Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

Using BPMN-Q to show violation of execution ordering compliance rules

Business Process Technology Seminar 2008

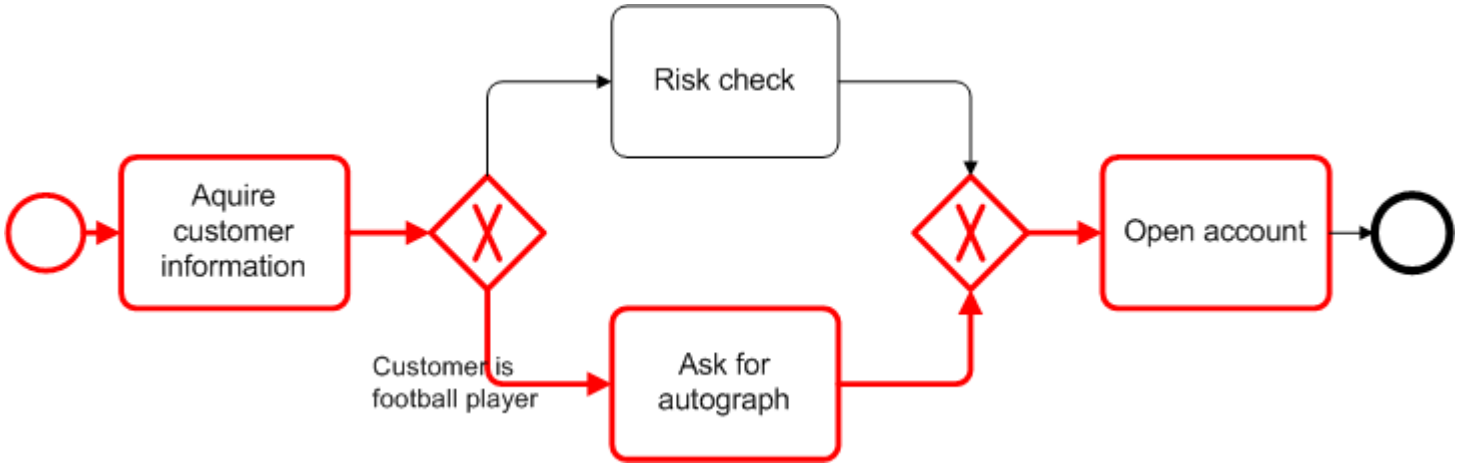
Manuel Blechschmidt

Overview

- The goal
- How to express and use validation rules
  - Ordering compliance checking
  - BPMN-Q
  - Validation rule
  - Annotating semantics
  - Basic rules
  - Deriving anti-patterns
- Formalized algorithm
- Example execution of validation
- Conclusion

The goal

- Visualize violation of a compliance rule in a BPMN process



- Rule: For every „Open account“ a „Risk check“ should be executed

Overview

- The goal
- How to express validation rules
  - Ordering compliance checking
  - BPMN-Q
  - Validation rule
  - Annotating semantics
  - Basic rules
  - Deriving anti-patterns
- Formalized algorithm
- Example execution of validation
- Conclusion

Narrowing validation rules

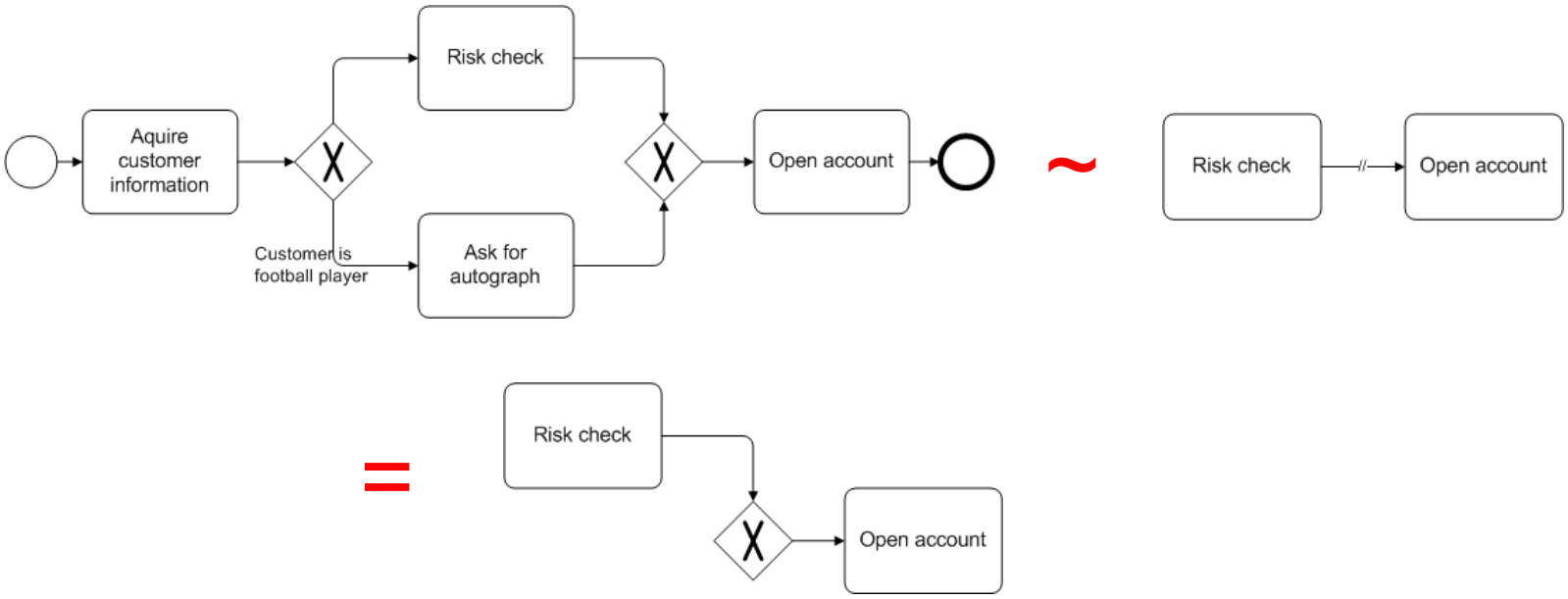
- validation rules can be used to check certain criterias for a business process
- compliance rules are an alias for validation rules which are often used if a repository of process has to comply to a certain law or standard
- ordering compliance is a rule which ensures that certain activities are always executed and appear in a certain order
  - <<precedes>>
  - <<leads to>>

