Towards the Universal View: Diamond Path Framework (Diamond of Attention, Diamond of Cognitive Elements)

PA116 – L9

EUROPEAN UNION
esf
MINISTRY OF EDUCATION, YOUTH AND SPORTS
OP Education for Competitivness
UNIVERSITAS MASARYKIANA BRUNENSIS
INVESTMENTS IN EDUCATION DEVELOPMENT

Topics

- Introductory comments
- MENTION – USE principle
- Diamond of attention – structure
- Diamond of attention – operations (behavior)
- Universality and Fixing-point
- Diamond of Cognition
- Duality of these Diamonds
- Diam2 explanation

CMM

1. Accidental
- ad hoc, non-formal

Basic standards

2. Repeatable
- descriptions exists
- experience from the past

Defining of enter-prise processes

3. Defined
- base for process monitoring
- comparing of projects

Measurable goals in terms of process and product quality

4. Managed
- metrics for process and products quality measurement

Permanent process improvement

5. Optimized
- capability of continual process improvement

?

What is our today position ???

What kind of IS is required by specific CMM Level?

- Level 2 = classic IS
- Level 3 = Level 2 + business process modeling in four BPM dimensions
  >> to allow to mention models
- Level 4 = Level 3 + simulations and work-flow, systems monitoring events in classic IS
  >> to allow to use models
- Level 5 = Is there any IS like this today?
  >> ???

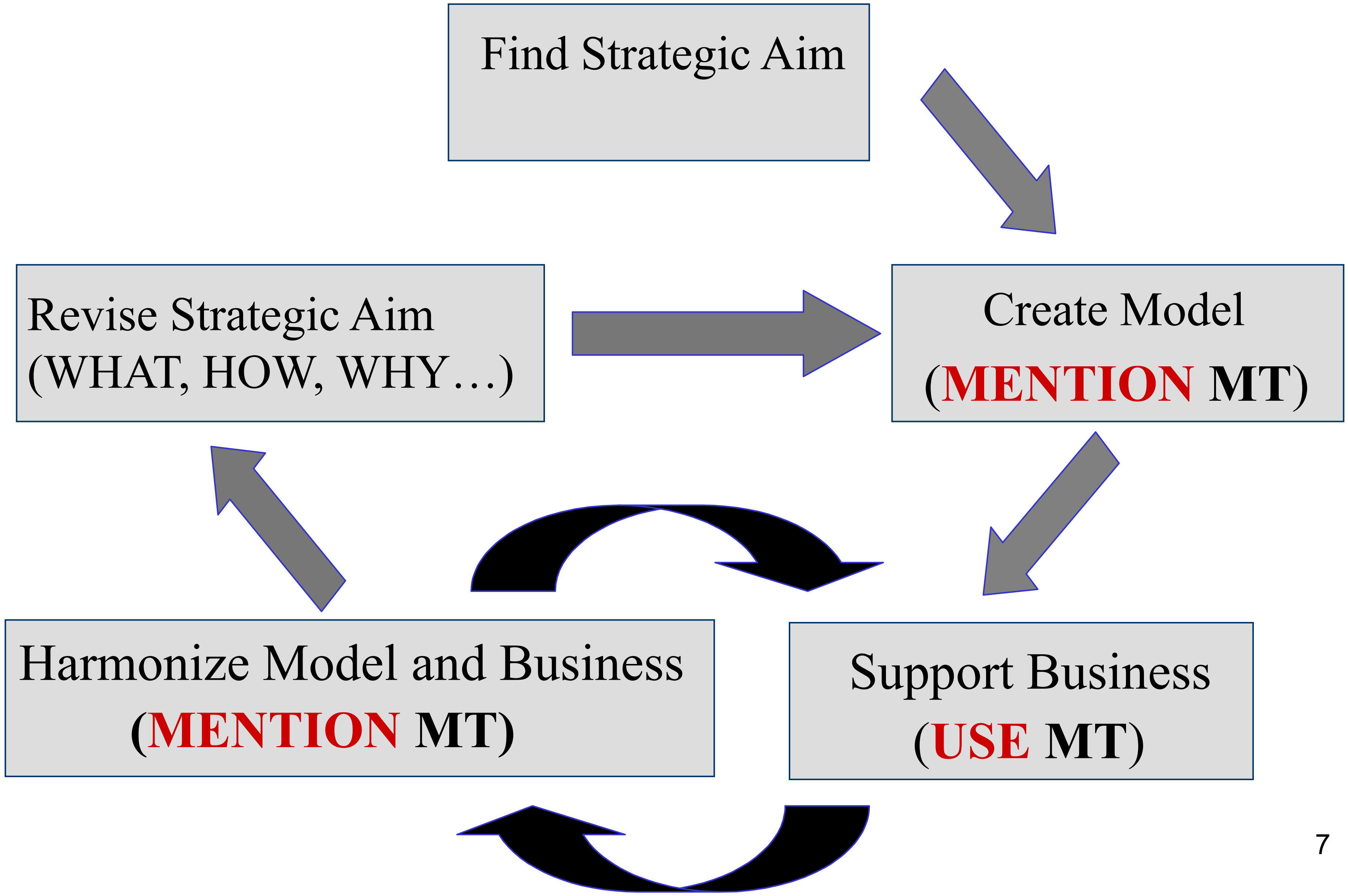
The Unity of Modeled and Modeling World principle

- *ICT tools and system of their deployment in a company/organization must be realized in such a way, that* ***they are able to model*** *present and intended processes in particular contexts, as well as* ***support*** *operativeness of these processes execution,* ***in one uniform environment****.*
- Otherwise it is not possible to reach CMM Level 5.

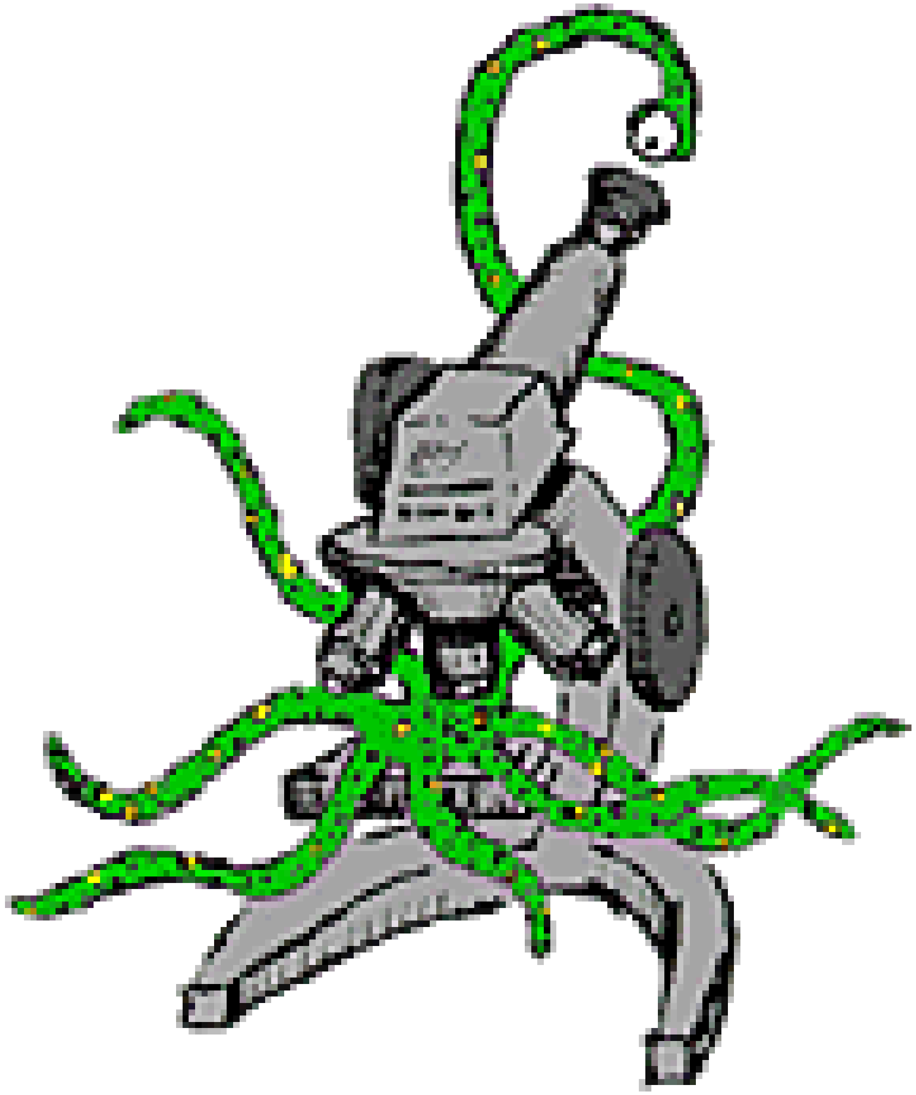
Work with the „unknown“ principle

- Careful examination of CMM Level 5 reveals that it is neccessary to formulate another one „very daring“ principle:
- *SW supporting business has to be designed in a way, that it is able to support even those requirements, that we are not today aware of.*

Review: Cyclical paradigm


Find Strategic Aim
Create Model (MENTION MT)
Support Business (USE MT)
Harmonize Model and Business (MENTION MT)
Revise Strategic Aim (WHAT, HOW, WHY…)

Self-reference !



How to solve it ?

We need two operations:
MENTION
and
USE

MENTION operation

- MENTION operation applied on an object, that we have focused our attention on, allows us to **directly** „speak about“, „describe“ and „elaborate on“ anything that we commonly use, e.g. to model/simulate business processes. MENTION has to enable development and update of anything, that (in *use mode*) supports knowledge acquisition and manipulation. MENTION has to provide possibility to continually develop knowledge about ‘what knowledge we need’ and ‘how to manipulate it’.
- MENTION operation returns focused object in *mention mode*

USE operation

- USE operation applied to an object, that we have focused our attention on, allows us to **directly** use anything that we have designed in *mention mode*. It has to provide possibility to carry out what the object is for.
- USE operation returns focused object in *use mode*

Infinite sequence of metamodels?

- When creating a model, we mention its elements (objects) and its structure. When using a model, we work with it as with a modeling tool.
- In order to create a model, we often ascend to metamodel (modeling tool) level, which allows us to create the model. In order to use a model, we descend to the original level and use the model itself as a modeling tool.
- The problem is in switching between model and metamodel level.
- This produces potentially infinite sequence of metamodels, provided we do not have MENTION and USE operations.

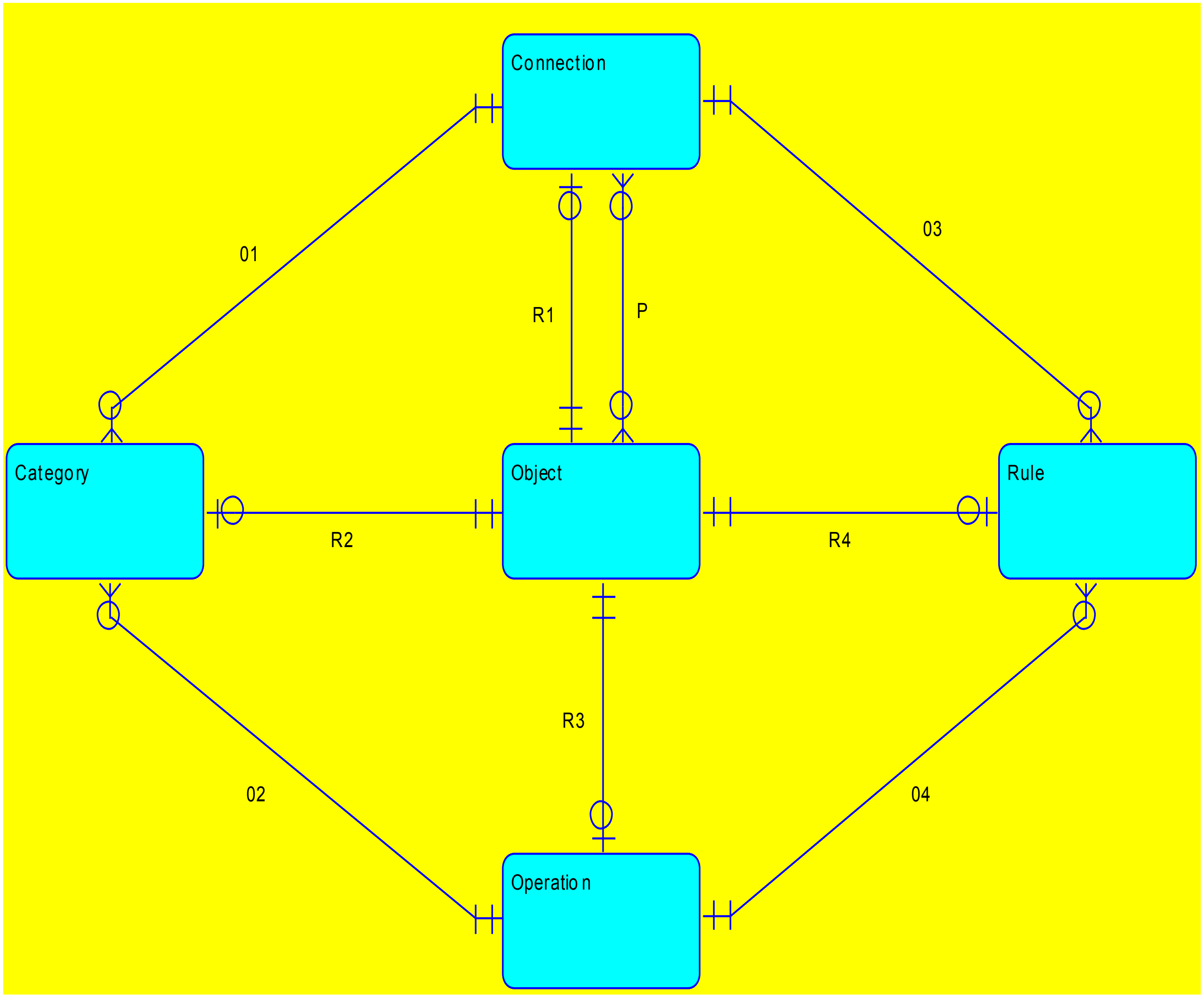
MENTION – USE principle

- *Modeling Tool MT is a universal one if:*
  (1) it has capability to model any model
  (in accordance with modeling tool definition)
  (2) let MT = (CAT, CNN, OPE, RUL), then there exist MENTION and USE operations in OPE, which have previously stated properties.

