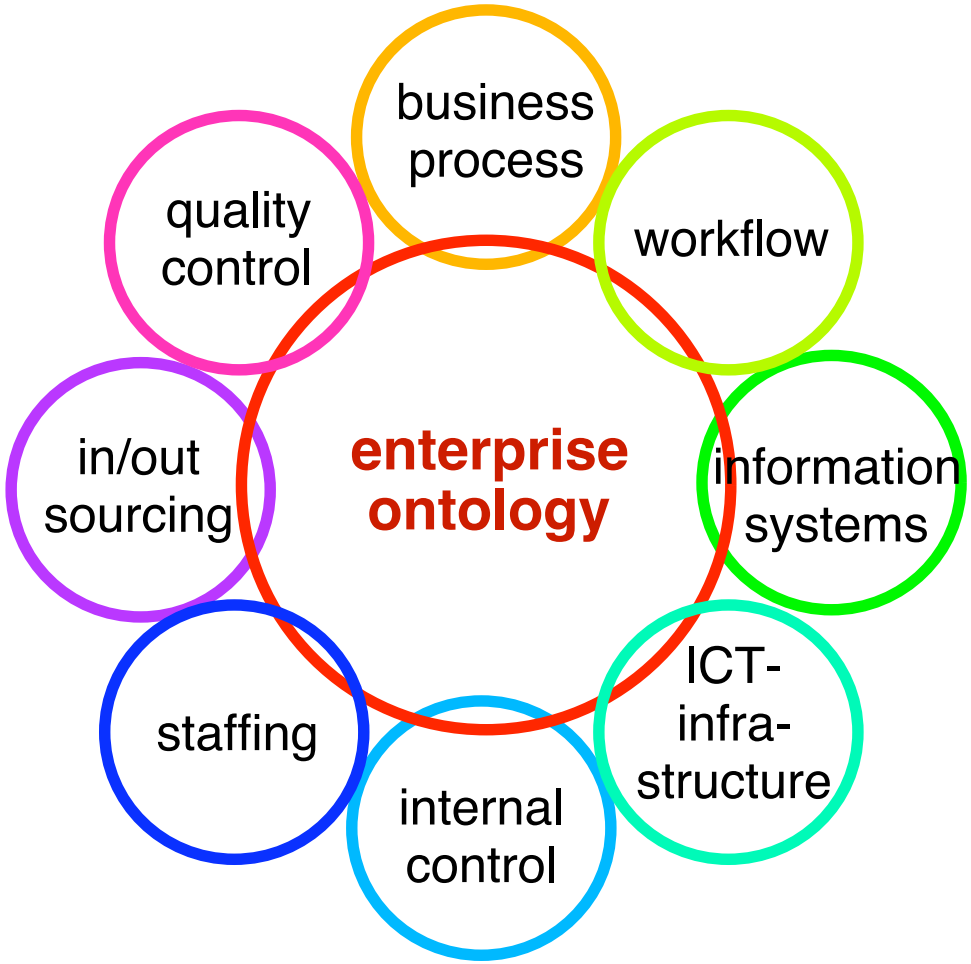
Enterprise Engineering

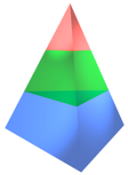
Enterprise Ontology

Jan L.G. Dietz
TU Delft

The problem

The solution



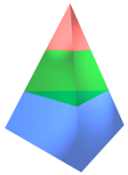


Intellectual manageability

Because we struggle with the small sizes of our heads as long as we exist, we need *intellectual techniques* that help us in mastering the complexity we are faced with (adapted from Edsger W. Dijkstra):

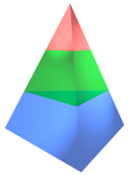
- *separation* of concerns
- effective use of *abstraction*
- devising appropriate *concepts*

Dijkstra successfully separated the concern for *correctness* from the concern for *efficiency* in programming and thereby made programming *intellectually manageable*.



The first wave (before 1970)

1st wave: ***Data Systems Engineering***

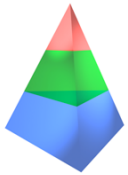


The 1970 revolution

In the sixties, Börje Langefors proposed the distinction between the *infological* view and the *datalogical* view.

Langefors' separation of *content* and *form* created a new field - *information systems engineering* - and made that *intellectually manageable*.

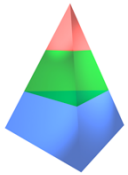
Few people understood the message deeply and changed their profession accordingly. Most people only dressed their windows, sometimes rather loudly.



The second wave (1970-2005)

1st wave: ***Data Systems Engineering (datalogical view)***

2nd wave: ***Information Systems Engineering (infological view)***

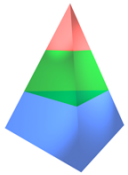


The current revolution

Like Langefors articulated the concern for the *content* of information, on top of its *form* ...

... we need to articulate the concern for the *intention* of information on top of its *content*, and develop the *ontological view* on enterprise.

This separation of intention and content will create a new field - *enterprise engineering* - and make that *intellectually manageable*.



Examples of intentions

(**P1** to **P2**) *I'd like to have such a bouquet*

P1 : request : P2 : person P1 has a bouquet B

(**P2** to **P1**) *Very well, sir*

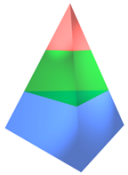
P2 : promise : P1 : person P1 has a bouquet B

(**P2** to **P1**) *Here you are*

P2 : state : P1 : person P1 has a bouquet B

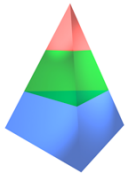
(**P1** to **P2**) *Thanks*

P1 : accept : P2 : person P1 has a bouquet B



The third wave (after 2005)

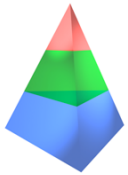
1st wave:	***Data Systems Engineering (datalogical view on enterprise)***
2nd wave:	***Information Systems Engineering (infological view on enterprise)***
3rd wave:	***Enterprise Engineering (ontological view on enterprise)***



What is ontology?

As a branch of philosophy, ontology investigates and explains the nature and essential properties and relations of all beings, as such, or the principles and causes of being. Otherwise said, ontology is the metaphysical study of the nature of being and existence.

As a modern concept in Computer Science (Artificial Intelligence), an ontology is a formal and explicit specification of a shared conceptualization among a community of people (and agents) of a common area of interest.



What is enterprise ontology?

An enterprise ontology is a formal and explicit specification of a shared conceptualization among a community of people of an enterprise (or a part of it). It includes static, kinematic, and dynamic aspects.

In particular, an enterprise ontology satisfies the next five quality requirements (C_4E):

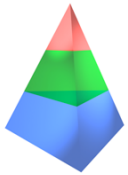
Coherence

Comprehensiveness

Consistency

Conciseness

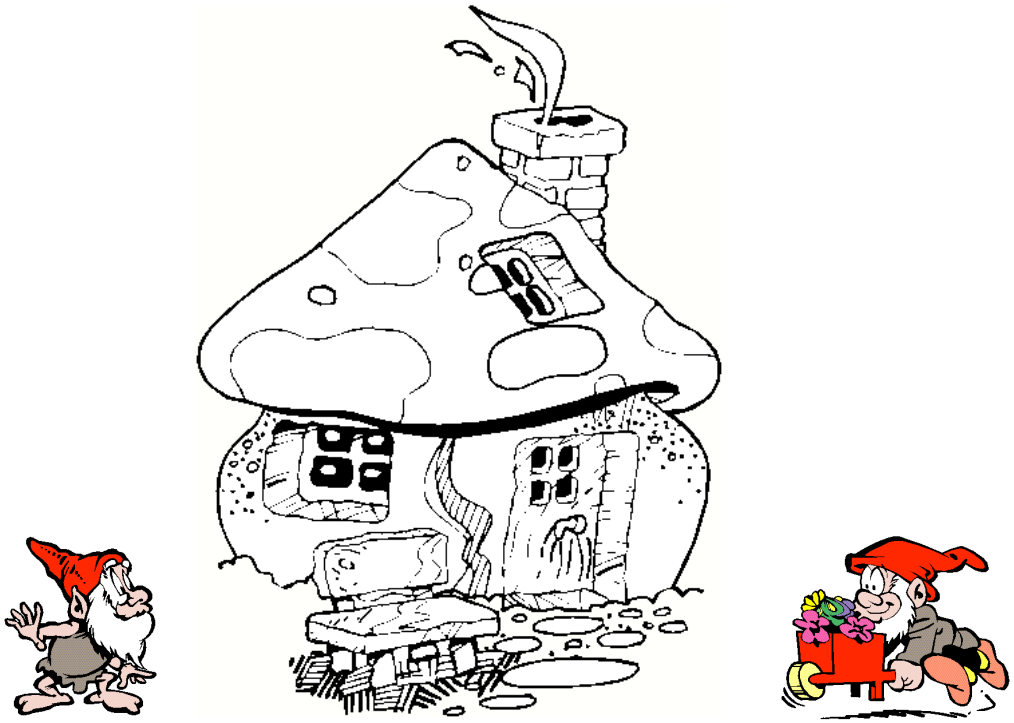
Essence

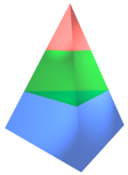


Who needs Enterprise Ontology?

- **Managers** need to understand the ontological essence of their enterprise because they are held accountable.
- **Developers** need to understand the organization, independent of its implementation.
- **Employees** - only the ontology of an enterprise shows the roles they fulfill deeply.
- **Users** - why should the operation of an enterprise be fully opaque to its users? Enterprise Ontology provides them the transparency they deserve!

The notion of system





What is a system?

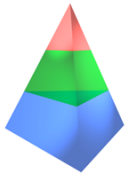
A system is a set of related elements (?)

The distinctive difference between system and aggregate is that a system has *emergent behavior*.

Like any type, the type system is defined by its properties. A thing is either a system or not. It makes no sense to consider something as a system if it is not a system.

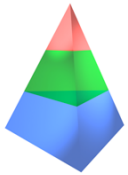
An important property of a system is the *category* to which it belongs (physical, chemical, mechanical etc.)

If a system belongs to exactly one category, it is called homogeneous, otherwise its is called heterogeneous.



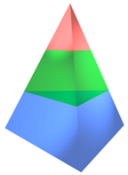
The teleological system concept

- Is about the *function* and *behavior* of a system
- Reflects the *purpose* of a system
- Is the *dominant* system concept in both the *natural* and the *social sciences*
- Is perfectly adequate for *using* and *controlling* systems
- Has the *black-box model* as the corresponding kind of model



The ontological system concept

- Is about the *construction* and *operation* of a system
- Is *indifferent* to the *purpose* of a system
- Is the *dominant* system concept in the *engineering sciences*
- Is perfectly adequate for *building* and *changing* systems
- Has the *white-box model* as the corresponding kind of model



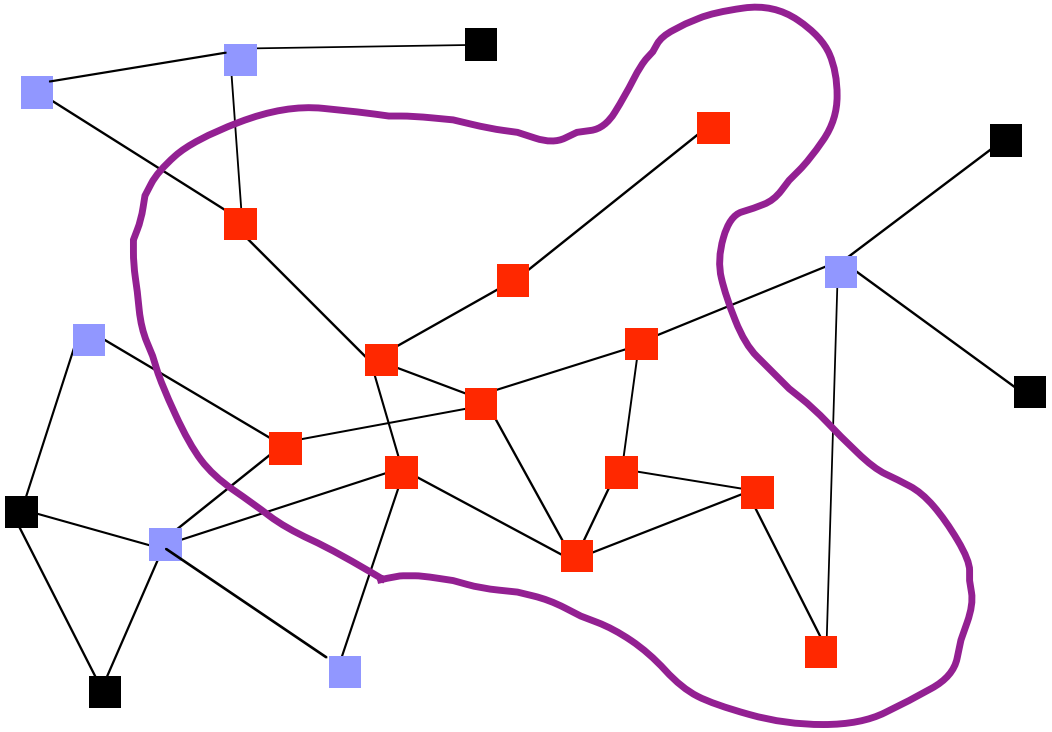
The ontological system definition

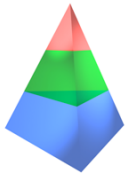
Something is a (homogeneous) *system* if and only if it has the next properties:

- *Composition*: a set of elements of some category (physical, social, biological)
- *Environment*: a set of elements of the same category
- *Production*: the elements in the composition produce things (products or services) that are delivered to the elements in the environment
- *Structure*: a set of influence bonds among the elements in the composition and between these and the elements in the environment

Depiction of the construction of a system

construction = composition + environment + structure





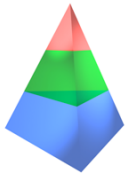
Heterogeneous systems

A system is *homogeneous* if its elements belong to *one category* (e.g. physical or biological or social). The elements are called *atomic* with respect to the category.

