Enterprise Modelling and Information Systems Architecture
Vol. 3, No. 1, July 2008

J.A.P. Hoogervorst, J.L.G. Dietz

Enterprise Architecture in Enterprise Engineering

Originating from quite different fields of theory and practice, the terms "Enterprise Ontology" and "Enterprise Architecture" currently belong to the standard vocabulary of those professionals who are concerned with (re)designing and (re)engineering enterprises, thereby exploiting modern information and communication technologies for innovating products and services as well as for optimizing operational performance. Because of the inherent characteristics of modern enterprises, often operating within networks of cooperating enterprises, the task of these professionals can rightly be characterized as having to master unprecedented high complexity. The statement, put forward in the paper, that the current notion of Enterprise Architecture does not offer satisfactory help and thus need to evolve into an effective conceptual tool, is clarified in a historical context. In order to let Enterprise Architecture become a sensible, effective notion, complementary to Enterprise Ontology, it is proposed to define it conceptually as normative restriction of design freedom, and operationally as a coherent and consistent set of design principles. The new, evolved notion of Enterprise Architecture is clarified and illustrated using a case example.

1 Introduction

The traditional organizational sciences fall increasingly short in helping enterprises to implement strategies effectively, and in a controlled way. Between 70% and 90% of the strategic initiatives appear to fail, meaning that enterprises are unable to derive success from their strategy [KaNo04], [Mint94]. These high failure rates are reported from various domains: total quality management [OaPo94], business process reengineering [Burl01], [SmFi03], six sigma [Ecke01], e-business [KaRo99], customer relationship management [Kirb01], and mergers and acquisitions [WoHa02]. Whereas all too often, unforeseen or uncontrollable events are presented, for convenience sake, as the causes of failure, research has shown that strategic failure is mostly the avoidable result of inadequate strategy implementation. Rarely is it the inevitable consequence of a poor strategy [KaNo04]. A plethora of literature indicates that the key reason for strategic failures is the lack of coherence and consistency, collectively also called congruence, among the various components of an enterprise [BeES90], [GaBa98], [Hoog98], [Kauf92], [Kott95], [MiSn84], [Pett98]. At the same time, the need to operate as an integrated whole is becoming increasingly important. Globalization, the removal of trade barriers, deregulation, etc., have led to networks of cooperating enterprises on a large scale, enabled by the enormous possibilities of modern information and communication technology. Future enterprises will therefore have to operate in an even more dynamic and global environment than the current ones. They need to be more agile, more adaptive, and more transparent. Moreover, they will be held more publicly accountable for every effect they produce.

Said problems are traditionally addressed with *black box thinking* based knowledge, i.e., knowledge concerning the function and the behavior of enterprises. Such knowledge is definitely sufficient for managing an enterprise within the current range of control. However, it is totally inadequate for meeting performance goals that are outside that range, thus for changing an enterprise. In order to bring about changes in a systematic and controlled way, *white-box* based knowledge is needed, i.e., knowledge concerning the construction and the operation of enterprises. Developing and applying such knowledge requires no less than a *paradigm shift* in our thinking about enterprises, since the traditional organizational sciences are not able to ensure that enterprises are coherently and consistently integrated wholes. The needed new point of view is that enterprises are purposefully designed, engineered, and implemented systems. The needed new skill is to (re)design, (re)engineer, and (re)implement an enterprise in a comprehensive, coherent, and consistent way (such that it operates as an integrated whole), and to be able to do this whenever it is needed.

The current situation in the organizational sciences resembles very much the one that existed in the information systems sciences around 1970. At that time, a revolution took place in the way people conceived information technology and its applications [Dijk76], [Lang77]. Since then, people are aware of the distinction between the form and the content of information. This revolution marks the transition from the era of data systems engineering to the era of information systems engineering. The comparison we draw with the information sciences is not an arbitrary one. On the one hand, the key enabling technology for shaping future enterprises is the modern information (and communication) technology (IT). On the other hand, there is a growing insight in the information sciences that the central notion for understanding profoundly the relationship between organization and IT is the entering into and complying with commitments between social individuals [GoLy82], [WiFl86], [Diet06a]. These commitments are raised in communication, through the so-called intention of communicative acts. Examples of intentions are requesting, promising, stating, and accepting. Therefore, like the content of communication was put on top of its form in the 1970's, the intention of communication is now put on top of its content. It explains and clarifies the organizational notions of collaboration and cooperation, as well as notions like authority and responsibility. This current revolution in the information systems sciences marks the transition from the era of information systems engineering to the era of enterprise engineering. At the same time it enables it to converge with the traditional organizational sciences, as illustrated in Figure 1.

As said before, the basic premise of enterprise engineering is that an enterprise is a designed system. In order to ensure that the designing of a system is performed coherently and consistently, such that the resulting system is a truly integrated whole, two core notions are crucial: ontology and architecture. The *ontology* of a system is theoretically defined as the understanding of its construction and operation in a fully implementation independent way. Practically, it is the highest-level constructional model of a system,

