Requirements Eng (2005) 10: 175–183
DOI 10.1007/s00766-005-0002-9

ORIGINAL ARTICLE

Jan L. G. Dietz · Antonia Albani

Basic notions regarding business processes and supporting information systems

Received: 22 September 2004 / Accepted: 10 May 2005 / Published online: 9 August 2005

Abstract In order to achieve and maintain an optimal fit between business processes (BPs) and business process support systems (BPSs), both need to be understood thoroughly and coherently. Moreover, to benefit fully from the potentials of modern information and communication technology (ICT), the deep structure that lies behind the surface structure of BPs should be understood. The Ψ-theory, which is only summarized in this paper, provides the basis for such an understanding of BPs and BPSs as well as for some other basic notions. In particular, the notions of design and engineering and of architecture and ontology will be addressed. The conclusion is that these notions can consistently and coherently be related to each other, on the said theoretical basis, such that the concurrent (re)design and (re)engineering of BPs and BPSs can be performed more effectively.

1 Introduction

The core subjects in this paper are business processes (BPs) and business process support systems (BPSs), and the main research question to be addressed is how they can be understood in such a way that their continuous and concurrent (re)designing and (re)engineering can be performed more effectively than what is currently the case. The approach that is taken for finding answers to this question is to take advantage of the Ψ-theory (The Greek letter Ψ is pronounced as PSI which stands for performance in social interaction) that underlies the DEMO methodology[1] [1–3]. A summary is provided in Sect. 2.

Research in these matters is motivated by a number of factors that force the practitioners in BPs and BPSs to make them ever more flexible and adaptable, while at the same time keeping them cost-effective. To start with, the drastic changes in the competitive business landscape—like globalization of sales and sourcing markets, shortened product lifecycles, innovative pressure on processes, customer's request for individual products—drive companies to focus on their core competencies with the objective of saving time and costs while improving product as well as service quality in order to better react to fast changing market requirements. Next, innovations in information and communication technology (ICT), primarily the emergence of the Internet, offer possibilities to increase the interoperability of information systems and to improve intra-company as well as inter-company collaboration. Support systems for daily business activities exist in almost every company and include transaction processing, office automation, management support, planning and control as well as administration and scheduling systems. However, the deployment of modern ICT in building these systems does not always meet the expectations. From the early 1990s it has become clear that in order to cope with business and ICT problems successfully, a 90° shift in thinking about companies is necessary, namely from vertical, function-oriented or silo-thinking to horizontal, process-oriented thinking [4, 5]. Therefore BPs and BPSs are in the center of interest of managers, practitioners, and researchers. They have been and still are the subject of a series of annual workshops[2].

Still, we have the impression that the notions of BP and of BPS are mostly not as well-defined as they could be. First, a BP is a process and a BPS is a system. Is the use of these different terms deliberate or

J. L. G. Dietz (✉) · A. Albani
Delft University of Technology,
P.O. Box 5031, 2600 GA, Delft, The Netherlands
E-mail: j.l.g.dietz@ewi.tudelft.nl

[1]For more information about DEMO, the reader is referred to the web site http://www.demo.nl
[2]See http://www.ibissoft.se/bpmds.html

just by chance? We think that a precise and clear distinction is necessary. Therefore, we will provide a solid ontological system definition in Sect. 2, and define process as a view on or an aspect of system. Second, what makes a BP specific and in what sense does it differ from a general process? Third, what does it mean to say that a system supports some other system? These questions will be addressed in Sect. 3. Next, after having a new and more thorough and deep understanding of BPs and BPSs, how should one conceive designing and engineering? Lastly, what role does the emergent notions of architecture and ontology play in systems design and engineering? Clear, consistent and coherent answers to these questions are developed in Sect. 4. Section 5 presents and discusses the main conclusions.

As a leading example in this paper for clarifying our lines of thought, we consider a part of the organization that is described below and that is referred to as "case Volley":

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

Every Wednesday evening Charles takes the collected letters to Miranda, the secretary of Volley. He also takes the member register with him then. If Miranda decides that an applicant will become a member of Volley, she stamps "new member" on the letter and writes the date below it. This date counts as 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 so-called member number, by which the new member is indicated in the administration. Next, Miranda calculates the membership fee that the new member has to pay for the remaining part of the calendar year. She finds the annual fees, as settled by 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 back 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 made 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.

2 Summary of the Ψ-theory

There exist two different system notions, each with its own value, its own purpose, and its own type of model: the function-oriented or teleological and the construction-oriented or ontological system notion. The *teleological system* notion is about the function and the (external) behavior of a system. 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. Otherwise said, by manipulating the input variables, one is able to control the behavior.

