Evaluierung der Konsistenz zwischen Business Process Modellen und Business Role-Object Spezifikation

Lars Westermann

26.09.2019

Institut für Software- und Multimediatechnik, Professur für Softwaretechnologie

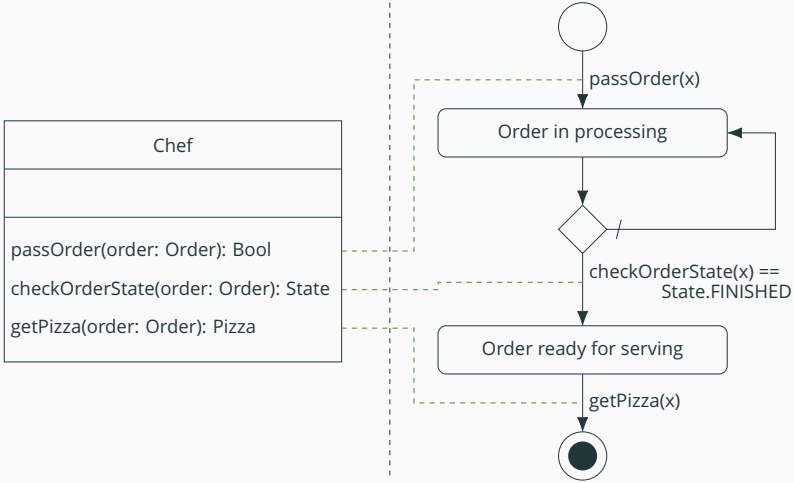
Prof. Dr. rer. nat. habil. Uwe Aßmann
Dr.-Ing. Thomas Küh
Hendrik Schön

Motivation

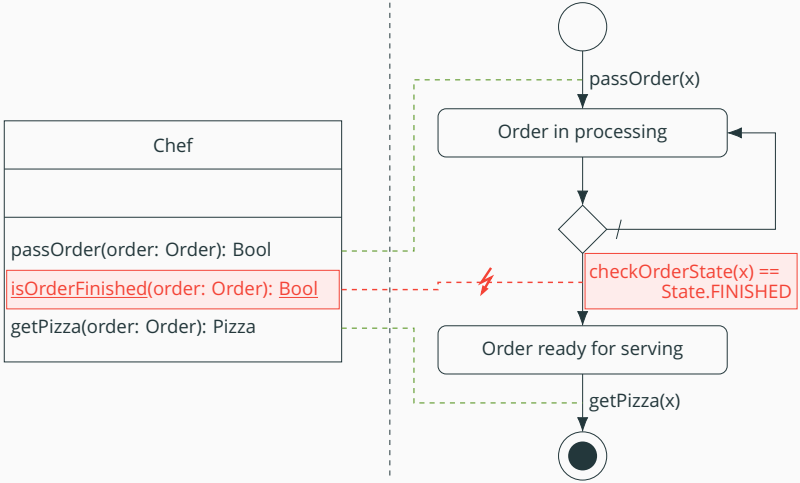
Diskrepanz zwischen Modellierungaspekten:

- **Strukturmodellierung** mittels z.B. UML-Klassen- oder Komponentendiagrammen
- **Verhaltensmodellierung** mittels z.B. UML-Sequenzdiagrammen oder Petrinetzen

Motivation


Chef
passOrder(order: Order): Bool
checkOrderState(order: Order): State
getPizza(order: Order): Pizza
passOrder(x)
Order in processing
checkOrderState(x) ==
State.FINISHED
Order ready for serving
getPizza(x)

Motivation


Chef
passOrder(order: Order): Bool
isOrderFinished(order: Order): Bool
getPizza(order: Order): Pizza
passOrder(x)
Order in processing
checkOrderState(x) ==
State.FINISHED
Order ready for serving
getPizza(x)

Problemdefinition

Es gibt viele Verfahren zur Konsistenzprüfung bestehender Struktur- und Verhaltensmodellierungsprachen

Dies gilt nicht für die Konsistenz zwischen **BPMN** und **BROS**:

- Es existieren noch keine Konsistenzbeziehungen
- Damit auch kein automatisierbares Verfahren zur Konsistenzprüfung

Forschungsfragen

F1 Welche Konsistenzbeziehungen bestehen zwischen BPMN- und BROS-Modellen?