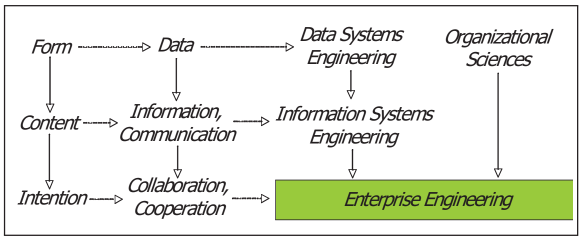

Figure 1:The roots of enterprise engineering

the implementation model being the lowest one. Compared to its implementation model, the ontological model of an enterprise offers a reduction of complexity of well over 90% [Diet06b]. The notion of Enterprise Ontology is discussed in [DiHo08]. *Architecture* is theoretically defined as the normative restriction of design freedom. Practically, it is a coherent and consistent set of principles that guide the design of a system. Any strategic initiative of an enterprise can only be made operational through applying the notion of architecture, namely, by expressing it in principles that guide the designing of the 'new' enterprise. The notion of Enterprise Architecture is discussed in Section 3. Only by applying these notions of ontology and architecture, strategic changes of enterprises can be made intellectually manageable. The purpose of this paper is to discuss this 'new' prescriptive notion of architecture and to show how it contributes to the (re)design and (re)engineering of enterprises. At the same time we will try to convince the reader that the currently adopted descriptive notion of architecture does not offer any help in these activities. Before elaborating on Enterprise Architecture as the second conceptual pillar of Enterprise Engineering, we will briefly resume some essential notions about systems and architecture in general. Subsequently, Enterprise Architecture will be discussed in Section 3. The major findings from the discussions in Sections 2, 3, and 4 are drawn in Section 5.

2 Systems and architecture

There exist many system definitions. Maier and Rechtin define a system as "a set of different elements so connected or related as to perform a unique function not performable by the elements alone" [MaRe02]. Others speak of "a set of elements standing in interrelation among themselves and with the environment" [Bert69]. Essentially, the central notion regards a set of elements having certain relationships with each other and with the environment in view of the realization of one or more goals. In view of the above, an enterprise is evidently a system.

According to one of the founding fathers of the General System Theory, a core problem facing modern science is developing a theory about 'organizing', otherwise said, a theory about 'organized complexity' [Bert69]. Based on the level of complexity, Weinberg identifies three areas [Wein01]. The first area regards (relative) low complexity, identified by Weinberg as "organized simplicity", such as exemplified by machines en mechanisms. This type of complexity can be addressed through analytical methods. At the other end of the spectrum lies the area "unorganized complexity". Here the variety is so large that complexity can be addressed through statistical methods (such as with gas molecules in a closed space, or with certain aspects of traffic). The large area between these extremes is that of "organized complexity": too complex for analytical methods and too organized for statistical methods. According to Weinberg, this area – in which enterprises are positioned – is preeminently suitable for, and in definite need of, the system approach. Therefore, the system approach offers a formal methodology to address the enterprise as a whole, while considering its constituent parts and their mutual relationships, in order to safeguard a unified and integrated system. Many authors submit that the system approach is the only meaningful way to address aforementioned core problem of modern science, hence to study and develop enterprises [Bert69], [Bung79], [Ghar99], [Rech00]. Ackoff argues therefore that the failing strategic initiatives mentioned earlier are due to the fact that the initiatives are fundamentally "anti-systemic" [Acko99].

As indicated, unity and integration, hence coherence and consistence between various enterprise aspects, is crucial for the success of the enterprise as a *whole*. There will be little debate about the conviction that unity and integration is not brought about spontaneously but has to be *designed* intentionally. Said intentional design aspect lies at the heart of the not 'accidental' but intentional character of the enterprise as a system, whereby realization of the enterprise function and construction should understandably not depend on chance but, as mentioned in Section 1, must be purposefully established. This begs the inescapable question of *how* the enterprise, in view of its goals and the required unity and integration, must be designed. The answer to this design question will be discussed later. For now we like to emphasize that any answer is essentially *normative*. We agree with Jackson arguing that the normative aspect of system design must be made explicit [Jack03]. Architecture, in our opinion, offers the formal answer to this necessity.

The emphasized normative character of the answer to the question how the enterprise should be designed implies that the answer guides the design process, hence limits design freedom. Herein lies the essence of architecture as mentioned in Section 1: conceptually it regards the normative restriction of design freedom. From a general system perspective, architecture is practically defined *as a coherent and consistent set of principles that guides system design*. The set must be coherent, meaning that it must form a unified totality, but also consistent, since principles should not be mutually conflicting.

Noticeably, architecture is often viewed as a 'blueprint' or a schematic depiction of the essential components of a design and their relationships. Apparently, the concept is used in a descriptive manner. Sometimes both viewpoints are used in one definition. For example the IEEE standard 1471 reads [MaRe02]: *Architecture is the fundamental organization of a system embodied in its components, their relationships to each other and to the environment, and the principles guiding its design and evolution*. To our view, this definition is equivocal; it tries to accommodate two very different points of view, something that should be avoided in any definition. Similar remarks can be made about the definition of architecture provided by the Open Group [TOG03]. Obviously, within the formal approach to architecture presented in this paper, architecture must be considered as a *prescriptive* concept, that – through design principles – *dictates* ex ante how a system must *become*, rather than a *descriptive* concept that *describes* ex post how a system *is*. We feel the descriptive use of the architecture concept is of little value from a design perspective, since the descriptive notion is essentially passive, hence – based on *after* the fact description – cannot provide *prior* active guidance in the design process.

In view of the above, architecture is essentially linked to the system concept. First, design guidance is evidently crucial for establishing system unity and integration. Second, system design must satisfy various requirements. These requirements do not only regard the system function, but also regard objectives pertinent to certain areas of concern. For a technical system, these concerns might be reliability, maintainability, or safety. Through architecture, these areas of concern are addressed, hence requirements are operationalized. Further, architecture ensures that areas of concern and their associated – possibly conflicting – requirements are addressed explicitly, and in a balanced manner. Figure 2 schematically shows the role of architecture in the generic system development process. It should be viewed as the fundaments of a design theory.