The *ontological system* notion is about the construction and operation of a system. The relationship with function and behavior is that the behavior is brought forward, and consequently explained, by the construction and the operation of a system; through this exhibition of behavior, the function of the system is realized. These definitions are in accordance with the work of Gero et al., if one substitutes their use of "structure" by "construction and operation" [6, 7]. The use of these terms is necessary because we consider dynamic systems. The ontological definition of a system, based on the one that is provided in [8], is as follows. Something is a system if and only if it has the following properties:

- *Composition*: a set of elements of some category (physical, biological, social, chemical, 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 (products or services) that are delivered to the elements in the environment.
- *Structure*: a set of interaction bonds among the elements in the composition and between these and the elements in the environment.

An important characteristic is the category to which the elements of a system belong. The corresponding type of model is the *white-box model*, which is a direct conceptualization of this ontological system definition.

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. For the purpose of building and changing a system, one needs to adopt the ontological system notion. It is therefore the dominant system notion in all engineering sciences.

The ontological definition of an *organization* is that it is a system in the category of social systems. This means that the elements are social individuals, i.e. human beings in their ability of entering into and complying with commitments about the things that are produced in collaboration. The Ψ -theory provides an explanation of the construction and the operation of organizations, regardless of their particular kind or branch (like industry or government, or manufacturing or service). It is based on several axioms, of which the relevant ones for this paper are presented hereafter.

2.1 The operation axiom

An organization consists of *actors* (human beings fulfilling an actor role) who perform two kinds of acts. By performing *production acts*, the actors bring about the function of the organization. A production act (P-act for short) may be material (e.g. a manufacturing or transportation act) or immaterial (e.g. deciding, judging, and diagnosing). By performing *coordination acts* (C-acts for short) 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 a particular, atomic 'amount' of authority, viz. the authority needed to perform precisely one kind of production act. The result of successfully performing a P-act is a *production fact* or P-fact. The P-facts in the case Volley are "membership M has started to exist" and "the fee for membership M is paid". The variable M denotes an instance of membership. Examples of C-acts are requesting and promising a P-fact (e.g. requesting to become member of Volley).

The result of successfully performing a C-act is a *coordination fact* or C-fact (e.g. the being requested of the production fact "membership no. 387 has started to exist"). Just as we distinguish between P-acts and C-acts, we also distinguish between two worlds in which these kinds of acts have effect: the *production world* or P-world and the *coordination world* or C-world respectively (see Fig. 1). At any moment, the C-world and the P-world are in a particular state, simply defined as a set of C-facts or P-facts, respectively, created up to that moment. When active, actors take the current state of the P-world and the C-world into account (indicated by the dotted arrows in Fig. 1). C-facts serve as agenda for actors, which they constantly try to deal with. Otherwise said, actors interact by creating and dealing with C-facts.

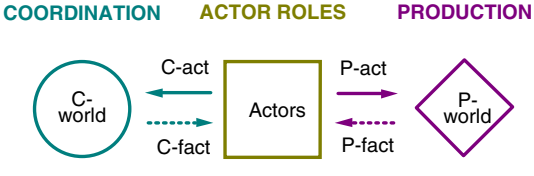

Fig. 1 The white-box model of an organization

2.2 The transaction axiom

P-acts and C-acts appear to occur in generic recurrent patterns called *transactions* [9]. The genericity of this pattern has turned out to be so omnipresent and persistent that we consider it to be a socionomic law. A transaction goes off in three phases: the order phase (O-phase), the execution phase (E-phase), and the result phase (R-phase). It is carried through by two actors, who alternately perform acts. The actor who starts the transaction and eventually completes it is called the *initiator*. The other, who actually performs the production act, is called the *executor*. The O-phase is a conversation that starts with a request by the initiator and ends (if successfully) with a promise by the executor. The R-phase is a conversation that starts with a statement by the executor and ends (if successfully) with an acceptance by the initiator. In between these two conversations there is the E-phase in which the executor performs the P-act.

Figure 2 exhibits the standard pattern of a transaction. A white box represents a C-act (type) and a white disk represents a C-fact (type). A gray box represents a P-act (type) and a gray diamond a P-fact (type). The initial C-act is drawn with a bold line, as is every terminal C-fact. The gray colored frames, denoted by "initiator" and "executor", represent the *responsibility areas* of the two partaking actor roles.