Example execution of validation

- Validation process
  - Identify process
  - Create pattern
  - run pattern against process to ensure that at least one valid execution path exists
  - if no match is found validation failed
  - Derive anti pattern
  - run anti pattern against process to find counter example
  - if one is found visualize violation with match otherwise accept

BPMN-Q

- BPMN-Q is a visual language to extract matches out of a process
  - In a nutshell BPMN-Q is for BPMN what is RegEx for Strings
- Example:



Definition of BPMN-Q query

Definition 1 *A query graph is a tuple* $G = (N, S, NS, P, NP, T, L, stereotype)$ *where*

- N = *finite set of nodes.* $N = CA \cup VA \cup EV \cup GW$ *where*
  - CA = *set of concrete activities.*
  - VA = *set of variable activities.*
  - EV = *set of events.*
  - GW = *set of gateways.*
  - $CA \cap VA \cap EV \cap GW = \emptyset$
- $S \subseteq N \times N$ = *sequence flow edges between nodes.*
- $NS \subseteq N \times N$ = *negative sequence flow edges between nodes.*
- $P \subseteq N \times N \times \mathcal{P}(N)$ = *path edges between nodes and set of excluded nodes.*
- $NP \subseteq N \times N$ = *negative path edges between nodes.*
- $T : N \rightarrow \{CONCRETEACTIVITY, VARIABLEACTIVITY, EVENT, GATEWAY\}$
- $L : N \rightarrow l$ *is a labeling partial function.*
- $stereotype : P \rightarrow \{$" << *leadsto* >> "," << *precedes* >> "$\}$ = *assigns semantics to the path*

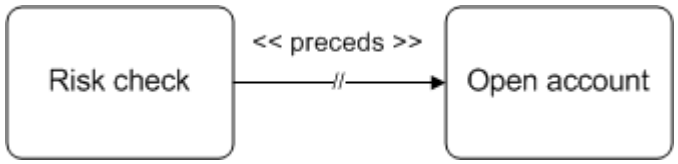
Validation rule

Definition 2 *A validation rule is a pair of an annotated BPMN-Q query called pattern and a set of BPMN-Q queries called anti-pattern whereat the anti-pattern can be automatically derived from the pattern.*

$$
\begin{aligned}
V &= (Q_P, A) \\
A &= \{Q_{A_1}, Q_{A_2} \dots Q_{A_i}\} \\
Q_P &\Rightarrow A
\end{aligned}
$$