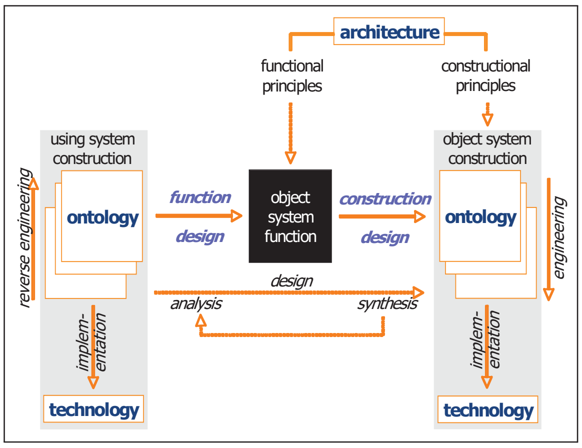

Figure 2: The generic system development process

The essential aspects of the generic system development process have been outlined in [Diet06a] and [DiH08]. After having designed a system at the ontological level, it has to be further engineered through detailed design (called engineering in Fig. 2), such that the system can be implemented. Detailed design basically regards 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 an information system is the source code in some programming language. Likewise, the implementation model of an enterprise 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.

Applying a design principle satisfies one or more requirements regarding the global design (referred to as "design" in Figure 2) as well as the detailed design (referred to as "engineering" in Figure 2) of a system. In line with the distinction between function and construction, we distinguish between *functional principles* or function architecture, and *constructional principles* or construction architecture. An example of a product that exhibits 'good' function architecture is the Apple MacOS; an example of a product that exhibits 'bad' function architecture is the (first) video recorder. An example of a product that exhibits 'good' construction architecture is the modern PC, whereas 'bad' construction architecture is exhibited by unstructured ('spaghetti') computer programs.

Understandably, the nature of a specific architecture is contingent upon the system category. Several system categories exist, such as mechanical, chemical, electronic, IT, or socio-technical systems [Bung79]. In our view, the concept of architecture becomes most useful if architecture does not apply to the design of one system, but holds for all systems of a particular type, within some category. Put differently, architecture holds for a certain *class* of systems. So, for example, IT architecture for data warehouses or applications is not only intended for one specific data warehouse or application, but intended for the class of data warehouses and applications respectively. IT architecture thus regards the normative design guidance for the class of IT systems. Likewise, enterprise architecture refers to the class of enterprises. Devising architectures can appropriately be labeled as *architecturing*, for which system design domains and areas of concern serve as the guiding context. This activity must be clearly distinguished from designing. This follows from the notion that architecture provides design guidance, hence must logically precede design. Moreover, architecturing is an autonomous

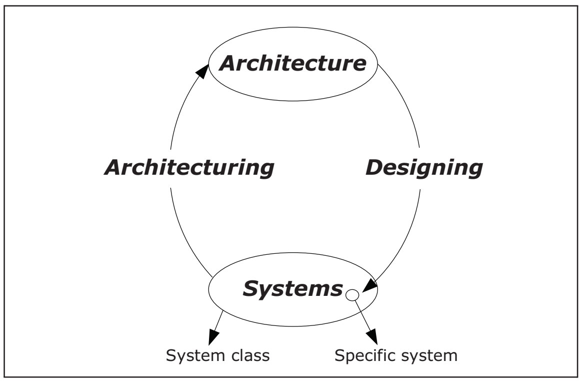

Figure 3: Architecturing and designing

activity that can be performed rather independently from design projects. It seems plausible to call the architecturing person an *architect*. Designing thus concerns the realization of a specific system within a certain system class, using architecture as a normative guidance. Summing up, the result of architecturing is an architecture, the result of designing is a design. Figure 3 schematically shows the difference between architecturing and designing.

3 Enterprise Architecture

3.1 The essence

The second tool for mastering the complexity of contemporary enterprises, next to Enterprise Ontology, is Enterprise Architecture (cf. Sec. 1). Contrary to Enterprise Ontology, it is abundantly discussed already in the literature about managing organizational change. Unfortunately, the term "Enterprise Architecture" has got many meanings also, meanings that are quire divers and sometimes even contradictory. Similarly, also the term "Enterprise Architect" is used in many different ways. To quote Thomas Kuhn, the current application of the concept of enterprise architecture does not yet show the characteristics of a 'normal science' [Kuhn70]. In addition, the acceptance and application of Enterprise Architecture differs largely. Gartner reports [Drob02]: *"People talk about enterprise architecture as if it is easily understood by technology and business professionals alike. In reality, technology professionals have a wide-ranging view of enterprise architecture…In contrast, business professionals tend to ignore the term as an IT-only issue".*

The interpretation of enterprise architecture given in this article aims to clarify this theme in view of the general notion of architecture outlined in Section 2. Subsequently, we will use the case of the Educational Administration, introduced when discussing

Enterprise Ontology [DiHo08], for illustrating Enterprise Architecture.