The standard pattern must always be passed through for establishing a new P-fact. A few comments are in place, however. First, performing a C-act does not necessarily mean that there is oral or written communication. Every (physical) act may count as a C-act. Second, C-acts may be performed *tacitly*, i.e. without any signs being produced. In particular the promise and the acceptance are often performed tacitly (according to the rule "no news is good news"). Third, next to the standard transaction pattern, four cancellations patterns are identified [9]. Together with the standard pattern they constitute the complete transaction pattern. Every *transaction process* is some path through this complete pattern, and every *business process* in every organization is a connected collection of such transaction processes. This holds also for processes across organizations, like in supply chains and networks. Therefore, the transaction pattern must be taken as a *socionomic law*: people always and everywhere conduct business (of whatever kind) along this pattern.

2.3 The distinction axiom

Three human abilities play a significant role in performing C-acts. They are called forma, informa, and performa, respectively [10]. The *forma* ability concerns being able to produce and perceive sentences (Note: By sentence is meant the atomic unit of information). The *informa* ability concerns being able to formulate thoughts into sentences and to interpret sentences. The term 'thought' is used in the most general sense. It may be a

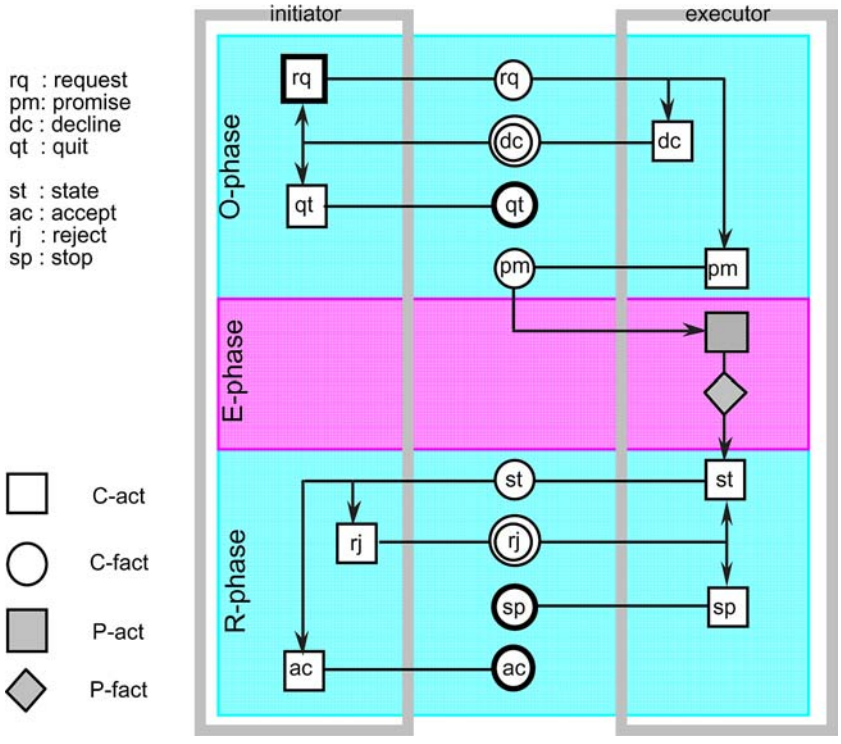

Fig. 2 The standard pattern of a transaction

fact, a wish, an emotion etc. The *performa* ability concerns being able to engage into commitments, either as a performer or as an addressee of a coordination act. This ability may be considered as the *essential* human ability for doing business (of any kind). A similar distinction in three levels of abstraction can be made on the production side. The *forma* ability now concerns being able to deal with recorded sentences, called documents (Note: The term 'document' is used here to refer in a most general sense to the forma aspect of information). The *informa* ability on the production side concerns being able to reason, to compute, derive, etc. Lastly, the *performa* ability concerns being able to establish original new things, e.g. creating material products or making decisions. Because this is at the core of doing business (on the production side), it is called the *essential* production.

Looked upon from the production side, the distinction levels may be understood as 'glasses' for viewing an organization (see Fig. 3). Looking through the *essential* glasses, one observes the essential business actors (B-actors) who perform P-acts that result in original (non-derivable) facts and who directly contribute to the organization's function. These essential acts and facts are collectively called B-things. Looking through the *informational* glasses, one observes intellectual actors (I-actors) who execute informational acts like deriving and computing knowledge about business facts. Informational acts and facts are collectively called I-things.

Looking through the *documental* glasses, one observes documental actors (D-actors) who execute documental acts like gathering, distributing, storing, and copying documents containing the knowledge mentioned above. Documental acts and facts are collectively called *D-things*. Recall that an actor is a person fulfilling an actor role. So, for example, a person may simultaneously fulfill a B-actor role, an I-actor role and a D-actor role: if, e.g. Charles receives an application letter, he may perform some documental acts (like archiving the letter), he may need to perform some informational acts (like asking the aspirant member for missing information) and he will actually deal with the request by tacitly performing a promise (see Fig. 4).

