

# Basic notions regarding business processes and supporting information systems

Jan L.G. Dietz

Delft University of Technology  
P.O. Box 5031, NL-2600GA Delft  
j.l.g.dietz@ewi.tudelft.nl

**Abstract.** Six basic notions concerning business processes and their supporting information systems are presented and discussed, on the basis of one common theory. This yields that these notions are defined not only clearly and precisely but also coherently and integrally. Regarding *system* and *model*, teleology (black-box) and ontology (white-box) are considered. Regarding *business process* and *information system*, a socioeconomics based definition of organization is provided as well as three corresponding levels of abstraction. Regarding *design* and *architecture*, general and special requirements are distinguished, in addition to the distinction between black-box and white-box models.

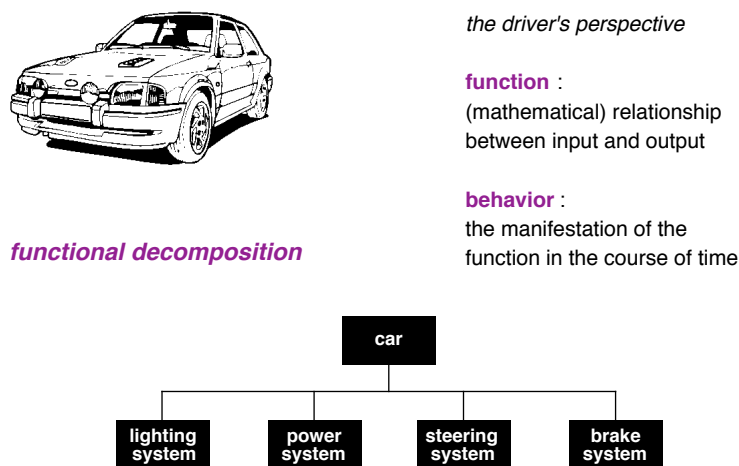
## 1 Introduction

This paper presents and discusses six notions that we consider basic for dealing with business processes (BPs) and their supporting information systems (BPSs) in an effective way. The notions that (in pairs) will pass in review are: system & model (section 2), organization & business process (section 3), and design & architecture (section 4). The findings are wrapped up in conclusions (section 5). The distinctive feature of the way in which the said notions are dealt with is that they are defined on the basis of the theory that underlies DEMO (Design & Engineering Methodology for Organizations) [4,5,6,9]. Consequently the notions are defined clearly, coherently, consistently and comprehensively.

The establishment and maintenance of the fit between BPs and BPSs is often considered to take place in two phases: the creation phase, in which a BP and its BPS are designed concurrently, followed by the maintenance phase, in which the parallel evolution of the two is managed. From our experience in numerous practical projects it seems more appropriate to think only and always in terms of evolution. It is very rare that one can start the design of a BP or a BPS from scratch. A second nuance we like to emphasize is that concurrency has to be understood such that the BP comes first and the BPS is second, since it is by definition derived from the BP. This holds also for the case that one implements a standard BPS (like an ERP or a WFM system) Moreover, the redesign of a BPS is a part of the complete re-engineering of the redesigned BP, next to other kinds of re-engineering (like human resource development, re-arranging the accommodations etc.).

## 2 System & model

There exist two distinct but complementary system notions, each with its own value, use, and type of model: the teleological and the ontological system notion. The *teleological system* notion is concerned with purpose; it is about the (external) function and behavior of a system [1,4]. The corresponding type of model is the *black-box model*. Ideally, such a model is a (mathematical) relation between a set of input variables and a set of output variables, called the transfer function. Knowing the transfer function means knowing how the system responds to variations in the values of the input variables by changing the values of the output variables. The teleological system notion is adequate for the purpose of *using* or *controlling* a system. It is therefore the dominant system concept in e.g. the social sciences, including the organizational sciences. Figure 1 exhibits the black-box model of a car (as just an example). It corresponds with the driver's perspective on a car. Through changing the values of the input variables (e.g. the position of the steering wheel) the driver is able to change the values of the output variables (e.g. the direction of the car).