Goal-oriented, coordinated activities of human endeavor are identified through various labels: a business, company, organization, or institution. In this article we use the, also frequently used, term 'enterprise'. Various entities that carry out commercial, non-profit or governmental activities are thus identified as enterprises. As we have mentioned in Section 1, the success rate of strategic initiatives is poor. Lack of coherence and consistency between the various aspects of an enterprise was identified as a primary cause for failures. According to the congruence theorem, enterprises will operate more effectively and perform better the higher the degree of unity and integration, hence the degree of coherence and consistency among its constituent parts [NaTu97]. These observations are likewise valid – if not even more so – if technology is part of strategic initiatives. Research about the effectiveness of technology deployment teaches us that effectiveness can only be obtained if the enterprise context in which technology operates matches and integrates with the technology and vice versa [Mort91]. Said condition is patently manifest regarding the impact of IT. Rather remarkable is the outcome of research conducted over a number of years showing no relationship between IT investments and measurable improvements in enterprise performance [PiSt03].

In our opinion, the underlying cause for this observation has to do with the suboptimal utilization of technology. Excellent performing enterprises, however, use technology in such a way that consistency and coherence is created between technology and the context in which it operates. This perspective is supported by earlier MIT research regarding the influence of IT on enterprise productivity. Only enterprises that in conjunction with the introduction of IT also changed the complementary enterprise context – hence realized an integrated enterprise design – achieved considerable gain in productivity [BrHi96]. Comparable results are reported pertinent to the introduction of IT in the area of Customer Relationship Management [Marc01], [KMP+03]. Enterprise performance therefore does not primarily depend on the use of modern technology, but depends on the overall quality of the enterprise design, of which technology is an important aspect. Herein lies the essential role of Enterprise Architecture: providing design guidance for establishing coherent and consistent enterprise design. Further, Enterprise Architecture determines how the implementation-independent design on the ontological level can be practically operationalized on the level of implementation. Noticeably, it is through enterprise architecture that the design becomes manifest. Otherwise said, identical enterprises at the ontological level (similar basic function) can only be implemented differently (and experienced by customers as differently) through different architecture. The well-known distinction between 'mechanistic' and 'organismic' ways of organizing manifests fundamentally different sets of architecture [Burn90].

In line with the general definition of architecture given in Section 2, we define Enterprise Architecture *as a coherent and consistent set of principles that guide how the enterprise must be designed*.

The ability to realize coherence and consistency in enterprise design will become more and more important. Progress in IT is, and will continue to be, a considerable driver in this respect. Said progress has led for example to the emergence of 'virtual' enterprises, and networks of enterprises ('extended' enterprise) where business partners and suppliers cooperate. Through various interfaces employees and customers are interacting and collaborating within and with these networks. This situation confronts enterprises with an enormous *integration* problem. Our observations about failing enterprise initiatives suggest that integration is more than establishing IT system 'interoperability'. We contend that addressing aforementioned integration problem without the concept of enterprise architecture will continue the astonishing high strategic failure rate mentioned before. Rightly so, various governments emphasize the importance of enterprise architecture for realizing governmental operation in a unified and integrated manner [USGA03], [MSTI03].

3.2 Enterprise architecturing

As indicated, architecture guides system design – hence applies to one or more design domains – and addresses requirements following from the system function and areas of concern. This similarly holds for systems like enterprises. Areas of concern have to do with strategic intentions and goals. Otherwise said, through strategy development insight emerges about important areas of concern, such as flexibility, customer satisfaction, time to market of new products and services, costs, or compliance to regulatory requirements. Congruent with the general system perspective, we might say that the areas of concern identify topics that necessitate certain (possibly yet to be specified) enterprise behavior. Figure 4 shows some typical areas of concern relevant for enterprises [Hoog04].

The normative guidance of the design process through architecture will be effectuated within a number of design domains. From the perspective of the enterprise as a whole, four main design domains can be identified:

Business. This domain regards the primary enterprise *function*, and has to do with the teleological

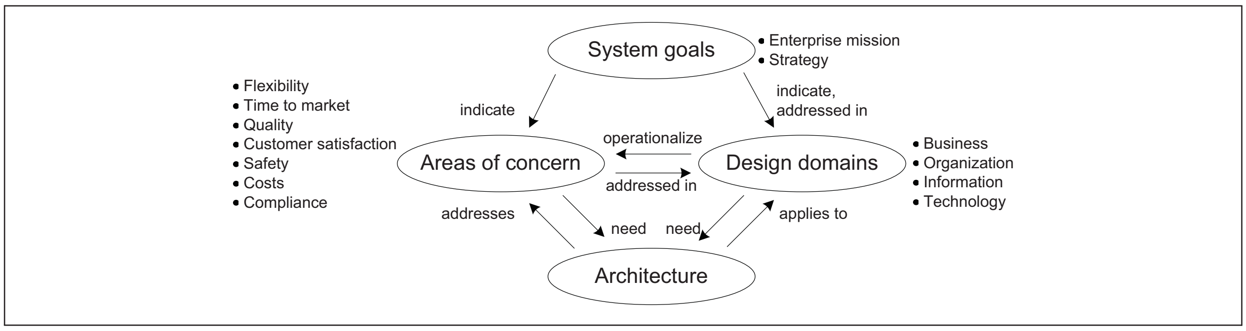

Figure 4: Reference framework for architecturing

(black-box) system perspective mentioned in Section 1. Aspects, such as products and services, delivery channels, customers, the economic model, as well as general relationships with the environment (market, competitors, stakeholders) are typical areas of attention within the business design domain. In view of our overall notion of architecture, the *business architecture* is defined as a coherent and consistent set of principles that prescribes how a certain domain of goal oriented activities must be exploited and explored. Recalling our observations in Section 2, the business architecture can be regarded as the function architecture of the enterprise.

