RuleML
Realize your Knowledge

October 25-26, 2007—Orlando, Florida

# Specifying Process-Aware Access Control Rules in SBVR

Stijn Goedertier[1], Christophe Mues[2], and Jan Vanthienen[1]

1. Department of Decision Sciences & Information Management, Katholieke Universiteit Leuven, Belgium
   jan.vanthienen@econ.kuleuven.be
2. School of Management, University of Southampton, United Kingdom

*Standard citation:* Goedertier, S., Mues, C., and Vanthienen, J. (2007), *Specifying Process-Aware Access Control Rules in SBVR*, in Paschke, A. and Biletskiy, Y., editors, Advances in Rule Interchange and Applications, Proceedings of The International RuleML Symposium (RuleML 2007), Lecture Notes in Computer Science (Springer), volume 4824, pp. 39-52. (Best Paper Award).



## Who?

**Prof. Dr. Jan Vanthienen**
**Katholieke Universiteit Leuven (BE)**
**Faculty of Business and Economics**
Business Information Systems Group

Research and teaching:

- Business rules, processes and information systems
- Verification & Validation of Knowledge
- Business intelligence, Knowledge discovery & management
- Decision tables

Email: jan.vanthienen@econ.kuleuven.be

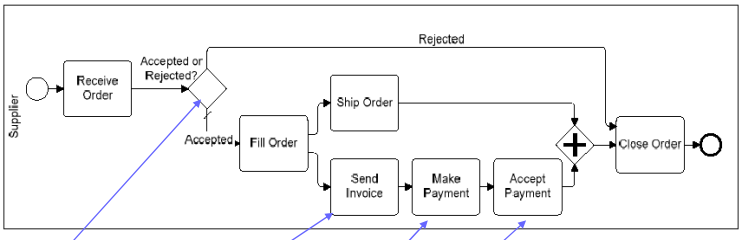
# Overview

- The paper points out that the SBVR is suitable for defining process-aware access control rules, but to date there exists no SBVR vocabulary with process related concepts such as agents, activities and events.
- A new SBVR vocabulary for process modeling is introduced in this paper, so as to specify defeasible access control policies that are able to refer to the state of a business process instance.
- As SBVR does not support defeasible rules, the paper furthermore presents a transformation mechanism to transform defeasible rules into standard, non-defeasible rules.





# Rules and processes




(BPMN 1.0: OMG Final Adopted Specification, 2006)

- Decision rules
- Calculation rules

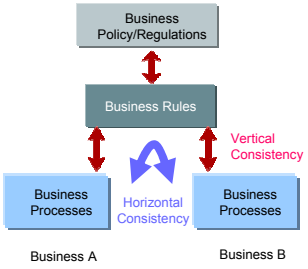
But also:

- Timing rules
- Process rules
- Access rules, …

# Which comes first?

Two architectural styles:

- "The chick": a procedural, process-first style
  - Execution scenarios are explicit, design choices are implicit.
  - Excellent for stable processes, highly standardized
- "The egg": a declarative, rules-first style
  - Rules, choices and goals are explicit, execution scenario is derived.
  - Compliance by design.
  - Excellent for volatile processes, many exceptions, **agility**
- And, of course, combinations of both > Balance





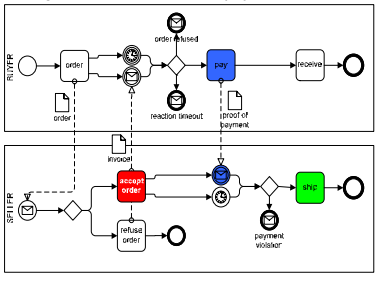



# Permissions and obligations

## accept-payment-ship

*"The buyer must pay the invoice, after the seller accepts the order."*

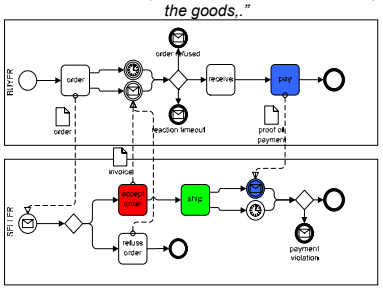
*"When the seller accepts an order, the seller must ship the goods after the customer pays the invoice."*