Homogeneous systems can be integrated in a *heterogeneous* system. This is possible if the (atomic) elements of the distinct homogeneous systems are somehow inseparably connected.

Examples of heterogeneous systems:

- A human being consists of a physical system, a chemical system, a biological system etc.
- A car consists of a physical system, a chemical system, an electrical system etc.



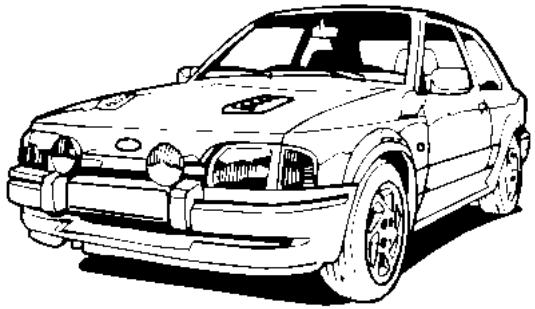
The white-box model (1)

A white-box model is a (direct) *conceptualization* of a concrete system.

A white-box model shows the construction and the operation of a system.

Example: Niels Bohr's model of the atom.

The white-box model (2)

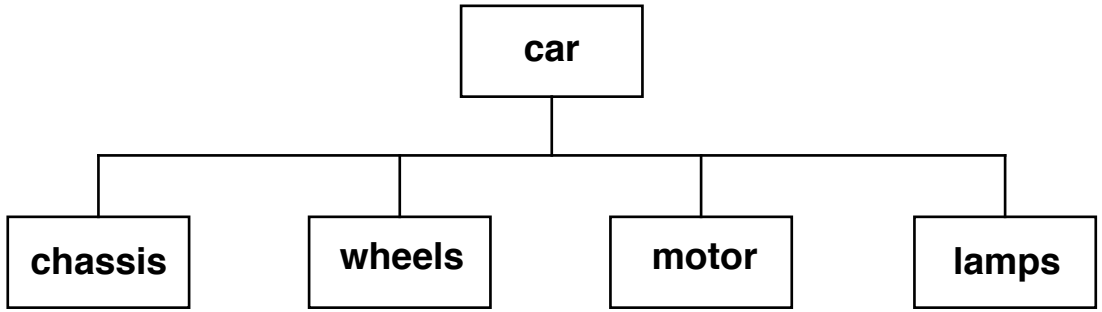


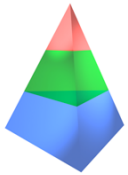
constructional (de)composition

the mechanic's perspective

construction :
the components and their interaction relationships

operation :
the manifestation of the construction in the course of time





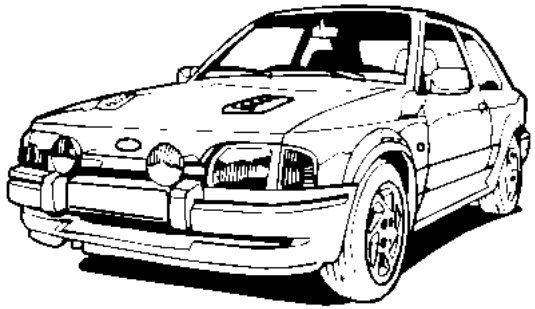
The black-box model (1)

A black-box model is a *conceptual system*. The relationship with a concrete system is not straightforward, because it is expressed in terms of the using system.

A black-box model shows the function and the (functional) behavior of a system.

Example: An economic model of an enterprise.

The black-box model (2)

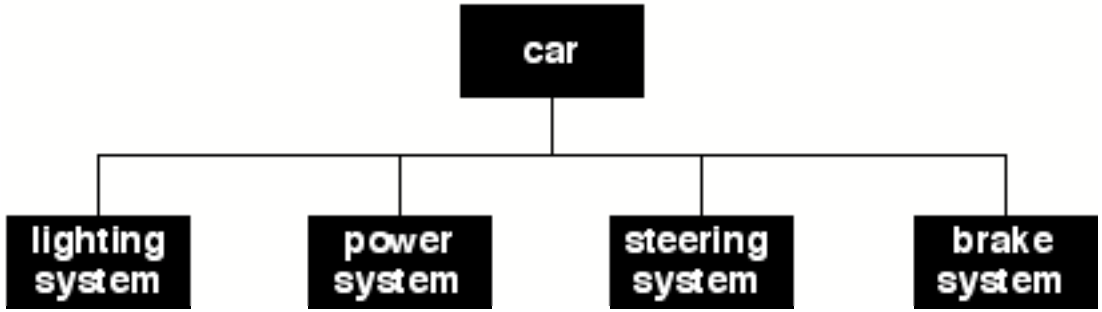


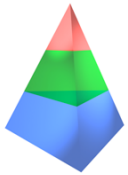
the driver's perspective

function :
relationship between
input and output

behavior :
the manifestation of the
function in the course of time

functional (de)composition



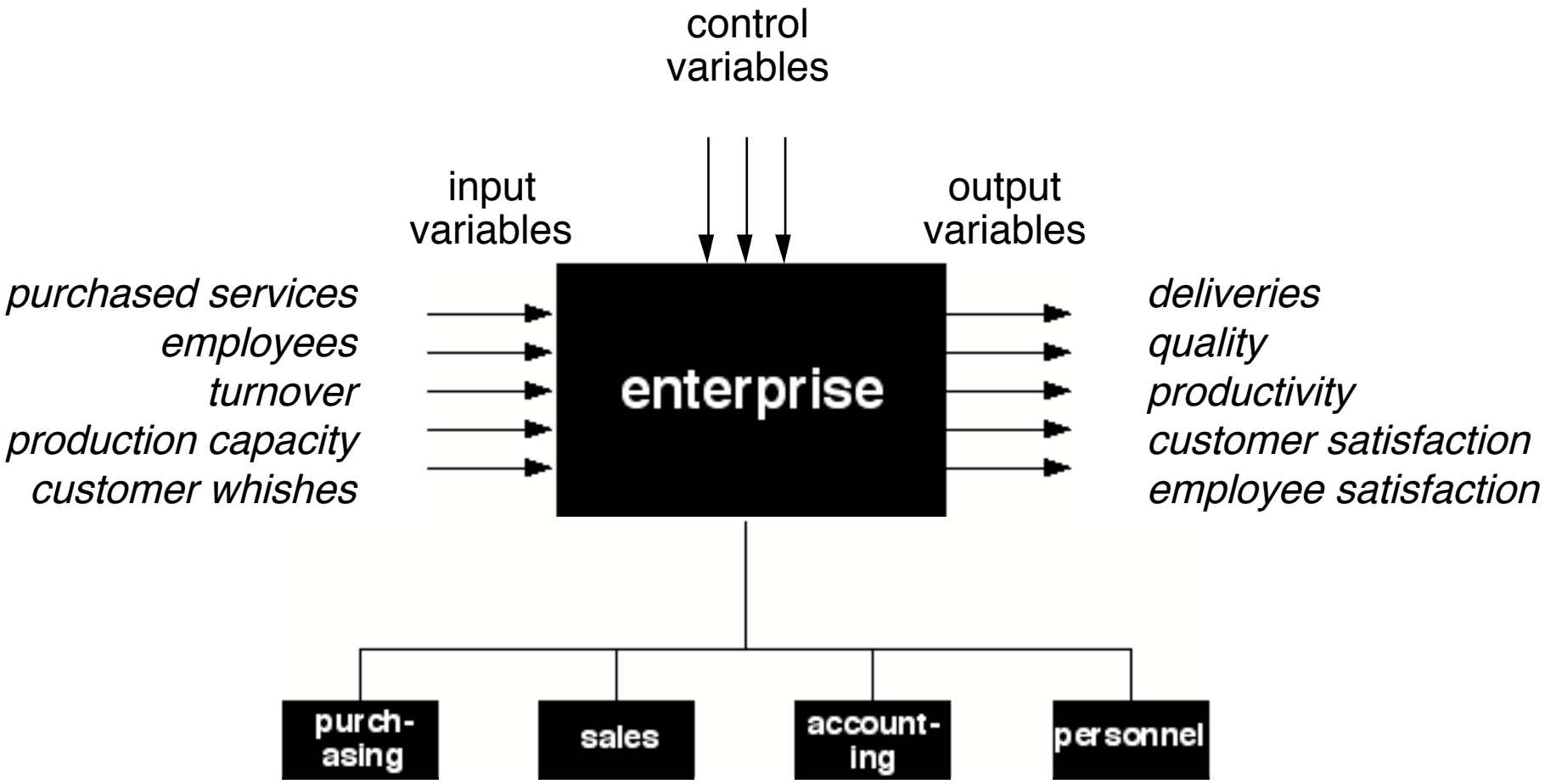


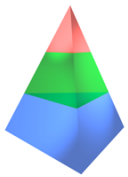
Business and organization

By the ***business*** of an enterprise is understood the *function* perspective on the enterprise. It is characterized by the products and services that are delivered to the environment.

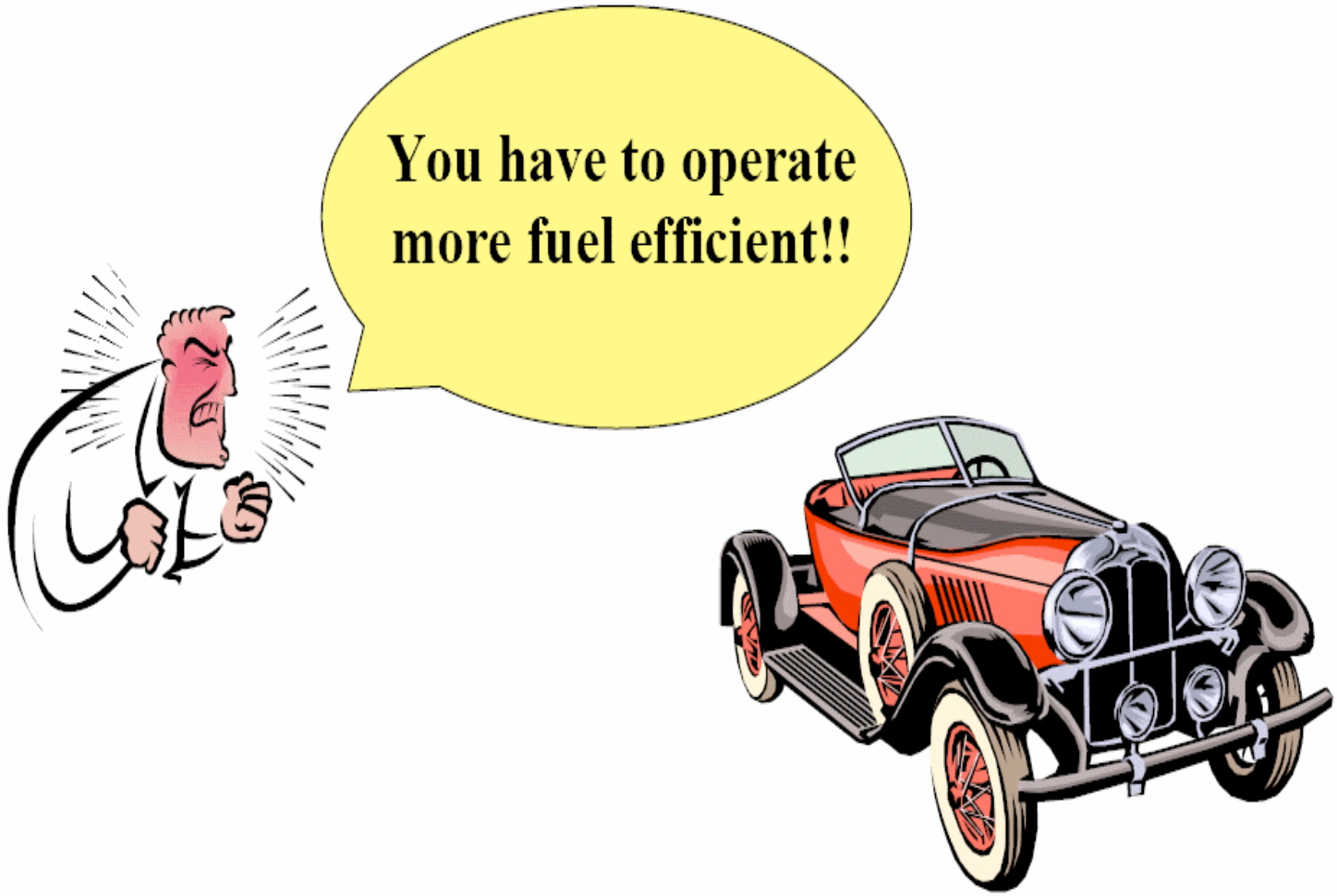
By the ***organization*** of an enterprise is understood the *construction* perspective on the enterprise. It is characterized by the processes in which the products and services are brought about.