The distinction levels as distinguished in the Ψ -theory are an example of a *layered nesting* of (sub) systems [8]. Generally spoken, the system in some layer *supports* the system in the next higher layer. Conversely, the system in some layer *uses* the system in the next lower layer. So, the B-organization uses the I-organization and the I-organization uses the D-organization. Conversely, the D-organization supports the I-organization and the I-organization supports the B-organization. If a system X supports a system Y, it means that the function of

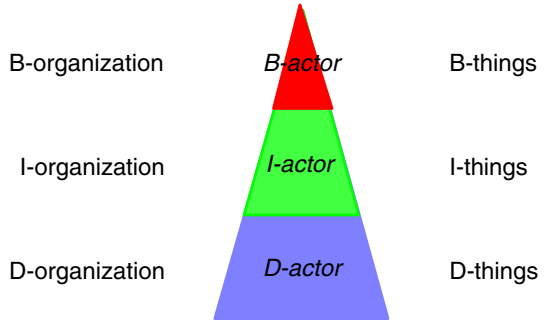

Fig. 3 Depiction of the distinction axiom

system X is expressed in terms of the construction and operation of system Y. For example, the actor in the B-system of the case Volley, who registers new members, needs to know the age of a candidate member. This information can by definition only be asked for in the I-organization. In order to get the information, the subject who fulfills the B-actor role has to take his 'shape' of an I-actor and initiate an (informational) transaction resulting in the provision of the needed knowledge by the executor of this transaction (the I-actor who is the proprietor of this piece of knowledge). On his turn, this I-actor may not know the requested knowledge by heart and thus has to initiate, in his 'shape' of D-actor, a (documental) transaction of which the executor is a D-actor who keeps record of the requested knowledge. A copy of the record (a document) is sent to the initiator who, in his shape of I-actor, is able to interpret the document and lastly, in his shape of B-actor, is able to take the appropriate action based on the acquired knowledge. What the layered nesting constitutes is an intrinsically solid integration of the three aspect organizations in the (complete) organization. The integration is solid because it builds on the inseparability of the human being, who possesses the forma, the informa, and the performa ability simultaneously.

2.4 Business process and business process support system

There are many techniques for modeling BPs. Next to the traditional flow chart, there is, e.g. the Petri Net [11, 12] and the Event Process Chain (EPC) [13]. In these techniques, the notion of BP is, however, not well-defined. Consequently the difference between a BP and some other type of process remains unclear. This leads us to the conclusion that they are not specifically meant for BPs but instead can be used for any discrete event process. Figure 4 exhibits a part of the flow chart one could come up with regarding the case Volley. Taking this example and the Ψ -theory as explained in Sect. 2, we will elaborate on the notion of BP hereafter. In particular, a BP is defined as an aspect of or view on the B-organization of an enterprise. The process model of a B-organization specifies the process steps (C-acts/facts and P-acts/facts in the transaction pattern), the causal and

