Inf Syst Front (2006) 8:67–79
DOI 10.1007/s10796-006-7971-1

Management and enterprise architecture click: The FAD(E)E framework

Frank G. Goethals · Monique Snoeck · Wilfried Lemahieu · Jacques Vandenbulcke

Abstract Enterprises are living things. They constantly need to be (re-)architected in order to achieve the necessary agility, alignment and integration. This paper gives a high-level overview of how companies can go about doing 'enterprise architecture' in the context of both the classic (isolated) enterprise and the Extended Enterprise. By discussing the goals that are pursued in an enterprise architecture effort we reveal some basic requirements that can be put on the process of architecting the enterprise. The relationship between managing and architecting the enterprise is discussed and clarified in the FAD(E)E, the Framework for the Architectural Development of the (Extended) Enterprise.

Keywords Enterprise architecture framework · Management levels · Architecture descriptions · FADE · FADEE · Framework for the Architectural Development of the Extended Enterprise · Zachman

1. Introduction

Nowadays, it is repeatedly stated that organizations should be agile, integrated and aligned (see e.g. Weill and Broadbent, 1998; Lipschutz, 2004). Unfortunately, achieving these three goals is not an easy task. Exponents of the enterprise architecture discipline have dropped major tips on how to go about pursuing these goals (see e.g. Zachman, 2004, 1987; Cook, 1996). John Zachman for example has been travelling around the globe for more than a decade to preach his course on "Enterprise Physics 101", arguing that companies cannot expect to achieve alignment, integration, or flexibility if they do not explicitly architect their enterprise (and their information systems) with these goals in mind.

As the name implies, the discipline called *enterprise architecture* looks at the architecture of the total enterprise. Basically this means that everything that has to do with the enterprise *can* be architected in an enterprise architecture effort. Typically the attention goes to architecting the business and the Information Technology (IT), rather than to the architecture of buildings, consumer products, etcetera. This is also the case in this paper.

Throughout the last decade, many enterprise architecture frameworks have been proposed by various authors. These frameworks have a number of issues in common. The enterprise architecture discipline is for example inextricably bound up with the concept of viewpoints. That is, enterprise architecture pays attention to the different views different people (typically business people vs. IT staff) have on the enterprise. Unfortunately, the architectural artefacts (the architecture descriptions) have become the centre of attention, rather than the practice of doing enterprise architecture itself. Of course, knowing which architecture descriptions ("models") to create is important; but not knowing how to get to the models, how to get buy-in for the architecture effort, and how to use the models obstructs companies from achieving advantages from the architecture effort. The creation of models takes a lot of time and effort (and thus money), and should thus not result in descriptions sitting on a shelf gathering dust.

This paper wants to propose a way to do enterprise architecture. Basically, the paper can be split into two parts based on what constitutes the 'enterprise' in the enterprise

F. G. Goethals · J. Vandenbulcke
SAP-leerstoel Extended Enterprise Infrastructures,
Naamsestraat 69, B-3000 Leuven, Belgium

F. G. Goethals (✉)· M. Snoeck · W. Lemahieu · J. Vandenbulcke
Department of Applied Economics-Katholieke Universiteit Leuven, Naamsestraat 69, B-3000 Leuven, Belgium
e-mail: {Frank.Goethals, Wilfried.Lemahieu, Monique.Snoeck, Jacques.Vandenbulcke}@econ.kuleuven.ac.be

architecture effort. First we investigate the practice of doing enterprise architecture in a classic, isolated enterprise. Next (in Section 5) we look at the practice of *Extended Enterprise* Architecture. For both domains of application we investigate why doing enterprise architecture is important or even vital, and how companies can go about doing it. Therefore, we give a high-level theoretical overview of what it means to architect an (extended) enterprise, and infer from this some basic requirements for the enterprise architecture practice. Fundamentally, we offer the building blocks of a high-level roadmap to the IT-enabled (extended) enterprise. The building blocks are defined by the Framework for the Architectural Development of the (Extended) Enterprise, the FAD(E)E. We assume that a number of architecture descriptions are being made, used as input, and updated when implementing the roadmap. To illustrate our ideas we often refer to the architecture artefacts described by the Zachman framework, as this is still the most famous architecture framework, and the framework that is most supported by tools.

2. Basic requirements for an enterprise

There are a number of requirements companies should fulfil in order to be successful. In what follows we discuss three basic ones, namely the need to be integrated, aligned, and agile. It will become apparent that Information and Communication Technology (ICT) has hampered, rather than eased, the fulfilment of these requirements. In Section 3 we will discuss how enterprise architecture can help companies out on this issue.

Firstly, enterprises need to be *aligned* and *integrated*. This means that the departments within an enterprise should work in harmony. As companies are operating in complex environments, enterprises need to be segmented in units, each of which has as its major task the problem of dealing with a part of the environment (Hatch, 1997). Each of the departments within a company does part of the total work that needs to be performed by the company in order to achieve the goals of the enterprise. Of course, all of the departments are expected to work together. Unfortunately, the problem of business-ICT alignment has challenged companies for many years (see Henderson and Venkatraman, 1993; Maes, 1999). Business people often don't understand why they need ICT, how they ought to use these systems, which systems might be useful for them, etcetera. Also, the other way around, ICT people often do not know the information system requirements of business people, and love to spend time playing with new technologies rather than constructing software that is attractive for business people (Cook, 1996). Still, if a company wants to be effective and efficient, the different departments of the enterprise should be aligned and integrated. Lawrence and Lorsch (1970) talk about the importance of (differentiation and) integration within enterprises. They found that a good integration is needed to be successful. Integration is defined as: '*the quality of the state of collaboration that exists among departments that are required to achieve unity of effort by the demands of the environment*' (Lawrence and Lorsch, 1970, p. 11). ICT can play a big role in achieving the necessary integration. After all, integration across departments is achieved by sending information between departments, a task for which ICT is highly appropriate. Unfortunately, in the past, computer applications have often been developed to function as standalone applications. These applications were paid by and built on behalf of departments and were tuned to their requirements. Each system had its own presentation layer, business processing logic, and database and formed as such a 'stovepipe' (Britton, 2001). These stovepipes, thus, do not share data or logic. That is, they are disintegrated. Nowadays, companies are integrating their systems. This practice is generally referred to as 'Enterprise Application Integration' (EAI). Please note that—in order to realize an integrated ICT infrastructure—information systems development should be done from an enterprise-wide point of view.

Furthermore, companies ought to be *agile*. The requirements of the environment (customers, suppliers, partners, etcetera) are changing and companies should be co-evolving with their environment. Unfortunately, people (employees) usually do not like change. Statements such as "Why change if it was working just fine before?" are typical. One often hears such statements in automation projects. Such automation projects not only suffer from inflexible people, they also suffer from inflexible ICT systems. If business people do want to change their business processes, the supporting ICT clearly needs changes as well. Disappointingly, changing existing ICT systems is a very hard job (Zachman, 2004). One of the basic problems in changing ICT systems is that companies often do not know what they have got: which applications do support which processes?, which data are used by which applications?, what does a specific data field in a database mean?, etcetera.
Clearly, realizing an enterprise (with an information system) that has these characteristics is not a sinecure.

