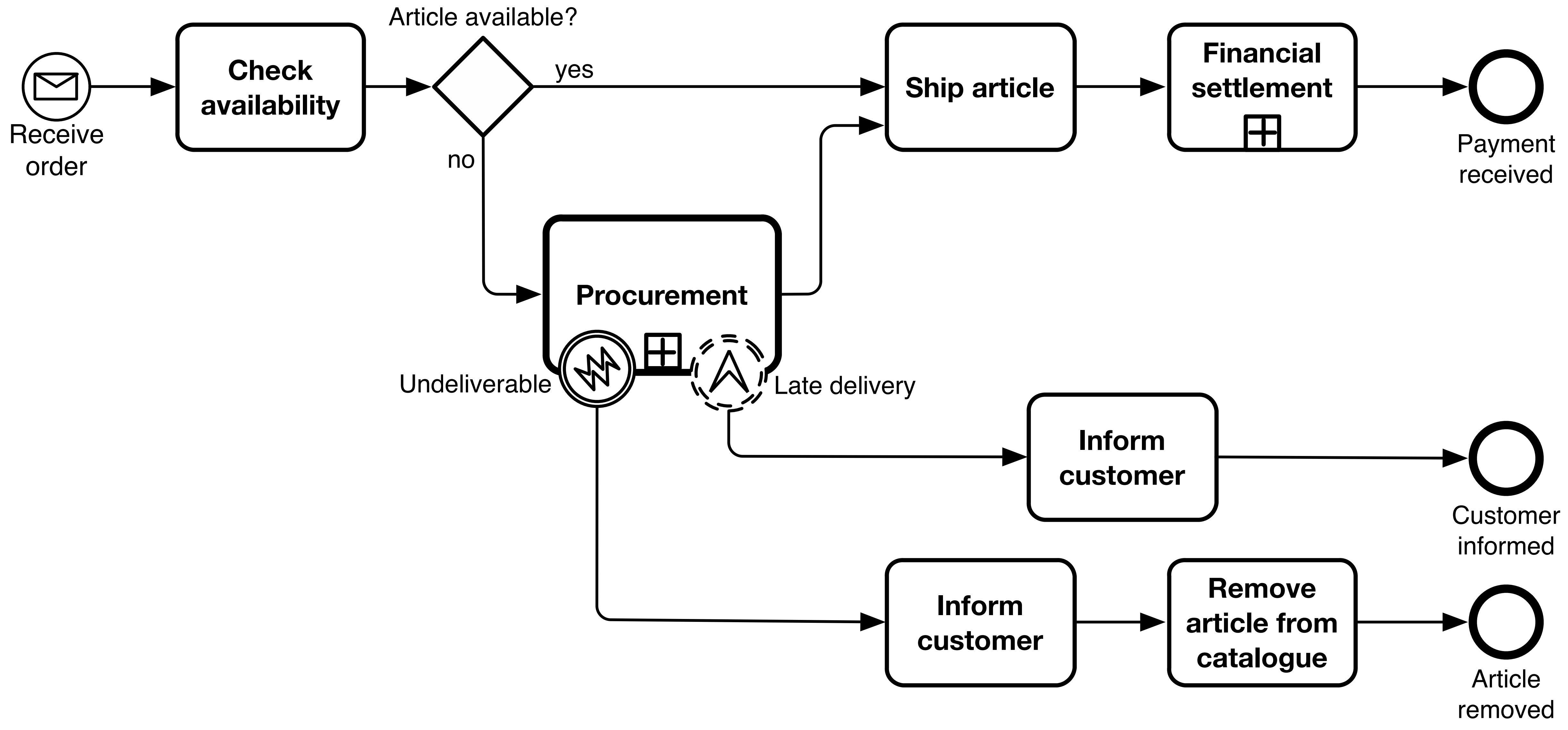
Focus: **how** things must be done

- Explicit description of the process control-flow

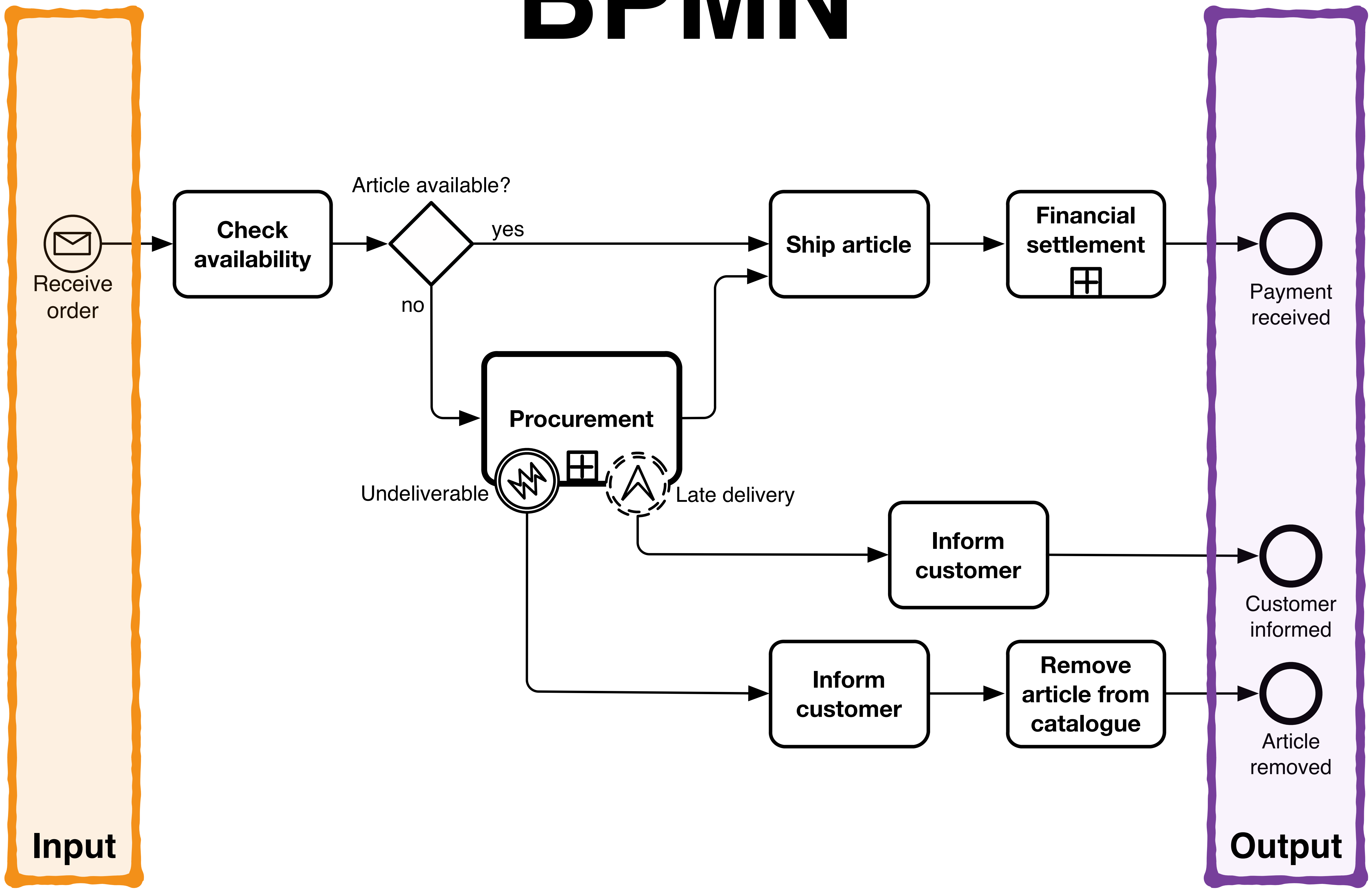
Closed modeling

- All that is not explicitly modeled is forbidden
- Exceptions/uncommon behaviors have to be explicitly enumerated at design time

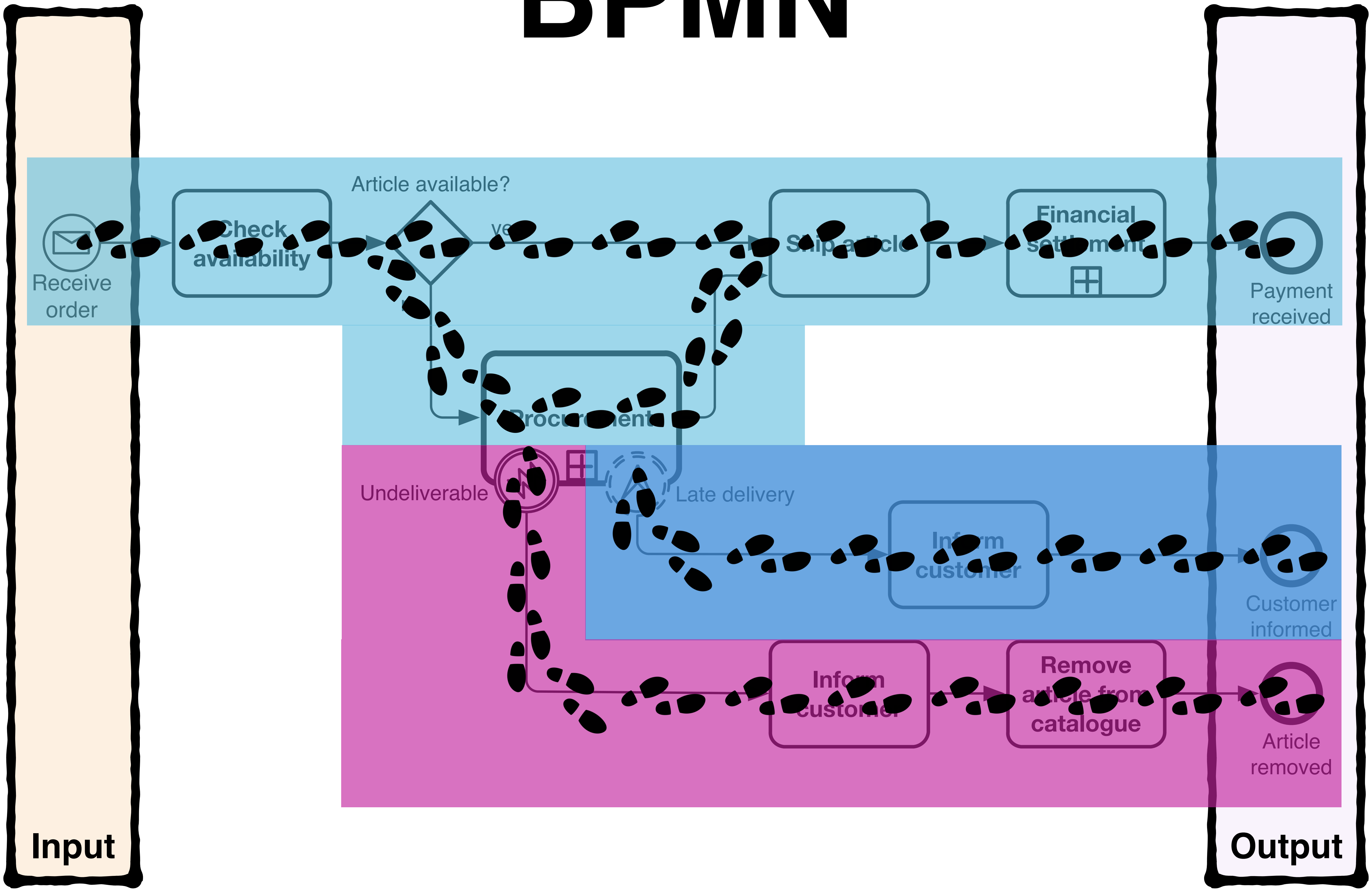
BPMN


Article available?
Receive order
Check availability
yes
no
Ship article
Financial settlement
Payment received
Procurement
Undeliverable
Late delivery
Inform customer
Customer informed
Inform customer
Remove article from catalogue
Article removed

BPMN

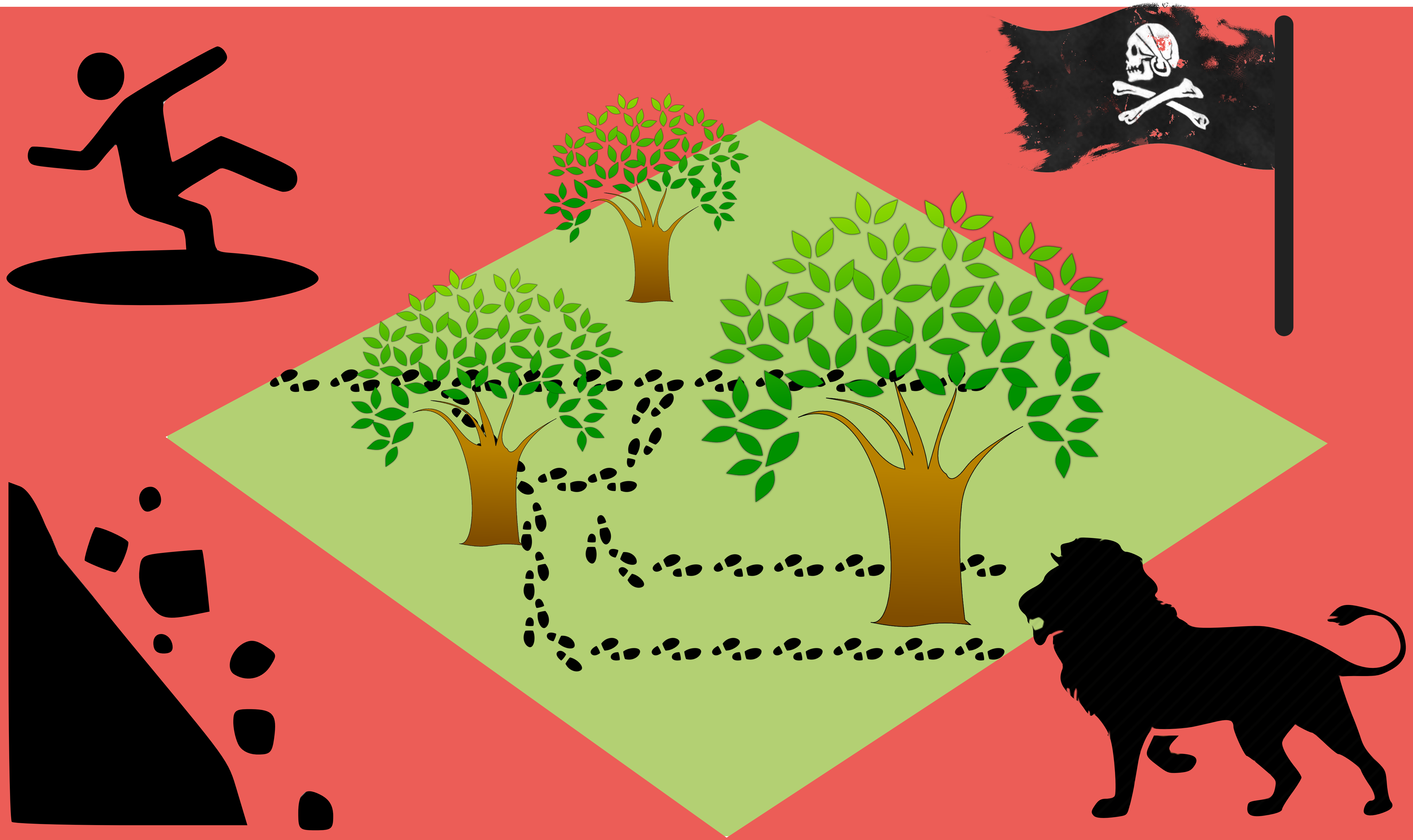

Receive order
Check availability
Article available?
yes
no
Ship article
Financial settlement
Payment received
Procurement
Undeliverable
Late delivery
Inform customer
Customer informed
Inform customer
Remove article from catalogue
Article removed
Input
Output

BPMN


Article available?
yes
Receive order
Check availability
Ship article
Financial settlement
Payment received
Procurement
Undeliverable
Late delivery
Inform customer
Customer informed
Inform customer
Remove article from catalogue
Article removed
Input
Output

Organizational boundaries

Imperative modeling



Example: rigid shopping

- Tasks: $\boldsymbol{\Sigma}$ = {pick **i**tem, **c**lose order, **p**ay}
- A customer starts the process by picking one or more items. She then decides to quit or close the order. In the latter case, she has to pay for the order.

Questions

- Tasks: **Σ** = {pick **i**tem, **c**lose order, **p**ay}
- A customer starts the process by picking one or more items. She then decides to quit or close the order. In the latter case, she has to pay for the order.
- Which traces are accepted?
  - **`<>`** (empty trace)
  - **`<i,i,i>`**
  - **`<i,i,i,c,p>`**
  - **`<i,i,i,c,p,p>`**
  - **`<i,i,i,p,c>`**
  - **`<i,c,p,i,i,c,p>`**

Questions

- Tasks: **Σ** = {pick **i**tem, **c**lose order, **p**ay}
- A customer starts the process by picking one or more items. She then decides to quit or close the order. In the latter case, she has to pay for the order.
- Which traces are accepted?
  - **<>** (empty trace)
  - **<i,i,i>**
  - **<i,i,i,c,p>**
  - **<i,i,i,c,p,p>**
  - **<i,i,i,p,c>**
  - **<i,c,p,i,i,c,p>**

Questions

Can you model this process in BPMN?