F2 Wie lassen sich die Konsistenzbedingungen automatisiert überprüfen?

F3 Mit welchem Aufwand ist dieses Verfahren erweiterbar?

Welche Konsistenzbeziehungen bestehen zwischen BPMN- und BROS-Modellen?

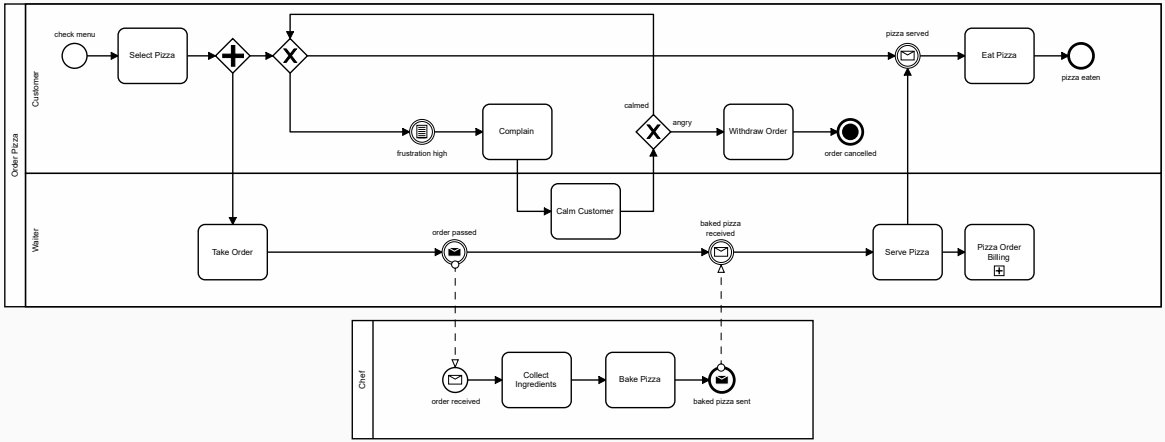
Konsistenzproblem

Eine **Inkonsistenz** tritt genau dann auf, wenn eine Konsistenzregel verletzt wird. (Nuseibeh 1996)

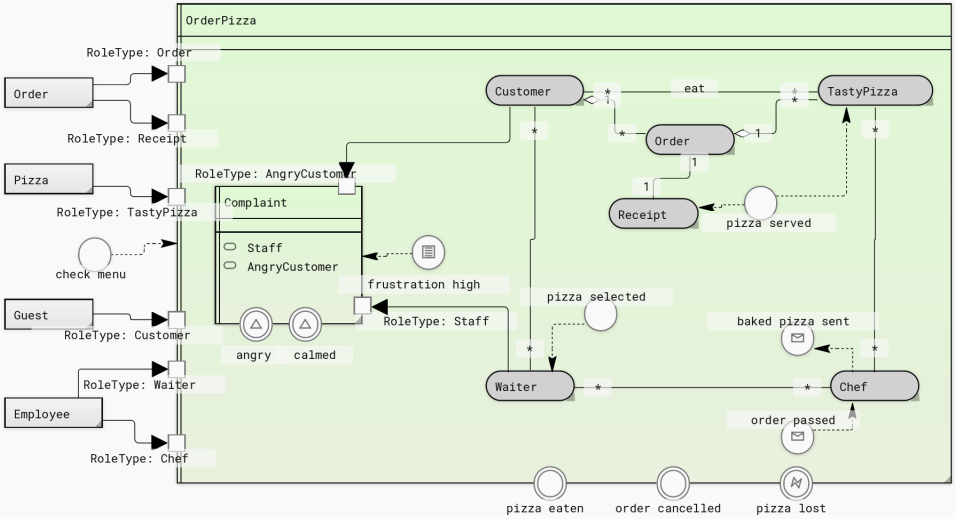
Eine **Konsistenzregel** ist eine Formalisierung von einem Aspekt der Konsistenz zwischen den betrachteten Modellen. Es beschreibt eine Relation oder einen Fakt der notwendig ist.

Das **Konsistenzproblem** beschreibt den Vorgang der Minimierung und Verhinderung von Inkonsistenzen mit Hilfe der Aufstellung und Prüfung von Konsistenzregeln.

