UNIVERSITÄT OSNABRÜCK

Enhancing Business Process Management with a Constraint-Based Approach

Wolfgang Runte

Software Engineering Research Group
Institute of Computer Science
University of Osnabrueck
P.O. Box 4469, 49069 Osnabrueck, Germany

SOMET 2012

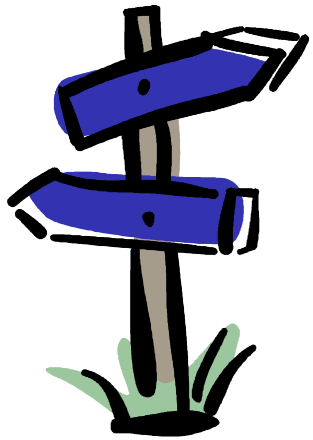
The 11th International Conference on
Intelligent Software Methodologies,
Tools and Techniques

September 26-28, 2012, Genoa, Italy

woru@informatik.uni-osnabrueck.de
http://www.inf.uos.de/woru/

Outline of the Talk

1. Motivation
2. Rule-Based Systems
3. Constraint-Based Systems
4. Examples
5. Conclusion & Outlook



Motivation

- **Business Process Management (BPM)**
  - compliances and relations between processes have to be considered by process engines

- **State of the Art**
  - for the definition of relations and for compliance management mainly the rule-based approach (namely *business rules*) is applied
  - the control flow in BPM systems is supported by *rule engines*
  - decisions are taken on basis of dependencies and relations between entities defined in business rules
  - there exist a number of well known drawbacks in the usage of rules in software systems
  - already in the 1980s the excessive usage of rules showed evident insufficiencies

Motivation

UNIVERSITÄT OSNABRÜCK

- **Drawbacks of Rule-Based Systems**
  - in the 1980s rule-based systems became very popular in the domain of *expert systems* (later on called *knowledge-based systems*)
  - it soon became apparent, that large rule-based systems causes enormous maintenance problems, because of the lack of separation between domain knowledge and control strategy; rules specify both:
    - *directed relationships* (domain knowledge), as well as
    - *actions* (procedural knowledge to control the execution task)
  - also the sharing of knowledge about a single entity over several rules makes the knowledge maintenance an extremely difficult task; example:
    - in 1989 the rule-based configuration system R1/XCON had in its rule base more than 31,000 object descriptions and approximately 17,500 rules
    - 40-50 percent of this rules had to be modified or regenerated every year
    - the maintenance of the knowledge base was carried out by up to 40 employees
  - modularization and the subsumption to *meta rules* (rule-based programming methodology called RIME) did not solve the problems

Motivation

- **Example: Difference between Rules and Constraints**
  - imagine the process model in the scenario of the organization and realization of a holiday tent camping of a children's group; the following is an example of an important rule during this camping:

    ```
    If there is a storm warning,
    then strike the tents and evacuate the camp.
    ```

  - this rule contains some control knowledge: which activities have to be executed next, if the condition is fulfilled
  - in the same scenario the appropriate constraint would be simple like this:

    ```
    no storm warning
    ```

    or in a slightly more formally way:

    ```
    stormWarning == false
    ```

  - which means that there *must not* be a storm warning, otherwise the constraint is not satisfied (constraint violation, inconsistent state)
  - what to do, if the constraint does not hold, has to be modeled as a process model executed by the process engine (*not* by a rule engine)

Rule-Based Systems

UNIVERSITÄT OSNABRÜCK

- **Principles of the Rule-Based Approach**
  - rules-based systems are a special form of knowledge based systems
  - rules are called *production rules*, *productions* or *condition-action rules* and have typically the following form:

    ```
    IF <condition> THEN DO <actions>
    ```

  - besides the definition of causal relations between objects the strongness of this approach is to describe and evaluate heuristic dependencies
  - each rule is an independent knowledge unit and will be interpreted and executed by a domain independent rule engine
  - *data-driven* rule engines match the conditions of rules with respect to existing data and identify rules to be executed next (*forward chaining*)
  - *demand-driven* rule engines are focused on the action parts of the rules (an action part has to contain a previously defined goal, *backward chaining*)
  - variations of the well known *Rete algorithm* (Forgy 1982) are still the core of most current rule-based systems

Rule-Based Systems

UNIVERSITÄT OSNABRÜCK

- **Discussion on Rule-Based Systems**
  - current rule-based systems come with enhanced capabilities for analysis and debugging improving the development and maintainability
  - but the essential weakness of rules still remains due to characteristics of the rule-based approach
  - the modeling of the process engineer and the size of the rule base are essential for the efficiency of the approach
  - used as the exclusive inference mechanism, rules in current rule-based systems will become as critical as in the past
  - it was one lesson from the 1980s to use different inference mechanisms in knowledge-based systems: different kinds of knowledge need different approaches to represent it adequately
  - for the appropriate modeling of existing dependencies in BPM an additional inference mechanism is reasonable which does not compete against the process engine for the control strategy

Constraint-Based Systems

UNIVERSITÄT OSNABRÜCK

- **Principles of the Constraint-Based Approach**
  - in the area of artificial intelligence (AI) constraints have been in the focus of intensive research for decades (Tsang 1993, Dechter 2003)
  - there exist efficient algorithms and heuristics for the reduction of problem size and for an efficient generation of solutions
  - constraint techniques can be used to guarantee that specific relations hold, so the principle is a declarative paradigm
  - each constraint defines a relation between a subset of variables and constrains the values for the involved variables
  - *constraint propagation*: if for any reason the domain of a variable is reduced to a smaller number of values, this domain modification can be propagated through the constraint net using the available constraints to determine smaller value domains for the rest of the involved variables
  - in general for the processing of constraints the problem is formulated as a *constraint satisfaction problem*

Constraint-Based Systems

A **Constraint Satisfaction Problem** (CSP) is a triple *CSP(V,D,C):*

$V = \{v_1, \ldots, v_n\}$ *a finite set of* ***variables***

$D = \{D_1, \ldots, D_n\}$ *associated value* ***domains*** $\{v_1 : D_1, \ldots, v_n : D_n\}$

C a finite set of ***constraints*** $c_i(V_i)$, $i \in \{1, \ldots, m\}$, *with*

$c_i(V_i)$ *to set the subset* $V_i = \{v_{i_1}, \ldots, v_{i_k}\} \subseteq V$ *in relation,*

solution space for $c_i(V_i)$: $\{D_{i_1} \times \ldots \times D_{i_k}\}$

Example:

- Variables: *a* and *b* each with the value domain {0,1,2,3,4,5,6,7,8,9}
- Constraints: $a + b = 10$ and $a - b = 2$
- Solution: $a = 6$ and $b = 4$
- Note: Besides arithmetic domains also symbolic domains are feasible.

