THE ESSENTIAL SYSTEM MODEL

Jan L.G. Dietz

University of Limburg
Faculty of Economics and Business Administration
P.O.Box 616, 6200 MD Maastricht, The Netherlands

Keywords: conceptual modelling, communication modelling, systems analysis.

Abstract

A particular way of modelling communication in an organization is presented as a new basis for systems analysis. Speech act theory provides the theoretical foundation of this approach. The so-called essential system model is identified as a missing kind of model in systems analysis, the usage of which however could mean a major improvement of current practice. The distinctive properties of essential system models are discussed, and a particular type of dynamic model, called the smartienet, is introduced. The potential impact of the application of smartienet based essential system models is illustrated using a simple business example.

1. INTRODUCTION

1.1. Definition of the problem

In the Business Analysis or Requirements Engineering phase of the systems development cycle, one is confronted with the problem of modelling an organization in such a way, that the more or less accidental realization aspects of the informational activities can clearly be separated from the essential business aspects. Let us call a model of an organization (and indeed of any system) which only shows its essential aspects an *essential* model of the system.

Separating essential aspects from realization aspects is certainly not a trivial task, particularly not if the essential activities seem to be nothing else than the processing of information, like in banks and insurance companies. As will be pointed out in section 3, the common understanding that the primary activities in organizations of this kind are information processing activities, is wrong. Every organization has its own essential activities, which are not just information processing activities.

Traditionally two perspectives on systems (e.g. organizations) are dominant in informational systems analysis: the so-called data perspective and the so-called process perspective.
The *data perspective* basically implies a static view upon the system, it is concerned with the definition of the elements constituting a state of the system as well as with the definition of the set of all possible states, i.e. the state space.
The *process perspective* basically means identifying the relevant operational or active components of the system and their interrelationships. A description of the operation or behaviour of these components usuallly is also taken to be part of the process perspective.
For the purpose of this paper it makes sense to distinguish between two levels of abstraction in both perspectives: the logical level and the essential level. Combining these levels with the two perspectives gives rise to four kinds of conceptual models (cf. figure 1).

	data perspective	*process perspective*
essential	**semantic data models**	**?**
logical	**logical structure models**	**logical flow models**

Figure 1. Categories of conceptual models in informational systems analysis

Characteristic for models at the *logical* level is that they abstract from any particular physical realization, while concentrating on the (logical) aspects which are common to all physical realizations. A logical model in the *data* perspective describes all information items which play a role in the system under concern, including their interrelationships (data structure). A well-known metamodel for this kind of models is the Relational Model. A logical model in the *process* perspective describes all activities in the system as well as the information flows, and if desired also the material flows, which interconnect them. Well-known metamodels for this kind of models are the Activity Graph (A-graph) from the ISAC methodology [9] and the Data Flow Diagram (DFD) from the Structured Analysis approaches (cf. e.g. [12]).

Characteristic for models at the *essential* level is that they abstract from any particular logical realization, while concentrating on the (essential) aspects which are common to all logical realizations. The question of course is what it means to abstract from any particular logical realization, and thus to seek for genericness in logical models. For a model in the data perspective it would mean that only the objects, including their properties, to which information items refer are described, but not the particular ways in which references and descriptions are made. It is with some hesitation that the semantic data models are mentioned in figure 1 as examples of metamodels for this kind of model, because many representatives of this class do not seem to reach the degree of abstraction which is necessary in order to be genuinely generic with regard to the logical level. Although essential data models are not the subject of this paper, some explanation seems to be appropriate.

Firstly, it should be stressed that an essential data model is a model of the objects to which information items refer. For example, if a property of a book is that it has one or more authors, then there is a set of persons associated with a book, not a set of person names. Names and other identification attributes definitely belong to the logical level. Secondly, although names are irrelevant, the notion of identity of an object is very important. All properties or components of an object may be variant, its identity however remains invariant. Once more, the identity of an object is not e.g. a social security number or a product bar code; these are just attributes, and thus principally variant.

It is the purpose of this paper to discuss what an essential model from the process perspective would be like, and what the relationship is with the logical flow model. Apparently, this class of essential models is empty, as the question mark in figure 1 indicates. We call this kind of model an essential system model, and not an essential process model for two reasons. The first is that the term 'process' is used wrongly in our field, as we discussed elsewhere [3]. The second reason is that this term usually is associated with informational processes, and of all things the contents of an essential system model is not informational processes.

To illustrate the thoughts to be developed in the next sections, we take as an example system a part of the operations of a mail order company (MOC). The narrative description (on the logical level) of the distinct operational functions hereafter applies to this system.

The Acceptance department receives orders from customers. Its main task is to accept or reject them. The acceptance or rejection of a customer order is reported to the Customer service department. In case of acceptance an internal order is sent to the Warehouse department. This information is also made known to Customer Service.

The Customer Service department is called into being in order to keep up reporting to customers and to respond quickly to all kinds of questions from them. It is informed by the Acceptance department about acceptance and rejection of orders, and by the Warehouse department about the planned and performed shipments to customers. As soon as the acceptance or rejection of an order is made known by Acceptance, a confirmation (or rejection) letter is sent to the customer.

