Evolving the Enterprise Engineering Framework

H.A. Proper[1,2] and M. Op 't Land[1,3,4]

e.proper@acm.org, martin.optland@capgemini.com

[1]Capgemini, Utrecht, The Netherlands

[2]Radboud University Nijmegen, Nijmegen, The Netherlands

[3]University of Lisbon, Lisbon, Portugal

[4]Delft University of Technology, Delft, The Netherlands

1 Introduction

Present day enterprises need to be agile to improve their chances of survival. In other words, they need the ability to quickly adapt themselves to changes in their environment, and seize opportunities as they avail themselves. These developments pose fundamental design challenges to present day enterprises, and fuels the need for an architecture based approach to the design and evolution of enterprises [12]. Moreover, it requires a clear engineering framework which relates the essential, *implementation independent*, perspective on the enterprise to its implementation in terms of different forms of technologies and other means (such as organisations and people). In this short paper, we are concerned with such a framework, which combines theoretical insights and experiences from practice.

2 Engineering frameworks

When designing and evolving enterprises (including their IT support) several frameworks [4,5,7,14,17] can be used to better structure the design activities. In particular the process of moving from the motivations of stakeholders, via an essential (*implementation independent*) view on the enterprise, to its implementation using different technologies, can benefit from the use of such a framework to structure the design activities. Inn a services based economy, where networked enterprises become the dominant form of organising, the design of an enterprise is not simply a matter of top-down design. In larger enterprises, in particular networked enterprises, there are bound to be emergent forces that change the enterprise from within. A networked enterprise is not a *machine*, but rather an *organism*. Another important requirement on such an enterprise engineering framework is therefore its ability to deal with a discrepancy between top-down design and bottom-up emergence of new/evolved structures.

In our research activities, as well as our work in industry, we strive to create a design framework that provides us with the ability to define the essence of an enterprise on a level abstracting from implementation details, while at the same time relating this to the chosen implementation. It is our opinion that in order to achieve this, a synthesis between aspects of existing frameworks is required. For example, DEMO [5] focuses on the productional essence of enterprises by providing an implementation independent way of describing the essence of the way the enterprise, as a collective of social individuals, produces its results. At the same time, ArchiMate [8] and the Integrated Architecture Framework (IAF) [4] provide frameworks which link across different levels of implementation and construction abstraction.

Each of these existing frameworks has their own merits. In our work we therefore try to integrate the benefits of these frameworks in an attempt to arrive at a framework which indeed makes a distinction between the essential perspective of an enterprise and several levels of abstractions with regards to its construction and implementation. In [10,11] we have reported on a first version of an enterprise engineering framework, which tried to combine DEMO [5] and some basic aspects of the IAF [4]. In [12] we also investigated the added value of combining value modelling [6] with transaction models from DEMO [5]. In this short note, we report on a further evolution of this framework.

3 Dimensions of the framework

Below, we briefly discuss the three dimensions of the enterprise engineering framework as it currently stands. It is important to realise that the proposed framework is primarily intended as a framework for *reasoning* and *understanding*. It allows designers to link the goals of stakeholders, via an implementation independent view on an enterprise, to its actual implementation. This reasoning can be applied in a top-down style, or can be used to understand the impact of new emergent structures that appear ('spontaneous') in the current implementation of the enterprise.

3.1 Design motivation

The first dimension of the framework is concerned with the motivation of the design of an enterprise in terms of the goals of stakeholders. It essentially follows a similar line of reasoning as advocated by the Business Motivation Model [1]. Within this dimension, we make a distinction between the *motivations* of the stakeholders involved, *directives* to be adhered to by the design, and the actual *design*.
The *motivations* refer to the desires and goals of stakeholders with regards to (the part of) the enterprise being (re)designed. At this level we can use languages such as i* [16] to express the goals of the stakeholders, and underpin the possible (positive or negative) impact which alternative designs of the (future) enterprise may have on these goals.
Based on the goals of the stakeholders, and the possible impact of different designs of the enterprise, the stakeholders may want to formulate *directives* to guard their interests. This is in line with the business motivation model [1], where the impact of *influencers* on the interests of stakeholders may lead to the formulation of directives. In the context of enterprise engineering, these directives take the form of design principles that apply in general, or more specific requirements that apply to specific parts of the enterprise being designed.
The *design* of the enterprise provides the actual specification of the enterprise in terms of value exchanges, the actors involved, transactions between the actors, services, processes, IT support, et cetera. In sum, the *motivations* of stakeholders lead to *directives*, which the *design* must comply to.