Constraint-Based Systems

UNIVERSITÄT OSNABRÜCK

- **Black Box Principle**
  - algorithms for constraint satisfaction are usually combined in a software component named *constraint solver*
  - constraint solvers are used to adopt the *black box* principle which makes integration easier and also allows the substitution of a solver with another solver
  - additionally the black box principle guarantees the separation of domain knowledge and control knowledge:
    - the control strategy is exclusively managed by the application (e.g. a process engine) using the solver component
    - the application gives relevant data (domain knowledge) as input to the constraint solver
    - the result of a solving process is output data that is updated values for the given input data
    - the application knows nothing about the internal solving process of the constraint solver and vice versa

Constraint-Based Systems

UNIVERSITÄT OSNABRÜCK

- **Constraints in Business Process Models (1)**
  - currently we see the following fields of application for constraints in BPM:
    1. **constraints as a replacement for business rules:** to bring the control as soon as possible back to the process engine (the user is able to model the control flow intuitively and graphically as process model instead of abstract business rule definitions)
    2. **classical application of constraints:** dependencies and relations of elements and attributes of a business process may be modeled as constraints to inference values due to specific (user) input and to guarantee a consistent process configuration
    3. **constraints as an instrument of *quality assurance* (QA):** known dependencies can be *additionally* made explicit through respective constraints in the process model to support further modeling and ensuring the consistency of the process model (in further revisions of the process model the user will get feedback in case of an inconsistent modeling or modeling errors)

Constraint-Based Systems

UNIVERSITÄT OSNABRÜCK

- **Constraints in Business Process Models (2)**
  - for the above cases we have generally to situations for the application of constraint technology in business processes:
    1. **static use of constraints at modeling:** constraints are used to check for a consistent process model
       - modeling support in a process editor
       - the user has to define known static dependencies in the model as constraints
       - a constraint solver checks efficiently for inconsistencies during the modeling
       - if a constraint does not hold because of an inconsistent modeling the user will get a notification
    2. **dynamic use of constraints at runtime (simulation or live):** constraints are used to check for consistent states of process instances at runtime
       - the constraints have to be applied in a simulation or in a real life system
       - dependencies defined for dynamic use can only be checked at runtime because they require elements or parameters which will not be known before concrete process instances are available
       - the constraint solver is checking for inconsistencies during the execution of the business processes
       - inconsistencies will result in predefined actions

Constraint-Based Systems

UNIVERSITÄT OSNABRÜCK

- **Constraints in Business Process Models (3)**
  - the process engine has to be enhanced by a component that monitors the constraints and reacts in case of constraint violations
  - we propose a *constraint handler* to perform these tasks: it decides what to do, if a specific constraint does not hold
  - the following alternative actions may be taken:
    - a predefined process is executed (handling the constraint violation) parallel to the origin process, who triggered the predefined process and which execution is ***continued***
    - the origin process is ***paused*** until a predefined process is executed and finished
    - the origin process is ***canceled*** and (optionally) a predefined process is executed
    - the constraint violation is ***ignored*** and the execution of the origin process ***continued***, for information only (optionally) a feedback may be generated and given to the user (model or non-modal)
  - additional actions for user feedback may optionally be triggered

Constraint-Based Systems

UNIVERSITÄT OSNABRÜCK

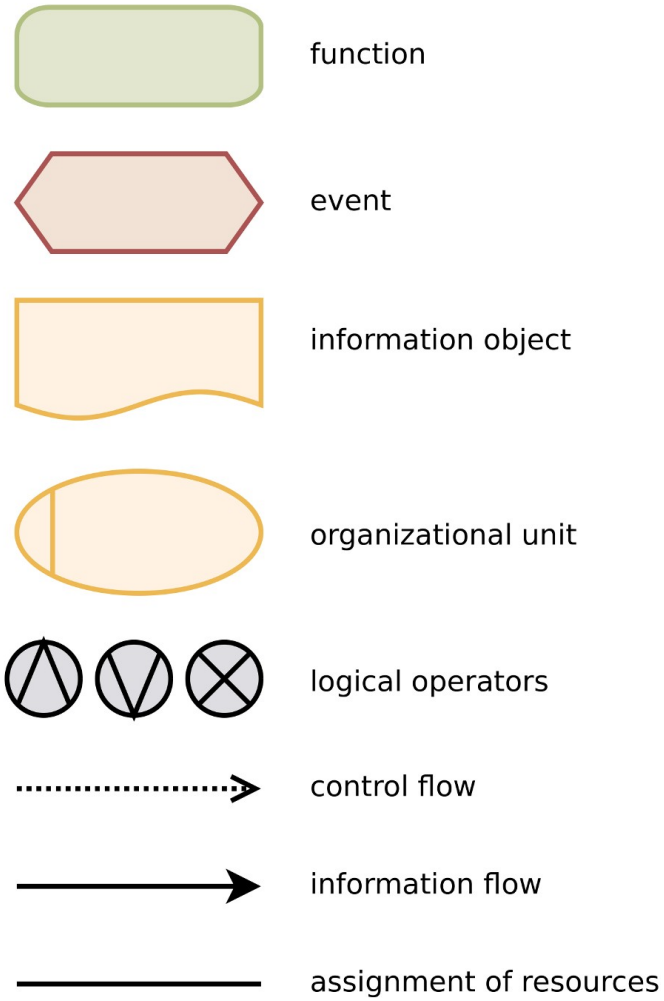
- **Constraints in Business Process Models (4)**
  - to define relations on business processes a *domain-specific language* (DSL) for constraint relations has to be specified
  - this constraint language has to allow the process designer to reference elements of process models and to specify the constraints themselves
  - it is necessary to reference processes, attributes and input/output parameters of processes as well as defining operators on these elements:
    - arithmetic operators like `=`, `!=`, `<`, `>`, `<=` , `>=`
    - temporal operators as introduced by Allen (1983) like `before`, `overlaps`, `during`, etc.
  - as graphical visualization the notation of *event driven process chains* (EPC) is used: a subset of EPC notation elements relevant for the examples in this work is shown on the next slide