The Warehouse department tries to deliver accepted customer orders as soon as possible. Actual stock information is received from Stockkeeping. If there is not sufficient stock to deliver immediately after acceptance, a note is sent to Customer Service indicating the planned

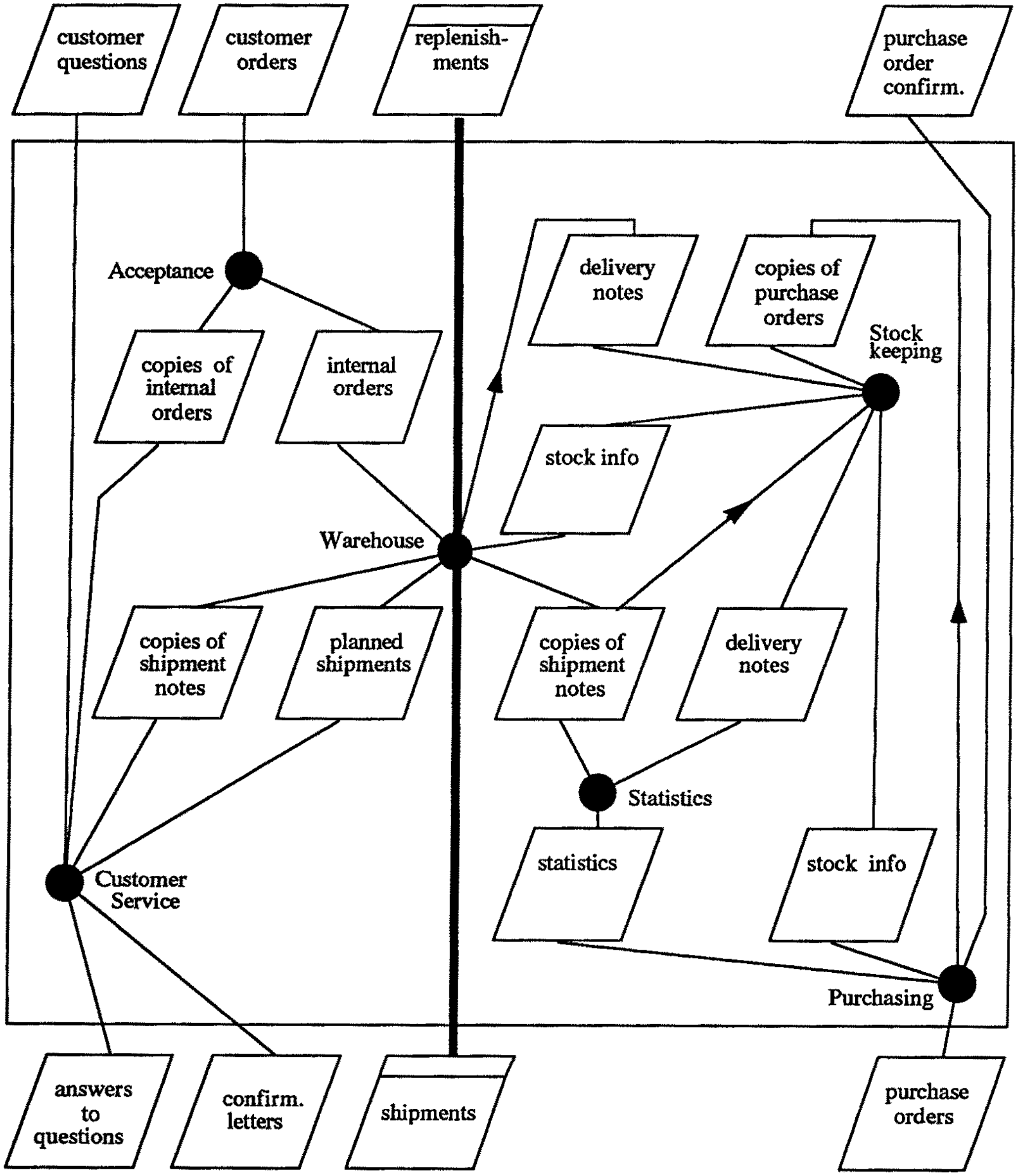

Figure 2. A-graph of the Mail Order Company

shipping date. When ordered goods are shipped, a copy of the Shipment Note is sent to Customer Service, such that this department disposes of up to date information about customer orders. This information is made available to the Statistics department too. The Warehouse department also receives replenishments delivered by suppliers. If a replenishment is received, the delivery note is sent to Stockkeeping.

The Stockkeeping department receives delivery notes from the Warehouse. After being processed these are sent to Statistics. The Stockkeeping department also receives copies of purchase orders and of shipment notes. On the basis of these three information flows it is able to provide Warehouse and Purchasing with adequate stock information.

The Purchasing department periodically inspects the stock levels and decides upon ordering replenishments. Statistical information about past orders and shipments, such as product group dependent trends and seasonal fluctuations, is received from the Statistics department. Copies of Purchase Orders are sent to the Stockkeeping department.

Suppliers of course also have some acceptation procedure with respect to purchase orders. The acceptation results are made known to the Purchasing department.

The Statistics department produces several kinds of statistical information based on actual shipments and deliveries.

Figure 2 exhibits an A-graph representation of the mail order company.

1.2. Rationale and outline

A good feeling of the nature of systems analysis in the field of business systems can be obtained from studying the Change Analysis phase of the ISAC methodology [9]. The main task of the systems analist in this phase is to propose a new logical design of the system under consideration, given the logical model of the present situation and a statement of the problems which have to be solved (cf. figure 3). The difficulty of this task can be compared to the difficulty of the task of generating a new logical data model, given only the present one and a list of new information requirements, but without having at one's disposal an essential, i.e. a semantic, data model.

One therefore may seriously doubt the 'optimality' of the proposed new situation when following the current procedure, even if the analysis has been performed by experienced analists. In order to exploit fully the possibilities of modern information technology in the design of organizations (and this is what information systems engineering is all about!), knowledge of the essential, i.e. information technology independent, aspects of an organization is indispensable. Only this knowledge provides the analist with the desired maximum design freedom at the logical level, and subsequently at the physical level. So, if one agrees with this conjecture, one better makes sure that an explicit account of that knowledge is available. The *Essential System Model*, as discussed in this paper, constitutes such an account.

