General Modeling Patterns: Component Architecture, Communication, and Consolidation

PA116 – L3

(c) Zdenko Staníček, Sept 2010

EUROPEAN UNION esf

MINISTRY OF EDUCATION, YOUTH AND SPORTS

OP Education for Competitivness

UNIVERSITAS MASARYKIANA BRUNENSIS

INVESTMENTS IN EDUCATION DEVELOPMENT

Three parts:

- Component architecture and component communication
- Consolidation of components
- An addition: Definition and Use of DM

Content of the first part

- Principles of decomposition of CDM/IDM to components
- Component interface—Deputy/Prime-Entity
- Mother component
- Inherited properties of Deputy from Prime-Entity in Mother component
- Attributes of Deputy which are not attributes of Prime-Entity in Mother component
- Possible implementation

Why decomposition

- Problem complexity vs. Time to solve it
- Constrained resource capacity to find solution
- Constrained budget and preference to build up system step-by-step – part-by-part
- Components made by “experts” are more effective
- Non-uniform further development of IS in time
- Non-uniform dynamics of usage

Component Recognition Principle

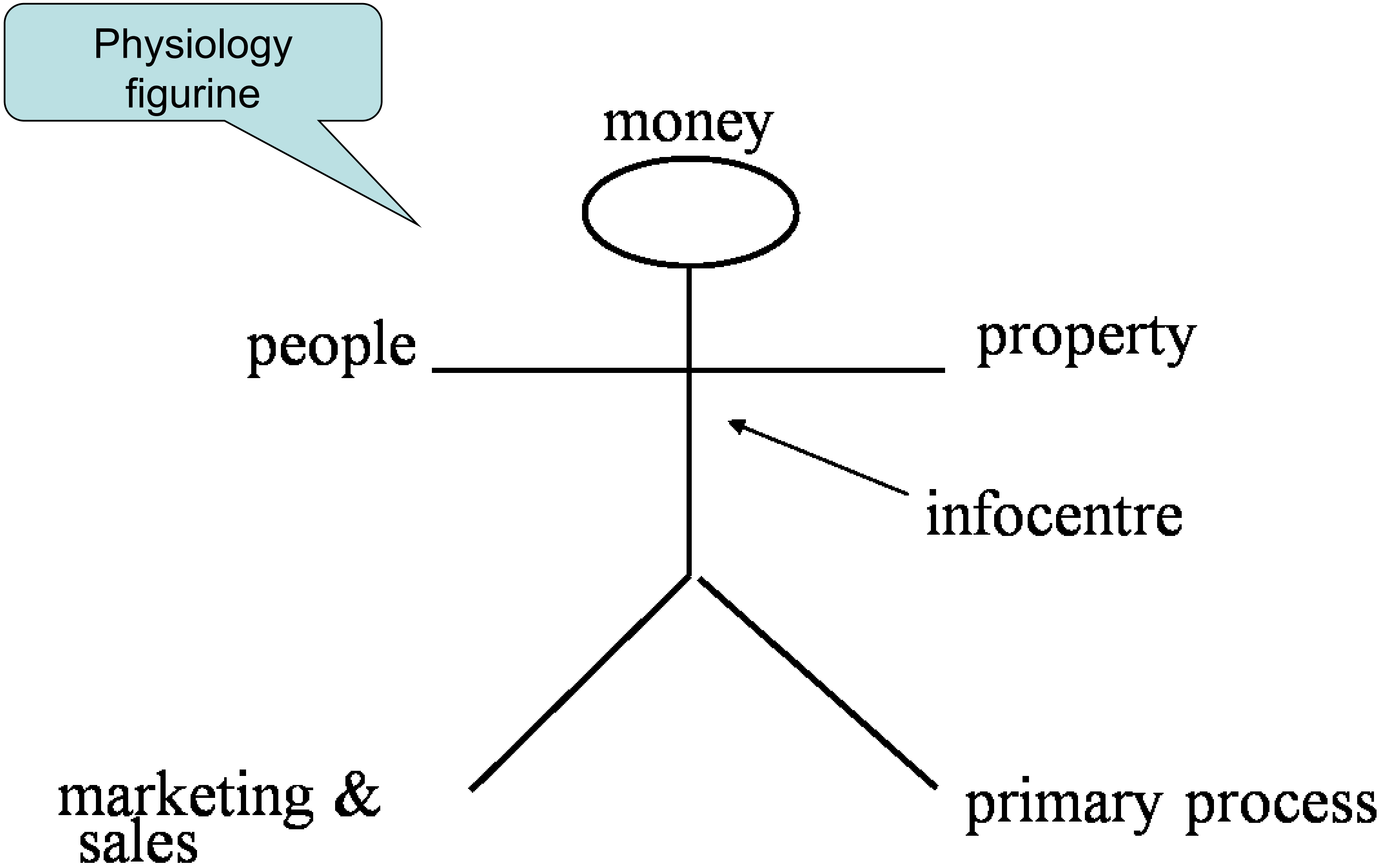
Component is described by one data model, i.e. all information needed for its correct work are stored in its own data sets in such a structure that component as a whole provides expected information capability no matter what the current situation in surrounding environment is.

The number of outer connections is minimal for every **component** and thus the need of communication with other components is minimized.

