Specific Relationship Types in Conceptual Modeling
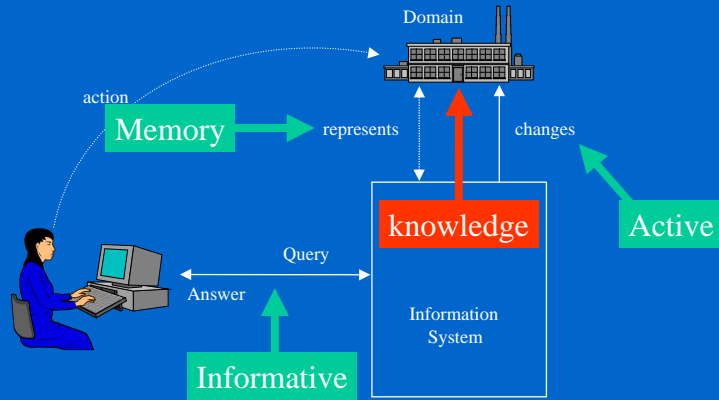
The Cases of Generic and with Common Participants

Antoni Olivé
Universitat Politècnica de Catalunya

Outline

- Conceptual Modeling of Enterprise Information Systems.
- Specific Structures of Knowledge in Conceptual Modeling.
- Generic Relationship Types.
    - Two new representation methods.
- Relationship Types with Common Participants.
    - One new representation method.
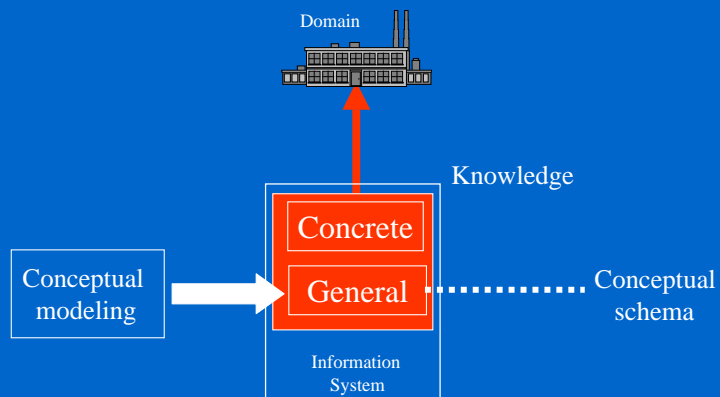- Concluding Remarks.

Enterprise Information System
Functions

Domain

action

Memory → represents

changes

knowledge

Active

Query

Answer

Informative

Information System

Conceptual Modeling

Domain

Knowledge

Concrete

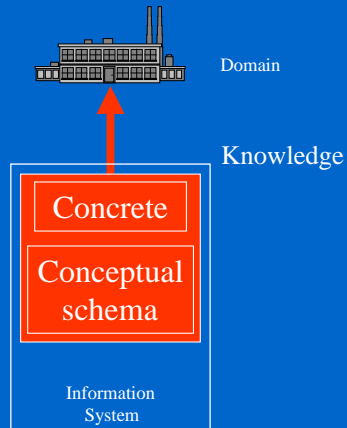Conceptual modeling →

General ········ Conceptual schema

Information System

Conceptual Schema
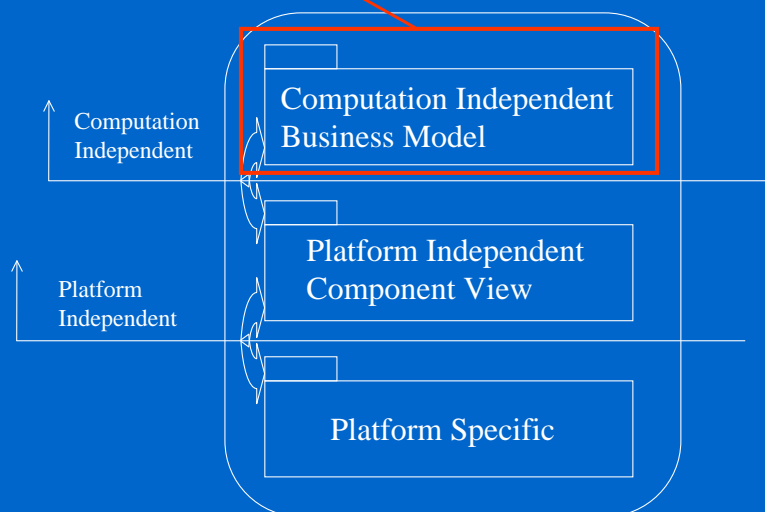
General knowledge about a domain
an information system needs to know
in order to perform the required functions

Domain

Knowledge

Concrete

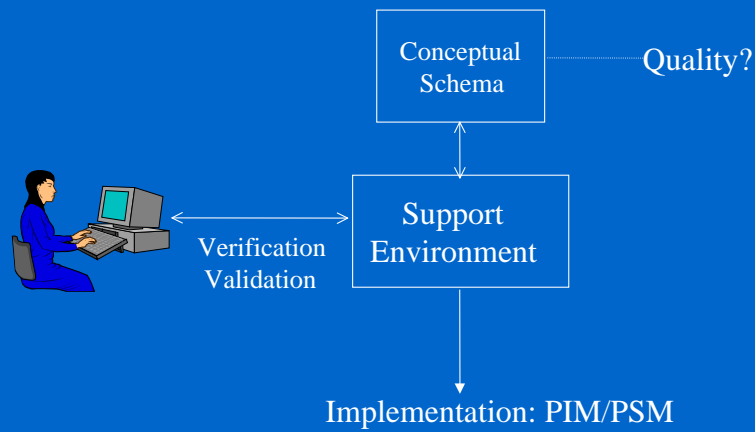Conceptual
schema

Information
System

ICEIS 2002

5

Conceptual Schemas in the OMG's MDA

Computation
Independent

Computation Independent
Business Model

Platform
Independent

Platform Independent
Component View

Platform Specific

ICEIS 2002

6

3

Some R&D Tracks in Conceptual Modeling

```
                    ┌──────────────┐
                    │  Conceptual  │········ Quality?
                    │    Schema    │
                    └──────────────┘
                           ↑
                           ↓
         ┌─────────┐  ┌──────────────┐
         │ [image] │←→│   Support    │
         └─────────┘  │ Environment  │
    Verification      └──────────────┘
    Validation               │
                             ↓
              Implementation: PIM/PSM
```
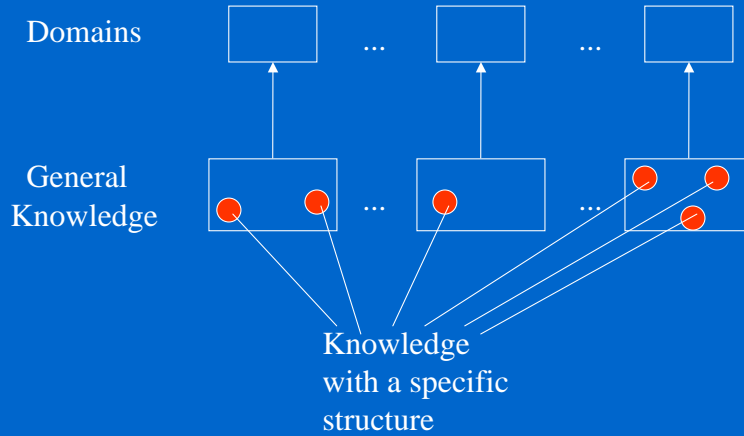