3. The basic solution: Enterprise architecture

We believe that alignment, integration, and agility can only be achieved if the enterprise practices enterprise architecture (where needed, see below). However, before we can give a ground for this statement, we need to explain what we mean when we use the term 'enterprise architecture'.

Through the years, the idea behind doing architecture has evolved, producing the IEEE 1471–2000 standard on 'Recommended Practice for Architectural Description of Software-Intensive Systems' (Maier, 2003). IEEE

1471–2000 defines ‘an architecture’ as *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*. An ‘architectural description’ is defined as *a collection of products to document an architecture* (Maier, 2003). While we are convinced about the accurateness of these definitions, many authors do not use these terms in this sense. In an overwhelming part of the literature the word ‘architecture’ is used when ‘architectural description’ is meant. We will use these terms with the meaning as defined by IEEE 1471.

Describing a software-intensive system is not a straightforward task. Before one can start drawing up architectural descriptions (ADs), the scope of the architecture activity should be determined. TOGAF (The Open Group Architecture Framework, (TOGAF, 2002)) mentions four dimensions on which the scope may be defined and limited:

– The *enterprise* scope, i.e., what is the ‘enterprise’ that will be described? This question is very important because the definition of what constitutes the enterprise determines the scope of the systems that need to be integrated. In the past, systems were often not developed for the total enterprise, but for departments. That is, the department equalled the enterprise.
– The level of *detail*. The description should be detailed enough to be useful, but too much detail makes the descriptions confusing and the process of detailing costs money and time.
– The *time* horizon, i.e., is the effort aimed at the description of the AS-IS situation, or is it meant to describe a TO-BE-architecture (or some transitional architectures)?
– The architecture *domains*. This has to do with the idea of viewpoints. IEEE 1471-2000 defines a ‘view’ as *a description of the entire system from the perspective of a set of related concerns*. As such, *a view is composed of one or more models*. A ‘viewpoint’ can be regarded as *a standard or template for constructing a view*, it says *where you look from*. The term ‘viewpoints’ attracted attention in 1996 in the Reference Model of Open Distributed Processing (RM-ODP, see (Farooqui, Logrippo and de Meer, 2004)).[1] The underlying idea was that for each system a number of roles could be identified that have an interest in the system. While each role (e.g. user, analyst, or programmer) is interested in the same system, their relative views of the system are different, and they have different requirements. Therefore, it seems interesting to describe the system from different viewpoints, each of which is chosen to reflect a set of concerns. Now, the question is which viewpoints are relevant, i.e., from which points of view the enterprise architecture should be described. Architecture descriptions are not a goal as such, but are a means to realise other goals. The viewpoint selection should be driven by the knowledge of what the descriptions will be used for when ready (see Maier, 2003; TOGAF, 2002). Consequently, before arbitrarily drawing up an architectural description, one should determine what the description will be used for.

In the past many enterprise architecture frameworks (see e.g. Zachman, 1987; Farooqui, Logrippo and de Meer 2004; Kruchten, 1995; Soni, Nord and Hofmeister, 1995; Tapscott and Caston, 1993; OMG, 2001; The Chief Information Officers Council, 1999; Department of Defense, 2004; Department of the Treasury, 2004) have been proposed, all presenting a number of viewpoints. These frameworks are not appropriate for just any setting. Organizations should investigate whether some framework is appropriate for the intended purposes, before investing time, money, effort and goodwill in the enterprise architecture practice. It is, however, often not clear in which case which framework is most appropriate (van de Heuvel, 2002). Please note that one should not only assess the different architecture description models the frameworks describe/prescribe, but also the relationships between the models. The Zachman framework for example relies on the concept of ‘primitives’, meaning that different models are made for (1) data, (2) processes, (3) locations, (4) people, (5) timing, and (6) motivations. Describing an architecture with these primitives results in models that for example only show the data, but do not show which processes use the data, who is interested in the data, and so on. A model that shows combinations of primitives, for example a model showing who is interested in what data, is called a composite. Tools such as Metis (http://www.metis.no/) allow the visualisation of composites, meaning that one can create a view showing which processes use which data, who is taking part in the processes, and so on. Other tools (such as ArchiMate (Leeuwen, Doest and Lankhorst, 2004)) can for example be used to link business process models to models showing the applications present in the company.[2] Tools which support the architecture frameworks have thus a very important role to play in an enterprise architecture effort. Without tools, one could not dynamically create views that combine information from different descriptions. Tools make it possible to manage the models, and to get value from the models by creating interesting views. Moreover, they may create attractive views. The ArchiMate tool for example allows switching pictographs so as to create models that are

[1] Other authors (as Zachman for example, see below) discussed the *concept* of viewpoints earlier, but did not use the term ‘viewpoint’. More recently, the term viewpoint is also talked about in the Model Driven Architecture (MDA) (Frankel and Parodi, 2002).

[2] Actually, the ArchiMate tool can be used to create many views based on concerns of managers and other stakeholders.

attractive for business managers. For an overview of enterprise architecture tools and the frameworks they support, see www.enterprise-architecture.info/EA_Tools.htm.

As stated earlier, doing enterprise architecture is a means to fulfil the requirements identified in Section 2 of this paper. First, Business-ICT *alignment* is one of the major points of interest of enterprise architecture. Alignment is pursued by making descriptions of the enterprise from the point of view of all who are involved in the realization of the enterprise. Zachman's Information Systems Architecture Framework, for example, distinguishes five perspectives, namely planner, owner, designer, builder and subcontractor. The realization of the relationships between the perspectives results in Business-ICT alignment. It is interesting to relate this idea to the concept of Commercial-Off-The-Shelf (COTS) software. Business software packages (such as those offered by SAP) have been built for specific types of businesses. This means that, underlying the ICT models are models of a hypothetic business to which the ICT models are aligned. If a company considers buying such a software package, she should try to find out how well her existing business model fits the hypothetic business model underlying the software package.
The second useful characteristic of enterprise architecture is to be found in its inherent attention for integration. *Integration* can only be achieved if an enterprise-wide point of view is taken. Clearly, one cannot achieve cross-component integration if no overview is available of the relationships between the different components (be it IT systems, departments, business processes, etc.). Enterprise-wide architecture descriptions are thus very powerful.

Thirdly, the availability of architecture descriptions enhances *agility*. After all, it is easier to change something you know well, and architecture descriptions are of major help in getting to know and understand existing systems. Such descriptions help in handling complexity, as they make abstraction of the issues that are not relevant (that is, if you make the right abstractions). The underlying idea is that decision makers can see all and only the relevant information in the models so as to enable informed decision making, for example concerning the adaptation of a business process. An enterprise architecture is a living thing. Organizations are constantly evolving and their architecture is thus constantly changing. To a big extent, managing the enterprise means managing the enterprise architecture. Through all kinds of implementation projects organizations are moving from an existing enterprise architecture (the AS-IS architecture) to a planned TO-BE architecture. By making issues explicit in a TO-BE architecture and discussing otherwise hidden assumptions upfront, many problems can be avoided. At this moment we want to tackle one of the most ventilated criticisms against enterprise architecture (see e.g. Appleton, 2004), namely the fact that enterprise architecture would be too deterministic, neglecting the autonomy of humans, and lowering flexibility. The fact is, however, that it is not enterprise architecture that is deterministic, it are the information systems that are deterministic. If people want to behave in a nondeterministic way, they cannot expect information systems to fully support their practices. The only things that can be supported are the deterministic ones.