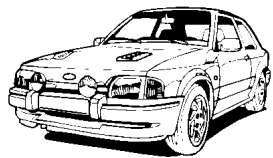
**Figure 1. Black-box model and functional decomposition**

If the transfer function is too complicated to understand, the technique of *functional decomposition* can be applied through which the black-box model of a system is replaced by a structure of sub models of which the transfer functions are more readily understandable. Figure 1 shows a possible decomposition of a car. One has to keep in mind that the knowledge acquired about the system through functional decomposition is still functional or behavioral knowledge. It is a widely spread misunderstanding to think that functional decomposition ultimately leads to knowing the construction of the system. This can never be true since a black-box model is a purely conceptual model of the input-output relationships, and since one can therefore make virtually any functional decomposition one likes.

The *ontological system* notion is about the (internal) construction and operation of a system [3,4]. The ontological definition of a system is as follows. Something is a system if and only if it has the next properties:

- *Composition*: a set of elements of some category (physical, social etc.).
- *Environment*: a set of elements of the same category; the composition and the environment are disjoint.
- *Production*: the elements in the composition produce things (goods or services) that are delivered to the elements in the environment.
- *Structure*: a set of interaction relationships among the elements in the composition and between them and the elements in the environment.

The corresponding type of model is the *white-box model*, which is a direct conceptualization of the ontological system definition. The ontological system notion is adequate for the purpose of *building* or *changing* a system. It is therefore the dominant system notion in all engineering sciences. An important characteristic is the category to which the elements of a system belong. It makes the elements being atomic for the category. Therefore such a system is called a *homogeneous* system. Homogeneous systems can be integrated to constitute heterogeneous systems. A car for example, like an organization, is a heterogeneous system. Figure 2 exhibits the white-box model of a car, which conveys the mechanic's perspective on a car.



*the mechanic's perspective*

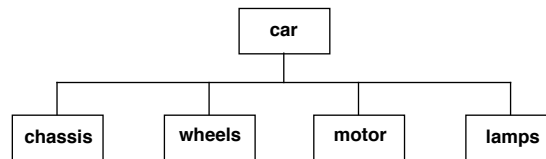
**constructional composition**

**construction :**

(active) elements and their interaction relationships

**operation :**

the manifestation of the construction in the course of time



**Figure 2. White-box model and constructional composition**

The relationship to function and behavior is that these are brought about, and explained, by the construction and the operation of the system. The counterpart of functional decomposition is *constructional composition*. It is the technique to compose a system as a construction of parts (elements or sub systems). Contrary to functional decomposition, there is only one way to construct a system out of its parts. Any other way of composing would not yield the same system (as everyone knows who has ever tried to repair a mechanical alarm clock or some other device).

### 3 Business process & information system

In the DEMO<sup>1</sup> theory [4,5,6] an *organization* is ontologically defined as a system of *actors* (human beings fulfilling an actor role) who perform two kinds of acts. By performing *production acts*, the actors bring about the mission of the organization. A production act may be material (e.g. manufacturing) or immaterial (e.g. diagnosing). By performing *coordination acts* actors enter into and comply with commitments. In doing so, they initiate and coordinate the execution of production acts. An *actor role* is defined as the authority to perform precisely one kind of production act. Production acts and coordination acts occur in generic socioeconomic patterns, called *transactions* [6]. A *business process* is defined as a collection of causally related transactions (which means that every transaction is initiated from within an other transaction).