Outline

- Conceptual Modeling of Enterprise Information Systems.
- **Specific Structures of Knowledge in Conceptual Modeling.**
- Generic Relationship Types.
  - Two new representation methods.
- Relationship Types with Common Participants.
  - One new representation method.
- Concluding Remarks.

Specific Structures of Knowledge

Domains

General
Knowledge

Knowledge
with a specific
structure

ICEIS 2002

9

Specific Structures of Knowledge
Examples
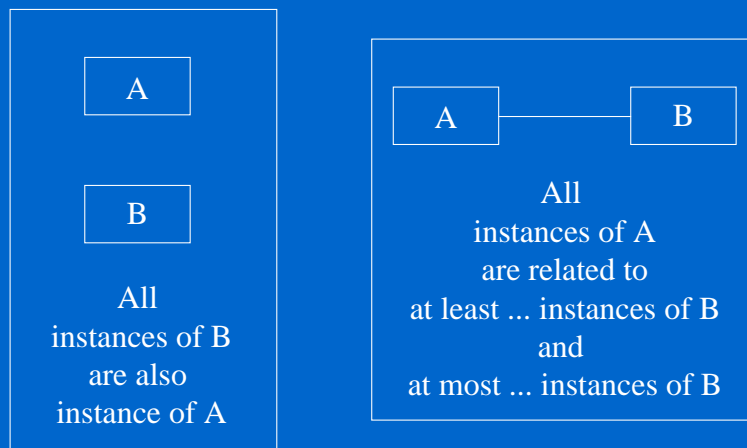
A

B

All
instances of B
are also
instance of A

A — B

All
instances of A
are related to
at least ... instances of B
and
at most ... instances of B

ICEIS 2002

10

5

Interest of Specific Structures of Knowledge

We can develop special ...

- Constructs for their adequate representation.

- Procedures for an effective reasoning.

- Techniques for their efficient implementation.

Specific Structures of Knowledge
Example

A

B

All
instances of B
are also
instance of A

A

B

Transitive
Irreflexive

TableA (Id, Atr1,.....)

TableB (Id,Atr3,...)

Specific Structures of Knowledge
Example

A — B

All
instances of A
are related to
at least ... instances of B
and
at most ... instances of B

A — 0..1 — * — B
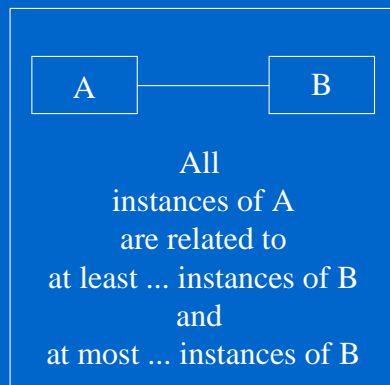
Conditions for the
satisfiability of
cardinality constraints
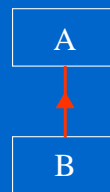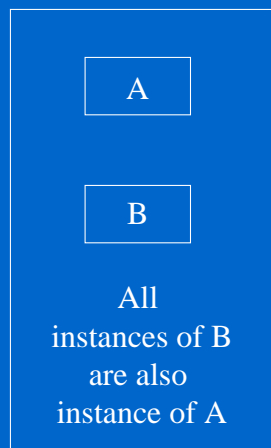
Methods for the
enforcement
in several data models

Interest of Specific Structures of Knowledge

We can develop special ...

• Constructs for their adequate representation.

• Procedures for an effective reasoning.

• Techniques for their efficient implementation.

Outline

- Conceptual Modeling of Enterprise Information Systems.
- Specific Structures of Knowledge in Conceptual Modeling.
- ## Generic Relationship Types.
  - Two new representation methods (adapted to the UML).
- Relationship Types with Common Participants.
  - One new representation method.
- Concluding Remarks.

Generic Relationship Types

Member	MemberOf	Group

Realizations

TennisPlayer	TennisClub
Person	Committee
Person	ProjectTeam
Manager	BoardOf Directors
~~Committee~~	~~Person~~
~~Manager~~	~~Person~~

Generic Relationship Types

Member ──── MemberOf ──── Group

TennisPlayer

Person

Manager

TennisClub

Committee

ProjectTeam

BoardOf
Directors

Valid combinations?
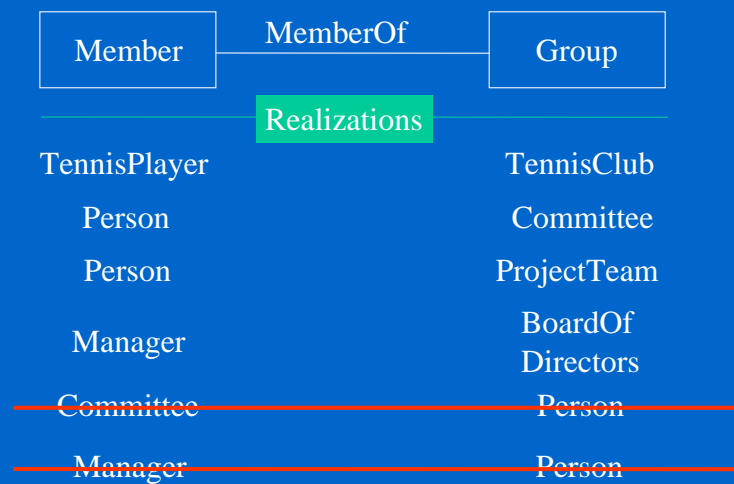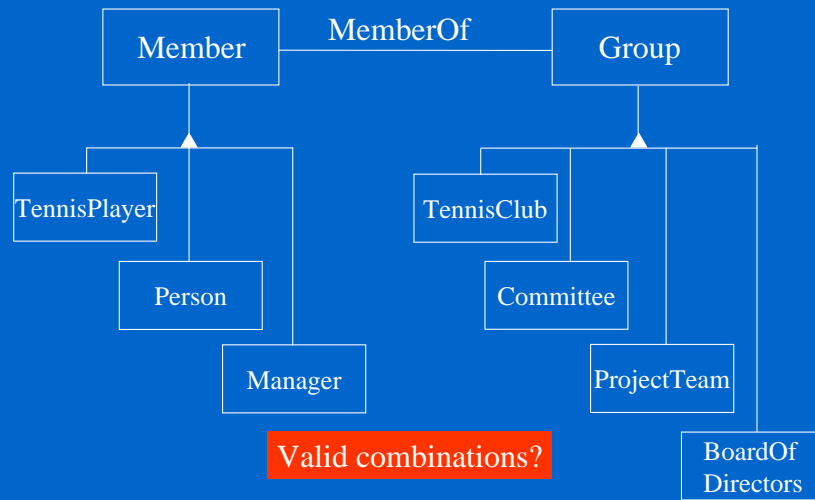
Generic Relationship Types
Realization Constraint