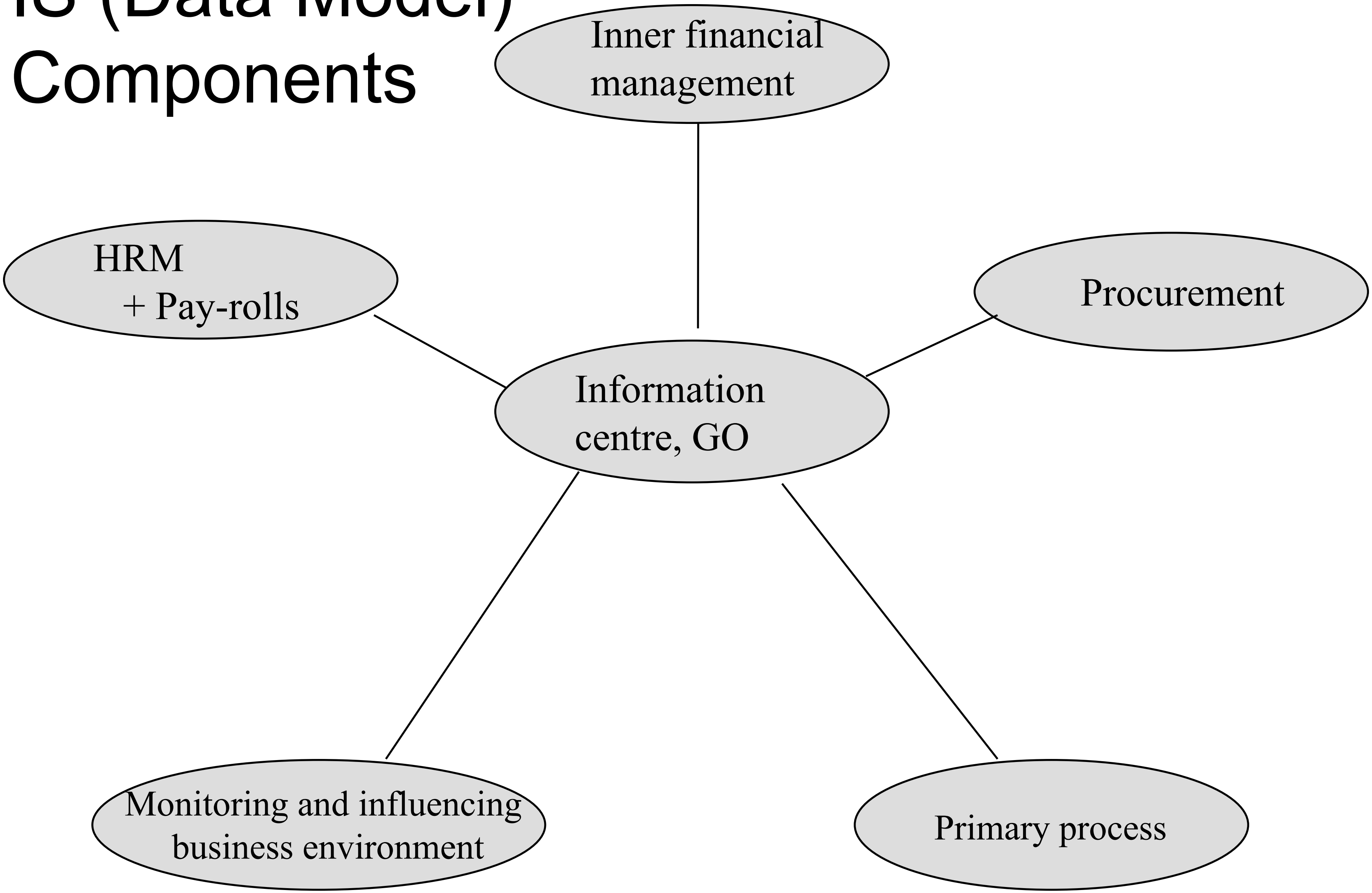
The inner complexity of **component** should not exceed the acceptable scale, i.e. it should be implemented in an acceptable time and it should start to provide benefits.

Import and export scheme, i.e. interface, of **component** is unambiguously specified.

Organization (business) physiology



IS (Data Model) Components


Inner financial management
HRM + Pay-rolls
Procurement
Information centre, GO
Monitoring and influencing business environment
Primary process

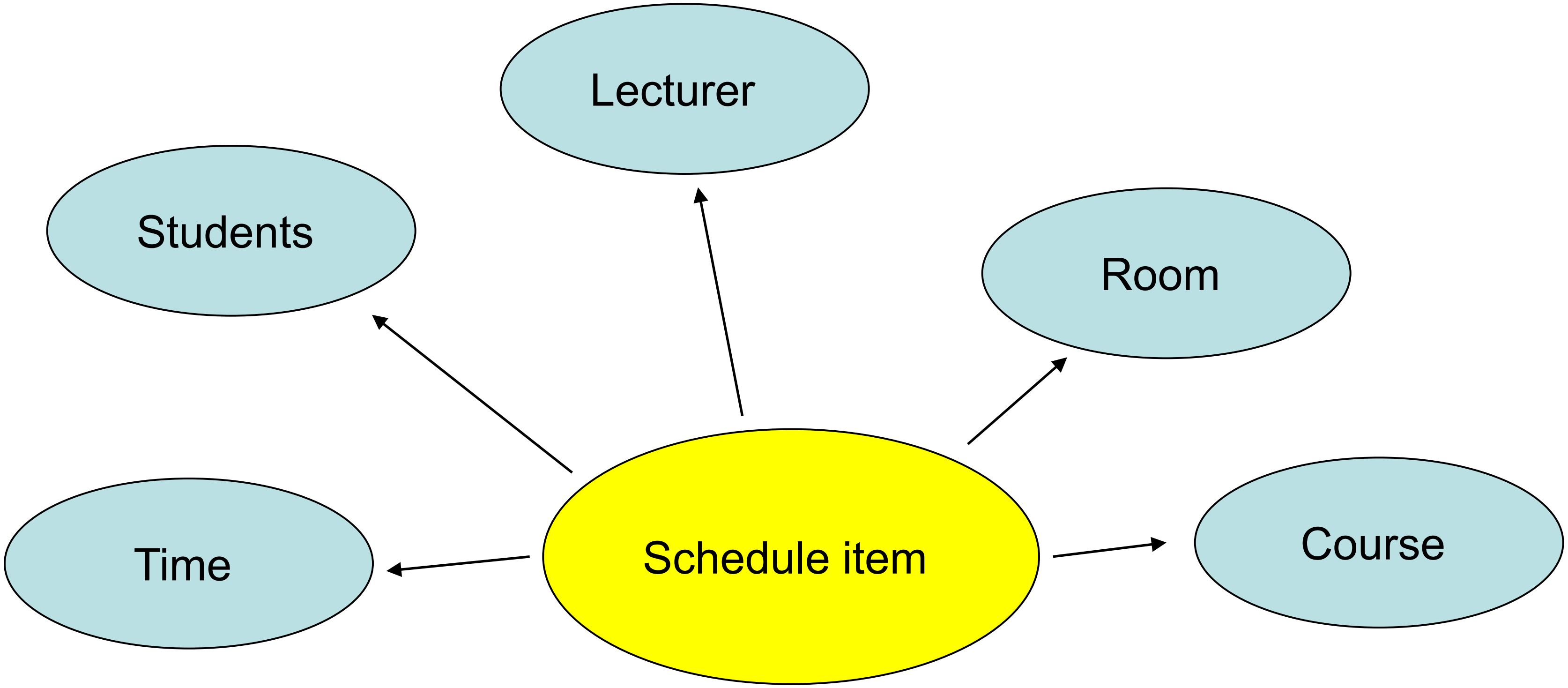
What is a Component?

- **Component:** entities + relationships + attributes
- **entity:** everything what is worthy of being separately investigated and what is interesting enough to file some characteristics about it..., and what is possible to distinguish from everything else
  - ”house”, “faculty”, “student”, “subject”, “product”, “customer”
- **relationship:** relation, assignment, binding
  - semantics of relationship
  - ”courses taught at given faculty“, ...
  - special connections: generalization-specialization, whole-part, …
- **attribute:** property, characteristics
  - attribute semantics
  - “name of student”, “number of ETCS”, “date of delivery”, ...