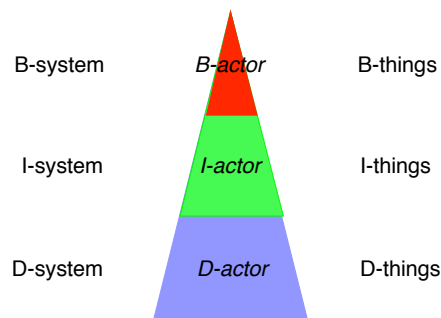


Figure 3. The three levels of abstraction

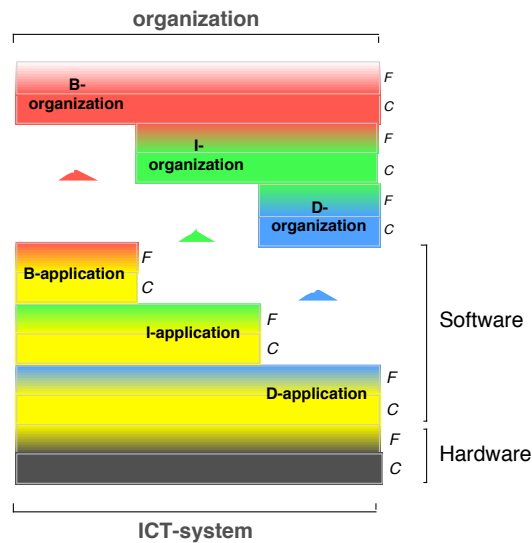
Three levels of abstraction levels are distinguished, corresponding to three human abilities (cf. Figure 3): *performa* (the ability to create original new things), *informa* (the ability to reason and thus derive knowledge) and *forma* (the ability to manage the documents that contain this original or derived knowledge) [5]. At the *performa* level, one observes the *essential* actors who bring about original (non-derivable) facts; they directly contribute to the organization's function. These essential acts and facts are collectively called *B-things* (from Business). At the *informa* level, one observes *informational* actors, who perform intellectual acts like collecting, providing, recalling and computing knowledge about business facts. Informational acts and facts are collectively called *I-things* (from Information). At the *forma* level, one observes *documental* actors, who execute documental acts like gathering, distributing, storing, and copying documents containing the knowledge mentioned above. These acts and facts are called *D-things* (from Document).

The abstraction levels correspond to the *layered nesting* of (sub) systems [3]. Generally spoken, a system in some layer *supports* (the operation of) a system in the next higher layer. Conversely, a system in some layer *uses* systems in the next lower

---

<sup>1</sup> For more information about DEMO one may visit the web site [www.demo.nl](http://www.demo.nl)

layer. So, B-systems use I-systems and I-systems use D-systems. Conversely, D-systems support I-systems and I-systems support B-systems. What the layered nesting constitutes is an intrinsically solid integration of three homogeneous systems into one *heterogeneous* system, which is the (total) organization. The integration is solid because it builds on the inseparability of the three human abilities.



**Figure 4. Relationships between aspect organizations and supporting ICT-systems**

Figure 4 exhibits in another way the layered nesting of systems, as well as their supporting information systems. To avoid confusion, we will call the layered systems *aspect organizations*, and the information systems *aspect applications*. The upper part shows the three aspect organizations: the B-organization, the I-organization and the D-organization. The distinction of the function perspective (F) and the construction perspective (C) serves to exhibit their layered nesting in a more precise way. So, the function of the I-organization supports the construction of the B-organization, and the function of the D-organization supports the construction of the I-organization. To emphasize the intermediate nature of the function perspective, the corresponding bars have the colors of the bar above and the bar below run into each other (Note. As the bar above the F-bar of the B-organization, one must think of the market context of the organization). The differences in size of the bars are primarily for the sake of relating ICT-systems to them. However, one may also interpret them for their own sake as follows: the B-organization does not have to be fully supported by the I-organization, and the I-organization does not have to be fully supported by the D-organization.

The lower part of the figure shows three aspect applications, as well as the hardware on which they run. The color for hardware is gray and the color for software is yellow. Also here, the distinction between function and construction is indicated.

Starting from the bottom, Figure 4 exhibits that the construction of the D-application software 'runs on' the function of the hardware. By this class of software is meant all generic, i.e. not organization specific, software. Examples are text processors, spread sheet programs, operating systems, network systems and data base management systems. Partly, the D-applications may directly support the D-organization (indicated by the fact that the I-application bars are shorter than the D-application bars). The I-applications are put on top of the D-applications to express that they commonly will not run directly on hardware but through the intermediate role of D-applications. I-applications are by definition organization specific, although they may have a generic character, like accounting systems. Typical I-applications are Management Information Systems; they contain all relevant information to monitor the business. Typical B-applications are Decision Support Systems and Process Management Systems. They are put on top of the I-applications for the same reason as the I-applications were put on top of the D-applications: they will commonly make use of the I-applications. And for the same reason as the bars of the I-applications are shorter than the bars of the D-applications, the bars of the B-applications are shorter than those of the I-applications.