Member ──── MemberOf ──── Group

MemberOf (m,g) ➡

- m InstanceOf TennisPlayer AND g InstanceOf TennisClub
  OR
- m InstanceOf Person AND g InstanceOf Committee
  OR
- m InstanceOf Person AND g InstanceOf ProjectTeam
  OR
- m InstanceOf Manager AND g InstanceOf BoardOfDirectors

Generic Relationship Types
Example: PartOf

whole

Entity

part

IsPartOf

part	Realizations	whole
Division		Company
Company		Company
Office		Building

Generic Relationship Types
Example: Materializes

Abstract ◄ Materializes Concrete

Realizations

CarModel	Car
Play	Performance
Book	Volume

Generic Relationship Types in Ontologies
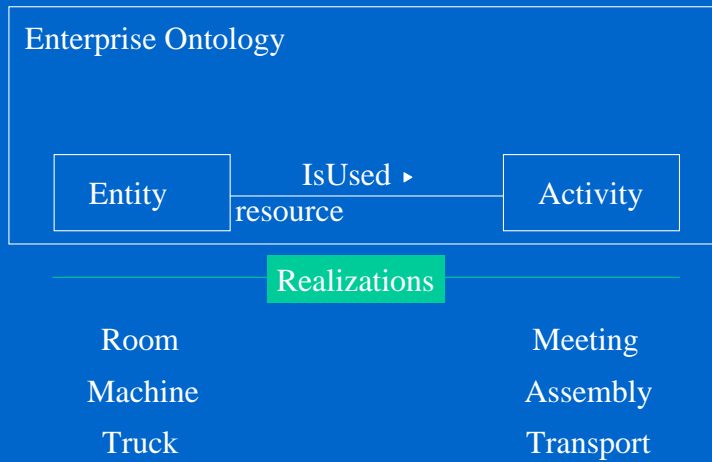
Enterprise Ontology

Entity	IsUsed ▸	Activity
	resource	

Realizations

Room	Meeting
Machine	Assembly
Truck	Transport

Generic Relationship Types in Ontologies

Enterprise Ontology

Entity	Performs ▸	Activity
	doer	

Realizations

Committee	Meeting
Machine	Assembly
Driver	Transport
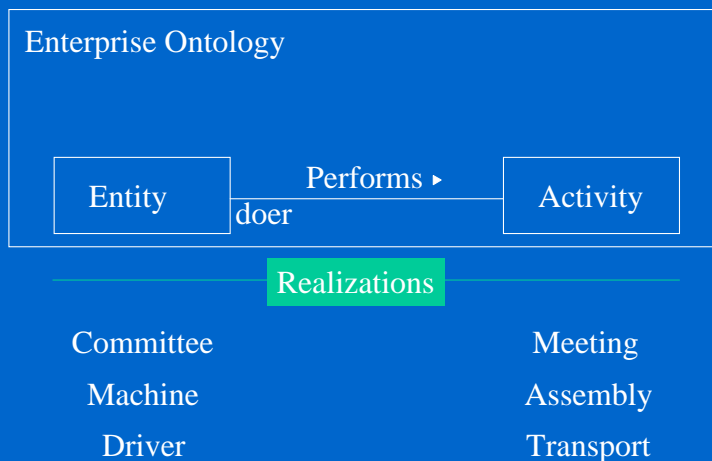
Generic Relationship Types
Representation

Usually: One relationship type for each realization

Person	— P_MemberOf_PT —	ProjectTeam
Person	— P_MemberOf_C —	Committee
Manager	— M_MemberOf_BD —	BoardOf Directors

Different relationship types
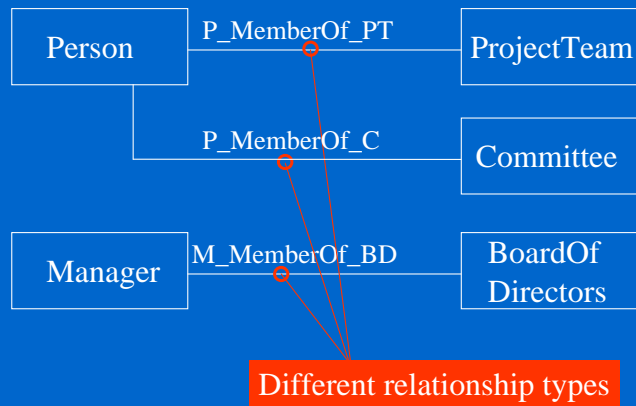
One relationship type for each realization

The generic relationship type is not represented:

| Member | — MemberOf — | Group |

Only its realizations:

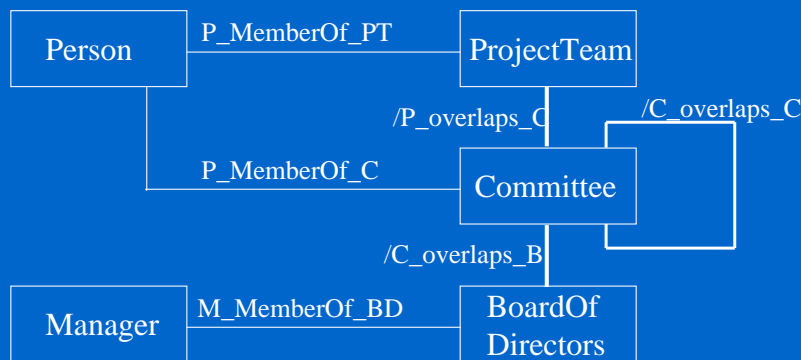| Person | — P_MemberOf_PT — | ProjectTeam |
| Person | — P_MemberOf_C — | Committee |
| Manager | — M_MemberOf_BD — | BoardOf Directors |
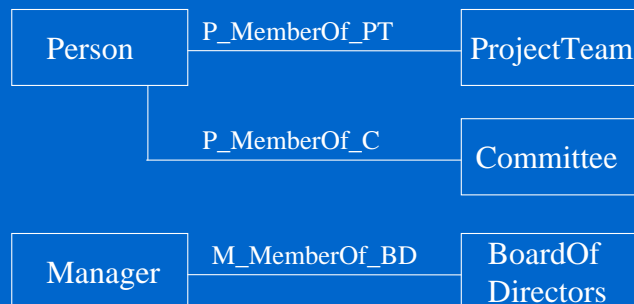
One relationship type for each realization

Problem:
Definition of knowledge related to all instances of MemberOf

```
┌──────────┐   P_MemberOf_PT   ┌──────────────┐
│  Person  │───────────────────│  ProjectTeam │
└──────────┘                   └──────────────┘
      │                    /P_overlaps_C      /C_overlaps_C
      │       P_MemberOf_C  ┌─────────────┐
      └────────────────────│  Committee  │
                           └─────────────┘
                    /C_overlaps_B
┌──────────┐  M_MemberOf_BD   ┌──────────────┐
│ Manager  │──────────────────│   BoardOf    │
└──────────┘                  │  Directors   │
                              └──────────────┘
```
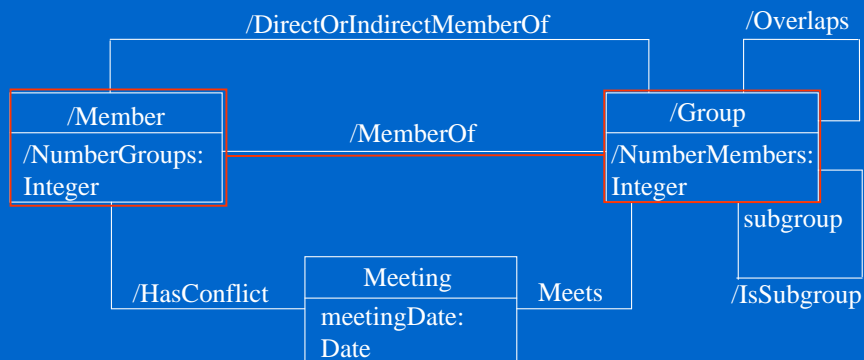

One relationship type for each realization

How do we define additional knowledge related to the
generic type?

```
        /DirectOrIndirectMemberOf              /Overlaps
   ┌─────────────────────────────────────┐   ┌───────┐
┌──────────────┐              ┌──────────────────┐
│ /Member      │  /MemberOf   │ /Group           │
│ /NumberGroups:│─────────────│ /NumberMembers:  │
│ Integer      │              │ Integer          │
└──────────────┘              └──────────────────┘
                                         subgroup
   /HasConflict  ┌───────────┐  Meets    /IsSubgroup
                 │ Meeting   │
                 │ meetingDate:│
                 │ Date      │
                 └───────────┘
```

13

The Realizations as Subtypes Method

/Member — /MemberOf — /Group

Person — P_MemberOf_PT — ProjectTeam

P_MemberOf_C — Committee

Manager — M_MemberOf_BD — BoardOf Directors
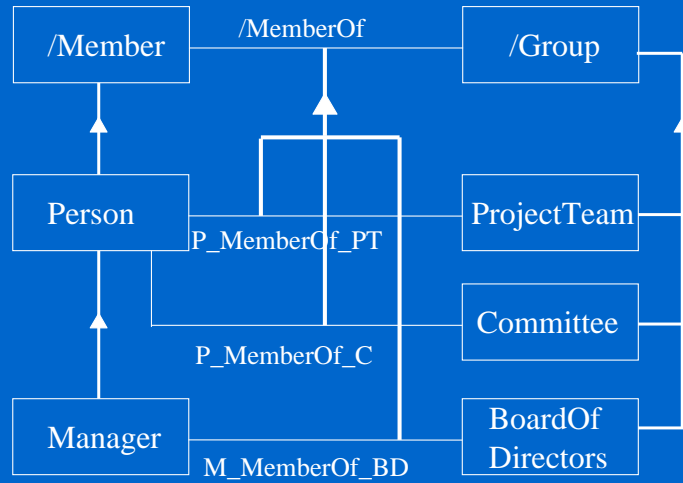
27

Integration with generic type knowledge

/DirectOrIndirectMemberOf

/Overlaps

/Member
/NumberGroups: Integer

/MemberOf

/Group
/NumberMembers: Integer

subgroup

/IsSubgroup

Person — P_MemberOf_PT — ProjectTeam

P_MemberOf_C — Committee

Manager — M_MemberOf_BD — BoardOf Directors

28

14
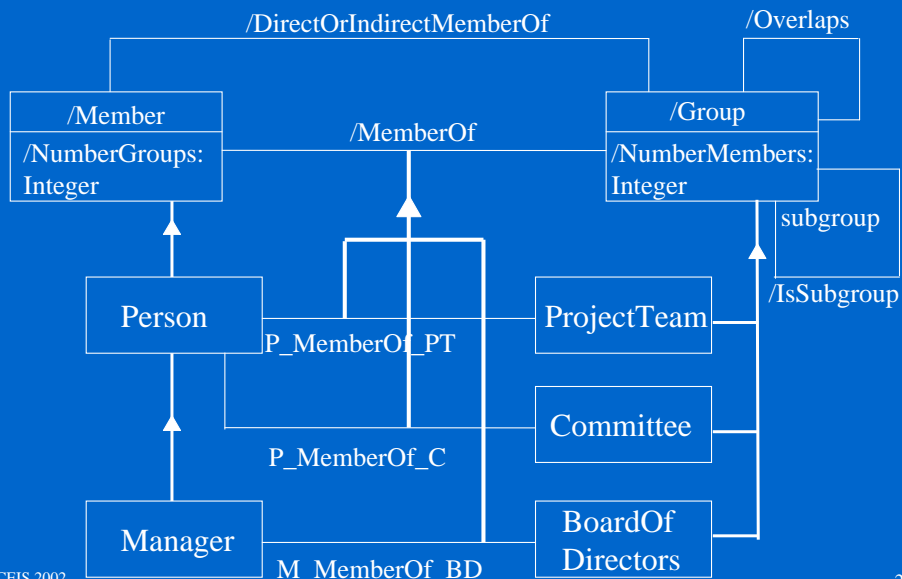
The Realizations as Subtypes Method
Drawbacks?