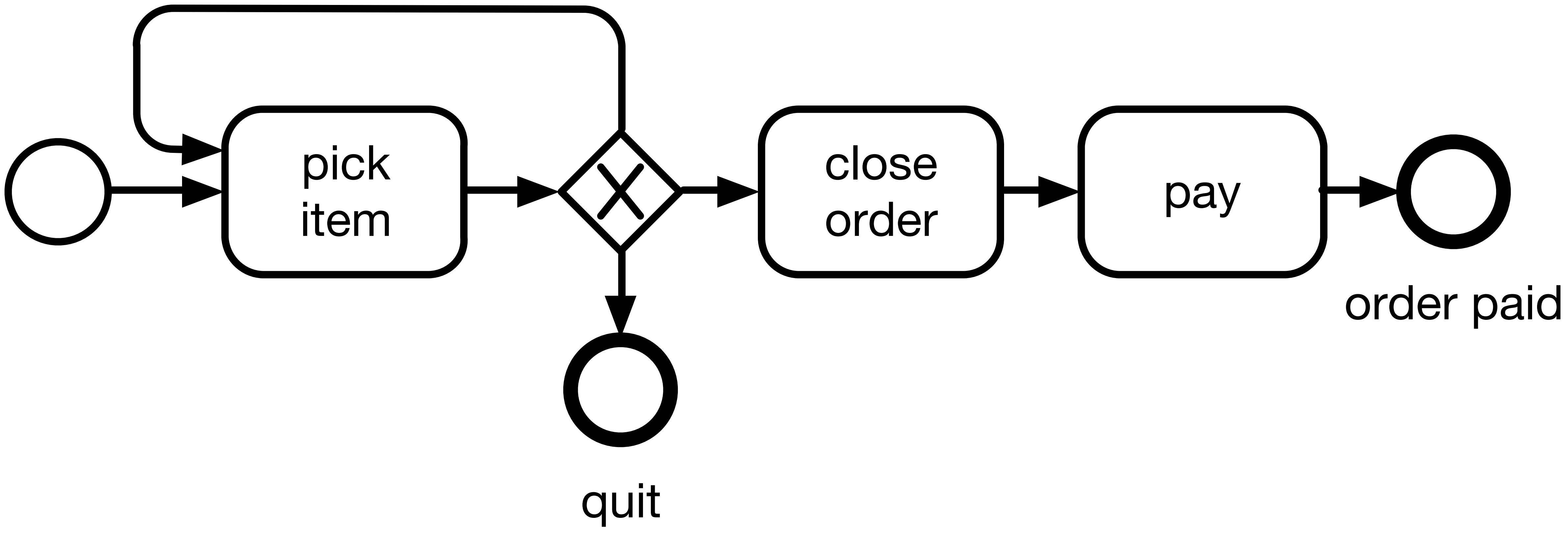
How difficult is it?

A customer starts the process by picking one or more items. She then decides to quit or close the order. In the latter case, she has to pay for the order.

Questions

Can you model this process in BPMN?

How difficult is it?



Example: flexible shopping

- Tasks: $\boldsymbol{\Sigma}$ = {pick **i**tem, **c**lose order, **p**ay}
- A customer may pick items, close several orders, and pay one or more orders at once.
- She may even close an empty order and pay for it, or pay multiple times (but this will result in a no-op)
- **Business constraint**: whenever you close an order, eventually you have to pay.

Questions

- Tasks: $\mathbf{\Sigma}$ = {pick **i**tem, **c**lose order, **p**ay}
- Business constraint: whenever you close an order, eventually you have to pay.
- Which traces are accepted?
  - **`<>`** (empty trace)
  - **`<i,i,i>`**
  - **`<i,i,i,c,p>`**
  - **`<i,i,i,c,p,p>`**
  - **`<i,i,i,p,c>`**
  - **`<i,c,p,i,i,c,p>`**

Questions

- Tasks: **Σ** = {pick **i**tem, **c**lose order, **p**ay}
- Business constraint: whenever you close an order, eventually you have to pay.
- Which traces are accepted?
  - **<>** (empty trace)
  - **<i,i,i>**
  - **<i,i,i,c,p>**
  - **<i,i,i,c,p,p>**
  - **<i,i,i,p,c>**
  - **<i,c,p,i,i,c,p>**

Questions

Can you model this process in BPMN?

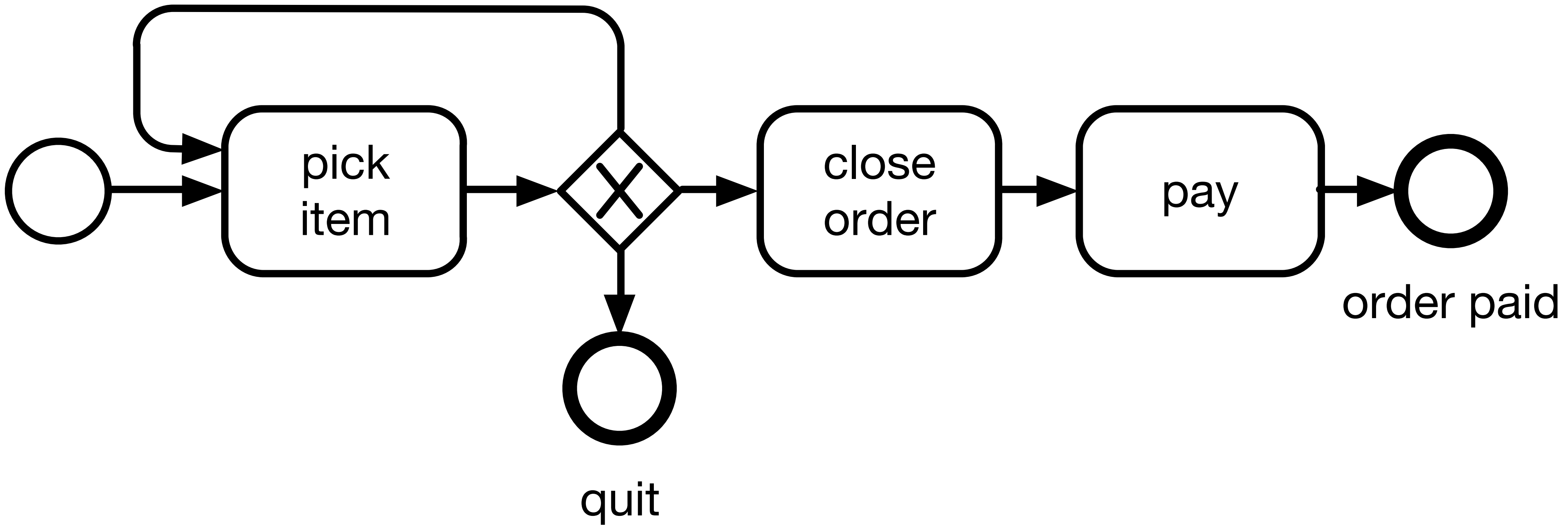
How difficult is it?

You may pick items, close and pay an order.
Whenever you close an order, eventually you have to pay.

Questions

Can you model this process in BPMN?

How difficult is it?

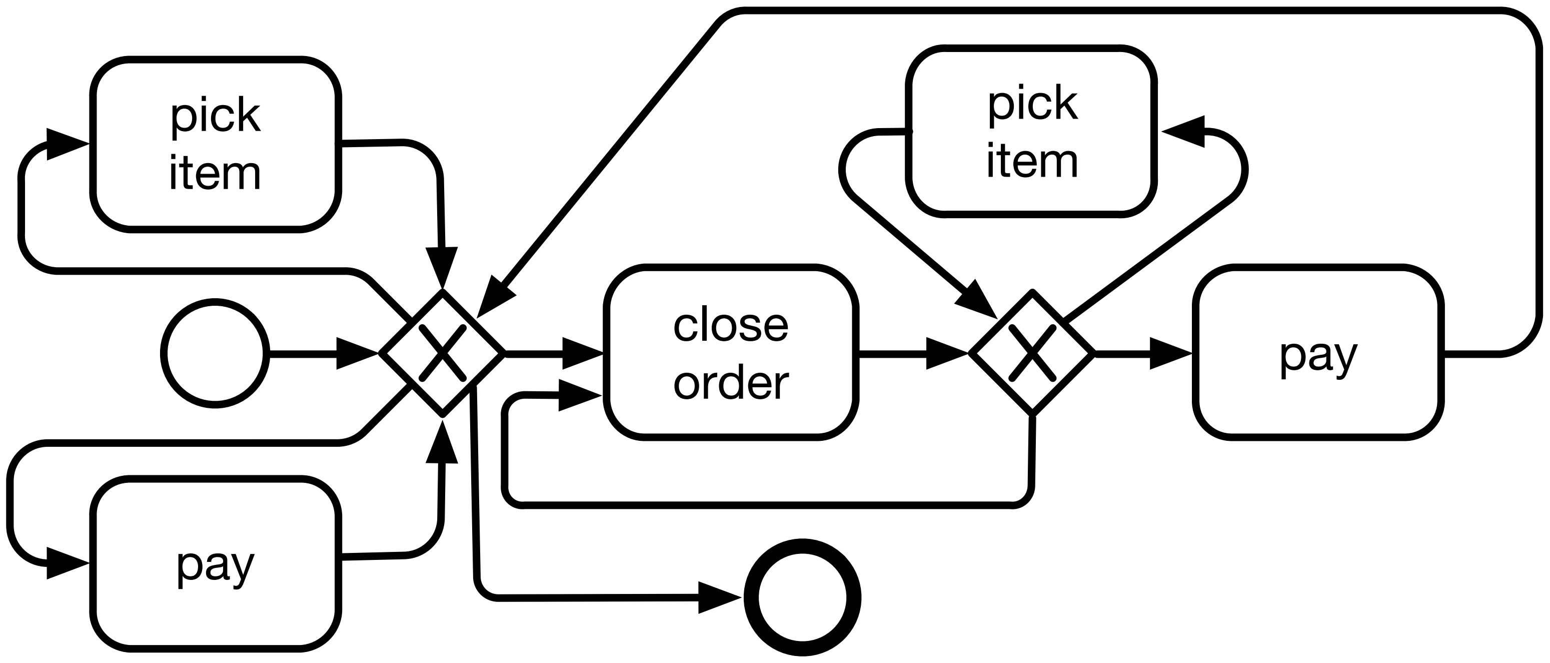


simple, correct, but **not general enough**!

Questions

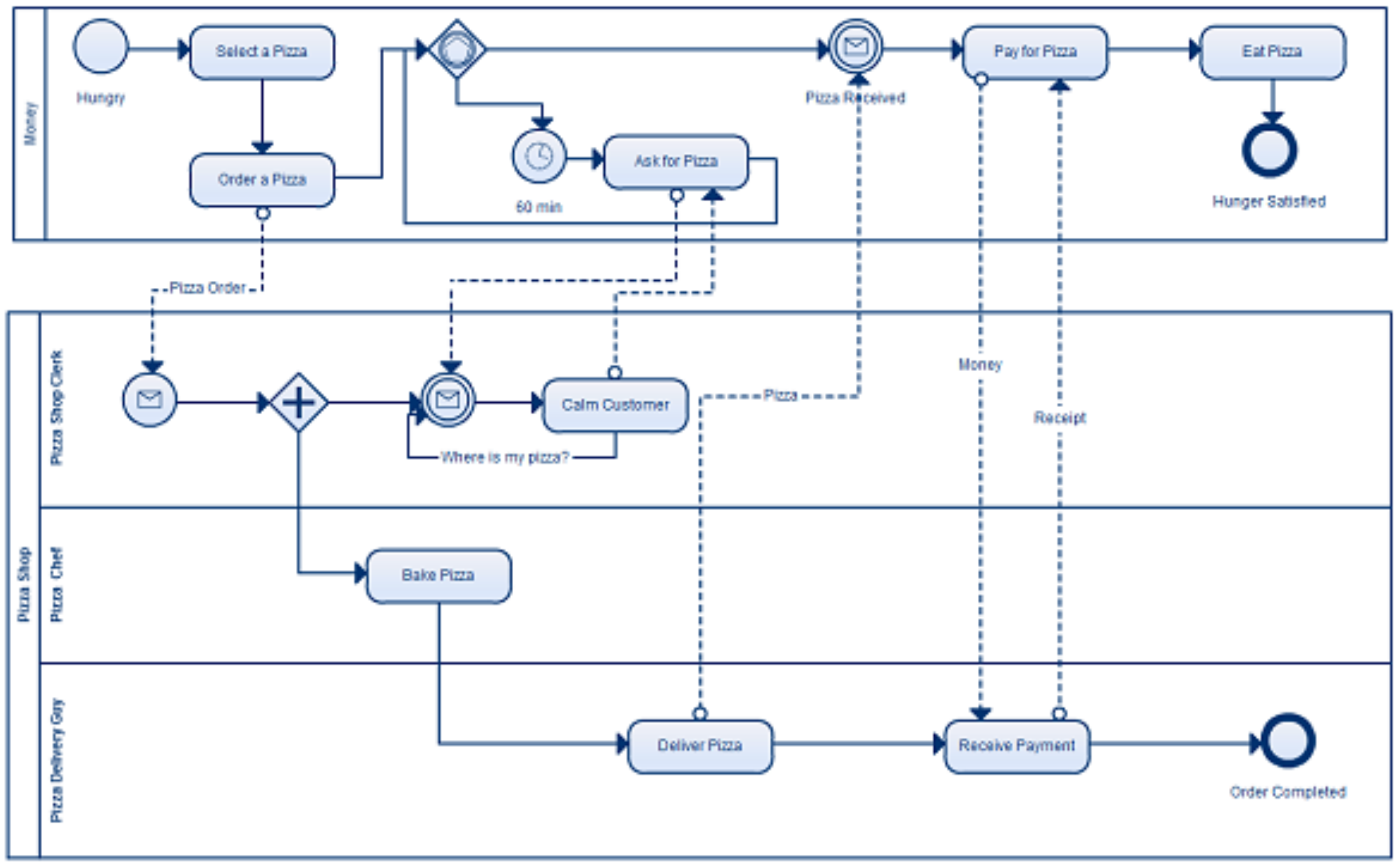
Can you model this process in BPMN?

How difficult is it?



correct, general, but **unreadable**!

Favourite Italian food?

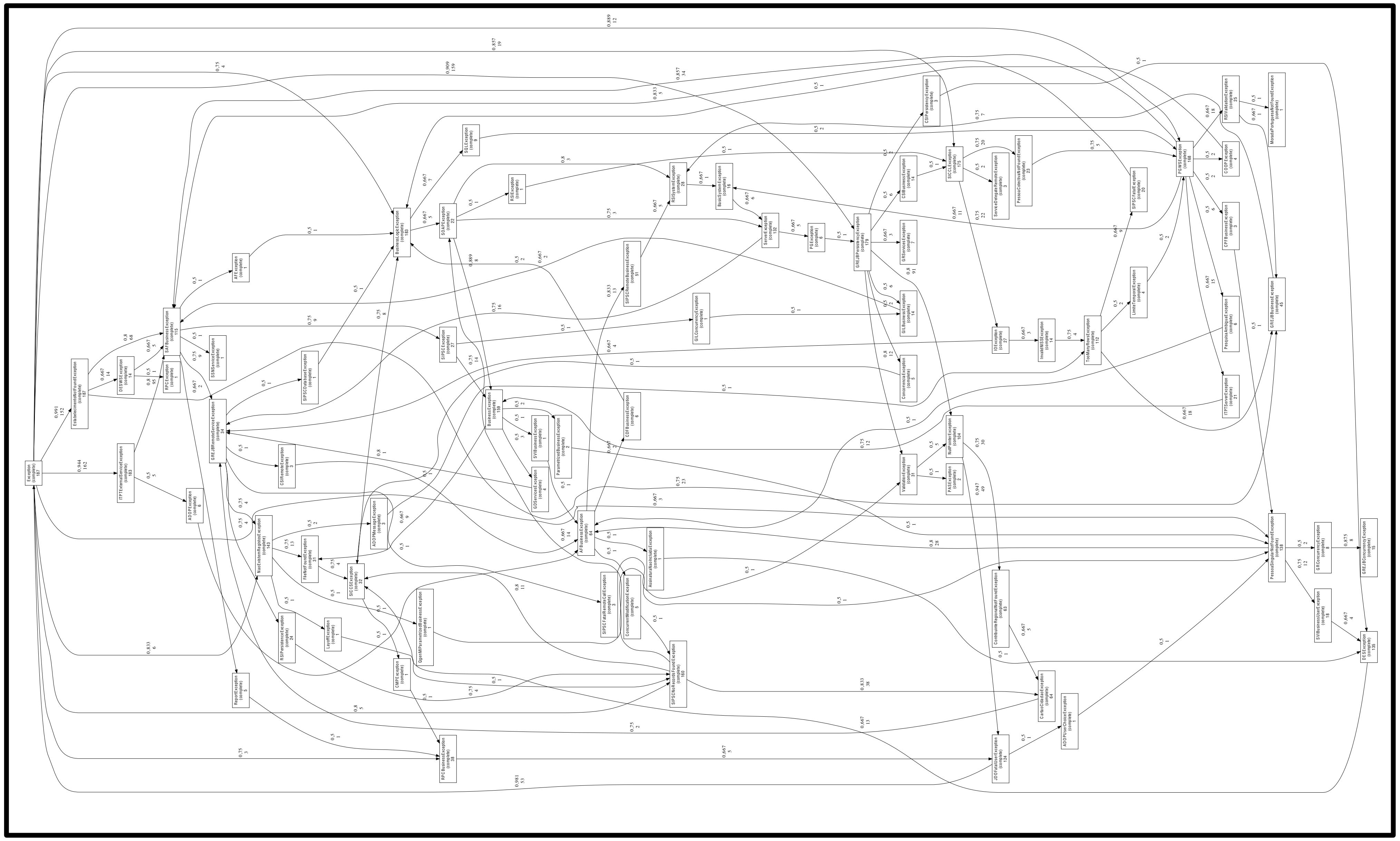


Order-to-delivery

Favourite Italian food?