We have found the term 'essential model' also in [Yourdon, 1989]. However, the notion covered by the term there is rather different from ours. In fact, it is nothing more than a genuine logical model. Apparently, there was a need to distinguish the usage of DFD's for this (logical) purpose from the common rather physical application. Although the discovery by Yourdon of the logical process model must be welcomed, it is at the same time regrettable that he has labelled it as the essential model.

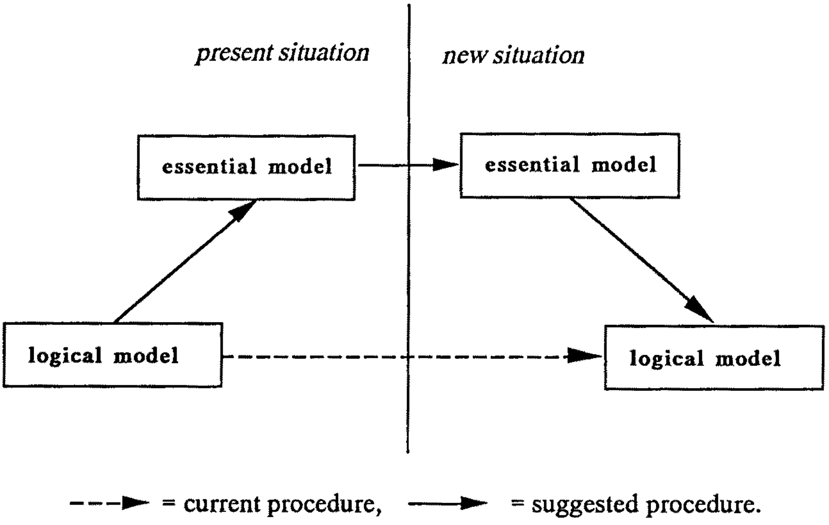

Figure 3. The use of essential models in systems analysis

The research reported about in this paper is carried out in the context of the SMARTIE project at the University of Limburg. The overall goal of this project is the search for sound theoretical foundations for the discipline of Information Systems Engineering. A more concrete aim, the one to which this paper contributes, is the development of adequate modelling support to the task of systems analysis. To this end a particular dynamic system concept is developed, called smartie, which is discussed in section 2. In section 3 a particular modelling approach, yielding essential system models, is presented. We conclude in section 4 with the elaboration of the essential model of the mail order company example.

2. MODELLING FOUNDATIONS

2.1. The smartie concept

The discussion of the concept of a smartie hereafter is rather concise and informal. For a more extended and formal description the reader is referred to [3,4,5,7]. As the smartie concept is still evolving, there are slight differences with the references mentioned. The one presented here can be considered as more general than the former conceptions.

A smartie is a conceptual mechanism, which manipulates two kinds of elements, called directives and constatives. A *directive* fulfills the role of an order or a command to a smartie. There is a validity period associated with every directive. The meaning of this validity period is that the smartie to which the directive is addressed must respond to it within that period. A *constative* expresses an elementary state of affairs in some part of the world, which is called the *object world* of the smartie to which the constative is addressed.

At any moment a smartie has at its disposal a set of directives and a set of constatives. These sets are called respectively the *agenda* ('agenda' means 'things to be done') and the *noscenda* ('noscenda' means 'things to be known'). A directive in the agenda is called actual at time t if t falls within its validity period.

The operation of a smartie now consists of cyclicly selecting actual directives and attempting to execute according operational rules. Rules can be executed simultaneously. The set of all rules which a smartie is able to execute, is called its *operations base* . The operations bases of different smarties are disjunct, in other words, every operational rule can be member of at most one operations base. An *operational rule* is defined as a tuple of the sort < **a**, **d**, **f** >, where **a** is a set of directives, and **d** and **f** are sets of directives and/or constatives. The semantics of the execution of a rule are as follows: if **a** is the set of selected actual directives, and if **d** equals the union of the agenda and the noscenda, then **f** is generated. The components **a**, **d** and **f** respectively are called the *action*, the *data* ('data' means 'the things given') and the *facta* ('facta' means 'the things made or done') of the rule. The directives in the facta are called the *dirigenda* ('dirigenda' means 'things to be issued') and the constatives in the facta are called the *statuenda* ('statuenda' means 'things to be stated').

For the specification of operational rules a sugared version of a first order language seems to be adequate. As an example of such a specification the next rule, applying to the MOC, may hold:

```
ON                                                        {action}
                    accept order x
IF                                                        {data (accidentally
                    order x is not accepted AND            all noscenda)}
                    the customer of x is creditworthy AND
                    the product of x is deliverable
THEN                                                      {facta}
        STATE                                             {statuenda}
                    order x is accepted
        ISSUE                                             {dirigenda}
                    deliver order x
```

The agenda, the noscenda, the dirigenda and the statuenda of a smartie at any moment are subsets of respectively the *agenda base*, the *noscenda base*, the *dirigenda base* and the *statuenda base*. A smartie therefore is completely defined by a tuple < A, N, D, S, O >, where A is the agenda base, N the noscenda base, D the dirigenda base, S the statuenda base and O the operations base.