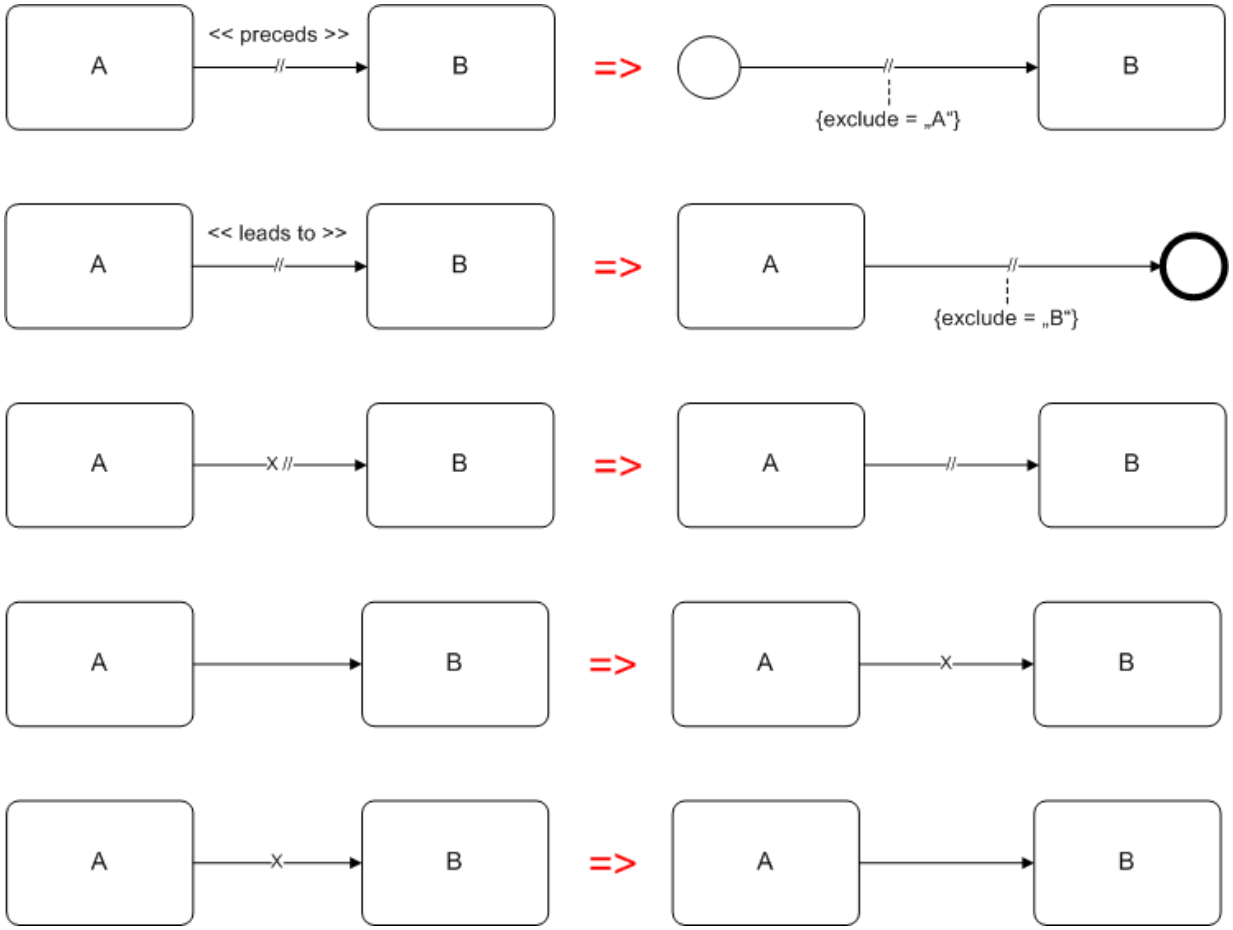
Annotating semantics

- A BPMN-Q pattern can't be translated without annotating semantics. We have to use the stereotypes
  - << leads to >>
  - << preceds >>



Basic rules

- Basic rules that are supported to derive the anti-patterns



Deriving anti-pattern

- Precondition
  - The process must be available
  - The annotated query must be available
- Postcondition
  - Set of anti-pattern is created and validation rule can be used to validate process
- Execution
  - Run algorithms againts process
  - Basic algorithm concept: receive all connections in the query and create the desired anti-pattern

Overview

- The goal
- How to express and use validation rules
  - Ordering compliance checking
  - BPMN-Q
  - Validation rule
  - Annotating semantics
  - Basic rules
  - Deriving anti-patterns
- Formalized algorithm
- Example execution of validation
- Conclusion
  - Power of expressiveness

Formalized algorithm

Show algorithm in paper and explain how it works

Overview

- The goal
- How to express validation rules
  - Ordering compliance checking
  - BPMN-Q
  - Validation rule
  - Annotating semantics
  - Basic rules
  - Deriving anti-patterns
- Formalized algorithm
- Example execution of validation
- Conclusion

Overview

- The goal
- How to express validation rules
  - Ordering compliance checking
  - BPMN-Q
  - Validation rule
  - Annotating semantics
  - Basic rules
  - Deriving anti-patterns
- Formalized algorithm
- Example execution of validation
- Conclusion

Conclusion

- Most important result
  - Automatic derivation of anti-pattern
- Visualization of violation is a huge help

- No mathematical proofs yet
- No quantification possible
  - for example for every car produce 4 times a tire
- Implementation not done yet

Possible next steps

- Implement BPMN-Q for oryx
- Implement validation in oryx
- Enhance basic pattern to support more validation scenarios
- proof algorithms and basic patterns mathematicly

Questions?

References

- Ahmed Awad: BPMN-Q: A Language to Query Business Processes. EMISA 2007: 115-128
- Ahmed Awad, Gero Decker, and Mathias Weske: Efficient Compliance Checking using BPMN-Q and Temporal Logic. 6th International Conference on Business Process Management BPM 2008
- Aditya Ghose and George Koliadis: Auditing Business Process Compliance
- Rik Eshuis: Symbolic Model Checking of UML Activity Diagrams
- Shazia Sadiq, Guido Governatori, Kioumars Naimiri: Modeling Control Objectives for Business Process Compliance
- OMG BPMN 1.1 - OMG Final Adopted Specification, January 2008