Business Process Model and Notation



Business Role-Object Specification



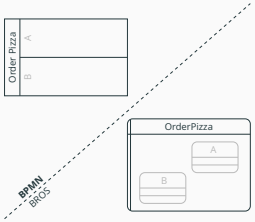
Prolog-Datenstrukutr

```
1  % Definition aller BPMN Elemente.
2  bpmn(Bpmn, Type).
3
4  % Definition aller BROS Elemente.
5  bros(Bros, Type).
6
7  % Definition aller Relationen.
8  relation(Source, Target, Type).
9
10 % Definition der Eltern-Kind Beziehung.
11 parent(Child, Parent).
```

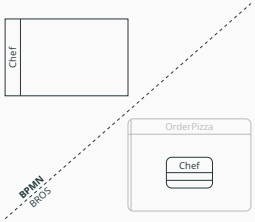
Hilfsfunktionen

```
1  % Konsistenz der Eltern-Kind Beziehung.
2  check_parent(C) :-
3    parent(C, P), parent(C, Q) -> P == Q.
4
5  % Transitiver Abschluss der Modellstruktur.
6  transitive_parent(Child, Parent).
7
8  % Orakel für das Matching von Modellelementen.
9  match(Bpmn, Bros).
```

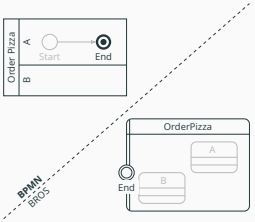
Konsistenzbeziehungen



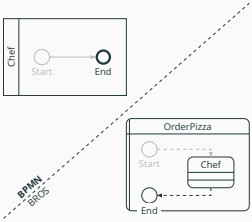
Regel 1: BPMN-Process



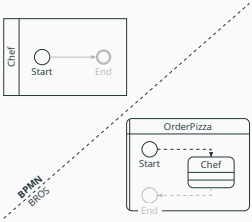
Regel 2: BPMN-Swimlane



Regel 3: BPMN-TerminationEvent



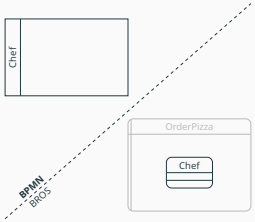
Regel 4: BPMN-EndEvent



Regel 5: BPMN-StartEvent

Konsistenzbeziehungen

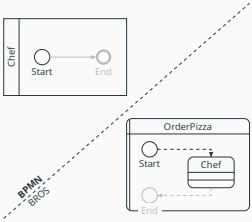
Regel 1: BPMN-Process



Regel 2: BPMN-Swimlane

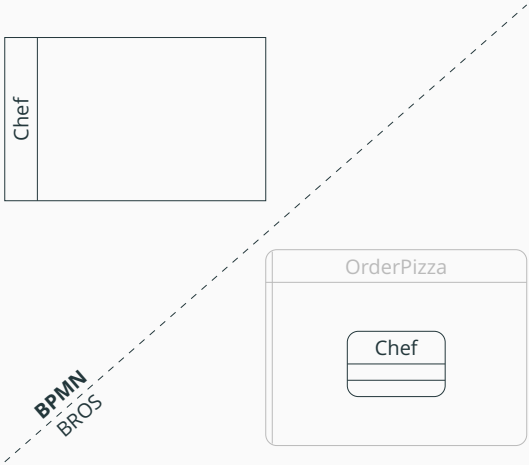
Regel 3: BPMN-TerminationEvent

Regel 4: BPMN-EndEvent



Regel 5: BPMN-StartEvent

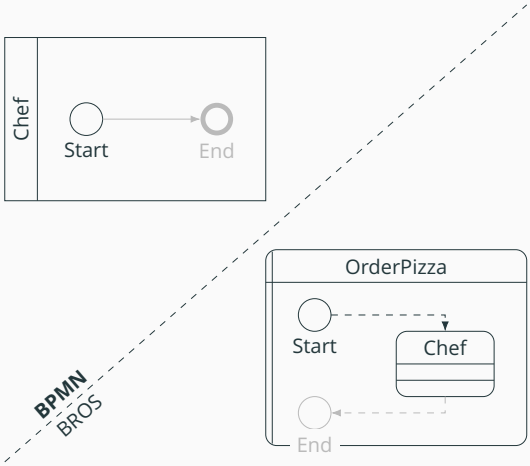
BPMN-Swimlane - BROS-RoleType



BPMN-Swimlane - BROS-RoleType

```
1  rule_2(Bpmn) :- bpmn(Bpmn, "Swimlane") ->
2    (
3      bros(Bros, "RoleType"), match(Bpmn, Bros)
4    ).
```