S

Diamond of Attention Focusing (Diam1)

In accordance with “enactive perception” principle:
a way how “to see” effectively


Connection
01
03
R1
P
Category
Object
Rule
R2
R4
R3
02
04
Operatio n

Container ($\neq$ type)

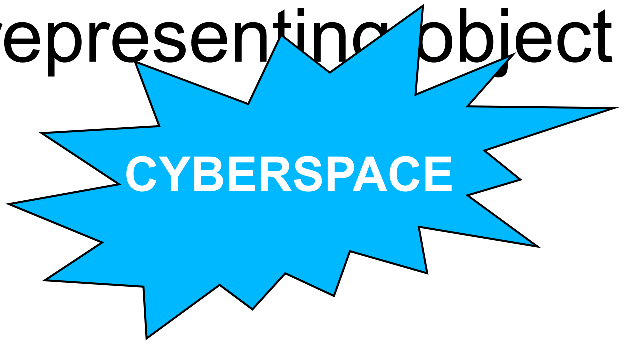
- **Agreement:** ***Any ideational construct that can be used for arrangement of some elements***, where an element can be anything on which we can focus, will be ***called a container. Even an implementation of such a construct in ICT will be considered to be a container.***
- Container may be empty or may contain finite or infinite number of elements.
- Elements can be inserted into the container as well as ***removed from it.***
- When an element is a member of a container, we say the element *dwells* in the container or it is *stored* in the container.

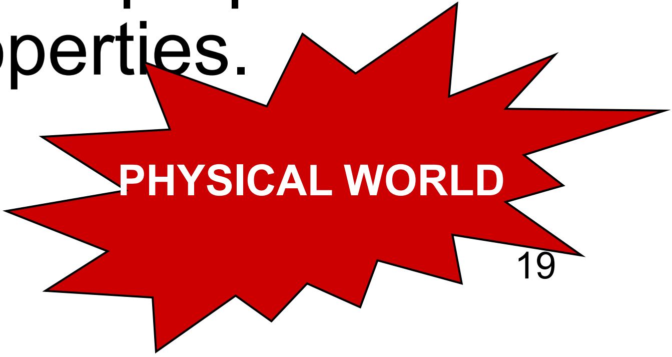
Examples

- Containers are classes of objects, categories, type or kind entities in a database, files, whole databases etc.
- In general, every MT is a container. It is a container for models.
- MT is a container composed of other containers, that are used for storing individual instances according to particular properties. We create models from these instances.

DMT-object

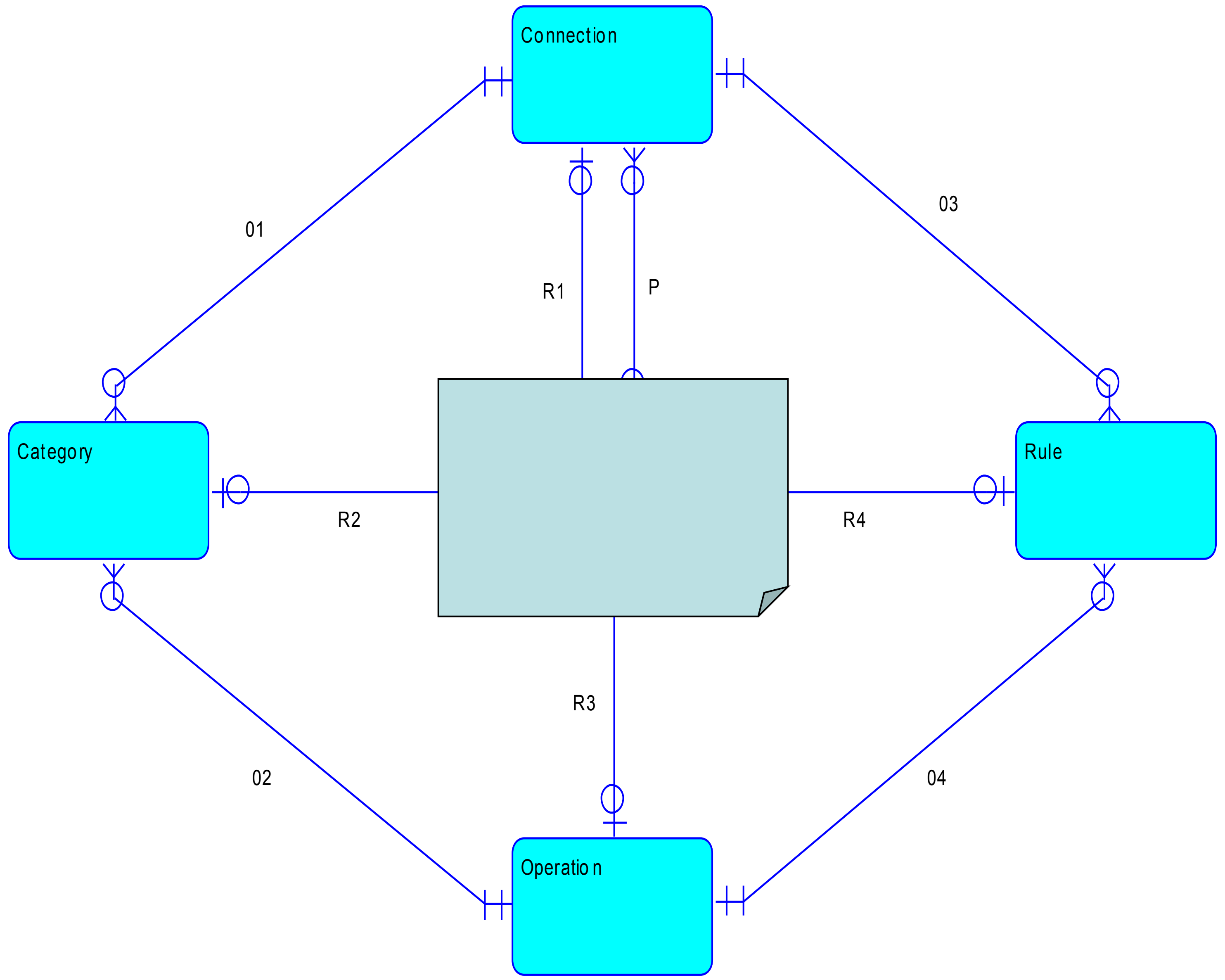
- Every element stored in DMT, that depicts an object (entity) of reality we are getting to know and are able to focus our attention on, will be called *dmt-object*. We say, that - for modeling purposes - dmt-object *represents* object of modeled reality.
- Object of modeled reality, that we focus attention on, will be called *focused object*. We speak about focused (represented) object properties as about its representing object properties.





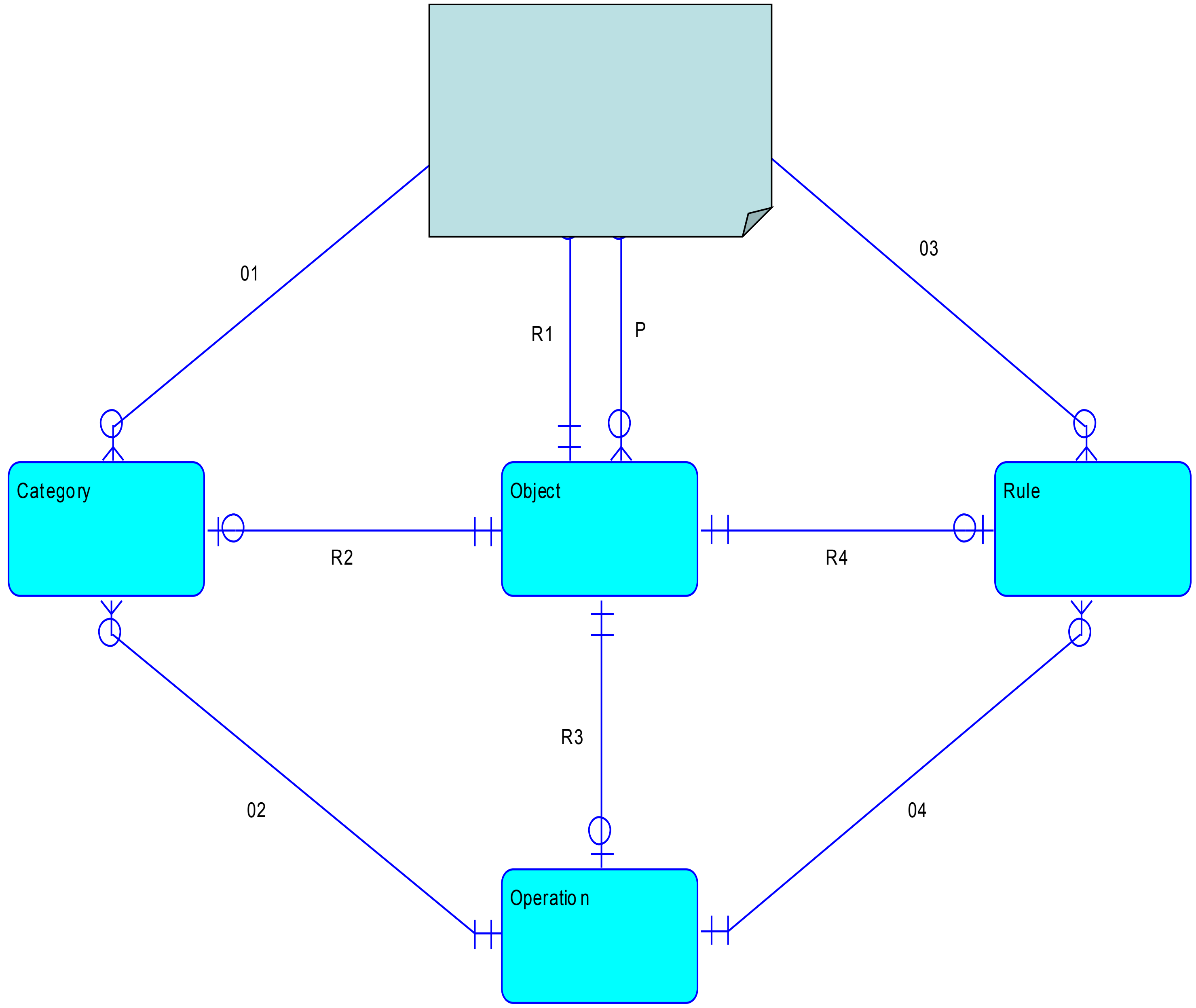
Container (#Object)

- Dmt-objects dwell in container (#Object) in Diamond. Containers used in Diamond will be defined by specification of their member elements.
- Container (#Object) is defined in such a way that it contains all such dmt-objects which can be mentioned in DMT, i.e. which ***can be assigned certain properties.***


Connection
Category
Rule
Operatio n
01
02
03
04
R1
P
R2
R3
R4

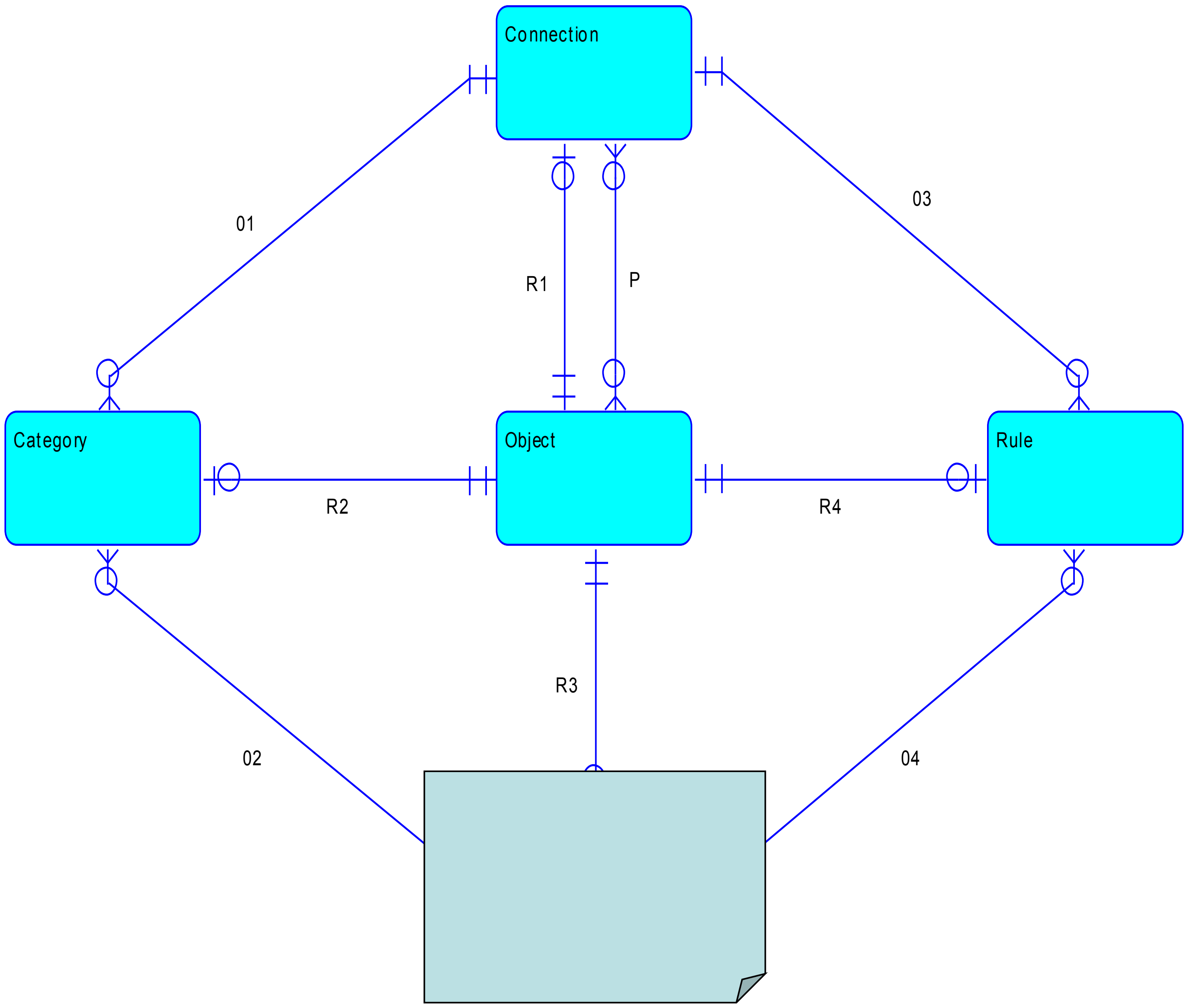
Container (#Connection)

- Container (#Connection) is defined in such a way that every of its elements is a sequence of the length n (n-tuple) of dmt-objects, where n is some finite natural number.
- Every element of the container (#Connection) is called a ***connection*** or a ***sequence.***
- Container (#Connection) contains by definition one special element $\perp$ called improper connection. It is not possible to determine objects constituting improper connection.