3.2 System type

In our view, enterprises are primarily social systems, in which the (atomic) elements are social individuals. The core operating principle of an enterprise is that the constituent social individuals enter into and comply with commitments regarding the products or services to be created or delivered [5]. In doing so, the social individuals communicate, in other words, they exchange information by means of information carriers (such as documents).
As discussed in [5] three types of systems can be discerned that are core to the functioning of an enterprise:

- The *performative* system focusses on the entering into and complying with commitments, by social individuals.
- The *informative* system supports the performative system, and is concerned with information exchanges between the social individuals.
- The *formative* system supports the informative system, and is concerned with the

In the actual implementation of the three system types, the social individuals are typically realised by human beings, who will use technologies or other means to conduct their activities.
DEMO [5] suggests to use Business-organisation, Information-organisation, and Document-organisation respectively as synonyms for the three types of systems. In this paper we refrain from these terms, since business and information typically have other co-notations in other engineering frameworks.

3.3 Levels of construction

In this dimension, we distinguish between the function of a system, its implementation independent construction, and its actual implementation:

- At the *function* level, the (different system types of the) enterprise being engineered is regarded as a *black box*, while describing its (desired) functionality from the perspective of the using system and/or its stakeholders.
- The *ontological construction* level takes a *white box* perspective on the enterprise. It is concerned with the actual construction of the enterprise, showing how it will realise the functionality identified at the function level, while doing so in a way which is completely independent of any implementation details.
- The *implementation construction* level provides a model of how the ontological design will be implemented using socio-technical means, such as people, organisations and technologies. Within this level, we identify two sub-levels:
  - The *infrastructure* level deals with the underlying socio-technological infrastructure as it will be used.
  - The *application* level shows how the socio-technological infrastructure for a specific technology is used in the implementation. In the case of IT, this corresponds to custom-made (or specially configured) software applications. In the case of people, this involves the definition of specific work processes.

  Furthermore, we distinguish between different socio-technical means that are used in the implementation: human, other parties, ICT, machines, et cetera.

4 The enterprise engineering framework

Figure 1 depicts the complete enterprise engineering framework. It also depicts how the distinctions between the different levels and system types fades when moving towards the directives and motivations. Typically, directives such as architecture principles impact on multiple levels and system types [11]. At the level of stakeholder motivations, this will be even more so.
Figure 1 also illustrates that a distinction should be made between the implementation of the enterprise as it exists in reality, and the model of this implementation. In the case of a top-down design process, one will endeavour to implement the enterprise conform its design. However, in reality, the implemented enterprise will continue to evolve, in particular due to the autonomicity of the social individuals. This means that as time moves on, discrepancies may surface between the model of the implementation and the actual implementation. In such cases, either the design (possibly including the ontological construction and function) need to be updated, or the changes in the actual enterprise need to be rectified in accordance to the design.
Existing languages and frameworks such as i* [16], SBVR [13], DEMO [5], e3Value [6], BPMN [3], UML [15] and ArchiMate [8] can be applied in different different cells of the framework:

- *Motivations* – As a goal modelling technique, i* is suitable to express the motivations of stakeholders.
- *Directives* – As a standard for the denotation of business rules, SBVR is quite suitable to express requirements and architecture principles in particular [2].
- *Function* – e3Value can be used to provide a functional model of the value exchanges in which the enterprise will participate, and/or the value web of a networked enterprise.
- *Ontological construction* – DEMO as a language and method is explicitly positioned for the creation of the ontological construction of an enterprise.