BPMN-StartEvent - BROS-Event

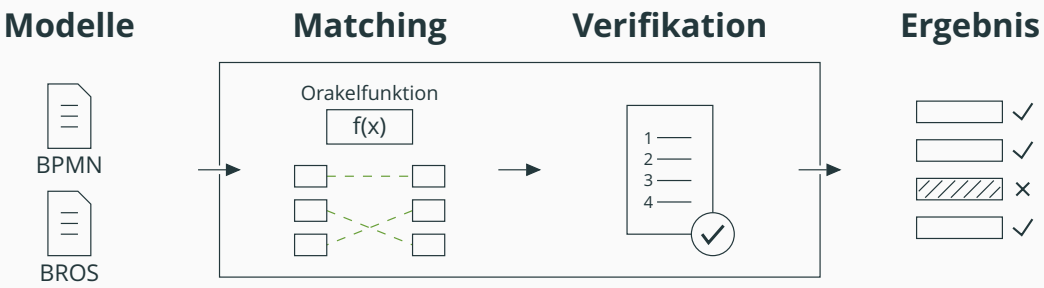


BPMN-StartEvent - BROS-Event

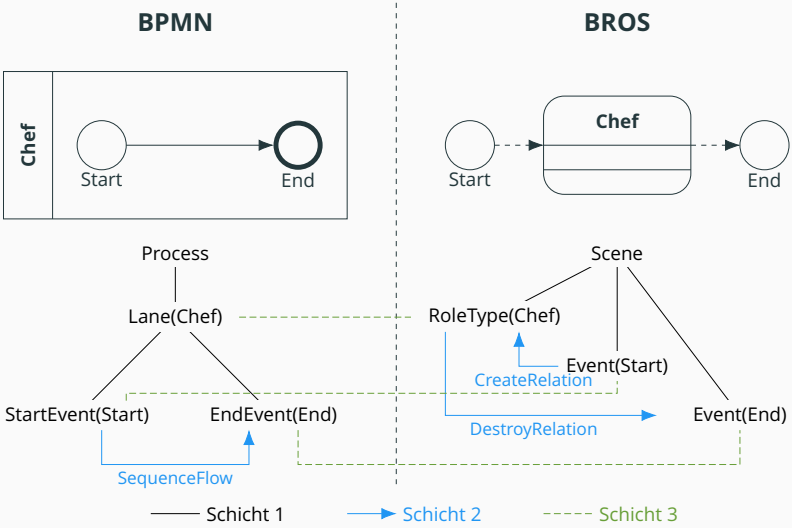
```
1  rule_5(Bpmn) :- bpmn(Bpmn, "StartEvent") ->
2    (
3      bros(Bros, "Event"),
4      match(Bpmn, Bros),
5      (
6        relation(Bros, X, "CreateRelation"),
7        transitive_parent(Bpmn, BpmnParent),
8        match(BpmnParent, X)
9      )
10   ).
```

Wie lassen sich die Konsistenzbedingungen automatisiert überprüfen?

Ablauf der Konsistenzprüfung



Modelle


BPMN
BROS
Chef
Start
End
Chef
Start
End
Process
Lane(Chef)
StartEvent(Start)
EndEvent(End)
SequenceFlow
Scene
RoleType(Chef)
Event(Start)
CreateRelation
DestroyRelation
Event(End)
Schicht 1
Schicht 2
Schicht 3

Matching

Matching der Modellelemente anhand des Namen und des Typs

Algorithmus des Name-Matching:

1. Namen in Teilwörter aufteilen (Anhand von “ ” und Groß/Kleinschreibung)
2. Endung der Teilwörter entfernen (Letzte 2 Zeichen entfernen)
3. Alle Teilwörter des kürzeren Namens müssen im längerem Namen enthalten sein

Matching

’Aktion war erfolgreich’ , ’ErfolgreicheAktion’

{’aktion’, ’erfolgreich’, ’war’} , {’aktion’, ’erfolgreiche’}

{’aktion’, ’erfolgreiche’} , {’aktion’, ’erfolgreich’, ’war’}

{’**akti**on’ $\subseteq$ ’**akti**on’} , {’**erfolgreic**he’ $\subseteq$ ’**erfolgrei**ch’}