Category
Object
Rule
Operatio n
01
02
03
04
R1
P
R2
R3
R4

Container (#Operation)

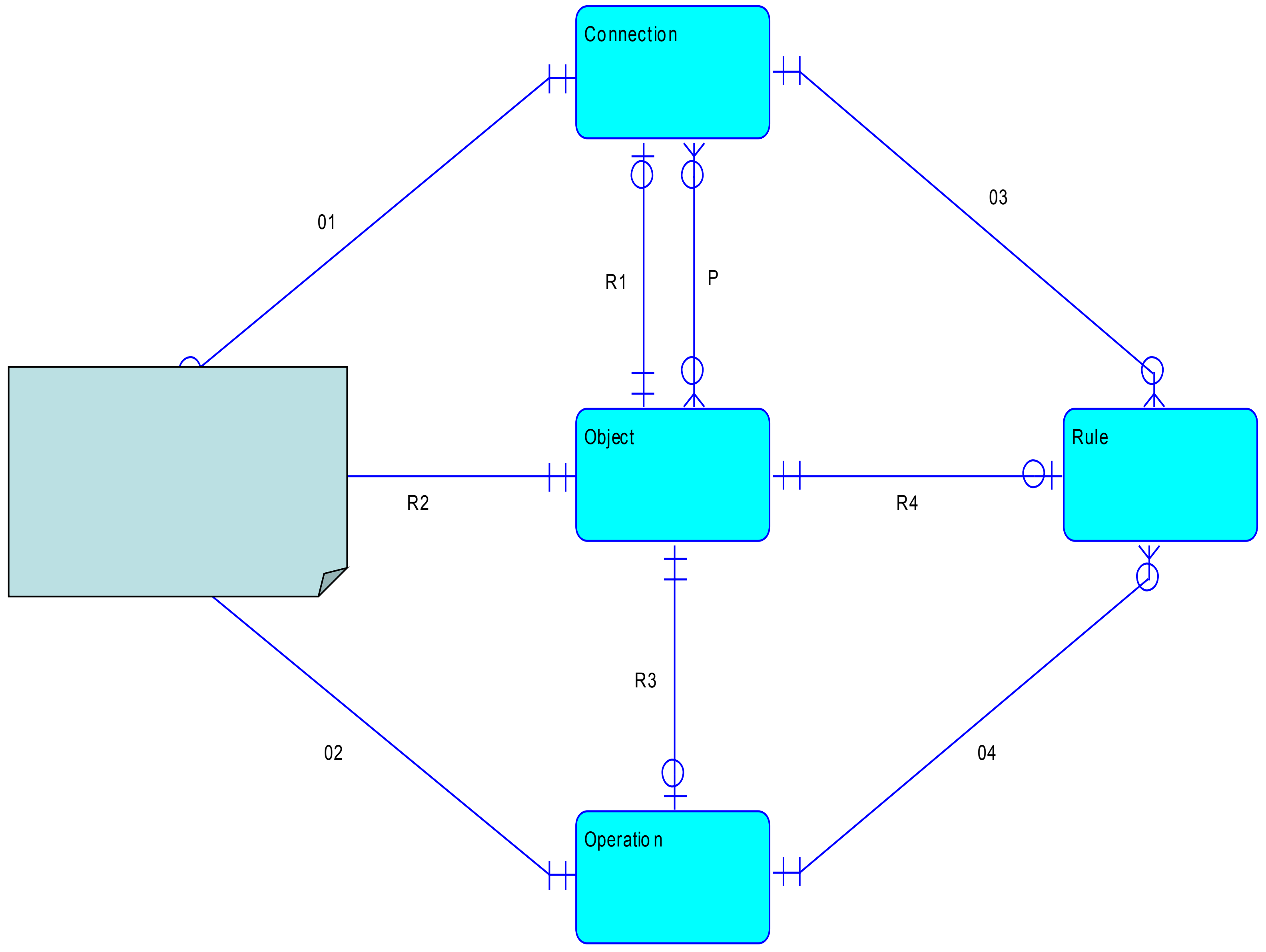
- Container (#Operation) is defined in such a way that every of its elements is an algorithmically computable transformation of one state of DMT to, generally, another state of DMT.
- By the state of DMT we mean one particular filling of DMT (as a container) by elements-instances, that may dwell in DMT.
- We assume there always are two (boolean) objects in DMT, one of them codes True, the other False.
- The elements of the container (#Operation) are called ***operations***.
- Some operations may *fail*, i.e. return object coding False, under certain circumstances and *succeed under different circumstances*. Some operations always succeed.


Connection
01
R1
P
03
Category
Object
Rule
R2
R4
R3
02
04

Container (#Category)

- Container (#Category) is defined in such a way that every of its elements has the following properties:
  1. it is a container for dmt-objects,
  2. it is one-to-one mapped to the pair <Cn, Op>, where

     Cn $\in$ (#Connection), Op $\in$ (#Operation), and
  3. it holds about the operation Op that by means of the connection Cn it can recognize whether a given object is or is not in this container.
- The elements of the container (#Category) are called ***categories***.
- The connection Cn is called a *defining connection* of this category, the operation Op is called a *defining operation* or an *evaluator* of this category.


Connection
01
03
R1
P
Object
Rule
R2
R4
R3
02
04
Operatio n

Observation 1

- Category definition allows connection Cn to be in some cases improper (i.e. Cn = $\perp$, which means, that list of connected objects is empty in all states of world). Then the determination of category members is based on evaluator application only.
- On the contrary, evaluator must not be improper under any circumstances. Otherwise it would be impossible to determine which objects are members of the category.

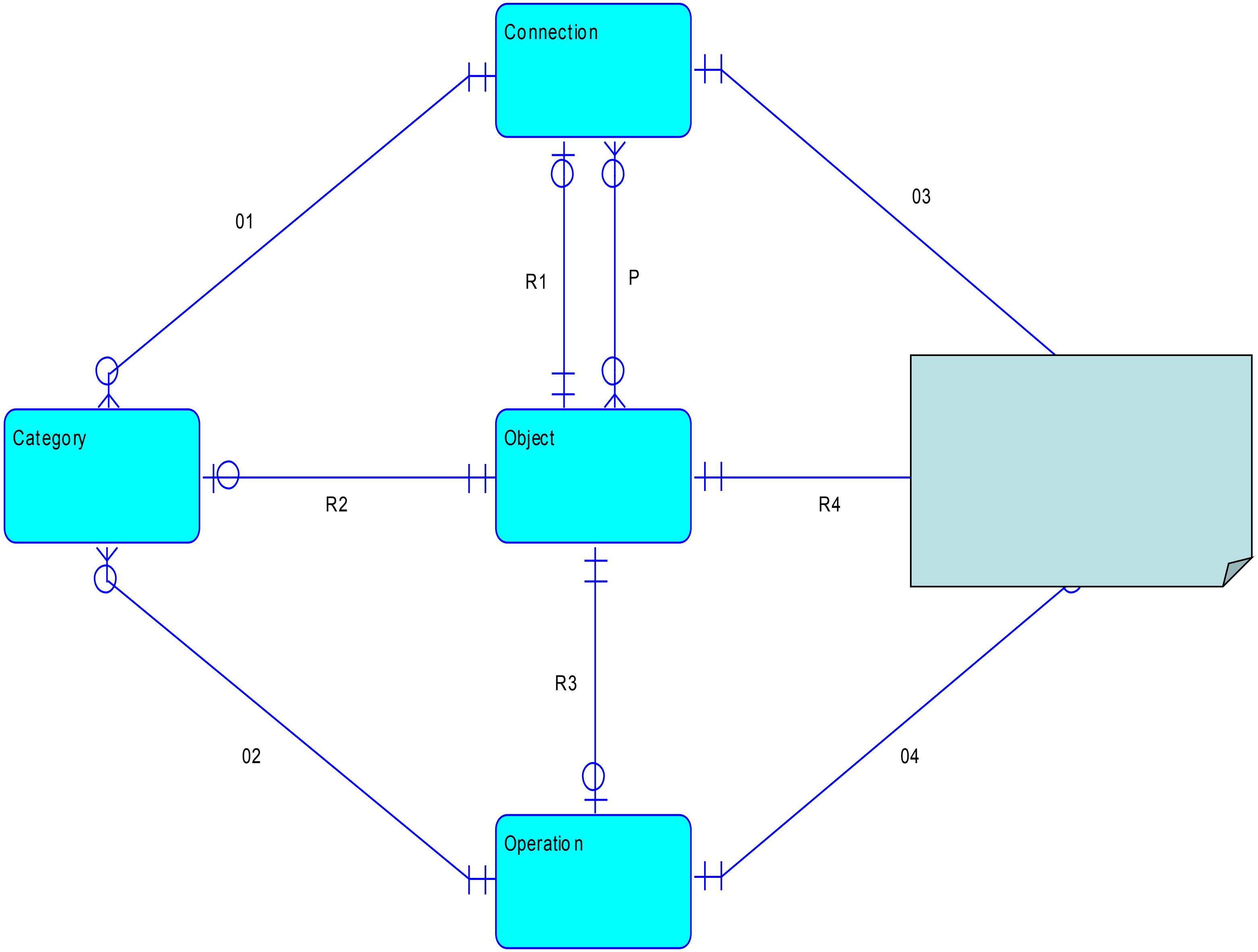
Observation 2

- Fundamental fact about this definition of category is, that categories are not some primitive elements created outside the scope of using DMT, but we have the categories creation and categorization process at our disposal directly in DMT.
- This fact makes DMT considerably different from most of other tools, that treat categories as already complete things. Such tools do not deal with categories creation, they allow only to manipulate them.

Container (#Rule)

- Container (#Rule) is defined in such a way that every of its elements has the following properties:
  1. it is a dmt-object,
  2. it is one-to-one mapped to the pair <Cn, Op>, where

     Cn $\in$ (#Connection), Op $\in$ (#Operation), and
  3. operation Op, by means of the connection Cn, carries out a test whether the rule is valid, i.e. the operation returns the value True if the test is *successful*, and value False in the *opposite case.*
- The elements of the container (#Rule) are called ***rules.***
- The connection Cn is called a *specifying connection* of this rule. The operation Op is called a *specifying* (or *testing*) *operation* of this rule.


Connection
Category
Object
Operatio n
01
02
03
04
R1
P
R2
R3
R4

Observation

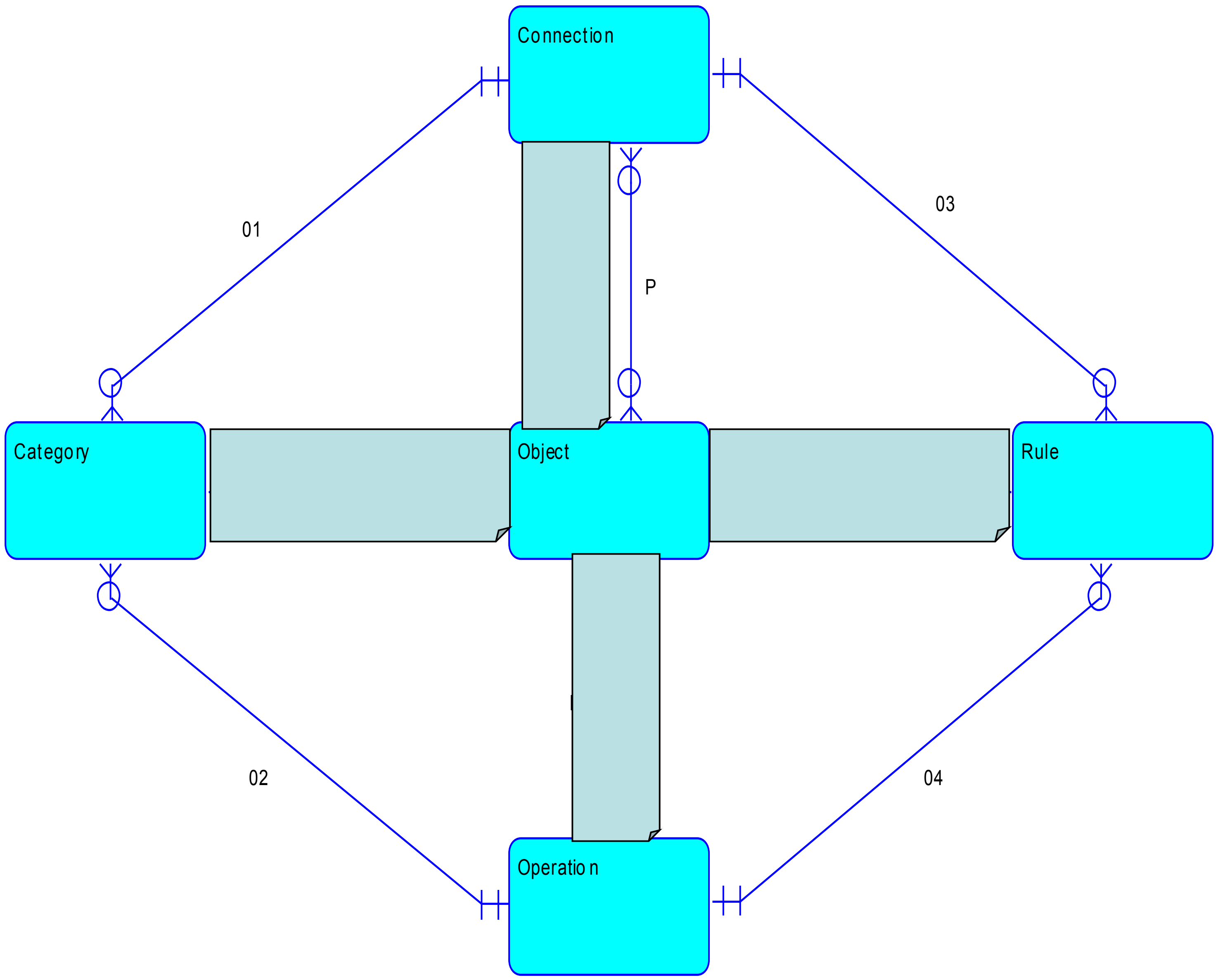
- Rule definition allows again connection Cn to be improper ($\perp$) in some cases. Then the determination of rule satisfaction is based only on specifying operation application. However, specifying operation must not be improper under any circumstances. Otherwise it would be impossible to carry out test defined by the rule.