Order-to-delivery

Favourite Italian food?



Healthcare

Favourite Italian food?

Healthcare

Our goal

Model

represents

Reality

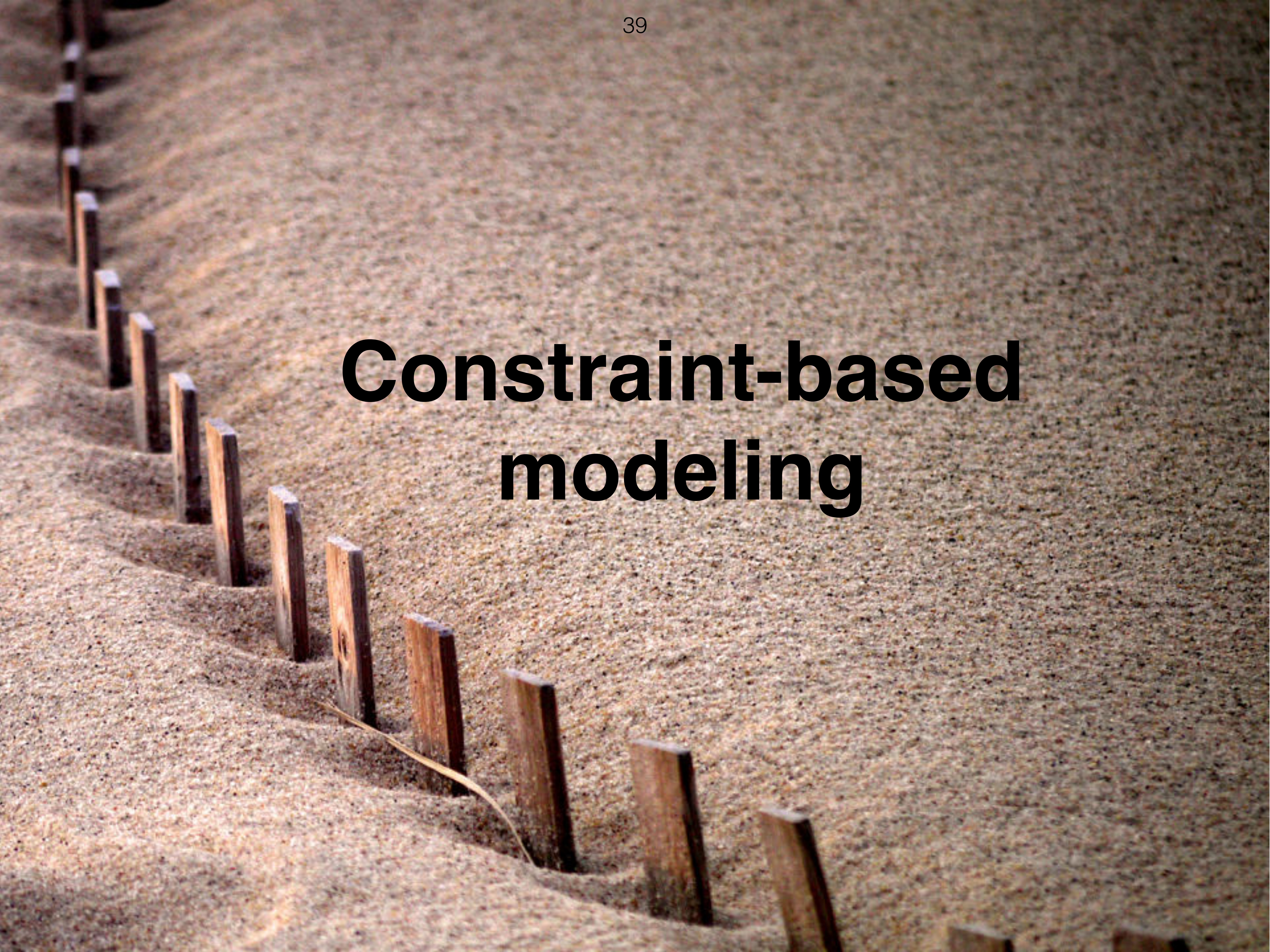
Our goal

Model

represents

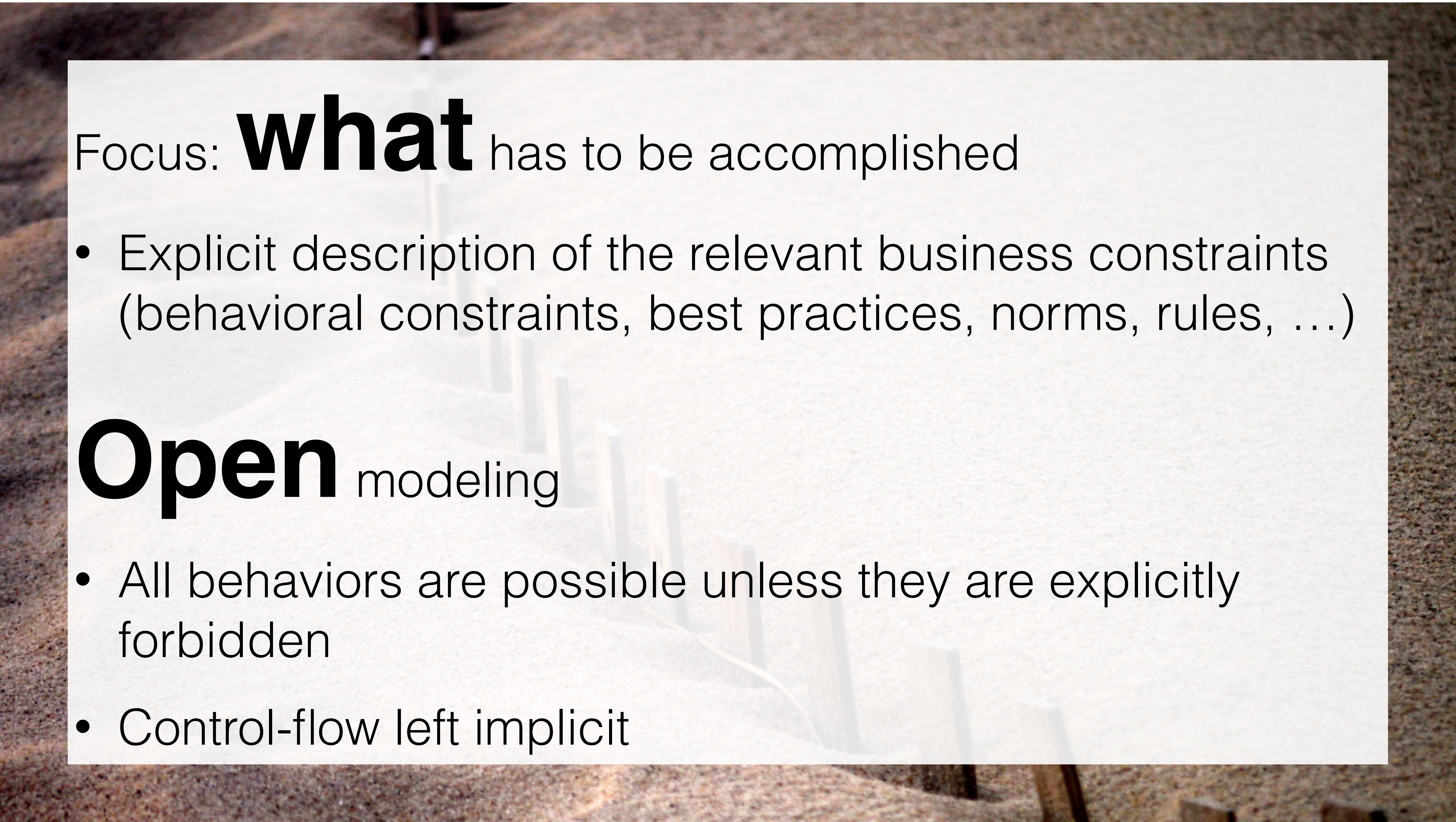
A **class** of spaghetti models, not all of them!

Reality



Constraint-based modeling

Idea



Focus: **what** has to be accomplished

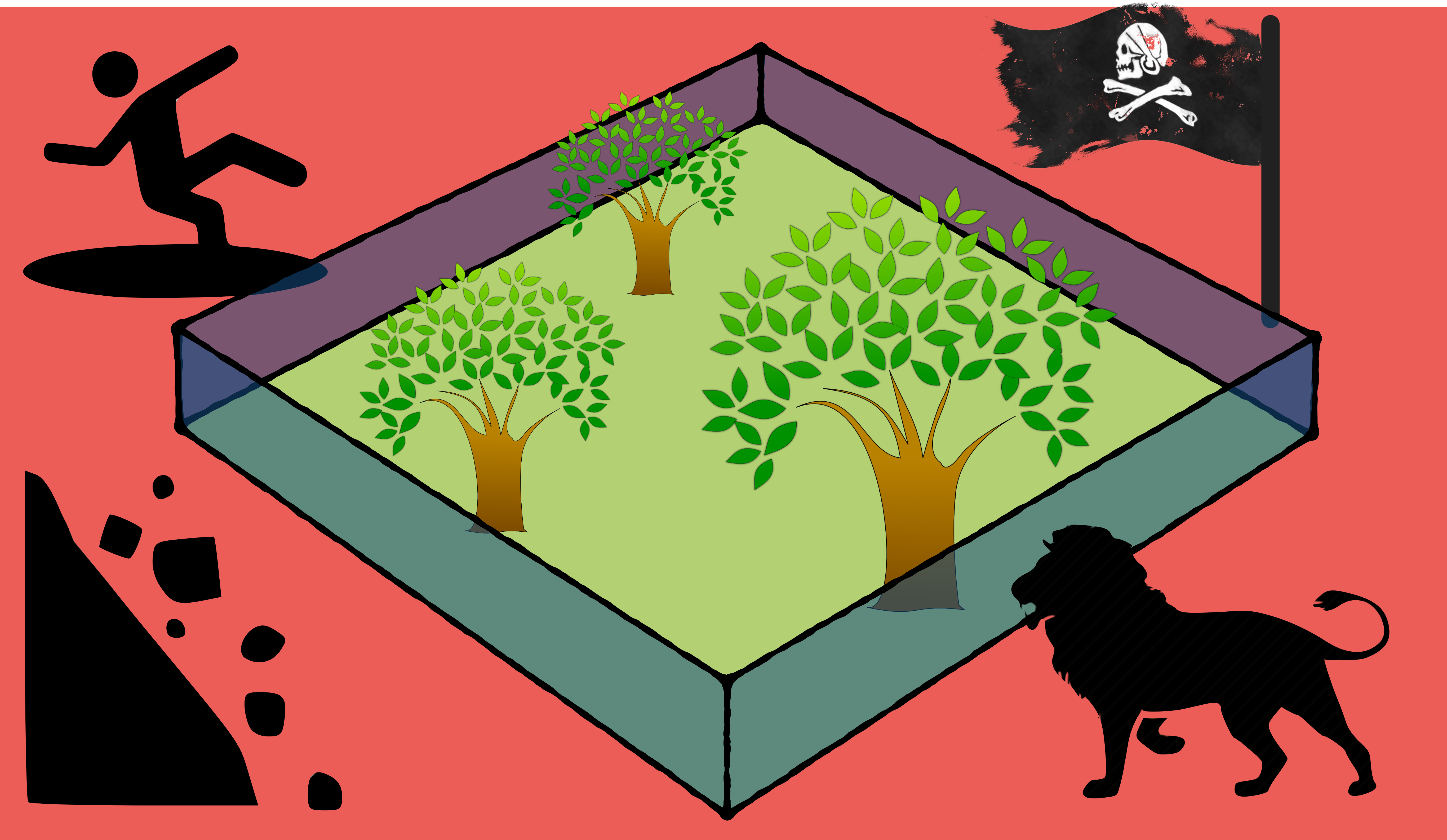
- Explicit description of the relevant business constraints (behavioral constraints, best practices, norms, rules, …)

Open modeling

- All behaviors are possible unless they are explicitly forbidden
- Control-flow left implicit

Organizational boundaries

Constraint-based modeling



Constraint-based modeling

Declare

- A constraint-based extensible language for flexible process modeling
- An execution engine for constraint-based processes
- Originally proposed by Pesic and van der Aalst
- Formalized by Pesic and myself (in our respective PhDs)

Constraints

- Much richer than the simple “precedence sequence flow connector” of imperative languages
- Supports:
    - **Presence/absence** of tasks (do/not do … N times)
    - **Negative** relationships (it is forbidden to…)
    - **Atemporal** relationships (no order imposed)
    - **Temporal** relationships, with different strengths

Global perspective

Constraints have to be globally satisfied along an entire process execution

- May stay quiescent or require events (not) to happen depending on the context

Example: if the conference format is changed, participants have to be notified

- The notification is not required to be immediate
- The format may change multiple times, or not change at all

Composing constraints

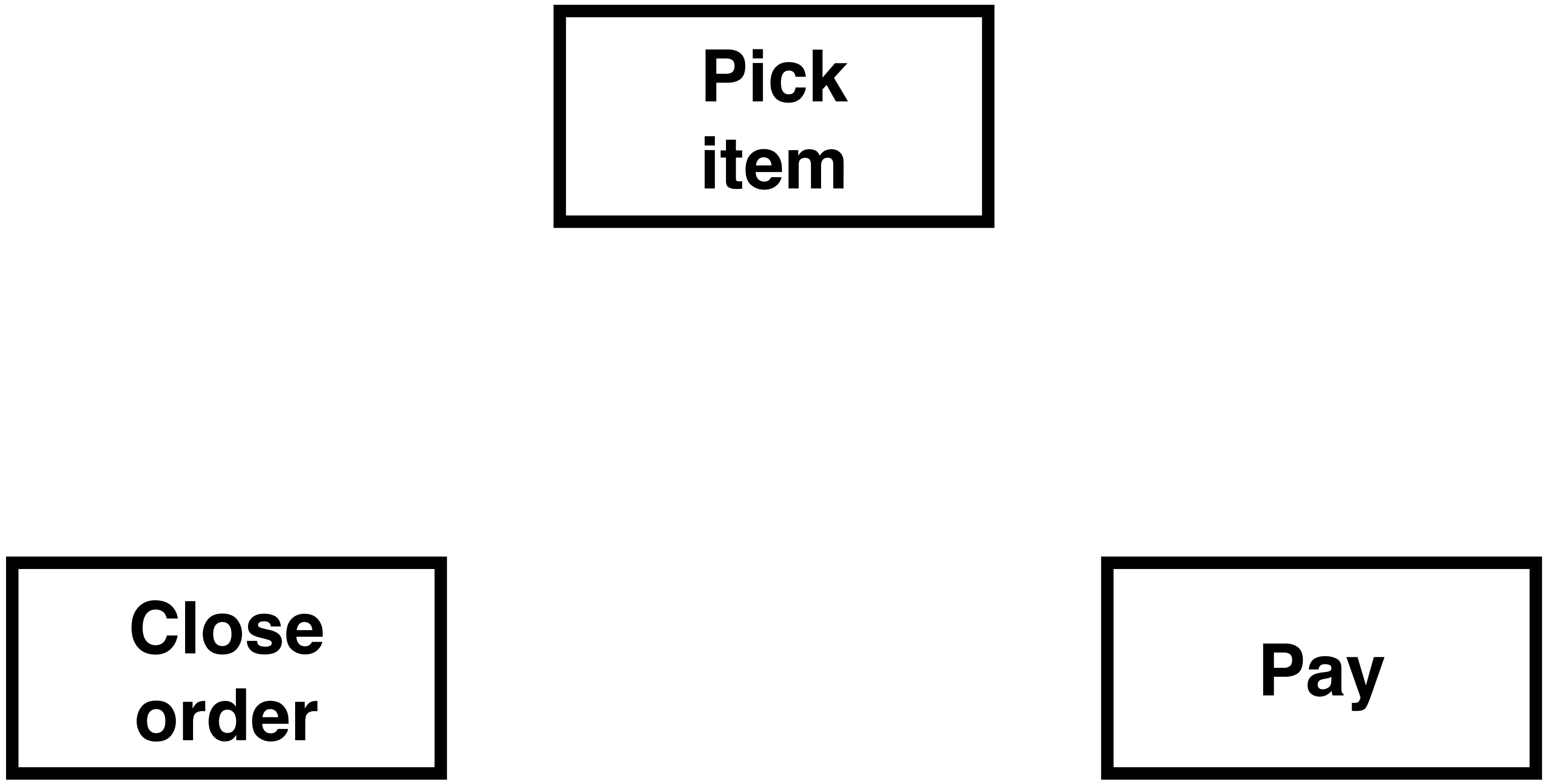
Individual constraints are simply composed by conjunction (they must be all satisfied)