Matching

```
1  fun matchStrings(shorterName: String, longerName: String): Boolean {
2    val shorterNameSet = splitNameToSet(shorterName)
3    val longerNameSet = splitNameToSet(longerName)
4
5    return shorterNameSet.all { short ->
6      val s = trimEnding(short)
7
8      longerNameSet.any { long ->
9        val l = trimEnding(long)
10
11       long.startsWith(s) || short.startsWith(l)
12     }
13   }
14 }
```

Matching

```
1 match<BpmnLane, BrosRoleType> { lane, role ->
2   matchStrings(lane.element.name, role.element.name)
3 }
```

Verifikation

```
1  rule_2(Bpmn) :- bpmn(Bpmn, "Swimlane") ->
2    (
3      bros(Bros, "RoleType"), match(Bpmn, Bros)
4    ).
```

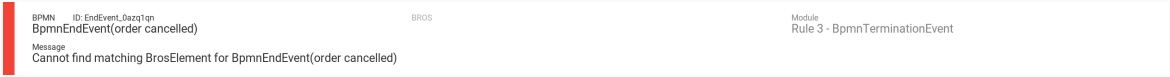
Verifikation

```
1  Context.verifyBpmn<BpmnLane> { bpmn ->
2    for (bros in bpmn.matchingElements) {
3      if (bros.checkType<BrosRoleType>()) {
4        return Result.match("...", bros = bros)
5      }
6    }
7    return Result.error("...")
8  }
```

Ergebnisse

Positive und Negative Konsistenzmeldungen

- Referenz auf Modellelemente
- Regel die zur Konsistenzmeldungen geführt hat
- Textuelle Beschreibung der Ursache



Demo

Mit welchem Aufwand ist dieses Verfahren erweiterbar?

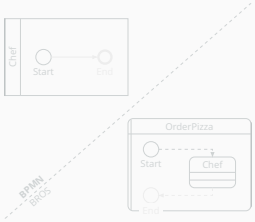
Konsistenzbeziehungen

Regel 1: BPMN-Process

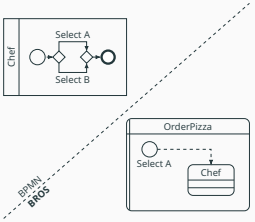
Regel 2: BPMN-Swimlane

Regel 3: BPMN-TerminationEvent

Regel 4: BPMN-EndEvent

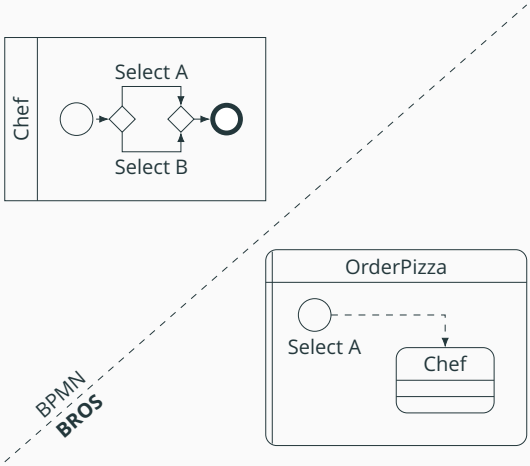


Regel 5: BPMN-StartEvent



Regel 6: BROS-Event

BROS-Event - BPMN-Element



BROS-Event - BPMN-Element

```
1  rule_6(Bros) :- (bros(Bros, "Event"); bros(Bros, "ReturnEvent")) ->
2    (
3      bpmn(Bpmn, "StartEvent"), match(Bpmn, Bros);
4      bpmn(Bpmn, "EndEvent"), match(Bpmn, Bros);
5      bpmn(Bpmn, "TerminationEvent"), match(Bpmn, Bros);
6      bpmn(Bpmn, "Event"), match(Bpmn, Bros);
7      bpmn(Bpmn, "Activity"), match(Bpmn, Bros);
8      bpmn(Bpmn, "Gateway"), match(Bpmn, Bros)
9    ).
```

BROS-Event - BPMN-Element

```
1  match<BpmnTask, BrosEvent> { bpmn, bros ->
2    matchStrings(bpmn.element.name, bros.element.desc)
3  }
```

BROS-Event - BPMN-Element

```
1  verifyBros<BrosEvent> { bros ->
2    for (bpmn in bros.matchingElements) {
3      if (bpmn.checkType<BpmnElement>()) {
4        return@verifyBros Result.match("...", bpmn = bpmn)
5      }
6    }
7    Result.error("...")
8  }
```

Ergebnisse und Ausblick

Forschungsfragen

F1 Welche Konsistenzbeziehungen bestehen zwischen BPMN- und BROS-Modellen?