  As a language, The Open Group's ArchiMate standard does not make a clear distinction between an ontological perspective and the implementation construction. However, the business layer of ArchiMate *can* be used to represent an implementation independent design of the enterprise. Furthermore, ArchiMate does also not make a very clear distinction between the performative, informative and formative system types. However, with the right methodological discipline, the language can be used.

- *Implementation construction* – ArchiMate's business layer can also be used to represent the implementation of that part of the enterprise which will be conducted by human beings, while its application and technology layer are dedicated to the IT parts of an enterprise's implementation. The OMG's BPMN and UML standards can also be used to model the human processes as well as IT support, albeit at a more detailed level than ArchiMate, which is specifically targeted at an enterprise architecture level.

As also hinted at in [12], it seems beneficial to more tightly integrate these languages. When looking at the architecture of the ArchiMate language [9], then the general idea of using a generic core for open-active systems can be used to integrate these languages in an attempt to completely cover the enterprise engineering framework, while also creating a strong cohesion between the different models.

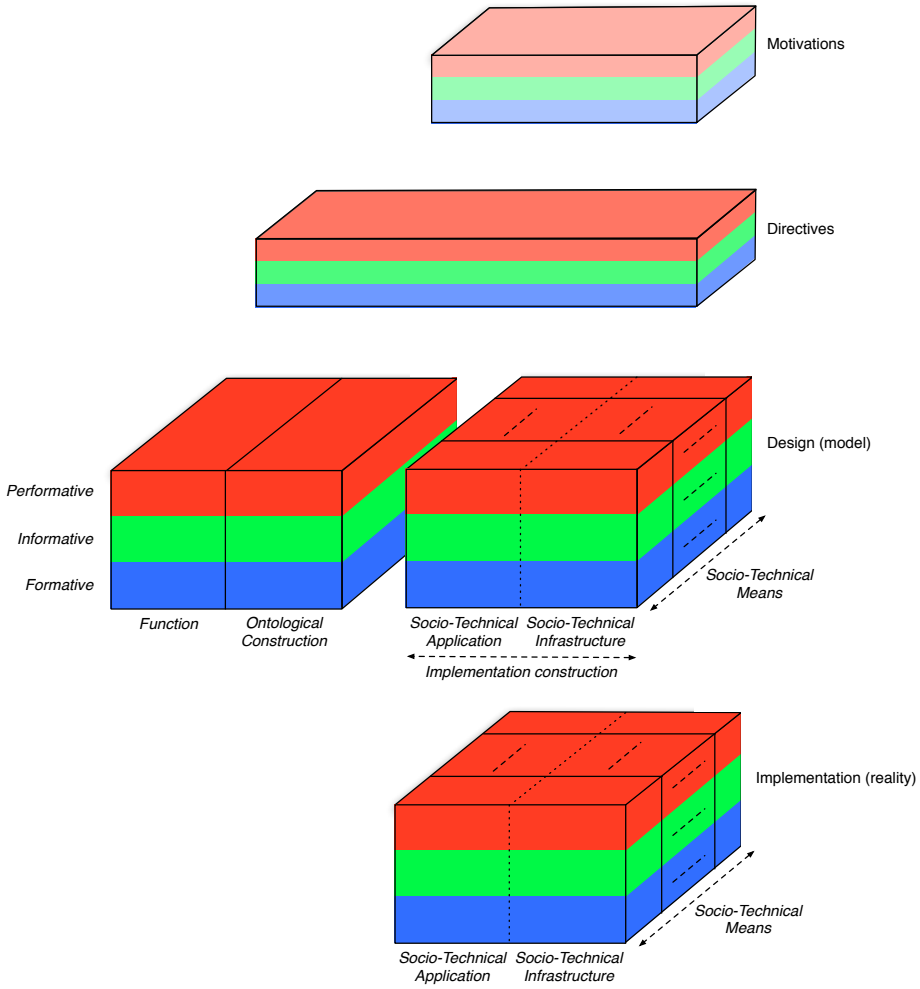

Figure 1: The enterprise engineering framework

5 Conclusion

In this short paper we discussed a new iteration of the enterprise engineering framework. We also showed how existing languages can be used to represent models of different aspects of the framework. We intend to further develop the framework by populating the framework in terms of some existing case studies. The next step will be to indeed better integrate the existing languages to more completely cover the enterprise engineering framework.