- Elegant compositional approach
- Creates interactions („hidden dependencies“) among constraints

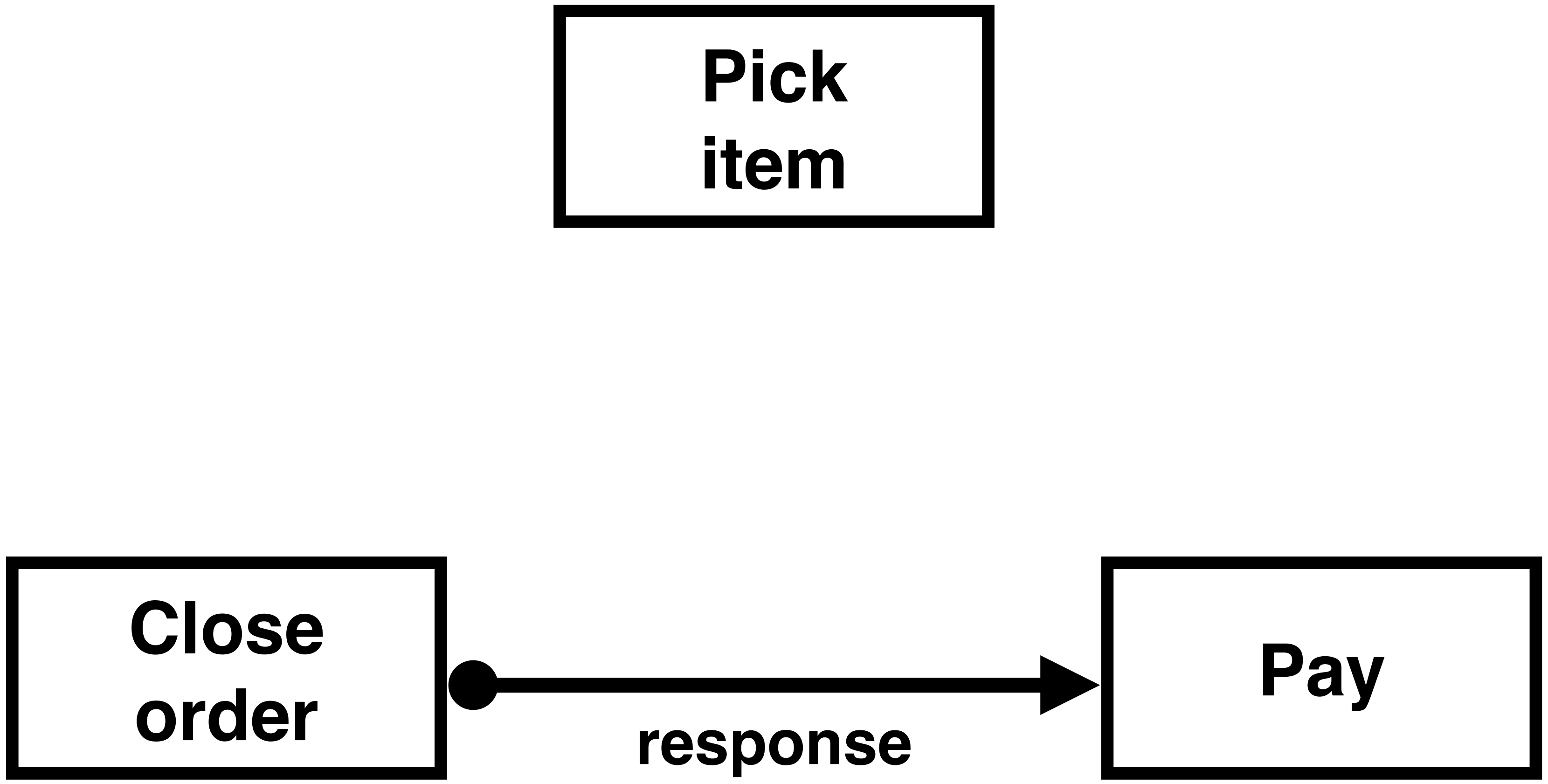
Example: if you cancel the order, you cannot pay for it. If you confirm the order, you must pay for it.

- —> It is not possible to confirm and cancel

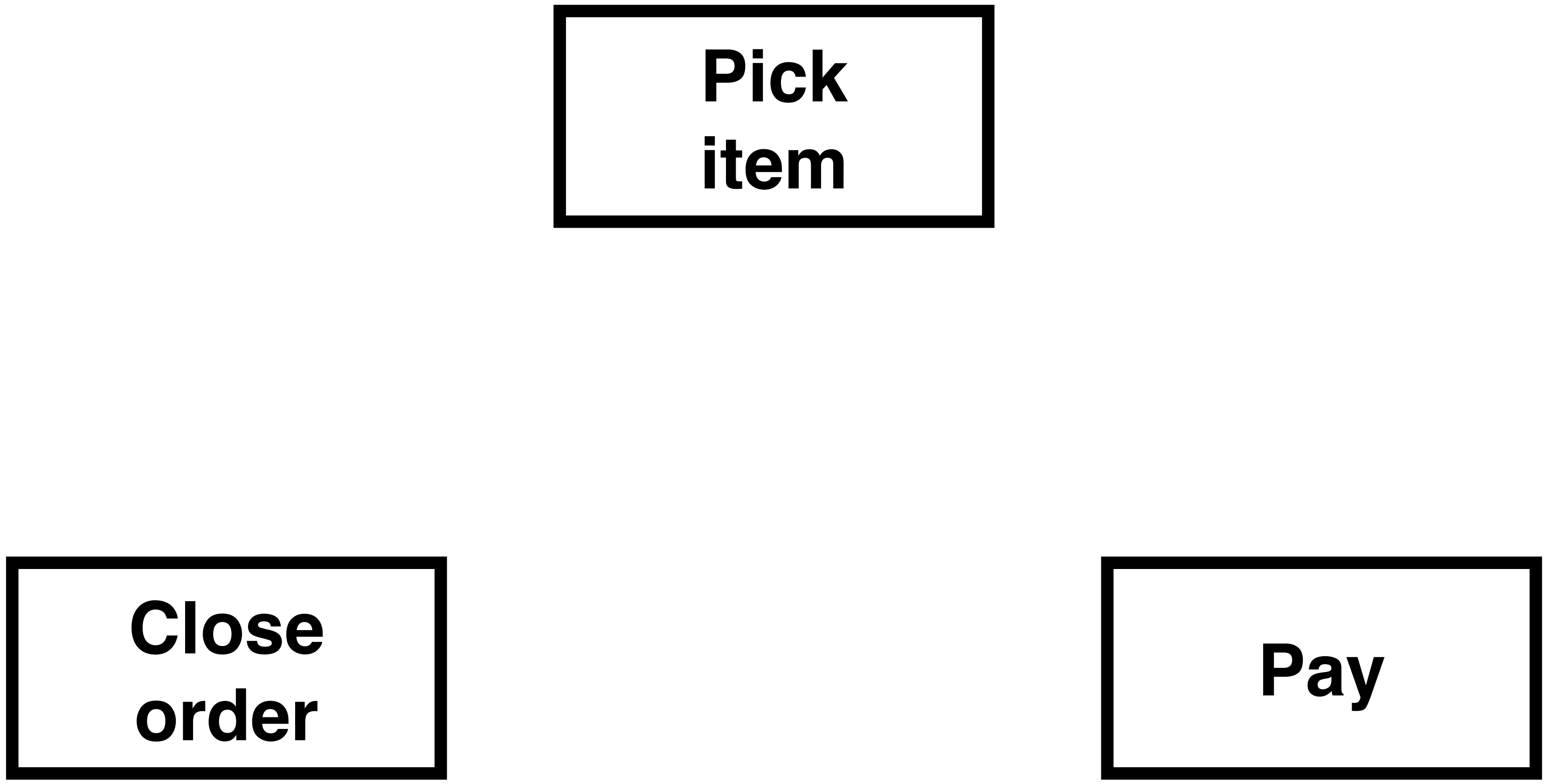
Flexible shopper



Flexible shopper

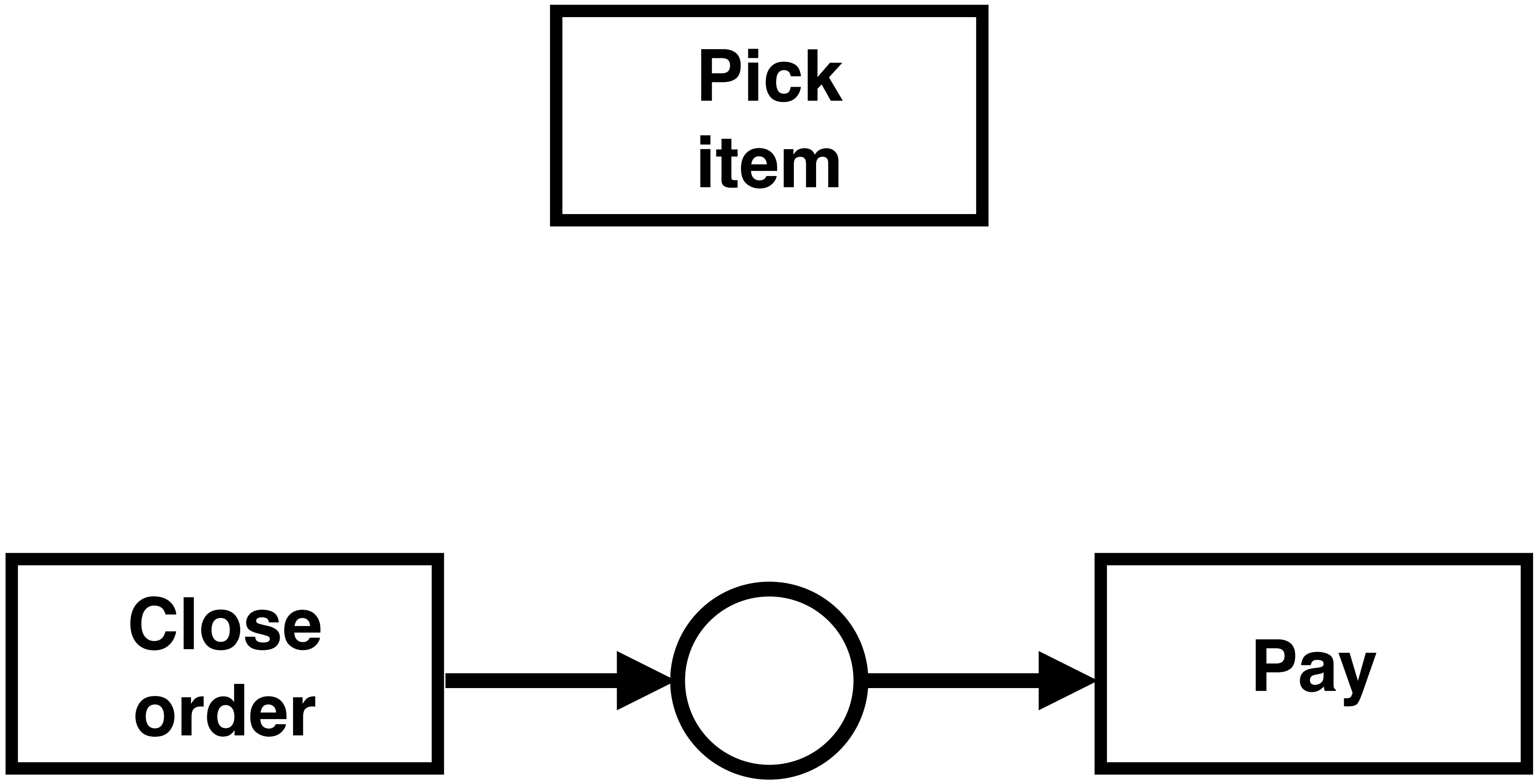


Detour: are Petri nets declarative?



A Petri net supporting any behavior on the given tasks!

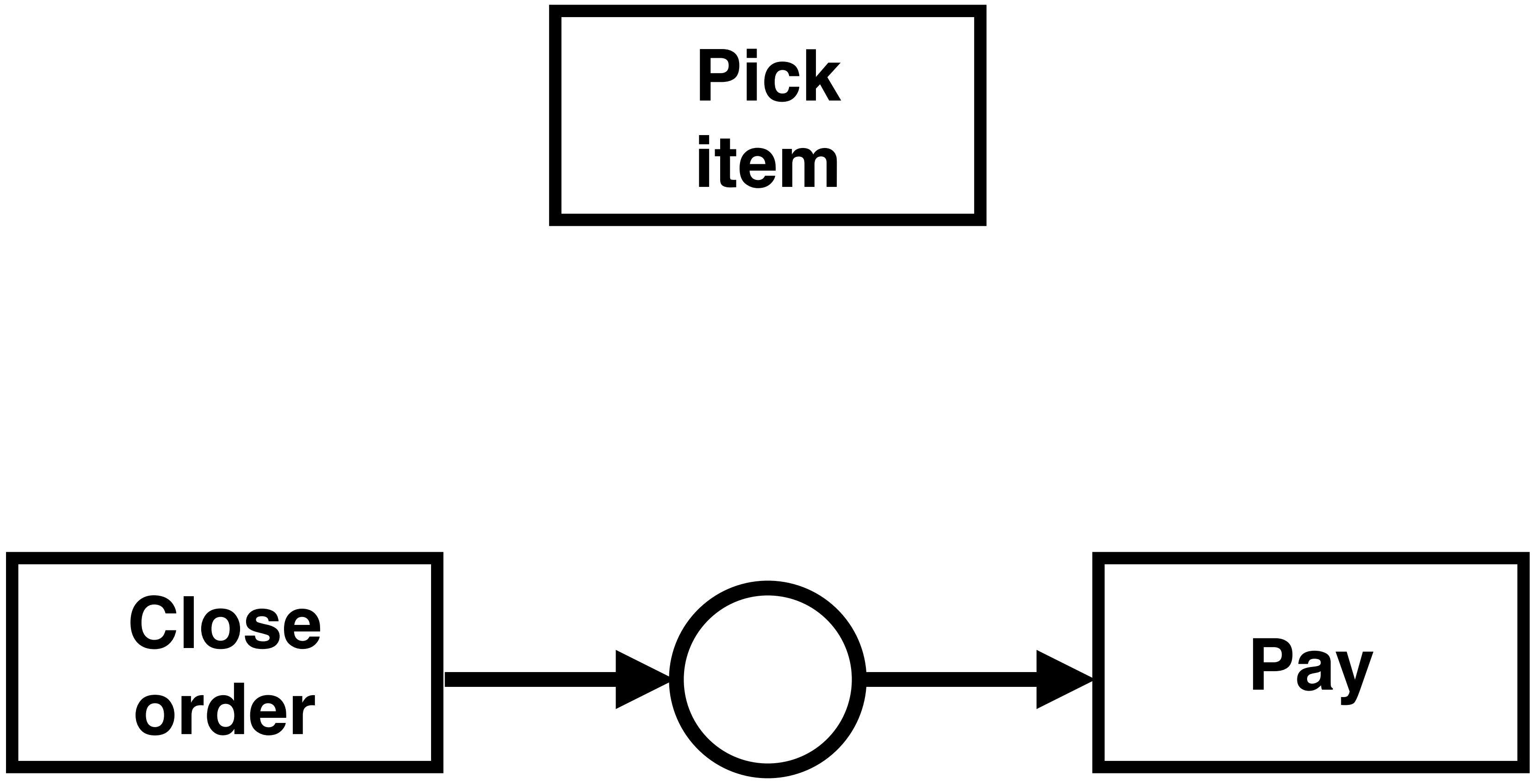
Detour:
are Petri nets declarative?