Black-box model of an enterprise



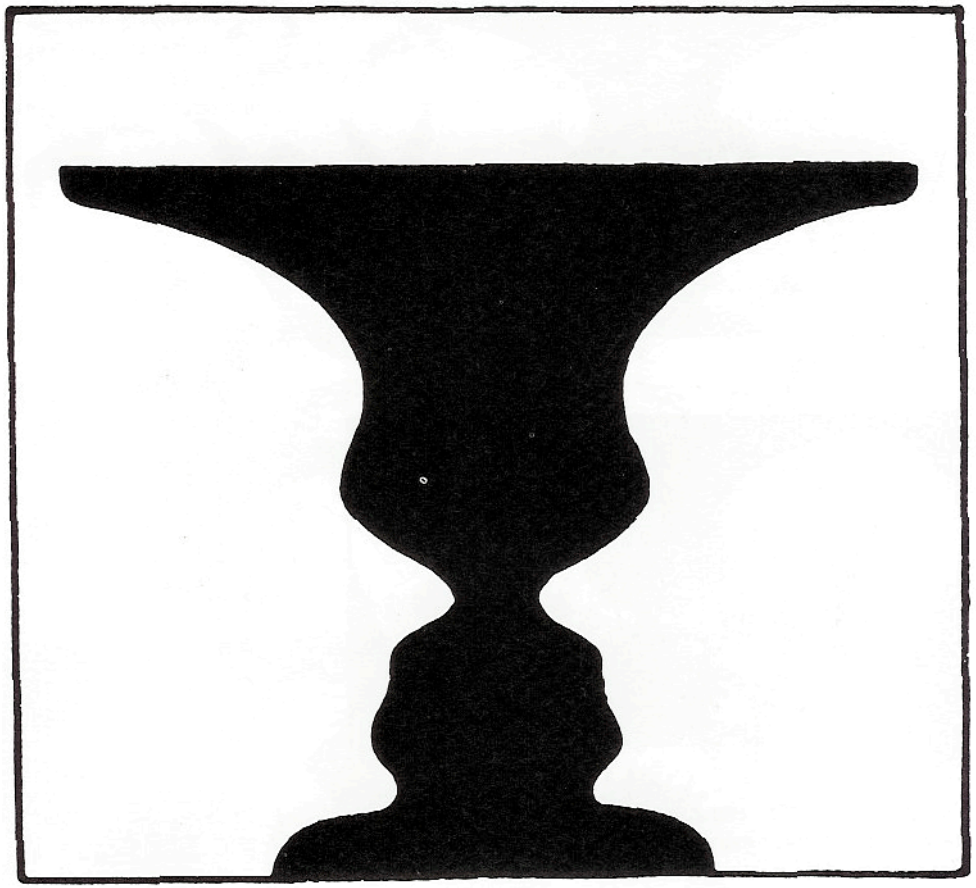


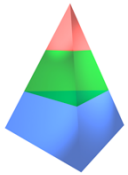
The business view on change



The organization view on change

The white box shapes the black box? [Rubin]

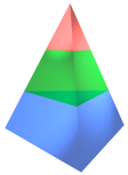




Volley: description (1)

One can become member of the Volley tennis club by sending a letter to the club by postal mail. In that letter one has to mention surname and first name, birth date, sex, telephone number, and postal address (street, house number, zip code, and residence). Charles, the administrator of Volley, empties daily the mailbox and checks whether the information provided is complete. If not, he makes a telephone call to the sender in order to complete the data. If a letter is completed, Charles adds an incoming mail number and the date, records the letter in the letter book, and archives it.

Every Wednesday evening, Charles takes the collected letters to Miranda, the Secretary of Volley. He also takes the member register with him. If Miranda decides that an applicant will become member of Volley, she stamps 'new member' on the letter and writes the date below it. This date is the commencement date of the membership. She then hands the letter to Charles in order to add the new member to the member register. This is a book with numbered lines. Each new member is entered on a new line. The line number is the member number, by which the new member is referenced in the administration.



Volley: description (2)

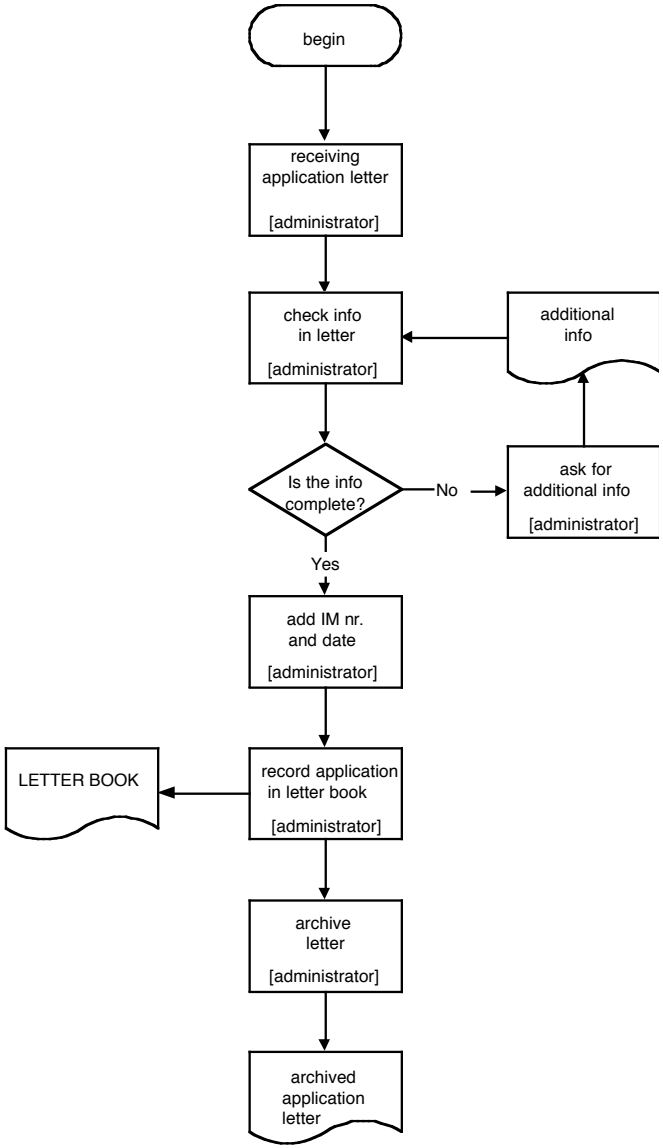
Next, Miranda calculates the membership fee that the new member has to pay for the remaining part of the calendar year. She finds the amount due for annual fees, as decided at the general meeting, on a piece of paper in the drawer of her desk. Then, she asks Charles to write down the amount in the member register.

If Miranda does not allow an applicant to become a member (e.g., because he or she is too young or because the maximum number of members has been reached), Charles will send a letter in which he explains why the applicant cannot (yet) become a member of Volley.

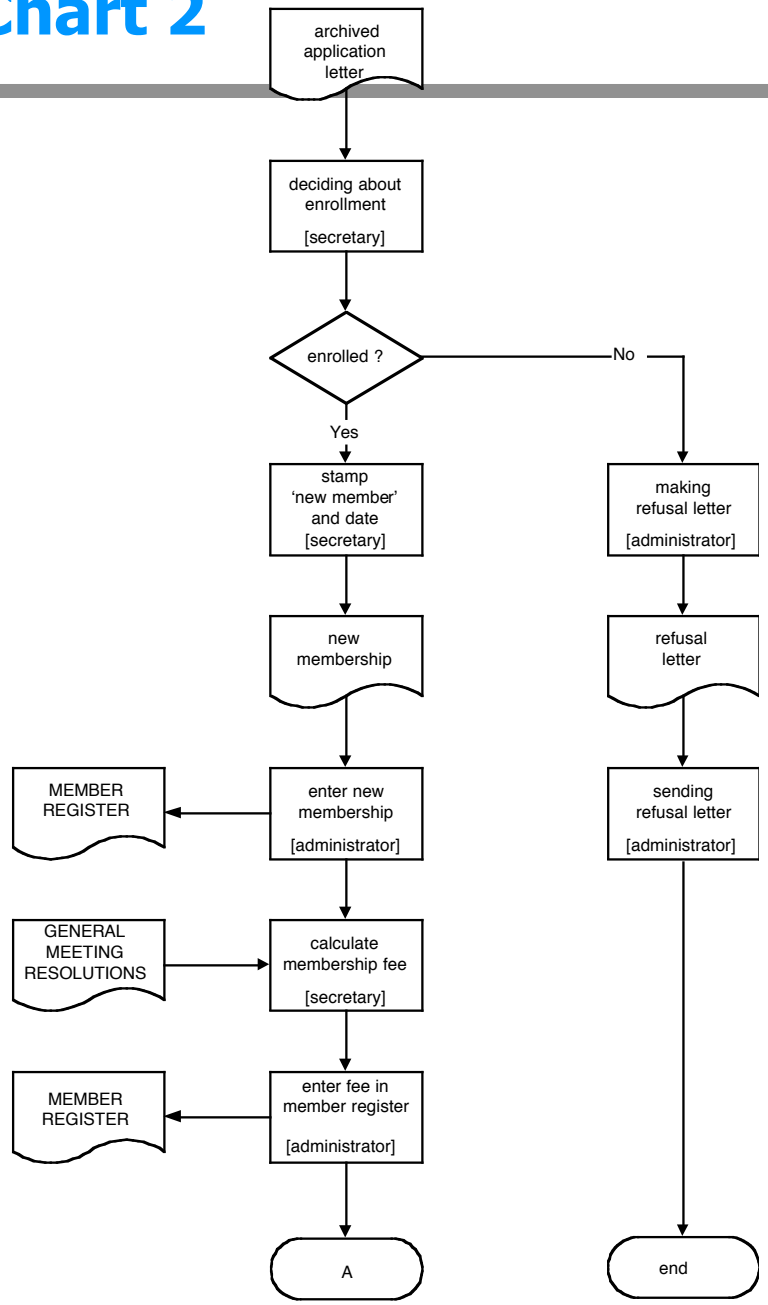
If all applications are processed, Charles takes the letters and the member register home and prepares an invoice to all new members for the payment of the first fee. He sends these invoices by postal mail. Payments have to be performed by bank transfers.

As soon as a payment is received, Charles prints a membership card on which are mentioned the membership number, the commencement date, the name, the birth date, and the postal address. The card is sent to the new member by postal mail.