## accept-ship-payment

*"The buyer must pay the invoice, after the buyer receives the goods."*

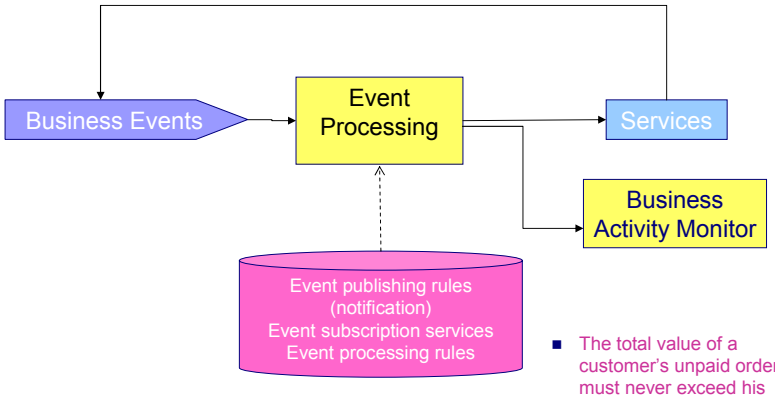
*"When the seller accepts an order, the seller must ship the goods,."*




*Business rules make the permissions and obligations explicit*
*If the permissions and obligations change, the process changes!*

Therefore we need to express permissions and obligations of actors, and a path from the set of rules to compliant process models

# An architecture of business rules, events and services




- The total value of a customer's unpaid orders must never exceed his credit limit
- A claimant must be notified within 5 days once their claim has been denied





# Access Control

**Access control is the ability to permit or deny access to physical or informational resources.**

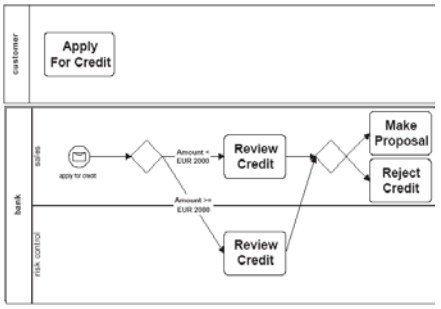
C*redit approval process:* A customer applies for credit and after a credit review, the bank can either make a credit proposal or reject the credit application. Credit approval requires the collaboration between the sales and the risk department. Suppose, for instance, that the following access control policy is formulated:

- **stakeholders**: the customer, the bank, regulators
- **threats**:
  - the bank accepts credit applications with a high probability of default.
  - credit reviews take up too much time and the customer defects to another bank.
- **concerns**:
  - In general, each credit application must be reviewed by the bank's risk department.
  - Credit applications of less or equal than 2000 euros may be reviewed by the sales department. (efficiency)
  - The employee who reviews a credit application can neither be the beneficiary nor the applicant.
  - The same employee should not both review a credit application and make a credit proposal for credit applications larger than 2000 euros.

# Access Control & Processes

To date, many access control specifications are either **process-agnostic** or **process-driven**.

- **Process-driven** access control specifications, hinder both design and run-time flexibility, because they are **embedded** within procedures, applications or process models, e.g. in a BPMN decision gateway.
  ⇒ Duplication, lack of traceability, no guaranteed enterprise-wide access control policy.



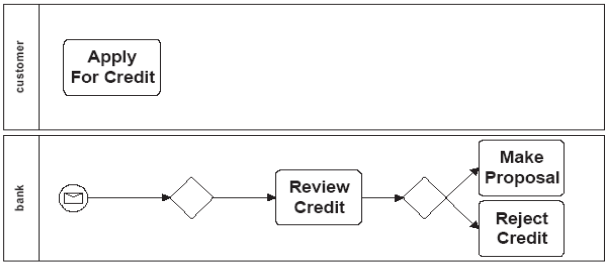

- **Process-agnostic** access control specifications have only **limited expressiveness**, because they cannot relate to the state of business processes to grant or deny access rights. For instance, the policy prevents the same employee from both reviewing and making a proposal. This cannot be expressed without an awareness for an underlying process model.





# Process-aware access control

- Expressive and flexible access control specifications are **process-aware** in that they can refer to an underlying business process context, but do not specify when and how they must be enforced.