Adding places means adding temporal constraints…

But shall we interpret them backwards? Or forward?

Detour:
are Petri nets declarative?



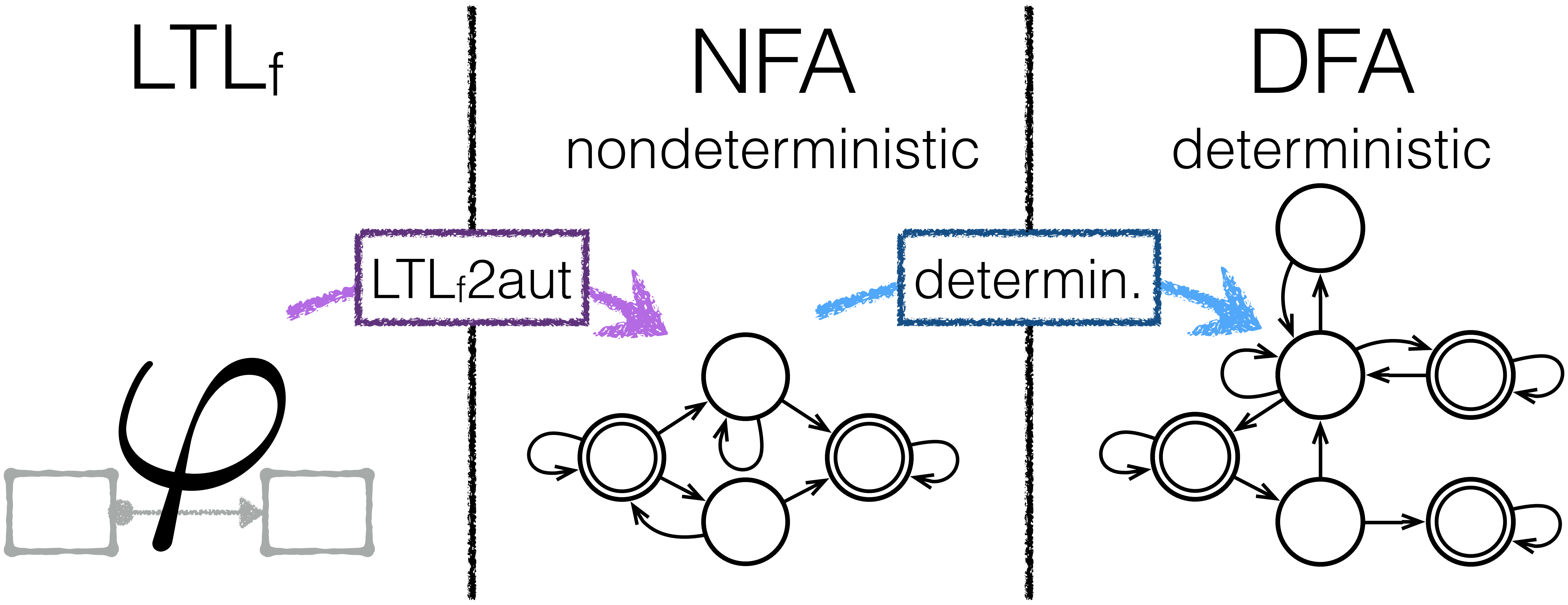
Atemporal and negative constraints cannot be added explicitly, nor incrementally.

Declare at work

From Declare to logics to automata

- Observation: Declare constraints are formalized using LTL over finite traces (LTL_f)
- LTL_f corresponds to the star-free fragment of regular expressions
- Intimate connection with finite-state automata

Declare -> automata



See keynote by Giuseppe De Giacomo
(Tomorrow at BPM 2021)

Our vision realized

LTL_f

NFA

nondeterministic

DFA

deterministic

φ

LTL_f2aut

determin.

exponential blow-up

exponential blow-up

Our vision realized

LTL_f | NFA nondeterministic | DFA deterministic

φ

LTL_f2aut

determin.

exponential blow-up

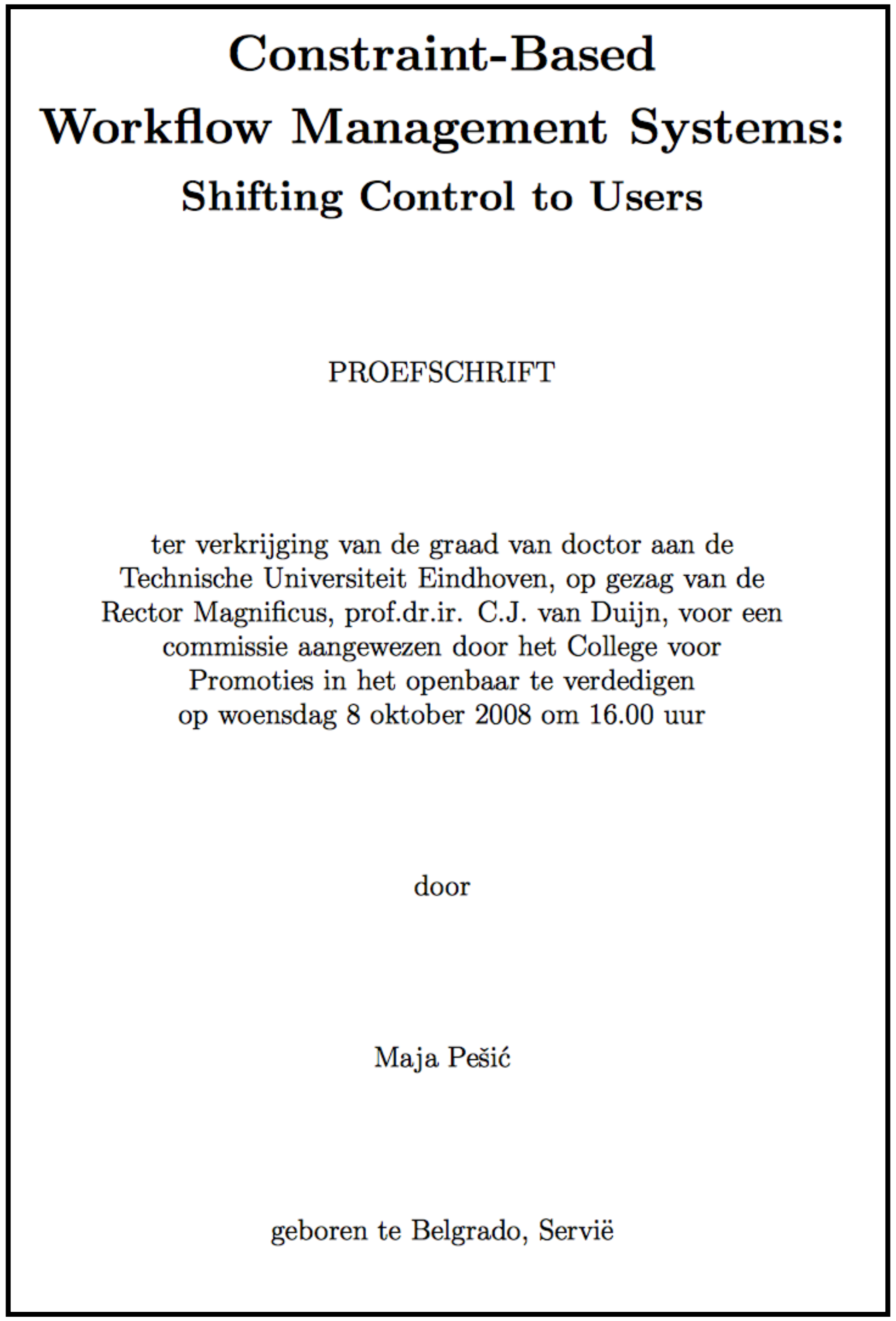
exponential blow-up

Basis for reasoning, execution, process mining.
Built for each single constraints (fine-grained feedback) and for the entire model (constraint interplay).

Some history: 3 waves

First wave

Declare, its formalization, verification, and execution


Constraint-Based
Workflow Management Systems:
Shifting Control to Users

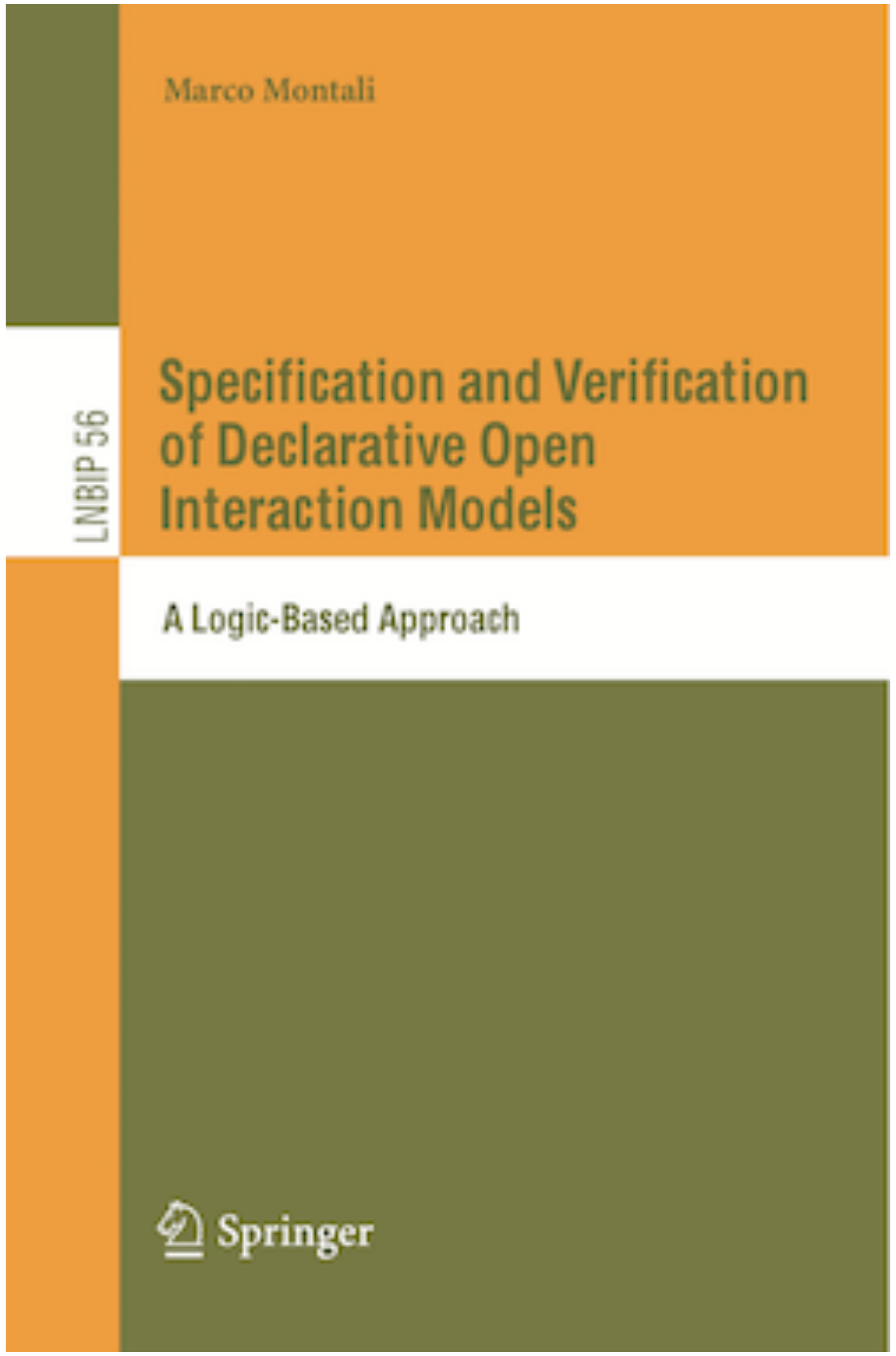
PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Technische Universiteit Eindhoven, op gezag van de
Rector Magnificus, prof.dr.ir. C.J. van Duijn, voor een
commissie aangewezen door het College voor
Promoties in het openbaar te verdedigen
op woensdag 8 oktober 2008 om 16.00 uur

door

Maja Pešić

geboren te Belgrado, Servië


Marco Montali

LNBIP 56

Specification and Verification of Declarative Open Interaction Models

A Logic-Based Approach

Springer

Second wave

Process discovery and operational support

- First work at BPM 2007 - with related works in Leuven
- Claudio‘s PhD thesis and Fabrizio‘s works: more scalable algorithms
- Collaboration between Bologna, TU/e, Tartu on operational decision support (in particular, monitoring)

Second wave

Parallel line of research

- „Human factor“ and understandability (Innsbruck, Copenhagen, St Gallen)
- Formal foundations using a variety of temporal logics over finite traces, with huge impact in AI (Rome, Rice, Bolzano) and BPM (Bolzano, Vienna)
- Model extensions and hybrid models (Tartu, Bolzano, Trento, Rome, Utrecht, Leuven, …)
- Variety of groups investigating several facets of the topic

- Parallel line of research on DCR graphs (Copenhagen)

Third wave: Rum…

… + multi-perspective models, real-life scaling

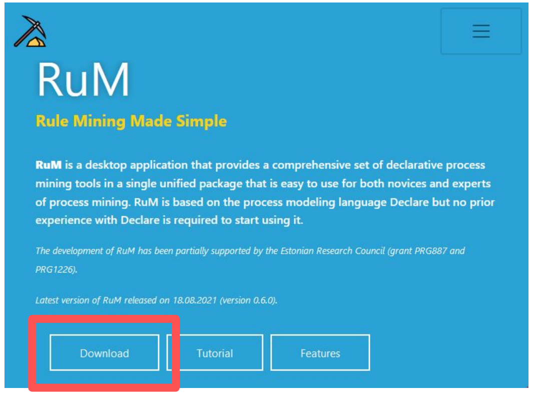
- See closing part

Declarative Process Mining with RuM

Overview of RuM

RuM - Rule Mining Made Simple

- Based on the Declare language
- Incorporates various well-known algorithms
- Available at: https://rulemining.org/
  - ~585 MB download
  - requires Java 11 JDK (available through the download link)



RuM

Rule Mining Made Simple

RuM is a desktop application that provides a comprehensive set of declarative process mining tools in a single unified package that is easy to use for both novices and experts of process mining. RuM is based on the process modeling language Declare but no prior experience with Declare is required to start using it.

The development of RuM has been partially supported by the Estonian Research Council (grant PRG887 and PRG1226).

Latest version of RuM released on 18.08.2021 (version 0.6.0).

RuM will be used in the following sections

Please start the download now - We will provide assistance

RuM download: https://rulemining.org/

Declare Example

- Consider the goal to give a tutorial

RuM download: https://rulemining.org/

Declare Example

- Consider the goal to give a tutorial

give tutorial

RuM download: https://rulemining.org/

Declare Example

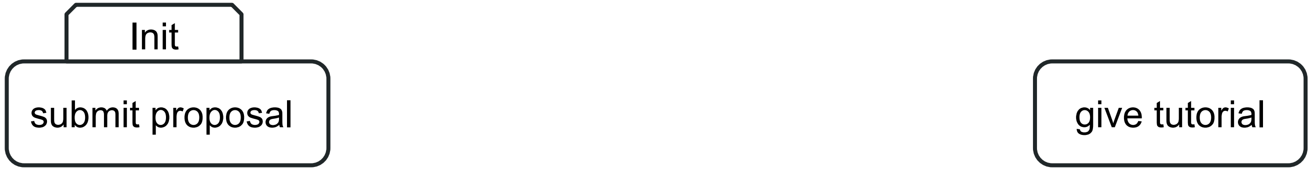
- Consider the goal to give a tutorial
- The process starts by submitting a proposal

RuM download: https://rulemining.org/

Declare Example

- Consider the goal to give a tutorial
- The process starts by submitting a proposal

init(*submit proposal*)



RuM download: https://rulemining.org/

Declare Example

- Consider the goal to give a tutorial
- The process starts by submitting a proposal
- After submitting a proposal,
  a decision is received

init(*submit proposal*)

Init
submit proposal
give tutorial

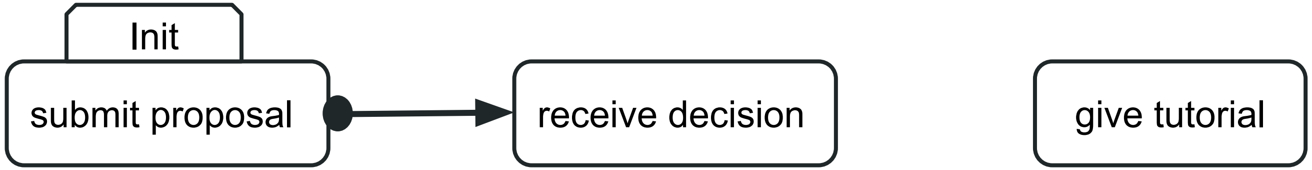
RuM download: https://rulemining.org/

Declare Example

- Consider the goal to give a tutorial
- The process starts by submitting a proposal
- After submitting a proposal, a decision is received

init(*submit proposal*)

response(submit proposal, receive decision)