/DirectOrIndirectMemberOf /Overlaps

/Member
/NumberGroups: Integer

/MemberOf

/Group
/NumberMembers: Integer

subgroup

/IsSubgroup

Person

P_MemberOf_PT

ProjectTeam

Committee

P_MemberOf_C

Manager

M_MemberOf_BD

BoardOf Directors

ICEIS 2002

29

Member	MemberOf	Group

Person

ProjectTeam

p

Committee

Realization constraint?

Manager

BoardOf Directors

bod

ICEIS 2002

30

15
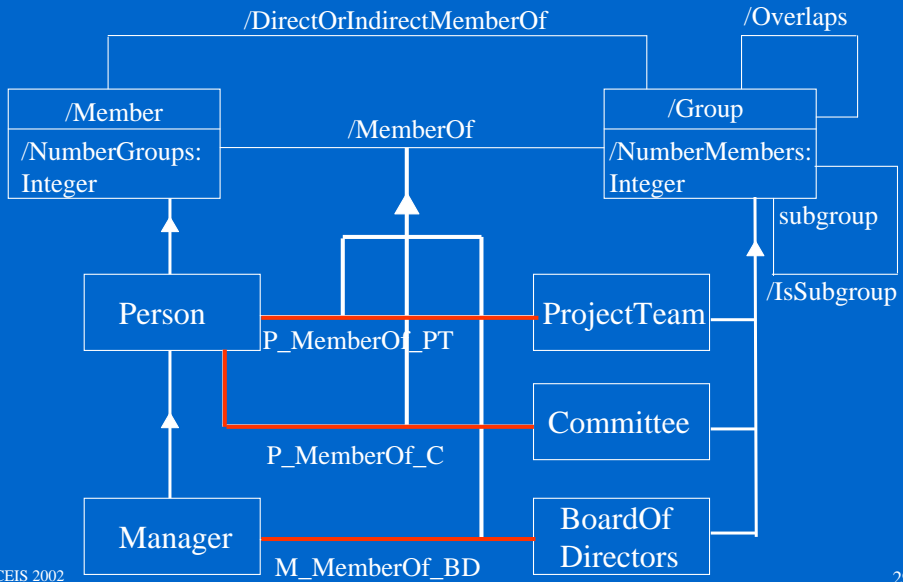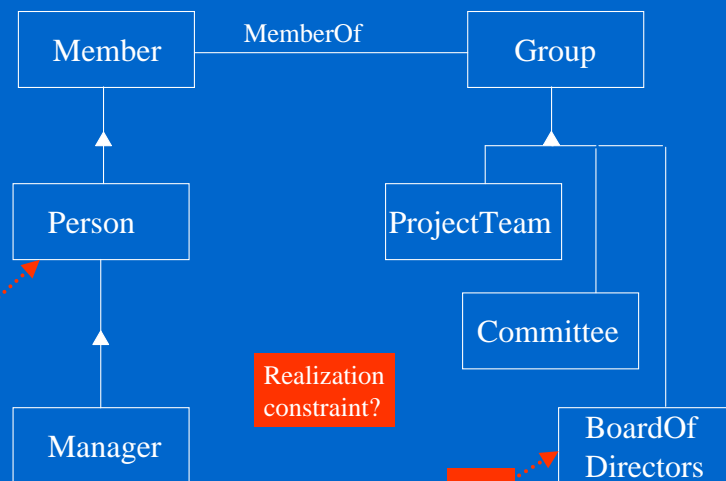
Defining the Realizations

<<metaclass>> MemberType — GovMemberOf — <<metaclass>> GroupType

member   group

<<instanceOf>>   <<instanceOf>>

Person — GovMemberOf — ProjectTeam

GovMemberOf — Committee

Manager — GovMemberOf — BoardOf Directors

ICEIS 2002   31

The Governs Constraint

<<metaclass>> MemberType — GovMemberOf — <<metaclass>> GroupType

member   group

{<<governs>>}

Member — MemberOf — Group

ICEIS 2002   32

16
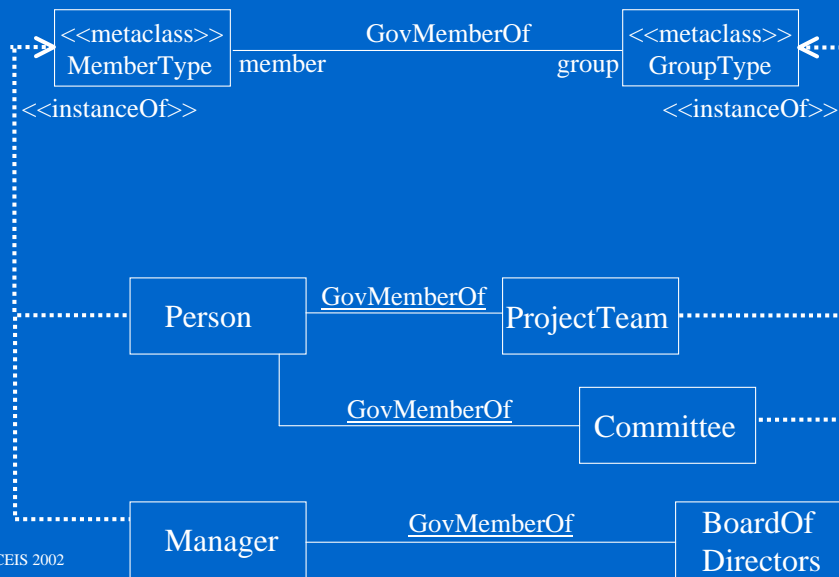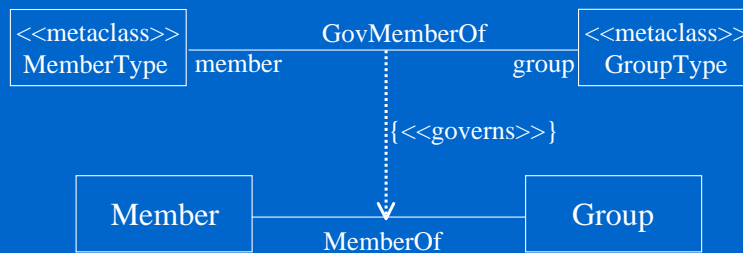
The Metalevel-governed Method
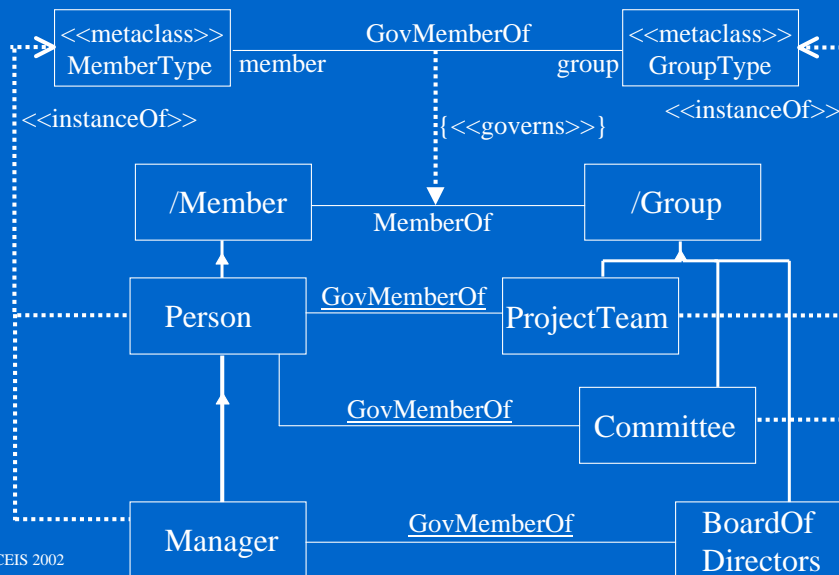
	GovMemberOf	
<<metaclass>> MemberType	member	<<metaclass>> GroupType