Up to now, we only spoke of an application supporting an aspect organization. One may go a step further and consider an application to replace, completely or partly, the corresponding aspect organization. Let us consider what it means to say that the D-organization is replaced (partly or fully) by D-applications. Two observations can be made. The first is, that documental acts can very well be performed by ICT-applications. As a matter of fact, these systems excel human actors to a high degree if it comes to store, retrieve, transport, copy etc. documents. After all, this was the very first kind of deployment of programmed computers. The second observation is that in a D-organization, it is always clear who is responsible for particular documental work (some actor in e.g. the internal post service or the archive). This clearness of responsibility must be preserved. This can easily be achieved if one strongly and consistently adheres to the core property of all organizations, namely their being systems of human actors. The consequence of this 'social' stance is that a D-organization can never really be replaced by ICT-systems, it can only be *supported* by them. Put differently, one may substitute human workers by ICT-systems, but their actor roles (responsibilities) remain and have to be fulfilled then by someone else. In the extreme situation, all documental work is done by machines and there is one person left who fulfills all the former actor roles, and who then is responsible for the well functioning of all these machines.

Similar reasonings hold for the 'replacement' of the I-organization by I-applications and for the 'replacement' of the B-organization by B-applications. As may have become clear by now, we consider the processes in the B-organization as the (real) *business processes* (BPs) of an organization. This notion of business process is a very stable one since in the B-organization one abstracts completely from all realization or implementation issues. For instance, it does not comprise information systems or information flows.

## 4 Design & architecture

In designing a system (of any kind) both the teleological and the ontological system definition are relevant, or, if one likes, both the functional and the constructional perspective on systems. Next to that, the notion of layered nesting of systems, as discussed in section 3, is of paramount importance. Figure 5 exhibits the most basic steps in a design process. The starting point is the need by some system, called the *using system* (US), of a supporting system, called the *target system* (TS). By nature, this need stems from the construction of the US, so one starts with a white-box model of the US. Then one determines the requirements for the TS in terms of the construction and operation of the US. Also by nature, these requirements are about the function and the behavior of the TS, thus in terms of a black-box model of the TS. Therefore they are called *functional requirements*. The next basic design step is to devise specifications for the construction and operation of the TS, in terms of a white-box model of the TS. A thorough analysis of this white-box model must guarantee that building the TS is feasible, given the available technology. These specifications have to be understood as the *constructional requirements* for the TS.

Let us focus now on BPs and BSPs. Even if, from the perspective of the US, the set of (functional) requirements regarding a TS is fully appropriate and complete, it may be desirable from the point of view of the enclosing enterprise to impose (additional) functional requirements that do not conflict with the determined requirements, but that would satisfy other goals of the enterprise. Suppose that the TS is a particular accounting system. Then an example of a general requirement could be: all accounting systems must be in conformity with European law. Next to that, there is generally a large amount of freedom in devising the specifications for the TS. So, there is room for (additional) general constructional requirements, put forward by the enterprise. An example of such a general constructional requirement is that all ICT-applications (thus not only accounting systems) should be coded in Java.

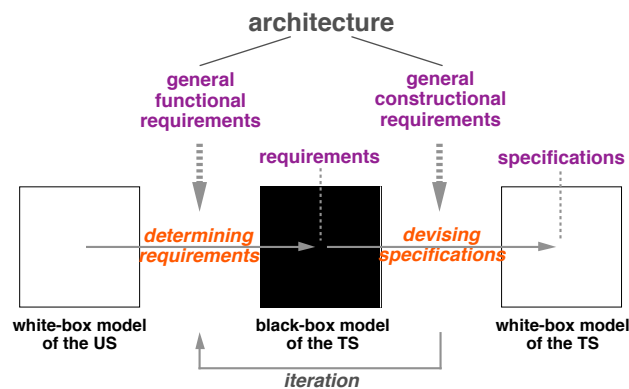
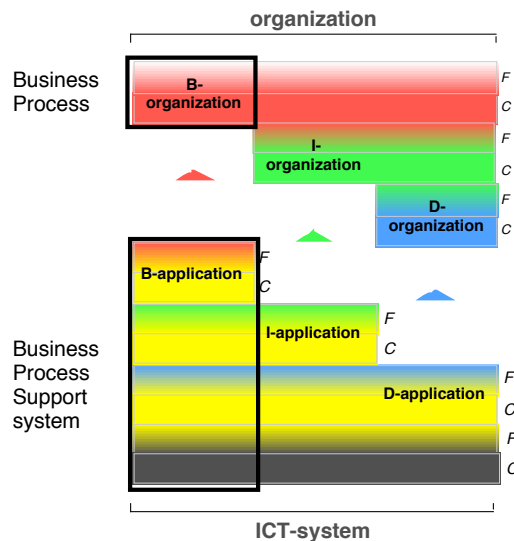


Figure 5. The design process and the role of architecture

*Architecture* is (or preferably should be) about satisfying these general functional and constructional requirements. This leads to the conception of architecture as *normative restriction of design freedom*. This idea of consciously applying normative restriction of design freedom is the really new thing regarding the designing and engineering of BPs and BPSs. We therefore consider it unfortunate that so often architecture is defined as rather synonymous to global design or blueprint.