Diamond semantics

- **Objects established by preceding definitions form the semantics of DMT modeling language. Universal modeling does not require anything more.**
- *Every connection, operation, category and rule is contained in container (#Object).*

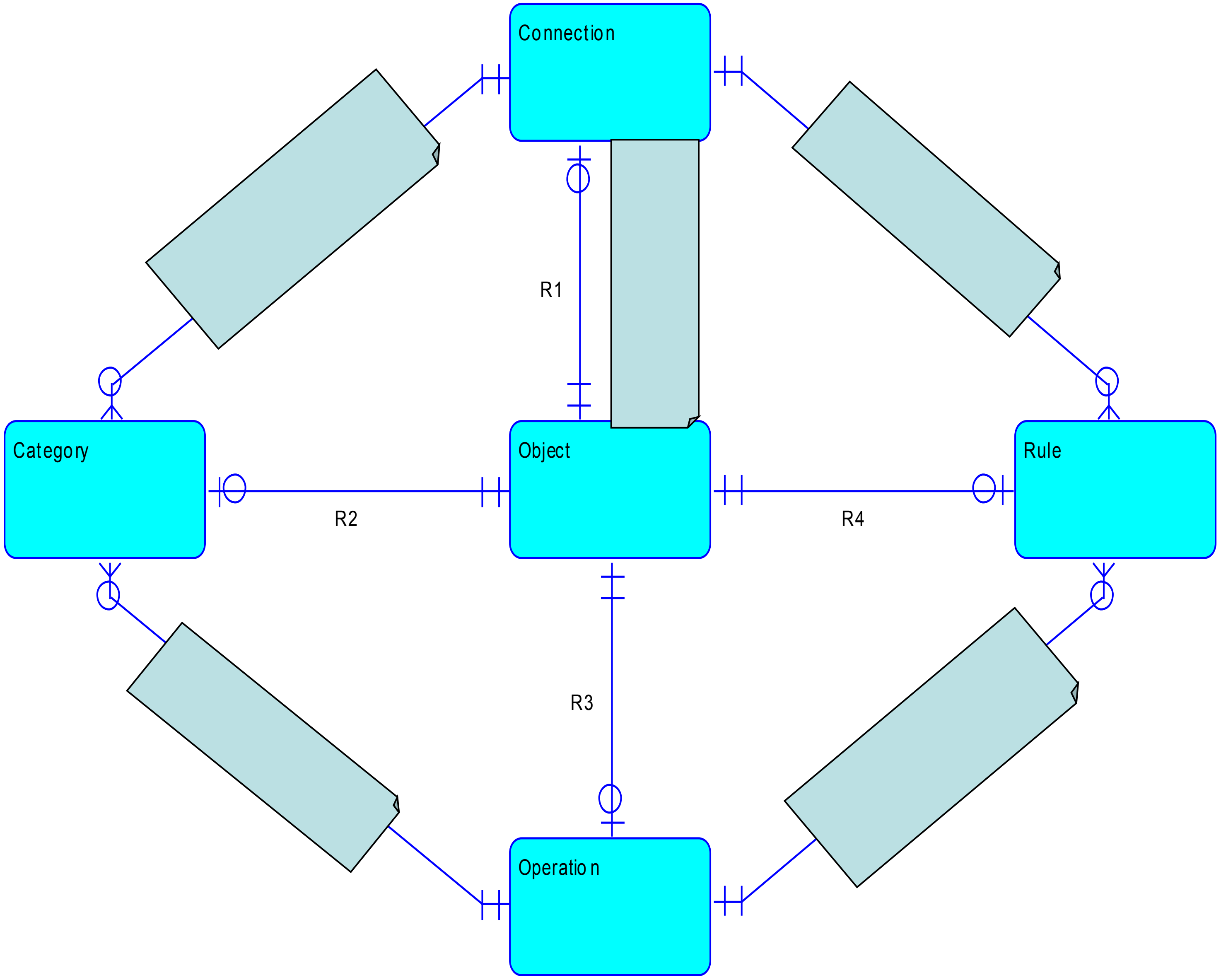
Universality Principle Implementation

- Edges R1, R2, R3 and R4 are together called R-edges. R-edges serve to perform transitions between mentioning and using and vice-versa.
- If we focus on a concrete object of the class #Object, which was spoken of in a way (MENTION), then by the operation USE we will pass the relevant R-edge and we will reach represented sequence (#Connection) or operation (#Operation) or category (#Category) or rule (#Rule), which could be then directly used, or we will reach nothing.
- If we focus any vertex of the Diamond graph on the other hand (which we used in a way - USE), then by the operation MENTION we will pass the relevant R-edge and we will reach this vertex representing object, which we can speak of directly.


Connection
01
03
P
Category
Object
Rule
02
04
Operatio n

Perimeter edges and p-edge

- Edges 01 and 02 from the figure connect category to this category *defining connection* and *defining operation*, respectively.
- Edges 03 and 04 connect rule to this rule *specifying connection* and *testing operation*, respectively.
- ‘Specifying connection’ of a rule can be an object representing the whole model described by this (or related to) the rule.
- Last, p-edge (projection) connects connection (sequence) to this sequence creating particular objects.


Connection
Category
Object
Rule
Operatio n
R1
R2
R3
R4

Basic Operations Outline

- Operations of creation, deletion, actualization, access and detection of dmt-objects
  - Create, Delete
  - Write, Read
  - Obtain
  - Label (detection what an object is about)
- Operations of „travelling“ on the Diamond perimeter
  - Get (Con, Ope, Cat, Rul)
- Operations with p-edge

Operation MENTION

- If the operation is applied to the centre of the Diamond, i.e. to an instance of dmt-entity (#Object), it will return this object.
- If the operation is applied to any vertex of the Diamond, it will pass relevant R-edge and will return object, which represents this Diamond vertex (just for the purpose of mentioning).
- This operation never fails. That is to say ‘mentioned can be anything’.

Operation USE

- **use_obj(+*Idobj*)**
  Returns instance of this vertex of the Diamond, which is represented by this given object, i.e. it pass from the Diamond centre to the relevant vertex. Operation fails, if the given object represents nothing.
- **use_con(+*Idcon*)**
  Returns a list of objects forming the given connection
- **use_ope(+*Idope*,+*Arglist*)**
  Executes given operation (*Idope*) and as arguments it submits the content of the list (*Arglist*).

Operation USE - continued

- **use_cat(+*Idcat*,+*Arglist*)**
  Executes operation which is connected to the given category by the edge 02. This operation obtains as arguments the category (*Idcat*), the connection connected to this category by the edge 01 and arguments from the list *Arglist.*
- **use_rul(+*Idrul*,+*Arglist*)**
  Executes operation which is connected to the given rule by the edge 04. This operation obtains as arguments the rule (*Idrul*), the connection connected to this rule by the edge 03 and arguments from the list *Arglist.*

Universality Statement

- *Let DMT = (OBJ^D, CAT^D, CNN^D, OPE^D, RUL^D), where OBJ^D, CAT^D, CNN^D, OPE^D and RUL^D is defined as described above.*
- *Let MT = (CAT, CNN, OPE, RUL) is arbitrary modeling tool.*
- *Then MT $\angle_{mc}$ DMT.*
- As DMT contains operations MENTION and USE, this is enough for its universality according to the MENTION – USE principle.
- Note: *CNN^D* is container for finite sequences; *CNN* is container for connection categories.

MENTION – USE principle or Universality Principle (reminder)

- *Modeling Tool MT is a universal one if:*
  (1) it has capability to model any model
  (in accordance with modeling tool definition)
  (2) let MT = (CAT, CNN, OPE, RUL), then there exist MENTION and USE operations in OPE, which have previously stated properties.

Univ. Statement: principle of the proof - 1

- The proof is done in accordance with MT definition and the definition of ‘model constructed by MT’.
- First, let’s assume that MT can be constructed by DMT. Then, if M is arbitrary model constructed by MT, it could be constructed by DMT, owing to operation USE: anyway M is constructed from elements of categories from CAT ∪ CNN with help of operations from OPE. But the above assumption yields: each category from CAT ∪ CNN (and each element of this category) could be constructed by DMT. As well each operation from OPE could be expressed using DMT (there is a complete programming language at our disposal). Applying relevant operation USE to this way constructed categories from CAT ∪ CNN and operations from OPE we construct the model M by using DMT.
- Hence, it is enough to prove that really MT could be constructed by DMT. The proof is a constructive one:

Univ. Statement: principle of the proof - 2

- We insert an instance into the container (#Category) in DMT for each category from CAT $\cup$ CNN of MT by means of relevant operation CREATE (**create_cat**). Each fact that a category from CAT is bound in a particular connection from CNN of MT, we express in DMT by inserting one separate instance of (#Connection) (using the operation **create_con**).
- Each operation from OPE of MT we insert into DMT as an instance of (#Operation) (by means of operation **create_ope**).
- Each rule from RUL of MT we insert into DMT as an instance of (#Rule) (by means of operation **create_rul**).
- All such inserted operations and rules could be executed owing to the operation USE.
- The categories, created in DMT according to patterns from CAT $\cup$ CNN, could be filled up by their respective instances owing to operation USE, again. Using the special connections inserted by means of the operation **create_con** we can simulate in DMT “the game” originally played in MT.
- Thus MT is a model constructed by DMT in accordance with the definition.

COROLLARY:

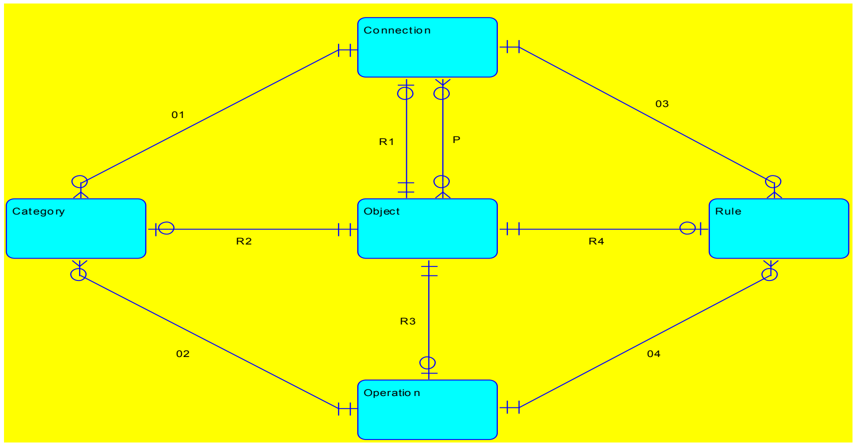
- *Diamond (DMT), as it is a working modeling tool, can be, as a whole, constructed by the Diamond (DMT).*
- To construct the Diamond by the Diamond is not an autotelic plaything.
- If we want really, and without exceptions, to mention anything what we use somewhere, it is necessary to be able to mention Diamond elements themselves, as we use them to model something.
- This means: self-reference is a consequence of universality

How to start to work with universal modeling tool

- One day we need to start to work with a new created modeling tool.
- But, for this purpose the tool has to be prepared for using.
- If DMT is empty, it will be, evidently impossible to start in MENTION mode, i.e. to mention its elements, e.g.. There is nothing to mention at all in this case.
- Thus, the work with Diamond can start in USE mode, only.
- And not to have problems in the future (problems of a kind that something is not possible to execute) it is expedient “to anchor” the tool along with its first using.
- This “anchoring” could be done right by the construction of Diamond by this Diamond itself.

The Primitive Fixing-point

- To create the *Primitive fixing-point* means:
  - Containers (#Object), (#Connection), (#Operation), (#Category) and (#Rule) of the Diamond are constructed by the Diamond
  - Container (#Operation) is filled up by basic operations (see figure in the right column) and by operations MENTION and USE



- Operations of creation, deletion, …
  - Create, Delete
  - Write, Read
  - Obtain
  - Label (detection what an object is about)
- Operations of „travelling“ on the Diamond perimeter
  - Get (Con, Ope, Cat, Rul)
- Operations with p-edge

Pragmatic approach

- Often we start to work with Diamond in a situation in which we want to understand a selected domain and to model it for the purpose to develop some Service System which will be able to operate.
- Such a domain could be: a business domain, scientific domain, a domain of learning of something, etc.
- It is necessary to take to account that such particular domains will be integrated somewhere in the future (this is the current state of world).
- Thus, it is better to prepare play-ground (often used and general categories and connections, often used and general operations and rules) along with the starting event with the Diamond, instead of the pure Primitive fixing-point.
- Hence, the starting construction of Diamond's elements by the Diamond is straight extended to the construction of the appropriate "play-ground" enabling information and knowledge synthesis.