As an illustration, researching for a PhD is a very creative job, and the process of getting a PhD is unique for every student. One can hardly call this a managed process. There are, however, a few stable concepts related to PhDs. Every student needs for example a research proposal, published papers, a dissertation, conference presentations, etc. The status of a PhD can be followed up by looking at the realizations of a student with respect to the stable concepts. An ICT system can be built to support this practice. However, no ICT system is available that guides a PhD-student through a workflow, at the end of which the PhD is a reality. Such an ICT system would be way too restricting. The architecture of a university would thus not include the description of a PhD-process; it would only mention the generic, stable issues.

Processes which are stable, for example the process of ordering a computer, could easily be supported by an automated workflow without taking away sorely needed freedom. If the process changes anyway after a year, architectural descriptions will be very welcome to show which applications need to be changed in order to follow the changes in the business process. Note that we are not saying that only the (business) issues that are being automated need to be architected. Rather, we argue that all issues that need to be managed (and which might be automated) need to be architected. The only thing ICT can do is support stable (subparts of) processes. At the moment your process becomes too unstable (i.e. unmanageable), it does not make any sense anymore to make descriptions of the process and to restrict your employees that way. If there is no reason to manage something (or even a reason *not* to manage something), than you — obviously — do not need to architect that something; you can leave your employees the required freedom.

Now that we have shown what enterprise architecture should take care of, we can look for a way to go about architecting the enterprise.

4. The framework for the architectural development of the enterprise (FADE)

Enterprises have a starting point and an endpoint (in time) and go through several phases throughout their life. The GERAM (Generalised Enterprise Reference Architecture and Methodology, (IFIP-IFAC Task Force, 2004)) presents a general model of the life cycle of an entity. Such a life cycle encompasses all activities from inception to decommissioning of the entity. These activities are categorized into "life

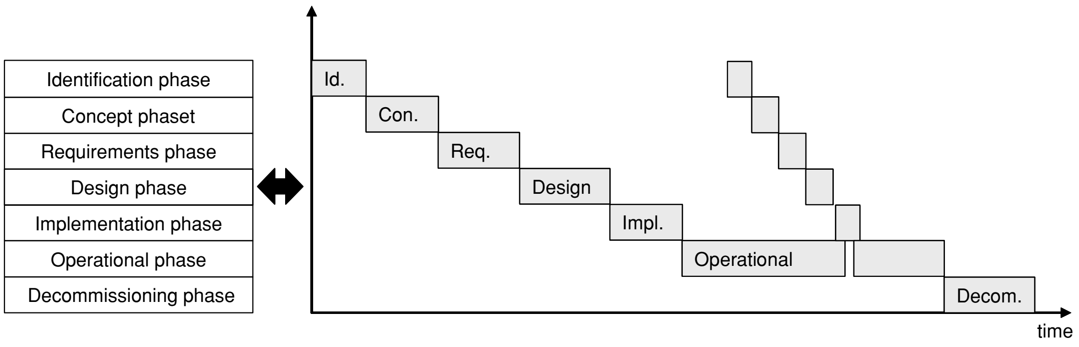

Fig. 1 The seven life cycle phases (*left*) and the life history of an enterprise (*right panel*)

cycle phases". A life cycle basically contains seven life cycle phases.

These are shown in the left panel of Fig. 1. The basic phases may be subdivided further. The design activities are for example often subdivided in two lower-level types of activities, resulting in a preliminary design phase and a detailed design phase.

It is important to note that life cycle phases do not imply a temporal aspect. Some of the processes may be performed repeatedly and in different succession and some may not occur at all during the existence of the entity. The concept of "life history" is used to take into account time and succession. The life history is seen as *the actual sequence of steps a system has gone (or will go) through during its lifetime*; the life cycle is defined as *the finite set of generic phases and steps a system may go through over its entire life history* (ISO/TC184/SC5/WG1. ISO/IS 15704, 2000, p 9; Bernus, Nemes and Schmidt, 2003, p. 81). The concept of life history is illustrated in the right panel of Fig. 1. Companies can be involved in several types of activities at the same time. For example, companies may redesign processes while they are in the operational phase.

Throughout the life history of an enterprise business processes may be redesigned, new information systems may be implemented, etcetera. The life cycle processes or activities may involve business related issues and ICT related issues. It is clear that a fit is desired between decisions made at the business side and those made at the ICT side.

For the purpose of our research we make a distinction between two groups of persons within an organization: (1) Business people and (2) ICT people. Both groups are involved in designing the enterprise. However, they have a different point of view on the enterprise. Therefore, we group the enterprise life cycle activities according to who is making decisions. Furthermore, orthogonally on this classification, we notice that people may be involved (1) in the *execution* of operations, or (2) in the *management* of operations. Typically three levels of management are discerned, namely strategic, tactical and operational (see e.g. Proper et al., 2001). All of these activities are related to the enterprise life cycle phases. The so created framework ('the FADE,' the Framework for the Architectural Development of the Enterprise) is illustrated in Fig. 2.

We believe that the life of an enterprise starts with the strategic identification phase, and ends with the strategic decommissioning phase. Therefore, if we speak about the enterprise life cycle, we mean the phases at the strategic level. At strategic level the desired future of the enterprise is puzzled out. Decisions made at strategic level typically cover a long time horizon of about five years. The mission, the vision, important principles, etcetera are defined. Clearly decisions at this level are not detailed at all, they are vague by nature. A more detailed picture of the enterprise life cycle can be created by focusing on the implementation phase of the strategic level. This implementation phase actually involves the creation of more concrete plans at tactical level. At the tactical level a planning is made to structure the different projects that will be executed during the following one or two years. It is important to note that these projects are placed and fitted within the total enterprise architecture. Therefore, this level is fundamental for companies to achieve an *integrated* enterprise. Again, the implementation phase at this (tactical) level involves life cycle phases at a lower level, namely the operational level. At operational level a detailed planning is made to execute the projects that were planned. Next the project is executed, i.e. the software is created, people are being trained, etcetera. Once the implementation phase (at operational level) has ended (e.g., the software has been programmed and installed) the operational phase is entered. By entering the operational phase at operational level the operational phase is also started at tactical level and at strategic level. At this phase the enterprise is up and running. After some time the created entity may be decommissioned. This may result in entering the decommissioning phase at tactical level and at strategic level.