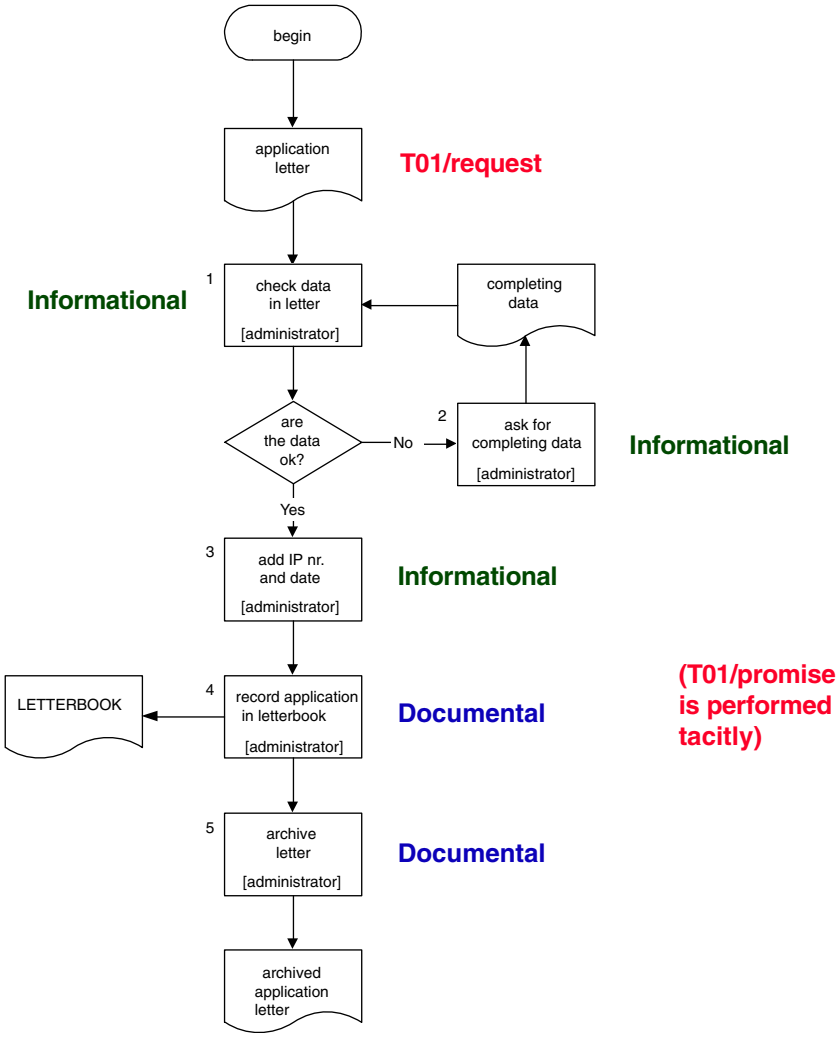

Fig. 4 Part of the flow chart of case volley

conditional relationships between them, as well as the actor roles who are responsible to perform them [9].

If the goal of being involved in the modeling and analysis of BPs is to (re)design and (re)engineer them, then it would be very profitable, if not necessary, to have a model that first does justice to being a BP model, and that next is fully independent of its implementation. This is exactly what the Ψ -theory provides the basis for. Let us first apply the implications of the *distinction axiom* to the flow chart in Fig. 4. This yields first that the actions 1–3 are informational. It means that they do not produce original new things and that they can trouble-free be realized in other ways. Next, the actions 4 and 5 appear to be only documental. They just support the I-organization of Volley and they can also trouble-free be realized in other ways. Applying the implications of the *transaction axiom* yields that the reception of an application letter does count as performing the request by the aspirant member, which is an essential C-act. The next interesting question is how the corresponding promise by (an actor in) Volley is performed. On close observation one has to conclude that this act is performed tacitly (which is, as we have seen in Sect. 2, quite common). This means practically that no confirmation letter or something similar is sent to the aspirant member. To verify that the associated commitment is indeed made, one has to ask the question what would happen if the process would break down somewhere in the part as shown in Fig. 4. Most probably, the aspirant member would contact Volley after some time of 'radio silence'. The only socially acceptable response by Volley then would be that there is a delay in the processing of the application. It would not be acceptable to say that no (implicit) promise to handle the application is made, but instead, e.g. that the application letter is just lying somewhere or even lost. So, the aspirant member may be confident that, although it may take some time, the application will be dealt with if (s)he has sent in the application (provided that there is no communication distortion [10]). If one would redesign and re-engineer the BPs of Volley it would be highly recommendable to have the promise be performed explicitly. Fortunately, modern ICT provides the means at little cost.

The analysis of the complete flow chart of Volley, or of a Petri Net or an EPC would yield the process model in Fig. 5 according to the diagramming technique as presented in [9]. This technique is actually a condensed version of Fig. 2: a C-act and the corresponding C-fact are combined in one symbol, as are a P-act and the resulting P-fact. Next, the exception steps "decline" and "reject" are omitted.

Two transactions are identified: T01 (membership start) and T02 (membership payment). Each of them follows the same standard pattern of course: request (rq), promise (pm), execution of P-act, state (st) and accept (ac). The relationship between T01 and T02 is that T02/rq is performed while dealing with T01/pm, and that performing T01/st (stating that a new membership has started) has to wait for the completion

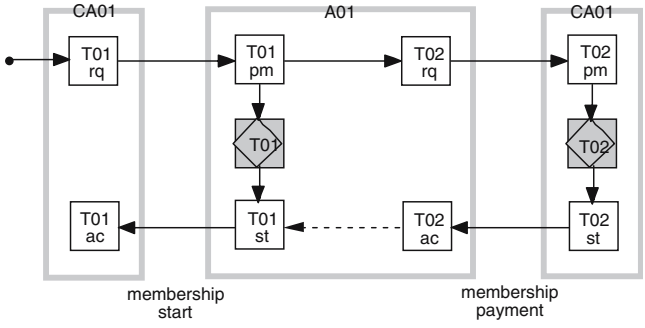

Fig. 5 Process model of the case volley

of T02/ac (having paid the fee). The grey colored rectangles represent the responsibility areas. For example, the actor role A01 is responsible for performing T01/pm, the P-act in T01, and the C-acts T01/st, T02/rq, and T02/ac.