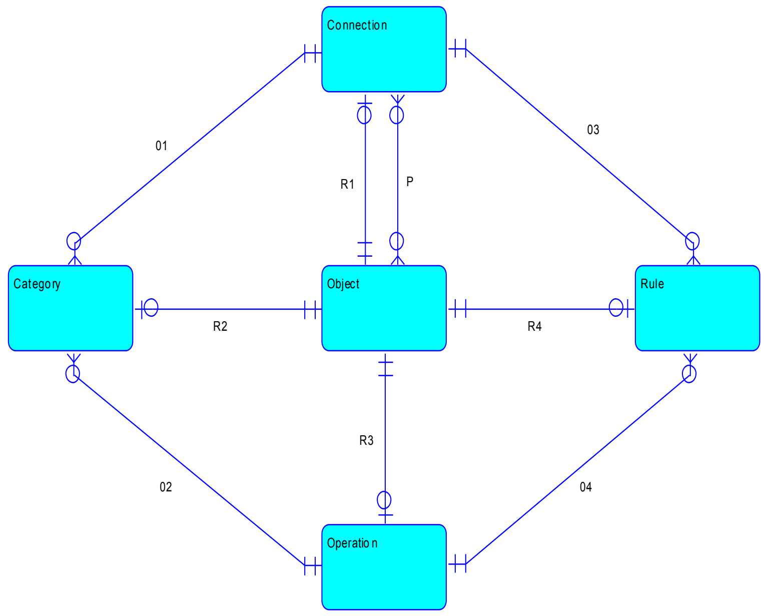
Fixing-point

- By the term *Fixing-point* we denote the very first construction of all instances of dmt-entities which will be useful in course of working with Diamond, i.e., which will be used or mentioned in some super-domain of the domain under discussion.
- Fixing-point is a pragmatic extension of the Primitive fixing-point.

S

This was the way how “to see” effectively in any domain using Universal Modeling Principle

Diamond of Attention Focusing (Diam1)



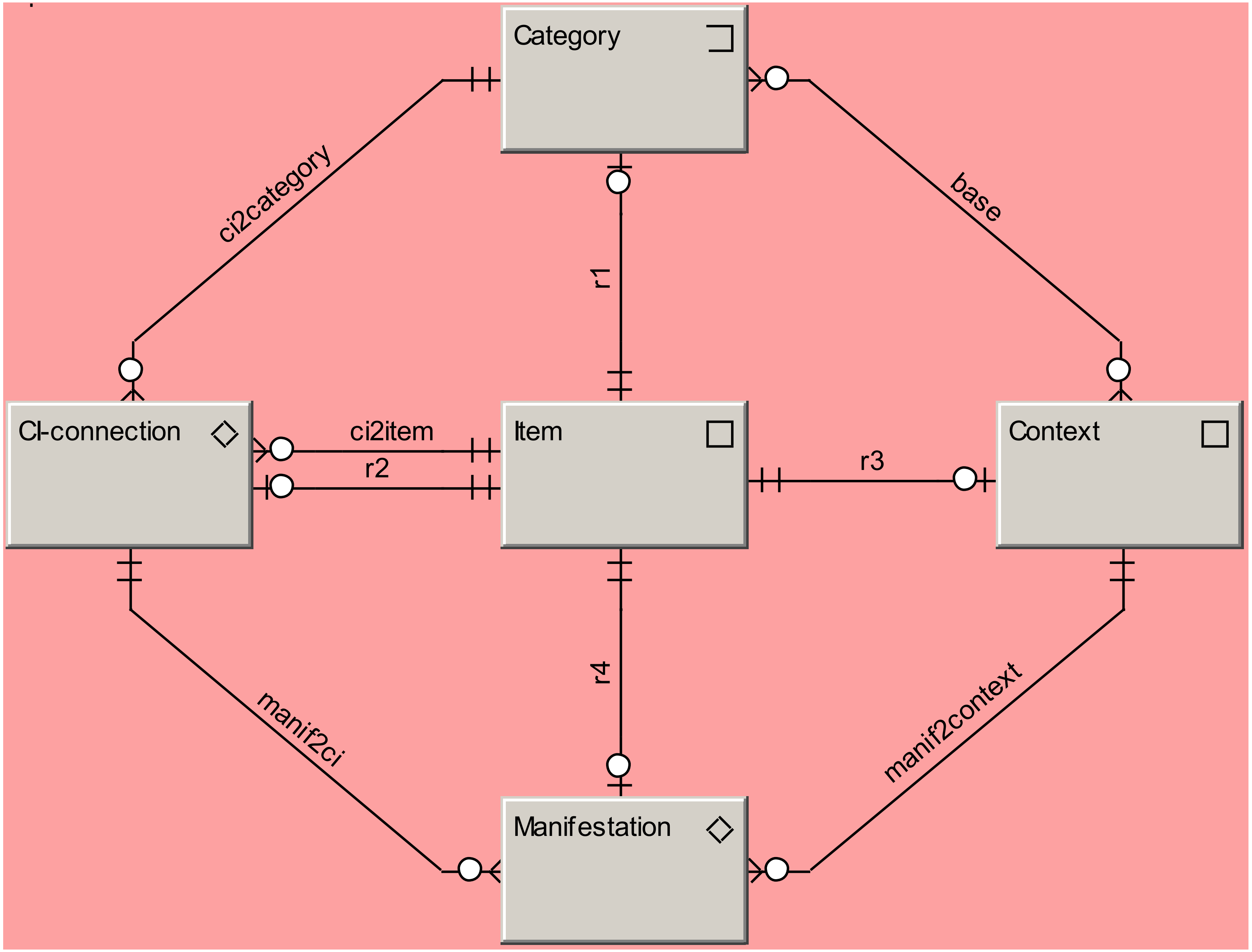
R

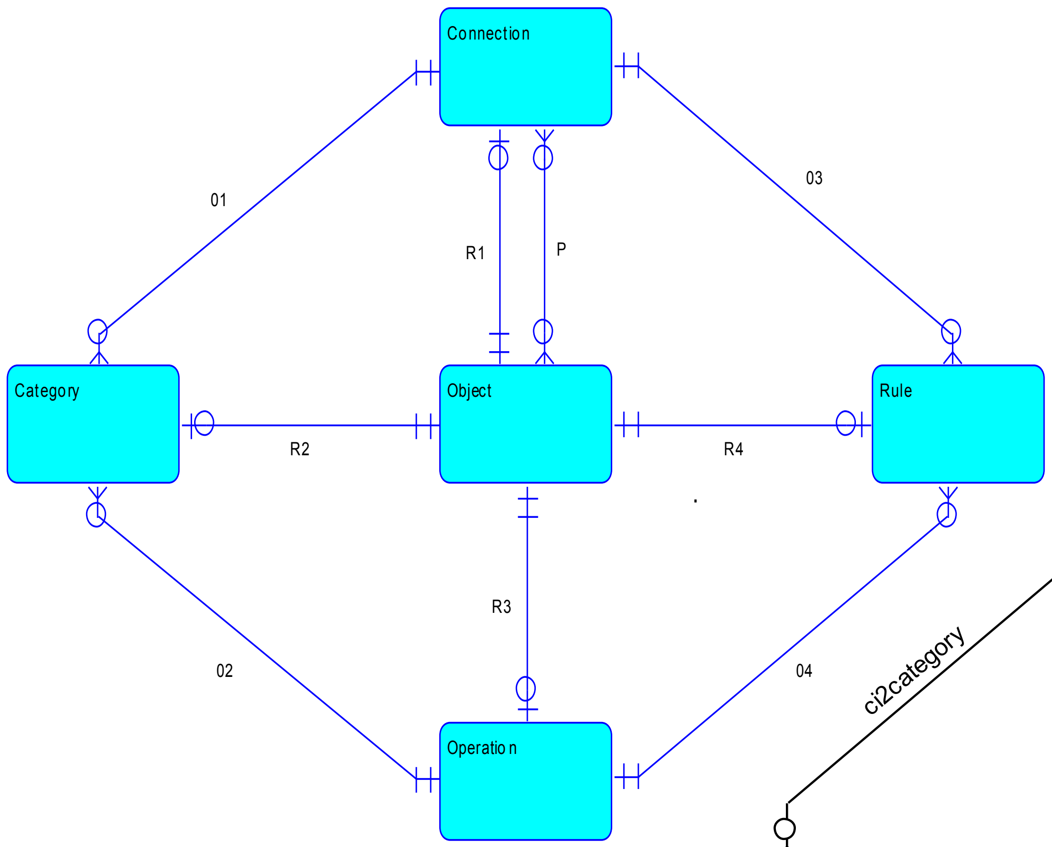
Diamond of Cognition (Diam2)

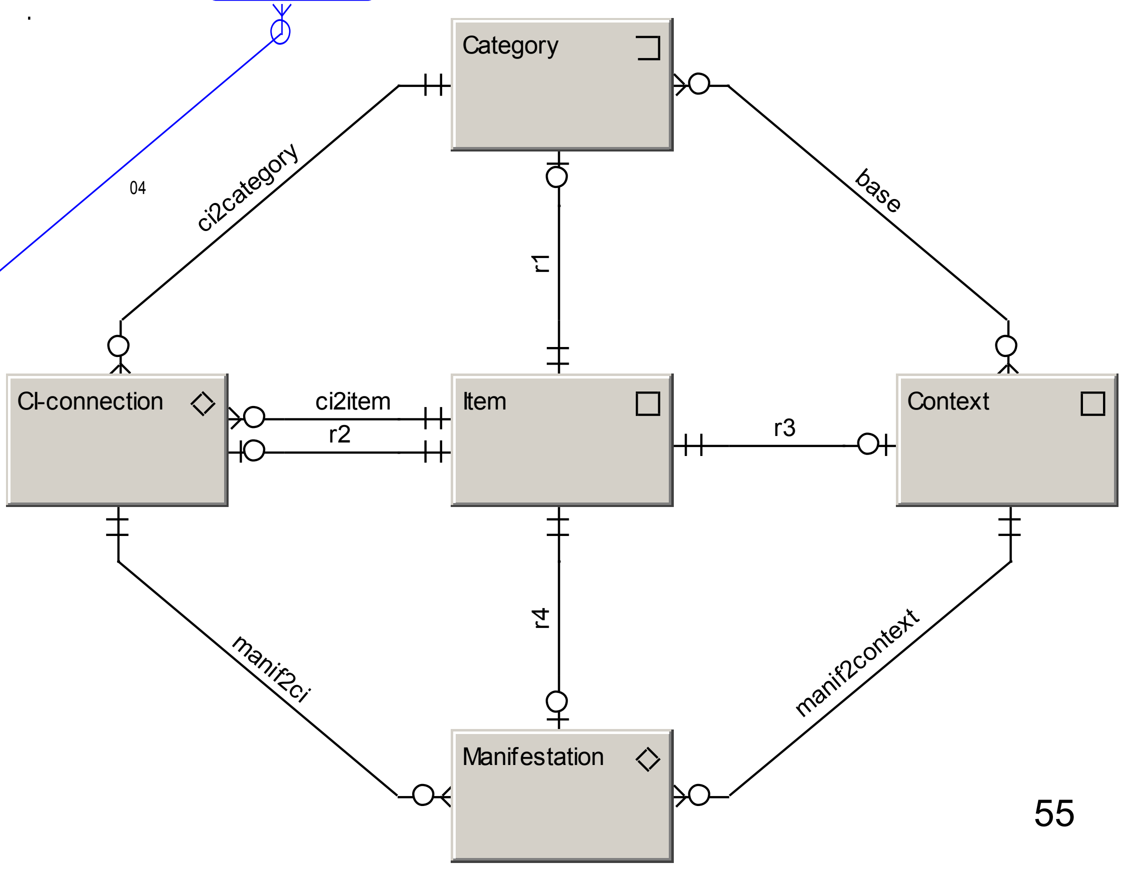
In accordance with “enactive perception” principle:
a way how “to recognize” effectively

Diamond of Cognition

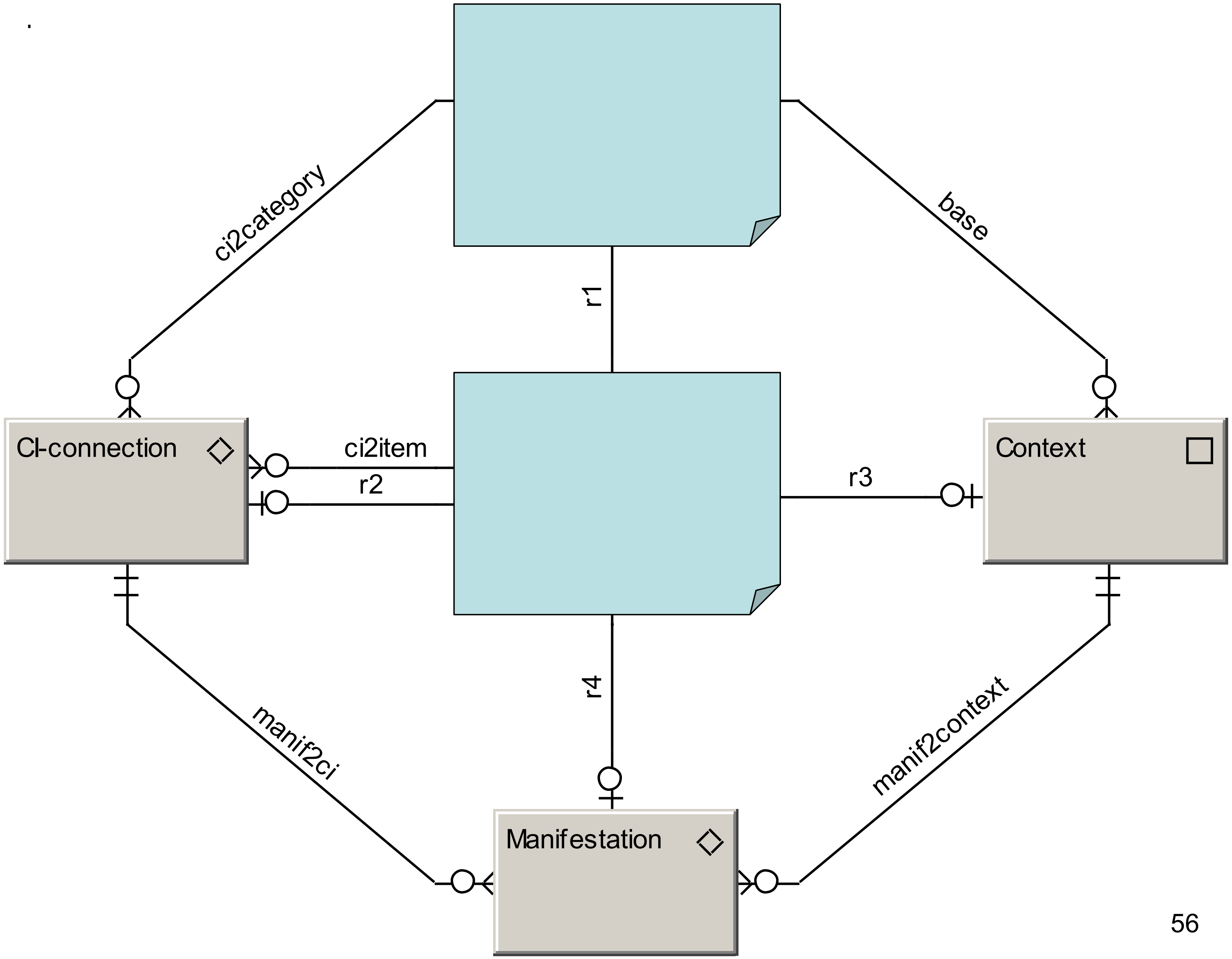
- Focus on Category
- Explicit work with Attention
- Explicit work with measure of certainty
- CI-connections manifested in a given context are “beliefs”.
- Beliefs are building blocks of our understanding of the World.