We notice the presence of the three management levels both at the business and the ICT side. Of course, the life cycle phases of business and ICT need to be intertwined if alignment is desired. Business and ICT should be co-evolving. By linking the management levels with the practice

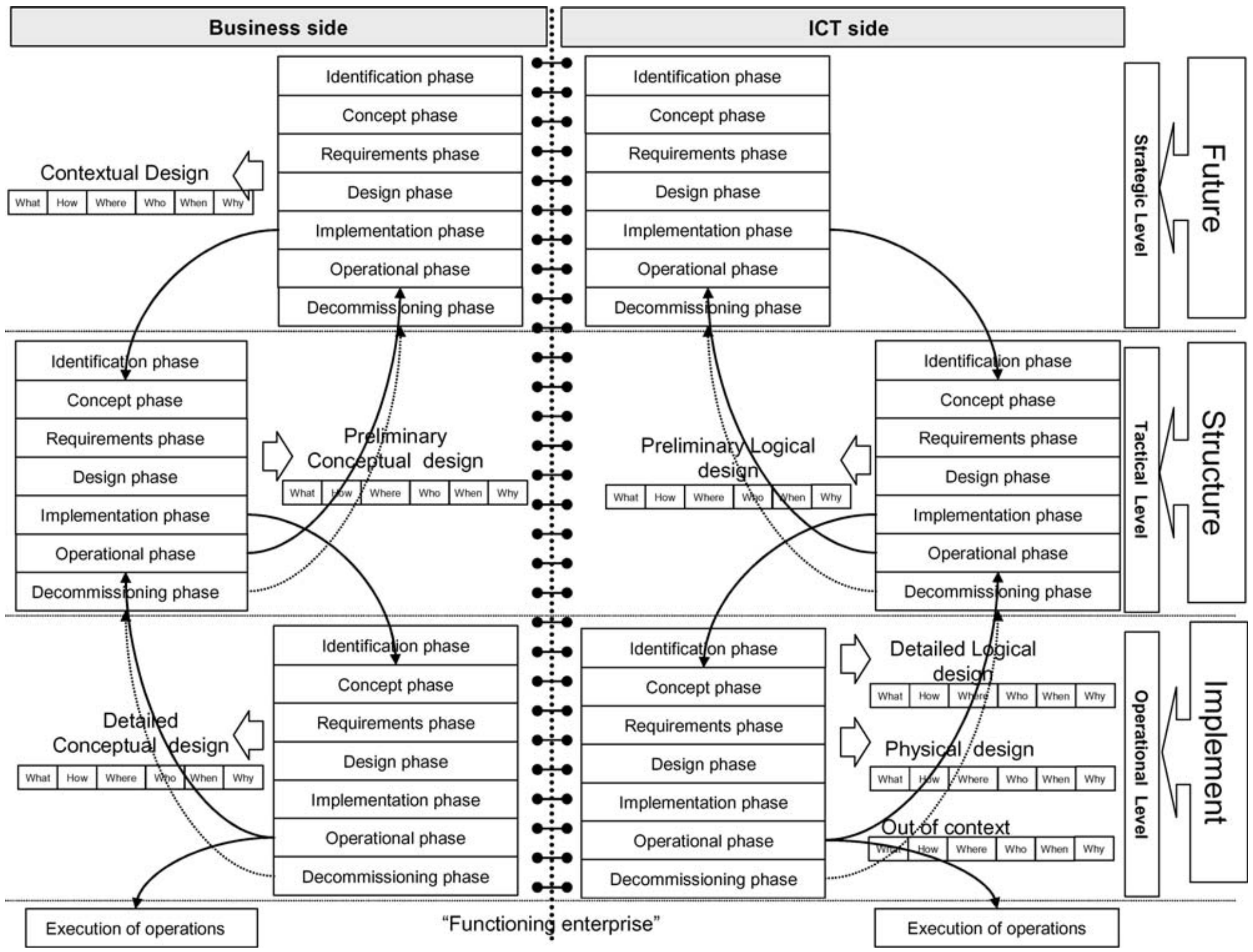

Fig. 2 Linking management levels, life cycle phases, and Zachman views in the FADE

of architecting the enterprise (and its ICT systems) it is clear how alignment should be taken care of. In all of the life cycle phases, alignment should be kept in mind. This can be done by having information communicated across life cycle phases. For example, concepts that are identified at the Strategic ICT level (such as the Web services oriented architecture) may be an interesting input factor for the Business strategy.

It is important to note that the ideas presented above do not represent a life history of an enterprise. Rather, it is shown how the life cycle concept is applied recursively. This is also the reason why Fig. 2 shows life cycles instead of life histories. By relating their current practices to Fig. 2, organizations can see their own life history. Managing this life history (i.e., setting up a roadmap) is very important. It is for example clear that companies that are simply walking top-down through this figure are missing something: giving feedback to previous phases is important (and may result in restarting those phases).

We did not (and will not) propose a life history of an enterprise. That is because we do not believe there is one best way to develop enterprises in general, as companies have different cultures, personnel, skills, principles, environments, etcetera. Every organization needs an approach that fits her situation. Also, organizations cannot just move from one way of working to another in a blink of an eye. They need intermediate (managed) ways of working (Goethals, Vandenbulcke, Lemahieu and Snoeck, 2004a). The FADE is a management framework: a collection of generic constructs that need to be taken in consideration when developing a roadmap to the IT-enabled Enterprise. It should be noted that it is not important for our ideas to be applicable that a company has established so-called "strategic," "tactical," and "operational" levels. It is, however, fundamental that companies live by the core meaning of these levels. That is, companies should have a long-term vision concerning the future of the enterprise. Of course, this vision is usually vague since the future is very uncertain. However, for a medium long period the future is more predictable, and more concrete plans can be made. At this level, organizations should establish some structure. In the short term, when things become clearer, organizations should implement the issues that were included in the plans for the medium range.

At this moment it is interesting to come back to the idea of creating stovepipes (mentioned in Section 2). In fact, to come to a perfectly integrated ICT system it is necessary to have a picture of all ICT systems that need to be created, and to split this picture into manageable autonomous components (as in Component Based Development). Decisions concerning the components that need to be created are decisions that are taken at tactical level. If wrong decisions are being made, as in the past, stovepipes will be created that lead to an integration mess later on. Componentization decisions should take as much as possible an image of the total enterprise into consideration, in order to find out where data and logic need to be shared. Having descriptions of the AS-IS situation as an input clearly helps in taking such decisions. At tactical level these descriptions are detailed enough (and yet not too detailed) to make a componentization decision. At operational level, more detail is added to every component.

Clearly, decisions made at one level form restrictions that should be respected at another level.[3] Programmers and the like may not like being restricted, but it does not make any sense to let them neglect hard constraints. It is illogical to let people work in a cocoon, without any coordination from outside (note that this is still true with the distributed computing paradigm, see Cook, 1996). The enterprise architecture way of working may not be very lightweight, but communicating existing restrictions is the only way to achieve integration and alignment. Architecture descriptions (related to the FADE, see below) should offer the users of the descriptions only and all the relevant information needed for them to make informed decisions.

Please note that our findings dovetail with the ideas of Maes (1999) concerning the vertical extension of the classic Strategic Alignment Model of Henderson and Venkatraman (1993). While the latter focus on the strategy and the operations, Maes stresses the importance of a 'structure' level in addition to the strategic and operational levels. We call this level the "tactical" level. Furthermore, the ideas presented here can easily be related to the Zachman framework (Zachman, 1987). More specifically, the management processes could go hand in hand with the creation of the models in the Zachman framework:

- Zachman's contextual models are created at the Strategic Business level.
- The contextual models serve as a constraint when creating Zachman's conceptual models. These are created in two phases: a preliminary conceptual design is made at the Tactical Business level, and a detailed conceptual design is created at the Operational Business level.[4]
- Zachman's conceptual models form the basis for Zachman's logical design models. Zachman's logical design models include those elements that are/will be computerized. These logical models are made in two steps; a preliminary logical design is constructed at the Tactical ICT level, and a detailed logical design is made at the Operational ICT level.
- Zachman's physical models are created at the Operational ICT level.
- The out-of-context models are created at the Operational ICT level as well.
- Finally, the functioning enterprise is found at the level of the execution of operations.