Organization. Given a certain primary enterprise function, various degrees of freedom exist regarding the question how the production for bringing about the products and services is actually organized. The organizational design domain deals with the internal arrangements of the enterprise, having for example to do with processes, employee behavior, organizational culture, management practices, human resource management, and various provisions, such as pertinent to financial, accounting, or reward structures. The *organization architecture* can be defined as a coherent and consistent set of principles that prescribes how the enterprise must be arranged.

Information. Within both the business and organizational design domain, information is a crucial factor. Various facets play a role, like the structure and quality of information (syntax, semantics, security), the management of information (acquisition, storage, and distribution), as well as the utilization of information (presentation, exploitation en exploration). Comparatively, *information architecture* is defined as a coherent and consistent set of principles that prescribes how information must be handled.

Technology. Evidently, technology is essential for current business, organizational and informational support, as well as for future developments in these domains. Specific technology entails specific associated architecture for providing the normative guidance regarding design activities for the technology in question. Taking information technology (IT) as an example, we have *IT architecture*, defined as a coherent and consistent set of principles that prescribe how IT systems must be designed.

An important distinction mentioned in Section 1 is that between the system *function* and the system *construction*. As identified in the above, the business domain regards the function of the enterprise: *what* products and services are delivered. The other three main design domains – organization, information, and technology – concern the construction of the enterprise: *how* products and services are brought about. Hence, the organization, information, and technology architectures constitute the construction architecture, mentioned in Figure 2. Said distinction is schematically depicted in Figure 5.

Lack of unity and integration has been identified in the above as a core reason for failing enterprise (strategic) initiatives. Avoiding these failures requires coherence and consistency within and between the main design areas. This must be safeguarded through coherent and consistent enterprise architecture, comprised of business, organization, information, and technology architectures. Overall coherence and consistence implies that important mutual relationships exist between the main design domains, like Figure 5 indicates.

As said, the specific character and description of a design domain depends on the system type and the level

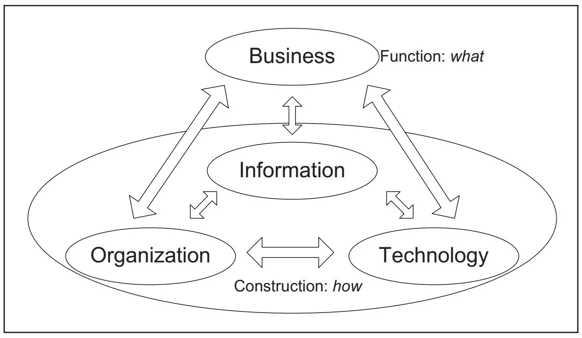

Figure 5: Main enterprise design domains

of perception. The perception can be directed to the total system or only part of it, such as a subsystem. So for a car, the design domains engine, chassis, interior, wheels, brakes, lamps, windows, etc., are necessary and sufficient design domains by observing the car as a whole. When observing a design domain in more detail also more detailed design domains play a role. For example, within the design domain 'engine' design domains like 'piston' and 'crankshaft' are relevant. So, from the perception of the car as a whole the design domain 'engine' suffices, but for designing the engine, more detailed design domains have to be defined. This process continues until a level is reached whereby further breakdown is not warranted. Evidently, the same notion holds for enterprise design domains.

As the illustration shows, there is specialization of design domains associated with more detailed observations. Such specialization thus creates a certain order whereby a more detailed design domain is subordinated under the next higher design domain, like 'engine' is subordinated under the overall design domain 'car', and 'piston' in turn is subordinated under the domain 'engine'. Since architecture pertains to one or more design domains, aforementioned order likewise holds for architecture. That is to say that a principle or standard a_j may not be in conflict with principle or standard a_i if $D_j \subset D_i$. Noticeably, this is an important condition for safeguarding coherence and consistency, which has been emphasized before as an important objective of defining architecture. Establishing unity and integration does not only require that architecture forms a coherent and consistent set, but also requires that the set of design domains pertinent to a chosen perception is *complete*: necessary and sufficient in view of the system purpose (function) and the objectives associated with the areas of concern. For complex systems, such as enterprise, establishing completeness is far from easy. Nonetheless, in all cases the total set of design domains within a certain perspective must be complete. Otherwise said, the set of design domains for a certain perspective must be necessary and sufficient for addressing relevant design activities pertinent to the given perspective. For the enterprise as a whole, we feel the four main design domains are necessary and sufficient. More specific design domains are obviously relevant within the four main design domains in order to carry out particular and definitive design activities (higher level of detail). For example, the design domain 'processes' can be seen a more specific design domain within the main design domain 'organization'. Process architecture is thus a subset of the organization architecture. Examples of specific enterprise design domains have been discussed elsewhere [Hoog04]. Verifying that the set of design domains is complete for a given perception is not always easy, specifically for an enterprise. Nonetheless, the completeness requirement holds and must be satisfied for every perception.

3.3 The case educational administration

The first remark to be made regarding the application of the notion of Enterprise Architecture in practice is that it is just emerging. Companies that have adopted it are still in the phase of understanding what it means: they are pioneers. Yet the potentials are enormous [Hoog04]. In fact, as argued previously, it seems to be the only feasible way of 'translating' high-level statements and areas of concern, as can be found in mission and strategy documents, into operationally useful principles for design.

It is possible that an area of concern directly identifies a design domain where the concern is addressed. Generally however, this is not the case. For example, it appears not immediately clear how concerns about customer satisfaction, flexibility, or societal responsible business conduct are actually operationalized, and which design domains are involved. Hence, pertinent to the main enterprise design domains, the enterprise architect must identify subsequent design domains that enable addressing the concerns through design principles. Defining the relevant design domains requires broad, design-oriented knowledge regarding business, organization, information, and technology. Following example for the Educational Administration might serve as an illustration.