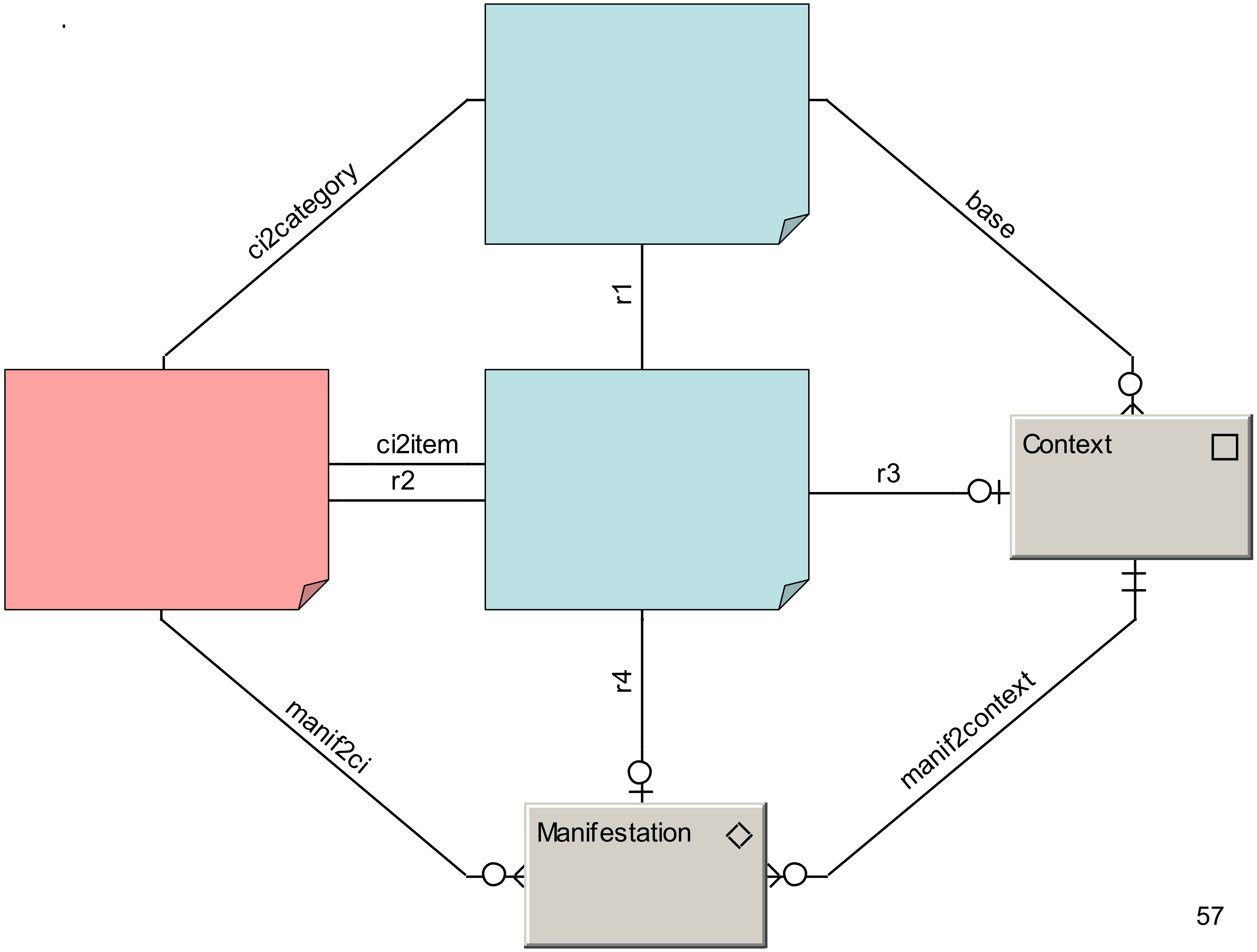

Category
ci2category
base
r1
CI-connection
ci2item
r2
Item
r3
Context
r4
manif2ci
manif2context
Manifestation


Connection
Category
Object
Rule
Operation
01
02
03
04
R1
P
R2
R3
R4

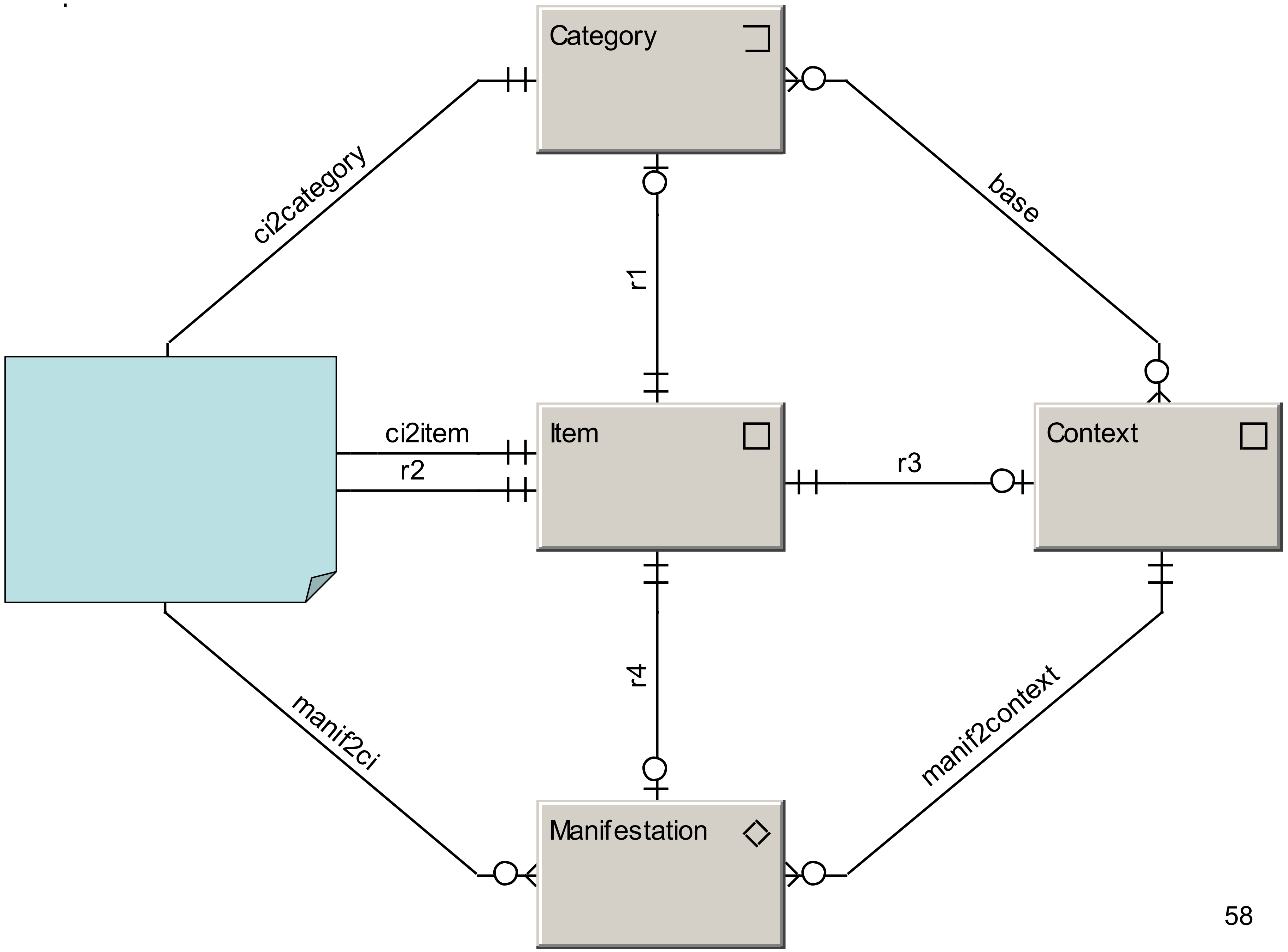

Category
CI-connection
Item
Context
Manifestation
ci2category
base
r1
ci2item
r2
r3
r4
manif2ci
manif2context

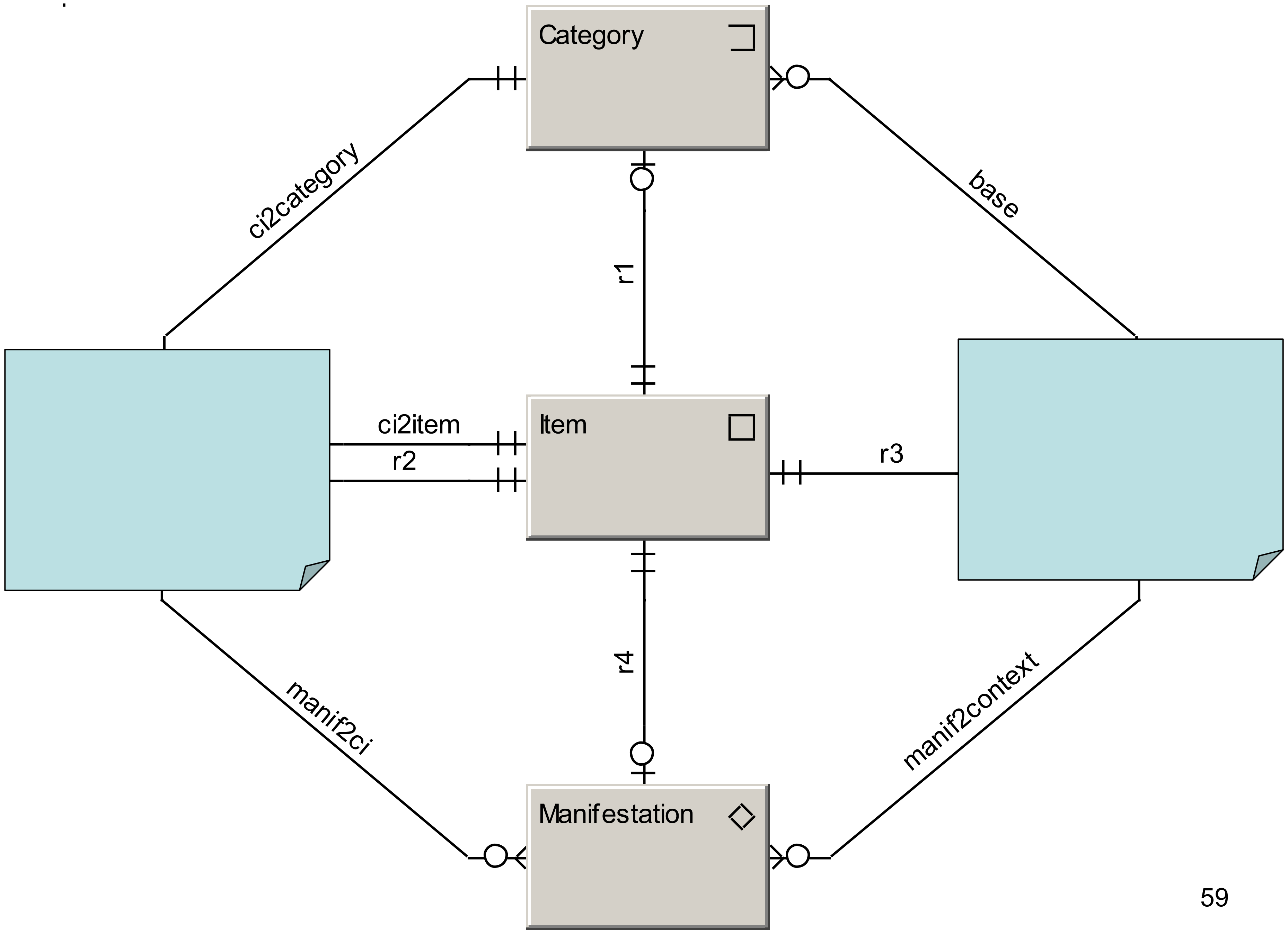
.