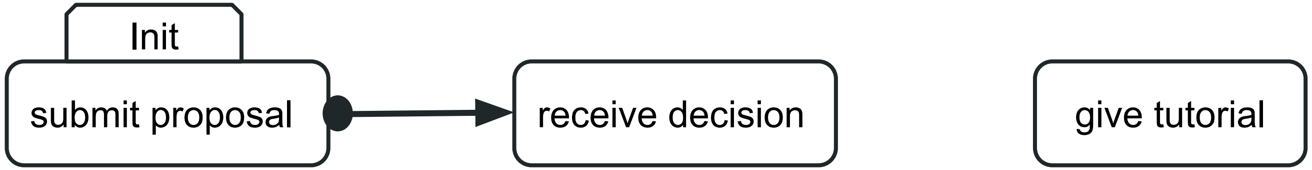
RuM download: https://rulemining.org/

Declare Example

- Consider the goal to give a tutorial
- The process starts by submitting a proposal
- After submitting a proposal,
  a decision is received
- The tutorial can be given if the decision is positive

init(*submit proposal*)

response(submit proposal,
receive decision)



RuM download: https://rulemining.org/

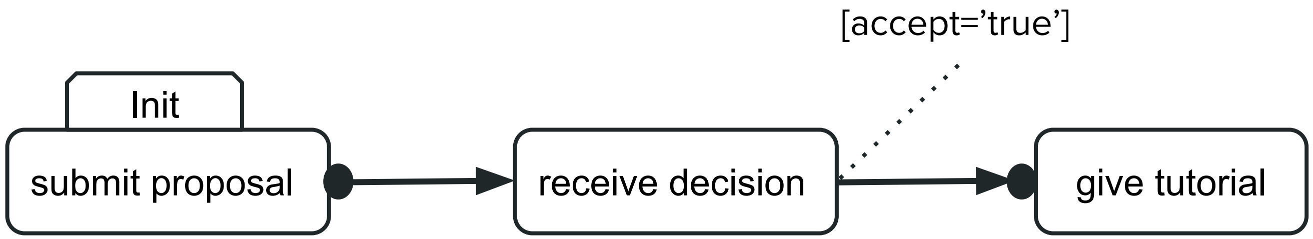
Declare Example

- Consider the goal to give a tutorial
- The process starts by submitting a proposal
- After submitting a proposal, a decision is received
- The tutorial can be given if the decision is positive

init(*submit proposal*)

response(submit proposal, receive decision)

precedence(give tutorial, receive decision [accept='true'])



Hands-on session with RuM

RuM download: https://rulemining.org/

Scenario

Overarching goal:

- Create a synthetic event log that mimics behavior found in a real event log

Inputs:

- 2 event logs of sepsis cases
  - Training set and test set
  - 391 traces each (50/50 split)
  - Completed cases only

RuM download: https://rulemining.org/

Analysis pipeline

Overarching goal

- Create a synthetic event log that mimics behavior found in a real event log

RuM download: https://rulemining.org/

Analysis pipeline

Overarching goal

- Create a synthetic event log that mimics behavior found in a real event log

Main steps

1. Discover a model

RuM download: https://rulemining.org/

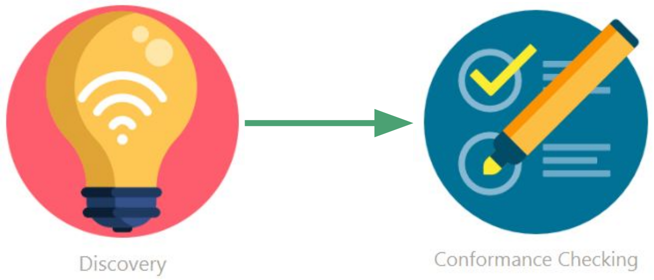
Analysis pipeline

Overarching goal

- Create a synthetic event log that mimics behavior found in a real event log

Main steps

1. Discover a model
2. Validate the model



RuM download: https://rulemining.org/

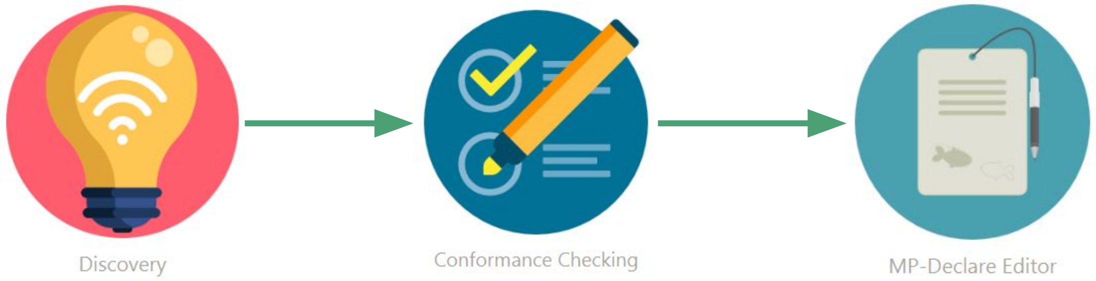
Analysis pipeline

Overarching goal

- Create a synthetic event log that mimics behavior found in a real event log

Main steps

1. Discover a model
2. Validate the model
3. Improve the model



RuM download: https://rulemining.org/

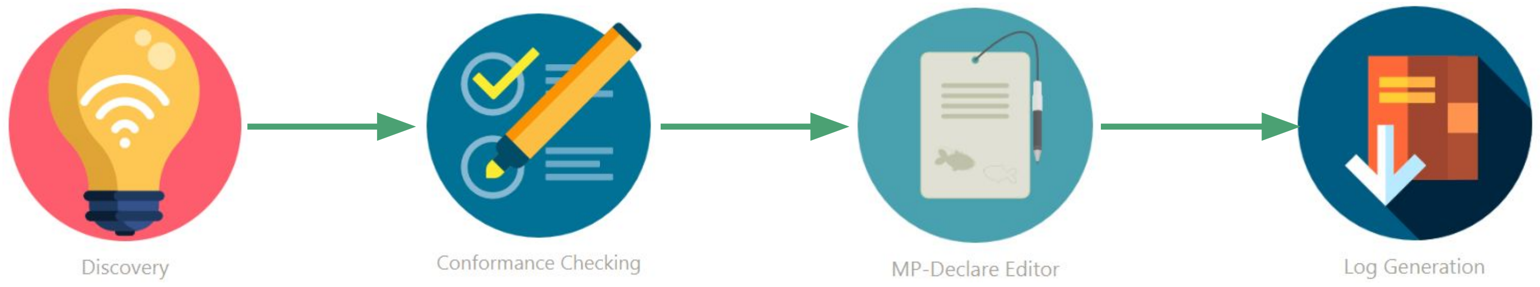
Analysis pipeline

Overarching goal

- Create a synthetic event log that mimics behavior found in a real event log

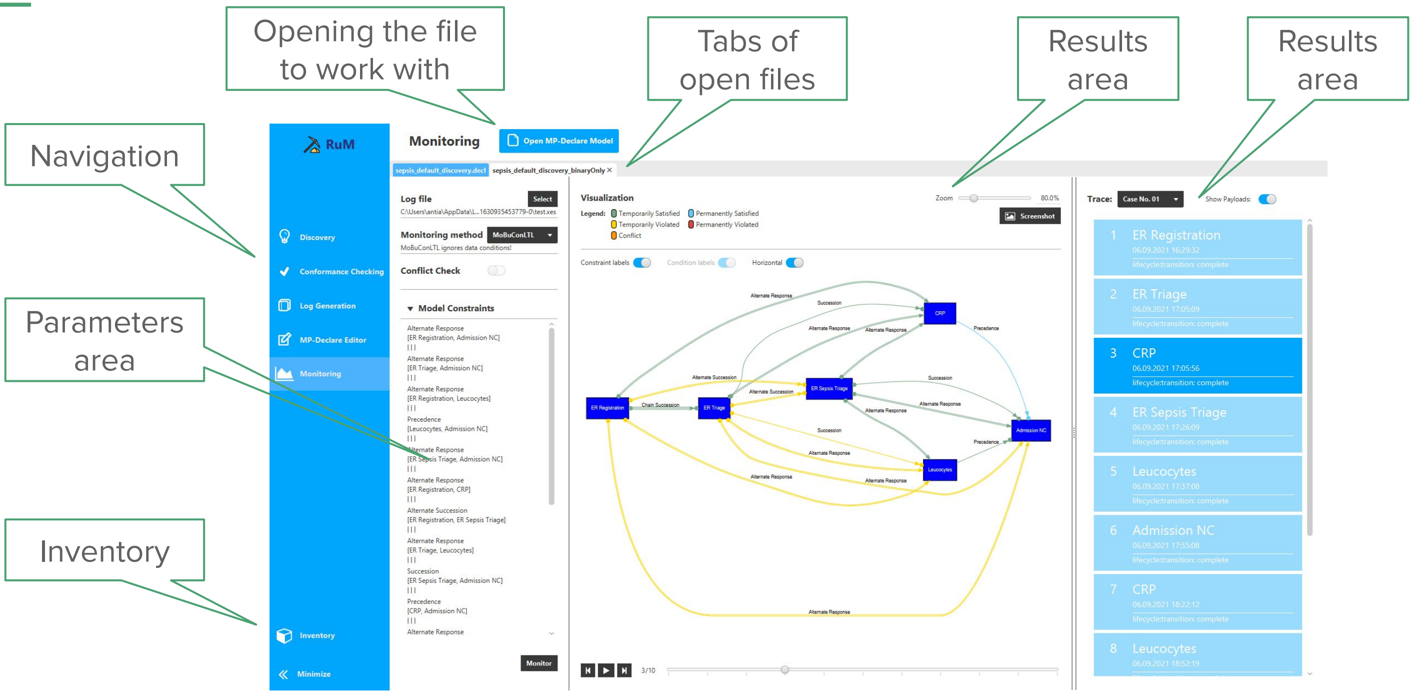
Main steps

1. Discover a model
2. Validate the model
3. Improve the model
4. Create an Event Log



RuM download: https://rulemining.org/

Tips for Using RuM



Monitoring

sepsis_default_discovery.decl | sepsis_default_discovery_binaryOnly ×

Log file
C:\Users\antia\AppData\L...1630935453779-0\test.xes

Monitoring method MoBuConLTL
MoBuConLTL ignores data conditions!

Conflict Check

Model Constraints

Alternate Response
[ER Registration, Admission NC]
| | |
Alternate Response
[ER Triage, Admission NC]
| | |
Alternate Response
[ER Registration, Leucocytes]
| | |
Precedence
[Leucocytes, Admission NC]
| | |
Alternate Response
[ER Sepsis Triage, Admission NC]
| | |
Alternate Response
[ER Registration, CRP]
| | |
Alternate Succession
[ER Registration, ER Sepsis Triage]
| | |
Alternate Response
[ER Triage, Leucocytes]
| | |
Succession
[ER Sepsis Triage, Admission NC]
| | |
Precedence
[CRP, Admission NC]
| | |
Alternate Response

Visualization

Zoom 80.0%

Legend: 🟩 Temporarily Satisfied 🟦 Permanently Satisfied
🟨 Temporarily Violated 🟥 Permanently Violated
🟧 Conflict

Constraint labels | Condition labels | Horizontal

3/10

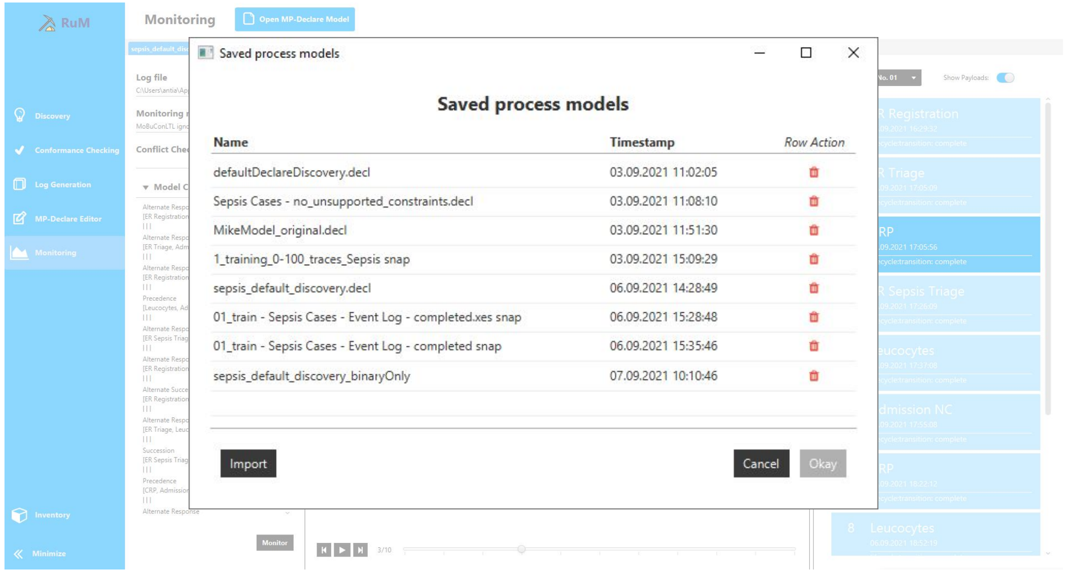
Trace: Case No. 01 Show Payloads:

1	ER Registration 06.09.2021 16:29:32 lifecycle:transition: complete
2	ER Triage 06.09.2021 17:05:09 lifecycle:transition: complete
3	CRP 06.09.2021 17:05:56 lifecycle:transition: complete
4	ER Sepsis Triage 06.09.2021 17:26:09 lifecycle:transition: complete
5	Leucocytes 06.09.2021 17:37:08 lifecycle:transition: complete
6	Admission NC 06.09.2021 17:55:08 lifecycle:transition: complete
7	CRP 06.09.2021 18:22:12 lifecycle:transition: complete
8	Leucocytes 06.09.2021 18:52:19

RuM download: https://rulemining.org/

The Inventory System

- For fast access to models and event logs
  - Reuse of models and event logs across RuM
- Allows for snapshots
  - A temporary model or event log for analysis

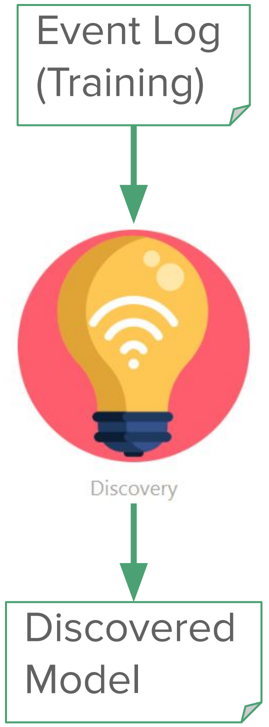


Saved process models

Name	Timestamp	Row Action
defaultDeclareDiscovery.decl	03.09.2021 11:02:05	
Sepsis Cases - no_unsupported_constraints.decl	03.09.2021 11:08:10	
MikeModel_original.decl	03.09.2021 11:51:30	
1_training_0-100_traces_Sepsis snap	03.09.2021 15:09:29	
sepsis_default_discovery.decl	06.09.2021 14:28:49	
01_train - Sepsis Cases - Event Log - completed.xes snap	06.09.2021 15:28:48	
01_train - Sepsis Cases - Event Log - completed snap	06.09.2021 15:35:46	
sepsis_default_discovery_binaryOnly	07.09.2021 10:10:46	

RuM download: https://rulemining.org/

Task 1: Discover a Model



Inputs:

- 01_train - Sepsis Cases - Event Log - completed.xes

Parameters:

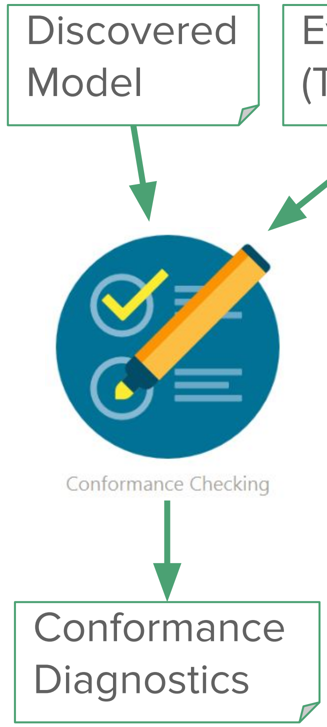
- Declare Miner
- Support: 90%
- All reductions
- Unary templates
- Positive Binary Templates
- Activity support: 95%

Helpful tips:

- Template descriptions can be found in the templates panel
- The textual view can be helpful for understanding the model
- Activity support can be changed after the discovery
- Activity support can be written with the keyboard

RuM download: https://rulemining.org/

Task 2: Validate the Model



Inputs:

- Discovered model from Task 1
- 02_test - Sepsis Cases - Event Log - completed.xes

Parameters:

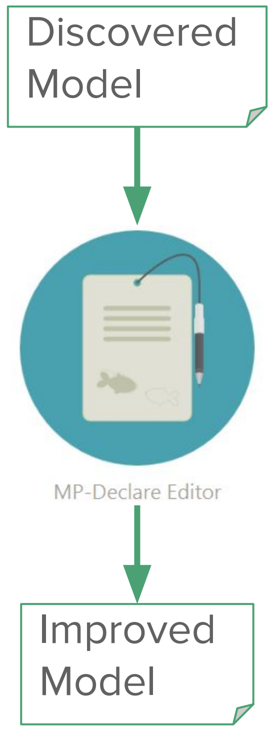
- Declare Analyzer

Helpful tips:

- Results are displayed in groups
- Each group can be opened
- Each line in a group can be clicked
- Results can be explored by traces and by constraints
- Results can be sorted

RuM download: https://rulemining.org/

Task 3: Improve the Model



Inputs:

- Discovered model from Task 1

What to modify:

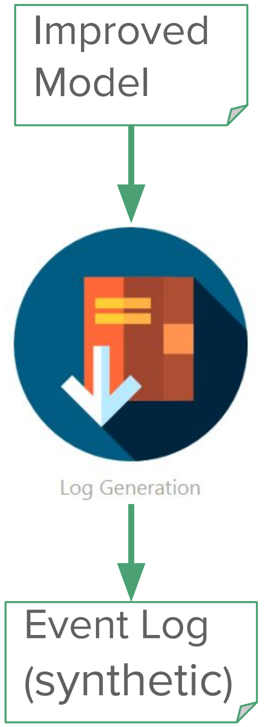
- Remove the constraints that caused 50 or more violations
  - (based on the Task 2)

Helpful tips:

- You can switch back and forth between the different views of RuM
- Constraints can be sorted by clicking on the column title
- Constraints can be modified using the 'Row actions' column

RuM download: https://rulemining.org/

Task 4: Create an Event Log



Inputs:

- Improved model from Task 3

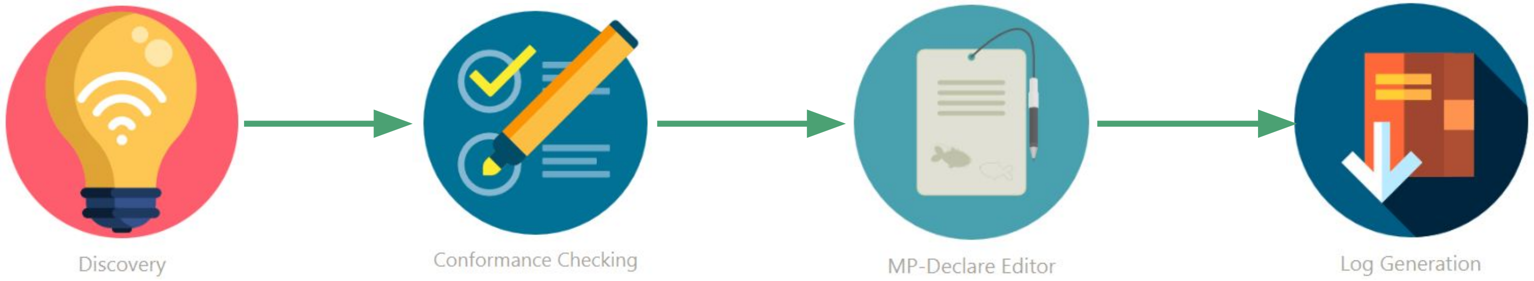
Parameters:

- AlloyLogGenerator
- Minimum events: 5
- Maximum events: 10
- Positive traces: 50
- Positive vacuous: 0%
- Negative traces: 0
- Negative vacuous: 0%

Helpful tips:

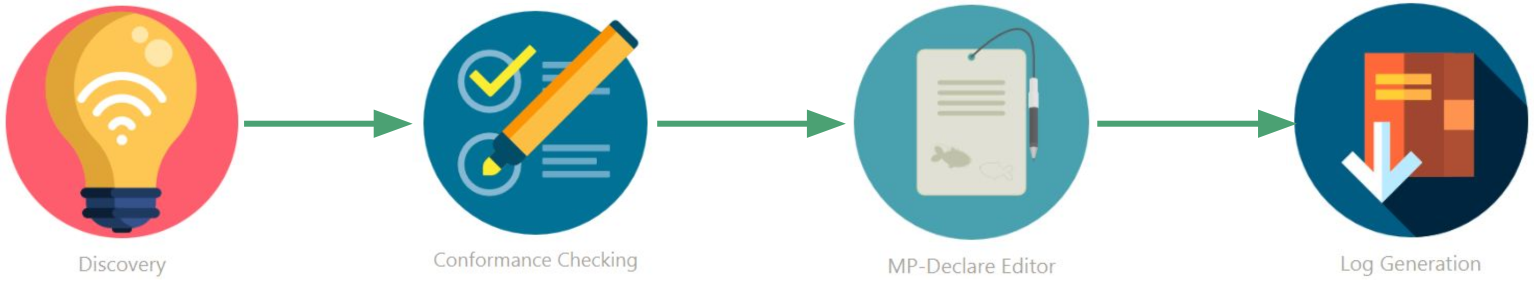
- The created event log can be seen on the left
- It is possible to explore each trace individually
- It is possible to explore the data payloads

Recap and Additional Functionalities



What more could have been done?

Recap and Additional Functionalities

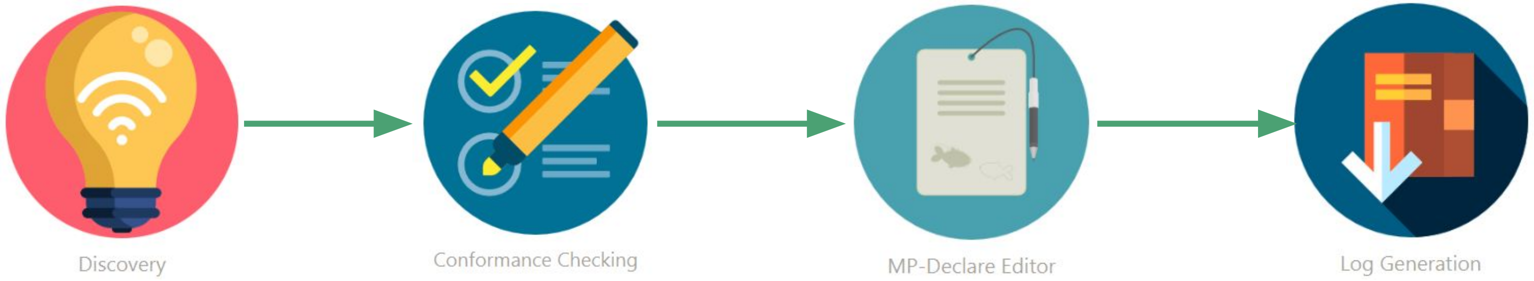


What more could have been done?

Discover a model

- MINERful
- Various templates
- Support thresholds
- Data conditions

Recap and Additional Functionalities



What more could have been done?

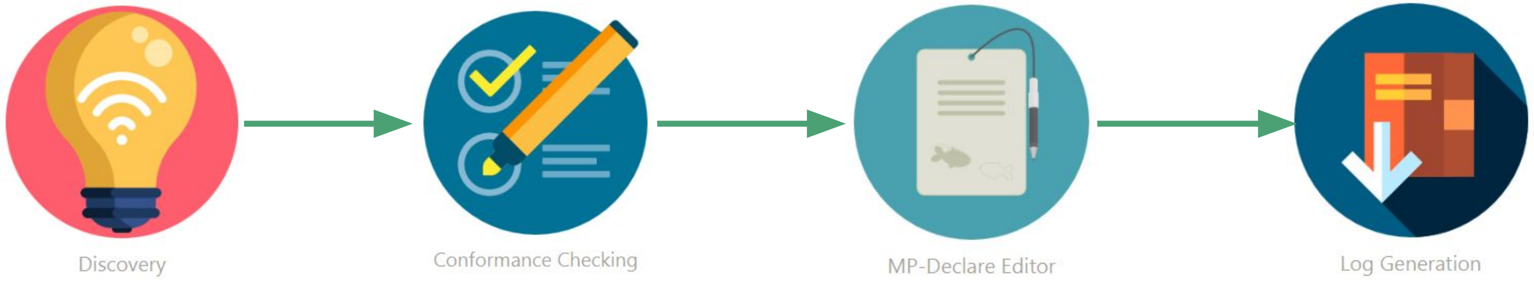
Discover a model

- MINERful
- Various templates
- Support thresholds
- Data conditions

Validate the model

- Alignments
- Monitoring
- Export fulfilled
- Assess statistics

Recap and Additional Functionalities



What more could have been done?

Discover a model

- MINERful
- Various templates
- Support thresholds
- Data conditions

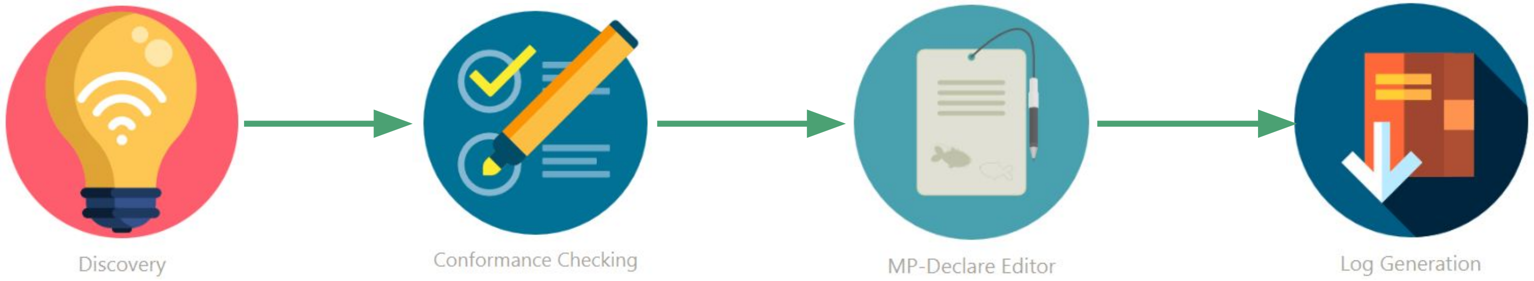
Validate the model

- Alignments
- Monitoring
- Export fulfilled
- Assess statistics

Improve the model

- Adding constraints
- Adding activities
- Data conditions
- Textual input
- Voice input

Recap and Additional Functionalities



What more could have been done?

Discover a model

- MINERful
- Various templates
- Support thresholds
- Data conditions
- Time conditions

Validate the model

- Alignments
- Monitoring
- Export subsets
- Assess statistics

Improve the model

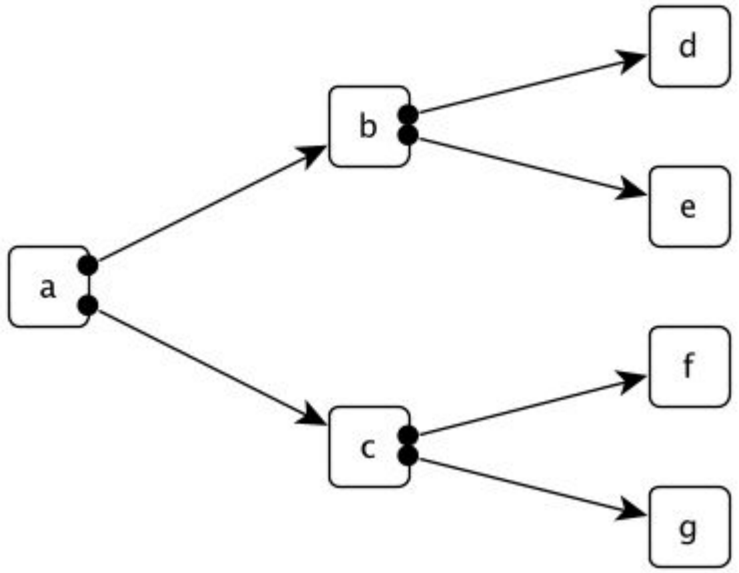
- Adding constraints
- Adding activities
- Data conditions
- Textual input
- Voice input

Create an event log

- MINERful Log Generator
- Negative traces
- Vacuous traces
- Data payloads

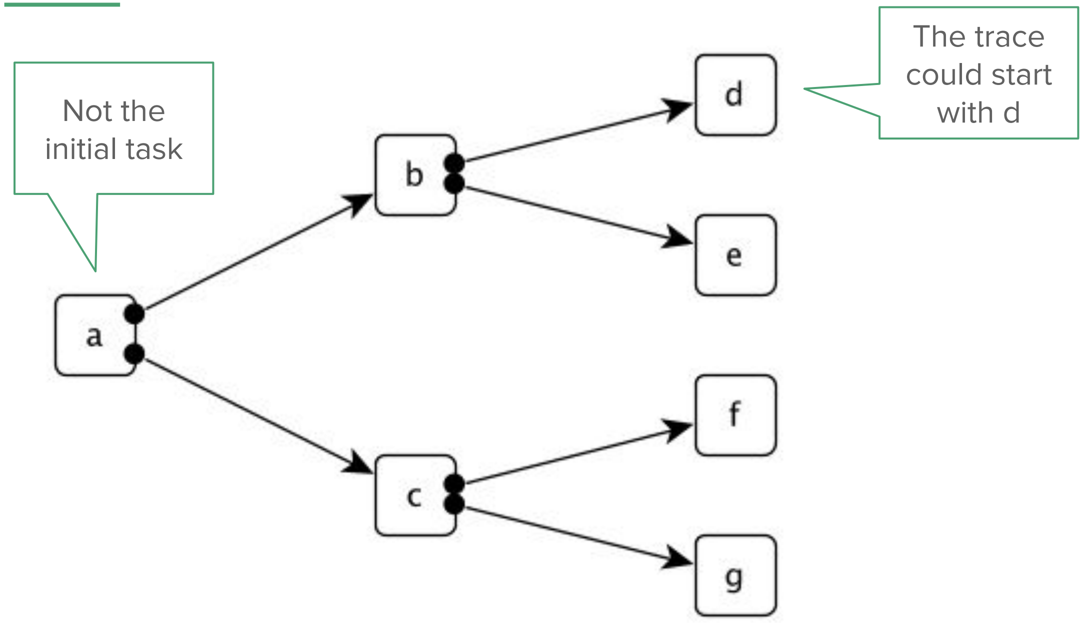
Some Research Opportunities

About the visual notation



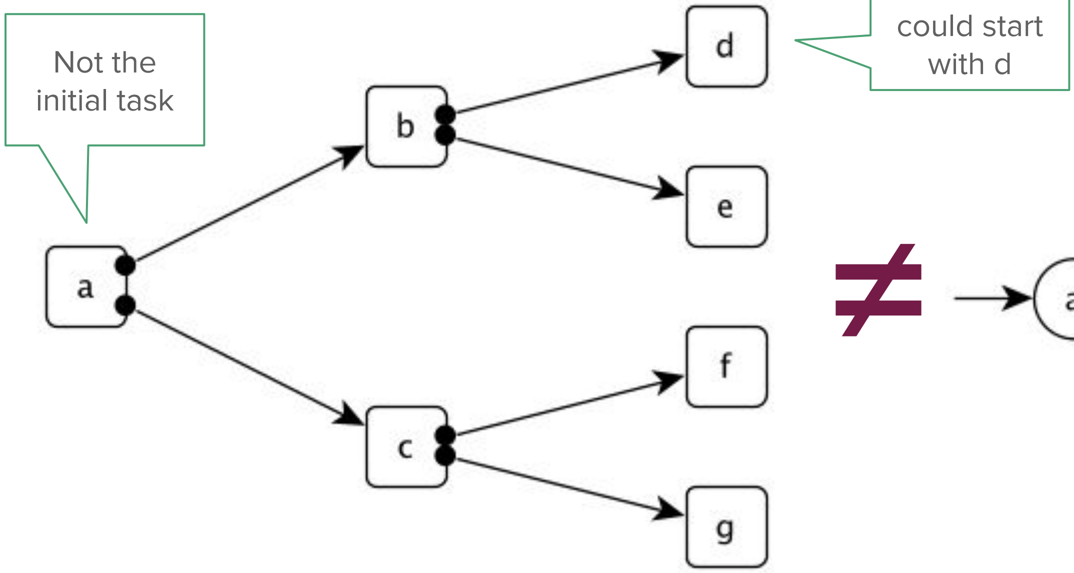
Possible traces: ⟨d, f, b, a, a, c, b, b, g, f, e, g, d⟩
⟨g, c, c, b, b, c, d, e, f, g, f, e⟩