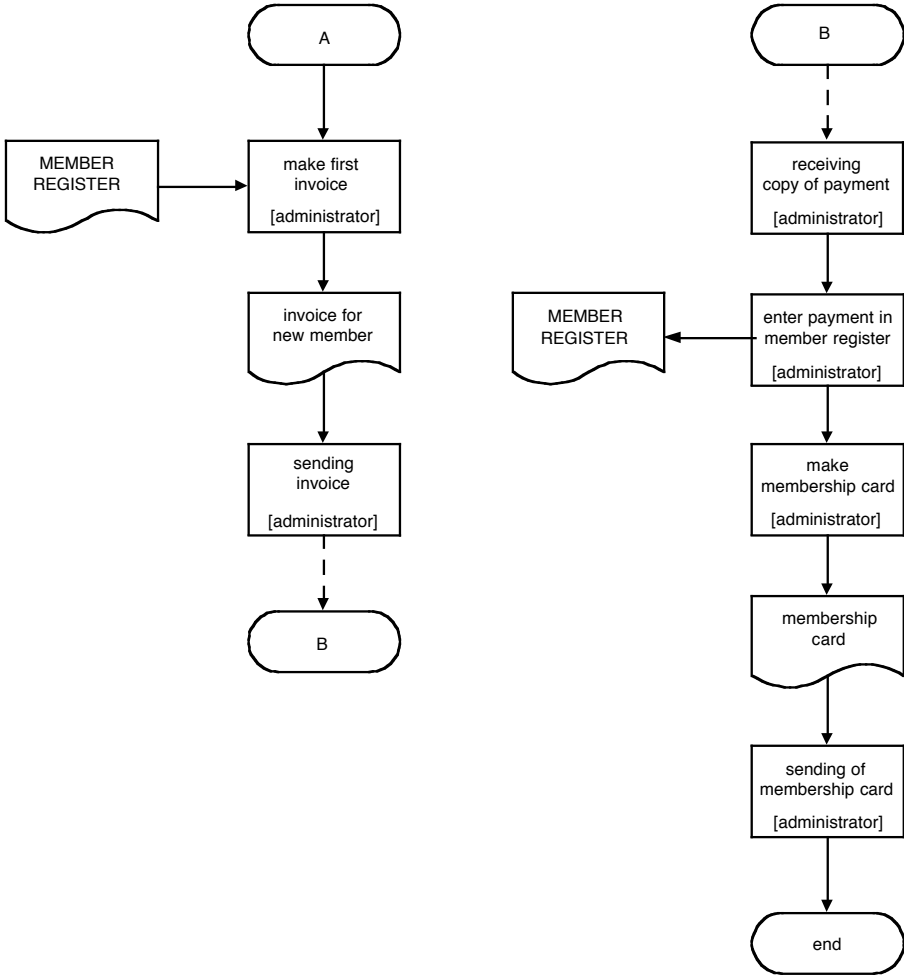
Volley: Flow Chart 1



Volley: Flow Chart 2



Volley: Flow Chart 3

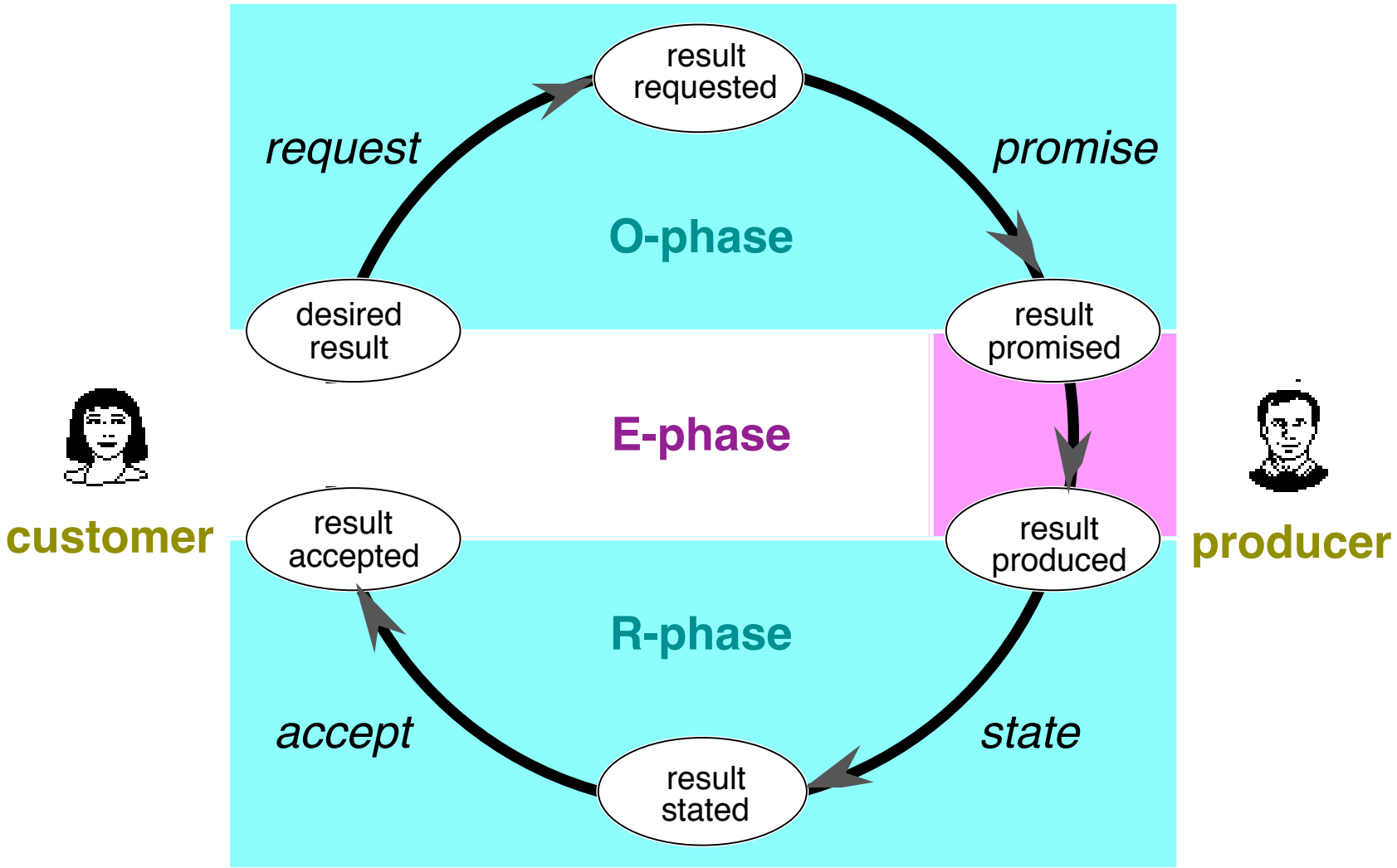


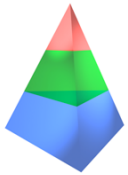
The Operation Axiom

COORDINATION **ACTOR ROLES** **PRODUCTION**

RESPONSIBILITY **AUTHORITY** **COMPETENCE**

The transaction axiom





Example of a transaction

Order phase

I'd like to have such a bouquet
Very well, sir

A1 : requests : A2 : person P has a bouquet B
A2 : promises : A1 : person P has a bouquet B

Execution phase : the actual delivery of the bouquet

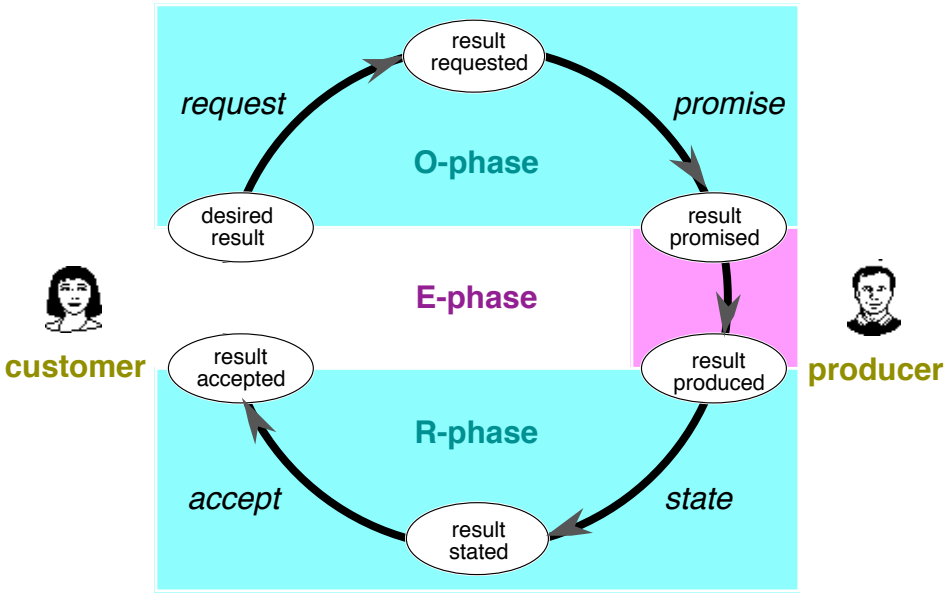
Result phase

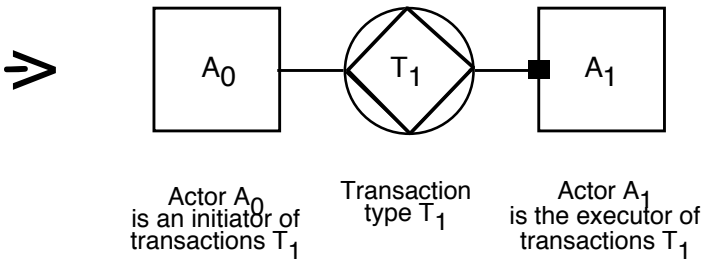
Here you are
Thanks

A2 : states : A1 : person P has a bouquet B
A1 : accepts : A2 : person P has a bouquet B

From Transaction Pattern to Construction Model

The transaction pattern provides a reduction of complexity of at least 70%! (1 transaction represents ≥ 4 actions)





In addition, it reveals all implicit (i.e., non-verbal) and all tacitly performed acts.

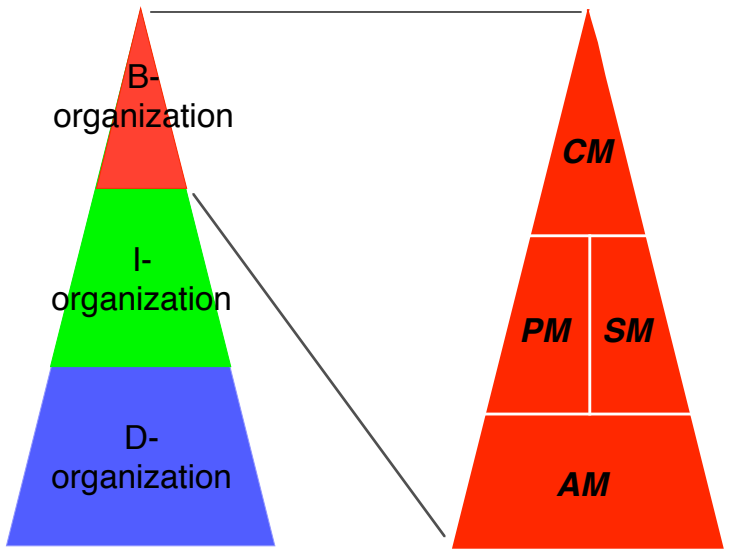
The distinction axiom

COORDINATION	HUMAN ABILITY	PRODUCTION
exposing commitment (as performer) *evoking commitment* (as addressee)	 ***performa***	*ontological action* (deciding, judging)
expressing thought (formulating) *educing thought* (interpreting)	 ***informa***	*infological action* (reproducing, deducing, reasoning, computing, etc.)
uttering information (speaking, writing) *perceiving information* (listening, reading)	 ***forma***	*datalogical action* (storing, transmitting, copying, destroying, etc.)

From distinction axiom to ontology

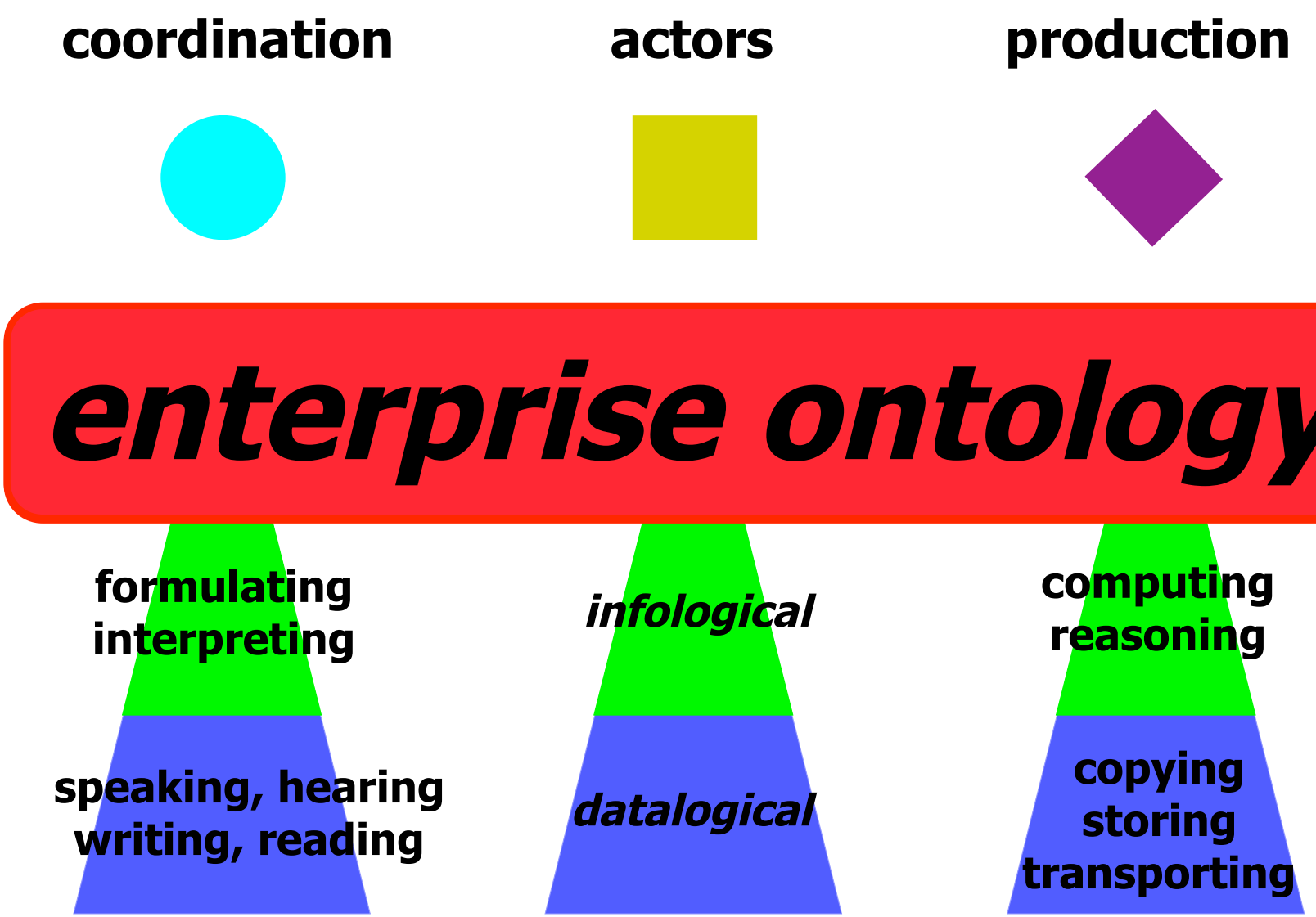
The focus on the B-organization of an enterprise provides for a reduction of complexity of an additional 70%!