The requirements that are acquired from the US (the future users of the TS) may be called the *special* requirements, to distinguish them from the *general* requirements (see Figure 5). To make architecture practical, the general requirements must be transformed into a consistent and coherent set of *design principles*. An *architecture* can thus be defined as a set of design principles. Applying a design principle satisfies one or more general requirements. The same architecture may also hold for other systems. Actually, only in its reuse lies the practical benefit of architecture. In line with the distinction between functional and constructional requirements we distinguish between *function architecture* and *construction architecture*. An example of ‘good’ function architecture is the Apple Mac OS; an example of ‘bad’ function architecture are the first video recorders. An example of ‘good’ construction architecture is the modern PC; an example of ‘bad’ construction architecture are the first (‘spaghetti’) computer programs.



**Figure 6. The BP and the BPS in the proposed theory**

If one combines the model in Figure 5, considering a BP as the US and its BPS as the TS, with the model in Figure 4, one gets what is exhibited in Figure 6. This figure once more demonstrates that the fit between a BP and its BPS is basically the fit between the white-box model of the BP and a black-box model of the BPS.

## 5 Conclusions

The distinction between the two system definitions and the corresponding kinds of models made clear that changing (the business processes of) an organization means changing its construction. As a consequence, no black-box model can be of any help. This also holds for such models as the value chain [8] and for the representations of business processes in techniques like IDEF0 and DFD's. On the other hand, Petri Nets [1] and EPC's [7] are fine but too detailed; they lack the compression that is offered by the abstraction levels and the transaction pattern.

The theory about organizations, as presented in section 3, demonstrates first that the notion of business process can be defined and understood much more clear and much more precise than is usually done. The socio-economic pattern of the transaction is the key to understanding business processes on a high level of compression but without loss of essential details. The distinction between the three levels of abstraction provided a further refinement of this notion, leading to the definition of a business process as primarily a process in the B-organization. This process is the most stable one, most changes are always on the D-level and the I-level.

Building on the distinction between black-box and white-box models, we presented a model of the design process of a (supporting) system, in which the notion of architecture can be placed very appropriately and clearly. Combining this model with the findings of section 3, we were able to place the notion of BP and the notion of BPS in the layered structure of aspect organizations and aspect information systems.

## References

1. Aalst, W.M.P. van der, Hee, K.M. van, *Workflow Management: Models, Methods and Systems*, MIT Press, MA, 2001
2. Bertalanffy, L. von (1968), *General Systems Theory*, Braziller, New York.
3. Bunge, M.A., *Treatise on Basic Philosophy*, vol.4, A World of Systems, D. Reidel Publishing Company, Dordrecht, The Netherlands, 1979
4. Dietz, J.L.G., Understanding and Modelling Business Processes with DEMO, Proc. 18<sup>th</sup> International Conference on Conceptual Modeling (ER'99), Paris, 1999
5. Dietz, J.L.G., The Atoms, Molecules and Fibers of Organizations, *Data and Knowledge Engineering*, vol. 47, pp 301-325, 2003
6. Dietz, J. L. G., Generic recurrent patterns in business processes. In: Aalst, W. van der, Hofstede, A. ter, & Weske, M. (Eds.), *Business Process Management*, LNCS 2678. Springer-Verlag, 2003.
7. Keller, G., M. Nüttgens, A.-W. Scheer. Semantische Prozessmodellierung auf der Grundlage „Ereignisgesteuerte Prozessketten (EPK)“. Veröffentlichung des Institut für Wirtschaftsinformatik, Paper 089, Saarbrücken, 1991 (<http://www.iwi.uni-sb.de/iwi-hefte/heft089.ps>).
8. Porter, M.E., V.E. Millar, 1985, How information gives you competitive advantages, *Harvard Business Review*.
9. Reijswoud, V.E. van, J.B.F. Mulder, J.L.G. Dietz, Speech Act Based Business Process and Information Modeling with DEMO, *Information Systems Journal*, 1999