Suppose the process of (University) strategy development has (among other ones) identified 'student satisfaction', 'flexibility', and 'compliance' as areas of concern. The first concern has to do with the ability to attract and retain students, whereas the second concern addresses the University's ability to quickly adapt to changing internal and external conditions. Further, the area of concern 'compliance' has to do with satisfying regulatory requirements. Subsequent analysis by enterprise architects concludes that the concern 'student satisfaction' will be addressed through three design domains: 'products' (part of main business design domain), human resource management (part of main organization design domain), 'processes' (part of the main organization design domain), and 'information exploitation' (part of main information design domain). Further analysis reveals that the drive for student satisfaction necessitates easy access of students to the University network. An additional concern 'security' is thereby identified, and addressed through the design domains 'information quality' (part of the main information design domain), and 'IT security' (part of the main technology design domain). The concern 'flexibility' is addressed through the design domains 'processes' and 'human resource management'. Finally, within the design domain 'processes' the concern for compliance will be taken into account. Ultimately, architecture pertinent to the design domains mentioned, determines how the concerns are addressed in the actual design. Examples of architecture principles are shown in Table 1 below. As the table shows, an architecture principle can address more than one area of concern, as we discussed before.

Area of concern	Design domain	Architecture principle
Student satisfaction	Business (products)	Products and services must allow personalization by students
	Organization (HRM)	Management must enable employee self-management
	Organization (processes)	Student administration must be locally present, under unified, central governance
	Information (exploitation)	Complete and up-to-date student information must be available at all student contact points
Security	Information (quality)	Student information must be available from one unified source
	Technology (IT security)	Network access must be based on authentication and role-based authorization
Flexibility	Organization (processes)	Process flow control logic must be separated from process execution logic
		Student administration must be locally present, under unified, central governance
	Organization (HRM)	Employee decision making must take place at the lowest possible level
Compliance	Organization (processes)	Admission rules must be in accordance with the Bologna treaty

Table 1: Examples of architecture principles

Defining detailed design domains within the four main enterprise design domains, and more specifically, defining the appropriate architecture pertinent to areas of concern, is not a simple analytical or algorithmic process. Experience plays an important role. Architecture principles are thus (also) based on experience, best practices, and insights that can be generalized into architecture. Architecturing is thus for a considerable part a *heuristic* process [MaRe02]. As said earlier, enterprise design affects various stakeholders: customers, employees, suppliers, etc. Orchestrating the role and input of stakeholders within the process of architecturing implies that next to the heuristic character, architecturing also has a *participative* character [MaRe02].

3.4 Architecture frameworks

The core aspects regarding architecture and architecturing define the elements of a so-called 'architecture framework'. These elements are: (1) the system type S, the design domains D, and (3) the areas of concern A. An architecture framework can be shortly identified as <S,D,A> [Diet04].

As indicated, specific design domains depend on the perception chosen. Hence, for a given system, multiple (related) architecture frameworks are possible. An architecture framework can be defined as *a conceptual structure pertinent to a certain system type, consisting of areas of concern and a necessary and sufficient set of design domains pertinent to a chosen perception*.

All too often, architecture frameworks do not satisfy the definition given before. It can be noticed that the label 'enterprise architecture' is used in cases that merely regard IT architecture for the whole enterprise. Despite the label 'enterprise' this appears not to be the system type of concern. This seems to be the case with the TOGAF architecture framework [TOG03]. Frequently design domains 'business' and 'process' are added to a set of IT system design domains (like 'data and 'application') in view of the fact that IT systems support business processes. For example, the architecture framework provided by Tapscott and Caston speaks of business architecture, process architecture, application architecture, information architecture, and technology architecture [TaCa93]. Comparatively, the TOGAF architecture framework identifies business, application, data, and technology architectures [TOG03]. Ignoring the somewhat illusive use of the term 'business' in these frameworks, we notice that from an IT system design perspective the set of design domains is incomplete, since more design domains are relevant than merely 'application' and 'data'. Apart from this incompleteness, it seems clear that the design domains 'business' and 'process' cannot be part of an IT system, hence, do not fit within an IT architecture framework. One might alternatively consider the frameworks mentioned as enterprise architecture frameworks. However, in that case considerably more design domains are relevant than only 'business' and 'process'. All in all, it is often rather unclear to which system type an architecture framework pertains, while for a given (or assumed) system type, the set of design domains is all too often incomplete. In fact, the very notion of design domains and the necessity for completeness appears to be absent in the architecture frameworks mentioned. These observations also hold for Zachman's architecture framework [Zach97].

More fundamentally, one might observe that many architecture frameworks (like the ones mentioned) do not use the normative notion of architecture. The absence of a formal, normative approach consequently also implies the absence of formal design guidance. Important objectives that the normative, prescriptive approach tries to establish are thus not addressed. This is detrimental for the design process itself, but also for professionalizing the architecture function and the architect's profession.

Finally, also development, implementation, and project related (planning) issues are frequently part of architecture frameworks, such as in the TOGAF and Zachman architecture frameworks. Evidently, these aspects have to be addressed professionally, but fall outside the scope of architecture and architecturing, hence should not be part of an architecture framework. Much of what is positioned as an architecture framework is in fact a concealed development and implementation framework. Development and implementation aspects are important areas of attention within an overall enterprise governance competence, within which also the enterprise architecture competence is positioned. Reflecting on enterprise governance is however outside our current scope of discussion.