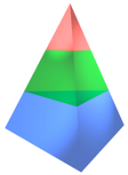
COORDINATION	ACTOR	PRODUCTION
performative	performa	*ontological*
informative	informa	*infological*
formative	forma	*datalogical*
RESPONSIBILITY	AUTHORITY	COMPETENCE



So, cumulated, we have a complexity reduction of ≈ 90%!

Enterprise Ontology - practical definition





Volley: analysis

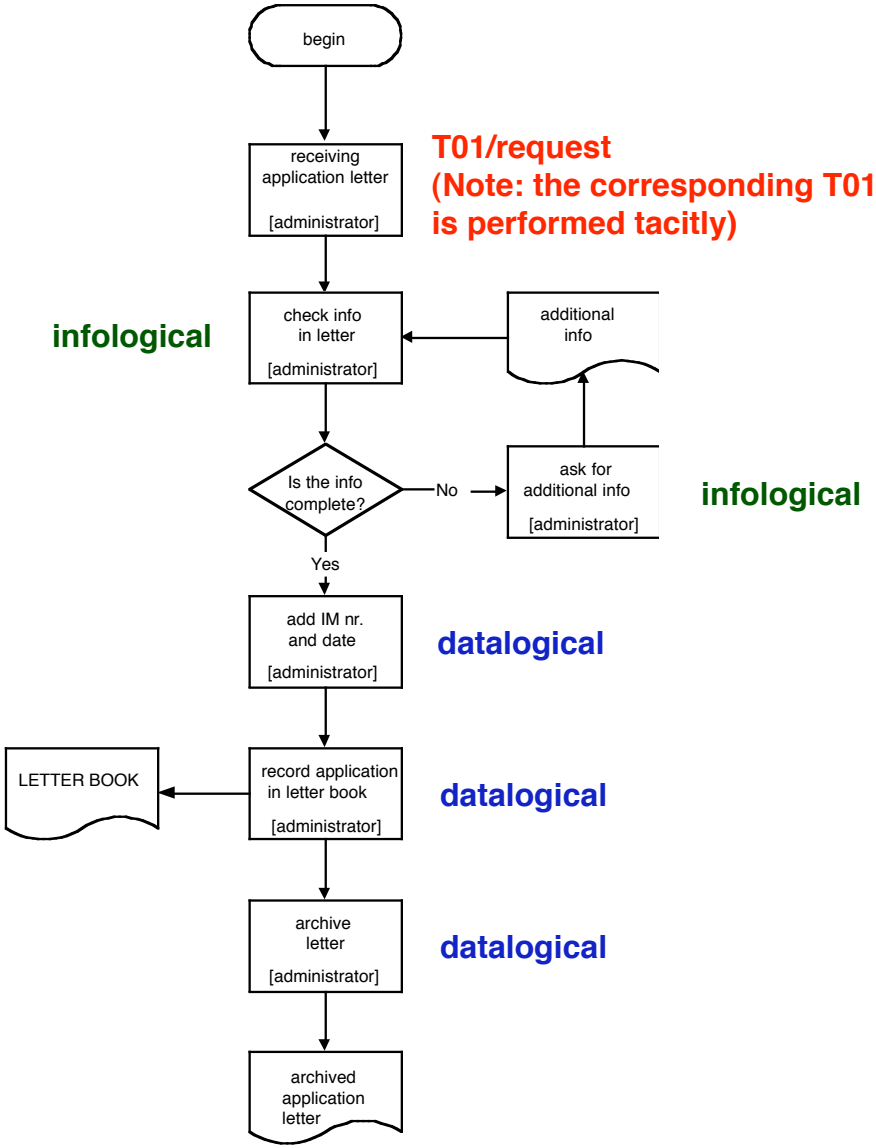
From the Flow Charts and the case description, two transaction types are identified:

T01 membership_start
T02 membership_payment

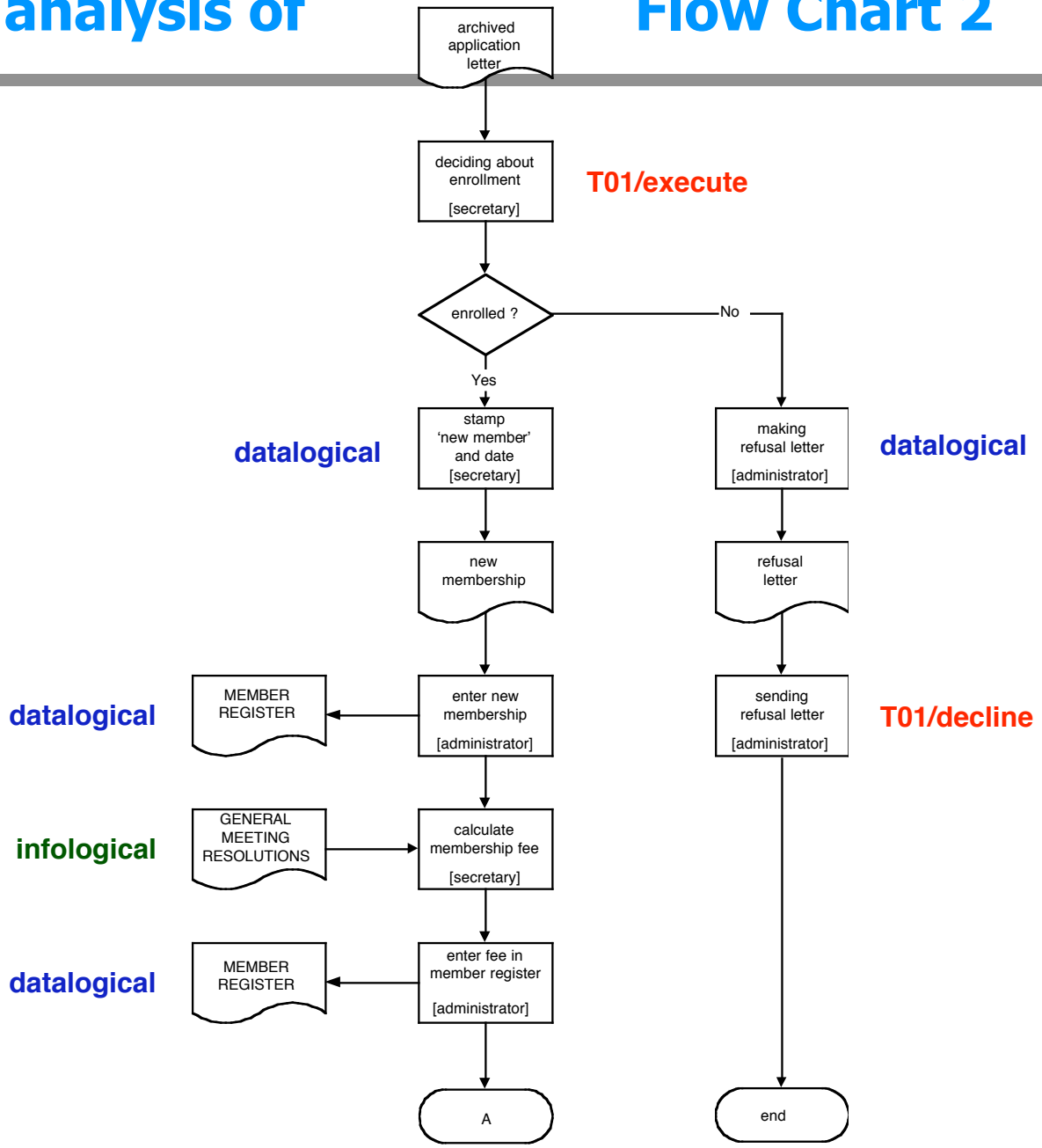
The corresponding production result types are:

R01 *membership M has been started*
R02 *the first fee for membership M is paid*

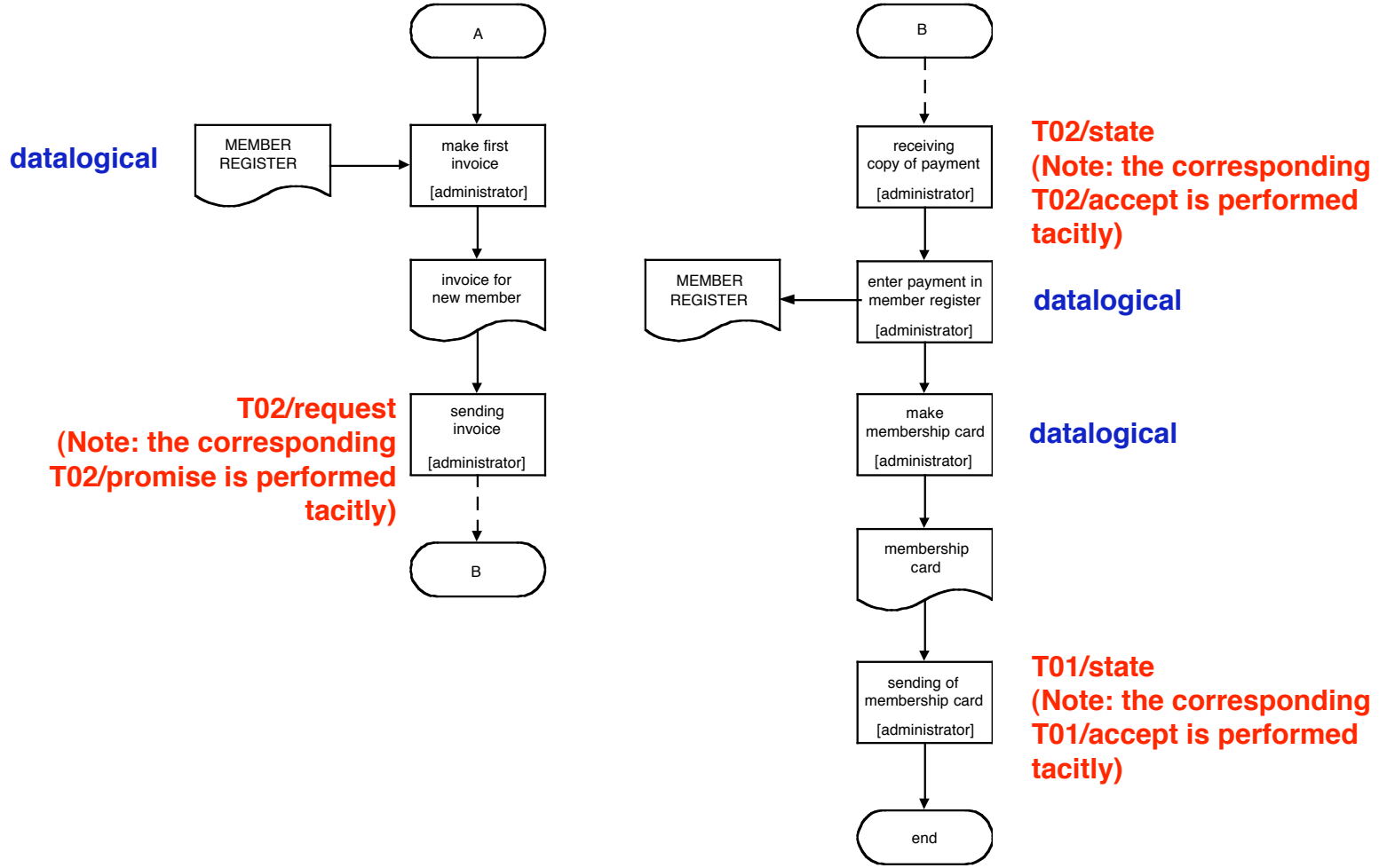
Volley: analysis of Flow Chart 1



Volley: analysis of Flow Chart 2



Volley: analysis of Flow Chart 3



datalogical

T02/request
(Note: the corresponding T02/promise is performed tacitly)