Now that we know what a BP is, the question what a BPS is has to be answered. To this end we return to the distinction axiom, but depicted in a slightly different way, as exhibited in Fig. 6. The upper part shows the three aspect organizations: the B-organization, the I-organization and the D-organization. The additional distinction between 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

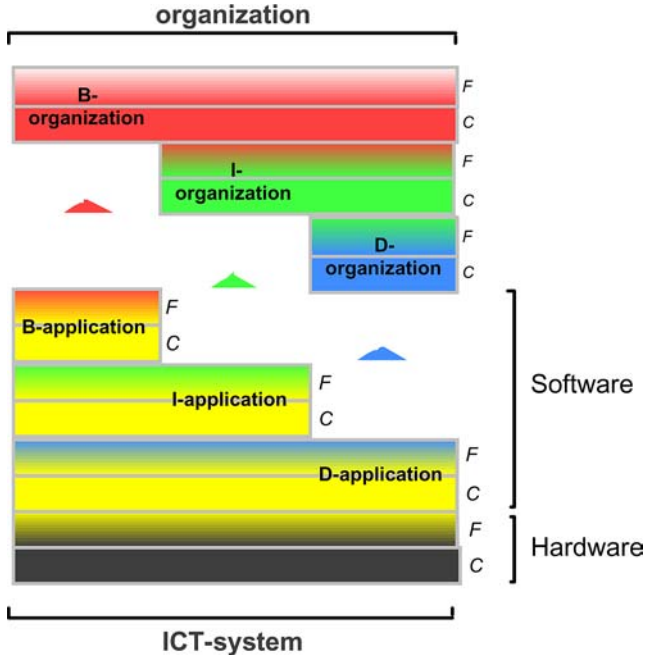

Fig. 6 Aspect organizations and supporting applications

(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 the size of bars are primarily for the sake of relating the applications 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, Fig. 6 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, spreadsheet 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.

Clearly then, a BPS is a B-application. It supports directly the BPs in the B-organization. A BPS will commonly be supported by I-applications, e.g. a management information system, and these I-applications will commonly be supported by D-applications, e.g. a database management system. To wrap it up, Fig. 7 shows the delineation of the notions of BP and BPS.

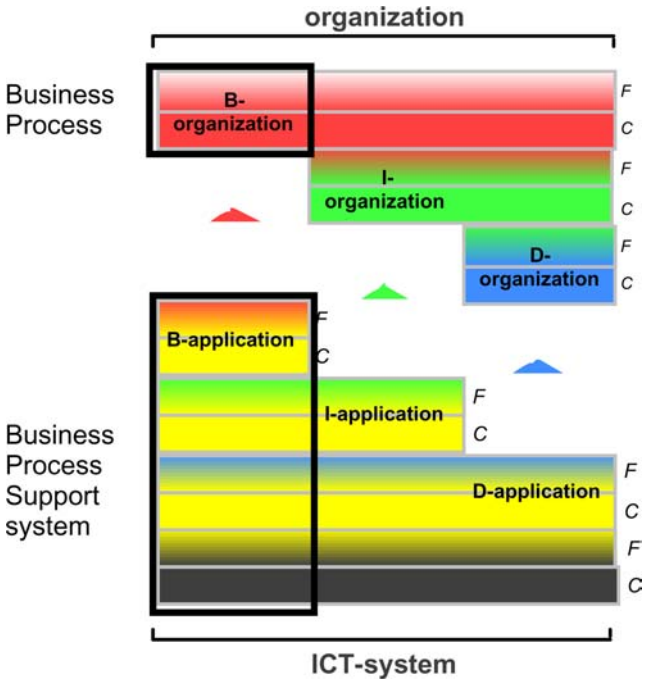

Fig. 7 Depiction of the notions of BP and BPS

3 System design and system engineering

In designing a system (of any kind) both the teleological and the ontological system definitions are relevant, or, if one likes, both the functional and the constructional perspectives on systems (Fig. 8). Next to that, the notion of layered nesting of systems, as discussed before, is of paramount importance. Figure 9 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 *object system* (OS). 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 OS in terms of the construction and operation of the US. Also by nature, these requirements are about the function and the behavior of the OS, thus in terms of a black-box model of the OS. We consider them to include the so-called non-functional requirements, regarding various performance and quality aspects. The next basic design step is to devise specifications for the construction and operation of the OS, in terms of a white-box model of the OS. A thorough analysis of this white-box model must guarantee that building the OS is feasible, given the available technology. These two steps correspond with the analysis and the synthesis step as discussed in Ref. [14]. We also adopt the important recognition that designing is an iterative process, indicated in the figure by the arrow back from synthesis to analysis. The end result of a design process is a balanced compromise