*In general, each credit application must be reviewed by the risk department.*
*For reasons of efficiency, credit applications of less than USD 2000 may be reviewed by the sales department.*
*The same employee should not both review a credit application and make a cred proposal.*

# An SBVR Vocabulary for Process Modeling

- The SBVR is a suitable base language for defining process-aware access control rules, but to date there exists no SBVR vocabulary with process related concepts such as *agents, activities and events*. Consequently, it is not possible to declaratively refer to the state of a business process.
- We define an SBVR vocabulary for expressing process-related concepts, called the EM-BrA$^2$CE Vocabulary ('Enterprise Modeling using Business Rules, Agents, Activities, Concepts and Events')'.
- The vocabulary thinks of a business process instance as a trajectory in a *state space* that consists of the possible sub-activities, events and business concepts. Each activity in a process instance can undergo a number of distinct *state transitions*. The occurrence of a state transition is logged as an activity event. Business rules determine whether or not a particular state transition can occur.





# SBVR

- The Semantics of Business Vocabulary and Business Rules (SBVR) is a standard for business modeling that currently is under finalization within the Object Management Group (OMG).
- The standard provides a number of conceptual vocabularies for modeling a business domain in the form of a vocabulary and a set of rules. In SBVR, meaning is kept separate from expression. As a consequence, the same meaning can be expressed in different ways.
- In real-life, meaning is more often expressed in textual form than in diagrams as statements provide more flexibility in defining vocabulary and expressing rules. For these reasons, the SBVR specification defines a structured, English vocabulary for describing vocabularies and verbalizing rules, called SBVR Structured English

# Example state transitions

– create(AId, AT,BId, PId,CoordinatorId): requests the creation of a new activity AId of type AT with business identifiers BId, parent activity PId by an agent CoordinatorId. Activity event type: created.

– assign(AId, AgentId,CoordinatorId): requests the assignment or evocation of the assignment of activity AId to an agent AgentId by an agent CoordinatorId. Activity event type: assigned.

– updateFact(AId,C1,C2,WorkerId): requests the update of a business fact C1 by C2 within the context of activity AId by an agent WorkerId. Activity event type: factUpdated.

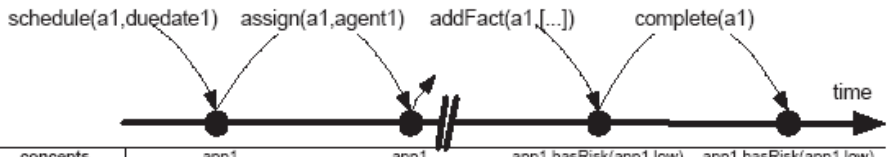
– complete(AId,WorkerId): requests the completion of activity AId by an agent WorkerId. Activity event type: completed.





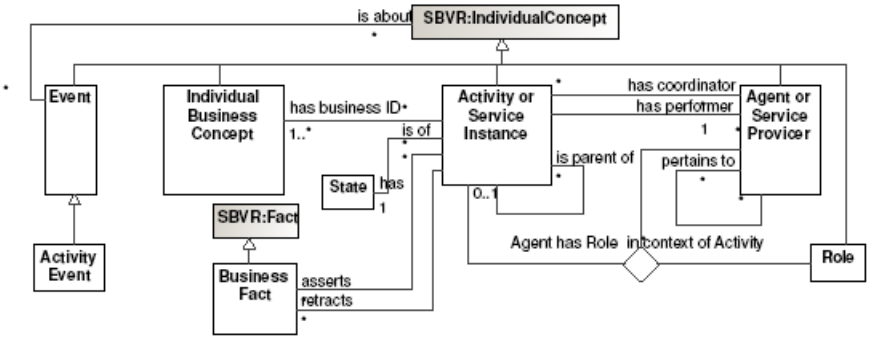
# An illustration of the state transitions for a 'review credit' activity 'a1'

The current state of an activity determines which state transitions can occur. For the purpose of access control, the assign(AId, AgentId, CoordinatorId) is the activity state transition of interest.