About the visual notation

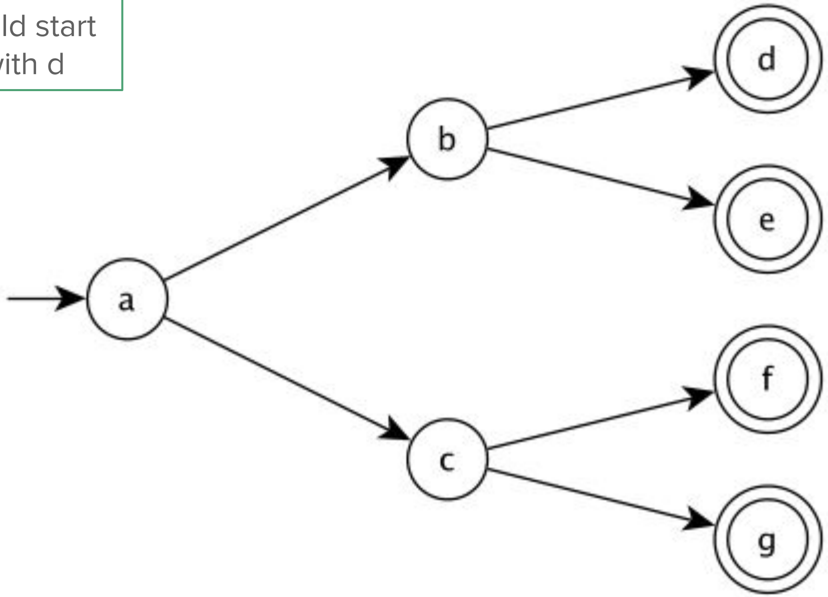


Possible traces: ⟨d, f, b, a, a, c, b, b, g, f, e, g, d⟩
⟨g, c, c, b, b, c, d, e, f, g, f, e⟩

About the visual notation

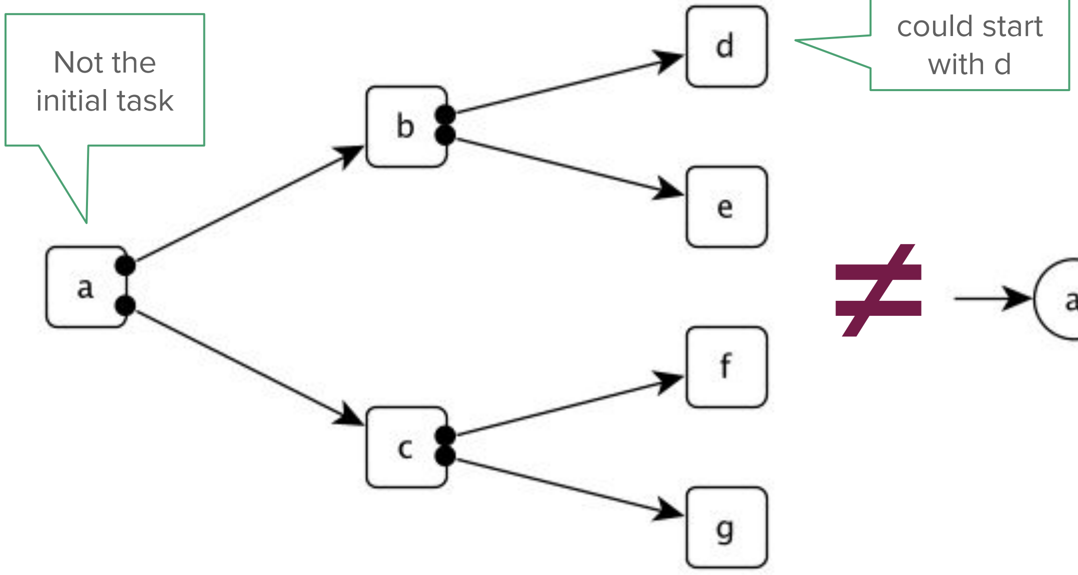


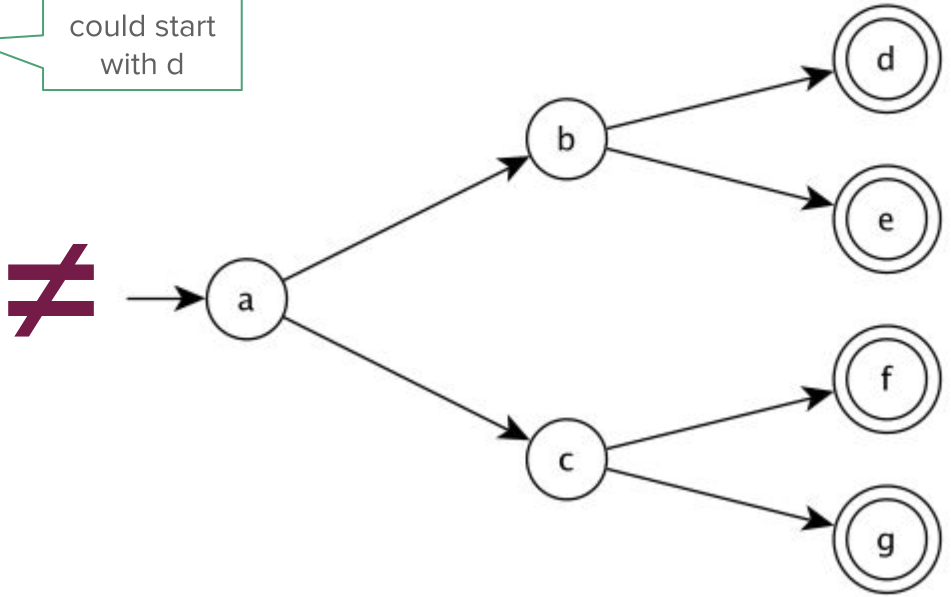
$\neq$



Possible traces: ⟨d, f, b, a, a, c, b, b, g, f, e, g, d⟩
⟨g, c, c, b, b, c, d, e, f, g, f, e⟩

About the visual notation





Possible traces: ⟨d, f, b, a, a, c, b, b, g, f, e, g, d⟩
⟨g, c, c, b, b, c, d, e, f, g, f, e⟩

Hybrid (mixed-paradigm) models

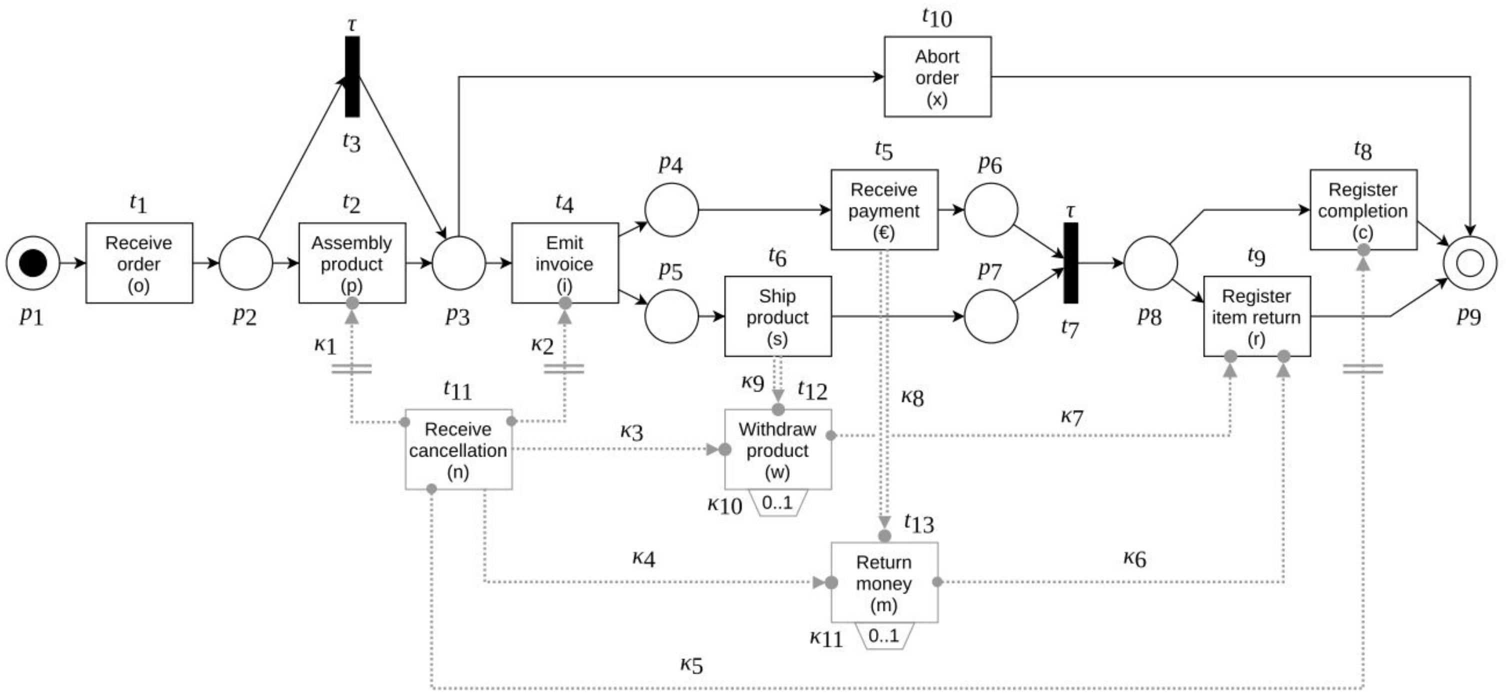
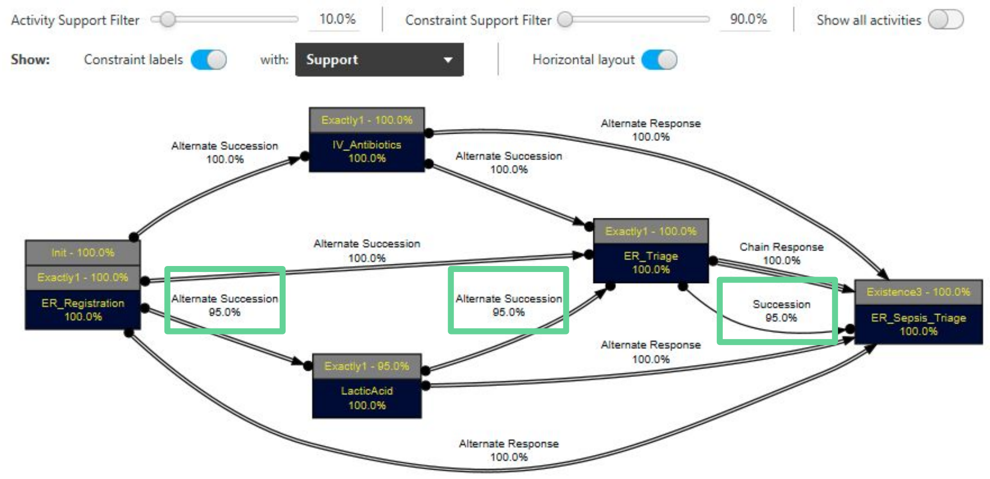


Image from: van Dongen et al. 2021, https://doi.org/10.1016/j.is.2020.101685

Crisp and probabilistic constraints

- Measures of mined constraints can lie beneath 100%...

- ... but if they lie above the threshold, they are treated as if they were undeniable



Activity Support Filter 10.0% | Constraint Support Filter 90.0% | Show all activities

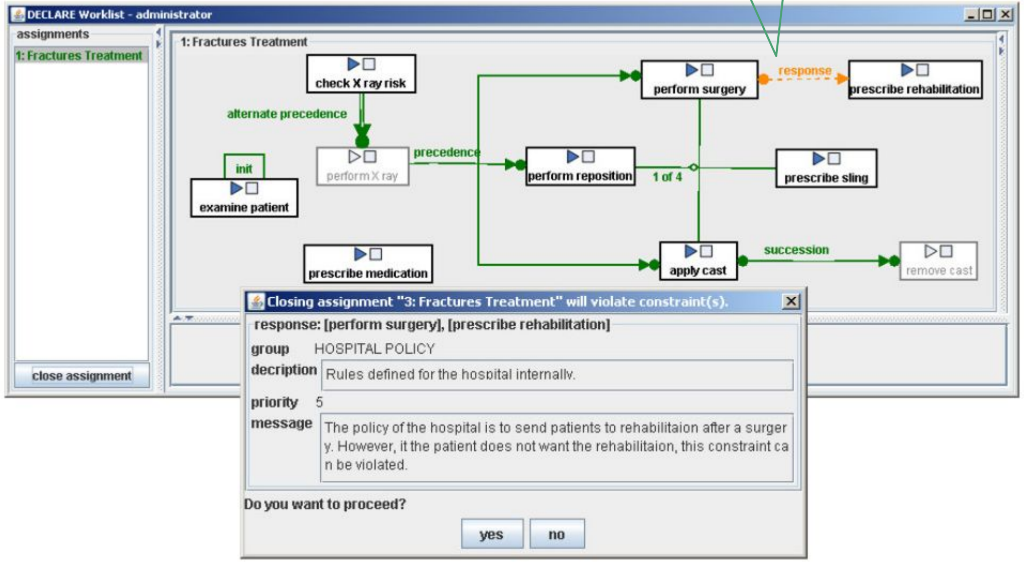
Show: Constraint labels with: Support | Horizontal layout

Crisp and probabilistic and optional constraints

Optional constraint

- Measures of mined constraints can lie beneath 100%...

- ... but if they lie above the threshold, they are treated as if they were undeniable



Closing assignment "3: Fractures Treatment" will violate constraint(s).

response: [perform surgery], [prescribe rehabilitation]

group HOSPITAL POLICY

decription Rules defined for the hospital internally.

priority 5

message The policy of the hospital is to send patients to rehabilitaion after a surgery. However, it the patient does not want the rehabilitaion, this constraint can be violated.

Do you want to proceed?

yes no

Image from: Pesic 2008, https://doi.org/10.6100/IR638413

Object-centric processes

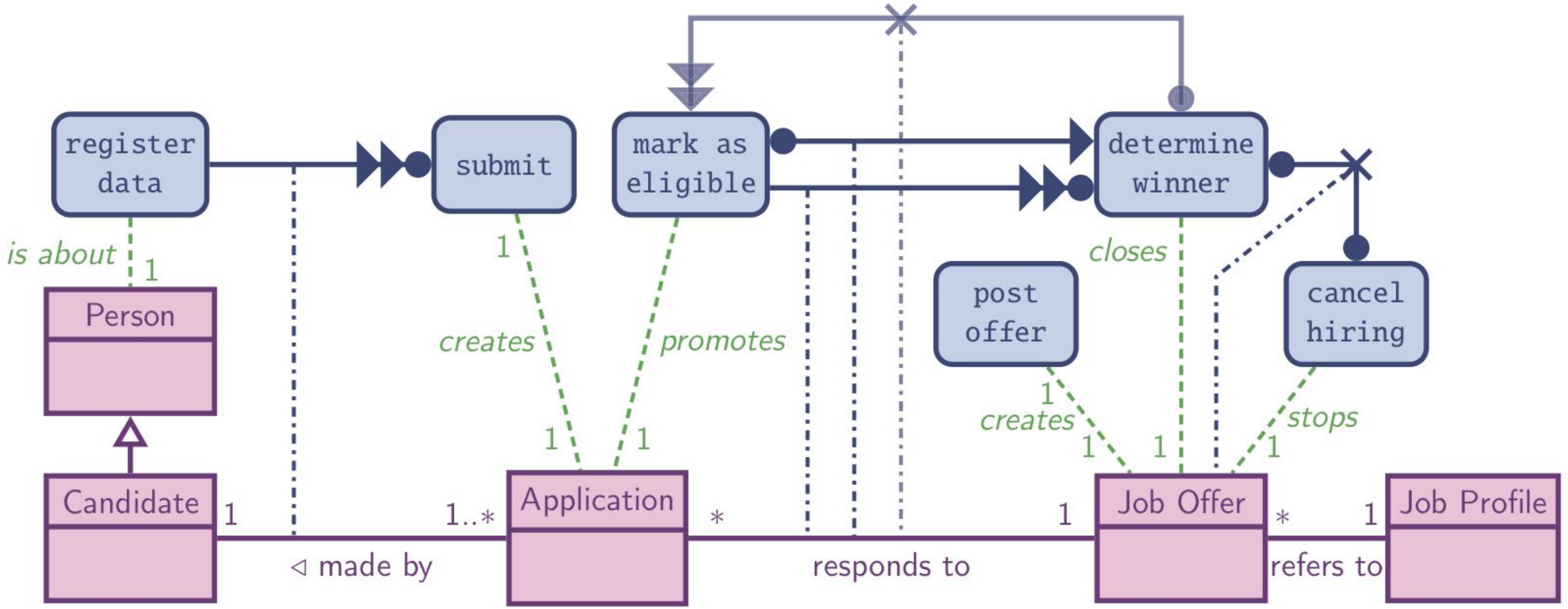


Image from: Artale et al 2019, https://doi.org/10.3233/978-1-61499-955-3-257

Your ideas?

RuM:
Declarative Process Mining, Distilled

Anti Alman, Claudio Di Ciccio, Fabrizio Maria Maggi, Marco Montali, Han van der Aa

https://rulemining.org/