Smarties are able to act upon (other) smarties by means of producing dirigenda and statuenda for the other smartie. The effectuation of dirigenda proceeds as follows. If a smartie 1 at some point in time generates as a result of the execution of an operational rule the dirigenda **d1**, then the agenda **a2** of every smartie 2 (which needs not differ from smartie 1!) is instantly changed to **a2'**, defined as (the symbol Δ denotes the symmetric set difference operator):

$$\mathbf{a2'} = \mathbf{a2}\ \Delta\ (\mathbf{d1} \cap \mathrm{A2}) \qquad (1)$$

Note that there can only be a real change if the next condition holds:

$$\mathrm{D1} \cap \mathrm{A2} \neq \emptyset \qquad (2)$$

(If $\mathrm{D1} \cap \mathrm{A2} = \emptyset$, then also $\mathbf{d1} \cap \mathrm{A2} = \emptyset$ since $\mathrm{D1} \supset \mathbf{d1}$, and thus $\mathbf{a2'} = \mathbf{a2}\ \Delta\ \emptyset = \mathbf{a2}$).

Similarly, the effectuation of statuenda proceeds as follows. If a smartie 1 at some point in time generates as a result of the execution of an operational rule the statuenda **s1**, then the noscenda **n2** of every smartie 2 (again, including smartie 1) is instantly changed to **n2'**, which is defined as:

$$\mathbf{n2'} = \mathbf{n2}\ \Delta\ (\mathbf{s1} \cap \mathrm{N2}) \qquad (3)$$

Note that there can only be a real change if the next condition holds:

$$S1 \cap N2 \neq \emptyset \tag{4}$$

(If $S1 \cap N2 = \emptyset$, then also $s1 \cap N2 = \emptyset$ since $S1 \supset s1$, and thus $n2' = n2 \Delta \emptyset = n2$).

2.2. The smartienet

The smartie concept is a systemic concept. It fulfills e.g. the very well articulated and formally defined notion of a system described in [2]. Therefore, a smartie can be viewed as a component of a (super) smartie, but also as a system of smarties. The composition and decomposition of smarties is simple and straightforward [3].

In order to provide more practical support to the modelling of systems as (compositions of) smarties, the notion of smartienet and an accompanying diagramming technique are developed. A *smartienet* is a network consisting of two kinds of components: processors and banks. The notion of *processor* is the mechanical interpretation of the operations base of a smartie. It is thus to be viewed as a device which cyclicly attempts to execute operational rules. Because processors represent the core of smarties, viz. the operations base, we will mostly identify them with smarties for the sake of convenience. Thus, we speak of the agenda of a processor, of its agenda base etc.

The notion of *bank* is the mechanical interpretation of the storage of non-empty intersections of dirigenda bases and agenda bases on the one hand, and of statuenda bases and noscenda bases on the other hand. Consequently, we distinguish between two classes of banks: directives banks, or *d-banks* for short, and constatives banks, or *c-banks* for short. A bank is defined by the set of all facts it is able to store, called the *storage base*. The storage bases of the banks in a smartienet are disjunct.

There are two kinds of connections between processors and banks, called *inspection* links and *generation* links. A bank, which is connected to a processor via an inspection link is called a *data bank* of the processor. Similarly, a bank, which is connected to a processor via a generation link, is called a *facta bank* of the processor.

The *agenda base* of a processor is the union of the storage bases of its data d-banks, and the *agenda* of a processor at any moment is the union of the contents of these banks. Similarly, the *noscenda base* of a processor is the union of the storage bases of its data c-banks, and the *noscenda* of a processor at any moment is the union of the contents of these banks. Furthermore, the *dirigenda base* of a processor is the union of the storage bases of its facta d-banks, and the *statuenda base* of a processor is the union of the bases of its facta c-banks.

The structure of a smartienet can be visualized by means of the so-called CSD (Communication Structure Diagram). Actually a CSD not only shows the *structure* of a system, but also its *composition* (the set of actors which are the components of the system) and its *environment* (cf. [2]). Figure 4 shows the graphical symbols used in a CSD.

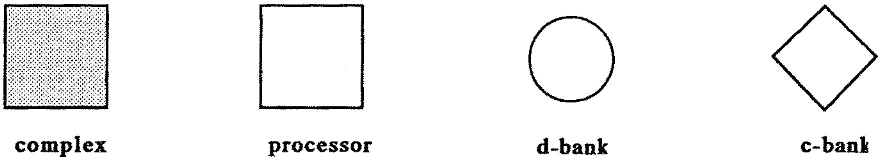

Figure 4. Graphical symbols used in a CSD

The processor symbol denotes an *elementary processor*, i.e. a processor for which the above definitions of the agenda base, the noscenda base, the dirigenda base and the statuenda base hold. The complex symbol represents a smartienet of arbitrary magnitude which 'looks' like a processor (cf. [3]). An inspection link is graphically represented by an arrow from a bank to a processor (or complex). A generation link is graphically represented by an arrow from a processor (or a complex) to a bank. The elementary constructs of a CSD are depicted in fig. 5. The dirigenda base of processor 1 and the agenda base of processor 2 in the upper diagram do have a non-empty intersection. The operational meaning of this property is that processor 1 generates directives, which then become elements of the agenda of processor 2. Within the validity periods of the directives processor 2 will execute the corresponding operational rule(s). Because of this relationship processor 1 is said to *direct* processor 2.

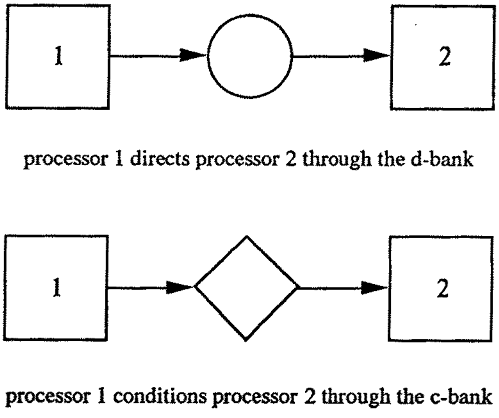
1
2
processor 1 directs processor 2 through the d-bank
1
2
processor 1 conditions processor 2 through the c-bank

Figure 5. Elementary constructs in a CSD

The statuenda base of processor 1 and the noscenda base of processor 2 in the lower diagram do also have a non-empty intersection. The operational meaning of this property is that processor 1 generates constatives, which then become elements of the noscenda of processor 2. In this way processor 1 is able to co-determine the operational rule(s) which processor 2 will execute at some time. Because of this relationship processor 1 is said to *condition* processor 2. If processor 1 and processor 2 are the same processor, we speak of self-directing and self-conditioning respectively. Self-directing and self-conditioning relationships are represented in a CSD by double arrows.