<<instanceOf>>   {<<governs>>}   <<instanceOf>>

/Member — MemberOf — /Group

Person — GovMemberOf — ProjectTeam

GovMemberOf — Committee
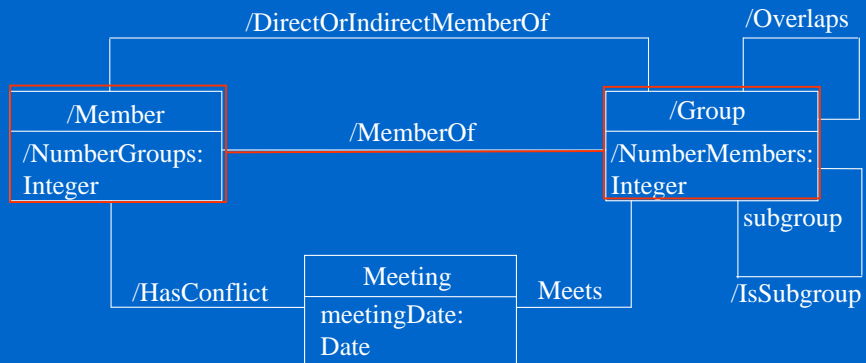
Manager — GovMemberOf — BoardOf Directors

33

Realizations as Subtypes (RS)
vs.
Metalevel-governed (MG)

• Definition of knowledge related to:

   • Generic relationship type: Equivalent.

34

17

Knowledge related to the generic type

/DirectOrIndirectMemberOf

/Overlaps

/Member
/NumberGroups: Integer

/MemberOf

/Group
/NumberMembers: Integer

subgroup

/HasConflict

Meeting
meetingDate: Date

Meets

/IsSubgroup

Realizations as Subtypes (RS)
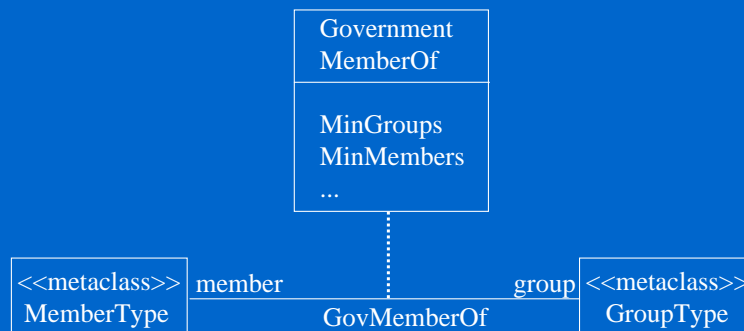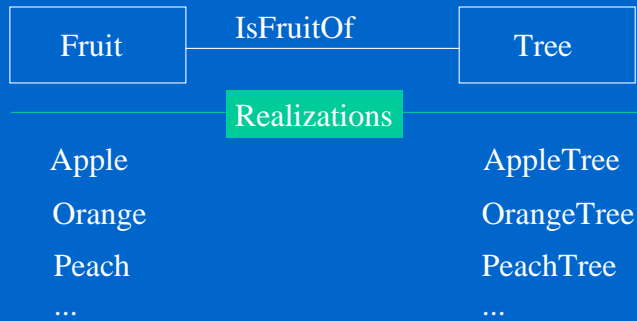## vs.
Metalevel-governed (MG)

- Definition of knowledge related to:

  - Generic relationship type: Equivalent.
  - Each realization: Easier in the MG (extended).

Reification of the Governing RT

```
                    ┌─────────────────┐
                    │ Government      │
                    │ MemberOf        │
                    ├─────────────────┤
                    │ MinGroups       │
                    │ MinMembers      │
                    │ ...             │
                    └────────┊────────┘
                             ┊
┌──────────────────┐        ┊         ┌──────────────────┐
│ <<metaclass>>    │ member ┊   group │ <<metaclass>>    │
│ MemberType       │────────┴─────────│ GroupType        │
└──────────────────┘  GovMemberOf     └──────────────────┘
```

Realizations as Subtypes (RS)
vs.
Metalevel-governed (MG)

• Definition of knowledge related to:

  • Generic relationship type: Equivalent.
  • Each realization: Easier in the MG (extended).
  • Particular realization: Easier in the RS.

• Both methods allow querying the schema
  about the defined realizations.
• Defining new realizations is easier in the MG.
• Simplicity:
  • Structurally: Equivalent.
  • Behaviorally: Simpler in the MG.

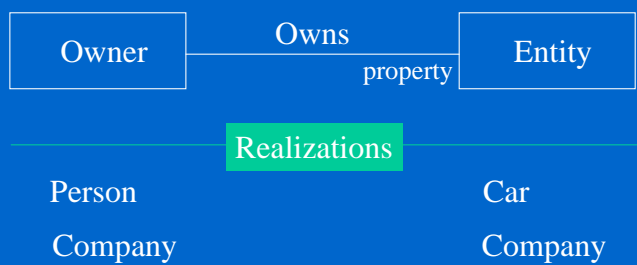## In One Extreme, the MG Is Better

Fruit	IsFruitOf	Tree

Realizations

Apple	AppleTree
Orange	OrangeTree
Peach	PeachTree
...	...