It is remarkable that the ICT strategy is not straightforwardly documented in the Zachman framework. That is so because the highest rows in the Zachman framework ought to describe the 'business' (although one may also use the framework—more artificially—to document the ICT strategy). Furthermore, it is interesting to see that Zachman does not consider the difference between 'preliminary' and 'detailed' designs. While he uses the concept of 'slivers' to select parts of cells, these are not considered to be 'natural' and Zachman would rather avoid their usage. Our framework recognizes that different viewpoints are needed on design in order to make different types of decisions, namely long term integration decisions and short term implementation decisions.

The six columns of the Zachman framework, which show six primitive English questions (what, how, where, who, when, and why), should be considered in each cell of our framework. Zachman argues that by building the 'primitive' models that answer the 6 questions and by documenting the relationship between the primitives, the enterprise is described entirely. Please note that John Zachman did not propose any methodology for realizing the models in the Zachman framework. By coupling his framework to the management levels, things become more evident. It makes it possible to actively manage and intervene into the enterprise architecture process. More specifically, in terms of the models needed, managing an enterprise actually requires 2 types of models:

(A) the models describing the current AS-IS situation, and
(B) the models describing the desired TO-BE situation.

When applying the FADE, the AS-IS models serve as an input to build the TO-BE models. For example, the existing logical design models can be used as a basis to identify the

[3] If possible. Else, the restrictions will need to be adapted.

[4] Please note that Zachman argues in favour of modelling 'at an excruciating level of detail'.

desired TO-BE logical design. The difference between both (the path of change) will have to be realized at a lower level (i.e., the Delta is an input for another level). Please note that the state of a model changes during the life cycle of an organization. For example, at a specific moment in time, a company has an AS-IS logical design. By walking through the design phase at Tactical ICT level, the TO-BE logical design is developed. This TO-BE model *becomes* the AS-IS model when the model has been implemented at operational level.

5. Architecture in the extended enterprise

Nowadays, companies are not only trying to optimize their internal processes, they also want to optimize (and automate) their relationships with other companies. Fortunately, the FADE is not only useful as a framework to architect the classic enterprise; it is — by nature — extensible so that it can be applied in the context of the Extended Enterprise as well. In this case one would talk about the FADEE, the Framework for the Architectural Development of the Extended Enterprise. In what follows, we first discuss the concept of "the Extended Enterprise." Next, we shortly investigate why doing enterprise architecture is important in an Extended Enterprise setting, and finally, we discuss the FADEE, an extension to the FADE.

5.1. Extended enterprise integration vs. market B2Bi

From organization theory one can conclude that there exist two basic forms of Business-to-Business integration (B2Bi), namely Extended Enterprise integration and Market B2Bi (Goethals, Vandenbulcke, Lemahieu, Snoeck and Cumps, 2005). In the context of *Extended Enterprise integration* (EEi) companies that dispose of capabilities that are interesting for one another try to cooperate/collaborate. As such, these companies pursue a long term relationship with each other. The Extended Enterprise can then be defined as "*a collection of* legal entities ($N \geq 2$) that pursue repeated, enduring exchange relations with one another" (Goethals, Vandenbulcke and Lemahieu, 2004). It is important to note that such partnering organizations have already 'chosen for each other,' i.e., they know the other company can deliver to a certain extent what is needed, and a partnership is set up to get more out of the other company than what is already being delivered. It is recognized that some form of coordination is necessary to realize additional benefits. Partnering enterprises need to find out how they can be of more value to each other. The development of attractive software is part of this value adding effort. Partner-specific IT investments can be made.

Essentially, this is not the case in *Market B2Bi*. Companies that do business in the marketplace do not cooperate/collaborate and have no long term relationship. Basically, for each individual transaction they try to find out who can deliver what is needed. In this situation, a company has for example the free choice to choose the Web services from any company (present in the marketplace) that fulfils her needs. During development there is no thorough coordination among the companies, as every company can freely choose whose services she will use. Market Web services are typically standard Web services (Goethals, Vandenbulcke, Lemahieu, Snoeck and Cumps, 2005).

In the remainder of this paper we focus on the case of Extended Enterprise integration. Here, the companies that make up the Extended Enterprise want to coordinate the development of B2Bi systems. Although an Extended Enterprise is usually not a real enterprise from a legal (financial) point of view, it is noticeable that an Extended Enterprise conceptually forms a new enterprise (the constituting organizations share and redesign processes, data, etcetera). This new enterprise has a starting point and an endpoint (in time). Actually, it seems reasonable to believe that in most cases the life history of an Extended Enterprise is encompassed by the life histories of its constituting enterprises. Generally, the Extended Enterprise will be created while its (individual) constituting enterprises are in operation. Of course, setting up an Extended Enterprise may result in new requirements for the individual enterprises, bringing these into the requirements phase. Naturally it is desirable to keep the individual enterprise operational while redesigning the enterprise to fulfil the new requirements. Therefore individual enterprises will be involved in two types of activities at the same time (e.g., the requirements phase and the operational phase). The life histories of an Extended Enterprise and of one of its constituting companies are shown in Fig. 3. A hypothetical relationship between both life histories can be seen.

5.2. Why do extended enterprise architecture

One complicating factor in developing B2Bi systems in an Extended Enterprise (EE) context concerns the communication about functional and non-functional requirements (Goethals et al., 2004), something that can hardly be automated (at this moment at least) with semantic markup and the like. The only way out is to give people an incentive to communicate and to support their communication, easing, improving, and speeding up the negotiations between companies.

Architecture descriptions are useful as a basis for communication, which yields advantages for diverse reasons (Goethals et al., 2004):