To conclude this section we like to stress some remarks concerning the interpretation of a CSD. Firstly, the symbols of a CSD only have the meaning as defined in this paper. Of course, they may be used as well in quite different meanings, but then they do not represent a smartienet (cf. e.g. [8]). Secondly, especially the arrows should never be taken to represent a kind of flow, because they essentially do not. Arrows in a CSD are just a graphical way of specifying the components A, N, D and S of the interacting smarties, which the diagram actually represents.

3. ANALYSIS OF THE PROBLEM

An organization is considered to be a system of acting, decision taking and communicating actors. We use the term actor to denote persons who fill an organizational function or position. A function may be filled by several persons and a person may fill simultaneously several functions.

The communication between actors consists of exchanging (knowledge of) facts. A *fact* is defined as a pair < illocution, proposition >.

The term *illocution* is borrowed from speech act theory [1,10,11]; it represents the pragmatic function or effect of the fact. Two illocutionary types are distinguished: directive and constative. Consequently we distinguish between two classes of facts: directive facts, or directives for short, and constative facts, or constatives for short. A *directive* is an order or command to operate in such a way as to make the proposition true. A *constative* is a statement of truth regarding the proposition.

The *proposition* is a predicate with respect to some reference world. It consists of a predicate symbol and a number of arguments. The arguments refer to objects in the reference world, and the predicate symbol represents some relation type. Consequently, we only deal with simple or elementary facts.

The purpose of this section is to find out which actions, decisions and communications must be viewed as essential (and thus be incorporated in the essential model), and which must not (and thus are only relevant at the logical level).

3.1. Action and decision

The result of an *act* is always a change of state of some world. We will call the world which is affected by the actors in an organization, the *object world* of that organization. An *essential act* can now be defined as an act which causes a change of state of the object world. Although this definition of essential acts looks like a circle argument, it appears to be quite clear in any context what should be called essential and what should not. An example regarding the mail order company may clarify this point: the acceptance of an order and the shipping of goods are essential acts, whereas the calculation of a purchase quantity or the sending of internal orders are not. With respect to the former, the latter can be considered as supporting acts.

We distinguish between two kinds of essential acts: material acts and institutional acts. Accordingly, we distinguish between two part-worlds of object worlds in which these acts respectively have meaning: the material world and the institutional world. At any point in time both the material world and the institutional world are in a particular state. A state of an object world is represented by a set of constatives. Because of the distinction between the two worlds we also distinguish between material constatives and institutional constatives.

A *material act* is an act of which the result is a change of state of the material world. An example of a material act is the shipping of goods from a warehouse to a customer's place. States and state changes of the material world are governed by the laws of nature.

An *institutional act* is an act of which the result is a change of state of the institutional world. An example of an institutional act is the ordering of goods by a customer. States and state changes of the institutional world are governed by institutional laws or rules. Institutions are human inventions which help to regulate societal life. The working of institutions ultimately rests on accepted norms and rules. Well-known examples of institutions are all kinds of games (like chess and football) and all legal institutions (like marriage and enterprise).
Institutional acts are speech acts of the type declaration (cf. [10,11]). For example, if the Registrar of a municipality declares two persons to be married, a marriage is performed. Otherwise said, a new institutional fact is created. Also, if the umpire in a tennis match calls a ball out that ball is out. Note however that the umpire in doing so creates an institutional fact, not a material one. The material fact of the ball being out may or may not conform to the institutional fact: a ball can very well be (materially) in and (institutionally) out, and vice versa. Only actors who have the right authority can perform institutional acts. If you, as spectator of a tennis match, declares a ball to be out, that speech act will have no effect on the course of the match. Declarations constitute a very important subclass of speech acts. They pre-eminently legitimate the title of Austin's seminal work: 'Doing things with words' [1].
A constative which refers to the institutional world is (an expression of) the result of a declaration. We have met already two examples of such constatives above: the fact of a

marriage and the fact of a ball in a tennis match being out. An example concerning the mail order company is the fact that the price of product x is y.
A constative, which refers to the material world is an expression of the result of an observation. An example of such a constative is that product x is stored in bin y.
The essential acts in most organizations are a mixture of material and institutional acts. In some organizations however there are only institutional acts. These organizations therefore can rightly be called *institutional organizations.* Examples of institutional organizations have already been mentioned in section 1: banks and insurance companies. It is evident from the analysis above that these organizations do have their own particular essential acts, which just happen to be purely institutional. The crucial point is that they are acts which do change the state of the object world, contrary to all information processing activities, which do not.