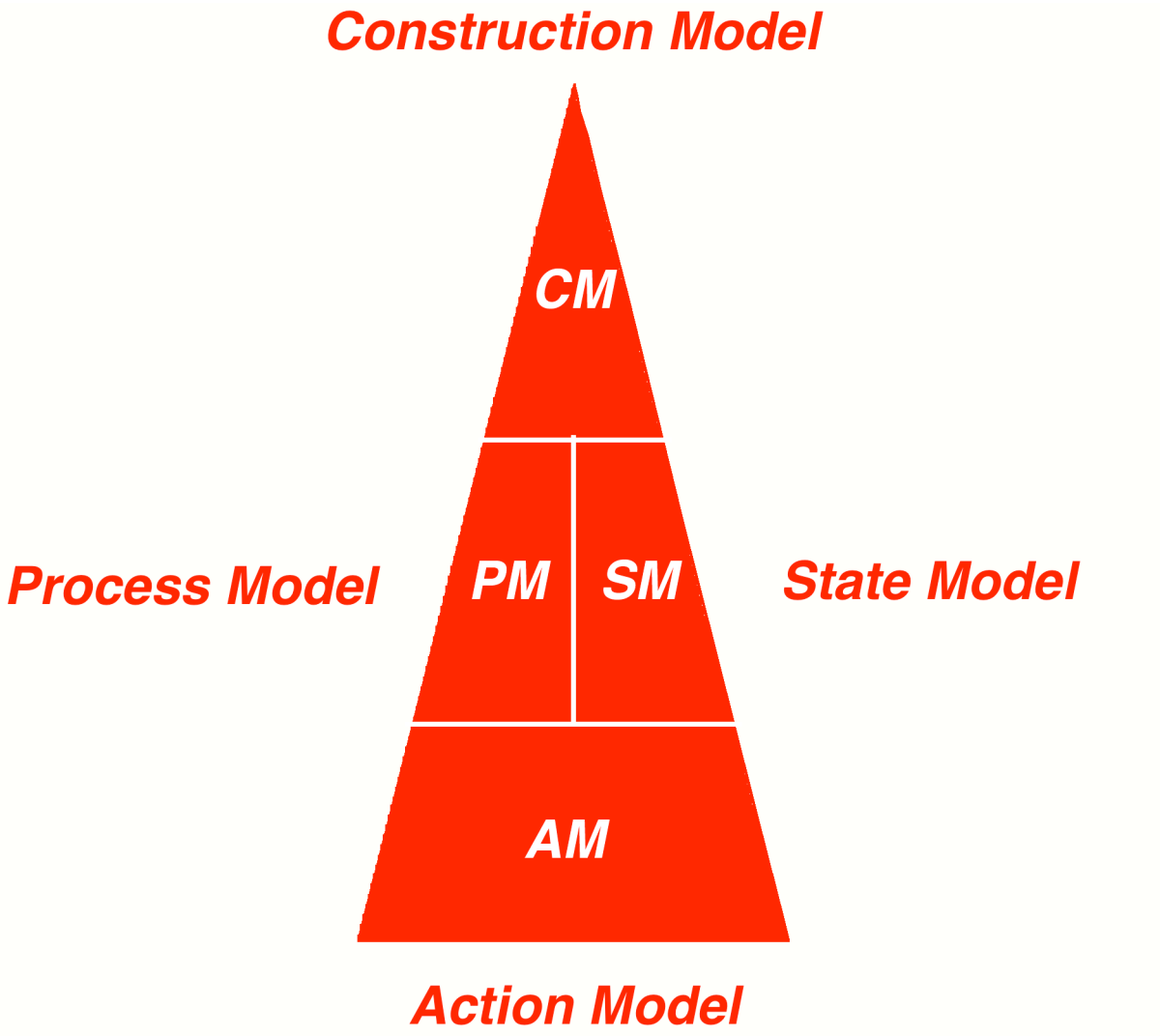
T02/state
(Note: the corresponding T02/accept is performed tacitly)

datalogical

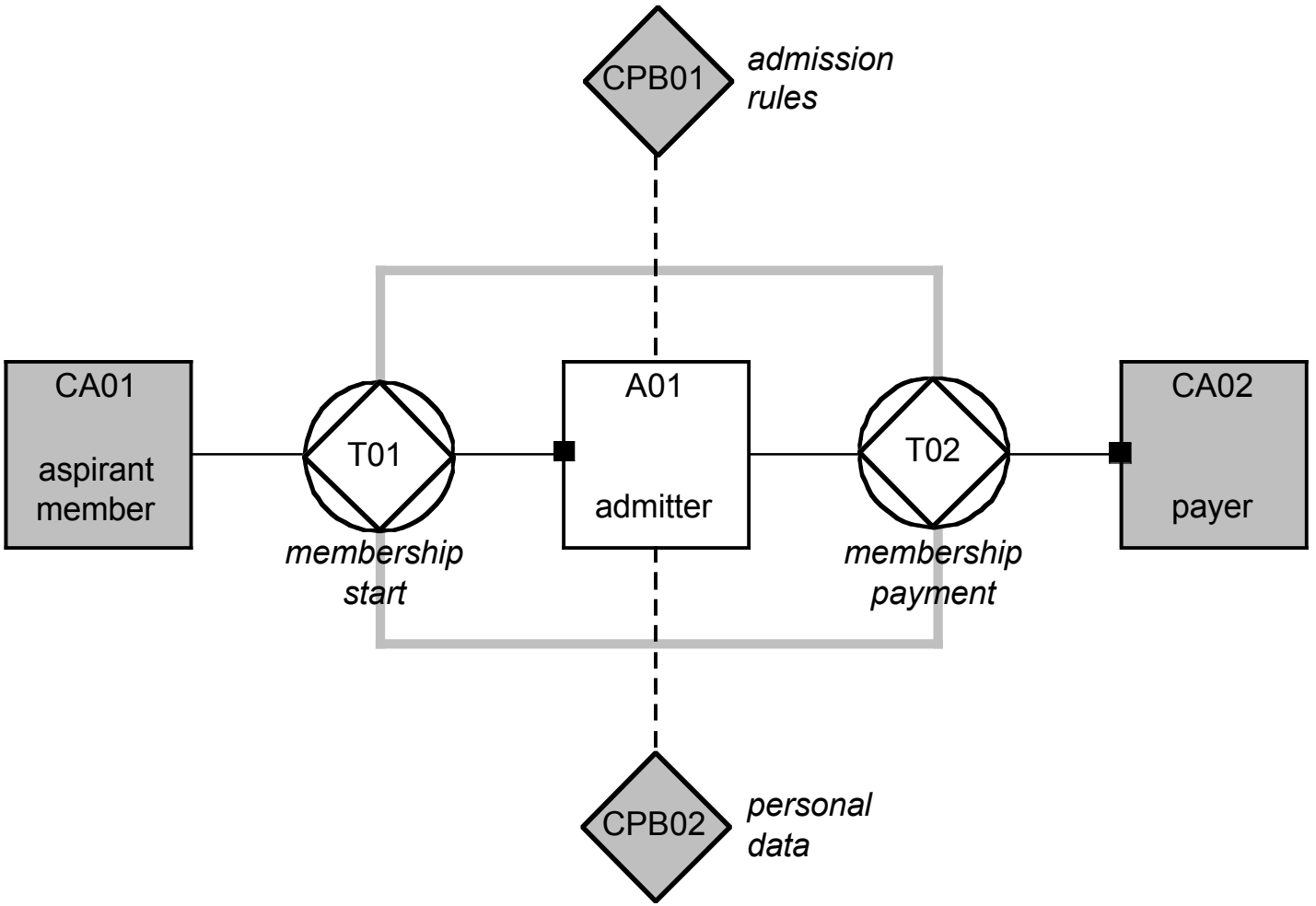
datalogical

T01/state
(Note: the corresponding T01/accept is performed tacitly)

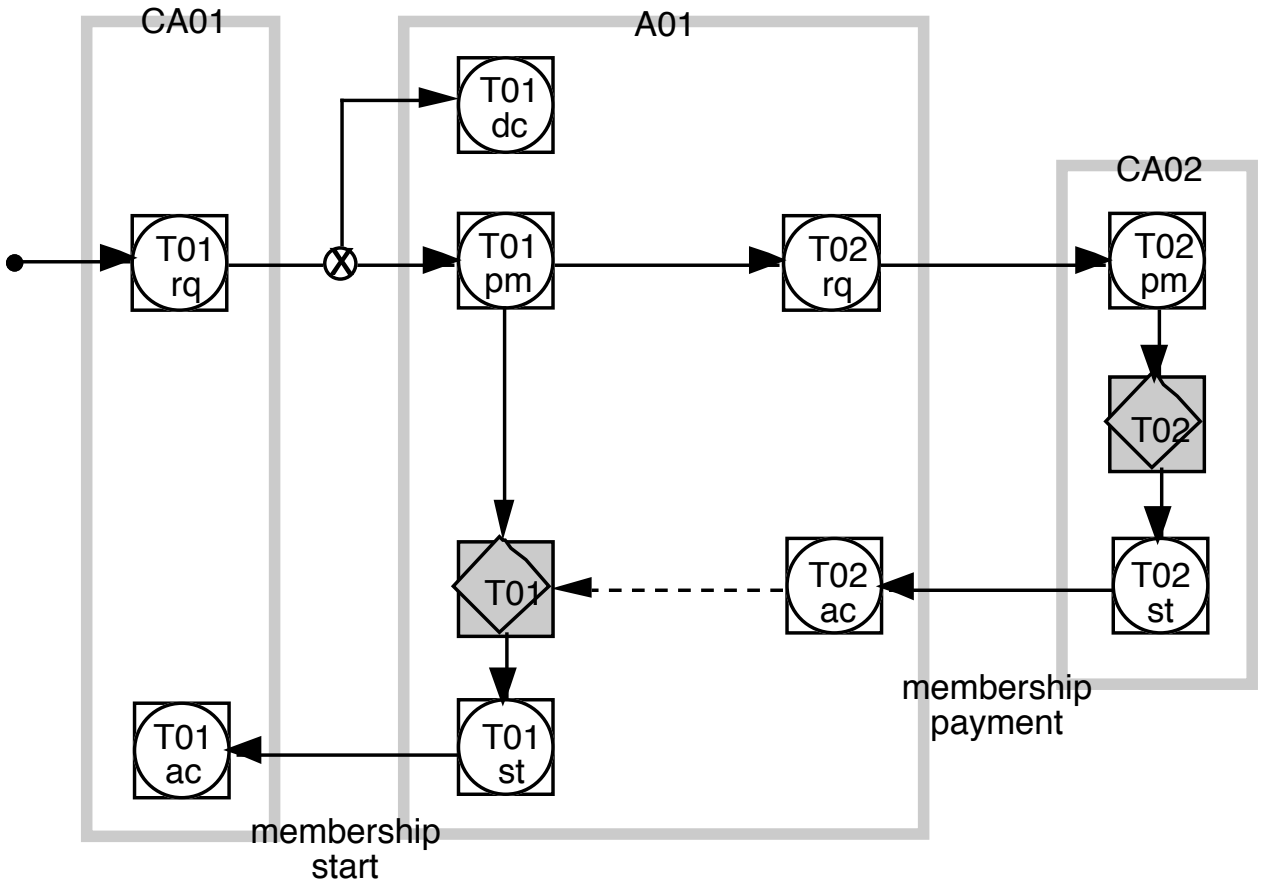
The aspect models

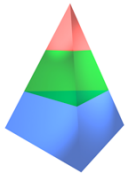


Volley: Ontological Construction Model



Volley: Ontological Process Model

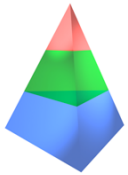




Volley: Ontological Action Model (1)

on requested T01(M) **with** member(new M) = P
    if age(P) < minimal_age **or**
        #members(Volley)=maximum_number(current_year) →
        decline T01(M)
    ◇ age(P) ≥ minimal_age **and**
        #members (Volley) < maximum_number(current_year) →
        promise T01(M)
    fi
no

on promised T01(M)
    request T02(M) **with** first_fee(M)
no



Volley: Ontological Action Model (2)

on stated T02(M)

if *payment is acceptable* → accept T02(M)

◊ **not** *payment is acceptable* → reject T02(M)

fi

no

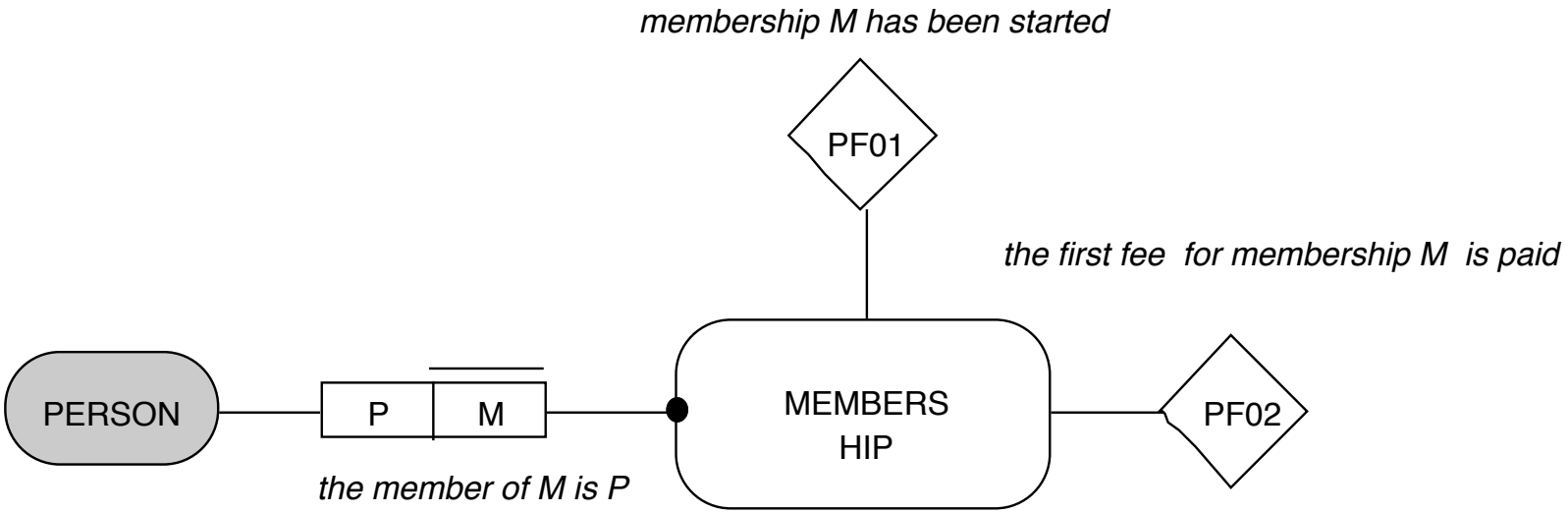
on accepted T02(M)

execute T01(M)