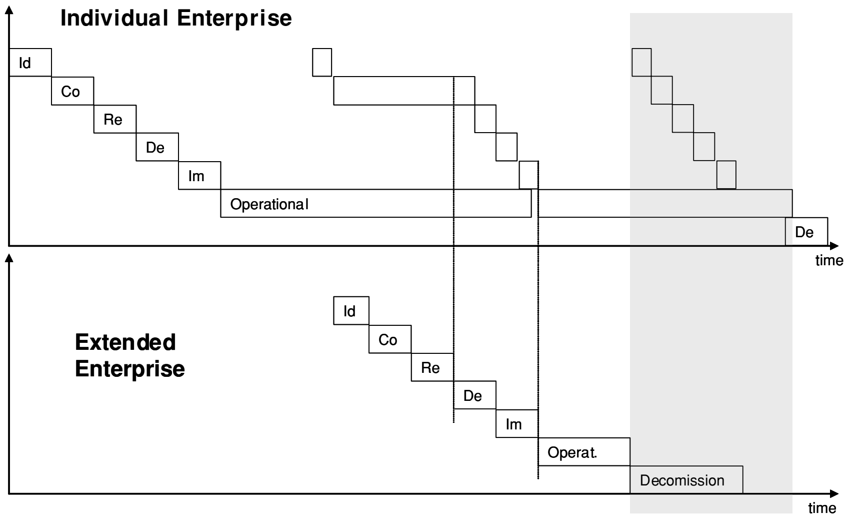

Fig. 3 Life history of an individual enterprise and an Extended Enterprise

- Understanding the organization of the other party is quite a difficult, though important task. By understanding other parties, new practices, procedures and opportunities can be revealed. This, however, requires someone who handles the complexity and oversees the total domain (at an appropriate level of abstraction). Architecture descriptions are a good means to handle such complexity by making interesting abstractions. Above this, architecture descriptions can serve as the basis for a brainstorming-session.
- Service Level Agreements (SLAs) could be negotiated on the basis of the architecture descriptions. After all, formulating SLAs also requires a translation of business requirements into technical requirements and technical measures. Note that internal SLAs are often deployed in order to manage the expectations of service users (see e.g. Koch, 2003). People all too often expect too much from IT, and this may also be the painful truth in an Extended Enterprise.
- Architecture descriptions can be used to inform, guide and constrain decisions, especially those related to IT investments. Architecture descriptions can be a facilitator for realizing B2Bi, as they ease the adaptation of the architecture. After all, it is easier to manage something you know well. An architecture description contains much valuable information for making decisions on investments and for systems development. It is good practice to evaluate the proposed architecture before getting into development (Clements, Kazman and Klein, 2002). By making issues explicit in a description, problems can be detected early on. One should not be making implicit assumptions about functionality, especially not in the global economy where customs may differ from partner to partner! Note that it is still very hard to test and validate choreographies of Web services. By discussing difficult issues upfront, many problems can be avoided.
- Furthermore, in a more futuristic vision, architecture descriptions of the systems could be made accessible to software agents, so they could find and understand the services a company is offering. Also, architecture descriptions might be made executable. The latter is a topic a number of software tools are paying ample attention to at this moment. The ARIS tool is for example being integrated in the SAP Net Weaver platform for this purpose (IDS Scheer, 2004).

5.3. The framework for the architectural development of the extended enterprise (fadee)

In an Extended Enterprise context companies want to coordinate the development of B2B systems. It is for example unrealistic to assume the IT personnel of one company automatically know which Web services the counterparties need (they may not even have a notion of what their business is about). If no active coordination is happening, the right Web services will only be developed by chance. Even more importantly, not only the ICT side of the Extended Enterprise has to be developed, the business processes may be modelled and redesigned as well (Clark and Stoddard, 1996). Enterprise architecture is thus also needed at the level of the Extended Enterprise. Fortunately, the FADE is—by nature—extensible. This results in the FADEE, the Framework for the Architectural Development of the Extended Enterprise (shown in Fig. 4). The FADEE, just like the FADE, offers the building blocks to create a roadmap to the IT-enabled (Extended) Enterprise. Making the software process explicit and managing this software process is very important for the successful development of ICT in an Extended Enterprise.

As is clear from Fig. 4, Extended Enterprise integration can concern different levels of integration (see Goethals, Vandenbulcke, Lemahieu and Snoeck, 2004b) for a more

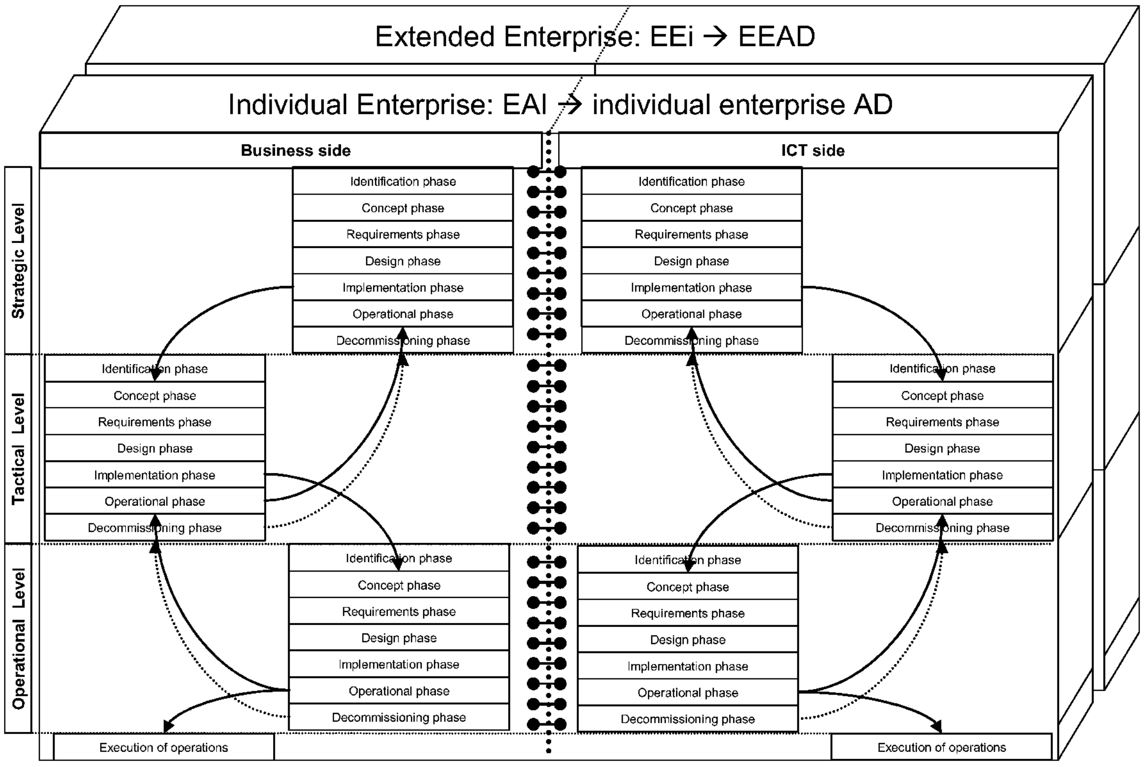

Fig. 4 Illustration of the FADEE

detailed discussion on this). In some cases a long term strategy is developed for the Extended Enterprise, and the individual enterprises that are part of the Extended Enterprise may have to adapt their strategy to that of the Extended Enterprise. Clearly, this asks for radical changes. Individual organizations need to be restructured, have to focus on their core competences, etcetera. In other cases, the Extended Enterprise will not have a new strategy of her own; the enterprises then try to realize benefits at a lower level. Tactical decisions made at the level of the Extended Enterprise are meant to ensure that an *integrated* Extended Enterprise is created. Different B2B processes may share logic and data. Hence, structuring all processes and data within the picture of the total Extended Enterprise is important. Integration at an operational level concerns the automation of individual processes (e.g., the ordering process). Decisions at all three levels could be made explicit in architectural descriptions (ADs).

In terms of the models, managing an Extended Enterprise requires 2 times 2 models:

1-A the models describing the AS-IS situation at the level of an individual enterprise,
1-B the models describing the TO-BE situation at the level of an individual enterprise,
2-A the models describing the AS-IS situation at the level of the EE (the "EEAD"), and
2-B the models describing the TO-BE situation at the level of the EE.