Another important subclass of speech acts comprises acts like requests, commands etc. The defining property of these speech acts is that they cause an actor to perform some (material or institutional) act in the future. In order to be succesful, it must be sure that the actor to which the request is addressed will act as requested. If this is the case, we say that a *directive* fact, or directive for short, is generated (cf. section 2). Examples of directives are the ordering of goods by a customer, the request to refrain from smoking by an air hostess, and asking a servant to polish your shoes (in the colonial era). Actually, the accomplishment of a directive is a very complicated and subtle process of negotiation. In order to get more insight in this subject matter the reader is referred to [6]. We simplify this process in that we allow an actor to issue a directive to some actor, meanwhile assuming that the actor to whom the directive is addressed implicitly promises to act accordingly. Otherwise said, the generated directive is the agreed

	material	*institutional*
directive	’Store product x in bin y’ ’Stop smoking’	’State the price of product x’ ’Marry us’
constative	’Product x is stored in bin y’ ’She has stopped smoking’	’The price of product x is y’ ’You are married’

Figure 6. The different categories of facts

upon outcome of the negotiation. Despite the simplification made however, our notion of a directive appears to be very useful in modelling the operation of organizations as well as other discrete dynamic systems.

Directives are not the result of, material or institutional, acts. Instead, we consider them to be the result of *decisions*. Thus we posit that, next to performing acts, actors are able to decide to accomplish directives. Depending on the kind of act, which a directive evokes, one may distinguish between material directives and institutional directives.
It is important to distinguish the (essential) decisions mentioned above from decisions taken in the course of information processing, e.g. during a computation. The results of decisions of this kind always are program or procedure calls. Since these calls do not evoke essential acts, they belong to the logical level.
Concludingly, one may distinguish four categories of facts, which are summarized in figure 6.

3.2. Reproduction and deduction

The actor who (authoritatively) performs an act, be it material or institutional, is the *originator* of the constative(s) which express(es) the result of the act. Similarly, the actor who takes a decision is the originator of the directive(s) which express(es) the result of the decision.
Next to an originator, every fact has some actor as addressee. The *addressee* of a fact is the actor who needs to know the fact in order to perform his essential acts and/or decisions. It is possible that there are two or more addressees. Moreover, the originator and the adressee of a fact may be the same actor (cf. self-directing and self-conditioning as explained in section 2).
Now, if one takes a look at the A-graph in figure 2, one can find several examples of communication between two actors of which either the sender is not the originator of the communicated facts or the receiver is not the addressee. One of the examples is the sending of confirmation letters by Customer Service to customers. Customer Service is not the originator of the facts contained in such a letter, that is the Acceptance department (to be very precise: the Acceptance department is only originator of the fact of an order being accepted. Other facts contained in a confirmation letter, like the date and the customer's address are originated by other actors, both outside the scope of the system considered, in any case not by Customer Service).
Apparently, Customer Service only *reproduces* facts. Since reproduction does not change the state of the object world, the act of producing a confirmation letter must be considered an *improductive*, i.e. not essential, act.

Let us take another look at figure 2, notably the actor Statistics. As can be learned from the description in section 1 this actor produces useful statistics for the Purchasing department, probably presented in modern fancy ways like coloured bar and pie charts. The acts of

Statistics do not change the state of the object world and therefore must be considered improductive. In this case it is not a matter of reproduction but of *deduction* of facts. The facts produced by the Statistics department are deductions of the facts produced by e.g. the Warehouse department. However complex these deductions may be, the facts produced can always be defined, declaratively or procedurally, in terms of the original facts. Therefore, although the deduced facts may offer a much better insight into the actual state (and/or past states) of the object world, they express that same state(s).

Because all acts performed by Customer Service and by Statistics consist of reproduction or deduction, they will not appear in the essential model of the MOC. By similar arguments, the (acts of the) Stockkeeping department will be eliminated. (If one takes the time to think of the consequences of this criterion of (im-)productivity, one will discover that very many if not most of the acts performed by people in organizations, are improductive. So much the better: it means that essential models are much simpler than logical ones!).

4. THE ESSENTIAL SYSTEM MODEL

We are now able to provide a more precise definition of the distinctive properties of an essential system model:

1. It regards only *essential acts*, i.e. acts of which the results are changes of state of the object world. Particularly the deduction of facts is eleminated.

2. It regards only *essential decisions*, i.e. decisions of which the results are directives evoking essential acts. Particularly computational decisions are eliminated.

3. It regards only *essential communication*, i.e. between originators and addressees of facts. Particularly the reproduction of facts is eliminated.

There is another, equivalent, definition, which we actually have mentioned a few times already. This definition ignores the different kinds of acts, decisions and communication, and therefore lacks to provide the insight that can be gained by the first one. It simply contrasts *essential* with *informational*, and thus defines an essential system model as a model which does not contain or represent any (logical or physical) informational activities, like computing, transporting, copying etc. We have found however that only people who have learned to understand the first definition, are able to apply the second one correctly. They also actually do so in most cases.

4.1. Working-out of the Mail Order Company example

The smartienet model of the MOC discussed in this section is designed in such a way as to correspond as much as possible to the logical model expressed by the A-graph in figure 2. The consequence of this is, among other things, that several missing elements and lacks of balance can be pointed at. The purpose of the CSDs shown however is only to convey the basic character of essential system modelling by means of smartienets.

Figure 7 shows the CSD of the MOC and its environment. The boundary of the system is marked as a grey-coloured rounded rectangle. Interaction between the system and the