Examples

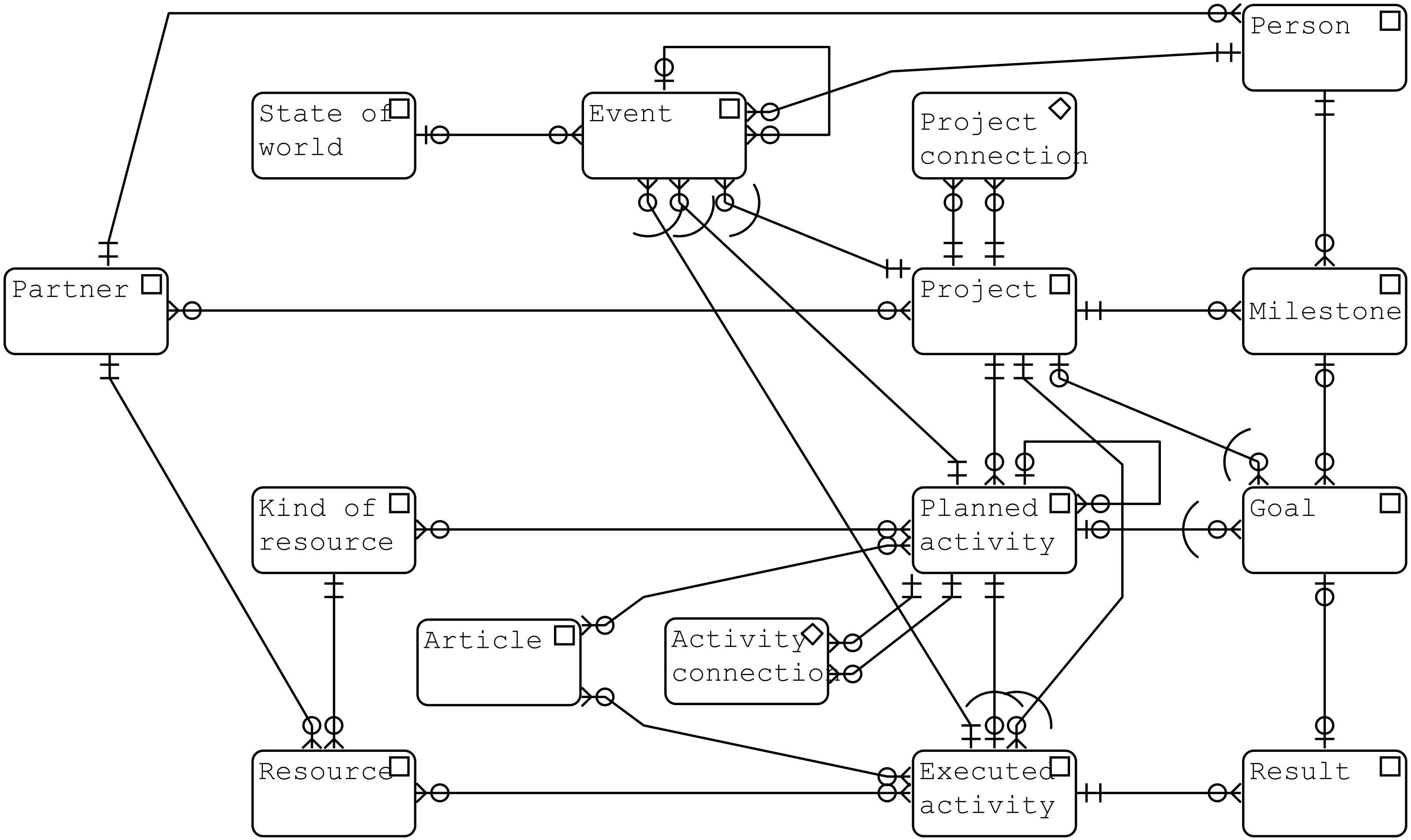
- Schedule component
  - to draw its bubble chart (next slide)
- Study component
- Pay-roll component
- Production component
- Marketing&Sales component
- Program, Porfolio Project Management component (will be demonstrated)
- Warehouse component (will be demonstrated)

Schedule

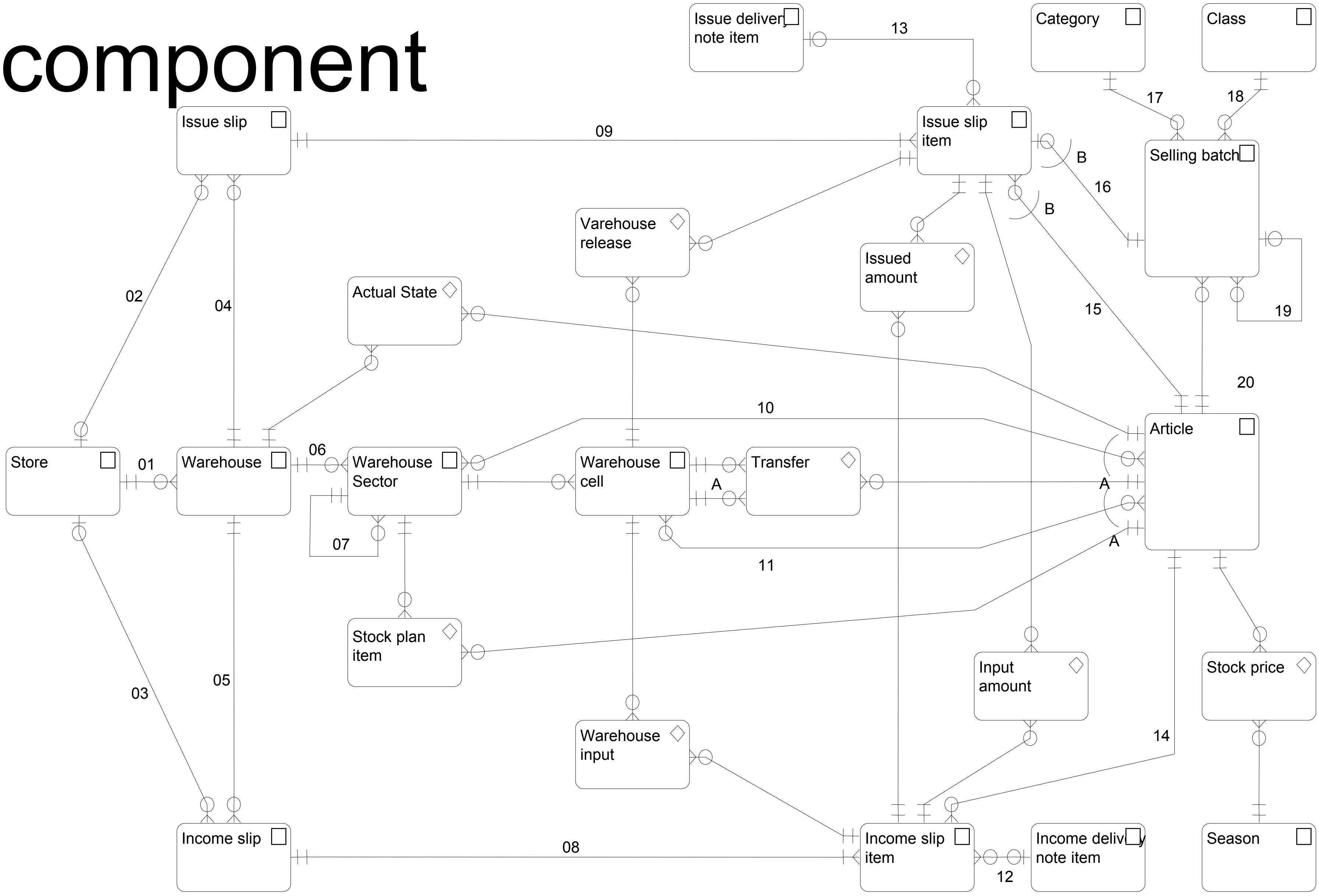


Time = (day_of_week, hour_from, hour_to)