| concepts | app1 | app1 | app1,hasRisk(app1,low) | app1,hasRisk(app1,low) |
|---|---|---|---|---|
| agent | | agent1 | agent1 | agent1 |
| activity | a1 | a1 | a1 | a1 |
| event | e1,e2 | e1,e2,e3 | e1,e2,e3,e4,e5 | e1,e2,e3,e4,e5,e6 |
| *has type* | ...,(e2,scheduled) | ...,(e3,assigned) | ...,(e5,factAdded) | ...,(e6,completed) |
| *has performer* | | (a1,agent1) | (a1,agent1) | (a1,agent1) |
| ... | | | | |

# EM-BrA²CE ('Enterprise Modeling using Business Rules, Agents, Activities, Concepts and Events')




A MOF/UML representation of the EM-BrA²CE Vocabulary

Goedertier, S., Haesen, R., and Vanthienen, J. (2007). *EM-BrA²CEv0.1: A vocabulary and execution model for declarative business process modeling*. FETEW Research Report KBI 0728, K.U.Leuven.
http://www.econ.kuleuven.ac.be/public/ndbaf38/EM-BrAACE





# Specifying Access Constraints

The EM-BrA²CE Vocabulary allows to specify access control policies that are able to refer to the state of a business process instance. In particular, each business process can be modeled by describing its state space and the set of business rules that constrain movements in this state space.

For instance, the state space of the credit approval process is described by facts about the following concepts:

- roles: *applicant, sales representative, risk assessor*
- atomic activity types: *apply for credit, review credit, make proposal, reject credit*
- activity event types: *created, assigned, started, completed*
- business concepts: *credit application, amount, collateral, applicant, income, risk*
- business fact types: *credit application has collateral, credit application has amount, applicant has income*

# Access control rules

Access control rules constrain
the *assign(AId, AgentId,CoordinatorId)* state transition.

In the EM-BrA$^2$CE Vocabulary three kinds of access rules can be specified:

- activity authorization constraints
- visibility constraints
- subscription constraints





# Activity authorization constraint

- An activity authorization constraint allows to constrain the agent-role assignments that can be granted to an agent.
- For instance, the fact '*sales representative can perform review credit*' is constrained by the rule that credit applications larger than 2000 euros cannot be reviewed by employees of the sales department.

## Visibility constraint

- A visibility constraint is a structural business rule that dynamically constrains the visibility of business facts to agents based on the properties of the business facts and the behavior of the agents.
- Example: It is not possible that the fact type '*credit application has reason of rejection*' is visible to the agent who has role applicant if the credit application has type consumer credit.





## Event subscription constraint

- An event subscription constraint is a structural business rule that constrains the conditions under which agents who have a particular role in the context of an activity can perceive the occurrence of an activity event.
- Example: It is not possible that an agent that has role risk assessor perceives a started event that is about an apply for credit activity that has subject a credit application that has an amount of less than 2000 euros.

# Specification of process-aware access control 4 steps:

1. Define an access control policy (non-actionable directive) that identifies security threads and safety concerns and **motivates** access control implementation.
2. Identify the access control roles. Roles are permissions involving the performance of activities or the involvement in activities that pertain to meaningful groups of activity types. Roles provide **stability**.
3. Make agent-role assignments. Agent-role assignment is the provisioning of agents with roles that represent access rights.
4. Specify access constraints. Access constraints refine the role-based access control policy to take into account issues that are beyond the scope of user-role assignment. Access constraints give an access control model **precision**, because they constrain the role-based access according to the properties of the agent, the activity and the business process event history.





# Defeasible Access Control Rules

Access control specifications adhering to the role-based access control (RBAC) have a non-monotonic semantics that can be expressed in defeasible logic. Defeasible logic is a means to formulate knowledge in terms of general rules and exceptions. To this end, defeasible logic allows for rules of which the conclusions can be defeated (defeasible rules) by contrary evidence provided by strict rules, other defeasible rules and defeaters.

In EMBrA$^2$CE the 'agent can perform activity' fact type is defined using a rule set of two generic defeasible rules:

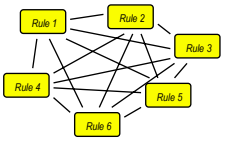
- r1 : true $\Rightarrow$ ¬canPerform(G,A),
- r2 : hasRole(G,R), hasT ype(A, At), canP erform(R, At) $\Rightarrow$ canPerform(G,A)

and a number of domain-specific activity authorization constraints, that translated to the following defeasible rules:

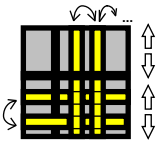
- r3 : A(3) $\Rightarrow$ ¬canPerform(G,A),
- ri : A(i) $\Rightarrow$ ¬canPerform(G,A),
- rn : A(n) $\Rightarrow$ ¬canPerform(G,A).