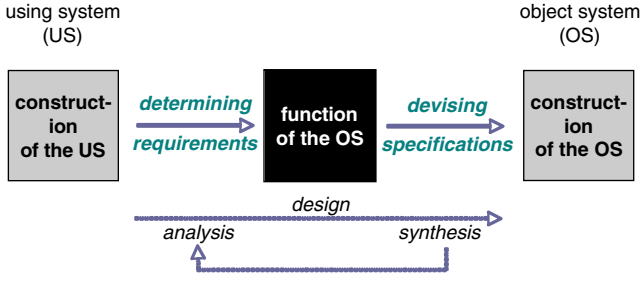

Fig. 8 The system design process

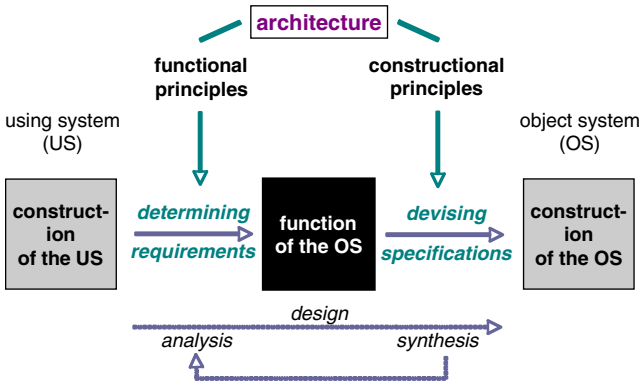

Fig. 9 The role of architecture in system design

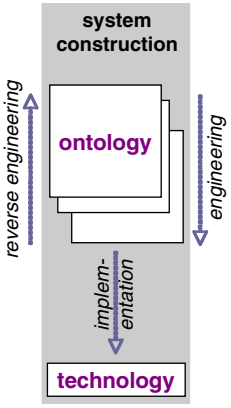

Fig. 10 The process of engineering a system

between (reasonable) requirements and (feasible) specifications.

Let us now focus on BPs and BPSs. Even if, from the perspective of the US, the set of (functional) requirements regarding an OS 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 OS is a particular accounting system. Then a possible general requirement is that all accounting systems must be in conformity with the European law. Next to that, there is generally a large amount of freedom in devising the specifications for the OS. 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.

If one searches on the Internet for the term "architecture" (in the business and ICT domain), one will find a large variety of definitions. The majority is rather synonymous to global design or blueprint, and a minority is about design principles. We adhere to the last conception and define architecture as *normativerestriction 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. To make this notion of architecture practically useful, the intended restriction must be expressed in a consistent and coherent set of *design principles*. Applying a design principle satisfies one or more general requirements regarding the design as well as the engineering of a system. Thus, some architecture may hold for many systems, typically for a class of similar systems. Actually, only in its reuse lies the practical benefit of architecture. In line with the distinction between requirements and specifications we distinguish between *functional principles* or function architecture and *constructional principles* or construction architecture (Fig. 9). An example of 'good' function architecture is the Apple Mac OS; an examples of 'bad' function architecture are the first video recorders. An example of 'good' construction architecture is the modern PC; an examples of 'bad' construction architecture are the unstructured ('spaghetti') computer programs.

After having designed a system (a B-organization or a BPS), it has to be engineered. Figure 10 exhibits the process of engineering. It consists basically of producing a coherent and consistent ordered set of white-box models of the system. The 'lowest' one is commonly called the implementation model. This model can straightforwardly be implemented on the available technological platform. For example, the implementation model of a BPS is the source code in some programming language. Likewise, the implementation model of an organization consists of the functions or task packages that can be assigned to human beings on the basis of their competencies. The 'highest' model is called the ontological model or ontology of the system. This model is fully independent of the implementation; it only shows the essential features of the system. The Ψ-theory provides the basis for making ontological models of organizations. Further discussion of the role of ontology in system engineering falls outside the scope of this paper; the interested reader is referred to Dietz and coworkers [15, 16].

4 Conclusions

The research question that has been addressed in this paper is how the notions of BP and BPS can be understood in such a way that their continuous and concurrent (re)designing and (re)engineering can be performed more effectively than what is currently the case. To exemplify the latter, we have indicated that common techniques for modeling BPs, like flow charts, Petri Nets and EPCs, are not specific for BPs. They are rightly based on the ontological system notion, but they make no distinction between surface structure and deep structure. Although we did not elaborate on modeling techniques that produce basically black-box models, like IDEF0 and DFD, it may be clear that these techniques

cannot be of any real help in changing the construction of a system (where redesign and reengineering is all about).