PPPM component



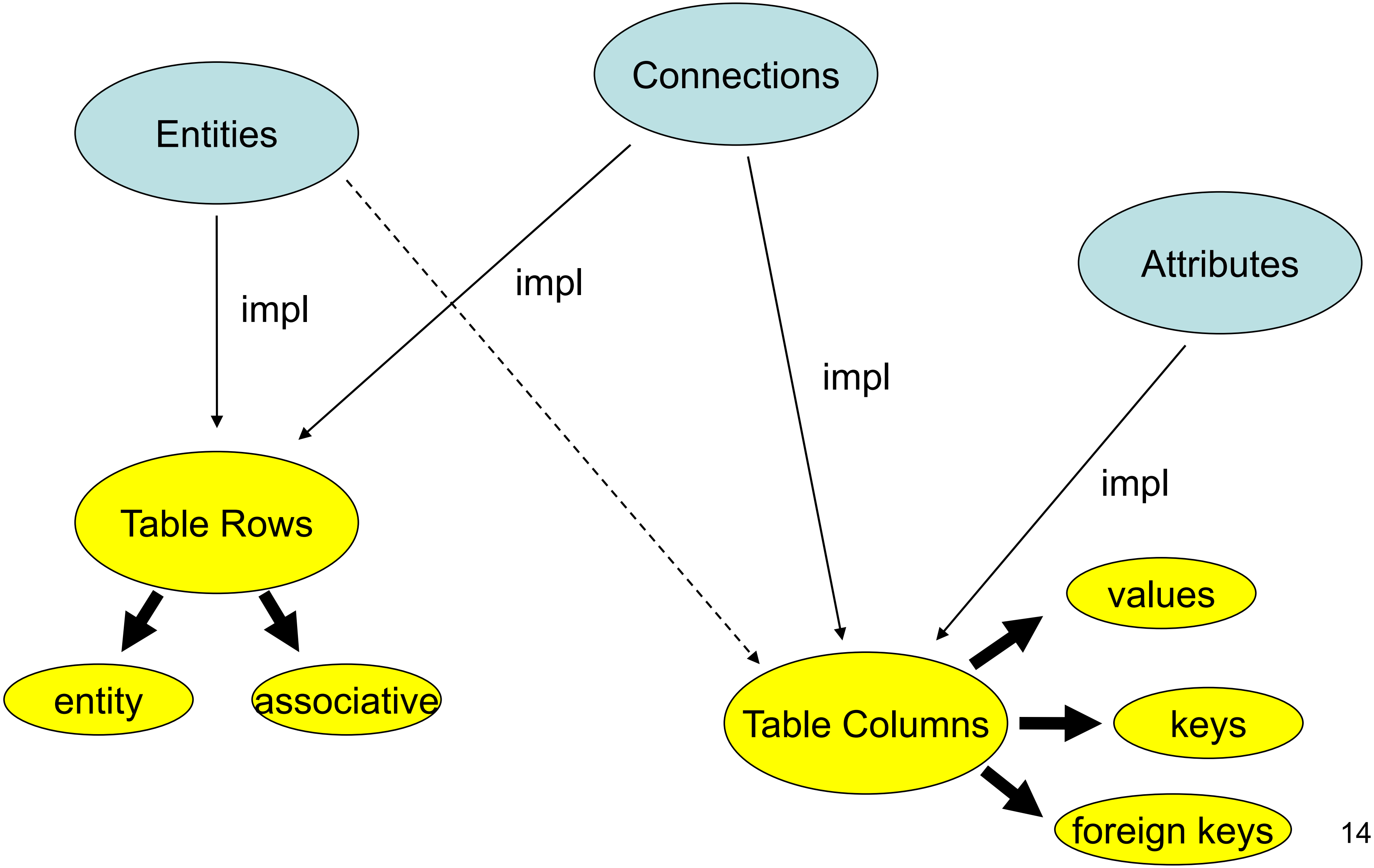
Warehouse component



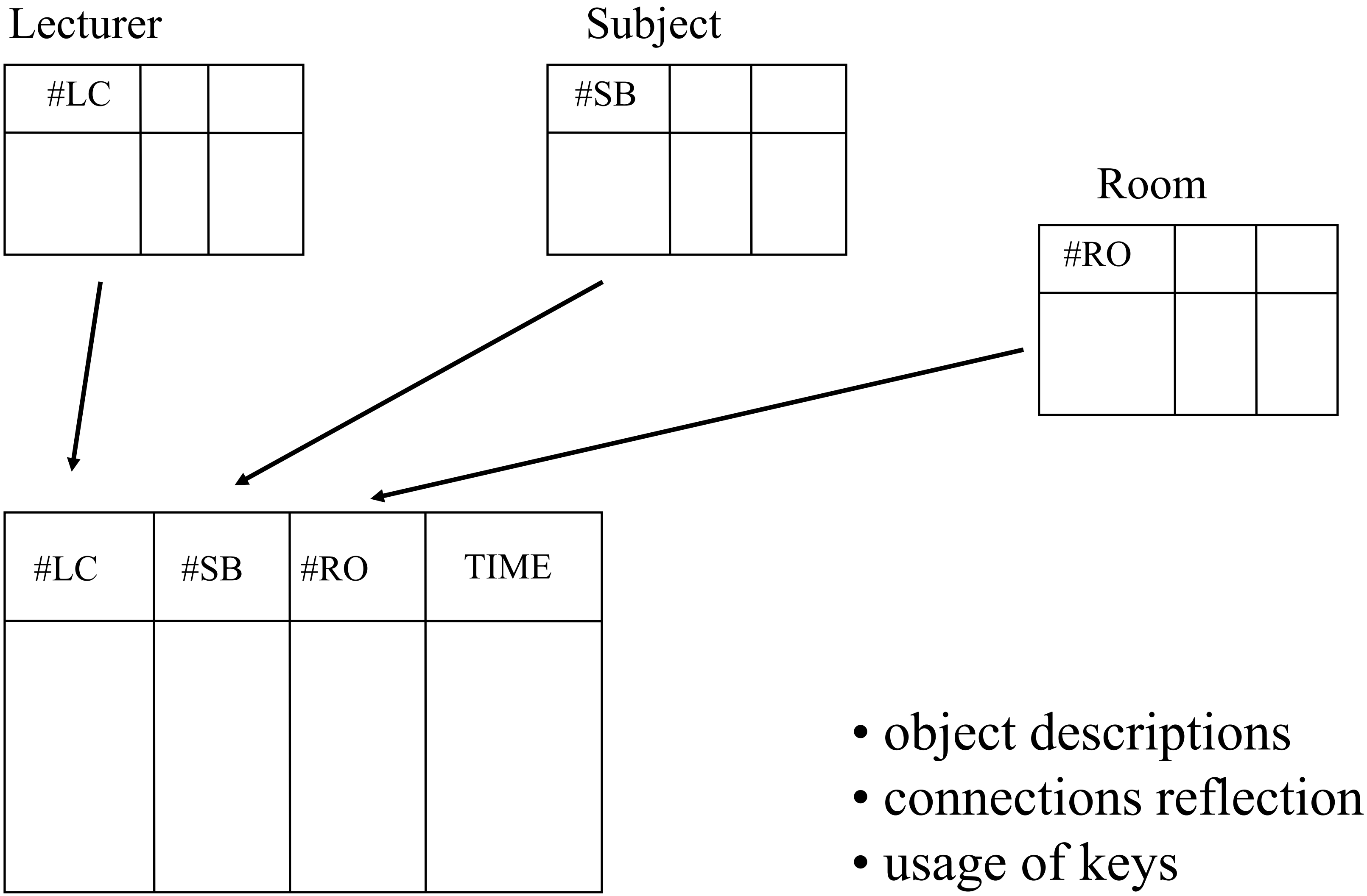
Implementation

- implementation:
  - Files / Tables,
  - Records / Rows,
  - Items / Columns,
  - Keys
- what is implemented in which way (bubble chart—next slide)
- Practical computer implementation—this is the USE form of Data Model
- tables for Schedule component example (will be demonstrated)

What is implemented in which way



Schedule



Lecturer

#LC		

Subject

#SB		

Room