Many realizations
No knowledge related to particular realizations

In the Other Extreme, the RS Is Better

Owner	Owns property	Entity

Realizations

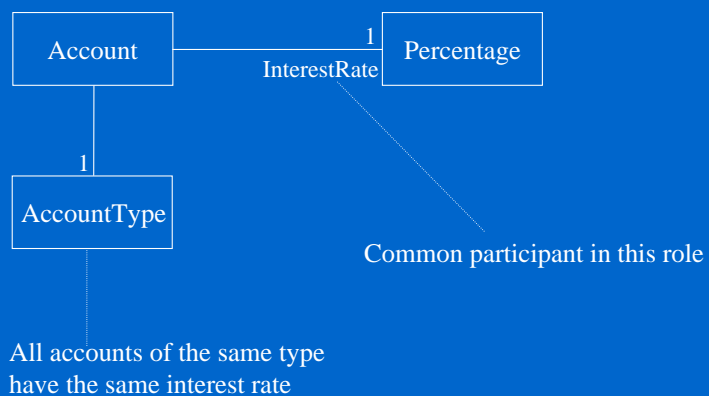|	
Person	Car
Company	Company

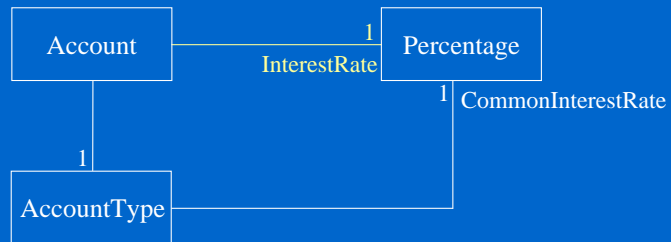Few realizations
Important differences between them

Outline

- Conceptual Modeling of Enterprise Information Systems.
- Specific Structures of Knowledge in Conceptual Modeling.
- Generic Relationship Types.
  - Two new representation methods (adapted to the UML).
- **Relationship Types with Common Participants.**
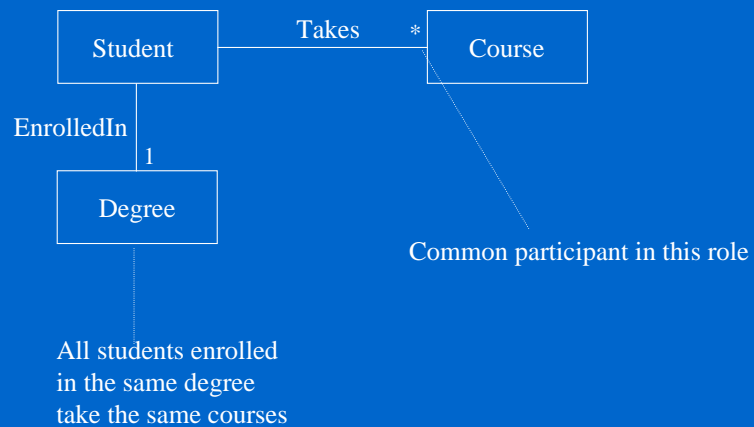 - One new representation method (adapted to the UML).
- Concluding Remarks.

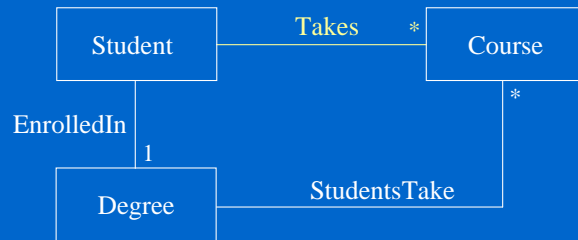## Relationship Type
with Common Participants (RCP)

Account — InterestRate — 1 — Percentage

Account — 1 — AccountType

Common participant in this role

All accounts of the same type
have the same interest rate

21

Usually, RCPs are implicit

```
          1   ┌──────────────┐
  ┌──────────┐     │  Percentage  │
  │  Account │─────┤              │
  └──────────┘ InterestRate └──────────────┘
       │                    1  CommonInterestRate
       │ 1
  ┌──────────────┐
  │  AccountType │──────────────────┘
  └──────────────┘
```

Relationship Type
with Common Participants (RCP)

```
  ┌──────────┐   Takes    *  ┌──────────┐
  │  Student │───────────────│  Course  │
  └──────────┘               └──────────┘
       │                          ╲
EnrolledIn                         ╲
       │ 1                          ╲
  ┌──────────┐
  │  Degree  │
  └──────────┘          Common participant in this role
       ┊

All students enrolled
in the same degree
take the same courses
```

22

Usual Representation of RCPs

```
                    Takes      *
  [ Student ]————————————————[ Course ]
       |                           |
  EnrolledIn                       | *
       | 1
  [ Degree ]——————StudentsTake—————
```

Timetable

ClassRoom: A4203		Monday	Tuesday	Wedn.	Thursday	Friday
	9	IS		Math.	Math.	SE
	10	IS	IS			
	11	DB				
	12		DB			
	13	SE	SE			

```
                      Timetable
                         ◇
        _____/|_____
       |              /     \              |
   [ Class ]  [ DayOfWeek ] [ HourOfDay ] [ Classroom ]
```
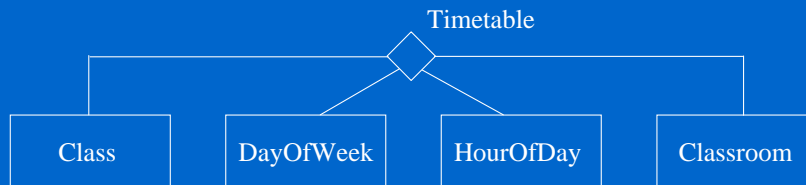
Which class meets in classroom A4202 at 12:00/30/March/2002?