Constraint-Based Systems

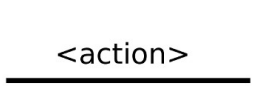
UNIVERSITÄT OSNABRÜCK

Legend of EPC notation elements and constraint related extensions

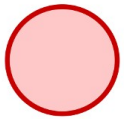
EPC notation elements:



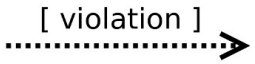
constraint related extensions:



assignment between constraint and origin function/process defining the action to be taken, if the constraint is violated; possible values:
<action> ::= "continue" | "pause" | "cancel" | "ignore"



associated constraint; the constraint relation may be specified optionally with a specific labeling referencing the name of the constraint



control flow between constraint and process/function to be executed, if the constraint is violated (labeling is optional)

Examples

UNIVERSITÄT OSNABRÜCK

1. value constraints

- these constraints relate to the *instance level* of a process model
- they constrain the values of the attributes of model elements which are provided as (input or output) parameters of a process instance

2. model constraints

- are related to the *conceptual level* in process modeling
- they constrain the model elements themselves and therefore the structure of a process model, according to this it is necessary to reference characteristics and properties of model elements:
  - number and type of (input/output) parameters
  - number of processes (e.g. sub-processes or a selection between specific elements) and type of processes
  - number and type of used documents/data resources
  - number and type of involved organizational units

Examples

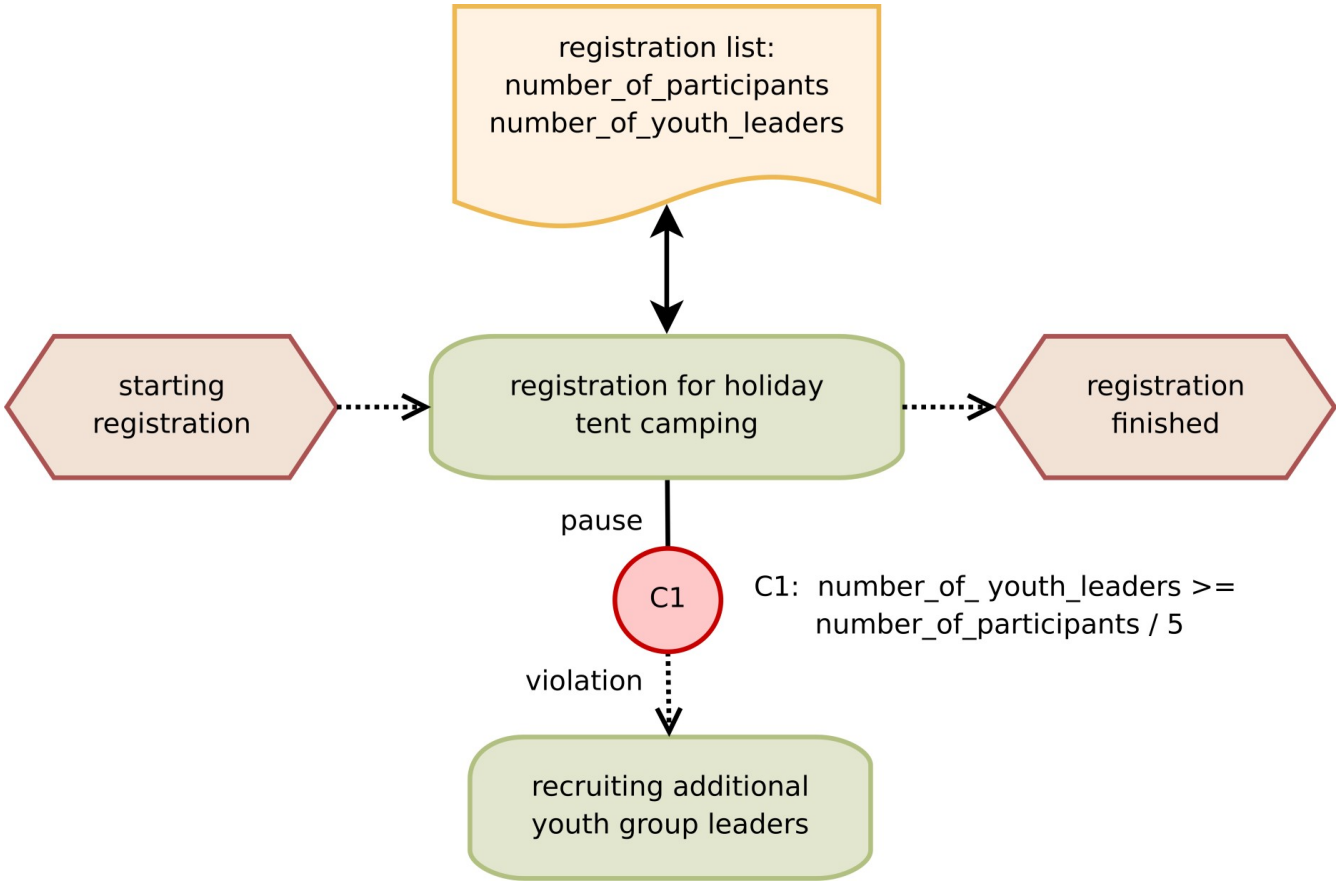
UNIVERSITÄT OSNABRÜCK

3. temporal constraints

- are a specific form of value constraints related to temporal aspects at the instance level of a process model
- with temporal constraints techniques known from temporal reasoning using constraint satisfaction can be applied to business processes
- the *temporal constraint satisfaction problem* (TCSP, Allen 1983) is used for planning and scheduling and it uses its own temporal logic to represent temporal relations
- in our approach temporal constraints are used for flow control instead for scheduling