#RO		

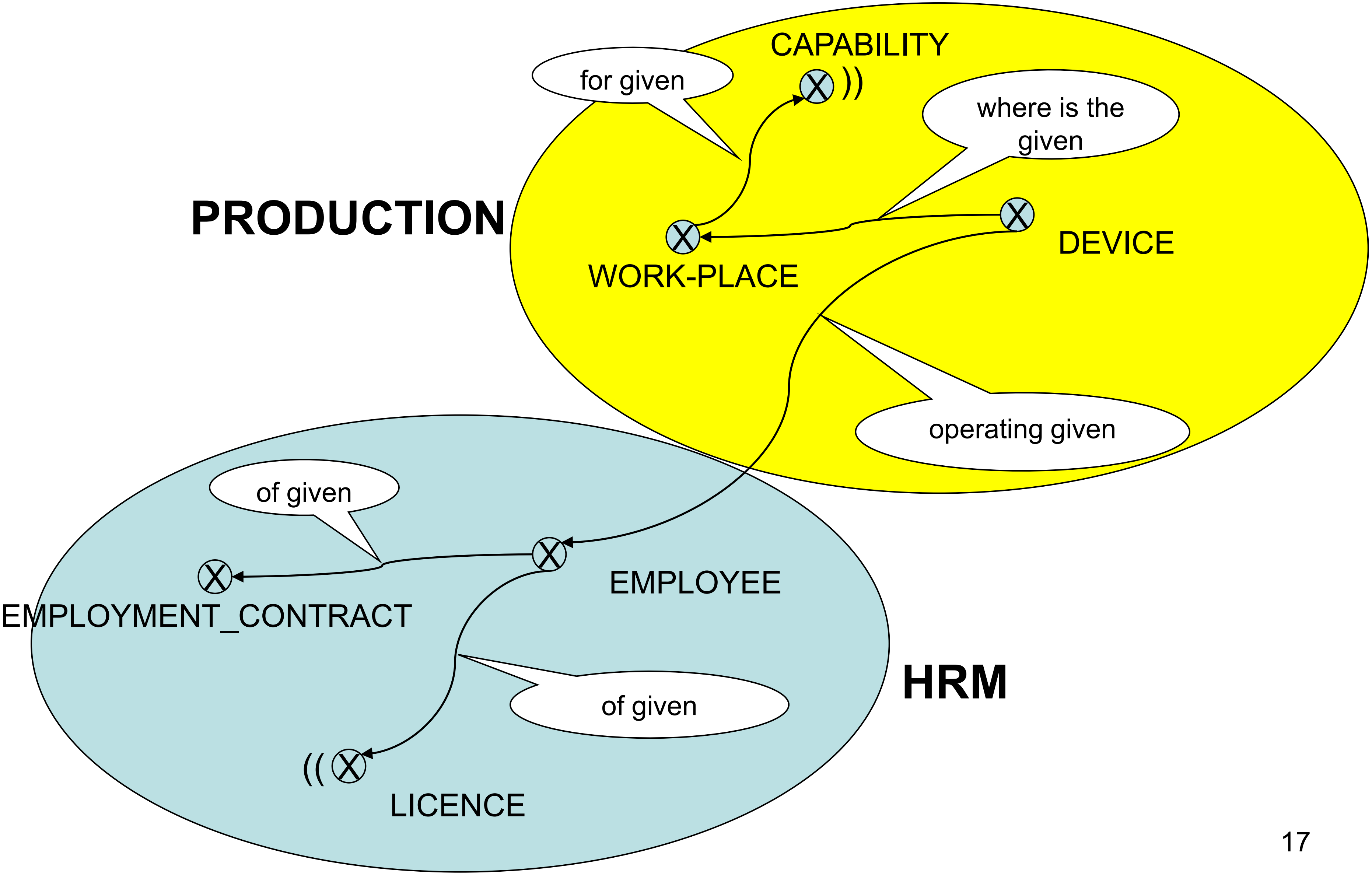
#LC	#SB	#RO	TIME

- object descriptions
- connections reflection
- usage of keys

Components communication

- „To answer a question arisen at one component it is often necessary to get several pieces of information from another component“
- Component interfaces – “Deputy”/”Prime Entity”
- Mother component
- Inherited properties of Deputy from Prime Entity in Mother component
- Attributes of Deputy which are not attributes of Prime Entity in Mother component