Again, when applying the FADEE, the AS-IS models serve as an input to build the TO-BE models. As an example, in the case of operational integration companies could automate existing business processes. Having the models of the AS-IS situation would ease the automation of the processes. Please remember that in the FAD(E)E it is assumed that all (more or less stable) parts of the business are modelled, not only the parts of the business that are being realized in the ICT systems. Clearly, automating the link between existing systems is easier if the link between the businesses they are supporting is clear.

It is (unfortunately) possible that the AS-IS situation has not been made explicit in architecture descriptions. Moreover, when setting up a new Extended Enterprise, it is very likely that the AS-IS relationship between the different organizations has not even been architected, let alone that the relationship would have been documented. It would, however, become even worse if companies are not engineering the TO-BE Extended Enterprise either, as this may result in two enterprises having a different image (in their heads) of the TO-BE situation. That is, the two companies may envision different TO-BE Extended Enterprise architectures. This may result in the impossibility to do business with each other (Snoeck et al., 2004).

Enterprise architecture restricts people, takes away their freedom. Sometimes it just doesn't make any sense to give people too much freedom. This is especially true in the case of the Extended Enterprise. There, companies are sharing a process and they do not want surprises to happen. That is why it is good practice to make all issues explicit in architecture descriptions.

We may again relate the FADEE to the Zachman framework. Doing enterprise architecture at the level of the Extended Enterprise requires the Zachman framework to be applied at two levels. Combining Architecture Descriptions at the level of the individual enterprises, and architecture descriptions at the level of the Extended Enterprise. The "EEAD" (Extended Enterprise Architecture Description) then for example contains all processes which require the invocation of services (at least 1) that have been implemented by other parties. However, *how* these services (the subtasks of the total process) are being realized is not mentioned in the EEAD. The parts of the process that are being realized by the individual enterprise herself are documented in the "individual enterprise AD," the services that are executed by other parties are considered to be black boxes, and no further information concerning these services is noted down.

Applying the Zachman framework at two levels can be considered to be a choice for a hybrid solution, halfway between centralizing all architectural information to the level of the total Extended Enterprise (i.e., making one monolithic description of the *collection* of all constituting companies), and decentralizing all architectural information to the level of the individual enterprises that are part of the Extended Enterprise. There are diverse arguments[5] supporting the choice for such a 'hybrid solution' (i.e., not centralizing nor decentralizing everything):

– Each enterprise has an *enterprise life cycle* (see the GERAM; (IFIP-IFAC Task Force, 2004)). As every enterprise needs to be engineered, it is logical to draw up architecture descriptions for all enterprises. This fits nicely with the ideas presented in Fig. 3.
– Using a Zachman framework at two levels (centralized and decentralized) implies that we actually have two definitions of what constitutes the 'enterprise'. Choosing the scope of the enterprise implies choosing which level of integration can be realized. As the Extended Enterprise is not a legal entity, and companies really want to realize a loose coupling between their systems (via XML Web services for example) it is logical to see the legal entity as the scope of the enterprise. However, if the goal is to integrate systems of different companies, a view is needed from the enterprise where the enterprise is the collection of collaborating companies.
– Why would one try to put all information in a monolithic, centralized architecture description, if only part of this is relevant for each company?

[5] A more detailed motivation can be found in Goethals, Vandenbulcke and Lemahieu (2004).

We refer to Goethals, Vandenbulcke and Lemahieu (2004) for an illustration of the application of the Zachman framework at two levels.

6. Conclusions

This paper has shown that doing enterprise architecture is a must. Without doing architecture, organizations cannot expect to get alignment, integration (things will only fit by chance if you don't engineer), or flexibility (how will you change things in a managed way if you do not know what there is to be changed?). This is of fundamental importance.

Doing enterprise architecture should be part of the normal way of doing business; it should be embedded in the classic management processes organizations know. Organizations want their systems to be implemented fast. That is, there is a focus on the short term. However, this has caused employees to neglect doing architecture, mortgaging the long term. Companies have to balance the short term and the long term by building this balance into their every day way of working. Moreover, enterprise architecture should not only be seen as a job of ICT people. Actually, it is striking to see that it is usually up to the ICT department to start up the enterprise architecture practice, and that they have to fight hard to get commitment from the business people. It is striking, because doing enterprise architecture is in the first place advantageous for the business. If business people do their part of the enterprise architecture job, they can finally get the best out of ICT.

The Framework for the Architectural Development of the (Extended) Enterprise presented in this paper shows how the enterprise architecture process is theoretically meant to look like; and this framework has been related to renowned frameworks such as the one of Zachman and the GERAM framework. Further development of the FAD(E)E and applying it in case studies is the subject of further research.

Enterprise architecture tools are of major importance in turning the enterprise architecture effort into a success, and so are Architecture Markup Languages (AMLs). As this paper has shown, a lot of people are/should be involved in the architecture process, each with their own view on the enterprise. These persons are interested in different information; want to see different kinds of representations. Still they are all considering the same thing: the enterprise. The information they are looking at or changing should therefore be integrated. Fortunately, architectures can be described in appropriate XML-variants (which are all grouped under the name 'Architecture Markup Languages'), such as the Architecture Description Markup Language (ADML). Please note that currently a lot of other descriptive languages are being developed as well. More specifically, in the context of Web services standards such as WSDL, BPEL4WS, BPML, DAML-S, and the like do nothing more than describing services and services choreographies. We hope to see an integration of these efforts with the architecture description efforts and architecture tools in the future.

If mature code generators one day find their way to the market, they can take the architecture descriptions as an input (see for example the AndroMDA project at www.andromda.org). Frankel and Parodi (2002) state that the MDA (Model-Driven Architecture) *sets the stage for automatic generation of at least part of the XML and code, such as Java code, that implements the* [web] *services*. Of course, code generation demands detailed descriptions, while architecture descriptions in general do not need to be that detailed (remember: the measure of detail of the description should reflect the goal of the description). There has been written a lot about code generation, but not much has been realised yet. Nevertheless, it looks like the future will bring improved code generation tools. In our opinion, this is a nice prospect as creativity should show in the architecture of the IT-system, rather than in the code as such.

Acknowledgments This paper has been written as part of the 'SAP-leerstoel'-project on 'Extended Enterprise Infrastructures' sponsored by SAP Belgium.

References

Appleton D. Why enterprise architecture is an oxymoron. *Business Rules Journal* 2004;5(4).

Bernus P, Nemes L, Schmidt G. *Handbook on Enterprise Architecture*, Berlin Heidelberg: Springer-Verlag, 2003, 778.

Britton C. *IT Architectures and Middleware*, Boston: Addison-Wesley, 2001, 284.

Clark T, Stoddard D. Interorganizational business process redesign: merging technological and process innovation. *Journal of Information Management* Fall 1996;13(2):9–28.

Clements P, Kazman R, Klein M. *Evaluating Software Architectures.*, Boston, MA: Addison-Wesley, 2002, p. 302.

Cook M. Building enterprise information architectures.*Upper Saddle River*, NJ: Prentice-Hall, 1996, 179.

Department of the Treasury, Chief Information Officer Council. Treasury Enterprise Architecture Framework Version 1; 164. Available at: URL:http://ustreasury.mondosearch.com. Accessed August 27, 2004.

Department of Defense-C4ISR Architectures Working Group. C4ISR architecture framework Version 2.0. 1997 Dec; 239. Available at: URL:http://www.afcea.org/education/courses/ archfwk2.pdf. Accessed, 2004.