Examples

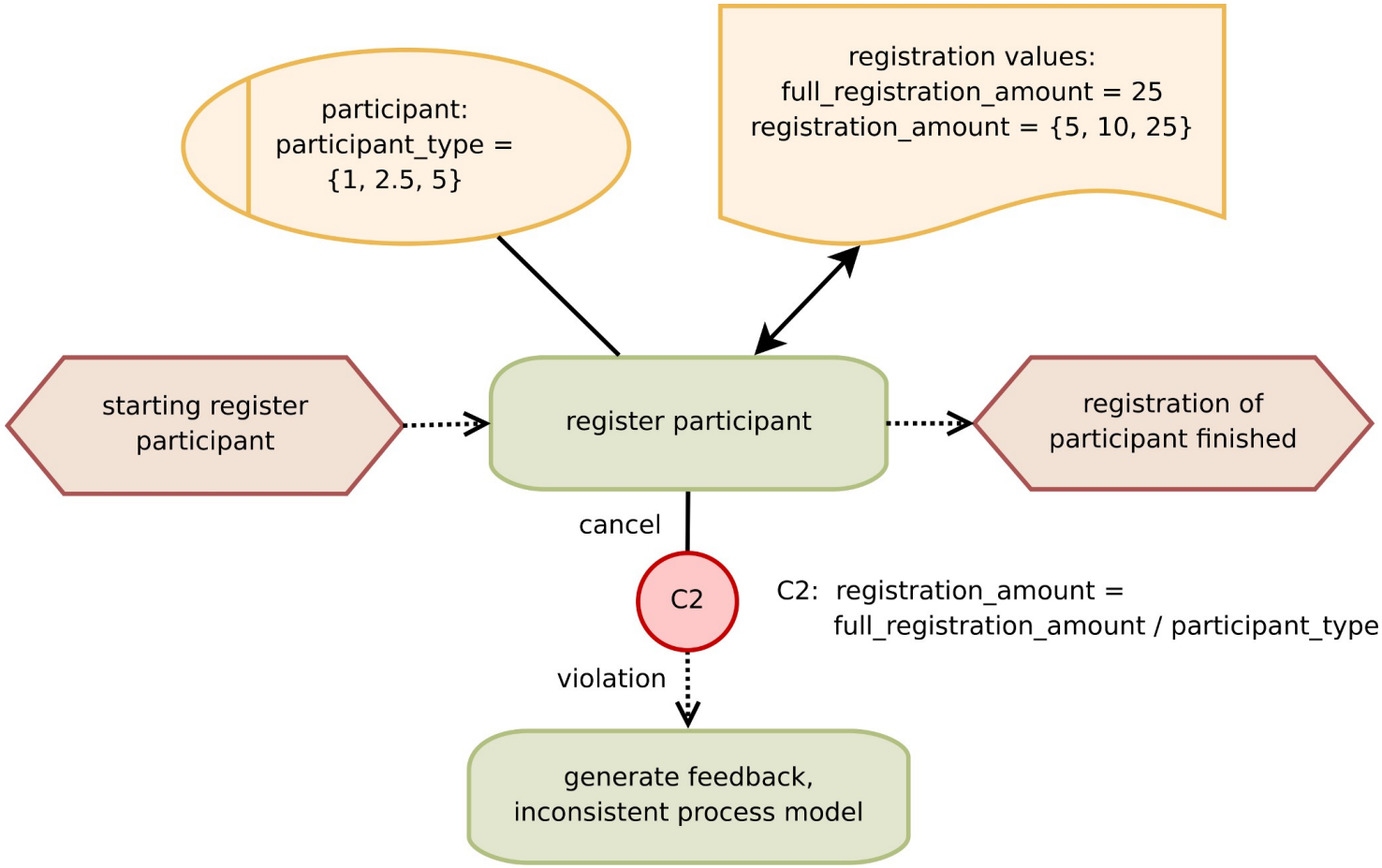
Example 1:



The number of youth group leaders is constrained in relation to the number of participants by a *value constraint*.