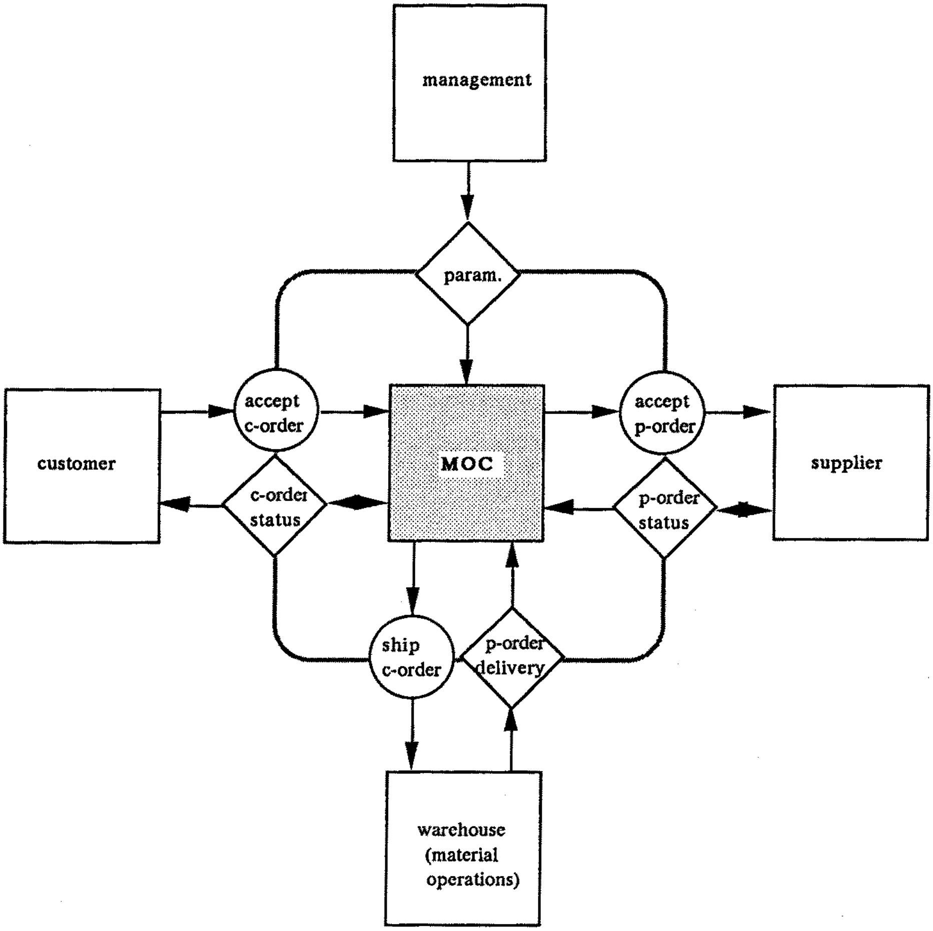

Figure 7. CSD of the system MOC and its environment

environment takes place via the *interface* banks. The core of the system is constituted by the complex 'MOC'.
The diagram expresses among other things that 'MOC' generates constatives concerning the status of orders, which are knowable to itself and to the processor 'customers'. What the diagram does not express is that within the company there is a file containing order status data, and that copies of these records are sent to customers, etc. All these things are the subject of a logical model.

The operation of 'MOC' is conditioned by facts of the type 'parameters', which are externally generated facts. In order to avoid unnecessary detailing, the originators of these facts are collected under the heading 'management'. Examples of these facts are facts concerning the deliverable assortment and facts concerning customers, as will become clear from figure 8. Note that these management influences cannot be expressed in a genuine A-graph (cf. fig. 2).
The material warehouse operations are concentrated in the processor (actor) 'warehouse' which is, contrary to the A-graph, considered to be not a part of the system modelled. The reason is that a smartienet does not model material acts themselves, but only the control of these acts.
The interaction between 'MOC' and 'warehouse' consists of the communication of constatives and of directives. The directives concern the material shipping of goods to customers ('ship c-order'). The constatives are the statements from 'warehouse' to 'MOC' concerning the reception of purchase order deliveries.

Figure 8 shows a decomposition of the system 'MOC' into four subsystems: 'acceptance', 'purchasing', 'delivery' and 'reception'. The internal structures of these subsystems are hided, in order to focuss on the interrelationships between the subsystems as well as on the interface connections with the environment. Each of the complexes however can be decomposed further, of course. As a consequence of the decomposition shown there was a need to distribute the bank 'parameters' into banks containing the specific information for the distinct subsystems.
Additionally, the parameters for the subsystem 'acceptance' are further distributed into 'products' and 'customers'. The latter is done only to provide as much insight into the operation of the system as possible, there was no formal necessity for it. Note that both 'acceptance' and 'delivery' are linked to the bank 'order status'. Both are allowed to update its contents, whereas only 'delivery' also needs the information contained in the bank for its own operations. Note also that 'purchasing' has no external inspection d-bank. Apparently, it operates by means of self-directing, which is in accordance with the narrative description of its operations (cf. section 1: '...The Purchasing department periodically inspects...').
What becomes also apparent from figure 8 is that only original facts play a role in a smartienet. For instance, whereas in the A-graph there is talk of 'stock information', in the CSD there are only delivery constatives and replenishment constatives. The stock of a product clearly is a deduced fact. It can be defined at any moment as the algebraic sum of all replenishments minus the algebraic sum of all deliveries concerning the product up to that moment.

Of course, it would be rather impractical to design an inventory information system such that the actual value of the stock of an article has to be computed, according to the above mentioned deduction rule, each time it is needed. However, the decision to keep record of the actual stock value as a separate fact is a design decision at the logical level, and justified only by efficiency considerations.