Timetable

Timetable

Class | DayOfWeek | HourOfDay | Classroom

Common participant

1 | 1

Hour

Meets
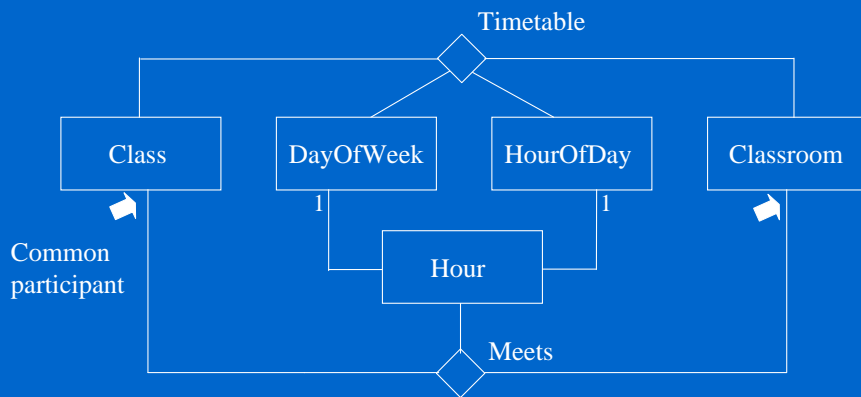
All instances with an Hour corresponding
to the same day of week and hour of day
have the same {(class, classroom)}

Binary RCP

E1 | R | E2
p1 | | p2

R is an RCP with common participant in role p2 if:
- The extension of E1 at a given time
  can be partitioned into a set of subsets, and
- All entities belonging to the same subset of E1
  are related in R to the same entities in the role p2.

Binary RCP
Example

Account — R / interestRate — Percentage

R is an RCP with common participant in role interestRate because:

- The extension of Account at a given time can be partitioned into a set of subsets, one for each AccountType.

- All entities belonging to the same subset (AccountType) are related in R to the same entities (percentage) in the role interestRate.

Variant Common to All

Account — R / interestRate — Percentage

- All Accounts are related to the same Percentage.
- (This percentage is common to all accounts).
- (In this variant, the extension of Account is partitioned into a single subset).

Variant Common in Subtype

Account —— R / interestRate —— Percentage

Account {disjoint, complete} — Savings Account, Checking Account
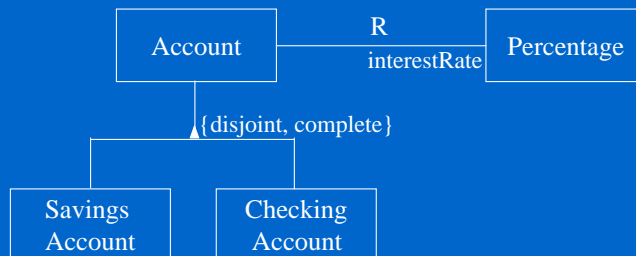
- Account is partitioned into two subtypes (subsets).
- All accounts of the same subtype have the same interestRate.
- The common role is interestRate.

Variant Common in Image

Account —— R / InterestRate —— 1 Percentage

function ···· F 1 — AccountType

1 CommonInterestRate
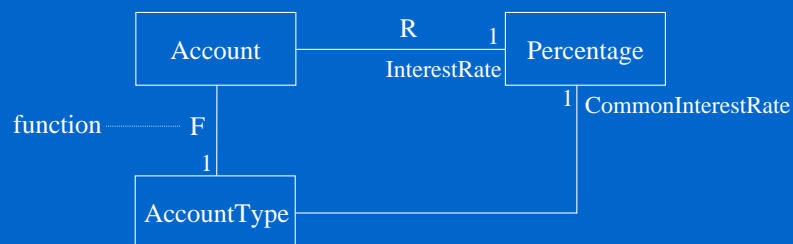
- Function F induces a partition on Account.
- A subset of the partition includes all accounts that have the same image under F.
- All accounts with the same image have the same interestRate.

Variant Common in Image
Representation in the UML
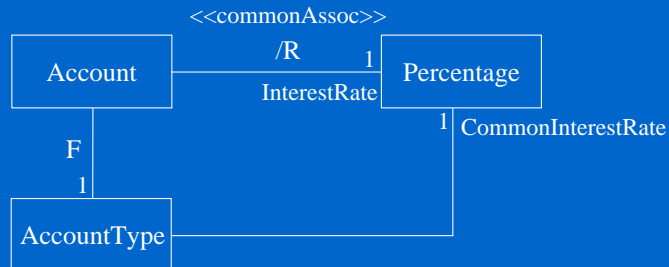
<<commonAssoc>>

/R   1

| Account | InterestRate | Percentage |

F
1

1 CommonInterestRate

| AccountType |

Tags:
• commonRole = interestRate
• variant = CommonInImage
• partitionedBy = F
• commonSet = commonInterestRate

Exceptions

<<commonAssoc>>

/R   1

| Account | InterestRate | Percentage |

F
1

1 CommonInterestRate

particularIR

| AccountType |
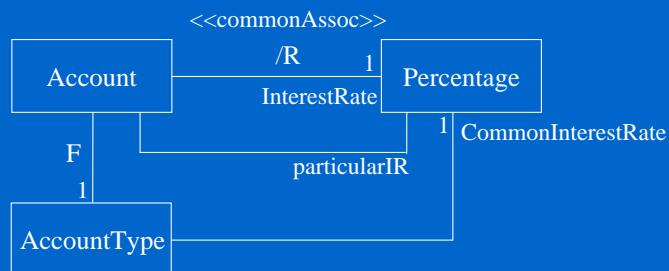
Tags:
• commonRole = interestRate
• variant = CommonInImage
• partitionedBy = F
• commonSet = commonInterestRate
• exception = particular IR

Methods of Representation of RCPs

• Implicit representation.

• Explicit representation:
  • As ordinary derived relationship type.
  • Using a special modeling construct (stereotypes).

Methods of Representation of RCPs
Implicit vs. Explicit

Evaluation based on references to instances of RCPs:
  • In constraints.
  • In derivation rules.
  • In pre/post conditions of operations.

Advantages of explicit representation:
  – Notational Economy.
  – Notational Consistency.
  – Evolution.

Methods of Representation of RCPs
Ordinary vs. Special Modeling Construct

The use of a special modeling construct (stereotype) provides three main benefits:

- Easier and more expressive representation. It is easier to define R:
  - saying that it is an RCP in the variant X with exceptions, than
  - giving the derivation rule (in the OCL).
- Development of special reasoning procedures.
  - Paraphrasing relationship types in natural language.
  - How can I change the interest rate of a particular account?.
  - ...
- Development of special implementation procedures.

Conclusions

- Conceptual modeling plays a key role in EIS development.
- The identification of specific structures of knowledge allows us to develop special:
  - Modeling constructs for their representation.
  - Reasoning procedures.
  - Implementation techniques.
- We have presented two new specific structures of knowledge:
  - Generic relationship types.
  - Relationship types with common participants.
- We have presented new (UML-based) methods for their representation, which improve on existing methods.

Thank you

First Conclusion

The represention method of
Generic Relationship Types and their realizations
in the conceptual schema
influences:

- The properties of the resulting schema.
    - Simplicity, definition of new knowledge, ...
- The reasoning about the schema.
- The implementation of the relationship types.