Examples

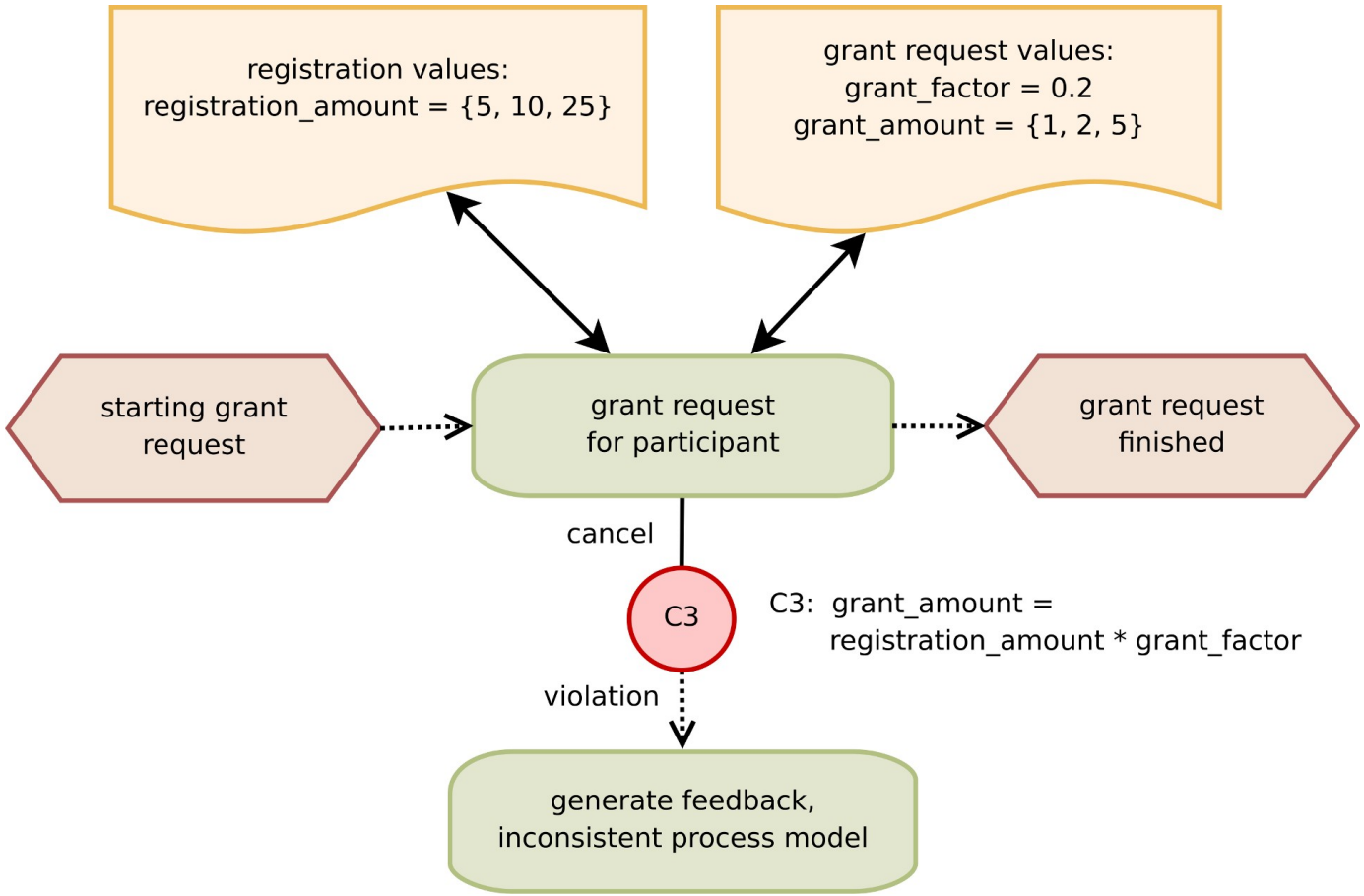
Example 2:



Value constraint **determining the registration amount for different classes of participants.**

Examples

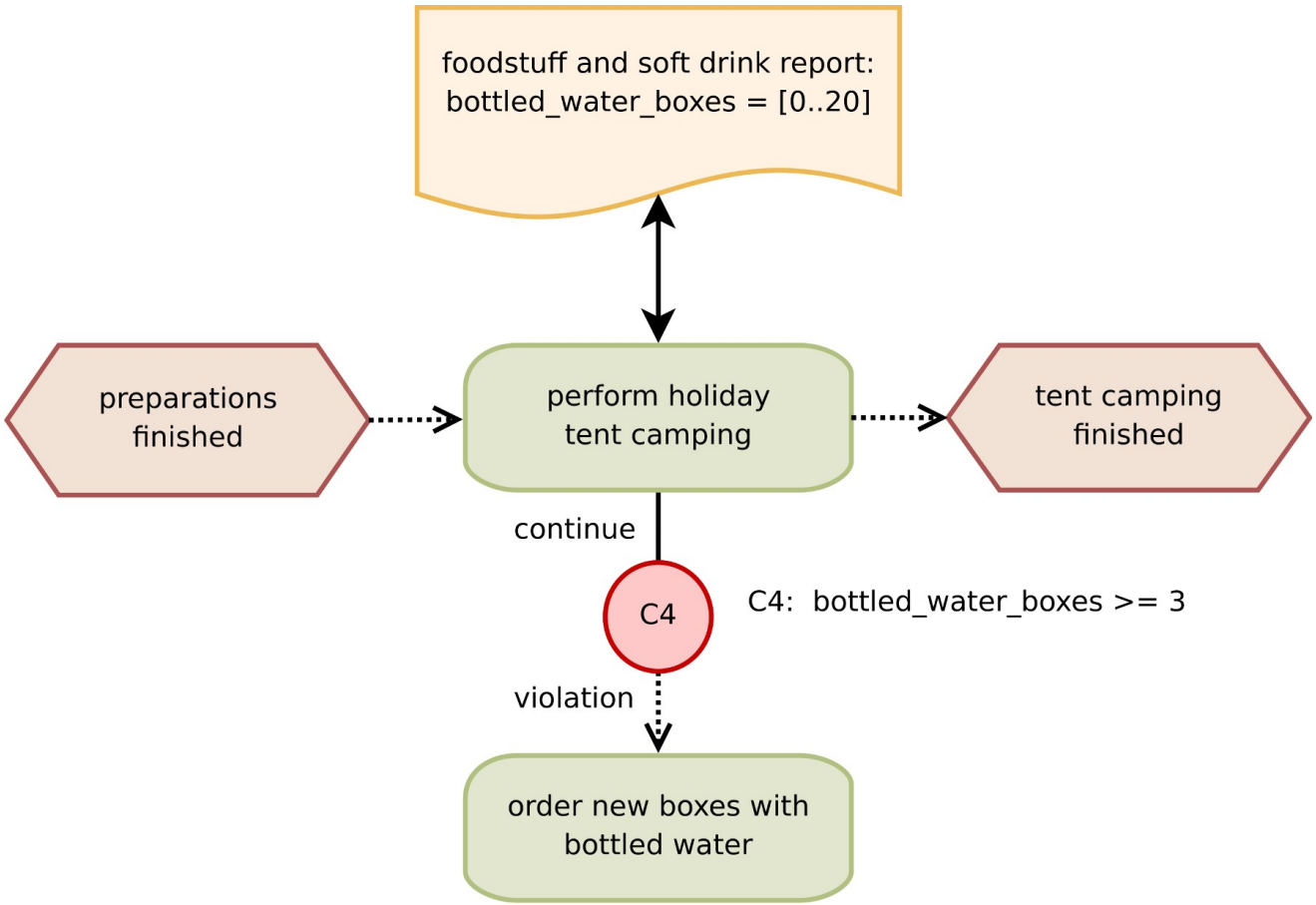
Example 3:



Using constraint propagation of *value constraints* to determine the grant amount per participant (in combination with constraint C2).