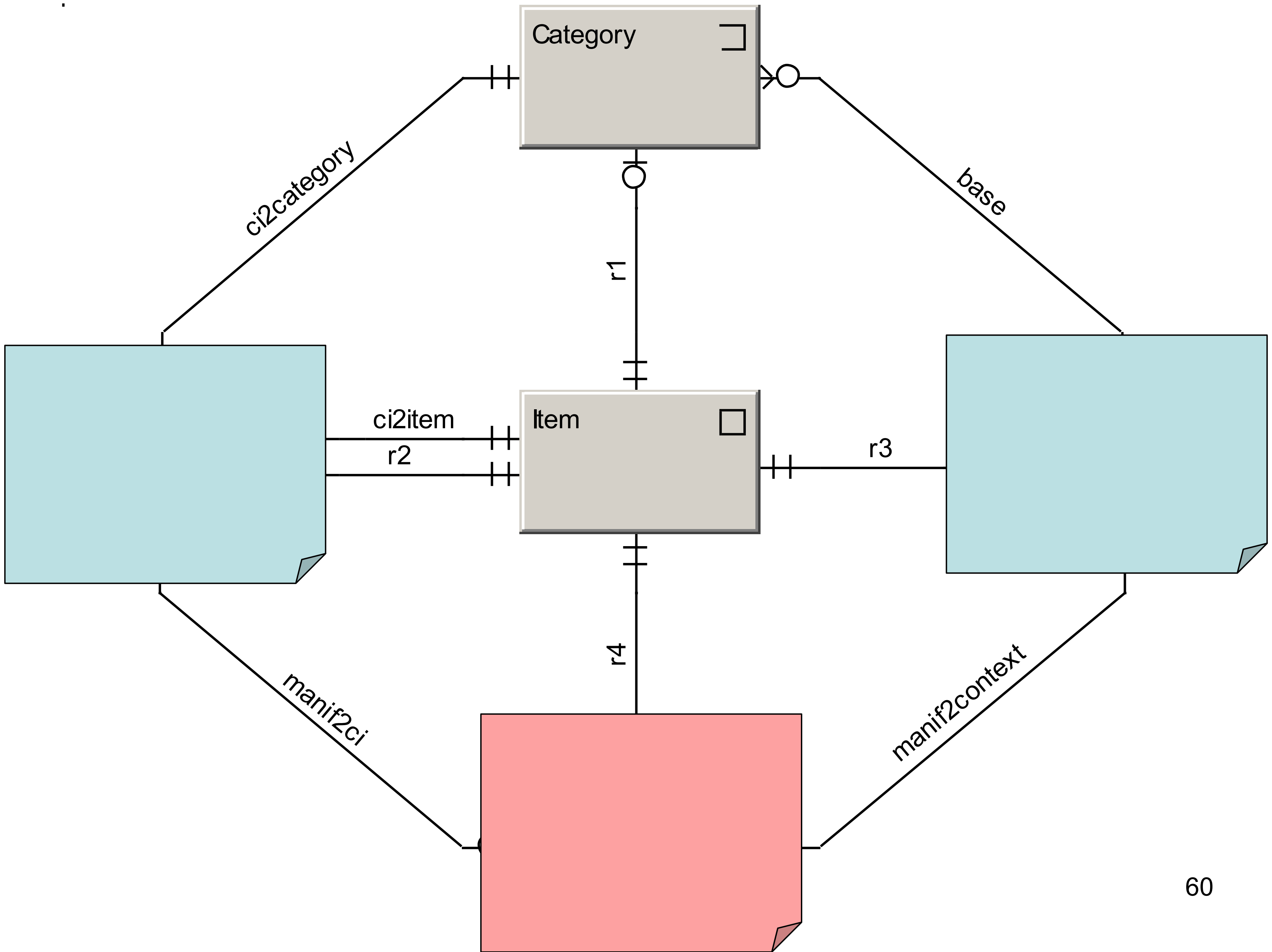

ci2category
base
r1
CI-connection
ci2item
r2
r3
Context
r4
manif2ci
manif2context
Manifestation



ci2category
base
r1
Context
ci2item
r2
r3
r4
manif2ci
manif2context
Manifestation


Category
Item
Context
Manifestation
ci2category
base
r1
ci2item
r2
r3
r4
manif2ci
manif2context


Category
Item
Manifestation
ci2category
base
r1
ci2item
r2
r3
r4
manif2ci
manif2context


Category
ci2category
base
r1
Item
ci2item
r2
r3
r4
manif2ci
manif2context

The “T”

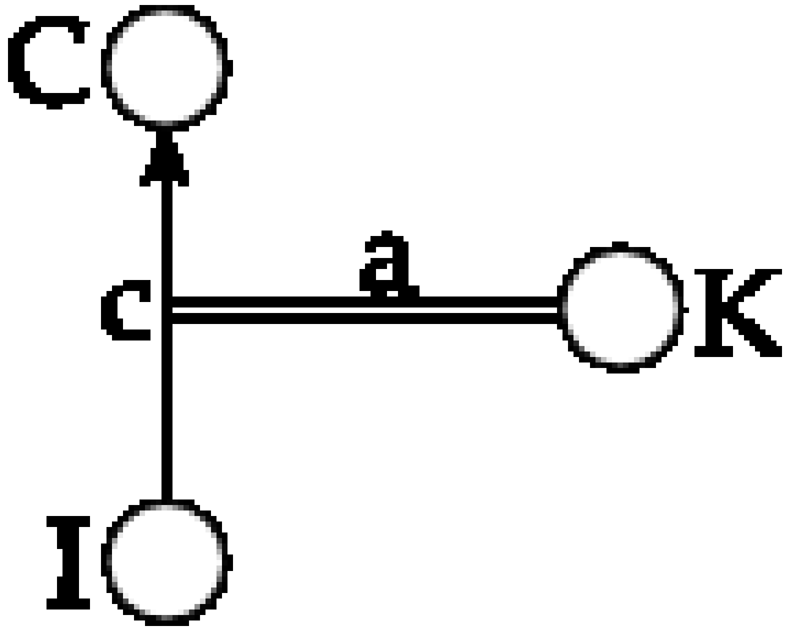


Fig. 1: T (elementary belief)

Context

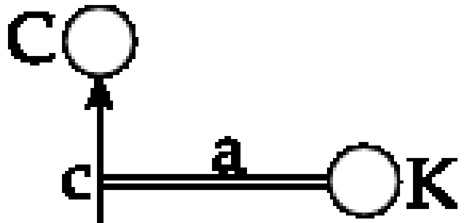
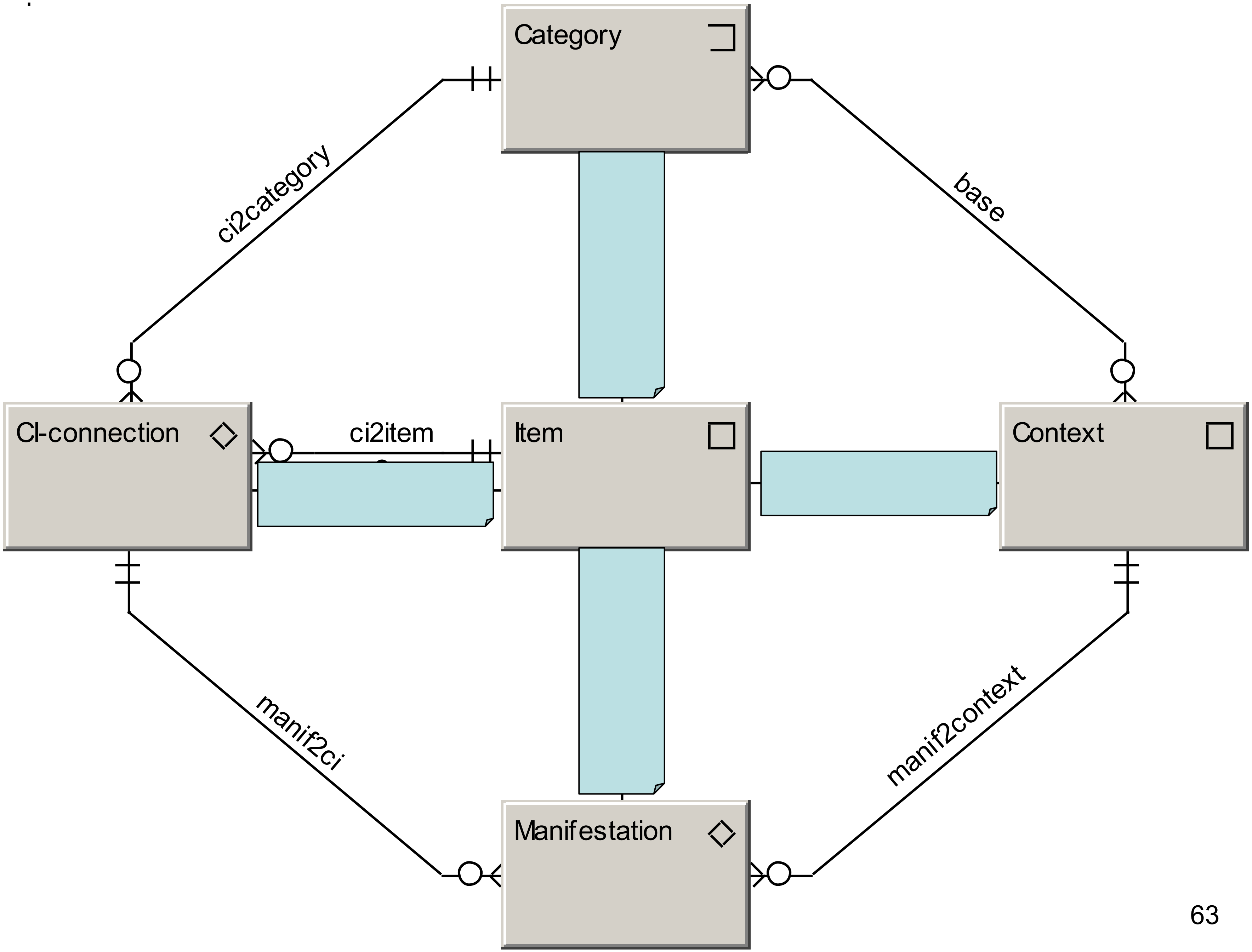
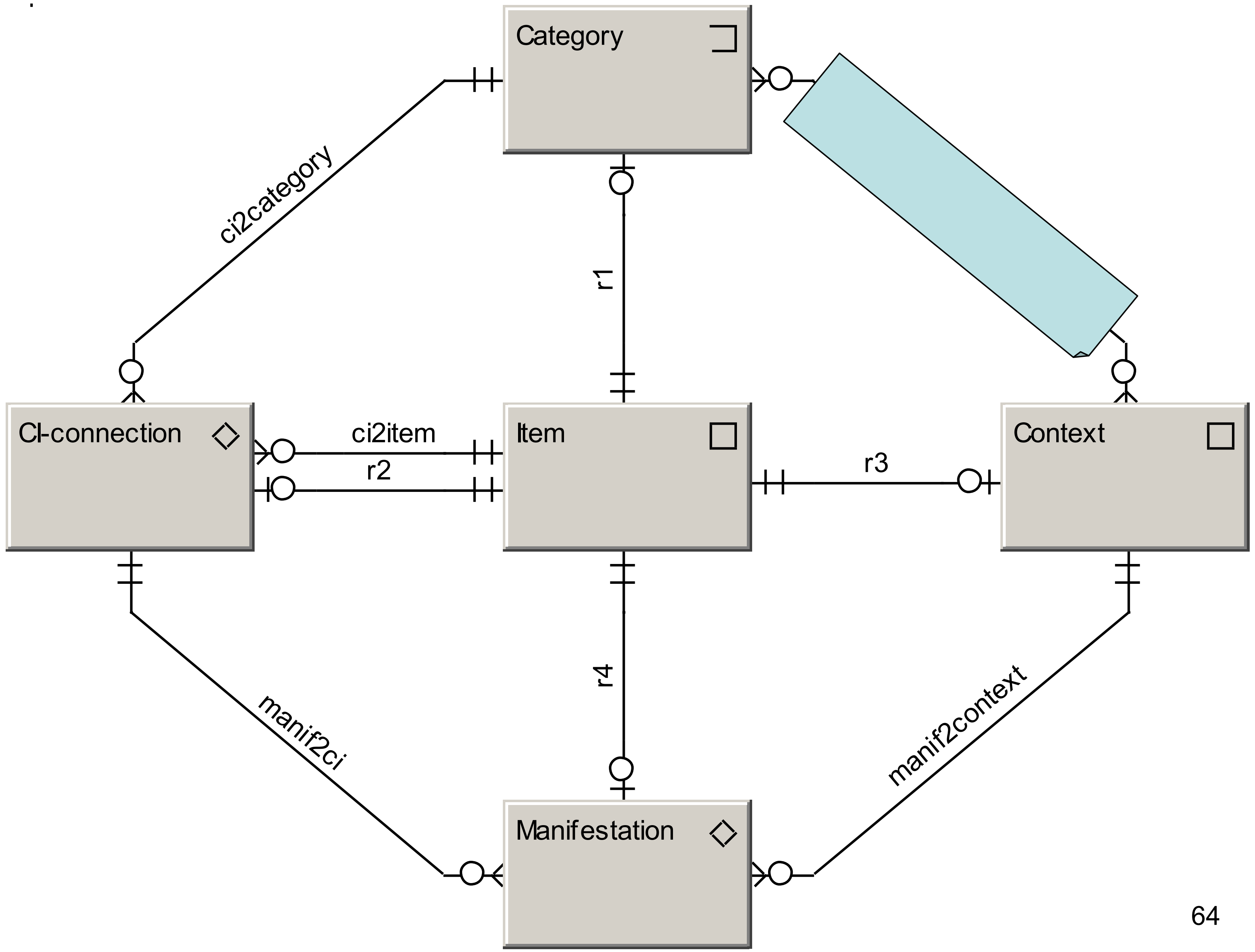
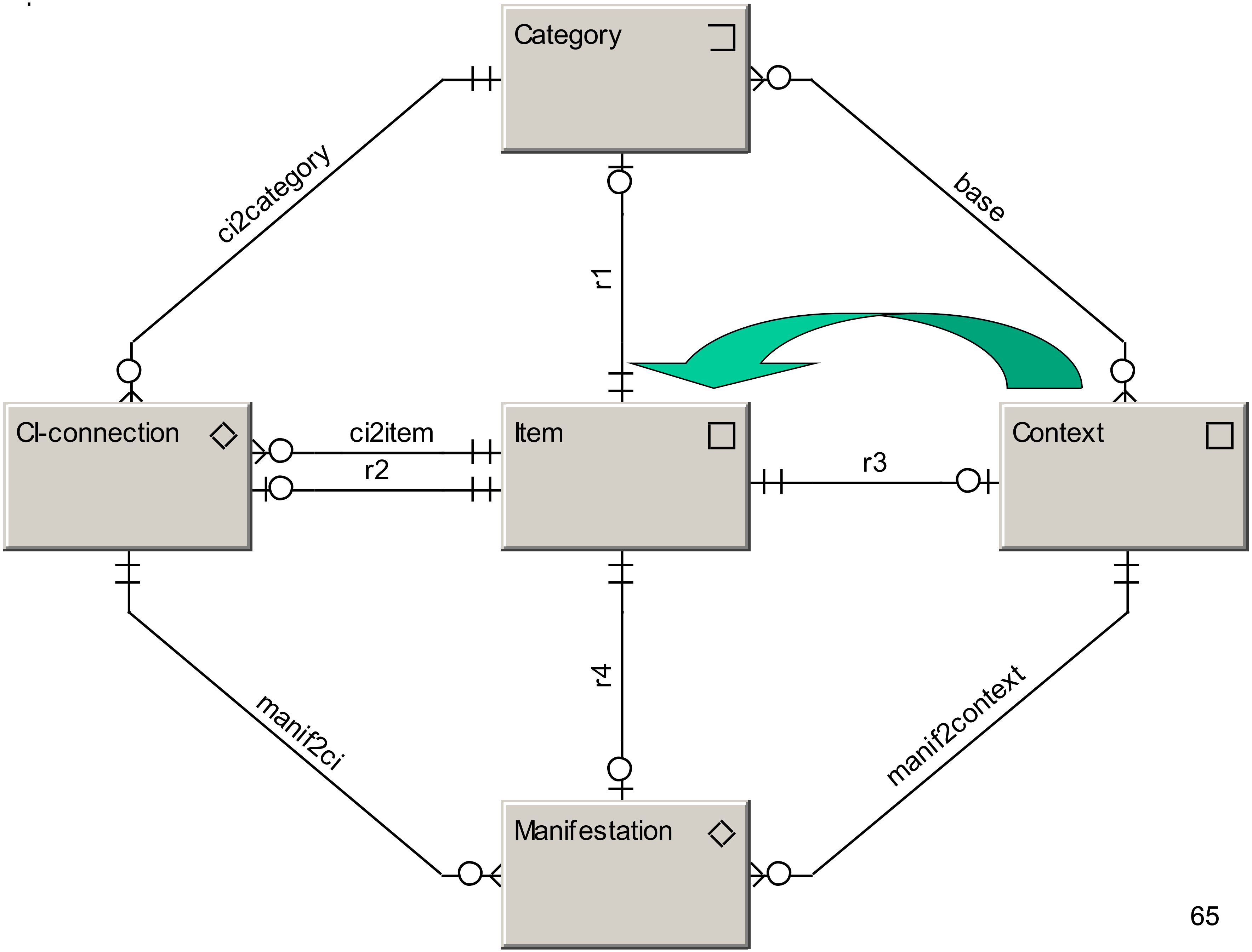