The following priority relationship applies between the generic and domainspecific defeasible rules:
r1 < r2, r2 < r3, r2 < r4, ...r2 < rn.

# Transforming the defeasible rules




*two-dimensional grouping of interrelated rules*



*per condition subject*

*per action subject*

Refined specification language with general rules, exceptions, etc.

*Actions **[generally] if** condition combinations*
*(**Not**) action **definitely if** condition combinations*
*Action **only possible if** condition combinations*
*Action **definitely if and only if** condition combinations*


www.econ.kuleuven.ac.be/prologa





# Transforming the defeasible rules

| review credit | | | | | | |
|---|---|---|---|---|---|---|
| 1. agent *is applicant of* credit application | Y | N | | | | |
| 2. agent *is beneficiary of* credit application | - | Y | N | | | |
| 3. agent *has role* risk assessor | - | - | Y | N | | |
| 4. agent *has role* sales representative | - | - | - | Y | | N |
| 5. credit application *has* amount larger than 2000 euros | - | - | - | Y | N | - |
| 1. agent can perform the review credit activity | - | - | x | x | - | - |
| 2. agent cannot perform the review credit activity | x | x | - | - | x | x |
| | 1 | 2 | 3 | 4 | 5 | 6 |

$R_1$ In general, conclusion 2.
$R_2$ In general, conclusion 1 if condition 3.
$R_3$ In general, conclusion 1 if condition 4.
$R_4$ In general, conclusion 2 if condition 1.
$R_5$ In general, conclusion 2 if condition 2.
$R_6$ In general, conclusion 2 if condition 4 and condition 5.

$R_1 \prec R_2 \prec R_3 \prec R_4 \prec R_5 \prec R_6$

It is necessary that an agent that *is* not *applicant of* the credit application and that *is* not *beneficiary of* the credit application and that *has* not *role* risk assessor and that *has role* sales representative, can not perform a review credit activity that *has subject* credit application that *has* an amount more than 2000 euros.

# Contributions

- The paper points out that the SBVR is suitable for defining process-aware access control rules but to date there exists no SBVR vocabulary with process related concepts such as agents, activities and events.
- A new SBVR vocabulary for process modeling is introduced in this paper, so as to specify defeasible access control policies that are able to declaratively refer to the state of a business process instance.
- As SBVR does not support defeasible rules, the paper furthermore presents a transformation mechanism to transform defeasible rules into standard, non-defeasible rules.





# References


- Goedertier, S. and Vanthienen, J. (2007). *Declarative Process Modeling with Business Vocabulary and Business Rules*. In Halpin, T., Nijssen, S., and Meersman, R., editors, Proceedings of Object-Role Modeling (ORM'07), volume (forthcoming) of Lecture Notes in Computer Science. Springer.
- Vanthienen, J. (2007*). How Business Rules (Re)define Business Processes: A Service Oriented View, 10*th International Business Rules Forum, Orlando, FL (USA), Oct. 21-25.
- Goedertier, S., Haesen, R., and Vanthienen, J. (2007). *EM-BrA²CEv0.1: A vocabulary and execution model for declarative business process modeling*. FETEW Research Report KBI 0728, K.U.Leuven. http://www.econ.kuleuven.ac.be/public/ndbaf38/EM-BrAACE
- Vanthienen, J. (2006). *Consistency by construction: Decision table experiences in business rules and processes,* 9th International Business Rules Forum, Washington, DC (USA), Nov. 5-9.
- Goedertier, S. and Vanthienen, J. (2006). *Designing compliant business processes with obligations and permissions.* In Eder and Dustdar, editors, Proceedings of BPM 2006 International Workshops, volume 4103 of Lecture Notes in Computer Science. Springer. pages 5–14.
- Vanthienen, J. (2006). *50 Ways to represent your rule sets*, Business Rules Journal, vol. 7, no. 1 (Jan.), pages 1-7.