Examples

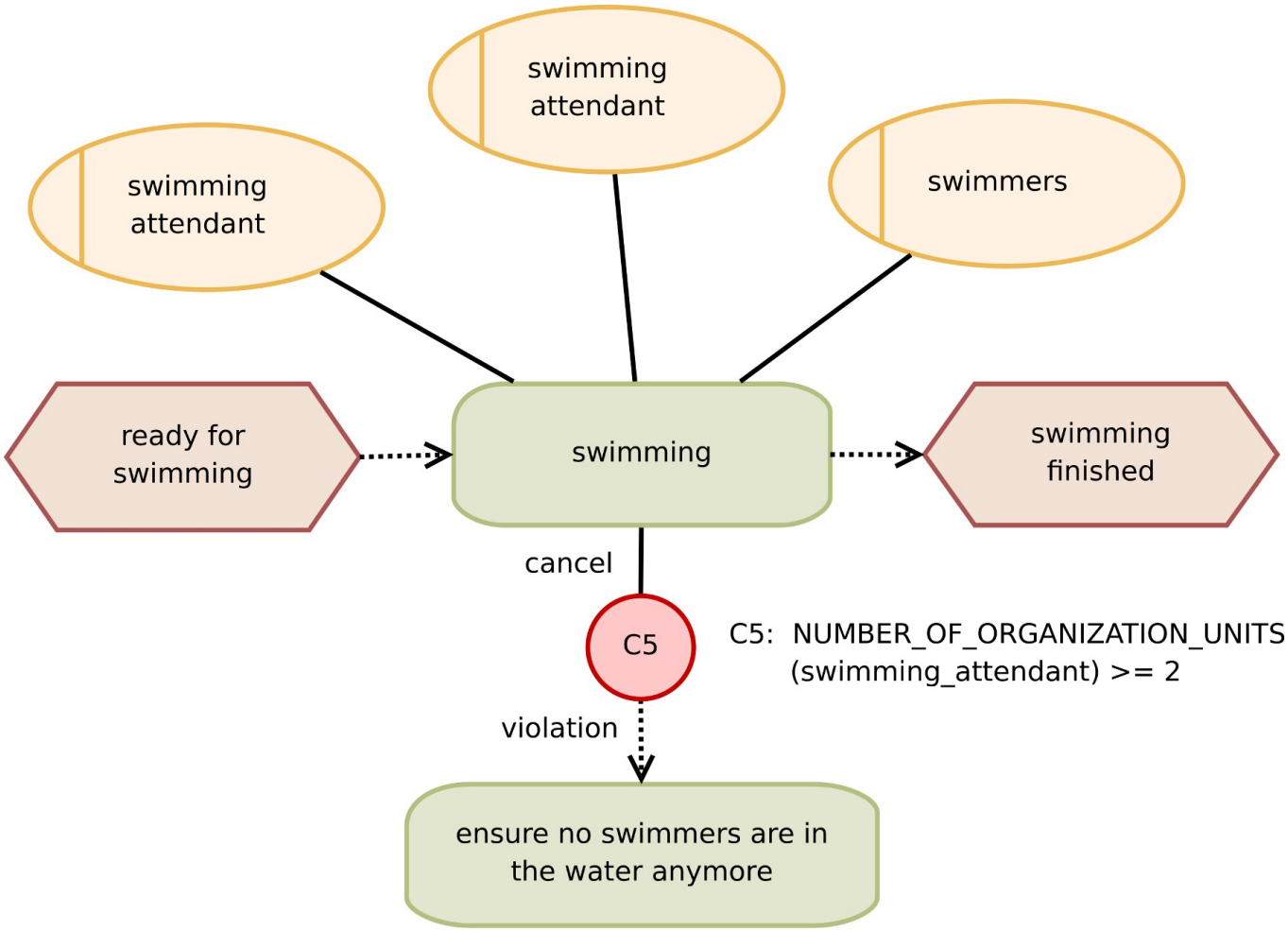
Example 4:



Example for a resource related *value constraint*, triggering an ordering action.

Examples

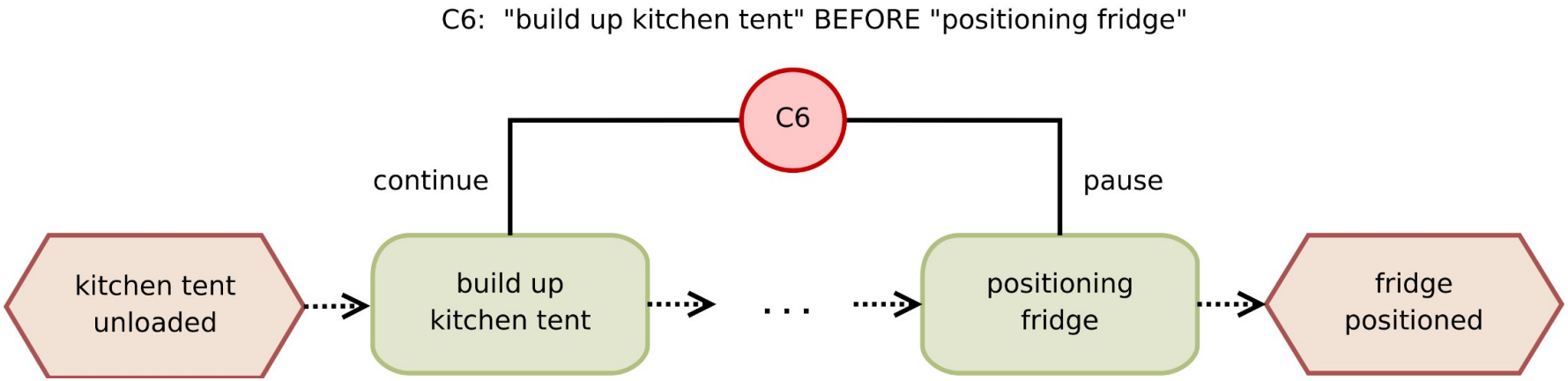
Example 5:



Model constraint **ensuring a minimum number of swimming attendants taking care for the swimmers.**

Examples

Example 6:

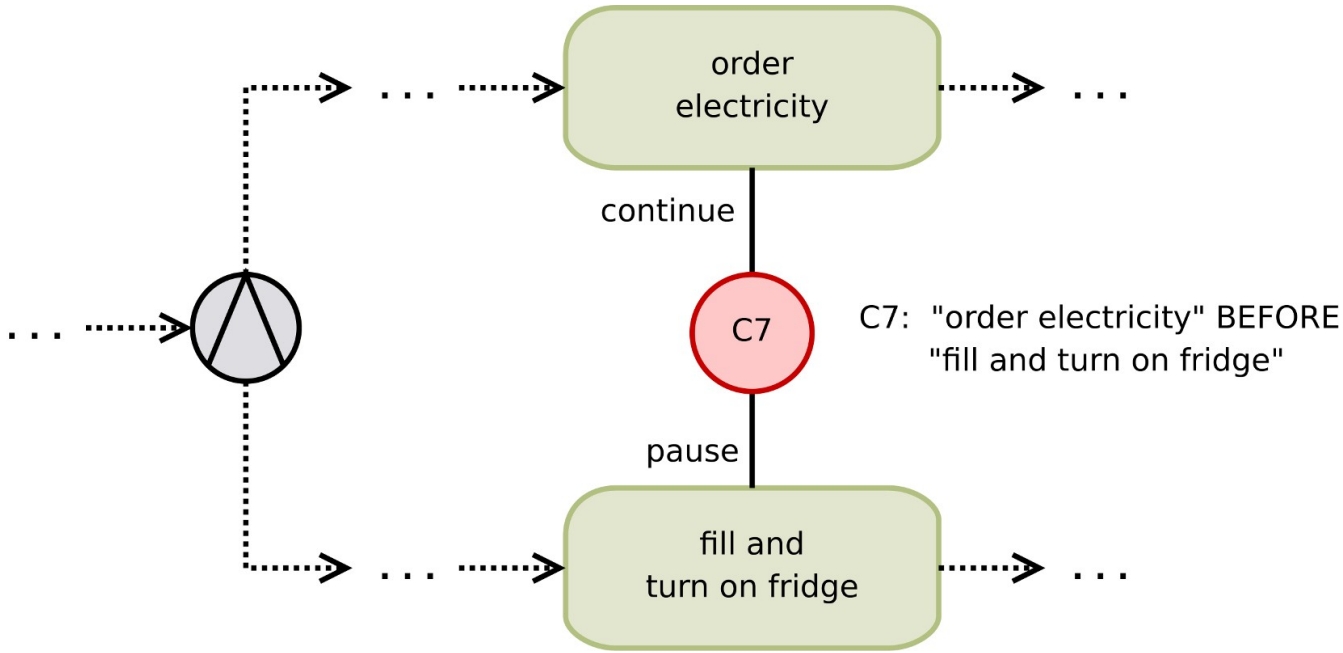


Temporal constraint **determining the sequential ordering of processes.**

Examples

UNIVERSITÄT OSNABRÜCK

Example 7:



Constraining the control flow in a concurrent situation with a *temporal constraint*.

Conclusion & Outlook

- **in business processes often rules are used for decision support**
  - even business processes defined and controlled exclusively by business rule engines are common
- **instead of using rules we propose a constraint-based approach**
  - constraints also have a declarative paradigm but avoid the lack of separation between domain knowledge and control strategy
- **the challenges of this approach lies in the useful enhancement of BPM with constraint technology**
  - further research has to be done with respect to process hierarchies, the scope of constraints, the integration of solutions of sub problems, and meta constraint solving
- **the next steps will be the preparation of case studies and examples for the application of constraint technology in BPM practice**
  - the intention is to generate more input for requirements relevant in practice

UNIVERSITÄT OSNABRÜCK

Thank you for your attention!

Motivation

- **Drawbacks of Rule-Based Systems (2)**
  - consider the problem of updating rules with respect to changes in specific entities:
    - it is very hard and very costly to be certain that all rules were found that need to be updated
    - the rule engine will not execute the action part if the condition of a rule is not up to date
    - the developer of the rule base has to ensure that all required conditions are covered by the set of rules and all desired actions can be reached
    - otherwise one cannot be sure a specific rule is considered at all
    - so every maintenance task has to be performed very carefully and accurately what will result in high costs for each update
    - also small changes will generate much costs

Motivation

- **Drawbacks of Rule-Based Systems (3)**
  - conclusions:
    - rules are declarative, weakly structured, and difficult to manage and maintain
    - for large rule bases the user can not be sure if the problem is completely covered by the rules
    - modifications often result in unwanted consequences
    - the rule-based approach is not well suited for the global control of processes
    - rules should be applied only in local contexts and with limited use
    - rules are well suited to represent local, causal relationships (implications)
    - global control should be performed exclusively by the process engine on basis of the process model
    - techniques using rules have structural drawbacks that limit their application significantly

Motivation

- **Example: Difference between Rules and Constraints (1)**
  - instead of using rules a process engine may be combined with other techniques
  - dependencies and relations can be modeled with *constraints* to eliminate the risk of mixing domain knowledge and control strategy
  - a *constraint solver* will support the process engine using *constraint satisfaction* techniques to ensure the consistency of the modeled dependencies
  - here the *black box principle* guarantees that the constraint solver is not used to take control over the process engine
  - the process engine should get back the flow control as soon as possible

Rule-Based Systems

UNIVERSITÄT OSNABRÜCK

- **Current Rule-Based Systems (selected examples)**
  - IBM WebSphere ILOG JRules
    - Java-based enterprise Business Rule Management System (BRMS)
    - commercial; part of IBM's application and integration middleware “WebSphere”
  - BOSCH Software Innovations Visual Rules
    - BRMS, generates automatically Java code containing the business logic
    - commercial, focused on the graphical and intuitive modeling of the rule tree
  - Red Hat JBoss Enterprise BRMS
    - Java-based BRMS, includes open source *JBoss Drools Rule Engine*
    - commercially distributed and supported by Red Hat
  - Jess
    - expert system shell: rule engine and scripting environment for the Java platform
    - free for educational and government use