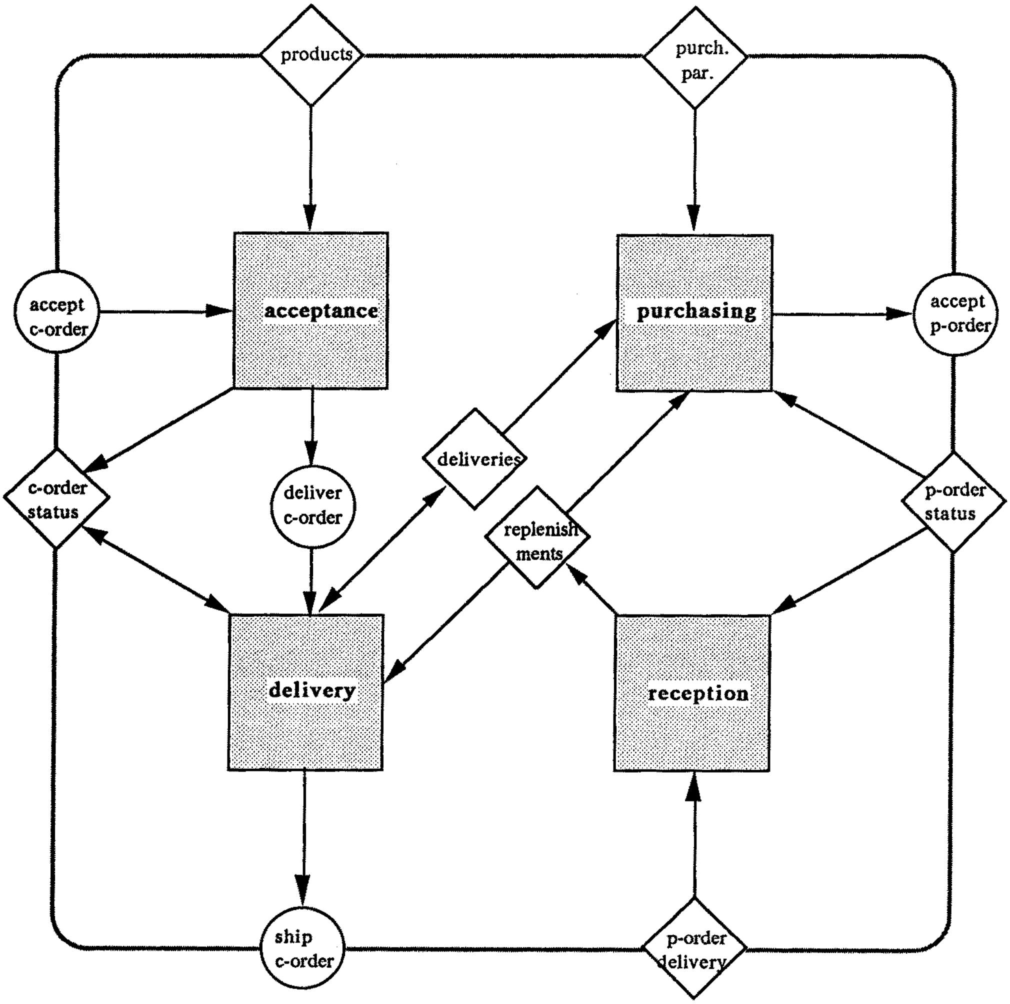

Figure 8. CSD of the MOC decomposition

4.2. Evaluation

Although the matter presented in this paper consists of interim results of ongoing research, it seems that the next conclusions can be drawn confidently.

The usage of essential system models is a major improvement of the *systems analysis* task, in that the concern for 'essential correctness' is separated from the concern for 'logical correctness'. As an example, determining which persons must have access to which information, and who is responsible for producing that information is a quite easy job if one disposes of an essential model.

The usage of essential system models is a major improvement of the *information planning* task, in that the (conceptual) information systems can be localized easily: in principle, every bank corresponds to a (conceptual) data base system, and every processor corresponds to a (conceptual) decision support system. This matter needs to be researched further.

Essential system models are of great help in understanding the 'nature' of some business. Also, the essential similarities between systems can easily be determined. Therefore they appear to be very useful in designing *reference models* for classes of companies.

The *smartienet* is a well defined metamodel for modelling the operations of discrete dynamic systems, which appears to be very well suited for expressing essential system models.

REFERENCES

1. Austin, J.L., *How to do things with words*,
Harvard University Press, Cambridge MA, 1962.

2. Bunge, M.A., Ontology II: a world of systems, *Treatise on Basic Philosophy*, vol. 4,
D. Reidel Publishing Company, Dordrecht, Netherlands, 1979.

3. Dietz, J.L.G., A Communication Oriented Approach to Conceptual Systems
Modelling, in: *Proc. of the Int. Working Conf. on Dynamic Modelling of Information Systems*, Delft University Print, the Netherlands, 1990.

4. Dietz, J.L.G., A communication oriented approach to conceptual modelling of
information systems , in: Steinholtz, B., Sølvberg, A., Bergman, L. (eds.), *Advanced*

Information Systems Engineering, Lecture Notes in Computer Science 436, Springer-Verlag, 1990.

5. Dietz, J.L.G., Hee, K.M. van, A framework for the conceptual modelling of discrete dynamic systems, in: Rolland,C., Bodart,F., Leonard,M. (eds.), *Temporal aspects in information systems*, Elsevier Science Publishing Company, Inc., 1988.

6. Habermas, *Theorie des kommunikatives Handelns*, Erster Band, Suhrkamp Verlag, Frankfurt am Main, 1981.

7. Hee, K.M. van, Houben, G-J., Dietz, J.L.G., Modelling of discrete dynamic systems; framework and examples, in: *Information Systems*, vol 14, no. 4, 1989.

8. Jackson, M., *System development*, Prentice-Hall International, 1983.

9. Lundeberg, M., Goldkuhl, G., Nilsson, A., A Systematic Approach to Information Systems Development, in: *Information Systems*, vol. 4, 1979.

10. Searle, J.R., *Speech Acts*, Cambridge University Press, Cambridge, 1969.

11. Searle, J.R., *Expression and Meaning*, Cambridge University Press, Cambridge, 1979.

12. Yourdon E., *Modern Structured Analysis*, Prentice-Hall International Inc., 1989.