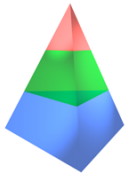
state T01(M)

no

Volley: Ontological State Model (1)



property	domain	range
minimal_age	VOLLEY	AGE
annual_fee	VOLLEY	EURO
maximum_number	VOLLEY	NUMBER
number_of members (*)	VOLLEY	NUMBER
first_fee (*)	MEMBERSHIP	EURO
date_of_birth	PERSON	DATE
age (*)	PERSON	AGE



Volley: Ontological State Model (2)

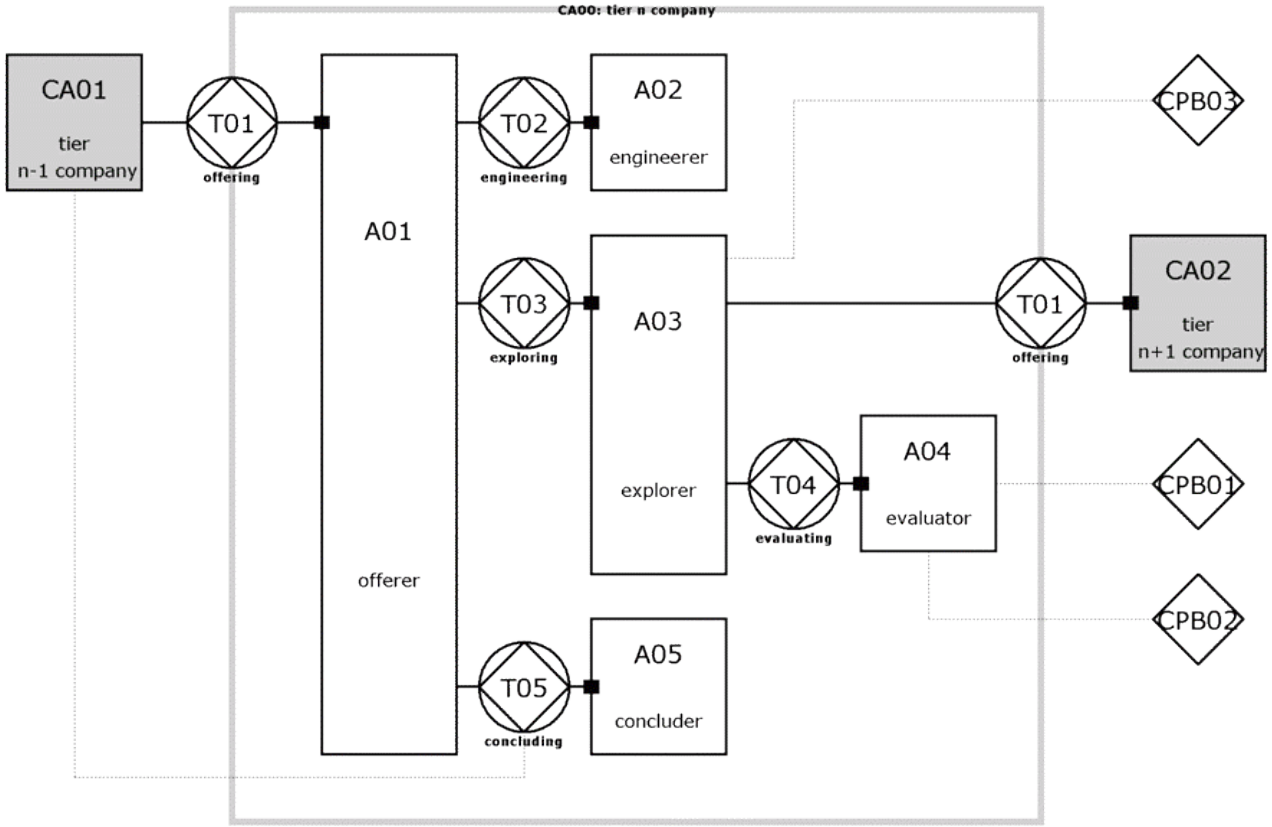
#members (Volley) = *number of persons for which there exists currently a membership*

first_fee (M) = ((12 - Current_Month#)/12) * annual_fee(Current_Year)

age(P) = *difference in years between birth year of P and current year*

Supply Chain - Construction Model

Actor Transaction Diagram



Transaction Result Table

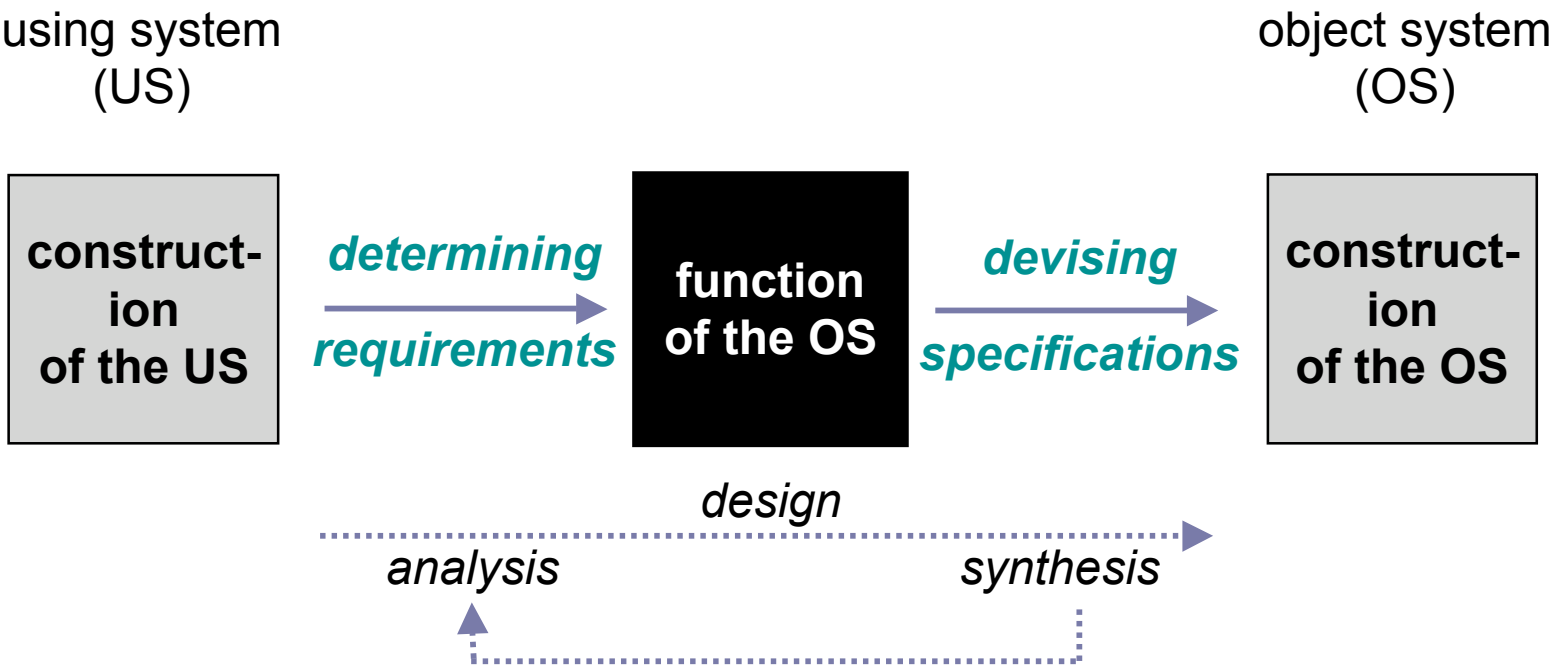
transaction type	resulting P-event type
T01 offering	PE01 supply contract C is offered
T02 engineering	PE02 the BoM of assembly A is determined
T03 exploring	PE03 supply contract C is a potential contract
T04 evaluating	PE04 supply contract C is evaluated
T05 concluding	PE05 supply contract C is concluded