Discussion

- **Constraints and Processes: “Graphical” Rules?**
  - someone may guess that constraints seems to be the condition part and the process to be executed in case of a constraint violation seems to be the action part of a rule
  - the difference is that rules would capture the control flow from the process engine, while in our approach in contrast the process engine resumes the control flow if a constraint is violated
  - if constraints in combination with processes are used like business rules with a significant effect on the control flow, they should be used as careful as business rules (limited application with sense of proportion and only in local contexts)
  - furthermore the usage of constraints as a replacement or “simulation” of business rules is only one aspect of the application of constraints in combination with business processes
  - in addition the classical inferences and the field of QA are grateful fields for the application of constraints in the area of BPM

Discussion

- **Constraints versus Rules**
  - the usage of rules differs generally from the usage of constraints: while rules contain a condition part and an action part (which is executed when the condition part is fulfilled) a constraint is assumed to always hold
  - in that it can be observed that a rule has a direction while a constraint is non-directional: the attributes of elements related in the condition part of a rule have a direct effect on the model elements related in the action part, but *not* automatically vice versa
  - in contrast constraint solving algorithms check if the relation defined by the constraint still holds or not: a binary constraint (contains two variables) is checked for both variables, so we have a bidirectional evaluation and also bidirectional effects on the model elements (same for n-ary constraints)
  - rules are usually executed only one time; constraints have to be checked permanently for violations
  - constraints are really a declarative formalism because they are totally free of control structures: the ordering for checking constraints is not relevant for the result (black box)

Discussion

- **Applying Constraints in Which Contexts?**
  - in general the application of constraints is reasonable if:
    - multiple elements are in relation to each other,
    - there exist alternative values in the value domains of these elements, and
    - these alternative values may be reduced by propagation so we can get a domain reduction and thus a problem reduction as result
  - constraints should rather not be used if the same aspect can be modeled as a simple decision using the respective control flow operators in a process model
  - constrains however allow compact modelings of decision processes to determine specific values in process models
  - constraints are well suited to model complex decisions and relations over multiple processes; rules in contrast should be applied limited and only in local contexts, they are suited to represent heuristic knowledge which is executed only one time

Discussion

- **Complexity**
  - the above differences between rules and constraints result in a higher complexity of the constraint technology, because of this constraint-based systems are often criticized to be inefficient managing large constraint nets
  - despite there exist a couple of efficient (heuristic) algorithms handling different kind of constraint problems the fact is less relevant in this case:
    - in the BPM approach the focus is on the control flow in process models managed by a process engine
    - constraints are used additional to BPM techniques, so there is less the problem of complexity of constraints as in the useful enhancement of BPM with constraints
    - in contrast to constraint-based systems, where constraints are the sole knowledge representation technique, supporting BPM we have only simple constraints
    - also the complexity of the constraint net is about a manageable size, because the control flow is mostly determined by the process model
    - constraint technology is used to provide a flexible way to react on violations of known limitations and restrictions and thus supporting the process engine

Discussion

UNIVERSITÄT OSNABRÜCK

- **Declarativity**
  - constraints are a declarative form of knowledge representation, which is often not intuitive and transparent to users thinking in procedural categories
  - in our approach constraint technology is just enhancing process models in BPM: different from from pure constraint-based systems the control flow is defined procedural by the process models
  - constraints (e.g. with limited scope) provide an intuitive way to define restrictions and limitations directly (e.g. graphically) inside a procedural process model, not stated “standalone” as an abstract and declarative constraint problem
  - processes triggered in case of constraint violations are directly executed by the process engine, so constraints are a flexible and declarative formalism, allowing powerful shortenings in process models without taking control from the process engine
  - … but like any other knowledge representation constraints should be used with sense of proportion, excessive usage is not suggested

Conclusion & Outlook

UNIVERSITÄT OSNABRÜCK

- **in business processes often BRMS are used for decision support**
  - even business processes defined and controlled exclusively by business rules are common
  - in large systems approaches using a sole declarative knowledge representation technique introduces serious problems
  - in BPM we suggest to model the control strategy explicit and as exclusively as possible as (procedural) process models
  - process models may be supported by business rules for local decisions, but rules should not be used for global control
  - using rule-based approaches introduces some serious problems because rules contain domain knowledge as well as control strategy
  - rules are declarative, weakly structured, and difficult to manage and maintain
  - for large rule bases the user can not be sure if the problem is completely covered by the rules
  - modifications often result in unwanted consequences

Conclusion & Outlook

UNIVERSITÄT OSNABRÜCK

- **instead of using rules we propose a constraint-based approach**
  - constraints also have a declarative paradigm but avoid the lack of separation between domain knowledge and control strategy
  - constraint solvers used as black box by a process engine will merely compute an output based on the given domain knowledge and return this as new input for the process engine
  - a constraint solver will give control a soon as possible back to the process engine, the process engine exclusively has to decide how to proceed
  - so constraints support a process engine without competing for the control strategy

Conclusion & Outlook

UNIVERSITÄT OSNABRÜCK

- **the challenges of this approach lies in the useful enhancement of BPM with constraint technology**
  - this requires the specification of an appropriate API containing a constraint language to reference business process model elements defining relations on them
  - furthermore constraints are an alternative for the business rules approach: the weakness of the widely used business rules can be moderated by constraint technology and respectively enhanced by constraints
  - so enhancing BPM with constraints may lead to process models containing both, constraint technology and also rules each in adequate contexts
  - another field of application for constraint technology in BPM is the area of QA: known dependencies in process models can be made explicit by additional constraints supporting further revisions by ensuring consistent modelings
  - further research has to be done with respect to process hierarchies, the scope of constraints, the integration of solutions of sub problems, and meta constraint solving

Conclusion & Outlook

- **the next steps will be the preparation of case studies and examples for the application of constraint technology in BPM practice**
  - this should preferably be supported by practitioners and experts in selected fields
  - the intention is to generate more input for requirements relevant in practice