Communication through interface



PRODUCTION

CAPABILITY

for given

where is the given

DEVICE

WORK-PLACE

operating given

HRM

EMPLOYEE

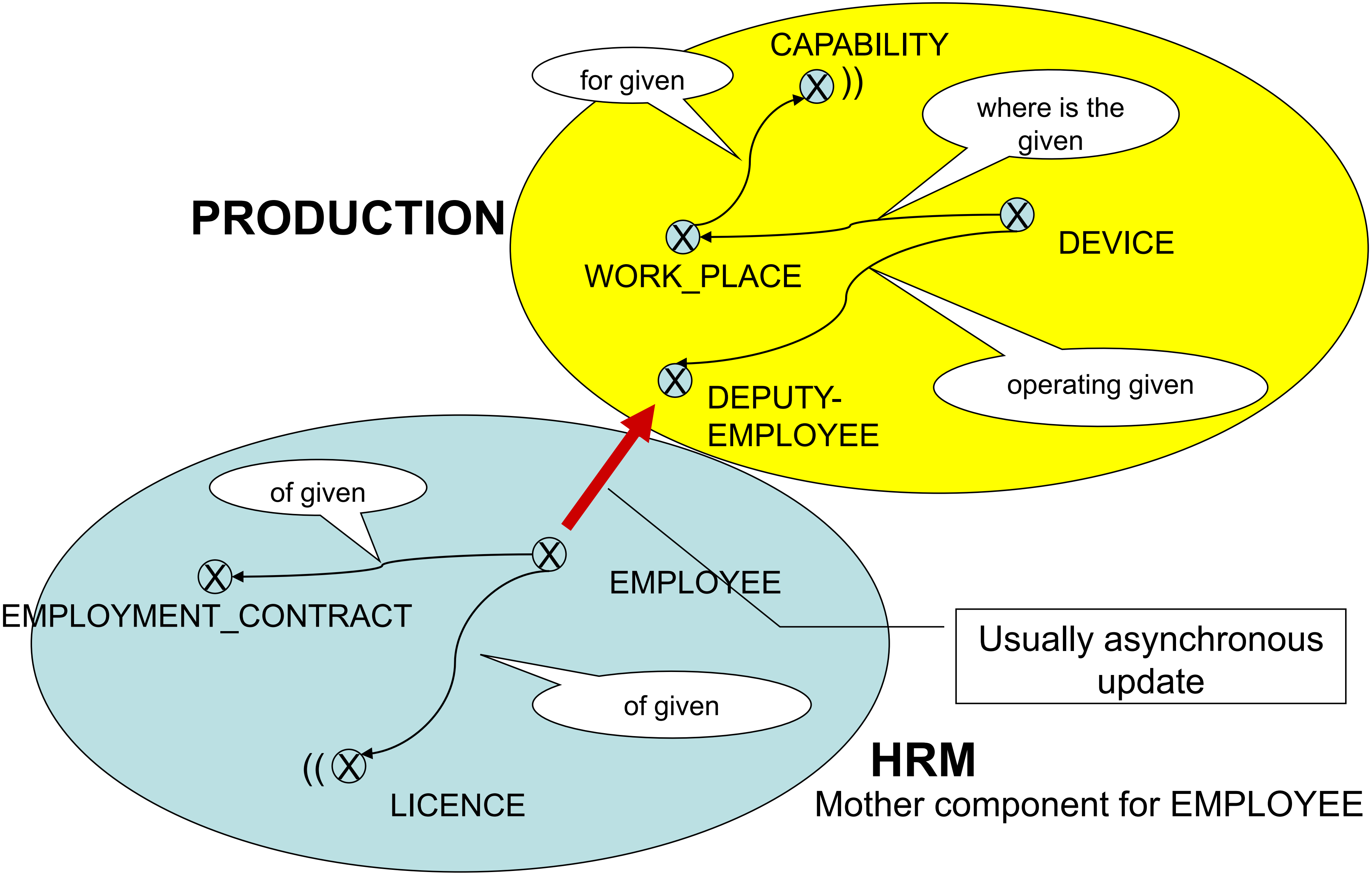
of given

EMPLOYMENT_CONTRACT

of given

LICENCE

Communication via “Deputies”



Usually asynchronous update

Mother component for EMPLOYEE

Practical implementation

- data replication among components
- on-line update
- update on demand
- etc.

- Important trick: descriptive attributes not existing in the mother component

Consolidation of data model

Content of the second part

- Why to consolidate
- Consolidation process
- Systematization of entities
- Classification of entitites
- Classification of connections
- Consolidation issues
- Rules of well-created data models

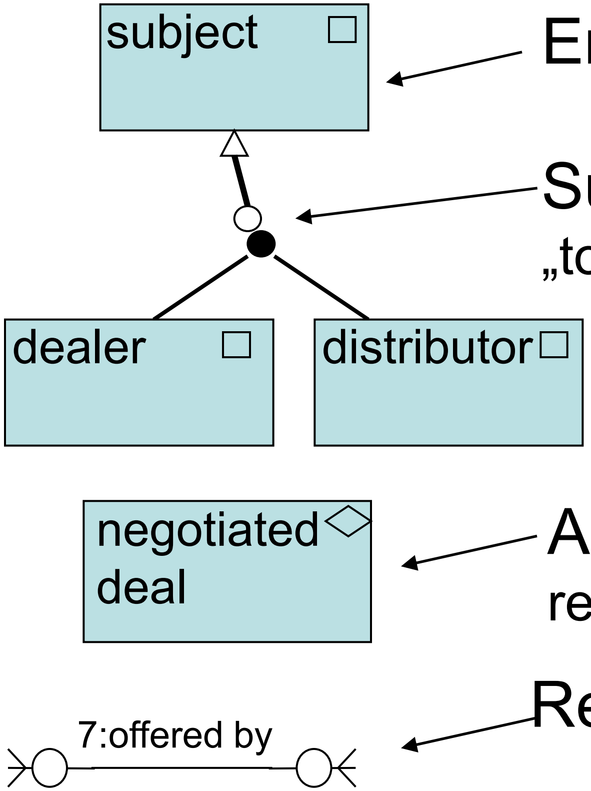
Why to consolidate

- Federative architecture -- Components
  - Domain complexity and extensiveness
- Components are designed by various teams of analysts – each of them focuses the reality in its own manner
- Synonymy and homonymy
- Different „handwritings“, inconsistency in the level of detail

Consolidation Process

- Revision of Definitions of all Entity Sorts in all Components
- Revision is based on certain categorizations
- After execution of Consolidation Process the Component’s Bases of Sorts are consistent in the way they were created by single “data modeler” person

What **objects** a model of a **Component** contains:



Entity sorts

Subtype – Supertype relationships

„total“ or „partial decomposition“

Particular sub-entities

Associative entities:

results of Binarisation Principle Application

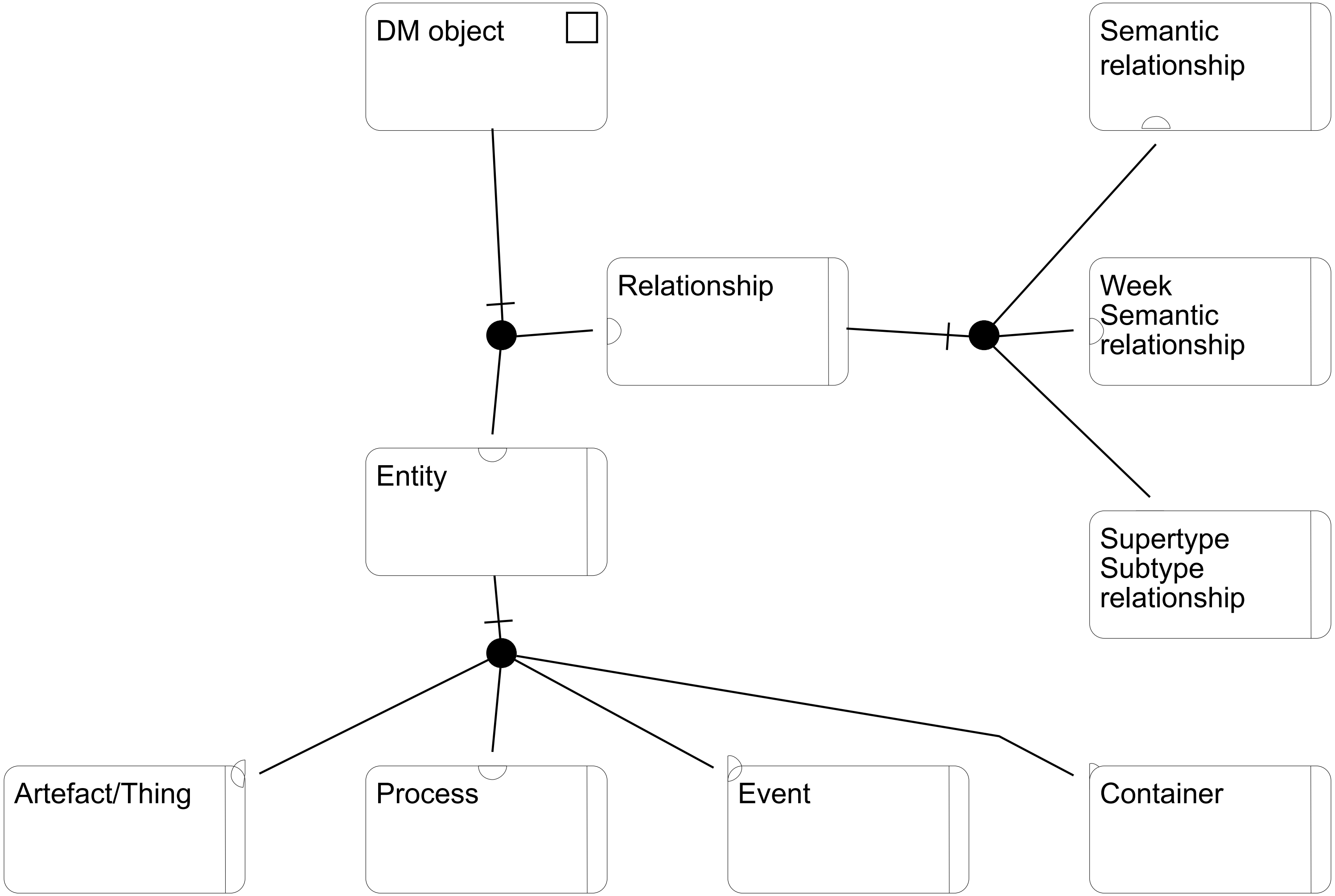
Relationships of entity sorts

Systematization of Entities

- The CATEG Data Model
- Generalization-Specialization Hierarchy only (another one than the Fundamental Hierarchy !!)
- Basic semantic types
  - **A**rtefact/Thing
  - **P**rocess
  - **E**vent
  - **C**ontainer

Categorization of DM objects

The root of the Tree



Classifications of connections

- Semantic connections
  its meaning is given by its semantics
- Weak semantic connections
  instance → type,
  item → associative entity
  (associative entity projection to its elements – binarization principle)
- Generalization-Specialization
  supertype–subtype
- Let's compare this approach with classification from Construction Patterns

Classification of entities

- logical point of view
  - Concepts ~ containers for concepts or categories
  - Instances ~ containers for items
  - Forms ~ containers for shapes or configurations
- philosophical point of view
  - tangible/material
  - intangible/immaterial
- existence status
  - plan
  - prescription / specification
  - realization /implementation / execution

Rules of well-designed data models

- Every two entities, which are differently classified or belong to different basic semantic types (A-P-E-C), cannot be connected by generalization-specialization connection
- It is not possible to connect an instance of thing to concept of process (type process) in the following meaning:
  Output (#Instance of Thing) from given (#Type Process)

Rules of well-designed data models

- If there is no week semantic connection of a concept entity, maybe appropriate instance entity is missing or a connection instance-to-category is missing.
- Entity of a type instance-of-process is usually bound to entities of a type instance-of-artefact (input or output of the process). If not, the model could be incomplete.

Consolidation issues

- concept of something versus container
- physical containers and abstract ones (constructions)
- ambiguousness of ordering of entities by categories (pragmatic stance: USE vs. MENTION)
- subjectivity of categorizations (analyst + agreements)
- principle of component consistency of single IDM