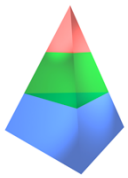
Enterprise Engineering

Enterprise Architecture

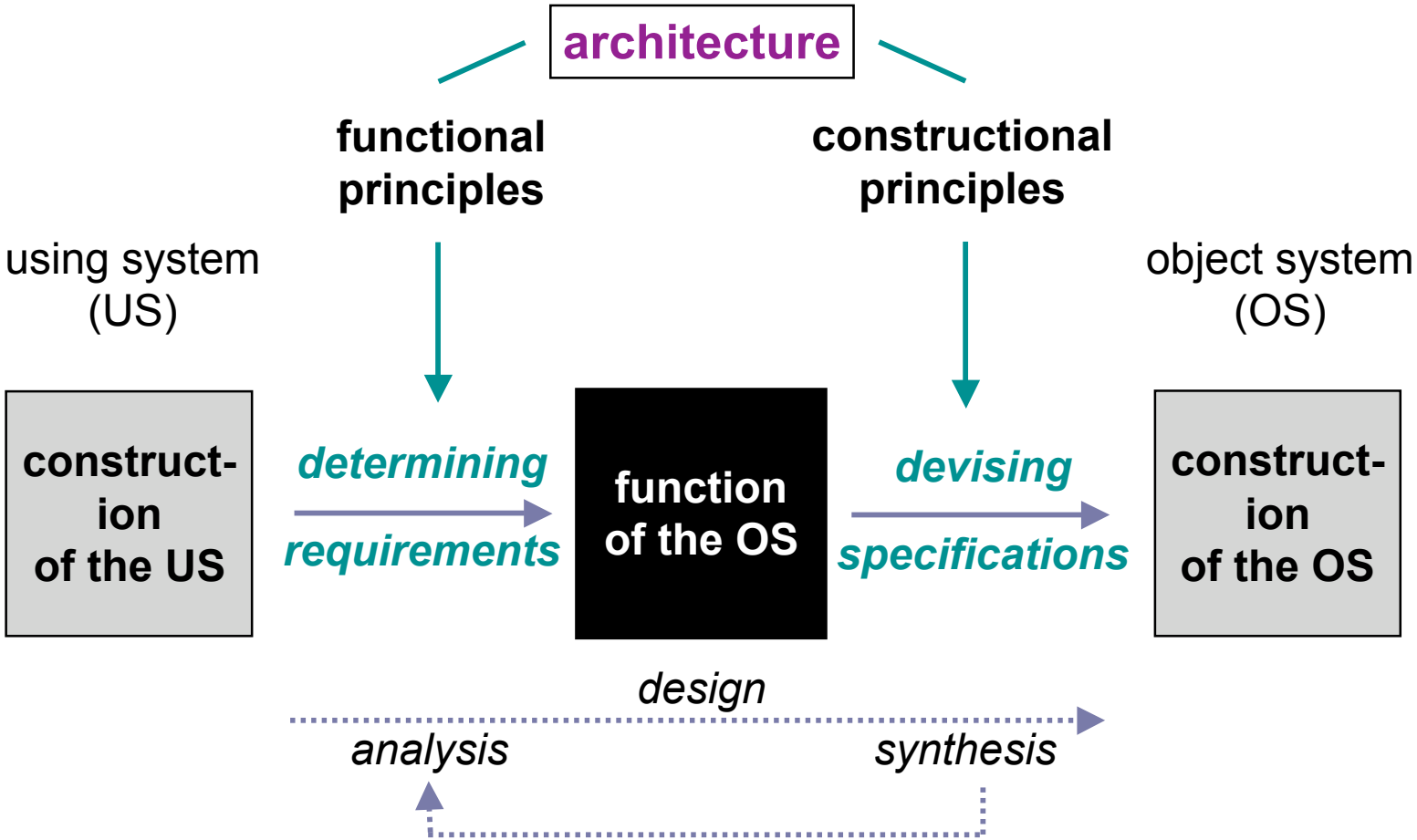
Jan L.G. Dietz
TU Delft

The System Design Process

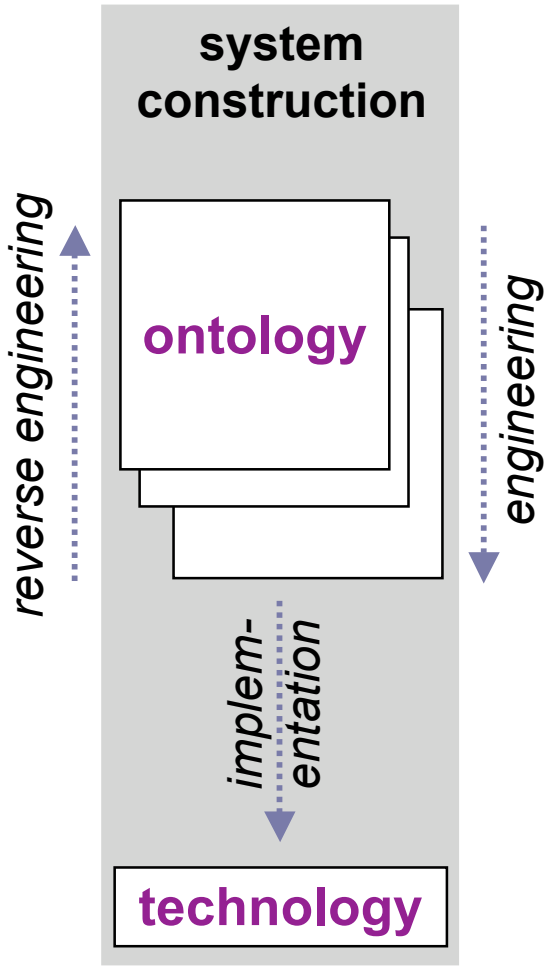




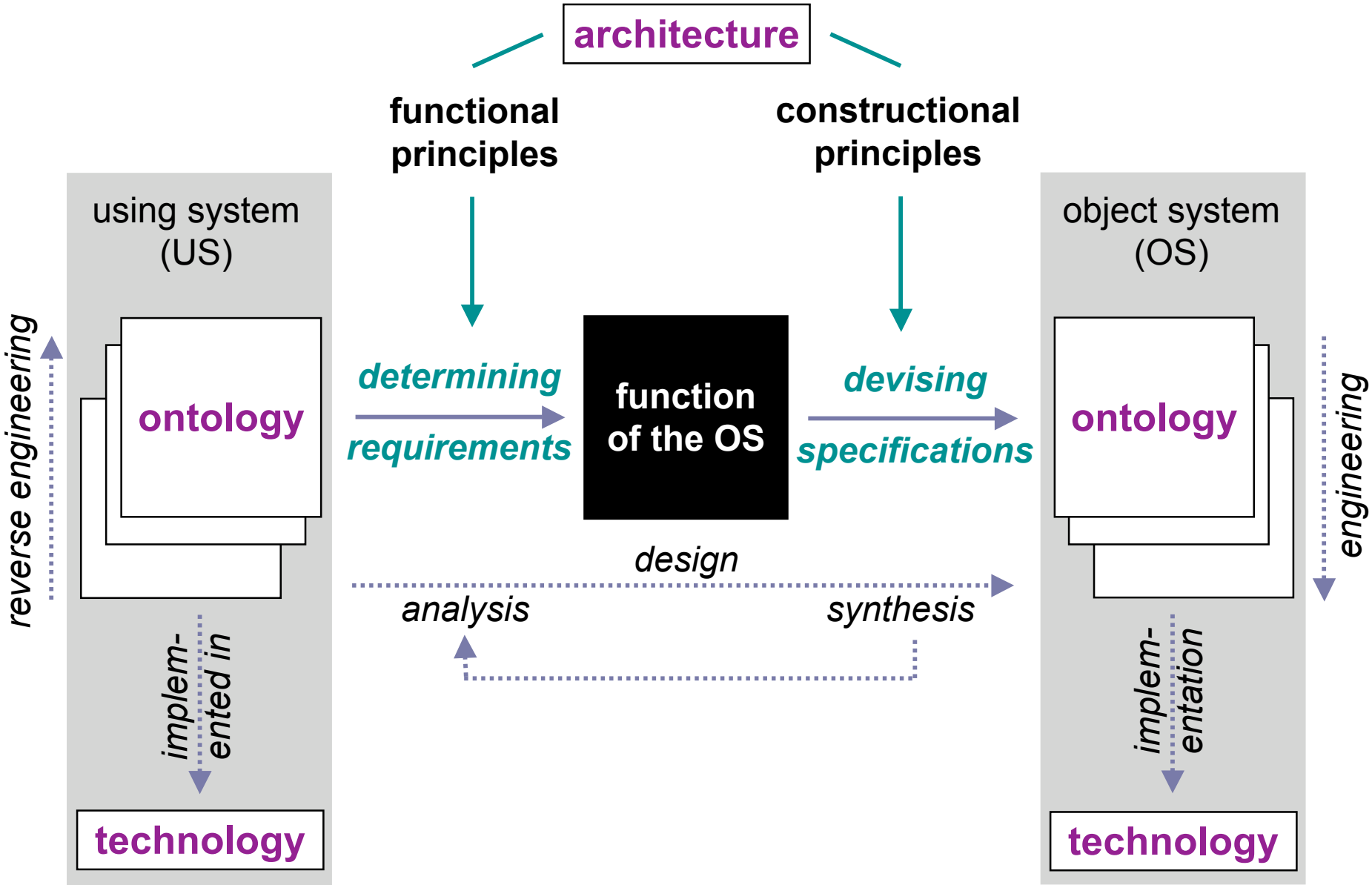
The role of Architecture in System Design

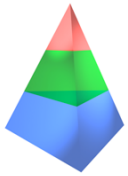


The role of Ontology in System Engineering



The System Development Process

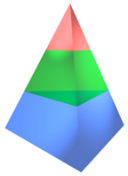




What is an object system?

- By *object system* (OS) we mean the system that is the object of attention.

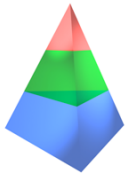
- As the explanatory example of the OS we take an accounting information system. One of the services of the OS is the delivery of the monthly turnover of the enterprise to the US.



What is a using system?

- By *using system* (US) we mean the system that makes use of the functionality (the services) of the object system (OS).

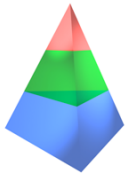
- As the explanatory example of the US we take the sales department of an enterprise. One of the services that the US takes from the OS is the delivery of the monthly turnover of the enterprise.



What is the technology of the US?

- By *technology* we mean the technological means by which a system is implemented.

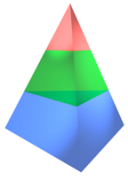
- Regarding an enterprise, and thus the example US, the technological means are human beings for fulfilling actor roles, and possibly material equipment for performing production acts.



What is the technology of the OS?

- By *technology* we mean the technological means by which a system is implemented.

- Regarding an information system, and thus the example OS, there is a choice in technological means. One could apply human beings but also ICT-systems.

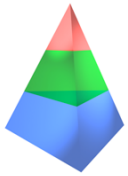


What is the ontology of the US?

- By *ontology* (or ontological model) is meant the 'highest' constructional model of a system. It shows the essential construction and operation of the system, fully independent of its implementation.

- As the meta ontology for enterprises, we adopt the CRISP model.

- The ontology of the sales department consists of the transaction types and the actor roles that constitute the department. One of the fact types that are produced is the sale (instance). One of the fact types that are used is the monthly turnover.

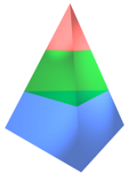


What is the ontology of the OS?

- By *ontology* (or ontological model) is meant the 'highest' constructional model of a system. It shows the essential construction and operation of the system, fully independent of its implementation.

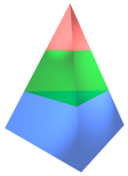
- As the meta ontology for information systems, we adopt the SMART model.

- The ontology of the accounting information system consists of the action channels, the fact banks and the processor roles that constitute the system. One of the fact types that it produces is the sum of a number of numerical values (interpreted by the US as the monthly turnover).



What is implementation?

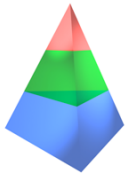
- By the *implementation* of a system we mean the assignment of technological means to the elements of its implementation model, i.e., the lowest level constructional model of the system, such that it can be put into operation.



What is reverse engineering?

- By *reverse engineering* we mean the activity of reconstructing the ontological model (and if necessary also the intermediate models) of a system from its implementation model (or from one of the intermediate models). Reverse engineering is only necessary if the ontological model is missing.

- If the implementation model is also missing, it is necessary to observe and analyze the implemented system.

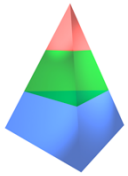


What is engineering?

- By *engineering* we mean activity of constructing the implementation model of a system from its ontological model.

- Usually a number of intermediate models are constructed before arriving at the implementation model. In each step the current 'lowest' model is transformed to a new 'lowest' model. In each step, implementation related details are added to the previous model.

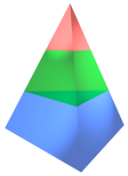
- The engineering of a system is governed by the construction architecture that is applicable.



What is a functional model?

- By *functional model* we mean the model of a system in which its functionality is specified. The functionality of the OS consists of the services it can deliver to the US.

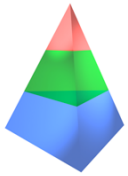
- These services are expressed in the terminology of the (ontological model) of the US. In the example OS (the accounting information system), one of the services is the delivery of the monthly turnover to the US.



What is determining requirements?

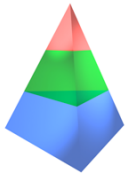
- By *determining requirements* we mean the activity that results into the functional model of an OS on the basis of the ontological model of the US, taking into account the functional principles in the applicable (function) architecture.

- Often a distinction is made between functional and non-functional requirements. By the latter is meant the performance characteristics with which the services are delivered by the OS. These include response time, availability, security etc. (Note the confusion in using the terms “functional” and “non-functional”; both pertain to the functional model.)



What is devising specifications?

- By *devising specifications* we mean the activity that results into the set of constructional requirements of an OS on the basis of its functional model, taking into account the constructional principles in the applicable (construction) architecture.

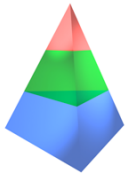


Constructional models

- A *constructional model* is a white-box model of a system. Generally, the process of engineering a system results in an ordered set of constructional models.

- The 'highest' one, i.e. the one that is most abstracted from implementation, is called the ontological model.

- The 'lowest' one, i.e. the one from which the system is or can be implemented, is called the implementation model.

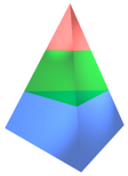


What is design?

- By *design* we mean the activity that starts from the ontological model of the US and results into the ontological model of the OS.

- The design process consists of two logically consecutive phases: the *analysis* phase, called "determining requirements" and the *synthesis* phase, called "devising specifications".

- Generally, these phases are passed through incrementally and iteratively.

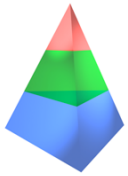


What is architecture?

- By *architecture* we mean conceptually the *normative restriction of design freedom*. The fact that the design freedom of designers is generally too large, is the rationale for applying architectures.

- Operationally, architecture is a consistent and coherent set of *design principles* that embody general requirements. Applying a design principle *satisfies* one or more *general requirements*.

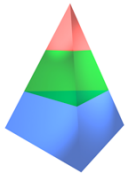
- General requirements hold for a class of systems instead of for one system specifically. For the accounting information system that we have taken as the example OS, there will be general and special requirements.



Functional principles

- The principles in an architecture can be divided into functional principles and constructional principles.

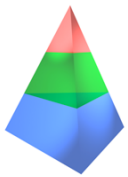
- *Functional principles* regard the development of the functional model of the OS. They have to be taken into account during the design phase "determining requirements".



Constructional principles

- The principles in an architecture can be divided into functional principles and constructional principles.

- *Constructional principles* regard the development of the constructional models of the OS. They have to be taken into account during the design phase "devising specifications" and during the engineering of the OS.

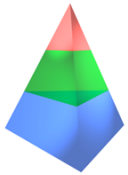


The function perspective

- In the *function* perspective one is concerned with the function and the behavior of a system, thus the (functional) relationship between input and output. Accordingly, a *black-box* model of the system is used.

- The *requirements* of a system regard its *external* function and behavior, including performance issues (the so-called non-functional requirements).

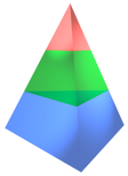
- Function is not identical to purpose. The *function* of a system is an inherent property of the system, whereas *purpose* is a relationship between a system and a stakeholder.



The construction perspective

- In the *construction* perspective one is concerned with the construction and the operation of a system, in particular how the elements in the composition collaborate to deliver services to the environment. Accordingly, a *white-box* model of the system is used.

- The *specifications* of a system regard its *internal* construction and operation, i.e. the interactions between the elements for providing the corresponding (internal) products, as well as the *interface* construction and operation, i.e. the interactions between the composition and the environment for the delivery of (end) products.



Conclusions

- The new point of view of EE is that enterprises are purposefully designed and engineered social systems.
- Enterprise Ontology reveals the essence of an enterprise in a comprehensive, coherent, consistent, and concise way, while reducing complexity by well over 90%.
- Enterprise Architecture guides the (re)design and (re)engineering of an enterprise such that its operation is compliant with its mission and strategy, and with all other laws and regulations.