Farooqui K, Logrippo L, de Meer J. The ISO reference model for open distributed processing—an introduction. 1996 Feb. Available at: URL:http://lotos.site.uottawa.ca/ftp/pub/Lotos/TechRep/CNIS-94.pdf. Accessed August 27, 2004.

Frankel D, Parodi J. Using model-driven architecture to develop web services. *IONA Technologies white paper*, 2002.

Goethals F, Vandenbulcke J, Lemahieu W, Snoeck M. A framework for managing concurrent business and ICT development. In: Melnik G, Holz H. eds. *Advances in Learning Software Organizations. Proceedings of the 6th International Workshop on Learning Software Organizations*, Banff, Canada. LNCS 3096, 2004a; 3096:131–136.

Goethals F, Vandenbulcke J, Lemahieu W, Snoeck M, De Backer M, Haesen R. Communication and enterprise architecture in extended enterprise integration. In: *Proceedings of the 6th International Conference on Enterprise Information Systems*, Porto, Portugal. 2004;3:14–17;332–337.

Goethals F, Vandenbulcke J, Lemahieu W. Developing the extended enterprise with the FADEE. In: *Proceedings of the ACM Symposium on Applied Computing*, Nicosia, Cyprus, March 14–17, 2004;1372–1379.

Goethals F, Vandenbulcke J, Lemahieu W, Snoeck M. Structuring the development of inter-organizational systems. In: *Proceedings of the Web Information Systems Engineering Conference*, Brisbane, Australia. Springer LNCS 3306, 2004(b);3306:454–465.

Goethals F, Vandenbulcke J, Lemahieu W, Snoeck M, Cumps B. Two basic types of Business-to-Business integration. *International Journal of E-Business Research* 2005;1(1):1–15.

Hatch MJ. *Organization Theory, Modern Symbolic and Postmodern Perspectives.* Oxford University Press, 1997:379.

Henderson J, Venkatraman N. Strategic alignment: leveraging information technology for transforming organizations. *IBM Systems Journal* 1993;32(1):4–16.

IDS Scheer. ARIS for SAP NetWeaver. Available at: URL:http:// www.ids-scheer.com. Accessed August 18, 2004.

IFIP-IFAC Task Force. GERAM: Generalised Enterprise Reference Architecture and Methodology, Version 1.6.2. 1998 Jun. Available at: URL:http://www.cit.gu.edu.au. Accessed January 20, 2004:29.

ISO/TC184/SC5/WG1. ISO/IS 15704: Industrial automation systems—requirements for enterprise-reference architectures and methodologies, 2000;43.

Koch C. Put IT in writing. CIO Magazine 1998 Nov 15. Available at: URL:http://www.cio.com/archive/111598/sla_content.html. Accessed January 29, 2003.

Kruchten P. The $4 + 1$ view model of architecture. *IEEE Software* 1995;42-50.

Lawrence P, Lorsch J. Organization and environment: managing differentiation and integration. *Homewood*, Il: Irwin-Dorsey 1970, 279.

Leeuwen D van, Doest H ter, Lankhorst MM. A tool integration workbench for enterprise architecture. In: *Proceedings of the 6th International Conference on Enterprise Information Systems*, 14–17; Porto, Portugal. 2004(3): 470-478.

Lipschutz R. A better blueprint for business. *PC Magazine* 2004;23(15):131–137.

Maes R. A generic framework for information management. *PrimaVera Working Paper* 1999;99-03:22.

Maier M. The IEEE 1471-2000 standard—architecture views and viewpoints. Available at: URL:www.opengroup.org/architecture/ togaf/agenda/0107aust/presents/maier_ 1471.pdf. Accessed January 29, 2003.

OMG Architecture Board ORMSC. Model Driven Architecture (MDA). 2001;31.

Proper H, Bosma H, Hoppenbrouwers S, Janssen R. An alignment perspective on architecture-driven information systems engineering.

In: *Proceedings of the Second National Architecture Congres; Amsterdam,* The Netherlands, 2001;11.

Snoeck M, Lemahieu W, Goethals F, Dedene G, Vandenbulcke J. Events as atomic contracts for application integration. *Data and Knowledge Engineering* 2004;51(1):81–107.

Soni D, Nord RL, Hofmeister C. Software architecture in industrial applications. In: R. Jeffrey, D. Notkin eds., *Proceedings of the 17th International Conference on Software Engineering*, ACM Press 1995;196–207.

Tapscott D, Caston A. *The New Promise of Information Technology,* New York: McGraw-Hill, 1993;313.

The Chief Information Officers Council. Federal Enterprise Architecture Framework Version 1.1. 1999;41.

TOGAF. The open group architecture framework (TOGAF) Version 8, Enterprise Edition 2002;303.

van de Heuvel W, Proper E. De pragmatiek van architectuur. *Informatie* 2002;12–16.

Weill P, Broadbent M. Leveraging the new infrastructure. Boston, MA: Harvard Business School Press, 1998;294.

Zachman J. A framework for information systems architecture. *IBM Systems Journal* 1987;26(3):276–292.

Zachman J. Enterprise architecture and legacy systems, getting beyond the “legacy”. Available at: URL:http://members.ozemail.com.au/visible/papers/zachman1.htm. Accessed January 26, 2004.

Frank G. Goethals completed his Master studies in economics (option informatics), at the Katholieke Universiteit Leuven, Belgium, in 2000. He is presently researching for a Ph.D. under the theme of ‘Managing data in the Extended Enterprise’. This research is conducted at the K.U.Leuven under the guidance of professor J. Vandenbulcke, and is financed by SAP Belgium. Frank has a strong interest in coordination and dependency theory and Enterprise Architecture.

Monique Snoeck obtained her Ph.D. in May 1995 from The Department of Computer Science of the Katholieke Universiteit Leuven with a thesis that lays the formal foundations of the object-oriented business modelling method MERODE. Since then she has done further research in the area of formal methods for object-oriented conceptual modelling. She now is Full Professor with the Management Information Systems Group of the Faculty of Economics and Applied Economics at the Katholieke Universiteit Leuven in Belgium. She has been involved in several industrial conceptual modelling projects. Her research interests are object oriented conceptual modelling, software architecture and software quality.

Wilfried Lemahieu holds a Ph.D. from the Department of Applied Economic Sciences of the Katholieke Universiteit Leuven, Belgium (1999). At present, he is associate professor at the Management Informatics research group of the Faculty of Economics and Applied Economics. His teaching includes Database Management, Data Storage Architectures and Management Informatics. His research interests comprise distributed object architectures and web services, object-relational and object-oriented database systems and hypermedia systems.

Jacques A. Vandenbulcke is professor at the Faculty of Economics and Applied Economics of the Katholieke Universiteit Leuven, Belgium. His main research interests are in Database management, Data modelling, and Business Information Systems. He is co-ordinator of the Leuven Institute for Research on Information Systems (LIRIS) and holder of the SAP-chair on ‘Extended enterprise infrastructures’. He is president of ‘Studiecentrum voor Automatische Informatieverwerking (SAI)’, the largest society for computer professionals in Belgium, and co-founder of the ‘Production and Inventory Control Society (PICS)’ in Belgium.