An Addition:
Definition and use of data model

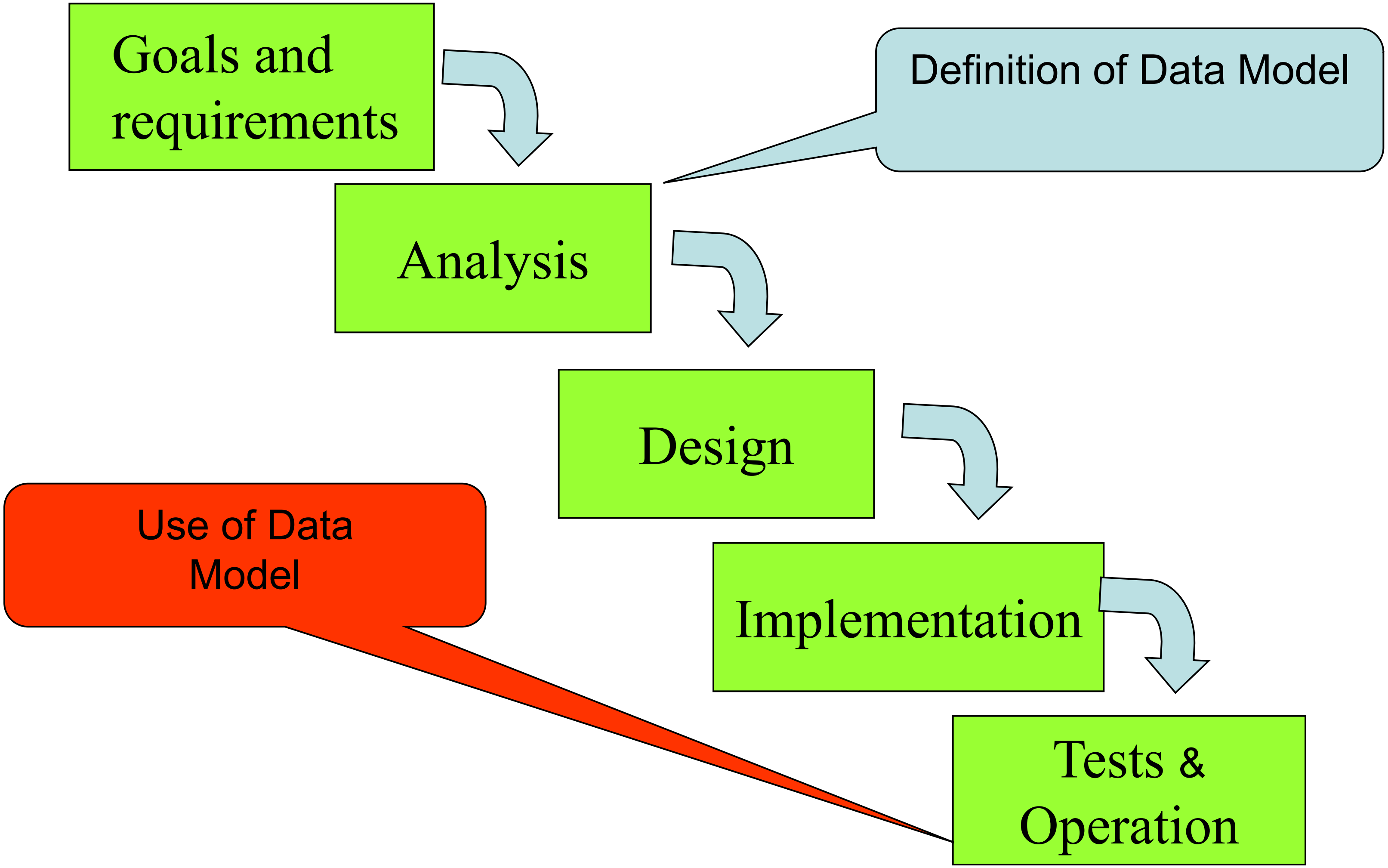
What means a DEFINITION of data model

- Specification of conceptual system which will be used for mentioning (MENTION) within a given ***Domain***.
- List of concepts used for structure, state, and behavior explication
  - Concepts identifying objects
  - Concepts identifying connections
  - Concepts identifying operations
- Exact determination of information capability needed for controlling and execution of processes in a given ***Domain***.

What means a USE of data model

- It is ensured that the given information capability (the one defined by the data model) is available in concrete controlling and execution of processes in a given ***Domain***.
- This is done by using technical means (HW, SW).
- Examples:
  - Implementation of IS with a given data model,
  - Data model as a “technical dictionary” used to make oneself understood to anybody and anything when controlling and executing processes in a given ***Domain***.

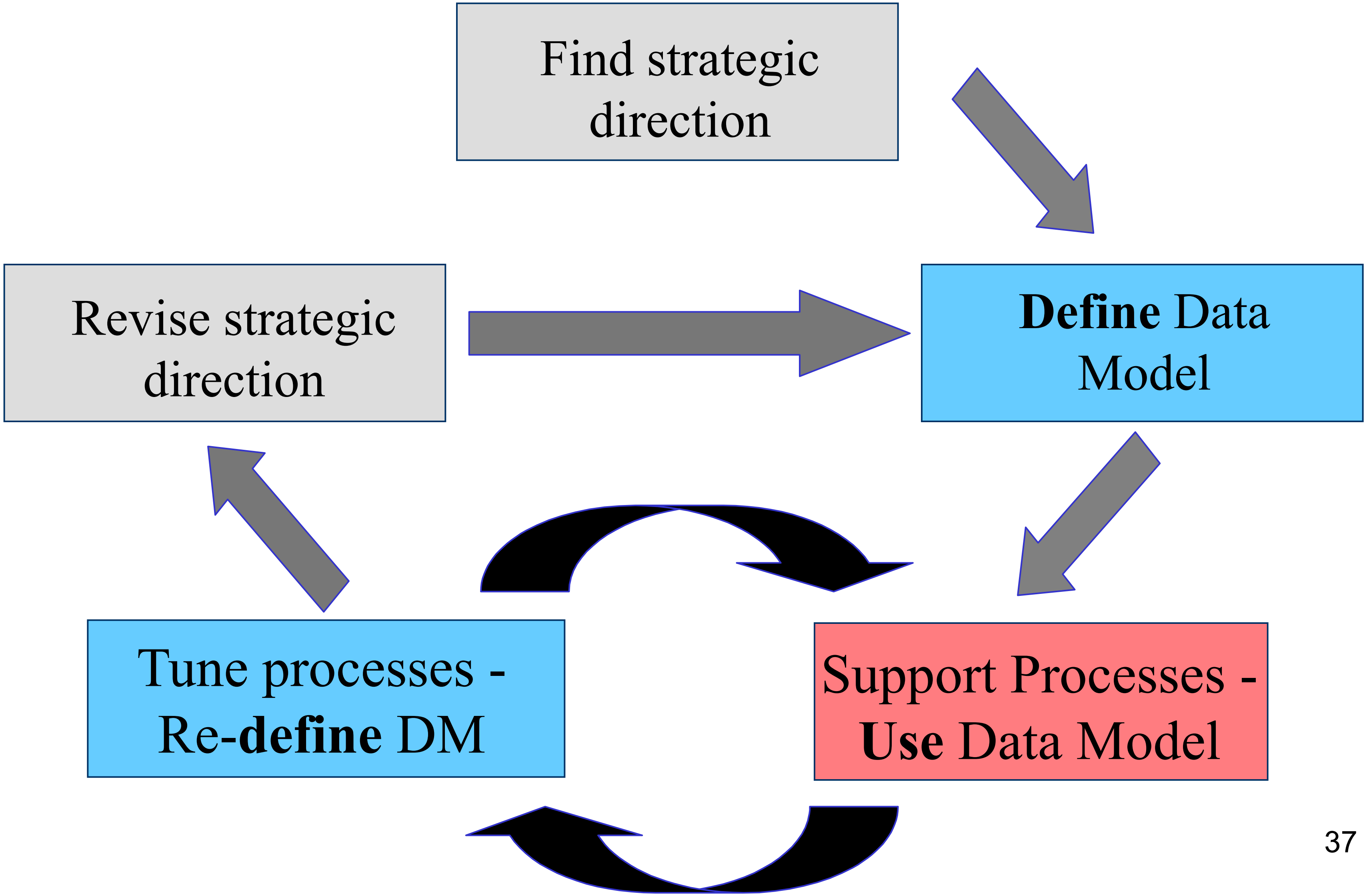
What do we have: conventional waterfall paradigm



Pitfalls of Waterfall Paradigm

- Long time from requirement specification to its satisfaction
- Lasting of steps „Design “ and „Implementation“
- Adaptation of application logic to the changed DB scheme
- To implement IS means to preserve or conserve status quo!

What do we need: Cyclical Paradigm at work



Find strategic direction

Define Data Model

Revise strategic direction

Support Processes - **Use** Data Model

Tune processes - Re-**define** DM