Fig. 1: T (elementary belief)

- (I, C, c = 1, a = a_{max}) is a context.
- Let T_1, …, T_n be arbitrary elementary beliefs. Than (T_1, …, T_n), i.e. a finite sequence of elementary beliefs is a context.


Category
ci2category
base
CI-connection
ci2item
Item
Context
manif2ci
manif2context
Manifestation


Category
ci2category
r1
CI-connection
ci2item
r2
Item
r3
Context
r4
manif2ci
manif2context
Manifestation


Category
ci2category
base
r1
CI-connection
ci2item
r2
Item
r3
Context
r4
manif2ci
manif2context
Manifestation

Context, Memory, Syplant

- Everything which is stored in Memory is a context.
- … more precisely a hierarchy of contexts.
- Contexts grow as plants.
- As they could be seen as “semantics plants”, we call them ‘syplants’.
- Each such ‘syplant’ can be an image of a domain.
- To have a Type System in one domain makes no problem.
- To extend this Type System to all domains makes a problem!
- (The metaphor of Syplant could help understanding of cognition and semantics.)

How to use this in practice?

Is there a practical way to avoid issues originating in unique Type System utilization?

Yes, it is …

- You have to implement:
  - the "T"
  - Manifestation operation
  - Embedding the result of manifestation into the particular context

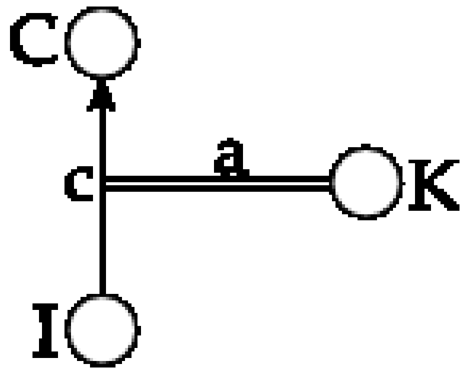


Fig. 1: T (elementary belief)

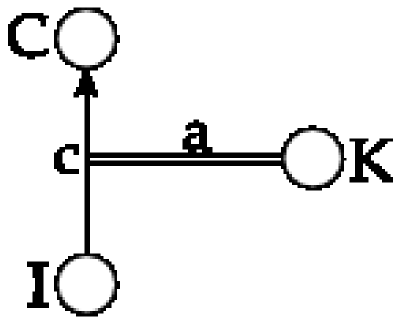


Fig. 1: T (elementary belief)

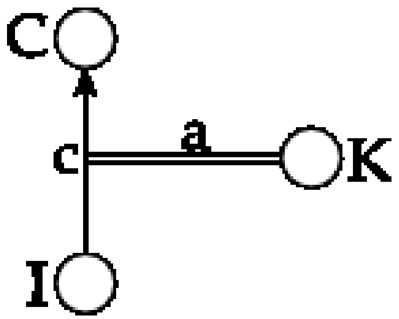
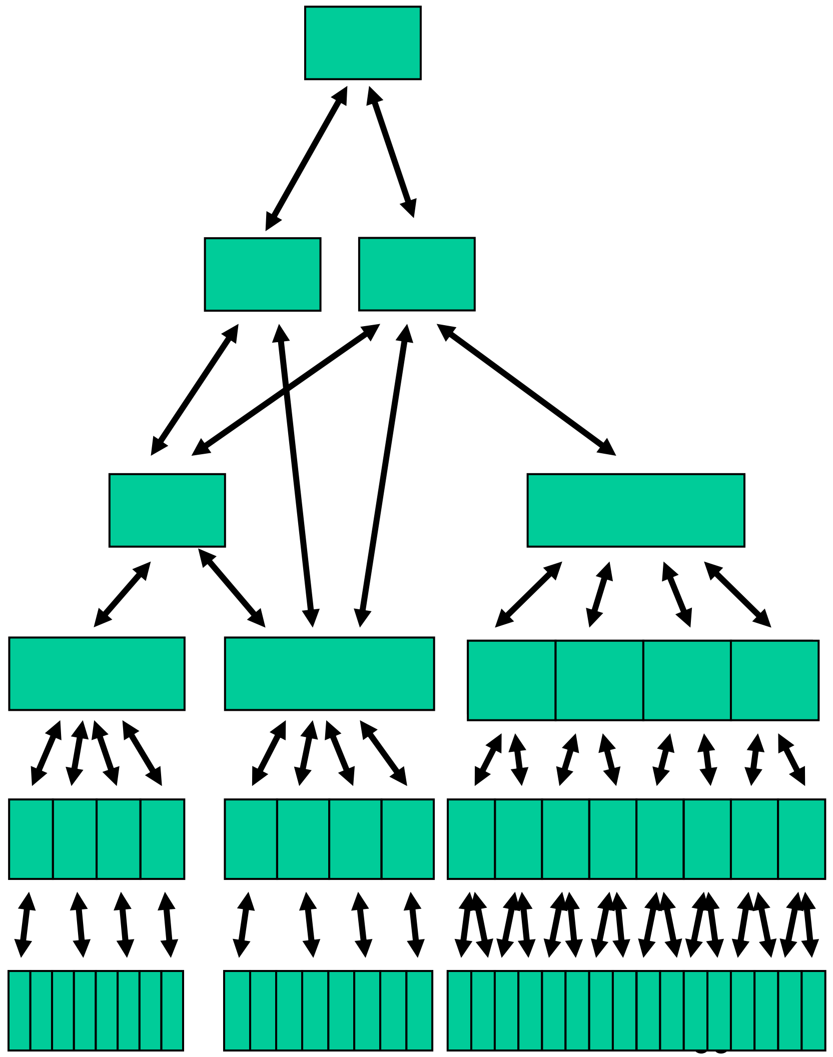
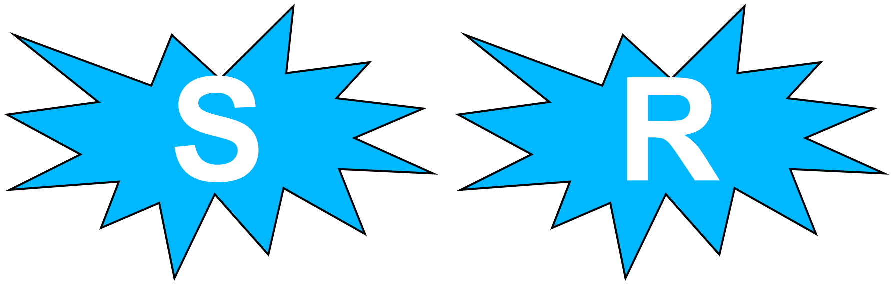


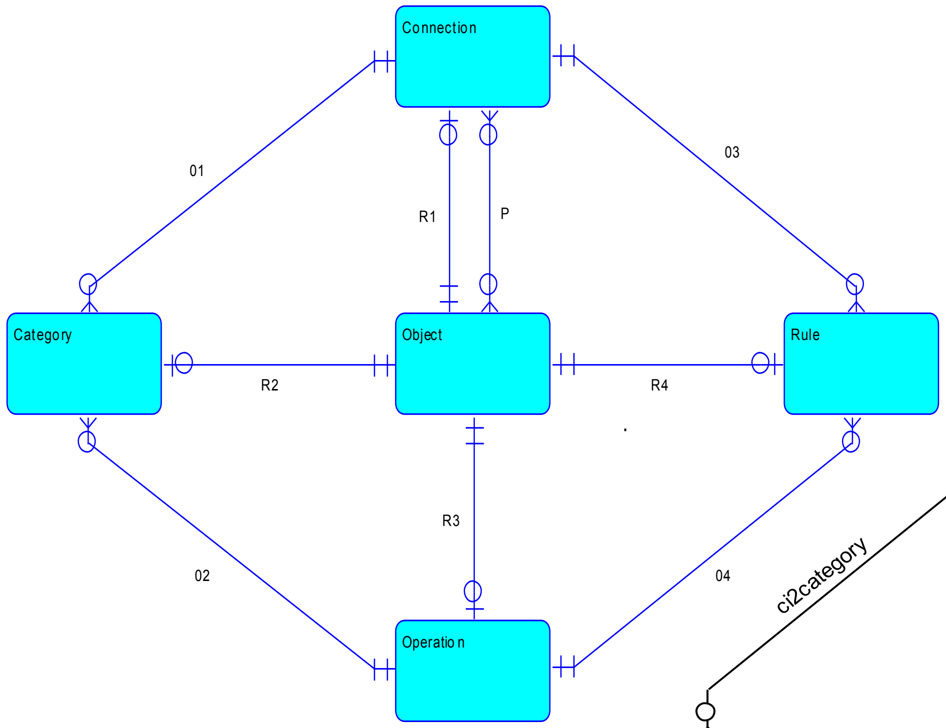
Fig. 1: T (elementary belief)

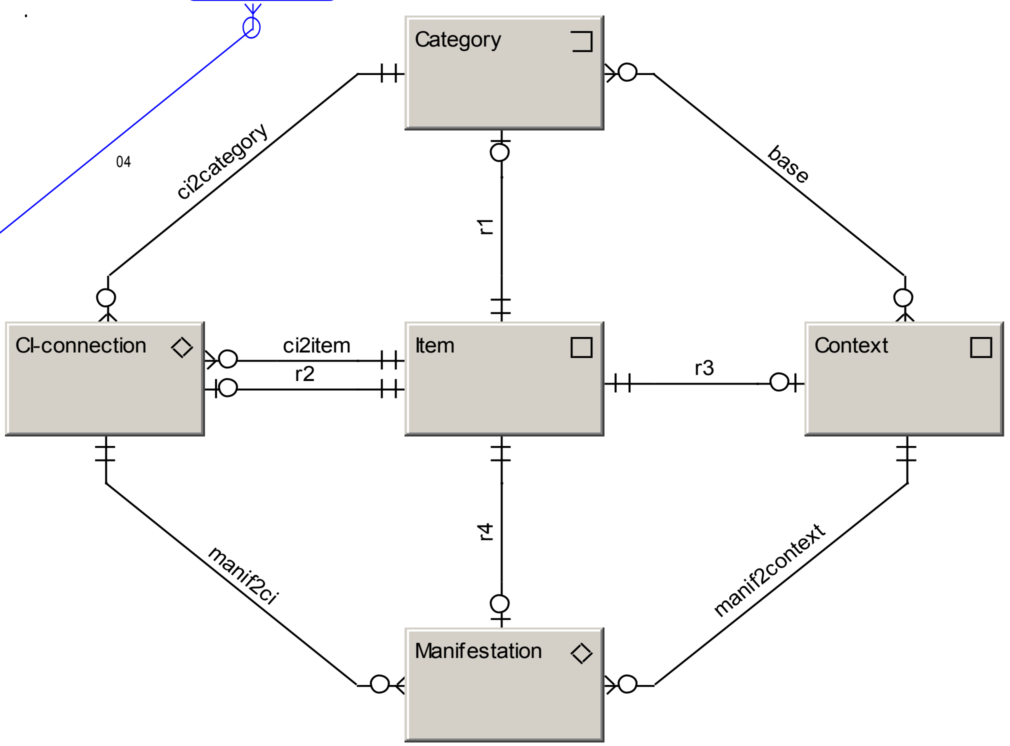
What are the lessons learnt?

- We were faced (again) with the USE-mode of the Fundamental hierarchy. But: another point of view than in Diam1 !
- Now we cognize (!) HOW to compose complex beliefs from less complex beliefs … … and from elementary beliefs.
- A complex belief is composed of elementary beliefs; each of them in turn can be a complex one.




S
R


Connection
01
R1
P
03
Category
Object
Rule
R2
R4
R3
02
04
Operation


Category
ci2category
base
r1
CI-connection
ci2item
r2
Item
r3
Context
manif2ci
r4
manif2context
Manifestation