The use of a architecture framework of the type <S,D,A> is important for a number of reasons. A framework structures the process of architecturing by making explicit formal attention to: (1) the system type for which architecture must be defined, (2) the areas of concern that must be addressed, and (3) the necessary and sufficient set of design domains where architecture must be applied. Next to structuring the architecturing process, another key purpose of an architecture framework is the following. The importance of coherence and consistency of design principles and standards has been emphasized. These are essential conditions for a unified and integrated system operation. The explicit structure of the architecture framework enables safeguarding and assessing the coherence and consistency of architecture within, and between design domains, as well as between different frameworks. For an enterprise this regards coherence and consistency between business architecture, organization architecture, information architecture, and technology architecture. Safeguarding coherence and consistency is no sinecure, specifically for complex systems. This is definitely the case with enterprises. Knowledge and experience of the architect play a crucial role, as well as the ability to assess consequences of design principles and standards in one design domain for other domains. A participative, multi-disciplinary approach is also here relevant.

4 Conclusions

The notions of ontology and architecture, particularly the notions of Enterprise Ontology and Enterprise Architecture, can be very powerful conceptual instruments, provided they are conceived appropriately. The current state-of-the-art unfortunately shows a large variety of definitions, which are often contradictory. Most importantly, however, they suffer from being ill defined and not rigorously founded in an all-encompassing theory. As a consequence, they are often not appropriate and therefore do not offer effective help to the professionals whose task it is to (re)design enterprises.

The first step towards defining the terms such that they constitute an effective and complementary pair has been the presentation and discussion of the Generic System Design Process (Section 2). From this framework the clear and unavoidable conclusion can be drawn that two notions are crucial for conceptually managing the development of systems of any kind. One is the notion of understanding the construction and the operation of a system in a way that is fully independent of its implementation, while exhibiting comprehensibly, coherently, consistently, and concisely the essence of the system. The other one is the notion that unified and integrated design is crucial for the performance of the enterprise as a whole and crucial for addressing mission and strategy related initiatives and areas of concern that are valid at some point in time.

The discussion of the notions of Enterprise Ontology and of Enterprise Architecture in Sections 3 and 4 respectively, has demonstrated that without them the complexity the said professionals are faced with can hardly be mastered. We have also shown that both notions can be defined very precisely and very consistently. At the same time, we have noticed that the practical application of Enterprise Architecture, as proposed in this paper, is still in its infancy. However, its future looks bright. A lot of research has to be undertaken yet, but it is the only feasible way to arrive at a practically effective notion of architecture, as effective as Enterprise Ontology.

Enterprise Ontology and Enterprise Architecture are taken as the basic elements of the new discipline of Enterprise Engineering, that is currently emerging from the convergence of the traditional organizational sciences and the information sciences. The word "Engineering" has to be taken in a broad sense, like it is used, e.g., in Mechanical Engineering and in Industrial Engineering. The most important premise in the notion of Enterprise Engineering is that an enterprise is a designed system instead of an organically growing entity. We hope that this paper contributes to evoking the necessary awareness among the professionals and scholars who are currently dealing with organizational change that the engineering approach we have presented is the right one for coping with today's and tomorrow's complexity.

REFERENCES

[Acko99] Ackoff, R.L.: Ackoff's Best – His Classic Writings on Management, New York, Wiley 1999.

[BeES90] Beer, M.; Eisenbach, R.A.; Spector, B.: Why Change Programs Don't Produce Change, Harvard Business Review, November/December 1990, pp. 158–166.

[Bert69] Bertalanffy, L. von: General Systems Theory, New York, George Braziller 1969.

[BrHi96] Brynjolfsson, E.; Hitt, L.: Paradox Lost? Firm Level Evidence on the Returns to Information System Spending, Management Science, Vol, 42, No. 4, 1996, pp. 541–558.

[Bung79] Bunge, M.A.: Treatise on Basic Philosophy, Vol.4 – A World of Systems, D. Reidel Publishing Company, Dordrecht, The Netherlands 1979.

[Burl01] Burlton, R.T.: Business Process Management, Indianapolis, Sams Publishing 2001.

[Burn90] Burns, T.: Mechanistic versus Organismic Structures (1963). In: Pugh, D.S., Organizational Theory, London, Penguin Books 1990.

[Diet04] Dietz, J.L.G.: The Extensible Architecture Framework (xAF), Version 2, Delft University of Technology 2004 (http://www.naf.nl/content/bestanden/xaf-1.1_fe.pdf).

[Diet06a] Dietz, J.L.G.: Enterprise Ontology – theory and methodology, Springer-Verlag Heidelberg, Berlin, New York 2006.

[Diet06b] Dietz, J.L.G.: The Deep Structure of Business Proc-esses. In: Communications of the ACM, May 2006, Vol. 49, no. 5, pp 59–64.

[DiHo08] Dietz, J.L.G.; Hoogervorst, J.A.P.: Enterprise Ontology in Enterprise Engineering, (accepted for ACM-SAC track OE, March 2008).

[Dijk76] Dijkstra, E.W.: A Discipline of Programming, Prentice-Hall series in Automatic Computation, New Jersey, 1976.

[Drob02] Drobik, A.: Enterprise Architecture – The Business Issues and Drivers, 2002, http://www.gartner.com/DisplayDocument?id=366199.

[Ecke01] Eckes, G.: The Six-Sigma Revolution, New York, Wiley 2001.