References

[1] BMM Team. Business Motivation Model (BMM) Specification. Technical Report dtc/06–08–03, Object Management Group, Needham, Massachusetts, August 2006.

[2] P. van Bommel, S.J.B.A. Hoppenbrouwers, H.A. Proper, and Th.P. van der Weide. Giving meaning to enterprise architectures – Architecture principles with ORM and ORC. In R. Meersman, Z. Tari, and P. Herrero, editors, *On the Move to Meaningful Internet Systems 2006: OTM Workshops*, LNCS, pages 1138–1147, Montpellier, France, October/November 2006. Springer, Berlin, Germany.

[3] Business process modeling notation, v1.1. OMG Available Specification OMG Document Number: formal/2008-01-17, Object Management Group, January 2008.

[4] Capgemini. Enterprise, Business and IT Architecture and the Integrated Architecture Framework. White paper, Utrecht, The Netherlands, 2007.

[5] J.L.G. Dietz. *Enterprise Ontology – Theory and Methodology*. Springer, Berlin, Germany, 2006. ISBN-10: 9783540291695

[6] J. Gordijn and H. Akkermans. Value based requirements engineering: Exploring innovative e-commerce ideas. *Requirements Engineering Journal*, 8(2):114–134, 2003.

[7] M.-E. Iacob, H. Jonkers, M.M. Lankhorst, and H.A. Proper. *ArchiMate 1.0 Specification*. The Open Group, 2009. ISBN-13: 9789087535025

[8] M.M. Lankhorst et al. *Enterprise Architecture at Work: Modelling, Communication and Analysis*. Springer, Berlin, Germany, 2005. ISBN-10: 3540243712

[9] M.M. Lankhorst, H.A. Proper, and H. Jonkers. The Architecture of the ArchiMate Language. In T.A. Halpin, J. Krogstie, S. Nurcan, H.A. Proper, R. Schmidt, P. Soffer, and R. Ukor, editors, *Enterpise, Business-Process and Information Systems Modeling – 10th Intl. Workshop, BPMDS 2009 and 14th Intl. Conference, EMMSAD 2009, held at CAiSE 2009, Amsterdam, The Netherlands, June 2009*, volume 29 of LNBIP, pages 367–380. Springer, Berlin, Germany, 2009. ISBN-13 9783642018619

[10] M. Op 't Land. Principles and architecture frameworks. Technical report, Radboud University Nijmegen, The Netherlands, 2005. Educational material of University-based Master *Architecture in the Digital World*.

[11] M. Op 't Land and H.A. Proper. Impact of Principles on Enterprise Engineering. In H. Österle, J. Schelp, and R Winter, editors, *Proceedings of the 15th European Conference on Information Systems*, pages 1965–1976. University of St. Gallen, St. Gallen, Switzerland, June 2007.

[12] M. Op 't Land, H.A. Proper, M. Waage, J. Cloo, and C. Steghuis. *Enterprise Architecture – Creating Value by Informed Governance*. Springer, Berlin, Germany, 2008. ISBN-13: 9783540852315

[13] SBVR Team. Semantics of Business Vocabulary and Rules (SBVR). Technical Report dtc/06–03–02, Object Management Group, Needham, Massachusetts, March 2006.

[14] *TOGAF Version 9*. The Open Group, Van Haren Publishing, Zaltbommel, The Netherlands, 2009. ISBN-13: 9789087532307

[15] UML 2.0 Superstructure Specification – Final Adopted Specification. Technical Report ptc/03–08–02, OMG, August 2003.

[16] E. Yu and J. Mylopoulos. Using goals, rules, and methods to support reasoning in business process reengineering. *International Journal of Intelligent Systems in Accounting, Finance and Management*, 5(1):1—13, January 1996. Special issue on Artificial Intelligence in Business Process Reengineering.

[17] J.A. Zachman. A framework for information systems architecture. *IBM Systems Journal*, 26(3), 1987.