F2 Wie lassen sich die Konsistenzbedingungen automatisiert überprüfen?

F3 Mit welchem Aufwand ist dieses Verfahren erweiterbar?

Klassifikationsschema

	Diagrams	Consistency Type	Consistency Strategy	Intermediate Representation	Case Study	Automatable	Tool Support	Model Extensibility	Rule Extensibility
Rasch 2003	CD, SM	Intra	Monitoring	CSP/OZ	✓	●	✗	●	◐
Shinkawa 2006	UCD, CD, SD, AD, SC	Inter	Analysis	CPN	✗	●	✗	◐	○
Mens 2005	CD, SD, SC	All	Monitoring	Extended UML	✓	●	✓	●	◐
Egyed 2001	CD, OD, SD	Intra, Inter	Construction		✗	●	∼	◐	◐
Egyed 2006	CD, SD, SC	Intra	Monitoring		✓	●	✓	○	◐
BROS	BPMN, BROS	Intra	Monitoring		✓	●	✓	○	●

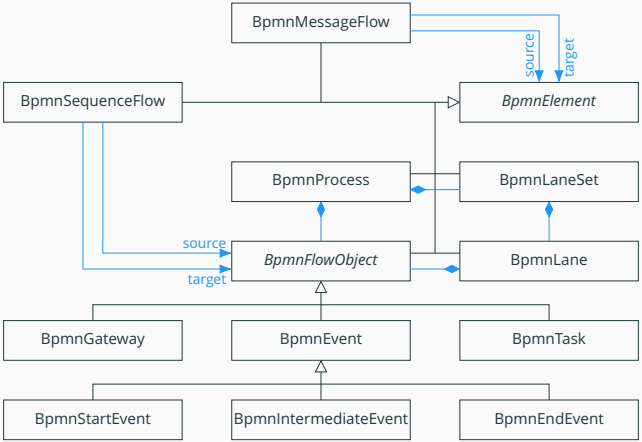
- ● mit geringem Aufwand
- ◐ mit mittlerem Aufwand
- ○ mit hohem Aufwand
- ✓ ja
- ✗ nein
- ∼ teilweise
- **CD** Class Diagram
- **SM** State Machine
- **USC** Use Case Diagram
- **SD** Sequence Diagram
- **AD** Activity Diagram
- **SC** Statechart

Quellen

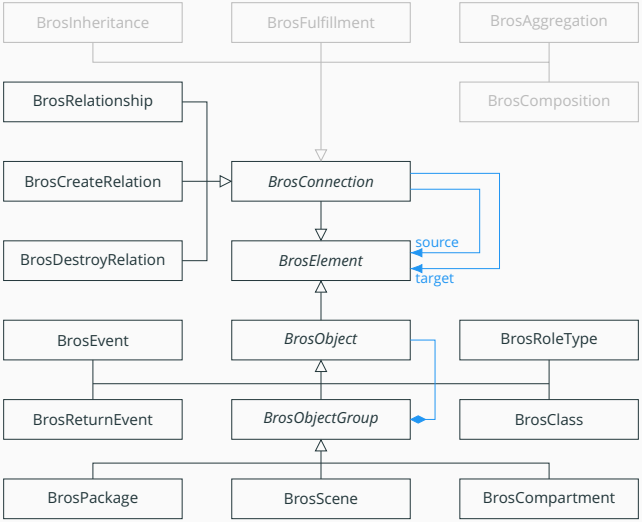
Nuseibeh, Bashar (1996). **?**To be and not to be: On managing inconsistency in software development**?** In: *Proceedings of the 8th International Workshop on Software Specification and Design*. IEEE, pp. 164–169.

Fragen?

Metamodell der Business Process Model and Notation



Metamodell der Business Role-Object Specification



Metamodell der Graphstruktur