The Ψ-theory of organizations, as presented in Sect. 2, demonstrates first that the notion of BP can be defined and understood much more clearly and much more precisely than is usually done. The operation and the transaction axiom are the key to understanding BPs on a high level of compression but without loss of essential details. The distinction axiom provided a further refinement of this notion, leading to the definition of a BP as primarily a process in the B-organization. This process is the most stable one, the majority of changes always take place on the D-level and the I-level. The Ψ -theory also offers the key to clearly distinguish between organization (a system in the category of social systems) and ICT-system, which is ultimately a physical system but which can sensibly be taken as a data processing system. As a corollary, an ICT-system can never take over actor roles since they cannot take social responsibility. ICT-systems can only support actor roles. Therefore the terms "BP" and "BPS" are rightly chosen.

On the basis of the coherent lines of thoughts in the previous sections, we are now able to bring the needed preciseness and distinctness to the definitions of the notion of BP and BPS. A *business process* is a collection of causally and conditionally connected transactions in the B-organizations of enterprises. Since the transaction pattern, as presented in this paper, appears to be followed by social individuals (human beings) in all kinds of enterprises, we consider it to be a socionomic law. Applying this notion of BP in the analysis and design of BPs warrants that one will never include redundant process steps and that one will never forget necessary steps that were just performed tacitly in the current implementation. A *business process support system* is an ICT-application of which the functionality supports the performing of the steps in a BP, as defined above, by recording performed steps and by providing the information that is needed by the actors to perform these steps. This information regards both the C-facts and P-facts in the BP at hand as well as C-facts and P-facts from other BPs, inside or outside the organization.

Being aware of the rather unorthodox approach we take towards BPs and BPSs, we can understand the initially skeptical attitude of the reader. In a journal paper one can effectively only evoke interest for new ideas by presenting them in a scientific, consistent, and coherent way. The experience over more than 10 years in over 100 practical projects in which DEMO has been the leading methodology has provided the evidence that these new ideas also really work. More than 50 projects regarded the redesign and/or reengineering of BPs and BPSs. Unfortunately most project documentation is proprietary or in Dutch only. An exception is a case that is described in [3].

References

1. Albani A, Dietz JLG, Zaha JM (2005) Identifying business components on the basis of an enterprise ontology. Interop-Esa 2005—First international conference on interoperability of enterprise software and applications. Geneva, Switzerland (to be published by Springer Science and Business Media)
2. Dietz JLG (1999) Understanding and modelling business processes with DEMO. In: Proceedings of 18th international conference on conceptual modeling (ER 1999), Paris
3. Reijswoud van VE, Mulder JBF, Dietz JLG (1999) Speech act based business process and information modeling with DEMO. Inf Syst J
4. Davenport TH (1993) Process innovation. Harvard Business School Press, Boston
5. Keen PGW (1991) Shaping the future: business design through information technology. Harvard Business School Press, Harvard
6. Gero JS (1990) Design prototypes: a knowledge representation schema for design. AI Mag 11(4):26–36
7. Gero JS, Kannengiesser U (2004) The situated function-behaviour-structure framework. Des Stud 25(4):373–391
8. Bunge MA (1979) Treatise on basic philosophy. A world of systems, vol 4, D. Reidel Publishing Company, Dordrecht, The Netherlands
9. Dietz JLG (2003) Generic recurrent patterns in business processes. In: van der Aalst W, ter Hofstede A, Weske M (eds) Business process management, LNCS 2678. Springer-Verlag, Berlin
10. Dietz JLG (2003) The atoms, molecules and fibers of organizations. Data Knowl Eng 47:301–325
11. van der Aalst WMP, van Hee KM (2001) Workflow management: models, methods and systems. MIT Press, MA
12. Jensen K (1997) Coloured petri nets, basic concepts, analysis methods and practical use. Basic concepts. monographs in theoretical computer science, vol 1. Springer-Verlag, Berlin
13. Keller G, Nüttgens M, Scheer A-W (1991) 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)
14. Alexander C (1964) Notes on the synthesis of form. Harvard University Press, Cambridge, MA
15. Dietz JLG, Habing N (2004) A meta ontology for organizations. In: Workshop on Modeling Inter-Organizational Systems (MIOS), LNCS 3292. Springer Verlag, Larnaca, Cyprus
16. Dietz JLG (2005) System Ontology and its role in Software Development (to appear in the proceedings of the CAiSE 2005 workshop Interop-EMOI)