[GaBa98] Galliers, R,D.; Baets, W.R.: Information Technology and Organizational Transformation, Chichester, Wiley 1998.

[Ghar99] Gharajedaghi, J.: Systems Thinking, Boston, Butterworth Heinemann 1999.

[GoLy82] Goldkuhl, G.; Lyytinen, K.: A language action view of information systems. In Ginzberg, M., Ross, C.A. (Eds.), Proceedings of the 3rd international conference on information systems, TIMS/SMIS/ACM, 1982.

[Hoog98] Hoogervorst, J.A.P.: Quality and Customer Oriented Behavior. Towards a Coherent Approach for Improvement, Delft, Eburon 1998.

[Hoog04] Hoogervorst, J.A.P.: Enterprise Architecture – Enabling Integration, Agility and Change, Journal of Cooperative InformationSystems, Vol. 13, No. 3, 2004, pp. 213–233.

[Jack03] Jackson, M.C.: Systems Thinking, Chichester, Wiley 2003.

[KaRo99] Kalakota, R.; Robinson, M.: E-Business. Roadmap for Success, Reading MA, Addison-Wesley 1999.

[KaNo04] Kaplan, R.S.; Norton, D.P.: Strategy Maps, Boston, Harvard Business School Press 2004.

[Kauf92] Kaufman, R.S.: Why Operations Improvement Programs fail – Four managerial Contradictions, Sloan Man-agement Review, Vol. 34, No. 1, 1992, pp. 83–93.

[Kirb01] Kirby, J.: Implementing CRM – Business Change Program, Not Project, Gartner Research, July 2001.

[KMP+03] Knox, S.; Maklan, S.; Payne, A.; Peppard, J.; Ryals, L.: Customer Relationship Management, Oxford, Butterworth Heinemann 2003.

[Kott95] Kotter, J.P.: Leading Change – Why transformation Efforts Fail, Harvard Business Review, Vol. 71, No. 2, 1995, pp. 59–67.

[Kuhn70] Kuhn, T.: The Structure of Scientific Revolutions, Chicago, Chicago University press 1970.

[Lang77] Langefors, B.: Information System Theory. Information Systems 2, 207–219 (1977).

[Marc01] Marcus, C.: CRM Deployment Can Focus on Most Valued Customers, Gartner Research July 2001.

[MaEH01] Maier, M.W.; Emery, D.; Hilliard, R.: Software Architecture – Introducing IEEE Standard 1471, IEEE Computer, April 2001,Vol. 34-4, pp 107–109.

[MaRe02] Maier, M.W.; Rechtin, E.: The Art of Systems Architecting, Boca Raton, CRC Press 2002.

[MiSn84] Miles, R.E.; Snow, C.C.: Fit, Failure and the Hall of Fame, California Management Review, Vol. 26, No. 3, 1984, pp. 128−145.

[Mint94] Mintzberg, H.: The Rise and Fall of Strategic Planning, New York, The Free Press 1994.

[MSTI03] Ministry of Science, Technology and Innovation: , White Paper on Enterprise Architecture, Copenhagen, Denmark 2003.

[NaTu97] Nadler, D.A.; Tushman, M.L.: Competing by Design − The Power of Organizational Architecture, New York, Oxford University Press 1997.

[OaPo94] Oakland, J.S.; Porter, L.J.: Cases in Total Quality Management, Oxford, Butterworth-Heinemann 1994.

[Pett98] Pettigrew, A.: Success and Failure in Corporate Transformation Initiatives. In: Galliers, R.D.; Baets, W.R.J.: Information Technology and Organizational Transformation, Chichester, Wiley 1998.

[PiSt03] Pisselo, T.; Strassmann, P.: IT Value Chain Management - Maximizing the Value from IT Investments, New Canaan, The Information Economics Press 2003.

[Rech00] Rechtin, E.: Systems Architecting of Organizations, Boca Raton, CRC Press 2000.

[Mort91] Morton, M.S. Scott: The Corporation of the 1990s, New York, Oxford University Press 1991.

[SmFi03] Smith, H.; Fingar, P.: Business Process Management − The Third Wave, Tampa Fl., Meghan-Kiffer Press 2003.

[TaCa93] Tapscott, D.; Caston, A.: Paradigm Shift - The New Promise of Information Technology, New York, McGraw-Hill 1993.

[TOG03] TOGAF - The Open Group Architecture Framework (2003) http://www.opengroup.org/bookstore/catalog/g051.htm.

[USGA03] USA, A Framework for Improving Enterprise Architecture Management, Version 1.1, US General Accounting Office, Washington D.C., April 2003.

[Wein01] Weinberg, G.M.: An introduction to General Systems Thinking, New York, Dorset House Publishing 2001.

[WiFl86] Winograd, T.; Flores, F.: Understanding Computers and Cognition − A New Foundation for Design. Ablex, Norwood NJ, 1986.

[WoHa02] Woolridge, L.; Hayden, F.: How to get value from mergers, acquisitions and divestments, CSC Research Journal, November 2002.

[Zach97] Zachman, J.: Concepts of the Framework for Enterprise Architecture, Zachman International Inc. 1997.

J.A.P. Hoogervorst

Sogeti
The Netherlands
P.O. Box 76
NL−4130EB Vianen
jan.hoogervorst@sogeti.nl

J.L.G. Dietz

Delft University of Technology
The Netherlands
P.O. Box 5031
NL−2600GA Delft
j.l.g.dietz@tudelft.nl